

A decorative graphic on the left side of the cover consists of several rounded squares of different colors: a large green square at the top right, a yellow square below it, an orange square below that, a blue square to the left of the orange one, a purple square below the blue one, and a teal square to the right of the purple one.

# MapBasic

Версия 9.0

СПРАВОЧНИК

---

Информация содержащаяся в этом документе может быть изменена без уведомления и не представляет собой обязательств со стороны продавца или его представителей. Никакая часть этого документа не может быть воспроизведена или передана в какой-либо форме любыми средствами, механическими или электронными, включая фотокопирование, без письменного разрешения корпорации MapInfo, One Global View, Troy, New York 12180-8399.

© 2007 Корпорация MapInfo. Все права защищены. MapInfo, логотип MapInfo и MapBasic – торговые марки корпорации MapInfo и/или её филиалов.

Головной офис корпорации MapInfo:

Телефон: (518) 285-6000

Факс: (518) 285-6070

Горячая линия по продажам: (800) 327-8627

Горячая линия по продажам государственным учреждениям: (800) 619-2333

Телефонная линия технической поддержки: (518) 285-7283

Факс технической поддержки: (518) 285-6080

Контактная информация о всех офисах MapInfo находится по адресу: <http://www.mapinfo.com/contactus>.

Adobe Acrobat® - зарегистрированная торговая марка Adobe Systems Incorporated в США.

Продукты, упоминаемые в этом тексте, могут являться торговыми марками соответствующих производителей, которые этим признаются. Названия зарегистрированных торговых марок используются в этом тексте, по нашему мнению, к пользе их владельцев, без намерения нарушить права на торговую марку.

December 2007

# Содержание

---

|   |           |
|---|-----------|
| <b>Глава 1: Введение в MapBasic</b> .....               | <b>19</b> |
| <b>Основные сведения о языке программирования</b> ..... | <b>20</b> |
| <b>Основы языка MapBasic</b> .....                      | <b>21</b> |
| Переменные .....  | 21        |
| Циклы и другие управляющие операторы .....              | 21        |
| Вывод на печать и экспорт .....                         | 21        |
| Процедуры (Main и подпрограммы Sub) .....               | 21        |
| Обработка ошибок .....                                  | 22        |
| <b>Функции</b> .....                                    | <b>22</b> |
| Функции, созданные пользователем .....                  | 22        |
| Функции преобразования .....                            | 22        |
| Функции даты и времени .....                            | 23        |
| Математические функции .....                            | 23        |
| Функции работы со строками .....                        | 23        |
| <b>Работа с таблицами</b> .....                         | <b>24</b> |
| Создание и изменение таблицы .....                      | 24        |
| Запросы к таблицам .....                                | 25        |
| Работа с удаленными базами данных .....                 | 25        |
| <b>Работа с файлами (не таблицами)</b> .....            | <b>26</b> |
| Ввод/Вывод в файлы .....                                | 26        |
| Имена файлов и каталогов .....                          | 27        |
| <b>Работа с картами и графическими объектами</b> .....  | <b>27</b> |
| Создание объектов на карте .....                        | 27        |
| Изменение объектов на карте .....                       | 28        |
| Информация об объектах Карты .....                      | 28        |
| Работа со стилями объектов .....                        | 29        |
| Работа с окнами карт .....                              | 29        |
| <b>Создание элементов интерфейса</b> .....              | <b>30</b> |
| Панели инструментов .....                               | 30        |
| Диалоги .....   | 30        |
| Меню .....  | 31        |
| Окна .....  | 31        |
| Обработчики системных событий .....                     | 31        |

|  |           |
|--|-----------|
| Обмен данными с другими программами .....                              | 32        |
| DDE (Динамический обмен данными) .....                                 | 32        |
| Интегрированная Картография .....                                      | 32        |
| Специальные операторы и функции .....                                  | 33        |
| <b>Глава 2: Новые и дополненные операторы и функции MapBasic .....</b> | <b>37</b> |
| Функция CoordSysName\$( ) .....  | 38        |
| Функция CurDateTime .....  | 39        |
| Функция CurTime .....  | 39        |
| Функция FormatTime\$ .....   | 40        |
| Оператор FME Refresh Table .....                                       | 41        |
| Функция GetDate .....  | 41        |
| Функция GetTime() .....  | 43        |
| Функция HotlinkInfo() .....  | 43        |
| Функция Hour .....   | 44        |
| Функция MakeDateTime() .....   | 45        |
| Функция Minute .....   | 45        |
| Функция NumberToDateTime() .....                                       | 46        |
| Функция NumberToTime() .....   | 46        |
| Функция RegionInfo( ) .....  | 47        |
| Функция Second .....   | 47        |
| Функция StringToDateTime .....   | 48        |
| Функция StringToTime .....   | 48        |
| Дополненные функции и операторы MapBasic .....                         | 50        |
| Оператор Commit Table .....  | 50        |
| Функция GeocodeInfo( ) .....   | 52        |
| Функция IsogramInfo( ) .....   | 53        |
| Функция LabelInfo( ) .....   | 54        |
| Функция LayerInfo( ) .....   | 55        |
| Оператор Register Table .....  | 56        |
| Оператор Server Create Table .....                                     | 57        |
| Оператор Set Map .....   | 58        |
| Предложение Layer Activate .....                                       | 59        |
| Функция SystemInfo .....   | 62        |
| Функция TableInfo( ) .....   | 62        |
| <b>Глава 3: Алфавитный справочник языка MapBasic .....</b>             | <b>65</b> |
| Функция Acos( ) .....  | 67        |
| Оператор Add Cartographic Frame .....                                  | 69        |
| Оператор Add Column .....  | 71        |
| Оператор Add Map .....   | 77        |
| Оператор Alter Button .....  | 79        |

|  |     |
|--|-----|
| Оператор Alter ButtonPad . . . . .           | 80  |
| Оператор Alter Cartographic Frame . . . . .  | 86  |
| Оператор Alter Control . . . . .             | 87  |
| Оператор Alter MapInfoDialog . . . . .       | 89  |
| Оператор Alter Menu . . . . .                | 91  |
| Оператор Alter Menu Bar . . . . .            | 95  |
| Оператор Alter Menu Item . . . . .           | 96  |
| Оператор Alter Object                        | 99  |
| Оператор Alter Table                         | 106 |
| Функция ApplicationDirectory\$( ) . . . . .  | 107 |
| Функция Area( ) . . . . .                    | 108 |
| Функция AreaOverlap( ) . . . . .             | 109 |
| Функция Asc( ) . . . . .                     | 110 |
| Функция Asin( ) . . . . .                    | 111 |
| Функция Ask( ) . . . . .                     | 112 |
| Функция Atn( ) . . . . .                     | 113 |
| Оператор AutoLabel . . . . .                 | 114 |
| Оператор Beep . . . . .                      | 115 |
| Оператор Browse . . . . .                    | 115 |
| Предложение Brush . . . . .                  | 117 |
| Функция Buffer( ) . . . . .                  | 119 |
| Функция ButtonPadInfo( ) . . . . .           | 120 |
| Оператор Call . . . . .                      | 121 |
| Функция CartesianArea( ) . . . . .           | 124 |
| Функция CartesianBuffer( ) . . . . .         | 125 |
| Функция CartesianConnectObjects( ) . . . . . | 126 |
| Функция CartesianDistance( ) . . . . .       | 127 |
| Функция CartesianObjectDistance( ) . . . . . | 128 |
| Функция CartesianObjectLen( ) . . . . .      | 129 |
| Функция CartesianOffset( ) . . . . .         | 130 |
| Функция CartesianOffsetXY( ) . . . . .       | 131 |
| Функция CartesianPerimeter( ) . . . . .      | 132 |
| Функция Centroid( ) . . . . .                | 133 |
| Функция CentroidX( ) . . . . .               | 134 |
| Функция CentroidY( ) . . . . .               | 135 |
| Предложение CharSet . . . . .                | 136 |
| Функция ChooseProjection\$( ) . . . . .      | 139 |
| Функция Chr\$( ) . . . . .                   | 140 |
| Оператор Close All . . . . .                 | 141 |
| Оператор Close Connection . . . . .          | 141 |
| Оператор Close File . . . . .                | 142 |

|   |     |
|---|-----|
| Оператор Close Table .....  | 142 |
| Оператор Close Window .....   | 144 |
| Функция ColumnInfo( ) .....   | 145 |
| Функция Combine( ) .....  | 147 |
| CommandInfo( ) функция .....  | 149 |
| Оператор Commit Table .....   | 154 |
| ConnectObjects( ) функция .....   | 158 |
| Оператор Continue .....   | 159 |
| Предложения Control Button / OKButton / CancelButton .....              | 161 |
| Предложение Control CheckBox .....                                      | 162 |
| Предложение Control DocumentWindow .....                                | 163 |
| Предложение Control EditText .....                                      | 164 |
| Предложение Control GroupBox .....                                      | 165 |
| Предложения Control ListBox / MultiListBox .....                        | 166 |
| Предложения Control PenPicker/BrushPicker/SymbolPicker/FontPicker ..... | 169 |
| Предложение Control PopupMenu .....                                     | 170 |
| Предложение Control RadioGroup .....                                    | 171 |
| Предложение Control StaticText .....                                    | 172 |
| Функция ConvertToPline( ) .....   | 173 |
| Функция ConvertToRegion( ) .....  | 174 |
| Функция ConvexHull( ) .....   | 175 |
| Оператор CoordSys .....   | 175 |
| Функция CoordSysName .....  | 179 |
| Функция Cos( ) .....  | 180 |
| Оператор Create Arc .....   | 181 |
| Оператор Create ButtonPad .....   | 182 |
| Оператор Create ButtonPads As Default .....                             | 186 |
| Оператор Create Cartographic Legend .....                               | 186 |
| Функция CreateCircle( ) .....   | 190 |
| Оператор Create Collection .....  | 192 |
| Оператор Create Cutter .....  | 193 |
| Оператор Create Ellipse .....   | 195 |
| Оператор Create Frame .....   | 196 |
| Оператор Create Grid .....  | 198 |
| Оператор Create Index .....   | 201 |
| Оператор Create Legend .....  | 201 |
| Функция CreateLine( ) .....   | 202 |
| Оператор Create Line .....  | 203 |
| Оператор Create Map .....   | 204 |
| Оператор Create Map3D .....   | 205 |
| Оператор Create Menu .....  | 207 |

|   |     |
|---|-----|
| Оператор Create Menu Bar . . . . .          | 213 |
| Оператор Create MultiPoint . . . . .        | 214 |
| Оператор Create Object. . . . .             | 217 |
| Оператор Create Pline . . . . .             | 223 |
| Функция CreatePoint( ) . . . . .            | 224 |
| Оператор Create Point. . . . .              | 225 |
| Оператор Create PrismMap. . . . .           | 226 |
| Оператор Create Ranges. . . . .             | 228 |
| Оператор Create Rect . . . . .              | 231 |
| Оператор Create Redistricter . . . . .      | 232 |
| Оператор Create Region . . . . .            | 233 |
| Оператор Create Report From Table . . . . . | 235 |
| Оператор Create RoundRect. . . . .          | 235 |
| Оператор Create Styles . . . . .            | 236 |
| Оператор Create Table . . . . .             | 239 |
| Функция CreateText( ) . . . . .             | 241 |
| Оператор Create Text . . . . .              | 242 |
| Функция CurDate( ) . . . . .                | 243 |
| Функция CurDateTime . . . . .               | 244 |
| Функция CurrentBorderPen( ) . . . . .       | 244 |
| Функция CurrentBrush( ) . . . . .           | 245 |
| Функция CurrentFont( ) . . . . .            | 246 |
| Функция CurrentLinePen( ) . . . . .         | 247 |
| Функция CurrentPen( ) . . . . .             | 247 |
| Функция CurrentSymbol( ) . . . . .          | 248 |
| Функция CurTime . . . . .                   | 249 |
| Функция DateWindow( ) . . . . .             | 251 |
| Функция Day( ) . . . . .                    | 251 |
| Оператор DDEExecute. . . . .                | 252 |
| Функция DDEInitiate( ) . . . . .            | 253 |
| Оператор DDEPoke . . . . .                  | 256 |
| Функция DDERequest\$( ) . . . . .           | 257 |
| Оператор DDETerminate . . . . .             | 260 |
| Оператор DDETerminateAll. . . . .           | 260 |
| Оператор Declare Function. . . . .          | 261 |
| Оператор Declare Sub . . . . .              | 263 |
| Оператор Define . . . . .                   | 265 |
| Функция DeformatNumber\$( ) . . . . .       | 266 |
| Оператор Delete . . . . .                   | 267 |
| Оператор Dialog . . . . .                   | 268 |
| Оператор Dialog Preserve. . . . .           | 273 |

|   |     |
|---|-----|
| Оператор Dialog Remove .....                | 274 |
| Оператор Dim .....                          | 275 |
| Функция Distance( ) .....                   | 280 |
| Оператор Do Case...End Case .....           | 281 |
| Оператор Do...Loop .....                    | 283 |
| Оператор Drop Index .....                   | 284 |
| Оператор Drop Map .....                     | 285 |
| Оператор Drop Table .....                   | 286 |
| Оператор End MapInfo .....                  | 287 |
| Оператор End Program .....                  | 288 |
| Процедура EndHandler .....                  | 289 |
| Функция EOF( ) .....                        | 289 |
| Функция EOT( ) .....                        | 290 |
| Функция EPSGToCoordSysString\$( ) .....     | 291 |
| Функция Erase( ) .....                      | 292 |
| Функция Err( ) .....                        | 292 |
| Оператор Error .....                        | 294 |
| Функция Error\$( ) .....                    | 294 |
| Оператор Exit Do .....                      | 295 |
| Оператор Exit For .....                     | 296 |
| Оператор Exit Function .....                | 296 |
| Оператор Exit Sub .....                     | 297 |
| Функция Exp( ) .....                        | 298 |
| Оператор Export .....                       | 299 |
| Функция ExtractNodes( ) .....               | 302 |
| Оператор Farthest .....                     | 303 |
| Оператор Fetch .....                        | 305 |
| Функция FileAttr( ) .....                   | 307 |
| Функция FileExists( ) .....                 | 308 |
| Функция FileOpenDlg( ) .....                | 309 |
| Функция FileSaveAsDlg( ) .....              | 311 |
| Оператор Find .....                         | 312 |
| Оператор Find Using .....                   | 315 |
| Оператор Fix( ) .....                       | 316 |
| Предложение Font .....                      | 317 |
| Оператор For...Next .....                   | 319 |
| Процедура ForegroundTaskSwitchHandler ..... | 321 |
| Функция Format\$( ) .....                   | 321 |
| Функция FormatDate\$( ) .....               | 324 |
| Функция FormatNumber\$( ) .....             | 325 |
| Функция FormatTime\$ .....                  | 326 |



|  |     |
|--|-----|
| Оператор FME Refresh Table .....       | 327 |
| Функция FrontWindow( ) .....           | 328 |
| Оператор Function...End Function ..... | 328 |
| Оператор Geocode .....                 | 332 |
| Функция GeocodeInfo( ) .....           | 337 |
| Оператор Get .....                     | 341 |
| Функция GetDate .....                  | 342 |
| Функция GetFolderPath\$( ) .....       | 343 |
| Функция GetMetadata\$( ) .....         | 344 |
| Функция GetSeamlessSheet( ) .....      | 345 |
| Функция GetTime( ) .....               | 346 |
| Оператор Global .....                  | 346 |
| Оператор Goto .....                    | 347 |
| Оператор Graph .....                   | 348 |
| Функция HomeDirectory\$( ) .....       | 351 |
| Функция HotlinkInfo( ) .....           | 351 |
| Функция Hour .....                     | 352 |
| Оператор If...Then .....               | 353 |
| Оператор Import .....                  | 354 |
| Оператор Include .....                 | 360 |
| Оператор Input # .....                 | 361 |
| Оператор Insert .....                  | 362 |
| Функция InStr( ) .....                 | 364 |
| Функция Int( ) .....                   | 365 |
| Функция IntersectNodes( ) .....        | 366 |
| Функция IsogramInfo( ) .....           | 366 |
| Функция IsPenWidthPixels( ) .....      | 370 |
| Оператор Kill .....                    | 371 |
| Функция LabelFindByID( ) .....         | 372 |
| Функция LabelFindFirst( ) .....        | 373 |
| Функция LabelFindNext( ) .....         | 374 |
| Функция Labelinfo( ) .....             | 375 |
| Функция LayerInfo( ) .....             | 378 |
| Оператор Layout .....                  | 384 |
| Функция LCase\$( ) .....               | 385 |
| Функция Left\$( ) .....                | 387 |
| Функция LegendFrameInfo( ) .....       | 387 |
| Функция LegendInfo( ) .....            | 389 |
| Функция LegendStyleInfo( ) .....       | 390 |
| Функция Len( ) .....                   | 391 |
| Функция Like( ) .....                  | 391 |

|  |     |
|--|-----|
| Оператор Line Input. . . . .               | 392 |
| Функция LocateFile\$( ) . . . . .          | 393 |
| Функция LOF( ) . . . . .                   | 395 |
| Функция Log( ) . . . . .                   | 395 |
| Функция LTrim\$( ) . . . . .               | 396 |
| Процедура Main . . . . .                   | 397 |
| Функция MakeBrush( ) . . . . .             | 398 |
| Функция MakeCustomSymbol( ) . . . . .      | 399 |
| Функция MakeDateTime( ) . . . . .          | 400 |
| Функция MakeFont( ) . . . . .              | 400 |
| Функция MakeFontSymbol( ) . . . . .        | 401 |
| Функция MakePen( ) . . . . .               | 403 |
| Функция MakeSymbol( ) . . . . .            | 403 |
| Оператор Map. . . . .                      | 404 |
| Функция Map3DInfo( ) . . . . .             | 406 |
| Функция MapperInfo( ) . . . . .            | 409 |
| Функция Maximum( ) . . . . .               | 414 |
| Функция MBR( ) . . . . .                   | 415 |
| Оператор Menu Bar . . . . .                | 416 |
| Функция MenuItemInfoByHandler( ) . . . . . | 416 |
| Функция MenuItemInfoByID( ) . . . . .      | 418 |
| Оператор Metadata . . . . .                | 419 |
| Функция MGRSToPoint( ) . . . . .           | 422 |
| Функция Mid\$( ) . . . . .                 | 423 |
| Функция MidByte\$( ) . . . . .             | 424 |
| Функция Minimum( ) . . . . .               | 425 |
| Функция Minute . . . . .                   | 425 |
| Функция Month( ) . . . . .                 | 426 |
| Оператор Nearest. . . . .                  | 426 |
| Оператор Note . . . . .                    | 430 |
| Функция NumAllWindows( ) . . . . .         | 430 |
| Функция NumberToDate( ) . . . . .          | 431 |
| Функция NumberToDateTime( ) . . . . .      | 432 |
| Функция NumberToTime( ) . . . . .          | 432 |
| Функция NumCols( ) . . . . .               | 433 |
| Функция NumTables( ) . . . . .             | 433 |
| Функция NumWindows( ) . . . . .            | 434 |
| Функция ObjectDistance( ) . . . . .        | 437 |
| Функция ObjectGeography( ) . . . . .       | 437 |
| Функция ObjectInfo( ) . . . . .            | 441 |
| Функция ObjectLen( ) . . . . .             | 445 |

|                               |     |
|-------------------------------|-----|
| Функция ObjectNodeHasM( )     | 446 |
| Функция ObjectNodeHasZ( )     | 448 |
| Функция ObjectNodeM( )        | 449 |
| Функция ObjectNodeX( )        | 451 |
| Функция ObjectNodeY( )        | 452 |
| Функция ObjectNodeZ( )        | 453 |
| Оператор Objects Check        | 454 |
| Оператор Objects Clean        | 456 |
| Оператор Objects Combine      | 457 |
| Оператор Objects Disaggregate | 459 |
| Оператор Objects Enclose      | 461 |
| Оператор Objects Erase        | 462 |
| Оператор Objects Intersect    | 464 |
| Оператор Objects Move         | 466 |
| Оператор Objects Offset       | 467 |
| Оператор Objects Overlay      | 468 |
| Оператор Objects Pline        | 469 |
| Оператор Objects Snap         | 470 |
| Оператор Objects Split        | 472 |
| Функция Offset( )             | 475 |
| Функция OffsetXY( )           | 476 |
| Оператор OnError              | 477 |
| Оператор Open Connection      | 478 |
| Оператор Open File            | 480 |
| Оператор Open Report          | 482 |
| Оператор Open Table           | 483 |
| Оператор Open Window          | 485 |
| Функция Overlap( )            | 486 |
| Функция OverlayNodes( )       | 487 |
| Оператор Pack Table           | 488 |
| Функция PathToDirectory\$( )  | 490 |
| Функция PathToFileName\$( )   | 490 |
| Функция PathToTableName\$( )  | 491 |
| Предложение Pen               | 492 |
| Функция PenWidthToPoints( )   | 495 |
| Функция Perimeter( )          | 496 |
| Функция PointsToPenWidth( )   | 497 |
| Функция PointToMGRS\$( )      | 497 |
| Оператор Print                | 499 |
| Оператор Print #              | 500 |
| Оператор PrintWin             | 501 |

|                                     |     |
|-------------------------------------|-----|
| Функция PrismMapInfo( )             | 502 |
| Функция ProgramDirectory\$( )       | 506 |
| Оператор ProgressBar                | 506 |
| Функция Proper\$( )                 | 509 |
| Функция ProportionOverlap( )        | 510 |
| Оператор Put                        | 510 |
| Оператор Randomize.                 | 511 |
| Функция RegionInfo( )               | 512 |
| Функция ReadControlValue( )         | 513 |
| Оператор ReDim                      | 516 |
| Оператор Register Table             | 518 |
| Оператор Relief Shade               | 527 |
| Оператор Reload Symbols             | 527 |
| Процедура RemoteMapGenHandler.      | 528 |
| Процедура RemoteMsgHandler          | 529 |
| Функция RemoteQueryHandler( )       | 530 |
| Оператор Remove Cartographic Frame. | 531 |
| Оператор Remove Map                 | 532 |
| Оператор Rename File.               | 533 |
| Оператор Rename Table               | 533 |
| Оператор Reproject.                 | 534 |
| Resume statement                    | 535 |
| Функция RGB( )                      | 536 |
| Функция Right\$( )                  | 537 |
| Функция Rnd( )                      | 537 |
| Оператор Rollback.                  | 538 |
| Функция Rotate( )                   | 539 |
| Функция RotateAtPoint( )            | 540 |
| Функция Round( )                    | 541 |
| Функция RTrim\$( )                  | 542 |
| Оператор Run Application            | 543 |
| Оператор Run Command                | 543 |
| Оператор Run Menu Command           | 546 |
| Оператор Run Program.               | 547 |
| Оператор Save File                  | 549 |
| Оператор Save MWS                   | 549 |
| Оператор Save Window.               | 551 |
| Оператор Save Workspace             | 553 |
| Функция SearchInfo( )               | 554 |
| Функция SearchPoint( )              | 557 |
| Функция SearchRect( )               | 558 |

|   |     |
|---|-----|
| Функция Second .....                    | 559 |
| Функция Seek( ) .....                   | 559 |
| Оператор Seek .....                     | 560 |
| Процедура SelChangedHandler .....       | 560 |
| Оператор Select .....                   | 562 |
| Функция SelectionInfo( ) .....          | 571 |
| Оператор Server Begin Transaction ..... | 572 |
| Оператор Server Bind Column .....       | 573 |
| Оператор Server Close .....             | 574 |
| Функция Server_ColumnInfo( ) .....      | 575 |
| Оператор Server Commit .....            | 577 |
| Функция Server_Connect( ) .....         | 578 |
| Функция Server_ConnectInfo( ) .....     | 587 |
| Оператор Server Create Map .....        | 588 |
| Оператор Server Create Style .....      | 590 |
| Оператор Server Create Table .....      | 591 |
| Оператор Server Create Workspace .....  | 594 |
| Оператор Server Disconnect .....        | 595 |
| Функция Server_DriverInfo( ) .....      | 595 |
| Функция Server_EOT( ) .....             | 596 |
| Функция Server_Execute( ) .....         | 597 |
| Оператор Server Fetch .....             | 599 |
| Функция Server_GetODBCHConn( ) .....    | 601 |
| Функция Server_GetODBCHStmt( ) .....    | 601 |
| Оператор Server Link Table .....        | 603 |
| Функция Server_NumCols( ) .....         | 606 |
| Функция Server_NumDrivers( ) .....      | 607 |
| Оператор Server Refresh .....           | 608 |
| Оператор Server Remove Workspace .....  | 609 |
| Оператор Server Rollback .....          | 609 |
| Оператор Server Set Map .....           | 610 |
| Оператор Server Versioning .....        | 611 |
| Оператор Server Workspace Merge .....   | 613 |
| Оператор Server Workspace Refresh ..... | 615 |
| Функция SessionInfo( ) .....            | 617 |
| Оператор Set Application Window .....   | 617 |
| Оператор Set Area Units .....           | 618 |
| Оператор Set Browse .....               | 620 |
| Оператор Set Cartographic Legend .....  | 621 |
| Оператор Set Command Info .....         | 622 |
| Оператор Set Connection Geocode .....   | 623 |

|  |     |
|--|-----|
| Оператор Set Connection Isogram .....      | 626 |
| Оператор Set CoordSys .....                | 629 |
| Оператор Set Date Window .....             | 630 |
| Оператор Set Datum Transform Version ..... | 631 |
| Оператор Set Digitizer .....               | 632 |
| Оператор Set Distance Units .....          | 633 |
| Оператор Set Drag Threshold .....          | 635 |
| Оператор Set Event Processing .....        | 635 |
| Оператор Set File Timeout .....            | 636 |
| Оператор Set Format .....                  | 637 |
| Оператор Set Graph .....                   | 638 |
| Оператор Set Handler .....                 | 644 |
| Оператор Set Layout .....                  | 645 |
| Оператор Set Legend .....                  | 647 |
| Оператор Set Map .....                     | 650 |
| Оператор Set Map3D .....                   | 667 |
| Оператор Set Next Document .....           | 669 |
| Оператор Set Paper Units .....             | 671 |
| Оператор Set PrismMap .....                | 672 |
| Оператор Set ProgressBars .....            | 673 |
| Оператор Set Redistricter .....            | 674 |
| Оператор Set Resolution .....              | 676 |
| Оператор Set Shade .....                   | 676 |
| Оператор Set Style .....                   | 677 |
| Оператор Set Table Datum .....             | 679 |
| Оператор Set Table .....                   | 679 |
| Оператор Set Target .....                  | 681 |
| Оператор Set Window .....                  | 682 |
| Функция Sgn( ) .....                       | 690 |
| Оператор Shade .....                       | 691 |
| Функция Sin( ) .....                       | 702 |
| Функция Space\$( ) .....                   | 703 |
| Функция SphericalArea( ) .....             | 703 |
| Функция SphericalConnectObjects( ) .....   | 704 |
| Функция SphericalDistance( ) .....         | 705 |
| Функция SphericalObjectDistance( ) .....   | 706 |
| Функция SphericalObjectLen( ) .....        | 707 |
| Функция SphericalOffset( ) .....           | 708 |
| Функция SphericalOffsetXY( ) .....         | 709 |
| Функция SphericalPerimeter( ) .....        | 710 |
| Функция Sqr( ) .....                       | 711 |

|  |     |
|--|-----|
| Оператор StatusBar . . . . .               | 712 |
| Параметр Stop . . . . .                    | 712 |
| Функция Str\$( ) . . . . .                 | 714 |
| Функция String\$( ) . . . . .              | 715 |
| Функция StringCompare( ) . . . . .         | 715 |
| Функция StringCompareIntl( ) . . . . .     | 716 |
| Функция StringToDate( ) . . . . .          | 717 |
| Функция StringToDateTime . . . . .         | 718 |
| Функция StringToTime . . . . .             | 720 |
| Функция StyleAttr( ) . . . . .             | 720 |
| Оператор Sub...End Sub . . . . .           | 723 |
| Предложение Symbol . . . . .               | 725 |
| Функция SystemInfo( ) . . . . .            | 728 |
| Функция TableInfo( ) . . . . .             | 730 |
| Функция Tan( ) . . . . .                   | 734 |
| Функция TempFileName\$( ) . . . . .        | 735 |
| Оператор Terminate Application . . . . .   | 736 |
| Функция TextSize( ) . . . . .              | 736 |
| Функция Time( ) . . . . .                  | 737 |
| Функция Timer( ) . . . . .                 | 737 |
| Процедура ToolHandler . . . . .            | 738 |
| Функция TriggerControl( ) . . . . .        | 740 |
| Функция TrueFileName\$( ) . . . . .        | 741 |
| Оператор Type . . . . .                    | 741 |
| Функция UBound( ) . . . . .                | 742 |
| Функция UCase\$( ) . . . . .               | 743 |
| Оператор UnDim . . . . .                   | 744 |
| Функция UnitAbbr\$( ) . . . . .            | 745 |
| Функция UnitName\$( ) . . . . .            | 745 |
| Оператор Unlink . . . . .                  | 746 |
| Оператор Update . . . . .                  | 747 |
| Оператор Update Window . . . . .           | 748 |
| Функция Val( ) . . . . .                   | 748 |
| Функция Weekday( ) . . . . .               | 749 |
| Оператор WFS Refresh Table . . . . .       | 750 |
| Оператор While...Wend . . . . .            | 750 |
| Процедура WinChangedHandler . . . . .      | 752 |
| Процедура WinClosedHandler . . . . .       | 753 |
| Функция WindowID( ) . . . . .              | 754 |
| Функция WindowInfo( ) . . . . .            | 755 |
| Процедура WinFocusChangedHandler . . . . . | 762 |

|   |            |
|---|------------|
| Оператор Write # .....                          | 763        |
| Функция Year( ) .....                           | 763        |
| <b>Приложение А: Библиотеки HTTP и FTP.....</b> | <b>765</b> |
| Процедура MICloseContent( ) 766 .....           | 766        |
| Процедура MICloseFtpConnection( ) .....         | 766        |
| Процедура MICloseFtpFileFind( ) .....           | 766        |
| Процедура MICloseHttpConnection( ) .....        | 767        |
| Процедура MICloseHttpFile( ) .....              | 767        |
| Процедура MICloseSession( ) .....               | 768        |
| Функция MCreateSession( ) .....                 | 768        |
| Функция MCreateSessionFull( ) .....             | 769        |
| Функция MErrorDlg( ) .....                      | 771        |
| Функция MFindFtpFile( ) .....                   | 772        |
| Функция MFindNextFtpFile( ) .....               | 773        |
| Функция MGetContent( ) .....                    | 773        |
| Функция MGetContentBuffer( ) .....              | 774        |
| Функция MGetContentLen( ) .....                 | 775        |
| Функция MGetContentString( ) .....              | 775        |
| Функция MGetContentToFile( ) .....              | 776        |
| Функция MGetContentType( ) .....                | 776        |
| Функция MGetCurrentFtpDirectory( ) .....        | 777        |
| Функция MGetErrorCode( ) .....                  | 778        |
| Функция MGetErrorMessage( ) .....               | 779        |
| Функция MGetFileURL( ) .....                    | 779        |
| Функция MGetFtpConnection( ) .....              | 780        |
| Функция MGetFtpFile( ) .....                    | 781        |
| Функция MGetFtpFileFind( ) .....                | 782        |
| Процедура MGetFtpFileName( ) .....              | 783        |
| Функция MGetHttpConnection( ) .....             | 784        |
| Функция MIsFtpDirectory( ) .....                | 784        |
| Функция MIsFtpDots( ) .....                     | 785        |
| Функция MOpenRequest( ) .....                   | 785        |
| Функция MOpenRequestFull( ) .....               | 786        |
| Функция MParseURL( ) .....                      | 788        |
| Функция MPutFtpFile( ) .....                    | 790        |
| Функция MQueryInfo( ) .....                     | 791        |
| Функция MQueryInfoStatusCode( ) .....           | 791        |
| Функция MSaveContent( ) .....                   | 793        |
| Функция MSendRequest( ) .....                   | 793        |
| Функция MSendSimpleRequest( ) .....             | 794        |
| Функция MSetCurrentFtpDirectory( ) .....        | 794        |



|  |            |
|--|------------|
| Функция MISetSessionTimeout( ) .....                 | 795        |
| <b>Приложение В: Библиотека XML .....</b>            | <b>797</b> |
| Процедура MIXmlAttributeListDestroy( ) 798           |            |
| Функция MIXmlDocumentCreate( ) .....                 | 798        |
| Процедура MIXmlDocumentDestroy( ) .....              | 799        |
| Функция MIXmlDocumentGetNamespaces( ) .....          | 799        |
| Функция MIXmlDocumentGetRootNode( ) .....            | 800        |
| Функция MIXmlDocumentLoad( ) .....                   | 800        |
| Функция MIXmlDocumentLoadXML( ) .....                | 801        |
| Функция MIXmlDocumentLoadXMLString( ) .....          | 802        |
| Функция MIXmlDocumentSetProperty( ) .....            | 803        |
| Функция MIXmlGetAttributeList( ) .....               | 804        |
| Функция MIXmlGetChildList( ) .....                   | 805        |
| Функция MIXmlGetNextAttribute( ) .....               | 805        |
| Функция MIXmlGetNextNode( ) .....                    | 806        |
| Процедура MIXmlNodeDestroy( ) .....                  | 806        |
| Функция MIXmlNodeGetAttributeValue( ) .....          | 807        |
| Функция MIXmlNodeGetFirstChild( ) .....              | 808        |
| Функция MIXmlNodeGetName( ) .....                    | 808        |
| Функция MIXmlNodeGetParent( ) .....                  | 809        |
| Функция MIXmlNodeGetText( ) .....                    | 809        |
| Функция MIXmlNodeGetValue( ) .....                   | 810        |
| Процедура MIXmlNodeListDestroy( ) .....              | 811        |
| Процедура MIXmlISCDestroy( ) .....                   | 811        |
| Функция MIXmlISCGetLength( ) .....                   | 812        |
| Функция MIXmlISCGetNamespace( ) .....                | 812        |
| Функция MIXmlSelectNodes( ) .....                    | 813        |
| Функция MIXmlSelectSingleNode( ) .....               | 814        |
| <b>Приложение С: Character Code Table .....</b>      | <b>815</b> |
| <b>Приложение D: Сведения об операторах .....</b>    | <b>817</b> |
| Операторы сравнения .....                            | 818        |
| Логические Операторы .....                           | 819        |
| Географические операторы .....                       | 819        |
| Старшинство выполнения операторов. ....              | 820        |
| Автоматическое преобразование типов .....            | 821        |
| <b>Приложение E: MapBasic Definitions File .....</b> | <b>823</b> |
| <b>Указатель .....</b>                               | <b>855</b> |



# Введение в MapBasic

В этом справочнике описаны все операторы и функции языка программирования MapBasic. Основы программирования на MapBasic и сведения об использовании среды разработки программ MapBasic приведены в *Руководстве пользователя MapBasic*.

## В этой главе...

|  |     |
|--|-----|
| ♦ Принятые обозначения . . . . .                       | .20 |
| ♦ Основные сведения о языке программирования . . . . . | .20 |
| ♦ Основы языка MapBasic . . . . .                      | .21 |
| ♦ Функции . . . . .                                    | .22 |
| ♦ Работа с таблицами . . . . .                         | .24 |
| ♦ Работа с файлами (не таблицами) . . . . .            | .26 |
| ♦ Работа с картами и графическими объектами . . . . .  | .27 |
| ♦ Создание элементов интерфейса . . . . .              | .30 |
| ♦ Обмен данными с другими программами . . . . .        | .32 |
| ♦ Специальные операторы и функции . . . . .            | .33 |

## Принятые обозначения

В этом справочнике приняты следующие условные обозначения в тексте:

| Условное обозначение                         | Смысл   |
|--|---|
| <b>If, Call, Map, Browse, Area</b>           | Слова и термины, начинающиеся с заглавной буквы, выделенные жирным шрифтом – это зарезервированные ключевые слова MapBasic.<br><br>В этом справочнике у всех зарезервированных ключевых слов первая буква набрана с заглавной буквы; однако, при написании MapBasic-программ эти зарезервированные ключевые слова можно вводить любым способом: в верхнем регистре, в нижнем или смешанном. |
| Main, Pen, Object                            | Слова, начинающиеся с заглавной буквы, но набранные нежирным шрифтом, обычно являются именами специальных процедур или типами переменных.   |
| <i>table, handler, window_id</i>             | Слова, набранные курсивом – параметры операторов MapBasic. При оформлении оператора MapBasic, требуется задать каждому параметру необходимое выражение.   |
| [ <i>window_id</i> ], [ <b>Interactive</b> ] | Зарезервированные ключевые слова или параметры в квадратных скобках являются необязательными.   |
| { <b>On</b>   <b>Off</b> }                   | Синтаксическое выражение внутри фигурных скобок содержит список ключевых зарезервированных слов или параметров, разделенных вертикальной чертой (   ). Требуется выбрать один из перечисленных вариантов. Например, в примере, показанном слева ({ <b>On</b>   <b>Off</b> } ), требуется выбрать либо <b>On</b> , либо <b>Off</b> .   |
| Note "Привет от MapBasic!"                   | Примеры программ набраны шрифтом Courier.   |

## Основные сведения о языке программирования

Далее приведены основные сведения о языке программирования MapBasic. Задача ставится слева; соответствующие ей операторы и функции перечислены справа и выделены **жирным** шрифтом. После имен функций стоят скобки ( ).

## Основы языка MapBasic

### Переменные

|   |                                |
|---|--------------------------------|
| Объявление локальных и глобальных переменных: | <b>Dim, Global</b>             |
| Изменение размера массива переменных:         | <b>ReDim, UBound( ), UnDim</b> |
| Объявление заданной структуры данных:         | <b>Тип</b>                     |

### Циклы и другие управляющие операторы

|                               |   |
|-------------------------------|---|
| Циклы:                        | <b>For...Next, Exit For, Do...Loop, Exit Do, While...Wend</b> |
| Ветвление:                    | <b>If...Then, Do Case, GoTo</b>                               |
| Другие управляющие операторы: | <b>End Program, Terminate Application, End MapInfo</b>        |

### Вывод на печать и экспорт

|                          |   |
|--------------------------|---|
| Печать содержимого окна: | <b>PrintWin</b>                         |
| Печать в окне сообщений: | <b>Print</b>                            |
| Настройка Отчета:        | <b>Layout, Create Frame, Set Window</b> |
| Экспорт окна в файл:     | <b>Save Window</b>                      |
| Управление принтером:    | <b>Set Window, Window Info( )</b>       |

### Процедуры (Main и подпрограммы Sub)

|                         |                                   |
|-------------------------|-----------------------------------|
| Определение процедур:   | <b>Declare Sub, Sub...End Sub</b> |
| Вызов процедуры:        | <b>Call</b>                       |
| Выход из процедуры:     | <b>Exit Sub</b>                   |
| Главная процедура Main: | <b>Main</b>                       |

## Обработка ошибок

|  |                           |
|--|---------------------------|
| Настройка процедуры обработки ошибок:    | <b>OnError</b>            |
| Возвратить сведения об очередной ошибке: | <b>Err( ), Error\$( )</b> |
| Выход из процедуры обработки ошибок:     | <b>Resume</b>             |
| Моделирование ошибки:                    | <b>Error</b>              |

## Функции

### Функции, созданные пользователем

|                            |  |
|----------------------------|--|
| Определение новых функций: | <b>Declare Function, Function...End Function</b> |
| Выход из функции:          | <b>Exit Function</b>                             |

### Функции преобразования

|   |  |
|---|--|
| Функции преобразования строки в код:                | <b>Asc( )</b>                                |
| Преобразование кода в строку:                       | <b>Chr\$( )</b>                              |
| Преобразование строки в число:                      | <b>Val( )</b>                                |
| Преобразование числа в строку:                      | <b>Str\$( ), Format\$( )</b>                 |
| Преобразование строки или числа в дату:             | <b>NumberToDate( ), StringToDate( )</b>      |
| Преобразование в номер года из двух цифр:           | <b>Set Date Window, DateWindow( )</b>        |
| Преобразование типа объекта:                        | <b>ConvertToRegion( ), ConvertToPline( )</b> |
| Преобразование подписей в текст:                    | <b>LabelInfo( )</b>                          |
| Преобразование точечного объекта в координаты MGRS: | <b>PointToMGRS\$( )</b>                      |
| Преобразование координат MGRS в точечный объект:    | <b>MGRSToPoint( )</b>                        |

## Функции даты и времени

|   |  |
|---|--|
| Получить текущую дату:                  | <b>CurDate( )</b>                            |
| Выделить часть даты:                    | <b>Day( ), Month( ), Weekday( ), Year( )</b> |
| Системное время:                        | <b>Timer( )</b>                              |
| Преобразование строки или числа в дату: | <b>NumberToDate( ), StringToDate( )</b>      |

## Математические функции

|                                |   |
|--------------------------------|---|
| Тригонометрические функции:    | <b>Cos( ), Sin( ), Tan( ), Acos( ), Asin( ), Atn( )</b>   |
| Географические функции:        | <b>Area( ), Perimeter( ), Distance( ),<br/>ObjectLen( ), CartesianArea( ),<br/>CartesianPerimeter( ), CartesianDistance( ),<br/>CartesianObjectLen( ), SphericalArea( ),<br/>SphericalPerimeter( ), SphericalDistance( ),<br/>SphericalObjectLen( )</b> |
| Случайные числа:               | <b>Randomize, Rnd( )</b>  |
| Функции работы со знаком:      | <b>Abs( ), Sgn( )</b>   |
| Округление:                    | <b>Fix( ), Int( ), Round( )</b>   |
| Другие математические функции: | <b>Exp( ), Log( ), Minimum( ), Maximum( ),<br/>Sqr( )</b>   |

## Функции работы со строками

|                           |  |
|---------------------------|--|
| Преобразование регистра:  | <b>UCase\$( ), LCase\$( ), Proper\$( )</b>   |
| Поиск подстроки:          | <b>InStr( )</b>  |
| Выделение части строки:   | <b>Left\$( ), Right\$( ), Mid\$( ), MidByte\$( )</b>                                 |
| Удаление пробелов:        | <b>LTrim\$( ), RTrim\$( )</b>  |
| Форматирование:           | <b>Format\$( ), Str\$( ), Set Format,<br/>FormatNumber\$( ), DeformatNumber\$( )</b> |
| Определение длины строки: | <b>Len( )</b>  |
| Преобразование кодов:     | <b>Chr\$( ), Asc( )</b>  |

|   |  |
|---|--|
| Сравнение:  | <b>Like( ), StringCompare( ), StringCompareIntl( )</b> |
| Повторение символов:                                | <b>Space\$( ), String\$( )</b>                         |
| Работа с единицами измерения:                       | <b>UnitAbbr\$( ), UnitName\$( )</b>                    |
| Преобразование точечного объекта в координаты MGRS: | <b>PointToMGRS\$( )</b>                                |
| Преобразование координат MGRS в точечный объект:    | <b>MGRSToPoint( )</b>                                  |
| Преобразование строки EPSG в выражение CoordSys:    | <b>EPSGToCoordSysString\$( )</b>                       |

## Работа с таблицами

### Создание и изменение таблицы

|                                       |  |
|---------------------------------------|--|
| Открытие таблицы:                     | <b>Open Table</b>  |
| Закрытие одной или более таблиц:      | <b>Close Table, Close All</b>  |
| Создание пустой таблицы:              | <b>Create Table</b>  |
| Создание таблицы из другого файла:    | <b>Register Table</b>  |
| Импорт/экспорт таблиц/файлов:         | <b>Import, Export</b>  |
| Изменение структуры таблицы:          | <b>Alter Table, Add Column, Create Index, Drop Index, Create Map, Drop Map</b> |
| Создание отчета Crystal Reports:      | <b>Onepartop Create Report From Table</b>                                      |
| Загрузка отчета Crystal Reports:      | <b>Open Report</b>   |
| Создание, изменение и удаление строк: | <b>Insert, Update, Delete</b>  |
| Упаковка таблицы:                     | <b>Pack Table</b>  |



|  |                     |
|--|---------------------|
| Управление режимами работы с таблицей: | <b>Set Table</b>    |
| Сохранение изменений в таблице:        | <b>Table</b>        |
| Восстановление после изменений:        | <b>Rollback</b>     |
| Переименование таблицы:                | <b>Rename Table</b> |
| Удаление таблицы:                      | <b>Drop Table</b>   |

## Запросы к таблицам

|                                   |   |
|-----------------------------------|---|
| Позиционировать курсор:           | <b>Fetch, EOT( )</b>                                |
| Выбор данных, работа с выборками: | <b>Select, SelectionInfo( )</b>                     |
| Поиск по адресу:                  | <b>Find, Find Using, CommandInfo( )</b>             |
| Поиск по координатам:             | <b>SearchPoint( ), SearchRect( ), SearchInfo( )</b> |
| Информация о таблице:             | <b>NumTables( ), TableInfo( )</b>                   |
| Информация о колонке:             | <b>NumCols( ), ColumnInfo( )</b>                    |
| Информация из метаданных таблицы: | <b>GetMetadata\$( ), Metadata</b>                   |
| Сшитые таблицы:                   | <b>TableInfo( ), GetSeamlessSheet( )</b>            |

## Работа с удаленными базами данных

|                             |   |
|-----------------------------|---|
| Создание новой таблицы:     | <b>Server Create Table</b>                      |
| Связь с сервером:           | <b>Server_Connect( ), Server_ConnectInfo( )</b> |
| Начало работы с сервером:   | <b>Server Begin Transaction</b>                 |
| Локальная область хранения: | <b>Server Bind Column</b>                       |
| Информация о колонке:       | <b>Server_ColumnInfo( ), Server_NumCols( )</b>  |
| Передача SQL-оператора:     | <b>Server_Execute( )</b>                        |
| Позиционировать курсор:     | <b>Server Fetch, Server_EOT( )</b>              |
| Сохранить изменения:        | <b>Сервер</b>                                   |

## Работа с файлами (не таблицами)

|  |  |
|--|--|
| Отменить изменения:                              | Server Rollback                            |
| Освободить ресурсы:                              | Server Close                               |
| Присоединение геоинформации к удаленной таблице: | Server Create Map                          |
| Изменение стилей оформления объектов:            | Server Set Map                             |
| Синхронизация связанной таблицы:                 | Server Refresh                             |
| Создание связанной таблицы:                      | Server Link Table                          |
| Разорвать связь с таблицей:                      | Unlink                                     |
| Отключение от сервера:                           | Server Disconnect                          |
| Информация о драйверах:                          | Server_DriverInfo( ), Server_NumDrivers( ) |
| Получение идентификатора ODBC-соединения:        | Server_GetODBCConn( )                      |
| Получение идентификатора ODBC-оператора:         | Server_GetODBCStmt( )                      |
| Установить стиль объекта:                        | Server Create Style                        |

## Работа с файлами (не таблицами)

### Ввод/Вывод в файлы

|                               |   |
|-------------------------------|---|
| Открыть или создать файл:     | Open File   |
| Закрыть файл:                 | Close File  |
| Удалить файл:                 | Kill  |
| Переименовать файл:           | Rename File   |
| Копировать файл:              | Save File   |
| Восстановить данные из файла: | Get, Seek, Input #, Line Input                      |
| Запись в файл:                | Put, Print #, Write #                               |
| Определить статус файла:      | EOF( ), LOF( ), Seek( ), FileAttr( ), FileExists( ) |

|   |                         |
|---|-------------------------|
| Создание таблицы из другого файла:          | <b>Register Table</b>   |
| Повторить после ошибки совместного доступа: | <b>Set File Timeout</b> |

## Имена файлов и каталогов

|                                 |   |
|---------------------------------|---|
| Сведения о системных каталогах: | <b>ProgramDirectory\$( ), HomeDirectory\$( ), ApplicationDirectory\$( )</b> |
| Выделение части имени файла:    | <b>PathToTableName\$( ), PathToDirectory\$( ), PathToFileName\$( )</b>      |
| Полное имя файла:               | <b>TrueFileName\$( )</b>  |
| Выбор файла пользователем:      | <b>FileOpenDlg( ), FileSaveAsDlg( )</b>                                     |
| Сведения о временном файле:     | <b>TempFileName\$( )</b>  |
| Поиск файлов:                   | <b>LocateFile\$( ), GetFolderPath\$( )</b>                                  |

## Работа с картами и графическими объектами

### Создание объектов на карте

|                              |   |
|------------------------------|---|
| Операторы создания:          | <b>Create Arc, Create Ellipse, Create Frame, Create Line, Create Object, Create PLine, Create Point, Create Rect, Create Region, Create RoundRect, Create Text, AutoLabel, Create Multipoint, Create Collection</b> |
| Функции создания:            | <b>CreateCircle( ), CreateLine( ), CreatePoint( ), CreateText( )</b>  |
| Сложные операции:            | <b>Create Object, Buffer( ), CartesianBuffer( ), CartesianOffset( ), CartesianOffsetXY( ), ConvexHull( ), Offset( ), OffsetXY( ), SphericalOffset( ), SphericalOffsetXY( )</b>                                      |
| Хранение объектов в таблице: | <b>Insert, Update</b>   |
| Создание областей:           | <b>Objects Enclose</b>  |

## Изменение объектов на карте

|                                 |   |
|---------------------------------|---|
| Изменение атрибутов объекта:    | <b>Alter Object</b>   |
| Изменение типа объекта:         | <b>ConvertToRegion( ), ConvertToPLine( )</b>                                |
| Сдвиг объектов:                 | <b>Objects Offset, Objects Move</b>   |
| Изменяемый объект:              | <b>Set Target</b>   |
| Удаление частей объекта:        | <b>CreateCutter, Objects Erase, Erase( ), Objects Intersect, Overlap( )</b> |
| Слияние объектов:               | <b>Objects Combine, Combine( ), Create Object</b>                           |
| Поворот объектов:               | <b>Rotate( ), RotateAtPoint( )</b>  |
| Разрезание объектов:            | <b>Objects Pline, Objects Split</b>   |
| Добавить узлы на пересечениях:  | <b>Objects Overlay, OverlayNodes( )</b>                                     |
| Управление разрешением объекта: | <b>Set Resolution</b>   |
| Помещение объектов в таблицу:   | <b>Insert, Update</b>   |
| Проверка объектов:              | <b>Objects Check</b>  |
| Работа с объектами:             | <b>Objects Disaggregate, Objects Snap, Objects Clean</b>                    |

## Информация об объектах Карты

|  |  |
|--|--|
| Чтение числовых параметров:  | <b>Area( ), Perimeter( ), Distance( ), ObjectLen( ), Overlap( ), AreaOverlap( ), ProportionOverlap( )</b>  |
| Чтение координат:  | <b>ObjectGeography( ), MBR( ), ObjectNodeX( ), ObjectNodeY( ), ObjectNodeZ( ), Centroid( ), CentroidX( ), CentroidY( ), ExtractNodes( ), IntersectNodes( )</b> |
| Чтение параметров координат и единиц измерения расстояний, площадей и листа: | <b>SessionInfo( )</b>  |
| Единицы измерения:   | <b>Set Area Units, Set Distance Units, Set Paper Units, UnitAbbr\$( ), UnitName\$( )</b>   |

|                                |  |
|--------------------------------|--|
| Координатные системы:          | <b>Set CoordSys</b>  |
| Стили оформления объектов:     | <b>ObjectInfo( )</b>   |
| Опрос подписей на слоях карты: | <b>LabelFindByID( ), LabelFindFirst( ),<br/>LabelFindNext( ), LabelInfo( )</b> |

## Работа со стилями объектов

|                                      |   |
|--------------------------------------|---|
| Текущие стили объектов:              | <b>CurrentPen( ), CurrentBorderPen( ),<br/>CurrentBrush( ), CurrentFont( ),<br/>CurrentLinePen( ), CurrentSymbol( ), Set<br/>Style, TextSize( )</b> |
| Атрибуты стилей:                     | <b>StyleAttr( )</b>   |
| Создание стилей:                     | <b>MakePen( ), MakeBrush( ), MakeFont( ),<br/>MakeSymbol( ), MakeCustomSymbol( ),<br/>MakeFontSymbol( ), Set Style, RGB( )</b>                      |
| Выбор объектов по стилям оформления: | <b>ObjectInfo( )</b>  |
| Изменение стиля объекта:             | <b>Alter Object</b>   |
| Восстановление стиля символов:       | <b>Reload Symbols</b>   |
| Выражения для стилей:                | <b>Pen clause, Brush clause, Symbol clause,<br/>Font clause</b>   |

## Работа с окнами карт

|                              |  |
|------------------------------|--|
| Открытие окна карты:         | <b>Map</b>   |
| Создание/изменение 3D-карт:  | <b>Create Map3D, Set Map3D, Map3DInfo( ),<br/>Create PrismMap, Set PrismMap,<br/>PrismMapInfo( )</b> |
| Добавление слоя на карту:    | <b>Add Map</b>   |
| Удаление слоя с карты:       | <b>Remove Map</b>  |
| Подписывание объектов слоя:  | <b>AutoLabel</b>   |
| Информация о Карте и слоях:  | <b>MapperInfo( ), LayerInfo( )</b>   |
| Управление настройкой Карты: | <b>Set Map</b>   |

|  |  |
|--|--|
| Создание и изменение тематического слоя: | <b>Shade, Set Shade, Create Ranges, Create Styles, Create Grid, Relief Shade</b> |
| Опрос подписей на слоях карты:           | <b>LabelFindById( ), LabelFindFirst( ), LabelFindNext( ), Labelinfo( )</b>       |

## Создание элементов интерфейса

### Панели инструментов

|                                     |  |
|-------------------------------------|--|
| Создание новой панели:              | <b>Create ButtonPad</b>                      |
| Изменение панели:                   | <b>Alter ButtonPad</b>                       |
| Изменение кнопки:                   | <b>Alter Button</b>                          |
| Состояние панели:                   | <b>ButtonPadInfo( )</b>                      |
| Определение нажатой кнопки:         | <b>CommandInfo( )</b>                        |
| Восстановление стандартных панелей: | <b>Oneparop Create ButtonPads As Default</b> |

### Диалоги

|  |  |
|--|--|
| Стандартные диалоги:                     | <b>Ask( ), Note, ProgressBar, FileOpenDlg( ), FileSaveAsDlg( ), GetSeamlessSheet( )</b>      |
| Показать новый диалог:                   | <b>Dialog</b>  |
| Обработка нового диалога:                | <b>Alter Control, TriggerControl( ), ReadControlValue( ), Dialog Preserve, Dialog Remove</b> |
| Определение нажатия ОК:                  | <b>CommandInfo(CMD_INFO_DLG_OK)</b>  |
| Подавление вывода индикатора выполнения: | <b>Set ProgressBars</b>  |
| Изменение диалога MapInfo Professional:  | <b>Alter MapInfoDialog</b>   |

## Меню

|                              |   |
|------------------------------|---|
| Создание нового меню:        | Create Menu                                   |
| Переопределение строки меню: | Create Menu Bar                               |
| Изменение меню:              | Alter Menu, Alter Menu Item                   |
| Изменение строки меню:       | Alter Menu Bar, Menu Bar                      |
| Вызов команды из меню:       | Run Menu Command                              |
| Состояние элемента меню:     | MenuItemInfoByHandler( ), MenuItemInfoByID( ) |

## Окна

|                                  |   |
|----------------------------------|---|
| Управление показом окон:         | Open Window, Close Window, Set Window   |
| Открытие новых окон:             | Map, Browse, Graph, Layout, Create Redistricter, Create Legend, Create Cartographic Legend, LegendFrameInfo   |
| Определение идентификатора окна: | FrontWindow( ), WindowID( )   |
| Изменение открытых окон:         | Set Map, Shade, Add Map, Remove Map, Set Browse, Set Graph, Set Layout, Create Frame, Set Legend, Set Cartographic Legend, Set Redistricter, StatusBar, Alter Cartographic Frame, Add Cartographic Frame, Remove Cartographic Frame |
| Чтение информации об окне:       | WindowInfo( ), MapperInfo( ), LayerInfo( )  |
| Печать окна:                     | PrintWin  |
| Управление обновлением окна:     | Set Event Processing, Update Window, Control DocumentWindow clause  |
| Количество окон:                 | NumWindows( ), NumAllWindows( )   |

## Обработчики системных событий

|   |                   |
|---|-------------------|
| Обработка изменения выбора в таблице:         | SelChangedHandler |
| Обработка закрытия окна:                      | WinClosedHandler  |
| Обработка изменения изображения в окне Карты: | WinChangedHandler |

|   |  |
|---|--|
| Обработка изменения фокуса окна:                                | <b>WinFocusChangedHandler</b>                  |
| Реакция на получение DDE-сообщения из другой программы:         | <b>RemoteMsgHandler, RemoteQueryHandler( )</b> |
| Управление OLE Automation                                       | <b>RemoteMapGenHandler</b>                     |
| Проверка использования программ MapBasic:                       | <b>ToolHandler</b>                             |
| Обработка завершения выполнения программы:                      | <b>EndHandler</b>                              |
| Обработка потери/приобретения фокуса окна MapInfo Professional: | <b>ForegroundTaskSwitchHandler</b>             |
| Отмена обработки событий:                                       | <b>Set Handler</b>                             |

## Обмен данными с другими программами

### DDE (Динамический обмен данными)

|  |   |
|--|---|
| Открытие канала DDE-связи:               | <b>DDEInitiate( )</b>   |
| Посылка команды по каналу DDE-связи:     | <b>DDEExecute</b>   |
| Передача значений по каналу DDE-связи:   | <b>DDEPoke</b>  |
| Получение значений по каналу DDE-связи:  | <b>DDERequest\$( )</b>  |
| Закрытие DDE-связи:                      | <b>DDETerminate, DDETerminateAll</b>                                      |
| Обработка приема выполняемого сообщения: | <b>RemoteMsgHandler, RemoteQueryHandler( ), CommandInfo(CMD_INFO_MSG)</b> |

### Интегрированная Картография

|                                  |                               |
|----------------------------------|-------------------------------|
| Вызов окна MapInfo Professional: | <b>Set Application Window</b> |
| Выбор порождающего окна карты:   | <b>Set Next Document</b>      |
| Создание окна легенды:           | <b>Create Legend</b>          |



## Специальные операторы и функции

|  |                           |
|--|---------------------------|
| Запуск другой программы:                       | <b>Run Program</b>        |
| Информация о системе:                          | <b>SystemInfo( )</b>      |
| Выполнение строки интерпретатором:             | <b>Run Command</b>        |
| Сохранение файла рабочего набора:              | <b>Save Workspace</b>     |
| Загрузка файла рабочего набора или .MBX-файла: | <b>Run Application</b>    |
| Настройка дигитайзера:                         | <b>Set Digitizer</b>      |
| Передача звукового сигнала:                    | <b>Beep</b>               |
| Настройка данных, считываемых CommandInfo:     | <b>Set Command Info</b>   |
| Настройка задержки при перетаскивании объекта: | <b>Set Drag Threshold</b> |

## Contacting Technical Support

MapInfo Corporation offers a free support period on all new software purchases and upgrades, so you can be productive from the start. Once the free period ends, MapInfo Corporation offers a broad selection of extended support services for individual, business, and corporate users.

Служба технической поддержки всегда готова помочь Вам. В этом разделе описывается информация, которую Вам потребуется представить при звонке в службу технической поддержки. Здесь также объясняются некоторые технические процедуры, которые помогут Вам в разрешении проблем.

MapInfo Corporation provides full technical support for MapInfo MapBasic for the previous two versions of the product and later. Please remember to include your serial number, partner number or contract number when contacting Technical Support.

contact the technical support personnel for your area:

### *The Americas*

Phone: 518.285.7283

Fax: 518.285.6080

E-mail: techsupport@mapinfo.com

Hours: Monday - Friday from 8:00am - 7:00pm EST, excluding MapInfo Holidays. Closed between 10:30am - 11:30 am on Mondays for training.

### *Asia-Pacific Headquarters*

Phone: 61.7.3844.7744

Fax: 61.7.3844.2400

E-mail: [ozsupport@mapinfo.com](mailto:ozsupport@mapinfo.com)

Hours: Monday - Friday from 9:00am and 5:00pm (EST) Australian Eastern Standard Time, excluding MapInfo Holidays.

### *Europe/Middle East/Africa*

Phone: 44.1753.848229

Fax: 44.1753.621140

E-mail: [support-europe@mapinfo.com](mailto:support-europe@mapinfo.com)

Hours: Monday - Friday from 8am to 5pm GMT, excluding MapInfo Holidays.

### *Germany*

Phone: +49 (0) 6142-203-400

Fax: +49 (0) 6142-203-444

E-mail: [supportgermany@mapinfo.com](mailto:supportgermany@mapinfo.com)

Hours: Monday - Friday from 9 am to 5 pm MEZ, excluding MapInfo Holidays.

To use Technical Support, you must register your product. This can be done very easily during installation. To receive more information on MapInfo's technical support programs, contact a representative in your area or one of our technical support offices.

In the United States, call 1-800-FASTMAP for more information. To purchase MapInfo technical support or renew your current contract, please contact MapInfo Customer Service at 1-800-552-2511, and press 3 at the main menu, or send an e-mail at [custserv@mapinfo.com](mailto:custserv@mapinfo.com).

Extended support options are available at each of our technical support centers in the United States, United Kingdom, and Australia.

## **Прежде чем позвонить**

Пожалуйста, имейте под рукой эту информацию, когда связываетесь со службой технической поддержки MapInfo Professional

1. Серийный номер. Для получения технической поддержки Вы должны иметь зарегистрированный серийный номер.
2. Наименование Вашей организации и имя пользователя. Обращающийся в службу поддержки должен связаться с лицом, указанным в соглашении о поддержке.
3. Номер версии продукта.
4. Наименование и версия операционной системы.

5. Краткое описание сути проблемы. Здесь может быть полезна следующая информация:

- Сообщения об ошибках
- Обстоятельства при которых возникла данная проблема
- Часто или нет возникает подобная проблема?

## Copyright Information

Информация содержащаяся в этом документе может быть изменена без уведомления и не представляет собой обязательств со стороны продавца или его представителей. Никакая часть этого документа не может быть воспроизведена или передана в какой-либо форме любыми средствами, механическими или электронными, включая фотокопирование, без письменного разрешения корпорации MapInfo, One Global View, Troy, New York 12180-8399.

© 2007 Корпорация MapInfo. Все права защищены. MapInfo, логотип MapInfo и MapBasic – торговые марки корпорации MapInfo и/или её филиалов.

Головной офис корпорации MapInfo:

Телефон: (518) 285-6000

Fax: (518) 285-6070

Горячая линия по продажам: (800) 327-8627

Горячая линия по продажам государственным учреждениям: (800) 619-2333

Телефонная линия технической поддержки: (518) 285-7283

Факс технической поддержки: (518) 285-6080

Контактная информация о всех офисах MapInfo находится по адресу:

<http://www.mapinfo.com/contactus>.

Adobe Acrobat® - зарегистрированная торговая марка Adobe Systems Incorporated в США.

Продукты, упоминаемые в этом тексте, могут являться торговыми марками соответствующих производителей, которые этим признаются. Названия зарегистрированных торговых марок используются в этом тексте, по нашему мнению, к пользе их владельцев, без намерения нарушить права на торговую марку.



# Новые и дополненные операторы и функции MapBasic

Здесь описаны новые операторы и функции, появившиеся в MapInfo Professional 9.0.

## В этой главе

- ♦ Новое в MapBasic .....20
- ♦ Дополненные функции и операторы MapBasic .....32

## Новое в MapBasic

Добавлены новые функции, обеспечивающие работу с типами данных: Дата и Время, которые облегчают использование переменных даты и времени в запросах и тематических картах.

Следующие новые функции были добавлены в язык MapBasic для MapInfo Professional версии 9.0. Эти функции описаны в справочнике в алфавитном порядке. Здесь мы повторяем их описание для удобства пользования.

- **Функция CoordSysName\$( )**
- **Функция CurDateTime**
- **Функция CurTime**
- **Функция FormatTime\$**
- **Оператор FME Refresh Table**
- **Функция GetDate**
- **Функция GetTime()**
- **Функция HotlinkInfo()**
- **Функция Hour**
- **Функция MakeDateTime()**
- **Функция Minute**
- **Функция NumberToDateTime()**
- **Функция NumberToTime()**
- **Функция RegionInfo( )**
- **Функция Second**
- **Функция StringToDateTime**
- **Функция StringToTime**

---

### Функция CoordSysName\$( )

#### Назначение

Возвращает строку с названием системы координат из описания системы координат MapBasic.

#### Синтаксис

```
CoordSysName$ ( string )
```

#### Возвращаемая величина

Строка

#### Пример:

```
Note CoordSysName$("Coordsys Earth Projection 1, 62")
```

Показывает в диалоге MapInfo следующую строку:

```
Longitude / Latitude (NAD 27 for Continental US)
```

**Внимание:** Если системы координат с таким названием не существует в файле MAPINFO.PRJ, например, когда план-схема дана в геодезических футах, то эта функция возвратит пустую строку.

```
Note CoordSysName$("CoordSys NonEarth Units " + ""survey ft"" +  
"Bounds (0, 0) (10, 10)")
```

Если параметр CoordSys передан некорректно (заданы неправильные единицы измерения):

```
Note CoordSysName$("CoordSys Earth Projection 3, 74, " + ""foo"" +  
"-90, 42, 42.7333333333, 44.0666666667, 1968500, 0")
```

то будет возвращена ошибка о неправильной системе координат (Ошибка #727).

```
Invalid Coordinate System: CoordSys Earth Projection <content>
```

---

## Функция CurDateTime

### Назначение

Возвращает текущие значения даты и времени.

### Синтаксис

```
CurDateTime
```

### Возвращаемая величина

DateTime

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim X as datetime  
X = CurDateTime()  
Print X
```

---

## Функция CurTime

### Назначение

Возвращает текущее значение времени.

### Синтаксис

```
CurTime
```

### Возвращаемая величина

Время

Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim Y as time
Y = CurTime()
Print Y
```

Функция FormatTime\$

Назначение

Возвращает строку с временем в формате, заданным вторым аргументом. Форматирующая строка должна удовлетворять стандартам Microsoft на настройки местного времени:

| Часы        | Смысл  |
|-------------|--|
| h           | Часы без нулей перед значениями часов (12-часовой-отсчет).   |
| hh          | Часы с нулями перед значениями часов (12-часовой-отсчет).  |
| H           | Часы без нулей перед значениями часов (24-часовой-отсчет).   |
| HH          | Часы с нулями перед значениями часов (24-часовой-отсчет).  |
| Минуты      | Смысл  |
| m           | Минуты без нулей перед значениями минут.   |
| mm          | Минуты с нулями перед значениями минут.  |
| Секунды     | Смысл  |
| s           | Секунды без нулей перед значениями секунд.   |
| ss          | Секунды с нулями перед значениями секунд.  |
| Time marker | Смысл  |
| t           | Строка отметки времени, состоящая из единственного символа.<br><br><b>Внимание:</b> В некоторых языках не применяется, например, японском (Япония). В случае использования такого формата отметка времени, заданная системными параметрами LOCALE_S1159 (AM) и LOCALE_S2359 (PM), сокращается до единственного символа. Поэтому приложение может задать неправильное форматирование, когда одна и та же строка будет использоваться и для значений AM, и для PM. |
| tt          | Строка отметки времени из нескольких символов.   |



Источник: <http://msdn2.microsoft.com/en-us/library/ms776320.aspx>

**Внимание:** В предыдущих форматах буквы m, s и t обязательно должны вводиться в нижнем регистре; буква h в нижнем регистре предназначена для 12-часового отсчета, в то время, как в верхнем регистре указывает на 24-часовой.

Используемый нами код отвесает правилам, установленным для системного местного формата времени. Кроме того, допускается использование f, ff или fff для десятых, сотых долей секунды или миллисекунд.

### Синтаксис

```
FormatTime$ (Time, String)
```

### Возвращаемая величина

Строка

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim Z as time
Z = CurTime()
Print FormatTime$(Z, "hh:mm:ss.fff tt")
```

---

## Оператор FME Refresh Table

### Назначение

Обновляет таблицу Universal Data Source (FME) значениями исходной таблицы

### Синтаксис

```
FME Refresh Table alias
```

где:

*alias* - псевдоним, выбранный для зарегистрированной таблицы Universal Data Source (FME).

### Пример:

Следующий пример демонстрирует обновление локальной таблицы под названием watershed.

```
FME Refresh Table watershed
```

---

## Функция GetDate

### Назначение

Возвращает компоненту даты DateTime.

### Синтаксис

`GetDate (DateTime)`

### Возвращаемая величина

Дата типа `Date`

**Пример:**

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim dtX as datetime
dim Z as date
dtX = "07.03.07 12:09:09.000 AM"
Z = GetDate(dtX)
Print FormatDate$(Z)
```

---

**Функция GetTime()****Назначение**

Возвращает компоненту времени DateTime.

**Синтаксис**

```
GetTime (DateTime)
```

**Возвращаемая величина**

Время

**Пример:**

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim dtX as datetime
dim Z as time
dtX = "07.03.07 12:09:09.000 AM"
Z = GetTime(dtX)
Print FormatTime$(Z, "hh:mm:ss.fff tt")
```

---

**Функция HotlinkInfo()****Назначение**

Возвращает информацию о геолинке на карте.

**Синтаксис**

```
HotlinkInfo ( map_window_id, layer_number, hotlink_number, attribute )
```

*map\_window\_id* – идентификатор окна Карты;

*layer\_number* номер слоя (1 – самый верхний слой); чтобы определить номер слоя в окне Карты, используется **Функция MapperInfo( )**.

*hotlink\_number* – индекс запрашиваемого геолинка. Первое определение геолинка на карте имеет номер индекса 1.

## Функция Hour

---

*attribute* - допускаются следующие значения атрибута:

|                       |   |
|-----------------------|---|
| HOTLINK_INFO_EXPR     | Возвращает выражение для файла, на который указывает геолинк.   |
| HOTLINK_INFO_MODE     | Возвращает текущий режим геолинка, одно из следующих predefined значений: <ul style="list-style-type: none"><li>HOTLINK_MODE_LABEL</li><li>HOTLINK_MODE_OBJ</li><li>HOTLINK_MODE_BOTH</li></ul> |
| HOTLINK_INFO_RELATIVE | Возвращает "Да" (TRUE), если маршрут в геолинке задан в относительной форме.  |
| HOTLINK_INFO_ENABLED  | Возвращает "Да" (TRUE), если геолинк активирован.   |

См. также:

[Оператор Set Map](#), [Функция LayerInfo\( \)](#),

---

## Функция Hour

### Назначение

Возвращает часовую компоненту времени

### Синтаксис

Hour (Time)

### Возвращаемая величина

Номер

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim Z as time
dim iHour as integer
Z = CurDateTime()
iHour = Hour(Z)
Print iHour
```

## Функция MakeDateTime()

### Назначение

Возвращает значение DateTime, составленное из заданных значений Date и Time.

### Синтаксис

```
MakeDateTime (Date, Time)
```

### Возвращаемая величина

DateTime

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim tX as time
dim dX as date
dim dtX as datetime
tX = 105604123
dX = 20070908
dtX = MakeDateTime(dX,tX)
Print FormatDate$(GetDate(dtX))
Print FormatTime$(GetTime(dtX), "hh:mm:ss.fff tt")
```

---

## Функция Minute

### Назначение

Возвращает минуты.

### Синтаксис

```
Minute (Time)
```

### Возвращаемая величина

Номер

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim X as time
dim iMin as integer
X = CurDateTime()
iMin = Minute(X)
Print iMin
```

### Функция NumberToDateTime()

#### Назначение

Returns a DateTime value.

#### Синтаксис

NumberToDateTime( numeric\_datetime )

*numeric\_datetime* – семнадцатизначное целое число типа Integer в форме ГодГодГодГодМесМесДеньДеньЧасЧасМинМинСекСекДоляДоляДоля (YYYYMMDDHHMMSSFFF). Например, 20070301214237582 представляет 1 Марта 2007 г. 9:42:37.582 PM.

#### Возвращаемая величина

Date/Time

#### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim fNum as float
dim Y as datetime
fNum = 20070301214237582
Y = NumberToDateTime (fNum)
Print FormatDate$(Y)
Print FormatTime$(Y, "hh:mm:ss.fff tt")
```

### Функция NumberToTime()

#### Назначение

Returns a Time value.

#### Синтаксис

NumberToTime( numeric\_time )

*numeric\_datetime* – девятизначное целое число типа Integer в форме ЧасЧасМинМинСекСекДоляДоляДоля (HHMMSSFFF). Например, 214237582 представляет 21:42:37.582 (или 9:42:37.582 P.M.)

#### Возвращаемая величина

Время

**Пример:**

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim fNum as float
dim Y as time
fNum = 214237582
Y = NumberToTime(fNum)
Print FormatTime$(Y, "hh:mm:ss.fff tt")
```

---

**Функция RegionInfo( )****Назначение**

Эта функция предназначена для определения направления обхода узлов полигона – по-часовой или против часовой. Единственный атрибут, который возвращает эта функция, – 'направление обхода' узлов заданного полигона.

**Синтаксис**

```
RegionInfo(object, REGION_INFO_IS_CLOCKWISE, polygon_num)
```

Существует единственный параметр этой функции:

|                          |   |
|--------------------------|---|
| REGION_INFO_IS_CLOCKWISE | 1 |
|--------------------------|---|

**Пример:**

Если на карте Соединенных Штатов Америки "STATES.TAB" выбрать штат Юта (Utah) и выполнить в окне MapBasic эту команду, то результатом будет F or False, поскольку узлы единственного полигона штата Юта имеют направление обхода против часовой. Узлы полигона штата Колорадо (Colorado) имеют направление обхода по-часовой и, в этом случае, будет возвращено значение T или True.

```
print RegionInfo(selection.obj,1,1)
```

---

**Функция Second****Назначение**

Возвращает секунды и доли секунд в виде вещественного числа.

**Синтаксис**

```
Second (Time)
```

**Возвращаемая величина**

Номер

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim X as time
dim iSec as integer
X = CurDateTime()
Sec = Second(X)
Print iSec
```

---

## Функция StringToDateTime

### Назначение

Возвращает значение типа DateTime (Дата/Время) на основе сведений из строки, в которой перечислены дата и время. MapBasic интерпретирует данные типа DateTime в соответствии с форматами, заданными в компьютере, где работает программа. Между датой и временем требуется хотя-бы один пробел. Подробнее см. разделы [Функция StringToDate\( \)](#) и [Функция StringToTime](#).

### Синтаксис

```
StringToDateTime (String)
```

### Возвращаемая величина

DateTime

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim strX as string
dim Z as datetime
strX = "19990912041345789"
Z = StringToDateTime(strX)
Print FormatDate$(Z)
Print FormatTime$(Z, "hh:mm:ss.fff tt")
```

---

## Функция StringToTime

### Назначение

Возвращает значение типа Time (Время) на основе сведений из строки с временем. MapBasic интерпретирует данные типа Time в соответствии с форматом, заданным в компьютере, где работает программа. При этом допускается либо 12-часовой, либо 24-часовой отсчет времени. Кроме того, малозначащие компоненты времени можно опустить. Другими словами, необходимо задавать часы, а минуты секунды и миллисекунды использовать необязательно.

### Синтаксис

```
StringToTime (String)
```



### **Возвращаемая величина**

Время

### **Пример:**

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim strY as string
dim X as time
strY = "010203000"
X = StringtoTime(strY)
Print FormatTime$(X,"hh:mm:ss.fff tt")
```

## Дополненные функции и операторы MapBasic

Следующие функции языка MapBasic были дополнены новыми параметрами.

- Оператор Commit Table
- Функция GeocodeInfo( )
- Функция IsogramInfo( )
- Функция LabelInfo( )
- Функция LayerInfo( )
- Оператор Register Table
- Оператор Server Create Table
- Оператор Set Map
- Функция SystemInfo
- Функция TableInfo( )

---

### Оператор Commit Table

#### Назначение

Сохраняет последнюю редакцию таблицы на диске или сохраняет ее копию. Теперь её можно использовать при обращении к данным типа Дата/Время.

#### Синтаксис

```
Commit Table table
[ As параметры_файла
  [ Type { NATIVE |
    DBF [ Charset набор_символов ] |
    Параметры базы данных Access параметры_файла_базы_данных
    Version номер_версии
    Table имя_таблицы
    [ Password пароль ] [ Charset набор_символов ] |
    QUERY |
    ODBC Connection номер_соединения Table имя_таблицы
    [ ConvertDateTime {ON | OFF | INTERACTIVE}} ] } ]
  [ CoordSys... ]
  [ Version номер_версии ] ]
[ { Interactive | Automatic commit_keyword } ]
[ ConvertObjects {ON | OFF | INTERACTIVE}} ]
```

*tableName* – строковая переменная, определяющая имя таблицы, которая появится в Access; Начиная с версии 9.0, имя таблицы может включать в себя имя схемы, которой принадлежит таблица. Если имени схемы не задано, то таблица принадлежит стандартной схеме, подобно тому как было в версии 8.5 и более ранних. Пользователь должен задать правильное имя схемы, должен знать имя учетной записи и обладать правами доступа к указанной схеме. Это касается только SQL Server 2005.

*ConvertDateTime* – если в исходной таблице существуют колонки любого из типов Time или Date, то, в зависимости от того, какой тип поддерживает сервер, они будут преобразованы в DATETIME или TIMESTAMP. Однако, этим преобразованием можно управлять, используя специальный параметр *ConvertDateTime*. Если в исходной таблице нет колонок любого из этих типов, то этот параметр не используется. Если параметру *ConvertDateTime* присвоено значение ON (режим по умолчанию), то колонки типов Time или Date будут преобразованы в DATETIME или TIMESTAMP. Если параметру *ConvertDateTime* присвоено значение OFF, то преобразований не будет, а операция будет при необходимости завершена. Если параметру *ConvertDateTime* присвоено значение INTERACTIVE, то появится диалог, в котором пользователю будет предложено выбрать необходимое действие. Если пользователь выберет преобразование, то тип будет преобразован, а если пользователь выберет отмену, то операция будет прекращена.

Тип Time требует преобразования всех доступных серверов (Oracle, IBM Informix, MS SQL Server и Access), а тип Date потребует преобразований только для баз данных серверов MS SQL Server и Access.

**Внимание:** Для серверов баз данных MS SQL Server и Access, это может вызвать проблемы совместимости с предыдущими версиями данных. Ранее такие преобразования выполнялись без предупреждения. В этой версии мы советуем использовать тип данных DateTime вместо типа данных Date. Если по-прежнему будет использоваться тип данных Date, то преобразование выполнено не будет.

*ConvertObjects ON* – автоматически конвертирует любые неподдерживаемые объекты, которыми обнаруживает в поддерживаемых объектах.

*ConvertObjects OFF* – в этом режиме не конвертируются никакие неподдерживаемые объекты. Если такие объекты встретятся, то будет выдано сообщение об ошибке – о том, что таблица не может быть сохранена. (До введения этой функции это был единственный вариант.)

*ConvertObjects Interactive* – если в таблице встретится неподдерживаемый объект, то пользователя может выбрать, что надо сделать.

### Пример 1

```
Commit Table DATETIME90 As "D:\MapInfo\Data\Remote\DATETIME90CPY.TAB"
Type ODBC Connection 1 Table ""EASYLOADER"". "DATETIME90CPY"
ConvertDateTime Interactive
```

Пример 2

```
Server 1 Create Table ""EASYLOADER"". "CITY_125AA" (Field1
Char(10),Field2 Char(10),Field3 Char(10),MI_STYLE Char(254)) KeyColumn
SW_MEMBER ObjectColumn SW_GEOMETRY
Or (оператор Или)
Server 1 Create Table "EASYLOADER.CITY_125AA" (Field1 Char(10),Field2
Char(10),Field3 Char(10),MI_STYLE Char(254)) KeyColumn SW_MEMBER
ObjectColumn SW_GEOMETRY
```

```
Commit Table City_125aa As
"C:\Projects\Data\TestScripts\English\remote\City_125aacpy.tab" Type ODBC
Connection 1 Table ""EASYLOADER"". "CITY_125AACPY"
Or (оператор Или)
```

## Функция **GeocodeInfo( )**

---

```
Commit Table City_125aa As
"C:\Projects\Data\TestScripts\English\remote\City_125aacpy.tab" Type ODBC
Connection 1 Table "EAZYLOADER.CITY_125AACPY"
```

## Функция **GeocodeInfo( )**

---

### Назначение

We have added a new attribute to this function to handle the maximum number of addresses that the server will permit to be sent to the service at a time.

### Синтаксис

**GeoCodeInfo**( *connection\_handle*, *attribute* )

*connection\_handle* – целое.

*attribute* это целое, определяющее, какого типа информация будет возвращена.

### Возвращаемая величина

Вещественное (Float), целое (Integer), короткое целое (SmallInt), логическое (Logical), или строка символов (String), в зависимости от параметра атрибута.

### Описание

Функция **GeoCodeInfo( )** возвращает характеристики соединения, используемые по умолчанию, или параметры, заданные оператором **Set GeoCode**. Подобно многим другим функциям MapBasic этого типа, возвращаемые значения могут зависеть от параметра *attribute*. Коды всех значений перечислены в MAPBASIC.DEF.

### **GEOCODE\_MAX\_BATCH\_SIZE**

целое, определяющее максимальное количество записей (адресов), которое может единовременно передаваться службе для обработки.

### Пример:

Этот фрагмент кода MapBasic печатает в окне сообщений MapInfo Professional сведения от сервиса Envinsa Location Utility Constraints:

```
Include "MapBasic.Def"
declare sub main
sub main
dim iConnect as integer

Open Connection Service Geocode Envinsa
URL
"http://envinsa_server:8066/LocationUtility/services/LocationUtility"
User "john"
Password "green"
into variable iConnect

Print "Geocode Max Batch Size: " +
GeoCodeInfo(iConnect,GEOCODE_MAX_BATCH_SIZE)
```

end sub

## Функция IsogramInfo( )

### Назначение

В эту функцию включены новые атрибуты, обеспечивающие обращение с максимальным числом записей на сервере, данными о времени и расстоянии.

### Синтаксис

**IsogramInfo**( *connection\_handle*, *attribute* )

*connection\_handle* – целое, определяющее число соединений, которое возвращает **Оператор Open Connection**.

*attribute* это целое, определяющее, какого типа информация будет возвращена.

### Возвращаемая величина

Вещественное (Float), логическое (Logical), или строка символов (String), в зависимости от параметра *атрибута*.

### Описание

Эта функция возвращает стандартные параметры соединения или, если применялась **Оператор Set Connection Isogram**, заданные ею параметры.

Существует несколько атрибутов, которые **IsogramInfo( )** может вернуть. Коды определены в MAPBASIC.DEF.

| Параметры attribute        | IsogramInfo Return Value  |
|----------------------------|---|
| ISOGRAM_MAX_BATCH_SIZE     | целое, определяющее максимальное количество записей (например, точек), которое можно одновременно передавать службе для обработки.                              |
| ISOGRAM_MAX_BANDS          | целое число – максимальное число зон времени или расстояний.  |
| ISOGRAM_MAX_DISTANCE       | вещественное, определяющее максимальную удаленность зоны запроса расстояний. расстояние измеряется в единицах, указанных параметром ISOGRAM_MAX_DISTANCE_UNITS. |
| ISOGRAM_MAX_DISTANCE_UNITS | строка задающая единицы измерения, используемые в ISOGRAM_MAX_DISTANCE.   |

## Функция Labelinfo( )

|                        |   |
|------------------------|---|
| ISOGRAM_MAX_TIME       | вещественное, определяющее максимальную удаленность по времени зоны запроса расстояний. Время измеряется в единицах, указанных параметром ISOGRAM_MAX_TIME_UNITS. |
| ISOGRAM_MAX_TIME_UNITS | строка, задающая единицы времени, используемые в ISOGRAM_MAX_TIME.  |

### Пример:

Следующий фрагмент кода MapBasic печатает в окне сообщений MapInfo Professional сведения от Envinsa Routing Constraints:

```
Include "MapBasic.Def"
declare sub main
sub main
dim iConnect as integer
Open Connection Service Isogram
    URL "http://envinsa_server:8062/Route/services/Route"
    User "john"
    Password "green"
    into variable iConnect

Print "Isogram_Max_Batch_Size: " +
IsogramInfo(iConnect,Isogram_Max_Batch_Size)
Print "Isogram_Max_Bands: " + IsogramInfo(iConnect, Isogram_Max_Bands)
Print "Isogram_Max_Distance: " + IsogramInfo(iConnect,
Isogram_Max_Distance)
Print "Isogram_Max_Distance_Units: " + IsogramInfo(iConnect,
Isogram_Max_Distance_Units)
Print "Isogram_Max_Time: " + IsogramInfo(iConnect,Isogram_Max_Time)
Print "Isogram_Max_Time_Units: " +
IsogramInfo(iConnect,Isogram_Max_Time_Units)
Close Connection iConnect
end sub
```

## Функция Labelinfo( )

### Назначение

Возвращает информацию о подписи на Карте. LabelInfo возвращает из подписи текстовый объект. Однако, если подписи выполнены вдоль кривой, то они будут возвращены как повернутый текст вдоль прямой.

### Синтаксис

```
Labelinfo( map_window_id, layer_number, attribute )
```

### Описание

Параметр attribute должен принимать одно из следующих значений в таблице; коды определены в MAPBASIC.DEF.

| Значения attribute     | Функция Labelinfo( ) возвращает  |
|------------------------|--|
| LABEL_INFO_ORIENTATION | <p>Возвращает короткое целое, описывающее ориентацию подписи. Подпись создается одной из следующих функций: LabelFindFirst, LabelFindByID или LabelFindNext. Результатом может быть один из следующих кодов:</p> <ul style="list-style-type: none"><li>• LAYER_INFO_LABEL_ORIENT_HORIZONTAL (подпись под углом равным 0)</li><li>• LAYER_INFO_LABEL_ORIENT_PARALLEL (подпись под любым углом отличным от 0)</li><li>• LAYER_INFO_LABEL_ORIENT_CURVED (подпись по кривой)</li></ul> |

Функция LayerInfo( )

Назначение новых атрибутов

Новый атрибут LAYER\_INFO\_HOTLINK\_COUNT позволяет задавать несколько геолинков на слое карты.

Для обеспечения обратной совместимости, поддерживается старый набор атрибутов, но при этом будет возвращаться определение первого геолинка на слое. Если геолинки не заданы, то эта функция будет возвращать следующие значения:

LAYER\_INFO\_HOTLINK\_EXPR – пустая строка ("" )

LAYER\_INFO\_HOTLINK\_MODE – возвращает принятое по умолчанию значение HOTLINK\_MODE\_LABEL

LAYER\_INFO\_HOTLINK\_RELATIVE – возвращает значение по умолчанию FALSE

Новый атрибут:

LAYER\_INFO\_LABEL\_ORIENTATION – возвращает короткое целое, описывающее ориентацию автоматически создаваемых подписей.. Результатом будет один из следующих кодов:

LAYER\_INFO\_LABEL\_ORIENT\_HORIZONTAL (подписи под углом равным 0)

LAYER\_INFO\_LABEL\_ORIENT\_PARALLEL (подписи под любым углом отличным от 0)

LAYER\_INFO\_LABEL\_ORIENT\_CURVED (подписи по кривой)

**Внимание:** Если возвращается LAYER\_INFO\_LABEL\_ORIENT\_PARALLEL, то LAYER\_INFO\_LABEL\_PARALLEL возвращает TRUE.

## Оператор Register Table

### Назначение

Создаёт таблицу MapInfo Professional из электронных таблиц, баз данных, текстовых файлов, растровых изображений и файлов регулярных поверхностей.

### Синтаксис

```
Register Table source_file
{ Type "NATIVE" |
  Type "DBF" [ CharSet char_set ] |
  Type "ASCII" [ Delimiter delim_char ][ Titles ][ CharSet char_set ] |
  Type "WKS" [ Titles ] [ Range range_name ] |
  Type "WMS" Coordsys...
  Type "WFS" [ CharSet char_set ] Coordsys... [ Symbol... ]
    [ Linestyle Pen(...) ] [ Regionstyle Pen(...) Brush(...) ]
  Type "XLS" [ Titles ] [ Range range_name ] [ Interactive ] |
  Type "Access" Table table_name [ Password pwd ] [ CharSet char_set ]}
Type ODBC
  Connection { Handle ConnectionNumber | ConnectionString }
  Toolkit toolkitname
  Cache { ON | OFF }
  [ Autokey { ON | OFF }}
  Table SQLQuery
  [ Versioned { ON | OFF }}
  [ Workspace WorkspaceName ]
  [ ParentWorkspace ParentWorkspaceName ]
Type "GRID" | Type "RASTER"
  [ ControlPoints ( MapX1, MapY1 ) ( RasterX1, RasterY1 ),
    ( MapX2, MapY2 ) ( RasterX2, RasterY2 ),
    ( MapX3, MapY3 ) ( RasterX3, RasterY3 )
    [, ... ]
  ]
  [ CoordSys ... ]
Type "FME" [ CharSet char_set ]
  CoordSys...
  Format format type
  Schema featuretype
  [ Use Color ]
  [ Database ]
  [ SingleFile ]
  [ Symbol...]
  [ Linestyle Pen(...) ]
  [ Regionstyle Pen(...) Brush(...) ]
  [ Font ... ]
  Settings string1 [, string2 .. ]
Type "SHAPEFILE" [ CharSet char_set ] CoordSys...
  [ PersistentCache { ON | OFF } ]
  [ Symbol...] [ Linestyle Pen(...) ]
  [ Regionstyle Pen(...) Brush(...) ]
```



[ Into destination\_file ]

*Charset char\_set* – задавать этот параметр необязательно. Если набор символов не задан MapBasic'ом, то будет использован набор символов системы. Подобный метод используется и для остальных форматов.

*CoordSys...* – обязательный параметр CoordSys.

*Format formattype* – строка formattype используется FME для определения формата открываемых данных.

*Schema featurtype* – задает тип свойств (по существу – название схемы).

*Settings string1 [ , string2 .. ]* – настройки Safe Software FME, которые могут меняться в зависимости от выбранного формата и необходимых преобразований.

*Use Color* – включает использование сведений о цвете из набора данных

*Database* – указывает, что исходные данные хранятся в базе данных

*SingleFile* – указывает, что исходные данные хранятся в единственном файле

### Пример:

```
Register Table "D:\MUT\DWG\Data\afrika_miller.DWG" Type "FME" CoordSys
Earth Projection 11, 104, "m", 0 Format "ACAD" Schema "afrika_miller" Use
Color SingleFile Symbol (35,0,16) Linestyle Pen (1,2,0) RegionStyle Pen
(1,2,0) Brush (2,16777215,16777215) Font ("Arial",0,9,0) Settings
"RUNTIME_MACROS", "METAFILE,acad,_EXPAND_BLOCKS,yes,ACAD_IN_USE_BLOCK_HEAD
ER_LAYER,yes,ACAD_IN_RESOLVE_ENTITY_COLOR,yes,_EXPAND_VISIBLE,yes,_BULGES
_AS_ARCS,no,_STORE_BULGE_INFO,no,_READ_PAPER_SPACE,no,ACAD_IN_READ_GROUPS
,no,_IGNORE_UCS,no,_ACADPreserveComplexHatches,no,_MERGE_SCHEMAS,YES",
"META_MACROS", "Source_EXPAND_BLOCKS,yes,SourceACAD_IN_USE_BLOCK_HEADER_LA
YER,yes,SourceACAD_IN_RESOLVE_ENTITY_COLOR,yes,Source_EXPAND_VISIBLE,yes,
Source_BULGES_AS_ARCS,no,Source_STORE_BULGE_INFO,no,Source_READ_PAPER_SPA
CE,no,SourceACAD_IN_READ_GROUPS,no,Source_IGNORE_UCS,no,Source_ACADPreser
veComplexHatches,no", "METAFILE", "acad", "COORDSYS", "", "IDLIST", "" Into
"C:\Temp\afrika_miller.tab"
Open table "C:\Temp\afrika_miller.tab"
Map From "afrika_miller"
```

---

## Оператор Server Create Table

Специальных синтаксических изменений при использовании нового типа данных нет. Существует следующие ограничения некоторых типов данных:

В версии 9.0 появились два новых типа данных MapInfo: Time (Время) и DateTime (Дата/Время). Однако, в большинство баз данных не включены соответствующие типы СУБД. Раньше мы обеспечивали работу только с типами данных Date (Дата). Даже тип Date преобразовывался в тип данных сервера, если сервер не поддерживал тип данных Date. Начиная с версии 9.0, этот оператор поддерживает только те типы данных, которые поддерживает выбранный сервер. Поэтому тип данных Time запрещен к использованию в этом операторе при обращении ко всем поддерживаемым серверам (Oracle, IBM Informix, MS

SQL Server и Access), а тип данных Date запрещен для MS SQL Server и Access. Если необходимо хранить в таблице базы данных сведения о времени в отдельной колонке, то эти "неподдерживаемые" типы следует заменять типом данных DateTime.

**Внимание:** Для серверов MS SQL Server и Access это может вызвать проблемы совместимости с предыдущими версиями. Ранее такие преобразования выполнялись в фоновом режиме. Теперь, для версии 9.0, необходимо вместо типа данных DATE использовать DATETIME. Если по-прежнему использовать тип данных DATE, то операция не будет выполнена.

```
Server ConnectionNumber Create Table TableName (ColumnName ColumnType [,...])
    [KeyColumn ColumnName]
    [ObjectColumn ColumnName]
    [StyleColumn ColumnName]
    [CoordSys... ]
```

*Tablename* – строковая переменная имени таблицы базы данных; Начиная с версии 9.0, имя таблицы может включать в себя имя схемы, которой принадлежит таблица. Если имени схемы не задано, то таблица принадлежит стандартной схеме, подобно тому как было в версии 8.5 и более ранних. Пользователь должен задать правильное имя схемы, должен знать имя учетной записи и обладать правами доступа к указанной схеме. Это касается только SQL Server 2005.

---

## Оператор Set Map

### Изменено предложение LabelClause

Теперь можно создавать подписи по кривой.

#### Синтаксис

LABELCLAUSE имеет следующий синтаксис:

```
[ Label [ Line { Simple | Arrow | None } ]
  [ Position [ Center ] [ Above | Below ] [ Left | Right ] ] ]
[ Font... ] [ Pen... ]
[ With label_expr ]
[ Parallel { On | Off } | Follow Path ]
[ Visibility { On | Off | Zoom( min_vis, max_vis ) [ Units dist_unit ] } ]
[ Auto [ { On | Off } ] ]
[ Overlap [ { On | Off } ] ]
[ PartialSegments { On | Off } ]
[ Duplicates [ { On | Off } ] ]
[ Max [ number_of_labels ] ]
[ Offset offset_amount ]
[ Default ]
[ Object ID
  [ Table alias ]
  [ Visibility { On | Off } ]
  [ Anchor ( anchor_x, anchor_y ) ]
  Text text_string
  [ Position [ Center ] [ Above | Below ] [ Left | Right ] ] ]
```

```

[ Font... ] [ Pen... ]
[ Line { Simple | Arrow | None } ]
[ Angle text_angle | Follow Path ]
[ Offset offset_amount ]
[ Callout ( callout_x, callout_y ) ] }
[ Object... ]
]

```

*Parallel* – можно использовать со следующими атрибутами:

*Parallel Off* = создать горизонтальные подписи без поворота вдоль линии

*Parallel On* = создать подписи вдоль линии

*Follow Path* = создать подпись вдоль кривой, которая будет обчислена, а её параметры будут сохранены

### Пример:

```
Set Map Layer 1 Label Follow Path
```

## Предложение Layer Activate

Параметры геолинков можно настраивать с помощью Set Map Layer Activate, который позволяет обрабатывать несколько определений геолинка. Можно добавлять новые элементы, изменять атрибуты существующих, удалять их и изменять порядок обращения к элементам. Ниже описаны преимущества использования предложения Activate, включая подробности синтаксиса. Смотрите раздел: **Исключения, обеспечивающие совместимость с предыдущими версиями**, в котором описано, как MapBasic обращается с прежним вариантом синтаксиса.

Обратите внимание, что свойства индивидуального геолинка остались такими же как и в прежних версиях, кроме нового параметра **Enable**, с помощью которого можно выключить активность геолинка, оставив его определение. (Ранее геолинк можно было отключить, только задав "" в его определении и потеряв исходное определение).

### Назначение

С помощью параметра **Activate** можно определять новые геолинки. Геолинки используются для того чтобы открывать файлы или новые адреса URL в Интернет-браузере прямо из окна Карты.

### Синтаксис

```

{ Using launch_expr[ On { [ [ Labels ] | [ Objects ] ] } ] | [ Relative
Path { On | Off } ] [ Enable { On | Off } ] ],
[ Using launch_expr[ On { [ [ Labels ] | [ Objects ] ] } ] | [ Relative
Path { On | Off } ] [ Enable { On | Off } ] ]...

```

### Пример:

```
Set Map Layer 1 Activate Using Url1 On Objects Relative Path Off Enable
On, Using Url2 On Objects Relative Path On Enable On
```

### Замечания

Эта версия команды уничтожит существующее определение, но создаст одно или несколько новых. Следует обязательно использовать параметр **Using**, не допуская пустых строк в `launch_expr` (например, ""). Если используется параметр `Enable`, а его значение установлено в состояние `Off`, то определение геолинка будет выключено. Параметры `On`, `Relative` и `Enable` можно опустить.

## Как добавить новые определения геолинка

### Синтаксис

```
Activate Add [ First ]
{ Using launch_expr[ On { [ [ Labels ] | [ Objects ] ] } ] | [ Relative
  Path { On | Off } ] [ Enable { On | Off } ] },
[ Using launch_expr[ On { [ [ Labels ] | [ Objects ] ] } ] | [ Relative
  Path { On | Off } ] [ Enable { On | Off } ] ]...
```

### Примеры

```
Activate Add Using URL1 On Objects Relative Path On, Using URL2 On Objects
Enabled Off
Activate Add First Using URL1 On Objects
```

### Комментарии

Версия этой команды, применяемая в MapBasic 9.0, добавит новое определение геолинка в конец списка определений.

Если включить в состав команды дополнительный параметр *First*, то новые определения будут добавлены в начало списка.

Если используется параметр *Enable*, а его значение установлено в состояние *Off*, то определение геолинка будет выключено.

Параметры *On*, *Relative* и *Enable* можно опустить.

## Как изменить существующие определения геолинка

### Синтаксис

```
Activate Modify
{ hotlink_id[ Using launch_expr ] | [ On { [ [ Labels ] | [ Objects ] ] } ]
| [ Relative Path { On | Off } ] [ Enable { On | Off } ] },
[ hotlink_id[ Using launch_expr ] | [ On { [ [ Labels ] | [ Objects ] ] } ]
| [ Relative Path { On | Off } ] [ Enable { On | Off } ] ]...
```

### Примеры

```
Activate Modify 1 Using URL1 On Objects, 2 Relative Path Off
Activate Modify 2 On Objects, 4 On Labels
Activate Modify 3 Relative Path On Enable Off
Activate Modify 2 Enable Off, 3 Enable On
```

### Комментарии

`hotlink_id` – целочисленный указатель, с помощью которого можно задать определение геолинка

Требуется использовать хотя-бы один параметр из *Using/On/Relative/Enabled*.

*launch\_expr* – не может быть пустой строкой (например, "").

Если используется параметр *Enable*, а его значение установлено в состояние *Off*, то определение геолинка будет выключено.

## Как удалять определения геолинка

### Синтаксис

```
Activate Remove { All | hotlink_id [ , hotlink_id, hotlink_id, ... ] }
```

### Примеры

```
Activate Remove 2, 4  
Activate Remove All
```

### Комментарии

`hotlink_id` – целочисленный указатель, с помощью которого можно задать определение геолинка

Необходимо задать хотя бы один *hotlink\_id*.

Если используется параметр *All*, то будут удалены все определения геолинка.

## Как изменить последовательность определений геолинка

### Синтаксис

```
Activate Order { hotlink_id [ , hotlink_id, hotlink_id, ... ] }
```

### Примеры

```
Activate Order 2, 3, 1
```

### Комментарии

`hotlink_id` – целочисленный указатель, с помощью которого можно задать определение геолинка

Необходимо задать хотя бы один *hotlink\_id*.

### Исключения, обеспечивающие совместимость с предыдущими версиями

Параметр **Using** можно опустить, но только для первого определения геолинка. Связанное с параметром **Using** выражение может быть пустым (""), но только для первого определения геолинка.

---

## Функция SystemInfo

### Назначение

В этой функции появился атрибут, с помощью которого MapBasic может выяснить номер версии MapInfo Professional. Например, SystemInfo(SYS\_INFO\_MIVERSION) возвращает 850 для MapInfo Professional 8.5 и 8.5.2. Отличить версию 8.5 от 8.5.2 можно с помощью SystemInfo(SYS\_INFO\_MIBUILD\_NUMBER); здесь будет возвращено 32 для 8.5 и 60 для 8.5.2.

### Синтаксис

```
SystemInfo(SYS_INFO_MIBUILD_NUMBER)
```

**Внимание:** Константа, определенная для этой функции в файле MapBasic.def равна 18.

---

## Функция TableInfo( )

### Назначение

Возвращает информацию об открытой таблице. Добавлен определитель таблиц FME.

### Синтаксис

```
TableInfo( table_id, attribute ), где
```

*table\_id* – либо целочисленный номер таблицы, либо имя таблицы в кавычках, либо 0;

*attribute* – целочисленный код, определяющий тему запроса.

### Возвращаемая величина

Строка, логическое значение, целое или короткое целое число. Величина типа String, Integer, SmallInt или Logical, в зависимости от значения параметра attribute.

Второй параметр *attribute* определяет вид информации о данной таблице MapInfo, которая будет получена. Коды в левом столбце (например, TAB\_INFO\_NAME) определены в файле стандартных определений MAPBASIC.DEF.

| Значения<br>attribute | TableInfo( ) возвращает:  |
|-----------------------|---|
| TAB_INFO_TYPE         | <p>Целое число (тип SmallInt), код, Результатом может быть один из следующих кодов:</p> <ul style="list-style-type: none"><li>• TAB_TYPE_BASE, если таблица нормальная;</li><li>• TAB_TYPE_RESULT, если таблица запроса;</li><li>• TAB_TYPE_IMAGE, если таблица растровая;</li><li>• TAB_TYPE_VIEW, если таблица является представлением (view), например, таблица STREETINFO является представлением;</li><li>• TAB_TYPE_LINKED, если таблица связанная.</li><li>• TAB_TYPE_WMS (если таблица Web Map Service).</li><li>• TAB_TYPE_WFS (если таблица Web Feature Service).</li><li>• TAB_TYPE_FME (если таблица была открыта с помощью FME).</li></ul> |





# Алфавитный справочник языка MapBasic

В этом разделе детально описываются операторы и функции языка MapBasic. Они расположены по алфавиту. При описании используются следующие блоки:

## Назначение

Краткое описание функции или оператора.

## Предупреждение

Информация об ограничениях (например, “Функция DDEInitiate доступна только в среде Microsoft Windows”, “Вы не можете использовать оператор **For...Next** в окне MapBasic”).

## Синтаксис

Формат, в котором применяется функция или оператор и объяснение аргументов.

## Возвращаемая величина

Тип возвращаемого функцией значения.

## Описание

Подробное описание функции или оператора и прочая нужная информация.

## Пример:

Короткий пример.

Родственные функции и операторы. Большинство операторов MapBasic можно запускать прямо в среде MapInfo Professional, вводя их в окно MapBasic. Если оператор нельзя запустить через окно MapBasic, об этом будет сказано в разделе Ограничения. Как правило, операторы управления программой (такие как циклы и переходы) нельзя вводить через окно MapBasic.

---

**См. также:**

от Функция Abs( ) до Оператор Continue

Функция ConvertToPline( ) – Функция CurrentSymbol( )

Функция DateWindow( ) – Оператор Graph

Функция HomeDirectory\$( ) – Функция NumWindows( )

Функция ObjectDistance( ) – Оператор Run Program

Оператор Save File – Функция Sgn( )

Оператор Shade – Функция Year( )

## Функция Abs( )

### Назначение

Возвращает абсолютное значение числа.

### Синтаксис

**Abs** ( *num\_expr* )

*num\_expr* – числовое выражение

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Abs( )** возвращает абсолютное значение числа, полученного в результате вычисления выражения *num\_expr*.

Если результат вычисления *num\_expr* больше или равен нулю, **Abs( )** возвращает результат *num\_expr* без изменений. Если результат вычисления *num\_expr* меньше нуля, **Abs( )** возвращает результат *num\_expr*, умноженный на минус единицу (-1).

### Пример:

```
Dim f_x, f_y As Float
f_x = -2.5
f_y = Abs(f_x)

' f_y теперь равно 2.5
```

### См. также:

**Функция Sgn( )**

---

## Функция Acos( )

### Назначение

Возвращает арккосинус числа.

### Синтаксис

**Acos** ( *num\_expr* )

*num\_expr* – численное выражение, результат которого должен находиться в диапазоне от единицы до минус единицы включительно.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Acos( )** вычисляет арккосинус числа, полученного в результате вычисления выражения *num\_expr*. Другими словами, **Acos( )** возвращает величину угла (в радианах), косинус которого равен *num\_expr*.

Результат, возвращаемый **Acos( )**, представляет из себя угол в радианах. Диапазон значения угла – между 0 и  $\pi/2$  радиан (число  $\pi$  равно приблизительно 3.141593, и  $\pi/2$  радиан равно 90 градусам).

Для перевода градусов в радианы число необходимо умножить на число DEG\_2\_RAD. Для обратного конвертирования используется коэффициент RAD\_2\_DEG. Для того, чтобы Ваша программа могла использовать эти коэффициенты конвертирования, она должна содержать ссылку на файл MAPBASIC.DEF.

Так как значения косинуса могут находиться в диапазоне от минус единицы до единицы, то и результат вычисления выражения *num\_expr* должен быть не менее минус единицы и не более единицы.

### Пример:

```
Include "MAPBASIC.DEF"
Dim x, y As Float
x = 0.5
y = Acos(x) * RAD_2_DEG
' y будет равен 60,
' потому что косинус 60 градусов равен 0.5
```

### См. также:

[Функция Asin\( \)](#), [Функция Atn\( \)](#), [Функция Cos\( \)](#), [Функция Sin\( \)](#), [Функция Tan\( \)](#)

## Оператор Add Cartographic Frame

Оператор **Add Cartographic Frame** позволяет добавлять разделы к существующей картографической легенде, созданной оператором **Оператор Create Cartographic Legend**.

### Синтаксис

**Add Cartographic Frame**

```
[ Window legend_window_id ]
[ Custom ]
[ Default Frame Title { def_frame_title } [ Font... ] ]
[ Default Frame Subtitle { def_frame_subtitle } [ Font... ] ]
[ Default Frame Style { def_frame_style } [ Font... ] ]
[ Default Frame Border Pen... pen_expr ]
Frame From Layer { map_layer_id | map_layer_name }
[ Position ( x , y ) [ Units paper_units ] ]
[ Using
  [ Column { column | object [ FromMapCatalog { On | Off } ] } ]
  [ Label { expression | default } ]
  [ Title [ frame_title ] [ Font... ] ]
  [ SubTitle [ frame_subtitle ] [ Font... ] ]
  [ Border Pen... ]
  [ Style [Font...] [ NoRefresh ] ]
  [ Text { style_name } { Line Pen...
    | Region Pen... Brush...
    | Symbol Symbol... } ]
  [ , ... ]
]
```

*legend\_window\_id* – это целочисленный идентификатор окна, который Вы можете получить при вызове функций **Функция FrontWindow( )** и **Функция WindowID( )**.

*def\_frame\_title* – это строковая величина, которая по умолчанию определяет заголовок раздела Легенды. Эта величина может включать специальный символ “#”, который будет замещаться именем текущего слоя.

*def\_frame\_subtitle* – это строковая величина, которая по умолчанию определяет заголовок подраздела легенды. Эта величина может включать специальный символ “#”, который будет замещаться именем текущего слоя.

*def\_frame\_style* – то строковая величина, которая определяет каждый символ в каждом разделе Легенды. Символ “#” будет замещаться именем текущего слоя. Символ “%” будет замещаться текстом “Line”, “Point”, “Region”, соответствующий символу легенды. Например, “% of #” будет соответствовать тексту “Region of States” для слоя STATES.TAB.

*pen\_expr* – это выражение, возвращающее объект типа Pen, то есть, `MakePen ( width, pattern, color )`. Если по умолчанию линия рамки определена, то она останется по умолчанию такой же для раздела Легенды. Если предложение Pen для рамки задано для раздела Легенды, то оно будет определять тип линии для рамки вместо заданного по умолчанию.

*map\_layer\_id* или *map\_layer\_name* определяют слой карты; задаются либо типом `SmallInt` (коротким целочисленным; например, используйте 1 для определения самого верхнего слоя, не считая косметического), либо это строковая переменная, соответствующая имени таблицы, отображенной на карте. Для тематического слоя необходимо определять параметр *map\_layer\_id*.

*paper\_units* – строка, представляющая "бумажные" единицы измерения (например, "cm" для сантиметров).

*frame\_title* – строковая переменная, определяющая заголовок раздела легенды; Если будет определено предложение заголовка раздела **Title**, то будет использоваться эта величина вместо величины *def\_frame\_title*.

*frame\_subtitle* – строковая переменная, определяющая подзаголовок раздела Легенды; Если определено предложение **SubTitle**, то оно будет использоваться вместо величины *def\_frame\_subtitle*.

*column* – атрибутивное имя колонки из раздела слоя таблицы или колонка 'object' (означает, что стили легенды базируются на уникальном стиле в сохраненном в файле карты). По умолчанию это 'object'.

*style\_name* – строковая переменная, которая указывает к какому типу: символу, линии или области относится объект в разделе легенды.

### Описание

Если ключевое слово **Custom** включено в описание, то каждый раздел легенды должен включать в себя и предложение **Position**. Если **Custom** пропущено и легенда отображается в виде книжной или альбомной ориентации, то разделы легенды будут добавлены в самый конец.

Предложение **Position** контролирует положение раздела в окне легенды. Верхний левый угол окна легенды имеет позицию 0, 0. **Position** использует "бумажные" единицы измерения, такие, как "in" (дюймы) или "cm" (сантиметры)). MapBasic имеет текущие единицы измерения, по умолчанию это дюймы; программа MapBasic может поменять эти единицы, используя оператор **Оператор Set Paper Units**. Вы можете изменить эти настройки единиц измерения, включив подпредложение **Units** в предложение **Position**.

По умолчанию в этом операторе настройки применяются только к разделам, созданным в этом операторе. Они не действуют на существующие разделы. Разделы, используемые по умолчанию в **Оператор Create Cartographic Legend**, или ранее, не действуют на разделы, созданные в этом операторе.

Когда Вы сохраняете Рабочий набор, то новое предложение **FromMapCatalog OFF** записывается в Рабочий набор. Номер версии Рабочего набора увеличивается до 800. Если предложение **FromMapCatalog ON** определено, мы не записываем его в Рабочий набор до тех пор, пока поведение Рабочего набора, является заданным по умолчанию. Номер версии Рабочего набора в этом случае не увеличивается.

**FromMapCatalog ON** извлекает стили из MapCatalog для таблицы прямого доступа. Если таблица не является таблицей прямого доступа, MapBasic возвращается к заданному по умолчанию поведению для таблицы с непрямым доступом вместо того, чтобы допустить ошибку. Стандартное поведение для таблицы с отключённым доступом это **FromMapCatalog Off** (то есть, используются стили карты).

**FromMapCatalog OFF** извлекает уникальные стили карты для таблицы прямого доступа сервера. Эта таблица должна быть таблицей прямого доступа, которая поддерживает стилизацию каждой записей. Если таблица прямого доступа не поддерживает стили записей, то применяется заданное по умолчанию поведение для таблиц прямого доступа – используются заданные по умолчанию стили из MapCatalog (**FromMapCatalog ON**).

**Label** это выражение или слово "default" (последнее означает, что при создании текста используется стиль оформления рамки по умолчанию, если только предложение стилизации не описывает текст). По умолчанию используется режим "default".

Предложение **Style** и ключевое слово **NoRefresh** позволяют Вам создавать собственные разделы легенды, которые не будут изменяться при обновлении легенды. Если ключевое слово **NoRefresh** используется в предложении **Style**, то таблица не проверяется на предмет используемых стилей. Вместо этого предложение **Style** должно содержать Ваш собственный список стилей, используемых в разделе легенды. Для этого определяются значения **Text** и предложения **Line**, **Region** или **Symbol**.

**См. также:**

**Оператор Create Cartographic Legend, Оператор Set Cartographic Legend, Оператор Alter Cartographic Frame, Оператор Remove Cartographic Frame**

---

## Оператор Add Column

### Назначение

Добавляет новую временную колонку в открытую таблицу или обновляет уже созданную колонку данными из другой таблицы.

### Синтаксис

```
Add Column table ( column [ datatype ] )
{ Values const [ , const ... ] |
  From source_table
  Set To expression
  [ Where { dest_column = source_column |
    Within | Contains | Intersects } ]
  [ Dynamic ] }
```

*table* – имя открытой таблицы, к которой будет добавляться колонка;

*column* – имя добавляемой колонки;

*datatype* – тип данных колонки, который может быть одним из следующих: Char(*width*), где *width* задает максимальное количество символов в строке; Float (используется по умолчанию); Integer; SmallInt; Decimal(*width*, *decimal\_places*), где *width* задает ширину поля и *decimal\_places* – позицию десятичной точки; Date или Logical;

*source\_table* – имя другой открытой таблицы, таблицы-источника;

*expression* – выражение, извлекающее данные из таблицы-источника *source\_table* для возможного обобщения;

*dest\_column* – имя колонки изменяемой таблицы (*table*);

*source\_column* – имя колонки-образца из таблицы-источника *source\_table*;

Dynamic – назначает колонку динамически вычисляемой: если оператор имеет это ключевое слово, то последующие изменения данных в таблице-источнике (*source\_table*) повлекут немедленное изменение в таблице с колонкой, созданной оператором.

### Описание

Оператор **Add Column** используется для создания новой временной колонки в открытой таблице MapInfo Professional. Созданная колонка не становится постоянной даже после сохранения таблицы на диске. Однако если временная колонка создана на базе данных из постоянных колонок другой таблицы и Вы сохранили Рабочий Набор, то последний будет включать информацию о временной колонке, которая позволит Вам вызвать вновь таблицу с этой колонкой, загрузив Рабочий Набор. Для создания постоянной колонки в таблице используйте операторы **Оператор Alter Table** и **Оператор Update**.

### Явное задание данных для новой колонки

Для этого используется предложение **Values**. После него задайте список постоянных значений, разделенных запятыми.

В следующем примере временная колонка таблицы WARDS заполняется числами, заданными в предложении **Value**.

```
Open Table "wards"
Add Column wards(percent_dem)
Values 31,17,22,24,47,41,66,35,32,88
```

### Заполнение новой колонки данными из другой таблицы

Если используется предложение **From** вместо **Values**, которое задает имя другой таблицы (*source\_table*), данные из которой будут использоваться в новой колонке. Обе таблицы должны быть открыты.

Если используется предложение **From**, MapInfo Professional проводит объединение двух таблиц. Условия объединения задаются в предложении **Where**. Если предложение **Where** опустить, MapInfo Professional автоматически примет решение о наиболее удобном способе объединения таблиц.



Простейшее предложение **Where** в форме `Where column = column` задает объединение двух таблиц по совпадающим значениям в их колонках. Этот метод приемлем, если колонка Вашей таблицы имеет величины, сходные с величинами в колонке из другой таблицы (например, Вы добавляете колонку в таблицу STATES и другая таблица также имеет колонку с именами штатов).

Если обе таблицы имеют графические объекты, то предложением **Where** можно задать географическое объединение. Так, если оператор включает предложение **Where Contains**, MapInfo Professional производит объединение, проверяя, могут ли объекты из таблицы *source\_table* содержать объекты из изменяемой таблицы.

Следующий оператор добавляет колонку в таблицу STORES. Новая колонка будет содержать имена округов, которые выделяются из таблицы COUNTY:

```
Add Column
stores(county char(20) 'add "county" column
From counties 'извлечь данные из таблицы counties...
Set to cname 'использовать из таблицы counties колонку "cname"
Where Contains 'объединение по критерию: county содержит торговую точку
```

Метод создания колонки с ключом **Where Contains** возможен в том случае, если таблица имеет точечные объекты и вторая таблица имеет объекты, которые могут содержать эти точки.

В следующем фрагменте таблице STATES добавляется временная колонка. Значения для нее извлекаются из таблицы CITY\_1K, содержащей данные о крупных городах США. После выполнения оператора **Add Column**, каждая строка таблицы STATES будет содержать количество крупных городов в каждом штате.

```
Open Table "states" Interactive
Open Table "city_1k" Interactive
```

```
Add Column states(num_cities)
  From city_1k 'извлечь данные из другой таблицы
  Set To Count(*) 'посчитать города в каждом штате
  Where Within 'объединение по критерию: города находятся в штатах
```

Предложение **Set To** задает правило обобщения данных функцией **Count(\*)**. Описание функций обобщения приводится ниже.

### Заполнение уже существующей колонки данными из другой таблицы.

Для того, чтобы обновить содержимое уже существующей колонки вместо того, чтобы создать новую временную колонку, надо опустить параметр **datatype** и использовать предложение **From** вместо предложения **Values**. При обновлении MapBasic будет игнорировать ключевое слово **Dynamic**.

### Обобщение данных в новой колонке.

Если Вы используете предложение **From**, то Вы можете вычислить значения для новой колонки, обобщая данные из другой таблицы (*source\_table*). Для определения правила обобщения данных используется предложение **Set To**, включающее в себя функции обобщения.

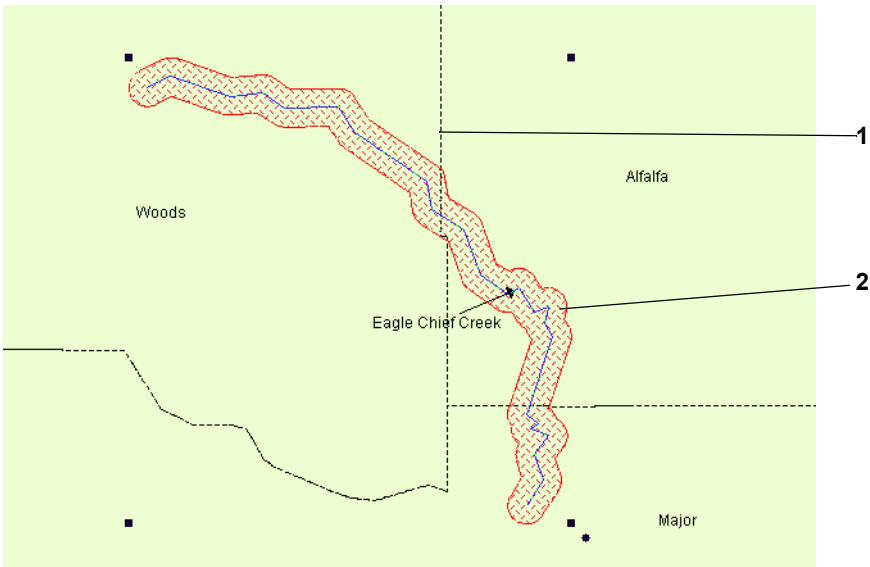
Следующая таблица приводит список функций обобщения:

| Функция                  | Что будет в новой колонке   |
|--------------------------|---|
| Avg( col )               | средняя величина всех значений поля col из строк таблицы-источника, которые участвуют в объединении;  |
| Count( * )               | число строк в таблице-источнике, которые используются в объединении;  |
| Max( col )               | наибольшая величина в колонке col из строк таблицы-источника, которые участвуют в объединении;  |
| Min( col )               | наименьшая величина в колонке col из строк таблицы-источника, которые участвуют в объединении;  |
| Sum( col )               | сумма значений из колонки по всем строкам таблицы-источника, которые используются в объединении;  |
| WtAvg( col, weight_col ) | взвешенная средняя величина значений поля col из строк таблицы-источника; колонки с большим значением <i>weight_col</i> вносят больший вклад в результат, чем колонки с меньшим значением <i>weight_col</i> ; |

| Функция                              | Что будет в новой колонке  |
|--------------------------------------|--|
| Proportion Avg( col )                | среднее число, вычисленное с учетом перекрытия одних областей другими; |
| Proportion Sum( col )                | сумма, вычисленная с учетом взаимного перекрытия областей;             |
| Proportion WtAvg( col , weight_col ) | взвешенная средняя величина с учетом перекрытия областей.              |

В основном функции обобщения работают только с неграфическими данными таблицы. Но последние три функции (Proportion Sum, Proportion Avg, Proportion WtAvg) выполняют вычисления с учетом взаимного расположения объектов. В следующем примере показан смысл этого предложения.

Допустим, Вы имеете таблицу "Районы", содержащую границы административных районов и демографическую информацию (например, население) каждого района области. Существует еще таблица RISK, которая содержит области. Объекты таблицы RISK представляют зоны различной степени риска, например, риска возможного затопления при разливе близлежащей реки.



1 Границы районов 2 Буферная зона риска

Пользуясь данными этих двух таблиц, Вы хотите рассчитать численность населения, живущего в зонах риска. Если половина района покрыта зоной риска, то будем считать, что половина населения попадает в зону риска; если треть района перекрыта зоной риска, то мы полагаем, что при наводнении опасность угрожает трети населения и т.д.

Следующий пример показывает, как с помощью функции обобщения **Proportion Sum** вычисляется новая колонка "population\_at\_risk", представляющая количество населения, подвергаемое риску:

```
Add Column Risk(population_at_risk Integer) From towns Set To Proportion  
Sum(town_pop) Where Intersects
```

Для каждого района, часть которого попадает в зону возможного затопления, MapInfo вычисляет долю пострадавшего населения и все это суммирует.

Функция **Proportion Sum** вычисляет результат на основании предположения, что суммируемые числа прямо пропорциональны ассоциированным с ними площадям. Если Вы используете **Proportion Sum** для подсчета статистики населения в регионе, половина которого попадает в другой регион, MapInfo Professional прибавит к сумме половину населения региона. На практике часто оказывается, что данные распределяются по территории неравномерно. Например, большая часть населения штата Нью-Йорк живет в городе Нью-Йорк.

Если Вы пользуетесь функцией **Proportion Sum** в случаях неравномерного распределения данных, то результаты могут быть некорректными. Чтобы добиться достоверности, разбивайте большие области на меньшие, в которых данные распределены равномерно.

Функция **Proportion Avg** вычисляет средний результат, учитывая взаимное перекрытие объектов. Продолжая пример с риском наводнения, представим, что в таблице "Районы" содержится колонка "средний\_возраст", которая содержит значения среднего возраста жителей каждого города.

Следующий оператор вычисляет средний возраст живущих в зоне риска:

```
Add Column Risk(age Float)  
  From Counties  
    Set To Proportion Avg(median_age)  
    Where Intersects
```

Для каждой записи в таблице "Районы" MapInfo Professional вычисляет отношение зоны риска к площади района. В результате получается число от нуля до единицы. MapInfo Professional умножает это число на средний возраст жителей города (средний\_возраст) и затем суммирует результаты. Так, если средний возраст равен 50 и в зоне риска находится 10% площади района, то MapInfo Professional добавит в общую сумму число 5, то есть 10% от числа 50.

**Proportion WtAvg** работает так же, как и **Proportion Avg**, но позволяет еще и извлекать весовые коэффициенты из отдельной колонки данных.

### Использование функций Proportion... с объектами иного типа, чем "область"

Если Вы используете одну из трех функций **Proportion** и таблица source\_column содержит объекты типа "область", MapInfo вычисляет процент перекрытия одной области другой. Однако, если к записям таблицы-источника присоединены объекты другого графического типа, MapInfo считает каждый объект лежащим либо внутри, либо снаружи обрабатываемой области (объект считается лежащим внутри области, если его центроид лежит внутри области).

### Динамически вычисляемая колонка

Если при создании новой колонки Вы использовали ключевое слово **Dynamic**, то новая колонка будет динамически вычисляемой, то есть изменения данных в таблице-источнике (*source\_table*) повлекут немедленное изменение в таблице с колонкой, созданной этим оператором.

Если Вы создадите динамически вычисляемую колонку и потом закроете таблицу-источник, то значения в колонке будут зафиксированы (то есть колонка больше уже не будет динамически вычисляемой).

Аналогичный эффект срабатывает при географическом объединении: если была создана динамически вычисляемая колонка и Вы закрыли какую-либо карту, используемую в географическом объединении, то значения в колонке зафиксируются.

См. также:

**Оператор Alter Table, Оператор Update**

---

## Оператор Add Map

### Назначение

Добавляет слой в окно Карты.

### Синтаксис

```
Add Map
  [ Window window_id ]
  [ Auto ]
  Layer table [ , table ... ]
  [ Animate ]
```

*window\_id* – идентификатор окна Карты, целое число;

*table* – имя открытой таблицы, объекты которой будут добавлены в окно Карты. Эта таблица должна иметь разрешение на присоединение графических объектов (*mappable*).

### Описание

Оператор **Add Map** добавляет новый слой, который содержит данные открытой таблицы, в окно Карты. Оператор может добавить сразу несколько слоев. После выполнения оператора автоматически перерисует картинку в окне Карты, если это не запрещено операторами **Оператор Set Event Processing Off** или **Оператор Set Map Redraw Off**.

Идентификатор открытого окна для параметра *window\_id* можно получить с помощью функций **Функция FrontWindow( )** и **Функция WindowID( )**. Если в операторе **Add Map** параметр *window\_id* опущен, то слой будет добавлен в самое верхнее открытое окно.

Ключевое слово **Auto** позволяет MapInfo автоматически подобрать порядковый номер слоя. Растровые таблицы и таблицы с объектами типа "область" помещаются в самый конец списка слоев, а таблицы с точечными объектами – в начало списка.

Если слово **Auto** было опущено, то слой с объектами таблицы table будет самым верхним неkosметическим слоем в окне, другими словами, при перерисовке изображения в окне объекты этого слоя выводятся на экран последними. Вы можете изменить порядок слоев оператором **Оператор Set Map**.

### Добавление слоев с разными проекциями

Если добавляется слой с растровой таблицей и Карта еще не содержит растровых слоев, то Карта воспринимает координатную систему и проекцию растрового изображения. Если Карта уже имеет два или более растровых слоев, то проекция окна заменяется на проекцию того растрового изображения, которое занимает большую часть окна.

Если добавляемый слой не является растровым, то MapInfo продолжает показывать окно Карты с использованием той проекции и координатной системы, которые действовали до применения оператора **If** добавляемый слой не является растровым, то MapInfo продолжает показывать окно Карты с использованием той проекции и координатной системы, которые действовали до применения оператора **Add Map**, даже если таблица table имела свою собственную, другую проекцию и координатную систему. Если собственные проекции таблицы не совпадают с проекциями карты, MapInfo динамически преобразует координаты, отображая данные таблицы в слое карты.

**Внимание:** При обновлении окна Карты, содержащей такие слои, перерисовка изображения будет замедлена, так как MapInfo производит математические вычисления для проецирования в слой таблицы с собственными проекциями, отличными от имеющихся.

### Использование слоя анимации для ускорения перерисовки Карты

Если в операторе **Add Map** присутствует слово **Animate**, то добавляемый слой становится анимационным. Когда объект на анимационном слое перемещается, окно Карты перерисовывается очень быстро, потому что MapInfo перерисовывает только слой с анимацией.

Пример работы анимационного слоя можно найти в программе ANIMATOR.MB из комплекта поставки.

Эффект анимации полезен в приложениях, отображающих процессы реального времени, в которых объекты Карты должны часто и быстро перерисовываться. Например, пусть Вы разрабатываете прикладную систему управления группой грузовых автомобилей, в которой каждый грузовик представлен точечным объектом. Информацию о положении грузовика Вы получаете с помощью устройства спутникового позиционирования GPS (Global Positioning Satellite), и эта информация должна незамедлительно отражаться в окне Карты. В задачах подобного типа, когда объекты на Карте постоянно перемещаются, их лучше размещать на анимационном слое, а не на обычном.

Следующие операторы открывают таблицу Vehicles и делают соответствующий слой анимационным:

```
Open Table "vehicles" Interactive
Add Map Layer vehicles Animate
```

Если с помощью оператора **Add Map** задано несколько слоев со атрибутом **Animate**, то только первый такой слой становится анимационным, а остальные слои добавляются как обычные.

Для завершения работы анимационного слоя, используйте оператор **Оператор Remove Map Layer Animate**.

Анимационные слои подчиняются специальным ограничениям. Например, пользователь не может применить инструмент Информация к объекту анимационного слоя. Каждое окно Карты может иметь только один анимационный слой. Более подробные сведения об анимационных слоях Вы можете найти в Руководстве пользователя *Руководстве пользователя MapBasic*.

### Пример:

```
Open Table "world"
Map From world
Open Table "cust1992" As customers
Open Table "lead1992" As leads
Add Map Auto Layer customers, leads
```

### См. также:

**Оператор Map, Оператор Remove Map, Оператор Set Map**

---

## Оператор Alter Button

### Назначение

Управляет выбором и доступностью кнопки на инструментальных панелях.

### Синтаксис

```
Alter Button { handler | ID button_id }
    [ { Enable | Disable } ]
    [ { Check | Uncheck } ]
```

*handler* – обработчик, который уже назначен существующей кнопке. Обработчиком *handler* может быть имя процедуры MapBasic и стандартный командный код из файла MENU.DEF, например, M\_TOOLS\_RULER или M\_WINDOW\_LEGEND);

*button\_id* – идентификатор кнопки.

### Описание

Если оператор **Alter Button** использует обработчик (например, имя процедуры), то изменения затронут все кнопки, вызывающие этот обработчик. Если используется предложение ID, то MapInfo изменяет только кнопку с номером *button\_id*.

Ключевое слово **Disable** делает кнопку недоступной. Кнопка закрашивается серым цветом и не реагирует на указание мышкой.

Ключевое слово **Enable** возвращает кнопку в активное состояние.

Слова **Check** и **Uncheck** нажимают или отжимают кнопку типа `ToggleButton` (кнопку, для которой состояние выбора фиксируется). Слово **Check** делает “нажатым” элемент `ToggleButton`, а слово **Uncheck** имеет эффект освобождения кнопки. Например, для кнопки “Показать окно статистики”:

```
Alter Button M_WINDOW_STATISTICS Check
```

**Внимание:** Нажатие кнопки таким образом не означает автоматическое выполнение действий, закрепленных за кнопкой; так, например, выбор кнопки Показать/скрыть Статистику оператором `Alter Button` не открывает окно “Статистика”, а только показывает эту кнопку нажатой на экране. Для выполнения действий кнопки в данном случае надо использовать соответствующие операторы, то есть открыть окно “Статистика” оператором **Оператор Open Window Statistics**.

Вы можете использовать слово **Check** для изменения состояния только для типа кнопок `ToolButton`. Однако, изменение состояния нажатия на кнопку типа `ToolButton` не то же самое, что и выбор инструмента этой кнопкой. Для выбора инструмента после изменения состояния кнопки Вы можете использовать оператор **Оператор Run Menu Command**, например:

```
Run Menu Command M_TOOLS_RULER
```

Для того, чтобы активизировать созданный инструмент, используется оператор `Run Menu Command ID IDnum`.

**См. также:**

**Оператор Alter ButtonPad, Оператор Create ButtonPad, Оператор Run Menu Command**

---

## Оператор Alter ButtonPad

### Назначение

Показывает/скрывает инструментальную панель или добавляет/переопределяет в ней кнопку.

### Синтаксис

```
Alter ButtonPad { current_title | ID pad_num }  
[ Add button_definition [ button_definition ... ] ]  
[ Remove { handler_num | ID button_id } [ , ... ] ]  
[ Title new_title ]  
[ Width w ]  
[ Position ( x, y ) [ Units unit_name ] ]  
[ ToolbarPosition ( row, column ) ]  
[ { Show | Hide } ]  
[ { Fixed | Float } ]  
[ Destroy ]
```

*current\_title* – имя панели (например, “Операции”);

*pad\_num* – идентификатор инструментальной панели (1 – для панели “Операции”, 2 – для панели “Пенал”, 3 – для панели “Программы”, 4 – “Команды”, 5 – “ODBC”);

*button\_id* – уникальный идентификатор для новой кнопки;



*handler\_num* – целочисленный код обработчика (например, M\_TOOLS\_RULER), определенный в файле MENU.DEF;

*new\_title* – заголовок инструментальной панели; пользователь его видит, когда панель показывается в виде вспомогательного окна;

*w* – ширина панели, измеряется количеством кнопок;

*x*, *y* – координаты верхнего левого угла панели в "бумажных" единицах;

*unit\_name* – имя "бумажной" единицы (например, "in" – дюйм, "cm" – сантиметр).

*row*, *column* – задает положение инструментальной панели, когда она находится в состоянии строки инструментов (docked) (например, 0, 0 задает расположение строки инструментов, прижатой к левому краю, и самой верхней строкой, а 0, 1 – расположение второй строкой сверху).

Каждый параметр *button\_definition* является либо ключевым словом **Separator**, либо группой предложений следующего синтаксиса:

```
{ PushButton | ToggleButton | ToolButton }
  Calling { procedure | menu_code | OLE methodname | DDE server, topic }
  [ ID button_id ]
  [ Icon icon_code [ File file_spec ] ]
  [ Cursor cursor_code [ File file_spec ] ]
  [ DrawMode dm_code ]
  [ HelpMsg msg ]
  [ ModifierKeys { On | Off } ]
  [ { Enable | Disable } ]
  [ { Check | Uncheck } ]
```

*procedure* – имя процедуры-обработчика, вызываемой при нажатии на кнопку;

*menu\_code* – стандартный в MapInfo командный код из файла MENU.DEF (например, M\_FILE\_OPEN); MapInfo начнет выполнение соответствующей команды при нажатии на кнопку;

*methodname* – строковая величина, задающая имя OLE-метода; Синтаксис **Calling OLE** смотрите в описании **Оператор Create ButtonPad on page 182**.

*server* и *topic* – строковые величины, задающие сервер DDE-связи и имя раздела (topic). Более подробно о синтаксисе **Calling DDE** смотрите в описании **Оператор Create ButtonPad on page 182**.

*button\_id* – назначает уникальный номер для кнопки. Этот номер можно использовать либо как число, используемое при вызове Справки (Tag in Help); либо как идентификатор кнопки в ее процедуре-обработчике, когда один и тот же обработчик вызывают несколько кнопок; либо как параметр в операторе **Оператор Alter Button**.

**Icon** *icon\_code* – задает пиктограмму, которая будет на кнопке. Здесь *icon\_code* может быть одним из специальных кодов файла ICONS.DEF (например, MI\_ICON\_RULER). Подпредложение File задает файл ресурсов изображений; в этом случае параметр *icon\_code* должен быть целочисленным идентификатором одного из ресурсов заданного файла.

**Cursor** *cursor\_code* задает картинку указателя, которая появится после выбора кнопки. Здесь *cursor\_code* может быть одним из специальных кодов из файла ICONS.DEF (например, MI\_CURSOR\_ARROW). Это предложение может входить только в описание кнопки инструмента (тип ToolButtons). Подпредложение File *file\_spec* задает имя файла ресурсов изображений; в этом случае параметр *cursor\_code* должен быть целочисленным идентификатором одного из ресурсов заданного файла.

*dm\_code* задает возможность инструмента рисовать (использование возможности передвигать мышку с нажатой клавишей) или только указывать (использование только возможности нажимать на клавишу мышки), при этом параметр *dm\_code* должен быть одним из специальных кодов из файла ICONS.DEF (например, DM\_CUSTOM\_LINE). то предложение может входить в описание кнопки инструмента (тип ToolButtons).

*msg* – строковая величина, задает текст подсказки, появляющейся в строке сообщений при указании на кнопку, а также может задавать текст для плавающей подсказки ToolTip. Первая часть строки *msg* используется строкой сообщений. Если строка *msg* включает в себя \n, то текст, следующий за \n, отображается в подсказке ToolTip.

Предложение **ModifierKeys** управляет использованием клавиш SHIFT и CTRL в режиме рисования, сопровождающемся прорисовкой образа объекта ("rubber-band"), инструментом кнопки типа ToolButton. По умолчанию применяется режим Off, не использующий клавиши SHIFT и CTRL.

### Описание

Оператор **Alter ButtonPad** используется для изменения состояния отображения на экране инструментальных панелей и изменения их состава. Более подробная информации об инструментальных панелях содержится в *Руководстве пользователя MapBasic*.

Для того, чтобы показать панель, в операторе используется ключевое слово **Show** ; для того, чтобы ее скрыть – ключевое слово **Hide** keyword; см. пример ниже. Пользователь может показывать и убирать с экрана инструментальные панели с помощью диалога команды **Настройка > Инструментальные панели**.

Ключевые слова **Fixed** и **Float** задают состояние панели, в котором она показывается на экране: закрепленной к верхнему краю окна или плавающей в виде вспомогательного окна. Пользователь может управлять положением инструментальной панели и помещать ее мышкой на экран, либо прикреплять к краю экрана.

Предложение **Position** задает расположение панели на экране, если она находится в плавающем состоянии. Предложение **ToolbarPosition** задает расположение панели, прикрепленной к верхнему краю рабочего окна.

С помощью ключевого слова **Destroy** Вы можете убрать панель совсем, то есть имя этой панели больше не будет отображаться в списке диалога команды **Настройка > Инструментальные панели**.

Оператор **Alter ButtonPad** может добавлять кнопки в инструментальные панели Операции и Пенал. Существуют три типа кнопок, которые можно добавить оператором: тип PushButton – кнопка, нажатие на которую приводит к определенному действию, например, к открытию диалога; тип ToggleButton – кнопка с фиксированным выбором, то есть после нажатия на

кнопку она остается в состоянии выбора (на панели она "утоплена"), для отмены выбора надо нажать на нее снова; тип `ToolButton` – кнопка инструмента, который пользователь может использовать только в окне Карты или Отчета.

Если Вы создаете кнопку с ключом **Disable**, новая кнопка будет недоступна для выбора (недоступная кнопка закрашивается серым). Доступной кнопку можно сделать следующим оператором **Оператор Alter Button**. Однако, если обработчиком (handler) кнопки определена стандартная в MapInfo команда, MapInfo автоматически изменит доступность кнопки в зависимости от возможности выполнения этой команды.

Если Вы добавите слово **Check** в описание элементов `ToggleButton` или `ToolButton`, то соответствующая кнопка при появлении будет автоматически выбрана.

Если пользователь нажал на кнопку типа `ToolButton`, MapInfo автоматически вызывает соответствующий обработчик инструмента, который будет действовать до тех пор, пока пользователь не отменит его (например, клавишей ESC). В процедуре обработчика может быть использована функция **CommandInfo( )** для определения места, на которое пользователь указал мышкой. Если две или более инструментальные кнопки вызывают один обработчик, то функция **CommandInfo( )** поможет Вам определить идентификатор кнопки, вызвавший его.

### Встроенные картинки для кнопок и указателя мышки

Предложение **Icon** определяет пиктограмму, которую пользователь увидит на кнопке. Если опущено подпредложение **File**, то параметр *n* должен быть одним из кодов, имена которых определены в файле `ICONS.DEF` (например, `MI_ICON_RULER`).

**Внимание:** MapInfo имеет много встроенных иконок для кнопок, не являющихся частью нормального пользовательского интерфейса. Просмотреть эти иконки Вы можете, запустив программу `ICONDEMO.MBX`. Эта программа позволяет просмотреть иконки и скопировать код иконки в почтовый ящик Windows (clipboard) для дальнейшего использования в Вашей программе.

В Windows подпредложение **File** *file\_spec* ссылается на DLL-файл, который содержит ресурсы растровых изображений, а параметр *n* в предложении **Icon** является идентификатором ID одного из них. Более подробно создание новых кнопок в Windows описано в *Руководстве пользователя MapBasic*.

Для кнопок типа `ToolButton` Вы также можете определить в предложении **Cursor** картинку, которую примет указатель мышки в то время, пока активен инструмент, вызванный этой кнопкой. Имена возможных кодов для этого определены в файле `ICONS.DEF` (например, `MI_CURSOR_CROSSHAIR` или `MI_CURSOR_ARROW`). Правила использования собственных ресурсов для указателя мыши такие же, как при определении пиктограммы кнопки.

### Режимы рисования

Определение `ToolButton` может включать предложение **DrawMode**, которое задает возможность указывать, перемещая инструмент (например, рисование линии) или только одним нажатием на клавишу мыши (например, рисование точечного объекта). В следующей таблице перечислены возможные режимы рисования. Имена для режимных кодов определены в файле `ICONS.DEF`.

| Значение DrawMode  | Описание  |
|--------------------|---|
| DM_CUSTOM_POINT    | Инструмент может только указывать, а не рисовать.   |
| DM_CUSTOM_LINE     | При рисовании инструментом от указателя тянется прямая линия, закрепленная другим концом в точке, в которой была нажата клавиша мышки.  |
| DM_CUSTOM_RECT     | При рисовании инструментом указатель растягивает прямоугольник, закрепленный противоположным по диагонали углом в точке, в которой была нажата клавиша мышки.                             |
| DM_CUSTOM_CIRCLE   | При рисовании указатель растягивает окружность.   |
| DM_CUSTOM_ELLIPSE  | При рисовании инструментом указатель растягивает эллипс. Если добавлено предложение ModifierKeys, то при нажатой клавише SHIFT будет растягиваться окружность.                            |
| DM_CUSTOM_POLYGON  | Рисует многоугольник (полигон). Чтобы узнать, какой объект был нарисован пользователем, надо вызвать функцию:<br><b>CommandInfo( )</b><br><b>функцияCommandInfo(CMD_INFO_CUSTOM_OBJ).</b> |
| DM_CUSTOM_POLYLINE | Рисует ломаную (полилинию). Чтобы узнать, какой объект был нарисован пользователем, надо вызвать функцию:<br><b>CommandInfo( )</b><br><b>функцияCommandInfo(CMD_INFO_CUSTOM_OBJ).</b>     |

Все режимы рисования, исключая режим DM\_CUSTOM\_POINT, поддерживают режим автопрокрутки, который позволяет автоматически перемещать окно Карты или Отчета при операциях выбора и рисования с участием мыши. Отключить автопрокрутку можно оператором **Оператор Set Window on page 682**.

**Внимание:** MapBasic поддерживает дополнительный режим рисования, недоступный из среды MapInfo Professional. Если **Calling** задано в виде **Calling M\_TOOLS\_SEARCH\_POLYGON**, то инструмент может рисовать многоугольник. Когда пользователь дважды укажет мышкой для закрытия полигона, MapInfo выбирает все объекты (с доступных слоев Карты), попавшие в многоугольник. Многоугольник не сохраняется.

## Примеры

Следующие операторы показывают инструментальную панель "Операции" и скрывают панель "Пенал":

```
Alter ButtonPad "Main" Show
Alter ButtonPad "Drawing" Hide
```

Следующий пример прикрепляет панель "Операции" к верхнему краю рабочего окна:

```
Alter ButtonPad "Main" Fixed ToolbarPosition(0,0)
```

Теперь переведем панель "Операции" в плавающее состояние и поместим ее на расстоянии полдюйма от верхнего и левого краев экрана.

```
Alter ButtonPad "Main" Float Position(0.5,0.5) Units "in"
```

Прикладная программа из стандартной поставки SCALEBAR содержит оператор **Alter ButtonPad**, который добавляет кнопку типа ToolButton на инструментальную панель "Программы". ("ID 3" обозначает панель "Программы".)

```
Alter ButtonPad ID 3
  Add
    Separator
    ToolButton
      Icon MI_ICON_CROSSHAIR
      HelpMsg "Draw a distance scale on a map\nScale Bar"
      Cursor MI_CURSOR_CROSSHAIR
      DrawMode DM_CUSTOM_POINT
      Calling custom_tool_routine
  Show
```

**Внимание:** Ключевое слово **Separator** вставляет пустое пространство между последней кнопкой панели "Программы" и новой "+" кнопкой.

**См. также:**

**Оператор Alter Button**, **Функция ButtonPadInfo( )**, **Оператор Create ButtonPad**, **Оператор Set Window**

---

## Оператор Alter Cartographic Frame

### Назначение

Оператор **Alter Cartographic Frame** изменяет положение, заголовок, подзаголовок, рамку и стиль существующего раздела легенды, созданной оператором **Оператор Create Cartographic Legend**. (Для изменения размера, позиции или заголовка окна легенды, используйте оператор **Оператор Set Window**.)

### Синтаксис

```
Alter Cartographic Frame
[ Window legend_window_id ]
Id { frame_id }
[ Position ( x, y ) [ Units paper_units ] ]
[ Title [ frame_title ] [ Font... ] ]
[ SubTitle [ frame_subtitle ] [ Font... ] ]
[ Border Pen... ]
[ Style [ Font... ]
  [ ID { id } Text { style_name } ]
  [ Line Pen... | Region Pen... Brush... | Symbol Symbol... ] ]
[ , ... ]
```

*legend\_window\_id* – это целочисленный идентификатор окна, который Вы можете получить при вызове функций **Функция FrontWindow( )** и **Функция WindowID( )**.

*frame\_id* – индекс ID-раздела легенд. Вы не можете использовать здесь имя слоя. Например, три раздела легенды могут иметь индексы ID 1, 2 и 3.

*frame\_title* – строковая переменная, определяющая заголовок раздела легенды;

*frame\_subtitle* – строковая переменная, определяющая подзаголовок раздела Легенды;

*id* – положение внутри списка стилей для данного раздела. В настоящее время нет функций MapBasic, которые могут дать информацию о номере стиля в разделе легенды.

*style\_name* – это строковая величина, которая отображает следующий за каждым символ для раздела с указанным индексом ID. Символ “#” будет замещаться именем текущего слоя. Символ “%” будет замещаться текстом “Line”, “Point”, “Region”, соответствующий символу легенды. Например, “% of #” будет заменено на “Region of States” для раздела легенды, соответствующей слою states.tab.

### Описание

Если предложение **Window** не определено, MapInfo будет использовать самое верхнее окно легенды.

Предложение **Position** контролирует положение раздела в окне легенды. Верхний левый угол окна легенды имеет позицию 0, 0. Position использует “бумажные” единицы измерения, такие, как “in” (дюймы) или “cm” (сантиметры)). MapBasic имеет по умолчанию установку в дюймах; программа MapBasic может поменять единицы, используя **Оператор Set Paper Units**. Оператор **Alter Cartographic Frame** может изменить единицы измерения с помощью подпредложения **Units** в предложении **Position**.

В предложения **Title** и **SubTitle** можно вводить новый текст, новый шрифт или и то и другое.

Предложение **Style** должно содержать список определений для стилей, отображающихся в разделе. Вы можете только обновлять Style для собственного стиля. Вы можете обновлять Text для любого стиля. Нет возможности добавлять или удалять стили для любых типов разделов легенды.

### См. также:

**Оператор Create Cartographic Legend, Оператор Set Cartographic Legend, Оператор Add Cartographic Frame, Оператор Remove Cartographic Frame**

---

## Оператор Alter Control

### Назначение

Изменяет состояние элемента диалога, созданного приложением.

### Синтаксис

```
Alter Control id_num
  [ Title { title | From Variable array_name } ]
  [ Value value ]
  [ { Enable | Disable } ]
  [ { Show | Hide } ]
  [ Active ]
```

*id\_num* – целочисленный идентификатор одного из элементов активного диалога;

*title* – новый заголовок для элемента диалога, строковая величина;

*array\_name* – имя или массив величин, используемый для элементов типа ListBox, MultiListBox, RadioGroup и PopupMenu;

*value* – новое значение для элемента диалога.

### Предупреждение

Этот оператор нельзя запускать из окна MapBasic.

### Описание

Оператор **Alter Control** изменяет атрибуты элемента активного диалога, окно которого было открыто оператором **Dialog**. Применение оператора **Alter Control** возможно только пока диалоговое окно открыто, т. е. в специальной подпрограмме, называемой процедурой-обработчиком элемента диалога, вызов которой учитывается при создании диалога. Если на экране находятся два или более диалоговых окон, то оператор **Alter Control** воздействует на активное окно, которое лежит поверх остальных.

Параметр *id\_num* определяет элемент диалога, который будет изменяться. Значение параметра соответствует значению параметра *id\_num*, заданного в предложении **ID** в операторе **Оператор Dialog**.

Изменение состояния и атрибутов элемента диалога производится при помощи предложений **Title**, **Value**, **Enable/Disable**, **Hide/Show**, **Active**. Оператор может использовать либо одно из этих предложений, либо одновременно несколько, либо все. То есть одновременно оператор **Alter Control** может изменить имя, значение и режим доступности элемента диалога.

Однако, не все атрибуты могут меняться для каждого типа элемента диалога. Например, элемент **Button control** может быть включен и выключен, но не имеет атрибута.

Предложение **Title** назначает текст для большинства элементов (исключение составляют элементы типа **Picker** и **EditText**; текст элемента **EditText** определяется значением через предложение **Value**). Если Вы меняете текстовый атрибут для элементов типа **ListBox**, **MultiListBox** или **PopupMenu**, предложение **Title** может читать новое содержимое элемента из строкового массива переменных, если оно дополнено ключевым словом **From Variable**.

Ключевое слово **Active** используется только для элемента **EditText**. Оператор **Alter Control...Active** помещает курсор в текстовое окошко элемента.

Ключевое слово **Hide** прячет элемент, оставляя пустое место в окне диалога на его месте. Показать вновь элемент можно, используя ключевое слово **Show**.

Для полной отмены выбора в списке элемента **MultiListBox** определите значение элемента (параметр *value*) равным нулю. Для того, чтобы добавить к текущему выбору в списке элемента **MultiListBox** еще одну строку, выполните оператор **Alter Control** с положительным значением, соответствующим номеру строки в списке.

**Внимание:** В этом случае не нужно выполнять оператор **Alter Control** изнутри обработчика элемента **MultiListBox**.

Вы можете использовать оператор **Alter Control** для изменения текста, появляющегося в элементе **StaticText**. Однако, MapInfo Professional не может увеличить величину элемента **StaticText** после того как он был создан. Поэтому, если Вы планируете поменять текст элемента **StaticText**, добавьте к нему при определении пробелы. Например, Ваш оператор **Оператор Dialog** может включать следующее предложение:

```
Control StaticText ID 1 Title "Сообщение" + Space$(30)
```



**Пример:**

Следующая программа создает диалог с двумя флажками и кнопками "ОК" и "Отмена" ("Cancel"). Когда диалог открывается, кнопка "ОК" не активна (окрашена серым). Кнопка становится доступной пользователю, когда он установит один или оба флажка.

```
Include "mapbasic.def" Declare Sub Main Declare Sub checkerSub Main Dim
browse_it, map_it As LogicalDialogTitle "Display a file"Control CheckBox
Title "Display in a Browse window"Value 0Calling checkerID 1Into
browse_itControl CheckBoxTitle "Показать окно Карты"Value 0 Calling
checker ID 2 Into map_itControl CancelButtonControl OKButtonID 3
DisableIf CommandInfo(CMD_INFO_DLG_OK) Then ' ... если пользователь нажал
OK... End If End Sub
Sub checker
' If either check box is checked,
' enable the OK button; otherwise, Disable it.
If ReadControlValue(1) Or ReadControlValue(2) ThenAlter Control 3
Enable Else Alter Control 3 Disable End IfEnd Sub
```

**См. также:**

**Оператор Dialog, Оператор Dialog Preserve, Функция ReadControlValue( )**

---

## Оператор Alter MapInfoDialog

**Назначение**

Делает недоступными, прячет или присваивает значение элементу стандартного диалогового окна в MapInfo Professional.

**Предупреждение**

**ВНИМАНИЕ:** Оператор Alter MapInfoDialog может не поддерживаться в будущих версиях MapInfo Professional. В результате программы MapBasic, которые используют этот оператор могут не работать корректно в следующих версиях MapInfo Professional. Используйте этот оператор осторожно.

**Синтаксис 1 (присвоение нестандартных установок)**

```
Alter MapInfoDialog dialog_ID
Control control_ID
{ Disable | Hide | Value new_value } [ , { Disable... } ]
[ Control... ]
```

**Синтаксис 2 (восстановление стандартных установок)**

```
Alter MapInfoDialog dialog_ID Default
```

*dialog\_ID* – целое число, идентификатор изменяемого диалогового окна MapInfo;

*control\_ID* – целое число от 1 и более, идентификатор изменяемого элемента диалога;

*new\_value* – новое значение элемента диалога.

### Описание

Оператор позволяет делать недоступными, скрытыми и присваивать значения элементам стандартных диалогов MapInfo - кнопкам, флажкам и т. п.

**Внимание:** Используйте этот оператор только для изменения стандартных диалогов MapInfo Professional. Пользуйтесь оператором **Оператор Dialog** для работы с элементами диалогов, построенных оператором **Оператор Alter Control**.

### Определение идентификатора ID

Для определения идентификатора диалога необходимо загрузить MapInfo для Windows командой:

```
mapinfow.exe -helpdiag
```

После того, как программа MapInfo будет запущена с аргументом `-helpdiag` выведите нужное диалоговое окно на экран и нажмите на кнопку Справки в нем. Обычно, эта кнопка вызывает окно Справочной системы, но в случае использования `-helpdiag` MapInfo показывает идентификатор данного диалога.

**Внимание:** Существуют разные общесистемные диалоги (такие как диалоги открытия и сохранения) в разных версиях Windows. Если Вы хотите внести изменения в общесистемный диалог, и если Ваше приложение используется в разных версиях Windows, Вам необходимо приготовить две версии оператора **Alter MapInfoDialog** по одному для каждой версии Windows.

Каждый элемент диалога также имеет идентификатор. Например, большинство кнопок "ОК" имеют идентификатор 1, а кнопка "Отмена" ("Cancel") - идентификатор 2. Определить идентификатор элемента диалога можно только с помощью программы, не входящей в состав пакета MapInfo, такой, как программа Microsoft Spy++ из пакета компилятора C. В пакет MapBasic программа Spy++ не входит.

Измененное состояние элементов диалога и значения, присвоенные оператором **Alter MapInfoDialog**, сохраняются, только если пользователь закрыл диалог с подтверждением **ОК**. Например, Вы используете оператор **Alter MapInfoDialog** для помещения адреса в диалог "Найти", но MapInfo не выполнит поиск, пока Вы не откроете диалог и пользователь не нажмет на кнопку **ОК**.

### Возможные виды изменений

Ключевое слово **Disable** используется для того, чтобы сделать элемент недоступным для пользователя (в диалоге элемент закрашивается серым цветом).

Ключевое слово **Hide** прячет элемент, делает его невидимым.

Ключевое слово **Value** изменяет значение элемента.

Если Вы вносите изменения в общесистемный диалог (например, "Открыть таблицу"), то можете изменить выбор в поле со списком (combo box), или Вы можете изменить текст подписи (static text), кнопки или в текстовом окошке.

Вы можете изменить положение элементов в диалоге "Настройка печати". Кнопкам "Книжная" и "Альбомная" соответствуют числа 1056 и 1057 соответственно.

Если Вы изменяете другой диалог MapInfo, то следующий список приводит виды изменений, которые Вы можете в нем сделать:

- **Кнопка, подпись (static text), текстовое окошко, окошко со списком (editable combo box):** можно менять текст, задавая новый текст параметром *new\_value*.
- **Список (list box), окошко со списком (combo box):** можно менять выбор в списке, задавая номер элемента списка в параметре *new\_value*.
- **Флажок:** можно устанавливать (значение равно 1) или сбрасывать (значение равно 0).
- **Переключатель:** можно менять выбор кнопки, задавая значение 0 (не выбрана) или 1 (выбрана).
- **Кнопка стиля символа:** можно менять установку стиля символа (например, используя вызов функции **Функция MakeSymbol( )**).
- **Кнопка стиля линии:** можно менять установку стиля линии.
- **Кнопка стиля штриха:** можно менять установку стиля заливки.
- **Кнопка стиля шрифта:** можно менять установку стиля заливки.
- **Кнопка стиля оформления области (линия/штрих):** можно задать новое значение стиля линии или стиля штриха. (Пример использования этого элемента Вы можете видеть диалоге "Стиль области" MapInfo Professional, который открывается если Вы дважды укажете мышкой на изменяемый регион.)

### Пример:

Следующий фрагмент вносит в диалог "Найти" текстовую строку с адресом ("Волхонка 13") в поле первого окна и прячет кнопку "Снова".

```
If SystemInfo(SYS_INFO_MIVERSION) = 400 Then Alter MapInfoDialog 2202
Control 5 Value "Волхонка 13"Control 12 Hide End If Run Menu Command
M_ANALYZE_FIND
```

Идентификатор окна "Найти" – 2202. Control 5 ссылается на текстовое поле, в которое вводится адрес для поиска. Control 12 – ссылается на кнопку "Снова". Все эти номера в будущей версии MapInfo будут изменены. Поэтому используется функция **Функция SystemInfo( )** для определения версии программы, в которой выполняется это приложение.

**См. также:**

**Оператор Alter Control, Оператор Dialog, Функция SystemInfo( )**

---

## Оператор Alter Menu

### Назначение

Добавляет элемент в список одного из существующих меню в окне MapInfo или убирает существующий элемент из списка меню.

### Синтаксис 1

```
Alter Menu { menuname | ID menu_id }
Add menundef [ , menundef... ]
```

Где *menundef* – идентификатор элемента списка меню, который имеет следующий синтаксис:

```
newmenuitem  
[ ID menu_item_id ]  
[ HelpMsg help ]  
[ { Calling handler | As menuname } ]
```

*menuname* – заголовок меню;

*menu\_id* – целочисленный идентификатор меню от 1 до 22, где единица представляет меню "Файл";

*newmenuitem* – строка: имя, которое будет добавлено к указанному меню;

*menu\_item\_id* – целочисленный идентификатор элемента меню, который может быть использован в дальнейшем оператором **Оператор Alter Menu Item**;

*help* – строковая величина, текст которой будет показываться в строке сообщений, когда элемент меню будет выбран (подсвечен);

*handler* – имя sub-процедуры обработчика или командный код, имена для которых определены в файле MENU.DEF (например, M\_FILE\_NEW) или же специальный код обработки события выбора меню механизмами OLE или DDE. Если Вы зададите код команды для стандартной команды MapInfo Professional типа Показать/Скрыть (например, M\_WINDOW\_STATISTICS), строка *newmenuitem* должна начинаться с восклицательного знака и включать в себя символ ^, чтобы отделять режимы показа и скрытия. Особенности разных способов обработки см. в описании оператора **Оператор Create Menu**.

### Синтаксис 2

```
Alter Menu { menuname | ID menu_id }  
  Remove { handler | submenu_name | ID menu_item_id }  
  [ , { handler | submenu_name | ID menu_item_id } ... ]
```

*menuname* – заголовок уже определенного меню;

*menu\_id* – целочисленный идентификатор меню от 1 до 22, где единица представляет меню "Файл";

*handler* – имя sub-процедуры обработчика или код для стандартной команды MapInfo;

*submenu\_name* – заголовок удаляемого подменю, иерархически подчиненного заданному меню;

*menu\_item\_id* – целочисленный идентификатор элемента меню.

### Описание

Оператор **Alter Menu** добавляет или убирает элемент в списке меню (или подменю).

Элемент меню *menuname* в операторе может быть идентифицирован его именем (например, "File" или "Файл"). Поэтому, если Вы планируете использовать Вашу прикладную программу в MapInfo другой языковой версии, используйте в операторе **Alter Menu** вместо имени номер, задаваемый предложением ID.

Если Вы собираетесь заменить одно из стандартных меню MapInfo Professional, задавайте в операторе **Alter Menu ID-номера меню**. Параметр ID принимает значения от 1 до 31.

Ниже приводится таблица с именами стандартной системы меню MapInfo и соответствующими им значениями идентификатора.

**Внимание:** Меню с 16 по 31 – это быстрые меню, которые появляются при нажатии на правую кнопку мыши. Быстрые меню доступны только в Windows.

#### ID-номер меню

| Имя меню            | ID | Описание   |
|---------------------|----|--|
| File                | 1  | Стандартное меню Файл.   |
| Edit                | 2  | Стандартное меню Правка.   |
| Query               | 3  | Стандартное меню Запрос.   |
| Tools               | 4  | Меню Программы.  |
| Options             | 5  | Стандартное меню Настройки.  |
| Window              | 6  | Стандартное меню Окно.   |
| Help                | 7  | Стандартное меню Справка.  |
| Browse              | 8  | Меню Список Используется, когда активно окно Списка.   |
| Map                 | 9  | Меню Карта. Используется, когда активно окно Карты.  |
| Layout              | 10 | Меню Отчет. Используется, когда активно окно Отчета.   |
| Graph               | 11 | Меню График. Используется, когда активно окно Графика.   |
| MapBasic            | 12 | Меню MapBasic. Используется, когда активно окно MapBasic.  |
| Redistrict          | 13 | Меню Районирование. Используется, когда активно окно Районирование.  |
| Objects             | 14 | Стандартное меню Объекты   |
| Table               | 15 | Стандартное меню Таблица.  |
| DefaultShortcut     | 16 | Стандартное короткое меню. Это меню открывается при нажатии на правую кнопку мыши в окнах, не имеющих собственных коротких меню. |
| MapperShortcut      | 17 | Стандартное короткое меню окна Карты.  |
| BrowserShortcut     | 18 | Стандартное короткое меню окна Списка.   |
| LayoutShortcut      | 19 | Стандартное короткое меню окна Отчета.   |
| GrapherShortcut (1) | 20 | Стандартное короткое меню окна Графика. Меню содержит команды создания графиков.   |

ID-номер меню (*continued*)

| Имя меню            | ID | Описание  |
|---------------------|----|---|
| CmdShortcut         | 21 | Стандартное короткое меню окна MapBasic.  |
| RedistrictShortcut  | 22 | Стандартное короткое меню окна районирования; открывается для окна списка районов.                                      |
| LegendShortcut      | 23 | Стандартное короткое меню окна Легенды.   |
| GrapherShortcut (2) | 24 | Второе стандартное короткое меню окна Графика. Меню содержит команды форматирования и изменения уже имеющегося графика. |
| 3DMapShortcut       | 25 | Стандартное короткое меню окна трехмерной Карты.  |
| 3DWindow            | 28 | Стандартное короткое меню окна трехмерного представления.   |
| Graph               | 29 | Меню График.  |
| Legend              | 31 | Меню Легенда.   |

## Примеры

Добавим в список меню Файл новую команду:

```
Alter Menu "File" Add
  "Special" Calling sub_procedure_name
```

Следующий фрагмент делает то же самое, что и предыдущий пример, только используется идентификатор меню.

```
Alter Menu ID 1 Add "Специальная команда" Calling sub_procedure_name
```

Теперь добавим значение идентификатора для команды **ID**, равный 300, который позже можно будет использовать в операторе **Оператор Alter Menu Item**.

```
Alter Menu ID 1 Add "Специальная команда" ID 300 Calling
sub_procedure_name
```

В следующем примере удаляется ранее созданный элемент из меню Файл.

```
Alter Menu ID 1 Remove sub_procedure_name
```

Программа TEXTBOX использует оператор **Оператор Create Menu** для создания меню "Рамка" и затем оператор **Alter Menu** для добавления его в качестве иерархически подчиненного в меню "Программы":

```
Alter Menu "Программы" Add "(-", "Рамка" As "Рамка"
```

В следующем примере добавляется новая команда в быстрое меню для окна Карты (это меню появляется при нажатии на вторую кнопку мыши).

```
Alter Menu ID 17 Add "Найти ближайший объект" Calling sub_procedure_name
```

См. также:

[Оператор Alter Menu Bar](#), [Оператор Alter Menu Item](#), [Оператор Create Menu](#), [Оператор Create Menu Bar](#)

## Оператор Alter Menu Bar

### Назначение

Добавляет или удаляет заголовки меню в строку меню окна MapInfo.

### Синтаксис

```
Alter Menu Bar { Add | Remove }
    { menuname | ID menu_id }
    [ , { menuname | ID menu_id } ... ]
```

*menuname* – имя меню (например, "Файл");

*menu\_id* – целочисленный идентификатор меню от 1 до 22, где единица представляет меню Файл.

### Описание

Оператор **Alter Menu Bar** добавляет или убирает один или более заголовков меню в строку меню.

Параметр *menuname* может быть строкой или выражением строкового типа, результатом которого является имя одного из стандартных меню в MapInfo (например, "Файл" или "Правка"). Параметр *menuname* может быть также именем меню, созданного оператором [Оператор Create Menu](#) (смотрите примеры)

**Внимание:** Поэтому, если Вы планируете использовать Вашу прикладную программу в MapInfo другой языковой версии, используйте в операторе **Alter Menu** вместо имени номер, задаваемый предложением ID. Каждое из стандартных меню MapInfo Professional ("Файл", "правка" и т.п.) имеет также уникальный ID-номер, который можно использовать независимо от того, как переведен заголовок меню.. Например, используя ID 2 мы всегда обращаемся к меню "Edit" или "Правка".

Список имен стандартной системы меню MapInfo и соответствующих им значений идентификаторов приведен в разделе, описывающем оператор [Оператор Alter Menu на стр. 45](#).

### Как добавить меню

Оператор **Alter Menu Bar Add** добавляет заголовок меню в строку меню с правого края. Если Вам необходимо вставить меню в определенное место в строке меню, то для переопределения строки меню выполните оператор [Оператор Create Menu Bar](#).

В Windows, если Вы добавите много заголовков меню в строку меню, то в ней появляется вторая строка.

### Как убрать меню

Оператор **Alter Menu Bar Remove...** удаляет заголовок из строки меню. При этом меню не пропадает и может быть в любой момент восстановлено так, как показано ниже. Следующая пара операторов сначала удаляет меню Запрос из строки меню, а затем помещает ее снова туда крайним справа:

```
Alter Menu Bar Remove "Запрос" Alter Menu Bar Add "Запрос"
```

После того, как оператор **Alter Menu Bar Remove...** удалит меню, MapInfo отменяет все клавишные сокращения, ранее назначенные командам, которые находились в удаленном списке меню. Например, пользователь MapInfo Professional может нажать **Ctrl-O** и открыть из меню **Файл** диалог "Открыть"; однако, если оператором **Alter Menu Bar Remove** меню **Файл** было удалено, то MapInfo Professional проигнорирует нажатие **Ctrl-O**.

### Пример:

В следующем примере создается меню "Данные", затем выполняется оператор **Alter Menu Bar Add** для добавления этого меню в строку меню MapInfo Professional.

```
Declare Sub addsubDeclare Sub editsubDeclare Sub delsub
Create Menu "Данные" As "Добавить" Calling addsub,"Правка" Calling
editsub,"Удалить" Calling delsub
'Удаляются меню Окно и Справка... Alter Menu Bar Remove ID 6, ID 7
'Добавляется меню Данные, а затем восстанавливаются меню Окно и
СправкаAlter Menu Bar Add "DataEntry", ID 6, ID 7
```

Перед тем как поместить заголовок созданного меню в строку меню, эта программа сначала удаляет меню **Справка** (идентификатор 7) и меню **Окно** (идентификатор 6). Затем в правый конец строки меню добавляется сначала заголовок меню **Данные**, а затем меню **Окно** и **Справка**. Так мы добиваемся того, что заголовки **Окно** и **Справка** всегда будут последними в строке меню.

### См. также:

**Оператор Alter Menu, Оператор Alter Menu Item, Оператор Create Menu, Оператор Create Menu Bar, Оператор Menu Bar**

---

## Оператор Alter Menu Item

### Назначение

Изменяет состояние элемента списка меню.

### Синтаксис

```
Alter Menu Item { handler | ID menu_item_id } { [ Check | Uncheck ] | [
Enable | Disable ] | [ Text itemname ] | [ Calling handler | As menuname
] }
```

*handler* – имя sub-процедуры или код для стандартной команды MapInfo;



*menu\_item\_id* – целочисленный идентификатор элемента меню *menu\_item\_id*, который задается при создании списка меню (оператором **Оператор Create Menu** или **Оператор Alter Menu**);

*itemname* – новый текст для элемента меню (может содержать управляющие коды);

*menuname* – заголовок уже определенного меню;

## Описание

Оператор **Alter Menu Item** изменяет значения атрибутов одного или более элементов списка меню. Например, оператор **Alter Menu Item** может сделать команду недоступной для выбора (на экране она закрашивается серым).

Элемент меню может задаваться либо именем обработчика handler, который запускается при выборе элемента в списке меню, либо идентификатором в предложении **ID**. Заметим, что один и тот же обработчик могут вызывать разные элементы меню. Поэтому, если оператор **Alter Menu Item** использует имя процедуры-обработчика handler, то MapInfo будет менять все элементы, вызывающие этот обработчик, из всех меню. Если Вы используете предложение **ID**, то MapInfo изменит атрибуты только одного элемента меню.

Оператор **Alter Menu Item** может использовать идентификатор **ID** только для тех элементов, для которых при создании списка меню он был определен.. Приложение MapBasic не может использовать идентификатор, который был задан другим приложением MapBasic.

Режимы **Check** и **Uncheck** регулируют наличие или отсутствие галочки у команды в меню. Заметим, что пункт меню может быть отмеченным галочкой только если при его при его определении была задана эта возможность (т.е..если **Оператор Create Map** в перед именем меню был поставлен символ "!").

Режимы **Disable** и **Enable** определяют доступность выбора элемента меню. Недоступные элементы закрашиваются серым цветом. Заметим, что MapInfo автоматически делает некоторые элементы меню доступными и недоступными в соответствии с текущим состоянием в среде программы MapInfo. Например, команда **Файл > Заккрыть** становится серой, если не открыто ни одной таблицы. Поэтому приложение MapBasic не может изменять доступность стандартного элемента MapInfo. Вы можете обращаться к инструментальным средствам как к элементам меню (например, M\_TOOLS\_RULER в MENU.DEF), но не можете сделать их недоступными с помощью оператора **Alter Menu Item**.

В предложении **Text** можно изменить имя элемента.

Предложение **Calling** задает имя процедуры-обработчика, вызываемой элементом меню. Если пользователь выберет этот элемент в меню, то MapInfo запустит на выполнение эту процедуру.

## Примеры

Создается меню **Данные**, содержащее четыре команды, и затем добавляется в строку заголовков меню MapInfo.

```
Declare Sub addsubDeclare Sub editsubDeclare Sub delsub
Create Menu "Данные" As
    "Добавить" Calling addsub,
    "Правка" Calling editsub,
```

```
"Удалить" ID 100 Calling delsub,  
"Удалить все" ID 101 Calling delsub  
'Удаляется меню Справка...  
Alter Menu Bar Remove ID 7  
'Add both the new menu and the Help menu  
Alter Menu Bar Add "Данные" , ID 7
```

Следующий оператор **Alter Menu Item** переименовывает команду Правка в команду Правка...

```
Alter Menu Item editsub Text "Правка..."
```

Следующий оператор делает команду Удалить все недоступной.

```
Alter Menu Item ID 101 Disable
```

Следующий оператор делает недоступными две команды: Удалить все и Удалить, так как они используют один и тот же обработчик delsub.

```
Alter Menu Item delsub Disable
```

**См. также:**

**Оператор Alter Menu, Оператор Alter Menu Bar, Оператор Create Menu**

## Оператор Alter Object

### Назначение

Изменяет форму, положение или графический тип существующего объекта.

### Синтаксис

```
Alter Object obj
{ Info object_info_code, new_info_value |
  Geography object_geo_code , new_geo_value |
  Node { Add [ Position polygon_num, node_num ] ( x, y ) |
        Set Position polygon_num, node_num ( x , y ) |
        Remove Position polygon_num, node_num
      }
}
```

*obj* – переменная типа Object;

*object\_info\_code* – целое число, код, возвращаемый функцией **Функция ObjectInfo( )** (например, OBJ\_INFO\_PEN).

*new\_info\_value* – новое значение для кода *object\_info\_code* (например, новая величина типа Pen);

*object\_geo\_code* – целое число, код, возвращаемый функцией **Функция ObjectGeography( )** (например, OBJ\_GEO\_POINTX).

*new\_geo\_value* – новое значение для кода *object\_geo\_code* (например, новая X-координата);

*polygon\_num* – короткое целое число, идентификатор для одного полигона в объекте регион (область);

*node\_num* – короткое целое число, идентификатор для одного узла в полилинии или полигоне;

*x*, *y* – X- и Y-координаты узла.

### Описание

Оператор **Alter Object** изменяет форму, местоположение, графический стиль существующего объекта.

Эффект действия оператора **Alter Object** зависит от того, какое предложение используется в конструкции оператора **Info**, **Node** или **Geography**. Если оператор использует предложение **Info** то MapBasic изменяет графический стиль оформления объекта (например, стиль линии и штриха). Если оператор использует предложение **Node** , то MapBasic добавляет, удаляет или передвигает узлы объекта типа "полилиния" или "область". Если оператор включает в себя предложение **Geography**, то MapBasic изменяет географические атрибуты всех объектов, не являющихся полилиниями и областями (например, X- или Y-координата точечного объекта).

### Предложение Info

Оператор **Alter Object** с предложением **Info** изменяет стиль оформления объекта (например, стиль линии и штриха). В предложении **Info** изменяются атрибуты, значения которых можно получить от функции **Функция ObjectInfo( )**. Например, Вы можете определить текущий стиль штриха (величину типа Brush), вызвав функцию **Функция ObjectInfo( )**:

```
Dim b_fillstyle As Brush
b_fillstyle = ObjectInfo(Selection.obj, OBJ_INFO_BRUSH)
```

И, наоборот, следующий оператор **Alter Object** восстанавливает прежнее значение стиля штриховки:

```
Alter Object obj_variable_name
Info OBJ_INFO_BRUSH, b_fillstyle
```

Заметьте, что Вы используете один и тот же код (OBJ\_INFO\_BRUSH) в функции **Функция ObjectInfo( )** и в операторе **Alter Object**.

В следующей таблице в первой колонке приводятся имена кодов для использования в предложении **Info** в качестве параметра *obj\_info\_code*. Значения *obj\_info\_code* находятся в файле MAPBASIC.DEF. Если Вы хотите использовать имена в операторе **Alter Object...Info**, включите в начало Вашей программы оператор Include "MAPBASIC.DEF".

| Значения <i>obj_info_code</i> | Результат выполнения <b>Alter Object</b>  |
|-------------------------------|---|
| OBJ_INFO_PEN                  | Изменяется стиль линии или контура. Параметр <i>new_info_value</i> должен иметь значение типа Pen.  |
| OBJ_INFO_BRUSH                | Изменяется стиль штриховки объекта. Параметр <i>new_info_value</i> должен иметь значение типа Brush.                                      |
| OBJ_INFO_TEXTFONT             | Изменяется стиль шрифта. Параметр; <i>new_info_value</i> должен иметь значение типа Font.   |
| OBJ_INFO_SYMBOL               | Изменяется стиль символа. Параметр <i>new_info_value</i> должен иметь значение типа Symbol.   |
| OBJ_INFO_SMOOTH               | Изменяется режим сглаживания углов для полилиний. Параметр <i>new_info_value</i> должен иметь значение логического типа (TRUE или FALSE). |
| OBJ_INFO_FRAMEWIN             | Меняет содержимое рамки на изображение другого окна. Параметр <i>new_info_value</i> должен быть целочисленным идентификатором окна.       |
| OBJ_INFO_FRAMETITLE           | Заменяет заголовок в рамке. Параметр <i>new_info_value</i> должен иметь значение типа String.   |
| OBJ_INFO_TEXTSTRING           | Меняет текст в текстовом объекте. Параметр <i>new_info_value</i> должен иметь значение типа String.                                       |

| Значения <i>obj_info_code</i> | Результат выполнения <b>Alter Object</b>   |
|-------------------------------|--|
| OBJ_INFO_TEXTSPACING          | Изменяет расстояние между строками в текстовом объекте. Параметр <i>new_info_value</i> должен иметь значение типа Float равным 1, 1.5, или 2.  |
| OBJ_INFO_TEXTJUSTIFY          | Изменяет значение выравнивания для текстовых объектов. Параметр <i>new_info_value</i> должен иметь одно из следующих целочисленных значений: /n0 – выравнивание влево, /n1 – центрирование, /n2 – выравнивание вправо. |
| OBJ_INFO_TEXTARROW            | Изменяет вид указки для текстового объекта. Параметр <i>new_info_value</i> должен иметь одно из следующих целочисленных значений: /n0 – нет указки, /n1 – только линия, /n2 – линия со стрелкой.                       |

### Предложение Geography

Оператор **Alter Object** с предложением **Geography** изменяет расположение объекта. Предложение **Geography** действительно для всех типов объектов за исключением полилиний и областей. Для изменения расположения объектов последних двух типов используйте предложение **Node** (которое описано ниже) вместо предложения **Geography**.

The **Geography** clause lets you modify the same attributes that you can query through the **Функция ObjectGeography( )**. Например, Вы можете получить координаты конца объекта "линия" выполнив **Функция ObjectGeography( )**:

```
Dim o_cable As Object
Dim x, y As Float
x = ObjectGeography(o_cable, OBJ_GEO_LINEENDX)
y = ObjectGeography(o_cable, OBJ_GEO_LINEENDY)
```

Оператор **Alter Object** изменяет координаты концов линейного объекта::

```
Alter Object o_cable
  Geography OBJ_GEO_LINEENDX, x
Alter Object o_cable
  Geography OBJ_GEO_LINEENDY, y
```

**Внимание:** Заметим, что используется один и тот же код (OBJ\_GEO\_LINEENDX) в функции **Функция ObjectGeography( )** и в операторе **Alter Object**.

В следующей таблице в первой колонке приводятся имена кодов для использования в предложении Geography в качестве параметра *obj\_geo\_code*. Имена *obj\_geo\_code* присвоены целочисленным кодам для удобства использования их в операторе и находятся в файле MAPBASIC.DEF. если Вы хотите использовать эти имена в операторе **Alter Object...Geography**, включите в начало Вашей программы оператор Include "MAPBASIC.DEF".

| Значения <i>obj_geo_code</i> | Результат выполнения <b>Alter Object</b>   |
|------------------------------|--|
| OBJ_GEO_MINX                 | Изменяет X-координату верхнего левого угла минимального прямоугольного покрытия (МПП). |
| OBJ_GEO_MINY                 | Изменяет Y-координату верхнего левого угла МПП.  |
| OBJ_GEO_MAXX                 | Изменяет X-координату нижнего правого угла МПП.  |
| OBJ_GEO_MAXY                 | Изменяет Y-координату нижнего правого угла МПП.  |
| OBJ_GEO_ARCBEGANGLE          | Изменяет начальный угол дуги.  |
| OBJ_GEO_ARCENDANGLE          | Изменяет конечный угол дуги.   |
| OBJ_GEO_LINEBEGX             | Изменяет X-координату начальной точки линии.   |
| OBJ_GEO_LINEBEGY             | Изменяет Y-координату начальной точки линии.   |
| OBJ_GEO_LINEENDX             | Изменяет X-координату конечной точки линии.  |
| OBJ_GEO_LINEENDY             | Изменяет Y-координату конечной точки линии.  |
| OBJ_GEO_POINTX               | Изменяет X-координату точечного объекта.   |
| OBJ_GEO_POINTY               | Изменяет Y-координату точечного объекта.   |
| OBJ_GEO_ROUNDRAIUS           | Изменяет радиус закругления для объекта типа "скругленный прямоугольник".              |
| OBJ_GEO_TEXTLINEX            | Изменяет координату по оси X конца текстовой строки объекта.                           |
| OBJ_GEO_TEXTLINEY            | Изменяет координату по оси Y конца текстовой строки объекта.                           |
| OBJ_GEO_TEXTANGLE            | Изменяет угол поворота текстового объекта.   |

### Предложение Node

Оператор **Alter Object** с предложением **Node** добавляет, перемещает или убирает узлы полилиний и многоугольников (полигонов), составляющих области.

Если предложение **Node** включает под-предложение **Add**, оператор **Alter Object** добавляет узел к **объекту**. Включив в **Node** предложение **Remove**, Вы можете удалять узел. Предложение **Set Position** меняет местоположение узла.

Обычно оператор **Alter Object** с предложением **Node** используется в связке с операторами **Оператор Create Pline** и **Оператор Create Region**. Эти операторы позволяют создавать полилинии или полигоны. Операторы Create требуют точного задания количества узлов объекта на этапе компиляции. Однако, в ряде случаев Вы не можете знать, сколько узлов будет содержать объект при работе программы.

Если Ваша программа не знает точное количество узлов, Вы можете, используя операторы **Оператор Create Pline** или **Оператор Create Region**, создать "пустые" объекты (т. е. количество узлов объекта будет нулевым). С помощью оператора **Alter Object...Node Add** Вы можете добавить к созданному объекту любое количество узлов.

В предложении **Node** подпредложение **Position** задает два параметра: *polygon\_num* и *node\_num*, указывающие на конкретный узел, который будет изменен или удален. Подпредложение **Position** при определении узла задавать необязательно. Параметры *polygon\_num* и *node\_num* могут принимать значения от 1 и более.

Параметр *polygon\_num* задает, какой полигон из мультиполигонального объекта будет обработан.

## Центроиды регионов

Центроиды регионов могут быть заданы используя команду **Alter Object** со следующим синтаксисом:

```
Alter Object Obj Geography OBJ_GEO_CENTROID, PointObj
```

Обратите внимание, что *PointObj* это точечный объект. Это отличает его от других значений, вводимых оператором **Alter Object Geography**, которые являются скалярными. Точечный объект для своего определения на карте нуждается в двух координатах. Вводимый объект *Point* проверяется на предмет того может ли он быть нормальным центроидом, то есть, попадает ли он внутрь данного региона. Если имеющийся объект *Obj* не является регионом, или если *PointObj* не является точечным объектом или если точка не является нормальным центроидом, то будет возвращено сообщение об ошибке.

Вот один из вариантов использования значений X и Y для центроида:

```
Alter Object Obj Geography OBJ_GEO_CENTROID, CreatePoint(X, Y)
```

Пользователь может также опросить центроид используя функцию **Функция ObjectGeography( )**:

```
PointObj = ObjectGeography(Obj, OBJ_GEO_CENTROID)
```

Существует и другой метод получения центроида, включая функции **Функция Centroid( )**, **Функция CentroidX( )** и **Функция CentroidY( )**.

OBJ\_GEO\_CENTROID определен в MAPBASIC.DEF.

## Объекты Группа точек и Коллекция

Оператор **Alter Object** расширен поддержкой следующих новых типов объектов.

- **Группа точек:** устанавливает символ группы точек, как показано ниже::

```
Alter Object obj_variable_mpoint  
Info OBJ_INFO_SYMBOL, NewSymbol
```

- **Коллекция:** Используя оператор **Alter Object** с предложением **Info**, можно переустановить части коллекции (регион, полилиния или группа точек) внутри объекта "коллекция". Предложение **Info** позволяет видоизменять те же атрибуты, которые Вы можете запрашивать через функцию **Функция ObjectInfo( )**. Например, можно определить часть регионов объекта коллекция вызовом функции **Функция ObjectInfo( )**:

```
Dim ObjRegion As Object  
ObjRegion = ObjectInfo(Selection.obj, OBJ_INFO_REGION)
```

Следующий оператор **Alter Object** позволяет переустановить часть регионов, входящих в коллекцию:

```
Alter Object obj_variable_mpoint  
Info OBJ_INFO_REGION, ObjRegion
```

**Внимание:** Вы используете тот же самый код ( OBJ\_INFO\_REGION) и в функции **Функция ObjectInfo( )** и в операторе **Alter Object**.

В операторе **Alter Object** поддерживается вариант, позволяющий вставлять и удалять узлы из объектов типа "группа точек".

```
Alter Object obj Node statement
```

Чтобы вставить узлы в группу точек::

```
Dim mpoint_obj as object  
Create Multipoint Into Variable mpoint_obj 0  
Alter Object mpoint_obj Node Add (0,1)  
Alter Object mpoint_obj Node Add (2,1)
```

**Внимание:** Узлы для группы точек всегда добавляются в конец таблицы.

Чтобы удалить узлы из группы точек:

```
Alter Object mpoint_obj Node Remove Position polygon_num, node_num
```

*mpoint\_obj* – объект типа "группа точек";

*polygon\_num* – игнорируется для группы точек, рекомендуется установить значение 1;

*node\_num* – число удаляемых узлов.

Чтобы установить точки внутри группы точек:

```
Alter Object mpoint_obj Node Set Position polygon_num, node_num (x,y)
```

*mpoint\_obj* – объект типа "группа точек";

*polygon\_num* – игнорируется для группы точек, рекомендуется установить значение 1;

*node\_num* – число узлов, которые будут изменены;

*x* and *y* – новые координаты узла *node\_num*.

### Пример:

```
Dim myobj As Object, i As Integer  
Create Region Into Variable myobj 0  
For i = 1 to 10  
    Alter Object myobj  
        Node Add (Rnd(1) * 100, Rnd(1) * 100)  
Next
```

**Внимание:** После использования оператора **Alter Object** для изменения объекта, используйте оператор **Оператор Insert** или **Оператор Update** для сохранения объекта в таблице.



См. также:

Оператор `Create Pline`, Оператор `Create Region`, Оператор `Insert`, Функция `ObjectGeography( )`, Функция `ObjectInfo( )`, Оператор `Update`

## Оператор Alter Table

### Назначение

Изменяет структуру открытой таблицы. Не может быть применен к связанным таблицам.

### Синтаксис

```
Alter Table table (  
    [ Add columnname columntype [, ...] ]  
    [ Modify columnname columntype [, ...] ]  
    [ Drop columnname [, ...] ]  
    [ Rename oldcolumnname newcolumnname [, ...] ]  
    [ Order columnname, columnname [,...]] )  
    [ Interactive ]
```

*table* – имя открытой таблицы;

*columnname* – имя колонки (поля) в открытой таблице, длина которой не должна превышать 31 символ и состоит из букв, цифр и символа подчеркивания и не может начинаться с цифры;

*columntype* – тип данных колонки (поля) в таблице (включая ширину поля, если необходимо);

*oldcolumnname* – старое имя колонки (поля) для переименования;

*newcolumnname* – новое имя колонки (поля) для переименования.

### Описание

Оператор **Alter Table** используется для изменения структуры открытой таблицы. Можно добавлять и удалять колонки, изменять у существующих колонок ширину или тип данных, переименовывать колонки и менять порядок их расположения.

**Внимание:** Если в таблице были изменения, то перед тем как выполнить оператор **Alter Table** Вы должны сохранить их на диске или восстановить таблицу с диска, удалив изменения.

Параметр *columntype* может иметь следующие значения: Integer – целое число (4 байта); SmallInt – короткое целое число (2 байта); Float – десятичное число с плавающей точкой; Decimal(size, decplaces) – десятичное число фиксированной длины (size) и числом знаков после запятой (decplaces); Char(size) – строка символов, где size – максимальное количество символов в поле; Date – дата; Logical – логическая величина.

Предложение **Add** в операторе **Alter Table** используется для добавления новой колонки в Вашу таблицу. Предложение **Modify** используется для изменения типа данных колонки. Предложение **Drop** используется для удаления колонки. Предложение **Rename** изменяет имя колонки. Предложение **Order** позволяет Вам задать свой порядок расположения колонок. Один оператор **Alter Table** одновременно может включать в себя все пять предложений. Каждое предложение может обращаться к списку колонок; поэтому одним оператором **Alter Table** Вы можете сделать все структурные изменения в таблице (смотрите пример).

Предложение **Order** работает только с колонками и не влияет на порядок записей в таблице. Перечислите через запятую имена полей в нужном Вам порядке. Первое имя в списке соответствует самой левой колонке в окне Списка. В таком же порядке будут расположены поля в окне "Информация" – в верхней строке будет значение из первой колонки таблицы и так далее по порядку.

Если программа применяет оператор **Alter Table** к таблице, которая имеет мемо-поля, то последние будут утеряны. Предупреждения об этом на экране не будет.

Оператор **Alter Table** может послужить причиной удаления слоев из окна Карты, а также привести к потере объектов тематических слоев или объектов с Косметического слоя.. Если Вы используете ключевое слово **Interactive**, то MapInfo предложит пользователю сохранить тематические или/и косметические объекты.

### Пример:

Таблица GCPOP.TAB состоит из следующих колонок: "pop\_88", "metsize", "fipscod" и "utmcode". Оператор **Alter Table** делает следующее: в предложении **Rename** меняет имя колонки "pop\_88" на "Население"; в предложении **Drop** удаляет колонки "metsize", "fipscod", "utmcode"; в предложении **Add** создает колонку "Школа" для 2-байтовых значений целого типа (SmallInt) и колонку "Субсидии" вещественного типа; в предложении **Order** заново определяет порядок колонок в таблице.

```
Open Table "gcpop"
Alter Table gcpop
  (Rename pop_88 population
   Drop metsize, fipscod, utmcode
   Add schoolcode SmallInt, federalaid Float
   Order schoolcode, population, federalaid)
```

**См. также:**

**Оператор Add Column, Оператор Create Index, Оператор Create Map, Оператор Create Table**

---

## Функция ApplicationDirectory\$( )

### Назначение

Возвращает имя каталога, в котором находится файл выполняющегося приложения MapBasic.

### Синтаксис

```
ApplicationDirectory$( )
```

### Возвращаемая величина

Строка. Величина типа String.

### Описание

Вызов функции **ApplicationDirectory\$( )** из приложения поможет Вам определить каталог (или папку), из которого была запущена прикладная программа. Если в данный момент никаких приложений не загружено (Вы вызвали функцию в окне MapBasic), то функцией **ApplicationDirectory\$( )** возвращается пустая строка.

Для определения каталога, в котором установлена программа MapInfo, используйте функцию **Функция ProgramDirectory\$( )**.

### Пример:

```
Dim sAppPath As String
sAppPath = ApplicationDirectory$( )
' Здесь переменная, sAppPath должна иметь значение:
'
' "C:\MAPBASIC\CODE\"
```

### См. также:

**Функция ProgramDirectory\$( )**

---

## Функция Area( )

### Назначение

Возвращает географическую площадь объекта.

### Синтаксис

**Area( *obj\_expr*, *unit\_name* )**

*obj\_expr* - выражение, определяющее объект.

*unit\_name* – единицы измерения площади (например, "sq km" – квадратные километры).

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Area( )** возвращает площадь географического объекта, определенного параметром *obj\_expr*.

Функция возвращает площадь области в единицах измерения, указанных в параметре *unit\_name*; к примеру, чтобы получить площадь в акрах, укажите “акры” в качестве значения параметра *unit\_name*. Список доступных единиц измерения смотрите в **Оператор Set Area Units on page 618**.

Только полигоны, эллипсы, прямоугольники и прямоугольники со скругленными углами имеют площадь. Результат применения функции к точечным, текстовым объектам, а также к прямой линии, дуге и полилинии будет равен нулю. Для скругленного прямоугольника функция **Area( )** возвращает приблизительное значение. MapBasic вычисляет площадь прямоугольников со скругленными углами как если бы они были обычными прямоугольниками.

В большинстве случаев MapInfo Professional проводит либо декартовы, либо сферические вычисления. Обычно выполняются сферические вычисления; если координатная система - план-схема, то выполняются декартовы вычисления.

### Примеры

В этом примере демонстрируется, как используется функция **Area( )** для вычисления площади географического объекта. Обратите внимание, что выражение *tablename.obj* (здесь *states.obj*) представляет географический объект текущей строки в указанной таблице.

```
Dim f_sq_miles As Float
Open Table "states"
Fetch First From states
f_sq_miles = Area(states.obj, "sq mi")
```

Можно также использовать функцию **Area( )** в составе оператора SQL Select:

```
Select state, Area(obj, "sq km")
  From states Into results
```

**См. также:**

**Функция ObjectLen( ), Функция Perimeter( ), Функция CartesianArea( ), Функция SphericalArea( ), Оператор Set Area Units**

---

## Функция AreaOverlap( )

### Назначение

Вычисляет площадь пересечения двух замкнутых объектов.

### Синтаксис

**AreaOverlap( object1, object2 )**

*object1* и *object2* – два замкнутых (имеющих площадь) объекта.

### Возвращаемая величина

Десятичное число с плавающей запятой. Величина типа Float.

**См. также:**

**Функция Overlap( ), Функция ProportionOverlap( ), Оператор Set Area Units**

### Функция Asc( )

#### Назначение

Возвращает код первого символа строки.

#### Синтаксис

**Asc**( *string\_expr* )

*string\_expr* – выражение, результат которого есть символьная строка.

#### Возвращаемая величина

Целое число типа Integer.

#### Описание

Функция **Asc( )** возвращает код первого символа строки *string\_expr*.

Если строка *string\_expr* пустая, то функция **Asc( )** возвращает ноль.

**Внимание:** Вычислительные платформы, на которых работает MapInfo Professional, имеют общие коды от 32 (пробел) до 126 (тильда). В этот диапазон входят все цифры и буквы латинского алфавита. Буквы русского алфавита в разных системах имеют разные коды.

В системах, поддерживающих двухбайтовые коды символов (например, Windows Japanese): если первый символ в строке *string\_expr* однобайтовый, то функция **Asc( )** вернет код от 0 до 255; если первый символ в строке *string\_expr* двухбайтовый, то функция, **Asc( )** вернет код от 256 до 65535.

В системах, не поддерживающих систему двухбайтовых кодов, функция **Asc( )** возвращает значения *returns* в диапазоне от 0 до 255.

#### Пример:

```
Dim code As SmallInt
code = Asc("Afghanistan")
' code will now be equal to 65,
' code равно 65, коду символа A
```

#### См. также:

**Функция Chr\$( )**

## Функция Asin( )

### Назначение

Возвращает арксинус.

### Синтаксис

**Asin**( *num\_expr* )

*num\_expr* – численное выражение, результат которого должен находиться в диапазоне от единицы до минус единицы включительно.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Asin( )** вычисляет арксинус числа, полученного в результате вычисления выражения *num\_expr*. Другими словами, **Asin( )** возвращает величину угла в радианах, синус которого равен параметру *num\_expr*.

Результатом вычисления **Asin( )** является угол, значение которого возвращается в радианах. Диапазон значения угла находится между  $-\pi/2$  и  $\pi/2$  радиан (число  $\pi$  равно приблизительно 3.141593, и  $\pi/2$  радиан равно 90 градусам).

Для перевода градусов в радианы число необходимо умножить на число DEG\_2\_RAD. Для обратного конвертирования используется коэффициент RAD\_2\_DEG. Чтобы Ваша программа могла использовать эти коэффициенты конвертирования, она должна содержать оператор Include "MAPBASIC.DEF".

Так как синус меняет свои значения от единицы до минус единицы, выражение *num\_expr* должно возвращать числа от 1 до -1.

### Пример:

```
Include "MAPBASIC.DEF"
Dim x, y As Float
x = 0.5
y = Asin(x) * RAD_2_DEG

' y равно 30,
' так как синус от 30 градусов равен 0.5
```

### См. также:

**Функция Acos( ), Функция Atn( ), Функция Cos( ), Функция Sin( ), Функция Tan( )**

### Функция Ask( )

#### Назначение

Показывает диалоговое окно с сообщением или вопросом и предлагающее подтвердить или отменить предложение (**OK** или **Cancel**).

#### Синтаксис

**Ask**( *prompt*, *ok\_text*, *cancel\_text* )

*prompt* – текст сообщения, заданный в кавычках;

*ok\_text* – текст на кнопке подтверждения (например, "Да" или "Продолжить");

*cancel\_text* – текст на кнопке отмены (например, "Отменить" или "Нет").

#### Возвращаемая величина

Логическое

#### Описание

Функция **Ask( )** формирует и выводит на экран диалоговое окно, в котором пользователю предлагается ответить "да" или "нет". Параметр *prompt* содержит текст сообщения или вопроса, на который надо ответить пользователю, например – "Файл уже существует. Заменить на новый?". Текст сообщения должен содержать не более 300 символов.

Диалоговое окно имеет две кнопки: одна для положительного ответа, другая для отрицательного. Параметры *ok\_text* и *cancel\_text* задают текст для этих кнопок (например, "OK" или "Да" – для положительного ответа, "Нет" или "Стоп" – для отрицательного).

Если пользователь нажмет на кнопку с надписью *ok\_text*, функция **Ask( )** вернет TRUE. При нажатии на кнопку с надписью *cancel\_text* или клавиши Esc в диалоге, функция **Ask( )** возвращает FALSE. Текст для кнопок *ok\_text* и *cancel\_text* должен быть лаконичен и сжат. Если Ваша фраза настолько велика, что не помещается в пределах стандартной кнопки, то лучше воспользуйтесь оператором **Оператор Dialog** вместо функции **Ask( )**. Кнопка *ok\_text* выбирается по умолчанию, то есть вместо нажатия мышкой эту кнопку можно выбрать клавишей ENTER.

#### Пример:

```
Dim more As Logical  
more = Ask("Вы согласны удалить объекты?", "Да", "Нет")
```

#### См. также:

**Оператор Dialog, Оператор Note, Оператор Print**



## Функция Atn( )

### Назначение

Возвращает арктангенс.

### Синтаксис

**Atn**( *num\_expr* )

*num\_expr* – числовое выражение

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Atn( )** возвращает арктангенс от числа, полученного в результате вычисления выражения *num\_expr*. Другими словами, **Atn( )** возвращает угловое значение в радианах, при котором тангенс равен числу *num\_expr*. Число *num\_expr* может быть любым.

Результатом вычисления **Atn( )** является угол, значение которого возвращается в радианах. Диапазон значения угла находится между  $-\pi/2$  и  $\pi/2$  радиан (число  $\pi$  равно приблизительно 3.141593, и  $\pi/2$  радиан равно 90 градусам).

Для перевода градусов в радианы число необходимо умножить на число DEG\_2\_RAD. Для обратного конвертирования используется коэффициент RAD\_2\_DEG. Чтобы Ваша программа могла использовать эти коэффициенты конвертирования, она должна содержать оператор Include "MAPBASIC.DEF".

### Пример:

```
Include "MAPBASIC.DEF"
Dim val As Float

val = Atn(1) * RAD_2_DEG
'val равен 45, потому что
'арктангенс от 1 равен 45 градусам
```

### См. также:

[Функция Acos\( \)](#), [Функция Asin\( \)](#), [Функция Cos\( \)](#), [Функция Sin\( \)](#), [Функция Tan\( \)](#)

## Оператор AutoLabel

### Назначение

Размещает подписи объектов в окне Карты на Косметическом слое.

### Синтаксис

#### AutoLabel

```
[ Window window_id ]  
[ { Selection | Layer layer_id } ]  
[ Overlap [ { On | Off } ] ]  
[ Duplicates [ { On | Off } ] ]
```

*window\_id* – идентификатор окна, целое число;

*layer\_id* – имя таблицы или идентификатор слоя, целое число.

### Описание

Оператор **AutoLabel** создает подписи для объектов (объекты типа “текст”) в окне Карты. Подписываются только те объекты, которые в данный момент видны в окне Карты. Предложение **Window** определяет окно Карты. Если предложения **Window** нет, MapBasic обращается к самому верхнему окну Карты.. Предложение **Selection**, определяет действие оператора на выборку.. Если **Selection** и **Layer** не заданы, то подписываются все слои..

Предложение **Overlap** воздает режимы взаимного перекрытия подписей. По умолчанию режимы имеют значение **Off** (подписи не могут накладываться друг на друга). Чтобы MapInfo Professional подписывала объекты, не взирая на перекрытия, задайте режим **Overlap On**. Предложение **Duplicates** управляет количеством подписываний одного объекта. По умолчанию режимы имеют значение **Off** (множественность подписей запрещена). Оператор **AutoLabel** использует текущие установки шрифта и расположение подписей. Пользователь может изменять эти установки в диалоге команды **Карта > Управление слоями**. Изменить установки шрифта и расположение подписей из приложения MapBasic можно, используя оператор **Оператор Set Map**.

### Пример:

```
Open Table "world" Interactive  
Open Table "worldcap" Interactive  
Map From world, worldcap  
AutoLabel  
  Window FrontWindow( )  
  Layer world
```

### См. также:

**Оператор Set Map**

## Оператор Веер

### Назначение

Подает звуковой сигнал.

### Синтаксис

**Веер**

### Описание

Оператор **Веер** посылает команду подачи звука на динамик Вашего компьютера.

---

## Оператор Browse

### Назначение

Открывает окно Списка.

### Синтаксис

```
Browse expression_list From table  
  [ Position ( x, y ) [ Units paper_units ] ]  
  [ Width window_width [ Units paper_units ] ]  
  [ Height window_height [ Units paper_units ] ]  
  [ Row n ]  
  [ Column n ]  
  [ Min | Max ]
```

*expression\_list* – выражения, задающие через запятую колонки, или звездочка (\*);

*table* – имя открытой таблицы;

*x*, *y* – координаты верхнего левого угла окна Списка в "бумажных" единицах *paper\_units*;

*paper\_units* – строка, представляющая "бумажные" единицы измерения (например, "см" для сантиметров).

*window\_width* и *window\_height* – определяют размер окна Списка в *paper\_units*;

*n* – положительное целое число.

### Описание

Оператор **Browse** открывает новое окно Списка для открытой таблицы.

Список может показать все поля таблицы, если параметр *expression\_list* равен звездочке (\*), или же вид окна Списка может быть определен списком выражений. Предложение *expression\_list* может, наоборот, состоять из списка выражений, каждое из которых задает одну колонку в окне Списка. Выражением может быть имя колонки, оператор, функция, число, которое определяет одну колонку окна Списка. Имена колонок, показываемые в самой верхней строке Списка, полностью зависят от параметра. Thus, if a column is defined by the expression `population / area(obj, "acre")`, то оно и будет именем колонки. Чтобы создать псевдоним выражения, добавьте это имя после выражения; см. пример ниже.

Предложение **Position** задает расположение окна на экране. Координаты *x* и *y* определяют верхний левый угол окна Списка относительно верхнего левого угла окна MapInfo. Предложения **Width** и **Height** определяют ширину и высоту окна Списка. Если предложений **Width** и **Height** нет в операторе, MapInfo самостоятельно определит размер следующим образом: площадь окна Списка приблизительно равна четверти рабочего окна MapInfo, так, чтобы строки и колонки были показаны в окне полностью.

Если оператор **Browse** включает в себя ключевое слово **Max**, то окно будет открыто полностью на всю рабочую область окна MapInfo. Аналогично, если оператор **Browse** включает в себя ключевое слово **Min**, то окно будет свернуто в икону.

Предложение **Row** используется для определения, какая строка будет самой верхней в окне Списка. Если в операторе **Browse** не задано предложение **Row**, то самой верхней будет первая строка.

Предложение **Column** используется для определения, какая колонка будет самой левой в окне Списка. Если в операторе **Browse** не задано предложение **Column**, то самой левой будет первая колонка.

### Пример:

Этот пример демонстрирует, как открытую таблицу WORLD показать в окне Списка.

```
Open Table "world"
Browse * From world
```

Данные из той же таблицы можно показать в окне Списка по-другому, явно задавая колонки Списка и используя выражения для них.

```
Open Table "world"
Browse
    country,
    population,
    population/area(obj, "sq km") "Плотность"
From world
```

Результатом будет окно Списка из трех колонок. Первые две колонки содержат данные так, как они хранятся в файле таблицы WORLD. Третья колонка является вычисляемой, и MapBasic помещает в нее результат деления населения на площадь страны для каждой строки таблицы. Ей присваивается псевдоним ("Плотность"), который пользователь увидит в окне в строке заголовков колонок.

См. также:

[Оператор Set Browse](#), [Оператор Set Window](#)

---

## Предложение Brush

### Назначение

Задаёт стиль штриховки графических объектов.

### Синтаксис

**Brush** *brush\_expr*

*brush\_expr* – выражение, результат которого есть величина типа Brush. Например, результат выполнения функции **MakeBrush**( *pattern*, *fgcolor*, *bgcolor* ). (См. более подробную информацию о переменной Brush в разделе [Функция MakeBrush\( \) on page 398](#).

### Описание

Предложение **Brush** входит в состав операторов, в которых необходимо задавать стиль штриха для следующих объектов: многоугольник (полигон), область (регион), прямоугольник и эллипс. **Brush** не является отдельным оператором. Различные операторы, такие как [Оператор Create Ellipse](#), могут возвращать переменную типа. Слово **Brush** может сопровождаться выражением, возвращающим значение типа Brush. Например, такого вида:

```
Brush br_var
```

Или же параметр может задаваться значением, например, полученным вызовом функций:

```
Brush MakeBrush(64, CYAN, BLUE)
```

В некоторых операторах (таких как [Оператор Set Map](#)), переменная **Brush** задается набором из трех целочисленных параметров (*pattern*, *foreground\_color*, *background\_color*), заключенным в скобки, например:

```
Brush(64, CYAN, BLUE)
```

Некоторые операторы MapBasic используют выражения типа стиля штриха в качестве параметра (например, переменная типа **Brush**) не используя при этом предложения **Brush**. Одним из примеров является [Оператор Alter Object](#).

В таблице приводится определение компонент стиля штриховки замкнутых объектов:

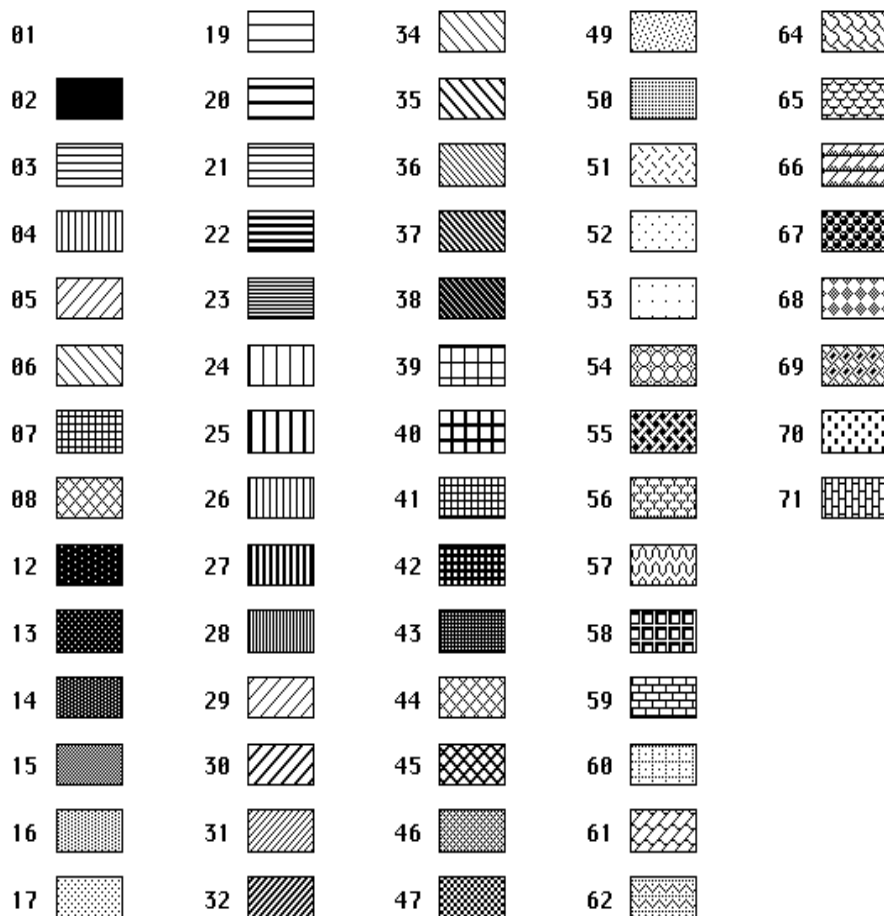
| Компонента<br>стиля | Описание   |
|---------------------|--|
| pattern             | Целое число от 1 до 8 и от 12 до 71. Задаёт рисунок штриха. Смотрите таблицу ниже.   |
| foreground color    | Целочисленный код цвета для рисунка штриха. Может быть заменен вызовом функции; см. <a href="#">Функция RGB( )</a> . Вы также можете использовать имена для кодов цвета (как в примере выше), если в тексте Вашей программы есть ссылка на файл стандартных определений MAPBASIC.DEF. В нем определены имена для следующих цветов: BLACK — черный; WHITE — белый; RED — красный; GREEN — зеленый; BLUE — синий; CYAN — голубой; MAGENTA — фиолетовый; YELLOW — желтый. |
| background color    | Целочисленный код цвета фона штриха.   |

Чтобы задать прозрачный фон, выберите штриховку под номером 3 или выше и не задавайте фон в предложении **Brush**.. Например, задав `Brush(5, BLUE)`, Вы получите штриховку тонкими синими линиями без фона. Не задавая параметр фона, Вы поступаете аналогично сбросу флажка **Фон** в диалоге MapInfo Professional "Стиль области".

Чтобы задать прозрачный фон, задайте при вызове [Функция MakeBrush\( \)](#) значение фона - 1.

Ниже приводится таблица кодов и соответствующих им рисунков штриховок. Заметим, что для штриха номер 1 (прозрачный) или 2 (ровная заливка), параметр `background_color` не влияет на заливку.

**Внимание:** Более подробный список штриховок приведен в *Справке* MapInfo Professional и на веб-сайте MapInfo.



См. также:

Функция **CurrentBrush( )**, Функция **MakeBrush( )**, Предложение **Pen**, Предложение **Font**, Предложение **Symbol**

## Функция Buffer( )

### Назначение

Возвращает объект типа "область", представляющий собой буферную зону вокруг выбранного объекта (область, граница которой отстоит от границы объекта на заданное расстояние).

### Синтаксис

**Buffer(** *inputobject*, *resolution*, *width*, *unit\_name* **)**

*inputobject* – объектное выражение (выражение, результат которого есть величина типа Object);

## Функция ButtonPadInfo( )

---

*resolution* – число узлов многоугольника, принимаемого как окружность (число типа SmallInt);

*width* – радиус буфера, число типа Float; если *width* отрицательно и *inputobject* закрытый объект, то возвращается объект меньший, чем исходный. Если задано отрицательное значение *width*, а объект является линейным (линией, полилинией или дугой) или же точкой, то используется положительное значение *width* и создается буфер больший чем объект.

*unit\_name* – имя единицы измерения расстояний *width* (например, "mi" – миля, "km" – километр).

### Возвращаемая величина

Область. Величина типа Object.

### Описание

Функция **Buffer( )** возвращает буферную зону (объект типа "полигон") вокруг объекта.

Функция **Buffer( )** может создавать буферные зоны только вокруг одного объекта. Для создания буфера вокруг ряда объектов, используйте оператор **Оператор Create Object As Buffer**. Объект должен быть создан с использованием координатной системы MapBasic. Метод расчета буфера зависит от координатной системы. Для планов используется декартов метод вычисления. В остальных случаях будет использован сферический метод вычисления

### Пример:

Следующий фрагмент программы создает объект "прямая линия". Создается буферная зона вокруг линии в 10 миль.

```
Dim o_line, o_region As Object
o_line = CreateLine(-73.5, 42.5, -73.6, 42.8)
o_region = Buffer( o_line, 20, 10, "mi")
```

### См. также:

**Оператор Create Object**

---

## Функция ButtonPadInfo( )

### Назначение

Возвращает информацию о состоянии инструментальной панели.

### Синтаксис

**ButtonPadInfo( *pad\_name*, *attribute* )**

*pad\_name* – строковая величина, представляющая имя инструментальной панели; например, "Операции", "Пенал", "Программы" или "Команды" для стандартных панелей или то имя, которое было определено при создании новых панелей.

*attribute* – целочисленный код, управляющий типом результата функции.



Возвращаемая величина

Зависит от значения параметра *attribute*.

Описание

В зависимости от значения *attribute* , функция вернет следующую информацию об инструментальной панели. (Приведенные здесь коды определены в файле MAPBASIC.DEF.)

| код атрибута         | Результат ButtonPadInfo( ):  |
|----------------------|--|
| BTNPAD_INFO_FLOATING | Логическая величина (Logical). Возвращается значение TRUE, если панель плавающая, и значение FALSE, если панель находится в прикрепленном состоянии, то есть вытянута вдоль верхнего края рабочего окна.   |
| BTNPAD_INFO_NBTNS    | Целое число типа SmallInt. количество кнопок на панели.  |
| BTNPAD_INFO_WIDTH    | Целое число типа SmallInt, ширина панели, причем единица измерения равна одной кнопке (не включая разделитель).  |
| BTNPAD_INFO_WINID    | Целое число типа Integer. Идентификатор панели.  |
| BTNPAD_INFO_X        | X-координата верхнего левого угла инструментальной панели. Если панель находится в прикрепленном состоянии, то возвращается целое число от 0 и более, если панель плавающая, то число типа Float (при этом значение представляется в "бумажных" единицах). |
| BTNPAD_INFO_Y        | Y-координата верхнего левого угла инструментальной панели.   |

Пример:

```
Include "mapbasic.def"
If ButtonPadInfo("Операции", BTNPAD_INFO_FLOATING) Then
    '...если панель Операции плавающая, то следующий оператор прикрепит ее.
    Alter ButtonPad "Main" ToolbarPosition(0,0) Fixed
End If
```

См. также:

[Оператор Alter ButtonPad](#)

Оператор Call

Назначение

Вызывает sub-процедуру или внешнюю процедуру (DLL, XCMD).

### Предупреждение

Вы не можете использовать оператор **Call** в окне MapBasic.

### Синтаксис

```
Call subproc [ ( [ parameter ] [ , ... ] ) ]
```

*subproc* – имя подпрограммы;

*parameter* – параметр, передающий подпрограмме значение из основного модуля.

### Описание

Оператор **Call** используется для вызова подпрограммы. Подпрограммой может быть процедура, написанная на MapBasic, первым оператором которой является оператор **Оператор Sub...End Sub**. Так же с помощью оператора **Call** Windows может быть вызвана процедура из динамической библиотеки (DLL).

Если выполняется оператор **Call**, MapBasic начинает выполнять операторы соответствующей подпрограммы, пока не встретит оператор **End Sub** или **Оператор Exit Sub**. Затем MapBasic возвращается в программу, откуда был произведен вызов, к следующему оператору после **Call statement**. Оператор **Call** может обращаться только к тем процедурам, которые являются частью текста программы.

Программа MapBasic должна в главной процедуре содержать оператор **Оператор Declare Sub**. Это требование должно выполняться как по отношению к процедурам, организованным оператором Sub, так и к внешним процедурам (DLL и XCMD).

### Передача значений параметров

Подпрограмма может не иметь параметров. Тогда вызов такой процедуры может выглядеть так:

```
Call subroutine
```

Or (оператор Или)

```
Call subroutine( )
```

Если sub-процедура имеет параметры, то они могут быть объявлены двумя способами: ссылкой ("by reference") или значением ("by value"). По умолчанию параметр объявляется как "ссылка". Если параметр подпрограммы был определен таким образом, то при вызове sub-процедуры соответствующим параметром вызова должна быть переменная.

Такой параметр не только передает значение переменной в процедуру, но и возвращает значение обратно. Впоследствии, если sub-процедура изменила значения этих параметров, то также изменятся значения переменных, использовавшихся как параметры вызова в операторе Call.

Объявление величины параметра как значения осуществляется при помощи ключевого слова **ByVal** перед именем параметра в списке операторов **Sub** и **Declare Sub**. Если один из параметров подпрограммы был определен таким образом, то при вызове sub-процедуры

соответствующим параметром вызова может быть как постоянная величина, так и переменная или выражение. Однако в этом случае новое значение параметра не может быть возвращено обратно той же переменной.

Вызов внешних процедур Sub-процедуры могут пересылать как простые параметры, так и массивы переменных. При этом в списке вызова Вы пишете только имя массива без скобок.

### **Вызов внешних процедур**

Если оператор **Call** вызвал процедуру DLL, то MapBasic выполняет процедуру до тех пор, пока она не вернет управление. Сама DLL-процедура находится в отдельном файле (например, "KERNEL.EXE"). Этот файл с внешней процедурой должен быть доступен в то время, как MapBasic выполняет внешний вызов.

Аналогично, если оператор **Call** вызывает XCMD, то файл, содержащий XCMD, должен быть доступным. В вызове XCMD-команд не могут участвовать массивы переменных и переменные сложных типов, составленных оператором Type, в качестве параметров.

### Пример:

Процедура Cube возводит число в куб (в третью степень) и возвращает результат. Процедура имеет два параметра: первый для возводимого степень вещественного числа, второй для результата.

```
Declare Sub Cube(ByVal original As Float, cubed As Float)
    Dim x, result As Float
    Call Cube( 2, result)
    ' в результате получается: 8 (2 x 2 x 2)
    x = 1
    Call Cube( x + 2, result)
    ' в результате получается: 27 (3 x 3 x 3)
End Program
Sub Cube (ByVal original As Float, cubed As Float)
    ' Параметр "original" - число
    ' параметр "cubed" - результат вычисления.
    cubed = original ^ 3
End Sub
```

### См. также:

[Оператор Declare Sub](#), [Оператор Exit Sub](#), [Оператор Global](#), [Оператор Sub...End Sub](#)

---

## Функция CartesianArea( )

### Назначение

Возвращает площадь, используя вычисления в системе координат Широта/Долгота. используется декартовый алгоритм вычислений.

### Синтаксис

**CartesianArea**( *obj\_expr*, *unit\_name* )

*obj\_expr* - выражение, определяющее объект.

*unit\_name* – единицы измерения площади (например, "sq km" – квадратные километры).

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **CartesianArea( )** возвращает площадь географического объекта, указанного выражением *obj\_expr*.

Функция возвращает площадь области в единицах измерения, указанных в параметре *unit\_name*; к примеру, чтобы получить площадь в акрах, укажите “акры” в качестве значения параметра *unit\_name*. Смотрите описание оператора [Оператор Set Area Units](#), где описаны возможные единицы измерения.

Функция **CartesianArea( )** всегда будет возвращать значение площади, рассчитанное по декартовым алгоритмам. Величина -1 будет возвращаться для данных в координатах Широта/Долгота, поскольку такие данные не могут быть конвертированы в декартовы.

Только полигоны, эллипсы, прямоугольники и прямоугольники со скругленными углами имеют площадь. По определению, значение функции **CartesianArea( )** для точки, дуги, текста, линии или полилинии это ноль. Функция **CartesianArea( )** возвращает приблизительные результаты, когда применяется к скругленным прямоугольникам. MapBasic вычисляет площадь прямоугольников со скругленными углами как если бы они были обычными прямоугольниками.

### Примеры

Следующие примеры показывают, как функция **CartesianArea( )** может вычислять площадь одиночного картографического объекта. Обратите внимание, что выражение *tablename.obj* (аналогично *states.obj*) представляет географический объект из текущей строки указанной таблицы.

```
Dim f_sq_miles As Float
Open Table "counties"
Fetch First From counties
f_sq_miles = CartesianArea(counties.obj, "sq mi")
```

Вы можете также использовать функцию **CartesianArea( )** внутри оператора **Оператор Select**, как показано в следующем примере.

```
Select lakes, CartesianArea(obj, "sq km")
  From lakes Into results
```

**См. также:**

**Функция Area( ), Функция SphericalArea( )**

---

## Функция CartesianBuffer( )

### Назначение

Возвращает объект типа "область", представляющий собой буферную зону вокруг выбранного объекта (область, граница которой отстоит от границы объекта на заданное расстояние).

### Синтаксис

**CartesianBuffer( *inputobject*, *resolution*, *width*, *unit\_name* )**

*inputobject* – объектное выражение (выражение, результат которого есть величина типа Object);

*resolution* – число узлов многоугольника, принимаемого как окружность (число типа SmallInt);

*width* – это величина с плавающей запятой, представляющая радиус буфера; если ширина отрицательна, и если объект *inputobject* является замкнутым, то возвращаемый объект будет по размерам меньше исходного.

## Функция CartesianConnectObjects( )

---

*unit\_name* – имя единицы измерения расстояний *width* (например, "mi" – миля, "km" – километр).

### Возвращаемая величина

Полигон, объект типа Object

### Описание

Функция **CartesianBuffer( )** возвращает полигон, представляющий буферную зону. Она обрабатывает только один объект за раз.

Для создания буфера вокруг нескольких объектов, используйте **Оператор Create Object As Buffer**. Если задано отрицательное значение *width*, а объект является линейным (линией, полилинией или дугой) или же точкой, то используется положительное значение *width* и создается буфер больший чем объект.

Функция **CartesianBuffer( )** будет рассчитывать буферную зону, в предположении, что объект спроецирован на плоскость и используя ширину *width* для расчета декартового расстояния буферной зоны вокруг объекта.

Если *inputobject* задан в проекции Широта/Долгота, то будут использоваться сферические вычисления независимо от того, какая будет применяться функция, связанная с буфером.

### Пример:

Следующая программа создает линейный объект, затем создает буфер вокруг него. Буферная зона занимает 10 миль во всех направлениях вокруг линии.

```
Dim o_line, o_region As Object
o_line = CreateLine(-73.5, 42.5, -73.6, 42.8)
o_region = CartesianBuffer( o_line, 20, 10, "mi")
```

### См. также:

**Функция Buffer( ), Создание объектов на карте**

---

## Функция CartesianConnectObjects( )

### Назначение

Возвращает объект, представляющий кратчайшее или самое большое расстояние между двумя объектами.

### Синтаксис

**CartesianConnectObjects** (*object1*, *object2*, *min*)

*object1* и *object2* - выражения, указывающие на объекты.

*min* - логическое выражение. Если это выражение принимает значение "Да" (TRUE), функция вычисляет минимальное расстояние между объектами, а если выражение принимает значение "Нет" (FALSE), вычисляется максимальное расстояние.

**Возвращаемая величина**

Оператор возвращает объект полилинии, состоящий из одного сегмента и двух точек, и представляющий собой кратчайшее расстояние (`min == TRUE`), или самое большое расстояние (`min == FALSE`) между объектами *object1* и *object2*.

**Описание**

Одна точка полученной полилинии принадлежит объекту *object1*, а другая объекту *object2*. Обратите внимание, что расстояние между двумя объектами можно вычислить при помощи **Функция ObjectLen( )**. Если существует множество вариантов кратчайших или наибольших расстояний (например, полученный сегмент представляет одно расстояние, а существуют еще и другие сегменты равной длины), эти функции возвращают лишь один из вариантов. Не существует способа определить, является ли полученный объект полилинии единственным кратчайшим расстоянием.

**CartesianConnectObjects( )** возвращает объект-полилинию, соединяющую *object1* и *object2* по кратчайшему (`min == TRUE`) или наидлиннейшему (`min == FALSE`) пути, вычисленному в декартовых координатах. Если вычисления не могут быть сделаны, используя метод расчётов на плоскости (то есть координатная система MapBasic – Долгота/Широта), то эта функция выдаст сообщение об ошибке.

---

**Функция CartesianDistance( )****Назначение**

Возвращает расстояние между двумя точками.

**Синтаксис**

**CartesianDistance( *x1*, *y1*, *x2*, *y2*, *unit\_name* )**

*x1* и *x2* – х-координаты (долгота);

*y1* и *y2* – у-координаты (широта);

*unit\_name* – это строковая величина, соответствующая имени единиц измерения расстояния (например, "km").

**Возвращаемая величина**

Вещественное число типа Float.

**Описание**

Функция **CartesianDistance( )** вычисляет расстояние между двумя точками. Функция возвращает измеренное расстояние в единицах, указанных параметром *unit\_name*; например, что бы получить расстояние в милях, укажите "mi" как *unit\_name*. Список доступных единиц измерения смотрите в **Опепатор Set Distance Units**.

Функция **CartesianDistance( )** всегда возвращает значение, используя для расчетов декартовый алгоритм. Будет возвращено значение -1 для данных в системе координат Широта/Долгота, поскольку система Широта/Долгота не проективная и не декартовая.

## Функция CartesianObjectDistance( )

---

Координаты по осям x и y должны быть заданы в текущей системе координат MapBasic. По умолчанию MapInfo Professional предполагает, что координаты заданы в мировой системе координат (Широта/Долгота). Вы можете установить систему координат по умолчанию при помощи **Оператор Set CoordSys**.

### Пример:

```
Dim dist, start_x, start_y, end_x, end_y As Float
Open Table "cities"
Fetch First From cities
start_x = CentroidX(cities.obj)
start_y = CentroidY(cities.obj)
Fetch Next From cities
end_x = CentroidX(cities.obj)
end_y = CentroidY(cities.obj)
dist = CartesianDistance(start_x, start_y, end_x, end_y, "mi")
```

### См. также:

**Математические функции, Функция CartesianDistance( ), Функция Distance( )**

---

## Функция CartesianObjectDistance( )

### Назначение

Возвращает расстояние между двумя объектами.

### Синтаксис

**CartesianObjectDistance**(*object1*, *object2*, *unit\_name*)

*object1* и *object2* - выражения, указывающие на объекты.

*unit\_name* - строка, определяющая единицы измерения расстояния.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

**CartesianObjectDistance( )** возвращает минимальное расстояние между объектами *object1* и *object2*, вычисленное в декартовых координатах. Возвращаемое значение измеряется в единицах *unit\_name*. Если вычисления не могут быть сделаны, используя метод расчётов на плоскости (то есть координатная система MapBasic – Долгота/Широта), то эта функция выдаст сообщение об ошибке.



## Функция **CartesianObjectLen( )**

### Назначение

Возвращает географическую протяженность объекта линии или полилинии.

### Синтаксис

**CartesianObjectLen**( *obj\_expr*, *unit\_name* )

*obj\_expr* - выражение, определяющее объект.

*unit\_name* – это строковая величина, соответствующая имени единиц измерения расстояния (например, "km").

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **CartesianObjectLen( )** возвращает длину объекта. Помните, что ненулевые длины имеют только объекты линий и полилиний, для измерения периметра прямоугольника, эллипса или полигона, используйте [Функция Perimeter\( \)](#).

Функция **CartesianObjectLen( )** всегда будет возвращать значение, используя декартовый алгоритм. Величина -1 будет возвращаться для данных в системе Широта/Долгота, поскольку Широта/Долгота не проективна и не декартова.

Функция **CartesianObjectLen( )** возвращает длину, измеренную в единицах длины, определенных параметром *unit\_name* parameter; например, для получения длины в милях, укажите "mi" как *unit\_name*. См. описание единиц измерения в разделе [Оператор Set Distance Units](#).

### Пример:

```
Dim geogr_length As Float
Open Table "streets"
Fetch First From streets
geogr_length = SphericalObjectLen(streets.obj, "mi")
' geogr_length теперь содержит протяженность
' сегмента улицы в милях
```

### См. также:

[Функция SphericalObjectLen\( \)](#), [Функция CartesianObjectLen\( \)](#), [Функция ObjectLen\( \)](#)

## Функция `CartesianOffset( )`

### Назначение

Возвращает копию исходного объекта, смещённого на определенную дистанцию и угол, определенный функцией `Cartesian DistanceType`.

### Синтаксис

`CartesianOffset( object, angle, distance, units )`, где

*object* – объект, подвергаемый сдвигу.

*angle* – угол перемещения объекта.

*distance* – расстояние перемещения объекта.

*units* – строка, представляющая единицы измерения расстояния *distance*.

### Возвращаемая величина

Объект

### Описание

Эта функция создает новый объект, являющийся копией исходного и сдвинутый на расстояние *distance* вдоль угла *angle* (в градусах, где 0 градусов соответствует положительному направлению по оси X, а угол увеличивается против часовой стрелки). Строковое значение *unit*, похоже на то, которое используется для операторов [Функция ObjectLen\( \)](#) или [Функция Perimeter\( \)](#), это единицы измерения расстояния. Используется декартовый тип `DistanceType`. Если система координат для исходного объекта это Долгота/Широта, то будет выдано сообщение об ошибке, поскольку декартовые расстояния не годятся для Долготы/Широты. В случае ошибки возвращается объект `NULL`. Используется координатная система исходного объекта.

При проведении вычислений в сферической системе координат существуют некоторые соображения, неприменимые для декартовой системы координат. Так, при перемещении объекта по карте, построенной в проекции “Долгота/Широта”, его форма не изменится, но поменяется площадь. Это происходит потому, что один из параметров сдвига определяется в градусах, а реальное измеренное расстояние одного градуса в различных точках неодинаково.

Для функций `Offset`, фактическое смещение рассчитано в некоторой установленной точке на объекте (например, центр описывающего прямоугольника), и затем это значение преобразовано из исходных единиц в единицы текущей системы координаты. Если система координат - Широта/Долгота, преобразование в градусы использует фиксированное число десятичных знаков. Действительное преобразованное расстояние может быть неодинаковым в различных точках объекта. Расстояние от исходного объекта до его перемещённой копии будет точным только в этой фиксированной точке.

**Пример:**

```
CartesianOffset(Rect, 45, 100, "mi")
```

**См. также:**

**Функция `CartesianOffsetXY()`**

---

## Функция `CartesianOffsetXY()`

**Назначение**

Возвращает копию исходного объекта, смещённую на заданное расстояние по X и Y, используя декартовы координаты.

**Синтаксис**

```
CartesianOffsetXY( object, xoffset, yoffset, units )
```

*object* - объект, подвергаемый сдвигу.

*xoffset* и *yoffset* – расстояние вдоль осей x и y, на которые будет сделано смещение;

*units* - строка, определяющая единицы измерения расстояний.

**Возвращаемая величина**

Объект

**Описание**

Эта функция создает новый объект *object* который является копией исходного объекта, смещенного на расстояние *xoffset* вдоль оси X и на расстояние *yoffset* вдоль оси Y. Строковая величина *unit*, как и в операторах **Функция `ObjectLen()`** или **Функция `Perimeter()`**, является единицей измерения расстояния. Используется декартовый тип `DistanceType`. Если система координат для исходного объекта это Долгота/Широта, то будет выдано сообщение об ошибке, поскольку декартовые расстояния не годятся для Долготы/Широты. В случае ошибки возвращается объект `NULL`. Используется координатная система исходного объекта.

При проведении вычислений в сферической системе координат существуют некоторые соображения, неприменимые для декартовой системы координат. Так, при перемещении объекта по карте, построенной в проекции “Долгота/Широта”, его форма не изменится, но поменяется площадь. Это происходит потому, что один из параметров сдвига определяется в градусах, а реальное измеренное расстояние одного градуса в различных точках неодинаково.

Для функций `Offset`, фактическое смещение рассчитано в некоторой установленной точке на объекте (например, центр описывающего прямоугольника), и затем это значение преобразовано из исходных единиц в единицы текущей системы координаты. Если система координат - Широта/Долгота, преобразование в градусы использует фиксированное число десятичных знаков. Действительное преобразованное расстояние может быть неодинаковым в различных точках объекта. Расстояние от исходного объекта до его перемещённой копии будет точным только в этой фиксированной точке.

### Пример:

```
CartesianOffset(Rect, 45, 100, "mi")
```

### См. также:

[Функция `CartesianOffset\( \)`](#)

---

## Функция `CartesianPerimeter( )`

### Назначение

Возвращает периметр графического объекта.

### Синтаксис

```
CartesianPerimeter( obj_expr , unit_name )
```

*obj\_expr* - выражение, определяющее объект.

*unit\_name* – это строковая величина, соответствующая имени единиц измерения расстояния (например, "km").

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **`CartesianPerimeter( )`** вычисляет периметр объекта *obj\_expr*. Функция [Функция `Perimeter\( \)`](#) определена для следующих типов объектов: эллипсов, прямоугольников, скругленных прямоугольников и полигонов. Другие типы объектов имеют периметр нулевой длины.

Функция **`CartesianPerimeter( )`** всегда будет возвращать значение, вычисленное по декартовому алгоритму. Величина -1 будет возвращаться для данных в системе Широта/Долгота, поскольку Широта/Долгота не проективна и не декартова.

Возвращаемое функцией **`CartesianPerimeter( )`** значение длины периметра измеряется в единицах длины, определенных параметром *unit\_name*; например, для получения длины в милях, укажите "mi" в качестве *unit\_name*. См. описание единиц измерения в разделе [Оператор Set Distance Units](#). **`CartesianPerimeter( )`** возвращает приближенный результат при измерении периметра скругленного прямоугольника. MapBasic вычисляет периметр прямоугольников со скругленными углами как если бы они были обычными прямоугольниками.

### Пример:

Следующий пример показывает как Вы можете использовать функцию **`CartesianPerimeter( )`** для определения периметра географического объекта.

```
Dim perim As Float
Open Table "world"
Fetch First From world
```

```
perim = CartesianPerimeter(world.obj, "km")
' переменная perim теперь содержит
' периметр полигона
' соответствующего первой записи таблицы World.
```

Вы можете также использовать функцию **CartesianPerimeter( )** внутри оператора **Оператор Select**. Следующий оператор **Оператор Select** извлекает информацию из таблицы States и сохраняет результаты во временной таблице Results. Поскольку оператор **Оператор Select** включает функцию **CartesianPerimeter( )**, таблица Results будет включать колонку, показывающую периметр каждого штата.

```
Open Table "states"
Select state, CartesianPerimeter(obj, "mi")
  From states
  Into results
```

**См. также:**

**Функция CartesianPerimeter( ), Функция SphericalPerimeter( ), Функция Perimeter( )**

---

## Функция Centroid( )

### Назначение

Возвращает центральную точку объекта (центроид).

### Синтаксис

```
Centroid( obj_expr )
```

*obj\_expr* - выражение, определяющее объект.

### Возвращаемая величина

точка

### Описание

Функция **Centroid( )** возвращает точечный объект, расположенный в центре объекта, представленного параметром *obj\_expr*. В MapInfo области не всегда имеют центроид в геометрическом центре объекта. От расположения центроида области зависит расположение подписей, полученных в результате автоподписывания, точек геокодирования и расположения графиков и диаграмм на тематическом слое Карты. Вы всегда можете передвинуть центроид области в нужное место на изменяемом слое Карты в режиме изменения формы.

Если объект *obj\_expr* является точечным, то функция **Centroid( )** его и вернет. Если объект *obj\_expr* – линия, то в результате применения **Centroid( )** получится точка между ее концами.

Если *obj\_expr* представляет объект типа "полилиния", то **Centroid( )** возвращает середину среднего сегмента полилинии.

## Функция CentroidX( )

---

Для других типов объектов *obj\_expr* функция **Centroid( )** возвращает точку с координатами действительного центроида. Если объект прямоугольник, дуга, текст или эллипс, то точка будет равноудалена между верхней и нижней границами, и между правой и левой границами. Для объекта типа "область" центроид всегда лежит внутри объекта, но не обязательно находится в геометрическом центре объекта.

### Пример:

```
Dim pos As Object
Open Table "world"
Fetch First From world
pos = Centroid(world.obj)
```

### См. также:

[Оператор Alter Object](#), [Функция CentroidX\( \)](#), [Функция CentroidY\( \)](#)

---

## Функция CentroidX( )

### Назначение

Возвращает координату центральной точки (центроида) по оси X.

### Синтаксис

**CentroidX**( *obj\_expr* )

*obj\_expr* – выражение, результат которого есть величина типа Object.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **CentroidX( )** возвращает X-координату (или долготу) центроида точки объекта. См. раздел [Функция Centroid\( \)](#), где приводится концепция центроида для разных географических объектов (линий, областей, и т.д.).

Информация о координатах возвращается в текущей системе координат MapBasic; по умолчанию MapBasic использует координатную систему Долгота/Широта. Оператором [Оператор Set CoordSys](#) можно менять координатную систему.

## Примеры

В следующем примере функция **CentroidX( )** вычисляет долготу центроида одного географического объекта.

```
Dim x As Float
Open Table "world"
Fetch First From world
x = CentroidX(world.obj)
```

Вы можете использовать функцию **CentroidX( )** внутри **Оператор Select**. Следующий оператор **Оператор Select** извлекает информацию из таблицы "World" и помещает ее во временную таблицу "Results". Поскольку **Оператор Select** включает в себя функцию **CentroidX( )** и **Функция CentroidY( )**, таблица "Results" будет включать колонки, показывающие широту и долготу для центроидов всех штатов.

```
Open Table "world"
Select country, CentroidX(obj), CentroidY(obj)
  From world Into results
```

См. также:

**Функция Centroid( )**, **Функция CentroidY( )**, **Оператор Set CoordSys**

---

## Функция CentroidY( )

### Назначение

Возвращает координату центральной точки (центроида) по оси Y.

### Синтаксис

```
CentroidY( obj_expr )
```

*obj\_expr* - выражение, определяющее объект.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **CentroidY( )** возвращает X-координату (или долготу) центроида точки объекта. См. раздел **Функция Centroid( )**, где приводится концепция центроида для разных географических объектов (линий, областей, и т.д.).

Информация о координатах возвращается в текущей системе координат MapBasic; по умолчанию MapBasic использует координатную систему Долгота/Широта. Оператором **Оператор Set CoordSys** можно менять координатную систему.

### Пример:

```
Dim y As Float
Open Table "world"
Fetch First From world
y = CentroidY(world.obj)
```

### См. также:

[Функция Centroid\( \)](#), [Функция CentroidX\( \)](#), [Оператор Set CoordSys](#)

---

## Предложение CharSet

### Назначение

Определяет набор кодов, используемый в MapBasic для интерпретирования символов.

**Внимание:** См. также *Руководство пользователя* MapInfo Professional 9.0.

### Синтаксис

**CharSet** *char\_set*

*char\_set* – строковая величина, содержащая имя кода, например, "ANSI".

### Описание

Предложение **CharSet** используется для перекодировки символов в операциях чтения из файла или записи в файл или таблицы. Note that **CharSet** is a clause, not a complete statement. Various file-related statements, such as the [Оператор Open File](#), can incorporate optional **CharSet** clauses.

### Для чего нужна система кодов?

В компьютере каждому символу клавиатуры сопоставлен некоторый числовой код. Например, английскому символу "A" на Вашей клавиатуре соответствует код символа 65. Набором символов (или кодировкой символов) называется совокупность символов, доступных в Вашем компьютере и сопоставленных им числовых кодов.

В разных странах могут использоваться различные наборы символов. Например, Windows версии для Северной Америки и Западной Европы использует код 126 для представления знака градуса (°), тогда как Windows другой языковой версии может использовать этот код для другого символа.

Для определения системы кодов символов используйте вызов функции `SystemInfo(SYS_INFO_CHARSET)`.

### Как кодировка символов влияет на программу MapBasic?

Если Ваши файлы используют только стандартные символы ASCII с номерами от 32 (пробел) до 126 (тильда), то Вам не нужно беспокоиться о конфликтах порождаемых наборами символов, и не нужно использовать предложение **CharSet**.



Даже если Вы используете “специальные” символы (т.е. вне диапазона с 32 по 126), но при этом работаете на одно и той же вычислительной платформе (например, Windows), использующей только один набор символов, то задавать предложение **CharSet** не нужно.

Однако, если файл включает интернациональные символы и Вы хотите его перенести в другую операционную среду, которая использует другую кодировку, Вам не обойтись без предложения **CharSet** для сохранения целостности Ваших данных.. Предложение **CharSet** показывает, какой набор символов использовался в момент создания таблицы.

Предложение **CharSet** имеет только один строковый параметр, возможные значения которого приведены в следующей таблице: В следующей таблице приведены все доступный наборы символов.

| Character Set  | Комментарии                              |
|--|--|
| “Neutral”  | нет преобразования кодов                 |
| “ISO8859_1”  | ISO 8859-1 (UNIX)                        |
| “ISO8859_2”  | ISO 8859-2 (UNIX)                        |
| “ISO8859_3”  | ISO 8859-3 (UNIX)                        |
| “ISO8859_4”  | ISO 8859-4 (UNIX)                        |
| “ISO8859_5”  | ISO 8859-5 (UNIX)                        |
| “ISO8859_6”  | ISO 8859-6 (UNIX)                        |
| “ISO8859_7”  | ISO 8859-7 (UNIX)                        |
| “ISO8859_8”  | ISO 8859-8 (UNIX)                        |
| “ISO8859_9”  | ISO 8859-9 (UNIX)                        |
| “PackedEUCJapanese”  | Unix, японский стандарт                  |
| “WindowsLatin2”<br>“WindowsArabic”<br>“WindowsCyrillic”<br>“WindowsGreek”<br>“WindowsHebrew”<br>“WindowsTurkish” | Windows, стандарт Восточной Европы       |
| “WindowsTradChinese”   | Windows, стандарт традиционный китайский |
| “WindowsSimpChinese”   | Windows, стандарт упрощенный китайский   |
| “WindowsJapanese”  |  |
| “WindowsKorean”  |  |

| Character Set | Комментарии   |
|---------------|---|
| "CodePage437" | DOS Code Page 437 = расширенная кодировка ASCII для IBM |
| "CodePage850" | DOS Code Page 850 = многоязычная                        |
| "CodePage852" | DOS Code Page 852 = Восточная Европа                    |
| "CodePage855" | DOS Code Page 855 = Кириллица                           |
| "CodePage857" |   |
| "CodePage860" | DOS Code Page 860 = Португальская                       |
| "CodePage861" | DOS Code Page 861 = Исландская                          |
| "CodePage863" | DOS Code Page 863 = Французско-Канадская                |
| "CodePage864" | DOS Code Page 864 = Арабская                            |
| "CodePage865" | DOS Code Page 865 = Норвежская                          |
| "CodePage869" | DOS Code Page 869 = Современная Греческая               |
| "LICS"        | Кодировка Lotus версии 1,2                              |
| "LMBCS"       | Кодировка Lotus версии 3,4                              |

**Внимание:** Заметим, что в операторе **Оператор Open Table** нет необходимости использовать предложение CharSet. Для каждой таблицы информация об используемом наборе символов хранится в файле TAB. При открытии таблицы MapInfo Professional прочитывает эту информацию прямо из TAB-файла, после чего автоматически выполняет нужные операции перекодировки.

Вы можете заставить MapInfo Professional сохранить файл в определенной кодировке; для этого включите предложение **CharSet** в **Оператор Commit Table**.

### Синтаксис предложения в языке MapBasic версии 2.x

MapBasic версии 2.x поддерживает только три системы кодов: "XASCII", "ANSI" и "MAC". Тексты программ, написанные во времена этой версии, и которые используют эти имена систем кодов, могут быть откомпилированы и запущены в MapBasic версии 3.0. Однако, использовать имена систем кодов из версии 2.x не рекомендуется.

Предложение CharSet "XASCII" соответствует предложению CharSet "CodePage437".

Предложение CharSet "MAC" соответствует предложению CharSet "MacRoman".

Если Вы работаете в Windows, то предложение CharSet "ANSI" задает кодировку, которая используется в Windows. Пример: текстовый файл PARCEL.TXT был создан в DOS, поэтому в следующем примере используется код "CodePage437".

```
Open File "parcel.txt"
  For INPUT As #1
    CharSet "CodePage437"
```

**См. также:**

[Оператор Commit Table](#), [Оператор Create Table](#), [Оператор Export](#), [Оператор Open File](#), [Оператор Register Table](#)

---

## Функция ChooseProjection\$ ( )

### Назначение

Показывает диалог выбора проекции и возвращает координатную систему, выбранную пользователем.

### Синтаксис

**ChooseProjection\$**( *initial\_coordsys*, *get\_bounds* )

*initial\_coordsys* – строковая величина из предложения [Оператор CoordSys](#). Она используется для установки той координатной системы, которая первый раз выбирается в диалоге. Если *initial\_coordsys* является пустой или предложение соответствует неправильной координатной системе, то по умолчанию в первом указании координатной системы используется система широта/долгота.

*get\_bounds* – логическая величина, которая определяет, какие границы ввести пользователю при использовании плановых координат. Если *get\_bounds* истинно, то появляется диалог, в котором надо определить границы карты. Если это выражение ложно, то диалог не появляется и используются границы, заданные по умолчанию.

### Описание

Эта функция отображает диалог выбора проекции и возвращает выбранную систему координат в виде строковой величины. Возвращаемая строковая величина имеет тот же формат, что и предложение. Используйте эту функцию, если Вы хотите позволить пользователю установить проекцию внутри Вашего приложения.

### Пример:

```
Dim strNewCoordSys As String
strNewCoordSys = ChooseProjection$ ( "", True)
strNewCoordSys = "Set " + strNewCoordSys
Run Command strNewCoordSys
```

**См. также:**

[Функция MapperInfo\( \)](#)

## Функция Chr\$( )

### Назначение

Возвращает символ, соответствующий заданному коду символа.

### Синтаксис

**Chr\$( num\_expr )**

*num\_expr* – целочисленное выражение 0 от 255 (для двухбайтных кодировок от 0 до 65,535), включительно.

### Возвращаемая величина

Строка

### Описание

Функция **Chr\$( )** возвращает строковое значение длиной в один символ, которое соответствует коду, полученному в результате вычисления выражения *num\_expr*. Для большинства систем параметр *num\_expr* должен быть положительным целым числом от 0 до 255. В операционных платформах, поддерживающих систему двухбайтовых кодов (например, Windows Japanese), параметр *num\_expr* может быть числом от 0 до 65535.

**Внимание:** Вычислительные платформы, на которых работает MapInfo Professional, имеют общие коды от 32 (пробел) до 126 (тильда). В этот диапазон входят все цифры и буквы латинского алфавита. Буквы русского алфавита в разных системах имеют разные коды.

Если в результате вычисления *num\_expr* получается дробное число, MapBasic округляет его до целого.

12-й символ в Windows соответствует переводу на новую страницу (form-feed). Оператор `Print Chr$(12)` удобно использовать для очистки окна "Сообщения". Символ под номером 10 создает новую строку; см. пример ниже.

34-й символ в Windows соответствует двойной кавычке ("). Когда строка включает `Chr$(34)`, MapBasic вставит в текст кавычки.

### Ошибки:

Функция вернет код ошибки `ERR_FCN_ARG_RANGE`, если значение аргумента выходит за пределы, заданные при его определении.

### Пример:

```
Dim s_letter As String * 1
s_letter = Chr$(65)
Note s_letter ' Этот пример показывает символ "А"
Note "А это сообщение " + Chr$(10) + "в две строки."
```

См. также:

[Функция Asc\( \)](#)

---

## Оператор Close All

### Назначение

Закрывает все открытые таблицы.

### Синтаксис

**Close All** [ **Interactive** ]

### Описание

Выполняя оператор **Close All**, MapBasic закрывает все открытые таблицы, включая те, которые изменялись. При этом все изменения после закрытия теряются. Предупреждения об этом на экране не будет. Если Вы не хотите потерять текущие изменения в таблицах, используйте директиву **Interactive** для вывода на экран диалога, предлагающего пользователю сохранить или игнорировать изменения.

См. также:

[Оператор Close Table](#)

---

## Оператор Close Connection

### Назначение

Закрывает соединение, открытое после выполнения [Оператор Open Connection](#).

### Синтаксис

**Close Connection** *connection\_handle*

*connection\_handle* – целое, определяющее значение, которое возвращает [Оператор Open Connection](#).

### Описание

Оператор Close Connection закрывает заданное соединение, определенное дескриптором, который возвращает [Оператор Open Connection](#). Любые параметры, связанные с выполнением службы, подключенной по этому соединению, будут утеряны.

См. также:

[Оператор Open Connection](#)

### Оператор Close File

#### Назначение

Закрывает открытый файл.

#### Синтаксис

```
Close File [ # ] filenum
```

*filenum* – целое число, идентификатор открытого файла.

#### Описание

Оператор **Close File** используется для закрытия файлов, открытых оператором **Оператор Open File**.

**Внимание:** Операторы **Оператор Open File** и **Оператор Close File** работают с файлами, но не с таблицами MapInfo. Для обращения к таблицам MapBasic имеет другие операторы; например, оператор **Оператор Open Table** используется для открытия таблицы.

#### Пример:

```
Open File "cxdata.txt" For INPUT As #1
'
' файл открыт, теперь его можно закрыть:
'
Close File #1
```

#### См. также:

**Оператор Open File**

---

### Оператор Close Table

#### Назначение

Закрывает открытую таблицу.

#### Синтаксис

```
Close Table table [ Interactive ]
```

*table* – имя открытой таблицы.

#### Описание

Оператор **Close Table** используется для закрытия таблицы. Для закрытия сразу всех открытых таблиц используется оператор **Оператор Close All**.

Когда таблица показывается в одном или нескольких окнах Графиков или Списков, то при закрытии таблицы эти окна также закрываются. Если оператор **Close Table** автоматически закрывает окне Карты, если закрывается последняя, отображаемая в окне Карты, таблица. Если применить оператор **Close Table** к связанной таблице, которая имеет несохраненные изменения, то MapInfo запомнит изменения до следующего сеанса работы.

### Сохранение изменений в таблице

Если Вы не зададите слово **Interactive**, изменения, которые не были сохранены на диске, при закрытии будут утеряны. Сообщений об этом не выводится. Если Вы не хотите потерять текущие изменения в таблицах, используйте слово **Interactive** для вывода на экран диалога, предлагающего пользователю сохранить или игнорировать изменения.

Если Вы хотите гарантировать несохранение изменений в таблице, опустите слово **Interactive** или используйте оператор **Оператор Rollback** перед **Close Table**. Если же, наоборот, Вы хотите гарантировать сохранение изменений в таблице, выполните оператор **Оператор Commit Table** перед **Close Table**. Для определения, есть ли в таблице изменения, не сохраненные на диске, используется функция **Функция TableInfo()** (*table, TAB\_INFO\_EDITED*).

### Сохранение тематических и косметических объектов

Если Вы закрываете таблицу, графические объекты которой показываются на единственном некосметическом слое окна Карты, то будет закрыто и это окно. Чтобы сохранить объекты Косметического и/или тематического слоев из этого окна, используйте ключевое слово **Interactive**.

Если **Interactive** опущено, выполнение оператора **Close Table** не сопровождается диалогом с предложением сохранить тематические и косметические объекты. Если включить слово **Interactive**, возникнет диалог с запросом о сохранении тематических и косметических объектов, если это имеет смысл. (Сообщение не будет выводиться, если на Карте нет тематических слоев и объектов в Косметическом слое.)

Примеры

```
Open Table "world"
' ... когда таблица WORLD не нужна,
' закройте ее оператором:
Close Table world

Чтобы отменить выбор строк, закройте таблицу Selection.

Close Table Selection
```

См. также:

[Оператор Close All](#), [Оператор Commit Table](#), [Оператор Open Table](#), [Оператор Rollback](#), [Функция TableInfo\( \)](#)

Оператор Close Window

Назначение

Закрывает или скрывает окна.

Синтаксис

```
Close Window window_spec [ Interactive ]
```

*window\_spec* – имя окна (такое как Ruler ), код окна (такой как WIN\_RULER ) или целое число, определяющее окно.

Описание

Оператор **Close Window** закрывает или прячет окна MapInfo Professional.

Для закрытия документального окна (окна Карты, Списка, Графика и Отчета) необходимо задать его целым числом в параметре *window\_spec*. Это число помогут Вам определить операторы [Функция FrontWindow\( \)](#) и [Функция WindowID\( \)](#).

Чтобы закрыть специальное окно или инструментальную панель, надо в *window\_spec* задать специальное имя окна или его код. Вы можете окно "Линейка" задать именем Ruler или кодом WIN\_RULER.

В следующей таблице приводятся имена и коды для параметра *window\_spec*:

| Значение <i>window_spec</i> | Описание окна  |
|-----------------------------|--|
| MapBasic                    | Окно MapBasic Можно также задавать кодом: WIN_MAPBASIC.        |
| Help                        | Окно Справочной системы Можно также задавать кодом: WIN_HELP.  |
| Статистика                  | Окно "Статистика". Можно также задавать кодом: WIN_STATISTICS. |
| Легенда                     | Окно "Легенда". Можно также задавать кодом: WIN_LEGEND.        |



| Значение<br>window_spec | Описание окна   |
|-------------------------|---|
| Информация              | Окно "Информация", которое открывается инструментом Информация. Можно также задавать кодом: WIN_INFO.                     |
| Линейка                 | Окно "Линейка", которое открывается инструментом Линейка. Можно также задавать кодом: WIN_RULER.                          |
| Окно Сообщение          | Окно "Сообщение", которое открывается, если использовать <b>Оператор Print</b> . Можно также задавать кодом: WIN_MESSAGE. |

**Сохранение тематических и косметических объектов**

Если Вы закрываете Карту, то пользователь может захотеть сохранить объекты с Косметического или тематических слоев. Чтобы сохранить объекты Косметического и/или тематического слоев из этого окна, используйте ключевое слово **Interactive**.

Если **Interactive** опущено, выполнение оператора **Close Window** не сопровождается диалогом с предложением сохранить тематические и косметические объекты. Если включить слово **Interactive**, возникнет диалог с запросом о сохранении тематических и косметических объектов, если это имеет смысл. (Сообщение не будет выводиться, если на Карте нет тематических слоев и объектов в Косметическом слое.)

**Пример:**

```
Close Window Legend
```

**См. также:**

**Оператор Open Window, Оператор Print, Оператор Set Window**

**Функция ColumnInfo( )**

**Назначение**

Возвращает информацию о колонке в открытой таблице.

**Синтаксис**

```
ColumnInfo( { tablename | tablenum } ,  
           { columnname | "COLn" } , attribute )
```

- tablename* – имя открытой таблицы;
- tablenum* – целое число, идентифицирующее таблицу;
- columnname* – имя колонки в этой таблице;
- n* – целое число, номер колонки в таблице;
- attribute* – код, управляющий типом результата функции.

### Возвращаемая величина

Зависит от значения параметра *attribute*.

### Описание

Функция **ColumnInfo( )** возвращает информацию об одной колонке в открытой таблице.

Первый параметр функции задает таблицу (именем или идентификатором). Второй параметр определяет колонку. Параметр *attribute* должен принимать значения целочисленного кода, задающего тип возвращаемой функцией информации. В следующей таблице в первой колонке приводятся имена кодов для использования в качестве параметра *attribute*.

| Параметры          | ColumnInfo( ) возвращает код:  |
|--------------------|--|
| COL_INFO_NAME      | Имя колонки (строка).  |
| COL_INFO_NUM       | Номер колонки (короткое целое число).  |
| COL_INFO_TYPE      | Тип колонки (короткое целое число). Смотрите вторую таблицу.   |
| COL_INFO_WIDTH     | Ширина символьного или десятичного поля (короткое целое число). Используется только для символьных и десятичных колонок. |
| COL_INFO_DECPLACES | Число знаков после десятичной точки в поле десятичного типа (короткое целое число).                                      |
| COL_INFO_INDEXED   | Признак индексирования колонки (логическая величина).  |
| COL_INFO_EDITABLE  | Признак наличия изменений значений колонки (логическая величина)   |

Если функция **ColumnInfo( )** имеет значение параметра *attribute* равным COL\_INFO\_TYPE, MapBasic возвращает одно из следующих:

| ColumnInfo( ) возвращает код: | Тип колонки:                       |
|-------------------------------|------------------------------------|
| COL_TYPE_CHAR                 | Символьный                         |
| COL_TYPE_DECIMAL              | Десятичный с фиксированной запятой |
| COL_TYPE_FLOAT                | Вещественный                       |
| COL_TYPE_INTEGER              | Целочисленный (4 байта)            |
| COL_TYPE_SmallInt             | Короткое целое число (2 байта)     |
| COL_TYPE_DATE                 | Дата                               |

| <b>ColumnInfo( )</b><br>возвращает код: | <b>Тип колонки:</b>  |
|---|--|
| COL_TYPE_LOGICAL                        | Логический (TRUE или FALSE)  |
| COL_TYPE_GRAPHIC                        | Специальный тип колонки "Obj" (представляет графические объекты, присоединенные к таблице) |

Все определения из этих таблиц находятся в файле MAPBASIC.DEF. Если Вы будете использовать имена, включите в начало Вашей программы оператор Include "MAPBASIC.DEF".

#### Ошибки:

ERR\_TABLE\_NOT\_FOUND, если не найдена данная таблица;

ERR\_FCN\_ARG\_RANGE, если значение аргумента выходит за пределы, заданные при его определении.

#### Пример:

```
Include "MAPBASIC.DEF"
Dim s_col_name As String, i_col_type As SmallInt
Open Table "world"
s_col_name = ColumnInfo("world", "col1", COL_INFO_NAME)
i_col_type = ColumnInfo("world", "col1", COL_INFO_TYPE)
```

#### См. также:

**Функция NumCols( ), Функция TableInfo( )**

## Функция Combine( )

#### Назначение

Возвращает либо объект типа "область", либо "полилиния", представляющий объединение двух объектов. Не применяется к объектам типа "текст".

#### Синтаксис

**Combine( object1, object2 )**

*object1, object2* – два объектных выражения, результатом вычисления которых должны быть либо два объекта, имеющих площадь (например, область и окружность), либо два линейных объекта (например, прямая линия и полилиния).

#### Возвращаемая величина

Объект, являющийся объединением *object1* и *object2*.

## Описание

Функция **Combine( )** возвращает объект, являющийся результатом географического объединения двух объектов. Объединение двух смежных объектов удаляет границу между ними.

Функция **Combine( )** обновлена и теперь поддерживает комбинирование разнородных объектов, а также может обрабатывать точки, группы точек и коллекции. Ранее, оба исходных объекта должны были быть либо линейными (Линии, Полилинии или Дуги), порождая в результате полилинию; либо замкнутыми (Области, Прямоугольники, Скругленные или Эллипсы), порождая в области. Не была разрешена комбинация разнородных объектов с участием Точек, Групп точек и Коллекций. К текстовым объектам функция **Combine( )** неприменима.

Точки, Группы точек и Коллекции, введенные в MapInfo Professional 6.5, могут участвовать в отдельных операциях комбинации. В следующей таблице приведены все возможные варианты комбинирования объектов и типы результатов:

| Тип исходного объекта   | Тип исходного объекта                               | Тип результата |
|---|---|----------------|
| Точка или Группа точек  | Точка или Группа точек                              | Группа точек   |
| Линейный (Линия, Полилиния, Дуга)                                     | Линейный  | Полилиния      |
| Замкнутый (Область, Прямоугольник, Скругленный Прямоугольник, Эллипс) | Замкнутый   | Область        |
| Точка, Группа точек, Линейный, Замкнутый, Коллекция                   | Точка, Группа точек, Линейный, Замкнутый, Коллекция | Коллекция      |

Результат функции **Combine( )** такой же как при выполнении географического объединения командой **Objects > Combine** за тем исключением, что команда **Combine** изменяет исходные объекты, а функция **Combine( )** не меняет *object1* и *object2*. Кроме того, функция **Combine( )** не проводит операцию обобщения данных.

Объект, полученный в результате выполнения функции **Combine( )**, оформляется так же, как объект *object1*. Коллекция объектов, полученная в результате, унаследует от объекта *object1* все возможные компоненты стиля, а остальные компоненты останутся такими же, как у *objects2*. Например, если *object1* - это область, а *object2* - это полилиния, то к коллекции, полученной в результате, будет применена штриховка и окаймление от объекта *object1* и стиль линии от объекта *object2*.

**См. также:**

**Оператор Objects Combine**

## CommandInfo( ) функция

### Назначение

Возвращает информацию о последних внутрисистемных событиях.

### Синтаксис

`CommandInfo( attribute )`, где

*attribute* – целочисленный код типа информации, которая будет возвращена.

### Возвращаемая величина

Целое число или число с плавающей запятой, или строка, или логическая величина.

### Описание

Функция **CommandInfo( )** возвращает информацию о последних событиях в среде MapInfo – например, какие изменения произошли во временной таблице "Selection", в какое место окна указал пользователь, использовал ли он только мышку или нажал на кнопку мышки в комбинации с клавишей SHIFT и так далее.

### Вызов после закрытия диалога

Вызывая функцию **CommandInfo( )** сразу после закрытия окна диалога, Вы можете использовать в качестве параметра *attribute* один из следующих кодов:

| <i>attribute</i> -код | Результат <b>CommandInfo( attribute )</b> :   |
|-----------------------|---|
| CMD_INFO_DLG_OK       | Величина типа Logical: TRUE - если в диалоговом окне была выбрана кнопка <b>OK</b> ; FALSE - если пользователь отменил диалог, нажав на кнопку <b>Cancel</b> или на клавишу ESC. (Этот код в функции предназначен только для информации о диалоговых окнах, вызываемых оператором <b>Оператор Dialog</b> .) |
| CMD_INFO_STATUS       | Величина типа Logical: TRUE – если пользователь позволил процессу, снабженному диалогом со шкалой (смотрите оператор <b>ProgressBar</b> ), завершиться самостоятельно; FALSE – если пользователь в этом диалоге нажал на кнопку <b>Cancel</b> , прекращающую выполнение.                                    |

### Вызов из обработчиков новых меню и диалогов

Если приложение строит свои меню с элементами, вызывающими обработчик, или строит диалог, элементы которого вызывают процедуры-обработчики, то в этих процедурах Вы можете использовать функцию **CommandInfo( )** со следующими кодами *attribute*:

| Значения attribute | Результат CommandInfo( attribute ):  |
|--------------------|--|
| CMD_INFO_MENUITEM  | Целое число (Integer), представляющее идентификатор элемента меню, который запустил этот обработчик. Этот код используется только внутри процедуры-обработчика элемента меню.  |
| CMD_INFO_DLG_DBL   | Величина типа Logical: TRUE - если пользователь использовал при указании на элемент списка элемент ListBox или MultiListBox двойное нажатие на клавишу мышки. Этот код используется только внутри процедуры-обработчика элемента диалога, построенного приложением MapBasic. |

### Вызов из обработчиков системных событий

Если Ваше приложение имеет процедуры-обработчики системных событий (например, такую как SelChangedHandler), то в этих процедурах Вы можете использовать функцию **CommandInfo( )** со следующими кодами. Более подробную информацию Вы можете найти в описании процедур SelChangedHandler, RemoteMsgHandler, WinChangedHandler и WinClosedHandler. Из процедуры SelChangedHandler:

| код атрибута       | Результат CommandInfo( attribute ):   |
|--------------------|---|
| CMD_INFO_SELTYPE   | 1, если строка была добавлена в выборку;<br>2, если строка была исключена из предыдущей выборки;<br>3, если все строки в таблице выбраны;<br>4, если все строки таблицы были исключены из предыдущей выборки. |
| CMD_INFO_ROWID     | Целое число (Integer) – номер строки, которая была добавлена в выбор или исключена из выбора (работает только, если одна строка была выбрана или исключена из выбора).  |
| CMD_INFO_INTERRUPT | Логическая величина (Logical). TRUE, если пользователь прервал выбор клавишей ESC, FALSE – иначе.   |

Из процедуры **Процедура RemoteMsgHandler**, или **Функция RemoteQueryHandler( )**, или **Процедура RemoteMapGenHandler**:

| код атрибута | Результат CommandInfo( attribute ):   |
|--------------|---|
| CMD_INFO_MSG | Строка, представляющая собой “выполняемое” сообщение или имя элемента послания MapInfo от программы-клиента. Более подробное описание см. в разделах <a href="#">Процедура RemoteMsgHandler</a> , <a href="#">Функция RemoteQueryHandler( )</a> или <a href="#">Процедура RemoteMapGenHandler</a> . |

Из процедуры [Процедура WinChangedHandler](#) или [Процедура WinClosedHandler](#):

| код атрибута | Результат CommandInfo( attribute ):  |
|--------------|--|
| CMD_INFO_WIN | Целое число (Integer), идентификатор окна, в котором было изменение или которое было закрыто. Более подробное описание см. в разделах <a href="#">Процедура WinChangedHandler</a> или <a href="#">Процедура WinClosedHandler</a> . |

Из процедуры [Процедура ForegroundTaskSwitchHandler](#):

| код атрибута         | Результат CommandInfo( attribute ):   |
|----------------------|---|
| CMD_INFO_TASK_SWITCH | Целое число (Integer), указывающее, что MapInfo либо только что стало активным приложением, либо только что перестало быть активным. Результатом может быть один из кодов: \nSWITCHING_INT0_MAPINFO (если MapInfo получает фокус)\nSWITCHING_OUT_OF_MAPINFO (если MapInfo теряет фокус) |

### Вызов после операции поиска

После выполнения оператора [Оператор Find](#), параметр функции *attribute* может принимать одно из следующих значений:

| код атрибута             | Результат CommandInfo( attribute ):  |
|--------------------------|--|
| CMD_INFO_FIND_RC         | Целое число (Integer), код успеха поиска <a href="#">Оператор Find</a> .               |
| CMD_INFO_FIND_ROWID      | Целое число (Integer), номер строки (Row ID ), которой соответствует найденный объект. |
| CMD_INFO_X or CMD_INFO_Y | Действительное число, координата (по оси X или Y) найденного места.                    |

**Вызов из процедуры-обработчика инструмента**

Внутри процедуры-обработчика **Процедура ToolHandler**, Вы можете использовать следующие коды:

| код <i>атрибута</i> | Результат CommandInfo( <i>attribute</i> ):   |
|---------------------|--|
| CMD_INFO_X          | <p>X-координата точки, где была нажата кнопка мыши:</p> <ul style="list-style-type: none"> <li>на Карте, долгота в установленных единицах и в соответствии с текущей в MapBasic системой координат;</li> <li>в Списке, номер колонки в окне (единица соответствует самой левой колонке в окне);</li> <li>в Отчете, расстояние между левым краем Отчета и указанной точкой в текущих "бумажных" единицах (ноль соответствует точке на левом крае листа Отчета).</li> </ul>    |
| CMD_INFO_Y          | <p>Y-координата точки, где была нажата кнопка мыши:</p> <ul style="list-style-type: none"> <li>на Карте, широта в установленных единицах и в соответствии с текущей в MapBasic системой координат;</li> <li>в Списке, номер строки в окне (единица соответствует самой верхней строке в окне);</li> <li>в Отчете, расстояние между верхним краем Отчета и указанной точкой в текущих "бумажных" единицах (ноль соответствует точке на верхнем крае листа Отчета);</li> </ul> |
| CMD_INFO_X2         | X-координата точки, в которой пользователь отпустил кнопку мыши. Использование этого кода возможно, если инструменту был назначен режим рисования (например, использовался код DM_CUSTOM_LINE).  |
| CMD_INFO_Y2         | Y-координата точки, в которой пользователь отпустил кнопку мыши.   |
| CMD_INFO_SHIFT      | Логическая величина: TRUE (логическое "Да"), если пользователь при указании нажал на клавишу SHIFT.  |
| CMD_INFO_CTRL       | Логическая величина: TRUE (логическое "Да"), если пользователь при указании нажал на клавишу CTRL.   |
| CMD_INFO_TOOLBTN    | Целое число (Integer), идентификатор кнопки инструмента, который вызвал этот обработчик.   |
| CMD_INFO_CUSTOM_OBJ | Объектная величина: полилиния или многоугольник, нарисованный пользователем. Применяется с режимами рисования DM_CUSTOM_POLYLINE или DM_CUSTOM_POLYGON.  |



### Поддержка Геолинк (Hotlink)

Приложения MapBasic, запущенные инструментом Геолинк (Hotlink), могут получать информацию об активном объекте через функцию **CommandInfo( )**. В таблице приведен список атрибутов которые могут быть запрошены:

| код <i>атрибута</i>    | Результат <b>CommandInfo( attribute )</b> :                                |
|------------------------|--|
| CMD_INFO_HL_WINDOW_ID  | ID-номер окна карты или списка.  |
| CMD_INFO_HL_TABLE_NAME | Имя таблицы, ассоциированной со слоем карты или окном списка.              |
| CMD_INFO_HL_ROWID      | ID-номер строки таблицы соответствующей объекту карты или записи в списке. |
| CMD_INFO_HL_LAYER_ID   | ID-номер слоя, если программа была запущена из окна карты.                 |
| CMD_INFO_HL_FILE_NAME  | Имя запущенного файла.   |

**См. также:**

**Функция FrontWindow( ), Функция SelectionInfo( ), Оператор Set Command Info, Функция WindowInfo( )**

## Оператор Commit Table

### Назначение

Сохраняет последнюю редакцию таблицы на диске или сохраняет ее копию.

### Синтаксис

```
Commit Table table
  [ As параметры_файла
    [ Type { NATIVE |
      DBF [ Charset набор_символов ] |
      Параметры базы данных Access параметры_файла_базы_данных
      Version номер_версии
      Table имя_таблицы
      [ Password пароль ] [ Charset набор_символов ] |
      QUERY |
      ODBC Connection номер_соединения Table имя_таблицы
      [ ConvertDateTime {ON | OFF | INTERACTIVE}} ] } ]
    [ CoordSys... ]
    [ Version номер_версии ] ]
  [ { Interactive | Automatic commit_keyword } ]
  [ ConvertObjects {ON | OFF | INTERACTIVE}} ]
```

*tableName* – строковая переменная, определяющая имя таблицы, которая появится в Access; Начиная с версии 9.0, имя таблицы может включать в себя имя схемы, которой принадлежит таблица. Если имени схемы не задано, то таблица принадлежит стандартной схеме, подобно тому как было в версии 8.5 и более ранних. Пользователь должен задать правильное имя схемы, должен знать имя учетной записи и обладать правами доступа к указанной схеме. Это касается только SQL Server 2005.

*filespec* – спецификация файла (включая DOS-маршрут); Это то место, где сохраняется TAB-файл;

*ConvertDateTime* – если в исходной таблице существуют колонки любого из типов Time или Date, то, в зависимости от того, какой тип поддерживает сервер, они будут преобразованы в DATETIME или TIMESTAMP. Однако, этим преобразованием можно управлять, используя специальный параметр *ConvertDateTime*. Если в исходной таблице нет колонок любого из этих типов, то этот параметр не используется. Если параметру *ConvertDateTime* присвоено значение ON (режим по умолчанию), то колонки типов Time или Date будут преобразованы в DATETIME или TIMESTAMP. Если параметру *ConvertDateTime* присвоено значение OFF, то преобразований не будет, а операция будет при необходимости завершена. Если параметру *ConvertDateTime* присвоено значение INTERACTIVE, то появится диалог, в котором пользователю будет предложено выбрать необходимое действие. Если пользователь выберет преобразование, то тип будет преобразован, а если пользователь выберет отмену, то операция будет прекращена.

Тип Time требует преобразования всех доступных серверов (Oracle, IBM Informix, MS SQL Server и Access), а тип Date потребует преобразований только для баз данных серверов MS SQL Server и Access.

**Внимание:** Для серверов баз данных MS SQL Server и Access, это может вызвать проблемы совместимости с предыдущими версиями данных. Ранее такие преобразования выполнялись без предупреждения. В этой версии мы советуем использовать тип данных DateTime вместо типа данных Date. Если по-прежнему будет использоваться тип данных Date, то преобразование выполнено не будет.

*version* – выражение, определяющее версию базы данных Microsoft Jet, используемую для новой базы данных. Допускаются значения 4.0 (для Access 2000) или 3.0 (для Access '95/'97). Если выражение пропущено, то используется версия 4.0. Если база данных, в которой создается таблица, уже существует, то определение версии базы данных игнорируется;

*ConvertObjects ON* – автоматически конвертирует любые неподдерживаемые объекты, которыми обнаруживает в поддерживаемых объектах.

*ConvertObjects OFF* – в этом режиме не конвертируются никакие неподдерживаемые объекты. Если такие объекты встретятся, то будет выдано сообщение об ошибке – о том, что таблица не может быть сохранена. (До введения этой функции это был единственный вариант.)

*ConvertObjects Interactive* – если в таблице встретится неподдерживаемый объект, то пользователя может выбрать, что надо сделать.

*char\_set* – имя набора символов; см. раздел, в котором описан [Предложение CharSet на стр. 90](#);

*database\_filespec* – строка, которая определяет имя и DOS-маршрут к доступной базе данных Access. Если такая база не существует, MapInfo создаст новый Access-файл MDB;

*pwd* – пароль базы данных, заданный при включении защиты базы данных;

**ODBC** – определяет копию таблицы Table, которая сохраняется в базе данных, определенной номером удаленного соединения *ConnectionNumber*;

*ConnectionNumber* – целое значение, номер соединения с базой данных;

**CoordSys** – система координат; см. подробнее раздел [Оператор CoordSys on page 175](#);

*version* – 100 (для таблиц, которые могут читаться ранними версиями MapInfo) или 300 (MapInfo 3.0 формат), для не-Access таблиц. Для таблиц Access, версия должна быть 410;

*commit\_keyword* – одно из следующих ключевых слов: **NoCollision**, **ApplyUpdates**, **DiscardUpdates**

## Примеры использования ConvertDateTime

### Пример 1

```
Commit Table DATETIME90 As "D:\MapInfo\Data\Remote\DATETIME90CPY.TAB"
Type ODBC Connection 1 Table ""EASYLOADER"". "DATETIME90CPY"
ConvertDateTime Interactive
```

## Пример 2

```
Server 1 Create Table ""EazyLoader"". "CITY_125AA" (Field1
Char(10),Field2 Char(10),Field3 Char(10),MI_STYLE Char(254)) KeyColumn
SW_MEMBER ObjectColumn SW_GEOMETRY
Or (оператор Или)
Server 1 Create Table "EazyLoader.CITY_125AA" (Field1 Char(10),Field2
Char(10),Field3 Char(10),MI_STYLE Char(254)) KeyColumn SW_MEMBER
ObjectColumn SW_GEOMETRY
```

```
Commit Table City_125aa As
"C:\Projects\Data\TestScripts\English\remote\City_125aacpy.tab" Type ODBC
Connection 1 Table ""EazyLoader"". "CITY_125AACPY"
Or (оператор Или)
Commit Table City_125aa As
"C:\Projects\Data\TestScripts\English\remote\City_125aacpy.tab" Type ODBC
Connection 1 Table "EazyLoader.CITY_125AACPY"
```

## Описание

Если предложение **As** не определено, оператор **Commit Table** сохраняет любые изменения в таблице аналогично команде **Файл > Сохранить**.

Оператор **Commit Table**, который включает предложение **As**, имеет тот же самый эффект, как и команда **Файл > Создать копию**. Предложение **As** может быть использовано для сохранения таблицы под другим именем, в другом месте, или в виде другого типа файла, проекции.

Для сохранения таблицы под новым именем, укажите новое имя в строковой переменной *filespec*. Для сохранения таблицы в другом месте, укажите путь в начале строковой переменной *filespec* string.

Для сохранения таблицы как файла нового типа, включите предложение **Type** внутри предложения **As**. По умолчанию, тип новой таблицы **NATIVE**, но она также может быть сохранена как **DBF**.

Предложение **CharSet** определяет набор символов. Параметр *char\_set* должен быть строковой константой, такой как "MacRoman" или "WindowsLatin1". Если предложение **CharSet** не определено, MapBasic использует по умолчанию текущий набор символов Windows. Более подробную информацию см. в разделе [Предложение CharSet на стр. 90](#).

Для сохранения таблицы с использованием других систем координат или проекций, включите предложение **CoordSys** в предложение **As**. Обратите внимание, что только геокодируемые таблицы могут иметь систему координат и проекцию.

Для сохранения запроса используйте тип таблицы **QUERY**. Может быть сохранен только запрос сделанный пользователем через интерфейс и запрос, созданный оператором **Оператор Run Command** в MapBasic. Оператор **Commit Table** создает файлы **TAB** и **QRY**.

Предложение **Version** контролирует формат таблицы. Если Вы укажите **Version 100**, MapInfo сохранит таблицу в формате, читаемом ранними версиями MapInfo. Если Вы укажите **Version 300**, MapInfo сохранит таблицу в формате, используемом MapInfo 3.0. Обратите внимание, что

объекты типа полилиния и регион, имеющие более 8,000 узлов и полилинии, состоящие из множества сегментов требуют версию 300. Если Вы пропустите предложение **Version**, то таблица сохранится в формате версии 300.

**Внимание:** Если приложение MapBasic использует оператор **Commit Table...As**, действующий на таблицу, у которой есть мемо-поля, то эти мемо-поля не сохраняются в новой таблице. Предупреждения об этом на экране не будет. Если таблица сохраняется в виде новой таблицы с помощью команды MapInfo (**Файл > Создать копию**), то MapInfo предупредит пользователя о потере мемо-полей. Однако, при сохранении новой таблицы из программы MapBasic, предупреждения не будет.

### Сохранение связанных таблиц

Сохранение связанной таблицы может породить конфликт, поскольку другие пользователи могут редактировать записи в удаленной базе данных. Следующие предложения позволяют Вам контролировать то, что может произойти при конфликте. (Эти предложения не действуют при сохранении обычной таблицы MapInfo.)

#### *Interactive*

В случае конфликта MapInfo показывает диалог “Разрешение конфликтов”. После успешного выполнения оператора **Commit Table Interactive**, MapInfo показывает диалог обновления.

#### *Automatic NoCollision*

В случае конфликта MapInfo не выполняет сохранение. (Этот режим используется по умолчанию, то есть в случае, если не используются предложения Interactive или Automatic.)

#### *Automatic ApplyUpdates*

В случае конфликта MapInfo сохраняет значения локальной копии связанной таблицы. (Это аналогично полному игнорированию конфликта.)

#### *Automatic DiscardUpdates*

В случае конфликта MapInfo сохраняет значения из РСУБД (локальные изменения отменяются). Вы можете сохранить копию связанной таблицы, используя предложение **As**; но полученная в результате таблица не будет связанной и не может быть обновлена с сервера.

### Предложение ODBC Connection

Длина имени таблицы *tablename* варьирует в зависимости от базы данных. Мы рекомендуем применять 14 или менее символов для имени таблицы, в этом случае гарантирована корректная работа с любой базой данных. В операторе установлена максимально возможная длина имени файла *tablename*, равная 31 символу.

Если используется предложение **As** и тип соединения это **ODBC**, то копия таблицы будет сохранена в базе данных, указанной номером соединения *ConnectionNumber* под именем *tablename*. Если исходная таблица имеет географические объекты, то к таблице *tablename*, создаваемой в удаленной базе, могут быть добавлены три колонки Key, Object и Style, независимо от того, есть такие колонки в исходной таблице или нет. Если исходная таблица

без географических объектов, то только одна колонка, Key, может быть добавлена к таблице базы данных, *tablename*, независимо от того, была ли такая колонка в исходных данных. Колонка Key будет использоваться для создания уникального индекса.

Пространственный индекс будет создаваться в колонке Object, если она есть. Поддерживаемые базы данных это Oracle, SQL Server, IIS (Informix Universal Server) и Microsoft Access. Однако, для сохранения таблицы с пространственной геометрией/объектами (включая сохранение таблицы только с точечными объектами), для SQL Server и IUS потребуется SpatialWare/Blade, в дополнение к пространственным настройкам для Oracle. Схема XY не поддерживается в этом операторе.

### Пример:

Следующий пример открывает таблицу STATES, затем использует оператор **Commit Table**, чтобы сделать копию этой таблицы под новым именем (ALBERS). Необязательное предложение **Оператор CoordSys** указывает на то, что таблица ALBERS сохранится с равноплощадной проекцией Альберта.

```
Open Table "STATES"
Commit Table STATES
  As "ALBERS"
  CoordSys Earth
  Projection 9,7, "m", -96.0, 23.0, 20.0, 60.0, 0.0, 0.0
```

Следующий пример иллюстрирует соединение с удаленной базой данных:

```
dim hodb as integer
hodb = server_connect("ODBC", "dlg=1")
Open table "C:\MapInfo\USA"
Commit Table USA
as "c:\temp\as\USA"
  Type ODBC Connection hodb Table "USA"
```

### См. также:

**Оператор Rollback**

---

## ConnectObjects( ) функция

### Назначение

Возвращает объект, представляющий кратчайшее или самое большое расстояние между двумя объектами.

### Синтаксис

```
ConnectObjects( object1, object2, min )
```

*object1* и *object2* - выражения, указывающие на объекты.

*min* - логическое выражение. Если это выражение принимает значение "Да" (TRUE), функция вычисляет минимальное расстояние между объектами, а если выражение принимает значение "Нет" (FALSE), вычисляется максимальное расстояние.

### Возвращаемая величина

Оператор возвращает объект полилинии, состоящий из одного сегмента и двух точек, и представляющий собой кратчайшее расстояние ((`min == TRUE`), или самое большое расстояние (`min == FALSE`) между объектами *object1* и *object2*.

### Описание

Одна точка полученной полилинии принадлежит объекту *object1*, а другая объекту *object2*. Обратите внимание, что расстояние между двумя объектами можно вычислить при помощи **Функция ObjectLen( )**. Если существует множество вариантов кратчайших или наибольших расстояний (например, полученный сегмент представляет одно расстояние, а существуют еще и другие сегменты равной длины), эти функции возвращают лишь один из вариантов. Не существует способа определить, является ли полученный объект полилинии единственным кратчайшим расстоянием.

**ConnectObjects( )** возвращает объект типа полилиния, соединяющий *object1* и *object2* по кратчайшему (`min == TRUE`) или наидлиннейшему (`min == FALSE`) пути, используя сферический метод вычислений. Если применить сферический методы вычислений не удастся, (например, если координатная система MapBasic - план), то используются декартовы вычисления.

---

## Оператор Continue

### Назначение

Возобновляет выполнение программы MapBasic (располагается за оператором **Параметр Stop**).

### Синтаксис

`Continue`

### Предупреждение

Оператор **Continue** используется только в окне MapBasic и не может быть частью программы.

### Описание

Оператор **Continue** используется для возобновления выполнения приложения MapBasic, остановленного оператором **Параметр Stop**.

Оператор **Параметр Stop** используется для отладочных целей. Когда программа выполняет оператор **Параметр Stop**, она приостанавливается, и в списке меню **Файл** в окне MapInfo команда **Запустить программу MapBasic** меняется на **Продолжить программу MapBasic**. Продолжить выполнение программы можно, выбрав команду **Файл > Продолжить программу MapBasic**. Можно также ввести в окно MapBasic оператор **Continue** – это даст тот же эффект, что и команда **Продолжить программу MapBasic**.





## Предложения Control Button / OKButton / CancelButton

### Назначение

Часть оператора **Оператор Dialog**; отвечают за создание кнопки с текстом.

### Синтаксис

```
Control { Button | OKButton | CancelButton }
  [ Position x, y ] [ Width w ] [ Height h ]
  [ ID control_ID ]
  [ Calling handler ]
  [ Title title_string ]
  [ Disable ] [ Hide ]
```

*x, y* – координаты левого верхнего угла кнопки в окне диалога в диалоговых единицах;

*w* – ширина кнопки в диалоговых единицах, по умолчанию 40;

*h* – высота кнопки в диалоговых единицах, по умолчанию 18;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*handler* – имя процедуры-обработчика, которая запускается при нажатии на кнопку;

*title\_string* – текст на кнопке.

### Описание

Если **Оператор Dialog** включает предложение **Control Button**, то в диалоге создаются кнопки с текстом, нажатие на которые приводит к выполнению определенных действий. Если задано **OKButton** в качестве **Button** то создается кнопка подтверждения; нажатие на **OKButton** приводит к закрытию диалогового окна с сохранением всех установленных значений в диалоге. Кнопка типа **CancelButton** создает кнопку, позволяющую закрывать диалог без сохранения значений. Каждый диалог должен содержать не более одной кнопки подтверждения **OKButton** и не более одной кнопки отмены **CancelButton**. **Disable** делает флажок недоступным для выбора (закрашивается серым). **Hide** делает флажок в диалоговом окне изначально скрытым.

Для изменения состояния элемента диалога используйте оператор **Оператор Alter Control** (например, для показа скрытой кнопки).

### Пример:

```
Control Button
  Title "&Reset"
  Calling reset_sub
  Position 10, 190
```

### См. также:

**Оператор Alter Control, Оператор Dialog**

## Предложение Control CheckBox

### Назначение

Часть оператора **Оператор Dialog**; Отвечает за создание флажка.

### Синтаксис

#### Control CheckBox

```
[ Position x, y ] [ Width w ]  
[ ID control_ID ]  
[ Calling handler ]  
[ Title title_string ]  
[ Value log_value ]  
[ Into log_variable ]  
[ Disable ] [ Hide ]
```

*x, y* – координаты левого верхнего угла элемента в окне диалога в специальных единицах измерения диалога;

*w* – ширина в диалоговых единицах;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*handler* – имя процедуры-обработчика, которая запускается при изменении режима флажка;

*title\_string* – текст справа от флажка;

*log\_value* – логическая величина, задающая начальное значение: FALSE – флажок сброшен;

*log\_variable* – имя логической переменной, которой будет присвоено значение элемента после закрытия диалога.

### Описание

Если **Оператор Dialog** включает предложение **Control CheckBox**, в диалоге создается флажок – элемент, который может иметь только два значения.

Предложение **Value** позволяет присваивать элементу диалога начальное значение. Если предложение **Value** опущено или, наоборот, присутствует и задает начальное значение TRUE, то флажок при открытии диалогового окна будет установлен. Если предложение **Value** задает значение FALSE, то флажок будет сброшен. **Disable** делает флажок недоступным для выбора (закрашивается серым). **Hide** делает флажок в диалоговом окне изначально скрытым.

### Пример:

```
Control CheckBox  
  Title "Include &Legend"  
  Into showlegend  
  ID 6  
  Position 115, 155
```

См. также:

Оператор **Alter Control**, Оператор **Dialog**, Функция **ReadControlValue( )**

## Предложение **Control DocumentWindow**

### Назначение

Часть оператора **Оператор Dialog**; добавляет управление окном документа к диалогу, который может быть порожденным при использовании интегрированной картографии.

### Синтаксис

```
Control DocumentWindow
  [ Position x, y ]
  [ Width w ] [ Height h ]
  [ ID control_ID ]
  [ Disable ] [ Hide ]
```

*x, y* – определяют позицию элемента управления в единицах диалога;

*w* – определяет ширину элемента управления в единицах диалога; стандартная ширина 100;

*h* – определяет высоту элемента управления в единицах диалога; стандартная высота 100;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

**Disable** – делает элемент управления изначально неактивным;

**Hide** – изначально скрывает элемент управления.

### Описание

Если оператор **Оператор Dialog** включает в себя предложение **Control DocumentWindow**, то диалог включает в себя элемент управления окна документа, который может быть порожден при использовании **Оператор Set Next Document**.

### Пример:

Следующий пример создает легенду в диалоге:

```
Control DocumentWindow
  ID ID_LEGENDWINDOW
  Position 160, 20
  Width 120 Height 150
```

Обработчик диалога должен породить окно как показано в следующем примере:

```
Sub DialogHandler
  OnError Goto HandleError
  Dim iHwnd As Integer
  Alter Control ID_LEGENDWINDOW Enable Show
  ' draw the legend
  iHwnd = ReadControlValue(ID_LEGENDWINDOW)
  Set Next Document Parent iHwnd Style WIN_STYLE_CHILD
```

```
Create Legend
Exit Sub
HandleError:
    Note "DialogHandler: " + Error$( )
End Sub
```

См. также:

[Оператор Dialog](#)

---

## Предложение Control EditText

### Назначение

Входит в [Оператор Dialog](#); отвечает за создание текстового окошка ввода.

### Синтаксис

#### Control EditText

```
[ Position x, y ] [ Width w ] [ Height h ]
[ ID control_ID ]
[ Value initial_value ]
[ Into variable ]
[ Disable ] [ Hide ] [ Password ]
```

*x*, *y* – координаты левого верхнего угла элемента в окне диалога в специальных единицах измерения диалога;

*w* – ширина в диалоговых единицах;

*h* – высота в единицах измерения диалога; если высота превышает 20 единиц, то текст разбивается на несколько строк;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*initial\_value* – строковая величина, задающая начальное значение;

*variable* – имя строковой переменной, которой будет присвоен текст из окошка после закрытия диалога кнопкой **OK**.

Ключевое слово **Disable** делает элемент недоступным (закрашивается серым).

Ключевое слово **Hide** прячет элемент из диалогового окна.

Ключевое слово **Password** включает "слепой" режим ввода текста (показывается звездочка вместо каждого введенного пользователем символа), используется для создания окошка ввода пароля.

### Описание

Если пользователь вводит текст длиной больше, чем может поместиться в окошко, MapInfo Professional автоматически прокручивает экран и создает место для продолжения текста. Элемент EditText может иметь строковое значение длиной до 32767 символов.

Если высота больше 20 единиц, то окошко будет иметь две и более строк. В этом случае текст, который пользователь введет, будет автоматически разбиваться на строки. Если пользователь желает ввести символ перевода строки (line-feed) внутри окошка EditText (в Windows для этого надо нажать на клавиши **CTRL+ENTER**), то в строку текста для элемента EditText нужно добавить `Chr$(10)`. Если в выражение *initial\_value* уже вложены вызовы `Chr$(10)`, то в диалоге показывается уже разбитый на строки текст.

Для перемещения фокуса в элемент EditText используйте оператор **Alter Control...Active**.

#### Пример:

```
Control EditText Value "Торговые точки" Position 65, 8 Width 90 ID 1 Into
s_map_title
```

#### См. также:

**Оператор Alter Control, Оператор Dialog, Функция ReadControlValue( )**

---

## Предложение Control GroupBox

### Назначение

Входит в **Оператор Dialog**; отвечает за создание прямоугольной рамки с текстом.

### Синтаксис

```
Control GroupBox
  [ Position x, y ] [ Width w ] [ Height h ]
  [ ID control_ID ]
  [ Title title_string ]
  [ Hide ]
```

*x, y* – координаты левого верхнего угла элемента в окне диалога в специальных единицах измерения диалога;

*w* – ширина в диалоговых единицах;

*h* – высота рамки в единицах измерения диалога;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*title\_string* – текст заголовка, который начинается от левого верхнего угла рамки.

Ключевое слово **Hide** прячет элемент из диалогового окна.

#### Пример:

```
Control GroupBox Title "Уровень детализации" Position 5, 30Height 40
Width 70
```

#### См. также:

**Оператор Alter Control, Оператор Dialog**

## Предложения Control ListBox / MultiListBox

### Назначение

Входит в **Оператор Dialog**; отвечают за создание списков.

### Синтаксис

```
Control { ListBox | MultiListBox }  
  [ Position x, y ] [ Width w ] [ Height h ]  
  [ ID control_ID ]  
  [ Calling handler ]  
  [ Title { str_expr | From Variable str_array_var } ]  
  [ Value i_selected ]  
  [ Into i_variable ]  
  [ Disable ] [ Hide ]
```

*x*, *y* – координаты левого верхнего угла элемента в окне диалога в специальных единицах измерения диалога;

*w* – ширина окошка в единицах измерения диалога, по умолчанию – 80;

*h* – высота окошка в единицах измерения диалога, по умолчанию – 70;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*handler* – имя процедуры-обработчика, которая запускается при изменении пользователем выбора в списке или двойным указанием в списке;

*str\_expr* – строковое выражение, которое задает тексты надписей при переключателях; элементы списка разделены точкой с запятой (;);

*str\_array\_var* – имя массива строковых переменных;

*i\_selected* – короткое целое число, задающее номер элемента списка, который будет выбран при открытии диалога, по умолчанию в списке не будет выбрано ни одной строки;

*i\_variable* – имя переменной типа SmallInt, которая будет использоваться для сохранения значения выбора в списке после закрытия диалога.

Ключевое слово **Disable** делает элемент недоступным (закрашивается серым).

Ключевое слово **Hide** прячет элемент из диалогового окна.

### Описание

Если **Оператор Dialog** включает предложение **Control ListBox**, то в диалог добавляется окошко списка. Если в окошко не вмещаются все строки списка, то окошко снабжается справа полосой прокрутки.

Элемент MultiListBox идентичен элементу ListBox, за исключением того, что в первом пользователь может выбрать несколько элементов при нажатой клавише SHIFT.

Предложение **Title** задает строки списка. Если за словом **Title** задан список текстов в кавычках, через точку с запятой, то каждый текст будет определять одну строчку списка. В следующем фрагменте за предложением **Title** следует строковый массив:

```
Title "1-й квартал;2-й квартал;3-й квартал;4-й квартал"
```

Также список в предложении **Title** можно задавать массивом. В следующем фрагменте за предложением **Title** следует строковый массив:

```
Title From Variable s_optionlist
```

### Чтение значений элемента MultiListBox

Так как в окошке списка этого элемента может быть не выбрано ни одной строки, выбрана одна строка, несколько строк или весь список, то для элемента MultiListBox нужно спроектировать специальную процедуру чтения значений (присваиваемую, например, элементу OKButton). Изнутри процедуры-обработчика для чтения всех значений несколько раз вызывается **Функция ReadControlValue( )**.

При первом вызове **Функция ReadControlValue( )** дает номер первой строки, выбранной в списке. При следующем вызове **Функция ReadControlValue( )** дает номер второй строки и т. д. Когда все значения будут прочитаны, **Функция ReadControlValue( )** вернет ноль. Если при первом вызове **Функция ReadControlValue( )** возвратит ноль, то в списке ничего не было выбрано.

### Двойной щелчок мышки в списке

Для списка может быть объявлена процедура-обработчик *handler*, которая вызывается всякий раз, когда пользователь указывает на список. Причем это может быть простое указание, а может быть двойное, то есть, если нажать два раза на клавишу мышки. Обычно, двойное указание в списке аналогично простому указанию и последующему нажатию на кнопку **ОК**, которая закрывает диалог с подтверждением всех изменений значений элементов в диалоге.

Пример можно найти в программе NIEWS.MB в <каталог установки  
MapBasic>\SAMPLES\MAPBASIC\SNIPPETS.

Для определения, применялось ли пользователем двойное указание или нет, в процедуре обработчика спискового элемента диалога используется **CommandInfo( )** функция. Вот пример такой процедуры обработчика:

```
Sub lb_handler
  Dim i As SmallInt
  If CommandInfo(CMD_INFO_DLG_DBL) Then
    ' ... then the user double-clicked.
    i = ReadControlValue( TriggerControl( ) )
    Dialog Remove
    ' at this point, the variable i represents
    ' the selected list item...
  End If
End Sub
```

### Пример:

```
Control ListBox Title "1-ый квартал;2-ой квартал;3-ий квартал;4-ый  
квартал" ID 3Value 1Into i_quarterPosition 10, 92 Height 40
```

В тексте программы NIEWS.MB можно видеть, как создается список и как можно применять двойное указание мышкой. Программа NIEWS показывает диалог с элементом ListBox. Этот диалог закрывается, если пользователь либо выбрал элемент списка и нажал **ОК**, либо если указал на элемент списка дважды.

В этом примере предложение **Control ListBox** добавляет список в диалог "Named Views" ("Именованные виды"). Заметим, что элементу ListBox сопоставлен обработчик "listbox\_handler."

```
Control ListBox  
    Title desc_list  
    ID 1  
    Position 10, 20 Width 245 Height 64  
    Calling listBox_handler
```

Если пользователь выбирает элемент списка, MapBasic вызывает процедуру-обработчик "listbox\_handler". Из процедуры вызывается **CommandInfo( ) функция** для определения, какой щелчок мышкой был сделан - одиночный или двойной. Если был сделан двойной щелчок, процедура закрывает диалог, используя **Оператор Dialog Remove**. Если не сработал **Оператор Dialog Remove**, диалог останется на экране, пока пользователь не выберет **ОК** или **Cancel**.

```
Sub listBox_handler  
    Dim i As SmallInt  
    ' Сначала, после того, как пользователь выбрал имя из списка,  
    ' мы можем активировать кнопки OK и Delete ("Удалить").  
    Alter Control 2 Enable  
    Alter Control 3 Enable  
    If CommandInfo(CMD_INFO_DLG_DBL) = TRUE Then  
        ' ...пользователь щелкнул мышкой дважды  
        ' определяем, где это произошло  
        i = ReadControlValue(1) ' найден элемент, выбранный пользователем  
        Dialog Remove Call go_to_view(i) ' дальнейшие действия для этого  
        случая  
    End If  
End Sub
```

MapBasic вызывает процедуру-обработчик и в случае одиночного, и в случае двойного щелчка мышкой. Процедура-обработчик должна различать одиночные и двойные щелчки.

### См. также:

**Оператор Alter Control, Оператор Dialog, Функция ReadControlValue( ), CommandInfo( ) функция**



## Предложения Control PenPicker/BrushPicker/SymbolPicker/FontPicker

### Назначение

Входит в **Оператор Dialog**; предложения отвечают за создание кнопок выбора стиля: линии, штриха, символа или шрифта.

### Синтаксис

```
Control { PenPicker | BrushPicker | SymbolPicker | FontPicker }
  [ Position x, y ] [ Width w ] [ Height h ]
  [ ID control_ID ]
  [ Calling handler ]
  [ Value style_expr ]
  [ Into style_var ]
  [ Disable ] [ Hide ]
```

*x, y* – координаты левого верхнего угла текста в окне диалога в специальных единицах измерения диалога;

*w* – ширина кнопки в единицах измерения диалога, по умолчанию 20;

*h* – высота кнопки в единицах измерения диалога, по умолчанию 20;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*handler* – имя процедуры-обработчика; если пользователь нажмет на кнопку определения стиля, а в появившемся диалоге нажмет **OK**, то MapBasic вызовет процедуру *handler*.

*style\_expr* – выражение типа Pen, Brush, Symbol или Font, которое задает начальное значение элемента;

*style\_var* – имя переменной типа Pen, Brush, Symbol или Font (тип переменной должен соответствовать типу элемента диалога).

Ключевое слово **Disable** делает элемент недоступным (закрашивается серым).

Ключевое слово **Hide** прячет элемент из диалогового окна.

### Описание

Ключевые слова (PenPicker, BrushPicker, SymbolPicker или FontPicker) позволяют создавать в диалоге кнопку, нажатие на которую открывает стандартный диалог выбора стиля оформления объектов.

### Пример:

```
Control SymbolPicker
  Position 140,42
  Into sym_storemarker
```

### См. также:

**Оператор Alter Control, Оператор Dialog, Функция ReadControlValue( )**

## Предложение Control PopupMenu

### Назначение

Входит в **Оператор Dialog**; отвечает за создание раскрывающегося меню.

### Синтаксис

#### Control PopupMenu

```
[ Position x, y ]  
[ Width w ]  
[ ID control_ID ]  
[ Calling handler ]  
[ Title { str_expr | From Variable str_array_var } ]  
[ Value i_selected ]  
[ Into i_variable ]  
[ Disable ]
```

*x*, *y* – координаты левого верхнего угла элемента в окне диалога в специальных единицах измерения диалога;

*w* – ширина элемента в единицах измерения диалога, по умолчанию 20;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*handler* – имя процедуры, вызываемой, если пользователь выбрал пункт меню.

*str\_expr* – строковое выражение, которое задает тексты надписей при переключателях; элементы списка разделены точкой с запятой (;);

*str\_array\_var* – имя массива строковых переменных;

*i\_selected* – короткое целое число, задающее номер переключателя, который будет выбран при открытии диалога, при задании единицы (1) или по умолчанию будет выбран первый переключатель;

*i\_variable* – имя переменной типа SmallInt, которая будет использоваться для сохранения номера выбранного элемента после закрытия диалога.

Ключевое слово **Disable** делает элемент недоступным (закрашивается серым).

### Описание

Если **Оператор Dialog** включает предложение **Control PopupMenu**, в диалоговом окне появится элемент в виде раскрывающегося меню. Такое меню можно открыть и выбрать из списка одно значение (пункт). При открытии диалогового окна показывается только тот пункт меню, который активен.

Если пользователь указывает на него мышкой, меню открывается полностью и можно выбрать другой пункт.

Предложение **Title** задает список элементов меню. Если за словом **Title** задан список текстов в кавычках, через точку с запятой, то каждый текст будет определять одну строчку списка. В следующем фрагменте за предложением **Title** следует строковый массив:

```
Title "Город;Область;Территория;Регион;Вся страна"
```

Список в предложении **Title** можно также задавать массивом.

В следующем фрагменте за предложением **Title** следует строковый массив:

```
Title From Variable s_optionlist
```

### Пример:

```
Control PopupMenu
  Title "Город;Область;Территория;Регион;Вся страна"
  Value 2
  ID 5
  Into i_map_scope
  Position 10, 150
```

**См. также:**

**Оператор Alter Control, Оператор Dialog, Функция ReadControlValue( )**

## Предложение Control RadioGroup

### Назначение

Входит в **Оператор Dialog**; отвечает за создание кнопок переключателя.

### Синтаксис

```
Control RadioGroup
  [ Position x, y ]
  [ ID control_ID ]
  [ Calling handler ]
  [ Title { str_expr | From Variable str_array_var } ]
  [ Value i_selected ]
  [ Into i_variable ]
  [ Disable ] [ Hide ]
```

*x, y* – координаты левого верхнего угла элемента в окне диалога в специальных единицах измерения диалога;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*handler* – имя процедуры-обработчика, которая запускается при выборе пользователем другого переключателя списке или двойным указанием на другой переключатель;

*str\_expr* – строковое выражение, которое задает тексты надписей при переключателях; элементы списка разделены точкой с запятой (;);

*str\_array\_var* – имя массива строковых переменных;

*i\_selected* – короткое целое число, задающее номер переключателя, который будет выбран при открытии диалога, при задании единицы (1) или по умолчанию будет выбран первый переключатель;

*i\_variable* – имя переменной типа SmallInt, которая будет использоваться для сохранения номера выбранного элемента после закрытия диалога.

Ключевое слово **Disable** делает элемент недоступным (закрашивается серым).

Ключевое слово **Hide** прячет элемент из диалогового окна.

### Описание

Если **Оператор Dialog** включает предложение **Control RadioGroup**, в диалоге появляется группа переключателей. Каждый переключатель состоит из надписи и кружочка. Выбранный переключатель сопровождается заполненным кружочком. Выбранным может быть только один переключатель.

Предложение **Title** задает надписи при переключателях. Если за словом **Title** задан список текстов в кавычках, через точку с запятой, то каждый текст будет определять одну строку списка.

В следующем фрагменте за предложением **Title** следует строковый массив:

```
Title "&Всё;В&ыборочно"
```

Список в предложении **Title** можно также задавать массивом. В следующем фрагменте за предложением **Title** следует строковый массив:

```
Title From Variable s_optionlist
```

### Пример:

```
Control RadioGroup
  Title "&Full Details;&Partial Details"
  Value 2
  ID 2
  Into i_details
  Calling rg_handler
  Position 15, 42
```

**См. также:**

**Оператор Alter Control**, **Оператор Dialog**, **Функция ReadControlValue( )**

---

## Предложение Control StaticText

### Назначение

Входит в **Оператор Dialog**; отвечает за создание текстового элемента в окне диалога, (неизменяемого текста).

### Синтаксис

```
Control StaticText
  [ Position x, y ]
  [ Width w ] [ Height h ]
  [ ID control_ID ]
```

```
[ Title title_string ]
[ Hide ]
```

$x$ ,  $y$  – координаты левого верхнего угла текста в окне диалога в специальных единицах измерения диалога;

$w$  – ширина подписи в единицах измерения диалога;

$h$  – высота подписи в единицах измерения диалога;

*control\_ID* – целое; не может совпадать с идентификаторами других элементов управления в диалоге;

*title\_string* – текст.

Ключевое слово **Hide** прячет элемент из диалогового окна.

### Описание

Если Вы хотите, чтобы текст в диалоге занимал несколько строк, то задайте для него прямоугольник, используя предложения **Width** и **Height**. Если предложения **Width** и **Height** не использовать, то текст будет состоять только из одной строки.

### Пример:

```
Control StaticText
    Title "Заголовок карты:"
    Position 5, 10
```

См. также:

[Оператор Alter Control](#), [Оператор Dialog](#)

---

## Функция ConvertToPline( )

### Назначение

Возвращает полилинию, которой аппроксимирован заданный объект.

### Синтаксис

```
ConvertToPline( object )
```

*object* – объект для преобразования любого типа, кроме текстового и точечного.

### Возвращаемая величина

Полилиния. Величина типа Object.

### Описание

Функция **ConvertToPline( )** возвращает объект типа "полилиния", которая описывает объект *object*. Так, если *object* задает область, то функция **ConvertToPline( )** вернет ломаную линию, представляющую границу области, и с тем же количеством узлов.

## Функция **ConvertToRegion( )**

---

Результат функции **ConvertToPline( )** будет таким же, как при выполнении команды MapInfo Professional **Объекты > Превратить в полилинии**. Отличие в том, что функция **ConvertToPline( )** что создает новый объект, не меняя исходный.

**См. также:**

[Оператор Objects Enclose](#)

---

## Функция **ConvertToRegion( )**

### Назначение

Возвращает область по форме заданного объекта.

### Синтаксис

**ConvertToRegion( *object* )**

*object* – объект для преобразования любого типа, кроме текстового и точечного.

### Возвращаемая величина

Область. Величина типа Object.

### Описание

При этой операции сохраняются все значения стилей оформления объекта. Недостающие атрибуты используются по текущим значениям стилей. Если первый узел совпадает с последним, то при преобразовании полилинии они сливаются. Если первый и последний узлы полилинии не совпадают, то при преобразовании в область добавляется последний узел, совпадающий с первым.

Функция **ConvertToRegion( )** возвращает область, имеющую такую же форму, какую имеет исходный объект. Так, если исходный объект – прямоугольник, то **ConvertToRegion( )** вернет область прямоугольной формы.

Результат функции **ConvertToRegion( )** такой же, как при выполнении команды MapInfo Professional **Объекты > Превратить в области**. Отличие в том, что **ConvertToRegion( )** создает новый объект, не меняя исходный.

**См. также:**

[Оператор Objects Enclose](#)

## Функция ConvexHull( )

### Назначение

Возвращает объект-регион, который представляет собой контур – полигон, созданный на основании узлов входного объекта. Такой полигон можно получить, если натянуть резинку, охватывающую все узлы. Он составлен из минимального количества точек (т.е. точки входного объекта лежат на границах или внутри полигона). Полигон получается выпуклый – то есть все внешние углы созданного полигона-контур больше, чем 180 градусов.

### Синтаксис

**ConvexHull**( *inputobject* )

*inputobject* – объектное выражение (выражение, результат которого есть величина типа Object);

### Возвращаемая величина

Область. Величина типа Object.

### Описание

Функция **ConvexHull( )** возвращает регион, представляющий контур, охватывающий точки входного объекта. Функция **ConvexHull( )** создает один контур за один раз. Чтобы создать контуры вокруг нескольких объектов, используйте оператор Create Object As ConvexHull.

### Пример:

Следующий пример выбирает штат New York из таблицы штатов, затем создает полигон-контур вокруг выборки.

```
Dim Resulting_object as object
select * from States
where State_Name = "New York"
Resulting_object = ConvexHull(selection.obj)
Insert Into States(obj) Values (Resulting_object)
```

См. также:

[Оператор Create Object](#)

## Оператор CoordSys

### Назначение

Задаёт систему координат.

**Синтаксис 1****CoordSys Earth**

```
[ Projection type, datum, unitname  
  [ , origin_longitude ] [ , origin_latitude ]  
  [ , standard_parallel_1 [ , standard_parallel_2 ] ]  
  [ , azimuth ] [ , scale_factor ]  
  [ , false_easting ] [ , false_northing ]  
  [ , range ] ]  
[ Affine Units unitname, A, B, C, D, E, F ]  
[ Bounds ( minx, miny ) ( maxx, maxy ) ]
```

**Синтаксис 2****CoordSys Nonearth**

```
[ Affine Units unitname, A, B, C, D, E, F ]  
Units unitname  
[ Bounds ( minx, miny ) ( maxx, maxy ) ]
```

**Синтаксис 3**

**CoordSys Layout Units** *paperunitname*

**Синтаксис (вариант 4):**

**CoordSys Table** *tablename*

**Синтаксис 4**

**CoordSys Window** *window\_id*

*type* – положительное целое число, представляющее тип координатной системы;

*datum* – положительное целое число, определяющее референс-эллипсоид;

*unitname* – единица измерения расстояний, строковая величина (например, "m" – метры, список единиц приведен в разделе [Оператор Set Distance Units on page 633](#);

*origin\_longitude* – долгота точки отсчета (нулевой точки) в градусах, вещественное число (тип Float);

*origin\_latitude* – широта точки отсчета (нулевой точки) в градусах, вещественное число (тип Float);

*standard\_parallel\_1* и *standard\_parallel\_2* – стандартные широты в градусах, вещественное число (тип Float);

*azimuth* – азимутальный угол в градусах, вещественное число (тип Float);

*scale\_factor* – коэффициент искажения (масштабный коэффициент), вещественное число (тип Float);

*range* – вещественное число от 1 до 180 (тип Float), определяющее, какая часть Земли видна;

*minx* – минимальная X-координата, вещественное число (тип Float);

*miny* – минимальная Y-координата, вещественное число (тип Float);



*maxx* – максимальная X-координата, вещественное число (тип Float);

*maxy* – максимальная Y-координата, вещественное число (тип Float);

*unitname* – строковая величина, представляющая имя "бумажной" единицы, например, "in" – дюймы, список единиц приведен в разделе [Оператор Set Paper Units on page 671](#);

*tablename* – имя открытой таблицы;

*window\_id* – целочисленный идентификатор окна Карты или Отчета.

*A* задает масштабирование или растяжение вдоль оси X.

*B* задает поворот или сдвиг вдоль оси X.

*C* задает смещение вдоль оси X.

*D* задает масштабирование или растяжение вдоль оси Y.

*E* задает поворот или сдвиг вдоль оси Y.

*F* задает смещение вдоль оси Y.

## Описание

Предложение **CoordSys** задает координатную систему, а также может дополнительно задавать проекцию для этой системы. Помните, что предложение **CoordSys** не является отдельным оператором. Предложение **CoordSys** может входить в состав разных операторов; например, [Оператор Set Map](#) может включать **CoordSys**, и в этом случае [Оператор Set Map](#) переопределяет проекцию, используемую в соответствующем окне Карты.

Первый вариант синтаксиса предложения используется для Карт мира. Параметры предложения **Projection** задают проекцию Карты (если она есть), и должны использоваться в соответствии с данной координатной системой. Если предложение **Projection** опущено, MapBasic использует референц-эллипсоид (датум) под номером 0. Предложение **Affine** описывает аффинное преобразование, используемое для трансформации координатной системы. Если предложение **Projection** опущено, то базовой назначается координатная система Долгота/Широта. Так как порожденные координаты могут быть представлены в других единицах, чем базовые, в предложении **Affine** требуется указать единицы для порожденной системы.

Второй вариант синтаксиса используется для координатных систем планов, например, для поэтажного плана или другого CAD-изображения. В случае, когда предложение **CoordSys** описывает не мировую систему, базовой координатной системой можно назначить любую декартову сеть. Предложение **Units** задает единицы измерения базовой координатной системы, а предложение **Affine** задает единицы измерения производной координатной системы.

Третий вариант синтаксиса предложения (**CoordSys Layout**) используется для задания координатной системы в окне Отчета. Программа MapBasic должна выполнить оператор Set CoordSys Layout перед тем, как создавать объекты Отчета и работать с ними. Параметр *unitname* задает "бумажные" единицы измерения, например, "in" для дюймов или "cm" для сантиметров. Например, следующий [Оператор Set CoordSys](#) определяет дюймы, как единицы измерений в окне Отчета:

```
Set CoordSys Layout Units "in"
```

Четвертый вариант синтаксиса (**CoordSys Table**) используется для задания координатной системы в таблице, данные которой сохраняются на диске.

Пятый вариант синтаксиса (**CoordSys Window**) позволяет установить координатную систему, как в текущем окне.

Если предложение **CoordSys** использует **Оператор Set Map** или **Оператор Set Digitizer**, то подпредложение **Bounds** игнорируется. Подпредложение **Bounds** требуется для задания карт в немировой системе координат, если задано предложение **CoordSys**; но это справедливо только для немировых карт.

Версии MapInfo Professional меньше, чем 4.1.2 не распознают константы аффинных преобразований в предложении **CoordSys**, файле MAPINFO.RRJ и MAP-файлах. Если MAP-файл создан с использованием аффинного преобразования, то более поздние версии MapInfo Professional будут использовать базовую координатную систему, а не производную.

Предложение **Bounds** задает границы, в которых показывается Карта. Объект не может быть создан за пределами заданных этим предложением границ. Если задается координатная система Карты мира (Earth), то Вы должны опустить предложение **Bounds**, так как MapInfo пытается по умолчанию охватить всю Землю.

**Внимание:** Используя **Оператор Create Map**, можно увеличить точность координат на Карте, задав более узкий охват предложением **Bounds**.

Каждая картографическая проекция задается уравнением, имеющим свой индивидуальный набор параметров. Поэтому предложение **CoordSys** может иметь разный набор параметров в предложении **Projection**. Например, уравнение, задающее проекции Робинсона, использует параметры:

*datum* (референс-эллипсоид),  
*unitname* (единицы измерения) и  
*origin\_latitude* (начальную широту),  
а уравнение для модифицированной проекции Меркатора -  
*datum* (референс-эллипсоид),  
*unitname* (начальную широту),  
*origin\_longitude* (начальную широту),  
*origin\_latitude* (начальную долготу),  
*scale\_factor* (коэффициент искажения),  
*false\_easting* (восточное смещение) и  
*false\_northing* (северное смещение).

Для дальнейшей информации о проекциях и координатных системах смотрите документацию MapInfo Professional.

Каждая программа MapBasic может иметь свои режимы предложения **CoordSys**, задающие индивидуальные координатные системы. Если программа MapBasic выполняет **Оператор Set CoordSys**, на другие действующие программы MapBasic это не оказывает никакого эффекта.

### Примеры

Оператор **Оператор Set Map** задает режим представления существующей Карты. Представленный в примере **Оператор Set Map** задает режим показа Карты в проекции Робинсона:

```
Set Map CoordSys Earth Projection 12, 12, "m", 0.
```

Первое число 12 определяет проекцию Робинсона; второе число 12 определяет применение этой проекции ко всему Земному шару; параметр "m" задает метры, как единицы измерения; ноль задает нулевую долготу как начальную.

Следующий оператор задает показ карты без проекций:

```
Set Map CoordSys Earth
```

В следующем примере открывается таблица WORLD, затем выполняется **Оператор Commit Table**, сохраняющий ее под именем RWORLD. Новая таблица RWORLD сохраняется в проекции Робинсона.

```
Open Table "world" As World
Table world As "RWORLD.TAB"
CoordSys Earth Projection 12, 12, "m", 0.
```

В следующем примере проекция в одном окне Карты устанавливается такой же как в другом. В этом примере подразумевается, что целочисленные переменные *first\_map\_id* и *second\_map\_id* уже содержат идентификаторы двух окон Карт.

```
Set Map
Window second_map_winid
CoordSys Window first_map_winid
```

Следующий пример определяет систему координат DCS, которая получена из системы координат UTM Zone 10, путем аффинного преобразования:

```
x1 = 1.57x - 0.21y + 84120.5
y1 = 0.19x + 2.81y - 20318.0
```

Здесь координаты (x1 , y1) представляют полученные координаты для DCS, а (x, y) исходные координаты UTM Zone 10. Если координаты DCS исчислялись в футах, то предложение **CoordSys** для DCS выглядело бы следующим образом:

```
CoordSys Earth
Projection 8, 74, "m", -123, 0, 0.9996, 500000, 0
Affine Units "ft", 1.57, -0.21, 84120.5, 0.19, 2.81, -20318.0
```

**См. также:**

**Оператор Commit Table, Оператор Set CoordSys, Оператор Set Map**

---

## Функция CoordSysName

### Назначение

Возвращает строку с названием системы координат из описания системы координат MapBasic.

### Синтаксис

```
CoordSysName$ ( string )
```

## Функция Cos( )

---

### Возвращаемая величина

Строка

### Пример:

```
Note CoordSysName$("CoordSys Earth Projection 1, 62")
```

Показывает в диалоге MapInfo следующую строку:

```
Longitude / Latitude (NAD 27 for Continental US)
```

**Внимание:** Если системы координат с таким названием не существует в файле MAPINFO.PRJ, например, когда план-схема дана в геодезических футах, то эта функция возвратит пустую строку.

```
Note CoordSysName$("CoordSys NonEarth Units " + ""survey ft"" +  
"Bounds (0, 0) (10, 10)")
```

Если параметр CoordSys передан некорректно (заданы неправильные единицы измерения):

```
Note CoordSysName$("CoordSys Earth Projection 3, 74, " + ""foo"" +  
"-90, 42, 42.7333333333, 44.0666666667, 1968500, 0")
```

то будет возвращена ошибка о неправильной системе координат (Ошибка #727).

```
Invalid Coordinate System: CoordSys Earth Projection <content>
```

---

## Функция Cos( )

### Назначение

Вычисляет косинус.

### Синтаксис

```
Cos(num_expr)
```

*num\_expr* - численное выражение угла в радианах.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Cos( )** вычисляет косинус числа, полученного в результате вычисления выражения *num\_expr*. Диапазон возвращаемого функцией **Cos( )** значения находится между единицей и минус единицей включительно.

Для перевода градусов в радианы число необходимо умножить на число DEG\_2\_RAD. Для обратного конвертирования используется коэффициент RAD\_2\_DEG. Чтобы Ваша программа могла использовать эти коэффициенты конвертирования, она должна содержать оператор Include "MAPBASIC.DEF".

**Пример:**

```
Include "MAPBASIC.DEF"Dim x, y As Float x = 60 * DEG_2_RAD y = Cos(x) ' y
равен 0.5, поскольку' косинус 60 градусов равен 0.5
```

**См. также:**

**Функция Acos( ), Функция Asin( ), Функция Atn( ), Функция Sin( ), Функция Tan( )**

---

## Оператор Create Arc

**Назначение**

Создает объект типа "дуга".

**Синтаксис**

```
Create Arc
[ Into { Window window_id | Variable var_name } ]
( x1, y1 ) ( x2, y2 )
start_angle end_angle
[ Pen... ]
```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*x1, y1* – координаты одного угла минимального прямоугольного покрытия (МПП), прямоугольника, описывающего дугу;

*x2, y2* – координаты противоположного по диагонали угла МПП дуги;

*start\_angle* – значение начального угла дуги, в градусах;

*end\_angle* – значение конечного угла дуги, в градусах;

Слово **Pen** начинает стандартное предложение для назначения стиля линии.

**Описание**

Результатом действия оператора **Create Arc** является новый объект типа "дуга".

Если оператор использует предложение **Into Variable**, то созданный объект объявляется как значение объектной переменной. Если слово **Into** указывает окно, объект помещается на заданное место в окне (например, на изменяемый слой). Если **Into** вообще нет в операторе, MapBasic попытается создать объект в самом верхнем окне. Если это невозможно (например, поверх всех окон лежит окно Графика), то объект не будет создан.

Параметры *x* и *y* задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**. При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты. Если объект создается для окна Отчет, измерения производятся в объявленных ранее единицах измерения листа: X-координата – это расстояние от левого края листа до точки, и Y-координата – расстояние от верхнего края листа. По умолчанию, MapBasic

использует дюймы. Чтобы задать другие "бумажные" единицы измерения, используйте **Оператор Set Paper Units**. Перед созданием объекта в окне Отчета не забудьте выполнить оператор Set CoordSys Layout.

Предложение **Pen** задает стиль линии; см. подробности в разделе **Предложение Pen on page 492**. Если не задан параметр Pen, в операторе **Create Arc** будет использован стандартный стиль линии MapInfo Professional (заданный в диалоге команды **Настройка > Стиль линий**).

**См. также:**

**Оператор Insert, Предложение Pen, Оператор Update, Оператор Set CoordSys**

---

## Оператор Create ButtonPad

### Назначение

Создает инструментальную панель.

### Синтаксис

```
Create ButtonPad { title_string | ID pad_num } As
    button_definition [ button_definition ... ]
[ Title title_string ]
[ Width w ]
[ Position ( x, y ) [ Units unit_name ] ]
[ ToolbarPosition ( row, column ) ]
[ { Show | Hide } ]
[ { Fixed | Float } ]
```

*title\_string* – заголовок инструментальной панели;

*pad\_num* – идентификатор стандартной инструментальной панели:

- 1 – для панели "Операции"
- 2 – для панели "Пенал"
- 3 – для панели "Программы"
- 4 – для панели "Команды"
- 5 – для панели "ODBC"

*w* – ширина панели, измеряется количеством кнопок;

*x, y* – координаты верхнего левого угла панели в "бумажных" единицах измерения;

*unit\_name* – имя "бумажной" единицы (например, "in" – дюйм, "cm" – сантиметр);

*row, column* – координаты инструментальной панели, когда она находится в прикрепленном состоянии (docked); например, координаты 0, 0 означают расположение строки инструментов прижатой к левому и верхнему краям рабочего окна, а 0, 1 задают положение панели второй строкой).

Каждый параметр *button\_definition* является либо ключевым словом **Separator**, либо группой предложений следующего синтаксиса:

```

{ PushButton | ToggleButton | ToolButton }
  Calling { procedure | menu_code | OLE methodname | DDE server, topic }
  [ ID button_id ]
  [ Icon n [ File file_spec ] ]
  [ Cursor n [ File file_spec ] ]
  [ DrawMode dm_code ]
  [ HelpMsg msg ]
  [ ModifierKeys { On | Off } ]
  [ Enable ] [ Disable ]
  [ Check ] [ Uncheck ]

```

*procedure* – имя процедуры-обработчика, вызываемой при нажатии на кнопку;

*menu\_code* – стандартный код меню MapInfo из файла MENU.DEF (например, M\_FILE\_OPEN); MapInfo Professional выполняет команду меню, когда пользователь нажимает на кнопку.

*methodname* – строковая величина, задающая имя OLE-метода;

*server, topic* – строковая величина, задающая DDE-сервера и имя темы (topic);

**ID button\_id** задает кнопке уникальный номер. Этот номер можно будет потом использовать как ее идентификатор, в случае, если несколько кнопок вызывают один обработчик. Его также использует **Onepatop Alter Button**.

**Icon n** задает картинку, которая будет на кнопке; *n* может быть одним из специальных кодов из файла ICONS.DEF (например, MI\_ICON\_RULER). Подпредложение **File** задает файл ресурсов изображений, в этом случае параметр *n* должен быть целочисленным идентификатором одного из ресурсов файла.

**Cursor n** задает форму, которую примет указатель мыши после выбора кнопки; *n* может быть одним из специальных кодов из файла ICONS.DEF (например, MI\_CURSOR\_ARROW). Это предложение может входить только в описание кнопки инструмента (тип ToolButtons). Подпредложение **File** задает файл ресурсов изображений, в этом случае параметр *n* должен быть целочисленным идентификатором одного из ресурсов файла.

**DrawMode dm\_code** задает возможность инструмента рисовать (использование возможности передвигать мышку с нажатой клавишей) или только указывать (использование только возможности нажимать на клавишу мышки), при этом параметр *dm\_code* должен быть одним из специальных кодов из файла ICONS.DEF (например, DM\_CUSTOM\_LINE). Предложение **DrawMode** может входить в описание кнопки инструмента (тип ToolButtons).

**HelpMsg msg** задает текст подсказки, которая появляется в строке сообщений при указании на кнопку, а также может задавать текст для плавающей подсказки ToolTip. Первая часть строки *msg* показывается в строке сообщений. Если строка *msg* включает в себя \n, то текст, следующий за \n, отображается в подсказке ToolTip.

**ModifierKeys** управляет использованием клавиш SHIFT и CTRL в режиме рисования, сопровождающемся прорисовкой образа объекта ("rubberband"), инструментом кнопки типа ToolButton. По умолчанию используется режим **Off**, не использующий клавиши SHIFT и CTRL.

## Описание

Оператор **Create ButtonPad** используется для создания новой инструментальной панели. Создав панель инструментов, можно ее изменять, используя **Оператор Alter Button** и **Оператор Alter ButtonPad**.

Каждая инструментальная панель может быть скрыта. Чтобы инструментальная панель была изначально скрыта, задайте слово **Hide**. Каждая инструментальная панель показывается либо прикрепленной к краю рабочего экрана, либо свободно "плавающей". "Плавающая" панель, также как и панель Информация, является отдельным окном. Для создания закрепленной панели используется слово **Fixed**. Для создания "плавающей" панели используется слово **Float**. Если панель "плавающая", ее начальное положение задается предложением **Position**; если же она прикреплена, то положение задается предложением **ToolbarPosition**.

Более подробно инструментальные панели описаны в *Руководстве пользователя MapBasic*. См. также раздел **Оператор Alter ButtonPad on page 80**.

## Режимы предложения Calling

Предложение **Calling** задает, что должно случиться, если пользователь нажмет на кнопку инструментальной панели. В таблице приведены возможные примеры использования этого предложения:

| Примеры                                   | Описание  |
|---|---|
| <code>Calling M_FILE_NEW</code>           | Если за словом <b>Calling</b> идет целочисленный код из файла MENU.DEF, MapInfo запускает на выполнение соответствующую стандартную команду MapInfo (например, <b>Файл &gt; Новая таблица</b> ).                      |
| <code>Calling my_procedure</code>         | Если задано имя sub-процедуры, MapInfo передает управление этой процедуре. Эта процедура должна быть частью программы MapBasic, создавшей эту инструментальную панель.  |
| <code>Calling OLE "methodname"</code>     | Только для Windows. MapInfo Professional управляет событиями, обращаясь с именем метода к объекту OLE Automation, установленному SetCallback-методом MapInfo. Детали см. в <i>Руководстве пользователя MapBasic</i> . |
| <code>Calling DDE "server","topic"</code> | Только для Windows. MapInfo управляет событиями, присоединяясь через DDE к паре "сервер тема" и посылая выполняемое сообщение на DDE-сервер.  |



В последних двух случаях строка послания для OLE или DDE сервера должна начинаться с трех символов "MI:" для того, чтобы сервер смог определить, что послание пришло от MarInfo. Остальная часть этой строки состоит из разделенного запятыми списка значений возврата функций с CommandInfo(1) по CommandInfo(8). Полное описание синтаксиса этой строки см. в *Руководстве пользователя MarBasic*.

### Пример:

```
Create ButtonPad "Новые Кнопки" As PushButton HelpMsg "Нажмите на эту
кнопку для вывода диалога запроса" Calling button_sub_proc Icon
MI_ICON_ZOOM_QUESTION ToolButton HelpMsg "Используйте этот инструмент для
рисования нового маршрута" Calling tool_sub_proc Icon MI_ICON_CROSSHAIR
DrawMode DM_CUSTOM_LINE ToggleButtonHelpMsg "Переключение показа
расстояний" Calling toggle_prox_check Icon MI_ICON_RULER Check Title
"Средства" Width 3Show
```

### См. также:

[Оператор Alter Button](#), [Оператор Alter ButtonPad](#)

---

## Оператор Create ButtonPads As Default

### Назначение

Восстанавливает стандартный вид и состав инструментальных панелей.

### Синтаксис

**Оператор Create ButtonPads As Default**

### Описание

Восстанавливает стандартные инструментальные панели в их начальном положении.

Используйте этот оператор осторожно. Оператор **Create ButtonPads As Default** убирает все инструментальные панели, построенные приложением, и восстанавливает три стандартные панели в их прежнем виде: Операции, Пенал и Программы.

### См. также:

[Оператор Alter Button](#), [Оператор Alter ButtonPad](#), [Оператор Create ButtonPad](#)

---

## Оператор Create Cartographic Legend

### Назначение

Позволяет создать и отобразить стиль картографических легенд, также как и тематических легенд для активного окна Карты.

### Синтаксис

```
Create Cartographic Legend
[ From Window map_window_id ]
[ Behind ]
[ Position ( x, y ) [ Units paper_units ] ]
[ Width win_width [ Units paper_units ] ]
[ Height win_height [ Units paper_units ] ]
[ Window Title { legend_window_title }
```

```
[ ScrollBars { On | Off } ]
[ Portrait | Landscape | Custom ]
[ Style Size { Small | Large } ]
[ Default Frame Title { def_frame_title } [ Font... ] } ]
[ Default Frame Subtitle { def_frame_subtitle } [ Font... ] } ]
[ Default Frame Style { def_frame_style } [ Font... ] } ]
[ Default Frame Border Pen [ [ pen_expr ]
Frame From Layer { map_layer_id | map_layer_name
  [ Using
    [ Column { column | Object } [ FromMapCatalog { On | Off } ] ]
    [ Label { expression | Default } ]
  [ Position ( x, y ) [ Units paper_units ] ]
  [ Title { frame_title [ Font... ] } ]
  [ SubTitle { frame_subtitle [ Font... ] } ]
  [ Border Pen pen_expr ]
  [ Style [ Font... ] [ Norefresh ] [ Text { style_name }
    { Line Pen... | Region Pen... Brush... | Symbol Symbol... } |
    Collection [ Symbol ... ]
  [ Line Pen... ] [ Region Pen... Brush ... ] } ]
  [ , ... ]
```

*map\_window\_id* – целочисленный идентификатор окна, для получения которого используются **Функция FrontWindow( )** и **Функция WindowID( )**;

*x* – определяет требуемое расстояние от верхнего края рабочего стола MapInfo до верхнего угла окна Легенды;

*y* – определяет требуемое расстояние от левого края рабочего стола MapInfo до левого угла окна Легенды;

*paper\_units* – строка, представляющая "бумажные" единицы измерения (например, "cm" для сантиметров).

*win\_width* - ширина окна

*win\_height* - высота окна

*legend\_window\_title* – это строковое выражение, соответствующее заголовку окна Легенды, по умолчанию это "Legend of xxx" где xxx это заголовок окна Карты;

*def\_frame\_title* – это строковая величина, которая по умолчанию определяет заголовок раздела Легенды. Эта величина может включать специальный символ "#", который будет замещаться именем текущего слоя.

*def\_frame\_subtitle* – это строковая величина, которая по умолчанию определяет заголовок подраздела легенды. Эта величина может включать специальный символ "#", который будет замещаться именем текущего слоя.

*def\_frame\_style* – то строковая величина, которая определяет каждый символ в каждом разделе Легенды. Символ "#" будет замещаться именем текущего слоя. Символ % будет замещаться словами "Line", "Point", "Region", которые будут соответствовать типу географических объектов (эти названия могут быть другими в локализованной версии). Например, "% of #" будет соответствовать тексту "Region of States" для слоя STATES.TAB.

*pen\_expr* – это выражение, возвращающее объект типа Pen, например, *MakePen(width, pattern, color)*. Если по умолчанию линия рамки определена, то она останется по умолчанию такой же для раздела Легенды. Если предложение Pen для рамки задано для раздела Легенды, то оно будет определять тип линии для рамки вместо заданного по умолчанию.

*map\_layer\_id* или *map\_layer\_name* определяют слой карты; это может быть целая величина Smallint (например, используйте 1 для определения самого верхнего слоя, не считая косметического) или строковая величина, соответствующая имени таблицы, отображенной на карте. Для тематического слоя необходимо определять параметр *map\_layer\_id*.

*frame\_title* – строковая переменная, определяющая заголовок раздела легенды; Если будет определено предложение заголовка раздела **Title**, то будет использоваться эта величина вместо величины *def\_frame\_title*.

*frame\_subtitle* – строковая переменная, определяющая подзаголовок раздела Легенды; если предложение **Subtitle** определено, то оно будет использоваться вместо имени, задаваемого по умолчанию величиной *def\_frame\_subtitle*.

*column* – имя атрибутивной колонки из раздела слоя таблицы.

*style\_name* – строковая переменная, которая указывает к какому типу: символу, линии или области относится объект в разделе легенды.

### Описание

Оператор **Create Cartographic Legend** позволяет создать и отобразить стиль картографических легенд, также как и тематических легенд для активного окна карты. Каждый стиль картографической и тематической легенды будет связан только с одним окном карты, так что одновременно может быть открыто несколько окон легенды.

Вы можете создать раздел легенды для каждого картографического или тематического слоя, который Вы захотите включить в легенду. Картографические и тематические разделы будут включать заголовки и подзаголовки легенды. Картографические разделы легенды показывают стили слоев карты; разделы легенды отражают цвета, символы и их размер для тематических слоев. Вы можете создать разделы, которые имеют стили, основанные на стиле окна карты или создать Ваши собственные разделы легенды.

Как минимум, требуется задать одно предложение **Frame**.

Все предложения, относящиеся ко всей легенде (scrollbars, width, и др.) должны соотноситься с первым предложением **Frame**.

Предложение **From Layer** должно быть первым предложением после **Frame**.

Если задано предложение **Behind**, Легенда размещается после окна тематической карты.

Предложение **Position** (необязательное) контролирует положение окна легенды на рабочем столе. Верхний левый угол рабочего стола MapInfo Professional имеет координаты 0, 0. Дополнительные предложения **Width** и **Height** задают размер окна. В параметре "Position" используются "бумажные" единицы измерения, такие, как "in" (дюймы) или "cm" (сантиметры)). MapBasic имеет по умолчанию установку в дюймах; программа MapBasic может поменять единицы, используя . Оператор **Create Cartographic Legend** может переопределить единицы измерения; для этого надо включить подпредложение **Units** в предложения **Position**, **Width** и/или **Height**.

и управляет показом строки (полосы) прокрутки.

**Portrait** и **Landscape** описывают ориентировку разделов легенды в окне. **Portrait** – это книжная ориентировка. **Landscape** – это альбомная ориентировка.

Если указано **Custom**, то Вы можете задать Ваше собственное предложение **Position** для позиционирования раздела.

Предложение **Position** определяет позиционирование раздела, если определено предложение **Custom**.

Предложение **Style Size** управляет размером образцов в окне Легенды. Если Вы задали режим **Style Size Small**, в окне Легенды будут показаны малые образцы. Если Вы задали режим **Style Size Large**, в окне Легенды будут показаны большие образцы.

Предложения **Position**, **Title**, **SubTitle**, **Border Pen** и **Style** для разделов легенды используются только для слоев карты. Они не используются для тематических слоев. Для тематических слоев вся необходимая информация задаётся автоматически при их создании.

Параметром этого оператора **Font** можно задать стиль оформления текста. Если по умолчанию заголовков раздела, подзаголовков или имя стиля объекта имеют определенный шрифт, то этот же шрифт по умолчанию будет использоваться и для раздела легенды. Если на уровне раздела определены предложения **Title**, **SubTitle** или **Style**, и в этих предложениях используется предложение **Font**, то будет использоваться шрифт, заданный этим предложением. Если ни на каком уровне шрифт не определен, то будет применяться текущий шрифт с размером букв 10, 9 и 8 для заголовка, подзаголовка и имени стиля соответственно.

Предложение **Style** и ключевое слово **NoRefresh** позволяют Вам создавать собственные разделы легенды, которые не будут изменяться при обновлении легенды. Если ключевое слово **NoRefresh** используется в предложении **Style**, то таблица не проверяется на предмет используемых стилей. Вместо этого предложение **Style** должно содержать Ваш собственный список стилей, используемых в разделе легенды. Это достигается заданием предложения **Text** и соответствующих предложений **Line**, **Region** или **Symbol**. Объекты типа Группа точек обрабатываются как точечные объекты.

Объекты типа Коллекция обрабатываются отдельно. Когда MapInfo Professional создает Легенду на основе типов объектов, то сначала рисуются точечные символы, затем линии, а затем области. Объекты типа Коллекция изображаются последними. В пределах объекта типа Коллекция порядок рисования таков: точки, линии, объекты.

Параметр предложения **Column**, если оно задано, есть название атрибутивной колонки в слое таблицы, для которой отображаются данные в разделе; либо предложение **Object** задает объектную колонку (а это значит, что стили Легенды основаны на уникальных стилях файла карты). По умолчанию используется **Object**.

**FromMapCatalog ON** извлекает стили из MapCatalog для таблицы прямого доступа. Если таблица не является таблицей прямого доступа, MapBasic возвращается к заданному по умолчанию поведению для таблицы с непрямым доступом вместо того, чтобы допустить ошибку. Стандартное поведение для таблицы с отключённым доступом это **FromMapCatalog Off** (то есть, используются стили карты).

**FromMapCatalog OFF** извлекает уникальные стили карты для таблицы прямого доступа сервера. Эта таблица должна быть таблицей прямого доступа, которая поддерживает стилизацию каждой записей. Если таблица прямого доступа не поддерживает стили записей, то применяется заданное по умолчанию поведение для таблиц прямого доступа – используются заданные по умолчанию стили из MapCatalog (**FromMapCatalog ON**).

Если задано предложение Label, то нужно либо задать корректное выражение *expression*, либо слово **Default** (означающее, что если стилевое предложение не содержит явно заданный текст, то используется стандартный режим заполнения легенды). Стандартным значением является **Default**.

Сначала, для каждого слоя, порождающего раздел легенды, исследуется TAB-файл на предмет наличия значений метаданных для заголовка легенды, подзаголовка, структуры колонок и т.п. Если для колонки не заданы метаданные, то по умолчанию используется **Object**. Если в метаданных нет данных для предложения Label, то по умолчанию используются текущие установки стиля раздела. Если же в метаданных есть значения, но их нужно заменить своими, то это можно сделать с помощью операторов MapBasic.

### Пример:

В следующем примере показано, как создать раздел в картографической легенде для окна Карты. У окон Легенды есть одна особенность: чтобы создать раздел в окне Легенды, нужно использовать предложение **Title** вместо **From Window**.

```
Dim i_layout_id, i_map_id As Integer Dim s_title As String
' here, you would store the Map window's ID in i_map_id,
' and store the Layout window's ID in i_layout_id.
' To obtain an ID, call FrontWindow( ) or WindowID( ).
s_title = "Тематическая легенда " + WindowInfo(i_map_id, WIN_INFO_NAME) Set
CoordSys Layout Units "in" Create Frame Into Window i_layout_id(1,2) (4,
5) Title s_title
```

Так создается раздел в окне картографической Легенды. Чтобы создать раздел для тематической Легенды, замените заголовок следующим образом:

```
S_title="Тематическая легенда: " + WindowInfo (I_map_id, WW_INFO_NAME)
```

### См. также:

**Оператор Set Cartographic Legend, Оператор Alter Cartographic Frame, Оператор Add Cartographic Frame, Оператор Remove Cartographic Frame, Оператор Create Legend, Оператор Set Window, Функция WindowInfo( )**

---

## Функция CreateCircle( )

### Назначение

Возвращает объект "окружность".

### Синтаксис

```
CreateCircle( x, y, radius )
```

$x$  – X-координата центра окружности (или широта), действительное число;

$y$  – Y-координата центра окружности (или долгота), действительное число;

*radius* – действительное число, назначающее радиус окружности.

## Возвращаемая величина

Объект

## Описание

Функция **CreateCircle( )** возвращает графический объект типа "окружность".

Параметры  $x$  и  $y$  задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**.

**Внимание:** При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты.

Параметр *radius* назначается в тех единицах измерения, которые были назначены MapBasic до выполнения этой функции. По умолчанию MapBasic использует мили, **Оператор Set Distance Units** может задать другие единицы измерения в среде MapBasic.

Окружность заполняется штриховкой текущего стиля. Чтобы создать окружность с заданной штриховкой, выполните **Оператор Set Style** перед **CreateCircle( )**. Другой способ - вместо **CreateCircle( )** выполнить **Оператор Create Ellipse**, в котором задать предложения **Pen** и **Brush**.

Графический объект, созданный функцией **CreateCircle( )**, может быть присвоен объектной переменной, которая задает значение для уже существующей строки таблицы (**Оператор Update**) или вновь созданной (**Оператор Insert**).

**Внимание:** Перед созданием объекта в окне Отчета не забудьте выполнить оператор Set CoordSys Layout.

## Ошибки:

Функция вернет код ошибки ERR\_FCN\_ARG\_RANGE, если значение аргумента выходит за пределы, заданные при его определении.

## Примеры

В примере используется **Оператор Insert** для создания новой строки в таблице SITES. Функция **CreateCircle( )** вложена в оператор Insert, и при его выполнении создается объект "окружность", данные которого содержатся в новой строке (записи).

```
Open Table "sites"
Insert Into sites (obj)
  Values ( CreateCircle(-72.5, 42.4, 20) )
```

В следующем примере используется таблица TOWERS, которая имеет три колонки: "Xcoord", "Ycoord" и "Radius". Колонки "Xcoord" и "Ycoord" содержат значения долготы и широты, где находятся радиостанции, а колонка "Radius" – значения радиусов областей их вещания. Каждая запись в таблице описывает радиотрасляционную башню, а колонка "Radius" column задает область распространения сигнала каждой башни.

Сначала **Оператор Update** (по порядку выполнения) функция **CreateCircle( )** создает окружности для каждой строки таблицы. Затем оператор Update присваивает окружность каждой записи таблицы Towers. Каждая окружность радиус из колонки "Radius", а центр будет задан координатами из колонок "Xcoord" и "Ycoord".

```
Open Table "towers"
Update towers
  Set obj = CreateCircle(xcoord, ycoord, radius)
```

**См. также:**

**Оператор Create Ellipse, Оператор Insert, Оператор Update**

---

## Оператор Create Collection

### Назначение

Объединяет точечные, линейные и площадные объекты в один объект. Коллекция показывается в окне Списка в виде одной записи.

### Синтаксис

```
Create Collection [ num_parts ]
  [ Into { Window window_id | Variable var_name } ]
  Multipoint
    [ num_points ]
    ( x1, y1 ) ( x2, y2 ) [ ... ]
    [ Symbol... ]
  Region
    num_polygons
    [ num_points1 ( x1, y1 ) ( x2, y2 ) [ ... ] ]
    [ num_points2 ( x1, y1 ) ( x2, y2 ) [ ... ] ... ]
    [ Pen... ]
    [ Brush... ]
    [ Center ( center_x, center_y ) ]
  Pline
    [ Multiple num_sections ]
    num_points
    ( x1, y1 ) ( x2, y2 ) [ ... ]
    [ Pen... ]
    [ Smooth... ]
```

*num\_parts* – число непустых частей в коллекции. Это число от 0 до 3 является дополнительным кодом MapBasic (но для MIF задание этого кода обязательно).

*num\_polygons* – число полигонов в коллекции.



*num\_sections* – задает, из скольких частей может состоять сегментированная (multi-section) полилиния;

### Пример:

```
create collection multipoint 2 (0,0) (1,1) region 3 3 (1,1) (2,2) (3,4) 4
(11,11) (12,12) (13,14) (19,20) 3 (21,21) (22,22) (23,24) pline 3 (-1,1)
(3,-2) (4,3)
dim a as object
create collection into variable a multipoint 2 (0,0) (1,1) region 1 3
(1,1) (2,2) (3,4) pline 3 (-1,1) (3,-2) (4,3)
insert into test (obj) values (a)
create collection region 2 4 (-5,-5) (5,-5) (5,5) (-5,5) 4 (-3,-3) (3,-3)
(3,3) (-3,3) pline multiple 2 2 (-6,-6) (6,6) 2 (-6,6) (6,-6) multipoint 6
(2,2) (-2,-2) (2,-2) (-2,2) (4,1) (-1,-4)
```

### См. также:

**Оператор Create MultiPoint**

## Оператор Create Cutter

### Назначение

Создает объект-область, который будет использоваться как разрезающий объект в операции разрезания, а также изменяемый объект или группу изменяемых объектов. Вы должны назначить изменяемый объект, а также задать разрезающий объект в виде набора полилиний до того, как запустите операцию разрезания.

### Синтаксис

**Create Cutter Into Target**

### Описание

Перед использованием Create Cutter, должны быть выбраны один или более объектов полилиний и должен существовать изменяемый объект. Это можно сделать командой

**Объекты > Выбрать изменяемый объект** или выполнив **Оператор Set Target**. Объекты-полилинии, содержащиеся в выборке, должны быть представлены одним непрерывным объектом, без разрывов и самопересечений.

Полилиния должна пересекать МОП (минимальный описывающий прямоугольник) изменяемого объекта, который в свою очередь должен соответствовать требованиям операции разрезания. Полилиния, однако, сама не должна пересекать изменяемый объект. Например, если изменяемый объект представляет группу Гавайских островов, то полилиния может использоваться для разделения островной гряды на 2 части, при этом не касаясь ни одного отдельного острова. Если МОП изменяемого объекта не пересекает полилинию, то тогда такой объект будет удален из списка изменяемых объектов.

После выделения изменяемых объектов, рассчитывается МОП - минимальный описывающий прямоугольник всех этих объектов; сам МОП представляет пространство, которое будет разделено оператором. Затем полилиния продолжается, если необходимо, так что она

попадает в область МОП. Это достигается так: полилиния продолжается до пересечения с МОП в направлении отрезков между двумя последними точками на каждом конце этой полилинии. Растянутая полилиния разделяет пространство изменяемых объектов на две части. В результате будет создан и возвращен объект Область, представляющий одну из этих двух частей.

Этот оператор будет возвращать обработанное множество изменяемых объектов и новый обрезающий объект Область. Этот объект область будет вставлен в таблицу с изменяемыми объектами (которая должна быть изменяемой). Исходный объект(ы) полилиния останется, но перестанет быть выбранным. Выбранным станет новый объект Область. Если получившийся объект-область удовлетворяет нашим нуждам, то можно сразу же проводить операцию разрезания, так как изменяемые объекты и разрезающий объект уже выбраны.

**Внимание:** Обрезающий объект останется на слое с изменяемыми объектами. Вы можете удалить его вручную.

**Пример:**

```

Open Table "C:\MapInfo_data\TUT_USA\USA\STATES.TAB"
Open Table "C:\MapInfo_data\TUT_USA\USA\US_HIWAY.TAB"
Map from States, Us_hiway
select * from States where state = "NY"
Set target On
select * from Us_hiway where highway = "I 90"
Create Cutter Into Target
Objects Split Into Target

```

**См. также:**

**Оператор Set Target**

---

## Оператор Create Ellipse

**Назначение**

Создает эллипсы и окружности.

**Синтаксис****Create Ellipse**

```

[ Into { Window window_id | Variable var_name } ]
( x1, y1 ) ( x2, y2 )
[ Pen... ]
[ Brush... ]

```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*x1, y1* – координаты одного угла прямоугольника, описывающего эллипс;

*x2, y2* – координаты противоположного по диагонали угла прямоугольника;

Слово **Pen** начинает стандартное предложение для назначения стиля линии.

Слово **Brush** начинает стандартное предложение для назначения стиля штриховки.

**Описание**

Результатом действия оператора **Create Ellipse** является новый объект типа "эллипс" или "окружность". MapBasic создает объект, вписывая его в прямоугольник, задаваемый координатами двух противоположных углов. Если оператор задает квадрат, то будет создана окружность, иначе – эллипс.

Если оператор использует предложение **Into Variable**, то созданный объект объявляется как значение объектной переменной. Если слово **Into** указывает окно, объект помещается на заданное место в окне (например, на изменяемый слой). Если предложения **Into** вообще нет в операторе, MapBasic попытается создать эллипс в самом верхнем окне. Если это невозможно (например, поверх всех окон лежит окно Графика), то объект не будет создан.

Параметры *x* и *y* задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Используйте **Оператор Set CoordSys** для того, чтобы задать в среде MapBasic другую координатную систему. При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты. Если объект создается для окна Отчет, измерения производятся в объявленных ранее единицах измерения листа: X-координата – это расстояние от левого края листа до точки, и Y-координата – расстояние от верхнего края листа. По умолчанию, MapBasic использует дюймы. Чтобы задать другие "бумажные" единицы измерения, используйте **Оператор Set Paper Units**. Перед созданием объекта в окне Отчета не забудьте выполнить оператор Set CoordSys Layout.

Если явно задано **Предложение Pen**, то оно определяет стиль линии; см. подробности в разделе **Предложение Pen on page 492**. Если параметр Pen не задан, в операторе **Create Ellipse** будет использован стандартный стиль линии MapInfo Professional (заданный в диалоге команды **Настройка > Стиль линий**). Аналогично, предложение Brush назначает стиль штриховки; подробнее см. раздел **Предложение Brush on page 117**.

**См. также:**

**Предложение Brush, Функция CreateCircle( ), Оператор Insert, Предложение Pen, Оператор Update**

---

## Оператор Create Frame

### Назначение

Создает новый объект "рамка" в окне Отчета.

### Синтаксис

```
Create Frame
[ Into { Window layout_win_id | Variable var_name } ]
( x1, y1 ) ( x2, y2 )
[ Pen... ]
[ Brush... ]
[ Title title ]
[ From Window contents_win_id ]
[ FillFrame { On | Off } ]
```

*x1, y1* – координаты одного угла рамки;

*x2, y2* – координаты другого угла рамки;

*layout\_win\_id* – идентификатор окна Отчета, целое число;

*var\_name* – имя объектной переменной;

Слово **Pen** начинает стандартное предложение для назначения стиля линии.

Слово **Brush** начинает стандартное предложение для назначения стиля штриховки.

*title* – строка, задающая заголовок окна, изображение из которого будет помещено в рамку (например, "WORLD Map"; не имеет смысла использовать, если в операторе используется предложение **From Window**).

*contents\_win\_id* – идентификатор окна, изображение из которого будет помещено в рамку, целое число.

## Описание

Результатом действия оператора **Create Frame** является новый объект типа "рамка" в окне Отчета. Если не задан параметр *layout\_win\_id*, то создается новая рамка в самом верхнем окне Отчета. Перед созданием объекта в окне Отчета не забудьте выполнить оператор **Set CoordSys Layout**.

MapInfo запоминает установки в окне Отчета, вставляя оператор **Create Frame** в файл Рабочего Набора. Чтобы увидеть, как записывается оператор **Create Frame**, создайте Отчет, сохраните Рабочий набор и изучите его текст в текстовом редакторе.

Стиль линии для рамки задает **Предложение Pen**, а стиль штриховки - **Предложение Brush**.

Предложение **From Window** задает, содержимое какого окна будет показано в рамке. Например, чтобы в рамке показывалось окно Карты, задайте оператора **From Window i\_map**, где *i\_map* является идентификатором окна Карты. Для этого необходимо знать идентификатор окна. Окно должно быть уже открыто в MapInfo.

Предложение **Title** является альтернативным способом задания окна, изображение из которого будет показано в рамке. Например, для окна Карты, в котором показаны данные таблицы WORLD, предложение **Title** будет таким: **Title "WORLD Map"**. Если параметр *title* не указывает на существующее окно или если *title* является пустой строкой (""), оператор создаст пустую рамку. Если в операторе присутствуют сразу два предложения: **Title** и **From Window**, то будет исполнено только последнее из них.

При создании рамки для окна Карты, можно применять предложение **FillFrame**, чтобы управлять заполнением рамки Картой. Задание **FillFrame On** помещает всю Карту в рамку. (Аналогом этого является установка флажка "Заполнить всю рамку" диалога MapInfo Professional "Рамка".) Если задать режим **FillFrame Off** или опустить все предложение **FillFrame**, то пропорции экрана будут влиять на пропорции рамки в Отчете.

## Пример:

Следующие примеры показывают, как можно создавать разделы для картографических и для тематических легенд.

Тематические легенды обрабатываются особым образом. Чтобы создать раздел в окне Тематической Легенды, нужно использовать предложение **Title** вместо **From Window**.

```
Dim i_layout_id, i_map_id As Integer
Dim s_title As String

' here, you would store the Map window's ID in i_map_id,
' and store the Layout window's ID in i_layout_id.
' To obtain an ID, call FrontWindow( ) or WindowID( ).

s_title = "Тематическая легенда " + WindowInfo(i_map_id, WIN_INFO_NAME)
Set CoordSys Layout Units "in" Create Frame Into Window i_layout_id(1,2) (4,
5) Title s_title
```

Для создания картографической легенды в том случае, если несколько Карт сопровождаются Легендами, нужно использовать предложение **From Window** с указанием того, какая из легенд (если их несколько) обрабатывается.

Dim i\_cartlgnd\_id As Integer' теперь ID-номер картографической легенды ' будет содержаться в переменной i\_cartlgnd\_id,' этот ID-номер можно получить, вызвав call FrontWindow( ) или WindowID( ).

```
Create Frame
  Into Window i_layout_id
  (1,2) (4, 5)
  From Window i_cartlgnd_id
```

**См. также:**

**Предложение Brush, Оператор Insert, Оператор Layout, Предложение Pen, Оператор Set CoordSys, Оператор Set Layout, Оператор Update**

---

## Оператор Create Grid

### Назначение

Создает грид-файл, который MapBasic показывает в виде растровой таблицы в окне Карты.

### Синтаксис

```
Create Grid
  From tablename
  With expression [ Ignore value_to_ignore ]
  Into filespec [ Type grid_type ]
  [ Coordsys... ]
  [ Clipping { Object obj } | { Table tablename } ]
  Inflect num_inflections By Percent at
    color : inflection_value [ color : inflection_value ...]
  [ Round rounding_factor ]
  {[ Cell Size cell_size [ Units distance_unit ]] | [ Cell Min n_cells ]}
  [ Border numcells ]
  Interpolate With interpolator_name Version version_string
  Using num_parameters parameter_name : parameter_value
  [ parameter_name : parameter_value ... ]
```

*tablename* – это "псевдоним" имени открытой таблицы, из которой берутся точки для расчета поверхности (грида).

*expression* – это выражение, которое выделяет необходимую часть таблицы, например, имя колонки.

*value\_to\_ignore* – значение, которое будет проигнорировано; это обычно ноль. Строка, в которой присутствует игнорируемое значение, не будет учитываться при создании регулярной поверхности.

*filespec* – определяет полный путь и новое имя грид-файла. У этого нового файла будет расширение MIG.

*grid\_type* – строка, в которой задан тип файла регулярной поверхности, который будет создан. По умолчанию это .MIG файл.

**Coordsys** – необязательный параметр CoordSys, определяющий координатную систему создаваемой регулярной поверхности. Если это предложение не используется, то грид файл будет иметь ту же систему координат, что и исходная таблица. См. подробности в разделе: "**Оператор CoordSys на стр. 33**".

*obj* – это объект, который обрезает ячейки грида. Будет отображена только часть ячеек внутри такого объекта. Если ячеек грида внутри такого объекта нет, то никаких значений для ячеек записано не будет, и запишутся ячейки с нулевыми значениями.

*tablename* – имя таблицы, содержащей объекты типа полигонов, которые будут объединены в единый полигон и будут использованы для обрезания ячеек грида;

*num\_inflections* – числовое выражение, определяющее значения для пары *color:inflection\_value*.

*color* – это выражение для обозначения цвета для пары *color:value inflection*;

*inflection\_value* – числовое выражение, определяющее числовое значение для пары *color:inflection\_value*.

*cell\_size* – числовое выражение, определяющее размер ячейки грида в единицах расстояния;

*n\_cells* – числовое выражение, которое определяет высоту и ширину грида в количестве ячеек;

*numcells* – число ячеек, которое будет добавлено вокруг границы грида. Число *numcells* задается для увеличения размер грида сверху, снизу, слева и справа;

*distance\_unit* – строковое выражение, определяющее единицы измерения размера ячейки. Задавать этот параметр необязательно. Если эта величина не задана, то используются единицы координатной системы из обрабатываемой таблицы.

*interpolator\_name* – строковое выражение, определяющее имя интерполятора, который используется для создания грида;

*version\_string* – это строковое выражение, определяющее версию интерполятора;

*num\_parameters* – это числовое выражение, определяющее число параметров интерполятора - т.е. число пар *parameter\_name:parameter\_value*.

*parameter\_name* – строковое выражение, определяющее имя в паре *parameter\_name:parameter\_value*.

*parameter\_value* – числовое выражение значение в паре *parameter\_name:parameter\_value*.

**By Percent at** сообщает, что следующие пары *color:inflection\_value* представляют цвета и процентные значения;

**Round** – это числовое выражение, задающее величину округления для пары имя-значение.

### Описание

Тематическая растровая поверхность интерполируется без разрывов по точечным данным. Оператор **Create Grid** берет данные из колонок, которые находятся в таблице с точечными данными, и передает координаты этих точек и их числовые значения интерполятору. Интерполятор создает растровый грид-файл, который MapBasic отображает в виде растровой таблицы в окне карты.

Оператор **Create Grid** считывает значения (x, y, z) из таблицы, определенной в предложении **From**. Он получает значения z, которые указаны в относящемся к данной таблице предложении **With**.

Размеры грида (сетки) могут быть определены двумя способами. Первый способ определяет размер ячейки грида, выраженной в единицах расстояния, например, милях. Другой способ заключается в задании количества ячеек грида по ширине и высоте. Например, если Вы хотите получить грид размером не менее 200 ячеек по ширине и 200 ячеек по высоте, то надо будет определить "cell min 200". В зависимости от площади, покрываемой гридом, действительный размер грида будет не менее 200 на 200.

### Пример:

```
Open Table "C:\States.tab" Interactive
Map From States
Open Table "C:\Us_elev.tab" Interactive
Add Map Auto Layer Us_elev
set map redraw off
Set Map Layer 1 Display Off
set map redraw on
```

```
create grid
  from Us_elev
  with Elevation_FT
  into "C:\Us_elev_grid"
  clipping table States
  inflect 5 at
    RGB(0, 0, 255) : 13
    RGB(0, 255, 255) : 3632.5
    RGB(0, 255, 0) : 7252
    RGB(255, 255, 0) : 10871.5
    RGB(255, 0, 0) : 14491
  cell min 200
  interpolate
    with "IDW" version "100"
    using 4
      "EXPONENT": "2"
      "MAX POINTS": "25"
      "MIN POINTS": "1"
      "SEARCH RADIUS": "100"
```

### См. также:

[Оператор Set Map](#)



## Оператор Create Index

### Назначение

Создает индекс для колонки в открытой таблице.

### Синтаксис

**Create Index On** *table* (*column*)

*table* – имя открытой таблицы;

*column* – имя колонки в открытой таблице.

### Описание

Оператор **Create Index** создает индекс для определенной колонки открытой таблицы. MapInfo Professional использует проиндексированные колонки при выполнении операций, таких как:

**Запрос > Найти**. Индексы также улучшают выполнение запросов.

**Внимание:** MapInfo Professional MapInfo не может индексировать колонки в таблице, в которой есть несохраненные изменения. Для сохранения изменений используется **Оператор Commit Table**.

### Пример:

Создается индекс колонки "Столица" в таблице WORLD:

```
Open Table "world" Interactive
Create Index on World(Capital)
```

### См. также:

**Оператор Alter Table, Оператор Create Table, Оператор Drop Index, Оператор Commit Table**

## Оператор Create Legend

### Назначение

Открывает новое окно тематической легенды для определенного окна Карты.

Начиная с версии 5.0 MapInfo Professional, оператор Create Cartographic Legend позволяет создать и отобразить легенду карты. См. подробности в разделе **Оператор Create Cartographic Legend на стр. 44**.

### Синтаксис

**Create Legend**

```
[ From Window window_ID ]
[ { Show | Hide } ]
```

*window\_ID* – целочисленный идентификатор открытого в рабочем окне MapInfo окна Карты.

### Описание

Этот оператор создает дополнительное, "плавающее" окно тематической легенды в рабочем окне MapInfo Professional. (Последнее Вы можете открыть, выполнив оператор Open Window Legend.)

Оператор **Create Legend** может быть Вам полезен, если необходимо вывести на экран Легенду Карты, когда окно Карты открыто, но не активно. Этот оператор полезен в приложениях, использующих т.н. "интегрированную картографию", в которых окно MapInfo вставляется в другую программу, например, созданную при участии Visual Basic. Концепция и описание интегрированной картографии содержатся в *Руководстве пользователя MapBasic*.

Если Вы добавите предложение **From Window**, новая тематическая легенда считается порожденной определенным в этом предложении окном; иначе новое окно Легенды считается порожденным последним активным окном Карты.

Если в операторе используется ключевое слово **Hide**, то окно будет создано в скрытом состоянии. Вывести на экран скрытое окно можно, использовав оператор Set Window...Show.

После выполнения оператора **Create Legend** определить идентификатор нового окна можно будет вызовом функции WindowID(0). Этим идентификатором Вы можете пользоваться в следующих операторах (таких как **Оператор Set Window**).

Новая тематическая легенда создается в соответствии с режимами наследования и стилизации, для задания которых используется **Оператор Set Next Document**.

### См. также:

**Оператор Create Cartographic Legend, Оператор Open Window, Оператор Set Next Document, Оператор Set Window**

---

## Функция CreateLine( )

### Назначение

Возвращает объект "линия".

### Синтаксис

**CreateLine**( *x1*, *y1*, *x2*, *y2* )

*x1* – X-координата начальной точки линии (например, долгота), действительное число;

*y1* – Y-координата начальной точки линии (или широта), действительное число;

*x2* – X-координата конечной точки линии, действительное число;

*y2* – Y-координата конечной точки линии, действительное число.

### Возвращаемая величина

Объект

## Описание

Функция **CreateLine( )** возвращает графический объект типа "линия". Параметры *x* и *y* задают координаты концов отрезка прямой линии в той координатной системе, которая была объявлена MapBasic ранее. По умолчанию, MapBasic использует долготу и широту. Используйте **Оператор Set CoordSys** для задания другой системы.

Создаваемый линейный объект будет нарисован с использованием текущей установки стиля линии Pen. Чтобы нарисовать линию в другом стиле, нужно выполнить **Оператор Set Style** перед вызовом **CreateLine( )** или выполнить **Оператор Create Line**, задав при этом **Предложение Pen**.

Графический объект, созданный функцией **CreateLine( )**, может быть присвоен объектной переменной, которая определяет значение уже существующей строки в таблице (**Оператор Update**) или вновь созданной (**Оператор Insert**). Перед созданием объекта в окне Отчета не забудьте выполнить оператор Set CoordSys Layout.

## Пример:

В примере используется **Оператор Insert** для создания новой строки в таблице ROUTES. Функция **CreateLine( )** используется внутри оператора Insert.

```
Open Table "Routes"
Insert Into routes (obj)
  Values (CreateLine(-72.55, 42.431, -72.568, 42.435))
```

## См. также:

**Оператор Create Line, Оператор Insert, Оператор Update**

# Оператор Create Line

## Назначение

Создает объект "прямая линия".

## Синтаксис

```
Create Line
  [ Into { Window window_id | Variable var_name } ]
  ( x1, y1 ) ( x2, y2 )
  [ Pen... ]
```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*x1, y1* – координаты начала отрезка прямой;

*x2, y2* – координаты конца отрезка прямой линии.

Слово **Предложение Pen** начинает стандартное предложение для назначения стиля линии объекта.

### Описание

Результатом действия оператора **Create Line** является новый графический объект типа "прямая линия".

Если оператор использует предложение **Into Variable**, то созданный объект объявляется как значение объектной переменной. Если слово **Into** указывает окно, объект помещается на заданное место в окне (например, на изменяемый слой). Если **Into** вообще нет в операторе, MapBasic попытается создать объект в самом верхнем окне. Если это невозможно (например, поверх всех окон лежит окно Графика), то объект не будет создан.

Параметры *x* и *y* задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**. При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты. Если объект создается для окна Отчет, измерения производятся в объявленных ранее единицах измерения листа: X-координата – это расстояние от левого края листа до точки, и Y-координата – расстояние от верхнего края листа. По умолчанию, MapBasic использует дюймы. Чтобы задать другие "бумажные" единицы измерения, используйте **Оператор Set Paper Units**.

**Внимание:** Перед созданием объекта в окне Отчета не забудьте выполнить оператор Set CoordSys Layout.

Если явно задано **Предложение Pen**, то оно определяет стиль линии; см. подробности в разделе **Предложение Pen on page 492**. Если в операторе **Предложение Pen** не задано, оператор **Create Line** использует текущую установку стиля линии в MapInfo Professional.

**См. также:**

**Функция CreateLine( )**, **Оператор Insert**, **Предложение Pen**, **Оператор Update**

---

## Оператор Create Map

### Назначение

Изменяет структуру существующей таблицы, разрешая сопоставлять ее записям графические объекты.

### Синтаксис

```
Create Map
  For table
    [ CoordSys... ] Using from_table
```

*table* – имя открытой таблицы;

*from\_table* – имя открытой таблицы, из которой будет извлечена координатная система.

## Описание

Оператор **Create Map** присоединяет географические объекты к открытой таблице, после чего их можно видеть в окне Карты. Этот оператор не открывает новое окно Карты. Чтобы открыть новое окно Карты, используйте **Оператор Map**.

Не надо применять оператор **Create Map** к таблице, которая уже имеет присоединенные географические объекты; поступая так, Вы удаляете все географические объекты из таблицы. Если таблица уже имеет прикрепленные географические объекты, и Вам надо постоянно изменять проекцию карты, используйте оператор **Commit Table As**. С другой стороны, если надо временно изменить проекцию, в которой отображается карта, используйте **Оператор Set Map** и задайте в нем **Оператор CoordSys**. Оператор **Create Map** не работает со связанными таблицами. Чтобы присоединить к связанной таблице географические объекты, используйте **Оператор Server Create Map**.

## Определение системы координат

Используйте один из следующих методов для задания системы координат:

- Используйте имя уже открытой таблицы с географическими объектами как часть *from\_table* предложения **Using**. В этом случае, используемая система координат будет такой же, как и используемая в *from\_table*. Параметр *from\_table* должен задавать уже открытую таблицу, причем с географическими объектами, иначе появится сообщение об ошибке.
- Предложение **CoordSys** используется для явного задания координатной системы (это регулируется в настройках). Если Вы опустили и предложение **CoordSys**, и предложение **Using**, то таблица будет использовать координатную систему MapBasic.

Обратите внимание на то, что **Оператор CoordSys** может влиять на точность карты. Для этого **Оператор CoordSys** включает в себя предложение **Bounds**, которое устанавливает допустимые значения минимальных и максимальных координат, которые могут быть на карте. Если предложение **Bounds** пропущено, MapInfo Professional использует стандартные значения, охватывающие всю Землю (в этом случае, координаты имеют точность - миллионную часть градуса, или приблизительно 4 дюйма). Если у Вас имеется информация, что карта создана в ограниченном районе, то можно увеличить точность координат карты, задав ее границы. Подробно синтаксис **CoordSys** описан в разделе **Оператор CoordSys на стр. 33**.

См. также:

**Оператор Commit Table**, **Оператор CoordSys**, **Оператор Create Table**, **Оператор Drop Map**, **Оператор Map**, **Оператор Server Create Map**, **Оператор Set Map**

## Оператор Create Map3D

### Назначение

Создает 3D карту с определенными параметрами.

**Синтаксис****Create Map3D**

```
[ From Window window_id | MapString mapper_creation_string ]  
[ Camera [ Pitch angle | Roll angle | Yaw angle | Elevation angle ] |  
[ Position ( x, y, z ) | FocalPoint ( x, y, z ) ] |  
[ Orientation ( vu_1, vu_2, vu_3, vpn_1, vpn_2, vpn_3,  
    clip_near, clip_far )]]  
[ Light [ Position ( x, y, z ) | Color lightcolor ] ]  
[ Resolution ( res_x, res_y ) ]  
[ Scale grid_scale ]  
[ Background backgroundcolor ]  
[ Units unit_name ]
```

*window\_id* – идентификатор окна карты, содержащего слой поверхности; Если такой слой не найден, появится сообщение об ошибке.

*mapper\_creation\_string* - указывает командную строку, которая создает изображение по гриду.

**Camera** - определяет позицию и ориентацию камеры.

*angle* - это угол, измеряемый в градусах. Горизонтальный угол в диалоге может изменяться от 0 до 360 и вращает карту вокруг центральной точки грида. Вертикальный угол в диалоге изменяется от 0 до 90 и измеряет вращение в вертикальной плоскости прямо от стартовой точки прямо над картой.

**Pitch** регулирует поворот камеры вокруг оси X

**Roll** регулирует поворот камеры вокруг оси Z

**Yaw** регулирует поворот камеры вокруг оси Y

**Elevation** – определяет поворот камеры (находящейся в точке фокуса) относительно оси X;

**Position** - регулирует позицию камеры/источника света

**FocalPoint** – определяет положение точки наблюдения/точки фокуса;

**Orientation** – определяет ориентацию камеры: ViewUp (*vu\_1*, *vu\_2*, *vu\_3*), ViewPlane Normal (*vpn\_1*, *vpn\_2*, *vpn\_3*) и Clipping Range (*clip\_near*, *clip\_far*) (фиксированное отображение);

**Resolution** – количество ячеек поверхности (по X, Y); Эти значения могут увеличиваться вплоть до максимального разрешения грида. (максимального размера ячейки *x*, *y*). Если грид имеет разрешение 200x200 то и разрешение в окне карты не будет больше чем это значение 200x200. Вы не можете увеличивать разрешение грида, можно изменить только разрешение его изображения.

*grid\_scale* – определяет вертикальный масштаб (в направлении Z-координаты). Значение >1 подчеркивает рельеф, значение <1 снижает топологические особенности (в направлении Z-координаты).

*backgroundcolor* - это цвет, используемый для фона и он определяется функцией **Функция RGB( )**.

*unit\_name* – определяет единицы измерения точек поверхности. Не указывайте единицы, например, для температурного поля или плотности. Эту настройку надо делать к моменту создания грида. Вы не сможете изменить (настроить) их позже с использованием диалога, который показывает **Оператор Set Map3D**, или диалога "Свойства".

### Описание

Окно 3DКарты всегда показывается отдельно. Так как оно строится по тем же правилам, что и обычное окно Карты, то при изменении или обновлении базовых таблиц или Рабочих наборов, окно 3DКарты перерисовывается автоматически. Ошибка создания будет иметь место в том случае, если *window\_id* не окно Карты или карта не содержит слой поверхности. Если окно карты содержит несколько поверхностей, каждое будет представлено в окне 3D Карты.

3D Карта хранит информацию (строку) об окне карты для генерации текстуры. Эта строка будет также "преобладать" в Рабочем наборе, если окно 3D Карты присутствует в данном Рабочем наборе. При инициализации прочитываются данные грид-слоя и создаются трехмерные и топологические объекты.

### Пример:

```
Create Map3D Resolution(75,75)
```

Создает окно 3D карты для последнего окна Карты. Операция не будет выполнена, если окно Карты не содержит слоя поверхности. Другой пример:

```
Create Map3D From Window FrontWindow( ) Resolution(100,100) Scale 2  
Background RGB(255,0,0) Units "ft".
```

Создает окно 3DКарты с красным фоном, единицы по оси Z -футы, масштабный коэффициент - 2, и разрешение 100x100.

### См. также:

**Оператор Set Map3D**

## Оператор Create Menu

### Назначение

Создает новое меню или переопределяет уже существующее.

### Синтаксис 1

```
Create Menu newmenuname [ ID menu_id ] As  
  menuitem [ ID menu_item_id ] [ HelpMsg help ]  
  { Calling handler | As menuname }  
  [ , menuitem ... ]
```

### Синтаксис 2

```
Create Menu newmenuname As Default
```

*newmenuname* – заголовок нового меню или стандартное имя переопределяемого меню, тип String;

*menuitem* – имя элемента списка меню, тип String;

*menu\_id* – идентификатор стандартного элемента меню, число типа SmallInt ID от 1 до 15;

*menu\_item\_id* – целочисленный идентификатор созданного элемента меню;

*help* – текст, который будет показываться в строке сообщений при указании на элемент в списке меню;

*handler* – либо имя процедуры-обработчика, либо код стандартной команды MapInfo, либо строка специального синтаксиса для управления событиями в системе меню через механизмы OLE или DDE; см. **Режимы предложения Calling на стр. 42**. Если Вы задаете код команды меню MapInfo (такой как M\_WINDOW\_STATISTICS), то аргумент *menuitem* должен начинаться с восклицательного знака и содержать внутри символ каретки (^), чтобы удовлетворять синтаксису команд типа Показать/Скрыть.

*menuname* – заголовок меню, которое будет включено в список как подменю.

### Описание

Если параметр *newmenuname* является именем одного из существующих меню (например, **File**), то оператор переопределяет это меню. Если параметр *newmenuname* имеет какое-то уникальное значение, то оператор **Create Menu** создает новое меню. Список стандартных имен меню MapInfo приводится в разделе **Оператор Alter Menu on page 91**.

Оператор **Create Menu** не показывает новое или переопределяемое меню. Для вывода меню на экран используются **Оператор Alter Menu Bar** или **Оператор Create Menu Bar**. Однако, если оператор **Create Menu** переопределяет меню, которое существует в строке меню, то изменения меню отображаются немедленно.

**Внимание:** MapInfo Professional может одновременно поддерживать не более 96 определений меню, включая стандартные в (**File** и так далее). Число это не зависит от количества меню, показываемых в данный момент на экране.

Параметр *menuitem* определяет имя элемента меню. В нем могут использоваться специальные управляющие символы, определяющие состояние данного элемента меню (например, доступность команды в меню). Смотрите таблицу ниже.

Следующие знаки являются зарезервированными и выполняют специальную роль: прямой слэш (/), обратный слэш (\) и знак меньше (<). Если Вы хотите использовать эти знаки в тексте меню или строке подсказки, то необходимо поставить обратный слэш в строке *menuitem* или в строке *help*. Например, для помещения в меню "Данные" такой команды как "Клиент\Сервер", надо:

```
Create Menu "Данные" As"Клиент\Сервер" Calling cs_proc
```

Если текст параметра *menuitem* начинается с символа @, то пользовательское меню делится на две колонки. Элемент, начинающийся с @, помещается первым во вторую колонку.

### Обработчики для новых команд меню

Большинство элементов меню определяются с помощью предложения **Calling handler**, где *handler* либо имя процедуры на MapBasic, либо код стандартной команды MapInfo (например, код M\_FILE\_SAVE соответствует команде **Файл > Сохранить**). Если для элемента определен



обработчик и если пользователь выберет в меню этот элемент, то MapBasic автоматически вызовет процедуру-обработчик. Если для элемента меню определен код, такой как `M_FILE_SAVE`, то Ваша программа должна в начале иметь оператор `Include "MENU.DEF"`, подключающий файл стандартных определений для кодов команд.

С помощью необязательного предложения **ID** Вы можете задать элементу меню уникальный идентификатор. Вы можете связать один и тот же обработчик с разными элементами меню. В этом случае идентификатор может оказаться полезным для определения, какой именно командой была вызвана эта процедура. Определить идентификатор элемента меню из процедуры-обработчика, которую он вызвал, можно функцией `CommandInfo (CMD_INFO_MENUITEM)`. Идентификатором элемента меню пользуется также **Оператор Alter Menu Item**. Если за параметром *handler* не задано ни команды, ни подменю, ни вызова процедуры *menuname*, то соответствующий элемент меню будет неактивным. Такие элементы в меню выполняют косметическую роль (например, горизонтальная линия, разделяющая список меню на части).

Создание иерархически-подчиненного меню

Чтобы включить иерархическое меню в качестве нового элемента, задайте предложение **As** вместо **Calling**. Предложение **As** должно задавать один из существующих элементов меню. В следующем примере создается новое меню из одной "простой" пользовательской и одной команды и стандартной команды с иерархической структурой.

```
Create Menu "Данные" As "Настройки" Calling config_sub_proc, "Объекты" As "Объекты"
```

Если Вы добавляете иерархическое меню, то оно замещает элемент меню *menuitem*.

Свойства элементов меню

Команды в меню могут быть активны и отключены; последние показываются серым цветом. Можно добавлять галочку к имени элемента меню. Элемент меню с галочкой может быть активирован или деактивирован.

Следующие коды должны предшествовать имени элемента меню, поведение которого мы желаем сделать каким-то особенным:

| Код  | Эффект  |
|------|---|
| (    | Элемент меню недоступен для выбора. Пример: (Закрыть  |
| (–   | Горизонтальная разделительная линия; такой элемент не может иметь обработчик. Пример: (–  |
| ( \$ | Код для помещения в меню Файл имен четырех последних открывавшихся файлов. Его можно применять в системе меню один раз и его нельзя использовать в быстрых меню. Чтобы удалить список последних открывавшихся файлов из меню Файл, либо удалите этот код из файла MAPINFOW.MNU, либо переопределите меню Файл оператором <b>Create Menu</b> . |

| Код         | Эффект   |
|-------------|--|
| (>          | Код для представления в меню списка открытых окон. Его можно применять в системе меню один раз.  |
| !           | Элемент меню при выборе может снабжаться галочкой, но на момент создания галочка отсутствует.<br><br>Пример: !Показывать предупреждения  |
| ! ... ^ ... | Если в тексте появляется знак (^), то при выборе элемента он заменяется на другой текст (например, Показать... на Скрыть...) вместо того, чтобы отмечать галочкой один текст. Текст перед знаком "^" показывается, если элемент уже был выбран. Пример: !Скрыть строку сообщений^Показать строку сообщений |
| !+          | Элемент меню при выборе может снабжаться галочкой и на момент создания галочка присутствует.<br><br>Пример: !+Показывать предупреждения  |

### Задание клавишных сокращений

Элементам меню могут быть назначены клавишные сокращения (акселераторы), то есть пользователь может вызвать команду с клавиатуры без использования мыши. Клавишные сокращения бывают двух типов.

Клавишные сокращения первого типа имеют и имена меню, и имена элементов меню. Например, для того, чтобы открыть окно Карты, в рабочем окне MapInfo пользователь нажимает сначала клавишное сокращение ALT+O, которым открывается список команд меню Окно, и только потом клавишу K (или ALT+K) для запуска команды **Карта**. Для того, чтобы определить буквы для клавишного сокращения в меню и элементах меню, используйте знак амперсанда (&) в текстах параметров `newmenuname` и `menuitem` (например, текст "&Карта" в параметре `menuitem` оператора **Create Menu**). Амперсанд должен располагаться перед буквой, которая будет использоваться в клавишном сокращении.

Другой тип клавишных сокращений позволяет вызвать команду немедленно, не открывая меню, где она находится. Например, для вызова вышеупомянутой команды Карта достаточно только нажать клавишу F3. Управляющие коды, используемые для задания клавишного сокращения второго типа, приведены в следующей таблице.

**Внимание:** Управляющие коды, использованные для задания сочетаний клавиш, приведены в следующей таблице.

| Управляющий код       | Эффект  |
|-----------------------|---|
| /W {letter   %number} | Задаёт клавишное сокращение для Windows; команда активизируется клавишей letter или клавишей, заданной ее числовым кодом number.<br><br>Примеры: Данные/WD или Данные/W%196 |

| Управляющий код                     | Эффект  |
|-------------------------------------|---|
| <code>/W# {letter   %number}</code> | Задаёт клавишное сокращение для Windows; команда активизируется клавишей letter или клавишей, заданной ее числовым кодом number, с нажатой клавишей SHIFT.<br><br>Примеры: Данные <code>/W#Д</code> или Данные <code>/W#%196</code> |
| <code>/W@ {letter   %number}</code> | Задаёт клавишное сокращение для Windows; команда активизируется клавишей letter или клавишей, заданной ее числовым кодом number, с нажатой клавишей ALT.<br><br>Примеры: Данные <code>/W@Д</code> или Данные <code>/W@%196</code>   |
| <code>/W^ {letter   %number}</code> | Задаёт клавишное сокращение для Windows; команда активизируется клавишей letter или клавишей, заданной ее числовым кодом number, с нажатой клавишей CTRL.<br><br>Примеры: Зап <code>/W^Д</code> или Зап <code>/W^%196</code>        |

В Windows для использования клавиш Fn используется возможность задавать клавишные сокращения символом %, за которым следует численный код, номер клавиши. Так, для клавиши F1 номером является 112, для F2 – 113 и так далее.

**Внимание:** Заметим, что оператор `Create Menu Bar As Default` убирает и переопределяет все пользовательские меню, заданные оператором **Create Menu**. Чтобы выборочно отменить одно из созданных пользователем меню, можно воспользоваться оператором **Create Menu *menuname* As Default**.

Аналогично, чтобы выборочно отменить одно из измененных стандартных меню, можно воспользоваться тем же оператором **Create Menu *menuname* As Default**.

Режимы предложения Calling

Предложение **Calling** задает, что должно случиться, если пользователь в меню выполнит команду, построенную приложением. В таблице приведены возможные примеры использования этого предложения:

| Примеры                           | Описание   |
|-----------------------------------|--|
| <code>Calling M_FILE_NEW</code>   | Если за словом <b>Calling</b> идет целочисленный код из файла MENU.DEF, MapInfo запускает на выполнение соответствующую стандартную команду MapInfo (например, <b>File &gt; New</b> ). |
| <code>Calling my_procedure</code> | Если задано имя sub-процедуры, MapInfo передает управление этой процедуре.   |

| Примеры                      | Описание   |
|------------------------------|--|
| Calling OLE "methodname"     | Только для Windows. MapInfo управляет событиями, обращаясь с именем метода к объекту OLE Automation, установленному SetCallback-методом MapInfo. |
| Calling DDE "server","topic" | Только для Windows. MapInfo управляет событиями, присоединяясь через DDE к паре "сервер тема" и посылая выполняемое сообщение на DDE-сервер.     |

В последних двух случаях строка послания для OLE или DDE сервера должна начинаться с трех символов "MI:" для того, чтобы сервер смог определить, что послание пришло от MapInfo. Остальная часть этой строки состоит из разделенного запятыми списка значений, которые возвращает **CommandInfo( )** функция. Полное описание синтаксиса этой строки см. в *Руководстве пользователя MapBasic*.

## Примеры

В следующем примере оператор **Create Menu** создает новое меню и добавляет его в строку меню MapInfo Professional. Сначала удаляется меню "Окно" (ID 6) и меню "Справка" (ID 7), после чего добавляется новое меню, а затем восстанавливаются меню "Окно" и меню "Справка". Так мы добиваемся того, что заголовки Окно и Справка всегда будут последними в строке меню.

```
Declare Sub Main
Declare Sub addsub
Declare Sub editsub
Declare Sub delsub
Sub Main Create Menu "Данные" As "Добавить" Calling addsub, "Правка"
Calling editsub, "Удалить" Calling delsub Alter Menu Bar Remove ID 6, ID 7
Alter Menu Bar Add "Данные", ID 6, ID 7 End Sub
```

В этом фрагменте создается сокращенная версия меню Файл. Управляющий символ "(" делает команды "Закрыть", "Сохранить" и "Печатать" недоступными для выбора. Командам Открыть и Сохранить заданы сочетания клавиш второго типа (Ctrl+O и Ctrl+Y). А в тексте определения элемента меню используется знак табуляции – Chr\$(9) – для отделения сочетания клавиш вправо от текста.

```
Include "MENU.DEF"Create Menu "Файл" As"Новый" Calling M_FILE_NEW,
"Открыть" +Chr$(9)+"Ctrl+O/W^J" Calling M_FILE_OPEN,"(-","(Заккрыть"
Calling M_FILE_CLOSE, "(Сохранить" +Chr$(9)+"Ctrl+Ы /W^S" Calling
M_FILE_SAVE,"(-","(Печатать" Calling M_FILE_PRINT,"(-","(Выход" Calling
M_FILE_EXIT
```

Если не хотите, чтобы пользователь мог открывать быстрые меню, используйте оператор **Create Menu** и переопределите эти меню одним только разделителем "(-". В следующем примере быстрое меню окна Карты делается недоступным.

```
Create Menu "MapperShortcut" As "(-"
```

См. также:

**Оператор Alter Menu Item, Оператор Create Menu Bar**

## Оператор Create Menu Bar

### Назначение

Перестраивает строку заголовков меню, используя стандартные и ранее определенные меню.

### Синтаксис 1

```
Create Menu Bar As
{ menu_name | ID menu_number }
[ , { menu_name | ID menu_number } ... ]
```

### Синтаксис 2

**Create Menu Bar As Default**

*menu\_name* – заголовок стандартного для MapInfo меню или заголовок специально определенного меню, для создания которого ранее выполняется **Оператор Create Menu**.

*menu\_number* – номер стандартного меню (например, 1 для меню Файл).

### Описание

Оператор **Create Menu Bar** говорит MapInfo, какие меню должны быть помещены в строку меню и в каком порядке. Если оператор задает не полный список стандартных меню, то результатом будет строка меню с сокращенным списком MapInfo Professional. Если в список меню включено одно или несколько специально определенных имен меню (для их создания используется ), оператор **Create Menu Bar** создаст строку меню с расширенным списком.

Меню, как стандартное меню, так и специально определенное, может задаваться именем (например, "Файл"). Каждое стандартное меню также имеет номер (идентификатор), который может использоваться при задании. Например, меню "Файл" имеет идентификатор 1.

Список стандартных имен меню MapInfo Professional см. в разделе **Оператор Alter Menu Item on page 96**.

После того как система меню была изменена, Вы можете оператором

```
Create Menu Bar As Default
```

вернуть ее к стандартному виду. При этом оператор **Create Menu Bar As Default** удаляет все изменения в строке меню, включая и те, которые были созданы другими приложениями. Поэтому следует быть внимательным, употребляя этот оператор. Используйте команду **Create Menu Bar As Default** осторожно, чтобы случайно не удалить из меню пользовательские программы MapBasic.

### Примеры

Строка меню сокращается до четырех заголовков: "Файл", "Правка", "Анализ" и меню, соответствующее открытому окну ("Карта", "График" и т. п.).

```
Create Menu Bar As "Файл", "Правка", "Анализ", "WinSpecific"
```

В стандартной строке меню такие меню как Карта и Список не показываются, если окна Карты или Списка соответственно не являются активными. Следуя этой логике, MapInfo Professional показывает меню Browse, если активным окном является Список. Следующий оператор помещает эти меню в строку меню так, что они не зависят от того, есть ли на экране окна Карты и Списка. Но при этом, если пользователь откроет меню Карта, когда на экране нет активного окна Карты, то он увидит, что все команды этого меню недоступны (показаны серым шрифтом). Аналогично работает меню Список.

```
Create Menu Bar As "Файл", "Правка", "Запрос", "Карта", "Список"
```

В следующем фрагменте создается пользовательское меню "Данные", и помещается в строку меню MapInfo Professional.

```
Declare Sub AddSubDeclare Sub EditSubDeclare Sub DelSubCreate Menu  
"Данные" As"Добавить" calling AddSub,"Правка" calling EditSub,"Удалить"  
calling DelSubCreate Menu Bar As "Файл", "Правка", "Данные"
```

**См. также:**

[Оператор Alter Menu Bar](#), [Оператор Create Menu](#), [Оператор Menu Bar](#)

---

## Оператор Create MultiPoint

### Назначение

Объединяет множество точек в единый объект. Все точки обозначаются одним и тем же символом. Объект Multipoint отображается в окне Списка как одна запись.

### Синтаксис

#### Create Multipoint

```
[ Into { Window window_id | Variable var_name } ]  
[ num_points ]  
( x1, y1 ) ( x2, y2 ) [ ... ]  
[ Symbol... ]
```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*num\_points* – номер точки в объекте Multipoint;

$x$   $y$  – координаты точки.

Предложение **Symbol** определяет стиль символа.

**Внимание:** Для всех точек, содержащихся в объекте, используется один символ.

В настоящее время MapInfo Professional использует четыре варианта синтаксиса для определения символа, используемого для точек.

### Синтаксис 1 (для символов MapInfo 3.0)

**Symbol** ( *shape*, *color*, *size* ), где

*shape* – целое число, величина типа Integer, (значение 31 задает невидимый символ);  
Символы MapInfo 3.0 были представлены в MapInfo for Windows версии 3.0 и поддерживаются всеми последующими версиями MapInfo Professional. Для создания невидимого символа используйте значение 31. (значение 31 задает невидимый символ);

*color* – целочисленный код цвета в системе RGB, смотрите описание функции **Функция RGB( ) on page 536**;

*size* – целое число, величина типа Integer, размер символа в точках от 1 до 48;

### Синтаксис 2 (для шрифтов TrueType)

**Symbol** ( *shape*, *color*, *size*, *fontname*, *fontstyle*, *rotation* ), где

*shape* – целое, имеющее значение 31 или больше, определяющее, какой символ из шрифтов TrueType используется. Для создания невидимого символа используйте значение 31.

*color* – целочисленный RGB код цвета, смотрите описание функции RGB( );

*size* – целое число, величина типа Integer, размер символа в точках от 1 до 48;

*fontname* – строка, имя шрифта TrueType (например, "Wingdings");

*fontstyle* – целое, код, контролирующий атрибуты, например, курсив;

*rotation* – вещественное число, угол поворота в градусах.

### Синтаксис 3 (для пользовательских символов Custom Bitmap File)

**Symbol** ( *filename*, *color*, *size*, *customstyle* )

*filename* – строка до 31 символа длиной с именем растрового файла. Файл должен находиться в папке CUSTSYMB (если не применен **Оператор Reload Symbols** для указания другой папки).

*color* – целочисленный код цвета в системе RGB, смотрите описание функции **Функция RGB( ) on page 536**;

*size* – целое число, величина типа Integer, размер символа в точках от 1 до 48;

*customstyle* – целочисленный код типа Integer, управляющий цветом и фоном символа. См. таблицу ниже.

### Синтаксис 4

**Symbol** *symbol\_expr*, где

*symbol\_expr* – выражение для Symbol, которое может быть или именем переменной символа или функцией, которая возвращает значение символа, например, **Функция MakeSymbol( )**.

### Пример:

```
Create Multipoint 7 (0,0) (1,1) (2,2) (3,4) (-1,1) (3,-2) (4,3)
```



## Оператор Create Object

### Назначение

Создает один или несколько объектов типа "область", используя географические операции Buffer, Merge, Intersect, Union, Voronoi или Isogram.

### Синтаксис 1

```
Create Object As { Buffer | Union | Intersect | Merge | ConvexHull |
Voronoi }
  From fromtable
  [ Into { Table intotable | Variable varname } ]
  [ Data column = expression [ , column = expression... ] ]
  [ Group By { column | RowID } ]
```

### Синтаксис 2 (для буферов)

```
Create Object As Buffer
  From fromtable
  [ Into { Table intotable | Variable varname } ]
  [ Width bufferwidth [ Units unitname ] ]
  [ Type { Spherical | Cartesian } ] ]
  [ Resolution smoothness ]
  [ Data column = expression [ , column = expression... ] ]
  [ Group By { column | RowID } ]
```

### Синтаксис 3 (для изолиний)

```
Create Object As Isogram
  From fromtable
  [ Into { Table intotable } ]
  [ Data column = expression [ , column = expression... ] ]
  Connection connection_handle
  [ Distance dist1 [[ Brush brushclause ] [ Pen penclause ] ]
    [, dist2 [ Brush brushclause ] [ Pen penclause ] ]
    [, distN [ Brush brushclause ] [ Pen penclause ] [,...]
    Units dist_unit ]
  [ Time time1 [[ Brush brushclause ] [ Pen penclause ] ]
    [, time2 [ Brush brushclause ] [ Pen penclause ] ]
    [, timeN [ Brush brushclause ] [ Pen penclause ] [,...]
    Units time_unit ]
```

*fromtable* – имя открытой таблицы, содержащий один или более графических объектов.

*intotable* – имя открытой таблицы в которой будут создаваться новые объекты.

*varname* – имя новой объектной переменной;

*bufferwidth* – число, определяющее смещение при выполнении операции построения буферной зоны; если число отрицательное, а исходный объект – замкнутый объект, то полученная буферная зона будет меньше исходного объекта. Если задано отрицательное значение *width*, а объект является линейным (линией, полилинией или дугой) или же точкой, то используется положительное значение *width* и создается буфер больший чем объект.

*dist\_unit* - наименование единиц расстояния (например, “km” для километров).

*smoothness* - целое от 2 до 100, указывающее число сегментов для каждой зоны в операции Buffer.

*column* - имя колонки в таблице.

*expression* - выражение для расчета значений *column*.

*connection\_handle* - численное выражение возвращаемое от **Оператор Open Connection** ссылающееся на используемое соединение.

*dist1, dist2, distN* - численные значения представляющие расстояние для зон транспортной доступности рассчитываемых в *dist\_units*.

*brushclause* – это корректно заданное **Предложение Brush**.

*penclause* – это корректно заданное **Предложение Pen**.

*dist\_unit* - единицы расстояния. См. полный список возможных значений в разделе **Оператор Set Distance Units on page 633**.

*time1, time2, timeN* - численные значения представляющие время для зон транспортной доступности рассчитываемых в *time\_units*.

*time\_unit* - строка представляющая единицы времени. Допустимые значения: “hr”, “min”, или “sec”.

### Описание

Оператор **Create Object** создает одну или более областей на базе уже существующих объектов, используя географические операции (Buffer, Merge, Intersect, Union, ConvexHull, Voronoi или Isogram).

Предложение **Into** задает, куда будет помещен объект, полученный в результате. Для сохранения результатов в таблице, укажите **Into Table**. Для сохранения результатов в переменной Object, укажите **Into Variable**. Если пропущено предложение **Into**, результаты сохраняются в исходной таблице.

**Внимание:** Если используется предложение **Group By** чтобы объединить данные, то надо сохранить результаты в таблице, а не в переменной.

Ключевое слово, которое следует за ключевым словом **As**, определяет, какой тип объектов будет создан. **ConvexHull**, **Buffer** и **Isogram** обсуждаются в отдельных разделах ниже.

В режиме **Intersect** создается объект, представляющий собой пересечение других объектов (например, если два региона пересекаются, то пересечение - это область, покрывающая сразу эти два объекта).

В режиме **Merge** создается объект, представляющий собой комбинированную площадь из объектов-источников. Операция Merge создает результирующий объект, который содержит все полигоны, относящиеся к исходным объектам. Если исходные объекты перекрываются, то операция слияния (merge) не удаляет перекрытия. Таким образом, если объединяются два перекрывающихся региона (каждый из которых содержит один полигон), то окончательный результат будет в виде одного региона, состоящего из двух перекрывающихся полигонов. Чтобы избежать этого, можно использовать объединение в режиме **Union**.

В режиме **Union** комбинирование удаляет любые области перекрытия. Если осуществляется операция объединения (union) двух перекрывающихся регионов (каждый из которых содержит один полигон), то результат будет в виде объекта-региона, содержащего один полигон.

Операции объединения (union) и слияния (merge) похожи, но их поведение сильно различается в том случае, если объекты полностью содержат в себе другие объекты. В этом случае, операция слияния (merge) удаляет область малого объекта из большого объекта, оставляя дыру там, где был малый объект. Операция объединения (union) не будет удалять область меньшего объекта.

**Create Objects As Union** похож на оператор **Оператор Objects Combine** тем, что **Оператор Objects Combine** удаляет исходные объекты и вставляет новый комбинированный объект. **Create Objects As Union** только добавляет объекты в группу, но не удаляет исходные объекты. Комбинирование с использованием изменяемого объекта и различных таблиц возможно только, применяя **Оператор Objects Combine**. Оператор Combine Objects, использующий функциональность Column, допустим только при использовании **Create Objects As Union** вместе с предложением **Group By**.

Если оператор **Create Object As Union** не включает в себя предложение **Group By**, MapInfo Professional создает один комбинированный объект для всех объектов в таблице. Если этот оператор включает в себя предложение **Group By**, то должно быть задано имя колонки в таблице, что позволит MapInfo Professional сгруппировать исходные объекты в соответствии с содержанием колонки и создать комбинированный объект для каждой группы объектов.

Если Вы определяете предложение **Group By**, MapInfo Professional сгруппирует все записи, имеющие одинаковые значения, и осуществит операцию (слияния) в группу.

Если Вы определяете предложение **Data**, MapInfo Professional осуществит объединение данных. Например, если осуществляется слияние или объединение, может возникнуть необходимость использовать предложение **Data**, чтобы присвоить значение данным, выполняя функции объединения Sum( ) или Avg( ).

Предложение **Voronoi** используется для создания регионов, представляющих полигоны Вороного для исходных точек. Значения данных их исходных точек могут быть присвоены результирующему полигону, для этого в предложении надо указать источник таких точек.

## Режим Convex Hull

В режиме **ConvexHull** оператор будет создавать полигон, представляющий оконтуривающий объект вокруг набора точек. Такой полигон можно получить, если натянуть резинку, охватывающую все узлы. Он составлен из минимального количества точек (т.е. точки входного объекта лежат на границах или внутри полигона). Полигон получается выпуклый - то есть все внешние углы созданного полигона-контура больше, чем 180 градусов.

Точки, используемые для оконтуривающего региона, могут быть любыми узлами объектов типа Region, Polyline или Point таблицы *fromtable*. Если оператор **Create Object As ConvexHull** не включает в себя предложение **Group By**, MapInfo Professional создаст один оконтуривающий полигон. Если оператор включает в себя предложение **Group By**, указывающее имя колонки в таблице, то MapInfo Professional сгруппирует исходные объекты исходя из содержания указанной колонки, затем создаст оконтуривающий полигон для каждой группы объектов. Если оператор включает в себя предложение **Group By RowID**, MapInfo Professional создаст один оконтуривающий полигон для каждого объекта в исходной таблице.

### Режим Buffer

Если оператор **Create Object** работает в режиме Buffer, то он может включать в себя предложения **Width** и **Resolution**. Предложение **Width** определяет ширину буферной зоны. Добавочное подпредложение **Units** позволяет указать имя единиц измерения расстояния (например, "km" для километров) относящихся к предложению **Width**. Если предложение **Width** не включает в себя подпредложение **Units**, то ширина буферной зоны будет интерпретироваться в текущих единицах измерения MapBasic. По умолчанию, MapBasic измеряет расстояния в милях; для того чтобы задать другую единицу измерения, используйте **Оператор Set Distance Units**.

**Type** позволяет указать тип алгоритма вычисления расстояний при создании буферной зоны. Он может быть либо сферическим (**Spherical**), либо декартовым (**Cartesian**). Если координатная система таблицы *intotable* - немировая, то вычисления производятся декартовыми методами вне зависимости от настроек оператора; если же координатная система *intotable* - это Долгота/Широта, то вычисления производятся сферическими методами вне зависимости от настроек оператора.

Подпредложение **Type** позволяет определять способ вычислений расстояний во время создания буфера. Если используется тип **Spherical**, то все вычисления производятся с учетом шарообразности поверхности Земли, то есть в координатах Долгота/Широта. Если используется **Cartesian** (декартовый) тип вычислений, то подразумевается, что вычисления производятся на карте, на которую нанесены объекты на плоскости (т.е. на плане). Если предложение **Width** не включает подпредложения **Type**, то по умолчанию используется тип **Spherical**. Если данные представлены в проекции Долгота/Широта, то используются сферические вычисления независимо от установок в предложении **Type**. Если данные в проекции план-схема, то, независимо от установки параметра **Type**, будут использованы декартовы вычисления.

Параметр **smoothness** позволяет определить число сегментов, из которых состоит каждая окружность в составе буферной зоны. Стандартное значение *smoothness* равно 12; это означает, что при рисовании буфера из простой окружности будет использовано 12 сегментов. Если задать большую величину *smoothness*, то получится более сглаженная буферная зона. Обратите внимание, что чем больше значение *smoothness*, тем дольше работает оператор **Create Object** и тем больше дискового пространства занимает результат.

Если оператор **Create Object As Buffer** не включает в себя предложение **Group By**, MapInfo Professional создаст один буферный регион. Если оператор включает в себя предложение **Group By**, которое является именем колонки в таблице, MapInfo Professional сгруппирует исходные объекты в соответствии с содержанием колонки, затем создаст по одному

буферному региону для каждой группы объектов. Если оператор включает в себя предложение **Group By RowID**, MapInfo Professional создаст по одному буферному региону для каждого объекта в исходной таблице.

## Изограмма

Если оператор **Create Object** используется для создания изолиний, необходимо передать *connection\_handle* соответствующий открытому соединению созданному с использованием **Оператор Open Connection**. Вы должны указать предложение **Distance** или **Time** чтобы указать требуемый размер зоны доступности. Предложение **Distance** может включать одно или несколько выражений для расстояния (зон), с возможностью указать стиль оформления для каждой зоны. Если не задано **Предложение Brush** или **Предложение Pen**, используются текущие установки штриховки или линии. Неважно сколько зон транспортной доступности по расстоянию будет создаваться, строка **Units** должна содержать единицы расстояния которые будут использоваться в вычислениях.

Задав предложение **Time**, Вы можете создавать области на основе отсчета времени, и для каждой задать **Предложение Brush** и/или **Предложение Pen**. Если не задано **Предложение Brush** или **Предложение Pen**, используются текущие установки штриховки или линии. Неважно сколько зон транспортной доступности по времени будет создаваться, строка **Units** должна содержать единицы времени которые будут использоваться в вычислениях. Максимальное допустимое значение 50. Каждое значение создаёт отдельную зону, которая может отображать транспортную доступность либо по времени, либо по расстоянию. Чем больше значение, тем больше время портебуется на операцию создания зон. Много факторов влияют на скорость обработки, но в целом использование **Оператор Set Connection Isogram** с параметром **MajorRoadsOnly**, значительно быстрее, чем использование всей транспортной сети. MapBasic допускает расстояние 150 миль для параметра **MajorRoadsOnly Off** и 600 миль для **MajorRoadsOnly On**. аналогично, максимальное время - 2 часа для параметра **MajorRoadsOnly Off** и 8 часов для параметра **MajorRoadsOnly On**.

## Пример:

Следующий пример показывает слияние объектов-регионов из таблицы PARCELS и хранит результирующие регионы в таблице ZONES. Так как оператор **Create Object** включает в себя предложение **Group By**, MapBasic группирует регионы таблицы PARCELS, а затем произведет слияние для каждой группы. Таким образом, таблица ZONES будет иметь по одному объекту-региону для каждой группы объектов в таблице PARCELS. Каждая группа будет состоять из тех частей, которые имеют одинаковые значения в колонке zone\_id

Оператор **Create Object** накапливает в колонке parcelcount число однородных объектов таблицы ZONES и показывает, сколько элементов сливается вместе, образуя единую зону. Колонка zonevalue в таблице ZONES будет показывать сумму элементов, которые включает в себя эта зона.

```
Open Table "PARCELS"
Open Table "ZONES"
Create Object As Merge
  From PARCELS Into Table ZONES Data
  parcelcount=Count(*), zonevalue=Sum(parcelvalue)
  Group By zone_id
```

Следующий пример создает объект-регион, представляющий зону шириной в четверть мили вокруг выбранных объектов. Объект-буфер хранится в объектной переменной corridor. Применив далее **Оператор Update** или **Оператор Insert**, можно скопировать объект в таблицу.

```
Dim corridor As Object
Create Object As Buffer
  From Selection
  Into Variable corridor
  Width 0.25 Units "mi"
  Resolution 60
```

Следующий объект показывает оконтуривающий полигон, состоящий из многих объектов, созданный оператором **Create Object As**.

```
Create Object As ConvexHull from state_caps into Table dump_table
```

**См. также:**

**Функция Buffer( ), Функция ConvexHull( ), Оператор Objects Combine, Оператор Objects Erase, Оператор Objects Intersect, Оператор Open Connection**

## Оператор Create Pline

### Назначение

Создает объект типа "полилиния".

### Синтаксис

#### Create Pline

```
[ Into { Window window_id | Variable var_name } ]
[ Multiple num_sections ]
num_points ( x1, y1 ) ( x2, y2 ) [ ... ]
[ Pen... ]
[ Smooth ]
```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*num\_points* – определяет число узлов для полилинии;

*num\_sections* – задает, из скольких частей может состоять сегментированная (multi-section) полилиния;

Каждая пара *x*, *y* задает координаты для одного узла полилинии.

Слово **Pen** начинает стандартное предложение для назначения стиля линии.

### Описание

Результатом действия оператора **Create Pline** является новый объект типа "полилиния". Если Вы хотите создать полилинию, но не знаете, сколько в ней будет узлов, поступайте так: сначала оператором **Create Pline** создайте объект вообще без узлов, а затем, выполнив в подходящий момент **Оператор Alter Object**, добавьте к нему нужно количество узлов. Более подробную информацию см. в разделе **Оператор Alter Object on page 99**.

Если оператор использует предложение **Into Variable**, то созданный объект объявляется как значение объектной переменной. Если слово **Into** указывает окно **Window**, объект помещается на заданное место в окне (например, в изменяемый слой). Если предложения **Into** вообще нет в операторе, MapBasic попытается создать прямоугольник в самом верхнем окне; если это невозможно (например, поверх всех окон лежит окно Графика), то объект не будет создан.

Параметры *x* и *y* являются координатами в текущей системе координат MapBasic (по умолчанию это Широта/Долгота; см. также раздел **Оператор Set CoordSys on page 629**). Если объект создается для окна Отчета, параметры *x* и *y* интерпретируются как координаты на листе в объявленных ранее "бумажных" единицах измерения листа. По умолчанию, MapBasic использует дюймы в качестве "бумажных" единиц расстояния. Чтобы задать другие "бумажные" единицы измерения, используйте **Оператор Set Paper Units**. Перед созданием объекта в окне Отчета не забудьте выполнить оператор **Set CoordSys Layout**.

Если явно задано **Предложение Pen**, то оно определяет стиль линии; см. подробности в разделе **Предложение Pen on page 492**. Если не задан параметр Pen, оператор **Create Pline** использует текущий стиль линии (то есть, тот, который задан в диалоге MapInfo Professional **Настройки > Стиль линии**). **Smooth** сгладит линию так, что она будет выглядеть без узлов.

Однофрагментный объект типа "полилиния" может содержать до 32 763 узлов. Для многофрагментных объектов лимит узлов меньше: на каждую ломаную линию надо убавлять по три узла.

**См. также:**

**Оператор Alter Object, Оператор Insert, Предложение Pen, Оператор Update**

---

## Функция CreatePoint( )

### Назначение

Возвращает объект "точка".

### Синтаксис

**CreatePoint( x, y )**

x – X-координата точки (или широта), вещественное число;

y – Y-координата точки (или долгота), вещественное число.

### Возвращаемая величина

Объект

### Описание

Функция **CreatePoint( )** возвращает графический объект типа "точка".

Параметры x и y задают координаты центра окружности в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**. При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты.

Для рисования точечного объекта используется текущий стиль символа. Точечный объект будет создаваться в соответствии с установкой стиля символа для точечных объектов, для чего нужно выполнить **Оператор Set Style** до функции CreatePoint( ). Вы можете также вместо создания объекта функцией **CreatePoint( )**, использовать **Оператор Create Point**, задав в нем предложение **Symbol** для определения стиля символа точки.

Графический объект, созданный функцией **CreatePoint( )**, может быть присвоен переменной объекта, которая определяет значение в существующей строке таблицы (**Оператор Update**) или вновь созданной (**Оператор Insert**).

**Внимание:** Перед созданием объекта в окне Отчета не забудьте выполнить **Оператор Set CoordSys**.



## Примеры

В примере используется **Оператор Insert** для создания новой строки в таблице SITES. Функция **CreatePoint( )** вложена в оператор Insert, для создания объекта типа "точка", данные которого будут помещены в этой строке.

```
Open Table "sites"
Insert Into sites (obj)
  Values ( CreatePoint(-72.5, 42.4) )
```

В следующем примере производится обращение к таблице SITES, которая имеет колонки Xcoord и Ycoord, представляющие долготу и широту объектов. **Оператор Update** использует функцию **CreatePoint( )** для сопоставления точечного объекта каждой строке таблицы. Следующий оператор Update присваивает каждой записи в таблице STATES точечный объект. Каждый точечный объект располагается в зависимости от значений в колонках Xcoord и Ycoord.

```
Open Table "sites"
Update sites
  Set obj = CreatePoint(xcoord, ycoord)
```

В приведенном выше примере подразумевается, что колонки "Xcoord" и "Ycoord" содержат значения широты и долготы. Файлы точек MapInfo для DOS содержат координаты в миллионных долях градуса, а не в целых градусах. Кроме того, в большинстве файлов точек MapInfo для DOS принято направление отсчета долгот с Востока на Запад. Поэтому, чтобы оператор Update мог корректно преобразовать файл точек из DOS в Windows, нужно разделить обе координаты на миллион и умножить "Xcoord" на минус единицу:

```
Open Table "sites"
Update sites
  Set obj = CreatePoint(-xcoord/1000000,ycoord/1000000)
```

**См. также:**

**Оператор Create Point, Оператор Insert, Оператор Update**

## Оператор Create Point

### Назначение

Создает объект типа "точка".

### Синтаксис

```
Create Point
  [ Into { Window window_id | Variable var_name } ]
  ( x, y )
  [ Symbol... ]
```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*x, y* – координаты точки.

Предложение **Symbol** определяет стиль символа.

### Описание

Оператор **Create Point** создает новый точечный объект.

Если оператор использует предложение **Into Variable**, то созданный объект объявляется как значение объектной переменной. Если слово **Into** указывает окно, объект помещается на заданное место в окне (например, на изменяемый слой). Если **Into** вообще нет в операторе, MapBasic попытается создать точку в самом верхнем окне; если это невозможно (например, если поверх всех окон лежит окно Графика), то объект не будет создан.

Параметры *x* и *y* задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**. При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты. Если объект создается для окна Отчет, измерения производятся в объявленных ранее единицах измерения листа: X-координата – это расстояние от левого края листа до точки, и Y-координата – расстояние от верхнего края листа. По умолчанию, MapBasic использует дюймы. Чтобы задать другие "бумажные" единицы измерения, используйте **Оператор Set Paper Units**.

**Внимание:** Перед созданием объекта в окне Отчета не забудьте выполнить оператор **Set CoordSys Layout**.

Необязательное **Предложение Symbol** задает стиль символа; см. описание в разделе **Предложение Symbol on page 725**. Если в операторе это **Предложение Symbol** не участвует, оператор **Create Point** использует установку соответствующего режима для стиля линии в MapInfo (стиль линии можно изменить командой **Настройка > Стиль символов**).

**См. также:**

**Функция CreatePoint( ), Оператор Insert, Предложение Symbol, Оператор Update**

---

## Оператор Create PrismMap

### Назначение

Создает Карту-призму.

### Синтаксис

#### Create PrismMap

```
[ From Window window_ID | MapString mapper_creation_string ]
  { layer_id | layer_name }
With expr
[ Camera [Pitch angle | Roll angle | Yaw angle | Elevation angle] |
  [ Position ( x, y, z ) | FocalPoint ( x, y, z ) ] |
  [ Orientation(vu_1, vu_2, vu_3, vpn_1, vpn_2, vpn_3,
    clip_near, clip_far) ] ]
  [ Light Color lightcolor ] ]
```

```
[ Scale grid_scale ]
[ Background backgroundcolor ]
```

*window\_id* – идентификатор окна для окна Карты, которое содержит слой полигонов. Если слой с полигонами не найден, появится сообщение об ошибке.

*mapper\_creation\_string* – определяет командную строку, которая создает текстуру для Карты-призмы;

*layer\_id* – идентификатор слоя Карты;

*layer\_name* – это имя слоя Карты;

*expr* – выражение, которое вычисляется для каждой записи таблицы;

**Camera** - определяет позицию и ориентацию камеры.

*angle* - это угол, измеряемый в градусах. Горизонтальный угол в диалоге может изменяться от 0 до 360 и вращает карту вокруг центральной точки грида. Вертикальный угол в диалоге изменяется от 0 до 90 и измеряет вращение в вертикальной плоскости прямо от стартовой точки прямо над картой.

**Pitch** – угловая величина определяет текущее положение точки наблюдения по оси X;

**Roll** – угловая величина определяет текущее положение точки наблюдения по оси Z;

**Yaw** – угловая величина определяет текущее положение точки наблюдения по оси Y;

**Elevation** – определяет поворот камеры (находящейся в точке фокуса) относительно оси X;

**Position** – определяет позицию камеры и/или источника освещения;

**FocalPoint** – определяет фокальную точку камеры и/или источника освещения;

**Orientation** – определяет для камеры значение параметров ViewUp (*vu\_1*, *vu\_2*, *vu\_3*), ViewPlane Normal (*vpn\_1*, *vpn\_2*, *vpn\_3*) и Clipping Range (*clip\_near* и *clip\_far*), используется для компенсации инерции зрительного восприятия).

*grid\_scale* – определяет вертикальный масштаб (в направлении Z-координаты). Значение >1 подчеркивает рельеф, значение <1 снижает топологические особенности (в направлении Z-координаты).

*backgroundcolor* - это цвет, используемый для фона и он определяется функцией **Функция RGB()**.

## Описание

Оператор **Create PrismMap** создает окно Карты-призмы. С помощью карты-призмы можно отображать несколько переменных для одного объекта. Например, цвет, ассоциированный с полигоном, может иметь тематическое значение одной колонки, а высота призмы может отражать значение другой колонки. Оператор **Create PrismMap** соответствует команде MapInfo Professional **Карта > Создать Карту-призму**.

Между сеансами работы MapInfo Professional сохраняет настройки карты-призмы, добавляя оператор **Create PrismMap** в файл Рабочего набора. Таким образом, чтобы увидеть пример оператора **Create PrismMap**, создайте карту, выполните команду **Карта > Создать тематическую карту**, сохраните Рабочий набор (например, PRISM.WOR), и проверьте Рабочий набор в окне MapBasic. После этого скопируйте оператор **Create PrismMap** в Вашу

программу MapBasic. Аналогично можно увидеть примеры операторов **Create PrismMap** при открытии окна MapBasic перед выполнением команды **Карта > Создать тематическую Карту**.

Каждый оператор **Create PrismMap** должен определять предложение для выражения *expr*. MapInfo оценивает это выражение для каждого объекта в таблице, из которой строятся призмы; обрабатывая действия оператора **Create PrismMap**, MapInfo выбирает стиль отображения каждого объекта, основываясь на значении записи *expr*. Выражения обычно содержат имена одной или более колонок из выбранной таблицы.

Дополнительное предложение *window\_id* определяет, какая карта послужит основой Карты-призмы; если предложение *window\_id* отсутствует, MapBasic создаст призмы для самого верхнего окна Карты. Оператор **Create PrismMap** указывает, какой слой надо использовать, даже если окно Карты имеет только один слой. Слой может быть идентифицирован по номеру (*layer\_id*), где самый верхний слой карты имеет *layer\_id* равный 1, следующий слой имеет *layer\_id* равный 2, и т.д. Или же оператор **Create PrismMap** может идентифицировать слой карты по имени (например, "world").

### Пример:

```
Open Table "STATES.TAB" Interactive
Map From STATES
Create PrismMap From Window FrontWindow( ) STATES With Pop_1980 Background
RGB(192,192,192)
```

### См. также:

[Оператор Set PrismMap](#), [Функция PrismMapInfo\( \)](#)

---

## Оператор Create Ranges

### Назначение

Вычисляет значения диапазонов для условного (тематического) выделения методом выделения диапазонов и помещает объекты в массив переменных, который может впоследствии использовать [Оператор Shade](#).

### Синтаксис

#### Create Ranges

```
From table
With expr
[ Use {"Equal Ranges" | "Equal Count" | "Natural Break" | "StdDev" } ]
[ Quantile Using q_expr ]
[ Number num_ranges ]
[ Round rounding_factor ]
Into Variable array_variable
```

*table* – имя открытой таблицы, объекты которой будут участвовать в тематическом выделении;

*expr* – выражение, которое вычисляется для каждой записи таблицы;

*q\_expr* – выражение, используемое при квантовании;

*num\_ranges* – задает число диапазонов (по умолчанию 4);

*rounding\_factor* – делитель, по которому округляются диапазоны при разделении (например, значение 10 задает округление до ближайшего десятка);

*array\_variable* – массив численных переменных типа Float, в который будут помещены значения для диапазонов.

## Описание

Оператор **Create Ranges** вычисляет значения диапазонов, которые впоследствии использует **Оператор Shade** для создания тематического слоя на Карте методом выделения диапазонов. Подробную информацию о тематических Картах смотрите в документации по MapInfo Professional.

Предложение **Use** задает метод, каким будут разделены данные на диапазоны. Задав "Equal Ranges", Вы разделите диапазоны, исходя из разброса значений (например, 0-25, 25-50, 50-75, 75-100). Задав "Equal Count", Вы разделите диапазоны, исходя из количества записей в таблице, т.е. в диапазоны попадет примерно одинаковое количество записей. Задав "Natural Break", Вы разделите диапазоны, исходя из естественно близких групп значений. Задав "StdDev", Вы сначала делите диапазоны по среднему значению, а затем добавляете один диапазон со значениями выше среднего, но не далее величины дисперсии от среднего, а также еще один диапазон со значениями ниже среднего, но не далее величины дисперсии от среднего. MapInfo Professional использует стандартное отклонение ( $N - 1$ ).

Предложение **Into Variable** определяет имя массива переменных типа Float, в который будет помещен результат. Не нужно следить за размерами этого массива; MapInfo автоматически изменяет его размер. Размер этого массива будет всегда в два раза больше количества диапазонов, так как в нем помещаются как нижняя, так и верхняя граница диапазона.

После выполнения оператора **Create Ranges**, выполните **Оператор Shade**, чтобы создать тематическую карту; включите дополнительный параметр **From Variable** в предложение Shade, с помощью которого можно прочесть массив границ диапазонов. Обычно оператор Shade использует ту же таблицу и то же выражение для колонки, что и оператор **Create Ranges**.

## Квантование диапазонов

Предложение **Quantile Using** отключает предложение **Use** и назначает разделение диапазонов квантованием, которое задается выражением *q\_expr*.

Квантование лучше всего проиллюстрировать следующим примером. Оператор вычисляет границы диапазонов покупательной способности населения США (BPI), квантуя их по значениям населения штатов.

```
Create Ranges From states With BPI_1990 Quantile Using Pop_1990 Number 5
Into Variable f_ranges
```

В этом примере создается пять диапазонов (Number 5).

Штаты с высокой покупательной способностью населения (With BPI\_1990) помещаются в "высшие" диапазоны (темные оттенки цвета), а с низкой – в "низшие" (светлые оттенки цвета).

Так как предложение `Quantile Using Pop_1990` использует колонку (`Pop_1990`), то при вычислении границ промежуточных диапазонов используется метод квантования по значениям населения штата. Границы диапазонов устанавливаются следующим образом: так как предложение **Quantile Using** задает колонку "`Pop_1990`", то MapInfo Professional сначала вычисляет общее количество населения США (около 250 миллионов). Затем MapInfo Professional делит результат на количество диапазонов (в нашем случае, 5) и получается пятьдесят миллионов. После этого MapInfo Professional делит диапазоны так, чтобы суммарное население штатов, охватываемых диапазоном, приближалось (но не превосходило) пятьдесят миллионов.

MapInfo Professional собирает штаты по порядку возрастания покупательной способности BPI, и в первый, "низший" диапазон попадают штаты с самой низкой покупательной способностью. MapInfo Professional продолжает добавлять штаты в первый диапазон до тех пор, пока суммарное значение численности населения не достигнет или не превысит пятидесяти миллионов. В этот момент MapInfo решает, что первый диапазон готов и приступает к обсчету следующего. Во второй диапазон MapInfo Professional помещает записи о штатах, пока чтобы суммарное значение численности населения не достигнет или не превысит сто миллионов; это означает, что второй диапазон заполнен; и т.д.

### Пример:

```
Include "mapbasic.def" Dim range_limits( ) As Float, brush_styles( ) As
Brush Dim col_name As Alias Open Table "states" Interactive Create Styles
From Brush(2, CYAN, 0) 'стиль для "низшего" диапазона To Brush(2, BLUE,
0) 'стиль для "высшего" диапазона Vary Color By "RGB" Number 5 Into
Variable brush_styles
' Присвоим имя колонки переменной типа Alias: col_name = "Pop_1990" Create
Ranges From states With col_name Use "Natural Break" Number 5 Into
Variable range_limits Map From states Shade states With col_name Ranges
From Variable range_limits Style Variable brush_styles ' Вывод окна
Легенды: Open Window Legend
```

### См. также:

[Оператор Create Styles](#), [Оператор Set Shade](#), [Оператор Shade](#)

## Оператор Create Rect

### Назначение

Создает объект "прямоугольник" или "квадрат".

### Синтаксис

#### Create Rect

```
[ Into { Window window_id | Variable var_name } ]
( x1, y1 ) ( x2, y2 )
[ Pen... ]
[ Brush... ]
```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*x1, y1* – координаты начального угла прямоугольника;

*x2, y2* – координаты противоположного по диагонали угла прямоугольника;

Слово **Предложение Pen** начинает стандартное предложение для назначения стиля линии объекта.

Предложение **Предложение Brush** определяет стиль штриховки.

### Описание

Если в операторе **Create Rect** присутствует предложение **Into Variable**, созданный объект объявляется значением объектной переменной *var\_name*. Если слово **Into** указывает окно **Window**, объект помещается на заданное место в окне (например, в изменяемый слой). Если **Into** вообще нет в операторе, MapBasic попытается создать объект в самом верхнем окне. Если это невозможно (например, поверх всех окон лежит окно Графика), то объект не будет создан.

Параметры *x* и *y* задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**. При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты. Если объект создается для окна Отчет, измерения производятся в объявленных ранее единицах измерения листа: X-координата – это расстояние от левого края листа до точки, и Y-координата – расстояние от верхнего края листа. По умолчанию, MapBasic использует дюймы. Чтобы сменить единицы измерения, выполните **Оператор Set Paper Units**.

**Внимание:** Перед созданием объекта в окне Отчета не забудьте выполнить оператор Set CoordSys Layout.

Если явно задано **Предложение Pen**, то оно определяет стиль линии; см. подробности в разделе **Предложение Pen on page 492**. Если в операторе не участвует **Предложение Pen**, оператор **Create Rect statement** использует текущий стиль линии в MapInfo (который задан в диалоге команды **Настройка > Стиль линий**). Аналогично, предложение Brush назначает стиль штриховки; подробнее см. раздел **Предложение Brush on page 117**.

См. также:

Предложение [Brush](#), Оператор [Create RoundRect](#), Оператор [Insert](#), Предложение [Pen](#), Оператор [Update](#)

---

## Оператор Create Redistricter

### Назначение

Открывает окно Районирование.

### Синтаксис

```
Create Redistricter source_table By district_column
  With
  [ Count ]
  [ , Brush ] [ , Symbol ] [ , Pen ]
  [ , { Sum | Percent } ( expr ) ] [ , { Sum | Percent } ( expr ) ... ]
  [ Percentage From expr]
  [ Order { "MRU" | "Alpha" | "Unordered" } ]
```

*source\_table* – имя открытой таблицы, объекты которой будут участвовать в районировании;

*district\_column* – имя колонки; начальный набор районов создается на базе значений из этой колонки и в нее же помещается новая структура районов;

Слово **Count** предопределяет показ в Списке Районов колонки с количеством объектов в каждой группе;

Слово **Brush** назначает показ колонки с образцами штриховки объектов;

Слово **Symbol** назначает показ колонки с образцами символов;

Слово **Pen** назначает показ колонки с образцами линий объектов;

*expr* – числовое выражение;

Предложение **Percentage From** задает режим вычисления процентов в строках.

Предложение **Order** задает порядок строк в Списке Районов (по алфавиту, произвольно или же все затронутые изменения строки помещаются в начало списка; последний режим (MRU) используется по умолчанию).

### Описание

Оператор **Create Redistricter** начинает сеанс районирования. Этот оператор соответствует команде MapInfo Professional **Окно > Районирование**. Правила работы с районами подробно описаны в документации MapInfo.

Для управления составом районов используется [Оператор Set Redistricter](#). Закончить сеанс районирования можно, выполнив [Оператор Close Window](#), закрывающий окно Районирование.



Если включить слово **Brush**, то в окно Районирование будет добавлена колонка с образцами штриховок каждого района. Обратите внимание на то, что это не **Предложение Brush**, а отдельное ключевое слово **Brush**. Аналогично, **Symbol** и **Pen** - это отдельные ключевые слова, а не **Предложение Symbol** или **Предложение Pen**. Если в Списке Районов есть колонки с образцами оформления районов, то пользователь может их изменять, указывая на них мышкой.

Если задано предложение **Percentage From**, то проценты рассчитываются построчным методом. Если предложение **Percentage From** не задано, то проценты рассчитываются по колонкам.

См. также:

**Оператор Set Redistrict**

## Оператор Create Region

### Назначение

Создает объект типа "область".

### Синтаксис

#### Create Region

```
[ Into { Window window_id | Variable var_name } ]
  num_polygons
[ num_points1 ( x1, y1 ) ( x2, y2 ) [ ... ] ]
[ num_points2 ( x1, y1 ) ( x2, y2 ) [ ... ] ... ]
[ Pen... ]
[ Brush... ]
[ Center ( center_x, center_y ) ]
```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*num\_polygons* – число полигонов в области (ноль или более);

*num\_points1* – число узлов в первом полигоне;

*num\_points2* – число узлов во втором полигоне, и т. д.;

Каждая пара *x*, *y* задает координаты узла полигона.

Слово **Pen** начинает стандартное предложение для назначения стиля линии.

Слово **Brush** начинает стандартное предложение для назначения стиля штриховки.

*center\_x* – X-координата центроида области;

*center\_y* – Y-координата центроида области.

### Описание

Результатом действия оператора **Create Region** является новый объект типа "область".

Параметр *num\_polygons* определяет количество многоугольников (полигонов), включенных в область. Если *num\_polygons* приравнять нулю, то оператор создаст пустую область (область без полигонов). Позже Вы можете, используя **Оператор Alter Object**, добавлять все необходимые детали в этот объект.

В некоторых приложениях Вам будет удобно разделить создание области на два этапа: сначала при помощи оператора **Create Region** создать объект, не имеющий полигонов, а затем добавлять в объект необходимые элементы, используя оператор **Alter Object**. Если на момент создания объекта нет полной информации о количестве и расположении многоугольников и узлов в будущей области, используйте для добавления узлов. Более подробную информацию см. в разделе **Оператор Alter Object on page 99**.

Если оператор использует предложение **Into Variable**, то созданный объект объявляется как значение объектной переменной. Если слово **Into** указывает окно, объект помещается на заданное место в окне (например, на изменяемый слой). Если **Into** вообще нет в операторе, MapBasic попытается создать объект в самом верхнем окне. Если это невозможно (например, поверх всех окон лежит окно Графика), то объект не будет создан.

Параметры *x* и *y* задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**. При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты. Если объект создается для окна Отчет, измерения производятся в объявленных ранее единицах измерения листа: X-координата – это расстояние от левого края листа до точки, и Y-координата – расстояние от верхнего края листа. По умолчанию, MapBasic использует дюймы. Чтобы использовать другие единицы, выполните **Оператор Set Paper Units**.

**Внимание:** Перед созданием объекта в окне Отчета не забудьте выполнить оператор **Set CoordSys Layout**.

Необязательный параметр *Pen* задает стиль линии контура объекта; подробнее см. раздел **Предложение Pen on page 492**. Если параметр *Pen* не задан, оператор **Create Region** использует текущий стиль линии (который задается в диалоге **Настройки > Стиль Линии**). Аналогично, предложение *Brush* назначает стиль штриховки; подробнее см. раздел **Предложение Brush on page 117**.

Объект типа "область", состоящий из одного полигона, может содержать до 1,048,572 узлов. Для многофрагментных объектов лимит узлов меньше: на каждый полигон надо убавлять по три узла. Общее количество фрагментов в полигоне не должно превышать 32,000.

**Пример:**

```
Dim obj_region As Object Dim x(100), y(100) As Float Dim i, node_count As Integer
' Если в массивы x() and y() ' мы поместили координаты узлов, ' то
следующие операторы 'создадут на их основе объект типа "область": '
Сначала создадим пустой объект Create Region Into Variable obj_region 0 '
Теперь внесем информацию об узлах: For i = 1 to node_count Alter Object
obj_region Node Add ( x(i), y(i) ) Next ' Теперь поместим область в
таблицу SITES: Insert Into Sites (Object) Values (obj_region)
```

**См. также:**

**Оператор Alter Object, Предложение Brush, Оператор Insert, Предложение Pen, Оператор Update**

---

## Оператор Create Report From Table

**Назначение**

Создает файл отчета для Crystal Reports из открытой таблицы MapInfo.

**Синтаксис**

```
Create Report From Table tablename [Into reportfilespec] [Interactive]
```

*tablename* – это имя открытой таблицы в MapInfo Professional;

*reportfilespec* – полный путь и имя файла для нового файла отчета.

Ключевое слово **Interactive** означает, что новый отчет будет немедленно загружен в модуль Crystal Report Designer. Режим Interactive употребляется, если опущено предложение **Into**. Нельзя создать отчет для растровой таблицы или грид-файла, сразу появится сообщение об ошибке.

**См. также:**

**Оператор Open Report**

---

## Оператор Create RoundRect

**Назначение**

Создает объект типа "скругленный прямоугольник".

**Синтаксис**

```
Create RoundRect
[ Into { Window window_id | Variable var_name } ]
( x1, y1 ) ( x2, y2 ) rounding
[ Pen... ]
[ Brush... ]
```

*window\_id* – идентификатор окна.

*var\_name* – имя объектной переменной;

*x1, y1* – координаты начального угла прямоугольника;

*x2, y2* – координаты противоположного по диагонали угла прямоугольника;

*rounding* – действительное число, задающее в единицах измерения окна (например, в дюймах для Отчета или в градусах для Карты) диаметр скругления угла прямоугольника.

Слово **Pen** начинает стандартное предложение для назначения стиля линии.

Слово **Brush** начинает стандартное предложение для назначения стиля штриховки.

### Описание

Результатом действия оператора **Create RoundRect** является новый объект типа "сглаженный прямоугольник" (прямоугольник со скругленными углами).

Параметры *x* и *y* задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**. При этом надо учесть, что MapBasic игнорирует координатную систему самого окна Карты. Если объект создается для окна Отчет, измерения производятся в объявленных ранее единицах измерения листа: X-координата – это расстояние от левого края листа до точки, и Y-координата – расстояние от верхнего края листа. По умолчанию, MapBasic использует дюймы. Чтобы сменить единицы измерения, выполните **Оператор Set Paper Units**.

**Внимание:** Перед созданием объекта в окне Отчета не забудьте выполнить оператор Set CoordSys Layout.

Необязательный параметр *Pen* задает стиль линии контура объекта; подробнее см. раздел **Предложение Pen on page 492**. Если параметр *Pen* не задан, оператор **Create RoundRect** использует текущий стиль линии (который задается в диалоге **Настройки > Стиль Линии**). Аналогично, предложение *Brush* назначает стиль штриховки; подробнее см. раздел **Предложение Brush on page 117**.

**См. также:**

**Предложение Brush, Оператор Create Rect, Оператор Insert, Предложение Pen, Оператор Update**

---

## Оператор Create Styles

### Назначение

Создает массив из значений стилей линии, штриха или символа.

### Синтаксис

#### Create Styles

```
From { Pen... | Brush... | Symbol... }  
To { Pen... | Brush... | Symbol... }  
Vary { Color By { "RGB" | "HSV" } | Background By { "RGB" | "HSV" } |
```

```

    Size By { "Log" | "Sqrt" | "Constant" }}
  [ Number num_styles ]
  [ Inflect At range_number With { Pen... | Brush... | Symbol... } ]
Into Variable array_variable

```

*num\_styles* – число создаваемых значений стиля, по умолчанию 4;

*range\_number* – число типа SmallInt, задающее номер диапазона последнего перед переломом;

*array\_variable* – массив переменных типа Pen, Brush или Symbol.

## Описание

Оператор **Create Styles** создает множество значений стиля линии, штриха или символа и присваивает его массиву переменных соответствующего типа. Массив этих значений в дальнейшем может использовать **Оператор Shade** для создания условных (тематических) Карт. О создании тематических Карт смотрите документацию MapInfo Professional.

Предложения **From** и **To** задают первое и последнее значения стиля: Pen, Brush или Symbol. Если массив позже будет использован в условном (тематическом) выделении, то стилем из предложения **From** будут выделены объекты самого "нижнего" диапазона, а объекты самого "верхнего" диапазона будут выделены стилем из предложения **To**.

Оператор **Create Styles** строит множество переходных значений от значения, начиная с заданного в предложении **From**, до значения стиля, заданного в предложении **To**. Например, после слова **From** может быть задано **Предложение Brush** – насыщенный синий для морских глубин, а после слова **To** задан – светло-голубой для отмелей. В этом случае MapInfo Professional построит последовательность значений стиля штриха (тип Brush), где первое будет иметь насыщенный синий цвет, последнее значение стиля – светло-голубой, а промежуточные значения – переходные оттенки.

Предложение **Number** задает количество значений стиля, включая заданные предложениями **To** и **From** первое и последнее значения. Это число должно соответствовать числу диапазонов условного (тематического) выделения, если затем выполняется **Оператор Shade**.

Предложение **Vary** задает порядок, согласно которому атрибуты значений стиля будут меняться в создаваемом ряду. Для задания изменения цвета штриховок и линий переднего плана используется подпредложение **Color**. Для задания изменения цвета фона используется подпредложение **Background**. В обоих случаях задается перебор значений по шкале RGB или по шкале HSV. Если Вы создаете ряд для стиля символа, Вы можете использовать подпредложение **Size** для интерполяции по размеру символа. Для стиля линии (тип Pen) подпредложение **Size** задает изменение толщины линии.

Если Вы используете предложение **Inflect At**, то Вы задаете переломную (или пороговую) точку в вычислении ряда значений стилей между значениями, заданными предложением **From**, и значениями, заданными предложением **To**. При этом MapInfo Professional создает две последовательности значений: одну от **From** до значения перелома **Inflect At**, и другую от **Inflect** до **To**. Например, переломное значение используется при построении условной карты прибылей и убытков. Области, где предприятия получают прибыль, будут окрашены разными оттенками зеленого, а области, где предприятия терпят убытки, будут окрашены разными оттенками красного. Переломное значение можно использовать только при раскраске разными цветами.

Предложение **Into Variable** задает имя массива переменных (без скобок). Вам нет необходимости заботиться о размерности массива, MapBasic автоматически подгонит размерность массива по числу значений, если это необходимо. Значения в массиве *array\_variable* (Pen, Brush или Symbol) должны соответствовать по стилю тем, которые заданы в предложениях **From - To**.

### Пример:

Следующий пример иллюстрирует применение оператора **Create Styles**.

```
Dim brush_styles( ) As Brush Create Styles From Brush(2, CYAN, 0) 'стиль для нижнего диапазона To Brush (2, BLUE, 0) 'стиль для верхнего диапазона Vary Color By "RGB" Number 5 Into Variable brush_styles
```

Этот оператор **Create Styles** создает ряд из пяти значений стиля штриха. Значения помещаются в ячейки массива "b\_ranges". Следующий **Оператор Shade** может создать условную карту, в которой будут использованы стили из массива "b\_ranges". См. пример в разделе **Оператор Create Ranges на стр. 86**.

### См. также:

**Оператор Create Ranges, Оператор Set Shade, Оператор Shade**

## Оператор Create Table

### Назначение

Создает новую таблицу.

### Синтаксис

```

Create Table table
( column columntype [ , ... ] ) | Using from_table {
  [ File filespec ]
  [ { Type NATIVE |
    Type DBF [ CharSet char_set ] |
    Type { Access | ODBC } database_filespec [ Version version ]
    Table tablename
    [ Password pwd ] [ CharSet char_set ]
  } ]
  [ Version version ]

```

*table* – имя для новой таблицы MapInfo.

*column* – имя колонки в новой таблице. Имя колонки может быть длиной до 31 символов, и может содержать буквы, числа, и знак подчеркивания. Имя колонки не может начинаться с цифры.

*from\_table* – имя открытой таблицы, в которой находится та колонка, которую надо сохранить в новой таблице. Таблица *from\_table* должна быть базовой и должна содержать колонку с данными. Таблицы запросов и растровые таблицы не могут использоваться для этих целей, их применение даст сообщение об ошибке. Структура колонок в новой таблице будет идентичной той, которая задана в исходной таблице.

*filespec* – определяет место, где создавать файлы TAB, MAP, и ID (и в случае Access, AID файлов). Если Вы пропускаете предложение **File**, то файлы создаются в текущем каталоге.

*char\_set* – имя набора символов; см. раздел, в котором описан [Предложение CharSet on page 136](#);

*database\_filespec* – строка, определяющая доступную базу данных Access. Если такая база не существует, MapInfo создаст новый Access-файл MDB;

*version* – выражение, определяющее версию базы данных Microsoft Jet, используемую для новой базы данных. Допускаются значения 4.0 (для Access 2000) или 3.0 (для Access '95/'97). Если выражение пропущено, то используется версия 4.0. Если база данных, в которой создается таблица, уже существует, то определение версии базы данных игнорируется;

*tablename* – строковая переменная, определяющая имя таблицы, которая появится в Access;

*pwd* – пароль базы данных, заданный при включении защиты базы данных;

*version* – равно 100 (при создании таблицы, которая может читаться ранними версиями MapInfo) или 300 (формат MapInfo). Этот параметр не применяется при создании таблицы Access; версия для таблицы Access управляется DAO.

*columnntype* – тип данных, связанных с колонкой. Каждый параметр *columnntype* задает тип данных колонки и имеет следующий синтаксис:

```
Char( width ) | Float | Integer | SmallInt |  
Decimal( width, decplaces ) | Date | Logical
```

*width* – определяет максимальный размер каждого поля (применяется не ко всем типам полей). Символьные поля могут содержать до 254 символов.

*decplaces* – определяет число знаков после десятичной точки для поля десятичного типа.

### Описание

Оператор **Create Table** создает новую пустую таблицу размером до 250 колонок. Определив параметр **ODBC**, можно создать новую таблицу на сервере DBMS.

Предложение **Using** позволяет создавать новую таблицу в ходе операции объединения объектов по колонке. Таблица *from\_table* должна быть базовой и должна содержать колонку с данными. Таблицы запросов и растровые таблицы не могут использоваться для этих целей, их применение даст сообщение об ошибке. Структура (колонок) создаваемой новой таблицы будет идентична структуре данной таблице.

Добавочное предложение **File** определяет место на диске, где создается новая таблица. Если предложение **File** не используется, таблица создается в текущем каталоге (папке).

Предложение **Type** определяет формат данных таблицы. По умолчанию это формат MapInfo (**NATIVE**), но может быть также и **DBF**. Формат MapInfo (NATIVE) занимает меньше места, чем формат DBF, зато формат DBF читается в любых dBASE-совместимых системах управления базами данных. Таким же образом создаются новые таблицы на серверах DBMS с помощью предложения **ODBC Type** оператора **Create Table**.

Предложение **CharSet** определяет набор символов. Параметр *char\_set* должен быть строковой константой, такой как "MacRoman" или "WindowsLatin1". Если предложение **CharSet** не определено, MapBasic использует по умолчанию текущий набор символов Windows. Более подробную информацию см. в разделе [Предложение CharSet on page 136](#).

Величина типа **SmallInt** для колонок резервирует два байта для каждого значения; так, колонка может содержать значения от -32,767 до +32,767. Величина типа целое (**Integer**) для колонок резервирует четыре байта для каждого значения; так, колонка может содержать значения от -2,147,483,647 до +2,147,483,647.

Предложение **Version** контролирует формат таблицы. Если Вы зададите `Version 100`, MapInfo Professional создаст таблицу в формате, читаемом ранними версиями MapInfo. Если Вы зададите `Version 300`, MapInfo Professional создаст таблицу в формате, используемом MapInfo 3.0. Обратите внимание, что объекты типа полилиния и регион, имеющие более 8,000 узлов и полилинии, состоящие из множества сегментов требуют версию 300. Если Вы пропустите предложение **Version**, то таблица сохранится в формате версии 300.



**Пример:**

Следующий пример показывает, как создать таблицу, названную TOWNS, содержащую 3 поля: символьное поле, названное townname, целочисленное поле, названное population, и десятичное поле, названное median\_income. Файл будет создан в поддиректории C:\MAPINFO\DATA. Поскольку использовано необязательное предложение **Type**, таблица будет создана в формате dBASE.

```
Create Table Towns
( townname Char(30),
  population SmallInt,
  median_income Decimal(9,2) )
File "C:\MAPINFO\TEMP\TOWNS"
Type DBF
```

**См. также:**

**Оператор Alter Table, Оператор Create Index, Оператор Create Map, Оператор Drop Table, Оператор Export, Оператор Import, Оператор Open Table**

---

## Функция CreateText( )

**Назначение**

Возвращает текстовый объект, созданный в определенном окне Карты.

**Синтаксис**

**CreateText**( window\_id, x, y, text, angle, anchor, offset )

*window\_id* – целочисленный идентификатор окна Карты;

*x, y* – координаты, задающие закрепленное положение подписи, действительные числа;

*text* – строка с текстом подписи, представляющим текстовый объект;

*angle* – угол поворота подписи в градусах, действительное число; для горизонтального текста он равен нулю;

*anchor* – целое число типа Integer от 0 до 8, контролирующее расположение подписи относительно места привязки. Ниже перечислены возможные значения кодов, которые описаны в MAPBASIC.DEF.

```
LAYER_INFO_LBL_POS_CC (0)
LAYER_INFO_LBL_POS_TL (1)
LAYER_INFO_LBL_POS_TC (2)
LAYER_INFO_LBL_POS_TR (3)
LAYER_INFO_LBL_POS_CL (4)
LAYER_INFO_LBL_POS_CR (5)
LAYER_INFO_LBL_POS_BL (6)
LAYER_INFO_LBL_POS_BC (7)
LAYER_INFO_LBL_POS_BR (8)
```

Двухбуквенное окончание определяет ориентацию подписи: T=Top, B=Bottom, C=Center, R=Right, L=Left. Например, для размещения подписи ниже и правее места привязки, укажите код LAYER\_INFO\_LBL\_POS\_BR, или укажите значение 8.

*offset* – целое число от 0 до 50, расстояние в точках от подписи до точки привязки на подписываемом объекте; *offset* игнорируется, если значение кода привязки равно 0.

### Возвращаемая величина

Объект

### Описание

Функция **CreateText( )** возвращает величину типа Object, являющуюся текстовым объектом.

Текстовый объект использует текущий стиль текста. Для создания текстового объекта с определенным стилем, используйте **Оператор Set Style** перед вызовом **CreateText( )**.

В тот момент, когда текст создан, его высота контролируется размером текущего шрифта. Таким образом, после создания текстового объекта его высота зависит от размера окна Карты; изменение масштаба ведет к соответственному изменению размера текста.

Возвращаемый объект имеет тип Object и сохраняется в существующей строке таблицы (используя **Оператор Update**) или вставляется в новую строку таблицы (используя **Оператор Insert**).

### Пример:

Следующий пример создает текстовый объект и помещает его на Косметический слой Карты (целочисленная переменная *i\_map\_id* содержит идентификатор окна Карты).

```
Insert Into Cosmetic1 (Obj)
  Values ( CreateText(i_map_id, -80, 42.4, "Sales Map", 0,0,0) )
```

### См. также:

**Оператор AutoLabel**, **Оператор Create Text**, **Предложение Font**, **Оператор Insert**, **Оператор Update**

---

## Оператор Create Text

### Назначение

Создает объект типа "текст".

### Синтаксис

#### Create Text

```
[ Into { Window window_id | Variable var_name } ]
text_string
( x1, y1 ) ( x2, y2 )
[ Font... ]
[ Label Line { Simple | Arrow } ( label_x, label_y ) ]
[ Spacing { 1.0 | 1.5 | 2.0 } ]
```

```
[ Justify { Left | Center | Right } ]
[ Angle text_angle ]
```

*window\_id* – целое число, идентификатор окна Карты или Отчета;

*var\_name* – имя объектной переменной;

*text\_string* – текст длиной до 255 символов (многострочный текст содержит символ Chr\$(10)).

*x1, y1* – координаты одного угла прямоугольника, заполненного текстом, действительные числа;

*x2, y2* – координаты противоположного по диагонали угла прямоугольника, действительные числа;

Предложение **Предложение Font** определяет стиль шрифта текстовых объектов. Размер шрифта в предложении Font игнорируется для окна Карты; о размере шрифта в окне Отчета сказано ниже.

*label\_x, label\_y* – координаты места, к которому прикреплен текст;

*text\_angle* – угол поворота текста в градусах, действительная величина.

### Описание

Параметры *x* и *y* задаются в той координатной системе, которая была объявлена MapBasic ранее. Если система не объявлялась, то координаты будут принимать значения широты и долготы. Для смены координатной системы в среде MapBasic используется **Оператор Set CoordSys**. Перед созданием объекта в окне Отчета не забудьте выполнить оператор Set CoordSys Layout.

Параметры *x1, y1, x2* и *y2* задают прямоугольник текста. Если текстовый объект создается в окне Карты, текст будет заполнять заданную текстовую область так, чтобы длина строки была равна ширине текстовой области. Размер шрифта, заданный предложением **Предложение Font**, будет проигнорирован. В окне Отчета текст будет рисоваться заданным в предложении размером, при этом координаты верхнего левого угла текстовой области (*x1, y1*) будут задавать расположение объекта на листе, а вторая пара (*x2, y2*) будет проигнорирована.

**См. также:**

**Оператор AutoLabel, Функция CreateText( ), Предложение Font, Оператор Insert, Оператор Update**

---

## Функция CurDate( )

### Назначение

Возвращает текущее значение даты в формате ГГГГММДД.

### Синтаксис

**CurDate** ( )

## Функция CurDateTime

---

### Возвращаемая величина

Дата типа Date

### Описание

Функция **Curdate( )** возвращает текущее значение даты. Дата всегда будет возвращаться в формате ГГГГММДД. Для того чтобы преобразовать возвращаемое значение в формат локальной системы, используются **Функция FormatDate\$( )** или **Функция Str\$( )**.

### Пример:

```
Dim d_today As Date  
d_today = CurDate( )
```

### См. также:

**Функция Day( )**, **Функция Format\$( )**, **Функция Month( )**, **Функция StringToDate( )**, **Функция Timer( )**, **Функция Weekday( )**, **Функция Year( )**

---

## Функция CurDateTime

### Назначение

Возвращает текущие значения даты и времени.

### Синтаксис

CurDateTime

### Возвращаемая величина

DateTime

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim X as datetime  
X = CurDateTime()  
Print X
```

---

## Функция CurrentBorderPen( )

### Назначение

Возвращает текущее значение стиля линии границы региона.

### Синтаксис

CurrentBorderPen( )

**Возвращаемая величина**

Pen

**Описание**

Функция **CurrentBorderPen( )** возвращает значение текущей установки стиля линии контура. MapInfo Professional изменяет текущий стиль линии к любому площадному объекту, нарисованному пользователем. Если программа MapBasic создает объект, используя **Оператор Create Region** или подобный, но в нем не задано **Предложение Pen**, то в объекте будет использован текущий стиль BorderPen.

Возвращаемое значение может быть присвоено переменной типа Pen, или может быть использовано как параметр внутри оператора, который использует параметр типа Pen (например **Оператор Set Map**).

Для извлечения отдельных атрибутов стиля Pen (таких как color) используется **Функция StyleAttr( )**. Более подробно настройки Pen описаны в разделе **Предложение Pen on page 492**.

**Пример:**

```
Dim p_user_pen As Pen p_user_pen = CurrentBorderPen( )
```

**См. также:**

**Функция CurrentPen( )**, **Предложение Pen**, **Оператор Set Style**, **Функция StyleAttr( )**

---

**Функция CurrentBrush( )****Назначение**

Возвращает значение установленного на данный момент стиля штриховки.

**Синтаксис**

```
CurrentBrush ( )
```

**Возвращаемая величина**

Brush

**Описание**

Функция **CurrentBrush( )** возвращает значение текущей установки стиля штриха. В MapInfo это значение изменяется в диалоге команды **Настройки > Стиль областей**. Когда Вы рисуете в окне такие объекты, как эллипс, прямоугольник, сглаженный прямоугольник или регион, MapInfo заполняет его заданной штриховкой. Когда программа на MapBasic создает такие объекты (например, используя **Оператор Create Region** или подобный), но не задано **Предложение Brush**, объект заштриховывается в соответствии с текущей установкой стиля штриха в MapInfo.

## Функция CurrentFont( )

---

Величина, возвращаемая функцией **CurrentBrush( )**, может быть присвоена переменной типа Brush или использована как параметр в операторах, в которых используется установка стиля штриха (таких, как **Оператор Set Map** или **Оператор Shade**).

Для вывода отдельных характеристик стиля штриха (например, цвета) используется **Функция StyleAttr( )**.

Более подробно о стиле можно прочитать в разделе **Предложение Brush on page 117**.

### Пример:

```
Dim b_current_fill As Brush  
b_current_fill = CurrentBrush( )
```

### См. также:

**Предложение Brush**, **Функция MakeBrush( )**, **Оператор Set Style**, **Функция StyleAttr( )**

---

## Функция CurrentFont( )

### Назначение

Возвращает значение шрифта, используемого в данный момент в окне Карты или Отчета.

### Синтаксис

**CurrentFont( )**

### Возвращаемая величина

Шрифт

### Описание

Функция **CurrentFont( )** возвращает значение текущей установки стиля шрифта. В MapInfo это значение изменяется в диалоге команды **Настройки > Стиль текста** в окне Карты или Отчета. Когда Вы рисуете в окне такие объекты, как эллипс, прямоугольник, сглаженный прямоугольник или регион, MapInfo заполняет его установленным штрихом. Когда программа на MapBasic создает такие объекты (например, используя **Оператор Create Text**), но не включает **Предложение Font**, объект заштриховывается в соответствии с текущей установкой стиля штриха в MapInfo.

Величина, возвращаемая функцией **CurrentFont( )**, может быть присвоена переменной типа Brush или использована как параметр в операторах, в которых используется установка стиля штриха (таких, как **Оператор Set Legend**).

Для вывода отдельных характеристик стиля шрифта (например, цвета) используется **Функция StyleAttr( )**.

Более подробно о стиле можно прочитать в разделе **Предложение Font on page 317**.

**Пример:**

```
Dim f_user_text As Font  
f_user_text = CurrentFont( )
```

**См. также:**

[Предложение Font](#), [Функция MakeFont\( \)](#), [Оператор Set Style](#), [Функция StyleAttr\( \)](#)

---

## Функция CurrentLinePen( )

**Назначение**

Возвращает значение шрифта, используемого в данный момент в окне Карты.

**Синтаксис**

```
CurrentLinePen( )
```

**Возвращаемая величина**

Pen

**Описание**

Функция **CurrentLinePen( )** возвращает значение текущей установки стиля шрифта. MapInfo Professional присваивает текущий стиль линии любой линии или полилинии, нарисованной пользователем. Если программа MapBasic создает объект, используя [Оператор Create Line](#), или подобный ему, но при этом сам оператор не включает в себя параметр Pen, то объект будет использовать текущий стиль линии. Возвращаемое значение может быть присвоено переменной типа Pen, или может быть использовано как параметр внутри оператора, который использует параметр типа Pen (например [Оператор Set Map](#)).

Для извлечения отдельных атрибутов стиля Pen (таких как color) используется [Функция StyleAttr\( \)](#). Более подробно настройки Pen описаны в разделе [Предложение Pen on page 492](#).

**Пример:**

```
Dim p_user_pen As Pen p_user_pen = CurrentPen( )
```

**См. также:**

[Функция CurrentBorderPen\( \)](#), [Предложение Pen](#), [Оператор Set Style](#), [Функция StyleAttr\( \)](#)

---

## Функция CurrentPen( )

**Назначение**

Возвращает текущий стиль линии Pen.

### Синтаксис

`CurrentPen( )`

### Возвращаемая величина

Pen

### Описание

Функция **CurrentPen( )** возвращает значение текущей установки стиля линии. MapInfo Professional присваивает текущий стиль линии любой линии или полилинии, нарисованной пользователем. Если программа MapBasic создает объект, используя **Оператор Create Line**, или подобный ему, но при этом сам оператор не включает в себя параметр Pen, то объект будет использовать текущий стиль линии. Если нужно применить текущий стиль линии, не переустанавливая стиль границы региона, то используется **Функция CurrentLinePen( )**.

Возвращаемое значение может быть присвоено переменной Pen, или может быть использовано как параметр внутри оператора, который использует настройку Pen (например **Оператор Set Map**).

Для извлечения отдельных атрибутов стиля Pen (таких как color) используется **Функция StyleAttr( )**. Более подробно настройки Pen описаны в разделе **Предложение Pen on page 492**.

### Пример:

```
Dim p_user_pen As Pen  
p_user_pen = CurrentPen( )
```

### См. также:

**Функция MakePen( )**, **Предложение Pen**, **Оператор Set Style**, **Функция StyleAttr( )**

---

## Функция CurrentSymbol( )

### Назначение

Возвращает значение установленного на данный момент стиля символа для точечного объекта.

### Синтаксис

`CurrentSymbol( )`

### Возвращаемая величина

Символ



## Описание

Функция **CurrentSymbol( )** возвращает значение текущей установки стиля символа для точечного объекта. В MapInfo это значение может изменяться в диалоге команды **Настройки > Стил ь с имвол ов**. Когда Вы рисуете в окне точечный объект, MapInfo обозначает его установленным символом. Когда программа MapBasic создает такой объект, используя **Оператор Create Point**, но не использует параметр Symbol, объект изображается в соответствии с текущей установкой стиля символа в MapInfo.

Величина, возвращаемая функцией **CurrentSymbol( )**, может быть присвоена переменной типа Symbol или использована как в операторах, в которых используется установка стиля символа (таких, как **Оператор Set Map** или **Оператор Shade**).

Для вывода отдельных характеристик стиля символа (например, цвета) используется **Функция StyleAttr( )**. Более подробно о стиле можно прочитать в разделе **Предложение Symbol on page 725**.

## Пример:

```
Dim sym_user_symbol As Symbol
sym_user_symbol = CurrentSymbol( )
```

## См. также:

**Функция MakeSymbol( )**, **Оператор Set Style**, **Функция StyleAttr( )**, **Предложение Symbol**

---

## Функция CurTime

### Назначение

Возвращает текущее значение времени.

### Синтаксис

```
CurTime
```

### Возвращаемая величина

Время

## Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim Y as time
Y = CurTime()
Print Y
```



## Функция DateWindow( )

### Назначение

Возвращает текущую дату, установленную как целое в диапазоне от 0 до 99, или (-1) если режим управления окном даты отключен.

### Синтаксис

**DateWindow**( *context* )

*context* –это короткое целое, которое может принимать значения DATE\_WIN\_CURPROG или DATE\_WIN\_SESSION.

### Описание

Использование функции зависит от того, какой ей передается контекст (context). Если контекст это DATE\_WIN\_SESSION, то возвращаются текущие настройки компьютера. Если контекст это DATE\_WIN\_CURPROG, то возвращаются локальные настройки MapBasic. Если контекст пропущен, то возвращаются настройки текущего сеанса.

MBX, скомпилированные до версии 5.5 будут конвертировать двузначный год в текущее столетие (поведение версии 5.0 и более ранних). Что бы получить новое поведение отнесения двузначных данных к разным столетиям, перекомпилируйте программу в MapBasic v5.5.

### Пример:

В следующем примере переменная Date1 = 19890120, Date2 = 20101203 и MyYear = 1990.

```
DIM Date1, Date2 as Date
DIM MyYear As Integer
Set Format Date "US"
Set Date Window 75
Date1 = StringToDate("20.01.89")
Date2 = StringToDate("03.12.10")
MyYear = Year("30.12.90")
```

### См. также:

**Оператор Set Date Window**

## Функция Day( )

### Назначение

Возвращает номер дня (от 1 до 31) из даты.

### Синтаксис

**Day**( *date\_expr* )

*date\_expr* – выражение, в результате которого получается дата.

### Возвращаемая величина

Короткое целое число (от 1 до 31). Величина типа SmallInt.

### Описание

Функция **Day( )** возвращает целочисленное значение от одного до тридцати одного, являющееся номером дня в месяце. Например, для даты 12/17/92 функция **Day( )** будет возвращать 17.

### Пример:

```
Dim day_var As SmallInt, date_var As Date
date_var = StringToDate("23.05.85")
day_var = Day(date_var)
```

### См. также:

[Функция CurDate\( \)](#), [Функция Month\( \)](#), [Функция Timer\( \)](#), [Функция Year\( \)](#)

---

## Оператор DDEExecute

### Назначение

Посылает по каналу DDE-связи (динамического обмена данными) в другую программу выполняемую команду.

### Синтаксис

**DDEExecute** *channel, command*

*channel* – номер открытого канала DDE-связи, целое число, возвращаемое функцией DDEInitiate( );

*command* – команда, посылаемая для выполнения в другую программу (DDE-сервер), строковая величина.

### Описание

Оператор **DDEExecute** посылает исполняемую команду в другую программу, присоединенную к каналу DDE-связи.

Номер канала задает [Функция DDEInitiate\( \)](#), которая открывает канал динамического обмена данными.

Параметр *command* должен представлять команду, воспринимаемую DDE-сервером (пассивной стороной связи). Правильный формат команды зависит от правил, принятых в подсоединенной программе. Вы должны изучить документацию другой программы, прежде чем налаживать DDE-связь.

### Ошибки:

ERR\_CMD\_NOT\_SUPPORTED, если программа выполняется не в среде Windows;

ERR\_NO\_RESPONSE\_FROM\_APP, если программа-сервер не отвечает.

**Пример:**

Программными средствами MapBasic Вы можете открыть канал DDE-связи с Microsoft Excel и обращаться к нему как к программе-серверу. Если канал был открыт для объекта (topic) "System", Вы можете с помощью оператора **DDEExecute** выполнить как команду, так и макрофункцию Excel. Чтобы Excel выполнил макрофункцию, ее нужно заключить в квадратные скобки. В следующем примере открывается рабочая таблица "TRIAL.XLS" в программе Excel:

```
Dim i_chan As Integer
i_chan = DDEInitiate("Excel", "System")
DDEExecute i_chan, "[OPEN(""C:\DATA\TRIAL.XLS"")] "
```

**См. также:**

**Функция DDEInitiate( ), Оператор DDEPoke, Функция DDERequest\$( )**

---

## Функция DDEInitiate( )

**Назначение**

Открывает новый канал DDE-связи и возвращает его номер.

**Синтаксис**

```
DDEInitiate( appl_name, topic_name )
```

*appl\_name* – имя подключаемой программы (например, "MapInfo"), строковая величина;

*topic\_name* – имя документа или некоторого объекта (topic) программы (например, "System"), строковая величина.

**Возвращаемая величина**

Целое число типа Integer.

**Описание**

Функция **DDEInitiate( )** иницирует один канал DDE-связи (динамического обмена данными), назначая ему номер.

Этот канал связи позволяет соединить две программы, работающие в среде Microsoft Windows, для пересылки информации. Как только канал открыт, MapBasic может читать информацию из документа другой программы (**Функция DDERequest\$( )**) и записывать в этот документ (**Оператор DDEPoke**). После информационного обмена каналы связи рекомендуется закрыть, используя **Оператор DDETerminate** или **Оператор DDETerminateAll**.

**Внимание:** DDE-связь возможна только в среде Microsoft Windows. Если Ваше приложение будет обращаться к услугам DDE в среде другой вычислительной платформы, то MapBasic выдаст ошибку. Для обхода такого рода конфликта используется **Функция SystemInfo( )**. С ее помощью Вы можете определить, в какой среде выполняется Ваша программа.

Параметр *appl\_name* задает имя программы, которое понятно для DDE-связи. Например, для установления связи с Microsoft Excel, присвойте параметру *appl\_name* значение "Excel". Windows-программа, с которой связывается MapBasic, заданная в параметре *appl\_name*, должна быть уже загружена; для этого в программе MapBasic используется . Не все программы Windows поддерживают DDE-связь. Для получения информации о поддержке DDE смотрите документацию этих программ.

Параметр *topic\_name* задает некоторый объект в программе, с которым возможен обмен информацией. Каждая программа обладает некоторым набором таких объектов. О списке объектов, поддерживаемых определенной программой, Вы можете прочитать в соответствующей документации. Для многих программ именем объекта, обменивающегося информацией, является имя файла документа. Например, пусть "ORDERS.XLS" – имя файла рабочей таблицы программы Microsoft Excel, тогда оператор

```
Dim i_chan As Integer
i_chan = DDEInitiate("Excel", "C:\ORDERS.XLS")
```

открывает канал связи с ним.

Многие программы поддерживают специальный объект "System". Начав DDE-обмен, используя объект "System", Вы можете получить список строк, представляющих корректные имена объектов, поддерживаемые данной программой, например, список открытых файлов; для этого используется **Функция DDERequest\$( )**. Получив список открытых файлов, можно начать новый сеанс DDE-связи с любым документом. Смотрите пример ниже.

В следующей таблице приводится список некоторых программ и их объектов для использования в функции **DDEInitiate( )**.

| Вызов функции DDEInitiate( )      | Что происходит при DDE-обмене  |
|-----------------------------------|--|
| DDEInitiate("Excel", "System")    | <b>Функция DDERequest\$( )</b> может извлечь системную информацию Excel, такую, как список рабочих таблиц, загруженных сейчас в среде Excel, а <b>Оператор DDEExecute</b> может послать команду на выполнение в Excel.                       |
| DDEInitiate("Excel", <i>wks</i> ) | Если <i>wks</i> – это имя документа Excel (например, "Sheet1" или "May.xls"), то следующий <b>Оператор DDEPoke</b> может поместить какую-нибудь величину в таблицу Excel и <b>Функция DDERequest\$( )</b> может прочитать информацию из нее. |

| Вызов функции DDEInitiate( )     | Что происходит при DDE-обмене   |
|----------------------------------|---|
| DDEInitiate("MapInfo", "System") | Функция DDERequest\$( ) может извлечь такую информацию, как список выполняемых сейчас в MapInfo программ MapBasic.  |
| DDEInitiate("MapInfo" mbx)       | Если mbx – это имя программы на MapBasic, которая сейчас выполняется (например, "C:\GRIDS.MBX"), то следующий оператор Оператор DDEPoke может назначить какую-нибудь величину глобальной переменной в этой программе, а Функция DDERequest\$( ) может прочесть текущее значение такой переменной. |

Когда программа MapBasic выполняет функцию DDEInitiate( ), то MapBasic выступает как "клиент" в DDE-связи. Другие программы Windows выступают как программы-"серверы". В пределах одной связи клиент всегда активен, а сервер только отвечает на запросы клиента. MapBasic может поддерживать одновременно сразу столько каналов, сколько могут позволить ресурсы памяти и системы Вашего компьютера. Прикладная программа одновременно может быть как клиентом в сеансе одной связи (выполняя такие операторы как DDEInitiate( )) так и сервером для другого сеанса связи (когда определена стандартная Процедура RemoteMsgHandler).

Ошибки:

ERR\_CMD\_NOT\_SUPPORTED, если программа выполняется не в среде Windows;  
ERR\_INVALID\_CHANNEL, если неправильно задан номер канала.

Пример:

Следующий фрагмент текста программы иллюстрирует DDE-связь с Microsoft Excel версий 4 и выше. Целью является передача простого сообщения ("Привет от MapInfo!") в первую ячейку первой рабочей таблицы, но только если она пуста. Если ячейка не пуста, ничего записываться не будет, содержимое ячейки будет показано пользователю MapInfo.

```
Dim chan_num, tab_marker As IntegerDim topiclist, topicname, cell As String
chan_num = DDEInitiate("EXCEL", "System")
If chan_num = 0 Then
  Note "Простите, но Excel не отвечает."
End Program
End If ' Вызов списка документов - рабочих таблиц сейчас загруженных в среде Excel
topiclist = DDERequest$(chan_num, "topics")
' Если работает Excel версии 4, здесь должен быть получен список topiclist '": Sheet1 System" ' (если раб. таблица еще не имеет имени) или в таком виде: '": C:Orders.XLS Sheet1 System" '
' Если мы имеем дело с Excel версии 5, то topiclist может выглядеть так: "[Book1]Sheet1 [Book2]Sheet2 ..." '
' Теперь выделим из списка первое имя (может быть "Sheet1") для этого извлечем из строки текст между первым и вторым символом табуляции; ' или для Excel 5 версии текст до первой ' табуляции.

If Left$(topiclist, 1) = ":" Then ' ...это Excel версии 4.
  tab_marker = InStr(3, topiclist, Chr$(9) )
  If tab_marker = 0 Then
    Note
```

```
"В программе Excel нет открытых документов! Остановка."End ProgramEnd If
topicname = Mid$(topiclist, 3, tab_marker - 3)Else ' ... для Excel 5.x
    tab_marker = Instr(1, topiclist, Chr$(9) )
    topicname = Left$( topiclist, tab_marker - 1)
End If

' open a channel to the specific document
' (e.g., "Sheet1")
DDETerminate chan_num
chan_num = DDEInitiate("Excel", topicname)
If chan_num = 0 Then
    Note "Problem communicating with " + topicname End Program
End If

' Let's examine the 1st cell in Excel.
' If cell is blank, put a message in the cell.
' If cell isn't blank, don't alter it -
' just display cell contents in a MapBasic NOTE.
' Note that a "Blank cell" gets returned as a
' carriage-return line-feed sequence:
' Chr$(13) + Chr$(10).
cell = DDERequest$( chan_num, "R1C1" ) If cell <> Chr$(13) + Chr$(10) Then
Note "Сообщение не послано; В ячейке уже содержится:" + cellElse DDEPoke
chan_num, "R1C1", "Привет от MapInfo!"Note "Сообщение послано в
Excel,"+topicname+ ",R1C1."End If DDETerminateAll
```

**Внимание:** Приведенный пример не гарантирует от ошибок. Например, в Excel может в момент вызова быть активно окно графика ("Chart1" или подобные ему); тогда обращение к ячейке R1C1 будет ошибочным. Кроме того, если Вы работаете с русской версией Excel, то обращаться нужно не к R1C1, а к C1K1 и т.д.

**См. также:**

**Оператор DDEExecute, Оператор DDEPoke, Функция DDERequest\$( ), Оператор DDETerminate, Оператор DDETerminateAll**

---

## Оператор DDEPoke

### Назначение

Посылает данные в программу-сервер по каналу DDE-связи.

### Синтаксис

**DDEPoke** *channel, itemname, data*

*channel* – номер открытого канала DDE-связи, целое число; его возвращает **Функция DDEInitiate( )**.

*itemname* – имя элемента объекта, документа программы-сервера, строковая величина;

*data* – символьная строка, посылаемая в itemname.



## Описание

Оператор **DDEPoke** посылает данные по каналу DDE-связи в объект программы-сервера.

Параметр *channel* – это целое число, которое возвращает **Функция DDEInitiate( )**.

Параметр *itemname* задает элемент, который соответствует определенному каналу. Каждая программа, поддерживающая DDE-связь, обладает некоторым набором объектов. О списке объектов, поддерживаемых определенной программой, Вы можете прочитать в соответствующей документации.

Для DDE-связи с документом программы Excel именем элемента является строка, такая как "R1C1" (или "C1K1" в русской версии Excel) для ячейки в первой строке и первой колонке. Именем элемента может быть также имя глобальной переменной из другой программы, если с этой программой открыт канал связи.

## Ошибки:

ERR\_CMD\_NOT\_SUPPORTED, если программа выполняется не в среде Windows;

ERR\_INVALID\_CHANNEL, если неправильно задан номер канала.

## Пример:

В этом примере посылается сообщение ("Привет от MapInfo!") в первую ячейку рабочей таблицы. Этот пример выполняется при условии, что Excel уже загружен и рабочая таблица (Sheet1) открыта и находится в режиме редактирования ячеек.

```
Dim i_chan_num As Integer
i_chan_num = DDEInitiate("EXCEL",
"Sheet1")
DDEPoke i_chan_num, "R1C1", "Привет от MapInfo!"
```

В этом фрагменте глобальной переменной "Address" из программы DISPATCH.MBX посылается новое значение. Оператор **DDEPoke** используется для изменения глобальной переменной Address.

```
i_chan_num = DDEInitiate("MapInfo", "C:\DISPATCH.MBX")
DDEPoke i_chan_num, "Address", "23 Main St."
```

## См. также:

**Оператор DDEExecute, Функция DDEInitiate( ), Функция DDERequest\$( )**

---

## Функция DDERequest\$( )

### Назначение

Возвращает данные, запрошенные через канал DDE-связи.

### Синтаксис

```
DDERequest$( channel, itemname )
```

*channel* – номер открытого канала DDE-связи, целое число; его возвращает **Функция DDEInitiate( )**.

*itemname* – имя объекта программы-сервера, который будет возвращать информацию, строковая величина.

### Возвращаемая величина

Строка

### Описание

Функция **DDERequest\$ ( )** возвращает информацию через канал DDE-связи. Если информация недоступна, **DDERequest\$ ( )** возвращает пустое значение ("").

Параметр *channel* – это целое число, которое возвращает **Функция DDEInitiate ( )**.

Параметр *itemname* задает элемент, который соответствует определенному каналу. Каждая программа, поддерживающая DDE-связь, обладает некоторым набором объектов. О списке объектов, поддерживаемых определенной программой, Вы можете прочитать в соответствующей документации.

В следующей таблице в первой колонке содержатся значения, которые были использованы функцией **DDEInitiate ( )** при открытии канала связи с Excel, как сервером, а во второй колонке соответствующие значения для параметра *itemname* в функции **DDERequest\$ ( )**.

| Имя объекта обмена                   | Значения <i>itemname</i> и результат функции <b>DDERequest\$ ( )</b>  |
|--------------------------------------|---|
| "System"                             | "Sysitems" для получения списка имен элементов, которые доступны по каналу обмена для "System";<br><br>"Topics" для получения списка объектов для DDE-связи с Excel, включая имена всех открытых рабочих таблиц;<br><br>"Formats" для получения списка форматов для системного буфера (Clipboard), поддерживаемых Excel (например, "TEXT, BITMAP ..."). |
| <i>wks</i> (имя электронной таблицы) | "RnCn" для получения строки с содержимым ячейки (здесь первый символ <i>n</i> должен быть номером строки, а второй – номером колонки).  |

**Внимание:** С помощью функции **DDERequest\$ ( )** можно считывать значения глобальных переменных одной программы в другую, выполняющуюся в то же время. Следующая таблица приводит комбинации значений *itemname* и объектов обмена, с которыми открыта связь, для MapInfo как сервера.

| Имя объекта обмена      | Значения itemname и результат функции DDERequest\$( )   |
|-------------------------|---|
| "System"                | "Sysitems" для получения списка имен элементов, которые доступны по каналу обмена для "System";<br><br>"Topics" для получения списка имен объектов для DDE-связи с MapInfo, включая имена выполняемых в данный момент прикладных программ;<br><br>"Formats" для получения списка форматов для системного буфера (Clipboard ), поддерживаемых MapInfo (например, "TEXT");<br><br>"Version" для получения числа, по которому можно определить версию MapInfo (например, "300" означает MapInfo 3.0) |
| mbx (имя MBX-программы) | "{items}" для получения списка имен глобальных переменных выполняющейся программы; задание имени глобальной переменной позволяет функции DDERequest\$( ) вернуть ее значение  |

**Ошибки:**

- ERR\_CMD\_NOT\_SUPPORTED, если программа выполняется не в среде Windows;
- ERR\_INVALID\_CHANNEL, если неправильно задан номер канала;
- ERR\_CANT\_INITIATE\_LINK, если MapBasic не связан с этим объектом обмена.

**Пример:**

В этом примере функция **DDERequest\$( )** читает содержимое первой ячейки рабочей таблицы. Пример работает, если Excel уже загружен.

```
Dim i_chan_num As Integer
Dim s_cell As String
i_chan_num = DDEInitiate("EXCEL", "Sheet1")
s_cell = DDERequest$(i_chan_num, "R1C1")
```

Следующий пример подразумевает, что уже действует другая MapBasic-программа под названием "Dispatch", в которой определена глобальная переменная "Address". С помощью функции **DDERequest\$( )** мы можем получить значение этой переменной.

```
Dim i_chan_num As Integer, s_addr_copy As String
i_chan_num = DDEInitiate("MapInfo", "C:\DISPATCH.MBX")
s_addr_copy = DDERequest$(i_chan_num, "Address")
```

**См. также:**

**Функция DDEInitiate( )**

## Оператор DDETerminate

### Назначение

Закрывает один сеанс DDE-обмена.

### Синтаксис

**DDETerminate** *channel*

*channel* – номер открытого канала DDE-связи, целое число; его возвращает **Функция DDEInitiate( )**.

### Описание

Оператор **DDETerminate** закрывает канал динамического обмена данными.

Номер канала возвращает **Функция DDEInitiate( )** при его открытии. Прделав необходимые Вам действия по обмену, необходимо завершить его закрытием канала, используя оператор **DDETerminate** или **Оператор DDETerminateAll**.

**Внимание:** Каждая из прикладных программ может открывать по одному или несколько DDE-каналов независимо от других программ. Каждая из программ, однако, может закрывать только свои каналы. Программа MapBasic не может закрыть DDE-канал, открытой другой программой MapBasic.

### Ошибки:

ERR\_CMD\_NOT\_SUPPORTED, если программа выполняется не в среде Windows;

ERR\_INVALID\_CHANNEL, если неправильно задан номер канала.

### Пример:

```
DDETerminate i_chan_num
```

### См. также:

**Функция DDEInitiate( )**, **Оператор DDETerminateAll**

---

## Оператор DDETerminateAll

### Назначение

Закрывает все сеансы DDE-обмена, открытые программой MapBasic.

### Синтаксис

**DDETerminateAll**

## Описание

Оператор **DDETerminateAll** закрывает все каналы DDE-связи, которые были открыты "клиентом", программой MapBasic. Программы MapBasic могут быть запущены одновременно, и в каждой может быть открыто множество каналов. Каждая из программ, однако, может закрывать только свои каналы. Программа MapBasic не может закрыть DDE-канал, открытой другой программой MapBasic.

Проделав необходимые Вам действия по обмену, необходимо завершить его закрытием канала, используя **Оператор DDETerminate** или **DDETerminateAll**.

## Ошибки:

ERR\_CMD\_NOT\_SUPPORTED, если программа выполняется не в среде Windows;

## См. также:

**Функция DDEInitiate( ), Оператор DDETerminate**

---

## Оператор Declare Function

### Назначение

Объявляет имя функции и список ее параметров.

### Предупреждение

Этот оператор не может быть использован в окне MapBasic.

Вызов внешних функций (во втором варианте синтаксиса) зависит от вычислительной платформы. К DLL-библиотекам могут обращаться только Windows-программы.

### Синтаксис 1

```
Declare Function fname
  ( [ [ ByVal ] parameter As var_type ]
    [ , [ ByVal ] parameter As var_type... ] ) As return_type
```

*fname* – имя функции;

*parameter* – имя параметра функции;

*var\_type* – стандартный для MapBasic тип данных для параметра или определенный с помощью оператора Type;

*return\_type* – стандартный для MapInfo скалярный тип для величины, полученной в результате.

### Синтаксис (вариант 2 – для Windows DLL):

```
Declare Function fname Lib "file_name" [ Alias "function_alias" ]
  ( [ [ ByVal ] parameter As var_type ]
    [, [ ByVal ] parameter As var_type... ] ) As return_type
```

*fname* – имя вызываемой функции;

*file\_name* – имя DLL-файла;

*function\_alias* – настоящее имя внешней функции;

*parameter* – имя параметра функции;

*var\_type* – стандартный для MapBasic тип данных для параметра или определенный с помощью оператора Type;

*return\_type* – стандартный для MapInfo скалярный тип для величины, полученной в результате.

### Описание

Оператор **Declare Function** объявляет об использовании функции, написанной на языке MapBasic, или внешней функции, а также объявляет типы параметров и результата функции.

Для написания функции на MapBasic используется **Оператор Function...End Function**. Каждая такая функция должна быть объявлена оператором **Declare Function**. Для более подробной информации о создании пользовательских функций, см. раздел **Оператор Function...End Function на стр. 96**.

Параметры могут пересылать значения функции двумя способами: ссылкой ("by-reference") или значением ("by-value", явно задавая ключевое слово **ByVal**). О различии в природе этих двух типов параметров можно прочитать в *Руководстве пользователя MapBasic*.

### Обращение к внешней функции

Второй вариант синтаксиса используется для объявления внешней функции оператором **Declare Function**. Внешняя функция может быть написана на другом языке (например, C или Pascal) и может храниться в отдельном файле. Если Ваша программа использует внешнюю функцию, то она должна быть объявлена так же, как и внутренняя функция программы MapBasic.

Если оператор **Declare Function** объявляет внешнюю функцию, то параметр *file\_name* должен задавать имя файла, содержащего его. Файл с внешними процедурами должен быть доступен во время выполнения программ.

Каждая внешняя функция имеет имя, которое ей дает автор при создании. Обычно, при задании функции оператором **Declare Function** параметр *fname* явно задает имя функции, извлекаемой из внешнего файла. Однако, в оператор **Declare Function** можно включить предложение **Alias** и с его помощью обращаться к внешней функции по произвольному имени. Обычно предложение **Alias** используется, если Вы хотите избежать конфликта с уже имеющимся именем стандартной функции MapBasic.

Если оператор **Declare Function** включает предложение **Alias**, то параметр *function\_alias* должен совпадать с настоящим именем функции, а параметр *fname* должен задавать имя, по которому к ней обращается MapBasic.

## Ограничения для параметров внешней функции Windows DLL

Вы можете использовать разные типы значений для передачи в DLL. DLL-библиотека должна быть откомпилирована в режиме упаковки структур ("structure packing"). См. подробности в *Руководстве пользователя MapBasic*.

### Пример:

В этом примере определяется функция пользователя под названием "CubeRoot", которая вычисляет кубический корень из числа.

```
Declare Sub Main
Declare Function CubeRoot(ByVal x As Float) As Float
Sub Main
    Note Str$( CubeRoot(23) )
End Sub
Function CubeRoot(ByVal x As Float) As Float
    CubeRoot = x ^ (1 / 3)
End Function
```

### См. также:

**Оператор Declare Sub, Оператор Function...End Function**

## Оператор Declare Sub

### Назначение

Объявляет имя и типы параметров подпрограммы.

### Предупреждение

Этот оператор не может быть использован в окне MapBasic.

Вызов внешних функций (во втором варианте синтаксиса) зависит от вычислительной платформы. К DLL-библиотекам могут обращаться только Windows-программы.

### Синтаксис 1

```
Sub proc_name [ ( [ ByVal ] parameter As var_type [ , ... ] ) ]
statement_list End Sub, где
```

*sub\_proc* – имя подпрограммы;

*parameter* – имя параметра из списка подпрограммы;

*var\_type* – тип значения параметра, стандартный или результат оператора Type.

### Синтаксис (вариант 2 – для Windows DLL):

```
Declare Sub sub_proc Lib "file_name" [ Alias "sub_alias" ]
    [ ( [ ByVal ] parameter As var_type [ , ... ] ) ]
```

*sub\_proc* – имя внешней процедуры;

*file\_name* – строка, имя DLL-файла;

*sub\_alias* – оригинальное имя процедуры;

*parameter* – имя параметра из списка подпрограммы;

*var\_type* – стандартный для MapBasic тип данных для параметра или определенный с помощью оператора Type;

### Описание

Оператор **Declare Sub** объявляет имена процедур и списки их параметров. Обычно, каждому оператору **Declare Sub** соответствует одна sub-процедура, которая находится ниже соответствующего оператора.

Такая процедура создается в программе, используя **Оператор Sub...End Sub**, и называется внутренней. Каждой такой процедуре должен предшествовать оператор **Declare Sub**. Более подробно о создании процедур см. в разделе **Оператор Sub...End Sub on page 723**.

Для того чтобы объявить параметр значением, надо перед его именем использовать ключевое слово **ByVal**. По умолчанию параметр объявляется ссылкой.

### Вызов внешних процедур

Второй вариант синтаксиса используется для объявления оператором **Declare Sub** внешней процедуры. Внешняя функция может быть написана на другом языке (например, C или Pascal) и может храниться в отдельном файле. Если Ваша программа использует внешнюю функцию, то она должна быть объявлена так же, как и внутренняя функция программы MapBasic.

Если **Declare Sub** объявляет внешнюю процедуру, то имя файла, в которой она содержится, должно быть задано в параметре *file\_name*. Файл с внешними процедурами должен быть доступен во время выполнения программ.

Каждая внешняя функция имеет имя, которое ей дает автор при создании. Обычно, при задании процедуры оператором **Declare Sub** параметр *sub\_proc* явно задает имя функции, извлекаемой из внешнего файла.. Однако, в оператор **Declare Sub** можно включить предложение **Alias** и с его помощью обращаться к внешней процедуре по произвольному имени. Обычно предложение **Alias** используется, если Вы хотите избежать конфликта с уже имеющимся именем стандартной процедурой MapBasic.

Если оператор **Declare Sub** включает предложение **Alias**, то параметр *sub\_alias* должен содержать оригинальное имя процедуры во внешнем файле, а параметр *sub\_proc* должен задавать имя, которое использует для обращения к ней MapBasic. Вы можете использовать разные типы значений для передачи в DLL. DLL-библиотека должна быть откомпилирована в режиме упаковки структур ("structure packing"). Информацию о том, какие типы переменных, может задавать пользователь, см. в разделе **Оператор Type on page 741**.



**Пример:**

```

Declare Sub Main
Declare Sub Cube(ByVal original As Float, cubed As Float)
Sub Main Dim x, result As Float
Call Cube(2, result) ' переменная result сейчас равна 8 (2 x 2 x 2)
x = 1
Call Cube(x + 2, result) ' переменная result сейчас равна 27 (3 x 3 x 3)
End Sub
Sub Cube (ByVal original As Float, cubed As Float) ' возвести в куб
"original" и поместить результат в "cubed".
    cubed = original ^ 3
End Sub

```

**См. также:**

**Оператор Call, Оператор Sub...End Sub**

---

## Оператор Define

**Назначение**

Назначает имя для постоянной величины.

**Предупреждение**

Оператор **Define** не может быть использован в окне MapBasic.

**Синтаксис**

**Define** *identifier definition*

*identifier* – имя, слово не более 31 символа длиной, начинающееся с буквы или символа подчеркивания (\_);

*definition* – значение, которое MapBasic подставит вместо каждого *identifier* в тексте программы.

**Описание**

Оператор **Define** задает идентификатор для постоянной величины. Перед компиляцией MapBasic сначала заменяет каждый идентификатор *identifier* на текст, который определен параметром *definition*. Примеры применения оператора **Define** Вы можете увидеть в файле MAPBASIC.DEF.

Строчные и прописные символы в именах идентификаторов, заданные оператором **Define**, не различаются. То есть, если Вы с помощью оператора **Define** задали некоторой величине идентификатор FOO, то в тексте программы можно использовать и Foo, и foo. В операторе **Define** нельзя применять для имен слова, используемые как ключевые, например, **Set** или **Create**. Список "запретных" слов приведен в разделе **Оператор Dim на стр. 43**.

**Примеры**

Оператор Define делает Вашу программу более понятной, т. к. Вы можете задать осмысленные имена константам. Например, в программе используется число "пи", которое примерно равно 3.141593. Вы можете присвоить этой константе имя PI и использовать его в тексте программы. Для этого в начале программы напишите:

## Функция DeformatNumber\$( )

---

```
Define PI 3.141593
```

Теперь Вы можете использовать слово PI вместо того, чтобы каждый раз набивать цифры 3.141593.

В операторе **Define** можно использовать кавычки. Например:

```
Define FILE_NAME "World.tab"
```

Следующий оператор **Define** входит в состав файла стандартных определений MAPBASIC.DEF. Выполнение этого **Define**-определения приводит к открытию пустого окна "Сообщение":

```
Define CLS Print Chr$(12)
```

---

## Функция DeformatNumber\$( )

### Назначение

Очищает строку, представляющую число, от форматирующих символов.

### Синтаксис

```
DeformatNumber$ ( numeric_string )
```

*numeric\_string* – строковая величина, представляющая число, такое как "12,345,678"

### Возвращаемая величина

Строка

### Описание

Функция возвращает строку, представляющую число. При этом из исходного значения удаляются разделители тысяч, если они были в значении параметра *numeric\_string*. Также заменяется знак отделения десятичной части на точку, даже если в исходной строке он не точка.

## Примеры

В следующем примере **Функция Val( )** используется для превращения строки в число. Перед тем, как выполнить **Функция Val( )**, в этом примере используется **Функция DeformatNumber\$( )**, удаляющая запятые из строки, потому что **Функция Val( )** воспринимает и разделители тысяч как просто текст.

```
Dim s_number As String
Dim f_value As Float
s_number = "1,222,333.4"
s_number = DeformatNumber$(s_number) ' теперь переменная s_number имеет '
значение: "1222333.4"
f_value = Val(s_number)
Print f_value
```

См. также:

**Функция FormatNumber\$( )**, **Функция Val( )**

## Оператор Delete

### Назначение

Удаляет один или более графических объектов. Или же удаляет одну или более строк из таблицы.

### Синтаксис

```
Delete [ Object ] From table [ Where Rowid = id_number ]
```

*table* – имя открытой таблицы;

*id\_number* – номер строки, целое число от 1 и более.

### Описание

Оператор **Delete** удаляет графический объект или всю запись, соответствующую этому объекту в таблице.

По умолчанию **Delete** удаляется все строки таблицы. Если в операторе указано ключевое слово **Object**, то, удалив графические объекты, MapBasic не удалит записи, к которым эти объекты были присоединены.

Можно сделать так, чтобы оператор **Delete** воздействовал только на избранные строки, а не на все. Для этого задается предложение **Where Rowid =...**, в котором будет указано, какие строки обрабатываются (или удаляются) оператором **Delete**.

Следует отметить разницу между тем, как действует оператор **Delete Object From** и **Оператор Drop Map**. Оператор **Delete Object From** действует только с объектами или записями и не влияет на структуру таблицы. Оператор **Drop Map** меняет структуру таблицы, и может случиться так, что Вы не сможете после его применения добавить графические объекты в таблицу.

## Примеры

В этом примере оператор **Delete** удаляет все записи из таблицы CLIENTS.TAB. Сама таблица не удаляется, а становится пустой, похожей на ту таблицу, которая создается в MapInfo командой **Файл > Новая таблица**.

```
Open Table "clients"  
Delete From clients  
Table clients
```

Оператор **Delete** удаляет только графический объект, присоединенный к десятой записи таблицы.

```
Open Table "clients"  
Delete Object From clients Where Rowid = 10  
Table clients
```

**См. также:**

**Оператор Drop Map, Оператор Insert**

---

## Оператор Dialog

### Назначение

Создает новое диалоговое окно.

### Предупреждение

Оператор **Dialog** не может быть использован в окне MapBasic.

### Синтаксис

#### Dialog

```
[ Title title ]  
[ Width w ] [ Height h ] [ Position x, y ]  
[ Calling handler ]  
Control control_clause  
[ Control control_clause... ]
```

*title* – строковая величина, которая помещается в строку заголовка диалогового окна;

*h* – задает высоту диалогового окна в специальных единицах измерения высоты диалога (высота одного символа в диалоге равна 8 единицам);

*w* – задает ширину диалогового окна в специальных единицах измерения ширины диалога (ширина одного символа в диалоге равна 4 единицам);

*x, y* – координаты верхнего левого угла диалога в пикселах относительно верхнего левого угла рабочей области окна MapInfo Professional; если не задано предложение **Position**, диалог будет расположен в середине окна.

*handler* – имя процедуры, которая выполняется перед выводом диалога на экран; обычно в эти процедуры помещается **Оператор Alter Control** (или несколько операторов).

Каждый параметр *control\_clause* может быть одной из конструкций, которая начинается одним из следующих ключевых слов:

- Кнопка (Button)
- Кнопка ОК (OKButton)
- Кнопка "Отмена" (CancelButton)

- EditText - окошко ввода текста
- StaticText - неизменяемый текст
- Всплывающее меню (PopupMenu)
- CheckBox - флажки
- Список множественного выбора (MultiListBox)
- Группа (GroupBox)
- Кнопки-переключатели (RadioGroup)
- PenPicker
- BrushPicker
- FontPicker
- SymbolPicker
- Список (ListBox)

Каждому управляющему элементу посвящен отдельный раздел этого настоящего Справочника (например, элементу CheckBox посвящен раздел [Предложение Control CheckBox on page 162](#); элементу Picker - раздел [Предложения Control PenPicker/BrushPicker/SymbolPicker/FontPicker](#) и т.п.).

С помощью этих ключевых слов можно создавать следующие управляющие элементы диалога:

- Button / OKButton / CancelButton - кнопки
- CheckBox - флажки
- Группа (GroupBox)
- Кнопки-переключатели (RadioGroup)
- EditText - окошко ввода текста
- StaticText - неизменяемый текст
- PenPicker / BrushPicker / SymbolPicker / FontPicker - кнопки стилизации
- ListBox / MultiListBox - окошки списка
- Всплывающее меню (PopupMenu)

## Описание

Оператор **Dialog** создает диалоговое окно произвольного вида для организации диалога программы с пользователем. Это так называемый модальный диалог; другими словами, пользователь должен закрыть диалог (например, нажав на кнопку **OK** или **Cancel**) для того, чтобы продолжить работу с MapInfo. Более подробно о создании диалоговых окон написано в *Руководстве пользователя MapBasic*.

Все, что находится внутри диалогового окна, называется управляющими элементами диалога или элементами. В каждом диалоге должен быть задан хотя бы один управляющий элемент. Каждый управляющий элемент описывается в своем разделе Справочника (например, элементу CheckBox посвящен раздел [Предложение Control CheckBox on page 162](#)). Обычно, каждый диалог должен содержать определения элементов OKButton и/или CancelButton чтобы его можно было хотя бы закрыть.

Оператор **Dialog** позволяет создавать Вам диалоговые окна произвольного вида. Если Вы хотите использовать стандартные диалоги в своей программе (например, диалог **Файл > Открыть**), используйте следующие операторы и функции: **Функция Ask( )**, **Оператор Note**, **Оператор ProgressBar**, **Функция FileOpenDlg( )**, **Функция FileSaveAsDlg( )** или **Функция GetSeamlessSheet( )**.

Информацию об основных концепциях построения диалогового окна в MapBasic Вы можете найти в *Руководстве пользователя MapBasic*.

### Размер и расположение элементов диалога

В операторе **Dialog** размер и расположение элемента диалога измеряются в долях шрифта диалога. Каждая единица измерения ширины в диалоге равна одной четвертой ширины символа (координата x) и каждая единица измерения высоты в диалоге – одной восьмой высоты символа (координата y). Так, если элемент имеет ширину 40 и высоту 40, то это значит, что в элемент можно уместить слово в десять букв и список в пять строк. Центром координат для определения места элементов диалога взят верхний левый угол окна диалога. Он имеет координаты (0,0).

Предложения **Position**, **Height** и **Width** не являются обязательными. Если Вы их опустили, то MapBasic разместит элементы по умолчанию в порядке следования соответствующих предложений Control в операторе.

### Заккрытие диалога

Диалоговое окно, открытое MapBasic оператором **Dialog**, можно закрыть одним из четырех способов:

- пользователь нажимает кнопку элемента OkButton (если этот элемент присутствует в диалоге);
- пользователь нажимает кнопку элемента CancelButton (если этот элемент присутствует в диалоге);
- пользователь указывает на элемент диалога, обработчик которого выполняет **Оператор Dialog Remove**;
- пользователь отменяет диалог, используя системное меню диалогового окна или клавишу ESC.

Чтобы диалог оставался на экране после нажатия кнопок **ОК** или **Cancel**, Вы должны создать специальные процедуры-обработчики нажатия этих кнопок и указать эти процедуры в описаниях элементов OkButton или CancelButton. Для достижения нужного "эффекта присутствия" в этих обработчиках должен использоваться **Оператор Dialog Preserve**.

### Чтение введенных значений

После оператора **Dialog** часто используется **CommandInfo( )** функция, чтобы определить, как был закрыт диалог пользователем, кнопкой **ОК** или **Cancel**. Если пользователь нажал на кнопку **ОК**, то следующая функция вернет значение TRUE:

```
CommandInfo (CMD_INFO_DLG_OK)
```

Для определения, какие значения ввел пользователь в окошки диалога и какие режимы выбрал, есть два способа: в состав оператора включается **Dialog** предложение Into или используется **Функция ReadControlValue( )** из процедуры-обработчика.

Если используется предложение **Into**, а диалог был закрыт кнопкой OK, MapInfo Professional присвоит финальные значения из элементов диалога соответствующим переменным.

**Внимание:** MapInfo Professional обновляет переменную после нажатия на **OK**. Также, MapInfo Professional обновляет переменную после закрытия диалога.

Чтобы прочесть значения из диалога, пока он не закрыт, можно только из процедуры-обработчика, используется **Функция ReadControlValue( )**.

### Задание клавишных сокращений для элементов диалога

Если прикладная программа выполняется в среде MapInfo для Windows, то элементам диалога могут быть назначены клавишные сокращения, которые позволяют пользователю обращаться к элементам диалога с клавиатуры.

В тексте заголовка перед нужным символом надо поставить знак амперсанда (&). Символ для клавишного сокращения выбирается из заголовков и подписей элементов, которые задаются элементам в предложении **Title**. Например, следующая конструкция создает элемент диалога Button с клавишным сокращением ALT+B:

```
Control ButtonTitle "&Восстановить"
```

Знак & (амперсанд), задаваемый в предложении **Title**, на экране не отображается. Если Вы хотите вывести сам знак амперсанда в заголовок элемента, (например, чтобы создать следующую подпись элемента "Вернуть & обработать"), то используйте в предложении **Title** его два амперсанда подряд:

```
Title "&Найти && заменить"
```

Если Вы задаете элемент StaticText сразу до или после элемента EditText, то определенное клавишное сокращение в тексте StaticText можно использовать для перехода в окошко элемента EditText.

### Порядок элементов диалога для клавиши TAB

Для изменения фокуса (точки ввода) в диалоге при помощи клавиатуры пользователь может использовать клавишу TAB. Фокус перемещается от элемента к элементу при каждой нажатии на TAB.

Порядок перемещения фокуса задается последовательностью, в которой соответствующие предложения **Control** следуют в операторе **Dialog**. Так, если фокус находится на третьем по порядку элементе, то нажатие на TAB перемещает фокус на следующий за ним в описании, четвертый элемент и т.д.. Изменить этот порядок можно, переставив в описании диалога предложения **Control**.

### Примеры

Следующий оператор строит диалог с окошком для ввода текста пользователем. Поскольку ни одно из предложений **Control** не использует подпредложения **Position**, MapBasic размещает элементы управления автоматически.

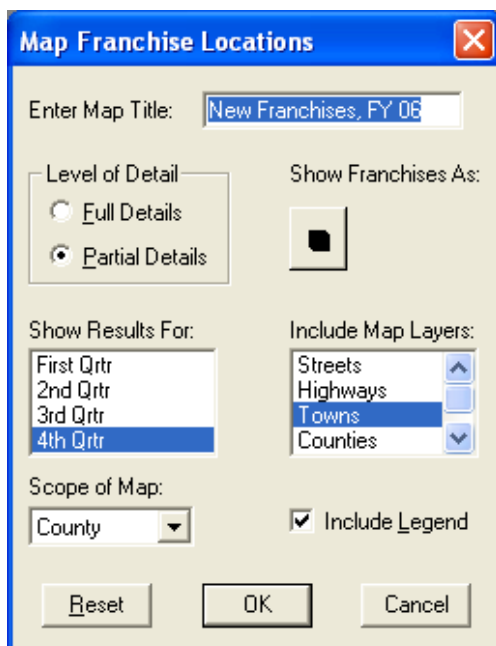
```
DialogTitle "Поиск строки" Control StaticTextTitle "Введите строку для
поиска:" Control EditText Value gs_searchfor 'это строковая глобальная
переменнаяInto gs_searchforControl OKButtonControl CancelButton
If CommandInfo(CMD_INFO_DLG_OK) Then ' ...если пользователь нажал на
кнопку "OK", то переменной "gs_searchfor" присваивается текст, который
ввел пользователь.
End If
```

В следующем фрагменте Вы сможете найти применение элементов диалога всех типов.

```
Include "mapbasic.def"
Declare Sub reset_sub 'resets dialog to default settings
Declare Sub ok_sub ' notes values when user clicks OK.
Include "mapbasic.def"Declare Sub reset_sub 'установка стандартных
значенийDeclare Sub ok_sub ' подпрограмма, подтверждающая нажатие кнопки
OK.Declare Sub MainSub Main Dim s_title As String 'заголовок картыDim
l_showlegend As Logical 'TRUE означает показ легендыDim i_details As
SmallInt '1 = за 1-ый квартал,... 4 = за 4-ый и т.д.Dim i_quarter As
SmallInt '1 = карта города;2 = области и т.д. Dim i_scope As SmallInt
'1=город, 2=область и т.д.Dim sym_variable As Symbol Dialog Title "Карта
торговых точек" Control StaticText Title "Заголовок карты:"Position 5, 10
Control EditTextValue "Торговые точки" Into s_title ID 1 Position 65, 8
Width 90
    Control GroupBox Title "Уровень детализации" Position 5, 30 Width 70
Height 40 Control RadioGroupTitle "&Всё;В&ыборчно" Value 2 Into i_details
    ID 2 Position 12, 42 Width 60 Control StaticText Title "Показывать
пункты как:" Position 95, 30
    Control SymbolPickerPosition 95, 45 Into sym_variable ID 3 Control
StaticText Title "Показать результаты для:" Position 5, 80 Control
ListBox Title "1-ого квартала;2-ого квартала;3-его квартала; 4-ого
квартала" Value 4 Into i_quarterID 4 Position 5, 90 Width 65 Height 35
    Control StaticText Title "Показать слои:" Position 95, 80 Control
MultiListBox Title "Улицы;Шоссе;Города;Области;Территории"Value 3ID 5
Position 95, 90 Width 65 Height 35
    Control StaticText Title "Охват карты:" Position 5, 130 Control
PopupMenu Title "Город;Область;Территория;Регион;Вся страна" Value 2 Into
i_scope ID 6 Position 5, 140
    Control CheckBox Title "Показывать &Легенду" Into l_showlegend ID 7
Position 95, 140
    Control Button Title "&Восстановить" Calling reset_sub Position 10,
165
    Control OKButton Position 65, 165 Calling ok_sub Control CancelButton
Position 120, 165
    If CommandInfo(CMD_INFO_DLG_OK) Then' ... пользователь выбирает кнопку
"OK"
        Else ' ... пользователь выбирает кнопку "Отмена".
        End If
    End Sub
Sub reset_sub' восстановление начальных значений в диалоге'с помощью
операторов Alter Control.
End Sub
Sub ok_sub' сюда помещается код обработки нажатия клавиши "OK"End Sub
```

Результатом этой процедуры будет следующий диалог:





См. также:

Оператор **Alter Control**, Функция **Ask( )**, Оператор **Dialog Preserve**, Оператор **Dialog Remove**, Функция **FileOpenDlg( )**, Функция **FileSaveAsDlg( )**, Оператор **Note**, Функция **ReadControlValue( )**

## Оператор Dialog Preserve

### Назначение

Открывает диалог снова после нажатия клавиш типа **OK** или **Cancel**.

### Синтаксис

`Dialog Preserve`

### Предупреждение

Этот оператор может быть использован только в процедуре обработчика кнопок типа `OKButton` и `CancelButton`.

Этот оператор не может быть использован в окне `MapBasic`.

### Описание

Оператор **Dialog Preserve** позволяет приостановить закрытие диалога, который создает программе **Оператор Dialog**, нажатием на кнопки типа `OkButton` или `CancelButton`.

Оператор **Dialog Preserve** позволяет Вам подтверждать "закрытие" диалога. Например, пользователь нажимает на кнопку **Отмена (Cancel)**. Обработчик кнопки вызывает диалог, спрашивающий: "Вы согласны отменить все изменения?" Если пользователь выберет кнопку "Нет", обработчик восстановит первоначальный диалог. Такого поведения можно добиться, включив оператор **Dialog Preserve** в тело процедуры-обработчика нажатия на кнопку "Нет", задаваемую элементом `CancelButton`.

### Пример:

Следующий отрывок может быть процедурой-обработчиком кнопки `CancelButton`.

```
Sub confirm_cancel If Ask("Изменения будут утеряны. Продолжить?", "Да",  
"Нет") = FALSE Then Dialog Preserve End If End Sub
```

### См. также:

[Оператор Alter Control](#), [Оператор Dialog](#), [Оператор Dialog Remove](#), [Функция ReadControlValue\( \)](#)

---

## Оператор Dialog Remove

### Назначение

Закрывает диалог, созданный оператором `Dialog`.

### Синтаксис

`Dialog Remove`

### Предупреждение

Оператор работает только в процедуре обработчика элемента диалога. Этот оператор не может быть использован в окне `MapBasic`.

### Описание

Оператор **Dialog Remove** закрывает диалог, который создал последний **Оператор Dialog**. Автоматически диалоговое окно закрывается по нажатию на кнопки `OkButton` или `CancelButton`. Оператор **Dialog Remove**, включенный в тело процедуры-обработчика элемента диалога, удаляет диалог с экрана, не дожидаясь того, что пользователь нажмет **ОК** или **Отмена**. Это полезно, если, например, если в диалоге Вы выбрали двойным указанием мышкой из окошка списка какую-нибудь строчку и желаете, чтобы диалог после этого закрывался автоматически.

**Внимание:** **Dialog Remove** посылает диалогу сигнал на закрытие после того, как процедура-обработчик закончит свою работу. (То есть диалог не закрывается немедленно после выполнения этого оператора в процедуре-обработчике).

**Пример:**

Следующая процедура – часть программы NIEWS.MB. Она исполняет роль обработчика спискового элемента (ListBox) диалога "Именованные Виды" (Named Views). Если пользователь указывает в список, используя одно нажатие на клавишу мышки, обработчик оставляет возможность закрыть диалог при помощи кнопок. Если пользователь указывает в список, используя двойное нажатие на клавишу мышки, обработчик использует оператор **Dialog Remove** для закрытия диалогового окна.

**Внимание:** MapInfo Professional вызывает этот обработчик и тогда, когда клавиша мышки нажимается один раз, и тогда, когда пользователь использует двойное нажатие.

```
Sub listbox_handler Dim i As SmallInt Alter Control 2 Enable Alter
Control 3 Enable If CommandInfo(CMD_INFO_DLG_DBL) = TRUE Then ' ' ... в
списке было использовано двойное указание
,
    i = ReadControlValue(1)
    Dialog Remove
    Call go_to_view(i)
End If
End Sub
```

**См. также:**

**Оператор Alter Control, Оператор Dialog, Оператор Dialog Preserve, Функция ReadControlValue( )**

---

## Оператор Dim

**Назначение**

Объявляет тип одной или более переменных.

**Предупреждение**

Если оператор **Dim** используется в окне MapBasic, то один оператор **Dim** может объявить тип только одной переменной. Оператор **Dim** может задавать несколько переменных только из прикладной программы MapBasic. В окне MapBasic нельзя также объявлять массивы переменных.

**Синтаксис**

```
Dim var_name [ , var_name ... ] As var_type
    [ , var_name [ , var_name ... ] As var_type ... ]
```

*var\_name* – имя объявляемой переменной;

*var\_type* – тип для переменной.

**Описание**

Оператор **Dim** объявляет тип одной или более переменных. В следующей таблице перечислены все стандартные типы переменных, которые Вы можете объявить в операторе **Dim**.

**Типы и описания переменных**

| Тип переменной                 | Описание  |
|--------------------------------|---|
| Целое число типа SmallInt.     | Число от -32767 до 32767 включительно; используется два байта.  |
| Целое число типа Integer.      | Число от -2147483647 до +2147483647 включительно; используется четыре байта..   |
| Вещественное число типа Float. | Число с плавающей запятой; занимает восемь байт в формате IEEE.   |
| Строка                         | Символьная строка не более 32767 байт.  |
| String * length                | Символьная строка фиксированной длины (здесь <i>length</i> задает длину строки до 32767 байт). Заранее предполагается, что это строки пробелов. |
| Логическое                     | TRUE или FALSE, 1 или 0; используется два байта.  |
| Дата типа Date                 | Дата в формате MM/DD/YYYY, используется четыре байта: два байта – год, один байт – месяц, один байт – день.                                     |
| Объект                         | Графический объект (Точечный, Регион, Линия, Полилиния, Дуга, Прямоугольник, Сглаженный Прямоугольник, Эллипс, Текстовый или Рамка).            |
| Псевдоним типа Alias           | Имя колонки.  |
| Pen                            | Установка стиля линии.  |
| Brush                          | Установка стиля штриха.   |
| Шрифт                          | Установка стиля шрифта.   |
| Символ                         | Установка стиля символа для точечного объекта.  |

Переменные, используемые в процедурах и функциях, должны быть объявлены операторами **Dim**, до обращения к ним. Поэтому операторы **Dim** располагаются в первых строках процедуры или функции.

Если оператор **Dim** использует **Оператор Sub...End Sub** или **Оператор Function...End Function**, то объявленные таким образом переменные являются локальными. К локальным переменным можно обращаться только из процедуры или функции, в которых эти переменные были объявлены операторами **Dim**.

Если оператор **Dim** используется вне тела процедуры или функции, то объявляются переменные на уровне программного модуля. Значения таких переменных могут использоваться во всех процедурах и функциях одного программного модуля, который участвует в сборке проекта.

Для объявления глобальных переменных используется **Оператор Global**. Значение глобальной переменной доступно всем процедурам и функциям всех модулей проекта.

### Объявление нескольких переменных в одном операторе

Один оператор **Dim** позволяет объявлять сразу несколько переменных одного типа. Их имена разделяются запятыми. Можно также в одном операторе **Dim** объединить объявление нескольких переменных разного типа. Например:

```
Dim jointer, i_min, i_max As Integer, s_name As String
```

### Массивы

MapBasic поддерживает одномерные массивы переменных. Если после имени переменной стоит пара скобок, то это имя понимается как имя массива. При объявлении типа массива переменных Вы можете также задать размерность массива.

Например, следующему массиву вещественных переменных задана размерность десять:

```
Dim f_stats(10) As Float
f_stats(1) = 17.23
```

Число в скобках во второй строке примера является индексом массива, и задает порядковый номер элемента массива. Первый элемент массива имеет порядковый номер один.

Чтобы изменить размерность массива в ходе программы, используйте **Оператор ReDim**. Для определения текущей размерности массива, используйте **Функция UBound( )**. Если оператор **Dim** объявляет массив с пустыми скобками, то массиву присваивается нулевая размерность. При этом массив не занимает в памяти места. Для его использования в нужный момент Вы всегда можете задать ему новую размерность, выполнив **Оператор ReDim**. В 32-битной версии Windows массивы могут иметь размерность до 32 767 элементов.

### Строковые переменные

Длина строковой переменной не должна превышать 32 К. Однако, символьная строка, участвующая в операции присваивания, не должна превышать 256 символов. В следующем примере происходит присвоение строки с определенной длиной строковой переменной:

```
Dim status As Stringstatus = "Это строковая константа... "
```

Такие конструкции корректны и допустимы всегда, если длина строки справа от знака равенства не превышает 256 символов.

MapBasic, подобно другим BASIC-языкам, автоматически заполняет строковые переменные фиксированной длины пробелами. Другими словами, когда Вы объявляете строковую переменную в 10 байт, а в нее помещаете строку из пяти символов, то оставшиеся пять позиций будут заполнены пробелами. (Это может быть полезно при формировании таблицы из строк текста.)

Строковые переменные свободной длины при объявлении ничем не заполняются (вернее, значение переменной типа String сразу после выполнения оператора объявления равно пустому значению). Это различие может порождать множество труднораспознаваемых ошибок: В следующем фрагменте **Оператор If...Then** определяет неравенство на двух строк, кажущихся равными:

```
Dim s_var_len As StringDim s_fixed_len As String * 10s_var_len =
"тест"s_fixed_len = "тест" If s_var_len = s_fixed_len Then Note "строки
равны" ' этого никогда не случитсяElseNote "строки не равны" 'это
случится обязательноEnd If
```

### Ограничения при выборе имени для переменной

В именах переменных большие и маленькие буквы не различаются. Так, если оператор **Dim** объявил переменную **abc**, то в дальнейшем к ней можно обращаться и **abc**, и **ABC**, и **Abc**.

Имя переменной может состоять не более чем из 31 символа, которыми могут быть буквы латинского алфавита, цифры и знак подчеркивания. ( \_ ). Имена переменных могут также использовать следующие знаки пунктуации: \$, %, &, !, # и @, но только в конце имени. Имя переменной также не может начинаться с цифры.

Для имен переменных Вы также не можете использовать слова, которые использует язык MapBasic как ключевые, например, такие как **Open**, **Close**, **Set** и **Do**. Если Вы объявили переменную под именем **Set**, то при компиляции MapBasic сгенерирует ошибку В следующей таблице приведены слова, которые не должны использоваться в качестве имен переменных.

|                   |                 |                     |                        |
|-------------------|-----------------|---------------------|------------------------|
| <b>Add</b>        | <b>Alter</b>    | <b>Browse</b>       | <b>Call</b>            |
| <b>Close</b>      |                 | <b>Create</b>       | <b>DDE</b>             |
| <b>DDEExecute</b> | <b>DDEPoke</b>  | <b>DDETerminate</b> | <b>DDETerminateAll</b> |
| <b>Declare</b>    | <b>Delete</b>   | <b>Dialog</b>       | <b>Dim</b>             |
| <b>Do</b>         | <b>Drop</b>     | <b>Else</b>         | <b>Elself</b>          |
| <b>End</b>        | <b>Error</b>    | <b>Event</b>        | <b>Exit</b>            |
| <b>Export</b>     | <b>Fetch</b>    | <b>Find</b>         | <b>For</b>             |
| <b>Функция</b>    | <b>Get</b>      | <b>Global</b>       | <b>Goto</b>            |
| <b>Graph</b>      | <b>If</b>       | <b>Import</b>       | <b>Insert</b>          |
| <b>Layout</b>     | <b>Map</b>      | <b>Меню (Menu)</b>  | <b>Note</b>            |
| <b>Objects</b>    | <b>OnError</b>  | <b>Open</b>         | <b>Pack</b>            |
| <b>Print</b>      | <b>PrintWin</b> | <b>ProgressBar</b>  | <b>Put</b>             |
| <b>ReDim</b>      | <b>Register</b> | <b>Reload</b>       | <b>Remove</b>          |
| <b>Rename</b>     | <b>Resume</b>   | <b>Rollback</b>     | <b>Run</b>             |

|       |           |       |     |
|-------|-----------|-------|-----|
| Save  | Seek      | Выбор | Set |
| Shade | StatusBar | Stop  | Sub |
| Тип   | Update    | While |     |

В некоторых BASIC-языках тип переменной диктует написание ее имени. Например, если переменная имеет имя со знаком доллара в конце (LastName\$), то она понимается как строковая. В языке MapBasic это не работает, тип переменной надо объявлять явно, оператором **Dim**.

### Начальные значения переменных

При объявлении численных переменных MapBasic присваивает им начальное значение 0 (ноль). Строковые переменные неопределенной длины имеют начальное значение пустая строка (""). Строковые переменные фиксированной длины заполняются пробелами.

Объектным переменным и переменным типа Pen, Brush, Font и Symbol при объявлении не присваивается начальных значений. Перед использованием этих переменных Вы должны их инициализировать.

### Пример:

```
' Создадим сложный тип данных Person' используя оператор TypeType Person
Name As String Age As IntegerPhone As StringEnd Type' Следующий оператор
Dim объявляет переменную типа PersonDim customer As Person ' Этот оператор
Dim объявляет массив переменных типа Person :Dim users(10) As Person' Этот
оператор Dim объявляет целочисленную переменную "counter"' массив
целочисленных переменных "counters" :Dim counter, counters(10) As Integer'
Этот оператор присваивает элементу "Name" первого значения' массива
"users" следующее значение :users(1).Name = "Поликарп"
```

### См. также:

[Оператор Global](#), [Оператор ReDim](#), [Оператор Type](#), [Функция UBound\( \)](#)

---

## Функция Distance( )

### Назначение

Возвращает расстояние между двумя точками.

### Синтаксис

**Distance** ( *x1*, *y1*, *x2*, *y2*, *unit\_name* )

*x1* и *x2* - координаты по оси x (например, долгота).

*y1* и *y2* - координаты по оси y (например, долгота).

*unit\_name* - строка, определяющая единицы измерения расстояния (к примеру, "км.").



## Возвращаемая величина

Вещественное число типа Float.

## Описание

Функция **Distance( )** вычисляет расстояние между двумя определенными точками.

Она возвращает значение в указанных единицах *unit\_name*; например, чтобы получить расстояние в милях, задайте "mi" в качестве *unit\_name*. Список доступных единиц измерения смотрите в [Оператор Set Distance Units on page 633](#).

Координаты по осям x и y должны быть заданы в текущей системе координат MapBasic. Координаты X и Y понимаются MapBasic относительно заданной системы координат. Если система координат не объявлялась, то используется система широта/долгота. Вы можете установить систему координат по умолчанию при помощи [Оператор Set CoordSys](#).

Если текущей системой координат является система координат Земли, то функция **Distance( )** возвращает расстояние между двумя точками по дуге большого земного сечения. Расстояние вычисляется по большой окружности на сфере (большая окружность получается в результате сечения земного шара плоскостью, заданной этими двумя точками и центром Земного шара).

В большинстве случаев MapInfo Professional проводит либо декартовы, либо сферические вычисления. Обычно выполняются сферические вычисления; если координатная система - план-схема, то выполняются декартовы вычисления.

## Пример:

```
Dim dist, start_x, start_y, end_x, end_y As Float
Open Table "cities"
Fetch First From cities
start_x = CentroidX(cities.obj)
start_y = CentroidY(cities.obj)
Fetch Next From cities
end_x = CentroidX(cities.obj)
end_y = CentroidY(cities.obj)
dist = Distance(start_x, start_y, end_x, end_y, "mi")
```

## См. также:

[Функция Area\( \)](#), [Функция ObjectLen\( \)](#), [Оператор Set CoordSys](#), [Оператор Set Distance Units](#)

---

## Оператор Do Case...End Case

### Назначение

Выполняет одну из групп операторов, условия для которой истинны.

### Предупреждение

Оператор **Do Case** не может быть выполнен в окне MapBasic.

### Синтаксис

```
Do Case do_expr
    Case case_expr [ , case_expr ]
        statement_list
    [ Case ... ]
    [ Case Else
        statement_list ]
End Case
```

*do\_expr* – выражение;

*case\_expr* – выражение, результат которого будет сравниваться со значением *do\_expr*;

*statement\_list* – группа операторов.

### Описание

Оператор **Do Case** действует так же, как и **Оператор If...Then**. **Do Case** проверяет истинность определенных выражений и, в зависимости от результата, выполняет одну из групп операторов. Оператор **Do Case** в языке MapBasic аналогичен оператору Select Case в языке BASIC. (Так как в MapBasic есть другой **Оператор Select**, имя оператора изменено для того, чтобы не возникало путаницы).

Выполняя оператор **Do Case**, MapBasic вычисляет выражение **Case case\_expr**. Если результат **Case case\_expr** равен результату вычисления выражения *do\_expr*, то MapBasic выполнит группу операторов *statement\_list*, которая расположена за первым словом Case и после этого управление программой передается оператору, следующему после ключевых слов **End Case**.

Если ни одно из выражений **Case case\_expr** не дает того же результата, что *do\_expr*, MapBasic переходит к следующему выражению **Case case\_expr**. Так MapBasic перебирает все выражения **Case case\_expr** пока либо не добьется совпадения, либо пока не переберет все.

MapBasic выполняет операторы *statement\_list* из оператора **Do Case** ровно один раз или ни разу. Так, обнаружив совпадение, MapBasic выполняет операторы *statement\_list* и сразу переходит на первый оператор после **End Case**.

Если же ни одно выражение *case\_expr* не дает того же результата, что выражение *do\_expr*, то операторы из списка *statement\_list* не выполняются. Однако, в оператор **Do Case** можно включить предложение **Case Else**, и тогда если ни одно из выражений **Case case\_expr** не даст совпадения с условием, то MapBasic выполнит операторы из раздела **Case Else**.

Заметим, что оператор **Do Case** такой формы:

```
Do Case expr1
    Case expr2
        statement_list1
    Case expr3, expr4
        statement_list2
    Case Else
        statement_list3
End Case
```

работает так же, как **Оператор If...Then** следующей конструкции:

```

If expr1 = expr2 Then
    statement_list1
ElseIf expr1 = expr3 Or expr1 = expr4 Then
    statement_list2
Else
    statement_list3
End If

```

### Пример:

В примере выбираются текстовые строки: "Первый квартал", "Второй квартал" и т.п., в зависимости от текущего месяца.

```

Dim cur_month As Integer, msg As String
cur_month = Month( CurDate( ) )
Do
    Case cur_monthCase 1, 2, 3msg = "Первый квартал"
    Case 4, 5, 6msg = "Второй квартал"
    Case 7, 8, 9msg = "Третий квартал"
    Case Elsemsg = "Четвертый квартал"
End Case

```

См. также:

**Оператор If...Then**

## Оператор Do...Loop

### Назначение

Выполняет последовательность действий по циклу до тех пор, пока выполнится (или, наоборот, не выполнится) некое условие.

### Предупреждение

Оператор **Do Loop** не может быть выполнен в окне MapBasic.

### Синтаксис 1

```

Do
    statement_list
Loop [ { Until | While } condition ]

```

### Синтаксис 2

```

Do [ { Until | While } condition ]
    statement_list
Loop

```

*statement\_list* – группа операторов для одного шага цикла;

*condition* – условие, задающее предложение или окончание цикла.

### Описание

Оператор **Do...Loop** задает цикл. Цикл **Do...Loop** повторяет группу операторов *statement\_list* до тех пор, пока предложение **While** сохраняет значение TRUE (или, наоборот, цикл повторяет операторы *statement\_list* пока условие **Until** не примет значение TRUE).

Если цикл **Do...Loop** не содержит слов **Until / While**, то цикл повторяется бесконечно. В этом случае управление выполнением цикла ведется из операторов самого цикла, используя **Оператор Goto** или **Оператор Exit Do**, которые прекращают цикл. Выполнив **Оператор Exit Do**, MapBasic останавливает цикл **Do...Loop** немедленно, не взирая на условия, заданные предложениями **Until / While**, и передает управление на первый оператор после предложения **Loop**.

Предложения **Until / While** могут находиться как после слова **Do**, так и после слова **Loop**. В зависимости от положения слов **Until / While**, проверка выполнения условия происходит до или после выполнения операторов *statement\_list*. Это важно для первой итерации. Следующая конструкция:

```
Do
    statement_list
Loop While condition
```

выполняет операторы *statement\_list* и потом вычисляет условие *condition*. Если *condition* равно TRUE, MapBasic вновь выполняет *statement\_list* пока условие *condition* не станет равным FALSE. Так, цикл **Do...Loop** в представленном выше варианте, выполняет набор операторов *statement\_list* ровно один раз.

В следующей конструкции оператор **Do...Loop** выполнит операторы набора *statement\_list*, если условие *condition* равно TRUE.

```
Do While condition
    statement_list
Loop
```

### Пример:

В цикле **Do...Loop** читаются первые десять записей таблицы WORLD:

```
Dim sum As Float, counter As Integer
Open Table "world"
Fetch First From world
counter = 1
Do
    sum = sum + world.population
    Fetch Next From world
    counter = counter + 1
Loop While counter <= 10
```

См. также:

**Оператор Exit Do, Оператор For...Next**

---

## Оператор Drop Index

### Назначение

Удаляет индекс из таблицы.

## Синтаксис

**Drop Index** *table* ( *column* )

*table* – имя открытой таблицы;

*column* – имя колонки в таблице.

## Описание

Оператор **Drop Index** отменяет индексирование колонки в открытой таблице. Удаление индекса освобождает место на диске, где расположена таблица. (Для создания индекса вновь используется **Оператор Create Index**.)

**Внимание:** MapInfo не может отменить индексирование в таблице с несохраненными изменениями. Для сохранения изменений используется **Оператор Commit Table**.

Используйте оператор **Drop Index** для удаления сразу всех графических объектов (точка, линия, регион, окружность и т. п.) из открытой таблицы, изменяя структуру таблицы так, что к ее записям уже не могут присоединяться графические объекты. Оператор **Drop Index** не имеет обратного действия и не требует последующего сохранения изменений на диске. Т. е. не работают команды **Файл > Восстановить** и **Правка > Отменить**. Аналогично, **Оператор Rollback** также не может отменить действие оператора **Drop Index**.

## Пример:

Следующий код отменяет индексирование колонки "Страна" в таблице WORLD:

```
Open Table "world"Drop Index world(Страна)
```

## См. также:

**Оператор Create Index**

## Оператор Drop Map

### Назначение

Удаляет все графические объекты из таблицы. Не может быть применен к связанным таблицам.

### Синтаксис

**Drop Map** *table*

*table* – имя открытой таблицы;

### Описание

После выполнения оператора **Drop Map**, соответствующая таблица более не может быть показана в окне Карты. Все ее графические объекты пропадают.

**Внимание:** Оператор **Drop Map** не имеет обратного действия и не требует последующего сохранения изменений на диске. Вы не можете отменить действие оператора **Drop Map** командами **Файл > Восстановить** или **Правка > Отменить**.

Аналогично, **Оператор Rollback** также не может отменить действие оператора **Drop Map**. Поэтому Вы должны быть очень внимательны, используя оператор **Drop Map**.

После выполнения оператора **Drop Map** соответствующая таблица более не может быть показана в окне Карты; оператор **Drop Map** удаляет графические объекты навсегда и структура таблицы изменяется так, что она уже не может присоединять их. Для изменения структуры таблицы так, чтобы к ней можно было вновь присоединять графические объекты, используйте **Оператор Create Map** (однако, оператор Create Map не восстанавливает графические объекты, уничтоженные оператором **Drop Map**). Оператор **Drop Map** не меняет количества записей в таблице. Оператор **Drop Map** не влияет на данные в записях таблицы, которые можно выводить в окно Списка.

Если Вам надо удалить все графические объекты без изменения структуры таблицы, воспользуйтесь оператором **Delete Object** вместо **Drop Map**.

Оператор **Drop Map** не работает со связанными таблицами.

### Пример:

```
Open Table "clients"  
Drop Map clients
```

### См. также:

**Оператор Create Map, Оператор Create Table, Оператор Delete**

---

## Оператор Drop Table

### Назначение

Удаляет таблицу полностью.

### Синтаксис

```
Drop Table table
```

*table* – имя открытой таблицы;

### Описание

Оператор **Drop Table** используется для удаления таблицы с диска. Таблица при этом должна быть открыта.

Отметим, что если таблица основана на файле базы данных или электронной таблице, то оператор **Drop Table** удалит их вместе с файлами-компонентами, которые добавляет к ним MapInfo Professional. Другими словами, **Drop Table** может удалять файлы, которые не создавались непосредственно в среде MapInfo Professional.

Оператор **Drop Table** имеет непосредственный эффект и не имеет обратного действия. После выполнения **Drop Table** не работают команды **Файл > Восстановить** или **Правка > Отменить**. Аналогично, **Оператор Rollback** не восстанавливает таблицу после применения **Drop Table**. С оператором **Drop Table** нужно быть осторожным!

**Внимание:** Многие операции с таблицами в MapInfo помещают результат во временные таблицы (например, "Query1" или "Запрос1"). Временная таблица удаляется автоматически с закрытием сеанса работы MapInfo и нет необходимости применять оператор **Drop Table** для удаления временной таблицы.

Оператор **Drop Table** нельзя применять для удаления таблиц, которые по природе своей являются представлениями ("view"). Например, таблицы формата StreetInfo (такие, как SF\_STRTS) являются представлениями, то есть представляют на экране комбинацию других таблиц (SF\_STRT1 и SF\_STRT2). Поэтому таблицу SF\_STRTS нельзя удалить оператором **Drop Table**.

### Пример:

```
Open Table "clients"
Drop Table clients
```

**См. также:**

**Оператор Create Table, Оператор Delete, Оператор Kill**

## Оператор End MapInfo

### Назначение

Прекращает выполнение MapInfo Professional.

### Синтаксис

```
End MapInfo [ Interactive ]
```

### Описание

Оператор **End MapInfo** используется для остановки выполнения прикладной программы и завершения сеанса работы MapInfo.

Программа может содержать процедуру-обработчик завершения работы, для которой в MapBasic зарезервировано имя **EndHandler**. Если программа выполняет оператор **End MapInfo**, то MapInfo Professional автоматически выполнит перед закрытием все процедуры **EndHandler**. Более подробную информацию см. в разделе **Процедура EndHandler на стр. 57**.

Если оператор **End MapInfo** выполнен тогда, когда были открыты таблицы и в них были произведены изменения, не сохраненные в файлах, MapInfo выдаст запрос о сохранении этих изменений.

Если оператор использует ключевое слово **Interactive** и если были открыты окна Карт с объектами на Косметическом слое или с не сохраненными тематическими объектами, то MapInfo выдаст пользователю сообщения, предлагающие сохранить эти объекты. Однако,

если в MapInfo включен режим автоматического сохранения Рабочего Набора MAPINFOW.WOR перед закрытием, то сообщения выдаваться не будут. Если слово **Interactive** опущено, то предложения сохранить косметические и тематические объекты выдаваться не будут.

Для остановки MapBasic-программы без завершения сеанса работы MapInfo, используйте **Оператор End Program**.

**См. также:**

**Оператор End Program, Процедура EndHandler**

---

## Оператор End Program

### Назначение

Прекращает выполнение прикладной программы, но не останавливает MapInfo.

### Предупреждение

Оператор **End Program** не может быть выполнен в окне MapBasic.

### Синтаксис

**End Program**

### Описание

Оператор **End Program** используется для остановки выполнения прикладной программы MapBasic. Прикладная программа на MapBasic может добавлять свои команды и даже заголовки в строку меню. Обычно, такие команды вызывают дополнительные процедуры из программы MapBasic. Такие процедуры неактивны, пока соответствующая команда не будет выбрана пользователем.

Если какая-либо процедура в рамках данной программы MapBasic выполнит оператор **End Program**, то вся программа прекращает работу, в том числе и все ее неактивные подпрограммы. Когда программа завершается с помощью этого оператора, MapInfo Professional автоматически удаляет все созданные ею команды и заголовки из меню.

Если в программе есть процедура с именем **EndHandler**, MapInfo Professional автоматически выполнит ее при остановке программы, каким бы образом эта остановка не совершалась.

**См. также:**

**Оператор End MapInfo, Процедура EndHandler**



---

## Процедура EndHandler

### Назначение

Специальная подпрограмма, sub-процедура, автоматически выполняющаяся при завершении выполнения программы.

### Синтаксис

**Declare Sub EndHandler Sub EndHandler** *statement\_list* End Sub, где *statement\_list* – список операторов.

### Описание

**EndHandler** – зарезервированное имя для процедуры MapBasic.

Процедура **EndHandler** называется обработчиком завершения программы. Когда пользователь запускает программу, в которой есть обработчик завершения, **EndHandler** то эта процедура будет автоматически вызвана при завершении программы. Обработчик сработает как при закрытии MapInfo, так и если программу останавливает **Оператор End Program**.

**Внимание:** Одновременно в состоянии ожидания могут находиться несколько прикладных программ. если эти программы снабжены обработчиками завершения **EndHandler**, то при закрытии MapInfo каждая из них сработает по очереди.

См. также:

**Процедура RemoteMsgHandler**, **Процедура SelChangedHandler**, **Процедура ToolHandler**, **Процедура WinChangedHandler**, **Процедура WinClosedHandler**

---

## Функция EOF( )

### Назначение

Возвращает FALSE, если MapBasic читает запись из файла, и TRUE, если пробует прочитать что-то после конца файла.

### Синтаксис

**EOF**( *filenum* )

*filenum* – номер открытого файла, который возвращает **Оператор Open File**.

### Возвращаемая величина

Логическое

### Описание

Функция **EOF( )** позволяет определить, достигнут ли конец файла, открытого в программе под номером `filenum`.

Если **Оператор Get** попыталась прочитать запись, следующую после последней, то функция **EOF( )** возвращает логическое "да" (TRUE); иначе, если была прочитана запись файла, **EOF( )** возвращает FALSE.

Функция **EOF( )** работает с открытыми файлами (текстовыми или произвольной природы); если нужно отследить конец чтения из открытой таблицы MapInfo, то используется **Функция EOF( )**.

Пример использования функции **EOF( )** можно посмотреть в тексте программы NIEWS.MB (Named Views).

### Ошибки:

В результате выполнения оператора может генерироваться код ошибки `ERR_FILEMGR_NOTOPEN`, если файл не был открыт.

### См. также:

**Функция EOF( ), Оператор Open File**

---

## Функция EOT( )

### Назначение

Возвращает FALSE, если MapBasic читает запись из таблицы, и TRUE, если программа пробует прочитать что-то после конца таблицы.

### Синтаксис

**EOT( table )**

`table` – имя открытой таблицы;

### Возвращаемая величина

Логическое

### Описание

Функция **EOT( )** Возвращается логическое "да" (TRUE), если MapBasic попытался прочитать запись, следующую после последней, и логическое "нет" (FALSE), если прочитана строка таблицы. Параметр `table` определяет имя таблицы, из которой производится чтение.

### Ошибки:

В результате выполнения оператора может генерироваться код ошибки `RR_TABLE_NOT_FOUND`, если не найдена данная таблица.

**Пример:**

Здесь результат функции **EOT( )** является критерием прекращения выполнения цикла. Цикл последовательно прочитывает записи из таблицы до тех пор, пока функция **EOT( )** не просигнализирует о достижении конца таблицы.

```
Dim f_total As Float
Open Table "customer"
Fetch First From customer
Do While Not EOT(customer)
    f_total = f_total + customer.order
    Fetch Next From customer
Loop
```

**См. также:**

**Функция EOF( ), Оператор Fetch, Оператор Open File, Оператор Open Table**

---

## Функция EPSGToCoordSysString\$( )

**Назначение**

Используется для преобразования строки в формате Spatial Reference System в **Оператор CoordSys**, чтобы она стала доступной другим функциям и операторам MapBasic.

**Синтаксис**

**EPSGToCoordSysString\$( *epsg\_string* )**

*epsg\_string* – строка с порядкового номера любой поддерживаемой географической системы координат (SRS). Дескрипторы SRS иногда называют дескрипторами EPSG (European Petroleum Survey Group) (например, epsg:2600). Полный список кодов EPSG, используемых MapInfo Professional, можно найти в файле MAPINFOW.PRJ, в каталоге в который была установлена программа MapInfo Professional. Коды EPSG можно определить по префиксу “\p”, за которым следует число.

**Описание**

Функция **EPSGToCoordSysString\$( )** преобразует строку SRS в значение параметра CoordSys, который можно применять в любой функции или операторе MapBasic, где этот параметр CoordSys может быть использован.

**Пример:**

В следующем примере задается стандартная система координат – проекция "Долгота / Широта (WGS 84) ( в форме параметров выражения: Earth Projection 1, 104).

```
run command("Set Map " + EPSGToCoordSysString$("EPSG:4326"))
```

**См. также:**

**Оператор CoordSys**

## Функция Erase( )

### Назначение

Возвращает объект, полученный из другого удалением части, которая перекрывается другим объектом.

### Синтаксис

**Erase**( *source\_object*, *eraser\_object* )

*source\_object* – объект любой природы, кроме точечного или текстового;

*eraser\_object* – замкнутый объект, выполняющий роль "ластика".

### Возвращаемая величина

Объект, который остается от объекта *source\_object* после удаления из него части *eraser\_object*.

### Описание

Функция **Erase( )** возвращает объект, полученный в результате удаления части объекта другим.

Объект *source\_object* может быть линейным (прямой линией, полилинией или дугой) или замкнутым объектом (область, прямоугольник, скругленный прямоугольник или эллипс), но не может быть точечным или текстовым объектом. Объект *eraser\_object* должен быть замкнутым. Объект, полученный в результате, унаследует стиль оформления (тип линии, штриховки и цвет) от объекта *source\_object*.

### Пример:

```
' В этом примере o1 и o2 – объектные переменные, ' которые уже имеют значения.  
If o1 Intersects o2 Then If o1 Entirely Within o2 Then Note "Удаление не  
состоялось: первый объект полностью перекрывается "ластиком"." Else o3 =  
Erase( o1, o2 ) End If Else Note "Удаление не состоялось: объекты не  
пересекаются." End If
```

### См. также:

[Оператор Objects Erase](#), [Оператор Objects Intersect](#)

---

## Функция Err( )

### Назначение

Возвращает числовой код, соответствующий текущей ошибке.

## Синтаксис

**Err** ( )

## Возвращаемая величина

Целое число типа Integer.

## Описание

Функция **Err**( ) возвращает целочисленный код, соответствующий последней из происшедших ошибок.

Обычно, когда в прикладной программе генерируется ошибка, выводится сообщение о ней и выполнение программы прекращается. Если в программе есть процедура обработчика ошибки, вход в которую обеспечивает **Оператор OnError**, выполнение программы передается обработчику, как только какой-нибудь оператор, находящийся после OnError, возвращает ошибку. В обработчике ошибок Вы можете применить функцию **Err**( ) для определения происшедшей ошибки.

Функция **Err**( ) возвращает код ошибки, только если она вызвана из процедуры-обработчика. После того, как программа выполнит **Resume statement** для возвращения из обработчика в процедуру, вызвавшую ее, код ошибки в программе обнуляется. Поэтому не имеет смысла использовать функцию **Err**( ) вне обработчика ошибок, т. к. функция, вызванная после обработчика, вернет ноль.

В этой книге после описаний некоторых операторов и функций следует раздел "Ошибки", в котором перечисляются коды некоторых возможных ошибок, генерируемых описываемым оператором или функцией. Но это не все возможные ошибки, а только связанные с наиболее часто встречающимися ситуациями.

Некоторые коды ошибок MapBasic генерируются только в специфических, редко встречающихся ситуациях. Например, код ERR\_INVALID\_CHANNEL генерируется только функциями и операторами, обслуживающими DDE-связь. Если оператор может генерировать такой "специальный" код, то он также приводится в разделе "Ошибки" для этого оператора.

Другие ошибки MapBasic, напротив, порождаются часто и многими функциями. Например, **Функция Area**( ) и **Функция ObjectInfo**( ) используют выражение типа Object как параметр. Этими функциями может генерироваться код ошибки ERR\_FCN\_OBJ\_FETCH\_FAILED, если Вы задали параметр в форме *tablename.obj*, тогда как текущая строка этой таблицы не имеет присоединенного объекта. Другими словами, каждая функция, обрабатывающая объекты, может породить ошибку ERR\_FCN\_OBJ\_FETCH\_FAILED. Так как ошибка ERR\_FCN\_OBJ\_FETCH\_FAILED может случиться во многих и разнообразных ситуациях, многие функции не рапортуют о ней как о явной причине сбоя.

Можно еще упомянуть коды двух математических ошибок – ERR\_FP\_MATH\_LIB\_DOMAIN и ERR\_FP\_MATH\_LIB\_RANGE, которые возникают в результате ошибки в численном параметре. Эти ошибки могут возникать в случае выполнения любой из следующих функций: **Функция Acos**( ), **Функция Asin**( ), **Функция Atn**( ), **Функция Cos**( ), **Функция Exp**( ), **Функция Log**( ), **Функция Sin**( ), **Функция Sqr**( ) или **Функция Tan**( ).

Полный список кодов ошибок MapBasic находится в файле ERRORS.DOC.

См. также:

Оператор Error, Функция Error\$( ), Оператор OnError

---

## Оператор Error

### Назначение

Генерирует ошибку определенного вида.

### Синтаксис

**Error** *error\_num*

*error\_num* – целочисленный код, соответствующий стандартной ошибке в MapBasic.

### Описание

Оператор **Error** генерирует ошибку по заданному коду. Обычно он используется при отладке.

Если в программе до этого была определена процедура обработчика ошибок (смотрите [Оператор OnError](#)), то MapBasic передает управление процедуре-обработчику ошибок. Если обработчик отсутствует в программе, то после выполнения оператора **Error** MapBasic выводит сообщение о ней и останавливает выполнение программы.

См. также:

Функция Err( ), Функция Error\$( ), Оператор OnError

---

## Функция Error\$( )

### Назначение

Возвращает сообщение о текущей ошибке.

### Синтаксис

**Error\$( )**

### Возвращаемая величина

Строка

### Описание

Функция **Error\$( )** возвращает текст сообщения о последней ошибке, если она была зафиксирована. Если ошибок не было, то функция **Error\$( )** возвращает пустую строку.

Функция **Error\$( )** может быть вызвана только из процедуры-обработчика ошибок. Более подробную информацию см. в разделе [Функция Err\( \) на стр. 60](#).

См. также:

Функция `Err( )`, Оператор `Error`, Оператор `OnError`

## Оператор `Exit Do`

### Назначение

Останавливает действующий Оператор `Do...Loop`.

### Предупреждение

Оператор `Exit Do` не может быть выполнен в окне MapBasic.

### Синтаксис

`Exit Do`

### Описание

Оператор `Exit Do` прекращает выполнение цикла, который начал Оператор `Do...Loop`. После выполнения `Exit Do`, MapBasic передает управление оператору, перед которым стоит Оператор `Do...Loop`. Оператор `Exit Do` действителен только внутри Оператор `Do...Loop`.

Оператор `Do...Loop` может быть вложенным, т.е. один Оператор `Do...Loop` может быть вложен в другой Оператор `Do...Loop`. Оператор `Exit Do` прекращает только Оператор `Do...Loop`, внутри которого употребляется. Так в конструкции:

```
Do While condition1
:
    Do While condition2
    :
        If error_condition
        Exit Do
        End If
    :
    Loop
:
Loop
```

оператор `Exit Do` прерывает вложенный цикл (организованный вторым оператором (`Do While условие2`), не останавливает первый цикл (`Do While условие1`).

См. также:

Оператор `Do...Loop`, Оператор `Exit For`, Оператор `Exit Sub`

## Оператор Exit For

### Назначение

Останавливает действующий **Оператор For...Next**.

### Предупреждение

Оператор **Exit For** не может быть выполнен в окне MapBasic.

### Синтаксис

**Exit For**

### Описание

Оператор **Exit For** прекращает выполнение цикла, который начал **Оператор For...Next**. После выполнения **Exit For**, MapBasic передает управление оператору, перед которым стоит **Оператор For...Next**. Оператор **Exit For** действителен только внутри **Оператор For...Next**.

**Оператор For...Next** может быть вложенным, т.е. один **Оператор For...Next** может быть вложен в другой **Оператор For...Next**. Оператор **Exit For** прекращает только **Оператор For...Next**, внутри которого употребляется. Так в конструкции:

```
For x = 1 to 5
:
  For y = 2 to 10 step 2
  :
    If error_condition
      Exit For
    End If
  :
Next
:
```

оператор **Exit For** прерывает вложенный цикл (организованный вторым оператором (**For y = 2 to 10 step 2**), но не останавливает первый цикл (**For x = 1 to 5**).

### См. также:

**Оператор Exit Do, Оператор For...Next**

---

## Оператор Exit Function

### Назначение

Останавливает подпрограмму, которую запустил **Оператор Function...End Function**.

### Предупреждение

Оператор **Exit Function** не может быть выполнен в окне MapBasic.



## Синтаксис

`Exit Function`

## Описание

Оператор **Exit Function** прекращает выполнение функции на MapBasic. Оператор **Exit Function** не может быть выполнен в **Оператор Function...End Function**.

Вызовы функции могут быть вложенными, то есть одна функция может вызывать другую. Каждый оператор **Exit Function** прекращает выполнение только той функции, внутри которой употребляется.

См. также:

**Оператор Function...End Function**

---

## Оператор Exit Sub

### Назначение

Останавливает подпрограмму, которую запустил **Оператор Sub...End Sub**.

### Предупреждение

Оператор **Exit Sub** не может быть выполнен в окне MapBasic.

## Синтаксис

`Exit Sub`

## Описание

Оператор **Exit Sub** осуществляет выход из текущей процедуры, организованной при помощи оператора Sub... End Sub. Оператор **Exit Sub** может выполняться в теле только таких процедур.

**Оператор Sub...End Sub** может быть вложенным; т.е. может вызывать другую sub-процедуру, которая, в свою очередь, вызывает третью и т. д. Каждый оператор **Exit Sub** прекращает выполнение только той подпрограммы, внутри которой употребляется.

См. также:

**Оператор Call, Оператор Sub...End Sub**

### Функция Exp( )

#### Назначение

Вычисляет значение экспоненты.

#### Синтаксис

**Exp**( *num\_expr* )

*num\_expr* – числовое выражение

#### Возвращаемая величина

Вещественное число типа Float.

#### Описание

Математическая функция **Exp( )** вычисляет значение числа  $e$ , возведенного в степень *num\_expr*. Число  $e$  иррационально и приблизительно равно 2.7182818.

**Внимание:** Операция возведения в степень производится с помощью математического оператора (^).

#### Пример:

```
Dim e As Floate = Exp(1) ' переменная e примет значение ' равное примерно 2.7182818
```

#### См. также:

[Функция Cos\( \)](#), [Функция Sin\( \)](#), [Функция Log\( \)](#)

## Оператор Export

### Назначение

Экспортирует таблицу в файл другого формата.

### Синтаксис (вариант 1 - для экспорта в формат MIF/MID, DBF или ASCII)

```
Export table
  Into file_name
  [ Type
    { "MIF" |
      "DBF" Charset char_set ] |
    "ASCII" Charset char_set ] [ Delimiter "d " ] [ Titles ] } ]
  "CSV" [ Charset char_set ] [ Titles ] } ]
  [ Overwrite ]
```

### Синтаксис (вариант 2 - для экспорта в формат DXF)

```
Export table
  Into file_name
  [ Type "DXF" ]
  [ Overwrite ]
  [ Preserve
    [ AttributeData ] [ Preserve ] [ MultiPolygonRgns [ As Blocks ] ] ]
  [ { Binary | ASCII [ DecimalPlaces decimal_places ] } ]
  [ Version { 12 | 13 } ]
  [ Transform
    ( MI_x1, MI_y1 ) ( MI_x2, MI_y2 )
    ( DXF_x1, DXF_y1 ) ( DXF_x2, DXF_y2 ) ]
```

*table* – имя открытой таблицы (в этом операторе кавычки при задании имени таблицы не используются);

*file\_name* – строка с именем файла, в которую будут экспортироваться данные (если строка не включает DOS-маршрут, то файл будет создан в рабочем каталоге);

*char\_set* – величина типа String, слово, определяющее кодировку символов в экспортированном файле, такое как “MacRoman” или “WindowsLatin1” (см. раздел [Предложение CharSet on page 136](#)).

*d* – символ, который будет использован как разделитель для ASCII-файла;

*decimal\_places* – целое число типа Small Integer (короткое целое) от 0 до 16 (по умолчанию 6), задающее количество позиций после десятичной точки для экспорта числа с плавающей точкой в текстовый формат ASCII;

*MI\_x1, MI\_y1* и т. п. – пары чисел, представляющие собой координаты в таблице MapInfo;

*DXF\_x1, DXF\_y1* и т. п. – пары чисел, представляющие собой координаты в файле DXF.

## Описание

Оператор **Export** копирует содержимое таблицы MapInfo в отдельный файл другого формата для работы с этим файлом в других программах и операционных средах. Например, Вы можете экспортировать содержимое таблицы в DXF-файл, чтобы импортировать его в AutoCAD или другую программу, которая работает с форматом DXF. Оператор **Export** не изменяет исходный файл.

## Задание формата файла

Формат будущего файла объявляется при помощи предложения **Type**.

Следующая таблица описывает форматы

| Значение<br>Type    | Формат будущего файла  |
|---------------------|--|
| <b>Type</b> "MIF"   | Формат обмена MapInfo (MapInfo Interchange File format). Описание формата MIF смотрите в документации MapInfo.                           |
| <b>Type</b> "DXF"   | Формат DXF (графический формат программ CAD, таких как AutoCAD).   |
| <b>Type</b> "DBF"   | Формат dBASE и других совместимых баз данных.<br><b>Внимание:</b> Графические объекты не экспортируются при использовании этого формата. |
| <b>Type</b> "ASCII" | Текстовый файл<br><b>Внимание:</b> Графические объекты не экспортируются при использовании этого формата.                                |
| <b>Type</b> "CSV"   | Формат данных, разделенных запятыми<br><b>Внимание:</b> Графические объекты не экспортируются при использовании этого формата.           |

Если предложение **Type** отсутствует, MapInfo попытается определить формат экспорта по расширению в имени файла. Например, если в операторе задается имя "PARCELS.DXF", то MapInfo создаст файл формата DXF.

Если файл с заданным именем уже существует и в операторе используется слово **Overwrite**, MapInfo создаст экспортный файл, записав его поверх старого. Если слово **Overwrite** опущено и файл с заданным именем уже есть на диске, MapInfo не будет переписывать файл.

## Экспорт в текстовые ASCII-файлы

Если таблица экспортируется в текстовый ASCII-файл, то последний будет содержать разделители. Разделителем называется специальный символ, разделяющий данные полей в строке записи. Файлы в формате CSV используют запятую (,) в качестве разделителя автоматически. Другие разделители для экспорта в формат CSV задавать нельзя.

Стандартным разделителем для текстовых ASCII-файлов является символ табуляции (Chr\$(9)). Для этого случая Вы можете задать в предложении **Delimiter** другой символ-разделитель. В следующем примере символом разделителя назначается двоеточие (:):

```
Export sites Into "sitedata.txt" Type "ASCII"
  Delimiter ":" Titles
```

При экспорте в файлы ASCII или CSV, Вы можете использовать слово **Titles**. Если задано слово **Titles**, то первая строка в тексте будет содержать заголовки колонок таблицы. Если слово **Titles** не задано, то названия колонок в текстовый файл не попадут (что может создать путаницу, если Вы часто экспортируете данный файл).

### Экспорт в DXF-файл

Для получения файла в формате DXF используется второй вариант синтаксиса оператора **Export**, который может включать следующие специфические для формата DXF предложения и ключевые слова:

Предложение **Preserve AttributeData** используется для экспорта табличных данных в атрибуты DXF-файла.

Предложение **Preserve MultiPolygonRgns As Blocks** используется для того, чтобы MapInfo экспортировала область, состоящую из нескольких полигонов, в качестве блока DXF. Если это предложение будет опущено, то каждый полигон сложносоставной области будет представлен отдельным объектом.

Ключевое слово **Binary** используется для экспорта таблицы в бинарный файл DXF; предложение **ASCII** задает текстовый вид файла DXF. Если не используется ни один из этих вариантов, то MapInfo создаст файл ASCII DXF. Бинарный файл DXF в основном меньше, чем текстовый, и поэтому изготавливается быстрее. При создании тестового файла можно задать число знаков после десятичной точки для экспорта численных данных, имеющих тип Float в таблице MapInfo. Параметр `decimal_places` может принимать значения от 0 до 16, по умолчанию 6.

Предложения **Version 12** и **Version 13** указывают, для какой версии программы AutoCAD будет создан файл DXF. Если предложение не задано, MapInfo будет экспортировать в DXF-файл 12-ой версии.

**Transform** используется для задания координатного искажения. После слова **Transform** задаются координатные пары минимума и максимума таблицы MapInfo и какие пары координат будут соответствовать им в файле DXF.

### Пример:

Здесь открывается таблица "FACILITY " и экспортируется в файл FACIL.DXF.

```
Open Table "facility"

Export facility
  Into "FACIL.DXF"
  Type "DXF"
  Overwrite
  Preserve AttributeData
  Preserve MultiPolygonRgns As Blocks
```

```
ASCII DecimalPlaces 3  
Transform (0, 0) (1, 1) (0, 0) (1, 1)
```

См. также:

[Оператор Import](#)

---

## Функция ExtractNodes( )

### Назначение

Возвращает полилинию или область из подмножества узлов существующего объекта.

### Синтаксис

**ExtractNodes**(*object*, *polygon\_index*, *begin\_node*, *end\_node*, *b\_region*)

*object* – объект типа "полилиния" или "область";

*polygon\_index* – короткое целое число, 1 или больше; в случае области задает номер полигона, в случае полилинии – номер ломаной;

*begin\_node* – короткое целое число, 1 или больше, которое задает первый узел для выбираемого подмножества узлов;

*end\_node* – короткое целое число, 1 или больше, которое задает последний узел для выбираемого подмножества узлов;

*b\_region* – логическая величина, управляющая типом объекта, полученным в результате: для области – логическое "да" (TRUE), для полилинии – логическое "нет" (FALSE).

### Возвращаемая величина

Объект типа "полилиния" или "область". Величина типа Object. MapBasic присваивает все стили (цвет и т. п.) оригинального объекта объекту, полученному в результате. Если каких-то стилей недостает, то берутся текущие в MapBasic значения.

### Описание

Если значение параметра *begin\_node* равно или больше значения *end\_node*, то MapBasic будет отсчитывать узлы в следующем порядке:

- *begin\_node* (начальный узел) - следующий узел -... - последний узел полигона;
- первый узел полигона - ... - конечный узел *end\_node*).

Если параметр *object* задает область и оба параметра *begin\_node* и *end\_node* равны 1 (единице), то MapBasic вернет полное множество узлов одного полигона (многоугольника). Это простой механизм для выделения одного полигона из многокомпонентной области. Для определения числа узлов полигона из области используется [Функция ObjectInfo\( \)](#).

**Ошибки:**

Функция может вернуть код ошибки `ERR_FCN_ARG_RANGE`, если значение параметра `b_region` равно `FALSE` и подмножество узлов состоит менее чем из двух узлов или если значение параметра `b_region` равно `TRUE` и подмножество узлов состоит менее чем из трех узлов.

**См. также:**

Функция `ObjectNodeX( )`, Функция `ObjectNodeY( )`

---

## Оператор Farthest

**Назначение**

Ищет объект в таблице, который находится далее всего от заданного. Результатом является объект полилиния, представляющая наибольшее расстояние.

**Синтаксис**

```
Farthest [ N | All ] From { Table fromtable | Variable fromvar }
To totable Into intotable
[ Type { Spherical | Cartesian } ]
[ Ignore [ Contains ] [ Min min_value ] [ Max max_value ] Units unitname ]
[ Data clause ]
```

*N* – параметр (необязательный), представляющий число наиболее удаленных объектов, которые будут найдены. По умолчанию – 1. Если используется параметр **All**, то объект-расстояние будет создан для каждой комбинации.

*fromtable* – таблица объектов, от которых требуется найти наиболее удаленные объекты.

*fromvar* – переменная MapBasic для объекта, от которого ищется самое большое расстояние;

*totable* – таблица объектов, до которых будут искаться наидлиннейшие расстояния;

*intotable* – таблица, в которую будут записаны полученные результаты;

*min\_value* – минимальная допустимая дистанция для результата;

*max\_value* – максимальная допустимая дистанция для результата;

*unitname* – единицы измерения параметров *min\_value* и/или *max\_value* (например, “km”).

**Описание**

Оператор **Farthest** находит все объекты из таблицы *fromtable*, которые расположены далее всего от заданного объекта. Рассматривается каждый объект в таблице *fromtable*. Для каждого объекта в таблице *fromtable* будет найден наиболее самый дальний объект в таблице *totable*. Если задан параметр *N*, то будет найдено *N* наиболее дальних объектов в таблице *totable*. Объект-полилиния, соединяющий наиболее удаленные точки между объектом *fromtable* и выбранным объектом в таблице *totable*, помещается в таблицу *intotable*. Если задан параметр **All**, то объект помещается в таблицу *intotable*, представляющую расстояние между объектом *fromtable* каждым объектом таблицы *totable*.

Если имеется несколько объектов в таблице *totable*, которые имеют одинаковое расстояние до заданного объекта в таблице *fromtable*, то в виде результата возвращается только один из них. Если возвращается несколько объектов (задана величина *N*, большая чем 1), то объекты с одинаковым расстоянием будут записаны последовательно. Если существует второй объект с таким же расстоянием, а запрашивается 3 объекта, то искомым объектом станет третий по счёту объект.

Типы объектов в таблицах *fromtable* и *totable* могут быть любыми кроме текстовых. Например, если обе таблицы содержат объекты-регионы, то находится максимальное расстояние между объектами-регионами и создаётся объект-полилиния, соединяющая две точки каждого объекта, использованных для расчёта расстояния. Если объекты-регионы пересекаются, минимальное расстояние будет нулевым, и объект-полилиния будет редуцированным, с двумя точками, имеющими те же самые координаты, как и у точки пересечения.

Расстояние, рассчитанное этой функцией, не принимается во внимание при расчёте маршрутов в транспортных задачах. Это расстояние можно сравнить с "перелётом птицы".

Предложение **Ignore** может использоваться для ограничения искомого расстояния, и может определять, сколько объектов таблицы *totable* найдено для каждого объекта таблицы *fromtable*. С помощью параметра **Min** можно исключать нулевые расстояния. Это может быть полезно в случае двух таблиц с точками, чтобы не рассчитывать расстояния от одинаковых точек. Например, если две таблицы с точками, представляют города, и мы хотим найти ближайшие города, мы можем исключить из поиска одинаковые города.

С помощью параметра **Max** можно ограничить число объектов, рассматриваемых в *totable*. Это полезно при совместном использовании с параметрами *N* или **All**. Например, пусть надо найти 5 аэропортов, находящихся ближе остальных для нескольких городов (где *fromtable* это набор городов, а *totable* это набор аэропортов), но мы не будем рассматривать аэродромы, удалённые более чем на 100 км. Результат может оказаться меньшим, чем пять аэропортов для какого-либо города. Этот параметр целесообразно использовать совместно с параметром **All**, так можно найти все аэропорты на расстоянии 100 км от города.

Применение параметра **Max** может улучшить поведение оператора **Farthest**, поскольку он эффективно ограничивает число искомым объектов *totable*.

Найденные расстояния отфильтровываются так, что они строго больше чем *min\_value* и меньше или равны *max\_value*:

```
min_value < distance <= max_value
```

Можно сделать так, чтобы расстояния возвращались в виде нескольких проходов оператора **Farthest**. Например, первый проход может вернуть все объекты на расстоянии от 0 до 100 км, второй проход может вернуть все объекты на расстоянии от 100 до 200 км, а результаты будут содержать повторяющихся записей (то есть, расстояние 100 встретилось только в первом проходе, а во втором уже не встречается).

**Type** – метод, используемый для расчёта расстояния между объектами. Может быть либо сферическим (Spherical), либо декартовым (Cartesian). Тип алгоритма вычислений должен соответствовать системе координат результирующей таблицы *intotable*, иначе произойдет ошибка. Если система координат результирующей таблицы *intotable* – Долгота/Широта, а метод расчёта декартовый, то будет выдано сообщение об ошибке. Если система координат результирующей таблицы *intotable* – Долгота/Широта, а метод расчёта декартовый, то будет выдано сообщение об ошибке.



Предложение **Ignore** ограничивает возвращаемые расстояния. Любые расстояния, найденные по запросу, которые меньше или равны *min\_value* или больше *max\_value* игнорируются; *min\_value* и *max\_value* являются расстояниями и измеряются в единицах, определенных параметром *unitname*. Если *unitname* не соответствует стандартным единицам измерения, будет выдано сообщение об ошибке. Список доступных единиц измерения смотрите в [Оператор Set Distance Units on page 633](#). Предложение **Ignore** не является обязательным, равно как и подпредложения **Min** и **Max**.

Если один объект находится внутри другого, то расстояние между ними считается равным нулю. Например, если таблица *fromtable* это WORLDCAPS, а таблица *totable* - это WORLD, то расстояние между Москвой и Россией будет равно нулю. Если же флажок **Contains** в предложении **Ignore** установлен, то расстояние не будет обязательно нулевым. Вместо этого результатом будет расстояние между Москвой и границей России. Все замкнутые объекты, типа областей (регионов) будут обработаны как полилинии ради этой операции.

Предложение **Data** используется для того, чтобы отметить от каких объектов из таблиц *fromtable* и *totable* будут получен результат.

### Предложение Data

**Data** *IntoColumn1=column1, IntoColumn2=column2*

Переменная *IntoColumn* в левой части уравнения должна корректно задавать колонку в *intotable*. Имя колонки справа от знака равенства должно быть именем колонки либо таблицы *totable*, либо *fromtable*. Если одинаковое имя колонки существует в обеих таблицах *totable* и *fromtable*, то будет использоваться колонка из таблицы *totable* (т.е. в таблице *totable* в первую очередь ищутся имена колонок, а это правая часть "уравнения").

Чтобы избежать конфликтов такого типа, имена колонок можно заменять псевдонимами таблиц: Например:

```
Data name1=states.state_name, name2=county.state_name
```

Чтобы заполнить колонку в таблице *intotable* значениями расстояний, нужно выполнить команду **Таблица > Обновить колонку** или использовать [Оператор Update](#).

**См. также:**

[Оператор Nearest](#), [Функция CartesianObjectDistance\(\)](#), [Функция ObjectDistance\(\)](#), [Функция SphericalObjectDistance\(\)](#), [Функция CartesianConnectObjects\(\)](#), [ConnectObjects\(\)](#) функция, [Функция SphericalConnectObjects\(\)](#)

## Оператор Fetch

### Назначение

Задание текущей позиции курсора в таблице (т. е. какая строка таблицы должна быть текущей).

### Синтаксис

```
Fetch { First | Last | Next | Prev | Rec n } From table
```

*n* – номер записи для чтения;

*table* – имя открытой таблицы;

### Описание

Оператор **Fetch** используется для позиционирования записи в открытой таблице. Выполняя оператор **Fetch**, Ваша программа помещает курсор на определенную строку таблицы. Это состояние записи в таблице называется “текущим”.

**Внимание:** Термин "курсор" используется здесь для отражения расположения строки в таблице. Он никак не связан с визуальным курсором экрана.

После оператора **Fetch** приложение MapBasic может извлекать данные из текущей записи, используя выражения:

| Синтаксис                  | Пример:        |
|----------------------------|----------------|
| <i>table.column</i>        | World.Country  |
| <i>table.col#</i>          | World.col1     |
| <i>table.col( number )</i> | World.col( 1 ) |

Оператор **Fetch First** переводит курсор на первую неудаленную запись таблицы.

Оператор **Fetch Last** переводит курсор на последнюю неудаленную запись таблицы.

Оператор **Fetch Next** переводит курсор на следующую неудаленную запись таблицы.

Оператор **Fetch Prev** переводит курсор на предыдущую неудаленную запись таблицы.

Оператор **Fetch Rec *n*** переводит текущую позицию на определенную строку, даже если она удалена.

**Внимание:** Если запись удалена, то оператор генерирует ошибку выполнения 404

Надо учитывать, что многие операторы MapBasic и команды MapInfo сбрасывают текущее положение курсора в таблице (например, оператор Select, оператор Update, перерисовка экрана). Используйте обращение к текущей строке сразу после выполнения оператора **Fetch**.

### Определение конца таблицы

После выполнения оператора **Fetch**, **Функция EOT( )** возвращает логическую величину (TRUE или FALSE) в зависимости от того, достигнут ли предел таблицы после последней записи.

Если оператор **Fetch** помещает курсор на действительную запись таблицы, то **Функция EOT( )** возвращает значение FALSE.

Если оператор **Fetch** попытается установить текущей запись после последней записи таблицы, то **Функция EOT( )** вернет значение TRUE.

В следующем примере оператор **Fetch Next** проходит по циклу по всем строкам таблицы. После каждого применения оператора **Fetch Next** вызывается функция **Функция EOT( )** и, если она вернет истинное значение, то цикл будет прерван.

```
Dim i As Integer i = 0 Fetch First From world Do While Not EOT(world) i =
i + 1 Fetch Next From world Loop Print "Число не удаленных записей: " + i
```

## Примеры

В этом примере показано, как подвести курсор к 3-ей записи в таблице STATES:

```
Open Table "states" Fetch Rec 3 From states 'позиционирование на 3-ю
запись Note states.state_name 'показ имени штата
```

Оператором **Fetch** можно извлекать значения из временных таблиц (например, из таблицы Selection).

```
Select * From states Where pop_1990 < pop_1980 Fetch First From
Selection Note Selection.col1 + " содержит отрицательный прирост населения"
```

См. также:

**Функция EOT( ), Оператор Open Table**

---

## Функция FileAttr( )

### Назначение

Возвращает информацию об открытом файле.

### Синтаксис

```
FileAttr( filenum, attribute )
```

*filenum* – целое число, присвоенное файлу оператором Open File.

*attribute* – код возвращаемого атрибута, см. таблицу ниже.

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция **FileAttr( )** используется для получения информации об открытом в файле MapInfo.

Параметр *attribute* должен принимать одно из следующих значений:

| параметр <i>attribute</i> | Возвращаемая величина   |
|---------------------------|---|
| FILE_ATTR_MODE            | Короткое целое, величина типа Small Integer, соответствующая коду режима, в котором был открыт файл. Значения: <ul style="list-style-type: none"><li>• MODE_INPUT</li><li>• MODE_OUTPUT</li><li>• MODE_APPEND</li><li>• MODE_RANDOM</li><li>• MODE_BINARY</li></ul> |
| FILE_ATTR_FILESIZE        | Целое число, величина типа Integer, размер файла в байтах.  |

### Ошибки:

В результате выполнения функции Вы можете получить код ошибки ERR\_FILEMGR\_NOTOPEN, если файл не был открыт.

### См. также:

[Функция EOF\( \)](#), [Оператор Get](#), [Оператор Open File](#), [Оператор Put](#)

---

## Функция FileExists( )

### Назначение

Возвращает логическую величину (TRUE или FALSE), в зависимости от того, существует файл или нет.

### Синтаксис

**FileExists**( *filespec* )

*filespec* – строка с полным именем файла, включая DOS-маршрут.

### Возвращаемая величина

Логическое: TRUE, если файл существует: иначе – FALSE.

### Пример:

```
If FileExists("C:\MapInfo\TODO.TXT") Then  
  
    Open File "C:\MapInfo\TODO.TXT" For INPUT As #1  
  
End If
```

### См. также:

[Функция TempFileName\\$\( \)](#)

## Функция FileOpenDlg( )

### Назначение

Вызывает диалоговое окно команды Файл > Открыть таблицу из меню в MapInfo и возвращает имя файла, выбранное пользователем.

### Синтаксис

**FileOpenDlg**( *path*, *filename*, *filetype*, *prompt* )

*path* – строковая величина, определяющая начальное значение выбранного каталога;

*filename* – строковая величина, определяющая начальное значение выбранного файла в каталоге;

*filetype* – строковая величина в три символа или меньше, определяющая тип файла;

*prompt* – строковая величина, задающая заголовок диалога.

### Возвращаемая величина

Строка с именем файла или пустая строка, если диалог был отменен пользователем. Величина типа String.

### Описание

Функция **FileOpenDlg**( ) открывает диалоговое окно подобно команде MapInfo **Файл > Открыть таблицу**.

Чтобы выбрать файл из списка, представленного в диалоге, Вы можете указать на имя файла и нажать на кнопку **OK**, или просто дважды указать мышкой на имя файла. В этом случае функция **FileOpenDlg**( ) возвращает полное имя выбранного файла, включая DOS-маршрут. Если Вы нажали на кнопку **Отмена**, то возвращаемая строка будет пуста ("").

Заметим, что функция **FileOpenDlg**( ) не открывает выбранный файл; она только показывает диалог, в котором файл можно выбрать. Для того чтобы открыть этот файл, программе необходим дополнительный оператор, например, **Оператор Open Table**. Если же Вам нужно, чтобы в Вашей программе выбранный файл открывался автоматически, используйте **Оператор Run Menu Command** с аргументами M\_FILE\_OPEN или M\_FILE\_ADD\_WORKSPACE.

Параметр *path* назначает каталог, в котором помещены файлы для выбора. Значение каталога является начальным, и ничто не мешает пользователю выбрать другой каталог в диалоговом окне. Если параметр не задан (пустое значение), то каталогом выбора будет текущий рабочий каталог.

Параметр *filename* принимает строковые значения и задает предварительное название файла.

Параметр *filetype* принимает строковые значения обычно длиной в три символа и определяет тип файлов для выбора в диалоговом окне. Диалог может автоматически реагировать на параметр *filetype*; так если, например, параметр *filetype* имеет значение "TAB", то диалог покажет список файлов таблиц MapInfo, если же задано значение *filetype* "WOR", то диалог покажет список файлов с Рабочими Наборами MapInfo.

Существуют и другие трехсимвольные значения. Эти специальные значения параметра *filetype* приведены в следующей таблице (только из трех символов). Если надо вывести список всех файлов, то в качестве параметра *filetype* используется маска "\*. \*".

| Параметр <i>filetype</i> | Типы файлов   |
|--------------------------|---|
| "TAB"                    | Таблицы MapInfo.  |
| "WOR"                    | Рабочие Наборы MapInfo.   |
| "MIF"                    | Формат обмена данными MapInfo (MapInfo Interchange Format), используется для импорта/экспорта карт в/из ASCII-текстовых файлов. |
| "DBF"                    | Формат dBASE или других совместимых баз данных.   |
| "WKS", "WK1"             | Табличные файлы Lotus.  |
| "XLS"                    | Табличные файлы Excel.  |
| "DXF"                    | Формат обмена данными AutoCAD.  |
| "MMI", "MBI"             | Формат обмена данными с MapInfo для DOS.  |
| "MB"                     | Файлы с текстами программ MapBasic.   |
| "MBX"                    | Файлы с откомпилированными программами MapBasic.  |
| "TXT"                    | Текстовые файлы.  |
| "BMP"                    | Формат растрового образа Windows.   |
| "WMF"                    | Метафайлы Windows.  |

В Windows каждому типу соответствует определенное DOS-расширение имени файла. Другими словами, тип *filetype* файла "WOR" говорит MapBasic, чтобы был показан список файлов с расширением ".WOR", т. к. имена с таким расширением имеют файлы Рабочих Наборов в MapInfo для Windows.

Если Вы хотите, чтобы Вашу программу можно было использовать в любой системе, используйте только трехсимвольную установку параметра *filetype*. В Windows некоторые специальные значения типа файла соответствуют строчке меню в нижнем левом углу диалога. Если в функции **FileOpenDlg ( )** параметр *filetype* задает значение не из приведенного выше списка, то в диалоге в окошке списка типов файлов будет показано пустое место.

**Пример:**

```
Dim s_filename As String
s_filename = FileOpenDlg("", "", "TAB", "Open Table")
```

**См. также:**

**Функция FileSaveAsDlg( ), Оператор Open File, Оператор Open Table**

---

## Функция FileSaveAsDlg( )

**Назначение**

Вызывает диалоговое окно команды Файл > Создать копию в MapInfo и возвращает имя файла, введенное пользователем.

**Синтаксис**

```
FileSaveAsDlg( path, filename, filetype, prompt )
```

*path* – строковая величина, определяющая начальное значение выбранного каталога;

*filename* – строковая величина, определяющая начальное значение выбранного файла в каталоге;

*filetype* – строковая величина в три символа или меньше, определяющая тип файла;

*prompt* – строковая величина, задающая заголовок диалога.

**Возвращаемая величина**

Строка с заданным именем файла (пустая строка, если введенные в диалоге данные были отменены).

**Описание**

Функция **FileSaveAsDlg( )**, подобно команде MapInfo **Файл > Создать копию**, открывает диалоговое окно сохранения файла.

Пользователь может ввести с клавиатуры имя файла или выбрать файл из списка, представленного в диалоге. Для этого Вы можете указать на имя файла и нажать на кнопку ОК, или просто дважды указать мышкой на имя файла. Если файл уже существует на диске, MapBasic спросит, действительно ли необходимо заменить старое содержимое файла на новое.

После того, как пользователь выберет файл и нажмет на кнопку **ОК**, функция **FileSaveAsDlg( )** возвращает полное имя введенного или выбранного файла, включая DOS-маршрут. Если Вы нажали на кнопку **Cancel**, то возвращаемая строка будет пуста ("").

Параметр *path* назначает каталог, в котором должен быть сохранен файл. Значение каталога является начальным, и пользователь может изменить каталог в диалоговом окне. Если параметр не задан (пустое значение), то каталогом будет текущий рабочий каталог.

Параметр *filename* принимает строковые значения и задает предварительное название файла.

Значение *filetype*, определяет расширение файла (до трех букв), то есть его тип. Например, если параметр *filetype* принимает значение "TAB", то диалог покажет список файлов таблиц MapInfo, а если – "WOR", то диалог покажет список файлов с Рабочими Наборами MapInfo. Все специальные значения параметра *filetype* приведены в таблице в разделе **Функция FileOpenDlg( )** на стр. 77.

Функция **FileSaveAsDlg( )** не сохраняет выбранный файл, но использует диалог соответствующей команды для получения полного имени, под которым надо сохранить файл. После этого для сохранения выбранного файла можно выполнить **Оператор Commit Table**.

**См. также:**

**Оператор Commit Table, Функция FileOpenDlg( )**

---

## Оператор Find

### Назначение

Поиск объекта или пересечения объектов таблицы на Карте.

### Синтаксис

```
Find address [ , region ] [ Interactive ]
```

*address* – строковая величина, определяющая адрес или имя искомого объекта Карты; для поиска пересечения двух улиц параметр использует синтаксис: *улица && улица*.

*region* – имя области, в которой будет осуществляться поиск.

### Описание

Оператор **Find** осуществляет поиск на Карте именованных элементов, заданных параметром *address*. Результаты поиска сохраняются в системных переменных, значения которых использует **CommandInfo( )** функция. Если оператор **Find** включает ключевое слово **Interactive** и если MapBasic не может сразу обнаружить адрес, то поиск сопровождается диалоговым окном, в котором можно выбирать из нескольких "похожих" адресов.

Оператор **Find** может просматривать только таблицы, имеющие присоединенные графические объекты. Таблица при этом должна быть открыта. Оператор **Find** проводит поиск в той колонке, которая назначена для поиска. Программа MapBasic может выполнить **Оператор Find Using**, чтобы узнать, в какой колонке производить поиск. Если перед оператором **Find** не был выполнен оператор **Find Using**, MapBasic будет вести поиск, пользуясь установками, которые были заданы ранее в первом диалоге команды MapInfo Professional **Запрос > Найти**.

Оператор **Find** может уточнять поиск, если Вы зададите необязательный параметр *region* – обычно это имя области, в которой должен содержаться искомый объект. Другими словами, Вы можете просто попробовать найти город по названию (например, "Albany") в таблице городов или сузить область поиска, задав имя города и область, где он может находиться (например, "Albany", "CA"). Оператор **Find** не отмечает найденный объект символом, так как это делает команда поиска. Если необходимо показать пользователю результаты поиска, используется **Функция CreatePoint( )** или **Оператор Create Point**; смотрите пример ниже.



## Анализ результата поиска

Если после выполнения оператора **Find** изменить функцию

**CommandInfo(CMD\_INFO\_FIND\_RC)**, то Вы сможете узнать результат поиска.. В случае успеха координаты X и Y найденного объекта можно узнать при помощи **CommandInfo( )** с аргументами **CMD\_INFO\_X** и **CMD\_INFO\_Y** соответственно. Если после оператора **Find** вызвать функцию **CommandInfo(CMD\_INFO\_FIND\_ROWID)**, то она вернет номер строки, которой соответствует найденный объект.

Оператор **Find** может завершиться либо нахождением объекта, либо выбором ближайшего объекта, либо объект не будет найден. Если поиск оператором **Find** точно находит объект, то функция **CommandInfo(CMD\_INFO\_FIND\_RC)** возвратит единицу. Если Вы выбрали ближайший вариант, то функция **Find** возвратит число большее единицы. В случае неудачи поиска функция **Find** возвращает отрицательное значение.

В следующей таблице собраны различные варианты кодов, возвращаемых функцией **CommandInfo(CMD\_INFO\_FIND\_RC)**. Каждое значение задается не более чем тремя значащими цифрами, каждая из которых представляет результат одного из трех возможных этапов поиска.

| Значения единиц | Смысл  |
|-----------------|--|
| xx1             | Точное совпадение.   |
| xx2             | Был задействован файл сокращений.                                  |
| xx3 ( - )       | Точного совпадения не найдено.                                     |
| xx4 ( - )       | Не было задано имя объекта; точного совпадения не найдено.         |
| xx5 ( + )       | Пользователь выбрал имя в диалоговом окне Interactive.             |
| x1x             | Не была определена сторона улицы.                                  |
| x2x ( + / - )   | Номер дома попал в заданные пределы для данной улицы.              |
| x3x ( + / - )   | Номер дома оказался вне заданных пределов для данной улицы.        |
| x4x ( + / - )   | Номер дома не задан.   |
| x5x ( - )       | Улицы не пересекаются.   |
| x6x ( - )       | Найденной строке не сопоставлен объект на карте.                   |
| x7x ( + )       | Пользователь выбрал номер дома в диалоговом окне Interactive.      |
| 1xx ( + / - )   | Имя найдено только один раз вне уточняющей области.                |
| 2xx ( - )       | Имя найдено в нескольких регионах, но не в уточняющей области.     |
| 3xx ( + / - )   | Имя найдено только один раз, но уточняющая область не была задана. |

| Значения единиц | Смысл  |
|-----------------|--|
| 4xx ( - )       | Имя найдено несколько раз, но уточняющая область не была задана. |
| 5xx ( + )       | Имя найдено несколько раз в уточняющей области.                  |
| 6xx ( + )       | Пользователь выбрал имя области в диалоговом окне Interactive.   |

Определить, какая цифра находится в определенном разряде числа, Вам поможет арифметический оператор Mod. Например, чтобы определить последнюю цифру в числе, используйте деление по модулю (число Mod 10). Чтобы определить две последние цифры в числе, используйте деление по модулю (число Mod 100) и т.д.

Разницу между точным совпадением и приблизительным можно пояснить на следующем примере. Если таблица, содержащая имена городов, содержит одну запись для города "Albany", и задает поиск без уточняющей области, а оператор **Find** использует значение "Albany" как параметр address, то результатом будет полное совпадение. Следующий за таким оператором **Find** вызов функции **CommandInfo(CMD\_INFO\_FIND\_RC)** возвратит 1 (единицу), что означает точное совпадение.

Теперь представим, что оператор действует в режиме использования уточняющей области; другими словами, оператор **Find** ожидает, что название города будет дополнено названием штата (например, "Albany", "NY"). Если программа выполнит оператор **Find** со значениями "Albany" как адреса и нулевой строки в качестве уточняющего названия штата, то, с технической точки зрения, точного совпадения не будет, потому что MapBasic ожидал явного задания уточняющей области. Однако, так как город "Albany" упоминается только в одной записи таблицы, то MapBasic найдет эту запись. Следующий оператор **CommandInfo(CMD\_INFO\_FIND\_RC)** возвратит 301. Цифра 1 обозначает совпадение адреса с названием города из таблицы, а цифра 3 означает частичный успех при поиске уточняющей области.

Если таблица улиц содержит запись для улицы "Main St" и оператор **Find** пытается найти улицу "Main Street", MapBasic оценит результат как частичное совпадение (принимая во внимание использование файла сокращений, смотрите также раздел **Оператор Find Using на стр. 83**). Строго говоря, строки "Main Street" и "Main St" не совпадают. Однако MapBasic определяет равенство этих строк, применяя подстановки из файла сокращений (MAPINFOW.ABB). Следующая за подобной группой операторов **Find** функция **CommandInfo(CMD\_INFO\_FIND\_RC)** возвратит 2.

Если операция поиска сопровождается диалогом, в котором пользователь вводит свой текст, то код возврата будет иметь 1 (единицу) в разряде миллионов.

### Пример:

```
Include "mapbasic.def" Dim x, y As Float, win_id As Integer Open Table
"states" InteractiveMap From States win_id = FrontWindow( ) Find Using
states(state) Find "NY" If CommandInfo(CMD_INFO_FIND_RC) >= 1 Then x =
CommandInfo(CMD_INFO_X) y = CommandInfo(CMD_INFO_Y) Set MapWindow win_id
Center (x, y) ' Теперь создадим объект для маркировки найденного.
' Точечный объект помещается на Косметическом слое.
```

```
Insert Into WindowInfo( win_id, WIN_INFO_TABLE) (Object) Values (
CreatePoint(x, y) ) ElseNote "Объект не найден." End If
```

**См. также:**

**Оператор Find Using, CommandInfo( ) функция**

## Оператор Find Using

### Назначение

Устанавливает, какие таблицы и какие колонки будут рассматриваться при последующем поиске оператором Find.

### Синтаксис

```
Find Using table ( column )
[ Refine Using table ( column ) ]
[ Options
  [ Abbrs { On | Off } ]
  [ ClosestAddr { On | Off } ]
  [ OtherBdy { On | Off } ]
  [ Symbol symbol_style ] ]
[ Inset inset_value { Percent | Distance Units dist_unit } ]
[ Offset value ] [ Distance Units dist_unit ] ]
```

*table* – имя открытой таблицы;

*column* - имя колонки в таблице.

*symbol\_style* – величина типа Symbol для задания стиля символа, которым будет отмечен найденный объект, если пользователь выполнит команду **Запрос > Найти**.

*inset\_value* – положительное целое значение, определяющее величину смещения от концов улиц для размещения адреса;

*value* – - определяет значение отступа Offset (расстояние от улицы - в глубину от улицы);

*dist\_unit* – строка, определяющая название единиц измерения расстояний (например, "mi" для миль или "m" для метров).

### Описание

Оператор **Find Using** определяет таблицу (таблицы) и колонку (колонки), в которых будет осуществлять поиск **Оператор Find**. Колонки должны быть индексированы.

Предложение **Refine** используется для назначения второй таблицы, уточняющей поиск. Вторая таблица должна содержать объекты типа "область". Заданная для уточнения поиска колонка должна быть индексирована. Если пропустить параметр **Refine**, то последующие операторы будут искать просто название (например, "Portland"). Если же параметр **Refine** задан, то последующие операторы Find будут искать название объекта и название района (например, "Portland", "OR").

Предложение **Abbrs** определяет, будет ли применяться файл сокращений в процедурах поиска. По умолчанию режим применения файла сокращений включен (**On**); чтобы отключить его, нужно явно задать предложение **Abbrs Off**.

Предложение **ClosestAddr** задает режим автоматической подстановки ближайшего адреса в случае не достижения точного совпадения. По умолчанию режим подстановки ближайшего адреса выключен (**Off**); чтобы включить его, нужно явно задать предложение **ClosestAddr On**.

Предложение **OtherBdy** задает режим автоматической подстановки адреса из другого региона, если в заданном уточняющем регионе ничего подходящего не было найдено. По умолчанию этот режим выключен (**Off**); чтобы включить его, нужно явно задать предложение **OtherBdy On**.

MapInfo сохраняет последние установки для **Inset** и **Offset**, сделанные пользователем в настройках поиска в диалогах команд **Запрос > Найти, Таблица > Геокодирование** или заданные последним оператором **Find Using**. Таким образом, последние настройки становятся параметрами по умолчанию для следующего сеанса работы.

Если указан параметр **Percent**, то смещение определяется как процент от длины улицы. Для **Percent** возможны следующие значения *inset\_value*: от 0 до 50. Если указан параметр **Distance Units**, то *inset\_value* представляет расстояние от концов улицы для размещения адреса. В этом случае возможны следующие значения *inset\_value*: от 0 до 32,767. Этот параметр используется для того, чтобы символы найденных объектов, расположенных близко к концу улицы сдвигались на экране, более равномерно распределяясь к центру улицы.

Значение **Offset** задает глубину размещения символа найденного от улицы, чтобы также избежать кучности близкорасположенных символов. *Value* – положительная целая величина, определяющая как далеко "отступать" от улицы для размещения адреса. Возможные значения - от 0 до 32,767.

### Пример:

```
Find Using city_1k(city)
  Refine Using states(state)

Find "Albany", "NY"
```

### См. также:

[Оператор Create Index](#), [Оператор Find](#)

---

## Оператор Fix( )

### Назначение

Возвращает целое число, полученное из целой части действительного числа.

### Синтаксис

```
Fix( num_expr )
```

*num\_expr* – числовое выражение

**Возвращаемая величина**

Целое число типа Integer.

**Описание**

Функция **Fix( )** отсекает дробную часть от действительного числа, и возвращает целую часть. Функция **Fix( )** и **Функция Int( )** похожи, но не идентичны. Функции различаются способом удаления дробной части отрицательного числа. Для отрицательного числа функция **Fix( )** возвращает ближайшее целое, больше или равное оригиналу. Например:

```
Fix(-2.3)
```

возвращает значение -2. Функция же **Int( )** возвращает ближайшее целое, меньше или равное оригиналу. Например:

```
Int(-2.3)
```

возвращает значение -3.

**Пример:**

```
Dim i_whole As Integer
i_whole = Fix(5.999)
' i_whole now has the value 5.

i_whole = Fix(-7.2)
' i_whole now has the value -7.
```

**См. также:**

**Функция Int( ), Функция Round( )**

---

**Предложение Font****Назначение**

Определяет шрифт для текстов.

**Синтаксис**

```
Font font_expr
```

*font\_expr* – описание шрифта, величина или имя переменной типа Font, вызов функции, такой как **MakeFont(fontname, style, size, fgcolor, bgcolor)**.

**Описание**

Параметром этого оператора **Font** можно задать стиль оформления текста. **Font** не является отдельным оператором, а используется в других операторах, где требуется задать стиль оформления текста (шрифта). Например, **Оператор Create Text** позволяет установить атрибуты шрифта будущего текстового объекта: имя шрифта, написание, размер. Если в

операторе Create Text опустить параметр **Font**, то для новых текстовых объектов будет использован текущий шрифт MapInfo Professional. За словом **Font** следовать выражение, имеющее значение типа Font,

например, переменная, тип которой объявлен как Font:

```
Font font_var
```

или вызов функций, возвращающих такой тип данных (например, **Функция CurrentFont( )** или **Функция MakeFont( )**):

```
Font MakeFont("Helvetica", 1, 12, BLACK, WHITE)
```

Некоторые операторы MapBasic (например, **Оператор Set Legend**) требуют, чтобы за словом **Font** могли следовать заключенные в скобки пять параметров, описывающих стиль: fontname, style, point\_size, foreground\_color, background\_color:

```
Font("Helvetica", 1, 12, BLACK, WHITE)
```

В следующей таблице приводится описание этих параметров шрифта:

| Компонента<br>стиля | Описание   |
|---------------------|--|
| font name           | Строковое значение, имя шрифта. Зависит от набора шрифтов, используемых операционной системой, в которой будет работать приложение MapBasic.   |
| style               | Целое число. Управляет написанием: жирным шрифтом, курсивом или подчеркнутым. Смотрите следующую таблицу.  |
| size                | Целое число, величина типа Integer, размер шрифта в пунктах. Размер 12 пунктов соответствует высоте буквы, равной одной шестой дюйма.  |
| foreground color    | Целое число, величина типа Integer, задающее цвет букв по шкале RGB. См. <b>Функция Rnd( ) on page 537</b> .   |
| background color    | Целочисленный код цвета фона штриха. Целое число, величина типа Integer, задающее цвет фона или каймы текста по шкале RGB.<br><br>Для задания прозрачного фона текста просто опустите этот параметр в предложении <b>Font</b> . Например: Font( "Helvetica", 1, 12, BLACK). Если используется <b>Функция MakeFont( )</b> , пятый параметр должен быть равен 1 для задания прозрачного фона текста. |

Следующая таблица содержит значения для параметра style, задающего написание шрифта:

| Значения style | Стиль написания |
|----------------|-----------------|
| 0              | Нормальное      |
| 1              | Жирное          |

| Значения style | Стиль написания      |
|----------------|----------------------|
| 2              | Курсив               |
| 4              | Подчеркнутый         |
| 8              | Перечеркнутый        |
| 32             | Оттененный           |
| 256            | С каймой             |
| 512            | Полная капитализация |
| 1024           | Разреженный          |

Для задания двух или более стилей написания коды складываются. Например, для написания жирным шрифтом и только заглавными буквами параметр style должен иметь значение 513.

#### Пример:

```
Include "MAPBASIC.DEF"Dim o_title As ObjectCreate TextInto Variable
o_title "Здесь может помещаться Ваше сообщение" (73.5, 42.6) (73.67,
42.9)Font MakeFont("Helvetica",1,12,BLACK,WHITE)
```

#### См. также:

**Оператор Alter Object, Функция Chr\$( ), Оператор Create Text, Функция RGB( )**

## Оператор For...Next

### Назначение

Выполняет последовательность действий в цикле с заданным числом итераций.

### Предупреждение

Оператор **For...Next** не может быть выполнен в окне MapBasic.

### Синтаксис

```
For var_name = start_expr To end_expr [ Step inc_expr ]
    statement_list
```

#### Next

*var\_name* – имя переменной, выполняющей роль счетчика цикла;

*start\_expr* – численное выражение;

*end\_expr* – численное выражение;

*inc\_expr* – численное выражение;

*statement\_list* – группа операторов, выполняющихся за одну итерацию цикла.

### Описание

Оператор **For...Next** задает цикл. Управление циклом происходит при помощи целочисленной переменной *var\_name*, выступающей в роли счетчика. Оператор **For...Next** последовательно выполняет группу операторов *statement\_list*) заданное число раз, либо пропускает всю группу *statement\_list* целиком. Выражения *start\_expr*, *end\_expr* и *inc\_expr* определяют, сколько раз будет выполнена группа операторов *statement\_list*.

Выполнение цикла **For...Next** MapBasic начинает с присваивания значения *start\_expr* переменной *var\_name*. Если это значение меньше или равно *end\_expr*, MapBasic выполняет набор операторов *statement\_list*, после чего прибавляет значение *inc\_expr* к начальному значению. Если предложение **Step** отсутствует в операторе, то счетчик увеличивается на одну единицу. Затем MapBasic сравнивает это значение со значением *end\_expr*, если счетчик меньше или равен *end\_expr*, MapBasic снова повторяет операторы из списка *statement\_list*. Если же *var\_name* больше, чем *end\_expr*, MapBasic останавливает цикл, и передает управление оператору, следующему за оператором **Next**.

Оператор **For...Next** может также работать в обратную сторону, используя отрицательное значение шага **Step**. В этом случае, каждый проход цикла For уменьшает значение *var\_name* и MapBasic повторяет операции цикла до тех пор, пока *var\_name* остается большим или равным *end\_expr*.

Каждое предложение **For** должно заканчиваться парным **Next**. Все операторы, которые находятся между операторами **For** и **Next**, являются операторами тела цикла *statement\_list* и выполняются за одну итерацию (проход) цикла.

Используйте **Оператор Exit For** для немедленного прекращения цикла, независимо от значения счетчика *var\_name*. **Оператор Exit For** просто передает управление программы следующему оператору после слова **Next**.

MapBasic позволяет Вам изменять величину переменной *var\_name* в теле цикла и тем самым влиять на выполнение цикла. Но для соблюдения хорошего стиля программирования переменную *var\_name* изменять не рекомендуется.



**Пример:**

```
Dim i As Integer' Сообщение Note будет выдаваться пять разFor i = 1 to 5
Note "Добро пожаловать в MapInfo!"Next' этот цикл выполнится за три
шагаFor i = 1 to 5 Step 2Note "Добро пожаловать в MapInfo!" Next' Этот
цикл выполнится за три шагаFor i = 5 to 1 Step -2 Note "Добро пожаловать
в MapInfo!" Next' MapBasic не сможет выполнить следующий оператор For'
потому что начальное значение ' больше конечногоFor i = 100 to 50 Step 5
Note "Это сообщение никогда не появится"Next
```

**См. также:**

**Оператор Do...Loop, Оператор Exit For**

---

## Процедура ForegroundTaskSwitchHandler

**Назначение**

Процедура, автоматически выполняющаяся при изменении фокуса окна программы MapInfo относительно других программ, выполняющихся в среде Windows.

**Синтаксис**

```
Declare Sub ForegroundTaskSwitchHandler Sub ForegroundTaskSwitchHandler
statement_list End Sub, где
```

*statement\_list* – список операторов процедуры.

**Описание**

Когда пользователь запускает программу, в которой есть процедура

**ForegroundTaskSwitchHandler**, программа не завершается после того, как выполняются все операторы процедуры Main и другие процедуры, вызванные из нее. В процедуре может быть использована **CommandInfo( ) функция** для определения, приобретает ли MapInfo фокус или теряет его.

**Пример:**

```
Sub ForegroundTaskSwitchHandler If CommandInfo(CMD_INFO_TASK_SWITCH) =
SWITCHING_INT0_MAPINFO Then ' ... когда активно окно MapInfoElse ' ...
когда активно окно другой программыEnd If End Sub
```

**См. также:**

**CommandInfo( ) функция**

---

## Функция Format\$( )

**Назначение**

Возвращает строковое представление числа в заданном формате.

## Синтаксис

**Format\$**( *value*, *pattern* )

*value* – численное выражение;

*pattern* – шаблонная строка, задающая формат.

## Возвращаемая величина

Строка

## Описание

Функция **Format\$ ( )** преобразует число в строковую величину по шаблону. Например, число 12345.67 функция **Format\$ ( )** может преобразовать в строку "\$12,345.67".

Параметр *value* задает число, которое нужно отформатировать.

Параметр *pattern* задает шаблон в виде строки, определяющий правила форматирования. Шаблон состоит из специальных символов, управляющих отображением числа в строке, таких как #, 0, %, запятая ( , ), точка ( . ) или точка с запятой ( ; ). Роль каждого из этих символов в формировании результирующей строки приведена в следующей таблице:

| Символ в <i>pattern</i> | Какую роль играет для результата функции   |
|-------------------------|--|
| #                       | Соответствует всем цифрам из <i>value</i> до десятичной точки и одной цифре из <i>value</i> после десятичной точки.  |
|                         | Если шаблон содержит один или более символов # левее десятичного знака, а формируемое число меньше единицы, но больше нуля, то в результате ноль в целых частях не будет отображен, строка будет начинаться с точки.   |
| 0                       | Задаёт место (подобно #), но только для одного цифрового символа. Если управляющая строка содержит один или более символов 0 левее десятичной точки и значение числа меньше единицы и больше нуля, то результирующая строка будет иметь ноль перед десятичной точкой. См. примеры ниже.  |
| . (точка)               | Соответствует десятичной точке. Число символов правее точки определяет число десятичных разрядов, то есть, сколько будет выведено символов после десятичной точки. Результирующая строка будет включать в себя тот десятичный разделитель, который установлен в Вашей системе. Чтобы назначить в качестве десятичного разделителя точку, даже если в системе принят другой знак, используется <b>Опепатор Set Format</b> . |

| Символ в pattern    | Какую роль играет для результата функции   |
|---------------------|--|
| , (запятая)         | Разделитель тысяч. Если управляющая строка содержит запятую перед знаком решетки, определяющей целую часть числа, то в результате каждые три цифры будут отделены запятой. Например, десять миллионов будут показаны так: "10,000,000". Результирующая строка будет включать в себя тот десятичный разделитель, который установлен в Вашей системе. Чтобы явно задать запятую в качестве разделителя тысяч, используйте оператор <b>Set Format</b> . |
| %                   | Процентное представление числа. Если управляющая строка содержит знак процента, то результатом будет строка с числом, умноженным на сто и знаком процента справа. Например, число 0.75 будет преобразовано в строку "75%". Если Вы не намерены преобразовывать число в проценты, но хотите иметь в строке знак %, используйте его в комбинации со знаком \ (обратный слэш).  |
| E+, e+, E, e        | Представление числа в научном (экспоненциальном) формате. Например, число 1234 будет преобразовано в строку "1.234e+03". Если экспонента положительна, то после символа "e" будет помещен плюс. Если экспонента отрицательна, после символа "e" будет помещен знак минус.  |
| E-, e-              | Представление числа в научном (экспоненциальном) формате. От предыдущего отличается тем, что в случае положительной экспоненты знак плюс не отображается.  |
| ; (точка с запятой) | Знаком "точка с запятой" Вы разделяете в одной управляющей строке два шаблона для положительного и отрицательного значения. Так, в управляющей строке сначала определяется формат для представления положительных значений числа. Затем через точку с запятой задается другой формат для отрицательных значений. Если необходимо иметь знак минус, включите "-" в формат для отрицательных значений.   |
| \                   | Обратный слэш инструктирует MapBasic не обрабатывать символ, следующий непосредственно за ним. Если обратный слэш стоит перед специальным символом (например, перед %), то MapBasic будет рассматривать этот символ в шаблоне как текстовый символ.  |

### Ошибки:

В результате выполнения функции Вы можете получить код ошибки ERR\_FCN\_INVALID\_FMT, если неправильно задан параметр pattern.

### Примеры

Следующие примеры показывают, какие результаты получаются при применении различных строковых шаблонов. Результаты показаны в комментариях.

**Внимание:** Внимание: в локализованных системах Вы можете получить те же результаты, но немного в другом виде.

```
Format$( 12345, ",#") ' возвращает "12,345"
Format$(-12345, ",#") '
возвращает "-12,345"
Format$( 12345, "$#") ' возвращает "$12345"
Format$(-12345, "$#") ' возвращает "-$12345"
Format$( 12345.678, "$,###") '
возвращает "$12,345.68"
Format$(-12345.678, "$,###") ' возвращает "-$12,345.68"
Format$( 12345.678, "$,###;($,###)") ' возвращает
"$12,345.68"
Format$(-12345.678, "$,###;($,###)") ' возвращает
"($12,345.68)"
Format$(12345.6789, ",#####") ' возвращает
"12,345.679"
Format$(12345.6789, ",#.##") ' возвращает "12,345.7"
```

```
Format$(-12345.6789, "#####E+00") ' возвращает "-1.235e+04"
Format$( 0.054321, "#####E+00") ' возвращает "5.432e-02"
Format$(-12345.6789, "#####E-00") ' возвращает "-1.235e04"
Format$( 0.054321, "#####E-00") '
возвращает "5.432e-02"
Format$(0.054321, "###%") ' возвращает
"5.43%"
Format$(0.054321, "###\%") ' возвращает ".05%"
Format$(0.054321, "0.##\%") ' возвращает "0.05%"
```

**См. также:**

**Функция Str\$( )**

---

## Функция FormatDate\$( )

### Назначение

Возвращает дату в укороченном формате, в том стиле, который определен в Панели управления.

### Синтаксис

Значение FormatDate

*value* - число или строка, представляющие дату в формате YYYYMMDD.

### Возвращаемая величина

Строка

### Описание

Функция **FormatDate\$( )** возвращает строковую величину, представляющую дату с локальным системном формате, как определено в Панели управления (Windows Control Panel).

Если вы зададите год двумя цифрами (например, 96), то MapInfo Professional припишет этот год к веку, который задает **Оператор Set Date Window**.

В большинстве случаев, год можно задавать двумя цифрами. (Для более точной настройки используется Date window ("годовой диапазон")). См. **Функция DateWindow( )** на стр. 19.

### Примеры

Предположим, что настройки Панели управления это d/m/y для даты, '-' это разделитель для даты и "dd-МММ-уууу" - это укороченный формат даты:

```
Dim d_Today As Dated_Today = CurDate( )Print d_Today 'возвращает
"19970910"Print FormatDate$( d_Today ) 'возвращает "10-Sep-1997"Dim
s_EnteredDate As Strings_EnteredDate = "03-02-61"Print FormatDate$(
s_EnteredDate ) 'возвращает "03-Feb-1961"s_EnteredDate = "12-31-61"Print
FormatDate$( s_EnteredDate ) ' возвращает ERROR: not d/m/y
orderings_EnteredDate = "31-12-61"Print FormatDate$( s_EnteredDate ) '
возвращает 31-Dec-1961"
```

См. также:

**Функция DateWindow( ), Оператор Set Date Window**

## Функция FormatNumber\$( )

### Назначение

Форматирование числа с использованием символов десятичной точки и разделителя тысяч – тех, которых в данное время использует система, в которой выполняется приложение.

### Синтаксис

**FormatNumber\$( num )**

*num* – численная или строковая величина, представляющее число, например, "1234.56".

### Возвращаемая величина

Строка

### Описание

Функция возвращает строку, представляющую число. Если число достаточно велико, то функция вставит разделители тысяч. MapInfo Professional прочитывает конфигурацию пользовательской системы и определяет, какие символы используются для десятичной точки и разделителя тысяч.

### Примеры

Следующая таблица демонстрирует, как работает функция **FormatNumber\$( )** в компьютере, в котором точка обозначает десятичную точку, а запятая – разделитель тысяч (стандартная установка в США):

Функция FormatTime\$

| Вызов функции                | Результат                                |
|------------------------------|--|
| FormatNumber\$ ("12345.67")  | "12,345.67" (добавлен разделитель тысяч) |
| FormatNumber\$ ("12,345.67") | "12,345.67" (без изменений)              |

Если в компьютере знак запятой зарезервирован за десятичной точкой, а точка разделяет тысячи, то функция будет работать так:

| Вызов функции                | Результат  |
|------------------------------|--|
| FormatNumber\$ ("12345.67")  | "12.345,67" (добавлен разделитель тысяч и изменен знак десятичной точки) |
| FormatNumber\$ ("12,345.67") | "12.345,67" (изменены оба знака)   |

См. также:

Функция DeformatNumber\$( )

Функция FormatTime\$

Назначение

Возвращает строку с временем в формате, заданным вторым аргументом. Форматирующая строка должна удовлетворять стандартам Microsoft на настройки местного времени:

| Часы    | Смысл  |
|---------|--|
| H       | Часы без нулей перед значениями часов (12-часовой-отсчет). |
| hh      | Часы с нулями перед значениями часов (12-часовой-отсчет).  |
| H       | Часы без нулей перед значениями часов (24-часовой-отсчет). |
| HH      | Часы с нулями перед значениями часов (24-часовой-отсчет).  |
| Минуты  | Смысл  |
| m       | Минуты без нулей перед значениями минут.                   |
| mm      | Минуты с нулями перед значениями минут.                    |
| Секунды | Смысл  |
| s       | Секунды без нулей перед значениями секунд.                 |
| ss      | Секунды с нулями перед значениями секунд.                  |

| Time marker | Смысл  |
|-------------|--|
| t           | Строка отметки времени, состоящая из единственного символа.<br><br><b>Внимание:</b> В некоторых языках не применяется, например, японском (Япония). В случае использования такого формата отметка времени, заданная системными параметрами LOCALE_S1159 (AM) и LOCALE_S2359 (PM), сокращается до единственного символа. Поэтому приложение может задать неправильное форматирование, когда одна и та же строка будет использоваться и для значений AM, и для PM. |
| tt          | Строка отметки времени из нескольких символов.   |

Источник: <http://msdn2.microsoft.com/en-us/library/ms776320.aspx>

**Внимание:** В предыдущих форматах буквы m, s и t обязательно должны вводиться в нижнем регистре; буква h в нижнем регистре предназначена для 12-часового отсчета, в то время, как в верхнем регистре указывает на 24-часовой.

Используемый нами код отвечает правилам, установленным для системного местного формата времени. Кроме того, допускается использование f, ff или fff для десятых, сотых долей секунды или миллисекунд.

**Синтаксис**

FormatTime\$ (Time, String)

**Возвращаемая величина**

Строка

**Пример:**

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim Z as time
Z = CurTime()
Print FormatTime$(Z, "hh:mm:ss.fff tt")
```

**Оператор FME Refresh Table**

**Назначение**

Обновляет таблицу Universal Data Source (FME) значениями исходной таблицы

**Синтаксис**

**FME Refresh Table** *alias*, где  
*alias* - псевдоним, выбранный для зарегистрированной таблицы Universal Data Source (FME).

## Функция FrontWindow( )

---

### Пример:

Следующий пример демонстрирует обновление локальной таблицы под названием watershed.

```
FME Refresh Table watershed
```

## Функция FrontWindow( )

---

### Назначение

Возвращает идентификатор активного окна.

### Синтаксис

```
FrontWindow( )
```

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция **FrontWindow( )** возвращает целочисленный идентификатор самого верхнего открытого окна в MapInfo. Этот оператор полезно применять сразу после создания нового окна (Карты, Графика, Списка или Отчета), чтобы запомнить значение идентификатора самого верхнего окна.

### Пример:

```
Dim map_win_id As Integer  
Open Table "states"  
Map From states  
map_win_id = FrontWindow( )
```

### См. также:

[Функция NumWindows\( \)](#), [Функция WindowID\( \)](#), [Функция WindowInfo\( \)](#)

## Оператор Function...End Function

---

### Назначение

Объявляет внутреннюю функцию.

### Предупреждение

Оператор **Function...End Function** не может быть выполнен в окне MapBasic.

### Синтаксис

```
Function name ( [ [ ByVal ] parameter As datatype ]  
               [, [ ByVal ] parameter As datatype... ] ) As return_type
```



```
statement_list
End Function
```

*name* – имя функции;

*parameter* – имя параметра функции;

*datatype* – тип данных, стандартный в MapInfo, или тип, созданный при помощи оператора Type;

*return\_type* – стандартный для MapInfo скалярный тип для величины, полученной в результате.

*statement\_list* – группа операторов, составляющая тело функции.

## Описание

Оператор **Function...End Function** позволяет Вам создавать свои функции. Созданные таким образом функции являются внутренними и могут использоваться в программе, в которой находятся наравне со стандартными функциями MapInfo.

Каждое имя функции, созданной при помощи оператора **Function...End Function**, а также типы параметров и результата должны быть ранее объявляет **Оператор Declare Function**.

Внутренняя функция подобна процедуре, сделанной при помощи оператора **Sub**, но в отличие от процедуры функция возвращает значение. Функция по сравнению с процедурой в применении более гибка, т.к. Вы можете составлять выражения, содержащие вызов к одной и более функциям. Например, в следующем выражении дважды вызывается **Функция Proper\$()**:

```
fullname = Proper$(firstname) + " " + Proper$(lastname)
```

В конструкции оператора **Function...End Function** параметр *name*, определяющий имя функции, трактуется как имя переменной. Этой переменной и присваивается значение, возвращаемое функцией. Если переменной *name* не было ничего присвоено, функция возвращает нулевое значение для числовых функций, FALSE для логических функций или нулевую строку ("" ) для строчных функций.

## Ограничения при передаче параметров

Результат функции может быть только скалярным. Другими словами, функция не может возвращать массивы и данные сложных типов, созданных оператором Type. По умолчанию, каждый параметр во внутреннюю функцию передается "ссылкой" ("by-reference"). Это означает, что оператор, вызывающий функцию, должен в качестве параметров использовать имена переменных. Если функция изменяет значение параметра, передаваемого ссылкой, то изменяется и переменная в вызывающей процедуре.

Любой или все параметры функции могут передаваться "значением", если перед ними явно указано ключевое слово **ByVal** в определении оператора **Function...End Function**. В этом случае оператор, вызывающий функцию, может использовать в качестве параметра целое выражение, а не одно только имя переменной. Однако, если изменить значение такого параметра, новое значение нельзя будет вернуть в вызывающий модуль. Вы не можете передавать в виде значения ( **ByVal**) во внутреннюю функцию массивы, а также переменные, заданные оператором. sub-процедуры. Если функция не должна иметь параметров, то

оператор **Function...End Function** может, включать пустые скобки или не использовать их вообще. Рекомендуется (и обязательно при вызове), однако, использовать в таких случаях пару пустых скобок. Например, задавая функцию Foo оператором **Function...End Function**, Вы можете использовать запись:

```
Function Foo( ) ' ... список операторов функции ...End Function
```

которая идентична конструкции:

```
Function Foo' ... список операторов функции ...End Function
```

Но при вызове этой функции использование скобок обязательно:

```
var_name = Foo( )
```

### Доступность функций пользователя

Пользователь не может помещать вызовы к внутренним функциям выполняющейся прикладной программы при заполнении стандартных диалогов в MapInfo. Однако сама программа может это делать. Так, пользователь не может обратиться к определенной программой функции из диалога команды SQL-запрос, однако, в скомпилированной MapBasic-программе может содержаться **Оператор Select**, использующий эту функцию.

Внутреннюю функцию можно вызывать только в той программе, в которой она была определена. Поэтому, если Вы хотите использовать такую функцию в другой программе, то Вам придется полностью скопировать определение **Function...End Function** в текст другой программы.

### Имена функций

Имя функции, определенное оператором **Function...End Function**, может совпадать с именем стандартной функции MapBasic, таким как **Abs** или **Chr\$**. Подобная функция полностью замещает одноименную стандартную в пределах прикладной программы, в которой она определена. Например, если Вы создадите свою функцию **Abs**, то все вызовы функции **Abs** будут обращаться к Вашей функции, а стандартная **Функция Abs( )** использоваться не будет.

Если произошло описанное переопределение, то на другие программы оно не действует. Так, если Вы пишете несколько программ и планируете, чтобы в каждой была своя особая **Функция Distance( )**, то в каждой из программ Вы должны отдельно задавать определение **Function...End Function**.

Когда в программе MapBasic переопределяется стандартная функция, то это переопределение распространяется на всю программу. В каждой процедуре программы вызовы будут адресоваться к переопределенной пользователем функции, а не к стандартной.

### Пример:

В этом примере определяется функция пользователя под названием "CubeRoot", которая вычисляет кубический корень из числа. Так как функция CubeRoot вызывается в тексте программы до определения самой функции, то в этом примере используется **Оператор Declare Function**, определяющий аргумент функции и тип возвращаемого значения.

```
Declare Function CubeRoot(ByVal x As Float) As Float  
Declare Sub Main
```

```
Sub Main
    Dim f_result As Float
    f_result = CubeRoot(23)
    Note Str$(f_result)
End Sub

Function CubeRoot(ByVal x As Float) As Float
    CubeRoot = x ^ 0.333333333333
End Function
```

**См. также:**

**Оператор Declare Function, Оператор Declare Sub, Оператор Sub...End Sub**

## Оператор Geocode

### Назначение

С помощью удаленной службы геокодирования, для соединения с которой использовался **Оператор Open Connection**, а для настройки **Оператор Set Connection Geocode**, будет геокодирована таблица (или отдельная запись).

### Синтаксис

**Geocode** *connection\_number*

#### Input

```
[ Table input_tablename ]
[ Country = Country_expr
  [ Street = Street_expr,
    [ IntersectingStreet = IntersectingStreet_expr ],
    Municipality = Municipality_expr,
    CountrySubdivision = CountrySubdiv_expr,
    PostalCode = PostalCode_expr,
    CountrySecondarySubdivision = CountrySecondarySubdiv_expr,
    SecondaryPostalCode = SecondaryPostalCode_expr,
    Placename = Placename_expr,
    Street2 = Street2_expr,
    MunicipalitySubdivision = MunicipalitySubdiv_expr ] ]
```

#### Output

```
[ Into
  [ Table out_tablename [ Key out_keycolumn = in_keyexpr ] ] |
  [ Variable variable_name ] ]
[ Point [ On | Off ] [ Symbol Symbol_expr ], ]
[ Street_column = Street, Municipality_column = Municipality,
  CountrySubdiv_column = CountrySubdivision,
  PostalCode_column = PostalCode,
  CountrySecondarySubdiv_column = CountrySecondarySubdivision,
  SecondaryPostalCode_column = SecondaryPostalCode,
  Placename_column = Placename,
  MunicipalitySubdiv_column = MunicipalitySubdivision,
  Country_column = Country, ResultCode_column = ResultCode,
  Latitude_column = Latitude, Longitude_column = Longitude
  Columns colname = geocoder_keyname [,] ...]
[ Interactive [ On [ Max Candidates candidates_expr | All ]
  [ CloseMatchesOnly [ On | Off ] ]
  | Off [ First | None ] ]
```

*connection\_number* – число полученное при создании соединения. См. **Оператор Open Connection on page 478**.

*input\_tablename* синоним открытой таблицы, включая результаты запросов и выборки (selection ).

*Country\_expr* – строка, определяющая трехбуквенный код страны по списку ISO.

*Street\_expr* – строка, задающая улицу адреса.

*IntersectingStreet\_expr* – выражение, определяющее улицу, пересекающую улицу, адрес которой задан в *Street\_expr*.

*Municipality\_expr* – выражение, в котором задано название муниципального образования.

*CountrySubdivision\_expr* – выражение, в котором задано название территориальной или административной единицы страны самого старшего уровня. Например, в США здесь может быть задано название штата. В Канаде – название провинции.

*PostalCode\_expr* – выражение, в котором задан код почтового индекса.

*CountrySecondarySubdiv\_expr* – выражение, в котором задано название территориальной или административной единицы страны следующего после самого старшего уровня. Например, в США здесь можно задать название графства, а в Канаде – название переписного участка.

*SecondaryPostalCode\_expr* – выражение, в котором задан код почтового индекса второго уровня. В США этому параметру соответствует код дополнительной части формы ZIP+4 кода почтового индекса ZIP.

*Placename\_expr* – выражение, в котором можно задать название хорошо известной достопримечательности, например здания, которое может иметь несколько адресов.

*Street2\_expr* – выражение, в котором задан второй уровень адреса улицы (в англоязычной практике указывается на отдельной строке почтового отправления).

*MunicipalitySubdiv\_expr* – выражение, в котором задано название муниципального образования младшего уровня.

*out\_tablename* – синоним таблицы, которая будет использоваться для хранения результатов геокодирования.

*out\_keycolumn* – строка, в которой задаётся имя ключевого поля таблицы с результатами, в котором будут храниться "ключевые" значения исходных записей. Используется для выделения записей по происхождению исходных данных геокодирования.

*in\_keyexpr* – выражение из (исходной таблице), значение которого будет вставлено в геокодированную запись.

*variable\_name* – имя переменной с единственным геометрическим примитивом.

*Symbol\_expr* – выражение для символа условного знака, который будет использоваться для отображения Точки из колонки с геометрическими примитивами. Более подробную информацию см. в разделе [Предложение Symbol on page 725](#).

*Street\_column* – синоним имени колонки, в которой будут храниться результаты с названиями улиц.

*Municipality\_column* – строка с именем колонки, в которой будут храниться результаты с названиями муниципальных образований.

*CountrySubdiv\_column* – строка с именем колонки, в которой будут храниться результаты с названиями территориальных или административных образований страны.

*PostalCode\_column* – строка с именем колонки, в которой будут храниться результаты с кодами почтовых индексов.

*CountrySecondarySubdiv\_column* – строка с именем колонки, в которой будут храниться результаты с названиями территориальных или административных образований второго уровня.

*SecondaryPostalCode\_column* – строка с именем колонки, в которой будут храниться результаты с дополнительными кодами почтовых индексов.

*Placename\_column* – строка с именем колонки, в которой будут храниться результаты с названиями достопримечательностей.

*MunicipalitySubdiv\_column* – строка с именем колонки, в которой будут храниться результаты с названиями младших муниципальных образований.

*Country\_column* – строка с именем колонки, в которой будут храниться результаты с названиями стран.

*ResultCode\_column* – строка с именем колонки, в которой будут храниться коды результатов порожденные геокодером.

*Latitude\_column* – строка с именем вещественной или десятичной колонки, в которой будут храниться "Широты" результатов.

*Longitude\_column* – строка с именем вещественной или десятичной колонки, в которой будут храниться "Долготы" результатов.

*colname* – строка с именем колонки для результатов, характерных для используемого геокодера.

*geocoder\_keyname* – строка с названием параметра страны, использованным при геокодировании. Эти параметры описаны в документации каждого геокодера.

*candidates\_expr* – выражение для количества кандидатур, которые будут возвращены в интерактивном сеансе.

### Описание

Каждый оператор **Geocode** должен обязательно содержать и выражение **Input**, и выражение **Output**. Параметр, заданный выражением *input\_tablename*, является необязательным, однако, если уж таблица не задана, то геокодирование будет проводиться по набору исходных строк (переменных или констант), вследствие чего по каждому запросу будет геокодирован единственный адрес. Параметр, заданный выражением *output\_tablename* также является необязательным. Смотрите **Разница между табличной и не-табличной формой исходных и выходных данных** ниже.

### Предложение Input

Предложение **Input** необходимо, поскольку для геокодирования требуются некоторые данные на входе.

Аргумент параметра **Country** должен быть указан либо явно, либо в виде ссылки на колонку в *input\_tablename*. Когда геокодируются данные для единственной страны, то этот параметр может быть константой, если отсутствуют дополнительные сведения. Необходимо использовать стандартные трехбуквенные коды стран, одобренные ISO.

Список полей *input\_table*, по которым следует выполнять геокодирование должен содержать хотя-бы одно значение. Чем больше выражений составляют запрос, тем более точными будут результаты геокодирования.

### Предложение Output

Предложение **Output** обязательно, поскольку при его отсутствии, результатом выполнения всего оператора будет – ничего.

**Into Table** определяет, что предложение **Output** ссылается на колонки не защищенной от записи таблицы *output\_table*. Если не объявлено явно, это предложение обращается к *input\_table*. Обратите внимание, если требуется, чтобы исходные колонки были обновлены, то необходимо объявлять их **ОДНОВРЕМЕННО** и в аргументе *input*, и в аргументе *output*.

Аргумент **Key** используется вместе с **Into Table**. Этим аргументом устанавливается связь между ключевыми колонками исходной таблицы и таблицы на выходе.

Аргумент **Variable** определяет, что геометрические примитивы после выполнения геокодирования будут храниться в переменной заданной аргументом *variable\_name*. Обратите внимание, что в случае, когда в качестве исходной используется таблица, и заданы дополнительные параметры для результатов, будет обработана только первая запись, а все остальные пропущены.

**Аргумент Point** указывает, что географический результат геокодирования будет хранить либо в таблице, либо в переменной. Если используется таблица, то для неё должна быть предусмотрена возможность присоединять географическую.

Для того чтобы хранить точки в колонке объектов, используйте либо аргумент **Point**, либо **Point On** (по умолчанию используется вариант "on") . Будет использоваться стандартный символ условного знака. Для того чтобы задать определенный символ условного знака, используйте последовательность аргументов **Point On Symbol** *symbol\_expr*. Если не требуется хранить точки в колонке объектов, используйте аргумент **Point Off**. В любом случае: будет создан объект или нет, можно будет хранить координаты *x* и *y* в численных колонках. Для этого, определите такие колонки в аргументах **Latitude** = *latitude\_column* **Longitude** = *longitude\_column*.

Остальные аргументы результатов определяют колонки выходной таблицы, в которых заранее известные геокодеры будут хранить полученные результаты. В целом, эти аргументы более своеобразны, чем аргументы исходных параметров. Например, можно геокодировать адрес, используя только коммерческое название "MapInfo" и код почтового индекса "12180". Однако, в результате будет получено гораздо больше данных. Дополнительный аргумент **Columns** позволяет использовать характерные для геокодера данные. Пользователь должен заранее знать имена ключей, определяемых геокодером.

## Разница между табличной и не-табличной формой исходных и выходных данных

Оператор **Geocode** может использоваться с исходными и выходными данными в форме таблиц или в не-табличной форме. Если в качестве исходных данных используется таблица, то результаты можно хранить в существующей или новой таблице или в переменных. Если для результатов используется переменная, то обрабатывается только первая запись, а сохраняться будет только географический объект.

Если в качестве исходных, используются данные не в форме таблицы, то аргументы оператора должны быть либо выражениями (не имена колонок), либо переменными, либо константами, а результаты могут либо храниться в таблице, либо присвоены переменной.

## Предложение Interactive

**Interactive** [ **On** | **Off** ] – параметры дополнительного аргумента, с помощью которого можно вызывать диалог, когда возвращаются несколько кандидатов каждого адреса. В этом случае, пользователь сможет выбрать, оформить запрос по-другому, пропустить или вовсе отменить обработку исходных данных.

Если сомневаетесь, пропустите обработку данных. Если используется параметр **On** – при возникновении описанной ситуации, появится диалог. Если используется параметр **Off**, и будет выявлено несколько совпадений, то будут использоваться либо первый полученный вариант, либо все варианты будут пропущены. По умолчанию все полученные результаты будут пропущены.

Случай когда аргумент **Interactive** пропущен, эквивалентен последовательности параметров аргумента **Interactive Off None**, и дополнительные параметры не могут быть заданы. Если используется аргумент **Interactive**, по умолчанию параметру присваивается значение **On**.

- **Interactive** – эквивалентно **Interactive On**. Если не предусмотрено никакого значения параметра **Max** – по умолчанию будет возвращено три (3) кандидатуры.
- **Interactive On Max Candidates All** – будут возвращены все кандидатуры
- **Interactive On Max Candidates**  $4 * \text{myMBVariable} / 6$  – будут возвращены кандидатуры в количестве, определенном выражением.
- **Interactive Off** – эквивалентно **Interactive Off None**.
- **Interactive Off First** – будет возвращена первая кандидатура списка.

Параметр **CloseMatchesOnly** – передает серверу команду возвращать только близкие совпадения, по правилам заданным для этого сервера. Если параметр **CloseMatchesOnly** установлен в состояние **Off**, – все возвращенные результаты вплоть до количества, определенного параметром аргумента **Max Candidates** вместе со считающимися близкими совпадениями, будут отмечены как близко совпадающие.

## Примеры

В следующем примере приведен запрос на геокодирование по таблице `nystreets`, из которой используются колонки "city, Streetname, state и postalcode".

```
Geocode connectionHandle Input Table nystreets municipality=city,
street=StreetName, countrysubdivision=state, postalcode=zip,
country="usa" OUTPUT StreetName=street, address=municipality
```



В этом примере приведен запрос на геокодирование по таблице nystreets и задан символ условного знака для полученных результатов.

```
Geocode connectionHandle Input Table nystreets street=StreetName,
country="usa"
Output Point Symbol MakeFontSymbol(65, 255 ,24,"MapInfo
Cartographic",32,0), StreetName=street
```

В этом примере запрос содержит аргумент Interactive в состоянии On, а возвращенные результаты будут помещены в колонку "street".

```
Geocode connectionHandle Input Table nystreets street=StreetName,
country="usa"
Output Point Symbol MakeFontSymbol(65, 255 ,24,"MapInfo
Cartographic",32,0),
StreetName=street Interactive on Max Candidates 5
```

В следующем примере показан вариант запроса на геокодирование без использования таблиц, с сохранением результатов в переменной:

```
Geocode connectionHandle Input street="1 Global View", country="usa",
countrysubdivison="NY", municipality="Troy" Output Variable outvar
```

**См. также:**

**Оператор Open Connection**

---

## Функция GeocodeInfo( )

### Назначение

Возвращает любой или все атрибуты соединения, для создания которого использовался **Оператор Set Connection Geocode**. Кроме того, **GeocodeInfo( )** может использоваться для получения некоторых параметров состояния после выполнения последней команды геокодирования по каждому из соединений. Вы также можете задавать атрибут, управляющий максимальным количеством адресов, которое допускается одновременно передавать от сервера службе.

### Синтаксис

```
GeoCodeInfo( connection_handle, attribute )
```

*connection\_handle* – целое.

*attribute* это целое, определяющее, какого типа информация будет возвращена.

### Возвращаемая величина

Вещественное (Float), целое (Integer), короткое целое (SmallInt), логическое (Logical), или строка символов (String), в зависимости от параметра атрибута.

**Описание**

Функция **GeoCodeInfo( )** возвращает характеристики соединения, используемые по умолчанию, или параметры, заданные оператором **Set GeoCode**. Подобно многим другим функциям MapBasic этого типа, возвращаемые значения могут зависеть от параметра *attribute*. Коды всех значений перечислены в MAPBASIC.DEF.

| Значение атрибута            | Возвращаемое значение GeoCodeInfo( )   |
|------------------------------|--|
| GEOCODE_STREET_NAME          | логическое, определяющее задано или нет совпадение по названиям улиц <b>StreetName</b> .   |
| GEOCODE_STREET_NUMBER        | логическое, определяющее задано или нет совпадение по номерам улиц <b>StreetNumber</b> .   |
| GEOCODE_MUNICIPALITY         | логическое, определяющее задано или нет совпадение по названиям муниципальных образований.   |
| GEOCODE_MUNICIPALITY2        | логическое, определяющее задано или нет совпадение по <b>MunicipalitySubdivision</b> .   |
| GEOCODE_COUNTRY_SUBDIVISION  | логическое, определяющее задано или нет совпадение по <b>CountrySubdivision</b> .  |
| GEOCODE_COUNTRY_SUBDIVISION2 | логическое, определяющее задано или нет совпадение по <b>CountrySecondarySubdivision</b> .   |
| GEOCODE_POSTAL_CODE          | логическое, определяющее задано или нет совпадение по <b>PostalCode</b> .  |
| GEOCODE_DICTIONARY           | короткое целое ( <b>SmallInt</b> ), принимающее одно из пяти значений: <ul style="list-style-type: none"> <li>• <b>DICTIONARY_ALL</b></li> <li>• <b>DICTIONARY_ADDRESS_ONLY</b></li> <li>• <b>DICTIONARY_USER_ONLY</b></li> <li>• <b>DICTIONARY_PREFER_ADDRESS</b></li> <li>• <b>DICTIONARY_PREFER_USER</b></li> </ul> |
| GEOCODE_BATCH_SIZE           | целое, определяющее объем пакета.  |
| GEOCODE_FALLBACK_GEOGRAPHIC  | логическое, определяющее как геокодеру использовать или нет приведение к географическому центру, когда все остальные возможности не выполнены.   |
| GEOCODE_FALLBACK_POSTAL      | логическое, определяющее как геокодеру использовать или нет приведение к центру почтового участка, когда все остальные возможности не выполнены.   |

| Значение атрибута            | Возвращаемое значение GeoCodeInfo( )   |
|------------------------------|--|
| GEOCODE_OFFSET_CENTER        | Вещественное, определяющее расстояние от середины дороги, на котором будут возвращены точки.   |
| GEOCODE_OFFSET_CENTER_UNITS  | строка символов, определяющая единицы измерения середины дорог.  |
| GEOCODE_OFFSET_END           | Вещественное, определяющее расстояние от конца дороги, на котором будут возвращены точки.  |
| GEOCODE_OFFSET_END_UNITS     | строка символов, определяющая единицы измерения отступа от концов дорог  |
| GEOCODE_MIXED_CASE           | логическое, определяющее как MapInfo Professional будет обрабатывать строки возвращенные в смешанном регистре: оставить как есть или переформатировать в верхний регистр. Для отдельных стран нет возможности выполнить эту команду. Этот параметр использует специфичные для каждой страны алгоритмы, в которых определено какая часть адреса должна быть отформатирована в верхнем регистре букв, а какая – в нижнем.  |
| GEOCODE_RESULT_MARK_MULTIPLE | <p>логическое, определяющее должна-ли MapInfo Professional изменить код результата, возвращенный сервером, добавив к коду результата индикатор того, что результат получен произвольным выбором между несколькими близкими совпадениями. Этот флаг может иметь значение только при следующих обстоятельствах:</p> <ol style="list-style-type: none"> <li>1. Геокодирование производится в автоматическом режиме и возможные варианты пользователю не демонстрируются.</li> <li>2. был установлен не интерактивный флаг выбора первой возвращаемой кандидатуры, а не отказа от получения любых многочисленных кандидатур. (смотрите команду Geocode "First"). Этим можно заставить MapInfo Professional принять единственную кандидатуру.</li> <li>3. Фактически на запрос были получены несколько вариантов с похожими на определенную запись значениями.</li> </ol> |

| Значение атрибута                     | Возвращаемое значение <b>GeoCodeInfo( )</b>  |
|---------------------------------------|--|
| <b>GEOCODE_COUNT_GEOCODED</b>         | целое, определяющее число геокодированных за последнюю операцию значений.  |
| <b>GEOCODE_COUNT_NOTGEOCODED</b>      | целое, определяющее число не геокодированных за последнюю операцию значений.   |
| <b>GEOCODE_UNABLE_TO_CONVERT_DATA</b> | логическое, определяющее была или нет обновлена после выполнения геокодирования колонка, может быть из-за несовпадения типов. Типичным примером такого несовпадения, является попытка задать числовую колонку для нечисловых почтовых кодов. |
| <b>GEOCODE_PASSTHROUGH</b>            | целое, определяющее число транзитных элементов, заданное для этого соединения. Для каждой пары существует два элемента. Это значение используется для определения момента прекращения подсчета количества значений без ошибок.               |
| <b>GEOCODE_PASSTHROUGH + <i>n</i></b> | Строка символов, определяющая имя и значение для каждой транзитной пары. <i>n</i> – является значимым вплоть до значения возвращенного вызовом <b>GEOCODE_INFO_PASSTHROUGH</b> .   |
| <b>GEOCODE_MAX_BATCH_SIZE</b>         | целое, определяющее максимальное количество записей (адресов), которое может одновременно передаваться службе для обработки.   |

**Пример:**

Этот фрагмент кода MapBasic печатает в окне сообщений MapInfo Professional сведения от сервиса Envinsa Location Utility Constraints:

```

Include "MapBasic.Def"
declare sub main
sub main
dim iConnect as integer

Open Connection Service Geocode Envinsa
    URL
"http://envinsa_server:8066/LocationUtility/services/LocationUtility"
    User "john"
    Password "green"
    into variable iConnect

Print "Geocode Max Batch Size: " +
GeoCodeInfo(iConnect,GEOCODE_MAX_BATCH_SIZE)
end sub

```

См. также:  
Оператор Open Connection, Оператор Set Connection Geocode

## Оператор Get

### Назначение

Читает данные из файла, открытого в бинарном режиме (BINARY) или в режиме произвольного доступа (RANDOM).

### Синтаксис

**Get** [#] *filenum*, [ *position* ], *var\_name*

*filenum* – номер файла, который открыл Оператор Open File;.

*position* – позиция, с которой начинается чтение из файла;

*var\_name* – имя переменной, в которой MapBasic сохранит результат.

### Описание

Оператор **Get** читает данные из открытого файла. Способ чтения оператором **Get** определяется режимом, который задает Оператор Open File при открытии файла.

Если Оператор Open File задал режим **Random**, то в операторе **Get** параметр Position определяет строку данных, с которой начинается чтение. Сразу после открытия файла позиция для записи – это начало (первая запись) файла. По мере чтения оператором **Get** позиция считывания автоматически наращивается, и нет нужды каждый раз менять значение Position. Однако, если Вы хотите перейти не к следующей по порядку строке, то Вам придется явно задать значение Position.

Если Оператор Open File задал режим **Binary**, одним оператором Get читается одно значение *var\_name*. То, как читаются данные, зависит от порядка задания байтов в исходном файле и типа переменной *var\_name*. Если переменная имеет тип Integer, то прочитываются 4 байта из двоичного файла и превращаются в переменную MapBasic. Переменные заполняются при чтении по следующим правилам:

| Тип переменной                 | Хранение в файле   |
|--------------------------------|--|
| Логическое                     | однобайтовое значение, или 0, или другое ненулевое число                       |
| Целое число типа SmallInt.     | Двухбайтовое значение, целое число.  |
| Целое число типа Integer.      | Четырехбайтовое значение, целое число.   |
| Вещественное число типа Float. | Восьмибайтовое число в формате IEEE.   |
| Строка                         | Длина строки плюс один байт для нулевого значения, обозначающего конец строки. |

| Тип переменной | Хранение в файле  |
|----------------|---|
| Дата типа Date | 4 байта: SmallInt для года, один байт для месяца и один байт для дня. |
| Другие типы    | Чтение не производится.   |

В режиме чтения двоичных кодов (BINARY) параметр *position* используется для задания позиции считывания на определенное значение смещения в файле. Сразу после открытия файла значение *position* устанавливается на единицу (начальный байт файла). По мере выполнения оператора Position, каждый новый оператор **Get** продолжает чтение с того места, на котором оно остановилось в прошлый раз. Если предложение Position опущено, оператор **Get** начинает чтение с того места, где завершилось предыдущее чтение.

**Внимание:** В операторе **Get** нужно поставить две запятые, если параметр *position* опущен.

Если файл был открыт в режиме BINARY, оператор **Get** не может заполнять строку (переменную типа String) неопределенной длины; любая переменная типа String в операторе **Get** должна иметь определенную длину.

**См. также:**

**Оператор Open File, Оператор Put**

---

## Функция GetDate

### Назначение

Возвращает компоненту даты DateTime.

### Синтаксис

```
GetDate (DateTime)
```

### Возвращаемая величина

Дата типа Date

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim dtX as datetime
dim Z as date
dtX = "07.03.07 12:09:09.000 AM"
Z = GetDate(dtX)
Print FormatDate$(Z)
```

## Функция **GetFolderPath( )**

### Назначение

Возвращает путь к специальным папкам MapInfo или Windows.

### Синтаксис

**GetFolderPath**( *folder\_id* )

*folder\_id* – может принимать одно из следующих значений:

```
FOLDER_MI_APPDATA
FOLDER_MI_LOCAL_APPDATA
FOLDER_MI_PREFERENCE
FOLDER_MI_COMMON_APPDATA
FOLDER_APPDATA
FOLDER_LOCAL_APPDATA
FOLDER_COMMON_APPDATA
FOLDER_COMMON_DOCS
FOLDER_MYDOCS
FOLDER_MYPICS
```

### Возвращаемая величина

Строка

### Описание

Получив идентификатор одной из определенных папок MapInfo или Windows, функция **GetFolderPath( )** возвращает путь к этой папке. Примером специальной папки Windows является папка My Documents. Другим примером специальной папки MapInfo является папка с настройками.

Расположение многих из этих папок варьируется в различных версиях Windows. Оно также может отличаться для отдельных пользователей. Обратите внимание, что FOLDER\_MI\_APPDATA, FOLDER\_MI\_LOCAL\_APPDATA и FOLDER\_MI\_COMMON\_APPDATA могут не существовать. Перед попыткой войти в одну из этих папок, убедитесь в их существовании, для чего используется [Функция FileExists\( \)](#). Папка FOLDER\_MI\_PREFERENCE существует всегда.

Идентификаторы папок, начинающиеся с FOLDER\_MI, возвращают путь к папкам, специфическим для MapInfo Professional. Остальные возвращают путь для папок Windows и соответствуют названиям, определенным для функции WIN32 API, а именно, SHGetFolderPath. Самые общие из этих названий можно использовать в приложениях MapBasic. Любые из названий, возможных для SHGetFolderPath, будут работать и с функцией **GetFolderPath( )**.

### Пример:

```
include "mapbasic.def"
declare sub main
sub main
dim sMiPrfFile as string
sMiPrfFile = GetFolderPath$(FOLDER_MI_PREFERENCE)
Print sMiPrfFile
end sub
```

### См. также:

[Функция LocateFile\\$\( \)](#)

---

## Функция GetMetadata\$( )

### Назначение

Извлекает из таблицы метаданные.

### Синтаксис

**GetMetadata\$( table\_name, key\_name )**

*table\_name* – имя открытой таблицы (например, "World") или её синонима (например, World);

*key\_name* – строковая величина, представляющая собой имя ключа метаданных.

### Возвращаемая величина

Величина типа String, строка длиной до 239 байт. Если ключ не существует или соответствующее значение пусто, то MapInfo Professional вернет пустую строку.

### Описание

Эта функция возвращает метаданные из таблицы. Более подробное описание процесса извлечения метаданных из таблицы см. в разделе [Оператор Metadata on page 419](#) или в *Руководстве пользователя MapBasic*.

### Пример:

Если в таблице PARCELS есть ключ метаданных "\Copyright", то следующий оператор распечатывает соответствующее значение метаданных:

```
Print GetMetadata$(Parcels, "\Copyright")
```

### См. также:

[Оператор Metadata](#)



## Функция GetSeamlessSheet( )

### Назначение

Выдает диалоговый запрос, в котором пользователь должен выбрать одну из таблиц-компонент сшитой таблицы и возвращает имя выбранной таблицы.

### Синтаксис

**GetSeamlessSheet**( *table\_name* )

*table\_name* – имя открытой сшитой таблицы.

### Возвращаемая величина

Строка с заданным именем таблицы (пустая строка, если введенные в диалоге данные были отменены).

### Описание

Эта функция показывает диалоговое окно со списком таблиц, составляющих сшитую таблицу. Если пользователь выберет таблицу и нажмет на кнопку **ОК**, то функция вернет имя выбранной таблицы. Если пользователь отменит диалог, то результатом будет пустая строка.

### Пример:

```
Sub Browse_A_Table(ByVal s_tab_name As String)
    Dim s_sheet As String

    If TableInfo(s_tab_name, TAB_INFO_SEAMLESS) Then
        s_sheet = GetSeamlessSheet(s_tab_name)
        If s_sheet <> "" Then
            Browse * From s_sheet
        End If
    Else
        Browse * from s_tab_name
    End If

End Sub
```

### См. также:

**Оператор Set Table, Функция TableInfo( )**

### Функция GetTime()

#### Назначение

Возвращает компоненту времени DateTime.

#### Синтаксис

```
GetTime (DateTime)
```

#### Возвращаемая величина

Время

#### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim dtX as datetime
dim Z as time
dtX = "07.03.07 12:09:09.000 AM"
Z = GetTime(dtX)
Print FormatTime$(Z, "hh:mm:ss.fff tt")
```

---

### Оператор Global

#### Назначение

Объявляет имена и типы глобальных переменных.

#### Синтаксис

```
Global var_name [ , var_name... ] As var_type
    [ , var_name... ] As var_type... ]
```

*var\_name* – имя глобальной переменной;

*var\_type* – тип данных: Integer, Float, Date, Logical, String или тип, созданный при помощи оператора Type.

#### Описание

Оператор **Global** объявляет одну или более глобальных переменных. **Global** употребляется вне текста процедур и функций.

Оператор **Global** и **Оператор Dim** используют одинаковый синтаксис; отличие в том, что переменные, объявленные оператором **Global**, глобальны, а **Оператор Dim** объявляет локальные переменные. Локальные переменные могут использоваться только в процедурах, в которых они объявлены. Значения глобальных переменных могут использоваться и изменяться во всех процедурах программы. В процедурах могут быть объявлены локальные переменные с именами, совпадающими с именами глобальных переменных. В этом случае, в процедуре под этим именем используется только локальная переменная, а значения

глобальной переменной с таким же именем из этой процедуры недоступно. Для изменения размерности массива глобальных переменных используется **Оператор ReDim**. В Windows значения глобальных переменных выполняющейся программы доступны другим программам, поддерживающим DDE-связь.

### Пример:

```
Declare Sub testing( )
Declare Sub Main( )
Global gi_var As Integer
Sub Main( )
    Call testing
    Note Str$(gi_var) ' this displays "23"
End Sub

Sub testing( )
    gi_var = 23
End Sub
```

### См. также:

**Оператор Dim, Оператор ReDim, Оператор Type, Функция UBound( )**

## Оператор Goto

### Назначение

Передаёт управление программой оператору с меткой.

### Предупреждение

Оператор **Goto** не может выполняться в окне MapInfo.

### Синтаксис

**Goto** *label*

*label* – имя метки в процедуре.

### Описание

Оператор **Goto** определяет безусловный переход выполнения программы. Выполнение продолжается с оператора, отмеченного меткой *label*. Метка *label* в тексте программы представляет собой произвольное слово перед оператором, отделенное двоеточием; слово *label* используется в операторе **Goto** без двоеточия.

Оператор **Goto** не рекомендуется использовать для выхода из цикла. Для этого используются **Оператор Exit Do** и **Оператор Exit For**. Также **Goto** нельзя использовать для входа в тело цикла.

Оператор **Goto** можно использовать для перехода на метку *label* только внутри одной процедуры.

### Пример:

```
Goto endproc  
  
...  
  
endproc: End Program
```

### См. также:

[Оператор Do Case...End Case](#), [Оператор Do...Loop](#), [Оператор For...Next](#), [Оператор OnError](#), [Resume statement](#)

---

## Оператор Graph

### Назначение

Открывает новое окно Графика.

### Синтаксис (версии 5.5):

#### Graph

```
label_column, expr [ , ... ]  
From table  
[ Position ( x, y ) [ Units paperunits ] ]  
[ Width window_width [ Units paperunits ] ]  
[ Height window_height [ Units paperunits ] ]  
[ Min | Max ]  
[ Using template_file [ Restore ] [ Series In Columns ] ]
```

*label\_column* – имя колонки, используемой для подписывания оси Y;

*expr* – выражение, вычисляющее значения для помещения на график;

*table* – имя открытой таблицы;

*paperunits* – строковая величина, задающая единицу измерения листа (например, "mm");

*x*, *y* – координаты позиции верхнего левого угла окна Графика в "бумажных" единицах измерения;

*window\_width* и *window\_height* – определяют размер окна Графика в "бумажных" единицах измерения;

*template\_file* – файл шаблона графика.

### Синтаксис (версии до 5.5)

#### Graph

```
label_column, expr [ , ... ]  
From table  
[ Position ( x, y ) [ Units paperunits ] ]  
[ Width window_width [ Units paperunits ] ]  
[ Height window_height [ Units paperunits ] ]  
[ Min | Max ]
```

*label\_column* – имя колонки, используемой для подписывания оси Y;

*expr* – выражение, вычисляющее значения для помещения на график;

*table* – имя открытой таблицы;

*paperunits* – строковая величина, задающая единицу измерения листа (например, "мм");

*x*, *y* – координаты позиции верхнего левого угла окна Графика в "бумажных" единицах измерения;

*window\_width* и *window\_height* – определяют размер окна Графика в "бумажных" единицах измерения;

### Описание

Если предложение **Using** присутствует и *template\_file* указывает файл шаблона, то график версии 5.5 будет создан на основе указанного шаблона. В противном случае будет создан график версии 5.0. Если включено предложение **Restore**, то текст заголовка из файла шаблона будет использован для окна графика. В противном случае для заголовка графика будет использован стандартный текст. Ключевое слово **Restore** будет использовано при записи команды **Graph** в Рабочий набор; при открытии такого Рабочего набора текст заголовка будет восстановлен в точности таким, каким он был при сохранении Рабочего набора. Ключевое слово **Restore** не используется в команде **Graph**, создаваемой с помощью Мастера графиков, и для каждого заголовка будет использован стандартный текст. Если включается **Series In Columns**, то серии графиков будут организовываться из колонок таблицы. В противном случае, серии будут образовываться из строк таблицы.

Команды **Graph** в рабочих наборах или программах, созданных в версиях ранее 5.5, будут генерировать окна графиков версии 5.0. Когда окно графика 5.0 активно в версии MapInfo 5.5, то меню графика версии 5.0 так же будет активным, и пользователь может настраивать график, используя диалоги версии 5.0. Мастер графиков будет всегда создавать окно графиков версии 5.5.

Оператор **Graph** создает новое окно Графика и показывает в нем данные, определенные в таблице *table*. Вид столбчатого графика в окне будет повернут, если Вы заранее не определили другой вид, используя **Оператор Set Graph**.

Команда MapInfo Professional **Окно > График в MapInfo** вызывает диалог, в котором пользователь выбирает имена полей, значения которых будут отображены в графике. Оператор **Graph** в MapBasic, кроме этого, позволяет задавать выражения с именами полей для построения графика. В диалоге команды График пользователь может задать только четыре колонки, тогда как оператор **Graph** может построить график из 255 колонок (если этот график имеет смысл).

Если в операторе **Graph** присутствует ключевое слово **Max**, то окно графика занимает все пространство окна MapInfo Professional. И, наоборот, присутствие в операторе **Graph** слова **Min** приводит к минимизации окна.

### Пример (графики версии 5.5):

```
Graph State_Name, Pop_1980, Pop_1990, Num_Hh_80 From States Using
"C:\Program Files\MapInfo\GRAPHSUPPORT\Templates\Column\Percent.3tf"
```

```
Graph City, Tot_hu, Tot_pop From City_125 Using "C:\Program  
Files\MapInfo\GRAPH SUPPORT\Templates\Bar\Clustered.3tf" Series In Columns
```

### Пример (графики версии до 5.5):

```
Graph страна, население From world
```

**См. также:**

[Оператор Set Graph](#)

## Функция HomeDirectory\$( )

**Назначение**

Возвращает строкой личный каталог пользователя.

**Синтаксис**

HomeDirectory\$( )

**Возвращаемая величина**

Строка

**Описание**

Функция **HomeDirectory\$( )** возвращает строку, представляющую личный каталог пользователя.

Значение, которое будет возвращено функцией, определяется системной платформой. Следующая таблица обобщает правила назначения личных каталогов в разных платформах:

| Система | Результат   |
|---------|---|
| Окна    | Каталог, заданный по правилам Windows и содержащий стартовые файлы Windows (например, WIN.INI). При работе в сети каждый пользователь может иметь свой личный Windows-каталог, что позволяет задавать свою личную конфигурацию. |

**Пример:**

```
Dim s_home_dir As String
s_home_dir = HomeDirectory$( )
```

**См. также:**

**Функция ApplicationDirectory\$( ), Функция ProgramDirectory\$( ), Функция SystemInfo( )**

## Функция HotlinkInfo()

**Назначение**

Возвращает информацию о геолинке на карте.

**Синтаксис**

HotlinkInfo ( *map\_window\_id*, *layer\_number*, *hotlink\_number*, *attribute* )

*map\_window\_id* – идентификатор окна Карты;

## Функция Hour

---

*layer\_number* номер слоя (1 – самый верхний слой); чтобы определить номер слоя в окне Карты, используется **Функция MapperInfo( )**.

*hotlink\_number* – индекс запрашиваемого геолинка. Первое определение геолинка на карте имеет номер индекса 1.

*attribute* - допускаются следующие значения атрибута:

|                       |   |
|-----------------------|---|
| HOTLINK_INFO_EXPR     | Возвращает выражение для файла, на который указывает геолинк.   |
| HOTLINK_INFO_MODE     | Возвращает текущий режим геолинка, одно из следующих predefined значений: <ul style="list-style-type: none"><li>• HOTLINK_MODE_LABEL</li><li>• HOTLINK_MODE_OBJ</li><li>• HOTLINK_MODE_BOTH</li></ul> |
| HOTLINK_INFO_RELATIVE | Возвращает "Да" (TRUE), если маршрут в геолинке задан в относительной форме.  |
| HOTLINK_INFO_ENABLED  | Возвращает "Да" (TRUE), если геолинк активирован.   |

**См. также:**

**Оператор Set Map, Функция LayerInfo( ),**

---

## Функция Hour

### Назначение

Возвращает часовую компоненту времени

### Синтаксис

Hour (Time)

### Возвращаемая величина

Номер

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim Z as time
dim iHour as integer
Z = CurDateTime()
iHour = Hour(Z)
Print iHour
```



## Оператор If...Then

### Назначение

Условное выполнение той или иной группы операторов.

### Синтаксис

```
If if_condition Then
    if_statement_list
    [ ElseIf elseif_condition Then
      elseif_statement_list ]
    [ ElseIf ... ]
    [ Else
      else_statement_list ]
End If
```

*condition* – выражение, результат которого есть логическая величина (TRUE или FALSE);

*statement\_list* – группа операторов, количество которых может быть нулевым.

### Предупреждение

Вы не можете использовать оператор **If...Then** в окне MapBasic.

### Описание

Оператор **If...Then** позволяет выбрать, какую группу операторов выполнить при удовлетворении определенных условий.

Возможна простая форма оператора **If** без предложений **Elseif** и **Else**:

```
If if_condition Then
    if_statement_list
End If
```

В этом случае вычисляется значение выражения *if\_condition*. Если выражение *if\_condition* явно логическому значению TRUE, то MapBasic выполнит операторы *if\_statement\_list*; иначе MapBasic пропустит группу операторов *if\_statement\_list*.

Второй вариант формы **If** включает конструкцию **Else**:

```
If if_condition Then
    if_statement_list
Else
    else_statement_list
End If
```

Здесь MapBasic вычисляет либо группу операторов *if\_statement\_list* (если условие condition TRUE, т.е. истинно), либо группу *else\_statement\_list* (если condition равно FALSE).

Третий вариант формы **If** включает предложение **Elseif**, следующее за **If** (а затем можно добавить предложение **Else clause**):

```
If if_condition Then
    if_statement_list
ElseIf elseif_condition Then
    elseif_statement_list
Else
    else_statement_list
End If
```

В этой форме оператора MapBasic проверяются два или более условий до тех пор, пока либо какое-нибудь условие не равно TRUE, либо пока не достигнут конец предложения **Else**, либо пока не встретится **End If**. Если условие *if\_condition* равно TRUE, MapBasic выполнит группу *if\_statement\_list* и передаст управление первому оператору, следующему за **End If**. Если условие равно FALSE, MapBasic переходит к вычислению условия *elseif\_condition* и, если оно оказывается истинным (TRUE), выполняет группу операторов *elseif\_statement\_list*.

Оператор **If** может содержать несколько конструкций **Elseif**, позволяющих рассматривать любое количество возможных состояний. Но если Вы хотите, чтобы программа была написана в хорошем стиле, используйте **Оператор Do Case...End Case** вместо оператора **If** с несколькими предложениями **Elseif**.

### Пример:

```
Dim today As DateDim today_mon, today_day, yearcount As Integer
today = CurDate( ) ' чтение текущей даты
today_mon = Month(today) ' чтение текущего месяца
today_day = Day(today) ' чтение текущего числа (1-31)
If today_mon = 1 And today_day = 1 Then
    Note "С Новым Годом!"
    yearcount = yearcount + 1
ElseIf today_mon = 2 And today_day = 14 Then
    Note "Вы разослали валентинки?"
ElseIf today_mon = 12 And today_day = 25 Then
    Note "С Рождеством!"
Else
    Note "Добрый день."
End If
```

См. также:

**Оператор Do Case...End Case**

---

## Оператор Import

### Назначение

Создает новую таблицу MapInfo Professional при импорте файла, такого как GML или DXF.

### Синтаксис 1 (для файлов MIF/MID, PICT или MapInfo для DOS)

```
Import file_name
[ Type file_type ]
[ Into table_name ]
[ Overwrite ]
```

**Синтаксис 2 (для файлов DXF)**

```

Import file_name
[ Type "DXF" ]
[ Into table_name ]
[ Overwrite ]
[ Warnings { On | Off } ]
[ Preserve
  [ AttributeData ] [ Preserve ] [ Blocks As MultiPolygonRgns ] ]
[ CoordSys... ]
[ Autoflip ]
[ Transform
  ( DXF_x1, DXF_y1 ) ( DXF_x2, DXF_y2 )
  ( MI_x1, MI_y1 ) ( MI_x2, MI_y2 ) ]
[ Read
  [ Integer As Decimal ] [ Read ] [ Float As Decimal ] ]
[ Store [ Handles ] [ Elevation ] [ VisibleOnly ] ]
[ Layer DXF_layer_name
  [ Into table_name ]
  [ Preserve
    [ AttributeData ] [ Preserve ] [ Blocks As MultiPolygonRgns ] ]
  ]
[ Layer... ]

```

**Синтаксис 3 (для файлов GML)**

```

Import file_name
[ Type "GML" ]
[ Layer layer_name ]
[ Into table_name ]
[ Style Auto [ On | Off ] ]

```

#### Синтаксис 4 (для файлов GML 2.1)

```
Import file_name
[ Type "GML21" ]
[ Layer layer_name ]
[ Into table_name ]
[ Overwrite ]
[ CoordSys... ]
```

*file\_name* – строка, определяющая имя импортируемого файла;

*file\_type* – строка, определяющая формат импортируемого файла (MIF, MBI, MMI, IMG, GML, GML21 или PICT);

*table\_name* – определяет имя новой создаваемой таблицы;

*DXF\_x1*, *DXF\_y1* и т.д. – числа, представляющие координаты в DXF файле;

*MI\_x1*, *MI\_y1* и т.д. – числа, представляющие координаты в таблице MapInfo;

*DXF\_layer\_name* – строка, имя слоя в DXF файле;

*layer\_name* – строка, имя слоя в GML файле.

#### Описание

Оператор **Import** создает новую таблицу MapInfo, импортируя содержимое существующего файла.

**Внимание:** Для создания таблицы MapInfo, оснований на электронной таблице или файле базы данных, используйте **Оператор Register Table**, а не оператор **Import**.

Предложение **Into** позволяет переписать имя и положение создаваемой таблицы MapInfo. Если предложение **Into** не определено, то новая таблица создается в том же каталоге, что и исходный файл, с соответствующим именем. Например, если в среде Windows импортируется текстовый файл "WORLD.MIF", то новая таблица по умолчанию будет иметь имя "WORLD.TAB".

Дополнительное предложение **Type** определяет формат файла при импорте. Предложение **Type** может иметь один из следующих вариантов:

| Значение Type | Формат будущего файла  |
|---------------|--|
| Типе "DXF"    | Формат DXF (графический формат программ CAD, таких как AutoCAD). |
| Типе "MIF"    | Файлы MIF / MID, создаваемые при экспорте таблиц MapInfo.        |
| Типе "MBI"    | Файл MapInfo Boundary Interchange, созданный в MapInfo для DOS.  |
| Типе "MMI"    | Файл MapInfo Map Interchange, созданный в MapInfo для DOS.       |
| Типе "IMG"    | Файл MapInfo Image, созданный в MapInfo для DOS.                 |

| Значение Type | Формат будущего файла  |
|---------------|------------------------|
| Type "GML"    | Файлы формата GML.     |
| Type "GML21"  | Файлы формата GML 2.1. |

Если пропущено предложение **Type**, предполагается, что формат файла определяется расширением. Например, файл с именем "PARCELS.DXF" будет считаться файлом DXF.

Если Вы включаете дополнительное ключевое слово **Overwrite**, MapInfo Professional создаст новую таблицу, независимо от того, существует или нет таблица с таким именем; новая таблица заменит существующую. Если ключевое слово **Overwrite** опущено, а таблица с таким именем уже существует, то MapInfo Professional не будет переписывать таблицу.

### Настройки импорта для файлов DXF

Если Вы импортируете файл DXF, то оператор **Import** может включать следующие предложения, определяющие настройки DXF.

**Внимание:** Порядок предложений крайне важен; помещение предложений в неправильном порядке приведет к ошибкам компиляции.

### Warnings On или Warnings Off

Контролирует, появляются ли предупреждающие сообщения во время импорта. По умолчанию установлено Warnings Off.

### Режим переноса атрибутов (Preserve AttributeData)

Включите это предложение, если надо, чтобы MapInfo Professional сохраняла атрибутивные данные из DXF.

### Режим создания многосвязных областей из блоков (Preserve Blocks As MultiPolygonRgns)

Включите это предложение, если надо, чтобы MapInfo Professional сохраняла все полигоны из блока записи DXF в один объект-регион со множеством полигонов. Если это предложение пропущено, каждый полигон DXF становится отдельным объектом-регионом MapInfo Professional.

### Предложение CoordSys

Контролирует проекцию и систему координат таблицы. Подробнее см.: "**Оператор CoordSys on page 175**".

### Режим Отразить автоматически (Autoflip)

Включите эту команду, если надо повернуть x-координаты карты относительно центральной линии карты. Команда применима только к плановым (не-мировым) координатам.

### Режим Transform

Определяет трансформацию координат. В предложении **Transform** задается минимум и максимум x- и y-координат импортируемого файла, и задается минимум и максимум тех координат, которые Вам понадобятся в таблице MapInfo.

### Режим Read Integer As Decimal

Используйте это предложение, если надо хранить целочисленные значения из файла DXF в десятичной колонке новой таблицы. Это предложение можно использовать только при применении предложения **Preserve AttributeData**.

### Режим Read Float As Decimal

Используйте это предложение, если надо хранить вещественные значения из файла DXF в десятичной колонке новой таблицы. Это предложение можно использовать только при применении предложения **Preserve AttributeData**.

### Режим Store [Handles] [Elevation] [VisibleOnly]

Если определено предложение **Handles**, то таблица MapInfo хранит уникальные ID-номера графических объектов в колонке, с именем **\_DXFHandle**. Если определено предложение **Elevation**, MapInfo Professional хранит высоту центра каждого объекта в колонке с именем **\_DXFElevation**. (Для линий, MI Professional хранит высоты центров линий; для регионов, MI Professional хранит среднее значение высоты объекта.) Если включено предложение **VisibleOnly**, MapInfo Professional игнорирует невидимые объекты.

### Параметр Layer

Если не задано ни одного предложения **Layer**, все объекты из файла DXF импортируются в одну таблицу MapInfo. Если используется одно или более предложений **Layer**, каждый слой DXF, который Вы перечислите, станет отдельной таблицей MapInfo.

Если файл DXF содержит много слоев, и Ваш оператор **Import** содержит одно или несколько предложений **Layer**, то MapInfo Professional будет импортировать слои в соответствии с Вашими именами таблиц. Например, пусть файл DXF содержит четыре слоя (под номерами 0, 1, 2 и 3). Следующий оператор **Import** будет импортировать все четыре слоя в одну таблицу MapInfo:

```
Import "FLOORS.DXF"  
  Into "FLOORS.TAB"  
  Preserve AttributeData
```

Следующий оператор будет импортировать слои 1 и 3, но не будет импортировать слои 0 или 2:

```
Import "FLOORS.DXF"  
  Layer "1"  
    Into "FLOOR_1.TAB"  
    Preserve AttributeData  
  Layer "3"
```

```
Into "FLOOR_3.TAB"
Preserve AttributeData
```

### Импорт GML-файлов

MapInfo Professional поддерживает импорт GML-файлов от OSGB (Ordnance Survey of Great Britain). Поддерживаются объекты типа Cartographic Symbol, Topographic Point, Topographic Line, Topographic Area и Boundary Line; тип Cartographic Text не поддерживается. Topographic Area может иметься в двух формах; MapInfo Professional поддерживает нетопологическую форму. Если файл содержит XLINKS, MapInfo Professional импортирует только атрибутивные данные и не импортирует пространственные объекты. Связи типа XLINK хранятся в файле GML как "xlink:href=". Если топологические объекты включены в файл, то появится предупреждающее сообщение о том, что пространственные объекты не могут быть импортированы. Просмотреть атрибутивные данные можно в окне Списка.

### Импорт GML-файлов

*file\_name* – имя импортируемого файла формата GML 2.1;

**Type** – "GML21" для файлов формата GML 2.1;

*layer\_name* – имя GML-слоя;

*table\_name* – имя таблицы MapInfo;

**Overwrite** – задает режим автоматической перезаписи TAB-файла. Если не задано предложение **Overwrite**, а TAB-файл существует, то будет выдано сообщение об ошибке.

Предложение **Coordsys** не обязательно. Если GML-файл содержит поддерживаемую проекцию, а предложение **Coordsys** не задано, то используется проекция из GML-файла. Если GML-файл содержит поддерживаемую проекцию и задано предложение **Coordsys**, то используется проекция из предложения **Coordsys**. Если GML-файл не содержит поддерживаемую проекцию, нужно явно задать предложение **Coordsys**.

**Внимание:** Если проекция в предложении **Coordsys** не совпадает с проекцией, заданной в GML-файле, импорт данных может быть некорректным. Координатная система должна соответствовать координатной системе данных в файле GML. Эта операция не приведет к преобразованию данных из одной проекции в другую.

### Примеры

Следующий пример импортирует GML-файл:

```
Import "D:\midata\GML\est.gml" Type "GML" layer "LandformArea" style auto
on Into "D:\midata\GML\est_LandformArea.TAB" Overwrite
```

Следующий пример импортирует GML21-файл:

```
Import "D:\midata\GML\GML2.1\mi_usa.xml" Type "GML21" layer "USA" Into
"D:\midata\GML\GML2.1\mi_usa_USA.TAB" Overwrite CoordSys Earth Projection
1, 104
```

Пример импортирования с применением текущего стиля MapInfo:

```
Import "D:\midata\GML\test.gml" Type "GML" layer "TopographicLine" style
auto off Into "D:\midata\GML\test_TopographicLine.TAB" Overwrite
```

Пример импорта в файл MIF (MapInfo Interchange Format):

```
Import "WORLD.MIF"  
  Type "MIF"  
  Into "world_2.tab"  
  
Map From world_2
```

**См. также:**

[Оператор Export](#)

---

## Оператор Include

### Назначение

Объединяет содержимое отдельного текстового файла с текстом программы MapBasic.

### Синтаксис

```
Include "filename"
```

*filename* – имя текстового файла.

### Предупреждение

Вы не можете использовать оператор **Include** в окне MapBasic.

### Описание

Когда MapBasic компилирует программный файл с оператором **Include**, текст определенного в этом операторе файла рассматривается как часть текста программы. Файл, заданный оператором **Include**, должен состоять из допустимых операторов MapBasic.

Если имя файла в параметре *filename* задано без каталога и этот файл не находится в активном каталоге, то компилятор языка MapBasic делает попытку найти его в том каталоге, в котором расположен программный файл MapBasic. Это позволяет Вам не копировать каждый раз файлы стандартных определений (такие, как MAPBASIC.DEF) в каталог разрабатываемой программы.

Оператор **Include** используется для подсоединения к файлам стандартных определений, таких, как MAPBASIC.DEF и MENU.DEF. В них с помощью оператора Define целочисленным кодам заданы имена (например, TRUE и FALSE), использование которых в Вашей программе делает ее текст более понятным.

Однако, если Вы изменили содержимое текстового файла, подсоединенного оператором **Include**, то для внесения этих изменений в программу ее необходимо перекомпилировать.

### Пример:

```
Include "MAPBASIC.DEF"
```



## Оператор Input #

### Назначение

Читает данные из файла и помещает их в переменные.

### Синтаксис

```
Input # filenum, var_name [ , var_name... ]
```

*filenum* – номер открытого файла, который возвращает **Оператор Open File**.

*var\_name* – имя переменной.

### Описание

Оператор **Input #** читает данные из файла, открытого в режиме последовательного доступа (INPUT, OUTPUT или APPEND), и присваивает их как значения одной или более переменным MapBasic.

Оператор **Input #** читает данные последовательно от верхней строки к последней, присваивая каждую порцию одной переменной *var\_name*. Оператор считывает порции данных из файла, используя в качестве разделителей запятую и символ конца строки. Поэтому, чтобы прочитать одну строку с запятыми в одну строковую переменную, надо использовать **Оператор Line Input**.

MapBasic автоматически преобразовывает считанные данные в соответствующие типы. Когда оператор **Input #** читает пустую строку, то значением строковой переменной будет пустая строка (""). Если переменная была объявлена численным типом, то **Input #** преобразует пустую строку в ноль.

После выполнения оператора **Input #** применяется **Функция EOF( )**, чтобы определить, корректно ли MapInfo Professional прочитала данные. Если чтение произошло успешно, **Функция EOF( )** возвращает логическое "нет" (FALSE); если же был прочитан признак конца файла, **Функция EOF( )** вернет логическое "да" (TRUE).

Пример использования оператора **Input #** Вы можете увидеть в программе NIEWS ("Именованные Виды") из стандартной поставки MapBasic.

Следующие типы данных нельзя считывать оператором **Input #**:

- Псевдоним типа Alias
- Pen
- Brush
- Шрифт
- Символ
- Объект

### См. также:

**Функция EOF( )**, **Оператор Line Input**, **Оператор Open File**, **Оператор Write #**

## Оператор Insert

### Назначение

Добавляет новую строку в открытую таблицу

### Синтаксис

```
Insert Into table  
[ ( columnlist ) ]  
{ Values ( exprlist ) | Select columnlist From table }
```

*table* – имя открытой таблицы;

*columnlist* – разделенной запятыми список колонок;

*exprlist* – одно или несколько выражений для колонок, разделенных запятыми.

### Описание

Оператор **Insert** вставляет новую строку в открытую таблицу. Этот оператор применяется в двух формах, позволяющих либо добавлять таблице по одной строке, либо добавлять группы строк из другой таблицы (с помощью предложения **Select**). В каждом случае порядок и количество выражений, которые будут вставлены в поля новой строки, должны соответствовать порядку колонок. Если колонки не перечислены, то будут рассматриваться все поля в новой строке. Если Вы хотите сохранить таблицу с новыми строками на диске, то после ввода используйте оператор **Оператор Commit Table**.

Если Вы точно знаете, сколько колонок в таблице, и если Вам уже известно, какие именно величины в какие поля надо поместить, то можно опустить предложение *columnlist*.

В следующем примере результатом оператора будет новая строка в таблице, состоящей из четырех колонок ("Имя", "Адрес", "Город", "Область"), и каждому полю будет определено значение:

```
Insert Into customers Values ("Мария Павловна Носова", "ул. Солнечная, 2,  
23", "Троицк", "Московская")
```

Результатом этого оператора может быть ошибка, если таблица будет иметь больше или меньше колонок, чем четыре. Если Вы не знаете точно, сколько колонок составляют таблицу, или не знаете, в каком порядке расположены колонки в таблице, необходимо в операторе использовать список имен колонок *columnlist*.

### Примеры

В следующем примере в таблицу клиентов вставляется новая строка с одним значением поля, а остальные колонки остаются пустыми. Заполняется только поле имени с помощью предложения **Values**, при этом можно не заботиться о том, какова структура таблицы и какое место в ней занимает колонка "Имя".

```
Insert Into customers (Имя)Values ("Степан Иванович Гарин")
```

Следующий оператор создает точечный объект и вставляет его в поле "Obj" новой записи в таблице SITES. "Obj" - это имя специальной колонки для хранения графических объектов. Таким образом, мы создаем новую чистую запись с присоединенным точечным объектом.

```
Insert Into sites (Obj)
  Values ( CreatePoint(-73.5, 42.8) )
```

Следующий пример иллюстрирует, как оператором **Insert** можно добавить запись в таблицу, используя запись из другой. Допустим, что в таблице NY\_ZIPS содержатся ZIP-коды штата Нью-Йорк и в таблице NJ\_ZIPS содержатся ZIP-коды штата Нью-Джерси. Добавим все ZIP-коды из одной таблицы в другую. (Такая операция иногда необходима для поиска, так как оператор Find может работать только с одной таблицей.)

Аналогично, оператор **Insert** добавляет записи из таблицы штата New Jersey в таблицу штата New York.

```
Insert Into NY_ZIPS
  Select * From NJ_ZIPS
```

## Функция InStr ( )

---

В следующем примере берутся все объекты из таблицы WORLD и каждый подсоединяется к новой строке таблицы OUTLINE.

```
Open Table "world"
Open Table "outline"
Insert Into outline (Obj)
    Select Obj From World
```

**См. также:**

[Оператор Commit Table](#), [Оператор Delete](#), [Оператор Rollback](#)

---

## Функция InStr( )

### Назначение

Возвращает позицию первого символа подстроки в строке.

### Синтаксис

**InStr**( *position*, *string*, *substring* )

*position* – стартовая позиция для поиска, положительное целое число;

*string* – строковое выражение;

*substring* – строковое выражение, искомая подстрока.

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция **InStr**( ) проверяет, входит ли в состав строки *string* подстрока *substring*. MapBasic просматривает исходную строку начиная с символа с номером *position*; если *position* равно единице, то MapBasic начинает поиск с начала строки *string*.

Если *string* не содержит *substring*, функция **InStr**( ) возвращает ноль.

Если строка *string* содержит подстроку *substring*, то функция **InStr**( ) возвращает позицию начала подстроки в строке. Например, если *substring* находится в самом начале строки *string*, функция **InStr**( ) возвращает единицу.

Если строка *substring* – пустая строка, то результатом функции **InStr**( ) будет 0 (ноль).

Функция **InStr**( ) различает строчные и прописные буквы. Другими словами, функция **InStr**( ) не найдет подстроку "ИЯ" в строке "Россия" и вернет ноль.

### Ошибки:

ERR\_FCN\_ARG\_RANGE, если значение аргумента выходит за пределы, заданные при его определении.

**Пример:**

```
Dim fullname As String, pos As Integer
fullname = "Галина Петровна Соколова"
pos = InStr(1, fullname, "Галина") ' переменная pos равна 1 (единице)
pos = InStr(1, fullname, "СОКОЛ") ' теперь pos равна 0; ' так как СОКОЛ записано большими буквами, ' в строке "Галина Петровна Соколова" InStr найти ее не может
```

**См. также:**

**Функция Mid\$( )**

---

## Функция Int( )

**Назначение**

Возвращает целое число, полученное из целой части действительного числа.

**Синтаксис**

**Int**( *num\_expr* )

*num\_expr* – числовое выражение

**Возвращаемая величина**

Целое число типа Integer.

**Описание**

Функция **Int( )** отсекает дробную часть от действительного числа, полученного в результате вычисления выражения *num\_expr*, и возвращает целую часть. **Оператор Fix( )** и **Int( )** похожи, но не идентичны. Функции различаются способом удаления дробной части отрицательного числа. Для отрицательного дробного аргумента **Оператор Fix( )** возвратит ближайшее целое число, большее или равное исходному, например **Fix(-2.3)** возвратит -2. Функция же **Int( )** возвращает ближайшее целое, меньше или равное оригиналу. Так, функция **Int(-2.3)** возвратит -3.

**Пример:**

```
Dim i_whole As Integer
i_whole = Int(5.999) ' whole сейчас имеет значение 5
i_whole = Int(-7.2) ' i_whole сейчас имеет значение -8.
```

**См. также:**

**Оператор Fix( ), Функция Round( )**

### Функция `IntersectNodes( )`

#### Назначение

Вычисляет точки пересекающихся объектов и возвращает полилинию, имеющую узлы в точках пересечения.

#### Синтаксис

`IntersectNodes( object1, object2, unit_name )`, где

*object1* и *object2* – объектные выражения, которые могут представлять объекты любого типа, кроме точечного и текстового;

*points\_to\_include* – один из следующих кодов:

- `INCL_CROSSINGS` – функция возвращает узлы, в которых сегменты обоих объектов пересекаются;
- `INCL_COMMON` – функция возвращает узлы отрезков, на которые накладываются сегменты обоих объектов;
- `INCL_ALL` – функция возвращает узлы, в которых сегменты пересекаются и накладываются.

#### Возвращаемая величина

Полилиния, составленная из точек пересечения. Величина типа `Object`.

#### Описание

Функция `IntersectNodes( )` возвращает объект типа "полилиния", узлы которого лежат в точках пересечения объектов.

---

### Функция `IsogramInfo( )`

#### Назначение

Возвращает любой или все атрибуты соединения, для создания которого использовался **Оператор Set Connection Isogram**. Включая атрибуты, задающие максимальное количество записей на сервере, значение времени и расстояния.

#### Синтаксис

`IsogramInfo( connection_handle, attribute )`

*connection\_handle* – целое, определяющее число соединений, которое возвращает **Оператор Open Connection**.

*attribute* это целое, определяющее, какого типа информация будет возвращена.

**Возвращаемая величина**

Вещественное (Float), логическое (Logical), или строка символов (String), в зависимости от параметра *атрибута*.

**Описание**

Эта функция возвращает стандартные параметры соединения или, если применялась **Оператор Set Connection Isogram**, заданные ею параметры.

Существует несколько атрибутов, которые **IsogramInfo( )** может вернуть. Коды определены в MAPBASIC.DEF.

| Параметры                                   | Возвращаемые значения IsogramInfo( )  |
|---|---|
| ISOGRAM_BANDING                             | логическое, определяющее оконтуривание  |
| ISOGRAM_MAJOR_ROADS_ONLY                    | логическое, определяющее выбор только основных дорог MajorRoadsOnly                                       |
| ISOGRAM_RETURN_HOLES                        | логическое, оопределяющее следует или нет принимать возвращаемые полигоны с пустотами.                    |
| ISOGRAM_MAJOR_POLYGON_ONLY                  | логическое, определяющее выбор только основного полигона возвращаемой области.                            |
| ISOGRAM_MAX_OFF_ROAD_DISTANCE               | Вещественное, определяющее значение максимального расстояния Maximum off Road Distance.                   |
| ISOGRAM_MAX_OFF_ROAD_DISTANCE_UNITS         | Строка символов, определяющая единицу измерения расстояний  |
| ISOGRAM_SIMPLIFICATION_FACTOR               | Вещественное, определяющее параметр упрощения (Simplification Factor). (относительное значение от 0 до 1) |
| ISOGRAM_DEFAULT_AMBIENT_SPEED               | вещественное, определяющее скорость потока.   |
| ISOGRAM_DEFAULT_AMBIENT_SPEED_DISTANCE_UNIT | строка символов, определяющая единицу измерения расстояний ("mi", "km").                                  |
| ISOGRAM_DEFAULT_AMBIENT_SPEED_TIME_UNIT     | Строка символов, определяющая единицу измерения времени ("hr", "min", "sec").                             |

| Параметры                          | Возвращаемые значения <b>IsogramInfo( )</b>  |
|------------------------------------|--|
| ISOGRAM_DEFAULT_PROPAGATION_FACTOR | Определяет процентное отношение проселочных дорог в оценке остатка (расстояний) допустимое при вычислении полос дальностей. Дороги не включенные в сеть дорог могут быть в том числе и проездами или подъездными путями. Параметр движения – процентное отношение оценки при вычислении дальности от начальной точки до заданной на заданном расстоянии. По умолчанию этот параметр имеет значение 0.16. |
| ISOGRAM_BATCH_SIZE                 | целое, определяющее максимальное количество записей, которое может одновременно передаваться службе для обработки.   |
| ISOGRAM_POINTS_ONLY                | логическое, определяющее, как обрабатывать не точечные объекты, пропускать или нет.  |
| ISOGRAM_RECORDS_INSERTED           | целое, в котором отражено количество записей вставленное последней командой.   |
| ISOGRAM_RECORDS_NOTINSERTED        | целое, в котором отражено количество записей не вставленное последней командой.  |
| ISOGRAM_MAX_BATCH_SIZE             | целое, определяющее максимальное количество записей (например, точек), которое можно одновременно передавать службе для обработки.   |
| ISOGRAM_MAX_BANDS                  | целое число – максимальное число зон времени или расстояний.   |
| ISOGRAM_MAX_DISTANCE               | вещественное, определяющее максимальную удаленность зоны запроса расстояний. расстояние измеряется в единицах, указанных параметром ISOGRAM_MAX_DISTANCE_UNITS.  |
| ISOGRAM_MAX_DISTANCE_UNITS         | строка задающая единицы измерения, используемые в ISOGRAM_MAX_DISTANCE.  |



| Параметры              | Возвращаемые значения IsogramInfo( )  |
|------------------------|---|
| ISOGRAM_MAX_TIME       | вещественное, определяющее максимальную удаленность по времени зоны запроса расстояний. Время измеряется в единицах, указанных параметром ISOGRAM_MAX_TIME_UNITS. |
| ISOGRAM_MAX_TIME_UNITS | строка, задающая единицы времени, используемые в ISOGRAM_MAX_TIME.  |

### Пример:

Следующий фрагмент кода MapBasic печатает в окне сообщений MapInfo Professional сведения от Envinsa Routing Constraints:

```
Include "MapBasic.Def"
declare sub main
sub main
dim iConnect as integer
Open Connection Service Isogram
    URL "http://envinsa_server:8062/Route/services/Route"
    User "john"
    Password "green"
    into variable iConnect

Print "Isogram_Max_Batch_Size: " +
IsogramInfo(iConnect,Isogram_Max_Batch_Size)
Print "Isogram_Max_Bands: " + IsogramInfo(iConnect, Isogram_Max_Bands)
Print "Isogram_Max_Distance: " + IsogramInfo(iConnect,
Isogram_Max_Distance)
Print "Isogram_Max_Distance_Units: " + IsogramInfo(iConnect,
Isogram_Max_Distance_Units)
Print "Isogram_Max_Time: " + IsogramInfo(iConnect,Isogram_Max_Time)
Print "Isogram_Max_Time_Units: " +
IsogramInfo(iConnect,Isogram_Max_Time_Units)
Close Connection iConnect
end sub
```

### См. также:

[Оператор Create Object](#), [Оператор Open Connection](#), [Оператор Set Connection Isogram](#)

---

## Функция IsPenWidthPixels( )

### Назначение

Определяет в каких величинах измеряется ширина линии - в пикселах или в пунктах.

### Синтаксис

**IsPenWidthPixels**( *penwidth* )

*penwidth* – это короткое целое, определяющее ширину линии.

### Возвращаемая величина

TRUE, если ширина линии задана в пикселах. FALSE, если ширина задана в пунктах.

### Описание

Функция **IsPenWidthPixels( )** возвратит истинное значение (TRUE), если ширина линии задана в пикселах. Ширину линии может определить, используя [Функция StyleAttr\( \)](#).

**Пример:**

```

Include "MAPBASIC.DEF"
Dim CurPen As Pen
Dim Width As Integer
Dim PointSize As Float
CurPen = CurrentPen( )
Width = StyleAttr(CurPen, PEN_WIDTH)
If Not IsPenWidthPixels(Width) Then
    PointSize = PenWidthToPoints(Width)
End If

```

**См. также:**

[Функция CurrentPen\( \)](#), [Функция MakePen\( \)](#), [Предложение Pen](#), [Функция PenWidthToPoints\( \)](#), [Функция StyleAttr\( \)](#)

---

## Оператор Kill

**Назначение**

Удаляет файл.

**Синтаксис**

```
kill filespec
```

*filespec* – строка, представляющая имя файла и, если необходимо, DOS-маршрут.

**Возвращаемая величина**

Строка

**Описание**

Оператор **Kill** удаляет файл с диска. Действие оператора **Kill** нельзя отменить. Поэтому применяйте **Kill** с осторожностью.

**Пример:**

```
kill "C:\TEMP\JUNK.TXT"
```

**См. также:**

[Оператор Open File](#)

## Функция LabelFindByID( )

### Назначение

Инициализирует внутренний указатель подписи; после этого Вы можете запросить подпись из определенной строки в слое Карты.

### Синтаксис

**LabelFindByID**( *map\_window\_id*, *layer\_number*, *row\_id*, *table*, *b\_mapper* )

*map\_window\_id* – целочисленный индекс, определяющий окно Карты;

*layer\_number* – номер слоя в текущем окне Карты (например, 1 для верхнего слоя);

*row\_id* – положительное целочисленное значение, указывающее номер строки, в которой запрашивается подпись;

*table* – имя таблицы или пустой строки (""): когда Вы делаете запрос к таблице, входящей в сшитую таблицу, укажите имя такой таблицы, входящей в сшитую; в противном случае, укажите пустую строковую переменную;

*b\_mapper* – логическая величина, "Да" (TRUE) при запросе подписей, которые появляются при активном окне Карты; "Нет" (FALSE) при запросе подписей, которые появляются, когда Карта помещена в Отчет.

### Возвращаемая величина

Логическая величина: "Да" (TRUE) означает, что подпись для определенной строки существует.

### Описание

Вызывайте **LabelFindByID**( ), когда Вы хотите запросить подпись из определенной строки в слое Карты. Если величина, полученная в результате – "Да" (TRUE), то подпись существует в строке, и ее может запросить **Функция Labelinfo**( ).

**Пример:**

Следующий пример изображает карту мира, автоматически ее подписывает и затем определяет, была ли изображена подпись из определенной строки таблицы.

```
Include "mapbasic.def"
Dim b_morelabels As Logical
Dim i_mapid As Integer
Dim obj_mytext As Object
Open Table "World" Interactive As World
Map From World
i_mapid = FrontWindow( )
Set Map Window i_mapid Layer 1 Label Auto On
' Убедитесь, что все подписи изображены, перед тем как мы продолжим...
Update Window i_mapid
' Теперь посмотрим, подписана ли автоматически строка # 1b_morelabels =
LabelFindByID(i_mapid, 1, 1, "", TRUE)
If b_morelabels Then ' Объект подписан, теперь запросим его подпись.
    obj_mytext = LabelInfo(i_mapid, 1, LABEL_INFO_OBJECT)' В этом месте Вы
    можете сохранить объект obj_mytext ' в постоянной таблице; или можете
    запросить его ' с помощью функций ObjectInfo( ) или ObjectGeography( ).
End If
```

**См. также:**

**Функция LabelFindFirst( ), Функция LabelFindNext( ), Функция LabelInfo( )**

---

## Функция LabelFindFirst( )

**Назначение**

Инициализирует внутренний указатель подписи, так что Вы можете запросить первую подпись на слое Карты.

**Синтаксис**

**LabelFindFirst**( *map\_window\_id*, *layer\_number*, *b\_mapper* )

*map\_window\_id* – целочисленный индекс, определяющий окно Карты;

*layer\_number* – номер слоя в текущем окне Карты (например, 1 для верхнего слоя);

*b\_mapper* – логическая величина, “Да” (TRUE) при запросе подписей, которые появляются при активном окне Карты; “Нет” (FALSE) при запросе подписей, которые появляются, когда Карта помещена в Отчет.

**Возвращаемая величина**

Логическая величина: “Да” (TRUE) означает, что подпись существует в определенном слое (либо подписи видимы в данный момент, либо пользователь их редактировал, либо эти отредактированные подписи в данный момент невидимы).

### Описание

Вызовите функцию **LabelFindFirst( )**, когда Вам понадобится запросить подписи на слое Карты в цикле. Запрос подписей является процессом из двух шагов:

1. Установите внутренний указатель подписи MapBasic. Для этого используется **LabelFindFirst( )**, **Функция LabelFindNext( )** или **Функция LabelFindByID( )**.
2. Если функция, которую Вы вызвали на шаге 1, не возвращает “Нет” (FALSE), Вы можете запросить текущую подпись, используя **Функция Labelinfo( )**.

Для продолжения запроса дополнительных подписей, вернитесь к шагу 1.

### Пример:

См. пример в разделе **Функция Labelinfo( )** на стр. 43.

### См. также:

**Функция LabelFindByID( )**, **Функция LabelFindNext( )**, **Функция Labelinfo( )**

---

## Функция LabelFindNext( )

### Назначение

Перемещает внутренний указатель подписи, так что Вы можете запрашивать следующую подпись на слое Карты.

### Синтаксис

**LabelFindNext( map\_window\_id, layer\_number )**

*map\_window\_id* – целочисленный индекс, определяющий окно Карты;

*layer\_number* – номер слоя в текущем окне Карты (например, 1 для верхнего слоя);

### Возвращаемая величина

Логическая величина: “Да” (TRUE) означает, что указатель подписи передвинут к следующей подписи; “Нет” (FALSE) означает, что на этом слое более нет подписей.

### Описание

После того, как Вы вызовете **Функция LabelFindFirst( )**, чтобы начать запрос подписей, можно вызвать функцию **LabelFindNext( )**, чтобы переместиться к следующей подписи на этом же слое.

### Пример:

См. пример в разделе **Функция Labelinfo( )** на стр. 43.

### См. также:

**Функция LabelFindByID( )**, **Функция LabelFindFirst( )**, **Функция Labelinfo( )**

## Функция Labelinfo( )

### Назначение

Возвращает информацию о подписи на Карте. LabelInfo возвращает подписи, как текстовые объекты., Эти тестовые объекты могут быть либо размещены вдоль кривой, либо повернуты под углом. Однако, если подписи выполнены вдоль кривой, то они будут возвращены как повернутый текст вдоль прямой.

### Синтаксис

**Labelinfo**( *map\_window\_id*, *layer\_number*, *attribute* )

*map\_window\_id* – целочисленный индекс, определяющий окно Карты;

*layer\_number* – номер слоя в текущем окне Карты (например, 1 для верхнего слоя);

*attribute* – целочисленный код, см. таблицу ниже.

### Возвращаемая величина

Тип результата зависит от значения параметра attribute.

### Описание

Функция **Labelinfo**( ) возвращает информацию о подписи в окне Карты.

**Внимание:** Подписи и текстовые объекты - это не одно и то же. Для запроса текстовых объектов используются **Функция ObjectInfo( )** или **Функция ObjectGeography( )**.

Перед вызовом **Labelinfo**( ), Вы должны инициализировать внутренний указатель подписей MapBasic. Для этого используются **Функция LabelFindFirst( )**, **Функция LabelFindNext( )** или **Функция LabelFindByID( )**. Смотрите пример ниже.

Параметр attribute должен принимать одно из следующих значений в таблице; коды определены в MAPBASIC.DEF.

| Значения attribute | Функция Labelinfo( ) возвращает  |
|--------------------|--|
| LABEL_INFO_ANCHORX | Величина типа Float, указывает X-координату места привязки подписи.    |
| LABEL_INFO_ANCHORY | Величина типа Float, указывает Y-координату места привязки подписи.    |
| LABEL_INFO_DRAWN   | Логическая величина; “Да” (TRUE) если подпись в текущий момент видима. |
| LABEL_INFO_EDIT    | Логическая величина; “Да” (TRUE) если подпись редактировалась.         |

| Значения attribute         | Функция Labelinfo( ) возвращает   |
|----------------------------|---|
| LABEL_INFO_EDIT_ANCHOR     | Логическая величина; “Да” (TRUE) если подпись переместилась.  |
| LABEL_INFO_EDIT_ANGLE      | Логическая величина; “Да” (TRUE) если угол наклона подписи изменился.   |
| LABEL_INFO_EDIT_FONT       | Логическая величина; “Да” (TRUE) если шрифт подписи изменился.  |
| LABEL_INFO_EDIT_OFFSET     | Логическая величина; “Да” (TRUE) если расстояние в точках от подписи до места прикрепления изменилось.  |
| LABEL_INFO_EDIT_PEN        | Логическая величина; “Да” (TRUE) если стиль линии указки изменился.   |
| LABEL_INFO_EDIT_POSITION   | Логическая величина; “Да” (TRUE) если позиция подписи (относительно места прикрепления) изменилась.   |
| LABEL_INFO_EDIT_TEXT       | Логическая величина; “Да” (TRUE) если текст подписи изменился.  |
| LABEL_INFO_EDIT_TEXTARROW  | Логическая величина; “Да” (TRUE) если стрелка указки подписи изменилась.  |
| LABEL_INFO_EDIT_TEXTLINE   | Логическая величина; “Да” TRUE если указка переместилась.   |
| LABEL_INFO_EDIT_VISIBILITY | Логическая величина; “Да” (TRUE) если выбран режим видимости подписи “Скрыть” (OFF).  |
| LABEL_INFO_OBJECT          | <p>Возвращает текстовый объект, который практически является подписью. Эта операция позволяет конвертировать подпись в текстовый объект, который Вы можете сохранить в постоянной таблице.</p> <p><b>Внимание:</b> LABEL_INFO_OBJECT возвращает текстовый объект, но если подпись выполнена вдоль кривой, то она будет возвращена в неискривленном виде. MapBasic не обеспечивает работу с подписями вдоль кривой как с текстовыми объектами.</p> |
| LABEL_INFO_OFFSET          | Целочисленное значение от 0 до 200, показывающее расстояние (в точках) от подписи до места прикрепления.  |



| Значения attribute     | Функция Labelinfo( ) возвращает   |
|------------------------|---|
| LABEL_INFO_ORIENTATION | <p>Возвращает короткое целое, описывающее ориентацию подписи. Подпись создается одной из следующих функций: LabelFindFirst, LabelFindByID или LabelFindNext. Результатом может быть один из следующих кодов:</p> <ul style="list-style-type: none"> <li>LAYER_INFO_LABEL_ORIENT_HORIZONTAL (подпись под углом равным 0)</li> <li>LAYER_INFO_LABEL_ORIENT_PARALLEL (подпись под любым углом отличным от 0)</li> <li>LAYER_INFO_LABEL_ORIENT_CURVED (подпись по кривой)</li> </ul>  |
| LABEL_INFO_POSITION    | <p>Целочисленное значение от 0 до 8, показывающее положение подписи относительно места прикрепления. Величина, полученная в результате соответствует одному из следующих кодов:</p> <ul style="list-style-type: none"> <li>LAYER_INFO_LBL_POS_CC (0),</li> <li>LAYER_INFO_LBL_POS_TL (1),</li> <li>LAYER_INFO_LBL_POS_TC (2),</li> <li>LAYER_INFO_LBL_POS_TR (3),</li> <li>LAYER_INFO_LBL_POS_CL (4),</li> <li>LAYER_INFO_LBL_POS_CR (5),</li> <li>LAYER_INFO_LBL_POS_BL (6),</li> <li>LAYER_INFO_LBL_POS_BC (7),</li> <li>LAYER_INFO_LBL_POS_BR (8).</li> </ul> <p>Например, если подпись ниже и правее места прикрепления, ее позиция обозначается кодом 8; если подпись располагается выше и левее места прикрепления, ее код будет 1.</p> |
| LABEL_INFO_ROWID       | Целочисленное значение, представляющее индекс ID строки, в которой расположена подпись; возвращает 0, если подписи не существует.   |
| LABEL_INFO_SELECT      | Логическая величина; "Да"(TRUE) если подпись выбрана.   |
| LABEL_INFO_TABLE       | Строковая величина, представляющая имя таблицы, в которой находится данная подпись. Используется при работе с сшитой таблицей, в том случае, если Вам надо узнать в какой из составляющих ее таблиц находится данная подпись.   |

### Пример:

Следующий пример показывает, как в цикле идет поиск подписи среди других подписей в определенной строке, используя функцию **LabelInfo( )**, запрашивающую каждую подпись.

```
Dim b_morelabels As Logical
Dim i_mapid, i_layernum As Integer
Dim obj_mytext As Object
' В этом месте Вы присваиваете окну Карты ID индекс i_mapid, ' и
присваиваете слою номер i_layernum.
b_morelabels = LabelFindFirst(i_mapid, i_layernum, TRUE) Do While
b_morelabels obj_mytext = LabelInfo(i_mapid, i_layernum,
LABEL_INFO_OBJECT) ' В этом месте Вы можете сохранить объект obj_mytext '
в постоянной таблице; или можете запросить его, ' функцией ObjectInfo( )
или ObjectGeography( ).
    b_morelabels = LabelFindNext(i_mapid, i_layernum)
Loop
```

### См. также:

**Функция LabelFindByID( ), Функция LabelFindFirst( ), Функция LabelFindNext( )**

---

## Функция LayerInfo( )

### Назначение

Возвращает информацию о слое в окне Карты.

### Синтаксис

**LayerInfo( map\_window\_id, layer\_number, attribute )**

*map\_window\_id* – идентификатор окна Карты;

*layer\_number* номер слоя (1 – самый верхний слой); чтобы определить номер слоя в окне Карты, используется **Функция MapperInfo( )**.

*attribute* – целочисленный код, см. таблицу ниже.

Новый атрибут LAYER\_INFO\_HOTLINK\_COUNT позволяет задавать несколько геолинков на слое карты.

Для обеспечения обратной совместимости, поддерживается старый набор атрибутов, но при этом будет возвращаться определение первого геолинка на слое. Если геолинки не заданы, то эта функция будет возвращать следующие значения:

- LAYER\_INFO\_HOTLINK\_EXPR – пустая строка ("" )
- LAYER\_INFO\_HOTLINK\_MODE – возвращает принятое по умолчанию значение HOTLINK\_MODE\_LABEL
- LAYER\_INFO\_HOTLINK\_RELATIVE – возвращает значение по умолчанию FALSE

LAYER\_INFO\_LABEL\_ORIENTATION – возвращает короткое целое, описывающее ориентацию автоматически создаваемых подписей.. Результатом будет один из следующих кодов:

- LAYER\_INFO\_LABEL\_ORIENT\_HORIZONTAL (подписи под углом равным 0)
- LAYER\_INFO\_LABEL\_ORIENT\_PARALLEL (подписи под любым углом отличным от 0)
- LAYER\_INFO\_LABEL\_ORIENT\_CURVED (подписи по кривой)

**Внимание:** Если возвращается LAYER\_INFO\_LABEL\_ORIENT\_PARALLEL, то LAYER\_INFO\_LABEL\_PARALLEL возвращает TRUE.

Возвращаемая величина

Тип величины зависит от значения attribute.

Предупреждение

Многие коды, которые использует **Функция LayerInfo( )**, не применимы к Косметическому слою, тематическому слою и слою с растровым изображением. Пример смотрите ниже.

Описание

Функция **Функция LayerInfo( )** возвращает информацию об определенном слое в окне Карты. Параметр *layer\_number* задает слой на карте (0 – Косметический слой, 1 – самый верхний слой, и т. д.). Параметр *attribute* должен принимать одно из следующих значений в таблице; коды определены в MAPBASIC.DEF. Здесь также приведены коды LAYER\_HOTLINK\_\* для получения информации об атрибутах геолинка.

| Значения attribute         | Результат LayerInfo( )  |
|----------------------------|---|
| LAYER_INFO_NAME            | Строка (тип String), содержащая имя таблицы, данные которой представлены на этом слое. Если слой Косметический, то строка будет именоваться "Cosmetic1"; это имя может быть использовано в других операторах (например, <b>Оператор Select</b> ). |
| LAYER_INFO_EDITABLE        | Логическая величина (тип Logical), показывающая, изменяем слой или нет.   |
| LAYER_INFO_LBL_PARTIALSEGS | Логическая величина; TRUE, если для этого слоя установлен флажок <b>Label Partial Objects</b> .   |
| LAYER_INFO_SELECTABLE      | Логическая величина (тип Logical), показывающая, доступен ли слой или нет.  |
| LAYER_INFO_PATH            | Строка (тип String), содержащая DOS-маршрут, по которому находится файл таблицы, данные из которой представлены на этом слое.   |

| Значения attribute             | Результат LayerInfo( )  |
|--------------------------------|---|
| LAYER_INFO_ZOOM_LAYERED        | Логическая величина (тип Logical), показывающая, применяется ли масштабный эффект для данного слоя.   |
| LAYER_INFO_ZOOM_MIN            | Число (тип Float), минимальный порог для масштабного эффекта (в текущих в MapBasic единицах измерения расстояний). Для установки единиц измерения расстояний используйте <b>Оператор Set Distance Units</b> .   |
| LAYER_INFO_ZOOM_MAX            | Число (тип Float), максимальный порог для масштабного эффекта.  |
| LAYER_INFO_COSMETIC            | Логическая величина (тип Logical). "Да" (TRUE), если данный слой косметический.   |
| LAYER_INFO_DISPLAY             | <p>Короткое целое число (SmallInt), задающее режим показа слоя:</p> <ul style="list-style-type: none"> <li>LAYER_INFO_DISPLAY_OFF (слой не показывается).</li> <li>LAYER_INFO_DISPLAY_GRAPHIC (объекты на слое показываются в собственном стиле, то есть в том, который сохранен в таблице).</li> <li>LAYER_INFO_DISPLAY_GLOBAL (объекты на слое показываются в стиле, заданном в диалоге "Управление слоями").</li> <li>LAYER_INFO_DISPLAY_VALUE (объекты на слое показываются в стиле, заданном режимам тематического выделения).</li> </ul>                    |
| LAYER_INFO_LABEL_ORIENTATION * | <p>LAYER_INFO_LABEL_ORIENTATION – возвращает короткое целое, описывающее ориентацию автоматически создаваемых подписей.. Результатом будет один из следующих кодов:</p> <ul style="list-style-type: none"> <li>LAYER_INFO_LABEL_ORIENT_HORIZONTAL (подписи под углом равным 0)</li> <li>LAYER_INFO_LABEL_ORIENT_PARALLEL (подписи под любым углом отличным от 0)</li> <li>LAYER_INFO_LABEL_ORIENT_CURVED (подписи по кривой)</li> </ul> <p><b>Внимание:</b> Если возвращается LAYER_INFO_LABEL_ORIENT_PARALLEL, то LAYER_INFO_LABEL_PARALLEL возвращает TRUE.</p> |

| Значения attribute      | Результат LayerInfo( )  |
|-------------------------|---|
| LAYER_INFO_OVR_LINE     | Стиль линии, используемый для отображения линейных объектов.  |
| LAYER_INFO_OVR_PEN      | Стиль линии, используемый для отображения контуров объектов, имеющих площадь.   |
| LAYER_INFO_OVR_BRUSH    | Стиль штриха, используемый для отображения объектов, имеющих площадь.   |
| LAYER_INFO_OVR_SYMBOL   | Стиль символа, используемый для отображения точечных объектов.  |
| LAYER_INFO_OVR_FONT     | Стиль шрифта, используемый для отображения текстовых объектов.  |
| LAYER_INFO_LBL_CURFONT  | <p>В приложениях, откомпилированных в среде MapBasic версии 3.x, результатом будет:</p> <p>"Да" (TRUE), если слой использует текущую установку для стиля шрифта, или "Нет" (FALSE), если слой использует специальную установку для стиля шрифта (смотрите LAYER_INFO_LBL_FONT).</p> <p>В приложениях, откомпилированных с MapBasic версии 4.0 и выше, результат всегда будет FALSE.</p> |
| LAYER_INFO_LBL_FONT     | Стиль шрифта для подписей.  |
| LAYER_INFO_LBL_EXPR     | Строка: выражение, создающее подписи.   |
| LAYER_INFO_LBL_LT       | <p>Целочисленный код (тип SmallInt), показывающий, какой используется тип указки. Возвращаются следующие значения:</p> <ul style="list-style-type: none"> <li>LAYER_INFO_LBL_LT_NONE (указки нет)</li> <li>LAYER_INFO_LBL_LT_SIMPLE (указка является простой линией)</li> <li>LAYER_INFO_LBL_LT_ARROW (указка является линией со стрелкой)</li> </ul>                                   |
| LAYER_INFO_LBL_PARALLEL | Логическая величина: "Да" (TRUE), если на слое разрешено поворачивать подписи.  |

| Значения attribute         | Результат LayerInfo( )  |
|----------------------------|---|
| LAYER_INFO_LBL_POS         | <p>Целочисленный код (тип SmallInt), показывающий ориентацию подписи. Литера Т обозначает верх, литера В – низ, С – центр, R – право, L – лево.</p> <ul style="list-style-type: none"> <li>• LAYER_INFO_LBL_POS_TL</li> <li>• LAYER_INFO_LBL_POS_TC</li> <li>• LAYER_INFO_LBL_POS_TR</li> <li>• LAYER_INFO_LBL_POS_CL</li> <li>• LAYER_INFO_LBL_POS_CC</li> <li>• LAYER_INFO_LBL_POS_CR</li> <li>• LAYER_INFO_LBL_POS_BL</li> <li>• LAYER_INFO_LBL_POS_BC</li> <li>• LAYER_INFO_LBL_POS_BR</li> </ul> |
| LAYER_INFO_LBL_VISIBILITY  | <p>Целочисленная величина, типа Smallint, соответствующая режиму показа подписей на слое (смотрите описание предложения <b>Visibility</b> в разделе <b>Оператор Set Map on page 650</b>. Результатом будет один из следующих кодов:</p> <ul style="list-style-type: none"> <li>• LAYER_INFO_LBL_VIS_ON (подписи на слое видимы)</li> <li>• LAYER_INFO_LBL_VIS_OFF (подписи на слое не видимы)</li> <li>• LAYER_INFO_LBL_VIS_ZOOM (подписи видимы в результате масштабного эффекта)</li> </ul>         |
| LAYER_INFO_LBL_ZOOM_MIN    | <p>Действительная величина (тип Float), минимальное значение показа подписей при масштабном эффекте.</p>  |
| LAYER_INFO_LBL_ZOOM_MAX    | <p>Действительная величина (тип Float), максимальное значение показа подписей при масштабном эффекте.</p>   |
| LAYER_INFO_LBL_AUTODISPLAY | <p>Логическая величина: TRUE, если слой подписывается автоматически. Смотрите описание предложения <b>Auto</b> в разделе <b>Оператор Set Map on page 650</b>.</p>   |
| LAYER_INFO_LBL_OVERLAP     | <p>Логическая величина: "Да" (TRUE), если допускается пересечение подписей.</p>   |
| LAYER_INFO_LBL_DUPLICATES  | <p>Логическая величина: "Да" (TRUE), если допускается дублирование подписей.</p>  |
| LAYER_INFO_LBL_OFFSET      | <p>Целочисленная величина типа Smallint о 0 до 50, смещение от подписи до центроида. Измеряется в точках (points).</p>  |

| Значения attribute         | Результат LayerInfo( )  |
|----------------------------|---|
| LAYER_INFO_LBL_MAX         | Целочисленная величина типа Integer, максимальное число подписей, разрешенное для этого слоя. Если такое число не назначено, то возвращается число 2 147 483 647.   |
| LAYER_INFO_LBL_PARTIALSEGS | Логическая величина; TRUE, если для этого слоя установлен флажок <b>Label Partial Objects</b> .   |
| LAYER_INFO_ARROWS          | Логическая величина: "Да" (TRUE), если в линейных объектах используется стрелка.  |
| LAYER_INFO_NODES           | Логическая величина: "Да" (TRUE), если показываются узлы объектов слоя.   |
| LAYER_INFO_CENTROIDS       | Логическая величина: "Да" (TRUE), если показываются центроиды объектов слоя.  |
| LAYER_INFO_SELECTABLE      | Логическая величина (тип Logical), показывающая, доступен ли слой или нет.  |
| LAYER_INFO_PATH            | Строка (тип String), содержащая DOS-маршрут, по которому находится файл таблицы, данные из которой представлены на этом слое.   |
| LAYER_INFO_TYPE            | <p>Целочисленный код (тип SmallInt), показывающий тип слоя:</p> <ul style="list-style-type: none"> <li>• LAYER_INFO_TYPE_NORMAL, если слой нормальный</li> <li>• LAYER_INFO_TYPE_COSMETIC, если слой Косметический</li> <li>• LAYER_INFO_TYPE_IMAGE, если слой растровый</li> <li>• LAYER_INFO_TYPE_THEMATIC, если слой тематический</li> <li>• LAYER_INFO_TYPE_GRID, если это растровый грид-слой</li> <li>• LAYER_INFO_TYPE_WMS, если слой получен от службы Web Service</li> </ul> |
| LAYER_HOTLINK_EXPR         | Возвращает выражение для файла, на который указывает геолинк. Допускается пустая строка ("")  |
| LAYER_INFO_HOTLINK_COUNT   | Позволяет задавать несколько геолинков на слое карты.   |

| Значения attribute     | Результат LayerInfo( )  |
|------------------------|---|
| LAYER_HOTLINK_MODE     | Возвращает текущий режим геолинка, один из следующих: <ul style="list-style-type: none"><li>HOTLINK_MODE_LABEL (default)</li><li>HOTLINK_MODE_OBJ</li><li>HOTLINK_MODE_BOTH</li></ul> |
| LAYER_HOTLINK_RELATIVE | Возвращает "Да" (TRUE), если маршрут в геолинке записан в относительной форме и "Нет" (FALSE) в обратном случае. По умолчанию – FALSE.  |

**Внимание:** \*Значения, которые возвращаются функцией LayerInfo() для LAYER\_INFO\_LABEL\_ORIENTATION и функцией LabelInfo() для LABEL\_INFO\_ORIENTATION, соответствуют константам MapBasic, заданным в MAPBASIC.DEF. Если что-то будет изменено, то потребуется изменить MAPBASIC.DEF.

```
#define LAYER_INFO_LABEL_ORIENT_HORIZONTAL 0
#define LAYER_INFO_LABEL_ORIENT_PARALLEL 1
#define LAYER_INFO_LABEL_ORIENT_CURVED 2
```

**Пример:**

Многие коды из приведенных выше не применимы к Косметическому слою, тематическому слою и слою с растровым изображением. Функцию **LayerInfo( )** с аргументом LAYER\_INFO\_TYPE используют для определения, каким является слой.

Например:

```
i_layer_type = LayerInfo( map_id, layer_number, LAYER_INFO_TYPE) If
i_layer_type = LAYER_INFO_TYPE_NORMAL Then ' ' ... то это "нормальный" слой
'End If
```

**См. также:**

[Функция MapperInfo\( \), Оператор Set Map](#)

Оператор Layout

**Назначение**

Открывает новое окно Отчета.

**Синтаксис**

```
Layout
[ Position ( x, y ) [ Units paperunits ] ]
[ Width window_width [ Units paperunits ] ]
```



```
[ Height window_height [ Units paperunits ] ]
[ { Min | Max } ]
```

*paperunits* – имя единицы измерения (например, "in" – дюйм);

*x*, *y* – координаты левого верхнего угла окна Отчета в определенных единицах измерения;

*window\_width* и *window\_height* – ширина и высота окна в "бумажных" единицах.

### Описание

Оператор **Layout** создает новое пустое окно Отчета. Если задано ключевое слово **Min**, окно Отчета показывается минимизированным. Если задано ключевое слово **Max**, окно Отчета заполняет все рабочее пространство MapInfo Professional.

Предложения **Width** и **Height** определяют ширину и высоту окна Отчета. Но эти характеристики не влияют на размеры страницы макета.

Для назначения размеров страниц и их количества используйте оператор **Оператор Set Layout on page 645**.

MapInfo Professional организует для каждого окна Отчета специальную, скрытую таблицу, которой дается имя LayoutN. Первому открытому окну Отчета дается имя "Layout1", следующему "Layout2" и так далее.

Программа, написанная на MapBasic, может создавать, выбирать или изменять объекты в окне Отчета, обращаясь к нему по этому имени. Например, следующий оператор выбирает все объекты в окне Отчета:

```
Select * From Layout1
```

### Пример:

В следующем примере создается окно Отчета в два дюйма шириной и четыре высотой. Верхний левый угол окна располагается в верхнем левом углу рабочей области MapInfo.

```
Layout Position (0, 0) Width 2 Height 4
```

См. также:

**Оператор Open Window, Оператор Set Layout**

---

## Функция LCase\$( )

### Назначение

Возвращает строку, преобразуя все прописные буквы в строчные.

### Синтаксис

```
LCase$( string_expr ), где
```

*string\_expr* - выражение, результат которого есть строка.

### Возвращаемая величина

Строка

### Описание

Функция **LCase\$( )** возвращает строку, полученную из строки, представленной выражением `string_expr`, преобразованием всех заглавных букв в строчные.

Преобразованию подвергаются только буквы: латинские – от А до Z, и русские – от А до Я. Цифры и другие текстовые символы не преобразуются. Например:

```
LCase$( "A#12a" )
```

возвращает строку "a#12a".

**Пример:**

```
Dim regular, lower_case As String
regular = "Вышний Волочек"
lower_case = LCase$(regular) ' ' Первый элемент массива равен строке "вышний волочек" '
```

**См. также:**

**Функция Proper\$( ), Функция UCase\$( )**

**Функция Left\$( )****Назначение**

Возвращает левую часть строки, выделяя определенное количество символов из исходной.

**Синтаксис**

```
Left$( string_expr, num_expr )
```

*string\_expr* - выражение, результат которого есть строка.

*num\_expr* – численное выражение, результат которого ноль или более.

**Возвращаемая величина**

Строка

**Описание**

Функция **Left\$( )** возвращает строковую величину, полученную из самых левых *num\_expr* символов строки *string\_expr*.

Параметр *num\_expr* должен принимать положительное целочисленное значение, ноль или больше. Если значение параметра *num\_expr* нецелочисленное, то MapBasic округлит его до целого. Если *num\_expr* равно нулю, **Left\$( )** возвращает пустую строку. Если численный параметр *num\_expr* больше, чем число символов в строке *string\_expr*, результат функции **Left\$( )** будет равен строке, представленной выражением *string\_expr*.

**Пример:**

```
Dim whole, partial As String
whole = "Казахстан"
partial = Left$(whole, 6) '
теперь переменная partial содержит строку "Казах"
```

**См. также:**

**Функция Mid\$( ), Функция Right\$( )**

**Функция LegendFrameInfo( )****Назначение**

Возвращает информацию о разделе в легенде.

### Синтаксис

**LegendFrameInfo**(*window\_id*, *frame\_id*, *style\_id*, *attribute*)

*window\_id* – число, указывающее какое окно легенды Вы хотите опросить;

*frame\_id* – число, указывающее какой раздел в окне легенды Вы хотите опросить; Разделы пронумерованы от 1 до *n*, где *n* это номер в легенде.

*attribute* – это целочисленный код, указывающий какой тип информации возвращается.

### Возвращаемая величина

Тип результата зависит от значения параметра *attribute*.

| Коды атрибута            | LegendFrameInfo( ) возвращает  |
|--------------------------|--|
| FRAME_INFO_TYPE          | Возвращает одно из следующих предопределенных констант, определяющих тип раздела: <ul style="list-style-type: none"> <li>FRAME_TYPE_STYLE</li> <li>FRAME_TYPE_THEME</li> </ul> |
| FRAME_INFO_MAP_LAYER_ID  | Возвращает индекс id слоя, с которым соотносится раздел.   |
| FRAME_INFO_REFRESHABLE   | Возвращает TRUE, если раздел был создан без ключевого слова <b>Norefresh</b> . Всегда возвращает TRUE для тематических разделов.   |
| FRAME_INFO_POS_X         | Возвращает расстояние от верхнего левого угла раздела до левого края канвы легенды (в "бумажных" единицах).  |
| FRAME_INFO_POS_Y         | Возвращает расстояние от верхнего левого угла раздела до верхнего края канвы легенды (в "бумажных" единицах).  |
| FRAME_INFO_WIDTH         | Возвращает ширину раздела (в "бумажных" единицах).   |
| FRAME_INFO_HEIGHT        | Возвращает высоту раздела (в "бумажных" единицах).   |
| FRAME_INFO_TITLE         | Возвращает заголовок раздела или тематического раздела.  |
| FRAME_INFO_TITLE_FONT    | Возвращает шрифт заголовка раздела. Возвращает стандартный шрифт заголовка, если раздел не имеет заголовка или это тематический раздел.  |
| FRAME_INFO_SUBTITLE      | Возвращает подзаголовок раздела.   |
| FRAME_INFO_SUBTITLE_FONT | То же, что и FRAME_INFO_TITLE_FONT.  |

| Коды атрибута         | LegendFrameInfo( ) возвращает  |
|-----------------------|--|
| FRAME_INFO_BORDER_PEN | Возвращает параметры линии, использованной для рамки.  |
| FRAME_INFO_NUM_STYLES | Возвращает число типа раздела. Ноль для тематического раздела.   |
| FRAME_INFO_VISIBLE    | Возвращает TRUE, если раздел видимый (тематические разделы могут быть невидимыми).                               |
| FRAME_INFO_COLUMN     | Возвращает атрибуты имени колонки для легенды в виде строки. Возвращает пустую строку, если раздел тематический. |
| FRAME_INFO_LABEL      | Возвращает выражение подписи в виде строки. Возвращает пустую строку, если раздел тематический.                  |

## Функция LegendInfo( )

### Назначение

Возвращает информацию о легенде.

### Синтаксис

**LegendInfo**( *window\_id*, *attribute* )

*window\_id* – число, указывающее какое окно легенды Вы хотите опросить;

*attribute* – это целочисленный код, указывающий какой тип информации возвращается.

### Возвращаемая величина

Зависит от параметра attribute.

| Коды атрибута           | LegendInfo( ) возвращает  |
|-------------------------|---|
| LEGEND_INFO_MAP_ID      | Возвращает id порождающего окна Карты (для его получения так же используется <b>Функция WindowInfo( )</b> с кодом WIN_INFO_TABLE).  |
| LEGEND_INFO_ORIENTATION | Возвращает предопределенное значение, характеризующее ориентацию легенды: <ul style="list-style-type: none"> <li>• ORIENTATION_PORTRAIT</li> <li>• ORIENTATION_LANDSCAPE</li> <li>• ORIENTATION_CUSTOM</li> </ul> |

## Функция LegendStyleInfo( )

| Коды атрибута                 | LegendInfo( ) возвращает   |
|-------------------------------|--|
| LEGEND_INFO_NUM_FRAMES        | Возвращает число разделов в легенде                                  |
| LEGEND_INFO_STYLE_SAMPLE_SIZE | Возвращает 0 для малых образцов символов<br>Легенды и 1 для больших. |

### Пример:

```
LegendInfo(FrontWindow( ) LEGEND_INFO_STYLE_SAMPLE_SIZE)
```

### См. также:

[Функция LegendStyleInfo\( \)](#)

## Функция LegendStyleInfo( )

### Назначение

Возвращает информацию о стиле, используемом в разделе легенды.

### Синтаксис

**LegendStyleInfo**(*window\_id*, *frame\_id*, *style\_id*, *attribute*)

*window\_id* – число, указывающее какое окно легенды Вы хотите опросить;

*frame\_id* – число, указывающее какой раздел в окне легенды Вы хотите опросить; Разделы пронумерованы от 1 до *n*, где *n* это номер в легенде.

*style\_id* – это число, определяющее, какой стиль внутри раздела Вы хотите опросить; Стили пронумерованы от 1 до *n*, где *n* это число стилей в разделе.

*attribute* – это целочисленный код, указывающий какой тип информации возвращается.

### Возвращаемая величина

| Коды атрибута          | LegendStyleInfo( ) возвращает |
|------------------------|-------------------------------|
| LEGEND_STYLE_INFO_TEXT | Возвращает текст стиля.       |
| LEGEND_STYLE_INFO_FONT | Возвращает шрифт стиля.       |
| LEGEND_STYLE_INFO_OBJ  | Возвращает объект стиля.      |

### Ошибки:

Генерируется сообщение об ошибке, когда раздел не имеет стилей (тематический раздел).

### См. также:

[Функция LegendInfo\( \)](#)

## Функция Len( )

### Назначение

Возвращает количество символов в строке или число байтов в переменной.

### Синтаксис

**Len** ( *expr* )

*expr* – выражение; *expr* не может иметь тип Pen, Brush, Symbol, Font или Alias.

### Возвращаемая величина

Целое число типа SmallInt.

### Описание

Информация, которую несет величина, возвращаемая функцией **Len( )**, зависит от типа результата выполнения *expr*.

Если выражение *expr* является строкой, то функция **Len( )** вернет количество символов в строке.

Если *expr* – переменная MapBasic, то результат функции **Len( )** будет размером переменной в байтах. Так, если тип переменной был объявлен как Integer (целое число), то функция **Len( )** т. к. переменной типа Integer отводится 4 байта. Для переменной типа SmallInt (короткое целое число), результатом функции **Len( )** т. к. переменной типа SmallInt отводится 2 байта.

### Пример:

```
Dim name_length As SmallInt
name_length = Len("Москва") ' переменная
name_length равна 6
```

### См. также:

**Функция ObjectLen( )**

## Функция Like( )

### Назначение

Возвращает TRUE или FALSE, сравнивая строку с шаблоном.

### Синтаксис

**Like** ( *string*, *pattern\_string*, *escape\_char* )

*string* – строка;

*pattern\_string* – шаблон для сравнения, который является строкой, состоящей из регулярных и специальных символов;

*escape\_char* – строковое выражение, задающее символ отмены проверки следующего за ним символа. В этой строке можно использовать "отменяющий" символ “\” в сочетании с символами, обозначающими один или несколько произвольных символов (“%” и “\_”). Если отменяющий символ не назначается, то используется пустая строка (“”).

### Возвращаемая величина

Логическое значение (TRUE , если строка совпадает с *pattern\_string*).

### Описание

Так же, как оператор **Like( )**, выполняет сравнение строк. В отличие от оператора Like, строчные и прописные буквы различаются.

Строка *pattern\_string* может содержать следующие символы, обозначающими один или несколько произвольных символов:

|                        |   |
|------------------------|---|
| _ (знак подчеркивания) | соответствует одному символу                    |
| % (процент)            | соответствует нескольким символам или не одному |

Для явного задания знаков подчеркивания и процента они используются вместе с символом *escape\_char* перед специальным. Примеры приведены в следующей таблице:

| Критерий совпадения        | Используйте функцию с аргументами               |
|----------------------------|---|
| начинается с “South”       | <code>Like( string_var, "South%", "" )</code>   |
| кончается на “America”     | <code>Like( string_var, "%America", "" )</code> |
| содержит “ing”             | <code>Like( string_var, "%ing%", "" )</code>    |
| начинается с подчеркивания | <code>Like( string_var, "\_%", "" )</code>      |

См. также:

[Функция Len\( \)](#), [Функция StringCompare\( \)](#)

## Оператор Line Input

### Назначение

Читает строку из текстового файла в переменную.

### Синтаксис

**Line Input** [#] *filenum*, *var\_name*



*filenum* – номер открытого файла, целое число;  
*var\_name* – имя переменной строкового типа.

Описание

Оператор **Line Input** читает текущую строку из текстового файла в переменную типа String. Текстовый файл должен быть открыт для последовательного доступа (INPUT).

Оператор **Line Input** читает каждую строку полностью. Если строка содержит список выражений, разделенный запятыми и Вы хотите каждое выражение присвоить отдельной переменной, то используйте **Оператор Input #** вместо **Line Input**.

Пример:

В нижеприведенном коде читается строка за строкой из первого файла и копируется во второй файл.

```
Dim str As String
Open File "original.txt" For Input As #1
Open File "copy.txt" For Output As #2
  Do While Not EOF(1)
    Line Input #1, str
    If Not EOF(1) Then
      Print #2, str
    End If
  Loop
Close File #1
Close File #2
```

См. также:

**Оператор Input #, Оператор Open File, Оператор Print #**

Функция LocateFile\$( )

Назначение

Возвращает маршрут к файлам данных MapInfo.

Синтаксис

**LocateFile\$( file\_id )**  
*file\_id* – см. таблицу ниже.

| Значение         | Описание   |
|------------------|--|
| LOCATE_PREF_FILE | Настройки (mapinfow.prf).                        |
| LOCATE_DEF_WOR   | Стандартный файл Рабочего набора (mapinfow.wor). |

| Значение                  | Описание   |
|---------------------------|--|
| LOCATE_CLR_FILE           | Определения цветов (mapinfow.clr).                         |
| LOCATE_PEN_FILE           | Стили линий (mapinfow.pen).                                |
| LOCATE_FNT_FILE           | Символы (mapinfow.fnt).                                    |
| LOCATE_ABB_FILE           | Адресные сокращения (mapinfow.abb).                        |
| LOCATE_PRJ_FILE           | Проекции (mapinfow.prj).                                   |
| LOCATE_MNU_FILE           | Меню (mapinfow.mnu).                                       |
| LOCATE_CUSTSYMB_DIR       | Символы, задаваемый пользователем (папка CUSTSYMB).        |
| LOCATE_THMTMPLT_DIR       | Тематические шаблоны (папка THMTMPL).                      |
| LOCATE_GRAPH_DIR          | Поддержка графики (папка GRAPH_SUPPORT).                   |
| LOCATE_WMS_SERVERLIST     | XML список WMS-серверов (MIWMSservers.xml).                |
| LOCATE_WFS_SERVERLIST     | XML список WFS-серверов (MIWFSservers.xml).                |
| LOCATE_GEOCODE_SERVERLIST | XML список серверов геокодирования (MIGeocodeServers.xml). |
| LOCATE_ROUTING_SERVERLIST | XML список серверов маршрутизации (MIRoutingServers.xml).  |

### Возвращаемая величина

Строка

### Описание

Эта функция, используя ID-номер файла данных MapInfo Professional, возвращает маршрут к нему. В версиях до 6.5 эти файлы в основном устанавливались в тот же каталог, что и файл MAPINFOW.EXE. Начиная с версии 6.5, установщик MapInfo Professional помещает эти файлы в пользовательский подкаталог каталога Application Data. До сих пор поддерживается возможность помещения некоторых из этих файлов в другие каталоги, включая каталог с MAPINFOW.EXE. Программы MapBasic, которые используют эти файлы, должны, однако, придерживаться правила всегда использовать функцию **LocateFile\$( )**.

### Пример:

```
include "mapbasic.def"
declare sub main
sub main
dim sGraphLocations as string
sGraphLocations = LocateFile$(LOCATE_GRAPH_DIR)
```

```
Print sGraphLocations
end sub
```

**См. также:**

**Функция `GetFolderPath$ ( )`**

---

## Функция `LOF ( )`

### Назначение

Возвращает длину открытого файла.

### Синтаксис

```
LOF ( filenum )
```

*filenum* – номер файла, который был присвоен ему при открытии;

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция `LOF ( )` возвращает размер открытого файла в байтах.

Параметр `file` должен быть целочисленным номером файла; этот номер определяется в предложении **As**, которое использует **Оператор Open File**.

### Ошибки:

В результате выполнения оператора может генерироваться код ошибки `ERR_FILEMGR_NOTOPEN`, если файл не был открыт..

### Пример:

```
Dim size As Integer
Open File "import.txt" For Binary As #1
size = LOF(1) ' переменная size теперь равна размеру файла, открытого под номером 1
```

**См. также:**

**Оператор Open File**

---

## Функция `Log ( )`

### Назначение

Вычисляет натуральный логарифм.

### Синтаксис

```
Log ( num_expr )
```

## Функция LTrim\$ ( )

---

*num\_expr* – числовое выражение

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Log( )** возвращает значение натурального логарифма от числа, полученного в результате вычисления выражения *num\_expr*.

Функция натурального логарифма обратна функции экспоненты (число *e*) в степени *num\_expr*. Число *e* иррационально и приблизительно равно 2.7182818.

Логарифм может вычисляться только от положительного числа; функция **Log( )** вернет ошибку, если *num\_expr* есть отрицательная величина.

Вы можете вычислить логарифм и по другому основанию (например, 10), используя натуральный логарифм. Для вычисления логарифма по основанию 10 от числа *n* надо разделить натуральный логарифм от числа *n* (**Log( *n* )**) на натуральный логарифм от 10 (**Log( 10 )**).

### Пример:

```
Dim original_val, log_val As Float
original_val = 2.7182818
log_val = Log(original_val) ' log_val будет равно 1 (приблизительно), ' т. к. число e в степени 1 (единица) равно 2.7182818 (приблизительно)
```

### См. также:

[Функция Exp\( \)](#)

---

## Функция LTrim\$ ( )

### Назначение

Удаляет пробелы в начале строки.

### Синтаксис

**LTrim\$( *string\_expr* )**, где

*string\_expr* - выражение, результат которого есть строка.

### Возвращаемая величина

Строка

### Описание

Функция **LTrim\$( )** возвращает строковую величину, полученную из выражения *string\_expr* удалением пробелов в начале строки, если они есть.

**Пример:**

```
Dim name As String name = "    Мария Анатольевна Смирнова" name =
LTRim$(name) ' name теперь имеет значение "Мария Анатольевна Смирнова"
```

**См. также:**

**Функция RTrim\$( )**

---

## Процедура Main

**Назначение**

Главная процедура, которая выполняется первой при загрузке прикладной программы.

**Синтаксис**

```
Declare Sub Main
Sub Main
    statement_list
End Sub
```

*statement\_list* – список операторов, составляющих программу.

**Описание**

**Main** – стандартное имя процедуры MapBasic. Если текст программы на MapBasic содержит процедуру **Main**, то выполнение программы начнется с этой процедуры. Процедура **Main** может вызывать другие процедуры (смотрите раздел **Оператор Call**).

Вы можете не объявлять процедуру **Main**. В этом случае первый оператор программы понимается как оператор из процедуры **Main**. MapBasic начинает выполнять программу, как если бы процедура **Main** была объявлена перед этим оператором. Назовем этот случай "неявным" заданием процедуры **Main** (как противоположность "явному" заданию **Main**).

**Пример:**

Прикладная программа на языке MapBasic может состоять из одной строки. Например, эта программа выполняет только один оператор:

```
Note "Проверка: один, два, три. Как видно?"
```

Вы можете компилировать использовать этот оператор как полноценную программу, при этом MapBasic будет неявно считать ее содержанием процедуры Main. При запуске этой программы, MapBasic выполняет **Оператор Note**.

Мы можем включить в эту программу объявление процедуры **Main** и она выполнит то же самое (т.е. **Оператор Note**).

```
Declare Sub Main Sub MainNote "Проверка: один, два, три. Как видно?"End Sub
```

Следующая программа также содержит неявную процедуру **Main**. Из нее вызывается подпрограмма, процедура которой объявлена под именем "Talk". Для вызова последней в процедуре **Main** используется **Оператор Call**.

## Функция MakeBrush( )

---

```
Declare Sub Talk(ByVal msg As String)Call Talk("Привет!")Call Talk("Всего  
хорошего") Sub Talk(ByVal msg As String)Note msgEnd Sub
```

Следующий пример содержит явную процедуру **Main**, из которой вызывается подпрограмма "Talk". Для вызова последней в процедуре **Main** также используется .

```
Declare Sub MainDeclare Sub Talk(ByVal msg As String)Sub Main Call  
Talk("Привет!") Call Talk("Всего хорошего") End SubSub Talk(ByVal msg As  
String)Note msg End Sub
```

**См. также:**

[Процедура EndHandler](#), [Процедура RemoteMsgHandler](#), [Процедура SelChangedHandler](#),  
[Оператор Sub...End Sub](#), [Процедура ToolHandler](#), [Процедура WinClosedHandler](#)

---

## Функция MakeBrush( )

### Назначение

Возвращает установку стиля штриха.

### Синтаксис

**MakeBrush**( *pattern*, *forecolor*, *backcolor*)

*pattern* – тип штриха, целое число от 1 до 8 или от 12 до 71. Рисунки штриха приведены в разделе [Предложение Brush on page 117](#).

*forecolor* – цвет штриха в системе RGB. Подробнее см. раздел [Функция RGB\( \) on page 536](#).

*backcolor* – цвет фона в системе RGB. Чтобы сделать фон прозрачным, задайте значение backcolor -1 и значение pattern 3 или более.

### Возвращаемая величина

Brush

### Описание

Функция **MakeBrush( )** возвращает величину типа Brush, определяющую стиль штриховки графического объекта. Возвращаемая величина может быть присвоена переменной типа Brush или использована как параметр оператора (таких как Create Ellipse, Set Map, Set Style или Shade).

См. более подробную информацию в разделе [Предложение Brush on page 117](#).

### Пример:

```
Include "mapbasic.def"Dim b_water As Brushb_water = MakeBrush(64, CYAN,  
BLUE)
```

**См. также:**

[Предложение Brush](#), [Функция CurrentBrush\( \)](#), [Функция RGB\( \)](#), [Функция StyleAttr\( \)](#)

## Функция MakeCustomSymbol( )

### Назначение

Возвращает символ, созданный из растрового файла.

### Синтаксис

**MakeCustomSymbol**( *filename*, *color*, *size*, *customstyle* )

*filename* – строка до 31 символа длиной с именем растрового файла. Файл должен находиться в каталоге CUSTSYMB в каталоге пользователя MapInfo.

*color* – цвет в системе RGB; см. также раздел [Функция RGB\( \) on page 536](#).

*size* – целое число, величина типа Integer, размер символа в точках от 1 до 48;

*customstyle* – целочисленный код типа Integer, управляющий цветом и фоном символа. См. таблицу ниже.

### Возвращаемая величина

Символ

### Описание

Функция **MakeCustomSymbol( )** возвращает величину типа Symbol, основанную на растровом файле. Смотрите в разделе [Предложение Symbol on page 725](#) описание других типов символа.

В следующей таблице перечислены возможности настройки растрового символа:

| Значение<br>customstyle | Стиль символа  |
|-------------------------|--|
| 0                       | Не действуют режимы из группы “Эффекты” диалога “Стиль символа”, и символ появляется таким, какой он есть. Все белые пиксели растра прозрачны. |
| 1                       | Действует режим “Добавить фон”; все белые пиксели растра непрозрачны.  |
| 2                       | Действует режим “Покрасить одним цветом”; все не белые точки растра закрашены одним цветом.  |
| 3                       | Установлены оба флажка (действуют оба режима).   |
| 4                       | Действует режим “Показать реальный размер”; растровый символ визуализируется в свою натуральную величину.                                      |

## Функция MakeDateTime()

---

| Значение<br>customstyle | Стиль символа  |
|-------------------------|--|
| 5                       | Действуют режимы "Добавить фон" и "Показать реальный размер".                            |
| 7                       | Действуют режимы: "Добавить фон", "Покрасить одним цветом" и "Показать реальный размер". |

### Пример:

```
Include "mapbasic.def"  
Dim sym_marker As Symbol  
sym_marker = MakeCustomSymbol("CAR1-64.BMP", BLUE, 18, 0)
```

### См. также:

[Функция CurrentSymbol\( \)](#), [Функция MakeFontSymbol\( \)](#), [Функция MakeSymbol\( \)](#),  
[Функция StyleAttr\( \)](#), [Предложение Symbol](#)

---

## Функция MakeDateTime()

### Назначение

Возвращает значение DateTime, составленное из заданных значений Date и Time.

### Синтаксис

```
MakeDateTime (Date, Time)
```

### Возвращаемая величина

DateTime

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim tX as time  
dim dX as date  
dim dtX as datetime  
tX = 105604123  
dX = 20070908  
dtX = MakeDateTime(dX,tX)  
Print FormatDate$(GetDate(dtX))  
Print FormatTime$(GetTime(dtX), "hh:mm:ss.fff tt")
```

---

## Функция MakeFont( )

### Назначение

Возвращает величину, являющуюся установкой стиля шрифта.



**Синтаксис**

**MakeFont**( *fontname, style, size, forecolor, backcolor* )

*fontname* – имя шрифта, строковая величина (например, "Arial"); Прописные и строчные буквы для этого параметра различаются.

*style* – численное выражение, в результате которого получается положительное целое число; подробности приведены в описании предложения Font.

*size* – размер шрифта, целое число;

*forecolor* – цвет символов шрифта в системе RGB; См. [Функция RGB\( \) on page 536](#).

*backcolor* – цвет фона или каймы в системе RGB. Чтобы сделать фон прозрачным, задайте значение -1.

**Возвращаемая величина**

Шрифт

**Описание**

Функция **MakeFont( )** возвращает величину типа Font для определения шрифта, который может быть назначен текстовому объекту. Возвращаемая величина может быть присвоена переменной типа Font или использована как параметр в других операторах (таких как [Оператор Create Text](#) или [Оператор Set Style](#)).

Для дополнительной информации о стиле шрифта см. раздел [Предложение Font on page 317](#).

**Пример:**

```
Include "mapbasic.def"
Dim big_title As Font
big_title = MakeFont("Arial", 1, 20,BLACK,WHITE)
```

**См. также:**

[Функция CurrentFont\( \)](#), [Предложение Font](#), [Функция StyleAttr\( \)](#)

---

**Функция MakeFontSymbol( )****Назначение**

Возвращает символ, используя букву (символ) шрифта TrueType.

**Синтаксис**

**Symbol** ( *shape, color, size, fontname, fontstyle, rotation* ), где

*shape* – целое число, величина типа SmallInt, от 31 или больше (31 – значение для невидимого символа), задающая код шрифта TrueType;

*color* – цвет в системе RGB; см. также раздел [Функция RGB\( \) on page 536](#).

*size* – целое число, величина типа SmallInt, от 1 до 48, назначающая размер символа в пунктах;

*fontname* – строка, имя шрифта TrueType (например, "WingDings"). Прописные и строчные буквы для этого параметра различаются.

*fontstyle* – численный код, управляющий атрибутами шрифта, такими как жирное написание, курсив, контур;

*rotation* – действительное число, задающее угол поворота символа в градусах.

### Возвращаемая величина

Символ

### Описание

Функция **MakeFontSymbol( )** возвращает величину типа Symbol, используя заданный символ шрифта TrueType. Смотрите в разделе [Предложение Symbol on page 725](#) описание других типов символа.

Следующая таблица приводит значения для параметра *fontstyle* задающего стиль символа шрифта:

| значение fontstyle | Стиль символа |
|--------------------|---------------|
| 0                  | Нормальное    |
| 1                  | Жирное        |
| 16                 | Черная кайма  |
| 32                 | Оттененное    |
| 256                | Белая кайма   |

Для задания двух или более стилей написания коды складываются. Например, для того, чтобы получить символ жирного и оттененного написания, параметр fontstyle должен быть равен 33. Белая и черная кайма взаимно исключают друг друга.

### Пример:

```
Include "mapbasic.def"  
Dim sym_marker As Symbol  
sym_marker = MakeFontSymbol(65,RED,24,"WingDings",32,0)
```

### См. также:

[Функция CurrentSymbol\( \)](#), [Функция MakeCustomSymbol\( \)](#), [Функция MakeSymbol\( \)](#), [Функция StyleAttr\( \)](#), [Предложение Symbol](#)

---

## Функция MakePen( )

### Назначение

Возвращает установку стиля линии.

### Синтаксис

**MakePen**( *width*, *pattern*, *color*)

*width* – толщина линии;

*pattern* – тип линии (см. список в описании предложения Pen);

*color* – цвет линии в системе RGB. См. подробности в разделе [Функция RGB\( \) on page 536](#).

### Возвращаемая величина

Pen

### Описание

Функция **MakePen( )** возвращает величину типа Pen, определяющую стиль линии графического объекта. Возвращаемая величина может быть присвоена переменной типа Pen или использована как параметр в других операторах (таких как [Оператор Create Line](#), [Оператор Create Pline](#), [Оператор Set Style](#) или [Оператор Set Map](#)).

См. раздел [Предложение Pen on page 492](#) для дополнительной информации о стиле линии.

### Пример:

```
Include "mapbasic.def"  
Dim p_bus_route As Pen  
p_bus_route = MakePen(3, 9, RED)
```

### См. также:

[Функция CurrentPen\( \)](#), [Предложение Pen](#), [Функция StyleAttr\( \)](#), [Функция RGB\( \)](#)

---

## Функция MakeSymbol( )

### Назначение

Возвращает установку стиля символа. Действует для символов формата MapInfo версии 3. Набор символов версии 3.0, введенный в MapInfo для работы в среде Windows 3.0, поддерживается во всех последующих версиях MapInfo Professional.

### Синтаксис

**MakeSymbol**( *shape*, *color*, *size* )

*shape* – форма символа, целое число от 31 и более (31 для невидимого знака); стандартный набор символов использует коды от 31 до 67 (список смотрите в разделе [Предложение Symbol on page 725](#)).

*color* – цвет в системе RGB; см. также раздел [Функция RGB\( \) on page 536](#).

*size* – целое число, величина типа SmallInt, от 1 до 48, назначающая размер символа в пунктах;

### Возвращаемая величина

Символ

### Описание

Функция **MakeSymbol( )** возвращает величину типа Symbol, определяющую стиль отображения точечного объекта. Возвращаемая величина может быть присвоена переменной типа Symbol или использована как параметр в других операторах (таких как [Предложение Symbol](#) [Оператор Create Point](#), [Оператор Set Map](#), [Оператор Set Style](#) или [Оператор Shade](#)).

Чтобы создать символ из буквы или знака шрифта TrueType, используется [Функция MakeFontSymbol\( \)](#).

Чтобы создать символ из растрового файла, используется [Функция MakeCustomSymbol\( \)](#).

См. раздел [Предложение Symbol on page 725](#) для дополнительной информации о стиле символа.

### Пример:

```
Include "mapbasic.def"
Dim sym_marker As Symbol
sym_marker = MakeSymbol(44, RED, 16)
```

### См. также:

[Функция CurrentSymbol\( \)](#), [Функция MakeCustomSymbol\( \)](#), [Функция MakeFontSymbol\( \)](#), [Функция StyleAttr\( \)](#), [Предложение Symbol](#)

---

## Оператор Map

### Назначение

Открывает новое окно Карты.

### Синтаксис

```
Map From table [ , table ... ]
[ Position ( x, y ) [ Units paperunits ] ]
[ Width window_width [ Units paperunits ] ]
[ Height window_height [ Units paperunits ] ]
[ { Min | Max } ]
```

*table* – имя открытой таблицы;

*paperunits* – строковая величина, задающая единицу измерения листа (например, "mm");

*x, y* – координаты верхнего левого угла окна Карты в "бумажных" единицах;

*window\_width* и *window\_height* – ширина и высота окна в "бумажных" единицах.

### Описание

Оператор **Map** открывает новое окно Карты. После выполнения оператора приложение может изменять это окно, используя **Оператор Set Map**.

Таблица должна быть заранее открыта. Таблица должна быть также картографируемой, то есть устроена так, чтобы к ней можно было присоединять графические объекты. Не обязательно, чтобы в таблице были графические объекты, но она должна иметь структуру, позволяющую иметь объекты, связанные с информацией в записях.

Оператор **Map** должен содержать указание хотя бы на одну таблицу, так как Карта должна иметь хотя бы один слой, кроме Косметического. Если Вы хотите одним оператором **Map** открыть окна Карты для нескольких таблиц, то задайте их списком через запятую. В этом же порядке оператор **Map** в этом же порядке таблицы будут отображены в слоях карты; то есть : первая таблица в операторе **Map** помещается на верхнем слое. Обычно первой (на верхний слой) оператор **Map** размещает таблицу с точечными объектами, а таблицу с областями (границами) **Map** помещает на нижний слой.

Размер окна по умолчанию равен примерно четверти экрана, и положение его зависит от того, сколько окон уже открыто. Задайте **Position, Height** и **Width**, если Вы хотите при создании карты сами определить размер и место окна Карты. Предложения **Height** и **Width** задают размеры окна в дюймах. Предложение **Position** задает расположение окна в окне MapInfo, независимо от его расположения на экране.

Если в операторе **Map** задано предложение **Max**, то окно Карты занимает при открытии все рабочее пространство окна MapInfo Professional. И, наоборот, включение в оператор **Map** предложения **Min** приводит к показу окна, свернутого в иконку.

Карта может иметь свою проекцию. Открывая окно Карты, MapInfo Professional использует проекции таблицы первого слоя. Пользователь может изменять проекции Карты при помощи команды **Карта >Единицы измерений**. Для изменения проекции карты в MapBasic используется **Оператор Set Map**.

### Пример:

Откроем окно Карты в 3 дюйма шириной и 2 дюйма высотой с двумя слоями (Косметический слой не считается). Верхний левый угол окна Карты будет ниже на 1 и правее на 1 дюйм от верхнего левого угла окна MapInfo. В окне Карты будет 2 слоя.

```
Open Table "world"
Open Table "cust1994" As customers
Map from customers, world
    Position (1,1) Width 3 Height 2
```

См. также:

Оператор Add Map, Оператор Remove Map, Оператор Set Map, Оператор Set Shade, Оператор Shade

Функция Map3DInfo( )

Назначение

Возвращает свойства окна 3DКарты.

Синтаксис

Map3DInfo( window\_id, attribute )

window\_id – идентификатор окна.

attribute это целое, определяющее, какого типа информация будет возвращена.

Возвращаемая величина

Вещественное (Float), логическое (Logical), или строка символов (String), в зависимости от параметра атрибута.

Описание

Функция Map3DInfo( ) возвращает информацию об окне 3DКарты.

Параметр window\_id определяет какое окно 3DКарты опрашивается. Чтобы получить идентификатор окна, вызывается Функция FrontWindow( ) сразу же после открытия окна или вызывается Функция WindowID( ) в любой момент после создания окна.

Есть несколько числовых атрибутов, которые Map3DInfo( ) может вернуть для любого окна 3DКарты. Параметр attribute сообщает функции Map3DInfo( ) какие данные об окне Карты надо возвратить. Параметр атрибута должен быть одним из кодов, представленных в следующей таблице; коды определены в файле MAPBASIC.DEF.

| Атрибут                 | Возвращаемая величина  |
|-------------------------|--|
| MAP3D_INFO_SCALE        | Вещественное, масштабный фактор 3DКарты.   |
| MAP3D_INFO_RESOLUTION_X | Целое, разрешение по оси X грида в окне 3DКарты.                                     |
| MAP3D_INFO_RESOLUTION_Y | Целое, разрешение по оси Y грида в окне 3DКарты.                                     |
| MAP3D_INFO_BACKGROUND   | Целое, цвет фона, см. функцию RGB.   |
| MAP3D_INFO_UNITS        | Строка, представляющая сокращение единиц измерения площади, например, "mi" для миль. |
| MAP3D_INFO_LIGHT_X      | Вещественное, координата X источника света.  |
| MAP3D_INFO_LIGHT_Y      | Вещественное, координата Y источника освещения.                                      |

| Атрибут                     | Возвращаемая величина  |
|-----------------------------|--|
| MAP3D_INFO_LIGHT_Z          | Вещественное, координата Z источника освещения.                                  |
| MAP3D_INFO_LIGHT_COLOR      | Целое, цвет источника света (см. раздел: " <b>Функция RGB( ) on page 536</b> "). |
| MAP3D_INFO_CAMERA_X         | Вещественное, координата X камеры.   |
| MAP3D_INFO_CAMERA_Y         | Вещественное, координата Y камеры.   |
| MAP3D_INFO_CAMERA_Z         | Вещественное, координата Z камеры.   |
| MAP3D_INFO_CAMERA_FOCAL_X   | Вещественное, координата X точки фокуса камеры.                                  |
| MAP3D_INFO_CAMERA_FOCAL_Y   | Вещественное, координата Y точки фокуса камеры.                                  |
| MAP3D_INFO_CAMERA_FOCAL_Z   | Вещественное, координата Z фокальной точки камеры.                               |
| MAP3D_INFO_CAMERA_VU_1      | Вещественное, первое значение параметра нормали вектора камеры.                  |
| MAP3D_INFO_CAMERA_VU_2      | Вещественное, второе значение параметра нормали вектора камеры.                  |
| MAP3D_INFO_CAMERA_VU_3      | Вещественное, третье значение параметра нормали вектора камеры.                  |
| MAP3D_INFO_CAMERA_VPN_1     | Вещественное, первое значение нормального вектора плоскости просмотра.           |
| MAP3D_INFO_CAMERA_VPN_2     | Вещественное, второе значение параметра нормали плоскости наблюдения.            |
| MAP3D_INFO_CAMERA_VPN_3     | Вещественное, третье значение параметра нормали плоскости наблюдения.            |
| MAP3D_INFO_CAMERA_CLIP_NEAR | Вещественное, ближняя плоскость кадра камеры.                                    |
| MAP3D_INFO_CAMERA_CLIP_FAR  | Вещественное, дальняя плоскость кадра камеры.                                    |

### Пример:

Распечатка текущих значений переменных, определяющих показ окна 3DКарты:

```
include "Mapbasic.def"
Print "MAP3D_INFO_SCALE: " + Map3DInfo(FrontWindow( ), MAP3D_INFO_SCALE)
Print "MAP3D_INFO_RESOLUTION_X: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_RESOLUTION_X)
Print "MAP3D_INFO_RESOLUTION_Y: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_RESOLUTION_Y)
Print "MAP3D_INFO_BACKGROUND: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_BACKGROUND)
Print "MAP3D_INFO_UNITS: " + Map3DInfo(FrontWindow( ), MAP3D_INFO_UNITS)
Print "MAP3D_INFO_LIGHT_X : " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_LIGHT_X )
Print "MAP3D_INFO_LIGHT_Y : " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_LIGHT_Y )
Print "MAP3D_INFO_LIGHT_Z: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_LIGHT_Z)
Print "MAP3D_INFO_LIGHT_COLOR: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_LIGHT_COLOR)
Print "MAP3D_INFO_CAMERA_X: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_X)
Print "MAP3D_INFO_CAMERA_Y : " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_Y )
Print "MAP3D_INFO_CAMERA_Z : " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_Z )
Print "MAP3D_INFO_CAMERA_FOCAL_X: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_FOCAL_X)
Print "MAP3D_INFO_CAMERA_FOCAL_Y: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_FOCAL_Y)
Print "MAP3D_INFO_CAMERA_FOCAL_Z: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_FOCAL_Z)
Print "MAP3D_INFO_CAMERA_VU_1: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_VU_1)
Print "MAP3D_INFO_CAMERA_VU_2: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_VU_2)
Print "MAP3D_INFO_CAMERA_VU_3: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_VU_3)
Print "MAP3D_INFO_CAMERA_VPN_1: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_VPN_1)
Print "MAP3D_INFO_CAMERA_VPN_2: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_VPN_2)
Print "MAP3D_INFO_CAMERA_VPN_3: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_VPN_3)
Print "MAP3D_INFO_CAMERA_CLIP_NEAR: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_CLIP_NEAR)
Print "MAP3D_INFO_CAMERA_CLIP_FAR: " + Map3DInfo(FrontWindow( ),
MAP3D_INFO_CAMERA_CLIP_FAR)
```

**См. также:**

**Оператор Create Map3D, Оператор Set Map3D**



## Функция MapperInfo( )

### Назначение

Возвращает информацию о координатах или расстояниях в окне Карты.

### Синтаксис

**MapperInfo**( *window\_id*, *attribute* )

*window\_id* – идентификатор окна.

*attribute* это целое, определяющее, какого типа информация будет возвращена. См. значения в таблице ниже.

### Возвращаемая величина

Вещественное (Float), логическое (Logical), или строка символов (String), в зависимости от параметра атрибута.

### Описание

Функция **MapperInfo( )** возвращает информацию об окне Карты.

Параметр *window\_id* задает идентификатор окна Карты. Чтобы получить идентификатор окна, вызывается **Функция FrontWindow( )** сразу же после открытия окна или вызывается **Функция WindowID( )** в любой момент после создания окна.

Есть несколько числовых атрибутов, которые **MapperInfo( )** может вернуть для любого окна Карты. Параметр *attribute* сообщает функции **MapperInfo( )** какие данные об окне Карты надо вернуть. Параметр *атрибута* должен быть одним из кодов, представленных в следующей таблице; коды определены в файле MAPBASIC.DEF.

| Параметры               | MapperInfo( ) возвращает   |
|-------------------------|--|
| MAPPER_INFO_AREAUNITS   | Строковая величина с именем единицы измерения площади (например, "sq mi" – квадратные мили).   |
| MAPPER_INFO_CENTERX     | Х-координата центральной точки окна.   |
| MAPPER_INFO_CENTERY     | Y-координата центральной точки окна.   |
| MAPPER_INFO_CLIP_REGION | Возвращает строку, определяющую, используется ли регион для отсечения части карты. Возвращает "on", если применяется регион отсечения. В других случаях, возвращает "off". |

| Параметры                               | MapperInfo( ) возвращает   |
|---|--|
| MAPPER_INFO_CLIP_TYPE                   | <p>Тип отсечения, применяемый к карте. Варианты включают:</p> <ul style="list-style-type: none"> <li>MAPPER_INFO_CLIP_DISPLAY_ALL</li> <li>MAPPER_INFO_CLIP_DISPLAY_POLYOBJ</li> <li>MAPPER_INFO_CLIP_OVERLAY</li> </ul>   |
| MAPPER_INFO_COORDSYS_CLAUSE             | Строка, <b>Оператор CoordSys</b> для этого окна.   |
| MAPPER_INFO_COORDSYS_CLAUSE_WITH_BOUNDS | Строка, <b>Оператор CoordSys</b> для этого окна, включая ограничивающий данную систему координат прямоугольник.  |
| MAPPER_INFO_COORDSYS_NAME               | Строка с именем координатной системы карты такая, как она обозначена в файле MAPINFOW.PRJ (но без суффикса "r...", который можно видеть в файле MAPINFOW.PRJ). Возвращает пустую строку, если значение CoordSys не найдено в файле MAPINFOW.PRJ.   |
| MAPPER_INFO_DISPLAY                     | <p>Целое число типа SmallInt, соответствующее типу информации, которая показывается в строке сообщений окна Карты. См. также раздел <b>Set Map Display</b>. Результатом может быть один из следующих кодов:</p> <ul style="list-style-type: none"> <li>MAPPER_INFO_DISPLAY_SCALE</li> <li>MAPPER_INFO_DISPLAY_ZOOM</li> <li>MAPPER_INFO_DISPLAY_POSITION</li> </ul>                  |
| MAPPER_INFO_DISPLAY_DMS                 | <p>Короткое целое, указывающее, в каких единицах показываются градусы на Карте - десятичные, DMS (градусы, минуты, секунды) или в формате Military Grid Reference System. Возвращает значения:</p> <ul style="list-style-type: none"> <li>MAPPER_INFO_DISPLAY_DECIMAL</li> <li>MAPPER_INFO_DISPLAY_DMS</li> <li>MAPPER_INFO_DISPLAY_MGRS (Military Grid Reference System)</li> </ul> |
| MAPPER_INFO_DIST_CALC_TYPE              | <p>Короткое целое, указывающее тип алгоритма вычисления расстояния, длины, периметра и площади. См. также раздел <b>Set Map Distance Type</b>. Возвращает значения:</p> <ul style="list-style-type: none"> <li>MAPPER_INFO_DIST_SPHERICAL</li> <li>MAPPER_INFO_DIST_CARTESIAN</li> </ul>   |

| Параметры                            | MapperInfo( ) возвращает  |
|--------------------------------------|---|
| MAPPER_INFO_DISTUNITS                | Имя единицы измерения расстояния (например, "mi").  |
| MAPPER_INFO_EDIT_LAYER               | Целое число типа SmallInt, являющееся номером изменяемого слоя. Ноль (0), если объекты изменяются в Косметическом слое; единица (1), если изменяемый слой первый некосметический, и т. д. Если результатом будет минус единица, то ни один слой не находится в изменяемом состоянии.  |
| MAPPER_INFO_LAYERS                   | Число слоев на Карте, включая Косметический (число типа SmallInt).  |
| MAPPER_INFO_MAXX                     | Максимальная X-координата части Карты, показанной в окне.   |
| MAPPER_INFO_MAXY                     | Максимальная Y-координата части Карты, показанной в окне.   |
| MAPPER_INFO_MERGE_MAP                | Строковое значение: строка, представляющая собой последовательность операторов MapBasic, которая может потребоваться для включения окна Карты в текущее окно Карты."  |
| MAPPER_INFO_MINX                     | Минимальная X-координата части Карты, показанной в окне.  |
| MAPPER_INFO_MINY                     | Минимальная Y-координата части Карты, показанной в окне.  |
| MAPPER_INFO_MOVE_DUPLICATE_NO<br>DES | Короткое целое, указывающее надо ли удалять дублирующиеся узлы в режиме Форма окна Карты. Если значение равно 0, дублирующиеся узлы не удаляются. Если значение равно 1, все дублирующиеся узлы с одного слоя удаляются. Чтобы восстановить стандартные режимы окна, выполните оператор <b>Set Map Move Nodes Default</b> . |
| MAPPER_INFO_NUM_THEMATIC             | Короткое целое число, номер слоя, который является тематическим.  |

| Параметры                | MapperInfo( ) возвращает   |
|--------------------------|--|
| MAPPER_INFO_REPROJECTION | <p>Строковое значение, указывающее текущий режим перепроектирования. Возможны следующие значения:</p> <ul style="list-style-type: none"> <li>None - никогда не перепроектировать карту.</li> <li>Always - всегда перепроектировать карту.</li> <li>Auto - перепроектировать карту или нет, будет решать Mapinfo Professional.</li> </ul> |
| MAPPER_INFO_RESAMPLING   | <p>Сроковое значение указывающее метод расчёта значений пикселей исходного растра при перепроектировании. Возможны следующие значения:</p> <ul style="list-style-type: none"> <li>Кубическая свертка</li> <li>Ближайшее соседство</li> </ul>   |
| MAPPER_INFO_SCALE        | <p>Текущий масштаб окна Карты, заданный количеством текущих единиц измерения (например, миль) в одной текущей "бумажной" единице (например, в дюйме). Результат возвращается в терминах текущих единиц измерения расстояний MapBasic.</p>  |
| MAPPER_INFO_SCROLLBARS   | <p>Логическая величина, показывающая, есть ли в окне Карты полосы прокрутки.</p>   |
| MAPPER_INFO_XYUNITS      | <p>Строковая величина с именем координатной единицы (например, "degree").</p>  |
| MAPPER_INFO_ZOOM         | <p>Размер показанной части Карты (расстояние от Западного до Восточного края) в единицах измерения расстояния, установленных в MapBasic (смотрите раздел <b>Оператор Set Distance Units on page 633</b>).</p>  |

Когда **MapperInfo( )** вызывается для получения значений координат (с указанием MAPPER\_INFO\_CENTERX в качестве attribute), возвращаемое значение будет координатами в текущей для MapBasic системе координат, которая может отличаться от системы координат в окне Карты. Используйте **Оператор Set CoordSys** для задания другой системы координат.

Настройки для окна Карты и обеспечение поддержки MapBasic можно делать для каждого окна Карты.  
Вы можете задавать способы совмещения узлов объектов при их перемещении в режиме Форма.

Когда создано новое окно Карты, можно настроить режим Совмещения при перемещении (Настройки > Режимы > Окно Карты > Совмещать при перемещении).

Существующее окно Карты может быть опрошено на предмет параметров режима Совмещения при перемещении использованием новых атрибутов функции **MapperInfo( )**.

Для изменения текущих настроек используется **Оператор Set Map**.

### Информация о возвращаемых координатах

**MapperInfo( )** не возвращает координат (например, MINX, MAXX, MINY, MAXY) в единицах измерения, заданных на карте. Такие значения возвращаются либо в единицах внутренней координатной системы MapInfo Professional, действующей в рамках данного сеанса, либо в терминах координатной системы программы MapBasic, вызывающей данную функцию. Также, задание атрибута MAPPER\_INFO\_XYUNITS возвращает единицы, которые используются для показа положения указателя мыши на экране в строке состояний (устанавливается оператором **Set Map Window Frontwindow**).

### Информация о врезке региона

Начиная с MapInfo Professional 6.0, существуют 3 метода, применяющиеся для создания врезки. Метод MAPPER\_INFO\_CLIP\_OVERLAY был единственным до версии MapInfo Professional 6.0. При использовании этого метода автоматически используется **Функция Overlap( )** (аналог команды **Объект > Удалить внешнюю часть**). Так как **Функция Overlap( )** не затрагивает текстовые объекты, тексты этим методом не обрезаются. Точечные объекты обрезаются только в том случае если они целиком выходят за область врезки. Подписи обрабатываются аналогично: они либо показываются, если подпись целиком попадает в область врезки, либо отбрасываются. Стилизованные элементы (линии с заданной шириной, символы, тексты и пр.) при использовании этого метода не обрезаются.

Метод MAPPER\_INFO\_DISPLAY\_ALL использует для обрезания изображения системную функцию Windows Display. Этот метод обрезает все типы объектов. Тематика, растры и сетки (гриды) тоже обрезаются. Стили (ширина линий, символы, текст) -тоже. Это стандартный метод обрезания.

Метод MAPPER\_INFO\_CLIP\_DISPLAY\_POLYOBJ есть сочетание системной функции Windows Display и функциональности метода MAPPER\_INFO\_CLIP\_OVERLAY. Функция Windows Display Clipping применяется к полигональным и полилинейным объектам и к объектам, которые могут быть преобразованы в таковые (прямоугольники, скругленные прямоугольники, эллипсы, дуги). Вся стилизация этих объектов и связанные с ними символы также обрезаются. Точечные объекты, подписи и тексты обрабатываются также, как и в методе MAPPER\_INFO\_CLIP\_OVERLAY. Этот комбинированный метод часто дает лучшие результаты, чем метод MAPPER\_INFO\_CLIP\_OVERLAY.

### Ошибки:

ERR\_BAD\_WINDOW, если нет такого окна;

ERR\_FCN\_ARG\_RANGE, если значение аргумента выходит за пределы, заданные при его определении.

ERR\_WANT\_MAPPER\_WIN, если окно не Карта.

См. также:

Функция **LayerInfo( )**, Оператор **Set Distance Units**, Оператор **Set Map**

---

## Функция **Maximum( )**

### Назначение

Возвращает наибольшее из двух заданных чисел.

### Синтаксис

**Maximum**( *num\_expr*, *num\_expr* )

*num\_expr* – числовое выражение

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Maximum( )** возвращает наибольшее из двух чисел, заданных численными выражениями.

**Пример:**

```
Dim x, y, z As Float
x = 42
y = 27
z = Maximum(x, y) ' z равно 42
```

**См. также:**

**Функция Minimum( )**

---

## Функция MBR( )

**Назначение**

Возвращает прямоугольный объект, представляющий минимальное прямоугольное покрытие заданного объекта.

**Синтаксис**

```
MBR( obj_expr )
```

*obj\_expr* - выражение, определяющее объект.

**Возвращаемая величина**

Величина типа Object. Графический объект типа "прямоугольник".

**Описание**

Функция **MBR( )** возвращает графический объект – наименьший прямоугольник, в который можно вписать объект, заданный выражением *obj\_expr*.

Такой прямоугольник представляет наименьший прямоугольник, в который покрывает объект целиком. Например, минимальное прямоугольное покрытие США представляет собой прямоугольник, у которого правая сторона включает в себя самую западную точку границы штата Мен, нижняя сторона – самую южную точку границы Гавайи, и левая и верхняя стороны – самую восточную и самую северную точки границы штата Аляска.

Минимальное прямоугольное покрытие точечного объекта имеет нулевую ширину и нулевую высоту.

**Пример:**

```
Dim o_mbr As Object
Open Table "world"
Fetch First From world
o_mbr = MBR(world.obj)
```

**См. также:**

**Функция Centroid( ), Функция CentroidX( ), Функция CentroidY( )**

## Оператор Menu Bar

### Назначение

Показывает или скрывает строку меню.

### Синтаксис

```
Menu Bar { Hide | Show }
```

### Описание

Оператор **Menu Bar** управляет отображением строки меню в рабочем окне MapInfo. Программа, используя этот оператор, может освободить больше места на экране для окна Карты, Списка, Отчета или Графика.

Чтобы вновь показать строку меню, скрытую оператором **Menu Bar Hide**, используйте оператор **Menu Bar Show**. Оператор **Menu Bar Hide** следует использовать аккуратно, так как пользователь может быть поставлен в тупик, оказавшись без строки меню. За каждым оператором **Menu Bar Hide** по возможности, должен следовать оператор **Menu Bar Show**.

Пока строка меню скрыта, MapInfo будет игнорировать клавишные сокращения для вызова команд. Т.е., например, для вызова диалога команды Файл > Открыть Вы можете использовать клавиши CTRL+O, но, если строка меню скрыта, то нажатие на эти клавиши ни к чему не приведет.

### См. также:

[Оператор Alter Menu Bar](#), [Оператор Create Menu Bar](#)

---

## Функция MenuItemInfoByHandler( )

### Назначение

Возвращает информацию об элементе меню MapInfo Professional.

### Синтаксис

```
MenuItemInfoByHandler( handler, attribute )
```

*handler* – либо строка с именем процедуры-обработчика, заданной для элемента меню предложением **Calling**, либо целое число (тип Integer), код, который был задан в предложении **Calling**.

*attribute* – целое число типа Integer, код, задающий, какая информация необходима в результате.



Описание

Параметр handler может быть как строковым, так и численным. Если Вы выбрали строковый вид (имя процедуры), а при этом соответствующую процедуру вызывают два или более элемента меню, то MapInfo будет возвратит идентификатор первого элемента меню, вызвавшего эту процедуру. Поэтому, если Вам необходима информация о другом элементе, то используйте для идентификации ID-номер, который был назначен элементу меню (см. **Оператор Create Menu**), а также используйте **Функция MenuItemInfoByID( )** вместо **MenuItemInfoByHandler( )**.

В следующей таблице в первой колонке приводятся имена кодов, установленных в файле стандартных определений MapBasic MAPBASIC.DEF:

| Параметры                  | Возвращаемое значение   |
|----------------------------|---|
| MENUIITEM_INFO_ACCELERATOR | Строка, величина типа String: строковый код акселератора элемента меню (например, "W^Z" или "W#%119") или пустая строка, акселератор не был назначен. Информацию о назначении элементу меню акселератора смотрите в разделе <b>Оператор Create Menu on page 207</b> .   |
| MENUIITEM_INFO_CHECKABLE   | Логическая величина: "Да" (TRUE), если элемент меню фиксируется (рядом с именем элемента в меню может появляться галочка)   |
| MENUIITEM_INFO_CHECKED     | Логическая величина: "Да" (TRUE), если элемент меню можно фиксировать и в данный момент он фиксирован (есть галочка); "Да" (TRUE) также, если элемент меню имеет несколько вариантов текста (например, <b>Показать...</b> и <b>Скрыть...</b> ), и при этом элемент меню находится в состоянии "Показать". "Нет" (FALSE) во всех остальных случаях.  |
| MENUIITEM_INFO_ENABLED     | Логическая величина: "Да" (TRUE), если элемент меню активен.  |
| MENUIITEM_INFO_HANDLER     | Целое число типа Integer, номер обработчика элемента меню. Если при создании элемента меню в предложении <b>Calling</b> был задан код (например, <code>Calling M_FILE_SAVE</code> ), то результатом будет соответствующая константа. Если предложение <b>Calling</b> или имя процедуры, то результатом будет уникальное целое число, которое может быть принимать в качестве аргумента функция <b>MenuItemInfoByHandler( )</b> или <b>Оператор Run Menu Command</b> . |

## Функция MenuItemInfoById ( )

| Параметры                  | Возвращаемое значение   |
|----------------------------|---|
| MENUITEM_INFO_HELPMSG      | Строка, величина типа String: подсказка для элемента меню, которая была назначена в предложении <b>HelpMsg</b> (см. <b>Оператор Create Menu</b> ) или пустая строка, если подсказка не назначалась.   |
| MENUITEM_INFO_ID           | Целое число типа Integer: идентификатор элемента меню, который был назначен предложением <b>ID</b> , см. раздел <b>Оператор Create Menu</b> , или 0, если элемент меню не имеет идентификатора.   |
| MENUITEM_INFO_SHOWHIDEABLE | Логическая величина: “Да” (TRUE), если элемент меню имеет несколько вариантов текста (например, <b>Показать...</b> и <b>Скрыть...</b> ). Несколько вариантов текста задаются помещением символа “!” в начало строки описания элемента меню ( <b>Оператор Create Menu</b> или <b>Оператор Alter Menu</b> ) и символа (^) перед началом альтернативного текста. |
| MENUITEM_INFO_TEXT         | Строка, величина типа String; полный текст, используемый при создании элемента меню (например, когда выполняется <b>Оператор Create Menu</b> ).   |

См. также:

**Функция MenuItemInfoById ( )**

## Функция MenuItemInfoById ( )

### Назначение

Возвращает информацию об элементе меню MapInfo Professional.

### Синтаксис

**MenuItemInfoById** (*menuitem\_ID*, *attribute*)

*menuitem\_ID* – целое число типа Integer, идентификатор элемента меню, который он получил при создании в предложении ID оператора Create Menu;

*attribute* – целое число типа Integer, код, задающий, какую информацию необходимо вернуть.

### Описание

Функция работает так же, как и **Функция MenuItemInfoByHandler ( )**, за исключением того, как задается элемент меню в первом параметре.

Эта функция запрашивает элемент меню по его идентификатору. **Функция MenuitemInfoByHandler( )** запрашивает элемент меню по его обработчику, который запускается, когда пользователь выберет этот элемент меню.

Параметр *attribute* должен быть целочисленным кодом, одним из тех, имена которым присвоены в файле MAPBASIC.DEF (например, MENUITEM\_INFO\_CHECKED). Список возможных значений параметра *attribute* и какой возвращаемые результаты см. в разделе **Функция MenuitemInfoByHandler( ) на стр. 84.**

См. также:

**Функция MenuitemInfoByHandler( )**

## Оператор Metadata

### Назначение

Управление метаданными таблицы.

### Синтаксис 1

```
Metadata Table table_name
{ SetKey key_name To key_value |
  DropKey key_name [ Hierarchical ] |
  SetTraverse starting_key_name [ Hierarchical ]
  Into ID traverse_ID_var }
```

*table\_name* – имя открытой таблицы;

*key\_name* – строковая величина, представляющая собой имя ключа метаданных. Она должна начинаться с обратного слэша (“\”) и не должна заканчиваться обратным слэшем.

*key\_value* – строка до 239 символов длиной, значение, присваиваемое ключу;

*starting\_key\_name* – строка, представляющая первое имя ключа для извлечения соответствующего значения из таблицы. Чтобы начать структурное извлечение с самого начала списка ключей, добавьте “\” (обратный слэш);

*traverse\_ID\_var* – имя переменной типа Integer. С помощью этой переменной MapInfo управляет последовательными операторами **Metadata Traverse**.

### Синтаксис 2

```
Metadata Traverse traverse_ID
{ Next Into Key key_name_var In key_value_var |
  Destroy }
```

*traverse\_ID* – целое число типа Integer (такое как значение переменной *traverse\_ID\_var* из предыдущего варианта синтаксиса оператора);

*key\_name\_var* – имя строковой переменной (MapInfo помещает в эту переменную название ключа для извлечения);

*key\_value\_var* – имя строковой переменной (MapInfo помещает в эту переменную значение ключа для извлечения);

### Описание

Оператор Metadata управляет метаданными, размещаемыми в таблице MapInfo. Метаданные – это информация, размещаемая в файле таблицы (TAB), а не в строках и столбцах файла данных.

Каждая таблица может иметь ключи метаданных - от нуля и выше. Каждый ключ определяет категорию хранимой информации, такой как имя автора, его права и т.д. Каждому ключу соответствует некое строчное значение. Например, ключу “\Copyright” может соответствовать значение “Copyright 1995 MapInfo Corporation.” Более подробно метаданные описываются в *Руководстве пользователя MapBasic*.

### Изменение метаданных в таблице

Чтобы создать, поменять или удалить метаданные, используйте Синтаксис 1. При этом используются следующие предложения:

#### SetKey

Присваивает значение ключу. Если ключ уже существует, то MapInfo присваивает ему новое значение. Если ключ не существует, MapInfo создает его. Если Вы создаете метаданные, то они записываются немедленно; не нужно проводить операцию сохранения. Пример:

```
MetaData Table Parcels SetKey "\Info\Date" To Str$(CurDate( ))
```

**Внимание:** MapInfo автоматически создает ключ метаданных “\IsReadOnly” (со стандартным значением “FALSE”) в первый раз, когда Вы добавляете метаданные в таблицу. Ключ “\IsReadOnly” – это специальный ключ, который MapInfo использует для своих нужд.

#### DropKey

Удаляет ключ из таблицы. Если добавить слово **Hierarchical**, MapInfo удаляет вместе с ключом всю структуру метаданных, подчиненных этому ключу. Например, если в таблице есть ключи “\Info\Author” и “\Info\Date”, то оба они удаляются следующим оператором:

```
MetaData Table Parcels DropKey "\Info" Hierarchical
```

### Чтение метаданных из таблицы

Чтобы прочитать метаданные из таблицы, используется предложение **SetTraverse**, которое инициализирует операцию структурного извлечения, продолжаемую затем одноразовыми извлечениями значений предложением **Next**. Для завершения операции извлечения применяется предложение **Destroy**, которое освобождает память, используемую при этой операции. При этом используются следующие предложения:

#### SetTraverse

Подготавливает операцию структурного извлечения метаданных, начиная с определенного ключа. Чтобы начать извлечение с самого первого в иерархии ключа, задайте в качестве имени начального ключа “\”. Если будет добавлено слово **Hierarchical**, то операция извлечения пройдет по всей структуре. Если слово **Hierarchical** опущено, то операция извлечения не будет опускаться на уровни ниже начального (т.е. извлечение, стартовавшее с ключа “\Info”, не затронет ключ “\Info\Date”).

**Next Into Key... Into Value...**

Попытка чтения следующего ключа. Если еще остаются ключи для чтения, то MapInfo помещает ключ в переменную *key\_name\_var* и значение, соответствующее ключу, в переменную *key\_value\_var*. Если ключи исчерпаны, MapInfo помещает в обе переменные пустые значения.

**Destroy**

Завершает операцию структурного извлечения и освобождает занятую под нее память.

**Внимание:** Операция структурного извлечения может проникать до десятого уровня иерархии (т.е. “\Один\Два\Три\Четыре\Пять\Шесть\Семь\Восемь\Девять\Десять”), начиная с нулевого или корневого (“\”). Если Вы хотите проникнуть глубже, чем на 10 уровней, то начинайте операцию структурного извлечения с ненулевого уровня (например, с уровня “\Один\Два\Три\Четыре\Пять”).

### Пример:

Следующая процедура извлекает все метаданные из таблицы; имя таблицы определяется вызывающей программой. Все ключи и соответствующие им значения распечатываются в окне Сообщения.

```
Sub Print_Metadata(ByVal table_name As String) Dim i_traversal As Integer
Dim s_keyname, s_keyvalue As String ' Инициализация операции извлечения:
Metadata Table table_name SetTraverse "\" Hierarchical Into ID
i_traversal ' Попытка извлечь значение по первому ключу: Metadata
Traverse i_traversal Next Into Key s_keyname Into Value s_keyvalue '
Теперь в цикле извлекаются все значения по одному ' до тех пор, пока есть
непустые ключи, ' и они распечатываются в окне Сообщений.
Do While s_keyname <> "" Print "Print "Key name: " & s_keyname Print
"Key value: " & s_keyvalueMetadata Traverse i_traversal Next Into Key
s_keyname Into Value s_keyvalue Loop ' Освобождение памяти, занятой под
операцию извлечения: MetaData Traverse i_traversal Destroy End Sub
```

### См. также:

[Функция GetMetadata\\$\( \)](#), [Функция TableInfo\( \)](#)

---

## Функция MGRSToPoint( )

### Назначение

Преобразует строку с координатами в формате MGRS (Military Grid Reference System) в точечные объекты в текущей координатной системе MapBasic.

### Синтаксис

MGRSToPoint( string )

*string* – строку с координатами в формате MGRS.

Как начальное значение используется стандартная система Долгота/Широта.

### Возвращаемая величина

Объект

### Описание

Возвращает точки текущей координатной системе MapBasic, которая по умолчанию есть Долгота/Широта (без референц-эллипсоида). Чтобы добиться максимальной точности, задайте координатную систему MapBasic такой же, как и в исходной таблице до вызова **MGRSToPoint( )**. Иначе MapInfo Professional проведет ненужные преобразования координатных систем, что негативно отразится на точности.

### Пример:

Пример 1:

```

dim obj1 as Object
dim s_mgrs As String
dim obj2 as Object
obj1 = CreatePoint(-74.669, 43.263)
s_mgrs = PointToMGRS$(obj1)
obj2 = MGRSToPoint(s_mgrs)

```

**Пример 2:**

```

Open Table "C:\Temp\MyTable.TAB" as MGRSfile
' При использовании функций PointToMGRS$( ) или MGRSToPoint( ) ' важно
удостовериться в том, что текущая система ' координат MapBasic
соответствует координатной системе ' таблицы, хранящей точечные объекты.
'Установим в MapBasic координатную систему таблицыSet CoordSys Table
MGRSfile
'Поместим в символьную колонку (например, в Col2) MGRS-строки из 'из
таблицы точек
Update MGRSfile
    Set Col2 = PointToMGRS$(obj)
'Поместим в колонки для действительных чисел (Col3 и Col4) 'значения
CentroidX и CentroidY 'из символьной колонки (Col2), содержащей строки в
формате MGRS.
Update MGRSfile
    Set Col3 = CentroidX(MGRSToPoint(Col2))
Update mgrstestfile ' MGRSfile
    Set Col4 = CentroidY(MGRSToPoint(Col2))
Commit Table MGRSfile
Close Table MGRSfile

```

**См. также:**

**Функция PointToMGRS\$( )**

---

## Функция Mid\$( )

**Назначение**

Возвращает строку, извлекая ее из середины другой.

**Синтаксис**

**Mid\$( *string\_expr*, *position*, *length* )**

*string\_expr* - выражение, результат которого есть строка.

*position* – целочисленное выражение, результат которого есть номер первого символа, извлекаемого из строки;

*length* – длина извлекаемой подстроки, целое число.

**Возвращаемая величина**

Строка

### Описание

Функция **Mid\$( )** возвращает подстроку из строки, заданной выражением *string\_expr*.

**Mid\$( )** копирует length символов из *string\_expr*, начиная с символа, номер которого определен параметром position. Если значение этого параметра меньше или равно единице, то копирование будет производиться с самого начала строки *string\_expr*.

Если длина строки *string\_expr* меньше, чем заданный параметр length, то функция **Mid\$( )** вернет укороченную строку. Если параметр position задает позицию за пределами строки *string\_expr*, то функция **Mid\$( )** возвратит пустую строку. Если параметр length меньше единицы, функция **Mid\$( )** также возвратит пустую строку.. Все дробные параметры MapBasic округляет до ближайших целых.

### Пример:

```
Dim str_var, substr_var As Stringstr_var = "New York City"substr_var =  
Mid$(str_var, 10, 4)' substr_var теперь равна "City"
```

### См. также:

[Функция InStr\( \)](#), [Функция Left\\$\( \)](#), [Функция Right\\$\( \)](#)

---

## Функция MidByte\$( )

### Назначение

Позволяет извлекать байты из строки, состоящей из двухбайтовых символов (например, Windows Japanese).

### Синтаксис

**MidByte\$(** *string\_expr*, *position*, *length* **)**

*string\_expr* - выражение, результат которого есть строка.

*position* – целочисленное выражение, результат которого есть номер первого символа, извлекаемого из строки;

*length* – целочисленное выражение, задающее количество извлекаемых байт.

### Возвращаемая величина

Строка

### Описание

Функция **MidByte\$( )** возвращает заданный байт из строки.

Функция **MidByte\$( )** используется для извлечения нескольких байтов из строки, образованной символами двухбайтовой кодировки. Например, в японской версии Microsoft Windows используется двухбайтовая система.

В системах с однобайтовыми наборами символов функция **MidByte\$( )** возвращает те же результаты, что и [Функция Mid\\$\( \)](#).



См. также:

Функция **InStr( )**, Функция **Left\$( )**, Функция **Right\$( )**

---

## Функция **Minimum( )**

### Назначение

Возвращает наименьшее из двух заданных чисел.

### Синтаксис

**Minimum**( *num\_expr*, *num\_expr* )

*num\_expr* – числовое выражение

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Minimum( )** возвращает наименьшее из двух чисел, заданных численными выражениями.

### Пример:

```
Dim x, y, z As Float
x = 42
y = -100
z = Minimum(x, y) ' z равно -100
```

См. также:

Функция **Maximum( )**

---

## Функция **Minute**

### Назначение

Возвращает минуты.

### Синтаксис

**Minute** (Time)

### Возвращаемая величина

Номер

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim X as time
dim iMin as integer
```

## Функция Month( )

---

```
X = CurDateTime()  
iMin = Minute(X)  
Print iMin
```

## Функция Month( )

---

### Назначение

Возвращает из даты компоненту, соответствующую номеру месяца в году (1 – 12).

### Синтаксис

**Month**( *date\_expr* )

*date\_expr* – выражение, результат которого есть дата (величина типа Data).

### Возвращаемая величина

Короткое целое число от 1 до 12, включительно. Величина типа SmallInt.

### Описание

Функция **Month( )** возвращает целое число, являющееся номером месяца в дате.

### Примеры

Определим, какой сейчас месяц с помощью функции **Month( )**:

```
If Month(CurDate( )) = 12 Then'' ... это декабрь...'End If
```

Функцию **Month( )** можно использовать в SQL-запросе. В примере подразумевается что в таблица ORDERS есть колонка с данными типа Date, названная "Order\_Date". Предложение Where оператора Select предписывает MapInfo Professional выбирать заказы только за декабрь 1993.

```
Open Table "orders"Select * From orders Where Month(orderdate) = 12 And  
Year(orderdate) = 1993
```

### См. также:

[Функция CurDate\( \)](#), [Функция Day\( \)](#), [Функция Weekday\( \)](#), [Функция Year\( \)](#)

---

## Оператор Nearest

### Назначение

Ищет объект в таблице, который находится ближе других объектов к заданному. Результат выдаётся в виде объекта-полилинии между 2 точек, и представляет самое кратчайшее расстояние между объектами.

## Синтаксис

```

Nearest [ N | All ]
  From { Table fromtable | Variable fromvar }
  To totable Into intotable
  [ Type { Spherical | Cartesian }]
  [ Ignore [ Contains ] [ Min min_value ] [ Max max_value ]
  Units unitname ] [ Data clause ]

```

*N* – параметр (необязательный), представляющий число "ближайших" объектов, которые будут найдены. По умолчанию – 1. Если используется параметр **All**, то объект-расстояние будет создан для каждой комбинации.

*fromtable* – таблица объектов, от которых будут искаться кратчайшие расстояния;

*fromvar* – переменная MapBasic для объекта, от которого ищется кратчайшее расстояние;

*totable* – таблица объектов, до которых будут искаться кратчайшие расстояния;

*intotable* – таблица, в которую будут записаны полученные результаты;

*min\_value* – минимальная допустимая дистанция для результата;

*max\_value* – максимальная допустимая дистанция для результата;

*unitname* – единицы измерения параметров *min\_value* и/или *max\_value* (например, "km").

*clause* – выражение, задающее таблицы *fromtable* и *totable*, из которых извлекаются результаты.

## Описание

Оператор **Nearest** находит все объекты из таблицы *fromtable* которые ближе всего к заданному объекту. Рассматривается каждый объект в таблице *fromtable*. Для каждого объекта в таблице *fromtable* будет найден наиболее близкий объект в таблице *totable*. Если задан параметр *N*, то будет найдено *N* наиболее близких объектов в таблице *totable*. Объект-полилиния, соединяющий наиболее близкие точки между объектом *fromtable* и выбранным объектом в таблице *totable*, помещается в таблицу *intotable*. Если задан параметр **All**, то объект помещается в таблицу *intotable*, представляющую расстояние между объектом *fromtable* каждым объектом таблицы *totable*.

Если имеется несколько объектов в таблице *totable*, которые имеют одинаковое расстояние до заданного объекта в таблице *fromtable*, то в виде результата возвращается только один из них. Если возвращается несколько объектов (задана величина *N*, большая чем 1), то объекты с одинаковым расстоянием будут записаны последовательно. Если существует второй объект с таким же расстоянием, а запрашивается 3 объекта, то искомым объектом станет третий по счёту объект.

Типы объектов в таблицах *fromtable* и *totable* могут быть любыми кроме текстовых. Например, если обе таблицы содержат объекты-регионы, то находится максимальное расстояние между объектами-регионами и создаётся объект-полилиния, соединяющая две точки каждого объекта, использованных для расчёта расстояния. Если объекты-регионы пересекаются, минимальное расстояние будет нулевым, и объект-полилиния будет редуцированным, с двумя точками, имеющими те же самые координаты, как и у точки пересечения.

Расстояние, рассчитанное этой функцией, не принимается во внимание при расчёте маршрутов в транспортных задачах. Это расстояние можно сравнить с "перелётом птицы".

**Type** – метод, используемый для расчёта расстояния между объектами. Он может быть либо сферическим (**Spherical**) , либо декартовым (**Cartesian**). Тип алгоритма вычислений должен соответствовать системе координат результирующей таблицы *intotable*, иначе произойдет ошибка. Если система координат результирующей таблицы *intotable* не Долгота/Широта, а метод расчёта сферический, то будет выдано сообщение об ошибке. Если система координат результирующей таблицы *intotable* – Долгота/Широта, а метод расчёта декартовый, то будет выдано сообщение об ошибке.

Предложение **Ignore** ограничивает возвращаемые расстояния. Любые расстояния, найденные по запросу, которые меньше или равны *min\_value* или больше *max\_value* игнорируются; *min\_value* и *max\_value* являются расстояниями и измеряются в единицах, определенных параметром *unitname*. Если *unitname* не соответствует стандартным единицам измерения, будет выдано сообщение об ошибке. С помощью параметра **Min** можно исключать нулевые расстояния. Это может быть полезно в случае двух таблиц с точками, чтобы не рассчитывать расстояния от одинаковых точек. Например, если две таблицы с точками, представляют города, и мы хотим найти ближайшие города, мы можем исключить из поиска одинаковые города. Предложение **Ignore** не является обязательным, равно как и подпредложения **Min** и **Max**.

С помощью параметра **Max** можно ограничить число объектов, рассматриваемых в *totable*. Это полезно при совместном использовании с параметрами *N* или **All**. Например, пусть надо найти 5 аэропортов, находящихся ближе остальных для нескольких городов (где *fromtable* это набор городов, а *totable* это набор аэропортов), но мы не будем рассматривать аэродромы, удалённые более чем на 100 км. Результат может оказаться меньшим, чем пять аэропортов для какого-либо города. Этот параметр целесообразно использовать совместно с параметром **All**, так можно найти все аэропорты на расстоянии 100 км от города. Применение параметра **Max** может улучшить поведение оператора **Nearest** , поскольку он эффективно ограничивает число искомых объектов *totable*.

Найденные расстояния отфильтровываются так, что они строго больше чем *min\_value* и меньше или равны *max\_value*:

```
min_value < distance <= max_value
```

Можно сделать так, чтобы расстояния возвращались в виде нескольких проходов оператора **Nearest**. Например, первый проход может вернуть все объекты на расстоянии от 0 до 100 км, второй проход может вернуть все объекты на расстоянии от 100 до 200 км, а результаты будут содержать повторяющихся записей (то есть, расстояние 100 встретилось только в первом проходе, а во втором уже не встречается).

Если один объект находится внутри другого, то расстояние между ними считается равным нулю. Например, если таблица *fromtable* это WORLDCAPS, а таблица *totable* - это WORLD, то расстояние между Москвой и Россией будет равно нулю. Если же флаг **Contains** в предложении **Ignore** установлен, то расстояние не будет обязательно нулевым. Вместо этого результатом будет расстояние между Москвой и границей России. Все замкнутые объекты, типа областей (регионов) будут обработаны как полилинии ради этой операции.

## Предложение Data

Предложение **Data** используется для того, чтобы отметить от каких объектов из таблиц *fromtable* и *totable* будут получен результат.

```
Data IntoColumn1=column1, IntoColumn2=column2
```

Переменная *IntoColumn* в левой части уравнения должна корректно задавать колонку в *intotable*. В правой части от знака равенства должны также находиться корректно заданные колонки из таблиц *totable* или *fromtable*. Если одинаковое имя колонки существует в обеих таблицах *totable* и *fromtable*, то будет использоваться колонка из таблицы *totable* (т.е. в таблице *totable* в первую очередь ищутся имена колонок, а это правая часть "уравнения"). Чтобы избежать конфликтов такого типа, имена колонок можно заменять псевдонимами таблиц:

```
Data name1=states.state_name, name2=county.state_name
```

Чтобы заполнить колонку в таблице *intotable* значениями расстояний, нужно выполнить команду **Таблица > Обновить колонку** или использовать **Оператор Update**.

## Примеры

Предположим, имеется таблица с точечными объектами, представляющими местоположение банкоматов, в ней имеется как минимум две колонки: колонка с фирмами, то есть с именами фирм, где есть банкоматы и адресную колонку с почтовым адресом фирмы. Предположим, что текущая выборка представляет наше текущее местоположение. Тогда можно найти точку с ближайшим от нас банкоматом:

```
Nearest From Table selection To atm Into result Data
where=Business,address=Address
```

Если надо найти 5 ближайших банкоматов:

```
Nearest 5 From Table selection To atm Into result Data
where=Business,address=Address
```

Если надо найти все банкоматы в радиусе 5 км:

```
Nearest All From Table selection To atm Into result Ignore Max 5 Units
"mi" Data where=buisness,address=address
```

Предположим, у нас есть таблица с размещением домов (*fromtable*) и таблица с береговой линией (*totable*). Чтобы найти расстояние от данного дома до береговой линии:

```
Nearest From Table customer To coastline Into result Data
who=customer.name,
where=customer.address,coast_loc=coastline.county,type=coastline.designat
ion
```

Исключим клиентов, которые расположены дальше, чем 30 миль:

```
Nearest From Table customer To coastline Into result Ignore Max 30 Units
"mi" Data who=customer.name,
where=customer.address,coast_loc=coastline.county,
type=coastline.designation
```

Предположим, имеется таблица городов (*fromtable*) и другая таблица столиц (*totable*), и надо найти ближайшую столицу для каждого города, но при этом проигнорировать те случаи, когда город в таблице *fromtable* является также столицей:

```
Nearest From Table uscty_1k To usa_caps Into result Ignore Min 0 Units  
"mi" Data city=uscty_1k.name, capital=usa_caps.capital
```

**См. также:**

Оператор **Farthest**, Функция **CartesianObjectDistance( )**, Функция **ObjectDistance( )**,  
Функция **SphericalObjectDistance( )**, Функция **CartesianConnectObjects( )**,  
**ConnectObjects( )** функция, Функция **SphericalConnectObjects( )**

---

## Оператор Note

### Назначение

Показывает сообщение в простом диалоговом окне.

### Синтаксис

**Note** *message*

*message* – выражение, результат которого будет показан в окне.

### Описание

Оператор **Note** создает простое диалоговое окно для сообщений. В нем есть кнопка **ОК**; сообщение присутствует на экране, пока кнопка **ОК** не будет нажата.

Параметр *message* может быть выражением, не обязательно строковым. Если в результате вычисления выражения *message* получается величина объектного типа (Object), MapBasic автоматически преобразует его в строку, представляющую тип объекта. Если *message* – строка, то максимальная длина должна быть не более 300 символов и может занимать только 6 строк.

### Пример:

```
Note "Всего использовано записей: " + Str$( i_count )
```

**См. также:**

Функция **Ask( )**, Оператор **Dialog**, Оператор **Print**

---

## Функция NumAllWindows( )

### Назначение

Возвращает количество окон, открытых MapInfo Professional, включая специальные окна, такие как инструментальные панели и окно Информации.

**Синтаксис**

**NumAllWindows ( )**

**Возвращаемая величина**

Целое число типа SmallInt.

**Описание**

Функция **NumAllWindows ( )** и возвращает количество окон, открытых MapInfo Professional.

Чтобы определить количество открытых “документальных” окон MapInfo (Карт, Списков, Графиков и Отчетов), используйте функцию **NumWindows ( )**.

**См. также:**

**Функция NumWindows ( ), Функция WindowID ( )**

---

**Функция NumberToDate ( )****Назначение**

Возвращает величину типа Date, созданную из величины типа Integer.

**Синтаксис**

**NumberToDate ( numeric\_date )**

*numeric\_date* – восьмизначное целое число типа Integer в форме ГГГГММДД (например, 19951231).

**Возвращаемая величина**

Дата типа Date

**Описание**

Функция **NumberToDate ( )** возвращает дату, величину типа Date, используя восьмизначное целое число. Например, следующая функция будет иметь результат, равный 31 декабря 2006 года:

```
NumberToDate(20061231)
```

**Пример:**

В следующем примере результат применения одной функции Date вычитается из другого. Получается количество дней между двумя датами.

```
Dim i_elapsed As Integer i_elapsed = CurDate( ) - NumberToDate(20060101) '
i_elapsed теперь равен числу дней, прошедших ' с 1 января 2006 г.
```

**См. также:**

**Функция StringToDate ( )**

## Функция NumberToDateTime()

### Назначение

Returns a DateTime value.

### Синтаксис

NumberToDateTime( numeric\_datetime )

*numeric\_datetime* – семнадцатизначное целое число типа Integer в форме ГодГодГодГодМесМесДеньДеньЧасЧасМинМинСекСекДоляДоляДоля (YYYYMMDDHHMMSSFFF). Например, 20070301214237582 представляет 1 Марта 2007 г. 9:42:37.582 PM.

### Возвращаемая величина

Date/Time

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim fNum as float
dim Y as datetime
fNum = 20070301214237582
Y = NumberToDateTime (fNum)
Print FormatDate$(Y)
Print FormatTime$(Y, "hh:mm:ss.fff tt")
```

---

## Функция NumberToTime()

### Назначение

Returns a Time value.

### Синтаксис

NumberToTime( numeric\_time )

*numeric\_datetime* – семнадцатизначное целое число типа Integer в форме ЧасЧасМинМинСекСекДоляДоляДоля (HHMMSSFFF). Например, 214237582 представляет 9:42:37.582 P.M.

### Возвращаемая величина

Время

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim fNum as float
```



```

dim Y as time
fNum = 214237582
Y = NumberToTime(fNum)
Print FormatTime$(Y, "hh:mm:ss.fff tt")

```

## Функция NumCols( )

### Назначение

Возвращает число колонок таблицы.

### Синтаксис

**NumCols**( *table* )

*table* – имя открытой таблицы;

### Возвращаемая величина

Целое число типа SmallInt.

### Описание

Функция **NumCols( )** возвращает число колонок, из которых состоит открытая таблица.

В число возвращаемых функцией **NumCols( )** колонок не входит специальная колонка "Object" (или "Obj" сокращенно), содержащая ссылки на графические объекты, присоединенные к таблице. Также в число колонок не включается другая специальная колонка RowID, содержащая номера строк таблицы.

**Внимание:** Если в таблице есть временные колонки (например, которые создает **Оператор Add Column**), то число колонок, полученное от функции **NumCols( )** будет включать и временные колонки.

### Ошибки:

ERR\_TABLE\_NOT\_FOUND, если не найдена данная таблица;

### Пример:

```

Dim i_counter As Integer
Open Table "world"
i_counter = NumCols(world)

```

### См. также:

**Функция ColumnInfo( ), Функция NumTables( ), Функция TableInfo( )**

## Функция NumTables( )

### Назначение

Возвращает число открытых на данный момент таблиц.

### Синтаксис

**NumTables** ( )

### Возвращаемая величина

Целое число типа SmallInt.

### Описание

Функция **NumTables**( ) возвращает число открытых на данный момент таблиц.

Если в MapInfo открыта таблица, содержащая Карту улиц (StreetInfo), то на самом деле открыты две связанные таблицы. Например, когда Вы открываете таблицу DCWASHS (карта улиц Вашингтона), MapInfo открывает две составляющие таблицы DCWASHS1.TAB и DCWASHS2.TAB. Тем не менее MapInfo считает DCWASHS одной таблицей, поскольку составляющие таблицы являются частями одной Карты. Так же и функция **NumTables**( ) таблицу, содержащую Карту улиц, будет считать одной открытой таблицей, несмотря на то, что практически она состоит из двух.

### Пример:

```
If NumTables( ) < 1 ThenNote "Нет открытых таблиц. Продолжение невозможно."End If
```

### См. также:

[Оператор Open Table](#), [Функция TableInfo\( \)](#), [Функция ColumnInfo\( \)](#)

---

## Функция NumWindows( )

### Назначение

Возвращает количество открытых на данный момент окон (Карт, Списков, Графиков и Отчетов).

### Синтаксис

**NumWindows** ( )

### Возвращаемая величина

Целое число типа SmallInt.

### Описание

Функция **NumWindows**( ) возвращает число открытых на данный момент окон Карт, Списков, Графиков и Отчетов. Результат функции не зависит от того, в каком состоянии находится окно: свернуто в иконку или нет.

Чтобы определить общее количество выведенных на экран окон, включая вспомогательные (такие как окно Легенды, Информации), используется функция **NumAllWindows**( ).

**Пример:**

```
Dim num_open_wins As SmallInt  
num_open_wins = NumWindows( )
```

**См. также:**

Функция [NumAllWindows\( \)](#), Функция [WindowID\( \)](#)



---

## Функция ObjectDistance( )

### Назначение

Возвращает расстояние между двумя объектами.

### Синтаксис

**ObjectDistance**( *object1*, *object2*, *unit\_name* ), где

*object1* и *object2* - выражения, указывающие на объекты.

*unit\_name* - строка, определяющая единицы измерения расстояния.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

**ObjectDistance**( ) возвращает минимальное расстояние между объектами *object1* и *object2*, вычисленное в сферических координатах. Возвращаемое значение измеряется в единицах *unit\_name*. Если применить сферический метод вычислений не удастся, (например, если координатная система MapBasic - план), то используются декартовы вычисления.

---

## Функция ObjectGeography( )

### Назначение

Возвращает информацию о графическом объекте, определяющую его расположение.

### Синтаксис

**ObjectGeography**( *object*, *attribute* )

*object* это выражение для объекта

*attribute* - это целое, определяющее, какого типа информация будет возвращена.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Параметр *attribute* должен принимать значения целочисленного кода, управляющего типом возвращаемой функцией информации. В следующей таблице в первой колонке приводятся имена кодов параметра *attribute* функции **ObjectGeography**( ), которые установлены в файле стандартных определений MapBasic MAPBASIC.DEF.

Некоторые значения параметра attribute могут адресоваться объектам определенного типа. Например, начальный и конечный угол может быть только у объекта типа "дуга", а текстовые объекты – единственные с атрибутом наклона шрифта. Если объект не поддерживает z- или m-значения или z- или m-значения для данного узла не определены, будет возвращена ошибка.

| Параметры           | Возвращаемая величина – вещественная (Float)   |
|---------------------|--|
| OBJ_GEO_MINX        | Минимальная X-координата минимального прямоугольного покрытия объекта, если его тип не "линия". Иначе возвратится значение, равное OBJ_GEO_LINEBEGX.                           |
| OBJ_GEO_MINY        | Минимальная Y-координата минимального прямоугольного покрытия объекта, если его тип не "линия". Иначе возвратится значение, равное OBJ_GEO_LINEBEGY.                           |
| OBJ_GEO_MAXX        | Максимальная X-координата объекта или его минимального прямоугольного покрытия. Код не применим для объекта типа "точка". Иначе возвратится значение, равное OBJ_GEO_LINEENDX. |
| OBJ_GEO_MAXY        | Максимальная Y-координата объекта или его минимального прямоугольного покрытия. Код не применим для объекта типа "точка". Иначе возвратится значение, равное OBJ_GEO_LINEENDY. |
| OBJ_GEO_ARCBEGANGLE | начальный угол дуги.   |
| OBJ_GEO_ARCENDANGLE | конечный угол дуги.  |
| OBJ_GEO_LINEBEGX    | X-координата начальной точки прямой линии. Только для объекта типа "линия".  |
| OBJ_GEO_LINEBEGY    | Y-координата начальной точки прямой линии. Только для объекта типа "линия".  |
| OBJ_GEO_LINEENDX    | X-координата конечной точки прямой линии. Только для объекта типа "линия".   |
| OBJ_GEO_LINEENDY    | Y-координата конечной точки прямой линии. Только для объекта типа "линия".   |
| OBJ_GEO_POINTX      | X-координата точечного объекта.  |
| OBJ_GEO_POINTY      | Y-координата точечного объекта.  |
| OBJ_GEO_POINTZ      | Z-координата объекта Точка.  |
| OBJ_GEO_POINTM      | M-значение объекта Точка.  |

| Параметры          | Возвращаемая величина – вещественная (Float)  |
|--------------------|---|
| OBJ_GEO_ROUNDADIUS | диаметр окружности, которую можно вписать в закругление угла объекта типа "скругленный прямоугольник". Результат выдается в текущих координатных единицах (например, в градусах). |
| OBJ_GEO_CENTROID   | возвращает точечный объект центроида областей, коллекций, групп точек и полилиний. Обычно используется <b>Оператор Alter Object</b> .   |
| OBJ_GEO_TEXTLINEX  | координата по оси X конца текстовой строки объекта.   |
| OBJ_GEO_TEXTLINEY  | координата по оси Y конца текстовой строки объекта.   |
| OBJ_GEO_TEXTANGLE  | угол поворота текстового объекта.   |

Функция **ObjectGeography( )** теперь поддерживает группы точек и коллекции. Для обоих этих типов поддерживаются атрибуты 1 - 4 (координаты минимального описывающего объект прямоугольника (МОП)).

|                  |  |
|------------------|--|
| OBJ_GEO_MINX (1) | минимальная X-координата описывающего прямоугольника.  |
| OBJ_GEO_MINY (2) | минимальная Y-координата описывающего прямоугольника.  |
| OBJ_GEO_MAXX (3) | максимальная X-координата описывающего прямоугольника. |
| OBJ_GEO_MAXY (4) | максимальная Y-координата описывающего прямоугольника. |

### Пример:

Будут найдены координаты начальной точки линии из таблицы City. Кроме того, используется **Оператор Set Map** для перемещения центра просмотра Карты в начальную точку прямой линии.

```

Include "MAPBASIC.DEF"
Dim i_obj_type As Integer, f_x, f_y As Float
Open Table "city"
Map From city
Fetch First From city
' at this point, the expression:
' city.obj
' represents the graphical object that's attached
' to the first record of the CITY table.
i_obj_type = ObjectInfo(city.obj, OBJ_INFO_TYPE)
If i_obj_type = OBJ_LINE Then
    f_x = ObjectGeography(city.obj, OBJ_GEO_LINEBEGX)
    f_y = ObjectGeography(city.obj, OBJ_GEO_LINEBEGY)
    Set Map Center (f_x, f_y)
End If

```

См. также:

Функция Centroid( ), Функция CentroidX( ), Функция CentroidY( ), Функция ObjectInfo( )



## Функция **ObjectInfo( )**

### Назначение

Возвращает стиль линии, штриха и другие величины, описывающие графический объект, а также его тип.

### Синтаксис

**ObjectInfo**( *object*, *attribute* )

*object* это выражение для объекта

*attribute* - это целое, определяющее, какого типа информация будет возвращена.

### Возвращаемая величина

SmallInt, integer, string, float, Pen, Brush, Symbol или Font, в зависимости от параметра атрибута

OBJ\_INFO\_NPOLYGONS (21) – целое число полигонов в объекте типа "область" или число ломаных компонент в объекте типа "полилиния".

OBJ\_INFO\_NPOLYGONS+N (21) – целое число узлов в N-ом полигоне "области" или в N-ой ломаной "полилинии".

**Внимание:** Для объектов типа "область" число узлов многоугольника будет на единицу больше, чем число вершин у многоугольника, потому что MapInfo Professional считает первый узел дважды (один раз как первый узел и второй раз как последний узел). Так, функция **ObjectInfo( )** возвращает 4 для треугольной области.

### Описание

Функция **ObjectInfo( )** возвращает полную информацию о единственной характеристике графического объекта. Первым параметром должен быть определитель объекта (например, имя переменной объекта, или выражение описывающее объекты таблицы вида *tablename.obj*).

Каждому объекту присвоено несколько атрибутивных параметров. Например, тип любого объекта: точка, линия, область и т. п., идентифицируется по атрибуту "type". Объекты большинства типов имеют атрибуты Pen и/или Brush, которые диктуют внешний вид объектов. Функция **ObjectInfo( )** возвращает единственный атрибут заданного объекта. Возвращаемый атрибут зависит от использованного параметра атрибута. Поэтому, если требуется определить несколько характеристик объекта, то функцию **ObjectInfo( )** следует вызывать несколько раз с разными параметрами в каждом из вызовов.

В таблице ниже перечислены разные атрибуты и возвращаемые значения для каждого из них.

| Параметры <i>attribute</i> | Возвращаемая величина   |
|----------------------------|---|
| OBJ_INFO_TYPE (1)          | Короткое целое, тип объекта; возвращаемые значения кода типа перечислены в таблице ниже (например, OBJ_TYPE_LINE). Атрибут, заданный в DEF-файле – 1 (ObjectInfo( Object, 1 )).   |
| OBJ_INFO_PEN (2)           | Величина типа Pen. Стилль линии для объектов типа "дуга", "эллипс", "линия", "полилиния", "рамка", "область", "прямоугольник" и "сглаженный прямоугольник".   |
| OBJ_INFO_BRUSH (3)         | Величина типа Brush. Стилль штриховки объектов типа "эллипс", "рамка", "область", "прямоугольник" и "сглаженный прямоугольник".   |
| OBJ_INFO_TEXTFONT (2)      | Величина типа Font. Стилль шрифта текстового объекта.<br><br><b>Внимание:</b> Если текстовый объект принадлежит таблице (а не Отчету), то размер шрифта равен нулю, а размер шрифта динамически определяется MapInfo в зависимости от размера окна Карты. |
| OBJ_INFO_SYMBOL (2)        | величина типа Symbol. Стилль символа точечного объекта.   |
| OBJ_INFO_NPTS (20)         | целое число узлов в полилинии или в многоугольнике области.   |
| OBJ_INFO_SMOOTH (4)        | логический признак сглаженности объекта типа "полилиния".   |
| OBJ_INFO_FRAMEWIN (4)      | целое число идентификатора окна, вставленного в рамку Отчета  |
| OBJ_INFO_FRAMETITLE (6)    | строка с заголовком рамки.  |
| OBJ_INFO_NPOLYGONS (21)    | короткое целое число полигонов в объекте типа "область" или число ломаных компонент в объекте типа "полилиния".   |

| Параметры <i>attribute</i> | Возвращаемая величина   |
|----------------------------|---|
| OBJ_INFO_NPOLYGONS+N (21)  | <p>Целое число узлов <i>N</i>-ого полигона области или <i>N</i>-ого сегмента полилинии.</p> <p><b>Внимание:</b> Для объектов типа "область" число узлов многоугольника будет на единицу больше, чем число вершин у многоугольника, потому что MapInfo Professional считает первый узел дважды (один раз как первый узел и второй раз как последний узел). Так, функция <b>ObjectInfo( )</b> возвращает 4 для треугольной области.</p> |
| OBJ_INFO_TEXTSTRING (3)    | Величина типа String. Текстовое содержимое объекта типа "текст". Если объект состоит из нескольких строк, то результат будет включать символ конца строки (Chr\$(10)).  |
| OBJ_INFO_TEXTSPACING (4)   | Вещественное число 1, 1.5 или 2, определяющее интерлиньяж в текстовом объекте.  |
| OBJ_INFO_TEXTJUSTIFY (5)   | Число типа SmallInt, определяющее выравнивание текста: 0 – по левому краю, 1 – по центру, 2 – по правому краю.  |
| OBJ_INFO_TEXTARROW (6)     | Число типа SmallInt, определяющее стиль указки в текстовом объекте: 0 – нет указки, 1 – просто линия, 2 – стрелка.  |
| OBJ_INFO_FILLFRAME (7)     | Величина типа Logical. "Да" (TRUE), если объект типа "рамка" показывает Карту и для него установлен режим "Заполнить Рамку Картой".   |
| OBJ_INFO_NONEMPTY (11)     | Логическая величина, TRUE, если коллекция имеет узлы, FALSE, если объект пустой.  |
| OBJ_INFO_REGION (8)        | Возвращает значение соответствующее полигонам, входящим в коллекцию. Если коллекция не содержит полигонов, будет возвращен пустой полигон. Этот запрос действует только для объектов типа "коллекция".  |
| OBJ_INFO_PLINE (9)         | Возвращает значение соответствующее полилиниям, входящим в коллекцию. Если коллекция не содержит полилиний, будет возвращена пустая полилиния. Этот запрос действует только для объектов типа "коллекция".  |

| Параметры <i>attribute</i> | Возвращаемая величина  |
|----------------------------|--|
| OBJ_INFO_MPOINT (10)       | Возвращает значение соответствующее группам точек, входящим в коллекцию. Если коллекция не содержит групп точек, будет возвращена пустая группа точек. Этот запрос действует только для объектов типа "коллекция". |
| OBJ_INFO_Z_UNIT_SET(12)    | Логическая величина – показывает, что единицы измерения Z- координат заданы.   |
| OBJ_INFO_Z_UNIT(13)        | Строка, указывающая единицы расстояния для z- значений. Пустая строка будет возвращена, если единицы измерения Z-координат не заданы.  |
| OBJ_INFO_HAS_Z(14)         | Логическая величина – показывает, что объект имеет Z- координаты.  |
| OBJ_INFO_HAS_M(15)         | Логическая величина – показывает, что объект имеет m- значения.  |

Коды параметров в левой колонке (например, OBJ\_INFO\_TYPE) заданы в MapBasic-файле определителей MAPBASIC.DEF. Если предполагается вызывать функцию **ObjectInfo( )**, то программа должна ссылаться на этот файл в процедуре Include "MAPBASIC.DEF".

Каждый графический атрибут может применяться только для определенных типов графических объектов. Например, точечные объекты являются единственными с атрибутами Symbol, а текстовые объекты – атрибутами Font. Поэтому, функция **ObjectInfo( )** не может возвращать любой атрибут объекта любого типа.

Если Вы используете код OBJ\_INFO\_TYPE как значение параметра attribute, то функция **ObjectInfo( )** вернет код, соответствующий типу графического объекта. В следующей таблице приведены имена этих кодов из файла стандартных определений.

**Код объекта OBJ\_INFO\_TYPE**

| Код объекта OBJ_INFO_TYPE | Соответствующий тип объекта |
|---------------------------|-----------------------------|
| OBJ_TYPE_ARC              | дуга                        |
| OBJ_TYPE_ELLIPSE          | эллипс / окружность         |
| OBJ_TYPE_LINE             | прямая                      |
| OBJ_TYPE_PLINE            | полилиния                   |
| OBJ_TYPE_POINT            | точка                       |
| OBJ_TYPE_FRAME            | рамка в окне Отчета         |
| OBJ_TYPE_REGION           | полигон                     |
| OBJ_TYPE_RECT             | прямоугольник               |

Код объекта OBJ\_INFO\_TYPE (continued)

| Код объекта OBJ_INFO_TYPE | Соответствующий тип объекта  |
|---------------------------|------------------------------|
| OBJ_TYPE_ROUNDRECT        | скругленный прямоугольник    |
| OBJ_TYPE_TEXT             | текстовый объект             |
| OBJ_TYPE_MULTIPPOINT      | коллекция текстовых объектов |

**Пример:**

```
Include "MAPBASIC.DEF"Dim counter, obj_type As Integer Open Table "city"
Fetch First From city' ' В этом месте программы выражение city.obj'
представляет графический объект, присоединенный ' к первой строке таблицы
CITY.
obj_type = ObjectInfo(city.obj, OBJ_INFO_TYPE) Do Case obj_type Case
OBJ_TYPE_LINENote "Первый объект в таблице - прямая линия."Case
OBJ_TYPE_PLINENote "Первый объект в таблице - полилиния,... " counter =
ObjectInfo(city.obj, OBJ_INFO_NPTS) Note " ... которая имеет " +
Str$(counter) + " узлов." Case OBJ_TYPE_REGIONNote "Первый объект в
таблице - область,... " counter = ObjectInfo(city.obj, OBJ_INFO_NPOLYGONS)
Note ", которая состоит из " + Str$(counter) + " полигонов....",counter =
ObjectInfo(city.obj, OBJ_INFO_NPOLYGONS+1)Note "а первый полигон имеет " +
Str$(counter) + " узлов"End Case
```

**См. также:**

[Оператор Alter Object](#), [Предложение Brush](#), [Предложение Font](#), [Функция ObjectGeography\( \)](#), [Предложение Pen](#), [Предложение Symbol](#)

**Функция ObjectLen( )**

**Назначение**

Возвращает географическую протяженность объекта линии или полилинии.

**Синтаксис**

```
ObjectLen( expr, unit_name )
```

*expr* – объектное выражение (выражение, результат которого есть величина типа Object);  
*unit\_name* – это строка, соответствующая имени единиц измерения расстояния (например, "km").

**Возвращаемая величина**

Вещественное число типа Float.

### Описание

Функция **ObjectLen( )** возвращает длину выражения объекта. Помните, что ненулевые длины имеют только объекты линий и полилиний, для измерения периметра прямоугольника, эллипса или полигона, используйте [Функция Perimeter\( \)](#).

Возвращаемое функцией **ObjectLen( )** значение длины периметра измеряется в единицах длины, определенных параметром *unit\_name*; например, для получения длины в милях, укажите "mi" в качестве *unit\_name*. Список допустимых единиц измерения смотрите в [Оператор Set Distance Units on page 633](#).

В большинстве случаев MapInfo Professional проводит либо декартовы, либо сферические вычисления. Обычно выполняются сферические вычисления; если координатная система - план, то выполняются декартовы вычисления..

### Пример:

```
Dim geogr_length As Float
Open Table "streets" Fetch First From
streets
geogr_length = SphericalObjectLen(streets.obj, "mi")
' geogr_length
теперь содержит протяженность ' сегмента улицы в милях
```

### См. также:

[Функция Distance\( \)](#), [Функция Perimeter\( \)](#), [Оператор Set Distance Units](#)

---

## Функция `ObjectNodeHasM( )`

### Назначение

Возвращает TRUE, если заданный узел области, полилинии или группы точек имеет m-значение.

### Синтаксис

**ObjectNodeHasM( *object*, *polygon\_num*, *node\_num* )**

*object* это выражение для объекта

*polygon\_num* – целое, положительное, определяющее, какой полигон или сегмент опрашиваются. Оно игнорируется для объектов типа Группа точек (используется только для регионов и полилиний).

*node\_num* – положительное целое число, показывающее какой узел считывается

### Возвращаемая величина

Логическое

### Описание

Функция **ObjectNodeHasM( )** возвращает TRUE, если заданный узел области, полилинии или группы точек имеет m-значение.

Параметр *polygon\_num* должен иметь значение от единицы и выше. Он указывает, какой полигон (если запрос сделан к региону) или какая секция (если запрос к полилинии) будут опрошены. Функция **ObjectInfo( )** вызывается для определения числа полигонов или сегментов объекта. Функция **ObjectNodeHasM( )** поддерживает объекты – группы точек и возвращает значение TRUE, если заданный узел объекта группа точек имеет m-значение.

Параметр *node\_num* должен иметь значение от единицы и выше; он указывает MapBasic, какой из узлов объекта будет опрошен. Функция **ObjectInfo( )** может быть использована для определения числа узлов в объекте.

Если объект не может содержать m-значений или m-значение для заданного узла не установлено, функция возвращает FALSE.

### Пример:

Следующий пример показывает опрос первого графического элемента в таблице Routes. Если первый объект является полилинией, программа определит, что первый узел содержит z-координату или m-значение, и запросит z-координату и m-значение первого узла полилинии.

```
Dim i_obj_type As SmallInt, z, m As FloathasZ, hasM as LogicalOpen Table
"routes" Fetch First From routes ' здесь, выражение: ' routes.obj '
определяет географический объект ' первой записи таблицы маршрутов.
i_obj_type = ObjectInfo(routes.obj, OBJ_INFO_TYPE) If i_obj_type =
OBJ_PLINE Then ' ... then the object is a polyline...If
(ObjectNodeHasZ(routes.obj, 1, 1)) Then z = ObjectNodeZ(routes.obj, 1, 1)
' получаем z-координатуEnd IfIf (ObjectNodeHasM(routes.obj, 1, 1)) Then m
= ObjectNodeM(routes.obj, 1, 1) ' получаем m-значениеEnd IfEnd If
```

### См. также:

[Информация об объектах Карты](#), [Функция ObjectInfo\( \)](#)

---

## Функция **ObjectNodeHasZ( )**

### Назначение

Возвращает TRUE, если заданный узел области, полилинии или группы точек имеет Z-координату.

### Синтаксис

**ObjectNodeHasZ**( *object*, *polygon\_num*, *node\_num* )

*object* это выражение для объекта

*polygon\_num* – целое, положительное, определяющее, какой полигон или сегмент опрашиваются. Оно игнорируется для объектов типа Группа точек (используется только для регионов и полилиний).

*node\_num* – положительное целое число, показывающее какой узел считывается

### Возвращаемая величина

Логическое

### Описание

Функция **ObjectNodeHasZ( )** возвращает значение TRUE, если заданный узел полигона, полилинии или группы-точек имеет z-координату. Параметр *polygon\_num* должен иметь значение от единицы и выше. Он указывает, какой полигон (если запрос сделан к региону) или какая секция (если запрос к полилинии) будут опрошены. [Функция ObjectInfo\( \)](#) вызывается для определения числа полигонов или сегментов объекта. Функция **ObjectNodeHasZ( )** поддерживает объекты – группы точек и возвращает значение TRUE, если заданный узел объекта группа точек имеет z-координату.



Параметр *node\_num* должен иметь значение от единицы и выше; он указывает MapBasic, какой из узлов объекта будет опрошен. **Функция ObjectInfo( )** может быть использована для определения числа узлов в объекте.

Если *объект* не поддерживает z-координаты или z-координата для этого узла не определена, то будет возвращено значение FALSE.

### Пример:

Следующий пример показывает опрос первого графического элемента в таблице Routes. Если первый объект является полилинией, программа определит, что первый узел содержит z-координату или m-значение, и запросит z-координату и m-значение первого узла полилинии.

```
Dim i_obj_type As SmallInt, z, m As FloathasZ, hasM as LogicalOpen Table
"routes" Fetch First From routes ' здесь, выражение: ' routes.obj'
определяет географический объект ' первой записи таблицы маршрутов.
i_obj_type = ObjectInfo(routes.obj, OBJ_INFO_TYPE) If i_obj_type =
OBJ_PLINE Then ' ... then the object is a polyline...If
(ObjectNodeHasZ(routes.obj, 1, 1)) Then z = ObjectNodeZ(routes.obj, 1, 1)
' получаем z-координатуEnd IfIf (ObjectNodeHasM(routes.obj, 1, 1)) Then m
= ObjectNodeM(routes.obj, 1, 1) ' получаем m-значениеEnd IfEnd If
```

### См. также:

**Информация об объектах Карты, Функция ObjectInfo( )**

## Функция ObjectNodeM( )

### Назначение

Возвращает m-координату заданного узла полигона, полилинии или группы точек.

### Синтаксис

**ObjectNodeM( object, polygon\_num, node\_num )**

*object* это выражение для объекта

*polygon\_num* – целое, положительное, определяющее, какой полигон или сегмент опрашиваются. Оно игнорируется для объектов типа Группа точек (используется только для регионов и полилиний).

*node\_num* – положительное целое число, показывающее какой узел считается

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **ObjectNodeM( )** возвращает M-значения указанного узла в регионе, полилинии или группе точек.

Параметр *polygon\_num* должен иметь значение от единицы и выше. Он указывает, какой полигон (если запрос сделан к региону) или какая секция (если запрос к полилинии) будут опрошены. Функция `ObjectInfo( )` вызывается для определения числа полигонов или сегментов объекта. Функция `ObjectNodeM( )` поддерживает объекты Группа точек и возвращает M-величину заданного узла группы точек.

Параметр *node\_num* должен иметь значение от единицы и выше; он указывает MapBasic, какой из узлов объекта будет опрошен. Функция `ObjectInfo( )` может быть использована для определения числа узлов в объекте.

Если объект не поддерживает m-значения или m-значения для данного узла не определены, будет возвращена ошибка.

**Пример:**

Следующий пример показывает опрос первого графического элемента в таблице Routes. Если первый объект это полилиния, то программа опросит Z-координаты и M-значения первого узла полилинии.

```
Dim i_obj_type As SmallInt, z, m As Float
Open Table "routes" Fetch First
From routes ' здесь, выражение: ' routes.obj' определяет географический
объект ' первой записи таблицы маршрутов.
i_obj_type = ObjectInfo(routes.obj, OBJ_INFO_TYPE)
If i_obj_type = OBJ_PLINE Then ' ... объект - полилиния...
z = ObjectNodeZ(routes.obj, 1, 1) ' читать z-координату
m = ObjectNodeM(routes.obj, 1, 1) ' читать m-значение
End If
```

**См. также:**

**Информация об объектах Карты, Функция ObjectInfo( )**

---

## Функция ObjectNodeX( )

**Назначение**

Возвращает X-координату заданного узла полигона или полилинии.

**Синтаксис**

**ObjectNodeX**( *object*, *polygon\_num*, *node\_num* )

*object* это выражение для объекта

*polygon\_num* – целое, положительное, определяющее, какой полигон или сегмент опрашиваются. Оно игнорируется для объектов типа Группа точек (используется только для регионов и полилиний).

*node\_num* – положительное целое число, показывающее какой узел считывается

**Возвращаемая величина**

Вещественное число типа Float.

**Описание**

Функция **ObjectNodeX( )** возвращает X-координату заданного узла в полигоне или полилинии. Похожая **Функция ObjectNodeY( )** возвращает Y-координату.

Параметр *polygon\_num* должен иметь значение от единицы и выше. Он указывает, какой полигон (если запрос сделан к региону) или какая секция (если запрос к полилинии) будут опрошены. **Функция ObjectInfo( )** вызывается для определения числа полигонов или сегментов объекта. Функция **ObjectNodeX( )** поддерживает объекты Группы точек и возвращает X-координату заданного узла группы точек.

Параметр *node\_num* должен иметь значение от единицы и выше; он указывает MapBasic, какой из узлов объекта будет опрошен. **Функция ObjectInfo( )** может быть использована для определения числа узлов в объекте. Координата, которую Вы получите в результате

применения **ObjectNodeX( )**, будет выведена в системе координат, которая определена как текущая для MapBasic. По умолчанию в MapBasic используется проекция Долгота/Широта.. Подробнее о проекциях и координатных системах смотрите в разделе: "**Оператор Set CoordSys on page 629**".

### Пример:

Следующий пример показывает опрос первого графического элемента в таблице Routes. Если первый объект полилиния, то считываются координаты первого узла. На этом месте создается точечный объект.

```
Dim i_obj_type As SmallInt, x, y As Float, new_pnt As Object
Open Table "routes" Fetch First From routes ' здесь, выражение: ' routes.obj '
определяет географический объект ' первой записи таблицы маршрутов.
i_obj_type = ObjectInfo(routes.obj, OBJ_INFO_TYPE)
If i_obj_type = OBJ_PLINE Then
' ... then the object is a polyline...
  x = ObjectNodeX(routes.obj, 1, 1) ' read longitude
  y = ObjectNodeY(routes.obj, 1, 1) ' read latitude
  Create Point Into Variable new_pnt (x, y)
  Insert Into routes (obj) Values (new_pnt)
End If
```

### См. также:

**Оператор Alter Object**, **Функция ObjectGeography( )**, **Функция ObjectInfo( )**, **Функция ObjectNodeY( )**, **Оператор Set CoordSys**

---

## Функция ObjectNodeY( )

### Назначение

Возвращает Y-координату заданного узла полигона или полилинии.

### Синтаксис

**ObjectNodeY( object, polygon\_num, node\_num )**

*object* это выражение для объекта

*polygon\_num* – целое, положительное, определяющее, какой полигон или сегмент опрашиваются. Оно игнорируется для объектов типа Группа точек (используется только для регионов и полилиний).

*node\_num* – положительное целое число, показывающее какой узел считывается

### Возвращаемая величина

Вещественное число типа Float.

**Описание**

Функция **ObjectNodeY( )** возвращает Y-координату заданного узла в полигоне или полилинии. Более подробную информацию см. в разделе **Функция ObjectNodeX( )** на стр. 33.

**Пример:**

См. **Функция ObjectNodeX( )** на стр. 33.

**См. также:**

**Оператор Alter Object**, **Функция ObjectGeography( )**, **Функция ObjectInfo( )**, **Оператор Set CoordSys**

---

**Функция ObjectNodeZ( )****Назначение**

Возвращает z-координату указанного узла полигона, полилинии или объекта "группа точек".

**Синтаксис**

**ObjectNodeZ( object, polygon\_num, node\_num )**

*object* это выражение для объекта

*polygon\_num* – целое, положительное, определяющее, какой полигон или сегмент опрашиваются. Оно игнорируется для объектов типа Группа точек (используется только для регионов и полилиний).

*node\_num* – положительное целое число, показывающее какой узел считывается.

**Возвращаемая величина**

Вещественное число типа Float.

**Описание**

Функция **ObjectNodeZ( )** возвращает Z-координату заданного узла полигона, полилинии или группы точек.

Параметр *polygon\_num* должен иметь значение от единицы и выше. Он указывает, какой полигон (если запрос сделан к региону) или какая секция (если запрос к полилинии) будут опрошены. **Функция ObjectInfo( )** вызывается для определения числа полигонов или сегментов объекта. Функция **ObjectNodeZ( )** поддерживает объекты Группа точек и возвращает Z-координату заданного узла группы точек.

Параметр *node\_num* должен иметь значение от единицы и выше; он указывает MapBasic, какой из узлов объекта будет опрошен. **Функция ObjectInfo( )** может быть использована для определения числа узлов в объекте.

Если объект не поддерживает Z-значения или Z-значения для данного узла не определены, будет возвращена ошибка.

### Пример:

Следующий пример показывает опрос первого графического элемента в таблице Routes. Если первый объект это полилиния, то программа опросит Z-координаты и M-значения первого узла полилинии.

```
Dim i_obj_type As SmallInt, z, m As Float
Open Table "routes" Fetch First
From routes ' здесь, выражение: ' routes.obj' определяет географический
объект ' первой записи таблицы маршрутов.
i_obj_type = ObjectInfo(routes.obj, OBJ_INFO_TYPE)
If i_obj_type = OBJ_PLINE Then ' ... объект - полилиния...
z = ObjectNodeZ(routes.obj, 1, 1) ' читать z-координату
m = ObjectNodeM(routes.obj, 1, 1) ' читать m-значение
End If
```

### См. также:

[Информация об объектах Карты, Функция ObjectInfo\( \)](#)

---

## Оператор Objects Check

### Назначение

Проверяет таблицы на предмет некорректности данных, что может вызвать проблемы и ошибочные результаты применения различных операций.

### Синтаксис

```
Objects Check From tablename Into Table tablename
[ SelfInt [ Symbol Clause ] ]
[ Overlap [ Pen Clause ] [ Brush Clause ] ]
[ Gap Area [ Unit Units ] [ Pen Clause ] [ Brush Clause ] ] ]
```

*tablename* – строка с именем открытой таблицы;

*Clause* – выражение;

*Units* - единицы измерения площадей.

### Описание

**Objects Check** проверит таблицу, указанную в предложении **From** на наличие некорректных данных, которые могут вызвать проблемы при выполнении различных операций. Проверка будет осуществляться только для областей. Области будут проверены на наличие самопересечений, взаимных пересечений и пустоты (бреши).

Самопересечения могут вызвать ошибки при выполнении вычислений, включая расчет площади области. Они также могут привести к некорректным результатам при выполнении различных операций над объектами, таких как комбинация, создание буферных зон, вырезание и деление.

Для обозначения самопересечений создаются точечные объекты, которые помещаются в выходную таблицу. Выходная таблица может быть определена в предложении **Into Table**. Если предложение **Into Table** не указано, выходные данные размещаются во входной таблице.

Если используется опция **SelfInt**, то таблица будет проверена на самопересечения. Стиль создаваемых точечных объектов определяется через *параметр Symbol*. По умолчанию, это - красная булавка размером 28-точек.

Большинство таблиц с областями относятся к таблицам границ. Подготовленные файлы `states.tab` и `world.tab` с образцами данных – примеры таблиц с границами. В таких таблицах границы не должны перекрываться (например, штат Utah не должен перекрывать Wyoming). Опция **Overlap** будет проверять таблицы на наличие перекрытия регионов. В выходной таблице будут созданы регионы, представляющие собой области перекрытия. Стиль регионов-перекрытий определяется через **Brush Clause** (заливка) и **Pen Clause** (границы). По умолчанию, для этих регионов предусмотрена сплошная желтая заливка и тонкая черная граница.

Пустоты (бреши) - это замкнутые области, которые не образуют полигонов. В таблице с границами полигоны должны иметь общие границы. В идеальном случае в таблице не должны встречаться перекрытия полигонов и пустоты между ними. В некоторых случаях пустоты между полигонами имеют смысл и право на существование. Например, Великие озера на карте мира являются "пустотой" между Канадой и США. Однако, в большинстве случаев пустоты являются результатом плохого согласования общих границ между полигонами. Такие бреши обычно имеют малые размеры.

Чтобы успешно отделить допустимые пустоты (например, Великие озера) от заведомо ненужных брешей, используется предложение **Gap Area**. Любые пустоты больше заданной площади будут оставляться без изменения. Единицы измерения площади **Gap Area** задаются предложением **Units**. Если подпредложение **Unit** не задано, то площадь брешей **Gap Area** будет измеряться в текущих единицах измерения MapBasic.

Пустоты (бреши) будут оформлены стилями, которые задаются параметрами **Pen** и **Brush** сразу после ключевого слова **Gap**. По умолчанию, эти полигоны изображаются с синей заливкой и тонкой черной границей.

### Пример:

В этом примере функция **Objects Check** используется для проверки областей таблицы `TestFile`, результаты сохраняются в таблице `DumpFile`. Кроме того, с помощью ключевого слова **Overlap** выполняется проверка на наличие перекрытий, изменены стандартные стили для точек (самопересечения) и полигонов (перекрытия).

```
objects check from TestFile into table Dumpfile Overlap
SelfInt Symbol (67,16711680,28)
Overlap Pen (1,2,0) Brush (2,16776960,0)
Gap 100000 Units "sq mi" Pen (1,2,0) Brush (2,255,0)
```

**См. также:**

**Оператор Objects Enclose**

## Оператор Objects Clean

### Назначение

Корректирует топологию объектов из данной таблицы, дополнительно может удалять перекрытия и закрывать бреши между полигонами. Таблица может быть таблицей выборки (Selection). Все объекты, подлежащие коррекции, должны быть замкнутыми (полигоны, прямоугольники, скругленные прямоугольники или эллипсы).

### Синтаксис

```
Objects Clean From tablename  
    [ Overlap ]  
    [ Gap Area [ Unit Units ]]
```

*tablename* – строка с именем открытой таблицы;

*Units* - единицы измерения площадей.

### Описание

Объекты из исходной таблицы *tablename* проверяются на предмет топологической корректности, например, самопересечений, наложений и пустот. Самопересекающиеся полигоны в форме "восьмерки" будут превращены в два полигона с общей вершиной. Полигоны, содержащие острые выступы подвергаются обработке, при которой часть таких выступов удаляется. Исправленный объект заменит исходный.

Если включено ключевое слово **Overlap**, то области наложения полигонов друг на друга удаляются из полигонов. Часть перекрытия будет удалена из всех перекрывающихся полигонов, кроме того, у которого наибольшая площадь.

**Внимание:** **Objects Clean** удаляет перекрытия, когда один объект полностью попадает в другой. Обратите внимание, что это исключение из правила "крупнейший объект побеждает". Если объект находится полностью внутри другого объекта, то меньший объект сохраняется, а в большом объекте образуется дырка, таким образом, что объекты больше не имеют перекрытий.

Пустоты (бреши) - это замкнутые области, которые не образуют полигонов. В таблице с границами полигоны должны иметь общие границы. В идеальном случае в таблице не должны встречаться перекрытия полигонов и пустоты между ними. В некоторых случаях пустоты между полигонами имеют смысл и право на существование. Например, Великие озера на карте мира являются "пустотой" между Канадой и США. Однако, в большинстве случаев пустоты являются результатом плохого согласования общих границ между полигонами. Такие бреши обычно имеют малые размеры.

Чтобы успешно отделить допустимые пустоты (например, Великие озера) от заведомо ненужных брешей, используется предложение **Gap Area**. Любые пустоты больше заданной площади будут оставляться без изменения. Единицы измерения площади пустот **Gap Area** задаются предложением **Units**. Если подпредложение **Unit** не задано, то площадь брешей **Gap Area** измеряется в текущих единицах MapBasic. Пустоты будут удалены путем их объединения с соседним полигоном, причем именно тем, у которого площадь больше. Чтобы иметь представление о величине площади пустот "Gap Area", используйте **Оператор Objects**



**Check.** Любая пустота, которую отметит **Оператор Objects Check**, будет удалена оператором **Objects Clean**, аналогично команде "Коррекция топологии" из MapInfo Professional.

### Пример:

```
Open Table "STATES.TAB" Interactive
Map From STATES
Set Map Layer 1 Editable On
select * from STATES
Objects Clean From Selection Overlap Gap 10 Units "sq m"
```

### См. также:

**Оператор Create Object, Оператор Objects Disaggregate, Оператор Objects Check**

## Оператор Objects Combine

### Назначение

Объединяет объекты в таблице так, как это делает команда MapInfo Professional **ОБЪЕКТЫ > КОМБИНАЦИЯ**.

### Синтаксис

```
Objects Combine
[ Into Target ]
[ Data column = expression [ , column = expression ... ] ]
```

*column* – строка с именем колонки изменяемой таблицы.

*expression* - выражение для расчета значений *column*.

### Описание

Оператор **Objects Combine** создает объект, представляющий собой географическое объединение выбранных объектов. С помощью оператора **Objects Combine** можно также обобщать данные, вычислять суммы и средние значения величин из записей, к которым присоединены объекты.

Оператор **Objects Combine** соответствует команде MapInfo Professional **ОБЪЕКТЫ > КОМБИНАЦИЯ**. Описание выполняющейся операции смотрите в описании команды **ОБЪЕКТЫ > КОМБИНАЦИЯ** в *Справочнике MapInfo Professional*. Если Вы в MapInfo Professional выполните команду **ОБЪЕКТЫ > КОМБИНАЦИЯ** и при этом будет открыто окно MapBasic, то в протоколе выполненных действий будет использован оператор **Objects Combine**. Объекты участвующие в операции combine должны быть или все замкнутыми (т.е. регионы, прямоугольники, скруглённые прямоугольники или эллипсы) или все линейными (т.е. линии, полилинии или дугу). Невозможно объединять замкнутые и линейные объекты или точечные и текстовые.

Дополнительный параметр **Into Target** допускается, если задан изменяемый объект (либо пользователем, либо, используя **Оператор Set Target**), и этот изменяемый состоит из единственного объекта. Если в операторе есть предложение **Into Target**, то MapInfo Professional будет комбинировать изменяемый объект с выбранными объектами на Карте. Объект, полученный в результате комбинирования, заменит изменяемый объект в таблице.

Если выбранные объекты, участвующие в комбинировании **Into Target**, находятся в той же таблице, что и изменяемый объект, то MapInfo Professional удалит записи, к которым присоединены выбранные объекты.

Если Вы не используете предложение **Into Target**, то MapInfo Professional комбинирует только выбранные объекты без использования изменяемого объекта, если он назначен. Если опустить параметр **Into Target**, MapInfo Professional объединит выбранные объекты, не трогая изменяемый (если такой изменяемый объект существует). Выбранные объекты и соответствующие им строки в таблице удаляются, новая строка с объектом, полученным в результате комбинирования, добавляется в конец таблицы.

Предложение **Data** управляет обобщением данных. Информация об обобщении данных приводится в описании команды **ОБЪЕКТЫ > КОМБИНАЦИЯ** в *Руководстве пользователя MapInfo Professional*. За ключевым словом **Data** должен следовать список определений через запятую. Каждое определение является выражением, по которому будет вычислено (или изменено) значение в определенной колонке для записи, которая будет получена в результате выполнения оператора Objects Combine.

Вычисления или изменения должны производиться в соответствии с типом колонки (числовым, строковым и т. п.)

| Выражение                               | Описание   |
|---|--|
| <code>col_name = col_name</code>        | содержимое колонки не меняется.  |
| <code>col_name = value</code>           | MapBasic помещает значение value в поле записи объекта, полученного в результате.  |
| <code>col_name = Sum( col_name )</code> | используется только для числовых колонок. MapBasic помещает сумму значений колонки col_name из всех записей объектов, участвующих в комбинации, в поле записи объекта, полученного в результате. |

| Выражение   | Описание   |
|---|--|
| <code>col_name = Avg( col_name )</code>             | используется только для числовых колонок. MapBasic помещает среднее из значений колонки <code>col_name</code> из всех записей объектов, участвующих в комбинации, в поле записи объекта, полученного в результате.   |
| <code>col_name = WtAvg( colname, wtcolname )</code> | используется только для числовых колонок. MapInfo Professional помещает взвешенное среднее число из значений колонки <code>colname</code> из всех записей объектов, участвующих в комбинации, в поле записи объекта, полученного в результате. В качестве коэффициентов веса используются значения из колонки <code>wtcolname</code> . |

Список **Data** может состоять из определений для всех колонок таблицы. Если в списке определены не все колонки, то MapBasic разместит пустые значения в неописанные поля новой записи. Если предложения **Data** нет в операторе, но используется предложение **Into Target**, то MapInfo сохранит значения изменяемого объекта в записи.

Если в операторе не используется ни предложение **Data**, ни предложение **Into Target**, то результирующий объект будет помещен в новую строку и MapInfo Professional разместит нулевые и пустые значения в поля этой записи.

См. также:

Функция **Combine( )**, Оператор **Set Target**

## Оператор Objects Disaggregate

### Назначение

Разъединяет объект на составляющие его компоненты.

### Синтаксис

```
Objects Disaggregate [ Into Table name ]
[ All | Collection ]
[ Data column_name = expression [ , column_name = expression ... ]
```

*name* – строка с именем таблицы, в которой будут храниться новые разобщенные объекты.

*column\_name* – строка с именем колонки изменяемой таблицы.

*expression* – выражение, которым задаётся то, что будет записано в колонку *column\_name*.

### Описание

Если объект включает в себя несколько других объектов, то для каждого объекта, входящего в состав основного объекта, в итоговой таблице создается новый объект.

По умолчанию, каждый составной объект будет разделен на элементарные одиночные объекты. Регион будет разделен на некоторое число полигонов, в зависимости от того, установлен ли флаг **All**. Если флажок **All** установлен, то будет создано множество отдельных одиночных объектов-полигонов. Для островов (внутренние границы) будут созданы отдельные объекты-полигоны. Если флажок **All** не выставлен, то в результирующих объектах острова сохраняются. Например, если исходный регион содержит 3 полигона и один из них является островом в другом полигоне, то в результате получится 2 объекта-полигона, один из которых содержит остров.

Сложные полилинейные объекты будут разбиты на отдельные полилинии, группы точек -на отдельные точечные объекты (по одному объекту для каждого узла в исходном объекте "Группа точек").

Коллекции обрабатываются рекурсивно. Если коллекция содержит регион, то в зависимости от переключателя **All** будут созданы новые объекты-полигоны; если она включает сложные полилинии, то из каждой отдельной полилинии будет создан отдельный объект; если в коллекции содержится группа точек, то будут созданы новые объекты-точки, по одной точке для каждого узла из группы. Все другие типы объектов, включая точки, линии, дуги, прямоугольники, скругленные прямоугольники и эллипсы, которые уже являются простыми объектами, этой операцией не изменяются.

Если регион содержит единственный полигон, то на выходе он останется без изменений. Если сложная полилиния содержит одну полилинию, то на выходе она останется без изменений. Если группа точек содержит единственный узел, то выходящий объект преобразуется в точечный объект, содержащий этот узел. Дуги, прямоугольники, скругленные прямоугольники, эллипсы на выходе остаются без изменений. Все другие типы объектов, например, текстовые не воспринимаются оператором **Objects Disaggregate** и вызовут ошибку.

Переключатель **Collection** разделяет только объекты коллекции. Если коллекция содержит регион, то этот регион на выходе станет новым объектом. Если объект коллекция содержит полилинию, то эта полилиния будет на выходе новым объектом. Если объект коллекция содержит группу точек, то эта группа точек будет на выходе новым объектом. В этом различие от опции, описанной в начале раздела, поскольку регион может содержать несколько полигонов, а сложная полилиния может содержать несколько полилиний. При использовании опции, описанной выше, группа точек не будет создана.

Переключатель **Collection** передает на выход без изменений все другие типы объектов, включая точки, группы точек, линии, дуги, регионы, прямоугольники, скругленные прямоугольники и эллипсы.

Если не задается **Into Table**, то в качестве таблицы для выходящих данных используется текущая редактируемая таблица. Входящие для обработки командой объекты берутся из текущей выборки.

Дополнительное предложение **Data** определяет, какие значения хранятся в колонках изменяемых объектов. Предложение **Data** может содержать список присваиваемых значений, разделенных точкой с запятой. Все значения, которые могут быть присвоены, описаны в таблице ниже:

| Выражение                                      | Эффект   |
|--|--|
| <code>col_name = col_name</code>               | Не изменяет величины, хранящиеся в колонке.  |
| <code>col_name = value</code>                  | Хранит указанные величины в колонке. Если колонка строковая, то значение тоже будет строковым; если колонка числовая, то значение будет числовым |
| <code>col_name = Proportion( col_name )</code> | Используется только для числовых колонок; уменьшает число, содержащееся в колонке, пропорционально удаленной площади объекта.                    |

Список **Data** может состоять из определений для всех колонок таблицы. Если предложение-список **Data** содержит назначения только для части колонок, пустые значения будут присвоены тем колонкам, которые не поименованы в предложении **Data**. Если Вы пропускаете предложение **Data** целиком, все колонки будут пустыми, содержащими нулевые значения для числовых колонок и пустые значения для строковых колонок.

**Пример:**

```
Open Table "STATES.TAB" Interactive
Map From STATES
Set Map Layer 1 Editable On
select * from STATES
Objects Disaggregate Into Table STATES
```

**См. также:**

[Оператор Create Object](#)

**Оператор Objects Enclose**

**Назначение**

Создает регионы в замкнутых областях, образованных полилиниями (соответствует команде **Замкнуть** в MapInfo Professional).

**Синтаксис**

```
Objects Enclose
[ Into Table tablename]
[ Region ]
```

*tablename* – строка с именем таблицы, в которой необходимо создать объекты.

### Описание

**Objects Enclose** создает объекты, представляющие "замыкание" полилиний. Новый регион создается для каждой замкнутой полигональной области. Входящие для обработки командой объекты берутся из текущей выборки. В отличие от случая, когда используется **Оператор Objects Combine**, **Objects Enclose** не будет удалять входные объекты-оригиналы. Объединения данных не происходит.

Дополнительный параметр **Region** позволяет использовать в операторе **Objects Enclose** замкнутые объекты (области, прямоугольники, скругленные прямоугольники и эллипсы). Для выполнения операции, исходные полигоны будут преобразованы в полилинии. Результат эквивалентен следующей последовательности действий: преобразованию всех замкнутых объектов в полилинии и последующему выполнению операции **Objects Enclose**. Все исходные объекты должны быть либо линейными, либо замкнутыми (например, точки, группы точек, коллекции и текст приведут к появлению ошибки). Если выбраны замкнутые объекты, а ключевое слово **Region** отсутствует, то такие объекты будут пропущены.

Оператор **Objects Enclose** соответствует команде меню MapInfo Professional – **Объекты > Замкнуть**. Описание выполняющейся операции смотрите в описании команды **ОБЪЕКТЫ > Замкнуть** в *Справочнике MapInfo Professional*. Если Вы в MapInfo Professional выполните команду **ОБЪЕКТЫ > Замкнуть** и при этом будет открыто окно MapBasic, то в протоколе выполненных действий будет использован оператор **Objects Enclose**.

С помощью дополнительного параметра **Into Table** можно сохранить созданные объекты в определенной таблице. Иначе, получаемые объекты будут храниться в той-же таблице, откуда берутся исходные объекты.

### Пример:

Будут выбраны все объекты таблицы testfile, выполнена операция **Objects Enclose**, а полученные результаты сохранены в таблице dump\_file.

```
select * from testfile  
Objects Enclose Into Table dump_file
```

### См. также:

**Оператор Objects Check**, **Оператор Objects Combine**

---

## Оператор Objects Erase

### Назначение

Удаляет часть изменяемого объекта, которая перекрывается другим объектом (или объектами). Оператор соответствует команде **ОБЪЕКТЫ > УДАЛИТЬ ЧАСТЬ** в MapInfo Professional.

### Синтаксис

**Objects Erase Into Target**

[ **Data** column\_name = expression [ , column\_name = expression ... ]

column\_name – строка с именем колонки изменяемой таблицы.

*expression* – выражение, которым задаётся то, что будет удалено из колонок *column\_name*.

Описание

Оператор **Objects Erase** удаляет часть объекта (или весь объект), который объявлен как изменяемый. Оператор **Objects Erase** соответствует команде MapInfo Professional **ОБЪЕКТЫ > УДАЛИТЬ ЧАСТЬ**. Описание команды **ОБЪЕКТЫ > УДАЛИТЬ ЧАСТЬ** ищите в *Справочнике MapInfo Professional*.

**Objects Erase** удаляет ту часть изменяемого объекта, которая перекрывает выбранный объект (или объекты). Если изменяемый объект перекрывается полностью, то он удаляется полностью. Если необходимо удалить часть не перекрытую выбранными объектами, то используйте **Оператор Objects Intersect**.

Перед выполнением оператора **Objects Erase** должен быть выбран изменяемый объект и один или более замкнутых объектов (типа "область ", "прямоугольник", "скругленный прямоугольник" или "эллипс"), играющих роль "ластика". Изменяемый объект может быть назначен командой **ОБЪЕКТЫ > ВЫБРАТЬ ИЗМЕНЯЕМЫЙ ОБЪЕКТ** или, используя **Оператор Set Target**, из прикладной программы.

Из каждого изменяемого объекта, будет создан единственный объект, состоящий из той части изменяемого объекта, которая не перекрывается "объектами-ластиком". Если изменяемый объект полностью лежит внутри "объектов-ластиков", то никакой объект не будет создан.

Дополнительное предложение **Data** определяет, какие значения хранятся в колонках изменяемых объектов. Предложение **Data** может содержать список присваиваемых значений, разделенных точкой с запятой.

Все значения, которые могут быть присвоены, описаны в таблице ниже:

| Выражение                                       | Эффект   |
|---|--|
| <i>col_name</i> = <i>col_name</i>               | MapBasic не изменяет величину, хранящуюся в колонке.   |
| <i>col_name</i> = <i>value</i>                  | MapBasic хранит указанные величины в колонке. Если тип колонки символьный, то <i>value</i> должно быть строкой. Если тип колонки числовой, то <i>value</i> должно быть числом.   |
| <i>col_name</i> = Proportion( <i>col_name</i> ) | Используется только для числовых колонок; MapBasic уменьшает число, содержащееся в колонке, пропорционально удаленной площади объекта. Так, если была удалена половина объекта, то значение в колонке уменьшится наполовину. |

Список **Data** может состоять из определений для всех колонок таблицы. Если предложение **Data** содержит назначения только для части колонок, пустые значения будут присвоены тем колонкам, которые не поименованы в предложении **Data**.

Если Вы пропускаете предложение **Data** целиком, все колонки будут пустыми, содержащими нулевые значения для числовых колонок и пустые значения для строковых колонок.

### Пример:

В следующем примере, в операторе **Objects Erase** не используется предложение **Data**. Все записи, к которым присоединены изменяемые объекты, теряют свои значения. В этом примере хотя бы один из объектов назначен изменяемым, а один или несколько объектов – выбраны.

```
Objects Erase Into Target
```

Следующий оператор **Objects Erase** имеет предложение **Data**, которое задает выражения для трех колонок ("State\_Name", "Pop\_1990" и "Med\_Inc\_80"). Этот оператор присваивает строку "остаток" колонке "State\_Name" и определяет, что значения в колонке "Pop\_1990" будут уменьшены пропорционально оставшейся после удаления площади. Значения в колонке "Med\_Inc\_80" сохраняются нетронутыми (какими были до применения оператора **Objects Erase**). Остальные колонки изменяемого объекта очищаются.

```
Objects Erase Into Target
```

```
Data
  State_Name = "area remaining",
  Pop_1990 = Proportion( Pop_1990 ),
  Med_Inc_80 = Med_Inc_80
```

См. также:

**Функция Erase( ), Оператор Objects Intersect**

---

## Оператор Objects Intersect

### Назначение

Удаляет часть изменяемого объекта, которая перекрывается другим объектом (или объектами). Оператор соответствует команде **ОБЪЕКТЫ > УДАЛИТЬ ВНЕШНЮЮ ЧАСТЬ** в MapInfo Professional.

### Синтаксис

```
Objects Intersect Into Target
```

```
[ Data column_name = expression [ , column_name = expression ... ] ]
```

*column\_name* – строка с именем колонки изменяемой таблицы.

*expression* – выражение, которым задаётся то, что будет удалено из колонок *column\_name*.

### Описание

Оператор **Objects Intersect** удаляет часть объекта (или объект полностью), который назначен как изменяемый. Оператор **Objects Intersect** соответствует команде MapInfo **ОБЪЕКТЫ > УДАЛИТЬ ВНЕШНЮЮ ЧАСТЬ**. Описание команды **ОБЪЕКТЫ > УДАЛИТЬ ВНЕШНЮЮ ЧАСТЬ** ищите в *Справочнике MapInfo Professional*.



Дополнительное предложение **Data** определяет, какие значения хранятся в колонках изменяемых объектов. Предложение **Data** может содержать список присваиваемых значений, разделенных точкой с запятой. Все значения, которые могут быть присвоены, описаны в таблице ниже:

| Выражение                                      | Эффект   |
|--|--|
| <code>col_name = col_name</code>               | MapBasic не изменяет величину, хранящуюся в колонке.   |
| <code>col_name = value</code>                  | MapBasic хранит указанные величины в колонке. Если тип колонки символьный, то value должно быть строкой; если тип колонки числовой, то value должно быть числом.   |
| <code>col_name = Proportion( col_name )</code> | Используется только для числовых колонок; MapBasic уменьшает число, содержащееся в колонке, пропорционально удаленной площади объекта. Так, если была удалена половина объекта, то значение в колонке уменьшится наполовину. |

Список **Data** может состоять из определений для всех колонок таблицы. Если предложение **Data** содержит назначения только для части колонок, пустые значения будут присвоены тем колонкам, которые не поименованы в предложении **Data**. Если Вы пропускаете предложение **Data** целиком, все колонки будут пустыми, содержащими нулевые значения для числовых колонок и пустые значения для строковых колонок.

Оператор **Objects Intersect** очень похож на **Оператор Objects Erase**, за одним важным исключением: **Objects Intersect** удаляет те части изменяемых объектов, которые не перекрыты выбранными объектами, а **Оператор Objects Erase** удаляет части изменяемых объектов. Из каждого изменяемого объекта, будет создан единственный объект, состоящий из той части изменяемого объекта, которая пресекается с "объектом-ластиком". Например, если изменяемый объект пересекается с тремя выбранными объектами, то будет создано три новых объекта. Части изменяемого объекта за пределами всех выбранных объектов будут удалены. Дополнительная информация в главе **Оператор Objects Erase на стр. 44**.

**Пример:**

```
Objects Intersect Into Target
Data
    Field2=Proportion(Field2)
```

**См. также:**

**Оператор Create Object, Функция Overlap( ), Оператор Objects Erase**

## Оператор Objects Move

### Назначение

Перемещает выбранные в исходной таблице объекты.

### Синтаксис

```
Objects Move  
  Angle angle  
  Distance distance  
  [ Units unit ]  
  [ Type { Spherical | Cartesian } ]
```

*angle* – величина, определяющая угол смещения выбранного объекта.

*distance* – число, определяющее расстояние смещения выбранного объекта.

*unit* – единицы измерения расстояний *distance*.

### Описание

**Objects Move** перемещает объекты в пределах исходной таблицы. Исходные объекты получены из текущей выборки. Результирующие объекты заменяют исходные объекты. Объединение данных не является необходимым условием и не осуществляется, так как данные, связанные с исходными объектами, являются неизменными.

Объект перемещен в направлении, заданном параметром *angle*, измеряемым от положительной оси X, указывающей восток (при этом угол измеряется против часовой стрелки), и смещен на расстояние, заданное параметром расстояния *distance*. Расстояние измеряется в единицах, указанных параметром *unit*, если он представлен. Если предложение **Units** пропущено, то текущая единица расстояния будет задана по умолчанию. По умолчанию, MapBasic измеряет расстояния в милях; для того чтобы задать другую единицу измерения, используйте **Оператор Set Distance Units**.

Дополнительная часть предложения **Type** позволяет задать тип расчета расстояния, используемого при смещении объектов. Если используется сферический тип – **Spherical**, то вычисления производятся в координатах “Широта/Долгота”, а расстояния рассчитываются на сфере. Если используется тип **Cartesian**, то вычисления производятся на плоскости, на которую спроектированные географические данные и расстояния рассчитываются по декартовым алгоритмам. Если часть предложения **Type** не задана, то используется сферический расчет расстояний. Если данные в проекции “Широта/Долгота”, то используется сферический тип расчетов независимо от настроек части предложения **Type**. Если данные представлены в проекции План-схема, используются декартовые вычисления независимо от настроек части предложения **Type**.

При проведении вычислений в сферической системе координат существуют некоторые соображения, неприменимые для декартовой системы координат. Так, при перемещении объекта по карте, построенной в проекции “Долгота/Широта”, его форма не изменится, но поменяется площадь. Это происходит потому, что один из параметров сдвига определяется в градусах, а реальное измеренное расстояние одного градуса в различных точках неодинаково.

Для функций сдвига реальная дельта переноса вычисляется в некоторой фиксированной точке объекта (например, в центре описанного прямоугольника), после чего это значение преобразуется из исходных единиц измерения в единицы измерения системы координат. Если система координат - Широта/Долгота, преобразование в градусы использует фиксированное число десятичных знаков. Действительное преобразованное расстояние может быть неодинаковым в различных точках объекта. Расстояние от исходного объекта до его перемещённой копии будет точным только в этой фиксированной точке.

### Пример:

```
Objects Move Angle 45 Distance 100 Units "mi" Type Spherical
```

### См. также:

**Оператор Objects Offset**

## Оператор Objects Offset

### Назначение

Копирует выбранные объекты, перемещая их

### Синтаксис

```
Objects Offset
  [ Into Table intotable ]
  Angle angle
  Distance distance
  [ Units unit ]
  [ Type { Spherical | Cartesian } ]
  [ Data column = expression [, column = expression ... ] ]
```

*intotable* – строка, определяющая таблицу, в которую будут скопированы новые значения.

*angle* – значение угла сдвига выбранных объектов.

*distance* – число, определяющее расстояние сдвига выбранного объекта.

*unit* – единицы измерения расстояний *distance*.

*column* – строка, определяющая колонку, которой выполняется сдвиг.

*expression* – выражение, с помощью которого вычисляется сдвиг относительно колонки.

### Описание

**Objects Offset** – копирует объекты со сдвигом относительно исходных. Исходные объекты получены из текущей выборки. Новые объекты будут сохранены в *intotable*, если присутствует предложение **Into**. Иначе объекты будут храниться в исходной таблице (например, базовой таблице выбранных объектов).

Объект перемещен в направлении, заданном параметром *angle*, измеряемым от положительной оси X, указывающей восток (при этом угол измеряется против часовой стрелки), и смещен на расстояние, заданное параметром расстояния *distance*. *Расстояние*

измеряется в единицах, указанных параметром *unit*, если он существует. Если предложение **Units** пропущено, то текущая единица расстояния будет задана по умолчанию. По умолчанию, MapBasic измеряет расстояния в милях; для того чтобы задать другую единицу измерения, используйте **Onepatop Set Distance Units**.

Дополнительная часть предложения **Type** позволяет задать тип расчета расстояния, используемого при смещении объектов. Если используется сферический тип – **Spherical**, то вычисления производятся в координатах “Широта/Долгота”, а расстояния рассчитываются на сфере. Если используется тип *Cartesian*, то вычисления производятся на плоскости, на которую спроектированные географические данные и расстояния рассчитываются по декартовым алгоритмам. Если часть предложения **Type** не задана, то используется сферический расчет расстояний. Если данные в проекции “Широта/Долгота”, то используется сферический тип расчетов независимо от настроек части предложения **Type**. Если данные представлены в проекции План-схема, используются декартовые вычисления независимо от настроек части предложения **Type**.

Если используется предложение **Data**, приложение выполнит объединение данных.

При проведении вычислений в сферической системе координат существуют некоторые соображения, неприменимые для декартовой системы координат. Так, при перемещении объекта по карте, построенной в проекции “Долгота/Широта”, его форма не изменится, но поменяется площадь. Это происходит потому, что один из параметров сдвига определяется в градусах, а реальное измеренное расстояние одного градуса в различных точках неодинаково.

Для функций сдвига реальная дельта переноса вычисляется в некоторой фиксированной точке объекта (например, в центре описанного прямоугольника), после чего это значение преобразуется из исходных единиц измерения в единицы измерения системы координат. Если система координат - Широта/Долгота, преобразование в градусы использует фиксированное число десятичных знаков. Действительное преобразованное расстояние может быть неодинаковым в различных точках объекта. Расстояние от исходного объекта до его перемещённой копии будет точным только в этой фиксированной точке.

### Пример:

```
Objects Offset Into Table c:\temp\table1.tbl Angle 45 Distance 100 Units  
"mi" Type Spherical
```

### См. также:

**Функция Offset( )**

---

## Оператор Objects Overlay

### Назначение

Добавляет узлы изменяемому объекту в точках пересечения линий или контуров выбранных объектов. Оператор соответствует команде **ОБЪЕКТЫ > ДОБАВИТЬ УЗЛЫ**.

### Синтаксис

```
Objects Overlay Into Target
```

## Описание

Перед выполнением оператора **Objects Overlay** должен быть назначен изменяемый объект и один или более объектов любого типа, кроме текстового или точечного. Изменяемый объект может быть назначен командой **ОБЪЕКТЫ > ВЫБРАТЬ ИЗМЕНЯЕМЫЙ ОБЪЕКТ** или, используя **Оператор Set Target**, из прикладной программы. Более подробная информация приводится в описании команды в Справочнике *MapInfo Professional*.

См. также:

**Функция OverlayNodes( ), Оператор Set Target**

## Оператор Objects Pline

### Назначение

Разделяет полилинию на две полилинии

### Синтаксис

```
Objects Pline Split At Node index
  [ Into Table name ]
  [ Data column_name = expression [ , column_name = expression ... ] ]
```

*index* – целое, с номером узла, в котором необходимо разрезать линию.

*name* – строка с именем таблицы, в которой будут храниться новые объекты.

*column\_name* – строка с именем колонки, в которой будут храниться значения новых объектов.

*expression* – выражение, которое будет использоваться для создания значений в колонке *column\_name*.

### Описание

Если объект является полилинией, состоящей из единственного сегмента, то в таблице будут созданы две новые полилинии, состоящие из единственного сегмента. **Индекс узла** (Node) должен быть корректным MapBasic индексом для разделяемой полилинии. Если **узел** (Node) является начальным или конечным узлом полилинии, операция деления не будет выполнена и будет показано сообщение об ошибке.

Дополнительное предложение **Data** определяет, какие значения хранятся в колонках объектов. Предложение **Data** может содержать список присваиваемых значений, разделенных запятыми. Все значения, которые могут быть присвоены, описаны в таблице ниже:

| Выражение                        | Эффект  |
|----------------------------------|---|
| <code>col_name = col_name</code> | Не изменяет величины, хранящиеся в колонке.   |
| <code>col_name = value</code>    | Хранит указанные величины в колонке. Если тип колонки символьный, то value должно быть строкой; если тип колонки числовой, то value должно быть числом. |

Список **Data** может состоять из определений для всех колонок таблицы. Если предложение **Data** содержит назначения только для части колонок, пустые значения будут присвоены тем колонкам, которые не поименованы в предложении **Data**.

Если Вы пропускаете предложение **Data** целиком, все колонки будут пустыми, содержащими нулевые значения для числовых колонок и пустые значения для строковых колонок.

**Пример:**

В следующем примере, выбранная полилиния будет разрезана в заданном узле (индекс узла 12). Значения исходных записей выбранной полилинии будут записаны в новые записи разрезанной полилинии без изменений.

```
Objects Pline Split At Node 12 Into Table WORLD Data
Country=Country,Capital=Capital,Continent=Continent,Numeric_code=Numeric_
code,FIPS=FIPS,ISO_2=ISO_2,ISO_3=ISO_3,Pop_1994=Pop_1994,Pop_Grw_Rt=Pop_G
rw_Rt,Pop_Male=Pop_Male,Pop_Fem=Pop_Fem...
```

**См. также:**

[Функция ObjectLen\( \)](#), [Функция ObjectNodeX\( \)](#), [Функция ObjectNodeY\( \)](#), [Оператор Objects Disaggregate](#)

Оператор Objects Snap

**Назначение**

Выполняет коррекцию объектов данной таблицы и осуществляет различные топологические операции над объектами, включая совмещение узлов разных объектов, прилегающих друг к другу и генерализацию/разреживание узлов. Таблица может быть таблицей выборки (Selection). Все объекты, подвергающиеся действию оператора, должны быть или все линейные (то есть, полилинии или дуги), или все замкнутые (то есть, полигоны, прямоугольники, скругленные прямоугольники или эллипсы). Смешанные линейные и замкнутые объекты не могут обрабатываться этим оператором, будет выдано сообщение об ошибке.

**Синтаксис**

```
Objects Snap From tablename
[ Tolerance [ Node node_distance ][ Vector vector_distance ]
  [ Units unit_string] ]
[ Thin [ Bend bend_distance ][ Distance spacing_distance ]
```

```
[ Units unit_string ]]
[ Cull Area cull_area [ Units unit_string ]]
```

*tablename* – строка с именем таблицы, объекты которой должны быть проверены.

*node\_distance* – численное значение радиуса окружностей вокруг концов полилинии.

*vector\_distance* – численное значение радиуса окружностей вокруг промежуточных узлов полилиний.

*bend\_distance* – численное значение допуска коллинеарности серии узлов.

*spacing\_distance* – численное значение минимального расстояния между узлами, которое будет использовано при удалении лишних узлов.

*unit\_string* – строка с названием единиц измерения расстояний.

*cull\_area* – численное значение пороговой площади полигонов, которые будут отсортированы.

*unit\_string* – строка с названием единиц измерения площадей.

## Описание

Объекты из исходной таблицы *tablename* проверяются на предмет различных проблем и несоответствий, таких, как самопересечения. Самопересекающиеся полигоны в форме "восьмерки" будут превращены в два полигона с общей вершиной. Полигоны, содержащие острые выступы подвергаются обработке, при которой часть таких выступов удаляется. Исправленный объект заменит исходный. Если существуют наложения полигонов друг на друга, то такие наложения будут удалены. Удаление взаимных перекрытий обычно состоит из вырезания перекрывающей части из одного объекта, в другом объекте эта часть остается. Область перекрытия состоит из нескольких полигонов. Один полигон представляет собой часть, которая не была перекрыта, а отдельный полигон – каждое перекрытие.

Значения **допусков Node** и **Vector** предназначены для совмещения узлов из разных близлежащих объектов и удаляют мелкие пустоты и мелкие пустоты между двумя объектами. Субпредложение **Units** предложения **Tolerances** позволяет задать единицы измерения расстояния (например, "km" для километров), применяемых к значениям **Node** и **Vector**. Если субпредложение **Units** отсутствует, то значения **Node** и **Vector** будут интерпретироваться в текущих значениях измерения расстояния MapBasic. По умолчанию, MapBasic измеряет расстояния в милях; для того чтобы задать другую единицу измерения, используйте **Оператор Set Distance Units**.

Допуск **Node** – это окружности заданного радиуса вокруг конечных узлов полилиний. Если узлы из других объектов попадают в этот радиус, то один или два узла перемещаются так, что попадают в одну точку (т.е. они совмещаются).

Допуск **Vector** - это окружности, заданного радиуса, вокруг промежуточных узлов полилиний. Его назначение тоже, что и у радиуса **Node**, кроме того, что он используется только для промежуточных точек полилиний. Обратите внимание, что для полигонов не определено понятие конечных точек (значение Node не используется) ввиду их замкнутости. Для них используется только величина **Vector**, которая применяется ко всем узлам объекта. Значение допуска **Node** не применяется к полигонам. Для полилиний значение допуска **Node** должно быть больше или равно значению **Vector**.

Значения **Bend** и **Distance** могут использоваться для удобства операций разреживания узлов и обобщения контуров. Они уменьшают число узлов, используемых в объекте, сохраняя основные черты формы объекта. Субпредложение **Units** предложения **Thin** позволяет указать имя единицы измерения расстояния (например, "km" для километров) в которых измеряется значение **Bend** и **Distance**. Если субпредложение **Units** отсутствует, то значения **Bend** и **Distance** будут интерпретироваться в текущих единицах измерения MapBasic.

Значение **Bend** используется для управления коллинеарным отклонением группы из 3 последовательных узлов. Эти 3 узла связываются в треугольник. Измеряется перпендикуляр, опущенный из средней точки на длинную сторону треугольника. Если это расстояние меньше значения **Bend**, то эти три узла рассматриваются как коллинеарные, и второй (средний) узел удаляется из объекта.

Расстояние **Distance** используется для удаления узлов из одного объекта, если узлы расположены слишком близко друг к другу. Измеряется расстояние между двумя соседними точками объекта. Если это расстояние меньше, чем **Distance**, то один из двух узлов будет удален.

Значение **Cull Area** используется для удаления избыточных полигонов, площадь которых меньше некоторого заданного значения. Субпредложение **Units** из предложения **Cull** позволяет настроить единицы измерения площади (например, "кв. км" для квадратных километров), применяемые в значении **Area**. Если подпредложение **Unit** не задано, то площадь брешей **Area** измеряется в текущих единицах MapBasic. По умолчанию, MapBasic измеряет площади в кв. милях; для того чтобы задать другую единицу измерения, используйте **Оператор Set Area Units**.

**Внимание:** Для всех расстояний и площадей, упомянутых выше, всегда используется тип измерений на плоскости. Систему координат и проекцию всегда надо учитывать. Вычисления расстояний и площадей в Долготе/ Широте на плоскости осуществляются с невысокой точностью. Перед тем как использовать этот оператор, убедитесь, что Вы работаете в подходящей системе координат (декартовой).

### Пример:

```
Open Table "STATES.TAB" Interactive
Map From STATES
Set Map Layer 1 Editable On
select * from STATES
Objects Snap From Selection Tolerance Node 3 Vector 3 Units "mi" Thin Bend
0.5 Distance 1 Units "mi" Cull Area 10 Units "sq mi"
```

**См. также:**

**Оператор Create Object, Функция Overlap( )**



## Оператор Objects Split

### Назначение

Разделяет изменяемые объекты на части, используя форму выбранных объектов как "ластик". Оператор соответствует команде **ОБЪЕКТЫ > РАЗРЕЗАТЬ**.

### Синтаксис

#### Objects Split Into Target

```
[ Data column_name = expression [ , column_name = expression ... ]
```

*column\_name* – строка с именем колонки, в которой будут храниться значения новых объектов.

*expression* – выражение, которое будет использоваться для создания значений в колонке *column\_name*.

### Описание

Оператор **Objects Split** разрезает каждый изменяемый объект на несколько. Оператор **Objects Split** соответствует команде MapInfo **ОБЪЕКТЫ > РАЗРЕЗАТЬ**. Более подробная информация о разрезах приводится в описании команды в *Справочнике MapInfo Professional*.

Перед выполнением оператора **Objects Split** должен быть выбран изменяемый объект и один или более замкнутых объектов (типа "область", "прямоугольник", "скругленный прямоугольник" или "эллипс"), играющих роль "ластика". Изменяемый объект может быть назначен командой **ОБЪЕКТЫ > ВЫБРАТЬ ИЗМЕНЯЕМЫЙ ОБЪЕКТ** или, используя **Оператор Set Target**, из прикладной программы.

Из каждого изменяемого объекта будет создан единственный объект, состоящий из той части изменяемого объекта, которая пресекается с "объектом-ластиком". Например, если изменяемый объект пересекается с тремя выбранными объектами, то будет создано три новых объекта. Кроме того, из всех частей изменяемого объекта, не попадающих внутрь "ластика", будет создан единственный объект. Это эквивалентно выполнению последовательности операций: **Оператор Objects Erase** и **Оператор Objects Intersect** (**Объекты > Удалить внешнюю часть**).

Дополнительное предложение **Data** определяет, какие значения хранятся в колонках изменяемых объектов. Предложение **Data** может содержать список присваиваемых значений, разделенных точкой с запятой. Все значения, которые могут быть присвоены, описаны в таблице ниже:

| Выражение                                      | Эффект   |
|--|--|
| <code>col_name = col_name</code>               | Содержимое колонки не меняется; каждый объект, полученный в результате, в своей записи имеет то же значение, что и объект, из которого он был получен.   |
| <code>col_name = value</code>                  | MapBasic хранит указанные величины в колонке. Если тип колонки символьный, то value должно быть строкой; если тип колонки числовой, то value должно быть числом. Каждый полученный после разреза объект получит заданное значение.                                     |
| <code>col_name = Proportion( col_name )</code> | Используется только для числовых колонок; MapInfo Professional разделяет значение в колонке объектов по графическим объектам, полученным после разреза. Каждому объекту будет сопоставлено значение пропорциональное площади полученного объекта и исходному значению. |

Список **Data** может состоять из определений для всех колонок таблицы. Если предложение **Data** содержит назначения только для части колонок, пустые значения будут присвоены тем колонкам, которые не поименованы в предложении **Data**.

Если Вы пропускаете предложение **Data** целиком, все колонки будут пустыми, содержащими нулевые значения для числовых колонок и пустые значения для строковых колонок.

**Пример:**

В следующем примере, в операторе **Objects Split** не используется предложение **Data**. Все записи, к которым присоединены изменяемые объекты, теряют свои значения.

Objects Split Into Target

В следующем операторе используется предложение **Data**, которое задает выражения для трех колонок ("State\_Name", "Pop\_1990" и "Med\_Inc\_80"). Первая часть предложения **Data** присваивает строку "подразделение" колонке "State\_Name"; то есть строка "подразделение" будет помещена в колонку "State\_Name" для каждого объекта, являющегося результатом разрезания. Далее в предложении **Data** определяется, что население из колонки "Pop\_1990" пропорционально разделяется между объектами, полученными после разреза. В третьей части параметра **Data** задано, что значение "Med\_Inc\_80" сохраняется для всех объектов.

Objects Split Into Target  
Data  
State\_Name = "sub-division",  
Pop\_1990 = Proportion( Pop\_1990 ),  
Med\_Inc\_80 = Med\_Inc\_80

См. также:

**Оператор Alter Object**

## Функция Offset( )

### Назначение

Возвращает копию исходного объекта, смещённого на определенную дистанцию и угол.

### Синтаксис

**SphericalOffset**( *object*, *angle*, *distance*, *units* ), где

*object* - объект, подвергаемый сдвигу.

*angle* - угол перемещения объекта.

*distance* – число, определяющее расстояние сдвига выбранного объекта.

*units* – строка, представляющая единицы измерения расстояния *distance*.

### Возвращаемая величина

Объект

### Описание

**Offset( )** создаёт новый объект, являющийся копией исходного объекта, смещённой на определённое *расстояние* и повернутой на заданный *угол* в градусах относительно горизонтали (это нулевой угол, положительные значения отсчитываются против часовой стрелки). Строковое значение *units*, похоже на то, которое используется для операторов **Функция ObjectLen( )** или **Функция Perimeter( )** - это единицы измерения *расстояния*. Для параметра *distance* применяется способ расчета на сфере кроме тех случаев, когда используется план-схема. Для план-схемы автоматически используется декартовый алгоритм расчета расстояний. Используется координатная система исходного объекта.

При проведении вычислений в сферической системе координат существуют некоторые соображения, неприменимые для декартовой системы координат. Так, при перемещении объекта по карте, построенной в проекции “Долгота/Широта”, его форма не изменится, но поменяется площадь. Это происходит потому, что один из параметров сдвига определяется в градусах, а реальное измеренное расстояние одного градуса в различных точках неодинаково.

Для функций сдвига реальная дельта переноса вычисляется в некоторой фиксированной точке объекта (например, в центре описанного прямоугольника), после чего это значение преобразуется из исходных единиц измерения в единицы измерения системы координат. Если система координат - Широта/Долгота, преобразование в градусы использует фиксированное число десятичных знаков. Действительное преобразованное расстояние может быть неодинаковым в различных точках объекта. Расстояние от исходного объекта до его перемещённой копии будет точным только в этой фиксированной точке.

### Пример:

```
Offset(Rect, 45, 100, "mi")
```

### См. также:

[Оператор Objects Offset](#), [Функция OffsetXY\( \)](#)

---

## Функция OffsetXY( )

### Назначение

Возвращает копию исходного объекта, смещённую на заданное расстояние по X и Y.

### Синтаксис

```
OffsetXY( object, xoffset, yoffset, units )
```

*object* - объект, подвергаемый сдвигу.

*xoffset* и *yoffset* – числа, определяющие расстояния вдоль осей x и y, на которые будет смещен объект;

*units* - строка, определяющая единицы измерения расстояний.

### Возвращаемая величина

Объект

### Описание

**OffsetXY( )** создает новый объект *object* который является копией исходного объекта, смещенного на расстояние *xoffset* вдоль оси X и на расстояние *yoffset* вдоль оси Y. Строковое значение *units*, похоже на то, которое используется для операторов [Функция ObjectLen\( \)](#) или [Функция Perimeter\( \)](#) - это единицы измерения расстояния. Для параметра *distance* применяется способ расчета на сфере кроме тех случаев, когда используется план-схема. Для план-схемы автоматически используется декартовый алгоритм расчета расстояний. Используется координатная система исходного объекта.

При проведении вычислений в сферической системе координат существуют некоторые соображения, неприменимые для декартовой системы координат. Так, при перемещении объекта по карте, построенной в проекции “Долгота/Широта”, его форма не изменится, но поменяется площадь. Это происходит потому, что один из параметров сдвига определяется в градусах, а реальное измеренное расстояние одного градуса в различных точках неодинаково.

Для функций сдвига реальная дельта переноса вычисляется в некоторой фиксированной точке объекта (например, в центре описанного прямоугольника), после чего это значение преобразуется из исходных единиц измерения в единицы измерения системы координат. Если система координат - Широта/Долгота, преобразование в градусы использует фиксированное число десятичных знаков. Действительное преобразованное расстояние может быть неодинаковым в различных точках объекта. Расстояние от исходного объекта до его перемещённой копии будет точным только в этой фиксированной точке.

**Пример:**

```
OffsetXY(Rect, 92, -22, "mi")
```

**См. также:**

**Функция Offset( )**

---

## Оператор OnError

**Назначение**

Обеспечивает выполнение процедуры-обработчика ошибок.

**Синтаксис**

```
OnError Goto { label | 0 }
```

*label* – строка с меткой в тексте некоторой подпрограммы или функции.

**Предупреждение**

Вы не можете использовать оператор **OnError** в окне MapBasic.

**Описание**

Оператор **OnError** используется либо для запуска процедуры-обработчика ошибок, если ошибка имела место, либо для отмены обработки ошибок (форма **OnError Goto 0**). (Процедура-обработчик ошибок представляет собой группу операторов, которые выполняются в случае ошибки).

В отличие от стандартных версий BASIC оператор **OnError** в MapBasic пишется в одно слово.

Оператор **OnError Goto label** объявляет, что операторы после метки *label* являются обработчиком ошибок. Если один из операторов, следующих за **OnError**, вернет код ошибки, то MapBasic передаст выполнение программы метке *label*. Предполагается, что операторы, выполняемые по метке *label*, должны обработать конфликт, возникший в результате ошибки, так, чтобы он не повлиял на корректность выполнения программы, или предупреждали пользователя о случившейся ошибке, или то и другое. В процедуре-обработчике ошибок используйте **Resume statement**, чтобы возобновить выполнение программы.

Заметим, что если Ваша программа имеет обработчик ошибок, то Вы должны перед оператором с меткой *label* расположить оператор управления выполнением программы (например, **Оператор Exit Sub** или **Оператор End Program**). Это не позволяет программе передать управление процедуре-обработчику без наличия ошибки. Смотрите пример ниже.

Оператор **OnError Goto 0** отменяет выполнение установленной до этого процедуры-обработчика ошибок. Если ошибка происходит в программе, где нет обработчика ошибок или он отменен, то MapBasic выводит на экран окно сообщения об ошибке и прекращает выполнение программы.

Операторы обработчика ошибок могут располагаться в отдельной процедуре или функции. Так, подпрограмма может запускать процедуру-обработчик ошибок следующим оператором:

```
OnError Goto recover
```

(при этом подразумевается, что в этой процедуре есть метка "recover"). Если после выполнения такого оператора **OnError** процедура выполнит **Оператор Call** и перейдет в другую подпрограмму, то процедура-обработчик по метке "recover" не будет способна реагировать на ошибку, пока действует другая процедура, которую вызывает. Это происходит потому, что каждая метка локальна по отношению к процедуре и функции, в которой она задана. Этот прием позволяет каждой функции и каждой процедуре сопоставить собственный обработчик ошибок.

**Внимание:** Если ошибка возникнет в процессе обработки других ошибок, то выполнение программы прекратится.

### Пример:

```
OnError GoTo no_states
Open Table "states"

OnError GoTo no_cities
Open Table "cities"

Map From cities, states

after_mapfrom:
    OnError GoTo 0
    '
    ' ...
    '
End Program

no_states:
    Note "Could not open table States... no Map used."
    Resume after_mapfrom

no_cities:
    Note "City data not available..."
    Map From states
    Resume after_mapfrom
```

### См. также:

**Функция Err( ), Оператор Error, Функция Error\$( ), Resume statement**

---

## Оператор Open Connection

### Назначение

Устанавливает соединение с внешней службой геокодирования или маршрутизации сервера MapMarker или Envinsa.

## Синтаксис

```
Open Connection ServiceGeocode [ MapMarker | Envinsa ] | Isogram URL
URLstring [ User name_string [ Password pwd_string ] ] [ Interactive [ On
| Off ] ] into variable var_name
```

*URLString* – строка с корректным адресом URL. *URLString* – должен корректно вести к серверу маршрутизации, если задано построение изограммы (**Isogram**), или к серверу геокодирования если выполняется операция геокодирования (**Geocode**).

*name\_string* – строка с именем пользователя Envinsa или MapMarker.

*pwd\_string* – строка с паролем, соответствующим *user\_name*.

*var\_name* – строка с названием переменной, в которой будет храниться номер соединения.

## Описание

Оператор **Open Connection** создаёт соединение с сервером геокодирования или маршрутизации. Каждый оператор должен содержать информацию о сервисе и провайдере соединения с которым устанавливается. Так как служба маршрутизации представляется только сервером Envinsa, другой провайдер указан быть не может. Если устанавливается соединение со службой геокодирования и провайдер не указан, по умолчанию предполагается, что это **Envinsa**.

Ключевому слову **Into variable** требуется переменная *var\_name*, содержащая возвращённый номер соединения, который затем передаётся в другие операторы, такие как **Оператор Set Connection Geocode**, **Оператор Geocode**, **Оператор Set Connection Isogram**, и **Оператор Create Object Isogram**.

**Interactive** - определяет отображется ли диалог имя пользователя/пароль, в случае, если авторизация не прошла успешно. Если **Interactive** в режиме **Off**, диалог не отображается и команда не выполняется, если авторизация не прошла успешно. Если **Interactive** включен, т.е находится в режиме **On**, в случае неудачи с авторизацией диалог отображается.

Значение по умолчанию для этой команды **Interactive Off**. Таким образом, если ключевое слово **Interactive** не используется, это эквивалентно установке параметра **Interactive Off**. Однако, если параметр **Interactive** указан, это равнозначно команде **Interactive On**.

## Примеры

**Внимание:** Все примеры без указания ключевого слова **MapMarker** подразумевают использование сервера Envinsa.

Следующий пример открывает соединение с сервером геокодирования без указания ключевого слова **Interactive**. **Interactive** установлен **Off** по умолчанию.

```
Open Connection Into Variable CnctNum Service Geocode URL
"http://EnvinsaServices/LocationUtility/services/LocationUtility"
```

Этот пример открывает соединение с сервером геокодирования с указанием ключевого слова **Interactive** с параметром **On**.

```
Open Connection Into Variable CnctNum Service Geocode URL
"http://EnvinsaServices/LocationUtility/services/LocationUtility"
```

В этом примере открывается соединение с сервером геокодирования требующим авторизации.

```
dim baseURLVariable as StringbaseURLVariable =  
"http://EnvinsaServices/"Open Connection Service Geocode URL  
baseURLVariable + "LocationUtility/services/LocationUtility" User  
"geocodeuser" Password "GeoMe" Into Variable CnctNum
```

Этот пример открывает соединение с сервером маршрутизации требующем авторизации.

```
dim baseURLVariable as StringbaseURLVariable =  
"http://EnvinsaServices/"Open Connection Service IsoGram URL  
baseURLVariable + "Route/services/Route" User "isogramuser" Password  
"ISOMe" Into Variable CnctNum
```

**См. также:**

[Оператор Close Connection](#), [Оператор Set Connection Geocode](#), [Оператор Set Connection Isogram](#)

---

## Оператор Open File

### Назначение

Открывает файл для ввода/вывода.

### Синтаксис

```
Open File filespec  
[ For { Input | Output | Append | Random | Binary } ]  
[ Access { Read | Write | Read Write } ]  
As [ # ] filenum  
[ Len = recordlength ]  
[ ByteOrder { LOWHIGH | HIGHLOW } ]  
[ CharSet char_set ]
```

*filespec* – строка, содержащая имя файла;

*filenum* – целочисленный номер, который будет присвоен файлу вплоть до завершения работы с ним (например, [Оператор Get](#) или [Оператор Put](#));

*recordlength* – число символов в одной записи (включая символ конца строки) для доступа к файлу в режиме Random;

*char\_set* – имя набора символов; см. раздел, в котором описан [Предложение CharSet on page 136](#);

### Предупреждение

Вы не можете использовать оператор **Open File** в окне MapBasic.

### Описание

Оператор **Open File** открывает текстовый файл для операций ввода/вывода прикладной программой.



MapBasic может считывать данные из файла или записывать в него только после открытия файла. Для операций ввода/вывода таблиц используется один набор операторов (например, **Оператор Open Table**, **Оператор Fetch** и **Оператор Select**), а для файлов других типов другой набор операторов (например, **Open File**, **Оператор Get**, **Оператор Put**, **Оператор Input #** и **Оператор Print #**).

Предложение **For** задает режим доступа к данным файла: режим последовательного доступа, режим произвольного доступа или режим бинарного доступа. Типы ввода/вывода описан ниже. Если предложения **For** нет в операторе, то для открытия файла используется режим **произвольного доступа**.

### Файлы последовательного доступа

Если Вы собираетесь читать текст из файла, записи которого имеют разную длину (например, одна строка имеет 55 символов, а следующая – 72 и т. д.), то Вам надо использовать режим последовательного доступа. Для задания этого режима в предложении **For** используется ключевое слово **Input**, **Output** или **Append**.

Если использовано предложение **For Input**, то для чтения Вы можете использовать **Оператор Input #** и оператор **Line Input #**.

Если использовано предложение **For Output** или предложение **For Append**, то для записи в файл Вы можете использовать **Оператор Print #** и **Оператор Write #**.

Если Вы используете предложение **For Input**, то в предложении **Access** Вы можете использовать только ключевое слово **Read**. Аналогично, с предложением **For Output** может использоваться в предложении **Access** только ключевое слово **Write**.

Предложение **Len** не должно использоваться в операторе, если задается режим последовательного доступа.

### Файлы произвольного доступа

Если Вы собираетесь читать текст из файла, записи которого имеют одинаковую длину (например, каждая строка по 80 символов длиной), то Вы должны открыть файл в режиме **произвольного доступа**. Для задания этого режима используется параметр **For Random**.

Для режима **произвольного доступа** необходимо задать длину записи в предложении **Len = recordlength**. Величина в параметре *recordlength* должна задать количество символов одной записи, включая символы конца записи, например, пара символов "возврат каретки" и "новая строка".

Для режима **произвольного доступа** в предложении **Access** Вы можете использовать все комбинации ключевых слов: **Read**, **Write** или **Read Write**. Для чтения из файла и записи в файл, открытый в режиме **произвольного доступа**, используются: **Оператор Get** и **Оператор Put**.

### Двоичные файлы

Если файл открыт в **бинарном** режиме, то MapBasic конвертирует величину переменной MapBasic в бинарную величину в случае записи и наоборот в случае чтения. Хранение числовых данных в бинарном файле более компактно, чем хранение бинарных данных в текстовом файле. Но бинарный файл нельзя показывать и распечатывать как текстовый.

Для открытия файла в бинарном режиме используется предложение **For Binary**.

Для режима **бинарного** доступа в предложении **Access** Вы можете использовать все комбинации ключевых слов: **Read**, **Write** или **Read Write**. Для чтения из файла и записи в файл, открытый в бинарном режиме, используются: **Оператор Get** и **Оператор Put**.

Предложение **Len** и **Предложение CharSet** не должны использоваться в операторе, если задается режим бинарного доступа.

### Управление работой с файлом

Предложение **CharSet** определяет набор символов. Параметр *char\_set* должен быть строковой константой, такой как "MacRoman" или "WindowsLatin1". Если предложение CharSet опущено, MapInfo Professional будет использовать заданный по умолчанию набор символов. Заметим, что используется только в том случае, если файл открывается в режимах **Input**, **Output**, или **Random**. Более подробную информацию см. в разделе **Предложение CharSet on page 136**.

Если Вы открыли файл в режиме произвольного или бинарного доступа (**Random** или **Binary**), предложение **ByteOrder** задает как число представлено в файле.

Если прикладная программа действует только в пределах одной вычислительной платформы, то Вам не надо беспокоиться о порядке разрядов байтов в файле. Но, если Вам необходимо читать из бинарного файла или писать в бинарный файл, который был создан или будет использоваться в другой платформе, то Вам придется контролировать порядок расположения байтов с помощью предложения **ByteOrder**.

### Примеры

```
Open File "cxdata.txt" For INPUT As #1
Open File "cydata.txt" For RANDOM As #2 Len=42
Open File "czdata.bin" For BINARY As #3
```

**См. также:**

**Оператор Close File, Функция EOF( ), Оператор Get, Оператор Input #, Оператор Open Table, Оператор Print #, Оператор Put, Оператор Write #, Предложение CharSet**

---

## Оператор Open Report

### Назначение

Загружает отчет в модуль Crystal Report Designer.

**Синтаксис**

**Open Report** *reportfilespec*

*reportfilespec* - полный путь и имя существующего файла отчета.

**См. также:**

**Оператор Create Report From Table**

---

## Оператор Open Table

**Назначение**

Открывает таблицу MapInfo Professional для ввода/вывода.

**Синтаксис**

```
Open Table filename [ As tablename ]
    [ Hide ] [ ReadOnly ] [ Interactive ] [ Password pwd ]
    [ NoIndex ] [ View Automatic ] [ DenyWrite ]
    [ VMGrid | VMRaster | VMDefault ]
```

*filename* – строка, задающая таблицу MapInfo, которую требуется открыть.

*tablename* – строка с "синонимом" имени, под которым открывается таблица;

*pwd* – строка, представляющая собой пароль на уровне базы данных, определяемый при включении защиты базы данных; Используется только для таблиц Access.

**VMGrid** рассматривает при открытии файлы Vertical Mapper GRD как слои поверхности.

**VMRaster** рассматривает при открытии файлы Vertical Mapper GRD как растровые слои.

**VMDefault** рассматривает при открытии GRD-файлы либо как растровые слои, либо как слои-поверхности, в зависимости от наличия в .TAB-файле тега RasterStyle 6 1.

**Описание**

Оператор **Open Table** открывает уже существующую таблицу. Эффект от этого оператора такой же, как и от команды MapInfo **ФАЙЛ > ОТКРЫТЬ ТАБЛИЦУ**. Таблица должна быть открыта командой или оператором, прежде чем MapInfo Professional сможет производить какие-либо действия с таблицей.

**Внимание:** Имя файла, который будет открыт (определяемый параметром *filespec*) должен соответствовать таблице, которая уже существует; для создания новой таблицы используется **Оператор Create Table**. Заметим так же, что оператор **Open Table** применяется только для таблиц MapInfo; для использования файлов других форматов, используйте **Оператор Register Table** и **Оператор Open File**.

Если оператор включает предложение **As**, MapInfo Professional открывает таблицу под именем, задаваемым параметром *tablename*, которое мы будем называть "синонимом" или "псевдонимом" таблицы. Это влияет на то, как имя таблицы будет показано в списках, например, в том списке, который появляется при выполнении команды **Файл > Закреть**. Более того, если оператор **Open Table** задает синоним имени таблицы, все следующие

операции с таблицами, выполняемыми MapBasic'ом (например, **Оператор Close Table**) должны использовать синоним имени таблицы, а не постоянное имя таблицы. Синоним имени далее представляет эту таблицу во всех списках MapInfo. Назначение имени-синонима не имеет ничего общего с операцией переименования таблицы.

Если оператор включает предложение **Hide**, то имя таблицы не появится ни в каком диалоге, показывающем список открытых таблиц (например, в диалоге **ФАЙЛ > ЗАКРЫТЬ ТАБЛИЦУ**). Используйте предложение **Hide** если Вам надо открыть таблицу, которая останется скрытой от пользователя. Если оператор включает предложение **ReadOnly**, пользователь не сможет редактировать таблицу.

Дополнительное ключевое слово **Interactive** дает команду MapBasic подсказать пользователю место нахождения таблицы, если она не найдена по указанному пути. Ключевое слово **Interactive** полезно в ситуации, когда Вы не знаете местонахождения нужных файлов. Если оператор включает ключевое слово **NoIndex**, индекс MapInfo не будет встраиваться в таблицу MS Access при ее открытии.

**View Automatic** – дополнительное предложение оператора **Open Table**, с помощью которого можно запустить в текущем сеансе MapInfo Professional ассоциированные с геолинком либо таблицу MapInfo, либо рабочий набор, либо файл программы, в том числе, начать новый сеанс, если ни одного не запущено. При наличии параметра **View Automatic**, MapInfo Professional либо добавит эти данные в существующее окно, либо откроет новое окно карты или списка. Это особенно полезно для геолинков.

**DenyWrite** – дополнительное предложение параметра только для таблиц MS Access; если задан этот параметр, другие пользователи не смогут редактировать эту таблицу. Если же другой пользователь уже редактирует такую таблицу, то оператор **Open Table** не будет выполнен.

### Открытие двух таблиц с одинаковым именем

MapInfo Professional может открыть две отдельные таблицы, которые имеют одно и тоже имя. В этом случае, MapInfo Professional должно открыть вторую таблицу под специальным именем, что бы избежать конфликтов. В зависимости от того, включает ли оператор **Open Table** ключевое слово **Interactive**, MapBasic или присваивает специальное имя таблице автоматически, или показывает диалог, позволяющий пользователю интерактивно выбрать специальное имя таблицы.

Например, пользователь может хранить две копии таблицы "Sites", одну копию в директории 2006 ("C:\2006\SITES.TAB") и другую, возможно более новую, в другой директории ("C:\2007\SITES.TAB"). Когда пользователь (или приложение) открывает первую таблицу Sites, MapInfo открывает таблицу под ее именем ("Sites"). Если приложение использует оператор **Open Table** для открытия второй таблицы Sites, MapInfo автоматически открывает вторую таблицу под измененным именем (например, "Sites\_2") что бы отличать ее от первой таблицы. С другой стороны, если оператор **Open Table** включает предложение **Interactive**, MapInfo откроет диалог, позволяющий пользователю выбрать для таблицы альтернативное имя.

Не смотря на то, использует ли оператор **Open Table** ключевое слово **Interactive**, в результате таблица может быть открыта с нестандартным именем. Вслед за оператором **Open Table**, вызывается функция `TableInfo(0, TAB_INFO_NAME)` для определения имени, под которым MapInfo открыло таблицу.

### Открытие таблицы, которая уже открыта

Если таблица уже открыта, и оператор **Open Table...As** пытается заново открыть ту же таблицу под новым именем, MapBasic сгенерирует код ошибки. Одна таблица не может быть открыта под двумя различными именами одновременно.

Однако, если таблица уже открыта, а затем оператор **Open Table** пытается заново открыть таблицу без указания нового имени, MapBasic не будет генерировать код ошибки. Таблица просто остается открытой под ее текущим именем.

### Пример:

Следующий пример открывает таблицу STATES.TAB, затем отображает таблицу в окне Карты. Поскольку оператор **Open Table** использует предложение **As** для открытия таблицы под псевдонимом (USA), в окне Карты появится заголовок "USA Map" а не "States Map."

```
Open Table "States" As USA
Map From USA
```

В следующем примере **Функция TableInfo()** используется вместе с оператором **Open Table**. Если определенная таблица уже открыта (States) и Вы выполняете программу ниже, MapBasic будет открывать эту таблицу "C:STATES.TAB" с определенным номером (например "STATES\_2"). **Функция TableInfo()** будет возвращать номер, под которым таблица "C:STATES.TAB" была открыта.

```
Include "MAPBASIC.DEF"
Dim s_tab As String
Open Table "C:states"
s_tab = TableInfo(0, TAB_INFO_NAME)
Browse * From s_tab
Map From tab
```

### См. также:

**Оператор Close Table, Оператор Create Table, Оператор Delete, Оператор Fetch, Оператор Insert, Функция TableInfo(), Оператор Update**

## Оператор Open Window

### Назначение

Открывает или показывает окно.

### Синтаксис

```
Open Window window_name
```

*window\_name* – имя окна (например, Ruler) или код окна (например, WIN\_RULER)

### Описание

Оператор **Open Window** показывает окно MapInfo Professional. Например, следующий оператор открывает окно "Статистика", как если бы пользователь открыл его командой **НАСТРОЙКИ > ПОКАЗАТЬ ОКНО СТАТИСТИКИ**.

## Функция Overlap( )

Open Window Statistics

Параметр *window\_name* должен быть именем окна или целочисленным кодом. В следующей таблице в первой колонке приводятся имена окон, а во второй – описание и имена кодов, которые установлены в файле стандартных определений MapBasic MAPBASIC.DEF.

| Название окна | Описание окна   |
|---------------|---|
| MapBasic      | Окно MapBasic В качестве ссылок на окна можно использовать коды, заданные в MAPBASIC.DEF (WIN_MAPBASIC).  |
| Статистика    | Окно "Статистика" (WIN_STATISTICS).   |
| Legend        | Окно "Легенда" (WIN_LEGEND).  |
| Информация    | Окно "Информация", которое открывается инструментом Информация (WIN_INFO).  |
| Линейка       | Окно "Линейка", которое открывается инструментом Линейка (WIN_RULER).   |
| Help          | Окно Справки (WIN_HELP)   |
| Message       | Окно "Сообщение", которое использует <b>Оператор Print</b> (WIN_MESSAGE).   |
| TableList     | Диалог отображающий список всех открытых в данный момент таблиц.<br><b>Внимание:</b> Этот диалог не является окном в точном значении термина, поэтому в MAPBASIC.DEF нет записи |

Нельзя открывать окно документа (карту, график, список, отчет) оператором **Open Window**. Для того чтобы открыть окно документа каждого из этих типов, существует отдельный оператор (смотрите **Оператор Map**, **Оператор Graph**, **Оператор Browse**, **Оператор Layout** и **Оператор Create Redistricter**).

См. также:

**Оператор Close Window**, **Оператор Print**, **Оператор Set Window**

## Функция Overlap( )

### Назначение

Возвращает объект, полученный в результате географического пересечения двух объектов; результат похож на получаемый командой MapInfo Professional **Объекты > Удалить внешнюю часть**.

### Синтаксис

**Overlap**( *object1*, *object2* )

*object1* – параметр, задающий пересекающиеся объекты, ни один из которых не может быть точечным.

*object2* – параметр, задающий пересекающиеся объекты, ни один из которых не может быть точечным.

### Возвращаемая величина

Объект, являющийся пересечением *object1* и *object2*.

### Описание

Функция **Overlap( )** вычисляет географическое пересечение двух объектов (площадь перекрытия одним объектом другого), и возвращает объект, представляющий пересечение.

MapBasic передает все стили оформления объекта *object1* результирующему объекту. Если необходимо, стиль оформления объекта можно изменить на текущий в прикладной программе.

Если один из объектов является линейным (например, полилинией), а второй замкнутым (например, область), функция **Overlap( )** вернет часть линейного объекта, которая находит на площадь замкнутого.

См. также:

**Функция AreaOverlap( ), Функция Erase( ), Оператор Objects Intersect**

---

## Функция OverlayNodes( )

### Назначение

Возвращает объект, созданный на основе существующего, добавлением узлов в точках пересечения со вторым объектом.

### Синтаксис

**OverlayNodes( *input\_object*, *overlay\_object* )**

*input\_object* – объект, на основе которого будет создан результирующий, и который не может быть точечным;

*overlay\_object* – объект, пересекающий объект *input\_object* (также не может быть точечным).

### Возвращаемая величина

Объект либо типа "область", либо полилиния

### Описание

Функция **OverlayNodes( )** возвращает объект, созданный из узлов первого плюс узлы, полученные пересечением линий или контуров объектов *input\_object* и *overlay\_object*.

Если объект *input\_object* замкнут (область, прямоугольник, скругленный прямоугольник или эллипс), то функция **OverlayNodes( )** вернет область. Если объект *input\_object* линейный (прямая линия, полилиния или дуга), то функция **OverlayNodes( )** вернет полилинию.

MapBasic передает все стили оформления объекта *input\_object* результирующему.

Для определения, прибавила ли функция *OverlayNodes( )* несколько узлов к тем, которые были у объекта *input\_object*, используется **Функция ObjectInfo( )**. Заметим, что если объект *input\_object*, пересекающийся с другим, уже имеет узлы в точках пересечения, то функция **OverlayNodes( )** не будет добавлять новых узлов к имеющимся в *input\_object*, то есть результирующий объект будет состоять только из тех узлов, которые были у первого объекта.

**См. также:**

**Оператор Objects Overlay**

---

## Оператор Pack Table

### Назначение

Соответствует команде MapInfo **ТАБЛИЦА > ИЗМЕНИТЬ > УПАКОВАТЬ**.

### Синтаксис

```
Pack Table table { Graphic | Data | Graphic Data } [ Interactive ]
```

*table* – имя открытой таблицы, которая не имеет не сохраненные изменения.

### Описание

Для упаковки неграфических данных таблицы в операторе используется ключевое слово **Data**. Когда Вы сжимаете данные таким образом, MapInfo физически удаляет все строки, которые были помечены как удаленные.

Для упаковки графических объектов таблицы в операторе используется ключевое слово **Graphic**. Упаковывая графику, удаляются пустые места из .MAP-файла таблицы. Однако упаковка графических объектов несколько замедляет графические операции.

Оператор **Pack Table** может использовать одновременно и слово **Data**, и слово **Graphic**, или должен включать хотя бы одно.

Оператор **Pack Table** может послужить причиной удаления слоев из окна Карты, а также привести к потере объектов тематических слоев или объектов с Косметического слоя..

Если Вы используете ключевое слово **Interactive**, то MapInfo предложит пользователю сохранить тематические или/и косметические объекты. Этот оператор не может сжимать связанные таблицы. Кроме того, MapInfo не может сжимать таблицу, если она была изменена и эти изменения не были сохранены на диске. Для сохранения на диск таблицы используйте **Оператор Commit Table**.

**Внимание:** Упаковка таблицы может повлиять на подписи, созданные или измененные пользователем и сохраненные в Рабочем Наборе. Это происходит потому, что в Рабочем Наборе подписи пользователя соотносятся с номером строки таблицы;



операция упаковки меняет порядок записей, потому что из таблицы исключаются удаленные записи; после упаковки подписи могут появляться не там, где ожидалось и иметь неправильный вид. Если же Вы удаляли записи из нижней части таблицы, а подписи соотнесены с верхними записями, то упаковка не испортит подписей. (Это происходит потому, что в Рабочем Наборе подписи пользователя соотносятся с номером строки таблицы; операция упаковки меняет порядок записей потому, что из таблицы исключаются удаленные записи; после упаковки подписи могут появляться не там, где ожидалось и иметь неправильный вид.) Если удалить последние записи таблицы (например, выбрав их в конце окна списка), то упаковка не повлияет на созданные подписи.

### Упаковка таблиц Access

Оператор **Pack Table** копирует исходную таблицу Microsoft Access без колонок тех типов, которые MapInfo Professional не поддерживает. Если в таблице Microsoft Access имеются колонки MEMO, OLE или LONG BINARY типов, то такие колонки при упаковке пропадут.

#### Пример:

```
Pack Table parcels Data
```

#### См. также:

**Оператор Open Table**

### Функция PathToDirectory\$( )

#### Назначение

Возвращает заданный каталог с файлами.

#### Синтаксис

**PathToDirectory\$( filespec )**

*filespec* – строка, содержащая полное имя файла (маршрут + имя).

#### Возвращаемая величина

Строка

#### Описание

Функция **PathToDirectory\$( )** извлекает из полного имени файла имя каталога.

Имя файла, заданное полностью, состоит из каталога и имени файла. Имя файла, заданное полностью, C:\MAPINFO\DATA\WORLD.TAB содержит каталог "C:\MAPINFO\DATA\".

#### Пример:

```
Dim s_filespec, s_filedir As String
s_filespec = "C:\MAPINFO\DATA\STATES.TAB"
s_filedir = PathToDirectory$(s_filespec)

' s_filedir now contains the string "C:\MAPINFO\DATA\"
```

#### См. также:

**Функция PathToFileName\$( ), Функция PathToTableName\$( )**

---

### Функция PathToFileName\$( )

#### Назначение

Извлекает из полного имени файла имя файла.

#### Синтаксис

**PathToFileName\$( filespec )**

*filespec* – строка, содержащая полное имя файла (маршрут + имя).

#### Возвращаемая величина

Строка

### Описание

Функция **PathToFileName\$( )** извлекает из полного имени файла только имя файла.

Имя файла, заданное полностью, состоит из каталога и имени файла. Функция **PathToFileName\$( )** возвращает часть полного имени, которая содержит имя файла.

Имя файла, заданное полностью, C:\MAPINFO\DATA\WORLD.TAB состоит из имени каталога ("C:\MAPINFO\DATA\") и имени файла ("WORLD.TAB").

### Пример:

```
Dim s_filespec, s_filename As String
s_filespec = "C:\MAPINFO\DATA\STATES.TAB"
s_filename = PathToFileName$(s_filespec)

' filename now contains the string "STATES.TAB"
```

### См. также:

**Функция PathToDirectory\$( )**, **Функция PathToTableName\$( )**

---

## Функция PathToTableName\$( )

### Назначение

Возвращает синоним имени таблицы (псевдоним, например, "\_1995\_Data"), полученный из полного имени файла таблицы (например, "C:\MapInfo\Data\1995 Data.tab").

### Синтаксис

**PathToTableName\$( filespec )**

*filespec* – строка, содержащая полное имя файла (маршрут + имя).

### Возвращаемая величина

Строка длиной до 31 символа. Величина типа String.

### Описание

Получая полное имя файла с расширением .TAB, функция возвращает строку, которая может быть для этой таблицы в данный момент псевдонимом (alias). Именно такой синоним видит пользователь в строке заголовка документального окна MapInfo (например, в строке заголовка Списка).

Для получения синонима таблицы при ее открытии MapInfo удаляет из полного имени файла имя носителя, каталога и расширение ".TAB" (для системы DOS). Любые специальные символы, такие как тире, пробелы и др. заменяются знаками подчеркивания (\_). Если имя файла начинается с цифры, то MapInfo вставляет знак подчеркивания в начало имени таблицы. Если результирующая строка получается длиннее 31 символа, то MapInfo отсекает лишние с конца.

Заметим, что таблица может быть открыта с именем-синонимом, отличающимся от имени файла. Программа MapBasic может открыть таблицу с именем-синонимом, используя **Оператор Open Table** с предложением **As**. Например, откроем таблицу WORLD под синонимом "Earth":

```
Open Table "C:\MapInfo\Data\World.tab" As Earth
```

Могут быть также открыты две одноименные таблицы, расположенные в разных каталогах, и MapInfo автоматически изменит имя одной из таблиц. В таких случаях имя таблицы, возвращаемое функцией **PathToTableName\$ ( )**, может не совпадать с именем, под которым она открыта в MapInfo. Для того, чтобы определить синонимическое имя открытой таблицы, используется **Функция TableInfo ( )** с параметром (TAB\_INFO\_NAME).

### Пример:

```
Dim s_filespec, s_tablename As String
s_filespec = "C:\MAPINFO\DATA\STATES.TAB"
s_tablename = PathToTableName$(s_filespec)
' s_tablename now contains the string "STATES"
```

См. также:

**Функция PathToDirectory\$ ( )**, **Функция PathToFileName\$ ( )**, **Функция TableInfo ( )**

---

## Предложение Pen

### Назначение

Задаёт стиль контуров графических объектов.

### Синтаксис

**Pen** *pen\_expr*

*pen\_expr* – это выражение, возвращающее объект типа Pen, то есть, **MakePen( width, pattern, color )**.

### Описание

Предложение **Pen** входит в состав некоторых операторов, в которых необходимо задавать стиль линии для некоторых графических объектов. Стиль линии представляет собой набор из атрибутов толщины линии, типа линии и цвета.

Предложение **Pen** не является полным оператором. Например, **Оператор Create Line** использует предложение **Pen**, когда создается новый объект типа "линия". Предложение Pen задаёт стиль для нового объекта. Если оператор не использует это предложение, то будет использована текущая настройка этого стиля в MapInfo. Слово **Pen** может сопровождаться выражением, возвращающим значение типа **Pen**. Например, выражение такого вида может быть переменной **Pen**:

**Pen** *pen\_var*

или вызов функции (например, **Функция CurrentPen( )** или **Функция MakePen( )**), которые возвращают значение Pen:

```
Pen MakePen(1, 2, BLUE)
```

В некоторых операторах (например, **Оператор Set Map**) после слова **Pen** стиль задается непосредственно набором из трех целочисленных параметров (width, pattern и color), например:

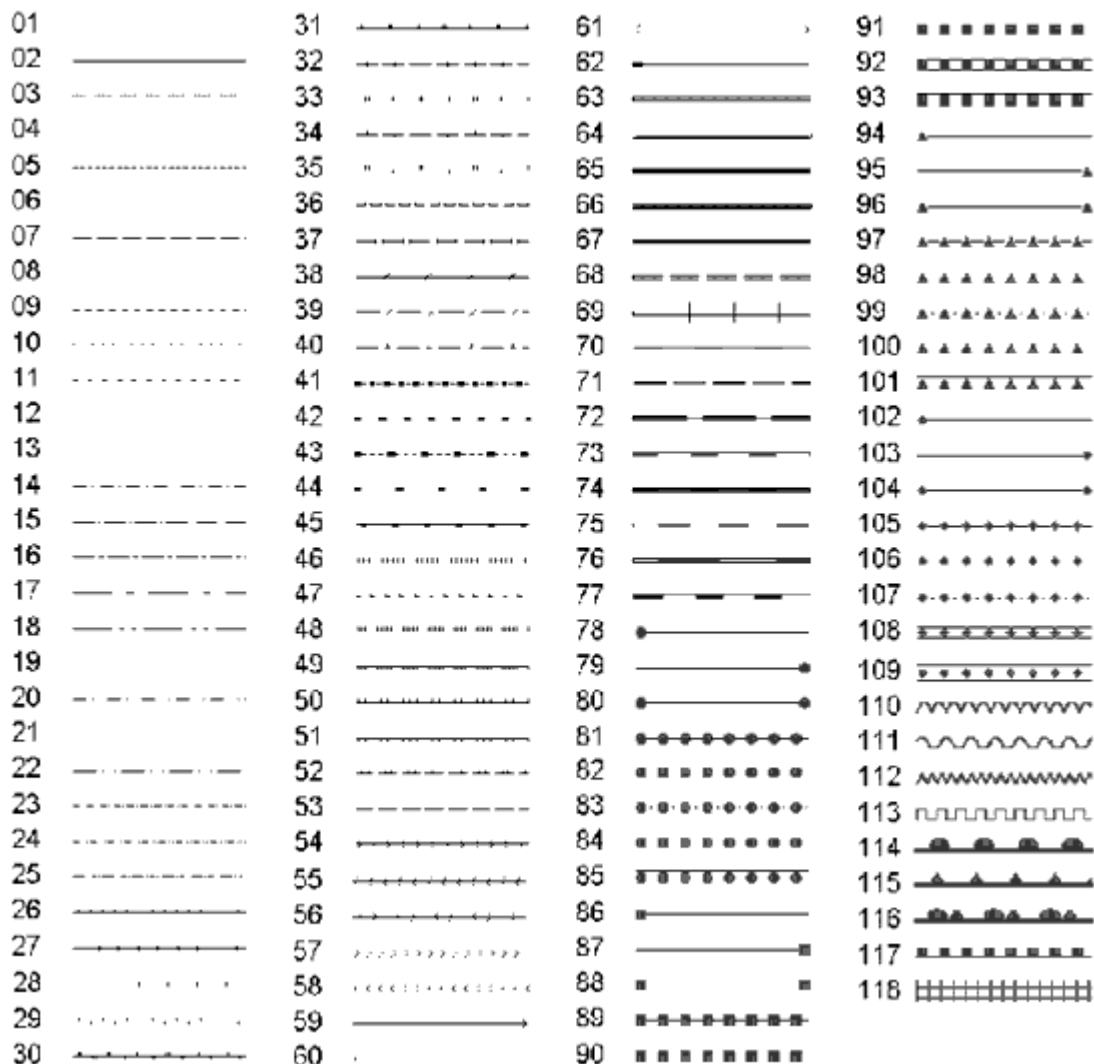
```
Pen(1, 2, BLUE)
```

Некоторые операторы MapBasic используют выражения типа стиля контура **Pen** в качестве параметра (например, переменная типа **Pen** не используя при этом предложения **Pen**). Одним из примеров является **Оператор Alter Object**.

В следующей таблице приводится описание параметров стиля линии:

| Компонента<br>стиля | Описание  |
|---------------------|---|
| width<br>(толщина)  | Толщина линии в точках, целочисленная величина типа Integer, от 1 до 7 включительно. Если Вы хотите иметь невидимую линию в объекте, то задайте нулевую толщину. При этом, если параметр будет равен 0, то параметр типа линии pattern должен быть равен 1 (единице). |
| pattern             | Тип линии, целочисленная величина типа Integer, от 1 до118 (смотрите таблицу ниже). Значение 1 обозначает невидимую линию.  |
| color (цвет)        | Целочисленный код цвета для рисунка штриха. Может быть заменен вызовом функции; см. <b>Функция RGB( ) на стр. 118</b> .   |

Образцы штриховок приведены на рисунке ниже.



### Пример:

```
Include "MAPBASIC.DEF"
Dim cable As Object
Create Line
    Into Variable cable
    (73.5, 42.6) (73.67, 42.9)
    Pen MakePen(1, 2, BLACK)
```

### См. также:

**Оператор Alter Object**, **Функция CreateLine()**, **Оператор Create Pline**, **Функция CurrentPen()**, **Функция MakePen()**, **Функция RGB()**, **Оператор Set Style**

## Функция PenWidthToPoints( )

### Назначение

Функция PenWidthToPoints возвращает толщину линии в американских типографских пойнтах (пунктах).

### Синтаксис

**PenWidthToPoints** ( *penwidth* )

*penwidth* – это целочисленное значение, задающее толщину линии.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **PenWidthToPoints( )** возвращает значение толщины линии в пойнтах. Толщину линии, использованную в стиле оформления линии, может возвращать **Функция StyleAttr( )**. Толщина линии, которую возвращает **Функция StyleAttr( )**, может измеряться в пойнтах или пикселах. В пикселах измеряется толщина линии меньше десяти. В пойнтах измеряется любая толщина линии, выраженная в единицах более десяти. **PenWidthToPoints( )** будет возвращать значение только для линий, толщина которых задана в точках. Чтобы определить в каких единицах задана толщина линии: пикселах или пойнтах, используется **Функция IsPenWidthPixels( )**.

### Пример:

```
Include "MAPBASIC.DEF"
Dim CurPen As Pen
Dim Width As Integer
Dim PointSize As Float
CurPen = CurrentPen( )
Width = StyleAttr(CurPen, PEN_WIDTH)
If Not IsPenWidthPixels(Width) Then
    PointSize = PenWidthToPoints(Width)
End If
```

### См. также:

**Функция CurrentPen( )**, **Функция IsPenWidthPixels( )**, **Функция MakePen( )**, **Предложение Pen**, **Функция PointsToPenWidth( )**, **Функция StyleAttr( )**

## Функция **Perimeter( )**

### Назначение

Возвращает периметр графического объекта.

### Синтаксис

**Perimeter**( *obj\_expr*, *unit\_name* )

*obj\_expr* - выражение, определяющее объект.

*unit\_name* - строка, определяющая единицы измерения расстояния (к примеру, "км.").

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Perimeter( )** вычисляет периметр объекта *obj\_expr*. Функция **Perimeter( )** определена для следующих типов объектов: эллипсов, прямоугольников, скругленных прямоугольников и полигонов. Другие типы объектов имеют периметр нулевой длины.

Возвращаемое функцией **Perimeter( )** значение длины периметра измеряется в единицах длины, определенных параметром *unit\_name*; например, для получения длины в милях, укажите "mi" в качестве *unit\_name*. Список допустимых единиц измерения смотрите в [Оператор Set Distance Units on page 633](#).

Функция **Perimeter( )** возвращает приблизительные результаты, когда применяется к скругленным прямоугольникам. MapBasic вычисляет периметр прямоугольников со скругленными углами как если бы они были обычными прямоугольниками. В большинстве случаев MapInfo Professional проводит либо декартовы, либо сферические вычисления. Обычно выполняются сферические вычисления; если координатная система - план, то выполняются декартовы вычисления..

### Пример:

Следующий пример показывает как Вы можете использовать функцию **Perimeter( )** для определения периметра географического объекта.

```
Dim perim As Float
Open Table "world"
Fetch First From world
perim = Perimeter(world.obj, "km")
' переменная perim теперь содержит ' периметр
полигона соответствующего первой записи таблицы World.
```

Вы можете использовать функцию **Perimeter( )** внутри [Оператор Select](#). Следующий оператор [Оператор Select](#) извлекает информацию из таблицы States и сохраняет результаты во временной таблице Results.

Поскольку оператор [Оператор Select](#) включает функцию **Perimeter( )**, таблица Results будет включать колонку, показывающую периметр каждого штата.

```
Open Table "states"
Select state, Perimeter(obj, "mi")
From states Into results
```



См. также:

Функция `Area( )`, Функция `ObjectLen( )`, Оператор `Set Distance Units`

---

## Функция `PointsToPenWidth( )`

### Назначение

Преобразует значение, заданное в типографских единицах “пункты”, в ширину линии.

### Синтаксис

`PointsToPenWidth( pointsize )`

*pointsize* – это десятичное число, представляющее значение в десятых долях “пункта”.

### Возвращаемая величина

Целое число типа `SmallInt`.

### Описание

Функция `PointsToPenWidth( )` преобразует значение заданное в “пунктах”, в значение ширины линии.

### Пример:

```
Include "MAPBASIC.DEF"
Dim Width As Integer
Dim p_bus_route As Pen
Width = PointsToPenWidth(1.7)
p_bus_route = MakePen(Width, 9, RED)
```

См. также:

Функция `CurrentPen( )`, Функция `IsPenWidthPixels( )`, Функция `MakePen( )`, Предложение `Pen`, Функция `PenWidthToPoints( )`, Функция `StyleAttr( )`

---

## Функция `PointToMGRS$( )`

### Назначение

Функция `PointToMGRS$( )` конвертирует координаты объекта, представляющего точку, в строковую величину координаты MGRS. Функция поддерживает только точечные объекты.

### Синтаксис

`PointToMGRS$( inputobject )`

*inputobject* – выражение для точечного объекта.

### Описание

MapInfo Professional заранее автоматически преобразует координаты исходной точки в текущей системе координат MapBasic в координаты в проекции Долгота/Широта (WGS84) до преобразования в строку координат MGRS. Однако, по умолчанию, координатная система MapBasic – Долгота/Широта (без проекции); при использовании такой промежуточной системы координат может возникнуть значительная потеря точности конечного результата, поскольку преобразования без использования сведений о проекции значительно снижают точность. Обычно, в качестве системы координат MapBasic либо задается Долгота/Широта (WGS84), либо оставляется исходная система координат таблицы, в этих случаях промежуточное преобразование не выполняется. Смотрите пример 2 ниже.

### Возвращаемая величина

Строка

### Примеры

Следующий пример демонстрирует использование функций MGRSToPoint( ) и PointToMGRS( )

#### Пример 1:

```
dim obj1 as Object
dim s_mgrs As String
dim obj2 as Object

obj1 = CreatePoint(-74.669, 43.263)
s_mgrs = PointToMGRS$(obj1)
obj2 = MGRSToPoint(s_mgrs)
```

**Пример 2:**

' При использовании функций `PointToMGRS$ ( )` или `MGRStoPoint ( )` ' важно удостовериться в том, что текущая система ' координат MapBasic соответствует координатной системе ' таблицы, хранящей точечные объекты.

```
'Set the MapBasic coordsys to that of the table used
Set CoordSys Table MGRSfile

'Update a Character column (e.g. COL2) with MGRS strings from
'a table of points

Update MGRSfile
  Set Col2 = PointToMGRS$(obj)

'Update two float columns (Col3 & Col4) with
'CentroidX & CentroidY information
'from a character column (Col2) that contains MGRS strings.

Update MGRSfile
  Set Col3 = CentroidX(MGRStoPoint(Col2))

Update mgrstestfile ' MGRSfile
  Set Col4 = CentroidY(MGRStoPoint(Col2))

Table MGRSfile
Close Table MGRSfile
```

**См. также:**

**Функция `MGRStoPoint ( )`**

---

## Оператор Print

**Назначение**

Печатает пояснительный текст или текст сообщения из программы в окне "Сообщение".

**Синтаксис**

**Print** *message*

*string\_expr* – выражение, результат которого – строка.

**Описание**

Оператор **Print** используется для вывода текста в окне "Сообщение". Это окно является одним из вспомогательных окон в MapInfo. Окно "Сообщение" предназначено для вывода информации из программ, написанных на MapBasic. Эти сообщения можно использовать для комментирования действий программы без ее остановки. Например, "Запись удалена". Задать шрифт в окне "Сообщение" можно, используя **Оператор Set Window**. Программа MapBasic может перед выполнением оператора Print открыть окно сообщений, используя **Оператор Open Window**.

Если оператор **Print** выполняется, когда окно сообщений закрыто, то MapBasic откроет его автоматически. Оператор **Print** похож на **Оператор Note**, и любой из них можно использовать для вывода служебной или отладочной информации. Однако, **Оператор Note** выводит сообщение в диалоговом окне, и пока пользователь не нажмет кнопку **ОК**, выполнение программы будет приостановлено.. Выполнение программы после оператора **Print** продолжается, а каждый следующий оператор **Print** будет печатать текст с новой строки в уже открытом окне "Сообщение". Если все окно сообщений будет заполнено или строка будет длиннее ширины окна, то пользователь может прокручивать окно в горизонтальном и вертикальном направлениях.

Чтобы очистить окно "Сообщение" перед выводом сообщения, используйте символ прогона листа (ASCII-код 12):

```
Print Chr$(12) 'Это сообщение очищает поле окна
```

Для начала новой строки в тексте сообщения используйте символ возврата каретки (ASCII-код 10). Следующий оператор **Print** выводит сообщение в две строки:

```
Print "Map Layers:" + Chr$(10) + " World, Capitals"
```

Оператор **Print** преобразует табуляцию (ASCII-код 09) в символ пробела (ASCII-код 32).

### Пример:

Откроем окно "Сообщение", назовем шрифт Arial Сур, жирный, размером в 10 пунктов, синего цвета. Назначим также размер окна, 3 на 1 дюйм, расположение ниже и правее на четверть дюйма от правого верхнего угла основного окна MapInfo. Теперь можно печатать:

```
Include "MAPBASIC.DEF" ' ' понадобится для цвета 'BLUEOpen Window Message  
' открываем окно Set Window Message Font ("Helv", 1, 10, BLUE) '  
назначаем полужирный шрифт Helvetica...Position (0.25, 0.25) ' выбираем  
позицию на экране Width 3.0 ' ширину окна Height 1.0 ' высоту Print  
"MapBasic-диспетчер на линии"
```

**Внимание:** Объем памяти, выделяемый для окна "Сообщение", был удвоен и составляет теперь 8191 символ.

**См. также:**

**Функция Ask( )**, **Оператор Close Window**, **Оператор Note**, **Оператор Open Window**, **Оператор Set Window**

---

## Оператор Print #

### Назначение

Записывает данные в файл, открытый в режиме последовательного доступа.

### Синтаксис

```
Print # file_num [ , expr ]
```

*file\_num* – номер файла, который открыл **Оператор Open File**;

*expr* – выражение для записи в файл.

### Описание

Оператор **Print #** выводит данные в файл, который должен быть открыт оператором Open File в последовательном режиме доступа, разрешающем запись (**OUTPUT** или **APPEND**).

оператором **Оператор Open File**, который закрепляет за файлом номер, используемый в параметре *file\_num*.

MapInfo Professional записывает выражение *expr* в одну строку файла. Для записи выражений списком, каждое в отдельную строку файла, используется **Оператор Write #** вместо **Print #**.

См. также:

**Оператор Line Input, Оператор Open File, Оператор Write #**

---

## Оператор PrintWin

### Назначение

Печатает содержимое окна.

### Синтаксис

```
PrintWin [ Window window_id ][ Interactive ][ File output_filename ]
        [ Overwrite ]
```

*window\_id* – идентификатор окна.

*output\_filename* – строка представляющая имя файла. Если файл уже существует и не указано ключевое слово **Overwrite** произойдёт ошибка.

### Описание

Оператор **PrintWin** используется для вывода содержимого окна на печать.

Если используется предложение **Window**, то MapBasic будет печатать заданное окно. Если окно не задано, то напечатается содержимое активного окна.

Параметр *window\_id* должен быть идентификатором окна, смотрите разделы: **Функция FrontWindow( ) on page 328** и **Функция WindowInfo( ) on page 755**, в которых получение идентификатора описано подробнее.

Если оператор включает ключевое слово **Interactive**, MapBasic показывает стандартный диалог "Печать". Без этого ключевого слова печать будет производиться автоматически, без диалога с пользователем, используя текущие установки печати.

### Примеры

#### Пример 1

```
Dim win_id As Integer
Open Table "world"
Map From world
win_id = FrontWindow( )
'
' knowing the ID of the Map window,
' the program could now print the map by
' issuing the statement:
'
PrintWin Window win_id Interactive
```

#### Пример 2

```
PrintWin Window FrontWindow( ) File "c:\output\file.plt"
```

См. также:

[Функция FrontWindow\( \)](#), [Оператор Run Menu Command](#), [Функция WindowInfo\( \)](#)

---

## Функция PrismMapInfo( )

### Назначение

Возвращает настройки окна Карты-призмы.

### Синтаксис

```
PrismMapInfo( window_id, attribute )
```

*window\_id* – идентификатор окна.

*attribute* это целое, определяющее, какого типа информация будет возвращена.

### Возвращаемая величина

Вещественное (Float), логическое (Logical), или строка символов (String), в зависимости от параметра атрибута.

### Описание

Функция **PrismMapInfo( )** возвращает информацию об окне Карты-призмы.

Параметр *window\_id* определяет, какое окно Карты-призмы обрабатывается функцией. Чтобы получить идентификатор окна, вызывается [Функция FrontWindow\( \)](#) сразу же после открытия окна или вызывается [Функция WindowID\( \)](#) в любой момент после создания окна.

Существует несколько числовых атрибутов, которые **PrismMapInfo( )** может вернуть для каждого окна Карты-призмы. Параметр *атрибута* сообщает функции **PrismMapInfo( )**, какие данные об окне Карты возвращаются. Параметр *атрибута* должен быть одним из кодов, представленных в следующей таблице; коды определены в файле MAPBASIC.DEF.

| Атрибут                      | Возвращаемая величина  |
|------------------------------|--|
| PRISMMAP_INFO_SCALE          | Вещественное, масштабный фактор Карты-призмы.                                    |
| PRISMMAP_INFO_BACKGROUND     | Целое, цвет фона (см. раздел: " <b>Функция RGB( ) на стр. 118</b> ").            |
| PRISMMAP_INFO_LIGHT_X        | Вещественное, координата X источника освещения.                                  |
| PRISMMAP_INFO_LIGHT_Y        | Вещественное, координата Y источника освещения.                                  |
| PRISMMAP_INFO_LIGHT_Z        | Вещественное, координата Z источника освещения.                                  |
| PRISMMAP_INFO_LIGHT_COLOR    | Целое, цвет источника света (см. раздел: " <b>Функция RGB( ) на стр. 118</b> "). |
| PRISMMAP_INFO_CAMERA_X       | Вещественное, координата X камеры.   |
| PRISMMAP_INFO_CAMERA_Y       | Вещественное, координата Y камеры.   |
| PRISMMAP_INFO_CAMERA_Z       | Вещественное, координата Z камеры.   |
| PRISMMAP_INFO_CAMERA_FOCAL_X | Вещественное, координата X точки фокуса камеры.                                  |
| PRISMMAP_INFO_CAMERA_FOCAL_Y | Вещественное, координата Y точки фокуса камеры.                                  |
| PRISMMAP_INFO_CAMERA_FOCAL_Z | Вещественное, координата Z точки фокуса камеры.                                  |
| PRISMMAP_INFO_CAMERA_VU_1    | Вещественное, первое значение параметра нормали вектора камеры.                  |
| PRISMMAP_INFO_CAMERA_VU_2    | Вещественное, второе значение параметра нормали вектора камеры.                  |
| PRISMMAP_INFO_CAMERA_VU_3    | Вещественное, третье значение параметра нормали вектора камеры.                  |
| PRISMMAP_INFO_CAMERA_VPN_1   | Вещественное, первое значение параметра нормали плоскости наблюдения.            |
| PRISMMAP_INFO_CAMERA_VPN_2   | Вещественное, второе значение параметра нормали плоскости наблюдения.            |

| Атрибут                        | Возвращаемая величина   |
|--------------------------------|---|
| PRISMMAP_INFO_CAMERA_VPN_3     | Вещественное, третье значение параметра нормали плоскости наблюдения. |
| PRISMMAP_INFO_CAMERA_CLIP_NEAR | Вещественное, ближняя плоскость кадра камеры.                         |
| PRISMMAP_INFO_CAMERA_CLIP_FAR  | Вещественное, дальняя плоскость кадра камеры.                         |
| PRISMMAP_INFO_INFOTIP_EXPR     | Строка для всплывающей подсказки.. Ранее не была документирована.     |



**Пример:**

Распечатываем все стандартные переменные, определенные для окна Карты-призмы:

```
include "Mapbasic.def"
Print "PRISMMAP_INFO_SCALE: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_SCALE)
Print "PRISMMAP_INFO_BACKGROUND: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_BACKGROUND)
Print "PRISMMAP_INFO_UNITS: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_UNITS)
Print "PRISMMAP_INFO_LIGHT_X : " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_LIGHT_X )
Print "PRISMMAP_INFO_LIGHT_Y : " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_LIGHT_Y )
Print "PRISMMAP_INFO_LIGHT_Z: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_LIGHT_Z)
Print "PRISMMAP_INFO_LIGHT_COLOR: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_LIGHT_COLOR)
Print "PRISMMAP_INFO_CAMERA_X: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_X)
Print "PRISMMAP_INFO_CAMERA_Y : " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_Y )
Print "PRISMMAP_INFO_CAMERA_Z : " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_Z )
Print "PRISMMAP_INFO_CAMERA_FOCAL_X: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_FOCAL_X)
Print "PRISMMAP_INFO_CAMERA_FOCAL_Y: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_FOCAL_Y)
Print "PRISMMAP_INFO_CAMERA_FOCAL_Z: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_FOCAL_Z)
Print "PRISMMAP_INFO_CAMERA_VU_1: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_VU_1)
Print "PRISMMAP_INFO_CAMERA_VU_2: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_VU_2)
Print "PRISMMAP_INFO_CAMERA_VU_3: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_VU_3)
Print "PRISMMAP_INFO_CAMERA_VPN_1: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_VPN_1)
Print "PRISMMAP_INFO_CAMERA_VPN_2: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_VPN_2)
Print "PRISMMAP_INFO_CAMERA_VPN_3: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_VPN_3)
Print "PRISMMAP_INFO_CAMERA_CLIP_NEAR: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_CLIP_NEAR)
Print "PRISMMAP_INFO_CAMERA_CLIP_FAR: " + PrismMapInfo(FrontWindow( ),
PRISMMAP_INFO_CAMERA_CLIP_FAR)
```

**См. также:**

**Оператор Create PrismMap, Оператор Set PrismMap**

## Функция ProgramDirectory\$( )

### Назначение

Возвращает маршрут, по которому установлена рабочая версия MapInfo Professional.

### Синтаксис

**ProgramDirectory\$( )**

### Возвращаемая величина

Строка

### Описание

Функция **ProgramDirectory\$( )** возвращает в виде строки маршрут, по которому установлена рабочая версия MapInfo.

### Пример:

```
Dim s_prog_dir As String  
s_prog_dir = ProgramDirectory$( )
```

### См. также:

[Функция HomeDirectory\\$\( \)](#), [Функция SystemInfo\( \)](#)

---

## Оператор ProgressBar

### Назначение

Показывает диалог с кнопкой "Отмена" и горизонтальным индикатором состояния.

### Синтаксис

```
ProgressBar status_message  
    Calling handler  
    [ Range n ]
```

*status\_message* – строка, которая будет выводиться в диалоге.

*handler* – имя подпрограммы;

*n* – число, при котором выполнение программы завершится.

### Предупреждение

Вы не можете использовать оператор **ProgressBar** в окне MapBasic.

## Описание

Оператор **ProgressBar** используется для создания диалогового окна-индикатора выполнения процесса. Такой диалог снабжен процентной шкалой и кнопкой "Отменить". Диалог сопровождает определенные действия и показывает, насколько они выполнены. Выполнение можно прервать, нажав кнопку "Отменить". Информацию о том, как завершился процесс – самостоятельно или был прерван – можно узнать при помощи функции `CommandInfo (CMD_INFO_DLG_OK)`, вызвав ее после оператора **ProgressBar**.

Строковый параметр *status\_message* может задавать поясняющий текст, который будет отображен внутри диалога процесса над шкалой. Например, это может быть: "Идет процесс копирования...".

Параметр *handler* является именем процедуры-обработчика, выполняющей те действия, которые сопровождаются диалогом процесса. Как будет описано дальше, подпрограмма должна выполнить определенные действия, чтобы взаимодействовать с оператором **ProgressBar**.

Параметр *n* задает число, которое будет стоять справа у шкалы в окне диалога. Например, процедура-обработчик проверяет 7000 строк таблицы. Вы задаете в операторе **ProgressBar** параметр *n* равным семи тысячам. Теперь шкала отображает информацию о построчном выполнении процедуры. Если предложение `Range n` отсутствует, то по умолчанию устанавливается значение 100.

Когда программа выполняет оператор **ProgressBar**, MapBasic вызывает подпрограмму *handler*. Работа, выполняемая этой процедурой, должна быть разбита на небольшие по длительности отрезки, не более нескольких секунд каждый. Это делается для того, чтобы пользователь мог нажать на кнопку "Отмена", после чего MapBasic удалит с экрана диалог, и выполнение программы передается следующему после **ProgressBar** оператору. Если же пользователь не нажимал на эту кнопку, MapBasic продолжает выполнение процедуры-обработчика. Если пользователь ни разу не нажал кнопку "Отмена", то процедура-обработчик благополучно завершает свое дело.

Поэтому в тексте процедуры-обработчика *handler* должны быть предусмотрены средства для разбиения процесса на небольшие отрезки, а также средства слежения за возобновлением процесса. Пока оператор **ProgressBar** работает, MapBasic периодически обращается к *процедуре-обработчику* до тех пор, пока пользователь не нажмет кнопку **отмены**, либо *процедура* не закончит работу. Для контроля за этими событиями поддерживается специальная переменная, имеющая имя `ProgressBar`.

Если обработчик присвоит этой переменной значение "минус единица" (`ProgressBar = -1`), то MapBasic прерывает процесс и убирает диалог с экрана. И, наоборот, любое положительное значение переменной *ProgressBar*, например, (`ProgressBar = 50`) используется MapBasic для отображения "процента выполнения" в диалоге. MapBasic вычисляет "процент выполнения" делением текущего значения переменной *ProgressBar* на значение переменной *Range*. Например, если переменной **Range** присвоено значение. `Range 400` то, если значение **ProgressBar** равно 100, "процент выполнения" будет равен 25% и это будет отображено в диалоге.

В выражениях, следующих за оператором **ProgressBar**, Вы можете определить причину окончания процесса, описанного в ProgressBar: либо нормальное завершение, либо нажатие на кнопку **"Отмена"**. Для этого оператора **ProgressBar** используется функция `CommandInfo(CMD_INFO_DLG_OK)` возвращающая значение TRUE, если процесс завершился нормально, и FALSE, если процесс был прерван пользователем.

### Пример:

Этот пример демонстрирует, как должна быть написана процедура, вызываемая оператором **ProgressBar**. Процесс в этом примере состоит из 600 итераций, это может быть однотипная обработка 600 строк таблицы. Оператор **ProgressBar** вызывает из основной процедуры sub-процедуру "write\_out". Sub-процедура "write\_out" обрабатывает записи в течение 2 секунд. Затем она возвращает управление, и MapBasic проверяет, не была ли нажата кнопка **отмены**. Если пользователь не нажимал на кнопку **отмены**, то MapBasic возвращает управление в процедуру-обработчик. Такая последовательность действий повторяется до тех пор, пока не будет выполнена вся задача.

```
Include "mapbasic.def"
Declare Sub Main
Declare Sub write_out

Global next_row As Integer

Sub Main
    next_row = 1
    ProgressBar "Writing data..." Calling write_out Range 600
    If CommandInfo(CMD_INFO_STATUS) Then
        Note "Operation complete! Thanks for waiting."
    Else
        Note "Operation interrupted!"
    End If
End Sub

Sub write_out
    Dim start_time As Float
    start_time = Timer( )
    ' process records until either (a) the job is done,
    ' or (b) more than 2 seconds elapse within this call
    Do While next_row <= 600 And Timer( ) - start_time < 2
        ' ' ' Here, we would do the actual work ' ' '
        ' ' ' of processing the file. ' ' '
        ' ' '
        next_row = next_row + 1
    Loop

    ' Now figure out why the Do loop terminated: was it
    ' because the job is done, or because more than 2
    ' seconds have elapsed within this iteration?
    If next_row > 600 Then
        ProgressBar = -1 'tell caller "All Done!"
    Else
        ProgressBar = next_row 'tell caller "Partly done"
```

```
End If
End Sub
```

**См. также:**

**CommandInfo( )** функция, **Оператор Note**, **Оператор Print**

---

## Функция Proper\$( )

### Назначение

Возвращает строку, преобразуя все первые буквы слов в прописные, а остальные в строчные.

### Синтаксис

```
Proper$( string_expr )
```

*string\_expr* - выражение, результат которого есть строка.

### Возвращаемая величина

Строка

### Описание

Функция **Proper\$( )** возвращает строку, полученную из строки, представленной выражением *string\_expr*, преобразованием всех первых букв слов в прописные и остальных в строчные. Такое преобразование имеет смысл для собственных имен и названий.

### Пример:

```
Dim name, propername As String

name = "ed bergen"
propername = Proper$(name)
' propername now contains the string "Ed Bergen"

name = "ABC 123"
propername = Proper$(name)
' propername now contains the string "Abc 123"

name = "a b c d"
propername = Proper$(name)
' propername now contains the string "A B C D"
```

**См. также:**

**Функция LCase\$( )**, **Функция UCase\$( )**

## Функция `ProportionOverlap( )`

### Назначение

Вычисляет процент перекрытия одного объекта другим.

### Синтаксис

`ProportionOverlap( object1, object2 )`

*object1* – объект снизу (не может быть точечным или текстовым);

*object2* – объект сверху (не может быть точечным или текстовым).

### Возвращаемая величина

вещественное число `AreaOverlap( object1, object2 ) / Area( object1 )`.

См. также:

[Функция `AreaOverlap\( \)`](#)

---

## Оператор `Put`

### Назначение

Записывает в открытый файл содержимое переменной.

### Синтаксис

`Put [ # ] filenum, [ position,] var_name`

*filenum* – номер файла, который открыл [Оператор `Open File`](#);

*position* – позиция файла для записи (не для последовательного доступа);

*var\_name* – имя переменной, значение которой будет использовано как данные для записи.

### Описание

Оператор **Put** позволяет записывать значение переменной в файл, открытый в режиме произвольного или бинарного доступа.

**Внимание:** Если [Оператор `Open File`](#) открыл файл в режиме последовательного доступа (**OUTPUT** или **APPEND**), то вместо оператора **Put** для записи в файл используйте либо [Оператор `Print #`](#), либо [Оператор `Write #`](#).

Если [Оператор `Open File`](#) открыл файл в режиме произвольного доступа (**RANDOM**), то параметр **Position** оператора **Put** задает номер записи, в которую будет записано новое значение. Сразу после открытия файла позиция для записи – это начало (первая запись) файла. Если [Оператор `Open File`](#) открыл файл в режиме бинарного доступа (**BINARY**), то одновременно можно записывать только одну переменную. Порядок, по которому байты записываются в файл, зависит от Вашей вычислительной платформы (см. также

предложение **ByteOrder** в разделе: "**Оператор Open File на стр. 62**"). Количество записанных байтов зависит от типа переменной `var_name`. Например, записывая целое число, оператор **Put** внесет четыре байта. Переменные **MapBasic** записываются следующим образом:

| Тип переменной                               | Хранение в файле  |
|--|---|
| Логическое                                   | однобайтовое значение, или 0, или другое ненулевое число                      |
| Целое число типа <code>SmallInt</code> .     | двухбайтовое значение, целое число  |
| Целое число типа <code>Integer</code> .      | четырёхбайтовое значение, целое число   |
| Вещественное число типа <code>Float</code> . | восьмибайтовое число в формате IEEE   |
| Строка                                       | длина строки плюс один байт для нулевого значения, обозначающего конец строки |
| Дата типа <code>Date</code>                  | 4 байта: Короткое целое число для года, ,байт для месяца и байт для дня       |
| Другие типы                                  | не записываются оператором <b>Put</b>   |

Параметр **position** задает смещение в файле в байтах. Сразу после открытия файла позиция для записи – это первый байт файла (начало файла). Выполнение оператора **Put** автоматически наращивает смещение по байтам. Если параметр **Position** не задан, то оператор **Put** начнет запись с текущей позиции. В файл, открытый в режиме **BINARY**, оператор **Put** не может записывать строки неопределенной длины; все переменные типа `String` должны иметь фиксированную длину. В файл, открытый в режиме **RANDOM**, оператор **Put** не может записывать строку, которая длиннее длины записи в файле.

См. также:

**Функция EOF( ), Оператор Get, Оператор Open File, Оператор Print #, Оператор Write #**

## Оператор Randomize

### Назначение

Инициализирует функцию случайных чисел в **MapBasic**.

### Синтаксис

**Randomize** [ **With** *seed* ]

*seed* – целочисленное выражение.

### Описание

Оператор **Randomize** включает генератор случайных чисел, результат которого используется, когда будет вызываться **Функция Rnd( )**. Если оператор Randomize не выполнен, то **Функция Rnd( )** будет образовывать хотя и случайную, но всегда одинаковую последовательность. Это происходит потому, что без оператора **Randomize** каждое новое значение, которое создает **Функция Rnd( )**, зависит от предыдущего значения случайной последовательности.

Оператор **Randomize** изменяет начальное значение для функции Rnd( ), и потому все последующие вызовы этой функции (**Функция Rnd( )**) будут образовывать другую случайную последовательность.

Если оператор Randomize используется с предложением **With**, то для создания начального случайного значения используется псевдослучайный генератор с входным значением *seed*. Если предложение **With** отсутствует, для построения начального случайного числа используется таймер Вашего компьютера. Используйте параметр **With**, если требуется повторяющийся в тестах набор псевдослучайных исходных данных, программа будет создавать "случайные" числа, последовательность которых будет постоянно повторяться.

### Пример:

```
Randomize
```

### См. также:

**Функция Rnd( )**

---

## Функция RegionInfo( )

### Назначение

Эта функция предназначена для определения направления обхода узлов полигона – по часовой или против часовой. Единственный атрибут, который возвращает эта функция, – 'направление обхода' узлов заданного полигона.

### Синтаксис

```
RegionInfo(object, REGION_INFO_IS_CLOCKWISE, polygon_num)
```

Существует единственный параметр этой функции:

|                          |   |
|--------------------------|---|
| REGION_INFO_IS_CLOCKWISE | 1 |
|--------------------------|---|

### Пример:

Если на карте Соединенных Штатов Америки "STATES.TAB" выбрать штат Юта (Utah) и выполнить в окне MapBasic эту команду, то результатом будет F or False, поскольку узлы единственного полигона штата Юта имеют направление обхода против часовой. Узлы полигона штата Колорадо (Colorado) имеют направление обхода по часовой и, в этом случае, будет возвращено значение T или True.

```
print RegionInfo(selection.obj,1,1)
```



## Функция ReadControlValue( )

### Назначение

Считывает состояние элемента активного диалога.

### Синтаксис

`ReadControlValue( id_num )`

*id\_num* – целочисленный идентификатор элемента диалога.

### Возвращаемая величина

Величина типа Integer, Logical, String, Pen, Brush, Symbol или Font. Тип величины зависит от вида элемента диалога.

### Описание

Функция **ReadControlValue( )** возвращает текущее значение одного из элементов активного диалога. Так как функция **ReadControlValue( )** может считывать значения только из открытого диалога, ее можно употреблять только в процедурах-обработчиках элементов диалога.

Параметр *id\_num* должен задавать идентификатор элемента, значение которого прочитает функция. Если значение идентификатора будет равно -1 (минус единица), функция **ReadControlValue( )** вернет значение элемента, который на текущий момент был изменен последним. Чтобы получить информацию о заданном элементе диалога, целочисленное значение идентификатора диалога необходимо передать функции **ReadControlValue( )**.

**Внимание:** Диалог не будет иметь уникального идентификатора, если не включить параметр **ID** предложения **Control** в **Оператор Dialog**. Так как пользователь может одновременно выбрать сразу несколько строк в списке элемента MultiListBox, то чтение значений этого элемента требует многократного вызова функции **ReadControlValue( )**.

В таблице ниже перечислены типы возвращаемых значений разных элементов диалога. Обратите внимание, что для элемента MultiListBox существует специальная процедура чтения текущего значения. Так как пользователь может одновременно выбрать сразу несколько строк в списке элемента MultiListBox, то чтение значений этого элемента требует многократного вызова функции **ReadControlValue( )**.

| Элемент диалога                | Возвращаемое значение ReadControlValue( )  |
|--------------------------------|--|
| EditText - окошко ввода текста | Строка (тип String) длиной до 32767 байт, содержащая текст из окошка; если окошко элемента многострочное, то текст может содержать символы конца строки (код Chr\$(10)). |
| CheckBox - флажки              | Логическое "Да" (TRUE), если флажок установлен, и логическое "Нет" (FALSE), если сброшен.  |

| Элемент диалога                             | Возвращаемое значение ReadControlValue( )   |
|---|---|
| DocumentWindow                              | Целое число идентификатора HWND элемента управления окна. Этот идентификатор должен передаваться в качестве маркера родительского окна в <b>Оператор Set Next Document</b> .  |
| Кнопки-переключатели (RadioGroup)           | Короткое целое число (тип SmallInt), номер выбранной кнопки (начиная с 1).  |
| Всплывающее меню (PopupMenu)                | Короткое целое число (тип SmallInt), номер выбранной строки в списке меню (начиная с 1).  |
| Список (ListBox)                            | Короткое целое число (тип SmallInt), номер выбранной строки в списке (1 – первая строка, 0 – ни одной).   |
| BrushPicker                                 | Величина типа Brush   |
| FontPicker                                  | Величина типа Font  |
| PenPicker                                   | Величина типа Pen   |
| SymbolPicker                                | Величина типа Symbol  |
| Список множественного выбора (MultiListBox) | <p>Целое число (тип Integer), номер выбранной строки в списке. Пользователь может выбрать одну или сразу несколько строк в списке MultiListBox. Но функция <b>ReadControlValue( )</b> может возвращать только одну величину за один вызов.</p> <p>Поэтому для чтения всех значений необходимо функцию вызывать несколько раз. Первый вызов функции дает номер первой строки, выбранной в списке. Следующий – второй, и т. д. Когда все значения будут прочитаны, функция вернет ноль. Если ноль будет получен при первом вызове, то, следовательно, в списке ничего не выбрано.</p> |

#### Ошибки:

Функция вернет код ошибки ERR\_FCN\_ARG\_RANGE, если значение аргумента выходит за пределы, заданные при его определении.

ERR\_INVALID\_READ\_CONTROL, если функция **ReadControlValue( )** вызвана не при активном диалоге.

#### Пример:

Создадим диалог, предлагающий пользователю ввести свое имя. Если после этого пользователь нажмет кнопку "ОК", программа вызовет функцию **ReadControlValue( )**, которая покажет приветствие (Например, "Добро пожаловать, Света!").

```
Declare Sub Main
Declare Sub okhandler
Sub Main
Dialog Title
"Представьтесь, пожалуйста" Control OKButtonPosition 135, 120 Width 50
```

```
Title "OK" Calling okhandler Control CancelButtonPosition 135, 100 Width
50 Title "Отмена" Control StaticText Position 5, 10 Title "Введите ваше
имя:" Control EditTextPosition 55, 10 Width 160Value "(имярек)" Id 23
'arbitrary ID number End Sub
```

```
Sub okhandler ' эта подпрограмма выполнится, если' нажата кнопка "OK Note  
"Добро пожаловать, " + ReadControlValue(23) + "!"End Sub
```

**См. также:**

[Оператор Alter Control](#), [Оператор Dialog](#), [Оператор Dialog Preserve](#), [Оператор Dialog Remove](#)

---

## Оператор ReDim

### Назначение

Изменяет размерность массива.

### Синтаксис

```
ReDim var_name ( newsize ) [ , ... ]
```

*var\_name* – имя массива локальных или глобальных переменных;

*newsize* – целое число, задающее новую размерность: По умолчанию значение равно 32 767.

### Описание

Оператор **ReDim** изменяет размерность, (то есть количество элементов) одного или более уже объявленных массивов. Имя массива переменных, заданное параметром *var\_name*, для объявления которого необходимо предварительно использовать либо [Оператор Dim](#), либо [Оператор Global](#).

Оператор **ReDim** может либо увеличивать, либо уменьшать размер существующего массива. Если программе более не требуется массив заданного размера, оператор **ReDim** может уменьшить размер массива до нуля (этим можно минимизировать объем памяти, необходимый для хранения переменных).

В отличие от других BASIC-языков, MapBasic не позволяет задавать произвольный номер первого элемента массива. Другими словами, первый элемент массива в MapBasic всегда имеет номер 1 (единица).

Если переменные хранились в массиве, а затем размер массива был увеличен оператором **ReDim**, хранившиеся значения не меняются.

**Пример:**

```
Dim names_list(10) As String, cur_size As Integer
' Следующий оператор считывает значение' размерности массива, оператор
ReDim увеличивает' это значение на 10 cur_size = UBound(names_list)ReDim
names_list(cur_size + 10) ' Следующий оператор ReDim обнуляет размерность
' нашего массива. Предположительно, этот массив больше ' не понадобится и
обнуление его размерности ' сэкономит нам ресурсы памяти.
```

```
ReDim names_list(0)
```

Оператор **ReDim** может применяться к массивам переменных сложного типа, составленным при помощи оператора **Type**, и к массивам, которые являются элементами переменных сложного типа.

```
Type customername As String serial_nums(0) As IntegerEnd TypeDim
new_customers(1) As customer ' Сначала увеличим размерность массива
new_customers ' до пяти элементовReDim new_customers(5) ' Теперь изменим
размерность массива serial_nums, ' который является элементом массива
new_customersReDim new_customers(1).serial_nums(10)
```

**См. также:**

**Оператор Dim, Оператор Global, Функция UBound( )**

## Оператор Register Table

### Назначение

Создаёт таблицу MapInfo Professional из электронных таблиц, баз данных, текстовых файлов, растровых изображений и файлов регулярных поверхностей.

### Синтаксис

```
Register Table source_file
{ Type "NATIVE" |
  Type "DBF" [ CharSet char_set ] |
  Type "ASCII" [ Delimiter delim_char ][ Titles ][ CharSet char_set ] |
  Type "WKS" [ Titles ] [ Range range_name ] |
  Type "WMS" Coordsys...
  Type "WFS" [ CharSet char_set ] Coordsys... [ Symbol... ]
    [ Linestyle Pen(...) ] [ Regionstyle Pen(...) Brush(...) ]
  Type "XLS" [ Titles ] [ Range range_name ] [ Interactive ] |
  Type "Access" Table table_name [ Password pwd ] [ CharSet char_set ]}
Type "ODBC"
  Connection { Handle ConnectionNumber | ConnectionString }
  Toolkit toolkitname
  Cache { ON | OFF }
  [ Autokey { ON | OFF }}
  Table SQLQuery
  [ Versioned { ON | OFF }}
  [ Workspace WorkspaceName ]
  [ ParentWorkspace ParentWorkspaceName ]
Type "GRID" | Type "RASTER"
  [ ControlPoints ( MapX1, MapY1 ) ( RasterX1, RasterY1 ),
    ( MapX2, MapY2 ) ( RasterX2, RasterY2 ),
    ( MapX3, MapY3 ) ( RasterX3, RasterY3 )
    [, ... ]
  ]
  [ CoordSys ... ]
Type "FME" [ CharSet char_set ]
  Предложение CoordSys
  Format format type
  Schema featuretype
  [ Use Color ]
  [ Database ]
  [ SingleFile ]
  [ Symbol...]
  [ Linestyle Pen(...) ]
  [ Regionstyle Pen(...) Brush(...) ]
  [ Font ... ]
  Settings string1 [, string2 .. ]
Type "SHAPEFILE" [ CharSet char_set ] CoordSys...
  [ PersistentCache { ON | OFF } ]
  [ Symbol...] [ Linestyle Pen(...) ]
  [ Regionstyle Pen(...) Brush(...) ]
```

[ Into destination\_file ]

*source\_file* – строка, определяющая имя существующей базы данных, электронной таблицы или текстового файла. Если Вы регистрируете таблицу Access, этот аргумент должен идентифицировать доступную базу данных Access.

*char\_set* – имя набора символов; см. раздел, в котором описан [Предложение CharSet on page 136](#);

*delim\_char* – знак разделителя между значениями полей таблицы (только для текстовых ASCII-файлов). Если в файле используется Tab в качестве разделителя, укажите 9. Если используется запятая, укажите 44.

*range\_name* – строка с именем области электронной таблицы (например, “MyTable”) или с ссылкой на ячейки (например, в Excel ячейки могут быть заданы как “Sheet1!R1C1:R9C6” или как “Sheet1!A1:F9”).

*table\_name* – строка, определяющая имя таблицы Access.

*pwd* – пароль базы данных, заданный при включении защиты базы данных;

*ConnectionNumber* – целое значение, которое определяет существующее соединение с ODBC базой данных.

*ConnectionString* – строка, используемая для подключения к серверу базы данных (см. описание функции Server Connect). См. [Функция Server\\_Connect\( \) on page 578](#).

*toolkitname* – “ODBC” или “ORAINET.”

*SQLQuery* –SQL-запрос, использованный для создания таблицы MapInfo.

*CoordSys...* – обязательный параметр CoordSys.

*Format formattype* – строка formattype используется FME для определения формата открываемых данных.

*Schema featuretype* – задает тип свойств (по существу – название схемы).

*Settings* string1 [, string2 .. ] – настройки Safe Software FME, которые могут меняться в зависимости от выбранного формата и необходимых преобразований.

*Use Color* – включает использование сведений о цвете из набора данных

*Database* – указывает, что исходные данные хранятся в базе данных

*SingleFile* – указывает, что исходные данные хранятся в единственном файле

**ControlPoints** – дополнительное предложение, но может быть определено, если тип данных Grid или Raster. Если задано ключевое слово **ControlPoints**, то требуется задать как минимум 3 пары точек координат Map или Raster, которые используются для регистрации изображения. Если заданы контрольные точки **ControlPoints**, то они будут использоваться вместо любых контрольных точек, которые уже ассоциировались с изображением или ассоциировались с регистрационным файлом World.

Для WMS-файлов и шейпфайлов необходим параметр **CoordSys**. При его отсутствии компилятор выдаст сообщение об ошибке. Для файлов других типов предложение **CoordSys** дополнительное, но может быть определено, если тип данных Grid или Raster. Если **CoordSys**

определено, то им будут переписаны любые системы координат, ассоциированные с изображением. Это полезно когда регистрируется растр, имеющий ассоциированный регистрационный файл World.

**PersistentCache On** – определяет сохраняются ли файлы .MAP и .ID, которые генерируются во время открытия шейп файлов, на жестком диске после закрытия таблицы или нет. Если **PersistentCache** установлено на **Off**, то эти файлы .MAP и .ID будут удалены после закрытия таблицы, и будут генерироваться каждый раз при открытии таблицы.

Параметр **Symbol** определяет стиль символа, применяемого для точечных объектов, создаваемых из шейп файла

Параметр **Linestyle Pen** определяет стиль линии, применяемого для линий, создаваемых из шейп файла

**Regionstyle Pen Brush** предложение определяет стиль линии и стиль заливки, применяемый для регионов, создаваемых из шейп файлов

Ключевое слово **Interactive** является дополнительным, но может быть задано, если тип данных это Grid или Raster. Если ключевое слово **Interactive** задано, пользователь получит подсказку при пропуске контрольной точки или информации о проекции. Если ключевое слово **Interactive** не определено, то .TAB файл будет генерироваться без интерактивного ввода пользователем, а сам файл будет создаваться таким образом, как и при выборе в диалоге "Открыть таблицу" команды **ФАЙЛ>ОТКРЫТЬ** при открытии растра. Параметр **Interactive** не применяется при регистрации шейпфайлов, имеющих расширение .SHP.

**Внимание:** Если параметр **Interactive** задан для файлов типа XLS, то при импорте файлов Excel появится диалог "Установка свойств поля".

*destination\_file* – имя будущей таблицы MapInfo table (.TAB file). Строка может включать в себя маршрут, если же он не указан, то файл будет строиться в той же директории, где и исходный файл.

Если параметр **Autokey** задан в состоянии **ON**, то таблица будет зарегистрирована с автоматическим приращением ключа. Если параметр **Autokey** установлен в положение **OFF** или этот параметр пропущен, то таблица будет зарегистрирована без автоматического приращения ключа.

**Versioned** – показывает, что для открываемой таблицы задан (ON) либо нет (OFF) режим контроля версий.

*WorkspaceName* – имя рабочей области таблицы. Чувствительно к регистру.

*ParentWorkspaceName* – имя родительской для текущей рабочей области.

*Charset char\_set* – задавать этот параметр необязательно. Если набор символов не задан MapBasic'ом, то будет использован набор символов системы. Подобный метод используется и для остальных форматов.

*CoordSys...* – обязательный параметр CoordSys.

*Format formattype* – строка formattype используется FME для определения формата открываемых данных.

*Schema featuretype* – задает тип свойств (по существу – название схемы).



*Settings* string1 [, string1 .. ] – настройки Safe Software FME, которые могут меняться в зависимости от выбранного формата и необходимых преобразований.

*Use Color* – включает использование сведений о цвете из набора данных

*Database* – указывает, что исходные данные хранятся в базе данных

*SingleFile* – указывает, что исходные данные хранятся в единственном файле

## Описание

Перед тем, как использовать в MapInfo файлы “неродных” форматов (например, dBASE файл), Вы должны их зарегистрировать. Оператор **Register Table** приводит к тому, что MapInfo проверяет, может ли использоваться этот формат. Далее MapInfo готовит для него файлы-компоненты таблицы (filename.TAB, и др.).

Оператор **Register Table** не копирует и не изменяет оригинал файла данных. Вместо этого, он проверяет данные, определяет типы данных в колонках и создает отдельную таблицу. Таблица не открывается автоматически. Для открытия таблицы надо использовать **Оператор Open Table**.

**Внимание:** Каждый файл данных должен быть зарегистрирован только раз. Так как при **регистрации** создаются файлы-компоненты, то при следующем сеансе работы MapInfo Professional может просто открывать зарегистрированную ранее таблицу (оператор Open), а не регистрировать файл заново оператор **Register Table**.

Предложение **Type** задает формат, в котором был создан файл данных. За словом **Type**, может следовать одна из следующих строк: NATIVE, DBF, ASCII, WKS, XLS, Raster, Grid или Access. Для определенных типов таблиц требуется дополнительная информация. Если тип регистрируемой таблицы это grid, то информация о системе координат будет считана из файла поверхности (grid) и MapInfo-файл .TAB будет создан. Если регистрируется растровый файл, .TAB будет создан в тот момент, когда пользователь выберет “Показать” при открытии растрового изображения в диалоге “Открыть таблицу”.

Если тип регистрируемой таблицы это grid, то информация о системе координат будет считана из файла поверхности (grid) и MapInfo-файл .TAB будет создан. Если регистрируется растровый снимок, то создаваемый .TAB-файл зависит от информации о геопривязки, которая может храниться в файле снимка или в связанном с ним World-файле.

Предложение CharSet определяет набор символов. Параметр *char\_set* должен быть строковой константой, такой как “MacRoman” или “WindowsLatin1”. Если предложение CharSet опущено, MapInfo Professional будет использовать заданный по умолчанию набор символов. Более подробную информацию см. в разделе **Предложение CharSet on page 136**.

Предложение **Delimiter** задает знак разделителя между значениями полей для текстовых файлов. По умолчанию принимается символ табуляции. Предложение **Titles** определяет первую строку данных как названия полей в таблице MapInfo. Предложение **Range** определяет именованную область из исходного файла как базу данных (для электронных таблиц). Предложение **Into** задает в виде строки имя файла .TAB и расположение на диске. Если предложения в операторе нет, имя файла таблицы будет таким же, как имя файла-источника и располагаться он должен в том же каталоге. В случае, если для диска, на котором расположен файл-источник, разрешено только чтение (например, CD-ROM), то файл таблицы будет сохранен где-то в другом месте, где запись разрешена.

### Регистрация таблиц Access

Когда Вы регистрируете таблицу Access, MapInfo проверяет колонку с уникальными индексами. Если уже существует такая колонка, MapInfo регистрирует ее в файле .TAB file. Эта колонка только для чтения.

Если в таблице Access нет такой колонки, MapInfo Professional модифицирует таблицу Access добавляя к ней колонку с именем MAPINFO\_ID с типом данных counter. В этом случае, колонка тоже не отображается в MapInfo.

**Внимание:** Не пытайтесь ни каким способом изменять колонку счетчика. Это автоматически выполняется только MapInfo Professional.

Тип данных баз данных Access транслируется в закрытый тип данных MapInfo. Специальные типы данных Access, такие, как OLE объекты и бинарные поля, в MapInfo не редактируются.

## Регистрация ODBC таблиц

Прежде чем установить прямой доступ к удаленной базе данных, крайне рекомендуется предварительно открыть картографическую таблицу для этой базы данных. В случае если Вы предварительно не откроете карту, вся таблица базы данных будет загружаться целиком, что может занять значительное время.

Откройте карты и установите для нее такой масштаб, чтобы охватить интересующую Вас область. Например, если Вы хотите извлечь из базы данных строки, относящиеся к Калужской области, установите такой масштаб карты, чтобы в окне отображалась Калужская область. В результате, когда Вы откроете таблицу базы данных, загрузятся только те строки, которые попадают в МОП карты (минимальный описывающий прямоугольник), в данном случае строки, относящиеся к Калужской области.

Ниже приводится список известных ограничений, которые могут встретиться при работе с прямым доступом:

- каждая таблица должна иметь единственную уникальную ключевую колонку
- режим быстрого редактирования (FastEdit) не поддерживается
- при работе MS ACCESS, в случаях, когда ключевое поле символьное, не будут отображаться строки, значения которых в ключевой колонке меньше чем полная ширина колонки. То есть если колонка определена как `char (5)` значение 'aaaa' будет выглядеть как удаленная строка.
- при прямом доступе, флажок **"Только чтение"** в диалоге сохранения таблицы будет недоступен.
- Изменения сделанные другим пользователем не будут видны до тех пор, пока окно не будет прокручено или обновлено. Строки, добавленные другим пользователем, не будут видны до тех пор, пока: 1). поиск по минимальному описывающему прямоугольнику (МОП) не возвратит запись или 2). применена команда УПАКОВАТЬ. Кроме того, при включенном режиме "Хранить в памяти" изменения, сделанные другим пользователем могут не показываться в списке до тех пор, пока не будет осуществлено масштабирование карты или сдвиг карты в окне.
- - Возникнут проблемы, если приложение-клиент в одном запросе (командой **SQL-ЗАПРОС** или с помощью MapBasic) обратится к двум или более таблицам SPATIALWARE, хранящимся в разных системах координат. (Эффективнее осуществлять такие выборки с помощью SQL-запросов на сервере). Эта проблема относится к текущей версии.
- при редактировании таблиц Oracle 7, индексированных по десятичному полю размером более 8 байт, возможно аварийное завершение работы MapInfo Professional.
- Если используется сервер Oracle, то индикатор **Autokey** оповещает о состоянии новой возможности – автоматическом приращении ключа.
- если оператор **Cache** имеет значение OFF перед соединением, то будет сгенерировано сообщение об ошибке при компиляции.

### Пример:

В следующем примере будет создан и открыт .tab-файл.

```
Register Table "SMALLINTEGER" TYPE ODBC
TABLE "Select * From "MIPRO"."SMALLINTEGER""
CONNECTION "SRVR=scout;UID=mipro;PWD=mipro "
toolkit "ORAINET"
Autokey ON
```

```
Into  
"C:\projects\data\testscripts\english\remote\SmallIntEGER.TAB"  
Open Table "C:\Projects\Data\TestScripts\English\remote\SmallIntEGER.TAB"  
Interactive  
Map From SMALLINTEGER
```

## Пример FME (Внешние данные)

```
Register Table "D:\MUT\DWG\Data\afrika_miller.DWG" Type "FME" CoordSys
Earth Projection 11, 104, "m", 0 Format "ACAD" Schema "afrika_miller" Use
Color SingleFile Symbol (35,0,16) Linestyle Pen (1,2,0) RegionStyle Pen
(1,2,0) Brush (2,16777215,16777215) Font ("Arial",0,9,0) Settings
"RUNTIME_MACROS", "METAFILE,acad,_EXPAND_BLOCKS,yes,ACAD_IN_USE_BLOCK_HEAD
ER_LAYER,yes,ACAD_IN_RESOLVE_ENTITY_COLOR,yes,_EXPAND_VISIBLE,yes,_BULGES
_AS_ARCS,no,_STORE_BULGE_INFO,no,_READ_PAPER_SPACE,no,ACAD_IN_READ_GROUPS
,no,_IGNORE_UCS,no,_ACADPreserveComplexHatches,no,_MERGE_SCHEMAS,YES",
"META_MACROS", "Source_EXPAND_BLOCKS,yes,SourceACAD_IN_USE_BLOCK_HEADER_LA
YER,yes,SourceACAD_IN_RESOLVE_ENTITY_COLOR,yes,Source_EXPAND_VISIBLE,yes,
Source_BULGES_AS_ARCS,no,Source_STORE_BULGE_INFO,no,Source_READ_PAPER_SPA
CE,no,SourceACAD_IN_READ_GROUPS,no,Source_IGNORE_UCS,no,Source_ACADPreser
veComplexHatches,no", "METAFILE", "acad", "COORDSYS", "", "IDLIST", "" Into
"C:\Temp\afrika_miller.tab"
Open table "C:\Temp\afrika_miller.tab"
Map From "afrika_miller"
```

## Регистрация шейпфайлов

Когда регистрируются шейпфайлы, то они могут открываться в MapInfo Professional исключительно в режиме "только для чтения". Поскольку шейпфайл не содержит в себе информации о системе координат, надо определить предложение **CoorSys**. Также возможно определить стили оформления объектов для отображения в MapInfo Professional. Информация о проекции и стилях хранится как метаданные в TAB файле.

**Внимание:** Параметр **Interactive** не используется при регистрации SHP-файлов.

## Примеры

### Пример 1

```
Register Table "c:\mapinfo\data\rpt23.dbf"
Type "DBF"
Into "Report23"
```

```
Open Table "c:\mapinfo\data\Report23"
```

### Пример 2

```
Open Table "C:\Data\CANADA\Canada.tab" Interactive
Map From Canada
set map redraw off
Set Map Zoom 1000 Units "mi"
set map redraw on
Register Table "odbc_cancaps"
TYPE ODBC
TABLE "Select * From informix.can_caps"
CONNECTION
DSN=ius_adak;UID=informix;PWD=informix;DATABASE=sw;HOST=adak;
SERVER=adak_tli;SERVICE=sqlexec;PROTOCOL=onsoctcp;"
Into
"D:\MI\odbc_cancaps.TAB"
```

```
Open Table "D:\MI\odbc_cancaps.TAB" Interactive
Map From odbc_cancaps
```

### Пример 3:

Регистрация полностью привязанного снимка (обработчик растров может вернуть не менее трех контрольных точек и информацию о проекции).

```
Register Table "GeoRef.tif" type "raster" into "GeoRef.TAB"
```

### Пример 4:

Регистрация снимка со связанным World-файлом, в котором заданы контрольные точки, но нет сведений о проекции.

```
Register Table "RasterWithWorld.tif" type "raster" coordsys earth
projection 9, 62, "m", -96, 23, 29.5, 45.5, 0, 0 into
"RasterWithWorld.TAB"
```

### Пример 5:

Регистрация растрового снимка без контрольных точек и сведений о проекции.

```
Register Table "NoRegistration.BMP" type "raster" controlpoints
(1000,2000) (1,2), (2000,3000) (2, 3), (5000,6000) (5,6) coordsys earth
projection 9, 62, "m", -96, 23, 29.5, 45.5, 0, 0 into "NoRegistration.tab"
```

### Пример 6:

Регистрация шейпфайла.

```
Register Table "C:\Shapefiles\CNTYLN.SHP" TYPE SHAPEFILE Charset
"WindowsLatin1" CoordSys Earth Projection 1, 33 PersistentCache Off
linestyle Pen (2,26,16711935) Into "C:\Temp\CNTYLN.TAB"
Open Table "C:\Temp\CNTYLN.TAB" Interactive
Map From CNTYLN
```

### Пример 7:

В следующем примере будет создан и открыт .tab-файл.

```
Register Table "Gwmusa" TYPE ODBC
TABLE "Select * From ""MIUSER"". ""GWMUSA""""
CONNECTION "SRVR=troyny;UID=miuser;PWD=miuser"
toolkit "ORAINET"
Versioned On
Workspace "MIUSER"
ParentWorkspace "LIVE"
Into "C:\projects\data\testscripts\english\remote\Gwmusa.tab"
Open Table "C:\Projects\Data\TestScripts\English\remote\Gwmusa.TAB"
Interactive Map From Gwmusa
```

### См. также:

[Оператор Open Table](#), [Оператор Create Table](#), [Оператор Server Create Workspace](#),  
[Оператор Server Link Table](#)

## Оператор Relief Shade

### Назначение

Отмывка рельефа открытой таблицы поверхности.

### Синтаксис

#### Relief Shade

```
Grid tablename
Horizontal xy_plane_angle
Vertical incident_angle
Scale z_scale_factor
```

*tablename* – псевдоним поверхности, для которой рассчитывается отмывка.

*xy\_plane\_angle* – дирекционный угол, в градусах, от источника света в горизонтальном *xy* плане. Значение *xy\_plane\_angle* равное нулю, представляет источник света, светящий с Востока. Положительные значения угла отсчитываются против часовой стрелки, так, например источник с Северо-Запада имеет значение *xy\_plane\_angle* равное 135.

*incident\_angle* – угол от источника света над горизонтом или плоскостью *xy*. Значение *Incident\_angle* равное нулю представляет источник, лежащий на линии горизонта. Значение *incident\_angle* равное 90 помещает источник света в зенит.

*z\_scale\_factor* – масштабный фактор, применяемый к параметру *z* для каждой ячейки поверхности. Увеличение *z\_scale\_factor* улучшает эффект отмывки. Это может быть использовано для выявления мелких деталей поверхности.

### Пример:

```
Relief Shade
Grid Lumens
Horizontal 135
Vertical 45
Scale 30
```

## Оператор Reload Symbols

### Назначение

Открывает и перезагружает файл символов MapInfo и тем самым меняет набор символов, используемый MapInfo и показываемый в диалоге НАСТРОЙКИ > СТИЛЬ СИМВОЛА.

### Синтаксис (вариант 1 - версия для символов MapInfo 3.0):

```
Reload Symbols
```

### Синтаксис 2 (Символы из растровых файлов)

```
Reload Custom Symbols From directory
```

*directory* – строка с маршрутом к каталогу.

### Описание

Оператор используется в утилите SYMBOL.MBX, которая дает пользователю возможность создавать свои символы.

**Внимание:** Символы MapInfo 3.0 были введены в MapInfo Professional для Windows 3.0 и поддерживаются во всех последующих версиях MapInfo Professional.

**См. также:**

[Оператор Alter Object](#)

---

## Процедура RemoteMapGenHandler

### Назначение

Имя специальной процедуры, вызываемой тогда, когда клиент OLE Automation вызывает метод MapGenHandler Automation.

### Синтаксис

```
Declare Sub RemoteMapGenHandler Sub RemoteMapGenHandler statement_list  
End Sub, где
```

*statement\_list* – список операторов MapBasic, выполняемых когда клиент OLE Automation вызывает метод MapGenHandler.

### Описание

**RemoteMapGenHandler** – это имя процедуры MapBasic специального назначения, которая вызывается через OLE Automation. Если Вы используете OLE Automation для управления MapInfo, и вызываете метод MapGenHandler, MapInfo вызывает процедуру

**RemoteMapGenHandler** любого работающего приложения MapBasic. Метод MapGenHandler – это компонента механизма MapGen Automation, представленного в MapInfo 4.1.

Метод MapGenHandler Automation берет один аргумент: строку. Внутри процедуры RemoteMapGenHandler Вы можете восстановить строковый аргумент с помощью вызова следующей функции CommandInfo (CMD\_INFO\_MSG) и присвоить результат строковой переменной.

### Пример:

Примером использования **RemoteMapGenHandler** является программа MAPSRVR.MB.



## Процедура RemoteMsgHandler

### Назначение

Процедура, автоматически выполняющаяся при получении командного сообщения из другой программы.

### Синтаксис

```
Declare Sub RemoteMsgHandler Sub RemoteMsgHandler statement_list End
Sub, где
```

*statement\_list* – список операторов процедуры-обработчика.

### Описание

**RemoteMsgHandler** – зарезервированное имя процедуры MapBasic, предназначенной для обслуживания связи между прикладными программами и другими программами. Если выполняется MapBasic-программа, в которой имеется процедура **RemoteMsgHandler**, а другая программа (например, электронная таблица или база данных) передает исполнимую команду “execute”, MapInfo Professional автоматически вызывает процедуру **RemoteMsgHandler**. После этого MapBasic-процедурой вызывается **CommandInfo()** функция, с помощью которой строка с исполнимой командой воспринимается программой.

Для того чтобы остановить выполнение процедуры **RemoteMsgHandler**, если она больше не нужна, может быть использована **Оператор End Program**. Обратно, **Оператор End Program** должна использоваться очень осторожно, чтобы случайно не остановить выполнение процедуры **RemoteMsgHandler**.

### DDE-связь между программами в среде Windows

Если программа Windows поддерживает DDE-связь (Dynamic Data Exchange – динамический обмен данными), то такая программа может инициировать обмен данными с MapInfo Professional. В таком обмене данными внешняя программа выступает в роли клиента (активная сторона обмена), а заданная MapBasic-программа – сервера (пассивная сторона обмена).

Всякий раз, когда DDE-клиент передает исполняемую команду, MapInfo Professional вызывает серверную процедуру **RemoteMsgHandler**. В процедуре **RemoteMsgHandler** можно использовать вызов функции:

```
CommandInfo( CMD_INFO_MSG )
```

для прочтения сообщения. В динамическом обмене данных имя программы, выступающей в роли сервера (например, "C:\MAPBASIC\DISPATCH.MBX") должно использоваться как объект (topic) обмена с помощью процедуры **RemoteMsgHandler**.

### См. также:

**Оператор DDEExecute, Функция DDEInitiate(), Процедура SelChangedHandler, Процедура ToolHandler, Процедура WinChangedHandler, Процедура WinClosedHandler**

## Функция RemoteQueryHandler( )

### Назначение

Функция, которая вызывается, когда MapBasic-программа работает DDE-сервером и получает от DDE-клиента внешний “считывающий” запрос.

### Синтаксис

```
Declare Function RemoteQueryHandler( ) As String
```

```
Function RemoteQueryHandler( ) As String  
    statement_list  
End Function
```

*statement\_list* – группа операторов, которую нужно выполнить при получении “считывающего” запроса **peek**.

### Описание

Функция **RemoteQueryHandler( )** работает с механизмом DDE (Dynamic Data Exchange). Более подробная информации о DDE содержится в *Руководстве пользователя MapBasic*. Внешняя программа может начать сеанс DDE-связи с Вашей MapBasic-программой. При инициации связи внешняя программа использует значение “MapInfo” как имя DDE-программы и имя Вашей MapBasic-программы как DDE-тему (topic). Как только связь установится, клиент (внешняя программа) может посылать считывающие запросы к MapBasic-программе (серверу).

Для обработки таких запросов в Вашу MapBasic-программу нужно включать обработчик **RemoteQueryHandler( )**. Как только от клиента поступает считывающий запрос, MapInfo автоматически вызывает функцию **RemoteQueryHandler( )**. Следующие считывающие запросы ожидают своей очереди, т.е. выполняются после того, как **RemoteQueryHandler( )** вернет значение.

**Внимание:** DDE-клиент может считывать значения глобальных переменных MapBasic-программы без помощи **RemoteQueryHandler( )**. Если клиент выдаст считывающий запрос к глобальной переменной MapBasic, MapInfo автоматически возвратит требуемое значение вместо того, чтобы обратиться к **RemoteQueryHandler( )**. Другими словами, если данные, которые требуется передать, хранятся в глобальных переменных, то в **RemoteQueryHandler( )** не нужна.

**Пример:**

В следующем примере вызывается **CommandInfo( ) функция**, чтобы определить, какой объект требует DDE-клиент. Если объект называется "code1", то действия продолжаются по одному сценарию, если нет, то по другому.

```
Function RemoteQueryHandler( ) As String
    Dim s_item_name As String

    s_item_name = CommandInfo(CMD_INFO_MSG)

    If s_item_name = "code1" Then
        RemoteQueryHandler = custom_function_1( )
    Else
        RemoteQueryHandler = custom_function_2( )
    End If

End Function
```

**См. также:**

**Функция DDEInitiate( ), Процедура RemoteMsgHandler**

---

## Оператор Remove Cartographic Frame

**Назначение**

Оператор Remove Cartographic Frame позволяет Вам удалять разделы из существующей картографической легенды, созданной оператором Create Cartographic Legend. **Оператор Create Cartographic Legend**

**Синтаксис****Cartographic Frame**

```
[ Window legend_window_id ]
Id frame_id, frame_id, frame_id, ...
```

*legend\_window\_id* – это целочисленный идентификатор окна, который Вы можете получить при вызове функций **Функция FrontWindow( )** и **Функция WindowID( )**.

*frame\_id* – индекс ID-раздела легенд. Вы не можете использовать здесь имя слоя. Например, три раздела легенды могут иметь идентификаторы ID 1, 2 и 3.

**См. также:**

**Оператор Add Cartographic Frame, Оператор Alter Cartographic Frame, Оператор Create Cartographic Legend, Оператор Set Cartographic Legend**

## Оператор Remove Map

### Назначение

Удаляет один или несколько слоев карты.

### Синтаксис

```
Remove Map [ Window window_id ]
           Layer map_layer [ , map_layer ... ] [ Interactive ]
```

*window\_id* – идентификатор окна Карты, целое число, для того чтобы его получить используются: [Функция FrontWindow\( \)](#) или [Функция WindowID\( \)](#).

*map\_layer* задает слой Карты, который требуется удалить; смотрите примеры ниже.

### Описание

Оператор **Remove Map** удаляет один или более слоев из окна Карты. Если оператор не содержит идентификатора *window\_id*, оператор будет работать с тем окном, которое является окном Карты и лежит выше остальных.

Параметр *map\_layer* может быть либо целым числом, большим нуля, либо строкой с именем таблицы, либо ключевым словом **Animate**, как показано в следующей таблице.

| Примеры                     | Описание  |
|-----------------------------|---|
| Remove Map Layer оператор 1 | Единица обозначает самый верхний некосметический слой. Данный оператор удаляет из Карты слой, идущий сразу за Косметическим слоем. Если параметр <i>map_layer</i> был бы “1, 2”, то были бы удалены два верхних некосметических слоя. |
| Remove Map Layer "Zones"    | Оператор удаляет слой Zones (то есть тот, который в списке слоев обозначен именем “Zones”).   |
| Remove Map Layer оператор 1 | Оператор удаляет первый тематический слой, построенный на основе слоя Zones.  |
| Remove Map Layer Animate    | Оператор удаляет слой анимации. Информация об анимационных слоях содержится в разделе: <a href="#">"Оператор Add Map on page 77"</a> .  |

Если используется ключевое слово **Interactive** и если удаление слоя ведет к потере подписей или тематических объектов, то MapInfo выведет на экран диалог, который позволит сохранить пользователю Рабочий Набор перед удалением слоя. Если ключевое слово в операторе **Interactive** опущено, то пользователь не предупреждается о потере.

Удаляя слой, который отображает данные некоторой открытой таблицы, оператор **Remove Map** не закрывает ее. Если оператор **Remove Map** удаляет последний не косметический слой окна карты, MapInfo Professional закроет это окно автоматически.

См. также:

[Оператор Create Map](#), [Оператор Map](#), [Оператор Set Map](#)

---

## Оператор Rename File

### Назначение

Изменяет имя открытого файла.

### Синтаксис

```
Rename File old_filespec As new_filespec
```

*old\_filespec* – строка с именем файла (и, если необходимо, DOS-маршрут); файл не должен быть открыт;

*new\_filespec* – строка с новым именем файла (и, если необходимо, маршрутом).

### Описание

Оператор **Rename File** переименовывает файл.

Параметр *new\_filespec* задает спецификацию с новым именем файла. Если *new\_filespec* маршрут отличается от оригинального маршрута, то MapInfo перенесет файл в заданный каталог.

### Пример:

```
Rename File "startup.wor" As "startup.bak"
```

См. также:

[Оператор Rename File](#), [Оператор Save File](#)

---

## Оператор Rename Table

### Назначение

Изменяет имя (и расположение на диске) файла таблицы MapInfo.

### Синтаксис

```
Rename Table table As newtablespect
```

*table* – имя открытой таблицы;

*newtablespect* – строка с новым именем файла таблицы (и, если необходимо, маршрутом).

### Описание

Оператор **Rename Table** позволяет изменить либо имя таблицы, либо ее расположение на диске, либо то и другое для таблицы, открытой под рабочим именем *table*.

Параметр *newtablespec* задает новое имя файла таблицы.. Если в параметре *newtablespec* задано имя каталога, то MapBasic, помимо переименования, переместит таблицу в заданный каталог. Оператор **Rename Table** переименовывает физически все файлы, которые являются компонентами таблицы. Так как оператор работает с файлами таблицы непосредственно на диске, то результат его работы отменить нельзя.

**Внимание:** Операция может повлиять на рабочие наборы. Если до переименования были сохранены файлы Рабочих Наборов, переименования его составляющих могут повлиять на их загрузку, так как Рабочие Наборы могут искать файлы таблиц там, где их нет.

Не следует с помощью оператора **Rename Table** назначать имена для строки заголовка таблицы. Этого можно добиться, применяя **Оператор Open Table** с дополнительным параметром **As**.

Оператор **Rename Table** нельзя применить к таблице, которая имеет в своем определении слово "View". Например, нельзя переименовывать таблицы улиц (стандарта StreetInfo), такие, как SF\_STRTS, состоящие фактически из двух связанных таблиц, (в нашем случае SF\_STRT1 и SF\_STRT2). Оператор также нельзя применять к временным таблицам (Например, QUERY1). Вы не можете переименовывать временные таблицы запросов (такие как QUERY1). Нельзя также переименовывать таблицы с несохраненными изменениями. Перед выполнением оператора Rename Table в этом случае надо сохранить или отменить удаление (операторами Commit и Rollback).

### Пример:

Переименуем таблицу CASANFRA в SF\_HIWAY, используя оператор Rename Table.

```
Open Table "C:\DATA\CASANFRA.TAB"  
Rename Table CASANFRA As "SF_HIWAY.TAB"
```

В этом примере таблица переименовывается из CASANFRA в SF\_HIWAY и перемещается на другой каталог.

```
Open Table "C:\DATA\CASANFRA.TAB"  
Rename Table CASANFRA As "c:\MAPINFO\SF_HIWAY"
```

### См. также:

**Оператор Close Table, Оператор Drop Table**

---

## Оператор Reproject

### Назначение

Позволяет определить, какая колонка таблицы появится следующей при открытии окна Списка. Этот оператор вызывал возражения.

## Resume statement

### Назначение

Выход из процедуры-обработчика ошибок, назначенного оператором **OnError**.

### Синтаксис

```
Resume { 0 | Next | label }
```

*label* – строка с меткой в тексте некоторой подпрограммы или функции.

### Предупреждение

Вы не можете использовать оператор **Resume** в окне MapBasic.

### Описание

Оператор **Resume** позволяет выйти из процедуры обработки ошибок.

Группа операторов после метки, которую задает **Оператор OnError**, выполняется, если при работе программы произошла ошибка. Обычно, эта процедура-обработчик ошибок может содержать один или несколько операторов **Resume**. Оператор **Resume** выводит программу MapBasic из процедуры обработки ошибок.

Несколько форм оператора **Resume** позволяют переходить к определенному оператору MapBasic после выхода из процедуры обработки ошибок:

Оператор **Resume 0** возвращает управление программой туда, откуда был вызван обработчик ошибок с повторением оператора, в котором произошла ошибка.

Оператор **Resume Next** осуществляет перевод управления программой к оператору, следующему за оператором, где произошла ошибка.

Оператор **Resume label** передает управление программой строке с меткой, заданной параметром label. Метка label должна находиться в теле какой-нибудь процедуры. Прописные и строчные символы в имени метки не различаются.

### Пример:

```
...
  OnError GoTo no_states
  Open Table "states"
  Map From states
after_mapfrom:
...
  End Program
no_states:
  Note "Could not open States; no Map used."
  Resume after_mapfrom
```

### См. также:

**Функция Err( ), Оператор Error, Функция Error\$( ), Оператор OnError**

### Функция RGB( )

#### Назначение

Возвращает значение цвета в системе RGB, вычисляя из установок концентрацию красного, зеленого и синего цветов.

#### Синтаксис

**RGB**( *red*, *green*, *blue* )

*red* – числовое выражение в диапазоне от 0 до 255, определяющее концентрацию красного;

*green* – числовое выражение в диапазоне от 0 до 255, определяющее концентрацию зеленого;

*blue* – числовое выражение в диапазоне от 0 до 255, определяющее концентрацию синего;

#### Возвращаемая величина

Целое число типа Integer.

#### Описание

MapBasic использует цвет в операторах как часть установки стиля линии, штриха или символа (например, **Оператор Create Point**). Каждый цвет в стилях линии и штриха задается целым числом, понимаемым как RGB-величина. Функция **RGB( )** позволяет получить это число.

В таком представлении цвета используется сочетание трех компонент – красного, зеленого и синего цветов. Соответственно, в функции **RGB( )** присутствуют три параметра – red, green и blue. Каждая цветовая компонента должна иметь целочисленное значение в диапазоне от 0 до 255, включительно.

RGB-величина вычисляется по формуле:

$$( red * 65536 ) + ( green * 256 ) + blue$$

Замечание: Файл стандартных определений MAPBASIC.DEF с помощью оператора **Define** назначает имена для основных 8 цветов: BLACK – черный; WHITE – белый; RED – красный; GREEN – зеленый; BLUE – синий; CYAN – голубой; MAGENTA – розовый; YELLOW – желтый. Если Вам необходимо задать красный цвет, то можно использовать имя RED вместо вызова функции **RGB(255,0,0)**.

#### Пример:

```
Dim red,green,blue,color As Integer
red = 255
green = 0
blue = 0
color = RGB(red, green, blue)

' the RGB value stored in the variable: color
' will represent pure, saturated red.
```



См. также:

Предложение Brush, Предложение Font, Предложение Pen, Предложение Symbol

---

## Функция Right\$( )

### Назначение

Извлекает из правой части строки определенное количество символов.

### Синтаксис

**Right\$(** *string\_expr*, *num\_expr* **)**

*string\_expr* - выражение, результат которого есть строка.

*num\_expr* – числовое выражение

### Возвращаемая величина

Строка

### Описание

Функция **Right\$( )** возвращает строку, составленную из *num\_expr* правых символов строки *string\_expr*.

Параметр *num\_expr* должен принимать положительное целочисленное значение, ноль или больше. Если значение параметра *num\_expr* нецелочисленное, то MapBasic округлит его до целого. Если параметр *num\_expr* равен нулю, то функция **Right\$( )** возвращает строку нулевой длины. Если значение параметра *num\_expr* больше, чем длина строки *string\_expr*, то результатом выполнения функции **Right\$( )** будет полная строка *string\_expr*.

### Пример:

```
Dim whole, partial As String whole = "Казахстан"partial = Left$(whole, 4) '
теперь переменная partial содержит строку "Казах"
```

См. также:

Функция Left\$( ), Функция Mid\$( )

---

## Функция Rnd( )

### Назначение

Генератор случайных чисел.

### Синтаксис

**Rnd(** *list\_type* **)**

*list\_type* – целое число, задающее режим случайной последовательности.

### Возвращаемая величина

Вещественное число в диапазоне от 0 до 1 (не включая). Вещественная величина типа Float.

### Описание

Функция **Rnd( )** возвращает случайное десятичное число, больше нуля, но меньше единицы.

Обычно используется в форме **Rnd(1)**, возвращающее случайное число. Последовательность случайных чисел всегда одна и та же, кроме случая, когда в программе используется **Оператор Randomize**. Любой положительный аргумент *list\_type* приводит к тем же результатам.

Реже используется другой вариант, когда этот оператор используется в форме: **Rnd(0)**; в этом случае возвращается то же самое случайное число, которое уже была порождено функцией **Rnd( )**. Такая возможность использования этой функции предусмотрена, в основном, для отладки программ.

Еще реже используется вариант, когда в качестве аргумента *list\_type* задается отрицательное значение, например, **Rnd(-1)**. При заданном отрицательном значении, функция **Rnd( )** всегда возвращает одно и то же число, независимо от того использовался или нет **Оператор Randomize**. Такая возможность использования этой функции предусмотрена, в основном, для отладки программ.

### Пример:

```
Chknum = 10 * Rnd(1)
```

### См. также:

**Оператор Randomize**

---

## Оператор Rollback

### Назначение

Отменяет все изменения в таблице, которые еще не были сохранены на диске.

### Синтаксис

```
Rollback Table tablename
```

*tablename* – имя открытой таблицы;

## Описание

Если в таблице были сделаны изменения, но эти изменения не были сохранены, оператор **Rollback** их отменяет. Оператор выполняет такие же действия, как команда **ФАЙЛ > ВОССТАНОВИТЬ** в MapInfo.

**Внимание:** Когда оператор Rollback применяется к таблице запроса, MapInfo Professional автоматически отменяет все изменения в постоянной таблице, на основе которой была сделана временная таблица (кроме тех случаев, когда в процессе запроса использовалось объединение или когда запрос возвратил обобщенные значения, т.е. использовался **Оператор Select** с параметром **Group By**).

Таблица запросов (например, "QUERY1") обычно отражает текущее содержание какой-то постоянной таблицы (например, "World"). Если применить оператор Rollback к таблице "QUERY1", MapInfo Professional отменит любые изменения, сделанные в исходной таблице "World".

Оператор **Rollback**, использованный по отношению к связанной таблице, отменит все несохраненные изменения такой таблицы и вернет её в состояние, в котором она была до изменений.

## Пример:

```
If keep_changes Then
    Table towns
Else
    Rollback Table towns
End If
```

## См. также:

**Оператор Commit Table**

---

## Функция Rotate( )

### Назначение

Позволяет вращать объект (не текстовый) вокруг центра.

### Синтаксис

**Rotate**( *object*, *angle* )

*object* – объект, который может вращаться. Этот объект не может быть текстовым.

*angle* – вещественная величина, определяющая угол (в градусах), на который поворачивается объект.

### Возвращаемая величина

Повернутый объект.

### Описание

Функция **Rotate( )** поворачивает все типы объектов, кроме текстовых, при этом исходный объект не изменяется.

Для поворота текстовых объектов используйте оператор **Alter Object OBJ\_GEO\_TEXTANGLE statement**.

Если поворачивается дуга, эллипс, прямоугольник и скругленный прямоугольник, то результирующий объект будет преобразован в полилинию/полигон для того, чтобы узлы могли быть повернуты.

### Пример:

```
dim RotateObject as object
Open Table "C:\MapInfo_data\TUT_USA\USA\STATES.TAB"
map from states
select * from States where state = "IN"
RotateObject = rotate(selection.obj, 45)
insert into states (obj) values (RotateObject)
```

### См. также:

[Функция RotateAtPoint\( \)](#)

---

## Функция RotateAtPoint( )

### Назначение

Позволяет вращать объект (не текстовый) вокруг заданной точки.

### Синтаксис

**RotateAtPoint( *object*, *angle*, *anchor\_point\_object* )**

*object* – объект, который может вращаться. Этот объект не может быть текстовым.

*angle* – вещественная величина, определяющая угол (в градусах), на который поворачивается объект.

*anchor\_point\_object* – объект-точка, вокруг которой поворачиваются узлы вращаемого объекта.

### Возвращаемая величина

Повернутый объект.

### Описание

Функция **RotateAtPoint( )** поворачивает все типы объектов, кроме текстовых, при этом исходный объект не изменяется.

Для поворота текстовых объектов используйте оператор **Alter Object OBJ\_GEO\_TEXTANGLE statement**.

Если поворачивается дуга, эллипс, прямоугольник и скругленный прямоугольник, то результирующий объект будет преобразован в полилинию/полигон для того, чтобы узлы могли быть повернуты.

### Пример:

```
dim RotateAtPointObject as object
dim obj1 as object
dim obj2 as object
Open Table "C:\MapInfo_data\TUT_USA\USA\STATES.TAB" ]
map from states
select * from States where state = "CA"
obj1 = selection.obj
select * from States where state = "NV"
obj2 = selection.obj
oRotateAtPointObject = RotateAtPoint(obj1 , 65, centroid(obj2))
insert into states (obj) values (RotateAtPointObject )
```

### См. также:

**Функция Rotate( )**

---

## Функция Round( )

### Назначение

Округляет число с заданной точностью.

### Синтаксис

**Round**( *num\_expr*, *round\_to* )

*num\_expr* – числовое выражение

*round\_to* – число, определяющее точность округления переменной *num\_expr*.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Round( )** возвращает округленное значение от числа, полученного в результате вычисления выражения *num\_expr*.

Число *round\_to* задает точность округления. Функция **Round( )** округляет переменную *num\_expr* до ближайшего, кратного *round\_to*. Например, если параметр *round\_to* равен 0.01, то число *num\_expr* будет округлено до сотых. Если *round\_to* равен 5, то MapInfo Professional вернет значение, ближайшее к *num\_expr* и кратное пяти.

### Пример:

```
Dim x, y As Float
x = 12345.6789

y = Round(x, 100)
' y now has the value 12300

y = Round(x, 1)
' y now has the value 12346

y = Round(x, 0.01)
' y now has the value 12345.68
```

### См. также:

[Оператор Fix\( \)](#), [Функция Format\\$\( \)](#), [Функция Int\( \)](#)

---

## Функция RTrim\$( )

### Назначение

Удаляет пробелы в конце строки.

### Синтаксис

**RTrim\$( *string\_expr* )**, где  
*string\_expr* - выражение, результат которого есть строка.

### Возвращаемая величина

Строка

### Описание

Функция **RTrim\$( )** возвращает строку, заданную параметром *string\_expr*, но без пробелов в конце строки.

### Пример:

```
Dim s_name As String
s_name = RTrim$("Mary Smith ")

' s_name now contains the string "Mary Smith"
' (no spaces at the end)
```

### См. также:

[Функция LTrim\\$\( \)](#)

---

## Оператор Run Application

### Назначение

Загружает прикладную программу или Рабочий Набор в MapInfo, окна которого будут добавлены к уже открытым.

### Синтаксис

**Run Application** *file*

*file* – имя файла прикладной программы или Рабочего Набора.

### Описание

Оператор **Run Application** запускает программу MapBasic или загружает Рабочий Набор. Выполняя оператор **Run Application**, одна прикладная программа, написанная на MapBasic, может запустить на выполнение другую программу, определенную параметром *file*. Оператор **Run Application** не может запускать файлы с расширением .MB, то есть файлы, содержащие тексты программ. Для остановки одной программы, запущенной оператором **Run Application**, из другой используется **Оператор Terminate Application**.

### Пример:

Следующий оператор загружает программу REPORT.MBX:

```
Run Application "C:\MAPBASIC\APP\REPORT.MBX"
```

Следующий оператор загружает Рабочий Набор, PARCELS.WOR:

```
Run Application "Parcels.wor"
```

### См. также:

**Оператор Run Command, Оператор Run Menu Command, Оператор Run Program, Оператор Terminate Application**

---

## Оператор Run Command

### Назначение

Выполняет оператор MapBasic, заданный строкой.

### Синтаксис

**Run Command** *command*

*command* – строка символов, представляющая оператор MapBasic.

### Описание

Оператор **Run Command** интерпретирует строку *command* в оператор MapBasic и выполняет его.

Оператор **Run Command** имеет некоторые ограничения, обусловленные тем, что интерпретация происходит уже из откомпилированной программы. *Переменные* не могут быть заданы в строке, т. к. может не произойти подстановки их значений. Нельзя использовать оператор **Run Command**, для того чтобы повторно применить **Оператор Dialog**. К тому же, в строке *command* не могут быть использованы имена переменных; то есть, переменные не могут появляться в кавычках. Например, следующая группа операторов не будет работать, так как указание на переменные *x* и *y* попадает в строку, ограниченную кавычками:

```
' этот пример работать НЕ будет
Dim cmd_string As String
Dim x, y As Float
cmd_string = " x = Abs(y) "
Run Command cmd_string
```

Однако, в строке *command* можно использовать имена переменных.



В следующем примере, строка *command* построена из выражений, в которых используются строковые переменные.

```
'а этот пример БУДЕТ работатьDim cmd_string As String Dim map_it,
browse_it As Logical Open Table "world"If map_it Thencmd_string = "Map
From "Run Command cmd_string + "world" End IfIf browse_it Thencmd_string
= "Browse * From " Run Command cmd_string + "world" End If
```

### Пример:

Оператор **Run Command** представляет гибкий путь выполнения операторов, аргументы которых представляются списками переменной длины. Так, **оператор Map From** может включать в себя имя одной таблицы или список имен таблиц через запятую. Программа может интерпретировать оператор во время выполнения, чтобы определить (используя данные, введенные пользователем), сколько таблиц включить в карту.Оператор Run Command позволяет на ходу сконструировать и выполнить оператор **Map From**. Один способ выполнения этого заключается в следующем: создать строку текста при выполнении программы и передать команду оператору **Run Command**.

```
Dim cmd_text As String
Dim cities_wanted, counties_wanted As Logical

Open Table "states"
Open Table "cities"
Open Table "counties"

cmd_text = "states" ' always include STATES layer

If counties_wanted Then
    cmd_text = "counties, " + cmd_text
End If

If cities_wanted Then
    cmd_text = "cities, " + cmd_text
End If

Run Command "Map From " + cmd_text
```

Следующий пример показывает как можно дублировать окно Карты с помощью функции WindowInfo( ) и оператора Run Command. Этот **Функция WindowInfo( )** возвращает строку с операторами MapBasic; а оператор **Run Command** выполняет эту строку.

```
Dim i_map_id As Integer

' First, get the ID of an existing Map window
' (assuming the Map window is the active window):
i_map_id = FrontWindow( )

' Now clone the active map window:
Run Command WindowInfo(i_map_id, WIN_INFO_CLONEWINDOW)
```

**См. также:**

[Оператор Run Application](#), [Оператор Run Menu Command](#), [Оператор Run Program](#)

---

## Оператор Run Menu Command

### Назначение

Позволяет вызвать стандартную команду MapInfo Professional так, как если бы Вы указали на ее имя в списке меню. Может быть использован для выбора кнопки панели инструментов.

### Синтаксис

**Run Menu Command** { *command\_code* | **ID** *command\_ID* }

*command\_code* – целочисленный код из файла MENU.DEF, соответствующий коду команды из действующей системы меню MapInfo или кнопке;

*command\_ID* – число, представляющее созданный элемент меню или кнопку.

### Описание

Если Ваша программа включает в себя файл стандартных определений *MENU.DEF*, то можно использовать в операторе Run Menu Command имена кодов *command\_code*, которые определены в вышеупомянутом файле при помощи оператора Define. Например, для вызова команды **ФАЙЛ > НОВАЯ ТАБЛИЦА** применяется следующий оператор:

```
Run Menu Command M_FILE_NEW
```

Если Ваша программа включает в себя файл стандартных определений MENU.DEF, то можно использовать в операторе Run Menu Command имена кодов, которые определены в вышеупомянутом файле при помощи оператора Define. Например, для вызова команды **ФАЙЛ > НОВАЯ ТАБЛИЦА** применяется следующий оператор:

```
Run Menu Command M_TOOLS_SEARCH_RADIUS
```

Для выбора созданных кнопки или команды в меню (то есть кнопка или элемент меню, созданные и обрабатываемые программой MapBasic), используется предложение **ID**.

Например, если Ваша программа создала кнопку инструмента и выполнила следующий оператор:

```
Alter ButtonPad ID 1 Add
  ToolButton
    Calling sub_procedure_name
    ID 23
    Icon MI_ICON_CROSSHAIR
```

... то кнопка инструмента имеет идентификатор 23. Следующий оператор выбирает эту кнопку:

```
Run Menu Command ID 23
```

Оператор **Run Menu Command** позволяет вызвать отдельно один из этих диалогов. Например, следующий оператор показывает поддиалог “**MapInfo Professional Tutorial on the Web...**”:

```
Run Menu Command M_HELP_MAPINFO_WWW_TUTORIAL
```

Диалог “Режимы” MapInfo Professional – это специальный случай. Этот диалог имеет несколько кнопок, которые вызывают другие поддиалоги. Оператор **Run Menu Command** позволяет вызвать отдельно один из этих диалогов. Например, следующий оператор показывает поддиалог “Режимы окна Карты”:

```
Run Menu Command M_EDIT_PREFERENCES_MAP.
```

Вы можете **обратить выборку**, используя следующую команду:

```
Run Menu Command M_QUERY_INVERTSELECT
```

Начиная с 6.0, доступ к Настройкам страницы в **Настройки > Режимы > Принтер** осуществляется с использованием следующего синтаксиса:

```
RUN MENU COMMAND M_EDIT_PREFERENCES_PRINTER
```

Or (оператор Или)

```
RUN MENU COMMAND 217
' if running from MapBasic window
```

**См. также:**

**Оператор Run Application, Оператор Run Program**

---

## Оператор Run Program

### Назначение

Выполняет другие программы.

### Синтаксис

```
Run Program program_spec
```

*program\_spec* – командная строка, задающая имя программы и, если необходимо, список аргументов.

### Описание

Если программа, заданная строкой *program\_spec*, не является программой для Windows, то MapBasic сначала создаст копию командного процессора (DOS shell), а потом будет загружена DOS-программа в его среде. Если параметр *program\_spec* имеет значение "COMMAND.COM", MapBasic только откроет окно командного процессора. Если из прикладной программы была запущена программа для DOS, то для возвращения в MapInfo надо ввести команду "Exit". Даже если Вы запустили DOS-программу оператором **Run Program**, Windows продолжает управлять компьютером: то есть может параллельно поддерживать другие Windows-программы и даже саму MapBasic-программу в многозадачном режиме. Возникающие в этой ситуации конфликты Ваша прикладная программа обычно не может разрешить. Поэтому, операторы MapBasic, выполняемые после **Run Program**, не должны использовать любые допущения о статусе вызванной программы.

Поэтому в программе должны быть предприняты меры предосторожности, чтобы избежать конфликтов многозадачности при выполнении оператора **Run Program**. Один из способов избежать таких конфликтов: поместить оператор **Run Program** в конце программы или группы операторов, которые могут конфликтовать с вызываемой программой. Например, Вы создаете элемент меню, который вызывает процедуру-обработчик, в которой в самом конце используется оператор **Run Program**.

### Пример:

Оператор **Run Program** загружает текстовый редактор Windows, который называется "Notepad", и открывает в нем текстовый файл "THINGS.2DO".

```
Run Program "notepad.exe things.2do"
```

Следующий оператор вызывает DOS-команду.

```
Run Program "command.com /c dir c:\mapinfo\ > C:\temp\dirlist.txt"
```

### См. также:

[Оператор Run Application](#), [Оператор Run Command](#), [Оператор Run Menu Command](#)

## Оператор Save File

### Назначение

Копирует файл.

### Синтаксис

**Save File** *old\_filespec* **As** *new\_filespec* [ **Append** ], где

*old\_filespec* – строка с именем (и, если необходимо, с маршрутом) существующего на диске файла, который не должен быть открыт;

*new\_filespec* - строка с именем (и, если необходимо, с маршрутом), под которым будет создана копия; при этом файл с этим именем не должен быть открыт.

### Описание

Оператор **Save File** копирует файл. Файл при этом должен быть закрыт для операций ввода/вывода.

Если в операторе используется ключевое слово **Append** и параметр *new\_filespec* задает имя и маршрут уже существующего файла, то содержимое файла *old\_filespec* будет дописано в конец файла *new\_filespec*.

Не надо использовать оператор **Save File** для копирования файлов, являющихся компонентами таблицы (таких как *filename.tab*, *filename.map* и т. п.). Для копирования таблиц правильно будет использовать оператор Commit Table...As.

Оператор **Save File** не может копировать файл сам в себя.

### Пример:

```
Save File "settings.txt" As "settings.bak"
```

### См. также:

[Оператор Kill](#), [Оператор Rename File](#)

## Оператор Save MWS

### Назначение

Этот оператор позволяет сохранять текущий рабочий набор в виде файла MWS в формате XML для использования в приложениях MapXtreme. К этим MWS-файлам можно организовывать совместный доступ с разных платформ способами, недоступными при работе с рабочими наборами.

### Синтаксис

**Save MWS Window** ( *window\_id* [ , *window\_id* ... ] ) **Default**  
*default\_window\_id* **As** *filespec*, где

*window\_id* – идентификатор окна, целое число;

*default\_window\_id* - целочисленный идентификатор окна Карты записываемый в MWS в качестве Карты по умолчанию.

### Описание

MapInfo Professional позволяет сохранять карты в ваших рабочих наборах в формате XML для использования в приложениях MapXtreme. При сохранении рабочего набора в формат MWS, сохраняются только окна карты и легенд. Все остальные окна отбрасываются, поскольку приложения MapXtreme не могут прочитать эту информацию. Сохраненный в этом формате рабочий набор может быть открыт при помощи программы Workspace Manager, которая устанавливается вместе с MapXtreme, или при помощи приложения, созданного с использованием MapXtreme. Файл является обычным XML-файлом, так что вы можете работать с ним, используя любой редактор или средство просмотра XML файлов. MWS-файлы, созданные MapInfo Professional 7.8 или более поздних версий, могут валидироваться по схемам, поставляемым с MapXtreme.

**Внимание:** Вы не сможете прочитать MWS-файлы, созданные MapInfo Professional версии 7.8 и старше.

В MapInfo Professional можно настроить модификатор тематической карты таким образом, что тематическая карта будет показана, даже если отключить показ слоя, на основе которого построена эта тематическая карта. В MapXtreme модифицированные тематические карты (плотности точек, диапазонов, отдельных значений) будут показаны, если слой, на основе которого построены такие карты, видим. Для того чтобы гарантировать, что модификаторы видимости, отмеченные в MapInfo Professional, в программах типа "Workspace Manager" будут показаны правильно, мы принудительно включаем показ слоя, на основе которого построены тематические карты.

Важно помнить, что многие операторы и функции MapBasic не переводятся в формат MWS. Ниже показано какие элементы карты могут быть сохранены в MWS-файле, а какие нет. Подробнее о совместимости MapBasic и MISQL смотрите в *Руководстве пользователя MapInfo Professional*.

### Что хранится в файлах MWS

В файлах рабочих наборов MWS содержится следующая информация:

- Имя таблицы и синоним
- Информация о системе координат
- Центр карты и масштаб
- Список слоев в заданном порядке
- Размер карты в пикселах (ширина и высота)
- Метод изменения границ рамки карты
- Принудительное изменение стилей оформления объектов
- Принудительное изменение стилей оформления растровых снимков
- Подписи и сделанные в них изменения
- Подписи, основанные на выражениях;
- Тематические картодиаграммы, основанные на выражениях;

- Измененные подписи;
- Запросы с соответствующими окнами карт;
- Тематические картодиаграммы отдельных значений
- Тематические картодиаграммы плотности точек
- Тематические картодиаграммы значков
- Тематические столбчатые картодиаграммы
- Тематические картодиаграммы диапазонов
- Тематические круговые картодиаграммы
- Тематические картодиаграммы поверхностей в виде слоев поверхностей MapXtreme с принудительно заданным стилем оформления
- Тематические выражения и выражения для создания подписей по колонке атрибутов.

### Что не хранится в файлах MWS

В файлах рабочих наборов MWS не содержится следующая информация:

- Любые окна, кроме карт (списки, схемы, окна районирования, трехмерные карты, карты-призмы);
- Единицы расстояния, площади, координаты XY и единицы армейской системы координат;
- Настройки режима совмещения, автопрокрутки и плавной прокрутки;
- Настройки принтера;
- Любые таблицы, созданные на основе запросов;

**Внимание:** Составное выражение, содержащее любой оператор или несколько опрашиваемых таблиц.

- Не поддерживаются тематические карты, созданные на основе расчётных данных;
- Запросы с операторами “подзапроса”;
- Слои, основанные на запросах, содержащих операторы “подзапроса”;

**Внимание:** Оператор “подзапроса” - это любой оператор **Select**, вложенные внутрь другого оператора **Select**.

- Запросы без соответствующих им окон карт;
- Настройки экспорта;
- Направления линий;
- Узлы;
- Геолинки.

**См. также:**

**Оператор Save Workspace**

---

## Оператор Save Window

### Назначение

Сохраняет снимок окна в файл; соответствует команде **Файл > Экспорт окна**.

### Синтаксис

```
Save Window window_id As filespec Type filetype [ Width image_width [
Units paper_units ] ] [ Height image_height [ Units paper_units ] ] [
Resolution output_dpi ][ Copyright notice [ Font... ] ], где
```

*window\_id* – целое число идентификатора окна карты, отчета, графика, легенды, информации или линейки; определить идентификатор окна поможет например, [Функция FrontWindow\( \)](#) или [Функция WindowID\( \)](#).

*filespec* – строка с именем создаваемого файла.

*filetype* – строка, определяющая формат файла:

- “BMP” определяет формат Bitmap;
- “WMF” определяет формат Windows Metafile;
- “JPEG” определяет формат JPEG;
- “JP2” определяет формат JPEG 2000;
- “PNG” определяет формат Portable Network Graphics;
- “TIFF” определяет формат TIFF;
- “TIFFCMYK” определяет формат TIFF CMYK;
- “TIFFG4” определяет формат TIFFG4;
- “TIFFLZW” определяет формат TIFFLZW;
- “GIF” определяет формат GIF;
- “PSD” определяет формат Photoshop 3.0;
- “EMF” определяет формат Windows Enhanced Metafile.

*image\_width* – число, определяющее нужную ширину снимка.

*image\_height* – число, определяющее нужную высоту снимка.

*paper\_units* – строка, представляющая "бумажные" единицы измерения (например, “cm” для сантиметров).

*output\_dpi* – число, определяющее нужное разрешение снимка в точках на дюйм (DPI).

*notice* – строка с сообщением об авторских правах; это сообщение появится в нижней части снимка.

Параметром этого оператора **Font** можно задать стиль оформления текста.

### Описание

Оператор **Save Window** сохраняет изображение окна в файл. Действие оператора аналогично действию команды **Файл > Экспорт окна**, с тем исключением, что оператор **Save Window** не выводит диалог на экран. Размер изображения, полученного из окна Карты, Отчета или Графика, по умолчанию будет равен размеру самого окна. Размер изображения, полученного из окна Легенды, Информации, Статистики или Линейки, по умолчанию будет устанавливаться таким, чтобы показать в окне все данные. Вы можете определить свои размеры для экспортируемых изображений в предложениях **Width** и **Height**. **Resolution** позволяет задать разрешение изображения в dpi при экспорте изображения в растровый формат. Предложение [Предложение Font](#) определяет стиль текста об авторских правах.



Чтобы включить текст авторских прав в нижнюю часть изображения, используйте дополнительное предложение **Copyright**. Смотрите пример ниже. Чтобы стереть текст авторских прав, задайте предложение **Copyright** с пустой строкой (").

В случае нехватки места на диске при экспорте окна, может быть зафиксирована ошибка под номером 408. Имейте это в виду, если Вы пытаетесь создать слишком большое изображение.

### Примеры

В этом примере создается метафайл Windows:

```
Save Window i_mapper_ID As "riskmap.wmf" Type "WMF"
```

В этом примере показано, как оформить сообщение об авторских правах. Функция **Chr\$( )** используется, для того чтобы вставить значок копирайта.

```
Save Window i_mapper_ID As "riskmap.bmp"
  Type "BMF"
  Copyright "Copyright " + Chr$(169) + " 1996, MapInfo Corp."
```

См. также:

**Оператор Export**

---

## Оператор Save Workspace

### Назначение

Создает файл Рабочего Набора с текущим состоянием рабочего окна MapInfo.

### Синтаксис

**Save Workspace As** *filespec*, где

*filespec* – строка с именем файла создаваемого рабочего набора.

### Описание

Оператор **Save Workspace** создает Рабочий Набор по текущему состоянию программы MapInfo Professional. Действие оператора аналогично действию команды **Файл > Экспорт окна**, с тем исключением, что оператор **Save Workspace** не выводит диалог на экран.

Для загрузки существующего Рабочего Набора используйте оператор **Оператор Run Application**.

### Пример:

```
Save Workspace As "market.wor"
```

См. также:

**Оператор Run Application**

## Функция SearchInfo( )

### Назначение

Возвращает информацию о результатах поиска, сделанного функцией SearchPoint( ) или SearchRect( ).

### Синтаксис

**SearchInfo**( *sequence\_number*, *attribute* ), где

*sequence\_number* - целое число от 1 до количества найденных объектов;

*attribute* – короткое целое число, код результата функции.

### Возвращаемая величина

Строка или целое число. Тип зависит от значения параметра *attribute*.

### Описание

После вызова функции SearchRect( ) или, осуществляющих поиск объектов на Карте, функция SearchInfo( ) обрабатывает результаты поиска.

Параметр *sequence\_number* должен принимать значения от 1 и более. Максимальное значение для этого параметра равняется результату функции SearchPoint( ) или SearchRect( ).

Параметр *attribute* должен принимать значение одного из кодов, имена для которых заданы в файле стандартных определений MAPBASIC.DEF:

| Значения <i>attribute</i> | SearchInfo( ) возвращает:   |
|---------------------------|---|
| SEARCH_INFO_TABLE         | Строка, величина типа String: имя таблицы, содержащей этот объект. Если объект принадлежит Косметическому слою, то строка будет равна "CosmeticN" (где N – число от 1 и более).         |
| SEARCH_INFO_ROW           | Целое число, величина типа Integer: номер записи. Этот номер Вы можете использовать в операторе <b>Оператор Fetch</b> или в предложении <b>Where</b> оператора <b>Оператор Select</b> . |

Результаты поиска хранятся в памяти до тех пор, пока выполняется программа или до следующего поиска. Результаты поиска не будут удалены из памяти, если пользователь закроет окно или таблицу, в которых осуществлялся поиск; поэтому не надо затягивать с обработкой результатов поиска. Чтобы форсировать удаление результатов поиска из памяти, проведите поиск, который обязательно ничего не найдет (например, поиск по координатам 0, 0).

MapInfo поддерживает отдельные наборы значений для результатов поиска для каждой действующей MapBasic-программы, а также набор для результатов поиска самой программы MapInfo (для команд, введенных из окна MapBasic).

**Ошибки:**

В результате выполнения функции может генерироваться код ошибки ERR\_FCN\_ARG\_RANGE, если значение параметра *sequence\_number* больше числа найденных объектов.

**Пример:**

Следующая программа создает кнопки для двух инструментов. Если пользователь пользуется точечным инструментом, то программа вызывает функцию **Функция SearchPoint( )**; если пользователь рисует рамки, то программа вызывает функцию **Функция SearchRect( )**. В каждом случае программа использует функцию **SearchInfo( )** для определения того, какие объекты попались пользователю.

```
Include "mapbasic.def" Include "icons.def" Declare Sub Main Declare Sub
tool_sub Sub Main Create ButtonPad "Поиск" AsToolButton Calling tool_sub
ID 1 Icon MI_ICON_ARROWCursor MI_CURSOR_ARROWDrawMode DM_CUSTOM_POINT
HelpMsg "Укажите мышкой на Карту\nПоиск в точке"SeparatorToolButton
Calling tool_sub ID 2 Icon MI_ICON_SEARCH_RECTCursor
MI_CURSOR_FINGER_LEFTDrawMode DM_CUSTOM_RECTHelpMsg "Нарисуйте
прямоугольник на Карте\nПоиск в рамке"Width 3 Print "Работает программа
поиска." Print "Выберите инструмент из панели Поиск"и укажите на Карту."
End Sub
Sub tool_sub' Эта процедура вызывается, если пользователь действует'одним
из инструментов из панели Поиск.
  Dim x, y, x2, y2 As Float,i, i_found, i_row_id, i_win_id As Integer,
  s_table As Alias i_win_id = FrontWindow( )If WindowInfo(i_win_id,
WIN_INFO_TYPE) <> WIN_MAPPER ThenNote "Этот инструмент работает только в
окне Карты."Exit SubEnd If' Определяем начальную точку действия
инструмента.
  x = CommandInfo(CMD_INFO_X)y = CommandInfo(CMD_INFO_Y)If
CommandInfo(CMD_INFO_TOOLBTN) = 1 Then' случай, когда действует точечный
инструмент.
    ' определяем, сколько объектов содержат данную точку.
    i_found = SearchPoint(i_win_id, x, y)Else' случай, когда действует
инструмент, рисующий рамку.
    ' определяем, сколько объектов содержатся в рамке.
    x2 = CommandInfo(CMD_INFO_X2)y2 = CommandInfo(CMD_INFO_Y2)i_found =
SearchRect(i_win_id, x, y, x2, y2)End IfIf i_found = 0 ThenBeep '
Объектов в этом месте нет.
    Else Print Chr$(12)If CommandInfo(CMD_INFO_TOOLBTN) = 2 ThenPrint
"Прямоугольник: x1= " + x + ", y1= " + yPrint "x2= " + x2 + ", y2= " + y2
Else Print "Point: x=" + x + ", y= " + yEnd If' Обработка результатов
поиска.
    For i = 1 to i_found' Определяем имя таблицы, содержащей найденный
объект.
      s_table = SearchInfo(i, SEARCH_INFO_TABLE)' Определяем номер записи,
содержащей найденный объект.
      i_row_id = SearchInfo(i, SEARCH_INFO_ROW)If Left$(s_table, 8) =
"Cosmetic" Then Print "Объект на Косметическом слое" Else ' извлекаем
строку таблицы, содержащую объект.
        Fetch rec i_row_id From s_table
        s_table = s_table + ".coll"
        Print s_table
      End If
    Next
  End If
End Sub
```

См. также:

Функция [SearchPoint\( \)](#), Функция [SearchRect\( \)](#)

---

## Функция [SearchPoint\( \)](#)

### Назначение

Ищет объекты в заданной точке Карты.

### Синтаксис

`SearchPoint( map_window_id, x, y )`, где

*map\_window\_id* - идентификатор окна Карты;

*x* – координата по оси X (например, долгота);

*y* – координата по оси Y (например, широта).

### Возвращаемая величина

Целое число найденных объектов.

### Описание

Функция [SearchPoint\( \)](#) осуществляет поиск объектов в заданной точке Карты. Поиск проводится по всем доступным слоям окна Карты, включая Косметический слой, если для него установлен режим доступности. Полученное значение указывает количество найденных объектов.

Функция не выбирает найденные объекты и не отменяет текущего выбора. Функция помещает список выбранных объектов в оперативную память. Для чтения этого списка используется функция [Функция SearchInfo\( \)](#) вызываемая после выполнения функции [SearchPoint\( \)](#).

Поиск имеет небольшой допуск, аналогичный допуску при действии инструмента Информация. Точки и линейные объекты, расположенные слишком близко к месту указания мышкой, считаются найденными, даже если координаты нажатия кнопки мышки не совпадают с точкой или ложатся непосредственно на линию.

Для того, чтобы пользователь мог осуществить выбор точки на карте при помощи мышки, можно использовать оператор [Оператор Create ButtonPad](#) или [Оператор Alter ButtonPad](#) для создания нового инструмента. Используйте код `DM_CUSTOM_POINT` как код рисования “точечным” инструментом. В обработке инструмента используйте вызов функции [CommandInfo\( \)](#) [функция](#) для определения координат точки, которую пользователь задал новым инструментом.

### Пример:

Смотрите пример в [Функция SearchInfo\( \)](#) на стр. 24.

См. также:

[Функция SearchInfo\( \)](#), [Функция SearchRect\( \)](#)

## Функция SearchRect( )

### Назначение

Ищет объекты на Карте в заданном прямоугольнике (рамке).

### Синтаксис

**SearchRect**( *map\_window\_id*, *x1*, *y1*, *x2*, *y2* ), где

*map\_window\_id* - идентификатор окна Карты;

*x1*, *y1* – координаты, задающие один из углов прямоугольника;

*x2*, *y2* – координаты, задающие противоположный по диагонали угол прямоугольника.

### Возвращаемая величина

Целое число найденных объектов.

### Описание

Функция **SearchRect( )** осуществляет поиск объектов в заданном прямоугольнике в окне Карты и возвращает количество найденных объектов. Поиск проводится по всем доступным слоям окна Карты, включая Косметический слой, если для него установлен режим доступности. Полученное значение указывает количество найденных объектов.

**Внимание:** Функция не выбирает найденные объекты и не отменяет текущего выбора. Функция помещает список выбранных объектов в оперативную память. Для чтения этого списка используется функция **Функция SearchInfo( )** вызываемая после выполнения функции **SearchRect( )**.

Механизм поиска работает аналогично механизму инструмента MapInfo Выбор-в-Рамке: если центр оид объекта попадает в прямоугольник, то объект включается в список найденных.

Для того, чтобы пользователь мог задавать прямоугольник на Карте при помощи мышки, можно использовать оператор **Оператор Create ButtonPad** или **Оператор Alter ButtonPad** для создания нового инструмента. Используйте код DM\_CUSTOM\_RECT как код рисования “рамочным” инструментом. В обработчике инструмента используйте вызов функции **CommandInfo( )** **функция** для определения координат прямоугольника, который пользователь задал новым инструментом.

### Пример:

Смотрите пример в **Функция SearchInfo( )** на стр. 24.

### См. также:

**Функция SearchInfo( )**, **Функция SearchPoint( )**

---

## Функция Second

### Назначение

Возвращает секунды и доли секунд в виде вещественного числа.

### Синтаксис

`Second (Time)`

### Возвращаемая величина

Номер

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim X as time
dim iSec as integer
X = CurDateTime()
Sec = Second(X)
Print iSec
```

---

## Функция Seek( )

### Назначение

Возвращает текущую позицию в файле для операций ввода/вывода.

### Синтаксис

`Seek( filenum ), где`

*filenum* – номер файла, который был присвоен ему при открытии;

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция **Seek( )** возвращает текущую позицию в открытом файле для операций ввода/вывода.

Значение параметра *filenum* должно быть номером файла, под которым он был открыт оператором **Оператор Open File**.

Результатом функции **Seek( )** будет целое число. Если файл был открыт в режиме прямого доступа, **Seek( )** вернет номер записи (запись, которая будет прочитана или записана). Если файл открыт в двоичном режиме, функция **Seek( )** вернет номер байта, который будет прочитан или записан следующей операцией ввода/вывода.

### Ошибки:

В результате выполнения функции может генерироваться код ошибки `ERR_FILEMGR_NOTOPEN`, если файл не был открыт.

### См. также:

[Оператор Get](#), [Оператор Open File](#), [Оператор Put](#), [Оператор Seek](#)

---

## Оператор Seek

### Назначение

Назначает в определенном файле текущую позицию для дальнейших операций ввода/вывода.

### Синтаксис

**Seek** [ # ] *filenum*, *position*, где

*filenum* – номер открытого файла, целое число;

*position* – целое число, определяющее новую текущую позицию.

### Описание

Оператор **Seek** устанавливает доступ к содержимому открытого файла, начиная с позиции *position*. Файл должен быть открыт оператором **Open File** под номером *filenum*.

Если файл был открыт в режиме прямого доступа (**RANDOM**), то значение *position* задает номер записи.

Если файл был открыт в режиме последовательного доступа, то оператор **Seek** устанавливает доступ к содержимому файла, начиная с байта, номер которого задается параметром *position*.

### См. также:

[Оператор Get](#), [Оператор Input #](#), [Оператор Open File](#), [Оператор Print #](#), [Оператор Put](#), [Функция Seek\( \)](#), [Оператор Write #](#)

---

## Процедура SelChangedHandler

### Назначение

Процедура, которая автоматически вызывается при изменении выбора строк в таблицах.

### Синтаксис

**Declare Sub SelChangedHandler Sub SelChangedHandler** *statement\_list* **End Sub**, где

*statement\_list* – список операторов процедуры-обработчика.



## Описание

**SelChangedHandler** – зарезервированное имя для процедуры MapBasic. Когда пользователь запускает программу, имеющую процедуру **SelChangedHandler**, программа не завершается после выполнения всех операторов процедуры Main и других процедур, вызванных из нее. Программа будет в режиме ожидания до тех пор, пока **Оператор End Program** не будет выполнен оператор. Пока приложение остается в памяти, MapInfo Professional автоматически вызывает **SelChangedHandler** в случае изменения в выборе строк таблиц.

Для получения информации об изменениях используйте в теле процедуры **SelChangedHandler** функцию **CommandInfo( )** функция со следующими кодами:

| Значения attribute | Результат CommandInfo( attribute ):  |
|--------------------|--|
| CMD_INFO_SELTYPE   | 1, если строка была добавлена в выборку; 2, если строка была исключена из предыдущей выборки; 3, если все строки в таблице выбраны; 4, если все строки таблицы были исключены из предыдущей выборки. |
| CMD_INFO_ROWID     | Целое число (Integer) – номер строки, которая была добавлена в выбор или исключена из выбора (работает только, если одна строка была выбрана или исключена из выбора).                               |
| CMD_INFO_INTERRUPT | Логическая величина (Logical): TRUE, если пользователь прервал выбор клавишей ESC, FALSE – иначе.  |

Когда процедура закончит свои действия (**Оператор End Program**), прикладная программа вновь переходит в режим ожидания. И так всякий раз при новом применении инструмента. Для завершения процедуры **SelChangedHandler** можно использовать **Оператор End Program**. Поэтому, если Вам снова понадобятся функции в процедуры **SelChangedHandler**, не используйте **Оператор End Program**.

Одновременно в состоянии ожидания могут находиться несколько прикладных программ. Поэтому при изменении выбора автоматически выполняются все процедуры **SelChangedHandler** из этих программ, одна за другой.

Процедура **SelChangedHandler** не может активизировать другие окна. Другими словами, в процедуре **SelChangedHandler** не могут быть выполнены такие операторы, как **Оператор Note**, **Оператор Print**, или **Оператор Dialog**.

**См. также:**

**CommandInfo( )** функция, **Функция SelectionInfo( )**

## Оператор Select

### Назначение

Выбирает отдельные строки и колонки из одной или более таблиц и составляет из них временные таблицы запросов. Этот оператор часто используется для сортировки и вычисления промежуточных сумм.

### Синтаксис

```
Select expression_list From table_name [ , ... ] [ Where expression_group ]  
[ Into results_table [ Noselect ] ] [ Group By column_list ] [ Order By  
column_list ], где
```

*expression\_list* – список выражений через запятую, задающих содержимое колонок временной таблицы "Selection";

*expression\_group* – одно выражение или список выражений, разделенных словами AND или OR;

*table\_name* – имя открытой таблицы;

*results\_table* – имя таблицы, в которую будут помещены результаты выбора;

*column\_list* – имя колонки или список имен, разделенных запятыми.

### Описание

Оператор **Select** предоставляет программисту MapBasic возможности диалога команды **Запрос > SQL-запрос**.

Оператор **Select** построен по образцу одноименного оператора в языке запросов SQL (Structured Query Language). В случае, если вы пользуетесь SQL-ориентированным программным обеспечением для работы с базами данными, оператор **Select** скорее всего вам знаком. Вариант оператора **Select**, применяемый в MapBasic, позволяет использовать уникальные географические возможности MapInfo, которые не имеют многие базы данных, использующие язык SQL.

Имена колонок, задаваемые выражениями типа *имятаблицы.имяколонки* в операторе **Select** могут использовать имена только тех таблиц, которые перечислены в предложении **From**. Например, оператор **Select** выбирает значения в колонке STATES.OBJ, если таблица STATES включена в список предложения **From** в операторе **Select**.

Оператор **Select** может выполнять множество различных задач. Один оператор **Select** может перестроить таблицу и отфильтровать (т.е. выбрать по критерию) записи. Также оператор **Select** может вычислить промежуточные и итоговые суммы для всей таблицы. Оператор **Select** может отсортировать записи в таблице, создать и заполнить новую колонку из других и скомбинировать колонки из разных таблиц.

Общий смысл оператора **Select** состоит в том, что из одной или нескольких таблиц выбираются строки, которые помещаются во временную таблицу под именем "Selection" или под именем, заданным параметром *results\_table*. Эта временная таблица в дальнейшем интерпретируется MapInfo Professional, как и любая другая.

После того, как оператор **Select** образует выборку, обращение к функции **Функция SelectionInfo( )** дает информацию о составе выборки.

Синтаксис оператора **Select** допускает несколько предложений, большинство которых не обязательны. Природа и функция оператора **Select** зависит от того, какие предложения входят в состав оператора. Например, если Вы хотите отфильтровать записи по критерию, то нужно включить предложение **Where**; если нужно просуммировать значения, то включите в состав оператора предложение **Group By**; для сортировки включите предложение **Order By**. Употребление каждого из этих предложений не исключает другого. Оператор **Select** может состоять из всех возможных предложений сразу.

### Предложение Select

Это предложение задает колонки, которые будут включены в результирующую таблицу. Список колонок располагается сразу за первым словом **Select** в операторе и является обязательным. Самое простое значение для параметра *expression\_list* – звездочка ("\*"). Это значит, что в результирующую таблицу будут включены все колонки. Например, оператор

```
Select * From world
```

говорит MapBasic включить все колонки во временную таблицу "Selection". Другим возможным значением параметра *expression\_list* является список выражений через запятую. Каждое из выражений представляет колонку для результирующей таблицы. Обычно такое выражение включает в себя имя одной колонки или даже имена нескольких колонок таблицы, из которой производится выбор. При составлении списка колонок можно использовать функции и/или операторы MapBasic.

Например, в следующем операторе **Select** список выражений *expression\_list* состоит из двух выражений:

```
Select country, Round(population/1000000)"миллионов" From World
```

Первое – имя колонки "Страна", второе – вызов функции округления (**Round( )**) значений из колонки "Население" до целых миллионов.

После выполнения этого оператора первая колонка в результирующей таблице будет содержать величины из колонки "Страна" с названиями стран в таблице WORLD, а вторая – величины из колонки "Население" с численностью населения, округленной до миллионов.

Выражение из списка *expression\_list* может также назначить синоним, которым будет именоваться соответствующая колонка, если, например, показывать эту результирующую таблицу в окне Списка. Следующий оператор назначает для второй колонки синоним "Миллионы":

```
Select country, Round(population/1000000)"миллионов" From World
```

Таблицы, которые могут иметь присоединенные графические объекты, имеют специальную колонку, которая имеет имя "object" (или "obj"). Если в список колонок включить слово "obj", то в результирующую таблицу будет добавлена строка, содержащая тип графического объекта, присоединенного к данной записи или ничего, если такого объекта нет.

Значение параметра *expression\_list* может быть или списком выражений, или звездочкой, но звездочка не может употребляться в списке выражений. Пример следующего оператора НЕ РАБОТАЕТ:

Select \*, object From world ' неправильно!

### Предложение From

Обязательное предложение, в котором должно указываться имя открытой таблицы, из которой производится выбор. Если вы создаете таблицу для совместного доступа, выбранные таблицы должны быть базовыми, а не результаты предыдущего запроса.

### Предложение Where

Одна из функций предложения **Where**, основная, заключается в задании критерия выбора строк в таблице `table_name`. Здесь могут использоваться любые выражения (смотрите раздел "Выражения" ниже). Но обратите внимание, что два или более выражения разделяются словами **And** или **Or**, а не запятыми. Например, предложение **Where**, состоящее из двух выражений должно выглядеть следующим образом:

```
Where Доход > 15000 And Доход < 25000
```

Оператор **And** играет роль логического "И", то есть оба условия должны выполняться, а оператор **Or** играет роль логического "ИЛИ", то есть MapBasic выберет запись, если она удовлетворяет любому из условий.

В соответствии с записью специальной колонки, предложение **Where** может контролировать географическую информацию, содержащуюся в каждой строке таблицы. Вы можете также применять выражение **Not**, отрицание.

Например, следующий оператор **Where** выбирает строки, лишённые присоединенного графического объекта.

```
Where Not Object
```

Если в операторе **Select** используются две или более таблицы, то предложение **Where** должно присутствовать обязательно, и, более того, в этом предложении должны быть заданы условия объединения. Обычно такой объединяющий оператор выглядит примерно так: **Where таблица1.поле = таблица2.поле**, где две колонки в разных таблицах сопоставляются друг другу. В следующем примере показано, как можно объединить таблицы RUSSIA и CITY200, если в колонке "Аббр" таблицы CITY200 и в колонке "Аббр" таблицы STATES содержатся аббревиатуры областей России:

```
Where RUSSIA.Аббр = CITY200.Аббр
```

Объединение Вы можете произвести, используя также географический оператор:

```
Where RUSSIA.obj Contains CITY200.obj
```

Предложение **Where** можно также применять для более изощренного выбора, для чего используются специальные операторы **Any** и **All**. Оператор **Any**, применяемый к некоему множеству значений, позволяет проверить, выполняется ли выражение в предложении **Where** для какого-либо из этих значений. Оператор **All**, наоборот, требует, чтобы выражение в предложении **Where** выполнялось для всех значений, объединяемых этим оператором.

Следующий запрос выбирает любую запись о клиентах, в колонке "Аббр" которой содержатся значения "МОС", "СПБ" или "ЯРС". Оператор **Any( )** работает так же, как и оператор **IN** в обычном языке SQL.

```
Select * From customers      Where Аббр = Any ("МОС", "СПБ", "ЯРС")
```

В предложении **Where** может быть включен самостоятельный оператор **Select**, называемый "подзапросом". В следующем примере используются две таблицы: PRODUCTS, содержащая записи о продаваемых товарах, и ORDERS, с данными о заказах на товары. Оплаченные товары могут в данный момент не находиться на складе. Задача состоит в том, чтобы выяснить номенклатуру заказанных товаров, находящихся на складе. Другими словами, нужно "выбрать все заказы, которые не находятся в списке распроданных товаров".

```
Select * From orders
      Where partnum <>
      All(Select partnum from products
          where not instock)
```

Во второй строке запроса слово **Select** появляется второй раз, уже как подзапрос. Этот подзапрос выбирает все товары, которых в данный момент НЕТ на складе. Главный запрос **Where** выполняет поставленную задачу с помощью оператора All( ) и результатов подзапроса. (Слово "instock" означает "на складе"; это логическая переменная).

В примере, приведенном выше, подзапрос создает набор значений, а предложение **Where** главного запроса проверяет условие запроса для них. В других случаях подзапрос может использовать функции обобщения, вычисляющие одно значение.

В следующем примере используется функция Avg( ) для вычисления среднего значения в поле "Население" таблицы RUSSIA.

В результате выполнения оператора **Select** выбираются записи о более-чем-среднем населении.

```
Select * From RUSSIAWhere население > (Select Avg(население) From RUSSIA)
```

MapInfo Professional также поддерживает ключевое слово SQL In. В операторе **Select** слово In может заменять "= Any". Другими словами, предложение:

```
Where state = Any ("МОС", "СПБ", "ЯРС")
```

эквивалентно предложению **Where** с ключевым словом **In**:

```
Where state In ("МОС", "СПБ", "ЯРС")
```

а фраза **Not In** в запросе эквивалентна фразе: <> All.

**Внимание:** Простой оператор **Select** может не содержать множественных невложенных подзапросов. Оператор **Select** не поддерживает в MapBasic "синхронные" или "коррелирующие" подзапросы. "Синхронные" подзапросы ссылаются на внешние таблицы. Внутренние запросы перезапрашиваются для каждой строки внешней таблицы. Таким образом достигается корреляция. Пример:

```
' Следующий запрос содержит в своем тексте ссылку' на внешнюю таблицу. Он
НЕ сработает в MapBasicSelect * from ДРУЗЬЯWhere друг.имя = (Select
люди.имя From людиWhere друг.имя = клиент.имя)
```

Последнее замечание адресовано профессионалам, привыкшим составлять сложные SQL-запросы к другим базам данных.

### Предложение Into

Задаёт имя для результирующей таблицы. Если предложения **Into** нет в операторе, то таблица с результатами выбора будет названа именем "Selection". Если последующие запросы обращаются к таблице "Selection", то результаты запросов будут называться именами ЗАПРОС*N* (например, ЗАПРОС1).

Если в предложении используется ключевое слово **Noselect**, то оператор формирует запрос не меняя предыдущую таблицу "Selection". То есть те записи, которые уже были выбраны, после оператора с ключом **Noselect** останутся выбранными.

Если в операторе использовалось ключевое слово **Noselect**, то операция не повлечет за собой запуска процедуры-обработчика **Процедура SelChangedHandler**.

Предложение Group By

Предложение Group By определяет порядок группировки строк при выполнении обобщающих действий (вычислений промежуточных сумм и других значений). В предложении **Group By** обычно задается имя колонки (или несколько имен колонок), на основании значений из которых группируются промежуточные результаты. Например, если нужно обобщить какие-либо данные для областей России, то в предложении **Group By** должна быть задана колонка, содержащая имена областей России. Предложение **Group By** может не ссылаться на функцию, возвращающую переменную, такую как **Функция ObjectInfo( )**.

Функции обобщения **Sum( )**, **Min( )**, **Max( )**, **Count(\*)**, **Avg( )**, и **WtAvg( )**

**Внимание:** обычно не используются в предложении **Group By**. Они помещаются в список *expression\_list* оператора **Select**, а предложение **Group By** только задает группирующую колонку.

В следующем примере таблица Q4SALES содержит информацию о продажах за четвертый квартал. Каждая запись содержит информацию о сумме каждой сделки в колонке "amount". В колонке "territory" задана территория, на которой произошла сделка. В запросе вычисляется, сколько сделок произошло на каждой территории и суммарный объем этих сделок.

```
Select territory, Count(*), Sum(amount)From q4salesGroup By territory
```

Предложение **Group By** заставляет MapBasic сначала сгруппировать записи по "территориям", а затем сосчитать сумму сделок на каждой "территории". Запрос **Select** создает три колонки: имя территории, количество записей в таблице Q4SALES, относящихся к данной территории, и сумму сделок на этой территории.

**Внимание:** Функция **Sum( )** принимает в качестве параметра имя колонки. Функция **Count( )** с параметром-звездочкой просто пересчитывает количество записей в каждой из групп. Функция **Count( )**, в отличие от других функций обобщения, не нуждается в аргументах – именах колонок.

В следующей таблице приведено описание функций обобщения данных:

| Функция              | Описание   |
|----------------------|--|
| <b>Avg( column )</b> | Возвращает среднее значение из значений в колонке column.                            |
| <b>Count(*)</b>      | Возвращает количество записей в группе. Укажите * (звездочку) вместо имени колонки.  |
| <b>Max( column )</b> | Возвращает наибольшее значение из значений в колонке column для всех записей группы. |
| <b>Min( column )</b> | Возвращает наименьшее значение из значений в колонке column для всех записей группы. |

| Функция                                | Описание  |
|--|---|
| <b>Sum( column )</b>                   | Возвращает сумму значений в колонке column для всех записей группы.   |
| <b>WtAvg( column , weight_column )</b> | Возвращает среднее взвешенное значение из значений в колонке column для всех записей группы. Смотрите ниже. |

### Вычисление взвешенной средней величины

Функция вычисления взвешенной средней величины **WtAvg( )** использует значения из дополнительной колонки в качестве коэффициентов значений из основной колонки. Например, следующий оператор использует функцию **Wtavg( )** для вычисления уровня грамотности на каждом континенте:

```
Select continent, Sum(нас_1994), WtAvg(грамотность, нас_1994)From World
Group By Континент Into Lit_query
```

Предложение **Group By** задает группировку строк таблицы. MapInfo группирует записи по значениям из колонки “Континент”. Все строки, имеющие значение “Северная Америка” в колонке “Континент” будут рассматриваться как одна группа; Все записи о континенте “Азия” попадут в другую группу и т.д. Для каждой группы записей, т.е. для каждого континента, MapInfo вычисляет взвешенное среднее показателей грамотности.

Простое среднее (функция **Avg( )**) вычисляется как сумма, деленная на количество слагаемых. Взвешенное среднее (функция **WtAvg( )**) более сложна; при вычислении взвешенного среднего одни значения могут иметь больший вес и больше влиять на результат. В нашем примере среднее значение грамотности вычисляется с учетом населения (колонка “Нас\_1994”); другими словами, страны с большим населением вносят больший вклад в результат.

### Выражения, задающие колонки, в предложении Group By

В предложении **Group By** колонки могут задаваться именем, например, как в приведенном выше примере группировки по территориям. Альтернативным способом задания колонки является использование ее порядкового номера в таблице, используя выражение вида col#. В таком выражении вместо знака # должно стоять целое число (единица и более), указывающее одну из колонок в предложении **Select**. Группирующее предложение в операторе **Select** из предыдущего примера может выглядеть как Group By col1, так и Group By 1, вместо **Group By** territory и не может содержать вызов функции, возвращающей переменное значение.

Альтернативный синтаксис в обозначениях колонок бывает необходим в тех случаях, когда нужно обозначить колонку, имя которой является результатом вычислений. В следующем примере функция **Month( )** используется для вычисления группирующих значений, но название колонки из выражения получить нельзя. Поэтому нужно применить альтернативный синтаксис:

```
Select Month(день_болезни), Count(*)From болезниGroup By 1
```



В этом примере каждая запись таблицы БОЛЕЗНИ должна содержать дату болезни. В результате запроса образуется таблица из 12 строк (по количеству месяцев); во второй колонке будет содержаться количество дней, пропущенных в этом месяце по болезни.

### Группировка по значениям из нескольких колонок

В некоторых случаях Вам надо будет задать более одного имени колонки в предложении **Group By**. Это нужно, когда информации в одной колонке недостаточно для точного задания условия группировки. Пусть, например, Вы располагаете картой районного деления России; названия некоторых районов, принадлежащих разным областям, совпадают. Например, Первомайский район есть в нескольких областях. Поэтому, определив в предложении **Group By** только название района, Вы получите в результате запроса недостоверные данные – обобщенные по всем районам с именем "Первомайский", независимо от того, в какой области они находятся.

Чтобы решить эту проблему (или аналогичную), задайте колонку имен областей, как дополнительную в предложении **Group By**. Например, предложение group by может выглядеть следующим образом:

```
Group By район, область
```

В этом случае для каждой уникальной пары *район, область* будет произведена отдельная группировка. Суммарные значения для пары "Первомайский-Нижегородская" будут различаться со значениями для пары "Первомайский-Дагестан".

### Предложение Order By

Предложение Order By задает колонку или колонки, по значениям которых должна происходить сортировка в результирующей таблице. Также как и в предложении **Group By**, колонки задаются списком имен полей или списком номеров этих полей. Элементы списка разделяются запятыми.

Стандартный порядок сортировки – по возрастанию, то есть от "А" до "Z", от "А" до "Я" и от 0 до 9. Сортировка по убыванию задается ключевым словом **Desc** как показано в следующем примере:

```
Select * From Города Order Область, население Desc
```

Этот запрос выполняет двухуровневую сортировку таблицы ГОРОДА. Сначала MapBasic сортирует города в областях по возрастанию имен областей, после чего MapBasic внутри каждой группы, представляющей одну область, сортирует по убыванию населения городов. Слово **Desc** отделяется пробелом, а не запятой.

Предложение **Order By** не может содержать вызов функции, возвращающей переменное значение. Например, так как функция **Функция ObjectInfo()** возвращает переменное значение, она не может быть использована для сортировки в операторе Select..

### Географические операторы

MapBasic поддерживает несколько географических операторов: Contains, Contains Part, Contains Entire, Within, Partly Within, Entirely Within, и Intersects. Они используются в любых выражениях и особенно полезны в предложении **Where** для задания критерия выбора на

основании взаимного расположения объектов на Карте. Все графические операторы работают только со значениями объектного типа и в результате дают логическую величину. Операторы перечислены в следующей таблице.

| Применение                      | Возвращает TRUE, если:                                    |
|---------------------------------|---|
| objectA Contains objectB        | центроид второго объекта находится внутри первого объекта |
| objectA Contains Part objectB   | первый объект содержит часть второго объекта              |
| objectA Contains Entire objectB | первый объект полностью содержит второй объект            |
| objectA Within objectB          | центроид первого объекта внутри второго объекта           |
| objectA Partly Within objectB   | часть первого объекта помещается внутри второго           |
| objectA Entirely Within objectB | первый объект полностью помещается внутри второго         |
| objectA Intersects objectB      | Два объекта пересекаются хотя бы в одной точке            |

### Ускоренный выбор

Некоторые варианты оператора **Select** выполняют операцию выбора быстрее, чем другие. Скорость выбора зависит от содержания предложения **Where**, задающего критерий выбора.

Если после слова **Where** стоит одно выражение в форме:

```
columnname = constant_expression
```

(где columnname – имя колонки, а constant\_expression – выражение из постоянных строковых величин) или выражений такой формы несколько и они разделены операционными словами And, то такой оператор **Select** будет выполняться быстрее, так как в этом случае максимально используются преимущества индексации. Если подобные выражения объединены оператором Or, то преимущества индексации не используются.

Также MapInfo сравнительно быстрее выполняет выбор с предложениями *Where* вида:

```
[ tablename. ] obj geographic_operator object_expression
```

(где tablename – имя таблицы, geographic\_operator – географический оператор, object\_expression – выражение из постоянных строковых величин) и вида:

```
RowID = constant_expression
```

**RowID** – это имя специальной колонки, содержащей номер записи в таблице и обычно скрытой от пользователя. Каждое значение RowID является номером строки соответствующей таблицы; другими словами, первая строка в таблице имеет RowID равным единице.

### Примеры

В этом примере выбираются все клиенты в Московской, Ленинградской и Ярославской областях. Каждая запись о клиенте не обязательно содержит имя области; запрос вместо этого опирается на географическое расположение клиента.

```
Select * From КлиентыWhere obj Within Any(Select obj From RUSSIAWhere
Аббр = "МОС" or Аббр = "СПБ" or Аббр = "ЯРС")
```

В этом примере демонстрируется действие подзапроса. Мы желаем выбрать все территории сбыта, в которых проживают клиенты, имеющие признак "Кадастр". Подзапрос сначала выбирает подмножество записей о клиентах с признаком "Кадастр", после чего основной запрос выбирает территории по значениям из подмножества.

```
Select * From ТерриторииWhere obj Contains Any (Select obj From клиенты
Where клиенты.тип = "Кадастр")
```

Следующий запрос выбирает все земельные участки, пересекающиеся с участком номер 120059.

```
Select * From УчасткиWhere obj Intersects (Select obj From УчасткиWhere
Ном_участка = 120059)
```

**См. также:**

**Оператор Open Table**

---

## Функция SelectionInfo( )

### Назначение

Возвращает информацию о текущем выборе в таблицах (информацию о временной таблице "Selection").

**Внимание:** Выбранные подписи не присоединяются к выбору, так как сами являются атрибутами других объектов и не являются отдельными объектами.

### Синтаксис

**SelectionInfo**( *attribute* ), где

*attribute* – короткое целое число.

### Возвращаемая величина

Целое число или строка в зависимости от значения attribute. Величина типа Integer или String.

### Описание

В файле стандартных определений MapBasic MAPBASIC.DEF определены имена для кодов, которые можно использовать в функции SelectionInfo.

| Параметры          | Результат, полученный SelectionInfo( )   |
|--------------------|--|
| SEL_INFO_TABLENAME | Строковая величина. Имя таблицы, на базе которой создана таблица "Selection". Если ничего не выбрано, будет возвращена ошибка. |
| SEL_INFO_SELNAME   | Строковая величина. Имя временной таблицы (например, "Запрос1"). Если ничего не выбрано, будет возвращена ошибка.              |
| SEL_INFO_NROWS     | Целое число. Количество выбранных строк в таблице. Если ничего не выбрано, будет возвращен 0 (ноль).                           |

**Внимание:** Если записи выбраны в таблице, которая является объединением двух или более таблиц, то функция **SelectionInfo**(SEL\_INFO\_NROWS) возвращает количество строк в базовой таблице, которое может быть меньшим, чем количество строк в таблице "Selection". Пример смотрите ниже.

### Ошибки:

ERR\_FCN\_ARG\_RANGE, если неправильно задано значение аргумента.

### Пример:

Оператор **Оператор Select** используется для объединения. После всего переменная *i* будет равна 40 (числу строк, выбранных в базовой таблице RUSSIA), а переменная *j* будет равна 125 (числу строк таблицы запроса).

```
Dim i, j As Integer Select * From RUSSIA, City200 Where RUSSIA.obj  
Contains City_125.obj Into РЕЗУЛЬТАТi = SelectionInfo(SEL_INFO_NROWS) j =  
TableInfo(РЕЗУЛЬТАТ, TAB_INFO_NROWS)
```

### См. также:

**Оператор Select, Функция TableInfo( )**

---

## Оператор Server Begin Transaction

### Назначение

Посылает уведомление на удаленный сервер о начале нового сеанса работы.

### Синтаксис

**Server** *ConnectionNumber* **Begin Transaction**, где

*ConnectionNumber* – целое значение, номер соединения с базой данных;

Описание

Оператор **Server Begin Transaction** используется для обозначения начала сеанса обработки транзакций. Результаты последующих операторов языка SQL Insert, Delete и Update (внести, удалить и обновить), выполняемых функцией **Функция Server\_Execute( )** , не сохраняются в базе данных до тех пор, пока не будет выполнена команда **Оператор Server Commit**. Команда **Оператор Server Rollback** используется для отмены изменений.

Пример:

```
Dim hdbc As Integerhdbc = Server_Connect("ODBC", "DLG=1")Server hdbc Begin Transaction' ... Другие операторы ...Server hdbc Commit
```

См. также:

**Оператор Server Commit, Оператор Server Rollback**

Оператор Server Bind Column

Назначение

Назначает локальную область хранения для удаленного сервера базы данных.

Синтаксис

**Server** *StatementNumber* **Bind Column** *n* **To** *Variable*, *StatusVariable*, где *StatementNumber* – целое значение, номер SQL-оператора.

*n* – номер столбца в результирующем наборе, связываемого с переменной.

*Variable* – MapBasic-переменная для хранения значения столбца после выборки.

*StatusVariable* – переменная состояния, в которую записывается код, указывающий статус значения: пустое, усеченное или целое положительное значение.

Описание

Команда **Server Bind Column** назначает переменную приложения для сохранения значения столбца в результирующем наборе, специфицированном удаленным SQL-оператором **Оператор Select**. Когда последующий оператор **Оператор Server Fetch** выбирает строку данных из базы, значение столбца *n* присваивается этой переменной оператора **Оператор Server Bind Column**. Статус результата сохраняется в переменной состояния, указанной параметром *StatusVariable*.

| Значение StatusVariable | Условие   |
|-------------------------|---|
| SRV_NULL_DATA           | Возвращается, если столбец не имеет данных в полученной строке (пустое значение). |

| Значение StatusVariable      | Условие  |
|------------------------------|--|
| SRV_TRUNCATED_DATA           | Возвращается, если столбец содержит больше данных, чем может быть сохранено в указанной MapBasic-переменной. |
| Целое положительное значение | Число байт, возвращенное сервером данных.  |

### Пример:

```
' Приложение для "печати" адресных этикеток' Предполагается, что
существует реляционная таблица ADDR с 6 столбцами...Dim hdbc, hstmt As
IntegerDim first_name, last_name, street, city, state, zip As String
Dim fn_stat, ln_stat, str_stat, ct_stat, st_stat, zip_stat As Integer
hdbc = Server_Connect("ODBC", "DLG=1")
hstmt = Server_Execute( hdbc, "select * from ADDR")
Server hstmt Bind Column 1 To first_name,fn_stat
Server hstmt Bind Column 2 To last_name, ln_stat
Server hstmt Bind Column 3 To street, str_stat
Server hstmt Bind Column 4 To city, ct_stat
Server hstmt Bind Column 5 To state, st_stat
Server hstmt Bind Column 6 To zip, zip_stat
Server hstmt Fetch NEXT
While Not Server_Eot(hstmt)
    Print first_name + " " + last_name
    Print street
    Print city + ", " + state + " " + zip
    Server hstmt Fetch NEXT
Wend
Server hstmt Close
Server hdbc Disconnect
```

### См. также:

[Функция Server\\_ColumnInfo\( \)](#)

## Оператор Server Close

### Назначение

Освобождает ресурсы, занятые удаленным SQL-оператором доступа к данным.

### Синтаксис

**Server** *StatementNumber* **Close**, где

*StatementNumber* – целое значение, номер SQL-оператора.

### Описание

Оператор **Server Close** используется для оповещения сервера о завершении обработки текущего удаленного SQL-оператора. Все ресурсы, ассоциированные с исполнением этого оператора, возвращаются в распоряжение системы. Не забывайте вызывать оператор **Оператор Server Close** сразу же после исполнения функции **Функция Server\_Execute( )** для любого не выполняющего запрос данных SQL-оператора, обработка которого закончена в Вашем приложении.

### Пример:

```
' Выбирает пятую запись и закрывает SQL-оператор Select hstmt =
Server_Execute(hdbc, "Select * from Massive_Database") Server hstmt Fetch
Rec 5 Server hstmt Close
```

### См. также:

**Функция Server\_Execute( )**

---

## Функция Server\_ColumnInfo( )

### Назначение

Возвращает информацию о столбцах результирующего набора.

### Синтаксис

**Server\_ColumnInfo**( *StatementNumber*, *ColumnNo*, *Attr* ), где

*StatementNumber* – целое значение, номер SQL-оператора.

*ColumnNo* – номер столбца в наборе; нумерация слева направо, начиная с 1.

*Attr* – код, указывающий на возвращаемую колонку.

### Возвращаемая величина

Возвращаемое значение зависит от значения атрибута (параметр *Attr*).

### Описание

Функция **Server\_ColumnInfo** возвращает информацию о текущем выбранном столбце результирующего набора (определенного ранее исполненным SQL-оператором **Оператор Select** в удаленной базе данных. Параметр *StatementNumber* задает номер-указатель (handle) SQL-оператора, ассоциированный с данным соединением с сервером данных. Параметр *ColumnNo* указывает столбец, информацию о котором Вы хотите получить. Параметр *Attr* выбирает тип возвращаемой информации.

В следующей таблице перечислены возможные атрибуты параметра *Attr*. Коды типов окон определены в MAPBASIC.DEF.

| Атрибут                | Server_ColumnInfo( ) возвращает:  |
|------------------------|---|
| SRV_COL_INFO_NAME      | Имя столбца.  |
| SRV_COL_INFO_TYPE      | <p>Целый результат; код типа столбца:</p> <ul style="list-style-type: none"> <li>• SRV_COL_TYPE_NONE</li> <li>• SRV_COL_TYPE_CHAR</li> <li>• SRV_COL_TYPE_DECIMAL</li> <li>• SRV_COL_TYPE_INTEGER</li> <li>• SRV_COL_TYPE_SmallInt</li> <li>• SRV_COL_TYPE_DATE</li> <li>• SRV_COL_TYPE_LOGICAL</li> <li>• SRV_COL_TYPE_FLOAT</li> <li>• SRV_COL_TYPE_FIXED_LEN_STRING</li> <li>• SRV_COL_TYPE_BIN_STRING</li> </ul> <p>Информацию об интерпретации типов данных приложением MapInfo Вы можете найти в Руководстве пользователя MapBasic.</p> |
| SRV_COL_INFO_SCALE     | Целый результат, указывающий число разрядов справа от десятичной точки для столбца типа SRV_COL_TYPE_DECIMAL, или -1 для столбца любого другого типа.   |
| SRV_COL_INFO_PRECISION | Целый результат, указывающий общее число разрядов для столбца типа SRV_COL_TYPE_DECIMAL, или -1 для столбца любого другого типа.  |
| SRV_COL_INFO_WIDTH     | <p>Целый результат, указывающий максимальное число символов в столбце типа SRV_COL_TYPE_CHAR или SRV_COL_TYPE_FIXED_LEN_CHAR.</p> <p>При использовании модуля QELIB пустой терминатор не учитывается. Возвращаемое значение совпадает с шириной столбца таблицы базы данных.</p>  |
| SRV_COL_INFO_VALUE     | Тип результата варьируется. Возвращается актуальное значение данных в столбце для текущей выбранной записи. Длинные строковые значения столбца, превышающие 32766 символов, усекаются. Двоичные (неструктурированные) значения столбца возвращаются в виде шестнадцатеричных символьных строк двойной длины.  |



| Атрибут             | Server_ColumnInfo( ) возвращает:   |
|---------------------|--|
| SRV_COL_INFO_STATUS | <p>Целый результат; статус значения столбца:</p> <ul style="list-style-type: none"> <li>• SRV_NULL_DATA - возвращается, если столбец не имеет данных для выбранной строки.</li> <li>• SRV_TRUNCATED_DATA - возвращается, если столбец содержит больше данных, чем может быть сохранено в указанной MapBasic-переменной.</li> <li>• Положительное целое значение – число байт, возвращенных сервером данных.</li> </ul> |
| SRV_COL_INFO_ALIAS  | <p>Строковый результат; псевдоним столбца, если в запросе данных использовался псевдоним.</p>  |

**Пример:**

```

Dim hdbc, Stmt As Integer
Dim Col As Integer
hdbc = Server_Connect("ODBC", "DLG=1")
Stmt = Server_Execute(hdbc, "Select * from emp")
Server Stmt Fetch NEXT
For Col = 1 To Server_NumCols(Stmt)
    Print Server_ColumnInfo(Stmt, Col, SRV_COL_INFO_NAME) +
        " = " +
        Server_ColumnInfo(Stmt, Col, SRV_COL_INFO_VALUE)
Next

```

**См. также:**

**Оператор Server Bind Column, Оператор Server Fetch, Функция Server\_NumCols( )**

## Оператор Server Commit

**Назначение**

Вызывает фиксацию транзакции в удаленной базе данных.

**Синтаксис**

**Server** *ConnectionNumber* **Commit**, где

*ConnectionNumber* – целое значение, номер соединения с базой данных;

**Описание**

Оператор **Server Commit** фиксирует транзакцию, т.е. сохраняет в базе данных изменения, произведенные в данном сеансе соединения всеми удаленными SQL-операторами, выполненными с момента исполнения оператора **Оператор Server Begin Transaction**. Оператор **Server Commit** выполняется только при наличии открытой транзакции,

## Функция `Server_Connect( )`

---

инициированной оператором **Оператор Server Begin Transaction**. Для запуска новой транзакции Вы должны выдать серверу новый оператор **Оператор Server Begin Transaction** , за которым в дальнейшем должен быть исполнен оператор **Server Commit**.

### Пример:

```
hdbc = Server_Connect("ODBC", "DLG=1")
Server hdbc Begin Transaction
hstmt = Server_Execute(hdbc, "Update Emp Set salary = salary * 1.5")
Server hdbc Commit
```

### См. также:

**Оператор Server Begin Transaction, Оператор Server Rollback**

---

## Функция `Server_Connect( )`

### Назначение

Устанавливает соединение с удаленным сервером данных.

### Синтаксис

**Server\_Connect**( *toolkit*, *connect\_string* )

*toolkit* указывает модуль интерфейса удаленного доступа MapInfo, "ODBC", "ORAINET", через который будет осуществляться соединение с сервером данных. Значения могут быть получены из функции **Функция Server\_DriverInfo( )**.

*connect\_string* параметр, который предоставляет интерфейсному модулю дополнительную информацию, необходимую для подключения к серверу данных.

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция **Server\_Connect( )** выполняет соединение с базой данных и возвращает номер-указатель соединения, который должен быть передан всем операторам удаленного доступа (как параметр ConnectionNumber), которые Вы хотите выполнить в данном сеансе соединения с сервером данных.

Параметр *toolkit* определяет модуль интерфейса удаленного доступа MapInfo (динамически загружаемую библиотеку), через который будет осуществляться соединение с сервером данных. Информация о возможных значениях параметра может быть получена вызовом функций **Функция Server\_NumDrivers( )** и **Функция Server\_DriverInfo( )**.

Параметр *connect\_string* передает модулю toolkit дополнительную информацию, необходимую для подключения к серверу данных. Значение строки подключения определяется требованиями удаленного сервера данных, к которому осуществляется доступ.

Строка подключения, задаваемая в функции **Server\_Connect( )**, имеет формат:

```
attribute=value[;attribute=value...]
```

**Внимание:** (В строке подключения должны отсутствовать пробелы.)

Прохождение соединения с DLG=1 обеспечивает удобный диалог соединения с активными кнопками справочной системы.

### Атрибуты Microsoft ACCESS

В следующей таблице перечислены атрибуты, используемые СУБД ACCESS:

| Атрибут | Описание   |
|---------|--|
| DSN     | Имя ODBC-источника для Microsoft ACCESS.   |
| UID     | Регистрационный идентификатор пользователя (ID).   |
| PWD     | Пароль пользователя.   |
| SCROLL  | По умолчанию значение равно NO. Если SCROLL=NO, то библиотека ODBC не используется для этого соединения, дающего возможность вызывать первую, последнюю, предыдущую или произвольную запись в базе данных. |

Пример строки подключения для СУБД ACCESS:

```
"DSN=MI ACCESS;UID=ADMIN;PWD=SECRET"
```

### Подключение к ORACLE через ODBC

Если Ваше приложение требует передачи строки параметров для обеспечения подключения к источнику данных, необходимо указать имя источника данных. Это имя, источника данных уже настроенного для использования в Вашей системе, который будет вызван драйвером, для определения стандартных параметров, обеспечивающих подключение. Кроме того, можно явно указать пары "*параметр-значение* параметра", которые могут быть использованы для подключения к базе данных без применения уже зарегистрированного в системе источника данных. В этом случае значения параметров не будут записаны в качестве системных.

Можно использовать как полные имена параметров, так и их сокращенную форму: Строка параметров выглядит следующим образом:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

Пример строки подключения к базе данных Oracle:

```
DSN=Accounting;HOST=server1;PORT=1522;SID=ORCL;UID=JOHN;PWD=XYZZY
```

Ниже перечислены полные имена параметров, их сокращенные формы, возможные используемые значения и краткие описания. Отдельно отмечены стандартные значения для каждого параметра, используемые при отсутствии этого параметра в строке подключения, как при явном указании имени параметра, так и при использовании системного имени подключения. При настройке параметра в подключении, имеющем имя в Вашей системе, такие настройки будут использованы как стандартные.

**ApplicationUsingThreads (AUT):** ApplicationUsingThreads={0 | 1}. Обеспечивает работу драйвера с многопоточковыми приложениями. Если установлен значение равно 1 (используется по умолчанию), драйвер гарантирует работу в каждом потоке. При использовании приложений, работающих с единственным потоком, можно установить значение параметра, равно 0. В этом случае не требуется дополнительная обработка запроса, требующаяся для обеспечения стандартов безопасности работы драйвера ODBC с несколькими потоками данных.

**ArraySize (AS):** Количество байт, которое использует драйвер при обращении к нескольким записям. Может принимать значение от 1 до 4GB. Большое значение увеличивает производительность за счет меньшего числа обращений по сети. Меньшее значение параметра уменьшает время отклика сервера, так как приходится передавать меньше данных. По умолчанию значение равно 60 000.

**CatalogOptions (CO):** CatalogOptions={0 | 1}. Определяет выполнение команд REMARKS для функций обращения к спискам SQLTables и SQLColumns и COLUMN\_DEF для функции обращения к списку SQLColumns при использовании Oracle. Если необходимо получить актуальные для сервера результаты выполнения таких команд, то необходимо установит значение CO равным 1. По умолчанию значение равно 0.

**DataSourceName (DSN):** Строка, определяющая имя источника данных в Вашей системе, обеспечивающего подключение к Oracle. В примерах используются имена "Accounting" или "Oracle-Serv1".

**DescribeAtPrepare (DAP):** DescribeAtPrepare={0 | 1}. Передает драйверу указание создавать описание команды SQL при ее подготовке. Стандартно используется значение 0, которое не приводит к созданию описания команды SQL при ее подготовке драйвером.

**EnableDescribeParam (EDP):** EnableDescribeParam={0 | 1}. Выключает или включает использование ODBC API функции SQLDescribeParam. Применение этой функции обеспечивает использование описателей для всех параметров, имеющих тип данных SQL\_VARCHAR. При использовании для доступа к данным Microsoft Remote Data Objects (RDO) этот атрибут необходимо установить, равным 1. По умолчанию значение равно 0.

**EnableStaticCursorsForLongData (ESCLD):** EnableStaticCursorsForLongData={0 | 1}. Определяет поддержку драйвером использования статического курсора при обращении к столбцам, имеющим тип данных Long. Поддержка статического курсора приводит к потерям производительности при обращении к данным, имеющим тип данных Long. По умолчанию значение равно 0.

**HostName (HOST):** HostName={servername | IP\_address}. Определяет имя сервера Oracle, к которому осуществляется подключение. Если Ваша сеть допускает обращение к серверу по имени, то можно указать нужное имя сервера, например, "Oracleserver". Во всех остальных случаях следует указать адрес IP, например, 199.226.224.34.

**LockTimeOut (LTO):** LockTimeOut={0 | -1}. Управляет включением блокировки ожидания отклика для генерации сообщения об ошибке при обработке запроса вида Select ...For **Onepatop Update**. Если установлено значение 0, то Oracle обрабатывает ошибку без задержки. Если установлено значение 1 (по умолчанию), задержка ожидания включена, но её значение не определено.

**LogonID (UID):** Задаёт идентификатор (имя) пользователя при входе в систему. Это имя используется приложением при установлении подключения (соединения) к базе данных Oracle. Этот идентификатор требуется только в том случае, если включены средства обеспечения безопасности и защиты базы данных. Для получения необходимого идентификатора следует обратиться к администратору базы данных.

**Password (PWD):** Пароль, который используется приложением для подключения к базе данных Oracle.

**PortNumber (PORT):** Устанавливает номер порта службы контроля соединений (listener) Oracle. По умолчанию значение равно 1521. Правильное значение можно узнать у администратора базы данных.

**ProcedureRetResults (PRR):** ProcedureRetResults={0 | 1}. Устанавливает возможность получения результатов от хранимых процедур функций. Если установлено значение 0 (используется по умолчанию) драйвер не возвращает результаты выполнения хранимых процедур. Если установлено значение 1, драйвер возвращает результаты выполнения хранимых процедур. В случае, если установлено 1, но после выполнения хранимой процедуры результат не может сформирован, следует ожидать небольшой потери производительности.

**SID (SID):** Системный идентификатор Oracle, по которому обращаются к экземпляру базы данных Oracle.

**UseCurrentSchema (UCS):** UseCurrentSchema={0 | 1}. Определяет, что при выполнении запросов вида SQLProcedures драйвером используются настройки только текущего пользователя. Если установлено значение, равное 0, то драйвер не определяет пользователя, от имени которого выполняется запрос. Если установлено значение, равное 1 (используется по умолчанию), то обращение к SQLProcedures оптимизируется, но возвращаются только процедуры, право применения которых, разрешено пользователю, от чьего имени осуществляется запрос.

## Oracle Spatial Attributes

Oracle8i Spatial - это новое издание пространственной базы данных от Oracle Corporation. Оно имеет некоторое сходство с ранним Oracle SDO. Oracle8i Spatial поддерживает Oracle SDO используя реляционную схему. MapInfo не поддерживает реляционную схему Oracle SDO через OCI. MapInfo не поддерживает одновременные соединения с Oracle8i через OCI и с другими базами данных через ODBC. MapInfo не поддерживает загрузку геометрических таблиц Oracle8i Spatial через ODBC используя текущий драйвер ODBC фирмы Intersolv. Здесь нет компонента DSN.

| Атрибут           | Описание  |
|-------------------|---|
| LogonID (UID)     | Имя пользователя (logon ID) которое приложение использует для связи с Вашей базой данных Oracle. Этот идентификатор требуется только в том случае, если включены средства обеспечения безопасности и защиты базы данных. Для получения необходимого идентификатора следует обратиться к администратору базы данных. |
| Password (PWD)    | Ваш пароль. Его тоже выдает системный администратор.  |
| ServerName (SRVR) | Имя сервера Oracle.   |

Пример строки соединения для доступа к серверу Oracle8i Spatial с использованием TCP/IP:

```
"SRVR=FATBOY;UID=SCOTT;PWD=TIGER"
```

### Атрибуты SQL SERVER

Если Ваше приложение требует передачи строки параметров для обеспечения подключения к источнику данных, необходимо указать имя источника данных. Это имя, источника данных уже настроенного для использования в Вашей системе, который будет вызван драйвером, для определения стандартных параметров, обеспечивающих подключение. Кроме того, можно явно указать пары "*параметр-значение* параметра", которые могут быть использованы для подключения к базе данных без применения уже зарегистрированного в системе источника данных. В этом случае значения параметров не будут записаны в качестве системных.

Строка параметров выглядит следующим образом:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

Пример строки подключения к базе данных SQL Server:

```
DSN=Accounting;UID=JOHN;PWD=XYZZY
```

Ниже перечислены полные имена параметров, их сокращенные формы, возможные используемые значения и краткие описания. Отдельно отмечены стандартные значения для каждого параметра, используемые при отсутствии этого параметра в строке подключения, как при явном указании имени параметра, так и при использовании системного имени подключения. При настройке параметра в подключении, имеющем имя в Вашей системе, такие настройки будут использованы как стандартные.

**Address:** сетевой адрес сервера, на котором выполняется SQL Server. Этот параметр следует использовать только в том случае, если в параметре Server явно не указано имя сервера на котором работает сервер базы данных SQL Server. Адресом может служить имя сервера в сети, а также другие способы адресации к этой вычислительной машине, например, имя или номер канала, TCP/IP адрес и номер порта, адрес программного интерфейса (socket). Например, для TCP/IP: 199.199.199.5, 1433 или MYSVR, 1433.

**AnsiNPW:** AnsiNPW={yes | no}. Определяет использование правил ANSI. В случае, если установлено значение, равное логической единице, драйвером используются правила ANSI (американского национального института стандартов), в которых определяются способы

обработки сравнений значений с пустыми элементами (NULL), последовательности символов и конкатенации значений с пустыми (NULL) элементами. В случае, если установлено значение, равное логическому нулю правила ANSI не применяются.

**APP:** Определяет имя приложения, вызывающего команду SQLDriverConnect (может использоваться для дополнительной проверки условий выполнения запроса). Если этот параметр используется, то значение указанное при его вызове будет сохранено в столбце master.dbo.sysprocesses. При вызове функций sp\_who и Transact-SQL APP\_NAME возвращается значение program\_name.

**AttachDBFileName:** Имя основного (primary) файла базы данных, к которой выполняется подключение. Должно содержать полный путь. Необходимо использовать выделение всех символов "косая черта" ( \ ) в случае, если используется объявление символьных переменных в C нотации:

```
AttachDBFileName=c:\\MyFolder\\MyDB.mdf
```

Устанавливает соединение с указанной базой данных и эта база данных становится используемой по умолчанию для этого подключения. Для того, чтобы применять адресацию к базе данных по имени файла с помощью параметра AttachDBFileName, необходимо объявлять соответствующее значение в параметре SQLDriverConnect, DATABASE или в атрибуте подключения через SQL\_COPT\_CURRENT\_CATALOG. Если соединение с базой данных было установлено ранее, SQL Server не будет заново его устанавливать. Соединение с этой базой данных будет использоваться по умолчанию.

**AutoTranslate:** AutoTranslate={yes | no}. Определяет способ перекодирования символов ANSI. В случае, если установлено значение равное логической единице, последовательности символов ANSI, передаваемые между сервером и клиентом, перекодируются в соответствии с кодовыми страницами Unicode. Таким образом удастся решить проблему соответствия дополнительных наборов символов на сервере и у клиента.

Перекодировка выполняется на вычислительной машине клиента драйвером SQL Server Wire Protocol. При этом необходимо, чтобы на сервере и у клиента были установлены одни и те же наборы кодовых страниц ANSI(ACP).

Настройка этого параметра не влияет на перекодировку символов при передаче данных следующих типов:

- Данные от клиента типа Unicode SQL\_C\_WCHAR передаются на сервер с типом данных char, varchar или text.
- Данные с сервера, имеющие тип char, varchar или text передаются клиенту для переменной типа Unicode SQL\_C\_WCHAR.
- Данные от клиента типа ANSI SQL\_C\_CHAR передаются на сервер в переменную с типом данных Unicode nchar, nvarchar или ntext.
- Данные с сервера, имеющие тип Unicode char, varchar или text передаются клиенту для переменной типа ANSI SQL\_C\_CHAR.
- Если значение параметра установлено равным логическому нулю (No) перекодировка не проводится.
- Драйвер SQL Server Wire Protocol не перекодирует символы ANSI SQL\_C\_CHAR при передаче от клиента на сервер данных в переменные типов char, varchar или text, параметров и имен столбцов. Не перекодируются символы из переменных типов char, varchar или text при передаче данных от сервера клиенту в переменные типа SQL\_C\_CHAR.

- Если на клиенте и на сервере SQL Server используются разные активные кодовые страницы (ACP), то в этом случае дополнительные символы могут обрабатываться неправильно.

**DATABASE:** Имя базы данных SQL Server, используемой в устанавливаемом подключении по умолчанию. Если этот параметр не задан, используется база данных определенная, как используемая по умолчанию для пользователя при входе в систему. Имя базы данных, определенное как используемое по умолчанию, в источнике данных ODBC, будет использовано вместо имени базы данных, определенной по умолчанию для пользователя при входе в систему, в случае если подключение устанавливается по имени источника данных ODBC. База данных должна существовать, кроме случая, когда применяется параметр `AttachDBFileName`. При совместном применении с параметром `AttachDBFileName`, используется основной (primary) файл, на который указывает значение параметра `AttachDBFileName`, а базе данных присваивается имя, определенное значением параметра `DATABASE`.

**LANGUAGE:** может использоваться как дополнительный, необязательный параметр. SQL Server может хранить системные сообщения на разных языках. При подключении к серверу SQL Server, для которого предусмотрена возможность выдавать системные сообщения на разных языках, можно указать язык, на котором следует создавать такие сообщения.

**Network:** Имя сетевой библиотеки. Для этого имени не нужно указывать полного пути и расширения имени файла .dll, например, `Network=dbnmpntw`.

**PWD:** Пароль, указываемый пользователем при входе в SQL Server, который определен в переменной `UID`. `PWD` не нужно определять, если для входа в систему установлен пустой пароль или в случае использовании авторизации пользователей средствами Windows NT (`Trusted_Connection=yes`).

**QueryLogFile:** Полное имя файла, который будет использоваться для создания журнала обработки данных при продолжительных запросах.

**QueryLog\_On:** `QueryLog_On={yes | no}`. Включает процесс создания журнала обработки продолжительных запросов. Если значение параметра установлено равным логической единице (Yes), то журнал обработки долгоиграющих запросов для этого подключения ведется. Если значение параметра установлено равным логическому нулю (No), то журнал обработки продолжительных запросов не будет вестись.

**QueryLogTime:** Последовательность цифр, определяющих интервал (в миллисекундах) записи журнала продолжительных запросов. Если отклик на запрос не пришел в течении заданного интервала, то этот запрос будет добавлен в журнал.

**QuotedID:** `QuotedID={yes | no}`. Включает или выключает обработку идентификаторов в кавычках. Если значение параметра установлено равным логической единице (Yes), то параметр `QUOTED_IDENTIFIER` — включен. SQL Server, в этом случае, использует правила SQL-92, определяющие использование кавычек в предложениях SQL. Если значение параметра установлено равным логическому нулю (No), то параметр `QUOTED_IDENTIFIER` — выключен. В этом случае, SQL Server, использует правила Transact-SQL, определяющие использование кавычек в предложениях SQL.

**Regional:** `Regional={yes | no}`. Включает автоматическое преобразование формы представления символов валюты, даты и времени. Если значение параметра установлено равным логической единице (Yes), то драйвер SQL Server Wire Protocol использует настройки



машины клиента при преобразовании символов валюты, даты и времени. Это преобразование — одностороннее: драйвер не распознает форматы символов валют или даты, не определенные стандартом ODBC. Если значение параметра установлено равным логическому нулю (No), то драйвер использует стандартные в ODBC форматы представления валюты, даты и времени и в соответствии с ними преобразует такие данные в последовательности символов.

**SAVEFILE:** Имя файла источника данных ODBC, в который записываются атрибуты соединения при успешном подключении.

**SERVER:** Имя сервера в сети, на котором работает SQL Server. В качестве такого имени можно использовать либо имя существующего в сети компьютера, либо имя описанное как существующее в SQL Server Client Network Utility. В случае использования копии SQL Server на том же самом компьютере, с которого выполняется подключение, можно ввести стандартное имя "(local)", как имя сервера Windows NT.

**StatsLogFile:** Полное имя, включая путь, файла, который будет использоваться для ведения журнала операций драйвера SQL Server Wire Protocol.

**StatsLog\_On:** StatsLog\_On={yes | no}. Включает и выключает сбор статистики операций, выполняемых драйвером SQL Server Wire Protocol. Если значение параметра установлено равным логической единице (Yes), то сведения об операциях, выполненных драйвером SQL Server Wire Protocol заносятся в журнал. Если значение параметра установлено равным логическому нулю (No), то сведения об операциях драйвера SQL Server Wire Protocol, выполненных с использованием этого подключения, не сохраняются.

**Trusted\_Connection:** Trusted\_Connection={yes | no}. Определяет сведения, которые будут использоваться драйвером SQL Server Wire Protocol, для подтверждения полномочий пользователя при входе в систему. Если значение параметра установлено равным логической единице (Yes), то драйвер SQL Server Wire Protocol будет работать в режиме проверки прав пользователей при входе в систему Windows NT Authentication Mode. Дополнительно можно задать параметры UID и PWD. Если значение параметра установлено равным логическому нулю (No), то для подтверждения прав доступа к данным пользователем драйвер SQL Server Wire Protocol будет использовать имя и пароль пользователя базы данных SQL Server. Обязательно нужно задать параметры UID и PWD.

**UID:** Полноценная учетная запись входа в систему SQL Server. Пользуясь штатными средствами подтверждения входа в систему Windows NT, параметр UID можно не задавать.

**WSID:** Идентификатор рабочей станции. Обычно, сетевое имя компьютера, на котором выполняется приложение. Использовать этот параметр — не обязательно. Если этот параметр задается, то его значение будет добавлено в столбец hostname таблицы master.dbo.sysprocesses. Это значение будет возвращаться по вызову функций sp\_who и Transact-SQL HOST\_NAME.

## Атрибуты Informix

Если Ваше приложение требует передачи строки параметров для обеспечения подключения к источнику данных, необходимо указать имя источника данных. Это имя, источника данных уже настроенного для использования в Вашей системе, который будет вызван драйвером, для определения стандартных параметров, обеспечивающих подключение. Кроме того, можно явно указать пары "*параметр-значение* параметра", которые могут быть использованы

для подключения к базе данных без применения уже зарегистрированного в системе источника данных. В этом случае значения параметров не будут записаны в качестве системных.

Можно использовать как полные имена параметров, так и их сокращенную форму: Строка параметров выглядит следующим образом:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

Строка подключения имеет следующий вид:

```
DSN=Informix TABLES;DB=PAYROLL
```

Ниже перечислены полные имена параметров, их сокращенные формы, возможные используемые значения и краткие описания. Отдельно отмечены стандартные значения для каждого параметра, используемые при отсутствии этого параметра в строке подключения, как при явном указании имени параметра, так и при использовании системного имени подключения. При настройке параметра в подключении, имеющем имя в Вашей системе, такие настройки будут использованы как стандартные.

**ApplicationUsingThreads (AUT):** ApplicationUsingThreads={0 | 1}. Обеспечивает работу драйвера с многопоточными приложениями. Если установлено значение равное 1 (используется по умолчанию), драйвер гарантирует работу в каждом потоке. При использовании приложений, работающих с единственным потоком, можно установить значение параметра, равное 0. В этом случае не требуется дополнительная обработка запроса, требующаяся для обеспечения стандартов безопасности работы драйвера ODBC с несколькими потоками данных.

**CancelDetectInterval (CDI):** Определяет значение в секундах, используемое как интервал проверки драйвером выполнения запроса, который может быть прерван командой SQLCancel. При обнаружении драйвером того факта, что была применена команда SQLCancel, запрос отменяется. Этот атрибут определяет окончены ли продолжительные запросы в многопоточных приложениях в случае использования команды SQLCancel. Если установлено значение, равное 0 (используется по умолчанию), выполнение запроса не прерывается даже в случае применения команды SQLCancel. Например, если установить значение параметра CancelDetectInterval равным 5, то драйвер будет проверять каждые пять секунд все ожидающие своей очереди на выполнение запросы, не пришло ли от приложения, в котором был создан этот запрос, прерывание его выполнения по команде SQLCancel.

**Database (DB):** Имя базы данных, к которой Вы хотите подключиться.

**DataSourceName (DSN):** Имя ODBC-источника данных для INFORMIX. В примерах используются имена "Accounting" или "Informix-Serv1".

**HostName (HOST):** Имя резидентной машины сервера INFORMIX.

**LogonID (UID):** Ваше имя пользователя для сервера INFORMIX.

**PortNumber (PORT):** Устанавливает номер порта службы контроля соединений (listener). Стандартного значения, используемого по умолчанию, нет.

**ServerName (SRVR):** Имя сервера на котором работает база данных Informix.

**TrimBlankFromIndexName (TBFIN):** TrimBlankFromIndexName={0 | 1}. Управляет способом обработки пустых символов или "пробелов" в начале имен индексов, автоматически созданных системой. Этот параметр предназначен для использования в приложениях,

которые не умеют обрабатывать пробелы в начале имен индексов. Если установлено значение, равное 1 (используется по умолчанию), драйвер отрезает пробелы в начале имен индексов. Если установлено значение, равное 0, драйвер оставляет пробелы в начале имен индексов.

**Пример:**

```
Dim hdbc As Integer
hdbc = Server_Connect("ODBC",
"DSN=Informix;SRVR=IUSSrvr;UID=atsmipro;PWD=miproats")
```

**См. также:**

**Оператор Server Disconnect**

**Функция Server\_ConnectInfo( )**

**Назначение**

Этот оператор присваивает геоинформацию таблице MapInfo, связанной с таблицей в удаленной базе данных.

**Синтаксис**

**Server\_ConnectInfo**( *ConnectionNo*, *Attr* ), где  
*ConnectionNumber* целое значение возвращаемое функцией **Функция Server\_Connect( )**, которая определяет номер соединения с сервером данных.  
*Attr* – код, определяющий характер возвращаемой информации.

**Возвращаемая величина**

Строка

**Описание**

Функция **Server\_ConnectInfo** возвращает информацию о соединении с базой данных. Первый параметр указывает номер соединения (начиная с 1). Второй параметр выбирает тип возвращаемой информации, как показано в следующей таблице:

| Атрибут                      | Server_ConnectInfo( ) возвращает:                                       |
|------------------------------|---|
| SRV_CONNECT_INFO_DRIVER_NAME | Строку, показывающую имя драйвера, соответствующего данному соединению. |
| SRV_CONNECT_INFO_DB_NAME     | Строку, возвращающую имя базы данных.                                   |
| SRV_CONNECT_INFO_SQL_USER_ID | Строку, возвращающую идентификатор пользователя SQL.                    |

| Атрибут                     | Server_ConnectInfo( ) возвращает:          |
|-----------------------------|--|
| SRV_CONNECT_INFO_DS_NAME    | Строку, возвращающую имя источника данных. |
| SRV_CONNECT_INFO_QUOTE_CHAR | Строку, возвращающую кавычки.              |

#### Пример:

```
Dim dbname as String
Dim hdbc As Integer
hdbc = Server_Connect("ODBC", "DLG=1")
dbname=Server_ConnectInfo(hdbc, SRV_CONNECT_INFO_DB_NAME)
Print dbname
```

#### См. также:

[Функция Server\\_Connect\( \)](#)

## Оператор Server Create Map

#### Назначение

Этот оператор присваивает геоинформацию таблице MapInfo, связанной с таблицей в удаленной базе данных. Модификация таблицы (например, добавлением в таблицу столбцов с пространственной информацией) при этом не осуществляет.

#### Синтаксис

```
Server ConnectionNumber Create Map
For linked_table
Type { MICODE columnname | XYINDEX columnname | SPATIALWARE }
CoordSys...
[ MapBounds { Data | Coordsys | Values ( x1, y1 ) ( x2, y2 ) } ]
[ ObjectType { Point | Line | Region | ALL } ]
[ Symbol (...) ]
[ Linestyle Pen(...) ]
[ Regionstyle Pen(...) Brush(...) ]
[ Style Type style_number [ Column column_name ] ]
```

*ConnectionNumber* – целое значение, номер соединения с базой данных;

*linked\_table* - имя открытой связанной ODBC-таблицы

*columnname* имя столбца, содержащего координаты специфицированного типа

*x1, y1, x2, y2* определяют границы системы координат.

**CoordSys...** предложение, задающее координатную систему и проекцию

**MapBounds** - предложение, позволяющее задавать границы видимости. По умолчанию используется значение **Data**, вычисляющее все границы слоя. (Для программ, созданных до версии 7.5, значение по умолчанию - **Coordsys**.).

Опция **Coordsys** определяет границы системы координат. Ее нежелательно использовать, поскольку слой может показаться пустым, если границы Coordsys значительно больше границ существующих данных. Большинство пользователей отдаляют карту слишком быстро и не успевают заметить свои данные.

Опция **Values** позволяет указать ваши собственные границы для MapCatalog.

**ObjectType** - предложение, задающее тип объекта в таблице: точки, линии, регионы или все объекты. Если это предложение не задано, по умолчанию используется тип **точки**.

**Symbol** (...) предложение, задающее стиль символа, используемого для точечного объекта.

**Linestyle Pen** (...) предложение, определяющее стиль линии, используемый для объекта типа линия.

**Regionstyle Pen** (...) **Brush**(...) предложение, задающее стиль линии и заливки фона, используемый для объекта типа область.

**StyleType** устанавливает символы для отдельных строк. *style\_number* - значение равно 0 или 1. Когда Type установлено на 1 (единицу), то подпредложение **Column** и его аргумент должны быть представлены. Когда *style\_number* установлен на ноль, то **Column** игнорируется и создаются колонки соответствия в MapCatalog.

## Описание

Оператор **Server Create Map** присваивает геоинформацию таблице MapInfo, связанной с таблицей в удаленной базе данных. Для таблицы SpatialWare, Oracle Spatial or Oracle SDO можно отразить на карте точки, линии и регионы. Для всех других таблиц можно отображать на карте только точки. Любая таблица MapInfo Professional может быть отображена в окне Списка, но только таблица с геоинформацией может иметь присоединенные графические объекты, и только такие таблицы могут быть отображены в окнах Карты MapInfo.

**Внимание:** Если сервер это Oracle9i и система координат определена как Долгота/Широта без определения датума, то по умолчанию будет использоваться стандартный датум World Geodetic System 1984(WGS 84). Такое поведение согласуется с оператором **Onepatop Server Create Table** и программой Easyloader.

| Типы атрибутов              | Описание  |
|-----------------------------|---|
| ORA_SP <i>columnname</i>    | OracleSpatial   |
| IUS_SW <i>columnname</i>    | SpatialWare IUS Blade                                 |
| IUS_MM_SW <i>columnname</i> | MapInfo MapMarker Geocoding DataBlade для SpatialWare |
| IUS_MM_XY <i>columnname</i> | MapInfo MapMarker Geocoding DataBlade для XY          |
| SPATIALWARE                 | SpatialWare для SQL Server                            |
| MICODE                      | XYINDEX   |

### Примеры

```
Sub Main
  Dim ConnNum As Integer
  ConnNum = Server_Connect("ODBC",
    "DSN=SQLServer;DB=QADB;UID=mipro;PWD=mipro")
  Server ConnNum Create Map For "Cities"
  Type SPATIALWARE
  CoordSys Earth Projection 1, 0
  ObjectType All
  ObjectType Point
  Symbol (35,0,12)
  Server ConnNum Disconnect
End Sub
```

См. также:

[Оператор Server Link Table](#), [Оператор Unlink](#)

---

## Оператор Server Create Style

### Назначение

Изменяет настройки стиля объекта для геокодированной таблицы. Оператор аналогичен оператору [Оператор Server Set Map](#) и возвращает результат или ошибку.

### Синтаксис

```
Server ConnectionNumber Set Map linked_table...  
[ Style Type style_number [ Column column_name ] ]
```

*ConnectionNumber* – целое значение, номер соединения с базой данных;

*linked\_table* - имя открытой связанной ODBC-таблицы

*columnname* имя столбца, содержащего координаты специфицированного типа

**StyleType** устанавливает символы для отдельных строк. *style\_number* - значение равное 0 или 1. Когда Type установлено на 1 (единицу), то подпредложение **Column** и его аргумент должны быть представлены. Когда *style\_number* установлен на ноль, то **Column** игнорируется и создаются колонки соответствия в MapCatalog.

### Описание

Чтобы оператор сработал правильно, Каталог карт должен иметь структуру, поддерживающую стили и должен содержать колонки RENDITIONTYPE, RENDITIONCOLUMN и RENDITIONTABLE. Команда не будет успешно выполнена, если колонки стилей не являются текстовыми. Оператор SQL сам выдаст ошибку, если попытается установить строковую величину в колонку с различными типами данных.

**Пример:**

```
Server 2 Create Map For "qadb:informix.arc"
Type MICODE "mi_sql_micode" ("mi_sql_x","mi_sql_y")
CoordSys Earth Projection 1, 0 ObjectType Point Symbol (35,0,12) Style
Type 1 Column "mi_symbology"
```

**См. также:**

**Функция `Server_Connect( )`**

---

## Оператор `Server Create Table`

**Назначение**

Создает новую таблицу в указанной удаленной базе данных.

**Синтаксис**

```
Server ConnectionNumber Create Table TableName
    ( ColumnName ColumnType [,...])
    [ KeyColumn ColumnName ]
    [ ObjectColumn ColumnName ]
    [ StyleColumn ColumnName ]
    [ CoordSys... ]
```

*ConnectionNumber* – целое значение, номер соединения с базой данных;

*Tablename* – строковая переменная, определяющая имя, которое нужно присвоить таблице базы данных;

*ColumnName* имя создаваемой колонки. Имя колонки может быть длиной до 31 символа, может содержать буквы, числа и символ подчеркивания(\_). Имя колонки не может начинаться с цифры.

*ColumnType* тип данных, ассоциированных с колонкой.

*KeyColumn* предложение, определяющее ключевую колонку таблицы.

*ObjectColumn* предложение, определяющее колонку пространственной геометрии/объектов таблицы.

*StyleColumn* предложение, определяющее колонку Per Row Style, которая позволяет использовать различные стили объектов для каждой записи таблицы.

*CoordSys*... предложение, задающее координатную систему и проекцию

**Описание**

Оператор **Server Create Table** создает новую пустую таблицу в базе данных с числом колонок до 250.

Длина имени таблицы *TableName* изменяется в зависимости от типа баз данных. Мы рекомендуем использовать 14 или менее символов для имени таблицы, чтобы быть уверенными при работе с любой базой данных. Таким образом, пусть максимальная длина имени таблицы будет 14 символов.

*ColumnType* использует тот же тип данных, что и определенный в **Оператор Create Table**. Некоторые типы данных могут быть конвертированы в те, которые поддерживаются используемой базой данных.

Если задано дополнительное предложение **KeyColumn**, то будет создан уникальный индекс для данной колонки. Мы рекомендуем использовать это предложение, так как оно позволяет MapInfo Professional открывать таблицу при прямом доступе к базе данных.

Дополнительное предложение **ObjectColumn** позволит Вам создать таблицу с колонкой пространственной геометрии/объектами. Если предложение определено, то пространственный индекс также будет создан для этой колонки. Таким образом, если сервер не имеет возможности обработать пространственную геометрию/объекты, то таблицы создана не будет. Если сервер это SQL Server со SpatialWare, то таблица будет настроена на пространственную геометрию/объекты с момента создания. Если сервер это Oracle Spatial, то пространственные метаданные обновятся в момент создания таблицы.

Если используется **Server Create Table**, и предложение **ObjectColumn** пропущено в операторе, Вам также надо будет использовать **Оператор Server Create Map** для того, чтобы открыть таблицу в MapInfo Professional.

Дополнительное предложение **Оператор CoordSys** становится обязательным только если таблица с пространственной геометрией/объектами создается на Oracle Spatial (Oracle8i или более поздние версии с пространственной поддержкой). Если сервер это Oracle9i и система координат определена как Долгота/Широта без определения датума, то по умолчанию будет использоваться стандартный датум World Geodetic System 1984(WGS 84). Система координат должна быть такой же как и система определенная в операторе **Оператор Server Create Map**. Для других баз данных это предложение не влияет на создание таблицы.

Поддерживаются следующие базы данных: Oracle, SQL Server, IUS (Informix Universal Server) и Microsoft Access. Таким образом, для создания таблицы с колонкой пространственной геометрии/объектами, SpatialWare/Blade требуется для SQL Server и IUS, а для Oracle требуются пространственные настройки.

### Замечания о типах данных Дата\Время, Время и Дата

Специальных синтаксических изменений при использовании нового типа данных нет. Существует следующие ограничения некоторых типов данных:

В версии 9.0 появились два новых типа данных MapInfo: Time (Время) и DateTime (Дата\Время). Однако, в большинство баз данных не включены соответствующие типы СУБД. Раньше мы обеспечивали работу только с типами данных Date (Дата). Даже тип Date преобразовывался в тип данных сервера, если сервер не поддерживал тип данных Date. Начиная с версии 9.0, этот оператор поддерживает только те типы данных, которые поддерживает выбранный сервер. Поэтому тип данных Time запрещен к использованию в этом операторе при обращении ко всем поддерживаемым серверам (Oracle, IBM Informix, MS



SQL Server и Access), а тип данных Date запрещен для MS SQL Server и Access. Если необходимо хранить в таблице базы данных сведения о времени в отдельной колонке, то эти "неподдерживаемые" типы следует заменять типом данных DateTime.

**Внимание:** Для серверов MS SQL Server и Access это может вызвать проблемы совместимости с предыдущими версиями. Ранее такие преобразования выполнялись в фоновом режиме. Теперь, для версии 9.0, необходимо вместо типа данных DATE использовать DATETIME. Если по-прежнему использовать тип данных DATE, то операция не будет выполнена.

```
Server ConnectionNumber Create Table TableName(ColumnName ColumnType [,...])
    [KeyColumn ColumnName]
    [ObjectColumn ColumnName]
    [StyleColumn ColumnName]
    Предложение CoordSys
```

*Tablename* – строковая переменная имени таблицы базы данных; Начиная с версии 9.0, имя таблицы может включать в себя имя схемы, которой принадлежит таблица. Если имени схемы не задано, то таблица принадлежит стандартной схеме, подобно тому как было в версии 8.5 и более ранних. Пользователь должен задать правильное имя схемы, должен знать имя учетной записи и обладать правами доступа к указанной схеме. Это касается только SQL Server 2005.

## Примеры

Следующие примеры показывают как создать таблицу с именем ALLTYPES, которая содержит семь колонок, охватывающих каждый из типов данных, поддерживаемых MI Pro, плюс три колонки Key, SpatialObject и Style. Всего колонок должно быть десять.

Для SQL Server со SpatialWare или IUS со SpatialWare Blade:

```
dim hodb as integer
hodb = server_connect("ODBC", "dlg=1")
Server hodb Create Table ALLTYPES( Field1 char(10),Field2 integer,Field3
SmallInt,Field4 float,Field5 decimal(10,4),Field6 date,Field7 logical)
KeyColumn SW_MEMBER
ObjectColumn SW_GEOMETRY
StyleColumn MI_STYLE
```

Для Oracle Spatial:

```
dim hodb as integer
hodb = server_connect("ORAINET", "SRVR=cygnus;UID=mipro;PWD=mipro")
Server hodb Create Table ALLTYPES( Field1 char(10),Field2 integer,Field3
SmallInt,Field4 float,Field5 decimal(10,4),Field6 date,Field7logical)
    KeyColumn MI_PRINX
    ObjectColumn GEOLOC
    StyleColumn MI_STYLE
    Coordsys Earth Projection 1, 0
```

**См. также:**

[Оператор Create Map](#), [Оператор Server Create Map](#), [Оператор Server Link Table](#), [Оператор Unlink](#)

## Оператор Server Create Workspace

### Назначение

Создает новый рабочий набор в базе данных (Oracle 9i или более поздние).

### Синтаксис

```
Server ConnectionNumber Create  
  Workspace WorkspaceName  
  [ Description Description ]  
  [ Parent ParentWorkspaceName ]
```

*ConnectionNumber* – целое значение, номер соединения с базой данных;

*WorkspaceName* - имя рабочего набора. Имя чувствительно к регистру и должно быть уникальным. Длина имени рабочего набора не должна превышать 300 символов.

*Description* - строка, описывающая рабочий набор.

*ParentWorkspaceName* - имя рабочего набора, являющегося родительским по отношению к новому рабочему набору *WorkspaceName*. По умолчанию, рабочий набор создается из последнего (LIVE) рабочего набора.

### Описание

Этот оператор применим только для Oracle9i и старше. Новый рабочий набор *WorkspaceName* является дочерним по отношению к родительскому рабочему набору *ParentWorkspaceName* или LIVE, если родительский рабочий набор не указан.

Более подробную информацию смотрите в *Oracle9i Application Developer's Guide - Workspace Manager*.

### Примеры

В следующем примере создается рабочий набор, названный MIUSER.

```
Dim hdbc As Integer  
hdbc = Server_Connect("ORAINET", "SRVR=TROYNY;UID=MIUSER;PWD=MIUSER")  
Server hdbc Create  
Workspace "MIUSER"  
Description "MIUser private workspace"
```

В следующем примере создается рабочий набор, дочерний по отношению к MIUSER.

```
Dim hdbc As Integer  
hdbc = Server_Connect("ORAINET", "SRVR=TROYNY;UID=MIUSER;PWD=MIUSER")  
Server hdbc Create Workspace "MBPROG" Description "MapBasic project"  
Parent "MIUSER"
```

### См. также:

[Оператор Server Remove Workspace](#), [Оператор Server Versioning](#)

---

## Оператор Server Disconnect

### Назначение

Прекращает связь, установленную с удаленным сервером данных вызовом функции **Функция Server\_Connect( )**.

### Синтаксис

**Server** *ConnectionNumber* **Disconnect**

*ConnectionNumber* – целое значение, номер соединения с базой данных;

### Описание

Оператор **Server Disconnect** отключает приложение от базы данных. Все ресурсы, выделенные для указанного соединения, возвращаются в распоряжение системы.

### Пример:

```
Dim hdbc As Integer  
hdbc = Server_Connect("ODBC", "DLG=1")  
Server hdbc Disconnect
```

### См. также:

**Функция Server\_Connect( )**

---

## Функция Server\_DriverInfo( )

### Назначение

Выдает информацию об установленном программном обеспечении и источниках данных.

### Синтаксис

**Server\_DriverInfo**( *DriverNo*, *Attr* ), где

*DriverNo* – целое значение, назначенное приложением MapInfo модулю интерфейса удаленного доступа при запуске MapInfo.

*Attr* – код, определяющий характер возвращаемой информации.

### Возвращаемая величина

Строка

## Функция Server\_EOT( )

### Описание

Функция **Server\_DriverInfo( )** возвращает информацию об источниках данных. Первый параметр выбирает модуль интерфейса удаленного доступа (начиная с 1). Общее число установленных интерфейсных модулей может быть получено вызовом функции **Функция Server\_NumDrivers( )**. Второй параметр выбирает тип возвращаемой информации, как показано в следующей таблице:

| Атрибут                | Server_DriverInfo( ) возвращает:  |
|------------------------|---|
| SRV_DRV_INFO_NAME      | Строку, показывающую имя модуля. ODBC указывает источник данных ODBC. ORAINET указывает соединение Oracle Spatial.  |
| SRV_DRV_INFO_NAME_LIST | Строку – список имен всех установленных интерфейсных модулей, разделенных точками с запятой. Specifically, ODBC, ORAINET. Параметр <i>DriverNo</i> игнорируется.  |
| SRV_DRV_DATA_SOURCE    | Строку – имена источников данных, поддерживаемых указанным модулем интерфейса. Последовательные вызовы функции последовательно выбирают имена источников. После выборки последнего имени для данного модуля функция возвратит пустую строку. Следующий вызов функции для того же модуля установит список на начало и возвратит первое имя в списке. |

### Пример:

```
Dim dlg_string, source As Stringdlg_string = Server_DriverInfo(0, SRV_DRV_INFO_NAME_LIST)source = Server_DriverInfo(1, SRV_DRV_DATA_SOURCE)While source <> ""Print "Доступные источники данных " + Server_DriverInfo(1, SRV_DRV_INFO_NAME) + ": " + sourcesource = Server_DriverInfo(1, SRV_DRV_DATA_SOURCE)Wend
```

### См. также:

**Функция Server\_NumDrivers( )**

## Функция Server\_EOT( )

### Назначение

Определяет, был ли достигнут конец результирующего набора в процессе последовательной выборки записей, выполнявшейся оператором **Оператор Server Fetch**.

### Синтаксис

```
Server_EOT( StatementNumber ), где  
StatementNumber – целое число, номер проверяемого оператора Server Fetch.
```

**Возвращаемая величина**

Логическое

**Описание**

Функция **Server\_EOT( )** возвращает TRUE или FALSE, в зависимости от результатов предыдущего оператора **Server Fetch**. Значение TRUE возвращается как при попытке выбрать предыдущую запись сразу же после выборки первой записи набора, так и в случае выборки следующей записи после последней записи набора.

**Пример:**

```
Dim hdbc, hstmt As Integerhdbc = Server_Connect("ODBC", "DLG=1")hstmt =
Server_Execute(hdbc, "Select * from ADDR")Server hstmt Fetch FIRSTWhile
Not Server_EOT(hstmt)' Обработка каждой строки данных ...Server hstmt
Fetch NextWend
```

**См. также:**

**Оператор Server Fetch**

**Функция Server\_Execute( )****Назначение**

Посылает SQL-строку для исполнения на удаленный сервер данных.

**Синтаксис**

**Server\_Execute( *ConnectionNumber*, *server\_string* ),** где

*ConnectionNumber* – целое значение, номер соединения с базой данных;

*server\_string* – строка, представляющая любой корректный SQL-оператор, поддерживаемый сервером, с которым установлено соединение. Информацию о корректных SQL-операторах смотрите в руководстве по языку SQL для СУБД на Вашем сервере.

**Возвращаемая величина**

Целое число типа Integer.

**Описание**

Функция **Server\_Execute( )** пересылает SQL-строку, заданную параметром *server\_string* и представляющую SQL-оператор, через соединение с сервером, указанное параметром *ConnectionNumber*. Любой корректный SQL-оператор, поддерживаемый активным сервером, является допустимым значением параметра *server\_string*. Информацию о корректных SQL-операторах смотрите в руководстве по языку SQL для СУБД на Вашем сервере.

Эта функция возвращает номер-указатель (handle) оператора, используемый для ассоциации (через параметр StatementNumber) последующих SQL-обращений (таких как **Оператор Server Fetch** и **Оператор Server Close**) с конкретным SQL-оператором.

Необходимо выполнить оператор **Server Close** для каждого вызова функции **Server\_Execute( )**. Для операторов типа **Select** – после выборки требуемых данных. При этом на удаленном сервере данных будет закрыт курсор и освобожден результирующий набор. В противном случае Вы можете превысить лимит на число открытых курсоров, и дальнейшие обращения к базе данных исполняться не будут. Не все серверы баз данных поддерживают курсоры с прямой и обратной прокруткой. Для других SQL-операторов используйте оператор **Server Close** сразу после вызова функции **Server\_Execute( )**.

```
Dim hdbc, hstmt As Integer
hdbc = Server_Connect("ODBC", "DLG=1")
hstmt = Server_Execute(hdbc, "Select * from ADDR")
Server hstmt Close
```

**Пример:**

```

Dim hdbc, hstmt As Integer
hdbc = Server_Connect("ODBC", DSN=ORACLE7;DLG=1")
hstmt = Server_Execute (hdbc,
    "CREATE TABLE NAME_TABLE (NAME CHAR (20))")
Server hstmt Close
hstmt = Server_Execute (hdbc,
    "INSERT INTO NAME_TABLE VALUES ('Steve')")
Server Close hstmt
hstmt = Server_Execute ( hdbc,
    "UPDATE NAME_TABLE SET name = 'Tim' ")
Server Close hstmt
Server hdbc Disconnect

```

**См. также:**

**Оператор Server Close, Оператор Server Fetch**

---

## Оператор Server Fetch

**Назначение**

Осуществляет выборку записей результирующего набора с удаленного сервера данных.

**Синтаксис**

**Server** *StatementNumber* **Fetch** [ **NEXT** | **PREV** | **FIRST** | **LAST** | [**REC**] *recno* ]

Or (оператор Или)

**Server** *StatementNumber* **Fetch INTO Table** [ **FILE** *path* ]

*StatementNumber* – целое значение, номер SQL-оператора.

*recno* - целое число, номер выбираемой записи.

*path* - путь к существующей таблице.

**Описание**

Оператор **Server Fetch** извлекает записи результирующего набора (заданного значением *StatementNumber* SQL-оператора, создавшего набор) из сервера данных. Для построчной выборки данных они помещаются в локальную область хранения, и могут быть связаны с переменными посредством команд **Оператор Server Bind Column**. Для выборки данных по столбцам применяется функция `Server_ColumnInfo(SRV_COL_INFO_VALUE)`. Можно также в одной операции выбрать полный результирующий набор в таблицу `MapInfo`, используя предложение **INTO Table**.

Выполнение операторов **Server Fetch** и **Server Fetch Into** прерывается с установкой кода ошибки `ERR( ) = ERR_SRV_ESC` при нажатии пользователем клавиши ESC, что позволяет Вашему MapBasic-приложению использовать команды **Server Fetch** для обработки этого события.

По исполнении оператора **Server Fetch Into** таблица MapInfo фиксируется, и для нее нет незавершенных транзакций. Все символьные поля, превышающие 254 байта, усекаются; все двоичные (неструктурированные) поля загружаются в таблицу как шестнадцатеричные символьные строки двойной длины. Имена столбцов в загруженной таблице будут использовать псевдонимы столбцов, если в запросе задавались псевдонимы.

### Обработка Null-значений

Если был исполнен SQL-оператор **Оператор Select** с последующей выборкой записи, включающей столбец таблицы, который содержит пустое (null) значение, то происходит следующее. Поскольку в MapInfo не поддерживается концепция пустых значений в таблице или переменной, используется значение по умолчанию в рамках домена для соответствующего типа данных – значение MapBasic-переменной, декларированной в инструкции Dim, но не инициализированной. Однако при этом обеспечивается индикация возврата пустого значения.

Для связанных переменных (см. оператор **Оператор Server Bind Column на стр. 43**) могут быть заданы переменные состояния, значения которых будут указывать на возврат пустого значения при выборке. Для столбцов, не связанных с переменными, функция Server\_ColumnInfo( ) с атрибутом SRV\_COL\_INFO\_STATUS будет возвращать статус столбца, информирующий о возможно пустом значении.

Смотрите в Приложении 4 *Руководства пользователя MapBasic* информацию о том, как MapInfo интерпретирует типы данных.

### Ошибки:

Оператор **Server n Fetch Into table** будет генерировать ошибку для любых неудачных попыток вставки записей в локальную таблицу MapInfo. Операторы типа **Server n Fetch** [**Next** | **Prev** | *recno*] генерируют ошибки, если запрашиваемой записи нет в наличии.

### Пример 1

```
' Пример выборки в таблицу MapInfoDim hdbc, hstmt As Integerhdbc =  
Server_Connect("ODBC", "DLG=1")hstmt = Server_Execute(hdbc, "Select * from  
emp")Server hstmt Fetch Into "MyEmp"Server hstmt Close
```

### Пример 2

```
' Пример выборки с использованием связанных переменныхDim hdbc, hstmt As  
Integerdim NameVar, AddrVar as Stringdim NameStatus, AddrStatus as  
Integerhdbc = Server_Connect("ODBC", "DLG=1")hstmt = Server_Execute(hdbc,  
"Select Name, Addr from emp")Server hstmt Bind Column 1 to NameVar,  
NameStatusServer hstmt Bind Column 2 to AddrVar, AddrStatusServer hstmt  
Fetch NextWhile Not Server_Eot(hstmt)Print "Name = " + NameVar + "  
Address = " + AddrVarServer hstmt Fetch NextWend
```

### См. также:

**Функция Server\_ColumnInfo( )**



## Функция `Server_GetODBCConn( )`

### Назначение

Возвращает целое значение, содержащее указатель связи ODBC, ассоциированной со связью с сервером данных.

### Синтаксис

`Server_GetODBCConn( ConnectionNumber )`, где

*ConnectionNumber* целое значение возвращаемое функцией [Функция `Server\_Connect\( \)`](#), которая определяет номер соединения с сервером данных.

### Описание

Эта функция возвращает целое значение, содержащее указатель связи ODBC, ассоциированной со связью с сервером данных. Это позволяет Вам любую функцию в ODBC DLL, чтобы расширить функциональные возможности, посредством использования MapBasic операторов типа **Server**.

### Пример:

```
'* Нахождение связи с сервером данных
DECLARE FUNCTION SQLGetInfo LIB
"ODBC32.DLL" (BYVAL odbchdbc AS INTEGER, BYVAL infoflag AS INTEGER, val AS
STRING, BYVAL len AS INTEGER, outlen AS INTEGER) AS INTEGER
Dim rc, outlen, hdbc, odbchdbc AS INTEGER
Dim DBName AS STRING
' Связь с сервером данных
hdbc = Server_Connect("ODBC", "DLG=1")
odbchdbc = Server_GetodbcHConn(hdbc) ' получение указателя связи ODBC
' Получение имени базы данных из ODBC
DBName = STRING$(33, "0")
Инициализация выходного буфера
rc = SQLGetInfo(odbchdbc, 17, DBName, 40,
outlen) ' получение имени базы данных ODBC
' Отображение результатов (имя базы данных)
if rc <> 0 THEN
Note "SQLGetInfo Error rc=" + rc + ", outlen=" + outlen
else
Note "Connected to
Database: " + DBName
end if
```

### См. также:

[Функция `Server\_GetODBCStmt\( \)`](#)

## Функция `Server_GetODBCStmt( )`

### Назначение

Возвращает указатель оператора ODBC, ассоциированный с MapBasic-оператором типа **Server**.

### Синтаксис

`Server_GetODBCStmt( StatementNumber )`, где

*StatementNumber* - целое значение, возвращаемое функцией [Функция `Server\_Execute\( \)`](#) которое указывает на результирующий набор исполненного SQL-оператора.

### Описание

Эта функция возвращает указатель оператора ODBC, ассоциированный с MapBasic-оператором типа **Server**. Это позволяет Вам вызывать любую функцию ODBC, для расширения функциональных возможностей посредством использования MapBasic-операторов типа **Server**.

### Пример:

```
' Найти количество строк, которые будут обновлятьсяDim rc, outlen, hdbc,
hstmt, odbchstmt AS INTEGERDim RowsUpdated AS INTEGER
' Найти количество строк, которые будут обновлятьсяDECLARE FUNCTION
SQLRowCount LIB "ODBC32.DLL" (BYVAL odbchstmt AS INTEGER, rowcnt AS
INTEGER) AS INTEGERRhdbc = Server_Connect("ODBC", "DLG=1")hstmt =
Server_Execute(hdbc, "UPDATE TIML.CUSTOMER SET STATE='NY' WHERE
STATE='NY'")odbchstmt = Server_GetodbcHStmt(hstmt)rc =
SQLRowCount(odbchstmt, RowsUpdated)
Note "Обновлено " + RowsUpdated + " новых клиентов первой галереи"
```

### См. также:

[Функция `Server\_GetODBCConn\( \)`](#)

## Оператор Server Link Table

### Назначение

Создает связанную таблицу.

### Синтаксис 1

```
Server Link Table
    SQLQuery
    Using ConnectionString
    Into TableName
    Toolkit Toolkitname
    [ File FileSpec ]
    [ ReadOnly ]
    [ Autokey { Off | On } ]
```

### Синтаксис 2

```
Server ConnectionNumber Link Table
    SQLQuery
    Into TableName
    Toolkit toolkitname
    [ File FileSpec ]
    [ ReadOnly ]
    [ Autokey { Off | On } ]
```

*ConnectionNumber* – целое значение, номер существующего соединения;

*SQLQuery* – SQL-оператор запроса (на языке SQL с добавлением объектных ключевых слов), который генерирует результирующий набор. Таблица MapInfo связывается именно с этим результирующим набором.

*ConnectionString* – строка, используемая для подключения к серверу базы данных (см. описание функции Server Connect). См. [Функция Server\\_Connect\( \) на стр. 48](#).

*TableName* – псевдоним создаваемой таблицы MapInfo.

*FileSpec* – имя табличного файла. Если этот параметр отсутствует, имя файла генерируется в текущем каталоге диска на базе псевдонима таблицы. Если параметр *FileSpec* задан, а табличный файл с указанным именем уже существует, то генерируется ошибка.

**ReadOnly** – задает использование таблицы только для чтения.

*Toolkitname* - строка, указывающая тип соединения, ODBC или ORAINET.

Если режим **Autokey** включен (**On**), таблица будет открыта в режиме автоинкремента. Если режим **Autokey** выключен (**Off**) или это опция отсутствует, таблица будет открыта без применения режима автоинкремента.

### Описание

Этот оператор создает связанную таблицу MapInfo на диске. Эта таблица открывается и к ней обращается запрос. Связанная таблица обрабатывается как обычная таблица MapInfo в большинстве случаев, кроме следующих: оператор **Оператор Alter Table** не выполняется для связанных таблиц. Связанные таблицы не могут быть упакованы. В список диалога “Упаковка” эти таблицы не включаются. Синтаксис **Server Link Table** используется для установки связи сервера базы данных и связанной таблицы. **Server ConnectionNumber Link Table** используются для связи таблицы с подключенным сервером базы данных. Связанные таблицы содержат информацию, позволяющую восстановить соединения и идентифицировать удаленные данные, в которые нужно внести изменения. Эта информация сохранена как метаданные в файле таблицы.

Отсутствие ключевого слова **ReadOnly** не означает возможности редактирования таблицы. Связанная таблица может быть запрещена для записи при следующих обстоятельствах: 1) результирующий набор разрешен только для чтения; 2) результирующий набор не содержит первичного ключа; 3) в результирующем наборе отсутствуют редактируемые столбцы; 4) указан режим **ReadOnly**. Если используется сервер Oracle, **Autokey** указывает используется ли режим автоинкремента.

### Синтаксис запросов SQL

Ключевое слово MapInfo **OBJECT** может быть использовано для установления связи между пространственными колонками и запросом SQL. MapInfo Professional преобразует ключевое слово **OBJECT** в соответствующие пространственные колонки. Таблица с именем **SELECT\*FROM (tablename)** всегда включает эти пространственные колонки, но если вы хотите указать подмножество колонок, используйте ключевое слово **OBJECT**. Например:

```
SELECT col1, col2, OBJECT
FROM tablename
```

будут загружены две колонки и пространственный объект. Этот синтаксис будет работать в любой базе данных, поддерживаемой MapInfo Professional.

### Пространственные запросы MapInfo Professional

MapInfo Professional поддерживает ключевое слово **WITHIN**, используемое в пространственных запросах. Оно используется для выбора пространственных объектов в таблице. Следующие два ключевых слова могут использоваться вместе с ключевым словом **WITHIN**:

- **CURRENT MAPPER**: вся прямоугольная область, отображаемая в текущем окне Карты.
- **SELECTION**: выбранная область в текущем окне Карты.

Синтаксис поиска всех строк, содержащих пространственные объекты, существующих в текущем окне Карты, будет выглядеть следующим образом:

```
SELECT col1, col2, OBJECT
FROM tablename
WHERE OBJECT WITHIN CURRENT_MAPPER
```

Этот синтаксис будет работать для любым поддерживаемых MapInfo Professional баз данных. MapInfo Professional будет также выполнять пространственные запросы SQL, написанных на языке SQL для пространственных баз данных. Корректные значения для *toolkitname* смотрите в **Функция `Server_DriverInfo( )` на стр. 65**.

### Примеры

```
Declare Sub Main
Sub Main
  Open table "C:\mapinfo\data\states.tab"
  Server Link Table "Select * from Statecap" Using
    "DSN=MS Access;DBQ=C:\MSOFFICE\ACCESS\DB1.mdb"
  Into test File "C:\tmp\test"
  Map From Test,States
End Sub 'Main
Declare Sub Main
Sub Main
  Dim ConnNum As Integer
  ConnNum = Server_Connect("ODBC", "DSN=SQS;PWD=sysmal;SRVR=seneca")
  Server ConnNum Link Table
    "Select * from CITY_1"
  Into temp
  Map From temp

  Server ConnNum Disconnect
End Sub
```

В следующем примере показано создание связанных таблиц.

```
Dim hdbc As Integer
hdbc = Server_Connect("ORAINET", "SRVR=ONTARIO;UID=MIPRO;PWD=MIPRO")
Server hdbc link table
  "Select * From ""MIPRO"". ""SMALLINTEGER""
  Toolkit "ORAINET"
  Into SMALLINTEGER
  Autokey ON
Map From SMALLINTEGER
```

**См. также:**

**Оператор Close Table, Оператор Commit Table, Оператор Drop Table, Оператор Rollback, Оператор Save File, Оператор Server Refresh, Оператор Unlink**

---

## Функция Server\_NumCols( )

### Назначение

Возвращает число столбцов в результирующем наборе.

### Синтаксис

**Server\_NumCols( *StatementNumber* ),** где

*StatementNumber* – целое значение, номер SQL-оператора.

### Возвращаемая величина

Целое число типа Integer.

**Описание**

Функция **Server\_NumCols( )** возвращает число столбцов в результирующем наборе, ссылка на который осуществляется по указателю *StatementNumber*.

**Пример:**

```
Dim hdbc, hstmt As Integerhdbc = Server_Connect("ODBC", "DLG=1")hstmt =  
Server_Execute(hdbc, "Select Name, Addr from emp")Print "Число колонок = "  
+ Server_NumCols(hstmt)
```

**См. также:**

[Функция Server\\_ColumnInfo\( \)](#)

---

**Функция Server\_NumDrivers( )****Назначение**

Возвращает число интерфейсных модулей удаленного доступа к базам данных, установленных в данный момент для доступа из MapInfo.

**Синтаксис**

```
Server_NumDrivers( )
```

**Возвращаемая величина**

Целое число типа Integer.

**Описание**

Функция **Server\_NumDrivers( )** возвращает число модулей интерфейса удаленного доступа, через которые может осуществляться соединение с удаленным сервером данных, установленных для использования приложением MapInfo.

**Пример:**

```
Print "Number of drivers = " + Server_NumDrivers( )
```

**См. также:**

[Функция Server\\_DriverInfo\( \)](#)

## Оператор Server Refresh

### Назначение

Осуществляет синхронизацию связанной таблицы MapInfo с данными в удаленной базе данных. Эта команда может выполняться только в отсутствие ожидающих запросов на редактирование связанной таблицы.

### Синтаксис

**Server Refresh** *TableName*

*TableName* – имя открытой связанной таблицы MapInfo.

### Описание

Если соединение с базой данных открыто, произойдет обновление данных. Если соединение с базой данных отсутствует, оно будет создано. Если потребуется какая-либо информация (например, пароль), она будет запрошена.

Синхронизация связанной таблицы включает следующие этапы:

1. Удаление всех записей и объектов из связанной таблицы (если она содержит записи) методом удаления и воссоздания табличного файла (не используя оператор MapBasic **Оператор Delete**).
2. Если указатель (handle) соединения сохранен вместе со структурой TABLE, используется этот указатель. В противном случае соединение с сервером базы данных производится с использованием строки подключения, сохраненной в метаданных связанной таблицы.
3. Выполняется преобразование SQL-запроса, сохраненного в метаданных, в запрос, специфичный для удаленной СУБД.
4. Этот SQL-запрос исполняется на удаленном сервере базы данных.
5. Таблица заполняется строками, выбранными курсором из удаленной СУБД. Эта операция может сопровождаться индикатором выполнения MapInfo.
6. Курсор в удаленной СУБД закрывается.

### Пример:

```
Server Refresh "City_1k"
```

### См. также:

**Оператор Commit Table, Оператор Server Link Table, Оператор Unlink**



---

## Оператор Server Remove Workspace

### Назначение

Удаляет все строки из рабочего набора и сам рабочий набор из базы данных (Oracle 9i и старше).

### Синтаксис

```
Server ConnectionNumber Remove  
Workspace WorkspaceName
```

*ConnectionNumber* – целое значение, номер соединения с базой данных;

*WorkspaceName* - имя рабочего набора. Чувствительно к регистру.

### Описание

Этот оператор применим только для Oracle9i и старше. Эта операция возможна только для терминальных рабочих наборов. В удаляемом рабочем наборе не должно быть других пользователей.

### Примеры

В следующем примере удаляется рабочий набор MIUSER.

```
Dim hdbc As Integer  
hdbc = Server_Connect("ORAINET", "SRVR=TROYNY;UID=MIUSER;PWD=MIUSER")  
Server hdbc Remove Workspace "MIUSER"
```

См. также:

[Оператор Server Create Workspace](#)

---

## Оператор Server Rollback

### Назначение

Выполняет откат транзакции на удаленном сервере данных.

### Синтаксис

```
Server ConnectionNumber Rollback
```

*ConnectionNumber* – целое значение, номер соединения с базой данных;

### Описание

Оператор **Server Rollback** ликвидирует все изменения, внесенные в базу данных SQL-операторами, исполненными в данном соединении с сервером с момента исполнения оператора **Оператор Server Begin Transaction**, и восстанавливает исходное состояние базы данных.. Для выдачи этого оператора необходимо иметь открытую транзакцию, инициированную оператором **Оператор Server Begin Transaction**.

### Пример:

```
hdbc = Server_Connect("ODBC", "DLG=1")Server hdbc Begin Transaction...'
Все изменения, внесенные с момента исполнения Begin_Transaction, ' будут
отменены (откат транзакции)Server hdbc Rollback
```

### См. также:

**Оператор Server Begin Transaction, Оператор Server Commit**

---

## Оператор Server Set Map

### Назначение

Этот оператор позволяет менять стили объектов для изображаемой в виде Карты таблицы ODBC. Это действие отображается в MapCatalog.

### Синтаксис

```
Server ConnectionNumber Set Map linked_table
[ ObjectType { Point | Line | Region } ]
[ Symbol(...) ]
[ Linestyle Pen(...) ]
[ Regionstyle Pen(...) Brush(...) ]
```

*ConnectionNumber* – целое значение, номер соединения с базой данных;

*linked\_Name* – имя открытой связанной таблицы СУБД.

**ObjectType** – указывает тип объектов в таблице и позволяет определять объекты как регионы, линии или объекты. Подробнее смотрите в **Оператор Server Create Map на стр. 58**.

**Symbol (...)** предложение, задающее стиль символа, используемого для точечного объекта.

**Linestyle Pen (...)** предложение, определяющее стиль линии, используемый для объекта типа линия.

**Regionstyle Pen (...) Brush(...)** предложение, задающее стиль линии и заливки фона, используемый для объекта типа область.

### Описание

Оператор **Server Set Map** изменяет стили объектов на Карте открытой таблицы ODBC. Таблица ODBC становится отображаемой в виде Карты с помощью оператора **Оператор Server Create Map**.

**Пример:**

```

Declare Sub Main
Sub Main
  Dim ConnNum As Integer
  ConnNum = Server_Connect("ODBC", "DSN=SQS;PWD=sys;SRVR=seneca")
  Server ConnNum Set Map "Cities"
  ObjectType Point
  Symbol (35,0,12)
  Server ConnNum Disconnect
End Sub

```

**См. также:**

**Оператор Server Create Map**

---

## Оператор Server Versioning

**Назначение**

Включает или выключает контроль версий для таблиц Oracle версии 9i или более поздних, которые создают или удаляют все необходимые структуры для поддержки истории строк, что позволяет воспользоваться преимуществами Oracle Workspace Manager.

**Синтаксис**

```

Server ConnectionNumber Versioning {
  ON [ History HistoryValue ] |
  OFF [ Force { OFF | ON } ]
} Table ServerTableName

```

**ON | OFF** указывает включен (**ON**) или выключен (**OFF**) контроль версий для таблицы.

*ConnectionNumber* – целое значение, номер соединения с базой данных;

*ServerTableName* - имя таблицы на сервере Oracle. Длина имени таблицы не должна превышать 300 символов. Имя не чувствительно к регистру.

**History** - не обязательный параметр при включенном контроле версий таблицы (**ON**).

Предложение **History** отслеживает изменения таблицы *ServerTableName*, т.е. позволяет запоминать изменения в таблице, к которой применяется контроль версий. *HistoryValue* должно принимать одно из следующих значений:

- **SRV\_WM\_HIST\_NONE** (0): Никаких изменений в таблице не отслежено. (Этот режим является стандартным.)
- **SRV\_WM\_HIST\_OVERWRITE** (1): используется перезаписывание (W\_OVERWRITE). В окне *ServerTableName\_HIST* хранится информация об изменениях. Но будут показаны только самые последние изменения для той же версии таблицы. История изменений этой версии не сохраняется, поскольку последние изменения записей перезаписываются на более ранние. (Колонка CREATETIME окна *TableName\_HIST* содержит только время последних изменений.)

- **SRV\_WM\_HIST\_NO\_OVERWRITE (2)**: перезаписывание не используется (**WO\_OVERWRITE**). В окне *ServerTableName\_HIST* хранится информация обо всех изменениях этой версии таблицы. История изменений этой версии сохраняется, поскольку последние изменения записей не перезаписываются на более ранние.

Но для использования этой функции существует много ограничений. Более подробную информацию смотрите в *Oracle9i Application Developer's Guide - Workspace Manager*.

**Force** - необязательный параметр, используемый при контроле версий (**OFF**).

Если режим **Force** включен (**ON**), то все данные рабочих наборов, за исключением LIVE, будут удалены перед отключением контроля версий. **OFF** (по умолчанию) предупреждает отключение контроля версий в случае, если *ServerTableName* была изменена в каком-либо из рабочих наборов (кроме LIVE) и если рабочий набор изменивший *ServerTableName* еще существует.

### Описание

Этот оператор применим только для Oracle9i и старше. Таблица *ServerTableName*, к которой применяется контроль версий, должна иметь первичный ключ. Только хозяин таблицы или пользователь с правом WM\_ADMIN может включать или отключать режим контроля версий таблицы. Таблицы, для которых осуществляется контроль версий, и пользователи, имеющие такие таблицы, не могут быть удалены. Для этого нужно сначала отключить контроль версий соответствующей таблицы. К таблицам, принадлежащим SYS не может применяться контроль версий. Более подробную информацию смотрите в *Oracle9i Application Developer's Guide - Workspace Manager*.

### Примеры

В следующем примере включается контроль версий для таблицы MIUUSA3.

```
Dim hdbc As Integer
hdbc = Server_Connect("ORAINET", "SRVR=TROINY;UID=MIUSER;PWD=MIUSER")
Server hdbc Versioning ON Table "MIUUSA3"
```

Or (оператор Или)

```
Server hdbc Versioning ON History 1 Table "MIUUSA3"
```

В следующем примере отключается контроль версий для таблицы MIUUSA3.

```
Dim hdbc As Integer
hdbc = Server_Connect("ORAINET", "SRVR=TROINY;UID=MIUSER;PWD=MIUSER")
Server hdbc Versioning OFF Force ON Table "MIUUSA3"
```

**См. также:**

**Оператор Server Create Workspace**

## Оператор Server Workspace Merge

### Назначение

Синхронизирует изменения таблицы (всех строк или тех, что указаны в предложении *Where*) в рабочем наборе с родительским рабочим набором в базе данных (Oracle 9i и старше).

### Синтаксис

```
Server Workspace Merge
  Table TableName
  [ Where WhereClause ]
  [ RemoveData { OFF | ON } ]
  [ { Interactive | Automatic merge_keyword } ]
```

*TableName* - имя таблицы (псевдоним), открытой MapInfo с сервера Oracle9i. Таблица содержит строки для слияния с родительским рабочим набором.

*WhereClause* - строка, указывающая записи для слияния с родительским рабочим набором.

*merge\_keyword* - ключевое слово, ограничивающее слияние в автоматическом режиме **Automatic**.

### Описание

Этот оператор применим только для Oracle9i и старше. Все данные, из таблицы *TableName*, указанные предложением *WhereClause* будут синхронизированы с родительским рабочим набором. Снимаются любые блокировки объединяемых строк. Если в процедуре слияния возникает конфликт, пользователю будут предложены возможные пути решения этой проблемы. Операция слияния будет выполнена только после разрешения всех конфликтов. В рабочем наборе LIVE таблица не может быть синхронизирована, поскольку этот рабочий набор не имеет родительского рабочего набора. Таблица не может подвергаться слиянию или обновлению если существует открытая транзакция базы данных, затрагивающая эту таблицу.

Более подробную информацию смотрите в *Oracle9i Application Developer's Guide - Workspace Manager*.

*WhereClause* - указывает строки для слияния с родительским рабочим набором. Само предложение не должно содержать ключевое слово **Where**. Например, 'MI\_PRINX = 20'. В предложении **Where** могут быть указаны только колонки с первичным ключом. Предложение **Where** не может содержать подзапрос. Если предложение *WhereClause* не указано, слиянию подлежат все строки в *TableName*.

Если режим **RemoveData** включен (**ON**), данные в таблице (в соответствии с *WhereClause*) дочернего рабочего набора будут удалены. Этот режим доступен только если рабочий набор не имеет дочернего рабочего набора. **OFF** (используется по умолчанию) - данные не будут удалены.

Если в процедуре слияния возникает конфликт, пользователю необходимо его разрешить. MapInfo Professional позволяет пользователю сначала разрешить конфликт, а затем выполнить процедуру слияния. С помощью параметров **Interactive** и **Automatic** можно управлять конфликтами. Эти параметры не действуют, если конфликта нет.

Если выбран параметр **Interactive**, MapInfo Professional откроет диалог Conflict Resolution. Конфликты будут разрешены по одному или все сразу в соответствии с выбором пользователя. После того, как все конфликты будут разрешены, операция слияния будет выполнена.

**Внимание:** Из-за системных ограничений эта функция не доступна для сервера Oracle9i.

В следующей таблице указаны возможные значения для *merge\_keyword*, используемые в режиме **Automatic**.

| значение<br>merge_keyword | Описание   |
|---------------------------|--|
| StopOnConflict            | При возникновении конфликта, MapInfo остановится в месте его появления. (Этот режим используется по умолчанию, то есть в случае, если не используются параметры <b>Interactive</b> или <b>Automatic</b> .)   |
| RevertToBase              | При возникновении конфликта, MapInfo восстанавливает исходные (базовые) значения. (исходные записи копируются в дочерний рабочий набор, но не в родительский. Однако, конфликт считается разрешенным, и когда выполняется объединение дочернего набора, базовые строки также копируются в родительский набор. Обратите внимание, что BASE игнорируется для конфликтов типа вставка-вставка, если базовая строка не существует; в этом случае предложение <b>Automatic</b> должно содержать UseParent или UseCurrent. |
| UseCurrent                | При возникновении конфликта, MapInfo использует значения дочернего рабочего набора.  |
| UseParent                 | При возникновении конфликта, MapInfo использует значения родительского рабочего набора.  |

### Примеры

В следующем примере объединяются изменения в таблице GWMUSA2 where MI\_PRINX=60 in MIUSER и ее родительским рабочим набором.

```
Server Workspace Merge
  Table "GWMUSA2"
  Where "MI_PRINX = 60"
  Automatic UseCurrent
```

**См. также:**

**Оператор Server Workspace Refresh**

## Оператор Server Workspace Refresh

### Назначение

Синхронизирует все изменения таблицы (все строки или указанные предложением **Where**) в родительском рабочем наборе с рабочим набором в базе данных (Oracle 9i или более новой).

### Синтаксис

**Server Workspace Refresh**

```
Table TableName
[ Where WhereClause ]
[ { Interactive | Automatic merge_keyword } ]
```

*TableName* - имя таблицы (псевдоним), открытой MapInfo с сервера Oracle9i. Таблица содержит строки, предназначенные для обновления значениями из родительского рабочего набора.

*WhereClause* - указывает строки для обновления из родительского рабочего набора. Само предложение не должно содержать ключевое слово **Where**.

*merge\_keyword* - строка с ключевыми словами, ограничивающими обновление в автоматическом режиме (**Automatic**).

### Описание

Этот оператор применим только для Oracle9i и старше. Он применяет к рабочему набору все изменения в строках таблицы родительского рабочего набора (указанных параметром *WhereClause*), которые были произведены с момента создания рабочего набора до последнего обновления. Если в процедуре обновления возникает конфликт, пользователю будут предложены возможные пути решения этой проблемы. Операция обновления будет применена только после разрешения всех конфликтов. В рабочем наборе LIVE таблица не может быть обновлена, поскольку этот рабочий набор не имеет родительского рабочего набора. Таблица не может подвергаться слиянию или обновлению если существует открытая транзакция базы данных, затрагивающая эту таблицу.

Более подробную информацию смотрите в *Oracle9i Application Developer's Guide - Workspace Manager*.

*WhereClause* - указывает строки для обновления из родительского рабочего набора. Само предложение не должно содержать ключевое слово **Where**. Например, `MI_PRINX = 20`. В предложении **Where** могут быть указаны только колонки с первичным ключом. Предложение **Where** не может содержать подзапрос. Если предложение *WhereClause* не указано, будут обновлены все строки в *TableName*.

Если в процедуре обновления возникает конфликт, пользователю необходимо его разрешить. MapInfo Professional позволяет пользователю сначала разрешить конфликт, а затем выполнить процедуру обновления. С помощью параметров **Interactive** и **Automatic** можно управлять конфликтами. Эти параметры не действуют, если конфликта нет.

Если выбран параметр **Interactive**, MapInfo Professional откроет диалог Conflict Resolution. Конфликты будут разрешены по одному или все сразу в соответствии с выбором пользователя. После того, как все конфликты будут разрешены, операция обновления будет выполнена.

**Внимание:** Из-за системных ограничений эта функция не доступна для сервера Oracle9i.

В следующей таблице указаны возможные значения для *merge\_keyword*, используемые в режиме **Automatic**.

| значение<br>merge_keyword | Описание   |
|---------------------------|--|
| StopOnConflict            | При возникновении конфликта, MapInfo остановится в месте его появления. (Этот режим используется по умолчанию, то есть в случае, если не используются параметры <b>Interactive</b> или <b>Automatic</b> .)   |
| RevertToBase              | При возникновении конфликта, MapInfo восстанавливает исходные (базовые) значения. (исходные записи копируются в дочерний рабочий набор, но не в родительский. Однако, конфликт считается разрешенным, и когда выполняется объединение дочернего набора, базовые строки также копируются в родительский набор. Обратите внимание, что BASE игнорируется для конфликтов типа вставка-вставка, если базовая строка не существует; в этом случае предложение <b>Automatic</b> должно содержать UseParent или UseCurrent. |
| UseCurrent                | При возникновении конфликта, MapInfo использует значения дочернего рабочего набора.  |
| UseParent                 | При возникновении конфликта, MapInfo использует значения родительского рабочего набора.  |

### Примеры

В следующем примере показано как обновить MIUSER, применяя изменения GWMUSA2 where MI\_PRINX=60 в родительском рабочем наборе.

```
Server Workspace Refresh
  Table "GWMUSA2"
  Where "MI_PRINX = 60"
  Automatic UseParent
```

**См. также:**

**Оператор Server Workspace Merge**



## Функция SessionInfo( )

### Назначение

Возвращает различные блоки информации о сеансе работы MapInfo Professional.

### Синтаксис

**SessionInfo**( *attribute* ), где

*attribute* – целочисленный код, определяющий тему запроса.

### Возвращаемая величина

Строка

### Описание

Функция **SystemInfo**( ) возвращает информацию о текущем сеансе программе MapInfo Professional. Вид информации задает параметр *attribute*. Коды определены в MAPBASIC.DEF.

| Значения <i>attribute</i>    | Возвращаемая величина                          |
|------------------------------|--|
| SESSION_INFO_COORDSYS_CLAUSE | Строка, возвращающая единицы измерения отчёта. |
| SESSION_INFO_DISTANCE_UNITS  | Строка возвращающая единицы расстояния.        |
| SESSION_INFO_AREA_UNITS      | Строка возвращающая единицы площади.           |
| SESSION_INFO_PAPER_UNITS     | Строка возвращающая единицы измерения отчёта.  |

### Ошибки:

ERR\_FCN\_ARG\_RANGE, если значение аргумента выходит за пределы, заданные при его определении.

### Пример:

```
Include "mapbasic.def"
print SessionInfo(SESSION_INFO_COORDSYS_CLAUSE)
```

## Оператор Set Application Window

### Назначение

Устанавливает, какое окно будет порождающим для всех новых диалогов и окон.

### Синтаксис

**Set Application Window** *HWND*

*HWND* – целое число типа Integer, уникальный системный номер окна.

### Описание

Этот оператор объявляет, какое окно будет окном приложения. Для всех последующих окон диалогов MapInfo будет считаться, что они порождены этим другим окном. Этот прием используется в “Интегрированной Картографии”, когда окна MapInfo показываются из других приложений, написанных, например, на Visual Basic.

Обычно Ваша программа, написанная на Visual Basic, сначала создает объект MapInfo Object и затем посылает MapInfo оператор **Оператор Set Application Window**, после чего приложение на Visual Basic становится порождающим окном для диалогов MapInfo. Если оператор **Оператор Set Application Window** не был выполнен, то становится очень трудно координировать передачу фокуса между MapInfo и Visual Basic.

Использование команды `Set Application Window 0` возвратит MapInfo Professional к ее исходному состоянию. Этот оператор переподчиняет окна диалогов. Для переподчинения документального окна, такого как окна Карты, используйте оператор **Оператор Set Next Document**.

**Внимание:** Если Вы задаете параметр *HWND* как шестнадцатеричное значение, то Вы должны использовать приставку &H с шестнадцатеричным числом. Иначе MapInfo попытается интерпретировать параметр как десятичное значение. (Это бывает, когда программа на Visual Basic создает командную строку, содержащую оператор **Оператор Set Application Window**.)

Для получения другой информации об интегрированной картографии смотрите 12 главу *Руководства пользователя MapBasic*.

**См. также:**

**Оператор Set Next Document**

---

## Оператор Set Area Units

### Назначение

Устанавливает единицы измерения площади для использования в операторах и функциях MapBasic по умолчанию.

### Синтаксис

**Set Area Units** *area\_name*

*area\_name* - строка, определяющая единицы измерения площади (к примеру, “акр”).

Описание

Оператор **Set Area Units** устанавливает единицы измерения площади. Установки единиц измерения площади используются в диалоге "SQL-запрос" в MapInfo. По умолчанию, MapBasic использует квадратные мили ("sq mi"), т. е. если в Вашей программе нет оператора **Set Area Units**, то единицами измерения площади будут квадратные мили. Параметр *area\_name* должен иметь строковое значение, список которых приведен в таблице:

| Название единицы измерения | Используемая единица измерения       |
|----------------------------|--------------------------------------|
| "acre"                     | акр                                  |
| "hectare"                  | гектар                               |
| "perch"                    | перч                                 |
| "rood"                     | руд                                  |
| "sq ch"                    | квадратные чейны                     |
| "sq cm"                    | квадратный сантиметр                 |
| "sq ft"                    | квадратный фут                       |
| "sq in"                    | квадратный дюйм                      |
| "sq km"                    | квадратный километр                  |
| "sq li"                    | квадратный линк                      |
| "sq m"                     | квадратный метр                      |
| "sq mi"                    | квадратная миля                      |
| "sq mm"                    | квадратный миллиметр                 |
| "sq rd"                    | квадратный род                       |
| "sq survey ft"             | квадратный топографический фут в США |
| "sq yd"                    | квадратный ярд                       |

Пример:

Set Area Units "acre"

См. также:

Функция **Area( )**, Оператор **Set Distance Units**

## Оператор Set Browse

### Назначение

Изменяет существующее окно списка.

### Синтаксис

#### Set Browse

```
[ Window window_id ]  
[ Grid { On | Off } ]  
[ Row row_num ]  
[ Column column_num ]
```

*window\_id* – идентификатор окна Списка, целое число;

*row\_num* – целое число типа SmallInt от одного и более, где 1 представляет первую строку таблицы;

*column\_num* – целое число типа SmallInt от нуля и более, где 0 представляет первую колонку таблицы.

### Описание

Оператор **Set Browse** управляет представлением окна Списка. Если параметр *window\_id* не задан, то действия оператора распространяются на самое верхнее из открытых окон Списка.

Предложения **Row** и **Column** позволяют назначить строку, которая будет видна первой, и колонку, которая будет самой левой в окне Списка.

Для того, чтобы изменить высоту, ширину и местоположение окна, используйте оператор **Оператор Set Window**.

### Пример:

```
Dim i_browser_id As Integer  
Open Table "world"  
Browse * From world  
i_browser_id = FrontWindow( )  
Set Browse Window i_browser_id Row 47
```

### См. также:

**Оператор Browse, Оператор Set Window**

## Оператор Set Cartographic Legend

### Назначение

Оператор Set Cartographic Legend позволяет Вам включать или выключать режим перерисовки, обновления, устанавливать книжную или альбомную ориентацию легенды, или изменять порядок следования разделов в картографической легенде, созданной оператором **Оператор Create Cartographic Legend**. (Для изменения размера, позиции или заголовка окна легенды, используйте: **Оператор Set Window**.)

### Синтаксис

```
Set Cartographic Legend
  [ Window legend_window_id ]
  Redraw { On | Off }
```

Or (оператор Или)

```
Set Cartographic Legend
  [ Window legend_window_id ]
  [ Refresh ]
  [ Portrait [ Columns number_of_columns ] |
    Landscape [Lines number_of_lines ] ]
  [ Align ]
  [ Style Size { Small | Large } ]
  [ Frame Order { frame_id, frame_id, frame_id, ... } ]
```

*legend\_window\_id* – это целочисленный идентификатор окна, который Вы можете получить при вызове функций **Функция FrontWindow( )** и **Функция WindowID( )**.

*frame\_id* – индекс ID-раздела легенд. Вы не можете использовать здесь имя слоя. Например, три раздела легенды могут иметь идентификаторы 1, 2 и 3.

*number\_of\_columns* - указывает ширину легенды.

*number\_of\_lines* - указывает высоту легенды.

### Описание

Оператор **Set Cartographic Legend** позволяет включать или выключать режим перерисовки, обновления, устанавливать книжную или альбомную ориентацию легенды, или изменять порядок следования разделов в картографической легенде, созданной оператором **Оператор Create Cartographic Legend**.

Если предложение **Window** не определено, MapInfo будет использовать самое верхнее окно легенды.

Нельзя использовать другие предложения, если используется **Redraw**.

Ключевое слово **Refresh** приводит к обновлению окна легенды. Таблицы, для которых будут обновляться разделы легенды, просканируются на предмет используемых в них стилей. Ключевые слова **Portrait** или **Landscape** определяют книжная или альбомная ориентация окна легенды будет использована.

**Align** - выравнивает стиль и текст в рамках независимо от того, в каком слое открыто окно легенды.

Предложение **Frame Order** изменяет порядок следования разделов в легенде.

### Пример:

Если вы использовали оператор **Оператор Create Cartographic Legend** для выбора легенды большого размера, следующий пример продемонстрирует обновление текущего окна легенды:

```
Set Cartographic Legend Window WindowID(0) Refresh Portrait Align Style  
Size Large
```

### См. также:

**Оператор Add Cartographic Frame, Оператор Alter Cartographic Frame, Оператор Create Cartographic Legend, Оператор Remove Cartographic Frame**

---

## Оператор Set Command Info

### Назначение

Помещает в память значения, которые могут быть прочитаны функцией **CommandInfo( ) функция** из другой процедуры.

### Синтаксис

```
Set Command Info attribute To new_value
```

*attribute* -- один из кодов, используемых функцией **CommandInfo( ) функция** (такой как CMD\_INFO\_ROWID);

*new\_value* -- новая величина, тип которой должен соответствовать значению кода *attribute* (например, если используется код CMD\_INFO\_ROWID, то параметр *new\_value* должен быть положительным целым числом типа Integer).

### Описание

Обычно функция **CommandInfo( )** возвращает значения, описывающие системные события. Оператор **Set Command Info** помещает значения в память, так что следующий за этим вызов функции **CommandInfo( )** возвращает заданные значения вместо системных.

### Пример:

Допустим, что Ваша программа имеет процедуру-обработчик **Процедура SelChangedHandler**. Из ее тела следующая функция определяет идентификатор строки, которая была выбрана или исключена из выбора:

```
CommandInfo( CMD_INFO_ROWID )
```

Когда MapInfo вызывает **Процедура SelChangedHandler** автоматически, то устанавливаются данные для чтения функцией . Теперь допустим, что для отладки Вы хотите вызвать процедуру оператором **Процедура SelChangedHandler**. Перед оператором **Оператор Call** должен быть оператор, посылающий значение для функции **CommandInfo( )** функция:

```
Set Command Info CMD_INFO_ROWID To 1
```

**См. также:**

**CommandInfo( )** функция, **Оператор Set Handler**

## Оператор Set Connection Geocode

### Назначение

Конфигурирует соединение с удаленной службой геокодирования. Соединение уже должно быть создано с использованием **Оператор Open Connection**.

### Синтаксис

```
Set Connection connection_number Geocode
[ Batch Size batch_size ]
[ ResultCode MarkMultiple [ On | Off ] ]
[ MixedCase [ On | Off ] ]
[ Match
  [ StreetName [ On | Off ] [,] ]
  [ StreetNumber [ On | Off ] [,] ]
  [ Municipality [ On | Off ] [,] ]
  [ CountrySubdivision [ On | Off ] [,] ]
  [ CountrySecondarySubdivision [ On | Off ] [,] ]
  [ PostalCode [ On | Off ] [,] ]
  [ MunicipalitySubdivision [ On | Off ] [,] ]
  [ All [ On | Off ] [,] ] ]
[ Fallback
  [ Geographic [ On | Off ] [,] ]
  [ PostalCode [ On | Off ] [,] ] ]
[ Dictionary
  [ All | Address | User | Prefer ( Address | User ) ]
[ Offset [ [ Center offset_num_expr Units distance_unit_name ]
  [ End offset_num_expr Units distance_unit_name ]
[ PassThrough name value, name value, ... ]
```

*connection\_number* - число указатель на соединение установленное используя **Оператор Open Connection**.

*batch\_size* - целое, указывающее максимальное число записей отправляемых на сервер за один раз.

*offset\_num\_expr* - численное выражение, указывающее сдвиг от концов или центра улицы. Эти значения просто сдвиг точки возвращаемой сервером от центра или концов улицы соответственно.

*distance\_unit\_name* - строка содержащая единицы в которых вычисляется *offset\_num\_expr*.

*name* - имя, одна из частей пары параметров передаваемых на сервер геокодирования.

*value* - значение, одна из частей пары параметров передаваемых на сервер геокодирования..

### Описание

Оператор **Set Connection** используется для установки предпочтительных параметров геокодирования, для того чтобы не задавать их каждый раз в запросе. Оператор **Set Connection** содержит шесть параметров: **Batch**, **Match**, **Fallback**, **Dictionary**, **Offset** и **PassThrough**.

### Параметр оператора Batch

Предложение **Batch** определяет максимальное количество записей передаваемых серверу за один раз. Это позволяет оптимизировать обработку записей таким образом, чтобы сохранить баланс времени затрачиваемого локальным компьютером и удалённой службой. Если это число велико, то потребуется больше времени на ожидание обработки данных на сервере. Если число мало, то пользователь имеет больше возможностей для отмены запроса. После того как запрос отправлен на сервер прервать его невозможно. Если прервать выполнение команды, оставшиеся запросы не обрабатываются

### Параметр оператора Match

Если параметр близкого соответствия **Match** установлен, то сервер геокодирования будет учитывать при обработке только передаваемые названия полностью совпадающие с названиями, по которым выполняется геокодирование. Например, если параметр **Match StreetName - On**, сервер геокодирования не будет учитывать названия улиц, которые не полностью соответствуют геокодируемому названию улицы, в качестве близкого варианта.

Для индивидуальных настроек, по умолчанию используется параметр **On** оператора **Match**. Если конкретный параметр упомянут, то это равносильно тому, что для него установлен режим **On**. Например, `Set Connection connectionHandle Geocode Match Municipality -` эквивалентно `Set Connection connectionHandle Geocode Match Municipality On`.

**Match StreetName** – определяет следует или нет опускать название улицы при поиске соответствия.

**Match StreetNumber** – определяет следует или нет опускать номер улицы при поиске соответствия.

**Match Municipality** – определяет следует или нет опускать муниципальное название при поиске соответствия.

**Match CountrySubdivision** – определяет следует или нет опускать название административного округа (обычно уровня области или провинции) при поиске соответствия.

**Match CountrySecondarySubdivision** – определяет следует или нет опускать название административного района при поиске соответствия.

**Match PostalCode** – определяет следует или нет опускать почтовый индекс при поиске соответствия.



**Match MunicipalitySubdivision** – определяет следует или нет опускать муниципального района при поиске соответствия.

**Match All** – включает (**On**) или отключает (**Off**) все параметры соответствия. Эта опция может использоваться в комбинации с другими опциями. Например, `Match All On, Match PostalCode Off` включает поиск всех параметров соответствия и только почтовый индекс выключен.

### Параметр оператора FallBack

**Fallback Geographic** - указывает использовать ли геокодирование введённого адреса по географическому центроиду, если геокодирование на уровне улиц не может быть осуществлено. Это значение имеет смысл, если введённый вами адрес содержит улицу. Если запись не содержит адреса улицы, это значение игнорируется.

**Fallback Postal** - указывает использовать ли геокодирование введённого адреса используя центроид почтовой зоны, если геокодирование на уровне улиц не может быть осуществлено.

### Предложение Dictionary

**Dictionary** - указывает комбинацию использования словарей, адресного словаря MapMarker и пользовательского словаря, используемых в процессе геокодирования. Существует пять возможных вариантов для предложения **Dictionary**:

- **Dictionary All** - использовать оба словаря и пользовательский, и адресный.
- **Dictionary Address** - использовать только адресный словарь.
- **Dictionary User** - использовать только пользовательский словарь.
- **Dictionary Prefer Address** - использовать оба словаря с приоритетом адресного.
- **Dictionary Prefer Address** - использовать оба словаря с приоритетом пользовательского.

### Предложение Offset

**Offset End** - указывает расстояние на которое точка сдвигается от перекрёстка.

**Offset End** - указывает расстояние на которое точка сдвигается от центральной линии улицы.

**Units** - строка описывающая единицы в которых исчисляется **Offset Center** и **Offset End**. Список доступных единиц измерения смотрите в **Оператор Set Distance Units на стр. 103**.

### Параметр оператора PassThrough

**PassThrough** - это набор пар имя/значение пересылаемых на сервер геокодирования. Эти пары специфичны для серверов геокодирования и документируются для каждого отдельного сервера.

### Пример:

Следующий пример устанавливает соединение с сервером геокодирования, с некоторыми опциями поиска соответствия.

```
set connection MapMarkerHandle1 geocode match streetname, streetnumber,
municipality, municipalitysubdivision, postalcode, countrysubdivision
```

Следующий пример добавляет предложение **PassThrough** к оператору **Set Connection**. В этом примере отбираются только сертифицированные CASS-результаты запроса для данных по США.

```
PassThrough "KEY_CASS_RULES" "true"
```

**См. также:**

[Оператор Geocode](#), [Оператор Open Connection](#)

---

## Оператор Set Connection Isogram

### Назначение

Позволяет пользователю установить соединение с опцией посторения зон транспортной доступности.

### Синтаксис

```
Set Connection connection_handle Isogram
[ Banding [ On | Off ] ]
[ MajorRoadsOnly [ On | Off ] ]
[ MaxOffRoadDistance distance_value Units distance_units ]
[ ReturnHoles [ On | Off ] ] [ MajorPolygonOnly [ On | Off ] ]
[ SimplificationFactor simplification ]
[ PointsOnly [ On | Off ] ]
[ DefaultAmbientSpeed ambient_speed
  Units distance_units Per time_units ]
[ DefaultPropagationFactor propagation_factor ]
[ Batch Size batch_size ]
```

*connection\_handle* – целое, определяющее число соединений, которое возвращает [Оператор Open Connection](#).

*distance\_value* - вещественное значение указывающее максимальное расстояние передвижения по просёлочным дорогам сети.

*distance\_units* – строка содержащая единицы в которых вычисляется *distance\_value*. Полный список допустимых единиц измерения расстояний приводится в разделе [Оператор Set Distance Units на стр. 103](#).

*simplification* - вещественное, контролирует число узлов в результирующем полигоне, выраженное через процент. Допустимые значения от 0 до 1 включительно.

*ambient\_speed* – численное значение, определяющее рекомендуемую по умолчанию скорость движения. Число используемое в параметрах *distance\_units* и *time\_units*.

*time\_units* – строка указывающая единицы времени. Допустимые значения "hr", "min" и "sec".

*propagation\_factor* - вещественное значение, указывающее значение по умолчанию для параметра движения. Допустимые значения от 0 до 1 включительно.

*batch\_size* - целое выражение указывающее размер каждого запроса, передаваемого на сервер. По умолчанию принимается значение 2, максимальное - 50.

## Описание

Оператор **Set Connection Isogram** конфигурирует соединение используемое для создания зон транспортной доступности (используя **Оператор Create Object**).

**Banding** - используется только когда создаётся несколько зон доступности по времени или расстоянию в операции построения зон транспортной доступности. Если это параметр **On**, зоны создаваемые для движения из одной точки будут неперекрывающимися. Маленькие территории вырезаются из результата. Таким образом, представляется время или расстояние от меньшего региона до его границы. Например, если создаются зоны транспортной доступности в 10, 20, и 30 минут, 20 минутная изолиния будет очерчивать территорию доступную в интервале времени от 10 до 20 минут, а 30 минутная изолиния территорию доступную в течении от 20 до 30 минут движения. Если задан параметр **Off**, то будут все зоны будут покрывать площадь доступную от 0 до заданного значения времени или расстояния.

**MajorRoadsOnly** - использовать ли при создании зон транспортной доступности только основные дороги. При использовании **MajorRoadsOnly** изолинии создаются значительно быстрее.

**MaxOffRoadDistance** - указывает максимально допустимое расстояние для движения по просёлкам.

**ReturnHoles** – определяет наличие или отсутствие брешей в полученных результатах.

**MajorPolygonOnly** – определяет, что полученная зона будет иметь только одну внешнюю границу.

**SimplificationFactor** – определяет параметр упрощения формы полигона. Параметр упрощения выражается в процентах от количества исходных узлов той части, по которым будет построен полигон-результат. Полигон или группа точек может содержать много точек-узлов. Параметр упрощения, это вещественное число между 0.01 и 1.0 (1 - 100% и 0.01 - 1%). Меньшее число, означает меньшее количество точек при создании зоны, а следовательно более быструю передачу данных через Интернет. Значение по умолчанию - 0.05

**PointsOnly** определяет, как обрабатывать не точечные объекты – пропускать или нет.

**DefaultAmbientSpeed** - используется только когда указано время. Пример синтаксиса:

```
DefaultAmbientSpeed 12 "mi" Per "hr"
```

**DefaultPropagationFactor** определяет процентное отношение проселочных дорог в оценке остатка (расстояний) допустимое при вычислении полос дальностей. Дороги не включенные в сеть дорог могут быть в том числе и проездами или подъездными путями. Параметр движения – процентное отношение оценки при вычислении дальности от начальной точки до заданной на максимальном расстоянии. **DefaultPropagationFactor** - используется только для Расстояния.

По умолчанию этот параметр имеет значение 0.16.

Допустимый диапазон значений от 0.01 до 1.

**Batch Size** - число записей передаваемых серверу на обработку за один раз. Этот параметр имеет существенное влияние на скорость обработки запроса. Чем больше отправляемы запрос, тем больше времени потребуется на создание изолиний и получение их и тем больше времени потребуется на то чтобы прервать запрос в Mapinfo Professional. Малое значение улучшает отклик на команду и уменьшает вероятность превышения времени ожидания. По умолчанию её значение равно 2.

**Пример:**

Следующий пример показывает использование оператора Set Connection Isogram.

```
Set Connection iConnect IsogramBanding On MajorRoadsOnly On
MaxOffRoadDistance 2 Units "mi" ReturnHoles On MajorPolygonOnly On
SimplificationFactor .05DefaultAmbientSpeed 50 Units "mi" Per "hr"
DefaultPropagationFactor .2Batch Size 2 Point On
```

**См. также:**

[Оператор Create Object](#), [Оператор Open Connection](#)

---

## Оператор Set CoordSys

**Назначение**

Назначает координатную систему, в дальнейшем используемую прикладной программой MapBasic.

**Синтаксис**

**Set CoordSys...**

**CoordSys...** — слово, с которого начинается стандартное предложение оператора для определения координатной системы.

**Описание**

Оператор **Set CoordSys** устанавливает координатную систему, которая в дальнейшем будет использоваться прикладной программой MapBasic. По умолчанию, MapBasic использует долготу и широту. Данная установка влияет на возвращаемые значения географическими функциями, такими как [Функция CentroidX\( \)](#) и [Функция ObjectNodeX\( \)](#). MapBasic-программа может выполнить собственный оператор **Set CoordSys**, вследствие чего значения, возвращаемые географическими функциями, будут автоматически отражать новую координатную систему. При этом координатная система в MapInfo может оставаться прежней.

Оператор **Set CoordSys** не действует на окно Карты. Чтобы задать проекцию или систему координат окна Карты, следует использовать оператор Set Map...CoordSys.

В параметре CoordSys может использоваться предложения **Table** или **Window** для установки системы координат, такой же, как в таблице, или такой же, как в окне. Более подробную информацию см. в разделе [Оператор CoordSys on page 175](#).

**Пример:**

Установим непроецированную координатную систему Земли:

```
Set CoordSys Earth
```

Следующий оператор **Set CoordSys** устанавливает в качестве координатной системы равноплощадную проекцию Алберса.

```
Set CoordSys Earth  
Projection 9,7,"m",-96.0,23.0,20.0, 60.0, 0.0, 0.0
```

В окне Отчета координатная система определяется оператором **Set CoordSys**. Перед работой с объектами Отчета Вы должны определить координатную систему на странице.

```
Set CoordSys Layout Units "in"
```

**Внимание:** Выполнив оператор **Set CoordSys Layout**, MapBasic продолжает использовать координатную систему Отчета до тех пор, пока Вы не переопределите ее. Аналогично, если Вы хотите работать с объектами на проецированной Карте, выполните оператор **Set CoordSys Earth**.

**См. также:**

[Оператор CoordSys](#), [Оператор Set Area Units](#), [Оператор Set Distance Units](#), [Оператор Set Paper Units](#)

---

## Оператор Set Date Window

### Назначение

Отображает окно даты, которое переводит двузначный год в четырехзначный. В нем можно поменять стандартную настройку.

### Синтаксис

```
Set Date Window { nYear | Off }
```

*nYear* - это короткое целое, от 0 до 99, которое определяет число, значения превышающие это число будут отнесены к 20-му столетию (19xx), а числа меньшие заданного будут отнесены к 21-му столетию (20xx). (Например, если *nYear* равно 70, двузначный год от 70-ти и выше будет соответствовать годам с 1970 по 1999, а двузначный год от 69-ти и ниже - с 2000 по 2069.)

**Off** - отключает окно даты. Двузначные годы будут конвертироваться в текущее столетие (по системному календарю и времени).

### Описание

Запуск команды Set Date Window из окна MapBasic изменит поведение дат, но не изменит системные настройки даты в MIPro. При запуске команды **Set Date Window** из программы MapBasic, поведение дат изменится только локально, настройки дат в системе не изменятся.

Команда **Set Date Window**, запущенная в окне MapBasic любого рабочего набора, включая Startup.WOR, или любого картографического приложения, выполняющего эту команду посредством интерфейса приложения MapInfo, изменит настройки текущего сеанса.

Когда команда **Set Date Window** запускается из программы MapBasic (так же как **Оператор Run Command**), новые настройки распространяются только на локальное содержимое программы. Настройки текущего сеанса и системные настройки не будут изменены. Локальный контекст программы инициализируется на основе настроек сеанса. Нумерация и датирование работает аналогично. Они действуют для программы, если программы запущена, в противном случае, они изменяют общие настройки.

MBX, скомпилированные до версии 5.5 будут конвертировать двузначный год в текущее столетие (поведение версии 5.0 и более ранних). Что бы получить новое поведение отнесения двузначных данных к разным столетиям, перекомпилируйте программу в MapBasic v5.5.

Введите число от 0 до 99. Выбранное число указывает какой префикс будет использоваться (19 или 20).

Например, если выбрано число 50, оператор укажет:

Года от 00 до 49 станут 2000-2049.

Года от 50 до 99 станут 1950-1999.

### Пример:

В следующем примере переменная Date1 = 19890120, Date2 = 20101203 и MyYear = 1990.

```
DIM Date1, Date2 as DateDIM MyYear As IntegerSet Format Date "US"Set Date Window 75Date1 = StringToDate("20.01.89")Date2 = StringToDate("03.12.10")MyYear = Year("30.12.90")
```

### См. также:

[Функция DateWindow\( \)](#)

## Оператор Set Datum Transform Version

### Назначение

С помощью этого оператора можно выбирать старый или новый вариант преобразования топоцентрической системы координат (датума).

### Синтаксис

Set Datum Transform Version *version\_number*

*version\_number* – целое число. Если *version\_number* равно или больше 800, будут использоваться новые алгоритмы преобразования датумов. Иначе, используются старые алгоритмы.

### Описание

По умолчанию, MapInfo использует новые алгоритмы преобразования топоцентрических систем координат (датумов). Эти алгоритмы лучше согласуются с используемыми в других программах и не следует отказываться от их применения. В предыдущих версиях MapInfo Professional использовались алгоритмы, оптимизированные по скорости выполнения, и результаты слегка отличались от полученных в других программах.

## Оператор Set Digitizer

### Назначение

Назначает координаты для оцифровки изображения с бумажной карты. А также включает и выключает режим дигитайзера.

### Синтаксис 1

```
Set Digitizer  
  ( mapx1, mapy1 ) ( tabletx1, tablety1 ) [ Label name ] ,  
  ( mapx2, mapy2 ) ( tabletx2, tablety2 ) [ Label name ]  
  [, ... ]  
CoordSys...  
  [ Units... ]  
  [ Width tabletwidth ]  
  [ Height tabletheight ]  
  [ Resolution xresolution, yresolution ]  
  [ Button click_button_num, double_click_button_num ]  
  [ Mode { On | Off } ]
```

### Синтаксис 2

```
Set Digitizer Mode { On | Off }
```

*mapx* – расположение относительно Запада и Востока на бумажной карте;

*mapy* – расположение относительно Севера и Юга на бумажной карте;

*tablet**x* – X-координата на планшете, соответствующая *mapx*;

*tablety* – Y-координата на планшете, соответствующая *mapy*;

*name* – имя контрольной точки;

Предложение **Оператор CoordSys** определяет систему координат, используемую на бумажной карте.

*click\_button\_num* – номер кнопки, которая симулирует один щелчок мыши;

*double\_click\_button\_num* – номер кнопки, которая симулирует двойной щелчок мыши.

### Описание

Оператор **Set Digitizer** используется для настройки планшета дигитайзера. Параметры оператора Set Digitizer соответствуют режимам и данным, которые пользователь MapInfo может задать при помощи диалогового окна команды Карта > Настройка дигитайзера. Все измерения проводятся в заданных пользователем единицах измерения бумажной карты. Оператор **Set Digitizer** не настраивает другие (системные) режимы работы с дигитайзером, такие как порты и скорость связи. Их пользователь должен настроить вне MapBasic и MapInfo.

Оператор **Set Digitizer** позволяет задать MapInfo координатную систему, используемую на бумажной карте, а также две или более контрольных точек. Каждая контрольная точка определяет соответствие координатной пары карты (долгота и широта) координатной паре на



планшете. Координатная пара на планшете указывает точку на планшете, соответствующую выбранным координатам карты. Координаты на планшете представляются в единицах измерения дигитайзера, измеряемых от левого верхнего угла планшета.

Параметр `CoordSys` определяет систему координат, используемую на бумажной карте. Подробнее см.: "[Оператор CoordSys on page 175](#)".

**Внимание:** Оператор `Set Digitizer` игнорирует предложение `Bounds` параметра `CoordSys`.

Предложения `Width`, `Height`, и `Resolution` предназначены только внутреннего использования в MapInfo Professional. MapInfo может сохранять настройку дигитайзера для следующих сеансов работы в файле Рабочего Набора. Программы MapBasic не требуют указывать эти предложения.

### Режим оцифровки

Если дигитайзер настроен, пользователь может включать и выключать режим оцифровки, нажимая на клавишу "D". Этот режим может включать и программа MapBasic оператором

```
Set Digitizer Mode On
```

Or (оператор Или)

```
Set Digitizer Mode Off
```

Для определения включения режима оцифровки используется функция `SystemInfo(SYS_INFO_DIG_MODE)`, которая возвращает "Да" (TRUE), если режим установлен.

При включенном режиме оцифровки в активном окне Карты помимо указателя мышки появляется также курсор дигитайзера в виде большого креста.

Если отключен режим оцифровки или окно Карты неактивно, курсор дигитайзера не показывается и панель дигитайзера начинает работать как мышь (если только Ваш дигитайзер поддерживает режим эмуляции мыши).

**См. также:**

[Оператор CoordSys](#), [Функция SystemInfo\( \)](#)

---

## Оператор Set Distance Units

### Назначение

Устанавливает единицы измерения расстояний, используемые в географических операциях, таких как [Оператор Create Object](#).

### Синтаксис

```
Set Distance Units unit_name
```

*unit\_name* - наименование единиц расстояния (например, "m" для метров).

Описание

Оператор **Set Distance Units** устанавливает единицы линейных измерений. По умолчанию MapBasic использует мили ("mi"), то есть, если в Вашей программе нет оператора **Set Distance Units**, единицами измерения расстояния будут мили. В следующих операторах и функциях будет использоваться установленная единица измерения расстояний, если единицы измерения специально не определяются в самих операторах или функциях. Например, оператор **Create Object** с параметром **Width** может либо задавать единицы измерения расстояний, либо нет. Если в параметре **Width** не указаны единицы расстояний, оператор использует текущую установку (мили или любые другие, используемые последним оператором **Set Distance Units**).

Параметр *unit\_name* должен являться одним из кодов, показанных в следующей таблице.

| Значение<br>unit_name | Используемая единица измерения   |
|-----------------------|--|
| "ch"                  | чейны  |
| "cm"                  | сантиметры   |
| "ft"                  | футы (иногда называется "международным футом"; один "международный фут" равен точно 30,48 см)  |
| "in"                  | дюймы  |
| "km"                  | километры  |
| "li"                  | линки  |
| "m"                   | метры  |
| "mi"                  | мили   |
| "mm"                  | миллиметры   |
| "nmi"                 | морские мили (1 морская миля равна 1852 метрам)  |
| "rd"                  | роды   |
| "survey ft"           | топографический фут в США (использовался при обмере территории США в 1927; один геодезический фут равен точно 12/39.37 метрам или, приблизительно, 30,48006 сантиметрам) |
| "yd"                  | ярды   |

Пример:

```
Set Distance Units "km"
```

См. также:

Функция `Distance()`, Функция `ObjectLen()`, Оператор `Set Area Units`, Оператор `Set Paper Units`

---

## Оператор `Set Drag Threshold`

### Назначение

Назначает временную задержку, необходимую для фиксации мыши на перемещаемом объекте.

### Синтаксис

`Set Drag Threshold pause`

*pause* – вещественное число, задающее задержку в секундах (по умолчанию – 1.0).

### Описание

Когда пользователь указывает на объект, оставляя клавишу мышки нажатой, MapInfo требует выдержать небольшую паузу. Эта задержка предотвращает случайное перемещение объекта. Оператор `Set Drag Threshold` назначает продолжительность этой задержки.

### Пример:

```
Set Drag Threshold 0.25
```

---

## Оператор `Set Event Processing`

### Назначение

Позволяет временно отключить реакцию на системные события и избежать лишних перерисовок экрана.

### Синтаксис

`Set Event Processing { On | Off }`

### Описание

Оператор `Set Event Processing` позволяет временно отключать реакцию на системные события и тем самым избежать ненужной перерисовки содержимого экрана. Тот же оператор позволяет затем снова включить реакцию на системные события.

Несколько последовательных команд могут изменять окно, что сопровождается перерисовкой его содержимого. Такая множественная перерисовка окна нежелательна, поскольку вынуждает пользователя тратить время на ожидание. Чтобы сэкономить время, пользователь может отключить перерисовку окон оператором

```
Set Event Processing Off
```

а после того, как все операторы, изменяющие окно (например, **Оператор Set Map**) отработают, включить перерисовку снова:

```
Set Event Processing On
```

Каждый оператор **Set Event Processing Off** должен иметь парный **Set Event Processing On**. Если, работая в многозадачных системах (например, Windows или System 7), Вы забудете включить обработку событий обратно, это может повлиять на работу других программ.

Вы также можете управлять перерисовкой изображения в окне оператором Set Map...Redraw Off, действие которого похоже на действие оператора **Set Event Processing Off**. Однако оператор **Оператор Set Map** управляет перерисовкой одного окна Карты, а действие оператора **Set Event Processing** распространяется на все окна MapInfo.

---

## Оператор Set File Timeout

### Назначение

Предписывает MapInfo повторять попытку доступа к файлу после сетевого конфликта.

### Синтаксис

```
Set File Timeout n
```

*n* – число от нуля и больше, задающее ожидание в секундах.

### Описание

Обычно, если операция не может быть продолжена из-за конфликта в сети, MapInfo показывает диалог типа “Повторить/Отменить”. Если программа MapBasic выполнит оператор **Set File Timeout**, то MapInfo вместо вывода диалога будет автоматически повторять попытки открыть файл, доступ к которому в сети запрещен.

Если число *n* больше нуля, то через каждые *n* секунд MapInfo делает очередную попытку открыть файл. В любой момент, когда пользователь попытается прочитать таблицу, доступа к которой нет (например, таблица сохраняется другим пользователем), MapInfo Professional повторяет попытки получить доступ к таблице. Если через *n* секунд таблица все еще не доступна, MapInfo показывает диалог с соответствующим сообщением. Этот диалог нельзя перехватить и обработать средствами обработки ошибок MapBasic.

Если *n*=0, MapInfo демонстрирует диалог немедленно, как только обнаруживает, что таблица недоступна.

Не используйте одновременно оператор **Set File Timeout** и обработчик ошибок OnError. Там, где обработка ошибок разрешена, значение задержки должно быть равно нулю.

Когда значение задержки обращения в файлу не равно нулю, обработка ошибок должна быть заблокирована. Для получения другой информации об интегрированной Более подробно возможные конфликты в сети описаны в *Руководстве пользователя MapBasic*.

### Пример:

```
Set File Timeout оператор 100
```

## Оператор Set Format

### Назначение

Задаёт, как MapBasic составляет строки из численных значений и значений даты и времени.

### Синтаксис 1

```
Set Format Date { "US" | "Local" }
```

### Синтаксис 2

```
Set Format Number { "9,999.9" | "Local" }
```

### Описание

Пользователь может установить разные форматы для даты и чисел в своём компьютере. Например, пользователь Windows может изменить формат даты, используя панель инструментов Windows.

Некоторые функции MapBasic, такие как **Функция Str\$( )**, используют эти системные установки. Другими словами, некоторые функции могут иметь разные результаты, так как выполнялись в компьютерах с разными системными конфигурациями.

Оператор **Set Format** заставляет программу MapBasic игнорировать внешние установки так, чтобы функции (такие как **Функция Str\$( )**) вела себя предсказуемо.

| Оператор                    | Эффект в программе MapBasic   |
|-----------------------------|---|
| Set Format Date "US"        | MapBasic использует форму Месяц/День/Год для представления даты вместо внешней установки компьютера пользователя.   |
| Set Format Date "Local"     | MapBasic использует форму для представления даты согласно установке компьютера пользователя.  |
| Set Format оператор 9,999.9 | Функция <b>Функция Format\$( )</b> использует установку для форматирования числа, принятую в США (десятичной точкой является точка, а разделителем тысяч – запятая) вместо внешней установки компьютера пользователя. |
| Set Format Number "Local"   | Функция <b>Функция Format\$( )</b> использует форму для представления чисел согласно установке компьютера пользователя.   |

Первый вариант синтаксиса (**Set Format Date**) имеет эффект для следующих случаев: вызов функции **Функция StringToDate( )**; использование даты в функции **Функция Str\$( )**; выполнение операций в MapBasic, ведущих к автоматическому конвертированию строковых значений и значений дат (например, применение оператора **Оператор Print** для печати даты или при присваивании переменной строкового типа величины типа Date).

Второй вариант синтаксиса (**Set Format Number**) влияет на функции **Функция Format\$( )** и **Функция FormatNumber\$( )**. Программы, откомпилированные MapBasic версии 3.0 и более ранней, по умолчанию используют стандарт США. Программы, откомпилированные в MapBasic версии 4.0 и в более поздних версиях по умолчанию используют установку "Local".

Для определения установленного режима форматирования, используется функция **Функция SystemInfo( )**. Установка оператора **Set Format**, сделанная в одной программе MapBasic, не влияют на другие выполняющие программы.

### Пример:

Пусть переменная типа даты *date\_var* содержит "June 11, 1995". Функция:

```
Str$( date_var )
```

может быть равно и "11.06.95", и "95/11/06", в зависимости от того, какой формат использует пользователь в своем компьютере. Если Вы вставите в текст Вашей программы оператор **Set Format Date "US"** перед местом, где используется функция **Функция Str\$( )**, вы предпишете функции **Функция Str\$( )** использовать формат вида (Месяц/День/Год). Результатом будет "11/06/95".

### См. также:

**Функция Format\$( )**, **Функция FormatNumber\$( )**, **Функция Str\$( )**, **Функция StringToDate( )**, **Функция SystemInfo( )**

---

## Оператор Set Graph

### Назначение

Изменяет существующее окно Графика.

### Синтаксис для графиков, построенных по правилам версии 5.5:

```
Set Graph
[ Window window_id ]
[ Title title_text ]
[ SubTitle subtitle_text ]
[ Footnote footnote_text ]
[ TitleSeries titleseries_text ]
[ TitleGroup titlegroup_text ]
[ TitleAxisY1 titleaxisy1_text ]
[ TitleAxisY2 titleaxisy2_text ]
```

*window\_id* – идентификатор окна Графика, целое число;

*title\_text* - заголовок, появляющийся в верху окна Графика.

*subtitle\_text* - текст подзаголовка графика.

*footnote* - текст сноски графика.

*titleseries\_text* - текст заголовка серий графика.

*titlegroup\_text* - текст заголовка групп графика.

*titleaxisY1\_text* - текст заголовка оси Y.

*titleaxisY2* - текст заголовка для оси Y2.

## Синтаксис (графики, построенные по правилам до версии 5.5)

## Set Graph

```

[ Window window_id ]
[ Type { Area | Bar | Line | Pie | XY } ]
[ Stacked { On | Off } ]
[ Overlapped { On | Off } ]
[ Droplines { On | Off } ]
[ Rotated { On | Off } ]
[ Show3d { On | Off } ]
[ Overlap overlap_percent ]
[ Gutter gutter_percent ]
[ Angle angle ]
[ Title graph_title [ Font... ] ]
[ Series series_num
  [ Pen... ]
  [ Brush... ]
  [ Line... ]
  [ Symbol... ]
  [ Title series_title ] ]
[ Wedge wedge_num
  [ Pen... ]
  [ Brush... ] ] ]
[ { Label | Value } Axis
  [ { Major | Minor } Tick { Cross | Inside | None | Outside } ]
  [ { Major | Minor } Grid { On | Off } Pen... ]
  [ Labels { None | At Axis } [ Font... ] ]
  [ Min { min_value | Auto } ]
  [ Max { max_value | Auto } ]
  [ Cross { cross_value | Auto } ]
  [ { Major | Minor } Unit { unit_value | Auto } ]
  [ Pen... ]
  [ Title axis_title [ Font... ] ] ]
[ Legend
  [ Title legend_title [ Font... ] ]
  [ Subtitle legend_subtitle [ Font... ] ]
  [ Range [ Font... ] ] ]

```

*window\_id* – идентификатор окна Графика, целое число;

*overlap\_percent* – целое число от 0 до 100, задающее процент перекрывания двух соседних столбцов;

*gutter\_percent* – целое число от 0 до 100, задающее расстояние между столбцами в процентах;

*angle* – целое число от 0 до 360, задающее стартовый угол для круговой диаграммы;

*graph\_title* – строка с текстом заголовка графика;

*axis\_title* – строка с текстом заголовка одной из осей графика;

*min\_value* – минимальная величина, показанная на оси графика;

*max\_value* – максимальная величина, показанная на оси графика;



*cross\_value* – точка пересечения осей;

*unit\_value* – единица измерения для делений на одной из осей;

*series\_num* – номер серии данных в графике, которая подвергается изменениям (например, 2, 3, ...);

*series\_title* – имя серии для отображения его в легенде с образцами линии и штриха;

*legend\_title* и *legend\_subtitle* – строки с заголовком и подзаголовком легенды графика.

Предложение **Line** определяет стиль линии для линейного графика.

Предложение **Brush** определяет стиль штриховки. Предложение **Pen** определяет стиль линии границы заштрихованной области.

Предложение **Symbol** определяет стиль символа.

Параметром этого оператора **Font** можно задать стиль оформления текста.

## Описание

Оператор **Set Graph** изменяет вид графика в уже открытом окне График. Если идентификатор окна не указан в операторе (параметр *window\_id*), то оператор будет работать с окном Графика, которое располагается выше остальных открытых окон Графиков. Этот оператор позволяет программе управлять графиком и легендой так же, как это может делать пользователь при помощи команд из меню График в окне MapInfo.

Оператор **Set Graph** может использоваться в файле Рабочего Набора. Для примера Вы можете открыть окно Графика и сохранить Рабочий Набор (например, под именем GRAPHNER.WOR). Теперь откройте файл Рабочего Набора в любом текстовом редакторе и увидите оператор **Set Graph**, задающий те настройки, которые были у открытого ранее окна Графика. Для изменения размеров окна Графика и расположения его на экране используйте оператор **Оператор Set Window**.

Команды Graph в рабочих наборах или программах, созданных в версиях ранее 5.5, будут генерировать окна графиков версии 5.0. Когда окно графика версии 5.0 активно в сеансе MapInfo 5.5, то появится меню графика 5.0, так что можно его редактировать в диалогах версии 5.0. Мастер графиков всегда генерирует окно графика версии 5.5.

## Пример:

**графики, построенные по правилам версии 5.5, и более поздние:**

```
include 'mapbasic.def'
graph_id = WindowId(4) ' window code for a graph is 4
Set Graph
    Window graph_id
    Title "United States"
    SubTitle "1990 Population"
    Footnote "Values from 1990 Census"
    TitleGroup "States"
    TitleAxisY1 "Population"
```

графики, построенные по правилам до версии 5.5:

Этот пример иллюстрирует использование оператора **Set Graph**, а также настройки элементов окна Легенды. Следующая за приведенными ниже операторами настроек команда **Оператор Graph** может открыть окно Графика для данных из двух колонок (orders\_rcvd и orders\_shipped) из таблицы SELECTION (окно может быть открыто оператором Graph).

Оператор **Оператор Graph** фактически определяет три колонки, первая из которых ("sales\_rep") используется для образования надписей у оси.

```
Open Window LegendSet Window Legend Position (3.0, 1.6) Width 3.3 Height
0.750000Graph sales_rep,orders_rcvd,orders_shippedFrom selectionPosition
(0.2, 0.1) Width 4.5 Height 3.9'' Первый оператор Set Graph задает тип'
графика и главный заголовок графика'Set GraphType Bar Stacked Off
Overlapped Off Droplines Off Rotated Off Show3d OffOverlap 30 Gutter 10
Angle 0Title "График выполнения заказов" Font ("Helv",1,18,0)'' Второй
Set Graph задает все атрибуты' оси X '.
```

```
,
Set Graph Label AxisMajor Tick Outside Major Grid Off Pen (1,2,117440512)
Minor Tick None Minor Grid Off Pen (1,2,117440512)Min 1.0 Max 5.0Cross
1.0 Major unit 1.0 Minor unit 0.5 Labels At Axis Font ("Helv",0,8,0)Pen
(1,2,117440512) Title "Торговый представитель" Font ("Helv",0,8,0)''
надпись "Торговый представитель"'' появляется у оси X''' Следующий
оператор Set Graph задает все атрибуты'
оси Y (оси значений)Major Tick Outside Major Grid Off Pen (1,2,117440512)
Minor Tick None Minor Grid Off Pen (1,2,117440512) Min 0.0 Max 300000.0
Cross 0.0 Major unit 50000.0 minor unit 25000.0 Labels At Axis Font
("Helv",0,8,0)Pen (1,2,117440512)Title "Сумма заказов" Font
("Helv",0,8,0)'' надпись "Сумма заказов"'' появляется у оси Y''' Далее
настраивается стиль оформления 'для второй серии данных. This dictates
what color bars will
```

```
' appear to represent the orders_rcvd column data.
' Also controls what description will appear in the
' legend
',
' Since this is a bar graph, the Brush is the style
' of prime importance; if this was a line graph,
' the Line and Symbol clauses would be important).
',
```

```
Set Graph Series 2
Brush (8,255,16777215)
Line (1,2,0,255) Symbol (32,255,12)
Title "Orders Received ($)"
',
'the above title will appear in the legend...
```

```
',
' The next set graph customizes the styles
' used by series 3 (orders_shipped).
',
```

```
Set Graph Series 3
Brush (2,12632256,201326591)
Line (1,2,0,0) Symbol (34,12632256,12)
Title "Orders Shipped ($)"
',
'the above title will appear in the legend...
```

```
',
' the last Set Graph statement dictates what
' Grapher-related title and subtitle will appear
```

```
' in the Legend window, as well as what fonts will  
' be used in the legend.  
,  
  
Set Graph Legend  
    Title "Orders Received vs. "Font ("Helv",0,10,0) 'шрифт для заголовка  
    Subtitle "(торговыми представителями)"Font ("Helv",0,8,0) 'шрифт для  
    подзаголовка'шрифт для элементов легендыRange font ("Helv",2,8,0)
```

**См. также:**

**Оператор Graph, Оператор Set Window**

---

## Оператор Set Handler

### Назначение

Включает и выключает вызов обработчика системных событий, вызов такой процедуры как **Процедура SelChangedHandler**.

### Предупреждение

Этот оператор нельзя запускать из окна MapBasic.

### Синтаксис

```
Set Handler handler_name { On | Off }
```

*handler\_name* – имя процедуры обработчика, такой как **Процедура SelChangedHandler**.

### Описание

Обычно, если Ваша программа имеет в своем теле процедуры-обработчики, MapInfo вызывает их автоматически, как только происходит соответствующее событие. Например, если Ваша программа содержит процедуру **Процедура SelChangedHandler**, MapInfo Professional вызовет ее автоматически, как только произойдет изменение в выборе.

Оператор **Set Handler** может отключить автоматический вызов определенного обработчика в Вашей программе MapBasic и включить его снова.

Оператор **Set Handler ... Off** имеет эффект только для автоматических вызовов, на вызовы процедуры оператором **Оператор Callon** не влияет.

### Пример:

Следующий пример показывает, как обойти лишние запуски обработчика изменения выбора при помощи оператора **Set Handler**.

```
Sub SelChangedHandler Set Handler SelChangedHandler Off ' На этом месте  
можно применить в цикле оператор Select ' не обрабатывая при этом  
изменения выбора.
```

```
Set Handler SelChangedHandler On End Sub
```

См. также:

Процедура `SelfChangedHandler`, Процедура `ToolHandler`

## Оператор Set Layout

### Назначение

Изменяет существующее окно Отчета.

### Синтаксис

```
Set Layout
[ Window window_id ]
[ Center ( center_x, center_y ) ]
[ Extents { To Fit | ( pages_across, pages_down ) } ]
[ Pagebreaks { On | Off } ]
[ Frame Contents { Active | On | Off } ]
[ Ruler { On | Off } ]
[ Zoom { To Fit | zoom_percent } ]
```

*window\_id* – идентификатор окна Отчета, целое число;

*center\_x* – горизонтальная координата центра окна Отчета;

*center\_y* – вертикальная координата центра окна Отчета;

*pages\_across* – число страниц (одна или более), расположенных подряд по горизонтали;

*pages\_down* – число страниц (одна или более), расположенных подряд по вертикали;

*zoom\_percent* – процентное соотношение размера отображения Отчета к реальному размеру.

### Описание

Оператор **Set Layout** управляет настройкой отображения содержимого окна Отчета. Если параметр *window\_id* не задан, то действия оператора распространяются на самое верхнее из открытых окон Отчета. Этот оператор позволяет программе управлять отображением данных в окне Отчета так же, как это делает пользователь при помощи команд из меню Отчет в окне MapInfo.

Предложение **Center** задает точку, которая будет центральной в окне Отчета.

Предложение **Extents** управляет количеством страниц Отчета. Следующий вариант оператора:

```
Set Layout Extents To Fit
```

определяет, на скольких страницах установленного формата для принтера, который на данный момент подключен к системе, смогут поместиться все существующие объекты Отчета. После слова **Extents** можно также назначать и определенное количество печатных страниц. Например, следующий оператор задает шесть страниц, по три в два ряда:

```
Set Layout Extents (3, 2)
```

Если Отчет состоит более чем из одной страницы, то переключатель **Pagebreaks** позволяет Вам управлять отображением линий, разделяющих изображение на страницы. По умолчанию используется режим On.

Предложение **Frame Contents** управляет режимом обновления содержимого рамок. Отчет может содержать от одного или более таких объектов, отображающих изображения из других открытых окон в MapInfo (Карт, Графиков, Легенд и т. п.). При изменении данных в окне, на базе которого был создан отчет, вы можете выбирать следует ли MapInfo Professional тратить время на перерисовывание окна Отчета. Некоторые пользователи предпочитают всегда видеть текущее состояние данных окна. Но поскольку эта операция требует времени, некоторые пользователи могут хотеть, чтобы окно Отчета перерисовывалось только когда оно активно.

Следующий оператор задает программе MapInfo синхронную перерисовку в окне Отчета:

```
Set Layout Frame Contents On
```

Перерисовка может происходить только тогда, когда окно Отчета становится активным. Для этого используется следующий режим:

```
Set Layout Frame Contents Active
```

Для того, чтобы программа MapInfo вообще не обновляла изображение в окне Отчета, используйте следующий вариант:

```
Set Layout Frame Contents Off
```

В последнем случае все объекты типа "рамка" показываются в окне Отчета в виде закрашенных прямоугольников с именами соответствующих окон (например, "РОССИЯ Карта").

Предложение **Ruler** включает режим отображения линеек вверху и слева в окне Отчета. По умолчанию используется режим **On**.

Предложение **Zoom** определяет отношение реальных размеров отчета и представленного на экране макета. Например, следующее предложение задает пятидесятипроцентный масштаб макета:

```
Set Layout Zoom 50.0
```

Другими словами, размеры изображения на экране будут ровно в два раза меньше реальных размеров Отчета. Помните, что страница отчета может иметь очень большие размеры в целях обеспечения высокой точности работы. Соответственно, окно Отчета отображаемое на 200 процентов, покажет увеличенную страницу. Параметр **zoom\_percent** может принимать значения от 6.25% до 800% включительно. В предложении **Zoom** можно не определять масштаб, а потребовать, чтобы все страницы поместились на экране:

```
Set Layout Zoom To Fit
```

**Внимание:** Если в окне выбран объект "рамка", то, используя операторы **Оператор Run Menu Command** (для имитации команд Достать наверх или Подложить вниз) и **Оператор Alter Object**, Вы можете изменить стили линий и штриховок, порядок наложения объектов друг на друга.

Для изменения таких параметров окна Отчета, как ширина окна, высота и расположение его на экране, используйте оператор **Оператор Set Window**.

**Пример:**

```
Set Layout
  Zoom To Fit Extents To Fit
  Ruler Off
  Frame Contents On
```

**См. также:**

**Оператор Alter Object, Оператор Create Frame, Оператор Layout, Оператор Run Menu Command, Оператор Set Window**

---

## Оператор Set Legend

**Назначение**

Изменяет окно легенды тематической карты.

**Синтаксис****Set Legend**

```
[ Window window_id ]
[ Layer { layer_id | layer_name | Prev }
  [ Display { On | Off } ]
  [ Shades { On | Off } ]
  [ Symbols { On | Off } ]
  [ Lines { On | Off } ]
  [ Count { On | Off } ]
  [ Title { Auto | layer_title [ Font... ] } ]
  [ SubTitle { Auto | layer_subtitle [ Font... ] } ]
  [ Style Size { Large | Small } ]
  [ Columns number_of_columns ]
  [ Ascending { On | Off } | Order { Ascending | Descending |
    Custom } ]
  [ Ranges { Auto | [ Font... ]
    [ Range { range_identifier | default } ]
    range_title [ Display { On | Off } ] }
    [ , ... ]
  ]
]
```

*window\_id* - целочисленный идентификатор окна Карты.

*layer\_id* – идентификатор слоя карты, короткое целое число (SmallInt);

*layer\_name* – строка, определяющая слой карты.

*layer\_title, layer\_subtitle* – строки заголовка и подзаголовка легенды тематической карты;

*range\_title* – строка, описывающая выделенный диапазон значений слоя.

### Описание

Оператор **Set Legend** задает настройки для окна "Легенда". Этот оператор позволяет программе управлять отображением условных обозначений. Для изменения таких параметров окна "Легенда", как ширина окна, высота и расположение его на экране, используйте оператор **Onepatop Set Window**.

Оператор **Set Legend** может использоваться в файле Рабочего Набора. Для примера Вы можете открыть окно Карты, создать условное выделение, открыть окно Легенды и сохранить Рабочий Набор (например, под именем LEGEND.WOR). Теперь откройте файл Рабочего Набора в любом текстовом редакторе и Вы увидите оператор **Set Legend**, задающий те настройки, которые были использованы в открытом ранее окне Легенды.

Независимо от количества открытых окон Карт, на экране может присутствовать только одно окно "Легенды", относящееся к одному окну Карты. Окно Легенды отображает информацию о Карте. В операторе **Set Legend** параметр *window\_id* задает окно Карты, а не легенды. Если параметр *window\_id* не задан, то оператор использует самое верхнее окно Карты.

Предложение **Layer** задает изменения в описании слоя в Легенде. Слой идентифицируется либо своим порядковым номером в окне Карты, либо именем, либо задается словами **Layer Prev**. Предложение **Layer Prev** идентифицирует слой, который был создан или изменен последним с помощью операторов **Onepatop Set Shade** или **Onepatop Shade**.

Если Карта содержит два или более тематических слоев, то оператор **Set Legend** может содержать столько же предложений **Layer**, по каждому на один тематический слой.

Оставшиеся ключевые слова в операторе **Set Legend** составляют предложение **Layer**, то есть описываемые ниже предложения являются ключевыми словами в предложении **Layer**.

Предложение **Count** определяет показ в скобках количества записей, принадлежащих к данному диапазону. Предложения **Shades**, **Symbols** и **Lines** задают показ в строке легенды элементов оформления, соответствующих диапазонам. Если оператор включает предложение **Shades On**, то в окне легенды будут показаны образцы штриховок, используемых картой. Если оператор включает предложение **Symbols On**, то в окне легенды будут показаны образцы символов точечных объектов. Если оператор включает предложение **Lines On**, то в окне легенды будут показаны образцы линий и контуров.

Предложения **Title** и **Subtitle** управляют показом заголовка и подзаголовка соответственно. Каждая из этих строк не должна быть длиннее 32 символов. В окне легенды строка заголовка всегда располагается сверху, строка подзаголовка ниже, а затем идут описания диапазонов. Если используется предложение **Auto**, текст генерируется автоматически.

Параметром этого оператора **Font** можно задать стиль оформления текста.

Если не задать предложение **Ascending On**, то строки легенды, описывающие диапазоны, будут располагаться в стандартном, возрастающем порядке. Иначе они расположатся в убывающем порядке.

Предложение **Ranges** задает текстовое описание диапазона (*range\_title*), скомбинированное с режимом, задаваемым предложением **Display**. Вы должны задать параметр *range\_title* для каждого диапазона, а управлять его представлением можно предложениями **Display On** или **Display Off**. Предложение **Ranges** должно включать предложение *range\_title Display* для каждого диапазона тематической Карты, даже для диапазонов, которые не будут показаны.



Если на некий слой условное выделение производилось методом размерных символов, то Вам понадобится две комбинации **range\_title Display**. Если условное выделение производилось методом задания плотности точек, то Вам понадобится задать одну комбинацию **range\_title Display**. В остальных случаях Вам надо будет дополнительно к описаниям диапазонов задать еще одну строку для диапазона "остальные", в который попадают все объекты, не охваченные диапазонами.

Предложения **Order** и **Range** изменяют версию рабочего набора на последнюю. Старые рабочие наборы будут анализироваться корректно, поскольку поддерживаются предложением **Ascending**. Если порядок нетипичный, Mapinfo Professional запишет исходное предложение **Ascending** и не будет обновлять версию рабочего набора.

Предложение **Order** - это еще один способ указать порядок записей (по возрастанию или по убыванию). Исходное предложение **Ascending** поддерживает обратную совместимость. Вы можете использовать предложение **Order** или **Ascending**, но не оба одновременно. Оба предложения не могут быть одновременно включены в один и тот же оператор MapBasic, поскольку это приведет к ошибке синтаксиса.

Вариант **Custom** для предложения **Order** разрешен только для Индивидуальных значений. Во всех остальных случаях появится ошибка. Эта ошибка выглядит следующим образом: "Только для карты индивидуальных значений можно задавать произвольный порядок сортировки."

Если вариант **Custom** используется, каждый диапазон в предложении **Ranges** должен содержать идентификатор. В противном случае появится ошибка синтаксиса. Идентификатор диапазонов должен идти перед заголовком диапазона и предложением **Display**. Идентификатор диапазонов является той же строкой или значением, которые используются в предложении **Values** в **Operator Shade**. Для категории "все остальные" используется идентификатор "по умолчанию".

Должны быть учтены все категории, включая категорию "по умолчанию", в противном случае появится ошибка. Ошибка выглядит следующим образом: "Неверное число диапазонов для произвольного порядка сортировки."

Порядок размещения категорий "по умолчанию" и "все остальные" также может быть изменен, независимо от того, что наиболее удачное место расположения этих аргументов в начале или в конце предложения **Ranges**.

Если идентификатор не соответствует правильной категории, появится ошибка. Ошибка выглядит следующим образом: "Неверный диапазон для произвольного порядка сортировки.."

Предложение **Style Size** изменяет размер.

Предложение **Columns** позволяет задавать ширину легенды. *number\_of\_columns* указывает ширину колонки.

**См. также:**

**Оператор Map, Оператор Open Window, Оператор Set Map, Оператор Set Window, Оператор Shade**

## Оператор Set Map

### Назначение

Изменяет существующее окно карты.

### Синтаксис

Основная часть оператора **Set Map**

имеет следующий синтаксис:

#### Set Map

```
[ Window window_id ]
[ Center ( longitude, latitude ) [ Smart Redraw ] ]
[ Clipping { Object clipper | Off | On } | Using
  [Display { All | PolyObj } | Overlay ] ] ]
[ Zoom { zoom_distance [ Units dist_unit ] |
  Entire [ Layer layer_id ] } ]
[ Preserve { Scale | Zoom } ]
[ Display { Scale | Position | Zoom } ]
[ Order layer_id, layer_id [ , layer_id ... ] ]
[ Pan pan_distance [ Units dist_unit ]
  { North | South | East | West } [ Smart Redraw ] ]
[ CoordSys... ]
[ Area Units area_unit ]
[ Distance Units dist_unit ]
[ Distance Type { Spherical | Cartesian } ]
[ XY Units xy_unit ]
[ Display Decimal { On | Off | [ Display Grid ] ]
[ Scale screen_dist [ Units dist_unit ] For map_dist
  [ Units dist_unit ] ]
[ Redraw { On | Off } ]
{ Image Reprojection { None | Always | Auto } ]
[ Image Resampling { CubicConvolution | NearestNeighbor } ]
[ Relief { On | Off } ]
[ Move Nodes { value | Default } ]
[ LAYERCLAUSE LAYERCLAUSE ... ]
```

*window\_id* - целочисленный идентификатор окна Карты.

*longitude, latitude* – координаты центра карты;

*clipper* – выражение Object; будет показана только часть карты, в пределах объекта. Смотрите описание в разделе Clipping.

*zoom\_distance* – числовое выражение, описывающая ширину области показа;

*map\_layer\_id* определяет слой карты; задаётся либо числом типа SmallInt (коротким целочисленным; например, используйте 1 для определения самого верхнего слоя, не считая косметического), либо строковая переменная, соответствующая имени таблицы, отображенной на карте.

*pan\_distance* – расстояние, на которое нужно сместить карту;

**Оператор CoordSys** определяет систему координат.

*area\_unit* – строка, определяющая название единицы измерения площади (например, “кв. км” для квадратных километров; список названий единиц измерения смотрите в **Оператор Set Area Units** на стр. 88).

*distance* – может быть рассчитана на сфере (**Spherical**) или на плоскости (**Cartesian**). Все расстояния, длины, периметры и площади для объекта в окне Карты могут рассчитываться по одному из двух методов. Обратите внимание, что если система координат окна Карты является план-схемой, то вычисления будут по декартовому методу (**Cartesian**), независимо от настроек, и если система координат окна Карты это Широта/Долгота, то вычисления будут осуществляться по сферическому методу (**Spherical**), независимо от настроек.

*xy\_unit* – строка, задающая единицу измерения X/Y-координат (например, “m” – метр или “degree” – градус); Если координаты **XY Units** – градусы, предложение **Display Decimal** определяет использование десятичных градусов. Поставьте **On**, чтобы отображать десятичные градусы или **Off**, чтобы отображать координаты в градусах, минутах или секундах. Используйте **Display Grid**, чтобы использовать единицы армейской системы координат.

*Image Reprojection* имеет три варианта использования: **Always**, **Auto** (в этом случае повторная дискретизация выполняется автоматически) и **None**.

*Always* означает что перепроецирование осуществляется всегда; координаты пересчитываются используя точные формулы и пиксели сглаживаются используя методы “cubic convolution” или “nearest neighbor”.

*Auto* – определяет, что перепроецирование выполняется на основе сведений о внешнем виде полученного изображения после трансформации к области занятой исходным изображением. Если этот прямоугольник выглядит “правильным” (обе пары сторон параллельны друг другу и осям), то будет использован старый код MI Pro, т. е. стандартные функции Windows для растяжения исходного изображения по обоим направлениям. Это самый быстрый способ отобразить результирующий растр. Если используется другой метод (сжатие не эффективно из-за нелинейности и/или искажений на полученном изображении), работает новый код перепроецирования.

*None* – при этом MapInfo Professional работает с растровыми слоями также как в предыдущих версиях (более ранних чем 8.5), изменяя векторные слои в соответствие с растровым.

*Image Resampling* – два варианта повторной дискретизации: **кубическая свертка (cubic convolution)** и **ближайшее соседство (nearest neighbor)**.

*CubicConvolution* – кубическая свёртка, метод сглаживания растра обеспечивает наилучшее “восстановление” значений пикселей, отсутствующих на исходном изображении (из-за дискретности). При этом значение пиксела на полученном изображении вычисляется по окну размером 4x4 вокруг “базового” пиксела на исходном изображении. Координаты (обычно, вещественные) исходного пиксела вычисляются для каждого пиксела получаемого изображения по специальной оптимизированной процедуре. Пиксели внутри окна, описанного выше, определенным образом взвешиваются по мантиссам координат исходного пиксела.

*NearestNeighbor* – ближайшее соседство, метод сглаживания растра, при котором значение исходного пиксела просто переносится на новое место.

*Relief* - включает и отключает отмывку рельефа в гридах. Грид должен иметь рассчитанные значения необходимые для создания отмывки, чтобы это предложение имело эффект. Информация об отмывке рельефа может быть рассчитана для грида командой **Оператор Relief Shade**.

*Move Nodes* – может принимать значения 0 или 1. Если значение равно 0, дублирующиеся узлы не удаляются. Если значение равно 1, все дублирующиеся узлы с одного слоя удаляются. Если **величина Move Node** задана, то будет использоваться заданное значение. Если эта величина не задана, то будет использоваться стандартное значение из Настроек.

*screen\_dist* и *map\_dist* – задают масштаб Карты (например, *screen\_dist* = 1 inch, *map\_dist* = 1 mile).

В синтаксисе выше *LAYERCLAUSE* - это не ключевое слово MapBasic, а название раздела, который будет описан ниже. *LAYERCLAUSE* соответствует одному слою Карты и имеет следующий синтаксис:

```
[ Layer layer_id
[ Activate { Using launch_expr[ On { [ [ Labels ] | [ Objects ] } ] } | [
Relative Path { On | Off } ] [ Enable { On | Off } ] },
[ Using launch_expr[ On { [ [ Labels ] | [ Objects ] } ] } | [ Relative
Path { On | Off } ] [ Enable { On | Off } ] ]
[ Editable { On | Off } ]
[ Selectable { On | Off } ]
[ Zoom ( min_zoom, max_zoom ) [ Units dist_unit ] [{ On | Off } ] ]
[ Arrows { On | Off } ]
[ Centroids { On | Off } ]
[ Default Zoom ]
[ Nodes { On | Off } ]
[ Inflect num_inflections [ by percent ] at
color:value [, color:value ]
[ Round rounding_factor ]
Contrast contrast_value ]
[ Brightness brightness_value ]
[ { Alpha alpha_value }|{ Translucency translucency_percent } ]
[ Transparency { Off | On } ]
[ Color transparent_color_value ]
[ GrayScale { On | Off } ]
[ Relief { On | Off } ]
LABELCLAUSE
[ Display { Off | Graphic | Global } ]
[ Global Line... ]
[ Global Pen... ]
[ Global Brush... ]
[ Global Symbol... ]
[ Global Font... ]
]
```

*layer\_id* определяет слой карты; задаётся либо числом типа SmallInt (коротким целочисленным; например, используйте 1 для определения самого верхнего слоя, не считая косметического), либо строковая переменная, соответствующая имени таблицы, отображенной на карте.

*launch\_expr* - это выражение, которое обеспечивает запуск файла, когда активизируется ассоциированный с ним объект.

*min\_zoom* – минимальное значение для масштабного эффекта слоя;

*max\_zoom* – максимальное значение для масштабного эффекта слоя.

*color* - выражение цвета, использующее функцию **Функция RGB( )**.

*value* - перелом, отображаемый парным цветом.

*num\_inflections* - это числовое выражение, определяющее число точек перелома цветов:значений.

*alpha\_value* - это целое число, представляющее значение альфа-канала для полупрозрачности. Значение изменяется в пределах 0-255. 0 - это полная прозрачность. 255 - это полная непрозрачность. Значения между 0-255 делают изображение слоя полупрозрачным.

*translucency\_percent* - это целое, представляющее процент полупрозрачности для растров и слоев поверхности. Значение изменяется в пределах 0-100. 0 - это полная непрозрачность. 100 - это полная прозрачность.

Или **Alpha**, или **Translucency** должны быть определены, но каждый из них определяет то же самое но разными путями. Если указано несколько таких параметров, будет использоваться последний. Параметры **Alpha** и **Translucency** являются новыми в операторе Set Map. Они применяются для растровых слоев и слоев поверхности.

Параметры **Contrast**, **Brightness**, и **Grayscale** теперь поддерживаются и для растровых слоев. Их можно применять и для растровых слоев, и для слоев поверхности.

Параметры **Transparency** и **Color** являются новыми для Set Map и применяются только для растровых слоев. Параметр **Transparency** определяет, является ли индивидуальный цвет прозрачным для растрового слоя. Параметр **Color** указывает какой из цветов на растровом слое является прозрачным.

Предложение **Using** устанавливает выражение для имени файла и предложение **On** устанавливает режим активизации. Требуется как минимум одно из этих предложений. Если включено предложение **Using**, то требуется задать параметр *filename\_expr*.

Если включено предложение **On clause**, то требуется или одно или оба предложения **Labels** и **Objects**. Если включено только предложение **Labels**, то произойдет активизация только подписей. Если включено предложение **Objects**, то произойдет активизация только объектов. Если присутствуют оба этих ключевых слова, то произойдет активизация и объектов, и подписей. В стандартном варианте активизируются только подписи то произойдет активизация только объектов.

Используйте **Relative Path On** когда запускаемые файлы хранятся в местах, которые определяются относительно таблицы, с которой они связаны. Используйте **Relative Path Off** когда геолинки (HotLinks) имеют адреса URL или полный путь к файлам;это стандартный вариант.

Предложение **Line** определяет стиль объектов "линия" и "полилиния".То же для предложения **Pen**, за исключением использования ключевого слова **Pen** вместо **Line**.

Предложение **Предложение Pen** определяет стиль линии контуров заштрихованных объектов.

Предложение **Предложение Brush** определяет стиль штриховки.

Предложение **Предложение Symbol** определяет стиль символа.

Предложение **Предложение Font** определяет стиль шрифта текстовых объектов.

В синтаксисе выше *LAYERCLAUSE* - это не ключевое слово MapBasic, а название раздела, который будет описан ниже. *LAYERCLAUSE* соответствует одному слою Карты и имеет следующий синтаксис:

```
[ Label [ Line { Simple | Arrow | None } ]
[ Position [ Center ] [ Above | Below ] [ Left | Right ] ]
[ Font... ] [ Pen... ]
[ With label_expr ]
[ Parallel { On | Off } | Follow Path ]
[ Visibility { On | Off | Zoom( min_vis, max_vis ) [ Units dist_unit ] } ]
[ Auto [ { On | Off } ] ]
[ Overlap [ { On | Off } ] ]
[ PartialSegments { On | Off } ]
[ Duplicates [ { On | Off } ] ]
[ Max [ number_of_labels ] ]
[ Offset offset_amount ]
[ Default ]
[ Object ID
  [ Table alias ]
  [ Visibility { On | Off } ]
  [ Anchor ( anchor_x, anchor_y ) ]
  Text text_string
  [ Position [ Center ] [ Above | Below ] [ Left | Right ] ]
  [ Font... ] [ Pen... ]
  [ Line { Simple | Arrow | None } ]
  [ Angle text_angle ]
  [ Offset offset_amount ]
  [ Callout ( callout_x, callout_y ) ] }
[ Object... ]
]
```

*label\_expr* – выражение, используемое для подписывания объекта;

*min\_vis*, *max\_vis* – минимальное и максимальные значения для масштабного эффекта подписи;

*dist\_unit* – строка с именем единицы измерения (например, “mi” для миль, “m” для метров; см. описание оператора **Оператор Set Distance Units на стр. 103**.

*number\_of\_labels* – целое число типа Integer, представляющее максимальное количество подписей, которое MapInfo Professional может показать на слое (по умолчанию лимита нет);

*offset\_amount* – целое число от 0 до 200, расстояние в точках от подписи до точки привязки на подписываемом объекте;

*ID* – целочисленный идентификатор изменяемой подписи (это предложение генерируется автоматически при сохранении Рабочего Набора); Идентификатор подписи (*ID*) равен идентификатору строки, к которой присоединен подписываемый объект

*alias* – псевдоним таблицы-компонента сшитой Карты. Предложение **Table** *alias* порождает ошибку, если слой не является компонентом сшитой Карты.

*anchor\_x*, *anchor\_y* – координаты, задающие закрепленное положение подписи;

*text\_string* – строка с текстом подписи;

*text\_angle* – угол поворота подписи в градусах;

*callout\_x*, *callout\_y* – координаты, определяют конец указки при подписи.

*Parallel* – можно использовать со следующими атрибутами:

*Parallel Off* = горизонтальные подписи, без поворота вдоль линии

*Parallel On* = подписи вдоль линии

*Follow Path* = подпись вдоль кривой, которая будет обчислена, а её параметры будут сохранены

## Описание

Оператор **Set Map** задает настройки для окна Карты. Если параметр *window\_id* не задан, то действия оператора распространяются на самое верхнее из открытых окон Карты. Этот оператор позволяет программе управлять отображением слоев Карты так же, как это может делать пользователь при помощи команд **MapInfo Карты > Управление слоями, Карта > Показать по-другому, Карта > Режимы**. Например, оператор **Set Map** позволяет задать слой карты, который будет доступен для редактирования, и установить масштаб или степень увеличения (zoom) карты.

**Внимание:** Заметим, что оператор **Set Map** управляет настройками в окне Карты. Для изменения таких атрибутов, как размер окна и его расположение на экране, используется оператор **Оператор Set Window**.

Оператор **Set Map** может использоваться в файле Рабочего Набора, если в нем есть хотя бы одно окно Карты. Для примера Вы можете открыть окно Карты и сохранить Рабочий Набор (например, под именем MAPPER.WOR).

Все предложения оператора **Set Map** не являются обязательными, но хотя бы одно должно присутствовать. При этом важен порядок расположения предложений, несоблюдение порядка может привести к синтаксической ошибке в программе.

## Изменение изображения в окне Карты

Следующие предложения изменяют показ Карты в окне, т.е. центральную точку и размеры показываемой в окне области.

**Center** Предложение задает центр карты в окне. Например, город Нью-Йорк расположен приблизительно на 74 долготы и 41 широты. Следующий оператор **Set Map** помещает Нью-Йорк в центр окна Карты. При этом широта и долгота должны задаваться в десятичных единицах, а не в градусах, минутах и секундах.

```
Set Map Center (-74.0, 41.0)
```

Выполнение оператора **Set Map...Center** ведет к полной перерисовке окна Карты, если в оператор не включено предложение **Smart Redraw**. Детали о предложении **Smart Redraw** смотрите ниже.

**Pan** Предложение перемещает окно карты в заданном направлении. Например, следующий оператор перемещает Карту на 100 километров к северу:

```
Set Map Pan 100 Units "km" North
```

Обычно, после выполнения оператора **Set Map ... Pan** изображение в окне Карты перерисовывается полностью. Если в оператор включено предложение **Smart Redraw**, то MapInfo будет обновлять только ту часть изображения, которая этого требует (обновление изображения в окне Карты будет происходить так, как, если бы пользователь использовал в окне инструмент Ладощка).

```
Set Map Pan 100 Units "km" North Smart Redraw
```

**ВНИМАНИЕ:** если используется предложение **Smart Redraw**, окно Карты передвигается шагами, кратными восьми пикселям. Из-за этого Карта может показываться не совсем так, как ожидалось. Например, передвигая Карту на Север на 100 км, Вы добьетесь только передвижения на 80 километров, из-за того, что 20 километров на экране занимают более восьми пикселей.

**Scale** Изменяет масштаб показа Карты в окне. Например, следующий оператор изменяет увеличение Карты так, чтобы в 1 дюйме изображения было показано 10 миль Карты:

```
Set Map Scale 1 Units "in" For 10 Units "mi"
```

**Zoom** Определяет размер в ширину фрагмента Карты, показанного в окне. Например, следующий оператор Set Map показывает участок шириной в 100 километров (если текущими единицами расстояний в MapBasic сейчас являются "km"):

```
Set Map Zoom 100 Units "km"
```

Предложение **Zoom Entire Layer layer\_id** является эквивалентом команды **Карта > Показать слой полностью** в MapInfo Professional. Если опустить из этого предложения слово "Layer", то будут полностью показаны все слои Карты.

```
Set Map Zoom Entire Layer 2 'Полностью показать 2 слой
Set Map Zoom Entire 'Полностью показать всю карту
```

### Управление поведением Карты в окне

Следующие предложения определяют поведение Карты в окне.

**Area Units** - Предложение задает единицы измерения площади. Список возможных единиц приведен в описании оператора **Оператор Set Area Units на стр. 88**.

```
Set Map Area Units "sq km"
```

**Clipping** Предложение задает фрагмент-врезку в окне Карты. Операция соответствует действию в MapInfo команды **Карта > Выбрать область врезки**. После назначения области врезки Вы можете управлять показом фрагмента-врезки операторами **Clipping On** или **Clipping Off**.



```
Set Map Clipping Object obj_variable_name
```

**Clipping** Предложение задает фрагмент-врезку в окне Карты. Операция соответствует действию в MapInfo команды **Карта > Выбрать область врезки**. После назначения области врезки Вы можете управлять показом фрагмента-врезки операторами **Clipping On** или **Clipping Off**.

Существует три режима, которые могут использоваться для создания врезки. Использование режима **Overlay** приводит к применению функциональности MapInfo Overlay (**Erase Outside**) для создания врезки. Полилинии и Регионы будут обрезаться по границе обрезающего полигона. Точки и Подписи будут показаны на врезке полностью только в том случае, если они лежат внутри обрезающего полигона. Текст всегда отображается и никогда не обрезается. Стили для всех объектов никогда не обрезаются. (Этот метод используется во всех версиях, ранее MapInfo 6.0.) Использование режима **Display All**, приводит к созданию врезки системными средствами Windows. Все объекты (включая точки, подписи и текст) будут обрезаться по границе полигона, ограничивающего врезку. Все стили также обрезаются по границе полигона. Этот режим является стандартным.

Использование режима **Display PolyObj** приводит к созданию врезки системными средствами Windows, причем обрезаются только полилинии и регионы. Стили для полилиний и регионов обрезаются по границе полигона врезки. Точки и Подписи будут показаны на врезке полностью только в том случае, если они лежат внутри обрезающего полигона. Текст всегда отображается и никогда не обрезается. Стили для точек, подписи и текст никогда не обрезаются. Этот режим по функциональности близок к режиму Overlay в ранних версиях MI Pro, до 6.0.

В главном, системные средства Windows из методов **Display All** и **Display PolyObj** обеспечивают лучший результат чем функциональность Overlay. Например:

```
Set Map Clipping Object obj_variable_name Using Display All
```

параметр CoordSys задает в окне Карты систему координат и проекцию. Синтаксис смотрите в описании стандартного предложения [Оператор CoordSys on page 175](#).

Система координат MapBasic должна быть точно задана при помощи [Оператор Set CoordSys](#) и может быть восстановлена при помощи [Функция SessionInfo\(\)](#).

**Внимание:** Если оператор **Set Map** используется с параметром CoordSys, то координатная система, установленная для программы MapBasic, автоматически приводится в соответствие с системой координат карты.

В версиях до 7.x этот пример установит систему координат в UTM как для карты, так и для внутренней системы координат MapBasic:

```
Set Map XY Units "m" CoordSys Earth Projection 8, 33, "m", -55.5, 0,
0.9999, 304800, 0
```

В версиях 7.x и старше этот пример изменит систему координат и единицы измерений только самой карты; система координат MapBasic останется без изменений.

**Display** Предложение Display выбирает, что показывать в левом нижнем углу активного окна Карты. **Display Zoom** отображает текущий размер (ширину показываемой области). **Display Scale** отображает текущий масштаб. **Display Position** положение курсора в десятичных единицах координат широта/долгота.

Set Map Display Position

**Distance Units** Предложение задает единицы измерения расстояний в окне "Линейка". Список возможных единиц приведен в описании оператора [Оператор Set Distance Units на стр. 103](#).

Set Map Distance Units "km"

**Preserve** Предложение определяет поведение Карты при изменении пользователем размеров окна. Предложение **Preserve Zoom** задает показ Карты всегда в одном масштабе, независимо от изменений размеров окна. И, наоборот, **Preserve Scale** задает увеличение или уменьшение масштаба карты в зависимости от увеличения или уменьшения окна, сохраняя в окне Карты фрагмент постоянного размера. Установки этого предложения такие же, как соответствующие режимы в диалоге, вызываемом командой **Карта > Показать**.

**Redraw** Предложение управляет автоматической перерисовкой Карты в окне. Если программа выполнила оператор **Set Map Redraw Off**, то следующие операторы, изменяющие карту (такие как **Set Map**, **Add Map Layer**, **Remove Map Layer**), будут выполняться без автоматического обновления изображения в окне Карты. После внесения всех нужных изменений в окно Карты, выполните оператор **Set Map Redraw On**, восстанавливающий режим автоматической перерисовки окна.

**Внимание:** В окне отобразятся все изменения, которые были выполнены "вслепую". Если оператор **Redraw Off** не предотвращает перерисовку окна, то можно использовать оператор **Set Event Processing Off**.

**XY Units** Предложение задает координатные единицы положения курсора, показываемые в левом нижнем углу окна Карты. Единицами могут быть градусы ("degree") для измерения широты и долготы или единицы измерения расстояния, например, метры ("m").

Если **XY Units** представлены в градусах, предложение **Display Decimal** укажет следует ли использовать десятичные градусы (**On**) или градусы, минуты, секунды (**Off**). **Display Grid** - координаты будут отображаться в единицах армейской системы координат, независимо от того как определены **XY Units**.

```
Set Map XY Units "m" Display Grid
Set Map XY Units "degree" Display Grid
Set Map XY Units "degree" Display Decimal On
Set Map XY Units "degree" Display Decimal Off
```

Следующий оператор указывает в качестве единиц системы координат метры:

```
Set Map XY Units "m"
```

### Изменение порядка слоев

Предложение **Order** задает порядок прорисовки слоев Карты на экране. Каждый параметр *layer\_num* – номер слоя Карты. Единица соответствует самому верхнему слою Карты (который рисуется последним, поверх остальных).

Косметический слой является специальным слоем и имеет номер слоя, равный 0 (нулю). Он рисуется всегда последним и его не нужно задавать в предложении **Order**. В следующем примере первый (верхний) слой Карты и второй (находящийся под ним) меняются местами. Косметический слой остается самым верхним.

```
Set Map Order 2, 1, 3, 4
```

## Изменение поведения отдельного слоя

**Editable** устанавливает атрибут изменяемости для соответствующего Слоя. В предложении Layer предложение **Editable** может установить режим изменяемости только для одного слоя. Если слой становится изменяемым, то он автоматически становится и доступным. Следующий оператор **Set Map** делает изменяемым только верхний некосметический слой:

```
Set Map
Layer 1 Editable On
```

**Selectable** Предложение устанавливает для данного слоя режим доступности, т.е. разрешает выбирать на нем объекты такими инструментами, как Стрелка. Доступными могут быть несколько слоев Карты или все слои. Следующий оператор **Set Map** делает доступным верхний некосметический слой, а следующие два – недоступными:

```
Set Map
Layer 1 Selectable On
Layer 2 Selectable Off
Layer 3 Selectable Off
```

**Zoom** Предложение задает пределы для масштабного эффекта, то есть режима показа слоя только в определенных пределах увеличения. У каждого слоя могут быть пределы масштабного эффекта. Когда эти пределы активированы, MapInfo Professional отображает слой карты только когда ее масштаб находится в указанном диапазоне. Например, улицы города можно показывать только тогда, когда размер карты уменьшится до 10 км.

```
Set Map Layer 1 Label Visibility Zoom (0, 10) Units "km"
```

Предложение Zoom может содержать слово **On**, включающее масштабный эффект для слоя. Предложение Zoom может содержать слово **Off**, отключающее масштабный эффект для слоя.

## Предложение Set Map Clause для геолинка (HotLink)

Активный объект – это такой объект в окне Карты, который имеет адрес URL или с которым связан ассоциированный файл. Выбор такого объекта инструментом Геолинк позволяет получить доступ к связанному с ним URL-адресу или файлу. Например, если строка `http://www.esti-map.ru` ассоциирована с точкой на карте, то щелкнув на таком точечном объекте, или подписи, получим в результат открывшейся Интернет-браузер, показывающий страницу `http://www.esti-map.ru`. Можно связывать с объектами карты другие типы файлов; Рабочие наборы MapInfo (.wof), таблицы (.tab) или приложения (.mbx), документы Word (.doc), исполняемые файлы (.exe), и другие. Любые типы файлов, которые система может запускать, могут быть связаны с объектами карты.

## Предложение Layer Activate

Параметры геолинков можно настраивать с помощью Set Map Layer Activate, который позволяет обрабатывать несколько определений геолинка. Можно добавлять новые элементы, изменять атрибуты существующих, удалять их и изменять порядок обращения к элементам. Ниже описаны преимущества использования предложения Activate, включая подробности синтаксиса. Смотрите также раздел **Исключения, обеспечивающие совместимость с предыдущими версиями**.

Обратите внимание, что свойства индивидуального геопинка остались такими же как и в прежних версиях, кроме нового параметра **Enable**, с помощью которого можно выключить активность геопинка, оставив его определение. (Ранее геопинк можно было отключить, только задав "" в его определении и потеряв исходное определение).

### Назначение

С помощью параметра **Activate** можно определять новые геопинки. Геопинки используются для того чтобы открывать файлы или новые адреса URL в Интернет-браузере прямо из окна Карты.

### Синтаксис

```
{ Using launch_expr[ On { [ [ Labels ] | [ Objects ] ] } ] | [ Relative  
Path { On | Off } ] [ Enable { On | Off } ] },  
[ Using launch_expr[ On { [ [ Labels ] | [ Objects ] ] } ] | [ Relative  
Path { On | Off } ] [ Enable { On | Off } ] ]...
```

### Пример:

```
Set Map Layer 1 Activate Using Url1 On Objects Relative Path Off Enable  
On, Using Url2 On Objects Relative Path On Enable On
```

### Замечания

Эта версия команды уничтожит существующее определение, но создаст одно или несколько новых. Следует обязательно использовать параметр **Using**, не допуская пустых строк в launch\_expr (например, ""). Если используется параметр Enable, а его значение установлено в состояние Off, то определение геопинка будет выключено. Параметры On, Relative и Enable можно опустить.

### Как добавить новые определения геопинка

### Синтаксис

```
Activate Add [ First ]  
{ Using launch_expr[ On { [ [ Labels ] | [ Objects ] ] } ] | [ Relative  
Path { On | Off } ] [ Enable { On | Off } ] },  
[ Using launch_expr[ On { [ [ Labels ] | [ Objects ] ] } ] | [ Relative  
Path { On | Off } ] [ Enable { On | Off } ] ]...
```

### Примеры

```
Activate Add Using URL1 On Objects Relative Path On, Using URL2 On Objects  
Enabled Off  
Activate Add First Using URL1 On Objects
```

### Комментарии

Версия этой команды, применяемая в MapBasic 9.0, добавит новое определение геопинка в конец списка определений.

Если включить в состав команды дополнительный параметр *First*, то новые определения будут добавлены в начало списка.

Если используется параметр *Enable*, а его значение установлено в состояние *Off*, то определение геолинка будет выключено.

Параметры *On*, *Relative* и *Enable* можно опустить.

### Как изменить существующие определения геолинка

#### Синтаксис

```
Activate Modify
{ hotlink_id[ Using launch_expr ] | [ On { [ [ Labels ] | [ Objects ] ] }
] | [ Relative Path { On | Off } ] [ Enable { On | Off } } ,
[ hotlink_id[ Using launch_expr ] | [ On { [ [ Labels ] | [ Objects ] ] }
] | [ Relative Path { On | Off } ] [ Enable { On | Off } ] ...
```

#### Примеры

```
Activate Modify 1 Using URL1 On Objects, 2 Relative Path Off
Activate Modify 2 On Objects, 4 On Labels
Activate Modify 3 Relative Path On Enable Off
Activate Modify 2 Enable Off, 3 Enable On
```

#### Комментарии

*hotlink\_id* – целочисленный указатель, с помощью которого можно задать определение геолинка

Требуется использовать хотя-бы один параметр из *Using/On/Relative/Enabled*.

*launch\_expr* – не может быть пустой строкой (например, "").

Если используется параметр *Enable*, а его значение установлено в состояние *Off*, то определение геолинка будет выключено.

### Как удалять определения геолинка

#### Синтаксис

```
Activate Remove { All | hotlink_id [ , hotlink_id, hotlink_id, ... ] }
```

#### Примеры

```
Activate Remove 2, 4
Activate Remove All
```

#### Комментарии

*hotlink\_id* – целочисленный указатель, с помощью которого можно задать определение геолинка

Необходимо задать хотя бы один *hotlink\_id*.

Если используется параметр *All*, то будут удалены все определения геолинка.

*Как изменить последовательность определений геолинка*

**Синтаксис**

```
Activate Order { hotlink_id [ , hotlink_id, hotlink_id, ... ] }
```

**Примеры**

```
Activate Order 2, 3, 1
```

**Комментарии**

*hotlink\_id* – целочисленный указатель, с помощью которого можно задать определение геолинка

Необходимо задать хотя бы один *hotlink\_id*.

*Исключения, обеспечивающие совместимость с предыдущими версиями*

Параметр **Using** можно опустить, но только для первого определения геолинка. Связанное с параметром **Using** выражение может быть пустым (""), но только для первого определения геолинка.

**Без параметра Using**

В обеих следующих командах опущен параметр *Using*, что в случае использования версии 850 можно использовать для обновления свойств единственного определения геолинка, даже если параметр *Using* никогда не задавался В версии 900 такие команды могут вызвать проблемы, поскольку слои карт создаются вообще без определений геолинка.

- Activate On Objects
- Activate Relative Path On

Для того чтобы решить проблему, если опущен параметр *Using*, появится сообщение об ошибке, за исключением случая, когда команда выполняется программой или Рабочим набором, созданными до версии 900. В таблице ниже перечислены примеры использования разных сценариев.

| Действие          | Количество геолинков | Результат  |
|-------------------|----------------------|--|
| Activate Using "" | Один или несколько   | Первое определение геолинка состоит из пустой строки. Определение отключено до тех пор, пока ему не будет присвоено какое-либо не пустое значение. |
| Activate Using "" | Ни одного            | Будет создан новый геолинк с пустым определением. Определение отключено до тех пор, пока ему не будет присвоено какое-либо не пустое значение.     |

|                     |                    |   |
|---------------------|--------------------|---|
| Activate On Objects | Один или несколько | <p>Если выполняется в приложении или Рабочем наборе, созданном ранее версии 9.0:</p> <ul style="list-style-type: none"> <li>• обновляется первое определение геолинка значением, задаваемым выполняемой командой. Выражение не будет изменено.</li> </ul> <p>Если выполняется в приложении или Рабочем наборе, созданном в версии 9.0 (или более новой):</p> <ul style="list-style-type: none"> <li>• появится сообщение о синтаксической ошибке "Пропущен обязательный параметр Using"</li> </ul>  |
| Activate On Object  | Ни одного          | <p>Если выполняется в приложении или Рабочем наборе, созданном ранее версии 9.0:</p> <ul style="list-style-type: none"> <li>• Будет создан новый геолинк с первым пустым определением и остальными значениями, задаваемыми командой. Определение будет отключено до тех пор, пока ему не присвоят какое-либо непустое значение.</li> </ul> <p>Если выполняется в приложении или Рабочем наборе, созданном в версии 9.0 (или более новой):</p> <ul style="list-style-type: none"> <li>• появится сообщение о синтаксической ошибке "Пропущен обязательный параметр Using"</li> </ul> |

Обратите внимание, что те же действия будут выполняться при чтении метаданных таблицы.

### ***Пустой параметр Using***

Следующая команда создает выражение с пустой строкой, что по существу отключает геолинк на слое. Если выражение, включенное в геолинк, является пустой строкой, то определение геолинка не используется, до тех пор пока оно не будет заменено непустой строкой. В версии 850 так можно было включить/выключить геолинк. В версии 900 используется включение/выключение геолинка в явном виде командой **Set map Layer Activate Enable On/Off**, поэтому использования пустых строк следует избегать.

Activate Using ""

Допускается использование пустого выражения, но только в первом определении геолинка. Как и в версиях, предшествующих 9.0, геолинк с пустым выражением будет отключен.

Если в команде **Set Map Layer Activate** нет параметра **Using** или параметр **Using** пустой, то результат зависит от состояния геолинков слоев. В следующей таблице перечислены возможные сценарии:

В качестве решения проблемы, MapInfo Professional позволяет использовать пустое выражение, но только в первом определении геолинка. Как и в версиях предшествующих 900, геолинк с пустым выражением будет отключен.

## Настройка относительного адреса (Relative Path)

Настройка относительного адреса (**Relative Path**) позволяет определить местоположение файла относительно таблицы. Например: пусть таблица `c:\data\states.tab` содержит геолинки к файлам рабочих наборов, хранящихся в директории `c:\data`. Файл рабочего набора для Нью-Йорка, `newyork.wor`, хранится в `c:\data\ny` и Геолинк связан с `"ny\newyork.wor"`. **Установка относительного пути** сообщает MapInfo, что надо добавить вторую часть к пути, по которому находится `.tab` файл, чтобы получился полный путь, и в результате получится строка `"c:\data\ny\newyork.wor"`.

**Внимание:** Геолинки, определенные как адреса URL, не изменяются перед запуском, независимо от установки относительного пути. Чтобы определить, является ли Геолинк адресом URL, используется функция ShellAPI.

## Изменение представления отдельного слоя

**Arrows** Предложение управляет показом стрелок.

**Centroids** Предложение управляет показом центроидов объектов.

**Inflect** - переопределяет пары цвет:значения, хранящиеся в файле поверхностей (.MIG).

**Nodes** Предложение управляет показом узлов на объектах.

Следующий пример включает режим показа стрелок, центроидов и узлов объектов первого некосметического слоя Карты:

```
Set Map Layer 1 Arrows On Centroids On Nodes On
```

Предложение **Display** управляет показом слоя в окне Карты.

Предложение **Display Off** отменяет показ слоя; **Display Graphic** показывает объекты слоя в собственном (сохраненным в таблице) оформлении; **Display Global** позволяет настраивать отдельные компоненты оформления объектов:

**Global Line** определяет стиль линейных объектов: линий и полилиний. Предложение **Line** имеет конструкцию, подобную **Предложение Pen** за исключением ключевого слова **Line** вместо **Pen**.

Предложение **Global Предложение Pen** задает стиль линий, окружающих замкнутые объекты.

Предложение **Global Предложение Brush** задает стиль штриховки замкнутых объектов.

Предложение **Global Предложение Symbol** задает стиль символов для точечных объектов.

Предложение **Global Предложение Font** задает шрифт для текстовых объектов.

Следующий оператор разрешает показ объектов первого слоя в собственном стиле:

```
Set Map Layer 1 Display Graphic
```

Следующий оператор устанавливает режим показа объектов первого слоя тонкими зелеными линиями, контурами и зеленой заливкой:

```
Set Map
  Layer 1 Display Global
  Global Line(1, 2, GREEN)
```



```
Global Pen (1, 2, GREEN)
Global Brush (2, GREEN, WHITE)
```

### Изменение режима подписывания отдельного слоя

Предложение **Label** задает режим подстановки и поведения подписей на слое. Предложение имеет следующие подпредложения:

**Line** Определяет тип указки, которая сопровождает подпись при ее перемещении, или ее отсутствие. Вы можете задать **Line Simple**, **Line Arrow**, или **Line None**. Например:

```
Set Map Layer 1
  Label Line Arrow
```

**Position** Управляет расположением подписи относительно центроида объекта, к которому она прикреплена. Например, следующий оператор располагает подписи сверху и справа от центроида:

```
Set Map Layer 1
  Label Position Above Right
```

**Font** Определяет шрифт подписи.

**Pen** Задает стиль линии указки. Стиль линии указки имеет смысл для режимов **Line Simple** и **Line Arrow**, и когда пользователь сдвигает подпись с заданного положения:

```
Set Map Layer 1 Label Line Arrow Pen( 2, 1, 255)
```

**With** Задает выражение, образующее текст надписи. Например, следующий оператор использует функцию **Функция Proper\$( )** к значениям из колонки с именами больших городов:

```
Set Map Layer 1 Label With Proper$(Cityname)
```

**Parallel** Определяет, будет ли строка подписи линейного объекта параллельна подписываемой линии.

```
Set Map Layer 1 Label Parallel On
```

**Visibility** Управляет показом подписей для одного слоя. Предложение **Visibility Off** выключает показ как автоматических подписей, так и созданных вручную. Предложение **Visibility Zoom ...** устанавливает показ подписей только, когда Карта находится в определенном масштабном диапазоне. Следующий пример разрешает подписывание тогда, когда размер Карты будет равен 2 километрам и менее.

```
Set Map Layer 1 Label Visibility Zoom (0, 2) Units "km"
```

**Auto** Управляет автоматическим подписыванием. Предложение **Auto Off** отключает автоматические подписи, но остаются созданные вручную.

**Overlap** Управляет режимом, разрешающим или запрещающим MapInfo рисовать пересекающиеся подписи. Чтобы избежать наложение подписей установите **Overlap Off**.

**PartialSegments** Управляет способностью MapInfo подписывать объекты, когда центроид вне зоны видимости окна карты. Если Вы определили **PartialSegments On** (это соответствует установке флажка **Подписывание сегментов линий** в MI, то MapInfo подпишет видимые

части объектов. Если Вы определили **PartialSegments Off**, объекты будут подписаны только если их центроид находится в окне Карты. В версии 7.0 эта возможность стала доступна для всех типов объектов. В версиях до 7.0 можно было подписывать только линейные объекты.

**Duplicates** Управляет режимом, разрешающим или запрещающим MapInfo дублировать подписи. Чтобы избежать дублирования подписей укажите **Duplicates Off**.

**Max number\_of\_labels** Устанавливает максимальное число подписей, которое MapInfo может показать на этом слое. По умолчанию лимита нет.

**Offset offset\_amount** Задаёт отступ подписи от центроида. Параметр *offset\_amount* может принимать значения от 0 до 50 шрифтовых точек. Если задать **Offset 0**, то подпись будет примыкать к центроиду. Если задать **Offset 10**, то подпись будет располагаться на 10 точек в сторону. Отступ будет игнорирован, если расположение подписи будет задано в центре объекта (**Position Center**).

Следующий оператор задаёт расположение подписей справа и на 10 точек в сторону от центроида:

```
Set Map Layer 1 Label Overlap On Position Right Offset 10
```

**Default** Восстанавливает все подписи на слое и удаляет все внесенные вручную подписи и изменения в автоматических. Следующим оператором удаляются все отредактированные подписи с верхнего слоя окна Карты, восстанавливая стандартные подписи:

```
Set Map Layer 1 Label Default
```

Предложение **Object** позволяет задать отдельную подпись для индивидуального объекта. Например, если Вы изменили одну подпись в MapInfo и сохранили Рабочий Набор, то он будет содержать оператор **Set Map** со столькоими предложениями **Object**, сколько подписей было индивидуально изменено.

Пример использования предложения **Object** Вы можете увидеть, если откроете такой Рабочий Набор в любом текстовом редакторе.

### Настройки отдельного слоя, имеющие постоянный эффект

Предложение **Default Zoom** воздействует на таблицу, а не на Карту. Оно используется для установки значений стандартного увеличения и центральной точки для таблицы такими, какие они сейчас в окне Карты.

Каждая таблица, к которой присоединена геоинформация, имеет стандартное увеличение и центральную точку. Эти значения применяются для представления открываемой Карты.

Если в операторе **Set Map...Layer** содержится предложение **Default Zoom**, MapInfo помещает в таблицу текущие значения увеличения и координат центральной точки. Например, следующий оператор изменяет размер и центр окна для таблицы, изображенной на первом слое:

```
Set Map Layer 1 Label Default
```

**Default Zoom** срабатывает сразу, не дожидаясь операции сохранения таблицы.

## Настройка режима совмещения

После использования значения **Set Map Move Nodes**, для карты используется пользовательская настройка. Если окно Карты имеет свои собственные настройки, в частности режима совмещения, то общие настройки не будут использоваться. Общие настройки (стандартные), будут также применяться ко всем новым окнам. Установки для существующих окон могут быть сделаны с помощью оператора **Set Map Move Nodes** языка MapBasic.

### Пример:

Программа открывает две таблицы и показывает их в окне Карты. Затем оператор **Set Map** изменяет окно.

```
Open Table "world"
Open Table "cust1993"
As customersMap From customers,
worldSet Map Center (100, 40) 'центрирование карты США
Zoom 4000 Units
"mi" 'показ континентальной части США
Preserve Zoom 'сохранять размер
КартыDisplay Position 'показывать в углу широту и долготу
Layer 1Editable
On Layer 2 Selectable OffDisplay Global Global Brush (2, 255, 65535)
```

### См. также:

[Оператор Add Map](#), [Функция LayerInfo\( \)](#), [Оператор Map](#), [Функция MapperInfo\( \)](#),  
[Оператор Remove Map](#), [Оператор Set Window](#)

## Оператор Set Map3D

### Назначение

Изменяет настройки существующего окна 3D-Карты.

### Синтаксис

#### Set Map3D

```
[ Window window_id ]
[ Camera [ Zoom factor | Pitch angle | Roll angle | Yaw angle |
  Elevation angle Position (x,y,z) | FocalPoint (x,y,z) ] ]
[ Orientation ( vu_1, vu_2, vu_3, vpn_1, vpn_2, vpn_3,
  clip_near, clip_far ) ] ]
[ Light [ Position ( x, y, z | Color lightcolor ) ] ]
[ Resolution ( res_x, res_y ) ]
[ Scale grid_scale ]
[ Background backgroundcolor ]
[ Refresh ]
```

*mapper\_creation\_string* - указывает командную строку, которая создает изображение по гриду.

*factor* - указывает коэффициент увеличения.

*angle* - это угол, измеряемый в градусах. Горизонтальный угол в диалоге может изменяться от 0 до 360 и вращает карту вокруг центральной точки грида. Вертикальный угол в диалоге изменяется от 0 до 90 и измеряет вращение в вертикальной плоскости прямо от стартовой точки прямо над картой.

*res\_x, res\_y* - количество образцов по осям x и y. Эти значения могут увеличиваться вплоть до максимального разрешения грида. Разрешение может увеличиваться вплоть до максимальных координат x,y. Если грид имеет разрешение 200x200 то и разрешение в окне карты не будет больше чем это значение 200x200. Вы не можете увеличивать разрешение грида, можно изменить только разрешение его изображения.

*grid\_scale* - коэффициент масштабирования грида в направлении оси z. Значение >1 увеличит топологию в направлении оси z, а значение < 1 уменьшит.

*backgroundcolor* - это цвет, используемый для фона и он определяется функцией **Функция RGB( )**.

### Описание

Оператор **Set Map3D** изменяет настройки уже созданной 3D-карты. Если исходная таблица, из которой была создана 3D-карта, была изменена добавлением подписей или изменением геометрии, **Refresh** отражает изменения и пересоздает 3D-карту с учетом этих изменений.

**Camera** - определяет позицию и ориентацию камеры.

**Pitch** регулирует поворот камеры вокруг оси X

**Roll** регулирует поворот камеры вокруг оси Z

**Yaw** регулирует поворот камеры вокруг оси Y

**Elevation** регулирует поворот камеры вокруг оси X относительно фокальной точки камеры

**Position** - регулирует позицию камеры/источника света

**FocalPoint** регулирует позицию фокуса камеры/источника света

**Orientation** регулирует позицию камеры ViewUp (*vu\_1, vu\_2, vu\_3*), ViewPlane Normal (*vpn\_1, vpn\_2, vpn\_3*) и Clipping Range (*clip\_near, clip\_far*). (Используется для повышения устойчивости точки наблюдения.)

**Resolution** - это разрешение в направлении X и Y. Эти значения могут увеличиваться вплоть до максимального разрешения грида. Разрешение может увеличиваться вплоть до максимальных координат x,y. Если грид имеет разрешение 200x200 то и разрешение в окне карты не будет больше чем это значение 200x200. Вы не можете увеличивать разрешение грида, можно изменить только разрешение его изображения.

**Units** - определяет единицы измерения грида (третьей компоненты). Не указывайте единицы, например, для температурного поля или плотности. Эту настройку надо делать к моменту создания грида. Если со значениями грида связаны единицы измерения, их следует указывать при создании 3D-карты. Нельзя изменить единицы измерения позднее, используя оператор **Set Map3D** или диалог Настройки.

**Refresh** создает новую текстуру из исходной таблицы.

**Пример:**

```
Dim win3D as Integer
Create Map3D Resolution(75,75) Resolution(100,100) Scale 2 Background
RGB(255,0,0)
win3D = FrontWindow( )
Set Map3D Window win3D Resolution(150,100) Scale 0.75 Background
RGB(255,255,0)
Changes the original 3DMap window's resolution in the X and Y, the scale
to de-emphasize the grid in the Z direction (< 1) and change the
background color to yellow.
```

**См. также:**

**Оператор Create Map3D, Функция Map3DInfo( )**

---

## Оператор Set Next Document

**Назначение**

Переподчиняет документальное окно в MapInfo (например, окно Карты становится подчиненным или порожденным окном приложения на Visual Basic).

**Синтаксис**

```
Set Next Document
{ Parent HWND | Style style_flag | Parent HWND Style style_flag }
```

*HWND* – целое число типа Integer, уникальный номер порождающего окна;

*style\_flag* – целочисленный код (смотрите таблицу ниже), задающий стиль окна.

**Описание**

Этот оператор используется в приложениях интегрированной картографии. Более подробная информации об интегрированной картографии содержится в *Руководстве пользователя MapBasic*.

Чтобы переподчинить окно MapInfo выполните оператор **Set Next Document**, а за ним любой из создающих окно операторов: **Оператор Map**, **Оператор Browse**, **Оператор Graph**, **Оператор Layout**, или **Оператор Create Legend**.

Предложение **Parent** используется для задания существующего окна, которое будет считаться порождающим по отношению к окну MapInfo, которое Вы собираетесь создать. Предложение **Style** определяет стиль окна. Если Вы создаете документальное окно (такое как Карту), то включите оба предложения.

Значение параметра *style\_flag* должно быть равно коду из следующей таблицы.

| Значение style_flag         | Воздействие на следующее окно документа:  |
|-----------------------------|---|
| WIN_STYLE_CHILD             | Следующее окно создается подчиненным. (Значение кода 1.)  |
| WIN_STYLE_POPUP             | Следующее создаваемое окно имеет стиль роруп и строка заголовка показывается в половину обычной ее высоты. (Значение кода 3.)   |
| WIN_STYLE_POPUP_FULLCAPTION | Следующее создаваемое окно имеет стиль роруп и создается с обычной строкой заголовка. (Значение кода 2.)  |
| WIN_STYLE_STANDARD          | Этот код восстанавливает стиль окна до стандартного вида. (Значение кода 0.) Если Вы выполнили оператор <b>Set Next Document Style 1</b> , а затем раздумали назначать окну подчиненный стиль, то восстановить стиль окна можно оператором <b>Set Next Document Style 0</b> . |

Установка стиля и подчиненности окна срабатывает только для следующего создаваемого окна. После того, как оно создается и к нему будут применены стили и подчиненность, MapInfo восстанавливает стандартные режимы подчиненности и стиля. То есть каждое новое переподчиняемое окно требует предварительного срабатывания оператора **Set Next Document**.

**Внимание:** Оператор **Оператор Create ButtonPad** переустанавливает режимы подчиненности и стиля, однако новые инструментальные панели не переподчиняются.

Этот оператор переподчиняет окна документов. Чтобы переподчинить окна диалогов, используйте оператор **Оператор Set Application Window**. Чтобы переподчинить специальные окна типа “Информация”, используйте оператор **Оператор Set Window**.

### Пример:

Программа LEGENDS.MB использует следующие операторы для создания окна Легенды для Карты:

```
Dim win As Integer
win = FrontWindow( )
...
Set Next Document
    Parent WindowInfo(win, WIN_INFO_WND)
    Style 1
Create Legend From Window win
```

**См. также:**

**Оператор Set Application Window, Оператор Set Window**

## Оператор Set Paper Units

### Назначение

Устанавливает единицы измерения, описывающие размеры и положения окон на экране.

### Синтаксис

**Set Paper Units** *unit*

*unit* – строка с именем единицы линейных измерений (например, "cm" – сантиметры).

### Описание

Оператор **Set Paper Units** назначает так называемые "бумажные" единицы, т.е. единицы линейных измерений на экране, которые используются по умолчанию операторами MapBasic при определении размеров и положений окон MapInfo на экране или объектов на печатном листе. Если оператор **Set Paper Units** не участвовал в программе, то по умолчанию используются дюймы ("in").

Некоторые операторы MapBasic (такие, как **Оператор Set Window**) включают предложения **Position**, **Width** и **Height**, с помощью которых устанавливают положение, ширину и высоту окон. Если эти предложения не содержат предложение Units, то численные параметры задают размеры в единицах, объявленных ранее оператором Set Paper Units, или в дюймах. Например, следующий оператор **Оператор Set Window** Set Window Width 5 устанавливает ширину окна на экране. Новая ширина окна зависит от используемых бумажных единиц. Если ранее MapBasic использовал дюймы в качестве бумажных единиц измерения, оператор **Оператор Set Window** задаст Карту шириной в пять дюймов.

Если единицами измерения ранее были установлены сантиметры, то оператор **Оператор Set Map** задаст Карту шириной в пять сантиметров.

Заметим, что "бумажные" единицы, устанавливаемые MapBasic, являются внутренним атрибутом и не доступны пользователю MapInfo.

В следующей таблице в первой колонке представлены значения параметра unit, которые могут использоваться в операторе:

| Название единицы измерения | Единица измерения листа |
|----------------------------|-------------------------|
| "cm"                       | сантиметры              |
| "in"                       | дюймы                   |
| "mm"                       | миллиметры              |
| "pt"                       | точки                   |
| "pica"                     | пайки                   |

См. также:

[Оператор Set Area Units](#), [Оператор Set Distance Units](#)

---

## Оператор Set PrismMap

### Назначение

Изменяет настройки существующего окна Карты-призмы.

### Синтаксис

**Set PrismMap**

```
[Window window_id ]  
[ Camera [ Zoom factor | Pitch angle | Roll angle | Yaw angle |  
    Elevation angle Position (x,y,z) | FocalPoint (x,y,z) ] ]  
[ Orientation ( vu_1, vu_2, vu_3, vpn_1, vpn_2, vpn_3,  
    clip_near, clip_far )]]  
[ Light [ Position (x,y,z) | Color lightcolor ] ]  
[ Scale grid_scale ]  
[ Background backgroundcolor ]  
[ Label With infotips_expr ]  
[ Refresh ]
```

*window\_id* - это идентификатор окна карты, которое содержит слой поверхности. Если такой слой не найден, появится сообщение об ошибке.

*mapper\_creation\_string* - указывает командную строку, которая создает изображение по гриду.

**Camera** - определяет позицию и ориентацию камеры.

*angle* - это угол, измеряемый в градусах. Горизонтальный угол в диалоге может изменяться от 0 до 360 и вращает карту вокруг центральной точки грида. Вертикальный угол в диалоге изменяется от 0 до 90 и измеряет вращение в вертикальной плоскости прямо от стартовой точки прямо над картой.

**Pitch** регулирует поворот камеры вокруг оси X

**Roll** регулирует поворот камеры вокруг оси Z

**Yaw** регулирует поворот камеры вокруг оси Y

**Elevation** регулирует поворот камеры вокруг оси X относительно фокальной точки камеры

**Position** - регулирует позицию камеры/источника света

**FocalPoint** регулирует позицию фокуса камеры/источника света

**Orientation** регулирует позицию камеры ViewUp (*vu\_1, vu\_2, vu\_3*), ViewPlane Normal (*vpn\_1, vpn\_2, vpn\_3*) и Clipping Range (*clip\_near, clip\_far*). (Используется для повышения устойчивости точки наблюдения.)

*backgroundcolor* - это цвет, используемый для фона и он определяется функцией [Функция RGB\(\)](#).

*infotips\_expr* - выражение, используемое для всплывающей подсказки InfoTips.



**Refresh** создает новую текстуру из исходной таблицы.

### Описание

Оператор **Set PrismMap** изменяет настройки уже созданной карты-призмы.

### Пример:

Здесь мы изменим разрешение окна карты-призмы по осям X и Y, масштаб по оси Z ( $< 1$ ) и изменим цвет фона на желтый.

```
Dim win3D as Integer
Create PrismMap Resolution(75,75) Resolution(100,100) Scale 2 Background
RGB(255,0,0)
win3D = FrontWindow( )
Set PrismMap Window win3D Resolution(150,100) Scale 0.75 Background
RGB(255,255,0)
```

**См. также:**

**Оператор Create PrismMap, Функция PrismMapInfo( )**

---

## Оператор Set ProgressBars

### Назначение

Показывает или скрывает диалог-индикатор выполнения процесса.

### Синтаксис

```
Set ProgressBars { On | Off }
```

### Описание

Некоторые операторы MapBasic, такие как Create Object As Buffer, автоматически выводят диалог, показывающий процент выполнения и кнопку **Отмена**. Оператор **Set ProgressBars Off** используется для подавления диалога-индикатора выполнения процесса. Если диалог не выводится на экран, то пользователь лишается возможности отменить выполнение процесса кнопкой "**Отмена**". Оператор **Set ProgressBars On** возобновляет вывод диалогов-индикаторов на экран.

Если оператор **Set ProgressBars Off** выполняется из MapBasic-программы (MBX-файла), то отключается только порожденный MBX-файлом диалог-индикатор. Те диалоги-индикаторы, которые иллюстрируют действия пользователя, не отключаются. Кроме этого, оператор **Оператор Run Menu Command** может игнорировать запрещение показа диалогов-индикаторов, потому что **Оператор Run Menu Command** симулирует выполнение команд меню пользователем.

Чтобы отключить показ диалога-индикатора, появляющегося при выполнении оператора **Оператор Run Menu Command**, выполните оператор **Set ProgressBars Off** в окне MapBasic (или пошлите эту команду в MapInfo через механизмы OLE Automation или DDE).

Если приложение свертывает MapInfo в иконку (например, оператором Set Window MapInfo Min), то диалог-индикатор выполнения нужно отключать, так как в этом случае диалог-индикатор “зависает” до тех пор, пока окно MapInfo не раскроется снова. Если Вы скроете диалог “Минуточку”, работа по-прежнему будет выполняться, даже если окно MapInfo минимизировано.

**См. также:**

**Оператор ProgressBar, Оператор Run Menu Command**

---

## Оператор Set Redistricter

### Назначение

Изменяет состав и характеристики районов.

### Синтаксис 1

```
Set Redistricter districts_table  
  [ Change district_name  
    [ To new_district_name ] [ Pen... ] [ Brush... ] [ Symbol... ] ]  
  [ Add new_district_name [ Pen... ] [ Brush... ] [ Symbol... ] ]  
  [ Remove district_name ]
```

### Синтаксис 2

```
Set Redistricter districts_table Order { "Alpha" | "MRU" | "Unordered" }
```

### Синтаксис 3

```
Set Redistricter districts_table Percentage from { column | row }
```

*districts\_table* – имя таблицы районов (например, Districts);

*district\_name* – строка с именем открытого окна Районирование;

*new\_district\_name* – строка с новым именем для открытого окна Районирование(используется при добавлении или переименовании района).

**Pen...** - это первое слово оператора **Предложение Pen**, например, **Pen MakePen** ( *width*, *pattern*, *color* ).

**Brush...** - это первое слово оператора **Предложение Brush**, например, **Brush MakeBrush** ( *pattern*, *forecolor*, *backcolor* ).

**Symbol...** - это первое слово оператора **Предложение Symbol**, например, **Symbol MakeSymbol** ( *shape*, *color*, *size* ).

### Описание

Оператор **Set Redistricter** изменяет состав районов. Процедура районирования начинается оператором **Оператор Create Redistricter**. Правила работы с районами подробно описаны в документации MapInfo.

Первый вариант синтаксиса используется для добавления, роспуска и изменения района. Предложение **Change** изменяет название и/или стиль оформления районов. Предложение **Add** добавляет новый район, а предложение **Remove** распускает существующий, при этом освободившиеся объекты переходят в район "остальные".

Параметры *district\_name* и *new\_district\_name* должны быть строковыми константами или выражениями, даже если колонка Районы численная. Например, если район представлен числом 33, то параметр должен задаваться строкой "33".

Для сортировки строк в окне Районирование используется второй вариант синтаксиса оператора. Ключевое слово Alpha задает сортировку в алфавитном порядке. Ключевое слово MRU используется, если Вы хотите, чтобы последняя группа, с которой Вы работали, автоматически становилась первой в Списке Районов. В режиме Unordered все новые районы добавляются в конец Списка.

## Примеры

В ходе процедуры районирования следующий оператор создает новый район:

```
Set Redistricter Districts
  Add "NorthWest" Brush MakeBrush(2, 255, 0)
```

Следующий оператор переименовывает район "NE" в "Северо-восток" в ходе процедуры районирования. Изначально любые строки из района "NE" содержат "NE" в соответствующей колонке. После применения оператора **Set Redistricter... Change**, запись в каждой из тех строк меняется на "Северо-восток".

```
Set Redistricter Districts Change "NE" To "Северо-восток"
```

Следующий оператор удаляет район "NorthWest" из таблицы DISTRICTS:

```
Set Redistricter Districts
  Remove "NorthWest"
```

Следующий оператор задает упорядочивание строк в Списке Районов по их использованию:

```
Set Redistricter Districts
  Order "MRU"
```

**См. также:**

**Оператор Create Redistricter**

## Оператор Set Resolution

### Назначение

Устанавливает параметр графического разрешения для операций изменения типа объекта. Эта характеристика влияет на количество узлов в объекте, полученном преобразованием типа объекта.

### Синтаксис

```
Set Resolution node_limit
```

*node\_limit* – целое число типа SmallInt от 2 до 32 762 включительно; по умолчанию 100.

### Описание

По умолчанию MapInfo создает 100 узлов на окружности или дуге при преобразовании их в область и полилинию. Оператор **Set Resolution** устанавливает число узлов для преобразования окружности в область. Прирост значения разрешения приводит к более гладким результатам.

Оператор **Set Resolution** влияет на результаты таких команд, как **Объекты > Превратить в области** и **Объекты > Превратить в полилинии**. Значение разрешения влияет также на результаты некоторых операторов и функций MapBasic, таких как **Функция ConvertToRegion( )** и **Функция ConvertToPline( )**. Кроме этого, от значения разрешения зависят результаты операций, в которых конвертирование производится автоматически (например, Objects Split, Combine).

Установка оператора **Set Resolution** не влияет на создание буферной области. Create Object As Buffer и **Функция Buffer( )** имеют обязательный параметр, явно задающий разрешение для создания области.

См. также:

**Функция ConvertToPline( )**, **Функция ConvertToRegion( )**

---

## Оператор Set Shade

### Назначение

Изменяет тематический слой Карты.

### Синтаксис

```
Set Shade  
[ Window window_id ] { map_layer_id | "table ( theme_layer_id )" }  
[ Style Replace { On | Off } ]  
...
```

*window\_id* – идентификатор окна.

*map\_layer\_id* – короткое целое число (SmallInt), задающее номер тематического слоя.

*table* – имя таблицы, на которой основывается тематический слой.

*theme\_layer\_id* – короткое целое число (SmallInt) от 1 и больше, задающее номер изменяемого тематического слоя (например, 1 используется для первого созданного слоя).

### Описание

После того, как оператор **Оператор Shade** создаст тематический слой карты, Вы с помощью оператора **Set Shade** можете изменять этот слой. Оператор **Set Shade** выполняет те же действия, что и команда в MapInfo **Карта > Настройка условного выделения**. Синтаксис оператора **Set Shade** такой же, как у оператора **Оператор Shade**, за исключением первых ключевых слов и параметров, задающих слой тематического выделения. Оператор **Set Shade** может идентифицировать слой по его номеру, как в следующем примере:

```
Set Shade
  Window i_map_winid
  2
  With Num_Hh_90
  Graduated 0.0:0 11000000:24 Vary Size By "SQRT"
```

Или по имени таблицы, на данных которой основано тематическое выделение (в скобках указывается номер созданного тематического слоя):

```
Set Shade
  Window i_map_winid
  "States(1)"
  With Num_Hh_90
  Graduated 0.0:0 11000000:24 Vary Size By "SQRT"
```

"RUSSIA(1)" означает первый слой из тематических, основанный на данных таблицы RUSSIA.

**Style Replace On** (по умолчанию) запрещает отображение слоёв под тематическим слоем.

**Style Replace Off** предписывает отображать слои под тематическим слоем, что позволяет создавать многовариантные прозрачные тематические слои.

**Style Replace On** - это режим по умолчанию, обеспечивающий обратную совместимость с существующими настройками, так что по умолчанию нижележащие слои не отображаются.

**См. также:**

**Оператор Shade**

## Оператор Set Style

### Назначение

Изменяет текущие стили Pen, Brush, Symbol или Font.

### Синтаксис

```
Set Style
  { Brush... | Font... | Pen... |
    BorderPen | LinePen | Symbol... }
```

Предложение **Предложение Brush** определяет стиль заливки.

Предложение **Предложение Font** определяет стиль текста.

**Предложение Pen** определяет стиль линии.

**Предложение Symbol** определяет стиль символа.

**BorderPen** использует предложение **Предложение Pen**, определяющее стиль границы региона.

**LinePen** использует предложение **Предложение Pen**, определяющее стиль линии.

### Описание

Оператор **Set Style** изменяет текущие стили **Pen**, **Brush**, **Symbol**, или **Font**.

Предложение **Предложение Pen** устанавливает оба стиля - для линий и для границ региона. Чтобы установить их отдельно, используйте предложение **LinePen** для установки стиля линий и предложение **BorderPen** для установки стиля границы региона. Когда пользователь рисует новый графический объект в окне Карты или Отчета, MapInfo Professional создает объект используя текущие стили **Font**, **Pen**, **Brush** и/или **Symbol**. Более подробная информация о параметрах **Pen**, **Brush**, **Symbol** и **Font** обсуждается в описаниях предложений **Предложение Brush on page 117**, **Предложение Font on page 317**, **Предложение Pen on page 492**, и **Предложение Symbol on page 725**

### Пример:

Пример изменения стилей **Brush**, **Symbol**, и **Font**:

```
Include "mapbasic.def"
Set Style Brush MakeBrush(64, CYAN, BLUE)
Set Style Symbol MakeSymbol( 9, BLUE, 14)
Set Style Font MakeFont("Arial", 1, 14, BLACK,WHITE)
```

Пример изменения стиля **Pen**:

В этом примере линия и граница региона красного цвета.

```
Include "mapbasic.def"Set Style Pen MakePen(3, 9, RED)
```

Пример изменения стиля **LinePen** и **BorderPen**:

В этом примере линия красная а граница региона зеленая.

```
Include "mapbasic.def"Set Style LinePen MakePen(6, 77, RED)Set Style
BorderPen MakePen(6, 77, GREEN)
```

### См. также:

**Функция CurrentBrush( )**, **Функция CurrentFont( )**, **Функция CurrentPen( )**, **Функция CurrentSymbol( )**, **Функция MakeBrush( )**, **Функция MakeFont( )**, **Функция MakePen( )**, **Функция MakeSymbol( )**, **Функция RGB( )**

## Оператор Set Table Datum

### Назначение

Записывает информацию о топоцентрической проекции (датуре) в MAP-файл, включая номер эллипсоида, 3 параметра сдвига, 3 параметра поворота и коэффициент масштабирования. Так как некоторые топоцентрические проекции (датуны) совпадают, то этот оператор хранит номер топоцентрической проекции (датура) вместе с параметрами в памяти, а затем записывает их в MAP-файл.

### Синтаксис

*table\_name datum datum\_number*

См. также:

[Оператор Set Datum Transform Version](#)

## Оператор Set Table

### Назначение

Изменяет некоторые режимы работы с открытой таблицей.

### Синтаксис

```
Set Table tablename
[ FastEdit { On | Off } ]
[ Undo { On | Off } ]
[ ReadOnly ]
[ Seamless { On | Off } [ Preserve ] ]
[ UserMap { On | Off } ]
[ UserBrowse { On | Off } ]
[ UserClose { On | Off } ]
[ UserEdit { On | Off } ]
[ UserRemoveMap { On | Off } ]
[ UserDisplayMap { On | Off } ]
```

*table* - строка, указывающая имя таблицы.

### Описание

Оператор **Set Table** устанавливает, можно ли изменять данные таблицы и как это сделать. Оператор **Set Table** может задать открытой таблице режим "только чтение", и пользователь не сможет внести в эту таблицу никакие изменения. Оператор **Set Table** может активизировать и выключать специальный режим редактирования, который выключает механизм защиты исходных данных таблицы ради ускорения выполнения действий правки.

### Режим FastEdit

Обычно, всякий раз, когда таблица редактируется (независимо, кем – пользователем или прикладной программой), то MapInfo не записывает изменения непосредственно в таблицу. Вместо этого MapInfo помещает информацию об изменениях во временный файл, который называется файлом транзакций. Записывая изменения в файл транзакций вместо того, чтобы вносить их непосредственно в таблицу, MapInfo дает возможность пользователю удалить эти изменения (например, командой **Файл > Восстановить** в MapInfo).

Если Вы выполнили оператор **Set Table** с предложением **FastEdit On**, то MapInfo будет записывать изменения непосредственно в таблицу, минуя файл транзакций. Операции правки таблицы в таком режиме будут производиться быстрее.

Пока включен режим **FastEdit**, таблица изменяется немедленно и не требуется выполнение оператора **Оператор Commit Table** для фиксации изменений на диске. Восстановить исходное состояние таблицы командами **Файл > Закрывать таблицу** и **Файл > Восстановить** нельзя.

Режим **FastEdit** может устанавливаться для нормальных, базовых таблиц. Вы не можете установить этот режим для временных таблиц, таких как Запрос1. Вы не можете установить режим редактирования **FastEdit** для таблиц, которые уже имеют несохраненные на диск изменения. Вы не можете установить режим редактирования **FastEdit** для связанных таблиц.

**ВНИМАНИЕ:** Пока открытая таблица редактируется в режиме **FastEdit**, другие пользователи сети не могут ее открыть. После того, как Вы выполнили все изменения в режиме **FastEdit**, выполните оператор **Оператор Commit Table** или **Оператор Rollback** для установки состояния таблицы, чтобы она была доступна другим пользователям сети.

Ключевое слово **ReadOnly** в операторе включает режим "только чтение" для таблицы tablename, так что пользователь не сможет ее изменять вплоть до конца сеанса в MapInfo. Оператор **Set Table** не может выключить режим "только чтение". Этот режим Вы также можете установить для таблицы при ее открытии оператором **Оператор Open Table**.

Предложение **Undo On** устанавливает режим работы с таблицей, при котором MapInfo запоминает всю информацию об изменениях, позволяя пользователю применять команду **Правка > Отменить**. Если Вы использовали в операторе **Set Table** предложение **Undo Off**, то MapInfo не запоминает информацию о последних изменениях в таблице. В последнем режиме экономятся ресурсы Вашего компьютера, а операции редактирования таблицы выполняются существенно быстрее.

### Управление сшитыми таблицами

MapInfo 4.0 поддерживает новый тип таблиц – сшитые таблицы. В сшитой таблице группируются несколько таблиц в единое целое. Концепция сшитых таблиц подробно описана в документации MapInfo.

Предложение **Seamless** устанавливает или отменяет атрибут сшитости для таблицы. Режим **Seamless Off** открывает таблицу, входящую в группу сшитых, для редактирования. Режим **Seamless On** восстанавливает атрибут сшитости. Ключевое слово **Preserve** сохраняет режим; то есть MapInfo записывает режим в таблицу. Без слова **Preserve** смена режима действует только до конца сеанса работы.



Предложения **User...** позволяют выборочно запрещать пользователю применять определенные операции к таблице. Это полезно, если Вы хотите запретить пользователю открывать, закрывать или изменять определенные таблицы или окна.

Эти предложения ограждают таблицы только от действий пользователя, но не от операторов программ MapBasic.

**Внимание:** Эти предложения не действуют на Косметический слой.

| Пример:            | Эффект   |
|--------------------|--|
| UserMap Off        | Таблица не появится в окне диалоге “Новое окно Карты” и “Добавить слой”.   |
| UserBrowse Off     | Таблица не появится в окне диалоге “Новое окно Списка”.  |
| UserClose Off      | Таблица не появится в окне диалоге “Заккрыть таблицу”.   |
| UserEdit Off       | Пользователь не может редактировать таблицу: окна Списка и Информации не редактируются и соответствующий слой Карты невозможно сделать изменяемым. |
| UserRemoveMap Off  | Когда эта таблица появляется в диалоге “Управление слоями”, кнопка “Удалить” для нее неактивна.  |
| UserDisplayMap Off | Когда эта таблица появляется в диалоге “Управление слоями”, флажок видимости для нее сброшен и отключен.   |

**Пример:**

Следующий пример не допускает таблицу World в диалог закрытия таблиц.

```
Set Table World UserClose Off
```

**См. также:**

**Функция TableInfo( )**

## Оператор Set Target

**Назначение**

Назначает изменяемый объект или объекты или отменяет назначение.

**Синтаксис**

```
Set Target { On | Off }
```

### Описание

Оператор **Set Target** управляет выбором изменяемого объекта на Карте. Действие оператора аналогично действию команд в **MapInfo Объекты > Выбрать изменяемый объект** и **Объекты > Освободить изменяемый объект**. Некоторые операции в MapInfo предваряются выбором объектов, которые надо изменить. Например, Вы должны сначала назначить изменяемым объект перед тем, как выполнить оператор **Оператор Objects Split**. Правила назначения изменяемого объекта описаны в документации MapInfo.

Оператор **Set Target On** соответствует команде **Объекты > Выбрать изменяемый объект**. После выполнения оператора или команды объекты, выбранные на данный момент, становятся изменяемыми. Если не выбрано ни одного объекта, оператор генерирует ошибку.

Оператор **Set Target Off** соответствует команде **Объекты > Освободить изменяемый объект**.

См. также:

**Оператор Objects Combine, Оператор Objects Erase, Оператор Objects Intersect, Оператор Objects Overlay, Оператор Objects Split**

---

## Оператор Set Window

### Назначение

Изменяет состояние, размер и положение окна на экране, управляет принтером, размером бумаги и полями.

### Синтаксис

```
Set Window window_id
[ Position ( x, y ) [ Units paper_units ] ]
[ Width win_width [ Units paper_units ] ]
[ Height win_height [ Units paper_units ] ]
[ Font... ]
[ Min | Max | Restore ]
[ Front ]
[ Title { new_title | Default } ]
[ Help [ { File help_file | File Default | Off } [ Permanent ] ]
  [ Contents ] [ ID context_ID ] [ { Show | Hide } ] ]
[ Printer { Default | Name printer_name }
  [ Orientation { Portrait | Landscape } ]
  [ Copies number ]
  [ Papersize number ]
  [ Border { On | Off } ]
  [ TrueColor { On | Off } ]
  [ Dither { Halftone | ErrorDiffusion } ]
  [ Method { Device | Emf } ]
  [ Transparency
    [ Raster { Device | ROP } ]
    [ Vector { Device | ROP } ] ]
  [ Margins
```

```

    [ Left d1 ] [ Right d2 ] [ Top d3 ] [ Bottom d4 ]
    Units paper_units ] } ]
[ Export { Default |
[ Border { On | Off } ]
[ TrueColor { On | Off } ]
[ Dither { Halftone | ErrorDiffusion } ]
[ Transparency
    [ Raster { Device | ROP } ]
    [ Vector { Device | ROP } ] ]
[ Scale Patterns { On | Off } ]
[ Antialiasing { On | Off } ]
[ Threshold threshold_value ]
[ MaskSize size_value ]
[ Filter filter_value ]
]
[ ScrollBars { On | Off } ]
[ Autoscroll { On | Off } ]
[ Parent HWND ]
[ ReadOnly | Default Access ]
[ Table table_name Rec record_number ]
[ Show | Hide ]
[ Smart Pan { On | Off } ]
[ SysMenuClose { On | Off } ]
[ Snap [ Mode { On | Off } ] [ Threshold { pixel_tolerance | Default } ]

```

*window\_id* - целочисленный идентификатор окна или имя специального (например, Statistics)

*x* - расстояние от верхнего края рабочего поля окна MapInfo до верхнего края перемещаемого окна

*x* - расстояние от левого края рабочего поля в окне MapInfo до левого края перемещаемого окна

*paper\_units* – строка, представляющая "бумажные" единицы измерения (например, "cm" для сантиметров).

Предложение **Предложение Font** определяет стиль шрифта текстовых объектов.

*win\_width* - ширина окна

*win\_height* - высота окна

*new\_title* - строка, задающая новый заголовок окна

*help\_file* - имя файла Справочника (например, в Windows "FILENAME.HLP")

*context\_ID* - целочисленный идентификатор контекста Справочника для задания раздела

*printer\_name* - имя принтера. Принтер может быть локальным или сетевым.

*number* - количество копий, которое будет передано на печать.

*HWND* - целочисленный номер окна. Окно с номером *HWND* станет порождающим окном по отношению к окну Легенды, Статистики, Информации, Линейки или Сообщений.

*table\_name* - имя открытой таблицы для показа в окне сообщений.

*record\_number* - целое число типа Integer: значение от 1 и больше для показа определенной записи в окне Информации или 0 для показа сообщения "Нет записей".

**Printer** - будет указывать окно, предназначенное для печати.

**Export** - будет указывать окно, предназначенное для экспорта.

**Default** - будут использоваться стандартные настройки печати/экспорта.

**Name** *printer\_name* - определяет имя используемого принтера.

**Orientation Portrait** - (книжная) ориентирует бумагу для печати в книжной ориентации.

**Orientation Landscape** - (альбомная) ориентирует бумагу для печати в альбомной ориентации.

**Copies number** - указывает число копий для печати.

**Papersize number** - информация о размере бумаги для данного окна. Эти числа(номера) универсальные для всех принтеров под Windows. Например, 1 соответствует размеру Letter, 5 соответствует размеру Legal. Этот номер может быть найден в файле MapBasic под названием PaperSize.def. Некоторые драйверы принтера (например, крупные плоттеры) могут использовать собственную нумерацию для идентификации размера бумаги, отличающуюся от той, которая в файле "PaperSize.def". По этой причине, пользователи, использующие разные драйверы принтера, возможно не смогут правильно идентифицировать информацию о размере бумаги, хранящуюся в рабочем наборе. В этом случае будет установлено стандартный для этого принтера размер бумаги.

**Border** - определяет будет ли отображаться черная рамка вокруг окна при печати и экспорте.

**Truecolor** - определяет, будет ли создан 24-bit true color вывод, если это вообще возможно. Если **Truecolor** отключен, вывод будет с 256 цветами.

**Dither** - определяет какой метод растеризации будет использован, если надо отконвертировать 24-битное изображение в 256 цветов. Эта настройка используется при печати растров и файлов поверхностей. Растеризация произойдет, если **truecolor** отключен или если выводное устройство не может поддерживать 24-битные цвета.

**Method** - это новое ключевое слово и определяет, будет ли печать проведена непосредственной передачей файла на принтер, или MapInfo создаст промежуточный Windows Enhanced Metafile и только потом пошлет его на принтер. Ранние версии MapInfo Professional всегда посылали файл прямо на устройство. Новый метод позволяет печатать карты с растрами, чего не было раньше.

**Transparency Raster Internal** - удалено в версии 7.0; таким образом, можно работать с данными предыдущих версий баз сообщения об ошибке.

**Transparency Raster** - определяет, как полупрозрачные пиксели будут отображаться. Выберите **Device** или **ROP** в зависимости от драйвера принтера или формата экспортного файла. Сначала попробуйте оба способа, сравните и сделайте потом выбор.

**Transparency Raster ROP** - обозначает "Использовать метод ROP для отображения полупрозрачного растра". Для этой настройки можно воспользоваться командой (**Режим > Параметры вывода, Файл > Печать > Дополнительно и Файл > Экспорт окна > Дополнительно**). Если выбран **ROP**, то полупрозрачное изображение прорисовывается с использованием растровой операции (ROP), обрабатывающей полупрозрачные пиксели.

Этот метод используется для обработки полупрозрачного изображения на экране; однако он не всегда хорошо работает при печати. Проведите эксперимент и посмотрите, как драйвер принтера обрабатывает методом ROP. Если Вы экспортируете изображение, используя команду **Экспорт окна**, то лучше выбрать формат метафайла (EMF или WMF). Использование метода ROP допускает отображение любых данных под полупрозрачным слоем в их исходной форме. Например, векторные данные, находящиеся под полупрозрачными пикселями не будут растеризованы. В метафайлах метод ROP не будет прорисовывать любые данные в области растровых пикселей, также возможно отображение базового изображения.

**Transparency Raster Device** не будет осуществлять специальные преобразования при выводе растров и гридов, содержащих прозрачность. Изображение генерируется тем же методом, что применяется для отображения на экране. Могут возникнуть некоторые проблемы при генерации вывода.

**Transparency Vector Internal** будет делать специальную обработку когда выводится прозрачная заливки или прозрачные растровые символы.

**Transparency Vector Device** MapInfo будет делать специальную обработку когда выводится прозрачная заливки или прозрачные растровые символы. Могут возникнуть некоторые проблемы при генерации вывода.

**Margins User** Пользователь может установить поля для принтера в виде вещественных значений в требуемых единицах. Эти значения могут быть увеличены драйвером принтера если они меньше, чем физически допустимые поля для данного принтера.

**Antialiasing** - определяет используется ли сглаживание при экспорте раstra. **Antialiasing** не используется при экспорте раstra в форматы EMF и WMF.

**Threshold** - указывает значение поределяющее какие пиксели будут сглаживаться. Применение фильтра сглаживания к изображению связывает значения с каждым пикселем. Только пиксели со значением больше *threshold\_value* сглаживаются. Если значение *threshold\_value* установлено равным нулю, все пиксели сглаживаются. *threshold\_value* - значение в диапазоне от 0 до 255.

**MaskSize** - число определяющее размер маски сглаживания. Например, значение три, определяет маску сглаживания 3x3. Если пользователь укажет слишком высокое значение *size\_value*, результирующий растр может получиться слишком размытым.

**Filter** - указывает какой сглаживающий фильтр будет применён. В настоящее время MapInfo поддерживает 6 различных фильтров, приведенных в таблице ниже.

| Фильтр                             | Описание                                       |
|------------------------------------|--|
| FILTER_VERTICALLY_AND_HORIZONTALLY | Сглаживание раstra по вертикали и горизонтали. |
| FILTER_ALL DIRECTIONS_1            | Сглаживание раstra во всех направлениях        |

| Фильтр                  | Описание   |
|-------------------------|--|
| FILTER_ALL_DIRECTIONS_2 | Сглаживание во всех направлениях. Этот фильтр отличается от фильтра FILTER_ALL_DIRECTIONS_1 и даёт лучшие результаты при сглаживании текста. |
| FILTER_DIAGONALLY       | Диагональное сглаживание растра.   |
| FILTER_HORIZONTALLY     | Горизонтальное сглаживание растра  |
| FILTER_VERTICALLY       | Вертикальное сглаживание растра  |

### Описание

Оператор **Set Window** используется для изменения размеров и положения окна, шрифта в окне, заданном параметром `window_id`.

Целочисленное значение параметра идентификатора окна `window_id` можно получить, используя функции: **Функция FrontWindow( )** и **Функция WindowInfo( )**. Для использования оператора **Set Window** по отношению к специальным окнам, таким как "Статистика", можно использовать имена окон (например, Statistics) или имена кодов (например, WIN\_STATISTICS), определенных в файле стандартных определений MAPBASIC.DEF.

Следующая таблица содержит имена окон и кодов окон, которые можно использовать в качестве параметра `window_id`.

| Имя окна   | Описание окна   |
|------------|---|
| MapInfo    | Окно программы MapInfo Professional. Можно также задавать кодом: WIN_MAPINFO.   |
| MapBasic   | Окно MapBasic. Можно также задавать кодом: WIN_MAPBASIC.  |
| Help       | Окно Справочной системы. Можно также задавать кодом: WIN_HELP.  |
| Статистика | Окно "Статистика". Можно также задавать кодом: WIN_STATISTICS.  |
| Legend     | Окно "Легенда". Можно также задавать кодом: WIN_LEGEND.   |
| Информация | Окно "Информация" (которое открывается при использовании инструмента Информация). Можно также задавать кодом: WIN_INFO.   |
| Линейка    | Окно "Линейка" (которое открывается при использовании инструмента Линейка). Можно также задавать кодом: WIN_RULER.        |
| Message    | Окно "Сообщение", которое открывается, если использовать <b>Оператор Print</b> . Можно также задавать кодом: WIN_MESSAGE. |

Дополнительное предложение **Position** задает расположение окна на экране в рабочем наборе MI Pro. Верхний левый угол окна с рабочим набором имеет координаты 0, 0. Дополнительные предложения **Width** и **Height** задают размер окна. В параметре "Position" используются "бумажные" единицы измерения, такие, как "in" (дюймы) или "cm" (сантиметры)). MapBasic имеет по умолчанию установку в дюймах; программа MapBasic может менять единицы, используя **Оператор Set Paper Units**. Оператор **Set Window** может изменить текущие единицы измерения, для этого надо включить дополнительное подпредложение **Units** в предложения **Position**, **Width** и/или **Height**.

Если оператор включает в себя дополнительное слово **Max**, то окно будет maximизировано (это действует на весь рабочий набор). Если оператор включает в себя дополнительное ключевое слово **Min**, то окно будет минимизировано (окно исчезнет, останется только иконка в нижней части экрана). Если окно уже минимизировано или maximизировано, и если оператор включает добавочное ключевое слово **Restore**, то окно будет восстановлено в прежнем виде.

Если оператор включает в себя дополнительное ключевое слово **Front**, то MapBasic сделает окно активным; также будет установлен фокус окна. Окно станет активным сразу после щелчка мыши на заголовке окна.

Оператор может всегда указывать предложение **Position** или предложение **Front**, независимо от типа окна. Таким образом, некоторые предложения в операторе **Set Window** применяются только к определенным типам окон. Например, окно Линейка не может изменять свой размер, maximизироваться или минимизироваться.

Для смены заголовка окна включите дополнительное предложение **Title**. Заголовок окна приложения (главный заголовок "MapInfo") не может быть изменен иначе, чем при запуске runtime MI Pro.

Предложение **SysMenuClose** позволяет отменить команду Закрыть (Close) в системном меню (меню, появляющемся при щелчке в верхнем левом углу окна). Отключение команды Закрыть действует только на пользовательский интерфейс; программы MapBasic могут отключать команду Закрыть оператором **Оператор Close Window**. Следующий пример отключения команды Закрыть для активного окна:

```
Set Window FrontWindow( ) SysMenuClose Off
```

## Синтаксис предложения Help

Для управления окном Справки, задайте ключевое слово **Help** вместо целочисленного аргумента *window\_id*. Например, следующий оператор покажет раздел 23 из пользовательского файла Справки:

```
Set Window Help File "custom.hlp" ID 23
```

Предложение **File help\_file** показывает, какой файл справки активен. В Windows, это действие автоматически показывает окно справки (если Вы не включили слово **Hide**). Если задать **File Default**, то MI Pro будет использовать стандартную справку MapInfo Professional, но не прорисует на экране файл справки. MI Pro имеет только один установленный файл справки, который применяется ко всем запущенным приложениям MapBasic. Если одно предложение устанавливает текущий файл справки, то и другие приложения могут им пользоваться.

Предложение **Off** отключает справку MI Pro, так что при нажатии F1 в MI Pro эффекта не будет. Используйте предложение **Off** если Вы интегрируете функциональность MI Pro в другое приложение (например, в программу Visual Basic), если Вы хотите, чтобы пользователь не видел справки MI Pro. (Справка MI Pro содержит ссылки к позициям меню MI Pro, которые могут не иметь действия в программе Visual Basic.)

Предложение **Permanent** заставляет MapInfo всегда пользоваться Справочником *help\_file*, даже если пользователь нажал F1 в диалоге MapInfo. (В среде Windows если параметра **Permanent** нет, то MapInfo обращается к стандартному файлу Справки MAPINFOW.HLP как только пользователь нажмет F1 в диалоге MapInfo.) Эта установка действует до конца сеанса MapInfo или до первого оператора **Set Window Help File**.

Чтобы сразу открыть Справочник на нужном месте, задавайте слова **Contents** (для показа оглавления Справки) или **ID** (для показа нужной информации).

В комплект поставки языка MapBasic не включен компилятор Справочных файлов в формате Windows (.HLP). Более подробно Справочная система описана в *Руководстве пользователя MapBasic*.

### Синтаксис для окон Карт и Отчетов

Предложение **ScrollBars** применимо только к окнам Карт и управляет показом строки (полосы) прокрутки.

Предложение **Autoscroll** применимо окнам Карт и Отчетов. По умолчанию, режим автоматической прокрутки действует в Картах и Отчетах, т.е. при выполнении операции с нажатой кнопкой мыши в окне Карты и Отчета, содержимое окна автоматически сдвигается вслед за мышью при приближении ее указателя к краю окна. Чтобы отключить автоматическую прокрутку, задайте **Autoscroll Off**. Функция **Функция WindowInfo( )** поможет определить, в каких окнах действует режим автоматической прокрутки.

**Smart Pan** изменяет статус операции сдвига в окне. Когда **Smart Pan** включен для окна Карты или Отчета, то при сдвиге и прокрутке используется отключение растровой прорисовки для уменьшения бликов. Стандартное состояние для **Smart Pan** - отключенное.

Когда **Smart Pan** активизирован для окна Отчета, то перерисовка действует только при использовании инструмента Сдвиг.

Когда **Smart Pan** активизирован для окна Карты, то эффект будет зависеть от метода перемещения Карты. Инструмент Сдвиг автоматически прорисовывает область окна при использовании самого инструмента. Карта будет перемещаться медленнее, чем при отключенном режиме Smart Pan. Чем сложнее карта, тем медленнее она будет перемещаться. Прокрутка и автопрокрутка действует похоже на инструмент Сдвиг, но скорость прокрутки не влияет на точность сдвига. Когда команда **Set Map** используется для центрирования или сдвига изображения со включенной командой **Smart Redraw**, изменения окна Карты при прорисовке произойдут без бликования.

**Внимание:** Если растровый режим прорисовки изображения на экране отключен, то предложение **Smart Pan** в окне Карты ведет себя также как и в окне Отчета.



## Синтаксис для вспомогательных окон (Легенда, Линейка и другие)

Предложение **Parent**, позволяющее задать новое порождающее окно для окон легенды, Статистики, Информации, Линейки или Сообщений, действует только в Windows. Окно с номером *window\_id* становится рорир-окном, подчиненным окну с номером-указателем *HWND*.

**Внимание:** Переподчинение окна таким способом изменяет значение ID для этого окна. Чтобы снова подчинить окно первоначальному “родителю”, MapInfo, задайте ноль в качестве *HWND*.

Предложения **ReadOnly** / **Default Access** применяются только к окну Информации и управляют возможностью изменения данных в нем. и управляют возможностью изменения данных в нем. **ReadOnly** запрещает редактирование данных. Предложение **Default Access** снимает контроль со стороны MapBasic, и тогда уже действуют запреты или разрешения для самой таблицы. Это работает для главной легенды и картографических легенд, созданных операторами **Оператор Create Legend** или **Оператор Create Cartographic Legend**.

Предложение **Table** позволяет выбирать данные для показа в окне Информации (и только для него). Это предложение форсирует показ окна Информации.

Предложения **Show** и **Hide** управляют показом или скрыванием окон, для которых эта операция существенна (например, для Линейки), но может применяться и для окна MapInfo.

## Управление принтером

По умолчанию, окна распечатываются используя настройки системного принтера. Это может быть принтер Windows или настроенный в MI Pro принтер, в зависимости от установок пользователя. Использование предложения **Name** в приложении, рабочем наборе или окне MapBasic может изменить принтер для отдельного документа. Некоторые настройки для принтера также могут задаваться некоторыми другими командными предложениями. Таким образом, когда настройки принтера изменяются через пользовательский интерфейс, соответственно генерируются команды MapBasic. Подобные изменения настроек хранятся в рабочих наборах, а отменить их можно командой **Set Window Printer Default**.

Если **Scale Patterns** установлено **On**, штриховки масштабируются на основе соотношений разрешения устройства вывода и разрешения экрана.

Коды атрибутов WIN\_INFO\_PRINTER\_NAME, WIN\_INFO\_PRINTER\_ORIENT или WIN\_INFO\_PRINTER\_COPIES также возвращаются функцией **Функция WindowInfo( )**.

### Пример:

```
Set Window frontwindow( )
  Printer Name "\\Discovery\HP 2500CP"
  Orientation Portrait
  Copies 10
```

**Внимание:** Чтобы узнать имя принтера, запустите MapInfo Professional, выполните **Файл>Настройка печати**. Нажмите кнопку **Принтер**. Используйте имя принтера из этого диалога.

### Управление радиусом совмещения

Вы можете настроить радиус совмещения, измеряемый в пикселах для данного окна, вернуть стандартный размер радиуса совмещения для этого окна или получить информацию о текущем радиусе совмещения. Можно также включать/отключать режим совмещения для данного окна или запрашивать включен или нет режим совмещения для данного окна.

Настройки радиуса совмещения для обычного окна могут запрашиваться с использованием новых атрибутов параметров в функции [Функция WindowInfo\( \)](#). Теперь режим совмещения может настраиваться для каждого окна Карты или Отчета. Эти настройки сохраняются в рабочем наборе для каждого окна.

### Пример:

```
Dim win_id As IntegerOpen Table "world" Map From worldwin_id =  
FrontWindow( ) Set Window win_id Width 5 Height 3
```

### См. также:

[Оператор Browse](#), [Оператор Graph](#), [Оператор Layout](#), [Оператор Map](#), [Оператор Set Paper Units](#)

---

## Функция Sgn( )

### Назначение

Распознает знак числа. Возвращает -1, 0, или 1, указывая является ли число отрицательным, положительным или равным нулю.

### Синтаксис

**Sgn**( *num\_expr* )

*num\_expr* – числовое выражение

### Возвращаемая величина

Вещественное (-1, 0 или 1)

### Описание

Функция **Sgn( )** возвращает -1 (минус единица), если число, заданное выражением *num\_expr*, отрицательно, 0, если число равно нулю, и 1 (единица), если число больше нуля.

### Пример:

```
Dim x As Integerx = Sgn(-0.5)' x равно -1
```

### См. также:

[Функция Abs\( \)](#)

## Оператор Shade

### Назначение

Создает тематический слой и добавляет его к существующей карте.

### Синтаксис 1 (способ диапазонов)

```
Shade [ Window window_id ]
      { layer_id | layer_name }
      With Metadata
      With expr
      [ Ignore value_to_ignore ]
      Ranges
      [ Apply { Color | Size | All } ]
      [ Use { Color | Size | All } [ Line... ] [ Brush... ]
        [ Symbol... ]
      ]
      { [ From Variable float_array Style Variable style_array ] |
        minimum : maximum [ Pen... ] [ Line... ] [ Brush... ]
        [ Symbol... ] [ , minimum : maximum [ Pen... ]
        [ Line... ] [ Brush... ] [ Symbol... ] ... ]
      }
      [ Style Replace { On | Off } ]
      [ Default [ Pen... ] [ Line... ] [ Brush... ] [ Symbol... ] ]
```

### Синтаксис 2 (способ отдельных значений)

```
Shade [ Window window_id ]
      { layer_id | layer_name }
      With Metadata
      With expr
      [ Ignore value_to_ignore ]
      Values const [ Pen... ] [ Line... ] [ Brush... ] [ Symbol... ]
        [ , const [ Pen... ] [ Line... ] [ Brush... ] [ Symbol... ] ... ]
        [ Vary { Color | All } ]
      [ Style Replace { On | Off } ]
      [ Default [ Pen... ] [ Brush... ] [ Symbol... ] ]
```

### Синтаксис 3 (метод плотности точек)

```
Shade [ Window window_id ]
      { layer_id | layer_name }
      With expr
      Density dot_value { Circle | Square }
      Width dot_size
      [ Color color ]
```

**Внимание:** В целях обратной совместимости, поддерживается синтаксис MapBasic версий 7.5 и более ранних.

**Синтаксис 4 (метод размерных символов)**

```
Shade [ Window window_id ]
  { layer_id | layer_name }
  With expr
  Graduated min_value : symbol_size max_value : symbol_size
  Symbol...
  [ Inflect Symbol... ]
  [ Vary Size By { "LOG" | "SQRT" | "CONST" } ]
```

**Синтаксис 5 (круговые диаграммы)**

```
Shade [ Window window_id ]
  { layer_id | layer_name | Selection }
  With expr [ , expr... ]
  [ Half ] Pie [ Angle angle ] [ Counter ]
  [ Fixed ] [ Max Size chart_size [ Units unitname ]
    [ At Value max_value [ Vary Size By {"LOG" | "SQRT" | "CONST" } ] ] ]
  [ Border Pen... ]
  [ Position [ { Left | Right | Center } ] [ { Above | Below | Center } ] ]
  [ Style Brush... [ , Brush... ] ]
```

**Синтаксис 6 (столбчатые диаграммы)**

```
Shade [ Window window_id ]
  { layer_id | layer_name | Selection }
  With expr [ , expr... ]
  { Bar [ Normalized ] | Stacked Bar [ Fixed ] }
  [ Max Size chart_size [ Units unitname ]
    [ At Value max_value [ Vary Size By {"LOG" | "SQRT" | "CONST" } ] ] ]
  [ Border Pen... ]
  [ Frame Brush... ]
  [ Width value [ Units unitname ] ]
  [ Position [ { Left | Right | Center } ] [ { Above | Below | Center } ] ]
  [ Style Brush... [ , Brush... ] ]
```

*symbol\_size* – размер символа в пунктах, используемого в методе размерных символов.

*window\_id* - целочисленный идентификатор окна Карты.

*layer\_id* - идентификатор слоя в окне Карты, от единицы и больше.

*layer\_name* – имя слоя в окне Карты.

*expr* выражение, задающее условие выделения и раскраски, такое как имя колонки.

*value\_to\_ignore* – величина, которая должна быть проигнорирована, т.е. не будет создано тематического объекта для строки, значение которой равно параметру *value\_to\_ignore*; обычно это 0, если выражение численное, или пустая строка, если выражение строковое.

*float\_array*– численный массив типа Float, полученный оператором **Оператор Create Ranges**.

*style\_array* массив величин типа **Pen**, **Brush** или **Symbol** полученный оператором **Оператор Create Styles**.

*const* – численная или строковая константа или выражение, в котором не используется переменных.

Предложение **Предложение Pen** начинает стандартное предложение оператора для определения стиля контура для замкнутых объектов. Например, **MakePen**(*width, pattern, color*).

Предложение **Line** начинает стандартное предложение оператора для определения линии для объектов типа "линия", "полилиния" и "дуга". Предложение **Line** по строению идентично предложению **Предложение Pen**, за исключением начального ключевого слова **Line**.

Предложение **Предложение Brush** начинает стандартное предложение оператора для определения стиля штриха. Например, **MakeBrush**(*pattern, forecolor, backcolor*).

Предложение **Предложение Symbol** начинает стандартное предложение оператора для определения стиля символа точечного объекта. Например, **MakeSymbol**(*shape, color, size*).

*minimum* – минимальная величина для диапазона.

*maximum* – максимальная величина для диапазона

*dot\_value*– число, соответствующее одной точке при использовании метода плотности точек.

*dot\_size* – размер одной точки в пикселах при использовании метода плотности точек.

*color* - значение RGB для цвета точек при использовании метода плотности точек.

*angle* – начальный угол для круговой диаграммы.

*chart\_size* – число типа Float, представляющее максимальную высоту для круговых диаграмм или столбчатых диаграмм.

*unitname* – "бумажная" единица измерения (например, "in" – дюйм, "cm" – сантиметр).

*max\_value* – число, соответствующее точке в методе размерных символов. Для каждой записи, если сумма выражений в колонке равна *max\_value*, значение высоты соответствующей круговой или столбчатой диаграммы будет равно *chart\_size*; меньшие диаграммы - для последовательностей с меньшими суммами.

## Описание

Оператор **Shade** используется для создания тематического слоя в окне Карты. Этот оператор позволяет программе создать тематическую Карту так же, как это может делать пользователь при помощи диалога команды **Карта > Создать тематическую Карту** в MapInfo. Правила **создания тематической Карты** читайте в документации MapInfo.

MapInfo может запоминать тип тематической Карты, помещая оператор **Shade** в файл Рабочего Набора, если в нем есть хотя бы одно окно Карты с тематическим слоем. Для примера Вы можете открыть окно Карты, выполнить команду **Карта > Создать тематическую Карту** и сохранить Рабочий Набор (например, под именем THEME.WOR). Теперь откройте файл Рабочего Набора в любом текстовом редакторе и Вы увидите оператор **Shade**, задающий те настройки слоя, которые были заданы в открытом ранее окне Карты. Вы можете копировать этот оператор из Рабочего Набора в свою программу. Если при тематическом картографировании было открыто окно MapBasic, то команда также будет запротоколирована оператором **Shade**.

Параметр *window\_id*, задающий окно Карты, является необязательным. Если он опущен, то MapBasic создаст тематический слой в самом верхнем окне Карты.

Оператор **Shade** должен явно задать слой, на котором будет производиться условное выделение, даже если это единственный слой на Карте. Слой может задаваться номером (*layer\_id*), причем первый номер соответствует самому верхнему некосметическому слою, слой под первым будет вторым, и т. д. Кроме того, слой может задаваться именем (например, "world").

Если указать **With Metadata**, тематическая карта сохранённая в метаданных открытой таблицы будет отображена.

Способ тематического оформления оператором **Shade** зависит от того, какое предложение следует за выражением *expr*. MapInfo Professional определяет эти выражения для каждого выделенного объекта таблицы; в соответствии с оператором **Shade** MapInfo Professional выбирает стиль отображения каждого объекта, основанный на значении *expr*. Выражения обычно содержат имена одной или более колонок из выбранной таблицы.

Ключевые слова, следующие за предложением *expr*, указывают тип выделения. Предложение **Ranges** включается в оператор для создания Карты диапазонов, предложение **Values** – для Карты отдельных значений, предложение **Density** – для Карты плотности точек, предложение **Graduated** – для Карты размерных символов, а предложения **Pie** и **Bar** – для Карты с круговыми диаграммами или столбчатыми диаграммами.

### Создание тематического слоя методом диапазонов

Особенности синтаксиса Карты диапазонов смотрите в [Синтаксис 1 \(способ диапазонов\) на стр. 19](#).

В первом варианте синтаксиса предложения **From Variable** и **Style Variable** используются для чтения заранее вычисленных значений, определяющих границы диапазонов и стили для их оформления. Значения для массивов переменных, имена которых задаются в этих предложениях, должны быть вычислены до оператора **Shade** с помощью операторов **Оператор Create Ranges** и **Оператор Create Styles**. Пример использования массивов в операторе **Shade** приводится в описании оператора **Оператор Create Ranges on page 228**.

Если оператор **Shade** строит тематический слой методом диапазонов (предложение **Ranges**) или отдельных значений (ключевое слово **Values**; смотрите описание ниже), то он может использовать предложение **Default**, задающее единый стиль оформления объектов, которые попадают в группу "остальные", т. е. для них не выполняется условие тематического выделения. Таким образом, Вы можете задать единый стиль оформления для оставшихся объектов. Всего может быть задано от двух до шестнадцати диапазонов. Каждое описание диапазона состоит из пары чисел (*minimum* и *maximum*), разделенных двоеточием и определяющих верхний и нижний предел диапазона, и следующими за ними описаниями стилей оформления объектов, принадлежащих этому диапазону. Если выражение *expr* для записи больше или равно *minimum* и меньше *maximum*, то эта запись принадлежит этому диапазону. Описания диапазонов в предложении **Ranges** разделяются друг от друга запятыми.

```
Open Table "RUSSIA"Map From RUSSIAShade RUSSIA With Население Ranges
4827000:29280000 Brush (2,0,201326591) ,1783000: 4827000 Brush
(8,0,16777215) ,449000: 1783000 Brush (5,0,16777215)
```

При создании Карты диапазонов нужно задавать стиль раскраски и штриха предложением **Предложение Brush**. Стиль раскраски и штриха точек определяется предложением **Предложение Symbol**. Линейные объекты раскрашиваются, используя предложение **Line**, а не **Предложение Pen**, синтаксис которых одинаков, кроме замены ключевого слова **Pen** ключевым словом **Line**. (В операторе **Shade** предложение **Pen** управляет стилем контура замкнутых объектов.)

**Style Replace On** (по умолчанию) запрещает отображение слоёв под тематическим слоем.

**Style Replace Off** предписывает отображать слои под тематическим слоем, что позволяет создавать многовариантные прозрачные тематические слои.

**Style Replace On** - это режим по умолчанию, обеспечивающий обратную совместимость с существующими настройками, так что по умолчанию нижележащие слои не отображаются.

С помощью предложения **Apply**, Вы можете выбрать способ, которым будут выделены графические объекты на тематическом слое.

| Предложение Apply | Эффект   |
|-------------------|--|
| Apply Color       | Выделение производится только цветом, а размеры и другие атрибуты, такие как размер символа и тип линии, будут такими же, как на Карте. Исходные форма и размер точечных объектов не меняются. Исходные тип и толщина линейных объектов не меняются. Раскрашенные объекты визуализируются в своем исходном цвете, но тематическое выделение управляет цветом тона. |
| Apply Size        | Выделение производится только изменением размеров символов и толщиной линии. Точечные объекты при этом не меняют свой рисунок и раскраску, а линейные объекты – тип линий и цвет. Исходные тип и толщина линейных объектов не меняются.  |
| Apply All         | Все атрибуты будут меняться при условном выделении – рисунок символа, его размер, тип линии, толщина линии, штриховка и раскраска объектов.  |

По умолчанию используется режим, который устанавливается предложением **Apply All**.

Предложение **Use** позволяет управлять тем, в каком объеме MapInfo применяет стилизацию объектов при раскраске диапазонов. В следующем примере показан смысл этого предложения. Пусть таблица WorldCap, содержащая точки, раскрашена без участия предложения **Use**.

```
Shade WorldCap With Cap_Pop Ranges
  Apply All
  0 : 300000 Symbol(35,YELLOW,9) ,
  300000 : 900000 Symbol(35,GREEN,18) ,
  900000 : 20000000 Symbol(35,BLUE,27)
```

В результате тематическая Карта будет раскрашена звездочками (код 35) в соответствии с указаниями предложений **Предложение Symbol**: наименьшие объекты будут показаны желтыми звездочками в 9 точек, объекты среднего объема будут показаны зелеными звездочками в 18 точек и большие объекты будут показаны голубыми 27-точечными звездочками.

В следующем примере добавлено предложение **Use Size**.

```
Shade WorldCap With Cap_Pop Ranges
  Apply All

  Use Size Symbol(34, RED, 24) ' <<<<< Note!

  0 : 300000 Symbol(35,YELLOW,9) ,
  300000 : 900000 Symbol(35, GREEN,18) ,
  900000 : 20000000 Symbol(35,BLUE,27)
```

**Внимание:** Обратите внимание на то, что **Use Size** вносит свой стиль символа (код 34 представляет кружок).

Теперь все символы будут показаны красными кружками; от группы трех последних предложений **Предложение Symbol** применяются только размеры (9, 18, 27 точки). MapInfo игнорирует остальные атрибуты (например, YELLOW, GREEN, BLUE). Тематическая карта отобразит красные кружки, поскольку предложение **Use Size Предложение Symbol** определяет красные кружки. Конечный результат: наименьшие объекты показываются красными кружками в 9 точек, средние объекты показываются красными кружками в 18 точек, крупные объекты показываются красными кружками в 27 точек.

Если Вы зададите **Use Color** вместо **Use Size**, MapInfo будет использовать только цвета, заданные в последнем предложении **Предложение Symbol**. На Карте будут показаны желтые, зеленые и голубые кружочки размером в 24 точки.

Предложение **Use All** равносильно отсутствию предложения **Use** вообще.

Предложение **Use** действует только в паре с **Apply All** (либо **Apply** отсутствует вообще).

### Создание тематического слоя методом отдельных значений

Синтаксис карты Индивидуальных Значений смотрите в **Синтаксис 2 (способ отдельных значений) на стр. 19**.

За ключевым словом **Values** (второй вариант синтаксиса) может следовать от одного до 255 описаний. Каждое состоит из уникального значения (строка или число) и описаний стилей оформления объектов, принадлежащих значению. Если выражение *expr* для записи равно одному из уникальных значений, определенных в операторе **Shade**, то объект, который присоединен к этой записи, будет оформлен соответственно описанию для этого значения. Список описаний разделяется запятыми.

Если оператор **Shade** имеет предложение **Ranges** или **Values**, то он может иметь и предложение **Default**, задающее единый стиль оформления объектов, которые попадают в группу "остальные", т. е. для них не выполняется условие тематического выделения. Таким образом, Вы можете задать единый стиль оформления для оставшихся объектов.

Предложение **Vary** позволяет указать, как будет меняться вид объектов. По умолчанию применяется стиль **Vary All**. Если применяется стиль **Vary All**, к тематическому слою



применяются все инструменты отображения объектов в каждом диапазоне размеров объектов. Если задать **Vary Color**, то для каждого диапазона будет использован заданный цвет.

**Style Replace On** (по умолчанию) запрещает отображение слоёв под тематическим слоем.

**Style Replace Off** предписывает отображать слои под тематическим слоем, что позволяет создавать многовариантные прозрачные тематические слои. Это позволяет отображать прозрачные штриховки в том же слое.

**Style Replace On** - это режим по умолчанию, обеспечивающий обратную совместимость с существующими настройками, так что по умолчанию нижележащие слои не отображаются.

В следующем примере таблица территорий Великобритании UK\_SALES содержит колонку "Sales\_Rep", заполненную фамилиями торговых представителей, действующих на территории Великобритании. Оператор **Shade** закрашивает каждое графство (область) в зависимости от того, какой торговый представитель его курирует. Так, территории, за которые отвечает Боб, будут закрашены одним цветом, а графства, в которых торгует Джон, другим и т.д.

```
Open Table "uk_sales"
Map From uk_sales

Shade 1 With Proper$(Sales_Rep)
  Ignore ""
  Values
    "Alan" ,
    "Amanda" ,
    "Bob" ,
    "Jan"
```

## Карты плотности точек

Синтаксис Карты плотности точек смотрите в [Синтаксис 3 \(метод плотности точек\) на стр. 19](#).

За ключевым словом **Density** (третий вариант синтаксиса) должно следовать только одно описание *dot\_value* :*dot\_size*. Можно выбирать между стилями Circle и Square.

Раскрашиваемая карта должна содержать области, которые будут заполняться точками. Например, каждая точка соответствует одной тысяче домов.

Для такой карты численное выражение *expr* вычисляет значение для каждой области слоя Карты и делит его на *dot\_value*. MapInfo Professional определяет, сколько точек отображать для полигона, путём деления значения *expr* этого полигона на значение *dot\_value* карты. Например, если для области значение выражения равно 100, а *dot\_value* было задано равным 5, то MapBasic нарисует 20 точек в этой области.

За ключевым словом **Width** следует параметр *dot\_size*. Параметр *dot\_size* определяет размер иллюстративной точки в пикселах. Для круглых точек параметр *dot\_size* может изменяться от 2 до 25 пикселей. Для квадратных точек параметр *dot\_size* может изменяться от 1 до 25 пикселей. Предложение **Color** можно использовать для указания цвета точек.

В следующем примере в таблице штатов STATES строится тематическая карта плотности точек по количеству домовладений в каждом штате в 1990 году по колонке "Pop\_1990". Каждая точка имеет размер в 4 пиксела и представляет 60 000 домовладений.

```
Open Table "states"  
Map From states  
shade window 176942288 7  
with Pop_1990  
density 600000 circle width 4  
color 255
```

**Внимание:** В целях обратной совместимости, поддерживается синтаксис MapBasic версий 7.5 и более ранних.

### Создание тематического слоя методом размерных символов

Синтаксис Карты размерных символов смотрите в [Синтаксис 4 \(метод размерных символов\) на стр. 20](#).

За ключевым словом **Graduated** (четвертый вариант синтаксиса) следует пара операторов *value* :*symbol\_size*. Параметр *value* из первого описания соответствует минимальному значению, представляемому минимальным размером символа *symbol\_size*. Параметр *value* из второго описания соответствует максимальному значению, представляемому максимальным размером символа *symbol\_size*. Размер символов промежуточных значений MapInfo подбирает автоматически.

Предложение [Предложение Symbol](#) диктует, какой тип символа выбирается для выделения (кружок, звездочка и т. п.). Если оператор использует предложение **Inflect**, задающее второй вариант типа символа, то MapInfo обозначает этими символами объекты, имеющие отрицательное значение.

Следующий оператор создает тематический слой, отображающий доходы и убытки. Пункты, приносящие доход, обозначаются зелеными треугольниками, направленными вверх. Пункты, терпящие убытки, обозначаются красными треугольниками, направленными вниз.

```
Shade stores With Net_Profit
  Graduated
  0.0:0 15000:24
  Symbol(36, GREEN, 24)
  Inflect Symbol(37, RED, 24)
  Vary Size By "SQRT"
```

Предложение **Vary Size By** управляет методом, которым размерные символы отражают разницу между значениями, которые они представляют. Если предложение **Vary Size By** не задано, то MapInfo использует метод "SQRT" (квадратный корень), который задает размер символа пропорциональным квадратному корню из выделяемого значения. При действии этого метода, если в одной записи значение в два раза больше соответствующего значения в другой записи, то и площадь, занятая на экране символом для первого значения будет в два раза больше площади символа, представляющего второе значение.

**Внимание:** Увеличение площади в два раза не идентично увеличению размера символа в точках. Увеличение размера символа в точках увеличивает и длину, и ширину; следовательно, площадь увеличивается в квадрате.

### Создание тематического слоя методом круговых диаграмм

Синтаксис построения круговой диаграммы смотрите: "[Синтаксис 5 \(круговые диаграммы\) на стр. 20](#)".

Ключевое слово **Pie** (пятый вариант синтаксиса) задает метод построения тематического выделения объектов путем создания для каждого объекта маленького графика типа круговая диаграмма. При этом в предложении **With** через запятую должны быть заданы два или более выражений, значения которых для каждого объекта будут показаны круговой диаграммой.

Диаграммы могут иметь форму полного круга, и могут иметь форму половины окружности, если перед словом **Pie** поставить слово **Half**.

Стартовый угол для диаграммы Вы можете задать в предложении **Angle**. По умолчанию – 180 градусов.

Если используется ключевое слово **Counter**, то сектора в диаграмме располагаются от стартового угла против часовой стрелки.

Предложение **Max Size** определяет максимальный размер круговой диаграммы в "бумажных" единицах. Предложение **Fixed** задает рисование круговых диаграмм в кругах одинакового размера.

Следующий оператор задает выделение диаграммами одинакового размера, в четверть дюйма:

```
Shade sales_95 With phone_sales, retail_sales
  Pie Fixed
  Max Size 0.25 Units "in"
```

Чтобы круговая диаграмма могла устанавливать размер самостоятельно, применяйте вместо слова **Fixed** предложение **At Value**. Например, следующий оператор создает тематическую Карту с изменяющимся размером круговой диаграммы. Если сумма значений в записи равна 85 000, то Круговая диаграмма будет иметь радиус 2 см; записи с меньшими суммами породят меньшие диаграммы.

```
Shade sales_95 With phone_sales, retail_sales
  Pie
  Max Size 0.25 Units "in" At Value 85000
```

Предложение **Vary Size By** можно включить для выбора метода градуировки размера круговых диаграмм. Это предложение описано выше в разделе **Создание тематического слоя методом размерных символов на стр. 26**.

Предложение **Position** задает место относительно объекта, где помещается диаграмма. По умолчанию диаграммы помещаются в центр объекта.

В предложении **Style** можно задать раскраску для секторов диаграммы. Каждый стиль Brush должен соответствовать выражению из списка в предложении **With**. Если стили не заданы, то MapInfo использует значение типа Brush, сохраненное пользователем.

Следующий оператор создает тематический слой, в котором круговые диаграммы для двух значений помещаются выше центра объекта.

```
Shade sales_95 With phone_sales, retail_sales
  Pie Angle 180
  Max Size 0.5 Units "in" At Value 85000
  Vary Size By "SQRT"
  Border Pen (1, 2, 0)
  Position Center Above
  Style Brush(2, RED, 0), Brush(2, BLUE, 0)
```

### Создание тематического слоя методом столбчатых диаграмм

Синтаксис построения столбчатой диаграммы смотрите: **"Синтаксис 6 (столбчатые диаграммы) на стр. 20"**.

Ключевое слово **Bar** (шестой вариант синтаксиса) задает метод построения тематического выделения объектов путем создания для каждого объекта маленькой столбчатой диаграммы. При этом в предложении **With** через запятую должны быть заданы два или более выражений, значения которых для каждого объекта будут показаны диаграммы.

Если Вы использовали ключевое слово **Stacked** перед словом **Bar**, то MapInfo размещает столбики стопкой, иначе столбики располагаются в ряд. Если слово **Stacked** отсутствует, то можно задать слово **Normalized** и тогда столбики могут иметь независимые шкалы.

Если задано ключевое слово **Stacked**, то его можно дополнить словом **Fixed**, определяющим, что все стопки будут иметь одинаковый размер. Если слово **Fixed** не задано, то MapInfo задает размер каждой стопки пропорционально сумме отображаемых значений.

Предложение **Frame** **Предложение Brush** задает стиль раскраски фона графика.

Предложение **Position** управляет как ориентацией столбчатых диаграмм, так и ориентацией диаграммы по отношению к центроиду объекта. Если в предложении **Position** заданы слова **Left** или **Right**, то столбцы горизонтальны, в других случаях они вертикальны.

Предложение **Style** содержит перечисленные через запятую стили Brush. Каждый из стилей штриховки соответствует выражению из предложения **With**.

## Функция Sin( )

---

В следующем примере создается тематический слой на Карте, и выделение производится столбчатыми диаграммами, расположенными над центроидами объектов.

```
Shade sales_93
  With phone_sales, retail_sales
  Bar
  Max Size 0.4 Units "in" At Value 1245000
  Vary Size By "CONST"
  Border Pen (1, 2, 0)
  Position Center Above
  Style Brush(2, RED, 0), Brush(2, BLUE, 0)
```

**См. также:**

[Оператор Create Ranges](#), [Оператор Create Styles](#), [Оператор Map](#), [Оператор Set Legend](#), [Оператор Set Map](#), [Оператор Set Shade](#)

---

## Функция Sin( )

### Назначение

Вычисляет синус.

### Синтаксис

**Sin( *num\_expr* )**, где

*num\_expr* - численное выражение угла в радианах.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Sin( )** возвращает синус числа, полученного в результате вычисления выражения *num\_expr*. Угол должен задаваться в радианах. Диапазон возвращаемого от **Sin( )** значения находится между единицей и минус единицей включительно. Для перевода градусов в радианы число необходимо умножить на число DEG\_2\_RAD. Для обратного конвертирования используется коэффициент RAD\_2\_DEG. Чтобы Ваша программа могла использовать эти коэффициенты конвертирования, она должна содержать оператор Include "MAPBASIC.DEF"

### Пример:

```
Include "mapbasic.def" Dim x, y As Float x = 30 * DEG_2_RAD y = Sin(x) ' y
равно 0.5 ' поскольку синус 30 градусов 0.5
```

**См. также:**

[Функция Acos\( \)](#), [Функция Asin\( \)](#), [Функция Atn\( \)](#), [Функция Cos\( \)](#), [Функция Tan\( \)](#)

---

## Функция Space\$( )

### Назначение

Возвращает строку, состоящую из пробелов.

### Синтаксис

**Space\$( num\_expr )**, где  
*num\_expr* - целочисленное выражение.

### Возвращаемая величина

Строка

### Описание

Функция **Space\$( )** возвращает строку из пробелов. Количество пробелов определяется выражением *num\_expr*. Если *num\_expr* принимает значение меньше или равное нулю, функция **Space\$( )** возвратит пустую строку.

### Пример:

```
Dim filler As String  
filler = Space$(7) ' filler будет равна строке  
"      " ' (7 пробелов)  
Note "Здравствуй" + filler + "господа!" ' будет  
показано сообщение "Здравствуйте      господа!"
```

### См. также:

**Функция String\$( )**

---

## Функция SphericalArea( )

### Назначение

Возвращает площадь, вычисленную на сфере в координатах Долгота/Широта по дугам больших кругов.

### Синтаксис

**SphericalArea( obj\_expr, unit\_name )**, где  
*obj\_expr* - выражение, определяющее объект.  
*unit\_name* - строка, определяющая единицы измерения площади (к примеру, "кв. км.).

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **SphericalArea( )** возвращает площадь географического объекта, обозначенного выражением *obj\_expr*. Функция возвращает площадь области в единицах измерения, указанных в параметре *unit\_name*; к примеру, чтобы получить площадь в акрах, укажите "акры" в качестве значения параметра *unit\_name*. Список доступных единиц измерения смотрите в [Оператор Set Area Units на стр. 88](#).

Функция **SphericalArea( )** всегда возвращает площадь на сфере, вычисленную в координатах Долгота/Широта. Для данных в плановой системе координат возвращается значение -1, поскольку эти данные не могут быть преобразованы в мировую систему координат (Широта/Долгота).

Только полигоны, эллипсы, прямоугольники и прямоугольники со скругленными углами имеют площадь. Для точки, дуги, текста или полилинии значение функции **SphericalArea( )** по определению равно нулю. Функция **SphericalArea( )** возвращает примерные результаты для прямоугольников со скругленными углами. MapBasic вычисляет площадь прямоугольников со скругленными углами как если бы они были обычными прямоугольниками.

### Примеры

Следующий пример демонстрирует вычисление площади одного географического объекта при помощи функции **SphericalArea( )**. Обратите внимание, что выражение *tablename.obj* (аналогично *states.obj*) представляет географический объект из текущей строки указанной таблицы.

```
Dim f_sq_miles As FloatOpen Table "states" Fetch First From  
statesf_sq_miles = Area(states.obj, "sq mi")
```

Вы также можете использовать функцию **SphericalArea( )** в операторе [Оператор Select](#), как показано в следующем примере.

```
Select state, Area(obj, "sq km") From states Into results
```

**См. также:**

[Функция CartesianArea\( \)](#), [Функция SphericalArea\( \)](#)

---

## Функция SphericalConnectObjects( )

### Назначение

Возвращает объект, представляющий кратчайшее или самое большое расстояние между двумя объектами.

### Синтаксис

**SphericalConnectObjects( object1, object2, min ),** где

*object1* и *object2* - выражения, указывающие на объекты.

*min* - логическое выражение. Если это выражение принимает значение "Да" (TRUE), функция вычисляет минимальное расстояние между объектами, а если выражение принимает значение "Нет" (FALSE), вычисляется максимальное расстояние.



### Возвращаемая величина

Оператор возвращает объект полилинии, состоящий из одного сегмента и двух точек, и представляющий собой кратчайшее расстояние ((`min == TRUE`), или самое большое расстояние (`min == FALSE`) между объектами *object1* и *object2*.

### Описание

Одна точка полученной полилинии принадлежит объекту *object1*, а другая объекту *object2*. Обратите внимание, что расстояние между двумя объектами можно вычислить при помощи **Функция ObjectLen( )**. Если существует множество вариантов кратчайших или наибольших расстояний (например, полученный сегмент представляет одно расстояние, а существуют еще и другие сегменты равной длины), эти функции возвращают лишь один из вариантов. Не существует способа определить, является ли полученный объект полилинии единственным кратчайшим расстоянием.

**SphericalConnectObjects( )** возвращает объект Полилинии, определяющий кратчайшее (`min == TRUE`) или наибольшее (`min == FALSE`) расстояние между объектами *object1* и *object2*, полученные методом сферических расчетов. Если вычисления не могут быть произведены методом сферических расстояний (например, если используется плановая система координат), вызов этой функции приведет к ошибке.

---

## Функция SphericalDistance( )

### Назначение

Возвращает расстояние между двумя точками.

### Синтаксис

**SphericalDistance( *x1*, *y1*, *x2*, *y2*, *unit\_name* )**, где

*x1* и *x2* - координаты по оси x (например, долгота).

*y1* и *y2* - координаты по оси y (например, долгота).

*unit\_name* - строка, определяющая единицы измерения расстояния (к примеру, "км.").

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **SphericalDistance( )** вычисляет расстояние между двумя точками.

Функция возвращает расстояние в единицах измерения, указанных в параметре *unit\_name*; к примеру, чтобы получить расстояние в милях, укажите "миль" в качестве значения параметра *unit\_name*. Список доступных единиц измерения смотрите в **Оператор Set Distance Units на стр. 103**.

## Функция SphericalObjectDistance( )

---

Координаты по осям x и y должны быть заданы в текущей системе координат MapBasic. По умолчанию MapInfo Professional предполагает, что координаты заданы в мировой системе координат (Широта/Долгота). Вы можете установить систему координат по умолчанию при помощи **Оператор Set CoordSys**.

Функция **SphericalDistance( )** всегда возвращает площадь на сфере, вычисленную в координатах Долгота/Широта по дугам больших кругов. Для данных в плановой системе координат возвращается значение -1, поскольку эти данные не могут быть преобразованы в мировую систему координат (Широта/Долгота).

### Пример:

```
Dim dist, start_x, start_y, end_x, end_y As Float
Open Table "cities"
Fetch First From cities
start_x = CentroidX(cities.obj)
start_y = CentroidY(cities.obj)
Fetch Next From cities
end_x = CentroidX(cities.obj)
end_y = CentroidY(cities.obj)
dist = SphericalDistance(start_x, start_y, end_x, end_y, "mi")
```

См. также:

**Функция CartesianDistance( ), Функция Distance( )**

---

## Функция SphericalObjectDistance( )

### Назначение

Возвращает расстояние между двумя объектами.

### Синтаксис

**SphericalObjectDistance( *object1*, *object2*, *unit\_name* )**, где

*object1* и *object2* - выражения, указывающие на объекты.

*unit\_name* - строка, определяющая единицы измерения расстояния.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

**SphericalObjectDistance( )** возвращает минимальное расстояние между объектами *object1* и *object2*, вычисленное по методу сферических расчетов в единицах *unit\_name*. Если вычисления не могут быть произведены методом сферических расстояний (например, если используется плановая система координат), вызов этой функции приведет к ошибке.

## Функция SphericalObjectLen( )

### Назначение

Возвращает географическую протяженность объекта линии или полилинии.

### Синтаксис

**SphericalObjectLen**( *obj\_expr*, *unit\_name* ), где

*obj\_expr* - выражение, определяющее объект.

*unit\_name* - строка, определяющая единицы измерения расстояния (к примеру, "км.").

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **SphericalObjectLen( )** возвращает длину объекта. Помните, что ненулевые длины имеют только объекты линий и полилиний, для измерения периметра прямоугольника, эллипса или полигона, используйте [Функция Perimeter\( \)](#).

Функция **SphericalObjectLen( )** всегда возвращает площадь на сфере, вычисленную в координатах Долгота/Широта. Для данных в плановой системе координат возвращается значение -1, поскольку эти данные не могут быть преобразованы в мировую систему координат (Широта/Долгота).

Функция **SphericalObjectLen( )** возвращает длину в единицах измерения, указанных в параметре *unit\_name*. Например, чтобы получить длину в милях, укажите "миль" в качестве значения параметра *unit\_name*. Список допустимых единиц измерения смотрите в [Оператор Set Distance Units на стр. 103](#).

### Пример:

```
Dim geogr_length As Float
Open Table "streets" Fetch First From
streets
geogr_length = SphericalObjectLen(streets.obj, "mi")
' geogr_length
теперь содержит протяженность ' сегмента улицы в милях
```

### См. также:

[Функция CartesianObjectLen\( \)](#), [Функция SphericalObjectLen\( \)](#)

## Функция `SphericalOffset( )`

### Назначение

Возвращает копию исходного объекта, сдвинутую на указанное расстояние и угол в сферических координатах.

### Синтаксис

`SphericalOffset( object, angle, distance, units )`, где

*object* - объект, подвергаемый сдвигу.

*angle* - угол перемещения объекта.

*distance* - расстояние перемещения объекта.

*units* - строка, определяющая единицы измерения расстояний.

### Возвращаемая величина

Объект

### Описание

Эта функция создает новый объект, являющийся копией исходного и сдвинутый на расстояние *distance* вдоль угла *angle* (в градусах, где 0 градусов соответствует положительному направлению по оси X, а угол увеличивается против часовой стрелки). Строка единиц измерения, аналогична используемой для [Функция ObjectLen\( \)](#) и [Функция Perimeter\( \)](#), определяет единицы измерения расстояний. Тип данных измерения расстояний (`DistanceType`) – сферический. Если системой координат исходного объекта является план-схема, возникнет ошибка, поскольку сферические координаты для план-схем не могут быть определены. В случае ошибки возвращается объект `NULL`. Используется координатная система исходного объекта.

При проведении вычислений в сферической системе координат существуют некоторые соображения, неприменимые для декартовой системы координат. Если переместить объект в мировой системе координат, контуры объекта сохраняются, однако площадь объекта изменяется. Это происходит потому, что один из параметров сдвига определяется в градусах, а реальное измеренное расстояние одного градуса в различных точках неодинаково.

Для функций сдвига реальная дельта переноса вычисляется в некоторой фиксированной точке объекта (например, в центре описанного прямоугольника), после чего это значение преобразуется из исходных единиц измерения в единицы измерения системы координат. Если используется мировая система координат, для преобразования в градусы используется фиксированная точка. Действительное преобразованное расстояние может быть неодинаковым в различных точках объекта. Расстояние от исходного объекта до его перемещённой копии будет точным только в этой фиксированной точке.

### Пример:

```
SphericalOffset(Rect, 45, -100, "mi")
```

См. также:

Функция **SphericalOffsetXY()**

---

## Функция SphericalOffsetXY()

### Назначение

Возвращает копию исходного объекта, передвинутую в точку и на указанное расстояние в сферических координатах.

### Синтаксис

**SphericalOffsetXY**( *object*, *xoffset*, *yoffset*, *units* ), где

*object* - объект, подвергаемый сдвигу.

*xoffset* и *yoffset* - расстояния по осям x и y, на которые требуется сдвинуть объект.

*units* - строка, определяющая единицы измерения расстояний.

### Возвращаемая величина

Объект

### Описание

Функция **SphericalOffsetXY()** создает копию исходного объекта, сдвинутую на *xoffset* по оси x и на *yoffset* по оси y. Строка единиц измерения, аналогична используемой для **Функция ObjectLen()** и **Функция Perimeter()**, определяет единицы измерения расстояний. Тип данных измерения расстояний (DistanceType) – сферический. Если системой координат исходного объекта является план-схема, возникнет ошибка, поскольку сферические координаты для план-схем не могут быть определены. В случае ошибки возвращается объект NULL. Используется координатная система исходного объекта.

При проведении вычислений в сферической системе координат существуют некоторые соображения, неприменимые для декартовой системы координат. Если переместить объект в мировой системе координат, контуры объекта сохраняются, однако площадь объекта изменяется. Это происходит потому, что один из параметров сдвига определяется в градусах, а реальное измеренное расстояние одного градуса в различных точках неодинаково.

Для функций сдвига реальная дельта переноса вычисляется в некоторой фиксированной точке объекта (например, в центре описанного прямоугольника), после чего это значение преобразуется из исходных единиц измерения в единицы измерения системы координат. Если используется мировая система координат, для преобразования в градусы используется фиксированная точка. Действительное преобразованное расстояние может быть неодинаковым в различных точках объекта. Расстояние от исходного объекта до его перемещённой копии будет точным только в этой фиксированной точке.

### Пример:

```
SphericalOffsetXY(Rect, 92, -22, "mi")
```

См. также:

[Функция SphericalOffset\( \)](#)

---

## Функция SphericalPerimeter( )

### Назначение

Возвращает периметр графического объекта.

### Синтаксис

**SphericalPerimeter**( *obj\_expr*, *unit\_name* ), где

*obj\_expr* - выражение, определяющее объект.

*unit\_name* - строка, определяющая единицы измерения расстояния (к примеру, "км.").

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **SphericalPerimeter( )** вычисляет периметр объекта *obj\_expr*. Функция **SphericalPerimeter( )** определена для следующих типов объектов: эллипсы, прямоугольники, прямоугольники со скругленными углами, многоугольники. Другие типы объектов имеют периметр нулевой длины. Функция **SphericalPerimeter( )** возвращает длину в единицах измерения, указанных в параметре *unit\_name*. Например, чтобы получить длину в милях, укажите "миль" в качестве значения параметра *unit\_name*. Список допустимых единиц измерения смотрите в [Опепатор Set Distance Units на стр. 103](#).

Функция **SphericalPerimeter( )** всегда возвращает площадь на сфере, вычисленную в координатах Долгота/Широта. Для данных в плановой системе координат возвращается значение -1, поскольку эти данные не могут быть преобразованы в мировую систему координат (Широта/Долгота). Функция **SphericalPerimeter( )** возвращает примерные результаты для прямоугольников со скругленными углами. MapBasic вычисляет периметр прямоугольников со скругленными углами как если бы они были обычными прямоугольниками.

### Пример:

В следующем примере демонстрируется применение функции **SphericalPerimeter( )** для вычисления периметра конкретного географического объекта.

```
Dim perim As Float
Open Table "world"
Fetch First From world
perim = SphericalPerimeter(world.obj, "km")
' Переменная perim теперь содержит '
длинну периметра многоугольника, связанного ' с первой записью в таблице
World.
```

Функцию **SphericalPerimeter( )** также можно использовать в операторе **Оператор Select**. Следующий оператор **Оператор Select** извлекает информацию из таблицы States и сохраняет результаты во временной таблице Results. Поскольку оператор **Оператор Select** содержит вызов функции **SphericalPerimeter( )**, таблица Results будет включать колонку, отражающую периметр каждого из штатов.

```
Open Table "states"Select state, Perimeter(obj, "mi")From states Into results
```

**См. также:**

**Функция CartesianPerimeter( ), Функция Perimeter( )**

---

## Функция Sqr( )

### Назначение

Вычисляет квадратный корень.

### Синтаксис

**Sqr**( *num\_expr* ), где

*num\_expr* - численное выражение, результатом которого должно быть положительное число.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **Sqr( )** возвращает квадратный корень от числа, полученного в результате вычисления выражения *num\_expr*. Это число должно быть большим или равным нулю.

Извлечение квадратного корня эквивалентно возведению числа в степень 0.5.

Соответственно, **Sqr(n)=n ^ 0.5**. Причем надо отметить, что MapBasic быстрее вычисляет функцию, а не возведение в степень 0.5.

### Пример:

```
Dim n As Float
n = Sqr(25)
```

**См. также:**

**Функция Cos( ), Функция Sin( ), Функция Tan( )**

## Оператор StatusBar

### Назначение

Показывает или прячет строку сообщений в рабочем окне MapInfo или выдает в ней сообщение.

### Синтаксис

```
StatusBar { Show | Hide } [ Message message ] [ ViewDisplayPopup { On | Off } ] [ EditLayerPopup { On | Off } ], где
```

*message* - это сообщение, которое будет показано в строке сообщений.

### Описание

Оператор **StatusBar** управляет отображением строки сообщений в рабочем окне MapInfo, а также позволяет печатать в ней сообщения из прикладной программы.

Чтобы напечатать сообщение в строке сообщений, используйте предложение **Message**.

```
StatusBar Message "Вычисление координат..."
```

MapInfo автоматически заменит Ваше сообщение другими, когда пользователь будет перемещать указатель мышки по командам меню или кнопкам инструментальных панелей. Поэтому текст сообщения должен быть лаконичен, что бы пользователь успел его заметить и прочесть за короткий промежуток времени. По этой причине не советуем Вам помещать важные сообщения в строку сообщений.

Для этих целей можно воспользоваться окном **Оператор Print**, открываемым оператором Print.

Используйте параметр **ViewDisplayPopup**, позволяющий изменять вид строки сообщений. Если этот параметр установлен на значение "On", то пользователь сможет изменять масштаб и положение курсора, устанавливаемых из строки сообщений.

Использование параметра **EditLayerPopup** позволяет пользователю устанавливать редактируемый слой в окне Карты из строки сообщений. Если этот параметр установлен на значение "On", то пользователь сможет выбирать редактируемый слой из строки сообщений.

### См. также:

**Оператор Note**, **Оператор Print**

---

## Параметр Stop

### Назначение

Приостанавливает выполнение прикладной программы для отладки.

### Синтаксис

```
Stop
```



## Предупреждение

Вы не можете использовать оператор **Stop** во внутренних функциях программы. Вы не можете использовать оператор **Stop** в обработчиках элемента диалога **Оператор Dialog**, так как отладочный процесс невозможен, пока открыто диалоговое окно.

## Описание

Оператор **Stop** является отладочным средством. Оператор приостанавливает выполнение программы и передает управление пользователю. В данном случае пользователь – это программист, который отлаживает свою программу.

Если программа была приостановлена при активном окне MapBasic, то в нем выводится сообщение с номером строки программы, в которой был выполнен оператор **Stop**.

Если окно MapBasic не было открыто, то оно будет открыто. Далее пользователь может использовать окно MapBasic для исследования текущего состояния программы. Если набрать:

```
? Dim
```

в окне MapBasic, то MapInfo выведет список всех используемых локальных переменных в программе. Если набрать:

```
? Global
```

в окне MapBasic, то MapInfo выведет список всех используемых глобальных переменных в программе.

Задав после знака вопроса имя переменной, пользователь может получить ее текущее значение. Пользователь сам может присвоить значение какой-нибудь переменной. Формат присвоения следующий:

```
variable_name = new_value,
```

где *variable\_name* – имя локальной или глобальной переменной, и *new\_value* – выражение, представляющее новую величину для этой переменной.

Для продолжения выполнения приостановленной программы надо выполнить команду **Файл > Продолжить** (команда **Продолжить** появится в меню **Файл** вместо команды **Выполнить**). Продолжить выполнение программы также можно, введя в окно MapBasic оператор **Оператор Continue**.

В режиме остановки (после оператора **Stop**) MapInfo продолжает держать открытым файл приложения. Поэтому его нельзя перекомпилировать. Поэтому выйдите из режима остановки командой **Файл > Продолжить** программу, если хотите перекомпилировать файл.

**См. также:**

**Оператор Continue**

## Функция Str\$( )

### Назначение

Возвращает строковое представление числа, объекта и стиля.

### Синтаксис

**Str\$( expression )**, где

*expression* – выражение, результатом которого является число или величина типа Date, Pen, Brush, Symbol, Font и Object.

### Возвращаемая величина

Строка

### Описание

Функция **Str\$( )** возвращает строку, которая представляет величину, полученную в результате вычисления выражения *expression*.

Если параметром *expression* является отрицательное число, то результатом будет строка, начинающаяся с символа знака минус (-). Если число положительно, то строка начинается с пробела.

Длина строки, возвращаемая функцией Str\$( ), зависит от точности вычисления выражения *expression* и количества цифр справа от запятой. Другими словами, функция **Str\$( )** вернет строку, используя округленную величину. Если Вы хотите управлять преобразованием, воспользуйтесь функций **Функция Format\$( )**.

Если величина в *expression* имеет объектный тип (Object), функция **Str\$( )** вернет одно из следующих значений: Arc, Ellipse, Frame, Line, Point, Polyline, Rectangle, Region, Rounded Rectangle, или Text.

Если параметр *expression* задает объект (тип Object) выражением в форме *tablename.obj* (где *tablename* – имя открытой таблицы) и если к текущей строке таблицы не присоединен графический объект, то функция **Str\$( )** вернет пустую строку.

**Внимание:** передача неинициализированной объектной переменной приведет к тому, что функция **Str\$( )** вернет ошибку.

Если параметр *expression* задает дату (Date), функция **Str\$( )** вернет строку в формате, установленном в вычислительной системе, в которой выполняется эта прикладная программа. Например, следующее выражение:

```
Str$( NumberToDate(19951231) )
```

может быть равно и "31.12.95", и "1995/12/31" (etc.), в зависимости от того, какой формат использует пользователь в своем компьютере. Для управления форматом преобразования функцией **Str\$( )** используйте оператор **Оператор Set Format**.

Если параметр *expression* задает число, то функция **Str\$( )** будет использовать точку в качестве десятичной точки, даже если в компьютере пользователя используется другой знак. Функция **Str\$( )** не использует разделитель тысяч при конвертировании числа в строку. Если Вы хотите получить число с разделителями тысяч и десятичной точкой такой, какая задана в данной вычислительной платформе, то используйте функцию **Функция FormatNumber\$( )**.

#### Пример:

```
Dim s_spelled_out As String, f_profits As Float
f_profits = 123456
s_spelled_out = "Число улиц:" + Str$(f_profits)
```

#### См. также:

**Функция Format\$( ), Функция FormatNumber\$( ), Оператор Set Format, Функция Val( )**

## Функция String\$( )

### Назначение

Строит строку, повторяя символ заданное количество раз.

### Синтаксис

**String\$( num\_expr, string\_expr )**, где

*num\_expr* - положительное целочисленное выражение.

*string\_expr* - выражение, результат которого есть строка.

### Возвращаемая величина

Строка

### Описание

Функция **String\$( )** возвращает строку, состоящую из первых символов строки, заданной параметром *string\_expr*. Параметр *num\_expr* задает длину будущей строки (в символах).

#### Пример:

```
Dim filler As String
filler = String$(5, "ABCDEFGH") ' переменная filler
равна строке "AAAAA" ' (5 раз повторяется первая буква)
```

#### См. также:

**Функция Space\$( )**

## Функция StringCompare( )

### Назначение

Сравнивает две строки, учитывая различия строчных и прописных символов.

### Синтаксис

**StringCompare**( *string1*, *string2* ), где  
*string1* и *string2* - строковые выражения.

### Возвращаемая величина

Короткое целое число -1, 1 или 0. Величина типа SmallInt.

### Описание

Функция **StringCompare( )** позволяет сравнивать строки с учетом регистра. MapBasic, сравнивая строки с использованием операции "=", не учитывает разницу между прописными и строчными буквами. Например, в операторе:

```
If "ABC" = "abc" Then
```

сравнение возвращает TRUE (истину).

Функция **StringCompare( )** позволяет делать сравнения строк, рассматривая строчные и прописные буквы как разные. Сравнение строк происходит посимвольно. Как только встречаются два разных символа, выполнение функции прекращается.

| Возвращаемая величина: | Результат:  |
|------------------------|---|
| -1                     | Код символа из первой строки меньше, чем код соответствующего символа из второй строки. |
| 0                      | Две строки равны.   |
| 1                      | Код символа из первой строки больше, чем код соответствующего символа из второй строки. |

### Пример:

Функция `StringCompare("ABC", "abc")` возвращает значение -1 (n= -1), поскольку ANSI-код "A" меньше "a".

### См. также:

[Функция Like\( \)](#), [Функция StringCompareIntl\( \)](#)

---

## Функция StringCompareIntl( )

### Назначение

Сравнивает две строки, учитывая особенности сортировки для разных языков.

### Синтаксис

**StringCompareIntl**( *string1*, *string2* ), где

*string1* и *string2* - строковые выражения.

**Возвращаемая величина**

Короткое целое число -1, 1 или 0. Величина типа SmallInt.

**Описание**

Функция **StringCompareIntl( )** позволяет сравнивать две строки, учитывая языковые особенности, заданные в "Панели управления. В состав сравниваемых строк могут входить символы, используемые в других языках.

Сравнение строк происходит посимвольно. Как только встречаются два разных символа, выполнение функции прекращается.

| Возвращаемая величина: | Результат:  |
|------------------------|---|
| -1                     | Код символа из первой строки меньше, чем код соответствующего символа из второй строки. |
| 0                      | Две строки равны.   |
| 1                      | Код символа из первой строки больше, чем код соответствующего символа из второй строки. |

См. также:

[Функция Like\( \)](#), [Функция StringCompare\( \)](#)

**Функция StringToDate( )**

**Назначение**

Переводит строку в величину даты.

**Синтаксис**

**StringToDate( *datestring* )**, где  
*datestring* - строка, представляющая дату.

**Возвращаемая величина**

Дата типа Date

**Описание**

Функция **StringToDate( )** создает из строки величину типа Date. MapBasic интерпретирует данные в соответствии с форматами, которые установлены в компьютере, в котором работает программа. Если компьютер использует стандарт США, то дата представляется в формате Месяц/День/Год, но также существуют другие стандарты, принятые в других странах

(например, День/Месяц/Год). Другим может быть также знак разделителя (например, точка вместо /). Для использования стандарта США в функции **StringToDate( )** используйте **Оператор Set Format**.

**Внимание:** Независимо от установок в компьютере, используйте оператор **Функция NumberToDate( )** вместо **StringToDate( )**. Оператор **Функция NumberToDate( )** не зависит от настроек компьютера пользователя.

Если используется формат США, то параметр *datestring* должен содержать компоненты числа, месяца и года, разделенные косой чертой (/). Первые два символа задают месяц, значение которого должно находиться в диапазоне от 1 до 12. Два символа второй компоненты задают число, значение которого должно находиться в диапазоне от 1 до 31. Год может задаваться двумя или четырьмя символами. Если год не указан, то принимается значение настоящего года. Если указать год в виде двух цифр (например, 96), MapInfo Professional составит дату для текущего столетия, как определено для: **"Оператор Set Date Window"**.

### Пример:

Данный пример показывает строку типа Date в формате, принятом в США: Месяц/День/Год. Эта программа вызывает оператор **Оператор Set Format** до использования функции **StringToDate( )**, Чтобы убедиться, что строки типа Date в формате США интерпретируются корректно независимо от установок системы.

```
Dim d_start, d_end As Date Set Format Date "US" d_start =  
StringToDate("17.12.92") d_end = StringToDate("02.01.95") Set Format Date  
"Local"
```

В этом примере переменная Date1 = 19890120, Date2 = 20101203 и MyYear = 1990.

```
DIM Date1, Date2 as Date DIM MyYear As Integer Set Format Date "US" Set Date  
Window 75 Date1 = StringToDate("20.01.89") Date2 = StringToDate("03.12.10")  
MyYear = Year("30.12.90")
```

Эти результаты соответствуют оператору **Оператор Set Date Window**, позволяющему контролировать год в двухзначном формате.

**См. также:**

**Функция NumberToDate( )**, **Оператор Set Format**, **Функция Str\$( )**

---

## Функция StringToDateTime

### Назначение

Возвращает значение типа DateTime (Дата/Время) на основе сведений из строки, в которой перечислены дата и время. MapBasic интерпретирует данные типа DateTime в соответствии с форматами, заданными в компьютере, где работает программа. Между датой и временем требуется хотя-бы один пробел. Подробнее см. разделы **Функция StringToDate( )** и **Функция StringToTime**.

**Синтаксис**

```
StringToDateTime (String)
```

**Возвращаемая величина**

```
DateTime
```

## Функция StringToTime

---

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim strX as string
dim Z as datetime
strX = "19990912041345789"
Z = StringToDateTime(strX)
Print FormatDate$(Z)
Print FormatTime$(Z, "hh:mm:ss.fff tt")
```

---

## Функция StringToTime

### Назначение

Возвращает значение типа Time (Время) на основе сведений из строки с временем. MapBasic интерпретирует данные типа Time в соответствии с форматом, заданным в компьютере, где работает программа. При этом допускается либо 12-часовой, либо 24-часовой отсчет времени. Кроме того, малозначащие компоненты времени можно опустить. Другими словами, необходимо задавать часы, а минуты секунды и миллисекунды использовать необязательно.

### Синтаксис

```
StringToTime (String)
```

### Возвращаемая величина

Время

### Пример:

Скопируйте текст этого примера в окно MapBasic и проверьте работу этой функции.

```
dim strY as string
dim X as time
strY = "010203000"
X = StringToTime(strY)
Print FormatTime$(X, "hh:mm:ss.fff tt")
```

---

## Функция StyleAttr( )

### Назначение

Возвращает значение одной из компонент стиля оформления объекта: Pen, Brush, Font, или Symbol.

### Синтаксис

```
StyleAttr( style, attribute ), где
```

*style* - величина, выражающая стиль (величина типа Pen, Brush, Font или Symbol).

*attribute*- целочисленный код, управляющий результатом функции.



Возвращаемая величина

Целое число или строка, в зависимости от значения параметра attribute.

Описание

Функция **StyleAttr( )** извлекает из величины типа Pen, Brush, Symbol или Font определенную компоненту.

Все типы в MapBasic являются сложносоставными. Например, определение стиля Brush состоит из трех компонентов: pattern, foreground color и background color. При вызове функции **StyleAttr( )**, параметр *attribute* контролирует, какой из атрибутов возвращается..

Параметр *attribute* должен являться одним из кодов, показанных в нижеследующей таблице. Коды в левом столбце (например, PEN\_WIDTH) определены в файле стандартных определений MAPBASIC.DEF.

| Параметры       | StyleAttr( ) возвращает:  |
|-----------------|---|
| BRUSH_PATTERN   | Целое число (Integer), атрибут стиля Brush, задающий номер штриха.  |
| BRUSH_FORECOLOR | Целое число (Integer), атрибут стиля Brush, задающий RGB-код цвета штриха.  |
| BRUSH_BACKCOLOR | Целое число (Integer), атрибут стиля Brush, задающий RGB-код цвета фона штриха или -1, если фон прозрачный.   |
| FONT_NAME       | Атрибут стиля Font, задающий строку с именем шрифта.  |
| FONT_STYLE      | Целое число (Integer) от 0 до 7, атрибут стиля Font (0 = простой, 1 = жирный, и т.д.); см. <b>Предложение Font на стр. 85</b> для более подробной информации.   |
| FONT_POINTSIZE  | Целое число (Integer), атрибут стиля Font, задающий размер шрифта в точках.<br><b>Внимание:</b> Если текстовый объект принадлежит таблице, а не Отчету, то размер будет равен 0 и высота букв будет определяться текущим масштабом Карты. |
| FONT_FORECOLOR  | Целое число (Integer), атрибут стиля Font, задающий RGB-код цвета символов строки.  |
| FONT_BACKCOLOR  | Целое число (Integer), атрибут стиля Font, задающий RGB-код цвета фона строки или -1, если фон прозрачный. Если стиль шрифта включает кайму, то задает RGB-цвет каймы.  |
| PEN_WIDTH       | Целое число (Integer), атрибут стиля Pen, задающий ширину линии в точках.   |

| Параметры           | StyleAttr( ) возвращает:   |
|---------------------|--|
| PEN_PATTERN         | Целое число (Integer), атрибут стиля Pen, задающий номер вида линии.   |
| PEN_COLOR           | Целое число (Integer), атрибут стиля Pen, задающий RGB-код цвета линии.  |
| PEN_INTERLEAVED     | Логическое: TRUE если стиль пересекающихся линий.  |
| PEN_INDEX           | Целое число (Integer), представляющее индекс линии из библиотеки линий.  |
| SYMBOL_KIND         | Целое число (Integer), тип символа: 1 - символ Map Info версии 3.0; 2 - символ шрифта TrueType; 3 - символ из растрового файла.                            |
| SYMBOL_CODE         | Целое число (Integer), атрибут стиля Symbol, задающий номер символа. Используется для типа символа версии MapInfo 3.0 и TrueType.                          |
| SYMBOL_COLOR        | Целое число (Integer), атрибут стиля Symbol, задающий RGB-код цвета символа.   |
| SYMBOL_POINTSIZE    | Целое число (Integer) от 1 до 48, атрибут стиля Symbol, задающий размер символа в пунктах.   |
| SYMBOL_FONT_NAME    | Строка (String), имя шрифта TrueType, который используется как библиотека символов.  |
| SYMBOL_FONT_STYLE   | Целое число (Integer), задающее написание символа TrueType (0 = нормальное написание, 1 = жирное, и т.д.). Смотрите <b>Предложение Symbol на стр. 53</b> . |
| SYMBOL_ANGLE        | Действительное число (Float), угол поворота символа TrueType.  |
| SYMBOL_CUSTOM_NAME  | Строка (String), имя растрового файла.   |
| SYMBOL_CUSTOM_STYLE | Целое число (Integer), задающее стиль для растрового символа (0 = нормальное, 1 = показать фон и т.д.). Смотрите <b>Предложение Symbol на стр. 53</b> .    |

#### Ошибки:

ERR\_FCN\_ARG\_RANGE, если неправильно задано значение аргумента.

#### Пример:

Воспользуемся функцией **Функция CurrentPen( )** для того, что бы узнать, какой установлен сейчас стиль линии в MapInfo, затем с помощью функции **StyleAttr( )** узнаем толщину линии в пикселах.

```

Include "mapbasic.def"
Dim cur_width As Integer
cur_width = StyleAttr(CurrentPen( ), PEN_WIDTH)

```

**См. также:**

**Предложение Brush, Предложение Font, Предложение Pen, Предложение Symbol, Функция MakeBrush( ), Функция MakeFont( ), Функция MakePen( ), Функция MakeSymbol( )**

---

## Оператор Sub...End Sub

### Назначение

Определяет процедуру, которую можно вызвать из другой процедуры оператором **Оператор Call**.

### Синтаксис

```

Sub proc_name [ ( [ ByVal ] parameter As var_type [ , ... ] ) ]
statement_list End Sub, где

```

*proc\_name* – имя процедуры;

*parameter* – имя параметра процедуры;

*var\_type* – стандартный в MapBasic тип переменных (например, Integer) или сложно-определенный тип, заранее объявленный оператором Type;

*statement\_list* – список (от 0 и более) операторов процедуры.

### Предупреждение

Вы не можете использовать оператор **Sub... End Sub** в окне MapBasic.

### Описание

Оператор **Sub... End Sub** описывает процедуру в MapBasic. Как только sub-процедура определена, к ней можно обращаться из любой другой части программы оператором **Оператор Call**.

Каждая процедура, задаваемая оператором **Sub... End Sub**, должна быть заранее объявлена оператором **Оператор Declare Sub**.

Каждая подпрограмма может иметь или не иметь параметры, значения которых передаются из процедуры, вызвавшей эту sub-процедуру. *parameter* – имя параметра. Каждое такое определение параметра процедуры должно быть уникально. Параметры для подпрограммы задаются списком через запятую, где каждый параметр определяется следующим образом: [ByVal] parameter As var\_type

Если при определении параметра процедуры ключевое слово ByVal не участвует, то это значит, что параметр определен ссылкой ("by reference"). То есть программа, вызывая процедуру, параметры которой заданы ссылкой, должна использовать только имена переменных в качестве параметров вызова. Вы не можете передавать константы или

выражения (такие как "Москва") ссылкой. Впоследствии, если sub-процедура изменила значения этих параметров, то также изменятся значения переменных, использовавшихся как параметры вызова в операторе Call. Таким образом, параметр, передаваемый ссылкой, позволяет не только доставлять значение из вызывающей процедуры, но и возвращать значение обратно в этом же параметре. Если при объявлении использовалось ключевое слово ByVal, то параметр процедуры определен для передачи процедуре значением ("by value"). Программа, вызывая процедуру, параметры которой заданы значением, может использовать как имена переменных, так и константы или выражения в качестве параметров вызова. Однако, если изменить значение такого параметра в теле процедуры, новое значение нельзя будет вернуть в вызывающий модуль. Параметр, определенный как "значение", не может передавать массивы, значения переменных сложного типа, созданного оператором Type и объявленных в предложении Alias.

Sub-процедура может обрабатывать массивы. В списке параметров массив объявляется с помощью пустых скобок, следующих за именем переменной (аналогично **Оператор Declare Sub**). В следующем примере в процедуре "ListProcessor" объявлен в качестве параметра массив целых величин "items".

```
Sub ListProcessor(items( ) As Integer)
```

При вызове такой процедуры оператором Call параметром вызова должно быть имя массива переменных целого типа без круглых скобок.

Следует помнить, что, если в процедуре объявляется локальная переменная с таким же именем, как у существующей уже глобальной переменной, то значение последней недоступно для использования в этой процедуре. Под данным именем в процедуре используется только локальная переменная.

Предложение **Оператор Exit Sub**

заканчивает работу sub-процедуры.

### Пример:

Процедура Cube возводит число в куб (в третью степень) и возвращает результат. Процедура имеет два параметра: первый для возводимого степень вещественного числа, второй для результата.

```
Declare Sub Main Declare Sub Cube(ByVal original As Float, cubed As  
Float)Sub Main Dim x, result As FloatCall Cube(2, result) ' переменная  
result теперь имеет значение: 8 (2 x 2 x 2)x = 1Call Cube(x + 2, result)  
' переменная result теперь имеет значение: 27 (3 x 3 x 3) End Sub Sub Cube  
(ByVal original As Float, cubed As Float) ' Возвести в куб значение  
параметра и сохранить ' в переменной "cubed".  
cubed = original ^ 3End Sub
```

**См. также:**

**Оператор Call, Оператор Declare Sub, Оператор Dim, Оператор Exit Sub, Оператор Function...End Function, Оператор Global**

## Предложение Symbol

### Назначение

Задание стиля символа для точечного объекта.

### Синтаксис (вариант 1 - версия для символов MapInfo 3.0):

**Symbol** ( *shape, color, size* ), где

*shape* – целое число, величина типа Integer, (значение 31 задает невидимый символ); от 31 или больше, задающий символ из стандартного набора MapInfo (значение 31 задает невидимый символ);

*color* – целочисленный код цвета в системе RGB, смотрите описание функции **Функция RGB( ) на стр. 118**;

*size* – целое число, величина типа Integer, размер символа в точках от 1 до 48;

### Синтаксис (вариант 2 - версия для символа из шрифта TrueType):

**Symbol** ( *shape, color, size, fontname, fontstyle, rotation* ), где

*shape* – целое число, величина типа Integer, от 31 или больше, задающий символ из шрифта TrueType (значение 31 задает невидимый символ);

*color* – целочисленный код цвета в системе RGB, смотрите описание функции **Функция RGB( ) на стр. 118**;

*size* – целое число, величина типа Integer, размер символа в точках от 1 до 48;

*fontname* – строка с именем шрифта TrueType (например, “WingDings”);

*fontstyle* – целочисленный код, величина типа Integer, управляющий написанием шрифта;

*rotation* – вещественное число, угол поворота в градусах.

### Синтаксис (вариант 3 - версия для символа из растрового файла):

**Symbol** ( *filename, color, size, customstyle* )

*filename* – строка до 31 символа длиной с именем растрового файла (файл должен находиться в каталоге, заданном пользователем);

*color* – целочисленный код цвета в системе RGB, смотрите описание функции **Функция RGB( ) на стр. 118**;

*size* – целое число, величина типа Integer, размер символа в точках от 1 до 48;

*customstyle* – целочисленный код типа Integer, управляющий цветом и фоном символа. См. таблицу ниже.

### Синтаксис (вариант 4):

**Symbol** *symbol\_expr*, где

*symbol\_expr* – выражение, результат которого есть величина типа Symbol, например, переменная типа Symbol или MakeSymbol(shape, color, size).

### Описание

**Внимание:** Предложение Symbol позволяет задавать стиль символа для точечного объекта. Предложение не является отдельным оператором, но входит в состав многих операторов, работающих с точечными объектами. Например, в операторе Create Point предложение определяет стиль символа для нового объекта. Если предложение Symbol в этом операторе опущено, то будет использован текущий стиль символа, установленный в среде MapInfo.

Некоторые операторы MapBasic (такие как **Alter Object...Info OBJ\_INFO\_SYMBOL**) используют выражение типа **Symbol** как параметр без ключевого слова **Symbol**.

### Стандартный набор символов MapInfo 3.0

В следующей таблице показаны символы и соответствующие им коды из стандартного набора, который используется в первом варианте синтаксиса предложения Symbol.

|    |   |    |   |    |   |    |   |
|----|---|----|---|----|---|----|---|
| 31 |   | 41 | ☆ | 51 | ✱ | 61 | 🛡 |
| 32 | ■ | 42 | △ | 52 | ✈ | 62 | 🛡 |
| 33 | ◆ | 43 | ▽ | 53 | 🚩 | 63 | 🏠 |
| 34 | ● | 44 | ■ | 54 | 🚩 | 64 | ⚡ |
| 35 | ☆ | 45 | ▲ | 55 | 🏠 | 65 | 🚩 |
| 36 | ▲ | 46 | ● | 56 | + | 66 | 🚩 |
| 37 | ▽ | 47 | ➡ | 57 | ⚡ | 67 | 🚩 |
| 38 | □ | 48 | ↙ | 58 | ⚓ |    |   |
| 39 | ◇ | 49 | + | 59 | ⊙ |    |   |
| 40 | ○ | 50 | × | 60 | 🏠 |    |   |

### Символы шрифта TrueType

Если Вы задаете символ из шрифта TrueType, то параметр fontstyle управляет написанием символа:

| значение <i>fontstyle</i> | Стиль символа |
|---------------------------|---------------|
| 0                         | Нормальное    |
| 1                         | Жирное        |
| 16                        | Черная кайма  |
| 32                        | Оттененное    |
| 256                       | Белая кайма   |

Для задания двух или более стилей написания коды складываются. Например, для того, чтобы получить символ жирного и оттененного написания, параметр *fontstyle* должен быть равен 33. Написание 16 и 256 взаимно исключают друг друга.

Растровый символ

Если Вы задаете новый символ, то параметр *customstyle* управляет, какими будут цвет и фон символа:

| Значение <i>customstyle</i> | Стиль символа  |
|-----------------------------|--|
| 0                           | Не действуют режимы из группы “Эффекты” диалога “Стиль символа”, и символ появляется таким, какой он есть. Все белые пиксели раstra прозрачны. |
| 1                           | Действует режим “Добавить фон”; все белые пиксели раstra непрозрачны.  |
| 2                           | Действует режим “Покрасить одним цветом”; все не белые точки раstra закрашены одним цветом.  |
| 3                           | Установлены оба флажка (действуют оба режима).   |
| 4                           | Действует режим “Показать реальный размер”; растровый символ визуализируется в свою натуральную величину.                                      |
| 5                           | Действуют режимы “Добавить фон” и “Показать реальный размер”.  |
| 7                           | Действуют режимы: “Добавить фон”, “Покрасить одним цветом” и “Показать реальный размер”.   |

Пример:

Оператор **Оператор Set Map** может использовать предложение **Symbol**. **Оператор Set Map** назначает стиль символов для показа точечных объектов первого слоя карты в виде кружочков (символ номер 34), закрашенных красным и размером в 18 пунктов.

Функция SystemInfo( )

```
Include "mapbasic.def"Set Map Layer 1 Display GlobalGlobal Symbol
MakeSymbol (34,RED,18)
```

См. также:

Функция MakeCustomSymbol( ), Функция MakeFontSymbol( ), Функция MakeSymbol( ),  
Функция StyleAttr( )

Функция SystemInfo( )

Назначение

Возвращает информацию об активной операционной среде и версии программы MapInfo. Эта функция имеет атрибут, с помощью которого MapBasic может выяснить номер версии MapInfo Professional. Например, SystemInfo(SYS\_INFO\_MIVERSION) возвращает 850 для MapInfo Professional 8.5 и 8.5.2. Отличить версию 8.5 от 8.5.2 можно с помощью SystemInfo(SYS\_INFO\_MIBUILD\_NUMBER); здесь будет возвращено 32 для 8.5 и 60 для 8.5.2.

Синтаксис

```
SystemInfo(SYS_INFO_MIBUILD_NUMBER)
```

**Внимание:** Константа, определенная для этой функции в файле MapBasic.def равна 18.

*attribute* – целочисленный код, определяющий тему запроса.

Возвращаемая величина

Величина типа SmallInt, Logical или String

Описание

Функция **SystemInfo( )** возвращает информацию об оперативной среде и версии программы MapInfo, в которых Вы сейчас работаете. Вид информации задает параметр *attribute*. В файле стандартных определений MapBasic MAPBASIC.DEF определены имена кодов, которые Вы можете использовать в качестве параметра функции SystemInfo( ). Для использования имен кодов Ваша программа должна иметь в начале оператор Include "MAPBASIC.DEF".

| Значения attribute      | Результат, полученный SystemInfo( )  |
|-------------------------|--|
| SYS_INFO_APPLICATIONWND | Целое число, уникальный номер Windows <i>HWND</i> , установленный оператором <b>Оператор Set Application Window</b> (или ноль, если <i>HWND</i> не устанавливался).              |
| SYS_INFO_APPVERSION     | Целое число, соответствующее версии языка MapBasic, на котором была написана Ваша программа, и компилятора, умноженное на 100. Например, если версия 3.0, то функция вернет 300. |
| SYS_INFO_CHARSET        | Возвращает имя используемой системы кодов.   |



| Значения attribute     | Результат, полученный SystemInfo( )  |
|------------------------|--|
| SYS_INFO_COPYPROTECTED | Логическая величина: “Да” (TRUE), если пользователь запустил защищенную от копирования версию MapInfo.   |
| SYS_INFO_DATE_FORMAT   | Строка: “US” или “Local”, соответствующие способу форматирования даты. Смотрите описание оператора <b>Оператор Set Format на стр. 107</b> .  |
| SYS_INFO_DDESTATUS     | Целочисленная величина, количество элементов в DDE-очереди на выполнение. Если очередь пуста, то <b>SystemInfo( )</b> вернет ноль (если приходящие команды будут выстраиваться в очередь) или -1 (если приходящие команды будут немедленно выполняться). |
| SYS_INFO_DIG_INSTALLED | Логическая величина: “Да” (TRUE), если дигитайзер и совместимый драйвер установлен.  |
| SYS_INFO_DIG_MODE      | Логическая величина: “Да” (TRUE), если включен режим оцифровки.  |
| SYS_INFO_MAPINFOWND    | Целое число, представляющее номер Windows <i>HWND</i> главного окна MapInfo. Функция возвращает 0, если программа выполняется не в Windows.  |
| SYS_INFO_MDICLIENTWND  | Целое число, представляющее номер Windows <i>HWND</i> для окна MapInfo MDICLIENT. Функция возвращает 0, если программа выполняется не в Windows.   |
| SYS_INFO_MIPLATFORM    | Целое число, тип программы MapInfo: MIPLATFORM_WIN16 (16–битная версия Windows), MIPLATFORM_WIN32 (32–битная версия Windows).  |
| SYS_INFO_MIVERSION     | Целое число, соответствующее версии программы MapInfo, в которой Вы сейчас работаете, умноженное на 100 (сто).   |
| SYS_INFO_NUMBER_FORMAT | Строка: “9,999.9” или “Local”, соответствующие способу форматирования чисел. Смотрите оператор <b>Оператор Set Format на стр. 107</b> .  |
| SYS_INFO_PLATFORM      | Целочисленный код, определяющий операционную систему. Возвращенное число будет кодом PLATFORM_WIN.   |
| SYS_INFO_PRODUCTLEVEL  | Целое число, отражающее уровень (product level) MapInfo (800 для MapInfo Professional 8.0).  |

## Функция TableInfo( )

| Значения attribute                     | Результат, полученный SystemInfo( )   |
|--|---|
| SYS_INFO_RUNTIME                       | Возвращает логическое "да" (TRUE) если работает runtime-версия MapInfo, и логическое "нет" (FALSE), если иначе. |
| SYS_INFO_APPIDISPATCH<br>(значение=17) | Целое число, представляющее указатель на интерфейс IDispatch OLE Automation для приложения MapInfo.             |

### Ошибки:

ERR\_FCN\_ARG\_RANGE, если неправильно задано значение аргумента.

### Пример:

В зависимости от того, в какой операционной среде выполняется программа (в Windows или нет), будет запущена или нет процедура, использующая DDE-связь.

```
Declare Sub DDE_Setup

If SystemInfo(SYS_INFO_PLATFORM) = PLATFORM_WIN Then
    Call DDE_Setup
End If
```

## Функция TableInfo( )

### Назначение

Возвращает информацию об открытой таблице. Функция может различать таблицы FME (внешних данных).

### Синтаксис

**TableInfo**( *table\_id*, *attribute* ), где

*table\_id* – либо целочисленный номер таблицы, либо имя таблицы в кавычках, либо 0;

*attribute* – целочисленный код, определяющий тему запроса.

### Возвращаемая величина

Строка, логическое значение, целое или короткое целое число. Величина типа String, Integer, SmallInt или Logical, в зависимости от значения параметра attribute.

Второй параметр *attribute* определяет вид информации о данной таблице MapInfo, которая будет получена. Коды в левом столбце (например, TAB\_INFO\_NAME) определены в файле стандартных определений MAPBASIC.DEF.

Описание

Функция **TableInfo( )** используется для получения определенной информации об открытой таблице. Первый параметр функции определяет из какой таблицы была затребована информация.

Параметр *table\_id* может быть строкой с именем открытой таблицы или числом, равным ее номеру. Если значение параметра *table\_id* равно 0, то функция **TableInfo( )** будет опрашивать таблицу, которая была открыта или создана самой последней. Вы можете использовать функцию с нулевым первым параметром сразу после оператора **Оператор Open Table**, в котором предложение **As** не использовалось. Если в момент вызова функции **TableInfo( )** не было открыто ни одной таблицы или все таблицы уже закрыты, то результатом функции будет ошибка.

Второй параметр *attribute* определяет вид информации о данной таблице MapInfo, которая будет получена. Коды в левом столбце (например, TAB\_INFO\_NAME) определены в файле стандартных определений MAPBASIC.DEF.

| Значения attribute  | TableInfo( ) возвращает:  |
|---|---|
| TAB_INFO_BROWSER_LIST   | Строка результата: указывает какие колонки будут отображаться в окне списка. Эта информация хранится в разделе метаданных таблицы. Если эта информация отсутствует возвращается пустая строка.                          |
| TAB_INFO_COORDSYS_CLAUSE  | Строка с предложением <b>Оператор CoordSys</b> , соответствующим проекции таблицы, например "CoordSys Earth Projection 1, 0". Возвращается пустая строка, если таблица не может иметь графические объекты.              |
| TAB_INFO_COORDSYS_MINX,<br>TAB_INFO_COORDSYS_MINY,<br>TAB_INFO_COORDSYS_MAXX,<br>TAB_INFO_COORDSYS_MAXY | Вещественные величины, минимальная и максимальная координаты по оси X и Y, которые могут быть сохранены в таблице. Если таблица не может иметь объектов, результатом будет ноль.  |
| TAB_INFO_COORDSYS_NAME  | Строка с именем проекции, каким она названа в файле MAPINFOW.PRJ (но без суффикса "\r..."). Результатом будет пустая строка, если таблица не может иметь графические объекты или нет соответствия в файле MAPINFOW.PRJ. |
| TAB_INFO_EDITED   | Логическая величина: "Да" (TRUE), если таблица имеет несохраненные на диске изменения.  |
| TAB_INFO_FASTEDIT   | Логическая величина: "Да" (TRUE), если для редактирования таблицы включен режим <b>FastEdit</b> , и "Нет" (FALSE), если выключен. (Смотрите описание оператора <b>Оператор Set Table</b> на стр. 149, режим FastEdit.)  |

| Значения attribute  | TableInfo( ) возвращает:  |
|---|---|
| TAB_INFO_MAPPABLE   | Логическая величина: "Да" (TRUE), если записям таблицы можно сопоставлять графические объекты.  |
| TAB_INFO_MAPPABLE_TABLE   | Строка, представляющая базовую таблицу, то есть такую, которая содержит графические объекты. Если таблица представляет собой результат объединения, то возвращает имя той, которая содержит графические объекты.  |
| TAB_INFO_MINX,<br>TAB_INFO_MINY,<br>TAB_INFO_MAXX,<br>TAB_INFO_MAXY | Вещественное число, минимальная и максимальная координаты по оси X и Y, координаты углов минимального прямоугольного покрытия всех объектов таблицы.  |
| TAB_INFO_NAME   | Строка, имя таблицы.  |
| TAB_INFO_NCOLS  | Целое число (тип SmallInt), количество колонок в таблице.   |
| TAB_INFO_NREFS  | Целое число (тип SmallInt), указывает количество таблиц, связанных с данной таблицей. (Возвращает ноль для большинства таблиц. Другое значение возможно в случае, когда таблица определена как слияние двух других таблиц, например, таблица StreetInfo.) Может использоваться только с базовыми таблицами (TAB_TYPE_BASE). |
| TAB_INFO_NROWS  | Целое число (тип Integer), количество строк в таблице.  |
| TAB_INFO_NUM  | Целое число (тип SmallInt), номер открытой таблицы.   |
| TAB_INFO_READONLY   | Логическая величина: "Да" (TRUE), если таблица открыта в режиме "только чтение".  |
| TAB_INFO_SEAMLESS   | Логическая величина: "Да" (TRUE), если для таблицы включен атрибут сшитости.  |
| TAB_INFO_TABFILE  | Строка с полным именем файла таблицы, включая DOS-маршрут. Пустая строка возвращается для таблицы запроса.  |
| TAB_INFO_TEMP   | Логическая величина: "Да" (TRUE), если таблица является временной (например, ЗАПРОС1).  |
| TAB_INFO_THEME_METADATA   | Результат - логическое значение; TRUE, если в метаданных есть описание тематической карты используемой по умолчанию.  |

| Значения attribute      | TableInfo( ) возвращает:   |
|-------------------------|--|
| TAB_INFO_TYPE           | <p>Целое число (тип SmallInt), код, Результатом может быть один из следующих кодов:</p> <ul style="list-style-type: none"> <li>• TAB_TYPE_BASE, если таблица нормальная;</li> <li>• TAB_TYPE_RESULT, если таблица запроса;</li> <li>• TAB_TYPE_IMAGE, если таблица растровая;</li> <li>• TAB_TYPE_VIEW, если таблица является представлением (view), например, таблица STREETINFO является представлением;</li> <li>• TAB_TYPE_LINKED, если таблица связанная.</li> <li>• TAB_TYPE_WMS (если таблица Web Map Service).</li> <li>• TAB_TYPE_WFS (если таблица Web Feature Service).</li> <li>• TAB_TYPE_FME (если таблица была открыта с помощью FME).</li> </ul> |
| TAB_INFO_UNDO           | <p>Логическая величина: "Да" (TRUE), если для редактирования таблицы включена система обратных действий (Undo), и "Нет" (FALSE), если система выключена оператором <b>Оператор Set Table</b>.</p>  |
| TAB_INFO_USERBROWSE     | <p>Логическая величина: "Нет" (FALSE), если оператором <b>Оператор Set Table</b> выключен режим <b>UserBrowse</b>.</p>   |
| TAB_INFO_USERCLOSE      | <p>Логическая величина: "Нет" (FALSE), если оператором <b>Оператор Set Table</b> выключен режим <b>UserClose</b>.</p>  |
| TAB_INFO_USERDISPLAYMAP | <p>Логическая величина: "Нет" (FALSE), если оператором <b>Оператор Set Table</b> выключен режим <b>UserDisplayMap</b>.</p>   |
| TAB_INFO_USEREDITABLE   | <p>Логическая величина: "Нет" (FALSE), если оператором <b>Оператор Set Table</b> выключен режим <b>UserEdit</b>.</p>   |
| TAB_INFO_USERMAP        | <p>Логическая величина: "Нет" (FALSE), если оператором <b>Оператор Set Table</b> выключен режим <b>UserMap</b>.</p>  |
| TAB_INFO_USERREMOVEMAP  | <p>Логическая величина: "Нет" (FALSE), если оператором <b>Оператор Set Table</b> выключен режим <b>UserRemoveMap</b>.</p>  |
| TAB_INFO_SUPPORT_MZ     | <p>Логическая величина: "Да" (TRUE), если таблица поддерживает значения m и z.</p>   |

Функция Tan( )

| Значения attribute  | TableInfo( ) возвращает:  |
|---------------------|---|
| TAB_INFO_Z_UNIT_SET | Логическая величина: "Да" (TRUE), если для z-значений установлены единицы измерения.                                |
| TAB_INFO_Z_UNIT     | Строка, указывающая единицы расстояния для z-значений. Возвращает пустую строку, если единицы измерения не указаны. |

Ошибки:

ERR\_TABLE\_NOT\_FOUND, если неправильно задана таблица или ее нет.

ERR\_FCN\_ARG\_RANGE, если неправильно задано значение аргумента.

Пример:

```
Include "mapbasic.def"
Dim i_numcols As SmallInt, L_mappable As Logical
Open Table "world"
i_numcols = TableInfo("world", TAB_INFO_NCOLS)
L_mappable = TableInfo("world", TAB_INFO_MAPPABLE)
```

См. также:

Оператор Open Table

Функция Tan( )

Назначение

Вычисляет тангенс.

Синтаксис

```
Tan( num_expr ), где
num_expr - численное выражение угла в радианах.
```

Возвращаемая величина

Вещественное число типа Float.

Описание

Функция Tan( ) возвращает тангенс от числа, полученного в результате вычисления num\_expr. Число num\_expr является угловой величиной в радианах.

Для перевода градусов в радианы число необходимо умножить на число DEG\_2\_RAD. Для обратного конвертирования используется коэффициент RAD\_2\_DEG. Чтобы Ваша программа могла использовать эти коэффициенты конвертирования, она должна содержать оператор Include "MAPBASIC.DEF".

**Пример:**

```
Include "mapbasic.def" Dim x, y As Float x = 45 * DEG_2_RAD y = Tan(x) ' y
равно 1 ' так как тангенс от 45 градусов равен 1
```

**См. также:**

**Функция Acos( ), Функция Asin( ), Функция Atn( ), Функция Cos( ), Функция Sin( )**

---

## Функция TempFileName\$( )

**Назначение**

Возвращает имя, которое можно использовать при создании временного файла.

**Синтаксис**

**TempFileName\$( dir ),** где

*dir* – строка с маршрутом (диск+каталог), где будет создан временный файл; пустая строка ("" ) задает системный каталог для создания временных файлов.

**Возвращаемая величина**

Строка. Величина типа String.

**Описание**

Функция **TempFileName\$( )** используется, когда Вам необходимо создать временный файл, но Вы не знаете, какое имя ему дать.

Если Вы вызвали функцию **TempFileName\$( )**, MapBasic вернет строку с полным уникальным именем файла. Сама функция **TempFileName\$( )** не создает временный файл. Создать файл Вы можете оператором **Оператор Open File**.

Если параметр *dir* задан пустой строкой (""), то полное имя файла будет включать в себя системный каталог для временных файлов, например, "G:\TEMP\~MAP0023.TMP".

При работе в сети возможна ситуация, когда два пользователя одновременно могут создать временный файл на одном и том же месте, с одинаковым именем. Если Вы пробуете создать файл с именем и на каталоге, которые Вы получили от функции **TempFileName\$( )**, может возникнуть конфликтная ситуация, потому что файл уже существует, так как другой пользователь сети успел создать его после того, как Ваша программа вызвала функцию **TempFileName\$( )**, и до оператора создания файла. Для снижения вероятности такой конфликтной ситуации выполняйте оператор **Оператор Open File** непосредственно сразу после вызова функции **TempFileName\$( )**. Для снижения вероятности сетевых конфликтов создайте обработчик ошибок и поместите оператор **Оператор OnError** после оператора **Оператор Open File**.

**См. также:**

**Функция FileExists( )**

## Оператор Terminate Application

### Назначение

Закрывает другие выполняющиеся программы MapBasic или программы MapBasic, находящиеся в состоянии ожидания.

### Синтаксис

**Terminate Application** *app\_name*, где

*app\_name* – строковая величина с именем работающей программы (например, "scalebar.mbx")

### Описание

Если прикладная программа создает свои меню, инструментальные панели, то она сама не завершается, а остается загруженной в режиме ожидания событий, которые надо обработать (пользователь выполняет команду, нажимает на кнопки, созданные прикладной программой и т. д.). Чтобы закрыть программу, находящуюся в режиме ожидания, в другой программе используется оператор **Terminate Application**. Например, если во время отладочного процесса Вам необходимо закрыть программу, находящуюся в режиме ожидания, выполните оператор **Terminate Application** в окне MapBasic.

Ваша прикладная программа может запустить другую прикладную программу, используя оператор **Оператор Run Application**, она же может и завершить его оператором **Terminate Application**.

**Внимание:** Заметим, что оператор **Terminate Application** используется для завершения только других прикладных программ, а для завершения самой программы используется оператор **Оператор End Program**.

См. также:

**Оператор End Program, Оператор Run Application**

---

## Функция TextSize( )

### Назначение

Возвращает в пунктах размер текста в окне.

### Синтаксис

**TextSize**( *window\_id*, *text\_obj* ), где

*window\_id* - целое, идентификатор, определяющий окно Карты или Отчета. Вызовите **Функция FrontWindow( )** или **Функция WindowID( )** чтобы получить идентификатор окна.



*text\_obj* текстовый объект.

**Внимание:** Если текстовый объект из окна Карты, идентификатор ID окна должен быть ID окна Карты. Если текстовый объект из окна Отчета, идентификатор ID окна должен быть ID окна Отчета.

### Возвращаемая величина

Вещественное число типа Float.

### Описание

Функция **TextSize( )** возвращает значение текстового объекта в пунктах исходя из текущего масштаба окна. Эта функция соотносится с выборкой текстового объекта и информацию о тексте можно получить выполнив **Правка>Геоинформация** или нажав клавишу F7.

### Пример:

Если активное окно это окно Карты, а текстовый объект выбран:

```
print TextSize(FrontWindow( ), selection.obj)
```

**См. также:**

**Предложение Font**

---

## Функция Time( )

### Назначение

Возвращает текущее системное время в строковом формате. Время может быть в 12-и часовом или 24-часовом формате.

### Синтаксис

```
StringVar = Time( Format )
```

### Описание

*StringVar* - это строковая переменная, возвращающая системное время в формате ЧЧ:ММ:СС. *Format* - это целочисленное значение, определяющее формат возвращаемой строки. Если формат равен 24, то время возвращается в 24-часовом формате. Для любого другого значения время возвращается в 12-часовом формате.

---

## Функция Timer( )

### Назначение

Возвращает число секунд.

### Синтаксис

`Timer ( )`

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция **Timer( )** возвращает число секунд, прошедших с полуночи первого января 1970 года. Вы можете вызывать функцию **Timer( )** до и после какой-либо операции, чтобы узнать, сколько времени в секундах она выполняется.

### Пример:

```
Declare Sub UbiDim start, elapsed As Integer start = Timer( )Call  
Ubielapsed = Timer( ) - start'' Переменная elapsed содержит время  
выполнения' подпрограммы Ubi '
```

---

## Процедура ToolHandler

### Назначение

Специальная процедура для обеспечения работы со специальной инструментальной кнопкой (инструмент MapBasic ).

### Синтаксис

`Declare Sub ToolHandler Sub ToolHandler statement_list End Sub`, где  
*statement\_list* – список операторов, выполняющихся при выборе пользователем инструмента MapBasic.

### Описание

ToolHandler – зарезервированное имя для процедуры MapBasic. Процедура работает совместно с инструментом MapBasic.

Одним из простых способов создания своей кнопки на инструментальной панели "Операции", является специальная процедура **ToolHandler**. Однако, представление кнопки, связанной с процедурой **ToolHandler**, имеет некоторые ограничения. Вы не можете использовать произвольную картинку для кнопки или назначить свой режим рисования этой кнопке. Для создания кнопок, не имеющих этих ограничений, используются операторы **Оператор Alter ButtonPad** и **Оператор Create ButtonPad**.

Если пользователь запустил программу, которая имеет процедуру **ToolHandler**, то на инструментальной панели "Операции" появляется кнопка, на которой изображен знак плюс. Эта кнопка называется инструментом MapBasic. Инструмент может использоваться в активном окне Списка, Карты или Отчета. Как только пользователь указал инструментом MapBasic в область окна Списка, Карты или Отчета, MapBasic автоматически запускает на выполнение процедуру **ToolHandler**.

В процедуре **ToolHandler** может быть использована функция **CommandInfo( )** функция для распознавания идентификатора окна, на которое было указано. Если инструмент был использован в окне Списка, функция может также дать информацию о строке и колонке, на которые указал инструмент. Если инструмент был использован в окне Карты, функцию можно использовать для получения координат указанной точки в текущей системе координат (смотрите оператор **Оператор Set CoordSys** на стр. 99).

Если инструмент был использован в окне Отчета, функция **Оператор Set Paper Units** на стр. 141 поможет Вам получить координаты точки на листе (т. е. расстояния от верхнего и левого края листа), в которой находился указатель мышки, когда Вы нажали на кнопку мыши. (См. ).

Координаты на листе Отчета возвращаются в текущих единицах измерения, которые устанавливаются оператором . Функция **CommandInfo( )** также может определить, была ли нажата клавиша SHIFT или CTRL или обе вместе, когда пользователь указывал на окно Списка, Карты или Отчета инструментом MapBasic. Это позволит Вам определить по крайней мере четыре варианта действий для разных способов использования инструмента.

Выбрать инструмент на инструментальной панели может сама прикладная программа, выполнив оператор:

```
Run Menu Command M_TOOLS_MAPBASIC
```

Чтобы воспользоваться процедурой **ToolHandler**, пользователю необходимо запустить приложение. Когда пользователь запускает программу, в которой есть процедура **ToolHandler**, программа не завершается после выполнения всех операторов процедуры Main и других процедур, вызванных из нее.

Процедура Main быть и объявлена, и задана неявно. Программа будет пребывать в режиме ожидания до тех пор, пока не будет выбран инструмент MapBasic и применен в окне Списка, Карты или Отчета. Программа автоматически начнет выполнение процедуры **ToolHandler**.

Когда процедура закончит свои действия (**Оператор End Program**), прикладная программа вновь переходит в режим ожидания. И так всякий раз при новом применении инструмента. Для завершения программы в процедуре ToolHandler используется оператор **Оператор End Program**. После использования оператора **Оператор End Program** полностью освобождается память, занимаемая всей прикладной программой, кнопка инструмента убирается с панели. Поэтому, если Вам снова понадобятся функции, выполняемые инструментом MapBasic, не используйте оператор **Оператор End Program** в процедуре **ToolHandler**, и оставьте программу в режиме ожидания до нового выбора инструмента.

В процедуре **ToolHandler**, в зависимости от окна, в котором будет использован инструмент, должен использоваться оператор **Оператор Set CoordSys** перед распознаванием координат выбранной инструментом точки в окне. Если процедура **ToolHandler** используется в окне Списка, то оператор **Оператор Set CoordSys** не нужен. Перед распознаванием координат точки в окне Отчета в процедуре **ToolHandler** используется вариант **Set CoordSys Layout statement**. Если пользователь указал на точку Карты, и координатная система Карты не совпадает с координатной системой, принятой в прикладной программе, то процедура **ToolHandler** должна выполнить либо оператор **Оператор Set CoordSys** либо **Оператор Set CoordSys** перед тем, как определять координаты выбранной точки.

### Пример:

Следующий фрагмент программы тестирует инструмент MapBasic. При запуске программы сначала открывается окно сообщений, приглашающее пользователя выбрать инструмент MapBasic. Когда инструмент будет выбран, процедура **ToolHandler** будет выдавать сообщения о координатах каждой выбранной точки в окнах Карт, Списков, Отчетов.

```
Include "mapbasic.def" Declare Sub ToolHandlerNote "Инструмент MapBasic  
готов для проверки." Sub ToolHandler Note "x:" +  
Round(CommandInfo(CMD_INFO_X), 0.1) + Chr$(10) + " y:" +  
Round(CommandInfo(CMD_INFO_Y), 0.1) End Sub
```

### См. также:

[CommandInfo\( \) функция](#)

---

## Функция TriggerControl( )

### Назначение

Возвращает идентификатор элемента диалога, к которому пользователь обращался последним.

### Синтаксис

**TriggerControl( )**

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция **TriggerControl( )** используется только внутри процедуры-обработчика диалога, построенного при помощи оператора **Оператор Dialog**. Функция возвращает номер элемента диалога, который был присвоен ему при создании диалога параметром из предложения ID.

Каждый элемент диалога может вызывать процедуру-обработчик. Это могут быть разные процедуры, а может быть так, что несколько элементов содержат вызов одного и того же обработчика. В последнем случае имеет смысл определить, какой именно элемент диалога был выбран пользователем.

### Ошибки:

ERR\_INVALID\_TRIG\_CONTROL, если функция вызвана не при активном диалоге.

### См. также:

[Оператор Alter Control](#), [Оператор Dialog](#), [Оператор Dialog Preserve](#), [Оператор Dialog Remove](#), [Функция ReadControlValue\( \)](#)

## Функция TrueFileName\$( )

### Назначение

Возвращает полное имя файла на основе его неполной спецификации.

### Синтаксис

**TrueFileName\$( file\_spec )**, где

*file\_spec* – строковая величина с неполной спецификацией файла (например, в Windows "C:\parcels.tab")

### Описание

Функция TrueFileName\$( ) возвращает полную спецификацию файла

(полное имя диска, все каталоги или папки и имя файла), используя неполную. Например, в DOS следующий файл специфицирован частично (есть имя диска и имя файла, но нет каталогов, образующих маршрут):

"C:\parcels.tab"

Если текущий каталог на диске C: является \MAPINFO\DATA, то функция

TrueFileName\$( "C:\parcels.tab" )

вернет следующую строку:

"C:\mapinfo\data\parcels.tab"

Функцию **TrueFileName\$( )** удобно использовать в программе, которая подсказывает пользователю спецификацию для его файла на диске. Например, в диалоге перед сохранением файла.

Функция **TrueFileName\$( )** не подтверждает наличие файла с таким именем, а только составляет полную спецификацию для файла, такую, какую бы он мог иметь. Для определения существования файла на диске используйте функцию **Функция FileExists( )**.

**См. также:**

**Функция ProgramDirectory\$( )**

## Оператор Type

### Назначение

Создает произвольную сложную структуру типа данных. Сложный тип позже можно будет использовать наравне со стандартными типами в операторах **Оператор Dim** и **Оператор Globals** для объявления переменных.

### Синтаксис

**Type** *type\_name* *element\_name* **As** *var\_type* [ ... ] **End Type**, где

*type\_name* – имя сложного типа данных;

*element\_name* – имя элемента типа;

*var\_type* – тип данных для элемента.

### Предупреждение

Все операторы **Type** должны находиться на "глобальном" уровне в тексте программы, то есть за рамками sub-процедур. Вы не можете использовать оператор **Type** в окне MapBasic. Вы не можете использовать переменные сложного типа, созданного оператором **Type** как параметр процедуры или функции для пересылки значением ("by-value"). Также нельзя использовать такие переменные для записи значений в файл оператором **Оператор Put**.

### Описание

Оператор **Type** используется для создания нового типа данных, состоящего из элементов стандартных типов или других заранее определенных сложных типов. Использование сложных типов данных позволяет создавать многоуровневые структуры данных, такие как очереди, двоичные деревья, графы. Для обращения к элементу переменной сложного типа надо использовать следующий формат: *variable\_name.element\_name* (имя переменной, затем, через точку, имя элемента первого уровня и т. д. ). В качестве элемента может использоваться другая сложная структура. Элементом также может быть массив, состоящий из элементов как стандартного типа данных, так и элементов сложного типа данных. Вы не можете присваивать значение одной переменной сложного типа, заданного оператором **Type**, другой сложной переменной в форме *var\_name = var\_name*.

### Пример:

```
Type Person
  fullname As String
  age As Integer
  dateofbirth As Date
End Type

Dim sales_mgr, sales_people(10) As Person

sales_mgr.fullname = "Otto Carto"
sales_people(1).fullname = "Melinda Robertson"
```

### См. также:

**Оператор Dim, Оператор Global, Оператор ReDim**

---

## Функция UBound( )

### Назначение

Возвращает размерность массива.

**Синтаксис**

`UBound( array )`, где

*array* – имя массива переменных.

**Возвращаемая величина**

Целое число типа Integer.

**Описание**

Функция **UBound( )** возвращает текущую размерность массива переменных.

Каждая переменная массива имеет размерность от нуля и выше. Этот размер указывается операторами **Оператор Dim** или **Оператор Global**. Размерность массивов может также изменяться по ходу программы **Оператор ReDim**. Функция **UBound( )** возвращает текущее значение размерности на момент вызова функции. В 16-битной версии Windows массивы MapBasic могут иметь размерность от 0 до 7000 включительно, и, следовательно, результат функции будет находиться в этом диапазоне. В 32-битной версии Windows массивы могут иметь размерность до 32 767 элементов.

**Пример:**

```
Dim matrix(10) As FloatDim depth As Integerdepth = UBound(matrix) '
переменная depth сейчас имеет значение 10ReDim matrix(20) depth =
UBound(matrix) ' переменная depth сейчас имеет значение 20
```

**См. также:**

**Оператор Dim, Оператор Global, Оператор ReDim**

---

**Функция UCase\$( )****Назначение**

Возвращает строку, в которой все буквы будут заглавными.

**Синтаксис**

`UCase$( string_expr )`, где

*string\_expr* - выражение, результат которого есть строка.

**Возвращаемая величина**

Строка

**Описание**

Функция **UCase\$( )** возвращает строку, преобразуя все строчные буквы в прописные, в строке, заданной параметром *string\_expr*.

Преобразованию подвергаются только буквы латинского и русского алфавитов. Цифры и другие символы остаются такими же, какими они были в строке `string_expr`. Например, функция `UCase$( "A#12a" )` возвращает строку "A#12A".

### Пример:

```
Dim regular, upper_case As String regular = "Вышний Волочек" upper_case = UCase$(regular) ' переменная lower_case теперь равна "ВЫШНИЙ ВОЛОЧЕК"
```

### См. также:

**Функция LCase\$( ), Функция Proper\$( )**

---

## Оператор UnDim

### Назначение

Распускает переменную.

### Синтаксис

`UnDim variable_name`, где

`variable_name` – имя переменной, которая была объявлена в Рабочем Наборе или окне MapBasic.

### Предупреждение

Оператор **UnDim** не может использоваться в компилированных программах MapBasic. Его использование разрешено только в Рабочем Наборе или в окне MapBasic..

### Описание

После выполнения оператора **Оператор Dim**, создавшего переменную, Вы можете использовать оператор **UnDim** для освобождения ресурсов, отведенных для этой переменной. Например, Вы ввели в окно MapBasic оператор **Оператор Dim**, объявляющий целочисленную переменную X:

```
Dim X As Integer
```

Но Вам стало необходимо изменить тип переменной. Следующие операторы переопределяют переменную X:

```
UnDim X
Dim X As Float
```

### См. также:

**Оператор Dim, Оператор ReDim**



## Функция UnitAbbr\$( )

### Назначение

Возвращает сокращенное имя стандартной единицы измерения в MapInfo.

### Синтаксис

`UnitAbbr$( unit_name )`, где

*unit\_name* – строка, представляющая стандартное имя единицы измерения MapInfo (например, "km").

### Возвращаемая величина

Строка, представляющая сокращенное название единицы измерения (например, "km")

### Описание

Параметр *unit\_name* – это строка, представляющая единицу измерения в MapInfo в англоязычном стандарте. Например, "km" (километр) or "sq km" (квадратный километр).

Функция **UnitAbbr\$( )** возвращает сокращенный вариант имени единицы измерения. Возвращаемая строка зависит от языковой версии MapInfo: там, где в английской и некоторых европейских версиях следующая функция возвратит "sq km", в русской будет возвращено "кв. км".

```
UnitAbbr$ ("sq km")
```

Список названий единиц измерения MapInfo, которые могут быть использованы в качестве аргументов функции **UnitAbbr\$( )**, приведен в описании операторов Set Distance Units, Set Area Units и Set Paper Units. Список стандартных единиц измерения расстояния, принятых в MapInfo Professional смотрите в **Оператор Set Distance Units на стр. 103**. Список единиц измерения площади (например, "sq km") смотрите в **Оператор Set Area Units на стр. 88**. Список "бумажных" единиц измерения (например, "in") смотрите в **Оператор Set Paper Units на стр. 141**. Единицы измерения времени включают: секунды ("sec"), минуты ("min"), и часы ("hr").

В качестве параметра *unit\_name* можно также задать "degree" (функция **UnitAbbr\$( )** возвратит "deg" или "град").

### См. также:

**Оператор Set Area Units, Оператор Set Distance Units, Оператор Set Paper Units, Функция UnitName\$( )**

## Функция UnitName\$( )

### Назначение

Возвращает полное имя стандартной в MapInfo единицы измерения.

### Синтаксис

`UnitName$( unit_name )`, где

*unit\_name* – строковая величина с именем стандартной в MapInfo единицы измерения (например, "km").

### Возвращаемая величина

Строка, представляющая полное название единицы измерения (например, "kilometers").

### Описание

Параметр *unit\_name* – это строка, представляющая единицу измерения в MapInfo в англоязычном стандарте. Например, "km" (километр) or "sq km" (квадратный километр).

Функция **UnitName\$( )** возвращает строку с полной версией имени единицы измерения. Возвращаемая строка зависит от языковой версии MapInfo: там, где в английской и некоторых европейских версиях следующая функция возвратит "square kilometers", в русской будет возвращено "квадратные километры".

```
UnitName$( "sq km" )
```

Список названий единиц измерения MapInfo, которые могут быть использованы в качестве аргументов функции **UnitName\$( )**, приведен в описании операторов Set Distance Units, Set Area Units и Set Paper Units. Список стандартных единиц измерения расстояния, принятых в MapInfo Professional смотрите в [Оператор Set Distance Units на стр. 103](#). Список единиц измерения площади (например, "sq km") смотрите в [Оператор Set Area Units на стр. 88](#). Список "бумажных" единиц измерения (например, "in") смотрите в [Оператор Set Paper Units на стр. 141](#). Единицы измерения времени включают: секунды ("sec"), минуты ("min"), и часы ("hr").

В качестве параметра *unit\_name* можно также задать "degree" (функция **UnitName\$( )** возвратит "degrees" или "градусы").

### См. также:

[Оператор Set Area Units](#), [Оператор Set Distance Units](#), [Оператор Set Paper Units](#), [Функция UnitAbbr\\$\( \)](#)

---

## Оператор Unlink

### Назначение

Разрыв связи с таблицей, которая была загружена с удаленной базой данных оператором [Оператор Server Link Table](#).

### Синтаксис

`Unlink TableName`, где

*TableName* – имя открытой связанной таблицы MapInfo.

## Описание

Отсоединение таблицы удаляет связь с удаленной базой данных. Этот оператор не работает, если в таблице есть изменения (другими словами, пользователь должен сначала сохранить или отменить изменения в таблице). Все метаданные, описывающие инструкции связи, удаляются. Поля, которые были помечены как неизменяемые, после разрыва связей становятся изменяемыми. В результате разъединения появится обычная базовая таблица MapInfo.

## Пример:

```
Unlink "City_1k"
```

## См. также:

[Оператор Commit Table](#), [Оператор Server Link Table](#)

# Оператор Update

## Назначение

Изменяет одну или более строк в таблице.

## Синтаксис

```
Update table Set column = expr [, column = expr, ...] [Where RowID = idnum ], где
```

*table* – имя открытой таблицы;

*column* – имя колонки в таблице;

*expr* – выражение для колонки;

*idnum* – номер строки в таблице.

## Описание

Оператор **Update** изменяет одну или более колонок в таблице. По умолчанию, оператор **Update** обновляет все строки таблицы *table*. Если в операторе используется предложение **Where RowID**, обновляются только указанные строки. В предложении **Set** определяются сами изменения в полях заданной строки или строк.

Используя имя *Obj* для специальной колонки графических объектов, присоединенных к строкам таблицы, Вы можете присоединять новые графические объекты к записям. Смотрите третий пример.

## Примеры

Мы имеем данные о служащих. Каждая запись содержит отдел, в котором работает служащий, и его жалование. Теперь повысим жалование служащим отдела управления продажами, жалование которых было меньше \$20,000, на 7%. Для выбора записей о служащих, которым надо повысить жалование, используем оператор [Оператор Select](#), а оператор **Update**, чтобы привести в соответствие колонку с жалованием .

## Оператор Update Window

---

```
Select * From employees Where department ="отдел_управления_продаж" And  
salary < 20000Update SelectionSet salary = salary * 1.07
```

Используя предложение **Where RowID**, Вы можете указать MapBasic применять операцию **Set** только к определенной строке таблицы. Теперь повысим жалование служащего, данные которого находятся в десятой записи:

```
Update employees  
Set salary = salary * 1.07  
Where Rowid = 10
```

Создадим точечный объект и присоединим его к первой записи таблицы:

```
Update sites  
Set Obj = CreatePoint(x, y)  
Where Rowid = 1
```

**См. также:**

[Оператор Insert](#)

---

## Оператор Update Window

### Назначение

Форсирует обновление изображения в окне.

### Синтаксис

**Update Window** *window\_id*, где  
*window\_id* – идентификатор окна.

### Описание

Оператор **Update Window** обновляет изображение в одном из окон MapInfo Professional.

В некоторых ситуациях операции в окне не отображаются сразу и увидеть изменения можно только после ближайшего обновления окна. Например, если программа использует оператор [Оператор Dialog](#), и из обработчика элементов диалога производятся изменения в окне Карты, то новое изображение пользователь увидит только после закрытия диалогового окна. Предписать MapInfo Professional обновить изображение можно с помощью оператора **Update Window**.

**См. также:**

[Оператор Set Event Processing](#)

---

## Функция Val( )

### Назначение

Возвращает численную величину, извлеченную из строки.

**Синтаксис**

**Val**( *string\_expr* ), где

*string\_expr* - выражение, результат которого есть строка.

**Возвращаемая величина**

Вещественное число типа Float.

**Описание**

Функция **Val**( ) возвращает число, выделяя его из строки, определенной выражением *string\_expr*. Считывание числа начинается с начала строки и заканчивается первым нечисленным символом. При этом функция **Val**( ) игнорирует пробелы, символы табуляции и новой строки в начале строки *string\_expr*. Если первый символ строки не является числом, одним из трех символов, описанных выше, точкой, знаком минус или плюс, амперсандом (&), то функция **Val**( ) вернет 0. Амперсанд используется для шестнадцатеричных чисел.

**Внимание:** Если строка включает разделитель целой части числа и десятичной, то этот знак должен быть точкой, независимо от того, какой стандарт форматирования чисел используется в компьютере пользователя. Строка также не должна содержать разделители тысяч. Для удаления разделителей тысяч используйте функцию **Функция DeformatNumber\$( )**.

**Пример:**

```
Dim f_num As Float
f_num = Val("12 тысяч") ' f_num равно 12
f_num = Val("12,345") ' f_num равно 12
f_num = Val(" 52 - 62 дома") ' f_num равно 52
f_num = Val("Восемнадцать") ' f_num равно 0 (zero)
f_num = Val("&H1A") ' f_num равно 26 (равно шестнадцатеричному 1A)
```

**См. также:**

**Функция DeformatNumber\$( )**, **Функция Format\$( )**, **Оператор Set Format**, **Функция Str\$( )**

---

**Функция Weekday( )****Назначение**

Возвращает целое число от 1 до 7, соответствующее дню недели.

**Синтаксис**

**Weekday**( *date\_expr* ), где

*date\_expr* – выражение, результат которого есть дата (величина типа Data).

**Возвращаемая величина**

Короткое целое число от 1 до 7 включительно. Величина типа SmallInt.

### Описание

Функция **Weekday( )** возвращает номер дня в неделе. Число 1 соответствует Воскресенью. Если, например, день, заданный выражением *date\_expr*, является Вторником, то результатом функции будет 3.

Функция **Weekday( )** работает с датами, значения которых – не ранее первого января 100 года. Если выражение *date\_expr* принимает значение даты ранее указанного значения, функция **Weekday( )** вернет ноль.

### Пример:

```
If Weekday( CurDate( ) ) = 6 Then ' сегодня ПЯТНИЦА ' End If
```

### См. также:

[Функция CurDate\( \)](#), [Функция Day\( \)](#), [Функция Month\( \)](#), [Функция Year\( \)](#)

---

## Оператор WFS Refresh Table

### Назначение

Обновляет таблицу WFS с сервера

### Синтаксис

```
WFS Refresh Table alias, где
```

*alias* - псевдоним, выбранный для зарегистрированной таблицы WFS.

### Пример:

Следующий пример демонстрирует обновление локальной таблицы под названием watershed.

```
WFS Refresh Table watershed
```

### См. также:

[Оператор Register Table](#), [Функция TableInfo\( \)](#)

---

## Оператор While...Wend

### Назначение

Циклически выполняет определенные действия, пока истинно определенное условие.

### Синтаксис

```
While condition statement_list Wend, где
```

*condition* – выражение, управляющее выполнением цикла;

*statement\_list* - группа операторов, выполняющаяся за один проход цикла.

## Предупреждение

Вы не можете использовать оператор **While...Wend** в окне MapBasic.

## Описание

Оператор **While...Wend** является одной из конструкций цикла. MapBasic проверяет условие, заданное выражением *condition*. Если условие истинно, MapBasic выполняет операторы *statement\_list*.

Далее снова проверяется условие *condition*, и, если оно истинно, все повторяется. Цикл выполняется до тех пор, пока значение *condition* не станет ложным. После этого MapBasic пропустит операторы *statement\_list* и передаст выполнение программы следующему после **Wend** оператору.

Заметим, что конструкция:

```
While condition statement_list Wend, где
```

фактически идентична конструкции:

```
Do While condition
    statement_list
Loop
```

Цикл **While...Wend** может быть заменен одной из форм оператора **Оператор Do...Loop**. Использование в программе цикла **While... Wend** обуславливается стилистическими предпочтениями каждого программиста.

## Пример:

```
Dim psum As Float, i As Integer
Open Table "world"
Fetch First From world
i = 1
While i <= 10
    psum = psum + world.population
    Fetch Next From world
    i = i + 1
Wend
```

См. также:

**Оператор Do...Loop, Оператор For...Next**

## Процедура WinChangedHandler

### Назначение

Процедура, автоматически выполняющаяся при перемещении или увеличении/уменьшении изображения в окне Карты, а также при добавлении или удалении слоя.

### Синтаксис

```
Declare Sub WinChangedHandler Sub WinChangedHandler statement_list End  
Sub, где
```

*statement\_list* – список операторов процедуры.

### Описание

WinChangedHandler – зарезервированное имя для процедуры MapBasic. Когда пользователь запускает программу, имеющую процедуру WinChangedHandler, программа не завершается после выполнения всех операторов процедуры Main и других процедур, вызванных из нее. Программа будет пребывать в режиме ожидания до тех пор, пока не произойдет перемещение и масштабирование Карты в окне или изменение размеров самого окна Карты. В процедуре WinChangedHandler может быть использована функция **CommandInfo( ) функция** для распознавания идентификатора окна, в котором произошло изменение.

Одновременно в состоянии ожидания могут находиться несколько прикладных программ. Поэтому при изменении в окне Карты, автоматически выполняются все процедуры **WinChangedHandler** из этих программ, одна за другой.

В некоторых случаях MapBasic вызывают процедуру **WinChangedHandler** на события, не связанные с изменениями размера окна. Например, рисование нового объекта вызывает процедуру **WinChangedHandler**. Чтобы закрыть обработчик и удалить его из памяти, применяйте оператор **Оператор End Program**.

### Автопрокрутка в окне Карты

MapInfo версии 4.0 автоматически сдвигает изображение в окне Карты, когда пользователь, используя, например, инструмент рисования прямоугольника, растягивая контур будущего объекта, подводит указатель мышки к краю окна. Если действия пользователя привели к автоматическому сдвигу изображения окна, MapInfo вызывает процедуру **WinChangedHandler** после выполнения или отмены действий инструмента.

Например, если Вы использовали инструмент Линейка и рисование каждого сегмента приводит к автопрокрутке, то MapInfo вызовет **WinChangedHandler** тогда, когда Вы закончите измерения двойным щелчком мышки или нажатием клавиши ESC. Если автопрокрутка произошла вследствие применения пользователем инструмента MapBasic, то MapInfo сначала вызовет обработчик инструмента, а затем процедуру **WinChangedHandler**.

MapInfo не будет вызывать процедуру **WinChangedHandler**, если пользователь вернет в предыдущее состояние изображение окна, в котором оно находилось перед автопрокруткой, или нажмет на клавишу ESC.



Автоматический сдвиг изображения Карты можно отключить с помощью оператора **Оператор Set Window**.

### Пример:

Пример использования процедуры **WinChangedHandler** смотрите в тексте программы OVERVIEW, которая входит в стандартную поставку MapBasic.

См. также:

**CommandInfo( )** функция, Процедура **WinClosedHandler**

---

## Процедура WinClosedHandler

### Назначение

Процедура автоматически выполняется при закрытии окна Карты, Списка, Графика, Отчета, Геогруппы или MapBasic.

### Синтаксис

```
Declare Sub WinClosedHandler Sub WinClosedHandler statement_list End
Sub, где
```

*statement\_list* – список операторов процедуры.

### Описание

**WinClosedHandler** – зарезервированное имя для процедуры MapBasic. Когда пользователь запускает программу, имеющую процедуру **WinClosedHandler**, программа не завершается после выполнения всех операторов процедуры Main и других процедур, вызванных из нее. Программа будет находиться в режиме ожидания до тех пор, пока пользователь не закроет какое-нибудь окно. Программа будет находиться в режиме ожидания до тех пор, пока пользователь не закроет какое-нибудь окно. Как только это произойдет, программа активизируется, выполняя процедуру с именем **WinClosedHandler**. После выполнения процедуры программа вновь переходит в режим ожидания.

В процедуре **WinChangedHandler** может быть использована функция

```
CommandInfo( CMD_INFO_WIN )
```

для распознавания идентификатора окна, в котором произошло изменение.

**Внимание:** Когда процедура закончит свои действия (**Оператор End Program**), прикладная программа вновь переходит в режим ожидания. И так всякий раз при новом применении инструмента. После использования оператора **Оператор End Program** полностью освобождается память, занимаемая всей прикладной программой, кнопка инструмента убирается с панели. Поэтому, если Вам снова понадобятся функции, выполняемые инструментом MapBasic, не используйте оператор **Оператор End Program** в процедуре **WinClosedHandler**, и оставьте программу в режиме ожидания до нового выбора инструмента.

## Функция WindowID( )

Одновременно в состоянии ожидания могут находиться несколько прикладных программ. Поэтому при изменении в окне Карты, автоматически выполняются все процедуры **WinClosedHandler** из этих программ, одна за другой.

См. также:

**CommandInfo( )** функция, Процедура **EndHandler**, Процедура **RemoteMsgHandler**, Процедура **SelChangedHandler**, Процедура **ToolHandler**, Процедура **WinChangedHandler**

## Функция WindowID( )

### Назначение

Возвращает идентификатор окна, заданного его номером на экране.

### Синтаксис

**WindowID**( *window\_num* ), где  
*window\_num* – величина типа SmallInt, номер окна.

### Возвращаемая величина

Целое число типа Integer.

### Описание

Функция **WindowID( )** возвращает уникальный номер окна. Некоторые операторы MapBasic, такие как **Оператор Set Map**, используют идентификатор в качестве параметра.

В следующей таблице приводятся возможные способы задания параметра *window\_num*:

| Значение <i>window_num</i>   | Результат   |
|--|---|
| Положительное целое число (величина типа Smallint) (1, 2, ... <i>n</i> )   | MapInfo возвращает идентификатор документального окна, такого, как Карта, Список. Например, если задана единица, то MapInfo возвращает идентификатор первого документального окна. Заметим, что значение <i>n</i> можно получить с помощью функции <b>Функция NumWindows( )</b>   |
| Отрицательное целое число (величина типа Smallint) (-1,-2, ...- <i>m</i> ) | MapInfo возвращает идентификатор как документального окна, так и другого плавающего окна, например, Информация. Заметим, что значение <i>m</i> можно получить с помощью функции <b>Функция NumAllWindows( )</b> . Используя этот синтаксис, Вы можете вызывать функцию <b>WindowID( )</b> в цикле для построения списка всех открытых окон. |

| Значение <code>window_num</code> | Результат   |
|----------------------------------|---|
| Ноль ( 0 )                       | MapInfo возвращает ID-номер окна либо последнего из открытых документов, либо легенды, созданной пользователем, либо инструментальной панели; или ноль, если окна не открывались.   |
| Код окна (например, WIN_RULER)   | Если Вы задали код от 1001 до 1013, то MapInfo вернет идентификатор соответствующего специального окна. Коды типов окон определены в MAPBASIC.DEF. Например, код WIN_RULER, имеющий значение 1007, представляет окно Линейка. |

**Ошибки:**

ERR\_BAD\_WINDOW\_NUM, если неверно значение аргумента.

**См. также:**

Функция `FrontWindow( )`, Функция `NumWindows( )`

**Функция WindowInfo( )**

**Назначение**

Возвращает информацию об открытом окне.

**Синтаксис**

`WindowInfo( window_spec, attribute )`, где  
*window\_spec* – целочисленный идентификатор окна;  
*attribute* – целое число, код необходимой информации.

**Возвращаемая величина**

Тип результата зависит от значения параметра *attribute*.

**Описание**

Функция **WindowInfo( )** используется для получения определенной информации об открытом окне.

Многие значения, используемые в качестве параметров функции **WindowInfo( )**, определены в файле MAPBASIC.DEF. Вы можете их использовать, если в начале программы есть строка Include "MAPBASIC.DEF".

В следующей таблице приводятся возможные способы задания параметра *window\_num*:

| Значение window_spec   | Описание   |
|--|--|
| Целочисленный идентификатор  | Для задания окна, о котором хотите получить информацию можно использовать его идентификатор, который, в свою очередь, можно получить с помощью функций <b>Функция WindowID( )</b> или <b>Функция FrontWindow( )</b> .  |
| Положительное целое число (величина типа Smallint) (1, 2, ... <i>n</i> )   | Функция возвращает информацию об окне документа, таком как Карта или Список. Например, если задана единица, то MapInfo возвращает информацию о первом окне документа. Заметим, что значение <i>n</i> можно получить с помощью функции <b>Функция NumWindows( )</b>   |
| Отрицательное целое число (величина типа Smallint) (-1,-2, ...- <i>m</i> ) | Функция возвращает информацию как об окне документа, так и о другом плавающем окне, таком, как Информация. Заметим, что значение <i>m</i> можно получить с помощью функции <b>Функция NumAllWindows( )</b> . Используя этот синтаксис, Вы можете вызывать функцию <b>WindowInfo( )</b> в цикле для построения списка информации обо всех открытых окнах. |
| Ноль ( 0 )   | Функция опрашивает последнее из открывавшихся окон. Если окна не открыты, порождается ошибка.  |
| Код окна (например, WIN_RULER)   | Если Вы задали код от 1001 до 1013, то MapInfo вернет информацию о соответствующем специальном окне. Коды типов окон определены в MAPBASIC.DEF. Например, код WIN_RULER, имеющий значение 1007, представляет окно Линейка.   |

Теперь рассмотрим зависимость результата от значения параметра `attribute`. Параметр `attribute` должен являться одним из кодов, показанных в нижеследующей таблице.

| Значения <code>attribute</code>         | <code>WindowInfo( attribute )</code> возвращает:   |
|---|--|
| <code>WIN_INFO_AUTOSCROLL</code> (17)   | Логическая величина: "Да" (TRUE), если режим автоматического сдвига включен. Изменение этого режима делается оператором <b>Оператор Set Window</b> .   |
| <code>WIN_INFO_CLONEWINDOW</code> (15)  | Строка с оператором MapBasic, который может быть использован в операторе <b>Оператор Run Command</b> для дублирования окна.  |
| <code>WIN_INFO_HEIGHT</code> (5)        | Число типа Float, высота окна в "бумажных" единицах измерения, установленных оператором <code>Set Paper Units</code> .   |
| <code>WIN_INFO_LEGENDS_MAP</code> (10)  | Целое число: если вы составляете запрос об окне Легенды, открытом оператором <b>Оператор Create Legend</b> , то результатом будет идентификатор соответствующего окна Карты или Графика. Если окно Легенды стандартно, результатом будет 0.                                  |
| <code>WIN_INFO_NAME</code> (1)          | Строка с именем окна.  |
| <code>WIN_INFO_OPEN</code> (11)         | Логическая величина, определяющая, открыто ли окно (используется для таких окон как "Статистика").   |
| <code>WIN_INFO_SMARTPAN</code> (18)     | Логическая величина; "Да" (TRUE) если режим <b>Smart Pan</b> включен.  |
| <code>WIN_INFO_STATE</code> (9)         | Короткое целое число (тип <code>SmallInt</code> ):<br><code>WIN_STATE_NORMAL</code> , если окно раскрыто, но меньше окна MapInfo;<br><code>WIN_STATE_MINIMIZED</code> , если окно свернуто в иконку; <code>WIN_STATE_MAXIMIZED</code> , если окно имеет максимальный размер. |
| <code>WIN_INFO_SYSMENUCLOSE</code> (16) | Логическая величина: "Нет" (FALSE), если оператор <b>Оператор Set Window</b> нейтрализовал команду <b>Close</b> из системного меню окна.   |

| Значения attribute           | WindowInfo( attribute ) возвращает:   |
|------------------------------|---|
| WIN_INFO_TABLE (10)          | Строка с именем временной таблицы:<br>"CosmeticN", если определено окно Карты;<br>"LayoutN", если определено окно Отчета; где N – номер окна. Для окон Списка и Графика результатом будет имя таблицы, показанной в окне.             |
| WIN_INFO_TOPMOST (8)         | Логическая величина. Если окно активно, значение будет истинно (TRUE).  |
| WIN_INFO_TYPE (3)            | Число типа SmallInt, определяющее тип окна (например, WIN_LAYOUT). См. таблицу ниже.  |
| WIN_INFO_WIDTH (4)           | Число типа Float, ширина окна в "бумажных" единицах измерения, установленных оператором Set Paper Units.  |
| WIN_INFO_WINDOWID (13)       | Целое число, идентификатор окна. Результат такой же, как у функции <b>Функция WindowID( )</b> . Это можно использовать для передачи нуля как <i>window_spec</i> .   |
| WIN_INFO_WND (12)            | Целое число. В среде Windows представляет Windows <i>HWND</i> для опрашиваемого окна.   |
| WIN_INFO_WORKSPACE (14)      | Строка с операторами MapBasic, с помощью которых запоминается Карта в Рабочем Наборе. Отличается от результата функции с кодом WIN_INFO_CLONEWINDOW тем, что результат включает в себя операторы <b>Оператор Open Table</b> и другие. |
| WIN_INFO_X (6)               | Число типа Float, расстояние от левого края нашего окна до левого края рабочего поля MapInfo (в "бумажных" единицах).   |
| WIN_INFO_Y (7)               | Число типа Float, расстояние от верхнего края нашего окна до верхнего края рабочего поля MapInfo (в "бумажных" единицах).   |
| WIN_INFO_PRINTER_NAME (21)   | Возвращает строку с идентификатором принтера (например, \\DISCOVERY\HP4_DEVEL)  |
| WIN_INFO_PRINTER_ORIENT (22) | Возвращает Returns<br>WIN_PRINTER_PORTRAIT или<br>WIN_PRINTER_LANDSCAPE   |

| Значения attribute                 | WindowInfo( attribute ) возвращает:  |
|------------------------------------|--|
| WIN_INFO_PRINTER_COPIES (23)       | Возвращает целое число копий.  |
| WIN_INFO_SNAPMODE (19)             | Возвращает логическую величину. TRUE - совмещение узлов включено. FALSE - если режим совмещения узлов отключен.  |
| WIN_INFO_SNAPTHRESHOLD (20)        | Возвращает короткое целое, устойчивость в пикселах.  |
| WIN_INFO_PRINTER_PAPERSIZE (24)    | Целое число. Смотрите в файле Papersize.def объяснение значений возвращаемой величины.   |
| WIN_INFO_PRINTER_LEFTMARGIN (25)   | Вещественное: левое поле области печати в текущих единицах длины.  |
| WIN_INFO_PRINTER_RIGHTMARGIN (26)  | Вещественное: правое поле области печати в текущих единицах длины.   |
| WIN_INFO_PRINTER_TOPMARGIN (27)    | Вещественное: верхнее поле области печати в текущих единицах длины.  |
| WIN_INFO_PRINTER_BOTTOMMARGIN (28) | Вещественное: нижнее поле области печати в текущих единицах длины.   |
| WIN_INFO_PRINTER_BORDER (29)       | Строковая величина: ON если рамка вокруг окна изображается при печати, OFF если нет.   |
| WIN_INFO_PRINTER_TRUECOLOR (30)    | Строковая величина: ON если используется 24-битный true color для печати растров и сеток (grid). Это бывает когда изображение 24 битное и принтер поддерживает более 256 цветов, OFF в противном случае.   |
| WIN_INFO_PRINTER_DITHER (31)       | Строковая величина: возвращает метод растеризации, который используется если надо конвертировать 24-битное изображение в 256 цветов. Возвращаемые значения HALFTONE и ERRORDIFFUSION. Эта настройка используется при печати растров и сеточных файлов (grid). Растеризация произойдет, если WIN_INFO_PRINTER_TRUECOLOR невозможно или если принтер поддерживает только 256 цветов или менее. |
| WIN_INFO_PRINTER_METHOD (32)       | Строковая величина: возвращает значения DEVICE и EMF.  |

| Значения attribute                   | WindowInfo( attribute ) возвращает:  |
|--------------------------------------|--|
| WIN_INFO_PRINTER_TRANSPRSTER (33)    | Строковая величина: возвращает значения DEVICE и INTERNAL.   |
| WIN_INFO_PRINTER TRANSPVECTOR (34)   | Строковая величина: возвращает значения DEVICE и INTERNAL.   |
| WIN_INFO_EXPORT_BORDER (35)          | Строковая величина: возвращает значения ON и OFF.  |
| WIN_INFO_EXPORT_TRUECOLOR (36)       | Строковая величина: возвращает значения ON и OFF.  |
| WIN_INFO_EXPORT_DITHER (37)          | Строковая величина: возвращает значения HALFTONE и ERRORDIFFUSION.   |
| WIN_INFO_EXPORT_TRANSPRSTER (38)     | Строковая величина: возвращает значения DEVICE и INTERNAL.   |
| WIN_INFO_EXPORT_TRANSPVECTOR (39)    | Строковая величина: возвращает значения DEVICE и INTERNAL.   |
| WIN_INFO_PRINTER_SCALE_PATTERNS (40) | Логическая величина. TRUE - если окно масштабировано для печати. FALSE - если не масштабировано.   |
| WIN_INFO_EXPORT_ANTIALIASING (41)    | Строковое значение: ON, если фильтр сглаживания будет использоваться при экспорте и OFF в противном случае.  |
| WIN_INFO_EXPORT_THRESHOLD (42)       | Целое значение от 0 до 255, определяющее порог сглаживания.  |
| WIN_INFO_EXPORT_MASKSIZE (43)        | Целое значение между 0 и 100   |
| WIN_INFO_EXPORT_FILTER (44)          | Целое значение, возвращающее один из доступных фильтров сглаживания: <ul style="list-style-type: none"> <li>• FILTER_VERTICALLY_AND_HORIZONTALLY</li> <li>• FILTER_ALL_DIRECTIONS_1</li> <li>• FILTER_ALL_DIRECTIONS_2</li> <li>• FILTER_DIAGONALLY</li> <li>• FILTER_HORIZONTALLY</li> <li>• FILTER_VERTICALLY</li> </ul> |

Если Вы используете в качестве параметра *attribute* код WIN\_INFO\_TYPE, функция **WindowInfo( )** вернет код вида окна. Имена для этого кода перечислены в первой колонке в следующей таблицы:



| Код в результате | Описание окна   |
|------------------|---|
| WIN_MAPPER       | Окно Карты  |
| WIN_BROWSER      | Окно Списка   |
| WIN_LAYOUT       | Окно Отчета   |
| WIN_GRAPH        | Окно Графика  |
| WIN_HELP         | Окно Справочной системы   |
| WIN_MAPBASIC     | Окно MapBasic   |
| WIN_MESSAGE      | Окно "Сообщение" (вызывается оператором <b>Оператор Print</b> ) |
| WIN_RULER        | "Линейка" (вызывается инструментом Линейка)                     |
| WIN_INFO         | Окно "Информация" (вызывается инструментом Информация)          |
| WIN_LEGEND       | Окно "Легенда"  |
| WIN_STATISTICS   | Окно "Статистика"   |
| WIN_MAPINFO      | Рабочее окно программы MapInfo                                  |
| WIN_BUTTONPAD    | Окно панели инструментов  |
| WIN_TOOLBAR      | Окно инструментальной панели                                    |
| WIN_CART_LEGEND  | Окно картографической легенды                                   |
| WIN_3DMAP        | Окно 3D-карты   |

Каждое окно Карты создает специальную временную таблицу, которая содержит данные для Косметического слоя Карты. Эти таблицы (имеющие имена "Cosmetic1", "Cosmetic2" и т.д.) пользователь не видит. Вы можете получить эти имена от функции WindowInfo( ) используя код WIN\_INFO\_TABLE. Аналогично, окна Отчетов также поддерживают временные скрытые от пользователя таблицы с именами вида "Layout1", "Layout2" и т.д.

#### Ошибки:

ERR\_BAD\_WINDOW\_NUM, если неверно значение параметра *window\_id*.

ERR\_FCN\_ARG\_RANGE, если неправильно задано значение аргумента.

#### Пример:

В следующем примере открывается окно Статистика, если оно уже не открыто.

```
If Not WindowInfo(WIN_STATISTICS,WIN_INFO_OPEN) Then
    Open Window WIN_STATISTICS
End If
```

См. также:

[Оператор Browse](#), [Оператор Graph](#), [Оператор Map](#)

---

## Процедура WinFocusChangedHandler

### Назначение

Процедура, автоматически выполняющаяся при изменении фокуса окна.

### Синтаксис

```
Declare Sub WinFocusChangedHandler Sub WinFocusChangedHandler  
statement_list End Sub, где
```

### Описание

Если загруженная программа имеет процедуру **WinFocusChangedHandler**, то MapInfo автоматически выполняет процедуру, когда меняется фокус окна (фокус имеет активное окно). Фокус может перемещаться между всеми типами окон MapInfo (Списки, Карты и т.п.). Для определения идентификатора окна, ставшего активным, используйте в процедуре обработчика функцию CommandInfo(CMD\_INFO\_WIN).

В процедуре **WinFocusChangedHandler** не может быть использован оператор [Оператор Note](#), процедура обработчика также не может закрывать или открывать какие-либо окна. Эти ограничения аналогичны тем, которые имеют и другие обработчики, например [Процедура SelChangedHandler](#).

Процедура **WinFocusChangedHandler** должна быть как можно короче, чтобы не замедлять работу системы.

### Пример:

Следующий фрагмент текста программы показывает, как можно делать доступной или недоступной команду меню в зависимости от того, активно ли окно Карты или нет.

```
Include "mapbasic.def" Include "menu.def"Declare Sub MainDeclare sub  
WinFocusChangedHandlerSub Main' Здесь вместо комментариев должны быть  
операторы, ' создающие элемент меню, который может быть доступным,' когда  
активно окно Карты... End SubSub WinFocusChangedHandler Dim i_win_type As  
SmallInt i_win_type=WindowInfo(CommandInfo(CMD_INFO_WIN),WIN_INFO_TYPE)If  
i_win_type = WIN_MAPPER Then ' Здесь вместо комментариев должен быть  
оператор, Else' который делает элемент меню доступнымEnd If End Sub
```

См. также:

[Процедура WinChangedHandler](#)

## Оператор Write #

### Назначение

Запись данных в открытый файл.

### Синтаксис

**Write #** *file\_num* [ , *expr ...* ], где

*file\_num* – номер файла, который был присвоен ему при открытии;

*expr* – выражение для записи в файл.

### Описание

Оператор **Write #** записывает определенные данные в открытый файл. Он должен быть открыт в режиме последовательного доступа

оператором **Оператор Open File**, который закрепляет за файлом номер, используемый в параметре *file\_num*.

Выражений *expr* может быть несколько, в операторе они должны быть разделены запятыми. В этом случае записанные значения в файле автоматически разделяются запятыми. Строчные значения при записи автоматически снабжаются кавычками. Если список выражений пуст, то записывается пустая строка.

Оператор **Write #** автоматически заключает строковые значения в кавычки при записи в файл. Чтобы записать текст в файл, не используя кавычек, используйте оператор **Оператор Print #**.

Для чтения записей из файла используйте оператор **Оператор Input #**.

См. также:

**Оператор Input #**, **Оператор Open File**, **Оператор Print #**

## Функция Year( )

### Назначение

Извлекает год из значения даты.

### Синтаксис

**Year**( *date\_expr* ), где

*date\_expr* – выражение, результат которого есть дата (величина типа Data).

### Возвращаемая величина

Целое число типа SmallInt.

### Описание

Если оператор **Оператор Set Date Window** принимает значение Off, год определяется часами системы.

### Примеры

Пример демонстрирует, как Вы можете использовать функцию **Year( )** для извлечения из даты только компоненты года.

```
Dim sampleDate as DateSet Date Window
OffsampleDate=StringToDate("01.10.98")Print Year(sampleDate) ' 2098 (или
1998 в случае, если системное время компьютера установлено в формате
1900),' поскольку при выключенном преобразовании дат MapInfo работает с
текущим столетием
Set Date Window 50
' now assume that two-digit dates fall in the period 1950-2049
print Year(sampleDate)
' still 2098, because date variable has already been assigned!
sampleDate=StringToDate("01.10.98")
' re-assign variable now that the date window has changed
print Year(sampleDate) ' 1998
Undim sampleDate
```

Функция **Year( )** в качестве аргумента может принимать не только переменную типа Date, но и строку. В таком случае произойдет неявное преобразование в формат даты. В примере это отображается:

```
Set Date Window OffPrint Year("01.10.99") ' выводит 2099Set Date Window
50Print Year("01.10.99") ' выводит 1999
```

Вы можете также использовать функцию **Year( )** в операторе Select, формирующем SQL-запрос. В этом примере выбираются определенные строки из таблицы Orders. Предполагается, что в таблице Orders есть колонка с датами, которая называется OrderDate. Предложение Where оператора Select предписывает MapInfo Professional выбирать заказы только за декабрь 1993.

```
Open Table "orders"Select * From orders Where Month(orderdate) = 12 And
Year(orderdate) = 1993
```

### См. также:

**Функция CurDate( ), Функция Day( ), Функция DateWindow( ), Функция Month( ), Функция Weekday( )**

# Библиотеки HTTP и FTP

В этом приложении подробно описаны библиотеки HTTP и FTP, с помощью которых программисты на языке MapBasic могут использовать веб-технологии. Эти библиотеки позволяют получать доступ к сведениям рассылаемым по RSS и другим источникам информации о местоположении, основанным на веб-технологиях, например, погоде, ситуации на дорогах, координатах транспортных средств и т.п., а также настраивать соединение по FTP, поиск, получение и отправку файлов из программы на языке MapBasic. Эти библиотеки используют общие DEF-файлы: HTTPLib.DEF, HTTPType.DEF и HTTPUtil.DEF, которые находятся по адресу: <Каталог, в котором установлен MapBasic>\Samples\MapBasic\INC. Эти файлы следует включить в состав секции заголовков программ. Все функции, описанные в этом приложении, также зависят от наличия файла GmlXlat.dll, который устанавливается вместе с MapInfo Professional.

Мы предлагаем несколько примеров приложений, в которых используются эти библиотеки. Примеры ищите по адресу <Каталог, в котором установлен MapBasic>\Samples\MapBasic\HTTPLib and Your MapBasic Installation Directory\Samples\MapBasic\FTPLib.

Все функции и процедуры, описанные в этом приложении, являются интерфейсными обложками соответствующих классов Microsoft MFC. Оболочки классов включают: CInternetSession, CHttpConnection, CFtpConnection, CHttpFile и CFtpFileFind. Более подробная информация об использовании зависимых классов приводится в MSDN в разделе о MFC классах: ([http://msdn2.microsoft.com/en-us/library/bk77x1wx\(en-US,vs.80\).aspx](http://msdn2.microsoft.com/en-us/library/bk77x1wx(en-US,vs.80).aspx)).

**Внимание:** Поскольку используются внешние библиотеки, все функции и процедуры, перечисленные в этом приложении, не могут быть использованы в окне MapBasic MapInfo Professional.

## Процедура MICloseContent( )

### Назначение

Закрывает и удаляет идентификатор CString и освобождает память.

### Синтаксис

```
MICloseContent( ByVal hContent As CString )
```

hContent – идентификатор объекта CString, который требуется удалить.

### Описание

Используйте процедуру MICloseContent( ), для того чтобы закрыть и освободить идентификатор CString, полученный вызовом, в котором используется **Функция MIGetContent( )**, если идентификатор не доступен.

См. также:

**Функция MIGetContent( )**

---

## Процедура MICloseFtpConnection( )

### Назначение

Закрывает и удаляет идентификатор CFtpConnection и освобождает память.

### Синтаксис

```
MICloseFtpConnection( ByVal hConnection As CFtpConnection )
```

hConnection – идентификатор объекта CFtpConnection, который требуется удалить.

### Описание

Используйте процедуру MICloseFtpConnection( ), для того чтобы закрыть и освободить идентификатор CFtpConnection, полученный вызовом, в котором используется **Функция MIGetFtpConnection( )**, если идентификатор не доступен.

См. также:

**Функция MIGetFtpConnection( )**

---

## Процедура MICloseFtpFileFind( )

### Назначение

Закрывает и удаляет идентификатор CFtpFileFind и освобождает память.

**Синтаксис**

```
MICloseFtpFileFind( ByVal hFTPFind As CFtpFileFind )
```

*hFTPFind* – идентификатор удаляемого объекта CFtpFileFind.

**Описание**

Используйте процедуру **MICloseFtpFileFind**( ), для того чтобы закрыть и освободить идентификатор CFtpFileFind, полученный вызовом, в котором используется **Функция MIGetFtpFileFind**( ), если идентификатор не доступен.

См. также:

**Функция MIGetFtpFileFind**( )

---

**Процедура MICloseHttpConnection( )****Назначение**

Закрывает и удаляет идентификатор CHttpConnection и освобождает память.

**Синтаксис**

```
MICloseHttpConnection( ByVal hConnection As CHttpConnection )
```

*hConnection* – идентификатор объекта CHttpConnection, который требуется удалить.

**Описание**

Используйте процедуру **MICloseHttpConnection**( ), для того чтобы закрыть и освободить идентификатор CHttpConnection, полученный вызовом, в котором используется **Функция MIGetHttpConnection**( ), если идентификатор не доступен.

См. также:

**Функция MIGetHttpConnection**( )

---

**Процедура MICloseHttpFile( )****Назначение**

Закрывает и удаляет идентификатор CHttpFile и освобождает память.

**Синтаксис**

```
MICloseHttpFile( ByVal hFile As CHttpFile )
```

*hFile* – идентификатор объекта CHttpFile, который требуется удалить.

## Процедура MICloseSession( )

---

### Описание

Используйте процедуру MICloseHttpFile( ), для того чтобы закрыть и освободить идентификатор CHttpFile, полученный вызовом, в котором используется **Функция MIOpenRequest( )** или **Функция MIOpenRequestFull( )**, если идентификатор не доступен.

### См. также:

**Функция MIOpenRequest( )**, **Функция MIOpenRequestFull( )**

## Процедура MICloseSession( )

---

### Назначение

Закрывает и удаляет идентификатор CInternetSession и освобождает память.

### Синтаксис

```
MICloseSession( ByVal hSession As CInternetSession )
```

hSession – идентификатор объекта CInternetSession, который требуется удалить.

### Описание

Используйте процедуру MICloseSession( ), для того чтобы закрыть и освободить идентификатор CInternetSession, полученный вызовом, в котором используется **Функция MICreateSession( )** или **Функция MICreateSessionFull( )**, если идентификатор не доступен.

### См. также:

**Функция MICreateSession( )**, **Функция MICreateSessionFull( )**

## Функция MICreateSession( )

---

### Назначение

Creates a CInternetSession object and returns the handle to it.

### Синтаксис

```
MICreateSession( ByVal strAgent As String ) As CInternetSession
```

strAgent – строка, определяющая имя приложения или иной службы (сущности), вызывающей интернет-функции (например, "MapInfo Professional"). Если строка пустая, то будет использовано имя приложения.

### Возвращаемая величина

Идентификатор объекта CInternetSession. Если запрос не будет выполнен, будет возвращено значение Null. Для того чтобы определить причину ошибки, обычно используется **Функция MIBGetErrorMessage( )**.



Описание

MICreateSession( ) – самая первая интернет-функция, вызываемая приложением. Для того чтобы создать и инициировать единственный или несколько одновременно открытых интернет-сеансов, а также предать параметры соединения прокси-серверу, используйте CInternetSession. Если требуется выполнить действия с файлами на сервере по определенному протоколу связи (например, HTTP, FTP), необходимо установить соответствующее соединение с этим сервером. Для того чтобы открыть соединение определенного типа к заданной службе, используйте подходящую функцию, такие как: **Функция MIGetHttpConnection( )** или **Функция MIGetFtpConnection( )**. За подробной информацией, обращайтесь к Microsoft MSDN library.

При вызове требуется использовать идентификатор, полученный вызовом, в котором используется **Процедура MICloseSession( )**, если такой идентификатор в момент использования не известен.

См. также:

**Процедура MICloseSession( )**

Функция MICreateSessionFull( )

Назначение

Создает объект CInternetSession и возвращает идентификатор этого объекта.

Синтаксис

```
MICreateSessionFull( ByVal strAgent As String, ByVal dwContext As Integer,
    ByVal dwAccessType As Integer, ByVal strProxyName As String,
    ByVal strProxyBypass As String, ByVal dwFlags As Integer
    As CInternetSession
```

strAgent – строка, определяющая имя приложения или иной службы (сущности), вызывающей интернет-функции (например, “MapInfo Professional”). Если строка пустая, то будет использовано имя приложения.

dwContext – идентификатор контекста операции.

dwAccessType – требующийся тип доступа. Ниже перечислены все использующиеся значения, только любое из них можно использовать:

| dwAccessType value           | Определение   |
|------------------------------|---|
| INTERNET_OPEN_TYPE_PRECONFIG | Соединение по параметрам восстановленным из реестра. По умолчанию используется этот тип организации доступа. Для подключения через TIS-прокси, задайте параметру dwAccessType это значение; реестр будет изменен. |

| dwAccessType value        | Определение                 |
|---------------------------|-----------------------------|
| INTERNET_OPEN_TYPE_DIRECT | Прямое интернет-соединение. |
| INTERNET_OPEN_TYPE_PROXY  | Соединение с CERN-прокси.   |

strProxyName – имя привилегированного CERN прокси-сервера, если параметру dwAccessType присвоено значение INTERNET\_OPEN\_TYPE\_PROXY. По умолчанию эта строка пустая.

strProxyBypass – строка, в которой перечислены адреса запасных серверов. Эти адреса можно пропустить при использовании прокси-сервера. Если задана пустая строка, то список исключений будет прочитан из реестра. Этот параметр имеет смысл только в случае, когда параметру dwAccessType присвоено значение INTERNET\_OPEN\_TYPE\_PROXY.

dwFlags – определяет некоторые настройки кэширования. По умолчанию этому параметру присвоено значение 0. Этому параметру можно присваивать следующие значения:

| значение dwFlag          | Определение  |
|--------------------------|--|
| INTERNET_FLAG_DONT_CACHE | Не кэшировать данные ни локально, ни на серверах-шлюзах.   |
| INTERNET_FLAG_OFFLINE    | Операции загрузки выполняются, используя только постоянное кэширование. Если результат запрос не будет найден в кэше, возвращается соответствующий код ошибки. |

### Возвращаемая величина

Идентификатор объекта CInternetSession. Если запрос не будет выполнен, будет возвращено значение Null. Для того чтобы определить причину ошибки, обычно используется **Функция MIGetErrorMessage( )**.

### Описание

MICreateSessionFull( ) – самая первая интернет-функция, вызываемая приложением. Для того чтобы создать и инициализировать единственный или несколько одновременно открытых интернет-сеансов, а также передать параметры соединения прокси-серверу, используйте CInternetSession. Если требуется выполнить действия с файлами на сервере по определенному протоколу связи (например, HTTP, FTP), необходимо установить соответствующее соединение с этим сервером. Для того чтобы открыть соединение определенного типа к заданной службе, используйте подходящую функцию, такие как: **Функция MIGetHttpConnection( )** или **Функция MIGetFtpConnection( )**. За подробной информацией, обращайтесь к Microsoft MSDN library.

При вызове требуется использовать идентификатор, полученный вызовом, в котором используется **Процедура MICloseSession( )**, если такой идентификатор в момент использования не известен.

См. также:

Функция `MICreateSession( )`, Процедура `MICloseSession( )`

Функция `MIErrorDlg( )`

Назначение

Появится диалог с сообщением об ошибке.

Появится диалог с ошибкой, переданной в `MIErrorDlg`, если соответствующий диалог существует. Кроме того, эта функция проверяет скрытые ошибки идентификаторов заданных `CHttpFile` и, при необходимости, показывает диалог.

Синтаксис

`MIErrorDlg( ByVal hFile As CHttpFile, ByVal dwError As Integer) As Integer`

`hFile` – идентификатор `CHttpFile`.

`dwError` – код ошибки, который присвоен сообщению об ошибке.

Возвращаемая величина

Вовращает либо одно из нижеперечисленных значений, либо код ошибки.

| Код ошибки                 | Описание   |
|----------------------------|--|
| ERROR_SUCCESS              | Функция выполнена полностью. При идентификация появление этого значения указывает, что пользователь нажал кнопку "Отмена".                                       |
| ERROR_CANCELLED            | Выполнение функции было остановлено пользователем.   |
| ERROR_INTERNET_FORCE_RETRY | Указывает, что функции должна повторно выполнить переданный ей запрос. При идентификация появление этого значения указывает, что пользователь нажал кнопку "ОК". |

Описание

Эту функцию следует использовать, для того чтобы сообщение об ошибке выводилось в окне диалога, если соответствующее окно диалога существует. Эта функция также проверяет идентификаторы любых скрытых ошибок и, при необходимости, показывает диалог. За дополнительной информацией, обращайтесь к Microsoft MSDN library. Допустимые коды ошибок:

| Код ошибки                               | Описание   |
|--|--|
| ERROR_INTERNET_HTTP_TO_HTTPS_ON_REDIRECT | Предупреждает пользователя о нулевом пересечении сообщений к и от защищенного сайта.   |
| ERROR_INTERNET_INCORRECT_PASSWORD        | Выводит диалог, запрашивающий имя пользователя и пароль.   |
| ERROR_INTERNET_INVALID_CA                | Предупреждает пользователя, что функция не опознала организацию, выдавшую сертификат для данного Secure Socket Layer (SSL) сайта.  |
| ERROR_INTERNET_POST_IS_NON_SECURE        | Показывает предупреждение о передаче данных на сервер по незащищенному подключению.  |
| ERROR_INTERNET_SEC_CERT_CN_INVALID       | Предупреждает, что параметр Common Name (поле имени хоста) сертификата SSL неправильный. Показывает диалог с недействительным параметром SSL Common Name и позволяет пользователю проверить неправильный сертификат. Кроме того позволяет пользователю выбрать сертификат в ответ на запрос сервера. |
| ERROR_INTERNET_SEC_CERT_DATE_INVALID     | Предупреждает пользователя, что срок действия SSL-сертификата истек.   |

За дополнительной информацией, обращайтесь к Microsoft MSDN library.

**См. также:**

**Функция MIOpenRequest( ), Функция MISendRequest( )**

---

## Функция MIFindFtpFile( )

### Назначение

Ищет FTP-файл по заданному идентификатору CFtpFileFind.

### Синтаксис

```
MIFindFtpFile( ByVal hFTPFind As CFtpFileFind, ByVal strName As String )  
As SmallInt
```

hFTPFind – идентификатор CFtpFileFind.

strDirName – строка с именем файла, который требуется найти. Если строка пустая, будут найдены все файлы по групповому признаку (\*).

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

### Описание

После вызова функции MIFindFtpFile( ), для того чтобы получить первый FTP-файл, можно повторить поиск других FTP-файлов, в котором используется [Функция MIFindNextFtpFile\( \)](#).

См. также:

[Функция MIGetFtpFileFind\( \)](#), [Функция MIFindNextFtpFile\( \)](#), [Процедура MIGetFtpFileName\( \)](#), [Функция MllsFtpDirectory\( \)](#), [Функция MllsFtpDots\( \)](#)

---

## Функция MIFindNextFtpFile( )

### Назначение

Продолжает поиск, начатый запросом, в котором использовалась [Функция MIFindFtpFile\( \)](#) с заданным идентификатором CFtpFileFind.

### Синтаксис

```
MIFindNextFtpFile( ByVal hFTPFind As CFtpFileFind ) As SmallInt
```

hFTPFind – идентификатор CFtpFileFind.

### Возвращаемая величина

Ненулевое значение, если существуют дополнительные файлы; нуль – если найденный файл последний в каталоге или произошла ошибка.

### Описание

Необходимо использовать функцию MifindNextFtpFile( ) хотя бы один раз, прежде чем вызывать любую функцию определения атрибутов, в числе которых: [Процедура MIGetFtpFileName\( \)](#), [Функция MllsFtpDirectory\( \)](#) и [Функция MllsFtpDots\( \)](#).

См. также:

[Функция MIGetFtpFileFind\( \)](#), [Функция MIFindFtpFile\( \)](#), [Процедура MIGetFtpFileName\( \)](#), [Функция MllsFtpDirectory\( \)](#), [Функция MllsFtpDots\( \)](#)

## Функция MIGetContent( )

### Назначение

Получить оглавление файла.

### Синтаксис

```
MIGetContent( ByVal hFile As CHttpFile ) As CString
```

*hFile* – идентификатор CHttpFile.

### Возвращаемая величина

Идентификатор объекта CString с оглавлением файла. Если запрос не будет выполнен, будет возвращено значение Null. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

### Описание

MIGetContent( ) – получает оглавление файла и сохраняет его в объекте CString, в отличие от способа, когда используется [Функция MIGetContentToFile\( \)](#). Следует учитывать, что функции MIGetContentToFile и MIGetContent взаимно исключают друг друга, можно использовать только одну из них за все время существования объекта CHttpFile object.

При вызове требуется использовать идентификатор, полученный вызовом, в котором используется [Процедура MICloseContent\( \)](#), если такой идентификатор в момент использования не известен.

### См. также:

[Функция MIOpenRequest\( \)](#), [Функция MISendRequest\( \)](#), [Процедура MICloseContent\( \)](#), [Функция MIGetContentToFile\( \)](#)

---

## Функция MIGetContentBuffer( )

### Назначение

Получить оглавление в формате строки заданной длины.

### Синтаксис

```
MIGetContentBuffer( ByVal hContent As CString, pBuffer As String,  
ByVal nLen As Integer As SmallInt
```

*hFile* – идентификатор CString.

*pBuffer* – ссылка на строку, в которой будет помещено оглавление файла.

*nLen* is the size of *pBuffer* in number of characters or bytes.

**Возвращаемая величина**

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется **Функция MIGetErrorMessage( )**.

**Описание**

Для того чтобы получить оглавление файла в формате строки заданной длины, используйте функцию MIGetContentBuffer( ). Требуется выделить память для pBuffer, до вызова этой функции.

См. также:

**Функция MIGetContent( )**, **Функция MIGetContentLen( )**, **Функция MIGetContentString( )**.

---

**Функция MIGetContentLen( )****Назначение**

Получить длину оглавления.

**Синтаксис**

```
MIGetContentLen( ByVal hContent As CString ) As Integer
```

hContent – идентификатор CString.

**Возвращаемая величина**

Длина оглавления – число букв или байт. Нуль может означать либо ошибку, либо пустое оглавление. Для того чтобы определить причину ошибки, обычно используется **Функция MIGetErrorMessage( )**.

**Описание**

Для того чтобы получить длину оглавления файла в виде числа букв или байт, используйте функцию MIGetContentLen( ).

См. также:

**Функция MIGetErrorMessage( )**

---

**Функция MIGetContentString( )****Назначение**

Получить оглавление в формате строки.

**Синтаксис**

```
MIGetContentString( ByVal hContent As CString ) As String
```

## Функция `MIGetContentToFile( )`

---

`hContent` – идентификатор `CString`.

### Возвращаемая величина

Строка с оглавлением файла. Для того чтобы определить причину ошибки, если возвращена пустая строка, следует составить вызов, в котором используется **Функция `MIGetErrorMessage( )`**.

### Описание

Эту функцию следует использовать, чтобы получить оглавление файла в формате строки.

**См. также:**

**Функция `MIGetContent( )`, Функция `MIGetContentLen( )`**

---

## Функция `MIGetContentToFile( )`

### Назначение

Сохраняет оглавление в заданном файле.

### Синтаксис

```
MIGetContentToFile ( ByVal hFile As CHttpFile,  
                    ByVal strFileName As String As SmallInt
```

`hFile` – идентификатор `CHttpFile`.

`strFileName` – строка, определяющая имя локального файла, в котором будет сохранено оглавление `hFile`.

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется **Функция `MIGetErrorMessage( )`**.

### Описание

Эта функция используется, для того чтобы сохранить оглавление `CHttpFile` в локальном файле, в отличие от способа, когда используется `MIGetContent`. Будет создан новый файл с заданным именем. Если такой файл уже существует, он будет обрезан до нулевой длины. Следует учитывать, что функции `MIGetContentToFile` и `MIGetContent` взаимно исключают друг друга, можно использовать только одну из них за все время существования объекта `CHttpFile object`.

**См. также:**

**Функция `MIOpenRequest( )`, Функция `MISendRequest( )`, Функция `MIGetContent( )`.**



## Функция `MIGetContentType( )`

### Назначение

Получить тип оглавления файла.

### Синтаксис

```
MIGetContentType( ByVal hFile As CHttpFile, pBuffer As String,  
                  pBufferLength As Integer As SmallInt
```

`hFile` – идентификатор `CHttpFile`.

`pBuffer` – ссылка на строку, в которой будет помещен тип оглавления файла.

`pBufferLength` – ссылка на целую переменную, в которой содержится длина `pBuffer` в виде числа букв или байт. При успешном выполнении этой функции (строка будет записана в `pBuffer`), будет храниться длина строки в буквах минус 1, для ограничения символа `NULL`.

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется [Функция `MIGetErrorMessage\( \)`](#).

### Описание

Функцию `MIGetContentType( )` следует использовать для определения типа оглавления файла в виде отформатированной строки. Например, в `pBuffer` можно записать `"image/jpeg"` или `"text/html"`.

### См. также:

[Функция `MIQueryInfo\( \)`](#)

## Функция `MIGetCurrentFtpDirectory( )`

### Назначение

Получить имя текущего каталог на FTP-сервере по заданному идентификатору `CFtpConnection`.

### Синтаксис

```
MIGetCurrentFtpDirectory( ByVal hConnection As CFtpConnection,  
                          pDirName As String, pLen As Integer As SmallInt
```

`hConnection` – идентификатор `CFtpConnection`.

`pDirName` – ссылка на строку, в которой будет помещено имя каталога.

`pLen` – ссылка на переменную целого типа, в которой хранится размер буфера, определенного `pDirName` на входе; число букв хранящихся в `pDirName` на выходе.

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

### Описание

Параметрами pDirName являются полностью или частично определенные имена файлов, относительно текущего каталога. В качестве разделителя имен каталогов и файлов можно использовать как символ обратной косой черты (\), так и прямой косой черты (/).

MIGetCurrentFtpDirectory( ) – транслирует разделители имен каталогов в нужные символы.

### См. также:

[Функция MIGetFtpConnection\( \)](#), [Функция MISetCurrentFtpDirectory\( \)](#)

---

## Функция MIGetErrorCode( )

### Назначение

Получить последнее сообщение об ошибке, полученное в результате выполнения функции этой библиотеки.

### Синтаксис

```
MIGetErrorCode( ) As Integer
```

### Возвращаемая величина

Код последней ошибки.

### Описание

Функцию MIGetErrorCode( ) следует использовать, если при выполнении какой-либо функции произойдет ошибка, для выяснения кода ошибки. Для того чтобы получить сообщение об ошибке в виде строки, следует использовать вызов, в котором применяется [Функция MIGetErrorMessage\( \)](#).

Значение кода ошибки появляется только при выполнении других функций API HTTP и FTP. Эту функцию, в основном, удобно применять для выявления сообщений об ошибках, произошедших в результате выполнения операций, в которых использовалась [Функция MIsendRequest\( \)](#) или [Функция MIsendSimpleRequest\( \)](#), а нужное значение можно передать вызову, в котором используется [Функция MIErrorDlg\( \)](#).

### См. также:

[Функция MIGetErrorMessage\( \)](#), [Функция MIsendRequest\( \)](#), [Функция MIsendSimpleRequest\( \)](#), [Функция MIErrorDlg\( \)](#)

---

## Функция `MIGetErrorMessage( )`

### Назначение

Retrieves the last error message.

### Синтаксис

```
MIGetErrorMessage( ) As String
```

### Возвращаемая величина

Строка с сообщением об ошибке.

### Описание

Если возвращенное функцией значение сигнализирует об ошибке и требуется установить её причину, то для того чтобы получить полезную информацию, функцию `MIGetErrorMessage( )` следует использовать непосредственно после функции, которая вызвала ошибку. Многие сообщения об ошибках являются системными.

---

## Функция `MIGetFileURL( )`

### Назначение

Получить имя HTTP-файла в виде URL.

### Синтаксис

```
MIGetFileURL( ByVal hFile As CHttpFile, pURL As String,  
              ByVal lURLLen As Integer ) As SmallInt
```

*hFile* – идентификатор `CHttpFile`.

*pURL* – ссылка на строковую переменную, в которую будет передано имя HTTP-файла в виде URL.

*pURLLen* – размер *pURL* в виде числа букв или байт.

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется [Функция `MIGetErrorMessage\( \)`](#).

### Описание

Для того чтобы получить имя HTTP-файла в виде URL, используйте `MIGetFileURL( )`.

### См. также:

[Функция `MISendRequest\( \)`](#), [Функция `MIOpenRequest\( \)`](#)

## Функция `MIGetFtpConnection( )`

### Назначение

Установить FTP-соединение и получить идентификатор объекта `CFtpConnection`.

### Синтаксис

```
MIGetFtpConnection( ByVal hSession As CInternetSession,  
    ByVal strServer As String, ByVal strUserName As String,  
    ByVal strPassword As String, ByVal nPort As INTERNET_PORT )  
    As CFtpConnection
```

`hSession` – идентификатор `CInternetSession`.

`strServer` – строка с именем FTP-сервера.

`strUserName` – строка с именем пользователя, устанавливающего соединение.

`strPassword` – строка с паролем, который следует использовать при соединении.

`nPort` – число, определяющее номер порта TCP/IP на сервере.

### Возвращаемая величина

Идентификатор объекта `CFtpConnection`. Если запрос не будет выполнен, будет возвращено значение `Null`. Для того чтобы определить причину ошибки, обычно используется [Функция `MIGetErrorMessage\( \)`](#).

### Описание

`MIGetFtpConnection( )` устанавливает соединение с FTP-сервером, создает и возвращает идентификатор объекта `CFtpConnection`. Она не выполняет никаких операций на сервере. Например, если требуется получить или передать файлы на сервер, то придется выполнять эти операции отдельно.

При вызове требуется использовать идентификатор, полученный вызовом, в котором используется [Процедура `MICloseFtpConnection\( \)`](#), если такой идентификатор в момент использования не известен.

### См. также:

[Процедура `MICloseFtpConnection\( \)`](#), [Функция `MIGetHttpConnection\( \)`](#), [Функция `MICreateSession\( \)`](#), [Функция `MICreateSessionFull\( \)`](#), [Функция `MIParseURL\( \)`](#).

## Функция MlGetFtpFile( )

### Назначение

Получить файл с FTP-сервера по заданному идентификатору CFtpConnection и сохранить его локально.

### Синтаксис

```
MlGetFtpFile( ByVal hConnection As CFtpConnection,
              ByVal strRemoteFile As String, ByVal strLocalFile As String,
              ByVal bFailIfExists As SmallInt, ByVal dwAttributes As Integer,
              ByVal dwFlags As Integer ) As SmallInt
```

- hConnection – идентификатор CFtpConnection.
- strRemoteFile – строка с именем файла, который требуется получить с FTP-сервера.
- strLocalFile – строка с именем, под которым полученный файл требуется сохранить на локальной системе.
- bFailIfExists – указывает на то, что файл с таким именем уже существует. Если локальный файл с таким именем уже существует, а для этого параметра установлено значение TRUE, функция MlGetFtpFile( ) не будет выполнена. Иначе, MlGetFtpFile( ) сотрет существующую копию файла.
- dwAttributes – определяет атрибуты файла. Возможна любая из ниже перечисленных комбинация флагов FILE\_ATTRIBUTE\_\* .

| dwAttribute value         | Определение  |
|---------------------------|--|
| FILE_ATTRIBUTE_ARCHIVE    | Для файла задан атрибут архивирования. Некоторые приложения используют этот атрибут, для того чтобы отметить файл для архивирования или удаления.                                |
| FILE_ATTRIBUTE_COMPRESSED | Файл или каталог сжаты. Сжатый файл, в котором все данные сжаты. Сжатый каталог, в котором все новые файлы и подкаталоги будут сжаты.  |
| FILE_ATTRIBUTE_DIRECTORY  | Файл является каталогом.   |
| FILE_ATTRIBUTE_NORMAL     | У файла нет дополнительных атрибутов. Этот атрибут может использоваться только индивидуально. Если установить любой другой атрибут, то атрибут FILE_ATTRIBUTE_NORMAL будет снят: |
| FILE_ATTRIBUTE_HIDDEN     | Файл является скрытым. Такой файл не будет включен в стандартный список каталога.  |
| FILE_ATTRIBUTE_READONLY   | Файл защищен от тзаписи. Приложения могут читать файл, но не могут записывать в него или удалять его.  |

| dwAttribute value        | Определение  |
|--------------------------|--|
| FILE_ATTRIBUTE_SYSTEM    | Файл является частью или используется исключительно операционной системой.   |
| FILE_ATTRIBUTE_TEMPORARY | Файл используется временно. Приложения могут записывать в файл только в случае экстренной необходимости. Основная часть данных в таком файле остается в памяти и не переносится на носитель, поскольку файл будет скоро уничтожен. |

dwFlags – определяют условия, при которых происходит перенос данных. Этот параметр может принимать любые из нижеперечисленных значений:

| значения dwFlags          | Определение  |
|---------------------------|--|
| FTP_TRANSFER_TYPE_ASCII   | Перенос файлов будет осуществляться по методу FTP ASCII (Type A). Информация о настройках и форматировании файла будет заменена локальными эквивалентами.              |
| FTP_TRANSFER_TYPE_BINARY  | Перенос файлов будет осуществляться бинарным методом FTP (Type I). Файлы будут точно скопированы, без всяких изменений. Этот метод передачи используется по умолчанию. |
| FTP_TRANSFER_TYPE_UNKNOWN | Будет использоваться вариант FTP_TRANSFER_TYPE_BINARY.   |

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

### Описание

Оба параметра strRemoteFile и strLocalFile могут быть либо именами определенными относительно текущего каталога, либо быть полностью определенным именем, включая имена каталогов. В качестве разделителя имен каталогов и файлов можно использовать как символ обратной косой черты (\), так и прямой косой черты (/).

См. также:

[Функция MIGetFtpConnection\( \)](#), [Функция MIPutFtpFile\( \)](#)

## Функция **MIGetFtpFileFind( )**

### Назначение

Получает идентификатор объекта CFtpFileFind.

### Синтаксис

```
MIGetFtpFileFind( ByVal hConnection As CFtpConnection ) As CFtpFileFind
```

*hConnection* – идентификатор CFtpConnection.

### Возвращаемая величина

Идентификатор объекта CFtpFileFind. Если запрос не будет выполнен, будет возвращено значение Null. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

### Описание

Класс CFtpFileFind включает поиск файлов в интернете по FTP-серверам.

При вызове требуется использовать идентификатор, полученный вызовом, в котором используется [Процедура MICloseFtpFileFind\( \)](#), если такой идентификатор в момент использования не известен.

### См. также:

[Функция MIGetFtpConnection\( \)](#), [Процедура MICloseFtpFileFind\( \)](#)

## Процедура **MIGetFtpFileName( )**

### Назначение

Получить имя найденного файла, идентификатор CFtpFileFind handle которого задан.

### Синтаксис

```
MIGetFtpFileName( ByVal hFTPFind As CFtpFileFind, pFileName As String,  
                  ByVal bufferlen As Integer )
```

*hFTPFind* – идентификатор CFtpFileFind.

*pFileName* – ссылка на строковую переменную, в которой будет храниться имя найденного файла.

*bufferlen* is the size of the buffer referenced by *pFileName*.

### Описание

Прежде, чем применять функцию MIGetFtpFileName( ), требуется организовать, хотя бы один раз, вызов, в котором используется [Функция MIFindNextFtpFile\( \)](#).

См. также:

Функция `MIGetFtpFileFind( )`, Функция `MIFindNextFtpFile( )`, Функция `MIFindFtpFile( )`

---

## Функция `MIGetHttpConnection( )`

### Назначение

Установить HTTP-соединение и получить идентификатор объекта `CHttpConnection`.

### Синтаксис

```
MIGetHttpConnection( ByVal hSession As CInternetSession,  
    ByVal strServer As String, ByVal nPort As INTERNET_PORT  
    As CHttpConnection
```

`hSession` – идентификатор `CInternetSession`.

`strServer` – строка с именем HTTP-сервера.

`nPort` – число, определяющее номер порта TCP/IP на сервере.

### Возвращаемая величина

Идентификатор объекта `CHttpConnection`. Если запрос не будет выполнен, будет возвращено значение `Null`. Для того чтобы определить причину ошибки, обычно используется [Функция `MIGetErrorMessage\( \)`](#).

### Описание

`MIGetHttpConnection( )` устанавливает соединение с HTTP-сервером, создает и возвращает идентификатор объекта `CHttpConnection`. Она не выполняет никаких операций на сервере. Например, если требуется получить идентификатор HTTP, то придется выполнять эти операции отдельно.

При вызове требуется использовать идентификатор, полученный вызовом, в котором используется [Процедура `MICloseHttpConnection\( \)`](#), если такой идентификатор в момент использования не известен.

См. также:

[Процедура `MICloseHttpConnection\( \)`](#), [Функция `MIGetFtpConnection\( \)`](#), [Функция `MICreateSession\( \)`](#), [Функция `MICreateSessionFull\( \)`](#), [Функция `MIParseURL\( \)`](#)

---

## Функция `MllsFtpDirectory( )`

### Назначение

Определить, является ли найденный файл каталогом заданным идентификатором `CFtpFileFind`.



**Синтаксис**

```
MIIsFtpDirectory( ByVal hFTPFind As CFtpFileFind ) As SmallInt
```

*hFTPFind* – идентификатор CFtpFileFind.

**Возвращаемая величина**

Ненулевое значение, если найденный файл является каталогом; иначе 0.

**Описание**

Требуется по крайней мере единственный вызов, в котором используется функция **Функция MIFindNextFtpFile( )**, выполненный прежде, чем вызывать функцию MIIsFtpDirectory( ).

**См. также:**

**Функция MIGetFtpFileFind( )**, **Функция MIFindNextFtpFile( )**, **Функция MIFindFtpFile( )**,  
**Процедура MIGetFtpFileName( )**

---

**Функция MIIsFtpDots( )****Назначение**

Проверить маркеры текущего и родительского каталога во время итерации по файлам с заданным идентификатором CFtpFileFind.

**Синтаксис**

```
MIIsFtpDots( ByVal hFTPFind As CFtpFileFind ) As SmallInt
```

*hFTPFind* – идентификатор CFtpFileFind.

**Возвращаемая величина**

Ненулевое значение, если имя файла “.” или “..”, что говорит о том, что найденный файл это каталог. Иначе 0.

**Описание**

Требуется по крайней мере единственный вызов, в котором используется функция **Функция MIFindNextFtpFile( )**, выполненный прежде, чем вызывать функцию MIIsFtpDots( ).

**См. также:**

**Функция MIGetFtpFileFind( )**, **Функция MIFindNextFtpFile( )**, **Функция MIFindFtpFile( )**,  
**Процедура MIGetFtpFileName( )**

## Функция `MIOpenRequest( )`

### Назначение

Установить HTTP-соединение.

### Синтаксис

```
MIOpenRequest( ByVal hConnection As CHttpConnection,  
               ByVal nVerb As Integer, ByVal strObjectName As String  
               ) As CHttpFile
```

*hConnection* – идентификатор `CHttpConnection`.

*nVerb* – число, ассоциированное с типом HTTP-запроса. Может быть одним из нижеперечисленных:

- `HTTP_VERB_POST`
- `HTTP_VERB_GET`
- `HTTP_VERB_HEAD`
- `HTTP_VERB_PUT`
- `HTTP_VERB_LINK`
- `HTTP_VERB_DELETE`
- `HTTP_VERB_UNLINK`

*strObjectName* – строка с объектом, над которым требуется выполнить выбранную команду. Обычно это имя файла, исполняемый модуль или описание объектов поиска.

### Возвращаемая величина

Идентификатор запрашиваемого объекта `CHttpFile`. Если запрос не будет выполнен, будет возвращено значение `Null`. Для того чтобы определить причину ошибки, обычно используется

[Функция `MIGetErrorMessage\( \)`](#).

### Описание

Эта функция устанавливает HTTP-соединение и возвращает идентификатор объекта `CHttpFile`, с помощью которого обеспечиваются запросы к службам и чтение файлов на HTTP-сервере. Если во время интернет-сеанса требуется прочесть данные с HTTP-сервера, необходимо получить идентификатор объекта `CHttpFile`.

При вызове требуется использовать идентификатор, полученный вызовом, в котором используется [Процедура `MICloseHttpFile\( \)`](#), если такой идентификатор в момент использования не известен.

### См. также:

[Функция `MIGetHttpConnection\( \)`](#), [Функция `MIOpenRequestFull\( \)`](#), [Функция `MIParseURL\( \)`](#)

## Функция MIOpenRequestFull( )

### Назначение

Установить HTTP-соединение.

### Синтаксис

```
MIOpenRequestFull( ByVal hConnection As CHttpConnection,  
    ByVal nVerb As Integer, ByVal strObjectName As String,  
    ByVal strReferer As String, ByVal dwContext As Integer,  
    ByVal strVersion As String, ByVal dwFlags As Integer )  
    As CHttpFile
```

*hConnection* – идентификатор CHttpConnection.

*nVerb* – число, ассоциированное с типом HTTP-запроса. Может быть одним из нижеперечисленных:

- HTTP\_VERB\_POST
- HTTP\_VERB\_GET
- HTTP\_VERB\_HEAD
- HTTP\_VERB\_PUT
- HTTP\_VERB\_LINK
- HTTP\_VERB\_DELETE
- HTTP\_VERB\_UNLINK

*strObjectName* – строка с объектом, над которым требуется выполнить выбранную команду. Обычно это имя файла, исполняемый модуль или описание объектов поиска.

*strReferer* – строка с адресом (URL) документа, от которого был получен URL в запросе (*strObjectName*). Если строка пустая, – никакого заголовка HTTP не задано.

*dwContext* – идентификатор контекста операции MIOpenRequestFull( ). За подробной информацией, обращайтесь к Microsoft MSDN library.

*strVersion* – строка, определяющая номер версии HTTP. Если строка пустая, будет использован вариант “HTTP/1.0”.

*dwFlags* – любая комбинация следующих флагов INTERNET\_FLAG\_\*:

| значение dwFlag          | Определение  |
|--------------------------|--|
| INTERNET_FLAG_RELOAD     | Принудительно загружает выбранный файл, объект или список файлов каталога с сервера, а не из кэша. |
| INTERNET_FLAG_DONT_CACHE | Не помещает полученное в кэш.  |

| значение dwFlag                | Определение   |
|--------------------------------|---|
| INTERNET_FLAG_MAKE_PERSISTENT  | Добавляет полученное в кэш для постоянного хранения. Это значит, что при стандартной очистке кэша, проверке на совместимость или сборе мусора, этот объект не будет удален из кэша. |
| INTERNET_FLAG_SECURE           | При транзакции будет использована защищенная семантика. При этом все сообщения транслируются SSL/PCT, что имеет значение только при HTTP-запросах.                                  |
| INTERNET_FLAG_NO_AUTO_REDIRECT | Используется только с HTTP, указывает, что функция <b>Функция MISendRequest( )</b> не должна автоматически применяться для переадресации.   |

**Возвращаемая величина**

Идентификатор запрашиваемого объекта CHttpFile. Если запрос не будет выполнен, будет возвращено значение Null. Для того чтобы определить причину ошибки, обычно используется **Функция MIMessage( )**.

**Описание**

Эта функция устанавливает HTTP-соединение и возвращает идентификатор объекта CHttpFile, с помощью которого обеспечиваются запросы к службам и чтение файлов на HTTP-сервере. Если во время интернет-сеанса требуется прочитать данные с HTTP-сервера, необходимо получить идентификатор объекта CHttpFile. Эта функция-надстройка – оболочка над MFC функцией OpenRequest с дополнительными параметрами, определяющими допустимые типы. В этой версии, функция-надстройка всегда присваивает этому параметру значение NULL.

При вызове требуется использовать идентификатор, полученный вызовом, в котором используется **Процедура MICloseHttpFile( )**, если такой идентификатор в момент использования не известен.

**См. также:**

**Функция MIGetHttpConnection( )**, **Функция MIOpenRequest( )**, **Функция MIParseURL( )**

---

**Функция MIParseURL( )****Назначение**

Разбирает строку с URL и возвращает тип и компоненты службы.

## Синтаксис

```
MIParseURL( ByVal strURL As String, pServiceType As Integer,
             pServer As String, ByVal nServerLen As Integer,
             pObject As String, ByVal nObjectLen As Integer,
             pPort As INTERNET_PORT )
             As SmallInt
```

*strURL* – строка с URL, который требуется разобрать.

*pServiceType* – ссылка на целую переменную, которой будет передан тип интернет-службы.

Допустимы следующие значения:

- INTERNET\_SERVICE\_FTP
- INTERNET\_SERVICE\_GOPHER
- INTERNET\_SERVICE\_HTTP
- AFX\_INET\_SERVICE\_UNK
- AFX\_INET\_SERVICE\_FILE
- AFX\_INET\_SERVICE\_MAILTO
- AFX\_INET\_SERVICE\_MID
- AFX\_INET\_SERVICE\_CID
- AFX\_INET\_SERVICE\_NEWS
- AFX\_INET\_SERVICE\_NNTP
- AFX\_INET\_SERVICE\_PROSPERO
- AFX\_INET\_SERVICE\_TELNET
- AFX\_INET\_SERVICE\_WAIS
- AFX\_INET\_SERVICE\_AFS
- AFX\_INET\_SERVICE\_HTTPS

*strsServer* – ссылка на строковую переменную, в которой указан первый после типа службы сегмент URL.

*nServerLen* – размер буфера, на который ссылается *strsServer*.

*pObject* – ссылка на объект, к которому обращается URL (может быть пустой).

*nObjectLen* – размер буфера, на который ссылается *pObject*.

*pPort* – ссылка на целую переменную с заданным номером порта из любого из разделов URL: либо сервера, либо объекта, если такой существует. Номер порта используется для идентификации порта TCP/IP, используемого на сервере.

## Возвращаемая величина

Ненулевое значение, если URL был успешно разобран; иначе, 0, если он пустой или не содержит знакомый тип интернет-службы. Для того чтобы определить причину ошибки, обычно используется **Функция MIGetErrorMessage( )**.

Описание

Функция MIParseURL( ) разбирает строку с URL и возвращает тип и компоненты службы. Например, при разборе URL, заданного в форме: ftp://ftp.mysite.org/, отдельные компоненты будут храниться следующим образом:

```
pServer == "ftp.mysite.org"
pObject == "/"
nPort == #port
pServiceType == INTERNET_SERVICE_FTP
```

Функция MIPutFtpFile( )

Назначение

Сохранить файл с заданным идентификатором CFtpConnection на FTP-сервере.

Синтаксис

```
MIPutFtpFile( ByVal hConnection As CFtpConnection,
    ByVal strLocalFile As String, ByVal strRemoteFile As String,
    ByVal dwFlags As Integer )
    As SmallInt
```

hConnection – идентификатор CFtpConnection.

strLocalFile – строка с локальным именем файла.

strRemoteFile – строка с именем, под которым полученный файл требуется сохранить на FTP-сервере.

dwFlags – определяют условия, при которых происходит пернос данных. Этот параметр может принимать любые из нижеперечисленных значений:

| значение dwFlag          | Определение   |
|--------------------------|---|
| FTP_TRANSFER_TYPE_ASCII  | Передача файла будет осуществляться ASCII методом FTP (Type A). Информация о настройках и форматировании будет заменена локальными эквивалентами.                       |
| FTP_TRANSFER_TYPE_BINARY | Передача файлов будет осуществляться бинарным методом FTP (Type I). Файлы будут точно скопированы, без всяких изменений. Этот метод передачи используется по умолчанию. |

Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется [Функция MIPGetErrorMessage\( \)](#).

**Описание**

Оба параметра `strRemoteFile` и `strLocalFile` могут быть либо именами определенными относительно текущего каталога, либо быть полностью определенным именем, включая имена каталогов. В качестве разделителя имен каталогов и файлов можно использовать как символ обратной косой черты (`\`), так и прямой косой черты (`/`).

См. также:

**Функция `MIGetFtpConnection( )`, Функция `MIGetFtpFile( )`**

---

**Функция `MIQueryInfo( )`****Назначение**

Returns response or request headers from an HTTP request.

**Синтаксис**

```
MIQueryInfo( ByVal hFile As CHttpFile, ByVal dwInfoLevel As Integer,
              pBuffer As String, pBufferLength As Integer ) As SmallInt
```

`hFile` – идентификатор `CHttpFile`.

`dwInfoLevel` – комбинация атрибутов запроса и флага модификатора, с помощью которого задается тип запрашиваемой информации: список модификаторов приводится в библиотеке Microsoft MSDN.

`pBuffer` – ссылка на строку, в которой будет помещена информация. Для атрибута `HTTP_QUERY_CUSTOM`, в `pBuffer` указывается, какое имя заголовка будет участвовать в запросе.

`pBufferLength` – ссылка на целую переменную, в которой содержится длина `pBuffer` в виде числа букв или байт. При успешном выполнении этой функции (строка будет записана в `pBuffer`), будет храниться длина строки в буквах минус 1, для ограничения символа `NULL`.

**Возвращаемая величина**

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется **Функция `MIGetErrorMessage( )`**.

**Описание**

Эту функцию следует использовать для получения отклика или заголовка из HTTP-запроса. Описание значений атрибутов содержится в библиотеке Microsoft MSDN, а информация о флагах запросов по этому адресу ([http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wininet/wininet/query\\_info\\_flags.asp](http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wininet/wininet/query_info_flags.asp)).

См. также:

**Функция `MIOpenRequest( )`, Функция `MISendRequest( )`**

## Функция MIQueryInfoStatusCode( )

### Назначение

Получить код состояния, связанного с HTTP-запросом.

### Синтаксис

```
MIQueryInfoStatusCode( ByVal hFile As CHttpFile, pStatusCode As Integer  
s SmallInt
```

hFile – идентификатор CHttpFile.

pStatusCode – ссылка на целую переменную для хранения кода состояния. Код состояния сигнализирует об успешном или неудачном выполнении заданного события. Коды состояния HTTP разделены на группы, по успешному или неудачному выполнению запроса. В следующей таблице выделены группы кодов и часто встречающиеся отдельные коды состояния HTTP.

**Группы кодов состояния HTTP**

| Группа  | Значение          |
|---------|-------------------|
| 200-299 | Успешно           |
| 300-399 | Информация        |
| 400-499 | Ошибка запроса    |
| 500-599 | Ошибка на сервере |

**Часто встречающиеся коды состояния HTTP**

| Код состояния | Значение                                  |
|---------------|---|
| 200           | URL найден, передача будет продолжена.    |
| 400           | Непонятный запрос.                        |
| 404           | Запрошенный URL не найден.                |
| 405           | Сервер не поддерживает запрошенный метод. |
| 500           | Неизвестная ошибка на сервере.            |
| 503           | Достигнута емкость сервера.               |



**Возвращаемая величина**

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется **Функция `MIGetErrorMessage( )`**.

**Описание**

Эту функцию следует использовать для получения кода состояния, связанного с HTTP-запросом. За подробной информацией, обращайтесь к Microsoft MSDN library.

См. также:

**Функция `MIOpenRequest( )`**, **Функция `MISendRequest( )`**

---

**Функция `MISaveContent( )`****Назначение**

Сохраняет оглавление в заданном файле.

**Синтаксис**

```
MISaveContent ( ByVal hContent As CString, ByVal strFileName As String  
                As SmallInt
```

*hContent* – идентификатор CString.

*strFileName* – строка, определяющая имя файла, в котором будет сохранено оглавление.

**Возвращаемая величина**

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется **Функция `MIGetErrorMessage( )`**.

**Описание**

С помощью этой функции можно сохранить содержимое в файл. Будет создан новый файл с заданным именем. Если такой файл уже существует, он будет обрезан до нулевой длины.

См. также:

**Функция `MIGetContent( )`**

---

**Функция `MISendRequest( )`****Назначение**

Передать запрос на HTTP-сервер.

**Синтаксис**

```
MISendRequest ( ByVal hFile As CHttpFile, ByVal strHeaders As String,  
                ByVal dwHeadersLen As Integer, ByVal strOptional As String,
```

## Функция **MI**SendSimpleRequest( )

---

```
ByVal dwOptionalLen As Integer, ByVal bAuthenticate As SmallInt  
As SmallInt
```

hFile – идентификатор CHttpFile.

strHeaders – строка с именами заголовков, которые требуется передать.

dwHeadersLen – длина заголовков, определенных в strHeaders.

strOptional – дополнительные данные, которые требуется передать сразу после заголовков запроса. Обычно используются операции POST и PUT. Может быть пустым, если дополнительной информации передавать не требуется.

dwOptionalLen – длина strOptional.

bAuthenticate – определяет, следует ли проверять идентификацию или нет.

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется **Функция MI**GetErrorMessage( ).

### Описание

Эта функция запрашивает HTTP-сервер.

**См. также:**

**Функция MI**OpenRequest( ), **Функция MI**OpenRequestFull( )

---

## Функция **MI**SendSimpleRequest( )

### Назначение

Передать запрос на HTTP-сервер.

### Синтаксис

```
MISendSimpleRequest( ByVal hFile As CHttpFile,  
ByVal bAuthenticate As SmallInt ) As SmallInt
```

hFile – идентификатор CHttpFile.

bAuthenticate – определяет, следует ли проверять идентификацию или нет.

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется **Функция MI**GetErrorMessage( ).

### Описание

Эта функция запрашивает HTTP-сервер.

См. также:

Функция [MIOpenRequest\( \)](#), Функция [MIOpenRequestFull\( \)](#), Функция [MISendRequest\( \)](#)

---

## Функция MISetCurrentFtpDirectory( )

### Назначение

Сменить каталог на FTP-сервере с заданным идентификатором CFtpConnection.

### Синтаксис

```
MISetCurrentFtpDirectory( Byval hConnection As CFtpConnection,  
    Byval strDirName As String ) As SmallInt
```

*hConnection* – идентификатор CFtpConnection.

*strDirName* – строка с именем каталога.

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется [Функция MlGetErrorMessage\( \)](#).

### Описание

Параметр *pDirName* может быть полностью или частично определенным именем файла, относительно текущего каталога. В качестве разделителя имен каталогов и файлов можно использовать как символ обратной косой черты (\), так и прямой косой черты (/).

MISetCurrentFtpDirectory( ) – транслирует разделители имен каталогов в нужные символы.

См. также:

Функция [MlGetFtpConnection\( \)](#), Функция [MlGetCurrentFtpDirectory\( \)](#)

---

## Функция MISetSessionTimeout( )

### Назначение

Задать интервала ожидания для интернет-сеанса.

### Синтаксис

```
MISetSessionTimeout( ByVal hSession As CInternetSession,  
    ByVal Connect As Integer, ByVal Send As Integer,  
    ByVal Receive As Integer ) As SmallInt
```

*hSession* – идентификатор объекта CInternetSession.

*Connect* – целая переменная с значением интервала ожидания, которое будет использоваться при запросе на интернет-соединение, в миллисекундах.

`Send` – целая переменная с интервалом ожидания, который будет использоваться при передаче запроса, в миллисекундах.

`Receive` – целая переменная с интервалом ожидания, который будет использоваться для отклика на запрос, в миллисекундах.

### Возвращаемая величина

Ненулевое значение, если поиск успешно выполнен, иначе 0. Для того чтобы определить причину ошибки, обычно используется [Функция `MIGetErrorMessage\( \)`](#).

### Описание

Эту функцию следует использовать, для того чтобы задать интервал ожидания для интернет-сеанса. По умолчанию, для каждого из параметров (`Connect`, `Send`, `Receive`) используется 0. За подробной информацией, обращайтесь к Microsoft MSDN library.

### См. также:

[Функция `MICreateSession\( \)`](#), [Функция `MICreateSessionFull\( \)`](#)

# Библиотека XML

В этом приложении подробно описаны библиотека документов XML, с помощью которых программисты на языке MapBasic могут создавать и разбирать документы XML и использовать другие веб-технологии. Эти библиотеки используют общие DEF-файлы: XMLLib.DEF and XMLTypes.DEF, которые находятся по адресу: <Каталог, в котором установлен MapBasic>\Samples\MapBasic\INC. Эти файлы следует включить в состав секции заголовков программ. Все функции, описанные в этом приложении, зависят от наличия файла GmlXlat.dll, который устанавливается вместе с MapInfo Professional.

Все функции и процедуры, описанные в этом приложении, являются оболочками соответствующих методов интерфейсов и классов Microsoft XML. Набор интерфейсных классов включает в себя: IXMLDOMNode, IXMLDOMNodeList, IXMLDOMNamedNodeMap, IXMLDOMSchemaCollection2 и IXMLDOMDocument2. Более подробная информация об использовании зависимых классов и интерфейсов доступна по ссылке MSDN

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xmlsdk/html/39b17b9c-04c7-4fa8-bcee-1f7d57eefd74.asp>

**Внимание:** Поскольку используется внешняя библиотека, все функции и процедуры, перечисленные в этом приложении, не могут быть использованы в окне MapBasic MapInfo Professional.

## Процедура MIXmlAttributeListDestroy( )

### Назначение

Удаляет объект MIXmlNamedNodeMap и освобождает память.

### Синтаксис

**MIXmlNodeListDestroy**( ByVal *hXMLNodeList* As MIXmlNodeList )

*hXMLAttributeList* идентификатор MIXmlNamedNodeMap удаляемого объекта.

### Описание

При вызове требуется вызвать эту функцию, для того чтобы освободить идентификатор MIXmlNamedNodeMap, для получения которого используется [Функция MIXmlGetAttributeList\( \)](#), если такой идентификатор больше не используется.

См. также:

[Функция MIXmlGetAttributeList\( \)](#)

---

## Функция MIXmlDocumentCreate( )

### Назначение

Создает объект MIXmlDocument и получает идентификатор этого объекта.

### Синтаксис

**MIXmlDocumentCreate**( ) As MIXmlDocument

### Возвращаемая величина

Идентификатор объекта MIXmlDocument. Если запрос не будет выполнен, будет возвращено значение Null. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

### Описание

MIXmlDocumentCreate( ) – создает и возвращает идентификатор объекта MIXmlDocument. Является самым старшим уровнем исходного кода XML.

При вызове требуется добавить идентификатор, для получения которого используется [Процедура MIXmlDocumentDestroy\( \)](#), если такой идентификатор больше не применяется.

См. также:

[Процедура MIXmlDocumentDestroy\( \)](#)

---

## Процедура MIXmlDocumentDestroy( )

### Назначение

Удаляет объект MIXmlDocument и освобождает память.

### Синтаксис

```
MIXmlDocumentDestroy( ByVal hXMLDocument As MIXmlDocument )
```

*hXMLDocument* идентификатор MIXmlDocument удаляемого объекта.

### Описание

При вызове требуется вызвать эту функцию, для того чтобы закрыть и освободить идентификатор MIXmlDocument, для получения которого используется **Функция MIXmlDocumentCreate( )**, если такой идентификатор больше не используется.

См. также:

**Функция MIXmlDocumentCreate( )**

---

## Функция MIXmlDocumentGetNamespaces( )

### Назначение

Создает объект MIXMLSchemaCollection и получает идентификатор этого объекта.

### Синтаксис

```
MIXmlDocumentGetNamespaces( ByVal hXMLDocument As MIXmlDocument )  
As MIXMLSchemaCollection
```

*hXMLDocument* – идентификатор объекта MIXmlDocument.

### Возвращаемая величина

Идентификатор объекта MIXMLSchemaCollection при успешном выполнении, иначе NULL.

### Описание

Этот метод создает объект MIXMLSchemaCollection.

При вызове требуется добавить идентификатор, для получения которого используется **Процедура MIXmlISCDestroy( )**, если такой идентификатор больше не применяется.

См. также:

**Функция MIXmlDocumentCreate( )**, **Процедура MIXmlISCDestroy( )**

## Функция MIXmlDocumentGetRootNode( )

### Назначение

Будет получен самый старший элемент документа.

### Синтаксис

```
MIXmlDocumentGetRootNode( ByVal hXMLDocument As MIXmlDocument )  
    As MIXmlNode
```

*hXMLDocument* – идентификатор объекта MIXmlDocument.

### Возвращаемая величина

Идентификатор объекта MIXmlNode, описывающий самый старший элемент документа, при успешном выполнении, иначе NULL.

### Описание

MIXmlDocumentGetRootNode( ) возвращает идентификатор объекта MIXmlNode, описывающий самый старший элемент дерева документа XML. Будет возвращено значение NULL, если иерархии не существует.

При вызове требуется добавить идентификатор, для получения которого используется [Процедура MIXmlNodeDestroy\( \)](#), если такой идентификатор больше не применяется.

### См. также:

[Функция MIXmlDocumentCreate\( \)](#), [Процедура MIXmlNodeDestroy\( \)](#)

---

## Функция MIXmlDocumentLoad( )

### Назначение

Загружает XML-документ из указанного места.

### Синтаксис

```
MIXmlDocumentLoad( ByVal hXMLDocument As MIXmlDocument,  
    ByVal strPath As String, pbParsingError As SmallInt,  
    ByVal bValidate As SmallInt, ByVal bResolveExternals As SmallInt )  
    As SmallInt
```

*hXMLDocument* – идентификатор объекта MIXmlDocument.

*strPath* – строка, содержащая маршрут/URL к XML-файлу.

*pbParsingError* – указывает на значение типа SmallInt, которое означает TRUE при успешной загрузке; FALSE если загрузка не выполнена.



bValidate – значение типа SmallInt, которое определяет требуется ли проверять синтаксис документ анализатором. Если TRUE (1), документ будет проверен анализатором. Если FALSE (0), будет проанализирован только правильно составленный XML-код.

bResolveExternals – короткая целая переменная (SmallInt), с помощью которой можно задать или отменить разрешение при разборе, независимо от проверки, внешних определителей, допустимости пространства имен, внешних подмножеств определителей типов документа (DTD) и ссылок на внешние источники. Если параметр bResolveExternals принимает значение TRUE (1), внешние определители будут исследоваться при разборе. При этом стандартные атрибуты и типы данных можно будет определять по элементам схемы, а для включения файлов использовать DTD. Этот параметр не зависит от выполнения проверки, о чем сигнализирует значение свойства bValidate. Если внешние параметры не могут быть разрешены при проверке, появится сообщение об ошибке при проверке. Если параметр bResolveExternals принимает значение FALSE (0), внешние параметры не будут исследованы, а проверка не будет выполняться.

### Возвращаемая величина

Ненулевое значение при успешном выполнении, иначе – 0. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

### Описание

Если URL не может быть определен, к нему нет доступа или он не ссылается на XML-документ, этот метод возвращает FALSE. Вызов функции MIXmlDocumentLoad( ) по отношению к существующему документу, моментально освобождает оглавление документа. При загрузке XML-документа из ресурса, загрузка должна выполняться асинхронно, иначе загрузка не будет успешной.

См. также:

[Функция MIXmlDocumentCreate\( \)](#), [Функция MIXmlDocumentLoadXML\( \)](#) [Функция MIXmlDocumentLoadXMLString\( \)](#)

---

## Функция MIXmlDocumentLoadXML( )

### Назначение

Загрузить XML-документ, используя заданную строку.

### Синтаксис

```
MIXmlDocumentLoadXML( ByVal hXMLDocument As MIXmlDocument,
    ByVal hContent As CString, pbParsingError As SmallInt,
    ByVal bValidate As SmallInt, ByVal bResolveExternals As SmallInt )
    As SmallInt
```

hXMLDocument – идентификатор объекта MIXmlDocument.

hContent – идентификатор строковой переменной CString со XML-строкой, которую необходимо загрузить в объект XML-документа. Эта строка может содержать либо весь XML-документ, либо правильно сформатированный фрагмент.

## Функция MIXmlDocumentLoadXMLString( )

---

pbParsingError – указывает на значение типа SmallInt, которое означает TRUE (любое ненулевое значение) при успешной загрузке; FALSE (0) если загрузка не выполнена.

bValidate – значение типа SmallInt, которое определяет требуется ли проверять синтаксис документ анализатором. Если TRUE (1), документ будет проверен анализатором. Если FALSE (0), будет проанализирован только правильно составленный XML-код.

bResolveExternals – короткая целая переменная (SmallInt), с помощью которой можно задать или отменить разрешение при разборе, независимо от проверки, внешних определителей, допустимости пространства имен, внешних подмножеств определителей типов документа (DTD) и ссылок на внешние источники. Если параметр bResolveExternals принимает значение TRUE (1), внешние определители будут исследоваться при разборе. При этом стандартные атрибуты и типы данных можно будет определять по элементам схемы, а для включения файлов использовать DTD. Этот параметр не зависит от выполнения проверки, о чем сигнализирует значение свойства bValidate. Если внешние параметры не могут быть разрешены при проверке, появится сообщение об ошибке при проверке. Если параметр bResolveExternals принимает значение FALSE (0), внешние параметры не будут исследованы, а проверка не будет выполняться.

### Возвращаемая величина

Ненулевое значение при успешном выполнении, иначе – 0. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

### Описание

Вызов функции MIXmlDocumentLoadXML( ) по отношению к существующему документу, моментально освобождает оглавление документа. Выполняется только в кодировках UTF-16 или UCS-2.

### См. также:

[Функция MIXmlDocumentCreate\( \)](#) [Функция MIXmlDocumentLoad\( \)](#) [Функция MIXmlDocumentLoadXMLString\( \)](#)

---

## Функция MIXmlDocumentLoadXMLString( )

### Назначение

Загрузить XML-документ, используя заданную строку.

### Синтаксис

```
MIXmlDocumentLoadXMLString( ByVal hXMLDocument As MIXmlDocument,  
    ByVal strXML As String, pbParsingError As SmallInt,  
    ByVal bValidate As SmallInt, ByVal bResolveExternals As SmallInt )  
As SmallInt
```

hXMLDocument – идентификатор объекта MIXmlDocument.

`strXML` – строковая переменная с XML-строкой, которую требуется загрузить в этот объект XML-документа. Эта строка может содержать либо весь XML-документ, либо правильно сформатированный фрагмент.

`pbParsingError` – указывает на значение типа `SmallInt`, которое означает `TRUE` (любое ненулевое значение) при успешной загрузке; `FALSE` (0) если загрузка не выполнена.

`bValidate` – значение типа `SmallInt`, которое определяет требуется-ли проверять синтаксис документ анализатором. Если `TRUE` (1), документ будет проверен анализатором. Если `FALSE` (0), будет проанализирован только правильно составленный XML-код.

`bResolveExternals` – короткая целая переменная (`SmallInt`), с помощью которой можно задать или отменить разрешение при разборе, независимо от проверки, внешних определителей, допустимости пространства имен, внешних подмножеств определителей типов документа (DTD) и ссылок на внешние источники. Если параметр `bResolveExternals` принимает значение `TRUE` (1), внешние определители будут исследоваться при разборе. При этом стандартные атрибуты и типы данных можно будет определять по элементам схемы, а для включения файлов использовать DTD. Этот параметр не зависит от выполнения проверки, о чем сигнализирует значение свойства `bValidate`. Если внешние параметры не могут быть разрешены при проверке, появится сообщение об ошибке при проверке. Если параметр `bResolveExternals` принимает значение `FALSE` (0), внешние параметры не будут исследованы, а проверка не будет выполняться.

### Возвращаемая величина

Ненулевое значение при успешном выполнении, иначе – 0. Для того чтобы определить причину ошибки, обычно используется [Функция `MIGetErrorMessage`](#) ( ).

### Описание

Вызов функции `MIXmlDocumentLoadXMLString`( ) по отношению, к существующему документу, моментально освобождает оглавление документа. Выполняется только в кодировках UTF-16 или UCS-2.

См. также:

[Функция `MIXmlDocumentCreate`](#)( ), [Функция `MIXmlDocumentLoad`](#)( ), [Функция `MIXmlDocumentLoadXML`](#)( ).

---

## Функция `MIXmlDocumentSetProperty`( )

### Назначение

Присвоить свойства объекту `MIXmlDocument`.

### Синтаксис

```
MIXmlDocumentSetProperty( ByVal hXMLDocument As MIXmlDocument,
    ByVal strPropertyName As String, ByVal strPropertyValue As String )
    As SmallInt
```

`hXMLDocument` – идентификатор объекта `MIXmlDocument`.

## Функция MIXmlGetAttributeList( )

---

strPropertyName – строка с именем свойства, которое требуется задать. Список свойств, которые можно задать этим методом, приводится в библиотеке MSDN.

strPropertyValue – строка со значением заданного свойства. Список значений свойств, которые можно задать этим методом, приводится в библиотеке MSDN.

### Возвращаемая величина

Ненулевое значение при успешном выполнении, иначе – 0.

### Описание

Этот метод используется для задания свойств объекта MIXmlDocument. Существуют некоторые ограничения в том, какие именно свойства можно задавать, используя этот метод. За подробной информацией, обращайтесь к Microsoft MSDN library.

См. также:

[Функция MIXmlDocumentCreate\( \)](#), [Функция MIXmlDocumentLoad\( \)](#)

---

## Функция MIXmlGetAttributeList( )

### Назначение

Получить объект MIXmlNamedNodeMap с заданным узлом.

### Синтаксис

```
MIXmlGetAttributeList( ByVal hXMLNode As MIXmlNode ) As MIXmlNamedNodeMap
```

hXMLNode – идентификатор объекта MIXmlNode.

### Возвращаемая величина

Идентификатор объекта MIXmlNamedNodeMap с узлами, от которых можно получить атрибуты. Значение NULL возвращается для всех остальных типов узлов.

### Описание

MIXmlGetAttributeList( ) – создает объект MIXmlNamedNodeMap и возвращает идентификатор объекта. В этом объекте могут присутствовать только узлы, атрибуты которых можно получить (узлы типа Element, Entity и Notation). Для всех остальных – возвращается значение Null. Для узлов, которые отвечают условиям, идентификатор объекта MIXmlNamedNodeMap всегда будет возвращаться; если элемент не атрибутирован, длина списка будет равной нулю. Подробная информация и список узлов, отвечающих условиям, приводится в библиотеке Microsoft MSDN.

При вызове требуется добавить идентификатор, для получения которого используется [Процедура MIXmlAttributeListDestroy\( \)](#), если возвращенный идентификатор больше не применяется.

См. также:

Функция [MIXmlDocumentGetRootNode\( \)](#), Процедура [MIXmlAttributeListDestroy\( \)](#)

---

## Функция [MIXmlGetChildList\( \)](#)

### Назначение

Получить объект `MIXmlNodeList` со списком потомков заданного узла.

### Синтаксис

```
MIXmlGetChildList( ByVal hXMLNode As MIXmlNode) As MIXmlNodeList
```

`hXMLNode` – идентификатор объекта `MIXmlNode`.

### Возвращаемая величина

Идентификатор объекта `MIXmlNodeList` со списком потомков заданного узла, если функция будет выполнена успешно; иначе `NULL`.

### Описание

`MIXmlGetChildList( )` – используется для получения списка дочерних узлов – потомков заданному. Объект `MIXmlNodeList` будет возвращен даже, если у узла нет потомков. В этом случае длина списка будет равна 0. Значение зависит от типа узла. За дополнительной информацией, обращайтесь к Microsoft MSDN library.

При вызове требуется добавить идентификатор, для получения которого используется [Процедура MIXmlNodeListDestroy\( \)](#), если такой идентификатор больше не применяется.

См. также:

[Процедура MIXmlNodeListDestroy\( \)](#), [Функция MIXmlSelectNodes\( \)](#)

---

## Функция [MIXmlGetNextAttribute\( \)](#)

### Назначение

Возвратить следующий узел коллекции.

### Синтаксис

```
MIXmlGetNextAttribute( ByVal hXMLAttributeList As MIXmlNamedNodeMap )  
As MIXmlNode
```

`hXMLAttributeList` – идентификатор объекта `MIXmlNamedNodeMap`.

### Возвращаемая величина

Если функция будет успешно выполнена, идентификатор объекта `MIXmlNode`, относящегося к следующему узлу коллекции; возвращается `NULL`, если следующего узла не существует.

### Описание

Итератор начинает работать с узла, предшествующего первому в списке, поэтому первый вызов функции MIXmlGetNextAttribute( ) возвращает первый узел в списке. Функция возвращает NULL, если текущий узел последний в списке или в списке ничего нет.

При вызове требуется добавить возвращенный идентификатор, для получения которого используется [Процедура MIXmlNodeDestroy\( \)](#), если такой идентификатор больше не применяется.

См. также:

[Функция MIXmlGetAttributeList\( \)](#), [Процедура MIXmlNodeDestroy\( \)](#)

---

## Функция MIXmlGetNextNode( )

### Назначение

Возвратить следующий узел коллекции.

### Синтаксис

```
MIXmlGetNextNode( ByVal hXMLNodeList As MIXmlNodeList ) As MIXmlNode
```

hXMLNodeList – идентификатор объекта MIXmlNodeList.

### Возвращаемая величина

Если функция будет успешно выполнена, идентификатор объекта hXMLNodeList, относящегося к следующему узлу коллекции; возвращается NULL, если следующего узла не существует.

### Описание

Итератор начинает работать с узла, предшествующего первому в списке, поэтому первый вызов функции MIXmlGetNextAttribute( ) возвращает первый узел в списке. Функция возвращает NULL, если текущий узел последний в списке или в списке ничего нет.

При вызове требуется добавить возвращенный идентификатор, для получения которого используется [Процедура MIXmlNodeDestroy\( \)](#), если такой идентификатор больше не применяется.

См. также:

[Процедура MIXmlNodeDestroy\( \)](#), [Функция MIXmlSelectNodes\( \)](#), [Функция MIXmlGetChildList\( \)](#)

---

## Процедура MIXmlNodeDestroy( )

### Назначение

Удалить объект MIXmlNode и освободить память.

**Синтаксис**

```
MIXmlNodeDestroy( ByVal hXMLNode As MIXmlNode )
```

hXMLNode идентификатор MIXmlNode удаляемого объекта.

**Описание**

При вызове требуется вызвать эту функцию, для того чтобы освободить идентификатор MIXmlNode, для получения которого используется **Функция MIXmlDocumentGetRootNode( )**, если такой идентификатор больше не используется.

**См. также:**

**Процедура MIXmlDocumentDestroy( )**

---

**Функция MIXmlNodeGetAttributeValue( )****Назначение**

Получить текст, ассоциированный с заданным именем.

**Синтаксис**

```
MIXmlNodeGetAttributeValue( ByVal hXMLNode As MIXmlNode,  
    ByVal strAttributeName As String, pValue As String,  
    ByVal nLen As Integer ) As SmallInt
```

hXMLNode – идентификатор объекта MIXmlNode.

strAttributeName – строка с заданным именем атрибута.

pValue – ссылка на строковую переменную, в которую будет помещено значение в узле с заданным атрибутом.

nObjectLen – размер буфера, на который ссылается pObject.

**Возвращаемая величина**

Ненулевое значение при успешном выполнении, иначе – 0.

**Описание**

MIXmlNodeGetAttributeValue( ) сначала находит объект MIXmlNamedNodeMap с заданным узлом, hXMLNode. Как заявлено в разделе, в котором описана **Функция MIXmlGetAttributeList( )**, такой объект содержит только узлы с возвращаемыми атрибутами (узлы типа Element, Entity и Notation). Если существует корректный объект MIXmlNamedNodeMap, заданное имя можно найти в этом объекте, значение в этом узле будет помещено в pValue.

**См. также:**

**Функция MIXmlGetAttributeList( )**, **Функция MIXmlNodeGetValue( )**

## Функция MIXmlNodeGetFirstChild( )

### Назначение

Возвратить первого потомка заданного узла.

### Синтаксис

```
MIXmlNodeGetFirstChild( ByVal hXMLNode As MIXmlNode ) As MIXmlNode
```

hXMLNode – идентификатор объекта MIXmlNode.

### Возвращаемая величина

Если функция будет выполнена успешно, идентификатор объекта MIXmlNode, который является первым потомком заданного узла hXMLNode; иначе NULL.

### Описание

MIXmlNodeGetFirstChild( ) – получает идентификатор объекта MIXmlNode, первого потомка заданного узла. Будет возвращено значение NULL, если подчиненного элемента не существует.

При вызове требуется добавить возвращенный идентификатор, для получения которого используется [Процедура MIXmlNodeDestroy\( \)](#), если такой идентификатор больше не применяется.

### См. также:

[Процедура MIXmlNodeDestroy\( \)](#), [Функция MIXmlNodeGetParent\( \)](#)

---

## Функция MIXmlNodeGetName( )

### Назначение

Получить имя заданного узла.

### Синтаксис

```
MIXmlNodeGetName( ByVal hXMLNode As MIXmlNode, pName As String,  
ByVal nLen As Integer ) As SmallInt
```

hXMLNode – идентификатор объекта MIXmlNode.

pName – ссылка на строковую переменную в которой будет храниться имя узла, которое может меняться в зависимости от типа узла.

nObjectLen – размер буфера, на который ссылается pObject.

### Возвращаемая величина

Ненулевое значение при успешном выполнении, иначе – 0.



**Описание**

Эта функция используется для того, чтобы получить имя заданного узла. Имя узла, это определенное название элемента, атрибута или ссылки на объект. Значение имени узла меняется в зависимости от типа узла.

**См. также:**

Функция [MIXmlDocumentGetRootNode\( \)](#), Функция [MIXmlNodeGetText\( \)](#), Функция [MIXmlNodeGetValue\( \)](#)

---

**Функция MIXmlNodeGetParent( )****Назначение**

Возвратить предка заданного узла.

**Синтаксис**

```
MIXmlNodeGetParent( ByVal hXMLNode As MIXmlNode ) As MIXmlNode
```

hXMLNode – идентификатор объекта MIXmlNode.

**Возвращаемая величина**

Если функция будет выполнена успешно, идентификатор объекта MIXmlNode, который является предком заданного узла hXMLNode; иначе NULL.

**Описание**

MIXmlNodeGetParent( ) – получает идентификатор объекта MIXmlNode, предка заданного узла. Будет возвращено значение NULL, если предка не существует.

При вызове требуется добавить идентификатор, для получения которого используется [Процедура MIXmlNodeDestroy\( \)](#), если такой идентификатор больше не применяется.

**См. также:**

[Процедура MIXmlNodeDestroy\( \)](#)

---

**Функция MIXmlNodeGetText( )****Назначение**

Получить текстовое содержимое заданного узла или сцепленный текст, относящийся к узлу и его потомкам.

**Синтаксис**

```
MIXmlNodeGetText( ByVal hXMLNode As MIXmlNode, pText As String,  
ByVal nLen As Integer ) As SmallInt
```

hXMLNode – идентификатор объекта MIXmlNode.

## Функция MIXmlNodeGetValue( )

---

pText – ссылка на строковую переменную, в которой будет храниться текстовое содержимое заданного узла и его потомков. Значение меняется в зависимости от типа узла.

nObjectLen – размер буфера, на который ссылается pObject.

### Возвращаемая величина

Ненулевое значение при успешном выполнении, иначе – 0.

### Описание

MIXmlNodegettext( ) – используется для того, чтобы получить текст в заданном узле. Значение меняется в зависимости от типа узла. Подробности и описание более точной работы с текстом в XML-документе, смотрите в библиотеке Microsoft MSDN.

### См. также:

[Функция MIXmlDocumentGetRootNode\( \)](#), [Функция MIXmlNodeGetName\( \)](#), [Функция MIXmlNodeGetValue\( \)](#)

---

## Функция MIXmlNodeGetValue( )

### Назначение

Получить текст ассоциированный с заданным узлом.

### Синтаксис

```
MIXmlNodeGetValue ( ByVal hXMLNode As MIXmlNode, pValue As String,  
                    ByVal nLen As Integer ) As SmallInt
```

hXMLNode – идентификатор объекта MIXmlNode.

pValue – ссылка на строковую переменную в которой будет храниться имя узла, которое может меняться в зависимости от типа узла.

nObjectLen – размер буфера, на который ссылается pObject.

### Возвращаемая величина

Ненулевое значение при успешном выполнении, иначе – 0.

### Описание

Эта функция используется для того, чтобы получить значение в заданном узле. Значение в узле меняется в зависимости от типа узла.

### См. также:

[Функция MIXmlDocumentGetRootNode\( \)](#), [Функция MIXmlNodeGetText\( \)](#), [Функция MIXmlNodeGetName\( \)](#)

---

## Процедура MIXmlNodeListDestroy( )

### Назначение

Удаляет объект MIXmlNodeList и освобождает память.

### Синтаксис

```
MIXmlNodeListDestroy( ByVal hXMLNodeList As MIXmlNodeList )
```

*hXMLNodeList* идентификатор MIXmlNodeList удаляемого объекта.

### Описание

Используйте функцию MIXmlNodeListDestroy( ) для того, чтобы освободить идентификатор MIXmlNodeList, полученный, вызывая другие функции, например, таких как: **Функция MIXmlSelectNodes( )** и **Функция MIXmlGetChildList( )**, если идентификатор MIXmlNodeList больше не используется.

### См. также:

**Процедура MIXmlDocumentDestroy( )**, **Функция MIXmlGetChildList( )**, **Функция MIXmlSelectNodes( )**

---

## Процедура MIXmlSCDestroy( )

### Назначение

Удалить объект MIXMLSchemaCollection и освободить память.

### Синтаксис

```
MIXmlSCDestroy( ByVal hXMLSchemaCollection As MIXMLSchemaCollection )
```

*hXMLSchemaCollection* идентификатор MIXMLSchemaCollection удаляемого объекта.

### Описание

Используйте функцию MIXmlSCDestroy( ) для того, чтобы освободить идентификатор MIXMLSchemaCollection, полученный, вызывая другие функции, например, такую как: **Функция MIXmlDocumentGetNamespaces( )**, если идентификатор больше не используется.

### См. также:

**Функция MIXmlDocumentGetNamespaces( )**

## Функция MIXmlSCGetLength( )

### Назначение

Получить пространство имен коллекции.

### Синтаксис

```
MIXmlSCGetLength( ByVal hXMLSchemaCollection As MIXMLSchemaCollection )  
    As Integer
```

*hXMLSchemaCollection* – идентификатор объекта MIXMLSchemaCollection.

### Возвращаемая величина

Число пространств имен коллекции.

### Описание

**MIXmlSCGetLength( )** позволяет получить число пространств имен, присутствующих в коллекции.

### См. также:

[Функция MIXmlDocumentGetNamespaces\( \)](#), [Функция MIXmlSCGetNamespace\( \)](#)

---

## Функция MIXmlSCGetNamespace( )

### Назначение

Получить пространство имен по заданному индексу.

### Синтаксис

```
MIXmlSCGetNamespace( ByVal hXMLSchemaCollection As MIXMLSchemaCollection,  
    ByVal index As Integer, pNamespace As String, ByVal nLen As Integer )  
    As SmallInt
```

*hXMLSchemaCollection* – идентификатор объекта MIXMLSchemaCollection.

*index* – целая переменная индекса от 0 до значения счетчика минус1.

*pNamespace* – ссылка на строку, в которой будет помещено имя множества имен.

*nLen* – размер буфера, на который ссылается *pNamespace*.

### Возвращаемая величина

Ненулевое значение при успешном выполнении, иначе – 0. Для того чтобы определить причину ошибки, обычно используется [Функция MIGetErrorMessage\( \)](#).

**Описание**

MIXmlISCGetNamespace( ) позволяет итеративно перебрать коллекцию и определить её содержимое.

**См. также:**

Функция [MIXmlDocumentGetNamespaces\( \)](#), Функция [MIXmlISCGetLength\( \)](#)

---

**Функция MIXmlSelectNodes( )**
**Назначение**

Сравнивает шаблон с контекстом узла и возвращает список подходящих узлов в объект MIXmlNodeList.

**Синтаксис**

```
MIXmlSelectNodes( ByVal hXMLNode As MIXmlNode, ByVal strPattern As String
    ) As MIXmlNodeList
```

*hXMLNode* – идентификатор объекта MIXmlNode.

*strPattern* – строковая переменная с выражением XPath.

**Возвращаемая величина**

Идентификатор объекта MIXmlNodeList. Это коллекция узлов, выбранных операцией сравнения с шаблоном. Если ни одного узла не выбрано, коллекция будет пустой. При ошибке, возвращается NULL.

**Описание**

MIXmlSelectNodes( ) – используется для того, чтобы передать коллекцию, совпадающих с шаблоном узлов после выполнения операции сравнения, в объект MIXmlNodeList. Функция MIXmlSelectNodes( ) работает также как [Функция MIXmlSelectSingleNode\( \)](#), но возвращает список совпадающих узлов, а не первый совпавший узел.

При вызове требуется добавить возвращенный идентификатор, для получения которого используется [Процедура MIXmlNodeListDestroy\( \)](#), если такой идентификатор больше не применяется.

**См. также:**

[Процедура MIXmlNodeListDestroy\( \)](#), [Функция MIXmlSelectSingleNode\( \)](#), [Функция MIXmlGetChildList\( \)](#)

## Функция MIXmlSelectSingleNode( )

### Назначение

Сравнивает шаблон с контекстом узла и возвращает первый совпавший узел в объект MIXmlNode.

### Синтаксис

```
MIXmlSelectSingleNode( ByVal hXMLNode As MIXmlNode,  
    ByVal strPattern As String ) As MIXmlNode
```

hXMLNode – идентификатор объекта MIXmlNode.

strPattern – строковая переменная с выражением XPath.

### Возвращаемая величина

Идентификатор объекта MIXmlNode. Возвращает первый совпадающий с шаблоном узел. Если совпадений не найдено, будет возвращено значение NULL.

### Описание

MIXmlSelectSingleNode( ) получает идентификатор объекта MIXmlNode, первого совпавшего с заданным шаблоном узла. Будет возвращено значение NULL, если потомка не существует. Функция MIXmlSelectSingleNode( ) работает также как [Функция MIXmlSelectNodes\( \)](#), но возвращает первый совпавший узел, а не список совпавших с заданным шаблоном узлов.

При вызове требуется добавить идентификатор, для получения которого используется [Процедура MIXmlNodeDestroy\( \)](#), если такой идентификатор больше не применяется.

### См. также:

[Процедура MIXmlNodeDestroy\( \)](#), [Функция MIXmlSelectNodes\( \)](#)

## C

# Character Code Table

The following table summarizes the displayable portion of the Windows Latin 1 character set. The range of characters from 32 (space) to 126 (tilde) are identical in most other character sets as well. Special characters of interest: 9 is a tab, 10 is a line feed, 12 is a form feed and 13 is a carriage return.

|      |      |       |       |       |       |     |
|------|------|-------|-------|-------|-------|-----|
| 2    | 64 @ | 96 `  | 128 ■ | 160   | 192 À | 224 |
| 3 !  | 65 A | 97 a  | 129 ■ | 161 ì | 193 Á | 225 |
| 4 "  | 66 B | 98 b  | 130 ■ | 162 ¢ | 194 Â | 226 |
| 5 #  | 67 C | 99 c  | 131 ■ | 163 £ | 195 Ã | 227 |
| 6 \$ | 68 D | 100 d | 132 ■ | 164 ¤ | 196 Ä | 228 |
| 7 %  | 69 E | 101 e | 133 ■ | 165 ¥ | 197 Å | 229 |
| 8 &  | 70 F | 102 f | 134 ■ | 166 ¦ | 198 Æ | 230 |
| 9 '  | 71 G | 103 g | 135 ■ | 167 § | 199 Ç | 231 |
| 0 (  | 72 H | 104 h | 136 ■ | 168 ¨ | 200 È | 232 |
| 1 )  | 73 I | 105 i | 137 ■ | 169 © | 201 É | 233 |
| 2 *  | 74 J | 106 j | 138 ■ | 170 ® | 202 Ê | 234 |
| 3 +  | 75 K | 107 k | 139 ■ | 171 « | 203 Ë | 235 |
| 4 ,  | 76 L | 108 l | 140 ■ | 172 ¬ | 204 Ì | 236 |
| 5 -  | 77 M | 109 m | 141 ■ | 173 ¯ | 205 Í | 237 |
| 6 .  | 78 N | 110 n | 142 ■ | 174 ® | 206 Î | 238 |
| 7 /  | 79 O | 111 o | 143 ■ | 175 ¯ | 207 Ï | 239 |
| 8 0  | 80 P | 112 p | 144 ■ | 176 ° | 208 Ð | 240 |
| 9 1  | 81 Q | 113 q | 145 ´ | 177 ± | 209 Ñ | 241 |
| 0 2  | 82 R | 114 r | 146 ´ | 178 º | 210 Ò | 242 |
| 1 3  | 83 S | 115 s | 147 ■ | 179 » | 211 Ó | 243 |
| 2 4  | 84 T | 116 t | 148 ■ | 180 ´ | 212 Ô | 244 |
| 3 5  | 85 U | 117 u | 149 ■ | 181 µ | 213 Õ | 245 |
| 4 6  | 86 V | 118 v | 150 ■ | 182 ¶ | 214 Ö | 246 |
| 5 7  | 87 W | 119 w | 151 ■ | 183 · | 215 × | 247 |
| 6 8  | 88 X | 120 x | 152 ■ | 184 ¸ | 216 Ø | 248 |
| 7 9  | 89 Y | 121 y | 153 ■ | 185 ¹ | 217 Ù | 249 |
| 8 :  | 90 Z | 122 z | 154 ■ | 186 º | 218 Ú | 250 |
| 9 ;  | 91 [ | 123 { | 155 ■ | 187 » | 219 Û | 251 |
| 0 <  | 92 \ | 124   | 156 ■ | 188 ¼ | 220 Ü | 252 |
| 1 =  | 93 ] | 125 } | 157 ■ | 189 ½ | 221 Ý | 253 |
| 2 >  | 94 ^ | 126 ~ | 158 ■ | 190 ¾ | 222 Þ | 254 |
| 3 ?  | 95 _ | 127   | 159 ■ | 191 ¿ | 223 ß | 255 |





# Сведения об операторах

Операторы производят некоторое действие над одним или несколькими значениями. Операторы можно классифицировать либо по типу значений, над которыми выполняются операции, либо по типу значения получаемого результата.

## В этом приложении:

|   |    |
|---|----|
| ♦ Числовые операторы .....                  | 20 |
| ♦ Операторы сравнения .....                 | 20 |
| ♦ Логические Операторы .....                | 21 |
| ♦ Географические операторы .....            | 21 |
| ♦ Автоматическое преобразование типов ..... | 23 |

## Числовые операторы

Следующие операторы выполняются с двумя числовыми значениями, а в результате должно быть получено числовое значение.

| Оператор | Действие                                 | Пример:            |
|----------|--|--------------------|
| +        | сложение                                 | $a + b$            |
| -        | вычитание                                | $a - b$            |
| *        | умножение                                | $a * b$            |
| /        | деление                                  | $a / b$            |
| \        | целочисленное деление (остаток отброшен) | $a \setminus b$    |
| Mod      | остаток целочисленного деления           | $a \text{ Mod } b$ |
| ^        | возведение в степень                     | $a ^ b$            |

Два оператора из перечисленных выше могут быть использованы и в другом значении. Знак сложения (+) используется для конкатенации (объединения) двух строк в третью. Знак минус (-) вместе с единственным числом используется в качестве оператора отрицания, результатом выполнения будет числовое значение. Знак амперсанда (&) также используется для конкатенации строк.

| Оператор | Действие            | Пример:  |
|----------|---------------------|----------|
| -        | численное отрицание | $- a$    |
| +        | конкатенация строк  | $a + b$  |
| &        | конкатенация строк  | $a \& b$ |

## Операторы сравнения

С помощью операторов сравнения две величины одного типа сопоставляются друг другу, результатом выполнения оператора будет логическая величина TRUE или FALSE. Хотя операторы сравнения нельзя использовать для сопоставления числовых и нечисловых данных (например, со строковыми выражениями), допустимо сравнение данных целых, коротких целых и вещественных типов. Операторы сравнения часто используются в выражениях с условием, например, If...Then.

| Оператор | Возвращается TRUE если: | Пример: |
|----------|-------------------------|---------|
| =        | a равно b               | a = b   |
| <>       | a не равно b            | a <> b  |
| <        | a меньше b              | a < b   |
| >        | a больше b              | a > b   |
| <=       | a меньше или равно b    | a <= b  |
| >=       | a больше или равно b    | a >= b  |

Логические Операторы

Логические операторы выполняются над логическими значениями, а результатом их выполнения является логическая величина либо TRUE, либо FALSE:

| Оператор          | Возвращается TRUE если:               | Пример: |
|-------------------|---------------------------------------|---------|
| And               | оба операнда истинны                  | a And b |
| Or (оператор Или) | значение одного из операндов – истина | a Or b  |
| Not               | значение операнда FALSE               | Not a   |

Географические операторы

Географические операторы выполняются над объектами, а результатом их выполнения является логическая величина либо TRUE, либо FALSE:

| Оператор        | Возвращается TRUE если:                             | Пример:                         |
|-----------------|---|---------------------------------|
| Contains        | Первый объект A содержит центроид второго объекта B | objectA Contains objectB        |
| Contains Part   | Объект A содержит некоторую часть объекта B         | objectA Contains Part objectB   |
| Contains Entire | Объект A содержит весь объект B                     | objectA Contains Entire objectB |

| Оператор        | Возвращается TRUE если:                           | Пример:                         |
|-----------------|---|---------------------------------|
| Within          | центроид первого объекта внутри второго объекта   | objectA Within objectB          |
| Partly Within   | часть первого объекта помещается внутри второго   | objectA Partly Within objectB   |
| Entirely Within | первый объект полностью помещается внутри второго | objectA Entirely Within objectB |
| Intersects      | Два объекта пересекаются хотя бы в одной точке    | objectA Intersects objectB      |

## Старшинство выполнения операторов

Специальным типом операторов являются скобки, в которые можно заключать выражения внутри выражений. С помощью скобок можно изменить порядок выполнения операторов. В таблице ниже приведен порядок выполнения операторов MapBasic. Операторы из одной строки имеют одинаковый порядок выполнения. Операторы с более высоким приоритетом выполняются ранее других. Операторы имеющие одинаковый приоритет выполняются слева направо (кроме операторов возведения в степень, котрый выполняется справа налево).

| Порядок выполнения операторов MapBasic | Операторы                                       |
|--|---|
| (В первую очередь)                     | скобки  |
|  | возведение в степень                            |
|  | отрицательный знак                              |
|  | умножение, деление, Mod, целочисленное деление, |
|  | сложение, вычитание                             |
|  | географические операторы                        |
|  | операторы сравнения, операторы Like             |
|  | Not   |
|  | And   |
| (в последнюю очередь)                  | Or (оператор Или)                               |

Например, результатом выполнения выражения  $3 + 4 * 2$  будет 11 (умножение выполняется до сложения). Результатом выполнения слегка измененного выражения  $(3 + 4) * 2$  будет 14 (с помощью скобок сложение выполняется в первую очередь). Если есть сомнения, используйте скобки.

## Автоматическое преобразование типов

Если создать выражение обрабатывающее данные разных типов, то, для того чтобы получить осмысленный результат, MapInfo Professional будет автоматически преобразовывать выполнять типы данных. Например, если программа вычитает дату из другой даты, MapBasic вычислит целое значение (определяющее количество дней прошедшее между двумя датами).

В таблице ниже приведен список правил, по которым MapBasic'ом выполняется автоматическое преобразование типов данных. В этой таблице слово "Целое" означает целое значение, которое может представлять собой целую переменную, короткую целую переменную или целую константу. Слово "Число" обозначает числовое выражение, которое обязательно является целочисленным.

| Оператор | Комбинация операндов | Результат                      |
|----------|----------------------|--------------------------------|
| +        | Дата + Число         | Дата типа Date                 |
|          | Число + Дата         | Дата типа Date                 |
|          | Целое + Целое        | Целое число типа Integer.      |
|          | Число + Число        | Вещественное число типа Float. |
|          | Другое + Другое      | Строка                         |
| -        | Дата - Число         | Дата типа Date                 |
|          | Дата - дата          | Целое число типа Integer.      |
|          | Целое - Целое        | Целое число типа Integer.      |
|          | Число - Число        | Вещественное число типа Float. |

| Оператор | Комбинация операндов | Результат                      |
|----------|----------------------|--------------------------------|
| *        | Целое * Целое        | Целое число типа Integer.      |
|          | Число * Число        | Вещественное число типа Float. |
| /        | Число / Число        | Вещественное число типа Float. |
| \        | Число \ Число        | Целое число типа Integer.      |
| MOD      | Число MOD Число      | Целое число типа Integer.      |
| ^        | Число ^ Число        | Вещественное число типа Float. |

# MapBasic Definitions File

The following MAPBASIC.DEF file lists definitions and defaults useful when programming in MapBasic.

## MapBasic.DEF файл

This file is installed in the MapBasic directory:

```
'=====
' MapInfo version 9.0 - System defines
'-----
' This file contains defines useful when programming in the MapBasic
' language. There are three versions of this file:
'     MAPBASIC.DEF - MapBasic syntax
'     MAPBASIC.BAS - Visual Basic syntax
'     MAPBASIC.H   - C/C++ syntax
'-----
' The defines in this file are organized into the following sections:
'     General Purpose defines:
'         macros, logical constants, angle conversion, colors, string length
'     ButtonPadInfo() defines
'     ColumnInfo() and column type defines
'     CommandInfo() and task switch defines
'     DateWindow() defines
'     FileAttr() and file access mode defines
'     GetFolderPath$() defines
'     IntersectNodes() parameters
'     LabelInfo() defines
'     LayerInfo(), display mode, label property, layer type, hotlink defines
'     LegendInfo() and legend orientation defines
'     LegendFrameInfo() and frame type defines
'     LegendStyleInfo() defines
'     LocateFile$() defines
'     Map3DInfo() defines
'     MapperInfo(), display mode, calculation type, and clip type defines
'     MenuItemInfoByID() and MenuItemInfoByHandler() defines
'     ObjectGeography() defines
'     ObjectInfo() and object type defines
'     PrismMapInfo() defines
'     SearchInfo() defines
'     SelectionInfo() defines
'     Server statement and function defines
```



```

' SessionInfo() defines
' Set Next Document Style defines
' StringCompare() return values
' StyleAttr() defines
' SystemInfo(), platform, and version defines
' TableInfo() and table type defines
' WindowInfo(), window type and state, and print orientation defines
' Abbreviated list of error codes
' Backward Compatibility defines
'=====
' MAPBASIC.DEF is converted into MAPBASIC.H by doing the following:
' - concatenate MAPBASIC.DEF and MENU.DEF into MAPBASIC.H
' - search & replace "\"" at beginning of a line with "/"
' - search & replace "Define" at beginning of a line with "#define"
' - delete the following sections:
'     * General Purpose defines:
'         Macros, Logical Constants, Angle Conversions
'     * Abbreviated list of error codes
'     * Backward Compatibility defines
'     * Menu constants whose names have changed
'     * Obsolete menu items
'=====
' MAPBASIC.DEF is converted into MAPBASIC.BAS by doing the following:
' - concatenate MAPBASIC.DEF and MENU.DEF into MAPBASIC.BAS
' - search & replace "Define <name>" with "Global Const <name> ="
'     e.g. "<Define {[!-z]+} +{[!-z]}" with "Global Const \0 = \1" with Brief
' - delete the following sections:
'     * General Purpose defines:
'         Macros, Logical Constants, Angle Conversions
'     * Abbreviated list of error codes
'     * Backward Compatibility defines
'     * Menu constants whose names have changed
'     * Obsolete menu items
'=====
'=====

```

```
' General Purpose defines
'=====
'-----
' Macros
'-----

Define CLS                                Print Chr$(12)

'-----

' Logical constants
'-----

Define TRUE                                1
Define FALSE                               0

'-----

' Angle conversion
'-----

Define DEG_2_RAD                           0.01745329252
Define RAD_2_DEG                           57.29577951

'-----

' Time conversion
'-----

Define SECONDS_PER_DAY                     86400

'-----

' Colors
'-----

Define BLACK                               0
Define WHITE                               16777215
Define RED                                 16711680
Define GREEN                               65280
Define BLUE                                255
Define CYAN                               65535
Define MAGENTA                             16711935
Define YELLOW                             16776960
```

```

'-----
'Maximum length for character string
'-----

Define MAX_STRING_LENGTH                      32767

'=====
' ButtonPadInfo() defines
'=====

Define BTNPAD_INFO_FLOATING                   1
Define BTNPAD_INFO_WIDTH                      2
Define BTNPAD_INFO_NBTNS                     3
Define BTNPAD_INFO_X                         4
Define BTNPAD_INFO_Y                         5
Define BTNPAD_INFO_WINID                     6

'=====
' ColumnInfo() defines
'=====

Define COL_INFO_NAME                         1
Define COL_INFO_NUM                          2
Define COL_INFO_TYPE                         3
Define COL_INFO_WIDTH                       4
Define COL_INFO_DECPLACES                   5
Define COL_INFO_INDEXED                     6
Define COL_INFO_EDITABLE                     7

'-----
' Column type defines, returned by ColumnInfo() for COL_INFO_TYPE
'-----

Define COL_TYPE_CHAR                         1
Define COL_TYPE_DECIMAL                      2
Define COL_TYPE_INTEGER                      3
Define COL_TYPE_SMALLINT                     4
Define COL_TYPE_DATE                         5
Define COL_TYPE_LOGICAL                      6
Define COL_TYPE_GRAPHIC                      7

```

|                          |    |
|--------------------------|----|
| Define COL_TYPE_FLOAT    | 8  |
| Define COL_TYPE_TIME     | 37 |
| Define COL_TYPE_DATETIME | 38 |

'=====

' CommandInfo() defines

'=====

|                               |      |
|-------------------------------|------|
| Define CMD_INFO_X             | 1    |
| Define CMD_INFO_Y             | 2    |
| Define CMD_INFO_SHIFT         | 3    |
| Define CMD_INFO_CTRL          | 4    |
| Define CMD_INFO_X2            | 5    |
| Define CMD_INFO_Y2            | 6    |
| Define CMD_INFO_TOOLBTN       | 7    |
| Define CMD_INFO_MENUITEM      | 8    |
| Define CMD_INFO_WIN           | 1    |
| Define CMD_INFO_SELTYPE       | 1    |
| Define CMD_INFO_ROWID         | 2    |
| Define CMD_INFO_INTERRUPT     | 3    |
| Define CMD_INFO_STATUS        | 1    |
| Define CMD_INFO_MSG           | 1000 |
| Define CMD_INFO_DLG_OK        | 1    |
| Define CMD_INFO_DLG_DBL       | 1    |
| Define CMD_INFO_FIND_RC       | 3    |
| Define CMD_INFO_FIND_ROWID    | 4    |
| Define CMD_INFO_XCMD          | 1    |
| Define CMD_INFO_CUSTOM_OBJ    | 1    |
| Define CMD_INFO_TASK_SWITCH   | 1    |
| Define CMD_INFO_EDIT_TABLE    | 1    |
| Define CMD_INFO_EDIT_STATUS   | 2    |
| Define CMD_INFO_EDIT_ASK      | 1    |
| Define CMD_INFO_EDIT_SAVE     | 2    |
| Define CMD_INFO_EDIT_DISCARD  | 3    |
| Define CMD_INFO_HL_WINDOW_ID  | 17   |
| Define CMD_INFO_HL_TABLE_NAME | 18   |
| Define CMD_INFO_HL_ROWID      | 19   |

```

Define CMD_INFO_HL_LAYER_ID                20
Define CMD_INFO_HL_FILE_NAME              21

'-----
' Task Switches, returned by CommandInfo() for CMD_INFO_TASK_SWITCH
'-----

Define SWITCHING_OUT_OF_MAPINFO           0
Define SWITCHING_INTO_MAPINFO             1

'=====
' DateWindow() defines
'=====

Define DATE_WIN_SESSION                   1
Define DATE_WIN_CURPROG                   2

'=====
' FileAttr() defines
'=====

Define FILE_ATTR_MODE                     1
Define FILE_ATTR_FILESIZE                 2

'-----
' File Access Modes, returned by FileAttr() for FILE_ATTR_MODE
'-----

Define MODE_INPUT                         0
Define MODE_OUTPUT                        1
Define MODE_APPEND                        2
Define MODE_RANDOM                        3
Define MODE_BINARY                        4

'=====
' GetFolderPath$() defines
'=====

Define FOLDER_MI_APPDATA                  -1
Define FOLDER_MI_LOCAL_APPDATA            -2

```

|                                 |    |
|---------------------------------|----|
| Define FOLDER_MI_PREFERENCE     | -3 |
| Define FOLDER_MI_COMMON_APPDATA | -4 |
| Define FOLDER_APPDATA           | 26 |
| Define FOLDER_LOCAL_APPDATA     | 28 |
| Define FOLDER_COMMON_APPDATA    | 35 |
| Define FOLDER_COMMON_DOCS       | 46 |
| Define FOLDER_MYDOCS            | 5  |
| Define FOLDER_MYPICS            | 39 |

'=====

' IntersectNodes() defines

'=====

|                       |   |
|-----------------------|---|
| Define INCL_CROSSINGS | 1 |
| Define INCL_COMMON    | 6 |
| Define INCL_ALL       | 7 |

'=====

' LabelInfo() defines

'=====

|                                   |    |
|-----------------------------------|----|
| Define LABEL_INFO_OBJECT          | 1  |
| Define LABEL_INFO_POSITION        | 2  |
| Define LABEL_INFO_ANCHORX         | 3  |
| Define LABEL_INFO_ANCHORY         | 4  |
| Define LABEL_INFO_OFFSET          | 5  |
| Define LABEL_INFO_ROWID           | 6  |
| Define LABEL_INFO_TABLE           | 7  |
| Define LABEL_INFO_EDIT            | 8  |
| Define LABEL_INFO_EDIT_VISIBILITY | 9  |
| Define LABEL_INFO_EDIT_ANCHOR     | 10 |
| Define LABEL_INFO_EDIT_OFFSET     | 11 |
| Define LABEL_INFO_EDIT_FONT       | 12 |
| Define LABEL_INFO_EDIT_PEN        | 13 |
| Define LABEL_INFO_EDIT_TEXT       | 14 |
| Define LABEL_INFO_EDIT_TEXTARROW  | 15 |
| Define LABEL_INFO_EDIT_ANGLE      | 16 |
| Define LABEL_INFO_EDIT_POSITION   | 17 |

```

Define LABEL_INFO_EDIT_TEXTLINE      18
Define LABEL_INFO_SELECT              19
Define LABEL_INFO_DRAWN               20
Define LABEL_INFO_ORIENTATION         21

```

```
'=====
```

```
' LayerInfo() defines
```

```
'=====
```

```

Define LAYER_INFO_NAME                1
Define LAYER_INFO_EDITABLE            2
Define LAYER_INFO_SELECTABLE          3
Define LAYER_INFO_ZOOM_LAYERED        4
Define LAYER_INFO_ZOOM_MIN            5
Define LAYER_INFO_ZOOM_MAX            6
Define LAYER_INFO_COSMETIC            7
Define LAYER_INFO_PATH                8
Define LAYER_INFO_DISPLAY             9
Define LAYER_INFO_OVR_LINE           10
Define LAYER_INFO_OVR_PEN            11
Define LAYER_INFO_OVR_BRUSH          12
Define LAYER_INFO_OVR_SYMBOL          13
Define LAYER_INFO_OVR_FONT           14
Define LAYER_INFO_LBL_EXPR           15
Define LAYER_INFO_LBL_LT             16
Define LAYER_INFO_LBL_CURFONT        17
Define LAYER_INFO_LBL_FONT           18
Define LAYER_INFO_LBL_PARALLEL       19
Define LAYER_INFO_LBL_POS            20
Define LAYER_INFO_ARROWS             21
Define LAYER_INFO_NODES              22
Define LAYER_INFO_CENTROIDS          23
Define LAYER_INFO_TYPE               24
Define LAYER_INFO_LBL_VISIBILITY     25
Define LAYER_INFO_LBL_ZOOM_MIN       26
Define LAYER_INFO_LBL_ZOOM_MAX       27
Define LAYER_INFO_LBL_AUTODISPLAY    28

```

```
Define LAYER_INFO_LBL_OVERLAP          29
Define LAYER_INFO_LBL_DUPLICATES       30
Define LAYER_INFO_LBL_OFFSET           31
Define LAYER_INFO_LBL_MAX              32
Define LAYER_INFO_LBL_PARTIALSEGS      33
Define LAYER_INFO_HOTLINK_EXPR         34
Define LAYER_INFO_HOTLINK_MODE         35
Define LAYER_INFO_HOTLINK_RELATIVE     36
Define LAYER_INFO_HOTLINK_COUNT        37
Define LAYER_INFO_LBL_ORIENTATION      38
```

```
'-----
' Values returned by LayerInfo() for LAYER_INFO_LABEL_ORIENTATION and
' LABEL_INFO_ORIENTATION.
'-----
```

```
Define LAYER_INFO_LABEL_ORIENT_HORIZONTAL  0
Define LAYER_INFO_LABEL_ORIENT_PARALLEL    1
Define LAYER_INFO_LABEL_ORIENT_CURVED      2
```

```
'-----
' Display Modes, returned by LayerInfo() for LAYER_INFO_DISPLAY
'-----
```

```
Define LAYER_INFO_DISPLAY_OFF              0
Define LAYER_INFO_DISPLAY_GRAPHIC          1
Define LAYER_INFO_DISPLAY_GLOBAL           2
Define LAYER_INFO_DISPLAY_VALUE            3
```

```
'-----
' Label Linetypes, returned by LayerInfo() for LAYER_INFO_LBL_LT
'-----
```

```
Define LAYER_INFO_LBL_LT_NONE              0
Define LAYER_INFO_LBL_LT_SIMPLE            1
Define LAYER_INFO_LBL_LT_ARROW             2
```

```
'-----
' Label Positions, returned by LayerInfo() for LAYER_INFO_LBL_POS
```



```

'-----
Define LAYER_INFO_LBL_POS_CC          0
Define LAYER_INFO_LBL_POS_TL          1
Define LAYER_INFO_LBL_POS_TC          2
Define LAYER_INFO_LBL_POS_TR          3
Define LAYER_INFO_LBL_POS_CL          4
Define LAYER_INFO_LBL_POS_CR          5
Define LAYER_INFO_LBL_POS_BL          6
Define LAYER_INFO_LBL_POS_BC          7
Define LAYER_INFO_LBL_POS_BR          8

'-----
' Layer Types, returned by LayerInfo() for LAYER_INFO_TYPE
'-----
Define LAYER_INFO_TYPE_NORMAL          0
Define LAYER_INFO_TYPE_COSMETIC        1
Define LAYER_INFO_TYPE_IMAGE           2
Define LAYER_INFO_TYPE_THEMATIC        3
Define LAYER_INFO_TYPE_GRID            4
Define LAYER_INFO_TYPE_WMS             5

'-----
' Label visibility modes, from LayerInfo() for LAYER_INFO_LBL_VISIBILITY
'-----
Define LAYER_INFO_LBL_VIS_OFF          1
Define LAYER_INFO_LBL_VIS_ZOOM         2
Define LAYER_INFO_LBL_VIS_ON           3

'-----
' HotlinkInfo() defines
'-----
Define HOTLINK_INFO_EXPR               1
Define HOTLINK_INFO_MODE               2
Define HOTLINK_INFO_RELATIVE           3
Define HOTLINK_INFO_ENABLED            4

```

```
'-----
' Hotlink activation modes, from LayerInfo() for LAYER_INFO_HOTLINK_MODE
'-----

Define HOTLINK_MODE_LABEL          0
Define HOTLINK_MODE_OBJ            1
Define HOTLINK_MODE_BOTH           2

'=====
' LegendInfo() defines
'=====

Define LEGEND_INFO_MAP_ID          1
Define LEGEND_INFO_ORIENTATION     2
Define LEGEND_INFO_NUM_FRAMES      3
Define LEGEND_INFO_STYLE_SAMPLE_SIZE 4

'=====
' Orientation codes, returned by LegendInfo() for LEGEND_INFO_ORIENTATION
'=====

Define ORIENTATION_PORTRAIT        1
Define ORIENTATION_LANDSCAPE       2
Define ORIENTATION_CUSTOM          3

'-----
' Style sample codes, from LegendInfo() for LEGEND_INFO_STYLE_SAMPLE_SIZE
'-----

Define STYLE_SAMPLE_SIZE_SMALL     0
Define STYLE_SAMPLE_SIZE_LARGE     1

'=====
' LegendFrameInfo() defines
'=====

Define FRAME_INFO_TYPE             1
Define FRAME_INFO_MAP_LAYER_ID     2
Define FRAME_INFO_REFRESHABLE      3
Define FRAME_INFO_POS_X            4
Define FRAME_INFO_POS_Y            5
```

```

Define FRAME_INFO_WIDTH                6
Define FRAME_INFO_HEIGHT               7
Define FRAME_INFO_TITLE                8
Define FRAME_INFO_TITLE_FONT           9
Define FRAME_INFO_SUBTITLE            10
Define FRAME_INFO_SUBTITLE_FONT        11
Define FRAME_INFO_BORDER_PEN           12
Define FRAME_INFO_NUM_STYLES           13
Define FRAME_INFO_VISIBLE               14
Define FRAME_INFO_COLUMN               15
Define FRAME_INFO_LABEL                 16

'=====
' Frame Types, returned by LegendFrameInfo() for FRAME_INFO_TYPE
'=====

Define FRAME_TYPE_STYLE                 1
Define FRAME_TYPE_THEME                 2

'=====
' Geocode Attributes, returned by GeocodeInfo()
'=====

Define GEOCODE_STREET_NAME              1
Define GEOCODE_STREET_NUMBER            2
Define GEOCODE_MUNICIPALITY             3
Define GEOCODE_MUNICIPALITY2            4
Define GEOCODE_COUNTRY_SUBDIVISION      5
Define GEOCODE_COUNTRY_SUBDIVISION2     6
Define GEOCODE_POSTAL_CODE              7

Define GEOCODE_DICTIONARY                9
Define GEOCODE_BATCH_SIZE               10
Define GEOCODE_FALLBACK_GEOGRAPHIC      11
Define GEOCODE_FALLBACK_POSTAL          12
Define GEOCODE_OFFSET_CENTER            13
Define GEOCODE_OFFSET_CENTER_UNITS      14
Define GEOCODE_OFFSET_END               15

```

```
Define GEOCODE_OFFSET_END_UNITS          16
Define GEOCODE_MIXED_CASE                 17
Define GEOCODE_RESULT_MARK_MULTIPLE      18
Define GEOCODE_COUNT_GEOCODED            19
Define GEOCODE_COUNT_NOTGEOCODED         20
Define GEOCODE_UNABLE_TO_CONVERT_DATA    21
Define GEOCODE_MAX_BATCH_SIZE             22
Define GEOCODE_PASSTHROUGH                100
```

```
Define DICTIONARY_ALL                     1
Define DICTIONARY_ADDRESS_ONLY            2
Define DICTIONARY_USER_ONLY               3
Define DICTIONARY_PREFER_ADDRESS          4
Define DICTIONARY_PREFER_USER             5
```

```
'=====
' ISOGRAM Attributes, returned by IsogramInfo()
'=====
```

```
Define ISOGRAM_BANDING                    1
Define ISOGRAM_MAJOR_ROADS_ONLY           2
Define ISOGRAM_RETURN_HOLES               3
Define ISOGRAM_MAJOR_POLYGON_ONLY         4
Define ISOGRAM_MAX_OFFROAD_DIST           5
Define ISOGRAM_MAX_OFFROAD_DIST_UNITS     6
Define ISOGRAM_SIMPLIFICATION_FACTOR       7
Define ISOGRAM_DEFAULT_AMBIENT_SPEED      8
Define ISOGRAM_AMBIENT_SPEED_DIST_UNIT    9
Define ISOGRAM_AMBIENT_SPEED_TIME_UNIT    10
Define ISOGRAM_PROPAGATION_FACTOR         11
Define ISOGRAM_BATCH_SIZE                 12
Define ISOGRAM_POINTS_ONLY                13
Define ISOGRAM_RECORDS_INSERTED           14
Define ISOGRAM_RECORDS_NOTINSERTED        15
Define ISOGRAM_MAX_BATCH_SIZE             16
Define ISOGRAM_MAX_BANDS                  17
Define ISOGRAM_MAX_DISTANCE               18
```

```

Define ISOGRAM_MAX_DISTANCE_UNITS      19
Define ISOGRAM_MAX_TIME                 20
Define ISOGRAM_MAX_TIME_UNITS           21

```

```

'=====
' LegendStyleInfo() defines
'=====
Define LEGEND_STYLE_INFO_TEXT           1
Define LEGEND_STYLE_INFO_FONT           2
Define LEGEND_STYLE_INFO_OBJ            3

```

```

'=====
' LocateFile$() defines
'=====

```

```

Define LOCATE_PREF_FILE                  0
Define LOCATE_DEF_WOR                    1
Define LOCATE_CLR_FILE                   2
Define LOCATE_PEN_FILE                   3
Define LOCATE_FNT_FILE                   4
Define LOCATE_ABB_FILE                   5
Define LOCATE_PRJ_FILE                   6
Define LOCATE_MNU_FILE                   7
Define LOCATE_CUSTSYMB_DIR               8
Define LOCATE_THMTMPLT_DIR              9
Define LOCATE_GRAPH_DIR                 10
Define LOCATE_WMS_SERVERLIST            11
Define LOCATE_WFS_SERVERLIST            12
Define LOCATE_GEOCODE_SERVERLIST        13
Define LOCATE_ROUTING_SERVERLIST        14
Define LOCATE_LAYOUT_TEMPLATE_DIR       15

```

```

'=====
' Map3DInfo() defines
'=====
Define MAP3D_INFO_SCALE                  1

```

```
Define MAP3D_INFO_RESOLUTION_X      2
Define MAP3D_INFO_RESOLUTION_Y      3
Define MAP3D_INFO_BACKGROUND        4
Define MAP3D_INFO_UNITS              5
Define MAP3D_INFO_LIGHT_X           6
Define MAP3D_INFO_LIGHT_Y           7
Define MAP3D_INFO_LIGHT_Z           8
Define MAP3D_INFO_LIGHT_COLOR        9
Define MAP3D_INFO_CAMERA_X          10
Define MAP3D_INFO_CAMERA_Y          11
Define MAP3D_INFO_CAMERA_Z          12
Define MAP3D_INFO_CAMERA_FOCAL_X    13
Define MAP3D_INFO_CAMERA_FOCAL_Y    14
Define MAP3D_INFO_CAMERA_FOCAL_Z    15
Define MAP3D_INFO_CAMERA_VU_1       16
Define MAP3D_INFO_CAMERA_VU_2       17
Define MAP3D_INFO_CAMERA_VU_3       18
Define MAP3D_INFO_CAMERA_VPN_1      19
Define MAP3D_INFO_CAMERA_VPN_2      20
Define MAP3D_INFO_CAMERA_VPN_3      21
Define MAP3D_INFO_CAMERA_CLIP_NEAR   22
Define MAP3D_INFO_CAMERA_CLIP_FAR    23

'=====
' MapperInfo() defines
'=====

Define MAPPER_INFO_ZOOM              1
Define MAPPER_INFO_SCALE              2
Define MAPPER_INFO_CENTERX           3
Define MAPPER_INFO_CENTERY           4
Define MAPPER_INFO_MINX              5
Define MAPPER_INFO_MINY              6
Define MAPPER_INFO_MAXX              7
Define MAPPER_INFO_MAXY              8
Define MAPPER_INFO_LAYERS            9
Define MAPPER_INFO_EDIT_LAYER        10
```

```

Define MAPPER_INFO_XYUNITS          11
Define MAPPER_INFO_DISTUNITS        12
Define MAPPER_INFO_AREAUNITS        13
Define MAPPER_INFO_SCROLLBARS       14
Define MAPPER_INFO_DISPLAY          15
Define MAPPER_INFO_NUM_THEMATIC      16
Define MAPPER_INFO_COORDSYS_CLAUSE  17
Define MAPPER_INFO_COORDSYS_NAME     18
Define MAPPER_INFO_MOVE_DUPLICATE_NODES 19
Define MAPPER_INFO_DIST_CALC_TYPE    20
Define MAPPER_INFO_DISPLAY_DMS       21
Define MAPPER_INFO_COORDSYS_CLAUSE_WITH_BOUNDS 22
Define MAPPER_INFO_CLIP_TYPE         23
Define MAPPER_INFO_CLIP_REGION       24
Define MAPPER_INFO_REPROJECTION      25
Define MAPPER_INFO_RESAMPLING        26
Define MAPPER_INFO_MERGE_MAP         27

```

```

'-----
' Display Modes, returned by MapperInfo() for MAPPER_INFO_DISPLAY_DMS
'-----

```

```

Define MAPPER_INFO_DISPLAY_DECIMAL    0
Define MAPPER_INFO_DISPLAY_DEGMINSEC  1
Define MAPPER_INFO_DISPLAY_MGRS       2

```

```

'-----
' Display Modes, returned by MapperInfo() for MAPPER_INFO_DISPLAY
'-----

```

```

Define MAPPER_INFO_DISPLAY_SCALE      0
Define MAPPER_INFO_DISPLAY_ZOOM       1
Define MAPPER_INFO_DISPLAY_POSITION   2

```

```

'-----
' Distance Calculation Types from MapperInfo() for MAPPER_INFO_DIST_CALC_TYPE
'-----

```

```

Define MAPPER_INFO_DIST_SPHERICAL     0

```

```
Define MAPPER_INFO_DIST_CARTESIAN          1

'-----
' Clip Types, returned by MapperInfo() for MAPPER_INFO_CLIP_TYPE
'-----

Define MAPPER_INFO_CLIP_DISPLAY_ALL        0
Define MAPPER_INFO_CLIP_DISPLAY_POLYOBJ    1
Define MAPPER_INFO_CLIP_OVERLAY            2

'=====
' MenuItemInfoByID() and MenuItemInfoByHandler() defines
'=====

Define MENUITEM_INFO_ENABLED               1
Define MENUITEM_INFO_CHECKED               2
Define MENUITEM_INFO_CHECKABLE             3
Define MENUITEM_INFO_SHOWHIDEABLE         4
Define MENUITEM_INFO_ACCELERATOR           5
Define MENUITEM_INFO_TEXT                  6
Define MENUITEM_INFO_HELPMSG               7
Define MENUITEM_INFO_HANDLER               8
Define MENUITEM_INFO_ID                    9

'=====
' ObjectGeography() defines
'=====

Define OBJ_GEO_MINX                        1
Define OBJ_GEO_LINEBEGX                    1
Define OBJ_GEO_POINTX                      1
Define OBJ_GEO_MINY                        2
Define OBJ_GEO_LINEBEGY                    2
Define OBJ_GEO_POINTY                      2
Define OBJ_GEO_MAXX                        3
Define OBJ_GEO_LINEENDX                    3
Define OBJ_GEO_MAXY                        4
Define OBJ_GEO_LINEENDY                    4
Define OBJ_GEO_ARCBEGANGLE                 5
```



---

|                            |   |
|----------------------------|---|
| Define OBJ_GEO_TEXTLINEX   | 5 |
| Define OBJ_GEO_ROUNDRAIDUS | 5 |
| Define OBJ_GEO_CENTROID    | 5 |
| Define OBJ_GEO_ARCENDANGLE | 6 |
| Define OBJ_GEO_TEXTLINEY   | 6 |
| Define OBJ_GEO_TEXTANGLE   | 7 |
| Define OBJ_GEO_POINTZ      | 8 |
| Define OBJ_GEO_POINTM      | 9 |

'=====

' ObjectInfo() defines

'=====

|                             |    |
|-----------------------------|----|
| Define OBJ_INFO_TYPE        | 1  |
| Define OBJ_INFO_PEN         | 2  |
| Define OBJ_INFO_SYMBOL      | 2  |
| Define OBJ_INFO_TEXTFONT    | 2  |
| Define OBJ_INFO_BRUSH       | 3  |
| Define OBJ_INFO_NPNTS       | 20 |
| Define OBJ_INFO_TEXTSTRING  | 3  |
| Define OBJ_INFO_SMOOTH      | 4  |
| Define OBJ_INFO_FRAMEWIN    | 4  |
| Define OBJ_INFO_NPOLYGONS   | 21 |
| Define OBJ_INFO_TEXTSPACING | 4  |
| Define OBJ_INFO_TEXTJUSTIFY | 5  |
| Define OBJ_INFO_FRAMETITLE  | 6  |
| Define OBJ_INFO_TEXTARROW   | 6  |
| Define OBJ_INFO_FILLFRAME   | 7  |
| Define OBJ_INFO_REGION      | 8  |
| Define OBJ_INFO_PLINE       | 9  |
| Define OBJ_INFO_MPOINT      | 10 |
| Define OBJ_INFO_NONEMPTY    | 11 |
| Define OBJ_INFO_Z_UNIT_SET  | 12 |
| Define OBJ_INFO_Z_UNIT      | 13 |
| Define OBJ_INFO_HAS_Z       | 14 |
| Define OBJ_INFO_HAS_M       | 15 |

```

'-----
' Object types, returned by ObjectInfo() for OBJ_INFO_TYPE
'-----

Define OBJ_TYPE_ARC                                1
Define OBJ_TYPE_ELLIPSE                            2
Define OBJ_TYPE_LINE                                3
Define OBJ_TYPE_PLINE                               4
Define OBJ_TYPE_POINT                               5
Define OBJ_TYPE_FRAME                               6
Define OBJ_TYPE_REGION                              7
Define OBJ_TYPE_RECT                                8
Define OBJ_TYPE_ROUNDRECT                           9
Define OBJ_TYPE_TEXT                                10
Define OBJ_TYPE_MPOINT                              11
Define OBJ_TYPE_COLLECTION                          12

'-----*
' RegionInfo() Defines
'-----*

Define REGION_INFO_IS_CLOCKWISE                    1

'=====
' PrismMapInfo() defines
'=====

Define PRISMMAP_INFO_SCALE                        1
Define PRISMMAP_INFO_BACKGROUND                    4
Define PRISMMAP_INFO_LIGHT_X                       6
Define PRISMMAP_INFO_LIGHT_Y                       7
Define PRISMMAP_INFO_LIGHT_Z                       8
Define PRISMMAP_INFO_LIGHT_COLOR                   9
Define PRISMMAP_INFO_CAMERA_X                     10
Define PRISMMAP_INFO_CAMERA_Y                     11
Define PRISMMAP_INFO_CAMERA_Z                     12
Define PRISMMAP_INFO_CAMERA_FOCAL_X               13
Define PRISMMAP_INFO_CAMERA_FOCAL_Y               14

```

```

Define PRISMMAP_INFO_CAMERA_FOCAL_Z          15
Define PRISMMAP_INFO_CAMERA_VU_1            16
Define PRISMMAP_INFO_CAMERA_VU_2            17
Define PRISMMAP_INFO_CAMERA_VU_3            18
Define PRISMMAP_INFO_CAMERA_VPN_1           19
Define PRISMMAP_INFO_CAMERA_VPN_2           20
Define PRISMMAP_INFO_CAMERA_VPN_3           21
Define PRISMMAP_INFO_CAMERA_CLIP_NEAR        22
Define PRISMMAP_INFO_CAMERA_CLIP_FAR         23
Define PRISMMAP_INFO_INFOTIP_EXPR            24

```

```
'=====
```

```
' SearchInfo() defines
```

```
'=====
```

```

Define SEARCH_INFO_TABLE          1
Define SEARCH_INFO_ROW            2

```

```
'=====
```

```
' SelectionInfo() defines
```

```
'=====
```

```

Define SEL_INFO_TABLENAME          1
Define SEL_INFO_SELNAME            2
Define SEL_INFO_NROWS              3

```

```
'=====
```

```
' Server statement and function defines
```

```
'=====
```

```
'-----
```

```
' Return Codes
```

```
'-----
```

```

Define SRV_SUCCESS                0
Define SRV_SUCCESS_WITH_INFO      1
Define SRV_ERROR                   -1
Define SRV_INVALID_HANDLE         -2
Define SRV_NEED_DATA               99
Define SRV_NO_MORE_DATA            100

```

```

'-----
' Special values for the status associated with a fetched value
'-----

Define SRV_NULL_DATA -1
Define SRV_TRUNCATED_DATA -2

'-----
' Server_ColumnInfo() defines
'-----

Define SRV_COL_INFO_NAME 1
Define SRV_COL_INFO_TYPE 2
Define SRV_COL_INFO_WIDTH 3
Define SRV_COL_INFO_PRECISION 4
Define SRV_COL_INFO_SCALE 5
Define SRV_COL_INFO_VALUE 6
Define SRV_COL_INFO_STATUS 7
Define SRV_COL_INFO_ALIAS 8

'-----
' Column types, returned by Server_ColumnInfo() for SRV_COL_INFO_TYPE
'-----

Define SRV_COL_TYPE_NONE 0
Define SRV_COL_TYPE_CHAR 1
Define SRV_COL_TYPE_DECIMAL 2
Define SRV_COL_TYPE_INTEGER 3
Define SRV_COL_TYPE_SMALLINT 4
Define SRV_COL_TYPE_DATE 5
Define SRV_COL_TYPE_LOGICAL 6
Define SRV_COL_TYPE_FLOAT 8
Define SRV_COL_TYPE_FIXED_LEN_STRING 16
Define SRV_COL_TYPE_BIN_STRING 17

'-----
' Server_DriverInfo() Attr defines
'-----

```

```

Define SRV_DRV_INFO_NAME          1
Define SRV_DRV_INFO_NAME_LIST     2
Define SRV_DRV_DATA_SOURCE        3

'-----
' Server_ConnectInfo() Attr defines
'-----

Define SRV_CONNECT_INFO_DRIVER_NAME 1
Define SRV_CONNECT_INFO_DB_NAME     2
Define SRV_CONNECT_INFO_SQL_USER_ID 3
Define SRV_CONNECT_INFO_DS_NAME     4
Define SRV_CONNECT_INFO_QUOTE_CHAR  5

'-----
' Fetch Directions (used by ServerFetch function in some code libraries)
'-----

Define SRV_FETCH_NEXT              -1
Define SRV_FETCH_PREV              -2
Define SRV_FETCH_FIRST              -3
Define SRV_FETCH_LAST              -4

'-----
'Oracle workspace manager
'-----

Define SRV_WM_HIST_NONE            0
Define SRV_WM_HIST_OVERWRITE       1
Define SRV_WM_HIST_NO_OVERWRITE    2

'=====
' SessionInfo() defines
'=====

Define SESSION_INFO_COORDSYS_CLAUSE 1
Define SESSION_INFO_DISTANCE_UNITS  2
Define SESSION_INFO_AREA_UNITS      3
Define SESSION_INFO_PAPER_UNITS     4

```

```
'=====
' Set Next Document Style defines
'=====

Define WIN_STYLE_STANDARD          0
Define WIN_STYLE_CHILD             1
Define WIN_STYLE_POPUP_FULLCAPTION 2
Define WIN_STYLE_POPUP             3

'=====
' StringCompare() defines
'=====

Define STR_LT                      -1
Define STR_GT                      1
Define STR_EQ                      0

'=====
' StyleAttr() defines
'=====

Define PEN_WIDTH                   1
Define PEN_PATTERN                 2
Define PEN_COLOR                   4
Define PEN_INDEX                   5
Define PEN_INTERLEAVED             6
Define BRUSH_PATTERN               1
Define BRUSH_FORECOLOR             2
Define BRUSH_BACKCOLOR             3
Define FONT_NAME                   1
Define FONT_STYLE                   2
Define FONT_POINTSIZE              3
Define FONT_FORECOLOR              4
Define FONT_BACKCOLOR              5
Define SYMBOL_CODE                 1
Define SYMBOL_COLOR                 2
Define SYMBOL_POINTSIZE            3
Define SYMBOL_ANGLE                 4
```

```

Define SYMBOL_FONT_NAME          5
Define SYMBOL_FONT_STYLE        6
Define SYMBOL_KIND              7
Define SYMBOL_CUSTOM_NAME       8
Define SYMBOL_CUSTOM_STYLE      9

```

```

'-----
' Symbol kinds returned by StyleAttr() for SYMBOL_KIND
'-----

```

```

Define SYMBOL_KIND_VECTOR      1
Define SYMBOL_KIND_FONT        2
Define SYMBOL_KIND_CUSTOM      3

```

```

'=====
' SystemInfo() defines
'=====

```

```

Define SYS_INFO_PLATFORM      1
Define SYS_INFO_APPVERSION    2
Define SYS_INFO_MIVERSION     3
Define SYS_INFO_RUNTIME       4
Define SYS_INFO_CHARSET       5
Define SYS_INFO_COPYPROTECTED 6
Define SYS_INFO_APPLICATIONWND 7
Define SYS_INFO_DDESTATUS     8
Define SYS_INFO_MAPINFOWND    9
Define SYS_INFO_NUMBER_FORMAT 10
Define SYS_INFO_DATE_FORMAT   11
Define SYS_INFO_DIG_INSTALLED 12
Define SYS_INFO_DIG_MODE      13
Define SYS_INFO_MIPLATFORM    14
Define SYS_INFO_MDICLIENTWND  15
Define SYS_INFO_PRODUCTLEVEL   16
Define SYS_INFO_APPIDISPATCH  17
Define SYS_INFO_MIBUILD_NUMBER 18

```

```

'-----

```

' Platform, returned by SystemInfo() for SYS\_INFO\_PLATFORM

'-----

|                         |   |
|-------------------------|---|
| Define PLATFORM_SPECIAL | 0 |
| Define PLATFORM_WIN     | 1 |
| Define PLATFORM_MAC     | 2 |
| Define PLATFORM_MOTIF   | 3 |
| Define PLATFORM_X11     | 4 |
| Define PLATFORM_XOL     | 5 |

'-----

' Version, returned by SystemInfo() for SYS\_INFO\_MIPLATFORM

'-----

|                            |   |
|----------------------------|---|
| Define MIPLATFORM_SPECIAL  | 0 |
| Define MIPLATFORM_WIN16    | 1 |
| Define MIPLATFORM_WIN32    | 2 |
| Define MIPLATFORM_POWERMAC | 3 |
| Define MIPLATFORM_MAC68K   | 4 |
| Define MIPLATFORM_HP       | 5 |
| Define MIPLATFORM_SUN      | 6 |

'=====

' TableInfo() defines

'=====

|                                |    |
|--------------------------------|----|
| Define TAB_INFO_NAME           | 1  |
| Define TAB_INFO_NUM            | 2  |
| Define TAB_INFO_TYPE           | 3  |
| Define TAB_INFO_NCOLS          | 4  |
| Define TAB_INFO_MAPPABLE       | 5  |
| Define TAB_INFO_READONLY       | 6  |
| Define TAB_INFO_TEMP           | 7  |
| Define TAB_INFO_NROWS          | 8  |
| Define TAB_INFO_EDITED         | 9  |
| Define TAB_INFO_FASTEDIT       | 10 |
| Define TAB_INFO_UNDO           | 11 |
| Define TAB_INFO_MAPPABLE_TABLE | 12 |
| Define TAB_INFO_USERMAP        | 13 |



---

|                                 |    |
|---------------------------------|----|
| Define TAB_INFO_USERBROWSE      | 14 |
| Define TAB_INFO_USERCLOSE       | 15 |
| Define TAB_INFO_USEREDITABLE    | 16 |
| Define TAB_INFO_USERREMOVEMAP   | 17 |
| Define TAB_INFO_USERDISPLAYMAP  | 18 |
| Define TAB_INFO_TABFILE         | 19 |
| Define TAB_INFO_MINX            | 20 |
| Define TAB_INFO_MINY            | 21 |
| Define TAB_INFO_MAXX            | 22 |
| Define TAB_INFO_MAXY            | 23 |
| Define TAB_INFO_SEAMLESS        | 24 |
| Define TAB_INFO_COORDSYS_MINX   | 25 |
| Define TAB_INFO_COORDSYS_MINY   | 26 |
| Define TAB_INFO_COORDSYS_MAXX   | 27 |
| Define TAB_INFO_COORDSYS_MAXY   | 28 |
| Define TAB_INFO_COORDSYS_CLAUSE | 29 |
| Define TAB_INFO_COORDSYS_NAME   | 30 |
| Define TAB_INFO_NREFS           | 31 |
| Define TAB_INFO_SUPPORT_MZ      | 32 |
| Define TAB_INFO_Z_UNIT_SET      | 33 |
| Define TAB_INFO_Z_UNIT          | 34 |
| Define TAB_INFO_BROWSER_LIST    | 35 |
| Define TAB_INFO_THEME_METADATA  | 36 |

```
'-----
' Table type defines, returned by TableInfo() for TAB_INFO_TYPE
'-----
```

|                        |   |
|------------------------|---|
| Define TAB_TYPE_BASE   | 1 |
| Define TAB_TYPE_RESULT | 2 |
| Define TAB_TYPE_VIEW   | 3 |
| Define TAB_TYPE_IMAGE  | 4 |
| Define TAB_TYPE_LINKED | 5 |
| Define TAB_TYPE_WMS    | 6 |
| Define TAB_TYPE_WFS    | 7 |
| Define TAB_TYPE_FME    | 8 |

```
'=====
' WindowInfo() defines
'=====

Define WIN_INFO_NAME                1
Define WIN_INFO_TYPE                3
Define WIN_INFO_WIDTH                4
Define WIN_INFO_HEIGHT              5
Define WIN_INFO_X                    6
Define WIN_INFO_Y                    7
Define WIN_INFO_TOPMOST              8
Define WIN_INFO_STATE                9
Define WIN_INFO_TABLE               10
Define WIN_INFO_LEGENDS_MAP         10
Define WIN_INFO_OPEN                11
Define WIN_INFO_WND                 12
Define WIN_INFO_WINDOWID            13
Define WIN_INFO_WORKSPACE            14
Define WIN_INFO_CLONEWINDOW         15
Define WIN_INFO_SYSMENUCLOSE        16
Define WIN_INFO_AUTOSCROLL          17
Define WIN_INFO_SMARTPAN             18
Define WIN_INFO_SNAPMODE            19
Define WIN_INFO_SNAPTHRESHOLD       20
Define WIN_INFO_PRINTER_NAME        21
Define WIN_INFO_PRINTER_ORIENT      22
Define WIN_INFO_PRINTER_COPIES      23
Define WIN_INFO_PRINTER_PAPERSIZE   24
Define WIN_INFO_PRINTER_LEFTMARGIN  25
Define WIN_INFO_PRINTER_RIGHTMARGIN 26
Define WIN_INFO_PRINTER_TOPMARGIN   27
Define WIN_INFO_PRINTER_BOTTOMMARGIN 28
Define WIN_INFO_PRINTER_BORDER      29
Define WIN_INFO_PRINTER_TRUECOLOR   30
Define WIN_INFO_PRINTER_DITHER     31
Define WIN_INFO_PRINTER_METHOD      32
Define WIN_INFO_PRINTER_TRANSPRSTER 33
```

```

Define WIN_INFO_PRINTER_TRANSPVECTOR      34
Define WIN_INFO_EXPORT_BORDER              35
Define WIN_INFO_EXPORT_TRUECOLOR           36
Define WIN_INFO_EXPORT_DITHER              37
Define WIN_INFO_EXPORT TRANSPRASTER        38
Define WIN_INFO_EXPORT_TRANSPVECTOR        39
Define WIN_INFO_PRINTER_SCALE_PATTERNS     40
Define WIN_INFO_EXPORT_ANTIALIASING         41
Define WIN_INFO_EXPORT_THRESHOLD            42
Define WIN_INFO_EXPORT_MASKSIZE            43
Define WIN_INFO_EXPORT_FILTER               44

```

```

'-----

```

```

' Window types, returned by WindowInfo() for WIN_INFO_TYPE

```

```

'-----

```

```

Define WIN_MAPPER                          1
Define WIN_BROWSER                         2
Define WIN_LAYOUT                          3
Define WIN_GRAPH                           4
Define WIN_BUTTONPAD                       19
Define WIN_TOOLBAR                         25
Define WIN_CART_LEGEND                     27
Define WIN_3DMAP                           28
Define WIN_HELP                            1001
Define WIN_MAPBASIC                        1002
Define WIN_MESSAGE                          1003
Define WIN_RULER                           1007
Define WIN_INFO                            1008
Define WIN_LEGEND                           1009
Define WIN_STATISTICS                       1010
Define WIN_MAPINFO                          1011

```

```

'-----

```

```

' Version 2 window types no longer used in version 3 or later versions

```

```

'-----

```

```

Define WIN_TOOLPICKER                      1004
Define WIN_PENPICKER                       1005

```

```
Define WIN_SYMBOLPICKER                                1006

'-----
' Window states, returned by WindowInfo() for WIN_INFO_STATE
'-----

Define WIN_STATE_NORMAL                                0
Define WIN_STATE_MINIMIZED                             1
Define WIN_STATE_MAXIMIZED                             2

'-----
' Print orientation, returned by WindowInfo() for WIN_INFO_PRINTER_ORIENT
'-----

Define WIN_PRINTER_PORTRAIT                            1
Define WIN_PRINTER_LANDSCAPE                           2

'-----
' Antialiasing filters, returned by WindowInfo() for WIN_INFO_EXPORT_FILTER
'-----

Define FILTER_VERTICALLY_AND_HORIZONTALLY              0
Define FILTER_ALL_DIRECTIONS_1                         1
Define FILTER_ALL_DIRECTIONS_2                         2
Define FILTER_DIAGONALLY                              3
Define FILTER_HORIZONTALLY                             4
Define FILTER_VERTICALLY                              5

'=====
' Abbreviated list of error codes
'
' The following are error codes described in the Reference manual. All
' other errors are listed in ERRORS.DOC.
'=====

Define ERR_BAD_WINDOW                                590
Define ERR_BAD_WINDOW_NUM                            648
Define ERR_CANT_INITIATE_LINK                         698
Define ERR_CMD_NOT_SUPPORTED                          642
Define ERR_FCN_ARG_RANGE                              644
```

```

Define ERR_FCN_INVALID_FMT                643
Define ERR_FCN_OBJ_FETCH_FAILED           650
Define ERR_FILEMGR_NOTOPEN                366
Define ERR_FP_MATH_LIB_DOMAIN             911
Define ERR_FP_MATH_LIB_RANGE              912
Define ERR_INVALID_CHANNEL                696
Define ERR_INVALID_READ_CONTROL            842
Define ERR_INVALID_TRIG_CONTROL            843
Define ERR_NO_FIELD                       319
Define ERR_NO_RESPONSE_FROM_APP           697
Define ERR_PROCESS_FAILED_IN_APP           699
Define ERR_NULL_SELECTION                 589
Define ERR_TABLE_NOT_FOUND                405
Define ERR_WANT_MAPPER_WIN                 313
Define ERR_CANT_ACCESS_FILE               825

'=====
' Backward Compatibility defines
'
' These defines are provided so that existing MapBasic code will continue
' to compile & run correctly. Please use the new define (on the right)
' when writing new code.
'=====
Define OBJ_ARC                           OBJ_TYPE_ARC
Define OBJ_ELLIPSE                       OBJ_TYPE_ELLIPSE
Define OBJ_LINE                           OBJ_TYPE_LINE
Define OBJ_PLINE                           OBJ_TYPE_PLINE
Define OBJ_POINT                           OBJ_TYPE_POINT
Define OBJ_FRAME                           OBJ_TYPE_FRAME
Define OBJ_REGION                           OBJ_TYPE_REGION
Define OBJ_RECT                           OBJ_TYPE_RECT
Define OBJ_ROUNDRECT                       OBJ_TYPE_ROUNDRECT
Define OBJ_TEXT                           OBJ_TYPE_TEXT

'=====
' end of MAPBASIC.DEF

```

' =====

# Указатель

---

## СИМВОЛЫ

- ! (exclamation point) in menus** 190
- & (амперсанд)**
  - dialog hotkeys 249
  - finding street intersections 290–293
  - шестнадцатеричные числа 698
  - menu hotkeys 190
  - конкатенация строк 766
- ( (open parenthesis) in menus** 190
- \* (звездочка)**
  - строка фиксированной длины 255
  - multiplication 766
- + (плюс)** 766
- / (slash)**
  - division 766
  - in menus 189, 191
- < (less than) character**
  - в меню 189
- Labelinfo( ) функция** 349
- \ (обратная косая черта)**
  - в меню 189
  - Деление нацело 766, 290
- ^ (caret)**
  - возведение в степень 766
  - show/hide menu text 190

## Numerics

- 3D Maps**
  - changing window settings 620–622
  - creating 186–187
  - prism maps 205–207
  - reading window settings 378–381

## A

- Abs( ) function** 60
- Absolute value**
  - Abs( ) function 60
- Ускоряющие клавиши**
  - в диалогах 249
  - в меню 190
- Access databases**
  - connection string attributes 540
- Acos( ) function** 60
- Оператор Add Cartographic Frame**

---

- Анимационный слой **70**
- buttons **72–76**
- columns to a table **64–68, 95–96**
- map layers **69–70**
- menu items **82–85**
- Узлы, добавление **92, 436, 453**
- Addresses, finding **290–293**
- Aggregate functions **528–529**
- Alias variables **255**
- All-caps text **295–297**
- Alter Button оператор **71**
- Оператор Alter Cartographic Frame
- Alter Control оператор **77**
- Alter MapInfoDialog statement **80**
- Alter Menu Bar оператор **85**
- Alter Menu Item оператор **87**
- Alter Menu оператор **82**
- Alter Object оператор **89, 207**
- Alter Table оператор **95**
- Слои анимации
  - adding **70**
  - removing **494**
- ApplicationDirectory\$( ) функция **96**
- Arc objects
  - создание **164**
  - determining length of **414**
  - modifying **89–92**
  - querying the pen style **410–414**
  - storing in a new row **337–339**
  - storing in an existing row **697**
- Единицы измерения **575**
- Area( ) функция **97**
- Area, spherical calculation **655**
- AreaOverlap( ) функция **98**
- Arithmetic functions. *See* Math functions
- Массив переменных
  - декларирование **256**
  - determining size of array **692**
  - resizing **480–481**
- Asc( ) function **99**
- Файлы формата ASCII.
  - exporting **277**
  - using as tables **482–488**
  - Смотрите File input/output
- Asin( ) function **100**
- Ask( ) функция **101**
- Assigning local storage
  - Server Bind Column **534–535**
- Atn( ) function **102**
- Файлы формата AutoCAD.
  - importing **331–335**
- AutoLabel оператор **103**
- Автоматическое преобразование типов **768**



**Automation**

- handling button event **167**
- handling menu event **191**

**Autoscroll feature**

- list of affected draw modes **74–75**
- reading current setting **707**
- turning on or off **641**
- WinChangedHandler **702**

**Avg() aggregate function 528–529****В****Background colors**

- Brush clause **107–108**
- Font clause **295–297**
- MakeBrush() функция **370**
- MakeFont() function **372–373**

**Столбчатые диаграммы**

- in graph windows **325–326**
- in thematic maps **653–654**

**Beep statement 104****Beginning a transaction**

- Server Begin Transaction **533**

**Двоичные файлы**

- closing files **128**
- opening files **447**
- reading data **318–319**
- writing data **475**

**BMP files, creating 513–515****Bold text 295–297****Boundaries. See Region objects****Bounding rectangle 387****Ветвление:**

- Do Case...End Case statement **260**
- If...Then statement **329**

**Точки останова (отладка) 664****Browse statement 104****Окна Списка**

- закрытие **130**
- determining the name of the table **707**
- modifying **576, 635–642**
- opening **104–106**
- restricting which columns appear **496**

**Brush clause 106–108****Brush styles**

- Brush clause defined **106–108**
- создание **370**
- modifying an object's style **90–91**
- querying an object's style **410–414**
- querying parts of **670–673**
- reading current style **224**
- setting current style **630–631**

**Brush variables 255**

---

**Элемент управления BrushPicker 153**

**Buffer regions**

Buffer( ) function 108

CartesianBuffer( ) функция 113

Create Object 199

Create Object statement 197

**Buffer( ) function 108**

**Элементы управления Button (в диалоге) 145**

**ButtonPadInfo( ) функция 109**

**ButtonPads**

adding/removing a button 72–76, 688–690

creating a new pad 165–168

docked vs. floating 73, 167

drawing modes 74–76

enabling/disabling a button 71

querying current settings 109

resetting to defaults 168

responding to user action 137

selecting/deselecting a button 71

setting which button is active 507

showing/hiding a pad 72–76

**Byte order in file i/o 448**

## C

**Call statement 110**

**Calling clause 167, 191**

**Callout lines**

map labels 617

text objects 221

**CancelButton clause 145**

**Элемент управления CancelButton 145**

**Capitalization**

lower case 358

mixed case 474

upper case 693

**CartesianArea( ) функция 112**

**CartesianBuffer( ) функция 113**

**CartesianDistance( ) функция 115**

**CartesianObjectDistance( ) функция 116**

**CartesianObjectLen( ) функция 116**

**CartesianOffset( ) функция 117**

**CartesianOffsetXY( ) функция 118**

**CartesianPerimeter( ) функция 119**

**Cartographic legends**

adding a frame 62–64

changing a frame 76

controlling settings 577

creating 169–172

removing a frame 493

**Case statement. See Do Case...End Case statement**

**Case, converting**

LCase\$( ) функция 358

- Proper\$( ) function **474**
  - UCase\$( ) функция **693**
- Centroid( ) function 120**
- Centroid, setting a region's 93**
- Centroids, displaying 617**
- CentroidX( ) функция 121**
- CentroidY( ) функция 122**
- Character codes**
  - character sets **123–125**
  - converting codes to strings **126**
  - converting strings to codes **99**
  - listing **763**
- CharSet clause 123**
- Checkable menu items, creating 189**
- Элемент управления CheckBox 145**
- Checking**
  - dialog box check boxes (custom) **77–79**
  - dialog box check boxes (standard) **80–82**
  - menu items **87–88**
- ChooseProjection\$( ) функция 126**
- Chr\$( ) function 126**
- Circle objects**
  - creating **172–174, 176**
  - determining area of **97**
  - determining perimeter of **461**
  - modifying **89–92**
  - querying the pen or brush style **410–414**
  - storing in a new row **337–339**
  - storing in an existing row **697**
- Cleaning objects 424**
- Clicking and dragging. Смотрите ButtonPads**
- Clipping a map 610**
- Cloning a map 507**
- Close All statement 127**
- Close Connection statement 128**
- Close File statement 128**
- Close Table statement 129**
- Close Window оператор 130, 235**
- Closing processing**
  - Server Close **535**
- Collection objects**
  - объединение **133**
  - создание **174**
  - resetting objects within collection **93–94**
- Color values, RGB 497**
- ColumnInfo( ) функция 131**
- Columns in a table**
  - adding **64–68, 95–96**
  - deleting **95–96**
  - determining column information **131–133, 404**
  - dynamic columns **68**
  - indexing **181**
- Combine( ) function 133**

---

- Combining objects**
  - Combine() function **133**
  - Create Object statement **197**
  - Objects Clean statement **424**
  - Objects Combine statement **425**
- CommandInfo() функция** **134**
- Commit Table оператор** **139**
- Comparing strings** **363, 667–668**
- Comparison operators** **766–767**
- Compiler directives**
  - Define оператор **243**
  - Include оператор **336**
- Сложение строк**
  - & оператор **766**
  - + оператор **766**
- Conditional execution**
  - Do Case...End Case statement **260**
  - If...Then statement **328–329**
- Conflict Resolution dialog box** **142**
- Connect option**
  - DLG=1 **539**
- Connect\_string**
  - определение **539**
- Connecting to a data source**
  - Server\_Connect **539–546**
- Connection number**
  - returning ??–**546**
- Connection number, returning** **539**
- ConnectObjects() функция** **143**
- Continue оператор** **144**
- Control BrushPicker clause** **153**
- Control Button clause** **145**
- Control CheckBox clause** **145**
- Control DocumentWindow clause** **147**
- Control EditText clause** **148**
- Control FontPicker clause** **153**
- Control GroupBox clause** **149**
- Control key**
  - detecting control-click **137**
  - entering line feeds in EditText boxes **148**
  - selecting multiple list items **150–152**
- Control ListBox clause** **150**
- Control MultiListBox clause** **150**
- Control panels**
  - date formatting **591–593**
  - number formatting **591–593**
- Control PenPicker clause** **153**
- Control PopupMenu clause** **154**
- Control RadioGroup clause** **155**
- Control StaticText clause** **156**
- Control SymbolPicker clause** **153**
- Элементы управления в диалогах** **131**
  - BrushPicker **153**

- Button **145**
- CancelButton **145**
- EditText **148**
- FontPicker **153**
- GroupBox **149**
- ListBox **150–152**
- MultiListBox **150**
- OKButton **145**
- PenPicker **153**
- RadioGroup **155–156**
- StaticText **156**
- SymbolPicker **153**
- Converting**
  - character codes to strings **126**
  - numbers to dates **402–403**
  - numbers to strings **665**
  - objects to polylines **157**
  - objects to regions **158**
  - strings to character codes **99**
  - strings to dates **669–670**
  - strings to numbers **698**
  - text to lower case **358**
  - text to mixed case **474**
  - text to upper case **693**
  - two-digit input into four-digit years **585**
- ConvertToPline( ) функция****157**
- ConvertToRegion( ) функция****158**
- Convex hull objects**
  - Create Object **199**
- ConvexHull( ) функция****158**
- Оператор CoordSys**
  - changing a table's CoordSys **139–143**
  - querying a table's coordinate system **682**
  - querying a window's coordinate system **383**
  - setting current MapBasic coordinate system **584**
  - specifying a coordinate system **159**
- CoordSysName( ) функция****163**
- Copying**
  - a projection
    - from a table **159–162**
    - from a window **159–162**
  - an object
    - offset by distance **660**
    - offset by XY values **661**
    - offset from source **434–435**
  - файлов **511**
  - object offset **441–442**
  - objects
    - offset by specified distance **117**
    - offset by XY values **118**
  - tables **139–143**
- Cos( ) function** **163**
- Cosmetic layer, accessing as a table** **707**

---

**Count( )** aggregate function **528–529**  
**Create Arc** statement **164**  
**Create ButtonPad** statement **165**  
**Create Cartographic Legend** оператор **169**  
**Create Collection** statement **174**  
**Create Cutter** statement **175**  
**Create Frame** оператор **177, 202**  
**Create Grid** statement **179**  
**Create Index** оператор **181**  
**Create Cartographic Legend** оператор **182**  
**Create Line** statement **184**  
**Create Map** оператор **185, 195**  
**Create Map3D** statement **186**  
**Create Menu Bar** оператор **193**  
**Create Menu** оператор **188**  
**Create Multipoint** statement **195**  
**Create Object** statement **197**  
**Create Pline** statement **202**  
**Create Point** statement **204**  
**Create PrismMap** statement **205**  
**Create Ranges** statement **207**  
**Create Rect** statement **210**  
**Create Redistricter** statement **211**  
**Create Region** statement **212**  
**Оператор Create Report From Table**  
**Create RoundRect** statement **215**  
**Create Styles** statement **216**  
**Create Table** statement **218**  
**Create Text** оператор **221, 198**  
**CreateCircle( )** функция **172**  
**CreateLine( )** функция **183**  
**CreatePoint( )** функция **203**  
**CreateText( )** функция **220**  
**Cross-reference.** *See* cross-reference in online Help  
**Crystal Reports**  
    создание **214**  
    loading **448**  
**CurDate( )** функция **222**  
**CurrDateTime( )** функция **223**  
**CurrentBorder Pen( )** **223**  
**CurrentBrush( )** функция **224**  
**CurrentFont( )** функция **225**  
**CurrentLinePen( )** функция **225**  
**CurrentPen( )** функция **226**  
**CurrentSymbol( )** функция **227**  
**Cursor** coordinates, displaying **611**  
**Cursor** shapes **74**  
**Курсор (позиционирование в таблице)** **160**  
    end-of-table condition **268**  
    positioning the row cursor **283–285**  
**CurTime( )** функция **227**  
**Custom** symbols  
    Reload Symbols statement **489**

syntax **676–679**

**Cutter objects, creating** **175**

## **D**

### **Data aggregation**

combining objects **424–426**

filling a column with data from another table **66–68**

grouping rows **528–529**

### **Data disaggregation**

удаление части объекта **208**

splitting objects **439–441**

### **Структура данных** **691**

**Databases, using as tables** **482–488**

### **Date functions**

converting numbers to dates **402**

converting strings to dates **591–593, 669–670**

current date **222**

date window setting **229**

extracting day-of-month **229**

extracting day-of-week **699**

extracting the month **397**

extracting the year **713**

formatting based on locale **591–593**

**Date variables** **255**

**DateWindow()** функция **229**

**Day()** function **229**

**DBF files, exporting** **277**

### **DDE, acting as client**

closing a conversation **238**

executing a command **230**

initiating a conversation **231–234**

reading data from the server **235–237**

sending data to the server **234**

### **DDE, acting as server**

handling execute event **491**

handling peek request **492**

retrieving execute string **136**

**DDEExecute оператор** **230**

**DDEInitiate function** **231**

**DDEPoke оператор** **234**

**DDERequest\$()** функция **235**

**DDETerminate оператор** **238**

**DDETerminateAll оператор** **238**

### **Debugging**

Continue оператор **144**

Stop оператор **664**

**Decimal separators** **244, 303, 591–593**

### **Decision-making**

Do Case...End Case statement **260**

If...Then statement **328–329**

**Declare Function оператор** **239, 222**

**Declare Sub оператор** **241, 222**

---

**Define оператор** 243

**Definitions file** 771

**DeformatNumber\$( ) функция** 244

**Delaying when user drags mouse** 590

**Delete statement** 245

**Удаление**

all objects from a table 264

колонок из таблицы 166

файлов 345

узлов в объекте 89

rows or objects 245

tables 265

**Диалоги, созданные пользователем**

accelerator keys 249

creating 246–252

determining ID of a control 690

determining if user clicked OK 135

determining if user double-clicked 135

модальный по сравнению с немодальным 247

modifying 77–79

preserving after user clicks OK 252

reading user's input 248, 477–480

sizes of dialog boxes and controls 248

tab order 249

terminating 248, 253

**Диалоги, стандартные**

altering MapInfo dialog boxes 80–82

запрос ОК/Отмена 101

opening a file 287–288

percent complete 471–473

сохранение файла 289

simple messages 401

suppressing progress bars 626

**Dialog Preserve statement** 252

**Dialog Remove statement** 253

**Dialog statement** 246

**Digitizer setup** 586–588

**Digitizer status** 680

**Dim оператор** 254

**Directory names**

extracting from a file name 455

user's home directory 327

user's Windows directory 327

where application is installed 96

where MapInfo is installed 471

**Disabling**

ButtonPad buttons 71

dialog box controls (custom) 77–79

dialog box controls (standard) 80–82

процедуры-обработки 598

menu items 87–88

Подавление изображения индикатора выполнения 626

shortcut menus 193



- system menu's Close command **640**
- Discarding changes**
  - to a local table **500**
  - to a remote server **566**
- Distance**
  - spherical calculation **657**
  - Единицы измерения **588**
- Distance( ) function** **259**
- DLG=1 connect option** **539**
- Библиотеки DLL**
  - declaring as functions **240–241**
  - declaring as procedures **242**
- Do Case...End Case statement** **260**
- Do...Loop statement** **262**
- Dockable ButtonPads**
  - docking after creation **73**
  - docking at creation **167**
  - querying current status **109**
- Document conventions** **20, 59**
- DOS commands, executing** **509**
- Dot density thematic maps** **651**
- Double byte character sets (DBCS)**
  - extracting part of a DBCS string **396**
- Double-clicking in dialog boxes** **135, 151**
- Dragging with the mouse**
  - time threshold **590**
  - turning off autoscroll **641**
- Drawing modes** **74–75**
- Drawing objects. See Objects, creating**
- Drawing tools, custom** **72–76**
- Drop Index statement** **263**
- Drop Map оператор** **264**
- Drop Table statement** **265**
- Duplicating a map** **507**
- Файлы формата DXF.**
  - exporting **277**
  - importing **331–335**
- Dynamic columns** **68**
- Dynamic Link Libraries. Смотрите DLLs**
- Е-, е-**
- Editable map layers** **384, 612**
- Editing an object. See specific object type**
  - Arc, Ellipse, Frame, Line, Point, Polyline, Rectangle, Region, Rounded Rectangle, Text
- Редактирование**
  - determining if there are unsaved edits **681–684**
  - discarding **500**
  - saving **139–143**
- Элемент управления EditText** **148**
- Elapsed time** **687**
- Ellipse objects**
  - Cartesian area of **112**

---

- Cartesian perimeter of **119**
- creating **172–174, 176**
- determining area of **97**
- determining perimeter of **461**
- modifying **89–92**
- querying pen or brush style **410–414**
- storing in a new row **337–339**
- storing in an existing row **697**
- Enabling**
  - ButtonPad buttons **71**
  - dialog box controls **77–79**
  - menu items **87–88**
- End MapInfo statement 266**
- End Program оператор 266**
- EndHandler процедура 267**
- Enlarging arrays 480–481**
- EOF() функция 268**
- EOT() функция 268**
- EPSGToCoordSysString\$() функция 269**
- Erase() function 270**
- Erasing**
  - entire objects **245**
  - файлов **345**
  - части объекта **270**
  - tables **265**
- Err() функция 271**
- Обработка ошибок**
  - determining error code **271**
  - determining error message **272**
  - enabling an error handler **443–444**
  - generating an error **272**
  - returning from an error handler **496**
- Error statement 272**
- Error\$() функция 272**
- Escape key**
  - cancelling draw operations **73–76**
  - в диалоге **135**
  - interrupting selection **523**
- События, обработка**
  - application terminated **267**
  - Automation method used **490**
  - execute string received **136, 491**
  - параметры окна карты (136) **702**
  - MapInfo got or lost focus **136, 299**
  - peek request received **492**
  - изменение выборки **135**
  - user clicked with custom tool **137, 688–690**
  - user double-clicked in a dialog box **135**
  - window closed **136, 703**
  - window focus changed **711**
  - See also* Error handling
- Excel files, opening 482–488**
- Executing**

- interpreted strings **505–507**
  - menu commands **507**
  - Run Application оператор **504**
  - Run Program statement **508**
  - Executing an SQL string**
    - Server\_Execute( ) **556**
  - Execution speed**
    - Анимационный слой **70**
    - screen updates **590**
    - table editing **632–634**
  - Exit Do statement 273**
  - Exit For statement 274**
  - Exit Function statement 274**
  - Exit Sub statement 275**
  - Exiting from MapInfo 266**
  - Exp( ) function 276**
  - Expanded text 295–297**
  - возведение в степень 276**
  - Export оператор 277**
  - Extents of entire table 682**
  - External functions 240–241**
  - Выделение части строки:**
    - Left( ) function **359**
    - Mid( ) function **395**
    - MidByte( ) функция **396**
    - Right( ) function **498**
  - ExtractNodes( ) функция 280**
- F**
- Farthest statement 281**
  - Fetch statement 283**
  - Fields. See Columns**
  - Файл, ввод/вывод**
    - closing a file **128**
    - determining if file exists **286**
    - end-of-file condition **268**
    - file attributes, reading **285**
    - length of file **367**
    - opening a file **446–448**
    - reading current position **521**
    - reading data in binary mode **318–319**
    - reading data in random mode **318–319**
    - reading data in sequential mode **336, 364**
    - setting current position **521**
    - writing data in binary mode **475**
    - writing data in random mode **475**
    - writing data in sequential mode **465, 713**
  - File names**
    - determining full file spec **691**
    - determining temporary name **685**
    - extracting directory from **455**
    - extracting from full file spec **455**

---

**File sharing conflicts** 591

**FileAttr( )** функция 285

**FileExists( )** функция 286

**FileOpenDlg( )** функция 287

**Files**

- copying 511
- deleting 345
- determining if file exists 286
- importing 331–335
- length 367
- locating 365
- renaming 495

**FileSaveAsDlg( )** функция 289

**Fill styles.** *See* Brush styles

**Filtering data** 525–527

**Find statement** 290

**Find Using statement** 293

**Finding**

- a substring within a string 339
- an address in a map 290–293
- an intersection of two streets 290–293
- objects from map coordinates 515–520

**Fix( )** function 294

**строка фиксированной длины** 255

**Floating point variables** 255

**Flow control**

- exiting a Do loop 273
- exiting a function 274
- exiting a procedure 275
- exiting an application 266
- exiting MapInfo 266
- halting another application 686
- unconditional jump 324

**в диалоге** 78

**Focus, active window changes** 711

**Focus, getting or losing** 136, 299

**Folder names.** *See* Directory names

**Font clause** 295–297

**Font styles**

- создание 373
- Font clause defined 295–297
- modifying an object's style 90–91
- querying an object's style 410–414
- querying parts of 670–673
- reading current style 225
- setting current style 630–631

**Font variables** 255

**Элемент управления FontPicker** 153

**For...Next statement** 297

**ForegroundTaskSwitchHandler** процедура 299

**Foreign character sets** 123–125

**FormatDate\$( )** функция 302

**FormatNumber\$( )** функция 303

**FormatTime() функция** 304

**Frame objects**

creating 177–179

inserting into a layout 337–339

modifying 89–92, 697

querying the pen or brush style 410–414

**Frames, cartographic legend**

adding a frame 62–64

controlling settings 577

creating 169–172

изменение 76

removing 493

**FrontWindow() функция** 304, 306

**FTP Library** 715

**Function...End Function statement** 307

**Functions, creating**

Declare Function оператор 239, 222

Exit Function statement 274

Function....End Function statement 307

## G

**Gaps**

checking in regions 422–424

cleaning 424

snapping nodes 437–439

**Geocode statement** 310

**GeocodeInfo() function** 314

**Geographic calculations**

area of object 97

area of overlap 98

distance 259

Длина объекта 414

perimeter of object 461

**Geographic calculations. See Objects, querying**

**Географические объекты** Смотрите Объекты

**Географические операторы** 530, 767

**Get statement** 318

**GetDate() функция** 319

**GetFolderPath() функция** 320

**GetMetadata() функция** 321

**GetSeamlessTable() функция** 322

**GetTime() функция** 323

**Global statement** 323

**GML files, importing** 331–335

**GoTo оператор** 324

**Приложения GPS** 70

**Карта градуированных символов** 651

**Graph statement** 325

**Окна Графиков**

закрытие 130

determining the name of the table 707

modifying 593–597, 635–642

---

- opening **319–320, 325–326**
- Great circle distance 259**
- Grid surfaces**
  - in thematic maps **179–181**
  - modifying **604–620**
- Grid tables**
  - adding relief shade information **489**
- Group By clause 528–530**
- Элемент управления GroupBox 149**

## **Н**

- Halo text 295–297**
- Halting another application 686**
- Handlers**
  - assigning to menu items **189**
- Hardware platform, determining 679–681**
- Help messages**
  - button tooltips **73, 166**
  - Подсказки в строке сообщений **73**
- Help window**
  - закрытие **130**
  - modifying **635–642**
  - opening **451, 635–642**
- Шестнадцатеричные числа 698**
- Hiding**
  - ButtonPads **72–76**
  - dialog box controls (custom) **77–79**
  - dialog box controls (standard) **80–82**
  - menu bar **388**
  - Подавление изображения индикатора выполнения **626**
  - screen activity **590**
- Hierarchical menus**
  - Alter Menu оператор **84**
  - Create Menu оператор **189**
- HomeDirectory\$( ) функция 327**
- HotLink tool**
  - querying object attributes **138**
  - Set Map clause **612**
- HotLinkInfo() функция 327**
- HotLinks, querying 352**
- Hour function 328**
- HTTP and FTP Library**
  - MICloseContent процедура **716**
  - MICloseFtpConnection процедура **716**
  - MICloseFtpFileFind процедура **716**
  - MICloseHttpConnection процедура **717**
  - MICloseHttpFile процедура **717**
  - MICloseSession процедура **718**
  - MICreateSession() function **718**
  - MICreateSessionFull() function **719**
  - MIErrorDlg() function **720**
  - MIFindFtpFile() function **722**

MIFindNextFtpFile() function **722**  
 MIGetContent() function **723**  
 MIGetContentBuffer() function **723**  
 MIGetContentLen() function **724**  
 MIGetContentString() function **725**  
 MIGetContentToFile() function **725**  
 MIGetContentType() function **726**  
 MIGetCurrentFtpDirectory() function **726**  
 MIGetErrorCode() function **727**  
 MIGetErrorMessage() function **728**  
 MIGetFileURL() function **728**  
 MIGetFtpConnection() function **729**  
 MIGetFtpFile() function **730**  
 MIGetFtpFileFind() function **732**  
 MIGetFtpFileName процедура **732**  
 MIGetHttpConnection() function **733**  
 MIIsFtpDirectory() function **733**  
 MIIsFtpDots() function **734**  
 MIOpenRequest() function **734**  
 MIOpenRequestFull() function **735**  
 MIParseURL() function **737**  
 MIPutFtpFile() function **738**  
 MIQueryInfo() function **739**  
 MIQueryInfoStatusCode() function **740**  
 MISaveContent() function **741**  
 MISendRequest() function **742**  
 MISendSimpleRequest() function **743**  
 MISetCurrentFtpDirectory() function **743**  
 MISetSessionTimeout() function **744**

#### **HWND values, querying**

SystemInfo( ) функция **679**  
 WindowInfo( ) функция **708**

### **Следующая таблица описывает форматы**

#### **Iconizing MapInfo**

Close Window оператор **635**, 235  
 suppressing progress bars **626**

#### **Пиктограммы для кнопок 74, 227**

#### **Идентификаторы, определение 243**

#### **If...Then statement 328–329**

#### **Import statement 331**

#### **Include оператор 336**

#### **Indexed columns**

creating an index **181**  
 deleting an index **263**

#### **Individual value thematic maps 650**

#### **Infinite loops, avoiding 598**

#### **Info tool**

closing Info window **130**  
 modifying Info window **635–642**  
 opening Info window **451**  
 setting to read-only **641**

---

- setting which data displays **641**
- Informix databases**
  - connection string attributes **545–546**
- Initializing variables 258**
- Input # оператор 336**
- Ввод/вывод Смотрите File input/output**
- Insert statement 337**
- Вставка**
  - columns in a table **64–68, 95–96**
  - узлов в объекте **92**
  - Строки в таблице
- InStr( ) функция 339**
- Int( ) function 340**
- Деление нацело 766, 290**
- Integer variables 255**
- Интегрированная Картография**
  - managing legends **182**
  - reparenting dialog boxes **574**
  - reparenting document windows **622–623**
- International character sets 123–125**
- International formatting 591–593**
- Interpreting strings as commands 505–507**
- Interrupting the selection 523**
- Intersection of objects**
  - Create Object statement **197**
  - Intersects оператор **530**
  - Objects Intersect statement **432**
  - Overlap( ) function **452**
- Intersection of two streets, finding 290–293**
- IntersectNodes( ) функция 341**
- IsogramInfo( ) функция 341**
- IsPenWidthPixels( ) функция 344**
- Italic text 295–297**

## **J**

- Joining tables 525–527**
- JPEG files, creating 513–515**

## **K**

- Keys, metadata 391–393**
- Keywords 257, 527**
- Kill оператор 345**

## **L**

- LabelFindByID( ) функция 346**
- LabelFindFirst( ) функция 347**
- LabelFindNext( ) функция 348**
- Подписи**
  - в диалогах **156**
  - в программе **324**
  - on maps **103, 346–351, 617–619**
  - reading label expressions **354**



**Launching other applications**Run Application оператор **504**Run Program statement **508****LayerInfo( ) функция****352****Слон**adding **69–70**Косметика **707**modifying settings **604–620**reading settings **352–357**removing **494**thematic maps **629, 645–654****Layout statement** **357****Окно Отчёта**accessing as tables **707**закрытие **130**creating frames **177–179**modifying **599–601, 635–642**открытие **357**specifying layout coordinates **159–162, 584****LCase\$( ) функция****358****Left\$( ) function** **359****Legend frames**querying attributes **360–361**querying styles **362****Legend window**закрытие **130**modifying **601–603, 635–642**opening **182, 451**запросы **361****Legend, cartographic**adding a frame **62–64**controlling settings **577**creating **169–172**modifying a frame **76**removing a frame **493****LegendFrameInfo( ) function** **360–361****LegendInfo( ) функция****361****LegendStyleInfo( ) функция****362****Len( ) function** **363****Length**of a file **367**Длина объекта **414**spherical calculation of **659****Like( ) function** **363****Line feed character**Chr\$(10) function **126**used in EditText controls **148**used in text objects **221****Line Input # оператор** **364****Линейные объекты**Cartesian length of **116**creating **183–184**determining length of **414**

---

- modifying **89–92**
- querying the pen style **410–414**
- storing in a new row **337–339**
- storing in an existing row **697**

**Line styles.** *See* Pen styles

**Linked tables**

- creating **561–563**
- determining if table is linked **683**
- refreshing **565**
- saving **142**
- unlinking **696**

**ListBox controls** **150–152**

**Locale settings** **244, 303, 591–593**

**LocateFile\$()** **365**

**LOF()** функция **367**

**Log()** function **367**

**Логические операторы** **767**

**Logical variables** **255**

**Циклы**

- Do...Loop statement **262**
- For...Next statement **297**
- While...Wend statement **700**

**Lotus files, opening** **482–488**

**Lower case, converting to** **358**

**LTrim\$()** функция **368**

## **M**

**Main procedure** **369–370**

**MakeBrush()** функция **370**

**MakeCustomSymbol()** функция **371**

**MakeDateTime()** функция **372**

**MakeFont()** function **372–373**

**MakeFontSymbol()** функция **374**

**MakePen()** функция **375**

**MakeSymbol()** функция **376**

**Map layers.** Смотрите Слои

**Объекты карты** Смотрите Объекты

**Map projections**

- changing a table's projection **139–143**
- copying from a table or window **159–162**
- querying a table's CoordSys **682**
- querying a window's CoordSys **383**
- setting the current MapBasic CoordSys **584**

**Map scale**

- determining in Map windows **385**
- displaying **609**

**Map statement** **377**

**Окна Карт**

- adding map layers **69–70**
- clipping **610**
- закрытие **130**
- controlling redrawing **69, 590, 611**

- creating thematic layers 645–654
- duplicating 507
- handling window-changed event 136, 702
- подписывание 617
- modifying 604–620, 635–642
- modifying thematic layers 629
- opening 377–378
- prism 205–207
- reading layer settings 352–357
- reading window settings 382–386
- removing map layers 494

**Map3DInfo() функция 378****mapbasic**

- Definitions file 771
- language overview 20–59

**MapInfo 3.0 symbols 676–679****Файл MAPINFOW.AVB 292****MapperInfo() функция 382****Математические функции**

- absolute value 60
- arc-cosine 60
- arc-sine 100
- arc-tangent 102
- area of object 97
- area of overlap 98
- converting strings to numbers 698
- cosine 163
- distance 259
- возведение в степень 276
- logarithms 367
- maximum value 386
- minimum value 396
- rounding off a number 294, 340, 503
- sign 643
- sine 654
- square root 663
- tangent 684

**Max() aggregate function 528–529****Maximum() function 386****MBR() функция 387****Memo fields 95, 139, 141****Create Menu Bar оператор 388****Menu commands, executing 507****MenuItemInfoByHandler() функция 388****MenuItemInfoByID() функция 390****Меню, настройка**

- adding hierarchical menus 84
- adding menu items 82–85
- altering menu items 87–88
- creating checkable menu items 189
- creating new menus 188–193
- disabling shortcut menus 193
- querying menu item status 388–390

---

- redefining the menu bar **85–87, 193–195**
- removing menu items **82–85**
- showing/hiding the menu bar **388**

**Merging objects.** *See* **Combining objects**

**Messages**

- displaying in a Note dialog box **401**
- displaying on the status bar **664**
- Очистка Окна Сообщения **451**
- printing to the Message window **464**

**Метаданные**

- code example **393**
- keys **391–393**
- managing in tables **390–393**
- reading keys **321**
- statement **390**

**Metric units**

- area **575**
- distance **588**

**MGRSToPoint() функция** **393**

**MICloseContent() procedure** **716**

**MICloseFtpConnection() procedure** **716**

**MICloseFtpFileFind() procedure** **716**

**MICloseHttpConnection() procedure** **717**

**MICloseHttpFile() procedure** **717**

**MICloseSession() procedure** **718**

**MICreateSession() функция** **718**

**MICreateSessionFull() функция** **719**

**Microsoft Access databases**

- connection string attributes **540**

**Mid\$( ) function** **395**

**MidByte\$( ) функция** **396**

**MIErrorDlg() функция** **720**

**Файлы формата MIF.**

- exporting **277**
- importing **331–335**

**MIFindFtpFile() функция** **722**

**MIFindNextFtpFile() функция** **722**

**MIGetContent() функция** **723**

**MIGetContentBuffer() функция** **723**

**MIGetContentLen() функция** **724**

**MIGetContentString() функция** **725**

**MIGetContentToFile() функция** **725**

**MIGetContentType() функция** **726**

**MIGetCurrentFtpDirectory() функция** **726**

**MIGetErrorCode() функция** **727**

**MIGetErrorMessage() функция** **728**

**MIGetFileURL() функция** **728**

**MIGetFtpConnection() функция** **729**

**MIGetFtpFile() функция** **730**

**MIGetFtpFileFind() функция** **732**

**MIGetFtpFileName() procedure** **732**

**MIGetHttpConnection() функция** **733**

**MIIsFtpDirectory() функция** **733**

- MIIsFtpDots( ) функция** 734
- Military grid reference format** 385, 611
- Min( ) aggregate function** 528–529
- Minimizing MapInfo**
  - Close Window оператор 635, 235
  - suppressing progress bars 626
- Minimum bounding rectangle**
  - Длина объекта 387
  - of entire table 682
- Minimum( ) function** 396
- Minute function** 397
- MIOpenRequest( ) функция** 734
- MIOpenRequestFull( ) функция** 735
- MIParseURL( ) функция** 737
- MIPutFtpFile( ) функция** 738
- MIQueryInfo( ) функция** 739
- MIQueryInfoStatusCode( ) функция** 740
- MISaveContent( ) функция** 741
- MISendRequest( ) функция** 742
- MISendSimpleRequest( ) функция** 743
- MISetCurrentFtpDirectory( ) функция** 743
- MISetSessionTimeout( ) функция** 744
- Mixed case, converting to** 474
- MIXmlAttributeListDestroy( ) procedure** 746
- MIXmlDocumentCreate( ) функция** 746
- MIXmlDocumentDestroy( ) procedure** 747
- MIXmlDocumentGetNamespaces( ) функция** 747
- MIXmlDocumentGetRootNode( ) функция** 748
- MIXmlDocumentLoad( ) функция** 748
- MIXmlDocumentLoadXML( ) функция** 749
- MIXmlDocumentLoadXMLString( ) функция** 750
- MIXmlDocumentSetProperty( ) функция** 751
- MIXmlGetAttributeList( ) функция** 752
- MIXmlGetChildList( ) функция** 752
- MIXmlGetNextAttribute( ) функция** 753
- MIXmlGetNextNode( ) функция** 754
- MIXmlNodeDestroy( ) procedure** 754
- MIXmlNodeGetAttributeValue( ) функция** 755
- MIXmlNodeGetFirstChild( ) функция** 755
- MIXmlNodeGetName( ) функция** 756
- MIXmlNodeGetParent( ) функция** 757
- MIXmlNodeGetText( ) функция** 757
- MIXmlNodeGetValue( ) функция** 758
- MIXmlNodeListDestroy( ) procedure** 758
- MIXmlSCDestroy( ) procedure** 759
- MIXmlSCGetLength( ) функция** 759
- MIXmlSCGetNamespace( ) функция** 760
- MIXmlSelectNodes( ) функция** 760
- MIXmlSelectSingleNode( ) функция** 761
- Mod оператор** 766
- Модальный диалог** 247
- Modifying an object.** *See specific object type*
  - Arc, Ellipse, Frame, Line, Point, Polyline, Rectangle, Region, Rounded Rectangle, Text

---

**Month( ) function** 397  
**Most-recently-used list (File menu)** 190  
**Mouse actions** 590  
**Mouse cursor**  
    customizing shape of 74  
    displaying coordinates of 611  
**Moving an object** 433–434  
**MRU list (File menu)** 190  
**Элемент управления MultiListBox** 150  
**Multipoint objects**  
    объединение 133  
    creating 195–196  
    inserting nodes 93–94

## N

**Natural Break thematic ranges** 207  
**Nearest statement** 398  
**Network file sharing** 591  
**Узлы**  
    Узлы, добавление 92, 436, 453  
    displaying 617  
    extracting a range of nodes from an object 280  
    maximum number per object 202, 213  
    querying number of nodes 411  
    querying x/y coordinates 419–420  
    removing 92  
**NoSelect ключевое слово** 527  
**Note оператор** 401  
**Null handling** 558  
**NumAllWindows( ) функция** 402  
**Number of characters in a string** 363  
**NumberToDate( ) function** 402–403  
**NumberToDateTime() функция** 403  
**NumCols( ) функция** 404  
**Арифметические операторы** 766  
**NumTables( ) функция** 405  
**NumWindows( ) функция** 406

## O

**Переменные Object** 255  
**ObjectGeography( ) функция** 407  
**ObjectInfo( ) функция** 410  
**ObjectLen( ) функция** 414  
**ObjectNodeHasM( ) функция** 415  
**ObjectNodeHasZ( ) функция** 416  
**ObjectNodeM( ) функция** 418  
**ObjectNodeX( ) функция** 419  
**ObjectNodeY( ) функция** 420  
**ObjectNodeZ( ) функция** 421  
**Objects Check statement** 422  
**Objects Clean statement** 424  
**Objects Combine statement** 425

- Objects Disaggregate statement** 427
- Objects Enclose statement** 429
- Objects Erase statement** 430
- Objects Intersect statement** 432
- Objects Move statement** 433
- Objects Offset statement** 434
- Objects Overlay statement** 436
- Objects Pline oneпароп** 436
- Objects Snap statement** 437
- Objects Split statement** 439
- Objects, copying**
  - offset by distance 441–442
  - to offset location 434–435
- Объекты, создание**
  - arcs 164
  - buffer regions 199
  - by buffering 108, 197–201
  - by combining objects 133
  - by intersecting objects 452
  - circles 172–174, 176
  - convex hull 199
  - ellipses 172–174, 176
  - frames 177–179
  - lines 183–184
  - map labels 103
  - multipoint 195–196
  - points 203–204
  - polylines 202
  - rectangles 210
  - regions 212–214
  - rounded rectangles 215
  - text 220–221
  - Voronoi polygons 199
- Объекты, редактирование**
  - добавление узлов 92, 436
  - combining 133, 425–426
  - converting to polylines 157
  - converting to regions 158
  - erasing entire object 245
  - удаление части объекта 270
  - moving nodes 89–94
  - removing nodes 92
  - resolution of converted objects 629
  - rotating 501
  - rotating around specified point 502
  - setting the target object 634
  - snap setting 437–439
  - splitting 439–441
- Objects, moving**
  - within input table 433–434
- Объекты, запросы**
  - area 97
  - boundary gaps 422

---

- boundary overlap **422**
- centroid **120–122**
- content of a text object **411**
- coordinates **407, 419–420**
- HotLink support **138**
- length **414**
- minimum bounding rectangle **387**
- number of nodes **411**
- number of polygons in a region **411**
- number of sections in a polyline **411**
- overlap, area of **98**
- overlap, proportion of **475**
- perimeter **461**
- Точки пересечения **341**
- styles **410–414**
- type of object **411–414**
- ODBC connection 142**
- ODBC таблицы**
  - changing object styles in mappable tables **567**
- Offset() function 441–442**
- OffsetXY() функция 442**
- OKButton clause 145**
- Элемент управления OKButton 145**
- OLE Automation**
  - handling button event **167**
  - handling menu event **191**
- OnError оператор 443**
- Open Connection statement 445**
- Open File оператор 446**
- Open Report statement 448**
- Open Table statement 449**
- Open Window оператор 451, 235**
- Opening windows**
  - Browse statement **104–106**
  - Create Redistricter statement **211**
  - Graph statement **319–320, 325**
  - Layout statement **357**
  - Map statement **377**
  - Open Window оператор **451, 235**
- Operating environment, determining 679–681**
- Операторы**
  - Автоматическое преобразование типов **768**
  - summary of **765–769**
- Оптимизация производительности**
  - Анимационный слой **70**
  - screen updates **590**
  - table editing **632–634**
- Oracle databases**
  - connection string attributes **540–541**
- Oracle8i databases**
  - connection string attributes **542**
- Предложение Order By**
  - sorting rows **530**



**Overlap( ) function 452**

**Overlaps**

checking in regions 422–424

cleaning 424

snapping nodes 437–439

**OverlayNodes( ) функция 453**

**P**

**Pack Table оператор 453**

**Page layout, opening 357**

**Единицы измерения 624**

**Papersize attribute 708**

**Parallel labels 618**

**Parent windows**

reparenting dialog boxes 574

reparenting document windows 622–623

**Partalsegments option 618**

**PathToDirectory\$( ) функция 455**

**PathToFileName\$( ) функция 455**

**PathToTableName\$( ) функция 456**

**Сравнение по шаблону 363**

**Peek requests 492**

**Pen clause 457**

**Pen styles**

создание 375

modifying an object's style 90–91

Pen clause defined 457

querying an object's style 410–414

querying parts of 670–673

reading current border style 223

reading current line style 225

reading current style 226

setting current style 630–631

**Pen variables 255**

**Выбор стиля линии (PenPicker) 153**

**PenWidthToPoints( ) функция 460**

**Диалог "Минуточку..." 128**

**Performance, improving**

Анимационный слой 70

screen updates 590

table editing 632–634

**Perimeter( ) функция 461**

**Per-object styles 549**

**Файлы формата PICT.**

creating 513–515

importing 331–335

**Круговые диаграммы**

in graph windows 319–320, 325–326

in thematic maps 652–653

**Platform, determining 679–681**

**Pline. See Polyline**

**PNG files, creating 513–515**

---

## **Точечные объекты**

- creating **203–204**
- modifying **89–92**
- querying the symbol style **410–414**
- storing in a new row **337–339**
- storing in an existing row **697**

**Point styles.** *See* **Symbol styles**

**PointsToPenWidth( )** функция **462**

**PointToMGRSS( )** функция **463**

**Polygon draw mode** **75**

**Polygons.** *See* **Region objects**

## **Линейные объекты**

- adding/removing nodes **92**
- Cartesian length of **116**
- converting objects to polylines **157**
- создание **202**
- creating cutter objects **175**
- determining length of **414**
- extracting a range of nodes from **280**
- modifying the pen style **90–91**
- querying the pen style **410–414**
- storing in a new row **337–339**
- storing in an existing row **697**

**Элемент управления PopupMenu** **154**

**Positioning the row cursor** **283–285**

**Приоритет операторов** **768, 292**

**Preferences dialog box(es)** **508**

**Preventing user from closing windows** **640**

**Print # оператор** **465**

**Print оператор** **464**

**Printer settings** **635–642**

- overriding default printer **641**

**Printing attributes** **708**

**PrintWin оператор** **466**

## **Prism maps**

- creating **205–207**
- properties **467–470**
- setting **625–626**

**PrismMapInfo( )** функция **467**

## **Procedures, creating**

- Call statement **110–112**
- Declare Sub оператор **241, 222**
- Exit Sub statement **275**
- Sub...End Sub statement **674**

## **Procedures, special**

- EndHandler **267**
- ForegroundTaskSwitchHandler **299**
- Main **369–370**
- RemoteMapGenHandler **490**
- RemoteMsgHandler **491**
- SelChangedHandler **522**
- ToolHandler **688–690**
- WinChangedHandler **702**

WinClosedHandler **703**

WinFocusChangedHandler **711**

**ProgramDirectory\$( )** функция **471**

**Индикатор выполнения, скрытие** **626**

**ProgressBar** оператор **471**

**Проекции**

changing a table's projection **139–143**

copying from a table or window **159–162**

querying a table's CoordSys **682**

querying a window's CoordSys **383**

setting the current MapBasic CoordSys **584**

setting within an application **126**

**Proper\$( )** function **474**

**Proportionate aggregates**

Proportion Avg( ) **66–68**

Proportion Sum( ) **66–68**

Proportion WtAvg( ) **66–68**

**ProportionOverlap( )** функция **475**

**PSD files, creating** **513–515**

**Put statement** **475**

## Q

**Quantiled ranges** **208**

## R

**RadioGroup** элемент управления **155**

**Файлы произвольного доступа**

closing files **128**

opening files **447**

reading data **318–319**

writing data **475**

**Случайные числа:**

Randomize statement **476**

Rnd( ) function **499**

**Randomize statement** **476**

**Ranged thematic maps** **207–209, 648–650**

**ReadControlValue( )** функция **477**

**Приложение реального времени** **70**

**Записи** Смотрите **Записи**

**Rectangle objects**

Cartesian area of **112**

Cartesian perimeter of **119**

creating **210, 215**

determining area of **97**

determining perimeter of **461**

modifying **89–92**

querying the pen or brush style **410–414**

storing in a new row **337–339**

storing in an existing row **697**

**ReDim** оператор **480**

**Redistricting windows**

заккрытие **130**

---

- modifying **627, 635–642**
- открытие **211**
- Полигональные объекты**
  - adding/removing nodes **92**
  - Cartesian area of **112**
  - Cartesian perimeter of **119**
  - checking for data errors **422**
  - converting objects to regions **158**
  - creating **212–214**
  - creating convex hull objects **158**
  - determining area of **97**
  - determining perimeter of **461**
  - extracting a range of nodes from **280**
  - modifying the pen or brush style **90–91**
  - querying the pen or brush style **410–414**
  - returning a buffer region **113**
  - storing in a new row **337–339**
  - storing in an existing row **697**
- Region, setting a centroid 93**
- Regional settings 591–593**
- Register Table оператор 482**
- Relational joins 525–527**
- Relief Shade statement 489**
- Reload Symbols statement 489**
- Remote databases**
  - creating new tables **550–552**
  - refreshing linked tables **565**
  - retrieving active database connection info **546**
  - shutting down server connection **553**
- RemoteMapGenHandler процедура 490**
- RemoteMsgHandler процедура 491**
- RemoteQueryHandler function 492**
- Оператор Remove Cartographic Frame**
- Remove Map Layer оператор 494**
- Removing**
  - buttons **72–76**
  - menu items **82–85**
  - nodes **92**
- Rename File оператор 495**
- Rename Table statement 495**
- Reports**
  - создание **214**
  - loading **448**
- Reproject statement 496**
- Reserved words 257**
- Resizing arrays 480–481**
- Responding to system events. Смотрите События, обработка**
- Resume оператор 496**
- Retrieving column information**
  - Server\_ColumnInfo( ) **536**
- Retrieving data source information**
  - Server\_DriverInfo( ) **554**
- Retrieving number of columns in a results set**

- Server\_NumCols( ) **563**
- Retrieving number of toolkits**
  - Server\_NumDrivers( ) **564**
- Retrieving records from an open table**
  - Fetch statement **283**
- Retrieving rows from a results set**
  - Server Fetch **557–558**
- Retrying on file access** **591**
- Returning a connection number**
  - Server\_Connect **539–546**
- Returning a coordinate system**
  - ChooseProjection\$( ) функция **126**
- Returning a date**
  - FormatDate\$( ) функция **302**
- Преобразует значение, заданное в типографских единицах “пункты”, в ширину линии.**
  - PointsToPenWidth( ) функция **462**
- Returning a point size for a pen width**
  - PenWidthToPoints( ) функция **460**
- Returning ODBC connection handle**
  - Server\_GetodbcHConn( ) function **559**
- Returning ODBC statement handle**
  - Server\_GetodbcHStmt( ) function **560**
- Returning pen width units**
  - IsPenWidthPixels( ) функция **344**
- RGB( ) функция** **497**
- Right\$( ) function** **498**
- Rnd( ) function** **499**
- RollBack оператор** **500**
- Rotate( ) function** **501**
- RotateAtPoint( ) функция** **502**
- Rotated map labels** **618**
- Rotated symbols** **374, 676**
- Round( ) function** **503**
- Rounded Rectangle objects**
  - Cartesian area of **112**
- Rounded rectangle objects**
  - Cartesian perimeter of **119**
  - создание **215**
  - modifying **89–92**
  - querying the pen or brush style **410–414**
  - storing in a new row **337–339**
  - storing in an existing row **697**
- Rounding off a number**
  - Fix( ) function **294**
  - Format\$( ) функция **300**
  - Int( ) function **340**
  - Round( ) function **503**
- RowID**
  - after Find operations **137**
  - with SelChangedHandler **136**
- Rows in a Browser, positioning** **576**
- Строки в таблице**
  - deleting rows **245**

---

- end-of-table condition **268**
- inserting new rows **337–339**
- packing (purging deleted rows) **453**
- positioning the row cursor **283–285**
- selecting rows that satisfy criteria **525–527**
- обновление существующих строк **697**
- RPC (Remote Procedure Calls) 241–243**
- RTrim\$() функция 504**
- Ruler tool**
  - closing Ruler window **130**
  - modifying Ruler window **635–642**
  - opening Ruler window **451**
- Run Application оператор 504**
- Run Command statement 505**
- Run Menu Command оператор 507, 143**
- Run Program statement 508**
- Runtime errors, trapping. *See* Error handling**

## **S**

- Save File оператор 511**
- Save MWS оператор 511**
- Save Window statement 513**
- Save Workspace statement 515**
- Saving changes to a table 139–143**
- Saving linked tables 142**
- Saving work to the database**
  - Server Commit **538**
- Scale of a map**
  - determining **385**
  - displaying **609**
- Область определения переменных**
  - глобальные **323**
  - local **254–258**
- Линейки прокрутки, показать или скрыть 640**
- Scrolling automatically 641**
- Сшитые таблицы**
  - determine if table is seamless **683**
  - prompt user to choose a sheet **322–323**
  - turn seamless behavior on/off **633**
- SearchInfo() функция 515**
- Searching for map objects**
  - at a point **518**
  - processing search results **515–518**
  - within a rectangle **519**
- SearchPoint() функция 518**
- SearchRect() function 519–520**
- Second function 520**
- Seconds, elapsed 687**
- Seek statement 521**
- Seek() function 521**
- SelChangedHandler процедура 522**
- Select Case. *See* Do Case...End Case statement**

Select statement **523**  
 Selectable map layers **612**  
 Выборка  
     handling selection-changed event **135, 522**  
     interrupted by Esc key **523**  
     querying current selection **532**  
     Select statement **523**  
 SelectionInfo() функция **532**  
 Self-intersections, checking in regions **422–424**  
 Файлы последовательного доступа  
     closing files **128**  
     opening files **447**  
     reading data **336, 364**  
     writing data **465, 713**  
 Оператор Server Begin Transaction  
 Оператор Server Bind Column  
 Server Close statement **535**  
 Server Commit statement **538**  
 Оператор Server Create Map  
 Оператор Server Create Style  
 Оператор Server Create Table  
 Оператор Server Create Workspace  
 Server Disconnect statement **553**  
 DriverInfo() функция **554**  
 Server Fetch statement **557**  
 Оператор Server Link Table  
 Server Refresh statement **565**  
 Оператор Server Remove Workspace  
 Server Rollback statement **566**  
 Оператор Server Set Map  
 Server Versioning statement **568**  
 Оператор Server Workspace Merge  
 Оператор Server Workspace Refresh  
 Server\_ColumnInfo() function **536**  
 Server\_Connect() function **539**  
 Server\_ConnectInfo() function **546**  
 Server\_EOT() function **555**  
 Server\_Execute function **556**  
 Server\_GetODBCConn() function **559**  
 Server\_GetODBCStmt() function **560**  
 Server\_NumCols() function **563**  
 Server\_NumDrivers() function **564**  
 server\_string, defined **556**  
 SessionInfo() функция **573**  
 Оператор Set Application Window  
 Оператор Set Area Units  
 Set Browse statement **576**  
 Set Cartographic Legend оператор **577**  
 Оператор Set Command Info  
 Оператор Set Connection Geocode  
 Оператор Set Connection Isogram  
 Set CoordSys оператор **584, 212**  
 Оператор Set Date Window

---

Set Datum Transform Version statement **586**  
Set Digitizer statement **587**  
Оператор Set Distance Units  
Оператор Set Drag Threshold  
Set Event Processing оператор **590**  
Set File Timeout оператор **591**  
Set Format оператор **591**  
Set Graph statement **593**  
Set Handler statement **598**  
Set Layout statement **599**  
Set Cartographic Legend оператор **601**  
Set Map statement **604**  
Set Map3D statement **620**  
Оператор Set Next Document  
Оператор Set Paper Units  
Set PrismMap statement **625**  
Set ProgressBars statement **626**  
Set Redistricter оператор **627**  
Set Resolution statement **629**  
Set Shade оператор **629**  
Set Style statement **630**  
Set Table оператор **632**  
Set Target оператор **634**  
Close Window оператор **635, 235**  
Sgn( ) function **643**  
Shade оператор **645**  
Shadow text **295–297**  
Shapefiles **487**  
Shift key  
    detecting shift-click **137**  
    effect on drawing tools **74–75**  
    selecting multiple list items **150–152**  
**Всплывающее меню**  
    disabling **193**  
    Пример 85:  
**Show/Hide menu commands 190**  
**Showing**  
    ButtonPads **72–76**  
    dialog box controls **77–79**  
    menu bar **388**  
**Shutting down the connection**  
    Server Disconnect **553**  
**Симуляция выбора команды меню 507**  
Sin( ) function **654**  
Small integer variables **255**  
Smart redraw **609**  
Snap tolerance, controlling **642**  
Snapping nodes **437–439**  
Сортировка строк в таблице **530**  
Sounds, beeping **104**  
Space\$( ) function **655**  
Spaces, trimming from a string **368, 504**  
Скорость, повышение



- Анимационный слой **70**
- screen updates **590**
- table editing **632–634**
- SphericalArea( ) функция****655**
- SphericalConnectObjects( ) функция****656**
- SphericalDistance( ) функция****657**
- SphericalObjectDistance( ) функция****658**
- SphericalObjectLen( ) функция****659**
- SphericalOffset( ) функция****660**
- SphericalOffsetXY( ) функция****661**
- SphericalPerimeter( ) функция****662**
- Splitting objects** **439–441**
- Spreadsheets, using as tables** **482–488**
- SQL Select** **523–532**
- SQL Server databases**
  - connection string attributes **542–545**
- Sqr( ) function** **663**
- Starting other applications**
  - Run Application оператор **504**
  - Run Program statement **508**
- Элемент управления StaticText** **156**
- Statistical calculations**
  - average **66, 528–529**
  - count **66, 528–529**
  - min/max **66, 528–529**
  - quantile **208**
  - standard deviation **207**
  - sum **66, 528–529**
  - weighted average **66, 529**
- Statistics window**
  - закрытие **130**
  - modifying **635–642**
  - открытие **451**
- Подсказки в строке сообщений** **73**
- StatusBar оператор** **664**
- Stop оператор** **664**
- Str\$( ) function** **665**
- Street addresses, finding** **290–293**
- Слияние строк**
  - & оператор **766**
  - + оператор **766**
- Функции работы со строками**
  - capitalization **358, 474, 693**
  - comparison **667–668**
  - converting codes to strings **126**
  - converting strings to codes **99**
  - converting strings to dates **591–593, 669–670**
  - converting strings to numbers **591–593, 698**
  - converting values to strings **665**
  - extracting part of a string **359, 395–396, 498**
  - finding a substring within a string **339**
  - formatting a number **244, 300–303, 591–593**
  - formatting based on locale **591–593**

---

- length of string **363**
- locale settings **591–593**
- Сравнение по шаблону **363**
- repeated strings **667**
- spaces **655**
- trimming spaces from end **504**
- trimming spaces from start **368**
- String variables 255**
- String\$( ) function 667**
- StringCompare( ) функция 667**
- StringCompareIntl( ) функция 668**
- StringToDate( ) функция 669**
- StringToDateTime( ) функция 670**
- StringToTime( ) функция 671**
- Структура 691**
- StyleAttr( ) функция 671**
- Styles.** *See specific type*
  - Pen, Brush, Font, Symbol
- Подпрограмма Смотрите Процедуры**
- Sub...End Sub statement 674**
- Subtotals, calculating 528–529**
- Sum( ) aggregate function 528–529**
- Symbol clause 676**
- Symbol styles**
  - creating **371, 374, 376**
  - modifying an object's style **90–91**
  - querying an object's style **410–414**
  - querying parts of **670–673**
  - reading current style **227**
  - reloading symbol sets **489**
  - setting current style **630–631**
  - Symbol clause defined **676**
- Symbol variables 255**
- SYMBOL.MBX utility**
  - custom symbols **489**
- Элемент управления SymbolPicker 153**
- SystemInfo( ) функция 679**

## T

- TAB files, storing metadata in 390–393**
- Tab order 249**
- Имена таблиц**
  - determining \_ in Browse or Graph window **707**
  - determining from file **456**
  - determining table name from number **681–684**
  - special names for Cosmetic layers **707**
  - special names for Layout windows **707**
- Структура таблицы**
  - 3DMap **186–187**
  - adding/removing columns **95–96**
  - определение числа колонок (404) **681**
  - Присоединение геоинформации к удаленной таблице

making an ODBC table mappable 547–549

**TableInfo( ) функция** 681

**Tables, closing**

Close All statement 127

Close Table statement 129

**Tables, copying** 139–143

**Tables, creating**

Создание файла проекта 218

importing a file 331–335

on remote databases 550–552

using a spreadsheet or database 482–488

**Tables, deleting** 265

**Tables, importing** 331–335

**Tables, modifying**

adding columns 64–68, 95–96

adding metadata 390–393

adding rows 337–339

creating an index 181

deleting a table's objects 264

deleting an index 263

deleting columns 95–96

deleting rows or objects 245

discarding changes 500

optimizing edit operations 632–634

packing 453

renaming 495

saving changes 139–143

setting a map's default view 619

setting a map's projection 139–143

setting to read-only 632–634

sorting rows 530

обновление существующих строк 697

**Tables, opening** 449–451

**Таблицы, запросы**

column information 131–133

directory path 683

end-of-table condition 268

finding a map address 290–293

joining 525–527

metadata 321, 390–393

число открытых таблиц 405

objects at a point 518

objects in a rectangle 519–520

positioning the row cursor 268, 283–285

SQL Select 523–532

table information 681–684

**Tan( ) function** 684

**TempFileName\$( ) функция** 685

**Temporary columns** 64

**Terminate Application statement** 686

**Текстовые файлы.**

using as tables 482–488

See also File input/output, Files

---

## Текстовые объекты

- creating [220–221](#)
- modifying [89–92](#)
- querying the font style or string [411](#)
- storing in a new row [337–339](#)
- storing in an existing row [697](#)

**Text styles.** *See* **Font styles**

**TextSize( )** функция [686](#)

## Thematic maps

- bar chart maps [653–654](#)
- counting themes in a 3D Map window [378–381](#)
- counting themes in a Map window [382–386](#)
- creating arrays of ranges [207–209](#)
- creating arrays of styles [216–217](#)
- dot density maps [651](#)
- Карта градуированных символов [651](#)
- grid surface maps [179–181](#)
- individual value maps [650](#)
- изменение [629](#)
- pie chart maps [652–653](#)
- quantiled ranges [208](#)
- ranged maps [648–650](#)

**Thinning objects** [437–439](#)

**Thousands separators** [244](#), [303](#), [591–593](#)

**TIFF files, creating** [513–515](#)

**Time delay when user drags mouse** [590](#)

**Time( )** function [687](#)

**Timer( )** function [687](#)

**ToolHandler** процедура [688](#)

**Tooltip help** [73](#), [166](#)

**Totals, calculating** [528–529](#)

**Transparent fill patterns** [107](#)

**Trapping errors.** *See* **Error handling**

**TriggerControl( )** функция [690](#)

## Тригонометрические функции:

- arc-cosine [60](#)
- arc-sine [100](#)
- arc-tangent [102](#)
- cosine [163](#)
- sine [654](#)
- tangent [684](#)

## Trimming spaces

- from end of string [504](#)
- from start of string [368](#)

**TrueFileName\$( )** функция [691](#)

**TrueType fonts, using as symbols** [374](#)

**TrueType symbols** [676–679](#)

**Type statement** [691](#)

## U

**UBound( )** функция [692](#)

**UCase\$( )** функция [693](#)

**Unchecking**

- dialog box check boxes (custom) 77–79
- dialog box check boxes (standard) 80–82
- menu items 87–88

**Underlined text 295–297****UnDim оператор 694****Undo system, disabling 632–634****UnitAbbr\$() функция 695****UnitName\$() функция 695****Единицы измерения**

- abbreviated names 695
- area 575
- distance 588
- full names 695
- paper 624

**Unlink statement 696****Unselecting 130****Update statement 697****Update Window statement 698****Upper case, converting to 693****Интерфейс пользователя See ButtonPads, Dialog boxes, Menus, Windows****V****Val() function 698****Variable length strings 255****Переменные**

- arrays 256, 480–481, 692
- custom types 691
- Глобальные переменные 323
- initializing 258
- list of types 254–255
- local variables 254–258
- Чтение переменных другого приложения 323
- ограничения на имена 257
- strings variables 256
- undefining 694

**Version number**

- .MBX version 679
- MapInfo version 680

**Промежуточные узлы (вертексы) Смотрите Узлы****Voronoi polygons**

- Create Object 199

**W****Weekday() function 699****Weighted averages 66, 529****Оператор WFS Refresh Table****While...Wend statement 700****Wildcards, matching 363****WinChangedHandler процедура 702****WinClosedHandler процедура 703****WindowID() функция 704**

---

**WindowInfo( ) функция** 705  
**Windows Latin 1 character set** 763  
**Windows operating system, 16- v. 32-bit** 680  
**Windows, closing**  
    Close Window оператор 130, 235  
    preventing user from closing windows 640  
**Windows, modifying**  
    adding map layers 69–70  
    Окно Списка 576  
    forcing windows to redraw 698  
    general window settings 635–642  
    graph windows 593–597  
    layout windows 599–601  
    legend window 601–603  
    map windows 604–620  
    redistrict windows 627–628  
    removing map layers 494  
**Windows, opening**  
    Browse statement 104–106  
    Create Redistrict statement 211  
    Graph statement 325  
    Layout statement 357  
    Map statement 377  
    Open Window оператор 451, 235  
**Windows, printing**  
    to a file 513–515  
    to an output device 466  
**Окна, запросы**  
    3D Map window settings 378–381  
    general window settings 705–711  
    ID of a window 704  
    ID of front window 306  
    map window settings 352–357, 382–386  
    number of document windows 406  
    total number of windows 402  
**WinFocusChangedHandler процедура** 711  
**WKS files, opening** 482–488  
**WMF files, creating** 513–515  
**Рабочие наборы**  
    loading 504  
    saving 515  
**Write # оператор** 713  
**WtAvg( ) функция** 529  
**X**  
**XCMDs** 242  
**XFCNs** 240–241  
**XLS files, opening** 482–488  
**Библиотека XML**  
    MIXmlAttributeListDestroy( ) procedure 746  
    MIXmlDocumentCreate( ) функция 746  
    MIXmlDocumentDestroy( ) procedure 747

MIXmlDocumentGetNamespaces( ) функция **747**  
MIXmlDocumentGetRootNode( ) функция **748**  
MIXmlDocumentLoad( ) функция **748**  
MIXmlDocumentLoadXML( ) функция **749**  
MIXmlDocumentLoadXMLString( ) функция **750**  
MIXmlDocumentSetProperty( ) функция **751**  
MIXmlGetAttributeList( ) функция **752**  
MIXmlGetChildList( ) функция **752**  
MIXmlGetNextAttribute( ) функция **753**  
MIXmlGetNextNode( ) функция **754**  
MIXmlNodeDestroy( ) procedure **754**  
MIXmlNodeGetAttributeValue( ) функция **755**  
MIXmlNodeGetFirstChild( ) функция **755**  
MIXmlNodeGetName( ) функция **756**  
MIXmlNodeGetParent( ) функция **757**  
MIXmlNodeGetText( ) функция **757**  
MIXmlNodeGetValue( ) функция **758**  
MIXmlNodeListDestroy( ) procedure **758**  
MIXmlSCDestroy( ) procedure **759**  
MIXmlSCGetLength( ) функция **759**  
MIXmlSCGetNamespace( ) функция **760**  
MIXmlSelectNodes( ) функция **760**  
MIXmlSelectSingleNode( ) функция **761**

## Y

Year( ) function **713**

