美国微软出版社授权Microsoft Visral中文版系列

新远工作室

Basic 6.0

中文版语言参考手册

Microsoft Corporation 著
微软（中国）有限公司 译
希望图书创作室 校
新远工作室 制作

北京希望电脑公司出品

内容提要

为满足中国用户更快、更好地学习和掌握新的微机编程和开发工具，美国微软公司的一种特别的做法是：新版软件配套的中文版使用手册和开发指南单独销售，这就为从事微软相关软件开发的群体和个人提供了很大的方便。通过配套手册的学习，可大大缩短掌握和应用相关软件的时间，从而提高开发效率。这次推出的 MicrosoftVisualStudio 中文版使用手册和开发指南将涉及 VisualFoxPro6.0、VisualBasic6.0、VisualC++6.0、VisualInterDev6.0、VisualJ++6.0，请广大读者密切留意出版信息。

本书是美国微软出版社授权的 MicrosoftVisualStudio 中文版系列中的一本。全书按字母顺序列出了 VisualBasic6.0 的函数、语句、方法、属性及事件，附录部分提供了 ANSI 字符集、数据类型、运算符等等的数学函数及转换函数。

本书不但是从事 VisualBasic6.0 应用和开发人员的工具书，也可作为大专院校相关专业的师生、科研院所的科技人员自学和教学的重要参考书。

本书还提供配套的电子书，以方便读者携带、学习和长久保存。电子书的配置要求和使用方法请参见光盘中的 Readme 文件。

欲购本书或需技术支持的用户请直接与北京海淀区 8721 信箱书刊服务部联系，电话：010-62562329，62541922，传真：010-62579874，62633308。邮政编码：100080。

VisualBasic6.0 中文版语言参考手册

MicrosoftCorporation 著
微软（中国）有限公司译
希望图书创作室校
新远工作室制作
责任编辑陆卫民

北京希望电脑公司出品
北京海淀路 82 号（100080）
新华书店新华书店音像发行所各地新华书店及软件专卖店经销

*****

1998 年 9 月第 1 版 1998 年 9 月第 1 次印刷
开本:787×10921/16 印张:76.5
字数:1785 千字印数:1~5000 册
新出音管字[1998]164 号
ISBN7-980021-20-7/TP·13

序

为了满足广大电脑爱好者学习 VisualBasic6.0 中文版软件的需要，及配合微软（中国）有限公司发布 VisualBasic6.0 中文版，美国微软出版社授权翻译、出版以下 VisualBasic6.0 系列手册：

编号	书名	估计定价（元）
CX-2463	VisualBasic6.0 中文版程序员指南	98
CX-2464	VisualBasic6.0 中文版语言参考手册	120
CX-2542	VisualBasic6.0 中文版控件参考手册	70
CX-2453	VisualBasic6.0 中文版组件工具指南	70
CX-2477	VisualBasic6.0 中文版编程基础	62
CX-2483	VisualBasic6.0 中文版技巧	70
CX-2484	VisualBasic6.0 中文版与 SQLSever 编程	78
CX-2485	VisualBasic6.0 中文版高级程序设计	78
CX-2486	VisualBasic6.0 中文版组件开发指南	62

该系列书具备权威性、准确性和启发性，内容全面，是 VB 编程用户的必备工具书，欢迎各界新老朋友订购。

欲购书或需技术支持的读者，请直接与北京希望电脑公司联系，联系电话：(010)-62562329,62541992 或传真至(010)-62579874，或写信至北京海淀 8721 信箱书刊部(邮编 100080)。

希望图书创作室

1998 年 9 月

目录

#Const 指令 34
#If...Then...#Else 指令 36
Abs 函数 39
AccessKeyPress 事件 39
AccessKeys 属性 40
Action 属性（“通用”对话框） 41
Action 属性（OLE 容器） 43
Activate 方法 45
Activate、Deactivate 事件 ... 45
ActiveCodePane 属性 47
ActiveControl 属性 48
ActiveForm 属性 52
ActiveVBProject 属性 57
ActiveWindow 属性 57
Add 方法（Add-In） 59
Add 方法(BindingCollection) . 63
Add 方法（Controls 集合） ... 69
Add 方法（DataMembers 集合） 75
Add 方法（DataObjectFiles 集合） 75
Add 方法(Dictionary) 76
Add 方法（ExportFormats 集合） 77
Add 方法(Folder) 80
Add 方法（Format 对象） 81
Add 方法（Licenses 集合） ... 81
Add 方法（VBA 外接程序对象模型） 85
Add 方法（VisualBasicforAPplication） 87
AddCustom 方法 90
AddFile 方法 92
AddFromFile 方法 93
AddFromFile 方法（VBA 外接程序对象模型） 93
AddFromGuid 方法 94

AddFromString 方法 95
AddFromTemplate 方法 96
AddIn 对象 97
“Add-In”工具栏 98
AddIns 集合 99
AddIns 属性 100
AddItem 方法 100
AddressOf 运算符 103
AddToAddInToolbar 方法 109
AddToolboxProgID 方法 111
AfterAddFile 事件 112
AfterChangeFileName 事件 ... 114
AfterCloseFile 事件 116
AfterRemoveFile 事件 118
AfterWriteFile 事件 119
Align 属性 122
Alignable 属性 125
Alignment 属性 126
Ambient 属性 128
AmbientChanged 事件 128
AmbientProperties 对象 129
App 对象 133
App 属性 134
AppActivate 语句 134
Appearance 属性 136
AppIsRunning 属性 138
Application 属性(WebClass) . 139
ApplyChanges 事件 140
Archive、Hidden、Normal 和 System 属性 141
Arrange 方法 143
Array 函数 146
Asc 函数 147
Assert 方法 148
AsyncCount 属性 150
AsyncLoad 属性 151
AsyncProgress 事件 152
AsyncProperty 对象 153
AsyncRead 方法 154
AsyncReadComplete 事件 157
AsyncReadProgress 事件 158
AsyncType 属性 160
AtEndOfLine 属性 161
AtEndOfStream 属性 162
Atn 函数 163
Attributes 属性 164

AutoActivate 属性 167
AutoRedraw 属性 169
AutoShowChildren 属性 172
AutoSize 属性 174
AutoVerbMenu 属性 175
AvailableSpace 属性 177
BackColor、ForeColor 属性 .. 178
BackStyle 属性 181
BackStyle 属性（UserControl 对象） 182
BaseWindow 属性 184
Beep 语句 185
BeforeLoadFile 事件 185
BeginRequest 事件 186
Bindable 属性 187
Binding 对象 188
BindingCollection 对象 191
Bold 属性 194
BorderColor 属性 196
BorderStyle 属性 198
BorderWidth 属性 201
BottomMargin、TopMargin 属性204
Browsable 属性 205
BrowserType 属性 206
BuildFile 属性 207
BuildFileName 属性 207
BuildPath 方法 207
BuiltIn 属性 208
BytesMax 属性 209
BytesRead 属性 210
Calendar 属性 210
Call 语句 211
CallByName 函数 213
Cancel 属性 214
CancelAsyncRead 方法 216
CancelError 属性 217
CanGetFocus 属性 218
CanGrow 属性 219
CanPaste 属性 220
CanPropertyChange 方法 220
Caption 属性 222
Caption 属性 224
Category 属性 225
CausesValidation 属性 225

Change 事件 229
Changed 属性 232
Charset 属性 233
ChDir 语句 234
ChDrive 语句 236
CheckBox 控件 237
Checked 属性 238
Choose 函数 240
Chr 函数 242
Circle 方法 243
类 246
Class 属性 246
ClassName 属性 248
Clear 方法 248
Clear 方法（Clipboard、ComBo Box、ListBox） 250
Clear 方法（DataObject 对象） 253
Clear 方法（Format 对象） .. 253
Click 事件 254
Click 事件（VBA 外接程序对象模型） 257
ClipBehavior 属性 259
Clipboard 对象 260
Clipboard 属性 261
ClipControls 属性 262
Close 方法（VBA 外接程序对象模型） 264
Close 方法 265
Close 方法（OLE 容器） 266
Close 语句 266
Cls 方法 267
CodeLocation 属性 269
CodeModule 对象 270
CodeModule 属性 271
CodePane 对象 272
CodePane 属性 273
CodePane 集合 274
CodePanes 属性 274
CodePaneView 属性 275
Col，Row 属性 276
Collection 对象 280
Collection 属性 283
Color 属性 284
ColorMode 属性 285
Column 属性 286

Columns 属性 287
ComboBox 控件 289
Command 函数 291
CommandBar 对象 293
CommandBarEvents 对象 295
CommandBarEvents 属性 295
CommandBars 集合 297
CommandBars 属性 298
CommandButton 控件 298
Comments 属性 300
CommonDialog 控件 300
CommonDialog 控件（“颜色”对话框） 303
CommonDialog 控件（“字体”对话框） 305
CommonDialog 控件（帮助） .. 308
CommonDialog 控件（“打开”、“另存为”对话框） 312
CommonDialog 控件（"打印"对话框） 314
CompanyName 属性 316
CompareMode 属性 317
CompatibleOLEServer 属性 ... 319
Connect 属性 320
Const 语句 320
ContainedControls 集合 322
ContainedControls 属性 322
ContainedVBControls 集合 ... 324
ContainedVBControls 属性 ... 325
Container 属性 325
Container 属性 327
ContainerHWnd 属性 328
ContinuousScroll 属性 328
Control 类 329
Control 对象 330
ControlBox 属性 330
ControlContainer 属性 331
ControlObject 属性 333
Controls 集合 333
Controls 属性 335
ControlType 属性 336
Copies 属性 336
Copy 方法 338
Copy 方法（OLE 容器控件） .. 338

Copy 方法 339
CopyFile 方法 340
CopyFolder 方法 342
Cos 函数 344
Count 属性 345
Count 属性 345
Count 属性 346
Count 属性（VB 集合） 347
CountOfDeclarationLines 属性 348
CountOfLines 属性 349
CountOfVisibleLines 属性 ... 350
CreateEmbed 方法 350
CreateEventProc 方法 352
CreateFolder 方法 353
CreateLink 方法 354
CreateObject 函数 355
CreateTextFile 方法 358
CreateToolWindow 方法 360
CurDir 函数 361
CurrentX，CurrentY 属性 362
Cut 方法 364
CVErr 函数 364
Data 属性 366
数据类型概述 367
DataBinding 对象 370
DataBindingBehavior 属性 ... 370
DataBindings 集合 372
DataBindings 属性 372
DataFormat 对象 373
DataFormat 属性 374
DataFormats 集合 375
DataFormats 属性 376
DataMember 属性 382
DataMembers 集合 390
DataMembers 属性 392
DataObject 对象 392
DataObjectFiles 集合 394
DataReport 对象 394
DataSourceBehavior 属性 396
DataText 属性 397
Date 数据类型 400
Date 函数 400
Date 语句 401
DateAdd 函数 402
DateCreated 属性 405

DateDiff 函数 406
DateLastAccessed 属性 409
DateLastModified 属性 411
DatePart 函数 412
DateSerial 函数 415
DateValue 函数 417
Day 函数 418
DblClick 事件 419
DDB 函数 422
Debug 对象 425
Decimal 数据类型 425
Declare 语句 426
Default 属性 430
DefaultBind 属性 432
DefaultCancel 属性 432
DefaultExt 属性 434
Deftype 语句 435
Delete 方法 439
Delete 方法（OLE 容器） 440
DeleteFile 方法 440
DeleteFolder 方法 441
DeleteLines 方法 442
DeleteSetting 语句 444
导出的数学函数 446
Description 属性（应用于 VisualBasic 子应用程序） ... 448
Description 属性（添加到 ObjectModelVBA） 449
Designer 属性 450
DesignerID 属性 451
DesignerWindow 方法 452
DeviceName 属性 453
DHTMLEvent 属性 454
DHTMLPage 对象 455
DHTMLPageDesigner 对象 456
DialogTitle 属性 456
Dictionary 对象 457
Dim 语句 458
Dir 函数 463
DirListBox 控件 466
DisabledPicture 属性 467
DisplayAsDefault 属性 469
DisplayBind 属性 470
DisplayModel 属性 470

DisplayName 属性 471
DisplayType 属性 472
Do...Loop 语句 473
Document 属性(DHTMLPageDesiGNer) 475
DoEvents 函数 476
DoGetNewFileName 事件 478
Double 数据类型 481
DoVerb 方法 481
DownPicture 属性 484
Drag 方法 486
DragDrop 事件 491
DragIcon 属性 494
DragMode 属性 497
DragOver 事件 499
DrawMode 属性 503
DrawStyle 属性 506
DrawWidth 属性 508
Drive 对象 510
Drive 属性 511
Drive 属性 513
DriveExists 方法 514
DriveLetter 属性 515
DriveListBox 控件 516
DriverName 属性 518
Drives 集合 519
Drives 属性 520
DriveType 属性 522
DropDown 事件 523
Duplex 属性 525
EditAtDesignTime 属性 527
EditProperty 事件 528
Empty 529
Enabled 属性 529
End 语句 532
EndDoc 方法 534
EndRequest 事件 536
EnterFocus 事件 537
Enum 语句 538
Environ 函数 541
EOF 函数 543
Erase 语句 544
Err 对象 546
Error 事件（数据报表设计器） 548
Error 函数 549
Error 对象（数据报表设计器） 550

Error 属性（WebClass 对象）551
Error 语句 551
ErrorNumber 属性 553
Event 语句 554
EventInfo 对象 559
EventParameter 对象 560
EventParameters 集合 560
EventParameters 属性（EventInfo 对象） 561
Events 对象 561
Events 属性 562
EventsFrozen 属性 562
EXEName 属性 563
Exists 方法 564
Exit 语句 564
ExitFocus 事件 566
Exp 函数 567
Export 方法（VBA 外接程序对象模型） 568
ExportFormat 对象 569
ExportFormats 集合 571
ExportFormats 属性 575
ExportReport 方法 577
Extender 对象 582
Extender 属性 586
False 586
FalseValue 属性 587
FatalErrorResponse 事件 588
FetchVerbs 方法 589
File 对象 589
FileAttr 函数 590
FileControlEvents 对象 592
FileControlEvents 属性 593
FileCopy 语句 593
FileCount 属性 594
FileDateTime 函数 595
FileDescription 属性 596
FileExists 方法 596
FileFilter 属性 597
FileFormatString 属性 600
FileLen 函数 603
FileListBox 控件 604
FileName 属性 605
FileName 属性 606

FileNames 属性 608
FileNumber 属性 609
Files 集合 610
Files 方法 610
Files 属性 612
Files 属性 613
FileSystem 属性 614
FileSystemObject 对象 615
FileTitle 属性 616
FillColor 属性 617
FillStyle 属性 618
Filter 函数 620
Filter 属性（公共对话框） .. 622
FilterIndex 属性 623
Find 方法（VBA 外接程序对象模型） 624
FirstDayOfWeek 属性 626
FirstWeekOfYear 属性 628
Flags 属性（"颜色"对话框） . 629
Flags 属性（“字体”对话框） 631
Flags 属性（“打开”、“另存为”对话框） 634
Flags 属性（“打印”对话框） 639
Folder 对象 642
FolderExists 方法 643
Folders 集合 644
Font 对象 645
Font 属性 646
FontBold、FontItalic、ontStrikethru、FontUnderline 属性 .. 647
FontChanged 事件 649
FontCount 属性 650
FontName 属性 651
Fonts 属性 654
FontSize 属性 655
FontTransparent 属性 657
ForEach...Next 语句 659
For...Next 语句 661
ForcePageBreak 属性 664
Form 对象、Forms 集合 666
Format 事件（StdDataFormat 对象） 669
Format 函数 670
Format 属性（StdDataFormat 对象） 674
Format 属性 674

FormatCurrency 函数 676
FormatDateTime 函数 678
FormatNumber 函数 680
FormatPercent 函数 681
FormatType 属性 683
Forms 属性 686
ForwardFocus 属性 687
Frame 控件 688
FreeFile 函数 689
FreeSpace 属性 690
Friend 691
FromPage,ToPage 属性 693
FullName 属性 694
FullPath 属性 695
Function 控件（数据报表设计器） 695
Function 语句 696
FunctionType 属性 704
FV 函数 705
Get 语句 709
Get 语句 710
GetAbsolutePathName 方法 ... 714
GetAllSettings 函数 715
GetAttr 函数 717
GetAutoServerSettings 函数 . 719
GetBaseName 方法 721
GetData 方法 722
GetData 方法（DataObject 对象） 725
GetDataMember 事件 727
GetDrive 方法 729
GetDriveName 方法 730
GetExtensionName 方法 732
GetFile 方法 732
GetFileName 方法 733
GetFolder 方法 734
GetFormat 方法 735
GetFormat 方法（DataObject 对象） 738
GetObject 函数 739
GetParentFolderName 方法 ... 746
GetSelection 方法 747
GetSetting 函数 749
GetSpecialFolder 方法 751

GetTempName 方法 752
GetText 方法 753
Global 对象 756
GoBack 方法 757
GoForward 方法 757
GoSub...Return 语句 758
GotFocus 事件 760
GotFocus 事件（UserControl 对象和 UserDocument 对象） ... 763
GoTo 语句 764
GridLineWidth 属性 765
GridX、GridY 属性 766
GUID 属性 766
Handle 属性 767
HasDC 属性 768
HasOpenDesigner 属性 769
hDC 属性 770
Height 属性（VBA 外接程序对象模型） 772
Height、Width 属性 773
HelpCommand 属性 777
HelpContext 属性 783
HelpContext 属性（"公共"对话框） 784
HelpContextID 属性 788
HelpContextID 属性（VBA 外接程序对象模型） 792
HelpFile 属性 792
HelpFile 属性（VBA 外接程序对象模型） 794
HelpFile 属性(App、Common Dialog、MenuLine) 794
HelpKey 属性 796
Hex 函数 797
Hidden 属性 798
Hide 事件（UserControl 对象） 799
Hide 事件（UserDocument 对象） 800
Hide 方法 801
HideSelection 属性 803
hInstance 属性 805
HitBehavior 属性 805
HitTest 事件 807
HostName 属性 810
Hour 函数 811
hPal 属性 811

HScrollBar、VScrollBar 控件 812
HScrollSmallChange 属性和 VScrollSmallChange 属性 814
hWnd 属性 814
Hyperlink 对象 818
Hyperlink 属性 819
Icon 属性 920
IconState 属性 822
ID 属性 824
IDTExtensibility 接口 824
If...Then...Else 语句 826
IIf 函数 830
Image 控件 831
Image 控件（数据报表设计器）833
Image 属性 834
Image 属性（ActiveX 控件） . 836
ImageList 属性（ActiveX 控件） 838
IMEStatus 函数 839
Implements 语句 841
Import 方法（VBA 外接程序对象模型） 846
Index 属性（控件数组） 847
IndexedValue 属性 850
IndexedValue 属性（VBA 外接程序对象模型） 851
InitDir 属性 852
Initialize 事件 853
InitProperties 事件 854
Input#语句 855
Input 函数 857
InputBox 函数 859
InSelection 属性 861
Insert 方法 862
InsertFile 方法 863
InsertLines 方法 864
InsertObjDlg 方法 865
Instancing 属性 866
InStr 函数 869
InStrRev 函数 872
Int、Fix 函数 874
Integer 数据类型 876
Interval 属性 876
InvisibleAtRuntime 属性 879

IPmt 函数 880
IRR 函数 883
IsArray 函数 885
IsBindable 属性 886
IsBroken 属性 886
IsDataSource 属性 887
IsDate 函数 888
IsDirty 属性 889
IsEmpty 函数 889
IsError 函数 890
IsMissing 函数 891
IsNull 函数 894
IsNumeric 函数 895
IsObject 函数 896
IsReady 属性 898
IsRootFolder 属性 899
Italic 属性 901
Item 方法（外接程序） 903
Item 方法 904
Item 方法（VBA 外接程序对象模型） 906
Item 属性 908
Item 属性(FileSystemObject 对象) 908
ItemActivated 事件 910
ItemAdded 事件 910
ItemAdded 事件（VBA 外接程序对象模型） 911
ItemCheck 事件 912
ItemData 属性 913
ItemReloaded 事件 916
ItemRemoved 事件 916
ItemRemoved 事件（VBA 外接程序对象模型） 917
ItemRenamed 事件 917
Items 方法 918
ItemSelected 事件 919
Join 函数 920
KeepTogether 属性 921
Key 属性 922
KeyDown、KeyUp 事件 922
KeyPress 事件 927
KeyPreview 属性 930
Keys 方法 932
Kill 语句 934
KillDoc 方法 934

Label 控件 936
Label 控件（数据报表设计器）938
LargeChange、SmallChange 属性 939
LastDLLError 属性 941
LastUsedPath 属性 943
LBound 函数 943
LBound 属性 945
LCase 函数 947
Left 函数 948
Left 属性 949
Left，Top 属性 950
LeftMargin，RightMargin 属性952
LegalCopyright 属性 952
LegalTrademarks 属性 953
Len 函数 954
Let 语句 957
LicenseInfo 对象 958
LicenseKey 属性 959
Licenses 集合 960
Licenses 属性 961
Line 控件 962
Line 控件（数据报表设计器） 962
LineInput#语句 963
Line 方法 964
Line 属性 967
Lines 方法 968
Lines 属性 969
LineSlant 属性 969
LinkClose 事件 970
LinkedWindowFrame 属性 971
LinkedWindows 集合 972
LinkedWindows 属性 973
LinkError 事件 977
LinkExecute 事件 979
LinkExecute 方法 981
LinkItem 属性 984
LinkMode 属性 988
LinkNotify 事件 990
LinkOpen 事件 990
LinkPoke 方法 991
LinkRequest 方法 993
LinkSend 方法 996
LinkTimeout 属性 997

LinkTopic 属性 9981053
List 属性 1002
ListBox 控件 1005
ListCount 属性 1006
ListIndex 属性 1009
Load 事件 1012
Load 事件(DHTMLPage) 1014
Load 语句 1014
LoadPicture 函数 1016
LoadResData 函数 1021
LoadResPicture 函数 1024
LoadResString 函数 1025
Loc 函数 1026
LocaleID 属性 1027
Lock，Unlock 语句 1028
Locked 属性 1031
LOF 函数 1033
Log 函数 1033
LogEvent 方法 1035
LogMode 属性 1036
LogPath 属性 1038
Long 数据类型 1039
LostFocus 事件 1039
LostFocus 事件（UserControl 对象和 UserDocument 对象） 1042
lpOleObject 属性 1043
LSet 语句 1043
Ltrim，RTrim 与 Trim 函数 .. 1045
MainWindow 属性 1046
Major 属性 1047
Major 属性(VBA 外接程序对象模型) 1048
MakeCompiledFile 方法 1049
MaskColor 属性 1049
MaskColor 属性（UserControl 对象） 1051
MaskPicture 属性（UserControl 对象） 1052
数学函数 1054
Max，Min 属性（公共对话框） 1054
Max、Min 属性（滚动条） ... 1056
MaxButton 属性 1058
MaxFileSize 属性 1060
MaxLength 属性 1061
MDIChild 属性 1062
MDIForm 对象 1064

Me 1066
Member 对象 1067
Members 集合 1068
Members 属性 1068
Menu 控件 1069
MessageReflect 属性 1070
Mid 函数 1071
Mid 语句 1072
MinButton 属性 1074
MinHeight 属性和 MinWidth 属性 1075
Minor 属性 1076
Minor 属性 1077
Minute 函数 1078
MIRR 函数 1078
MiscFlags 属性 1080
MkDir 语句 1082
Mode 属性 1082
Month 函数 1083
MonthName 函数 1084
MouseDown、MouseUp 事件 ... 1085
MouseIcon 属性 1090
MouseMove 事件 1093
MousePointer 属性 1098
Move 方法 1102
Move 方法（FileSystemObject 对象） 1105
Moveable 属性 1105
MoveFile 方法 1106
MoveFolder 方法 1108
MsgBox 函数 1109
MultiLine 属性 1114
MultiSelect 属性 1115
Name 属性 1117
Name 属性(FileSystemObject 对象) 1120
Name 属性（VBA 外接程序对象模型） 1122
Name 属性(WebClass、WebItem) 1123
Name 语句 1124
NavigateTo 方法 1125
NegotiateMenus 属性 1126
NegotiateMenus 属性 1128

NegotiatePosition 属性 1129
NegotiateToolbars 属性 1130
NewIndex 属性 1131
NewPage 方法 1132
NextItem 属性 1133
NonModalAllowed 属性 1135
Nothing 1135
Now 函数 1136
NPer 函数 1136
NPV 函数 1139
Null 1141
NullValue 属性 1141
Number 属性 1142
NumIndices 属性 1145
NumIndices 属性（VBA 外接程序对象模型） 1145
Object 属性 1146
Object 属性(VBA 外接程序对象模型) 1150
Object 属性（OLE 容器） ... 1151
ObjectAcceptFormats 属性 .. 1152
ObjectAcceptFormatsCount 属性 1152
ObjectEvent 事件 1154
ObjectGetFormats 属性 1155
ObjectGetFormatsCount 属性 1157
ObjectMove 事件 1159
ObjectVerbFlags 属性 1161
ObjectVerbs 属性 1162
ObjectVerbsCount 属性 1166
Oct 函数 1168
OLE 容器控件 1169
OLECompleteDrag 事件 1172
OLEDrag 方法 1174
OLEDragDrop 事件 1175
OLEDragMode 属性 1179
OLEDragOver 事件 1180
OLEDropAllowed 属性 1184
OLEDropMode 属性 1186
OLEGiveFeedback 事件 1188
OLERequestPendingMsgText 属性 1191
OLERequestPendingMsgTitle 属性 1193
OLERequestPendingTimeout 属性 1194

OLEServerBusyMsgText 属性 . 1195
OLEServerBusyMsgTitle 属性 1197
OLEServerBusyRaiseError 属性 1198
OLEServerBusyTimeout 属性 . 1200
OLESetData 事件 1201
OLEStartDrag 事件 1203
OLEType 属性 1206
OLETypeAllowed 属性 1208
OnError 语句 1209
On...GoSub、On...GoTo 语句 1215
OnAddinsUpdate 方法 1218
OnConnection 方法 1219
OnDisconnection 方法 1222
OnStartupComplete 方法 1224
Open 语句 1225
OpenAsTextStream 方法 1228
OpenTextFile 方法 1231
OptionBase 语句 1233
OptionCompare 语句 1234
OptionExplicit 语句 1236
OptionPrivate 语句 1237
OptionButton 控件 1238
Orientation 属性 1240
Orientation 属性 (CommonDialog 控件) 1241
Page 属性 1242
Paint 事件 1244
PaintPicture 方法 1247
Palette 属性 1250
PaletteMode 属性 1251
PaperBin 属性 1252
PaperSize 属性 1255
Parent 属性 1258
Parent 属性(VBA 外接程序对象模型) 1261
Parent 属性（UserControl 对象) 1262
ParentControls 集合 1263
ParentControls 属性 1263
ParentControlsType 属性 ... 1264
ParentFolder 属性 1266
Partition 函数 1267
PasswordChar 属性 1269

Paste 方法 1271
PasteOK 属性 1272
PasteSpecialDlg 方法 1274
Path 属性 1275
Path 属性（FileSystemObject 对象） 1277
PathChange 事件 1278
Pattern 属性 1280
PatternChange 事件 1282
Persistable 属性 1284
Picture 对象 1285
Picture 属性 1286
PictureAlignment 属性 1290
PictureBox 控件 1291
Pmt 函数 1294
Point 方法 1297
PopupMenu 方法 1299
Port 属性 1302
PPmt 函数 1304
PrevInstance 属性 1308
Print# 语句 1309
Print 方法 1312
Printer 对象和 Printers 集合 1315
Printer 属性 1317
PrinterDefault 属性 1318
Printers 属性 1320
PrintForm 方法 1321
PrintQuality 属性 1323
PrintReport 方法 1324
Private 语句 1326
ProcBodyLine 属性 1329
ProcCountLines 属性 1331
ProcessingTimeout 事件 1333
ProcessTag 事件 1334
ProcOfLine 属性 1335
ProcStartLine 属性 1337
ProductName 属性 1339
ProgID 属性 1340
ProgID 属性（LicenseInfo 对象） 1340
Properties 集合 1341
Properties 集合（VBA 外接对象模型） 1341
Properties 属性 1342
Properties 属性 1343

PropertyGet 语句 1343
PropertyLet 语句 1349
Property 对象 1354
PropertySet 语句 1355
PropertyBag 对象 1359
PropertyChanged 方法 1360
PropertyName 属性 1361
PropertyPage 对象 1363
PropertyPage 属性 1365
PropertyPages 属性 1365
Protection 属性 1366
PSet 方法 1367
Public 属性 1370
Public 语句 1370
Put 语句 1374
PV 函数 1376
QBColor 函数 1381
QueryClose 事件 1382
QueryUnload 事件 1384
Quit 方法（外接程序） 1388
Raise 方法 1388
RaiseEvent 语句 1391
Randomize 语句 1396
Rate 函数 1397
Read 方法 1400
ReadAll 方法 1400
ReadFromFile 方法 1401
ReadLine 方法 1402
ReadOnly 属性 1403
ReadOnlyMode 属性 1404
ReadProperties 事件 1405
Read 属性 1406
ReadProperty 函数（外接程序） 1407
ReDim 语句 1408
Reference 对象 1411
References 集合 1412
References 属性 1413
ReferencesEvents 对象 1413
ReferencesEvents 属性 1414
Refresh 方法 1416
ReleaseInstance 方法 1418
Reload 方法 1419
Rem 语句 1419

Remove 方法（用于 VisualBasic 的应用程序） 1420
Remove 方法（VBA 外接程序对象模型） 1422
Remove 方法（FileSystemObject 对象） 1423
Remove 方法（VisualBasic 可扩展性） 1424
RemoveAddInFromToolbar 方法1425
RemoveAll 方法 1426
RemoveItem 方法 1427
Render 方法 1429
Replace 函数 1431
ReplaceLine 方法 1433
ReportWidth 属性 1434
Request 属性 1435
RequestChangeFileName 事件 1436
RequestEdit 属性 1438
RequestWriteFile 事件 1438
RescanReplacements 属性 ... 1440
Reset 语句 1441
Resize 事件 1442
Respond 事件 1444
Response 属性 1445
Resume 语句 1445
Resync 方法（远程数据） ... 1447
Revision 属性 1448
RGB 函数 1449
Right 函数 1452
RightToLeft 属性 1453
RmDir 语句 1454
Rnd 函数 1455
RootFolder 属性 1456
Round 函数 1457
RowColChange 事件 1458
RowHeight 属性 1459
RSet 语句 1460
SaveAs 方法 1461
Saved 属性 1462
SavePicture 语句 1463
SaveSetting 语句 1465
SaveToFile 方法 1467
SaveToOle1File 方法 1468
Scale 方法 1469
ScaleHeight、ScaleWidth 属性147
ScaleLeft、ScaleTop 属性 .. 1475

ScaleMode 属性 1478
ScaleUnits 属性 1481
ScaleX、ScaleY 方法 1481
Scope 属性 1484
Screen 对象 1484
Screen 属性 1485
Scroll 事件 1485
ScrollBars 属性 1488
Second 函数 1490
Section 对象（数据报表设计器） 1491
Sections 集合（数据报表设计器） 1491
Sections 属性 1493
Seek 函数 1495
Seek 语句 1497
SelChange 事件 1499
SelCount 属性 1500
SelectCase 语句 1501
SelectAll 方法 1504
Selected 属性 1505
SelectedControls 集合 1506
SelectedControls 属性 1507
SelectedVBComponent 属性 .. 1508
SelectedVBControls 集合 ... 1508
SelectedVBControls 属性 ... 1509
SelectedVBControlsEvents 对象 1509
SelectedVBControlsEvents 属性 1510
SelectionChanged 事件 1511
SelEndCol、SelStartCol、SeLEndRow、SelStartRow 属性 1511
SelLength、SelStart、SelText 属性 1513
SendKeys 语句 1517
SerialNumber 属性 1521
Server 属性 1523
Session 属性 1524
Set 语句 1525
SetAttr 语句 1528
SetAutoServerSettings 方法 1529
SetData 方法 1532
SetData 方法（DataObject 对象）

..... 1534
SetFocus 方法 1536
SetFocus 方法(VBA 外接程序对象模型) 1538
SetSelection 方法 1539
SetText 方法 1540
SetViewport 方法 1542
Sgn 函数 1544
Shape 控件 1545
Shape 控件（数据报表设计器） 1546
Shape 属性 1547
ShareName 属性 1549
Shell 函数 1550
Shortcut 属性 1553
ShortName 属性 1554
ShortPath 属性 1555
Show 事件（UserControl 对象） 1556
Show 事件（UserDocument 对象） 1557
Show 方法 1558
Show 方法（VBA 外接程序对象模型） 1561
ShowColor 方法 1561
ShowFont 方法 1564
ShowGrabHandles 属性 1567
ShowHatching 属性 1568
ShowHelp 方法 1569
ShowInTaskbar 属性 1572
ShowOpen 方法 1573
ShowPrinter 方法 1576
ShowSave 方法 1579
ShowWhatsThis 方法 1582
Sin 函数 1584
Size 方法 1585
Size 属性 1586
Size 属性（字体） 1587
SizeMode 属性 1588
SizeMode 属性（RptImage 控件） 1591
Skip 方法 1592
SkipLine 方法 1592
SLN 函数 1593
Sorted 属性 1595
Source 属性 1596
Source 属性(用于 VB 的应用) 1596

SourceDoc 属性 1598
SourceFile 属性 1599
SourceItem 属性 1600
Space 函数 1602
Spc 函数 1603
Split 函数 1604
Sqr 函数 1606
StandardMethod 属性 1607
StandardSize 属性 1607
Start 事件 1609
StartLogging 方法 1610
StartMode 属性 1612
StartMode 属性 1612
StartProject 属性 1615
StartUpObject 属性 1615
StartUpPosition 属性 1616
StateManagement 属性 1617
Static 属性 1618
Static 语句 1619
Status 属性（AsyncProperty 对象） 1623
StatusCode 属性 1624
StdDataFormat 对象 1626
StdDataFormats 集合 1627
Stop 语句 1628
StrComp 函数 1629
StrConv 函数 1631
Stretch 属性 1633
StrikeThrough 属性 1636
String 函数 1638
StrReverse 函数 1639
Style 属性 1639
Sub 语句 1643
SubFolders 属性 1648
SupportsMnemonics 属性 1649
Switch 函数 1650
SYD 函数 1652
Tab 函数 1654
TabIndex 属性 1656
TabStop 属性 1659
Tag 属性 1660
TagPrefix 属性 1663
Tan 函数 1664
Target 属性 1665

TargetObject 属性 1665
TaskVisible 属性 1666
Template 属性 1667
TemplatePath 属性 1671
Terminate 事件 1672
Text 属性 1673
TextAlign 属性 1675
TextBox 控件 1676
TextBox 控件（数据报表设计器）
..... 1678
TextHeight 方法 1679
TextStream 对象 1681
TextWidth 方法 1682
ThreadID 属性 1684
Time 函数 1684
Time 语句 1685
Timer 控件 1686
Timer 事件 1686
Timer 函数 1689
TimeSerial 函数 1690
TimeValue 函数 1692
Title 属性（DataReport 对象） .
1693
Title 属性 1695
ToolboxBitmap 属性 1696
ToolTipText 属性 1697
Top 属性 1698
TopIndex 属性 1699
TopLine 属性 1701
TotalSize 属性 1702
Trace 方法 1703
TrackDefault 属性 1703
True 1705
TrueValue 属性 1705
TwipsPerPixelX、TwipsPerPixe
lY 属性 1706
类型转换函数 1707
Type 属性（Format 对象） .. 1714
Type 属性 1717
Type 属性（VBA 外接程序对象模
型） 1718
Type 属性(FileSystemObject 对
象) 1720
Type 属性（图片） 1721
Type 语句 1723
TypeName 函数 1727

UBound 函数 1729
UBound 属性 1731
UCase 函数 1732
UIDead 属性 1733
UIDefault 属性 1734
UnattendedApp 属性 1735
Underline 属性 1735
Unformat 事件 1737
Unload 事件 1738
Unload 事件(DHTMLPage) 1741
Unload 语句 1741
Update 方法 1743
Update 方法（OLE 容器） ... 1744
Updated 事件 1744
UpdateMode 属性 1745
UpdateOptions 属性 1747
URLData 属性 1748
URLFor 方法 1749
UseMaskColor 属性 1751
UseMnemonic 属性 1752
UserControl 对象 1754
UserDocument 对象 1756
UserEvent 事件 1758
UserMode 属性 1760
Val 数 1760
Validate 事件 1762
Value 属性 1765
Value 属性(VBA 外接程序对象模型) 1768
Value 属性（Format 对象） . 1769
VarType 函数 1770
VBComponent 对象 1773
VBComponents 集合 1774
VBComponents 属性 1775
VBComponentsEvents 对象 ... 1775
VBComponentsEvents 属性 ... 1776
VBControl 对象 1776
VBControlExtender 对象 1777
VBControls 集合 1779
VBControls 属性 1780
VBControlsEvents 对象 1780
VBControlsEvents 属性 1781
VBE 对象 1782
VBE 属性 1783

VBForm 对象 1784
VBNewProjects 集合 1785
VBProject 对象 1786
VBProjects 集合 1787
VBProjects 属性 1788
VBProjectsEvents 对象 1788
VBProjectsEvents 属性 1789
Verb 属性 1789
Version 属性 1790
ViewportHeight、ViewportLeFt、ViewportTop、ViewportWidth 属性 1791
Visible 属性 1792
Visible 属性（VBA 外接程序对象模型） 1795
VolumeName 属性 1796
WebClass 对象 1798
WebClassError 对象 1799
WebItem 对象 1799
WebItemProperties 对象 1800
Weekday 函数 1801
WeekdayName 函数 1803
Weight 属性 1805
WhatsThisButton 属性 1807
WhatsThisHelp 属性 1808
WhatsThisHelpID 属性 1810
WhatsThisMode 方法 1811
While...Wend 语句 1813
Width#语句 1814
Width 属性 1815
Window 对象 1816
Window 属性 1818
Windowless 属性 1818
WindowList 属性 1820
Windows 集合 1822
Windows 属性 1823
WindowState 属性 1824
WindowState 属性(VBA 外接程序对象模型) 1826
With 语句 1827
WordWrap 属性 1829
Write#语句 1832
Write 方法 1834
WriteBlankLines 方法 1835
WriteLine 方法 1836
WriteProperties 事件 1837

WriteProperty 方法 1838
WriteTemplate 方法 1839
X1、Y1、X2、Y2 属性 1840
Year 函数 1843
Zoom 属性 1843
ZOrder 方法 1845
附录 AANSI 字符集 1847
附录 B 数据类型 1851
附录 C 运算符 1862
附录 D 导出的数学函数 1897
附录 EAsc 函数 1900

#Const 指令

用来定义 VisualBasic 的条件编译常数。

语法

#**Const***constname*=*expression*

#Const 编译指令的语法具有以下几个部分：

部分	描述
constname	必需的；Variant(String)。常数；名称要遵守变量命名的约定。
expression	必需的。文字、其他的条件编译常数或包含除了 Is 以外的算术或逻辑运算符的任意组合。

说明

条件编译常数在其出现的模块中总是 Private。不可能利用#Const 指令建立 Public 编译常数。Public 编译常数只能在用户接口中建立。

在 expression 中只能使用编译常数及文字。使用一个用 Const 定义的标准常数，或者使用一个未定义的常数，都会导致错误发生。反之，用#Const 关键字定义的常数也只能用于条件编译。

不管条件编译常数在程序中的位置如何，都总是在模块级别中进行计算。

请参阅

#If...Then...#Else 指令,Const 语句

示例

本示例使用#Const 伪指令声明条件式编译器常数，以便在#If...#Else...#EndIf 构造中使用。

```
#ConstDebugVersion=1   'DebugVersion 常数在#If 块中计为 TRUE。
```

#If...Then...#Else 指令

条件编译已选择的 VisualBasic 代码块。

语法

#If*expression*Then
    statements
[#ElseIf*expression-n*Then
    [*elseifstatements*]]
[#Else
    [*elsestatements*]]
#EndIf

#If...Then...#Else 指令的语法具有以下几个部分：

部分	描述
expression	必需的。包含一个或多个条件编译常数、文字与运算符的任何表达式，其值为 True 或 False
statements	必需的。VisualBasic 程序行或编译指令，如果关联的表达式为 True，则运行它们
expression-n	可选的。由一或多个条件编译常数、文字和运算符组成的任何一个表达式，其值为 True 或 False
Elseifstatements	可选的。一个或多个程序行或编译命令，如果 expression-n 为 True，则运行它们
elsestatements	可选的。一个或多个程序行或编译命令，如果以前的 expression 或 expression-n 中没有一个为 True，则运行它们

说明

#If...Then...#Else 指令的作用与 If...Then...Else 语句相同，其差异在于#If、#Else、#ElseIf，及#EndIf 指令没有单独成行的形式，也就是说，在指令所在的那一行，不能有其他代码出现。条件编译通常用来编译不同平台上的同一个程序。也可以用来避免调试程序代码出现在可执行程序中。条件编译时被排除的程序代码在最后的可执行文件中被完全略去，所以不会对程序的大小或功能有任何影响。

无论结果如何，都要计算所有表达式。所以，在表达式中用到的所有常数都必须加以定义–任何未定义的常数都会被当作 Empty 来计算取值。

注意 OptionCompare 语句不会影响#If 及#ElseIf 语句中的表达式。条件编译指令中的表达式总是用 OptionCompareText 计算值。

请参阅

#Const 指令，If...Then...Else 语句

示例

本示例在#If...Then...#Else 构造中引用条件式编译器常数，来决定编译那部分语句。

```
'如果 Mac 常数为 True，则编译#If 后面的语句。
#IfMacThen
    '.将 Mac 语句写在此处。
    '.
    '.
'否则，如果是 32 位窗口程序，则编译这个语句：
#ElseIfWin32Then
    '.将 32 位窗口程序语句写在此处。
    '.
    '.
'再否则，则编译以下语句：
#Else
    '.将其他作业平台适用的语句写在此处。
    '.
    '.
#EndIf
```

Abs 函数

返回参数的绝对值，其类型和参数相同。

语法

Abs(*number*)

必要的 number 参数是任何有效的数值表达式，如果 number 包含 Null，则返回 Null，如果 number 是未初始化的变量，则返回 0。

说明

一个数的绝对值是将正负号去掉以后的值。例如，ABS(-1)和 ABS(1)都返回 1。

请参阅

Sgn 函数

示例

本示例使用 Abs 函数计算数的绝对值。

```
DimMyNumber
MyNumber=Abs(50.3) '返回 50.3。
MyNumber=Abs(-50.3)     '返回 50.3。
```

AccessKeyPress 事件

下列情况下发生该事件：按下控件的某个访问键，或者将 Default 属性设置为 True 后按下 ENTER 键，或者将 Cancel 属性设置为 True

后按下 ESCAPE 键。如果将控件的 DefaultCancel 属性设置为 True，则使 Default 属性和 Cancel 属性成为可用。

应用于

UserControl 对象

语法

Sub*object*_AccessKeyPress(*KeyAscii*AsInteger)

AccessKeyPress 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
keyAscii	整数，包含引发 AccessKeyPress 事件的按键（除 ALT 键外）的 Ascii 值，与标准的 KeyPress 事件采用的方法相同

AccessKeys 属性

返回或设置字符串，其中包含那些作为控件访问键（或热键）的按键。

应用于

UserControl 对象

语法

object.AccessKeys[=AccessKeyString]

AccessKeys 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
accessKeyString	字符串，其中包含作为访问键的那些键

说明

AccessKeys 属性是一个字符串，它包含控件所有的访问键。例如，若要将字母 S 和 Y 设置为访问键，则应把 AccessKeys 属性设置为“sy”。

按 ALT 键的同时再按下某个访问键，控件将获得焦点。(取决于 ForwardFocus 属性的设置)。

虽然子控件的访问键不出现在 AccessKeys 属性中，但它们隐含作为 AccessKeys

请参阅

ForwardFocus 属性

Action 属性（“通用”对话框)

返回或设置被显示的对话框的类型。在设计时无效。

注意此 Action 属性是为了与 VisualBasic 早期版本兼容而提供的。如需附加功能，可使用下列新方法：ShowColor，ShowFont，ShowHelp，ShowOpen，ShowPrinter 和 ShowSave。

应用于

CommonDialog 控件

语法

object.**Action**[=*value*]

Action 属性语法有下列三部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	数值表达式，如“设置值”中所描述，用以指定所显示对话框的类型

设置值

用于 value **的设置值是：**

设置	描述
0	没有操作。
1	显示“打开”对话框。
2	显示“另存为”对话框。
3	显示“颜色”对话框。
4	显示“字体”对话框。
5	显示“打印”对话框。
6	运行 WINHLP32.EXE。

数据类型

Integer

请参阅

CommonDialog 控件，ShowColor 方法，ShowFont 方法，ShowHelp 方法，ShowOpen 方法，ShowPrinter 方法，ShowSave 方法

Action 属性（OLE 容器）

设置一个确定操作的值。在设计时不可用。

注意包含 SourceDoc 属性是为了与早期版本的 Action 属性兼容。要获得目前的该功能，可使用设置值中所列的方法。

应用于

OLE 容器控件

语法

object.**Action**=*value*

Action 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	常数或整数，指定操作的类型，在“设置值”中有详细说明

设置值

value 的设置值是：

值	描述	当前的方法
0	创建内嵌对象。	CreateEmbed
1	从文件的内容中创建链接对象。	CreateLink
4	将对象复制到系统剪贴板。	Copy
5	将对象从系统剪贴板复制到 OLE 容器控件。	Paste
6	从支持对象的应用程序检索当前数据，并在 OLE 容器控件中将该数据作为图片显示。	Update
7	打开一个对象，用于进行诸如编辑那样的操作。	DoVerb
9	关闭对象，并与提供该对象的应用程序终止连接。	Close
10	将指定的对象删除，释放与之关联的内存。	Delete
11	将对象保存到数据文件中。	SaveToFile
12	加载保存到数据文件中的对象。	ReadFromFile
14	显示插入对象对话框。	InsertObjDlg
15	显示特殊粘贴对话框。	PasteSpecialDlg
17	更新对象支持的谓词列表。	FetchVerbs
18	将对象以 OLEversion1.0 版本的文件格式	SaveToOle1File

保存。

请参阅

PasteSpeciaDlg 方法,Copy 方法,CreateEmbed 方法,CreateLink 方法，Delete 方法（OLE 容器），InsertObjDlg 方法，Paste 方法，ReadFromFile 方法，SaveToFile 方法，DoVerb 方法，FetchVerbs 方法，Close 方法(OLE 容器)，Update 方法(OLE 容器),SaveToOle1File 方法，FetchVerbs 方法（ActiveX 控件）

Activate 方法

该方法可激活工程窗口中当前选中的部件，如同双击它一样。

应用于

VBComponent 对象

语法

object.**Activate**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Activate、Deactivate 事件

Activate－当一个对象成为活动窗口时发生。

Deactivate－当一个对象不再是活动窗口时发生。

应用于

DataReport 对象，Form 对象，Forms 集，MDIForm 对象

语法

PrivateSub*object*_**Activate**()

PrivateSub*object*_**Deactivate**()

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

一个对象可以通过诸如单击它，或使用代码中的 Show 或 SetFocus 方法之类的用户操作而变成活动的。

Activate 事件仅当一个对象可见时才发生。例如，除非使用 Show 方法或将窗体的 Visible 属性设置为 True，否则，一个用 Load 语句加载的窗体是不可见的。

Activate 和 Deactivate 事件仅当焦点在一个应用程序内移动时才发生。在另一个应用程序中将焦点移向或移离一个对象时，不会触发任何一个事件。当一个对象卸载时，不会发生 Deactivate 事件。

Activate 事件在 GotFocus 事件之前发生，LostFocus 事件在 Deactivate 事件之前发生。

对 MDI 子窗体来说，这些事件仅当焦点从一个子窗体改变到另一个子窗体时才会发生。例如，在一个带有两个子窗体的 MDIForm 对象中，当焦点在子窗体之间移动时，它们能接收这些事件。然而，当焦点在一个 MDI 子窗体和一个非 MDI 子窗体之间移动时，父 MDIForm 将接收 Activate 和 Deactivate 事件。

如果一个由 VisualBasic 建立的.exe 文件显示一个对话框，此

对话框也是由建立在 VisualBasic 中的一个.dll 文件所创建的，那么.exe 文件的窗体将获得 Deactivate 和 LostFocus 事件。这可能是不期望的，因为下列情况下不应获得 Deactivate 事件：

如果对象是一个过程之外的部件。

如果对象不是用 VisualBasic 编写的。

在开发环境中调用一个 VisualBasic 建立中的 DLL 时。

请参阅

GotFocus 事件，LostFocus 事件，SetFocus 事件，Show 方法，Visible 属性

示例

本例修改状态栏中正文，显示活动窗体标题。在本例中，创建一个 Form 对象（Form1）和一个新的 MDIForm 对象（MDIForm1）。在 MDIForm1 上画一个包含 Lobel 控件的 PictureBox 控件。在 Form1 上，将 MPIChild 属性设置为 True。粘贴 MDIForm_Load 事件代码到 MDIForm 对象的声明段。粘贴 Form_Activate 事件过程代码到 MDI 子窗体的声明段，然后按 F5。

ActiveCodePane 属性

返回当前活动的或上一次活动的 CodePane 对象，或设置活动的 CodePane 对象。可读/写。

应用于

VBE 对象

说明

可以将 ActiveCodePane 属性设置为任何有效 CodePane 对象，如下例所示：

```
SetMyApp.VBE.ActiveCodePane=MyApp.VBE.CodePanes(1)
```

上例将代码窗格集合中的第一个代码窗格设置为活动代码窗格。也可以用 Set 方法来激活一个代码窗格

请参阅

CodePane 对象，CodePanes 集合，ActiveWindow 属性，CodePaneView 属性

示例

下面示例使用 ActiveCodePane 属性和 TopLine 属性获得活动代码窗口中顶行的数量。

```
Debug.PrintApplicationVBE.ActiveCodePane.TopLine
```

ActiveControl 属性

返回拥有焦点的控件。当窗体被引用时，如在 ChildForm.ActiveControl 中，如果被引用的窗体是活动的，ActiveControl 指定将拥有焦点的控件。在设计时是不可用的；在运行时是只读的。

应用于

PropertyPage 对象,UserControl 对象,UserDocumnt 对象,Screen

对象，Form 对象，Forms 集合，MDIForm 对象

语法

object.**ActiveControl**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

可以使用 ActiveControl 来访问控件的属性或调用其方法：例如，Screen.ActiveControl.Tag="0"。如果在窗体上的所有控件都是不可见的或不可使用的，那么将产生一个运行时错误。

每个窗体都可以有一个活动控件(Form.ActiveControl)，而不管窗体是否是活动的。在应用程序中，可以在每个窗体上编写处理活动控件的代码，即使此窗体不是活动窗体。

这个属性在多文档接口(MDI)应用程序中尤其有用，其中工具栏上的一个按钮必须初始化为 MDI 子窗体中的控件上的一个动作。当用户单击工具栏上的“复制”按钮时，代码可以引用 MDI 子窗体的活动控件中的文本，例如 ActiveForm.ActiveControl.SelText。

注意如果计划将 Screen.ActiveControl 传递给一个过程，那么在那个过程中必须用子句 AsControl 而不是指定控件的类型（AsTextBox 或 AsListBox）声明参数，即使 ActiveControl 总是引用相同类型的控件。

请参阅

GotFocus 事件，LostFocus 事件，ActiveForm 属性

示例

这个例子显示活动控件的文本。要试用此例，可以先将下面的代码粘贴到包含 TextBox、Label 和 CommandButton 控件的窗体的声明部分中，然后按下 F5 键并单击此窗体。

```
PrivateSubForm_Click()
    IfTypeOfScreen.ActiveControlIsTextBoxThen
        Label1.Caption=Screen.ActiveControl.Text
    Else
        Label1.Caption="Button:"+Screen.ActiveControl.Caption
    EndIf
EndSub
```

这个例子显示了如何使用工具栏上的按钮在实现剪切、复制、粘贴和删除操作中，利用 Clipboard 对象。要试一试这个例子，可以将 TextBox 和 CheckBox 控件放到 Form1 上，然后创建一个新的 MDI 窗体。在 MDI 窗体上，插入一个 PictureBox 控件，然后在 PictureBox 中插入一个CommandButton。将CommandButton 的Index 属性设置为 0（创建一个控件数组）。将 Form1 的 MDIChild 属性设置为 True。

要运行该例子，只须将这些代码复制到 MDIForm 的声明部分，然后按下 F5 键。注意，当 CheckBox 拥有焦点时，按钮将不可用，因为现在 CheckBox 取代了 TextBox 而成为活动控件。

```
PrivateSubMDIForm_Load()
```

```
    DimI'声明变量。
    Command1(0).Move0,0,700,300 '在工具栏上为按钮定位。
    ForI=1To3    '创建别的按钮。
        LoadCommand1(I) '创建按钮。
        Command1(I).MoveI*700,0,700,300'放置并调整按钮大小。
        Command1(I).Visible=True'显示按钮。
    NextI
    Command1(0).Caption="Cut"    '设置按钮标题。
    Command1(1).Caption="Copy"
    Command1(2).Caption="Paste"
    Command1(3).Caption="Del"
EndSub

PrivateSubCommand1_Click(IndexAsInteger)
    'ActiveForm 是指 MDI 窗体中的活动窗体。
    IfTypeOfActiveForm.ActiveControlIsTextBoxThen
        SelectCaseIndex
            Case0    '剪切。
                '复制选中的文本到剪贴板上。
                Clipboard.SetTextActiveForm.ActiveControl.SelText
                '删除选中的文本。
                ActiveForm.ActiveControl.SelText=""
```

```
            Case1    '复制。
                '复制选中的文本到剪贴板上。
                Clipboard.SetTextActiveForm.ActiveControl.SelText
            Case2    '粘贴。
                '把剪贴板上的文本放到文本框中。

    ActiveForm.ActiveControl.SelText=lipboard.GetText()
            Case3    '删除。
                '删除选中的文本。
                ActiveForm.ActiveControl.SelText=""
        EndSelect
    EndIf
EndSub
```

ActiveForm 属性

返回活动窗口的窗体。如果 MDIForm 对象是活动的或者是被引用的，则所指定的是活动的 MDI 子窗体。

应用于

Screen 对象，From 对象，Froms 集合，MDIorm 对象

语法

object.**ActiveForm**

object 所在处代表一个对象表达式，其值是“应用于”列表中的

一个对象。

说明

为了访问窗体的属性或者调用其方法需使用 ActiveForm 属性一例如，Screen.ActiveForm.MousePointer=4。

这个属性在多文档接口(MDI)应用程序中尤其有用，其中，工具条上的一个按钮必须初始化为 MDI 子窗体中控件的一个动作。当用户单击工具条上的“复制”按钮时，代码可以引用 MDI 子窗体上的活动控件中的文本一例如,ActiveForm.ActiveControl.SelText。当窗体上的控件拥有焦点时，该窗体就是屏幕上的活动窗体(Screen.ActiveForm)。另外，一个 MDIForm 对象能够包含一个在 MDI 父窗体(MDIForm.ActiveForm)的上下文中是活动窗体的子窗体。屏幕上的 ActiveForm 不必与 MDI 窗体中的 ActiveForm 一致，比如当对话框为活动时。由于这个原因，当对话框有机会成为 ActiveForm 的属性设置时，用 ActiveForm 指定 MDIForm。

注意当一个活动的 MDI 子窗体没有被最大化时，父窗体和子窗体的标题栏都显示为活动的。

如果打算将 Screen.ActiveForm 或 MDIForm.ActiveForm 传递给一个过程，必须用类属的类型(AsForm)而不是具体的窗体类型(AsMyForm)来声明那个过程中的参数，即使 ActiveForm 总是引用相同类型的窗体。

ActiveForm 属性为 ProjectTemplate 对象确定缺省值。

请参阅

Activate,Deactivate 事件，MDIChild 属性

示例

该例子在 MDIForm 对象中的活动子窗体上打印时间。要试用此例，先创建一个 MDIForm，然后在其上添加一个 PictureBox 控件并在 PictureBox 控件中添加一个 CommandButton 控件。在 Form1 中，把 MDIChild 属性设置为 True。（你也可以把 AutoRedraw 设置为 True，这样即使在别的窗体覆盖了它后，仍可在窗体上保持文本。）将相应的代码粘贴到每个窗体的声明部分，然后按下 F5 键。

```
'将所有的代码复制到 MDI 窗体中。
PrivateSubMDIForm_Load()
    DimNewFormAsNewForm1'创建 Form1 的新实例。
    NewForm.Show
EndSub

PrivateSubCommand1_Click()
    '在活动窗体上打印时间。
    ActiveForm.Print"Thetimeis"&Format(Now,"LongTime")
EndSub
```

这个例子显示了如何使用工具栏上的按钮在实现剪切、复制、粘贴和删除操作中，利用 Clipboard 对象。要试用此例，先创建一个新的工程，在 Form1 上放置 TextBox 和 CheckBox 控件，然后创建一个新的 MDI 窗体。在 MDI 窗体上，放置一个 PictureBox 控件，

然后在PictureBox中插入一个CommandButton控件。将CommandButton的Index属性设为0（创建一个控件数组）。将Form1的MDIChild属性设为True。

要运行该例子，将该代码复制到MDIForm的声明部分，然后按下F5键。注意，当CheckBox拥有焦点时，按钮将不工作，因为CheckBox现在代替了TextBox而成为活动控件。

```
PrivateSubMDIForm_Load()
   DimI          '声明变量。
   Command1(0).Move0,0,700,300  '在工具栏上定位按钮。
   ForI=1To3    '创建别的按钮。
       LoadCommand1(I)  '创建按钮。
       Command1(I).MoveI*700,0,700,300'放置并调整按钮的尺寸。
       Command1(I).Visible=True'显示按钮。
   NextI
   Command1(0).Caption="Cut"     '设置按钮标题。
   Command1(1).Caption="Copy"
   Command1(2).Caption="Paste"
   Command1(3).Caption="Del"
EndSub

PrivateSubCommand1_Click(IndexAsInteger)
   'ActiveForm指的是MDI窗体中的活动窗体。
```

```
    IfTypeOfActiveForm.ActiveControlIsTextBoxThen
        SelectCaseIndex
            Case0    '剪切。
                '复制选中的文本到剪贴板。
                Clipboard.SetTextActiveForm.ActiveControl.SelText
                '删除选中的文本。
                ActiveForm.ActiveControl.SelText=""
            Case1    '复制。
                '复制选中的文本到剪贴板。
                Clipboard.SetTextActiveForm.ActiveControl.SelText
            Case2    '粘贴。
                '将剪贴板上的文本放到文本框中。

    ActiveForm.ActiveControl.SelText=Clipboard.GetText()
            Case3    '删除。
                '删除选中的文本。
                ActiveForm.ActiveControl.SelText=""
        EndSelect
    EndIf
EndSub
```

ActiveVBProject 属性

返回在工程窗口中的活动工程，此属性为只读。

应用于

VBE 对象

说明

ActiveVBProject 属性返回“工程”窗口中选定的工程，或部件被选定的工程。对后者来说，工程本身并不须要被选定。无论此工程是否被显式地选定，都有一个工程是活动的。

请参阅

VBProject 对象，VBProjects 集合

示例

下例使用 ActiveVBProject 属性返回活动项目名。

```
Debug.PrintApplicationVBE.ActiveVBProject.Name
```

ActiveWindow 属性

返回开发环境中活动的窗口，此属性为只读。

应用于

VBE 对象

说明

当开发环境中有一个以上的窗口被打开时，ActiveWindow 属性的设置值是拥有焦点的那个窗口。如果拥有焦点的是主窗口，则

ActiveWindow 属性返回 Nothing。

请参阅

SetFocus 方法，Window 对象，ActiveCodePane 属性，MainWindow 属性

Add 方法（Add-In）

ContainedVBControls 集合：给 ContainedVBControls 集合添加一个新的 VBControl 对象。

VBControls 集合：给 VBControls 集合添加一个新的 VBControl 对象。

VBProjects 集合：给 VBProjects 集合中的工程集添加一个新的空值工程。

应用于

ContainedVBControls 集合，VBControls 集合

语法

object.**Add**(*progid***As String**,[*relativevbcontrol* **As VBControl**][*before***As Boolean**])**As VBControl**

object.**Add**(*projecttype***As** vbext_ProjectType,[exclusive As Boolean]) As VBProject

Add 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
progid	必需的。字符串表达式，指定被添加部件的 ProgID
relativevbcontrol	可选的。指定新部件插入点位置的已有 VBControl 对象
before	可选的。缺省值＝False。布尔表达式，指定是在 relativevbcontrol 之前还是在其后放置新 VBControl
projecttype	必需的。指定新工程类型的 VBProject 对象。对于工程种类的列表，请参阅 Kind 属性
exclusive	可选的。缺省值＝False。布尔表达式，指定是在已存在的工程集中添加一个新工程，还是作为唯一的工程被添加

说明

如果指定 exclusive 参数为 True，则关闭已存在的组工程，而且新工程变成集合中唯一的工程。

Add 方法(BindingCollection)

添加一个 Binding 对象到 BindingCollection 对象。

应用于

BindingCollection 对象

语法

object.Add(*object*,*PropertyName*,Data*Field*,Data*Format*,*Key*)

Add 方法的语法包含如下部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
object	必需的。控件或其它要绑定的数据使用者
PropertyName	必需的。数据字段要绑定到的数据使用者的属性
dataField	必需的。将被绑定到 PropertyName 参数所指定属性的数据源列
dataFormat	可选的。一个 DataFormat 对象或一个对 DataFormat 变量的引用，将用于格式化绑定属性
key	可选的。一个唯一的字符串，标识集合成员

说明

Binding 对象代表一个绑定到数据源数据字段的对象的属性。使用 BindingCollection 对象绑定没有设计时界面的数据源，如配置为数据源的 Class，到一个数据使用者。也可以使用 Binding 对象把一个 OLESimpleProvider(OSP)数据源绑定到一个数据使用者。

不能使用 Binding 对象把一个复杂绑定控件（例如 DataGrid 控件）绑定到一个数据源。而仅需要把该控件的 DataSource 属性设置为该数据源即可。

示例

本例使用 BindingCollection 对象把一个数据源绑定到两个 TextBox 控件。首先打开一个 ADODB 记录集对象，然后设置 BindingCollection 的 DataSource 属性为该记录集。随后，程序代码把两个 Binding 对象添加到该集合，同时也就把这两个 TextBox 控件绑定到该记录集的不同字段了。

要试验该例，在“引用...”对话框设置对 MicrosoftDataBindingCollection 的引用。在同一个对话框中，设置对 MicrosoftActiveXDataObjectsLibrary 的引用。在窗体上绘制两个 TextBox 控件，把代码粘贴到“声明”部分。按 F5 键，并单击窗体移动到记录集的下一条记录。

```
OptionExplicit
PrivatecolBndNwindAsNewBindingCollection
PrivatersNwindAsNewADODB.Recordset
PrivatecnAsNewADODB.Connection

PrivateSubForm_Load()

    '设置 Connection 对象参数。
    Withcn
        '下列的连接在您的计算机上可能能够正常工作，也可能不能正常工作。
        '请改变它以定位 Nwind.mdb 文件。
```

```
        '该文件包括在 VisualBasic 中。
        .Provider="Microsoft.Jet.OLEDB.3.51"
        .Open"C:\ProgramFiles\DevStudio\VB\Nwind.mdb"
EndWith

'打开该记录集对象。
rsNwind.Open"Select*FromProducts",cn

'设置 Bindings 集合的 DataSource 为该记录集。
SetcolBndNwind.DataSource=rsNwind

'添加到 Bindings 集合。
WithcolBndNwind
        .AddText1,"Text","ProductName",,"product"
        .AddText2,"Text","SupplierID",,"ID"
EndWith

'打印集合中对象的属性。
DimbndObjAsBinding
ForEachbndObjIncolBndNwind
        Debug.Print"DataField","PropertyName","Key"

Debug.PrintbndObj.DataField,bndObj.PropertyName,bndObj.Key
```

```
        Debug.Print
    Next
EndSub

PrivateSubForm_Click()
    '单击窗体移动到下一条记录。
    rsNwind.MoveNext
EndSub
```

Add 方法（Controls 集合）

在 Controls 集合中添加一个控件并返回一个对该控件的引用。

语法

object.**Add**(*ProgID*,*name*,*container*)

Add **方法语法有这些部分:**

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
ProgID	必需的。一个标识控件的字符串。大多数控件的 ProgID 都可通过查看对象浏览器来决定。控件的 ProgID 是由控件的库和类组成的。例如，CommandButton 控件的 ProgID 是 VB.CommandButton。在 ProgID 与对象浏览器中所显示的不一样的情况下，VisualBasic 将显示一个包括正确 ProgId 的错误信息
name	必要的。一个字符串，用来标识集合的成员
container	可选的。一个对象引用，它指定控件的容器。如果没有指定或为 NULL，缺省值为 Controls 集合所属的容器。通过指定该参数，可以把一个控件放置在任何现存的容器控件（如 Frame 控件）中。用户控件或 ActiveX 文档也可以作为一个容器

说明

注意 Controls 集合是后期绑定的集合。这意味着编译器不能预先决定集合以及它们的对象或它们的界面包含哪些控件。没有这些信息，自动语句生成器不能工作。

该方法允许您在运行时向应用程序中添加控件。即使在应用程序已经被编译与部署之后，动态控件添加也可以用来向应用程序添

加一个控件的功能。例如，您可能有几个复杂的用户控件，每个适合于一个不同的任务。由于外部因素不同，如时间或日期或用户输入，不同的用户控件可以被添加到一个应用程序中的某个现存窗体。您也可以利用这个方法的 container 参数来指定一个容器控件（如 Frame 控件）来放置这个控件。或您可以设计一个自动读取文件、数据库或注册表项来加载新控件的应用程序。用这种方法，您可以修改一个应用程序而不必重新部署它。

重点当您添加一个未引用的需要许可证的控件到一个现存的（已部署好的）应用程序时，在使用 Add 方法之前您必须也添加这个控件的许可证关键字。关于什么时候和如何添加许可证有关信息，请参阅“请参阅”列表中的“Licenses 集合”。

在运行时添加未引用的控件

您也可以利用 Add 方法来动态添加一个在工程中没有被引用的控件。(“未引用的”控件是不出现在 Toolbox 中的控件)。为此，您必须也把控件的 License 关键字添加到 Licenses 集合中。下面的示例中在添加控件本身之前添加了控件的许可证关键字：

```
OptionExplicit
PrivateWithEventsextCtlAsVBControlExtender

PrivateSubForm_Load()
   Licenses.Add"prjWeeks.WeeksCtl","xydsfasfjewfe"
```

```
    SetextCtl=Form1.Controls.Add("prjWeeks.WeeksCtl","ctl1")
    extCtl.Visible=True'Thecontrolisinvisiblebydefault.
EndSub
```

注意请参阅关于检索控件许可证关键字的详细信息的“请参阅”列表中的“增加方法(Licenses 集合)”。
但是，为了编程这样一个未引用控件的事件，您必须使用 WithEvents 关键字声明一个对象变量为 VBControlExtender 对象(如上)，并且设置该对象变量到 Add 方法返回的引用上。然后，利用 VBControlExtender 对象的 ObjectEvent 事件来编程该控件的事件。下面是一个简单的例子。

```
OptionExplicit
DimWithEventsobjExtAsVBControlExtender'声明 Extender 变量

PrivateSubLoadControl()
    Licenses.Add"Project1.Control1","xydsfasfjewfe"
    SetobjExt=Controls.Add("Project1.Control1","myCtl")
    objExt.Visible=True
EndSub

PrivateSubextObj_ObjectEvent(InfoAsEventInfo)
    '使用 SelectCase 编程控件的事件。
    SelectCaseInfo.Name
```

```
    Case"Click"
        '这里处理 Click 事件。
    '现在显示其他的 case
    CaseElse'未知事件
        '这里处理未知事件。
    EndSelect
EndSub
```

Note 不能把一个固有的控件指定给这个 VBControlExtender 变量;任何这种试图将引起类型不匹配错误。

但是，您也可以通过使用 WithEvents 关键字声明一个对象变量，并且设置该方法返回的引用为该变量，从而编程一个动态添加控件的事件，如下所示。

```
OptionExplicit
'声明对象变量为 CommandButton。
PrivateWithEventscmdObjectAsCommandButton

PrivateSubForm_Load()
    SetcmdObject=Form1.Controls.Add("VB.CommandButton","cmdOne")
    cmdObject.Visible=True
    cmdObject.Caption="DynamicCommandButton"
EndSub
```

```
PrivateSubcmdObject_Click()
Print"Thisisadynamicallyaddedcontrol"
EndSub
```

如果希望添加一个用户控件或任何 ActiveX 控件到您的窗体，必须或者把这个控件添加到“工具箱”，或者把控件的 License 关键字添加到 Licenses 集合中。有关详细信息请参阅“增加方法 (Licenses 集合)”。

注意如果您添加一个 ActiveX 或用户控件到您的工程，但是没有在窗体中使用它，您也必须不要选定“工程属性”对话框的“生成”选项卡上的“删除有关未使用的 ActiveX 控件”选项。如果您的应用程序试图添加该控件，那么该 Add 方法将失败，因为必需的信息已经被丢弃。

删除控件

要删除动态添加的控件，用“移除”方法。请注意，您只能删除那些用“添加”方法添加的控件（与用“Load”语句添加的控件进行对比)下面的示例删除一个动态添加的控件：

```
Form1.Controls.Remove"ctl1"'该控件的名称为 ctl1。
```

Add 方法（DataMembers 集合）

添加一个数据成员到 DataMembers 集合。

应用于

DataMembers 集合

语法

object.**Add**(Data*Member*)

Add 方法的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
dataMember	必需的。一个唯一的字符串，可以用来标识一个特定的数据成员

说明

数据成员可以是 ADO 记录集、或一个用 OSP 类和 Implements 语句实现了 OLESimpleProvider(OSP) 接口的对象，也可以是用 VisualBasic 创建的一个 OLEDB 供应程序。

示例

本例使用一个类模块作为数据源。当有关设置两个 Binding 对象的 DataSource 和 DataMember 属性的代码执行时，该类模块的 Initialize 事件发生；两个 ADO 记录集将在该事件中被创建，并且记录集的名称被添加到 DataMembers 集合中。GetDataMember

事件和它的参数用来返回数据给数据使用者。

要试验本例，在“工程”菜单中单击“引用...”，并设置对 MicrosoftDataBindingCollection 和 MicrosoftActiveXDataObjects 的引用。在“工程”菜单上单击 AddClassModule。把类名更改为 MyDataClass，并且设置 DataSourceBehavior 属性为 vbDataSource。然后，在窗体上绘制两个 TextBox 控件，将下列代码粘贴到 Form 对象的代码模块中。

```
OptionExplicit
'声明对象变量，一个是名为 MyDataClass 的类模块，
'另外两个是 BindingCollection 对象，
'每个记录集一个。

PrivateclsDataAsNewMyDataClass'类模块。
PrivatebndColProductsAsNewBindingCollection'绑定集合。
PrivatebndColSuppliersAsNewBindingCollection'绑定集合。

PrivateSubForm_Load()
   '为每个绑定集合对象设置 DataSource 和 DataMember 属性。
   WithbndColProducts
        .DataMember="Products"
        Set.DataSource=clsData
        .AddText1,"Text","ProductName"'绑定到一个 TextBox。
   EndWith
```

```
    WithbndColSuppliers
        .DataMember="Suppliers"
        Set.DataSource=clsData
        .AddText2,"Text","CompanyName"'绑定到一个 TextBox。
    EndWith

    '更改 Command1 的标题
    Command1.Caption="MoveNext"

EndSub

PrivateSubCommand1_Click()
    clsData.MoveNext
EndSub
```

把下面的代码粘贴到 MyDataClass 模块。为了看到 GetDataMember 事件，必须把 DataSourceBehavior 属性设置为 vbDataSource。运行该工程。

```
OptionExplicit
'为 ADORecordset 和 Connection 对象声明对象变量。
PrivateWithEventsrsProductsAsADODB.Recordset
PrivateWithEventsrsSuppliersAsADODB.Recordset
PrivatecnNwindAsADODB.Connection
```

```
PrivateSubClass_Initialize()
    '添加字符串到DataMembers 集合。
    WithDataMembers
        .Add"Products"
        .Add"Suppliers"
    EndWith

    '设置 Recordset 对象。
    SetrsProducts=NewADODB.Recordset
    SetrsSuppliers=NewADODB.Recordset
    SetcnNwind=NewADODB.Connection

    '设置 Connection 对象的参数。
    WithcnNwind
        '随 VisualBasic 发行的 Nwind.mdb 必须安装到
        '计算机上，否则本代码将失败。另外，改变路径
        '以找到计算机中的该文件。
        .Provider="Microsoft.Jet.OLEDB.3.51"
        .Open"C:\ProgramFiles\DevStudio\VB\Nwind.mdb"
    EndWith

    '打开 recordset 对象。
```

```
    rsSuppliers.Open"SELECT*FROMSuppliers",cnNwind,_
    adOpenStatic,adLockOptimistic
    rsProducts.Open"SELECT*FROMProducts",cnNwind,_
    adOpenStatic,adLockOptimistic

EndSub

'当一个数据使用者的DataSource属性设置时，GetDataMember发生。
'在这种情况下，绑定的集合对象是该使用者。
PrivateSubClass_GetDataMember(DataMemberAsString,DataAsObject)
    SelectCaseDataMember
    Case"Products"
        SetData=rsProducts
    Case"Suppliers"
        SetData=rsSuppliers
    Case""
        '当没有指定数据成员时提供一个缺省的记录源。
        SetData=rsProducts
    EndSelect

EndSub

PublicFunctionMoveNext()
```

```
    IfrsProducts.EOFThen
        rsProducts.MoveFirst
    Else
        rsProducts.MoveNext
    EndIf
EndFunction

PrivateSubrsProducts_MoveComplete(ByValadReasonAs_
ADODB.EventReasonEnum,ByValpErrorAsADODB.Error,adStatusAs_
ADODB.EventStatusEnum,ByValpRecordsetAsADODB.Recordset)
    '保持两个记录集同步。第一个文本框显示
    '该产品的供货商。如果两个记录集的SupplierID
    '是相等的，则不需要更改。否则，
    '移到第一条记录，并且测试SupplierID。
    '本例只供演示，因为它并不是最有效的。

    IfrsSuppliers("SupplierID").Value=_
    pRecordset("SupplierID").ValueThenExitSub

    rsSuppliers.MoveFirst
    DoWhileNotrsSuppliers.EOF
        IfrsSuppliers("SupplierID").Value=_
        pRecordset("SupplierID").ValueThen
```

```
            ExitSub
        Else
            rsSuppliers.MoveNext
        EndIf
    Loop
EndSub
```

Add 方法（DataObjectFiles 集合）

添加一个文件名到 DataObject **对象的** Files **集合。**

应用于

DataObjectFiles 集合

语法

object.**Add**(*filename*,[*index*])

Add 方法的语法包含如下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
filename	必需的。一个设置文件名的字符串
index	可选的。一个整数，指定在集合中插入文件名的位置。如果不指定索引值，文件名将被添加到集合的末尾

说明

Files 集合只能用 vbCFFiles 类型（如在对象浏览器的 ClipBoardConstants 列表中找到的）的文件名填充。然而，DataObject 对象本身可以包含多种不同类型的数据。要检索文件名的列表，请循环遍历 Files 集合。

Add 方法(Dictionary)

添加一对相对应的关键字和条目到 Dictionary 对象。

应用于

Dictionary 对象

语法

object.**Add***key*,*item*

Add 方法的语法有如下几部分：

部分	描述
object	必需的。一个 Dictionary 对象的名字
key	必需的。与所添加的条目相关联的关键字
item	必需的。与所添加的关键字相关联的条目

说明

如果该关键字已经存在，则产生一个错误。

请参阅

AddFolders 方法，Exists 方法，Items 方法，Keys 方法，Remove 方法（FileSystemObject 对象），RemoveAll 方法

Add 方法（ExportFormats 集合）

添加 ExportFormat 对象到 ExportFormats 集合，并返回一个对新创建对象的引用。

应用于

ExportFormats 集合

语法

object.**Add**(*Key*,*FormatType*,*FileFormatString*,File*Filter*,Tem*plate*)

Add 方法的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
key	必需的。一个标识集合成员的唯一的字符串
formatType	必需的。设置对象的报表类型，如在设置值中所示
FileFormatStri ng	必需的。设置显示在“导出”对话框的“另存为...”组合框中的文本
fileFilter	必需的。如果用户选择‘导出’对话框中的 ExportFormat 对象，设置要使用的文件扩展名。如果用到多个文件过滤器，第一个将作为缺省的扩展名
template	可选的。为报表设置要使用的模板

设置值

对 *FormatType* 的设置如下：

常数	值	描述
rptFmtHTML	0	HTML
rptFmtText	1	文本
rptFmtUnicodeTe xt	2	Unicode
RptFmtUnicodeH TMl_UTF8	3	使用通用字符集（UTF–8）编码的 HTML

返回类型

ExportFormat 对象

说明

如果不指定模板，VisualBasic 提供一个适合 FormatType 的缺省模板。

请参阅

DataReport 对象，ExportFormat 对象，ExportFormats 属性，FormatType 属性，Template 属性

示例

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
   DataReport1.Title="DailyReport"'本标题出现在报表中。
   DimstrTemplateAsString
```

```
'创建模板
strTemplate=_
"<HTML>"&vbCrLf&_
"<HEAD>"&vbCrLf&_
"<TITLE>"&"MyCompany:"&rptTagTitle&_
"</TITLE>"&vbCrLf&_
"<BODY>"&vbCrLf&_
rptTagBody&vbCrLf&_
"<BODY>"&vbCrLf&_
"</HTML>"

'使用该模板添加一个新 ExportFormat 对象。
DataReport1.ExportFormats.Add_
Key:="DailyReport",_
FormatType:=rptFmtHTML,_
FileFormatString:="DailyReport(*.htm)",_
FileFilter:="*.HTM",_
Template:=strTemplate

'使用新的 ExportFormat 对象导出报表。
DataReport1.ExportReport_
FormatIndexOrKey:="DailyReport",_
```

```
    FileName:="C:\Temp\DailyRpt",_
    Overwrite:=True,_
    ShowDialog:=False,_
    Range:=rptRangeFromTo,_
    Pagefrom:=1,_
    Pageto:=10
EndSub
```

Add 方法(Folder)

添加一个新的 Folder 到 Folders 集合。

应用于

Folders 集合

语法

object.**AddFolders***folderName*

AddFolders 方法的语法有如下几部分：

部分	描述
object	必需的。一个 Folders 集合的名称
FolderName	必需的。新添加的 Folder 的名称

说明

如果 folderName 已经存在，则产生一个错误。

请参阅

Add 方法 (Dictionary)

Add 方法（Format 对象）

添加一个 StdDataFormat 对象到 StdDataFormats 集合。

应用于

StdDataFormats 集合

语法

object.**Add**(*dataformat*,[*index*])

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
dataformat	必需的。一个字符串表达式，指定将要添加到该集合中的对象的名字
index	可选的。一个整数，唯一标识该集合的一个成员

请参阅

StdDataFormat 对象

Add 方法（Licenses 集合）

添加一个许可证到 Licenses 集合并返回其许可证关键字。

语法

object.**Add**(*ProgID*,*LicenseKey*)

Add 方法语法有这些部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
progID	必需的。一个字符串，它指定要添加许可证关键字的控件
licenseKey	可选的。一个字符串，它指定许可证关键字

说明

任何时候当您想动态添加一个要求许可证关键字的控件时，请使用 Add 方法。关于要求许可证关键字的控件的详细信息，请参阅在“请参阅”列表中的“控件许可证问题”。

当您编译一个需要许可证关键字的用户控件，而且您想动态地添加该控件到一个现存的应用程序，您就必须按两种不同的途径对 Licenses 集合使用 Add 方法。

首先，使用该方法返回这个许可证关键字，它是被硬编码到一个用户控件上的。第二步，在添加用户控件到 Controls 集合之前，使用该方法把同一个许可证关键字添加到 Licenses 集合。

在大多数情况下，为了正确地部署一个已编译好的用户控件，您将必需按这两种途径使用该方法。其步骤概述如下。

在您编译完一个需要许可证关键字的用户控件后，使用 Add 方法返回其许可证关键字。把这个许可证关键字存储到部署应用程序能够检索到的地方。例如，下面的例子把关键字写到一个文件中。

您也可以把它存储到数据库或 Windows 注册表中。

```
PrivateSubGenerateLicenseKey()
   DimintFileAsInteger
   intFile=FreeFile
   '打开一个用来写入许可证关键字的文件。
   Open"c:\Temp\Ctl_Licenses.txt"ForOutputAs#intFile
   DimstrLicenseAsString
   strLicense=Licenses.Add("prjWeeks.WeeksCtl")
   '写许可证关键字到该文件。
   Write#intFile,strLicense
   Close#intFile
EndSub
```

当您部署您的控件时，在添加该控件到 Controls 集合之前，已部署的应用程序将把许可证关键字添加到 Licenses 集合。（当然，控件同时必须已经安装在机器上）下面的代码例子添加许可证关键字，然后添加控件：

```
OptionExplicit
DimWithEventsextObjAsVBControlExtender


PrivateSubLoadDynamicControl()
```

```
    DimintFileAsInteger
    intFile=FreeFile
    Open"c:\Download\Ctl_Licenses.txt"ForInputAs#intFile
    DimstrKeyAsString
    '在客户机上，从文件读许可证关键字。
    Input#intFile,strKey
    Licenses.Add"prjWeeks.WeeksCtl",strKey
    Close#intFile
    SetextObj=Controls.Add("prjWeeks.WeeksCtl","ctl1")
    WithControls("ctl1")
        .Visible=True
    EndWith
EndSub
```

什么时候添加许可证关键字

当您创建一个用户控件而且您想为动态控件添加而发布该控件时，您必须考虑下面的问题：用户控件仅包含固有控件吗？如果回答是“是”，则问“我想要求最终用户为了使用该控件而必需有许可证关键字吗？”如果两个问题的回答都是“是”，那么请一定要复选“工程属性”对话框中的“通用”选项卡上的“要求许可证关键字”选项。

注意即使您清除了“要求许可证关键字”选项，一个包含第三方

控件的用户控件将仍然要求一个许可证关键字。

什么时候不需要许可证关键字

有两种情况不需要在添加一个控件到 Controls 集合时添加许可证关键字：

1．当控件是一个固有控件，而且您没有复选“要求许可证关键字”选项。

2．当您添加一个在工程中已经被引用的控件时。换句话说，如果该控件是显示在“工具框”中的。

注意当您有一个控件在“工具箱”中，而且您计划仅仅在运行时添加该控件，一定要清除“工程属性”对话框的“生成”选项卡上的“删除有关未使用的 ActiveX 控件”选项，不然的话，试图添加控件将会失败。

Add 方法（VBA 外接程序对象模型）

将一个对象添加到集合。

应用于

LinkedWindows 集合，VBComponents 集合，VBProjects 集合

语法

object.**Add**(*component*)

Add 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
component	必需的。对于 LinkedWindows 集合，为一个对象。对于 VBComponents 集合，则为表示类模块、窗体、标准模块的列举常数

可以使用以下的常数作为 component 参数：

常数	描述
vbext_ct_ClassModule	将一个类模块添加到集合
vbext_ct_MSForm	将窗体添加到集合
vbext_ct_StdModule	将标准模块添加到集合

说明

对于 LinkedWindows 集合，Add 方法可将一个窗口添加到当前链接窗口的集合。

注意可以添加一个链接窗口框架中的窗格到另一个链接窗口框架；实际上就是将一个窗口从一个窗格移动到另一个窗格。如果链接窗口框架的所有窗格均删除，那么将撤消此无窗格的链接窗口框架。

重点用于控制链接窗口、链接窗口框架以及连接窗口的对象、属性和方法，为与 Windows 中书写的代码兼容，都包括在了 Macintosh 中。然而，当在 Macintosh 上运行时，这些语言元素

将产生运行时错误。

对于 VBComponents 集合，Add 方法将创建一个新的标准部件，并且将其添加到工程。

对于 VBComponents 集合，Add 方法返回 VBComponent 对象。对于 LinkedWindows 集合，Add 方法返回 Nothing。

请参阅

AddFomFile 方法,AddFromGuid 方法,AddFromString 方法,Remove 方法（VBA 外接程序对象模型）

Add 方法（VisualBasicforApplication）

添加一个成员到 Collection 对象。

应用于

Collection 对象

语法

object.Add*item,key,before,after*

Add 方法的语法具有下列对象限定符和命名参数：

部分	描述
object	必需的。对象表达式，其值为“应用于”列表中的对象
item	必需的。任意类型的表达式，指定要添加到集合中的成员
key	可选的。唯一字符串表达式，指定可以使用的键字符串，代替位置索引来访问集合中的成员
before	可选的。表达式，指定集合中的相对位置。在集合中将添加的成员放置在 before 参数识别的成员之前。如果为一数值表达式，则 before 必须是介于 1 和集合 Count 属性值之间的值。如果为一字符串表达式，则当添加一个被引用的成员到集合时，before 必须对应于指定的 key 值。可以指定 before 位置或 after 位置，但不能同时指定这两个位置
after	可选。表达式，指定集合中的相对位置。在集合中将添加的成员放置在 After 参数识别的成员之后。如果为一数值表达式，则 after 必须是介于 1 和集合 Count 属性值之间的值；如果为一字符串表达式，则当添加一个被引用的成员到集合时，after 必须对应于指定的 key 值。可以指定 before 位置或 after 位置，但不能同时指定这两个位置

说明

before 或 *after* 参数是字符串表达式或数值表达式，均须引用集合中现有成员，否则将导致错误发生。

如果指定的 *key* 和集合中现有成员的 *key* 发生重复，则也会导致错误发生。

请参阅

AddFromFile 方法，AddFromGuid 方法，AddFromString 方法，Remove 方法（VBA 外接程序对象模型），Item 方法，Remove 方法

示例

本示例使用 Add 方法将 Inst 对象（类 Class1 之示例，其中包含 Public 变量 InstanceName）加到 MyClasses 集合对象中。若要观察程序的运作，插入一个类模块，并在 Class1 的模块级中声明公用变量 InstanceName（键入 PublicInstanceName）来保存每个示例的名称。类名就用缺省的 Class1。将下列代码复制与粘贴到某个窗体模块的 Form_Load 事件过程中。

```
DimMyClassesAsNewCollection    '建立集合对象。
DimNumAsInteger    '定义计数变量。
DimMsg
DimTheName '用来保存用户指定的名称的变量。
Do
   DimInstAsNewClass1  '建立 Class1 的新示例。
   Num=Num+1    '将计数变量加一，并让用户输入名称。
   Msg="Pleaseenteranameforthisobject."&Chr(13)_
```

```
    &"PressCanceltoseenamesincollection."
    TheName=InputBox(Msg,"NametheCollectionItems")
    Inst.InstanceName=TheName     '将输入的名称存到对象的示例中。
    '如果名称不是空字符串，则将该示例加到集合对象中。
    IfInst.InstanceName<>""Then
        '将该示例加到集合对象中。
        MyClasses.Additem:=Inst,key:=CStr(Num)
    EndIf
    '清除对当前示例的引用，以准备下一个。
    SetInst=Nothing
LoopUntilTheName=""
ForEachxInMyClasses
    MsgBoxx.instancename,,"InstanceName"
Next
```

AddCustom 方法

该方法返回 VBComponent **对象，或者创建一个新的自定义部件并将它添加到该工程。**

应用于

VBComponents 集合

语法

object.**AddCustom**(ByVal*progid*AsString)AsVBComponent

AddCustom 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
progid	必需的。要创建的自定义部件的 ProgID

AddFile 方法

该方法返回新添加的部件。

语法

object.**AddFile**(ByVal*pathname* As String,[*relateddocument* As Boolean])
As VBComponent

AddFile 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
pathname	必需的。字符串表达式，指定作为模板打开的文件的路径和文件名
relateddocument	可选的（仅限于文本文件）。缺省值＝False。布尔表达式，指定该文件是作为标准模块处理，还是作为一个文档处理。如果设置为 True，则添加的文件作为文档文件处理

说明

如果 relateddocument 参数设置为 True，则通常是 VisualBasic 工

程部件的那些文件（如窗体文件）会引起错误。只有把添加的文本文件作为标准模块或文档处理时，才需要 relateddocument 参数。

AddFromFile 方法

该方法添加或打开一个工程或者组工程。

应用于

CodeModule 对象，VBProjects 集合

语法

object.**AddFromFile**(ByVal*pathname*AsString,[*exclusive*AsBoolean])As VBNewProjects

AddFromFile 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
pathname	必需的。字符串表达式，指定用作模板文件的路径
exclusive	可选的。缺省值=False。布尔表达式。如果设置为 True，则关闭已存在的组工程，并创建新的工程作为唯一的打开工程

说明

如果文件是组工程文件并且 exclusive 设置为 False，则那个组工程中的全部工程均被添加到当前的组工程中。如果文件是组工程文件并且 exclusive 设置为 True，则当前的组工程被指定的工程

所替换。

AddFromFile 方法（VBA 外接程序对象模型）

对于 References 集合，此方法添加一个从文件到工程的引用。对于 CodeModule 对象，此方法添加文件内容到模块中。

应用于

CodeModule 对象，Eeferences 集合

语法

object.**AddFromFile**(*filename*)

AddFromFile 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
filename	必需的。一个字符串表达式，用来指定欲添加到工程或模块的文件名。如果找不到文件名或未指明路径名，WindowsOpenFile 函数可搜寻目录

说明

对于 CodeModule 对象，AddFromFile 方法可在代码模块中第一个过程之前的行开始插入文件的内容。如果模块没有包含过程，AddFromString 可将文件的内容放置在模块的最后。

请参阅

ItemAdded 事件 (VBA 外接程序对象模型) ,ItemRemoved 事件 (VBA 外接程序对象模型) ,Add 方法 (VBA 外接程序对象模型) , AddFromGuid 方法, AddFromString 方法, Remove 方法 (VBA 外接程序对象模型)

下例使用 AddFromFile 方法，将文件内容加到特定代码窗格中。

```
ApplicationVBE.CodePanes(3).CodeModule.AddFromFile"c:\CodeFiles\book2.frm"
```

AddFromGuid 方法

使用引用的全域唯一标识符(GUID)，将一个引用添加到 References 集合。

应用于

References 集合

语法

object.**AddFromGuid**(*guid*,*major*,*minor*)AsReference

AddFromGuid 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
guid	必需的。一个字符串表达式，用来指定引用的 GUID
major	必需的。一个 Long 型数，用来指定引用的主版本号
minor	必需的。一个 Long 型数，用来指定引用的次版本号

说明

AddFromGuid 方法可搜寻注册表来找寻要添加的引用。GUID 可以是类型库、控件、类标识符等。

请参阅

ItemAdded 事件（VBAAdd-InObjectModel），ItemRemoved 事件（VBAAdd-InObjectModel），Add 方法（VBAAdd-InObjectModel），AddFromString 方法，Remove 方法（VBAAdd-InObjectMode）

示例

下例使用 AddFromGuid 方法把引用加到当前工程中，使用全程性的唯一 ID 值标识引用。

```
Application.VBE.ActiveVBProject.References.AddFromGuid("{000204ef-0000-0000-c000-000000000046}",5,0)
```

AddFromString 方法

将文本添加到模块。

应用于

CodeMoule 对象

语法

object.**AddFromString**

object 为一个对象表达式，其值是“应用于”列表中的一个对象。

说明

AddFromString 方法可在模块中第一个过程之前插入文件的内容。

如果模块没有包含过程，AddFromString 可将文件的内容安置在模块最后。

请参阅

CreateEventProc 方法，DeleteLines 方法，InsertLines 方法，Lines 方法，ReplaceLine 方法，ProcBodyLine 属性，ProcCountLines 属性，ProcOfLine 属性，ProcStartLine 属性

示例

下例使用 AddFromString 方法在特定代码窗格中加入一行，“DimintJackAsInteger”.

```
Application.VBE.CodePanes(3).CodeModule.AddFromString“DimintJackAsInteger”
```

AddFromTemplate 方法

VBComponents 集合：返回新创建的部件，并按照模板创建一个新部件。

VBProjects 集合：返回作为调用这个方法的结果而添加的全部工程的集合，或用已存在的工程作为模板创建一个新的工程。

语法

object.AddFromTemplate(*filename* AsString) AsVBComponent

object.**AddFromTemplate**(ByVal*pathname* AsString,[*exclusive* AsBoolean]) AsVBNewProjects

AddFromTemplate 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
filename	必需的。字符串表达式，指定作为模板而打开的文件的路径和文件名的
exclusive	可选的。缺省值=False。布尔表达式。如果设置为 True，则关闭已存在的组工程并且创建新的工程作为唯一的打开的工程
pathname	必需的。字符串表达式，指定用作模板文件的路径

说明

如果引用的文件类型是组工程文件，并且 exclusive 设置为 False，则在该文件中的全部工程被作为模板而创建，并添加到当前打开的工程集。然而，如果 exclusive 设置为 True，则关闭当前组工程，并创建一个新的组工程，而且在组工程模板中的全部工程被创建为工程模板。被该方法返回的对象是 Nothing。
赋予新工程以常用的缺省名。

AddIn 对象

AddIn 对象为一个外接程序对其它外接程序提供信息。

语法

AddIn

说明

为 Vbaddin.ini 文件中的每个外接程序创建 AddIn 对象。

“Add-In”工具栏



其上放有“外接程序(Add-In)”和“向导”的工具栏，以便用户轻松迅速地进行访问。要启动外接程序或向导，只须单击工具栏上的相应图标。

“外接程序”和“向导”置于“外接程序”工具栏上，只有单击其相应的按钮时才激活它。有了“外接程序”工具栏，就没有必要通过“外接程序管理器”对话框激活外接程序了。

可通过“添加/删除工具栏项”(+/-)按钮将向导及外接程序添加到“外接程序”工具栏上。单击该按钮时就可以得到以下对话框：



为将外接程序或向导添加到可用的外接程序列表中，可以单击“浏

览”按钮。将指针指向对话框中的外接程序或者向导的.Exe 文件或.Dll 文件，然后单击“打开”。这样，它就应该出现在“可用外接程序”列表中，但又不会出现在“外接程序”工具栏上，除非在“可用外接程序”列表中选中了它的复选框。

可用“确定”按钮将“添加/删除工具栏项”对话框关闭，所选中的各项将更新“外接程序”工具栏。

可用“取消”按钮将“添加/删除工具栏项”对话框关闭，并忽略在打开该对话框时所作的任何更改。

当单击“删除”按钮时，在“可用外接程序”列表中删除当前选定的外接程序或向导。注意，这样做并不会将系统中的外接程序或向导删除掉，也不会将“外接程序管理器”对话框中对它的引用删除掉。“删除”按钮只是把“外接程序”工具栏上的“可用外接程序”列表中的项删除掉。

请参阅

AddToAddInToolbar 方法，RemoveAddInFromToolbar 方法

AddIns 集合

返回 Vbaddin.ini 文件中列举的外接程序集合。

语法

AddIns

说明

AddIns 集合是通过 VBE 对象来访问的。VisualBasic 的加载管理

器中列举的每个外接程序都在 AddIns 集合中有一个对象。该集合取代了 VisualBasic4.0 版中的 ExternalObjects 集合。

属性

VBE 属性，Count 属性（VB 集合）Parent 属性

方法

Item 方法，Update 方法

请参阅

AddIn 对象

AddIns 属性

返回一个集合，给外接程序用来向扩展的对象模型注册它们的自动单元。

语法

object.**AddIns**

object 所在处是一个对象表达式，其值是“应用于”列表中的一个对象。

AddItem 方法

用于将项目添加到 ListBox 或 ComboBox 控件，或者将行添加到 MSFlexGrid 控件。不支持命名参数。

应用于

ComboBox 控件，ListBox 控件

语法

object.**AddItem***item*[,*index*]

AddItem 方法的语法包含下列部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
item	必需的。字符串表达式，它用来指定添加到该对象的项目。这仅仅对于 MSFlexGrid 控件，才能使用 tab 字符（字符代码 09）分隔将要插入到新添加行中每列的多重字符串
index	可选的。是整数，它用来指定新项目或行在该对象中的位置。对于 ListBox 或 ComboBox 控件的首项，或者对于 MSFlexGrid 控件的首行，index 为 0

说明

如果所给出的 index 值有效，则 item 将放置在 object 中相应的位置。如果省略 index，当 Sorted 属性设置为 True 时，item 将添加到恰当的排序位置，当 Sorted 属性设置为 False 时，item 将添加到列表的结尾。

绑定到 Data 控件的 ListBox 或 ComboBox 控件不支持 AddItem 方法。

请参阅

Clear 方法(剪贴板,ComboBox,ListBox),RemoveItem 方法,Index 属性（ActiveX 控件），Key 属性（ActiveX 控件）

示例

本示例使用 Addltem 方法增加 100 项给一个列表框。要检验此示例，可将本例代码粘贴到一个带有被命名为 List1 的 ListBox 控件的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimEntry,I,Msg   '声明变量。
   Msg="ChooseOKtoadd100itemstoyourlistbox."
   MsgBoxMsg    '显示信息。
   ForI=1To100  '计数值从 1 到 100。
      Entry="Entry"&I  '创建输入项。
      List1.AddItemEntry   '添加该输入项。
   NextI
   Msg="ChooseOKtoremoveeveryotherentry."
   MsgBoxMsg    '显示信息。
   ForI=1To50   '确定如何
      List1.RemoveItemI    '每隔一项
   NextI    '删除。
   Msg="ChooseOKtoremoveallitemsfromthelistbox."
   MsgBoxMsg    '显示信息。
   List1.Clear '清除列表框。
EndSub
```

AddressOf 运算符

一个一元运算符，它将其后面的过程的地址传递给一个 API 过程，该 API 过程在参数表对应位置中需要一个函数指针。

语法

AddressOf*procedurename*

必需的 *procedurename* 指定要传递的地址是哪一个过程的地址。这个过程必须是发出调用命令的工程中的一个标准模块模块里的一个过程。

说明

当一个过程的名称出现在一个参数列表中时，通常该过程已经被计算，并且该过程返回值的地址也会被传递。AddressOf 允许该过程的地址被传递给动态链接库(DLL)中的一个 WindowsAPI 函数，而不是传递该过程的返回值。API 函数然后就可以使用这个地址来调用相应的 Basic 过程，这个过程就是我们所知道的回调。AddressOf 运算符只出现在对 API 过程的调用中。

尽管可以使用 AddressOf 运算符在 Basic 过程之间传递过程指针，却不能通过这样的一个指针从 Basic 内部调用一个函数。这就是说，例如，一个使用 Basic 编写的类不能使用这样的一个指针来回调自己的控制器。当使用 AddressOf 运算符在 Basic 内的过程之间传递一个过程的指针时，被调用过程的参数类型必须定义为 AsLong。

警告如果不能完全理解函数回调的概念，则使用 AddressOf 运算符可能会导致意想不到的结果。必须理解回调的 Basic 部份是如何工作的，以及接受所传递的函数地址的 DLL 的代码是如何工作的。调试这样的交互操作是非常困难的，因为该程序和开发环境运行在相同的进程中。在某些情况下，系统的调试也许是不可能的。

注意可以使用 MicrosoftVisualC++（或者类似的工具）编译的 DLLs 来创建自己的回调函数原型。要使用 AddressOf 运算符来工作，您的原型必需使用__stdcall 调用约定。缺省的调用约定(__cdecl)不能与 AddressOf 运算符一起工作。

因为一个回调的调用程序不在用户自己的程序中，所以很重要的一点是要保证回调过程中的错误不被回传到调用者。可以通过把 OnErrorResumeNext 语句放置于回调过程的起始处来达到这个要求。

请参阅

Declare 语句，Function 语句，PropertyGet 语句，PropertyLet 语句，ProperySet 语句，Sub 语句

示例

下面的示例创建一个带有一个列表框的窗体，该列表框包含您的系统中的字体的按字母顺序排序的列表。

要运行该示例，请创建一个带有一个列表框的窗体。窗体部分的代码如下：

```
OptionExplicit
```

```
PrivateSubForm_Load()
Module1.FillListWithFontsList1
EndSub
```

把下面的代码放置到一个模块中。EnumFontFamilies 函数定义中的第三个参数是一个长整形，它代表一个过程。该参数必须包含这个过程的地址，而不是这个过程的返回值。在对 EnumFontFamilies 的调用中，第三个参数需要 AddressOf 运算符来返回 EnumFontFamProc 过程的地址，该过程是当调用 WindowsAPI 函数，即 EnumFontFamilies 时提供的回调过程的名称。当把 AddressOfEnumFontFamProc 传递给 EnumFontFamilies 时，Windows 就会为系统中的每一个字体库调用一次 EnumFontFamProc。传递给 EnumFontFamilies 的最后一个参数指定用来显示信息的列表框。

```
'字体枚举类型
PublicConstLF_FACESIZE=32
PublicConstLF_FULLFACESIZE=64

TypeLOGFONT
lfHeightAsLong
lfWidthAsLong
```

```
lfEscapementAsLong
lfOrientationAsLong
lfWeightAsLong
lfItalicAsByte
lfUnderlineAsByte
lfStrikeOutAsByte
lfCharSetAsByte
lfOutPrecisionAsByte
lfClipPrecisionAsByte
lfQualityAsByte
lfPitchAndFamilyAsByte
lfFaceName(LF_FACESIZE)AsByte
EndType

TypeNEWTEXTMETRIC
tmHeightAsLong
tmAscentAsLong
tmDescentAsLong
tmInternalLeadingAsLong
tmExternalLeadingAsLong
tmAveCharWidthAsLong
tmMaxCharWidthAsLong
tmWeightAsLong
```

```
tmOverhangAsLong
tmDigitizedAspectXAsLong
tmDigitizedAspectYAsLong
tmFirstCharAsByte
tmLastCharAsByte
tmDefaultCharAsByte
tmBreakCharAsByte
tmItalicAsByte
tmUnderlinedAsByte
tmStruckOutAsByte
tmPitchAndFamilyAsByte
tmCharSetAsByte
ntmFlagsAsLong
ntmSizeEMAsLong
ntmCellHeightAsLong
ntmAveWidthAsLong
EndType

'ntmFlags 段标志
PublicConstNTM_REGULAR=&H40&
PublicConstNTM_BOLD=&H20&
PublicConstNTM_ITALIC=&H1&
```

```
'tmPitchAndFamily 标志
PublicConstTMPF_FIXED_PITCH=&H1
PublicConstTMPF_VECTOR=&H2
PublicConstTMPF_DEVICE=&H8
PublicConstTMPF_TRUETYPE=&H4

PublicConstELF_VERSION=0
PublicConstELF_CULTURE_LATIN=0

'EnumFonts 掩码
PublicConstRASTER_FONTTYPE=&H1
PublicConstDEVICE_FONTTYPE=&H2
PublicConstTRUETYPE_FONTTYPE=&H4

DeclareFunctionEnumFontFamiliesLib"gdi32"Alias_
   "EnumFontFamiliesA"_
   (ByValhDCAsLong,ByVallpszFamilyAsString,_
   ByVallpEnumFontFamProcAsLong,LParamAsAny)AsLong
DeclareFunctionGetDCLib"user32"(ByValhWndAsLong)AsLong
DeclareFunctionReleaseDCLib"user32"(ByValhWndAsLong,_
   ByValhDCAsLong)AsLong
```

```
FunctionEnumFontFamProc(lpNLFAsLOGFONT,lpNTMAsNEWTEXTMETRIC,_
   ByValFontTypeAsLong,LParamAsListBox)AsLong
DimFaceNameAsString
DimFullNameAsString
FaceName=StrConv(lpNLF.lfFaceName,vbUnicode)
LParam.AddItemLeft$(FaceName,InStr(FaceName,vbNullChar)-1)
EnumFontFamProc=1
EndFunction

SubFillListWithFonts(LBAsListBox)
DimhDCAsLong
LB.Clear
hDC=GetDC(LB.hWnd)
EnumFontFamilieshDC,vbNullString,AddressOfEnumFontFamProc,LB
ReleaseDCLB.hWnd,hDC
EndSub
```

AddToAddInToolbar 方法

在引用某个外接程序或向导的“外接程序”工具栏上插入按钮。

应用于

Add-In 工具栏

语法

object.**AddToAddInToolbar**(*sfilename* AsString,sprogidAsString,*showontoolbar* AsBoolean,*forceaddintoolbar* AsBoolean)

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
sfilename	必需的。字符串表达式，它指定了外接程序或者向导的路径及其.Exe 或者.Dll 文件名
sprogid	必需的。字符串表达式，它指定了外接程序或者向导的程序 ID(ProgID)
saddinname	必需的。字符串表达式，它指定了外接程序或者向导的标题
showontoolbar	必需的。布尔表达式，它指定了所引用的外接程序或者向导是否将出现在“外接程序”工具栏上。True=yes，False=no
Forceaddintoolbar	必需的。布尔表达式，它指定了下一次启动 VisualBasic 是否将自动显示“外接程序”工具栏。True=yes，False=no

请参阅

Add-In 工具栏，RemoveAddInFromToolbar 方法

示例

在该示例中，对一个名为 MyAdd.Dll 的外接程序使用 AddToAddInToolbar 方法在“外接程序”工具栏中添加一个按钮。将 ForceAddInToolbar 设置为 True，这样就可保证下一次启动 VisualBasic 时加载“外接程序”工具栏。

可在一个比较小的 VisualBasic 应用程序中对下列代码进行修改，从而充当自己的外接程序的安装。

```
SubMain()
dimxasObject
Setx=CreateObject("AddInToolbar.Manager")
x.AddToAddInToolbarsFileName:="C:\VB5\MyAdd.DLL",_
sProgID:="MyAddIn.Connect",_
sAddInName:="MyAddInTitle"_
ShowOnToolBar:=True,_
ForceAddInToolbar:=True
EndSub
```

AddToolboxProgID 方法

该方法把控件或嵌入的部件放入工具箱并将控件引用添加到该工程中。

应用于

VBProject 对象

语法

object.**AddToolboxProgID**(ByVal*progid*AsString,[*filename*AsString])

AddToolboxProgID 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
progid	必需的。字符串表达式，指定复合文档对象的程序标识符(ProgID)，以添加到 VisualBasic 工具箱。与版本有关或无关的 ProgID 均可使用。如果指定了与版本无关的 progid，则大部分新的版本均可使用。如果复合文档对象有关联的类型库，则该类型库也被引
filename	可选的。字符串表达式，指定要添加到 VisualBasic 中的类型库的文件名。可以使用完整的路径名，但如果文件没有找到，即使指定了完整的路径名，也要搜索被 Windows 的 OpenFile 函数搜索过的目录

AfterAddFile 事件

该事件发生在用“工程”菜单中的“添加文件”命令，将控件添加到当前 VisualBasic 工程之后。

应用于

FileControlEvents 对象

语法

Sub*object*_**AfterAddFile**(*vbproject* AsVBProject, *filetype* Asvbext_FileType, *filename* AsString)

AfterAddFile 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
vbproject	VBProject 对象，指出其中已被添加文件的工程名
filetype	数字值(vbext_FileType)，指出被添加的文件的类型，如“设置值”中列表所示
filename	字符串表达式，指出被添加的文件名

设置值

vbext_FileType 的数字值是：

常数	值	描述
vbext_ft_Form	0	文件类型是窗体。
vbext_ft_Module	1	文件类型是基本模块。
vbext_ft_Class	2	文件类型是类模块。
vbext_ft_Project	3	文件类型是工程。
vbext_ft_Exe	4	文件类型是可执行文件。
vbext_ft_Res	6	文件类型是资源文件。
vbext_ft_UserControl	7	文件类型是用户控件。
vbext_ft_PropertyPage	8	文件类型是属性页。
vbext_ft_DocObject	9	文件类型是用户文档。
vbext_ft_Binary	10	文件类型是二进制文件。
vbext_ft_GroupProject	11	文件类型是工程组。
vbext_ft_Designer	12	文件类型是设计器对象。

说明

VisualBasic 仅对能从“工程”菜单添加的文件触发该事件。(即窗体、类、用户控件、属性页和模块)。如果从“工程”菜单中选择“添加”object，则不发生 AfterAddFile 事件。当.Frx 文件第一次创建时也不发生，并且在窗体第二次被添加时也不再发生。该事件发生在所有连接到 FileControl 对象的外接程序中。外接程序不能阻止文件写盘，因为写盘操作已经完成。然而，可以用该事件执行其它任务，如：

?记录关于该事件的信息。

?更新有关该文件的信息。

?备份该文件。

AfterChangeFileName 事件

该事件发生在当前工程中的文件第一次被保存，或用新名字保存之后，也发生在工程首次被编译为.Exe 文件或编译为新的.Exe 名字时。

应用于

FileControlsEvents 对象

语法

Sub*object*_**AfterChangeFileName**(*vbproject*AsVBProject,*filetype*Asvbext_FileType,*newname*AsString,*oldname*AsString)

AfterChangeFileName 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbproject	VBProject 对象，指出其中文件被更改的工程名
filetype	数字值(vbext_FileType)，指出被更改的文件类型，如“设置值”中列表所示
newname	字符串表达式，指出文件的新名字
oldname	字符串表达式，指出文件的旧名字

设置值

vbext_FileType 的数字值是：

常数	值	描述
vbext_ft_Form	0	文件类型是窗体。
vbext_ft_Module	1	文件类型是基本模块。
vbext_ft_Class	2	文件类型是类模块。
vbext_ft_Project	3	文件类型是工程。
vbext_ft_Exe	4	文件类型是可执行文件。
vbext_ft_Res	6	文件类型是资源文件。
vbext_ft_UserControl	7	文件类型是用户控件。
vbext_ft_PropertyPage	8	文件类型是属性页。
vbext_ft_DocObject	9	文件类型是用户文档。
vbext_ft_Binary	10	文件类型是二进制文件。
vbext_ft_GroupProject	11	文件类型是工程组。
vbext_ft_Designer	12	文件类型是设计器对象。

AfterCloseFile 事件

该事件发生在工程被关闭之后，不论是由用户直接关闭，还是当退出程序时由 VisualBasic 关闭。

应用于

FileControlEvnts 对象

语法

Sub*object*_**AfterCloseFile**(*vbproject*AsVBProject,*filetype*Asvbext_FileType,*filename*AsString,*wasdirty*AsBoolean)

AfterCloseFile 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbproject	VBProject 对象，指出其中文件被关闭的工程名
filetype	数字值(vbext_FileType)，指出被关闭的文件类型，如“设置值”列表所示
filename	字符串表达式，指出被关闭的文件名
wasdirty	布尔表达式，指出在文件关闭前是否保存对文件的更改，如“设置值”中列表所示

设置值

vbext_FileType 的数字值是：

常数	值	描述
vbext_ft_Form	0	文件类型是窗体。
vbext_ft_Module	1	文件类型是基本模块。
vbext_ft_Class	2	文件类型是类模块。
vbext_ft_Project	3	文件类型是工程。
vbext_ft_Exe	4	文件类型是可执行文件。
vbext_ft_Res	6	文件类型是资源文件。
vbext_ft_UserControl	7	文件类型是用户控件。
vbext_ft_PropertyPage	8	文件类型是属性页。
vbext_ft_DocObject	9	文件类型是用户文档。
vbext_ft_Binary	10	文件类型是二进制文件。
vbext_ft_GroupProject	11	文件类型是工程组。
vbext_ft_Designer	12	文件类型是设计器对象。

wasdirty 设置值是：

设置值	描述
True	当文件被关闭时是不干净的。（即在关闭它之前选择了不保存对文件所做的更改。）
False	当文件被关闭时是干净的。（即关闭它之前选择了保存对文件所做的更改。）

说明

该事件对每个工程中每个连接到 FileControl 对象的外接程序都发

生一次；对每个窗体、模块、类、和控制文件都发生一次，对工程文件也发生一次。

如果窗体不干净，且在“保存下列文件的更改吗?”对话框上选择了“否”，则 AfterCloseFile 事件不发生。同样当工程关闭时，对.Frx 文件该事件也不发生。而当.Frm 文件被储存时该事件发生。

该事件发生在所有连接到 FileControl 对象的外接程序中。外接程序不能阻止文件写盘，因为写盘操作已经完成。然而，可以用该事件执行其它任务，如：

?记录关于该事件的信息。

?更新有关该文件的信息。

?备份该文件。

?比较可执行文件(.EXE)的版本。

AfterRemoveFile 事件

该事件发生在从当前 VisualBasic 工程中删文件之后。

应用于

FileControlEvents 对象

语法

Sub*object*_**AfterRemoveFile**(*vbproject*AsVBProject,*filetype*Asvbext_FileType,filenameAsString)

AfterRemoveFile 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbproject	VBProject 对象，指出其中文件被删除的工程名
filetype	数字值(vbext_FileType)，指出被删除文件的类型，如“设置值”中列表所示
filename	字符串表达式，指出被删除的文件名

说明

对于在保存之前已被删除的部件，AfterRemoveFile 事件不发生。

该事件发生在所有连接到 FileControl 对象的外接程序中。外接程序不能阻止文件写盘，因为写盘操作已经完成。然而，可以用该事件执行其它任务，如：

?记录关于该事件的信息。

?更新有关该文件的信息。

?备份该文件。

请参阅

AfterCloseFile 事件

AfterWriteFile 事件

该事件发生在文件被写盘之后。

应用于

FileControlsEvents 对象

语法

Sub*object*_**AfterWriteFile**(*vbproject*AsVBProject,*filetype*Asvbext_FileTyp e,*filename*AsString,*result*AsInteger)

AfterWriteFile 事件语法包含下面部分：

部分	描述
object	**对象表达式，其值是“应用于”列表中的对象**
vbproject	VBProject **对象，指出其中已被写入文件的工程名**
filetype	**数字值**(vbext_FileType)**，指出写盘文件的类型，如“设置值”中列表所示**
filename	**字符串表达式，指出写盘文件名**
result	**数值表达式，指出写操作的结果，如“设置值”列表所示**

设置值

vbext_FileType **的数字值是：**

常数	值	描述
vbext_ft_Form	0	文件类型是窗体。
vbext_ft_Module	1	文件类型是基本模块。
vbext_ft_Class	2	文件类型是类模块。
vbext_ft_Project	3	文件类型是工程。
vbext_ft_Exe	4	文件类型是可执行文件。
vbext_ft_Res	6	文件类型是资源文件。
vbext_ft_UserControl	7	文件类型是用户控件。
vbext_ft_PropertyPage	8	文件类型是属性页。
vbext_ft_DocObject	9	文件类型是用户文档。
vbext_ft_Binary	10	文件类型是二进制文件。
vbext_ft_GroupProject	11	文件类型是工程组。
vbext_ft_Designer	12	文件类型是设计器对象。

result 的设置值是：

值	描述
0	写盘成功。
1	写盘取消。
2	写盘失败。

说明

当与部件（如.Frx 文件）相关联的二进制数据文件第一次被保存时，AfterWriteFile 事件发生，并且发生在所有连接到

FileControl 对象的外接程序中。外接程序不能阻止文件写盘，因为写盘操作已经完成。然而，可以用该事件执行其它任务，如：

?记录关于该事件的信息。

?更改有关该文件的信息。

?备份该文件。

?比较可执行文件(.EXE)的版本。

Align 属性

返回或设置一个值，确定对象是否可在窗体上以任意大小、在任意位置上显示，或是显示在窗体的顶端、底端、左边或右边，而且自动改变大小以适合窗体的宽度。

应用于

ADOData 控件，ProgressBar 控件，StatusBar 控件，Toolbar 控件，CoolBar 控件，RemoteData 控件，Extender 对象，PictureBox 控件，Data 控件

语法

object.**Align**[=*number*]

Align 属性语法包含下面部分:

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	整数值，用来确定如何显示对象，“设置值”中有详细描述

设置值

number 的设置值如下：

常数	设置值	描述
vbAlignNone	0	（非 MDI 窗体的缺省值）无－可以在设计时或在程序中确定大小和位置。如果对象在 MDI 窗体上，则忽略该设置值。
vbAlignTop	1	（MDI 窗体的缺省值）顶部－对象显示在窗体的顶部，其宽度等于窗体的 ScaleWidth 属性设置值。
vbAlignBottom	2	底部－对象显示在窗体的底部，其宽度等于窗体的 ScaleWidth 属性设置值。
vbAlignLeft	3	左边－对象在窗体的左面，其宽度等于窗体的 ScaleWidth 属性设置值。
vbAlignRight	4	右边－对象在窗体的右面，其宽度为窗体的 ScaleWidth 属性设置值。

说明

用 Align 属性可以很快地在窗体的顶部或底部创建工具栏或状态栏。当用户改变窗体的大小时，Align 值设置为 1 或 2 的对象，会自动地改变大小以适合窗体的宽度。

PictureBox 和 Data 控件是唯一能放在 MDI 窗体上的标准控件。MDI 窗体的内部区域定义为没有被控件覆盖的空间。当 MDI 子窗体在 MDI 父窗体中最大化时，它不会覆盖任何控件。

number 设置值 3 和 4 可以用来将工具栏对齐于窗体或 MDI 窗体的左边和右边。如果在 MDI 窗体的一个角上有两个工具栏，上或下对齐会优先占满整个角，而不用左右对齐。就象上下对齐的对象一样，左右对齐的对象占据 MDI 窗体的内部区域。

请参阅

Alignment 属性，ScaleHeight,ScaleWidth 属性

示例

这个例子是在 MDIForm 对象中把 PictureBox 控件作为工具栏，利用 CommandButton 控件把 PictureBox 从窗体的顶部移动到底部。要试用此例，先创建一个新的 MDIForm，并设置 Form1 的 MDIChild 属性为 True。在 MDIForm 中画一个 PictureBox，并在 PictureBox 中放一个 CommandButton。把代码粘贴到 MDIForm 的声明部分，然后按下 F5 键。单击 CommandButton 移动 PictureBox。

```
PrivateSubCommand1_Click()
   IfPicture1.Align=vbAlignTopThen
       Picture1.Align=vbAlignBottom
'Aligntobottomofform.
   Else
       Picture1.Align=vbAlignTop
'Aligntotopofform.
   EndIf
EndSub
```

Alignable 属性

返回或设置一个数值，此数值决定控件是否能对齐以及是否能使用扩展 Align 属性。在控件创建时，Alignable 属性可读可写，但在运行时，此属性不可用。

应用于

UserControl 对象

设置值

Alignable 的设置值为：

设置值	描述
True	控件能够对齐；容器将 Align 属性添加到扩展对象中。
False	（缺省）控件不能对齐。

说明

控件本身的对齐由容器处理；在对齐操作时，可使用 Align 扩展属性决定如何重新绘制控件以及如何调整子控件的位置。

注意并非所有的容器都支持可对齐的控件。使用 Align 扩展属性判断控件如何对齐时，应使用错误捕获。

请参阅

Align 属性

Alignment 属性

设置或返回一个值，决定 CheckBox 或 OptionButton 控件、控件中的文本、或 DataGrid 控件列中的值的对齐方式。对 CheckBox、OptionButton 和 TextBox 控件在运行时为只读。

应用于

Column 对象，Function 控件（数据报表设计器），Label 控件（数据报表设计器），TextBox 控件（数据报表设计器），CheckBox 控件，Label 控件，OptionButton 控件，TextBox 控件

语法

object.**Alignment**[=*number*]

Alignment 属性语法具有下列组成部分:

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	整型值，指定对齐类型，“设置值”中有详细描述

设置值

对 CheckBox 和 OptionButton 控件，number 的设置值如下：

常数	设置值	描述
vbLeftJustify	0	（缺省值）文本是左对齐的，控件是右对齐的
vbRightJustify	1	文本右对齐，控件左对齐。

对 Label 和 TextBox 控件，number 设置值为：

常数	设置值	描述
vbLeftJustify	0	（缺省值）文本左对齐。
vbRightJustify	1	文本右对齐
vbCenter	2	文本居中

对 DataGrid 列,number 设置值为:

常数	设置值	描述
dbgLeft	0	文本左对齐。
dbgRight	1	文本右对齐。
dbgCenter	2	文本居中。
dbgGeneral	3	（缺省）通用形式－文本左对齐；数值右对齐。

说明

可以在 OptionButton 和 CheckBox 控件的右边或左边显示文本。缺省值情况下，文本是左对齐的。

为保证 Alignment 属性能够准确工作，Textbox 控件中的 MultiLine 属性必须设置为 True。如果 Textbox 控件中的 MultiLine 设置为 False，则忽略 Alignment 属性。

请参阅

ColumnHeader 对象，ColumnHeaders 集合，Add 方法（ColUmn Headers 集合），Align 属性

Ambient 属性

返回包含容器环境信息的 AmbientProperties 对象。控件创建时，Ambient 属性不可用，而控件运行时，该属性是只读的。

应用于

UserControl 对象

语法

object.**Ambient**

Ambient 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

请参阅

AmbientProperties 对象

AmbientChanged 事件

当环境属性的数值变更时，发生该事件。

应用于

UserControl 对象

语法

Sub*object*_**AmbientChanged**(*Property*NameAsString)

AmbientChanged 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
PropertyName	字符串，它标识环境属性已更改

说明

使用 PropertyName，控件能够访问 Ambient 属性中的 Ambient Properties 对象，来检查已变更的环境属性的新数值。

如果在 VisualBasic 窗体上放置了该控件的实例，而且窗体的 FontTransparent 属性被变更，则并不产生 AmbientChanged 事件。

请参阅

AmbientProperties 对象，FontTransparent 属性

AmbientProperties 对象

AmbientProperties 对象保持来自容器的环境信息，这些信息决定了容器内控件的行为。

说明

容器提供的环境属性可影响控件的行为。例如，BackColor 是一个标准的环境属性；容器指示控件在正常情况下如何设置自己的背景颜色属性。

AmbientProperties 对象的属性是容器的环境属性。这些属性是只读的。

有些环境属性是标准的，有些则是某些容器特有的。控件可访问非标准的环境属性，但是这样使控件只适用于某些容器。当前容器没有提供某个环境属性时，控件也应该能够处理这种情况。

编译控件后，VisualBasic 无法获知控件运行时能提供哪种特定容器的环境属性；因此特定容器环境属性的引用总是后期绑定的。

初始化事件产生时，AmbientProperties 对象不可用；但是 InitProperties 事件和 ReadProperties 事件产生时，此对象可用。

AmbientProperties 对象具有若干标准属性：

?BackColor 属性，颜色值，它包含推荐使用的所含控件的内部颜色。如果容器不支持该属性，VisualBasic 提供的缺省值为 0x80000005：这是窗口背景的系统颜色。

?DisplayAsDefault 属性，布尔值，它指定控件是否为缺省控件。如果容器不支持该属性，VisualBasic 提供的缺省设置值为 False。

?DisplayName 属性，字符串，它包含控件自己显示的名称。如果容器不支持该属性，VisualBasic 提供的缺省值是空字符串：""。

?Font 属性，Font 对象，它包含推荐使用的控件所含的字体信息。如果容器不支持该属性，VisualBasic 提供的缺省值为 MSSansSerif8。

?ForeColor 属性，颜色值，它包含推荐使用的控件所含的前景色。如果容器不支持该属性，VisualBasic 提供的缺省值为 0x80000008：这是窗口文本的系统颜色。

?LocaleID 属性，长整型数值，它指定用户的语言和国家/地区。如果容器不支持该属性，VisualBasic 提供的缺省值是当前的系统区域 ID。

?MessageReflect 属性，布尔值，它指示控件是否支持消息返回。如果容器不支持该属性，VisualBasic 提供的缺省值为 False。

?Palette 属性，Picture 对象，其调色板指定推荐使用的控件所含调色板。

?RightToLeft 属性，布尔值，它指示双向系统上文本的显示方向及控件的外观。如果容器不支持该属性，VisualBasic 提供的缺省值为 False。

?ScaleUnits 属性，字符串，它包含容器使用的坐标单位的名称。如果容器不支持该属性，VisualBasic 提供的缺省值是空字符串："""。

?ShowGrabHandles 属性，布尔值，它指定控件是否处理抓取处理的显示。如果容器不支持该属性，VisualBasic 提供的缺省值为 True。

?ShowHatchings 属性，布尔值，它指定容器是否处理阴影的显示。如果容器不支持该属性，VisualBasic 提供的缺省值为 True。

?SupportsMnemonics 属性，布尔值，它指示容器是否处理控件的访问键。如果容器不支持该属性，VisualBasic 提供的缺省值为 False。

?TextAlign 属性，枚举，它指定文本对齐方式。如果容器不支持

该属性，VisualBasic 提供的缺省值为 0-GeneralAlign。

?UserMode 属性，布尔值，它指定环境是处于设计模式还是处于最终用户模式。如果容器不支持该属性，VisualBasic 提供的缺省值为 True。

?UIDead 属性，布尔值，它指定用户界面是否为不响应的。如果容器不支持该属性，VisualBasic 提供的缺省值为 False。

注意当访问 AmbientProperties 对象的 Font 属性时，没有得到对容器字体的引用，而是得到一个对字体的复制（clone）。

这样做的原因是每当容器中增加一个控件实例时，通常用 AmbientProperties.Font 来初始化控件的字体。如果提供对字体自身的一个引用，那么控件字体的改变同样也使容器的字体改变。

如果由于某种原因控件需要对容器字体的引用时，可以通过 UserControl 对象的 Parent 属性访问容器得到它。

属性

DisplayAsDefault 属性，DisplayName 属性，LocaleID 属性，MessageReflect 属性，RightToLeft 属性，ScaleUnits 属性，ShowGrabHandles 属性，ShowHatching 属性，SupportsMnemonics 属性，TextAlign 属性，UIDead 属性，UserMode 属性，Palette 属性，BackColor,ForeColor 属性，Font 属性

App 对象

App 对象是通过关键字 App 访问的全局对象。它指定如下信息：应用程序的标题、版本信息、可执行文件和帮助文件的路径及名称以及是否运行前一个应用程序的示例。

语法

App

属性

OLERequestPendingMsgText 属性，OLERequestPendingMsgTitle 属性，OLERequestPendingTimeout 属性，OLEServerBusyMsgText 属性，OLEServerBusyMsgTitle 属性，OLEServerBusyRaiseError 属性 OLEServerBusyTimeout 属性，EXEName 属性，TaskVisible 属性，hInstance 属性，Comments 属性，CompanyName 属性，FileDescription 属性，LegalCopyright 属性，LegalTrademarks 属性，Major 属性，Minor 属性，ProductName 属性，Revision 属性，LogMode 属性，LogPath 属性，NonModalAllowed 属性，ThreadID 属性，UnattendedApp 属性，HelpFile 属性（App,CommonDialog,MenuLine），PrevInstance 属性，Title 属性，StartMode 属性，Path 属性

方法

LogEvent 方法，StartLogging 方法

请参阅

HelpContextID 属性

App 属性

返回 App 对象，一个是用 App 关键字来访问的全局对象。它决定或者指定了有关应用程序标题、版本信息、其可执行文件以及帮助文件的路径和名称的信息，并判定应用程序的一个以前版本的示例是否正在运行。

应用于

Global 对象

语法

App

说明

App 对象没有任何事件或者方法。

请参阅

Global 对象，App 对象

AppActivate 语句

激活一应用程序窗口。

语法

AppActivate*title*[,*wait*]

AppActivate 语句的语法具有以下几个命名参数：

部分	描述
title	必需的。字符串表达式，所要激活的应用程序窗口的标题。可以使用 Shell 函数返回的任务识别码来替换 title，以激活应用程序
wait	可选的。Boolean 值，说明在激活另一个应用程序之前调用的应用程序是否有焦点。如果为 False（缺省），那么，即使调用的应用程序没有焦点，也直接激活指定的应用程序。如果为 True，则调用的应用程序会等到有焦点后，才激活指定的应用程序

说明

AppActivate 语句将焦点移动到命名的应用程序或窗口，但并不影响焦点是否最大化或最小化。当用户采取行动改变焦点或将窗口关闭时，就会将焦点从活动的应用程序窗口移动出去。可用 Shell 函数启动一个应用程序并设置窗口样式。

在决定激活哪个应用程序时，请将 *title* 与每一个运行中的应用程序的标题字符串进行比较。如果没有完全匹配，就激活任何这样的应用程序，其标题字符串以 *title* 开头。如果以 *title* 命名的应用程序有很多实例，则激活任何一个实例。

请参阅

Shell 函数，SendKeys 语句

示例

本示例说明使用 AppActivate 语句来激活应用程序的各种用法。示

例中用到的 Shell 语句均假定该应用程序已经存在指定的路径中。在 Macintosh 上，缺省驱动器名为"HD"，且路径名部分由冒号而不是反斜杠分隔。

```
DimMyAppID,ReturnValue
AppActivate"MicrosoftWord" '激活Microsoft
   'Word。

'AppActivate 也可利用 Shell 函数的返回值。
MyAppID=Shell("C:\WORD\WINWORD.EXE",1) '运行MicrosoftWord。
AppActivateMyAppID '激活 Microsoft
   'Word。

'您也可使用 Shell 函数的返回值。
ReturnValue=Shell("c:\EXCEL\EXCEL.EXE",1)  '运行 MicrosoftExcel。
AppActivateReturnValue '激活Microsoft
   'Excel。
```

Appearance 属性

返回或设置 MDIForm 或 Form 对象上的控件在设计时的绘图风格。在运行时是只读的。

应用于

ADO 数据控件，DBCombo 控件，DBList 控件，RemoteData 控件，

Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，FormsCollection，Frame 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件，OLEContainer 控件

语法

object.Appearance

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

设置值

Appearance 属性的设置值是：

设置值	描述
0	平面绘制控件和没有可视化效果的窗体。
1	（缺省值）3D。带有三维效果的绘制控件。

说明

如果在设计时将其设置为 1，那么 Appearance 属性在画出控件时带有三维效果。如果窗体的 BorderStyle 属性被设置为固定双边框（vbFixedDouble，或 3），窗体的标题和边框也是以有三维效果的方式绘画的。将 Appearance 属性设置为 1，也导致窗体及其控件的 BackColor 属性被设置为这样的颜色，该颜色是为操作系统的“显示属性”对话框的“外观”选项卡中的 3D 对象选定的。

将 MDIForm 对象的 Appearance 属性设置为 1，只对 MDI 父窗体产生影响。想要在 MDI 子窗体上具有三维效果，必须将每个子窗体的 Appearance 属性设置为 1。

请参阅

BackColor,ForeColor 属性，BorderStyle 属性，BackColor,ForeColor 属性 (ActiveX 控件)，BorderStyle 属性 (ActiveX 控件)

AppIsRunning 属性

返回或设置一个值，指示在 OLE 容器控件中创建对象的应用程序是否正在运行。在设计时不可用。

应用于

OLE 容器控件

语法

object.**AppIsRunning**[=*boolean*]

AppIsRunning 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指示在 OLE 容器控件中创建对象的应用程序是否正在运行，在“设置值”中有详细说明。

设置值

boolean 的设置值是：

设置值	描述
True	在 OLE 容器控件中创建对象的应用程序正在运行。
False	在 OLE 容器控件中创建对象的应用程序没有运行。

说明

可以设置 AppIsRunning 属性的值，来启动在 OLE 容器控件中创建对象的应用程序。这样作使对象激活得更快。当对象丢失焦点后，也可以将这个属性设置为 False 来关闭应用程序。

请参阅

OLEType 属性

Application 属性(WebClass)

返回活动服务器页面的 Application 对象。

应用于

WebClass 对象

语法

***object*.Application**

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

WebClass 使用 Application 对象管理通过多个 WebClass 对象实例共享的状态。

关于属性、方法和 Application 对象事件的详细内容，请参阅活动服务器页面文档。

请参阅

《MicrosoftVisualBasic6.0 参考库手册——MicrosoftVisualBasic6.0 组件工具指南》第 5 部分第 3 章“开发 IIS 应用程序”中的“IIS 应用程序对象模型”和“在对象中储存状态”。

ApplyChanges 事件

当按下属性页上的“确定”按钮或“应用”按钮时，或者由于选择选项卡切换了属性页时，发生该事件。

应用于

MSChart 对象，PropertyPage 对象

语法

Sub*object*_**ApplyChanges**()

ApplyChanges 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

ApplyChanges 事件发生时，属性页的创建者需要处理控件所有

新属性数值的设置；创建者最好能始终监视被更改的属性，否则需要设置所有的属性。若要知道哪个控件被更改了，可使用 SelectedControls 属性。

只有当 Changed 属性设置为 True 时，才能产生 ApplyChanges 事件。

请参阅

Changed 属性，SelecteControls 属性

Archive、Hidden、Normal 和 System 属性

设置或返回一值，决定 FileListBox 是否以档案、隐藏、普通或是系统属性来显示文件。

应用于

TreeView 控件，ListView 控件，FileListBox 控件

语法

object.**Archive**[=*boolean*]

object.**Hidden**[=*boolean*]

object.**Normal**[=*boolean*]

object.**System**[=*boolean*]

Archive，Hidden，Normal 和 System 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指定显示文件的类型，“设置值”中有详细描述

设置值

boolean 的设置值为:

设置值	描述
True	（档案和普通文件的缺省值）以 FileListBoxs 控件的属性特征显示文件。
False	（隐藏和系统文件的缺省值）不以 FileListBoxs 控件的属性特征显示文件。

说明

基于运行系统使用的标准文件特征，可以用这些属性来指定在 FileListBox 控件中所显示文件的类型。运行时在程序中设置这些属性中的任一个都会重设 FileListBox 控件使其只显示具有指定属性的文件。

例如，在查找和替换操作中，可以通过设置 System 属性为 True、而其它属性为 False，只显示系统文件；或者，作为文件备份过程的一部分，可以设置 Archive 属性为 True，以便只列出以前备份后修改过的文件。

请参阅

Pattern 属性，Path 属性

Arrange 方法

用以重排 MDIForm 对象中的窗口或图标。不支持命名参数。

应用于

MDIForm 对象

语法

object.**Arrange**arrangement

Arrange 方法的语法包含下列部分：

部分	描述
object	必需的。是一个对象表达式，其值为“应用于”列表中的一个对象
arrangement	必需的。一个数值或常数，如“设置值”中所描述的，它指定如何重排 MDIForm 中的窗口或图标

设置值

arrangement 的设置值有：

常数	值	描述
vbCascade	0	层叠所有非最小化 MDI 子窗体。
VbTileHorizontal	1	水平平铺所有非最小化 MDI 子窗体。
VbTileVertical	2	垂直平铺所有非最小化 MDI 子窗体。
VbArrangeIcons	3	重排最小化 MDI 子窗体的图标。

请参阅

Zorder 方法

说明

上述常数在 VisualBasic(VB)的对象浏览器中的对象库里列出。即使 MDIForm 对象被最小化，仍要重排窗口或图标。MDIForm 最大化时可以看到重排的结果。

示例

本示例使用 Arrange 方法重排一个 MDI 窗体中的窗口和图标。要检验此示例，可将本例代码粘贴到带有一个 MDI 子窗体（被命名为 Form1，其 MDIChild 属性设定为 True）和一个图片框（名为 Picture1）的被命名为 MDIForm1 的 MDI 窗体的声明部分，然后按 F5 并单击图片框中任何地方查看 Arrange 方法的效果。

```
ConstFORMCOUNT=5
DimF(1ToFORMCOUNT)AsNewForm1
PrivateSubMDIForm_Load()
   DimI'声明局部变量。
   LoadForm1    '加载原始 Form1 窗体。
   ForI=1ToFORMCOUNT
       F(I).Caption="Form"&I+1  '改变副本上的标题。
   NextI
EndSub

PrivateSubPicture1_Click()
```

```
    StaticClickCount'声明变量。
    DimI,PrevWidth,Start
    ClickCount=ClickCount+1 '增量单击计数器。
    SelectCaseClickCount
        Case1
            MDIForm1.Arrange1    '水平平铺。
        Case2
            MDIForm1.Arrange2    '垂直平铺。
        Case3    '最小化每个窗体。
            PrevWidth=MDIForm1.Width'取 MDI 的窗体宽度。
            MDIForm1.Width=PrevWidth/2'将它除以 2。
            Form1.WindowState=1 '使该原件最小化。
            ForI=1ToFORMCOUNT    '看 F 的每一实例。
                F(I).WindowState=1'最小化 F 的每个副本。
            NextI
            Start=Timer
            Do
            LoopUntilTimer=Start+5
            MDIForm1.Width=PrevWidth'回复到原始尺寸。
            MDIForm1.Arrange3    '重排图标。
    EndSelect
EndSub
```

Array 函数

返回一个包含数组的 Variant。

语法

Array(arglist)

所需的 arglist 参数是一个用逗号隔开的值表，这些值用于给 Variant 所包含的数组的各元素赋值。如果不提供参数，则创建一个长度为 0 的数组。

说明

用来表示数组元素的符号由变量名、圆括号以及括号中的所需元素的索引号组成。在下面的示例中，第一条语句创建一个 Variant 的变量 A。第二条语句将一个数组赋给变量 A。最后一条语句将该数组的第二个元素的值赋给另一个变量。

```
DimAAsVariant
A=Array(10,20,30)
B=A(2)
```

使用 Array 函数创建的数组的下界受 OptionBase 语句指定的下界的决定，除非 Array 是由类型库（例如 VBA.Array）名称限定。如果是由类型库名称限定，则 Array 不受 OptionBase 的影响。

注意没有作为数组声明的 Variant 也可以表示数组。除了长度固定的字符串以及用户定义类型之外，Variant 变量可以表示任何类型的数组。尽管一个包含数组的 Variant 和一个元素为 Variant 类型的

数组在概念上有所不同，但对数组元素的访问方式是相同的。

请参阅

Variant 数据类型,Deftype 语句,Dim 语句,Let 语句,OptionBase 语句

示例

该示例使用 Array 函数来返回一个包含 Variant 的数组。

```
DimMyWeek,MyDay
MyWeek=Array("Mon","Tue","Wed","Thu","Fri","Sat","Sun")
'返回值假设下界的设置为 1（使用 OptionBase
'语句)。
MyDay=MyWeek(2)      'MyDay 的值为"Tue"。
MyDay=MyWeek(4)      'MyDay 的值为"Thu"。
```

Asc 函数

返回一个 Integer，代表字符串中首字母的字符代码。

请参阅

Conversion 函数，Type 转换函数，Chr 函数

语法

Asc(*string*)

必要的 string 参数可以是任何有效的字符串表达式。如果 string 中没有包含任何字符，则会产生运行时错误。

说明

在非 DBCS 系统下，返回值范围为 0–255。在 DBCS 系统下，则为 -32768–32767。

注意 AscB 函数作用于包含在字符串中的字节数据，AscB 返回第一个字节的字符代码，而非字符的字符代码。AscW 函数返回 Unicode 字符代码，若平台不支持 Unicode，则与 Asc 函数功能相同。

示例

本示例使用 Asc 函数返回字符串首字母的字符值（ASCII 值）。

```
DimMyNumber
MyNumber=Asc("A")   '返回 65。
MyNumber=Asc("a")   '返回 97。
MyNumber=Asc("Apple")   '返回 65。
```

Assert 方法

有条件地在该方法出现的行上挂起执行。

应用于

Debug 对象

语法

object.**Assert***booleanexpression*

Assert 方法的语法有如下的对象限定符和参数：

部分	描述
object	必需的。总是 Debug 对象
booleanexpression	必需的。一个值为 True 或者 False 的表达式。

说明

Assert 调用只在开发环境中工作。当模块被编译成为一个可执行的文件时，调用 Debug 对象的方法就会被忽略。

全部 booleanexpression 常常被计算。例如，即使一个 And 表达式的第一部分被计算为 False，整个表达式还要被计算。

请参阅

Print 方法

示例

下面的示例说明如何使用 Assert 方法。示例需要一个带有两个按钮控件的窗体。缺省的按钮名称是 Command1 和 Command2。

当示例运行时，单击 Command1 按钮使得按钮上的文本在 0 和 1 之间进行切换。单击 Command2 按钮可能不做任何事，也可能引起一个确认，应该执行哪一个操作取决于 Command1 按钮上所显示的值。该确认将在最后一个语句执行之后使整个执行停止，并且 Debug.Assert 行被突出显式。

```
OptionExplicit

PrivatebInAssertAsBoolean

PrivateintNumberAsInteger
```

```
PrivateSubCommand1_Click()
blnAssert=NotblnAssert
intNumber=IIf(intNumber<>0,0,1)
Command1.Caption=intNumber
EndSub

PrivateSubCommand2_Click()
Debug.AssertblnAssert
EndSub

PrivateSubForm_Load()
Command1.Caption=intNumber
Command2.Caption="AssertTester"
EndSub
```

AsyncCount 属性

返回仍在执行的异步操作数量。

应用于

DataReport 对象

语法

object.**AsyncCount**

object 所在处是一个对象表达式，其值为“应用于”列表中的一个对象。

返回类型

Integer

说明

关闭数据报表设计器之前，查询 AsyncCount 属性以决定仍然执行的操作的数量。您也许要在所有操作都已经结束之前取消关闭设计器。

示例

本例在 While 循环中调用 ExportReport 方法并查询 AsyncCount 属性，来决定是否有异步的操作仍在执行。当所有的操作都结束，DataReport 对象被卸载。

```
DataReport1.ExportReportrptKeyText,"c:\MyDocuments\Report",True
WhileDataReport1.AsyncCount>0
   DoEvents
Wend
UnloadDataReport1
```

AsyncLoad 属性

设置一个值，决定代码的执行是否与页面上对象的加载异步。运行时不可用。

应用于

DHTMLPageDesigner 对象

说明

缺省情况下，AsyncLoad 设置为 False，意味着直到浏览器检索到页面上的所有元素以后代码才开始执行。如果设置为 True，代码一旦被浏览器下载了就马上执行，不管其它组件的下载状态。在更改该属性值之前，请确保您的代码已经作好准备以处理以下的情况，那就是代码开始执行时页面上的控件可能还未被加载。

请参阅

在《MicrosoftVisualBasic6.0 参考库手册》系列的《VisualBasic6.0部件工具指南》一书中，第5部分“构造Inernet应用程序”的第2章“DHTML 应用程序设计思想”。

AsyncProgress 事件

在一个异步的操作过程中发生，数据的每一页面都已被指定的操作（由 JobType 参数决定）处理过。

应用于

DataReport 对象

语法

PrivateSub*object*_**AsyncProgress**(*JobType*AsAsyncTypeConstants,*Cookie*AsLong,*PageCompleted*AsLong,*TotalPages*AsLong)

AsyncProgress 事件的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
jobType	一个整数，指定操作类型，如在设置值中所示
cookie	一个长整形数，标识特定的操作
pageComplet ed	一个变量，返回完成的页面数
totalPages	一个变量，返回将要处理的页面总数

设置值

对 JobType 的设置如下：

常数	值	描述
rptAsyncPreview	0	报表正在处理一个预览操作。
rptAsyncPrint	1	报表正在处理一个打印报表操作。
rptAsyncReport	2	报表正在处理一个导出报表操作。

请参阅

PocessingTimeout 事件

AsyncProperty 对象

AsyncProperty 对象被传递到 AsyncReadComplete 事件中，并包含了 AsyncRead 方法的结果。

属性

BytesMax 属性，BytesRead 属性，Status 属性（AsyncProperty 对象），StatusCode 属性，Target 属性，AsyncType 属性，PropettyName 属性，Value 属性

请参阅

AsyncRead 方法，AsyncReadComplete 事件

AsyncRead 方法

从文件或 URL 中，通过容器 ActiveX 部件，初始化异步数据读取。

应用于

UserControl 对象，UserDocument 对象

语法

object.**AsyncRead***Target*,*AsyncType*[,*PropertyName*],[*AsyncReadOptions*]

AsyncRead 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
target	字符串表达式，它指定数据的位置。可以是路径或 URL
asyncType	整型表达式，它指定显示数据的方法，如“设置值”中所描述的
propertyName	可选择的字符串表达式，它指定要加载的属性名。在区别同时下载时需要
asyncReadOptions	一个可选的字符串表达式，指出 AsyncRead 的附加选项，如“设置值”中所描述的

设置值

AsyncType 的设置值为：

设置值	描述
vbAsyncTypeFile	数据由用 VisualBasic 创建的文件提供。
vbAsyncTypeByteArray	数据作为字节数组提供，字节数组中包含检索到的数据。这里假设部件的创建者知道如何处理数据。
vbAsyncTypePicture	数据由 Picture 对象提供。

AsyncReadOptions 的设置值为：

常数	设置	描述
vbAsyncReadSynchronousDownload	1	AsyncRead 不返回，直到 AsyncReadComplete 事件发生。
vbAsyncReadOfflineOperation	8	AsyncRead 只使用本地缓存资源。
vbAsyncReadForceUpdate	16	AsyncRead 强迫从服务器上的一个资源下载，忽略本地缓存副本。
vbAsyncReadResynchronize	512	AsyncRead 仅当服务器版本较新时更新本地缓存的副本。
vbAsyncReadGetFromCacheIfNetFail	524288	如果服务器连接失败，AsyncRead 使用缓存的副本。

说明

可以通过 AsyncReadProgress 事件跟踪 AsyncRead 方法请求的下载进程。一旦数据可用，在对象中将产生 AsyncReadComplete 事件。在异步读取完成之前，可以通过调用 CancelAsyncRead 方法取消异步读取操作。

PropertyName 参数是一个标记，可以为任意的串，因为它唯一的

功能是充当这种特殊数据请求的标识符。PropertyName 中的数值用来标识 CancelAsyncRead 方法中要取消的某个异步读取，或者用来标识 AsyncReadComplete 事件中刚完成的某个异步读取。一次只能激活一个给定 PropertyName 的 AsyncRead 事件。

AsyncRead 方法初始化一个异步下载。如果数据在客户机上已经可以得到，AsyncRead 事件同时激发（在这个方法返回之前）。AsyncRead 能够同时引起某些错误（如“错误参数”、“未知协议”、“UrlMon.dll 丢失”等等），所以在调用 AsyncRead 之前有合适的错误处理代码是一个好想法。如果数据不能在本地得到，那么 AsyncRead 立即返回，AsyncRead 事件随后发生。

请参阅

AsyncReadComplete 事件，CancelAsyncRead 方法，Picture 对象

AsyncReadComplete 事件

当容器刚完成一个异步读取请求时，发生该事件。

应用于

UserControl 对象，UserDocument 对象

语法

Sub*object*_**AsyncReadComplete**(AsyncPropAsAsyncProperty)

AsyncReadComplete 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
asyncProp	AsyncProperty 对象

说明

AsyncProp 中的数值指定了已完成的某个异步数据读取请求，它与前一个 AsyncRead 方法调用中的数据匹配。

AsyncReadComplete 事件过程中应包含错误处理代码，因为错误状态会终止下载。如果发生了这种情况，当访问 AsyncProperty 对象的 Value 属性时将会发生错误。

请参阅

AsyncRead 方法，AsyncProperty 对象，AsyncType 属性

AsyncReadProgress 事件

当由于 AsyncRead 方法可以得到更多数据时发生。

应用于

UserControl 对象，UserDocument 对象

语法

Sub*object*_**AsyncReadProgress**(*AsyncProp*AsAsyncProperty)

AsyncReadProgress 事件语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
asyncProp	一个 AsyncProperty 对象

说明

AsyncReadProgress 事件使您可以访问异步下载的进行状态。

如果 AsyncRead 方法的 AsyncType 设置为 vbAsyncTypeFileName 或 vbAsyncTypeByteArray，在下载过程中您可以对已下载的部分数据进行访问。通常在整个下载过程中，VisualBasic 保持文件打开。为 vbAsyncTypeFileName 获取 Value 使下载过程变慢（但在较慢链接时并不明显）。因为 VisualBasic 在每次激发事件之前必须关闭文件，接着在下次可获得数据时再打开文件。

如果打开文件，则必须在执行 AsyncReadProgress 事件内代码之前（或在调用 DoEvents 之前，或以发送 Windows 消息方式）关闭它。否则将发生错误阻止激发 AsyncReadProgress 事件。为避免这些问题，在 AsyncReadComplete 事件过程中应包括错误处理代码。

注意 AsyncReadProgress 和 AsyncReadComplete 事件的激发是基于消息的，即一个 Windows 通知消息发布时，应用程序的消息处理器必须处理此消息来激发事件。因此，在这些事件的代码中应避免循环结构。也应避免使用 DoEvents，因为它将不可预料地引

起再次进入到代码中。

当下载完成时，AsyncReadProgress 和 AsyncReadComplete 事件都发生。（即 AsyncProp.StatusCode=vbAsyncStatusCodeEndDownload Data。）

请参阅

AsyncRead 方法，AsyncReadComplete 事件

AsyncType 属性

返回 Value 属性所返回的数据类型，或者对其进行设置。该属性仅仅是作为 AsyncRead 方法的一个参数来使用的。

应用于

AsyncProperty 对象

语法

object.**AsyncType**=*dataType*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
dataType	整数，它指定了数据类型，如下列“设置值”中所示

设置值

dataType 的设置值是：

常数	值	描述
vbAsyncTypePicture	0	缺省的。Picture 对象。
vbAsyncTypeFile	1	数据是由 VisualBasic 创建的文件提供。
vbAsyncTypeByteArray	2	数据是作为包含检索数据的字节数组提供的。

请参阅

AsyncRead 方法，AsyncReadComplete 事件

AtEndOfLine 属性

只读属性，在 TextStream 文件中，如果文件指针正好在行尾标记的前面，那么该属性值返回 True；否则返回 False。

应用于

TextStream 对象

语法

object.**AtEndOfLine**

object 总是一个 TextStream 对象的名称。

说明

AtEndOfLine 属性仅应用于已打开供读取的 TextStream 文件；否则就会出错。

下面的代码举例说明了 AtEndOfLine 属性的用法：

```
Dimfs,a,retstring
Setfs=CreateObject("Scripting.FileSystemObject")
Seta=fs.OpenTextFile("c:\testfile.txt",ForReading,False)
DoWhilea.AtEndOfLine<>True
retstring=a.Read(1)
...
Loop
a.Close
```

请参阅

AtEndOfStream 属性，Column 属性，Line 属性

AtEndOfStream 属性

只读属性，如果文件指针在 TextStream 文件末尾，则该属性值返回 True；否则返回 False。

应用于

TextStream 对象

语法

object.**AtEndOfStream**

object 总是一个 TextStream 对象的名称。

说明

AtEndOfStream 属性仅应用于已打开供读取的 TextStream 文件；否

则就会出错。

下面的代码举例说明了 AtEndOfStream 属性的用法：

```
Dimfs,a,retstring
Setfs=CreateObject("Scripting.FileSystemObject")
Seta=fs.OpenTextFile("c:\testfile.txt",ForReading,False)
DoWhilea.AtEndOfStream<>True
retstring=a.ReadLine
...
Loop
a.Close
```

请参阅

AtEndOfLine 属性，Column 属性，Line 属性

Atn 函数

返回一个 Double，指定一个数的反正切值。

语法

Atn(*number*)

必要的 number 参数是一个 Double 或任何有效的数值表达式。

说明

Atn 函数的参数值(number)为直角三角形两边的比值并返回以弧度为单位的角。这个比值是角的对边长度除以角的邻边长度之商。

值的范围在-pi/2 和 pi/2 弧度之间。

为了将角度转换为弧度，请将角度乘以 pi/180。为了将弧度转换为角度，请将弧度乘以 180/pi。

注意 Atn 是 Tan 的反三角函数，Tan 的参数值为角度，返回直角三角形的两条边的比值。不要将 Atn 和余切函数混淆，余切函数值是正切函数值的倒数，cotangent=(1/tangent)。

请参阅

Cos 函数，Sin 函数，Tan 函数

Attributes 属性

设置或者返回文件或文件夹的属性。读/写或只读，取决于属性。

应用于

File 对象，Folder 对象

语法

object.**Attributes**[=*newattributes*]

Attributes 属性有下列几个部分：

部分	描述
object	必需的。总是某个 File 或者 Folder 对象的名字
newattributes	可选的。如果提供的话，newattributes 就是所指定 object 的新属性值

设置

newattributes 参数可以是具有下列值中的任意一个或任意的逻辑组合：

常数	值	描述
normal	0	一般文件。未设置属性。
readOnly	1	只读文件。属性为读/写。
hidden	2	隐藏文件。属性为读/写。
system	4	系统文件。属性为读/写。
volume	8	磁盘驱动器卷标。属性为只读。
directory	16	文件夹或目录。属性为只读。
archive	32	自上次备份后已经改变的文件。属性为读/写。
alias	64	链接或快捷方式。属性为只读。
compressed	128	压缩文件。属性为只读。

说明

下面的代码用一个文件举例说明了 Attributes 属性的用法：

```
SubSetClearArchiveBit(filespec)
Dimfs,f,r
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFile(fs.GetFileName(filespec))
Iff.attributesand32Then
r=MsgBox("TheArchivebitisset,doyouwanttoclearit?",vbYesNo,"Set/C
learArchiveBit")
Ifr=vbYesThen
f.attributes=f.attributes-32
```

```
MsgBox"Archivebitiscleared."
Else
MsgBox"Archivebitremainsset."
EndIf
Else
r=MsgBox("TheArchivebitisnotset.Doyouwanttosetit?",vbYesNo,"Set/
ClearArchiveBit")
Ifr=vbYesThen
f.attribute4s=f.attributes+32
MsgBox"Archivebitisset."
Else
MsgBox"Archivebitremainsclear."
EndIf
EndIf
EndSub
```

请参阅

DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性，ParentFolder 属性，Path 属性，ShortName 属性，ShortPath 属性，Size 属性，SubFolders 属性，Type 属性

AutoActivate 属性

返回或设置一个值，允许通过双击 OLE 容器控件或将焦点移到 OLE 容器控件，来激活对象。

应用于

OLE 容器控件

语法

object.**AutoActivate** [=*value*]

AutoActivate 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	整数或常数，指定将 OLE 容器控件内的对象激活的技术，在“设置值”中有详细说明

设置值

value 的设置值是：

常数	值	描述
VbOLEActivateManual	0	手工的。对象不能自动激活。可以使用程序的 DoVerb 方法激活对象。
VbOLEActivateGetFocus	1	焦点的。如果 OLE 容器控件包含的对象支持单击激活，当 OLE 容器控件接收焦点时，将提供对象的应用程序激活。
VbOLEActivateDoubleclick	2	（缺省值）双击。如果 OLE 容器控件包含对象，当控件有了焦点，在 OLE 容器控件上双击或按 ENTER 键时，将提供对象的应用程序激活。
VbOLEActivateAuto	3	自动的。如果 OLE 容器控件包含对象，当控件接收焦点或当双击控件时，均根据对象规范的激活方法，将提供对象的应用程序激活。

说明

通过检查 OLEType 属性可以确定 OLE 容器控件是否包含对象。注意当 AutoActivate 被置为 2（双击）时，如果在 OLE 容器控件上双击，不会产生 DblClick 事件。

请参阅

OLEType 属性，DoVerb 方法

AutoRedraw 属性

返回或设置从图形方法到持久图形.的输出。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Form 对象，Forms 集合，PictureBox 控件

语法

object.**AutoRedraw**[=*boolean*]

AutoRedraw 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指定如何重绘对象，“设置值”中有详细描述

设置值

boolean 的设置值如下：

设置值	描述
True	使 Form 对象或 PictureBox 控件的自动重绘有效。图形和文本输出到屏幕，并存储在内存的图象中。该对象不接受绘制事件，必要时，用存储在内存中的图象进行重绘。
False	（缺省值）使对象的自动重绘无效，且将图形或文本只写到屏幕上。当需要重画该对象时，VisualBasic 会激活对象绘制事件。

说明

使用下列图形方法工作，如：Circle、Cls、Line、Point、Print 和 Pset，该属性极为重要。利用这些方法，在改变对象大小或隐藏在另一个对象后又重新显示的情况下，设置 AutoRedraw 为 True，将在 Form 或 PictureBox 控件中自动重绘输出。

运行时在程序中设置 AutoRedraw，可以在画持久图形（如背景色或网格）和临时图形之间切换。如果设置 AutoRedraw 为 False，以前的输出成为背景屏幕的一部分。当 AutoRedraw 设置为 False，用 Cls 方法清除绘图区时不会删除背景图形。把 AutoRedraw 改回 True 后，再用 Cls 将清除背景图形。

注意如果设置 BackColor 属性，所有图形和文本，包括持久图形，都被清除。一般来说，除非 AutoRedraw 设置为 True，所有图形都需用 Paint 事件显示。

要取回在 AutoRedraw 设置为 True 时创建的持久图形，用 Image 属性。当 AutoRedraw 设置为 True 时，用对象的 hDC 属性可以将持久图形传送给 WindowsAPI。

如果设置窗体的 AutoRedraw 属性为 False，然后最小化该窗体，则将 ScaleHeight 和 ScaleWidth 属性设置为图标大小。在 AutoRedraw 设置为 True 时，ScaleHeight 和 ScaleWidth 保持为恢复窗口的尺寸。如果设置 AutoRedraw 属性为 False，Print 方法将在诸如 Image 和 Shape 等图形控件的顶部附近印。

请参阅

Paint 事件，Cls 方法，Point 方法，hDC 属性，Image 属性，ScaleHeight,ScaleWidth 属性

示例

这个例子是在 PictureBox 控件中交替显示两个图形：一个永久的实心圆和临时的垂直线。单击 PictureBox 画或重画这些线。调整窗体的大小要求重画临时的图形。要试用此例，先把代码粘贴到窗体的声明部分，该窗体有一个名叫 Picture1 的 PictureBox 控件。然后按下 F5 键运行该程序，每次调整窗体大小时单击图形。

```
PrivateSubForm_Load()
   Picture1.ScaleHeight=100 '设置比例为 100.
   Picture1.ScaleWidth=100
   Picture1.AutoRedraw=True '打开 AutoRedraw.
   Picture1.ForeColor=0'设置 ForeColor.
```

```
    Picture1.FillColor=QBColor(9)    '设置FillColor.
    Picture1.FillStyle=0'设置FillStyle.
    Picture1.Circle(50,50),30    '画一个圆.
    Picture1.AutoRedraw=False    '关闭AutoRedraw.
EndSub

PrivateSubPicture1_Click()
    DimI'Declarevariable.
    Picture1.ForeColor=Rgb(Rnd*255,0,0)  '选择随机颜色.
    ForI=5To95Step10'画线.
        Picture1.Line(I,0)-(I,100)
    Next
EndSub
```

AutoShowChildren 属性

返回或设置一个值，确定在加载 MDI 子窗体时是否显示它。

应用于

MDIForm 对象

语法

object.AutoShowChildren[=*boolean*]

AutoShowChildren 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指定 MDI 子窗体是否自动可见，“设置值”中有详细描述

设置值

boolean 的设置值如下:

设置值	描述
True	（缺省值）MDI 子窗体在加载时自动显示。
False	MDI 子窗体在加载时不会自动显示。

说明

可以用 AutoShowChildren 属性加载 MDI 子窗体，并保持其隐藏状态直到用 Show 方法将它显示出来。

请参阅

MDIChild 属性，ActiveForm 属性

示例

这个例子介绍有 MDI 子窗体的 MDIForm 对象，利用 AutoShowChildren 属性创建一个隐藏的窗体作为 MDI 子窗体的另一个实例，然后创建一个可见的 MDI 子窗体。要试用此例，在 Form1 中设置 MDIChild 属性为 True，然后用“工程”菜单的“添加 MDI 窗体”命令创建一 MDIForm。拷贝代码到 MDIForm 的声明部分，然后按 F5 键运行该程序。

```
PrivateSubMDIForm_Load()
   MDIForm1.AutoShowChildren=False '设置为隐藏的子窗体.
   DimHideFormAsNewForm1    '声明新的窗体.
   HideForm.Caption="HideForm" '设置它的标题.
   LoadHideForm'加载;其为隐藏窗体.
   MDIForm1.AutoShowChildren=True   '设置显示子窗体.
   DimShowFormAsNewForm1    '声明另一个新的窗体.
   ShowForm.Caption="ShowForm" '设置它的标题.
   LoadShowForm'加载;并显示窗体.
EndSub
```

AutoSize 属性

返回或设置一个值，以决定控件是否自动改变大小以显示其全部内容。

应用于

Label 控件，PictureBox 控件

语法

object.**AutoSize** [=*boolean*]

AutoSize 属性语法具有下列组成部分:

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指定是否改变控件的尺寸，“设置值”中有详细描述

设置值

boolean 设置值如下：

设置值	描述
True	自动改变控件大小以显示全部内容。
False	（缺省值）保持控件大小不变。超出控件区域的内容被裁剪掉。

请参阅

Resize 事件，Alignment 属性

AutoVerbMenu 属性

返回或设置一个值，用于确定当用鼠标右键单击 OLE 容器控件时，是否显示含有对象谓词的弹出式菜单。

应用于

OLE 容器控件，OLEObject 对象

语法

object.**AutoVerbMenu**[=*boolean*]

AutoVerbMenu 属性的语法包含 描述

下面部分：部分

object	**对象表达式，其值是“应用于”列表中的一个对象**
boolean	**布尔表达式，指示是否显示弹出式菜单，在“设置值”中有详细说明**

设置值

boolean **的设置值是：**

设置	**描述**
True	**（缺省值）当用鼠标右键单击** OLE **容器控件时，显示弹出式菜单，显示对象支持的命令。**
False	**不显示弹出式菜单。**

说明

当这个属性设置为 True **时，如果用鼠标右键单击** OLE **容器控件，不发生** Click **事件和** MouseDown **事件。**

为了显示用户自己的菜单，必须将 AutoVerbMenu **属性置为** False**。**

请参阅

ObjectVerbs 属性，objectVerbsCount 属性，Verb 属性

AvailableSpace 属性

返回在指定的驱动器或网络共享上的用户可用空间容量。

应用于

Drive 对象

语法

object.**AvailableSpace**

object 总是一个 Drive 对象。

说明

一般来说，AvailableSpace 属性的返回值与 FreeSpace 属性的返回值是相同的。对于支持限额的计算机系统来说，两个值之间可能会有所不同。

下面的代码举例说明了 AvailableSpace 属性的用法：

```
SubShowAvailableSpace(drvPath)
Dimfs,d,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(fs.GetDriveName(drvPath))
s="Drive"&UCase(drvPath)&"-"
s=s&d.VolumeName&vbCrLf
s=s&"AvailableSpace:"&FormatNumber(d.AvailableSpace/1024,0)
s=s&"Kbytes"
MsgBoxs
```

EndSub

请参阅

DriveLetter 属性，DriveType 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，Paty 属性，RootFolder 属性，SerialNumber 属性，ShareName 属性，TotalSize 属性，VolumeName 属性

BackColor、ForeColor 属性

?BackColor—返回或设置对象的背景颜色。

?ForeColor—返回或设置在对象里显示图片和文本的前景颜色。

应用于

Function 控件（数据报表设计器），Image 控件（数据报表设计器），Label 控件（数据报表设计器），Shape 控件（数据报表设计器），TextBox 控件（数据报表设计器），AmbientProperties 对象，PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，Printers 集合，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，OLEContainer 控件，ADOData 控件，ImageList 控件，Animation 控件，CoolBar 控件，Band 对象，DBCombo 控件，DBList 控件，SSTab 控件，RemoteData 控件，RichTextBox 控件，Data 控件

语法

object.**BackColor**[=*color*]

object.**ForeColor**[=*color*]

BackColor 和 ForeColor 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
color	值或常数，确定对象前景或背景的颜色，“设置值”中有详细说明

设置值

VisualBasic 用 MicrosoftWindows 运行环境的红-绿-蓝(RGB)颜色方案。color 的设置值如下：

设置值	描述
标准 RGB 颜色	使用调色板或在代码中使用 RGB 或 QBColor 函数指定的颜色。
系统缺省颜色	由对象浏览器中的 VisualBasic(VB)对象库所列的系统颜色常数指定的颜色。Windows 运行环境替换使用用户在控制面板设置值中的选择。

对所有的窗体和控件，在设计时的缺省设置值如下：

?BackColor 一设置为由常数 vbWindowBackground 定义的系统缺省颜色。

?ForeColor 一设置为由常数 vbWindowText 定义的系统缺省颜色。

说明

在 Label 和 Shape，控件中，如果 BackStyle 属性的设置值为 0（透明），则忽略 BackColor 属性。

如果在 Form 对象或 Picturebox 控件中设置 BackColor 属性，则所有的文本和图片，包括指定的图片，都被擦除。设置 ForeColor 属性值不会影响已经绘出的图片或打印输出。在其它的所有控件中，屏幕的颜色会立即改变。

标准 RGB 颜色的有效取值范围是 0 到 16,777,215(&HFFFFFF)。该范围内数的高字节为 0；较低的 3 个字节，从最低字节到最高字节依次决定红、绿和蓝的量。红、绿和蓝的成分，分别由一个介于 0 与 255(&HFF)之间的数来表示。如果最高字节不为 0，VisualBasic 将使用系统颜色，这些颜色由用户的控制面板设置值和由对象浏览器中的 VisualBasic(VB)对象库所列出的常数来确定。

在 Windows 运行环境中显示文本，文本和背景的颜色必须都是原色。如果所选择的文本或背景颜色没有显示出来，则选择颜色中可能有抖动色－也就是说，最多由三种不同颜色的像素组成的颜色。如果对文本或背景选择了抖动色，则会用最接近的原色来代替。

请参阅

FillColor 属性，FillStyle 属性，TreeView 控件

示例

这个例子每秒钟两次随机地重新设置窗体(form)和 PictureBox 控件的前景颜色和背景颜色。要尝试这个例子，请将代码粘贴到包

含 PictureBox 控件和 Timer 控件的窗体的声明部分，然后按 F5 键。

```
PrivateSubForm_Load()
   Timer1.Interval=500
EndSub

PrivateSubTimer1_Timer()
   BackColor=QBColor(Rnd*15)

   Picture1.BackColor=QBColor(Rnd*15)
   Picture1.ForeColor=QBColor(Rnd*10)
EndSub
```

BackStyle 属性

返回或设置一个值，它指定 Label 控件或 Shape 控件的背景是透明的还是非透明的。

应用于

Function 控件（数据报表设计器），Image 控件（数据报表设计器），Label 控件（数据报表设计器），Shape 控件（数据报表设计器），TextBox 控件（数据报表设计器），UserControl 对象，Label 控件，Shape 控件，OLEContainer 控件

语法

object.**BackStyle**[=*number*]

BackStyle 语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	数值表达式，用于确定透明性，“设置值”中有详细说明

设置值

number 的设置值如下：

设置值	描述
0	透明－在控件后的背景色和任何图片都是可见的。
1	（缺省值）非透明－用控件的 BackColor 属性设置值填充该控件，并隐藏该控件后面的所有颜色和图片。

说明

在 Form 对象或 Picturebox 控件上使用背景色、或在图片上放置控件时，可以利用 BackStyle 属性来创建透明控件；如果想要突出控件，可以使用非透明控件。

如果 BackStyle=0，则忽略该控件的 BackColor 属性。

请参阅

BackColor,ForeColor 属性

BackStyle 属性（UserControl 对象）

返回或设置指示控件背景类型的数值。

应用于

UserControl 对象

语法

object.BackStyle[=*enum*]

BackStyle 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
enum	枚举值，它决定控件背景的显示方式，如“设置值”中所描述

设置值

enum 的设置值为：

设置值	描述
1-Opaque	（缺省）不透明的背景。控件接收所有的鼠标事件。
2-Invisible	只有当 Windowless 属性设置为 True 才能应用。否则行为与透明 BackStyle 一样。

说明

当 *enum* 设置为 2，则 Windowless 属性设为 True 的控件外观和行为基于 MaskPicture、MaskColor、HitBehavior 和 ClipBehavior 属性的设置。

请参阅

Animation 控件，Center 属性

BaseWindow 属性

从 DHTML 对象模型返回最高层的窗口对象。最高层的窗口表示浏览器窗口和它所包含的任何帧。

应用于

DHTMLPage 对象

语法

object.BaseWindow

object 所在处表示一个对象表达式，其值为“应用于”列表中的一个对象。

说明

在最高层窗口（例如：history、event、navigator 等等）中可以得到的对象总是可以通过 BaseWindow 属性得到。VisualBasic 中的 BaseWindow 与 InternetExplorer4.0 浏览器的 DynamicHTML 对象模型中的 Window 对象相对应。

请参阅

《MicrosoftVisualBasic6.0 组件工具指南》第 5 部分“创建 Internet 应用程序”的第 2 章“开发 DHTML 应用程序”中的“DHTML 应用程序中的关键事件”，“VisualBasic 中的动态 HTML 对象模型”和“DHTML 应用程序概述”。

Beep 语句

通过计算机喇叭发出声音。

语法

Beep

说明

呼叫的频率与时间长短取决于硬件和系统软件，并随电脑不同而不同。

示例

本示例使用 Beep 语句让计算机连续响三声。

```
Dim I
For I=1 To 3  '循环 3 次。
   Beep '发出一声。
Next I
```

BeforeLoadFile 事件

该事件发生在当部件被添加（不打开）到工程中，或与部件的相关联的二进制文件（如.Frx 文件）被访问时。

应用于

FileControlEvents 对象

语法

Sub*object*_**BeforeLoadFile**(*vbproject* As VBProject , *filenames*() As String)

BeforeLoadFile 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbproject	VBProject 对象，指出其中已被加载文件的工程名
filenames	字符串表达式，指出要加载的文件名

说明

该事件发生在所有连接到 FileControl 对象的外接程序中，该事件对工程发生多次：对工程文件发生一次；对所有的窗体、模块、类、用户控件、属性页和控件文件发生一次；对每一个 .Frx 文件发生一次。如果窗体文件连同相关的 .Frx 文件被保存时该事件也发生，因为当 .Frm 文件被保存时 .Frx 文件被加载。

该事件发生在所有连接到 FileControl 对象的外接程序中。外接程序不能阻止文件写盘，因为写盘操作已经完成。然而，可以用该事件执行其它任务，如：

记录关于该事件的信息。

更新有关该文件的信息。

备份该文件。

BeginRequest 事件

当用户在 HTML 页面上选择一个元素时发生，该 HTML 页面发送请求给 WebClass 对象。标志着对一个 HTTP 请求处理的开始。

应用于

WebClass 对象

语法

PrivateSub*object*_BeginRequest()

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

当接收到一个 HTTP 请求时，此事件发生在其他所有事件之前。

请参阅

《MicrosoftVisualBasic6.0 组件工具指南》第 5 部分“创建 Internet 应用程序”中第 3 章“开发 IIS 应用程序”中的“WebClass 生命周期”

Bindable 属性

返回与 Member 对象相关联的 Bindable 属性，或者对其进行设置。

应用于

Member 对象

语法

object.Bindable

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Binding 对象

代表一个特定的数据使用者属性到一个数据源的数据字段的运行时绑定。

语法

Binding

说明

Binding 集合中的每一个 Binding 对象都代表一个数据使用者（例如控件）与该使用者的单个属性（例如 Text 属性）的组合，绑定到一个特定的数据源字段。

属性

Key 属性，DataFormat 属性，PropertyName 属性，Object 属性，DataChanged 属性，DataField 属性

示例

本例使用 BindingCollection 对象把一个数据源绑定到两个 TextBox 控件。首先打开一个 ADODB 记录集对象，然后设置 BindingCollection 的 DataSource 属性为该记录集。随后，程序代码把两个 Binding 对象添加到该集合，同时也就把这两个 TextBox 控件绑定到该记录集的不同字段了。

要试验该例，在“引用...”对话框设置对 MicrosoftDataBinding Collection 的引用。在同一个对话框中，设置对 Microsoft ActiveXDataObjectsLibrary 的引用。在窗体上绘制两个 TextBox 控件，把代码粘贴到“声明”部分。按 F5 键，并单击窗体移动到记录

集的下一条记录。

```
OptionExplicit
PrivatecolBndNwindAsNewBindingCollection
PrivatersNwindAsNewADODB.Recordset
PrivatecnAsNewADODB.Connection

PrivateSubForm_Load()

    '设置 Connection 对象参数。
    Withcn
        '下列的连接在您的计算机上可能能够正常工作，也可能不能正常工作。
        '请改变它以定位 Nwind.mdb 文件。
        '该文件包括在 VisualBasic 中。
        .Provider="Microsoft.Jet.OLEDB.3.51"
        .Open"C:\ProgramFiles\DevStudio\VB\Nwind.mdb"
    EndWith

    '打开该记录集对象。
    rsNwind.Open"Select*FromProducts",cn

    '设置 Bindings 集合的 DataSource 为该记录集。
```

```
    SetcolBndNwind.DataSource=rsNwind

    '添加到 Bindings 集合。
    WithcolBndNwind
        .AddText1,"Text","ProductName",,"product"
        .AddText2,"Text","SupplierID",,"ID"
    EndWith

    '打印集合中对象的属性。
    DimbndObjAsBinding
    ForEachbndObjIncolBndNwind
        Debug.Print"DataField","PropertyName","Key"

    Debug.PrintbndObj.DataField,bndObj.PropertyName,bndObj.Key
        Debug.Print
    Next
EndSub

PrivateSubForm_Click()
    '单击窗体移动到下一条记录。
    rsNwind.MoveNext
EndSub
```

BindingCollection 对象

一个 Binding 对象的集合。

语法

BindingCollection

说明

BindingCollection 对象允许把任意的数据供应程序绑定到任意的数据使用者。要把使用者绑定到数据供应程序，使用 Add 方法向该集合添加一个 Binding 对象。每一个 Binding 对象代表一个特定的使用者到 BindingCollection 对象提供的 DataSource 的绑定。

没有设计时界面的数据源，例如一个配置为数据源的 Class（通过设置它的 DataSourceBehavior 属性为 VbDataSource），或者一个 ADORecordset，可以在运行时使用 BindingObject 集合来绑定。

使用标准的集合语法返回或设置集合成员的属性。

属性

DataMember 属性，UpdateMode 属性，Count 属性（VB 集合），Item 属性，DataSource 属性

方法

Add 方法（Binding 集合），Remove 方法。

示例

本例使用 BindingCollection 对象绑定数据源到两个 TextBox 控件。本例首先打开一个 ADODB 记录集对象，然后将

BindingCollection 的 DataSource 属性设置给记录集。然后本代码向该集合添加两个 Binding 对象，这样就将两个 TextBox 控件绑定到记录集的不同域中。

要试用此例，在 References 对话框中为 MicrosoftDataBinding Collection 设置一个引用。在同一对话框中设置 MicrosoFtActiveXDataObjectsLibrary 的引用。将两个 TextBox 控件拖到窗体上，将本代码粘贴到声明部分。按 F5，然后单击窗体。

```
OptionExplicit
PrivatecolBndNwindAsNewBindingCollection
PrivatersNwindAsNewADODB.Recordset
PrivatecnAsNewADODB.Connection

PrivateSubForm_Load()

    '设置 Connection 对象参数。
    Withcn
        '下列的连接在您的计算机上可能能够正常工作，也可能不能正常工作。
        '请改变它以定位 Nwind.mdb 文件。
        '该文件包括在 VisualBasic 中。
        .Provider="Microsoft.Jet.OLEDB.3.51"
        .Open"C:\ProgramFiles\DevStudio\VB\Nwind.mdb"
    EndWith
```

```
    '打开该记录集对象。
    rsNwind.Open"Select*FromProducts",cn

    '设置 Bindings 集合的 DataSource 为该记录集。
    SetcolBndNwind.DataSource=rsNwind

    '添加到 Bindings 集合。
    WithcolBndNwind
        .AddText1,"Text","ProductName",,"product"
        .AddText2,"Text","SupplierID",,"ID"
    EndWith

    '打印集合中对象的属性。
    DimbndObjAsBinding
    ForEachbndObjIncolBndNwind
        Debug.Print"DataField","PropertyName","Key"

    Debug.PrintbndObj.DataField,bndObj.PropertyName,bndObj.Key
        Debug.Print
    Next
EndSub
```

```
PrivateSubForm_Click()
    '单击窗体移动到下一条记录。
    rsNwind.MoveNext
EndSub
```

Bold 属性

返回或设置 Font 对象的字形是粗体或非粗体。应用于 Font 对象。

应用于

Font 对象

语法

object.**Bold**[=*boolean*]

Bold 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定字形的布尔表达式，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
True	打开粗体格式化。
False	（缺省值）关闭粗体格式化。

说明

Font 对象在设计时不能直接使用。取而代之的是，在“属性”窗口中通过选择控件的 Font 属性并单击“属性”按钮来设置 Bold 属性。在“字体”对话框的“字形”框中，选择“粗体”或“粗斜体”。然而，在运行时，通过为 Font 对象指定 Bold 属性值可直接设置 Bold。

请参阅

FontTransparent 属性,Italic 属性,Size 属性(Font),Strike Trrough 属性,Underline 属性,Weight 属性,Name 属性

示例

这个例子完成下面的功能：每次单击鼠标时在窗体上打印文本。要试用此例，可以先将下面的代码粘贴到一个窗体的声明部分中，然后按下 F5 键并双击此窗体。

```
PrivateSubForm_Click()
   Font.Bold=NotFont.Bold   '转换 Bold。
   Font.StrikeThrough=NotFont.StrikeThrough'转换 StrikeThrough。
   Font.Italic=NotFont.Italic   '转换 Italic。
   Font.Underline=NotFont.Underline'转换 Underline。
```

```
    Font.Size=16'设置 Size 属性。
    IfFont.BoldThen
        Print"Fontweightis"&Font.Weight&"(bold)."
    Else
        Print"Fontweightis"&Font.Weight&"(notbold)."
    EndIf
EndSub
```

BorderColor 属性

返回或设置对象的边框颜色。

应用于

Function 控件(DataReportDesigner),Image 控件(DataReportDesigner),Label 控件(DataReportDesigner),Line 控件(DataReportDesigner),Shape 控件(DataReportDesigner),TestBox 控件(DataReportDesigner),Line 控件,Shape 控件

语法

object.**BorderColor**[=*color*]

BorderColor 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
color	值或常数，用来确定边框颜色，“设置值”中有详细说明

设置值

VisualBasic 使用 MicrosoftWindows 运行环境的红-绿-蓝(RGB)颜色方案。color 的设置值如下：

设置值	描述
标准 RGB 颜色	使用调色板或在代码中使用 RGB 或 QBColor 函数指定的颜色。
系统缺省颜色	由系统颜色常数指定的颜色，这些常数在对象浏览器中的 VisualBasic(VB)对象库中列出。系统的缺省颜色由 vbWindowText 常数指定。Windows 运行环境替换使用用户在控制面板设置值中的选择。

说明

正常 RGB 颜色的有效取值范围是 0 到 16,777,215(&HFFFFFF)。该范围内数的高字节为 0；较低的 3 个字节，从最低字节到最高字节依次决定红、绿和蓝的量。红、绿和蓝的成分分别由一个 0 到 255(&HFF)之间的数表示。如果最高字节不为 0，VisualBasic 使用系统颜色，该颜色由用户在“控制面板”设置和由“对象浏览器”中的 VisualBasic(VB)对象库中列出的常数确定。

请参阅

BackColor,forecolor 属性,DrawWidth 属性,BackColor,ForeColor 属性(ActiveX 控件)

BorderStyle 属性

返回或设置对象的边框样式。对 Form 对象和 Textbox 控件在运行时是只读的。

应用于

Function 控件(DataReportDesigner),Image 控件(DataReportDesigner),Label 控件(DataReportDesigner),Line 控件(DataReportDesigner),Shape 控件(DataReportDesigner),User Control 对象,Form 对象,Forms 集合,Frame 控件,Image 控件,Label 控件,Line 控件,PictureBox 控件,Shape 控件,TextBox 控件,OLE 容器控件,Slider 控件,MSChart 对象。

语法

object.**BorderStyle**=[*value*]

BorderStyle 属性语法有这些组成部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	值或常数，用于决定边框样式，“设置值”中有详细说明

设置值

Form 对象的 BorderStyle 属性设置值如下：

常数	设置值	描述
vbBSNone	0	无（没有边框或与边框相关的元素）。
vbFixedSingle	1	固定单边框。可以包含控制菜单框，标题栏，“最大化”按钮，和“最小化”按钮。只有使用最大化和最小化按钮才能改变大小。
VbSizable	2	（缺省值）可调整的边框。可以使用设置值 1 列出的任何可选边框元素重新改变尺寸。
VbFixedDouble	3	固定对话框。可以包含控制菜单框和标题栏，不能包含最大化和最小化按钮，不能改变尺寸。
VbFixedToolWindow	4	固定工具窗口。不能改变尺寸。显示关闭按钮并用缩小的字体显示标题栏。窗体在 Windows95 的任务条中不显示。
VbSizableToolWindow	5	可变尺寸工具窗口。可变大小。显示关闭按钮并用缩小的字体显示标题栏。窗体在 Windows95 的任务条中不显示。

MSFlexGrid、Image、Label、OLE 容器、PictureBox、Frame 和 TextBox 控件的 BorderStyle 属性设置值如下：

设置值	描述
0	（Image 和 Label 控件的缺省值）无。
1	（MSFlexGrid、PictureBox、TextBox 和 OLE 容器控件的缺省值）固定单边框。

Line 和 Shape 控件的 BorderStyle 属性设置值如下：

常数	设置值	描述
VbTransparent	0	透明
VbBSSolid	1	（缺省值）实线。边框处于形状边缘的中心。
VbBSDash	2	虚线
VbBSDot	3	点线
VbBSDashDot	4	点划线
VbBSDashDotDot	5	双点划线
VbBSInsideSolid	6	内收实线。边框的外边界就是形状的外边缘。

说明

对于窗体，BorderStyle 属性决定了其主要特征，这些特征从外观上就能确定窗体是通用窗口或对话框。设置值 3（固定对话框）用于标准对话框。设置值 4（固定工具窗口）和 5（可变工具窗口）用于创建工具箱样式的窗口。

设置值为 2（可变尺寸）的 MDI 子窗体，以 Windows 运行环境运

行时定义的缺省尺寸，在 MDI 窗体内显示。对于任何其它设置值，窗体按设计时指定的大小显示。

改变 Form 对象的 BorderStyle 属性设置值，可能会改变 MinButton、MaxButton 和 ShowInTaskbar 属性的设置值。当 BorderStyle 设置为 1（固定单边框）或 2（可变尺寸）时，MinButton、MaxButton 和 ShowInTaskbar 属性自动设置为 True。当 BorderStyle 设置为 0（无）、3（固定对话框）、4（固定工具窗口）或 5（可变工具窗口）、MinButton、MaxButton 和 ShowInTaskbar 属性自动设置为 False。

注意如果带有菜单的窗体设置为 3（固定对话框），该窗体将按设置值 1（固定单边框）显示。

运行时，窗体或者是模式的或者是无模式的，都可以用 Show 方法指定。

BorderWidth 属性

返回或设置控件边框的宽度。

应用于

Line 控件,Shape 控件

语法

object.**BorderWidth**[=*number*]

BorderWidth 属性语法有这些　描述

组成部分：部分

object	**对象表达式，其值是“应用于”列表中的一个对象**
number	**数值表达式，其值从** 1 **到** 8192**，包括** 1 **和** 8192

说明

用 BorderWidth **和** BorderStyle **属性来指定所需的** Line **或** Shape **控件边框类型，下表给出了** BorderStyle **设置值对** BorderWidth **属性的影响：**

边框样式	**对** BorderWidth **的影响**
0	**忽略** BorderWidth **设置。**
1-5	**边框宽度从边框中心扩大，控件的宽度和高度从边框的中心度量。**
6	**边框的宽度在控件上从边框的外边向内扩大，控件的宽度和高度从边框的外面度量。**

如果 BorderWidth **属性设置大于** 1**，有效的** BorderStyle **设置值为** 1 **（实线）和** 6 **（内收实线）。**

请参阅

BorderStyle 属性,DrawWidth 属性,Height,Width 属性,Borderstyle 属性(ActiveX 控件)

示例

这个例子使用两个 ComboBox 控件来为 Shape 控件的边框选定不同的宽度和样式。要尝试这个例子，请将代码粘贴到包含 Shape 控件和 ComboBox 控件的窗体的声明部分。对 ComboBox，将设置 Style=2 和 Index=0（创建一个控件数组），然后按 F5 键并单击窗体

```
PrivateSubForm_Load()
    Combo1(0).Width=1440*1.5
    LoadCombo1(1)
    Combo1(1).Top=Combo1(0).Top+Combo1(0).Height*1.5
    Combo1(1).Visible=True
    ForI=0To6
        Combo1(0).AddItem"BorderStyle="&I
    NextI
    ForI=1To10
        Combo1(1).AddItem"BorderWidth="&I
    NextI
    Combo1(0).ListIndex=1
    Combo1(1).ListIndex=0
EndSub

PrivateSubCombo1_Click(IndexAsInteger)
    IfIndex=0Then
```

```
        Shape1.BorderStyle=Combo1(0).ListIndex
    Else
        Shape1.BorderWidth=Combo1(1).ListIndex+1
    EndIf
EndSub
```

BottomMargin、TopMargin 属性

以缇为单位返回或设置底边距和顶边距的高度。

应用于

DataReport 对象

语法

object.**BottomMargin**[=*number*]

object.**TopMargin**[=*number*]

BottomMargin 和 TopMargin 属性包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
number	可选的。一个数值表达式，指定了底边距和顶边距的高度

Browsable 属性

返回与 Member 对象相关联的 Browsable 属性，或者对其进行设置。

应用于

Member 对象

语法

object.**Browsable**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

BrowserType 属性

返回活动服务器页面的 BrowserType 对象。

应用于

WebClass 对象

语法

***object*.BrowserType**

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

WebClass 使用 BrowserType 对象决定用户浏览器的属性，并基于这些属性作出处理决定。

关于 BrowserType 对象的属性、方法、和事件的详细信息，请参

阅活动服务器页面文档。

请参阅

《MicrosoftVisualBasic6.0 部件工具指南》一书的第五部分的“开发 IIS 应用程序”第三章的“IIS 应用程序的对象模型”的“构造 Internet 应用程序”。

BuildFile 属性

该属性设置了一个 HTML 文件的路径和文件名，页面设计器将编译之后的结果保存在该位置。它在运行时是是不可使用的。

应用于

DHTMLPageDesigner 对象

说明

该属性的初始路径设置值与 SourceFile 属性相同。该属性可以被修改成可以用来保存编译之后的 HTML 文件的任何位置。

构造位置必须是绝对路径，它需要指出完整的驱动器盘符和用来保存编译之后的文件的目录结构。如果您将来需要与其他开发者共享自己的工程，而该开发者使用的目录结构或者驱动器名称与您原来使用的不同，那么后来的开发者将需要修改 BuildFile 属性的值，使之反映他的计算机上编译的 HTML 文件的位置。运行时用户并不会受到这种限制。

请参阅

《MicrosoftVisualBasic6.0 部件工具指南》一书第五部分“开

发 DHTML 应用程序”第二章“构造 DHTML 应用程序”的“构造 Internet 应用程序”。

BuildFileName 属性

该属性返回编译工程时使用的可执行文件名或动态链接库名。

应用于

VBProject 对象

语法

object.**BuildFileName**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

BuildPath 方法

追加一个名字到一个已经存在的路径。

应用于

FileSystmeObject 对象

语法

object.**BuildPath**(*path*,*name*)

BuildPath 方法语法有如下几部分：

部分	描述
object	必需的。始终是 FileSystemObject 的名字
path	必需的。要对其追加名字的已存在路径。路径可以是绝对的也可以是相对的，不必指定一个已存在的文件夹
name	必需的。要追加到已存在路径上的名字

说明

只有在需要时，BuildPath 方法才在已存在路径和名字之间插入一个附加的路径分隔符。

BuiltIn 属性

返回一个布尔值，指出该引用是否是不能被删除的缺省引用。此属性为只读。

应用于

Reference 对象

返回值

BuiltIn 属性返回下列值：

值	描述
True	该引用是不能被删除的缺省引用。
False	该引用不是缺省引用，可以被删除。

请参阅

Remove 方法(VBAAdd-InObjectModel),References 集合,Descri

ption 属性,FullPath 属性,GUID 属性,IsBroken 属性

示例

下列示例使用 BuiltIn 属性来返回一个 Boolean 值，以判断当前活动工程中的某引用是否为内置的。

```
Debug.PrintApplication.VBE.ActiveVBProject.References(1).BuiltIn
```

BytesMax 属性

返回估计读取的最大字节数。

应用于

AsyncProperty 对象

语法

object.**BytesMax**

BytesMax 属性语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象

说明

BytesMax 属性返回一个长整数。如果可以得到更准确的估计，在下载过程中 BytesMax 的值可以改变。在某些下载中如果服务器不能决定下载的大小这个值可能为 0。例如，ASP 主页并不总是提供这个信息。

BytesRead 属性

返回现已读取或下载的字节总数。

应用于

AsncProperty 对象

语法

object.**BytesRead**

BytesRead 属性语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象

说明

BytesRead 属性返回一个长整数。

Calendar 属性

返回或设置一个值，指出工程中所使用日历的类型。

可以为 Calendar 使用下列设置之一:

设置	值	描述
vbCalGreg	0	使用 Gregorian 日历(缺省)。
vbCalHijri	1	使用 Hijri 日历。

说明

可以程序化地只设置 Calendar 属性。例如，要使用 Hijri 日历，使用：

Calendar=vbCalHijri

Call 语句

将控制权转移到一个 Sub 过程，Function 过程，或动态链接库(DLL)过程。

语法

[Call]*name*[*argumentlist*]

Call 语句的语法具有以下几个部分：

部分	描述
call	可选参数；关键字。如果指定了这个关键字，则 argumentlist 必须加上括号，例如： CallMyProc(0)
name	必要参数，要调用的过程名称
argumentlist	可选参数，传递给过程的变量、数组或表达式的列表，各参数间以逗号隔开。Argumentlist 的每个参数都可以加上关键字 ByVal 或 ByRef，来描述被调用的过程将如何处理这些参数。不过，ByVal 和 ByRef 只能用于调用 DLL 过程的 Call 语句。在 Macintosh 中，在调用一个 Macintosh 代码资源时，ByVal 和 ByRef 可以与 Call 一同使用。

说明

调用一个过程时，并不一定要使用 Call 关键字。如果使用 Call 关键字来调用一个需要参数的过程，argumentlist 就必须要加上括号。如果省略了 Call 关键字，那么也必须要省略 argumentlis 外面的括号。如果使用 Call 语法来调用内建函数或用户定义函数，则函数的返回值将被丢弃。

若要将整个数组传给一个过程，使用数组名，然后在数组名后加上空括号。

示例

下列示例示范如何使用 Call 语句来将控制转移到子过程、内在函数和动态链接库（DLL）过程，或是 Macintosh 代码资源内的过程。

```
'调用一个子过程。
CallPrintToDebugWindow("HelloWorld")
'上面的语句将控制转移到下面的子过程。
SubPrintToDebugWindow(AnyString)
    Debug.PrintAnyString'在“立即”窗口中显示。
EndSub

'调用一个内在函数，函数的返回值被忽略不处理。
CallShell(AppName,1)    'AppName 包含可执行文件的路径。

'调用 MicrosoftWindowsDLL 过程。该声明语句必需是类模块中的私有的，而不是标准模块中的。
```

```
PrivateDeclareSubMessageBeepLib"User"(ByValNAsInteger)
SubCallMyDll()
    CallMessageBeep(0)    '调用WindowsDLL过程。
    MessageBeep0'再次调用，但不用“调用”这个关键字。
EndSub
```

CallByName 函数

执行一个对象的方法，或者设置或返回一个对象的属性。

语法

CallByName(*object*,*procedurename*,*calltype*,[*arguments*()])

CallByName 函数的语法有以下部分：

部分	描述
object	必需的；变体型（对象）。函数将要执行的对象的名称
Procedurename	必需的；变体型（字符串）。一个包含该对象的属性名称或者方法名称的字符串表达式
calltype	必需的；常数。一个 vbCallType 类型的常数，代表正在被调用的过程的类型
arguments()	可选的：变体型（数组）

说明

CallByName **函数用于获取或者设置一个属性，或者在运行时使用**

一个字符串名称来调用一个方法。

在下面的例子中，第一行使用 CallByName 来设置一个文本框的 MousePointer 属性，第二行得到 MousePointer 属性的值，第三行调用 Move 方法来移动文本框：

```
CallByNameText1,"MousePointer",vbLet,vbCrosshair
Result=CallByName(Text1,"MousePointer",vbGet)
CallByNameText1,"Move",vbMethod,100,100
```

Cancel 属性

返回或设置一个值，用来指示窗体中命令按钮是否为取消按钮。该命令按钮可以是 CommandButton 控件或 OLE 容器控件中的任何可作用命令按钮的对象。

应用于

Extender 对象,CommandButton 控件

语法

object.**Cancel** [=*boolean*]

Cancel 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式指定对象是否为取消按钮，“设置值”中有详细说明

设置值

boolean 的设置值如下：

设置值	描述
True	CommandButton 控件是取消按钮。
False	（缺省值）CommandButton 控件不是取消按钮。

说明

使用 Cancel 属性使得用户可以取消未提交的改变，并把窗体恢复到先前状态。

窗体中只能有一个 CommandButton 控件为取消按钮。当一个 CommandButton 控件的 Cancel 属性被设置为 True，窗体中其它 CommandButton 控件的 Cancel 属性自动地被设置为 False。当一个 CommandButton 控件的 Cancel 属性设置为 True 而且该窗体是活动窗体时，用户可以通过单击它，按 ESC 键，或者在该按钮获得焦点时按 ENTER 键来选择它。

对于 OLE 容器控件，只有那些作用象命令按钮的对象才有 Cancel 属性。

提示如果窗体支持不可恢复操作，如删除操作，一个好主意是将取消按钮设置为缺省按钮。为此，将 Cancel 属性和 Default 属性都设为 True。

请参阅

KeyDown,KeyUpEvents,KeyPressEvent,Default 属性

CancelAsyncRead 方法

取消异步数据请求。

应用于

UserControl 对象,UserDocument 对象

语法

object.**CancelAsyncRead**[*PropertyName*]

CancelAsyncRead 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
PropertyNam e	可选择的字符串表达式，它指定要取消的异步数据请求的名称

说明

该方法仅取消由 PropertyName 指定的异步数据读取请求；所有其它的请求仍将继续正常进行。

PropertyName 中的数值指定要取消的特定异步数据读取请求，并应与前面的 AsyncRead 方法调用中给出的数值匹配。若未给定 PropertyName，则最后一个没有给定 PropertyName 的 AsyncRead 方法调用被取消。

请参阅

AsyncRead 方法

CancelError 属性

返回或设置一个值，该值指示当选取“取消”按钮时是否出错。

应用于

CommonDialog 控件

语法

object.**CancelError**[=*boolean*]

CancelError 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
boolean	布尔表达式，如“设置值”中所描述，用以指示是否出错

设置值

用于 boolean 的设置有：

设置值	描述
True	有错。
False	（缺省值）没错。

说明

当该属性设置为 True 时，无论何时选取“取消”按钮，均产生 32755(cdlCancel)号错误。

数据类型

Boolean

请参阅

Cancel 属性

CanGetFocus 属性

返回或设置一个决定控件是否能够获得焦点的数值。在控件创建时，CanGetFocus 属性可读可写，在控件运行时，该属性是只读的。

应用于

UserControl 对象

设置值

CanGetFocus 的设置值为：

设置值	描述
True	（缺省）控件可以接收焦点。如果控件包含子控件，那么只有当其所有的子控件都不能接收焦点时，该控件本身才能接收焦点。可由控件的创建者来决定：是否要编写这样的代码，在控件接收到焦点时在控件上画出一个代表焦点的矩形。
False	控件不能接收焦点。

说明

只要控件至少包含一个设置为能够接收焦点的子控件，CanGetFocus 属性就不能设置为 False。如果 CanGetFocus 设置为 False，则其所有的子控件都不能设置为接收焦点。

CanGrow 属性

返回或设置一个值，决定在返回的文本超出控件预置尺寸时，控件是否能垂直地增长。

应用于

Function 控件(DataReportDesigner),Lanbel 控件(DataReportDesigner),TextBox 控件(DataReportDesigner)

语法

object.CanGrow[=*boolean*]

CanGrow 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
boolean	可选的。一个布尔表达式，指定控件是否能增长，如在设置值中所示

设置值

对 boolean 的设置如下：

设置值	描述
True	控件可以增长。
False	（缺省的）控件不能增长。

CanPaste 属性

该属性返回布尔值指示剪贴板是否包含粘贴到窗体的适当的信息（即控件）。

应用于

VBForm 对象

语法

object.**CanPaste**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

CanPropertyChange 方法

询问容器是否能够变更某个与数据源绑定的属性值。当由 *PropertyName* 指定的属性与数据源绑定时，CanPropertyChange 方法十分有用。

应用于

UserControl 对象

语法

object.**CanPropertyChange***PropertyName***

CanPropertyChange 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
PropertyName	字符串表达式，它表示控件请求变更的属性名

返回值

CanPropertyChange 可能的返回值为：

设置值	描述
True	此时可以变更 PropertyName 中指定的属性。
False	此时不能变更 PropertyName 中指定的属性；容器以只读方式打开绑定的数据表。这时请不要设置属性值；否则将在某些控件容器中导致错误。

说明

在变更一个能够数据绑定的属性值之前，控件必须调用 CanPropertyChange。

注意目前，在 VisualBasic 中，CanPropertyChange 总是返回 True，即使在数据源中绑定字段是只读的。当控件试图变更某个只读字段时，VisualBasic 并不产生错误；它只是不更新数据源。

作为示例，下面的代码说明了如何使用 CanPropertyChange 方法：

```
PublicPropertyLetAddress(ByValcValueAsString)
   IfCanPropertyChange("Address")Then
       m_Address=cValue
       PropertyChanged"Address"
   EndIf
EndProperty
```

Caption 属性

窗体——确定显示在 Form 或 MDIForm 对象的标题栏中的文本。当窗体为最小化时，该文本被显示在窗体图标的下面。

控件——确定显示在控件中还是附在控件之后的文本。

MenuLine 对象——确定为 Menu 控件还是为 MenuItems 集合中的对象显示的文本。

对于 Menu 控件，Caption 在运行时通常是可读/写的。但是对于被 VisualBasic 的加载宏遗弃或提供的菜单项来说，Caption 是只读的，例如 MenuLine 对象。

应用于

Label 控件(DataReportDesigner),PropertyPage 对象,CheckBox 控件,CommandButton 控件,Form 对象,Forms 集合,Frame 控件,Label 控件,MDIForm 对象,Menu 控件,OptionButton 控件,ADOData 属性,RemoteData 控件,Data 控件

语法

object.**Caption**[=*string*]

Caption 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象。如果 object 被省略，那么与活动窗体模块相联系的窗体被认为是 object
string	字符串表达式，其值是被显示为标题的文本

说明

当创建一个新的对象时，其缺省标题为缺省的 Name 属性设置。该缺省标题包括对象名和一个整数，如 Command1 或 Form1。为了获得一个描述更清楚的标签，应对 Caption 属性进行设置。

可以使用 Caption 属性赋予控件一个访问键。在标题中，在想要指定为访问键的字符前加一个(&)符号。该字符就带有一个下划线。同时按下 ALT 键和带下划线的字符就可把焦点移动到那个控件上。为了在标题中加入一个(&)符号而不是创建访问键，需要在标题中加入两个即(&&)符号。这样，在标题中只有单个(&)符号被显示而没有带下划线的字符。

Label 控件标题的大小没有限制。对于窗体和所有别的有标题的控件，标题大小的限制是 255 个字符。

要为窗体显示标题，可以将 BorderStyle 属性设为定长单线（1 或 vbFixedSingle）、复长（2 或 vbSizable）、或定长对话框(3 或 vbFixedDialog)。超出窗体标题栏的标题将被裁切。当一个 MDI 子窗体在一个 MDIForm 对象中被最大化时，子窗体的标题将被包括在父窗体的标题中。

提示对于标签来说，将 AutoSize 属性设为 True 自动调整控件的大小以与其标题相适合。

请参阅

BorderStyle 属性,AutoSize 属性,Name 属性,BorderStyle 属性 (ActiveX 控件)

示例

当用户每次单击按钮时，该例子将改变 CommandButton 控件的 Caption 属性。要试用此例，先将下面的代码粘贴到一个包含名为 Command1 的 CommandButton 的窗体的声明部分，然后按下 F5 键并单击按钮。

```
PrivateSubCommand1_Click()
    '检查标题，然后改变它。
    IfCommand1.Caption="Clicked"Then
        Command1.Caption="OK"
    Else
        Command1.Caption="Clicked"
    EndIf
EndSub
```

Caption 属性

返回一个包含活动窗口标题的字符串。此属性为只读。

应用于

Window 对象

说明

活动窗口标题是指显示在窗口标题栏的文本。

请参阅

SetFocus 方法

示例

本例用 Caption 属性显示活动窗口的标题：

```
Debug.PrintApplication.VBE.ActiveWindow.Caption
```

Category 属性

返回与某个 Member **对象相关联的 Category 属性，或者对其进行设置。**

应用于

Member 对象

语法

object.**Category**

object **所在处代表对象表达式，其值是“应用于”列表中的对象。**

CausesValidation 属性

返回或设置一个值，该值确定正在获得焦点的第二个控件上 Validate 事件是否将发生。

语法

object.**CausesValidation**[=*boolean*]

CausesValidation 属性的语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
boolean	一个布尔表达式，确定获得焦点的控件是否引发 Validate 事件

设置值

boolean 的设置值如下：

设置值	描述
True	(缺省)已获得焦点的控件引发它的 Validate 事件。
False	已获得焦点的控件不引发它的 Validate 事件。

说明

当一个控件会失去焦点时，CausesValidation 属性和 Validate 事件联合进行限定。

请参阅

ValidateEvent；“MicrosoftVisualBasic6.0 程序员指南第七章

示例

该示例使用三个控件来示范 Validate 事件和 CausesValidation 属性的使用。在缺省情况下，两个 TextBox 控件的 CausesValidation 属性设置为 True，这样当您想把焦点从一个 TextBox 转换到另一个时，Validate 事件发生。如果 Text1 没有包含日期或 Text2 没有包含一个大于 10 的数字，焦点的转换将被阻止。由于 Command1 控件的 CausesValidation 属性设置为 False，因此您无论何时都可以

单击 Help 按钮。

要试验该示例，在窗体中放置一个 CommandButton 和两个 TextBox 控件，将代码粘接到窗体的“声明”部分并运行此工程。按 Tab 键尝试转换焦点。

```
PrivateSubForm_Load()
    '设置按钮的 CausesValidation 属性为 False。当用户
    '单击按钮时，Validate 事件不发生。
    '设置按钮的 Caption 属性为“帮助”。
    WithCommand1
        .CausesValidation=False
        .Caption="Help"
    EndWith

    Show
    WithText1'选择 Text1 的文本并为它设置焦点。
        .SelLength=Len(Text1.Text)
        .SetFocus
    EndWith
EndSub

PrivateSubCommand1_Click()
    '当单击此按钮时给出用户帮助信息。
```

```
    MsgBox_
    "Text1mustbesettoadate."&VbCrLF&_
    "Text2mustbeanumberlessthan10."
EndSub

PrivateSubText1_Validate(KeepFocusAsBoolean)
    '如果值不是一个日期，则保持焦点，除非用户
    '单击 Help。
    IfNotIsDate(Text1.Text)Then
        KeepFocus=True
        MsgBox"Pleaseinsertadateinthisfield.",, "Text1"
    Endif
EndSub

PrivateSubText2_Validate(KeepFocusAsBoolean)
    '如果值是一个大于 10 的数字，保持焦点。
    IfNotIsNumeric(Text2.Text)OrVal(Text2.Text)>10Then
        KeepFocus=True
MsgBox_
"Pleaseinsertanumberlessthanorequalto10.",,"Text2"
    EndIf
EndSub
```

Change 事件

指示一个控件的内容已经改变的。此事件如何和何时发生则随控件的不同而不同：

ComboBox—改变控件的文本框部分的正文。该事件仅在 Style 属性设置为 0（下拉 Combo）或 1（简单 Combo）和正文被改变或者通过代码改变了 Text 属性的设置时才会发生。

DirListBox—改变所选择的目录。该事件在双击一个新的目录或通过代码改变 Path 属性的设置时发生。

DriveListBox—改变所选择的驱动器。该事件当选择一个新的驱动器或通过代码改变 Drive 属性的设置时发生。

HScrollBar 和 VScrollBar（水平和垂直滚动条）—移动滚动条的滚动框部分。该事件在进行滚动或通过代码改变 Value 属性的设置时发生。

Label—改变 Label 的内容。该事件在一个 DDE 链接更新数据或通过代码改变 Caption 属性的设置时发生。

PictureBox—改变 PictureBox 的内容。该事件当一个 DDE 链接更新数据或通过代码改变 Picture 属性的设置时发生。

TextBox—改变文本框的内容。该事件当一个 DDE 链接更新数据、用户改变正文或通过代码改变 Text 属性的设置时发生。

应用于

ComboBox 控件,DirListBox 控件,DriveListBox 控件,HscrolL

Bar,VscrollBar 控件,Label 控件,PictureBox 控件,TextBox 控件,DBCombo 控件

语法

PrivateSub*object*_**Change**(**[***index***AsInteger]**)

Change 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中控件

说明

Change 事件过程可协调在各控件间显示的数据或使它们同步。例如，可用一个滚动条的 Change 事件过程更新一个 TextBox 控件中滚动条的 Value 属性的设置。或者可以利用 Change 事件过程在一个工作区里显示数据和公式，在另一个区域里显示结果。

Change 事件过程在更新系统控件（DirListBox，DriveListBox 和 FileListBox）中的各属性时也是有用的。例如，可更新一个 DirListBox 控件的 Path 属性设置来反映一个 DriveListBox 控件的 Drive 属性设置的改变。

注意一个 Change 事件过程有时会导致一个层叠事件。这种情况在控件的 Change 事件过程改变该控件的内容时会发生，例如，通过用代码设置一个决定该控件的值的属性，如对一个 TextBox 控件的 Text 属性之类的设置。为了避免层叠事件：

如果可能，应避免为控件编写能改变该控件内容的 Change 事件过程。如果编写了那样的过程，应确保设置一个标志用来防止在当前变化进行中更进一步的变化。

避免创建两个或两个以上其 Change 事件过程互相影响的控件。例如，两个 TextBox 控件在它们的 Change 事件期间互相更新。

避免对 HScrollBar 和 VScrollBar 控件在 Change 事件中使用 MsgBox 函数或语句。

请参阅

KeyDown,KeyUpEvents,KeyPressEvent,LostFocusEvent,PathChangeEvent,PatternChangeEvent,Picture 属性,tExt 属性,Value 属性,Drive 属性,LinkTopic 属性,Style 属性,Caption 属性,Path 属性,AllowCustomize 属性,Customize 方法,RestoreToolbar 方法,SaveToolBar 方法,Picture 属性(ActiveX 控件),Text 属性(ActiveXOntrols),Value 属性(ActiveX 控件),Caption 属性(ActiveX 控件)

示例

本例在 TextBox 控件中显示水平滚动条的 Value 属性的数值。要尝试这个例子，需创建一个带有 TextBox 控件及 HScrollBar 控件的窗体，然后将码粘贴到一个带有水平滚动条(HScrollBar 控件)和 TextBox 控件的窗体的声明部分。按 F5 键并单击水平滚动条。

```
PrivateSubForm_Load()
   HScroll1.Min=0                    '设置最小值。
```

```
    HScroll1.Max=1000                    '设置最大值。
    HScroll1.LargeChange=100 '设置 LargeChange.
    HScroll1.SmallChange=1       '设置 SmallChange.
EndSub

PrivateSubHScroll1_Change()
    Text1.Text=HScroll1.Value
EndSub
```

Changed 属性

返回或设置一个数值，它指示某个属性页上的属性值已经变更。在属性页创建时，Changed 属性不可用，在属性页运行时，该属性可读可写。

应用于

PropertyPage 对象

语法

object.**Changed**[=*boolean*]

Changed 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
boolean	布尔值，它判断属性页上的某个属性是否已经变更，并使属性页被重写

设置值

boolean 的设置值为：

设置值	描述
True	属性页已经被重写，因为页上的某个属性已被变更。
False	属性页未被重写，页上的属性都未变更。

说明

当变更属性页上的属性值时，这些变更不会立即生效；相反，只有当按下“应用”按钮、“确定”按钮，或因为选择选项卡改变属性页时，这些变更才能生效。这样做可以有机会更正对属性页所做的变更。

例如，当变更某个属性页上的属性值时，应将 Changed 属性设置为 True。把 Changed 属性设置为 True 将通知属性页使“应用”按钮可用。

Charset 属性

设置或者返回字体中所用字符集。

应用于

Font 对象

语法

object.**Charset**[=*value*]

Charset 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	整数值，它指定了字体所用的字符集，如下列“设置值””中所述

设置值

下面是 value 的一些常用的设置值：

值	描述
0	标准 Windows 字符
2	符号字符集。
128	双字节字符集(DBCS)，唯一适用于 Windows 的日语版本。
255	扩展字符，一般由 DOS 应用程序显示。

说明

将 Charset 属性设置为有效值之后，就选定了仅在当前字体中可用的字符集。

ChDir 语句

改变当前的目录或文件夹。

语法

ChDir*path*

必要的 *path* 参数是一个字符串表达式，它指明哪个目录或文件夹将成为新的缺省目录或文件夹。*path* 可能会包含驱动器。如果没有指定驱动器，则 ChDir 在当前的驱动器上改变缺省目录或文

件夹。

说明

ChDir 语句改变缺省目录位置，但不会改变缺省驱动器位置。例如，如果缺省的驱动器是 C，则下面的语句将会改变驱动器 D 上的缺省目录，但是 C 仍然是缺省的驱动器：

```
ChDir"D:\TMP"
```

在 PowerMacintosh 中，默认驱动器总是改为在 *path* 语句中指定的驱动器。完整路径指定由卷标名开始，相对路径由冒号(:)开始.ChDir 可以辨认路径中指定的别名：

```
ChDir"MacDrive:Tmp" '在 Macintosh 中。
```

注意当改变相对路径时，在 MicrosoftWindows 和 Macintosh 中使用不同符号：

```
ChDir".."'在 MicrosoftWindows 中，上移一层目录.
ChDir"::"'在 Macintosh 中，上移一层目录。
```

请参阅

ChDrive 语句,CurDir 函数,DirFunctioln,MkDir 语句,RmDir 语句

示例

本示例使用 ChDir 语句来改变当前目录或文件夹在 Macintosh 中，默认驱动器名称是 HD″并且路径部分由冒号分隔代替反斜线。同样，可以指定 Macintosh 的文件夹来替代\Windows。最后，在 Macintosh 中，通配符不再包含特殊意义，只作为普通字符。。

```
'将当前目录或文件夹改为“MYDIR”。
ChDir"MYDIR"
'假设当前的驱动器是“C:”，下列语句将把
'缺省目录改至“D:”，而“C:”仍旧是当前驱动器。
ChDir"D:\WINDOWS\SYSTEM"
```

ChDrive 语句

改变当前的驱动器。

语法

ChDrive*drive*

必需的 *drive* 参数是一个字符串表达式，它指定一个存在的驱动器。如果使用零长度的字符串(""),则当前的驱动器将不会改变。如果 drive 参数中有多个字符，则 ChDrive 只会使用首字母。在 Macintosh 中，ChDrive 将当前文件夹改变到指定驱动器的根文件夹。

请参阅

ChDir 语句,CurDir 函数,MkDir 语句,RmDir 语句

示例

本示例使用 ChDrive 语句来改变当前的驱动器。在 Macintosh 中，“HD:”是默认驱动器名称，而 ChDrive 将改变当前文件夹到指定驱动器的根文件夹。以下示例假设机器存在 D 驱动器。

ChDrive"D"　'使“D”成为当前驱动器。

CheckBox 控件

选择 CheckBox 控件后，该控件将显示 X，而清除 CheckBox 控件后，X 消失。该控件可用来提供 True/False 或者 Yes/No 选项。组中可以使用 CheckBox 控件显示多项选择，从而可选择其中的一项或多项。也可以通过对 Value 属性编程设置 CheckBox 的值。

语法

CheckBox

说明

CheckBox 和 OptionButton 控件功能相似，但二者之间也存在着重要差别：在一个窗体中可以同时选择任意数量的 CheckBox 控件。而反过来，在一个组中，在任何时候则只能选择一个 OptionButton 控件。

为了在 CheckBox 后面显示文本，需要设置 Caption 属性。Value 属性用来确定控件的状态—选择、清除、或不可用。

属性

DataMember 属性,DataFormat 属性,RightToLeft 属性,OLEDrop Mode 属性,BackColor,ForeColor 属性,FontBold,FontItalic, FontStrikethru,FontUnderline 属性,FontName 属性,FontSize 属性,Height,Width 属性,Left,Top 属性,Picture 属性,TabIndex 属性,Tag 属性,Value 属性,Visible 属性,Alignment 属

性,DragIcon 属性,DragMode 属性,hWnd 属性,MouseIcon 属性,MousePointer 属性,Style 属性,TabStop 属性,Appearance 属性,Caption 属性,Enabled 属性,HelpContextID 属性,Index 属性(ControlArray),Name 属性,parent 属性,Font 属性,Container 属性,ToolTipText 属性,DisabledPicture 属性,DownPicture 属性,MaskColor 属性,useMaskColor 属性,whatsThisHelpIS 属性,DataChanged 属性,DataField 属性,CheckBox

方法

Refresh 方法,SetFocus 方法,Drag 方法,Move 方法,Zorder 方法,OLEDrag 方法,ShowwhatsThis 方法

事件

Click 事件,DragDrop 事件,DragOver 事件,GotFocus 事件,KeyDown,keyUp 事件,KeyPress 事件,LostFocus 事件,MouseDown,MouseUp 事件,MouseMove 事件,Valicdate 事件,OLECompleteDrag 事件,OLEDragDrop 事件,OLEDragOver 事件,OLEGiveFeedback 事件,OLESetData 事件,OLEStartDrag 事件

请参阅

Value 属性,Caption 属性,Caption 属性(ActiveX 控件)

Checked 属性

返回或设置一个值，该值用来确定是否在一个菜单项后显示复选标记。

应用于

Menu 控件

语法

object.**Checked**[=*boolean*]

Checked 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定是否在一个菜单项后显示复选标记的布尔表达式

设置

boolean 的设置为：

设置	描述
True	在菜单项之后放置复选标记。
False	（缺省）不在紧接菜单项之后放置一个检查标记。

说明

在设计时可以使用“菜单编辑器”将 Checked 设置为 True。在运行时，作为 Menu 控件所附加的单击事件过程的一部分，能够将 Checked 在开和关的状态之间转换。同样可以在启动过程中或在窗体的装入事件过程中设置 Checked 的值。

对于一个 Menu 控件，Checked 在运行时通常是可读/写的。但是，对于那些被 VisualBasic 的加载宏遗弃或提供的菜单项来说，Checked 是只读的，例如在“外接程序”菜单中的“外接程序管

理器”命令。

示例

该例子显示并删除紧跟一个菜单项的检查标记。要试用此例，先创建一个带有 Menu 控件的窗体，该 Menu 控件有一个菜单项（将 Caption 和 Name 属性都设为 MyMenuItem），然后按下 F5 键并选择该菜单项。

```
PrivateSubMyMenuItem_Click()
   '将菜单项上的检查标记打开或关闭。
   MyMenuItem.Checked=NotMyMenuItem.Checked
EndSub
```

Choose 函数

从参数列表中选择并返回一个值。

语法

Choose(*index*,*choice-1*[,*choice-2*,...[,*choice-n*]])

Choose 函数的语法具有以下几个部分：

部分	描述
index	必要参数，数值表达式或字段，它的运算结果是一个数值，且界于 1 和可选择的项目数之间
choice	必要参数，Variant 表达式，包含可选择项目的其中之一

说明

Choose 会根据index 的值来返回选择项列表中的某个值。如果index 是 1，则 Choose 会返回列表中的第 1 个选择项。如果 index 是 2，则会返回列表中的第 2 个选择项，以此类推。

可以使用 Choose 来查阅一个列表中的项目。例如，如果 index 所指定的值为 3，而 choice-1="one"、choice-2="two"、且 choice-3="three"，那么 Choose 将返回"three"。当 index 代表一选项组中的值时，则这项功能将会特别有用。

即使它只返回一个选项值，Choose 仍然会计算列表中的每个选择项。所以应该注意到这项副作用。例如，当在每个选择项表达式中使用了 MsgBox 函数作为其中的一部分时，每计算一个选择项，就会显示一次消息框。

当 index 小于 1 或大于列出的选择项数目时，Choose 函数返回 Null。

如果 index 不是整数，则会先四舍五入为与其最接近的整数。

请参阅

Iif 函数,SelectCase 语句,Switch 函数

示例

本示例使用 Choose 函数来显示一个名称，该名称对应於用 Ind 参数传递到过程之中的索引。

```
FunctionGetChoice(IndAsInteger)
   GetChoice=Choose(Ind,"Speedy","United","Federal")
EndFunction
```

Chr 函数

返回 String，其中包含有与指定的字符代码相关的字符。

语法

Chr(*charcode*)

必要的 *charcode* 参数是一个用来识别某字符的 Long。

说明

0 到 31 之间的数字与标准的非打印 ASCII 代码相同。例如，Chr(10) 可以返回换行字符。charcode 的正常范围为 0–255。然而，在 DBCS 系统，charcode 的实际范围为-32768 到 65535。

注意 ChrB 函数作用于包含在 String 中的字节数据。ChrB 总是返回一个单字节，而不是返回一个字符，一个字符可能是一个或两个字节。ChrW 函数返回包含 Unicode 的 String，若在不支持 Unicode 的平台上，则其功能与 Chr 函数相同。

请参阅

Asc 函数，输入转换函数，字符集（0-127），Str 函数，字符集（128-255）

示例

本示例使用 Chr 函数来返回指定字符码所代表的字符。

```
DimMyChar
MyChar=Chr(65) '返回 A。
MyChar=Chr(97) '返回 a。
MyChar=Chr(62) '返回>。
```

```
MyChar=Chr(37) '返回%。
```

Circle 方法

在对象上画圆、椭圆或弧。

应用于

Propettypage 对象,UserControl 对象,UserDocument 对象,Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**Circle**[**Step**](*x*,*y*),*radius*,[*color*,*start*,*end*,*aspect*]

Circle 方法的语法有如下的对象限定符和部分：

部分	描述
object	可选的。对象表达式，其值为“应用于”列表中的对象。如果 object 省略，具有焦点的窗体作为 object
step	可选的。关键字，指定圆、椭圆或弧的中心，它们相对于当前 object 的 CurrentX 和 CurrentY 属性提供的坐标
(*x*,*y*)	必需的。Single（单精度浮点数），圆、椭圆或弧的中心坐标。object 的 ScaleMode 属性决定了使用的度量单位
radius	必需的。Single（单精度浮点数），圆、椭圆或弧的半径。object 的 ScaleMode 属性决定了使用的度量单位
color	可选的。Long（长整型数），圆的轮廓的 RGB 颜色。如果它被省略，则使用 ForeColor 属性值。可用 RGB 函数或 QBColor 函数指定颜色
start,*end*	可选的。Single（单精度浮点数），当弧、或部分圆或椭圆画完以后，start 和 end 指定（以弧度为单位）弧的起点和终点位置。其范围从-2pi 到 2pi。起点的缺省值是 0; 终点的缺省值是 2*pi
aspect	可选的。Single（单精度浮点数），圆的纵横尺寸比。缺省值为 1.0，它在如何屏幕上都产生一个标准圆（非椭圆）

说明

想要填充圆，使用圆或椭圆所属对象的 FillColor 和 FillStyle 属性。只有封闭的图形才能填充。封闭图形包括圆、椭圆、或扇形。

画部分圆或椭圆时，如果 start 为负，Circle 画一半径到 start，并将角度处理为正的；如果 end 为负，Circle 画一半径到 end，并将角度处理为正的。Circle 方法总是逆时针（正）方向绘图。

画圆、椭圆或弧时线段的粗细取决于 DrawWidth 属性值。在背景上画圆的方法取决于 DrawMode 和 DrawStyle 属性值。

画角度为 0 的扇形时，要画出一条半径（向右画一水平线段），这时给 start 规定一很小的负值，不要给 0。

可以省略语法中间的某个参数，但不能省略分隔参数的逗号。您指定的最后一个参数后面的逗号是可以省略的。

Circle 执行时，CurrentX 和 CurrentY 属性被参数设置为中心点。

这个方法不能用在 WithAndWith 语句块中。

请参阅

Line 方法

示例

这个示例用 Circle 方法在窗体中央画许多同心圆。要运行这个示例，将此代码放入窗体的 General 部分。按 F5 并单击窗体。

```
SubForm_Click()
   DimCX,CY,Radius,Limit    'Declarevariable.
   ScaleMode=3 '以像素为单位。
   CX=ScaleWidth/2 'X 位置。
   CY=ScaleHeight/2'Y 位置。
   IfCX>CYThenLimit=CYElseLimit=CX
```

```
    ForRadius=0ToLimit  '半径。
        Circle(CX,CY),Radius,RGB(Rnd*255,Rnd*255,Rnd*255)
    NextRadius
EndSub
```

类

类是一个模板，对象是由它而创建的。类模块中的代码描述了从该类创建的对象的特性(attribute)和行为。
虽然类不是对象，但是它的确有定义其特性(attribute)的设计时属性(property)和定义其行为的事件。

Class 属性

返回或设置内嵌对象的类名。

应用于

OLE 容器控件

语法

object.**Class**[=*string*]

Class 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	一个指定类名的字符串表达式

说明

类名定义对象的类型。支持 ActiveX 部件的应用程序，使用下列语法之一可完全限定对象的类名：

application.objecttype.version

objecttype.version

ActiveX 部件类名的语法包含下面部分：

部分	描述
application	支持对象的应用程序的名称
objecttype	对象库中定义的对象名
version	支持对象的应用程序或对象的版本号

例如，MicrosoftExcelversion5.0 支持一些对象，包括工作表和图表。它们的类名是 Excel.Sheet.5 和 Excel.Chart.5。Microsoft WordArtversion2.0 支持单个对象，其类名为 MSWordArt.2。

注意一些 ActiveX 部件的编程文件引用类名的语法为程序 ID。

为了查看系统中可用的类名列表，先选择 OLE 容器控件，再选择“属性”窗口中的 Class 属性，然后单击“生成器”按钮。

从系统剪贴板复制对象会更新控件的 Class 属性。例如，如果从系统剪贴板将 MicrosoftExcel 图表粘贴到 OLE 容器控件中，而该 OLE 容器控件在此之前含有一个 MicrosoftExcel 工作表，那么它的 Class 属性设置就从 Excel.Sheet.5 改为 Excel.Chart.5。在运行时，可以使用 Paste 方法或 PasteSpecialDlg 方法，从系统剪贴板粘贴对象。

请参阅

PasteSpecialDlg 方法,Paste 方法,PasteOKPropery

ClassName 属性

返回一个 VBControl 对象。

应用于

VBControl 对象

语法

object.**ClassName**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Clear 方法

清除 Err 对象的所有属性设置。

应用于

ExportFormats 集合,DataMembers 集合,BindingCollection 对象,Err 对象

语法

object.**Clear**

objec 总是 Err 对象。

说明

在处理错误之后使用 Clear 来清除 Err 对象，例如，在对

OnErrorResumeNext 使用拖延错误处理时就可使用 Clear。每当执行下列语句时就会自动调用 Clear 方法：

任意类型的 Resume 语句。

ExitSub，ExitFunction，ExitProperty

任何 OnError 语句。

注意当处理因访问其他对象产生的错误时，与其使用 OnErrorGoTo，不如使用 OnErrorResumeNext。每一次与对象打交道之后都检查 Err，则可消除代码访问对象时的含混之处。可以确认是哪个对象将错误引入 Err.Number 中，也可以确认最初是哪个对象产生了这个错误（Err.Source 中指定的对象）。

示例

本示例使用 Err 对象的 Clear 方法将 Err 对象之数值属性重新设置为零，并将其字符串属性设置为零长度字符串。如果在代码中省略 Clear 方法，则每完成一次循环便会显示一次错误信息（发生错误之后），不论程序中的计算结果是否真的有错误。

```
DimResult(10)AsInteger '声明数组变量
                '其元素容易溢出
Dimindx
OnErrorResumeNext  '将错误处理的方式改为“继续下一行”。
DoUntilindx=10
   '下面计算若有错误发生，便显示错误信息。
   Result(indx)=Rnd*indx*20000
```

```
    IfErr.Number<>0Then
        MsgBoxErr,,"ErrorGenerated:",Err.HelpFile,Err.HelpContext
        Err.Clear    '清除 Err 对象的属性。
    Else
        indx=indx+1
    EndIf
Loop
```

Clear 方法（Clipboard、ComboBox、ListBox）

用于清除 ListBox，ComboBox 或系统剪贴板的内容。

应用于

Clipboard 对象,ComboBox 控件,ListBox 控件

语法

object.**Clear**

object 所在处代表一个对象表达式，其值为“应用于”列表中的一个对象。

说明

绑定到 Data 控件的 ListBox 或 ComboBox 控件不支持 Clear 方法。

请参阅

AddItem 方法,Cls 方法,RemoveItem 方法

示例

本示例使用 Clear 方法清除一个图表框中的所有项目。要检验此

示例，可将本例代码粘贴到带有一个名为 List1 的 ListBox 控件的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
    DimEntry,I,Msg                          '声明变量。
    Msg="ChooseOKtoadd100itemstoyourlistbox."
    MsgBoxMsg                               '显示信息。
    ForI=1To100                             '计数值从 1 到 100。
        Entry="Entry"&I                     '创建输入项。
        List1.AddItemEntry                      '添加该输入项。
    NextI
    Msg="ChooseOKtoremoveeveryotherentry."
    MsgBoxMsg                               '显示信息。
    ForI=1To50                              '确定如何
        List1.RemoveItemI                       '每隔一项
    NextI                                   '删除。
    Msg="ChooseOKtoremoveallitemsfromthelistbox."
    MsgBoxMsg                               '显示信息。
    List1.Clear                                 '清除列表框。
EndSub
```

本示例使用 Clear 方法清除 Clipboard 对象。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体

```
PrivateSubForm_Click()
    ConstCF_BITMAP=2                    '定义位图各种格式。
    DimMsg                              '声明变量。
    OnErrorResumeNext                   '设置错误处理。
    Msg="ChooseOKtoloadabitmapontotheClipboard."
    MsgBoxMsg                           '显示信息。
    Clipboard.Clear                     '清除剪贴板。
    Clipboard.SetDataLoadPicture("PAPER.BMP")    '取得位图。
    IfErrThen
        Msg="Can'tfindthe.BMPfile."
        MsgBoxMsg                       '显示错误信息。
        ExitSub
    EndIf
    Msg="AbitmapisnowontheClipboard.ChooseOKtocopy"
    Msg=Msg&"thebitmapfromtheClipboardtotheform."
    MsgBoxMsg                           '显示信息。
    Picture=Clipboard.GetData()         '从剪贴板上复制。
    Msg="ChooseOKtoclearthepicture."
    MsgBoxMsg                           '显示信息。
    Picture=LoadPicture()               '清除图片。
EndSub
```

Clear 方法（DataObject 对象）

删除 DataObject 对象的内容。

应用于

DataObject 对象

语法

object.**Clear**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

此方法仅对部件拖放源有效。如果 Clear 是在部件放目标中被调用，则会产生错误。

大部分部件支持人工 OLE 拖放事件，也有一些支持自动 OLE 拖放事件。

Clear 方法（Format 对象）

删除一个集合中的所有对象。

应用于

StdDataFormats 集合

语法

object.**Clear**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

要仅从集合中删除一个对象，请使用 Remove 方法。

请参阅

StdDataFormat 对象

Click 事件

此事件是在一个对象上按下然后释放一个鼠标按钮时发生。它也会发生在一个控件的值改变时。

对一个 Form 对象来说，该事件是在单击一个空白区或一个无效控件时发生。对一个控件来说，这类事件的发生是当：

用鼠标的左键或右键单击一个控件。对 CheckBox，CommandButton，Listbox 或 OptionButton 控件来说，Click 事件仅当单击鼠标左键时发生。

通过按下箭头键或者单击鼠标按钮，对 ComboBox 或 ListBox 控件中的项目进行选择。

当 CommandButton，OptionButton 或 CheckBox 控件具有焦点时，按下 SPACEBAR 键。

当窗体带有其 Default 属性设置为 True 的 CommandButton 控件时，按下 ENTER 键。

当窗体带有一个 Cancel 按钮—其 Cancel 属性设置为 True 的 CommandButton 控件时，按下 ESC 键。

对控件按下一个访问键。例如，如果一个 CommandButton 控件的

标题是"&Go"，则按下 ALT+G 键可触发该事件。

也可在代码中触发 Click 事件，通过：

将一个 CommandButton 控件的 Value 属性设置为 True。

将一个 OptionButton 控件的 Value 属性设置为 True。

改变一个 CheckBox 控件的 Value 属性的设置。

应用于

PropertyPage 对象,UserControl 对象,UserDocument 对象,CheckBox 控件,ComboBox 控件,CommandButtonContro,DirList Box 控件,FileListBox 控件,Form 对象,Forms 集合,Frame 控件,Image 控件,Label 控件,ListBox 控件,MDIFrom 对象,Menu 控件,OptionButton 控件,PictureBox 控件,TextBox 控件,OLE 容器控件,RichTextBox 控件,DBList 控件,DBCombo 控件,Animation 控件,Toolbar 控件,TabStrip 控件,StatusBar 控件,Slider 控件,ProgressBar 控件,ListView 控件,TreeView 控件

语法

PrivateSubForm_Click()

PrivateSub*object*_**Click**([*indexAsInteger*])

Click 事件的语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件

说明

通常，将一个 Click 事件过程附加到一个 CommandButton 控件，Menu 对象或 PictureBox 控件上用来执行命令或类似命令的操作。对其它可应用的控件来说，使用这个事件来触发一个动作以响应控件中的变化。

可用一个控件的 Value 属性从代码中测试该控件的状态。单击一个控件除了产生 Click 事件以外还可产生 MouseDown 和 MouseUp 事件。这三种事件发生的顺序根据控件的不同而不同。例如，对 ListBox 和 CommandButton 控件来说，这些事件按下列顺序发生：MouseDown、Click、MouseUp。但对 FileListBox，Label 或 PictureBox 控件来说，这些事件按下列顺序发生：MouseDown、MouseUp 和 Click。当给这些相关的事件附加事件过程时，要确保它们的操作不互相冲突。如果在应用程序中事件发生的顺序是重要的，则应对控件进行测试以确定事件的顺序。

注意为区别鼠标的左、中、右按钮，应使用 MouseDown 和 MouseUp 事件。

如果在 Click 事件中有代码，则 DlbClick 事件将永远不会被触发，因为 Click 事件是两个事件中首先被触发的事件。其结果是鼠标单击被 Click 事件截断，从而使 DblClick 事件不会发生。

请参阅

MouseDown,MouseUp 事件,DblClick 事件,Cancel 属性,Value 属性,Default 属性,Value 属性(ActiveX 控件)

示例

在本例中，每单击一次 PictureBox 控件时，它都沿对角线方向在

窗体中移动。要尝试这个例子，先将代码粘贴到窗体的声明部分，该窗体的左下角处有一个 PictureBox 控件，然后按 F5 键并单击 PictureBox。

```
PrivateSubPicture1_Click()
   Picture1.MovePicture1.Left+750,Picture1.Top-550
EndSub
```

Click 事件（VBA 外接程序对象模型）

发生在对一个命令条控件的 OnAction 属性进行设置时。

语法

Sub*object*_**Click**(**ByVal***ctrl***AsObject,ByRef***handled***AsBoolean,ByRef***cancel*
*default***AsBoolean**)

Click 事件的语法具有以下几个命名参数：

部分	描述
ctrl	必需的；一个对象，用来指定 Click 事件的来源。
handled	必需的；一个布尔型数，若为 True，就由其它外接程序处理此事件；若为 False，则表示不处理命令条项目的动作。
canceldefault	必需的；一个 Boolean 型数，若为 True，就执行缺省的动作；若为 False，则不执行缺省的动作。

说明

Click 事件是 CommandBarEvent 事件对象的特定事件，可使用 WithEvent 事件关键字声明一个变量，用来接收 CommandBar 控件的 Click 事件。此变量必须设置为 Events 事件对象的 CommandBar 事件属性返回值，CommandBarEvents 属性会以 CommandBar 控件作为参数，当 CommandBar 被单击时(对于使用 WithEvents 关键字声明了变量的情况下)，则代码被执行。

请参阅

CommandBarEvents 属性

示例

下列示例示范如何利用 WithEvents 及 Set 来设置 Click 事件过程的代码。请注意，对象引用 ce 是以 Click 事件的形式用于菜单名“工具”中。

```
PrivateWithEventsceAsCommandBarEvents
SubTest()
DimcAsCommandBarControl
Setc=Application.VBE.CommandBars("Tools").Controls(1)
Setce=Application.VBE.Events.CommandBarEvents(c)
EndSub

PrivateSubce_Click(ByValCommandBarControlAsObject,HandledAsBool
ean,CancelDefaultAsBoolean)
'将事件代码写在此处
EndSub
```

ClipBehavior 属性

返回或设置一个值，定义 HitTest 事件在 WindowlessUserControl 对象上的剪切行为。

语法

object.**ClipBehavior**[=*number*]

ClipBehavior 属性的语法包含如下部分：

部分	描述
object	一个 UserControl 对象
number	一个整数，它指定剪切行为。如在设置值中描述

设置值

对 number 的设置如下：

常数	设置值	描述
none	0	图形方法的输出将出现在控件矩形框内的任意位置。
useRegion	1	（缺省的）图形方法的输出将被剪切到控件的 MaskRegion。

说明

可以使用 ClipBehavior 属性来决定在 UserControl 的哪一位置显示图形方法的输出。按照缺省规定，任何使用图形方法绘制的图只在控件的 MaskRegion 上是可见的。MaskRegion 由任何子控件加上任何由 MaskPicture 和 MaskColor 属性定义的掩码组成。任何

在 MaskRegion 外绘制的图都是不可见的。
通过设置 ClipBehavior 为 None，绘制的图在控件的 MaskRegion 和 TransparentRegion 中都是可见的。
当与 HitBehavior 属性结合起来使用时，这一属性有助于决定 HitTest 事件的 HitResult 参数。
注意如果 UserControl 对象的 Windowless 属性被设置为 False，或者 BackStyle 属性被设置为 Opaque，该属性将被忽略。
重点并非所有的控件容器都支持 Windowless 属性。仅当知道它将用在支持 Windowless 激活的容器中时，需要更改 HitBehavior 属性。

Clipboard 对象

提供对系统 Clipboard 的访问。

语法

Clipboard

说明

Clipboard 对象用于操作剪贴板上的文本和图形。它使用户能够复制、剪切和粘贴应用程序中的文本和图形。在复制任何信息到 Clipboard 对象中之前，应使用 Clear 方法清除 Clipboard 对象中的内容，例如 Clipboard.Clear。
注意所有 Windows 应用程序共享 Clipboard 对象，因此当切换到其它应用程序时，剪贴板内容会改变。

Clipboard 对象可包含多段数据，只要每段数据的格式不同。例如，可用 SetData 方法把位图以 vbCFDIB 格式放到 Clipboard 中，接着再用 SetText 方法以 vbCFText 格式将文本放到 Clipboard 中。然后用 GetText 方法检索文本或用 GetData 方法检索图形。当用代码或菜单命令把另一段数据放到 Clipboard 中时，原 Clipboard 中相同格式的数据会丢失。

方法

Clear 方法(Clipboard,ComboBox,ListBox),GetData 方法, GetFormat 方法,GetText 方法,SetData 方法,SetText 方法

Clipboard 属性

返回一个 Clipboard 对象，该对象提供了对系统 Clipboard 的访问途径。

应用于

Global 对象

语法

Clipboard

说明

Clipboard 对象是用来对 Clipboard 上的文本和图形进行操作的。使用该对象就可以让用户把文本或者图形复制、剪切并粘贴到应用程序中。在把任何材料复制到 Clipboard 对象中之前，应先执行 Clear 方法（比如 Clipboard.Clear）来清除该对象的内容。

注意，Clipboard 对象为所有 Windows 应用程序所共享，因此，当切换到另一个应用程序时，其内容可能会被更改。

Clipboard 对象可以包含许多数据片段，只要每个片段都有不同的格式。例如，可以用 SetData 方法将一个具有 vbCFDIB 格式的位图放到的 Clipboard 上，然后用 SetText 方法将具有 vbCFText 格式的文本放到 Clipboard 上。然后就可用 GetText 方法获取文本或用 GetData 方法获取图形了。不论是通过代码还是菜单命令，只要把具有同样格式的另一个数据集合放到 Clipboard 上去，Clipboard 上的数据就会丢失。

请参阅

Global 对象,Clipboard 对象

ClipControls 属性

返回或设置一个值，决定 Paint 事件中的图形方法是重绘整个对象，还是只绘刚刚露出的区域。它还决定 MicrosoftWindows 运行环境是否创建一个不包括该对象的非图形控件的剪裁区。在运行时为只读。

应用于

PropertyPage 对象,UserControl 对象,userDocument 对象,Form 对象,Forms 集合,Frame 控件,PictureBox 控件

语法

object.**ClipControls**

ClipControls 有下列组成部分:

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指定如何重绘对象，“设置值”中有详细描述

设置值

boolean **的设置值如下**:

设置值	描述
True	（缺省值）Paint 事件中的图形方法重绘整个对象。在绘制之前，在该窗体中非图形控件的周围创建剪裁区。
False	Paint 事件中的图形方法只绘制刚刚露出的区域。在绘制之前，不在该窗体非图形控件的周围创建剪裁区。ClipControls 设置为 False 时，加载和重绘复杂窗体比较快。

说明

剪裁是确定当显示窗体时，对诸如 Frame 或 PictureBox 控件的窗体或容器的哪一部分进行重绘。在内存中建立窗体和控件的大纲。Windows 运行环境利用这个大纲来重画某些部分，诸如背景色，而不会影响其它部分，例如 TextBox 控件的内容。因为剪裁区建立在内存中，所以将该属性设置为 False 可以减少绘制或重绘窗体所需的时间。

剪裁区包括大部分控件，但是不在 Image、Label、Line 或 Shape 控件的周围剪裁。

避免将 ClipControls 设置为 True 的固有控件，嵌套在 ClipControls 属性设置为 False 的控件内（例如：命令按钮在图片框内）。这种控件嵌套引起控件不能准确重绘。为解决这个问题，将容器控件和嵌套的控件的 ClipControls 属性设置为 True。

请参阅

PaintEvnet,AutoRedraw 属性

示例

这个例子说明 ClipControls 属性将如何影响窗体的重画功能。要试用此例，先把代码粘贴到窗体的声明部分，然后按下 F5 键。注意，每次调整窗体的大小，或者用另一个窗体或应用程序覆盖部分窗体，都会改变整个窗体的颜色。结束这个程序并设置 ClipControls 为 False，然后再一次运行该程序。注意，只有新近显露的窗体部分才被重画。

```
PrivateSubForm_Paint()
   '随机选择背景颜色.
   BackColor=&HFFFFFF*Rnd
EndSub
```

Close 方法（VBA 外接程序对象模型）

关闭并清除一个窗口。

应用于

Window 对象

语法

object.**Close**

object **为一个对象表达式，其值是“应用于”列表中的一个对象。**

说明

Close **方法对于以下的窗口类型有不同的处理方式：**

对于一个代码窗格的窗口，Close **方法可撤消此代码窗格。**

对于一个设计器的窗口，Close **方法可撤消包含的设计器。**

对于在“视图”菜单中一直可见的窗口，Close **方法可将其隐藏起来。**

请参阅

Add 方法（VBA 外接程序对象模型），Remove 方法（VBA 外接程序对象模型）

示例

下列示例使用 Close **方法来关闭** Windows **集合里的指定成员。**

```
Application.VBE.Windows(9).Close
```

Close 方法

关闭一个打开的 TextStream **文件。**

应用于

TextStream 对象

语法

object.**Close**

object 始终是一个 TextStream 对象的名字。

请参阅

Read 方法,ReadAll 方法,ReadLine 方法,Skip 方法,SkipLine 方法,Write 方法,WriteBlankLines 方法,WriteLine 方法

Close 方法（OLE 容器）

关闭对象，并与提供该对象的应用程序终止连接。

应用于

OLE 容器控件

语法

object.**Close**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

这个方法只应用于内嵌对象，并且它与关闭对象是等效的。它对链接对象不起作用。

Close 语句

关闭 Open 语句所打开的输入/输出(I/O)文件。

语法

Close [*filenumberlist*]

可选的 *filenumberlist* 参数为一个或多个文件号，其中 *filenumber*

为任何有效的文件号，语法如下：

[[#]*filenumber*][,[#]*filenumber*]...

说明

若省略 filenumberlist，则将关闭 Open 语句打开的所有活动文件。

当关闭 Output 或 Append 打开的文件时，将属于此文件的最终输出缓冲区写入操作系统缓冲区。所有与该文件相关联的缓冲区空间都被释放。

在执行 Close 语句时，文件与其文件号之间的关联将终结。

请参阅

End 语句,Stop 语句,Open 语句,Reset 语句

示例

本示例使用 Close 语句来关闭所有为 Output 而打开的三个文件。

```
Dim I,FileName
For I=1To3                          '循环三次。
   FileName="TEST"&I                '创建文件名。
   Open FileName For Output As #I   '打开文件。
   Print #I,"This is a test."       '将字符串写入文件。
Next I
Close  '将三个已打开的文件全部关闭。
```

Cls 方法

清除运行时 Form 或 PictureBox 所生成的图形和文本。

应用于

PropertyPage 对象,UserControl 对象,UserDocument 对象,Form 对象,Forms 集合,PictureBox 控件

语法

object.**Cls**

object 所在处代表一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 object，则带有焦点的 Form 就被认为是 object。

说明

Cls 将清除图形和打印语句在运行时所产生的文本和图形，而设计时在 Form 中使用 Picture 属性设置的背景位图和放置的控件不受 Cls 影响。如果激活 Cls 之前 AutoRedraw 属性设置为 False，调用时该属性设置为 True，则放置在 Form 或 PictureBox 中的图形和文本也不受影响。这就是说，通过对正在处理的对象的 AutoRedraw 属性进行操作，可以保持 Form 或 PictureBox 中的图形和文本。

调用 Cls 之后，object 的 CurrentX 和 CurrentY 属性复位为 0。

请参阅

Clear 方法 (Clipboard,ComboBox,ListBox),AutoRedraw 属性,CurrentX,CurrentY 属性

示例

本示例使用 Cls 方法从一个窗体中删除打印信息。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 并单击该窗体。

```
PrivateSubForm_Click()
    DimMsg                                  '声明变量。
    AutoRedraw=-1    '打开 AutoRedraw。
    ForeColor=QBColor(15)                   '将前景设置为白色。
    BackColor=QBColor(1)                    '将背景设置为蓝色。
    FillStyle=7                             '设置对角线菱形。
    Line(0,0)-(ScaleWidth,ScaleHeight),,B           '将框放在窗体上。
    Msg="Thisisinformationprintedontheformbackground."
    CurrentX=ScaleWidth/2-TextWidth(Msg)/2   '设置 X 的位置。
    CurrentY=2*TextHeight(Msg)                  '设置 Y 的位置。
    PrintMsg                                '打印信息至窗体。
    Msg="ChooseOKtocleartheinformationandbackground"
    Msg=Msg&"patternjustdisplayedontheform."
    MsgBoxMsg                               '显示信息。
    Cls                                     '清除窗体的背景。
EndSub
```

CodeLocation 属性

该属性返回代码模块中定义成员的行位置。

应用于

Member 对象

语法

object.**CodeLocation**[=*propkind*]

CodeLocation 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
propkind	Vbext_PropertyKind 的数值，如“设置值”中所描述

设置值

vbext_PropertyKind 的设置值是：

常数	值	描述
vbext_PropertyGet	1	（缺省值）返回属性的 Get 元素的代码位置。
vbext_PropertyLet	2	返回属性的 Let 元素的代码位置。
vbext_PropertySet	3	返回属性的 Set 元素的代码位置。

CodeModule 对象

在诸如窗体、类或文档等部件之后表示程序代码。

说明

可用 CodeModule 对象来修改（添加、删除、编辑）与部件相关联的代码。

每个部件都与一个 CodeModule 对象相关联。但是，一个 CodeModule 对象可以与多个代码窗格相关联。

与 CodeModule 对象相关联的方法，能让您操作并返回有关逐行代码文本的信息。例如，可用 AddFromString 方法将文本添加到

模块中。AddFromString 将文本放在第一个过程之上，如果没有过程的话，则将文本放在模块尾端。

可用 Parent 属性返回与一代码模块相关联的 VBComponent 对象。

属性

Lines 属性,Members 属性,Lines 方法,CodePane 属性,CountOf Lines 属性,CountOfDeclarationLines 属性,Parent 属性,Proc BodyLine 属性,ProcCountLines 属性,ProcOfLine 属性,VBE 属性

方法

AddFromFile 方法,AddFromString 方法,GreateEventProc 方法,DeleteLines 方法,Find 方法(VBA 外接程序对象模型),InsertLines 方法,Lines 方法,ReplaceLine 方法,ProcBodyLine 属性,ProcCountLiens 属性,ProcOfLine 属性,ProcStartLine 属性

请参阅

CodePane 对象,CodePanes 集合,VBComponent 对象,VBComponents 集合

CodeModule 属性

返回一个对象，代表与部件相关的代码。此属性为只读。

应用于

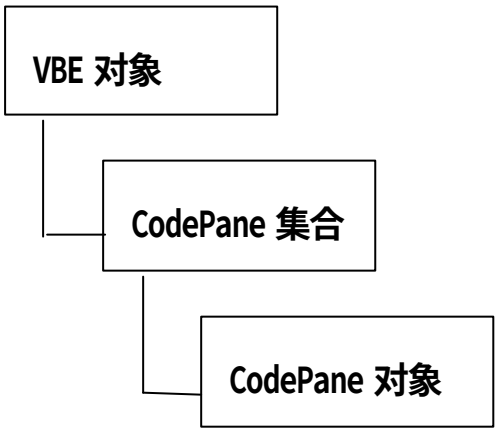
CodePane 对象,VBComponent 对象

说明

如果该部件没有关联的代码模块，CodeModule 属性返回 Nothing。

注意 CodePane 对象代表一个可见的代码窗口，一个给定的部件可以有多个 CodePane 对象，CodeModule 对象代表部件内的代码。一个部件只能有一个 CodeModule 对象。

CodePane 对象



表示代码窗格。

说明

用 CodePane 对象来操作 CodePane 中可视文本的位置或者代码窗格中显示的文本选择。

可用 Show 方法使指定的代码窗格可见。在代码窗格中使用 SetSelection 方法来设置选择，并使用 GetSelection 方法返回代码窗格中的选择位置。

属性

CodeModule 属性，CodePaneViews 属性，Collection 属

性,CountOfVisibleLines 属性,TopLine 属性,VBE 属性,Window 属性

方法

GetSelection 方法,SetSelection 方法,Show 方法(VBAadd-InObjectModel)

特性

CodePanes 集合

请参阅

CodeModule 对象,CodePanes 集合

CodePane 属性

返回一个 CodePane 对象，此属性为只读。

应用于

CodeModule 对象

说明

如果存在一个代码窗格，则它将变成活动的代码窗格，并且，包含它的窗口将变成活动的窗口。如果该模块的代码窗格不存在，则 CodePane 属性将创建一个代码窗格。

特性

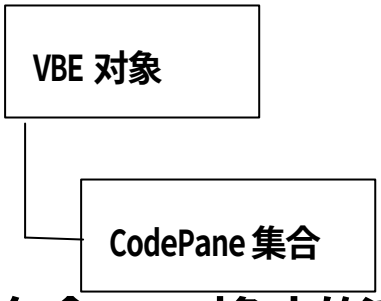
CodePanes 属性

请参阅

Codepane 对象,CodePanes 集合,ActiveCodePane 属性,CodePanes

属性

CodePane 集合



包含 VBE 对象中的活动代码窗格。

说明

使用 CodePane 集合来访问工程中打开的代码窗格。

可用 Count 属性返回集合中的活动代码窗格的数目。

属性

Count 属性,parent 属性,VBE 属性

方法

Item 方法(VBA 外接程序对象模型)

请参阅

CodeModule 对象,CodePane 对象,VBE 对象

CodePanes 属性

返回活动的 CodePan 对象的集合，此属性为只读。

应用于

VBE 对象

请参阅

CodePanes 集合,ActiveCodePane 属性

示例

下列示例使用 CodePanes 及 TopLine 属性来显示代码窗格中的首行行号。

```
Debug.PrintApplication.VBE.CodePanes(3).TopLine
```

CodePaneView 属性

返回一个值，指出代码窗格是“过程查看”还是“全模块查看”。此属性为只读。

应用于

CodePane 对象

返回值

CodePaneView 属性的返回值如下：

常数	描述
vbext_cv_ProcedureView	指定的代码窗格处于“过程查看”模式。
vbext_cv_FullModuleView	指定的工程处于“全模块查看”模式。

示例

下列示例使用 CodePaneView 属性返回一个值，表明指定的代码窗格是在过程视图中，还是在整个模块视图中。

Debug.PrintApplication.VBE.CodePanes(3).CodePaneView

Col，Row 属性

返回或设置 DataGrid 控件中的活动单元，设计时不可用。

语法

object.**Col**[=*number*]

object.**Row**[=*number*]

Col 和 Row 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
numbe	包含活动单元的列或行的数目。

说明

用这些属性来指定 DataGrid 控件中的某一单元，或在选定的区域内查找哪一个行或列含有活动单元。行和列从 0 开始计数，行从顶部开始而列从左边开始计数。运行时设置这些属性不会改变所选的单元。用 SelEndCol、SelStartCol、SelEndRow 和 SelStartRow 属性来指定选择区域。

注意 Col 和 Row 属性与 Cols 和 Rows 属性不同。

请参阅

Text 属性,SelEndCol,SelStartCol,SelEndRow,SelStartRow 属性

示例

这个例子将“Here”放到当前单元中，然后把活动单元改变成第三行中的第三个单元，并且把“There”放到那个单元中。要尝试这个例子，请使用“部件”对话框对工具框添加一个MSFlexGrid控件（在“工程”菜单中，选择“部件”，然后选定MicrosoftFlexGridControl），然后在新窗体中绘制一个网格。要运行该程序，请按F5键，然后再单击该网格。

```
PrivateSubForm_Load()
   MSFlexGrid1.Rows=8   '设置行数和列数。
   MSFlexGrid1.Cols=5
EndSub

PrivateSubMSFlexGrid1_Click()
   '将文本放到当前单元中。
   MSFlexGrid1.Text="Here"
   '将文本放到第三行，第三列。
   MSFlexGrid1.Col=2
   MSFlexGrid1.Row=2
   MSFlexGrid1.Text="There"
EndSub
```

下一个例子显示出活动单元的位置以及当用户选定一个单元或单元范围时的选定范围。注意，在选定一个范围时，活动的单元并

不改变。选定一个范围，然后单击窗体使活动单元沿着选中范围的周边移动。注意，被选中的范围并不改变。

要尝试这个例子，请创建一个新工程，使用“部件”对话框（在“工程”菜单中，选择“部件”，然后检查 MicrosoftFlexGrid Control）添加一个 MSFlexGrid 控件，然后绘制一个 MSFlexGrid 和两个标签。将代码复制到声明部分，然后按 F5 键运行这个程序。

```
PrivateSubForm_Load()
   MSFlexGrid1.Cols=6   '设置行数和列数。
   MSFlexGrid1.Rows=7
EndSub

PrivateSubMSFlexGrid1_RowColChange()
   Msg="ActiveCell:"&Chr(64+MSFlexGrid1.Col)
   Mst=Msg&MSFlexGrid1.Row
   Label1.Caption=Msg
EndSub

PrivateSubMSFlexGrid1_SelChange()
   Msg="Selection:"&Chr(64+MSFlexGrid1.SelStartCol)
   Msg=Msg&MSFlexGrid1.SelStartRow
   Msg=Msg&":"&Chr(64+MSFlexGrid1.SelEndCol)
   Msg=Msg&MSFlexGrid1.SelEndRow
```

```
    Label2.Caption=Msg
EndSub

PrivateSubForm_Click()
    '每单击一次窗体，这一过程
    '就沿着选中
    '范围的周边移动活动单元。
    DimGR,GCAsInteger
    IfMSFlexGrid1.Row=MSFlexGrid1.SelStartRowThen
        IfMSFlexGrid1.Col=MSFlexGrid1.SelEndColThen
            GR=1:GC=0
        Else
            GR=0:GC=1
        EndIf
    ElseIfMSFlexGrid1.Row=MSFlexGrid1.SelEndRowThen
        IfMSFlexGrid1.Col=MSFlexGrid1.SelStartColThen
            GR=-1:GC=0
        Else
            GR=0:GC=-1
        EndIf
    Else
        IfMSFlexGrid1.Col=MSFlexGrid1.SelStartColThen
```

```
            GR=-1:GC=0
        Else
            GR=1:GC=0
        EndIf
    EndIf
    MSFlexGrid1.Row=MSFlexGrid1.Row+GR
    MSFlexGrid1.Col=MSFlexGrid1.Col+GC
EndSub
```

Collection 对象

Collection **对象是项目所组成的有序集合，可以把这个集合作为单元来引用。**

说明

Collection **对象提供了简便方法，直截了当将一组相关的项目视为单一对象来引用。集合中的项目或成员被这样一个事实联系起来：它们都属于这个集合。集合的成员不一定都是同一种数据类型的。**

建立集合的方法与建立其它对象的方法一样。例如：

```
DimXAsNewCollection
```

一旦建立集合之后，就可以用 Add **方法添加成员，用** Remove **方法删除成员。在用** ForEach...Next **语句重复整个集合时，可以用** Item

方法从集合返回特定成员。

属性

Count 属性

方法

Add 方法,Item 方法,Remove 方法

请参阅

ForEach...Next 语句,Add 方法,Item 方法,Remove 方法

示例

本示例建立一个命名为 MyClasses 的 Collection 对象，再建立一个对话框，让用户可从对话框中将对象建立到该集合对象中。若要观察程序如何工作，请先在“插入”菜单上选择“类模块”命令，然后在 Class1 的模块级中声明一个公用变量,命名为 InstanceName（类型为 PublicInstanceName），此变量用来保存每个类实例的名称。不要更改类的缺省名称 Class1。将下列代码复制到另一个模块的“通用”节中，然后在另一个过程中用 ClassNamer 语句使它启动激活。（本示例仅使用支持类的主机应用程序。）

```
SubClassNamer()
   DimMyClassesAsNewCollection      '建立一个集合对象(Collection)。
   DimNum                           '计数用变量，用来对对象的个数计数。
   DimMsgAsString                   '提示信息用变量。
```

```
DimTheName,MyObject,NameList      '对象信息用变体。
Do
    DimInstAsNewClass1            '建立 Class1 的新实例。
    Num=Num+1    '把计数变量 Num 加一，然后要求输入新对象个体的
名称。
    Msg="Pleaseenteranameforthisobject."&Chr(13)_
    &"PressCanceltoseenamesincollection."
    TheName=InputBox(Msg,"NametheCollectionItems")
    Inst.InstanceName=TheName          '将名称送入对象实例。
    '若用户输入了名称，将它加入集合。
    IfInst.InstanceName<>""Then
        '将命名的对象加入集合。
        MyClasses.Additem:=Inst,key:=CStr(Num)
    EndIf
    '清除当前的引用，为对下一个对象做准备。
    SetInst=Nothing
LoopUntilTheName=""
ForEachMyObjectInMyClasses              '建立名称列表。
    NameList=NameList&MyObject.InstanceName&Chr(13)
NextMyObject
'将名称列表在消息框中显示出来。
MsgBoxNameList,,"InstanceNamesInMyClassesCollection"
```

```
    ForNum=1ToMyClasses.Count                    '从集合中删除名字。
        MyClasses.Remove1                        '因为每删除一个对象后，集
合
                                                 '会自动重排顺序，故每次迭代时只需删除第一个
    Next                                         '对象即可。
EndSub
```

Collection 属性

返回一个集合，它包含您正在使用的对象。此属性为只读。

应用于

CodePane 对象,Property 对象,Reference 对象,VBComponent 对象,VBProject 对象,Window 对象,AddIn 对象

说明

大部分这种对象模型的对象，都具有 Parent 属性或 Collection 属性指向该对象的父对象。

使用 Collection 属性，可以访问该对象所在集合的属性、方法及控件。

请参阅

Parent 属性

示例

下列示例使用 Collection 及 Count 属性返回当前活动的工程包含多少对象，如同视其为对象的集合。

Debug.PrintApplication.VBE.ActiveVBProject.Collection.Count

Color 属性

返回或设置选定的颜色。

应用于

CommonDialog 控件

语法

object.**Color**[=*number*]

Color 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
number	指定颜色的数值表达式。

设置值

number 的设置值是：

设置值	描述
normalRGBcolors	在代码中用 RGB 或 QBColor 函数设置颜色。
systemdefaultcolors	用对象浏览器中 VisualBasic(VB)对象库中的系统颜色常数来指定颜色。MicrosoftWindows 操作环境按照用户控件面板设置的指定替换用户选择。

说明

为了该属性能返回“颜色”对话框中的一种颜色，必须先设置

cdlCCRGBInit 标志。在“字体”对话框，必须设置 cdlCFEffects 标志。

数据类型

Long

请参阅

Flags 属性(colorDialog)

ColorMode 属性

返回或设置一个值，以决定彩色打印机是按彩色还是单色打印输出。运行时不可用。

应用于

Printer 对象,printers 集合

语法

object.**ColorMode**[=*value*]

ColorMode 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	常数或整数，指定打印方式，“设置值”中有详细描述

设置值

value 的设置值为:

设置值	值	描述
VbPRCMMonochrome	1	以单色打印输出（一般为黑白阴影）
VbPRCMColor	2	以彩色打印输出

说明

缺省值取决于打印机驱动程序和当前打印机设置。单色打印机忽略该属性。

注意 Printer 对象属性的效果，取决于打印机生产商提供的驱动程序。有些属性设置可能不起作用，或有可能几个不同的属性设置具有相同的结果。如果对不支持彩色的打印机设置 ColorMode 属性，设置值将被忽略。但如果试图引用该 ColorMode 属性，则给出错误信息。在可接受范围外的设置，也有可能导致错误。有关更多的信息，请参阅有关驱动程序的生产商文档。

请参阅

printer 对象,printers 集合

Column 属性

描述

只读属性，返回 TextStream 文件中当前字符位置的列号。

应用于

TextStream 对象

语法

object.**Column**

object 总是一个 TextStream 对象的名称。

说明

在写了一个新行字符后，但在写任何其他字符之前，Column 的值是 1。

请参阅

AtEndOfLine 属性,AtEndOfStream 属性,Line 属性

Columns 属性

返回或设置一个值，以决定 ListBox 控件是水平还是垂直滚动、以及如何显示列中的项目。如果水平滚动，则 Columns 属性决定显示多少列。

应用于

ListBox 控件

语法

object.**Columns**[=*number*]

Columns 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个整型值，指定控件如何滚动、以及列中的项目如何排列，“设置值”中有详细描述

设置值

number 的设置值为：

设置值	描述
0	（缺省值）项目安排在一列中、且 ListBox 竖直滚动。
1 到 n	项目安排在多个列中，先填第一列，再填第二列，等等。ListBox 水平滚动并显示指定数目的列。

说明

对于水平滚动的 ListBox 控件，列宽等于 ListBox 宽度除以列的个数。

该属性不能设置为 0，在运行时也不能从 0 改变—也就是说，不能在运行时，将多列 ListBox 变为单列 ListBox 或将单列 ListBox 变为多列 ListBox。

请参阅

AddItem 方法,RemoveItem 方法,List 属性,ListCount 属性,ListIndex 属性,TopIndex 属性

示例

这个例子说明当包含的数据相同时，两种不同的 ListBox 控件是如何工作的。要试用此例，先把代码粘贴到包含两个 ListBox 控件的窗体的声明部分。并对 List2 设置 Columns 属性为 2，然后按 F5 键并单击窗体。

```
PrivateSubForm_Load()
   DimI                                  '声明变量.
   List1.Move50,50,2000,1750             '排列列表框.
```

```
    List2.Move2500,50,3000,1750
    For I=0ToScreen.FontCount-1        '使用屏幕字体.
        List1.AddItemScreen.Fonts(I)'填充两个列表框
        List2.AddItemScreen.Fonts(I)
    NextI
EndSub
```

ComboBox 控件

ComboBox 控件将 TextBox 控件和 ListBox 控件的特性结合在一起一既可以在控件的文本框部分输入信息，也可以在控件的列表框部分选择一项。

语法

ComboBox

说明

为了添加或删除 ComboBox 控件中的项目，需要使用 AddItem 或 RemoveItem 方法。设置 List、ListCount、和 ListIndex 属性，使访问 ComboBox 中的项目成为可能。也可以在设计时使用 List 属性将项目添加到列表中。

注意只有当 ComboBox 的下拉部分的内容被滚动时，Scroll 事件才在 ComboBox 中发生，而不是每次 ComboBox 的内容改变时。例如，如果 ComboBox 的下拉部分包含五行，并且最顶上的项为突出显示，则在您按完向下箭头键六下（或按一次 PgUp 键）之前

Scroll 事件不发生。再往后，每按一次向上箭头键引发一次 Scroll 事件。

属性

DataMember 属性,DataFormat 属性,RightToLeft 属性,OLEDrag Mode 属性,OLEDropMode 属性,BackColor,foreColor 属性,FontBod, FontItalic,FontStrikethru,FontUnderline 属性,FontName 属性,Font Size 属性,Height,Width 属性,Left,Top 属性,List 属性,ListCount 属性,ListIndex 属性,Sorted 属性,TabIndex 属性,Tag 属性,Text 属性,Visible 属性,DragIcon 属性,DragMode 属性,hWnd 属性,ItemData 属性,Locked 属性,MouseIcon 属性,MousePointer 属性,NewIndex 属性,SelLength,SelStart,SelText 属性,Style 属性,TabStop 属性,TopIndex 属性,Appearance 属性,Enabled 属性,HelpContextID 属性,Index 属性(控件数组),Name 属性,Parent 属性,Font 属性,Container 属性,ToolTipText 属性,WhatsThisHelpID 属性,DataChanged 属性,DataField 属性,IntegralHeight 属性,SelLength,SelStart,SelText 属性(ActiveX 控件),ComboBox

方法

Refresh 方法,SetFocus 方法,AddItem 方法,Clear 方法(剪切板，ComboBox,ListBox)，Drag 方法,Move 方法,RemoveItem 方法, Zorder 方法,OLEDrag 方法,ShowWhatsThis 方法

请参阅

ListBox 控件,TextBox 控件

Command 函数

返回命令行的参数部分，该命令行用于装入 MicrosoftVisualBasic 或 VisualBasic 开发的可执行程序。VisualBasicCommand 函数在 MicrosoftOffice 应用程序中不可用。

语法

Command

说明

当从命令行装入 VisualBasic 时，/cmd 之后的命令行的任何部分作为命令行的参数传递给程序。下面的示例中，cmdlineargs 代表 Command 函数返回的参数信息。

VB/cmdcmdlineargs

对于使用 VisualBasic 开发并编译为.exe 文件的应用程序，Command 返回出现在命令行中应用程序名之后的任何参数。例如：

MyAppcmdlineargs

想知道如何在正在使用的应用程序的用户界面中改变命令行参数，请搜寻关于“命令行参数”的帮助。

示例

本示例在某个函数中用 Command 函数获得命令行参数，并将命令行参数以 Variant 类型之数组返回。在 MicrosoftOffice 中不可

用。

```
FunctionGetCommandLine(OptionalMaxArgs)
    '声明变量。
    DimC,CmdLine,CmdLnLen,InArg,I,NumArgs
    '检查是否提供了 MaxArgs 参数。
    IfIsMissing(MaxArgs)ThenMaxArgs=10
    '使数组的大小合适。
    ReDimArgArray(MaxArgs)
    NumArgs=0:InArg=False
    '取得命令行参数。
    CmdLine=Command()
    CmdLnLen=Len(CmdLine)
    '以一次一个字符的方式取出命令行参数。
    ForI=1ToCmdLnLen
        C=Mid(CmdLine,I,1)
        '检测是否为 space 或 tab。
        If(C<>""AndC<>vbTab)Then
            '若既不是 space 键，也不是 tab 键，
            '则检测是否为参数内含之字符。
            IfNotInArgThen
            '新的参数。
            '检测参数是否过多。
                IfNumArgs=MaxArgsThenExitFor
```

```
                    NumArgs=NumArgs+1
        InArg=True
                EndIf
            '将字符连接到当前参数中。
            ArgArray(NumArgs)=ArgArray(NumArgs)&C
        Else
            '找到 space 或 tab。
            '将 InArg 标志设置成 False。
            InArg=False
        EndIf
    NextI
    '调整数组大小使其刚好符合参数个数。
    ReDimPreserveArgArray(NumArgs)
    '将数组返回。
    GetCommandLine=ArgArray()
EndFunction
```

CommandBar 对象

CommandBar **对象包含其它的** CommandBar **对象，这些对象是作为按钮或菜单命令来用的。**

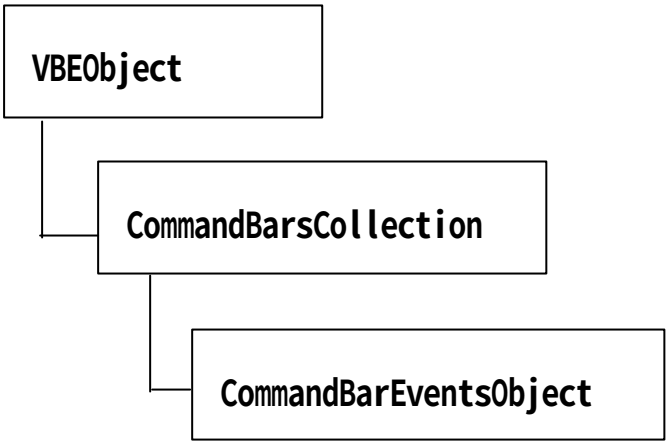
语法

CommandBar

属性

RightToLeft 属性

CommandBarEvents 对象



CommandBarEvents 属性返回的对象。当在命令栏上按下一个控件时，CommandBarEvents 对象将触发一个事件。

说明

Events 对象的 CommandBarEvents 属性返回 CommandBarEvents 对象。返回的对象在接口上有一个 Click 事件。可用 WithEvents 对象声明来处理此事件。

请参阅

CommandBars 集合,Events 对象

CommandBarEvents 属性

返回 CommandBarEvents 对象。此属性为只读。

应用于

Events 对象

设置值

传递给 CommandBarEvents 属性的参数设置值如下：

参数	描述
vbcontrol	必须是一个类型为 CommandBarControl 的对象。

说明

使用 CommandBarEvents 属性来返回一个事件源对象，当一个命令栏按钮被单击时，该事件源对象将触发一个事件。传递给 CommandBarEvents 属性的参数，是将要被触发 Click 事件的命令栏按钮控件。

请参阅

ClickEvent(VBAAdd-InObjectModel),CommandBarEvents 对象,CommandBars 集合,ReferencesEvents 属性

示例

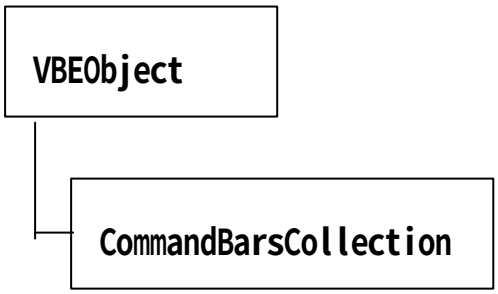
下列示例使用 CommandBarEvents 属性来支持任何处理命令条上鼠标单击事件的代码。

```
PrivateWithEventsceAsCommandBarEvents

SubTest()
DimcAsCommandBarControl
Setc=Application.VBE.CommandBars("Tools").Controls(1)
Setce=Application.VBE.Events.CommandBarEvents(c)
EndSub
```

```
PrivateSubce_Click(ByValCommandBarControlAsObject,HandledAsBoole
an,CancelDefaultAsBoolean)
'将事件的处理程序码写于此处
EndSub
```

CommandBars 集合



包含工程中的所有命令栏，包括支持快速菜单的命令栏。

说明

使用 CommandBars 集合对象之后就可用外接程序来添加命令栏和控件，或是将控件添加到现有的内建命令栏。

属性

Count 属性,Parent 属性,VBE 属性

方法

Item 方法(VBAAdd-InObjectModel)

请参阅

CommandBarEvents 对象,Events 对象,CommandBarEvents 属性

CommandBars 属性

包含工程中的所有命令条，包括支持快捷方式菜单的命令条。

应用于

VBE 对象

请参阅

CommandBars 集合

CommandButton 控件

CommandButton 控件可以开始、中断或者结束一个进程。选取这个控件后，CommandButton 显示按下的形状，所以有时也称之为下压按钮。

语法

CommandButton

说明

为了在 CommandButton 控件上显示文本，需要设置其 Caption 属性。可以通过单击 CommandButton 选中这个按钮。为了能够在按 ENTER 键时也选中命令按钮，需要将其 Default 属性设置为 True。为了能够按 ESC 键时也选中 CommandButton，则需要将 CommandButton 的 Cancel 属性设置成 True。

属性

OLEDropMode 属性,BackColor,ForeColor 属性,Cancel 属性,Font

Bold,FontItalic,FontStrikethru,FontUnderline 属性,Font Name 属性,FontSize 属性,Height,Width 属性,Left,Top 属性,Picture 属性,TabIndex 属性,Tag 属性,Value 属性,Visible 属性,Default 属性,DragIcon 属性,DragMode 属性,hWnd 属性,MouseIcon 属性,MousePinter 属性,Style 属性,TabStop 属性,Appearance 属性,Caption 属性,Enabled 属性,HelpcontextID 属性,Index 属性(ControlArray),Name 属性,Parent 属性,Font 属性,Container 属性,ToolTiopText 属性,DisabledPicture 属性 ,DownPicturePorperty,MaskColor 属性 ,UseMaskColor 属性,WhatsThisHelpID 属性,CommandButton 控件(LightWeight),CAption 属性(ActiveX 控件)

方法

Refresh 方法,SetFocus 方法,Drag 方法,Move 方法,Zorder 方法,OLEDrag 方法,ShowWhatsThis 方法

事件

Click 事件,DragDrop 事件,DragOver 事件,GotFocus 事件,KeyDown,KeyUp 事件,KeyPress 事件,LostFocus 事件,MouseDown,MouseUp 事件,ValidateEvnet,OLECompleteDrag 事件,OLEDragDrop 事件,OLEDragOver 事件,OLEGiveFeedback 事件,OLESetData 事件,OLEStartDrag 事件

请参阅

Cancel 属性,Default 属性,Caption 属性

Comments 属性

返回或设置一个字符串，该字符串包括运行中的应用程序的注释。该属性在运行时是只读的。

应用于

App 对象

语法

object.Comments

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

在设计时，使用位于“工程属性”对话框中“生成”选项卡上的“类型”框可设置该属性。

请参阅

CompanyName 属性,FileDescrioption 属性,LegalCopyright 属性,LegalTrademarks 属性,ProductName 属性

CommonDialog 控件

CommonDialog 控件提供一组标准的操作对话框，进行诸如打开和保存文件，设置打印选项，以及选择颜色和字体等操作。通过运行 Windows 帮助引擎控件还能显示帮助。

语法

CommonDialog

说明

CommonDialog 控件在 VisualBasic 和 MicrosoftWindows 动态链接库 ommdlg.dll.的例程之间提供了一个接口。为了用这个控件创建一个对话框，ommdlg.dll.必须在 MicrosoftWindows 的 SYSTEM 目录下。

在应用程序中要使用 CommonDialog 控件，可将其添加到窗体中并设置其属性。控件所显示的对话框由控件的方法确定。在运行时，当相应的方法被调用时，将显示一个对话框或是执行帮助引擎；在设计时，CommonDialog 控件是以图标的形式显示在窗体中。该图标的大小不能改变。

使用指定的方法，CommonDialog 控件能够显示下列对话。

方法	所显示的对话框
ShowOpen	显示“打开”对话框
ShowSave	显示“另存为”对话框
ShowColor	显示“颜色”对话框
ShowFont	显示“字体”对话框
ShowPrinter	显示“打印”或“打印选项”对话框
ShowHelp	调用 Windows 帮助引擎

在对话框接口上单击，CommonDialog 控件将自动提供与上下文有关的帮助：

单击标题栏中的“这是什么？”帮助按钮，然后单击想详细信息

的项目。

将鼠标放在想进一步详细信息的项目上，单击右键，然后在所显示的上下文菜单中选择这是什么命令。

操作系统提供在 Windows95 帮助弹出中显示的文本。也可以通过设置 Flags 属性，在带有 CommonDialog 控件的对话框中显示一个帮助按钮，但是，必须在这个位置提供帮助主题。

注意无法指定对话框显示在什么地方。

详细信息要查看各对话的帮助主题，单击“请参阅”。

属性

Action 属性(CommonDialog),CancelError 属性,Color 属性,DefaultExt 属性,DialogTitle 属性,FileTitle 属性,Filter 属性(CommonDialog),FilterIndex 属性,Flags 属性(ColOrDialog),Flags 属性(Open,SaveAsDialogs),Flags 属性(FontDialog),Flags 属性(PrintDialog),FromPage,ToPage 属性,HelpCommand 属性,HelpContext 属性(CommonDialog),HelPKey 属性,InitDir 属性,Max,Min 属性(CommonDialog),MaxFileSize 属性,PrinterDefault 属性,Orientation 属性(CommonDialog 控件),HelpFile 属性(App,CommonDialog,MenuLine),FontBold,FontItalic,FontStrikethru,FontUnderline 属性,FontName 属性,FontSize 属性,Left,Top 属性,Copies 属性,FileName 属性,hDC 属性,Name 属性,Parent 属性,hDC 属性(ActiveX 控件),Index 属性(ActiveX 控件),Object 属性(ActiveX 控件),FontName 属性(ActiveX 控件),FontSize 属性(ActiveX 控件)

方法

ShowColor 方法,showFont 方法,ShowHelpMehtod,ShowOpen 方法,ShowPrinterMerthod,ShowSave 方法,AboutBox 方法

请参阅

CommonDialog 控件(color 对话框),CommonDialog 控件（Font 对话框）,CommonDialog 控件(Open,SaveAs 对话框),CommonDialog 控件(Print 对话框)，commonDialog 控件(Help),Flags 属性(Color 对话框),Flags 属性(Open,SaveAs 对话框),Flags 属性(Font 对话框),Flags 属性(Print 对话框)

CommonDialog 控件（“颜色”对话框）

通过使用 CommonDialog 控件的 ShowColor 方法可显示“颜色”对话框。“颜色”对话框用以从调色板选择颜色，或是生成和选择自定义颜色。

如要使用“颜色”对话框，先设置 CommonDialog 控件中与颜色对话相关的属性。然后使用 ShowColor 方法显示对话框，使用 Color 属性检索所选的颜色。

属性

Action 属性(CommonDialog),CancelError 属性,Color 属性,Flags 属性(ColorDialog),HelpCommand 属性,HelpContext 属性(CommonDialog),HelpoKey 属性,HelpFile 属性(Ap,CommonDialog,MenuLine),Left,Top 属性,Name 属性,Parent 属性,Index 属

性(ActiveX控件),Object属性(ActiveX控件),Tag属性(ActiveX控件)

方法

ShowColor 方法

请参阅

CommonDialog 控件(FontDialog),CommonDialog 控件(Open, SaveAsialogs),CommonDialog 控件(PrintDialog),Common Dialog 控件(Help),CommonDialog控件

示例

下面的示例显示“颜色”对话框并设置窗体的 BackColor 为选定的颜色：

```
PrivateSubCommand1_Click()
'设置“取消”为True
CommonDialog1.CancelError=True
OnErrorGoToErrHandler
'设置Flags 属性
CommonDialog1.Flags=cdlCCRGBInit
'显示“颜色”对话框
CommonDialog1.ShowColor
'设置窗体的背景颜色为选定的颜色
Form1.BackColor=CommonDialog1.Color
ExitSub
```

```
ErrHandler：
'用户按了“取消”按钮
EndSub
```

CommonDialog 控件（“字体”对话框）

使用 CommonDialog 控件的 ShowFont 方法可显示“字体”对话框。“字体”对话框用以通过指定字体、大小、颜色、样式选择一种字体。

如要使用“字体”对话框,先设置 CommonDialog 控件中与字体对话相关的属性。然后使用 ShowFont 方法实际显示对话。一旦在“字体”对话框中作出了选择，下列属性即包括与该选择有关的信息：

属性	决定
Color	选定的颜色。如要使用这个属性，必须先将 Flags 属性设置为 cdlCFEffects。
FontBold	是否选定了粗体。
FontItalic	是否选定了斜体。
FontStrikethru	是否选定删除线。如要使用这个属性，必须先将 Flags 属性设置为 cdlCFEffects。
FontUnderline	是否选定下划线。如要使用这个属性，必须先将 Flags 属性设置为 cdlCFEffects。
FontName	选定字体的名称。
FontSize	选定字体的大小。

属性

Action 属性(CommonDialog),CancelError 属性,Color 属性,Flags 属性(FontDialog),Helopcommadn 属性,HelpContext 属性(CommonDialog),HelpKey 属性,Max,Min 属性(CommOnDialog),HelpFile 属性(App,CommonDialog,MenuLine),FontBold,FontItalic,FontStrikethru,FontUnderline 属性,FontName 属性,FontSize 属性,Left,TopPropeties,Name 属性,Parent 属性,Index 属性(ActiveX 控件),Object 属性(ActiveX 控件),Tag 属性(ActiveX 控件)

方法

ShowFont 方法

示例

下面的示例显示“字体”对话框，将文本框中的字体属性设置为所选定的：

```
PrivateSubCommand1_Click()
'设置“取消”为True
CommonDialog1.CancelError=True
OnErrorGoToErrHandler
'设置Flags 属性
CommonDialog1.Flags=cdlCFEffectsOrcdlCFBoth
'显示“字体”对话框
CommonDialog1.ShowFont
Text1.Font.Name=CommonDialog1.FontName
Text1.Font.Size=CommonDialog1.FontSize
Text1.Font.Bold=CommonDialog1.FontBold
Text1.Font.Italic=CommonDialog1.FontItalic
Text1.Font.Underline=CommonDialog1.FontUnderline
Text1.FontStrikethru=CommonDialog1.FontStrikethru
Text1.ForeColor=CommonDialog1.Color
ExitSub
ErrHandler：
'用户按了“取消”按钮
```

```
ExitSub
EndSub
```

CommonDialog 控件（帮助）

CommonDialog 控件的 ShowHelp 方法可运行 Windows 的帮助引擎(WINHELP.EXE)，并显示由 HelpFile 属性设定的一个帮助文件。通过 HelpCommand 属性的设置，可以告诉该帮助引擎想要哪种类型的联机帮助，比如是上下文相关，或是特定关键字的帮助，等等。

属性

Action 属性(CommonDialog),HelpComand 属性,HelpKey 属性,HelpFile 属性(App,CommonDialog,MenuLine),Left,Top 属性,HelpContextID 属性,Name 属性,Parent 属性,Index 属性(ActiveX 控件),Object 属性(ActiveX 控件),Tag 属性(ActiveX 控件)

方法

ShowFont 方法

请参阅

CommonDialog 控件(ColorDialog),CommonDialog 控件(FontDialog),CommonDialog 控件(Open,SaveAsDialogs),CommonDialog 控件(printDialog),CommonDialog 控件

示例

下面示例示范了几个帮助命令。要试验该示例，在窗体中放一个

CommonDialog 控件、五个 CommandButton 控件，将代码粘贴到“声明”部分，按下 F5 键并单击每个按钮。

```
OptionExplicit
ConstHelpCNT=&HB

PrivateSubCommand1_Click()
   WithCommonDialog1
        '必须设置 Help 文件名。
        .HelpFile="VB5.hlp"
        '显示目录。注意 HelpCNT 常数不是一个内部常数。
        'cdlHelpSetContents 确保了只显示目录
        '(而不显示索引或查找)。
        .HelpCommand=HelpCNTOrcdlHelpSetContents
        .ShowHelp
   EndWith

EndSub

PrivateSubCommand2_Click()
   WithCommonDialog1
        .HelpFile="VB5.hlp"
        '进入“Help“文件的 ClickEvent 标题。
```

```
        '“.chm”文件的数字值是由“.HPJ”文件
        '的[MAP]部分决定的。只有当你用
        '“MicrosoftHelpWorkshop“建立
        '自己的帮助文件时，
        '才可以编辑此数字值。
        .HelpContext=916302
        .HelpCommand=cdlHelpContext
        .ShowHelp
    EndWith
EndSub

PrivateSubCommand3_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
        '显示关于帮助的帮助.
        .HelpCommand=cdlHelpHelpOnHelp
        .ShowHelp
    EndWith
EndSub

PrivateSubCommand4_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
```

```
        .HelpKey="data"
        '显示所选择关键字的索引。
        .HelpCommand=cdlHelpKey
        .ShowHelp
    EndWith
EndSub

PrivateSubCommand5_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
        .HelpKey="arrays,"
        '显示用“HelpKey“找到的标题列表。
        .HelpCommand=cdlHelpPartialKey
        .ShowHelp
    EndWith

EndSub

PrivateSubForm_Load()
    '标记“CommandButton”控件。
    Command1.Caption="Contents"
    Command2.Caption="SpecifiedTopic"
```

```
    Command3.Caption="HelpOnHelp"
    Command4.Caption="IndexofTopics"
    Command5.Caption="FoundTopics"
EndSub
```

CommonDialog控件（“打开”、“另存为”对话框）

通过使用 CommonDialog 控件的 ShowOpen 和 ShowSave 方法可显示“打开”和“另存为”对话框。

两个对话框均可用以指定驱动器，目录，文件扩展名和文件名。

除对话的标题不同外，另存为对话外观上与打开对话相似。

在运行时，当用户选择一个文件“关闭”对话框时，FileName 属性既为选定的文件名。

可以设置 Filter 属性，这样对话就只显示某种文件类型，如文本文件。Flags 属性可用来改变对话的元素，当诸如覆盖文件之类的动作发生时，还可用来提示用户。

属性

Action 属性(CommonDialog),CancelError 属性,DefaultExt 属性,DialogTitle 属性,FileTitle 属性,Filter 属性(CommonDialog),FilterIndex 属性,Flags 属性(Open,SaveAsDialogs),HelpKey 属性,InitDir 属性,MaxFileSize 属性,HelopFile 属性(App,CommonDialog,MenuLine),Left,Top 属性,FilName 属性,Name 属性,Parent 属性,Index 属性(ActiveX 控件),Object 属性

(ActiveX 控件),Tag 属性(ActiveX 控件)

方法

ShowOpen 方法,ShowSave 方法

请参阅

CommonDialog 控件(ColorDialog),CommonDialog 控件(FontDialog),CommonDialog 控件(PrintDialog),CommonDialog 控件(Help),CommonDialog 控件

示例

下例显示“打开”对话框然后在信息框中显示所选的文件名：

```
PrivateSubCommand1_Click()
'设置“CancelError”为 True
CommonDialog1.CancelError=True
OnErrorGoToErrHandler
'设置标志
CommonDialog1.Flags=cdlOFNHideReadOnly
'设置过滤器
CommonDialog1.Filter="AllFiles(*.*)|*.*|TextFiles"&_
"(*.txt)|*.txt|BatchFiles(*.bat)|*.bat"
'指定缺省的过滤器
CommonDialog1.FilterIndex=2
'显示“打开”对话框
CommonDialog1.ShowOpen
```

```
'显示选定文件的名字
MsgBoxCommonDialog1.filename
ExitSub

ErrHandler：
'用户按了“取消”按钮
ExitSub
EndSub
```

CommonDialog 控件（“打印”对话框）

通过使用 CommonDialog 控件的 ShowPrinter 方法可显示“打印”对话框。“打印”对话框可用以指定打印输出方式。可以指定被打印页的范围，打印质量，打印的份数等等。这个对话框还包含当前安装的打印机的信息，并允许配置或重新安装缺省打印机。注意这个对话框并不给打印机传送数据，只是指定希望打印数据的情况。如果 PrinterDefault 属性为 True，可以使用 Printer 对象按选定的格式打印数据。

在运行时，一旦“打印”对话框中作出选择，下列属性即包括与该选择有关的信息：

属性	决定
copies	打印的份数。
fromPage	开始打印页。
toPage	结束打印页。
HDC	所选打印机的设备描述体。

属性

CancelError 属性,Flags 属性(PrintDialog),FromPage,ToPage Prperties,HelpCommand 属性,HelpContext 属性(CommonDialog),HelopKey 属性,MaxMin 属性(CommonDialog),MaxFileSize 属性,PrinterDefault 属性,HelpFile 属性(Appm,CommonDialog,MenuiLine),Left,Top 属性,Copies 属性,hDC 属性,ActiveConTrol 属性,Name 属性,Parent 属性, Index 属性(ActiveX 控件),Object 属性(ActiveX 控件),Tag 属性(ActiveX 控件)

方法

ShowPrinter 方法

请参阅

CommonDialog 控件(ColorDialog),CommonDialog 控件(FoNtDialog),CommonDialog 控件(Open,SaveAsDialogs),CommonDialog 控件(Help),OrOperator,hDC 属性,Const 语句

示例

下例显示“打印”对话框。

```
PrivateSubCommand1_Click()
DimBeginPage,EndPage,NumCopies,i
'设置“取消”为True
CommonDialog1.CancelError=True
OnErrorGoToErrHandler
'显示“打印”对话框
CommonDialog1.ShowPrinter
'从该对话框取得选定的值
BeginPage=CommonDialog1.FromPage
EndPage=CommonDialog1.ToPage
NumCopies=CommonDialog1.Copies
Fori=1ToNumCopies
'此处放置将数据发送到打印机的代码
Nexti
ExitSub
ErrHandler：
'用户按了“取消”按钮
ExitSub
EndSub
```

CompanyName 属性

返回或设置一个字符串，该字符串包括运行中的应用程序的公司

或创建者名称。该属性在运行时是只读的。

应用于

App 对象

语法

object.CompanyName

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

在设计时，使用位于“工程属性”对话框中“生成”选项卡上的“类型”框可设置该属性。

请参阅

Comments 属性,FileDescription 属性,LegalCopyright 属性,LegalTrademarks 属性,ProductName 属性

CompareMode 属性

设置或返回某个 Dictionary 对象中的比较字符串关键字的比较模式。

应用于

Dictionary 对象

语法

object.CompareMode[=*compare*]

CompareMode 属性具有下列部分：

部分	描述
object	必需的。总是一个 Dictionary 对象的名称
compare	可选的。如果提供的话，compare 是一个代表比较模式的值，该比较模式用于象 StrComp 这样的函数

设置

comp*are* 参数可以具有下列值：

常数	值	描述
vbUseCompareOption	–1	使用 OptionCompare 语句的设置值进行比较。
vbBinaryCompare	0	进行二进制比较。
vbTextCompare	1	进行文字比较。
vbDatabaseCompare	2	仅用于 MicrosoftAccess。进行基于您自己数据库中信息的比较。

说明

如果试图对已经包含数据的 Dictionary 对象的比较模式进行更改的话，就会出错。

CompareMode 属性所用的参数值与 StrComp 函数所用的 compare 参数相同。可以用大于 2 的值表示使用特定 LocaleIDs(LCID)的比较。

请参阅

Count 属性,Item 属性,Key 属性

CompatibleOLEServer 属性

该属性返回或设置该工程兼容的 ActiveX 部件。

应用于

VBProject 对象

语法

object.**CompatibleOLEServer**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Connect 属性

返回某个外接程序的连接状态，或者对其进行设置。

应用于

AddIn 对象

语法

object.**Connect**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

如果外接程序注册了，并且当前是连接的（活动的），则返回 True。

如果外接程序注册了，但当前未连接（未激活），则返回 False。

Const 语句

声明用于代替文字量的常数。

语法

[Public|Private]Const*constname*[As*type*]=*expression*

Const 语句的语法包含下面部分：

部分	描述
public	可选的。该关键字用于在模块级别中声明在所有模块中对所有过程都可以使用的常数。在过程中不能使用
private	可选的。该关键字用于在模块级声明只能在包含该声明的模块中使用的常数。不能在过程中使用
constname	必需的。常数的名称；遵循标准的变量命名约定
type	可选的。常数的数据类型；可以是 Byte、布尔、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String 或 Variant。所声明的每个变量都要使用一个单独的 As 类型子句
expression	必需的。文字，其它常数，或由除 Is 之外的任意的算术操作符和逻辑操作符所构成的任意组合

说明

在缺省情况下常数是私有的。过程中的常数总是私有的；它们的可见性无法改变。在标准模块中，可以用 Public 关键字来改变模块级常数可见性的缺省值。不过，在类模块中，常数只能是私有的，而且用 Public 关键字也不能改变其可见性。

为了在一行中声明若干个常数，可以使用逗号将每个常数赋值分开。用这种方法声明常数时，如果使用了 Public 或 Private 关键字，则该关键字对该行中所有常数都有效。

在给常量赋值的表达式中，不能使用变量，用户自定义的函数，或 VisualBasic 的内部函数（如 Chr）。

注意常数可以使程序更具可读性，以及易于修改。在程序运行时，常数不会象变量那样无意中被改变。

如果在声明常数时没有显式地使用 Astype 子句，则该常数的数据类型是最适合其表达式的数据类型。

在 Sub、Function 或 Property 过程中声明的常数都是该过程的局部常数。在过程外声明的常数，在包含该声明的模块中被定义。在可以使用表达式的地方，都可以使用常数。

请参阅

#ConstDirective,Deftype 语句,Function 语句,Let 语句,PropertyGet 语句,PropertyLet 语句,Set 语句,Sub 语句

示例

该示例使用 Const 语句来声明用于代替文字值的常数。Public 常数在标准模块的通用部分声明，而不是在类模块中声明。Private 常数可以在任何模块类型的通用部分声明。

```
'缺省情况下常数是私有的。
Const MyVar=459
```

```
'声明公用常数。
PublicConstMyString="HELP"

'声明私有的整数常数。
PrivateConstMyIntAsInteger=5

'在一行中声明多个常数。
ConstMyStr="Hello",MyDoubleAsDouble=3.4567
```

ContainedControls 集合

一个集合，允许访问使用控件的开发者所添加控件中的控件。

语法

ContainedControls(*index*)

index 所在处表示从 0 到 ContainedControls.Count–1 的整数。

请参阅

ContainedControls 属性,ContainedVBControls 集合

ContainedControls 属性

返回一个控件的集合，该集合中的控件将由开发者或最终用户在控件运行时添加到控件中去。在控件创建时，ContainedControls 属性不可用，在控件运行时，该属性是不可用。

应用于

UserControl 对象

语法

object.**ContainedControls**

ContainedControls 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

ContainedControls 集合被开发者或最终用户添加到控件中的所有控件填满。控件可以使用 ContainedControls 集合对其中任何一个控件执行操作。

此集合的功能类似于窗体上的控件集合。

要使包含的控件能够放置到控件上，必须将 ControlContainer 属性设置为 True。

不能通过 ContainedControls 集合添加或删除包含的控件；必须以容器允许的方式变更包含控件。

并非所有的容器都支持 ContainedControls 属性，即使该容器支持包含控件的控件；但是，VisualBasic 窗体的确支持这个属性。如果不支持该属性，那么调用 ContainedControls 集合将导致错误；因此，在访问集合时应使用错误处理。然而请注意：如果在事件过程中（例如，InitProperties 事件过程或 ReadProperties 事件过程）调用错误处理，则错误处理程序不应产生错误事件；

因为这样做对容器来说可能是致命的。

在执行初始化事件时，ContainedControls 集合不可用；但是执行 InitProperties 事件或 ReadProperties 事件时，此集合可用。当 ContainedControls 集合出现时，它也许没有立即包含对于开发者放置到控件上的控件的引用。例如，如果控件在 VisualBasic 窗体上，则在所有 ReadProperties 事件过程执行完成之前，ContainedControls 集合的 Count 属性值都为 0。

请参阅

InitPropertiesEvent,ReadPropertiesEvent,InitializeEvent,控件集合,Count 属性(VB 集合),Item 属性(ActiveX 控件)

ContainedVBControls 集合

ContainedVBControls 集合代表 VBControl 对象的集合。

属性

Count 属性(VB 集合),Parent 属性,VBE 属性

方法

Add 方法,Item 方法,Remove 方法(VisualBasicExtensibility)

请参阅

ContainedControlsPrperty,VBControl 对象

ContainedVBControls 属性

返回包含控件的集合。

应用于

VBControl 对象,VBForm 对象

语法

object.**ContainedVBControls**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Container 属性

返回或设置 Form 上控件的容器。在设计时不能使用。

应用于

CheckBox 控件,ComboBox 控件,CommandButton 控件,DirListBox 控件,DriveListBox 控件,FileListBox 控件,Frame 控件,HscrollBar,VScrollBar 控件,Image 控件,Label 控件,Line 控件,LisBox 控件,OptionButton 控件,PictureBox 控件,Shape 控件,TextBox 控件,OLE 容器控件,RichTextBox 控件,RemoteData 控件,SSTab 控件,MSFlexGrid 控件,MSHFlexGrid 控件,MSChart 对象,MultimediaMCI 控件,MaskedEdit 控件,DBList 控件,DBCombo 控件,Datalist 控件,DataRepeater 控件,CoolBar 控件,UpDown 控件,MonthView 控件,FlatScrollBar 控件,Anomation 控件,DateTimePicker 控件,Toolbar 控件,TabStrip 控件,StatusBar 控件,Slider 控

件,ProgressBar 控件,ListView 控件,ImageCombo 控件,TreeView 控件,ADOData 控件

语法

Set*object*.**Container** [=*container*]

Container 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
container	一个对象表达式其值是能够作为别的控件容器使用的对象，按照说明的描述

说明

下面的控件能够容纳别的控件：

Frame 控件

PictureBox 控件.

SSTab 控件

请参阅

Frame 控件,PictureBox 控件,Parent 属性

示例

该例子演示在 Form 对象上把 CommandButton 控件从一个容器移动到另一个容器。要试用此例，先将以下代码粘贴到包含一个 Frame 控件、一个 PictureBox 控件和一个 CommandButton 控件的窗体的声明部分，然后按下 F5 键。

```
PrivateSubForm_Click()
```

```
    StaticintXAsInteger
    SelectCaseintX
        Case0
            SetCommand1.Container=Picture1
            Command1.Top=0
            Command1.Left=0
        Case1
            SetCommand1.Container=Frame1
            Command1.Top=0
            Command1.Left=0
        Case2
            SetCommand1.Container=Form1
            Command1.Top=0
            Command1.Left=0
    EndSelect
    intX=intX+1
EndSub
```

Container 属性

返回容器控件或窗体。

应用于

CoolBar 控件,DataCombo 控件,VBControl 对象

语法

object.**Container**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

ContainerHWnd 属性

返回一个 UserControl 容器的窗口句柄（hWnd）。

语法

object.**ContainerHWnd**

object 所在处表示一个对象表达式，其值为一个 UserControl 对象。

说明

ContainerHWnd 属性返回 UserControl 对象所位于的容器的句柄。ContainerHWnd 可以作为参数被传送到要求一个 hWnd 参数的 WindowsAPI 调用。

当 UserControl 的 Windowless 属性被设置为 True 时，UserControl 不再有它自己的句柄。在某些情况下，ContainerHWnd 属性可以用来代替 hWnd 属性，使 UserControls 获取有关一个窗口的信息。ContainerHWnd 不能被用于修改 hWnd 值的 WindowsAPI 调用中。

ContinuousScroll 属性

该属性返回或者设置一个值，该值决定了滚动是否是连续的，或

者是否仅当释放滚动缩微化时，UserDocument 才会重画。

应用于

UserDocument 对象

语法

object.**ContinuousScroll**=*boolean*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
boolean	布尔表达式，它指定了滚动是否连续

设置值

布尔设置值是：

设置值	描述
True	缺省的。滚动是连续的。
False	仅当释放缩微化时，UserDocument 才重画。

Control 类

VisualBasic 提供 Control 类作为控件的一般变量类型。当声明一个变量为 AsControl，就可以将任何控件引用赋给该变量。用户不能创建 Control 类的实例。

注意如果使用与控件声明为同一类型的变量（例如，AsTreeView 或 AsCommandButton），则访问控件的属性和方法要更快。。因为 VisualBasic 可以使用事前绑定。VisualBasic 必须使用后期绑

定对赋给一个声明为 AsControl 变量的控件的属性和方法进行访问。

Control 对象

所有 VisualBasic 内部控件的类名。

语法

Control

说明

可以将一个变量标为 Control 对象，象把控件放到窗体上一样来引用它。下面对此进行演示：

```
DimCasControl
SetC=Command1
```

ControlBox 属性

返回或设置一个值，指示在运行时控制菜单框是否在窗体中显示。在运行时为只读。

应用于

Form 对象,Forms 集合

语法

object.**ControlBox**

object 所在处表示对象表达式，其值是“应用于”列表中的一个

对象。

设置值

ControlBox 属性设置值如下：

设置值	描述
True	（缺省值）显示控制菜单框。
False	删除控制菜单框。

说明

为了显示控制菜单框，还必须将窗体的 BorderStyle 属性值设置为 1（固定单边框），2（可变尺寸）或 3（固定对话框）。

模式的和无模式的窗口都可以包含一个控制菜单框。

相关属性的设置值决定运行时可以使用的命令－例如，将 MaxButton 和 MinButton 设置为 False，将使控制菜单中的最大化和最小化命令无效，但移动和关闭命令仍然有效。

注意给 ControlBox、BorderStyle、MaxButton 和 MinButton 属性指定的设置值，只有在运行时才能从窗体的外观上表现出来。

请参阅

BorderStyle 属性,MaxButton 属性,MinButton 属性,BorderStyle 属性(ActiveX 控件)

ControlContainer 属性

返回或设置一个值，此值决定控件是否能包含由开发者或最终用

户在控件运行时放在它上面的控件；以及 PictureBox 控件是否能够包含其它控件。在控件创建时，ControlContainer 属性可读可写，在控件运行时，此属性是只读的。

应用于

UserControl 对象

设置值

ControlContainer 的设置值为：

设置值	描述
True	控件可以包含放在它上面的控件。如果将这种控件的实例放置在不能感知 ISimpleFrame 的容器上，支持包含控件的功能将被禁止。控件在其它方面仍可正常工作，但开发者或最终用户无法将其它控件放置到这种控件的实例上。
False	（缺省）控件不能包含放在它上面的其它控件。

说明

在 VisualBasic 窗体上是支持包含控件的。

放置在有透明背景的控件上的包含控件，只有当它们的位置与任何子控件重叠时才可见。仅当鼠标事件发生在包含控件可见的地方，这些鼠标事件才传递给包含控件。

请参阅

PictureBox 控件

ControlObject 属性

该属性返回对设计时 IDispatch 指针实例的引用，该指针是由控件提供的。如果没有实例，则返回 Nothing。

应用于

VBControl 对象

语法

object.**ControlObject**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

例如，Toolbar 控件通过属性页的 ControlObject 属性提供一个对象来设置按钮号。

Controls 集合

一个集合，其元素代表部件中的控件。Controls 集合的 Count 属性用于表明集合中的控件数量，而其 Item 方法则返回集合中的一个成员。

语法

object.**Controls.Count**

object.**Controls**(*index*)

Controls 集合的语法包括下述部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的对象
index	一个整数，范围从 0 到 Controls.Count–1。

注意如果部件是一个 VisualBasic 模块，如 Form 或 UserControl，则在模块中编写代码时不必使用该对象表达式。然而，如果容器是一个编译后的 ActiveX 控件，如 ToolBar 控件，则必须使用该对象表达式。

说明

Controls 集合枚举部件中装入的控件，可用于对这些控件的遍历。例如，可以用来改变一个 Form 中所有 Label 控件的 BackColor 属性。

Controls 集合标识了一个内在的名为 Controls 的窗体级变量。如果省略了可选的 object 占位符，则必须包括 Controls 关键字。不过，如果包括了 object，则可以省略 Controls 关键字。例如，下面两行代码具有相同的作用：

```
MyForm.Controls(6).Top=MyForm.Controls(5).Top+increment
MyForm(6).Top=MyForm(5).Top+increment
```

可以将 Controls(index)传递给一个参数指定为 Controls 类的函数。也可以使用它们的名称来访问成员。例如：

```
Controls("Command1").Top
```

可以在 If 语句中使用 TypeOf 关键字，或使用 TypeName 函数来确定 Controls 集合中控件的类型。

注意 Controls 集合不是 VisualBasicCollection 类的成员。其属性和方法的集合要小于 Collection 对象的属性和方法的集合，而且用户不能创建该集合的实例。

方法

Add 方法(Controls 集合)

请参阅

Form 对象,Forms 集合,Licenses 集合,Add 方法(Licenses 集合)

Controls 属性

返回对 Control 对象集合的引用。

应用于

Section 对象(数据报表设计器),PropertyPage 对象,UserControl 对象,UserDocument 对象,Form 对象,Forms 集合,MDIForm 对象

语法

object.**Controls**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

可以使用 Controls 属性所返回的引用对 Control 对象进行操作。

请参阅

PropertyPage 对象,UserDocument 对象,Form 对象,Forms 集合

,MDIForm 对象

ControlType 属性

该属性返回由控件创建的运行时窗口的类型。

应用于

VBControl 对象

语法

object.**ControlTypeAsvbext_ControlType**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

设置值

vbext_ControlType 的设置值是：

Constant 常数	值	描述
vbext_ct_Light	1	（缺省值）没有 hWnd 在运行时。
vbext_ct_Standard	2	hWnd 在运行时。
vbext_ct_Container	3	hWnd 在运行时，且可包含其它控件。

Copies 属性

返回或设置需要打印的份数。对于 Printer 对象，在设计时不可用。

应用于

CommonDialog 控件(PrintDialog),Printer 对象,Printers 集合

语法

object.**Copies**[=*number*]

Copies 属性语法包含下面部分:

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	数值表达式，指定需要打印的份数。该值必须是整型值

说明

对于“打印”对话框，该属性返回在“份数”框中用户输入的份数。如果设置 CommonDialog 控件的 cdlPDUseDevModeCopies 标志，则该属性始终返回 1。

对于 Printer 对象，对多份打印可能进行、也可能不进行核对，这取决于打印机驱动程序。可以将整个文档或将每一页打印多份。对于不支持核对的打印机，设置 Copies=1，然后在程序中使用循环，就可以将整个文档打印多份。

注意 Printer 对象属性的效果取决于打印机生产商提供的驱动程序。一些属性设置可能不起作用，或几个不同的属性设置具有相同的结果。如果设置值超出可接受范围，就会产生错误。更多的信息，参阅有关驱动程序的生产商文档。

请参阅

Duplex 属性,PaperBin 属性

Copy 方法

将窗体中所选的控件复制到剪贴板上。

语法

object.**Copy**

object 所在处表示对象表达式，其值是“应用于”列表中的对象。

Copy 方法（OLE 容器控件）

将 OLE 容器控件内的对象复制到系统剪贴板。

应用于

OLE 容器控件

语法

object.**Copy**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

当将对象复制到系统剪贴板时，所有与该对象关联的数据和链接信息均被放置到系统剪贴板上。链接的对象和嵌入的对象都能复制到剪贴板。

使用这个方法可以支持菜单中的“编辑复制”命令。

请参阅

Paste 方法,MAPISession 控件,MAPIMessages 控件,MsgIndex 属性

Copy 方法

把一个指定的文件或文件夹从一个地方复制到另一个地方。

应用于

File 对象,Folder 对象

语法

object.Copydestination[,overwrite]

Copy 方法语法有如下几部分：

部分	描述
Object	必需的。始终是一个 File 或 Folder 对象的名字
Destination	必需的。文件或文件夹要复制到的接受端。不允许有通配符
Overwrite	可选的。Boolean 值，如果该值为 True（缺省），则已存在的文件或文件夹将被覆盖。如果为 False，则它们不被覆盖

说明

对一个 File 或 Folder，Copy 方法的结果和执行 FileSystem Object.CopyFile 或 FileSystemObject.CopyFolder 操作的结果是一样的，在后者中，object 所引用的文件或文件夹是作为参数传递的。应当注意，后面的方法能够复制多个文件或文件夹。

请参阅

CopyFile 方法,CopyFolder 方法,Delete 方法,Move 方法,Open

AsTextStream 方法

CopyFile 方法

把一个或多个文件从一个地方复制到另一个地方。

应用于

FileSystemObject 对象

语法

object.CopyFile*source*,*destination*[,*overwrite*]

CopyFile 方法语法有如下几部分：

部分	描述
object	必需的。Object 始终是一个 FileSystemObject 的名字
source	必需的。指明一个或多个要被复制文件的字符串文件说明，它可以包括通配符
destination	必需的。指明 source 中的一个或多个文件要被复制到的接受端的字符串，不允许有通配符
overwrite	选项的。Boolean 值，它表示存在的文件是否被覆盖。如果是 True，文件将被覆盖；如果是 False，它们不被覆盖。缺省值是 True。注意如果 destination 具有只读属性设置，不论 overwrite 值如何，CopyFile 都将失败

说明

通配符只能用在 source 参数的最后一个路径部件。例如，你可以在下面请况使用通配符：

```
FileSystemObject.CopyFile"c:\mydocuments\letters\*.doc",
"c:\tempfolder\"
```

但下面情况不能使用：

```
FileSystemObject.CopyFile"c:\mydocuments\*\R1???97.xls",
"c:\tempfolder"
```

如果 source 包含通配符或 destination 以路径分隔符（\）为结尾，则认为 destination 是一个已存在文件夹，在其中复制相匹配的文件。否则认为 destination 是一个要创建文件的名字。不论是那种情况，当复制一个文件时，可能发生三种事件。

如果 destination 不存在，source 得到复制。这是通常的情况。

如果 destination 是一个已存在的文件，则当 overwrite 值为 False 时发生一个错误，否则，source 的复制文件将试图覆盖已存在文件。

如果 destination 是一个目录，发生一个错误。

如果使用通配符的 source 不能和任何文件匹配，同样产生一个错误。CopyFile 方法停止在它遇到的第一个错误上。不要试图回卷或撤消错误发生前所做的任何改变。

请参阅

Copy 方法,CopyFolder 方法,DeleteFile 方法,MoveFile 方法

CopyFolder 方法

从一个地方递归地复制一个文件夹到另一个地方。

应用于

FileSystemObject 对象

语法

object.**CopyFolder***source*,*destination*[,overwrite]

CopyFolder 方法语法有如下几部分：

部分	描述
object	必需的。始终为一个 FileSystemObject 的名字
source	必需的。指明一个或多个被复制文件夹的字符串文件夹说明，可以包括通配符
destination	必需的。指明 source 中被复制文件夹和子文件夹的接受端的字符串，不允许有通配
overwrite	选项的。Boolean 值，它表示已存在的文件夹是否被覆盖。如果为 True，文件被覆盖。如果为 False，文件不被覆盖。缺省值为 True

说明

通配符仅可用于 source 参数的最后一个路径部件。例如你可以在下面情况使用它：

```
FileSystemObject.CopyFolder"c:\mydocuments\letters\*","c
\tempfolder\"
```

但不能在下面情况使用它：

```
FileSystemObject.CopyFolder"c:\mydocuments\*\*","c:\temp folder\"
```

如果 source 包含通配符或 destination 以路径分隔符（\）为结尾，则认为 destination 是一个已存在的文件夹，在其中复制相匹配的文件夹和子文件夹。否则认为 destination 是一个要创建的文件夹的名字。不论何种情况，当复制一个文件夹时，可能发生四种事件。

如果 destination 不存在，source 文件夹和它所有的内容得到复制。这是通常的情况。

如果 destination 是一个已存在的文件，则发生一个错误。

如果 destination 是一个目录，它将尝试复制文件夹和它所有的内容。如果一个包含在 source 的文件已在 destination 中存在，当 overwrite 为 False 时发生一个错误，否则它将尝试覆盖这个文件。

如果 destination 是一个只读目录，当尝试去复制一个已存在的只读文件到此目录并且 overwrite 为 False 时，则发生一个错误。

如果 source 使用的通配符不能和任何文件夹匹配，也发生一个错误。

CopyFolder 方法停止在它遇到的第一个错误上。不要尝试回卷错误发生前所做的任何改变。

请参阅

Copy 方法,CopyFile 方法,CreateFolder 方法,DeleteFolder 方法,MoveFolder 方法

Cos 函数

返回一个 Double，指定一个角的余弦值。

语法

Cos(*number*)

必要的 *number* 参数是一 Double 或任何有效的数值表达式，表示一个以弧度为单位的角。

说明

Cos 函数的参数为一个角，并返回直角三角形两边的比值。该比值为角的邻边长度除以斜边长度之商。

结果的取值范围在-1 到 1 之间。

为了将角度转换成弧度，请将角度乘以 pi/180。为了将弧度转换成角度，请将弧度乘以 180/pi。

请参阅

Atn 函数,Sin 函数,Tan 函数,DerivedMath 函数

示例

本示例使用 Cos 函数计算一个角的余弦。

```
DimMyAngle,MySecant
MyAngle=1.3                    '定义角度（以“弧度”为单位）。
MySecant=1/Cos(MyAngle)        '利用余弦计算正割（sec()）。
```

Count 属性

返回 Long（长整数），包含集合中的对象数目。只读。

应用于

LindedWindows 集合,Windows 集合,Collection 对象,Licenses 集合

请参阅

Item 方法(VBA 外接对象模型),Add 方法,Item 方法,Remove 方法

示例

本示例使用 Collection 对象的 Count 属性，用来指示要将 MyClasses 集合对象中所有成员全部删除所需完成的次数（一次删除一个）。当集合对象以数字做索引值时，缺省方式是数值 1 代表第一个成员。因为集合对象的成员若被删除，便会自动重新编排索引值，所以下列代码中只在集合中连续删除第一个成员，最后即可删除所有成员。

```
DimNum,MyClasses
ForNum=1ToMyClasses.Count  '删除集合对象里的成员。
   MyClasses.Remove1    '集合对象索引值缺省方式为由 1 开始。
Next
```

Count 属性

返回一个 Long 型数，包含一个集合中成员的数目。此属性为只

读。

应用于

CodePanes 集合,LinkedWindows 集合,Properties 集合(外接程序对象模型)，References 集合,VBComponents 集合,VBProjects 集合,Windows 集合

示例

下列示例使用 Count 属性返回某个工程中所含的 VBComponent 对象的数量。

```
Debug.PrintApplication.VBE.VBProjects(1).VBComponents.Count
```

Count 属性

返回集合或 Dictionary 对象中的条目数。只读。

应用于

Dictionary 对象,Drives 集合,Files 集合,Folders 集合

语法

object.Count

object 总是“应用于”列表中某一项的名称。

说明

下面的代码举例说明了 Count 属性的使用方法：

```
Dima,d,i                    '创建一些变量
Setd=CreateObject("Scripting.Dictionary")
```

```
d.Add"a","Athens"                    '添加一些关键字和条目。
d.Add"b","Belgrade"
d.Add"c","Cairo"
a=d.Keys                  '获得关键字
Fori=0Tod.Count-1              '遍及数组
Printa(i)                    '打印关键字
Next
...
```

请参阅

CompareMode 属性,Item 属性,Key 属性

Count 属性（VB 集合）

返回集合中对象的数目。

应用于

ExportFormats 集合,Sections 集合(数据报表设计器),DataMembers 集合,Binding 集合对象,StdDataFormats 集合,PropertyPage 对象,Printer 对象,Printers 集合,Form 对象,Forms 集合,MDIForm 对象,DataObjectFiles 集合,Control 集合,userControl 对象,UserDocument 对象,Parameters 集合(VisualBasic),OLEObject 对象,SSTab 控件,Columns 集合,Splits 集合,Tab 对象,ListItem 对象,ListItem 集合,ColumnHeader 对象,ColumnHeader 集合,Node 对象,Nodes 集合

语法

object.**Count**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

能够与 For...Next 语句一起使用该属性对集合中的窗体或控件上执行操作。例如，下面的代码将一个窗体上的所有控件向右移动 0.5 英寸（ScaleMode 属性设置为 1 或 vbTwips）：

```
For I=0 To Form1.Controls.Count-1
   Form1.Controls(I).Left=Form1.Controls(I).Left+720
Next I
```

也可以使用这种结构快速地使窗体中的所有控件有效或失效。当与 IfTypeOf 语句一起使用时，可以经过所有的控件进行循环并进行改变，例如，仅对文本框的 Enabled 属性设置或仅对选项按钮的 BackColor 属性的设置进行处理。

请参阅

Count 属性(VB 集合),Count 属性(ActiveX 控件)

CountOfDeclarationLines 属性

返回一个 Long 型数，它包含代码模块的“声明”部分的代码行数。此属性为只读。

应用于

CodeModule 对象

请参阅

DeleteLines 方法,InsertLines 方法,Lines 方法,ReplaceLine 方法,CountOfLines 属性,CountOfVisibleLines 属性

示例

下列示例使用 CountOfDeclarationLines 属性返回指定的代码窗格中声明行的数目。

```
Debug.PrintApplication.VBE.CodePanes(2).CodeModule.CountOfDeclarationLines
```

CountOfLines 属性

返回一个 Long 型数，包含一个代码模块中代码的行数。此属性为只读。

应用于

CodeModule 对象

请参阅

DeleteLines 方法,InsertLines 方法,Lines 方法,replaceLine 方法,CountOfDeclarationLines 属性,CountOfVisibleOnesProerty

示例

下列示例使用 CountOfLines 属性返回指定代码窗格中的总行

数。

```
Application.VBE.CodePanes(2).CodeModule.CountOfLines
```

CountOfVisibleLines 属性

返回一个 Long 型数，它包含代码窗格中可见的代码行数。此属性为只读。

应用于

CodePane 对象

请参阅

DelectLines 方法,InsertLines 方法,Lines 方法,ReplaceLine 方法,CountOfLines 属性,CountOfDeclarationLines 属性

示例

下列示例使用 CountOfVisibleLines 属性返回指定的代码窗格中，根据窗格的高度，同时可见的行数。

```
Debug.PrintApplication.VBE.Codepanes(3).CountOfVisibleLines
```

CreateEmbed 方法

创建内嵌对象。不支持命名的参数。

应用于

OLE 容器控件

语法

object.**CreateEmbed***sourcedoc*,*class*

CreateEmbed 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
sourcedoc	必要的。将文档的文件名当作嵌入对象的模板来使用。如果不指定源文档，它必须是一个零长度字符串("")
class	可选的。是嵌入对象的类的名称。如果已经为 sourcedoc 指定了文件名，该参数忽略

说明

为了查看系统中可用的、有效的类名列表，先选择诸如 OLE 容器控件那样的控件，再在“属性”窗口选择 Class 属性，然后单击“生成器”按钮。

注意当使用 CreateEmbed 方法创建嵌入的对象时，不需要设置 Class 和 SourceDoc 属性。

当创建新对象时，与类名（如 Excel.exe）关联的应用程序，必须已在操作系统中正确地作了注册。（应用程序的安装程序应该将该应用程序正确地作了注册。）

请参阅

Class 属性,LOETypeAllowed 属性

CreateEventProc 方法

创建一个事件过程。

应用于

CodeModule 对象

语法

object.**CreateEventProc**(*eventname*,*objectname*)**AsLong**

CreateEventProc 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
eventname	必需的。一个字符串表达式，用来指定欲添加到模块的事件名称
objectname	必需的。一个字符串表达式，用来指定事件源的对象名称

说明

使用 CreateEventProc 方法来创建一个事件过程。例如，欲为名为 Command1 的 CommandButton 控件创建 Click 事件过程，可以用以下的代码，其中 CM 表示 CodeModule 类型的对象：

```
TextLocation=CM.CreateEventProc("Click","Command1")
```

CreateEventProc 方法可返回事件过程的开始行。如果参数引用到一个不存在的事件，CreateEventProc 将失败。

请参阅

AddFromString 方法,DeleteLines 方法,InsertLines 方法,ReplaceLine 方法,ProcBodyLine 属性,ProcCountLines 属性,ProcOfLine 属性,ProcStartLine 属性

示例

下列示例使用 CreateEventProc 方法来创建 Button_Click 过程。

```
Debug.PrintApplication.VBE.SelectVBComponents.CodeModule.CreateEventProc("Click","Button")
```

CreateFolder 方法

创建一个文件夹。

应用于

FileSystemObject 对象

语法

object.**CreateFolder**(*foldername*)

CreateFolder 方法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
foldername	必需的。字符串表达式，它标识创建的文件夹

说明

如果指定的文件夹已经存在，则发生一个错误。

请参阅

CopyFolder 方法,CreateTextFile 方法,DeleteFolder 方法,MoveFolder 方法

CreateLink 方法

从文件的内容中创建链接对象。不支持命名的参数。

应用于

OLE 容器控件

语法

object.**CreateLink***sourcedoc*,*sourceitem*

CreateLink 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
sourcedoc	必要的。对象将从该文件中创建
sourceitem	可选的。文件内的数据将在链接对象中被链接

说明

如果为这个方法的参数指定值，这些值将取代 SourceDoc 和 SourceItem 属性的设置值。当调用方法时，这些属性按参数值更新。

当使用方法创建对象时，OLE 容器控件就显示由 SourceDoc 属性指定的文件的图象。如果保存该对象，就只保存链接引用，因为 OLE 容器控件只包含数据的元文件图象，并没有实际的源数据。

当创建新对象时，与类名（如 Excel.exe）关联的应用程序必须已在操作系统中正确地作了注册。（应用程序的安装程序应该将该应用程序正确地作了注册。）

请参阅

SourceDoc 属性,SourceItem 属性,OLETypeAllowed 属性,CreateEmbed 方法

CreateObject 函数

创建并返回一个对 ActiveX 对象的引用。

语法

CreateObject(*class*,[*servername*])

CreateObject 函数的语法有如下部分：

部分	描述
class	必需的。Variant(String).要创建的应用程序名称和类
servername	可选的。Variant(String).要在其上创建对象的网络服务器名称

class 参数使用 appname.objecttype 这种语法，包括以下部分：

部分	描述
appname	必需的；Variant（字符串）。提供该对象的应用程序名。
objecttype	必需的；Variant（字符串）。待创建对象的类型或类。

说明

每个支持自动化的应用程序都至少提供一种对象类型。例如，一个字处理应用程序可能会提供 Application 对象，Document 对象，以及 Toolbar 对象。

要创建 ActiveX 对象，只需将 CreateObject 返回的对象赋给一个对象变量：

```
'声明一个对象变量来存放该对象
'的引用。DimasObject 采用后期绑定方式。
DimExcelSheetAsObject
SetExcelSheet=CreateObject("Excel.Sheet")
```

上述代码将启动该应用程序创建该对象，在本例中就是创建一个 MicrosoftExcel 电子数据表。对象创建后，就可以在代码中使用自定义的对象变量来引用该对象。在下面的示例中，可以使用对象变量 ExcelSheet 来访问新建对象的属性和方法，以及访问 MicrosoftExcel 的其它对象，包括应用程序对象和单元格集合。

```
'设置 Application 对象使 Excel 可见
ExcelSheet.Application.Visible=True
'在表格的第一个单元中写些文本
ExcelSheet.Cells(1,1).Value="ThisiscolumnA,row1"
'将该表格保存到 C:\test.doc 目录
ExcelSheet.SaveAs"C:\TEST.DOC"
'使用应用程序对象的 Quit 方法关闭 Excel。
ExcelSheet.Application.Quit
'释放该对象变量
```

SetExcelSheet=Nothing

使用 AsObject 子句声明对象变量，可以创建一个能包含任何类型对象引用的变量。不过，该变量访问对象是后期绑定的，也就是说，绑定在程序运行时才进行。要创建一个使用前期绑定方式的对象变量，也就是说，在程序编译时就完成绑定，则对象变量在声明时应指定类 ID。例如，可以声明并创建下列 MicrosoftExcel 引用：

```
DimxlAppAsExcel.Application
DimxlBookAsExcel.Workbook
DimxlSheetAsExcel.WorkSheet
SetxlApp=CreateObject("Excel.Application")
SetxlBook=xlApp.Workbooks.Add
SetxlSheet=xlBook.Worksheets(1)
```

前期绑定的变量引用可以提供更好的性能，但该变量只能存放声明中所指定的类的引用。

可以将 CreateObject 函数返回的对象传给一个参数为对象的函数。例如，下面的代码创建并传递了一个 Excel.Application 对象的引用：

```
CallMySub(CreateObject("Excel.Application"))
```

可以在一个远端连网的计算机上创建一个对象，方法是把计算机

的名称传递给 CreateObject 的 servername 参数。这个名称与共享名称的机器名部份相同：对于一个共享名称为"\\\\MyServer\\Public,"的 servername 参数是"MyServer"。
下面的代码返回在一个名为 MyServer 的远端计算机上运行的 Excel 实例的版本号：

```
DimxlAppAsObject
SetxlApp=CreateObject("Excel.Application","MyServer")
Debug.PrintxlApp.Version
```

如果远端服务器不存在或者不可用，则会发生一个运行时错误。
注意当该对象当前没有实例时，应使用 CreateObject。如果该对象已有实例在运行，就会启动一个新的实例，并创建一个指定类型的对象。要使用当前实例，或要启动该应用程序并加载一个文件，可以使用 GetObject 函数。
如果对象已登记为单个实例对象，则不管执行多少次 CreateObject，都只能创建该对象的一个实例。

CreateTextFile 方法

创建一个指定的文件名并且返回一个用于该文件读写的 TextStream 对象。

应用于

FileSystemObject 对象,Folder 对象

语法

object.**CreateTextFile**(*filename*[,*overwrite*[,*unicode*]])

CreateTextFile 方法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 或 Folder 对象的名字
filename	必需的。字符串表达式，它标识创建的文件
overwrite	可选的。Boolean 值，表示一个已存在文件是否可被覆盖。如果可被覆盖其值为 True，其值为 False 时不能覆盖。如果它被省略，则已存在文件不能覆盖
unicode	可选的。Boolean 值，表示文件是作为一个 Unicode 文件创建的还是作为一个 ASCII 文件创建的。如果作为一个 Unicode 文件创建，其值为 True，作为一个 ASCII 文件创建，其值为 False。如果省略的话，则认为是一个 ASCII 文件

说明

下面的代码举例说明如何使用 CreateTextFile 方法创建和打开文本文件。

```
SubCreateAfile
Setfs=CreateObject("Scripting.FileSystemObject")
Seta=fs.CreateTextFile("c:\testfile.txt",True)
a.WriteLine("Thisisatest.")
```

```
a.Close
EndSub
```

对于一个已经存在的 filename，如果 overwrite 参数是 False 或者没有提供，则发生一个错误。

请参阅

CopyFile 方法,DeleteFile 方法,OpenAsTextStream 方法,OpenTextFile 方法

CreateToolWindow 方法

创建一个包含指定 UserDocument 对象的新工具窗口。

语法

object.**CreateToolWindow**(*AddInInst*,*ProgID*,*Caption*,*GuidPosition*,*DocObj*)**AsWindow**

CreateToolWindow 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象。
addInInst	必需的。外接程序实例变量，表示在开发环境中的外接程序。
ProgID	必需的。串表示 UserDocument 对象的 progID。
caption	必需的。串包括窗口标题。
guidPosition	必需的。串包括窗口的唯一标识符。
DocObj	必需的。对象表示 UserDocument 对象。对象将被设置在这个函数的调用中。

CurDir 函数

返回一个 Variant(String)，用来代表当前的路径。

语法

CurDir[(*drive*)]

可选的 *drive* 参数是一个字符串表达式，它指定一个存在的驱动器。如果没有指定驱动器，或 *drive* 是零长度字符串(""),则 CurDir 会返回当前驱动器的路径。在 Macintosh 上，CurDir 忽略任何指定的 *drive*，并只简单地返回当前驱动器的路径。

请参阅

ChDir 语句,ChDrive 语句,Mkdir 语句,Rmdir 语句

示例

本示例使用 CurDir 函数来返回当前的路径在 Macintosh 中，使用 CurDir 可以省略驱动器名称。默认驱动器名称是 HD″并且路径部分由冒号取代反斜线隔开。同样，可以指定 Macintosh 的文件夹代替\Windows。

```
'假设 C 驱动器的当前路径为“ C:\WINDOWS\SYSTEM ”(在 MicrosoftWindows 中)。
'假设 D 驱动器的当前路径为“ D:\EXCEL ”。
'假设 C 为当前的驱动器。
DimMyPath
MyPath=CurDir  '返回“ C:\WINDOWS\SYSTEM ”。
MyPath=CurDir("C") '返回“ C:\WINDOWS\SYSTEM ”。
MyPath=CurDir("D") '返回“ D:\EXCEL ”。
```

CurrentX，CurrentY 属性

返回或设置下一次打印或绘图方法的水平(CurrentX)或垂直(CurrentY)坐标。设计时不可用。

应用于

PropertyPage 对象,UserControl 对象,userDocument 对象,Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**CurrentX**[=*x*]

object.**CurrentY**[=*y*]

CurrentX 和 CurrentY 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
x	确定水平坐标的数值
y	确定垂直坐标的数值

说明

坐标从对象的左上角开始测量。在对象的左边 CurrentX 属性值为 0，上边的 CurrentY 为 0。坐标以缇为单位表示，或以 ScaleHeight、ScaleWidth、ScaleLeft、ScaleTop 和 ScaleMode 属性定义的度量单位来表示。

用下面的图形方法时，CurrentX 和 CurrentY 的设置值按下述说明改变：

方法	设置 CurrentX,CurrentY 为：
Circle	对象的中心。
Cls	0，0。
EndDoc	0，0。
Line	线终点。
NewPage	0，0。
Print	下一个打印位置。
Pset	画出的点。

请参阅

Cls 方法,EndDoc 方法,NewPage 方法,Left,Top 属性,DrawMode 属性,DrawStyle 属性,ScaleHeight,ScaleWidth 属性,ScaleLeft,ScaleTop 属性

Cut 方法

从窗体中删除所选的控件。

应用于

SelectedVBControls 集合

语法

object.Cut

object 所在处表示对象表达式，其值是“应用于”列表中的对象。

CVErr 函数

返回 Error 子类型的 Variant，其中包含指定的错误号。

语法

CVErr(*errornumber*)

必要的 *errornumber* 参数可以是任何有效的错误号代码。

说明

可以在过程中，使用 CVErr 函数来创建用户自定义错误。例如，如果创建一个函数，它可以接受若干个参数，且正常返回一个字符串，则可以让函数来判断输入的参数，确认它们是在可接受的

范围内。如果不是的话，此函数将不会返回所要的字符串。在这种情况下，CVErr 可以返回一个错误号，并告知应该采取的行动。注意，Error 的隐式转换是不允许的，例如，不能直接把 CVErr 的返回值赋值给一个非 Variant 的变量。然而，可以对 CVErr 的返回值进行显式转换（使用 CInt、CDbl 等等），并赋值给适当的数据类型变量。

请参阅

转换函数，输入转换函数，IsError 函数，数据输入汇总

示例

本示例使用 CVErr 函数返回一 Variant 类型的值，其 VarType 为 vbError(10)。如果传进去的参数不是一个数字，用户自定义函数 CalculateDouble 将会返回错误信息。可用 CVErr 返回来自用户自定义过程的自定义错误，或改变处理运行时错误的方式。IsError 函数可用来测试返回值是否代表错误状态。

```
'调用 CalculateDouble，且传入一会产生错误的参数。
SubTest()
   Debug.PrintCalculateDouble("345.45robert")
EndSub
'定义 CalculateDouble 函数过程。
FunctionCalculateDouble(Number)
   IfIsNumeric(Number)Then
      CalculateDouble=Number*2 '返回结果。
```

```
    Else
        CalculateDouble=CVErr(2001) '返回一自定义错误码。
    EndIf
EndFunction
```

Data 属性

返回或设置内存对象或含有指定格式数据的图形设备接口(GDI)对象的句柄。在设计时不可用。

应用于

OLE 包容器控件

语法

object.**Data**[=*number*]

Data 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个指定句柄的长整数

说明

设置这个属性将数据发送到创建对象的应用程序。在使用 Data 属性之前，先设置 Format 属性，以指定内存对象或 GDI 对象中包含的数据的类型。

使用 ObjectAcceptFormats 和 ObjectGetFormats 属性可以得到对象的可接受格式的列表。

将这个属性设置为 0 就释放与句柄相关的内存。

提示 Automation 提供一个更容易、更可靠的方法，该方法发送数据和命令给对象，或从对象获取数据和命令。如果对象支持 Automation,可以通过 Object 属性或者使用 CreateObject 及 GetObject 函数访问该对象。

请参阅

DataText 属性，ObjectAcceptFormatsCount 属性，ObjectGetFormats 属性，ObjectGetFormatsCount 属性，Object 属性（OLE 容器），Format 属性

数据类型概述

下面这个表格显示所支持的数据类型，以及存储空间大小与范围。

数据类型	存储空间大小	范围
Byte	1 个字节	0 到 255
Boolean	2 个字节	True 或 False
Integer	2 个字节	-32,768 到 32,767
Long (长整型)	4 个字节	-2,147,483,648 到 2,147,483,647

Single (单精度浮点型)	4 个字节	负数时从-3.402823E38 到-1.401298E-45；正数时从1.401298E-45 到 3.402823E38
Double (双精度浮点型)	8 个字节	负数时从-1.79769313486232E308 到-4.94065645841247E-324；正数时从 4.94065645841247E-324 到 1.79769313486232E308
Currency (变比整型)	8 个字节	从-922,337,203,685,477.5808 到 922,337,203,685,477.5807
Decimal	14 个字节	没有小数点时为+/-79,228,162,514,264,337,593,543,950,335，而小数点右边有 28 位数时为+/-7.9228162514264337593543950335；最小的非零值为+/-0.0000000000000000000000000001
Date	8 个字节	100 年 1 月 1 日到 9999 年 12 月 31 日
Object	4 个字节	任何 Object 引用
String (变长)	10 字节加字符串长度	0 到大约 20 亿

String (定长)	字符串长度	1 到大约 65,400
Variant (数字)	16 个字节	任何数字值，最大可达 Double 的范围
Variant (字符)	22 个字节加字符串长度	与变长 String 有相同的范围
用户自定义（利用 Type）	所有元素所需数目	每个元素的范围与它本身的数据类型的范围相同。

注意任何数据类型的数组都需要 20 个字节的内存空间，加上每一数组维数占 4 个字节，再加上数据本身所占用的空间。数据所占用的内存空间可以用数据元数目乘上每个元素的大小加以计算。例如，以 4 个 2 字节之 Integer 数据元所组成的一维数组中的数据，占 8 个字节。这 8 个字节加上额外的 24 个字节，使得这个数组所需总内存空间为 32 个字节。

包含一数组的 Variant 比单独的一个数组需要多 12 个字节。

请参阅

TypeConversion 函数，BooleanData 类型，ByteData 类型，CurrencyData 类型，DateData 类型，DecimalData 类型，DoubleData 类型，IntegerData 类型，LongData 类型，ObjectData 类型，SingleData 类型，StringData 类型，User-DefinedData 类型，VariantData 类型，Deftype 语句，Type 语句，Int 函数

DataBinding 对象

DataBinding 对象代表部件的可绑定属性。

语法

DataBinding

说明

在“过程属性”对话框中，部件的每个属性都有标记为 Bindable 的 DataBinding 对象。

VisualBasicVersion4.0 支持这样的绑定，即一次只把控件的一个属性绑定到数据库中。以后的 VisualBasic 版本赋予将控件的多个属性绑定到数据库中的能力。这通常要与 User 控件并用。

如果想知道关于这方面的更详细信息，请参阅《部件工具指南》中的第九章“连编 ActiveX 控件”。

属性

DataMemeber 属性，DataFormat 属性，PropertyName 属性，IsBindable 属性，IsDataSource 属性，DataChanged 属性，DataField 属性，DataSource 属性

DataBindingBehavior 属性

设置一个值，它确定一个对象是否可以被绑定到一个数据源。仅在设计时可用。

应用于

Class

语法

object.**DataBindingBehavior** [=*number*]

DataBindingBehavior 属性语法有这些部分：

部分	描述
Object	一个对象表达式，其值是“应用于”列表中的一个对象
Number	一个整数，它指定数据绑定行为，如在设置值中所描述的

设置值

number 的设置值如下：

常数	设置值	描述
VbNone	0	(缺省)对象不能被绑定到数据源。
VbSimpleBound	1	对象可以使用简单绑定被绑定到数据源。
VbComplexBound	2	对象可以使用复杂绑定被绑定到数据源。

说明

如果您希望一个对象充当对象提供数据的使用者时，请使用 DataBindingBehavior 属性。一个数据使用者可以是简单绑定（绑定到单一字段）或者是复杂绑定（绑定到一个行集合）。

当 DataBindingBehavior 设置为 1(vbSimpleBound)时，PropertyChanged 事件和 CanPropertyChange 方法被添加到该对象的过程中。

DataBindings 集合

DataBindings 集合是一个扩展属性，它集中了提供给开发者和最终用户的可绑定的属性。

说明

在最终用户运行时，所有的可绑定属性都出现在 DataBindings 集合中。在开发者设计时（控件运行时），只有那些标记为“设计时在 DataBindings 集合中显示”的属性才能在访问“属性”窗口中的 DataBindings 属性时出现。

DataBindings 属性

返回 DataBindings 集合对象，该对象包含了有用的可绑定属性。

应用于

Slider 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，MonthView 控件，DataRepeater 控件，DataCombo 控件，DataList 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，MultimediaMCI 控件，MSChart 对象，MSHFlexGird 控件，MSFlexgrid 控件，SSTab 控件，RichTextBox 控件，Extender 对象

语法

object.**DataBindings**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

DataFormat 对象

在象 TextBox 控件这样的简单绑定对象中支持 DataFormat 属性。这个对象也要求支持用户创建的绑定控件中的数据格式化功能。

请参阅

DataFormat 属性，DataFormats 集合

DataFormat 属性

设置或返回 StdDataFormat 对象，一个绑定对象将附加到它。在设计时或运行时都可读写。

应用于

ImageCombo 控件，DateTimePicker 控件，MonthView 控件，RepeaterBinding 对象，DataGrid 控件，Column 对象，DataCombo 控件，DataList 控件，DBList 控件，CheckBox 控件（Lightweight），ComboBox 控件（Lightweight），MaskedEdit 控件，RichTextBox 控件，Function 控件（数据报表设计程序），TextBox 控件（数据报表设计程序），Binding 对象，Extender 对象，DataBinding 对象，CheckBox 控件，ComboBox 控件，Image 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

object.**DataFormat**=*formatobject*

DataFormat 属性语法有如下的部分：

部分	描述
Object	一个对象表达式，其值是“应用于”列表中的一个对象。
Formatobject	必需的。一个 StdDataFormat 对象。

说明

DataFormat 属性也能通过属性窗口的属性页设置。如果代码中的设置值与属性页的设置值不一样，那么属性页的设置值将应用于第一个取到的记录，之后，将使用代码中的设置值。

DataCombo 控件的 DataFormat 属性是 Extender 属性。因此在属性单上它总是可见的并且能在代码中设置。然而 DataCombo 控件仅仅格式化列表顶部的项。对于看到一个顶端项格式化、而仅给出一个未格式化项的列表供选择的终端用户来说，这可能是令人不安的。格式化的项也可能误导用户，使他认为项要以格式化的形式输入数据库。由于这些原因，建议在使用 DataCombo 控件时不要设置 DataFormat 属性。

请参阅

DataFormat 对象，DataFormats 属性，StdDataFormat 对象

DataFormats 集合

在象 DataGrid 控件这样的复杂绑定对象中支持 DataFormat 属性。这个对象也要求支持用户创建的绑定控件中的数据格式化功能。

请参阅

DataFormat 对象，DataFormats 属性

DataFormats 属性

设置或返回 StdDataFormat 对象，一个被绑定对象将附加到它。在设计时或运行时都可读写。

应用于

DataGrid 控件

语法

object.**DataFormats**

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

复杂绑定对象可以被绑定到多个 StdDataFormat 对象上。例如，DataGrid 的每一列都有一个绑定。DataFormats 属性允许访问绑定到一个控件上的多个 StdDataFormat 对象。

DataFormats 属性也能通过属性窗口的属性页设置。如果代码的设置值与属性页的设置值不一样，那么属性页的设置值将应用于第一个取到的记录，之后，将使用代码中的设置值。

请参阅

DataFormat 属性，DataFormats 集合，StdDataFormat 对象

DataMember 属性

从数据供应程序提供的几个数据成员中返回或设置一个特定的数据成员。

请参阅

DataReport 对象，BindingCollection 对象，Extender 对象，DataBinding 对象，CheckBox 控件，ComboBox 控件，Image 控件，Label 控件，ListBox 控件，PictureBox 控件，TextBox 控件，DataList 控件，RichTextBox 控件

语法

object.**DataMember**[=*string*]

部分	描述
Object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
String	数据成员名

说明

数据供应程序可以有多组数据供数据使用者选择以绑定到其上。每组数据都称作“数据成员”，并且用一个唯一的字符串标识。例如，当使用包含多个 Command 对象的 DataEnvironment 作为一个 DataSource 时，DataMember 将指定使用哪一个 Command 对象。

当使用一个类模块或一个用户控件作为数据源时，编程 GetDataMember 事件返回一个合适的数据成员。该事件的 DataMember 参数包含 DataMember 属性的值。通过查询该参数，

可以决定需要什么数据成员，并且通过 Data 参数传回合适的数据。

请参阅

GetDataMember 事件，DataSourceBehavior 属性

示例

本例把类模块用作数据源。当设定 DataSOURCE 和两个 Binding 对象执行的 DataMember 属性的代码时，类模块初始化事件发生，在那上事件中生成两个 ADO 记录集，并添加记录集的名字到 DataMembers 集合中。使用 GetDataMember 事件和它的参数返回数据到数据消耗者。要尝试实例，在 Project 菜单上，单击 Reference,并设定参数列 MicrosoftDataBindingCollection 和 MicrosoftActiveXDataObject.在 Project 菜单上，单击 AddClassModule。改变到 MyDataClass 的类名，并设定 DatSourceBehavior 属性到 vbDataSouce。然后拖动窗体上的两个 TextBox 控件，粘贴代码到 Form 对象的代码模块。

```
OptionExplicit
'声明对象变量，一个是名为 MyDataClass 的类模块，
'另外两个是 BindingCollection 对象，
'每个记录集一个。

PrivateclsDataAsNewMyDataClass        '类模块。
PrivatebndColProductsAsNewBindingCollection          '绑定集合。
```

```
PrivatebndColSuppliersAsNewBindingCollection
        '绑定集合。

PrivateSubForm_Load()
   '为每个绑定集合对象设置 DataSource 和 DataMember 属性。
   WithbndColProducts
        .DataMember="Products"
        Set.DataSource=clsData
        .AddText1,"Text","ProductName"
            '绑定到一个 TextBox。
   EndWith

   WithbndColSuppliers
        .DataMember="Suppliers"
        Set.DataSource=clsData
        .AddText2,"Text","CompanyName"   '绑定到一个 TextBox。
   EndWith

   '更改 Command1 的标题
   Command1.Caption="MoveNext"

EndSub
```

```
PrivateSubCommand1_Click()
    clsData.MoveNext
EndSub
```

粘贴下列代码到 MyDataClass 模块。为了查看 GetDataMember 事件，DataSourceBehavior 属性必须设定为 vbDataSource。运行项目。

```
OptionExplicit
'为 ADORecordset 和 Connection 对象声明对象变量。
PrivateWithEventsrsProductsAsADODB.Recordset
PrivateWithEventsrsSuppliersAsADODB.Recordset
PrivatecnNwindAsADODB.Connection

PrivateSubClass_Initialize()
    '添加字符串到 DataMembers 集合。
    WithDataMembers
        .Add"Products"
        .Add"Suppliers"
    EndWith

    '设置 Recordset 对象。
    SetrsProducts=NewADODB.Recordset
    SetrsSuppliers=NewADODB.Recordset
```

```
    SetcnNwind=NewADODB.Connection

    '设置 Connection 对象的参数。
    WithcnNwind
        '随 VisualBasic 发行的 Nwind.mdb 必须安装到
        '计算机上，否则本代码将失败。另外，改变路径
        '以找到计算机中的该文件。
        .Provider="Microsoft.Jet.OLEDB.3.51"
        .Open"C:\ProgramFiles\DevStudio\VB\Nwind.mdb"
    EndWith

    '打开 recordset 对象。
    rsSuppliers.Open"SELECT*FROMSuppliers",cnNwind,_
    adOpenStatic,adLockOptimistic
    rsProducts.Open"SELECT*FROMProducts",cnNwind,_
    adOpenStatic,adLockOptimistic

EndSub

'当一个数据使用者的 DataSource 属性设置时，GetDataMember 发生。
'在这种情况下，绑定的集合对象是该使用者。
PrivateSubClass_GetDataMember(DataMemberAsString,DataAsObject)
    SelectCaseDataMember
```

```
    Case"Products"
        SetData=rsProducts
    Case"Suppliers"
        SetData=rsSuppliers
    Case""
        '当没有指定数据成员时提供一个缺省的记录源。
        SetData=rsProducts
    EndSelect

EndSub

PublicFunctionMoveNext()
    IfrsProducts.EOFThen
        rsProducts.MoveFirst
    Else
        rsProducts.MoveNext
    EndIf
EndFunction

PrivateSubrsProducts_MoveComplete(ByValadReasonAs_
ADODB.EventReasonEnum,ByValpErrorAsADODB.Error,adStatusAs_
ADODB.EventStatusEnum,ByValpRecordsetAsADODB.Recordset)
```

```
    '保持两个记录集同步。第一个文本框显示
    '该产品的供货商。如果两个记录集的 SupplierID
    '是相等的，则不需要更改。否则，
    '移到第一条记录，并且测试 SupplierID。
    '本例只供演示，因为它并不是最有效的。

    IfrsSuppliers("SupplierID").Value=_
    pRecordset("SupplierID").ValueThenExitSub

    rsSuppliers.MoveFirst
    DoWhileNotrsSuppliers.EOF
        IfrsSuppliers("SupplierID").Value=_
        pRecordset("SupplierID").ValueThen
            ExitSub
        Else
            rsSuppliers.MoveNext
        EndIf
    Loop
EndSub
```

DataMembers 集合

一个数据源的数据成员集合。

语法

DataMembers

说明

数据供应程序可以有多组数据供数据使用者选择以绑定到其上。每组数据都称作一个“数据成员”，它可以是一个 ADORecordset、可以是提供 OLESimpleProvider（OSP）界面的 Class，也可以是用 VisualBasic 创建的 OLEDB 供应程序。在任何情况下，一个任意但唯一的字符串可以同数据成员关联起来，并且这个标识字符串正是使用 Add 方法加入到 DataMembers 集合中的。

当配置一个使用复杂数据源的数据使用者时，必须同时设置 DataSource 和 DataMember 两个属性以完全限定一个数据源。例如，如果配置一个使用 DataEnvironment 作为数据源的 TextBox 控件，必须设置它的 DataSource 属性为 DataEnvironment，并且设置它的 DataMember 属性为特定的 Command 对象。相反，当您创建自己的复杂数据源时，DataMembers 集合允许您的数据源为任何的数据使用者提供多组数据。

例如，通过设置 DataSourceBehavior 为 vbDataSource，可以创建一个配置为数据源的 UserControl。在 Initialize 事件中，使用 Add 方法把每一个数据成员的标识字符串添加到 DataMembers 集合。结果，当最终用户设置 DataSource 属性为您的用户控件，并且单击“属性”窗口的 DataMembers 时，添加到 DataMembers 集合的那些成员就会显现在下拉列表中。

类似地，当创建一个配置为数据源的 Class 时，在 Initialize 事件中，调用 Add 方法把标识字符串添加到 DataMembers 集合。然后，要配置使用该类的一个数据使用者，需要把它的 DataSource 属性设置为该类，并且把它的 DataMember 属性设置为 DataMembers 集合的一个成员。

当数据使用者的 DataSource 属性设置为一个类或者被配置为数据源的 Usercontrol 时，GetDataMember 事件发生。该事件包含一个 DataMember 参数，它传递 DataMember 属性的值。该事件还有一个 Data 参数，您可以使用它返回数据给使用者。换句话说，在 GetDataMember 事件中，查询 DataMember 值可以决定哪个数据成员已经被请求，并且设置该事件的 Data 参数为被请求的数据源（例如，ADO 记录集、OLEDB 供应程序或实现 OSP 接口的类）。数据成员也可以是一个空串。当为 GetDataMember 事件编程时，请一定通过设置缺省的数据成员来处理这种可能性。

方法

Add 方法（数据类型集合），Remove 方法，Clear 方法

请参阅

DataMembers 属性；《MicrosoftVisualBasic6.0 程序员指南》中第九章“创建和实施一个界面”的“用 Objects 进行程序设计”。

示例

本例把类模块用作数据源。当设定 DataSource 和两个 Binging 对象执行的 DataMember 属性的代码时，类模块和初始化事件发生，在那个事件中生成两个 ADO 记录集，并添加记录集的名字到

DataMembers 集合中。使用 GetDataMember 事件和这的参数返加数据到数据消耗者。

要尝试示例，在 Propject 菜单上，单击 Reference,并设定参数到 MicrosoftDataBindingCollection 和 MicrosoftActiveXData Class 的类名，并设定 DataSourceBehavior 属性到 vbDataSource。然后拖动窗体上的两个 TextBox 控件，粘贴代码到 Form 对象的代码模块。

```
OptionExplicit
'声明对象变量，一个是名为 MyDataClass 的类模块，
'另外两个是 BindingCollection 对象，
'每个记录集一个。

PrivateclsDataAsNewMyDataClass'类模块。
PrivatebndColProductsAsNewBindingCollection'绑定集合。
PrivatebndColSuppliersAsNewBindingCollection      '绑定集合。

PrivateSubForm_Load()
   '为每个绑定集合对象设置 DataSource 和 DataMember 属性。
   WithbndColProducts
        .DataMember="Products"
        Set.DataSource=clsData
        .AddText1,"Text","ProductName"          '绑定到一个 TextBox。
```

```
    EndWith

    WithbndColSuppliers
        .DataMember="Suppliers"
        Set.DataSource=clsData
        .AddText2,"Text","CompanyName"          '绑定到一个 TextBox。
    EndWith

    '更改 Command1 的标题
    Command1.Caption="MoveNext"

EndSub

PrivateSubCommand1_Click()
    clsData.MoveNext
EndSub
```

粘贴下列代码到 MyData(Class 模块。为了查看 GetDataMember 事件，DataSourceBehavior 属性必须设定为 vbDataSource。运行项目。

```
OptionExplicit
'为 ADORecordset 和 Connection 对象声明对象变量。
PrivateWithEventsrsProductsAsADODB.Recordset
PrivateWithEventsrsSuppliersAsADODB.Recordset
```

```
PrivatecnNwindAsADODB.Connection
PrivateSubClass_Initialize()
    '添加字符串到DataMembers 集合。
    WithDataMembers
        .Add"Products"
        .Add"Suppliers"
    EndWith

    '设置 Recordset 对象。
    SetrsProducts=NewADODB.Recordset
    SetrsSuppliers=NewADODB.Recordset
    SetcnNwind=NewADODB.Connection

    '设置 Connection 对象的参数。
    WithcnNwind
        '随 VisualBasic 发行的 Nwind.mdb 必须安装到
        '计算机上，否则本代码将失败。另外，改变路径
        '以找到计算机中的该文件。
        .Provider="Microsoft.Jet.OLEDB.3.51"
        .Open"C:\ProgramFiles\DevStudio\VB\Nwind.mdb"
    EndWith
```

```
    '打开 recordset 对象。
    rsSuppliers.Open"SELECT*FROMSuppliers",cnNwind,_
    adOpenStatic,adLockOptimistic
    rsProducts.Open"SELECT*FROMProducts",cnNwind,_
    adOpenStatic,adLockOptimistic

EndSub

'当一个数据使用者的 DataSource 属性设置时，GetDataMember 发生。
'在这种情况下，绑定的集合对象是该使用者。
PrivateSubClass_GetDataMember(DataMemberAsString,DataAsObject)
    SelectCaseDataMember
    Case"Products"
        SetData=rsProducts
    Case"Suppliers"
        SetData=rsSuppliers
    Case""
        '当没有指定数据成员时提供一个缺省的记录源。
        SetData=rsProducts
    EndSelect

EndSub
```

```
PublicFunctionMoveNext()
    IfrsProducts.EOFThen
        rsProducts.MoveFirst
    Else
        rsProducts.MoveNext
    EndIf
EndFunction

PrivateSubrsProducts_MoveComplete(ByValadReasonAs_
ADODB.EventReasonEnum,ByValpErrorAsADODB.Error,adStatusAs_
ADODB.EventStatusEnum,ByValpRecordsetAsADODB.Recordset)
    '保持两个记录集同步。第一个文本框显示
    '该产品的供货商。如果两个记录集的 SupplierID
    '是相等的，则不需要更改。否则，
    '移到第一条记录，并且测试 SupplierID。
    '本例只供演示，因为它并不是最有效的。

    IfrsSuppliers("SupplierID").Value=_
    pRecordset("SupplierID").ValueThenExitSub

    rsSuppliers.MoveFirst
    DoWhileNotrsSuppliers.EOF
```

```
        IfrsSuppliers("SupplierID").Value=_
        pRecordset("SupplierID").ValueThen
            ExitSub
        Else
            rsSuppliers.MoveNext
        EndIf
    Loop
EndSub
```

DataMembers 属性

返回一个对 DataMembers 集合的引用。

应用于

UserControl 对象

语法

object.**DataMembers**

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

数据供应程序可以有多组数据供数据使用者选择以绑定到其上。每组数据都称作“数据成员”，并且用一个唯一的字符串标识。DataMembers 集合包含数据使用者可以访问的所有数据成员名。

请参阅

DataMembers 集合

示例

本例使用 Add 方法向 DataMembers 集合添加两个数据成员。

```
OptionExplicit
'为 ADOrecordsets 集和 Connection 对象声明对象变量。
PrivateWithEventsrsProductsAsADODB.Recordset
PrivateWithEventsrsSuppliersAsADODB.Recordset
PrivatecnNwindAsADODB.Connection

PrivateSubClass_Initialize()
    '添加 DataMember 名到 DataMembers 集合。
    WithDataMembers
        .Add"Products"
        .Add"Suppliers"
    EndWith

    '设置 Recordset 对象。
    SetrsProducts=NewADODB.Recordset
    SetrsSuppliers=NewADODB.Recordset

    '有关打开记录集的代码没有显示出来。
```

EndSub

DataObject 对象

DataObject 对象是数据的容器，这些数据被从部件源传送到部件目标，它们存储的格式由使用 DataObject 对象的方法定义。

语法

DataObject

说明

映出 IDataObject 接口的 DataObject 对象，允许实现 OLE 拖放及剪贴板操作。

大部分部件支持人工 OLE 拖放事件，也有一些支持自动 OLE 拖放事件。

属性

Files 属性

方法

Clear 方法（DataObject 对象），GetData 方法（DataObject 对象），GetFormat 方法（DataObject 对象），SetData 方法（DataObject 对象）

DataObjectFiles 集合

字符串的集合，这里字符串类型为 DataObject 对象的 Files 属

性的类型。

语法

object.**DataObjectFiles**

其中 *object* 为对象表达式，其值为“应用于”列表中的一个对象。

说明

DataObjectFiles 集合为字符串集合，表示一组文件，它们是通过 GetData 方法选择的，或者是通过象 WindowsExplorer 这样的应用程序选择的。

虽然 DataObjectFiles 集合有它自己的方法和属性，还是应当用 DataObject 对象的 Files 属性来查看和操作 DataObjectFiles 集合的内容。

示例

这里有几个代码示例，显示如何用 Files 属性来查看和操作 DataObjectFiles 集合包含的数据（这里“数据”表示 DataObject 类型的对象）：

```
Debug.PrintData.Files(index)
ForEachvinData.Files
Debug.Printv
Nextv
Data.Files.Add"autoexec.bat"
Data.Files.Removeindex
Data.Files.Clear
```

```
Fori=1toData.Files.Count
Debug.printData.Files(i)
Nexti
```

注意只有当 DataObject 对象中的数据为 vbCFFiles 格式时，此集合才被 Files 属性所应用。

属性

Count 属性（VB 集合）

方法

Add 方法（DataObjectFiles 集合），Clear 方法，Remove 方法

DataReport 对象

DataReport 对象是一个可编程对象，代表数据报表设计器（Data Reportdesigner）。

语法

DataReport

说明

DataReport 使用数据库中的记录生成报表。要使用它：

1. 配置一个数据源，例如 Microsoft 数据环境，以访问数据库。
2. 设定 DataReport 对象的 DataSource 属性为数据源。
3. 设定 DataReport 对象的 DataMember 属性为数据成员。
4. 右键单击设计器，并单击“检索结构”。

5.向相应的节添加相应的控件。

6.为每一个控件设定 DataMember 和 DataField 属性。

7.运行时，使用 Show 方法显示数据报表。

使用 DataReport 对象通过更改每一 Section 对象的布局，来编程改变数据报表的外观和行为。

数据报表设计器还有使用 ExportReport 方法导出报表的功能。这一方法允许从 ExportFormats 集合中指定一个 ExportFormat 对象，以用作报表的模板。

属性

BottomMargin,TopMargin 属性，ExportFormats 属性，GridX，GridY 属性，LeftMargin,RightMargin 属性，ReportWidth 属性，Sections 属性，Title 属性，AsyncCount 属性，DataMember 属性，Font 属性，DataSource 属性

方法

ExportReport 方法，PrintReport 方法，Refresh 方法

事件

AsyncPropgress 事件，Error 事件（数据报表设计程序），ProcessingTimeout 事件，QuerryClose 事件，Activate,Deactivate 事件，Resize 事件，Initialize,Terminate 事件

请参阅

ExportFormat 对象，Error 对象（数据报表设计程序），Section 对象（数据报表设计程序）Add 方法（ExportFormats 集合）

DataSourceBehavior 属性

设置一个值，它确定一个对象是否能充当其它对象的数据源。只能在设计时设置。

应用于

Class

语法

object.**DataSourceBehavior** [=*number*]

DataSourceBehavior 属性语法有这些部分：

部分	描述
Object	一个对象表达式，其值是“应用于”列表中的一个对象
Number	一个整数，它指定数据源行为，如在设置值中所

设置值

number 的设置值如下：

常数	设置值	描述
VbNone	0	(缺省)该对象不能作为数据源。
VbDataSource	1	该对象可以作为数据源。
VbOLEDBProvider	2	该对象可以作为数据源和一个 OLEDB 供应程序。仅对公共类可用。

说明

当您希望某个对象作为其它对象的数据源时，请使用 DataSourceBehavior 属性。当 DataSourceBehavior 被设置成 1(vbData

Source)或 2(vbOLEDBProvider)时,GetDataMember 事件被添加到该对象的过程中。

当对象是一个公共类时,DataSourceBehavior 只可以被设置成 2(vbOLEDBProvider)。在这种情况下,OnDataConnection 事件也被添加到该对象的过程中。

请参阅

DataMember 属性

DataText 属性

从指定的对象返回字符串,或为指定的对象设置字符串。

应用于

OLE 包容器控件

语法

object.**DataText**[=*string*]

DataText 属性的语法包含下面部分:

部分	描述
Object	对象表达式,其值是“应用于”列表中的一个对象。
String	一个指定字符串的字符串表达式。

说明

为了将字符串发送给对象,先将 Format 属性设置为对象支持的格式。使用 ObjectGetFormats 和 ObjectAcceptFormats 属性得到对象所支

持格式的列表。

当从对象得到数据时，DataText 属性返回从该对象发送的字符串，它以第一个 null 字符作为结束。

只要内存许可，DataText 字符串可以很大。

提示 Automation 提供一个更容易、更可靠的方法，该方法发送数据和命令给对象，或从对象获取数据和命令。如果对象支持 Automation，可以通过 Object 属性或者使用 CreateObject 及 GetObject 函数访问该对象。

请参阅

Data 属性，ObjectAcceptFormatsCount 属性，ObjectGetFormats 属性，ObjectGetFormatsCount 属性，Object 属性（OLE 包容器），Format 属性

示例

本例是将数据传送给 MicrosoftGraph 的应用程序，因此系统必须安装有 MSGraph 才能运行本例，（大多数 MicrosoftOffice 组件均带有 MSGraph）。建立一个约占屏幕二分之一大小的窗体，在该窗体左上角有一个 CommandButton 控件（Command1）并且在 CommandButton 的下面放有一个 OLE 容器控件（OLE1）。

在窗体上放置 OLE 容器控件时，会出现“插入对象”对话框。选择“取消”并按 F5 键可运行本例。

```
PrivateSubCommand1_Click()
DimMsg,NL,TB                    '变量声明。
   TB=Chr(9)                    'Tab 字符。
```

```
    NL=Chr(10)                    '换行字符。
    '建立数据以替换默认图表的数据。
    Msg=TB+"Drew"&TB&"Teresa"&TB&"Bob"
    Msg=Msg+NL&"Eric"&TB&"1"&TB&"2"&TB&"3"
    Msg=Msg+NL&"Ted"&TB&"11"&TB&"22"&TB&"33"
    Msg=Msg+NL&"Arthur"&TB&"21"&TB&"32"&TB&"23"
    '使用 DataText 属性传送数据。
    '激活 MSGRAPH 为隐藏状态。
Ole1.DoVerb-3
    IfOle1.AppIsRunningThen
        Ole1.DataText=Msg
        '更新该对象。
        Ole1.Update
    Else
        MsgBox"Graphisn'tactive."
    EndIf
EndSub
SubForm_Load()
    Ole1.Format="CF_TEXT"     '置文件格式为文本方式。
    Ole1.SizeMode=2  '自动调整大小。
    Ole1.CreateEmbed"","MSGRAPH"
EndSub
```

Date 数据类型

Date 变量存储为 IEEE64 位（8 个字节）浮点数值形式，其可以表示的日期范围从 100 年 1 月 1 日到 9999 年 12 月 31 日，而时间可以从 0:00:00 到 23:59:59。任何可辨认的文本日期都可以赋值给 Date 变量。日期文字须以数字符号(#)扩起来，例如，#January1,1993#或#1Jan93#。

Date 变量会根据计算机中的短日期格式来显示。时间则根据计算机的时间格式（12 或 24 小时制）来显示。

当其他的数值类型要转换为 Date 型时，小数点左边的值表示日期信息，而小数点右边的值则表示时间。午夜为 0 而中午为 0.5。负整数表示 1899 年 12 月 30 日之前的日期。

请参阅

TypeConversion 函数，DataType 总结，DoubleData 类型，VariantData 类型，Deftype 语句

Date 函数

返回包含系统日期的 Variant(Date)。

语法

Date

说明

为了设置系统日期，请使用 Date 语句。

请参阅

Date 语句，Now 函数，Time 函数，Time 语句，Format 函数

示例

本示例使用 Date 函数返回系统当前的日期。

```
DimMyDate
MyDate=Date   'MyDate 的值为系统当前的日期。
```

Date 语句

设置当前系统日期。

语法

Date=*date*

对于运行 MicrosoftWindows95 的系统，要设置的 date 必须介于 1980 年 1 月 1 日与 2099 年 12 月 31 日之间。对于运行 MicrosoftWindowsNT 的系统，date 必须介于公元 1980 年 1 月 1 日到 2079 年 12 月 31 日之间。

对于 Macintosh，date 值必须是介于 1904 年 1 月 1 日到 2040 年 2 月 5 日之间的日期。

请参阅

Date 函数，Time 函数，Time 语句

示例

本示例使用 Date 语句来设置系统日期。在开发环境中，日期原义会根据系统的地区设置，以短式日期格式显示。

```
DimMyDate
MyDate=#February12,1985#    '指定某个日期。
Date=MyDate     '改变系统日期。
```

DateAdd 函数

返回包含一个日期的 Variant(Date)**，这一日期还加上了一段时间间隔。**

语法

DateAdd(*interval*,*number*,*date*)

DateAdd **函数语法中有下列命名参数：**

部分	**描述**
Interval	**必需的。字符串表达式，是所要加上去的时间间隔。**
Number	**必需的。数值表达式，是要加上的时间间隔的数目。其数值可以为正数（得到未来的日期），也可以为负数（得到过去的日期）。**
Date	**必要。**Variant(Date)**或表示日期的文字，这一日期还加上了时间间隔。**

设置

interval **参数具有以下设定值：**

设置	描述
Yyyy	年
Q	季
M	月
Y	一年的日数
D	日
W	一周的日数
Ww	周
H	时
N	分钟
S	秒

说明

可以使用 DateAdd 函数对日期加上或减去指定的时间间隔。例如，可以用 DateAdd 来计算距今天为三十天的日期；或者计算距现在为 45 分钟的时间。

为了对 date 加上“日”，可以使用“一年的日数”(“y”)，“日”(”d”)或“一周的日数”(”w”)。

DateAdd 函数将不返回有效日期。在以下实例中将 1 月 31 日加上一个月：

```
DateAdd(m,1,31-Jan-95)
```

上例中，DateAdd 返回 1995 年 2 月 28 日，而不是 1995 年 2 月 31 日。如果 date 是 1996 年 1 月 31 日，则由于 1996 年是闰年，返回值是 1996 年 2 月 29 日。

如果计算的日期超前 100 年（减去的年度超过 date 中的年份），就会导致错误发生。

如果 number 不是一个 Long 值，则在计算时取最接近的整数值来计算。

注意 DateAdd 返回值的格式由 ControlPanel 设置决定，而不是由传递到 date 参数的格式决定。

请参阅

DateDiff 函数，DatePart 函数，Day 函数，Now 函数，Weekday 函数，Year 函数，Format 函数

示例

本示例先取得一个日期，再用 DateAdd 函数显示未来数月后的日期。

```
DimFirstDateAsDate '声明变量。
DimIntervalTypeAsString
DimNumberAsInteger
DimMsg
IntervalType="m""m"指定以“月份”作为间隔。
FirstDate=InputBox("Enteradate")
Number=InputBox("Enternumberofmonthstoadd")
Msg="Newdate:"&DateAdd(IntervalType,Number,FirstDate)
```

MsgBoxMsg

DateCreated 属性

返回指定文件或文件夹的创建日期和时间。只读。

应用于

File 对象，Folder 对象

语法

object.**DateCreated**

object 总是一个 file 或 Folder 对象。

说明

下面的代码用一个文件举例说明了 DateCreated 属性的用法：

```
SubShowFileInfo(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFile(filespec)
s="Created:"&f.DateCreated
MsgBoxs
EndSub
```

请参阅

Attributes 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性，ParentFolder 属性，Path 属性，ShortName 属性，ShortPath 属

性，Size 属性，SubFolders 属性，Type 属性

DateDiff 函数

返回 Variant(Long)的值，表示两个指定日期间的时间间隔数目。

语法

DateDiff(*interval*,*date1*,*date2*[,*firstdayofweek*[,*firstweekofyear*]])

DateDiff 函数语法中有下列命名参数：

部分	描述
interval	必需的。字符串表达式，表示用来计算 date1 和 date2 的时间差的时间间隔
date1□*date2*	必需的。Variant(Date)。计算中要用到的两个日期。
Firstdayofweek	可选的。指定一个星期的第一天的常数。如果未予指定，则以星期日为第一天。
Firstweekofyear	可选的。指定一年的第一周的常数。如果未予指定，则以包含 1 月 1 日的星期为第一周。

设置

interval 参数的设定值如下：

设置	描述
Yyyy	年
q	季
m	月
Y	一年的日数
d	日
w	一周的日数
ww	周
h	时
n	分钟
s	秒

firstdayofweek 参数的设定值如下：

常数	值	描述
VbUseSystem	0	使用 NLSAPI 设置。
vbSunday	1	星期日（缺省值）
vbMonday	2	星期一
vbTuesday	3	星期二
vbWednesday	4	星期三
vbThursday	5	星期四
vbFriday	6	星期五
vbSaturday	7	星期六

常数	值	描述
vbUseSystem	0	用 NLSAPI 设置。
vbFirstJan1	1	从包含 1 月 1 日的星期开始（缺省值）。
vbFirstFourDays	2	从第一个其大半个星期在新的一年的一周开始。
vbFirstFullWeek	3	从第一个无跨年度的星期开始。

说明

DateDiff 函数可用来决定两个日期之间所指定的时间间隔数目。例如，可以使用 DateDiff 来计算两个日期之间相隔几日，或计算从今天起到年底还有多少个星期。

为了计算 date1 与 date2 相差的日数，可以使用“一年的日数”(y)或“日”(d)。当 interval 是“一周的日数”(w)时，DateDiff 返回两日期间的周数。如果 date1 是星期一，DateDiff 计算到 date2 为止的星期一的个数。这个数包含 date2 但不包含 date1。不过，如果 interval 是“周”(ww)，则 DateDiff 函数返回两日期间的“日历周”数。由计算 date1 与 date2 之间星期日的个数而得。如果 date2 刚好是星期日，则 date2 也会被加进 DateDiff 的计数结果中；但不论 date1 是否为星期日，都不将它算进去。

如果 date1 比 date2 来得晚，则 DateDiff 函数的返回值为负数。

firstdayofweek 参数会影响使用时间间隔符号“W”或“WW”计算的结果。

如果 date1 或 date2 是日期文字，则指定的年份成为该日期的固定部分。但是，如果 date1 或 date2 用双引号("")括起来，且年

份略而不提，则在每次计算表达式 date1 或 date2 时，当前年份都会插入到代码之中。这样就可以书写适用于不同年份的程序代码。

在计算 12 月 31 日和来年的 1 月 1 日的年份差时，DateDiff 返回 1 表示相差一个年份，虽然实际上只相差一天而已。

请参阅

DateAdd 函数，DatePart 函数，Day 函数，Now 函数，Weekday 函数，Year 函数，Format 函数

示例

本示例使用 DateDiff 函数来显示某个日期与今日相差几天。

```
DimTheDateAsDate    '声明变量。
DimMsg
TheDate=InputBox("Enteradate")
Msg="Daysfromtoday:"&DateDiff("d",Now,TheDate)
MsgBoxMsg
```

DateLastAccessed 属性

返回最后一次访问指定文件或文件夹的日期和时间。只读。

应用于

File 对象，Folder 对象

语法

object.**DateLastAccessed**

object 总是一个 *file* 或 *Folder* 对象。

说明

下面的代码用一个文件举例说明了 DateLastAccessed 属性的用法：

```
SubShowFileAccessInfo(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFile(filespec)
s=UCase(filespec)&vbCrLf
s=s&"Created:"&f.DateCreated&vbCrLf
s=s&"LastAccessed:"&f.DateLastAccessed&vbCrLf
s=s&"LastModified:"&f.DateLastModified
MsgBoxs,0,"FileAccessInfo"
EndSub
```

重点这种方法由其下一级操作系统决定其行为。如果操作系统不支持提供时间信息，则没有返回信息。

请参阅

Attributes 属性，DateCreated 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性（FileSystemObject 对象），ParentFolder 属性，Path 属性(FileSystemObjedct 对象)，ShortName 属性，ShortPath 属性，Size 属性（FileSystemObject 对象），SubFolders 属性，Type 属性

DateLastModified 属性

返回最后一次修改指定文件或文件夹的日期和时间。只读。

应用于

File 对象，Folder 对象

语法

object.**DateLastModified**

object 总是一个 file 或 Folder 对象。

说明

下面的代码用一个文件举例说明了 DateLastModified 属性的用法：

```
SubShowFileAccessInfo(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFile(filespec)
s=UCase(filespec)&vbCrLf
s=s&"Created:"&f.DateCreated&vbCrLf
s=s&"LastAccessed:"&f.DateLastAccessed&vbCrLf
s=s&"LastModified:"&f.DateLastModified
MsgBoxs,0,"FileAccessInfo"
EndSub
```

请参阅

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，

Drive 属性，Files 属性 IsRootFolder 属性，Name 属性（FileSystemObject 对象），ParentFolder 属性，Path 属性（FileSystemObject 对象），ShortName 属性，ShortPath 属性，Size 属性（FileSystemObject 对象），SubFolders 属性，Type 属性

DatePart 函数

返回一个包含已知日期的指定时间部分的 Variant (Integer)。

语法

DatePart(*interval*,*date*[,*firstdayofweek*[,*firstweekofyear*]])

DatePart 函数语法中有下列命名参数：

部分	描述
interval	必需的。字符串表达式，是要返回的时间间隔。
date	必需的。要计算的 Variant(Date)值。
firstdayofweek	可选的。指定一个星期的第一天的常数。如果未予指定，则以星期日为第一天。
firstweekofyear	可选的。指定一年第一周的常数。如果未予指定，则以包含 1 月 1 日的星期为第一周。

设置

interval 参数的设定值如下：

设置	描述
yyyy	年
q	季
m	月
y	一年的日数
d	日
w	一周的日数
ww	周
h	时
n	分钟
s	秒

firstdayofweek 参数的设定值如下：

常数	值	描述
vbUseSystem	0	使用 NLSAPI 设置。
vbSunday	1	星期日（缺省值）
vbMonday	2	星期一
vbTuesday	3	星期二
vbWednesday	4	星期三
vbThursday	5	星期四
vbFriday	6	星期五
vbSaturday	7	星期六

firstweekofyear 参数的设定值如下：

常数	值	描述
vbUseSystem	0	使用 NLSAPI 设置。
vbFirstJan1	1	从包含 1 月 1 日的星期开始（缺省值）。
vbFirstFourDays	2	从第一个其大半个星期在新的一年的一周开始。
vbFirstFullWeek	3	从第一个无跨年度的星期开始。

说明

DatePart 函数可以用来计算日期并返回指定的时间间隔。例如，可以使用 DatePart 计算某个日期是星期几或目前为几点钟。

firstdayofweek 参数会影响使用时间间隔符号“W”或“WW”计算的结果。

如果 date 是日期文字，则指定的年份成为该日期的固定部分。但是，如果 date 用双引号("")括起来，且年份略而不提，则在每次计算 date 表达式时，当前年份都会插入到代码之中。这样就可以书写适用于不同年份的程序代码。

请参阅

DateAdd 函数，DateDiff 函数，Day 函数，Now 函数，Weekday 函数，Year 函数，Format 函数

示例

本示例使用 DateDiff 函数来显示某个日期与今日相差几天。

```
DimTheDateAsDate    '声明变量。
DimMsg
TheDate=InputBox("Enteradate:")
Msg="Qudeter:"&Datepart(“q”,ThePase)
MsgBoxMsg
```

DateSerial 函数

返回包含指定的年、月、日的 Variant (Date)。

语法

DateSerial(*year*,*month*,*day*)

DateSerial 函数语法有下列的命名参数：

部分	描述
year	必需的；Integer。从 100 到 9999 间的整数，或一数值表达式。
month	必需的；Integer。任何数值表达式。
day	必需的；Integer。任何数值表达式。

说明

为了指定某个日期，如 1991 年 12 月 31 日，DateSerial 函数中的每个参数的取值范围应该是可接受的；即，日的取值范围应在 1-31 之间，而月的取值范围应在 1-12 之间。但是，当一个数值表达式表示某日之前或其后的年、月、日数时，也可以为每个使用这个数值表达式的参数指定相对日期。

以下示例中使用了数值表达式代替绝对日期。这里，DateSerial 函数返回 1990 年 8 月 1 日的十年(1990-10)零两个月(8-2)又一天(1-1)之前的日期；换句话说，就是 1980 年 5 月 31 日。

```
DateSerial(1990-10,8-2,1-1)
```

year 参数的数值若介于 0 与 29 之间，则将其解释为 2000–2029 年，若介于 30 和 99 之间则解释为 1930–1999 年。而对所有其它 year 参数，则请用四位数值表示（如 1800）。

当任何一个参数的取值超出可接受的范围时，它会适时进位到下一个较大的时间单位。例如，如果指定了 35 天，则这个天数被解释成一个月加上多出来的日数，多出来的日数将由其年份与月

份来决定。如果一个参数值超出-32，768 到 32，767 的范围，就会导致错误发生。

请参阅

Date 函数，Date 语句，DateValue 函数，Day 函数，Month 函数，Now 函数，TimeSerial 函数，TimeValue 函数，Weekday 函数，Year 函数

示例

本示例使用 DateSerial 函数来将指定的年月日转换为 Date 类型的表达式。

```
DimMyDate
'MyDate 的值为 February12，1969，Date 类型。
MyDate=DateSerial(1969,2,12)    '返回日期。
```

DateValue 函数

返回一个 Variant(Date)。

语法

DateValue(*date*)

必要的 date 参数，date 通常是字符串表达式，表示从 100 年 1 月 1 日到 9999 年 12 月 31 日之间的一个日期。但是，date 也可以是任何表达式，其所代表的日期、时间在上述范围内。

说明

如果 date 是一个字符串，且其内容只有数字以及分隔数字的日

期分隔符，则 DateValue 就会根据系统中指定的短日期格式来识别月、日、年的顺序。DateValue 也识别明确的英文月份名称，全名或缩写均可。例如，除了 12/30/1991 和 12/30/91 之外，DateValue 也识别 December30,1991 和 Dec30,1991。

如果 date 中略去了年这一部分，DateValue 就会使用由计算机系统日期设置的当前年份。

如果 date 参数包含时间信息，则 DateValue 不会返回它。但是，如果 date 包含无效时间信息（如 89:98），则会导致错误发生。

请参阅

Date 函数，Date 语句，DateSerial 函数，Day 函数，Month 函数，Now 函数，TimeSerial 函数，TimeValue 函数，Weekday 函数

示例

本示例使用 DateValue 函数将字符串转换为日期。也可以使用日期原义直接给 Variant 或 Date 类型的变量赋值日期，例如 MyDate=#2/12/69#。

```
DimMyDate
MyDate=DateValue("February12,1969")      '返回日期。
```

Day 函数

返回一个 Variant(Integer)，其值为 1 到 31 之间的整数，表示一个月中的某一日。

语法

Day(*date*)

必要的 date 参数，可以是任何能够表示日期的 Variant、数值表达式、字符串表达式或它们的组合。如果 date 包含 Null，则返回 Null。

请参阅

Date 函数，Date 语句，Hour 函数，Minute 函数，Month 函数，Now 函数，Second 函数，Weekday 函数，Year 函数

示例

本示例使用 Day 函数将指定的日期转换为该月的第几天。在开发环境中，日期原义会根据系统的地区设置，以短式日期格式显示。。

```
DimMyDate,MyDay
MyDate=#February12,1969#    '指定一日期。
MyDay=Day(MyDate)   'MyDay 的值为 12。
```

DblClick 事件

当在一个对象上按下和释放鼠标按钮并再次按下和释放鼠标按钮时，该事件发生。

对于窗体而言，当双击被禁用的控件或窗体的空白区域时，DblClick 事件发生。对于控件而言，DblClick 事件在以下情形下发生：

用鼠标左键双击控件。

双击 Style 属性设置为 1(Simple)的 ComboBox 控件中的项目，或者在 FileListBox、ListBox、DataCombo 或 DataList 控件中的项目。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，ComboBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 容器控件，Animatio 控件，DBCombo 控件，DBList 控件

语法

PrivateSubForm_DblClick()

PrivateSub*object*_**DblClick**(*index***AsInteger**)

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
index	如果控件在控件数组内，则这个 index 值就用来标识该控件

说明

如果控件在一个控件数组内，则这个 Index 参数唯一地标识这个控件。可以使用 DblClick 事件过程执行一个隐式操作，如双击图标来打开一个窗口或文档。也可用这类过程执行单一操作的多个步骤，例如用双击在列表框中选定一项并关闭对话框。

要在 VisualBasic 中产生这类快捷效果，可以使用带有缺省按钮

的列表框的或叠层排列文件列表框的 DblClick 事件过程；所谓缺省按钮就是 Default 属性设置为 True 的 CommandButton 控件。作为列表框 DblClick 事件过程的一部分，只需简单调用缺省按钮的 Click 事件。

对于那些接收 Mouse 事件的对象，事件按这样的次序发生：MouseDown,MouseUp,Click,DblClick 和 MouseUp。

如果 DblClick 在系统双击时间限制内没有出现，则对象识别另一个 Click 事件。双击时间限制可以变化，因为用户可在控制面板设置双击速度。当与这些相关事件过程打交道时，必须确保它们的活动不发生冲突。不接受 DblClick 事件的控件可能接受两次单击而不是 DblClick 事件。

调试事件时，不要使用 MsgBox 语句显示事件何时发生，因为这样做将会干扰许多事件的正常功能。(例如，Click 事件中的 MsgBox 将会阻止 DblClick 事件的发生。)而应该用 Debug.Print 来显示事件发生的顺序。

注意要想区别鼠标的左、右、中按钮，使用 MouseDown 和 MouseUp 事件。

请参阅

Click 事件，MouseDown,MouseUp 事件，Click 事件 (ActiveX 控件)

示例

无论是通过单击 CommandButton 控件还是双击列表项，本范例将

显示 TextBox 控件中选定的列表项。要试用此例，将代码粘贴包含 ListBox 控件、TextBox 控件和 CommandButton 控件的 Form 对象的声明部分。然后运行此例并单击 CommandButton 控件或双击在 ListBox 控件中的一项。

```
PrivateSubForm_Load()
   List1.AddItem"John" '添加列表框项。
   List1.AddItem"Paul"
   List1.AddItem"George"
   List1.AddItem"Ringo"
EndSub

PrivateSubList1_DblClick()
   Command1.Value=True '触发 Click 事件。
EndSub

PrivateSubCommand1_Click()
   Text1.Text=List1.Text    '显示选定。
EndSub
```

DDB 函数

返回一个 Double，指定一笔资产在一特定期间内的折旧。可使用双下落收复平衡方法或其它指定的方法进行计算。

语法

DDB(*cost*,*salvage*,*life*,*period*[,*factor*])

DDB 函数具有下列命名参数：

部分	描述
cost	必需的。Double 指定资产的初始成本
salvage	必需的。Double.指定使用年限结束时的资产价值
life	必需的。Double 指定资产的可用年限
period	必需的。Double 指定计算资产折旧所用的那一期间
factor	可选的。Variant 指定收复平衡下落时的速度。如果省略的话，2（双下落方法）为缺省值

说明

双下落收复平衡方法用加速利率法计算折旧。在第一段时期，折旧为最高，而在接下来的期间内降低。

life 和 period 参数必须用相同的单位表示。例如，如果 life 用月份表示，则 period 也必须用月份表示。所有参数都必须是正值。

DDB 函数使用下列公式计算在一定时期后的折旧：

折旧/period=((cost–salvage)*factor)/life

请参阅

FV 函数，Ipmt 函数，IRR 函数，MIRR 函数，Nper 函数，NPV 函数，Pmt 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

本示例使用 DDB 函数来计算某项资产在一特定期间内的折旧。计算时需给定资产的初始成本（InitCost），使用年限退出时所余的价值（SalvageVal），以“年”为单位之可用的生命长度（LifeTime），以及所要计算折旧的期间年限（Depr）。

```
DimFmt,InitCost,SalvageVal,MonthLife,LifeTime,DepYear,Depr
ConstYRMOS=12  '一年之中的月份数。
Fmt="###,##0.00"
InitCost=InputBox("What'stheinitialcostoftheasset?")
SalvageVal=InputBox("Entertheasset'svalueatendofitslife.")
MonthLife=InputBox("What'stheasset'susefullifeinmonths?")
DoWhileMonthLife<YRMOS '确保计算期间大于等于一年。
   MsgBox"Assetlifemustbeayearormore."
   MonthLife=InputBox("What'stheasset'susefullifeinmonths?")
Loop
LifeTime=MonthLife/YRMOS   '将月份数转成年份数。
IfLifeTime<>Int(MonthLife/YRMOS)Then
   LifeTime=Int(LifeTime+1) '四舍五入至最接近的年份。
EndIf
DepYear=CInt(InputBox("Enteryearfordepreciationcalculation."))
DoWhileDepYear<1OrDepYear>LifeTime
   MsgBox"Youmustenteratleast1butnotmorethan"&LifeTime
   DepYear=InputBox("Enteryearfordepreciationcalculation.")
Loop
```

```
Depr=DDB(InitCost,SalvageVal,LifeTime,DepYear)
MsgBox"Thedepreciationforyear"&DepYear&"is"&_
Format(Depr,Fmt)&"."
```

Debug 对象

Debug 对象在运行时将输出发送到 Immediate 窗口。

方法

Print 方法，Assert 方法

Decimal 数据类型

Decimal 变量存储为 96 位（12 个字节）无符号的整型形式，并除以一个 10 的幂数。这个变比因子决定了小数点右面的数字位数，其范围从 0 到 28。变比因子为 0（没有小数位）的情形下，最大的可能值为+/-79,228,162,514,264,337,593,543,950,335。而在有 28 个小数位的情况下，最大值为+/-7.9228162514264337593543950335，而最小的非零值为+/-0.0000000000000000000000000001。

注意此时，Decimal 数据类型只能在 Variant 中使用，也就是说，不能声明一变量为 Decimal 的类型。不过可用 Cdec 函数，创建一个子类型为 Decimal 的 Variant。

请参阅

TypeConversion 函数，DataType 总结，Deftype 语句

Declare 语句

用于在模块级别中声明对动态链接库(DLL)中外部过程的引用。

语法 1

[Public|Private]DeclareSub*name*Lib"*libname*"[Alias"*aliasname*"][([*arglist*])]

语法 2

[Public|Private]DeclareFunction*name*Lib"*libname*"[Alias"*aliasname*"][([*arglist*])][As*type*]

Declare 语句的语法包含下面部分：

部分	描述
Public	可选的。用于声明对所有模块中的所有其它过程都可以使用的过程。
Private	可选的。用于声明只能在包含该声明的模块中使用的过程。
Sub	可选的（但 Sub 或 Function 二者需选其一）。表示该过程没有返回值。
Function	可选的（但 Sub 或 Function 二者需选其一）。表示该过程会返回一个可用于表达式的值。
Name	必需的。任何合法的过程名。注意动态链接库的入口处（entrypoints）区分大小写。

Lib	必需的。指明包含所声明过程的动态链接库或代码资源。所有声明都需要 Lib 子句。
Libname	必需的。包含所声明的过程动态链接库名或代码资源名。
Alias	可选的。表示将被调用的过程在动态链接库(DLL)中还有另外的名称。当外部过程名与某个关键字重名时，就可以使用这个参数。当动态链接库的过程与同一范围内的公用变量、常数或任何其它过程的名称相同时，也可以使用 Alias。如果该动态链接库过程中的某个字符不符合动态链接库的命名约定时，也可以使用 Alias。
Aliasname	可选的。动态链接库或代码资源中的过程名。如果首字符不是数字符号(#)，则 aliasname 是动态链接库中该过程的入口处的名称。如果首字符是(#)，则随后的字符必须指定该过程的入口处的顺序号。
Arglist	可选的。代表调用该过程时需要传递的参数的变量表。
Type	可选的。Function 过程返回值的数据类型；可以是 Byte、布尔、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（只支持变长）或 Variant，用户定义类型，或对象类型。

arglist 参数的语法以及语法各个部分如下：

[**Optional**][**ByVal**|**ByRef**][**ParamArray**]*varname*[()][**As***type*]

部分	描述
Optional	可选的。表示参数不是必需的。如果使用该选项，则 arglist 中的后续参数都必需是可选的，而且必须都使用 Optional 关键字声明。如果使用了 ParamArray，则任何参数都不能使用 Optional。
ByVal	可选的。表示该参数按值传递。
ByRef	表示该参数按地址传递。ByRef 是 VisualBasic 的缺省选项。
ParamArray	可选的。只用于 arglist 的最后一个参数，表示最后的参数是一个 Variant 元素的 Optional 的数组。使用 ParamArray 关键字可以提供任意数目的参数。ParamArray 关键字不能与 ByVal、ByRef 或 Optional 一起使用。
Varname	必需的。代表传给该过程的参数的变量名；遵循标准的变量命名约定。
()	对数组变量是必需的。指明 varname 是一个数组。
Type	可选的。传递给该过程的参数的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（只支持变长）、Object、Variant、用户自定义的类型或对象类型。

说明

对 Function 过程而言，过程的数据类型决定其返回值的数据类型。可以在 arglist 之后使用 As 子句来指定函数返回值的数据类型。在 arglist 中，可以使用 As 子句来指定任何传给该过程的参数的数据类型。不但可以指定为任何标准数据类型，还可以在 arglist 中指定 AsAny 来禁止类型检查，从而允许将任意数据类型传递给该过程。

空圆括号表示该 Sub 或 Function 过程没有参数，且 VisualBasic 应确保不会传递任何参数。在下面的示例中，First 不带任何参数。如果对 First 的调用中使用了参数，就会产生错误：

```
DeclareSubFirstLib"MyLib"()
```

如果带参数表，则每次调用该过程时都要检查参数的个数和类型。在下面的示例中，First 有一个 Long 参数：

```
DeclareSubFirstLib"MyLib"(XAsLong)
```

注意在 Declare 语句的参数表中不能有定长的字符串；只有变长的字符串才能传给过程。定长的字符串可以作为过程参数使用，但在传递前都要被转换为变长的字符串。

注意当所调用的外部过程需要一个值为 0 的字符串时，就要使用 vbNullString 常数。该常数与零长度字符串("")是不相同的。

示例

该示例演示如何在标准模块的模块级使用 Declare 语句来声明对

动态链接库(DLL)中的一个外部过程的引用。如果是 Private 的 Declare 语句，则可以用于类模块。

```
'在 MicrosoftWindows（16 位）中：
DeclareSubMessageBeepLib"User"(ByValNAsInteger)
'假设 SomeBeep 是该过程名的别名。
DeclareSubMessageBeepLib"User"Alias"SomeBeep"(ByValNAsInteger)
'在 Alias 子句使用顺序号来调用 GetWinFlags。
DeclareFunctionGetWinFlagsLib"Kernel"Alias"#132"()AsLong
'在 32 位 MicrosoftWindows 系统中，指定的库应是 User32.DLL，
'而不是 User.DLL。可以使用条件编译来编写
'既可在 Win32 又可在 Win16 环境运行的代码。
#IfWin32Then
   DeclareSubMessageBeepLib"User32"(ByValNAsLong)
#Else
   DeclareSubMessageBeepLib"User"(ByValNAsInteger)
#EndIf
```

请参阅

Call 语句，Function 语句，Sub 语句，LastDLLError 属性，AddressOf 运算符

Default 属性

返回或设置一个值，以确定哪一个 CommandButton 控件是窗体的

缺省命令按钮。

请参阅

Extender 对象，CommandButton 控件，OLE 包容器控件

语法

object.**Default**[=*boolean*]

Default 属性语法包含下面部分：

部分	描述
Object	对象表达式，其值是“应用于”列表中的一个对象。
Boolean	布尔表达式，指定该命令按钮是否为缺省按钮，“设置值”中有详细描述。

设置值

boolean 的设置值为

设置值	描述
True	该 CommandButton 是缺省命令按钮。
False	（缺省值）该 CommandButton 不是缺省命令按钮。

说明

窗体中只能有一个命令按钮可以为缺省命令按钮。当某个命令按钮的 Default 设置为 True 时，窗体中其它的命令按钮自动设置为 False。当命令按钮的 Default 设置为 True 而且其父窗体是活动的，用户可以按 ENTER 键选择该按钮（激活其单击事件）。任何其它有焦点的控件都不接受 ENTER 键的键盘事件（KeyDown，KeyPress 或 KeyUp），除非用户将焦点移到同一窗体的另外一个命令按钮

上。在这种情况下，按 ENTER 键选择有焦点的命令按钮而不是缺省命令按钮。

对于支持如删除等不可恢复操作的窗体或对话框，将取消按钮的 Default 属性设置成 True，使其成为缺省命令按钮。

对于 OLE 容器控件，只为那些行为象 CommandButton 控件的对象才有 Default 属性。

请参阅

Cancel 属性

DefaultBind 属性

返回某个 Member 对象的 DefaultBind 属性，或者对其进行设置。

应用于

Member 对象

语法

object.**DefaultBind**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

DefaultCancel 属性

返回或设置一个数值，该数值决定某个控件是否可以充当标准的命令按钮。在控件创建时，DefaultCancel 属性可读可写，在控件运行时，该属性不可用。

应用于

UserControl 对象

设置值

DefaultCancel 的设置值为：

设置值	描述
true	控件可以充当缺省的或取消命令的按钮。容器将在扩展对象中添加 Default 和 Cancel 属性。在添加 Default 和 Cancel 属性后，控件就可充当标准命令按钮。于是，该控件可设置这些添加的扩展属性。
false	（缺省）控件不能充当缺省的或取消命令的按钮。所有子控件都不能将其 Default 或 Cancel 属性设置为 True。

说明

同时将 Default 属性和某个子控件的 Default 属性设置为 True，其结果是当按下 ENTER 键时导致按下该子控件，否则，当按下 ENTER 键时将引发控件的 AccessKeyPress 事件。

同时将 Cancel 属性和某个子控件的 Cancel 属性设置为 True，其结果是当按下 ESCAPE 键时导致按下该子控件，否则，当按下 ESCAPE 键时将引发控件的 AccessKeyPress 事件。

重点任何时候都可以改变缺省按钮或取消按钮的状态。代码必须放在控件的 AmbientChanged 事件过程中，以检测 DisplayAsDefault 属性的变更，以及调整后控件的外观。

请参阅

AccessKeyPress 事件，AmbientChanged 事件，Cancel 属性，Default 属性

DefaultExt 属性

为该对话框返回或设置缺省的文件扩展名。

应用于

CommonDialog 控件（“打开”，“另存为”对话框）

语法

object.**DefaultExt**[=*string*]

DefaultExt 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
string	字符串表达式，它用来指定文件的扩展名

说明

使用该属性指定缺省的扩展文件名，如.txt 或.doc。

当保存一个没有扩展名的文件时，自动给该文件指定由 DefaultExt 属性指定的扩展名。

数据类型

String

Deftype 语句

在模块级别上，为变量和传给过程的参数，设置缺省数据类型，以及为其名称以指定的字符开头的 Function 和 PropertyGet 过程，设置返回值类型。

语法

DefBool*letterrange*[,*letterrange*]...

DefByte*letterrange*[,*letterrange*]...

DefInt*letterrange*[,*letterrange*]...

DefLng*letterrange*[,*letterrange*]...

DefCur*letterrange*[,*letterrange*]...

DefSng*letterrange*[,*letterrange*]...

DefDbl*letterrange*[,*letterrange*]...

DefDec*letterrange*[,*letterrange*]...

DefDate*letterrange*[,*letterrange*]...

DefStr*letterrange*[,*letterrange*]...

DefObj*letterrange*[,*letterrange*]...

DefVar*letterrange*[,*letterrange*]...

所需的 letterrange 参数遵循下述语法：

letter1[-*letter2*]

letter1 和 letter2 参数指定设置缺省数据类型的名称范围。每个参数都是指变量，参数和 Function 过程，或 PropertyGet 过程名称的

首字母，且参数可以是字母表中的任意字母。letterrange 中不区分字母的大小写。

说明

语句的名字就确定相应的数据类型：

语句	数据类型
DefBool	布尔
DefByte	Byte
DefInt	Integer
DefLng	Long
DefCur	Currency
DefSng	Single
DefDbl	Double
DefDec	Decimal（目前尚不支持）
DefDate	Date
DefStr	String
DefObj	Object
DefVar	Variant

例如，在下面的程序段中，Message 就是一个字符串变量：

```
DefStrA-Q
...
Message="Outofstackspace."
```

Deftype 语句只在使用该语句的模块中有效。例如，一个模块中

的 DefInt 语句只对在该模块中声明的变量和传给过程的参数的缺省数据类型，以及 Function 和 PropertyGet 过程的返回值的类型有效；而其它模块中的变量、参数、以及返回值的缺省数据类型就不受影响。如果不用 Deftype 语句显式地声明，则所有变量、参数、Function 过程、以及 PropertyGet 过程的缺省数据类型都是 Variant。

当指定字符范围时，通常为以字符集的前 128 个字符中的字符开始的变量定义数据类型。不过，如果指定的字符范围是 A–Z，则将所有的变量，包括以字符集的扩展部分(128–255)中的国际字符开始的变量的缺省类型都设为指定的类型。

在指定了 A–Z 范围之后，就不能再使用 Deftype 语句来重新定义任何子范围的变量了。在指定一个范围后，如果另一个 Deftype 语句定义的范围中含有前面已定义的字符，就会产生错误。不过，不管变量是否已定义，都可以使用带 Astype 子句的 Dim 语句来显式指定其数据类型。例如，可以在模块级使用如下代码将一个缺省数据类型为 Integer 的变量定义为 Double：

```
DefIntA-Z
DimTaxRateAsDouble
```

Deftype 语句对用户定义类型中的元素无影响，因为这些元素必须显式声明。

请参阅

Function 语句，Let 语句，PropetyGet 语句

示例

该示例演示了 Deftype 语句的多种用法，来设置那些名称是以指定字符开头的变量和函数过程的缺省数据类型。只有使用 Dim 语句的显式赋值，才可以覆盖这种缺省数据类型。Deftype 语句只能在模块级使用（即不能在过程内使用）。

```
'将名称以 A 至 K 开头变量的缺省数据类型设为 Integer 类型。
DefIntA-K
'将名称以 L 至 Z 开头变量的缺省数据类型设为 String 类型。
DefStrL-Z
CalcVar=4  '初始化为 Integer。
StringVar="Hellothere" '初始化为 String。
AnyVar="Hello" '导致"Typemismatch"错误。
DimCalcAsDouble     '赋给一个 Double 类型。
Calc=2.3455     '允许指定为一个 Double 数。

'Deftype 语句也可以应用于函数过程。
CalcNum=ATestFunction(4)    '调用用户自定义的函数。
'ATestFunction 函数过程的定义。
FunctionATestFunction(INumber)
   ATestFunction=INumber*2  '返回值是一个 Integer。
EndFunction
```

Delete 方法

删除一个指定的文件或文件夹。

应用于

File 对象，Folder 对象

语法

object.**Delete***force*

Delete 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 File 或 Folder 对象的名
force	可选的。Boolean 值，如果要删除具有只读属性设置的文件或文件夹，其值为 True。当其值为 False 时（缺省），不能删除具有只读属性设置的文件或文件夹

说明

如果指定的文件或文件夹不存在，则发生一个错误。

对于一个 File 或 Folder，Delete 方法的结果和执行 FileSystemObject.DeleteFile 或 FileSystemObject.DeleteFolder 操作的结果是一样的。

Delete 方法对于文件夹内是否有内容不做区别。不管指定的文件夹是否有内容，它都被删除。

请参阅

Copy 方法，DeleteFile 方法，DeleteFolder 方法，Move 方法（FileSystemObject 对象），OpenAsTextStream 方法

Delete 方法（OLE 容器）

删除指定的对象，而释放与之关联的内存。

应用于

OLE 容器控件

语法

object.**Delete**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

这个方法允许显式地删除对象。当关闭窗体时或对象被一个新对象取代时，对象都将被自动删除。

请参阅

MAPI 控件,MAPISession 控件,MAPIMessages 控,AttachmentCount 属性，AttachementIndex 属性，MsgCount 属性，MsgIndex 属性，RecipCount 属性，RecipIndex 属性

DeleteFile 方法

删除一个指定的文件。

应用于

FileSystemObject 对象

语法

object.**DeleteFile***filespec*[,*force*]

DeleteFile 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
filespec	必需的。要删除文件的名字。Filespec 可以在最后的路径部件中包含通配符
force	可选的。Boolean 值，如果要删除具有只读属性设置的文件，其值为 True。如果其值为 False（缺省），则不能删除具有只读属性设置的文件

说明

如果没有发现相匹配的文件，则产生一个错误。DeleteFile 方法停在它遇到的第一个错误上。不要尝试回卷或撤消错误发生前所做的任何改变。

请参阅

CopyFile 方法，Delete 方法，DeleteFolder 方法，MoveFile 方法

DeleteFolder 方法

删除一个指定的文件夹和它的内容。

应用于

FileSystemObject 对象

语法

object.**DeleteFolder***folderspec*[,*force*]

DeleteFolder 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
folderspec	必需的。要删除的文件夹的名字。Folderspec 可以在最后的路径部件中包含通配符
force	可选的。Boolean 值，如果要删除具有只读属性设置的文件夹，其值为 True，如果值为 False（缺省），则不能删除具有只读属性设置的文件夹

说明

DeleteFolder 方法对文件夹中有无内容不做区别。不管指定的文件夹中是否有内容，它都被删除。

如果没有发现相匹配的文件夹，则发生一个错误。DeleteFolder 方法停止在它遇到的第一个错误上，不要尝试回卷或撤消错误发生前所做的任何改变。

请参阅

CopyFolder 方法, CreateFolder 方法, Delete 方法, DeleteFile 方法，MoveFolder 方法

DeleteLines 方法

删除一个单行或指定行范围的代码。

应用于

CodeModule 对象

语法

object.**DeleteLines**(*startline*)[*count*]

DeleteLines 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
startline	必需的。一个 Long 型数，用来指定欲删除的开始行
count	可选的参数。一个 Long 型数，用来指定欲删除的行数

说明

如果没有指定欲删除的行数，DeleteLines 将只删除一行。

请参阅

InsertLines 方法，Lines 方法

示例

下列示例包含两个步骤。第一个步骤里的 ForNext 循环使用 InsertLines 方法将 26 个英文字母排成 26 行文本（第一行只有一个字母“A”，第二行有两个字母“ab”，其余依此类推，最后一行就包含“A”到“z”共26个字母），插入到 CodePanes(1)中。第二个步骤里的 For-Next 循环使用 DeleteLines 方法，将第一个步骤所生成的 26 行文本之单数行删除。虽然在第二个循环里，表面上看起来好象是在删除前面 13 行文本（For I=1to13），但实际上每删除掉一行，其后每一行的行号就都减少了，所以其实删除

的只有单数行。下面的解释应使读者更清楚：当 I=1 时，删除第一行，则第二行变成第一行，第三行变成第二行；因此当 I=2 时，表面上删除的是第二行，但这个第二行是原本的第三行，因此删除的是原本的单数行，并且此时第三行变成第二行，第四行变成第三行；依次而下，直到删除所有的单数行为止。

```
ForI=1to26
   Application.VBE.SelectedVBComponent.CodeModule.InsertLines;i,
Mid$("abcdefghijklmnopqrstuvwxyz",1,I)
Next
ForI=1to13
   Application.VBE.SelectedVBComponent.CodeModule.DeleteLinesI
Next
```

DeleteSetting 语句

在 Windows 注册表中或(Macintosh 中)应用程序初始化文件中的信息，从应用程序项目里删除区域或注册表项设置。

语法

DeleteSetting*appname*,*section*[,*key*]

DeleteSetting 语句的语法具有下列命名参数：

部分	描述
appname	必需的。字符串表达式，包含应用程序或工程的名称，区域或注册表项用于这些应用程序或工程。在 Macintosh 中，这是 System 文件夹中 Preferences 文件夹中初始化文件的文件名。
section	必需的。字符串表达式，包含要删除注册表项设置的区域名称。如果只有 appname 和 section，则将指定的区域连同所有有关的注册表项设置都删除。
key	可选。字符串表达式，包含要删除的注册表项设置。

说明

如果提供了所有参数，则删除指定的注册表项设置。如果试图使用不存在的区域或注册表项设置上的 DeleteSetting 语句，则发生一个运行时错误。

请参阅

GetAllSettings 函数，GetSetting 函数，SaveSetting 语句

示例

下列示例先使用 SaveSetting 语句，来建立 WindowsMacintosh 注册区（或 16 位 Windows 平台的 ini 文件）里 MyApp 应用程序的项目，然后使用 DeleteSetting 语句将之删除。因为没有指定 key 参数，整个区段都会被删除掉，包括区段名称及其所有的机码（key）。

```
'在注册区中添加一些设置值。
SaveSettingappname:="MyApp",section:="Startup",_
key:="Top",setting:=75
SaveSetting"MyApp","Startup","Left",50
'删除区段及所有的设置值。
DeleteSetting"MyApp","Startup"
```

导出的数学函数

以下为非基本数学函数的列表，皆可由基本数学函数导出：

函数	由基本函数导出之公式
Secant（正割）	Sec(X)=1/Cos(X)
Cosecant（余割）	Cosec(X)=1/Sin(X)
Cotangent（余切）	Cotan(X)=1/Tan(X)
InverseSine（反正弦）	Arcsin(X)=Atn(X/Sqr(-X*X+1))
InverseCosine（反余弦）	Arccos(X)=Atn(-X/Sqr(-X*X+1))+2*Atn(1)
InverseSecant（反正割）	Arcsec(X)=Atn(X/Sqr(X*X-1))+Sgn((X)-1)*(2*Atn(1))
InverseCosecant（反余割）	Arccosec(X)=Atn(X/Sqr(X*X-1))+(Sgn(X)-1)*(2*Atn(1))
InverseCotangent	Arccotan(X)=Atn(X)+2*Atn(1)

（反余切）

HyperbolicSine　　HSin(X)=(Exp(X)-Exp(-X))/2

（双曲正弦）

HyperbolicCosine　　HCos(X)=(Exp(X)+Exp(-X))/2

（双曲余弦）

HyperbolicTangent　　HTan(X)=(Exp(X)-Exp(-

（双曲正切）　　X))/(Exp(X)+Exp(-X))

HyperbolicSecant　　HSec(X)=2/(Exp(X)+Exp(-X))

（双曲正割）

HyperbolicCosecant　　HCosec(X)=2/(Exp(X)-Exp(-X))

（双曲余割）

HyperbolicCotangent　　HCotan(X)=(Exp(X)+Exp(-X))/(Exp(X)-

（双曲余切）　　Exp(-X))

InverseHyperbolicSine　　HArcsin(X)=Log(X+Sqr(X*X+1))

（反双曲正弦）

InverseHyperbolicCosine　　HArccos(X)=Log(X+Sqr(X*X-1))

（反双曲余弦）

InverseHyperbolicTangent　　HArctan(X)=Log((1+X)/(1-X))/2

（反双曲正切）

InverseHyperbolicSecant　　HArcsec(X)=Log((Sqr(-X*X+1)+1)/X)

（反双曲正割）

InverseHyperbolicCosecant　　HArccosec(X)=Log((Sgn(X)*Sqr(X*X+1)

+1)/X)

InverseHyperbolicCotangent（反双曲余切）	HArccotan(X)=Log((X+1)/(X-1))/2
以 N 为底的对数	LogN(X)=Log(X)/Log(N)

请参阅

Atn 函数，Cos 函数，Exp 函数，Log 函数，Sin 函数，Sqr 函数

Description 属性（应用于 VisualBasic 子应用程序）

返回或设置一个字符串表达式，包含与对象相关联的描述性字符串。可读/可写。

对于 Err 对象，返回或设置与错误相关联的描述性字符串。

应用于

Error 对象（数据报表设计程序），Err 对象

说明

Description 属性设置对错误进行简短描述。当无法处理或不想处理错误的时候，可以使用这个属性提醒用户。在生成用户自定义的错误时，将有关此错误的一个简短陈述指定给 Description 属性。如果 Description 未填入数据，而且 Number 的值与 VisualBasic 运行时错误一致，那么在生成错误时，将 Error 函数返回的字符串放置在 Description 中。

请参阅

GUID 属性，Major 属性，Minor 属性，Version 属性，Error 函

数，Err 对象，HelpContext 属性，HelpFile 属性，LastDLLError 属性，Number 属性，Source 属性

示例

本示例将一个用户自定义的信息，指定给 Err 对象的 Description 属性。

```
Err.Description="Itwasnotpossibletoaccessanobjectnecessary"_
&"forthisoperation。"
```

Description 属性（添加到 ObjectModelVBA）

返回或设置一个字符串表达式，包含对某个对象的描述性字符串。对于 VBProject 对象，此属性可读/写；对于 Reference 对象，此属性为只读。

应用于

Reference 对象，VBProject 对象

说明

对于 VBProject 对象，Description 属性返回或设置一个关于活动的工程的描述性字符串。

对于 Reference 对象，Description 属性返回该引用的描述性的名称。

请参阅

GUID 属性，HelpContextID 属性（VBAAdd-InObjectModel），Major 属性，Minor 属性，Name 属性（VBAAdd-InObjectModel），Version 属性

示例

下列示例中，第一个示例使用 Description 属性将一个描述赋予指定的工程；该示例还将该描述显示出来以供确认赋值是否成功。

第二个示例使用 Description 属性返回某个工程中指定的 Reference 对象的描述名称。

```
Application.VBE.VBProjects(1).Description="HotSauce"
Debug.PrintApplication.VBE.VBProjects(1).Description

Debug.PrintApplication.VBE.VBProjects(1).References(1).Description

Debug.PrintApplication.VBE.VBProjects(1).References(2).Description
```

Designer 属性

返回一个可以访问部件设计特征的对象。

应用于

VBComponent 对象

说明

如果该对象有一个打开设计器，Designer 属性可返回该打开设计器；否则便可创建一个新的设计器。这个设计器是某一个特定

VBComponent 对象的特征。譬如，当创建某一特定类型的 VBComponent 对象，一个设计器便可伴随着此对象而创建。一个部件只能有一个设计器，而且通常都是同一个设计器。可以访问具有 Designer 属性的指定部件对象。在某些情况下，例如在标准模块及类模块中，设计器将不可被创建，因为这种类型的 VBComponent 对象并不支持设计器。

如果 VBComponent 对象没有设计器的话，Designer 属性可返回 Nothing。

请参阅

DesignerWindow 方法，HasOpenDesigner 属性

示例

下列示例使用 Designer 及 Count 属性返回窗体所包含的控件总数。请注意，包含该窗体的窗口必需处在被选择的状态才行。Designer 对象则是指窗体本身。

```
Debug.PrintApplication.VBE.SelectVBComponent.Designer.Controls.Count
```

DesignerID 属性

返回由对象 VBComponent 表示的设计器类型，是只读属性。

应用于

VBComponent 对象

请参阅

ActiveCodePane 属性，ActiveVBProject 属性，ActiveWindow 属性，CodePanes 属性，MainWindow 属性，NameProPerty(VBAAdd-InObjectModel),SelectedVBComponent 属性，Version 属性，CommandBars 属性，AddIns 属性，Events 属性，VBProjects 属性，Windows 属性，DisplayModel 属性，FullName 属性，LastUsedPath 属性，ReadOnlyMode 属性，TemplatePath 属性

DesignerWindow 方法

返回代表部件设计器的 Window 对象。

应用于

VBComponent 对象

语法

object.**DesignerWindow**

该 *object* 所在处是一个对象表达式，其值为“应用于”列表中的对象。

说明

如果部件支持设计器，但没有打开的设计器，使用 DesignerWindow 方法创建设计器，但却是不可见的。要使该窗口可见，将 Window 对象的 Visible 属性设置为 True。

请参阅

Window 对象，Designer 属性，HasOpenDesigne 属性，Visible 属性

示例

下列示例使用 DesignerWindow 方法和 Visible 属性查找一个特定的设计器是否是可见的。注意 VBComponent 对象必须是一个窗体。

```
Debug.PrintApplication.VBE.VBProjects(1).VBComponents(1).DesignerWindow.Visible
```

DeviceName 属性

返回驱动器支持的设备名。

应用于

Printer 对象，Printers 集合

语法

object.**DeviceName**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

每个打印机驱动程序可以支持一个或多个设备——例如，HPLaserJetIIISi 是一个设备名。

注意 Printer 对象属性的效果，取决于打印机生产商提供的驱动程序。一些属性设置可能不起作用，或几个不同的属性设置具有相同的结果。如果设置值超出可接受范围会产生错误。有关更多的信息，请参阅具体驱动程序的由生产商提供的文档。

请参阅

hDC 属性

DHTMLEvent 属性

在 DynamicHTML 对象模型中返回 event 对象。

应用于

DHTMLPage 对象

语法

object.DHTMLEvent

object 所在处表示一个对象表达式，其值为“应用于”列表中的一个对象。

说明

要获得某次鼠标单击时的 X 坐标，使用 DynamicHTML 对象模型的用户在编程时通常必需键入：

```
Document.parentWindow.event.x
```

但是使用这个 VisualBasic 快捷方式，用户只需键入：

```
DHTMLEvent.x
```

要取消 DynamicHTML 中一个事件的触发，可以写成：

```
Window.event.cancelBubble=1
```

-或-

Document.parentWindow.event.cancelBubble=1

然而，用 DHTMLEvent 属性，VisualBasic 用户只需键入：
DHTMLEvent.cancelBubble=1

DHTMLPage 对象

运行时部件，在 VisualBasic 运行库和 DynamicHTML 对象模型之间挂接事件。

语法

DHTMLPage

说明

DHTMLPage 对象是 VisualBasicDHTMLPageDesigner 应用程序的一个运行时部件。

属性

Document 属性（DHTMLPageDesigner），DHTMLEvent 属性，BaseWindow 属性

事件

Load 事件，Unload 事件

请参阅

《MicrosoftVisualBasic6.0 部件工具》一书的第五部分“构造 Internet 应用程序”第二章“开发 DHTML 应用程序”中的“DHTML 应用程序中的事件”，“DHTML 应用程序中的事件产生”和“动态

HTML 中的事件”。

DHTMLPageDesigner 对象

表示设计时的页面设计器。运行时不可用。

说明

只有在 VisualBasic 环境中，使用 DHTMLPageDesigner 创建工程时，DHTMLPageDesigner 对象的属性才可用。DHTMLPageDesigner 下的各项是设计时属性，因而不能通过代码访问。

属性

AsyncLoad 属性，ID 属性，SourceFile 属性，BuildFile 属性

请参阅

KeyEventsinDHTMLapplications

DialogTitle 属性

返回或设置该对话框标题栏所显示的字符串。

应用于

CommonDialog 控件（“打开”，“另存为“对话框）

语法

object.**DialogTitle**[=*title*]

DialogTitle 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
title	指定该对话框名称的字符串表达式

说明

使用该属性显示标题栏中对话框的名称。

注意当显示“颜色”、“字体”或“打印”对话框时，CommonDialog 控件忽略 DialogTitle 属性的设置。

“打开”对话框缺省的标题是“打开”；“另存为”对话框缺省的标题是“另存为”。

数据类型

String

Dictionary 对象

对象，用于存储数据关键字和条目对。

语法

Scripting.Dictionary

说明

Dictionary 对象与 PERL 关联数组等价。可以是任何形式的数据的条目被存储在数组中。每个条目都与一个唯一的关键字相关联。该关键字用来检索单个条目，通常是整数或字符串，可以是除数

组外的任何类型。

下面的代码举例说明了如何创建一个 Dictionary 对象：

```
Dimd'创建一个变量
Setd=CreateObject(Scripting.Dictionary)
d.Add"a","Athens"'添加一些关键字和条目
d.Add"b","Belgrade"
d.Add"c","Cairo"
...
```

属性

CompareMode 属性，Count 属性，Item 属性，Key 属性

方法

Add 方法（FileSystemObject 对象），Exists 方法，Items 方法，Keys 方法，Remove 方法（FileSystemObject 对象），RemoveAll 方法

请参阅

FileSystemObject 对象，TextStream 对象

Dim 语句

声明变量并分配存储空间。

语法

Dim[**WithEvents**]*varname*[([*subscripts*])][**As**[**New**]*type*][,[**WithEvents**]*varname*[([*subscripts*])][**As**[**New**]*type*]]...

Dim 语句的语法包含下面部分：

部分	描述
WithEvents	可选的。关键字，说明 varname 是一个用来响应由 ActiveX 对象触发的事件的对象变量。只有在类模块中才是合法的。使用 WithEvents，可以声明任意个所需的单变量，但不能使用 WithEvents 创建数组。New 和 WithEvents 不能一起使用。
varname	必需的。变量的名称；遵循标准的变量命名约定。
subscripts	可选的。数组变量的维数；最多可以定义 60 维的多维数组。subscripts 参数使用下面的语法： [lowerTo]upper[,[lowerTo]upper]... 如果不显式指定 lower，则数组的下界由 OptionBase 语句控制。如果没有使用 OptionBase 语句，则下界为 0。
New	可选的。可隐式地创建对象的关键字。如果使用 New 来声明对象变量，则在第一次引用该变量时将新建该对象的实例，因此不必使用 Set 语句来给该对象引用赋值。New 关键字不能声明任何内部数据类型的变量，以及从属对象的实例，也不能与 WithEvents 一起使用。

Type 可选的。变量的数据类型；可以是 Byte、布尔、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（对变长的字符串）、String*length（对定长的字符串）、Object、Variant、用户定义类型、或对象类型。所声明的每个变量都要一个单独的 Astype 子句。

说明

在模块级别中用 Dim 声明的变量，对该模块中的所有过程都是可用的。在过程级别中声明的变量，只在过程内是可用的。
可以使用 Dim 语句在模块级或过程级中声明变量的数据类型。例如，下面的语句声明了 Integer 类型的变量。

```
DimNumberOfEmployeesAsInteger
```

也可以使用 Dim 语句来声明变量的对象类型。下面的语句为工作表的新建实例声明了一个变量。

```
DimXAsNewWorksheet
```

如果定义对象变量时没有使用 New 关键字，则在使用该变量之前，必须使用 Set 语句将该引用对象的变量赋值为一个已有对象。在该变量被赋值之前，所声明的对象变量有一个特定值 Nothing，这个值表示该变量没有指向任一个对象实例。
也可以使用带空圆括号的 Dim 语句来定义动态数组。在定义动态数组后，可以在过程内使用 ReDim 语句来定义该数组的维数和元

素。如果试图在 Private，Public 或 Dim 语句中重新定义一个已显式定义了大小的数组的维数时，就会发生错误。

如果不指定数据类型或对象类型，且在模块中没有 Deftype 语句，则该变量按缺省设置是 Variant 类型。

当初始化变量时，数值变量被初始化为 0，变长的字符串被初始化为一个零长度的字符串(""），而定长的字符串则用 0 填充。Variant 变量被初始化为 Empty。用户自定义类型的变量的每个元素作为各自独立的变量进行初始化。

注意当在过程中使用 Dim 语句时，通常将 Dim 语句放在过程的开始处。

请参阅

数据类型总结，Array 函数，OptionBase 语句，Private 语句，Public 语句，ReDim 语句，Set 语句，Static 语句，Type 语句

示例

该示例演示使用 Dim 语句来声明变量，也演示了用 Dim 语句来声明数组。数组的缺省下界为 0，可以在模块级使用 OptionBase 语句来取代数组的缺省下界。

```
'AnyValue 和 MyValue 按缺省情况被声明为 Variant，
'同时值被设为 Empty。
DimAnyValue,MyValue

'显式声明一个 Integer 类型的变量。
```

```
DimNumberAsInteger

'在一行中声明多个变量。AnotherVar 为 Variant 类型，
'因为它的类型被省略了。
DimAnotherVar,ChoiceAsBoolean,BirthDateAsDate

'DayArray 是一个有 51 个索引（从 0 到 50）元素的 Variant 数组，
'假设在当前模块中 OptionBase 被设为 0（缺省设置）。
DimDayArray(50)

'Matrix 是一个二维 Integer 数组。
DimMatrix(3,4)AsInteger

'MyMatrix 是一个显式指定了上下界
'的三维 double 数组。
DimMyMatrix(1To5,4To9,3To5)AsDouble

'BirthDay 是一个索引从 1 到 10 的 date 数组。
DimBirthDay(1To10)AsDate

'MyArray 是一个 variant 动态数组。
DimMyArray()
```

Dir 函数

返回一个 String，用以表示一个文件名、目录名或文件夹名称，它必须与指定的模式或文件属性、或磁盘卷标相匹配。

语法

Dir[(*pathname*[,*attributes*])]

Dir 函数的语法具有以下几个部分：

部分	描述
pathname	可选参数。用来指定文件名的字符串表达式，可能包含目录或文件夹、以及驱动器。如果没有找到 *pathname*，则会返回零长度字符串("")。
attributes	可选参数。常数或数值表达式，其总和用来指定文件属性。如果省略，则会返回匹配 *pathname* 但不包含属性的文件。

设置值

attributes 参数的设置可为：

常数	值	描述
vbNormal	0	(缺省)指定没有属性的文件。
vbReadOnly	1	指定无属性的只读文件
vbHidden	2	指定无属性的隐藏文件
vbSystem	4	指定无属性的系统文件在 Macintosh 中不可用。
vbVolume	8	指定卷标文件；如果指定了其它属性，则忽略 vbVolume 在 Macintosh 中不可用。
vbDirectory	16	指定无属性文件及其路径和文件夹。

注意这些常数是由 VBA 所指定的，在程序代码中的任何位置，可以使用这些常数来替换真正的数值。

说明

在 MicrosoftWindows 中，Dir 支持多字符(*)和单字符(?)的通配符来指定多重文件。

请参阅

ChDir 语句，CurDir 函数

示例

本示例使用 Dir 函数来检查某些文件或目录是否存在。

```
DimMyFile,MyPath,MyName
'返回“WIN.INI”(在MicrosoftWindows 中)（如果该文件存在)。
MyFile=Dir("C:\WINDOWS\WIN.Ini")
```

```
'返回带指定扩展名的文件名。如果超过一个*.ini 文件存在，
'函数将返回按条件第一个找到的文件名。
MyFile=Dir("C:\WINDOWS\*.Ini")

'若第二次调用 Dir 函数，但不带任何参数，则函数将返回同一目录下的
下一个*.ini 文件。
MyFile=Dir

'返回找到的第一个隐式*.TXT 文件。
MyFile=Dir("*.TXT",vbHidden)

'显示 C:\目录下的名称。
MyPath="c:\"    '指定路径。
MyName=Dir(MyPath,vbDirectory) '找寻第一项。
DoWhileMyName<>""  '开始循环。
   '跳过当前的目录及上层目录。
   IfMyName<>"."AndMyName<>".."Then
        '使用位比较来确定 MyName 代表一目录。
        If(GetAttr(MyPath&MyName)AndvbDirectory)=vbDirectoryThen
            Debug.PrintMyName     '如果它是一个目录，将其名称显示出
来。
        EndIf
```

```
    EndIf
    MyName=Dir   '查找下一个目录。
Loop
```

DirListBox 控件

在运行时，DirListBox 控件显示目录和路径。这个控件可以显示分层的目录列表。例如，可以创建对话框，在所有可用目录中，从文件列表打开一个文件。

语法

DirListBox

说明

设置 List、ListCount 和 ListIndex 属性，就可以访问列表中的项目。如果需要显示 DriveListBox 和 FileListBox 控件，那么可以编写代码，使它们与 DirListBox 同步，并使它们之间彼此同步。

属性

OLEDragMode 属性，OLEDropMode 属性，BackColor,ForeColor 属性，FontBold,FontItalic,FontStrikethru,FontUnderline 属性，FontName 属性，FontSize 属性，Height,Width 属性，Left,Top 属性，List 属性，ListCount 属性，ListIndex 属性，TabIndex 属性，Tag 属性，Visible 属性，DragIcon 属性，DragMode 属性，hWnd 属性，MouseIcon 属性，MousePointer 属性，TabStop 属性，TopIndex 属性，Appearance 属性，Enabled 属性，HelpCo

ntextID 属性，Index 属性（ControlArray），Name 属性，Parent 属性，Path 属性，Font 属性，Container 属性，ToolTipText 属性，WhatsThisHelpID 属性

方法

Refresh 方法，SetFocus 方法，Drag 方法，Move 方法，Zorder 方法，OLEDrag 方法，ShowWhatsThis 方法

事件

Change 事件,Click 事件,DragDrop 事件,DragOver 事件,GotFocus 事件，KeyDown,KeyUp 事件，KeyPress 事件，LostFocus 事件，MouseDown,MouseUp 事件,MouseMove 事件,Scroll 事件,Validate 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件

请参阅

DriveListBox 控件，FileListBox 控件，List 属性，ListCount 属性，ListIndex 属性

DisabledPicture 属性

返回或设置一个对图片的引用，该图片在控件无效时显示在控件中。（也就是说，当控件 Enabled 属性被设置为 False 时。）

应用于

CheckBox 控件，CommandButton 控件，OptionButton 控件

语法

object.**DisabledPicture**[=*picture*]

DisabledPicture 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象。
picture	一个包含图形的 Picture 对象，“设置值”中有详细的描述。

设置值

picture 的设置值为：

设置值	描述
(None)	（缺省）没有图片。
(Bitmap,icon,metafile)	指定一个图形。在设计时可以从属性窗口加载该图形。在运行时，也可以通过在一个位图、图标或元文件上使用 LoadPicture 函数，或通过将其设置到另一个控件的 Picture 属性上来设置这个属性。

说明

DisabledPicture 属性指定一个图片对象在控件（如 CommandButton）无效时显示。除非控件的 Style 属性设置为 1（图形的），否则 DisabledPicture 属性将被忽略。

图片在控件上位于水平和垂直位置的中央。如果与图片一起使用标题，那么图片将位于标题上面的中央处。如果图片对象太大而

与控件不相适合，那么将被裁剪一部分。

如果没有图片赋值给 DisabledPicture 属性，而有一个图片赋值给 Picture 属性，那么在控件无效时将显示该图片的灰色形式。

DisplayAsDefault 属性

返回布尔值，判断控件是否为容器的缺省按钮，如果是缺省按钮，则将其显示为缺省控件。

应用于

AmbientProperties 对象

语法

object.**DisplayAsDefault**

DisplayAsDefault 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

设置值

DisplayAsDefault 属性返回的可能的布尔值是：

设置值	描述
true	控件是缺省按钮。
false	控件不是缺省按钮。如果容器不能实现这种环境属性，这将是缺省设置值。

说明

容器中只能有一个控件可作为缺省控件；控件的容器判断哪个控件是当前的缺省控件，并通过 DisplayAsDefault 环境属性通知哪个控件。被通知的控件应绘制自身，表明它是缺省控件。所有其它控件都将自己的 DisplayAsDefault 环境属性设置为 False。
只有按钮类型控件才能作为缺省控件。

DisplayBind 属性

返回某个 Member 对象的 DisplayBind 属性，或者对其进行设置。

应用于

Member 对象

语法

object.DisplayBind

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

DisplayModel 属性

该属性返回或设置系统使用的显示模式。

应用于

VBE 对象

语法

object.**DisplayModelAsvbext_VBADisplayModel**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

设置值

vbext_VBADisplayModel 的设置值是：

常数	值	描述
vbext_dm_SDI	0	（缺省值）显示模式是 SDI（单一文档接口）。
vbext_dm_MDI	1	显示模式是 MDI（多文档接口）。

DisplayName 属性

返回一个字符串值，其中包含在错误消息中的控件用来标识自己的名称。

应用于

AmbientProperties 对象

语法

object.**DisplayName**

DisplayName 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

通过这个环境属性控件找出哪个容器（例如 VisualBasic）正在调用该控件的实例。在错误信息中，该字符串被当作该控件实例的名称。

如果容器没有实现这种环境属性，缺省值将是空字符串。

DisplayType 属性

返回或设置一个值，用于指示对象是显示其内容还是显示图标。

应用于

OLE 容器控件，OLEObject 对象

语法

object.**DisplayType**[=*value*]

DisplayType 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	整数或常数，指示对象是显示其内容还是显示图标，在“设置值”中有详细说明

设置值

value 的设置值是：

常数	值	描述
VbOLEDisplayContent	0	（缺省值）内容。当 OLE 容器控件包含对象时，在控件中显示该对象的数据。
VbOLEDisplayIcon	1	图标。当 OLE 容器控件包含对象时，在控件中显示该对象的图标。

说明

在插入对象及特殊粘贴对话框中，这个属性指定作为图标显示的

复选框的缺省设置。如果这个属性被设置为 1（图标），无论是在运行时（用 InsertObjDlg 或 PasteSpecialDlg 方法)还是在设计时，当显示这些对话框时，作为图标显示的复选框是自动被选择的。在运行时，当使用 CreateEmbed 或 CreateLink 方法创建对象时，要使用 DisplayType 属性确定，在控件中该对象是显示图标（设置 DisplayType=1），还是显示数据（设置 DisplayType=0）。

对象一旦创建，就不能再改变其显示类型。

请参阅

PasteSpecialDlg 方法，CreateEmbed 方法，CreateLink 方法，InsertObjDlg 方法

Do...Loop 语句

当条件为 True 时，或直到条件变为 True 时，重复执行一个语句块中的命令。

语法

Do[{**While**|**Until**}*condition*]

[*statements*]

[**ExitDo**]

[*statements*]

Loop

或者可以使用下面这种语法：

Do

[*statements*]

[**ExitDo**]

[*statements*]

Loop[{**While**|**Until**}*condition*]

DoLoop 语句的语法具有以下几个部分：

部分	描述
condition	可选参数。数值表达式或字符串表达式，其值为 True 或 False。如果 condition 是 Null，则 condition 会被当作 False
statements	一条或多条命令，它们将被重复当或直到 condition 为 True

说明

在 Do…Loop 中可以在任何位置放置任意个数的 ExitDo 语句，随时跳出 Do…Loop 循环。ExitDo 通常用于条件判断之后，例如 I & T ，在这种情况下，ExitDo 语句将控制权转移到紧接在 Loop 命令之后的语句。

如果 ExitDo 使用在嵌套的 D & L 语句中，则 ExitDo 会将控制权转移到 ExitDo 所在位置的外层循环。

请参阅

Exit 语句，For...Next 语句，While...Wend 语句

示例

本示例示范如何使用 Do...Loop 语句。内层的 Do...Loop 语句循环到第 10 次时将标志值设置为 False，并用 ExitDo 语句强制退出内层循环。外层循环则在检查到标志值为 False 时，马上退出。

```
DimCheck,Counter
Check=True:Counter=0    '设置变量初始值。
Do '外层循环。
   DoWhileCounter<20    '内层循环。
      Counter=Counter+1    '计数器加一。
      IfCounter=10Then'如果条件成立。
         Check=False '将标志值设成 False。
         ExitDo   '退出内层循环。
      EndIf
   Loop
LoopUntilCheck=False   '退出外层循环。
```

Document 属性(DHTMLPageDesigner)

从 DynamicHTML 对象模型中返回 Document 对象，代表您在浏览器中看到的 HTML 页面。

应用于

DHTMLPage 对象

语法

object.Document

object 所在处表示一个对象表达式，其值为“应用于”列表中的一个对象。

说明

使用该对象响应发生在页面中的事件，并设置页面的属性。

来自 DynamicHTMLDocument 对象的任何可用的集合都是可以使用的。

请参阅

《MicrosoftVisualBasic6.0 部件工具指南》书中第五部分“创建 Internet 应用程序”的第二章的“开发 DHTML 应用程序”。

DoEvents 函数

转让控制权，以便让操作系统处理其它的事件。

语法

DoEvents()

说明

DoEvents 函数会返回一个 Integer，以代表 VisualBasic 独立版本中打开的窗体数目，例如，VisualBasic，专业版，在其它的应用程序中，DoEvents 返回 0。

DoEvents 会将控制权传给操作系统。当操作系统处理完队列中的事件，并且在 SendKeys 队列中的所有键也都已送出之后，返回控制权。

DoEvents 对于简化诸如允许用户取消一个已启动的过程 例如搜

寻一个文件特别有用。对于长时间过程，放弃控制权最好使用定时器或通过委派任务给 ActiveXEXE 部件来完成。以后，任务还是完全独立于应用程序，多任务及时间片由操作系统来处理。小心确保以 DoEvents 放弃控制权的过程，在第一次 DoEvents 返回之前，不能再次被其他部分的代码调用；否则会产生不可预料的结果。此外，如果其它的应用程序可能会和本过程以不可预知的方式进行交互操作，那么也不要使用 DoEvents，因为此时不能放弃控制权。

请参阅

SendKeys 语句

示例

下列示例在循环中使用 DoEvents 函数，每当循环完成 1000 次时，将执行让给操作系统。DoEvents 返回仅当主应用程序是 VisualBasic 时，打开的窗体个数。

```
'创建一个变量来保存加载的 VisualBasic 可见窗体的个数。
Dim I,OpenForms
For I=1To150000 '循环开始。
    If IMod1000=0Then '如果循环已完成了 1000 次。
        OpenForms=DoEvents   '将执行让给操作系统。
    EndIf
Next I  '将循环计数器加一。
```

DoGetNewFileName 事件

每当在任何部件或工程上执行“另存为”操作时，无论是从“文件”菜单上手工执行的还是由程序执行的，该事件都发生。

应用于

FileControlEvents 对象

语法

Sub DoGetNewFileName(*vbproject* **As VBProject ,** *filetype* **As vbext_FileTyp e ,** *newname* **As String ,** *oldname* **As String ,** *canceldefault* **As Boolean)**

DoGetNewFileName 事件语法包含下面部分：

部分	描述
vbproject	VBProject 对象，指出被写的工程名
filetype	数字值(vbext_FileType)，指出被写文件的类型，如“设置值”中列表所示
newname	字符串表达式，指出新的文件名，该文件规格必须与当前的 LastUsedPath 属性相关，或者是完全限定的文件名
oldname	字符串表达式，指出文件的旧名字
canceldefault	布尔表达式，确定缺省的 VisualBasic 动作，如“设置值”中所描述

设置值

vbext_FileType 的数字值是：

常数	值	描述
vbext_ft_Form	0	文件类型是窗体。
vbext_ft_Module	1	文件类型是基本模块。
vbext_ft_Class	2	文件类型是类模块。
vbext_ft_Project	3	文件类型是工程。
vbext_ft_Exe	4	文件类型是可执行文件。
vbext_ft_Res	6	文件类型是资源文件。
vbext_ft_UserControl	7	文件类型是用户控件。
vbext_ft_PropertyPage	8	文件类型是属性页。
vbext_ft_DocObject	9	文件类型是用户文档。
vbext_ft_Binary	10	文件类型是二进制文件。
vbext_ft_GroupProject	11	文件类型是工程组。
vbext_ft_Designer	12	文件类型是设计器对象。

canceldefault **的设计值是：**

设置值	描述
true	对任何后续的连接到 FileControl 对象的外接程序停止触发该事件。如果当 canceldefault 设置为 True 时 newname 是零字符串("")，则事件被取消；反之，输入到 newname 的名字做为新的文件名。
false	对后续的连接到 FileControl 对象的外接程序继续触发该事件。如果没有外接程序将 canceldefault 设置为 True，则“文件另存为”或“生成 Exe”对话框就用输入到选定的 newname 中的字符串来显示。

说明

如果 canceldefault 参数设置为 True，则不显示“文件另存为”对话框。如果 canceldefault 设置为 False，则显示“文件另存为”对话框。如果不止一个外接程序被连接，且 canceldefault 在“另存为”操作期间都置为 True，则“文件另存为”对话框对任何外接程序都不显示，直至下一个“另存为”操作被执行。

newname 参数最初设置为与 oldname 相同的值，但接受该事件的任何外接程序都能改变它。为此一种可用的方法是，通过定制的用户界面获得新的文件名并设置 newname 为用户的选择。然而，如果 canceldefault 是 True（意味着先前的外接程序设置为 True），就不应当再次设置 newname。

该事件发生在所有连接到 FileControl 对象的外接程序中。外接程

序不能阻止文件写盘，因为写盘操作已经完成。然而，可以用该事件执行其它任务，如：

?记录关于该事件的信息。

?更新有关该文件的信息。

?备份该文件。

Double 数据类型

Double（双精度浮点型）变量存储为 IEEE64 位（8 个字节）浮点数值的形式，它的范围在负数的时候是从-1.79769313486232E308 到-4.94065645841247E-324，而正数的时候是从 4.94065645841247E-324 到 1.79769313486232E308。Double 的类型声明字符是数字符号(#)。

请参阅

TypeConversion 函数,DataType 总结,SingleData 类型,Deftype 语句，ConversionKeyword 总结

DoVerb 方法

打开一个对象，用于进行诸如编辑那样的操作。不支持命名的参数。

应用于

OLE 容器控件，OLEObject 对象

语法

object.**DoVerb**(*verb*)

DoVerb 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
verb	可选的。在 OLE 容器控件内要执行的对象的谓词。如不指定，就执行缺省谓词。这个参数的值可以是一个所有对象都能支持的标准谓词，也可以是 ObjectVerbs 属性数组的一个索引

说明

将 AutoActivate 属性设置为 2（双击），当双击对象时，OLE 容器控件将当前对象自动激活。

每个对象都有它自己支持的谓词集合。下面各值表示的是所有对象都能支持的标准谓词：

常数	值	描述
vbOLEPrimary	0	对象的缺省动作。
vbOLEShow	-1	激活对象进行编辑。如果创建对象的应用程序支持现场激活，该对象在 OLE 容器控件内激活。
vbOLEOpen	-2	在分隔的应用程序窗口打开对象。如果创建对象的应用程序支持现场激活，该对象在其自己的窗口中激活。
vbOLEHide	-3	对于嵌入的对象，隐藏创建该对象的应用程序。
vbOLEUIActivate	-4	如果对象支持现场激活，则将该对象激活为现场激活，并显示所有的用户接口工具。如果对象不支持现场激活，则不激活对象，并产生一个错误。
vbOLEInPlaceActivate	-5	如果将焦点移到 OLE 容器控件，为对象创建一个窗口，并为对象作好编辑的准备。如果对象不支持单击鼠标的激活，则产生一个错误。
vbOLEDiscardUndoState	-6	当激活对象进行编辑时，用于放弃所有改变的记录，这些改变可由对象的应用程序撤消。

注意这些谓词也许未列在 ObjectVerbs 属性数组中。

请参阅

OvjectVerbs 属性，Verb 属性，AutoActivate 属性

DownPicture 属性

返回或设置一个对图片的引用，该图片在控件被单击并处于压下状态时显示在控件中。

应用于

CheckBox 控件，CommandButton 控件，OptionButton 控件

语法

object.**DownPicture** [=*picture*]

DownPicture 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
picture	一个包含图形的 Picture 对象，“设置值”中有详细的描述

设置值

picture 的设置值为：

设置值	描述
(None)	（缺省）没有图片。
(Bitmap,icon,metafile)	指定一个图形。在设计时可以从属性窗口加载该图形。在运行时，也可以通过在位图、图标或元文件上使用 LoadPicture 函数，或通过将其设置到别的 Picture 属性上来设置这个属性。

说明

DownPicture 属性指的是按钮处于被压下状态时显示的一个图片对象。除非控件的 Style 属性设置为 1（图形的），否则 DownPicture 属性将被忽略。值得注意的是，当 OptionButton 或 CheckBox 控件的 Style 属性被设置为图形方式并且其按钮被压下时，该按钮的背景将是抖动的，但按钮上的图片却不这样。

图片在控件上位于水平和垂直位置的中央。如果与图片一起使用标题，那么图片将位于标题上面的中央处。如果在按钮被压下时没有图片赋值给 DownPicture 属性，那么将显示当前被赋值给 Picture 属性的图片。如果既没有图片赋值给 Picture 属性也没有图片赋值给 DownPicture 属性，则只显示标题。如果图片对象太大而超出按钮边框，那么它将被裁剪一部分。

请参阅

DisabledPicture 属性，Enabled 属性，Picture 属性，Sytle 属性

Drag 方法

用于除了 Line、Menu、Shape、Timer 或 CommonDialog 控件之外的任何控件的开始、结束或取消拖动操作。不支持命名参数。

应用于

ADODate 控件，TreeView 控件，ImageCombo 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，CoolBar 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，TextBox 控件，MaskedEdit 控件，MSHFlexGrid 控件，MSFLexGrid 控件，RemoteData 控件，RichTextBox 控件，Data 控件，CheckBox 控件，FileListBox 控件，Frame 控件，HscrollBar,VscrollBar 控件，Image 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 容器控件

语法

object.**Drag***action*

Drag 方法的语法包含下列部分：

部分	描述
object	必需的。是一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 object，则认为该对象事件过程包含有 Drag 方法
action	可选的。是一个常数或数值，如“设置值”中所描述的，它指定要执行的动作。如果省略 action，则缺省值为开始拖动对象

设置值

action 的设置值有：

常数	值	描述
vbCancel	0	取消拖动操作
vbBeginDrag	1	开始拖动 *object*
vbEndDrag	2	结束拖放 *object*

说明

上述常数在 VisualBasic(VB)对象浏览器的对象库里列出。

只有当对象的 DragMode 属性设置为手工（0）时，才需要使用 Drag 方法控制拖放操作。但是，也可以对 DragMode 属性设置为自动（1 或 vbAutomatic）的对象使用 Drag。

如果在拖动对象过程中想改变鼠标指针形状，使用 DragIcon 或 MousePointer 属性。如果没有指定 DragIcon 属性，则只能使用 MousePointer 属性。

Drag 方法一般是同步的，这意味着其后的语句直到拖动操作完成之后才执行。然而，如果该控件的 DragMode 属性设置为 Manual(0 or vbManual)，则它可以异步执行。

请参阅

DragDrop 事件，DragOver 事件，DragIcon 属性，DragMode 属性，MousePointer 属性

示例

本示例使用 Drag 方法将一个位图(bmp)文件的文件名拖到显示该位图的图片框中。要检验此示例，可将本例所有代码粘贴到一个含有 DriveListBox、DirListBox、FileListBox、PictureBox 和 Label 控件的窗体的声明部分。对所有控件要使用缺省名。所有控件大小和位置的确定应使它们易于被看到和使用。标签的大小和位置并不重要，因为它们在运行时会改变。当程序开始时，可以浏览文件系统并装载任何需要的位图。装入所需要显示的位图后，可单击该位图的文件名并将它拖到图片框中。

```
PrivateSubForm_Load()
   Picture1.AutoSize=-1'打开 AutoSize。
   Label1.Visible=0'使该标签不可见。
   File1.Pattern="*.BMP;*.ICO;*.WMF"    '设置文件的各种样式。
EndSub

PrivateSubDir1_Change()    '任何 Dir1 中的变更
   File1.Path=Dir1.Path'都显示在 file1 中。
```

```
EndSub

PrivateSubDrive1_Change()  '任何Drive1中的变更
   Dir1.Path=Drive1.Drive   '都显示在Dir1中。
EndSub

PrivateSubFile1_MouseDown(ButtonAsInteger,ShiftAsInteger,XAsSing
le,YAsSingle)
   DimDY    '声明变量。
   DY=TextHeight("A")   '取得一行的高度。
   Label1.MoveFile1.Left,File1.Top+Y-DY/2,File1.Width,DY
   Label1.Drag '拖标签轮廓。
EndSub

PrivateSubDir1_DragOver(SourceAsControl,XAsSingle,YAsSingle,Stat
eAsInteger)
   '改变指针为不放。
   IfState=0ThenSource.MousePointer=12
   '使用缺省鼠标指针。
   IfState=1ThenSource.MousePointer=0
EndSub

PrivateSubDrive1_DragOver(SourceAsControl,XAsSingle,YAsSingle,St
```

```
ateAsInteger)
    '改变指针为不放。
    IfState=0ThenSource.MousePointer=12
    '使用缺省鼠标指针。
    IfState=1ThenSource.MousePointer=0
EndSub

PrivateSubForm_DragOver(SourceAsControl,XAsSingle,YAsSingle,Stat
eAsInteger)
    '改变指针为不放。
    IfState=0ThenSource.MousePointer=12
    '使用缺省鼠标指针。
    IfState=1ThenSource.MousePointer=0
EndSub

PrivateSubFile1_DragOver(SourceAsControl,XAsSingle,YAsSingle,Sta
teAsInteger)
    OnErrorResumeNext
    IfState=0AndRight$(File1.Filename,4)=".ICO"Then

    Label1.DragIcon=LoadPicture(File1.Path+"\"+File1.Filename)
    IfErrThenMsgBox"Theiconfilecan'tbeloaded."
    ElseIfState=1Then
```

```
        Label1.DragIcon=LoadPicture()    '使用非拖式图标。
    EndIf
EndSub

PrivateSubPicture1_DragDrop(SourceAsControl,XAsSingle,YAsSingle)
    OnErrorResumeNext
    Picture1.Picture=LoadPicture(File1.Path+"\"+File1.Filename)
    IfErrThenMsgBox"Thepicturefilecan'tbeloaded."
EndSub
```

DragDrop 事件

在一个完整的拖放动作（即将一个控件拖动到一个对象上，并释放鼠标按钮）完成，或使用 Drag 方法，并将其 action 参数被设置为 2(Drop)时，该事件发生。

应用于

ADODate 控件，TreeView 控件，ImageCombo 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，CoolBar 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，MSHFlexGrid 控件，MSFlexGrid 控件，RemoteData 控件，RichTextBox 控件，PropertyPage 控件，Use

rControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HscrollBar,VscrollBar 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 容器控件

语法

PrivateSubForm_DragDrop(*source***AsControl**,*x***AsSingle**,*y***AsSingle**)

PrivateSubMDIForm_DragDrop(*source***AsControl**,*x*AsSingle,*y***AsSingle**)

PrivateSub*object*_**DragDrop**([*index***AsInteger**,]*source***AsControl**,*x***AsSingle**,*y***AsSingle**)

DragDrop 事件语法包含下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件
source	正在被拖动的控件。可用此参数将属性和方法包括在事件过程中—例如，Source.Visible=0
x,*y*	是一个指定当前鼠标指针在目标窗体或控件中水平(x)和垂直(y)位置的数字。这些坐标值通常用目标坐标系统来表示，该坐标系是通过 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 属性而设置的

说明

DragDrop 事件过程用来控制在一个拖动操作完成时将会发生的情况。例如，可将源控件移到一个新的位置或将一个文件从一个位置复制到另一个位置。

当 source 参数中可能使用多个控件时：

应使用 TypeOf 关键字和 If 语句一起确定与 source 表示的控件的类型。

应使用该控件的 Tag 属性来标识一个控件，然后使用 DragDrop 事件过程。

注意应使用 DragMode 属性和 Drag 方法来指定开始拖动的方法。

一旦开始拖动，可使用 DragOver 事件过程来处理位于 DragDrop 事件前面的事件。

请参阅

DragOve 事件，MouseDown,MouseUp 事件，MouseMove 事件，Drag 方法，DragIcon 属性，DragMode 属性

示例

本例演示将一个 PictureBox 控件放到另一个 PictureBox 控件上的视觉效果。要尝试这个例子,可将代码粘贴到一个含有 3 个 PictureBox 控件的窗体声明部分。将 Picture1 和 Picture2 的 DragMode 属性设置为 1（自动）。使用 Picture 属性将位图赋值给 Picture1 和 Picture2,然后按 F5 键并将 Picture1 或 Picture2 拖到 Picture3 上。

```
PrivateSubPicture3_DragDrop(SourceAsControl,XasSingle,YAsSingle)
    IfTypeOfSourceIsPictureBoxThen
        '将 Picture3 位图设置为与源控件相同。
        Picture3.Picture=Source.Picture
    EndIf
EndSub
```

DragIcon 属性

返回或设置图标，它将在拖放操作中作为指针显示。

应用于

ADODate 控件，TreeView 控件，ImageCombo 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicke 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，CoolBar 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，MSChart 对象，MSHFlexGrid 控件，MSFlexGrid 控件，SSTab 控件，RemoteData 控件，RichTetBox 控件，Data 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Frame 控件，Hscroll Bar,VscrollBar 控件，Image 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 容器控件

语法

object.**DragIcon**[=*icon*]

DragIcon 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
con	任何返回图标的程序引用，例如引用窗体图标(Form1.Icon)，引用另外控件的 DragIcon 属性(Text1.DragIcon)，或是 LoadPicture 函数

设置值

icon 的设置值如下:

设置值	描述
(none)	（缺省值）矩形框内的箭头指针。
icon	自定义的鼠标指针。在设计时用属性窗口设置图标。运行时可以用 LoadPicture 函数。所加载的文件必须有.ico 文件扩展名和格式。

说明

在拖放操作中，用 DragIcon 属性可以提供可见的信息反馈—例如，它可以指定原控件在一个适当的目标之上。DragIcon 属性在用户开始拖放操作时起作用。典型做法是把 DragIcon 设置为 MouseDown 或 DragOver 事件过程的一部分。

注意运行时，DragIcon 属性可以设置为任何对象的 DragIcon 或 Icon 属性，或者可以用 LoadPicture 函数返回的图标给它赋值。

在运行中，把一个控件的 Picture 属性值赋给另一个控件的 DragIcon 属性时，Picture 属性必须包含一个 .ico 文件，而不是 .bmp 文件。

请参阅

DragDrop 事件，LoadPicture 函数，Drag 方法，DragMode 属性，MousePointer 属性

示例

每当拖动 PictureBox 控件时，这个例子改变 DragIcon 属性的设置值。要试用此例，先把代码粘贴到包含 PictureBox 控件的窗体的声明部分。并设置 DragMode 属性=1，然后按 F5 键，并单击和拖动 PictureBox 控件。

```
PrivateSubForm_DragDrop(SourceAsControl,XAsSingle,YAsSingle)
   DimPic  '声明变量.
   Source.MoveX,Y  '设置控件的位置.
   Pic="ICONS\OFFICE\CRDFLE01.ICO" '取图标文件的名字.
   IfSource.DragIcon=FalseThen '如果没有图片加载，
       Source.DragIcon=LoadPicture(Pic)'加载图片.
   Else
       Source.DragIcon=LoadPicture()   '不加载图片.
   EndIf
EndSub
```

DragMode 属性

返回或设置一个值，确定在拖放操作中所用的是手动还是自动拖动方式。

应用于

ADODate 控件，TreeView 控件，ImageCombo 控件，ListView 控件，PropgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，CoolBar 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，MSChart 对象，MSHFlexGrid 控件，MSFlexGrid 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Frame 控件，HscrollBar,VscrollBar 控件，Image 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 容器控件

语法

object.**DragMode**[=*number*]

DragMode 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	整数，指定拖动方式，“设置值”中有详细描述

设置值

number 的设置值如下：

常数	设置值	描述
vbManual	0	（缺省值）手动方式——需要在源控件中用 Drag 方法来启动拖放操作。
vbAutomatic	1	自动方式——单击源控件自动启动拖放操作。只有在 OLE 容器控件没有焦点时，它才能自动地被拖动。

说明

当 DragMode 属性设置为 1（自动方式）时，控件不能正常响应鼠标事件。可以用方式 0（手动方式）来确定拖放操作何时开始或何时结束；用该设置值可以启动一个拖放操作以响应键盘或菜单命令，或允许源控件在拖放操作之前识别 MouseDown 事件。

在拖放操作过程中，当鼠标器指针在一个目标对象或窗体上时，单击鼠标会对目标对象产生 DragDrop 事件。这将终止拖放操作。拖放操作也可能产生 DragOver 事件。

注意当拖动控件时，该控件不能识别用户发出的其它鼠标或键盘事件（KeyDown、KeyPress 或 KeyUp，MouseDown、MouseMove 或 MouseUp）。但是，控件可以接收程序或 DDE 链接启动的事件。

请参阅

DragDrop 事件，DragOver 事件，DragIcon 属性

示例

每次单击窗体时，这个例子使拖动 CommandButton 控件的操作有效和无效。要试用此例，先把代码粘贴到包含 CommandButton 的窗体的声明部分，然后按 F5 键并单击窗体。

```
PrivateSubForm_Click()
'检查 DragMode.
   IfCommand1.DragMode=vbManualThen
      '打开 DragMode.
Command1.DragMode=vbAutomatic
   Else
      '或关闭 DragMode.
   Command1.DragMode=vbManual
   EndIf
EndSub
```

DragOver 事件

它在拖放操作正在进行时发生。可使用此事件对鼠标指针在一个有效目标上的进入、离开或停顿等进行监控。鼠标指针的位置决定接收此事件的目标对象。

应用于

ADODate 控件，TreeView 控件，ImageCombo 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，Animation 控件，

FlatScrollBar 控件，MonthView 控件，UpDown 控件，CoolBar 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，SHFlexGrid 控件，MSFlexGrid 控件，RemoteData 控件，RichTextBox 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HscrollBar，VscrollBar 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 容器控件

语法

PrivateSubForm_DragOver(*source***AsControl**, *x***AsSingle**, *y***AsSingle**, *state* **AsInteger**)

PrivateSubMDIForm_DragOver(*source***AsControl**, *x***AsSingle**, *y***AsSingle**, *state***AsInteger**)

PrivateSub*object*_DragOver([*index***AsInteger**,]*source***AsControl**, *x***AsSingle**, *y***AsSingle**, *state***AsInteger**)

DragOver 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件
source	正在被拖动的控件。可用此参数在事件过程中引用各属性和方法—例如，Source.Visible=False
x,y	是一个指定当前鼠标指针在目标窗体或控件中水平(x)和垂直(y)位置的数字。这些坐标值通常用目标坐标系统来表示，该坐标系是通过 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 属性而设置的
state	是一个整数，它相应于一个控件的转变状态，该控件在相关目标窗体或控件中正在被拖动： 0=进入（源控件正被向一个目标范围内拖动） 1=离去（源控件正被向一个目标范围外拖动） 2=跨越（源控件在目标范围内从一个位置移到了另一位置）

说明

为了确定在拖动开始后和控件放在目标上之前发生些什么，应使用 DragOver 事件过程。例如，通过加亮目标（由代码设置 BackColor 或 ForeColor 属性）或者显示一个特定的拖动指针（由代码设置 DragIcon 或 MousePointer 属性）可验证有效的目标范围。

为了确定一些关键转变点处的操作，应使用 state 参数。例如，

当 state 的设置为 0（进入）时可使一个可能的目标加亮，而当 state 的设置为 1（离去）时可恢复该对象先前的外观。

在 state 的设置为 0（进入）对象接收 DragOver 事件的情况：

如果源控件被放在该对象上，则该对象就接收一个 DragDrop 事件。

如果源控件没有被放在该对象上，则当 state 的设置为 1（离去）时该对象就接收另一个 DragOver 事件。

注意应使用 DragMode 属性和 Drag 方法指定开始拖动的方式。关于 source 参数的使用技巧，请参阅 DragDrop 事件说明主题部分。

请参阅

DragDrop 事件，MouseDown，MouseUp 事件，MouseMove 事件，Drag 方法，DragIcon 属性，DragMode 属性

示例

本例演示一种指示有效的拖放目标的方法。当一个 TextBox 控件被拖过一个 PictureBox 控件时，指针从缺省的箭头变为特定的图标。当源被拖到其它地方时，指针恢复到缺省的状态。要尝试这个例子，可将代码粘贴到一个包含 1 个小 TextBox 和一个 PictureBox 的窗体的声明部分。将 TextBox 控件的 DragMode 属性设置为 1，然后按 F5 键并把 TextBox 拖过 PictureBox。

```
PrivateSubPicture1_DragOver(SourceAsControl,XAsSingle,YAsSingle,
StateAsInteger)
    SelectCaseState
        CasevbEnter
```

```
    '装载图标。

    Source.DragIcon=LoadPicture("ICONS\ARROWS\POINT03.ICO")
        CasevbLeave
            Source.DragIcon=LoadPicture()    '卸载图标.
    EndSelect
EndSub

PrivateSubPicture1_DragDrop(SourceAsControl,XAsSingle,YAsSingle)
    Source.DragIcon=LoadPicture()    '卸载图标。
EndSub
```

DrawMode 属性

返回或设置一个值，以决定图形方法的输出外观或者 Shape 及 Line 控件的外观。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，Printers 集合，Form 对象，Forms 集合，Line 控件，PictureBox 控件，Shape 控件

语法

object.**DrawMode**[=*number*]

DrawMode 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	整型值，指定外观，“设置值”中有详细描述

设置值

number 的设置值为：

常数	设置值	描述
vbBlackness	1	黑色。
vbNotMergePen	2	非或笔－与设置值 15 相反(MergePen)。
vbMaskNotPen	3	与非笔－背景色以及画笔反相二者共有颜色的组合。
vbNotCopyPen	4	非复制笔－设置值 13(CopyPen)的反相
vbMaskPenNot	5	与笔非－画笔以及显示反相二者共有颜色的组合。
vbInvert	6	反转－显示颜色的反相
vbXorPen	7	异或笔－画笔的颜色以及显示颜色的组合，只取其一。
vbNotMaskPen	8	非与笔－设置值 9(MaskPen)的反相。
vbMaskPen	9	与笔－画笔和显示二者共有颜色的组合。
vbNotXorPen	10	非异或笔－方式 7 的反相(XorPen)。
vbNop	11	无操作－输出保持不变。该设置实际上

		关闭画图。
vbMergeNotPen	12	或非笔－显示颜色与画笔颜色反相的组合。
vbCopyPen	13	复制笔（缺省值）－由 ForeColor 属性指定的颜色。
vbMergePenNot	14	或笔非－画笔颜色与显示颜色的反相的组合。
vbMergePen	15	或笔－画笔颜色与显示颜色的组合。
vbWhiteness	16	白色。

说明

当用 Shape 或 Line 控件，或者用图形方法画图时，使用这个属性产生可视效果。VisualBasic 将绘图模式的每一个像素与现存背景色中相应的像素做比较，然后进行逐位比较操作。例如，设置值 7（异或笔）用 Xor 操作符将绘图模式像素和背景像素组合起来。

DrawMode 设置值的真正效果，取决于运行时所画线的颜色与屏幕已存在颜色的合成。对于设置值 1，6，7，11，13 和 16 可以最可靠地预知该属性的输出结果。

请参阅

BackColor,ForeColor 属性，DrawWidth 属性，DrawStyle 属性，FillColor 属性，FillStyle 属性，BackColor,ForeColor 属性（ActiveX 控件）

示例

这个例子允许通过拖动鼠标在窗体上画图。每单击一次鼠标为 DrawMode 属性设置一个不同的值。要试用此例，先把代码粘贴到窗体的声明部分，然后按 F5 键，并单击窗体。

```
PrivateSubForm_Load
    DrawWidth=10'设置 DrawWidth.
EndSub
PrivateSubForm_Click()
    StaticMAsInteger'当前 DrawMode 的设置值.
    ForeColor=QBColor(Int(Rnd*15))    '选择一种颜色.
    M=((M+1)Mod16)+1'使 DrawMode 小于或等于 16.
    DrawMode=M    '设置 DrawMode.
EndSub
PrivateSubForm_MouseMove(ButtonAsInteger,ShiftAsInteger,XAsSingl
e,YAsSingle)
    IfButtonThen'当按钮被按下时，
        PSet(X,Y)    '画一个大点.
    EndIf
EndSub
```

DrawStyle 属性

返回或设置一个值，以决定图形方法输出的线型的样式。

应用于

PropertyPage 对象，UseControl 对象，UserDocument 对象，Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**DrawStyle**[=*number*]

DrawStyle 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	整数，指定线型，“设置值”中有详细描述

设置值

number 的设置值为：

常数	设置值	描述
vbSolid	0	（缺省值）实线
vbDash	1	虚线
vbDot	2	点线
vbDashDot	3	点划线
vbDashDotDot	4	双点划线
vbInvisible	5	无线
vbInsideSolid	6	内收实线

说明

若 DrawWidth 属性设置为大于 1，DrawStyle 属性设置值为 1 到 4

会画一条实线（DrawStyle 属性值不改变）。若 DrawWidth 设置为 1，DrawStyle 产生的效果如前面表中的各设置值所述。

请参阅

BackColor,ForeColor 属性，DrawWidth 属性，DrawMode 属性，FillColor 属性，FillStyle 属性，BackColor,ForeColor 属性（ActiveX 控件）

示例

这个例子是穿过窗体画七条线，每条线显示不同的 DrawStyle 属性(如果设置 AutoRedraw=True，每当调整窗体的大小并单击窗体时，窗体便累加一新的线集。)。要试用此例，先把代码粘贴到窗体的声明部分，然后按 F5 键，并单击窗体。

```
PrivateSubForm_Click()
   DimI '声明变量.
   ScaleHeight=8    '用 8 除高.
   ForI=0To6
        DrawStyle=I  '改变线形.
        Line(0,I+1)-(ScaleWidth,I+1) '画新线.
   NextI
EndSub
```

DrawWidth 属性

返回或设置图形方法输出的线宽。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**DrawWidth**[=*size*]

DrawWidth 属性语法具有下列组成部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
size	数值表达式，其范围从 1 到 32,767。该值以像素为单位表示线宽。缺省值为 1，即，一个像素宽

说明

增大该属性值会增加线的宽度。如果 DrawWidth 属性值大于 1，DrawStyle 属性值设置为 1 到 4 时会画出一条实线来（DrawStyle 属性值不会改变）。将 DrawWidth 设置为 1，允许 DrawStyle 产生 DrawStyle 属性表中列出的结果。

请参阅

BackColor,ForeColor 属性，DrawMode 属性，DrawStyle 属性，FillColor 属性，FillStyle 属性，BackColor，ForeColor 属性（ActiveX 控件）

示例

这个例子在窗体中绘制一条逐渐加粗的直线。要尝试这个例子，

请将代码粘贴到窗体的声明部分，然后按 F5 键并单击窗体。

```
PrivateSubForm_Click()
    DimI'声明变量。
    DrawWidth=1 '设置笔的起始宽度。
    PSet(0,ScaleHeight/2)    '设置起始点。
    ForeColor=QBColor(5)'设置笔的颜色。
    ForI=1To100Step10    '建立一个循环。
        DrawWidth=I '重新设置笔的宽度。
        Line-Step(ScaleWidth/10,0)'绘制一条直线。
    NextI
EndSub
```

Drive 对象

对特定磁盘驱动器或网络共享的属性提供访问。

说明

下面的代码举例说明了用 Drive 对象来访问驱动器属性：

```
SubShowFreeSpace(drvPath)
Dimfs,d,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(fs.GetDriveName(drvPath))
s="Drive"&UCase(drvPath)&"-"
s=s&d.VolumeName&vbCrLf
```

```
s=s&"FreeSpace:"&FormatNumber(d.FreeSpace/1024,0)
s=s&"Kbytes"
MsgBoxs
EndSub
```

属性

AvailableSpace 属性，DriveLetter 属性，DriveType 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，Path 属性（FileSystemObject 对象），RootFolder 属性，SerialNumber 属性，ShareName 属性，TotalSize 属性，VolumeName 属性

请参阅

DriveExists 方法 Drives 集合，File 对象，Files 集合，FileSystemObject 对象，Folder 对象，Folder 集合

Drive 属性

返回或设置运行时选择的驱动器。在设计时不可用。

应用于

DriveListBox 控件

语法

object.**Drive**[=*drive*]

Drive 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
drive	字符串表达式，指定所选择的驱动器

说明

Drive 属性的有效驱动器包括：在运行中控件创建和刷新时系统已有的、或连接到系统上的所有驱动器。Drive 属性的缺省值为当前驱动器。

读该属性值时，按下述格式之一返回所选择的驱动器：

软磁盘—"a:"或"b:"，等等

固定介质—"c:[volumeid]"

网络连接—"x:\\server\share"

设置该属性时：

该字符串的第一个字符是有效字符（字符串不区别大小写）。

改变 Drive 属性的设置值会激活 Change 事件。

选择不存在的驱动器会产生错误。

设置该属性也会重新生成驱动器列表，从而提供了在程序中，跟踪创建控件后增加的网络连接的方法。

如果 FileName 属性设置为没有驱动器指示的限定网络路径，则 Drive 属性值是一个 0 长度字符串("")，没有选择驱动器，ListIndex 属性值为-1。

注意 Drive 属性的返回值与 ListIndex 属性值不同，ListIndex 属性返

回列表框中的选择。

请参阅

PathChange 事件,PatternChange 事件,ListIndex 属性,FileName 属性，Pattern 属性，Path 属性

示例

这个例子显示当前驱动器和目录的文件列表。要试用此例，先把代码粘贴到包含 DriveListBox 控件、DirListBox 控件和 FileListBox 控件的窗体的声明部分，然后按 F5 键。利用鼠标改变驱动器或目录。

```
PrivateSubDrive1_Change()
   Dir1.Path=Drive1.Drive   '当驱动器改变时，设置目录路径.
EndSub
PrivateSubDir1_Change()
   File1.Path=Dir1.Path'当目录改变时，设置文件路径.
EndSub
```

Drive 属性

返回指定文件或文件夹所在的驱动器符号。只读。

应用于

File 对象，Folder 对象

语法

object.Drive

object 总是一个 File 或 Folder 对象。

说明

下面的代码举例说明了 Drive 属性的用法：

```
SubShowFileAccessInfo(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFile(filespec)
s=f.Name&"onDrive"&UCase(f.Drive)&vbCrLf
s=s&"Created:"&f.DateCreated&vbCrLf
s=s&"LastAccessed:"&f.DateLastAccessed&vbCrLf
s=s&"LastModified:"&f.DateLastModified
MsgBoxs,0,"FileAccessInfo"
EndSub
```

请参阅

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，IsRootFolder 属性，Name 属性（FileSystemObject 对象），ParentFolder 属性，Path 属性（FileSystemObject 对象），ShortPath 属性，Size 属性（FileSystemObject 对象），SubFolders 属性，Type 属性

DriveExists 方法

如果指定的驱动器存在，返回 True，如果不存在返回 False。

应用于

FileSystemObject 对象

语法

object.**DriveExists**(*drivespec*)

DriveExists **方法语法有如下几部分：**

部分	**描述**
object	**必需的。始终是一个** FileSystemObject **的名字**
drivespec	**必需的。一个驱动器字符或一个完整的路径说明**

说明

对于可删除介质的驱动器，即使没有介质存在，DriveExists **方法也返回** True**。使用** Drive **对象的** IsReady **属性确定驱动器是否准备好。**

DriveLetter 属性

返回某个物理本地驱动器或网络共享的驱动器字母。只读。

应用于

Drive 对象

语法

object.**DriveLetter**

object **总是一个** Drive **对象。**

说明

如果指定的驱动器没有同某个驱动器字母关联起来，例如，未被映射到一个驱动器字母的网络共享，则 DriveLetter 属性返回一个 0 字节长度的字符串("")。

下面的代码举例说明了 DriveLetter 属性的用法：

```
SubShowDriveLetter(drvPath)
Dimfs,d,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(fs.GetDriveName(drvPath))
s="Drive"&d.DriveLetter&":-"
s=s&d.VolumeName&vbCrLf
s=s&"FreeSpace:"&FormatNumber(d.FreeSpace/1024,0)
s=s&"Kbytes"
MsgBoxs
EndSub
```

请参阅

AvailableSpace 属性，DriveType 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，Path 属性 (FileSystemObject 对象)，RootFolder 属性，SerialNumber 属性，ShareName 属性，ShortName 属性，TotalSize 属性，VolumeName 属性

DriveListBox 控件

在运行时，由于有 DriveListBox 控件，所以可选择一个有效的磁

盘驱动器。该控件用来显示用户系统中所有有效磁盘驱动器的列表。可以创建对话框，通过它从任一可用驱动器的磁盘文件列表中打开文件。

语法

DriveListBox

说明

设置 List、ListCount、和 ListIndex 属性，就可以访问列表中的项目。如果需要显示 DirListBox 和 FileListBox 控件，那么可以编写代码，使它们与 DriveListBox 控件同步，并使它们之间彼此同步。

属性

OLEDropMode 属性，BackColor,ForeColor 属性，FontBold,FontItalic,FontStrikethru,FontUnderline 属性，FontName 属性，FontSize 属性，Height,Width 属性，Left,Top 属性，List 属性，ListCount 属性，ListIndex 属性，TabIndex 属性，Tag 属性，Visible 属性，DragIcon 属性，DragMode 属性，Drive 属性，hWnd 属性，MouseIcon 属性，MousePointer 属性，TabStop 属性，TopIndwx 属性，Appearance 属性，Enabled 属性，HelpContextID 属性，Index 属性（ControlArray），Name 属性，Parent 属性，Font 属性,Container 属性,ToolTipText 属性,WhatsThisHelpID 属性

方法

Refresh 方法，SetFocus 方法，Drag 方法，Move 方法，Zorder

方法，OLEDrag 方法，ShowWhatsThis 方法

事件

Change 事件，DragDrop 事件，DragOver 事件，GotFocus 事件，KeyDown,KeyUp 事件，KeyPress 事件，LostFocus 事件，Scroll 事件，Validate 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件

请参阅

DirListBox 控件，FileListBox 控件，List 属性，ListCount 属性，ListIndex 属性

DriverName 属性

返回 Printer 对象的驱动器名。

应用于

Printer 对象，Printers 集合

语法

object.**DriverName**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

每个驱动程序都有其唯一的名称。例如，几种 Hewlett-Packard 打印机的 DriverName 是 HPPCL5MS。DriverName 一般是去掉扩展名

的驱动程序文件名。

注意 Printer 对象属性的效果，取决于打印机生产商提供的驱动程序。一些属性设置可能不起作用，或几个不同的属性设置具有相同的结果。设置值超出可接受范围会产生错误。有关进一步信息，参阅有关驱动程序的由生产商提供的文档。

请参阅

DeviceName 属性

Drives 集合

所有可用驱动器的只读集合。

说明

对于可删除的驱动器，不需要将媒体插入其中，它就可以在 Drives 集合中显示出来。

下面的代码举例说明了如何获得 Drives 集合，以及如何用 ForEach...Next 语句来访问该集合中的每个 Drive：

```
SubShowDriveList
Dimfs,d,dc,s,n
Setfs=CreateObject("Scripting.FileSystemObject")
Setdc=fs.Drives
ForEachdindc
s=s&d.DriveLetter&"-"
Ifd.DriveType=RemoteThen
```

```
n=d.ShareName
Else
n=d.VolumeName
EndIf
s=s&n&vbCrLf
Next
MsgBoxs
EndSub
```

属性

Count 属性（FileSystemObject 对象），Item 属性

请参阅

DriveExists 方法，Drive 对象，File 对象，Files 集合，FileSystemObject 对象，Folder 对象，Folders 集合，Drives 属性

Drives 属性

返回包含本地机器上所有可用 Drive 对象的 Drives 集合。

应用于

FileSystemObject 对象

语法

object.**Drives**

object 总是一个 **FileSystemObject**。

说明

对于可删除媒体驱动器来说，不需要插入媒体，就可使其出现在 Drives 集合中。

可以用 ForEach...Next 结构遍及 Drives 集合中的成员，如下面的代码所示：

```
SubShowDriveList
Dimfs,d,dc,s,n
Setfs=CreateObject("Scripting.FileSystemObject")
Setdc=fs.Drives
ForEachdindc
s=s&d.DriveLetter&"-"
Ifd.DriveType=3Then
n=d.ShareName
Else
n=d.VolumeName
EndIf
s=s&n&vbCrLf
Next
MsgBoxs
EndSub
```

请参阅

Drives 集合，Files 属性

DriveType 属性

返回一个值，表示指定驱动器的类型。

应用于

Drive 对象

语法

object.**DriveType**

object 总是一个 Drive 对象。

说明

下面的代码举例说明了 DriveType 属性的用法：

```
SubShowDriveType(drvpath)
Dimfs,d,s,t
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(drvpath)
SelectCased.DriveType
Case0:t="Unknown"
Case1:t="Removable"
Case2:t="Fixed"
Case3:t="Network"
Case4:t="CD-ROM"
Case5:t="RAMDisk"
EndSelect
s="Drive"&d.DriveLetter&":-"&t
```

MsgBoxs

EndSub

请参阅

AvailableSpace 属性，DriveLetter 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，Path 属性（FileSystemObject 对象），RootFolder 属性，SerialNumber 属性，ShareName 属性，TotalSize 属性，VolumeName 属性

DropDown 事件

该事件是当 ComboBox 控件的列表部分正要被放下时发生；如果 ComboBox 控件的 Style 属性设置为 1（简单的 Combo）时此事件不会发生。

应用于

ComboBox 控件

语法

PrivateSub*object*_**DropDown**(**[***index***AsInteger]**)

DropDown 事件的语法包含下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件

说明

DropDown 事件过程可以用在接受选择之前对 ComboBox 列表进行最后的各种更新。于是允许使用 AddItem 或 RemoveItem 方法从该列表添加或删除条目。在需要控件间有某些相互作用时，这种灵活性是很有用的，例如，需要根据在 OptionButton 组中的选择才能决定加载到 ComboBox 列表的内容时，就可以利用该事件。

请参阅

Style 属性

示例

本例根据用户在一个选项按钮组中的选择来更新一个 ComboBox 控件。要尝试这个例子，可将代码粘贴到一个包含 1 个 ComboBox 控件和 2 个 OptionButton 控件的窗体的声明部分。将两个 OptionButton 控件的 Name 属性都设置为 OptionGroup，然后按 F5 键并单击 OptionButton 控件。根据 OptionButton 的选择，ComboBox 控件反映出不同的载体。

```
PrivateSubForm_Load()
   Combo1.Text=""   '清除组合框。
EndSub

PrivateSubCombo1_DropDown()
   Combo1.Clear                '删除已有的项。
   IfOptionGroup(0).Value=TrueThen
      Combo1.AddItem"GrayGooseExpress",0
      Combo1.AddItem"WildFargoCarriers",1
```

```
    Else
        Combo1.AddItem"SummitTechnologiesOvernight"
    EndIf
EndSub
```

Duplex 属性

返回或设置一个值，以决定是否要双面打印（若打印机支持该功能）。在设计时不可用。

应用于

Printer 对象，Printers 集合

语法

object.**Duplex**[=*value*]

Duplex 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	值或常数，指定打印类型，“设置值”中有详细描述

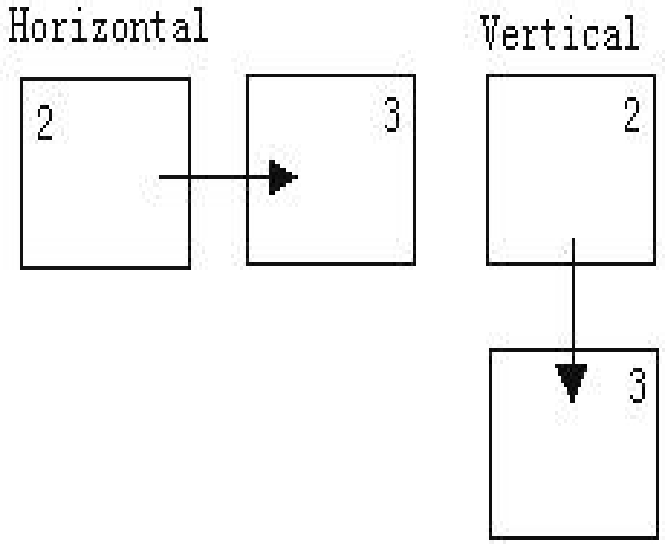
设置值

value 的设置值为:

常数	值	描述
VbPRDPSimplex	1	按当前设置的方向单面打印。
VbPRDPHorizontal	2	用水平页面旋转格式双面打印。
VbPRDPVertical	3	用垂直页面旋转格式双面打印。

说明

对于水平双面打印，每页两面的顶部在纸的同一端．而垂直双面打印，一页的底部和另一页的顶端在纸的同一端。下面是水平和竖直双面打印的示意图：



注意 Printer 对象属性的效果，取决于打印机生产商提供的驱动程序。一些属性设置可能不起作用，或几个不同的属性设置产生相同的结果。如果设置值超出可接受范围，则产生错误。有关进一步信息，请参阅有关驱动程序的由生产商提供的文档。

EditAtDesignTime 属性

返回或设置一个值，该值决定控件在设计时是否能变为活动的。在创建控件时，EditAtDesignTime 属性可读可写，在控件运行时，该属性不可用。

应用于

UserControl 对象

设置值

EditAtDesignTime 的设置值为：

设置值	描述
true	允许控件在开发者设计时变为活动的。“编辑”动词将出现在控件的上下文菜单中。选择“编辑”命令后，控件将变为活动的，其表现就象在最终用户运行时一样。
false	（缺省）控件在设计时不能变为活动的。

说明

仅当选定控件时，控件才保持活动状态。选定控件后，即使再次单击控件，该控件也不再处于活动状态。这时，必须再次从上下文菜单中选择“编辑”命令激活控件。

注意当用这种方法激活控件时，UserControl 对象将产生事件，以使控件

能正常操作，但是控件将不能产生任何事件。此时只忽略 RaiseEvent 方法

并不会产生错误。

EditProperty 事件

由于按下省略按钮显示要编辑的某个属性，而打开属性页时，发生该事件。

应用于

PropertyPage 对象

语法

Sub*object*_**EditProperty**(*PropertyName* **As String**)

EditProperty 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的一个对象
propertyName	字符串，它标识由属性页显示和编辑的属性

说明

当通过“属性”对话框将一个属性页赋予某属性时，发生该事件。通过“属性”对话框赋值属性页意味着：在“属性”窗口中，属性旁显示一个省略号(　，可以按下这个省略号按钮，属性页将自动打开；此时将产生 EditProperty 事件，以使属性页创建者能将游标放到正确的字段上。

Empty

Empty 关键字是用作 Variant 子类型。它表示未初始化的变量值。

请参阅

VarType 函数，Variant 数据类型

Enabled 属性

返回或设置一个值，该值用来确定一个窗体或控件是否能够对用户产生的事件作出反应。

应用于

UserControl 对象,CheckBox 控件,ComboBox 控件,CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件,Form 对象,Forms 集合,Frame 控件,HScrollBar,VScrollBar 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，Menu 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，Timer 控件，OLE 包容器控件，TreeView 控件，ButtonMenu 对象,ImageCombo 控件,ComboItem 对象,ListView 控件,Slider 控件，StatusBar 控件，Panel 对象，TabStrip 控件，Toolbar 控件,Button 对象,MonthView 控件,DataRepeater 控件,DataList 控件，RemoteData 控件，Data 控件

语法

object.**Enabled**[=*boolean*]

Enabled 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象。如果 object 被省略，则与活动窗体模块相联系的窗体被认为是 object
boolean	一个用来指定 object 是否能够对用户产生的事件作出反应的布尔表达式

设置

boolean 的设置为：

设置	描述
true	（缺省）允许 object 对事件作出反应
false	阻止 object 对事件作出反应

说明

Enabled 属性允许在运行时使窗体和控件成为有效或无效。例如，可以使对象成为不能用于应用程序的当前状态的无效状态。也可以使之纯粹用来显示，比如一个提供只读信息的文本框的控件无效。

通过把 Enabled 设置为 False 来使 Timer 控件成为无效，将取消由控件的 Interval 属性所建立的倒计数。

对于 Menu 控件，Enabled 在运行时可正常地读/写。但是对于那些被 VisualBasic 的加载宏遗弃或提供的菜单项来说，Enabled 是只读的，例如在“外接程序”菜单中的“外接程序管理器”命令。

请参阅

Interval 属性, Visible 属性, MultimediaMCI 控件, AutoEnable 属性(MultimediaMCI 控件),ButtonEnabled 属性(MultimediaMCI 控件)

示例

该例子使一个 CommandButton 控件有效而不管 TextBox 控件是否包含文本。要使用此例，先将下面的代码粘贴到带有 CommandButton 和 TextBox 控件的一个窗体的声明部分，然后按下 F5 键并在文本框中随意输入一些内容。

```
PrivateSubForm_Load()
   Text1.Text=""    '清除文本框的内容。
   Command1.Caption="Save" '在按钮上放置标题。
EndSub

PrivateSubText1_Change()
   IfText1.Text=""Then '查看文本框是否为空。
       Command1.Enabled=False  '使按钮无效。
   Else
       Command1.Enabled=True   '使按钮有效。
   EndIf
EndSub
```

End 语句

结束一个过程或块。

语法

End

EndFunction

EndIf

EndProperty

EndSelect

EndSub

EndType

EndWith

End **语句的语法有以下几种形式：**

语句	描述
End	停止执行。不是必要的，可以放在过程中的任何位置关闭代码执行、关闭以 Open 语句打开的文件并清除变量
EndFunction	必要的，用于结束一个 Function 语句
EndIf	必要的，用于结束一个 If…Then…Else 语句块
EndProperty	必要的，用于结束一个 PropertyLet、PropertyGet、或 PropertySet 过程
EndSelect	必要的，用于结束一个 SelectCase 语句
EndSub	必要的，用于结束一个 Sub 语句
EndType	必要的，用于结束一个用户定义类型的定义（Type 语句）
EndWith	必要的，用于结束一个 With 语句

说明

在执行时，End 语句会重置所有模块级别变量和所有模块的静态局部变量。若要保留这些变量的值，改为使用 Stop 语句，则可以在保留这些变量值的基础上恢复执行。

注意 End 语句不调用 Unload、QueryUnload、或 Terminate 事件或任何其它 VisualBasic 代码，只是生硬地终止代码执行。窗体和类模块中的 Unload、QueryUnload、和 Terminate 事件代码未被执行。类模块创建的对象被破坏，由 Open 语句打开的文件被

关闭，并且释放程序所占用的内存。其它程序的对象引用无效。End 语句提供了一种强迫中止程序的方法。VisualBasic 程序正常结束应该卸载所有的窗体。只要没有其它程序引用该程序公共类模块创建的对象并无代码执行，程序将立即关闭。

请参阅

Exit 语句，If...Then...Else 语句，SelectCase 语句，Stop 语句，With 语句，With 语句，Function 语句，PropertyGet 语句，PropertyLet 语句，PropertySet 语句,Sub 语句,类型语句

示例

本示例使用 End 语句，在用户输入错误密码时结束代码执行。

```
SubForm_Load
    DimPassword,Pword
    PassWord="Swordfish"
    Pword=InputBox("Typeinyourpassword")
    IfPword<>PassWordThen
        MsgBox"Sorry,incorrectpassword"
        End
    EndIf
EndSub
```

EndDoc 方法

用于终止发送给 Printer 对象的打印操作，将文档释放到打印设备

或后台打印程序。

应用于

Printer 对象，Printers 集合

语法

object.**EndDoc**

object 所在处代表一个对象表达式，其值为“应用于”列表中的一个对象。

说明

如果在运行 NewPage 方法后立即调用 EndDoc，则不会打印额外的空白页。

请参阅

KillDoc 方法，NewPage 方法

示例

本示例使用 EndDoc 方法在打印完两页之后结束一个文件，该被打印的每页按正文行居中方式显示页号。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimHWidth,HHeight,I,Msg          '声明变量。
   OnErrorGoToErrorHandler              '设置错误处理程序。
   Msg="Thisisprintedonpage"
   ForI=1To2                        '设置 2 个迭代。
      HWidth=Printer.TextWidth(Msg)/2  '取得半宽。
```

```
        HHeight=Printer.TextHeight(Msg)/2    '取得半高。
        Printer.CurrentX=Printer.ScaleWidth/2-HWidth
        Printer.CurrentY=Printer.ScaleHeight/2-HHeight
        Printer.PrintMsg&Printer.Page&"."    '打印。
        Printer.NewPage                      '发送新页。
    NextI
    Printer.EndDoc                           '打印完成。
    Msg="Twopages,eachwithasingle,centeredlineoftext,"
    Msg=Msg&"havebeensenttoyourprinter."
    MsgBoxMsg                                '显示信息。
    ExitSub
ErrorHandler:
    MsgBox"Therewasaproblemprintingtoyourprinter."
    ExitSub
EndSub
```

EndRequest 事件

在 WebClass 对象完成了对一个 HTTP 请求的处理并返回给用户一个响应时发生。

应用于

WebClass 对象

语法

PrivateSub*object*_EndRequest()

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

除非出现致命的错误，否则，对于一个给定的 HTTP 请求，EndRequest 是 WebClass 接收的最后一个事件。如果在 EndRequest 被激发之前，发生了致命的错误，EndRequest 事件将不会被激发。如果致命的错误发生在 EndRequest 事件中，FatalErrorResponse 事件将在 EndRequest 事件之后被激发。

请参阅

《MicrosoftVisualBasic6.0 部件工具参考手册》第 3 章“WebClass 生存周期”，第五部分“Internet 应用程序创建”的“IIS 应用程序开发”

EnterFocus 事件

当焦点进入对象时，发生该事件。对象本身可以接收焦点，或者子控件可以接收焦点。

语法

Sub*object*_EnterFocus()

EnterFocus 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

通过此事件，object 可以获知焦点是否在其内部。

EnterFocus 事件产生在任何 GotFocus 事件之前；GotFocus 事件仅产生在确实获得焦点的 object 或 object 的子控件中。

请参阅

ExitFocus 事件，GotFocus(UserControl 对象和 UserDocument 对象)

Enum 语句

定义枚举类型。

语法

[Public|Private]Enumname

membername[=constantexpression]

membername[=constantexpression]

...

EndEnum

Enum 语句包含下面部分：

部分	描述
Public	可选的。表示该 Enum 类型在整个工程中都是可见的 Enum 类型的缺省情况是 Public
private	可选的。表示该 Enum 类型只在所声明的模块中是可见的
Name	必需的。该 Enum 类型的名称。*name* 必须是一个合法的 VisualBasic 标识符，在定义该 *Enum* 类型的变量或参数时用该名称来指定类型
Membername	必需的。用于指定该 *Enum* 类型的组成元素名称的合法 VisualBasic 标识符
Constantexpression	可选的。元素的值（为 Long 类型）。可以是别的 Enum 类型。如果没有指定 *constantexpression*，则所赋给的值或者是 0（如果该元素是第一个 *membername*），或者比其前一个的值大 1

说明

所谓枚举变量，就是指用 Enum 类型定义的变量。变量和参数都可以定义为 Enum 类型。Enum 类型中的元素被初始化为 Enum 语句中指定的常数值。所赋给的值可以包括正数和负数，且在运行时不能改变。例如：

```
EnumSecurityLevel
   IllegalEntry=-1
   SecurityLevel1=0
   SecurityLevel2=1
```

```
EndEnum
```

Enum 语句只能在模块级别中出现。定义 Enum 类型后，就可以用它来定义变量，参数或返回该类型的过程。不能用模块名来限定 Enum 类型。类模块中的 PublicEnum 类型并不是该类的成员；只不过它们也被写入到类型库中。在标准模块中定义的 Enum 类型则不写到类型库中。具有相同名字的 PublicEnum 类型不能既在标准模块中定义，又在类模块中定义，因为它们共享相同的命名空间。若不同的类型库中有两个 Enum 类型的名字相同，但成员不同，则对这种类型的变量的引用，将取决于哪一个类型库具有更高的引用优先级。

不能在 With 块中使用 Enum 类型作为目标。

请参阅

Type 语句

示例

下面的示例显示用 Enum 语句定义一个命名常数的集合。在本例中，是一些用于设计数据库的数据输入窗体可以选择的颜色常数。

```
PublicEnumInterfaceColors
    icMistyRose=&HE1E4FF&
    icSlateGray=&H908070&
    icDodgerBlue=&HFF901E&
    icDeepSkyBlue=&HFFBF00&
```

```
    icSpringGreen=&H7FFF00&
    icForestGreen=&H228B22&
    icGoldenrod=&H20A5DA&
    icFirebrick=&H2222B2&
End Enum
```

Environ 函数

返回 String，它关连于一个操作系统环境变量。

语法

Environ({*envstring* | *number*})

Environ 函数的语法含有以下这些命名参数：

部分	描述
envstring	可选参数。包含一个环境变量名的字符串表达式
number	可选参数。数值表达式，用来表示环境字符串在环境字符串表格中的数值顺序。*number* 参数可以是任意的数值表达式，不过在计算前，它会先转换为一个整数

说明

如果在环境字符串表格中找不到 envstring，**则会返回一个零长度字符串("")。如果找到，则** Environ **会返回一段文本，文本是赋值给指定的** envstring **的，也就是说，在环境字符串表格中对应那**

个环境变量的等号(=)后面的那段文本。

如果指定了 number，则在环境字符串表格中相应位置上的字符串会返回。在这种情况下，Environ 会返回整个文本，包括 envstring。如果在指定位置上没有环境字符串，那么 Environ 会返回一个零长度字符串。

示例

本示例使用 Environ 函数来提供来自环境变量表中 PATH 语句的长度及路径入号。

```
DimEnvString,Indx,Msg,PathLen          '声明变量。
Indx=1                                 '设置索引值的初值为 1。
Do
   EnvString=Environ(Indx)             '取得环境变量。
   IfLeft(EnvString,5)="PATH="Then     '检查 PATH 项。
      PathLen=Len(Environ("PATH"))     '取得长度。
      Msg="PATHentry="&Indx&"andlength="&PathLen
      ExitDo
   Else
      Indx=Indx+1 '不是 PATH 项，
   EndIf                               '则跳过此项，继续检查下一项。
LoopUntilEnvString=""
IfPathLen>0Then
   MsgBoxMsg                           '显示消息。
Else
```

```
    MsgBox"NoPATHenvironmentvariableexists."
EndIf
```

EOF 函数

返回一个 Integer，它包含 Boolean 值 True，表明已经到达为 Random 或顺序 Input 打开的文件的结尾。

语法

EOF(*filenumber*)

必要的 *filenumber* 参数是一个 Integer，包含任何有效的文件号。

说明

使用 EOF 是为了避免因试图在文件结尾处进行输入而产生的错误。

当到达文件的结尾时，EOF 函数返回 False。对于为访问 Random 或 Binary 而打开的文件，直到最后一次执行的 Get 语句无法读出完整的记录时，EOF 都返回 False。

对于为访问 Binary 而打开的文件，在 EOF 函数返回 True 之前，试图使用 Input 函数读出整个文件的任何尝试都会导致错误发生。在用 Input 函数读出二进制文件时，要用 LOF 和 Loc 函数来替换 EOF 函数，或者将 Get 函数与 EOF 函数配合使用。对于为 Output 打开的文件，EOF 总是返回 True。

请参阅

Get 语句，Loc 函数，LOF 函数，Open 语句

示例

本示例使用 EOF 函数来检测文件尾。示例中假设 MYFILE 为有数个文本行的文本文件。

```
DimInputData
Open"MYFILE"ForInputAs#1    '为输入打开文件。
DoWhileNotEOF(1)            '检查文件尾。
   LineInput#1,InputData        '读入一行数据。
   Debug.PrintInputData         '在立即窗口中显示。
Loop
Close#1                         '关闭文件。
```

Erase 语句

重新初始化大小固定的数组的元素，以及释放动态数组的存储空间。

语法

Erase*arraylist*

所需的 *arraylist* 参数是一个或多个用逗号隔开的需要清除的数组变量。

说明

Erase 根据是固定大小（常规的）数组还是动态数组，来采取完全不同的行为。Erase 无需为固定大小的数组恢复内存。Erase 按下表来设置固定数组的元素

数组类型	Erase 对固定数组元素的影响
固定数值数组	将每个元素设为 0
固定字符串数组（长度可变）	将每个元素设为零长度字符串("")
固定字符串数组（长度固定）	将每个元素设为 0
固定 Variant 数组	将每个元素设为 Empty
用户定义类型的数组	将每个元素作为单独的变量来设置
对象数组	将每个元素设为特定值 Nothing

Erase 释放动态数组所使用的内存。在下次引用该动态数组之前，程序必须使用 ReDim 语句来重新定义该数组变量的维数。

请参阅

Array 函数，Dim 语句，Prirate 语句，Public 语句，ReDim 语句，Static 语句，Nothing

示例

该示例使用 Erase 语句重新初始化固定大小的数组中的元素，以及释放动态数组存储空间。

```
'声明数组变量。
DimNumArray(10)AsInteger    'Integer 数组。
DimStrVarArray(10)AsString '变长的 String 数组。
DimStrFixArray(10)AsString*10  '定长的 String 数组。
DimVarArray(10)AsVariant    'Variant 数组。
DimDynamicArray()AsInteger '动态数组。
ReDimDynamicArray(10)  '分配存储空间。
EraseNumArray  '将每个元素设为 0。
```

```
EraseStrVarArray    '将每个元素设为零长度字符串("")。
EraseStrFixArray    '将每个元素设为 0。
EraseVarArray   '将每个元素设为 Empty。
EraseDynamicArray   '释放数组所用内存。
```

Err 对象

含有关于运行时错误的信息。

属性

Deseription 属性，HelpContext 属性，HelpFile 属性，LastDLLError 属性，Number 属性，Source 属性

方法

Clear 方法，Raise 方法

说明

Err 对象的属性由错误的生成者来设置，这个生成者或者是 VisualBasic，或者是对象，或者是程序设计员。

Err 对象的缺省属性是 Number。因为该缺省属性可以用对象名称 Err 表示，所以不必修改以前用 Err 函数或 Err 语句书写的代码。

当运行时错误发生时，Err 对象的属性被填入明确识别错误的信息以及处理这个错误所使用的信息。为了在代码中生成运行时错误，请用 Raise 方法。

在任意形式的 Resume 或 OnError 语句之后以及在错误处理子程序内的 ExitSub、ExitFunction、ExitProperty 或 ResumeNext 语句之后，

将 Err 对象的属性重新设置为零或长度为零的字符串("")。可使用 Clear 方法重新明确设置 Err。

为了对系统错误和类模块生成运行时错误，要使用 Raise 方法而不使用 Error 语句。在其它代码中是否使用 Raise 方法，这要看想要返回的信息量有多大。

Err 对象是具有全局范围的固有对象。在代码中没有必要建立这些对象的实例。

请参阅

Error 函数,Error 语句,OnError 语句,Resume 语句,Dictionary 对象，FileSystemObjec 对象

示例

本示例使用 Err 对象的各个属性，来建立一个显示错误信息的对话框。请注意，如过使用 Raise 方法产生 VisualBasic 的错误时，首先使用 Clear 方法，则 VisualBasic 的缺省值会变成 Err 对象的属性值。

```
DimMsg
'如果有错误发生，就构成一个错误信息
OnErrorResumeNext  '改变错误处理的方式。
Err.Clear
Err.Raise6          '生成一个溢出（Overflow）的错误。
'检查错误代号，显示相关错误信息。
IfErr.Number<>0Then
```

```
    Msg="Error#"&Str(Err.Number)&"wasgeneratedby"_
            &Err.Source&Chr(13)&Err.Description
    MsgBoxMsg,,"Error",Err.Helpfile,Err.HelpContext
EndIf
```

Error 事件（数据报表设计器）

当错误中止一个操作时发生。

应用于

DataReport 对象

语法

PrivateSub*object*_**Error**(*JobType***AsAsyncTypeConstants**,Cookie**AsLong** *ErrObj***AsRptError**,*ShowError***AsBoolean**)

Error **事件的语法包含如下部分：**

部分	**描述**
object	**必需的。一个对象表达式，其值为“应用于”列表中的一个对象**
jobType	**返回操作类型，如在设置值中所示**
cookie	**返回操作的 ID。该 ID 在一个异步方法（如** ExportReport **或** PrintReport**）被调用时设置**
ErrObj	**返回错误号**
ShowError	**设定一个值，指定是否显示错误对话框**

设置值

对 JobType 的设置如下：

常数	值	描述
rptAsyncPreview	0	报表正在处理一个预览操作
rptAsyncPrint	1	报表正在处理一个打印操作
rptAsyncReport	2	报表正在处理一个导出报表操作

请参阅

ProcessingTimeout 事件

Error 函数

返回对应于已知错误号的错误信息。

语法

Error[(*errornumber*)]

这个可选的 *errornumber* 参数可以为任何有效的错误号。如果 *errornumber* 是有效的错误号，但尚未被定义，则 **Error** 将返回字符串“应用程序定义的错误或对象定义的错误”。如果 *errornumber* 不是有效的错误号，则会导致错误发生。如果省略了 *errornumber*，就会返回与最近一次运行时错误对应的消息。如果没有发生运行时错误，或者 errornumber 是 0，则 **Error** 返回一个长度为零的字符串("")。

说明

请检查 Err 对象的属性设置，以便认定最近一次运行时错误。Error 函数的返回值对应于 Err 对象的 Description 属性。

请参阅：

Err 对象

示例

本示例使用 Error 函数来显示指定的错误代号所代表之错误信息。

```
DimErrorNumber
ForErrorNumber=61To64          '从错误代号 61 循环到 64。
   Debug.PrintError(ErrorNumber)     '将错误信息在立即窗口中显示。
NextErrorNumber
```

Error 对象（数据报表设计器）

错误对象包含有关运行时错误的详细内容。

语法

RptError

属性

ErrorNumber 属性，Source 属性，Descroption 属性，HelpFile 属性 (App,CommonDialog,MenuLine)HelpContextID 属性

请参阅

DataReport 对象

Error 属性（WebClass 对象）

返回对活动服务器页面的 Error 对象的引用。

应用于

WebClass 对象

语法

object.Error

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

当正常执行时期间，Error 属性值是空，但当 FatalError 事件执行时，却是有效的。

请参阅

FatalErrorRespouse 事件；WebClassError 对象；《MicrosoftVisualBasic6.0 部件工具参考手册》第 3 章“IIS 应用程序调试”，第五部分“Internet 应用程序创建”的“IIS 应用程序开发”。

Error 语句

模拟错误的发生。

语法

Error*errornumber*

必需的 *errornumber* 可以是任何有效的错误号。

说明

Error 语句获得的支持是向后兼容的。在新的代码中，特别是在建立对象时，要使用 Err 对象的 Raise 方法产生运行时错误。

如果已经定义 errornumber，那么，在 Err 对象的属性被赋予下列值之后，Error 语句会调用错误处理程序：

属性	值
Number	作为参数指定给 Error 语句的值。可以是任何有效的错误号
Source	当前 VisualBasic 工程的名称
description	字符串表达式，如果这个字符串存在，则表达式将与 Error 函数的返回值一致，该返回值是 Error 函数对指定的 Number 的返回值。如果这个字符串不存在，则 Description 包含一个长度为零的字符串("")
helpFile	VisualBasic 帮助文件的完整限定的驱动器、路径和文件名
helpContext	对于与 Number 属性一致的错误所指定的 VisualBasic 帮助文件上下文 ID
lastDLLError	零

如果不存在错误处理程序，或未做任何启动动作，那么就会由 Err 对象属性发布一个错误信息并将错误信息显示出来。

注意并非所有 VisualBasic 主应用程序都可以建立对象。请参考

主机应用程序的文档来判断它能否建立类和对象。

请参阅

Err 函数，OnError 语句，Resume 语句，Clear 方法，Raise 方法，Err 对象

示例

本示例使用 Error 语句来模拟发生错误代号 11 的状况。

```
OnErrorResumeNext  '错误处理。
Error11                    '模拟“除以零”的错误。
```

ErrorNumber 属性

返回指定一个错误的数值。

应用于

Error 对象（数据报表设计器）

语法

object.**ErrorNumber**

object 所在处是一个对象表达式，其值为“应用于”列表中的一个对象。

返回类型

Long

Event 语句

定义用户自定义的事件。

语法

[**Public**]**Event***procedurename*[(*arglist*)]

Event 语句包含下面部分：

部分	描述
public	**可选的。指定该** Event **在整个工程中都是可见的。缺省情况下** Events **类型是** Public**。应注意，事件只能在所声明的模块中产生**
procedurename	**必需的。事件的名称；遵循标准的变量命名约定**

arglist **参数的语法及语法的各个部分如下：**

[**ByVal**|**ByRef**]*varname*[**()**][**As***type*]

部分	描述
byVal	可选的。表示该参数是按值传递的
byRef	可选的。表示该参数是按地址传递的。ByRef 是 VisualBasic 的缺省设置
varname	必需的。代表要传递给过程的参数变量的名称；遵循标准的变量命名约定
type	可选的。指传递给过程的参数的数据类型；可以是 Byte、布尔、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（只支持变长）、Object、Variant、用户定义类型或对象类型

说明

一旦事件被声明之后，就可以使用 RaiseEvent 语句来激活该事件。如果在标准模块中出现 Event 声明，就会产生语法错误。不能声明带返回值的事件。在下面的代码段中，给出了声明事件和产生事件的典型事件：

```
'在类模块的模块级中声明一个事件

EventLogonCompleted(UserNameasString)

Sub
   RaiseEventLogonCompleted("AntoineJan")
```

```
EndSub
```

注意可以象声明过程的参数一样来声明事件的参数，但有以下不同：事件不能有带命名参数、Optional 参数、或者 ParamArray 参数。事件没有返回值。

请参阅

《MicrosoftVisualBasic6.0 组件工具指南》的第 6 章“为类添加事件”，第二部分“ActiveX 组件创建”的“组件设计的一般规则”；第 9 章“从控件激发事件”，第二部分“ActiveX 组件创建”的“ActiveX 控件创建”；《MicrosoftVisualBasic6.0 程序员指南》第 9 章“为类添加事件”；RaiseEvent 语句。

示例

下面的示例是用事件来计算百米赛跑中最快时间。代码说明了所有与事件相关的方法、属性和语句，包括 Event 语句。

产生事件的类是事件源，实现该事件的类则是事件接收器。一个事件源可以有多个针对其所产生的事件的接收器。事件可以被每个选定出为对象的实例接收器事件的类所触发。

该示例使用了一个窗体(Form1)，该窗体有一个按钮(Command1)，一个标签(Label1)，以及两个文本框(Text1 和 Text2)。单击按钮后，第一个文本框显示"FromNow"，第二个文本框中时钟开始计时。当经过 9.84 秒（百米记录）之后，第一个文本框显示"UntilNow"，第二个则显示"9.84"。

Form1 代码用于指定该窗体的初始状态和终止状态，也包含了事件产生后

所要执行的代码。

```
OptionExplicit

PrivateWithEventsmTextAsTimerState

PrivateSubCommand1_Click()
Text1.Text="FromNow"
Text1.Refresh
Text2.Text="0"
Text2.Refresh
CallmText.TimerTask(9.84)
EndSub

PrivateSubForm_Load()
Command1.Caption="ClicktoStartTimer"
Text1.Text=""
Text2.Text=""
Label1.Caption="Thefastest100meterruntookthislong:"
SetmText=NewTimerState
EndSub

PrivateSubmText_ChangeText()
```

```
Text1.Text="UntilNow"
Text2.Text="9.84"
EndSub

PrivateSubmText_UpdateTime(ByValdblJumpAsDouble)
Text2.Text=Str(Format(dblJump,"0"))
DoEvents
EndSub
```

下面是 TimerState 类模块的代码。事件产生后 Event 语句声明被初始化的过程。

```
OptionExplicit
PublicEventUpdateTime(ByValdblJumpAsDouble)
PublicEventChangeText()

PublicSubTimerTask(ByValDurationAsDouble)
DimdblStartAsDouble
DimdblSecondAsDouble
DimdblSoFarAsDouble
dblStart=Timer
dblSoFar=dblStart

DoWhileTimer<dblStart+Duration
```

```
IfTimer-dblSoFar>=1Then
dblSoFar=dblSoFar+1
RaiseEventUpdateTime(Timer-dblStart)
EndIf
Loop

RaiseEventChangeText

EndSub
```

EventInfo 对象

表示由一个指定给 VBControlExtender 对象变量的控件引起的事件信息。

语法

EventInfo

说明

EventInfo 对象在 ObjectEvent 事件中是可用的，它是 VBControlExtender 对象的一个事件。在通常情况下，一个使用 Add 方法动态添加到 Controls 集合的控件将被指定给一个 VBControlExtender 类型的对象变量。ObjectEvent 事件就可被用来捕获该控件引起的所有事件，并且 EventInfo 对象特别表示由引起事件传递的任何参数。

EventInfo 对象有两个属性：Name 属性返回所引起的事件的名称：EventParameters 属性返回一个对 EventParameters 集合的引用，允许返回所有事件参数的值。

EventParameter 对象

表示一个控件事件参数。

语法

EventParameter

说明

EventParameter 对象是用来计算和返回一个已动态地添加到 Controls 集合的控件的事件所包含的值。

EventParameters 集合是 EventInfo 对象的一部分。

EventParameter 对象有两个属性：Name 和 Value。Name 属性返回一个参数的名称。然后可以正确地计算和设置 Value 属性。

EventParameters 集合

一个 EventParameter 对象的集合。

语法

EventParameters

说明

使用 EventParameters 集合决定指定给一个控件所引起事件的任何

参数的值。

EventParameters 属性（EventInfo 对象）

返回一个对 Parameters 集合的引用。

语法

object.**Parameters**

object 所在处是一个对象表达式，其值是“应用于”列表中的一个对象。

Events 对象

提供属性，使外接程序连接到 VisualBasicforApplications 中的所有事件。

说明

Events 对象提供了返回事件源对象的属性。可用此属性返回事件源对象，就 VisualBasicforApplications 环境中发生的变化发出通知。

Events 对象的属性可返回与属性名相同的对象。例如，CommandBarEvents 属性返回 CommandBarEvents 对象。

属性

CommandBarEvents 属性，ReferencesEvents 属性，FileControlEvents 属性，SelectedVBControlEvents 属性，VBCompon

entsEvents 属性，BControlsEvents 属性，VBProjectsEvents 属性

事件

ItemAdded 事件,ItemRemoved 事件

请参阅

Click 事件,CommandBarEvents 对象，ReferencesEvents 对象

Events 属性

提供允许外接程序连接到所有 VisualBasicforApplications 事件的属性。

应用于

VBE 对象

语法

object.Events

object 所在处是一个对象表达式，其值是“应用于”列表中的一个对象。

请参阅

EventsFrozen 属性，Cancel 属性

EventsFrozen 属性

返回一个数值，该数值指示现在容器是否正忽略由控件引发的事

件。在创建控件时，EventsFrozen 属性不可用，在控件运行时，该属性是只读的。

应用于

UserControl 对象

语法

object.EventsFrozen

EventsFrozen 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

EventsFrozen 属性设置为 True 后，容器将忽略由控件引发的任何事件。如果控件需要产生不能丢失的事件，则在 EventsFrozen 为 False 之前，这些事件必须排入队列中

EXEName 属性

返回当前正运行的可执行文件的根名（不带扩展名）。如果是在开发环境下运行，则返回该工程名。

语法

object.EXEName

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

Exists 方法

如果在 Dictionary 对象中指定的关键字存在，返回 True，若不存在，返回 False。

应用于

Dictionary 对象

语法

object.**Exists**(*key*)

Exists 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 Dictionary 对象的名字
key	必需的。在 Dictionary 对象中搜索的 *Key* 值

请参阅

Add 方法，Items 方法，Keys 方法，Remove 方法，RemoveAll 方法

Exit 语句

退出 Do...Loop、For...Next、Function、Sub 或 Property 代码块。

语法

ExitDo

ExitFor

ExitFunction

ExitProperty

ExitSub

Exit 语句的语法有以下几种形式：

语句	描述
ExitDo	提供一种退出 Do...Loop 循环的方法，并且只能在 Do...Loop 循环中使用。ExitDo 会将控制权转移到 Loop 语句之后的语句。当 ExitDo 用在嵌套的 Do...Loop 循环中时，ExitDo 会将控制权转移到 ExitDo 所在位置的外层循环
ExitFor	提供一种退出 For 循环的方法，并且只能在 For...Next 或 ForEach...Next 循环中使用。ExitFor 会将控制权转移到 Next 之后的语句。当 ExitFor 用在嵌套的 For 循环中时，ExitFor 将控制权转移到 ExitFor 所在位置的外层循环
ExitFunction	立即从包含该语句的 Function 过程中退出。程序会从调用 Function 的语句之后的语句继续执行
ExitProperty	立即从包含该语句的 Property 过程中退出。程序会从调用 Property 过程的语句之后的语句继续执行
ExitSub	立即从包含该语句的 Sub 过程中退出。程序会从调用 Sub 过程的语句之后的语句继续执行

说明

不要将 Exit 语句与 End 语句搞混了。Exit 并不标志一个结构的结

束。

示例

本示例使用 Exit 语句退出 For...Next 循环、Do...Loop 循环及子过程。

```
SubExitStatementDemo()
DimI,MyNum
   Do   '建立无穷循环。
      ForI=1To1000      '循环 1000 次。
         MyNum=Int(Rnd*1000)  '生成一随机数码。
         SelectCaseMyNum      '检查随机数码。
            Case7:ExitFor    '如果是 7，退出 For...Next 循环。
            Case29:ExitDo    '如果是 29，退出 Do...Loop 循环。
            Case54:ExitSub   '如果是 54，退出子过程。
         EndSelect
      NextI
   Loop
EndSub
```

ExitFocus 事件

当焦点离开对象时，发生该事件。可能是对象本身失去焦点，也可能是子控件失去焦点。

应用于

UserControl 对象，UserDocument 对象

语法

Sub*object*_ExitFocus()

ExitFocus 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

通过此事件，object 可以获知焦点是否正在离开。

ExitFocus 事件产生在任何 LostFocus 事件之后；LostFocus 事件仅产生在确实丢失焦点的 object 或 object 的子控件中。

请参阅

EnterFocus 事件,LostFocus 事件

Exp 函数

返回 Double，指定 e（自然对数的底）的某次方。

语法

Exp(*number*)

必需的 number 参数是 Double 或任何有效的数值表达式。

说明

如果 number 的值超过 709.782712893，则会导致错误发生。常数 e 的值大约是 2.718282。

注意 Exp 函数的作用和 Log 的作用互补，所以有时也称做反对数。

请参阅

Log 函数，DerivedMath 函数

示例

本示例使用 Exp 函数计算 e（e~2.71828）的某次方。

```
DimMyAngle,MyHSin
'定义角度（以“弧度”为单位）。
MyAngle=1.3
'计算双曲正弦函数值（sin()）。
MyHSin=(Exp(MyAngle)-Exp(-1*MyAngle))/2
```

Export 方法（VBA 外接程序对象模型）

将部件按单独文件或一些文件进行保存。

应用于

VBComponent 对象

语法

object.**Export**(*filename*)

Export 方法语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
filename	必需的。一个 String 型数，用来指定部件输出为文件的文件名

说明

当使用 Export 方法将部件存为文件时，使用的文件名不能重名；否则，将可产生错误。

请参阅

Import 方法（VBA 外接程序对象模型）

示例

下面示例生成一个名为 test.bas 的文件，并使用 Export 方法把 VBComponents(1)代码模块中的内容拷贝到 test.bas 文件。

```
Application.VBE.ActiveVBProject.VBComponents(1).ExPort("test.bas")
```

ExportFormat 对象

使用 ExportFormat 对象，可以通过编程决定从数据报表中导出的文本的各种属性。

语法

ExportFormat

说明

使用 ExportReport 方法导出一个报表时，必须指定一个 ExportFormat 对象。

属性

FormatType 属性,FileFormatString 属性,FileFilter 属性,Template 属性,Key 属性

请参阅

DataReport 对象,ExportFormats 集合,ExportFormats 属性,ExportReport 方法,Title 属性,Add 方法(ExportFormats 集合)

示例

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
    DataReport1.Title="DailyReport"'本标题出现在报表中。
    DimstrTemplateAsString
    '创建模板
    strTemplate=_
    "<HTML>"&vbCrLf&_
    "<HEAD>"&vbCrLf&_
    "<TITLE>"&"MyCompany:"&rptTagTitle&_
    "</TITLE>"&vbCrLf&_
    "<BODY>"&vbCrLf&_
    rptTagBody&vbCrLf&_
    "<BODY>"&vbCrLf&_
    "</HTML>"

    '使用该模板添加一个新 ExportFormat 对象。
    DataReport1.ExportFormats.Add_
    Key:="DailyReport",_
    FormatType:=rptFmtHTML,_
```

```
    FileFormatString:="DailyReport(*.htm)",_
    FileFilter:="*.HTM",_
    Template:=strTemplate

    '使用新的 ExportFormat 对象导出报表。
    DataReport1.ExportReport_
    FormatIndexOrKey:="DailyReport",_
    FileName:="C:\Temp\DailyRpt",_
    Overwrite:=True,_
    ShowDialog:=False,_
    Range:=rptRangeFromTo,_
    Pagefrom:=1,_
    Pageto:=10
EndSub
```

ExportFormats 集合

一个 ExportFormat 对象的集合。

语法

ExportFormats

说明

使用 ExportFormat 对象连同 ExportReport 方法一起来创建自定义导出的报表。当调用 ExportReport 方法时，要指定使用哪一个

ExportFormat 对象，如下所示：

```
'使用名为 MyReport 的 ExportFormat 对象。
DataReport1.ExportReport"MyReport"
```

集合中的每一对象可以表示一个不同的模板。要创建一个模板，使用 Template 属性。

当最终用户调用 ExportReport 方法时，FileFormatString 属性将决定在“另存类型”框中显示什么文本。

属性

Count 属性，(VB 集合),Item 属性

方法

Insert 方法,Add 方法(ExportFormats 集合),Clear 方法,Remove 方法

请参阅

ExprotFormat 对象，ExprotReport 方法。

第一个示例使用 ExportReport 方法显示“导出”对话框。第二个示例导出文件，而不显示“导出”对话框。第三个示例指定导出报表时要使用的一个 ExportFormat 对象。

```
PrivateSubExportTheReport()
   DataReport1.ExportReport,,True,True
EndSub

PrivateSubExportWithoutDialog()
   '导出到一个名为 Output.htm 的文件，如果有必要就覆盖。
```

```
    DataReport1.ExportReportrptKeyHTML,"C:\Temp\Output",True,Fa
lse
EndSub

PrivateSubExportMyReport()
    '使用 MyReportExportFormat 导出到一个名为 Daily.htm 的文件。
    DataReport1.ExportReport"MyReport","C:\Temp\Daily",True,Fal
se
EndSub
```

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
    DataReport1.Title="DailyReport"'本标题出现在报表中。
    DimstrTemplateAsString
    '创建模板
    strTemplate=_
    "<HTML>"&vbCrLf&_
    "<HEAD>"&vbCrLf&_
    "<TITLE>"&"MyCompany:"&rptTagTitle&_
    "</TITLE>"&vbCrLf&_
    "<BODY>"&vbCrLf&_
    rptTagBody&vbCrLf&_
```

```
    "<BODY>"&vbCrLf&_
    "</HTML>"

    '使用该模板添加一个新 ExportFormat 对象。
    DataReport1.ExportFormats.Add_
    Key:="DailyReport",_
    FormatType:=rptFmtHTML,_
    FileFormatString:="DailyReport(*.htm)",_
    FileFilter:="*.HTM",_
    Template:=strTemplate

    '使用新的 ExportFormat 对象导出报表。
    DataReport1.ExportReport_
    FormatIndexOrKey:="DailyReport",_
    FileName:="C:\Temp\DailyRpt",_
    Overwrite:=True,_
    ShowDialog:=False,_
    Range:=rptRangeFromTo,_
    Pagefrom:=1,_
    Pageto:=10
EndSub
```

ExportFormats 属性

返回一个对 ExportFormats 集合的引用。

应用于

DataReport 对象。

语法

object.**ExportFormats**

object.**ExportFormats**(*index*)

ExportFormats 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的对象
index	可选的。集合中一个成员的索引或关键字

说明

该属性使用集合语法指定要返回的是集合，还是集合的一个成员。

ExportFormat 对象的 FileFormatString 属性决定在“导出”对话框的“另存类型”框中显示的文本。

示例

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
```

```
DataReport1.Title="DailyReport"'本标题出现在报表中。
DimstrTemplateAsString
'创建模板
strTemplate=_
"<HTML>"&vbCrLf&_
"<HEAD>"&vbCrLf&_
"<TITLE>"&"MyCompany:"&rptTagTitle&_
"</TITLE>"&vbCrLf&_
"<BODY>"&vbCrLf&_
rptTagBody&vbCrLf&_
"<BODY>"&vbCrLf&_
"</HTML>"

'使用该模板添加一个新 ExportFormat 对象。
DataReport1.ExportFormats.Add_
Key:="DailyReport",_
FormatType:=rptFmtHTML,_
FileFormatString:="DailyReport(*.htm)",_
FileFilter:="*.HTM",_
Template:=strTemplate

'使用新的 ExportFormat 对象导出报表。
DataReport1.ExportReport_
```

```
    FormatIndexOrKey:="DailyReport",_
    FileName:="C:\Temp\DailyRpt",_
    Overwrite:=True,_
    ShowDialog:=False,_
    Range:=rptRangeFromTo,_
    Pagefrom:=1,_
    Pageto:=10
EndSub
```

ExportReport 方法

使用一个指定的 ExportFormat 对象导出报表的文本到一个文件。图像和图形不能被导出。

应用于

DataReport 对象

语法

object.**ExportReport**(*index*,*filename*,*Overwrite*,*ShowDialog*,*Range*,*PageFrom*,*PageTo*)

ExportReport 方法的语法包含如下部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
index	可选的。一个索引、关键字或指定被使用的 ExportFormat 对象的 ExportFormat 对象引用
filename	可选的。一个字符串表达式，其值为文件名。如果未被指定，会显示“导出”对话框
overwrite	可选的。一个布尔表达式，决定文件是否被覆盖
showDialog	可选的。一个布尔表达式，决定是否显示“另存为...”对话框。如果没有指定 ExportFormat 对象或 *filename*，即使这一参数被设定为 False，也将显示“导出”对话框
range	可选的。设置一个整数，决定是否包含报表中的所有页面，或者是仅包含一定范围的页面。如在设置值中所示
pageFrom	可选的。一个数值表达式，指定了导出开始的页面
pageTo	可选的。一个数值表达式，指定了导出终止的页面

对 Range 的设置如下：

常数	值	描述
rptRangeAllPages	0	（缺省的）所有页面都将打印
rptRangeFromTo	1	只有指定范围的页面将被导出

返回值

Long

说明

如果未给该方法提供参数，将显示一个对话框，提示用户提供相应的信息（如文件名）。

ExportReport 方法执行一个异步操作。该方法返回“cookie”的标识符，来标识异步的操作。

重点指定范围的页面与在“打印预览”方式中看到的页面是不匹配的。因为导出页面数目是基于 ExportFormat 对象 ExportType 属性的字体属性，而打印和预览页面是基于计算机使用的当前打印机对象。

请参阅

ExportFormat 对象，ExportFormat 集，ExportFormat 属性，ExportType 属性，FileFilter 属性，Template 属性，ProcessingTimeout 事件

示例

第一个示例使用 ExportReport 方法显示“导出”对话框。第二个示例导出文件，而不显示“导出”对话框。第三个示例指定导出报表时要使用的一个 ExportFormat 对象。

```
PrivateSubExportTheReport()
   DataReport1.ExportReport,,True,True
EndSub
```

```
PrivateSubExportWithoutDialog()
   '导出到一个名为 Output.htm 的文件，如果有必要就覆盖。
   DataReport1.ExportReportrptKeyHTML,"C:\Temp\Output",True,Fals
e
EndSub

PrivateSubExportMyReport()
   '使用 MyReportExportFormat 导出到一个名为 Daily.htm 的文件。
   DataReport1.ExportReport"MyReport","C:\Temp\Daily",True,False
EndSub
```

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
   DataReport1.Title="DailyReport"'本标题出现在报表中。
   DimstrTemplateAsString
   '创建模板
   strTemplate=_
   "<HTML>"&vbCrLf&_
   "<HEAD>"&vbCrLf&_
   "<TITLE>"&"MyCompany:"&rptTagTitle&_
   "</TITLE>"&vbCrLf&_
   "<BODY>"&vbCrLf&_
```

```
    rptTagBody&vbCrLf&_
    "<BODY>"&vbCrLf&_
    "</HTML>"

    '使用该模板添加一个新 ExportFormat 对象。
    DataReport1.ExportFormats.Add_
    Key:="DailyReport",_
    FormatType:=rptFmtHTML,_
    FileFormatString:="DailyReport(*.htm)",_
    FileFilter:="*.HTM",_
    Template:=strTemplate

    '使用新的 ExportFormat 对象导出报表。
    DataReport1.ExportReport_
    FormatIndexOrKey:="DailyReport",_
    FileName:="C:\Temp\DailyRpt",_
    Overwrite:=True,_
    ShowDialog:=False,_
    Range:=rptRangeFromTo,_
    Pagefrom:=1,_
    Pageto:=10
EndSub
```

Extender 对象

Extender 对象中包括那些实际由控件的容器控制的属性，而不是由控件本身控制的控件属性。

语法

Extender

说明

某些控件的属性由容器而不是由控件提供的；这些就是扩展属性。扩展属性的例子有：Name、Tag 和 Left。控件仍需知道这些扩展属性的属性值，有时还需要能够更改某个扩展属性；Extender 对象使控件能够访问这些属性。

某些扩展属性是标准的，另外一些属性是某种容器特定的。控件可以访问非标准扩展属性，但这样做将使控件只适用于特定的容器。如果控件使用某个扩展属性，该控件应处理当前容器不支持该扩展属性的情况。

编译控件时，VisualBasic 无法获知控件运行时可以使用何种扩展属性；因此对扩展属性的引用总是后期绑定的。

执行初始化事件时，Extender 对象不可用；但是执行 InitProperties 事件或 ReadProperties 事件时，该对象可用。

Extender 对象有下列标准属性：

Name 属性，只读字符串，其中包含用户定义的控件名。

Visible 属性，可读可写的布尔值，它指定控件是否可见。

Parent 属性，代表控件容器的只读对象，例如 VisualBasic 中的

窗体。

Cancel 属性，只读的布尔值，它指示控件是否为容器的缺省“取消”按钮。

Default 属性，只读的布尔值，它指示控件是否为容器的缺省按钮。

VisualBasic 提供一些其它扩展的方法、属性和事件；其它的容器不保证提供这些扩展的方法、属性和事件。这些 VisualBasic 特定的扩展方法、属性和事件是：

Container 属性，代表控件可视的控件容器的只读对象。

DragIcon 属性，可读可写的图片，它指定拖动控件时使用的图标。

DragMode 属性，可读可写的整型值，它指定控件是否可以自动拖动，还是必须调用 Drag 方法。

Enabled 属性，只读的布尔值，它指定是否启用控件。除非控件也有带正确过程 ID 的 Enabled 属性，否则不提供该扩展属性。详细信息，请参阅《MicrosoftVisualBasic6.0 部件工具指南》的第九章“连编 ActiveX 控件”中的“允许启用和禁止控件”主题。

Height 属性，可读可写的整型值，它使用容器的刻度单位指定控件的高度。

HelpContextID 属性，可读可写的整型值，它指定在控件获得焦点时按下 F1 键使用的上下文 ID。

Index 属性，只读的整型值，它指定这个控件的实例在控件数组中占据的位置。

Left 属性，可读可写的整型值，它使用容器的刻度单位指定控件

左边缘相对于容器左边缘的位置。

TabIndex 属性，可读可写的整型值，它指定控件在容器控件的 Tab 键次序中的位置。

TabStop 属性，可读可写的布尔值，它指定 Tab 是否停在该控件上。

Tag 属性，可读可写的字符串型，其中包含用户定义值。

ToolTipText 属性，可读可写的字符串，其中包括游标停留在控件上方一秒钟以上时显示的文本。

Top 属性，可读可写的整型值，它使用容器的刻度单位指定控件的上边缘相对于容器上边缘的位置。

WhatThisHelpID 属性，可读可写的整型值，它指定在控件上使用“这是什么”弹出文本时使用的上下文 ID。

Width 属性，可读可写的整型值，它使用容器的刻度单位指定控件的宽度。

Drag 方法，启动、终止或取消控件拖动操作的方法。

Move 方法，移动控件位置的方法。

SetFocus 方法，将焦点设置到控件上的方法。

ShowWhatsThis 方法，使用由“帮助”提供的弹出文本“这是什么’在帮助文件中显示选定主题的方法。

ZOrder 方法，将控件放置在其图形级 z 顺序的前端或后端的方法。

DragDrop 事件，将窗体上另外一个控件拖放到控件上时产生的事件。

DragOver 事件，拖动窗体上另一个控件经过此控件时产生的事

件。

GotFocus 事件，控件获得焦点时产生的事件。

LostFocus 事件，控件失去焦点时产生的事件。

属性

DataMember 属性,DataFormat 属性,DataBindings 属性,Cancel 属性,Height,Width 属性,Left,Top 属性,TakIndex 属性,Tag 属性,Visible 属性,Default 属性,DragIcon 属性,TabStop 属性,Enabled 属性,HelpContextID 属性,Name 属性,Parent 属性,Container 属性,TooltipText 属性,WhatsThisHelpID 属性,Align,CausesValidation 属性,DataChanged 属性,DataField 属性,DataSource 属性

方法

SetFocus方法,Drag 方法,Move 方法,Zorder 方法,ShowWhatsThis 方法,LinkExecute 方法,LinkPoke 方法,LinkRequest 方法,LinkSend 方法

事件

ObjectEvent 事件,DragDrop 事件,DragOver 事件,GoFocus 事件,LinkClose 事件,LinkError 事件,LinkNotify 事件,LinkOpen 事件,LostFocus 事件,Validate 事件

请参阅

InitProperties 事件,ReadProperties 事件，Initialize 事件,Parameter 对象(VisualBasic)

Extender 属性

返回该控件的 Extender 对象，它包含了由容器监视的控件属性。在控件创建时，Extender 属性不可用，在控件运行时，该属性是只读的。

应用于

UserControl 对象

语法

object.**Extender**

Extender 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

请参阅

Extender 对象

False

False 关键字的值等于 0

请参阅

True,Boolean 数据类型

FalseValue 属性

设置或返回一个值，用于格式化和去除格式化一个布尔 False 值。在运行时或设计时都可读写。

应用于

StdDataFormat 对象

语法

object.**FalseValue**[=*value*]

FalseValue 属性语法有以下的部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
value	可选的变体型。在 Format 事件中，如果数据是布尔 False，那么将返回 value。如果 value 为空，则返回整型数 1。在 Unformat 事件中，如果数据与 value 相匹配,则一个布尔 False 将被写到数据库中。缺省值为 False

说明

除非 Type 属性设置为 fmtBoolean，否则它将被忽略。每次取数据时都会读 FalseValue 属性。

请参阅

NullValue 属性，TrueValue 属性，Type 属性

FatalErrorResponse 事件

在 WebClass 对象的处理因为错误而终止时发生。

应用于

WebClass 对象

语法

PrivateSub*object*_FatalErrorResponse(*senddefault*AsBoolean)

FatalErrorResponse 事件的语法有这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
senddefault	当设置成 True 时，senddefault 意味着应将缺省活动服务器页面错误消息返回给浏览器。如果 WebClass 对象写入它自己的消息，应将 *senddefault* 设置成 False 以避免缺省处理

说明

如果 WebClass 被终止，而且您选择了写入自己的错误信息，例如 senddefault=false

那么必须将写入信息的代码放置在该事件内部。任何 Visual Basic 错误或终端错误都将导致 FatalErrorResponse 事件。

FetchVerbs 方法

更新对象支持的谓词列表。

应用于

OLE 包容器控件，OLEObject 对象

语法

object.**FetchVerbs**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

可用 ObjectVerbs 属性读取更新的谓词列表。

请参阅

ObjectVerbs 属性，ObjectVerbsCount 属性，AutoVerbMenu 属性，DoVerb 方法

File 对象

提供对文件所有属性的访问。

说明

下面的代码举例说明了如何获得一个 File 对象，以及如何查看它的一个属性。

```
SubShowFileInfo(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
```

```
Setf=fs.GetFile(filespec)
s=f.DateCreated
MsgBoxs
EndSub
```

属性

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Name 属性（FileSyStem Object 对象），ParentFolder 属性，Path 属性（FileSystemObject 对象），ShortName 属性，ShortPath 属性，Size 属性（FileSystemObject 对象），Type 属性，

方法

Copy 方法，Delete 方法，Move 方法，OpenAsTextStream 方法

请参阅

Drive 对象，Drives 集合，Files 集合，FileSystemObject 对象，Folder 对象，Folders 集合

FileAttr 函数

返回一个 Long，表示使用 Open 语句所打开文件的文件方式。

语法

FileAttr(*filenumber,returntype*)

FileAttr 函数的语法具有以下几个命名参数：

部分	描述
filenumber	必需的。Integer 类型，任何有效的文件号
returntype	必需的。Integer 类型。它是数字，指出返回信息的类型。指定 1 则可返回一个代表文件方式的数值。而仅仅在 16 位系统中，指定 2 才可以恢复操作系统的文件句柄。在 32 位系统中不支持 Returntype2，它会导致错误发生

返回值

当 returntype 参数值为 1 时，下列返回值指出文件访问方式：

方式	值
Input	1
Output	2
Random	4
Append	8
Binary	32

请参阅

GetAttr 函数，AetAttr 语句，Open 语句

示例

本示例使用 FileAttr 函数来返回一个已打开文件的文件模式以及文件句柄。返回文件句柄只适用于 16 位系统;在 32 位系统中,把 2 作为第二个参数时将产生错误。

```
DimFileNum,Mode,Handle
FileNum=1 '指定文件号。
Open"TESTFILE"ForAppendAsFileNum  '打开文件。
Mode=FileAttr(FileNum,1)  '返回8（增加文件方式）。
Handle=FileAttr(FileNum,2) '返回文件句柄。
CloseFileNum'关闭文件。
```

FileControlEvents 对象

代表所有由 VisualBasic 提供的、支持文件控制的事件。

语法

FileControlEvents

说明

FileControlEvents 对象取代了 VisualBasic4.0 版中的 FileControl 对象。除了对其事件做些改动以允许多个工程支持以外，FileControlEvents 对象一如既往，按相同方式工作。

事件

AfterAddFile 事件,AfterChangeFileName 事件,AfterColseFile 事件，AfterRemoveFile 事件，AfterWriteFile 事件，Before LoadFile 事件,DoGetNewFileName 事件,RequestChangeFileName 事件，RequestWriteFile 事件

请参阅

FileControlEvents 属性

FileControlEvents 属性

该属性返回 FileControlEvents 类型的事件对象。

应用于

Events 对象

语法

object.**FileControlEvents**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

请参阅

FileControlEvents 对象

FileCopy 语句

复制一个文件。

语法

FileCopy*source*,*destination*

FileCopy 语句的语法含有以下这些命名参数：

部分	描述
source	必需的。字符串表达式，用来表示要被复制的文件名。source 可以包含目录或文件夹、以及驱动器
destination	必需的。字符串表达式，用来指定要复制的目地文件名 *destination* 可以包含目录或文件夹、以及驱动器

说明

如果想要对一个已打开的文件使用 FileCopy 语句，则会产生错误。

请参阅

Kill 语句，Name 语句

示例

本示例使用 FileCopy 语句来复制文件。示例中假设 SRCFILE 为含有数据的文件。

```
DimSourceFile,DestinationFile
SourceFile="SRCFILE"                    '指定源文件名。
DestinationFile="DESTFILE"              '指定目的文件名。
FileCopySourceFile,DestinationFile      '将源文件的内容复制到目的文件中。
```

FileCount 属性

该属性返回与给定部件相关联的文件号。

语法

object.**FileCount**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

FileCount 的主要用途是提醒用户部件是否有关联的.frx 文件。对于大部分部件，该属性设置值是 1，但可以更大。例如，如果.FRX 文件与窗体文件相关联，则 FileCount 属性的设置值是 2。

请参阅

FileNames 属性

FileDateTime 函数

返回一个 Variant(Date)，此为一个文件被创建或最后修改后的日期和时间。

语法

FileDateTime(*pathname*)

必需的 *pathname* 参数是用来指定一个文件名的字符串表达式。*pathname* 可以包含目录或文件夹、以及驱动器。

请参阅

FileLen 函数，GetAttr 函数，VarType 函数

示例

本示例使用 FileDateTime 函数来得知文件创建或最近修改的日期与时间。日期与时间的显示格式依系统的地区设置而定。

```
DimMyStamp
'假设 TESTFILE 上次被修改的时间为 1993 年 2 月 12 日下午 4 时 35 分 47 秒。
'假设 English/U.S.地区设置。
MyStamp=Filedatetime("TESTFILE")    '返回“2/12/934:35:47PM”。
```

FileDescription 属性

返回或设置一个字符串，该字符串包括运行中应用程序的文件说明信息。该属性在运行时是只读的。

应用于

App 对象

语法

object.**FileDescription**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

在设计时，使用位于“工程属性”对话框中的“生成”选项卡上的“类型”框可设置该属性。

FileExists 方法

如果指定的文件存在，返回 True，若不存在，则返回 False。

应用于

FileSystemObject 对象

语法

object.**FileExists**(*filespec*)

FileExists 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
filespec	必需的。要确定是否存在的文件的名字。如果认为文件不在当前文件夹中，必须提供一个完整的路径说明（绝对的或相对的）

请参阅

DriveExists 方法，FolderExists 方法

FileFilter 属性

返回或设置导出报表时使用的文件过滤器（扩展名）。

应用于

ExportFormat 对象

语法

object.**FileFilter**[=*string*]

FileFilter 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
string	可选的。创建文件时所使用的扩展名，格式如“.xxx”

说明

FileFilter 属性决定将在“导出”对话框中显示什么过滤器。该属性可以被指定多个过滤器，过滤器之间用分号隔开（不能有空格），如下所示：

```
DataReport1.ExportFormats(5).FileFilter="*.htm;*.html"
```

然而要注意，FileFilter 应该与分配给 ExportFormat 对象的 FormatType 类型相匹配。例如，设定 FileFilter 为“*.htm”，而设置 FormatType 为 rptFmtText，则这一对设置将无效。还要注意的时，这种情况不会产生错误。

请参阅

ExportReport 方法

示例

本例创建一个新的 ExportFormat 对象，并把它插入到 ExportFormats 集合。

```
DimexfSpecialAsNewExportFormat
WithexfSpecial
    .FileFormatString="SpecialReport(*.txt)"
    .FileFilter="*.txt"
    .FormatType=rptFmtText
    .Key="Special"
    .Template="SpecialReport"&vbCrLf&rptTagBody
EndWith
```

```
WithDataReport1
    .ExportFormats.InsertexfSpecial
    .ExportReport"Special"
EndWith
```

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
    DataReport1.Title="DailyReport"'本标题出现在报表中。
    DimstrTemplateAsString
    '创建模板
    strTemplate=_
    "<HTML>"&vbCrLf&_
    "<HEAD>"&vbCrLf&_
    "<TITLE>"&"MyCompany:"&rptTagTitle&_
    "</TITLE>"&vbCrLf&_
    "<BODY>"&vbCrLf&_
    rptTagBody&vbCrLf&_
    "<BODY>"&vbCrLf&_
    "</HTML>"

    '使用该模板添加一个新 ExportFormat 对象。
    DataReport1.ExportFormats.Add_
```

```
    Key:="DailyReport",_
    FormatType:=rptFmtHTML,_
    FileFormatString:="DailyReport(*.htm)",_
    FileFilter:="*.HTM",_
    Template:=strTemplate

    '使用新的 ExportFormat 对象导出报表。
    DataReport1.ExportReport_
    FormatIndexOrKey:="DailyReport",_
    FileName:="C:\Temp\DailyRpt",_
    Overwrite:=True,_
    ShowDialog:=False,_
    Range:=rptRangeFromTo,_
    Pagefrom:=1,_
    Pageto:=10
End Sub
```

FileFormatString 属性

返回或设置被显示在“导出”对话框的“另存为...”组合框中的字符串。

应用于

ExportFormat 对象

语法

object.**FileFormatString**[=*string*]

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
string	可选的。与 ExportFormat 对象相关联，并且显示在“导出”对话框中

说明

FileFormatString 属性一般包括一个标识 ExportFormat 对象的名称和 FileFilter 属性的值，如下例中所示：

```
DataReport1.ExportFormats(5).FileFormatString="DailyReport(*.HTM)"
```

示例

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
   DataReport1.Title="DailyReport"'本标题出现在报表中。
   DimstrTemplateAsString
   '创建模板
   strTemplate=_
   "<HTML>"&vbCrLf&_
```

```
"<HEAD>"&vbCrLf&_
"<TITLE>"&"MyCompany:"&rptTagTitle&_
"</TITLE>"&vbCrLf&_
"<BODY>"&vbCrLf&_
rptTagBody&vbCrLf&_
"<BODY>"&vbCrLf&_
"</HTML>"

'使用该模板添加一个新 ExportFormat 对象。
DataReport1.ExportFormats.Add_
Key:="DailyReport",_
FormatType:=rptFmtHTML,_
FileFormatString:="DailyReport(*.htm)",_
FileFilter:="*.HTM",_
Template:=strTemplate

'使用新的 ExportFormat 对象导出报表。
DataReport1.ExportReport_
FormatIndexOrKey:="DailyReport",_
FileName:="C:\Temp\DailyRpt",_
Overwrite:=True,_
ShowDialog:=False,_
Range:=rptRangeFromTo,_
```

```
    Pagefrom:=1,_
    Pageto:=10
EndSub
```

FileLen 函数

返回一个 Long，代表一个文件的长度，单位是字节。

语法

FileLen(*pathname*)

必需的 *pathname* 参数是用来指定一个文件名的字符串表达式。*pathname* 可以包含目录或文件夹、以及驱动器。

说明

当调用 FileLen 函数时，如果所指定的文件已经打开，则返回的值是这个文件在打开前的大小。

注意若要取得一个打开文件的长度大小，使用 LOF 函数。

请参阅

FileDateTime 函数，GetAttr 函数，LOD 函数

示例

本示例使用 FileLen 来返回文件的字节长度。示例中假设 TESTFILE 为含有数据的文件。

```
DimMySize
MySize=FileLen("TESTFILE")Filelen("TESTFILE")'返回文件的字节长度。
```

FileListBox 控件

在运行时，在 Path 属性指定的目录中，FileListBox 控件将文件定位并列举出来。该控件用来显示所选择文件类型的文件列表。例如，可以在应用程序中创建对话框，通过它选择一个文件或者一组文件。

语法

FileListBox

说明

设置 List、ListCount 和 ListIndex 属性，可以访问列表中的项目。如果需要显示 DirListBox 和 DriveListBox 控件，那么可以编写代码，使它们与 FileListBox 控件同步，并使它们之间彼此同步。

属性

OLEDragMode 属性，OLEDropMode 属性，BackColor,ForeColor 属性，FontBold,FontItalic,FontStrikethru,FontUnderline 属性，FontName 属性，FontSize 属性，Height,Width 属性，Left,Top 属性,List 属性,ListCount 属性,ListIndex 属性,TabIndex 属性，Tag 属性，Visisble 属性，Archive,Hidden,Normal,System 属性，DragIcon 属性，DragMode 属性，FileName 属性，hWnd 属性,Locked 属性,ReadOnly 属性,MouseIcon 属性,MousePointer 属性，MultiSelect 属性，Pattern 属性，Selected 属性，TabStop 属性，TopIndex 属性，Appearance 属性，Enabled 属性，Help ContextID 属性，Index 属性（ControlArray），Name 属性，Parent

属性，Path 属性，Font 属性，Container 属性，ToolTipText 属性，WhatsThisHelpID 属性

方法

Refresh 方法，SetFocus 方法，Drag 方法，Move 方法，Zorder 方法，OLEDrag 方法，ShowWhatsThis 方法

事件

Click 事件，DragDrop 事件，DragOver 事件，GotFocus 事件,KeyDown,KeyUp 事件，KeyPress 事件，LostFocus 事件，MouseDown,MouseUp 事件，MouseMove 事件，PathChange 事件，PatternChange 事件，Scroll 事件，Validate 事件，DblClick 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件

请参阅

DirListBox 控件，DriveListBox 控件，List 属性，ListCount 属性，ListIndex 属性，Path 属性

FileName 属性

该属性返回组工程文件的完整路径名。

应用于

VBProject 对象

语法

object.**Filename**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

组工程与文件名不同，它们没有名字。

返回的路径名总是以绝对路径的形式提供（例如，"c:\projects\myproject.vbp"），即使它在 VisualBasic 中表示为相对路径（如"..\projects"）。

请参阅

FileName 属性，FileNames 属性

FileName 属性

返回或设置所选文件的路径和文件名。对于 FileListBox 控件该属性在设计时不可用。

应用于

CommonDialog 控件，FileListBox 控件

语法

object.**FileName**[=*pathname*]

FileName 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
pathname	字符串表达式，指定路径和文件名

说明

运行时创建控件时，FileName 属性设置为 0 长度字符串("")，表示当前没有选择文件。

在 CommonDialog 控件里，可以在打开对话框之前设置 FileName 属性以设定初始文件名。

读该属性，返回当前从列表中选择的文件名。路径用 Path 属性单独获取。在功能上，该值与 List(*ListIndex*)等价。如果没有选择文件，FileName 返回 0 长度字符串。

设置这个属性时：

若字符串中包含驱动器、路径或模式，则会相应地改变 Drive、Path 和 Pattern 属性。

若字符串中包含存在的文件名（不包含通配符），则会选择该文件。

改变该属性值可能会产生一个或多个如下事件：PathChange（如果改变路径），PatternChange（如果改变模式），或 DblClick（如果指定存在的文件名）。

该属性值可以是限定的网络路径和文件名，可用下述语法：

\\servername\sharename\pathname

说明

FileNames 属性

请参阅

PathChange 事件，PatternChange 事件，List 属性，ListIndex 属性，Drive 属性，Pattern 属性，Path 属性，RichTextBox 控

件，LoadFile 方法，SaveFile 方法，SupportedRTFCodes(RichTextBox)控件，MultimediaMCI 控件，Command 属性（MultimediaMCI 控件）

FileNames 属性

该属性返回被存储部件的当前路径名。

应用于

VBComponent 对象

语法

object.**FileNames**(*index*)

FileNames 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
index	长整数，指定文件名在返回的索引字符串中的位置

说明

返回的路径名总是以绝对路径的形式提供（例如"c:\projects\myproject.vbp"），即使它在 VisualBasic 中表示为相对路径（如"..\projects"）。

索引字符串的条目数由 FileCount 属性的设置值所决定。类和模块的索引字符串仅包含一个文件名，而对窗体的索引字符串则包含窗体的.Frm 和.Frx 的文件名。当在对象上调用 SaveAs 方法时，这些文件名的值即被更新。

请参阅

FileCount 属性

FileNumber 属性

返回或设置当保存或加载对象时要使用的文件号，或者返回最近使用的文件号。在设计时不可用。

注意包含 FileNumber 属性是为了与早期版本的 Action 属性兼容。要获得目前的该功能，可使用 SaveToFile 和 ReadFromFile 方法。

应用于

OLE 容器控件

语法

object.**FileNumber**[=*number*]

FileNumber 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个指定文件号的数值表达式

说明

文件号必须与一个打开的二进制文件对应。

可以使用这个属性指定要打开的、具有 ReadFromFile 方法的、或是要保存的、具有 SaveToFile 或 SaveToOle1File 方法的文件号。

请参阅

ReadFromFile 方法，SaveToFile 方法，SaveToOlelFile 方法

Files 集合

在一个文件夹内的所有 File 对象的集合。

说明

下面的代码举例说明了如何获得一个 Files 集合，以及如何用 ForEach...Next 语句来访问这个集合中的每个文件：

```
SubShowFolderList(folderspec)
Dimfs,f,f1,fc,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFolder(folderspec)
Setfc=f.Files
ForEachf1infc
s=s&f1.name
s=s&vbCrLf
Next
MsgBoxs
EndSub
```

Files 方法

返回一个文件名的集合，这些文件名是由 vbCFFiles 格式（一个

DataObjectFiles 集合）来使用的，该格式反过来又包含 DataObject 对象使用的所有文件名的列表；例如，被用户拖动到 Windows“文件管理器”的文件名或从该处拖出来的文件名。

应用于

DataObject 对象

语法

object.**Files**(*index*)

Files 集合的语法有下面这些部分：

部分	描述
object	对象表达式，其值是 DataObject 对象
index	整数，该整数是某个文件名数组的索引

说明

仅当 DataObject 对象包含类型为 vbCFFiles 的数据时，才可用文件名来对 Files 集合进行填充。DataObject 对象可以包含几个不同的数据类型。通过遍历该集合就可以获取文件名列表。

可对 Files 集合进行填充，所以 VisualBasic 应用程序可作为文件列表的拖放源。

请参阅

DataObject 对象，DataObjectFiles 集合，DataObject 对象（ActiveX 控件），DataObjectFiles 集合（ActiveX 控件）

Files 属性

返回 DataObjectFiles 集合，它依次包含 DataObject 对象所用的全部文件名的列表。（例如拖出或放入 Windows 文件资源管理器的文件名）。

应用于

DataObject 对象

语法

object.**Files**(*index*)

Files 集合语法包含下面部分：

部分	描述
object	对象表达式，其值是 DataObject 对象
index	代表文件名数组索引的整数

说明

只有当 DataObject 对象包含 vbCFFiles 类型的数据时，Files 集合才充满文件名（DataObject 对象可以包含几种不同的数据类型）。可以重复地在集合中检索文件名列表。

可填充 Files 集合以允许 VisualBasic 应用程序作为文件列表的拖放源。

请参阅

DataOjbectFiles 集合

Files 属性

返回由所有 File 对象组成的 Files 集合，这些 File 对象包含在指定的文件夹中——包括设置了隐藏和系统文件属性的那些文件。

语法

object.**Files**

object 总是一个 **Folder** 对象。

说明

下面的代码举例说明了 Files 属性的用法：

```
SubShowFileList(folderspec)
Dimfs,f,f1,fc,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFolder(folderspec)
Setfc=f.Files
ForEachf1infc
s=s&f1.name
s=s&vbCrLf
Next
MsgBoxs
EndSub
```

请参阅

DateCreated 属性，DateLastAccessed 属性，DateLastModified

属性，Drive 属性，Drives 属性，IsRootFolder 属性，Name 属性，(FileSystemObject 对象)，ParentFolder 属性，Path 属性，ShortName 属性，ShortPath 属性，Size 属性，SubFolders 属性，Type 属性

FileSystem 属性

返回指定驱动器所使用的文件系统类型。

应用于

Drive 对象

语法

object.FileSystem

object 总是一个 Drive 对象。

说明

可以得到的返回类型包括 FAT、NTFS、以及 CDFS。

下面的代码举例说明了 FileSystem 属性的用法：

```
SubShowFileSystemType
Dimfs,d,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive("e:")
s=d.FileSystem
MsgBoxs
EndSub
```

FileSystemObject 对象

提供对计算机文件系统的访问。

语法

Scripting.FileSystemObject

说明

下面的代码举例说明了如何使用 FileSystemObject 返回一个 TextStream 对象，该对象是可读或可写的：

```
Setfs=CreateObject("Scripting.FileSystemObject")
Seta=fs.CreateTextFile("c:\testfile.txt",True)
a.WriteLine("Thisisatest.")
a.Close
```

在上面列出的代码中，CreateObject 函数返回 FileSystemObject(fs)。CreateTextFile 方法接着创建文件作为一个 TextStream 对象(a)，而 WriteLine 方法则向创建的文本文件中写入一行文本。Close 方法刷新缓冲区并关闭文件。

属性

Drives 属性

方法

BuildPath 方法, CopyFile 方法, CopyFolder 方法, CreateFolder 方法, CreateTextFile 方法, DeleteFile 方法, DeleteFolder

方法，DriveExists 方法，FileExists 方法，FolderExists 方法，GetDriveName 方法，GetExtensionName 方法，GetFile 方法，GetFileName 方法，GetFolder 方法，GetParentFolderName 方法，GetSpecialFolder 方法，GetTempName 方法，MoveFile 方法，MoveFolder 方法，OpenTextFile 方法

请参阅

Dictionary 对象，TextStream 对象

FileTitle 属性

返回要打开或保存文件的名称（没有路径）。

应用于

CommonDialog 控件（“打开”和“另存为”对话框）

语法

object.**FileTitle**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

当在“文件”对话框中选择一个文件并单击“确定”按钮时，FileTitle 属性就记录一个值，该值将被用于打开或保存所选的文件。

注意如果设置了 cdlOFNNoValidate 标志，则 FileTitle 属性不返回值。

数据类型

String

FillColor 属性

返回或设置用于填充形状的颜色；FillColor 也可以用来填充由 Circle 和 Line 图形方法生成的圆和方框。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，Printers 集合，Form 对象，Forms 集合，PictureBox 控件，Shape 控件

语法

object.**FillColor**[=*value*]

FillColor 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	值或常数，确定填充颜色，“设置值”中有详细描述

设置值

value 的设置值如下：

设置值	描述
标准 RGB 颜色	代码中用 RGB 或 QBColor 函数设置的颜色
系统缺省颜色	对象浏览器中的 VisualBasic(VB)对象库中指定的系统颜色常量，MicrosoftWindows 操作环境替换用户在控制面板中的设置值

缺省情况下，FillColor 设置为 0（黑色）。

说明

除 Form 对象之外，当 FillStyle 属性设置为缺省值 1（透明）时，则忽略 FillColor 设置值。

示例

在这个例子中单击鼠标时，可用随机的 FillColor 和 FillStyle 属性设置值在窗体中构造一个圆。要试用此例，先把代码粘贴到窗体的声明部分，然后按 F5 键，并单击窗体。

```
PrivateSubForm_MouseDown(ButtonAsInteger,ShiftAsInteger,XAsSingle,YAsSingle)
   FillColor=QBColor(Int(Rnd*15))   '选择随机的FillColor.
   FillStyle=Int(Rnd*8)'选择随机的FillStyle.
   Circle(X,Y),250  '画一个圆.
EndSub
```

FillStyle 属性

返回或设置用来填充 Shape 控件、以及由 Circle 和 Line 图形方法生成的圆和方框的模式。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，Printers 集合，Form 对象，Forms 集合，PictureBox 控件，Shape 控件

语法

object.**FillStyle**[=*number*]

FillStyle 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	整数，指定填充样式，“设置值”中有详细描述

设置值

number **的设置值为：**

常数	设置值	描述
VbFSSolid	0	实线
VbFSTransparent	1	（缺省值）透明
VbHorizontalLine	2	水平直线
VbVerticalLine	3	垂直直线
VbUpwardDiagonal	4	上斜对角线
VbDownwardDiagonal	5	下斜对角线
VbCross	6	十字线
VbDiagonalCross	7	交叉对角线

说明

如果 FillStyle 设置为 1（透明），则忽略 FillColor 属性，但是 Form 对象除外。

请参阅

Paint 事件，BorderStyle 属性，FillColor 属性，BorderStyle 属性（ActiveX 控件）

示例

在这个例子中单击鼠标时，可用随机的 FillColor 和 FillStyle 属性的设置值在窗体中显示一个圆。要试用此例，先把代码粘贴到声明部分，然后按 F5 键运行该程序。

```
PrivateSubForm_MouseDown(ButtonAsInteger,ShiftAsInteger,XAsSingle,YAsSingle)
    FillColor=QBColor(Rnd*15)    '选择随机的FillColor.
    FillStyle=Int(Rnd*8)'选择随机的FillStyle.
    Circle(X,Y),250  '画一个圆.
EndSub
```

Filter 函数

返回一个下标从零开始的数组，该数组包含基于指定筛选条件的一个字符串数组的子集。

语法

Filter(*InputStrings*,*Value*[,*Include*[,*Compare*]])

Filter 函数语法有如下几部分：

部分	描述
inputStrings	必需的。要执行搜索的一维字符串数组
value	必需的。要搜索的字符串
include	可选的。Boolean 值，表示返回子串包含还是不包含 Value 字符串。如果 *Include* 是 True，Filter 返回的是包含 *Value* 子字符串的数组子集。如果 *Include* 是 False，Filter 返回的是不包含 *Value* 子字符串的数组子集
compare	可选的。数字值，表示所使用的字符串比较类型。有关其设置，请参阅下面的“设置值”部分

设置值

Compare 参数的设置值如下：

常数	值	描述
vbUseCompareOption	–1	使用 OptionCompare 语句的设置值来执行比较
vbBinaryCompare	0	执行二进制比较
vbTextCompare	1	执行文字比较
vbDatabaseCompare	2	只用于 MicrosoftAccess。基于您的数据库信息来执行比较

说明

如果在 *InputStrings* 中没有发现与 *Value* 相匹配的值，Filter 返回一个空数组。如果 *InputStrings* 是 Null 或不是一个一维数组，则

产生错误。

Filter 函数所返回的数组，其元素数目刚好是所找到的匹配项目数。

请参阅

Replace 函数

Filter 属性（公共对话框）

返回或设置在对话框的类型列表框中所显示的过滤器。

应用于

CommonDialog 控件（“打开”和“另存为”对话框）

语法

object.**Filter**[=*description1*|*filter1*|*description2*|*filter2*...]

Filter 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
description	描述文件类型的字符串表达式
filter	是指定文件名扩展的字符串表达式

说明

过滤器指定在对话框的文件列表框中显示的文件的类型。例如，选择过滤器为*.txt，就显示所有的文本文件。

使用该属性可在对话框显示时提供一个过滤器列表，用它可以进行选择。

使用管道(|)符号(ASCII124)将 *fifter* 与 *description* 的值隔开。管道符号的前后都不要加空格，因为这些空格会被与 fifter 与 description 的值一起显示。

下列代码给出一个过滤器的例子，该过滤器允许选择文本文件或含有位图和图标的图形文件：

```
Text(*.txt)|*.txt|Pictures(*.bmp;*.ico)|*.bmp;*.ico
```

当为一个对话框要指定一个以上的过滤器时，需使用 FilterIndex 属性确定哪一个作为缺省过滤器显示。

数据类型

String

请参阅

FilterIndex 属性

FilterIndex 属性

返回或设置“打开”或“另存为”对话框中一个缺省的过滤器。

应用于

CommonDialog 控件（“打开”,“另存为”对话框）

语法

object.**FilterIndex**[=*number*]

FilterIndex 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
number	是指定缺省过滤器的数值表达式

说明

当使用 Filter 属性为“打开”或“另存为”对话框指定过滤器时，该属性指定缺省的过滤器。

对于所定义的第一个过滤器其索引是 1。

数据类型

Integer

请参阅

Filter 属性（CommanDialog）

Find 方法（VBA 外接程序对象模型）

在活动模块上搜寻一个特定字符串。

应用于

CodeModule 对象

语法

object.**Find**(*target*,*startline*,*startcol*,*endline*,*endcol*[,wholeword][,matchcase][,*patternsearch*])**AsBoolean**

Find 方法语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
target	必需的。一个 String 型数，用来指定欲查找的文本或样式
startline	必需的。一个 Long 型数，用来指定欲搜寻的器始行：如果找到的话，将设为匹配行。第一行为数字 1
startcol	必需的。一个 Long 型数，用来指定欲搜寻的起始列：如果找到的话，将设为匹配列。第一列是数字 1
endline	必需的。一个 Long 型数，表示最后一个匹配的字符串所在的行数。最后一行可以被指定为-1
endcol	必需的。一个 Long 型数，表示最后一个匹配的字符串所在的列数。最后一列可以被指定为-1
wholeword	可选的参数。一个 Boolean 型数，表示是否只匹配整个字。如为 True，表示只匹配整个字。默认为 False
matchcase	可选的参数。Boolean，表示是否进行大小写匹配。如为 True，表示大小写皆匹配。默认为 False
patternsearch	可选的参数。Boolean，表示欲查找的字符串是否为常规表达式。如为 True，表示为常规表达式；默认为 False

说明

如找到匹配项，Find 方法可返回 True，否则返回 False。

matchcase 和 *patternmatch* 参数是互斥的；如果两参数皆为 True，将产生错误。

Find 方法不影响 Find 对话框中的内容。

指定的行和列范围是“或”的关系，因此如果所提供的 endcol 是-1 或行的长度，则能够找到指定的最后一行的样式。

请参阅

GetSelection 方法，Lines 方法，CodePane 对象，ProcBodyLine 属性，ProcCountLines 属性，ProcOfLine 属性，ProcStartLine 属性

示例

下列示例使用 Find 方法来证实线中的特定块，从线 1261 到 1279，其特定代码框内不包括字符串“Tabs.Clear.”

FirstDayOfWeek 属性

设置或返回一个星期第一天。在设计时或运行时都可读/写。

应用于

StdDataFormat 对象

语法

object.**FirstDayOfWeek**[=*value*]

FirstDayOfWeek 属性语法有如下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
value	可选的枚举整型数。其值决定一个星期中哪一天被认为是第一天，如设置值中所述

设置值

value 设置值如下：

常数	设置值	描述
fmtDayUseSystem	0	使用 NLSAPI 设置
fmtSunday	1	星期天，这是缺省值
fmtMonday	2	星期一
fmtTuesday	3	星期二
fmtWednesday	4	星期三
fmtThursday	5	星期四
fmtFriday	6	星期五
fmtSaturday	7	星期六

说明

这个属性仅在格式化日期时使用。在绑定之前设置该属性，能够得到更快的速度。

请参阅

FirstWeekOfYear 属性

FirstWeekOfYear 属性

设置或返回一年中的第一个星期。在设计时或运行时都可读/写。

应用于

StdDataFormat 对象

语法

object.**FirstWeekOfYear**[=*value*]

FirstWeekOfYear 属性语法有如下的部分：

部分	描叙
object	一个对象表达式，其值是“应用于”列表中的一个对象
value	可选的枚举整型数。其值决定一年中的哪个星期被认为是第一个星期，如设置值中所述

设置值

value 的设置值如下：

常数	设置值	描述
fmtWeekUseSystem	0	使用 NLSAPI 设置
fmtFirstJan1	1	包含一月一号的星期，这是缺省值
fmtFirstFourDays	2	一年中第一个至少含有四天的星期
fmtFirstFullWeek	3	一年中的第一个满七天的星期

说明

该属性仅在格式化日期时使用。

请参阅

FirstDagOfWeek 属性

Flags 属性（“颜色”对话框）

返回或设置“颜色”对话框选项。

应用于

CommonDialog 控件

语法

object.**Flags**[=*value*]

Flags 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	如“设置值”中所描述，是为“颜色”对话框指定选项的常数或值

设置值

value 的设置值是：

常数	值	描述
cdlCCFullOpen	&H2	显示全部的对话框，包括定义自定义颜色部分
cdlCCShowHelpButton	&H8	使对话框显示帮助按钮
cdlCCPreventFullOpen	&H4	使定义自定义颜色命令按钮无效并防止定义自定义颜色
cdlCCRGBInit	&H1	为对话框设置初始颜色值

说明

这些常数在对象浏览器的MicrosoftCommonDialog控件(MSComDlg)对象库中列出。

也可以定义所选择的标志。使用启动窗体声明部分的 Const 关键字定义想使用的标志。

使用 Or 运算符可以为一个对话框设置多个标志。例如：

```
CommonDialog1.Flags=&H10&Or&H200&
```

将所希望的常数值相加能产生同样的结果。下例与上例等效：

```
CommonDialog1.Flags=&H210&
```

数据类型

Long

请参阅

Or 运算符，Const 语句

Flags 属性（“字体”对话框）

返回或设置“字体”对话框的选项。

应用于

CommonDialog 控件

语法

object.**Flags**[=*value*]

Flags 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	如“设置值”中所描述，它为“字体”对话框指定选项的常数或值

设置值

value 的设置是：

常数	值	描述
cdlCFANSIOnly	&H400	它指定对话框只允许选择 Windows 字符集的字体。如果该标志被设置，就不能选择仅含符号的字体
cdlCFApply	&H200	它使对话框中的“应用”按钮有效
cdlCFBoth	&H3	使对话框列出可用的打印机和屏幕字体 hDC 属性标识与打印机相关的设备描述体

cdlCFEffects	&H100	**它指定对话框允许删除线，下划线，以及颜色效果**
cdlCFFixedPitchOnly	&H4000	**它指定对话框只能选择固定间距的字体**
cdlCFForceFontExist	&H10000	**如果用户试图选择一个并不存在的字体或样式，显示错误信息框**
cdlCFHelpButton	&H4	**使对话框显示帮助按钮**
cdlCFLimitSize	&H2000	**它指定对话框只能在由** Min **和** Max **属性规定的范围内选择字体大小**
cdlCFNoFaceSel	&H80000	**没有选择字体名称**
cdlCFNoSimulations	&H1000	**它指定对话框不允许图形设备接口**(GDI)**字体模拟**
cdlCFNoSizeSel	&H200000	**没有选择字体大小**
cdlCFNoStyleSel	&H100000	**没有选择样式**
cdlCFNoVectorFonts	&H800	**它指定对话框不允许矢量字体选择**
cdlCFPrinterFonts	&H2	**使对话框只列出由** hDC **属性指定的打印机支持的字体**
cdlCFScalableOnly	&H20000	**它指定对话框只允许选择可缩放的字体**
cdlCFScreenFonts	&H1	**使对话框只列出系统支持的屏幕字体**
dlCFTTOnly	&H40000	**它指定对话框只允许选择** TrueType

		型字体
cdlCFWYSIWYG	&H8000	它指定对话框只允许选择在打印机和屏幕上均可用的字体。如果该标志被设置，则 cdlCFBoth 和 cdlCFScalableOnly 标志也应该设置

说明

这些常数在对象浏览器中的 MicrosoftCommonDialog 控件(MSComDlg)对象库中列出。

也可以定义所选择的标志。可使用启动窗体声明部分的 Const 关键字来定义想使用的标志。例如：

```
ConstReadOnly=&H00000001&
ConstEffects=&H00000100&
```

使用 Or 运算符可以为一个对话框设置多个标志。例如：

```
CommonDialog1.Flags=&H10&Or&H200&
```

将所希望的常数值相加能产生同样的结果。下例与上例等效：

```
CommonDialog1.Flags=&H210&
```

注意在显示“字体”对话框前，必须先将 Flags 属性设置为 cdlCFScreenFonts，cdlCFPrinterFonts，或 cdlCFBoth。否则，会发生字体不存在的错误。

数据类型

Long

请参阅

Or 运算符，Const 语句，Max,Min 属性（CommonDialog）

Flags 属性（“打开”、“另存为”对话框）

为“打开”和“另存为”对话框返回或设置选项。

应用于

CommonDialog 控件

语法

object.**Flags**[=*value*]

Flags 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	如“设置值”中所描述，是为“打开”和“另存为”对话框指定选项的常数或值

设置值

Value 的设置值是：

常数	值	描述
cdlOFNAllowMultiselect	&H200	它指定文件名列表框允许多重选择 运行时，通过按 SHIFT 键以及

		使用 UPARROW 和 DOWNARROW 键可选择多个文件。作完此操作后，FileName 属性就返回一个包含全部所选文件名的字符串。串中各文件名用空格隔开
cdlOFNCreatePrompt	&H2000	当文件不存在时对话框要提示创建文件。该标志自动设置 cdlOFNPathMustExist 和 cdlOFNFileMustExist 标志
cdlOFNExplorer	&H80000	它使用类似资源管理器的打开一个文件的对话框模板。适用于 Windows95 和 WindowsNT4.0
cdlOFNExtensionDifferent	&H400	它指示返回的文件扩展名与 DefaultExt 属性指定的扩展名不一致。如果 DefaultExt 属性是 Null，或者扩展相匹配，或者没有扩展时，此标志不设置。当关闭对话框时，可以检查这个标志的值
cdlOFNFileMustExist	&H1000	它指定只能输入文件名文本框已经存在的文件名。如果该标

		志被设置，则当用户输入非法的文件名时，要显示一个警告。该标志自动设置 cdlOFNPathMustExist 标志
cdlOFNHelpButton	&H10	使对话框显示帮助按钮
cdlOFNHideReadOnly	&H4	隐藏只读复选框。
cdlOFNLongNames	&H200000	使用长文件名
cdlOFNNoChangeDir	&H8	强制对话框将对话框打开时的目录置成当前目录
cdlOFNNoDereferenceLinks	&H100000	不要间接引用外壳链接（也称作快捷方式）。缺省时，选取外壳链接会引起它被外壳间接引用
cdlOFNNoLongNames	&H40000	无长文件名
cdlOFNNoReadOnlyReturn	&H8000	它指定返回的文件不能具有只读属性，也不能在写保护目录下面
cdlOFNNoValidate	&H100	它指定公共对话框允许返回的文件名中含有非法字符
cdlOFNOverwritePrompt	&H2	使“另存为”对话框当选择的文件已经存在时应产生一个信息框，用户必须确认是否覆盖

		该文件
cdlOFNPathMustExist	&H800	它指定只能输入有效路径。如果设置该标志，输入非法路径时，应显示一个警告信息
cdlOFNReadOnly	&H1	建立对话框时，只读复选框初始化为选定。该标志也指示对话框关闭时只读复选框的状态
cdlOFNShareAware	&H4000	它指定忽略共享冲突错误

说明

cdlOFNExplorer 和 cdlOFNNoDereferenceLinks 标志适用于 Windows95 和 WindowsNT4.0。Windows95 中 cdlOFNExplorer 的公共对话框使用字符作为分隔符；而在没有 Windows95 外壳的 WindowsNT 的早期版本中，多重选择是使用空格作为分隔符（因而不能支持长文件名）。

无论是在 WindowsNT4.0 还是在 Windows95 中，如果不选取 cdlOFNAllowMultiselect 标志，cdlOFNExplorer 和 cdlOFNLongNames 标志均没有意义，并且实际上是缺省值。

无论是在 WindowsNT4.0 还是在 Windows95 中，如果 cdlOFNallowMultiselect 标志被单独使用，都不能支持长文件名。这是因为多重文件名要复现空格分隔符，而长文件名也可能包括空格符。在 WindowsNT3.5 中，无法避免这种情况。如果使用 cdlOFNAllowMultiselect，就不能看到长文件名。如果在 Windows95

中添加 cdlOFNExplorer 标志，就可以既能进行文件多选，又能看到长文件名。但是，这些文件名显现空字符分隔符，而不是空格分隔符隔开。因此，cdlOFNAllowMultiselect 和 cdlOFNExplorer 一起使用时，在 Windows95 和 WindowsNT4.0 中需要不同的文件名所得结果的语法分析。

这些常数在对象浏览器中的 MicrosoftCommonDialog 控件(MSComDlg)对象库中列出。

也可以定义所选择的标志。应使用启动窗体声明部分的 Const 关键字来定义想使用的标志。例如：

```
ConstReadOnly=&H00000001&
ConstEffects=&H00000100&
CommonDialog1.Flags=&H10&Or&H200&
```

将所需常数值相加能产生同样的结果。下例与上例等效：

```
CommonDialog1.Flags=&H210&
```

数据类型

Long

请参阅

Const 语句，DefaultExit 属性，FileName 属性

Flags 属性（“打印”对话框）

返回或设置“打印”对话框的选项。

应用于

CommonDialog 控件

语法

object.**Flags**[=*value*]

Flags 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	如“设置值”中所描述，为“打印”对话框指定选项的常数或值

设置值

value 的设置值是：

常数	值	描述
cdlPDAllPages	&H0	返回或设置全部页选项按钮的状态
cdlPDCollate	&H10	返回或设置分页复选框的状态
cdlPDDisablePrintToFile	&H80000	使打印到文件复选框无效
cdlPDHelpButton	&H800	要求对话框显示帮助按钮
cdlPDHidePrintToFile	&H10000	隐藏打印到文件复选框

	0	
cdlPDNoPageNums	&H8	使页选项按钮和相关的编辑控件无效
cdlPDNoSelection	&H4	使选择选项按钮无效
cdlPDNoWarning	&H80	防止没有缺省打印机时显示警告信息
cdlPDPageNums	&H2	返回或设置页选项按钮的状态
cdlPDPrintSetup	&H40	使系统显示“打印设置”对话框而不是“打印”对话框
cdlPDPrintToFile	&H20	返回或设置打印到文件复选框的状态
cdlPDReturnDC	&H100	为该对话框中选择的打印机返回一个设备描述体。设备描述体返回到对话框的 hDC 属性中
cdlPDReturnDefault	&H400	返回缺省的打印机名称
cdlPDReturnIC	&H200	为该对话框中选择的打印机返回一个信息上下文。信息上下文提供了一个不用建立设备描述体就能得到设备信息的快速方法。信息上下文返回到对话框的 hDC 属性中
cdlPDSelection	&H1	返回或设置选择选项按钮的状态。如果 cdlPDPageNums 或

	cdlPDSelection 均未指定，全部选项按钮就处于被选状态
cdlPDUseDevModeCop ies	&H40000 如果打印机驱动程序不支持多份数打印，则设置该属性将使打印对话中的份数微调控件的数值无效。如果驱动程序支持多份数打印，则设置该属性指示对话框将所要的份数值存放在 Copies 属性中

说明

这些常数在对象浏览器的 MicrosoftCommonDialog 控件(MSComDlg)对象库中列出。

也可以定义所选择的标志。使用启动窗体声明部分的 Const 关键字来定义想使用的标志。例如：

```
ConstReadOnly=&H00000001&
ConstEffects=&H00000100&
```

使用 Or 运算符可以为一个对话框设置多个标志。如：

```
CommonDialog1.Flags=&H10&Or&H200&
```

将所希望的常数值相加能产生同样的结果。下例与上例等效：

```
CommonDialog1.Flags=&H210&
```

数据类型

Long

请参阅

Or 运算符，Const 语句，Copies 属性，hDC 属性

Folder 对象

提供对一个文件夹所有属性的访问。

说明

下面的代码举例说明了如何获得一个 Folder 对象，以及如何返回它的一个属性：

```
SubShowFolderInfo(folderspec)
Dimfs,f,s,
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFolder(folderspec)
s=f.DateCreated
MsgBoxs
EndSub
```

属性

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性（FileSystemObject 对象）ParentFolder 属性，Path 属性（FileSystemObject 对象），ShortName 属性，ShortPath

属性，Size 属性（FileSystemObject 对象），SubFolders 属性

方法

Copy 方法，CreateTextFile 方法，Delete 方法，Move 方法

请参阅

Drive 对象，Drives 集合，File 对象，Files 集合，FileSystemObject 对象

FolderExists 方法

如果指定的文件夹存在返回 True，不存在返回 False。

应用于

FileSystemObject 对象

语法

object.**FolderExists**(*folderspec*)

FolderExists 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 *FileSystemObject* 对象的名字
folderspec	必需的。要确定是否存在的文件夹名字。如果认为文件夹不在当前文件夹中，必须提供一个完整的路径说明（绝对的或相对的）

请参阅

DriveExists 方法，FileExists 方法

Folders 集合

包含在一个 Folder 对象内的所有 Folder 对象的集合。

说明

下面的代码举例说明了如何获得一个 Folders 集合，以及如何用 ForEach...Next 语句来访问该集合中的每个 Folder：

```
SubShowFolderList(folderspec)
Dimfs,f,f1,fc,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFolder(folderspec)
Setfc=f.SubFolders
ForEachf1infc
s=s&f1.name
s=s&vbCrLf
Next
MsgBoxs
EndSub
```

属性

Count 属性，Item 属性

方法

AddFolders 方法

请参阅

Drive 对象，Drives 集合，File 对象，Files 集合，FileSyst

emObject 对象，Folder 对象

Font 对象

Font 对象包含格式化文本所需要的信息；这些文本，有的要在应用程序界面中显示，有的要打印输出。

语法

Font

说明

经常用显示文本的对象（例如 Form 对象或 Printer 对象）的 Font 属性来标识 Font 对象。

不能使用类似于 DimXAsNewFont 的代码建立 Font 对象。而必须使用 StdFont 对象建立 Font 对象，如下面这种方式：

```
DimXAsNewStdFont
```

如果把名为 Text1 的 TextBox 控件放到窗体上，则可以使用 Set 语句动态改变其 Font 对象，如下例所示：

```
DimXAsNewStdFont
X.Bold=True
X.Name="Arial"
SetText1.Font=X
```

属性

Charset 属性，Bold 属性，Italic 属性，Size 属性（Font），

StrikeThrough 属性，Underline 属性，Weight 属性，Name 属性

事件

FontChanged 事件

请参阅

TextHeight 方法，TextWidth 方法，Fonts 属性，Font 属性，Font 属性（MSCart）

Font 属性

返回一个 Font 对象。

语法

object.**Font**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

为了标识一个具体的要使用其属性的 Font 对象应使用一个对象的 Font 属性。例如，下面的代码将改变一个 Font 对象的 Bold 属性设置，该 Font 对象被 TextBox 对象的 Font 属性所标识：

```
txtFirstName.Font.Bold=True
```

请参阅

Font 对象

FontBold、FontItalic、FontStrikethru、FontUnderline 属性

按下述格式返回或设置字体样式：Bold、*Italic*、~~Strikethru~~ 和 Underline。

注意包含 FontBold、FontItalic、FontStrikethru 和 FontUnderline 属性是为了 CommonDialog 控件的使用，并与早期的 VisualBasic 版本保持兼容。如果需要其它的功能，请使用新的 Font 对象属性（对 CommonDialog 控件不可用）。

应用于

ADOData 控件，ImageCombo 控件，ListView 控件，StatusBar 控件，TabStrip 控件，MonthView 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，SSTab 控件，RemoteData 控件，RichTextBox 控件，Data 控件，DataReport 对象，Function 控件（DataReportDesigner），Label 控件（DataReportDesigner），TextBox 控件（DataReportDesigner），PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，Printers 集合，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

object.**FontBold**[=*boolean*]

object.**FontItalic**[=*boolean*]

object.**FontStrikethru**[=*boolean*]

object.**FontUnderline**[=*boolean*]

FontBold、FontItalic、FontStrikethru 和 FontUnderline 属性语法包含下面部分:

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指定字体样式，“设置值”中有详细说明

设置值

boolean 的设置值如下：

设置值	描述
true	（除 CommonDialog 控件外，FontBold 的缺省值）使用该样式格式
false	（FontItalic、FontStrikethru 和 FontUnderline，以及 CommonDialog 控件中的 FontBold 的缺省值）不使用该样式格式

说明

可以通过这些字体属性，在设计时用属性窗口或在运行时使用代码来格式化文本。对于 Picturebox 控件及 Form 和 Printer 对象，设置这些属性不会影响在控件或对象上已经绘出的图片和文本。对于其它控件，改变字体将会在屏幕上立刻生效。

若和 CommonDialog 控件一起使用这些属性，必须将 Effects 标志置

位。

注意 VisualBasic 中可用的字体取决于系统的配置、显示设备和打印设备。与字体相关的属性只能设置为真正存在的字体的值。一般来说，在用 FontSize、FontBold、FontItalic、FontStrikethru 和 FontUnderline 属性来设置大小和样式属性前，要先改变 FontName 属性。然而，在设置的 TrueType 字体小于 8 磅时，要用 FontSize 属性设置字体大小，再设置 FontName 属性，然后用 FontSize 属性再一次设置字体大小。MicrosoftWindows 运行环境对于小于 8 磅的 TrueType 字体使用不同的字体。

示例

这个例子中，每单击一次鼠标，就将窗体中的文本置成两种样式组合之一。要调试这个例子，请将代码粘贴到窗体的声明部分，然后按 F5 键并单击窗体。

```
PrivateSubForm_Click()
   FontStrikethru=NotFontStrikethru '转换 strikethrough。
   FontItalic=NotFontItalic '转换字体样式。
   Print"Nowisthetime!"'输出文本。
EndSub
```

FontChanged 事件

运行时当 stdFont 对象的一个属性改变时发生。

应用于

Font 对象

语法

PrivateSub*object*_FontChanged(ByVal*PropertyName*AsString)

FontChanged 事件语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
propertyName	一个字符串，用来返回 stdFont 对象已改变的属性名。可能的值包括 Bold、Italic、Name 和 Size

说明

只有对 stdFont 对象并且 stdFont 对象是用 WithEvents 关键字声明的，FontChanged 事件才会发生。一个声明为 Font 而不是声明为 stdFont 的对象不能用 WithEvents 关键字，否则将产生一个自动化错误。FontChanged 事件主要对用户控件非常有用。

FontCount 属性

返回或设置当前显示设备或活动打印机可用的字体。

应用于

Screen 对象，Printer 对象，Printers 集合

语法

object.FontCount

object 所在处代表一个对象表达式，其值是“应用于”列表中的

一个对象。

说明

可将该属性和 Fonts 属性一起使用，来查看屏幕或打印机可用字体的列表。VisualBasic 可用字体随系统配置、显示设备和打印设备的不同而不同。

请参阅

FontName 属性，Fonts 属性

示例

这个例子在 ListBox 控件中显示打印机字体的列表。要调试此例，把代码粘贴到名叫 List1 的 ListBox 控件的窗体的声明部分，然后按下 F5 键，并单击窗体。

```
PrivateSubForm_Click()
   DimI'声明变量.
   ForI=0ToPrinter.FontCount-1 '确定字体数.
       List1.AddItemPrinter.Fonts(I)    '把每一种字体放进列表框.
   NextI
EndSub
```

FontName 属性

返回或设置在控件中或在运行时画图或打印操作中，显示文本所用的字体。

注意包含 FontName 属性是为了和 CommonDialog 控件一起使用，

以及与先前的 VisualBasic 版本兼容。对于其它的功能，请使用新的 Font 对象属性（对 CommonDialog 控件不可用）。

应用于

CommonDialog 控件（FontDialog），PropertyPage 对象，User Control 对象，UserDocument 对象，Printer 对象，Printers 集合，CheckBox 控件，ComboBox 控件，CommandButton 控件，Dir ListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Label 控件，ListBox 控件，Option Button 控件，PictureBox 控件，TextBox 控件

语法

object.**FontName**[=*font*]

FontName 属性语法包括下列组成部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
font	字符串表达式，指定所用的字体名

说明

该属性的缺省值取决于系统，VisualBasic 中可用的字体取决于系统的配置、显示设备和打印设备。与字体相关的属性只能设置为真正存在的字体的值。

一般来说，用 FontSize、FontBold、FontItalic、FontStrikethru 和 FontUnderline 属性来设置大小和样式属性前，要先改变 FontName 属性。

注意在运行时，可以用 FontCount 和 Fonts 属性获得系统可用字体的信息。

请参阅

TextHeight 方法，TextWidth 方法，FontBold,FontItalic,FontStrikethru,FontUnderline 属性，FontSize 属性，FontCount 属性，Fonts 属性

示例

这个例子使用指定字体输出每种字体的名称。要调试这个例子，请将代码粘贴到窗体的声明部分。按 F5 键运行该程序，然后单击窗体。每单击一次窗体就输出一个字体的名称。

```
PrivateSubForm_Click()
   StaticI                  '声明变量
   DimOldFont
   OldFont=FontName                 '保留原来的字体
   FontName=Screen.Fonts(I)             '改变到新的字体
   PrintScreen.Fonts(I)             '输出字体的名称
   I=I+1                '计数器增一
   IfI=FontCountThenI=0             '重新开始
   FontName=OldFont      '恢复原来的字体
EndSub
```

Fonts 属性

返回当前显示设备或活动打印机可用的所有字体名。

应用于

Screen 对象，Printer 对象，Printers 集合

语法

object.**Fonts**(*index*)

Fonts 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
index	介于 0 和 FontCount?1 之间的一个整型值

说明

Fonts 属性和 FontCount 属性一起使用，FontCount 属性返回对象可用的字体名的数量。VisualBasic 可用字体随系统配置、显示设备和打印设备的不同而不同。利用 Fonts 和 FontCount 属性可以获得有关屏幕或打印机可用字体的信息。

请参阅

TextHeight 方法，TextWidth 方法，FontBold,FontItalic,FontStrikethru,FontUnderline 属性，FontName 属性，FontSize 属性

示例

这个例子在 ListBox 控件中显示打印机字体的列表。要试用此例，先把代码粘贴到名叫 List1 的 ListBox 控件的窗体的声明部分。

然后按 F5 键运行该程序，再单击窗体。

```
PrivateSubForm_Click()
   DimI'Declarevariable.
   ForI=0ToPrinter.FontCount-1  '确定字体数.
       List1.AddItemPrinter.Fonts(I)     '把每一种字体放进列表框.
   NextI
EndSub
```

FontSize 属性

返回或设置在控件中或在运行时画图或打印操作中，显示文本所用的字体的大小。

注意包含 FontSize 属性是为了和 CommonDialog 控件一起使用，以及与以前的 VisualBasic 版本兼容。对于其它的功能，请使用新的 Font 对象属性（对 CommonDialog 控件不可用）。

应用于

CommonDialog 控件（FontDialog），PropertyPage 对象，User Control 对象，UserDocument 对象，数据控件，Printer 对象，Printers 集合，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

object.**FontSize**[=*points*]

FontSize 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
points	数值表达式，用磅为单位指定所用字体的大小

说明

用该属性以所要的字体格式化文本。缺省值由系统决定。要改变缺省值，以磅为单位指定字体尺寸。

FontSize 的最大值为 2160 磅。

注意 VisualBasic 中可用的字体取决于系统的配置、显示设备和打印设备。与字体相关的属性只能设置为真正存在的字体的值。一般来说，用 FontSize、FontBold、FontItalic、FontStrikethru 和 FontUnderline 属性来设置大小和样式属性前，应该先改变 FontName 属性。然而，在设置 TrueType 字体尺寸小于 8 磅时，应用 FontSize 属性来设置字体大小，然后设置 FontName 属性，用 FontSize 属性再一次设置字体大小。MicrosoftWindows 运行环境对于小于 8 磅的 TrueType 字体使用不同的字体。

请参阅

FontBold,FontItalic,FontStrickthru,FontUnderline 属性，FontName 属性，FontCount 属性，Fonts 属性

示例

这个例子中，每单击一次鼠标，就使用两种不同大小的字体在窗

体中输出文本。要调试这个例子，请将代码粘贴到窗体的声明部分。按 F5 键运行该程序，然后单击窗体。

```
PrivateSubForm_Click()
   FontSize=24 '设置字体大小(FontSize)。
   Print"Thisis24-pointtype."  '使用大字体输出。
   FontSize=8  '设置 FontSize.
   Print"Thisis8-pointtype"'使用小字体输出。
EndSub
```

FontTransparent 属性

返回或设置一个值，该值用来决定是 Form 或 Printer 对象还是 PictureBox 控件上的背景文本和图形被显示在字符周围的空区。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**FontTransparent**[=*boolean*]

FontTransparent 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定背景文本和图形状态的布尔表达式，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
true	（缺省值）允许背景图形和文本以某种字体在字符的周围空区显示出来
false	屏蔽在字体字符周围已有的背景图形和文本

说明

在设计时使用“属性”窗口或在运行时使用代码来设置 FontTransparent。在运行时改变 FontTransparent 不会影响已经画到 Form、Printer、或 PictureBox 上的图形和文本。

请参阅

Bold 属性，Italic 属性，Size 属性(Font)，StrikeThrough 属性，Underline 属性，Weight 属性，Name 属性

示例

这个例子在 PictureBox 控件中的图形的顶部附打印文本。在窗体中放一个 PictureBox，将其 AutoSize 属性设为 True，加载具有一个位图文件（.bmp）的 Picture 属性。要调试此例，可以先将该代码粘

贴到一个窗体的声明部分中，然后按下 F5 键并双击此窗体。

```
PrivateSubForm_Click()
    '转换属性。
    Picture1.FontTransparent=NotPicture1.FontTransparent
Picture1.Print"DemoofFontTransparentproperty."
EndSub
```

ForEach...Next 语句

针对一个数组或集合中的每个元素，重复执行一组语句。

语法

For Each *element* **In** *group*
[*statements*]
[**Exit For**]
[*statements*]
Next [*element*]

For...Each...Next 语句的语法具有以下几个部分：

部分	描述
element	必需的。用来遍历集合或数组中所有元素的变量。对于集合来说，*element* 可能是一个 Variant 变量、一个通用对象变量或任何特殊对象变量。对于数组而言，*element* 只能是一个 Variant 变量
group	必需的。对象集合或数组的名称（用户定义类型的数组除外）
statements	可选的，针对 *group* 中的每一项执行的一条或多条语句

说明

如果 gruop 中至少有一个元素，就会进入 For...Each 块执行。一旦进入循环，便先针对 *group* 中第一个元素执行循环中的所有语句。如果 *group* 中还有其它的元素，则会针对它们执行循环中的语句，当 *group* 中的所有元素都执行完了，便会退出循环，然后从 Next 语句之后的语句继续执行。

在循环中可以在任何位置放置任意个 ExitFor 语句，随时退出循环。ExitFor 经常在条件判断之后使用，例如 I & T ，并将控制权转移到紧接在 Next 之后的语句。

可以将一个 For...Each...Next 循环放在另一个之中来组成嵌套式 For...Each...Next 循环。但是每个循环的 *element* 必须是唯一的。

注意如果省略 Next 语句中的 *element*，就像 *element* 存在时一样执行。如果 Next 语句在它相对应的 For 语句之前出现，则会产生

错误。

不能在 For...Each...Next 语句中使用用户自定义类型数组，因为 Variant 不能包含用户自定义类型。

请参阅

Do...Loop 语句，Exit 语句，For...Next 语句，While...end 语句

示例

本示例使用 ForEach...Next 语句搜寻集合中的所有成员的 Text 属性，查找“Hello”字符串。示例中，MyObject 是面向文本的对象，并且是 MyCollection 集合的成员，这两个名字只是为示范目的而使用的通用名称而已。

```
DimFound,MyObject,MyCollection
Found=False     '设置变量初始值。
ForEachMyObjectInMyCollection  '对每个成员作一次迭代。
   IfMyObject.Text="Hello"Then  '如果 Text 属性值等于“Hello”。
      Found=True   '将变量 Found 的值设成 True。
      ExitFor '退出循环。
   EndIf
Next
```

For...Next 语句

以指定次数来重复执行一组语句。

语法

For*counter*=*start***To***end*[**Step***step*]

[*statements*]

[**ExitFor**]

[*statements*]

Next[*counter*]

For…Next 语句的语法具有以下几个部分：

部分	描述
counter	**必需的。用做循环计数器的数值变量。这个变量不能是布尔或数组元素**
start	**必需的。**counter **的初值**
end	**必需的，**counter **的终值**
step	**可选的。**counter **的步长。如果没有指定，则** step **的缺省值为** 1
statements	**可选的。放在** For **和** Next **之间的一条或多条语句，它们将被执行指定的次数**

说明

step **参数可以是正数或负数。***step* **参数值决定循环的执行情况，如下所示：**

值	循环执行，如果
正数或 0	*counter<=end*
负数	*counter>=end*

当所有循环中的语句都执行后，*step* **的值会加到** *counter* **中。此时，循环中的语句可能会再次执行（基于循环开始执行时同样的**

测试），也可能是退出循环并从 Next 语句之后的语句继续执行。

提示在循环中改变 *counter* 的值，将会使程序代码的阅读和调试变得更加困难。

循环中可以在任何位置放置任意个 ExitFor 语句，随时退出循环。ExitFor 经常在条件判断之后使用，例如 If...Then，并将控制权转移到紧接在 Next 之后的语句。

可以将一个 For...Next 循环放置在另一个 For...Next 循环中，组成嵌套循环。不过在每个循环中的 *counter* 要使用不同的变量名。下面的体系结构是正确的：

```
For I=1To10
   For J=1To10
      For K=1To10
         ...
      NextK
   NextJ
NextI
```

注意如果省略 Next 语句中的 *counter*，可以像 *counter* 存在时一样执行。但如果 Next 语句在它相对应的 For 语句之前出现，则会产生错误。

示例

本示例使用 For...Next 语句创建一个字符串，其内容为由 0 到 9 的

十个数字所组成的字符串，每个字符串之间用空格隔开。外层循环使用一个变量当作循环计数器，每循环一次，变量值减一。

```
DimWords,Chars,MyString
ForWords=10To1Step-1    '建立 10 次循环。
   ForChars=0To9     '建立 10 次循环。
      MyString=MyString&Chars  '将数字添加到字符串中。
   NextChars     'Incrementcounter
   MyString=MyString&""'添加一个空格。
NextWords
```

ForcePageBreak 属性

返回或设置一个值，决定分页如何出现。

应用于

Section 对象（数据报表生成器）

语法

object.**ForcePageBreak**[=*integer*]

ForcePageBreak 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
integer	可选的。一个数值表达式，指定分页如何出现，如在设置值中所示

设置值

对 integer 的设置如下：

常数	值	描述
rptPageBreakNone	0	（缺省的）没有分页
rptPageBreakBefore	1	在当前部分前面出现分页
rptPageBreakAfter	2	在当前部分后面出现分页
RptPageBreakBeforeAndAfter	3	在当前部分的前面、后面都出现分页

说明

分页并不一定在每一部分上固定出现。下面的表格显示了根据部分的不同而出现的例外情况。

部分	忽略
报表标头	rptPageBreakBefore
页标头	rptPageBreakBefore
	rptPageBreakAfter
	rptPageBreakBeforeAndAfter
页脚	rptPageBreakBefore
	rptPageBreakAfter
	rptPageBreakBeforeAndAfter

Form 对象、Forms 集合

Form 对象是窗口、或者对话框，它组成应用程序用户界面的一部分。

Forms 集合是一个集合，它的元素代表每一个在应用程序中加载的窗体。集合包括应用程序的 MDI 窗体中,MDI 子窗体和非 MDI 窗体。Forms 集合只有一个属性 Count，指定集合中元素的数目。

语法

Form

Forms(*index*)

index 所在处是一个整数，变化范围从 0 到 Forms.Count-1。

说明

在应用程序中可以使用 Forms 集合在所有装载的窗体之间迭代。它标识一个命名为 Forms 的内在全局变量。可以把 Forms(*index*) 传给函数，指定其参数为 Forms 类。

窗体有一些属性确定了它们的外观，例如位置、大小、颜色；这些属性还确定了它们的行为，例如是否可调整大小。

窗体还可以对用户初始化或系统触发的事件作出反应。例如，可以在窗体的 Click 事件过程中编写代码，从而通过单击窗体改变窗体的颜色。

除了属性和事件外，还可以通过代码，使用方法来操作窗体。例如，可以使用 Move 方法改变窗体的位置和大小。

一种称作 MDI 窗体的特殊窗体还包含 MDI 子窗体。MDI 窗体由“插

入”菜单的“添加 MDI 窗体”命令产生；在“文件”菜单中选择“新建窗体”，然后将 MDIChild 属性设置成 True，这样就创建了 MDI 子窗体。

在代码中使用 Dim、Set 和 Static 语句里的 New 关键字可以创建多个窗体实例。

在设计窗体时，设置 BorderStyle 属性定义窗体的边框，设置 Caption 属性把文本放入标题栏。可以在代码中使用 Hide 和 Show 方法使窗体在运行时可见或不可见。

注意将 BorderStyle 设置为 0 就会删除边框。如果希望窗体有边框而没有标题栏、控制菜单框、最大化按钮和最小化按钮，则应从窗体 Caption 属性中删除所有文本，同时将窗体的 ControlBox、MaxButton 和 MinButton 属性设置为 False。

Form 是 Object 数据类型。在将变量设置成一种窗体的实例之前，可以先声明变量的类型为 Form，并在设计时声明这种窗体的实例。与此相似，可以把参数以 Form 类型传给过程。

窗体还可以作为 DDE 对话中的资源，通过 Label、PictureBox 或者 TextBox 控件提供数据。

可以使用 Controls 集合访问 Form 中的控件集合。例如，可以使用如下代码隐藏 Form 中的控件：

```
ForEachControlinForm1.Controls
   Control.Visible=False
NextControl
```

属性

RightToLeft 属性，Controls 属性，Moveable 属性，Palette 属性，PaletteMode 属性，StartUpPosition 属性，StartUpobject 属性，NegotiateMenus 属性，OLEDropMode 属性，BackColor ,ForeColor 属性,BorderStyle 属性,ControlBox 属性,DrawWidth 属 性 ， FontBold,FontItalic,FontStrikethru,FontUnderline 属性，FontName 属性，FontSize 属性，Height,Width 属性，Icon 属性，Left,Top 属性，MaxButton 属性，MinButton 属性，Picture 属性，Tag 属性，Visible 属性，AutoRedraw 属性，ClipControls 属性，CurrentX,CurrentY 属性，DrawMode 属性，DrawStyle 属性，FillColor 属性，FillStyle 属性，Hdc 属性，hWnd 属性，Image 属性，KeyPreview 属性，LinkMode 属性，LinkTopic 属性，MDIChild 属性，MouseIcon 属性，MoUsePointer 属性，ScaleHeight,ScaleWidth 属性，ScaleLeft,ScaleTop 属性，ScaleMode 属性，WindowState 属性，ActiveControl 属性，Appearance 属性，FontTransparent 属性，Caption 属性，Count 属性（VB 集合），Enabled 属性，HelpContextID 属性，Name 属性，Font 属性，WhatsThisButton 属性，WhatsThisHelp 属性，ShowInTaskbar 属性

方法

Refresh 方法，SetFocus 方法，Circle 方法，Line 方法，Pset 方法，Cls 方法，Hide 方法，Move 方法，PaintPicture 方法，Point 方法，PopupMenu 方法，PrintForm 方法，Scale 方法，ScaleX，ScaleY 方法，Show 方法，TextHeight 方法，TextWidth 方法，Zorder

方法，OLEDrag 方法，WhatsThisMode 方法，Add 方法，Print 方法

请参阅

UserDocument 对象，Hide 方法，Move 方法，Label 控件，MDIForm 对象，PictureBox 控件，TextBox 控件，Controls 集合，MDIChild 属性，Count 属性（VB 集合），Count 属性（ActiveX 控件）

Format 事件（StdDataFormat 对象）

在 StdDataFormat 对象格式化值之后发生。

应用于

StdDataFormat 对象

语法

Sub*object*_**Format**(**ByRef** *datavalue* **As** *StdDataValue*)

Format 事件语法有如下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
datavalue	StdDataValue 对象

说明

Format 事件允许您进行 StdDataFormat 对象标准设置所不能完成的格式化操作。

Format 函数

返回 Variant(String)，**其中含有一个表达式，它是根据格式表达式中的指令来格式化的。**

语法

Format(*expression*[,*format*[,*firstdayofweek*[,*firstweekofyear*]]])

Format 函数的语法具有下面几个部分：

部分	说明
expression	**必需的。任何有效的表达式**
format	**可选的。有效的命名表达式或用户自定义格式表达式**
firstdayofweek	**可选的。常数，表示一星期的第一天**
firstweekofyear	**可选的。常数，表示一年的第一周**

设置值

firstdayofweek **参数有下面设置：**

常数	值	说明
vbUseSystem	0	使用 NLSAPI 设置
vbSunday	1	星期日（缺省）
vbMonday	2	星期一
vbTuesday	3	星期二
vbWednesday	4	星期三
vbThursday	5	星期四
vbFriday	6	星期五
vbSaturday	7	星期六

firstweekofyear 参数有下面设置：

常数	值	说明
vbUseSystem	0	使用 NLSAPI 设置
vbFirstJan1	1	从包含一月一日的那一周开始（缺省）
vbFirstFourDays	2	从本年第一周开始，而此周至少有四天在本年中
vbFirstFullWeek	3	从本年第一周开始，而此周完全在本年中

说明

格式化	作法
数字	使用预先定义的命名数值格式或创建用户自定义数值格式
日期和时间	使用预先定义的命名日期/时间格式或创建用户自定义日期/时间格式
日期和时间序数	使用日期和时间格式或数值格式
字符串	创建自定义的字符串格式

如果在格式化数字时没有指定 *format*，Format 会提供与 Str 函数类似的功能，尽管它是国际化的。但是，用 Format 函数将正数格式化为字符串时，不包含保留作为符号位的前面的空格；用 Str 函转换则保留。

请参阅

类型转换函数，Str 函数

示例

本示例显示用 Format 函数做格式化输出的不同用法。对于日期分隔号（/），时间分隔号（:），以及 AM/PM 等文本而言，系统实际的输出显示格式取决于代码运行的设置，当日期和时间在开发环境中显示时，采用代码地区的短的日期格式和短的时间格式，它也许与代码地区有差别。下例假设用美式英语。

MyTime 及 MyDate 在开发环境下，使用系统的短日期设置显示出来的。

```
DimMyTime,MyDate,MyStr
```

```
MyTime=#17:04:23#
MyDate=#January27,1993#

'以系统设置的长时间格式返回当前系统时间。
MyStr=Format(Time,"LongTime")

'以系统设置的长日期格式返回当前系统日期。
MyStr=Format(Date,"LongDate")

MyStr=Format(MyTime,"h:m:s")    '返回"17:4:23"。
MyStr=Format(MyTime,"hh:mm:ssAMPM")     '返回"05:04:23PM"。
MyStr=Format(MyDate,"dddd,mmmdyyyy")    '返回
"Wednesday,Jan271993"。
'如果没有指定格式，则返回字符串。
MyStr=Format(23)    '返回"23"。

'用户自定义的格式。
MyStr=Format(5459.4,"##,##0。00")   '返回"5,459.40"。
MyStr=Format(334。9,"###0。00")'返回"334.90"。
MyStr=Format(5,"0。00%")    '返回"500.00%"。
MyStr=Format("HELLO","<")   '返回"hello"。
MyStr=Format("Thisisit",">")    '返回"THISISIT"。
```

Format 属性（StdDataFormat 对象）

设置或返回一个格式化字符串。在设计时或运行时都可读/写。

应用于

StdDataFormat 对象

语法

object.**Format**=*formatstring*

Format 属性的语法有如下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
formatstring	必需的。指定所需格式的字符串。Format 属性所支持的格式表达式与 Format 函数所支持的相同

说明

如果您使用复杂格式字符串，那么就需要您自己来去除该数据的格式化。

格式化过程发生在 Format 事件之前。而去除格式化的过程由数据库处理，并且发生在 Unformat 事件之后。

请参阅

Format 事件，Unformat 事件，Format 函数

Format 属性

当将数据发送到创建对象的应用程序及从该应用程序获取数据

时，返回或设置格式，在设计时不可用。

应用于

OLE 包容器控件

语法

object.**Format**[=*format*]

Format 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
format	字符串表达式，用于指定使用 Data 和 DataText 属性的格式

说明

使用 ObjectAcceptFormats、ObjectAcceptFormatsCount、ObjectGetFormats 和 ObjectGetFormatsCount 属性，为指定的对象类获取可接受的数据格式列表。

很多提供对象的应用程序只能支持一两种格式。例如，MicrosoftDraw 只能接受 CF_METAFILEPICT 格式。CF_METAFILEPICT 格式虽然与内在的常数 vbCFMetafile（数字值 3）类似，但实际上它是一个字符串文字，按以下格式赋值，这个常数在“对象浏览器”的 VisualBasic(VB)对象库中定义：

```
Ole1.Format="CF_METAFILEPICT"
```

在很多情况下，对象可以接受(ObjectAcceptFormats)的格式列表，

与对象可以提供(ObjectGetFormats)的格式列表是不同的。

请参阅

Data 属性，DataText 属性，ObjectAccepFormatsCount 属性，ObjectGetFormats 属性，ObjectGetFormatsCount 属性，MaskedEdit 控件，Format 函数

FormatCurrency 函数

返回一个货币值格式的表达式，它使用系统控制面板中定义的货币符号。

语法

FormatCurrency(*Expression*[,*NumDigitsAfterDecimal*[,*IncludeLeadingDigit*[,*UseParensForNegativeNumbers*[,*GroupDigits*]]]])

FormatCurrency 函数语法有如下几部分：

部分	描述
expression	必需的。要格式化的表达式
numDigitsAfterDecimal	可选的。数字值，表示小数点右边的显示位数缺省值为 1，表示使用计算机的区域设置值
includeLeadingDigit	可选的。三态常数，表示小数点前是否显示一个零。关于其值，请参阅“设置值”部分
UseParensForNegativeNumbers	可选的。三态常数，表示是否把负数值放在园括号内。关于其值，请参阅“设置值”部分
groupDigits	可选的。三态常数，表示是否用组分隔符对数字进行分组，组分隔符由计算机的区域设置值指定。关于其值，请参阅“设置值”部分

设置值

IncludeLeadingDigit、UseParensForNegativeNumbers 和 GroupDigits 参数的设置值如下：

常数	值	描述
TristateTrue	–1	True
TristateFalse	0	False
TristateUseDefault	–2	使用计算机区域设置中的设置值。

说明

当忽略一个或多个选项参数时，被忽略的参数值由计算机的区域设置值提供。

货币符号相对货币值的位置由计算机的区域设置值确定。

注意除起始的零外，所有设置值信息都来自“区域设置”的“货币”选项卡，起始的零来自“数字”选项卡。

请参阅

FormatDataTime 函数，FormatNumber 函数，FormatPercent 函数

FormatDateTime 函数

返回一个日期或时间格式的表达式。

语法

FormatDateTime(*Date*[,*NamedFormat*])

FormatDateTime 函数语法有如下几部分：

部分	描述
Date	必需的。要被格式化的日期表达式
NamedFormat	可选的。数字值，表示日期/时间所使用的格式。如果忽略该值，则使用 vbGeneralDate

设置值

NamedFormat 参数的设置值如下：

常数	值	描述
vbGeneralDate	0	显示日期和/或时间。如果有日期部分，则用短日期格式显示。如果有时间部分，则用长时间格式显示。如果都有，两部分都显示
vbLongDate	1	用计算机区域设置值中指定的长日期格式显示日期
vbShortDate	2	用计算机区域设置值中指定的短日期格式显示日期
vbLongTime	3	用计算机区域设置值中指定的时间格式显示时
vbShortTime	4	用 24 小时格式（hh：mm）显示时间

请参阅

FormatCurrency 函数，FormatNumber 函数，FormatPercent 函数

FormatNumber 函数

返回一个数字格式的表达式。

语法

FormatNumber(*Expression*[,*NumDigitsAfterDecimal*[,*IncludeLeadingDigit*[,*UseParensForNegativeNumbers*[,*GroupDigits*]]]])

FormatNumber 函数语法有如下几部分：

部分	描述
Expression	必需的。要被格式化的表达式
NumDigitsAfterDecimal	可选的。数字值，表示小数点右边的显示位数。缺省值为 1，表示使用计算机的区域设置值
IncludeLeadingDigit	可选的。三态常数，表示小数点前是否显示零。关于其值，请参阅“设置值”部分
UseParensForNegativeNumbers	可选的。三态常数，表示是否把负数值放在圆括号内。关于其值，请参阅“设置值”部分
GroupDigits	可选的。的三态常数，表示是否用组分隔符对数字分组，组分隔符在计算机的区域设置值中指定。关于其值，请参阅“设置值”部分

设置值

IncludeLeadingDigit、*UseParensForNegativeNumbers* 和 *GroupDigits* 参数的

设置值如下：

常数	值	描述
TristateTrue	–1	True
TristateFalse	0	False
TristateUseDefault	–2	用计算机区域设置值中的设置值

说明

当忽略一个或多个选项参数时，被忽略的参数值由计算机的区域设置值提供。

注意所有设置值信息都来自“区域设置”的“数字”选项卡。

请参阅

FomratCurrency 函数，FormatDateTime 函数，FormatPercent 函数

FormatPercent 函数

返回一个百分比格式（乘以 100）的表达式，后面有%符号。

语法

FormatPercent(*Expression*[,*NumDigitsAfterDecimal*[,*IncludeLeading Digit*[,*UseParensForNegativeNumbers*[,***GroupDigits***]]]])

FormatPercent 函数语法有如下几部分：

部分	描述
Expression	必需的。要格式化的表达式
NumDigitsAfterDecimal	可选的。表示小数点右边的显示位数。缺省值为 1，表示使用计算机的区域设置值
IncludeLeadingDigit	可选的。三态常数，表示小数点前是否显示零。关于其值，请参阅“设置值”部分
UseParensForNegativeNumbers	可选的。三态常数，表示是否把负数放在圆括号内。关于其值，请参阅“设置值”部分
GroupDigits	可选的。三态常数，表示是否用组分隔符对数字进行分组，组分隔符在计算机的区域设置值中指定。关于其值，请参阅“设置值”部分

设置值

IncludeLeadingDigit、*UseParensForNegativeNumbers* 和 *GroupDigits* 参数的设置值如下：

常数	值	描述
TristateTrue	–1	True
TristateFalse	0	False
TristateUseDefault	–2	使用计算机区域设置值中的设置值

说明

当忽略一个或多个选项参数时，被忽略的参数值由计算机的区域设置值提供。

注意所有的设置值信息都来自“区域设置”的“数字”选项卡。

请参阅

FormatCurrency 函数，FormatDateTime 函数，FormatNumber 函数

FormatType 属性

返回或设置导出一个报表时所使用的格式的类型。

应用于

ExportFormat 对象

语法

object.**FormatType**[=*integer*]

FormatType 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
integer	可选的。一个数值表达式，指定了格式类型，如在设置值中所示

设置值

对 *integer* 的设置如下：

常数	值	描述
rptFmtHTML	0	HTML
rptFmtText	1	文本
rptFmtUnicodeText	2	Unicode
rptFmtUnicodeHTML_UTF8	3	在通用字符集（UTF-8）中编码的 HTML

请参阅

ExportReport 方法，Add 方法（ExportFormats 集合）

示例

```
PrivateSubExportDailyReport()
DataReportl.Title="DailyReport"'Thistitleappearsinthereport.
DimstrTemplateAsString
,Createthetemplate.
```

```
StrTemplate=_
"<HTML>"&vbCrlf&_
"<HEAD>"&vbCrlf&_
"<TITLE>"&"MyCompany:"&rptTagTitle&_
"</TITLE>"&vbCrlf&_
"<BODY>"&vbCrlf&_
rptTagBody&vbCrlf&_
"<BODY>"&vbCrlf&_
"</HTML>"
'AddanewExortFormatobjectsuingthetemplate.
DataReportl.ExportFormats.Add_
Key :="DailyReport",_
FormatType:=rptFmtHTML,_
FileFormatString:="DailyReport(*.htm)",_
FileFilter:="*.HTM",_
Template:strTemplate
, ExportthereportusingthenewExprotFormatobject.
DataReportl.ExportReport_
FormatIndexOrKey:="DailyReport",_
FileName:="c:\Temp\DailyRpt",_
Overwrite:=True,_
ShowDialog:=False,_
```

```
Range:=rptRangeFromTo,_
Pagefrom:=1,_
Pageto:=10
EndSub
```

Forms 属性

返回一个 Forms 集合，其元素代表应用程序中的每个已加载窗体。这个集合包括应用程序的 MDI 窗体、MDI 子窗体以及非 MDI 窗体。Forms 集合有单个属性，即 Count 属性，该属性指定了该集合中元素的数目。

应用于

Global 对象

语法

Forms(*index*)

index 所在处代表一个整数，其范围从 0 到 Forms.Count-1。

说明

可用 Forms 集合遍历应用程序中所有已加载的窗体。它识别名为 Forms 的固定全局变量。可将 Forms(*index*)传递给一个函数，其参数被指定为 *Forms* 类。

请参阅

Global 对象，Form 对象，Forms 集合

ForwardFocus 属性

返回或设置一个值，该值决定按下控件的某个访问键时，哪个控件接收焦点。在创建控件时，ForwardFocus 属性可读/写，在控件运行时，该属性不可用。

应用于

UserControl 对象

设置值

ForwardFocus 的设置值为：

设置值	描述
true	当按下控件的某个访问键时，tab 键次序中的下一个控件接收焦点
false	（缺省）如果 CanGetFocus 属性为真，则当按下控件的某个访问键时，控件本身接收焦点

说明

ForwardFocus 属性使控件能够实现带有访问键的 Lable 控件的行为。

访问键用 AccessKeys 属性设置。当同时按下 ALT 键和某个访问键时，产生控件的 AccessKeyPress 事件。

请参阅

Accesskeys 属性，CanGetFocus 属性

Frame 控件

Frame 控件为控件提供可标识的分组。Frame 可以在功能上进一步分割一个窗体一例如，把 OptionButton 控件分成几组。

语法

Frame

说明

为了将控件分组，首先需要绘制 Frame 控件，然后绘制 Frame 里面的控件。这样就可以把框架和里面的控件同时移动。如果在 Frame 外部绘制了一个控件并试图把它移到框架内部，那么控件将在 Frame 的上部，这时需分别移动 Frame 和控件。

为了在 Frame 中选择多个控件，在使用鼠标在控件周围绘制框时，按住 CTRL 键。

属性

OLEDropMode 属性，VackColor,ForeColor 属性，BorderStyle 属性，FontBold,FontItalic,FontStrikethru,FontUnderline 属性，FontName 属性，FontSize 属性，Height,Width 属性，Left,Top 属性，TabIndex 属性，Tag 属性，Visible 属性，Clip 属性，DragIcon 属性，Appearance 属性，Caption 属性，Enabled 属性，HelpContextID 属性，Index 属性（ControlArray），Name 属性，Parent 属性，Font 属性，Container 属性，ToolTipText 属性，WhatsThisHelpID 属性，Frame

方法

Refresh 方法，Drag 方法，Move 方法，Zorder 方法，OLEDrag 方法，ShowWhatsThis 方法

事件

Click 事件，DragDrop 事件，DragOver 事件，MouseDown，MouseUp 事件，MouseMove 事件，DblClick 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件

请参阅

OptionButton 控件

FreeFile 函数

返回一个 Integer，代表下一个可供 Open 语句使用的文件号。

语法

FreeFile[(*rangenumber*)]

可选的参数 *rangenumber* 是一个 Variant，它指定一个范围，以便返回该范围之内的下一个可用文件号。指定 0（缺省值）则返回一个介于 1–255 之间的文件号。指定 1 则返回一个介于 256–511 之间的文件号。

说明

使用 FreeFile 提供一个尚未使用的文件号。

请参阅

Open 语句

示例

本示例使用 FreeFile 函数来返回下一个可用的文件号。在循环中，共打开五个输出文件，并在每个文件中写入一些数据。

```
DimMyIndex,FileNumber
ForMyIndex=1To5    '循环五次。
   FileNumber=FreeFile '取得未使用的文件号。
   Open"TEST"&MyIndexForOutputAs#FileNumber '创建文件名。
   Write#FileNumber,"Thisisasample."    '输出文本至文件中。
   Close#FileNumber'关闭文件。
NextMyIndex
```

FreeSpace 属性

返回指定驱动器上或网络共享的用户可用磁盘剩余空间容量。只读。

应用于

Drive 对象

语法

object.**FreeSpace**

object 总是一个 Drive 对象。

说明

一般来说，FreeSpace 属性的返回值和 AvailableSpace 属性的返回值是相同的。对于支持限额的计算机系统来说，二者之间可能有所

不同。

下面的代码举例说明了 FreeSpace 属性的用法：

```
SubShowFreeSpace(drvPath)
Dimfs,d,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(fs.GetDriveName(drvPath))
s="Drive"&UCase(drvPath)&"-"
s=s&d.VolumeName&vbCrLf
s=s&"FreeSpace:"&FormatNumber(d.FreeSpace/1024,0)
s=s&"Kbytes"
MsgBoxs
EndSub
```

请参阅

AvailableSpace 属性，DriveLetter 属性，DriveType 属性，FileSystem 属性，IsReady 属性，Path 属性（FileSystemObject 对象），RootFolder 属性，SerialNumber 属性，ShareName 属性，TotalSize 属性，VolumeName 属性

Friend

修改窗体模块或者类模块中的一个过程的定义，使得该过程可以被类外的模块调用，但必须是在定义该类的工程的部分内。Friend 过程在标准的模块中是不能使用的。

语法

[Private|Friend|Public][Static][Sub|Function|Property]*procedurename*

必需的 *procedurename* 是所要设定为在整个工程中都是可见的过程的名称，但是该过程对于类的控制器却是不可见的。

说明

类中的 Public 过程在任何地方都可以调用，即使是被该类实例的控制器所调用也可以。声明一个过程为 Private 就阻止了该对象的控制器对于该过程的调用，但是同时也阻止了定义该类的工程内部对该过程的调用。Friend 使得该过程在整个工程中都是可见的，但是对于一个对象实例的控制器却是不可见的。Friend 只能出现在窗体模块和类模块中，并且只能修改过程的名称，不能修改变量或者类型。一个类中的过程可以访问某个工程所有其它类的 Friend 过程。Friend 过程不会出现在类的类型库中。一个 Friend 过程不能被后期绑定。

请参阅

《MicrosoftVisualBasic6.0 部件工具指南》一书的第六章“在对象之间的私有通信”，第二部分的“部件设计的普遍原理”的“生成 ActiveX 部件”，《MicrosoftVisualBasic6.0 程序员指南》一书的第九章的“友好属性和方法”的“对象编程”，Function 语句，PropertyGet 语句，PropertyLet 语句，PropertySet 语句，Sub 语句

示例

当下面的代码被放置在一个类模块中时，该段代码会使得成员变量 dblBalance 对该工程中的类的所有用户来说都是可访问的。类的任何用户都可以获得该变量的值；但是只有工程内的代码可以对该变量赋值。

```
PrivatedblBalanceAsDouble

PublicPropertyGetBalance()AsDouble
Balance=dblBalance
EndProperty

FriendPropertyLetBalance(dblNewBalanceAsDouble)
dblBalance=dblNewBalance
EndProperty
```

FromPage,ToPage 属性

返回或设置“打印”对话框的 From 和 To 文本框的值。

应用于

CommonDialog 控件（“打印”对话框）

语法

object.**FromPage**[=*number*]

object.**ToPage**[=*number*]

FromPage 和 ToPage 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	数值表达式，指定要打印的首、尾页号

说明

这些属性只有当 cdlPDPageNums 标志被置时才有效。

数据类型

Integer

FullName 属性

该属性返回 VisualBasicIDE 的完整路径名（即 vb6.exe 运行的路径）。

应用于

VBE 对象

语法

object.**FullName**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

返回的路径名总是以绝对路径的形式提供（例如"c:\projects\myproject.vbp"），即使它在 VisualBasic 中表示为相对路径（如"..\projects"）。

请参阅

FileName 属性

FullPath 属性

返回一个 String，其内容为被引用的类型库的路径和文件名，此属性为只读。

应用于

Reference 对象

请参阅

AddFromFile 方法（VBA 外接程序对象模型）,AddFromGuid 方法，BuiltIn 属性，GUID 属性，IsBroken 属性

示例

下列示例使用 FullPath 属性返回指定引用的对象库的路径全名称。

```
Debug.PrintApplication.VBE.ActiveVBProject.References(1)
FullPath
```

Function 控件（数据报表设计器）

Function 控件显示运行时使用各种内置函数计算出的数字。

语法

rptFunction

说明

Function 控件包含的函数包括：

函数	描述
Sum	合计一个字段的值
Min	显示一个字段的最小值
Max	显示一个字段的最大值
Average	显示一个字段的平均值
StandardDeviation	显示一列数字的标准偏差
StandardError	显示一列数字的标准错误
ValueCount	显示包含非空值的字段数
RowCount	显示一个报表部分中的行数

Function 控件只能被放置在数据报表设计器的组注脚部分中。DataEnvironment 也有合计字段的功能，有些与 Function 控件相似的功能。

属性

CanGrow 属性，FunctionType 属性，DataFormat 属性，BackColor,ForeColor 属性，BackStyle 属性，BorderColor 属性，BorderStyle 属性，Height,Width 属性，Left,Top 属性，Visible 属性，Algnment 属性，Name 属性，Font 属性，DataField 属性

Function 语句

声明 Function 过程的名称，参数以及构成其主体的代码。

语法

[**Public**|**Private**|**Friend**] [**Static**] **Function** *name* [(*arglist*)] [**As** *type*]

[*statements*]

[*name=expression*]

[**ExitFunction**]

[*statements*]

[*name=expression*]

EndFunction

Function 语句的语法包含下面部分：

部分	描述
public	可选的。表示所有模块的所有其它过程都可访问这个 Function 过程。如果是在包含 OptionPrivate 的模块中使用，则这个过程在该工程外是不可使用的
private	可选的。表示只有包含其声明的模块的其它过程可以访问该 Function 过程
friend	可选的。只能在类模块中使用。表示该 Function 过程在整个工程中都是可见的，但对于对象实例的控制者是不可见的
static	可选的。表示在调用之间将保留 Function 过程的局部变量值。Static 属性对在该 Function 外声明的变量不会产生影响，即使过程中也使用了这些变量
name	必需的。Function 的名称；遵循标准的变量命名约定
arglist	可选的。代表在调用时要传递给 Function 过程的参数变量列表。多个变量应用逗号隔开
type	可选的。Function 过程的返回值的数据类型，可以是 Byte、布尔、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（除定长）、Object、Variant 或任何用户定义类型
statements	可选的。在 Function 过程中执行的任何语句组。
expression	可选的。Function 的返回值。

其中的 *arglist* **参数的语法以及语法各个部分如下：**

[**Optional**][**ByVal**|**ByRef**][**ParamArray**]*varname*[()][**As***type*][=*defaultvalue*]

部分	**描述**
Optional	**可选的。表示参数不是必需的。如果使用了该选项，则** *arglist* **中的后续参数都必须是可选的，而且必须都使用** Optional **关键字声明。如果使用了** ParamArray**，则任何参数都不能使用** Optional **声明**
ByVal	**可选的。表示该参数按值传递**
ByRef	**可选的。表示该参数按地址传递。**ByRef **是** VisualBasic **的缺省选项**
ParamArray	**可选的。只用于** *arglist* **的最后一个参数，指明最后这个参数是一个** Variant **元素的** Optional **数组。使用** ParamArray **关键字可以提供任意数目的参数。**ParamArray **关键字不能与** ByVal**，**ByRef**，或** Optional **一起使用**
varname	**必需的。代表参数的变量的名称；遵循标准的变量命名约定。**
type	**可选的。传递给该过程的参数的数据类型；可以是** Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal**（目前尚不支持）**、Date、String**（只支持变长）**、Object **或** Variant**。如果参数不是** Optional**，则也可以是用户定义类型，或对象类型**

defaultvalue 可选的。任何常数或常数表达式。只对于 Optional 参数时是合法的。如果类型为 Object，则显式缺省值只能是 Nothing

说明

如果没有使用 Public、Private 或 Friend 显式指定，则 Function 过程缺省为公用。如果没有使用 Static，则局部变量的值在调用之后不会保留。Friend 关键字只能在类模块中使用。但 Friend 过程可以被工程的任何模块中的过程访问。Friend 过程不会在其父类的类型库中出现，且 Friend 过程不能被后期绑定。

警告 Function 过程可以是递归的；也就是说，该过程可以调用自己来完成某个特定的任务。不过，递归可能会导致堆栈上溢。通常 Static 关键字和递归的 Function 过程不在一起使用。

所有的可执行代码都必须属于某个过程。不能在另外的 Function、Sub 或 Property 过程中定义 Function 过程。

ExitFunction 语句使执行立即从一个 Function 过程中退出。程序接着从调用该 Function 过程的语句之后的语句执行。在 Function 过程的任何位置都可以有 ExitFunction 语句。

Function 过程与 Sub 过程的相似之处是：Function 过程是一个可以获取参数，执行一系列语句，以及改变其参数值的独立过程，而与子过程不同的是：当要使用该函数的返回值时，可以在表达式的右边使用 Function 过程，这与内部函数，诸如 Sqr、Cos 或 Chr 的使用方式一样。

在表达式中，可以通过使用函数名，并在其后用圆括号给出相应

的参数列表来调用一个 Function 过程。请参阅 Call 语句关于如何调用 Function 过程的详细说明。

要从函数返回一个值，只需将该值赋给函数名。在过程的任意位置都可以出现这种赋值。如果没有对 *name* 赋值，则过程将返回一个缺省值：数值函数返回 0，字符串函数返回一个零长度字符串(""),Variant 函数则返回 Empty。如果在返回对象引用的 Function 过程中没有将对象引用赋给 name（使用 Set），则函数返回 Nothing。

下面的示例说明如何给一个名为 BinarySearch 的函数赋返回值。在这个示例中，将 False 赋给了该函数名，表示没有找到某个值。

```
FunctionBinarySearch(...)AsBoolean
  …
   '值未找到，返回一个 False 值。
   Iflower>upperThen
        BinarySearch=False
        ExitFunction
   EndIf
…
EndFunction
```

在 Function 过程中使用的变量分为两类：一类是在过程内明确声明的，另一类则不是。在过程内明确声明的变量（使用 Dim 或等效方法）都是局部变量。对于那些没有在过程中明确声明的变量，

除非它们在该过程外更高级别的位置有明确地声明，否则也是局部的。

警告过程可以使用没有在过程内明确声明的变量，但只要有任何在模块级别中定义的名称与之相同，就会产生名称冲突。如果过程中使用的未声明的变量与另一个过程，常数，或变量的名称相同，则会认为过程使用的是模块级的名称。明确声明变量就可以避免这类冲突。可以使用 OptionExplicit 语句来强制明确声明变量。

警告 VisualBasic 可能会重新安排数学表达式以提高内部效率。若 Function 过程会改变某个数学表达式中变量的值，则应避免在此表达式中使用该函数。

请参阅

Call 语句，Dim 语句，OptionExplicit 语句，PropertyGet 语句，PropertyLet 语句，PropertySet 语句，Sub 语句，AddressOf 语句，Friend

示例

该示例使用 Function 语句来声明 Function 过程的名称、参数、以及构成 Function 过程主体的代码。最后一个例子中使用了确定类型的、初始化的 Optional 参数。

```
'下面的用户自定义函数返回
'它的参数的平方根。
FunctionCalculateSquareRoot(NumberArgAsDouble)AsDouble
    IfNumberArg<0Then     '评估参数。
        ExitFunction '退出调用过程。
```

```
    Else
        CalculateSquareRoot=Sqr(NumberArg)    '返回平方根。
    EndIf
EndFunction
```

使用 ParamArray 关键字可以使函数接收数目可变的参数。在下面的定义中，FirstArg 是按值传递的。

```
FunctionCalcSum(ByValFirstArgAsInteger,ParamArrayOtherArgs())
DimReturnValue
'如果用如下代码调用该函数：
ReturnValue=CalcSum(4,3,2,1)
'则局部变量被赋予以下值：FirstArg=4，
'OtherArgs(1)=3，OtherArgs(2)=2，等等。
'假设缺省数组下界=1。
```

Optional 参数可以带缺省值，可以是除 Variant 之外的任何类型。

```
'如果函数的参数定义如下：
FunctionMyFunc(MyStrAsString,OptionalMyArg1As_Integer=5,Optional
MyArg2="Dolly")
DimRetVal
'则可用如下代码调用该函数：
RetVal=MyFunc("Hello",2,"World")    '提供了所有 3 个参数。
```

```
RetVal=MyFunc("Test",,5)  '省略了参数 2。
'参数 1 和参数 3 使用了命名的参数。
RetVal=MyFunc(MyStr:="Hello",MyArg1:=7)
```

FunctionType 属性

返回或设置用于计算由 Function 控件显示的值的函数。

应用于

Function 控件（数据报表生成器）

语法

object.**FunctionType**[=*integer*]

FunctionType 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
integer	可选的。一个指定函数的数值表达式，如在设置值中所示

设置值

对 integer 的设置如下：

常数	值	描述
rptFuncSum	0	（缺省的）合计数据字段的值
rptFuncAve	1	计算数据字段的平均值
rptFuncMin	2	返回数据字段的最小值
rptFuncMax	3	返回数据字段的最大值
rptFuncRCnt	4	计数部分中的行数
rptFuncVCnt	5	计数包含非空值的字段
rptFuncSDEV	6	计算标准偏差
rptFuncSERR	7	计算标准错误

FV 函数

返回一个 Double，指定未来的定期定额支付且利率固定的年金。

语法

FV(*rate*,*nper*,*pmt*[,*pv*[,*type*]])

FV 函数有下列命名参数：

部分	描述
rate	必需的。Double，指定每一期的利率。例如，如果有一笔贷款年百分率(APR)为百分之十且按月付款的汽车贷款，则利率为 0.1/12 或 0.0083
nper	必需的。Integer，指定一笔年金的付款总期限。例如，如果对一笔为期四年的汽车贷款选择按月付款方式，则贷款期限共有 4*12（或 48）个付款期
pmt	必需的。Double 指定每一期的付款金额。付款金额通常包含本金和利息，而且此付款金额在年金的有效期间是不会改变的
pv	可选的。Variant，指定未来一系列付款（或一次付清款项）的现值。例如，当借钱买一辆汽车时时，向贷方所借的金额为未来每月付款给贷方的现值。如果省略的话，缺省值为 0
type	可选的。Variant，指定贷款到期时间。如果贷款在贷款周期结束时到期，请使用 0。如果贷款在周期开始时到期，请使用 1。如果省略的话，缺省值为 0

说明

年金是一段时间内一系列固定现金支付。年金可以是贷款（如房屋抵押贷款），也可以是一笔投资（如按月储蓄计划）。

在支付期间，必须用相同的单位来计算 *rate* 和 *nper* 参数。例如，如果 *rate* 用月份来计算，则 *nper* 也必须用月份来计算。

对所有参数，用负数表示现金支出（如储蓄存款），而用正数表

示现金收入（如红利支票）。

请参阅

DDB 函数，Ipmt 函数，IRR 函数，MIRR 函数，Nper 函数，NPV 函数，Pmt 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

本示例使用 FV 函数计算某项投资的未来价值，计算时给定每期的利率（APR/12），投资的总期数（TotPmts），每期的投资额（Payment），已投资的现额（PVal）及付款方式（以数值表示期初或期末付款（PayType））。请注意，因为 Payment 代表支付出去的现金，故其为负数。

```
DimFmt,Payment,APR,TotPmts,PayType,PVal,FVal
ConstENDPERIOD=0,BEGINPERIOD=1 '付款方式。
Fmt="###,###,##0.00"  '定义金额格式。
Payment=InputBox("Howmuchdoyouplantosaveeachmonth?")
APR=InputBox("Entertheexpectedinterest 年利率.")
IfAPR>1ThenAPR=APR/100 '确保格式正确。
TotPmts=InputBox("Forhowmanymonthsdoyouexpecttosave?")
PayType=MsgBox("Doyoumakepaymentsattheendofmonth?",vbYesNo)
IfPayType=vbNoThenPayType=BEGINPERIODElsePayType=ENDPERIOD
PVal=InputBox("Howmuchisinthissavingsaccountnow?")
FVal=FV(APR/12,TotPmts,-Payment,-PVal,PayType)
MsgBox"Yoursavingswillbeworth"&Format(FVal,Fmt)&"."
```

Get 语句

将一个已打开的磁盘文件读入一个变量之中。

语法

Get [#]*filenumber*,[*recnumber*],*varname*

Get 语句的语法具有以下几个部分：

部分	描述
filenumber	必需的。任何有效的文件号
recnumber	可选的。Variant(Long)。记录号（Random 方式的文件）或字节数（Binary 方式的文件），以表示在此处开始读出数据
varname	必需的。一个有效的变量名，将读出的数据放入其中

说明

通常用 Put 将 Get 读出的数据写入一个文件。

文件中第一个记录或字节位于位置 1，第二个记录或字节位于位置 2，依此类推。若省略 recnumber，则会读出紧随上一个 Get 或 Put 语句之后的下一个记录或字节（或读出最后一个 Seek 函数指出的记录或字节）。所有

用于分界的逗号都必须罗列出来，例如：

Get 语句

将一个已打开的磁盘文件读入一个变量之中。

语法

Get [#]*filenumber*,[*recnumber*],*varname*

Get 语句的语法具有以下几个部分：

部分	描述
filenumber	必需的。任何有效的文件号
recnumber	可选的。Variant(Long)。记录号（Random 方式的文件）或字节数（Binary 方式的文件），以表示在此处开始读出数据
varname	必需的。一个有效的变量名，将读出的数据放入其中

说明

通常用 Put 将 Get 读出的数据写入一个文件。

文件中第一个记录或字节位于位置 1，第二个记录或字节位于位置 2，依此类推。若省略 recnumber，则会读出紧随上一个 Get 或 Put 语句之后的下一个记录或字节（或读出最后一个 Seek 函数指出的记录或字节）。所有用于分界的逗号都必须罗列出来，例如：

```
Get#4,,FileBuffer
```

下列规则适用于以 Random 方式打开的文件：

如果要读出的数据的长度小于 Open 语句的 Len 子句中所指定的长度，Get 会在某个边界之内读出随后的记录，在这里，边界的长度等于记录的长度。将此文件缓冲区内的现有内容填入到一个空间之内，该空间介于一个记录的结尾与下一个记录的开头之间。因为无法确定填入的数据量，所以，应设法使记录的长度与读出的数据长度一致，这通常是个好办法。

如果读出的变量是一个可变长度字符串，则 Get 语句先读出一个含有字符串长度的双字节描述符，然后读出放入变量中的数据。所以，Open 语句中的 Len 子句所指定的记录长度必须至少比字符串的实际长度多两个字节。

如果读出的变量是一个数值类型的 Variant，则 Get 先读出两个字节，识别 Variant 的 VarType，然后读出放入此变量中的数据。例如，在读出 VarType3 的 Variant 时，Get 读出六个字节：前两个字节说明 Variant 为 VarType3(Long)，后四个字节则包含 LongdefLong@veendf98.chm 类型数据。Open 语句中的 Len 子句所指定的记录长度必须至少比储存变量所需的实际长度多两个字节。

注意可以使用 Get 语句从磁盘中读出一个 Variant 数组，但不能使用它读出包含数组的标量 Variant。也不能使用 Get 从磁盘中读出对象。

如果读出的变量是 VarType8(String)的 Variant，则 Get 先读出两个字节，识别 VarType，接下来的两个字节指出字符串的长度，

然后读出字符串数据。Open 语句中的 Len 子句所指定的记录长度必须比实际字符串的长度至少多四个字节。
如果读出的变量是一个动态数组，则 Get 会读出一个描述符，其长度等于 2 加上 8 乘以维数，即 2+8*NumberOfDimensions。读出数组数据和数组描述符就需要占据字节，而 Open 语句中的 Len 子句所指定的记录长度必须大于或等於这些字节数的总和。例如，在将数组写入磁盘时，下列数组声明需要 118 个字节：

```
DimMyArray(1To5,1To10)AsInteger
```

这 118 个字节的分配情况如下：18 个字节用于描述符(2+8*2)，100 个字节用于数据(5*10*2)。
如果读出的变量是一个大小固定的数组，则 Get 只读出数据。它不读出描述符。
如果读出的变量是任何其他类型的变量（不是可变长度的字符串或 Variant），则 Get 只读出变量数据。Open 语句中的 Len 子句所指定的记录长度必须大于或等於要读出的数据的长度。
Get 在读出用户定义类型的元素时，好象是单独地读出每一个元素，只是不在元素之间进行填充。在磁盘上，（用 Put 写入的）用户定义的类型的动态数组之前有一个描述符，其长度等于 2 加上 8 乘以维数，即 2+8*NumberOfDimensions。Open 语句中的 Len 子句所指定的记录长度必须大于或等於读出个别元素（包括任何数组及其描述符在内）所需的全部字节数总和。
对于以 Binary 方式打开的文件，Random 的所有规则都适用，但以

下情况除外：

当 Open 语句中的 Len 子句不起作用时，Get 连续从磁盘中读出所有变量；也就是说，两个记录之间没有任何填充。

对于任何不属于用户定义的类型的数组，Get 只读出数据。它不会读出描述符。

Get 读出可变长度字符串，不管这些字符串是否具有长度为 2 字节的描述符，它们都不是用户定义的类型的元素。所读出的字节数等於字符串已包含的字符数。例如，下列语句从文件号为 1 的文件中读出十个字节：

```
VarString=String(10,)
Get#1,,VarString
```

请参阅

Type 语句，Open 语句，PutSeek 函数，VarType 函数。

示例

本示例使用 Get 语句来将数据从文件读到变量中。示例中假设 TESTFILE 文件中含有五个用户自定义类型的记录。

```
TypeRecord '定义用户自定义的数据类型。
   IDAsInteger
   NameAsString*20
EndType

DimMyRecordAsRecord,Position   '声明变量。
```

```
Open"TESTFILE"ForRandomAs#1Len=Len(MyRecord)
'使用 Get 语句来读样本文件。
Position=3                    '定义记录号。
Get#1,Position,MyRecord       '读第三个记录。
Close#1    '关闭文件。
```

GetAbsolutePathName 方法

从提供的路径说明中返回一个完整、明确的路径。

应用于

FileSystemObject 对象。

语法

object.**GetAbsolutePathName**(*pathspec*)

GetAbsolutePathName 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
pathspec	必需的。要改变到一个完整、明确路径的路径说明

说明

一个路径如果提供了从指定驱动器根目录的一个完整引用，则该路径是完整、明确的。一个完整的路径如果指定一个被映射驱动器的根文件夹，它只能以路径分隔符（\）为结尾。

假设当前目录是 c:\mydocuments\reports，下面的表说明了 GetAbsolutePathName 方法的行为。

Pathspec	返回的路径
"c:"	"c:\mydocuments\reports"
"c:.."	"c:\mydocuments"
"c:\\\"	"c:\"
"c:*.*\may97"	"c:\mydocuments\reports*.*\may97"
"region1"	"c:\mydocuments\reports\region1"
"c:\..\..\mydocuments"	"c:\mydocuments"

请参阅

BuildPath 方法,GetBaseName 方法,GetDrive 方法,GetDriveName 方法,GetExtensionName 方法,GetFile 方法,GetFileName 方法,GetFolder 方法,GetParentFolderName 方法,GetSpecialFolDer 方法,GetTempName

GetAllSettings 函数

从 Windows 注册表或(Macintosh 中)应用程序初始化文件中的信息中返回应用程序项目的所有注册表项设置及其相应值（开始是由 SaveSetting 产生)。

语法

GetAllSettings(*appname,section*)

GetAllSettings 函数的语法具有下列命名参数：

部分	描述
appname	必需的。字符串表达式，包含应用程序或工程的名称，并要求这些应用程序或工程有注册表项设置在 Macintosh 中，这是 System 文件夹中 Preferences 文件夹中初始化文件的文件名
section	必需的。字符串表达式，包含区域名称，并要求该区域有注册表项设置。GetAllSettings 返回 Variant，其内容为字符串的二维数组，该二维数组包含指定区域中的所有注册表项设置及其对应值

说明

如果 appname 或 section 不存在，则 GetAllSettings 返回未初始化的 Variant。

请参阅

DeleteSetting 语句,GetSetting 函数,SaveSetting 语句

示例

本示例首先使用 SaveSetting 语句来建立 Windows 注册表（或 16 位 Windows 平台的.ini 文档）里 *appname* 应用程序的项目，然后再使用 GetAllSettings 函数来取得设置值并显示出来。请注意，应用程序名和 *section* 名称不能用 GetAllSettings 函数取得。最后，使用 DeleteSetting 语句将该应用程序项删除。

```
'用来保存 GetAllSettings 函数所返回之二维数组数据的变量
```

```
'整型数是用来计数用。
DimMySettingsAsVariant,intSettingsAsInteger
'在注册区中添加设置值。
SaveSettingappname:="MyApp",section:="Startup",_
key:="Top",setting:=75
SaveSetting"MyApp","Startup","Left",50
'取得输入项的设置值。
MySettings=GetAllSettings(appname:="MyApp",section:="Startup")
   ForintSettings=LBound(MySettings,1)ToUBound(MySettings,1)

   Debug.PrintMySettings(intSettings,0),MySettings(intSettings,1
)
   NextintSettings
DeleteSetting"MyApp","Startup"
```

GetAttr 函数

返回一个 Integer，此为一个文件、目录、或文件夹的属性。

语法

GetAttr(*pathname*)

必需的 *pathname* 参数是用来指定一个文件名的字符串表达式。*pathname* 可以包含目录或文件夹、以及驱动器。

返回值

由 GetAttr 返回的值，是下面这些属性值的总和：

常数	值	描述
vbNormal	0	常规
vbReadOnly	1	只读
vbHidden	2	隐藏
vbSystem	4	系统文件
vbDirectory	16	目录或文件夹
vbArchive	32	上次备份以后，文件已经改变

注意这些常数是由 VBA 指定的，在程序代码中的任何位置，可以使用这些常数来替换真正的值。

说明

若要判断是否设置了某个属性，在 GetAttr 函数与想要得知的属性值之间使用 And 运算符与逐位比较。如果所得的结果不为零，则表示设置了这个属性值。例如，在下面的 And 表达式中，如果档案属性没有设置，则返回值为零：

```
Result=GetAttr(FName)AndvbArchive
```

如果文件的档案属性已设置，则返回非零的数值。

请参阅

SetAttr 语句，FileAttr 函数，And 运算符

示例

本示例使用 GetAttr 函数来得知文件及目录或文件夹的属性。只有常量 vbNormal、vbReadOnly、vbHidden 和 vbAlias 是可用的。

```
DimMyAttr
```

```
'假设 TESTFILE 具有隐含属性。
MyAttr=GetAttr("TESTFILE") '返回 2。

'如果 TESTFILE 有隐含属性，则返回非零值。
Debug.PrintMyAttrAndvbHidden

'假设 TESTFILE 具有隐含的只读属性。
MyAttr=GetAttr("TESTFILE") '返回 3。

'如果 TESTFILE 含有隐含属性，则返回非零值。
Debug.PrintMyAttrAnd(vbHidden+vbReadOnly)

'假设 MYDIR 代表一目录或文件夹。
MyAttr=GetAttr("MYDIR")     '返回 16。
```

GetAutoServerSettings 函数

返回关于 ActiveX 部件的注册状态的信息。

语法

object.**GetAutoServerSettings**([*progid*],[*clsid*])

GetAutoServerSettings **函数语法有这些部分：**

部分	描述
object	必需的。对象表达式，其值为“应用于”列表中的对象
progid	可选的。Variant 表达式，为该部件指定 ProgID
clsid	可选的。Variant 表达式，为该部件指定 CLSID

返回值

GetAutoServerSettings 函数返回含有关于给定的 ActiveX 部件的值的数组的 Variant 变量。其索引值和描述如下：

值	描述
1	若 ActiveX 部件被远程注册则为 True
2	远程机器名。
3	RPC 网络协议名
4	RPC 身份验证级

说明

如果值丢失或无效，则该值为空字符串。如果在使用方法期间出错，则返回值为 Empty 类型的 Variant 变量。

示例

这个示例检索关于远程注册对象"Hello"的信息：

```
SubViewHello()
   DimoRegClassAsNewRegClass
   DimvRCAsVariant
   vRC=oRegClass.GetAutoServerSettings_
```

```
("HelloProj.HelloClass")
   IfNot(IsEmpty(vRC))Then
      IfvRC(1)Then
         MsgBox"Helloisregisteredremotelyona"_
         &"servernamed:"&vRC(1)
      Else
         MsgBox"Helloisregisteredlocally."
      EndIf
   Endif
EndSub
```

GetBaseName 方法

返回一个包含路径中最后部件的基本名字（去掉任何文件扩展名）的字符串。

应用于

FilesSystemObiect 对象

语法

object.**GetBaseName**(*path*)

GetBaseName 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
path	必需的。要返回其基本名字的部件的路径说明

说明

如果没有部件和 path 参数匹配，GetBaseName 方法返回一个长度为零的字符串（""）。

注意 GetBaseName 方法只对提供的 path 字符串起作用。它既不试图去辨认路径，也不检查指定路径是否存在。

请参阅

BuildPATH 方法，GetAbsolutePathName 方法，GetDrive 方法，GetDriveName 方法，GetExtensionName 方法，GetFile 方法，GetFileName 方法，GetFolder 方法，GetParentFolderName 方法，GetSpecialFolder 方法，GetTempName 方法

GetData 方法

用于从 Clipboard 对象返回一个图形。不支持命名参数。

应用于

Clipboard 对象

语法

object.**GetData**(*format*)

GetData 方法的语法包含下列部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
format	可选的。一个常数或数值，如“设置值”中所描述的，它指定 Clipboard 图形的格式。必须用括号将该常数或数值括起来。如果 format 为 0 或省略，GetData 自动使用适当的格式

设置值

format 的设置值有：

常数	值	描述
vbCFBitmap	2	位图（.bmp 文件）
vbCFMetafile	3	元文件（.wmf 文件)
vbCFDIB	8	设备无关位图(DIB)
vbCFPalette	9	调色板

说明

上述常数在 VisualBasic(VB)对象浏览器中的对象库里列出。

如果 Clipboard 对象没有与期望的格式相匹配的图形，则返回空。

如果 Clipboard 对象中只有一个调色版，则创建最小尺寸(1x1)的 DIB。

请参阅

GetFormat 方法，GetText 方法，SetDate 方法，SetText 方法

示例

本示例使用 GetData 方法从 Clipboard 对象中将一个位图复制到一

个窗体上。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
    ConstCF_BITMAP=2                    '定义位图各种格式。
    DimMsg                              '声明变量。
    OnErrorResumeNext                   '设置错误处理。
    Msg="ChooseOKtoloadabitmapontotheClipboard."
    MsgBoxMsg                           '显示信息。
    Clipboard.Clear                     '清除剪贴板。
    Clipboard.SetDataLoadPicture("PAPER.BMP")    '取得位图。
    IfErrThen
        Msg="Can'tfindthe.bmpfile."
        MsgBoxMsg                       '显示错误信息。
        ExitSub
    EndIf
    Msg="AbitmapisnowontheClipboard.ChooseOKtocopy"
    Msg=Msg&"thebitmapfromtheClipboardtotheform"
    MsgBoxMsg                           '显示信息。
    Picture=Clipboard.GetData()         '从剪贴板上复制。
    Msg="ChooseOKtoclear theform."
    MsgBoxMsg                           '显示信息。
    Picture=LoadPicture()               '清除窗体。
EndSub
```

GetData 方法（DataObject 对象）

以变体型格式从 DataObject 对象返回数据。

应用于

DataObject 对象

语法

object.**GetData**(*format*)

GetData 方法语法包含下面部分：

部分	描述
object	必需的。对象表达式，其值是“应用于”列表中的一个对象
format	确定数据格式的常数或值，如“设置值”中所述。它们必须在括号之内。如果 *format* 是 0 或被省略，GetData 自动采用合适的格式

设置值

format 设置如下：

常数	值	描述
VbCFText	1	文本（.txt 文件）
vbCFBitmap	2	位图（.bmp 文件）
vbCFMetafile	3	元文件（.wmf 文件）
vbCFEMetafile	14	增强元文件（.emf 文件）
vbCFDIB	8	与设备无关的位图(DIB)
vbCFPalette	9	调色板
vbCFFiles	15	文件列表
VbCFRTF	-16639	丰富文本格式（.rtf 文件）

说明

这些常数列于对象浏览器的 VisualBasic(VB)对象库。

对于 GetData 和 SetData 方法，可以使用列于设置值以外的数据格式。包括通过 RegisterClipboardFormat() API 函数注册的自定义格式。

然而，有几点要注意的：

当 SetData 方法不能识别指定的数据格式时，它要求数据的格式为字节数组。

尽管 VisualBasic 能够透明地将返回的字节数组转化成其它数据类型，如字符串，但是，当 GetData 方法不能识别数据的格式时，它总是返回字节数组格式的数据。

在某些操作系统运行时，GetData 返回的字节数组将大于实际数据，并在数组末尾带有随机的字节。原因是 VisualBasic 不知道

数据的格式，仅知道操作系统分配给该数据的内存数量。所分配的内存经常要大于数据实际所需要的，所以在分配的内存段的末尾附近有附加的字节。因此要以有意义的方式，采用合适的函数来解释返回的数据（例如，数据是文本格式时，可用 Left 函数以特定的长度来截取字符串）。

注意不是所有的应用程序都支持 vbcfBitmap 或 vbCFPalette，因此建议如果可能要尽量使用 vbCFDIB。

GetDataMember 事件

当一个数据使用者要求一个新的数据源时发生。

应用于

Class

语法

Private Sub *object*_**GetDataMember**(*DataMember* **As String**, *Data As Object*)

GetDataMember 事件语法有这些部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
dataMember	一个字符串，它包含作为数据源被绑定的数据成员的名称
data	一个对象引用一个 ADORecordSet 对象或实现 OLEDBSimpleProvider 界面的一个类

说明

只有当一个对象的 DataSourceBehavior 属性被设置为 vbDataSource 或 vbOLEDBProvider (仅对于类)时，GetDataMember 事件才是可用的。您可以添加代码到 GetDataMember 事件过程来初始化一个数据成员，或从一个对象中的多个数据成员中选择。

请参阅

DataMember 属性

示例

这个示例使用 GetDataMember 事件来确定一个数据源类将提供什么数据。

```
OptionExplicit
PrivatersFirstAsADODB.Recordset
PrivatersSecondAsADODB.Recordset
PrivatersDefaultAsADODB.Recordset
```

```
PrivateSubClass_GetDataMember(DataMemberAsString,DataAsObject)
    SelectCaseDataMember
        Case"First"
            SetData=rsFirst
        Case"Second"
            SetData=rsSecond
        Case“”‘default
            SetData=rsDefault
        CaseElse
            Err.Raise99999,"DataSource","InvalidDataMember"
    EndSelect
EndSub
```

GetDrive 方法

返回一个与指定路径中的驱动器相对应的 Drive 对象。

应用于

FileSystemObject 对象

语法

object.**GetDrive***drivespec*

GetDrive 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
drivespec	必需的。*drivespec* 参数可以是一个驱动器字符（c）、一个驱动器字符加一个冒号（c:）、一个驱动器字符加冒号和路径分隔符（c:\）或任何网络共享的说明（\\computer2\share1）

说明

对于网络共享，要进行检查以确保共享存在。

如果 drivespec 不符合任何一种可以接受的形式或者不存在，则发生一个错误。

对一个普通路径字符串调用 GetDrive 方法，使用下面步骤得到一个适合作为 drivespec 使用的字符串：

```
DriveSpec=GetDriveName(GetAbsolutePathName(Path))
```

请参阅

DriveExists 方法，GetAbsolutePathName 方法，GetBaseName 方法，GetDriveName 方法，GetExetnsionName 方法，GetFile 方法，GetFileName 方法，GetFolder 方法，GetParentFolderName 方法，GetSpecialFolder 方法，GetTempName 方法，Drive 对象

GetDriveName 方法

返回一个包含指定路径的驱动器名字的字符串。

应用于

FileSystemObject 对象

语法

object.**GetDriveName**(*path*)

GetDriveName 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
path	必需的。要返回其驱动器名字的部件的路径说明

说明

如果驱动器不能确定，GetDriveName 方法返回一个长度为零的字符串（""）。

注意 GetDriveName 方法只对提供的路径字符串起作用。它没有尝试去辨认路径，也不对指定路径是否存在进行检查。

请参阅

BuildPath 方法，DriveExists 方法，GetAbsolutePathName 方法，GetBaseName 方法，GetDrive 方法，GetExetnsionName 方法，GetFile 方法，GetFileName 方法，GetFolder 方法，GetParentFolderName 方法，GetSpecialFolder 方法，GetTempName 方法

GetExtensionName 方法

返回一个包含路径中最后部件扩展名的字符串。

应用于

FileSystemObject 对象

语法

object.**GetExtensionName**(*path*)

GetExtensionName 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
path	必需的。.要返回其扩展名的部件的路径说明

说明

对于网络驱动器，根目录（\）被认为是一个部件。

如果没有部件和 path 参数相匹配，GetExtensionName 方法返回一个长度为零的字符串("")。

请参阅

BuilePath 方法,GetBaseName 方法,GetDrive 方法,GetDriveName 方法，GetFile 方法，GetFolder 方法，GetParentFolderName 方法，GetSpecialFolder 方法，GetTempName 方法

GetFile 方法

返回一个和指定路径中文件相对应的 File 对象。

应用于

FileSysetmObject 对象

语法

object.**GetFile**(*filespec*)

GetFile 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject.的名字
filespec	必需的。Filespec 是到一个指定文件的路径（绝对的或相对的）

说明

如果指定的文件不存在，则发生一个错误。

请参阅

FileExists 方法，GetAbsolutePathName 方法，GetBaseName 方法，GetDrive 方法，GetDriveName 方法，GetExtensionName 方法，GetFileName 方法，GetFolder 方法，GetParentFolderName 方法，GetSpecialFolder 方法，GetTempName 方法，MoveFile 方法

GetFileName 方法

返回指定路径中的最后部件，该路径不是驱动器说明的一部分。

应用于

FileSystemObject 对象

语法

object.**GetFileName**(*pathspec*)

GetFileName 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
pathspec	必需的。到一个指定文件的路径（绝对的或相对的）

说明

如果 pathspec 不是以已命名部件结尾，GetFileName 方法返回一个零长度字符串（""）。

注意 GetFileName 方法仅在提供的路径字符串上起作用。它没有尝试去辨认路径，也不对指定路径是否存在进行检查。

请参阅

BuildPath 方法，FileExists 方法，GetAbsolutePathNane 方法，GetBaseName 方法，GetDrive 方法，GetDriveName 方法，GetExtensionName 方法，GetFile 方法，GetFolder 方法，GetParentFolderName 方法，GetSpecialFolder 方法，GetTempName 方法，MoveFile 方法

GetFolder 方法

返回一个和指定路径中文件夹相对应的 Folder 对象。

应用于

FileSystemObject 对象

语法

object.**GetFolder**(*folderspec*)

GetFolder 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject.的名字
folderspec	必需的。Folderspec 是指定文件夹的路径（绝对的和相对的）

说明

如果指定的文件夹不存在，则发生一个错误。

请参阅

FolderExists 方法，GetAbsolutePathName 方法，GetBaseName 方法，GetDrive 方法，GetDriveName 方法，GetExtensionName 方法，GetFile 方法，GetFileName 方法，GetParentFolderName 方法，GetSpecialFolder 方法，GetTempName 方法，MoveFolder 方法

GetFormat 方法

返回一个整数，指出 Clipboard 对象中的项目是否匹配期望的格式。不支持命名参数。

应用于

Clipboard 对象

语法

object.**GetFormat**(*format*)

GetFormat 方法的语法包含下列部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
format	必需的。一个数值或常数，如“设置值”中所描述的，它指定 Clipboard 对象的格式。必须用括号包括该常数或数值

设置值

用于 format 的设置值有：

常数	值	描述
vbCFLink	&HBF00	DDE 对话信息
vbCFText	1	文本
vbCFBitmap	2	位图（.bmp 文件）
VbCFMetafile	3	元文件（.wmf 文件)
vbCFDIB	8	设备无关位图(DIB)
vbCFPalette	9	调色板

说明

上述常数在 VisualBasic(VB)对象浏览器中的对象库里列出。

如果 Clipboard 对象中一个项目匹配指定的格式，则 GetFormat 方

法返回 True。否则，返回 False。

对于 vbCFDIB 和 vbCFBitmap 两种格式，显示图形时不管 Clipboard 中是什么样的调色板都要使用。

请参阅

GetData 方法，GetText 方法，SetData 方法，SetText 方法

示例

本示例使用 GetFormat 方法确定 Clipboard 对象上数据的格式。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   '定义位图各种格式。
   DimClpFmt,Msg    '声明变量。
   OnErrorResumeNext    '设置错误处理。
   IfClipboard.GetFormat(vbCFText)ThenClpFmt=ClpFmt+1
   IfClipboard.GetFormat(vbCFBitmap)ThenClpFmt=ClpFmt+2
   IfClipboard.GetFormat(vbCFDIB)ThenClpFmt=ClpFmt+4
   IfClipboard.GetFormat(vbCFRTF)ThenClpFmt=ClpFmt+8
   SelectCaseClpFmt
      Case1
         Msg="TheClipboardcontainsonlytext."
      Case2,4,6
         Msg="TheClipboardcontainsonlyabitmap."
```

```
        Case3,5,7
            Msg="TheClipboardcontainstextandabitmap."
        Case8,9
            Msg="TheClipboardcontainsonlyrichtext."
        CaseElse
            Msg="ThereisnothingontheClipboard."
    EndSelect
    MsgBoxMsg    '显示信息。
EndSub
```

GetFormat 方法（DataObject 对象）

返回布尔值，指出 DataObject 对象中的一项是否与规定格式匹配。不支持命名参数。

应用于

DataObject 对象

语法

object.**GetFormat***format*

GetFormat 方法语法包含下面部分：

部分	描述
object	必需的。对象表达式，其值是“应用于”列表中的一个对象
format	一个确定数据格式的常数或值，如“设置值”所述

设置值

format 设置如下：

常数	值	描述
VbCFText	1	文本（.txt 文件）
VbCFBitmap	2	位图（.bmp 文件）
VbCFMetafile	3	元文件（.wmf 文件）
VbCFEMetafile	14	增强元文件（.emf 文件）
VbCFDIB	8	与设备无关的位图(DIB)
VbCFPalette	9	调色板
VbCFFiles	15	文件列表
VbCFRTF	-16639	丰富文本格式（.rtf 文件）

说明

这些常数列于 VisualBasic(VB)对象浏览器中的对象库。

如果 DataObject 对象中的项与规定格式匹配，GetFormat 方法返回 True，否则返回 False。

GetObject 函数

返回文件中的 ActiveX 对象的引用。

语法

GetObject([*pathname*] [,*class*])

GetObject 函数的语法包含下面几个命名参数：

部分	描述
pathname	可选的；Variant(String)。包含待检索对象的文件的全路径和名称。如果省略 *pathname*，则 *class* 是必需的
class	可选的；Variant(String)。代表该对象的类的字符串

其中，class 参数的语法格式为 appname.objecttype，且语法的各个部分如下：

部分	描述
appname	必需的；Variant(String)。提供该对象的应用程序名称
objecttype	必需的；Variant(String)。待创建对象的类型或类

说明

使用 GetObject 函数可以访问文件中的 ActiveX 对象，而且可以将该对象赋给对象变量。可以使用 Set 语句将 GetObject 返回的对象赋给对象变量。例如：

```
DimCADObjectAsObject
SetCADObject=GetObject("C:\CAD\SCHEMA.CAD")
```

当执行上述代码时，就会启动与指定的 pathname 相关联的应用程序，同时激活指定文件中的对象。

如果 pathname 是一个零长度的字符串("")，则 GetObject 返回指定类型的新的对象实例。如果省略了 pathname 参数，则 GetObject 返回指定类型的当前活动的对象。如果当前没有指定类型的对象，

就会出错。
有些应用程序允许只激活文件的一部分，其方法是在文件名后加上一个惊叹号(!)以及用于标识想要激活的文件部分的字符串。关于如何创建这种字符串的信息，请参阅有关应用程序创建对象的文档。
例如，在绘图应用程序中，一个存放在文件中的图可能有多层。可以使用下述代码来激活图中被称为 SCHEMA.CAD 的层：

```
SetLayerObject=GetObject("C:\CAD\SCHEMA.CAD!Layer3")
```

如果不指定对象的 class，则自动化会根据所提供的文件名，来确定被启动的应用程序以及被激活的对象。不过，有些文件可能不止支持一种对象类。例如，图片可能支持三种不同类型的对象：Application 对象，Drawing 对象以及 Toolbar 对象，所有这些都是同一个文件中的一部分。为了说明要具体激活文件中的哪种对象，就应使用这个可选的 class 参数。例如：

```
DimMyObjectAsObjectSetMyObject=GetObject("C:\DRAWINGS\SAMPLE.DRW
","FIGMENT.DRAWING")
```

在上述例子中，FIGMENT 是一个绘图应用程序的名称，而 DRAWING 则是它支持的一种对象类型。
对象被激活之后，就可以在代码中使用所定义的对象变量来引用它。在前面的例子中，可以使用对象变量 MyObject 来访问这个新对象的属性和方法。例如：

```
MyObject.Line9,90
MyObject.InsertText9,100,"Hello,world."
MyObject.SaveAs"C:\DRAWINGS\SAMPLE.DRW"
```

注意当对象当前已有实例，或要创建已加载的文件的对象时，就使用 GetObject 函数。如果对象当前还没有实例，或不想启动已加载文件的对象，则应使用 CreateObject 函数。

如果对象已注册为单个实例的对象，则不管执行多少次 CreateObject，都只能创建该对象的一个实例。若使用单个实例对象，当使用零长度字符串("")语法调用时，GetObject 总是返回同一个实例，而若省略 pathname 参数，就会出错。不能使用 GetObject 来获取 VisualBasic 创建的类的引用。

请参阅

CreateObject 函数，Set 语句

示例

该示例使用 GetObject 函数来获取对指定的 MicrosoftExcel 的工作表(MyXL)的引用。它使用工作表的 Application 属性来显示或关闭 MicrosoftExcel 等等。DetectExcelSub 过程通过调用两个 API 函数，来查找 MicrosoftExcel。如果 MicrosoftExcel 正在运行，则将其放入运行对象表（RunningObjectTable）中。如果 MicrosoftExcel 不在运行，则第一次调用 GetObject 将导致错误。在本例中，出现该错误则把 ExcelWasNotRunning 标志设为 True。第二次调用 GetObject 是指定要打开的一个文件。如果

MicrosoftExcel 不在运行，则这个第二次的调用将启动该程序，并返回一个指定文件(mytest.xls)所对应的工作表的引用。该文件必须位于指定的位置；否则将产生 VisualBasic 错误及自动化错误。随后的示例代码将 MicrosoftExcel 及包含指定工作表的窗口设为可见。最后，如果在此前没有 MicrosoftExcel 的副本在运行，代码就使用 Application 对象的 Quit 方法来关闭 MicrosoftExcel。如果该应用程序原来就在运行，则不要试图关闭它。引用本身在设为 Nothing 后被释放。

```
'声明必要的API例程：
DeclareFunctionFindWindowLib"user32"Alias_
"FindWindowA"(ByVallpClassNameasString,_
                    ByVallpWindowNameAsLong)AsLong

DeclareFunctionSendMessageLib"user32"Alias_
"SendMessageA"(ByValhWndasLong,ByValwMsgasLong_
                    ByValwParamasLong,_
                    ByVallParamAsLong)AsLong

SubGetExcel()
  DimMyXLAsObject '用于存放
                                  'MicrosoftExcel引用的变量。
  DimExcelWasNotRunningAsBoolean  '用于最后释放的标记。
```

```
'测试MicrosoftExcel 的副本是否在运行。
   OnErrorResumeNext     '延迟错误捕获。
'不带第一个参数调用 Getobject 函数将
'返回对该应用程序的实例的引用。
'如果该应用程序不在运行，则会产生错误。
   SetMyXL=Getobject(,"Excel.Application")
   IfErr.Number<>0ThenExcelWasNotRunning=True
   Err.Clear     '如果发生错误则要清除 Err 对象。

'检测MicrosoftExcel。如果MicrosoftExcel 在运行，
'则将其加入运行对象表。
   DetectExcel

'将对象变量设为对要看的文件的引用。
   SetMyXL=Getobject("c:\vb4\MYTEST.XLS")

'设置其 Application 属性，显示MicrosoftExcel。
'然后使用 MyXL 对象引用的Windows 集合
'显示包含该文件的实际窗口。
   MyXL.Application.Visible=True
   MyXL.Parent.Windows(1).Visible=True
   '在此处对文件
```

```
    '进行操作。
    '...
'如果在启动时，MicrosoftExcel 的这份副本不在运行中，
'则使用 Application 属性的 Quit 方法来关闭它。
'注意，当试图退出 MicrosoftExcel 时，
'标题栏会闪烁，并显示一条消息
'询问是否保存所加载的文件。
    IfExcelWasNotRunning=TrueThen
        MyXL.Application.Quit
    EndIF

    SetMyXL=Nothing  '释放对该应用程序
                     '和电子数据表的引用。
EndSub

SubDetectExcel()
'该过程检测并登记正在运行的 Excel。
    ConstWM_USER=1024
    DimhWndAsLong
'如果 Excel 在运行，则该 API 调用将返回其句柄。
    hWnd=FindWindow("XLMAIN",0)
    IfhWnd=0Then'0 表示没有 Excel 在运行。
```

```
        ExitSub
    Else
    'Excel 在运行，因此可以使用 SendMessageAPI
    '函数将其放入运行对象表。
        SendMessagehWnd,WM_USER+18,0,0
    EndIf
EndSub
```

GetParentFolderName 方法

返回一个包含指定路径最后部件父文件夹名字的字符串。

应用于

FileSystemObject 对象

语法

object.**GetParentFolderName**(*path*)

GetParentFolderName 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
path	必需的。要返回其父文件夹名字的部件的路径说明

说明

如果 path 参数指定的部件没有父文件夹，则 GetParentFolderName 方法返回一个零长度字符串（""）。

注意 GetParentFolderName 方法仅对提供的 path 字符串起作用。它

没有尝试去辨认路径，也不对指定路径是否存在进行检查。

请参阅

BuildPath 方法，FolderExists 方法，GetAbsolutePathName 方法，GetBaseName 方法，GetDrive 方法，GetDriveName 方法，GetExtensionName 方法，GetFile 方法，GetFileName 方法，GetFolder 方法，GetSpecialFolder 方法，GetTempName 方法，MoveFolder 方法

GetSelection 方法

返回在代码窗格中所选的内容。

应用于

CodePane 对象

语法

object.GetSelection(startline,startcol,endline,endcol)

GetSelection 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
startline	必需的。一个 Long，可返回一个值用来指示在代码窗格中所选的起始行
startcol	必需的。一个 Long 型数，可返回一个值用来指示在代码窗口中所选的起始列
endline	必需的。一个 Long 型数，可返回一个值用来指示在代码窗口中所选的最后一行
endcol	必需的。一个 Long 型数，可返回一个值用来指示在代码窗口中所选的最后一列

说明

使用 GetSelection 方法时，在输出参数中将有返回信息。因为变量将被更改以便包含返回信息，所以必须用变量进行传递。

请参阅

DeleteLines 方法，Find 方法(VBA 外接程序对象模型)，InsertLines 方法，Lines 方法，CodeModule 对象，ProcBodyLine 属性，ProcCountLines 属性，ProcCfLine 属性，ProcStartLine 属性

示例

下例返回在代码窗格 CodePane(1)中当前选择的起点和终点位

置。例中最后一行使用 GetSelection 方法为 4 个变量赋值。

```
DimmAsLong
DimnAsLong
DimxAsLong
DimyAsLong
Application.VBE.CodePanes(1).GetSelectionm, n, x, y
```

GetSetting 函数

从 Windows 注册表中或(Macintosh 中)应用程序初始化文件中的信息的应用程序项目返回注册表项设置值。

语法

GetSetting(*appname*,*section*,*key*[,*default*])

GetSetting 函数的语法具有下列命名参数：

部分	描述
appname	必需的。字符串表达式，包含应用程序或工程的名称，要求这些应用程序或工程有注册表项设置。在 Macintosh 中，这是 System 文件夹中 Preferences 文件夹中初始化文件的文件名
section	必需的。字符串表达式，包含区域名称，要求该区域有注册表项设置
key	必需的。字符串表达式，返回注册表项设置的名称
default	可选的。表达式，如果注册表项设置中没有设置值，则返回缺省值。如果省略，则 *default* 取值为长度为零的字符串("")

说明

如果 GetSetting 的参数中的任何一项都不存在，则 GetSetting 返回 default 的值。

请参阅

DeleteSettting 语句，GetAllSettings 函数，SaveSetting 语句

示例

本示例首先使用 SaveSetting 语句来建立 WindowsMacintosh 注册区（或 16 位 Windows 平台的.ini 档）里 appname 应用程序的项目，然后使用 GetSetting 函数来得到其中一项设置并显示出来。

因为有传入参数 default，GetSetting 函数一定会有返回值。请注意，section 名称不能用 GetSetting 函数取得。最后，使用 DeleteSetting 语句将该应用程序项删除。

```
'用来保存 GetSetting 函数所返回之二维数组数据的变量。
DimMySettingsAsVariant
'在注册区中添加项目。
SaveSetting"MyApp","Startup","Top",75
SaveSetting"MyApp","Startup","Left",50

Debug.PrintGetSetting(appname:="MyApp",section:="Startup",_
key:="Left",default:="25")

DeleteSetting"MyApp","Startup"
```

GetSpecialFolder 方法

返回指定的特殊文件夹。

应用于

FileSystemObject 对象

语法

object.**GetSpecialFolder**(*folderspec*)

GetSpecialFolder 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
folderspec	必需的。要返回的特殊文件夹的名字。可以是在设置值部分中列出的任何常数

设置值

folderspec 参数可为任何的下列值：

常数	值	描述
WindowsFolder	0	Windows 文件夹，包含由 Windows 操作系统安装的文件
SystemFolder	1	系统文件夹，包含库、字体、设备驱动程序
TemporaryFolder	2	Temp 文件夹，用于存储临时文件。它的路径在 TMP 环境变量中

请参阅

GetAbsolutePathNane 方法，GetBaseName 方法，GetDrive 方法，GetDriveName 方法，GetExtensionName 方法，GetFile 方法，GetFileName 方法，GetFolder 方法，GetParentFolderName，GetTempName 方法

GetTempName 方法

返回一个随机产生的临时文件或文件夹的名字，该名字在执行需要临时文件或文件夹的操作时有用。

应用于

FileSystemObject 对象

语法

object.**GetTempName**

可选的 object 始终是一个 FileSystemObject .的名字。

说明

GetTempName 方法不产生一个文件，它仅提供一个临时文件名字，该名字可被 CreateTextFile 用于创建一个文件。

请参阅

BuildPath 方法，GetAbsolutePathName 方法，GetBaseName 方法，GetDrive 方法，GetDriveName 方法，GetExtendionName 方法，GetFile 方法，GetFileName 方法，GetFolder 方法，GetParentFolderName 方法，GetSpecialFolder 方法

GetText 方法

用于返回 Clipboard 对象中的文本字符串。不支持命名参数。

应用于

Clipboard 对象

语法

object.**GetText**(*format*)

GetText 方法的语法包含下列部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
format	可选的。一个数值或常数，如“设置值”中所描述的，它指定 Clipboard 对象的格式。必须用括号将常数或数值括起来

设置值

format 的设置值有：

常数	值	描述
VbCFLink	&HBF00	DDE 对话信息
VbCFText	1	（缺省值）文本
VbCFRTF	&HBF01	RTF（.rtf 文件）

说明

上述常数在 VisualBasic(VB)对象浏览器中的对象库里列出。

如果 Clipboard 对象中没有与期望的格式相匹配的字符串，则返回一个零长度字符串("")。

请参阅

GetData 方法，GetFormat 方法，SetData 方法，SetText 方法

示例

本示例使用 GetText 方法从 Clipboard 对象中复制一个文字串至一字符串变量。要检验此示例，可将本例代码粘贴到一个带有一名

为 Text1 的 TextBox 控件的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
    DimI,Msg,Temp    '声明变量。
    OnErrorResumeNext    '设置错误处理。
    Msg="Typeanythingyoulikeintothetextboxbelow."
    Text1.Text=InputBox(Msg)'取得用户正文。
    Msg="ChooseOKtocopythecontentsofthetextbox"
    Msg=Msg&"totheClipboard."
    MsgBoxMsg    '显示信息。
    Clipboard.Clear  '清除剪贴板。
    Clipboard.SetTextText1.Text'将正文放置在剪贴板上。
    IfClipboard.GetFormat(vbCFText)Then
        Text1.Text=""    '清除该正文框。
        Msg="ThetextisnowontheClipboard.ChooseOK"
        Msg=Msg&"tocopythetextfromtheClipboardback"
        Msg=Msg&"tothetextbox."
        MsgBoxMsg    '显示信息。
        Temp=Clipboard.GetText(vbCFText)'取得剪贴板正文。
        ForI=Len(Temp)To1Step-1  '使该正文反向。
            Text1.Text=Text1.Text&Mid(Temp,I,1)
        NextI
```

```
    Else
        Msg="ThereisnotextontheClipboard."
        MsgBoxMsg     '显示错误信息。
    EndIf
EndSub
```

Global 对象

Global 对象是应用程序对象，使用该对象，就可对应用程序级的属性和方法进行存取。

语法

Global

说明

Global 是 Object 数据类型的。因为 Global 对象是一个被自动引用的应用程序对象，所以不需要为引用该对象而编制特定的代码。

属性

App 属性，Clipboard 属性，Forms 属性，Printer 属性，Printers 属性，Screen 属性

方法

Load 语句，Unload 语句，LoadPicture 函数，SavePicture 语句，LoadResData 函数，LoadResPicture 函数，LoadResString 函数

请参阅

App 对象

GoBack 方法

执行历史列表中的超链接后跳。

应用于

Hyperlink 对象

语法

object.**GoBack**

GoBack 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

如果对象在支持 OLE 超链接的容器中，那么容器将回跳至历史列表中的上一个位置。如果对象在不支持 OLE 超连接的容器中，该方法将产生错误。

请参阅

GoForward 方法

GoForward 方法

执行历史列表中的超链接上跳。

应用于

Hyperlink 对象

语法

object.**GoForward**

GoForward **方法的语法包含下面部分：**

部分	**描述**
object	**对象表达式，其值为“应用于”列表中的对象**

说明

如果对象在支持 OLE 超链接的容器中，那么容器将后跳至历史列表中的下一个位置。如果对象在不支持 OLE 超链接的容器中，该方法将产生错误。

请参阅

GoBack 方法

GoSub...Return 语句

在一个过程中跳到另一个子程序中执行，执行后再返回。

语法

GoSub*line*

...

line

...

Return

必要的 line **参数可以是任何行标签或行号。**

说明

可以在过程中的任何地方使用 GoSub 和 Return，但是 GoSub 和与之相应的 Return 语句必须放在同一个过程中。一个子程序中可以包含一个以上的 Return 语句，但是当碰到第一个 Return 语句时，程序就会返回到紧接在刚刚执行的 GoSub 语句之后的语句继续执行。

注意不能使用 GoSub...Return 来进入或退出 Sub 过程。

提示创建分开的过程，并使用 GoSub...Return 来调用，可以使程序更具结构化。

请参阅

GoTo 语句，On…GoSub，On…GoTo 语句，Sub 语句

示例

本示例使用 GoSub 来调用子过程里的一段子程序。Return 语句则将执行返回到 GoSub 的下一个语句。ExitSub 语句则是用来避免控制意外进入该子程序的情形发生。

```
SubGosubDemo()
DimNum
'请求用户输入一个数字。
   Num=InputBox("Enterapositivenumbertobedividedby2.")
'如果用户输入一个正整型，则使用子程序。
   IfNum>0ThenGoSubMyRoutine
   Debug.PrintNum
   ExitSub '使用 Exit 命令来避免错误发生。
```

```
MyRoutine:
    Num=Num/2    '将数除以2。
    Return    '将控制返回 GoSub 之后的语句。
EndSub
```

GotFocus 事件

当对象获得焦点时产生该事件；获得焦点可以通过诸如 TAB 切换，或单击对象之类的用户动作，或在代码中用 SetFocus 方法改变焦点来实现。

应用于

ADOData 控件，TreeView 控件，ImageCombo 控件，ListView 控件,Slider 控件,TabStrip 控件,DataTimePicker控件,Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，Animation 控件，DataList 控件，DBCombo 控件，DBList 控件，MadkedEdit 控件，MSHFlexGrid 控件，MSFlexGrid 控件，RichTextBox 控件，Extender 对象，PropertyPage 对象，UserControl 对象,UserDocument 对象,CheckBox 控件,ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，HScrollBar，VScrollBar 控件,ListBox 控件,OptionButton 控件,PictureBox 控件，TextBox 控件，OLEContainer 控件

语法

PrivateSubForm_GotFocus()

PrivateSub*object*_GotFocus([*index***AsInteger**])

GotFocus 事件包含下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件

说明

通常，GotFocus 事件过程用以指定当控件或窗体首次接收焦点时发生的操作。例如，通过给窗体上每个控件附加一个 GotFocus 事件过程，就可以显示简要说明或状态条信息给外界提供指导。根据获取焦点控件的不同，通过使其有效、禁止或者是显示其它控件的方式，也可以提供出可视的提示。

注意一个控件仅当其 Enabled 和 Visible 属性都设置为 True 时才能接收焦点。为了移动焦点，在 VisualBasic 中要自定义键盘接口，对于窗体上的控件需设置 Tab 键顺序或指定访问键。

请参阅

LostFocus 事件，SetFocus 方法，TabIndex 属性，TabStop 属性，ActiveControl 属性，ActiveForm 属性

示例

本例在 OptionButton 组中的一个按钮得到焦点时显示一个状态栏信息。要尝试这个例子，可将代码粘贴到一个包含两个 OptionButton 控件和一个 Label 的窗体的声明部分。将两个 OptionButton 控件的

Name 属性都设置为 OptionGroup，然后按 F5 键并单击 OptionButton 控件。

```
PrivateSubForm_Load()
   Label1.AutoSize=True
EndSub

PrivateSubOptionGroup_GotFocus(IndexAsInteger)
   SelectCaseIndex
        Case0
            Label1.Caption="Option1hasthefocus."
        Case1
            Label1.Caption="Option2hasthefocus."
   EndSelect
EndSub

PrivateSubOptionGroup_LostFocus(IndexAsInteger)

   Label1.Caption=""

EndSub
```

GotFocus 事件（UserControl 对象和 UserDocument 对象）

当焦点进入对象或子控件时，发生该事件。

应用于

UserControl 对象，UserDoccument 对象

语法

Sub*object*_GotFocus()

GotFocus 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

GotFocus 事件不是使用 object 的开发者处理的 GotFocus 扩展事件。该 GotFocus 事件为 object 的创建者提供，而且属于 object 内部。

通过该事件，可使 object 获知控件是否获得了焦点。

只有当 CanGetFocus 属性设置为 True，并且没有能够接受焦点的子控件时，object 本身才能获得焦点。

EnterFocus 事件在 GotFocus 事件前产生。

不要在该事件中产生 GotFocus 扩展事件。

请参阅

CanGetFocus 属性，EnterFocus 事件，LostFocus 事件

GoTo 语句

无条件地转移到过程中指定的行。

语法

GoTo*line*

必要的 line 参数可以是任意的行标签或行号。

说明

GoTo 只能跳到它所在过程中的行。

注意太多的 GoTo 语句，会使程序代码不容易阅读及调试。尽可能使用结构化控制语句(Do...Loop 、 For...Next 、 If...Then...Else 、 SelectCase)。

请参阅

Do…Loop 语句,For…Next 语句,GoSub…Return 语句,If…Then…Else 语句，SelectCase 语句

示例

本示例使用 GoTo 语句在一个过程内的不同程序段间作流程控制，不同程序段用不同的“程序标签”来区隔。

```
SubGotoStatementDemo()
DimNumber,MyString
   Number=1'设置变量初始值。
   '判断 Number 的值以决定要完成那一个程序区段（以“程序标签”来
表式)。
   IfNumber=1ThenGoToLine1ElseGoToLine2
```

```
Line1:
    MyString="Numberequals1"
    GoToLastLine'完成最后一行。
Line2:
    '下列的语句根本不会被完成。
    MyString="Numberequals2"
LastLine:
    Debug.PrintMyString '将“"Numberequals1"”显示在“立即”窗口。
EndSub
```

GridLineWidth 属性

返回或设置 MSFlexGrid **控件网格线的像素数的宽度。**

语法

object.**GridLineWidth**[=*value*]

GridLineWidth **属性的语法包含下面部分：**

部分	**描述**
object	**对象表达式，其值是“应用于”列表中的一个对象**
value	**一个指定网格线宽度的整数。最小设置值为** 1 **（缺省值）；最大设置值为** 10

GridX、GridY 属性

指定设计时对齐网格的间隔（运行时被忽略）。

应用于

DataReport 对象

语法

object.**GridX**

object.**GridY**

object 所在处表示一个对象表达式，其值为“应用于”列表中的一个对象。

说明

GridX 和 GridY 属性只在设计时用作在 DataReport 对象上对齐控件的一种辅助手段。

GUID 属性

返回一个 String，其内容为一个对象的类标识符，此属性为只读。

应用于

Reference 对象，AddIn 对象

请参阅

AddFromGuid 方法，FullPath 属性

示例

下列示例使用 GUID 属性返回指定工程中指定的 Reference 对象的全局唯一标识符。

```
Debug.PrintApplication.VBE.VBProjects(1).References(1).GUID
```

Handle 属性

返回一个句柄，指向 Picture 对象内包含的图形。

应用于

Picture 对象

语法

object.**Handle**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

返回值

Handle 属性的返回值取决于当前 Type 属性的设置，如下表所示：

Type 属性	返回值
1（位图）	HBITMAP 句柄
2（元文件）	HMETAFILE 句柄
3（图标）	HICON 或 HCURSOR 句柄
4（增强的元文件）	HENHMETAFILE 句柄

说明

作为调用动态链接库(DLL)或 WindowsAPI 函数的一个部分需传递图形的句柄时，Handle 属性是有用的。

请参阅

Type 属性(Picture)

HasDC 属性

返回或设置一个值，以决定唯一显示上下文（或 hDC）是否被分配给控件。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Monthview 控件，Form 对象，Forms 集合，PictureBox 控件

语法

object.**HasDC**[=*boolean*]

HasDC 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指示该控件是否显示只读属性的文件，“设置值”中有详细描述

设置值

boolean 的设置值如下：

设置值	描述
true	(缺省)分配一个 hDC 给控件
false	不分配 hDC 给控件

说明

如果 HasDC 属性设置为 False，则只能将控件的 hDC 复制到一个局部变量。用一个事件范围之外得到的 hDC 调用 APIs 会引起冲突或其它不可预料的结果。

对于无窗口的 UserControls，只有 UserControl 是由一个窗口（即，它是在一个不支持无窗口激活的容器中，例如 VisualBasic4.0 版或 InternetExplorer3.0 版）创建时，HasDC 属性才起作用。被无窗口激活的无窗口 UserControls 决不会有自己的 hDC，因此忽略它们的 HasDC 属性值。

请参阅

hWnd 属性

HasOpenDesigner 属性

返回一个布尔值，指定此 VBComponent 对象是否具有一个打开设计器，此属性为只读。

应用于

VBComponent 对象

返回值

HasOpenDesigner 属性返回下列这些值：

值	描述
true	VBComponent 对象具有一个打开的设计窗口
false	VBComponent 对象没有打开设计窗口

请参阅

DesignerWindow 方法，Designer 属性

示例

下列示例使用 HasOpenDesigner 属性返回某工程中的指定部件是否拥有开放式的设计器。

```
Debug.PrintApplication.VBE.VBProjects(1).VBComponents(1).HasOpen
Designer
```

hDC 属性

返回一个句柄，该句柄是由 MicrosoftWindows 运行环境提供给一个对象的设备描述体。

应用于

CommonDialog 控件 (PrintDialog)，PropertyPage 对象，UserControl 对象，UserDocument 对象，Print 对象，Printers 集合，Form 对象，Forms 集合，PictureBox 控件

语法

object.**hDC**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

该属性是 Windows 运行环境的设备关联句柄。Windows 运行环境，通过给 Grid 对象和应用程序中每个 Grid 和 Grid 控件分配一个设备关联，管理系统显示。可以用 Print()属性引用对象的设备关联句柄。这提供了一个传递给 WindowsAPI 调用的值。

对于 Printer 窗体 Picture 控件，在设置 cdlReturnDC 标志时，该属性为打印机对话框中选择的打印机，返回一设备关联，或设置 cdlReturnIC 标志时，返回一信息关联。

hDC 属性值可以在程序运行中改变，因此不要将该值存储在变量中，应在每次需要时使用 hDC 属性。

AutoRedraw 属性能引起 hDC 属性改变。如果窗体或窗体 PictureBox 容器的 AutoRedraw 属性设置为 True，hDC 将作为持久图形的设备关联句柄（等价于 Image 属性）。当 AutoRedraw 属性为 False 时，hDC 属性是窗体窗口或 PictureBox 容器的真正的 hDC 值。程序运行中，不论 AutoRedraw 设置为何值，hDC 属性都可以改变。

如果 HasDC 属性设为 False,新的设备上下文将由系统建立，并且 hDC 属性值在每次被调用时改变。

请参阅

Icon 属性，AutoRedraw 属性，hWnd 属性

示例

这个例子画一个三角形，然后使用 MicrosoftWindows 的函数用颜色填充该三角形。要试用此例，先用“工程”菜单中的“添加

模块”命令创建一个新的模块。把 Declare 语句粘贴到新模块的声明部分，确保该语句在一行，并且没有断点或隐藏字。然后把 Sub 过程粘贴到窗体的声明部分。按 F5 键，并单击窗体。

```
'声明 Windows 例程。该语句是模块的。
DeclareSubFloodFillLib"GDI32"Alias"FloodFill"_
(ByValhDCAsLong,ByValXAsLong,ByValYAs_
Long,ByValcrColorAsLong)AsLong
'将下列代码放入窗体。
PrivateSubForm_Click()
    ScaleMode=vbPixels                    'Windows 用像素画。
    ForeColor=vbBlack                     '设置画的线为黑色。
    Line(100,50)-(300,50)                 '画一个三角形。
    Line-(200,200)
    Line-(100,50)
    FillStyle=vbFSSolid             '设置 FillStyle 为实线。
    FillColor=RGB(128,128,255)              '设置 FillColor。
'调用 WindowsAPI 填充。
    FloodFillhDC,200,100,ForeColor
EndSub
```

Height 属性（VBA 外接程序对象模型）

返回或设置一个 Single 数，其内容为以缇为单位的窗口高度，

此属性可读/写。

应用于

Windows 对象

说明

只要窗口保持链接或连接，改变链接窗口或连接窗口的 Height 属性设置没有任何效果。

请参阅

Left 属性，Top 属性，Width 属性

示例

下列示例使用 Height 及 Width 属性返回指定的窗口之高度与宽度（以缇为单位）。这些属性的设置值在窗口被链接或连接后会有所改变，因为此后它们引用了原始窗口与之链接或连接的 Window 对象。

```
Debug.PrintApplication.VBE.Windows(9).Height
Debug.PrintApplication.VBE.Windows(9).Width
```

Height、Width 属性

返回或设置对象的维数、或 DataGrid 控件 Columns 对象的宽度。对于 Printer 和 Screen 对象，在设计时不可用。

应用于

ADOData 控件，DataRepeater 控件，Column 对象，DataList 控件，RemoteData 控件，Data 控件，Function 控件(数据报表设

计器)，ImageControl(数据报表设计器)，Label 控件(数据报表设计器)，Line 控件(数据报表设计器)，Section 对象(数据报表设计器)，Shape 控件(数据报表设计器)，TextBox 控件(数据报表设计器)，Picture 对象，PropertyPage 对象，UserControl 对象，UserDocument 对象，Screen 对象，Printer 对象，Printers 集合，CheckBox 控件，ComboBox 控件，CommadButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HscrollBar，VscrollBar 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，OLE 容器控件

语法

object.**Height**[=*number*]

object.**Width**[=*number*]

Height 和 Width 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	数值表达式，指定对象的维数，“设置值”中有详细说明

设置值

大小如下计算：

Form——窗体的外部高度和宽度，包括边框和标题栏。

Control——从控件边框的中心度量，以使边框宽度不同的控件能够正确对齐。这些属性使用控件容器的度量单位

Printer 对象——为打印设备设置的纸张物理尺寸，在设计时无效。如果在运行时设置该属性，则使用这些属性的值而不用 PaperSize 属性的设置。

Screen 对象——屏幕的高度和宽度；在设计时无效，在运行时为只读。

Picture 对象——以 HiMetric 为单位的图片的高度和宽度。

说明

对于 Form、Printer 和 Screen 对象，这些属性值是以缇来度量的。对于窗体或控件，这些属性值随着用户或你的代码调整对象大小而改变。所有对象的这些属性的最大值与系统有关。

对不允许设置这些属性的打印机驱动程序，若设置 Height 和 Width 属性，不会发生错误，但纸张的大小保持不变。对只允许某些缇值的打印机驱动程序，设置 Height 和 Width 属性时，不会发生错误，且该属性被设置为驱动程序所允许的值。例如，可以将 Height 设置成 150，但驱动程序可能会把它设置成 144。

对基于对象全区代作或计算，如改变对象大小或移动对象，要使用 Height、Width、Left 和 Top 属性。对基于对象内部区域的操作或计算，如在一对象内绘制或移动对象，要使用 ScaleLeft、ScaleTop、ScaleHeight 和 ScaleWidth 属性。

注意 DriveListBox 控件或 ComboBox 控件的 Height 属性不能改变，

这两控件的 Style 属性设置为 0（下拉组合框）或 2（下拉列表框）。

对于 DataGrid 控件的 Columns 对象，Width 按包含 DataGrid 的对象的度量单位来指定。Width 的缺省值为 DataGrid 的 DefColWidth 属性值。

对于 Picture 对象，用 ScaleX 和 ScaleY 方法将 HiMetric 单位转换为所需的单位。

请参阅

Move 方法，ScaleX，ScaleY 方法，Left，Top 属性，PaperSize 属性，ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ClientHeight，ClientWidth，ClientLeft，ClientTop 属性

示例

这个例子在窗体被加载时，将窗体的大小设置为屏幕大小的百分之七十五并使窗体居中显示。要尝试这个例子，请将代码粘贴到窗体的声明部分。然后按 F5 键并单击窗体。

```
PrivateSubForm_Click()
   Width=Screen.Width*.75   '设置窗体的宽度。
   Height=Screen.Height*.75'设置窗体的高度。
   Left=(Screen.Width-Width)/2  '在水平方向上居中显示。
   Top=(Screen.Height-Height)/2'在垂直方向上居中显示。
EndSub
```

HelpCommand 属性

返回或设置需要的联机帮助的类型。

应用于

CommonDialog 控件

语法

object.HelpCommand[=value]

HelpCommand 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	如“设置值”所描述，它是指定帮助类型的常数或值

设置值

value 的设置值是：

常数	值	描述
cdlHelpCommand	&H102 &	执行帮助宏
cdlHelpContents	&H3&	按照.hpj 文件[OPTION]节中内容选项所定义的那样，显示帮助内容主题。参阅下面的说明，以获得有关用 MicrosoftHelpWorkshop4.0X 创建的 Help 文件的信息
cdlHelpContext	&H1&	为特定的上下文显示帮助。当使用这

		个设置时，还必须用 HelpContext 属性指定一个上下文
cdlHelpContextPopup	&H8&	在弹出窗口显示一个特定的帮助主题，该主题由.hpj 文件[MAP]节中定义的上下文编号来标识
CdlHelpContextPopup	&H9&	确保 WinHelp 显示正确的帮助文件。如果正确的帮助文件正被显示，则不出现任何动作。如果被显示的不是正确的帮助文件，则 Winhelp 将打开正确的文件
cdlHelpHelpOnHelp	&H4&	为使用帮助应用程序本身显示帮助
cdlHelpIndex	&H3&	显示指定的帮助文件的索引。应用程序应将该值用于只有一个索引的帮助文件
cdlHelpKey	&H101&	为特定的关键字显示帮助。当使用这个设置时，还必须用 HelpKey 属性指定一个关键字
cdlHelpPartialKey	&H105&	显示在关键字列表中能找到的一个与由 dwData 参数所传送的关键字精确匹配的主题。如果存在多个匹配的话，则显示带有已找到的在“转到”列表框中列出的哪些主题的“搜索”对话框。如果没有匹配的，则显示“搜

		索”对话框。为了显示没有传送关键字的“搜索”对话框，可使用一个指向空字符串的长指针
cdlHelpQuit	&H2&	通知帮助应用程序，所指定的帮助文件不再使用
cdlHelpSetContents	&H5&	当用户按 F1 键时确定显示哪个内容的主题
cdlHelpSetIndex	&H5&	将由 HelpContext 属性指定的上下文，设置为由 HelpFile 属性指定的帮助文件当前的索引。直到用户访问另一个帮助文件，该索引一直保持为当前。该值仅用于有多个索引的帮助文件

说明

用于 HelpCommand 属性常数的这些值在对象浏览器的 MicrosoftCommonDialog 控件(MSComDlg)对象库中列出。

cdlHelpContents 常数在用“MicrosoftHelpWorkshopVersion4.0X”创建的“帮助”文件中不起作用。你可以用值“&HB”得到同样的效果。请参阅“HelpCommand 属性示例”中一个可运行的代码示例。

数据类型

Integer

请参阅

HelpContext 属性(CommonDialog)，HelpKey 属性，HelpFile 属性(App，CommonDialog，MenuLine)

示例

下面示例示范了几个帮助命令。要试验该示例，在窗体中放一个 CommonDialog 控件、五个 CommandButton 控件，将代码粘贴到“声明”部分，按下 F5 键并单击每个按钮。

```
OptionExplicit
ConstHelpCNT=&HB

PrivateSubCommand1_Click()
   WithCommonDialog1
        '必须设置 Help 文件名。
        .HelpFile="VB5.hlp"
        '显示目录。注意 HelpCNT 常数不是一个内部常数。
        'cdlHelpSetContents 确保了只显示目录
        '(而不显示索引或查找)。
        .HelpCommand=HelpCNTOrcdlHelpSetContents
        .ShowHelp
   EndWith

EndSub
```

```
PrivateSubCommand2_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
        '进入“Help“文件的 ClickEvent 标题。
        '“.chm”文件的数字值是由“.HPJ”文件
        '的[MAP]部分决定的。只有当你用
        '“MicrosoftHelpWorkshop“建立
        '自己的帮助文件时，
        '才可以编辑此数字值。
        .HelpContext=916302
        .HelpCommand=cdlHelpContext
        .ShowHelp
    EndWith
EndSub

PrivateSubCommand3_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
        '显示关于帮助的帮助.
        .HelpCommand=cdlHelpHelpOnHelp
        .ShowHelp
    EndWith
```

```
EndSub

PrivateSubCommand4_Click()
   WithCommonDialog1
        .HelpFile="VB5.hlp"
        .HelpKey="data"
        '显示所选择关键字的索引。
        .HelpCommand=cdlHelpKey
        .ShowHelp
   EndWith
EndSub

PrivateSubCommand5_Click()
   WithCommonDialog1
        .HelpFile="VB5.hlp"
        .HelpKey="arrays,"
        '显示用“HelpKey“找到的标题列表。
        .HelpCommand=cdlHelpPartialKey
        .ShowHelp
   EndWith

EndSub
```

```
PrivateSubForm_Load()
    '标记“CommandButton”控件。
    Command1.Caption="Contents"
    Command2.Caption="SpecifiedTopic"
    Command3.Caption="HelpOnHelp"
    Command4.Caption="IndexofTopics"
    Command5.Caption="FoundTopics"
EndSub
```

HelpContext 属性

返回或设置一个字符串表达式，包含 MicrosoftWindows 帮助文件中的主题的上下文 ID。可读/可写。

应用于

Err 对象

说明

HelpContext 属性被用来自动显示 HelpFile 属性中指定的帮助主题。如果 HelpFile 和 HelpContext 都是空的，则检查 Number 的值。如果 Number 的值与 VisualBasic 运行时错误一致，则对此错误使用 VisualBasic 帮助上下文 ID。如果 Number 的值与 VisualBasic 错误不一致，则在屏幕上显示 VisualBasic 帮助文件的内容。

注意应该在应用程序中写入一些例程来处理常见错误。当使用对象编程时，可以用该对象的帮助文件来提高处理错误的质量，而

如果错误无法补救，则要为用户显示一段有意义的消息。

请参阅

Err 对象，Description 属性，HelpFile 属性，LastDLLError 属性，Number 属性，Source 属性

示例

本示例使用 Err 对象的 HelpContext 属性，来显示 VisualBasic 对于“溢出”(Overflow）错误的帮助主题。

```
DimMsg
Err.Clear
OnErrorResumeNext
Err.Raise6'引发“溢出”错误。
IfErr.Number<>0Then
   Msg="PressF1orHELPtosee"&Err.HelpFile&"topicfor"&_
   "thefollowingHelpContext:"&Err.HelpContext
   MsgBoxMsg,,"Error:"&Err.Description,Err.HelpFile,_
Err.HelpContext
EndIf
```

HelpContext 属性（“公共”对话框）

返回或设置请求的帮助主题的上下文 ID。

应用于

CommonDialog 控件

语法

object.**HelpContext**[=*value*]

HelpContext 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	字符串表达式，它指定请求的帮助主题的上下文 ID

说明

与 HelpCommand 属性一起使用该属性（设置 HelpCommand=cdlHelpContext）可指定要显示的帮助主题。

数据类型

Long

请参阅

HelpCommand 属性

示例

下面示例示范了几个帮助命令。要试验该示例，在窗体中放一个 CommonDialog 控件、五个 CommandButton 控件，将代码粘贴到“声明”部分，按下 F5 键并单击每个按钮。

```
OptionExplicit
ConstHelpCNT=&HB

PrivateSubCommand1_Click()
   WithCommonDialog1
```

```
        '必须设置 Help 文件名。
        .HelpFile="VB5.hlp"
        '显示目录。注意 HelpCNT 常数不是一个内部常数。
        'cdlHelpSetContents 确保了只显示目录
        '(而不显示索引或查找)。
        .HelpCommand=HelpCNTOrcdlHelpSetContents
        .ShowHelp
    EndWith

EndSub

PrivateSubCommand2_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
        '进入“Help“文件的 ClickEvent 标题。
        '“.chm”文件的数字值是由“.HPJ”文件
        '的[MAP]部分决定的。只有当你用
        '“MicrosoftHelpWorkshop“建立
        '自己的帮助文件时,
        '才可以编辑此数字值。
        .HelpContext=916302
        .HelpCommand=cdlHelpContext
        .ShowHelp
```

```
    EndWith
EndSub

PrivateSubCommand3_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
        '显示关于帮助的帮助.
        .HelpCommand=cdlHelpHelpOnHelp
        .ShowHelp
    EndWith
EndSub

PrivateSubCommand4_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
        .HelpKey="data"
        '显示所选择关键字的索引。
        .HelpCommand=cdlHelpKey
        .ShowHelp
    EndWith
EndSub
```

```
PrivateSubCommand5_Click()
    WithCommonDialog1
        .HelpFile="VB5.hlp"
        .HelpKey="arrays,"
        '显示用“HelpKey“找到的标题列表。
        .HelpCommand=cdlHelpPartialKey
        .ShowHelp
    EndWith

EndSub

PrivateSubForm_Load()
    '标记“CommandButton”控件。
    Command1.Caption="Contents"
    Command2.Caption="SpecifiedTopic"
    Command3.Caption="HelpOnHelp"
    Command4.Caption="IndexofTopics"
    Command5.Caption="FoundTopics"
EndSub
```

HelpContextID 属性

为一个对象返回或设置一个相关联上下文的编号。它用于为应用

程序提供上下文有关的帮助。

应用于

ADOData 控件，TreeView 控件，ImageCombo 控件，ListView 控件,Slider 控件,TabStrip 控件,DataTimePicker 控件,Animation 控件，FlatSCrollBar 控件，MonthView 控件，UpDown 控件，DataaRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，MSChart 对象，MSHFlexGrid 控件，MSFlexGrid 控件, SSTab 控件, RichTextBox 控件, Errol 对象(数据报表设计器)，PropertyPage 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HscrollBar，VscrollBar 控件，ListBox 控件，MDIForm 对象，Menu 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLEContainer 控件

语法

object.**HelpContextID**[=*number*]

HelpContextID 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象。如果 object 被删去，则与活动窗体模块相联系的窗体被认为是 object
number	用来指定与 *object* 相关联的“帮助”主题上下文编号的数值表达式

设置

number 的设置为：

设置	描述
0	（缺省）没有上下文编号被指定
>0	用来指定有效上下文编号的一个整数

说明

对于应用程序中对象上下文有关的帮助，当编译帮助文件时，必须对 object 和相关联的帮助主题赋予相同的上下文编号。

如果已经为应用程序建立了 MicrosoftWindows 操作系统环境的帮助文件并设置了应用程序的 HelpFile 属性，那么当用户按 F1 键时，VisualBasic 将自动地调用帮助，并查找被当前上下文编号所定义的主题。

当前上下文编号是拥有焦点的对象的 HelpContextID 的值。如果 HelpContextID 被设为 0，那么 VisualBasic 将在对象的容器 HelpContextID 中查找。若仍然为 0，那么将在那个对象的容器中

查找，如此继续下去。如果不能找到非 0 的当前上下文号，那么 F1 键被忽略。

对于一个 Menu 控件，HelpContextID 在运行时可正常地读/写。但是对于那些被 VisualBasic 的加载宏遗弃或提供的菜单项来说，HelpContextID 是只读的，比如在“外接程序”菜单中的“外接程序管理器”命令。

注意建立 Help 文件必需 MicrosoftWindowsHelpCompiler（在 VisualBasic 专业版中）。

请参阅

HelpFile 属性(App，CommonDialog，MenuLine)

示例

该例子使用 VisualBasic 帮助文件中的主题来演示如何为帮助主题指定上下文号。要试用此例，先将下面的代码粘贴到包含一个 TextBox 控件以及内有一个 OptionButton 控件的 Frame 控件的窗体的声明部分。然后按下 F5 键。一旦程序运行，就将焦点移到一个控件上，并按下 F1 键。

```
'VisualBasic 帮助文件中的真正的上下文号。
ConstwinColorPalette=21004 '定义常数。
ConstwinToolbox=21001
ConstwinCodeWindow=21005

PrivateSubForm_Load()
```

```
    App.HelpFile="VB.HLP"
    Frame1.HelpContextID=winColorPalette
    Text1.HelpContextID=winToolbox
    Form1.HelpContextID=winCodeWindow
EndSub
```

HelpContextID 属性（VBA 外接程序对象模型）

返回或设置一个 String，其内容为 MicrosoftWindows 联机帮助文件中一个主题的上下文 ID，此属性可读/写。

应用于

VBProject 对象

请参阅

HelpFile 属性(VBA 外接程序对象模型)

示例

下面示例使用 HelpContextID 属性返回工程所属的帮助文件的上下文标识符。

```
Debug.PrintApplication.VBE.VBProjects(1).HelpContextID
```

HelpFile 属性

返回或设置一个字符串表达式，表示帮助文件的完整限定路径。可读/可写。

应用于

Err 对象

说明

如果 HelpFile 中指定了帮助文件，则当用户在错误消息对话框中按下 Help 按钮（或 Windows 中的 F1 键或 Macintosh 中的 HELP 键)）时，帮助文件被自动调用。如果 HelpContext 属性包含被指定文件的有效的上下文 ID，则自动显示那一主题。如果未指定 HelpFile，则显示 VisualBasic 帮助文件。

注意应该在应用程序中写入一些例程来处理常见错误。当使用对象编程时，可以用该对象的帮助文件来提高处理错误的质量。如果错误无法补救，则要为用户显示一段有意义的消息。

请参阅

Err 对象，Description 属性，HelpContext 属性，LastDLLError 属性，Number 属性，Source 属性

示例

本示例使用 Err 对象的 HelpFile 属性来激活在线帮助系统。HelpFile 属性缺省值为 VisualBasic 帮助文件之文件名。

```
DimMsg
Err.Clear
OnErrorResumeNext  '由于示范目的，改变错误处理的方式。
Err.Raise6 '引发“溢出”错误。
Msg="PressF1orHELPtosee"&Err.HelpFile&_
```

```
"topicforthiserror"
MsgBoxMsg,,"Error:"&Err.Description,Err.HelpFile,Err.HelpContext
```

HelpFile 属性（VBA 外接程序对象模型）

返回或设置一个 String，指定一个工程的 MicrosoftWindows 联机帮助文件。可读写。

应用于

VBProject 对象

请参阅

HelpContextID 属性(VBA 外接程序对象模型)

示例

下列示例使用 HelpFile 属性指派一个帮助文件给工程；示例显示帮助文件的全路径名以确认这次赋值是否成功。

```
Application.VBE.VBProjects(1).HelpFile="C:\HelpStuff\veenob3.hlp
"
Debug.PrintApplication.VBE.VBProjects(1).HelpFile
```

HelpFile 属性(App、CommonDialog、MenuLine)

确定 MicrosoftWindowsHelp 文件的路径和文件名，应用程序使用这个文件显示 Help 或联机文档。

应用于

CommonDialog 控件(颜色对话框)，CommomDialog 控件(字体对话框)，CommonDialog 控件(Open，SaveAs 对话框)，CommonDialog 控件(打印对话框)，CommonDialog 控件(Help)，Error 对象(数据报表设计器)，App 对象

语法

object.**HelpFile**[=*filename*]

HelpFile 属性语法有以下部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
filename	字符串表达式，确定应用程序中 MicrosoftWindowsHelp 文件的路径和文件名

说明

如果为应用程序创建了一个 WindowsHelp 文件并设置了应用程序 HelpFile 属性，当按 F1 键时，VisualBasic 自动调用 Help。无论对活动的控件还是活动的窗体，如果在 HelpContextID 属性中有一个上下文号，则 Help 显示对应当前 Help 上下文的主题；否则显示主要目录屏幕。

当从对象浏览器请求 ActiveX 部件 Help 时，也可使用 HelpFile 属性决定显示哪一个 Help 文件。

注意构造 Windows 帮助文件必需 MicrosoftWindowsHelpCompiler，在 VisualBasic 专业版中可以得到它。

请参阅

HelpContextID 属性

示例

示例使用 VisualBasicHelp 文件中的主题并演示怎样确定 Help 主题的上下文号。要试用此例，将代码粘贴 Form 对象声明部分中，该对象包含 TextBox 控件和内含 OptionButton 控件的 Frame 控件。运行此例，当程序在运行时，移动焦点至其中之一部件，并按 F1 键。

```
'VisualBasicHelp 文件实际上下文号。
'定义常数。
ConstwinPictureBox=2016002
ConstwinCommandButton=2007557

PrivateSubForm_Load()
   App.HelpFile="VB98.CHM"
   Text1.HelpContextID=winPictureBox
   Form1.HelpContextID=winCommandButton
EndSub
```

HelpKey 属性

返回或设置标识请求的帮助主题的关键字。

应用于

CommonDialog 控件

语法

object.**HelpKey**[=*string*]

HelpKey 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
sting	字符串表达式，它指定标识的帮助主题的关键字

说明

与 HelpCommand 属性一起使用该属性（设置 HelpCommand=cdlHelpKey）可指定要显示的帮助主题。

数据类型

String

请参阅

HelpCommand 属性

Hex 函数

返回代表十六进制数值的 String。

语法

Hex(*number*)

必要的 number 参数为任何有效的数值表达式或字符串表达式。

说明

如果 number 还不是一个整数，那么在执行前会先被四舍五入成最接近的整

数。

如果 number 为	所得为
Null	Null
Empty	零(0)
任何其他的数字	最多可到八个十六进制字符

适当范围内的数字，前缀以&H，可以直接表示十六进制数字。例如，十六进制表示法的&H10 代表十进制的 16。

请参阅

Oct 函数

示例

本示例使用 Hex 函数来得到某数值的 16 进制值。

```
DimMyHex
MyHex=Hex(5)    '返回 5。
MyHex=Hex(10)   '返回 A。
MyHex=Hex(459) '返回 1CB。
```

Hidden 属性

返回某个 Member 对象的 Hidden 属性，或者对其进行设置。

应用于

SeriesPosition 对象

语法

object.**Hidden**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Hide 事件（UserControl 对象）

当对象的 Visible 属性变为 False 时，该事件产生。

应用于

UserControl 对象

语法

Sub*object*_Hide()

Hide 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

要在 Windows 中绘制屏幕，每个对象都必须有一个窗口，无论是临时窗口还是永久窗口；VisualBasic 的 ActiveX 控件具有永久的窗口。将控件定位在窗体上之前，控件的窗口不在容器中。删除窗口时，控件接收 Hide 事件。

控件窗口在窗体上的时候，控件的 Visible 属性变更为 False 时，对象接收 Hide 事件。

如果窗体隐藏后又显示出来，或者窗体最小化后又还原，那么控件不接收 Hide 事件。在进行这些操作期间，控件的窗口保留在窗体上，而且其 Visible 属性没有改变。

如果在 internet 浏览器中显示控件，则当该页移至历史列表中时产生 Hide 事件。

注意在早于 5.0 版的 VisualBasic 中使用控件时，控件在设计时不接收 Hide 事件。这时因为早期版本的 VisualBasic 在设计时不将任何可见的窗口放置到窗体上。

请参阅

Visible 属性

Hide 事件（UserDocument 对象）

当对象的 Visible 属性变更为 False 时，该事件产生。

应用于

UserDocument 对象

语法

Sub*object*_**Hide**()

Hide 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

要在 Windows 中绘制屏幕，每个对象都必须有一个窗口，无论是临时窗口还是永久窗口；VisualBasic 的 ActiveX 文档具有永久的窗口。删除窗口时，UserDocument 对象接收 Hide 事件。

当 object 的窗口位于容器上，object 的 Visible 属性变更为 False 时，

object 接收 Hide 事件。

如果容器隐藏后又显示出来，或者容器最小化后又还原，则 object 不接收 Hide 事件。在进行这些操作期间，object 的窗口仍保留在容器上，而且其 Visible 属性没有改变。

如果在 internet 浏览器中显示 object，则在下述两种情况下产生 Hide 事件：将页移至历史列表时（通过 object 定位到另一个文档），或者在结束 InternetExplorer3.0 时，object 正在被查阅，或 object 仍在活动文档的缓存中。在定位到另一个文档之前使用该事件，将破坏所有的全局对象引用。

注意在早于 5.0 版的 VisualBasic 中使用 object 时，object 在设计时不接收 Hide 事件。这时因为早期版本的 VisualBasic 在设计时不将任何可见的窗口放置到窗体上。

请参阅

Visible 属性

Hide 方法

用以隐藏 MDIForm 或 Form 对象，但不能使其卸载。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

object.**Hide**

object 所在处代表一个对象表达式，其值为“应用于”列表中的

一个对象。如果省略 object，则带有焦点的窗体就认为是该 object。

说明

隐藏窗体时，它就从屏幕上被删除，并将其 Visible 属性设置为 False。用户将无法访问隐藏窗体上的控件，但是对于运行中的 VisualBasic 应用程序，或对于通过 DDE 与该应用程序通讯的进程及对于 Timer 控件的事件，隐藏窗体的控件仍然是可用的。

窗体被隐藏时，用户只有等到被隐藏窗体的事件过程的全部代码执行完后才能够与该应用程序交互。

如果调用 Hide 方法时窗体还没有加载，那么 Hide 方法将加载该窗体但不显示它。

注意当关闭从另一个模式窗体打开的模式窗体时，以下代码在 VisualBasic 以前的版本中起作用：

```
Me.Hide
Me.Hide'这将会产生错误。
```

在 VisualBasic 当前版本中，该代码错在第二个 Me.Hide。可以用 Me.Visible=False 来代替 Me.Hide，如下所示：

```
Me.Visible=False
Me.Visible=False'没有错误发生。
```

请参阅

Load 语句，Unload 语句，Show 方法，Visible 属性

示例

本示例使用 Hide 方法来隐藏一个窗体。要检验此示例，可将本例代码粘贴到一个非 MDI 窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
    DimMsg                          '声明变量。
    Hide                        '隐藏窗体。
    Msg="ChooseOKtomaketheformreappear."
    MsgBoxMsg                       '显示信息。
    Show                        '使窗体重显。
EndSub
```

HideSelection 属性

返回一个值，以决定当控件失去焦点时选择文本是否加亮显示。

应用于

TextBox 控件

语法

object.**HideSelection**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

返回值

HideSelection 属性返回值为:

值	描述
true	（缺省值）当控件失去焦点时，选择文本不加亮显示
false	当控件失去焦点时，选择文本加亮显示

说明

该属性用于指示在另一个窗体或对话框拥有焦点时，哪些文本要加亮显示——例如，可用在拼写检查程序中。

示例

这个例子允许在每一个窗体中选择文本，并且可以通过单击窗体的标题栏在窗体之间进行切换。选定的部分即使在窗体不是活动时也保持可见。要试用此例，先创建两个窗体，并在每个窗体中画一 TextBox 控件。把两个 TextBox 控件的 MultiLine 属性设置为 True，并把其中一个 TextBox 控件的 HideSelection 属性设置为 False。把代码粘贴到两个窗体模块的声明部分，然后按 F5 键。

```
PrivateSubForm_Load()
   Open"README.TXT"ForInputAs1 '把文件加载到文本框.
   Text1.Text=Input$(LOF(1),1)
   Close1
   Form2.Visible=True  '若尚未加载，则加载 Form2。
   '使窗体并排.
   Form1.Move0,1050,Screen.Width/2,Screen.Height
   Form2.MoveScreen.Width/2,1050,Screen.Width/2,Screen.Height
   '放大文本框填充窗体.
   Text1.Move0,0,ScaleWidth,ScaleHeight
```

EndSub

hInstance 属性

返回一个应用程序实例的句柄。

应用于

App 对象

语法

object.**hInstance**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

hInstance 属性返回一个 Long 数据类型。

当在 VisualBasic 开发环境中使用工程进行工作时，hInstance 属性返回 VisualBasic 实例的实例句柄。

请参阅

hDC 属性，hWnd 属性

HitBehavior 属性

返回或设置一个值，它定义在 WindowlessUserControl 对象上 HitTest 事件的击点检测行为。

语法

object.**HitBehavior**[=*number*]

HitBehavior 属性的语法包含如下部分：

部分	描述
object	一个 UserControl 对象
number	一个整数，它指定击点检测行为，如在设置值中描述

设置值

对 number 的设置如下：

常数	设置值	描述
none	0	HitTest 事件总是返回值为 0 的 HitResult（vbHitResultOutside）。
useRegion	1	（缺省的）当光标位于控件的 MaskRegion 上时，HitTest 事件返回值为 3 的 HitResult（vbHitResultHit）。
usePaint	2	当光标位于控件的任何已绘图区域时，HitTest 事件将返回值为 3 的 HitResult（vbHitResultHit）。

说明

可以使用 HitBehavior 属性来决定点击将在 UserControl 的哪一位置发生。按照缺省设定，在 HitTest 事件中只有控件的 MaskRegion 才能返回点击。MaskRegion 由任何子控件加上任何由 MaskPicture 和 MaskColor 属性定义的掩码组成。MaskRegion 外的任何区域将返回值为 0 的 HitResult。

通过设置 HitBehavior 为 None，HitTest 事件将不再返回点击。当 HitBehavior 被设置为 UsePaint 时，屏蔽区域加上图形方法所绘图的区域将返回点击。

当与 ClipBehavior 属性结合使用时，这一属性有助于决定 HitTest 事件的 HitResult 参数。

注意如果 UserControl 对象的 Windowless 属性被设置为 False，或者 BackStyle 属性被设置为 Opaque，该属性将被忽略。

重点并非所有的控件容器都支持 Windowless 属性。仅当知道它将用在支持 Windowless 激活的容器中时，需要更改 HitBehavior 属性。

HitTest 事件

当用户将鼠标移动到一个 UserControl 对象上时发生。只在 UserControl 的 Windowless 属性被设置为 True，并且 BackStyle 属性被设置为 Transparent 时发生。

应用于

UserControl 对象

语法

Private Sub *object*_**HitTest**(*x* **As Single**, *y* **As Single**, *HitResult* **As Integer**)

HitTest 事件的语法包含如下部分：

部分	描述
object	一个 UserControl 对象
x,y	一个数值，指定鼠标指针当前位置。x 和 y 值总是用由对象的 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 属性设置的坐标系表示的
HitResult	一个整数，它用来指定点击行为，如在设置值中的描述

设置值

对 HitResult 的设置值如下：

常数	设置值	描述
vbHitResultOutside	0	鼠标指针位于控件的可视区域之外。鼠标消息被提前
vbHitResultTransparent	1	鼠标指针位于控件的透明区域上。鼠标消息可以被提前。必须在运行时设置
vbHitResultClose	2	鼠标指针接近控件的可视区域。接近区域必须由开发者在设计时定义。鼠标消息可以被提前。必须在运行时设置
vbHitResultHit	3	鼠标指针位于控件的可视区域上。控件接收鼠标消息

说明

HitTest 事件用于决定 UserControl 是否接收如下鼠标事件：MouseUp、MouseDown、MouseOver、Click 和 DblClick。HitTest 在任何鼠标消息之前发生。

按照缺省规定，当鼠标指针位于控件的透明区域时，UserControl 返回值为 0 的 HitResult，而当鼠标指针位于 MaskRegion 时返回值为 3 的 HitResult。MaskRegion 由 MaskPicture 和 MaskColor 属性共同定义。

在 HitTest 事件中，通过对 HitResult 指定一个不同的值，可以更改控件的点击检测行为。例如，如果有一个 UserControl，在它的中心有一块透明区域，在那一区域下面另一个控件就可能是可见的。通过在 HitTest 事件中把 HitResult 的值更改为 1 或 2，可以允许下面的控件接收到鼠标消息。如果运行时下面没有任何控件，或下面的控件拒绝点击，则该控件将获得第二或第三次机会接收鼠标消息。

对于多个重叠控件，点击检测将按照如下的顺序执行：

Zorder 中最顶层的、返回值为 3 的 HitResult (vbHitResultHit) 的控件将接收鼠标消息。

如果在 Zorder 中没有控件返回点击，返回值为 2 的 HitResult（vbHitResultClose）的最顶层控件将接收鼠标消息。

如果没有控件返回点击，返回值为 1 的 HitResult（vbHitResultTransparent）的最顶层控件将接收鼠标消息。

如果没有控件返回点击，鼠标消息将提前给下一级的容器。

只在 UserControl 的 Windowless 属性被设置为 True，并且 BackStyle 属性被设置为 Transparent 时，HitTest 事件才会发生。如果 Windowless 属性为 False 或者是 BackStyle 属性等于 Opaque，HitTest 事件过程的任何代码都将被忽略。

注意以上描述的点击检测顺序适用于 VisualBasic 容器中的 UserControl 对象。其它容器可能不是按照相同的顺序执行点击检测。

HostName 属性

返回或设置用户可读的、VisualBasic 应用程序的主机名称。

应用于

OLE 包容器控件

语法

object.**HostName**[=*name*]

HostName 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
name	一个指定主机名称的字符串表达式

说明

当编辑对象时，HostName 属性的设置也许在对象窗口的标题中显示。不过某些提供对象的应用程序不显示 HostName。

Hour 函数

返回一个 Variant(Integer)，其值为 0 到 23 之间的整数，表示一天之中的某一钟点。

语法

Hour(*time*)

必要的 time 参数，可以是任何能够表示时刻的 Variant、数值表达式、字符串表达式或它们的组合。如果 time 包含 Null，则返回 Null。

请参阅

Day 函数，Minute 函数，Now 函数，Second 函数，Time 函数，Time 语句

示例

本示例使用 Hour 函数将指定的时间转换为小时数。在开发环境中，日期和时间的原义会根据系统的地区设置，以短式日期和时间格式显示。

```
DimMyTime,MyHour
MyTime=#4:35:17PM# '指定一时间。
MyHour=Hour(MyTime)      'MyHour 的值为 16。
```

hPal 属性

返回或设置句柄，指向 Pictrue 对象中图片的调色板。

应用于

Picture 对象

语法

object.**hPal**[=*value*]

hPal 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	指向图片调色板的句柄(HPAL)

说明

作为调用动态链接库(DLL)或 WindowsAPI 函数的一个部分需传递调色板的句柄时，hPal 属性是有用的。

HScrollBar、VScrollBar 控件

在项目列表很长或者信息量很大时，可以使用滚动条来提供简便的定位。它还可以模拟当前所在的位置。滚动条可以作为输入设备，或者速度、数量的指示器来使用——例如，可以用它来控制计算机游戏的音量，或者查看定时处理中已用的时间。

语法

HScrollBar

VScrollBar

说明

使用滚动条作为数量或速度的指示器、或者作为输入设备时，可

以利用 Max 和 Min 属性设置控件的适当变化范围。

为了指定滚动条内所示变化量，在单击滚动条时要使用 LargeChange 属性，在单击滚动条两端的箭头时，要使用 SmallChange 属性。滚动条的 Value 属性或递增或递减，增减的量是通过 LargeChange 和 SmallChange 属性设置的值。在运行时，在 0 和 32,767 之间设置 Value 的值，就可以将滚动框定位。

属性

HscrollBar，RightToLeft 属性，Height，Width 属性，Left，Top 属性，TabIndex 属性，Tag 属性，Value 属性，Visible 属性，DragIcon 属性，DragMode 属性，hWnd 属性，LargeChange，SmallChange 属性，Max，Min 属性(滚动条)，MouseIcon 属性，MousePointer 属性，TabStop 属性，Enabled 属性，HelpContextID 属性，Index 属性(控件数组)，Name 属性，Parent 属性，Containet 属性，WhatsThisHelpID 属性

方法

Refresh 方法，SetFocus 方法，Drag 方法，Move 方法，Zorder 方法，ShowWhatsThis 方法

请参阅

Value 属性，LargeChange，SmallChange 属性，Max，Min 属性(滚动条)

HScrollSmallChange 属性和 VScrollSmallChange 属性

当单击某个滚动箭头时返回 UserDocument 将滚动的距离，或者对其进行设置。

应用于

UserDocument 对象

语法

object.**HScrollSmallChange**=*single*

object.**VScrollSmallChange**=*single*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
single	当单击滚动箭头时，UserDocument 用缇表示的滚动的距离

说明

并不存在跟 HScrollSmallChange 和 VScrollSmallChange 属性相对应的任何“大幅调整”属性。“大幅调整”是由 ViewPort 对象的 ViewPortHeight 属性和 ViewPortWidth 属性所决定的。

hWnd 属性

返回窗体或控件的句柄。

注意 OLE 容器控件不支持该属性。

应用于

ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，ToolbarControl，Animation 控件，UpDown 控件，DBCombo 控件，DBList 控件，SSTab 控件，PicturtClip 控件，RichTextBox 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，Dir ListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HscrollBar，VScrollBar 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件，OLEContainer 控件

语法

object.**hWnd**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

MicrosoftWindows 运行环境，通过给应用程序中的每个窗体和控件分配一个句柄（或 hWnd）来标识它们。hWnd 属性用于 WindowsAPI 调用。许多 Windows 运行环境函数需要活动窗口的 hWnd 作为参数。

由于该属性值在程序运行时可以改变，绝对不要将 hWnd 存储在变量中。

如果 UserControl 的 Windowless 属性设为 True，则 hWnd 属性将返回 0。

请参阅

Icon 属性，hDC 属性

示例

这个例子强制窗体保持在最前面。要试用此例，先创建一个窗体（不是一个 MDI 子窗体），然后为该窗体创建一菜单叫做 Main。在其中插入一子菜单叫做 AlwaysOnTop，并且把它的名字设置为 mnuTopmost。利用“工程”菜单中的“添加模块”命令创建一个新模块。把 Declare 语句粘贴到新模块的声明部分，确保该语句在一行，并且没有断点或隐藏字。然后把 Sub 过程粘贴到窗体的声明部分并按 F5 键。

```
'声明Windows 例程。
'该语句应在模块中。
DeclareFunctionSetWindowPosLib"user32"Alias_
"SetWindowPos"(ByValhwndAsLong,ByVal_
hWndInsertAfterAsLong,ByValxAsLong,ByValyAs_
Long,ByValcxAsLong,ByValcyAsLong,ByValwFlags_
AsLong)AsLong

'设置一些常数值（从WIN32API.TXT）.
ConstconHwndTopmost=-1
ConstconHwndNoTopmost=-2
ConstconSwpNoActivate=&H10
ConstconSwpShowWindow=&H40
```

```
PrivateSubmnuTopmost_Click()
    '从菜单中加入或删除检查标志.
    mnuTopmost.Checked=NotmnuTopmost.Checked
    IfmnuTopmost.CheckedThen
        '打开 TopMost 属性.
        SetWindowPoshWnd,conHwndTopmost,0,0,0,0,_
conSwpNoActivateOrconSwpShowWindow
    Else
        '关闭 TopMost 属性.
        SetWindowPoshWnd,conHwndNoTopmost,0,0,0,0,_
conSwpNoActivateOrconSwpShowWindow
    EndIf
EndSub
```

无论何时只要 ComboBox 收到焦点，在这个例子中都会自动放下 ComboBox 控件的列表部分。要试用此例，创建一个包含 ComboBox 控件和 OptionButton 控件的新窗体（只为获得焦点用）。用先获得“工程”菜单中的“添加模块”命令创建一个新的模块。把 Declare 语句粘贴到新模块的声明部分，确保该语句在一行，并且没有断点或自动换行。然后把 Sub 过程粘贴到窗体的声明部分，并按 F5 键。用 TAB 键将焦点移动到 ComboBox 以及从 ComboBox 移出。

```
DeclareFunctionSendMessageLib"user32"Alias"SendMessageA"_
(ByValhwndAsLong,ByValwMsgAsLong,ByValwParamAsLong,_
lParamAsLong)AsLong

PrivateSubCombo1_GotFocus()
    ConstCB_SHOWDROPDOWN=&H14F
    DimTmp
    Tmp=SendMessage(Combo1.hWnd,CB_SHOWDROPDOWN,1,ByVal0&)
EndSub
```

Hyperlink 对象

使用 Hyperlink 对象的属性和方法，ActiveX 文档或者 ActiveX 控件就可以请求一个超链接意识的容器，比如 Microsoft Internet Explorer，以跳到给定的 URL。

说明

可以用 NavigateTo 方法跳到 URL。例如，下列代码假定存在一个名为"axdMyDoc"的 ActiveX 文档：

```
UserDocument.Hyperlink.NavigateTo_
"c:\mydocs\axdmydoc.vbd"
```

如果 ActiveX 文档被包含在一个超链接意识的容器（比如 InternetExplorer）中，而容器又维护文档历史，则可用 GoBack

或 GoForward 方法在整个列表中来回移动。但是，一定要确保使用错误检查，如下例所示：

```
PrivateSubcmdGoForward_Click()
   OnErrorGoTonoDocInHistory
   UserDocument.Hyperlink.GoForward
   ExitSub
noDocInHistory:
   ResumeNext
EndSub
```

方法

GoBack 方法，GoForward 方法，NavigageTo 方法

请参阅

UserDocument 对象

Hyperlink 属性

返回对 Hyperlink 对象的引用。

应用于

UserControl 对象，UserDocument 对象

语法

object.**Hyperlink**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

请参阅

Hyperlink 对象

Icon 属性

返回在运行时窗体处于最小化时显示的图标。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

object.**Icon**

object 所在处表示对象表达式，其值为“应用于”列表中的一个对象。

说明

使用该属性在运行时用户可以码最小化的窗体指定图标。

例如，可以为窗体分配唯一的图标以标识该窗体的功能。通过在设计时使用属性窗口加载而指定图标。所加载的文件必须有.ico 文件扩展名和格式。如果不指定图标，窗体会使用 VisualBasic 缺省图标。

可以使用 VisualBasic 的图标库（在图标子目录下）作为图标源。

在创建可执行文件时，使用应用程序中的任一窗体的 Icon 属性可以给这个应用程序分配一个图标。

注意在 Windows95 中可以在窗体的左上角看到窗体的图标，或者在 Windows95 和 WindowsNT 中当窗体最小化时可看到图标。当窗体最小化时，BorderStyle 属性值必须设置为 1（固定单边框）或 2

（可变尺寸）并且 MinButton 属性必须设置为 True 以使图标可见。在运行时，可以将一对象的 Icon 属性赋值给另一对象的 DragIcon 或 Icon 属性。还可以分配 LoadPicture 函数的返回的图标，用无参数的 LoadPicture 给窗体分配一个空(null)图标，从而在运行时可在该图标上绘图。

请参阅

LoadPicture 函数

示例

这个例子在窗体被最小化时为窗体创建一个空白的图标并在图标上面绘制彩色的点。要尝试这个例子，请将代码粘贴到窗体的声明部分，然后按 F5 键并将窗体最小化。

注意这个例子只能在 WindowsNT3.5x 下运行。

```
PrivateSubForm_Resize()
   DimX,Y                                  '声明变量。
   IfForm1.WindowState=vbMinimizedThen
       Form1.Icon=LoadPicture()            '加载一个空白的图标。
       DoWhileForm1.WindowState=vbMinimized
'窗体最小化。
           Form1.DrawWidth=10              '设置点的大小。
           '随机选择点的颜色。
           Form1.ForeColor=QBColor(Int(Rnd*15))
           '随机设置图标的位置。
```

```
            X=Form1.Width*Rnd
            Y=Form1.Height*Rnd
            PSet(X,Y)                    '在图标上绘制点。
            DoEvents                  '允许处理其它事件。
        Loop
    EndIf
EndSub
```

这是同样的例子，除了用 LoadPicture 方法来设置 Icon 属性以外。这个例子可用在所有版本的 Windows 之上。

```
PrivateSubForm_Resize()
    DimX,Y'声明变量。
    IfForm1.WindowState=vbMinimizedThen
        Form1.Icon=LoadPicture("c:\myicon.ico")
'"myicon.ico"图标必须在
'c:\目录中，这个例子才能正常运行。
    EndIf
EndSub
```

IconState 属性

返回或设置工程窗口中该工程的源代码控件图标（或"glyph"），指示其状态。

应用于

VBComponent 对象，VBProject 对象，VBProject 集合

语法

object.**IconState**[=*value*]

IconState 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	决定文件状态的长整数或常数，如“设置值”中所描述

设置值

value **的设置值是：**

常数	值	描述
vbextSCCStatusNotControlled	0	文件不在源代码控件之下
vbextSCCStatusControlled	1	文件在源代码控件之下
vbextSCCStatusCheckedOut	2	文件签出给当前用户
vbextSCCStatusOutOther	4	文件签出给其它用户
vbextSCCStatusOutOfDate	32	文件不是最新的
vbextSCCStatusShared	512	.文件在工程之间共享

说明

IconState **属性可通过逻辑** OR **来形成组合的状态。**

请参阅

Reload 方法，FileCount 属性，FileNames 属性

ID 属性

设置 HTML 页面上一个元素或 DHTMLPageDesigner 对象的名字。运行时不可用。

应用于

DHTMLPageDesigner 对象

说明

该属性允许为每一 DHTMLPageDesigner 对象或选中的元素指定一个标识符。为在 HTML 元素后面放置代码，必须为它指定一个标识符。

除非元素有一个标识符，否则它不会在“代码窗口”窗口的“对象窗口”下拉列表中出现。设计器窗口中的树形视图使用粗体显示有标识符的元素。

IDTExtensibility 接口

IDTExtensibility 接口包含了当外接程序与 VisualBasic 连接时 VisualBasic 调用的一些方法，无论是通过外接程序管理器，还是其它一些手段。

IDTExtensibility 接口包括一些管理 VisualBasic 中各个外接程序所需的预配置过程模板（其中包括它们的参数列表）。

应用于

OnAddinsUpdate 方法，OnConnection 方法，OnDisconnection

方法，OnStartupComplete 方法

语法

ImplementsIDTExtensiblity

说明

接口的用法在 VisualBasic5.0 中介绍过。接口能够从模块的“过程”下拉列表框中选择预配置的过程模板，消除参数表输入错误并能加快编程速度。

接口的方法是通过 Implements 语句展现的。当上面的语法输入处理外接程序事件的类模块的声明部分时，通过模块的“过程”和“对象”下拉框便可使用该接口的方法。要在模块中添加代码，只要在下拉框中简单地选定它就行。

IDTExtensiblity 接口通常包含四种方法：

OnAddinsUpdate 方法

OnConnection 方法

OnDisconnection 方法

OnStartupComplete 方法

虽然，它们的作用和表现很像事件，但对 VisualBasic 的程序员来讲，这些仍然是 IDTExtensibility 接口的方法。换言之，当外接程序连接到 VisualBasic 时，OnConnection 方法被自动调用，类似于事件启动。当连接撤销时，OnDisconnection 方法被自动调用，以此类推。

重点由于接口是对象和 VisualBasic 之间的约定，必须确保实现

接口中的所有方法。这就意味着全部四种 IDTExtensibility 接口方法必须在类模块中提供，每一种至少包含一个可执行语句。它能小到只有一个说明语句，但是它们必须至少包含一个可执行语句，以免编译程序把它们作为空过程删除。

方法

OnAddinsUpdate 方法，OnConnection 方法，OnDisconnection 方法，OnStartupComplete 方法

请参阅

OnAddinsUpdate 方法，OnConnection 方法，OnDisconnection 方法，OnStartupComplete 方法

If...Then...Else 语句

根据表达式的值有条件地执行一组语句。

语法

If*condition***Then** [*statements*] [**Else***elsestatements*]

或者，可以使用块形式的语法：

If*condition***Then**

[*statements*]

[**ElseIf***condition-n***Then**

[*elseifstatements*]...

[**Else**

[*elsestatements*]]

EndIf

If...Then...Else **语句的语法具有以下几个部分：**

部分	**描述**
Condition	**必要参数。一个或多个具有下面两种类型的表达式：数值表达式或字符串表达式，其运算结果为** True **或** False**。如果** condition **为** Null**，则** condition **会视为** False**。** TypeOfobject nameIsobjecttype **形式的表达式。其中的** Objectname **是任何对象的引用，而** objecttype **则是任何有效的对象类型。如果** objectname **是** objecttype **所指定的一种对象类型，则表达式为** True**，否则为** False
statements	**在块形式中是可选参数；但是在单行形式中，且没有** Else **子句时，则为必要参数。一条或多条以冒号分开的语句，它们在** *condition* **为** True **时执行**
condition-n	**可选参数。与** *condition* **同**
elseifstatements	**可选参数。一条或多条语句，它们在相关的** *condition-n* **为** True **时执行**
elsestatements	**可选参数。一条或多条语句，它们在前面的** condition **或** *condition-n* **都不为** True **时执行**

说明

可以使用单行形式（第一种语法）来做短小简单的测试。但是，块形式（第二种语法）则提供了更强的结构化与适应性，并且通

常也是比较容易阅读、维护及调试的。

注意在单行形式中，按照 If...Then 判断的结果也可以执行多条语句。所有语句必须在同一行上并且以冒号分开，如下面语句所示：

```
IfA>10ThenA=A+1:B=B+A:C=C+B
```

在块形式中，If 语句必须是第一行语句，其中的 Else、ElseIf，和 EndIf 部分可以只在之前加上行号或行标签。If 块必须以一个 EndIf 语句结束。

要决定某个语句是否为一个 If 块，可检查 Then 关键字之后是什么。如果在 Then 同一行之后，还有其它非注释的内容，则此语句就是单行形式的 If 语句。

Else 和 ElseIf 子句都是可选的。在 If 块中，可以放置任意多个 ElseIf 子句，但是都必须在 Else 子句之前。If 块也可以是嵌套的。

当程序运行到一个 If 块（第二种语法）时，condition 将被测试。如果 condition 为 True，则在 Then 之后的语句会被执行。如果 condition 为 False，则每个 ElseIf 部分的条件式（如果有的话）会依次计算并加以测试。如果找到某个为 True 的条件时，则其紧接在相关的 Then 之后的语句会被执行。如果没有一个 ElseIf 条件式为 True（或是根本就没有 ElseIf 子句），则程序会执行 Else 部分的语句。而在执行完 Then 或 Else 之后的语句后，会从 EndIf 之后的语句继续执行。

提示根据单一表达式来执行多种可能的动作时，SelectCase 更为有用。不过，TypeOfobjectnameIsobjecttype 子句不能在 SelectCase 语句中使用。

注意 TypeOf 不能与诸如 Long、Integer 以及其他不是 Object 的固定数据类型一起使用。

请参阅

#If…Then…#Else 语句，Choose 函数，SelectCase 语句，Switch 函数

示例

本示例示范 If...Then...Else 语句的两种格式：“区块格式”及“单行格式”，也示范了 IfTypeOf...Then...Else 的用法。

```
DimNumber,Digits,MyString
Number=53 '设置变量初始值。
IfNumber<10Then
   Digits=1
ElseIfNumber<100Then
'若判断结果为 True，则完成下一行语句。
   Digits=2
Else
   Digits=3
EndIf

'使用“单行格式”语法来设置变量值。
IfDigits=1ThenMyString="One"ElseMyString="Morethanone"
```

利用 IfTypeOf 可以判断传入过程的控件是否为一文本框。

```
SubControlProcessor(MyControlAsControl)
    IfTypeOfMyControlIsCommandButtonThen
        Debug.Print"Youpassedina"&TypeName(MyControl)
    ElseIfTypeOfMyControlIsCheckBoxThen
        Debug.Print"Youpassedina"&TypeName(MyControl)
    ElseIfTypeOfMyControlIsTextBoxThen
        Debug.Print"Youpassedina"&TypeName(MyControl)
    EndIf
EndSub
```

IIf 函数

根据表达式的值，来返回两部分中的其中一个。

语法

IIf(*expr*,*truepart*,*falsepart*)

IIf 函数的语法含有下面这些命名参数：

部分	描述
expr	必需的。用来判断真伪的表达式
truepart	必需的。如果 *expr* 为 True，则返回这部分的值或表达式
falsepart	必需的。如果 *expr* 为 False，则返回这部分的值或表达式

说明

由于 IIf 会计算 truepart 和 falsepart，虽然它只返回其中的一个，因此要注意到这个副作用。例如，如果 falsepart 产生一个被零除错误，那么程序就会发生错误，即使 expr 为 True。

请参阅

Choose 函数，If…Then…Else 语句，SelectCase 语句，Switch 函数

示例

本示例使用 IIf 函数来判断 CheckIt 过程之 TestMe 参数的值，如果参数值大于 1000 则传回“Large”；否则传回“Small”。

```
FunctionCheckIt(TestMeAsInteger)
   CheckIt=IIf(TestMe>1000,"Large","Small")
EndFunction
```

Image 控件

Image 控件用来显示图形。Image 控件可以显示来自位图、图标或元文件的图形，也可以显示增强的元文件、JPEG 或 GIF 文件。

语法

Image

说明

因为 Image 控件使用较少的系统资源，所以重画起来比 PictureBox 控件要快，但是它只支持 PictureBox 控件的一部分属性、事件和

方法。用 Stretch 属性确定是否缩放图形来适应控件大小，反之亦然。虽然可以把 Image 控件放在容器里，但是 Image 控件不能作为容器。

注意 UnisysCorporation 有一项专利,该专利声称涉及到 GIF-LZW 压缩技术的某些方面，在该技术中使用了 PictureBox 和 Image 控件。MicrosoftCorporation 于 1996 年 9 月获得了对 UnisysLZW 专利的使用许可。然而，Microsoft 的许可证并不延伸到那些软件开发商或第三方，他们使用任何 Microsoft 工具包、语言开发或操作系统产品来在他们自己的产品中提供 GIF 读/写和/或任何其他 LZW 能力（例如，通过 DLL 和 API）。

如果您的商业应用程序使用了这些控件之一（并且因此使用了 LZW 技术），您可能会希望获得有关专利的独立的法律意见，详细信息请与 http://www.unisys.com/的 UnisysUSA 联系。

属性

DataMember 属性，DataFormat 属性，OLEDragMode 属性，OLEDropMode 属性,BorderStyle 属性,Height,Width 属性,Left,Top 属性，Picture 属性，Tag 属性，Visible 属性，DragIcon 属性，DragMode 属性，MouseIcon 属性，MousePointer 属性，Stretch 属性，Appearance 属性，Enabled 属性，Index 属性(控件数组) , Name 属性, Parent 属性, Containet 属性, ToolTipText 属性，WhatsThisHelpID 属性，DataChanged 属性，DataField 属性，DataSource 属性

方法

Refresh 方法，Drag 方法，Move 方法，Zorder 方法，OLEDrag 方法，ShowWhatsThis 方法

事件

Click 事件，DragDrop 事件，DragOver 事件，Mousedown，MouseUp 事件，MouseMove 事件，DbleClick 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetdata 事件，OLEStartDrag 事件

请参阅

ImageList 属性（ActiveX 控件），PictureBox 控件，Stretch 属性

Image 控件（数据报表设计器）

Image 控件可以显示来自一个位图、图标或图元文件，以及增强的图元文件、.jpeg 或.gif 文件的图形。

语法

RptImage

说明

Image 控件的数据报表设计器版可以在任何应用程序上显示图像，类似于标准的 VisualBasic 固有 Image 控件。然而除此基本功能之外，某些标准 Image 控件的属性在数据报表版中是不可用的。

属性

SizeMode 属性(RptImage 控件)，PictureAlignment 属性，BackColor，ForeColor 属性，BackStyle 属性，BorderColor 属性，BorderStyle 属性，Height，Width 属性，Left，Top 属性，Picture 属性，Visible 属性，Name 属性

Image 属性

返回持久图形的句柄，该句柄由 MicrosoftWindows 运行环境提供。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Form 对象，Forms 集合，PictureBox 控件

语法

object.**Image**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

对象的 AutoRedraw 属性决定是否用持久图形或通过 Paint 事件重绘对象。Windows 运行环境通过给对象的持久图形分配一个句柄来标识它；用 Image 属性可以得到该句柄。

Image 值的存在不受 AutoRedraw 属性设置值的影响。如果 AutoRedraw 为 True，并且还没有绘任何内容，图像仅显示由 BackColor 属性和图片确定的颜色。

可以给 Picture 属性分配 Image 的值。Image 属性还提供了一个传递给 WindowsAPI 调用的值。

Image、DragIcon 和 Picture 属性，通常用在给其它属性分配值的情况，如用 SavePicture 语句保存，或在剪贴板上放置一些内容。除图像数据类型外，不能把它们赋给临时变量。

AutoRedraw 属性可以引起 Image 改变，Image 是指向位图的句柄。当 AutoRedraw 为 True 时，对象的 hDC 属性成为指向设备描述体的句柄，该设备描述体包含 Image 返回的位图。

请参阅

Paint 事件，SavePicture 语句，BackColor，ForeColor 属性，Picture 属性，AutoRedraw 属性，DragIcon 属性，BackColor，ForeColor 属性(ActiveX 控件)，Picture 属性(ActiveX 控件)

示例

在这个例子中，每当单击第一个 PictureBox 控件时，都会在其中画一个圆。当单击第二个 PictureBox 时，图形从第一个 PictureBox 拷贝到其中。要试用此例，n 把代码粘贴到包含两个尺寸偏大、大小相等的 PictureBox 控件的窗体的声明部分。按 F5 键运行该程序，然后单击 PictureBox 控件。

```
PrivateSubForm_Load()
   '设置 AutoRedraw 为 True.
   Picture1.AutoReDraw=True
EndSub
```

```
PrivateSubPicture1_Click()
'声明变量.
   DimPW,PH
'设置 FillStyle 为 solid.
   Picture1.FillStyle=vbFSSolid
'选择随机颜色.
   Picture1.FillColor=QBColor(Int(Rnd*15))
   PW=Picture1.ScaleWidth                    '设置 ScaleWidth.
   PH=Picture1.ScaleHeight                   '设置 ScaleHeight.
   '在随机的位置画一个圆。
   Picture1.Circle(Int(Rnd*PW),Int(Rnd*PH)),250
EndSub

PrivateSubPicture2_Click()
'拷贝 Image 到 Picture2。
   Picture2.Picture=Picture1.Image
EndSub
```

Image 属性（ActiveX 控件）

返回或设置一个值，指定 ImageList **控件中哪个** ListImage **对象与另一对象一起使用。**

应用于

Node 对象，Nodes 集合，Tab 对象，Button 对象

语法

object.**Image**[=*index*]

Image 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的一个对象
index	整数或唯一的字符串，指定与 object 一起使用的 ListImage 对象。Index 属性值是整数，Key 属性值是字符串

说明

在设置 Image 属性之前，必须将 ImageList 属性与 Toolbar、TreeView 或 TabStrip 控件相关联，方法是设置每个控件的 ImageList 属性为 ImageList 控件。

在设计时，将 ImageList 控件放于窗体并加载图像，每个图像是在 ListImages 集合中被赋值索引号的一个 ListImage 对象。在该控件的“属性页”对话框中的“通用”选项卡上，从 ImageList 列表框中选择所要的 ImageList，如 ImageList1。对于 Tab 和 Button 对象，亦可在‘Tabs’或‘按钮’选项卡的图像字段中键入指定的 ListImage 对象的索引号，以指定要同这些对象关联的图像。

在运行时，使用下列代码将 ImageList 与控件相关联，并将 ListImage 与指定的对象相关联。

```
SetTabStrip1.ImageList=ImageList1
```

```
TabStrip1.Tabs(1).Image=2
```

希望自己的代码成为自编文件，可用 Key 属性指定 ImageList 控件的 ListImage 对象，如下所示：

```
'假设存在带有 Key 属性值的 ListImage 对象，其属性值＝
'"close,"，工具栏按钮使用该图像。
Toolbar1.Buttons(1).Image="close"

'这比仅指定一个索引值更易读，如下所示：
Toolbar1.Buttons(1).Image=4
'要求 Index 属性=4 的 ListImage 对象作为"close"图像。
```

当集合中的对象被重新排序时，如在设置 Sorted 属性为 True 时，对象的 Index 属性值会改变。如果想要 Index 属性动态地改变，那么应该通过使用 Key 属性引用集合中的对象。

如果 Tabs 集合没有图像，则 index 值是-1。

请参阅

ListImage 对象，ListImages 集合，Add 方法(ListImages 集合)

ImageList 属性（ActiveX 控件）

返回或设置与另一控件相关的 ImageList 控件。

应用于

TreeVies 控件，ImageConmbo 控件，TabStrip 控件，ToolBar 控件，CoolBar 控件

语法

object.**ImageList**[=*imagelist*]

ImageList 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
imagelist	对象引用，指定使用哪个 ImageList 控件

说明

控件要使用 ImageList 属性，必须先将 ImageList 控件放在窗体上。然后，在设计时，可在相关控件的‘属性页’对话框中设置 ImageList 属性。为了将 ImageList 在运行时与控件相关联，可设置控件的 ImageList 属性为要用的 ImageList 控件，如此例所示：

```
SetTabStrip1.ImageList=ImageList1
```

请参阅

ImageList 控件

IMEStatus 函数

返回一个 Integer，用来指定当前 MicrosoftWindows 的输入法(IME)方式；只对东亚区版本有效。

语法

IMEStatus

返回值

下面是日文地区的返回值：

常数	值	描述
vbIMENoOP	0	**不安装** IME（**缺省**）
vbIMEOn	1	**打开** IME
vbIMEOff	2	**关闭** IME
vbIMEDisable	3	IME **无效**
vbIMEHiragana	4	Hiragana **双字节字符**（DBC）
vbIMEKatakanaDbe	5	Katakana **双字节字符**（DBC）
vbIMEKatakanaSng	6	Katakana **单字节字符**（SBC）
vbIMEAlphaDbe	7	Alphanumeric **双字节字符**（DBC）
vbIMEAlphaSng	8	Alphanumeric **单字节字符**（SBC）

下面是中文地区的返回值：

常数	值	描述
vbIMENoOP	0	**不安装** IME（**缺省**）
vbIMEOn	1	**打开** IME
vbIMEOff	2	**关闭** IME

下面是朝鲜文地区的返回值：

字位（bit）	值	描述	值描述
0	0	不安装 IME （缺省）	1 安装 IME
1	0	禁用 IME	1 允许使用 IME
2	0	IMO 英语方式	1Hangeill 方式
3	0	Banja 方式（单字节）	1Junia 方式（双字节）
4	0	一般方式	1Hanja 会话方式

Implements 语句

指定要在包含该语句的类模块中实现的接口或类。

语法

Implements [*InterfaceName* | *Class*]

所需的 InterfaceName 或 Class 是类型库中的接口或类的名称，该类型库中的方法将用与 VisualBasic 类中相一致的方法来实现。

说明

所谓接口就是代表接口封装的成员（方法以及属性）的原型集合；也就是说，它只包含成员过程的声明部分。一个类提供一个或多个接口的所有方法以及属性的一种实现方案。类的控制者每次调用函数时，该函数所执行的代码由类来提供。每个类至少应实现

一个缺省接口。在 VisualBasic 中，一个已实现的接口中任何没有显式声明的成员都是缺省接口的隐式成员。

当 VisualBasic 类实现接口时，都会提供该接口的类型库中说明的所有 Public 过程的版本。除了提供接口原型与自编过程之间的映射关系之外，Implements 语句还使这个类接收对指定接口 ID 的 COMQueryInterface 调用。

注意 VisualBasic 不能实现派生出来的类或接口。

在实现接口或类时，必须包括所用到的 Public 过程。如果在实现接口或类时遗漏了成员，就会产生错误。如果正在实现的类中某个过程还没有代码，则可以产生一个适当的错误信息(ConstE_NOTIMPL=&H80004001)，以便用户意识到该成员还没有实现。

Implements 语句不能在标准模块中使用。

示例

下面的示例说明如何使用 Implements 语句来编写多个类都可以使用的一系列声明。通过 Implements 语句共享这些声明，所有的类都不必再自己进行声明。

假设有两个窗体。其中 Selector 窗体有两个按钮，CustomerData 和 SupplierData。若要输入客户或供应商的名称及地址信息，用户就单击 Selector 窗体的 Customer 按钮或 Supplier 按钮，然后使用 DataEntry 窗体来输入名称和地址。DataEntry 窗体有两个文本字段，Name 和 Address。

下面的共享声明的代码位于称为 PersonalData 的类中：

```
PublicNameAsString
PublicAddressAsString
```

有关处理客户数据的代码位于 Customer 类模块：

```
ImplementsPersonalData
PrivatePropertyGetPersonalData_Address()AsString
PersonalData_Address="CustomerAddress"
EndProperty

PrivatePropertyLetPersonalData_Address(ByValRHSAsString)
'
EndProperty

PrivatePropertyLetPersonalData_Name(ByValRHSAsString)
'
EndProperty

PrivatePropertyGetPersonalData_Name()AsString
PersonalData_Name="CustomerName"
EndProperty
```

有关处理供应商数据的代码位于 Supplier 类模块：

```
ImplementsPersonalData

PrivatePropertyGetPersonalData_Address()AsString
PersonalData_Address="SupplierAddress"
EndProperty

PrivatePropertyLetPersonalData_Address(ByValRHSAsString)
'
EndProperty

PrivatePropertyLetPersonalData_Name(ByValRHSAsString)
'
EndProperty

PrivatePropertyGetPersonalData_Name()AsString
PersonalData_Name="SupplierName"
EndProperty
```

下面的代码负责处理 Selector 窗体：

```
PrivatecustAsNewCustomer
PrivatesupAsNewSupplier

PrivateSubCommand1_Click()
```

```
Dimfrm2AsNewForm2
Setfrm2.PD=cust
frm2.Show1
EndSub

PrivateSubCommand2_Click()
Dimfrm2AsNewForm2
Setfrm2.PD=sup
frm2.Show1
EndSub
```

下面的代码负责处理 DataEntry 窗体：

```
Privatem_pdAsPersonalData
PrivateSubForm_Load()
Withm_pd
Text1=.Name
Text2=.Address
EndWith
EndSub
PublicPropertySetPD(DataAsPersonalData)
Setm_pd=Data
EndProperty
```

请参阅

参阅《MicrosoftVisualBasic6.0 部件工具指南》第二部分“创建 ActiveX 组件”，第六章“组件设计的一般原理”中的“多态性，接口，库的类型和 GUID”和“通过实现接口提供多态性”。参阅《VisualBasic6.0 程序员指南》第九章“用对象编程”中的“多态性”相关内容。

Import 方法（VBA 外接程序对象模型）

从文件给工程添加部件；返回该被添加的新部件。

应用于

VBComponents 集合

语法

object.**Import**(*filename*)**AsVBComponent**

Import 方法语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
filename	必需的。一个 String 型数，它指示欲添加部件的路径及文件名

说明

可以使用 Import 方法给工程添加部件、窗体、模块、类等。

请参阅

Export 方法(VBA 外接程序对象模型)

示例

下面的示例使用 VBComponents 集合中的 Import 方法来复制 test.bas 文件的内容到代码模块中。

```
Application.VBE.ActiveVBProject.VBComponents.Improt("test.bas")
```

Index 属性（控件数组）

返回或设置唯一标识控件数组中一个控件的编号。仅当控件是控件数组的元素时是有效的。

应用于

CommonDialog 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Frame 控件，Hscrol，VScrollBar 控件，Image 控件，Lablel 控件，Line 控件，ListBox 控件，Menu 控件，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，Timer 控件，OLEContainer 控件，DataRepeater 控件，Split 对象，DataList 控件，RemoteData 控件，Data 控件，ADOData 控件

语法

object[(*number*)].**Index**

Index 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	整数值的数值表达式，用来标识控件数组中的一个控件

设置

number 的设置为：

设置	描述
无值	（缺省）不是控件数组的元素
0 至 32,767	是数组的元素。指定一个大于或等于 0 的整数标识控件数组中的一个控件。控件数组中的所有控件具有相同的 Name 属性。VisualBasic 自动地分配在控件数组中有效的下一个整数

说明

因为控件数组元素共享同一个 Name 属性设置，所以必须在代码中使用 Index 属性来指定数组中的一个特定的控件。Index 必须以整数的形式（或一个能计算出一个整数的数字表达式）出现在紧接控件数组之后的圆括号内——例如，MyButtons(3)。也能够使用 Tag 属性的设置在控件数组中区分控件。

当数组中的控件识别出一个事件已经发生时，VisualBasic 就调用控件数组的事件过程，并把可应用的 Index 设置当作附加参数传递。当在运行时动态地用 Load 语句创建控件或用 Unload 语句撤销它们时，该属性也被使用。

虽然缺省状态下 VisualBasic 分配下一个可用的整数作为控件数

组中新的控件的 Index 的值，但也可以改变该分配值并跳过一些数。也可以为数组中的第一个控件的 Index 设一个非 0 的整数。如果在代码中引用一个 Index 的值而在控件数组中没有所标识的控件，那么将产生一个 VisualBasic 运行时错误。

注意要从控件数组中撤销一个控件，需改变该控件的 Name 属性设置，并删除该控件的 Index 属性设置。

请参阅

Load 语句，Unload 语句，Tag 属性，Name 属性，Index 属性(Brush)，Index 属性(Intersection)

示例

该例子开始时有两个 OptionButton 控件，并在每次单击 CommandButton 控件时在窗体中加入一个新的 OptionButton。当单击一个 OptionButton 时，FillStyle 属性被设置并且画一个新的圆。要试用此例，将下面的代码粘贴到具有两个 OptionButton 控件、一个 CommandButton、以及一个大的 PictureBox 控件的窗体的声明部分之中。将两个 OptionButton 控件的 Name 属性都设置为 optButton 以创建一个控件数组。

```
PrivateSubOptButton_Click(IndexAsInteger)
   DimH,W                    '声明变量。
   Picture1.Cls                  '清除图片。
   Picture1.FillStyle=Index        '设置 FillStyle。
   W=Picture1.ScaleWidth/2         '获取圆的大小。
```

```
    H=Picture1.ScaleHeight/2
    Picture1.Circle(W,H),W/2      '画圆。
EndSub

PrivateSubCommand1_Click()
    StaticMaxIdx                  '数组中的最大索引值。
    IfMaxIdx=0ThenMaxIdx=1        '预置 MaxIdx。
    MaxIdx=MaxIdx+1               '索引值增加 1。
    IfMaxIdx>7ThenExitSub         '在窗体中放置八个按钮。
    LoadOptButton(MaxIdx)             '在数组中创建新的项。
    '在前一个按钮下面设置新选项按钮的位置。
    OptButton(MaxIdx).Top=OptButton(MaxIdx-1).Top+360
    OptButton(MaxIdx).Visible=True    '使新的按钮可见。
EndSub
```

IndexedValue 属性

返回或设置在索引列表或数组中的值。

应用于

Property 对象

语法

object.**IndexedValue**[*index1* ,*index2* ,][*index3*,][*index4*]][=*value*]

IndexedValue 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
indexn	指定索引位置的数值表达式。IndexedValue 最多可接受四个指示器。被 IndexedValue 接受的指示器号是 NumIndices 属性返回的同一个属性的值
value	表达式，其值是控件的可接受类型

说明

如果该属性是由 NumIndices 指示的列表，则 IndexedValue 返回由索引参数指定的列表的元素。

只有 NumIndices 属性值大于零时，才使用 IndexedValue。像在 ListBox 控件的 List 属性中那样，索引列表中的各值被设置或返回为单一索引。

请参阅

NumIndices 属性

IndexedValue 属性（VBA 外接程序对象模型）

返回或设置一个属性之成员的值，该属性为索引串行或数组。

应用于

Property 对象

说明

由 IndexedValue 属性返回或设置的值是一个表达式，其值对应的类型是可以被该对象所接受的。对一个索引串行或数组属性来

说，必须使用 IndexedValue 属性而非 Value 属性。一个索引串行是一个指定索引位置的数值表达式。

IndexedValue 可以接受至多四个索引。IndexedValue 可以接受的索引数目由 NumIndices 属性返回。

IndexedValue 属性只有在 NumIndices 属性大于零时才可使用。索引串行中的值由单一索引来设置或返回。

请参阅

Item 方法(VBA 外接程序对象模型)，NumIndices 属性(VBA 外接程序对象模型)，Value 属性

InitDir 属性

初始返回或设置文件目录。

应用于

CommonDialog 控件(Open，SaveAs 对话框)

语法

object.**InitDir**[=*string*]

InitDir 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
string	字符串表达式，它指定初始文件目录

说明

该属性用于为打开或另存为对话指定初始的目录。如果此属性没

有指定，则使用当前目录。

数据类型

String

Initialize 事件

当应用程序创建 Form、MDIForm、User 控件、PropertyPage 或类的实例时发生。

应用于

DataReport 对象，WebClass 对象，DHTMLPageDesigner 对象，PropertyPage 对象，UserControl 对象，UserDocument 对象，Class，From 对象，Forms 集合，MDIForm 对象

语法

PrivateSub*object*_**Initialize**()

object 所在处代表对象表达式，其值是“应用于”列表中的一个对象。

说明

如下情况触发 Initialize 事件：

用 CreateObject 函数创建类的一个实例，如：

```
SetX=CreateObject("Project1.MyClass")
```

引用窗体或类实例的属性或事件，该属性或事件由用户代码自动创建。

MyForm.Caption="Example"

应用此事件初始化 Form、MDIForm、或类所用的数据。对 Form 或 MDIForm，Initialize 事件在 Load 事件之前发生。

请参阅

Load 事件，Terminate 事件

InitProperties 事件

创建对象的新实例时，发生该事件。

应用于

UserControl 对象，UserDocument 对象，Class

语法

Sub*object*_InitProperties()

InitProperties 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

该事件允许对象的创建者初始化对象的新实例。只有当创建对象的新实例时，才发生该事件；这使得对象的创建者能够区分创建对象的新实例和加载对象的旧实例。

将初始化新实例的代码放置到 InitProperties 事件（而不是 Initialize 事件）中，可以避免这种情况的发生：使用 ReadPro

perties 事件将数据加载到对象的旧实例中，从而撤销了对象的初始化。

请参阅

ReadProperties 事件，Initialize 事件

Input#语句

从已打开的顺序文件中读出数据并将数据指定给变量。

语法

Input#*filenumber*,*varlist*

Input#语句的语法具有以下几个部分：

部分	描述
filenumber	必需。任何有效的文件号
varlist	必需。用逗号分界的变量列表，将文件中读出的值分配给这些变量；这些变量不可能是一个数组或对象变量。但是，可以使用变量描述数组元素或用户定义类型的元素

说明

通常用 Write# 将 Input# 语句读出的数据写入文件。该语句只能用于以 Input 或 Binary 方式打开的文件。

在读出数据时不经修改就可直接将标准的字符串或数值数据指定给变量。下表说明如何处理其它输入数据：

数据	指定给变量的值
分隔逗号或空白行	Empty
#NULL#	Null
#TRUE#或#FALSE#	True 或 False
#yyyy-mm-ddhh:mm:ss#	用表达式表示的日期与/或时间
#ERRORerrornumber#	errornumber（此变量是一个 Variant，用来标记错误）

输入数据中的双引号符号("")将被忽略。

文件中数据项目的顺序必须与 varlist 中变量的顺序相同，而且与相同数据类型的变量匹配。如果变量为数值类型而数据不是数值类型，则指定变量的值为零。

在输入数据项目时，如果已到达文件结尾，则会终止输入，并产生一个错误。

注意为了能够用 Input#语句将文件的数据正确读入到变量中，在将数据写入文件时，要使用 Write#语句而不使用 Print#语句。使用 Write#语句可以确保将各个单独的数据域正确分隔开。

请参阅

Input 函数，Open 语句，Print#语句，Write#语句

示例

本示例使用 Input#语句将文件内的数据读入两个变量中。本示例假设 TESTFILE 文件内含数行以 Write#语句写入的数据；也就是说，每一行数据中的字符串部分都是用双引号括起来，而与数字用逗号隔开，例如，("Hello",234)。

```
DimMyString,MyNumber
Open"TESTFILE"ForInputAs#1 '打开输入文件。
DoWhileNotEOF(1)   '循环至文件尾。
   Input#1,MyString,MyNumber    '将数据读入两个变量。
   Debug.PrintMyString,MyNumber'在立即窗口中显示数据。
Loop
Close#1    '关闭文件。
```

Input 函数

返回 String，它包含以 Input 或 Binary 方式打开的文件中的字符。

语法

Input(*number*,[#]*filenumber*)

Input 函数的语法具有以下几个部分：

部分	描述
number	必需的。任何有效的数值表达式，指定要返回的字符个数
filenumber	必需的。任何有效的文件号

说明

通常用 Print#或 Put 将 Input 函数读出的数据写入文件。Input 函数只用于以 Input 或 Binary 方式打开的文件。

与 Input#语句不同，Input 函数返回它所读出的所有字符，包括逗

号、回车符、空白列、换行符、引号和前导空格等。

对于 Binary 访问类型打开的文件，如果试图用 Input 函数读出整个文件，则会在 EOF 返回 True 时产生错误。在用 Input 读出二进制文件时，要用 LOF 和 Loc 函数代替 EOF 函数，而在使用 EOF 函数时要配合以 Get 函数。

注意对于文本文件中包含的字节数据要使用 InputB 函数。对于 InputB 来说，number 指定的是要返回的字节个数，而不是要返回的字符个数。

请参阅

Input#语句

示例

本示例使用 Input 函数来一次读文件中一个字符，并将它显示到“立即”窗口。本示例假设 TESTFILE 文件内含数行文本数据。

```
DimMyChar
Open"TESTFILE"ForInputAs#1 '打开文件。
DoWhileNotEOF(1)   '循环至文件尾。
   MyChar=Input(1,#1)   '读入一个字符。
   Debug.PrintMyChar    '显示到立即窗口。
Loop
Close#1    '关闭文件。
```

InputBox 函数

在一对话框来中显示提示，等待用户输入正文或按下按钮，并返回包含文本框内容的 String。

语法

InputBox(*prompt*[,*title*][,*default*][,*xpos*][,*ypos*][,*helpfile*,*context*])

InputBox 函数的语法具有以下几个命名参数：

部分	描述
prompt	必需的。作为对话框消息出现的字符串表达式。*prompt* 的最大长度大约是 1024 个字符，由所用字符的宽度决定。如果 *prompt* 包含多个行，则可在各行之间用回车符(Chr(13))、换行符(Chr(10))或回车换行符的组合(Chr(13)&Chr(10))来分隔
title	可选的。显示对话框标题栏中的字符串表达式。如果省略 *title*，则把应用程序名放入标题栏中
default	可选的。显示文本框中的字符串表达式，在没有其它输入时作为缺省值。如果省略 *default*，则文本框为空
xpos	可选的。数值表达式，成对出现，指定对话框的左边与屏幕左边的水平距离。如果省略 *xpos*，则对话框会在水平方向居中

ypos 可选的。数值表达式，成对出现，指定对话框的上边与屏幕上边的距离。如果省略 *ypos*，则对话框被放置在屏幕垂直方向距下边大约三分之一的位置

helpfile 可选的。字符串表达式，识别帮助文件，用该文件为对话框提供上下文相关的帮助。如果已提供 *helpfile*，则也必须提供 *context*

context 可选的。数值表达式，由帮助文件的作者指定给某个帮助主题的帮助上下文编号。如果已提供 *context*，则也必须要提供 *helpfile*

说明

如果同时提供了 helpfile 与 context，用户可以按 F1(Windows)or HELP(Macintosh)来查看与 context 相应的帮助主题。某些主应用程序，例如，MicrosoftExcel，会在对话框中自动添加一个 Help 按钮。如果用户单击 OK 或按下 ENTERRETURN，则 InputBox 函数返回文本框中的内容。如果用户单击 Cancel，则此函数返回一个长度为零的字符串("")。

注意如果还要指定第一个命名参数以外的参数，则必须在表达式中使用 InputBox。如果要省略某些位置参数，则必须加入相应的逗号分界符。

请参阅

MsgBox 函数

示例

本示例说明使用 InputBox 函数来显示用户输入数据的不同用法。

如果省略 x 及 y 坐标值，则会自动将对话框放置在两个坐标的正中。如果用户单击“确定”按钮或按下“ENTER”按键，则变量 MyValue 保存用户输入的数据。如果用户单击“取消”按钮，则返回一零长度字符串。

```
DimMessage,Title,Default,MyValue
Message="Enteravaluebetween1and3"  '设置提示信息。
Title="InputBoxDemo"  '设置标题。
Default="1"  '设置缺省值。
'显示信息、标题及缺省值。
MyValue=InputBox(Message,Title,Default)

'使用帮助文件及上下文。“帮助”按钮便会自动出现。
MyValue=InputBox(Message,Title,,,,"DEMO.HLP",10)

'在 100,100 的位置显示对话框。
MyValue=InputBox(Message,Title,Default,100,100)
```

InSelection 属性

该属性返回或赋值控件的选择状态。

应用于

VBControl 对象

语法

object.**InSelection**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

如果包含在另一个控件之中的控件的 InSelection 属性被设置为 True，则不在那个控件中的所有控件（或在其它控件中的所有控件）将不被选择。

请参阅

SelectedControls 属性，SelectedVBControls 集合

Insert 方法

插入一个现存的 ExportFormat 对象到 ExportFormats 集合。

应用于

ExportFormats 集合

语法

object.**Insert***format***

Insert 方法的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
format	要插入的 ExportFormat 对象

示例

本例创建一个新的 ExportFormat 对象，并把它插入到

ExportFormats 集合。

```
DimexfSpecialAsNewExportFormat
WithexfSpecial
    .FileFormatString="SpecialReport(*.txt)"
    .FileFilter="*.txt"
    .FormatType=rptFmtText
    .Key="Special"
    .Template="SpecialReport"&vbCrLf&rptTagBody
EndWith

WithDataReport1
    .ExportFormats.InsertexfSpecial
    .ExportReport"Special"
EndWith
```

InsertFile 方法

从文件中将代码插入到代码模块。

语法

object.**InsertFile**(*filename*)**AsString**

该 InsertFile **语法有这些部分：**

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的对象
filename	必需的。一个字符串表达式，指定包含插入到代码模块中代码的文件

应用于

VBComponent 对象

InsertLines 方法

在一个代码块的某个指定位置，插入一行或多行的代码。

应用于

CodeModule 对象

语法

object.**InsertLines**(*line*,*code*)

InsertLines 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
line	必需的。一个 Long 型数，用来指定要插入代码的位置
code	必需的。一个 String 型数据，即要插入的代码

说明

如果使用 InsertLines 方法所插入的文本用换行回车分隔，则将依此插入一些行。

请参阅

DeleteLines 方法，Lines 方法

示例

下列示例使用 InsertLines 方法将“OptionExplicit”这行文本插入到指定的代码窗格中。

```
Application.VBE.CodePanes(1).CodeModule.InsertLines1 ,
"OptionExplicit"
```

InsertObjDlg 方法

显示插入对象对话框。

应用于

OLE 包容器控件

语法

object.**InsertObjDlg**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

在运行时，显示这个对话框，通过选取对象的类型（链接的或嵌入的）来创建链接对象或内嵌对象，并由应用程序提供该对象。使用 OLETypeAllowed 属性，确定可用这个对话框创建的对象的类型（链接的、嵌入的、或均可）。

当创建新对象时，与类名（如 Excel.exe）关联的应用程序，必须已在操作系统中正确地作了注册（应用程序的安装程序应该将该应用程序正确地作了注册。）

请参阅

OLETypeAllowed 属性

Instancing 属性

设置一个值，确定能否在工程之外创建公共类的实例。如果可以，如何实现。运行时无效。

设置值

Instancing 属性设置如下：

设置	描述
1	（缺省）私有。不允许其它应用程序访问关于类的类型库信息，也不能创建该类的实例。私有对象只能在自己的部件中使用。 工程类型不同，实例属性缺省也不同。只有对于 StandardExe 工程中的类模块，私有才是缺省的 当在 ActiveXExe 工程或 ActiveXDLL 工程中插入一个新的类模块时，Instancing 属性的缺省值是 MultiUse。当在 ActiveX 控件工程中插入一个新的类模块时，Instancing 属性的缺省值是 PublicNotCreatable
2	PublicNotCreatable。只有在自己的部件首先创建了对象

	的前提下，其它应用程序才能应用此类的对象。不能用 CreateObject 函数或 New 操作符来创建类对象
3	SingleUse。允许其它应用程序从这个类创建对象，但客户创建的该类的每个对象都启动部件的一个新的实例。在 ActiveXDLL 工程中是不允许的
4	GlobalSingleUse。类似于 SingleUse，除了此类的各种属性和方法可以像简单的全局函数那样被调用。在 ActiveXDLL 工程中是不允许的
5	MultiUse。允许其它应用程序从这个类创建对象。部件的一个实例可提供按此方式创建的任意数目的对象
6	GlobalMultiUse。类似于 MultiUse，只附加了一点：这个类的各种属性和方法可以像简单的全局函数那样被调用。该类的实例不需要显式创建，因为它会自动创建

设置	应用的工程类型			
	ActiveXExe	ActiveXDLL	ActiveXControl	Std.Exe
Private	X	X	X	X
PublicNotCreatable	X	X	X	
SingleUse	X			
GlobalSingleUse	X			
MultiUse	X	X		
GlobalMultiUse	X	X		

说明

在 VisualBasic5.0 里，Instancing **属性应用于 Class 模块，并得到扩展，以便与 VisualBasic4.0**Public **属性的功能合并在一起。如果类是可创建的，从其它应用程序创建该类的实例，可用下面的任何一种技术：**

用 CreateObject **函数，如：**

```
SetMyInstance=CreateObject("MyProject.MyClass")
```

在同一工程里（或当 Public **属性设为** True **时，在工程之外）用** Dim **语句，如：**

```
DimMyInstanceAsNewMyClass
```

关键字 New 表明 MyInstance 被声明为 MyClass 的一个新实例。如果 Public 属性为 False，则 Instancing 属性被忽略。总可以在定义该类的工程内创建类实例。如果 Public 属性为 True，则此类可视。因此一旦该类的实例存在，那么它是能受控于其它应用程序的。注意 GlobalMultiUse 对象的各种属性和方法，不是提供对象的部件的全局名空间的一部分。例如，在包含 GlobalUtility 类模块的工程内部，为了使用这个对象的各种属性和方法，必须显式创建 GlobalUtility 的实例。全局对象的其它限制，列在“部件工具指南”的“编译代码部件”的“全局对象和代码库”中。

InStr 函数

返回 Variant(Long)，指定一字符串在另一字符串中最先出现的位置。

语法

InStr([*start*,]*string1*,*string2*[,*compare*])

InStr 函数的语法具有下面的参数：

部分	说明
start	可选的。为数值表达式，设置每次搜索的起点。如果省略，将从第一个字符的位置开始。如果 *start* 包含 Null，将发生错误。如果指定了 *compare* 参数，则一定要有 *start* 参数
string1	必需的。接受搜索的字符串表达式
string2	必需的。被搜索的字符串表达式
compare	可选的。指定字符串比较。如果 *compare* 是 Null，将发生错误。如果省略 *compare*，OptionCompare 的设置将决定比较的类型

设置

compare 参数设置为：

常数	值	描述
vbUseCompareOption	-1	使用 OptionCompare 语句设置执行一个比较。
vbBinaryCompare	0	执行一个二进制比较。
vbTextCompare	1	执行一个按照原文的比较。
vbDatabaseCompare	2	仅适用于 MicrosoftAccess，执行一个基于数据库中信息的比较。

返回值

如果	InStr 返回
String1 为零长度	0
String1 为 Null	Null
String2 为零长度	Start
String2 为 Null	Null
String2 找不到	0
在 string1 中找到 string2	找到的位置
Start>string2	0

说明

InStrB 函数作用于包含在字符串中的字节数据。所以 InStrB 返回的是字节位置，而不是字符位置。

示例

本示例使用 InStr 函数来查找某字符串在另一个字符串中首次出现的位置。

```
DimSearchString,SearchChar,MyPos
SearchString="XXpXXpXXPXXP"     '被搜索的字符串。
SearchChar="P" '要查找字符串"P"。

'从第四个字符开始，以文本比较的方式找起。返回值为 6（小写 p）。
'小写 p 和大写 P 在文本比较下是一样的。
MyPos=Instr(4,SearchString,SearchChar,1)
```

```
'从第一个字符开使，以二进制比较的方式找起。返回值为 9（大写 P）。
'小写 p 和大写 P 在二进制比较下是不一样的。
MyPos=Instr(1,SearchString,SearchChar,0)

'缺省的比对方式为二进制比较（最后一个参数可省略）。
MyPos=Instr(SearchString,SearchChar)    '返回 9。

MyPos=Instr(1,SearchString,"W")     '返回 0。
```

InStrRev 函数

返回一个字符串在另一个字符串中出现的位置，从字符串的末尾算起。

语法

InstrRev(*string1*,*string2*[,*start*[,*compare*]])

InstrRev 函数语法有如下几部分：

部分	描述
string1	必需的。要执行搜索的字符串表达式
string2	必需的。要搜索的字符串表达式
start	可选的。数值表达式，设置每次搜索的开始位置。如果忽略，则使用–1，它表示从上一个字符位置开始搜索。如果 start 包含 Null，则产生一个错误
compare	可选的。数字值，指出在判断子字符串时所使用的比较方法。如果忽略，则执行二进制比较。关于其值，请参阅“设置值”部分

设置值

compare 参数值如下：

常数	值	描述
VbUseCompareOption	–1	用 OptionCompare 语句的设置值来执行比较
vbBinaryCompare	0	执行二进制比较
vbTextCompare	1	执行文字比较
vbDatabaseCompare	2	只用于 MicrosoftAccess。基于您的数据库信息执行比较

返回值

InStrRev 返回值如下：

如果	InStrRev 返回
string1 长度为零	0
string1 为 Null	Null
string2 长度为零	Start
string2 为 Null	Null
string2 没有找到	0
string2 在 string1 中找到	找到匹配字符串的位置
start>Len(string2)	0

说明

请注意，InstrRev 函数的语法和 Instr 函数的语法不相同。

Int、Fix 函数

返回参数的整数部分。

语法

Int(*number*)

Fix(*number*)

必要的 number 参数是 Double 或任何有效的数值表达式。如果 number 包含 Null，则返回 Null。

说明

Int 和 Fix 都会删除 number 的小数部份而返回剩下的整数。

Int 和 Fix 的不同之处在于，如果 number 为负数，则 Int 返回小于

或等于 number 的第一个负整数，而 Fix 则会返回大于或等于 number 的第一个负整数。例如，Int 将-8.4 转换成-9，而 Fix 将-8.4 转换成-8。

Fix(number)等于：

```
Sgn(number)*Int(Abs(number))
```

请参阅

Math 函数，Found 函数，IntegerData 类型

示例

本示例说明 Int 及 Fix 函数在返回某数值的整数部分时有何不同。当参数为负数时，Int 函数返回小于或等于该参数之最大整数，而 Fix 函数则返回大于或等于该参数之最小整数。

```
DimMyNumber
MyNumber=Int(99.8) '返回 99。
MyNumber=Fix(99.2) '返回 99。

MyNumber=Int(-99.8)    '返回-100。
MyNumber=Fix(-99.8)    '返回-99。

MyNumber=Int(-99.2)    '返回-100。
MyNumber=Fix(-99.2)    '返回-99。
```

Integer 数据类型

Integer 变量存储为 16 位（2 个字节）的数值形式，其范围为-32,768 到 32,767 之间。Integer 的类型声明字符是百分比符号(%)。

也可以用 Integer 变量来表示枚举值。枚举值可包含一个有限集合，该集合包含的元素都是唯一的整数，每一个整数都在它使用时的上下文当中有其特殊意义。枚举值为在已知数量的选项中做出选择提供了一种方便的方法，例如，black=0，white=1 等等。较好的编程作法是使用 Const 语句将每个枚举值定义成常数。

请参阅

TypeConversion 函数，Data 类型 Summary，LongData 类型，VariantData 类型，Deftype 语句

Interval 属性

返回或设置对 Timer 控件的计时事件各调用间的毫秒数。

应用于

Timer 控件

语法

object.**Interval**[=*milliseconds*]

Interval 属性语法有以下组成部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
milliseconds	数值表达式，指定毫秒数，“设置值”中有详细说明

设置值

milliseconds 的设置值为：

设置值	描述
0	（缺省值）使 Timer 控件无效
1 至 65,535	设置的时间间隔（以毫秒计），在 Timer 控件 Enabled 属性设置为 True 时开始有效，例如，10，000 毫秒等于 10 秒。最大值为 65,535 毫秒，等于 1 分钟多一些

说明

可以在设计时或在运行时设置 Timer 控件的 Interval 属性。使用 Interval 属性时，请记住：

Timer 控件的 Enabled 属性决定该控件是否对时间的推移做响应。将 Enabled 设置为 False 会关闭 Timer 控件，设置为 True 则打开它。当 Timer 控件置为有效时，倒计时总是从其 Interval 属性的设置值开始。

创建 Timer 事件程序用以告诉 VisualBasic 在每次 Interval 到时该做什么。

请参阅

Enabled 属性，Enabled 属性(ActiveX 控件)

示例

这个例子可以调整窗体切换颜色的速度。要尝试这个例子，请将代码粘贴到包含 Timer 控件、HScrollBar 控件（水平滚动条）和 PictureBox 控件的窗体的声明部分，然后按 F5 键并单击滚动条。

```
PrivateSubForm_Load()
   Timer1.Interval=900 '设置时间间隔。
   HScroll1.Min=100'设置最小值。
   HScroll1.Max=900'设置最大值。
EndSub
PrivateSubHScroll1_Change()
   '根据滚动条的数值设置时间间隔。
   Timer1.Interval=1000-HScroll1.Value
EndSub
PrivateSubTimer1_Timer()
   '在红色和蓝色之间切换背景色。
   IfPicture1.BackColor=RGB(255,0,0)Then
   Picture1.BackColor=RGB(0,0,255)
   Else
   Picture1.BackColor=RGB(255,0,0)
   EndIf
EndSub
```

InvisibleAtRuntime 属性

返回或设置一个值，它决定控件在运行时是否应有不可见的窗口。在创建控件时，InvisibleAtRuntime 属性可读可写，在控件运行时，该属性是不可用。

应用于

UserControl 对象

设置值

InvisibleAtRuntime 的设置值为：

设置值	描述
true	使控件在运行时具有不可见的窗口。控件的容器可保持控件在运行时不可见，就像 Timer 控件一样。此时控件仍然处于活动状态，因此仍可编写与控件交互的程序。扩展对象中没有 Visible 属性
false	（缺省）控件在运行时与正常的控件相同，此时 Visible 扩展属性的状况决定了控件的可见性

说明

重点要使控件在运行时不可见，不要使用 Visible 扩展属性。如果使用了这个属性，控件在运行时仍将具有可见控件的所有开销。除此之外，因为开发者和最终用户都可以使用扩展属性，而他们可能会使控件成为可见的。

有些容器可能不支持 InvisibleAtRuntime 属性；在这种情况下，控

件在运行时将是可见的。

在创建运行时不可见的控件之前，请考虑创建一个由进程内的代码部件(ActiveXDLL)提供的普通对象代替。进程内的代码部件提供的对象所需要的资源比控件需要的资源少，甚至比不可见控件所需的资源还少。实现不可见控件的唯一原因就是想利用只有 ActiveX 控件才能提供的优越性。

请参阅

Visible 属性

IPmt 函数

返回一个 Double，指定在一段时间内对定期定额支付且利率固定的年金所支付的利息值。

语法

IPmt(*rate*,*per*,*nper*,*pv*[,*fv*[,*type*]])

IPmt 函数有下列命名参数：

部分	描述
rate	必要。Double 指定每一期的利率。例如，如果有一笔贷款年百分率(APR)为 100%且按月付款的汽车贷款，则每一期的利率为 0.1/12，或 0.0083
per	必要。Double 指定在 *nper* 间范围 1 中的付款周期
nper	必要。Double 指定一笔年金的付款总期数。例如，如果在一笔为期四年的汽车贷款中选择按月付款方式，则贷款共有 4*12（或 48）个付款期
pv	必要。Double，指定未来一系列付款或收款的现值。例如，当借钱买汽车时，向贷方所借金额为将来每月偿付给贷方款项的现值
fv	可选。Variant 指定在付清贷款后所希望的未来值或现金结存。例如，贷款的未来值在贷款付清后为 0 美元。但是，如果想要在 18 年间存下 50，000 美元作为子女教育基金，那么 50，000 美元为未来值。如果省略的话，缺省值为 0
type	可选。Variant 指定贷款到期时间。如果贷款在贷款周期结束时到期，请使用 0。如果贷款在周期开始时到期，请使用 1。如果省略的话，缺省值为 0

说明

年金是指在一段时间内的一系列固定现金支付。年金可以是贷款（如房屋抵押贷款），也可以是一笔投资（如按月储蓄计划）。

在支付期间必须用相同的单位计算 rate 和 nper 参数。例如，如果 rate 用月份计算，则 nper 也必须用月份计算。

对所有参数，用负数表示现金支出（如储蓄存款），而用正数表示现金收入（如红利支票）。

请参阅

DDB 函数，IRR 函数，MIRR 函数，Nper 函数，NPV 函数，Pmt 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

本示例使用 IPmt 函数来计算每期的付款中有多少是属于利息，其中每期付款金额相同。计算时需给定每期利率（APR/12），利息的付款周期（Period），付款总期数（TotPmts），贷款现值或本金（PVal），贷款的未来值（FVal）及付款方式，以数值表示期初或期末付款（PayType）。

```
DimFVal,Fmt,PVal,APR,TotPmts,PayType,Period,IntPmt,TotInt,Msg
ConstENDPERIOD=0,BEGINPERIOD=1 '付款方式。
FVal=0 '对贷款而言通常为零。
Fmt="###,###,##0.00"   '定义金额格式。
PVal=InputBox("Howmuchdoyouwanttoborrow?")
APR=InputBox("Whatistheannualpercentagerateofyourloan?")
IfAPR>1ThenAPR=APR/100 '确保格式正确。
TotPmts=InputBox("Howmanymonthlypayments?")
PayType=MsgBox("Doyoumakepaymentsatendofthemonth?",vbYesNo)
IfPayType=vbNoThenPayType=BEGINPERIODElsePayType=ENDPERIOD
ForPeriod=1ToTotPmts   '将所有利息求和。
   IntPmt=IPmt(APR/12,Period,TotPmts,-PVal,FVal,PayType)
```

```
    TotInt=TotInt+IntPmt
NextPeriod
Msg="You'llpayatotalof"&Format(TotInt,Fmt)
Msg=Msg&"ininterestforthisloan."
MsgBoxMsg  '显示结果。
```

IRR 函数

返回一个 Double，指定一系列周期性现金流（支出或收入）的内部利率。

语法

IRR(*values*()[,*guess*])

IRR 函数有下列命名参数：

部分	描述
values()	必要。Double 数组，指定现金流值。此数组必须至少含有一个负值（支付）和一个正值（收入）
guess	可选。Variant，指定 IRR 返回的估算值。如果省略，guess 为 0.1(10%)

说明

返回的内部利率是在正常的时间间隔内，一笔含有支出及收入的投资得到的利率。

IRR 函数使用数组中数值的顺序来解释支付和收入的顺序。要确

保支付和收入的顺序正确。每一时期的现金流不必像年金那样固定不变。

IRR 是利用叠代进行计算。先从 guess 的值开始，IRR 反复循环进行计算，直到精确度达到 0.00001%。如果经过 20 次反复叠代测试还不能得到结果，则 IRR 计算失败。

请参阅

DDB 函数，FV 函数，Ipmt 函数，MIRR 函数，NPer 函数，NPV 函数，Pmt 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

在本示例中，IRR 函数会计算由数组 Values()内所含一系列期间的 5 笔现金流量之实质报酬率（internalrateofreturn）。第一个数组元素为一负数现金流，代表事业初始成本（事业初始成本）。其余 4 个正数现金流代表后续 4 年内的收入状况。Guess 为实质报酬率之估计值。

```
DimGuess,Fmt,RetRate,Msg
StaticValues(5)AsDouble                    '声明数组
Guess=.1                                   'Guess 由 10%开始
Fmt="#0.00"                                '定义百分比的显示格式
Values(0)=-70000                           '事业初始成本
'正数现金流代表连续四年的收入状况。
Values(1)=22000:Values(2)=25000
Values(3)=28000:Values(4)=31000
```

```
RetRate=IRR(Values(),Guess)*100                    '计算实质报酬率
Msg="Theinternalrateofreturnforthesefivecashflowsis"
Msg=Msg&Format(RetRate,Fmt)&"percent."
MsgBoxMsg                                          '显示实质报酬率
```

IsArray 函数

返回 Boolean 值，指出变量是否为一个数组。

语法

IsArray(*varname*)

必要的 varname 参数是一个指定变量的标识符。

说明

如果变量是数组，则 IsArray 返回 True；否则返回 False。对于包含数组的 variant 表达式来说，IsArray 尤为有用。

请参阅

IsData 函数,IsEmpty 函数,IsError 函数,IsMissing 函数,IsNull 函数，IsNumeric 函数，IsObject 函数，TypeName 函数，VarType 函数，VariantData 类型，Array 函数

示例

本示例使用 IsArray 函数来检查变量是否为数组。

```
DimMyArray(1To5)AsInteger,YourArray,MyCheck      '声明数组变量。
YourArray=Array(1,2,3)                           '使用数组函数。
MyCheck=IsArray(MyArray)                         '返回True。
```

```
MyCheck=IsArray(YourArray)                    '返回 True。
```

IsBindable 属性

返回布尔型值，指示属性是否可绑定。该属性是只读的。

应用于

DataBinding 对象

语法

object.**IsBindable**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

使用该属性来决定属性是否可绑定。

注意该属性通常用于 Wizard 中，检查某个属性是否可绑定。

IsBroken 属性

返回一个布尔值，它指定 Reference 对象是否指向一个注册表中有效的引用，此属性为只读。

应用于

Reference 对象

返回值

IsBroken 属性返回下列这些值：

值	描述
true	Reference 对象不再指向一个在注册表中的有效引用
false	Reference 对象指向一个在注册表中的有效引用

请参阅

BuiltIn 属性，FullPath 属性

示例

下列示例使用 IsBroken 属性返回一个值，指示特定工程中的指定的 Reference 对象是否已被中断。

```
Debug.PrintApplication.VBE.vbprojects(1).References(1).IsBroken
```

IsDataSource 属性

返回布尔型值，指示属性是否是数据源。该属性是只读的。

应用于

DataBinding 对象

语法

object.**IsDataSource**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

使用该属性来判定属性是否是数据源，从而决定是否可以将之附加到数据控件上。

注意该属性通常用于向导中，检查某个属性是否是数据源。

IsDate 函数

返回 Boolean 值，指出一个表达式是否可以转换成日期。

语法

IsDate(*expression*)

必要的 *expression* 参数是一个 Variant，包含日期表达式或字符串表达式，这里的字符串表达式是可以作为日期或时间来认定的。

说明

如果表达式是一个日期，或可以作为有效日期识别，则 IsDate 返回 True；否则返回 False。在 MicrosoftWindows 中，有效日期的范围介于公元 100 年 1 月 1 日与公元 9999 年 12 月 31 日之间；其有效范围随操作系统不同而不同。

请参阅

IsArray 函数，IsEmpty 函数，IsError 函数，IsMissing 函数，IsNull 函数，IsNumeric 函数，IsObject 函数，TypeName 函数，VarType 函数，DateData 类型，VariantData 类型

示例

本示例使用 IsDate 函数判断表达式是否可以转换为日期格式。

```
DimMyDate,YourDate,NoDate,MyCheck
MyDate="February12,1969":YourDate=#2/12/69#:NoDate="Hello"
```

```
MyCheck=IsDate(MyDate) '返回True。
MyCheck=IsDate(YourDate)   '返回True。
MyCheck=IsDate(NoDate) '返回False。
```

IsDirty 属性

返回一个值，指示该部件自上次保存以来是否被编辑过。

应用于

VBComponent 对象，VBProject 对象

语法

object.**IsDirty**

其中 *object* 是一个对象表达式，其值是“应用于”列表中的对象。

IsEmpty 函数

返回 Boolean 值，指出变量是否已经初始化。

语法

IsEmpty(*expression*)

必需的 *expression* 参数是一个 Variant，包含一个数值或字符串表达式。但是，因为 IsEmpty 被用来确定个别变量是否已初始化，所以 *expression* 参数通常是单一变量名。

说明

如果变量未初始化或已明确设置为 Empty，则 IsEmpty 返回 True；

否则返回 False。如果 expression 含有多个变量，则 IsEmpty 总是返回 False。IsEmpty 只返回对 variant 表达式有意义的信息。

请参阅

IsArray 函数,IsDate 函数,IsError 函数,IsMissing 函数,IsNull 函数，IsNumeric 函数，IsObject 函数，TypeName 函数，VarType 函数，VariantData 类型

示例

本示例使用 IsEmpty 函数检查变量是否已经初始化。

```
DimMyVar,MyCheck
MyCheck=IsEmpty(MyVar) '返回True。

MyVar=Null '赋以Null。
MyCheck=IsEmpty(MyVar) '返回False。

MyVar=Empty     '赋以Empty。
MyCheck=IsEmpty(MyVar) '返回True。
```

IsError 函数

返回 Boolean 值，指出表达式是否为一个错误值。

语法

IsError(*expression*)

必需的 *expression* 参数，可以是任何有效表达式。

说明

利用 CVErr 函数将实数转换成错误值就会建立错误值。IsError 函数被用来确定一个数值表达式是否表示一个错误。如果 expression 参数表示一个错误，则 IsError 返回 True；否则返回 False。

请参阅

CVErr 函数，IsAray 函数，IsDate 函数，IsEmpt 函数，IsMissing 函数，IsNull 函数，IsNumeric 函数，IsObject 函数，TypeName 函数，VarType 函数，VariantData 类型

示例

本示例使用 IsError 函数检查数值表达式的结果是否为错误代号。用 CVErr 函数从用户自定义函数返回 ErrorVariant。假设 UserFunction 为返回错误值的用户自定义函数过程的名称。例如，函数中返回值写法可用 UserFunction=CVErr(32767)，其中 32767 为用户自定义的数。

```
DimReturnVal,MyCheck
ReturnVal=UserFunction()
MyCheck=IsError(ReturnVal) '返回 True。
```

IsMissing 函数

返回 Boolean 值，指出一个可选的 Variant 参数是否已经传递给过程。

语法

IsMissing(*argname*)

必需的 *argname* 参数包含一个可选的 Variant 过程参数名。

说明

使用 IsMissing 函数来检测在调用一个程序时是否提供了可选 Variant 参数。如果对特定参数没有传递值过去，则 IsMissing 返回 True；否则返回 False。如果 IsMissing 对某个参数返回 True，则在其它代码中使用这个丢失的参数将产生一个用户自定义的错误。如果对 ParamArray 参数使用 IsMissing，则函数总是返回 False。为了检测空的 ParamArray，可试看一下数组的上界是否小于它的下界。

注意 IsMissing 对简单数据类型（例如 Integer 或 Double）不起作用，因为与 Variants 不同，它们没有“丢失”标志位的前提。正由于此，对于可选参数类型，可以指定缺省值。如果调用过程时，参数被忽略，则该参数将具有该缺省值，如下列示例中所示：

```
SubMySub(OptionalMyVarAsString="specialvalue")
IfMyVar="specialvalue"Then
   'MyVar 被忽略。
Else
...
EndSub
```

在许多情况下，如果用户从函数调用中忽略，则可以通过使缺省值等于希望 MyVar 所包含的值来完全忽略 IfMyVar 测试。这将使

您的代码更简洁有效。

请参阅

IsArray 函数，IsDate 函数，IsEmpty 函数，IsError 函数，IsNullFunnction，IsNumeric 函数，Is 对象函数，TypeName 函数,VarType 函数,VariantData 类型,Function 语句,PropertyGet 语句，PropertyLet 语句，PropertySet 语句，Sub 语句

示例

本示例使用 IsMissing 函数检查是否把可选参数传送给用户自定义过程。请注意，除了 Variant 类型以外，其它皆可当作 Optional 参数的缺省值及类型。

```
DimReturnValue
'下列语句调用用户自定义函数。
ReturnValue=ReturnTwice()  '返回 Null。
ReturnValue=ReturnTwice(2) '返回 4。

'函数过程定义。
FunctionReturnTwice(OptionalA)
   IfIsMissing(A)Then
        '如果参数丢失，则返回 Null。
        ReturnTwice=Null
   Else
        '如果参数出现，则返回两倍的值。
```

```
        ReturnTwice=A*2
    EndIf
EndFunciton
```

IsNull 函数

返回 Boolean 值，指出表达式是否不包含任何有效数据(Null)。

语法

IsNull(*expression*)

必需的 *expression* 参数是一个 Variant，其中包含数值表达式或字符串表达式。

说明

如果 expression 为 Null，则 IsNull 返回 True；否则 IsNull 返回 False。如果 expression 由多个变量组成，则表达式的任何作为变量组成成分的 Null 都会使整个表达式返回 True。

Null 值指出 Variant 不包含有效数据。Null 与 Empty 不同，后者指出变量尚未初始化。Null 与长度为零的字符串(“ ”)也不同，长度为零的字符串指的是空串。

重要使用 IsNull 函数是为了确定表达式是否包含 Null 值的。在某些情况下，希望表达式取值为 True，比如希望 IfVar=Null 和 IfVar<>Null 取值为 True，而它们总取值为 False。这是因为任何包含 Null 的表达式本身就是 Null，所以为 False。

请参阅

IsArray 函数，IsDate 函数，IsEmpty 函数，IsError 函数，IsMissing 函数，IsNumeric 函数 IsObject 函数，TypeName 函数，VarType 函数，VariantData 类型

示例

本示例使用 IsNull 函数检查变量值是否为 Null。

```
DimMyVar,MyCheck
MyCheck=IsNull(MyVar)  '返回 False。

MyVar=""
MyCheck=IsNull(MyVar)  '返回 False。

MyVar=Null
MyCheck=IsNull(MyVar)  '返回 True。
```

IsNumeric 函数

返回 Boolean 值，指出表达式的运算结果是否为数。

语法

IsNumeric(*expression*)

必需的 *expression* 参数是一个 Variant，包含数值表达式或字符串表达式。

说明

如果整个 expression 的运算结果为数字，则 IsNumeric 返回 True；

否则返回 False。

如果 expression 是日期表达式，则 IsNumeric 返回 False。

请参阅

IsArray 函数，IsDate 函数，IsFmpty 函数，IsError 函数，IsMissing 函数，IsNull 函数，IsObject 函数，TypeName 函数，VarType 函数，VariantData 类型

示例

本示例使用 IsNumeric 函数判断变量的值是否可为数值。

```
DimMyVar,MyCheck
MyVar="53" '指定值。
MyCheck=IsNumeric(MyVar)    '返回 True。

MyVar="459.95" '指定值。
MyCheck=IsNumeric(MyVar)    '返回 True。

MyVar="45Help" '指定值。
MyCheck=IsNumeric(MyVar)    '返回 False。
```

IsObject 函数

返回 Boolean 值，指出标识符是否表示对象变量。

语法

IsObject(*identifier*)

必需的 *identifier* 参数是一个变量名。

说明

IsObject 只用于确定 Variant 是否属于 VarTypevbObject。如果 Variant 实际引用（或曾经引用过）一个对象，或者如果 Variant 包含 Nothing，则可能出现这种情况。

如果 identifier 是 Object 类型或任何有效的类类型，或者，如果 identifier 是 VarTypevbObject 的 Variant 或用户自定义的对象，则 IsObject 返回 True；否则返回 False。即使变量已设置成 Nothing，IsObject 也仍返回 True。

使用错误捕获方法可以确认对象引用是否有效。

请参阅

IsArray 函数，IsDate 函数，IsEmpty 函数，IsError 函数，IsMissing 函数，IsNull 函数，IsNumeric 函数，TypeName 函数，VarType 函数，VariantData 类型，Set 语句，ObjectData

示例

本示例使用 IsObject 函数判断标识符是否代表对象变量。MyObject 及 YourObject 是同类型的对象变量。它们是只用来说明用途的一般名称。

```
DimMyIntAsInteger,YourObject,MyCheck    '声明变量。
DimMyObjectAsObject
SetYourObject=MyObject '指定对象引用。
MyCheck=IsObject(YourObject)    '返回 True。
```

```
MyCheck=IsObject(MyInt)     '返回False。
```

IsReady 属性

如果指定的驱动器已准备好，返回 True；否则返回 False。

语法

object.IsReady

object 总是一个 Drive 对象。

说明

对于可删除媒体驱动器和 CD-ROM 驱动器来说，仅当插入了适当的媒体，并已准备好供访问时，IsReady 才返回 True。

下面的代码举例说明了 IsReady 属性的用法：

```
SubShowDriveInfo(drvpath)
Dimfs,d,s,t
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(drvpath)
SelectCased.DriveType
Case0:t="Unknown"
Case1:t="Removable"
Case2:t="Fixed"
Case3:t="Network"
Case4:t="CD-ROM"
Case5:t="RAMDisk"
```

```
EndSelect
s="Drive"&d.DriveLetter&":-"&t
Ifd.IsReadyThen
s=s&vbCrLf&"DriveisReady."
Else
s=s&vbCrLf&"DriveisnotReady."
EndIf
MsgBoxs
EndSub
```

请参阅

DriveExists 方法，AvailableSpace 属性，DriveLetter 属性，DriveType 属性，FileSystem 属性，FreeSpace 属性，Path 属性，RootFolder 属性，SerialNumber 属性，ShareName 属性，TotalSize 属性，VolumeName 属性

IsRootFolder 属性

如果指定的文件夹是根文件夹，则返回 True；否则返回 False。

应用于

Folder 对象

语法

object.**IsRootFolder**

object 总是一个 Folder 对象。

说明

下面的代码举例说明了 IsRootFolder 属性的用法：

```
Dimfs
Setfs=CreateObject("Scripting.FileSystemObject")
SubDisplayLevelDepth(pathspec)
Dimf,n
Setf=fs.GetFolder(pathspec)
Iff.IsRootFolderThen
MsgBox"Thespecifiedfolderistherootfolder."
Else
DoUntilf.IsRootFolder
Setf=f.ParentFolder
n=n+1
Loop
MsgBox"Thespecifiedfolderisnested"&n&"levelsdeep."
EndIf
EndSub
```

请参阅

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，Name 属性，ParentFolder 属性，Path 属性，ShortName 属性，ShortPath 属性，Size 属性，SubFolders 属性，Type 属性

Italic 属性

返回或设置 Font 对象的字形为斜体或非斜体。

应用于

Font 对象

语法

object.**Italic**[=*boolean*]

Italic 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定字形的布尔表达式，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
true	打开斜体格式化
false	（缺省值）关闭斜体格式化

说明

Font 对象在设计时不能直接使用。取而代之的是，在“属性”窗口中通过选择控件的 Font 属性并单击“属性”按钮来设置 Italic 属性。在“字体”对话框的“字形”框中，选择“斜体”或“粗斜体”。然而，在运行时，通过为 Font 对象指定 Italic 属性值可直

接设置 Italic。

请参阅

Bold 属性，FontTransparent 属性，Size 属性(Font)，StrikeThrough 属性，Underline 属性，Weight 属性，Name 属性

示例

这个例子完成下面的功能：每次单击鼠标时在窗体上打印文本。要试用此例，可以先将下面的代码粘贴到一个窗体的声明部分中，然后按下 F5 键并双击此窗体。

```
PrivateSubForm_Click()
   Font.Bold=NotFont.Bold   '转换 Bold
   Font.StrikeThrough=NotFont.StrikeThrough'转换 StrikeThrough
   Font.Italic=NotFont.Italic   '转换 Italic
   Font.Underline=NotFont.Underline'转换 Underline
   Font.Size=16'设置 Size 属性
   IfFont.BoldThen
       Print"Fontweightis"&Font.Weight&"(bold)."
   Else
       Print"Fontweightis"&Font.Weight&"(notbold)."
   EndIf
EndSub
```

Item 方法（外接程序）

该方法从指定的集合中返回一个项的名字或索引。

AddIns 集合：

ContainedVBControls 集合：

Members 集合：

SelectedVBControls 集合：

VBControls 集合：

VBNewProjects 集合：

应用于

ContainedVBControls 集合，AddIns 集合，Members 集合，Properties 集合，SelectedVBControls 集合，VBControls 集合，Bands 集合，Linkedwindows 集合，VBProjects 集合

语法

object.**Item**(*index*)

object.**Item**(*collectionindex*,[*controlindex*])**AsVBControl**

object.**Item**(*var*)**AsMember**

Item 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
index	必需的。指定要访问的对象的集合中的名字或索引的 variant 表达式
collectionindex	必需的。指定索引的数值表达式
controlindex	可选的
var	必需的

说明

不能保证给定的集合索引号总是指向相同的项，因为集合中的项可能被添加或删除。只有穿过全部集合进行迭代，且在迭代过程中不添加或删除任何项时，使用集合的索引号才是有用的。

请参阅

Count 属性(VB 集合)，Count 属性(ActiveX 控件)

Item 方法

利用位置或键返回 Collection 对象的指定成员。

应用于

StdDataFormats 集合，Collection 对象，Controls 集合，SelBookMarks 集合，Splits 集合

语法

object.**Item**(*index*)

Item 方法的语法具有下列对象限定符和部分：

部分	描述
Object	必要。对象表达式，其值为“应用于”列表中对象
index	必要。为一表达式，指定集合中成员的位置。如果是数值表达式，则 *index* 必须是从 1 到集合 Count 属性值之间的数值。如果是字符串表达式，则当加入一被引用的成员到集合时，*index* 必须和 *key* 参数对应

说明

如果 index 值无法与集合的任何现有成员匹配，就会导致错误发生。

Item 方法是集合的缺省方法。因此，以下两行程序代码等价：

```
PrintMyCollection(1)
PrintMyCollection.Item(1)
```

请参阅

Caption 属性，Count 属性，Name 属性(VBA 外接程序对象模型)，Add 方法，Remove 方法

示例

下面的示例包括显示 CodePanes 集合的特定数目的两种方法，一个使用 Item 方法。

```
Application.VBE.CodePanes.Item(2).Show
```

本示例使用 Item 方法取得集合对象中对象的引用。假设 Birthdays

为 Collection 对象，下列代码用键“SmithBill”及“SmithAdam’作为 index 参数，从集合引用中取出代表 BillSmith 的生日和 AdamSmith 的生日的对象。请注意，示例中的第一次调用明确使用了 Item 方法，但第二次却没有。两次调用都会成功，因为 Item 方法就是 Collection 的缺省方法。示例中用 Set 指定给 SmithBillBD 及 SmithAdamBD 对象的引用，可用来访问该对象的属性及方法。若要试用本示例程序，请自行建立一个集合，并至少添加引用到的两个对象。

```
DimSmithBillBDAsObject
DimSmithAdamBDAsObject
DimBirthdays
SetSmithBillBD=Birthdays.Item("SmithBill")
SetSmithAdamBD=Birthdays("SmithAdam")
```

Item 方法（VBA 外接程序对象模型）

返回集合中所索引的成员。

应用于

CodePanes 集合，LinkedWindows 集合，Properties 集合(VBA 外接程序对象模型)，References 集合，VBComponents 集合，VBProjects 集合，Windows 集合

语法

object.**Item**(*index*)

Item 方法语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
index	必需的。一个表达式，用来指定集合成员的位置。如为数值表达式，*index* 必须为介於 1 到集合 Count 属性值之间的数字。如为字符串表达式，*index* 必须和将成员添加到集合时指定的 *key* 参数相同

以下表格列出使用 Item **方法时可选的集合及对应的** key **参数。传给** Item **方法的字符串必须和集合的** key **参数匹配。**

集合	Key 参数
Windows	Caption 属性设置
LinkedWindows	Caption 属性设置
CodePanes	无唯一字符串与此集合相关
VBProjects	Name 属性设置
VBComponents	Name 属性设置
References	Name 属性设置
Properties	Name 属性设置

说明

index **参数可以是数值或包含对象标题的字符串。**

Item 属性

返回指定数量的 Collection 对象，或者通过位置，或者通过索引。

应用于

ExportFormats 集合，Sections 集合(DataReportDesigner)，DataMembers 集合，BindingCollection 对象，StdDataFormats 集合，WebItemProperties 对象，DataBindings 集合，DatAObjectFiles 集合，Licenses 集合，Parameters 集合(VisualBasic)

语法

object.**Item**(*index*)

Item 属性语法有下列对象限定符和部分：

部分	描述
object	必需的。对象表达式，其值为“应用于”列表中的对象
index	必需的。指定集合成员位置的表达式。若是数值表达式，索引必须从 1 到集合的 Count 属性值。若是字符串表达式，索引必须符合被引用的成员添加到集合中去时指定的 key 参数

说明

如果索引值与集合中的任何成员都不匹配，则产生错误。Item 是集合的缺省属性。因此，下面的代码行是等效的：

```
PrintMyCollection(1)
PrintMyCollection.Item(1)
```

Item 属性(FileSystemObject 对象)

对 Dictionary 对象中指定的 *Key*，设置或返回一个 Item。对于集合来说，基于指定的 *Key*，返回一个 *Item*。读/写属性。

应用于

Dictionary 对象，Drives 集合，Files 集合，Folders 集合

语法

object.**Item**(*key*)[=*newitem*]

Item 属性具有下列部分：

部分	描述
object	必需的。总是一个集合或 Dictionary 对象的名称。
key	必需的。与被检索或添加的条目相关联的 Key。
newitem	可选的。仅用于 Dictionary 对象；没有用于集合的应用程序。如果提供的话，newitem 是与指定的 *Key* 相关联的新值。

说明

如果在改变某个 item 时，没有找到 key，则用指定的 newitem 创建一个新的 key。如果在试图返回某个已存在条目时，没有找到 key，则创建一个新 key，且其相应的条目为空。

请参阅

CompareMode 属性，Count 属性，Key 属性

ItemActivated 事件

该事件发生在当部件在“工程”窗口被双击时，在有多个工程被加载到 IDE 的情况下，“工程”窗口中某个工程被单击时也发生该事件。

应用于

VBComponentsEvents 对象，VBProjectsEvents 对象，VBProjects 集合

语法

Sub*object*_**ItemActivated**(*vbcomponent***AsVBComponent**)

Sub*object*_**ItemActivated**(*vbproject***AsVBProject**)

ItemActivated 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbcomponent	VBComponent 对象，指出被双击的部件名
vbproject	VBProject 对象，指出被双击的工程名

ItemAdded 事件

该事件发生在工程、控件或部件添加到当前工程之后。

应用于

References 集合，VBComponents 集合，SelectedVBControls 集合，SelectedVBControlsEvents 对象，VBComponentsEvents 对

象,VBControls 集合,VBControlsEvents 对象,VBProjectsEvents 对象,ReferencesEvents 对象,VBProjectsEvents 对象

语法

Sub*object*_ItemAdded(*vbproject*AsVBProject)

Sub*object*_ItemAdded(*vbcomponent*AsVBComponent)

Sub*object*_ItemAdded(*vbcontrol*AsVBControl)

ItemAdded 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbproject	VBProject 对象，指出被加载的工程名
vbcomponent	VBComponent 对象，指出被加载的部件名
vbcontrol	VBControl 对象，指出被加载的控件名

ItemAdded 事件（VBA 外接程序对象模型）

发生在添加一个引用之后。

应用于

References 集合，ReferencesEvents 对象

语法

Sub*object*_ItemAdded(ByVal*item*AsReference)

所需的 item 参数用来指定添加的项目。

说明

ItemAdded 事件发生在 Reference 被加到该 Reference 集合对象中时。

ItemCheck 事件

当 ListBox 控件的 Style 属性设置为 1（复选框），并且 ListBox 控件中一个项目的复选框被选定或者被清除时该事件发生。

应用于

ListBox 控件

语法

PrivateSub*object*_**ItemCheck**(**[***index***AsInteger]**)

ItemCheck 事件语法包含以下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，唯一地标识该列表框中被选中的项

说明

注意 ItemCheck 事件在列表项目为突出显示时并不出现；确切地说，它是在该列表项目的复选框被选定或者被清除时出现。

每当 ListBox 的被选择数组中的元素改变（并且其 Style 属性设置为 1）时，ItemCheck 事件也能有计划地出现。

ItemCheck 事件出现在 Click 事件之前。

试图用 ListBox 控件的 ItemCheck 事件中的 UnloadMe 下载一个窗体导致一般性保护出错(GPF)，例如：

```
PrivateSubForm_Load()
   Dimi%
   Fori=0To20
   List1.AddItemi
   Nexti
   EndSub

PrivateSubList1_ItemCheck(ItemAsInteger)
UnloadMe
EndSub
```

如果按下空格键，或试图使用 CommandButton **在** ListBox **中选择一项时，该示例也会产生** GPF。
建议通过将Unload **语句添加入**CommandButton **或**Menu **控件的**Click **事件的步骤来下载窗体。**

ItemData 属性

返回或设置 ComboBox **或** ListBox **控件中每个项目具体的编号。**

应用于

ComboBox 控件，ListBox 控件

语法

object.**ItemData**(*index*)[=*number*]

ItemData 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
index	对象中指定项目的编号
number	与指定项目相关联的数

说明

ItemData 属性是一个长整型数的数组，它有与控件的 List 属性相同数目的项目。可以用与每一项相关的数来标识它们。例如，在 ListBox 控件中，可以用雇员身份号来标识每一个雇员的名字。填 ListBox 时，也要将雇员号填入 ItemData 相应的元素中。

ItemData 常常用做与 ListBox 控件中项目相关的数据结构数组的索引。

注意利用 AddItem 方法在列表中插入一个项目时，在 ItemData 数组中也会自动插入一项。但是其值不会重新初始化为 0；它保持列表在插入项目之前该位置的值。用 ItemData 属性时，一定要在向列表中加入新项时，设置它的值。

请参阅

AddItem 方法，RemoveItem 方法，List 属性，ListIndex 属性，NewIndex 属性，Selected 属性，TopIndex 属性

示例

这个例子用员工的名字填充 ListBox 控件，并用员工的代号填充 ItemData 属性数组，并用 NewIndex 属性使代号与排序列表同步。

当用户做选择时，Label 控件显示选项的名字和代号。要试用此例，r 把代码粘贴到包含 ListBox 和 Label 的窗体的声明部分。设置 ListBox 的 Sorted 属性为 True，然后按 F5 键，并单击 ListBox。

```
PrivateSubForm_Load()
   '以排序顺序将相应的项目填充 List1 和 ItemData 数组.
   List1.AddItem"JudyPhelps"
   List1.ItemData(List1.NewIndex)=42310
   List1.AddItem"ChienLieu"
   List1.ItemData(List1.NewIndex)=52855
   List1.AddItem"MauroSorrento"
   List1.ItemData(List1.NewIndex)=64932
   List1.AddItem"CynthiaBennet"
   List1.ItemData(List1.NewIndex)=39227
EndSub

PrivateSubList1_Click()
   '追加员工数字和员工名字.
   Msg=List1.ItemData(List1.ListIndex)&""
   Msg=Msg&List1.List(List1.ListIndex)
   Label1.Caption=Msg
EndSub
```

ItemReloaded 事件

在重新加载部件后出现。

应用于

VBComponentsEvents 对象

语法

PrivateSubobject_ItemReloaded(vbcomponentAsVBComponent)

ItemReloaded 事件的语法有下面这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbcomponent	VBComponent 对象，它代表重新加载的部件

ItemRemoved 事件

该事件发生在工程、控件或部件从当前工程删除之后。

应用于

References 集合，VBComponents 集合，SelectedVBControls 集合，SelectedVBControlsEvents 对象，VBComponentsEvents 对象,VBControls 集合,VBControlsEvents 对象,VBProjectsEvents 对象，ReferencesEvents 对象，VBProjects 集合

语法

Sub*object*_**ItemRemoved**(*vbcontrol***AsVBControl**)

Sub*object*_**ItemRemoved**(*vbproject***AsVBProject**)

Sub*object*_**ItemRemoved**(*vbcomponent***As VBComponent**)

ItemRemoved 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbproject	VBProject 对象，指出被删除的工程名
vbcomponent	VBComponent 对象，指出被删除的部件名
vbcontrol	VBControl 或 SelectedVBControl 对象，指出被删除的控件名

ItemRemoved 事件（VBA 外接程序对象模型）

发生在从工程中删除一个引用之后。

应用于

References 集合，ReferencesEvents 对象

语法

Sub*object*_**ItemRemoved**(**ByVal***item***As Reference**)

其中 *item* 参数用来指定被删除的引用。

ItemRenamed 事件

该事件发生在工程、控件或部件在当前工程中被重新命名之后。

应用于

VBComponentsEvents 对象，VBControls 集合，VBControlsEvents

对象，VBProjectsEvents 对象，VBProjects 集合

语法

Sub*object*_**ItemRenamed**(*vbproject***AsVBProject**)

Sub*object*_**ItemRenamed**(*vbcomponent***AsVBComponent**)

Sub*object*_**ItemRenamed**(*vbcontrol***AsVBControl**)

ItemRenamed 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbproject	VBProject 对象，指出被重新命名的工程名
vbcomponent	VBComponent 对象，指出被重新命名的部件名
vbcontrol	VBControl 对象，指出被重新命名的控件名

Items 方法

返回一个包含 Dictionary 对象中所有条目的数组。

应用于

Dictionary 对象

语法

object.**Items**

object 始终是一个 **Dictionary** 对象的名字。

说明

下面的代码举例说明了 Items 方法的使用：

```
Dima,d,i                    '创建一些变量
```

```
Setd=CreateObject("Scripting.Dictionary")
d.Add"a","Athens"                    '添加一些关键字和条目。
d.Add"b","Belgrade"
d.Add"c","Cairo"
a=d.Items          '取得条目
Fori=0Tod.Count-1                    '重复数组
Printa(i)                  '打印条目
Next
...
```

请参阅

Add 方法(Dictionary)，Exists 方法，Keys 方法，Remove 方法，RemoveAll 方法

ItemSelected 事件

该事件发生在“工程”窗口中的部件或打开的设计器窗口中的部件被单击时。

应用于

VBComponentsEvents 对象

语法

Sub*object*_**ItemSelected**(*vbcomponent***AsVBComponent**)

ItemSelected 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbcomponent	VBComponent 对象，指出被选定的部件名

Join 函数

返回一个字符串，该字符串是通过连接某个数组中的多个子字符串而创建的。

语法

Join(*list*[,*delimiter*])

Join 函数语法有如下几部分：

部分	描述
list	必需的。包含被连接子字符串的一维数组
delimiter	可选的。在返回字符串中用于分隔子字符串的字符。如果忽略该项，则使用空格("")来分隔子字符串。如果 delimiter 是零长度字符串("")，则列表中的所有项目都连接在一起，中间没有分隔符

请参阅

Split 函数

KeepTogether 属性

返回或设置一个值，指定一个部分中的文本是否保持在一起（不被分页符分开）。

应用于

Section 对象(DataReportDesigner)

语法

object.**KeepTogether**[=*boolean*]

KeepTogether 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
boolean	可选的。一个布尔表达式，指定是否开始一个新页，如在设置值中所示

设置值

对 boolean 的设置如下：

设置值	描述
True	出现一个分页
False	（缺省的）分页不出现

Key 属性

在 Dictionary 对象中设置 key。

应用于

Dictionary 对象

语法

object.**Key**(*key*)=newkey

Key 属性有以下几部分：

部分	描述
object	必要的参数。总为 Dictionary 对象句
String	必要的参数。改变 key 值
newkey	替换指定的 key 为新值

说明

如果在改变 key 时找不到 key，则发生实时错误。

请参阅

CompareMode 属性，Count 属性，Item 属性

KeyDown、KeyUp 事件

这些事件是当一个对象具有焦点时按下(KeyDown)或松开(KeyUp)一个键时发生的（要解释 ANSI 字符，应使用 KeyPress 事件)。

应用于

ListView控件,Slider控件,TabStrip控件,DBCombo 控件,DBList

控件，RichTextBox 控件，PropertyPate 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，HScrollBar，VScrollBar 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLEContainer 控件

语法

PrivateSubForm_KeyDown(*keycode***AsInteger**,*shift***AsInteger**)

PrivateSub*object*_**KeyDown**([*index***AsInteger**,]*keycode***AsInteger**,*shift* **AsInteger**)

PrivateSubForm_**KeyUp**(*keycode***AsInteger**,*shift***AsInteger**)

PrivateSub*object*_**KeyUp**([*index***AsInteger**,]*keycode***AsInteger**,*shift***AsInteger**)

KeyDown 和 KeyUp 事件包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	是一个整数，它用来唯一标识一个在控件数组中的控件
keycode	是一个键代码，诸如 vbKeyF1（F1 键）或 vbKeyHome（HOME 键）。要指定键代码，可使用对象浏览器中的 VisualBasic(VB)对象库中的常数
shift	是在该事件发生时响应 SHIFT,CTRL 和 ALT 键的状态的一个整数。Shift 参数是一个位域，它用最少的位响应 SHIFT 键（位 0）、CTRL 键（位 1）和 ALT 键（位 2）。这些位分别对应于值 1、2 和 4。可通过对一些、所有或无位的设置来指明有一些、所有或零个键被按下。例如，如果 CTRL 和 ALT 这两个键都被按下，则 shift 的值为 6

说明

对于这两个事件来说，带焦点的对象都接收所有击键。一个窗体只有在不具有可视的和有效的控件时才可以获得焦点。虽然 KeyDown 和 KeyUp 事件可应用于大多数键，它们最经常地还是应用于：

扩展的字符键如功能键等。

定位键

键盘修饰键和按键的组合。

区别数字小键盘和常规数字键。

在需要对按下和松开一个键都响应时，可使用 KeyDown 和 KeyUp 事件过程。

下列情况不能引用 KeyDown 和 KeyUp 事件：

窗体有一个 CommandButton 控件，并且 Default 属性设置为 True 时的 ENTER 键。

窗体有一个 CommandButton 控件，并且 Cancel 属性设置为 True 时的 ESC 键。

TAB 键。

KeyDown 和 KeyUp 用两种参数解释每个字符的大写形式和小写形式：keycode—显示物理的键（将 A 和 a 作为同一个键返回）和 shift—显示 shift+key 键的状态而且返回 A 或 a 其中之一。

如果需要测试 shift 参数，可使用该参数中定义各位的 shift 常数。该常数有下列值：

常数	值	描述
VbShiftMask	1	SHIFT 键的位屏蔽
VbCtrlMask	2	CTRL 键的位屏蔽
VbAltMask	4	ALT 键的位屏蔽

该常数用作位屏蔽。它可被用来测试任何键组合。

测试一个条件时，首先将每个结果分配给一个临时整数变量，然后将 shift 与一个位屏蔽进行对比。如下例，可用 And 运算符和 shift 参数一起来测试条件是否大于 0。该条件说明该修正键被按下：

```
ShiftDown=(ShiftAndvbShiftMask)>0
```

可按此例在一个过程中测试任何条件的组合：

IfShiftDownAndCtrlDownThen

注意如果 KeyPreview 属性被设置为 True，则一个窗体先于该窗体上的控件接收到此事件。可用 KeyPreview 属性来创建全局键盘处理例程。

请参阅

KeyPreview 属性

示例

本例演示一个响应 F2 以及与 ALT，SHIFT 和 CTRL 相联的组合键类属键盘处理程序。键常数列在对象浏览器中的VisualBasic(VB)对象库里。要尝试这个例子，可将代码粘贴到一个含 1 个 TextBox 控件的窗体的声明部分，然后按 F5 键和 ALT+F2,SHIFT+F2 和 CTRL+F2 组合键。

```
PrivateSubText1_KeyDown(KeyCodeAsInteger,ShiftAsInteger)
   DimShiftDown,AltDown,CtrlDown,Txt
   ShiftDown=(ShiftAndvbShiftMask)>0
   AltDown=(ShiftAndvbAltMask)>0
   CtrlDown=(ShiftAndvbCtrlMask)>0
   IfKeyCode=vbKeyF2Then     '显示键组合.
   IfShiftDownAndCtrlDownAndAltDownThen
       Txt="SHIFT+CTRL+ALT+F2."
   ElseIfShiftDownAndAltDownThen
       Txt="SHIFT+ALT+F2."
```

```
    ElseIfShiftDownAndCtrlDownThen
        Txt="SHIFT+CTRL+F2."
    ElseIfCtrlDownAndAltDownThen
        Txt="CTRL+ALT+F2."
    ElseIfShiftDownThen
        Txt="SHIFT+F2."
    ElseIfCtrlDownThen
    Txt="CTRL+F2."
    ElseIfAltDownThen
        Txt="ALT+F2."
    ElseIfSHIFT=0Then
        Txt="F2."
    EndIf
    Text1.Text="Youpressed"&Txt
    EndIf
EndSub
```

KeyPress 事件

此事件当用户按下和松开一个 ANSI 键时发生。

应用于

ListView控件,Slider控件,TabStrip控件,DBCombo 控件,DBList 控件，RichTextBox 控件，PropertyPage 对象，UserControl 对

象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，HScrollBar，VScrollBar 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLEContainer 控件

语法

PrivateSubForm_KeyPress(*keyascii***AsInteger**)

PrivateSub*object*_**KeyPress**([*index***AsInteger**,]*keyascii***AsInteger**)

KeyPress 事件语法包含下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，它用来唯一标识一个在控件数组中的控件
keyascii	是返回一个标准数字 ANSI 键代码的整数。*Keyascii* 通过引用传递，对它进行改变可给对象发送一个不同的字符。将 *keyascii* 改变为 0 时可取消击键，这样一来对象便接收不到字符

说明

具有焦点的对象接收该事件。一个窗体仅在它没有可视和有效的控件或 KeyPreview 属性被设置为 True 时才能接收该事件。一个 KeyPress 事件可以引用任何可打印的键盘字符，一个来自标准字母表的字符或少数几个特殊字符之一的字符与 CTRL 键的组合，以及 ENTER 或 BACKSPACE 键。KeyPress 事件过程在截取 TextBox 或

ComboBox 控件所输入的击键时是非常有用的。它可立即测试击键的有效性或在字符输入时对其进行格式处理。改变 keyascii 参数的值会改变所显示的字符。

可使用下列表达式将 keyascii 参数转变为一个字符：

```
Chr(KeyAscii)
```

然后执行字符串操作，并将该字符反译成一个控件可通过该表达式解释的 ANSI 数字：

```
KeyAscii=Asc(char)
```

应当使用 KeyDown 和 KeyUP 事件过程来处理任何不被 KeyPress 识别的击键，诸如：功能键、编辑键、定位键以及任何这些键和键盘换档键的组合等。与 KeyDown 和 KeyUp 事件不同的是，KeyPress 不显示键盘的物理状态，而只是传递一个字符。

KeyPress 将每个字符的大、小写形式作为不同的键代码解释，即作为两种不同的字符。而 KeyDown 和 KeyUp 用两种参数解释每个字符的大写形式和小写形式：keycode—显示物理的键（将 A 和 a 作为同一个键返回）和 shift—指示 shift+key 键的状态而且返回 A 或 a 其中之一。

如果 KeyPreview 属性被设置为 True，窗体将先于该窗体上的控件接收此事件。可用 KeyPreview 属性来创建全局键盘处理例程。

注意 CTRL+@的键盘组合的 ANSI 编号是 0。因为 VisualBasic 将一个零值的 keyascii 识别为一个长度为零的字符串("")，在应用程序中应避免使用 CTRL+@的组合。

请参阅

Change 事件，KeyDown，KeyUp 事件，KeyPreview 属性，Change 事件(ActiveX 控件)

示例

本例将输入到 TextBox 控件的文本转换为大写。要尝试这个例子，可将代码粘贴到一个包含一个 TextBox 控件窗体的声明部分，然后按 F5 键并在 TextBox 中输入内容。

```
PrivateSubText1_KeyPress(KeyAsciiAsInteger)
    Char=Chr(KeyAscii)
    KeyAscii=Asc(UCase(Char))
EndSub
```

KeyPreview 属性

返回或设置一个值，以决定是否在控件的键盘事件之前激活窗体的键盘事件。键盘事件为：KeyDown、KeyUp 和 KeyPress。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Form 对象，Forms 集合

语法

object.**KeyPreview**[=*boolean*]

KeyPreview 属性语法有以下组成部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指定如何接收事件。Setting 中有其说明

设置值

boolean 的设置值为：

设置值	描述
True	窗体先接收键盘事件，然后是活动控件接收事件
False	（缺省值）活动控件接收键盘事件，而窗体不接收

说明

可以用该属性，生成窗体的键盘处理程序，例如，应用程序利用功能键时，需要在窗体级处理击键，而不是为每个可以接收击键事件的控件编写程序。

如果窗体中没有可见和有效的控件，它将自动接收所有键盘事件。

若要在窗体级处理键盘事件、而不允许控件接收键盘事件时，在窗体的 KeyPress 事件中设置 KeyAscii 为 0，在窗体的 KeyDown 事件中设置 KeyCode 为 0。

一些控件能够拦截键盘事件，以致窗体不能接收它们。这样的例子有：CommandButton 控件有焦点时的 ENTER 键、以及焦点在 ListBox 控件上时的方向键。

请参阅

KeyDown，KeyUp 事件，KeyPress 事件

示例

这个例子在 KeyDown 事件中创建一个窗体键盘处理器。前四个功能键显示不同的信息。要试用此例，先把代码粘贴到窗体的声明部分，然后按 F5 键。当程序运行时，按下前四个(F1-F4)功能键中的任意一个。

```
PrivateSubForm_Load()
   KeyPreview=True
EndSub

PrivateSubForm_KeyDown(KeyCodeAsInteger,ShiftAsInteger)
   SelectCaseKeyCode
        CasevbKeyF1:MsgBox"F1isyourfriend."
        CasevbKeyF2:MsgBox"F2couldcopytext."
        CasevbKeyF3:MsgBox"F3couldpastetext."
        CasevbKeyF4:MsgBox"F4couldformattext."
   EndSelect
EndSub
```

Keys 方法

返回一个数组，该数组包含一个 Dictionary 对象中的全部已有的关键字。

应用于

Dictionary 对象

语法

object.**Keys**

object 始终是一个 Dictionary 对象的名字。

说明

下面的代码举例说明了 Keys 方法的使用。

```
Dima,d,i                    '创建一些变量
Setd=CreateObject("Scripting.Dictionary")
d.Add"a","Athens"                   '添加一些关键字和条目。
d.Add"b","Belgrade"
d.Add"c","Cairo"
a=d.keys            '取得关键字
Fori=0Tod.Count–1                   '重复数组
Printa(i)                   '打印关键字
Next
...
```

请参阅

Add 方法(Dictionary)(FileSystemObject 对象)，Items 方法，Remove 方法(FileSystemObject 对象)，RemoveAll 方法

Kill 语句

从磁盘中删除文件。

语法

Kill*pathname*

必需的 *pathname* 参数是用来指定一个文件名的字符串表达式。*pathname* 可以包含目录或文件夹以及驱动器。

说明

在 MicrosoftWindows 中，Kill 支持多字符(*)和单字符(?)的通配符来指定多重文件。

请参阅

RmDir 语句

示例

本示例使用 Kill 语句将磁盘中的文件删除。

```
'假设 TESTFILE 是一数据文件。
Kill"TestFile" '删除文件。

'将当前目录下所有*.TXT 文件全部删除。
Kill"*.TXT"
```

KillDoc 方法

用于立即终止当前打印作业。

应用于

Printer 对象，Printers 集合

语法

object.**KillDoc**

object 所在处代表一个对象表达式，其值为“应用于”列表中的一个对象。

说明

如果操作系统的打印管理器正在处理该打印作业（打印管理器正在运行并且允许后台打印），那么 KillDoc 将删除当前打印作业并且使打印机不接收任何信息。

如果打印管理器不是正在处理该打印作业（没有选用后台打印），部分或全部数据可能在 KillDoc 生效前已发送到打印机。此时，打印机驱动程序将尽可能使打印机复位并终止该打印作业。

请参阅

EndDoc 方法

示例

本示例使用 KillDoc 方法来终止当前打印工作。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   Fori=1To40
       Printer.CurrentX=1440    '设置左边距。
```

```
        Printer.CurrentY=(i*300) '进页到下一行。
        Printer.Print"Thisisline"&Str$(i)&"oftext."
        OnErrorResumeNext          '捕获任何打印机错误。
        Ifi=26Then
            Printer.KillDoc        '立即终止打印作业。
            Printer.EndDoc
            End
        EndIf
    Nexti
EndSub
```

Label 控件

Label 控件是图形控件，可以显示用户不能直接改变的文本。

语法

Label

说明

可以编写代码来改变 Label 控件显示的文本，以响应运行时的事件。例如，如果一个应用程序要几分钟时间作改动，则可以显示 Label 中的进程状态信息。还可以使用 Label 来标识控件，例如 TextBox 控件没有自己的 Caption 属性，这时就可以使用 Label 来标识这个控件。

如果希望 Label 显示可变长度的行或变化的行数，就要设置

AutoSize 和 WordWrap 属性。

在 DDE 对话中，Label 控件还可以起接收端的作用。设置 LinkTopic 属性建立链接，设置 LinkItem 属性指定对话项，设置 LinkMode 属性激活连接。这些属性全部显示后，VisualBasic 就要初始化对话，如果不能初始化，就会显示消息。

如果要将 Label 的 Caption 属性中的一个字符定义为访问键，则应将 UseMnemonic 属性设置为 True。在 Label 控件中定义访问键时，就可以按 ALT+设定的字符来把焦点移动到 TAB 键顺序中的下一个控件。

属性

Label，DataMember 属性，DataFormat 属性，RightToLeft 属性，OLEDrag 方法，OLEDropMode 属性，BackColor，ForeColor 属性，BackStyle 属性，BorderStyle 属性，FontBold，FontItalic，FontStrikethru，FontUnderline 属性，FontName 属性，FontSize 属性，Height，Width 属性，Left，Top 属性，TabIndex 属性，Tag属性,Visible属性,Alignment属性,AutoSize 属性,DragIcon 属性,DragMode 属性,LinkItem 属性,LinkMode 属性,LinkTimeout 属性，LinkTopic 属性，MouseIcon 属性，MousePinter 属性，WordWrap 属性，Appearance 属性，UseMnemonic 属性，Caption 属性，Enabled 属性，Enabled 属性，Index 属性(ControlArray)，Name 属性,Parent 属性,Font 属性,Container 属性,ToolTipText 属性，DataChanged 属性，DataField 属性，DataSource 属性，

WhatsThisHelpID 属性

方法

Refresh 方法，Drag 方法，LinkExecute 方法，LinkPoke 方法，LinkRequest 方法，LinkSend 方法，Move 方法，ZOrder 方法，OLEDrag 方法，ShowWhatsThis 方法

请参阅

TextBox 控件，TabIndex 属性，AutoSize 属性，LinkItem 属性，LinkMode 属性，LinKTopic 属性，WordWrap 属性，Caption 属性，Caption 属性(ActiveX 控件)

Label 控件（数据报表设计器）

Label 控件是一个图形控件，可以用于显示用户不能直接更改的文本。

语法

RptLabel

说明

Label 控件的数据报表设计器版可以在任何应用程序上提供显示文本的方法，类似于标准的 VisualBasic 固有 Label 控件。然而除此基本功能之外，某些标准 Label 控件的属性在数据报表版中是不可用的。

属性

CanGrow 属性，BackColor，ForeColor 属性，BackStyle 属性，

BorderColor 属性,BorderStyle 属性,Height,Width 属性,Left,Top 属性，Visible 属性，Alignment 属性，Caption 属性，Name 属性，Font 属性

LargeChange、SmallChange 属性

LargeChange 一返回和设置当用户单击滚动条和滚动箭头之间的区域时，滚动条控件（HScrollBar 或 VScrollBar）的 Value 属性值的改变量。

SmallChange 一返回或设置当用户单击滚动箭头时，滚动条控件的 Value 属性值的改变量。

应用于

HScrollBar，VScrollBar 控件

语法

object.**LargeChange**[=*number*]

object.**SmallChange**[=*number*]

LargeChange 和 SmallChange 属性语法有以下组成部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	指定 Value 属性改变量的整型值

说明

对这两个属性，都可以指定 1 和 32,767 之间的整数，包括 1 和

32，767。缺省时，每个属性都设置为 1。

基于对象中数据的量，MicrosoftWindows 运行环境自动对 MDIForm 对象、ComboBox 控件和 ListBox 控件按比例设置滚动条的增量。但是，对于 HScrollBar 和 VScrollBar 控件，必须指定这些增量。用 LargeChange 和 SmallChange 设置使用滚动条时，相应的滚动增量。

一般来说，在设计时设置 LargeChange 和 SmallChange 属性。当滚动条必须动态改变时，也可以在运行时用程序对其重新设置。

注意用 Max 和 Min 属性设置 HScrollBar 和 VScrollBar 的最大和最小范围。

请参阅

Slider 控件，Change 事件(ActiveX 控件)，Change 事件，Value 属性，Max，Min 属性(ScrollBar)

示例

这个例子用滚动条移动 PictureBox 控件穿过窗体。要试用此例，先把代码粘贴到包含一小的 PictureBox 控件和 HScrollBar 控件的窗体的声明部分，然后按 F5 键，并单击滚动条。

```
PrivateSubForm_Load()
   HScroll1.Max=100                    '设置最大值.
   HScroll1.LargeChange=20                  '敲击 5 次后穿过.
   HScroll1.SmallChange=5                   '敲击 20 次后穿过.
   Picture1.Left=0                     '图形从左边开始.
   Picture1.BackColor=QBColor(3)                ' 设置图形框的颜
```

色.

EndSub

PrivateSubHScroll1_Change()

'按照滚动条移动图形.

Picture1.Left=(HScroll1.Value/100)*ScaleWidth

EndSub

LastDLLError 属性

返回因调用动态链接库(DLL)而产生的系统错误号。只读。在 Macintosh 中，LastDLLError 总是返回零。

应用于

Err 对象

说明

LastDLLError 属性只适用于由 VisualBasic 代码进行的 DLL 调用。在调用时，被调用的函数通常返回一个表明成功还是失败的代码，同时对 LastDLLError 属性填充数据。请检查 DLL 函数的文档，确定返回值，表明是成功还是失败。只要返回失败代码，VisualBasic 的应用程序就应立即检查 LastDLLError 属性。在设置 LastDLLError 属性时不会有任何例外。

请参阅

Declare 语句, HelpContextID 属性(VBA 外接程序对象模型), Err 对象，Description 属性，HelpContext 属性，HelpFile 属性，

Number 属性，Source 属性

示例

下列代码（您可将之粘帖到一个 UserForm 模块中）会试图调用某个 DLL（动态链接程序库）里的函数。因为程序中故意传入一个会生成错误的参数值（传入一 NULL 指针），该调用会失败，而且 SQL 如果没有在运行的话，便不能取消它的作业。程序接下来会检查该函数的返回值，并将 Err 对象的 LastDLLError 属性值显示出来。在没有 DLL 的操作系统中，LastDLLError 总是返回零。

```
PrivateDeclareFunctionSQLCancelLib"ODBC32.dll"_
(ByValhstmtAsLong)AsInteger

PrivateSubUserForm_Click()
   DimRetVal
   '调用函数，但传入了无效的参数值。
   RetVal=SQLCancel(myhandle&)
   '检查 SQL 的返回值。
   IfRetVal=-2Then
        '显示错误代码。
        MsgBox"Errorcodeis:"&Err.LastDllError
EndIf
EndSub
```

LastUsedPath 属性

该属性返回或设置用于 VisualBasic 中文件对话框的最终路径，如“打开工程”对话框。

应用于

VBE 对象

语法

object.**LastUsedPath**([*pathname*As**String**])

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

LastUsedPath 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
pathname	（可选的）包含用于文件对话框的最终路径名的字符串

LBound 函数

返回一个 Long 型数据，其值为指定数组维数可用的最小下标。

语法

sLBound(*arrayname*[,*dimension*])

LBound 函数的语法包含下面部分：I

部分	描述
arrayname	必需的。数组变量的名称，遵循标准的变量命名约定
dimension	可选的；Variant(Long)。指定返回哪一维的下界。1 表示第一维，2 表示第二维，依此类推。如果省略 *dimension*，就认为是 1

说明

LBound 函数与 UBound 函数一起使用，用来确定一个数组的大小。UBound 用来确定数组某一维的上界。

对具有下述维数的数组而言，LBound 的返回值见下表：

```
DimA(1To100,0To3,-3To4):
```

语句	返回值
LBound(A,1)	1
LBound(A,2)	0
LBound(A,3)	-3

所有维的缺省下界都是 0 或 1，这取决于 OptionBase 语句的设置。使用 Array 函数创建的数组的下界为 0；它不受 OptionBase 的影响。对于那些在 Dim 中用 To 子句来设定维数的数组而言，Private、Public、ReDim 或 Static 语句可以用任何整数作为下界。

请参阅

Dim 语句，OptionBase 语句，Private 语句，Public 语句，ReDim 语句，Static 语句，UBound 函数

示例

该示例使用 LBound 函数来返回数组的指定维数的最小可用下标。可以使用 OptionBase 语句取代数组缺省下标值 0。

```
DimLower
DimMyArray(1To10,5To15,10To20) '声明数组变量。
DimAnyArray(10)
Lower=Lbound(MyArray,1)     '返回 1。
Lower=Lbound(MyArray,3)     '返回 10。
Lower=Lbound(AnyArray) '返回 0 或 1，取决于'OptionBase 的设置。
```

LBound 属性

返回控件数组中的控件的最低有序值。

语法

object.**LBound**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

LBound 属性的设置值与数组中第一个控件的 Index 属性值相等。该值的典型值为 0，因为 VisualBasic 自动地给控件数组中的第一个控件分配为 0 的 Index 值。如果将控件数组中的第一个控件的 Index 值手工地改为别的值（例如 1），那么 LBound 将返回手工分配的 Index 值（在这个例子中为 1）。

请参阅

UBound 属性，Count 属性(VB 集合)，Index 属性(ControlArray)，Count 属性(ActiveX 控件)

示例

该例子为控件数组打印以上提及的两个属性的值。在一个窗体上放一个 OptionButton 控件，并设置其 Index 属性为 0（用来创建一个控件数组）。要试用此例，可以将该代码粘贴到一个窗体的声明部分中，然后按下 F5 键并单击此窗体。

```
PrivateSubForm_Paint()
   StaticFlagFormPaintedAsInteger
   IfFlagFormPainted<>TrueThen              '当窗体进行首次绘画时。
      Fori=1To3
         LoadOption1(i)              '在数组中加入三个选项按钮。
         Option1(i).Top=Option1(i-1).Top+350
         Option1(i).Visible=True
      NextI
      ForI=0to3                          '在选项按钮上放置标题。
         Option1(i).Caption="Option#"&CStr(i)
      NextI
      Option1(0).Value=True                    '选中第一个选项按钮。
      FlagFormPainted=True                '窗体已完成绘画。
   EndIf
EndSub
```

```
PrivateSubForm_Click()
   Print"Controlarray'sCountpropertyis"&Option1().Count
   Print"Controlarray'sLBoundpropertyis"&Option1().LBound
   Print"Controlarray'sUBoundpropertyis"&Option1().UBound
EndSub
```

LCase 函数

返回转成小写的 String。

语法

LCase(*string*)

必要的 *string* 参数可以是任何有效的字符串表达式。如果 *string* 包含 Null，将返回 Null。

说明

只有大写的字母会转成小写；所有小写字母和非字母字符保持不变。

请参阅

UCase 函数

示例

本示例使用 LCase 函数来将某字符串转成全部小写。

```
DimUpperCase,LowerCase
Uppercase="HelloWorld1234" '要输送的字符串。
Lowercase=Lcase(UpperCase) '返回"helloworld1234"。
```

Left 函数

返回 Variant(String)，其中包含字符串中从左边算起指定数量的字符。

语法

Left(*string*,*length*)

Left 函数的语法有下面的命名参数：

部分	说明
string	必需的。字符串表达式其中最左边的那些字符将被返回。如果 *string* 包含 Null，将返回 Null
length	必需的。为 Variant(Long)。数值表达式指出将返回多少个字符。如果为 0，返回零长度字符串("")。如果大于或等于 *string* 的字符数，则返回整个字符串

说明

欲知 string 的字符数，使用 Len 函数。

注意 LeftB 函数作用于包含在字符串中的字节数据。所以 length 指定的是字节数，而不是要返回的字符数。

请参阅

Len 函数，Mid 函数，Right 函数

示例

本示例使用 Left 函数来得到某字符串最左边的几个字符。

```
DimAnyString,MyStr
AnyString="HelloWorld" '定义字符串。
```

```
MyStr=Left(AnyString,1)     '返回"H"。
MyStr=Left(AnyString,7)     '返回"HelloW"。
MyStr=Left(AnyString,20)    '返回"HelloWorld"。
```

Left 属性

返回或设置一个 Single，它以缇为单位指出该窗口左边缘在屏幕中的位置。可读写。

应用于

Window 对象

说明

由 Left 属性所返回的值取决于此窗口是否为链接窗口或连接窗口。

注意只要窗口保持链接或连接，改变链接或连接窗口的 Left 属性设置没有任何效果。

请参阅

Height 属性(VBA 外接程序对象模型)，Top 属性，Width 属性

示例

下列示例使用 Left 及 Top 属性返回某个窗口左上角的坐标值（以缇为单位）。这些属性的设置值在窗口被链接或连接后会有所改变，因为此后它们引用了原始窗口与之链接或连接的 Window 对象。

```
Debug.PrintApplication.VBE.Windows(9).Left
```

Debug.PrintApplication.VBE.Windows(9).Top

Left，Top 属性

Left 一返回或设置对象内部的左边与它的容器的左边之间的距离。

Top 一返回或设置对象的内顶部和它的容器的顶边之间的距离。

应用于

ADOData 控件，DataRepeater 控件，Column 对象，DataList 控件，DBCombo 控件，DBList 控件，RemoteData 控件，Data 控件，CommonDialog 控件，Function 控件(DataReportDesigner)，Image 控件(DataReportDesigner)，Label 控件(DataReportDesigner)，Line 控件(DataReportDesigner)，Shape 控件(DataReportDesigner)，TextBox 控件(DataReportDesigner)，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，Timer 控件，OLEContainer 控件

语法

object.**Left**[=*value*]

object.**Top**[=*value*]

Left 和 Top 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	数值表达式，用于指定距离

说明

对于窗体，Left 和 Top 属性总以缇为单位来表达；对于控件，它们的度量单位决定于它的容器的坐标系统。这些属性值随着用户或程序中移动该对象而改变。对于公共对话框和 Timer 控件，这些属性在运行时无效。

对任一个属性，都可以指定单精度数值。

用 Left、Top、Height 和 Width 属性来完成基于对象外部维数的操作，如移动或改变尺寸。用 ScaleLeft、ScaleTop、ScaleHeight 和 ScaleWidth 来完成基于对象内部尺寸的操作，如绘出或移动包含在该对象中的对象。与比例相关的属性只适用于 Picturebox 控件和 Form 以及 Printer 对象。

示例

这个例子在窗体被加载时将窗体的大小设置为屏幕大小的 75%并使窗体居中。要尝试这个例子，请将代码粘贴到窗体的声明部分，然后按 F5 键并单击窗体。

```
PrivateSubForm_Click()
   Width=Screen.Width*.75   '设置窗体的宽度。
   Height=Screen.Height*.75 '设置窗体的高度。
```

```
    Left=(Screen.Width-Width)/2  '在水平方向上居中。
    Top=(Screen.Height-Height)/2 '在垂直方向上居中。
EndSub
```

LeftMargin，RightMargin 属性

以缇为单位返回或设置左或右边距的宽度。

应用于

DataReport 对象

语法

object.**LeftMargin**[=*number*]

object.**RightMargin**[=*number*]

LeftMargin 和 RightMargin 属性的语法包含如下部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
number	可选的。一个数值表达式，指定了左或右边距的宽度

LegalCopyright 属性

返回或设置一个字符串，该字符串包括运行中的应用程序的合法版权信息。该属性在运行时是只读的。

应用于

APP 对象

语法

object.LegalCopyright

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

在设计时，使用位于“工程属性”对话框中的“生成”选项卡上的“类型”框可设置该属性。

请参阅

Comments 属性，CompanyName 属性，FileDescription 属性，

LegalTrademarks 属性

返回或设置一个字符串，该字符串包括运行中的应用程序的合法商标信息。该属性在运行时是只读的。

应用于

App 对象

语法

object.LegalTrademarks

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

在设计时，使用位于“工程属性”对话框中的“生成”选项卡上的“类型”框可设置该属性。

请参阅

Comments 属性，CompanyName 属性，FileDescription 属性，

Len 函数

返回 Long，其中包含字符串内字符的数目，或是存储一变量所需的字节数。

语法

Len(*string*|*varname*)

Len 函数的语法有下面这些部分：

部分	说明
string	任何有效的字符串表达式。如果 string 包含 Null，会返回 Null
varname	任何有效的变量名称。如果 *varname* 包含 Null，会返回 Null。如果 *varname* 是 Variant，Len 会视其为 *String* 并且总是返回其包含的字符数

说明

两个可能的参数必须有其一（而且只能有其一）。如为用户定义类型，Len 会返回其写至文件的大小。

注意 LenB 函数作用于字符串中的字节数据，如同在双字节字符

集(DBCS)语言中一样。所以 LenB 返回的是用于代表字符串的字节数，而不是返回字符串中字符的数量。如为用户自定义类型，LenB 返回在内存中的大小，包括元素之间的衬垫。对于使用 LenB 的示例代码，请参阅示例主题中的第二个示例。

注意当在用户自定义数据类型中使用变长字符串时，Len 可能不能确定实际存储所需的字节数目。

请参阅

InStr 函数，DataType 提要

示例

本示例使用 Len 函数来得知某字符串的长度（字符数）或某变量的大小（位数）。Type...EndType 程序区块定义一个自定义数据类型 CustomerRecord。如果该数据类型定义在对象类模块中，则必须以关键字 Private 开头（表示为私有）。若定义在常规模块中，Type 定义就可以为 Public。

```
TypeCustomerRecord '定义用户自定义的数据类型。
   IDAsInteger '将此定义放在常规模块中。
   NameAsString*10
   AddressAsString*30
EndType

DimCustomerAsCustomerRecord    '声明变量。
DimMyIntAsInteger,MyCurAsCurrency
```

```
DimMyString,MyLen
MyString="HelloWorld"  '设置变量初值。
MyLen=Len(MyInt)    '返回 2。
MyLen=Len(Customer)     '返回 42。
MyLen=Len(MyString)     '返回 11。
MyLen=Len(MyCur)    '返回 8。
```

在用 ANSI 代表字符串的时候，第二个示例使用 LenB 和用户定义的函数(LenMbcs)字符串中字符的字节数。

```
FunctionLenMbcs(ByValstrasString)




´Where"A"and"B"areDBCSand"C"isSBCS.


'Returns3-3charactersintheString.
MyLen=Len(MyString)
´Returns6-6bytesusedforUnicode.


´Retruns5-5bytesusedforANSI.
```

Let 语句

将表达式的值赋给变量或属性。

语法

[Let]varname=expression

Let 语句的语法包含下面部分：

部分	描述
let	可选的。显式使用 Let 关键字也是一种格式，但通常都省略该关键字
varname	必需的。变量或属性的名称；遵循标准变量命名约定
expression	必需的。赋给变量或属性的值

说明

只有当表达式是一种与变量兼容的数据类型时，该表达式的值才可以赋给变量或属性。不能将字符串表达式的值赋给数值变量，也不能将数值表达式的值赋给字符串变量。如果这样做，就会在编译时间出现错误。

可以用字符串或数值表达式赋值给 Variant 变量，但反过来不一定正确。任何除 Null 之外的 Variant 都可以赋给字符串变量，但只有当 Variant 的值可以解释为某个数时才能赋给数值变量。可以使用 IsNumeric 函数来确认 Variant 是否可以转换为一个数。

小心将一种数值类型的表达式赋给另一种数值类型的变量时，会强制将该表达式的值转换为结果变量的数值类型。

Let 语句可以将一个记录类型的变量赋给属于同一用户定义类型的变量。使用 LSet 语句可以给不同用户自定义类型的记录变量赋值。使用 Set 语句可以将对象引用赋给变量。

请参阅

DataType提要.VainatData 类型,Const 语句,Set 语句,IsNumeric 函数, LSet 语句

示例

该示例使用显式的 Let 语句将表达式的值赋给变量。

```
DimMyStr,MyInt
'下面的变量赋值使用了 Let 语句。
LetMyStr="HelloWorld"
LetMyInt=5
```

下面是没有使用 Let 语句的相同赋值。

```
DimMyStr,MyInt
MyStr="HelloWorld"
MyInt=5
```

LicenseInfo 对象

该对象提供了控件的许可证编号和程序 ID(progID)。

语法

LicenseInfo

说明

在将一个需要许可证编号的控件添加到 Controls 集合的时候，必须先将控件的许可证编号添加到 Licenses 集合中。利用 Add 方法，每对许可证编号和 progID 将成为 LicenseInfo 对象的属性。

要确定每个控件的许可证编号和 progId，可以使用 ForEach 依次处理每个 LicenseInfo 对象，如下所示：

```
DimliAsLicenseInfo
ForEachliinLicenses
   Debug.Printli.ProgID,li.LicenseKey
Next
```

LicenseKey 属性

返回一个控件的许可证编号。

语法

object.**LicenseKey**

object 所在的位置是一个对象表达式，它的值将等于“应用于”列表中的一个对象。

说明

在试图动态地将一个需要许可证的控件添加到 Controls 集合之前，必须先将许可证编号添加到 Licenses 集合中。

Licenses 集合

一个许可证 objectsthatcontainlicensekeyinformation 集合，它是当添加一个得到许可的控件到 Controls 集合时所需要的。

语法

Licenses

说明

如果一个用户控件要求一个许可证关键字，您必须在添加该控件之前把关键字添加到 Licenses 集合。

Licenses 属性

返回指向 Licenses 集合的引用。

语法

object.**Licenses**

object 所在的位置是一个对象表达式，它的值将等于“应用于”列表中的一个对象。

说明

Licenses 集合被用于包含未被引用的控件的许可证号码。在将控件本身添加到 Controls 集合中之前必须先将控件的许可证编号添加到集合中。要添加一个许可证编号，请使用 Licenses 集合的 Add 方法。

Licenses 属性是 Global 对象的属性。由于 Global 对象是自动被引用

的，在使用 Licenses 属性的时候不需要显式地指明对象，如下面的代码所示：

```
Licenses.Add"myProject.myControl","xydsfasfjewfe"
```

Line 控件

Line 控件是图形控件，它显示水平线、垂直线或者对角线。

语法

Line

说明

在设计时，可以使用 Line 控件在窗体上绘制线。在运行时，除了使用 Line 方法外，还可以使用 Line 控件，或者使用后者代替前者。即使 AutoRedraw 属性设置为 False，Line 控件绘制的线也仍会保留在窗体上。可以在窗体、图片框和框架中显示 Line 控件。运行时不能使用 Move 方法移动 Line 控件，但是可以通过改变 X1、X2、Y1 和 Y2 属性来移动它或者调整它的大小。设置 BorderStyle 属性的效果取决于 BorderWidth 属性的设置。如果 BorderWidth 不是 1 而 BorderStyle 不是 0 或 6，则将 BorderStyle 设置成 1。

属性

BorderColor 属性，BorderStyle 属性，BorderWidth 属性，Tag 属性，Visible 属性，DrawMode 属性，X1，Y1，X2，Y2 属性，Index 属性(ControlArray)，Name 属性，Parent 属性，Container 属性

方法

Refresh 方法，ZOrder 方法

请参阅

Move 方法，Frame 控件，PictureBox 控件，BorderStyle 属性，BorderWidth 属性，BorderStyle 属性(ActiveX 控件)

Line 控件（数据报表设计器）

Line 控件是一个图形控件，显示为一条水平的、垂直的或倾斜的线条。

语法

RptLine

说明

Line 控件的数据报表设计器版可以在任何应用程序上显示线条，类似于标准的 VisualBasic 固有 Line 控件。然而，数据报表版使用 LineSlant 属性决定线条的倾斜方向，而不是使用 X、Y 坐标系来决定线条的外观。

属性

LineSlant 属性，BorderColor 属性，BorderStyle 属性，Height，Width 属性，Left，Top 属性，Visible 属性，Name 属性

LineInput#语句

从已打开的顺序文件中读出一行并将字符串变量指定给变量。

语法

LineInput#filenumber,varname

LineInput#语句的语法具有以下几个部分：

部分	描述
filenumber	必需的。任何有效的文件号
varlist	必需的。有效变量或字符串变量名

说明

用 LineInput#读取数据通常是从用 Print#的文件写入。LineInput#语句一次从文件中读一个字符，直到它遇过回车（Chr(13)）或回连返回—换行（Chr(13)+Chr(10)）序列。回车返回—换行序列跳过而不是附加字符串。

请参阅

Input#语句，Chr 函数

示例

本例使用 LineInput#语句从一顺序文件读取线并赋值给一个变量。本例假设 TESTFILE 是具有很少几个样例数据线的文本文件。

```
DimTextLine
```

```
LineInput#1, Textline
Debug.PrintTextLine
Loop
Close#1
```

Line 方法

在对象上画直线和矩形。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Form 对象，Forms 集合

语法

object.**Line** [**Step**](x1, 1) [**Step**](*x2*, *y2*), [*color*], [**B**] [**F**]

Line 方法的语法有以下对象限定符和部分：

部分	描述
object	可选的。对象表达式，其值为“应用于”列表中的对象。如果 object 省略，具有焦点的 Form 作为 *object*
step	可选的。关键字，指定起点坐标，它们相对于由 CurrentX 和 CurrentY 属性提供的当前图形位置
(*x1*,*y1*)	可选的。Single（单精度浮点数），直线或矩形的起点坐标。ScaleMode 属性决定了使用的度量单位。如果省略，线起始于由 CurrentX 和 CurrentY 指示的位置
Step	可选的。关键字，指定相对于线的起点的终点坐标。
(*x2*,*y2*)	必需的。Single（单精度浮点数），直线或矩形的终点坐标
color	可选的。Long（长整型数），画线时用的 RGB 颜色。如果它被省略，则使用 ForeColor 属性值。可用 RGB 函数或 QBColor 函数指定颜色
B	可选的。如果包括，则利用对角坐标画出矩形
F	可选的。如果使用了 B 选项，则 F 选项规定矩形以矩形边框的颜色填充。不能不用 B 而用 F。如果不用 F 光用 B，则矩形用当前的 FillColor 和 FillStyle 填充。FillStyle 的缺省值为 transparent

说明

画连结的线时，前一条线的终点就是后一条线的起点。

线的宽度取决于 DrawWidth 属性值。在背景上画线和矩形的方法取决于 DrawMode 和 DrawStyle 属性值。

执行 Line 方法时，CurrentX 和 CurrentY 属性被参数设置为终点。

这个方法不能用于 With...EndWith 语句块。

请参阅

Circle 方法

示例

这个示例用 Line 方法在窗体上画了几个同心矩形。要运行这个示例，将此代码放入窗体的 General 部分。按 F5 并单击窗体。

```
SubForm_Click()
    DimCX,CY,F,F1,F2,I                  '声明变量。
    ScaleMode=3                         '设置 ScaleMode 为像素。
    CX=ScaleWidth/2                     '水平中点。
    CY=ScaleHeight/2                    '垂直中点。
    DrawWidth=8                         '设置 DrawWidth。
    ForI=50ToOStep-2
        F=I/50                          '执行中间步骤。
        F1=1-F:F2=1+F                   '计算。
        Forecolor=QBColor(IMod15)           '设置前景颜色。
        Line(CX*F1,CY*F1)-(CX*F2,CY*F2),,BF
    NextI
    DoEvents                            '做其它处理。
    IfCY>CXThen                         '设置 DrawWidth。
```

```
        DrawWidth=ScaleWidth/25
    Else
        DrawWidth=ScaleHeight/25
    EndIf
    ForI=0To50Step2 'Setuploop.
        F=I/50                          '执行中间。
        F1=1-F:F2=1+F                   '计算。
        Line(CX*F1,CY)-(CX,CY*F1)       '画左上角。
        Line-(CX*F2,CY)                 '画右上角。
        Line-(CX,CY*F2)                 '画右下角。
        Line-(CX*F1,CY)                 '画左下角。
        Forecolor=QBColor(IMod15)           '每次改变颜色。
    NextI
    DoEvents                            '进行其它处理。
EndSub
```

Line 属性

只读属性，返回一个 TextStream 文件中的当前行号。

应用于

TextStream 对象

语法

object.Line

object 总是一个 TextStream 对象的名字。

说明

文件初次打开后，在写任何东西之前，Line 的值为 1。

请参阅

AtEndOfLine 属性，AtEndOfStream 属性，Column 属性

Lines 方法

返回指定的代码行。

应用于

CodeModule 对象

语法

object.**Lines**(*startline*,*count*)**AsString**

Lines 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
startline	必需的。一个 Long 型数，用来指定要返回的起始行
count	必需的。一个 Long 型数，用来指定要返回的行数

说明

在代码模块中行号是从 1 开始。

请参阅

DeleteLines 方法，Find 方法(VBA 外接程序对象模型)，

GetSelection 方法，InsertLines 方法，CodePane 对象，ProcBodyLine 属性，ProcCountLines 属性，ProcOfLine 属性，ProcStartLine 属性

示例

下列示例使用 Lines 方法返回在某个代码窗格中从第一行到第四行的代码。

```
Debug.PrintApplication.VBE.CodePanes(1).CodeModule.Lines(1,4)
```

Lines 属性

返回一个字符串，该字符串包含行的指定块。

应用于

CodeModule 对象

语法

object.**Lines**(*startline* **As Long**, *count* **As Long**)

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
startline	Long 数据类型，它指定了起始行号
count	Long 数据类型，它指定了突出显示的行数

LineSlant 属性

返回或设置线条倾斜的方向。

应用于

Line 控件(DataReportDesigner)

语法

object.**LineSlant** [=*integer*]

LineSlant 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
integer	可选的。一个数值表达式，指定了线条的倾斜方向，如在设置值中所示

设置值

对 integer **的设置如下：**

常数	值	描述
RptSlantNWSE	0	线条从左上斜向右下（从西北向东南）
RptSlantNESW	1	线条从右上斜向左下（从东北向西南）

LinkClose 事件

此事件在一个 DDE 对话结束时发生。DDE 对话的两个应用程序任何时候都可以终止对话。

应用于

Extender 对象，Form 对象，Forms 集合，Label 控件，MDIForm 对象，PictureBox 控件，TextBox 控件

语法

PrivateSubForm_LinkClose()

PrivateSubMDIForm_LinkClose()

PrivateSub*object*_LinkClose([*index*AsInteger])

LinkClose 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，它用来唯一标识一个在控件数组中的控件

说明

通常，一个 LinkClose 事件过程被用来通知 DDE 对话已经终止。也可包括关于重建一个连接的故障排除信息或何处寻求帮助的信息。为了显示简短的信息，可使用 MsgBox 函数。

请参阅

LinkError 事件，LinkExecute 事件，LinkNotify 事件，LinkOpen 事件，LinkExecute 方法，LinkMode 属性，LinkTopic 属性

LinkedWindowFrame 属性

返回一个代表包含该窗口的框架的 Window 对象，此属性为只读。

应用于

Window 对象

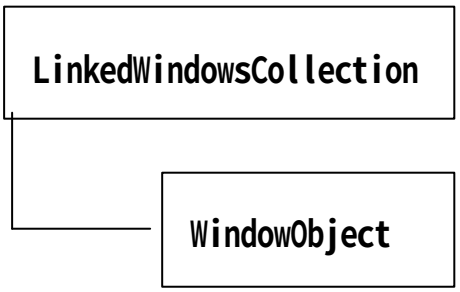
说明

LinkedWindowFrame 属性可访问代表链接窗口框架的对象，该对象具有一些属性，那些属性与该窗口或包含窗口所具有的不同。如果该窗口没有被链接，LinkedWindowFrame 将属性返回 Nothing。

请参阅

Add 方法(VBA 外接程序对象模型)，DesignerWindow 方法，Remove 方法(VBA 外接程序对象模型)，SetFocus 方法，LinkedWindows 集合，Visible 属性，WindowState 属性

LinkedWindows 集合



包含链接窗口框架中的所有链接窗口。

说明

可用 LinkedWindows 集合修改开发环境中的窗口为可连接窗口和链接窗口。

窗口对象的 LinkedWindowFrame 属性返回拥有有效 LinkedWindows 集合的窗口对象。

链接窗口框架包含所有可以链接或停放的窗口。除了代码窗口、

设计器、对象浏览器和“搜索和替换”窗口之外，它包含所有窗口。

如果链接窗口框架的所有窗格都移动到另一窗口，则将破坏无窗格的链接窗口框架。但是，如果将所有窗格从主窗口移走，则不会破坏这些框架。

可用 Visible 属性来查看并设置窗口的可见性。

可用 Add 方法将一个窗口添加到当前链接窗口的集合。当窗口是链接窗口框架中的窗格时，可将该窗口添加到其它链接窗口框架。可用 Remove 方法从当前链接窗口的集合中删除窗口；这就使窗口解除链接或不予停放。

用 LinkedWindows 集合可停放或不停放主窗口框架的窗口。

属性

Count 属性，Parent 属性，VBE 属性

方法

Add 方法(VBA 外接程序对象模型)，Item 方法(VBA 外接程序对象模型)，Remove 方法(VBA 外接程序对象模型)

请参阅

Window 对象，Windows 集合，LinkedWindowFrame 属性，Visible 属性

LinkedWindows 属性

返回一个集合，此集合中的所有链接窗口都包含在一个链接窗口

框架，此属性为只读。

应用于

Window 对象

说明

LinkedWindows 属性是一个访问者属性（即，返回和属性名相同类型的对象的属性名）。

LinkError 事件

当一个 DDE 对话过程中出现错误时，该事件发生。仅在发生了一个 DDE 有关的错误并且没有 VisualBasic 代码被执行来处理这些错误时，其错误号才会作为参数传递。

应用于

Extender 对象，Form 对象，Forms 集合，Label 控件，MDIForm 对象，PictureBox 控件，TextBox 控件

语法

PrivateSubForm_LinkError(*linkerr***AsInteger**)

PrivateSubMDIForm_LinkError(*linkerr***AsInteger**)

PrivateSub*object*_**LinkError**([*index***AsInteger**,]*linkerr***AsInteger**)

LinkError 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
linkerr	如下边返回值中所述，是一个与 DDE 有关错误的错误号
index	一个整数，它用来唯一标识一个在控件数组中的控件

返回值

下列表格列出了返回给 linkerr 参数的所有错误号以及对每个错误的简短说明：

值	描述
1	另一个应用程序按错误的格式已经请求过数据。在 VisualBasic 试图找出另一种应用程序识别的格式时，这种错误可能连续出现几次
6	当发送端窗体上的 LinkMode 属性设置为 0（无）之后，接收端应用程序试图继续 DDE 对话
7	所有源链接都在使用中（每个源最多有 128 个链接）
8	对目标控件来说：一个自动链接或 LinkRequest 方法更新控件中的数据失败 对源窗体来说：目标试图将数据放置到一个控件中去，并且失败
11	无足够的内存给 DDE 使用

说明

可用一个 LinkError 事件过程通知已发生的错误。也可将关于重

建一个连接的恢复问题或故障排除的代码或关于到何处寻求帮助的代码包括在内。为了获得简短消息，可使用 MsgBox 函数。

请参阅

LinkClose 事件，LinkExecute 事件，LinkNotify 事件，LinkOpen 事件，LinkExecute 方法，LinkItem 属性，LinkMode 属性，LinkTopic 属性

示例

本例演示了一个 TextBox 控件，MyTextBox 它能够处理选定的错误。过程根据作为参数 LinkErr 传递的错误号显示一条信息（根据 LinkError 事件主题中的错误列表中改编）。可以直接修改窗体源代码，只需将 Form_LinkError 替换为 MyTextBox_LinkError。

```
PrivateSubMyTextBox_LinkError(LinkErrAsInteger)
DimMsg
SelectCaseLinkErr
   Case1
      Msg="Datainwrongformat."
   Case11
      Msg="OutofmemoryforDDE."
EndSelect
MsgBoxMsg,vbExclamation,"MyTextBox"
EndSub
```

LinkExecute 事件

此事件是当一个 DDE 对话中的命令字符串由一个接收端应用程序发出时而发生的。目标应用程序希望发送端应用程序执行该字符串所描述的操作。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

PrivateSubobject_LinkExecute(cmdstrAsString,cancelAsInteger)

LinkExecute 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
cmdstr	由目标应用程序所发出的命令字符串表达式
cancel	一个告诉目标该命令字符串是否被接受的整数。将 cancel 设置为 0 是通知目标该命令字符串被接受。将 cancel 设置为任何一个非零的值通知目标该命令字符串被拒绝（缺省值被设置为-1，指示的是 cancel）

说明

对 cmdstr 来说没有固定的语法。应用程序如何响应不同的字串完全取决于程序。

如果未创建 LinkExecute 事件过程，VisualBasic 将拒绝来自目标应用程序的命令字符串。

请参阅

LinkClose 事件，LinkError 事件，LinkNotify 事件，LinkOpen 事件，LinkExecute 方法，LinkMode 属性，LinkTopic 属性

示例

本例为 DDE 会话的目标程序定义了一组命令，这些 DDE 会话正是应用程序所要响应的。本例仅用作示范。

```
PrivateSubForm_LinkExecute(CmdStrAsString,CancelAsInteger)
   Cancel=False
   SelectCaseLCase(CmdStr)
   Case"{big}"
       WindowState=2                    '最大化窗口。
   Case"{little}"
       WindowState=1                    '最小化窗口。
   Case"{hide}"
       Visible=False                    '隐藏窗体。
   Case"{view}"
       Visible=True                 '显示窗体。
   CaseElse
       Cancel=True                  '不允许执行。
   EndSelect
EndSub
```

LinkExecute 方法

在一次 DDE 对话过程中将命令字符串发送给发送端应用程序。不支持命名参数。

应用于

Extender 对象，Label 控件，PictureBox 控件，TextBox 控件

语法

object.LinkExecutestring

LinkExecute 方法的语法包含下列部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
string	必需的。一个字符串表达式，它含有源应用程序所识别的命令

说明

string 的实际值根据源应用程序而改变。例如，MicrosoftExcel 和 MicrosoftWordforWindows 接受括在方括号([])中宏命令所组成的命令字符串。要查看源应用程序所接受的命令字符串，请查阅该应用程序的文档。

请参阅

LinkExecute 事件，LinkPoke 方法，LinkRequest 方法，LinkSend 方法

示例

本示例建立一个 MicrosoftExcel 的 DDE 链接，将一些值放置到一个新工作单的第一行的单元里，并按照这些值画图。LinkExecute 向 MicrosoftExcel 发送激活工作单的命令，选择一些值并按照它们画图。要检验此示例，计算机中必须装有 MicrosoftExcel 而且要在 Autoexec.bat 文件的路径中声明。将本例代码粘贴到一个带缺省名 Text1 的 TextBox 控件的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimCmd,I,Q,Row,Z                    '声明变量。
   Q=Chr(34)                           '定义引用标记。
   '创建一个含有MicrosoftExcel宏指令的字串。
   Cmd="[ACTIVATE("&Q&"SHEET1"&Q&")]"
   Cmd=Cmd&"[SELECT("&Q&"R1C1:R5C2"&Q&")]"
   Cmd=Cmd&"[NEW(2,1)][ARRANGE.ALL()]"
   IfText1.LinkMode=vbNoneThen
      Z=Shell("Excel",4)               '启动MicrosoftExcel。
      Text1.LinkTopic="Excel|Sheet1"   '设置连接主题。
      Text1.LinkItem="R1C1"            '设置连接项目。
      Text1.LinkMode=vbLinkManual      '设置连接模式。
   EndIf
   ForI=1To5
      Row=I                            '定义行号。
```

```
        Text1.LinkItem="R"&Row&"C1"      '设置连接项目。
        Text1.Text=Chr(64+I)             '将值放置在 Text 中。
        Text1.LinkPoke                       '将值放入单元。
        Text1.LinkItem="R"&Row&"C2"      '设置连接项目。
        Text1.Text=Row                   '将值放置在 Text 中。
        Text1.LinkPoke                       '将值放入单元。
    NextI
    OnErrorResumeNext
    Text1.LinkExecuteCmd                 '执行MicrosoftExcel 命令。
    MsgBox"LinkExecuteDDEdemowithMicrosoftExcelfinished.",64
End
EndSub
```

LinkItem 属性

在 DDE 与另一个应用程序会话时，返回或设置传给接收端的数据。

应用于

Label 控件，PictureBox 控件，TextBox 控件

语法

object.LinkItem[=*string*]

LinkItem 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	字符串表达式，指定传给目标控件的数据

说明

该属性对应标准 DDE 语法中的 item 参数，标准 DDE 语法以 application，topic 和 item 为参数。要设置该属性，在程序中选择一组可识别的数据作为引用—例如，MicrosoftExcel 中"R1C1"的单元引用。

与 LinkTopic 属性结合使用 LinkItem，确定发送端应用程序目标控件的完整数据链接。设置 LinkMode 属性激活该链接。

只能对用做目标的控件设置 LinkItem。当 VisualBasic 窗体是 DDE 的会话源时，窗体上任何 Label 名、PictureBox 或 TextBox 控件，都可以是目标使用的 application|topic!item 字符串的 item 参数。例如，下面的语法表示从 MicrosoftExcel 到 VisualBasic 应用程序的有效引用。

=VizBasicApplication|MyForm!TextBox1

可以在 MicrosoftExcel 公式条的目标单元中输入前面的语法。

DDE 控件可以同时做为源和目标，但如果目标-源对也是它本身的源-目标对，则会产生无穷循环。例如，TextBox 控件可以同时是 MicrosoftExcel 同一单元的源（通过其父窗体）和目标。当 VisualBasicTextBox 中的数据改变时，数据送到 MicrosoftExcel，MicrosoftExcel 中的单元发生改变，改变的数据送到 TextBox，依次进行，产生循环。

为了避免这种循环，在应用程序之间，用相关联但不同的项目作为目标-源和源-目标的双向链接。例如，在 MicrosoftExcel 中，用相关单元（前提或依赖）来链接工作表和 VisualBasic 控件，不要使用一个项目同时作为目标和源。如果为运行时使用而包含粘贴链接命令，则要为建立的 application|topic 建立文档。

在设计时，用“编辑”菜单的“粘贴连接”命令建立固定的数据链结也同时设置 LinkMode、LinkTopic 和 LinkItem 属性。这样建立的链结与窗体同时保存。每次加载窗体时，VisualBasic 都会试图重新建立会话。

请参阅

LinkMode 属性，LinkTopic 属性

示例

在本例中，每次鼠标单击都导致 MicrosoftExcel 工作表的单元格更新 VisualBasic 的 TextBox 控件的内容。要试验这个例子，启动 MicrosoftExcel 打开一个名为 Sheet1 的新工作表，在第一栏中放入一些数据。在 VisualBasic 中，用 TextBox 控件生成一个表，在声明部分粘贴代码，然后按 F5 运行程序。

```
PrivateSubForm
```

```
´MakeSurethelinkisn'tactive.
Text1.LinkMode=0
´Settheapplicationnameandtopicname.

´Updatetherowinthedataitem.
```

LinkMode 属性

返回或设置用于 DDE 会话的链接类型并同时激活下面的链接：

控件—允许 VisualBasic 窗体上的接收端控件启动会话，它由控件的 LinkTopic 和 LinkItem 属性确定。

窗体—允许目标应用启动与 VisualBasic 发送端窗体的会话，它由目标应用的 application|topic!item 表达式确定。

应用于

Form 对象，Forms 集合，Label 控件，MDIForm 对象，PictureBox 控件，TextBox 控件

语法

object.**LinkMode**[=*number*]

LinkMode 属性有下列组成部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	指定连接类型的整数，“设置值”中有详细描述

设置值

对于 DDE **会话中用做目标的控件** number **的设置值为：**

常数	设置值	描述
vbLinkNone	0	（缺省值）无－无 DDE 交互
vbLinkAutomatic	1	自动－每次链接数据改变，目标控件都要更新
vbLinkManual	2	手动－只有激活 LinkRequest 方法时，才更新目标控件
vbLinkNotify	3	通知－链接数据改变时，会产生 LinkNotify 事件，但是只有在 LinkRequest 方法激活时才会更新目标控件

对作为 DDE **会话源的窗体，**number **的设置值为：**

常数	设置值	描述
vbLinkNone	0	（缺省值）无－没有 DDE 交互。没有目标应用程序能够启动与源窗体的主题会话，没有应用程序能够向窗体放置数据。如果在设计时 LinkMode 为 0（无），在运行时不能将其改变为 1（源）
vbLinkSource	1	源－允许窗体上的任何 Label、PictureBox 或 TextBox 控件为与该窗体建立 DDE 会话的目标应用程序提供数据。如果存在这种链接，VisualBasic 在控件内容改变时会自动提醒目标应用程序。另外，目标应用程序能够向窗体上的 Label、PictureBox、TextBox 控件存放数据。如果设计时 LinkMode 为 1（源），运行时可以将它改为 0（无）也可以再改回来

说明

LinkMode 属性也依赖下列条件：

将目标控件的 LinkMode 设置为非 0 值，将导致 VisualBasic 试图启动由 LinkTopic 和 LinkItem 属性指定的会话。源根据指定的链接类型（自动、手动或提示）更新目标控件。

如果源应用程序终止与 VisualBasic 目标控件的会话，那个控件的 LinkMode 设置值变为 0（无）。

设计时，如果窗体的 LinkMode 设置为缺省值 0（无），运行时不能改变 LinkMode。如果要让窗体作为源，设计时必须设置 LinkMode 属性为 1（源）。然后可以在运行时改变 LinkMode 的值。

注意在设计时，用“编辑”菜单的“粘贴链接”命令建立固定的数据链接，也同时设置 LinkMode、LinkTopic 和 LinkItem 属性。这样建立的链接与窗体同时保存。每次加载窗体时，VisualBasic 都会试图重新建立会话。

请参阅

LinkClose事件，LinkError事件，LinkExecute事件，LinkNotify事件，LinkOpen事件，LinkExecute方法，LinkRequest方法，LinkSend方法，LinkItem属性，LinkTimeout属性，LinkTopic属性

示例

在这个例子中，每一次敲击鼠标都会使 MicrosoftExcel 工作单中的单元更新 VisualBasic 的 TextBox 控件的内容。要试用此例，必须启动 MicrosoftExcel，打开一个新的名叫 Sheet1 的工作单，在第一列中放入一些数据。在 VisualBasic 中，创建一个有 TextBox 控件的窗体。把代码粘贴到声明部分，然后按 F5 键运行该程序。

```
PrivateSubForm_Click()
    DimCurRowAsString
    StaticRow                    '工作单的行数.
```

```
    Row=Row+1                    '增加行.
    IfRow=1Then                  '只第一次.
        '确保连接不是活动的.
        Text1.LinkMode=0
        '设置应用程序的名字和题目名.
        Text1.LinkTopic="Excel|Sheet1"
        Text1.LinkItem="R1C1"    '设置 LinkItem.
        Text1.LinkMode=1         '设置 LinkMode 为自动.
    Else
        '在数据项目中更新行.
        CurRow="R"&Row&"C1"
        Text1.LinkItem=CurRow    '设置 LinkItem.
    EndIf
EndSub
```

LinkNotify 事件

如果接收端控件的 LinkMode 属性被设置为 3（通知），当发送端已经改变了由 DDE 链接定义的数据时，此事件发生。

应用于

Extender 对象，Label 控件，PictureBox 控件，TextBox 控件

语法

PrivateSub*object*_LinkNotify([*index*AsInteger])

LinkNotify 事件语法包括下列部分：

部分	描述
object	是一个对象表达式，其值是“应用于”列表中的一个对象
index	是一个整数，它用来唯一地标识一个在控件数组中的控件

说明

通常，在 LinkNotify 事件中，代码可以向用户发出通知，立即取得新数据或延迟到晚一些时候再取数据。可用 LinkRequest 方法从源中获得该新数据。

请参阅

LinkClose 事件，LinkError 事件，LinkOpen 事件，LinkExecute 方法，LinkRequest 方法，LinkMode 属性，LinkTopic 属性

示例

本例将附加一个称为 Picture1 的 PictureBox 控件，Picture1 的 LinkTopic 和 LinkItem 属性设置在数据源中指定一个图形，其 LinkMode 属性设置为 3(Notify)。当源改变数据时，若 PictureBox 控件在活动窗体上时，过程会立即更新它；否则，过程将设置一个标志变量。该例仅用作示范。

```
PrivateSubPicture1_LinkNotify()
    IfScreen.ActiveFormIsMeThen
    Picture1.LinkRequest                    '图片在活动窗体上，所以更新。
```

```
    Else
        NewDataFlag=True          '假定是一个模块级变量。
    EndIf
EndSub
```

LinkOpen 事件

此事件在一个 DDE 对话正在启动时发生。

应用于

Extender 对象，Form 对象，Forms 集合，Label 控件，MDIForm 对象，PictureBox 控件，TextBox 控件

语法

PrivateSubForm_LinkOpen(*cancel***AsInteger**)

PrivateSubMDIForm_LinkOpen(*cancel***AsInteger**)

P**rivateSub***object*_**LinkOpen**(**[***index***AsInteger,]***cancel***AsInteger**)

LinkOpen 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
cancel	是一个整数，它用来确定该 DDE 对话是否建立。让 *cancel* 设置为 0（缺省值）可建立该对话。将 *cancel* 设置为任何非零值则拒绝对话
index	一个整数，它用来唯一地标识一个在控件数组中的控件

说明

对窗体来说，此事件在一个接收端应用程序正在与该窗体建立一个 DDE 对话时发生。对控件来说，此事件在一个控件正在与一个源应用程序建立一个 DDE 对话时发生。

请参阅

LinkClose 事件，LinkError 事件，LinkExecute 事件，LinkNotify 事件，LinkExecute 方法，LinkMode 属性，LinkTopic 属性

LinkPoke 方法

在 DDE 对话过程中将 Label、PictureBox 或 TextBox 控件的内容传送给发送端应用程序。

应用于

Extender 对象，Label 控件，PictureBox 控件，TextBox 控件

语法

object.**LinkPoke**

object 所在处代表一个对象表达式，其值为“应用于”列表中的一个对象。

说明

object 是包含在 DDE 对话中作为接收端的 Label、PictureBox 或 TextBox 的名称。如果 object 为一个 Label，则 LinkPoke 将 Caption 属性的内容传输到源端。如果 object 为一个 PictureBox，则 LinkPoke 将 Picture 属性的内容传送给源端。如果 object 为一个 TextBox，则 LinkPoke 将 Text 属性的内容传送给源端。

典型情况下，一次 DDE 对话的信息从源端向接收端流动。但是，LinkPoke 允许接收端对象给源端提供数据。并非所有的源应用程序都接受这种方法提供的数据，如果源应用程序不接受数据，则会有一个错误发生。

请参阅

LinkExecute 方法，LinkRequest 方法，LinkSend 方法

示例

本示例建立一个 MicrosoftExcel 的 DDE 链接，将一些值放置到一个新工作单的第一行的单元里，并按照这些值画图。LinkPoke 将画图需要的值发送到 MicrosoftExcel 的工作单上。要检验此示例,计算机中必须装有 MicrosoftExcel 而且要在 Autoexec.bat 文件的路径中声明。将本例代码粘贴到一个带缺省名 Text1 的 TextBox 控件的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimCmd,I,Q,Row,Z                    '声明变量。
   Q=Chr(34)                           '定义引用标记。
   '创建一个含有 MicrosoftExcel 宏指令的字串。
   Cmd="[ACTIVATE("&Q&"SHEET1"&Q&")]"
   Cmd=Cmd&"[SELECT("&Q&"R1C1:R5C2"&Q&")]"
   Cmd=Cmd&"[NEW(2,1)][ARRANGE.ALL()]"
   IfText1.LinkMode=vbNoneThen
      Z=Shell("Excel",4)               '启动
MicrosoftExcel。
```

```
        Text1.LinkTopic="Excel|Sheet1"          '设置连接主题。
        Text1.LinkItem="R1C1"                   '设置连接项目。
        Text1.LinkMode=vbLinkManual             '设置连接模式。
    EndIf
    ForI=1To5
        Row=I    '定义行号。
        Text1.LinkItem="R"&Row&"C1"         '设置连接项目。
        Text1.Text=Chr(64+I)                '将值放置在 Text 中。
        Text1.LinkPoke                          '将值放入单元。
        Text1.LinkItem="R"&Row&"C2"         '设置连接项目。
        Text1.Text=Row                      '将值放置在 Text 中。
        Text1.LinkPoke                          '将值放入单元。
    NextI
    Text1.LinkExecuteCmd                '执行MicrosoftExcel 命令。
    OnErrorResumeNext
    MsgBox"LinkPokeDDEdemowithMicrosoftExcelfinished.",64
End
EndSub
```

LinkRequest 方法

在一次 DDE 对话中请求发送端应用程序更新 Label、PictureBox **或** TextBox **控件中的内容。**

应用于

Extender 对象，Label 控件，PictureBox 控件，TextBox 控件

语法

object.**LinkRequest**

object 所在处代表一个对象表达式，其值为“应用于”列表中的一个对象。

说明

object 是包含在一次 DDE 对话中作为接收端的 Label、PictureBox 或 TextBox 的名称。LinkRequest 引起源应用程序将最当前的数据发送给 object，并且，如果 object 为 Label，则更新 Caption 属性设置，如果 object 为 PictureBox，则更新 Picture 属性设置，如果 object 为 TextBox，则更新 Text 属性设置。

如果 object 的 LinkMode 属性设置为自动（1 或 vbLinkAutomatic），则源应用程序自动更新 object 而不需要 LinkRequest。如果 object 的 LinkMode 属性设置为手工（2 或 vbLinkManual），则只有使用 LinkRequest 时源应用程序才更新 object。如果 object 的 LinkMode 属性设置为通知 Notify（3 或 vbLinkNotify），则源端通过调用 LinkNotify 事件通知接收端已更改数据。然后接收端必须使用 LinkRequest 更新数据。

请参阅

LinkNotify 事件，LinkExecute 方法，LinkPoke 方法，LinkSend 方法，LinkMode 属性

示例

本示例使用 LinkRequest 更新含有 MicrosoftExcel 工作单中值的正文框的内容。要检验此示例，计算机上必须正在运行着 MicrosoftExcel。将一些数据放到缺省工作单(Sheet1.xls)第一列开始的一些单元中。将本例代码粘贴到一个被称为 Text1 的 TextBox 控件的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
    IfText1.LinkMode=vbNoneThen  '测试连接模式。
        Text1.LinkTopic="Excel|Sheet1"   '设置连接主题。
        Text1.LinkItem="R1C1"          '设置连接项目。
        Text1.LinkMode=vbLinkManual  '设置连接模式。
        Text1.LinkRequest              '更新正文框内容。
    Else
        IfText1.LinkItem="R1C1"Then
            Text1.LinkItem="R2C1"
            Text1.LinkRequest          '更新正文框内容。
        Else
            Text1.LinkItem="R1C1"
            Text1.LinkRequest          '更新正文框内容。
        EndIf
    EndIf
EndSub
```

LinkSend 方法

在一次 DDE 对话中将 PictureBox 控件的内容传输到的接收端应用程序。

应用于

Extender 对象，Label 控件，PictureBox 控件，TextBox 控件

语法

object.**LinkSend**

object 位置保存区代表一个对象表达式，该对象一定能在“应用于”列表中找到。

说明

object 必须是 Form 对象中的 PictureBox，它是一次 DDE 对话中的发送端。

当其他一些应用程序与您的应用程序中的 Form 建立自动链接时，VisualBasic 在 Form 中的 TextBox 或 Label 内容改变时会通知它们。但是，当源 Form 中的 PictureBox 的 Pictur 属性设置更改时，VisualBasic 不会自动通知 DDE 接收端应用程序。由于图形的数据量可以非常大，而且由于在每个像素更改时都更新接收端应用程序意义不大，所以 VisualBasic 要求在 PictureBox 的内容更改时使用 LinkSend 方法显式地通知 DDE 的各接收端应用程序。

请参阅

LinkExecute 方法，LinkPoke 方法，LinkRequest 方法，Picture 属性，Picture 属性(ActiveX 控件)

LinkTimeout 属性

返回或设置等待 DDE 响应消息的时间。

应用于

Label 控件，PictureBox 控件，TextBox 控件

语法

object.**LinkTimeout**[=*number*]

LinkTimeout 属性语法包含下面部分

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	数值表达式，指定等待时间

说明

缺省情况下，LinkTimeout 属性设置为 50（相当于 5 秒）。对其它设置可以用十分之一秒为单位进行设定。

DDE 发送端应用的响应时间是不同的。用该属性调整接收端控件等待源应用响应的时间。在源应用需要太长的时间响应时，用 LinkTimeout 可以避免产生 VisualBasic 错误。

注意控件最大等待时间长度为 65，535 个十分之一秒，或大约为 1 小时 49 分钟。如果将 LinkTimeout 设置为-1，则在 DDE 会话中控件以最大时间长度等待响应。用户可以按 ESC 键来强制终止等待过程。

请参阅

LinkClose 事件，LinkOpen 事件，LinkSend 方法，LinkItem 属性，LinkMode 属性，LinkTopic 属性

LinkTopic 属性

对于接收端控件—返回或设置发送端应用程序和主题（使用于该应用程序中的基本数据的分组）。为了指定完整的数据链，应当与 LinkItem 属性一起使用 LinkTopic。

对于发送端窗体—返回或设置在 DDE 对话中发送端窗体需响应的主题。

应用于

Form 对象，Forms 集合，Label 控件，MDIForm 对象，Picture 控件，TextBox 控件

语法

object.**LinkTopic**[=*value*]

LinkTopic 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个指定 DDE 语法元素的字符串表达式

说明

LinkTopic 属性由字符串组成，该字符串用来提供为建立接收端链或者发送端链必须的部分信息。该字符串依赖于正在使用的是接收端控件还是发送端窗体。每个字符串对应于标准 DDE 语法的一

个或多个元素，它包括 application、topic 和 item。

注意当 DDE 链接的标准定义包括 application、topic 和 item 元素时，对于接收端链接到发送端应用程序来说，应用程序内实际使用的语法可作少许的变更。例如，在 MicrosoftExcel 中，可使用下面的语法：

application|topic!item

而在 MicrosoftWordforWindows 中，使用的是：

applicationtopicitem

（不要使用管道符[|]或惊叹号[!]。）

在 VisualBasic 应用程序中，使用的是：

application|topic

惊叹号对于 topic 是隐含的。

接收端控件为了给接收端控件设置 LinkTopic，要使用具有 application|topic 语法的字符串，如下：

application 是请求数据的应用程序的名字，通常是不带扩展名的可执行文件名－例如，Excel（对应 MicrosoftExcel）。

管道字符（|，或字符代码 124）将应用程序与主题分开。

topic 是用在发送端应用程序中的基本数据的分组－例如，MicrosoftExcel 中的工作表。

另外，仅就接收端控件而言，为了指定链接的 item 元素必须设置相关的 LinkItem 属性。一个单元引用，比如 R1C1，相当于 MicrosoftExcel 工作表中的一个项。

发送端

窗体为了给窗体设置 LinkTopic，要将 value 设置为一个适当的窗体标识符。当与该窗体建立 DDE 链接时，接收端应用程序将该字符串用作 topic 参数。虽然该字符串完全是为了发送端窗体在 VisualBasic 中设置 LinkTopic 的所需，但接收端应用程序也需要指定：

接收端应用程序使用的 application 元素，它或者是没有.vbp 扩展名的 VisualBasic 工程文件名（如果正在 VisualBasic 开发环境中运行应用程序）或者是没有.exe 扩展名的 VisualBasic 应用程序文件名（如果正以独立的可执行文件的形式运行应用程序）。App 对象的 EXEName 属性在 VisualBasic 代码中提供该字符串，除非文件名没有改变（EXEName 总是返回磁盘上应用程序的实际文件名；DDE 总是使用在“工程属性”对话框中指定的原始的名字）.

接收端应用程序使用的 item 元素，它与发送端窗体上的为 Label、PictureBox 或 TextBox 控件的 Name 属性设置值对应。

下面的语法是一个例子，它是从 MicrosoftExcel 到 VisualBasic 应用程序的一个有效引用：

=VizBasicApplication|FormN!TextBox1

在 MicrosoftExcel 公式条中能够输入对接收端单元的该引用。

为了激活用 LinkTopic 设置的数据链接，为了指定想要的链接的类型应将 LinkMode 属性设置为适当的非零数值。一般来说，在设置 LinkTopic 之后应设置 LinkMode。对于接收端控件，LinkTopic 的

改变将中断现存的链接并终止 DDE 对话。对于发送端窗体，LinkTopic 的改变将中断使用那个主题的所有接收端链接。由于这些原因，在改变 LinkTopic 之前总是先把 LinkMode 属性设置为 0。在接收端控件 LinkTopic 改变之后，必须再将 LinkMode 设置为 1（自动），2（手动）或 3（通知），以便用新的主题建立一个对话。注意在设计时在“编辑”菜单上用“粘贴链接”命令设置一个永久性的数据链接，也会设置 LinkMode、LinkTopic 和 LinkItem 属性。这将创建一个与窗体一起保存的链接。每次加载窗体时，VisualBasic 就尝试重建该对话。

请参阅

LinkClose 事件，LinkError 事件，LinkOpen 事件，LinkSend 方法，LinkItem 属性，LinkMode 属性

示例

在这个例子中，每一次敲击鼠标都会使 MicrosoftExcel 工作单中的单元更新 VisualBasic 的 TextBox 控件的内容。要试用此例，必须启动 MicrosoftExcel，打开一个新的名叫 Sheet1 的工作单，在第一列中放入一些数据。在 VisualBasic 中，创建一个有 TextBox 控件的窗体。把代码粘贴到声明部分，然后按 F5 键运行该程序。

```
PrivateSubForm_Click()
    DimCurRowAsString
    StaticRow                          '工作单的行数.
    Row=Row+1                          '增加行.
```

```
    IfRow=1Then                    '只第一次.
        '确保连接不是活动的.
        Text1.LinkMode=0
        '设置应用程序的名字和题目名.
        Text1.LinkTopic="Excel|Sheet1"
        Text1.LinkItem="R1C1"        '设置LinkItem.
        Text1.LinkMode=1             '设置 LinkMode 为自动.
    Else
        '在数据项目中更新行.
        CurRow="R"&Row&"C1"
        Text1.LinkItem=CurRow        '设置LinkItem.
    EndIf
EndSub
```

List 属性

返回或设置控件的列表部分的项目。列表是一个字符串数组，数组的每一项都是一列表项目，对 ListBox 和 ComboBox 控件在设计时可以通过属性浏览器得到，对 DirListBox、DriveListBox 和 FileListBox 控件在运行时是只读的，对 ComboBox 和 ListBox 控件在运行时是可读写的。

应用于

ComboBox 控件，DirListBox 控件，DriveListBox 控件，

FileListBox 控件，ListBox 控件

语法

object.**List**(*index*)[=*string*]

List 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
index	列表中具体某一项目的号码
string	字符串表达式，指定列表项目

说明

用该属性可以访问列表项目。

对于除 DirListBox 之外的所有控件，第一个项目的索引为 0 而最后一个项目的索引为 ListCount-1。

对于 DirListBox 控件，索引号序列基于在运行中创建该控件时的当前目录和子目录。当前展开的目录用索引值-1 表示。当前展开目录的上一级目录用绝对值更大一些的负索引值来表示。例如，-2 是当前展开目录的父目录，-3 又是它上一级的目录。当前展开的目录以下的目录的范围是从 0 到 ListCount-1。

起初，ComboBox 和 ListBox 控件包含一空列表。对于文件系统控件，列表内容由控件在运行中创建时存在的条件决定：

DirListBox 一包含所有目录的列表，使用-n 到 ListCount-1 的范围。

DriveListBox 一包含有效的驱动连接列表。

FileListBox 一包含匹配 Pattern 属性的当前展开目录的文件列表。

不包含路径。

List 属性和 ListCount、ListIndex 属性结合起来使用。

对除DirListBox控件之外的所有可用控件，对列表从0到ListCount-1逐个取值，得到列表中的所有项目。对于 DirListBox 控件，对列表从-n 到 ListCount-1 逐个取值得到在当前展开目录中能够见到的所有目录和子目录的列表。在这种情况下 n 是当前展开目录以上的目录级数。

注意要确定在ComboBox 或ListBox控件中显示的项目，可用AddItem方法。要删除项目，用 RemoveItem 方法。若要使项目按字母表排序，在把项目加入到列表中之前将控件的 Sorted 属性设置为 True。

在声明部分用 OptionBase=1 语句不会影响 VisualBasic 控件中元素的编号。第一个元素始终为 0。

当列表索引值超出列表框的实际条数的范围时，则返回一个零长度字符串("")，例如，对于 ComboBox 或 ListBox 控件 List(-1)返回一个零长度字符串。

请参阅

AddItem 方法，RemoveItem 方法，ListCount 属性，ListIndex 属性，Sorted 属性，Pattern 属性

示例

这个例子加载一个带有三明治名称列表的 ComboBox 控件，并显示列表中的第一项。要尝试这个例子，请将代码粘贴到包含 ComboBox 控件的窗体的声明部分，然后按 F5 键。

```
PrivateSubForm_Load()
```

```
    Combo1.AddItem"DenverSandwich"    '对列表添加项。
    Combo1.AddItem"ReubenSandwich"
    Combo1.AddItem"TurkeySandwich"
    Combo1.Text=Combo1.List(0)        '显示列表中的第一项。
EndSub.
```

ListBox 控件

ListBox 控件显示项目列表，从其中可以选择一项或多项。如果项目总数超过了可显示的项目数，就自动在 ListBox 控件上添加滚动条。

如果未选定项目，则 ListIndex 属性值是-1。列表的第一项是 ListIndex0，ListCount 属性值总是比最大的 ListIndex 值大 1。

语法

ListBox

说明

使用 AddItem 或者 RemoveItem 方法可以添加或者删除 ListBox 控件中的项目。对 List、ListCount 和 ListIndex 属性进行设置就可以访问 ListBox 中的项目。也可以在设计时使用 List 属性在列表中增加项目。

属性

ListBox，DataMember 属性，DataFormat 属性，RightToLeft 属性，OLEDragMode 属性，OLEDropMode 属性 BackColor，ForeColor

属性，FontBold，FontItalic，FontStrikethru，FontUnderline 属性，FontName 属性，FontSize 属性，Height，Width 属性，Left，Top 属性，List 属性，ListCount 属性，ListIndex 属性，Sorted 属性，TabIndex 属性，Tag 属性，Text 属性，Visible 属性，Volumns 属性(ListBox)，DragIcon 属性，DargMode 属性，hWnd 属性，ItEmData 属性，MouseIcon 属性，MousePointer 属性，MultisSelect 属性，NewIndex 属性，SelCount 属性，Selected 属性，Style 属性，TabStop 属性，TopIndex 属性，Appearance 属性，Enabled 属性，HelpContextID 属性，Index 属性(ControlArray)，Name 属性，Parent 属性，Font 属性，Container 属性，ToolTipText 属性，DataChanged 属性，DataField 属性，DataSource 属性，IntegralHeight 属性，WhatsThisHelpID 属性

方法

Refresh 方法，SetFocus 方法，AddItem 方法，Clear 方法(Clipboard，ComboBox，ListBox)，Drag 方法，Move 方法，RemoveItem 方法，ZOrder 方法，OLEDrag 方法，ShowWhatsTHis 方法

请参阅

AddItem 方法，RemoveItem 方法，List 属性，ListCount 属性，ListIndex 属性

ListCount 属性

返回控件的列表部分项目的个数。

应用于

ComboBox 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，ListBox 控件

语法

object.**ListCount**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

ListCount 对每个控件提供具体的信息：

ComboBox 和 ListBox 控件—列表中的项目数。

DirListBox 控件—当前目录中子目录的个数。

DriveListBox 控件—连接的驱动器个数。

FileListBox 控件—当前目录中匹配 Pattern 属性设置的文件个数。

如果没有选择项目，ListIndex 属性值为-1。列表中的第一项是 ListIndex=0，并且 ListCount 始终比最大的 ListIndex 值大 1。

请参阅

AddITem 方法，RemoveItem 方法，List 属性，ListIndex 属性，Pattern 属性

示例

这个例子将打印机字体列表装载到一个 ComboBox 控件中，显示列表中的第一项，然后输出字体的总数。每次单击命令按钮都将列表中的所有项改变为大写或小写。要尝试这个例子，请将代码

粘贴到包含一个 ComboBox 控件(Style=2)和一个 CommandButton 控件的窗体的声明部分,然后按 F5 键并单击 CommandButton。

```
PrivateSubForm_Load()
    DimI                                    '声明变量。
    AutoRedraw=True                         '设置 AutoRedraw。
    ForI=0ToPrinter.FontCount-1             '将字体名字放入列表中。
        Combo1.AddItemPrinter.Fonts(I)
    NextI
    Combo1.ListIndex=0                      '设置文本为第一项。
    '在窗体中输出 ListCount 信息。
    Print"Numberofprinterfonts:";Combo1.ListCount
EndSub
PrivateSubCommand1_Click()
    StaticUpperCase
    DimI                                    '声明变量。
    ForI=0ToCombo1.ListCount-1              '在列表中循环。
        IfUpperCaseThen
            Combo1.List(I)=UCase(Combo1.List(I))
        Else
            Combo1.List(I)=LCase(Combo1.List(I))
        EndIf
    NextI
    UpperCase=NotUpperCase                  '改变大小写。
```

```
EndSub
```

ListIndex 属性

返回或设置控件中当前选择项目的索引在设计时不可用。

应用于

ComboBox 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Listbox 控件

语法

object.**ListIndex**[=*index*]

ListIndex 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
index	数值表达式，指定当前项目的索引，“设置值”中有详细说明

设置值

index 的设置值为：

设置值	描述
-1	（ComboBox、DirListBox 和 DriveListBox 控件的缺省值）表示当前没有选择项目；对于 ComboBox 控件，表示用户向文本框部分输入了新文本；对于 DirListBox 控件，表示当前路径的索引。对 DriveListBox 控件，表示在运行时创建该控件时的当前驱动器的索引
n	（FileListBox 和 ListBox 控件的缺省值）表明当前选择项目的索引

说明

表达式 List(List1.ListIndex)返回当前选择项目的字符串。

列表中的第一项是 ListIndex=0，ListCount 始终比最大的 ListIndex 值大 1。

对于可以做多重选择的控件，该属性的行为取决于所选择项目的个数。如果只选择了一个项目，ListIndex 返回该项目的索引。在多重选择中，ListIndex 返回包含在焦点矩形内的项目的索引，而不管该项目是否被选。

请参阅

AddItem 方法，RemoveItem 方法，List 属性，ListCount 属性，Drive 属性，Path 属性

示例

这个例子在 ListBox 控件中显示三个演员的名字，在 Label 控件中

显示被选中的演员所对应的薪金。要尝试这个例子，请将代码粘贴到包含一个 ComboBox 控件和一个 Label 控件的窗体的声明部分，然后按 F5 键并在 ComboBox 中选择一个名字。

```
DimPlayer(0To2)                    '说明两个数组的大小。
DimSalary(0To2)
PrivateSubForm_Load()
   DimI                            '声明变量。
   AutoSize=True
   Player(0)="MiggeyMcMoo"  '在数组中输入数据。
   Player(1)="AlfHinshaw"
   Player(2)="WooferDean"
   Salary(0)="$234,500"
   Salary(1)="$158,900"
   Salary(2)="$1,030,500"
   ForI=0To2                       '在列表中添加名字。
       Combo1.AddItemPlayer(I)
   NextI
   Combo1.ListIndex=0              '显示列表中的第一项。
EndSub

PrivateSubCombo1_Click()
   '显示名字所对应的薪金。
```

```
    Label1.Caption=Salary(Combo1.ListIndex)
EndSub
```

Load 事件

此事件是在一个窗体被装载时发生。当使用 Load 语句启动应用程序，或引用未装载的窗体属性或控件时，此事件发生。

应用于

PropertyPage 对象，Form 对象，Forms 集合，MDIForm 对象

语法

```
PrivateSubForm_Load()
PrivateSubMDIForm_Load()
```

说明

通常，Load 事件过程用来包含一个窗体的启动代码—例如，指定控件缺省设置值，指明将要装入 ComboBox 或 ListBox 控件的内容，以及初始窗体级变量等。

Load 事件是在 Initialize 事件之后发生。

当在代码中引用一个未装载窗体的属性时，该窗体自动被装载但不自动成为可视窗体，除非 MDIChild 属性被设置为 True。如果一个 MDIForm 对象未被装载而一个 MDI 子窗体却被装载，则 MDIForm 和该子窗体都自动被装载并且都将成为可视的窗体。除非使用 Show 方法或将 Visible 属性设置为 True，否则其它窗体都不能显示出来。

MDIFormLoad 事件中的下列代码可自动装载一个 MDI 子窗体（假设 Form1 本身的 MDIChild 属性被设置为 True）：

```
DimNewFormAsNewForm1
NewForm.Caption="NewForm"  '.按引用装载窗体。
```

由于所有子窗体在装载时都可视，对 Caption 属性的引用将装载该窗体并使其可视。

注意当为诸如 Activate,GotFocus,Paint 和 Resize 等相关事件创建过程时，要确保它们的操作不冲突，而且它们也不会导致循环的事件。

请参阅

Activate，Deactivate 事件，QueryUnload 事件，Unload 事件，Load 语句，Unload 语句

示例

本例演示当一个窗体装载完毕时，向一个 ComboBox 控件中装入数据项。要尝试这个例子，可将代码粘贴到一个包含有一个 ComboBox 控件窗体的声明部分，然后按 F5 键。

```
PrivateSubForm_Load()
   Combo1.AddItem"Mozart"        '向列表中添加项。
   Combo1.AddItem"Beethoven"
   Combo1.AddItem"Rock'nRoll"
   Combo1.AddItem"Reggae"
```

```
    Combo1.ListIndex=2              '设置缺省选择。
EndSub
```

Load 事件(DHTMLPage)

当浏览器在应用程序中加载一个 HTML 页面时发生。

应用于

DHTMLPage 对象

语法

PrivateSub*object*_Load

object 所在处表示一个对象表达式，其值为“应用于”列表中的一个对象。

说明

根据页面是否设置为异步加载，该事件有两种不同的行为。

当异步加载时，当页面上的第一个元素创建后就激发 Load 事件。

同步加载时，在所有元素都创建后激发 Load 事件。

当异步运行时（AsyncLoad 属性设置为 True 时），编程人员可以使用该事件作为一个通知，即所有元素都已经加载到页面。

Load 语句

把窗体或控件加载到内存中。

语法

Load*object*

object 所在处是要加载的 Form 对象、MDIForm 对象或控件数组元素的名称。

说明

除非在加载窗体时不需要显示窗体，否则对于窗体不需要使用 Load 语句。在窗体还未被加载时，对窗体的任何引用（在 Set 或 If...TypeOf 语句以外）会自动加载该窗体。例如，Show 方法在显示窗体前会先加载它。一旦窗体被加载，不管它是否可见，它的属性及控件会被应用程序所改变。在某些情况下，也许需要在初始化时加载所有的窗体并在以后需要的时候显示它们。

当 VisualBasic 加载 Form 对象时，先把窗体属性设置为初始值，再执行 Load 事件过程。当应用程序开始运行时，VisualBasic 自动加载并显示应用程序的启动窗体。

在加载 MDIChild 属性为 True 的 Form（换言之，子窗体）时，此时 MDIForm 还未被加载，那么会在子窗体前自动加载 MDIForm。

因为 MDI 子窗体不能被隐藏起来，所以在 Form_Load 事件过程结束时马上可以见到 MID 子窗体。

由 VisualBasic 函数产生的标准对话框，诸如 MsgBox 和 InputBox，不需要加载、显示或卸载即可简单地直接调用。

请参阅

Unload 语句，Hide 方法，Show 方法

示例

这个示例使用 Load 语句加载 Form 对象。要试用此示例，在 Form 对象的声明部分粘贴以下代码，然后运行此例并单击该 Form 对象。

```
PrivateSubForm_Click()
    DimAnswer,MsgasString                    '声明变量。
    UnloadForm1                              '卸载窗体。
    Msg="Form1hasbeenunloaded.ChooseYestoloadand"
    Msg=Msg&"displaytheform.ChooseNotoloadtheform"
    Msg=Msg&"andleaveitinvisible."
    Answer=MsgBox(Msg,vbYesNo)               '获得用户响应。
    IfAnswer=vbYesThen                       '测试应答。
        Show                                 '如果回答 Yes，则显示窗体。
    Else
        LoadForm1                            '如果回答 NO，则仅加载窗体。
        Msg="Form1isnowloaded.ChooseOKtodisplayit."
        MsgBoxMsg                            '显示消息。
        Show                                 '显示窗体。
    EndIf
EndSub
```

LoadPicture 函数

将图形载入到窗体的 Picture 属性、PictureBox 控件或 Image 控件。

语法

LoadPicture([*filename*],[size],[colordepth],[x,y])

LoadPicture 函数语法有下列部分：

部分	描述
Filename	可选的。字符串表达式指定一个文件名。可以包括文件夹和驱动器。如果未指定文件名，LoadPicture 清除图像或 PictureBox 控件
Size	可选变体。如果 *filename* 是光标或图标文件，指定想要的图像大小
colordepth	可选变体。如果 *filename* 是一个光标或图标文件，指定想要的颜色深度
x	可选变体，如果使用 y，则必须使用。如果 *filename* 是一个光标或图标文件，指定想要的宽度。在包含多个独立图像的文件中，如果那样大小的图像不能得到时，则使用可能的最好匹配。只有当 *colordepth* 设为 vbLPCustom 时，才使用 X 和 Y 值。Foriconfiles255 isthemaximumpossiblevalue
y	可选变体，如果使用 x，则必须使用。如果 *filename* 是一个光标或图标文件，指定想要的高度。在包含多个独立图像的文件中，如果那样大小的图像不能得到时，则使用可能的最好匹配。Foriconfiles255isthemaximumpossiblevalue

设置值

size 的设置值为：

常量	值	描述
vbLPSmall	0	系统小图标
vbLPLarge	1	系统大图标，由视频驱动程序决定
vbLPSmallShell	2	外壳小图标大小，由“控制面板”中的 DisplayProperties 对话框中 Appearance 选项卡上的 CaptionButtons 的大小设置决定
vbLPLargeShell	3	外壳大图标大小，由“控制面板”中的 DisplayProperties 对话框中 Appearance 选项卡上的图标大小设置决定
vbLPCustom	4	自定义大小，由 x 和 y 参数提供值

colordepth 的设置值为：

常量	值	描述
vbLPDefault	0	如果使用指定文件，则为最佳可用匹配
vbLPMonochrome	1	2 色
vbLPVGAColor	2	16 色
vbLPColor	3	256 色

说明

VisualBasic 可以识别的图形格式有：位图(.bmp)文件、图标(.ico)文件、光标(.cur)文件，行程编码(.rle)文件、元(.wmf)文件、增强的元文件(.emf),GIF(.gif)文件以及 JPEG(.jpg)文件。

赋值不带参数的 LoadPicture 将清除窗体、图片框及图像控件中的图形。

为了加载在 PictureBox 控件和 Image 控件中显示的图形或加载作为窗体背景的图形，必须将 LoadPicture 的返回值赋给要显示该图片的对象的 Picture 属性。例如：

```
SetPicture=LoadPicture("PARTY.BMP")
SetPicture1.Picture=LoadPicture("PARTY.BMP")
```

如果要将图标赋予窗体，则要把 LoadPicture 函数的返回值赋给 Form 对象的 Icon 属性：

```
SetForm1.Icon=LoadPicture("MYICON.ICO")
```

图标也可以被赋予除 Timer 控件和 Menu 控件外的其它控件的 DragIcon 属性，例如：

```
SetCommand1.DragIcon=LoadPicture("MYICON.ICO")
```

使用 LoadPicture 可将图形文件载入到系统剪贴板，如下所示：

```
Clipboard.SetDataLoadPicture("PARTY.BMP")
```

请参阅

SavePicture 语句，SetData 方法，Icon 属性，Picture 属性，DragIcon 属性，Picture 属性(ActiveX 控件)

示例

本例使用 LoadPicture 函数将图片加载到窗体的 PictureBox 控件并从控件上清除掉该图片。要试用此例，将 PictureBox 控件添加入 Form 对象，然后将以下代码粘贴到 Form 的声明部分，然后运行

此例，单击 Form。

```
PrivateSubForm_Click()
    DimMsgasString                  '声明变量。
    OnErrorResumeNext                   '设置错误句柄。
    Height=3990
    Width=4890                      '设置高度和宽度。
    Picture1.Picture=LoadPicture("PAPER.CUR",vbLPCustom,vbLPColor,
32,32) '加载光标。
    IfErrThen
        Msg="Couldn'tfindthe.curfile."
        MsgBoxMsg                   '显示错误消息。
        ExitSub                     '如果发生错误则退出。
    EndIf
    Msg="ChooseOKtoclearthebitmapfromtheform."
    MsgBoxMsg
    Picture1.Picture=LoadPicture()'清除 picturebox。
EndSub
```

LoadResData 函数

用以从资源(.res)文件装载若干可能类型的数据，并返回一个 Byte 数组。

语法

LoadResData(*index*,*format*)

LoadResData 函数的语法包含下列部分：

部分	描述
index	必需的。一个整数或字符串，它用来指定资源文件中数据的标识符(ID)。ID 标识符为 1 的资源保留给应用程序的图标
format	必需的。一个数值，它用来按照下列“设置值”中的描述，指定返回数据的原始格式。该数值也可以是用户定义资源的字符串名

设置值

用于 format **的设置值有：**

设置	描述
1	光标资源
2	位图资源
3	图标资源
4	菜单资源
5	对话框
6	字符串资源
7	字体目录资源
8	字体资源
9	加速键表
10	用户定义资源
12	群组光标
14	群组图标

说明

LoadResData 从资源文件装载的数据可以达到 64K。

对位图、图标、光标资源类型使用 LoadResData 将返回包含该资源中的实际二进制位的字符串。如果想使用实际的位图、图标或资源，请使用 LoadResPicture 函数.

LoadResData 的使用对于 VisualBasic 应用程序本地化有好处，这是因为需要翻译的资源在一个资源文件中被隔离，并且无须访问源代码或重新编译该应用程序。

请参阅

LoadResPicture 函数，LoadResString 函数

LoadResPicture 函数

用以从资源(.res)文件装载位图、图标或光标。

语法

LoadResPicture(*index*,*format*)

LoadResPicture 函数的语法包含下列部分：

部分	描述
index	必需的。一个整数或字符串，它用来指定资源文件中数据的标识符(ID)。ID 标识符为 1 的资源保留给应用程序的图标
format	必需的。一个数值或常数，如下列“设置值”中所描述的，指定返回数据的格式

设置值

用于 format 的设置值有：

常数	值	描述
vbResBitmap	0	位图资源
vbResIcon	1	图标资源
VbResCursor	2	光标资源

说明

可以使用 LoadResPicture 函数代替对存储在 Form 或控件的 Picture 属性中的图形的引用。

将位图、图标或光标存储在资源文件中并从该资源文件对它们进

行访问能改进装载时间，这是因为可以根据需要从资源文件单独装载，而不是在装载 Form 时一并全部装载。

LoadResPicture 的使用对于 VisualBasic 应用程序的本地化有好处，这是因为需要翻译的资源在一个资源文件中被隔离，并且无须访问源代码或重新编译该应用程序。

请参阅

LoadResData 函数，LoadResString 函数

LoadResString 函数

用以从资源(.res)文件装载字符串。

语法

LoadResString(*index*)

LoadResString 函数的语法包含下列部分：

部分	描述
index	必需的。一个整数，它用来指定资源文件中数据的标识符(ID)。ID 标识符为 1 的资源保留给应用程序图标

说明

可以使用 LoadResString 函数代替代码中的字符串文字量。

将数据的长字符串存储在资源文件中并从该资源文件访问它们能改进装载时间，这是因为您可以根据需要从资源文件单独装载，而不是在装载窗体时一并全部装载。

LoadResString 的使用对于 VisualBasic 应用程序的本地化有好处，这是因为需要翻译的资源在一个资源文件中被隔离，并且无须访问源代码或重新编译该应用程序。

请参阅

LoadResData 函数，LoadResPicture 函数

Loc 函数

返回一个 Long，在已打开的文件中指定当前读/写位置。

语法

Loc(*filenumber*)

必需的 *filenumber* 参数是任何一个有效的 Integer 文件号。

说明

Loc 函数对各种文件访问方式的返回值如下：

方式	返回值
random	上一次对文件进行读出或写入的记录号
sequential	文件中当前字节位置除以 128 的值。但是，对于顺序文件而言，不会使用 Loc 的返回值，也不需要使用 Loc 的返回值
binary	上一次读出或写入的字节位置

请参阅

EOF 函数，LOF 函数，Seek 函数，Seek 语句

示例

本示例使用 Loc 函数来返回在打开的文件中当前读写的位置。本示例假设 TESTFILE 文件内含数行文本数据。

```
DimMyLocation,MyLine
Open"TESTFILE"ForBinaryAs#1      '打开刚创建的文件。
DoWhileMyLocation<LOF(1)    '循环至文件尾。
   MyLine=MyLine&Input(1,#1)     '读入一个字符到变量中。
   MyLocation=Loc(1)             '取得当前位置。
'在立即窗口中显示。
   Debug.PrintMyLine:Tab:MyLocation
Loop
Close#1                          '关闭文件。
```

LocaleID 属性

返回包含用户区域标识（语言和国家/地区）的长整型数。

应用于

AmbientProperties 对象

语法

object.LocaleID

LocaleID 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

LocaleID 环境属性中包含当前用户语言和国家/地区的标识。控件可以使用该标识修改自身的行为以适合当前用户的语言和国家/地区。这种修改可能是简单的：例如，用用户的语言显示错误信息，也可能是比较复杂的：例如，用用户的语言修改属性、方法和事件名称。

如果容器没有实现这种环境属性，则缺省值将是当前系统的 LocaleID。

Lock，Unlock 语句

对于用 Open 语句打开的全部文件或一部分文件，其它进程所进行的控制访问。

语法

Lock[#]*filenumber*[,*recordrange*]

...

Unlock[#]*filenumber*[,*recordrange*]

Lock 和 Unlock 语句的语法具有以下几个部分：

部分	描述
filenumber	必需的。任何有效的文件号
recordrange	可选的。要锁定或解锁的记录范围

设置

recordrange 参数的设置如下：

recnumber|[start]Toend

设置	描述
Recnumber	记录号（Random 方式的文件）或字节数（Binary 方式的文件），指定要开始锁定或解锁的位置
Start	第一个要锁定或解锁的记录号或字节数
End	最后一个要锁定或解锁的记录号或字节数

说明

在有若干个进程访问同一个文件的环境中，要使用 Lock 与 Unlock 语句。

在使用 Lock 和 Unlock 语句时，这两个语句总是成对出现。Lock 和 Unlock 的参数必须完全一致。

文件中的第一个记录或字节在位置 1，第二个记录或字节在位置 2，依此类推。若只指定一个记录，则只对该记录进行锁定或解锁。若指定某个范围内的记录并省略起始记录(start)，则将此范围内的所有记录从头到尾(end)进行锁定或解锁。如果使用无 recnumber 的 Lock，则会将整个文件都锁定；而使用无 recnumber 的 Unlock 则会将整个文件都解锁。

若已经以顺序输入或输出的方式打开文件，则无论 start 和 end 指定什么范围，Lock 和 Unlock 都将影响整个文件。

警告在关闭一个文件或退出程序之前，务必要确保用 Unlock 语句删除对文件进行的所有锁定。若不能删除锁定，则会产生无法预料的后果。.

请参阅

Open 语句

示例

本示例示范如何使用 Lock 及 Unlock 语句。当用户正在修改数据中某个记录时，其他过程不可以来访问这条记录。本示例假设 TESTFILE 文件内含五个用户自定义类型 Record 的记录。

```
TypeRecord '定义用户自定义数据类型。
    IDAsInteger
    NameAsString*20
EndType

DimMyRecordAsRecord,RecordNumber          '声明变量。
'以随机访问的方式来打开文件。
Open"TESTFILE"ForRandomSharedAs#1Len=Len(MyRecord)
RecordNumber=4                            '指定记录编号。
Lock#1,RecordNumber                       '锁住该记录。
Get#1,RecordNumber,MyRecord               '读记录。
MyRecord.ID=234                           '修改记录。
MyRecord.Name="JohnSmith"
Put#1,RecordNumber,MyRecord               '将修改过的记录存回文件中。
Unlock#1,RecordNumber                     '当前记录解锁。
Close#1                                   '关闭文件。
```

Locked 属性

返回或设置一个值，以指定控件是否可被编辑。

应用于

ImageCombo 控件,Split 对象,Column 对象,DBCombo 控件,DBList 控件，TextBox 控件

语法

object.**Locked**[=*boolean*]

Locked 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，指定控件是否可被编辑，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
True	TextBox 控件－可以滚动和加亮控件中的文本，但不能编辑。程序可以通过改变 Text 属性来改变文本 Column 对象－不能编辑列对象中的值 ComboBox 对象－不能在文本框中输入文字或拖放其列表
False	TextBox 控件－可以编辑控件中的文本 Column 对象－可以编辑列中的值 ComboBox 对象－可以在文本框中输入文字及下拉其列表

说明

对于 Column 对象，Locked 将基本字段的 DataUpdatable 属性值作为缺省设置值；但是，如果 Column 是无界的或数据源不支持 DataUpdatable，则缺省值为 True。如果基本字段的 DataUpdatable 为 False，则将该属性设置为 True 不会产生错误。但是当该控件要把改变的数据写到数据库中时，就会发生错误。

对于 ComboBox 控件，当 Locked 属性设置为 True 时，用户不能改变任何数据，但是可以加亮文本框中的数据和进行复制。该属性不会影响程序访问 ComboBox。

请参阅

Text 属性，Locked 属性

LOF 函数

返回一个 Long，表示用 Open 语句打开的文件的大小，该大小以字节为单位。

语法

LOF(*filenumber*)

必要的 *filenumber* 参数是一个 Integer，包含一个有效的文件号。

注意对于尚未打开的文件，使用 FileLen 函数将得到其长度。

请参阅

FileLen 函数，EOF 函数，Loc 函数，Open 语句

示例

本示例使用 LOF 函数来得知已打开文件的大小。本示例假设 TESTFILE 文件内含文本数据。

```
DimFileLength
Open"TESTFILE"ForInputAs#1 '打开文件。
FileLength=LOF(1)  '取得文件长度。
Close#1                    '关闭文件。
```

Log 函数

返回一个 Double，指定参数的自然对数值。

语法

Log(*number*)

必要的 *number* 参数是 Double 或任何有效的大于 0 的数值表达式。

说明

自然对数是以 e 为底的对数。常数 e 的值大约是 2.718282。

如下所示，将 x 的自然对数值除以 n 的自然对数值，就可以对任意底 n 来计算数值 x 的对数值：

```
Logn(x)=Log(x)/Log(n)
```

下面的示例说明如何编写一个函数来求以 10 为底的对数值：

```
StaticFunctionLog10(X)
    Log10=Log(X)/Log(10#)
EndFunction
```

请参阅

Math 函数，Exp 函数，DerivedMath 函数

示例

本示例使用 Log 函数得到某数的自然对数值。

```
DimMyAngle,MyLog
'定义角度（以“弧度”为单位）。
MyAngle=1.3
'计算反双曲正弦函数值（inversesinh()）。
MyLog=Log(MyAngle+Sqr(MyAngle*MyAngle+1))
```

LogEvent 方法

在应用程序的日志目标中，把某个事件记入日志。在 WindowsNT 平台上，该方法会把内容写到 NT 的 Event 日志中。在 Windows95 平台上，该方法会把内容写到 LogPath 属性指定的文件中。按照缺省规定，如果不指定文件，事件被写入 vbevents.log 文件。

应用于

App 对象

语法

object.**LogEvent**(*logBuffer*,*eventType*)

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
logBuffer	必需的。写到日志中的字符串
eventType	可选的。Long 整数，它指定了事件的类型，如下列“设置值”中所示

设置值

eventType 的设置值是：

常数	值	描述
VbLogEventTypeError	1	错误
VbLogEventTypeWarning	2	警告
VbLogEventTypeInformation	4	信息

说明

可以在 Win32SDK 中得到记入日志的准则，不管是记入到 NT 的 Event 日志中，还是（在 Windows95 平台上）记入到 LogPath 属性所指定文件，都应该遵守那些准则。

请参阅

LogMode 属性，LogPath 属性

LogMode 属性

返回一个值，该值决定怎样（通过 LogEvent 方法）记入日志。在运行时只读。

应用于

App 对象

语法

object.**LogMode**=*mode*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
mode	Long。它决定了记入日志的办法，如下列“设置值”中所示

设置值

mode 的设置值是：

常数	值	描述
vbLogAuto	0	如果在 Windows95 上运行，则该选项就把日志消息记入到 LogPath 属性所指定的文件中。如果在 WindowsNT 上运行，则日志消息将连同应用程序都被记入到NT 应用程序 EventLog 中 。 "VBRunTime" 当 作 应 用 程 序 源 andApp.Titleappearinginthedescription 使用
vbLogOff	1	关闭所有记入日志。来自 UI 分路及 LogEvent 方法中的消息都被忽略并抛弃
vbLogToFile	2	强制记入日志到文件中。如果在 LogPath 中未出现有效文件名，则忽略记入日志，并将属性设置为 vbLogOff
vbLogToNT	3	强制记入到 NT 事件日志中。如果不是在 WindowsNT 上运行，或者事件日志无效，则忽略记入日志，并将属性设置为 vbLogOff
vbLogOverwrite	0x10	指示每次启动应用程序时，应该重新创建日志文件。这个值可与使用 OR 操作符的其它模式选项结合使用。记入日志的缺省动作将追加到现有文件中。在 NT 事件记入日志的情况下，该标记没有任何意义
vbLogThreadID	0x20	指示在窗体"[T:0nnn]"中，当前线程 ID 被考虑到消息中。这个值可以与使用 OR 操作符的其它模式选项结合使用。缺省动作表明，

仅当应用程序是多线程时（要么明确标记为线程安全的，要么作为不明确的多线程应用程序—比如一个其实例属性设置为 Single-Use、多线程的本地服务器来实现），才显示线程 ID

返回值

Long

请参阅

LogEvent 方法，LogPath 属性

LogPath 属性

返回用来捕获 LogEvent 方法输出文件的路径及其文件名。在设计时，该属性无效；而在运行时只读。

应用于

App 对象

语法

object.**LogPath**=*path*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
path	字符串。日志文件的路径及其文件名

说明

LogMode 属性决定了记入日志的方式。如果没有设置 LogPath，那

么 LogEvent 方法写到 NTLogEvent 文件中。

请参阅

LogEvent 方法，LogMode 属性

Long 数据类型

Long（长整型）变量存储为 32 位（4 个字节）有符号的数值形式，其范围从-2,147,483,648 到 2,147,483,647。Long 的类型声明字符为和号(&)。

请参阅

TypeConversion 函数，DataTypeSummary，IntegerData 类型，Deftype 语句

LostFocus 事件

此事件是在一个对象失去焦点时发生，焦点的丢失或者是由于制表键移动或单击另一个对象操作的结果，或者是代码中使用 SetFocus 方法改变焦点的结果。

应用于

ADOData 控件，TreeView 控件，ImageCombo 控件，ListView 控件,Slider 控件,TabStrip 控件,DateTimePicker 控件,Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控

件，MaskedEdit 控件，MSHFlexGrid 控件，MsFlexGrid 控件，RichTextBox 控件，Extender 对象，PropertyPage 对象，User Control 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，HScrollBar，VScrollBar 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLEContainer 控件

语法

PrivateSubForm_LostFocus()

PrivateSub*object*_**LostFocus**(**[***index***AsInteger]**)

LostFocus 事件包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，它用来唯一地标识一个在控件数组中的控件

说明

LostFocus 事件过程主要是用来对更新进行验证和确认。使用 LostFocus 可以在焦点移离控件时引进确认。这类事件过程的另一种用途与在 GotFocus 事件过程中的应用类似，可以隐藏、显示其它对象或者使它们有效或无效。也可使设置在该对象的 GotFocus 事件过程中的条件取反或对其进行更改。

如果由 VisualBasic 所建立的.exe 文件要显示一个也是用 VisualBasic 所建立的.dll 文件所创建的对话框，则该.exe 文

件的窗体将得到 Deactivate 和 LostFocus 事件。这可能是不希望的，因为对话框将不能获得 Deactivate 事件：

如果该对象是进程外的部件。

如果该对象不是用 VisualBasic 编写成。

在开发环境中当调用一个用 VisualBasic 所建立的 DDL 时。

请参阅

GotFocus 事件，SetFocus 方法，TabIndex 属性，TabStop 属性，ActiveControl 属性，ActiveForm 属性

示例

本例演示了在 TextBox 获得或失去焦点（用鼠标或 TAB 键选择）时，改变颜色，并在 Label 控件中显示适当的文字。

要尝试这个例子，可将代码粘贴到一个包含 2 个 TextBox 控件和一个 Label 的窗体声明部分，然后按 F5 键并在 Text1 和 Text2 之间移动焦点。

```
PrivateSubText1_GotFocus()
   '将焦点用红色显示.
   Text1.BackColor=RGB(255,0,0)
   Label1.Caption="Text1hasthefocus."
EndSub

PrivateSubText1_LostFocus()
   '用蓝色显示失去焦点.
```

```
    Text1.BackColor=RGB(0,0,255)
    Label1.Caption="Text1does'thavethefocus。"
EndSub
```

LostFocus 事件（UserControl 对象和 UserDocument 对象）

当焦点离开对象或子控件时，发生该事件。

应用于

UserControl 对象，UserDocument 对象

语法

Sub*object*_LostFocus()

LostFocus 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

该 LostFocus 事件不是使用 object 的开发者处理的 LostFocus 扩展事件。该LostFocus事件为object的创建者提供，而且属于object内部。

通过该事件，可使 object 获知焦点现在是否还在它的上面。

只有当 CanGetFocus 属性设置为 True 并且没有能够接受焦点的子控件时，object 本身才能获得焦点。

LostFocus 事件在 ExitFocus 事件前产生。

请参阅

CanGetFocus 属性，ExitFocus 事件

lpOleObject 属性

返回对象的地址。

应用于

OLEContainer 控件

语法

object.**lpOleObject**

object 所在处是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

在 ActiveXDLLs 中的很多函数调用需要对象的地址作为参数。当 API 调用 ActiveXDLLs 时，要传递在 lpOleObject 属性中指定的值。如果当前没有对象显示，该值为 0。如果调用了回调 OLE 容器控件的 API，其结果不可预料。

这个属性返回的地址是一个指向活动对象的 lpOleObject 接口的指针。

LSet 语句

在一字符串变量中将一字符串往左对齐，或是将一用户定义类型变量复制到另一用户自定义类型变量。

语法

LSet*stringvar=string*

LSet*varname1=varname2*

LSet 语句的语法有下面这些部分：

部分	说明
stringvar	必需的。字符串变量名称
string	必需的。在 stringvar 内想往左对齐的字符串表达式
varname1	必需的。用户自定义类型变量名，内容将复制进来
varname2	必需的。用户自定义类型变量名，内容将复制出去

说明

Lset 会将 stringvar 中空余的字符以空白替换。

如果 string 比 stringvar 还长，Lset 只在 stringvar 中放置最左边几个字符，且长度为 stringvar 的长度。

警告不能使用 Lset 将一用户自定义类型变量复制到另一用户自定义类型变量。将一个数据类型的数据复制到留给另一数据类型的空间，可能会造成不可预料的结果。

当从一用户自定义类型复制一变量给其他变量时，此变量的二进制数据会从一个变量复制到另一个变量的内存空间中，但并不关心为元素指定的数据类型。

请参阅

DataTypeSummary，RSet 语句

示例

本示例使用 LSet 语句，将某字符串插入到另一字符串的最左边。虽然 LSet 也能用来复制一个用户自定义类型变量的内容到另一个不一定完全一样，但却兼容的用户自定义类型变量中，但并不建议使用这种方法。因为不同操作系统间对数据结构的做法不同，使用 LSet 的程序并不能保证有很好的移植性。

```
DimMyString
MyString="0123456789"  '设置字符串初值。
LsetMyString="<-Left"  'MyString 的内容为"<-Left"。
```

Ltrim，RTrim 与 Trim 函数

返回 Variant(String)，其中包含指定字符串的拷贝，没有前导空白(LTrim)、尾随空白(RTrim)或前导和尾随空白(Trim)。

语法

LTrim(*string*)

RTrim(*string*)

Trim(*string*)

必要的 *string* 参数可以是任何有效的字符串表达式。如果 *string* 包含 Null，将返回 Null。

请参阅

Left 函数，Right 函数

示例

本示例使用 LTrim 及 RTrim 函数将某字符串的开头及结尾的空格

全部去除。事实上只使用 Trim 函数也可以做到将两头空格全部去除。

```
DimMyString,TrimString
MyString="<-Trim->"                    '设置字符串初值。
TrimString=LTrim(MyString)             'TrimString="<-Trim->"。
TrimString=RTrim(MyString)             'TrimString="<-Trim->"。
TrimString=LTrim(RTrim(MyString))      'TrimString="<-Trim->"。
'只使用 Trim 函数也同样将两头空格去除。
TrimString=Trim(MyString)'TrimString="<-Trim->"。
```

MainWindow 属性

返回一个代表 VisualBasic 开发环境主窗口的 Window 对象，此属性为只读。

应用于

VBE 对象

说明

可以使用由 MainWindow 属性所返回的 Window 对象来增加或删除可连接窗口。亦可使用由 MainWindow 属性所返回的 Window 对象来将 VisualBasic 开发环境的主窗口放到最大、缩到最小、隐含

或是还原。

请参阅

Add 方法(VBA 外接程序对象模型)，Close 方法(VBA 外接程序对象模型)，Remove 方法(VBA 外接程序对象模型)，SetFocus 方法，Window 对象，Windows 集合，ActiveWindow 属性，Caption 属性，Visible 属性

示例

下列示例使用 MainWindow 属性返回代表主窗口的 Window 对象，并打印主窗口标题。

```
Debug.PrintApplication.VBE.MainWindow.Caption
```

Major 属性

返回或设置该工程的主要版本号。该属性在运行时是只读的。

应用于

App 对象

语法

object.**Major**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

Major 属性的取值范围在 0 到 9999 之间。

该属性提供运行中的应用程序的版本信息。

在设计时，使用位于“工程属性”对话框中的“生成”选项卡上的“主版本”框可设置该属性。

请参阅

Minor 属性，Revision 属性

Major 属性(VBAAdd-In 对象模型)

返回一个 Long，其内容为被引用的类型库的主版本号，此属性为只读。

应用于

Refeence 对象

说明

由 Major 属性所返回的代码，相当于存在曾引用过的类型库里的主版本号。

请参阅

Minor 属性，Version 属性

示例

下列示例使用 Major 属性返回某工程中指定的 Reference 对象的主版本号。

```
Debug.PrintApplication.VBE.VBProjects(1).References(1).Major
```

MakeCompiledFile 方法

该方法依据工程的类型使当前的工程被写成可执行文件、动态链接库或控件。

应用于

VBProject 对象

语法

object.MakeCompiledFile

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

MaskColor 属性

返回或设置一个在按钮的图片中作为“掩码”（也就是说，透明）的颜色。

应用于

CheckBox 控件，CommandButton 控件，OptionButton 控件

语法

object.MaskColor [=*color*]

MaskColor 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
color	一个决定作为掩码的颜色的值或常数，“设置值”中有详细的描述

设置值

VisualBasic 使用 MicrosoftWindows 操作环境的红-绿-蓝(RGB)配色方案。color 的设置值为：

设置值	描述
NormalRGBcolors	由调色板或通过在代码中使用 RGB 或 QBColor 函数指定的颜色
&H00C0C0C0	（缺省）浅灰色

说明

如果系统颜色改变，则透明的颜色也改变，使图片的外观无法预见。最好不用系统颜色编程图片。

只有当 UseMaskColor 属性设置为 True 并且按钮中有赋值给其 Picture 属性的位图-风格图片时，该属性才被使用（图标和元文件已经包含透明信息）

如果 MaskColor 属性在运行时被改变，按钮将自己用被当作掩码的新颜色重画。

MaskColor 属性（UserControl 对象）

返回或设置赋给 UserControl 对象的 MaskPicture 属性的位图的透明区域的颜色，该 UserControl 对象的 BackStyle 属性设置为 0（透明）。

语法

object.**MaskColor** [=*color*]

MaskColor 属性的语法包括下述部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的对象
color	决定用作屏蔽色的值或常数，参见“设置值”的说明

设置值

VisualBasic 使用 MicrosoftWindows 操作环境的红-绿-蓝(RGB)颜色方案。*color* 的设置值为：

设置值	描述
RGB 颜色	使用颜色调色盘或代码中的 RGB 或 QBColor 函数所指定的颜色

说明

当将位图赋给一个 BackStyle 属性为 0（透明）的 UserControl 的 MaskPicture 属性时，该控件被位图中颜色为 MaskColor 属性值的区域所覆盖的部分就成为透明的。

在该透明区域发生的鼠标事件由该容器接收，或由本来应覆盖这

一区域的 UserControl 控件接收。

如果没有将位图赋给 MaskPicture 属性，或 UserControl 的 BackStyle 属性不为 0（透明），则对 MaskColor 属性的设置不起作用。

若要进一步详细了解，请参阅 UserControl 对象的 MaskPicture 属性。

注意尽管 MaskColor 接受了对象浏览器中 VisualBasic(VB)对象库所列的系统颜色常数，如在有关 BackColor 和 ForeColor 属性的帮助中所描述的，也只有当 MaskPicture 的位图包含系统颜色时才有用。

MaskPicture 属性（UserControl 对象）

返回或设置一个位图，该位图与 MaskColor 属性一起，决定了 BackStyle 属性为 0（透明）时的 UserControl 对象的透明区域和可见区域。

语法

object.**MaskPicture**[=*picture*]

MaskPicture 属性的语法包括下述部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的对象
picture	一幅图像，其类型为设置值中所描述的一种

设置值

picture 的设置值如下：

设置值	描述
Nothing	（缺省）没有图片。在设计时，突出显示该属性值，并按下 Delete 键；"(None)"将出现在属性窗口。在运行时，可以将参数为空文件名的 LoadPicture 函数的返回值赋给该属性，或使用 Set 语句将该属性的值设为 Nothing
(Bitmap,DIB,GIF,orJPEG)	指定图像类型的位图。在设计时，可以使用属性窗口键入字符串表达式来指定包含图片的文件。在运行时，可以将该属性设为一个 Picture 对象、另一个对象的 Picture 属性或使用 LoadPicture 函数装入一个图像类型的位图文件

重点这一特性只适用于图像类型的位图，如 GIF、JPEG 以及 DIB。而不适用于 Windows 图元文件、图标或光标。

说明

当将位图赋给一个 BackStyle 属性为 0（透明）的 UserControl 的 MaskPicture 属性时，该控件被位图中颜色为 MaskColor 属性值的区域所覆盖的部分就成为透明的。

在该透明区域发生的鼠标事件由该容器接收，或由本来应覆盖这一区域的 UserControl 控件接收。MaskPicture 位图的不透明部分使用由 UserControl 的 BackColor 属性所指定的颜色绘制。这些区域，不必是相邻的区域，定义了在 UserControl 上绘制的剪裁区域。也

就是说，在 UserControl 的表面进行的任何绘制都要根据 MaskPicture 位图的不透明区域进行剪裁。

重点赋给 MaskPicture 属性的位图只用于定义 UserControl 的透明区域和剪裁区域。该位图永不会显示。要在由 MaskPicture 以及 MaskColor 所定义的剪裁区域上显示一幅位图，可以将该位图赋给 UserControl 的 Picture 属性，或在 UserControl 的绘制事件中使用 PaintPicture 将其绘制在该 UserControl 上。

通过在一个隐藏的 PictureBox 上绘制，然后把结果图像传给 UserControl 的 MaskPicture 属性，就可以为控件创建一个活动的剪裁区域。

注意若在设计时设置 MaskPicture 属性，则该图形用定义 UserControl 的.ctl 和.ctx 文件保存和加载。如果将该工程作成一个控件部件（.ocx 文件），则文件中就包含了该图像。相反，若在运行时加载一个图形，则该图形并不与部件一起保存。

详细信息请参阅《部件工具指南》中的“创建 ActiveX 部件”的“生成 ActiveX 控件”中的“控件背景透明”。

数学函数

Abs 函数

Atn 函数

Cos 函数

Exp 函数

Fix 函数

Int 函数

Log 函数

Rnd 函数

Sgn 函数

Sin 函数

Sqr 函数

Tan 函数

导出的数学函数

Max，Min 属性（公共对话框）

可为下列各项设置 Max 和 Min 属性：

“字体”对话框～返回或设置在“大小”列表框显示的字体的最大和最小尺寸。

“打印”对话框～返回或设置打印范围允许的最大和最小值。

应用于

CommonDialog 控件

语法

object.**Min***[=points]*

object.**Max***[=points]*

object.**Min***[=numbe*$^{r]}$

object.**Max***[=number]*

Max 和 Min 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
points	数值表达式，它指定最小和最大的字体尺寸
number	数值表达式，它指定页码的最小值和最大值

说明

对于“字体”对话框，使用这些属性前必须先置 cdlCFLimitSize 标志。

对于“打印”对话框，Min 属性确定能在从文本框中指定的最小号。Max 属性确定能在到文本框中指定的最大号。

数据类型

Integer

请参阅

Printer 对象，Printers 集合，Font 对象

Max、Min 属性（滚动条）

Max——返回或设置当滚动框处于底部或最右位置时，一个滚动条位置的 Value 属性最大设置值。对于 ProgressBar 控件，它返回或设置其最大值。

Min——返回或设置当滚动框处于顶部或最右位置时，一个滚动条位置的 Value 属性最小设置值。对于 ProgressBar 控件，它返

回或设置其最小值。

应用于

HScrollBar，VScrollBar 控件

语法

object.**Max**[=*value*]

object.**Min**[=*value*]

Max 和 Min 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	数值表达式，按照设置值中所述，指定最大或最小的 Value 属性设置值

设置值

对于每个属性，可指定在-32,768 和 32,767 范围之间的一个整数，包括-32,768 和 32,767。缺省设置值为：

Max——32,767.

Min——0.

说明

MicrosoftWindows 操作环境自动地将滚动条范围设置为与窗体、ComboBox 控件和 ListBox 控件的内容成比例。对于一个滚动条（HScrollBar 或 VScrollBar）控件，无论如何，必须指定这些范围。根据滚动条的具体使用情况——例如，作为输入设备或作为一个速度或数量的指示器，使用 Max 和 Min 设置一个合适的范围。

典型地，你在设计时设置 Max 和 Min。如果滚动范围必须动态地改变，你也能够在运行时在代码中设置它们——例如，当向能够被滚动的数据库增加记录时。使用属性 LargeChange 和 SmallChange，为一个滚动条控件设置最大和最小滚动增量。

注意如果 Max 被设为比 Min 小的值，那么最大值将被分别设为水平或垂直滚动条的最左或最上位置处。ProgressBar 控件的 Max 属性必须总是比其 Min 属性大一些，并且其 Min 属性必须总是大于或等于 0。

Max 和 Min 属性定义了控件的范围。缺省 ProgressBar 控件的 Min 属性是 0 并且其 Max 属性设置值为 100，表示操作的百分比持续时间。

请参阅

LargeChange，SmallChange 属性

MaxButton 属性

返回一个值，标识是否一个窗体具有“最大化”按钮

应用于

Form 对象，Forms 集合

语法

object.**MaxButton**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

设置值

MaxButton 属性设置值为：

设置值	描述
True	（缺省值）窗体具有最大化按钮
False	窗体没有最大化按钮

说明

利用最大化按钮可以将窗体窗口扩大为全屏幕。要显示最大化按钮，必须将 BorderStyle 属性设置为 1（固定单边框）、2（可变尺寸）或 3（固定双边框）。

窗口最大化后，最大化按钮自动地变成恢复按钮，将窗口最小化或恢复窗口把恢复按钮变回最大化按钮。

为 MaxButton、MinButton、BorderStyle 和 ControlBox 属性指定的设置值直到运行时才能在窗体的外观上反映出来。

注意运行时最大化窗体会产生 Resize 事件。WindowState 属性反映窗口的当前状态。如果将 WindowState 属性设置为 2（最大化的），则无论 MaxButton 和 BorderStyle 属性是何种有效设置窗体都将最大化。

请参阅

ControlBox 属性，MinButton 属性

MaxFileSize 属性

返回或设置使用 CommonDialog 控件被打开的文件名的最大尺寸。

应用于

CommonDialgo 控件

语法

object.**MaxFileSize**[=*value*]

MaxFileSize 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	整数，它指定文件名的最大字节数。该属性的范围是 1~32K。缺省值是 256

说明

MaxFileSize 属性要分配内存以便存储所选的一个或多个文件的实际名称。当使用 cdlOFNAllowMultiselect 标志时，可能希望增加 MaxFileSize 属性的尺寸以便对所选文件名有足够的内存。

数据类型

Integer

请参阅

FileName 属性

MaxLength 属性

返回或设置一个值，它指出在 TextBox 控件中能够输入的字符是否有一个最大数量，如果是，则指定能够输入的字符的最大数量。注意在 DBCS（双字节字符集）系统中，每个字符能够取两个字节而不是一个字节，以此来限制你能够输入的字符的数量。

应用于

TextBox 控件

语法

object.**MaxLength**[=*value*]

MaxLength 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	整数，用来指定在 TextBox 控件中能够输入的最大字符数。MaxLength 属性的缺省值为 0，指出对于用户系统上单行 TextBox 控件来说，最大值不能超过被内存强制建立的值，并且对于多行 TextBox 控件而言，最大值大约为 32K。任何大于 0 的数表示字符数的最大值

说明

使用 MaxLength 属性来限制在 TextBox 中能够输入的字符数量。如果长度超过 MaxLength 属性设置值的文本从代码中赋给 TextBox，不会发生错误；但是，仅只有最大数量的字符被赋给 Text

属性，而额外的字符被截去。改变该属性不会对 TextBox 的当前内容产生影响，但将影响以后对内容的任何改变。

请参阅

Text 属性，MultiLine 属性，Text 属性(ActiveX 控件)

示例

本例使用 TextBox 控件中的数字值限制另一个 TextBox 控件中的文本长度。要试用此例，先将代码粘贴到包含两个 TextBox 控件窗体的声明部分。使 Text1 有足够大小，然后按 F5 键。向 Text2 输入一个数字并向 Text1 输入文本。

```
PrivateSubText1_Change()
   Text1.MaxLength=Text2.Text
EndSub
```

MDIChild 属性

返回或设置一个值，它指示一个窗体是否被作为 MDI 子窗体在一个 MDI 窗体内部显示。在运行时是只读的。

应用于

Form 对象，Forms 集合

语法

object.**MDIChild**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

设置值

MDIChild 属性的设置值为：

设置值	描述
True	窗体是一个 MDI 子窗体并且被显示在父 MDI 窗体内
False	（缺省值）窗体不是一个 MDI 子窗体

说明

当建立一个多文档接口(MDI)应用程序时要使用该属性。在运行时，该属性被设置为 True 的窗体被显示在 MDI 窗体内。一个 MDI 子窗体能够被最大化、最小化和移动，都在父 MDI 窗体内部进行。

为了使 MDI 子窗体起作用时，注意下面的事项：

在运行时，当一个 MDI 子窗体被最大化时，其标题将与父 MDI 窗体的相结合。

在设计时，一个 MDI 子窗体将像其它窗体一样显示，因为该窗体仅在运行时才被显示在父窗体内部。在“工程”窗口中一个 MDI 子窗体的图标与别的类型的窗体的图标是不同的。

MDI 子窗体不能是模式的。

MDI 子窗体的初始化大小和位置被 MicrosoftWindows 操作环境控制，除非特别在 Load 事件过程中设置它们。

如果 MDI 子窗体在其父窗体装入之前被引用，则其父 MDI 窗体将被自动装入。然而，如果父 MDI 窗体在 MDI 子窗体装入前被引用，则子窗体并不被装入。

注意所有 MDI 子窗体都有可调整大小的边框、控制菜单框以及最

小化和最大化按钮，而不管 BorderStyle、ControlBox、MinButton 和 MaxButton 属性的设置值如何。

对 MDIForm 对象的任何引用，包括读和属性设置在内，都将导致该窗体被装入并成为可见的。

请参阅

BorderStyle 属性，控件 Box 属性，MaxButton 属性，MinButton 属性，WindowList 属性，ActiveControl 属性，ActiveForm 属性，BorderStyle 属性(ActiveX 控件)

示例

本例在 MDIForm 对象中创建第二个 MDI 子窗体的实例。要试用此例，先在 Form1 上将 MDIChild 属性设为 True，然后用“工程”菜单上的“添加 MDI 窗体”命令来创建一个 MDIForm 对象。将下列代码粘贴到 MDIForm 的声明部分，然后按 F5 键以运行该程序。

```
PrivateSubMDIForm_Load()
    DimNewFormAsNewForm1'声明新窗体。
    NewForm.Show'显示新窗体。
EndSub
```

MDIForm 对象

MDI（多文档接口）窗体可作为应用程序背景的窗口，也是其 MDIChild 属性设置为 True 的窗体的容器。

语法

MDIForm

说明

在“插入”菜单中选择“添加 MDI 窗体”就可以创建 MDIForm 对象。

一个应用程序只能有一个 MDIForm 对象，但是可以有多个 MDI 子窗体。如果 MDI 子窗体有菜单，那么，当 MDI 子窗体为活动窗体时，子窗体的菜单条自动取代 MDIForm 对象的菜单条。最小化的 MDI 子窗体以图标形式出现在 MDIForm 中。

MDIForm 对象只包含 Menu 和 PictureBox 控件以及具有 Align 属性的自定义控件。为了把其它的控件放入 MDIForm，可以在窗体上绘制一个图片框，然后在图片框中绘制上其它控件。可以在 MDIForm 的图片框中使用 Print 方法显示文本,但是不能在 MDIForm 自身使用该方法显示文本。

MDIForm 对象不能是模式。

MDI 子窗体的设计与 MDIForm 无关，但在运行时总是包含在 MDIForm 中。

可以在 MDIForm 中使用 Controls 集合访问控件集合。例如，使用如下代码可以将 MDIForm 中的所有控件隐藏起来：

```
ForEachControlinMDIForm1.Controls
   Control.Visible=False
NextControl
```

MDIForm 的 Count 属性指出了 Controls 集合中控件的数目。

属性

RightToLeft 属性，控件属性，Moveable 属性，StartUpPosition 属性，NegotiateToolbars 属性，OLEDropMode 属性，BackColor，ForeColor 属性，Height，Width 属性，Icon 属性，Left，Top 属性，Picture 属性，Tag 属性，Visible 属性，AutoShowChildren 属性，hWnd 属性，LinkMode 属性，LinkTopic 属性，MouseIcon 属性，MousePointer 属性，ScaleHeight，ScaleWidth 属性，ScrollBars 属性，WindowState 属性，ActiveControl 属性，Appearance 属性，ActiveForm 属性，Caption 属性，Count 属性(VB 集合)，Enabled 属性，HelpContextID 属性，Name 属性，WhatsThisHelp 属性

Me

Me 关键字像是隐含声明的变量。这个关键字适用于类模块中的每个过程。当类有多个实例时，Me 在代码正在执行的地方提供引用具体实例的方法。要把当前执行类实例的有关信息传递到另一个模块的过程，Me 非常有用。例如，假定模块中有以下过程：

```
SubChangeFormColor(FormNameAsForm)
FormName.BackColor=RGB(Rnd*256,Rnd*256,Rnd*256)
EndSub
```

可以调用这个过程并使用下列语句将窗体类的当前实例作为参数传递。

ChangeFormColorMe

请参阅

Set 语句

Member 对象

Member 对象表示基于代码的成员属性和基于类型库的成员属性的混合。

语法

Member

说明

基于代码的属性（如 Name）是只读的，因此，为了更改这些属性，必须对外接程序修改代码。

属性

Collection 属性，Description 属性，VBE 属性，Bindable 属性，Browsable 属性，Category 属性，DefaultBind 属性，DisplayBind 属性，Hidden 属性，PropertyPage 属性，RequestEdit 属性，Standard 方法属性，UIDefault 属性，CodeLocation 属性，Scope 属性，Static 属性，类型属性，HelpContextID 属性，Name 属性

请参阅

CodeLocation 属性，CodePane 对象，CodeModule 对象

Members 集合

返回代码模块等级成员组成的集合。

语法

Members

说明

代码模块的成员是具有模块等级范围的标识符，可将此成员看作代码模块的属性、方法或事件。

属性

VBE 属性，Count 属性(VB 集合)，Parent 属性

方法

Item 方法

请参阅

CodePane 对象，CodeModule 对象

Members 属性

包含具有模块级范围的标志，而且可将这些标志作指定 CodeModule 对象的属性、方法或事件。

应用于

CodeModule 对象

语法

object.**Members**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Menu 控件

Menu 控件显示应用程序的自定义菜单。命令、子菜单和分隔符条都可包括在菜单之中。每一个创建的菜单至多有四级子菜单。

语法

Menu

说明

为了创建 Menu 控件，要使用“菜单编辑器”。在“标题框”中输入 Menu 控件的名称。为了创建分隔符栏，要在“标题框”中输入单连字符(-)。为了在菜单项的左侧显示复选标记，则要选择“复选框”。

可以使用“菜单编辑器”设置某些 Menu 控件属性，所有 Menu 控件属性都要显示在“属性”窗口中。为了显示 Menu 控件的属性，要在“属性”窗口上部的“对象”列表中选择菜单名称。

在创建 MDI 应用程序时，当子窗体为活动窗体的时候，MDI 子窗体上的菜单条将替换 MDIForm 对象上的菜单条。

属性

NegotiatePosition 属性，Tag 属性，Visible 属性，Shortcut 属性，WindowList 属性，Caption 属性，Checked 属性，Enabled

属性，HelpContextID 属性，Index 属性(控件数组)，Name 属性，Parent 属性

请参阅

MDIForm 对象

MessageReflect 属性

返回布尔值，它规定控件容器是否自动处理返回消息。

应用于

AmbientProperties 对象

语法

object.**MessageReflect**

MessageReflect 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

设置值

MessageReflect 属性可能返回的布尔值为：

设置值	描述
True	控件的容器返回消息
False	控件的容器不能返回消息。如果容器不能实现这种环境属性，这将是缺省的设置值

说明

当控件作为子类时，存在着某些正在发送给父控件的消息。在正

常的情况下，这些消息将反映到发送它们的控件中，这样可以使控件处理它自身的消息。这种消息返回可由容器处理，它以事件的形式返回消息。MessageReflect 属性表明某个控件的容器是否返回消息。

如果控件放置在不返回消息的容器中，则将严重损害控件的操作；控件的大多数操作取决于返回的消息。

Mid 函数

返回 Variant(String)，其中包含字符串中指定数量的字符。

语法

Mid(*string*,*start*[,*length*])

Mid 函数的语法具有下面的命名参数：

部分	说明
string	必需的。字符串表达式，从中返回字符。如果 string 包含 Null，将返回 Null
start	必需的。为 Long。string 中被取出部分的字符位置。如果 start 超过 string 的字符数，Mid 返回零长度字符串("")
length	可选的。为 Variant(Long)。要返回的字符数。如果省略或 length 超过文本的字符数（包括 start 处的字符），将返回字符串中从 start 到尾端的所有字符

说明

欲知 string 的字符数，可用 Len 函数。

注意 MidB 函数作用于字符串中包含的字节数据，如同在双字节字符集(DBCS)语言中一样。因此其参数指定的是字节数，而不是字符数。对于使用 MidB 的示例代码，请参阅示例主题中的第二个示例。

请参阅

Left 函数，Len 函数，Ltrim，Rtrim 和 Trim 函数，Mid 语句，Right 函数

示例

本示例使用 Mid 语句来得到某个字符串中的几个字符。

```
DimMyString,FirstWord,LastWord,MidWords
MyString="MidFunctionDemo" '建立一个字符串。
FirstWord=Mid(MyString,1,3)      '返回"Mid"。
LastWord=Mid(MyString,14,4)      '返回"Demo"。
MidWords=Mid(MyString,5)    '返回"FuncionDemo"。
```

第二个实例使用 MidB 和用户定义的函数 MidMbcs 来从字符串中返回字符。这里的区别是输入的字符串是 ANSI 码和字节长度。

Mid 语句

在一 Variant(String)变量中以另一个字符串中的字符替换其中指定数量的字符。

语法

Mid(*stringvar*,*start*[,*length*])=*string*

Mid 语句的语法具有下面几个部分：

部分	描述
stringvar	必需的。被更改的字符串变量名
start	必需的。Variant(Long)。stringvar 中被替换的字符开头位置
length	可选的。Variant(Long)。被替换的字符数。如果省略，string 将全部用上
string	必需的。字符串表达式，替换部分 stringvar 的字符串

说明

被替换的字符数量总是小于或等于 stringvar 的字符数。

注意 MidB 语句作用于包含在字符串中的字节数据。在 MidB 语句中，start 指定 stringvar 中被替换的字节开头位置，而 length 为替换的字节数。

```
MyNewString=Mid(MyString,3,4)
‘Returns“”CdEf”
MyNewString=MidB(MyString,3,4)
‘Returns“”bc”
MyNewString=MidMbcs(MyString,3,4)
‘Returns“bcd”
```

示例

本示例使用 Mid 语句将某字符串中的几个字符替换为其他的字

符。

```
DimMyString
MyString="Thedogjumps" '设置字符串初值。
Mid(MyString,5,3)="fox"    'MyString="Thefoxjumps"。
Mid(MyString,5)="cow"  'MyString="Thecowjumps"。
Mid(MyString,5)="cowjumpedover"    'MyString="Thecowjumpe"。
Mid(MyString,5,3)="duck"   'MyString="Theducjumpe"。
```

MinButton 属性

返回一个值，指示窗体是否有“最小化”按钮。

应用于

Form 对象，Forms 集合

语法

object.**MinButton**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

返回值

MinButton 返回值如下：

设置值	描述
True	（缺省值）窗体具有最小化按钮
False	窗体没有最小化按钮

说明

最小化按钮能够将窗体窗口最小化为图标。要显示最小化按钮，必须将 BorderStyle 属性设置为 1（固定单边框）、2（可变尺寸）或 3（固定双边框）。

为 MaxButton、MinButton、BorderStyle 和 ControlBox 属性指定的设置值直到运行时才能在窗体的外观上反映出来。

注意运行时将窗体最小化成图标会产生 Resize 事件。WindowState 属性反映窗口的当前状态。如果将 WindowState 属性设置为 2（最大化的），则无论 MaxButton 和 BorderStyle 属性是何种有效设置窗体都将最大化。

请参阅

ControlBox 属性，MaxButton 属性

MinHeight 属性和 MinWidth 属性

返回视口的最小高度或宽度，或者对其进行设置，滚动条将以这样的高度或宽度出现在容器上。

应用于

UserDocument 对象

语法

object.**MinHeight**=*single*

object.**MinWidth**=*single*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
single	UserDocument 的高度或者宽度，滚动条将以这样的高度或宽度出现在某个容器上

说明

MinHeight 属性和 MinWidth 属性的缺省值是由 UserDocument 的 Height 和 Width 属性设置的。

如果将 ScrollBars 属性设置为 False，那么 MinWidth 和 MinHeight 属性将没有任何影响。

Minor 属性

返回或设置该工程的小版本号。该属性在运行时是只读的。

应用于

App 对象

语法

object.**Minor**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

Minor 属性的取值范围在 0 到 9999 之间。

该属性提供运行中的应用程序的版本信息。

在设计时，使用位于“工程属性”对话框中的“生成”选项卡上的“次版本”框可设置该属性。

请参阅

Major 属性，Revision 属性

Minor 属性

返回一个 Long，它指定被引用的类型库的次版本号，此属性为只读。

应用于

Reference 对象

说明

由 Minor 属性所返回的代码，相当于存在曾引用过的类型库里的次版本号。

请参阅

Major 属性，Version 属性

示例

下列示例使用 Minor 属性返回某工程中指定的 Reference 对象的次版本号。

```
Debug.PrintApplication.VBE.VBProjects(1).References(1).Minor
```

Minute 函数

返回一个 Variant(Integer)，其值为 0 到 59 之间的整数，表示一小时中的某分钟。

语法

Minute(*time*)

必需的 time 参数，可以是任何能够表示时刻的 Variant、数值表达式、字符串表达式或它们的组合。如果 time 包含 Null，则返回 Null。

请参阅

Day 函数，Hour 函数，Now 函数，Second 函数，Time 函数，Time 语句

示例

本示例使用 Minute 函数转换指定的时间，得到小时后面的分钟数。在开发环境中，日期和时间原义会根据系统的地区设置，以短式日期和时间格式显示。

```
DimMyTime,MyMinute
MyTime=#4:35:17PM# '指定一时间。
MyMinute=Minute(MyTime) 'MyMinute 的值为 35。
```

MIRR 函数

返回一个 Double，指定一系列修改过的周期性现金流（支出或

收入）的内部利率。

语法

MIRR(*values(),finance_rate*,*reinvest_rate*)

MIRR 函数有下列命名参数：

部分	描述
values()	必需的。Double 数组，指定现金流值。此数组至少要包含一个负值（支付）和一个正值（收入）
finance_rate	必需的。Double 指定财务成本上的支付利率
reinvest_rate	必需的。Double 指定由现金再投资所得利率

说明

修改过的返回内部利率是指在用不同的利率计算支出和收入时的内部利率。MIRR 函数既考虑投资成本(finance_rate)，也考虑现金再投资所得利率(reinvest_rate)。

finance_rate 和 reinvest_rate 参数是用十进制数值表示的百分比。例如，0.12 表示百分之十二。

MIRR 函数用数组中的数值顺序来解释支付和收入的顺序。要确保支付和收入的输入顺序正确。

请参阅

DDB 函数，FV 函数，Ipmt 函数，IRR 函数，Nper 函数，NPV 函数，Pmt 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

本示例使用 MIRR 函数计算由数组 Values()内所含一系列期间现金

流量的修正过的实质报酬率。LoanAPR 代表财务成本上的支付利息利率，而 InvAPR 则代表由现金再投资上所得到的利息利率。

```
DimLoanAPR,InvAPR,Fmt,RetRate,Msg
StaticValues(5)AsDouble     '声明数组。
LoanAPR=.1 '贷款利率。
InvAPR=.12 '再投资利率。
Fmt="#0.00"     '定义金额格式。
Values(0)=-70000    '事业初始成本。
'正数现金流代表连续四年的收入状况。
Values(1)=22000:Values(2)=25000
Values(3)=28000:Values(4)=31000
RetRate=MIRR(Values(),LoanAPR,InvAPR)   '计算实质报酬率。
Msg="Themodifiedinternalrateofreturnforthesefivecashflowsis"
Msg=Msg&Format(Abs(RetRate)*100,Fmt)&"%."
MsgBoxMsg   '显示实质报酬率。
```

MiscFlags 属性

返回或设置一个值，确定存取一个或多个附加的 OLE 容器控件的特性。

应用于

OLEContainer 控件

语法

object.**MiscFlags**[=*value*]

MiscFlags 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	整数或常数，指定存取附加的特性，在“设置值”中有详细说明

设置值

value **的设置值是：**

常数	值	描述
VbOLEMiscFlagMemStorage	1	对象加载时，控件使用内存保存对象
VbOLEMiscFlagDisableInPlace	2	替代控件缺省的、允许对象现场激活的行为，该对象支持现场激活

说明

vbOLEMiscFlagMemStorage **标志设置比对象的缺省动作快，后者是将其作为临时文件保存到硬盘上。不过，如用了设置，对象会由于其数据需要很多空间而占用大量的内存，比如像画图程序的位图。**

如果对象支持现场激活，可以使用 vbOLEMiscFlagDisableInPlace **设置将对象在分隔窗口强制激活。**

为了组合各值，可使用 Or **操作符。例如，为了组合两个标志，**

可使用如下代码：

```
Ole1.MiscFlags=vbOLEMiscFlagMemStorageOr_vbOLEMiscFlagDisableInPlace
```

MkDir 语句

创建一个新的目录或文件夹。

语法

MkDirpath

必要的 *path* 参数是用来指定所要创建的目录或文件夹的字符串表达式。*path* 可以包含驱动器。如果没有指定驱动器，则 MkDir 会在当前驱动器上创建新的目录或文件夹。

请参阅

ChDir 语句，CurDir 函数，RmDir 语句

示例

本示例使用 MkDir 语句来创建目录或文件夹。如果没有指定驱动器，新目录或文件夹将会建在当前驱动器中。

```
MkDir"MYDIR" '建立新的目录或文件夹。
```

Mode 属性

返回一个值，其内容为指定工程当前所处的方式，此属性为只读。

应用于

VBProject 对象

返回值

Mode 属性返回下列这些值：

常数	描述
vbext_vm_Run	指定的工程处于运行方式
vbext_vm_Break	指定的工程处于中断方式
vbext_vm_Design	指定的工程处于设计方式

请参阅

ActiveVBProject 属性，Protection 属性

示例

下列示例使用 Mode 属性返回活动工程的模式。返回的值是事先定义好的代表工程模式的常量。

```
Debug.PrintApplication.VBE.ActiveVBProject.Mode
```

Month 函数

返回一个 Variant(Integer)，其值为 1 到 12 之间的整数，表示一年中的某月。

语法

Month(*date*)

必需的 *date* 参数，可以是任何能够表示日期的 Variant、数值表达式、字符串表达式或它们的组合。如果 *date* 包含 Null，则返

回 Null。

请参阅

Date 函数，Data 语句，Day 函数，Now 函数，Weekday 函数，Year 函数

示例

本示例使用 Month 函数来得知某个日期的月份。在开发环境中，日期原义会根据系统的地区设置，以短式日期格式显示。

```
DimMyDate,MyMonth
MyDate=#February12,1969#   '指定一日期。
MyMonth=Month(MyDate)  'MyMonth 的值为 2。
```

MonthName 函数

返回一个表示指定月份的字符串。

语法

MonthName(*month*[,*abbreviate*])

MonthName 函数语法有如下几部分：

部分	描述
month	必需的。月份的数值表示。例如一月是 1，二月是 2，等等
abbreviate	可选的。Boolean 值，表示月份名是否缩写。如果忽略，缺省值为 False，表明月份名不能被缩写

MouseDown、MouseUp 事件

这些事件是当按下(MouseDown)或者释放(MouseUp)鼠标按钮时发生。

应用于

ADOData 控件，TreeView 控件，ListView 控件，ProgressBar 控件，Slider
控件，StatusBar 控件，TabStrip 控件，ToolBar 控件，Animation 控件，UpDown 控件，DBCombo 控件，DBList 控件，MSHFlexGrid 控件，MSFlexGrid 控件，RemoteData 控件，RichTextBox 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，CommandButton 控件，DirListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件，OLEContainer 控件

语法

PrivateSubForm_MouseDown(*button*AsInteger,shiftAsInteger,xAsSingle,yAsSingle)

PrivateSubMDIForm_MouseDown(button**AsInteger**,*shift***AsInteger**,*x***AsSingle**,*y***AsSingle**)

PrivateSubobject_MouseDown([*index***AsInteger**,]*button***AsInteger**,*shift***AsInteger**,*x***AsSingle**,*y***AsSingle**)

PrivateSubForm_MouseUp(*button***AsInteger**,*shift***AsInteger**,*x***AsSingle**,*y***AsSingle**)

PrivateSubMDIForm_MouseUp(*button***AsInteger**,*shift***AsInteger**,*x***AsSingle**,*y***AsSingle**)

PrivateSubobject_MouseUp([*index***AsInteger**,]*button***AsInteger**,*shift***AsInteger**,*x***AsSingle**,*y***AsSingle**)

MouseDown **和** MouseUp **事件各种语法包含下列部分：**

部分	描述
Object	返回一个对象表达式，其值是“应用于”列表中的一个对象
Index	返回一个整数，用来唯一地标识一个在控件数组中的控件
Button	返回一个整数，用来标识该事件的产生是按下(MouseDown)或者释放(MouseUp)按钮引起的。button 参数是具有相应于左按钮（位 0），右按钮（位 1），以及中间按钮（位 2）的一个位字段。这些位的值分别等于 1，2，和 4。其中仅有一位被设置，指示出引起该事件的那个按钮
Shift	返回一个整数，在 button 参数指定的按钮被按下或者被释放的情况下，该整数相应于 SHIFT,CTRL 和 ALT 键的状态。某键被按下使得一个二进制位被设置。shift 参数是具有相应于 SHIFT 键（位 0），CTRL 键（位 1），以及 ALT 键（位 2）最少二进制位的一个位字段。这些位的值分别等于 1，2，和 4。shift 参数指示这些键的状态。这些位中可能有一些、全部或者一个也没有被设置，指示这些键中的一些、全部或者一个也没有被按下。例如，CTRL 和 ALT 键都被按下，则 shift 的值就是 6
x,y	返回一个指定鼠标指针当前位置的数。x 和 y 的值所表示的总是通过该对象 ScaleHeight,ScaleWidth,ScaleLeft,和 ScaleTop 属性所建立的坐标系统的方式

说明

为了在给定的一个鼠标按钮按下或释放时指定将引起的一些操作，应当使用 MouseDown 或者 MouseUp 事件过程。不同于 Click 和 DblClick 事件的是，MouseDown 和 MouseUp 事件能够区分出鼠标的左、右和中间按钮。也可以为使用 SHIFT，CTRL 和 ALT 等键盘换挡键编写用于鼠标——键盘组合操作的代码。

下列情况对 Click 和 DblClick 事件都适用：

如果鼠标按钮是当其指针在窗体或控件之上时被按下，则该对象将“捕获”鼠标并接收包括最后 MouseUp 事件在内的全部鼠标事件。这暗示了通过鼠标事件所返回的 x，y 鼠标指针坐标值，可以不总是在接收它们的对象的内部区域之内。

如果鼠标被持续地按下，则第一次按下之后捕获鼠标的对象将接收全部鼠标事件直至所有按钮被释放为止。

如果要测试 button 或 shift 参数，可以使用对象浏览器中的 VisualBasic(VB) 对象库中所列出的常数，用来定义该参数中的各个二进制位：

常数（按钮）	值	描述
vbLeftButton	1	左按钮被按下
vbRightButton	2	右按钮被按下
vbMiddleButton	4	中间按钮被按下

常数（换挡）	值	描述
vbShiftMask	1	SHIFT 键被按下
vbCtrlMask	2	CTRL 键被按下
vbAltMask	4	ALT 键被按下

随后这些常数作为位屏蔽，对于按钮的各种组合，无须计算各个组合的唯一的位字段即可进行测试。

注意可使用 MouseMove 事件过程对由于鼠标移动而引起的事件进行响应。MouseDown 和 MouseUp 所使用的 button 参数与 MouseMove 所使用的 button 参数是不同的。对于 MouseDown 和 MouseUp 来说，button 参数要精确地指出每个事件的一个按钮，而对于 MouseMove 来说，它指示的是所有按钮的当前状态。

请参阅

Click 事件，MouseMove 事件，Dblclick 事件，MousePointer 属性，Click 事件(ActiveX 控件)

示例

本例演示一个简单的绘图应用程序。当任意鼠标按钮按下并拖动时，MouseDown 事件过程和一个相关的 MouseMove 事件过程协同工作以启动绘图。MouseUp 事件过程禁止绘图。要尝试这个例子，可将代码粘贴到一个窗体的声明部分，然后按 F5 键，单击窗体并在鼠标按钮按下时移动鼠标。

```
DimPaintNowAsBoolean

PrivateSubForm_MouseDown(ButtonAsInteger,ShiftAsInteger,XAs
```

```
Single,YAsSingle)
    PaintNow=True     '启动绘图。
EndSub

PrivateSubForm_MouseUp(ButtonAsInteger,ShiftAsInteger,XAsSingle,
YAsSingle)
    PaintNow=False    '禁止绘图。
EndSub

PrivateSubForm_MouseMove(ButtonAsInteger,ShiftAsInteger,XAsSingl
e,YAsSingle)
    IfPaintNowThen
        PSet(X,Y)          '画一个点。
    EndIf
EndSub
PrivateSubForm_Load()
    DrawWidth=10      '使用更宽的刷子。
    ForeColor=RGB(0,0,255)   '设置绘图颜色。
EndSub
```

MouseIcon 属性

返回或设置自定义的鼠标图标。

应用于

TreeView 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，MSFlexGrid 控件，SSTab 控件，RichTextBox 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，Screen 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件，TextBox 控件，OLEContainer 控件

语法

object.**MouseIcon=LoadPicture***(*pathname*)**

object.**MouseIcon***[=picture]*

MouseIcon 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“适用于”列表中的一个对象
pathname	字符串表达式，指定包含自定义图标文件的路径和文件名
picture	Form 对象、PictureBox 控件、或 Image 控件的 Picture 属性

说明

MouseIcon 属性提供一个自定义图标，它在 MousePointer 属性设为

99 时使用。

MouseIcon 属性使程序能够很容易地访问自定义光标，它可以是任意大小并具有任何热点位置的光标。VisualBasic 不能装入动画光标(.ani)文件，即使 32-位版的 Windows 支持这些光标。

请参阅

LoadPicture 函数，SavePicture 语句，Icon 属性，Picture 属性，DragIcon 属属，MousePointer 属性，Picture 属性，(ActiveX 控件)

示例

本例说明 MouseIcon 属性是如何设置自定义鼠标图标的。要试用此例，先在窗体上创建一个 ListBox 控件，然后将 MultiSelect 属性设置为 1 或 2。在运行时期，能选择一个或多个项。根据选择的是单项还是多项，将显示不同的图标。

```
PrivateSubForm_Load()
   '在列表框中放置一些项。
   List1.AddItem"Selection1"
   List1.AddItem"Selection2"
   List1.AddItem"Selection3"
   List1.AddItem"Selection4"
   List1.AddItem"Selection5"
EndSub
PrivateSubList1_MouseDown(ButtonAsInteger,ShiftAsInteger,XAsSing
le,YAsSingle)
```

```
    '为多项设置自定义鼠标图标。
    IfList1.SelCount>1Then
        List1.MouseIcon=LoadPicture("ICONS\COMPUTER\MOUSE04.ICO")
        List1.MousePointer=99
    Else'为单项设置自定义鼠标图标。
        List1.MouseIcon=LoadPicture("ICONS\COMPUTER\MOUSE02.ICO")
        List1.MousePointer=99
    EndIf
EndSub
```

MouseMove 事件

此事件在移动鼠标时发生。

应用于

ADOData 控件，TreeView 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip，Toolbar 控件，Animation 控件，UpDown 控件，DBCombo 控件，DBList 控件，MSHFlexGrid 控件，MSFlexGrid 控件，RemoteData 控件，RichTextBox 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，CommandButton 控件，DirListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE

Container 控件

语法

PrivateSubForm_MouseMove(*button***AsInteger**,*shift***AsInteger**,*x***AsSingle**,
*y***AsSingle**)

PrivateSubMDIForm_MouseMove(*button***AsInteger**,*shift***AsInteger**,*x***AsS**
ingle,*y***AsSingle**)

PrivateSubobject_MouseMove([*index***AsInteger**,]*button***AsInteger**,*shift*
AsInteger,*x***AsSingle**,*y***AsSingle**)

MouseMove **事件语法包含下列部分：**

部分	描述
object	一个对象表达式，其值是“适用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件
button	一个整数，它对应鼠标各个按钮的状态，如果某个按钮按下，其中就有一个二进制位被设置。button 参数是具有相应于左按钮（位 0），右按钮（位 1），以及中间按钮（位 2）的一个位字段。这些位的值分别等于 1，2 和 4。它指示这些鼠标按钮的整体状态；三个二进制位中的一些、全部或一个也没有被设置，指示这些按钮中的一些、全部或一个也没有被按下
shift	一个整数，该整数相应于 SHIFT,CTRL,和 ALT 键的状态。某键被按下使得一个二进制位被设置。shift 参数是具有相应于 SHIFT 键（位 0），CTRL 键（位 1），以及 ALT 键（位 2）最少二进制位的一个位字段。这些位的值分别等于 1，2 和 4。shift 参数指示这些键的状态。这些位中可能有一些、全部或者一个也没有被设置，指示这些键中的一些、全部或者一个也没有被按下。例如，CTRL 和 ALT 键都被按下，则 shift 的值就是 6
x,y	一个指定鼠标指针当前位置的数。x 和 y 的值所表示的总是通过该对象 ScaleHeight,ScaleWidth,ScaleLeft 和 ScaleTop 属性所建立的坐标系统的方式

说明

MouseMove 事件伴随鼠标指针在对象间移动时连续不断地产生。

除非有另一个对象捕获了鼠标，否则，当鼠标位置在对象的边界范围内时该对象就能接收 MouseMove 事件。

要测试 *button* 或 *shift* 参数，可使用对象浏览器中的 VisualBasic(VB)对象库中所列出的常数，用来定义该参数中的各个位：

常数（按钮）	值	描述
vbLeftButton	1	左按钮按下
vbRightButton	2	右按钮按下
vbMiddleButton	4	中间按钮按下

常数（换挡）	值	描述
vbShiftMask	1	SHIFT 键按下
vbCtrlMask	2	CTRL 键按下
vbAltMask	4	ALT 键按下

然后这些常数用作位屏蔽，对于按钮的各种组合，无须计算出各个组合的唯一的位字段值即可进行检测。

要测试某一条件，首先将各个结果赋给一个临时整型变量然后再与一个位屏蔽的 *button* 或 *shift* 参数进行比较。测试应当用各个参数进行 And 运算，若结果大于零，则说明该键或按钮被按下。其操作如下：

```
LeftDown=(ButtonAndvbLeftButton)>0
CtrlDown=(ShiftAndvbCtrlMask)>0
```

然后，接下去可对结果的各种组合进行检测，其操作如下：

```
IfLeftDownAndCtrlDownThen
```

注意为了对鼠标按钮按下和释放所引起的事件进行处理，可使用 MouseDown 和 MouseUp 事件过程。

MouseMove 事件的 *button* 参数与 MouseDown 和 MouseUp 事件的 *button* 参数是不同的。对于 MouseMove 事件来说，*button* 参数指示的是所有按钮当前的状态；一个 MouseMove 事件可指示某些、全部或没有一个按钮被按下。对于 MouseDown 和 MouseUp 事件来说，*button* 参数在每个事件精确地指示一个按钮。

在 MouseMove 事件中任何时候移动窗口，都能引起层叠事件。当该窗口移动到指针下面时 MouseMove 事件将产生。即使是鼠标完全不动 MouseMove 事件也能产生。

请参阅

Click 事件，MouseDown，MouseUp 事件，DblClick 事件，MousePointer 属性，Click 事件(ActiveX 控件)

示例

本例演示一个简单的绘图应用程序。当任鼠标按钮按下并拖动时，MouseDown 事件过程和一个相关的 MouseMove 事件协同工作以启动绘图。MouseUp 事件过程禁止绘图。要尝试这个例子，可将代码粘贴到一个窗体的声明部分，然后按 F5 键，单击窗体并在鼠标按钮按下时移动鼠标。

```
DimPaintNowAsBoolean    '声明变量.
PrivateSubForm_MouseDown(ButtonAsInteger,ShiftAsInteger,XAsSingl
e,YAsSingle)
    PaintNow=True     '启动绘图。
EndSub
PrivateSubForm_MouseUp(ButtonAsInteger,XAsSingle,YAsSingle)
    PaintNow=False    '关闭绘图。
EndSub
PrivateSubForm_MouseMove(ButtonAsInteger,ShiftAsInteger,XAsSingl
e,YAsSingle)
    IfPaintNowThen
        PSet(X,Y)          '画一个点。
    EndIf
EndSub

PrivateSubForm_Load()
    DrawWidth=10       '使用更宽的刷子。
    ForeColor=RGB(0,0,255)   '设置绘图颜色。
EndSub
```

MousePointer 属性

返回或设置一个值，该值指示在运行时当鼠标移动到对象的一个

特定部分时，被显示的鼠标指针的类型。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Screen 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveLIstBox 控件，FileLIstBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件，OLEContainer 控件，Data 控件

语法

object.**MousePointer**[=*value*]

MousePointer 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	整数，按照设置值中的描述指定被显示的鼠标指针类型

设置值

value 的设置值为：

常数	值	描述
vbDefault	0	（缺省值）形状由对象决定
vbArrow	1	箭头
vbCrosshair	2	十字线（crosshair 指针）
vbIbeam	3	I 型
vbIconPointer	4	图标（矩形内的小矩形）
vbSizePointer	5	尺寸线（指向东、南、西和北四方向的箭头）
vbSizeNESW	6	右上-左下尺寸线（指向东北和西南方向的双箭头）
vbSizeNS	7	垂-直尺寸线（指向南和北的双箭头）
vbSizeNWSE	8	左上-右下尺寸线（指向东南和西北方向的双箭头）
vbSizeWE	9	水-平尺寸线（指向东和西两个方向的双箭头。
vbUpArrow	10	向上的箭头
vbHourglass	11	沙漏（表示等待状态）
vbNoDrop	12	不允许放下
vbArrowHourglass	13	箭头和沙漏
VbArrowQuestion	14	箭头和问号

VbSizeAll	15	四向尺寸线
VbCustom	99	通过 MouseIcon 属性所指定的自定义图标

说明

在鼠标指针越过窗体或对话框上的控件时，为了指出功能上的改变，可以使用该属性。沙漏标形状设置值(11)是很有用的，用来指示用户需要等待过程或操作的完成。

注意如果应用程序调用 DoEvents，那么 MousePointer 属性在经过 ActiveX 部件时可能暂时地改变。

请参阅

MouseMove 事件，DragIcon 属性，MouseIcon 属性

示例

本例当在屏幕上画圆时将鼠标指针改变为沙漏标，然后在过程结束时将沙漏标恢复为鼠标指针。要试用此例，先将下面的代码粘贴到窗体的声明部分。按 F5 键以运行该程序，然后单击窗体。

```
PrivateSubForm_Click()
   DimI'声明变量。
   '将鼠标指针改变为沙漏标。
   Screen.MousePointer=vbHourglass
'设置随机的颜色和在窗体上画圆。
   ForI=0ToScaleWidthStep50
        ForeColor=RGB(Rnd*255,Rnd*255,Rnd*255)
        Circle(I,ScaleHeight*Rnd),400
```

```
    Next
    '返回鼠标指针到正常状态。
    Screen.MousePointer=vbDefault
EndSub
```

Move 方法

用以移动 MDIForm、Form 或控件。不支持命名参数。

应用于

ADOData 控件，TreeView 控件，ImageCombo 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，ToolBar 控件，DataTimePicker 控件，Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，CoolBar 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，TextBox 控件 (Lightweight)，MaskedEdit 控件，MultimediaMCI 控件，MSChart 对象，MSHFlexGrid 控件，MSFlexGrid 控件，SSTab 控件，RemoteData 控件，RichTextBox 控件，Data 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，dirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，OLEContainer 控件

语法

object.**Move***left*,*top*,*width*,*height*

Move 方法的语法包含下列部分：

部分	描述
object	可选的参数。一个对象表达式，其值为“适用于”列表中的一个对象。如果省略 object，带有焦点的窗体缺省为 object
left	必要的。单精度值，指示 object 左边的水平坐标(x-轴)
top	可选的。单精度值，指示 object 顶边的垂直坐标(y-轴)
width	可选的。单精度值，指示 object 新的宽度
height	可选的。单精度值，指示 object 新的高度

说明

只有 left 参数是必须的。但是，要指定任何其它的参数，必须先指定出现在语法中该参数前面的全部参数。例如，如果不先指定 left 和 top 参数，则无法指定 width 参数。任何没有指定的尾部的参数则保持不变。

对于 Frame 控件中的窗体和控件，坐标系统总是用缇（twips）。移动屏幕上的窗体或移动 Frame 中的控件总是相对于左上角的原点(0,0)。移动 Form 对象或 PictureBox 中的控件（或 MDIForm 对象中的 MDI 子窗体）时，则使用该容器对象的坐标系统。坐标系统或度量单位是在设计时用 ScaleMode 属性设置。在运行时使用 Scale 方法可以更改该坐标系统。

请参阅

Height，Width 属性，Left，Top 属性，ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性

示例

本示例使用 Move 方法在屏幕上移动一个窗体。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimInch,Msg                                    '声明变量。
   Msg="ChooseOKtoresizeandmovethisformby"
   Msg=Msg&"changingthevalueofproperties."
   MsgBoxMsg                                      '显示信息。
   Inch=1440                                      '将英寸设置为缇。
   Width=4*Inch                                   '设置宽度。
   Height=2*Inch                                      '设置高度。
   Left=0                                         '将左边对准起点。
   Top=0                                          '将顶部对准起点。
   Msg="NowchooseOKtoresizeandmovethisform"
   Msg=Msg&"usingtheMovemethod."
   MsgBoxMsg                                      '显示信息。
   MoveScreen.Width-2*Inch,Screen.Height-Inch,2*Inch,Inch
EndSub
```

Move 方法（FileSystemObject 对象）

将一个指定的文件或文件夹从一个地方移动到另一个地方。

应用于

File 对象，Folder 对象

语法

object.**Move***destination*

Move 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 File 或 Folder 对象的名字
destination	必需的。文件或文件夹要移动到的目标。不允许有通配符

说明

Move 方法对一个 File 或 Folder 的结果和执行 FileSystemObject .MoveFile 或 FileSystemObject.MoveFolder 操作的结果是一样的。理应当注意，后面的方法能够移动多个文件或文件夹。

Moveable 属性

返回或设置一个值，该值指定了对象是否可移动。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

object.**Moveable**=*boolean*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
boolean	布尔表达式，指定对象是否可以移动

设置值

boolean 的设置值是：

常数	值	描述
True	-1	可移动对象
False	0	不可移动对象

MoveFile 方法

将一个或多个文件从一个地方移动到另一个地方。

应用于

FileSystemObject 对象

语法

object.**MoveFile***source*,*destination*

MoveFile 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
source	必需的。一个或多个要移动文件的路径。Source 参数字符串只能在路径的最后部件中包含通配符
destination	必需的。一个或多个文件要移动到的目标路径。Destination 参数不能包含通配符

说明

如果 *source* 包含通配符或 *destination* 以路径分隔符（\）为结尾，则认为 destination 指定了一个存在的文件夹，在此文件夹中移动相匹配的文件。否则，认为 destination 是一个要创建的目标文件名。在上面任一情况下，当移动一个文件时可能出现三种事件。

如果 *destination* 不存在，文件得到移动。这是通常的情况。

如果 *destination* 是一个已存在文件，则发生一个错误。

如果 *destination* 是一个目录，则发生一个错误。

如果一个在 *source* 中使用的通配符不能和任何一个文件匹配，也发生一个错误。MoveFile 方法停止在它遇到的第一个错误上。不要尝试回卷错误发生前所做的任何改变。

重点只有操作系统支持的情况下，这个方法才允许在卷之间移动文件。

请参阅

CopyFile 方法，DeleteFile 方法，Move 方法(FileSystemObject

对象)，MoveFolder 方法

MoveFolder 方法

将一个或多个文件夹从一个地方移动到另一个地方。

应用于

FileSystemObject 对象

语法

object.**MoveFolder***source*,*destination*

MoveFolder 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
source	必需的。一个或多个要移动的文件夹路径。Source 参数字符串只能在路径的最后部件中包含通配符
destination	必需的。一个或多个文件夹要移动到的目标路径。Destination 参数不能包含通配符

说明

如果 *source* 包含通配符或 *destination* 以路径分隔符（\）为结尾，则认为 destination 指定了一个已存在的文件夹，在此文件夹中移动相匹配的文件。否则，认为 destination 是一个要创建的目标文件夹名字。在上面的任一种情况下，当一个文件夹被移动时可能出现三种情况。

如果 *destination* 不存在，文件夹得到移动。这是通常的情况。

如果 *destination* 是一个已存在文件，则发生一个错误。

如果 *destination* 是一个目录，则发生一个错误。

如果 *source* 中使用的一个通配符不能和任何文件夹相匹配，也发生一个错误。MoveFolder 方法停止在它遇到的第一个错误上。不要尝试回卷在错误发生前所做的任何改变。

重点只有在操作系统支持的情况下，这个方法才允许文件夹在卷之间移动。

请参阅

CopyFolder 方法，CreateFolder 方法，DeleteFolder 方法，Move 方法(FileSystemObject 对象)，MoveFile 方法

MsgBox 函数

在对话框中显示消息，等待用户单击按钮，并返回一个 Integer 告诉用户单击哪一个按钮。

语法

MsgBox(*prompt*[,*buttons*][,*title*][,*helpfile*,*context*])

MsgBox 函数的语法具有以下几个命名参数：

部分	描述
prompt	必需的参数。字符串表达式，作为显示在对话框中的消息。prompt 的最大长度大约为 1024 个字符，由所用字符的宽度决定。如果 prompt 的内容超过一行，则可以在每一行之间用回车符(Chr(13))、换行符(Chr(10))或是回车与换行符的组合(Chr(13)&Chr(10))将各行分隔开来
buttons	可选的参数。数值表达式是值的总和，指定显示按钮的数目及形式，使用的图标样式，缺省按钮是什么以及消息框的强制回应等。如果省略，则 buttons 的缺省值为 0
title	可选的参数。在对话框标题栏中显示的字符串表达式。如果省略 title，则将应用程序名放在标题栏中
helpfile	可选的参数。字符串表达式，识别用来向对话框提供上下文相关帮助的帮助文件。如果提供了 helpfile，则也必须提供 context
context	可选的参数。数值表达式，由帮助文件的作者指定给适当的帮助主题的帮助上下文编号。如果提供了 context，则也必须提供 helpfile

设置值

buttons 参数有下列设置值：

常数	值	描述
vbOKOnly	0	只显示 OK 按钮
vbOKCancel	1	显示 OK 及 Cancel 按钮
vbAbortRetryIgnore	2	显示 Abort、Retry 及 Ignore 按钮
vbYesNoCancel	3	显示 Yes、No 及 Cancel 按钮
vbYesNo	4	显示 Yes 及 No 按钮
vbRetryCancel	5	显示 Retry 及 Cancel 按钮
vbCritical	16	显示 CriticalMessage 图标
vbQuestion	32	显示 WarningQuery 图标
vbExclamation	48	显示 WarningMessage 图标
vbInformation	64	显示 InformationMessage 图标
vbDefaultButton1	0	第一个按钮是缺省值
vbDefaultButton2	256	第二个按钮是缺省值
vbDefaultButton3	512	第三个按钮是缺省值
vbDefaultButton4	768	第四个按钮是缺省值
vbApplicationModal	0	应用程序强制返回；应用程序一直被挂起，直到用户对消息框作出响应才继续工作
vbSystemModal	4096	系统强制返回；全部应用程序都被挂起，直到用户对消息框作出响应才继续工作
vbMsgBoxHelpButton	16384	将 Help 按钮添加到消息框

vbMsgBoxSetForeground	65536	指定消息框窗口作为前景窗口
vbMsgBoxRight	524288	文本为右对齐
vbMsgBoxRtlReading	1048576	指定文本应为在希伯来和阿拉伯语系统中的从右到左显示

第一组值(0–5)描述了对话框中显示的按钮的类型与数目；第二组值(16,32,48,64)描述了图标的样式；第三组值(0,256,512)说明哪一个按钮是缺省值；而第四组值(0,4096)则决定消息框的强制返回性。将这些数字相加以生成 buttons 参数值的时候，只能由每组值取用一个数字。

注意这些常数都是 VisualBasicforApplications(VBA)指定的。结果，可以在程序代码中到处使用这些常数名称，而不必使用实际数值。

返回值

常数	值	描述
vbOK	1	OK
vbCancel	2	Cancel
vbAbort	3	Abort
vbRetry	4	Retry
vbIgnore	5	Ignore
vbYes	6	Yes
vbNo	7	No

说明

在提供了 *helpfile* 与 *context* 的时候，用户可以按 F1 (Windows) or HELP (Macintosh) 来查看与 context 相应的帮助主题。像 Microsoft Excel 这样一些主机应用程序也会在对话框中自动添加一个 Help 按钮。

如果对话框显示 Cancel 按钮，则按下 ESC 键与单击 Cancel 按钮的效果相同。如果对话框中有 Help 按钮，则对话框中提供有上下文相关的帮助。但是，直到其它按钮中有一个被单击之前，都不会返回任何值。

注意如果还要指定第一个命名参数以外的参数，则必须在表达式中使用 MsgBox。为了省略某些位置参数，必须加入相应的逗号分界符。

请参阅

InputBox 函数

示例

本示例使用 MsgBox 函数，在具有“是”及“否”按钮的对话框中显示一条严重错误信息。示例中的缺省按钮为“否”，MsgBox 函数的返回值视用户按哪一个钮而定。本示例假设 DEMO.HLP 为一帮助文件，其中有一个内容代码为 1000。

```
Dim Msg, Style, Title, Help, Ctxt, Response, MyString
Msg = "Do you want to continue ?"          ' 定义信息。
Style = vbYesNo + vbCritical + vbDefaultButton2  ' 定义按钮。
Title = "MsgBox Demonstration"             ' 定义标题。
```

```
Help="DEMO.HLP"                          '定义帮助文件。
Ctxt=1000                                '定义标题
        '上下文。
        '显示信息。
Response=MsgBox(Msg,Style,Title,Help,Ctxt)
IfResponse=vbYesThen                        '用户按下“是”。
   MyString="Yes"                        '完成某操作。
Else                                     '用户按下“否”。
   MyString="No"                             '完成某操作。
EndIf
```

MultiLine 属性

返回或设置一个值，该值指示 TextBox 控件是否能够接受和显示多行文本。在运行时是只读的。

应用于

TextBox 控件

语法

object.**MultiLine**

object 所在处代表一个对象表达式，其值是“适用于”列表中的一个对象。

设置值

MultiLine 属性设置值有：

设置值	描述
True	允许多行文本
False	（缺省值）忽略回车符并将数据限制在一行内

说明

当键入文本超出文本框时多行 TextBox 控件将使正文卷绕。

使用 ScrollBars 属性也能够在 TextBox 控件中加入滚动条来加大 TextBox 控件的显示范围。如果没有指定水平滚动条，那么在多行 TextBox 中文本将自动地卷绕。

注意在一个没有缺省按钮的窗体上，在多行 TextBox 控件中按下 ENTER 将把焦点移动到下一行。如果有缺省按钮存在，那么必须按下 CTRL+ENTERE 才能移动到下一行。

请参阅

Default 属性，ScrollBars 属性

MultiSelect 属性

返回或设置一个值，该值指示是否能够在 FileListBox 或 ListBox 控件中进行复选以及如何进行复选。在运行时是只读的。

应用于

FileListBox 控件，ListBox 控件

语法

object.MultiSelect

object 所在处代表一个对象表达式，其值是“适用于”列表中的

一个对象。

设置值

MultiSelect 属性设置值是：

设置值	描述
0	（缺省值）不允许复选
1	简单复选。鼠标单击或按下 SPACEBAR（空格键）在列表中选中或取消选中项。（箭头键移动焦点。）
2	扩展复选。按下 SHIFT 并单击鼠标或按下 SHIFT 以及一个箭头键（上箭头、下箭头、左箭头、和右箭头）将在以前选中项的基础上扩展选择到当前选中项。按下 CTRL 并单击鼠标来在列表中选中或取消选中项

示例

本例用屏幕字体的名字填充 ListBox 控件，并说明 MultiSelect 属性是如何影响 ListBox 行为的。要试用此例，在窗体上创建两个 ListBox 控件以及一个 CommandButton 控件。在第一个 ListBox 中，将 MultiSelect 属性设置为 1 或 2。在运行时期，在第一个 ListBox 中选择几个项，然后单击 CommandButton。所有被选的项都显示在第二个 ListBox 中。使用不同的 MultiSelect 属性设置值来运行几次这个例子。将下面的代码粘贴到声明部分，然后按 F5 键以运行这个程序。

```
PrivateSubForm_Load()
   DimI                                    '声明变量。
```

```
    '用屏幕字体名字填充列表框。
    ForI=0ToScreen.FontCount-1
        List1.AddItemScreen.Fonts(I)
    NextI
EndSub

PrivateSubCommand1_Click()
    DimI                                    '声明变量。
    '清除列表中所有的项。
    List2.Clear
    '如果一个项被选中，那么将它加入到List2。
    ForI=0ToList1.ListCount-1
        IfList1.Selected(I)Then
            List2.AddItemList1.List(I)
        EndIf
    NextI
EndSub
```

Name 属性

?返回在代码中用于标识窗体、控件或数据访问对象的名字。在运行时是只读的。

?返回或设置字体对象的名字。

应用于

CommonDialog 控件，Function 控件(数据报表设计程序)，Image 控件(数据报表设计程序)，Label 控件(数据报表设计程序)，Line 控件(数据报表设计程序)，Section 对象(DataReportDesigner)，Shape 控件(数据报表设计程序)，TextBox 控件(数据报表设计程序)，Extender 对象，PropertyPage 对象，UserControl 对象，UserDocument 对象，Class，Font 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DrivListBox Control，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件，Line 控件，ListBox 控件，MDIForm 对象，Menu 控件，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，Timer 控件，OLEContainer 控件，TreeView 控件，ImageCombo 控件，ImageList 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，ToolBar 控件，DAtaTimePicker 控件，Animation 控件，FlatScrollBar 控件，MonthVeiw 控件，UpDown 控件，CoolBar 控件，MSComm 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MAPISession 控件，MAPIMessage 控件，MaskedEdit 控件，SSTab 控件，PictureClip 控件，RichTextBox 控件，EventInfo 对象，Parameter 对象(VisualBasic)，Data 控件

语法

object.**Name**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

如果 object 被删去，则与活动窗体模块相联系的窗体被认为是 *object*。

说明

新对象的缺省名字由对象类型加上一个唯一的整数组成。例如，第一个新的 Form 对象是 Form1，一个新的 MDIForm 对象是 MDIForm1，以及在窗体上创建的第三个 TextBox 控件是 Text3。

一个对象的 Name 属性必须以一个字母开始并且最长可达 40 个字符。它可以包括数字和带下划线(_)的字符，但不能包括标点符号或空格。窗体不能具有与别的公共对象相同的名字，例如 Clipboard、Screen 或 App。虽然 Name 属性设置可以是一个关键字、属性名字、或别的对象的名字，但这会在你的代码中产生冲突。

能够在运行时与 Dim 语句一起使用一个窗体的 Name 属性，以此创建该窗体的其它实例。在设计时不能有两个窗体有相同的名字。

能够通过设置 Name 属性为相同的值来创建相同类型的控件数组。例如，当将群组中的所有选项按钮的名字设置为 MyOpt 时，VisualBasic 将为每个控件的 Index 属性分配一个唯一的值以便使数组中的控件相互区分。不同类型的两个控件不能共享相同的名字。

注意虽然VisualBasic经常将Name属性设置作为Caption、LinkTopic

和 Text 属性的缺省值使用，但是这些属性中一个的改变对别的属性并没有影响。

然而，更改窗体或其它模块 Name 属性值的大小写而不更改名称本身，下一次包含该窗体或模块的工程加载时，会造成“名称冲突”错误消息。例如，将“Form1”改为“form1”将会引起错误；而将“Form1”改为“formX”则不会。

错误是由模块名在工程文件中存储的方式引起的–在工程文件内部更改名称的过程是不分大小写的，而在工程加载中读取名称的过程是区分大小写的。

请参阅

Text 属性，LinkTopic 属性，ActiveForm 属性，Caption 属性，Index 属性(ControlArray)，Caption 属性(Tab 对象)，Text 属性(ActiveX 控件)，Caption 属性(ActiveX 控件)

Name 属性(FileSystemObject 对象)

设置或返回指定文件或文件夹名。读/写属性。

应用于

File 对象，Folder 对象

语法

object.**Name**[=*newname*]

Name 属性有下列几部分：

部分	描述
object	必需的。总是某个 File 或 Folder 对象的名字
newname	可选的。如果提供的话，newname 是指定的 object 的新名

说明

下面的代码举例说明了 Name 属性的用法：

```
SubShowFileAccessInfo(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFile(filespec)
s=f.Name&"onDrive"&UCase(f.Drive)&vbCrLf
s=s&"Created:"&f.DateCreated&vbCrLf
s=s&"LastAccessed:"&f.DateLastAccessed&vbCrLf
s=s&"LastModified:"&f.DateLastModified
MsgBoxs,0,"FileAccessInfo"
EndSub
```

请参阅

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，ParentFolder 属性，Path 属性，ShortName 属性，ShortPath 属性，Size 属性，SubFolders 属性，Type 属性

Name 属性（VBA 外接程序对象模型）

返回或设置一个 String，其内容为一个在代码中用来识别对象的名称。对 VBProject 对象及 VBComponent 对象，可读/写；对 Property 对象及 Reference 对象，此属性为只读。

应用于

Property 对象，Reference 对象，VBComponent 对象，VBProject 对象

说明

下面这张表格描述 Name 属性设置在不同的对象中如何应用。

对象	使用 Name 属性设置值的结果
VBProject	返回或设置活动的工程的名称。
VBComponent	返回或设置部件名称。若试图将 Name 属性设置成一个已经被使用或无效的名称，将可生成一个错误。
Property	返回出现在 PropertyBrowser 中的属性名。这个值用来索引 Properties 集合。此名称不能被设置。
Reference	返回代码中的引用的名称。此名称不能被设置。

一个新对象的默认名称是这个对象的类型再加上一个唯一的整型。譬如，第一个新窗体对象就是 Form1，一个新的窗体对象就是 Form1，而你在一个窗体中创建的第三个文本框控件就是 TextBox3。

一个对象的 Name 属性必须以一个英文字母开始且最多 40 个字

符。它可以包含数字及下划线(_)字符但不可包含标点符号或空白字符。窗体及模块不能与另一个公共对象如 Clipboard、Screen 或 App 具有相同名称。虽然 Name 属性可以是一个关键字、属性名或是另一对象的名称，但这会在代码中造成一些冲突。

请参阅

Description 属性，Type 属性(VBA 外接程序模型)

示例

下列示例使用 Name 属性返回某工程中指定的 VBComponents 集合中某个成员的名称。

```
Debug.PrintApplication.VBE.VBProjects(1).VBComponents(1).Name
```

Name 属性(WebClass、WebItem)

设置或返回名称，用于在代码中标识 WebClass 或 WebItem 对象。在运行时是只读的。

应用于

WebClass 对象，WebItem 对象

语法

object.**Name**=*[value]*

部分	描述
Object	对象表达式，其值是“适用于”列表中的一个对象
value	字母数字值；名称必须以字母起头，并且最多能够有 40 个字符。可以包括数字和下划线(_)字符，但不能包括标点符号或空格

说明

新对象的缺省名称是对象类型加上一个唯一的整数。例如，第一个新的 WebClass 对象是 WebClass1。

Name 语句

重新命名一个文件、目录、或文件夹。

语法

Name*oldpathname* **As***newpathname*

Name 语句的语法具有以下几个部分：

部分	描述
oldpathname	必要参数。字符串表达式，指定已存在的文件名和位置，可以包含目录或文件夹以及驱动器
newpathname	必要参数。字符串表达式，指定新的文件名和位置，可以包含目录或文件夹、以及驱动器。而由 newpathname 所指定的文件名不能存在

说明

Name 语句重新命名文件并将其移动到一个不同的目录或文件夹中。如有必要，Name 可跨驱动器移动文件。但当 newpathname 和 oldpathname 都在相同的驱动器中时，只能重新命名已经存在的目录或文件夹。Name 不能创建新文件、目录或文件夹。

在一个已打开的文件上使用 Name，将会产生错误。必须在改变名称之前，先关闭打开的文件。Name 参数不能包括多字符(*)和单字符(?)的通配符。

请参阅

Kill 语句

示例

本示例使用 Name 语句来更改文件的名称。示例中假设所有使用到的目录或文件夹都已存在。在 Macintosh 中，默认驱动器名称是"HD"并且路径部分由冒号取代反斜线隔开。

```
DimOldName,NewName
OldName="OLDFILE":NewName="NEWFILE"      '定义文件名。
NameOldNameAsNewName                     '更改文件名。

OldName="C:\MYDIR\OLDFILE":NewName="C:\YOURDIR\NEWFILE"
NameOldNameAsNewName                     '更改文件名，并移动文件。
```

NavigateTo 方法

执行到特定目标的超链接跳转。

应用于

Hyperlink 对象

语法

object.**NavigateTo** *Target*[,*Location*[,*FrameName*]]

NavigateTo 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值为“适用于”列表中的对象
target	字符串表达式，它指定跳转的位置。该位置可以是文档或者是 URL
location	字符串表达式，它指定跳转到 Target 指定的 URL 中的位置。如未指定 Location，则将跳转到缺省的文档中
FrameName	字符串表达式，它指定跳转到 Target 指定的 URL 中的帧。如未指定 FrameName，则将跳转到缺省的帧

说明

如果对象在支持 OLE 超链接的容器中，则容器将跳转到指定的位置。如果对象在不支持 OLE 超链接的容器中，则将启动一个注册为支持超链接的应用程序来处理这个请求。

如果 Target 未能指定有效的位置，则将产生错误。

NegotiateMenus 属性

设置一个值，决定窗体是否把其上对象的菜单合并到窗体菜单栏

上。运行时无效。

应用于

PictureBox 控件

设置值

NegotiateMenus 属性设置如下：

设置	描述
True	（缺省）当窗体的活动对象可编辑时，此对象的菜单显示在窗体的菜单栏上
False	窗体对象的菜单不在其菜单栏上显示

说明

使用 NegotiateMenus 属性，决定窗体的菜单栏是否与窗体活动对象的菜单共用（协商）空间。如果不想这样，可把 NegotiateMenus 设为 False。

无法在 MDIForm 对象和其上的对象之间协商菜单位置。

如果 NegotiateMenus 设为 True，此窗体必须有一个定义的菜单栏，即使该菜单栏不可见。如果窗体的 MDIChild 属性设为 True，则活动对象的菜单显示在 MDI 父窗口菜单栏中（MDIForm 对象）。

当 NegotiateMenus 设为 True 时，使用单个 Menu 控件的 NegotiatePosition 属性可决定窗体菜单和活动对象的菜单一起显示。

请参阅

NegotiateToolbars 属性，Align 属性

NegotiateMenus 属性

设定一个值来决定窗体是否将菜单与在窗体菜单栏上窗体的对象合并。运行时无效。

应用于

Form 对象,Forms 集合

设置值

NegotiateMenus 属性有如下设置值：

设置值	描述
True	(缺省值)当窗体上一个对象是活动的编辑状态时，对象菜单显示在窗体菜单栏上
False	窗体上的对象菜单不显示在窗体菜单栏上

说明

使用 NegotiateMenus 属性，决定窗体菜单栏是否与窗体上活动对象的菜单享（或协商）空间。如果不要包括在窗体菜单栏上的活动对象的菜单，设定 NegotiateMenus 为 False。

不能在 MDIForm 对象和 MDIForm 上的对象之间协商菜单。如果 NegotiateMenus 设定为 True,窗体必须有一个定义的菜单栏，即使菜单栏不可见了。如果窗体的 MDIChild 设定为 True,活动对象的菜单显示在 MDI 父窗口（MDIForm 对象）的菜单栏上。

当 NegotiteMenus 设定为 True 时，可以用单个菜单控件的 NegotiatePosition 属性决定与窗体一起显示的活动对象菜单的菜单。

请参阅

NegotiatePosition 属性

NegotiatePosition 属性

设置一个值，决定当窗体的链接对象或内嵌对象活动而且显示菜单时，是否在菜单栏显示最上层 Menu 控件。运行时无效。

应用于

Menu 控件

设置值

NegotiatePosition 属性设置如下：

设置	描述
0	（缺省）无。对象活动时，菜单栏上不显示菜单
1	左。对象活动时，菜单显示在菜单栏的左端
2	中。对象活动时，菜单显示在菜单栏的中间
3	右。对象活动时，菜单显示在菜单栏的右端

说明

用 NegotiatePosition 属性，决定窗体菜单栏的单个菜单与窗体活动对象的菜单共用（或协商）菜单栏空间。所有 NegotiatePosition 为非零值的菜单与活动对象的菜单在窗体的菜单栏上一起显示。如果 NegotiateMenus 属性设为 False，则该属性的设置不起作用。

请参阅

NegotiateMenus 属性

NegotiateToolbars 属性

设置一个值，决定当 MDI 子窗体对象活动时，MDI 子窗体上对象的工具栏是否在 MDIForm 上显示。运行时无效。

应用于

MDIForm 对象

设置值

NegotiateToolbars 属性设置如下：

设置	描述
True	（缺省）MDIForm 对象在 MDIForm 的顶部或底部显示活动对象的工具栏。活动对象决定是否在 MDIForm 的顶部或底部显示工具栏
False	活动对象的工具栏根本不显示或只作为浮动工具板显示，由活动对象决定

说明

当创建一个包括 MDI 子窗体对象的多文档窗口(MDI)应用程序时，使用 NegotiateToolbars 属性。该属性决定活动对象如何显示它的工具栏。该属性设为 True 时，MDIForm 共用（或协商）窗体的顶部或底部空间来显示活动对象的工具栏。

如果 MDIForm 也含有一个工具栏，用 Negotiate 属性决定各种工具栏如何共享可用空间。

请参阅

Negotiate 属性

NewIndex 属性

返回最近加入 ComboBox 或 ListBox 控件的项的索引。在运行时是只读的。

应用于

ComboBox 控件，ListBox 控件

语法

object.**NewIndex**

object 所在处代表一个对象表达式，其值是“适用于”列表中的一个对象。

说明

当需要一个与 ItemData 属性数组中的每个项相对应的值的列表时，可以与排序列表一起使用该属性。当往排序的列表中加入一个项时，VisualBasic 将在列表中按字母顺序插入一项。该属性表示项被插入的位置，以便能够在 ItemData 属性中的相同索引处插入相应的值。

如果在列表中已没有项或在最后的项被加入之后一个项被删除，那么 NewIndex 属性将返回-1。

请参阅

AddItem 方法，Clear 方法(Clipboard，ComboBox，LIstBox)，

RemoveItem 方法，List 属性，ListCount 属性，ItemData 属性，MultiSelect 属性，Selected 属性，TopIndex 属性

NewPage 方法

用以结束 Printer 对象中的当前页并前进到下一页。

应用于

Printer 对象，Printers 集合

语法

object.**NewPage**

object 所在处代表一个对象表达式，其值为“适用于”列表中的一个对象。

说明

NewPage 前进到下一个打印机页，并将打印位置重置到新页的左上角。调用 NewPage 时，它将 Printer 对象的 Page 属性加 1。

请参阅

EndDoc 方法，KillDoc 方法，Page 属性

示例

本示例使用 NewPage 方法在一页上打印出一行居中正文后开始一个新的打印页。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimHWidth,HHeight,I,Msg              '声明变量。
```

```
    OnErrorGoToErrorHandler             '设置错误处理程序。
    Msg="Thisisprintedonpage"
    ForI=1To2    '设置 2 个迭代。
        HWidth=Printer.TextWidth(Msg)/2 '取半宽。
        HHeight=Printer.TextHeight(Msg)/2    '取半高。
        Printer.CurrentX=Printer.ScaleWidth/2-HWidth
        Printer.CurrentY=Printer.ScaleHeight/2-HHeight
        Printer.PrintMsg&Printer.Page&"."    '打印。
        Printer.NewPage             '发送新页。
    NextI
    Printer.EndDoc                          '打印完毕。
    Msg="Twopages,eachwithasingle,centeredlineoftext,"
    Msg=Msg&"havebeensenttoyourprinter."
    MsgBoxMsg                               '显示信息。
    ExitSub
ErrorHandler:
    MsgBox"Therewasaproblemprintingtoyourprinter."
    ExitSub
EndSub
```

NextItem 属性

返回当前 WebClass **对象中的另一个** WebItem **对象。用于在单个请**

求期间将处理从一个 WebItem 变换到另一个。

应用于

WebClass 对象

语法

object.**NextItem**

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

在 WebClass 中激发事件之前，该属性被设置成 Nothing。完成一个事件过程的执行时，如果该属性被设置成 WebClass 中另一个 WebItem，则激发 WebItem 的 Respond 事件。该属性可以用在 WebClass 的 Start 事件中，以显示初始的 WebItem。NextItem 的值在以下事件中被忽略：

EndRequest

ProcessTag

FatalErrorResponse

请参阅

第 3 章的“标准 Webclass 事件”，第五部分的“开发 IIS 应用程序”，《MicrosoftVisualBasic6.0 部件工具指南》的“构造 Internet 应用程序”。

NonModalAllowed 属性

返回一个值，该值指示了是否可以没有模式地（无模式地）显示窗体。在设计时该属性是无效的。

应用于

App 对象

语法

object.**nonModalAllowed**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

返回值类型

Boolean

Nothing

使用 Nothing 关键字被将对象变量从实际对象中分离开来。要使用 Set 语句将 Nothing 赋值给对象变量。例如：

```
SetMyObject=Nothing
```

几个对象变量可以引用同一个实际对象。当 Nothing 被赋值给一个对象变量时，该变量不再引用任何实际对象。当几个对象变量引用同一个对象时，只有将全部对象变量都设置成 Nothing 之后，与被引用的对象有关联的内存资源及系统资源才会被释放掉，在这里，或者明确使用 Set，或者在最后一个设置成 Nothing 的对象

变量超出范围后隐含地使用 Set。

请参阅

Dim 语句，Private 语句，Public 语句，Set 语句

Now 函数

返回一个 Variant(Date)，根据计算机系统设置的日期和时间来指定日期和时间。

语法

Now

请参阅

Date 函数，Date 语句，Day 函数，Hour 函数，Minute 函数，Month 函数，Second 函数，Time 函数，Time 语句，Weekday 函数，Year 函数

示例

本示例使用 Now 函数返回系统当前的日期与时间。

```
DimToday
Today=Now  '将系统当前的日期与时间给变量。
```

NPer 函数

返回一个 Double，指定定期定额支付且利率固定的总期数。

语法

NPer(*rate*,*pmt*,*pv*[,*fv*[,*type*]])

NPer 函数有下列命名参数：

部分	描述
rate	必需的。Double 指定每一期的利率。例如，如果有一笔贷款年百分率(APR)为百分之十并按月付款的汽车贷款，则每一期的利率为 0.1/12 或 0.0083
pmt	必需的。Double 指定每一期所付金额。付款金额通常包含本金和利息，且付款金额在年金的有效期间不变
pv	必需的。Double 指定未来一系列付款或收款的现值。例如，当贷款买一辆汽车时，向贷方所借贷的金额为将来每月偿付给贷方款项的现值
fv	可选的。Variant 指定在付清贷款后所希望的未来值或现金结存。例如，贷款的未来值在贷款付清后为 0 美元。但是，如果想要在 18 年间存下 50，000 美元作为子女教育基金，那么 50，000 美元为未来值。如果省略的话，缺省值为 0
type	可选的。Variant 指定贷款到期时间。如果贷款是在贷款周期结束时到期，请使用 0，如果贷款是在周期开始时到期，请使用 1。如果省略的话，缺省值为 0

说明

年金是在一段时间内一系列固定现金支付。年金可以是贷款（如房屋抵押贷款），也可以是一笔投资（如按月储蓄计划）。

对所有参数，用负数表示现金支出（如储蓄存款），而用正数表示现金收入（如红利支票）。

请参阅

DDB 函数，FV 函数，IPmt 函数，IRR 函数，MIRR 函数，NPV 函数，Pmt 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

本示例使用 NPer 函数计算分期偿还贷款之总期数，贷款金额为 PVal。计算时尚需给定每期利率（APR/12），每期付款金额（Payment），贷款的未来值（FVal）及付款方式，以数值表示期初或期末付款（PayType）。

```
DimFVal,PVal,APR,Payment,PayType,TotPmts
ConstENDPERIOD=0,BEGINPERIOD=1 '付款方式。
FVal=0                                    '对贷款而言通常为零。
PVal=InputBox("Howmuchdoyouwanttoborrow?")
APR=InputBox("Whatistheannualpercentagerateofyourloan?")
IfAPR>1ThenAPR=APR/100                    '确保格式正确。
Payment=InputBox("Howmuchdoyouwanttopayeachmonth?")
PayType=MsgBox("Doyoumakepaymentsattheendofmonth?",vbYesNo)
IfPayType=vbNoThenPayType=BEGINPERIODElsePayType=ENDPERIOD
TotPmts=NPer(APR/12,-Payment,PVal,FVal,PayType)
IfInt（TotPmts）<>TotPmtsThenTotPmts=Int（TotPmts）+1
MsgBox"Itwilltakeyou"&TotPmts&"monthstopayoffyourloan."
```

NPV 函数

返回一个 Double，指定根据一系列定期的现金流（支付和收入）和贴现率而定的投资净现值。

语法

NPV*(rate*,*values())*

NPV 函数有下列命名参数：

部分	描述
Rate	必要。Double 指定在一期间内的贴现率，用十进制表示
Values()	必要。Double 数组指定现金流值。此数组至少要包含一个负值（支付）和一个正值（收入）

说明

投资的净现值是未来一系列支付或收入的当前价值。

NPV 函数使用数组中数值的顺序来解释支付和收入的顺序。要确保支付和收入值是用正确的顺序输入的。

NPV 投资在第一笔现金流值之前开始计算周期，而结束于数组中最后的现金流值。

净现值是根据未来的现金流进行计算的。如果第一笔现金流在第一期开始时发生，那么 NPV 返回的值必须加上第一笔值才是净现值。而且 *values()*数组不可包含第一笔值。

NPV 函数与 PV 函数（现值）相似，只是 PV 函数在一个期间的开始或结束时才允许有现金流。与可变的 NPV 现金流值不同，PV

的现金流在整个投资期间必须固定。

请参阅

DDB 函数，FV 函数，IPmt 函数，IRR 函数，MIRR 函数，Nper 函数，Pmt 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

本示例使用 NPV 函数根据数组 Values()内所含的一系列现金流量计算投资之净现值。RetRate 表示固定之实质报酬率。

```
DimFmt,Guess,RetRate,NetPVal,Msg
StaticValues(5)AsDouble          '声明数组。
Fmt="###,##0.00"                 '定义金额格式。
Guess=.1                  'Guess 从 10%开始。
RetRate=.0625                '设置固定实质报酬率。
Values(0)=-70000                 '事业初始成本。
'正数现金流代表连续四年的收入状况。
Values(1)=22000:Values(2)=25000
Values(3)=28000:Values(4)=31000
NetPVal=NPV(RetRate,Values())  '计算投资净现值。
Msg="Thenetpresentvalueofthesecashflowsis"
Msg=Msg&Format(NetPVal,Fmt)&"."
MsgBoxMsg                        '显示投资净现值。
```

Null

Null 关键字被用来作为 Variant 子类型。它说明变量不包含有效数据。

请参阅

VarType 函数

NullValue 属性

设置或返回一个值，用于格式化或去除格式化一个 Null 值。在设计时或运行时都可读写。

应用于

StdDataFormat 对象

语法

object.**NullValue**[=*value*]

NullValue 属性的语法有如下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
value	可选的变体型。在格式化时，如果数据为 Null，则返回 value。当数据返回到数据库时，如果该数据与 value 相匹配，则写入 Null

说明

如果 Type 属性设置为 fmtGeneral，则它将被忽略。每次取得一个空字段时都将读 NullValue 属性。

请参阅

FalseValue 属性，TrueValue 属性，Type 属性

Number 属性

返回或设置表示错误的数值。Number 是 Err 对象的缺省属性。可读/可写。

应用于

Err 对象

说明

从对象返回用户自定义的错误时，把被选作错误代码的数与 vbObjectError 常数相加，并由此设置 Err.Number。例如，用下列代码返回作为错误代码的数字 1051：

```
Err.RaiseNumber:=vbObjectError+1051,Source:=SomeClass
```

请参阅

HelpContextID 属性(VBA 外接程序对象模型)，Err 对象，Description 属性，
HelpContext 属性，HelpFile 属性，LastDLLerror 属性，Source 属性

示例

第一个示例说明 Number 属性在错误处理过程中的典型用法。第

二个示例则检查Err对象的Number属性,以确定由某个Automation对象所返回的错误代号为该对象定义的，或是 VisualBasic 所定义的。请注意，常数 vbObjectError 为一个绝对值很大之负数，通常由 OLEAutomation 对象所定义的错误代号会加上这个值，而由 VisualBasic 所定义的就不会。所以，将 Err.Number 减去 vbObjectError 后便可判断是否为 OLEAutomation 对象定义的错误代号（引用代码中的注释）。示例中如果错误代号为 OLEAutomation 对象定义的，则该错误代号便是 MyError 的值，并会和错误来源一起显示出来。如果 Err.Number 为 VisualBasic 定义的错误代号，则只将VisualBasic错误代号显示出来。

```
'Number 属性的典型用法
Subtest()
   OnErrorGoToout

   Dimx,y
   x=1/y    '引发一个“除以零”的错误
   ExitSub
   out:
   MsgBoxErr.Number
   MsgBoxErr.Description
   '检查是否发生“除以零”的错误
   IfErr.Number=11Then
```

```
        y=y+1
    EndIf
    Resume
EndSub


'使用 Number 属性来判断 Automation 对象返回的错误代号
DimMyError,Msg
'首先，减去 OLEAutomation 对象所加上的常数。
MyError=Err.Number-vbObjectError
'如果减去常数 vbObjectError 后的值仍然在
'0 到 65,535 之间，便是 OLEAutomation 对象定义的错误代号。
IfMyError>0AndMyError<65535Then
    Msg="Theobjectyouaccessedassignedthisnumbertotheerror:"_
            &MyError&".Theoriginatoroftheerrorwas:"_
            &Err.Source&".PressF1toseeoriginator'sHelptopic。"
'否则便是 VisualBasic 定义的错误代号。
Else
    Msg="Thiserror(#"&Err.Number&")isaVisualBasicerror"&_
            "number.PressHelpbuttonorF1fortheVisualBasicHelp"_
            &"topicforthiserror."
EndIf
    MsgBoxMsg,,"ObjectError",Err.HelpFile,Err.HelpContext
```

NumIndices 属性

该属性返回被 Property 对象返回的属性的指示器号，它是访问该值所需的指示器号。

应用于

Property 对象

语法

object.**NumIndices**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

NumIndices 的值可以从 0 到 4。对于象在 ForeColor 属性中的正规属性，NumIndices 返回 0。对于常规的索引属性，如 ListBox 控件的 List 属性，NumIndices 返回 1。属性数组返回 2。

请参阅

IndexedValue 属性

NumIndices 属性（VBA 外接程序对象模型）

返回由 Property 对象所返回的属性的索引数目。

应用于

Property 对象

说明

NumIndices 属性的值可以是一个由 0 到 4 的整型。对大部分的属

性，NumIndices 返回 0。照惯例，索引属性可返回 1。属性数组可能返回 2。

请参阅

Item 方法(VBA 外接程序对象模型)，IndexedValue 属性(VBA 外接程序对象模型)，Object 属性，Value 属性

示例

下列示例使用 NumIndices 属性返回属于特定 VBComponent 对象的指定属性的索引值的个数。

```
Debug.PrintApplication.VBE.VBProjects(1).VBComponents(1).Properties(40).NumIndices
```

Object 属性

返回一个对控件的属性或方法的引用，它们具有 VisualBasic 对该控件自动扩展的属性或方法相同的名字。

应用于

Binding 对象，Extender 对象，ADOData 控件，DataRepeater 控件，DataList
Control，RemoteData 控件

语法

object.**Object**[.*property*|.*method*]

Object 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
property	与 VisualBasic 所提供的属性名一致的控件的属性
method	与 VisualBasic 所提供的方法名一致的控件的方法

说明

注意 Object 属性返回这的对象是没有被 VisualBasic 自动为其扩展属性或方法的控件的基础。因此，也可以通过 Object 属性来引用控件的“custom”属性和方法，例如 PrintSSTab1.Object.Tabs .

VisualBasic 为控件提供部分或全部 VisualBasic 工程中的属性和方法的标准集。对于控件或 ActiveX 部件（如 MicrosoftExcel 或 MicrosoftWord）来说，定义与这些标准属性或方法具有相同名字的属性或方法是可能的。当这种情况发生时，VisualBasic 自动地使用它提供的属性或方法来代替在控件中定义的具有相同名字的属性或方法。Object 属性允许跨越 VisualBasic 提供的属性或方法，并使用控件中定义的相同命名的属性或方法。

例如，Tag 属性是一个提供给 VisualBasic 工程中所有控件的属性。如果工程中一个控件具有名字 ctlDemo，那么可使用下面的语法来访问 Tag 属性：

```
ctlDemo.Tag
```

VisualBasic 自动使用它提供的 Tag 属性。然而，如果控件定义

了它自己的 Tag 属性并且想访问这个属性，那么可以按下面语法使用 Object 属性：

```
ctlDemo.Object.Tag
```

VisualBasic 自动地将下面这些属性、方法、和事件部分或全部扩展到 VisualBasic 工程中的控件：

属性

Align	Height	Object
Binding	HelpContextID	Parent
Bindings	Index	TabIndex
Cancel	Left	TabStop
Container	LeftNoRun	TagParent
DataChanged	LinkItem	ToolTipText
DataField	LinkMode	Top
DataSource	LinkTimeout	TopNoRun
Default	LinkTopic	VisibleTabStop
DragIcon	Name	WhatsThisHelpID
DragMode	NegotiateLinkItem	Width

方法

Drag	LinkSend	ShowWhatsThis
LinkExecute	Move	Zorder
LinkPoke	Refresh	
LinkRequest	SetFocus	

事件

GotFocus	LinkError	LinkOpen
LinkClose	LinkNotify	LostFocus

如果在使用了控件的属性或方法而没有获得期望的行为，那么检查一下该属性或方法是否与以上列表中的那些属性与方法具有相同的名字。如果名字相匹配，则检查与控件一起提供的文档，看

获得的行为与 VisualBasic 提供的属性或方法是否相匹配。如果行为不同，那么可能需要使用 Object 属性来访问想要的控件的特性。

请参阅

Object 属性(OLE 包容器控件)

Object 属性(VBAAdd-In 对象模式)

返回或设置一个由属性所返回的对象的值，此属性可读/写。

应用于

Property 对象

说明

若一个属性返回对象，必须用 Object 属性返回或设置此对象的值。

请参阅

IndexedValue 属性(VBA 外接程序对象模型)，NumIndices 属性(VBA 外接程序对象模型)，Value 属性

示例

下列示例加载图标名称，到指定的对象（必须是个窗体）的图标列表中。

```
Application.VBE.ActiveVBProject.VBComponents(1).Properties("Icon").Object=adPicture("Baseball.ico")
```

Object 属性（OLE 容器）

返回对象和/或 OLE 容器控件中对象的方法或属性的设置。

应用于

TreeView 控件，ImageList 控件，LIstView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，MSComm 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，PictureClip 控件，RichTextBox 控件，OLEContainer 控件

语法

object.**Object**[.*property*|.*method*]

Object 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
property	对象支持的属性
method	对象支持的方法

说明

使用这个属性，指定想在 Automation 任务中使用的对象。

可以用对象的属性和方法，来使用在自动化任务中 Object 属性返回的对象。有关对象支持的属性和方法的信息，请参阅创建对象的应用程序的文档。

ObjectAcceptFormats 属性

返回对象可接受的格式列表。

应用于

OLEContainer 控件

语法

object.**ObjectAcceptFormats**(*number*)

ObjectAcceptFormats 属性的语法有下面这些部分：

部分	描述
object	对象表达式，其值是“适用于”列表中的对象。
number	数值表达式，其值是指示了数组中元素的整数

说明

列表是一个基于 0 的字符串数组。当使用 Data 和 DataText 属性从对象获得数据时，可用数组的元素对 Format 属性进行设置。

ObjectAcceptFormatsCount 属性

返回对象可以接受的格式的个数。

应用于

OLEContainer 控件

语法

object.**ObjectAcceptFormatsCount**

object 所在处是一个对象表达式，其值是“应用于”列表中的一

个对象。

说明

使用这个属性得到在 ObjectAcceptFormats 属性数组中的元素的个数。

请参阅

Data 属性，DataText 属性，ObjectGetFormats 属性，ObjectGetFormatsCount 属性，Format 属性

示例

为了运行本例，应将 OLE 容器控件及三个 ListBox 控件放置在窗体中。将该示例程序粘贴到窗体的声明部分并按 F5 键。当出现“插入对象”对话框时，在“新建对象”列表框中选择一个应用程序，再选取“确定”建立一个对象。

```
PrivateSubForm_Click()
   DimI                    '变量声明。
   '显示“插入对象”对话框。
   Ole1.InsertObjDlg
   '更新可用动词列表。
   Ole1.FetchVerbs         '取动词。
   '清除该列表框。
   List1.Clear
   List2.Clear
   List3.Clear
```

```
    '填写动词列表框。因为 ObjectVerbs(0)是默认动词，
    '所以数组 ObjectVerbs()中的循环从 1 开始记数。
    For I=1To Ole1.ObjectVerbsCount-1
        List1.AddItem Ole1.ObjectVerbs(I)
    Next I
    '填写接受格式列表框。
    For I=0To Ole1.ObjectAcceptFormatsCount-1
        List2.AddItem Ole1.ObjectAcceptFormats(I)
    Next I
    '填写获取格式列表框。
    For I=0To Ole1.ObjectGetFormatsCount-1
        List3.AddItem Ole1.ObjectGetFormats(I)
    Next I
End Sub
```

ObjectEvent 事件

当一个被指定给 VBControlExtender **对象变量的控件引发一个事件时发生。**

语法

Private Sub *object*_**ObjectEvent**(*Info* **As EventInfo**)

ObjectEvent 事件语法有这些部分:

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
info	返回一个对 EventInfo 对象的引用

说明

ObjectEvent 事件是一个通用事件，允许您通过该控件的事件参数处理一个控件的事件和返回值。

可以利用 ObjectEvent 事件来捕获一个控件产生的通用事件，保证您部署的任何控件包含部署应用程序所要求的某些基本功能。

ObjectGetFormats 属性

返回对象可以提供的格式列表。

应用于

OLEContainer 控件

语法

object.**ObjectGetFormats**(*number*)

ObjectGetFormats 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“适用于”列表中的一个对象
number	一个指示数组中元素的数值表达式

说明

这个列表是一个基于 0 的字符串数组。当使用 Data 和 DataText 属

性从对象中获取数据时，该数组中的每个元素均可用于设置 Format 属性。

请参阅

Data 属性，DataText 属性，ObjectGetFormatsCount 属性，Format 属性

示例

为了运行本例，应将 OLE 容器控件及三个 ListBox 控件放置在窗体中。将该示例程序粘贴到窗体的声明部分并按 F5 键。当出现“插入对象”对话框时，在“新建对象”列表框中选择一个应用程序，再选取“确定”建立一个对象。

```
PrivateSubForm_Click()
   DimI'变量声明。
   '显示“插入对象”对话框。
   Ole1.InsertObjDlg
   '更新可用动词列表。
   Ole1.FetchVerbs  '取动词。
   '清除该列表框。
   List1.Clear
   List2.Clear
   List3.Clear
   '填写动词列表框。因为 ObjectVerbs(0)是默认动词，
   '所以数组 ObjectVerbs()中的循环从 1 开始记数。
   ForI=1ToOle1.ObjectVerbsCount-1
```

```
        List1.AddItemOle1.ObjectVerbs(I)
    NextI
    '填写接受格式列表框。
    ForI=0ToOle1.ObjectAcceptFormatsCount-1
        List2.AddItemOle1.ObjectAcceptFormats(I)
    NextI
    '填写获取格式列表框。
    ForI=0ToOle1.ObjectGetFormatsCount-1
        List3.AddItemOle1.ObjectGetFormats(I)
    NextI
EndSub
```

ObjectGetFormatsCount 属性

返回对象可以提供格式的个数。

应用于

OLEContainer 控件

语法

object.**ObjectGetFormatsCount**

object 所在处是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

使用这个属性确定 ObjectGetFormats 属性数组中元素的个数。

请参阅

Data 属性，DataText 属性，ObjectAcceptFormatsCount 属性，Format 属性

示例

为了运行本例，应将 OLE 容器控件及三个 ListBox 控件放置在窗体中。将该示例程序粘贴到窗体的声明部分并按 F5 键。当出现“插入对象”对话框时，在“新建对象”列表框中选择一个应用程序，再选取“确定”建立一个对象。

```
PrivateSubForm_Click()
   DimI'变量声明。
   '显示“插入对象”对话框。
   Ole1.InsertObjDlg
   '更新可用动词列表。
   Ole1.FetchVerbs  '取动词。
   '清除该列表框。
   List1.Clear
   List2.Clear
   List3.Clear
   '填写动词列表框。因为 ObjectVerbs(0)是默认动词，
   '所以数组 ObjectVerbs()中的循环从 1 开始记数。
   ForI=1ToOle1.ObjectVerbsCount-1
       List1.AddItemOle1.ObjectVerbs(I)
   NextI
```

```
    '填写接受格式列表框。
    For I=0 To Ole1.ObjectAcceptFormatsCount-1
        List2.AddItem Ole1.ObjectAcceptFormats(I)
    Next I
    '填写获取格式列表框。
    For I=0 To Ole1.ObjectGetFormatsCount-1
        List3.AddItem Ole1.ObjectGetFormats(I)
    Next I
End Sub
```

ObjectMove 事件

OLE 容器控件中的活动对象被移动或调整其大小后立即发生。

应用于

OLE 包容器控件

语法

Private Sub *object*_**ObjectMove**(*left* **As Single**, *top* **As Single**, *width* **As Single**, *height* **As Single**)

ObjectMove 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
left	OLE 容器控件被移动或调整大小后，其左边框的瞬时坐标值
top	OLE 容器控件被移动或调整大小后，其上边框的瞬时坐标值
width	OLE 容器控件被移动或调整大小后的瞬时宽度
height	OLE 容器控件被移动或调整大小后的瞬时高度

说明

用户移动 OLE 容器控件或调整其大小时，应用程序能用 ObjectMove 事件确定是否真正改变控件的尺寸和位置。如果 ObjectMove 事件过程未改变 OLE 容器控件的位置和大小，则其中的对象返回它的初始位置，并被通知以新的尺寸大小。此事件中作为传递参数的坐标包括 OLE 容器控件的边界。

当 OLE 容器控件接收到其所包含对象的尺寸信息时，触发 ObjectMove 和 Resize 事件。然而 Resize 事件不接收关于控件位置的任何信息。如果 OLE 容器控件被移离窗体，则参数具有正号或负号，此值代表对象相对于窗体上边和左边的位置。

请参阅

Resize 事件，Move 方法，SizeMode 属性，DragMode 属性

ObjectVerbFlags 属性

在给定的 ObjectVerbs 数组中，为每个谓词返回菜单的状态（比如是可用还是禁用，是否复选，等等）。

应用于

OLEObject 对象，OLEContainer 控件

语法

object.**ObjectVerbFlags**(*number*)

ObjectVerbFlags 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个指示数组中元素的数值表达式

返回值

ObjectVerbFlags 属性返回下列值：

常数	值	描述
vbOLEFlagChecked	&H0008	菜单项是复选的
vbOLEFlagDisabled	&H0002	菜单项是禁用的（但不是暗淡的）
vbOLEFlagEnabled	&H0000	菜单项是可用的
vbOLEFlagGrayed	&H0001	菜单项是暗淡的
vbOLEFlagSeparator	&H0800	菜单项是分割符条

注意这些常数也列在对象浏览器的 VisualBasic 对象和过程库中。

说明

在 ObjectVerbs 数组中的第一个谓词是缺省的谓词。ObjectVerbFlags 数组包含 ObjectVerbs 数组中的每个谓词的菜单状态的信息（比如，暗淡的、复选的，等等）。

当显示含有对象谓词的菜单时，要检查它的属性值是设置为如何显示的。

请参阅

ObjectVerbs 属性，ObjectVerbsCount 属性，Verb 属性，AutoVerbMenu 属性

ObjectVerbs 属性

返回对象所支持的谓词列表。

应用于

RichTextBox 控件，OLEContainer 控件

语法

object.**ObjectVerbs**(*number*)

ObjectVerbs 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“适用于”列表中的一个对象
number	一个指示数组中元素的数值表达式

说明

ObjectVerbs 是基于 0 的字符串数组。这个属性与 ObjectVerbsCount

属性一起使用，可得到对象所支持的谓词。当对象使用 DoVerb 方法激活时，这些谓词用于确定要执行的动作。数组中谓词的列表，取决于当前条件，并随对象的不同而变化。

每个对象都有它自己支持的谓词集合。下面各值表示的是所有对象都能支持的标准谓词：

常数	值	描述
vbOLEPrimary	0	对象的缺省动作
vbOLEShow	-1	激活对象进行编辑。如果创建对象的应用程序支持现场激活，该对象在 OLE 容器控件内激活
vbOLEOpen	-2	在分隔的应用程序窗口打开对象。如果创建对象的应用程序支持现场激活，该对象在其自己的窗口中激活
vbOLEHide	-3	对于嵌入的对象，隐藏创建对象的应用程序
vbOLEUInPlaceUIActivate	-4	如果对象支持现场激活，则将该对象激活为现场激活，并显示所有的用户接口工具。如果对象不支持现场激活，则不激活对象，并产生一个错误
vbOLEInPlaceActivate	-5	如果将焦点移到 OLE 容器控件，为

		对象创建一个窗口，并为对象作好编辑的准备。如果对象不支持单击鼠标的激活，则产生一个错误
vbOLEDiscardUndoState	-6	当激活对象进行编辑时，用于放弃所有改变的记录，这些改变可由对象的应用程序撤消

注意这些谓词也许未列在 ObjectVerbs 属性数组中。

在 ObjectVerbs 数组中的第一个谓词 ObjectVerbs(0)，就是缺省的谓词。除非另外指定，这个谓词将激活对象。

数组中其余的谓词可以在菜单中显示。如果在菜单中适合于显示缺省谓词，则缺省谓词在 ObjectVerbs 数组中有两个项。

显示对象的应用程序一般在“编辑”菜单上包含有“对象”命令。当选取“编辑对象”时，菜单显示该对象的各个谓词。在运行时，使用 ObjectVerbs、ObjectVerbsCount 和 ObjectVerbFlags 属性创建这样的菜单。

对象支持的谓词列表会随着对象的状态而变化。为了更新对象所支持的谓词列表，可使用 FetchVerbs 方法。在显示谓词列表之前一定要先作更新。

当用鼠标右键单击对象时，为了在弹出式菜单中自动显示 ObjectVerbs 数组中的谓词，应将 AutoV.

erbMenu 属性设为 True。

请参阅

ObjectVerbFlags 属性，ObjectVerbsCount 属性，Verb 属性，

AutoVerbMenu 属性，DoVerb 方法，FetchVerbs 方法，FetchVerbs 方法(ActiveX 控件)

示例

为了运行本例，应将 OLE 容器控件及三个 ListBox 控件放置在窗体中。将该示例程序粘贴到窗体的声明部分并按 F5 键。当出现“插入对象”对话框时，在“新建对象”列表框中选择一个应用程序，再选取“确定”建立一个对象。

```
PrivateSubForm_Click()
   DimI      '变量声明。
   '显示“插入对象”对话框。
   Ole1.InsertObjDlg
   '更新可用动词列表。
   Ole1.FetchVerbs  '取动词。
   '清除该列表框。
   List1.Clear
   List2.Clear
   List3.Clear
   '填写动词列表框。因为 ObjectVerbs(0)是默认动词，
   '所以数组 ObjectVerbs()中的循环从 1 开始记数。
   ForI=1ToOle1.ObjectVerbsCount-1
       List1.AddItemOle1.ObjectVerbs(I)
   NextI
```

```
    '填写接受格式列表框。
    For I=0 To Ole1.ObjectAcceptFormatsCount-1
        List2.AddItem Ole1.ObjectAcceptFormats(I)
    Next I
    '填写获取格式列表框。
    For I=0 To Ole1.ObjectGetFormatsCount-1
        List3.AddItem Ole1.ObjectGetFormats(I)
    Next I
End Sub
```

ObjectVerbsCount 属性

返回对象支持的谓词的个数。

应用于

RichTextBox 控件，OLEContainer 控件

语法

object.**ObjectVerbsCount**

object 是一个对象表达式，其值是一个 OLE 容器控件。

说明

使用这个属性确定 ObjectVerbs 属性数组中元素的个数。

对象支持的谓词列表会随着对象的状态而变化。为了更新对象所支持的谓词列表，可使用 FetchVerbs 方法。

请参阅

ObjectVerbFlags 属性,ObjectVerbs 属性,Verb 属性,FetchVerbs 方法，FetchVerbs 方法(ActiveX 控件)

示例

为了运行本例，应将 OLE 容器控件及三个 ListBox 控件放置在窗体中。将该示例程序粘贴到窗体的声明部分并按 F5 键。当出现“插入对象”对话框时，在“新建对象”列表框中选择一个应用程序，再选取“确定”建立一个对象。

```
PrivateSubForm_Click()
   DimI      '变量声明。
   '显示“插入对象”对话框。
   Ole1.InsertObjDlg
   '更新可用动词列表。
   Ole1.FetchVerbs  '取动词。
   '清除该列表框。
   List1.Clear
   List2.Clear
   List3.Clear
   '填写动词列表框。因为 ObjectVerbs(0)是默认动词，
   '所以数组 ObjectVerbs()中的循环从 1 开始记数。
   ForI=1ToOle1.ObjectVerbsCount-1
       List1.AddItemOle1.ObjectVerbs(I)
   NextI
```

```
    '填写接受格式列表框。
    For I=0 To Ole1.ObjectAcceptFormatsCount-1
        List2.AddItem Ole1.ObjectAcceptFormats(I)
    Next I
    '填写获取格式列表框。
    For I=0 To Ole1.ObjectGetFormatsCount-1
        List3.AddItem Ole1.ObjectGetFormats(I)
    Next I
End Sub
```

Oct 函数

返回 Variant(String)，**代表一数值的八进制值。**

语法

Oct(*number*)

必需的 number **参数为任何有效的数值表达式或字符串表达式。**

说明

如果 *number* **尚非整数，那么在执行前会先四舍五入成最接近的整数。**

如果 number **为**	Oct **返回**
Null	Null
Empty	**零**(0)
任何其他的数字	**最多可到** 11 **个八进制字符**

可以将适当范围的数前缀以&O **来直接表示八进制数字。例如，**

八进制表示法的&O10 代表十进制的 8。

请参阅

TypeConversion 函数，Hex 函数

示例

本示例使用 Oct 函数将某数值转换为 8 进制表达式。

```
DimMyOct
MyOct=Oct(4)    '返回 4。
MyOct=Oct(8)    '返回 10。
MyOct=Oct(459) '返回 713。
```

OLE 容器控件

OLE 容器控件允许将可插入的对象添加到 VisualBasic 应用程序的窗体中。使用 OLE 容器控件，可以实现：

在应用程序中为可插入对象建立占位符。在运行时，可以创建要在 OLE 容器控件内显示的对象，或者改变在设计时放置于 OLE 容器控件内的对象。

在应用程序中创建一个链接对象。

使用 Data 控件将 OLE 容器控件绑定到一个数据库。

可以在设计时使用插入对象对话框（它包含插入对象、特殊粘贴等命令）来创建对象，也可以在运行时通过设置相应的属性来创建对象。

当使用 ObjectMove 方法在窗体上移动一个 OLE 容器控件时，该对

象的 Height 和 Width 属性值在移动之后会稍有不同。这是因为 ObjectMove 方法的参数是一些像素值，它们是按当前窗体的比例模式转换。从像素到缇的转换和反转换得到的值并不总是相同。

使用 OLE 容器控件的弹出式菜单

在窗体上每绘制一个 OLE 容器控件，插入对象对话框就显示一次。使用这个对话框创建链接的或嵌入的对象。如果选取“取消”，对象就不被创建。

在设计时，用鼠标右键单击 OLE 容器控件会显示弹出式菜单。哪些命令可以显示在该弹出式菜单上，取决于 OLE 容器控件的状态，如下表所示：

命令	遇到下列情况，允许在弹出式菜单中显示
InsertObject	允许任何情况
PasteSpecial	Clipboard 对象含有一个有效的对象
DeleteEmbeddedObject	OLE 容器控件含有一个嵌入的对象
DeleteLinkedObject	OLE 容器控件含有一个链接的对象
CreateLink	设置了 SourceDoc 属性
CreateEmbeddedObject	设置了 Class 或 SourceDoc 属性

OLE 容器控件一次只能包含一个对象。可以使用以下几种方法，创建一个链接的或内嵌对象：

使用“插入对象”或“特殊粘贴”对话框（在运行时或设计时）。

在属性窗口设置 Class 属性，用鼠标右键单击 OLE 容器控件，然

后再选择相应的命令（只能在设计时）。
使用 OLE 容器控件的相应的方法。

查找类名

在属性窗口选择 Class 属性，并且单击“属性”按钮，可以得到应用程序可用的类名的列表。
注意插入对象对话框并不显示类名的列表。这个对话框显示每个对象类的名称，该名称一般比较长，也比较容易理解。

属性

ObjectAcceptFormats 属性，Class 属性，对象 VerbFlags 属性，SizeMode 属性，Data 属性，DataText 属性，FileNumber 属性，HostName 属性，IpOleObject 属性，ObjectAcceptFormatsCount 属性，DisplayType 属性，ObjectGetFormats 属性，ObjectGetFormatsCount 属性，OLETyp 属性，ObjectVerbs 属性，ObjectVerbsCount 属性，SourceDoc 属性，SourceItem 属性，AppIsRunning 属性，UpdataOptions 属性，Verb 属性，AutoActivate 属性，AutoVerbMenu 属性，Object 属性(OLEContainer)，OLETypeAllowed 属性，Action 属性(OLEContainer)，MiscFlags 属性，Format 属性,PasteOK 属性,DragMode 属性,hWnd 属性,MouseIcon 属性，MousePointer 属性，TabStop 属性，Appearance 属性，Enabled 属性,HelpContextID 属性,Index 属性(ControlArray)，Name 属性，Parent 属性，Container 属性，DataChanged 属性，

DataField 属性，WhatsThisHelpID 属性

方法

Refresh 方法，SetFocus 方法，Drag 方法，Move 方法，Zorder 方法，PasteSpecialDlg 方法，Copy 方法，CreateEmbed 方法，CreateLink 方法，Delete 方法(OLEContainer)，InsertObjDlg 方法，Paste 方法，ReadFromFile 方法，SaveToFile 方法，DoVerb 方法，Close 方法(OLEContainer)，UpdataFetchVerbs 方法(ActiveX 控件)

事件

Click 事件，DragDrop 事件，DragOver 事件，GotFocus 事件，KeyDown，KeyUp 事件，KeyPress 事件，LostFocus 事件，MouseDown，MouseUp 事件，MouseMove 事件，Resiza 事件，DblClick 事件，Updated 事件，对象 Move 事件

请参阅

Object 属性(OLE 包容器控件)

OLECompleteDrag 事件

当源部件被放到目标部件时发生，并通知源部件拖放操作被执行或取消。

应用于

CoolBar 控件，DBCombo 控件，DBList 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，Chec

kBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileLIstBox 控件，Form 对象，Forms 集合，Frame 控件，Image 控件，Label 控件，LIstBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

PrivateSub*object*_**OLECompleteDrag**([*effect***AsLong**])

OLECompleteDrag 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
effect	源对象设置的长整型数，用来识别执行的动作，这样当部件被移动后允许源采取适当的动作（例如，如果源被从一个部件移到另一个部件，则执行删除数据操作）。可能的取值列于“设置值”中

设置值

effect 设置如下：

常数	值	描述
vbDropEffectNone	0	放目标不接受数据，或者放操作被取消
vbDropEffectCopy	1	放结果保存于从源到目标的数据拷贝中。初始数据没有被拖放操作改变
vbDropEffectMove	2	放结果保存于初始数据的连接中，该数据是在拖放源和放目标之间产生的

说明

OLECompleteDrag 事件是 OLE 拖放操作最后调用的事件。当对象被放到目标部件时，此事件通知源部件所执行的动作。目标通过 OLEDragDrop 事件的 effect 参数设置此值。基于此，源可决定需采取的适当动作。例如，若对象被移到目标(vbDropEffectMove)，移动后源需要在自身删除该对象。

如果 OLEDragMode 设置为 Automatic，则 VisualBasic 执行缺省的动作。此事件仍然发生，允许添加或改变动作。

大部分部件支持人工 OLE 拖放事件，也有一些支持自动化 OLE 拖放事件。

OLEDrag 方法

引起部件初始化 OLE 拖放操作。

应用于

DataList 控件，DBCombo 控件，DBList 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DrivListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Image 控件，Label 控件，LIstBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

object.**OLEDrag**

object 所在处代表对象表达式，其值是“应用于”列表中的一个

对象。

说明

当调用 OLEDrag 方法时，部件的 OLEStartDrag 事件发生，允许向目标部件提供数据。

OLEDragDrop 事件

当源部件决定放操作能发生，且源部件被放到目标部件时，此事件发生。

注意仅当 OLEDropMode 被设置为 1 (Manual)时，此事件才发生。

应用于

DBCombo 控件，DBList 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

Private Sub *object*_**OLEDragDrop**(*data* **As DataObject**, *effect* **As Long**, *button* **As Integer**, *shift* **As Integer**, *x* **As Single**, *y* **As Single**)

OLEDragDrop 事件语法　描述
包含下面部分：部分

Object	**对象表达式，其值是“适用于”列表中的一个对象**
Data	DataObject **对象，包含源提供的格式，另外也可能包含这些格式的数据。若** DataObject **不包含数据，则当控件调用** GetData **方法时提供数据。** SetData **和** Clear **方法不能用在这里**
effect	**源对象设置的长整型数，用来识别执行的动作，这样当部件被移动后允许源采取适当的动作（例如，如果源被从一个部件移到另一个部件，则执行删除数据操作）。可能的取值列于“设置值”中**
button	**整数，当按下鼠标键时，与鼠标状态相对应。左键为位** 0**，右键为位** 1**，中键为位** 2**。这些位相应的值分别为** 1**，**2 **和** 4**，它代表了鼠标键的状态。可设置三个位中的部分、全部或根本不设置，相应地表明部分、全部按键被按下或没有按键按下**
shift	**整数，当按下** SHIFT**、**CTRL **和** ALT **键时，**

	与这些键状态相对应。SHIFT 键为位 0，CTRL 键为位 1，ALT 键为位 2。这些位相应的值分别为 1，2 和 4，*shift* 参数代表了这些键的状态。可设置三个位中的部分、全部或根本不设置，相应地表明部分、全部按键被按下或没有按键按下。例如，同时按下 CTRL 和 ALT 键，*shift* 值为 6
x,*y*	确定鼠标指针当前位置的数值。x 和 y 值由对象的 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 属性设置的坐标系统的格式来表示
VbDropEffectScroll	-2147483648 一个掩码，指出放下目标窗口已经滚动或应该滚动

设置值

effect 设置如下：

常数	值	描述
vbDropEffectNone	0	Drop 目标不接受数据
vbDropEffectCopy	1	Drop 结果保存于从源到目标的数据拷贝中。初始数据没有被拖放操作改变
vbDropEffectMove	2	Drop 结果保存于要从拖放源移到放源的数据中移动后，拖放源要删除数据

说明

源 ActiveX 部件应总是屏蔽 *effect* 参数值，以确保同将来实现的 ActiveX 部件兼容。目前，仅使用了 *effect* 参数 32 位中的 3 位，然而将来的 VisualBasic 版本就可能用到其它位。所以考虑到将来的问题，拖源和放目标在进行任何比较之前应屏蔽这些值。

例如，源部件不能把 *effect* 同 vbDropEffectCopy 相比，如：

```
IfEffect=vbDropEffectCopy...
```

而是应该屏蔽该值或被搜寻的值，如：

```
IfEffectAndvbDropEffectCopy=vbDropEffectCopy...
```

-或-

```
If(EffectAndvbDropEffectCopy)...
```

这样，允许在 VisualBasic 新版本中定义新的放效果，并与现存的代码保持向后兼容性。

大部分部件支持人工 OLE 拖放事件，也有一些支持自动化 OLE 拖放事件。

请参阅

OLECompleteDrag 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件，OLECompleteDrag 事件(ActiveX 控件)，OLEDragOver 事件(ActiveX 控件)，OLEGiveFeedback 事件(ActiveX 控件)，OLESetData 事件(ActiveX 控件)，OLEStartDrag 事件(ActiveX 控件)

OLEDragMode 属性

返回或设置是由部件还是由程序员来处理 OLE 拖放操作。

应用于

DBCombo 控件，DBList 控件，ComboBox 控件，DirListBox 控件，FileListBox 控件，Image 控件，LIstBox 控件，PictureBox 控件，TextBox 控件

语法

object.**OLEDragMode**=*mode*

OLEDragMode 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
mode	整数，规定部件处理 OLE 拖放操作的方法，如“设置值”中所述

设置值

mode 设置如下：

常数	值	描述
vbOLEDragManual	0	（缺省）手动。程序员处理所有的 OLE 拖放操作
vbOLEDragAutomatic	1	自动。部件处理所有的 OLE 拖放操作

说明

当设置 OLEDragMode 为 Manual 时，必须调用 OLEDrag 方法启动拖

放，进而触发 OLEStartDrag 事件。

当设置 OLEDragMode 为 Automatic 时，若想拖出控件，源部件用其包含的数据填充 DataObject 对象，并在初始化 OLEStartDrag 事件（也包括 OLESetData 和其它源级的 OLE 拖放事件）之前设置 *effects* 参数。这样可以控制拖放操作，允许通过添加其它格式，或者用 Clear 或 SetData 方法忽略或禁用自动数据来进行调整。

若源的 OLEDragMode 属性被设为 Automatic，并且也没有数据被加载到 OLEStartDrag 事件中，或 aftereffects 设置为 0，则 OLE 拖放操作不发生。

注意如果控件的 DragMode 属性被设为 Automatic，则 OLEDragMode 的设置被忽略，因为常规的 VisualBasic 拖放事件优先发生。

OLEDragOver 事件

当一个部件在另一个部件上拖动时发生。

应用于

DBCombo 控件，DBList 控件，Data 控件，ProprtyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

Private Sub *object*_**OLEDragOver**(*data* **As DataObject**, *effect* **As Long**, *button* **As Integer**, *shift* **As Integer**, *x* **As Single**, *y* **As Single**, *state* **As Integer**)

OLEDragOver 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“适用于”列表中的一个对象
ddta	DataObject 对象，包含源提供的格式，另外也可能包含这些格式的数据。若 DataObject 不包含数据，则当控件调用 GetData 方法时提供数据。SetData 和 Clear 方法不能用在这里
effect	源对象最初设置的长整型数，用来识别它支持的效果。在事件过程中，此参数必须被目标部件正确地设置。*effect* 值由所有活动的效果（如设置值表）逻辑 Or 确定。目标部件应检查这些效果以及其它参数以确定哪个动作适合于它，然后把此参数设置为允许的效果之一（如源所规定），以便确定放置选项到部件该执行哪个动作。可能的取值列于“设置值”中
button	整数，当按下鼠标键时，与鼠标状态相对应。左键为位 0，右键为位 1，中键为位 2。这些位相应的值分别为 1，2 和 4，它代表了鼠标键的状态。可设置三个位中的部分、全部或根本不设置，相应地表明部分、全部按键被按下或没有按键按下
shift	整数，当按下 SHIFT、CTRL 和 ALT 键时，与这些键状

	态相对应。SHIFT 键为位 0，CTRL 键为位 1，ALT 键为位 2。这些位相应的值分别为 1，2 和 4，*shift* 参数代表了这些键的状态。可设置三个位中的部分、全部或根本不设置，相应地表明部分、全部按键被按下或没有按键按下。例如，同时按下 CTRL 和 ALT 键，shift 值为 6
x,y	在目标窗体或控件中，规定当前鼠标指针水平(x)和垂直(y)位置的数。x 和 y 值由对象的 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 属性设置的坐标系统格式来表示
state	整数，相应于控件的转换状态，此控件将被拖放到与其相关的目标窗体或控件中。可能的取值列于“设置值”中

设置值

effect 设置如下：

常数	值	描述
VbDropEffectNone	0	放目标不接受数据
VbDropEffectCopy	1	放结果保存于从源到目标的数据拷贝中。初始数据没有被拖放操作改变
VbDropEffectMove	2	放结果保存于要从拖放源移到放源的数据中。移动后，拖放源要删除数据
vbDropEffectScroll	-2147483648 (&H80000000)	在目标部件中，滚动正在或将要发生。此值与其它值共同使用。注意仅当在部件中执行自己的滚动时才能应用

state 设置如下：

常数	值	描述
vbEnter	0	在目标范围内源部件正被拖动
vbLeave	1	在目标范围之外源部件正被拖动
vbOver	2	在目标内源部件已经从一个位置移到另一个位置

说明

注意如果 state 参数是 vbLeave，表明鼠标指针已离开目标，则 x 和 y 参数保持为零值。

源 ActiveX 部件应总是屏蔽 effect 参数值，以确保同将来实现的 ActiveX 部件兼容。现在仅使用了 effect 参数 32 位中的 3 位，然而将来的 VisualBasic 版本就可能用到其它位。所以考虑到将来的问题，拖源和放目标在进行任何比较之前应屏蔽这些值。

例如，源部件不能把 effect 同 vbDropEffectCopy 相比，如：

```
IfEffect=vbDropEffectCopy...
```

而应屏蔽该值或被搜寻的值，如：

```
IfEffectAndvbDropEffectCopy=vbDropEffectCopy...
```

-或-

```
If(EffectAndvbDropEffectCopy)...
```

允许在 VisualBasic 新版本中定义新的放效果，并与现存的代码保持向后兼容性。

大部分部件支持人工 OLE 拖放事件，也有一些支持自动化 OLE 拖放事件。

OLEDropAllowed 属性

返回或设置一个值，决定 OLE 容器控件是否是 OLE 拖放操作的放目标。

应用于

UserControl 对象，OLE 包容器控件

语法

object.**OLEDropAllowed**[=*boolean*]

OLEDropAllowed 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，确定 OLE 容器控件是否为放目标，如“设置值”中所述

设置值

boolean **设置如下：**

设置	描述
True	拖动链接对象或内嵌对象时，当鼠标指针在 OLE 容器控件上移动时会出现放图标。放置对象到 OLE 容器控件同用 Paste 方法从系统剪贴板粘贴对象具有同样的效果
False	（缺省）拖动链接或嵌入对象时，在 OLE 容器控件上不出现放图标。将对象放到 OLE 容器控件上对控件不会有影响

说明

OLEDropAllowed 属性设为 True 时，由 MousePointer 属性确定鼠标指针的形状。如果 MousePointer 设置值为 0（缺省），对正在发生的动作，VisualBasic 显示标准的拖放图标。

为了移动或拷贝链接或嵌入对象，OLETypeAllowed 属性必须设为 1(vbOLEEmbedded)或 2(vbOLEEither)。为了链接对象，则要设为 0(vbOLELinked)或 2。对于 Class、SourceDoc 以及 SourceItem 属性的设置，OLEDropAllowed 属性设为 True 时放一个对象，同使用 OLE

容器控件的 Paste 方法具有一样的效果。

如果 OLEDropAllowed 属性设为 True，则拖动对象时，OLE 容器控件不接收 DragDrop 或 DragOver 事件。同样，当 OLEDropAllowed 属性设为 True 时，DragMode 属性的设置对 OLE 容器控件的拖放动作没有影响。

请参阅

Class 属性，SourceDoc 属性，SourceItem 属性，OLETypeAllowed 属性，Paste 方法，DragMode 属性

OLEDropMode 属性

返回或设置目标部件如何处理放操作。

应用于

DBCombo 控件，DBList 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

object.**OLEDropMode**[=*mode*]

OLEDropMode 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
mode	枚举整数，规定部件处理 OLE 拖放操作的方法，如“设置值”中所述

设置值

mode 设置如下：

常数	值	描述
vbOLEDropNone	0	（缺省）无。目标部件不接受 OLE 放操作，并且显示 NoDrop 光标
vbOLEDropManual	1	人工。目标部件触发 OLE 放事件，允许程序员用代码处理 OLE 放操作
VbOLEDropAutomatic	2	自动。如果 DataObject 对象包含目标部件能识别的格式的数据，则自动接受 OLE 放操作。当 OLEDropMode 设为 vbOLEDropAutomatic 时，在目标上鼠标事件和 OLE 拖放事件都不会发生

说明

注意目标部件检查将在其上拖动的内容，以便确定触发何种事件，是 OLE 拖放事件还是 VisualBasic 拖放事件。在同一时间内仅能拖动一种对象，所以没有部件的冲突，没有触发何种事件的疑惑。

OLEGiveFeedback 事件

在每个 OLEDragOver 事件后发生。OLEGiveFeedback 允许源部件提供可视的反馈，例如通过改变鼠标的图标来表明放目标后将发生什么，或者在选项上提供可视的反馈（在源部件中）以指出将发生什么。

应用于

DBCombo 控件，DBList 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

Private Sub *object*_**OLEGiveFeedback**(*effect* **As Long**, ***defaultcursors*** **As Boolean**)

OLEGiveFeedback 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
effectt	在 OLEDragOver 事件中目标部件设置的长整型数，若放选定项，则由此数确定要执行的动作。允许源采取合适的动作（例如给出可视的反馈）。可能的取值列于“设置值”中
defaultcursors	布尔值，决定 VisualBasic 使用部件缺省鼠标光标，还是自定义鼠标光标 True（缺省）=使用缺省鼠标光标 False= 不用缺省光标。必须用 Screen 对象的 MousePointer 属性设置鼠标光标

设置值

effect 设置如下：

常数	值	描述
vbDropEffectNone	0	放目标不接受数据
vbDropEffectCopy	1	放结果保存于从源到目标的数据拷贝中。初始数据没有被拖放操作改变。
vbDropEffectMove	2	放结果保存于要从拖放源移到放源的数据中。移动后，拖放源要删除数据
vbDropEffectScroll	-2147483648 (&H80000000)	在目标部件中，滚动正在或将要发生此值与其它值共同使用。注意仅当在部件中执行自己的滚动时才能应用

说明

如果 OLEGiveFeedback 事件中没有代码，或者如果 *defaultcursors* 参数设为 True，则 VisualBasic 自动地把鼠标图标设为部件提供的缺省图标。

源 ActiveX 部件应总是屏蔽 *effect* 参数值，以确保同将来实现的 ActiveX 部件兼容。现在仅使用了 *effect* 参数 32 位中的 3 位，然而将来的 VisualBasic 版本就可能用到其它位。所以考虑到将来的问题，拖源和放目标在进行任何比较之前应屏蔽这些值。

例如，源部件不能把 effect 同 vbDropEffectCopy 相比，如：

IfEffect=vbDropEffectCopy...

而应屏蔽该值或被搜寻的值，如：

IfEffectAndvbDropEffectCopy=

vbDropEffectCopy...

-或-

If(EffectAndvbDropEffectCopy)...

允许在 VisualBasic 新版本中定义新的放效果，并与现存的代码保持向后兼容性。

大部分部件支持人工 OLE 拖放事件，也有一些支持自动化 OLE 拖放事件。

OLERequestPendingMsgText 属性

当自动请求处于挂起状态且接收到鼠标或键盘的输入时，返回或设置的显示替补的“忙”消息的文本。在设计时无效。

语法

object.**OLERequestPendingMsgText**[=*string*]

OLERequestPendingMsgText 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	字符串表达式，其值是在 ActiveX 请求被挂起的条件下，替补的消息框中显示的消息文本

说明

当自动请求挂起且接收到鼠标或键盘的输入时，VisualBasic 显示缺省的“部件请求挂起”对话框。此对话框包含文本和“切换到”按钮,此按钮是为了使用如MicrosoftExcel 等可视的ActiveX 部件而设的。有时，缺省对话框不能满足用户的要求，如：

程序调用了由无用户界面的 ActiveX 部件所提供的对象的方法。例如，由 VisualBasic 专业版生成的ActiveX 部件，可以不带任何可视窗体而在后台运行。

所调用的 ActivX 部件可能是应用 VisualBasic 企业版的远程自动化特征生成的，并且可能是在距用户一定距离的另一台计算机上运行。

如果程序应用 GetObject 函数加载了 MicrosoftExcel 工作手册，则当切换到 MicrosoftExcel 时，此工作手册是不可见的。实际上，MicrosoftExcel 本身可能是不可见的，这种情况下“切换到”按钮不做任何事情。

在这些情况下，缺省文本和“切换到”按钮是不合适的，它可能使程序的使用者感到困惑。

OLERequestPendingMsgText 属性允许以替补消息框代替缺省“部件请求挂起”对话框。将 OLERequestPendingMsgText 设置成自定义的消息字符串，就能产生一个包含此消息文本和“确定”按钮的简单消息框，它来代替缺省的“部件请求挂起”对话框。

注意一旦自动请求被ActiveX 部件接受便无法取消。

如果 OLERequestPendingMsgText 等于空字符串(“”)，则显示缺省

的“部件请求挂起”对话框。

重点当知道自动请求将需要数秒钟以上，并且应用的是隐含或远程 ActiveX 部件时，就应该设置替补消息。因为网络传输可能会使即便是很短的 ActiveX 请求也需耗费数秒钟，所以，对于远程 ActiveX 部件来说，对所有的请求都推荐使用替补消息。

OLERequestPendingMsgTitle 属性

当自动请求处于挂起状态，接收到鼠标或键盘的输入时，返回或设置显示替补的“忙”消息的标题。在设计时无效。

应用于

App 对象

语法

object.**OLERequestPendingMsgTitle**[=*string*]

OLERequestPendingMsgTitle 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	字符串表达式，其值是 ActiveX 请求被挂起的条件下，替补的消息框的标题

说明

如果已经设置了 OLERequestPendingMsgText 属性，则以 OLERequestPendingMsgTitle 属性的值作为替补的“忙”消息框的标

题，它替代了缺省的“部件请求挂起”对话框。此属性的缺省值是 App 对象 Title 属性的当前值。这是推荐的设置。

如果 OLERequestPendingMsgText 属性设置为空字符串("")，则 OLERequestPendingMsgTitle 属性被忽略。

请参阅

OLERequestPendingMsgText 属性，OLERequestPendingTimeout 属性 OLEServerBusyMsgText 属性，OLEServerBusyMsgTitle 属性，OLEServerBusyRaiseError 属性，OlEServerBusyTimeout 属性

OLERequestPendingTimeout 属性

返回或设置在自动请求挂起时、接受到鼠标或键盘输入而引发“部件请求挂起”对话框（或其它替补消息）之前，所需延迟的毫秒数。在设计时无效。

应用于

App 对象

语法

object.**OLERequestPendingTimeout***[**=*milliseconds***]**

OLERequestPendingTimeout 属性语法包含下面部分：

部分	描述
Object	对象表达式，其值是“应用于”列表中的一个对象
milliseconds	Long 整型数，代表产生忙消息之前所需消耗的毫秒数

说明

此属性缺省值为 5000 毫秒（5 秒）。

重点此超时值也影响用 OLEContainer 控件或“工具箱”链接或嵌入的文件。如果正在使用链接或嵌入的文件，并且在自动请求前改变此属性，则最好在操作完成后重设置该值。

OLEServerBusyMsgText 属性

如果 ActiveX 部件拒绝自动请求，则返回或设置替补“忙”消息的文本，将它显示在缺省的“部件请求挂起”对话框的地方。在设计时无效。

应用于

App 对象

语法

object.**OLEServerBusyMsgText** [=*string*]

OLEServerBusyMsgText 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	字符串表达式，其值是 ActiveX 部件忙的条件下，在替补的消息框中显示的消息文本

说明

VisualBasic 按 OLEServerBusyTimeout 属性规定的毫秒数不断地重试自动请求。如果在此时间间隔内 ActiveX 部件没有接受请求，则 VisualBasic 显示缺省的“部件忙”对话框。此对话框包含文本和“切换到”按钮，此按钮是为了应用如 MicrosoftExcel 等可视的 ActiveX 部件而设的。有时，缺省对话框可能不能满足用户的要求，如：

程序调用了由无用户界面的 ActiveX 部件所提供对象的方法。例如，由 VisualBasic 专业版生成的 ActiveX 部件，可以不带任何可视窗体而在后台运行。

所调用的 ActivX 部件可能是应用 VisualBasic 企业版的远程自动化特征生成的，并且可能是在距用户一定距离的另一台计算机上运行。

如果程序应用 GetObject 函数加载了 MicrosoftExcel 工作手册，当切换到 MicrosoftExcel 时，此工作手册是不可见的。实际上，MicrosoftExcel 本身可能是不可见的，这种情况下“切换到”按钮不做任何事情。

在这些情况下，缺省文本和“切换到”按钮是不合适的，它可能使程序使用者感到困惑。

OLEServerBusyMsgText 属性允许以替补消息框代替缺省“部件忙”对话框。将 OLEServerBusyMsgText 设置成自定义的消息字符串，就能产生一个包含此消息文本、“确定”按钮及“取消”按钮的简单消息框，它用来代替缺省的“部件忙”对话框。

如果 OLERequestPendingMsgText 等于空字符串(“”)，则显示缺省的“部件忙”对话框。

如果按下缺省“部件忙”对话框或替补的消息框的“取消”按钮，就会在产生自动请求过程中引起 ActiveX 错误-2147418111(&H80010001)。

当知道自动请求将需要数秒钟以上，并且应用的是隐含或远程 ActiveX 部件，就应该设置替补消息。因为网络传输可能会使即便是很短的请求也需耗时数秒钟，所以，对于远程 ActiveX 部件来说，对所有的请求都推荐使用替补消息。

OLEServerBusyMsgTitle 属性

当 ActiveX 部件拒绝自动请求时，返回或设置的显示替补的“忙”消息的标题。在设计时无效。

应用于

App 对象

语法

object.**OLEServerBusyMsgTitle**[=*string*]

OLEServerBusyMsgTitle 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	字符串表达式，其值是 ActiveX 部件忙的条件下，替补的消息框的标题

说明

如果设置了 OLEServerBusyMsgText 属性，则以 OLEServerBusyMsgTitle 属性值作为替补的忙消息的标题，这个忙消息代替了缺省的“部件忙”对话框。OLEServerBusyMsgTitle 属性的缺省值是 App 对象 Title 属性的当前值。这是推荐的设置。

如果 OLEServerBusyMsgText 属性设置为空字符串("")，则 OLEServerBusyMsgTitle 属性被忽略。

请参阅

OLERequestPendingMsgText 属性，OLERequestPendingMsgTitle 属性，OLERequestPendingTimeout 属性，OLEServerBusyMsgText 属性，OLEServerBusyRaiseError 属性，OLEServerBusyTimeout 属性

OLEServerBusyRaiseError 属性

确定拒绝自动请求是否引起错误，而非显示缺省的“部件忙”对话框或其它的替补消息。设计时无效。

应用于

App 对象

语法

object.**OLEServerBusyRaiseError***[**=*boolean***]**

OLEServerBusyRaiseError 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式，说明是否产生了错误，如设置值中所述

设置

boolean **设置值如下：**

设置值	描述
True	（缺省）当 OLEServerBusyTimeout 属性规定的毫秒数消耗完时，产生错误
False	由 OLEServerBusyMsgText 属性的设置决定显示缺省服务器忙对话框还是替补的忙消息

说明

当 ActiveX 部件拒绝自动请求返回控件到程序时，将产生错误，这就允许提供自定义对话框来代替缺省“部件忙”对话框、或替补的忙消息。

产生的自动化错误是-2147418111(&H80010001)。

请参阅

OLERequestPendingMsgText 属性，OLERequestPendingMsgTitle

属性，OLERequestPendingTimeout 属性，OLEServerBusyMsgText 属性，OLEServerBusyMsgTitle 属性，OLEServerBusyTimeout 属性

OLEServerBusyTimeout 属性

在显示缺省的“部件忙”对话框（或其它替补消息）前，返回或设置自动请求持续重试所需的毫秒数。设计时无效。

应用于

App 对象

语法

object.**OLEServerBusyTimeout** [=*milliseconds*]

OLEServerBusyTimeout 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
milliseconds	代表毫秒时间数的 Long 整型数，在此时间内将重试自动请求

说明

此属性的缺省值是 1000 毫秒（10 秒）。

重点此超时值也影响用 OLEContainer 控件或“工具箱”链接或嵌入的文件。如果正在使用链接或嵌入的文件，并且在自动请求前改变此属性，则最好在操作完成后重设置该值。

请参阅

OLERequestPendingMsgText 属性，OLERequestPendingMsgTitle 属性，OLERequestPendingTimeout 属性，OLEServerBusyMsgText 属性，OLEServerBusyMsgTitle 属性，OLEServerBusyRaiseError 属性

OLESetData 事件

当目标部件在源的 DataObject 对象上执行 GetData 方法，但是还没有加载规定格式的数据时，在源部件上发生。

应用于

DBCombo 控件，DBList 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileList 控件，对象，Forms 集合，Frame 控件，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

PrivateSubobject_OLESetData(dataAsDataObject,dataformatAsInteger)

OLESetData 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
data	放置所需数据的 DataObject 对象。部件调用 SetData 方法加载所需的格式
dataformat	整数，确定目标部件所需要的数据格式。源部件用此值来确定向 DataObject 对象加载的内容

说明

在某些情况下，特别是在源部件支持多种格式时，可能希望延迟向源部件 DataObject 对象加载数据以节省时间。此事件允许源对给定的数据格式仅响应一个请求。调用此事件时，源应该检查 *format* 参数以确定需加载的内容，然后在 DataObject 上执行 SetData 方法加载数据，这些数据随后被送回到目标部件。

OLEStartDrag 事件

当部件的 OLEDrag 方法被执行时，或者在 OLEDragMode 属性被设为 Automatic，部件初始化 OLE 拖放操作时发生。

此事件指定源部件支持的数据格式和放效果。也可用于向 DataObject 对象中插入数据。

应用于

DBCombo 控件，DBList 控件，Data 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommanfButton 控件，DirListBox 控件，DriveListBox

控件，FileListBox 控件，Form 对象，Forms 集合，Image 控件，Label 控件，ListBox 控件，MDIForm 对象，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

PrivateSub*object*_**OLEStartDrag**(*data***AsDataObject**,*allowedeffects***AsLong**)

OLEStartDrag 事件语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
data	DataObject 对象，含源所提供的格式以及（可选）适合那些格式的数据。若 DataObject 不包含数据，则当控件调用 GetData 方法时会提供。程序员应提供此事件中该参数的值。SetData 和 Clear 方法不能用于此处
allowedeffects	长整型数，包含源部件支持的效果。其可能的取值列于“设置值”中。程序员应提供此事件中该参数的值

设置值

allowedeffects 设置如下：

常数	值	描述
vbDropEffectNone	0	放目标不接受数据
vbDropEffectCopy	1	放结果保存于从源到目标的数据拷贝中。初始数据没有被拖放操作改变
vbDropEffectMove	2	放结果保存于要从拖源移到放源的数据中。移动后，拖源要删除数据

说明

源部件应该针对所支持的数据使用逻辑 Or 运算符，并把结果放到 allowedeffects 参数中。目标部件使用此值确定合适的动作（以及合适的用户反馈）。

如果部件的 OLEDragMode 属性被设为 Automatic，OLEStartDrag 事件也会发生。这样就允许在部件执行该操作后，再次向 DataObject 对象加入格式及数据。也可以通过清除 DataObject 对象（用 Clear 方法）忽略部件的缺省行为，然后加入自己的数据和格式。

除非目标部件需要，可能希望延迟向 DataObject 对象中加载数据。这样，由于没有加载多余的数据格式，使源部件节省了时间。当目标在 DataObject 上执行 GetData 方法时，若 DataObject 中未包含所需的数据，则源的 OLESetData 事件会发生。此时，数据可被加载到 DataObject 中，这将依次地向目标提供数据。

OLEType 属性

返回在 OLE 容器控件中对象的状态。

应用于

OLEContainer 控件

语法

object.**OLEType**

object 所在处是一个对象表达式，其值是“应用于”列表中的一个对象。

返回值

OLEType 属性返回下列值：

常数	值	描述
VbOLELinked	0	链接的。OLE 容器控件包含一个链接对象。所有对象的数据均由将其创建的应用程序管理。当使用 SaveToFile 方法保存对象时，只将诸如象 SourceDoc、SourceItem 那样的链接信息等，保存到 VisualBasic 应用程序指定的文件中
VbOLEEmbedded	1	嵌入的。OLE 容器控件包含一个内嵌对象。所有对象的数据均在 VisualBasic 应用程序内管理。当使用 SaveToFile 方法保存对象时，将与该对象关联的所有数据，保存到指定的文件中
VbOLENone	3	无用的。OLE 容器控件不包含对象

说明

使用这个属性确定 OLE 容器控件是否包含对象，或是确定 OLE 容器控件所包含的对象的类型。

使用 AppIsRunning 属性可确定创建对象的应用程序是否正在运行。

当创建对象时，可使用 OLETypeAllowed 属性确定所能创建的对象的类型。

请参阅

FileNumber 属性，AppIsRunning 属性，OLETypeAllowed 属性，SaveToFile 方法

OLETypeAllowed 属性

返回或设置 OLE 容器控件所能包含的对象的类型。

应用于

OLEContainer 控件

语法

object.**OLETypeAllowed**[=*value*]

OLETypeAllowed 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	整数或常数，指示对象的类型，在“设置值”中有详细说明

设置值

value 的设置值是：

常数	值	描述
vbOLELinked	0	链接的。OLE 容器控件只能包含链接对象
vbOLEEmbedded	1	嵌入的。OLE 容器控件只能包含内嵌对象
vbOLEEither	2	（缺省值）二者均可。OLE 容器控件既能包含链接的对象，又能包含嵌入的对象

说明

这个属性确定可以创建的对象的类型：

当使用插入对象对话框时，使用 InsertObjDlg 方法来显示这个对话

框。

当使用特殊粘贴对话框时，使用 PasteSpecialDlg 方法来显示这个对话框。

当从系统剪贴板粘贴对象时，使用 Paste 方法从系统剪贴板粘贴对象。

使用 OLEType 属性来确定对象的类型（链接的、嵌入的或都不是）。

请参阅

OLEType 属性，AppIsRunning 属性，PasteSpecialDlg 方法，InsertObjDlg 方法，Paste 方法

OnError 语句

启动一个错误处理程序并指定该子程序在一个过程中的位置；也可用来禁止一个错误处理程序。

语法

OnErrorGoTo*line*

OnErrorResumeNext

OnErrorGoTo0

OnError 语句的语法可以具有以下任何一种形式：

语句	描述
OnErrorGoTo*line*	启动错误处理程序，且该例程从必要的 *line* 参数中指定的 *line* 开始。*line* 参数可以是任何行标签或行号如果发生一个运行时错误，则控件会跳到 *line*，激活错误处理程序。指定的 *line* 必须在一个过程中，这个过程与 OnError 语句相同；否则会发生编译时间错误
OnErrorResume*Next*	说明当一个运行时错误发生时，控件转到紧接着发生错误的语句之后的语句，并在此继续运行。访问对象时要使用这种形式而不使用 OnErrorGoTo
OnErrorGoTo0	禁止当前过程中任何已启动的错误处理程序

说明

如果不使用 OnError 语句，则任何运行时错误都是致命的；也就是说，结果会导致显示错误信息并中止运行。

一个“允许的”错误处理程序是由 OnError 语句打开的一个处理程序；一个“活动的”错误处理程序是处理错误的过程中允许的错误处理程序。如果在错误处理程序处于活动状态时（在发生错误和执行 Resume、ExitSub、ExitFunction 或 ExitProperty 语句之间这段时间）又发生错误，则当前过程的错误处理程序将无法处理这个错误。控件返回调用的过程。如果调用过程有一个已启动的错

误处理程序，则激活错误处理程序来处理该错误。如果调用过程的错误处理程序也是活动的，则控件将再往回传到前面的调用过程，这样一直进行下去，直到找到一个被允许的但不是活动的错误处理程序为止。如果没有找到被允许而且不活动的错误处理程序，那么在错误实际发生的地方，错误本身是严重的。错误处理程序每次将控件返回调用过程时，该过程就成为当前过程。在任何过程中，一旦错误处理程序处理了错误，在当前过程中就会从 Resume 语句指定的位置恢复运行。

注意一个错误处理程序不是 Sub 过程或 Function 过程。它是一段用行标签或行号标记的代码。

错误处理程序依靠 Err 对象的 Number 属性中的值来确定错误发生的原因。在其它任何错误发生之前，或在调用一个可能会导致错误发生的过程之前，错误处理程序应该先测试或存储 Err 对象中相关的属性值。Err 对象中的属性值只反映最近发生的错误。Err.Description 中包含有与 Err.Number 相关联的错误信息。

OnErrorResumeNext 会使程序从紧随产生错误的语句之后的语句继续执行，或是从紧随最近一次调用含有 OnErrorResumeNext 语句的过程的语句继续运行。这个语句可以置运行时错误于不顾，使程序得以继续执行。可以将错误处理程序放置在错误发生的地方，而不必将控件传输到过程中的其它位置。在调用另一个过程时，OnErrorResumeNext 语句成为非活动的，所以，如果希望在例程中进行嵌入错误处理，则应在每一个调用的例程中执行 OnErrorRe

sumeNext 语句。

注意当处理在访问其它对象期间产生的错误时，与其使用 OnErrorGoTo 指令，不如使用 OnErrorResumeNext。每次和对象打交道，在不知道用代码访问哪个对象时，检查一下 Err 都会打消这种疑虑。可以确定是哪个对象产生错误（Err.Source 中指定的对象），也可以确定是哪个对象将错误代码放在 Err.Number 中。

OnErrorGoTo0 停止在当前过程中处理错误。即使过程中包含编号为 0 的行，它也不把行 0 指定为处理错误的代码的起点。如果没有 OnErrorGoTo0 语句，在退出过程时，错误处理程序会自动关闭。

在错误未发生的时候，为了防止错误处理程序代码运行，请像在下段程序中那样，在紧靠着错误处理程序的前面写入 ExitSub、ExitFunction 或 ExitProperty 语句。

```
SubInitializeMatrix(Var1,Var2,Var3,Var4)
    OnErrorGoToErrorHandler
    …
    ExitSub
ErrorHandler:
    …
    ResumeNext
EndSub
```

此处，错误处理程序代码在 ExitSub 语句之后，而在 EndSub 语句之前，从而与过程中的流程分开。错误处理程序代码可以在程序

中的任何地方写入。

当对象作为文件运行时，对象中未捕获的错误都被返回控制应用程序。在开发环境中，如果设置了正确选项，未捕获的错误只返回控制应用程序。请参考主应用程序的文档的有关描述，从而得知，在调试时应该设置哪些选项、如何设置这些选项以及主机能否建立类。

如果建立一个访问其它对象的对象，则应该着手处理从那些对象返回的未处理错误。如果无法处理这种错误，请将 Err.Number 中的错误代码当作自己的一个错误，然后将错误回传给对象的调用者。应该将错误代码添加到 vbObjectError 常数上来指定这个错误。举例来说，如果错误代码为 1052，则使用如下方法指定错误：

```
Err.Number=vbObjectError+1052
```

注意调用动态链接库(DLL)或 Macintosh 代码源期间产生的系统错误不会产生例外情况，也不会被 VisualBasic 的错误捕获操作所捕获。当调用 DLL 函数时，应该（根据 API 的详细说明）检查每一个返回值以确定是成功还是失败，如果失败，则检查 Err 对象的 LastDLLError 属性中的值在 Macintosh 中，LastDLLError 总是返回 0。

请参阅

第七章的“产生和处理错误”，第二部分的“调试、测试和配置部件”，《MicrosoftVisualBasic6.0 部件工具指南》一书中的“生成 ActiveX 部件”，第十章的“处理 ActiveX 部件运行时的错误。

《MicrosoftVisualBasic6.0 程序员指南》一书的“部件编程”，第十三章的‘调试代码和处理错误”，End 语句，Exit 语句，Resume 语句，Err 对象，LastDLLError 属性

示例

本示例先使用 OnErrorGoTo 语句在一个过程中指定错误处理的代码所在。本示例中，试图删除一已经打开的文件从而生成的错误码为 55。这个错误将由示例中的错误处理程序码来处理，处理完后，控制会回到发生错误的语句处。OnErrorGoTo0 语句关闭错误陷阱。然后 OnErrorResumeNext 语句用来改变错误陷阱，以便发觉下一个语句产生的错误的范围。请注意示例中使用 Err.Clear 在错误处理完后，清除 Err 对象的属性。

```
SubOnErrorStatementDemo()
   OnErrorGoToErrorHandler              '打开错误处理程序。
   Open"TESTFILE"ForOutputAs#1          '打开输出文件。
   Kill"TESTFILE"                       '试图删除已打开的文件。
   OnErrorGoto0                         '关闭错误陷阱。
   OnErrorResumeNext                    '改变错误陷阱。
   ObjectRef=GetObject("MyWord.Basic")  '试图启动不存在的对象
'检查可能发生的 Automation 错误。
   IfErr.Number=4400rErr.Number=432Then
      '告诉用户出了什么事。然后清除 Err 对象。
      Msg="Therewasanerrorattemptingtoopentheautomationobject!"
      MsgBoxMsg,,"DeferredErrorTest"
```

```
        Err.Clear               '清除 Err 对象字段。
    EndIf
ExitSub                         '退出程序，以避免进入错误处理程序。
ErrorHandler:                   '错误处理程序。
    SelectCaseErr.Number'检查错误代号。
        Case55              '发生“文件已打开”的错误。
            Close#1         '关闭已打开的文件。
        CaseElse
                            '处理其他错误状态...
    EndSelect
    Resume                  '将控制返回到产生错误的语句。
EndSub
```

On...GoSub、On...GoTo 语句

根据表达式的值，转到特定行执行。

语法

On*expression***GoSub***destinationlist*

On*expression***GoTo***destinationlist*

On...GoSub 和 On...GoTo 语句的语法具有以下几个部分：

部分	描述
expression	必需的。数值表达式，其运算结果应该是一个界于 0 到 255 之间的整数，包含 0 和 255。如果 *expression* 的计算结果不是一个整数，则它会先四舍五入为一个整数
destinationlist	必需的。行号或行标签的列表，之间要以逗号隔开

说明

expression 的值会决定转到 *destinationlist* 中的哪一行。如果 *expression* 的值小于 1 或大于列表的项目个数，则会产生下面的结果之一：

如果表达式的值	则
等于 0	控制权会转移到 On...GoSub 或 On...GoTo 之后的语句
大于串的项目个数	控制权会转移到 On...GoSub 或 On...GoTo 之后的语句
负数	会发生错误
大于 255	会发生错误

可以在同一个列表中混合使用行号和行标签。在 On...GoSub 和 On...GoTo 中也可随意使用任意个行号和行标签。但是，如果使用了太多的行标签或行号，以至于在一行中放不下，那么就必须在一行后使用续行符来衔接到下一行。

提示若要执行多重分支，SelectCase 提供了一种结构化与适应

性更强的方法。

请参阅

GoSub…Return 语句，GoTo 语句，SelectCase 语句

示例

本示例使用 On...GoSub 及 On...GoTo 语句来完成不同的子程序或程序区段。

```
SubOnGosubGotoDemo()
DimNumber,MyString
   Number=2'设置变量初值。
   'BranchtoSub2.
   OnNumberGoSubSub1,Sub2  '在 On...GoSub 退出後，程序会回到此处来继续完成。
   OnNumberGoToLine1,Line2 '完成 Line2 标记之区段。
   '在 On...GoTo 退出之后，程序不会回到此处来。
   ExitSub
Sub1:
   MyString="InSub1":Return
Sub2:
   MyString="InSub2":Return
Line1:
   MyString="InLine1"
Line2:
```

```
    MyString="InLine2"
EndSub
```

OnAddinsUpdate 方法

在保存 Vbaddin.Ini 文件的更改时，该方法便自动地发生。

应用于

IDTExtensibilityInterface

语法

object.IDTExtensibility_OnAddinsUpdate(*custom()*AsVariant)

OnAddinsUpdate 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
custom()	保持自定义数据的 variant 表达式数组

说明

该方法是 IDTExtensibility 接口的部分，该接口应当在提供连接对象的类中实现。

重点不能直接输入上面给出的语法。而应当使用 Implements 语句来为接口生成适当的方法模板。为此，在提供外接程序连接对象的类模块的声明部分输入：

```
ImplementsIDTExtensibility
```

添加这行之后，就能从模块“对象”的下拉框中选择“IDTexten

sibility”。从“过程”下拉框中选择每个方法来获取上面语法中显示的过程模板。注意，所需的代码被自动地添加到类模块中。通过 Implements 语句显露各种接口的方法。当上面的语法输入处理外接程序事件的类模块的声明部分之后，通过模块的“过程”和“对象”下拉框，就可使用接口方法。要在模块中添加代码，从“过程”下拉框中选取方法。

注意虽然 OnAddinsUpdate 方法是 IDTExtensibility 接口的方法，但对于 VisualBasic 程序员而言，它的作用和性能象个事件。换句话说，当保存 Vbaddin.Ini 文件的变更时，在 OnAddinsUpdate 方法中的所有代码便自动地产生，正好象它是一个事件过程。

重要因为接口就是对象与 VisualBasic 之间的协议，所以必须确保实现接口中的所有方法。这意味着全部四个 IDTExtensibility 接口方法均应在类模块中存在，而且，每个方法最少包含一个可执行语句。这可以只有一个说明语句，但它们每个必须最少包含一个可执行语句，以免编译程序把它们当作空过程删除。

OnConnection 方法

当一个外接程序通过“外接程序管理器”对话框或另一个外接程序与 VisualBasicIDE 连接时，该方法便发生。

应用于

IDTExtensibilityInterface

语法

object.**IDTExtensibility_OnConnection**(*vbinst***AsObject**,*connectmode***As vbext_ConnectMode**,*addininst***AsAddIn**,*custom()***AsVariant**)

OnConnection **方法的语法包含下面部分：**

部分	**描述**
object	**对象表达式，其值是“应用于”列表中的一个对象**
vbinst	**对象，代表当前** VisualBasic **会话期实例**
connectmode	**类型** vbext_ConnectMode **的计算值，如“设置值”所指定**
addininst	**代表外接程序实例的** AddIn **对象**
custom()	**保存自定义数据的** variant **表达式数组**

设置值

connectmode(vbext_ConnectMode)**的设置值是：**

常数	**值**	**说明**
vbext_cm_AfterStartup	0	**初始“打开项目”对话框显示后外接程序启动**
vbext_cm_Startup	1	**初始“打开项目”对话框显示前外接程序启动**
vbext_cm_External	2	**外接程序被其它程序或部件从外部启动**

说明

该方法是 IDTExtensibility 接口部分，该接口应当在提供连接对象的类中实现。

重点不能直接输入上面给出的语法。而应当使用 Implements **语句**

来为接口生成适当的方法模板。为此，在提供外接程序连接对象的类模块的声明部分输入：

ImplementsIDTExtensibility

添加这行之后，就能从模块“对象”的下拉框中选择“IDTExtensibility”。从“过程”下拉框中选择每个方法来获取上面语法中显示的过程模板。注意，所需的代码被自动地添加到类模块中。

通过 Implements 语句显露各种接口的方法。当上面的语法输入处理外接程序事件的类模块的声明部分之后，通过模块的“过程”和“对象”下拉框，就可使用接口方法。要在模块中添加代码，从“过程”下拉框中选取方法。

注意虽然 OnAddinsUpdate 方法是 IDTExtensibility 接口的方法，但对于 VisualBasic 程序员而言，它的作用和性能象个事件。换句话说，当一个外接程序通过“外接程序管理器”对话框或另一个外接程序与 VisualBasicIDE 连接时，OnConnection 方法中的所有代码便自动地产生，正如它是个事件过程一样。

重点因为接口就是对象与 VisualBasic 之间的协议，所以必须确保实现接口的所有方法。这意味着全部四个 IDTExtensibility 接口方法均应在类模块中存在，而且，每个方法最少包含一个可执行语句。这可以只有一个说明语句，但它们每个必须最少包含一个可执行语句，以免编译程序把它们当作空过程删除。

OnDisconnection 方法

当外接程序通过编程或“外接程序管理器”对话框与 VisualBasic IDE 分离时，该方法便发生。

应用于

IDTExtensibilityInterface

语法

object.IDTExtensibility_OnDisconnection(*removemode*As vbext_DisconnectMode,*custom()*AsVariant)OnDisconnection 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
removemode	类型 vbext_DisconnectMode 的计算值，如设置值所指定
custom()	保存自定义数据的 variant 表达式数组

设置值

removemode(vbext_ConnectMode)的设置值是：

常数	值	描述
vbext_dm_HostShutdown	0	外接程序因外接程序的主机关闭而删除
vbext_dm_UserClosed	1	外接程序因用户关闭它而删除

说明

这个方法是 IDTExtensibility 接口的部分，该接口应当在提供

连接对象的类中实现。
重点不能直接输入上面给出的语法。而应当使用 Implements 语句来为接口生成适当的方法模板。为此，在提供外接程序的连接对象的类模块的声明部分输入：

```
ImplementsIDTExtensibility
```

添加这行之后，就能从模块“对象”的下拉框中选择“IDTExtensibility”。从“过程”下拉框中选择每个方法来获取上面语法中显示的过程模板。注意，所需的代码被自动地添加进类模块中。
通过 Implements 语句显露各种接口的方法。当上面的语法输入处理外接程序事件的类模块的声明部分之后，通过模块的“过程”和“对象”下拉框，就可使用接口方法。要在模块中添加代码，从“过程”下拉框中选取方法。
注意虽然 OnAddinsUpdate 方法是 IDTExtensibility 接口的方法，但对于 VisualBasic 程序员而言，它的作用和性能象个事件。换句话说，当一个外接程序通过编程或“外接程序管理器”对话框与 VisualBasicIDE 分离时，OnConnection 方法中的所有代码便自动地产生，正如它是一个事件过程一样。
重点因为接口就是对象与 VisualBasic 之间的协议，所以必须确保实现接口中的所有方法。这意味着全部四个 IDTExtensibility 接口方法均应在类模块中存在，而且，每个方法最少包含一个可执

行语句。这可以只有一个说明语句，但它们每个必须最少包含一个可执行语句，以免编译程序把它们当作空过程删除。

OnStartupComplete 方法

在 VisualBasicIDE 启动完成时该方法发生。

应用于

IDTExtensibilityInterface

语法

object.**IDTExtensibility_OnStartupComplete**(*custom()*)**AsVariant**)

OnStartupComplete 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
custom()	保存自定义数据的 variant 表达式数组

说明

该方法是 IDTExtensibility 接口的部分，该接口应当在提供连接对象的类中实现。

重点不能直接输入上面给出的语法。而应当使用 Implements 语句来为接口生成适当的方法模板。为此，在提供外接程序连接对象的类模块的声明部分输入：

ImplementsIDTExtensibility

添加这行之后，就能从模块“对象”的下拉框中选择“IDTExtensibility”。从“过程”下拉框中选择每个方法来获取上

面语法中显示的过程模板。注意，所需的代码被自动地添加进类模块中。

通过 Implements 语句显露各种接口的方法。当上面的语法输入处理外接程序事件的类模块的声明部分之后，通过模块的“过程”和“对象”下拉框，就可使用接口方法。要在模块中添加代码，从“过程”下拉框中选取方法。

注意虽然 OnAddinsUpdate 方法是 IDTExtensibility 接口的方法，但对于 VisualBasic 程序员而言，它的作用和性能象个事件。换句话说，当启动 VisualBasicIDE 完成时，OnStartup%Complete 方法中的所有代码便自动地产生，正如它是一个事件过程一样。

重点因为接口就是对象与 VisualBasic 之间的协议，所以必须确保实现接口中的所有方法。这意味着全部四个 IDTExtensibility 接口方法均应在类模块中存在，而且，每个方法最少包含一个可执行语句。这可以只有一个说明语句，但它们每个必须最少包含一个可执行语句，以免编译程序把它们当作空过程删除。

Open 语句

能够对文件输入/输出(I/O)。

语法

Open*pathname***For***mode*[**Access***access*][*lock*]**As**[#]*filenumber*[**Len=***reclength*]

Open 语句的语法具有以下几个部分：

部分	描述
pathname	必需的。字符串表达式，指定文件名，该文件名可能还包括目录、文件夹及驱动器
mode	必需的。关键字指定文件方式，有 Append、Binary、Input、Output、或 Random 方式。如果未指定方式，则以 Random 访问方式打开文件
access	可选。关键字说明打开的文件可以进行的操作，有 Read、Write、或 ReadWrite 操作
lock	可选。关键字说明限定于其它进程打开的文件的操作，有 Shared、LockRead、LockWrite、和 LockReadWrite 操作
filenumber	必要。一个有效的文件号，范围在 1 到 511 之间。使用 FreeFile 函数可得到下一个可用的文件号
reclength	可选。小于或等于 32，767（字节）的一个数。对于用随机访问方式打开的文件，该值就是记录长度。对于顺序文件，该值就是缓冲字符数

说明

对文件做任何 I/O 操作之前都必须先打开文件。Open 语句分配一个缓冲区供文件进行 I/O 之用，并决定缓冲区所使用的访问方式。

如果 pathname 指定的文件不存在，那么，在用 Append、Binary、Output、或 Random 方式打开文件时，可以建立这一文件。

如果文件已由其它进程打开，而且不允许指定的访问类型，则 Open

操作失败，而且会有错误发生。

如果 mode 是 Binary 方式，则 Len 子句会被忽略掉。

重要在 Binary、Input 和 Random 方式下可以用不同的文件号打开同一文件，而不必先将该文件关闭。在 Append 和 Output 方式下，如果要用不同的文件号打开同一文件，则必须在打开文件之前先关闭该文件。

请参阅

Close 语句，FreeFile 函数

示例

本示例示范 Open 语句的不同用法来做到文件的输出与输入。

下列代码以顺序输入模式打开 TESTFILE 文件。

```
Open"TESTFILE"ForInputAs#1
'若要以其他方式打开文件，必需先关闭此文件。
Close#1
```

下列代码以只允许写操作的二进制方式打开文件。

```
Open"TESTFILE"ForBinaryAccessWriteAs#1
'若要以其他方式打开文件，必需先关闭此文件。
Close#1
```

下列代码以随机方式打开文件，文件中含有用户自定义数据类型 Record 的记录。

```
TypeRecord '定义用户自定义数据类型。
    IDAsInteger
    NameAsString*20
EndType

DimMyRecordAsRecord     '声明变量。
Open"TESTFILE"ForRandomAs#1Len=Len(MyRecord)
'若要以其他方式打开文件，必需先关闭此文件。
Close#1

下列代码以顺序输出方式打开文件；任何过程都可以读写该文件。
Open"TESTFILE"ForOutputSharedAs#1
'若要以其他方式打开文件，必需先关闭此文件。
Close#1

下列代码以只允许读的二进制方式打开文件；其他过程不可以读该文件。
Open"TESTFILE"ForBinaryAccessReadLockReadAs#1
```

OpenAsTextStream 方法

打开一个指定的文件并返回一个 TextStream **对象，该对象可用来对文件进行读、写、追加操作。**

应用于

File 对象

语法

object.**OpenAsTextStream**([*iomode*,[*format*]])

OpenAsTextStream 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 File 对象的名字
iomode	可选的。表明输入/输出方式。可为三个常数之一：ForReading、ForWriting 或 ForAppending
format	可选的。三个 Tristate 值之一，用于指示打开文件的格式。如果省略，则文件以 ASCII 格式打开

设置值

iomode 参数可为下面设置值中的任何值：

常数	值	描述
ForReading	1	打开一个只读文件，不能对此文件进行写操作
ForWriting	2	打开一个用于写操作的文件。如果和此文件同名的文件已存在，则覆盖以前内容
ForAppending	8	打开一个文件并写到文件的尾部。

format 参数可为下面设置值中的任何值：

常数	值	描述
TristateUseDefault	–2	使用系统缺省打开文件
TristateTrue	–1	以 Unicode 格式打开文件
TristateFalse	0	以 ASCII 格式打开文件

说明

OpenAsTextStream 方法提供了和 FileSystemObject 的 OpenTextFile 方法相同的功能。此外，OpenAsTextStream 方法还可以用于对一个文件进行写操作。

下面的代码举例说明了 OpenAsTextStream 方法的使用：

```
SubTextStreamTest
ConstForReading=1,ForWriting=2,ForAppending=3
ConstTristateUseDefault=-2,TristateTrue=-1,
TristateFalse=0
Dimfs,f,ts,s
Setfs=CreateObject("Scripting.FileSystemObject")
fs.CreateTextFile"test1.txt"'创建一个文件
Setf=fs.GetFile("test1.txt")
Setts=f.OpenAsTextStream(ForWriting,TristateUseDefault)
ts.Write"HelloWorld"
ts.Close
Setts=f.OpenAsTextStream(ForReading,TristateUseDefault)
s=ts.ReadLine
MsgBoxs
ts.Close
EndSub
```

请参阅

Copy 方法，CreateTextFile 方法，Delete 方法，Move 方法

(FileSystemObject 对象)，OpenTextFile 方法

OpenTextFile 方法

打开一个指定的文件并返回一个 TextStream 对象，该对象可用于对文件进行读操作或追加操作。

应用于

FileSystemObject 对象

语法

object.**OpenTextFile**(*filename*[,*iomode*[,*create*[,*format*]]])

OpenTextFile 方法有如下几部分：

部分	描述
object	必需的。始终是一个 FileSystemObject 的名字
filename	必需的。字符串表达式，它标识了打开的文件
iomode	可选的。表示输入/输出方式。可为两个常数之一：ForReading 或 ForAppending
create	可选的。Boolean 值，它表示如果指定的 filename 不存在是否可以创建一个新文件。如果创建新文件，其值为 True。若不创建文件其值为 False。缺省值为 False
format	可选的。三种 Tristate 值之一，用于指示打开文件的格式。如果省略，则文件以 ASCII 格式打开

设置值

iomode 参数可为下面设置值的任何一个：

常数	值	描述
ForReading	1	打开一个只读文件。不能对此文件进行写操
ForAppending	8	打开一个文件并写到文件的尾部

format 参数可为下面设置值的任何值：

常数	值	描述
TristateUseDefault	–2	使用系统缺省打开文件
TristateTrue	–1	以 Unicode 格式打开文件
TristateFalse	0	以 ASCII 格式打开文件

说明

下面的代码举例说明了使用 OpenTextFile 方法打开一个用于追加文本的文件：

```
SubOpenTextFileTest
ConstForReading=1,ForWriting=2,ForAppending=3
Dimfs,f
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.OpenTextFile("c:\testfile.txt",ForAppending,TristateFalse)
f.Write"Helloworld!"
f.Close
EndSub
```

请参阅

CreateTextFile 方法，MoveFile 方法，OpenAsTextStream 方法

OptionBase 语句

在模块级别中使用，用来声明数组下标的缺省下界。

语法

OptionBase{*0*|*1*}

说明

由于下界的缺省设置是 0，因此无需使用 OptionBase 语句。如果使用该语句，则必须写在模块的所有过程之前。一个模块中只能出现一次 OptionBase，且必须位于带维数的数组声明之前。

注意 Dim、Private、Public、ReDim 以及 Static 语句中的 To 子句提供了一种更灵活的方式来控制数组的下标。不过，如果没有使用 To 子句显式地指定下界，则可以使用 OptionBase 将缺省下界设为 1。使用 Array 函数或 ParamArray 关键字创建的数组的下界为 0；OptionBase 对 Array 或 ParamArray 不起作用。

OptionBase 语句只影响位于包含该语句的模块中的数组下界。

请参阅

Dim 语句，LBound 函数，OptionCompare 语句，OptionExplicit 语句，OptionPrivate 语句，Private 语句，Public 语句，ReDim 语句，Static 语句

示例

该示例使用 OptionBase 语句来取代数组下标的缺省值 0。LBound 函数返回数组的指定维数的最小可用下标。OptionBase 语句只能

在模块级使用。

```
Optionbase1     '将缺省的数组下标设为 1。

DimLower
DimMyArray(20),TwoDArray(3,4)  '声明数组变量。
DimZeroArray(0To5)             '取代缺省的下标。
'使用 LBound 函数来测试数组的下界。
Lower=LBound(MyArray)          '返回 1。
Lower=LBound(TwoDArray,2)  '返回 1。
Lower=LBound(ZeroArray)            '返回 0。
```

OptionCompare 语句

在模块级别中使用，用于声明字符串比较时所用的缺省比较方法。

语法

OptionCompare{Binary|Text|Database}

说明

如果使用，则 OptionCompare 语句必须写在模块的所有过程之前。OptionCompare 语句为模块指定字符串比较的方法（Binary、Text 或 Database）。如果模块中没有 OptionCompare 语句，则缺省的文本比较方法是 Binary。

OptionCompareBinary 是根据字符的内部二进制表示而导出的一种

排序顺序来进行字符串比较。在 MicrosoftWindows 中，排序顺序由代码页确定。典型的二进制排序顺序如下例所示：A<B<E<Z<a<b<e<z<À<Ê<Ø<à<ê<øOptionCompareText 根据由系统国别确定的一种不区分大小写的文本排序级别来进行字符串比较。当使用 OptionCompareText 对相同字符排序时，会产生下述文本排序级别：

(A=a)<(À=à)<(B=b)<(E=e)<(Ê=ê)<(Z=z)<(Ø=ø)OptionCompareDatabase 只能在 MicrosoftAccess 中使用。当需要字符串比较时，将根据数据库的国别 ID 确定的排序级别进行比较。

请参阅

OptionBase 语句，OptionExplicit 语句，OptionPrivate 语句，Comparison 操作符，InStr 函数，StrComp 函数，InstrRev 函数，Replace 函数，Split 函数

示例

该示例使用 OptionCompare 语句设置缺省的字符串比较方法。OptionCompare 语句只能在模块级使用。

```
'将字符串比较方法设为 Binary。
OptioncompareBinary    '这样，"AAA"将小于"aaa"。
'将字符串比较方法设为 Text。
OptioncompareText  '这样，"AAA"将等于"aaa"。
```

OptionExplicit 语句

在模块级别中使用，强制显式声明模块中的所有变量。

语法

OptionExplicit

说明

如果使用，OptionExplicit 语句必须写在模块的所有过程之前。

如果模块中使用了 OptionExplicit，则必须使用 Dim、Private、Public、ReDim 或 Static 语句来显式声明所有的变量。如果使用了未声明的变量名在编译时间会出现错误。

如果没有使用 OptionExplicit 语句，除非使用 Def*type* 语句指定了缺省类型，否则所有未声明的变量都是 Variant 类型的。

注意使用 OptionExplicit 可以避免在键入已有变量时出错，在变量的范围不是很清楚的代码中使用该语句可以避免混乱。

请参阅

Const 语句，Deftype 语句 s，Dim 语句，函数语句，OptionBase，语句，OptionCompare 语句，OptionPrivate 语句，Private 语句，Public 语句，ReDim 语句，Static 语句，Sub 语句

示例

该示例使用 OptionExplicit 语句来强制显式声明所有变量。如果试图使用一个未声明的变量，则会在编译时导致错误。OptionExplicit 语句只能在模块级使用。

```
Optionexplicit '强制显式地声明变量。
```

```
DimMyVar   '声明变量。
MyInt=10   '未声明的变量将产生错误。
MyVar=10   '已声明的变量则不会产生错误。
```

OptionPrivate 语句

在允许引用跨越多个工程的主机应用程序中使用 OptionPrivateModule，可以防止在模块所属的工程外引用该模块的内容。在不允许这种引用的主机应用程序中，例如，VisualBasic 的独立方式版本，OptionPrivate 就不起作用。

语法

OptionPrivateModule

说明

如果使用 OptionPrivate 语句，则必须是写在模块级别中的任何过程之前。

如果模块中使用了 OptionPrivateModule，则其公用部分（例如，在模块级定义的变量，对象，以及用户定义类型）在该模块所属的工程内仍是可用的，但对其它应用程序或工程则是不可用的。

注意只有在支持同时加载多个工程，且允许在加载的工程间引用的主应用程序中可使用 OptionPrivate。例如，MicrosoftExcel 允许加载多个工程，OptionPrivateModule 就可以用来限制跨工程的可见性。尽管 VisualBasic 允许加载多个工程，但在 VisualBasic 中是从不允许跨工程引用的。

请参阅

OptionBase 语句，OptionCompare 语句，OptionExplicit 语句，Private 语句，Public 语句

示例

该示例演示 OptionPrivate 语句的用法，该语句用于模块级，表示整个模块都是私有的。若使用 OptionPrivateModule，则该项目中的其它模块可以使用没有声明为 Private 的模块级部分，但其它工程或应用程序不能使用。

```
OptionprivateModule    '表示该模块是私有的。
```

OptionButton 控件

OptionButton 控件显示一个可以打开或者关闭的选项。

语法

OptionButton

说明

在选项组中用 OptionButton 显示选项，用户只能选择其中的一项。在 Frame 控件、PictureBox 控件或者窗体这样的容器中绘制 OptionButton 控件，就可以把这些控件分组。为了在 Frame 或者 PictureBox 中将 OptionButton 控件分组，首先绘制 Frame 或 PictureBox，然后在内部绘制 OptionButton 控件。同一容器中的 OptionButton 控件为一个组。

OptionButton 控件和 CheckBox 控件功能相似，但是二者间也存在

着重要差别。在选择一个OptionButton时,同组中的其它OptionButton控件自动无效。相反，可以选择任意数量的CheckBox控件。

属性

OptionButton,DataFormat属性,RightToLeft属性,OLEDropMode属性, BackColor, ForeColor属性, FontBold, FontItalic, FontStrikethru, FontUnderline 属性, FontName 属性, FontSize属性,Height,Width属性,Left,Top属性,Picture属性,TabIndex属性, Tag属性, value属性, Visible属性, Alignment属性, DragIcon 属性, DragMode 属性, hWnd 属性, MouseIcon 属性, MouserPointer 属性, Style 属性, TabStop 属性, Appearance属性, Caption属性, Enabled属性, HelpContextID属性, Index属性(ControlArray), Name属性, Parent属性, Font属性, Container属性, ToolTipText属性, DisabledPicture属性, DownPicture属性, MaskColor属性, UseMaskColor属性, WhatsThisHelpID属性

方法

Refresh 方法, SetFocus 方法, Drag 方法, Move 方法, Zorder方法, OLEDrag方法, ShowWhatsThis方法

事件

Click 事件, DragDrop 事件, DragOver 事件, GotFocus 事件, KeyDown, KeyUp事件, KeyPress事件, LostFocus事件, MouseDown, MouseUp事件, MouseMove事件, Validate事件, DblClick

事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件

请参阅

CheckBox 控件，Form 对象，Forms 集合，Frame 控件，PictureBox 控件

Orientation 属性

返回或设置一个值，该值指出文档是以纵向还是横向的方式被打印。在设计时是不可用的。

应用于

ADOData 控件，Printer 对象，Printers 集合

语法

object.**Orientation**[=*value*]

Orientation 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个决定页方向的值或常数，“设置值”中有详细描述

设置值

value 的设置值为：

常数	值	描述
vbPRORPortrait	1	文档打印以纸的窄边作顶部
vbPRORLandscape	2	文档打印以纸的宽边作顶部

说明

这些常数在的对象浏览器中的 VisualBasic(VB)对象库中列出。注意 Printer 对象的此属性的效果依赖于打印机厂商提供的驱动程序。某些属性设置值可能不起作用，或者一些不同的属性设置值可能有相同的效果。可接受范围之外的设置值可能产生也可能不产生错误。详细信息，请参阅厂家的具体驱动程序的文档。

请参阅

Printer 对象，Printers 集合，Slider 控件

Orientation 属性（CommonDialog 控件）

返回或设置一个值，指示是否文档以纵向或横向模式打印。在设计时不可用。

应用于

CommonDialog 控件

语法

object.**Orientation**[=*value*]

Orientation 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	决定页面方向的值或常数，可参阅“设置值”中的描述

设置值

value 设置值：

常数	值	描述
cdlPortrait	1	以纸的短边为顶端打印文档
cdlLandScape	2	以纸的长边为顶端打印文档

说明

在对象浏览器.的 VisualBasic(VB)对象库中有这些常数的列表。

注意 Printer 对象属性的效果依赖于打印机厂商提供的驱动程序。有些属性设置值可能没有效果，或者几个不同的属性设置值可能有相同的效果。在允许范围之外的设置值可能产生或者不产生错误。关于更多的信息，请参阅厂商文档中指定的驱动程序。

Page 属性

返回当前页号。

应用于

Printer 对象，Printers 集合

语法

object.**Page**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

VisualBasic 保持一个已打印页数的计数器，它从应用程序开始或从在 Printer 对象上上次使用 EndDoc 语句起计数。在下述情况下该计数器从一开始并每次加一：

使用 NewPage 方法。

使用 Print 方法并且要打印的文本与当前页容纳不下。

注意该页容纳不下的图形方法输出不能产生新页。输出被裁剪以与页的可打印区域相适合。

请参阅

EndDoc 方法，NewPage 方法

示例

本例打印三页文本，每页顶部都有当前页号。要试用此例，先将下面的代码粘贴到一个窗体的声明部分，然后按 F5 键并单击窗体。

```
PrivateSubForm_Click()
   DimHeader,I,Y                          '声明变量。
   Print"Nowprinting..."                      '在窗体上放置注意信息。
   Header="PrintingDemo-Page"                 '设置页眉字符串。
   ForI=1To3
```

```
        Printer.PrintHeader;                    '打印页眉。
        Printer.PrintPrinter.Page               '打印页号。
        Y=Printer.CurrentY+10                   '设置行位置。
        '画一条跨页横线。
        Printer.Line(0,Y)-(Printer.ScaleWidth,Y) '画线。
        ForK=1To50
            Printer.PrintString(K,"");          '打印空格字符串。
            Printer.Print"VisualBasic";         '打印文本。
            Printer.PrintPrinter.Page           '打印页号。
        Next
        Printer.NewPage
    NextI
    Printer.EndDoc
    End
EndSub
```

Paint 事件

在一个对象被移动或放大之后，或在一个覆盖该对象的窗体被移开之后，该对象部分或全部暴露时，此事件发生。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Form 对象，

Forms 集合，PictureBox 控件

语法

PrivateSubForm_Paint()

PrivateSub*object*_**Paint**([*index***AsInteger**])

Paint 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件

说明

如果需要代码中各种图形方法的输出，则 Paint 事件过程就很有用。使用 Paint 过程，可以确保这样的输出在必要时能被重绘。使用 Refresh 方法时，Paint 事件即被调用。如果 AutoRedraw 属性被设置为 True，重新绘图会自动进行，于是就不需要 Paint 事件。如果 ClipControls 属性被设置为 False，在 Paint 事件过程中的绘图方法仅影响该窗体中新暴露的区域；否则，绘图方法将在该窗体未被控件覆盖的所有区域（Image、Label、Line 和 Shape 控件除外）。在 Resize 事件过程中使用 Refresh 方法可在每次调整窗体大小时强制对整个对象进行。

注意对某些任务使用 Paint 事件过程能导致一个层叠事件。通常来说，在下列情况下，要避免用 Paint 事件过程：

?移动一个窗体或控件，或者是调整其大小。

?对影响大小或外观的任何变量进行改变，如：设置对象的

BackColor 属性等。

调用 Refresh 方法。

对上述这些任务来说，Resize 事件可能更为合适。

请参阅

Resize 事件，Refresh 方法，AutoRedraw 属性，ClipControls 属性

示例

本例将画出一个与一个窗体各边的中点相交的菱形，并且当窗体的大小改变时，菱型也随着自动调整。要尝试这个例子，可将代码粘贴到一个窗体的声明部分，然后按 F5 键并调整窗体的大小。

```
PrivateSubForm_Paint()
   DimHalfX,HalfY                    '声明变量.
   HalfX=ScaleLeft+ScaleWidth/2 '设置到宽度的一半。
   HalfY=ScaleTop+ScaleHeight/2 '设置到高度的一半。
   '画一个菱形。
   Line(ScaleLeft,HalfY)-(HalfX,ScaleTop)
   Line-(ScaleWidth+ScaleLeft,HalfY)
   Line-(HalfX,ScaleHeight+ScaleTop)
   Line-(ScaleLeft,HalfY)
EndSub

PrivateSubForm_Resize
   Refresh
```

EndSub

PaintPicture 方法

用以在 Form,PictureBox 或 Printer 上绘制图形文件（.bmp、.wmf、.emf、.cur、.ico 或.dib）的内容。不支持命名参数。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**PaintPicture***picture*,*x1*,*y1*,*width1*,*height1*,*x2*,*y2*,*width2*,*height2*,*opcode*

PaintPicture 方法的语法包含下列部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 *object*，带有焦点的 Form 对象缺省为 *object*
picture	必需的。要绘制到 *object* 上的图形源。Form 或 PictureBox 必须是 Picture 属性
x1,*y1*	必需的。均为单精度值，指定在 *object* 上绘制 picture 的目标坐标（x-轴和 y-轴）。*object* 的 ScaleMode 属性决定使

用的度量单位。

width1 可选的。单精度值，指示 *picture* 的目标宽度。*object* 的 ScaleMode 属性决定使用的度量单位。如果目标宽度比源宽度(width2)大或小，将适当地拉伸或压缩 *picture*。如果该参数省略，则使用源宽度

height1 可选的。单精度值，指示 *picture* 的目标高度。*object* 的 ScaleMode 属性决定使用的度量单位。如果目标高度比源高度(height2)大或小，将适当地拉伸或压缩 *picture*。如果该参数省略，则使用源高度

x2,y2 可选的。均为单精度值，指示 *picture* 内剪贴区的坐标（x-轴和 y-轴）。*object* 的 ScaleMode 属性决定使用的度量单位。如果该参数省略，则缺省为 0

width2 可选的。单精度值，指示 *picture* 内剪贴区的源宽度。*object* 的 ScaleMode 属性决定使用的度量单位。如果该参数省略，则使用整个源宽度

height2 可选的。单精度值，指示 *picture* 内剪贴区的源高度。*object* 的 ScaleMode 属性决定使用的度量单位。如果该参数省略，则使用整个源高度

opcode 可选的。是长型值或仅由位图使用的代码。它用来定义在将 *picture* 绘制到 *object* 上时对 *picture* 执行的位操作（例如，vbMergeCopy 或 vbSrcAnd 操作符）。关于位操作符常数的完整列表，请参阅 VisualBasicHelp 文件中的 RasterOpConstants 主题

在使用 opcode 时有一些限制。例如，如果资源是图标或图元文件，则只能使用 vbSrcCopy，而不能使用其他的 opcode；并且，与图案(或 SDK 术语中的"画笔")，如 MERGECOPY、PATCOPY、PATPAINT 和 PATINVERT，相交互的 opcode 实际上是同目标的 FillStyle 属性交互

注意 *Opcode* 用于将按位操作传递到位图。当传递其他图像类型时将一个值给该参数会造成“无效过程调用或参数”错误。这是设计的原因。要避免这个错误，对于除位图外的图像，将 *Opcode* 参数置为空

说明

通过使用负的目标高度值(height1)和/或目标宽度值(width1)，可以水平或垂直翻转位图。

可以省略任何多个可选的尾部的参数。如果省略了一个或多个可选尾部参数，则不能在指定的最后一个参数后面使用逗号。如果想指定某个可选参数，则必须先指定语法中出现在该参数前面的全部参数。

注意，在将一个.Bmp 加载入 PictureBox 控件和使用 WindowsAPI 函数 BitBlt()添加图片之间有一点不同。当您对一个图像使用 BitBlt()时，PictureBox 控件不知道象您使用 LoadPicture 方法那样去调整大小。将 ScaleWidth 和 ScaleHeight 属性设置为图像的大小也不起作用。如果您想在使用 BitBlt 之后用 PictureBox 调整新图片的大小，必须用代码手工做，转换单位并处理边框，下面是如

何这样做的一个简单示例：

```
SubResizePictureBoxToImage(picasPictureBox,twipWd_
asInteger,twipHtasInteger)
'该代码假设所有的单位都为缇。如果
'不是，必须在调用该例程之前，转换为缇。
'这里也假设图像显示在 0,0 处。
DimBorderHtasInteger,BorderWdasInteger
BorderWd=Pic.Width-Pic.ScaleWidth
BorderHt=Pic.Height-Pic.ScaleHeight
pic.Movepic.Left,pic.Top,twipWd+BorderWd,_
twipHt+BorderHt
EndSub
```

请参阅

Scale 方法，ScaleX，ScaleY 方法，ScaleMode 属性

Palette 属性

返回或设置一幅图像，该图像包含了用于控件的调色板。

应用于

AmbientProperties 对象，PropertyPage 对象，UserControl 对象，UserDocument 对象，Form 对象，Forms 集合

语法

object.**Palette**=*path*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
path	包含了所用调色板的位图图像的路径

说明

可以使用.dib、.gif 或.bmp 文件来设置调色板。

PaletteMode 属性

返回或设置一个值，该值决定了哪个调色板将用于对象上的控件。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Form 对象，Forms 集合

语法

object.**PaletteMode**=*integer*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
integer	决定使用的调色板模式，如下列“设置值”中所示

设置值

integer 的设置值是：

常数	值	描述
vbPaletteModeHalfTone	0	（缺省的）使用 Halftone 调色板
vbPaletteModeUseZOrder	1	使用来自于含有调色板的最高层控件的调色板
vbPaletteModeCustom	2	使用 Palette 属性中指定的调色板
vbPaletteModeContainer	3	对于支持环境 Palette 属性的容器使用容器调色板。只适用于 UserControls
vbPaletteModeNone	4	不使用任何调色板。只适用于 UserControls
vbPaletteMode*object*	5	使用 ActiveX 设计器的调色板（只适用于包含调色板的 ActiveX 设计器）

说明

如果未获得调色板，则半色调调色板就成为缺省的调色板。

注意对于 VisualBasic 老版本，PaletteMode 相当于 UseZOrder。

PaperBin 属性

返回或设置一个值，该值指出在打印时打印机上供纸的缺省纸盒。在设计时不可用。

应用于

Printer 对象，Printers 集合

语法

object.**PaperBin**[*=value*]

PaperBin 属性的语法包含下面部分：

部分	描述
Object	对象表达式，其值是“应用于”列表中的一个对象
Value	一个用来指定缺省纸盒的值或常数，“设置值”中有详细描述

设置值

value **的设置值是：**

常数	值	描述
vbPRBNUpper	1	从上层纸盒进纸
vbPRBNLower	2	从下层纸盒进纸
vbPRBNMiddle	3	从中间纸盒进纸
VbPRBNManual	4	等待手动插入每页纸
VbPRBNEnvelope	5	从信封进纸器进纸
VbPRBNEnvManual	6	从信封进纸器进纸；但要等待手动插入
VbPRBNAuto	7	（缺省值）从当前缺省纸盒进纸
VbPRBNTractor	8	从拖拉进纸器进纸
vbPRBNSmallFmt	9	从小型进纸器进纸
vbPRBNLargeFmt	10	从大型纸盒进纸
vbPRBNLargeCapacity	11	从大容量进纸器进纸
vbPRBNCassette	14	从附加的卡式纸盒进纸

说明

这些常数在 VisualBasic(VB)的对象浏览器中的对象库中列出。并非所有的纸盒选项在每个打印机上都可使用。查看打印机的文档可获得更多的关于这些选项的具体说明。

注意 Printer 对象的此属性的效果依赖于打印机厂家提供的驱动程序。某些属性设置值可能不起作用，或者一些不同的属性设置值可能有相同的效果。可接受范围之外的设置值可能产生也可能不

产生错误。详细信息，请参阅厂家的具体驱动程序的文档。

请参阅

Printer 对象，Printers 集合

PaperSize 属性

返回或设置一个值，该值指出当前打印机的纸张大小。在设计时是不可用。

应用于

Printer 对象，Printers 集合

语法

object.**PaperSize** [=*value*]

PaperSize 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定纸张大小的值或常数，“设置值”中有详细描述

设置值

value 的设置值是：

常数	值	描述
VbPRPSLetter	1	信笺，81/2x11 英寸
VbPRPSLetterSmall	2	小型信笺，81/2x11 英寸

VbPRPSTabloid	3	**小型报，**11x17 **英寸**
VbPRPSLedger	4	**分类帐，**17x11 **英寸**
VbPRPSLegal	5	**法律文件，**81/2x14 **英寸**
VbPRPSStatement	6	**声明书，**51/2x81/2 **英寸**
VbPRPSExecutive	7	**行政文件，**71/2x101/2 **英寸**
VbPRPSA3	8	A3,297x420 **毫米**
VbPRPSA4	9	A4,210x297 **毫米**
VbPRPSA4Small	10	A4 **小号，**210x297 **毫米**
VbPRPSA5	11	A5,148x210 **毫米**
VbPRPSB4	12	B4,250x354 **毫米**
VbPRPSB5	13	B5,182x257 **毫米**
VbPRPSFolio	14	**对开本，**81/2x13 **英寸**
VbPRPSQuarto	15	**四开本，**215x275 **毫米**
VbPRPS10x14	16	10x14 **英寸**
VbPRPS11x17	17	11x17 **英寸**
VbPRPSNote	18	**便条，**81/2x11 **英寸**
VbPRPSEnv9	19	#9 **信封，**37/8x87/8 **英寸**
VbPRPSEnv10	20	#10 **信封，**41/8x91/2 **英寸**
VbPRPSEnv11	21	#11 **信封，**41/2x103/8 **英寸**
VbPRPSEnv12	22	#12 **信封，**41/2x11 **英寸**
VbPRPSEnv14	23	#14 **信封，**5x111/2 **英寸**
VbPRPSCSheet	24	C **尺寸工作单**
VbPRPSDSheet	25	D **尺寸工作单**

VbPRPSESheet	26	E 尺寸工作单
VbPRPSEnvDL	27	DL 型信封，110x220 毫米
VbPRPSEnvC3	29	C3 型信封，324x458 毫米
VbPRPSEnvC4	30	C4 型信封，229x324 毫米
VbPRPSEnvC5	28	C5 型信封，162x229 毫米
VbPRPSEnvC6	31	C6 型信封，114x162 毫米
VbPRPSEnvC65	32	C65 型信封，114x229 毫米
VbPRPSEnvB4	33	B4 型信封，250x353 毫米
VbPRPSEnvB5	34	B5 型信封，176x250 毫米
VbPRPSEnvB6	35	B6 型信封，176x125 毫米
VbPRPSEnvItaly	36	信封，110x230 毫米
VbPRPSEnvMonarch	37	信封大王，37/8x71/2 英寸
VbPRPSEnvPersonal	38	信封，35/8x61/2 英寸
VbPRPSFanfoldUS	39	U.S.标准复写簿，147/8x11 英寸
VbPRPSFanfoldStdGerman	40	德国标准复写簿，81/2x12 英寸
VbPRPSFanfoldLglGerman	41	德国法律复写簿，81/2x13 英寸
VbPRPSUser	256	用户定义

说明

这些常数在 VisualBasic(VB)中的对象浏览器中的对象库中列出。

打印机 Height **和** Width **属性的设置自动地将** PaperSize **设置为**

vbPRPSUser。

注意 Printer 对象此属性的效果依赖于打印机厂家提供的驱动程序。某些属性设置值可能不起作用，或者一些不同的属性设置值可能有相同的效果。接受范围之外的设置值可能产生也可能不产生错误。详细信息，请参阅厂家具体驱动程序的文档。

请参阅

Printer 对象，Printes 集合

Parent 属性

返回包含控件、或其它对象或者集合的窗体、对象或集合。

应用于

ADOData 控件，TreeView 控件，ImageComboCoontrol，ImageList 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，CoolBar 控件，CommonDialog 控件，MSCommConreol，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MicosoftInternetTransfer 控件，MAPISession 控件，MAPIMessagesCoontrol，MaskedEdit 控件，MSHFlexGrid 控件，MSFlexGrid 控件，SSTab 控件，PictureClip 控件，RemoteData 控件，RichTextBox 控件，Extender 对象，UserDocument 对象，Data 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirL

istBox 控件，DriveListBox 控件，FileListBox 控件，Frame 控件，HscrollBar，VScrollBar 控件，Image 控件，Label 控件，Line 控件，ListBox 控件，Menu 控件，OptionButton 控件，PictureBoxCoontrol，Shape 控件，TextBox 控件，Timer 控件，OLEContainer 控件

语法

object.**Parent**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

使用 Parent 属性可访问一个对象的父亲的属性、方法或控件。例如：

```
MyButton.Parent.MousePointer=4
```

Parent 属性在应用程序中是很有用的，其中可将对象作为参数来传递。例如，可以传递一个控件变量给模块中的一个一般的过程，并使用 Parent 属性访问其父窗体。

在 Parent 属性和 MDIChild 属性之间没有任何联系。尽管如此，但在一个 MDIForm 对象和任何已经将其 MDIChild 属性设为 True 的 Form 对象之间仍有父-子关系。

请参阅

MDIChild 属性，Active 控件属性，ActiveForm 属性

示例

该例子从一个没有焦点的窗体传递一个控件给模块中的一个过程，然后在父窗体上显示控件的状态。要试用此例，须创建三个窗体：包含一个 CommandButton 控件的 Form1，以及各包含一个 CheckBox 控件的 Form2 和 Form3。还必须创建一个新的模块（在“工程”菜单中单击“添加模块”）。将下面的代码粘贴到各自的窗体或模块的声明部分中，然后按下 F5 键以运行该程序。

```
'在 Form1 中输入下面的代码。
PrivateSubForm_Load()
    Form2.Show          '显示所有的窗体。
    Form3.Show
    Form2.AutoRedraw=True
    Form3.AutoRedraw=True
EndSub

PrivateSubCommand1_Click()
    ReadCheckBoxForm2.Check1          '调用别的模块中的过程。
    ReadCheckBoxForm3.Check1          '并将控件作为参数进行发送。
EndSub

'在 Module1 中输入下面的代码。
SubReadCheckBox(SourceAsControl)
    IfSource.ValueThen
        Source.Parent.Cls                    '清除父窗体。
```

```
        Source.Parent.Print"CheckBoxisON."   '在父窗体上显示。
    Else
        Source.Parent.Cls                    '清除父窗体。
        Source.Parent.Print"CheckBoxisOFF."'在父窗体上显示。
    EndIf
EndSub
```

Parent 属性(VBA 外接程序对象模型)

返回包含另一对象或集合的对象或集合，此属性为只读。

应用于

CodeModule 对象，CodePanes 集合，LinkedWindows 集合，属性集合(VBA 外接程序对象模型)，References 集合，VBComponents 集合，VBProjects 集合，Windows 集合

说明

大部分的对象都可有一个 Parent 属性或是 Collection 属性，它们是用来在这个对象模式中指向此对象的父对象。Collection 属性只有在父对象是集合时才可被使用。

使用 Parent 属性来访问一个对象的父对象的属性、方法及控件。

请参阅

集合属性

示例

下列示例使用 Parent 属性返回在对象层次中某一对象之父对象

的名称。

```
Debug.PrintApplication.VBE.ActiveVBProject.VBComponents.Parent.Name
```

Parent 属性（UserControl 对象）

返回对该控件所属的容器对象的引用。在设计时不可用，在运行时只读。

语法

object.**Parent**

Parent 属性的语法包括下述部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的对象

说明

Parent 属性返回一个容器对象的引用，即使 UserControl 的 AmbientProperties 对象没有 Parent 属性。可以使用 UserControl 对象的 Parent 属性访问容器的对象模型。

通过测试 TypeName(Parent)，可以确定控件所处的容器。

Excel 返回工作簿。

Word 返回文档。

Powerpoint 返回演示文稿。

VB4/VB5 返回窗体。

InternetExplorer 返回一个 Script 属性为 IOmWindow 对象的对

象。

例如，在 InternetExplorer 中，下述代码将改变控件所处的 HTML 页面的背景色：

```
Parent.Script.get_document.bgColor="Blue"
```

有关 InternetExplorerScripting*object*Model 的更详细信息可以在 Microsoft 的 Web 站点获得。

重点总是使用后期绑定来调用 InternetExplorerScripting*object*Model。使用事前绑定几乎肯定会导致今后的兼容性问题，而采用后期绑定将会正常工作。在其它容器中，可以使用事前绑定。

ParentControls 集合

一个集合，允许访问另一个控件容器中的控件。

语法

ParentControls(*index*)

index 所在处表示从 0 到 ParentControls.Count–1 的整数。

ParentControls 属性

返回控件的容器中其它控件的集合。在创建控件时，ParentControls 属性不可用，在控件运行时，该属性是只读的。

应用于

User 控件对象

语法

object.**ParentControls**

ParentControls 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

大多数情况下，控件的容器是窗体；该集合的功能与窗体上的控件集合类似，但此集合中还包括窗体本身。

如果控件想在窗体的控件上执行某些操作时，该集合十分有用；控件可通过集合进行迭代。

不能通过该集合添加和删除控件；必须用容器允许的方式变更控件。

该集合的内容完全由容器决定。

请参阅

控件集合，Count 属性(VB 集合)，Item 属性(ActiveX 控件)

ParentControlsType 属性

返回或设置一个值，该值决定 ParentControls 集合是包含对有容器的 Extender 对象控件的引用，还是包含对无容器的 Extender 对象控件的引用。

语法

object.**ParentControlsType**[=type]

ParentControlsType 属性的语法包括下述部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的对象
type	一个整数或命名常数，指定 ParentControls 集合返回的内容

设置值

type 的设置值如下：

常数	值	描述
VbExtender	1	（缺省）ParentControls 集合返回控件及扩展对象
vbNoExtender	0	ParentControls 集合返回控件自身，不包括扩展对象

说明

使用 ParentControls 集合可以访问该控件所处的容器中的其它控件。为引用这些控件，默认设置是包括由该容器提供的扩展对象的属性和方法。

有些容器，如 InternetExplorer，提供一种 VisualBasic 不能使用的扩展对象。在这种容器中，当用户试图使用缺省设置访问 ParentControls 中的对象时，VisualBasic 将产生一个错误。

可以通过将 ParentControlsType 设为 vbNoExtender，以使 ParentControls 只包含对该控件本身的引用，而不包含扩展对象的属性和方法在 InternetExplorer 的 HTML 页面中访问该控件。

由于可以在运行时设置该属性，用户可以根据控件所在的容器，在 vbExtender 和 vbNoExtender 之间来回切换。如果有必要，可以在支持两种设置的容器中交替使用这些设置。

ParentFolder 属性

返回指定文件或文件夹的父文件夹对象。只读。

应用于

File 对象，Folder 对象

语法

object.**ParentFolder**

object 总是一个 File 或 Folder 对象。

说明

下面的代码用一个文件举例说明了 ParentFolder 属性的用法：

```
SubShowFileAccessInfo(filespec)
Dimfs,f,s
Setfs=Createobject("Scripting.FileSystem object")
Setf=fs.GetFile(filespec)
s=UCase(f.Name)&"in"&UCase(f.ParentFolder)&vbCrLf
s=s&"Created:"&f.DateCreated&vbCrLf
s=s&"LastAccessed:"&f.DateLastAccessed&vbCrLf
s=s&"LastModified:"&f.DateLastModified
MsgBoxs,0,"FileAccessInfo"
```

```
EndSub
```

请参阅

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性，Path 属性，ShortName 属性，ShortPath 属性，Size 属性，Type 属性

Partition 函数

返回一个 Variant(String)，指定一个范围，在一系列计算的范围中指定的数字出现在这个范围内。

语法

```
Partition(number,start,stop,interval)
```

Partition 函数的语法含有下面这些命名参数：

部分	描述
number	必需的。整数，在所有范围中判断这个整数是否出现
start	必需的。整数，数值范围的开始值，这个数值不能小于 0
stop	必需的。整数，数值范围的结束值，这个数值不能等于或小于 *start*
Interval	必需的。整数，是一些区间，被从 start 到 stop 的一系列范围跨过。该数值不能小于 1

说明

Partition 函数会标识 number 值出现的特定范围，并返回一个 Variant(String)来描述这个范围。Partition 函数在查询中是最有用的。可以创建一个选择查询显示有多少订单落在几个变化的范围内，例如，定单数从 1 到 1000、1001 到 2000，以此类推。

下面的表格使用三组 *start*，*stop* 以及 *interval* 部分，来显示怎样决定这个范围。第一个范围和最后一个范围两列显示 Partition 的返回值，此范围的低端(*lowervalue*)和高端(*uppervalue*)是以冒号分开的。

Start	stop	Interval	第一个范围之前	第一个范围	最后一个范围	最后一个范围之后
0	99	5	":-1"	"0:4"	"95:99"	"100:"
20	199	10	":19"	"20:29"	"190:199"	"200:"
100	1010	20	":99"	"100:119"	"1000:1010"	"1011:"

从上面的表格中得知，在第三行中，由 *start* 和 *stop* 所定义的数值范围不能以 *interval* 来均分。所以，即使 *interval* 是 20，最后一个范围也只能扩展到 *stop*（11 个数）。

如果需要的话，Partition 会在返回的范围中加上足够的空白，以便让返回值在冒号的左右两侧有相同的字符数，其值就是 *stop* 中的字符数再加一。如此可确保当要使用 Partition 与其它的数值作运算时，所得的字符串，可以在之后的排序操作中得到正确的结

果。

如果 *interval* 是 1，则范围便是 *number*:*number*，而不管 *start* 和 *stop* 参数如何。比如说，如果 *interval* 是 1，*number* 是 100，而 *stop* 是 1000，则 Partition 会返回"100:100"。

如果任何部分是 Null，则 Partition 会返回一个 Null。

示例

本示例假设您有一个“Orders”表，且里头含有一个“Freight’字段。程序建立一个“选择”来计算运费落在某些范围内的订单数量。Partition 函数是用来确定这些范围，然后调用 SQLCount 函数来计算在每个范围内的订单数量。本示例中，Partition 函数的参数值为 *start*=0，*stop*=500，*interval*=50。第一个范围会是 0：49，每隔 50 一个范围，依次而下直到运费为 500 为止。

```
SELECTDISTINCTROWPartition([freight],0,500,50)ASRange,
Count(Orders.Freight)ASCount
FROMOrders
GROUPBYPartition([freight],0,500,50);
```

PasswordChar 属性

返回或设置一个值，该值指示所键入的字符或占位符在 TextBox 控件中是否要显示出来；返回或设置用作占位符。

应用于

TextBox 控件

语法

object.**PasswordChar**[=*value*]

PasswordChar 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个指定占位符的字符串表达式

说明

为了在对话框中创建一个密码域应使用此属性。虽然能够使用任何字符，但是大多数基于 Windows 的应用程序使用星号(*)(Chr(42))。

此属性不影响 Text 属性；Text 准确地包括所键入或代码中所设置的内容。将 PasswordChar 设置为长度为 0 的字符串("")（缺省值），将显示实际的文本。

能够将任意字符串赋予此属性，但只有第一个字符是有效的，所有其它的字符将被忽略。

注意如果 MultiLine 属性被设为 True，那么设置 PasswordChar 属性将不起效果。

请参阅

Text 属性

示例

本例说明 PasswordChar 属性是如何影响 TextBox 控件显示文本的方法的。要试用此例，先将下面的代码粘贴到包含 TextBox 的窗体

的声明部分，然后按 F5 键并单击窗体。每次单击窗体，文本将在星号(*)密码和普通文本之间转换。

```
PrivateSubForm_Click()
    IfText1.PasswordChar=""Then
        Text1.PasswordChar="*"
    Else
        Text1.PasswordChar=""
    EndIf
EndSub
```

Paste 方法

将数据从系统剪贴板复制到 OLE 容器控件。

应用于

OLEContainer 控件，VBForm 对象

语法

object.**Paste**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

为了使用这个方法，先设置 OLETypeAllowed 属性，然后再检查 PasteOK 属性的值。只有 PasteOK 返回 True 值时，粘贴才能成功。如果执行 Paste 方法，OLEType 属性被置为 vbOLELinked(0)或 vbOLEEmbedded(1)。如果不执行 Paste 方法，OLEType 属性被置为

vbOLENone(3)。

使用方法可以支持菜单中的“编辑粘贴”命令。

如果 PasteOK 属性设置是 True，并且 VisualBasic 不能粘贴对象，那么 OLE 容器控件就将控件中已有的对象删除。

请参阅

OLEType 属性，OLETypeAllowed 属性，PasteOK 属性

PasteOK 属性

返回值，用于确定系统剪贴板的内容是否可以粘贴到 OLE 容器控件中。

应用于

OLEContainer 控件

语法

object.**PasteOK**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

当将这个属性设为 True，可将系统剪贴板的内容粘贴到 OLE 容器控件中。

使用 OLETypeAllowed 属性，指定想粘贴到 OLE 容器控件中的对象的类型（链接对象或内嵌对象）。一旦将对象成功地粘贴到 OLE 容器控件中，就可以检查 OLEType 属性设置来确定所创建的对象的类型。

如果应用程序支持“编辑”菜单的“粘贴”命令，可以使用属性。如果 PasteOK 是 False，就禁止使用菜单命令；否则，允许使用菜单命令。分别将 Enabled 属性设置为 True 或 False，来允许和禁止菜单命令的使用。

使用 Paste 方法，将对象粘贴到 OLE 容器控件中。

为了更方便灵活，当选取“编辑粘贴”命令时（设置 OLETypeAllowed=2，然后再使用 PasteSpecialDlg 方法），显示特殊粘贴对话框。当显示这个对话框时，根据在对话框中所做的选择，将对象粘贴到系统剪贴板上。

请参阅

OLEType 属性，OLETypeAllowed 属性，PasteSpecialDlg 方法，Paste 方法，Enabled 属性，Enabled 属性(ActiveX 控件)

示例

如果 PasteOK 属性的设置为 True，本例将一个对象粘贴到 OLE 容器控件。否则它将显示一个信息框。

```
PrivateSubmnuEditPaste_Click()
   '检查 PasteOK 的值。
   IfOle1.PasteOKThen
        Ole1.Paste              '为 True，使 Paste 命令有效。
   Else                         '否则，使 Paste 命令无效
mnuEditPaste.Enabled=False '菜单命令和
        MsgBox"Can'tpaste." '给出适当的信息。
```

```
    EndIf
EndSub
```

PasteSpecialDlg 方法

显示特殊粘贴对话框。

应用于

OLEContainer 控件

语法

object.**PasteSpecialDlg**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

在运行时，可以显示对话框，用于从系统剪贴板粘贴对象。对话框显示几个选项，包括粘贴链接对象或者是内嵌对象。

使用 OLETypeAllowed 属性，确定可用这个对话框创建的对象的类型（链接的、嵌入的或均可）。

如果 PasteOK 属性设置为 True，并且 VisualBasic 不能粘贴对象，那么 OLE 容器控件就将控件中已有的对象删除。

请参阅

OLETypeAllowed 属性

Path 属性

返回或设置当前路径。在设计时是不可用的。对于 App 对象，在运行时是只读的。

应用于

App 对象，DirListBox 控件，FileListBox 控件

语法

object.**Path**[=*pathname*]

Path 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
pathname	一个用来计算路径名的字符串表达式

说明

Path 属性的值是一个指示路径的字符串，例如 C:\Ob 或 C:\Windows\System。对于 DirListBox 或 FileListBox 控件，在运行时当控件被创建时，其缺省值是当前路径。对于 App 对象，当从开发环境运行该应用程序时 Path 指定.VBP 工程文件的路径，或者当把应用程序当作一个可执行文件运行时 Path 指定.exe 文件。当建立一个应用程序的文件浏览器和操作能力时使用这个属性。设置 Path 属性对控件产生的影响与 MS-DOS 的 chdir 命令相似——相关的路径可以带有或不带有驱动器的描述。只有指定带冒号(:)的驱动器才能在那个驱动器上选择当前目录。

使用下面的语法，Path 属性也可以设置限定的网络路径而不需要驱动器连接：

*servername**sharename**path*

前面的语法将 Drive 属性改变成了 0 长度的字符串("")。

Path 值的改变将产生以下影响：

对于一个 DirListBox 控件，将产生一个 Change 事件。

对于一个 FileListBox 控件，将产生一个 PathChange 事件.

注意对于 DirListBox，Path 的返回值与只返回选定内容的 List(ListIndex)是不同的。

请参阅

Change 事件，PathChange 事件，PatternChange 事件，List 属性，ListCount 属性，FileName 属性，Locked 属性，ReadOnly 属性，Pattern 属性，Change 事件(ActiveX 控件)

示例

该例子为选中的驱动器和目录显示一文件列表。要试用此例，先将以下代码粘贴到包含 DriveListBox、DirListBox 和 FileListBox 控件的窗体的声明部分。然后按下 F5 键。使用鼠标来改变驱动器或目录。

```
PrivateSubDrive1_Change()
   Dir1.Path=Drive1.Drive   '设置目录路径。
EndSub
PrivateSubDirl_change()
   File1.path=Dirl.path     '设置文件路径
```

```
EndSub
```

Path 属性（FileSystemObject 对象）

返回指定文件、文件夹或驱动器的路径。

应用于

Drive 对象，File 对象，Folder 对象

语法

object.**Path**

object 总是一个 File、Folder 或 Drive 对象。

说明

对于驱动器字母来说，不包括根驱动器。例如，C 驱动器的路径是 C:，而不是 C:\。

下面的代码用一个 File 对象举例说明了 Path 属性的用法：

```
SubShowFileAccessInfo(filespec)
Dimfs,d,f,s
Setfs=Create*object*("Scripting.FileSystem*object*")
Setf=fs.GetFile(filespec)
s=UCase(f.Path)&vbCrLf
s=s&"Created:"&f.DateCreated&vbCrLf
s=s&"LastAccessed:"&f.DateLastAccessed&vbCrLf
s=s&"LastModified:"&f.DateLastModified
MsgBoxs,0,"FileAccessInfo"
```

EndSub

请参阅

Attributes 属性，AvailableSpace 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，DriveLetter 属性，DriveType 属性，Files 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，ParentFolder 属性，RootFolder 属性，SerialNumber 属性，ShareName 属性，ShortName 属性，ShortPath 属性，Size 属性，SubFolders 属性，TotalSize 属性，Type 属性，VolumeName 属性

PathChange 事件

当路径被代码中 FileName 或 Path 属性的设置所改变时，此事件发生。

应用于

FileListBox 控件

语法

PrivateSub*object*_**PathChange**([*index***AsInteger**])

PathChange 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件

说明

可使用 PathChange 事件过程来响应 FileListBox 控件中路径的改变。当将包含新路径的字符串给 FileName 属性赋值时，FileListBox 控件就调用 PathChange 事件。

请参阅

PatternChange 事件，FileName 属性，Pattern 属性，Path 属性

示例

本例演示如何更新一个 Label 控件以显示一个 FileListBox 控件的当前路径。在 FileListBox 中双击一个目录名可在 FileListBox 中显示该目录文件的列表，同时也在 Label 控件中显示目录的全路径。要尝试这个例子，可将代码粘贴到一个包含一个 Label 控件、一个 DirListBox 控件和一个 FileListBox 控件的窗体的声明部分，然后按 F5 键。

双击目录改变路径。

```
PrivateSubFile1_PathChange()
   Label1.Caption="Path:"&Dir1.Path '在 Label 中显示路径。
EndSub

PrivateSubDir1_Change()
   File1.Path=Dir1.Path                    '设置文件路径。
EndSub

PrivateSubForm_Load()
```

```
    Label1.Caption="Path:"&Dir1.Path '在 Label 中显示路径。
EndSub
```

Pattern 属性

返回或设置一个值，该值指示在运行时显示在 FileListBox 控件中的文件名。

应用于

FileListBox 控件

语法

object.**Pattern**[=*value*]

Pattern 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定文件规格的字符串表达式，例如"*.*"或"*.FRM"。缺省值是"*.*"它返回所有文件的列表。除使用通配符外，还能够使用分号(;)分隔的多种模式。例如，"*.exe;*.bat"将返回所有可执行文件和所有 MS-DOS 批处理文件的列表

说明

在设计应用程序的文件浏览和操作功能中，Pattern 属性具有一个关键作用。Pattern 与其它文件控件属性结合起来使用，可提供对相似文件或文件组资源管理的方法。例如，在一个专门为了启动

别的程序而工作的应用程序中，只会指定.exe 文件显示在文件列表框中。其它关键性的文件控件属性包括 Drive，FileName 和 Path。

Pattern 属性的值的改变将产生一个 PatternChange 事件。

请参阅

PathChange 事件，PatternChange 事件，Archive，Hidden，Normal，System 属性，Drive 属性，FileName 属性，Path 属性

示例

本例使用在 FileListBox 控件中所选择的新模式来更新 TextBox 控件。这些控件的建立，使得当用户在 TextBox 中输入一个模式时，比如*.txt，它将被反映在 FileListBox 中，这很象基于 Windows 的应用程序中的典型“文件打开”对话框中所见到的交互作用。如果完整的路径，如 C:\Bin*.exe 被输入到 TextBox 控件中，那么该文本将自动地被 FileListBox 控件分析为路径和模式两个部分。要试用此例，将下面的代码粘贴到包含以下控件的窗体的声明部分：DirListBox、FileListBox、TextBox 和 CommandButton。按 F5 键并给 TextBox 键入一个有效的文件模式。

```
PrivateSubForm_Load()
   Command1.Default=True    '设置缺省属性。
EndSub

PrivateSubCommand1_Click()
```

```
    '文本被分解为路径和模式组件。
    File1.Filename=Text1.Text
    Dir1.Path=File1.Path      '设置目录路径。
EndSub

PrivateSubFile1_PatternChange()
    Text1.Text=File1.Pattern  '将文本设置为新模式。
EndSub

PrivateSubDir1_Change
    File1.Path=Dir1.Path      '设置文件列表框路径。
EndSub
```

PatternChange 事件

当文件的列表样式，如："*.*"，被代码中对 FileName 或 Path 属性的设置所改变时，此事件发生。

应用于

FileListBox 控件

语法

PrivateSub*object*_**PatternChange**([*index*]**AsInteger**])

PatternChange 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中控件

说明

可使用 PatternChange 事件过程来响应在 FileListBox 控件中样式的改变。当将包含新样式的字符串给 FileName 属性赋值时，FileListBox 控件将调用 PathChange 事件。

请参阅

PathChange 事件，FileName 属性，Pattern 属性，Path 属性

示例

本例更新一个带有与一个 TextBox 控件中输入的模式的匹配的各类文件的 FileListBox 控件。如果在 TextBox 中输入一个完整路径，比如 C:\Bin*.exe，文字被自动分解为路径和模式组件。要尝试这个例子，可将代码粘贴到一个包含一个 TextBox 控件、一个 Label 控件、一个 FileListBox 控件和一个 CommandButton 控件窗体的声明部分，然后按 F5 键并在 TextBox 中输入一个有效的文件模式。

```
PrivateSubForm_Load()
   Command1.Default=True          '设置缺省属性。
   Command1.Caption="OK"          '设置标题。
EndSub
```

```
PrivateSubCommand1_Click()              '单击 OK 按钮。
   '文本被分解为路径和模式组件。
   File1.FileName=Text1.Text
   Label1.Caption="Path:"&File1.Path
EndSub

PrivateSubFile1_PatternChange()
   Text1.Text=File1.Pattern             '将文字设置到新的模式。
EndSub
```

Persistable 属性

设置一个值，它决定一个对象是否能跨实例保存和恢复数据。仅能在设计时被设置。

应用于

Class

语法

object.**Persistable**[=*number*]

Persistable 属性语法有这些部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
number	一个整数，它指定存留行为，如在设置值中所描述的

设置值

number 的设置值如下：

常数	设置	描述
vbNotPersistable	0	(缺省)对象不能被存留
vbPersistable	1	对象可以被存留

说明

Persistable 属性仅对公共的和可创建的类是可用的。当 Persistable 设置为 vbPersistable 时，下列事件被添加到该类：InitProperties、ReadProperties 和 WriteProperties。PropertyChanged 方法也被添加到该类。

Picture 对象

将位图、图标、元文件、增强元文件、GIF 和 JPEG 等各类图像赋值给具有 Picture 属性的对象，可以用 Picture 对象来操纵它们。

语法

Picture

说明

经常使用能显示图形对象的 Picture 属性标识一个 Picture 对象（例如 Form 对象或 PictureBox 控件）。假设有一个名为 Picture1 的 PictureBox 控件，用 Set 语句可将一个 Picture 对象设置成与另一个相等，示例如下：

```
DimXAsPicture
```

```
SetX=LoadPicture("PARTY.BMP")
SetPicture1.Picture=X
```

利用一个 Picture 对象的数组便可以在内存中保存一系列图形，而不必利用包含多个 PictureBox 或 Image 控件的窗体。

不能这样来创建一个 Picture 对象：DimXAsNewPicture。如果需要创建一个 Picture 对象，就必须利用 StdPicture 对象，例如：

```
DimXAsNewStdPicture
```

属性

Handle 属性,hPal 属性,Height,Width 属性,Type 属性(Picture)

方法

Render 方法

请参阅

LoadPicture 函数，PictureBox 控件，Picture 属性，Picture 属性(ActiveX 控件)

Picture 属性

返回或设置控件中要显示的图片。对于 OLE 容器控件，在设计时不可用在运行时为只读。

应用于

Image 控件(DataReportDesigner)，PropertyPage 对象，User 控件对象 UserDocument 对象，CheckBox 控件，CommandButton 控件，PictureBox

语法

object.**Picture**[=*picture*]

Picture 属性有下列组成部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
picture	字符串表达式，指定一个包含图片的文件，“设置值”中有详细说明

设置值

picture **的设置值如下：**

设置值	描述
(None)	（缺省值）无图片
(Bitmap,icon,metafile,GIF,JPEG)	指定一个图片。设计时可以从属性窗口中加载图片。在运行时，也可以在位图，图标，或元文件上使用 Load-Picture 函数来设置该属性

说明

在设计时，利用“编辑”菜单中的“复制”、“剪切”和“粘贴”命令通过剪贴板来传递图片，运行时，可以使用剪贴板方法，诸如具有非文本剪贴板常数 vbCFBitmap、vbCFMetafile 和 vbCFDIB 的 GetData、SetData 和 GetFormat，它们列在对象浏览器中的 VisualBasic(VB)对象库中。

在设计时设置 Picture 属性，图片被保存起来并与窗体同时加载。

如果创建可执行文件，该文件中包含该图像。如果在运行时加载图片，该图片不和应用程序一起保存。用 SavePicture 语句可以从窗体或图片框的图片存储到文件中。

注意运行时，Picture 属性可以被设置为任何其它对象的 DragIcon、Icon、Image 或 Picture 属性，或者可将 LoadPicture 函数返回的图片分配给它。

注意 UnisysCorporation 有一项专利，该专利声称涉及到 GIF-LZW 压缩技术的某些方面，在该技术中使用了 PictureBox 和 Image 控件。MicrosoftCorporation 于 1996 年 9 月获得了对 UnisysLZW 专利的使用许可。然而，Microsoft 的许可证并不延伸到那些软件开发商或第三方，他们使用任何 Microsoft 工具包、语言开发或操作系统产品来在他们自己的产品中提供 GIF 读/写和/或任何其他 LZW 能力（例如，通过 DLL 和 API）。

如果您的商业应用程序使用了这些控件之一（并且因此使用了 LZW 技术），您可能会希望获得有关专利的独立的法律意见，详细信息请与 http://www.unisys.com/的 UnisysUSA 联系。

请参阅

Icon，SmallIcon 属性(ListItem 对象)，Panel 对象，Add 方法(Panels 集合)，Picture 对象，LoadPicture 函数，SavePicture 属性，Icon 属性，AutoRedraw 属性，Image 属性，Image 属性(ActiveX 控件)，PictureClip 控件，Picture 属性（ActiveX 控件）

示例

这个例子从VisualBasic的图标库中将图标加载到三个PictureBox控件中的两个之中。在单击窗体时，第三个PictureBox用来切换图标。可以使用任意两个图标。请将代码粘贴到包含三个小的PictureBox控件的窗体的声明部分（对于Picture3，设置Visible=False）。按F5键运行该程序，然后单击窗体。

```
PrivateSubForm_Load()
   '加载图标。
   Picture1.Picture=LoadPicture("ICONS\COMPUTER\TRASH02A.ICO")
   Picture2.Picture=LoadPicture("ICONS\COMPUTER\TRASH02B.ICO")
EndSub

PrivateSubForm_Click()
   '切换图标。
   Picture3.Picture=Picture1.Picture
   Picture1.Picture=Picture2.Picture
   Picture2.Picture=Picture3.Picture
   '清除第三张图片（如果图片不可见则不需要清除）。
   Picture3.Picture=LoadPicture()
EndSub
```

这个例子将剪贴板上的一张位图粘贴到PictureBox控件中。为了查找（以vbCF开始的）剪贴板格式常数的值，请参阅对象浏览器中的VisualBasic(VB)对象库。要尝试这个例子，请将代码粘

贴到包含一个 PictureBox 控件的窗体的声明部分。按 F5 键，然后在另一个应用程序中把一个图标复制到剪贴板上，切换到 VisualBasic 并单击窗体。

```
PrivateSubForm_Click()
   Picture1.Picture=Clipboard.GetData(vbCFDIB)
EndSub
```

PictureAlignment 属性

返回或设置一个值，决定图片在数据报表设计器的 Image 控件中显现的位置。

应用于

Image 控件(DataReportDesigner

语法

object.**PictureAlignment** [=*integer*]

PictureAlignment 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
integer	可选的。一个数值表达式，指定图片的位置，如在设置值中所示

设置值

对 *integer* 的设置如下：

常数	值	描述
rptPATopLeft	0	图片显现在左上部
rptPATop	1	图片显现在上部
rptPATopRight	2	图片显现在右上部
rptPARight	3	图片显现在右部
RptPABottomRight	4	图片显现在右下部
rptPABottom	5	图片显现在底部
rptPABottomLeft	6	图片显现在左下部
rptPALeft	7	图片显现在左部
rptPACenter	8	图片居中显现

PictureBox 控件

PictureBox 控件可以显示来自位图、图标或者元文件，以及来自增强的元文件、JPEG 或 GIF 文件的图形。如果控件不足以显示整幅图像，则裁剪图像以适应控件的大小。

语法

PictureBox

说明

也可以用 PictureBox 控件将 OptionButton 控件分组，并用该控件显示图形方法的输出和 Print 方法写入的文本。

为了使 PictureBox 控件能够自动调整大小以显示整幅图形，将它的 AutoSize 属性设置成 True。

可在代码中操作图形属性和方法，以创建动画或进行仿真。对运行时的打印操作，例如修改屏幕窗体格式以便打印，Graphics 属性和事件是很有用的。

在 DDE 对话中，PictureBox 控件还可以起接收端链接的作用。

PictureBox 控件和 Data 控件是唯一可以放置在 MDI 窗体内部区域的标准 VisualBasic 控件。可以使用该控件在内部区域的顶部或底部对控件分组，以创建工具栏或状态栏。

注意 UnisysCorporation 有一项专利,该专利声称涉及到 GIF-LZW 压缩技术的某些方面，在该技术中使用了 PictureBox 和 Image 控件。MicrosoftCorporation 于 1996 年 9 月获得了对 UnisysLZW 专利的使用许可。然而，Microsoft 的许可证并不延伸到那些软件开发商或第三方，他们使用任何 Microsoft 工具包、语言开发或操作系统产品来在他们自己的产品中提供 GIF 读/写和/或任何其他 LZW 能力（例如，通过 DLL 和 API）。

如果您的商业应用程序使用了这些控件之一（并且因此使用了 LZW 技术），您可能会希望获得有关专利的独立的法律意见，详细信息请与 http://www.unisys.com/的 UnisysUSA 联系。

属性

PictureBox，DataMember 属性，DataFormat 属性，RightToleft 属性，Negotiate 属性，OLEDragMode 属性，OLEDropMode 属性，BackColor，ForeColor 属性，BorderStyle 属性，DrawWidth 属

性，FontBold，FontItalic，FontStrikethru，FontUnderline 属性，FonName 属性，FontSize 属性，Height，Width 属性，Left，Top 属性，Picture 属性，TabIndex，属性，Tag 属性，Visible 属性，Align 属性，AutoRedraw 属性，AutoSize 属性，Clip 控件属性，CurrentX，CurrentY 属性，DragIcon 属性，DragMode 属性，DrawMode 属性，DrawStyle 属性，，FfillColor 属性，FillStyle 属性，hDC 属性，hWnd 属性，Image 属性，LinkItem 属性，LinkMode 属性，LinkTimeout 属性，LinkTopic 属性，MouseIcon 属性，MousePointer 属性，ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性，TabStop 属性，Appearance 属性，FontTransparent 属性，Enabled 属性，HelpContextID 属性 Index 属性(控件 Array)，Name 属性，Parent 属性，Font 属性，Container，属性，ToolTipText 属性，DataChanged 属性，DataField 属件，DataSource 属性，WhatsThisHelpID 属性

方法

Refresh 方法，SetFocus 方法，Circle 方法，LinePSet 方法，Cls 方法，Drag 方法，LinkExecute 方法，LinkPoke 方法，LinkRequest 方法，LinkSend 方法，Move 方法，PaintPicture 方法，Point 方法，Scale 方法，ScaleX，ScaleY 方法，TextHeight 方法，TextWidth 方法，ZOrder 方法，OLEDrag 方法，ShowWhatsThis 方法

事件

Change 事件,Click 事件,DragDrop 事件,DragOver 事件,GotFocus 事件，KeyDown，KeyUp 事件，KeyPress 事件，LinkClose 事件，LinkError 事件，LinkNotify 事件，LinkOpen 事件，LostFocus 事件，MouseDown，MouseUp 事件，MouseMove 事件，Paint 事件，Resize 事件，Validate 事件，DblClick 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件,OLEGiveFeedbacd 事件，OLESetData 事件，OLEStartDrag 事件

请参阅

Image 控件，MDIForm 对象，AutoSize 属性

Pmt 函数

返回一个 Double，指定根据定期定额支付且利率固定的年金支付额。

语法

Pmt(*rate*,*nper*,*pv*[,*fv*[,*type*]])

Pmt 函数有下列命名参数：

部分	描述
rate	必需的。Double 指定每一期的利率。例如，如果有一笔贷款年百分比率(APR)为百分之十且按月付款的汽车贷款，则每一期的利率为 0.1/12 或 0.0083
nper	必需的。Integer 指定一笔年金的付款总期数。例如，如果对一笔为期四年的汽车贷款选择按月付款，则贷款共有 4*12（或 48）个付款期
pv	必需的。Double 指定未来一系列付款或收款的现值。例如，当贷款买一辆汽车时，向贷方所借贷的金额为将来每月偿付给贷方款项的现值
fv	可选的。Variant 指定在付清贷款后所希望的未来值或现金结存。例如，贷款的未来值在贷款付清后为 0 美元。但是，如果想要在 18 年间存下 50，000 美元作为子女教育基金，那么 50，000 美元为未来值。如果省略的话，缺省值为 0
Type	可选的。Variant，指定贷款到期时间。如果贷款是在贷款周期结束时到期，请使用 0。如果贷款是在周期开始时到期，则请使用 1。如果省略的话，缺省值为 0

说明

年金是在一段时间内一系列固定现金支付，年金可以是贷款（如房屋抵押贷款），也可以是一笔投资（如按月储蓄计划）。

在支付期间必须用相同的单位计算 *rate* 和 *nper* 参数。例如，如果

rate 用月份计算，则 *nper* 也必须用月份计算。

对所有参数，用负数表示现金支出（如储蓄存款），而用正数表示现金收入（如红利支票）。

请参阅

DDB 函数，FV 函数，IPmt 函数，IRR 函数，MIRR 函数，NPer 函数，NPV 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

本示例使用 Pmt 函数计算以定期方式偿还贷款之每期月付款金额。计算时需给定每期利率（APR/12），付款总期数（TotPmts），贷款的现值或本金（PVal），贷款的未来值（FVal）及付款方式，以数值表示期初或期末付款（PayType）。

```
DimFmt,FVal,PVal,APR,TotPmts,PayType,Payment
ConstENDPERIOD=0,BEGINPERIOD=1 '付款方式。
Fmt="###,###,##0.00"                    '定义金额格式。
FVal=0                                  '对贷款而言通常为零。
PVal=InputBox("Howmuchdoyouwanttoborrow?")
APR=InputBox("Whatistheannualpercentagerateofyourloan?")
IfAPR>1ThenAPR=APR/100                  '确保格式正确。
TotPmts=InputBox("Howmanymonthlypaymentswillyoumake?")
PayType=MsgBox("Doyoumakepaymentsattheendofmonth?",vbYesNo)
IfPayType=vbNoThenPayType=BEGINPERIODElsePayType=ENDPERIOD
Payment=Pmt(APR/12,TotPmts,-PVal,FVal,PayType)
```

```
MsgBox"Yourpaymentwillbe"&Format(Payment,Fmt)&"permonth."
```

Point 方法

按照长整数，返回在 Form 或 PictureBox 上所指定磅的红-绿-蓝(RGB)颜色。不支持命名参数。

应用于

PropertyPage 对象，User 控件对象，UserDocument 对象，Form 对象.Forms 集合，PictureBox 控件

语法

object.**Point**(*x*,*y*)

Point 方法的语法包含下列部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 *object*，带有焦点的 Form 对象缺省为 *object*
x,*y*	必需的。均为单精度值，指示 Form 或 PictureBox 的 ScaleMode 属性中该点的水平（x-轴）和垂直（y-轴）坐标。必须用括号包括这些值

说明

如果由 *x* 和 *y* 坐标所引用的点位于 *object* 之外，Point 方法将返回 -1。

示例

本示例使用 Point 方法来确定一个窗体上的一个指定点的颜色。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimLeftColor,MidColor,Msg,RightColor '声明变量。
   AutoRedraw=-1                        '打开 AutoRedraw。
   Height=3*1440                        '将高度设置为 3 英寸。
   Width=5*1440                         '将宽度设置为 5 英寸。
   BackColor=QBColor(1)                 '将背景设置为蓝色。
   ForeColor=QBColor(4)                 '将前景设置为红色。
   Line(0,0)-(Width/3,Height),,BF       '红框。
   ForeColor=QBColor(15)                '将前景设置为白色。
   Line(Width/3,0)-((Width/3)*2,Height),,BF
   LeftColor=Point(0,0)                 '查找左框颜色，,
   MidColor=Point(Width/2,Height/2)     '中框，和
   RightColor=Point(Width,Height)       '右框。
   Msg="Thecolornumberfortheredboxontheleftsideof"
   Msg=Msg&"theformis"&LeftColor&".The"
   Msg=Msg&"colorofthewhiteboxinthecenteris"
   Msg=Msg&MidColor&".Thecoloroftheblue"
   Msg=Msg&"boxontherightis"&RightColor&"."
   MsgBoxMsg                            '显示信息。
```

EndSub

PopupMenu 方法

用以在 MDIForm 或 Form 对象上的当前鼠标位置或指定的坐标位置显示弹出式菜单。不支持命名参数。

应用于

PropertyPage 对象，User 控件对象，UserDocument 对象，Form 对象，Forms 集合，MDIForm 对象

语法

object.**PopupMenu***menuname*,*flags*,*x*,*y*,*boldcommand*

PopupMenu 方法的语法包含下列部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 *object*，则带有焦点的 Form 对象缺省为 *object*
Menuname	必需的。要显示的弹出式菜单名。指定的菜单必须含有至少一个子菜单
flags	可选的。一个数值或常数，按照下列设置中的描述，用以指定弹出式菜单的位置和行为
x	可选的。指定显示弹出式菜单的 x 坐标。如果该参数省略，则使用鼠标的坐标
y	可选的。指定显示弹出式菜单的 y 坐标。如果该参数省略，则使用鼠标的坐标
boldcommand	可选的。指定弹出式菜单中的菜单控件的名字，用以显示其黑体正文标题。如果该参数省略，则弹出式菜单中没有以黑体字出现的控件

设置值

用于 flag 的设置值有：

常数位置	值	描述
vbPopupMenuLeftAlign	0	（缺省值）。弹出式菜单的左边定位于 x
vbPopupMenuCenterAlign	4	弹出式菜单的于 x 居中位
vbPopupMenuRightAlign	8	弹出式菜单的右边定位于 x

常数行为	值	描述
vbPopupMenuLeftButton	0	（缺省值）。仅当使用鼠标左按钮时，弹出式菜单中的项目才响应鼠标单击
vbPopupMenuRightButton	2	不论使用鼠标右按钮还是左按钮，弹出式菜单中的项目都响应鼠标单击

说明

上述常数在对象浏览器中的 VisualBasic(VB)对象库里列出。

可使用 ScaleMode 属性指定 x 和 y 坐标的度量单位。x 和 y 坐标定义弹出式菜单相对于指定窗体显示的位置。如果没有包括 x 和 y 坐标，则弹出式菜单就显示在鼠标指针当前的位置。

在显示弹出式菜单时，调用 PopupMenu 方法后面的代码直到用户或者从菜单中选择了命令（这时，该命令的 Click 事件的代码比 PopupMenu 语句后面的代码先执行）或者取消该菜单时才能执行。此外，每次只能显示一个弹出式菜单，因此，如果已经显示了一

个弹出式菜单或打开了一个下拉式菜单时，该方法的其它调用将被忽略。

请参阅

MouseDown，MouseUp 事件，Visible 属性，ScaleMode 属性

示例

本示例显示当用户在窗体上单击鼠标右键时在光标处出现的一个弹出式菜单。要检验此示例，创建一个窗体，它包含一个称为 mnuFile（mnuFile 必需至少有一个 submenu）的 Menu 控件。将本例代码粘贴到一个窗体的声明部分，然后按 F5 键。

```
PrivateSubForm_MouseDown(ButtonAsInteger,ShiftAsInteger,XAsSingle,YAsSingle)
    IfButton=2Then
        PopupMenumnuFile
    EndIf
EndSub
```

Port 属性

返回端口的名字，文档通过此端口发送给打印机。

应用于

Printer 对象，Printers 集合

语法

object.**Port**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

操作系统决定端口的名字，例如 LPT1: 或 LPT2:。

注意 Printer 对象此属性的效果依赖于打印机厂家提供的驱动程序。某些属性设置值可能不起作用，或者一些不同的属性设置值可能有相同的效果。可接受范围之外的设置值可能产生也可能不产生错误。详细信息，请参阅厂家具体驱动程序的文档。

请参阅

Printer 对象，Printers 集合

示例

本例对 Printers 集合中每个 Printer 对象进行检查，以找到一个与特定端口相连接的打印机，并使其成为缺省打印机。

```
DimPAs object
ForEachPInPrinters
   IfP.Port="LPT2:"OrP.DeviceNameLike"*LaserJet*"Then
      SetPrinter=P
      ExitFor
   EndIf
NextP
```

PPmt 函数

返回一个 Double，指定在定期定额支付且利率固定的年金的指定期间内的本金偿付额。

语法

PPmt(*rate*,*per*,*nper*,*pv*[,*fv*[,*type*]])

PPmt 函数有下列命名参数：

部分	描述
rate	必需的。Double 指定每一期的利率。例如，如果有一笔贷款年百分比率(APR)为百分之十且按月付款的汽车贷款，则每一期的利率为 0.1/12 或 0.0083
per	必需的。Integer 指定在 ***nper*** 间范围 1 中的付款周期
nper	必需的。Integer 指定一笔年金的付款总期数。例如，如果对一笔为期四年的汽车贷款选择按月付款，则贷款共有 4*12（或 48）个付款期
pv	必需的。Double 指定未来一系列付款或收款的现值。例如，当贷款买一辆汽车时，向贷方所借贷的金额为将来每月偿付给贷方款项的现值
fv	可选的。Variant 指定在付清贷款后所希望的未来值或现金结存值。例如，贷款的未来值在贷款付清后为 0 美元。但是，如果想要在 18 年间存下 50，000 美元作为子女教育基金，那么 50，000 美元为未来值。如果省略的话，缺省值为 0
type	可选的。Variant 指定贷款到期时间。如果贷款是在贷款周期结束时到期，则请使用 0。如果贷款是在周期开始时到期，则请使用 1。如果省略的话，缺省值为 0

说明

年金是在一段时间内一系列固定现金支付。年金可以是贷款（如

房屋抵押贷款），也可以是一笔投资（如按月储蓄计划）。

在支付期间必须用相同的单位计算 *rate* 和 *nper* 参数。例如，如果 *rate* 用月份计算，则 *nper* 也必须用月份计算。

对所有参数，用负数表示现金支出（如储蓄存款），而用正数表示现金收入（如红利支票）。

请参阅

DDB 函数，FV 函数，IPmt 函数，IRR 函数，MIRR 函数，NPer 函数，NPV 函数，Pmt 函数，PV 函数，Rate 函数，SLN 函数，SYD 函数

示例

本示例使用 PPmt 函数计算以定期定额且固定利率方式偿还贷款时，每期付款中偿还本金之金额。计算时给定每期利率（APR/12），本金分期偿还周期（Period），付款总期数（TotPmts），贷款的现值或本金（PVal），贷款的未来值（FVal）及付款方式，以数值表示期初或期末付款（PayType）。

```
DimNL,TB,Fmt,FVal,PVal,APR,TotPmts,PayType,Payment,—
Msg,MakeChart,Period,P,I
ConstENDPERIOD=0,BEGINPERIOD=1 '付款方式。
NL=Chr(13)&Chr(10)                    '定义换行字符。
TB=Chr(9)                             '定义 tab。
Fmt="###,###,##0.00"                      '定义金额格式。
FVal=0                                '对贷款而言通常为零。
PVal=InputBox("Howmuchdoyouwanttoborrow?")
```

```
APR=InputBox("Whatistheannualpercentagerateofyourloan?")
IfAPR>1ThenAPR=APR/100              '确保格式正确。
TotPmts=InputBox("Howmanymonthlypaymentsdoyouhavetomake?")
PayType=MsgBox("Doyoumakepaymentsattheendofmonth?",vbYesNo)
IfPayType=vbNoThenPayType=BEGINPERIODElsePayType=ENDPERIOD
Payment=Abs(-Pmt(APR/12,TotPmts,PVal,FVal,PayType))
Msg="Yourmonthlypaymentis"&Format(Payment,Fmt)&"."
Msg=Msg&"Wouldyoulikeabreakdownofyourprincipaland"
Msg=Msg&"interestperperiod?"
MakeChart=MsgBox(Msg,vbYesNo)        '询问是否要显示。
IfMakeChart<>vbNoThen
   IfTotPmts>12ThenMsgBox"Onlyfirstyearwillbeshown.”
   Msg="MonthPaymentPrincipalInterest"&NL
   ForPeriod=1ToTotPmts
       IfPeriod>12ThenExitFor   'Showonlyfirst12。
       P=PPmt(APR/12,Period,TotPmts,-PVal,FVal,PayType)
       P=(Int((P+.005)*100)/100)          '四舍五入计算本金。
       I=Payment-P
       I=(Int((I+.005)*100)/100)          '四舍五入计算利息。
       Msg=Msg&Period&TB&Format(Payment,Fmt)
       Msg=Msg&TB&Format(P,Fmt)&TB&Format(I,Fmt)&NL
   NextPeriod
```

```
    MsgBoxMsg                                    '显示分期偿还表。
EndIf
```

PrevInstance 属性

返回一个值，该值指示是否已经有前一个应用程序实例在运行。

应用于

App 对象

语法

object.**PrevInstance**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

能够在 Load 事件过程中，使用此属性来指示是否已经运行了应用程序的一个实例。根据应用程序的要求，在 MicrosoftWindows 操作环境中可能每次只想运行一个实例。

注意由于运行 WindowsNT 的计算机可以支持多个平台，如果使用旨在同分布式 COM 一同使用的部件，则会导致下列情况：

用户平台上的客户程序请求部件提供一个对象，因为部件物理地位于同一台机器上，部件是在用户平台上启动的。

相应地，在另一台使用分布式 COM 的计算机上的客户程序请求部件提供一个对象。第二个部件的实例在系统平台上启动。

现在有两个部件实例运行在不同平台上的同一台 NT 计算机上。

这种情况并不是问题，除非部件的作者将对 App.PrevInstance 的测试放入部件启动代码以防止部件的多个复本运行在同一台计算机上。在这种情况下，远程部件创建将会失败。

Print#语句

将格式化显示的数据写入顺序文件中。

语法

Print#*filenumber*,[*outputlist*]

Print#语句的语法具有以下几个部分：

部分	描述
filenumber	必需的。任何有效的文件号
outputlist	可选的。表达式或是要打印的表达式列表

设置

outputlist 参数的设置如下：

[{Spc(*n*)|Tab[(*n*)]}][*expression*][*charpos*]

设置	描述
Spc(n)	用来在输出数据中插入空白字符，而 *n* 指的是要插入的空白字符数
Tab(n)	用来将插入点定位在某一绝对列号上，这里，*n* 是列号。使用无参数的 Tab 将插入点定位在下一个打印区的起始位置
expression	要打印的数值表达式或字符串表达式
charpos	指定下一个字符的插入点。使用分号将插入点定位在上一个显示字符之后。用 Tab(*n*)将插入点定位在某一绝对的列号上，用无参数的 Tab 将插入点定位在下一个打印区的起始处。如果省略 *charpos*，则在下一行打印下一个字符

说明

通常用 LineInput#或 Input 读出 Print#在文件中写入的数据。

如果省略参数 *outputlist*，而且，*filenumber* 之后只含有一个列表分隔符，则将一空白行打印到文件中。多个表达式之间可用一个空白或一个分号隔开。空白与分号等效。

对于 Boolean 类型的数据，打印的是 True 或 False。无论在什么国别，都不将 True 和 False 这两个关键字翻译出来。

使用操作系统所能辨认的标准短日期格式可将 Date 类型的数据写入文件中。在未指定日期或时间部件或这些部件的设置为零时，只将指定的部分写入文件中。

如果 *outputlist* 的数据是 Empty，则不将任何数据写入文件。但是，如果 *outputlist* 的数据是 Null，则将 Null 写入文件。

对于 Error 类型的数据而言，输出的数据看起来与 Errorerrorcode 一样。而且无论在什么地区，都不将 Error 关键字翻译出来。

用 Print#写入文件的所有数据都是国际通用的；也就是说，可以正确利用十进制分隔符将这些数据格式化。

因为 Print#将数据的图像写入文件，所以必须将各项数据分隔开来，以便正确打印。如果使用无参数的 Tab 将打印位置移动到下一个打印区，则 Print#也会将打印字段之间的空白写入文件中。

注意如果今后想用 Input#语句读出文件的数据，就要用 Write#语句而不用 Print#语句将数据写入文件。因为在使用 Write#时，将数据域分界就可确保每个数据域的完整性，因此可用 Input#再将数据读出来。使用 Write#还能确保任何地区的数据都被正确读出。

请参阅

Open 语句，Spc 函数，Tab 函数，Write#语句，Print 方法

示例

本示例使用 Print#语句将数据写入一个文件。

```
Open"TESTFILE"ForOutputAs#1          '打开输出文件。
Print#1,"Thisisatest"                '将文本数据写入文件。
Print#1,                             '将空白行写入文件。
Print#1,"Zone1";Tab;"Zone2"   '数据写入两个区（printzones）。
Print#1,"Hello";"";"World"           '以空格隔开两个字符串。
```

```
Print#1,Spc(5);"5leadingspaces"    '在字符串之前写入五个空格。
Print#1,Tab(10);"Hello"            '将数据写在第十列。

'赋值 Boolean、Date、Null 及 Error 等。
DimMyBool,MyDate,MyNull,MyError
MyBool=False:MyDate=#February12,1969#:MyNull=Null
MyError=CVErr(32767)
'True、False、Null 及 Error 会根据系统的地区设置自动转换格式。
'日期将以标准的短式日期的格式显示。
Print#1,MyBool;"isaBooleanvalue"
Print#1,MyDate;"isadate"
Print#1,MyNull;"isanullvalue"
Print#1,MyError;"isanerrorvalue"
Close#1                            '关闭文件。
```

Print 方法

在 Immediate 窗口中显示文本。

应用于

UserControl 对象，Debug 对象，Form 对象，Forms 集合

语法

object.**Print**[*outputlist*]

Print 方法的语法具有下列对象限定符和部分：

部分	描述
object	必需的。对象表达式，其值为“应用于”列表中的对象
outputlist	可选的。要打印的表达式或表达式的列表。如果省略，则打印一空白行

outputlist 参数具有以下语法和部分：

{Spc(*n*)|Tab(*n*)}*expressioncharpos*

部分	描述
Spc(*n*)	可选的。用来在输出中插入空白字符，这里，*n* 为要插入的空白字符数
Tab(*n*)	可选的。用来将插入点定位在绝对列号上，这里，*n* 为列号。使用无参数的 Tab(*n*)将插入点定位在下一个打印区的起始位置
expressio n	可选。要打印的数值表达式或字符串表达式
charpos	可选。指定下个字符的插入点。使用分号(;)直接将插入点定位在上一个被显示的字符之后。使用 Tab(*n*)将插入点定位在绝对列号上。使用无参数的 Tab 将插入点定位在下一个打印区的起始位置。如果省略 *charpos*，则在下一行打印下一字符

说明

可以用空白或分号来分隔多个表达式。

对系统指定的国别设置，用小数点分隔符将所有打印到 Immediate 视窗的数据正确格式化。关键字要用适用于主应用程序的语言输出。

对于 Boolean 数据，或者打印 True 或者打印 False。根据主机应用程序的地区设置来翻译 True 和 False 关键字。

使用系统能识别的标准短日期格式书写 Date 数据。当日期或时间部件丢失或为零时，只书写已提供的部件。

如果 *outputlist* 数据是 Empty，则无内容可写。但是，如果 *outputlist* 数据是 Null，则输出 Null。在输出 Null 关键字时，要把关键字正确翻译出来。

要把错误数据作为 Errorerrorcode 输出。在输出 Error 关键字时，要把关键字正确翻译出来。

如果在具有缺省显示空间的模块外使用此方法，则需要 *object*。例如，如果没有指定 *object* 就在标准模块上调用此方法，则将导致错误发生，但是，如果在窗体模块上进行调用，则会在窗体上显示“*outputlist*”。

注意因为 Print 方法是按照字符比例进行打印，所以字符数与字符所占据的宽度固定的列的数目无关。例如，像“W”这样的宽字母占据的宽度超过一固定列宽，而像"i"这样的窄字母占据的宽度则较小。考虑到要使用比平均字符更宽的空间，表列一定要留有足够余地。另外，也可以使用固定间距的字体（像 Courier 字体）来确保每一字符均只占一列。

请参阅

Print#语句，Spc 函数，Tab 函数，Assert 方法

示例

本示例使用 Print 方法在“调试”窗口的“立即”面板中显示变量 MyVar 的值。请注意，Print 方法只能用于可显示文本的对象。

```
DimMyVar
MyVar="ComeseemeintheImmediatepane."
Debug.PrintMyVar
```

Printer 对象和 Printers 集合

使用 Printer 对象可以实现与系统打印机的通讯（最初是缺省系统打印机）。

用 Printers 集合可获取有关系统上所有可用打印机的信息。

语法

Printer

Printers(*index*)

index 所在处表示从 0 到 Printers.Count-1 之间的整数。

说明

用图形方法在 Printer 对象上绘制文本和图形。一旦 Printer 对象中含有将要打印的输出信息，可用 EndDoc 方法直接将输出信息送到应用程序的缺省打印机上。

如果要打印这些信息，应该检查和可能还要修正窗体的布局。例如，如果用 PrintForm 方法打印窗体，则到页底图形图像可能被剪

切，而文本则被移到下一页。

Printers 集合可用来查询可用的打印机，这样就可以为应用程序指定一台缺省打印机。例如也许要找出哪些可用打印机用了指定的打印驱动程序。下面的代码查找所有的可用打印机，定位在第一个将页码方向设置为纵向的打印机，然后将其设置为 Portrait。

```
DimXAsPrinter
ForEachXInPrinters
   IfX.Orientation=vbPRORPortraitThen
      '设定为系统缺省打印机。
      SetPrinter=X
      '终止查找打印机。
      ExitFor
   EndIf
Next
```

用 Set 语句指定 Printers 集合中的某一打印机为缺省打印机。前面的示例指定由对象变量 X 标识的打印机为应用程序的缺省打印机。

注意如果用 Printers 集合来确定某一特定打印机，如 Printers(3)，则只能访问只读属性。如果想访问个别打印机的可读写属性，那么首先要使那个打印机成为应用程序的缺省打印机。

属性

RightToLeft 属性, DrawWidth 属性, FontBold, FontItalic, FontStrikethru, FontUnderline 属性, FontName 属性, FontSize

属性，Height，Width 属性，ColorMode 属性，Copies 属性，CurrentX，CurrentY 属性，DeviceName 属性，DrawMode 属性，DrawStyle 属性，DriverName 属性，Duplex 属性，FillColor 属性，FillStyle 属性，FontCount 属性，Fonts 属性，hDC 属性，Orientation 属性，Page 属性，PaperBin 属性，PaperSize 属性，Port 属性，PrintQuality 属性，ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性，TwipsPerPixelX，TwipsPerPixelY 属性，FontTransparent 属性，TrackDefault 属性，Count 属性(VB 集合)，Zoom 属性，Font 属性

方法

Circle 方法，Line 方法，PSet 方法，EndDoc 方法，KillDoc 方法，NewPage 方法，PaintPicture 方法，Scale 方法，ScaleX，ScaleYMetnods，TextHeight 方法，TextWidth 方法

Printer 属性

返回一个 Printer 对象，该对象允许跟一个系统打印机（最初是缺省的系统打印机）进行通信。

应用于

Global 对象

语法

Printer

说明

可用图形方法在 Printer 对象上画出文本和图形。一旦 Printer 对象包含要打印的输出，就可用 EndDoc 方法将输出直接发送到应用程序的缺省打印机上。

如果要打印窗体就应检查窗体的布局，可能还要修改窗体的布局。例如，如果用 PrintForm 方法打印一个窗体，则在页底以及跨页的文本中可能会剪切图形图像。

请参阅

Global 对象，Printer 对象，Printers 集合

PrinterDefault 属性

返回或设置一个选项，用确定在“打印”对话框中的选择是否用于改变系统缺省的打印机设置。

应用于

CommonDialog 控件

语法

object.**PrinterDefault**[=*value*]

PrinterDefault 属性语法有下列部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	布尔表达式，如“设置值”中所描述，它指定用户的选择是否用于改变系统缺省的打印机设置

设置值

value 的设置值是：

设置值	描述
True	在“打印”对话框的安装部分所作的任何选择（打印机选择，定向，等等），都将用于改变 WIN.INI 文件中打印机的设置（在 WindowsNT 操作系统中，这个信息存储在注册中）
False	其选择不能用于改变系统缺省打印机的设置

说明

当 PrinterDefault 为 True 时，可以编写代码直接向 VisualBasic Printer 对象打印。否则，必须通过图形设备接口(GDI)调用，往由控件的 hDC 属性指定的打印机上进行打印，。

注意如果前面已有 Printer 对象打印的，则应保证使用 Printer.EndDoc 使该打印作业终止。释放与打印机相关的 hDC。将取得缺省打印机的新的 hDC，用于下一次打印的 Printer 对象。如果不这样作的话，很可能出现选择一个新打印机而 Printer 对象还包含一个老打印机的句柄。

数据类型

Boolean

请参阅

Printer 对象，Printers 集合，hDC 属性

Printers 属性

返回一个 Printers 集合，该集合允许收集系统上所有可用打印机的信息。

应用于

Global 对象

语法

Printers(*index*)

index 所在处代表一个整数，其范围从 0 到 Printers.Count-1。

说明

Printers 集合允许对可用打印机进行查询，这样就可以为自己的应用程序指定缺省打印机。例如，您可能想查出哪种可用打印机使用特定的打印机驱动程序。下列代码对所有可用打印机进行搜索，对第一个打印机定位，并将其走纸方向设置为纵向，然后将打印机设置为缺省打印机：

```
DimXAsPrinter
ForEachXInPrinters
   IfX.Orientation=vbPRORPortraitThen
      '将打印机设置为系统缺省的打印机。
      SetPrinter=X
      '停止寻找打印机。
      ExitFor
   EndIf
```

Next

可以用 Set 语句将 Printers 集合中的一种打印机设置为缺省打印机。上例中将对象变量 X 所标识的打印机指定为应用程序的缺省打印机。

注意如果象在 Printers(3)中那样，用 Printers 集合指定特定打印机，那么只能以只读方式来访问属性。为了读写单个打印机的属性，必须首先将打印机设置为应用程序的缺省打印机。

请参阅

Global 对象，Printer 对象，Printers 集合

PrintForm 方法

用以将 Form 对象的图像逐位发送给打印机。

应用于

UserDocument 对象，Form 对象，Forms 集合

语法

object.**PrintForm**

object 所在处代表一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 *object*，则带有焦点的 Form 对象缺省为 *object*。

说明

PrintForm 将打印 Form 对象的全部可见对象和位图。在绘制图形时，如果 AutoRedraw 属性为 True，则在运行时 PrintForm 将打印 Form

对象或 PictureBox 控件上的图形。

PrintForm 所使用的打印机是由操作系统的控制面板中的设置来决定。

请参阅

Printer 对象，Printers 集合，EndDoc 方法，AutoRedraw 属性 TrackDefault 属性

示例

本示例使用 PrintForm 方法打印当前窗体。要检验此示例，可将本例代码粘贴到一个窗体的声明部分。在窗体上放置任何在打印出来的窗体上想要看到的控件，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimMsg                    '声明变量。
   OnErrorGoToErrorHandler  '设置错误处理程序。
   PrintForm                 '打印窗体。
   ExitSub
ErrorHandler:
   Msg="Theformcan'tbeprinted."
   MsgBoxMsg                 '显示信息。
   ResumeNext
EndSub
```

PrintQuality 属性

返回或设置一个值，该值指示打印机的分辨率。在设计时是不可用。

应用于

Printer 对象，Printers 集合

语法

object.**PrintQuality**[=*value*]

PrintQuality 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定打印机分辨率的值或常数，“设置值”中有详细描述

设置值

value 的设置值是：

常数	值	描述
vbPRPQDraft	-1	草稿分辨率
vbPRPQLow	-2	低分辨率
vbPRPQMedium	-3	中等分辨率
vbPRPQHigh	-4	高分辨率

除了预定义的负值之外，也可将 *value* 设置为一个表示每英寸点数的(dpi)正值，例如 300。

说明

这些常数在的对象浏览器的 VisualBasic(VB)对象库中被列出。缺省值依赖于打印机驱动程序和打印机的当前设置值。这些设置在打印机和打印机驱动程序中的影响是不同的。在某些打印机上，一些或所有的设置值可能产生相同的结果。

注意 Printer 对象此属性的效果依赖于打印机厂家提供的驱动程序。某些属性设置值可能不起作用，或者一些不同的属性设置值可能有相同的效果。可接受范围之外的设置值可能产生也可能不产生错误。详细信息，请参阅厂家具体驱动程序的文档。

请参阅

Printer 对象，Printers 集合

PrintReport 方法

运行时打印用数据报表设计器创建的数据报表。

应用于

DataReport 对象

语法

object.**PrintReport**(*ShowDialog*,*Range*,*PageFrom*,*PageTo*)

PrintReport 方法的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
ShowDialog	可选的。一个布尔表达式，决定是否显示“打印”对话框
Range	可选的。设定一个整数，决定是否包含报表中所有页面，或者仅包含一定范围的页面，如在设置值中所示
PageFrom	可选的。一个整数，设定了打印开始的页面
PageTo	可选的。一个整数，设定了打印终止的页面

设置值

常数	值	描述
rptRangeAllPages	0	（缺省的）所有页面都将被打印
rptRangeFromTo	1	只有指定范围的页面将被导出

返回类型

Long

说明

如果未给该方法提供参数，将显示一个对话框，提示用户提供相应的信息。

PrintReport 方法执行一个异步操作。PrintReport 方法返回“cookie”的标识符，来标识异步的操作。

请参阅

ProcessingTimeout 事件

示例

第一个示例显示“打印”对话框，允许用户指定文件名和页面范围。第二个示例打印报表，而不显示对话框。

```
PrivateSubDisplayPrintDialog()
   DataReport1.PrintReportTrue
EndSub

PrivateSubPrintWithoutDialog()
   '打印报表中的所有页面。
   DataReport1.PrintReportFalse,rptRangeAllPages
EndSub
```

Private 语句

在模块级别中使用，用于声明私有变量及分配存储空间。

语法

Private[**WithEvents**]*varname*[([*subscripts*])][*As*[*New*]*type*][,[*WithEvents*]*varname*[([*subscripts*])][*As*[*New*]*type*]]...

Private 语句的语法包含下面部分：

部分	描述
withEvents	可选的。关键字，说明*varname* 是用来响应由 ActiveX 对象所触发的事件的对象变量。只有在类模块中才是合法的。使用 WithEvents，可以定义任意个所需的单变量，但不能用 WithEvents 创建数组。New 和 WithEvents 不能一起使用
varname	必需的。变量的名称；遵循标准的变量命名约定
subscripts	可选的。数组变量的维数；最多可以定义 60 维的多维数组 *subscripts* 参数的使用语法如下： [*lower*To]*upper*[,[*lower*To]*upper*]... 如果不显式指定 *lower*，则数组的下界由 OptionBase 语句控制。如果没有 OptionBase 语句则下界为 0
New	可选的。使其可以隐式地创建对象的关键字。如果使用 New 声明对象变量，则在第一次引用该变量时创建该对象的新实例，因此不必使用 Set 语句来对该对象引用赋值。New 关键字不能声明任何内部数据类型的变量以及从属对象的实例，也不能与 WithEvents 一起使用

type 可选的。变量的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（对可变长的字符串）、String*length（对定长的字符串）、*objec*、Variant、用户定义类型或对象类型。所声明的每个变量都要一个单独的 Astype 子句

说明

Private 变量只能在包含其声明的模块中使用。

可以使用 Private 语句声明变量的数据类型。例如，下面的语句声明了一个 Integer 类型的变量。

```
PrivateNumberOfEmployeesAsInteger
```

也可以使用 Private 语句来声明变量的对象类型。下面的语句为工作表的新实例声明了一个变量。

```
PrivateXAsNewWorksheet
```

如果在定义对象变量时没有使用 New 关键字，则在使用该变量之前，必须使用 Set 语句将一个已有的对象赋给该引用对象的变量。在赋值之前，所声明的这个对象变量有一个特定值 Nothing，这个值表示该变量没有指向任何对象的实例。

如果不指定数据类型或对象类型，且在模块中没有使用 Deftype 语句，则按缺省情况该变量为 Variant 类型。

可以用带空圆括号的 Private 语句来声明动态数组，然后可以在过程内用 ReDim 语句来定义该数组的维数和元素。如果试图在 Private，Public 或 Dim 语句中重新定义一个已显式定义了大小的数

组的维数，就会发生错误。

当初始化变量时，数值变量被初始化为 0，变长的字符串被初始化为一个零长度的字符串(""),而定长的字符串则用 0 填充。Variant 变量被初始化为 Empty。用户自定义类型的变量的每个元素作为各自独立的变量进行初始化。

注意当在过程中使用 Private 语句时，通常将 Private 语句放在过程的开始。

请参阅

Array 函数，Const 语句，Dim 语句，Function 语句，OptionBase 语句，OptionPrivate 语句，PropertyGet 语句，PropertyLet 语句，PropertySet 语句，Public 语句，ReDim 语句，Set 语句，Static 语句，Sub，Type 语句

示例

该示例演示在模块级使用 Private 语句来声明私有变量；即这些变量只能在包含其声明的模块中使用。

```
PrivateNumberAsInteger              '私有的整数变量。
PrivateNameArray(1To5)AsString '私有的数组变量。
'多个声明，两个变体型，以及一个整型，都是私有的。
PrivateMyVar,YourVar,ThisVarAsInteger
```

ProcBodyLine 属性

返回过程的第一行。

应用于

CodeModule 对象

语法

object.**ProcBodyLine**(*procname*,*prockind*)**AsLong**

ProcBodyLine 语法有以下部分：

部分	描述
object	必需的。对象表达式，其值是“应用于”列表中的对象
procname	必需的。包含过程名的字符串。
prockind	必需的。指定要定位的过程种类。属性过程在模块中可以有多种表示，必须指定要定位的过程种类。除了使用 vbext_pk_Proc 的属性过程(即，Sub 和 Function 过程)之外的所有过程

对 *prockind* 参数可以使用如下的常数之一：

常数	描述
vbext_pk_Get	指定一个返回属性值的过程
vbext_pk_Let	指定一个赋值给属性的过程
vbext_pk_Set	指定一个给对象设置引用的过程
vbext_pk_Proc	指定所有过程除了属性过程

说明

Sub、Function 或 Property 出现在一个过程的第一行。

请参阅

DeleteLines 方法，Find 方法(VBA 处接程序对象模型)，GetSelection 方法，InsertLines 方法，CodePsne 对象，ProcCountLines 属性，ProcOfLine 属性，ProcStarLine 属性

示例

以下示例使用 ProcBodyLine 属性，返回指定过程 SetupTabs 中代码第一行的行号到特定的代码窗格。

```
Debug.PrintApplication.VBE.CodePanes(3).CodeModule.ProcBodyLine("
SetupTabs", vbext_pk_Proc)
```

ProcCountLines 属性

返回特定过程的行数。

应用于

CodeModule 对象

语法

object.**ProcCountLines**(*procname*,*prockind*)**AsLong**

ProcCountLines 语法有三部分

部分	描述
object	必需的。对象表达式，其值是“应用于”列表中的对象
procname	必需的。包含过程名的字符串。
prockind	必需的。指定要定位的过程种类。属性过程在模块中可以有多种表示，必须指定要定位的过程种类。所有过程除了属性过程(即，Sub 和 Function 过程)用 vbext_pk_Proc

对 *prockind* 参数可以使用如下的常数之一：

常数	描述
vbext_pk_Get	指定一个返回属性值的过程
vbext_pk_Let	指定一个赋值给属性的过程
vbext_pk_Set	指定一个给对象设置引用的过程
vbext_pk_Proc	指定所有过程除了属性过程

说明

ProcCountLines 属性返回在过程声明之前的所有空行及注释行的计数，并且，如果该过程是一段代码模块的最后一个，那么此过程之后的所有空行也计入。

请参阅

DeleteLines 方法，Find 方法(VBA 处接程序对象模型)，GetSelection 方法，InsertLines 方法，CodePane 对象，ProcBodyLine 属性，ProcOfLine 属性 ProcStartLine 属性

示例

以下示例使用 ProcCountLines 属性，返回指定过程 SetupTabs 中代码的行数到特定的代码窗格。

```
Debug.PrintApplication.VBE.CodePanes(3).CodeModule.ProcCountLines("SetupTabs",vbext_pk_Proc)
```

ProcessingTimeout 事件

在预设的时间间隔发生，允许用户取消一个异步操作。

应用于

DataReport 对象

语法

Private Sub *object*_**ProcessingTimeout**(*Seconds* **As Long**, *Cancel* **As Boolean**, *JobType* **As AsyncTypeConstants**, *Cookie* **As Long**)

处理超时事件的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
Seconds	返回开始异步操作后经过的秒数
Cancel	设定一个值，决定操作是否被取消，如在设置值中所示
JobType	返回操作的类型，如在设置值中所示
Cookie	返回操作的 ID。该 ID 在一个异步方法（ExportReport 或 PrintReport）被调用时设置

设置值

对 *JobType* 的设置如下：

常数	值	描述
rptAsyncPreview	0	报表正在处理一个预览操作
rptAsyncPrint	1	报表正在处理一个打印操
rptAsyncReport	2	报表正在处理一个导出报表操作

对 *cancel* 的设置如下：

设置值	描述
True	操作被取消
False	（缺省的）操作继续

说明

使用 ProcessingTimeout 事件使用户可以在指定的时间段内取消一个异步操作。

请参阅

ExportReport 方法，PrintReport 方法，AsyncProgress 事件，Error 事件(DataReportDesigner

ProcessTag 事件

在 WriteTemplate 处理中，当在 HTML 模板中发现令牌前缀标记时，该事件被激发，允许 WebClass 替换标记的内容。

应用于

WebItem 对象

语法

Private Sub *object*_**ProcessTag**(*TagName* **As String**, *TagContents* **As String**, *SendTag* **As Boolean**)

ProcessTag 事件的语法有这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
TagName	被替换的标记的名称
TagContents	当该事件被激发时，包含标记当前的内容。WebClass 代码可以改变该变量，以替换合适的内容
SendTag	当设置为 True 时，指定替换区域中的标记和内容都被替换。当设置为 False 时，只有 TagContents 参数的值被发送到文件，以替换整个替换区域。缺省值为 False

说明

仅当处理一个包含替换标记的模板时事件才会激发。

请参阅

RescanReplacements 属性；第三章的“在 Webclass 中执行文本置换”，第五部分的“开发 IIS 应用程序”，《MicrOsoftVisual Basic6.0 部件工具指南》一书的“构造 Internet 应用程序”。

ProcOfLine 属性

返回特定的行所在的过程名。

应用于

CodeModule 对象

语法

object.**ProcOfLine**(*line*,*prockind*)**AsString**

ProcOfLine 语法有三部分：

部分	描述
object	必需的。对象表达式，其值是“应用于”列表中的对象
line	必需的。指定要检查的行长型。
prockind	必需的。指定要定位的过程种类。属性过程在模块中可以有多种表示，必须指定要定位的过程种类。所有过程除了属性过程(即，Sub 和 Function 过程)用 vbext_pk_Proc

可以给 *prockind* 参数使用如下的常数之一：

常数	描述
vbext_pk_Get	指定一个返回属性值的过程
vbext_pk_Let	指定一个赋值给属性的过程
vbext_pk_Set	指定一个给对象设置引用的过程
vbext_pk_Proc	指定所有过程除了属性过程

说明

一个过程声明之前的空行或注释行属于该行，并且，如果该过程是一个代码模块的最后一个过程，则该模块后面将有一行或多行空白行。

ProcCountLines 属性返回在过程声明之前的所有空行及注释行的计

数，并且，如果该过程是一段代码模块的最后一个，那么紧接着该过程之后的空行或行也是。

请参阅

DeleteLines 方法，Find 方法(VBA 处接程序对象模型)，GetSelection 方法，InsertLines 方法，CodePane 对象，ProcBodyLine 属性，ProcCountLines 属性，ProcStartLine 属性

示例

以下示例使用 ProcOfLine 属性返回在特定代码窗格中包含指定行号的过程名。

```
Debug.PrintApplication.VBE.CodePanes(3).CodeModule.ProcOfLine(1270,vbext_pk_Proc)
```

ProcStartLine 属性

返回指定过程的起始行。

应用于

CodeModule 对象

语法

object.**ProcStartLine**(*procname*,*prockind*)**AsLong**

ProcStartLine 语法有三部分：

部分	描述
object	必需的。对象表达式，其值是“应用于”列表的对象
Procname	必需的。包含过程名称的字符串
Prockind	必需的。指定要定位的过程种类。属性过程在模块中可以有多种表示，必须指定要定位的过程种类。所有过程除了属性过程(即，Sub 和 Function 过程)用 vbext_pk_Proc

对 *prockind* 参数可以使用如下的常数之一：

常数	描述
vbext_pk_Get	指定一个返回属性值的过程
vbext_pk_Let	指定一个赋值给属性的过程
vbext_pk_Set	指定一个给对象设置引用的过程
vbext_pk_Proc	指定所有过程除了属性过程

说明

一个过程在前面过程的 EndSub 语句下开始第一行。如果该过程是第一个，那么此过程在通用声明部分之后开始。

请参阅

DeleteLines 方法，Find 方法(VBA 处接程序对象模型)，GetSelection 方法 InsertLines 方法，CodePane 对象，ProcBodyLine 属性，ProCountlines 属性，ProcOfLine 属性

示例

以下示例使用 ProcStartLine 属性返回指定过程开始的行到指定的

代码窗格。

```
Debug.PrintApplication.VBE.CodePanes(3).CodeModule.ProcStartLine("SetupTabs",vbext_pk_Proc)
```

ProductName 属性

返回或设定一个包括运行应用程序产品名的字任串值。在运行时只读。

应用于

App 对象

语法

*object*ProductName

object 占位符表示对象表达式，该对象表达式在“应用于”列表中表示出。

说明

可以在设计时在 Project 属性对话框的 Make 选项卡的 Type 框中设定此属性。

请参阅

Comments 属性，CompanyName 属性，FileDescription 属性，LegalCopyright 属性，LegalTrademarks 属性

ProgID 属性

返回由 VBControl 对象代表控件的 ProgID（程序 ID）。

应用于

AddIn 对象，VBControl 对象

语法

object.**ProgID**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

ProgID 属性（LicenseInfo 对象）

返回一个控件的程序 ID(progID)。

语法

object.**LicenseKey**

object 所在的位置是一个对象表达式，它的值将等于“应用于”列表中的一个对象。

说明

对于大多数的控件来说，通过检查 VisualBasic 对象浏览器可以得到程序 ID。程序 ID 包括两个部分：控件的 Library 的名称，以及它的 Class。例如，VisualBasicCommandButton 对象的程序 ID 是 VB.CommandButton。

Properties 集合

返回控件或部件的可用属性。

语法

Properties

说明

该对象或集合包含在设计时可正常访问的所有属性。

Properties 集合的缺省值由 Item 方法来决定。

可用 Properties 集合访问“属性”窗口中显示的属性。对“属性”窗口列出的每个属性，Properties 集合都有一个对象与之对应。

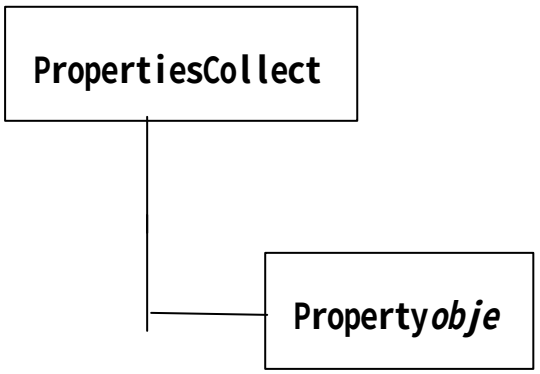
属性

VBE 属性，Count 属性(VB 集合)，Parent 属性

方法

Item 方法

Properties 集合（VBA 外接对象模型）



表示对象的属性。

说明

用 Properties 集合访问属性窗口中所显示的属性。对 Properties 窗口中列举的每一属性，Properties 集合都有一对象。

属性

Count 属性，Parent 属性，VBE 属性

方法

Item 方法(VBA 处接程序对象模型)

请参阅

Property 对象，Object 属性

Properties 属性

返回一个 WebItemProperties 对象，它是 WebItem 的用户定义属性的集合。

应用于

WebItem 对象

语法

object.Properties

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

允许创建和删除 WebItem 上的属性。

请参阅

WebItem 属性对象

Properties 属性

返回一个对象的属性，该属性为只读。

应用于

VBComponent 对象

说明

Properties 属性是一个访问者属性(即，该属性返回的对象类型与属性名相同)。

PropertyGet 语句

声明 Property 过程的名称，参数以及构成其主体的代码，该过程获取一个属性的值。

语法

[**Public**|**Private**|**Friend**][**Static**]**PropertyGet***name*[(*arglist*)][**As***type*]
[*statements*]
[*name=expression*]
[**ExitProperty**]
[*statements*]
[*name=expression*]
EndProperty

PropertyGet 语句的语法包含下面部分：

部分	描述
Public	可选的。表示所有模块的所有其它过程都可访问 PropertyGet 过程。如果在包含 OptionPrivate 的模块中使用，则该过程在该工程外是不可使用的
Private	可选的。表示只有包含其声明的模块的其它过程可以访问该 PropertyGet 过程
Friend	可选的。只能在类模块中使用。表示该 PropertyGet 过程在整个工程中都是可见的，但对对象实例的控制者是不可见的
Static	可选的。表示在调用之间保留该 PropertyGet 过程的局部变量的值。Static 属性对在该 PropertyGet 过程外声明变量不会产生影响，即使过程中也使用了这些变量
name	必需的。PropertyGet 过程的名称；遵循标准的变量命名约定，但不能与同一模块中的 PropertyLet 或 PropertySet 过程同名
arglist	可选的。代表在调用时要传递给 PropertyGet 过程的参数的变量列表。多个变量则用逗号隔开。PropertyGet 过程中的每个参数的名称和数据类型必须与相应 PropertyLet 过程（如果存在）中的参数一致

type	**可选的。该** PropertyGet **过程的返回值的数据类型；可以是** Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal**（目前尚不支持）**、Date、String**（除定长）**、*object*、Variant **或任何用户定义类型。任何类型的数组都不能作为返回值，但包含数组的** Variant **可以作为返回值** PropertyGet **过程的返回值类型必须与相应的** PropertyLet **过程（如果有）的最后一个（有时是仅有的）参数的数据类型相同，该** PropertyLet **过程将其右边表达式的值赋给属性**
statements	**可选的。**PropertyGet **过程体中所执行的任何语句组**
Expression	**可选的。**PropertyGet **语句所定义的过程返回的属性值**

其中的 *arglist* **参数的语法及语法的各个部分如下：**

[**Optional**][**ByVal**|**ByRef**][**ParamArray**]*varname*[()][**As***type*][=*defaultvalue*]

部分	描述
Optional	可选的。表示参数不是必需的。如果使用了该选项，则 arglist 中的后续参数都是可选的，而且必须都使用 Optional 关键字声明
ByVal	可选的。表示该参数按值传递
ByRef	可选的。表示该参数按地址传递。ByRef 是 VisualBasic 的缺省选项
ParamArray	可选的。只用于 *arglist* 的最后一个参数，指明最后这个参数是一个 Variant 元素的 Optional 数组。使用 ParamArray 关键字可以提供任意数目的参数。ParamArray 关键字不能与 ByVal、ByRef 或 Optional 一起使用
varname	必需的。代表参数的变量名称；遵循标准的变量命名约定。
type	可选的。传递给该过程的参数的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、**Decimal（目前尚不支持）**、Date、String（只支持变长）、*object* 或 Variant。如果参数不是 Optional，则也可以是用户自定义的类型，或对象类型

defaultvalue 可选的。任何常数或常量表达式。只在 Optional 参数时是合法的。如果类型为 *object*，则显式的缺省值只能是 Nothing

说明

如果没有使用 Public，Private 或 Friend 显式指定，则 Property 过程缺省为公用。如果没有使用 Static，则在调用之后不会保留局部变量的值。Friend 关键字只能在类模块中使用。Friend 过程可以被工程中的任何模块的过程访问。Friend 过程不会在其父类的类型库中出现，且 Friend 过程不能被后期绑定。

所有的可执行代码都必须属于某个过程。不能在别的 Property、Sub 或 Function 过程中定义 PropertyGet 过程。

ExitProperty 语句使执行立即从一个 PropertyGet 过程中退出。程序接着从调用该 PropertyGet 过程的语句下一条语句开始执行。在 PropertyGet 过程中的任何位置都可以有 ExitProperty 语句。

PropertyGet 过程与 Sub 和 PropertyLet 过程的相似之处是：PropertyGet 过程是一个可以获取参数，执行一系列语句，以及改变其参数的值的独立过程，而与 Sub 和 PropertyLet 过程不同的是：当要返回属性的值时，可以在表达式的右边使用 PropertyGet 过程，这与使用 Function 或属性名的方式一样。

请参阅

Function 语句，PropertyLet 语句，PropertySet 语句，Sub 语句，AddressOf 操作符，Friend

示例

该示例使用 PropertyGet 语句，定义获取属性值的属性过程。该属性用一个字符串来标识画笔的当前颜色。

```
DimCurrentColorAsInteger
ConstBLACK=0,RED=1,GREEN=2,BLUE=3

'用一个字符串返回画笔的当前颜色。
PropertyGetPenColor()AsString
   SelectCaseCurrentColor
      CaseRED
         PenColor="Red"
      CaseGREEN
         PenColor="Green"
      CaseBLUE
         PenColor="Blue"
   EndSelect
EndProperty

'下面的代码通过调用 PropertyGet 过程
'来获取画笔的颜色。
ColorName=PenColor
```

PropertyLet 语句

声明 PropertyLet **过程的名称，参数以及构成其主体的代码，该过程给一个属性值。**

语法

[Public|Private|Friend][Static]PropertyLet*name*(**[***arglist*,**]***value*)
    [*statements***]**
    [Exit*Property***]**
    [*statements***]**
EndProperty

PropertyLet **语句的语法包含下面部分：**

部分	描述
Public	可选的。表示所有模块的所有其它过程都可访问该 PropertyLet 过程。如果在包含 OptionPrivate 的模块中使用，则这个过程在该工程是不可使用的
Private	可选的。表示只有在包含其声明的模块的其它过程中可以访问该 PropertyLet 过程
Friend	可选的。只能在类模块中使用。表示该 PropertyLet 过程在整个工程都是可见的，但对于对象实例的控制者是不可见的
Static	可选的。表示在调用之间将保留 PropertyLet 过程的局部变量值。Static 属性对在该 PropertyLet 过程外声明的变量不会产生影响，即使过程中也使用了这些变量
name	必需的。PropertyLet 过程的名称；遵循标准的变量命名约定，但不能与同一模块中的 PropertyGet 或 PropertySet 过程同名
arglist	可选的。代表在调用时要传递给 PropertyLet 过程的参数的变量列表。多个变量则用逗号隔开。PropertyLet 过程中的每个参数的名称和数据类型必须与 PropertyGet 过程中的相应参数一致

value	必需的。指用于给属性赋值的变量。当调用该过程时，这个参数出现在调用表达式的右边。value 的数据类型必须和相应的 PropertyGet 过程的返回值类型一致
statements	可选的。PropertyLet 过程中执行的任何语句组

其中的 *arglist* 参数的语法和语法的各个部分如下：

[Optional][ByVal|ByRef][ParamArray]varname[()][Astype][=defaultvalue]

部分	描述
Optional	可选的。表示参数不是必需的。如果使用了该选项，则 arglist 中的后续参数都必须是可选的，而且必须都使用 Optional 关键字声明。注意，一个 PropertyLet 表达式的右边是不可能为 Optional 的
ByVal	可选的。表示该参数按值传递
ByRef	可选的。表示该参数按地址传递。ByRef 是 VisualBasic 的缺省选项
ParamArray	可选的。只用于 *arglist* 的最后一个参数，指明最后这个参数是一个 Variant 元素的 Optional 数组。使用 ParamArray 关键字可以提供任意数目的参数。ParamArray 关键字不能与 ByVal、ByRef 或 Optional 一起使用
varname	必需的。代表参数变量的名称；遵循标准的变量命名约定

type	可选的。传递给该过程的参数的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（只支持变长）、*objec* 或 Variant。如果参数不是 Optional，则也可以是用户定义类型，或对象类型
defaultvalue	可选的。任何常数或常数表达式。只在 Optional 参数时是合法的。如果类型为 *object*，则显式的缺省值只能是 Nothing

注意每个 PropertyLet 语句必须为其所定义的过程定义至少一个参数。当这个 PropertyLet 语句所定义的过程被调用时，该参数（如果有多个参数则指最后一个）就包含了将赋给属性的实际值。该参数就是前述语法中的 value。

说明

如果没有使用 Public、Private 或 Friend 显式指定，则 Property 过程缺省为公用。如果没有使用 Static，则不会在调用之后保留局部变量的值。Friend 关键字只能在类模块中使用。不过 Friend 过程可以被工程中的任何模块的过程访问。Friend 过程不会在其父类的类型库中出现，且 Friend 过程不能被后期绑定。

所有的可执行代码都必须属于某个过程。不能在别的 Property、Sub 或 Function 过程中定义 PropertyLet 过程。

ExitProperty 语句使执行立即从一个 PropertyLet 过程中退出。程序接着从调用该 PropertyLet 过程的语句下一条语句开始执行。在 PropertyLet 过程的任何位置都可以有 ExitProperty 语句。

PropertyLet 过程与 Function 和 PropertyGet 过程的相似之处是：PropertyLet 过程是一个可以获取参数，执行一系列语句，以及改变其参数值的独立过程。而与 Function 和 PropertyGet 过程不同的是：这两个过程都有返回值，而 PropertyLet 过程只能用于属性表达式或 Let 语句的左边。

请参阅

Function 语句，Let 语句，PropertyGet 语句，PropertySet 语句，Sub 语句，AddressOf 操作符，Friend

示例

该示例使用 PropertyLet 语句，定义给属性赋值的过程。该属性标识绘图盒的画笔颜色。

```
DimCurrentColorAsInteger
ConstBLACK=0,RED=1,GREEN=2,BLUE=3

'设置绘图盒的画笔颜色属性。
'模块级变量 CurrentColor 设为
'用于绘图的颜色值。
PropertyLetPenColor(ColorNameAsString)
   SelectCaseColorName              '检查颜色名称字符串。
      Case"Red"
         CurrentColor=RED       '设为 Red。
      Case"Green"
```

```
            CurrentColor=GREEN    '设为 Green。
        Case"Blue"
            CurrentColor=BLUE          '设为 Blue。
        CaseElse
            CurrentColor=BLACK    '设为缺省值。
    EndSelect
EndProperty

'下面的代码通过调用 Propertylet 过程
'来设置绘图盒的 PenColor 属性。

PenColor="Red"
```

Property 对象



用来描述一对象的属性，而且此对象在属性窗口中可见。

说明

可用 Property 对象的 Value 属性来返回或设置一个部件的属性值。

在最低限度上，所有部件都有 Name 属性。可用 Property 对象的 Value 属性来返回或设置属性的值。Value 属性可返回适当类型的 Variant。如果返回值是对象，则 Value 属性返回 Properties 集合，其中包含表示对象个别属性的 Property 对象。可用返回的 Properties 集合上的 Item 方法访问每个 Property 对象。

如果 Property 对象返回的值为一对象，则可用 *object* 属性将 Property 对象设置成新对象。

属性

集合属性，IndexedValue 属性(VBA 处接程序对象模型)，Name 属性(VBA 处接程序对象模型)，NumIndices 属性(VBA 外接程序对象模型)，Object 属性，Value 属性，VBE 属性

请参阅

Item 方法(VBA 外接程序对象模型)，属性集合(VBA 处接程序对象模型)

PropertySet 语句

声明 Property 过程的名称，参数以及构成其主体的代码，该过程设置一个对象引用。

语法

[Public|Private|Friend][Static]PropertySet*name*([arglist,]*reference*)

[*statements*]

[**ExitProperty**]

[*statements*]

EndProperty

PropertySet **语句的语法包含下面部分：**

部分	**描述**
Optional	**可选的。表示调用者可以提供或不提供该参数**
Public	**可选的。表示所有模块的所有其它过程都可访问这个** PropertySet **过程。如果在包含** OptionPrivate **的模块中使用，则这个过程在该工程外是不可使用的**
Private	**可选的。表示只有包含其声明的模块的其它过程可以访问该** PropertySet **过程**
Friend	**可选的。只能在类模块中使用。表示该** PropertySet **过程在整个工程中都是可见的，但对对象实例的控制者是不可见的**
Static	**可选的。表示在调用之间保留** PropertySet **过程的局部变量的值。**Static **属性对在该** PropertySet **外声明的变量不会产生影响，即使过程中也使用了这些变量**
name	**必需的。**PropertySet **过程的名称；遵循标准的变量命名约定，但不能与同一模块中的** PropertyGet **或** PropertyLet **过程同名**
arglist	**可选的。代表在调用时要传递给** PropertySet **过程的参数的变量列表。对于多个变量则用逗号隔开**

reference	必需的。对象引用赋值的右边所使用的包含对象引用的变量
statements	可选的。Property 过程体中所执行的任何语句组

其中的 arglist 参数的语法以及语法各个部分如下：

[**Optional**][**ByVal**|**ByRef**][**ParamArray**]*varname*[()][**As***type*][=*defaultvalue*]

部分	描述
Optional	可选的。表示参数不是必需的。如果使用了该选项，则 arglist 中的后续参数都必须是可选的，而且必须都使用 Optional 关键字声明。注意：PropertySet 表达式的右边不可能是 Optional
ByVal	可选的。表示该参数按值传递
ByRef	可选的。表示该参数按地址传递。ByRef 是 VisalBasic 的缺省选项
ParamArray	可选的。只用于 *arglist* 的最后一个参数，指明最后这个参数是一个 Variant 元素的 Optional 数组。使用 ParamArray 关键字可以提供任意数目的参数。ParamArray 关键字不能与 ByVal、ByRef 或 Optional 一起使用
varname	必需的。代表参数的变量的名称；遵循标准的变量命名约定

type	可选的。传递给该过程的参数的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（只支持变长）、*objec* 或 Variant。如果参数不是 Optional，则也可以是用户定义类型，或对象类型
defaultvalue	可选的。任何常数或常数表达式。只在 Optional 参数时是合法的。如果类型为 *object*，则显式的缺省值只能是 Nothing。

注意每个 PropertySet 语句必须为其所定义的过程定义至少一个参数。当 PropertySet 语句所定义的过程被调用时，这个参数（如果有多个参数则指最后一个）就包含了将赋给属性的实际的对象引用。这个参数就是前述语法中的 reference。它不能是 Optional。

说明

如果没有使用 Public、Private 或 Friend 显式指定，则 Property 过程按缺省情况是公用的。如果没有使用 Static，则在调用之后不会保留局部变量的值。Friend 关键字只能在类模块中使用。不过 Friend 过程可以被工程中的任何模块的过程访问。Friend 过程不会在其父类的类型库中出现，且 Friend 过程不能被后期绑定。

所有的可执行代码都必须属于某个过程。不能在别的 Property、Sub 或 Function 过程中定义 PropertySet 过程。

ExitProperty 语句使执行立即从一个 PropertySet 过程中退出。程序接着从调用该 PropertySet 过程的语句下一条语句执行。在 PropertySet 过程的任何位置都可以有 ExitProperty 语句。

PropertySet 过程与 Function 和 PropertyGet 过程的相似之处是：它们都是一个可以获取参数，执行一系列语句，以及改变其参数的值的独立过程。而与 Function 和 PropertyGet 过程不同的是：这两个过程都有返回值，而 PropertySet 过程只能用于对象引用赋值（Set 语句）的左边。

请参阅

Function 语句，Let 语句，PropertyGet 语句，PropertyLet 语句，Set 语句，Sub 语句，AddressOf 操作符，Friend

示例

该示例使用 PropertySet 语句，定义设置对象引用的属性过程。

```
'该 Pen 属性可以设置为不同的 Pen 实现。
PropertySetPen(PAs object)
   SetCurrentPen=P  '将 Pen 赋给对象。
EndProperty
```

PropertyBag 对象

PropertyBag 对象包含跨越对象调用时需要保存和恢复的信息。

说明

通过 ReadProperties 事件和 WriteProperties 事件，可将 PropertyBag 对象传送到对象中来保存和恢复对象的状态。使用 PropertyBag 对象的方法，对象能够读写自身的属性。PropertyBag 对象的 ReadProperty 方法用来读取属性值，而 PropertyBag 对象的

WriteProperty 方法用来写属性值。属性值本身可以是一个对象；在这种情况下 PropertyBag 对象将试图保存它。

方法

ReadProperty 方法，WriteProperty 方法

请参阅

Read 属性事件，WriteProperties 事件

PropertyChanged 方法

通知容器某个属性的值已经变更。

应用于

UserControl 对象，UserDocument 对象

语法

object.**PropertyChanged***PropertyName*

PropertyChanged 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
PropertyName	字符串表达式，它代表已被控件变更的属性名

说明

通知容器某属性值已变更后，容器就可以用对象的新属性值同步自身的属性窗口。如果不通知容器某个属性值已经变更，容器就不知道某个对象的实例是否需要保存（通过产生 WriteProperties 事件）。

例如，当更改属性页上的某个属性值时，或者当对象自己更改属性值时，需要调用该方法。当修改某个数据连接属性时，也需要调用该方法；否则数据源无法更新。

仅仅在运行时可用的属性不需要调用 PropertyChanged 方法，除非它们能被数据绑定。

作为示例，下面的代码说明了如何使用 PropertyChanged 方法：

```
PublicPropertyLetAddress(ByValcValueAsString)
    m_Address=cValue
    PropertyChanged"Address"
EndProperty
```

请参阅

WriteProperty 事件

PropertyName 属性

PropertyName 属性的行为取决于使用它的上下文。

AsyncRead 方法——设置属性名称，它将与 AsyncProperty 对象的 Value 属性相关联。

AsyncReadComplete 事件——指定当前正被读取的属性名称。在调用 AsyncRead 方法时，应对赋给 AsyncProperty 对象的名称作出响应。

DataBinding 对象——只读。返回 DataBinding 对象引用的属性名。

应用于

Binding 对象，AsyncProperty 对象，DataBinding 对象

语法

object.**PropertyName**=*string*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
string	将被保存或者获取的属性的名称

请参阅

AsyncRead 方法，AsyncReadComplete 事件

示例

该示例将一个值赋给 AsyncRead 方法中的 PropertyName 属性。用同样的值将方法的结果赋给 PictureBox 控件。要试用此例，将 PictureBox 控件放到 UserDocument 对象上。将代码键入到通用部分，并按 F5 键运行。启动 InternetExplorer3.0（或者更新版本），并将 UserDocument.vbd 文件的路径及其文件名粘贴到“地址”框中。

```
PrivateSubUserDocument_InitProperties()
   DimstrPathAsString
   '将变量设置为自己计算机上位图的
   '有效路径。
   strPath="C:\ProgramFiles\MicrosoftVisualStudio\VB\"&_
       "Samples\VCR\Bfly1.bmp"
   AsyncReadstrPath,vbAsyncTypeFile,_
       PropertyName:="butterfly"
```

```
EndSub

PrivateSubUserDocument_AsyncReadComplete(AsyncProp_
   AsAsyncProperty)
   '使用 Select 语句决定所返回的
   '属性。
   SelectCaseAsyncProp.PropertyName
        Case"butterfly"
             Picture1.Picture=_
             LoadPicture(AsyncProp.Value)
   EndSelect
EndSub
```

PropertyPage 对象

用来创建 ActiveX 属性页的基类对象。

说明

属性页为“属性”窗口提供了查看属性的另一种途径。可在一页上将几个相关的属性组成组，或用页面为那些对“属性”窗口来说太复杂的属性提供一个话框形式的接口，而这个属性对了。PropertyPage 对象代表一个页面，该页面可以说就是“属性页”对话框中的一个选项卡。

属性

Changed 属性，RightToLeft 属性，SelectedControls 属性，Conthols 属性，Palette 属性，PaletteMode 属性，，StandardSize 属性，OLEDropMode 属性，BackColor，ForeColor 属性，DrawWidth 属性，FontBold，FontItalic，FontStrikethru，FontUnderline 属性，FontName 属性，FontSize 属性，Height，Width 属性，Picture 属性，Tag 属性，AutoRedraw 属性，ClipControls 属性，CurrentX，CurrentY 属性，DrawMode 属性，DrawStyle 属性，FillColor 属性，FillStyle 属性，hDC 属性，hWnd 属性，Image 属性，KeyPreview 属性，MouseIcon 属性，MousePointer 属性，ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性，Active 控件属性，Appearance 属性，FontTransparent 属性，Caption 属性，Count 属性，(VB 集合)，HelpContextID 属性，Name 属性，Font 属性

方法

Refresh 方法，SetFocus 方法，Circle 方法，Line 方法，PSet 方法，Cls 方法，PainPicture 方法，Point 方法，PopupMenu 方法，Scale 方法，ScaleX，ScaleY 方法，TextHeight 方法，TextWidth 方法，OLEDrag 方法

事件

ApplyChanges 事件，EditProperty 事件，GotFocus 事件(User 控件对象和 UserDocument 对象)，LostFocus 事件(User 控件对象和 UserDocument 对象)，SelectionChanged 事件，Activate，Deactivate 事件，Click 事件，DragDrop 事件，DragOver 事件，

KeyDown，KeyUp 事件，KeyPress 事件，Load 事件，MouseDown，MouseUp 事件，MouseMove 事件，Paint 事件，Unload 事件，DbleClick 事件，Initialize 事件，Terminate 事件，OLECOmpleteDrag 事件，OledragDrop 事件，OLEDragOver 事件，OLDGiveFeedback 事件，OLDSetData 事件，OLEStartDrag 事件

PropertyPage 属性

返回某个 Member 对象的 PropertyPage 属性，或者对其进行设置。

应用于

Member 对象

语法

object.**PropertyPage**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

PropertyPages 属性

返回或设置字符串，该字符串是与控件关联的属性页的名称。

应用于

User 控件对象

语法

object.**PropertyPages**(*index*)[=*PropPageName*]

PropertyPages 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
index	字符串数组的索引
PropPageName	字符串，其中包含工程中属性页的名称

说明

PropertyPages 属性是字符串数组，其中包含了在工程中与该控件关联的属性页名称。可以通过设置数组的最后一项（此项通常为空）的方法在数组中添加属性页。要从数组中删除属性页，可将数组中那个元素的值设置为空字符串。

数组中属性页名称次序决定了这些页在控件的属性页对话框中显示的次序。

Protection 属性

返回一个值，指示一个工程的保护状态，此属性为只读。

应用于

VBProject 对象

返回值

Protection 属性返回值有：

常数	描述
Vbext_locked	指定的工程是被锁住的
Vbext_pp_none	指定的工程并没有被保护

请参阅

Mode 属性，Saved 属性

示例

下列示例使用 Protection 属性来返回一个值，指示工程是否处于保护状态。返回的值为一事先定义好的常量，表示工程的状态。

```
Debug.PrintApplication.VBE.ActiveVBProject.Protection
```

PSet 方法

将对象上的点设置为指定颜色。

应用于

PropertyPage 对象，User 控件对象，UserDocument 对象，Printer 对象，Printers 集合，Form 对象，Forms 集合，PictureBox 控件

语法

object.**PSet**[**Step**](*x*,*y*),[*color*]

PSet 方法的语法有如下对象限定符和部分：

部分	描述
object	可选的。对象表达式，其值为“应用于”列表中的对象。如果 *object* 省略，具有焦点的窗体作为 *object*
Step	可选的。关键字，指定相对于由 CurrentX 和 CurrentY 属性提供的当前图形位置的坐标
(*x*,*y*)	必需的。Single（单精度浮点数），被设置点的水平（x 轴）和垂直（y 轴）坐标
color	可选的。Long（长整型数），为该点指定的 RGB 颜色。如果它被省略，则使用当前的 ForeColor 属性值。可用 RGB 函数或 QBColor 函数指定颜色

说明

所画点的尺寸取决于 DrawWidth 属性值。当 DrawWidth 为 1，PSet 将一个像素的点设置为指定颜色。当 DrawWidth 大于 1，则点的中心位于指定坐标。

画点的方法取决于 DrawMode 和 DrawStyle 属性值。

执行 PSet 时，CurrentX 和 CurrentY 属性被设置为参数指定的点。

想用 PSet 方法清除单一像素，规定该像素的坐标，并用 BackColor 属性设置作为 color 参数。

这个方法不能用在 With‥EndWith 语句块中。

示例

这个示例用 PSet 方法在窗体上画五彩碎纸。想运行这个示例，将

代码放入窗体的General 部分。按F5 并单击窗体。

```
SubForm_Click()
    DimCX,CY,Msg,XPos,YPos              'Declarevariables.
    ScaleMode=3                         '设置ScaleMode 为像素。
    DrawWidth=5                         '设置DrawWidth.
    ForeColor=QBColor(4)                '设置前景为红色。
    FontSize=24                         '设置点的大小。
    CX=ScaleWidth/2                     '得到水平中点。
    CY=ScaleHeight/2                    '得到垂直中点。
    Cls                                 '清除窗体。
    Msg="HappyNewYear!"
    CurrentX=CX-TextWidth(Msg)/2        '水平位置。
    CurrentY=CY-TextHeight(Msg)             '垂直位置。
    PrintMsg                            '打印消息。
    Do
        XPos=Rnd*ScaleWidth             '得到水平位置。
        YPos=Rnd*ScaleHeight            '得到垂直位置。
        PSet(XPos,YPos),QBColor(Rnd*15)  '画五彩碎纸。
        DoEvents                        '进行其它处理。
    Loop
EndSub
```

Public 属性

返回或设置一个值，该值决定控件能否被其它应用程序共享使用。在创建控件时，Public 属性可读可写，在控件运行时，该属性不可用。

应用于

User 控件对象

设置值

Public 的设置值为：

设置值	描述
True	可以让其它应用程序共享控件。这是 ActiveX 控件工程类型的缺省设置值
False	不能让其它应用程序共享控件。控件包含在 ActiveX 控件工程中时，在该工程外控件不可见。这意味着工程中的其它控件和窗体可以使用该控件，但是外部的应用程序不能使用该控件。对于 ActiveX 控件工程类型以外的工程类型，这是唯一有效的设置值

Public 语句

在模块级别中使用，用于声明公用变量和分配存储空间。

语法

Public[**WithEvents**]*varname*[(*[subscripts]*)][As[**New**]*type*][,[**WithE**

vents]*varname*[([*subscripts*])][**As**[**New**]*type*]]...

Public 语句的语法包含下面部分：

部分	描述
WithEvents	**可选的。关键字，说明** *varname* **是用来响应由** ActiveX **对象触发的事件的对象变量。只有在类模块中才是合法的。使用** WithEvents**，可以定义任意个所需的单个变量，但不能用** WithEvents **创建数组。**New **和** WithEvents **不能一起使用**
varname	**必需的。变量的名称；遵循标准的变量命名约定**
subscripts	**可选的。数组变量的维数；最多可以定义** 60 **维的多维数组** subscripts **参数使用下面的语法：** [*lower*To]*upper*[,[*lower*To]*upper*]... **如果不显式指定** lower**，则数组的下界由** OptionBase **语句控制。如果没有** OptionBase **语句则下界为** 0
New	**可选的。用它可以隐式地创建对象的关键字。如果使用** New **声明对象变量，则在第一次引用该变量时创建该对象的新实例，因此不必使用** Set **语句来对该对象引用赋值。**New **关键字不能用来声明任何内部数据类型的变量以及从属对象的实例，也不能与** WithEvents **一起使用**

type	可选的。变量的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（对变长的字符串）、String*_length_（对定长的字符串）、*object* 或 Variant，用户定义类型，或对象类型。所声明的每个变量都要有一个单独的 As*type* 子句

说明

Public 语句声明的变量在所有应用程序的所有没有使用 OptionPrivateModule 的模块的任何过程中都是可用的；若该模块使用了 OptionPrivateModule，则该变量只是在其所属工程中是公用的。

小心不能在类模块中使用 Public 语句来声明一个定长的字符串变量。

使用 Public 语句可以声明变量的数据类型。例如，下面的语句声明了一个 Integer 类型的变量。

```
PublicNumberOfEmployeesAsInteger
```

也可以使用 Public 语句来声明变量的对象类型。下面的语句为工作表的新实例声明了一个变量。

```
PublicXAsNewWorksheet
```

如果在定义对象变量时没有使用 New 关键字，则在使用该变量之前，必须使用 Set 语句将一个已有的对象赋给这个引用对象的变量。在被赋值之前，所声明的这个对象变量有一个特定值 Nothing，这个值表示该变量没有指向任何对象的实例。

可以用带空圆括号的 Public 语句来声明动态数组。在声明了动态数组之后，可以在过程内用 ReDim 语句来定义该数组的维数和元素。如果试图在 Private、Public 或 Dim 语句中重定义一个已被显式定义了大小的数组的维数，就会发生错误。

如果不指定数据类型或对象类型，且在模块中没有使用 Def*type* 语句，则按缺省情况该变量为 Variant 类型。

当初始化变量时，数值变量被初始化为 0，变长的字符串被初始化为一个零长度的字符串(""),而定长的字符串则用 0 填充。Variant 变量被初始化为 Empty。用户自定义类型的变量的每个元素都作为各自独立的变量进行初始化。

请参阅

Array 函数,Const 语句,Dim 语句,OptionBase 语句,Optionrivate 语句，Private 语句，PropertyGet 语句，PropertyLet 语句，PropertySet 语句，ReDim 语句，Set 语句，Static 语句，Type 语句

示例

该示例在标准模块的模块级（通用部分）使用 Public 语句，来显式声明公用的变量，指如果没有使用 OptionPrivateModule，则在所有应用程序的所有模块的所有过程中都可以使用的变量。

```
Public Number As Integer              '公用的整数变量。
Public NameArray(1 To 5) As String    '公用的字符串数组变量。
'多个声明，两个变体型变量，以及一个整数型变量，都是公用的。
```

```
PublicMyVar,YourVar,ThisVarAsInteger
```

Put 语句

将一个变量的数据写入磁盘文件中。

语法

Put[#]*filenumber*,[*recnumber*],*varname*

Put 语句的语法具有以下几个部分：

部分	描述
filenumber	必需的。任何有效的文件号
recnumber	可选的。Variant(Long)。记录号（Random 方式的文件）或字节数（Binary 方式的文件），指明在此处开始写入
varname	必需的。包含要写入磁盘的数据的变量名

说明

通常用 Get **将** Put **写入的文件数据读出来。**

文件中的第一个记录或字节位于位置 1，第二个记录或字节位于位置 2，依此类推。如果省略 recnumber**，则将上一个** Get **或** Put **语句之后的（或上一个** Seek **函数指出的）下一个记录或字节写入。所有用于分界的逗号都必须罗列出来，例如：**

```
Put#4,,FileBuffer
```

下列规则适用于以 Random **方式打开的文件：**

如果已写入的数据的长度小于 Open **语句的** Len **子句指定的长度，**

则 Put 以记录长度为边界写入随后的记录。记录终点与下一个记录起点之间的空白将用现有文件缓冲区内的内容填充。因为填入的数据量无法确定，所以一般说来，最好设法使记录的长度与写入的数据长度一致。如果写入的数据长度大于由 Open 语句的 Len 子句所指定的长度，就会导致错误发生。

如果写入的变量是一个可变长度的字符串，则 Put 先写入一个含有字符串长度的双字节描述符，然后再写入变量。Open 语句的 Len 子句所指定的记录长度至少要比实际字符串的长度多两个字节。

如果写入的变量是数值类型的 Variant 则 Put 先写入两个字节来辨认 Variant 的 VarType，然后才写入变量。例如，当写入 VarType3 的 Variant 时，Put 会写入六个字节：其中，前两个字节辨认出 Variant 为 VarType3(Long)，后四个字节则包含 Long 类型的数据。Open 语句的 Len 子句所指定的记录长度必须至少比储存变量所需的实际字节多两个字节。

注意 Put 语句可用来将一个 Variant 数组写入磁盘，但不能用来将包含数组的标量 Variant 写入磁盘。Put 也不能用来将对象写入磁盘。

如果写入的变量是 VarType8(String)的 Variant，则 Put 先写入两个字节来辨认 VarType，接下来的两个字节则指出字符串的长度，然后再写入字符串数据。Open 语句的 Len 子句所指定的记录长度必须至少比实际的字符串长度多四个字节。

如果写入的变量是动态数组，则 Put 写入一个描述符，其长度等

于 2 加上 8 乘以维数，即 2+8*NumberOfDimensions。Open 语句的 Len 子句所指定的记录长度必须大于或等於为读出数组数据和数组描述符所需要的所有字节数总和。例如，在将数组写入磁盘时，下列数组声名需要 118 个字节的空间：

```
DimMyArray(1To5,1To10)AsInteger
```

这 118 个字节的分配情况如下：18 个字节用于描述符(2+8*2)，100 个字节用于数据(5*10*2)。

如果写入的变量是大小固定的数组，则 Put 只写入数据。它不将描述符写入磁盘。

如果写入的变量是任何其他类型的变量（不是可变长度的字符串或 Variant），则 Put 只写入变量数据。Open 语句的 Len 子句所指定的记录长度必须大于或等于要读出的数据长度。

Put 写入用户定义类型的元素时，除了不在元素之间进行填充外，好象是单独地写入每一个元素。在磁盘上，有一个描述符位于 Put 写入的用户定义的类型的动态数组之前，其长度等于 2 加上 8 乘以维数，即 2+8*NumberOfDimensions。Open 语句中的 Len 子句所指定的记录长度必须大于或等于为写入各个元素（包括任何数组及其描述符在内）所需的全部字节数总和。

对于以 Binary 方式打开的文件，上述所有 Random 规则都适用，除了：

Open 语句中的 Len 子句不起作用。Put 语句连续地将所有变量写入磁盘；也就是说，两个记录之间没有任何填充。

对于任何不属于用户定义的类型的数组，Put 只写入数据。它不

会写入描述符。

对于非用户定义的类型的可变长度字符串，Put 将其直接写入，而无需有双字节描述符。写入的字节数等于字符串所包含的字符数。例如，下列语句将十个字节写入文件号为 1 的文件中：

```
VarString$=String$(10,“ ”)
Put#1,,VarString$
```

请参阅

Get 语句，Open 语句，Seek 函数，VarType 函数

示例

本示例使用 Put 语句将数据写入文件中。示例中写入五个用户自定义数据类型 Record 的记录。

```
TypeRecord                                    '定义用户自定义数据类型。
   IDAsInteger
   NameAsString*20
EndType

DimMyRecordAsRecord,RecordNumber              '声明变量。
'以随机访问方式打开文件。
Open"TESTFILE"ForRandomAs#1Len=Len(MyRecord)
ForRecordNumber=1To5                          '循环五次。
   MyRecord.ID=RecordNumber                   '定义 ID。
```

```
    MyRecord.Name="MyName"&RecordNumber  '建立字符串。
    Put#1,RecordNumber,MyRecord          '将记录写入文件中。
NextRecordNumber
Close#1                                  '关闭文件。
```

PV 函数

返回一个 Double 指定在未来定期、定额支付且利率固定的年金现值。

语法

PV(*rate*,*nper*,*pmt*[,*fv*[,*type*]])

PV 函数有下列命名参数：

部分	描述
rate	必要。Double 指定每一期的利率。例如，如果有一笔贷款年百分比率(APR)为百分之十且按月付款的汽车贷款，则每一期的利率为 0.1/12 或 0.0083
nper	必要。Integer 指定一笔年金的付款总期数。例如，如果对一笔为期四年的汽车贷款选择按月付款，则贷款共有 4*12（或 48）个付款期
pmt	必要。Double 指定每一期的付款金额。付款金额通常包含本金和利息，且此付款金额在年金的有效期间不变
fv	可选。Variant，指定在付清贷款后所希望的未来值或现金结存。例如，贷款的未来值在贷款付清后为 0 美元。但是，如果想要在 18 年间存下 50，000 美元作为子女教育基金，那么 50，000 美元为未来值。如果省略的话，缺省值为 0
type	可选。Variant 指定贷款到期时间。如果贷款是在贷款周期结束时到期，则请使用 0。如果贷款是在周期开始时到期，则请使用 1。如果省略的话，缺省值为 0

说明

年金是在一段时间内一系列固定现金支付。年金可以是贷款（如房屋抵押贷款），也可以是一笔投资（如按月储蓄计划）。

在支付期间必须用相同的单位计算 *rate* 和 *nper* 参数。例如，如果 *rate* 用月份计算，则 *nper* 也必须用月份计算。

对所有参数，现金支出（如储蓄存款）用负数表示，而现金收入（如红利支票）用正数表示。

请参阅

DDB 函数，FV 函数，IPmt 函数，IRR 函数，MIRR 函数，NperFunction，NPV 函数，Pmt 函数，PPmt 函数，Rate 函数，SLN 函数，SYD 函数

示例

在本示例中，PV 函数计算一笔养老金的现值，该养老金将以每年支付$50,000 的方式，分二十年来支付$1,000,000 的总金额。计算时给定值为年利率期望值（APR），付款总期数（TotPmts），每期付款金额（YrIncome），未来总值（FVal）以及付款方式，以数值表示期初或期末付款（PayType）。请注意，YrIncome 是一个负数，因为它代表每年从养老金支付出去的现金。

```
DimFmt,APR,TotPmts,YrIncome,FVal,PayType,PVal
ConstENDPERIOD=0,BEGINPERIOD=1 '付款方式。
Fmt="###,##0.00"    '定义金额格式。
APR=.0825  '年利率。
TotPmts=20 '付款总期数。
YrIncome=50000 '每年收到金额。
FVal=1000000    '未来总值。
PayType=BEGINPERIOD     '期初付款。
PVal=PV(APR,TotPmts,-YrIncome,FVal,PayType)
MsgBox"Thepresentvalueis"&Format(PVal,Fmt)&"."
```

QBColor 函数

返回一个 Long，用来表示所对应颜色值的 RGB 颜色码。

语法

QBColor(*color*)

必要的 *color* 参数是一个界于 0 到 15 的整型。

设置值

color 参数有以下这些设置：

值	颜色	值	颜色
0	黑色	8	灰色
1	兰色	9	亮兰色
2	绿色	10	亮绿色
3	青色	11	亮青色
4	红色	12	亮红色
5	洋红色	13	亮洋红色
6	黄色	14	亮黄色
7	白色	15	亮白色

说明

color 参数代表使用于早期版本的 Basic（诸如 MicrosoftVisualBasicforMS-DOS 以及 BasicCompiler）的颜色值。始于最低有效字节，返回值指定了红、绿、蓝三原色的值，用于设置成 VBA 中 RGB 系统的对应颜色。

请参阅

RGB 函数

示例

本示例使用 QBColor 函数将 MyForm 窗体的 BackColor 属性，改成 ColorCode 参数指定的色彩。QBColor 可接受 0 到 15 的整型值。

```
SubChangeBackColor(ColorCodeAsInteger,MyFormAsForm)
    MyForm.BackColor=QBColor(ColorCode)
EndSub
```

QueryClose 事件

在窗体终止之前发生。

应用于

DataReport 对象

语法

PublicSub*object*_**QueryClose**(*cancel***AsInteger**, *closemode***AsInteger**)

QueryClose 事件的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
cancel	设置为除 0 以外的任意值，以终止设计器的关闭操作
closemode	一个值或常数，指示 QueryClose 事件的起因，如在返回值中描述

返回值

closemode 参数返回如下值：

常数	值	描述
VbFormControlMenu	0	用户已经从设计器上的“控件”菜单中选择了“关闭”命令，或是单击了“关闭”按钮
vbFormCode	1	代码中调用 Unload 语句
vbAppWindows	2	当前窗口操作环境对话正在结束
vbAppTaskManager	3	“任务管理器”窗口正在关闭应用程序
vbFormMDIForm	4	一个 MDI 子窗体因为 MDI 窗体正在关闭而关闭

说明

该事件的典型应用是确保在应用程序关闭之前，包含在应用程序中的设计器中没有未完成的任务。例如，如果用户还没有保存任何一个设计器中的新数据，应用程序就会提示用户保存数据。

当应用程序关闭时，可以使用 QueryClose 事件过程设定 Cancel 属性为 True，终止关闭过程。

QueryUnload 事件

在一个窗体或应用程序关闭之前发生。当一个 MDIForm 对象关闭时，QueryUnload 事件先在 MDI 窗体发生，然后在所有 MDI 子窗体中发生。如果没有窗体取消 QueryUnload 事件，该 Unload 事件首先发生在所有其它窗体中，然后再发生在 MDI 窗体中。当一个子窗体或一个 Form 对象关闭时，在那个窗体中的 QueryUnload 事件先于该窗体的 Unload 事件发生。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

PrivateSubForm_QueryUnload(*cancel***AsInteger**,*unloadmode***AsInteger**)

PrivateSubMDIForm_QueryUnload(*cancel***AsInteger**,*unloadmode***AsIntege r**)

QueryUnload 事件语法包括下列部分：

部分	描述
cancel	一个整数。将此参数设定为除 0 以外的任何值，可在所有已装载的窗体中停止 QueryUnload 事件，并阻止该窗体和应用程序的关闭
unloadmode	一个值或一个常数，如返回值中所描述的，它指示引起 QueryUnload 事件的原因

返回值

unloadmode 参数返回下列值：

常数	值	描述
vbFormControlMenu	0	用户从窗体上的“控件”菜单中选择“关闭”指令
vbFormCode	1	Unload 语句被代码调用
vbAppWindows	2	当前 MicrosoftWindows 操作环境会话结束
vbAppTaskManager	3	MicrosoftWindows 任务管理器正在关闭应用程序
vbFormMDIForm	4	MDI 子窗体正在关闭，因为 MDI 窗体正在关闭
vbFormOwner	5	因为窗体的所有者正在关闭，所以窗体也在关闭

这些常数是在对象浏览器中的 VisualBasic(VB)对象库中列出。

说明

此事件的典型用法是在关闭一个应用程序之前用来确保包含在该应用程序中的窗体中没有未完成的任务。例如，如果还未保存某一窗体中的新数据，则应用程序会提示保存该数据。

当一个应用程序关闭时，可使用 QueryUnload 或 Unload 事件过程将 Cancel 属性设置为 True 来阻止关闭过程。但是，QueryUnload 事件是在任一个卸载之前在所有窗体中发生，而 Unload 是在每个窗体卸载时发生。

请参阅

Unload 事件，Unload 语句

示例

本例使用一个包含两个 MDI 子窗体的 MDIForm 对象。当从 Control 菜单中选择 Close 命令关闭一个窗体时，所显示的信息与从 File 菜单中选择 Exit 命令是不同的。要尝试这个例子，可以先创建一个 MDIForm，然后使用菜单编辑器创建 File 菜单，该菜单中包含名为 FileExit 的 Exit 命令。确信这个菜单项可用。在 Form1 上，将 MDIChild 属性设置为 True。将代码粘贴到相应窗体的声明部分，然后按 F5 键执行程序。

```
'粘贴到 MDIForm1 的声明部分。
PrivateSubMDIForm_Load()
   DimNewFormAsNewForm1'Form1 的新实例。
   NewForm.Caption="Form2" '设置标题并显示。
EndSub
```

```
PrivateSubFileExit_Click()
    UnloadMDIForm1                  '退出应用。
EndSub

PrivateSubMDIForm_QueryUnload(CancelAsInteger,UnloadModeAsIntege
r)
    DimMsg                          '声明变量。
    '设置信息文本。
    Msg="Doyoureallywanttoexittheapplication?"
    '如果用户单击 No 按钮，则停止 QueryUnload。
    IfMsgBox(Msg,vbQuestion+vbYesNo,Me.Caption)=vbNoThenCancel=Tr
ue
EndSub

'粘贴到 Form1 的声明部分。
PrivateSubForm_QueryUnload(CancelAsInteger,UnloadModeAsInteger)
    DimMsg                          '声明变量。
    IfUnloadMode>0Then
        '如果正在退出应用。
        Msg="Doyoureallywanttoexittheapplication?"
    Else
        '如果正好在关闭窗体。
```

```
        Msg="Doyoureallywanttoclosetheform?"
    EndIf
    '如果用户单击No按钮，则停止QueryUnload。
    IfMsgBox(Msg,vbQuestion+vbYesNo,Me.Caption)=vbNoThenCancel=Tr
ue
EndSub
```

Quit 方法（外接程序）

试图退出 VisualBasic。

应用于

VBE 对象

语法

object.**Quit**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Raise 方法

产生运行时错误。

应用于

Err 对象

语法

object.**Raise***number*,*source*,*description*,*helpfile*,*helpcontext*

Raise 方法具有下列对象限定符和命名参数：

参数	描述
object	必需的。总是 Err 对象
number	必需的。Long 整数，识别错误性质。VisualBasic 错误（既有 VisualBasic 定义的错误也有用户定义的错误）的范围在 0–65535 之间。从 0–512 的范围保留为系统错误；从 513–65535 的范围可以用做用户定义的错误。当在类模块中将 Number 属性设置成自己的错误代码时，可将错误代码号添加到 vbObjectError 常数上。例如，为了产生错误号 513，可将 vbObjectError+513 赋值到 Number 属性
source	可选的。字符串表达式，为产生错误的对象或应用程序命名。当设置对象的这一属性时，要使用窗体 project.class。如果没有指定 source，则使用当前 VisualBasic 工程的程序设计 ID
description	可选的。描述错误的字符串表达式。如果没有指定，则检查 Number 的值。如果可以将错误映射成 VisualBasic 运行时错误代码，则将 Error 函数返回的字符串作为 Description 使用。如果没有与 Number 对应的 VisualBasic 错误，则要用到消息“应用程序定义的错误或对象定义的错误”

helpfile 可选的。帮助文件的完整限定的路径，在帮助文件中可以找到有关错误的帮助信息。如果没有指定，则 VisualBasic 会使用 VisualBasic 帮助文件的完整限定的驱动器、路径和文件名

helpcontext 可选的。识别 helpfile 内的标题的上下文 ID，而 helpfile 提供有助于了解错误的描述。如果省略，则使用处理有关错误的 VisualBasic 帮助文件的上下文 ID，该 ID 与 Number 属性对应

说明

除了 number 之外，所有参数都是可选的。如果使用 Raise 而不指定一些参数，并且 Err 对象的属性设置含有未清除的值，则视这些值为错误的值。

Raise 被用来生成运行时错误，并可用来代替 Error 语句。当书写类模块时要生成错误，Raise 是有用的，因为 Err 对象比 Error 语句可能提供更丰富的信息。例如，用 Raise 方法，可以在 Source 属性中说明生成错误的来源，可以引用该错误的联机帮助。

请参阅

HelpContextID 属性(VBA 外接程序对象模型)，Error 语句，OnError 语句，Clear 方法，Err 对象，Description 属性，HelpContext 属性，HelpFile 属性，LastDLLError 属性，Number 属性，Source 属性

示例

本示例使用 Err 对象的 Raise 方法在用 VisualBasic 写成的

Automation 对象中生成错误，该对象的程序代码为 MyProj.MyObject。在 Macintosh 中，缺省驱动器名为“HD”，且路径名部分由冒号而不是反斜杠分隔。

```
ConstMyContextID=1010407   '定义上下文 ID 的常数。
FunctionTestName(CurrentName,NewName)
   IfInstr(NewName,"bob")Then   '检测 NewName 的有效性。
      '产生例外
      Err.RaisevbObjectError+513,"MyProj.MyObject",_
      "No""bob""allowedinyourname","c:\MyProj\MyHelp.Hlp",_
      MyContextID
   EndIf
EndFunction
```

RaiseEvent 语句

引发在一个类、窗体、或者文档中的模块级中声明的一个事件。

语法

RaiseEvent*eventname*[(*argumentlist*)]

必需的 *eventname* 是在模块中声明的，并且符合 Basic 变量命名约定的一个事件的名称。

RaiseEvent 语句的语法有如下部分：

部分	描述
eventname	必需的。所引发的事件的名称
argumentlist	可选的。用逗号分隔的变量、数组，或者表达式的列表 Argumentlist 必须用圆括号括起来。如果没有参数，则圆括号必须被忽略

说明

如果在事件被引发的模块内该事件没有被声明，就会发生一个错误。下面这个程序片段说明了一个事件的声明以及一个引发该事件的过程。

```
'在类模块的模块级声明一个事件
EventLogonCompleted(UserNameasString)

Sub
   '引发该事件。
   RaiseEventLogonCompleted("AntoineJan")
EndSub
```

如果在 RaiseEvent 中该事件没有参数，包括空的圆括号，则对该事件的调用就会导致一个错误。不能使用 RaiseEvent 来引发在模块中没有明确定义的事件。例如，如果一个窗体有一个 Click 事件，则不能使用 RaiseEvent 来引发该窗体的 Click 事件。如果在窗体模块中声明了一个 Click 事件,则它将覆盖窗体自己的 Click

事件。仍然可以使用调用该事件的正常语法来调用该窗体的 Click 事件，但是不能使用 RaiseEvent 语句。

事件的引发是根据连接建立的顺序来进行的。因为事件可以有 ByRef 参数，所以后来连接的进程可能接收已经被一个更早的事件处理程序更改的参数。

请参阅

Event 语句

示例

下面的示例使用事件来计数 100 米短跑比赛的演示的最短时间秒数。代码演示了所有与事件有关的方法、属性、以及语句，包括 RaiseEvent 语句。

引发一个事件的类是事件源，而实现这个事件的类是吸收类。一个事件源可以有多个吸收类用于吸收这个事件源所产生的事件。当类引发事件时，该事件将在每一个被选择来为该对象实例吸收事件的类上被引发。

该示例使用了一个带有一个按钮（Command1）、一个标签（Label1）、和两个文本框（Text1 和 Text2）的窗体（Form1）。当单击该按钮时，第一个文本框显示“FromNow”并且第二个文本框开始时间计数。当整个时间（9.84 秒）结束时，第一个文本框将显示“UntilNow”，并且第二个文本框显示“9.84”。

Form1 的代码指定了窗体的最初状态和结束状态。它同时还包括当事件被引发时所要执行的代码。

```
OptionExplicit

PrivateWithEventsmTextAsTimerState

PrivateSubCommand1_Click()
Text1.Text="FromNow"
Text1.Refresh
Text2.Text="0"
Text2.Refresh
CallmText.TimerTask(9.84)
EndSub

PrivateSubForm_Load()
Command1.Caption="ClicktoStartTimer"
Text1.Text=""
Text2.Text=""
Label1.Caption="Thefastest100meterseverruntookthislong:"
SetmText=NewTimerState
EndSub

PrivateSubmText_ChangeText()
Text1.Text="UntilNow"
Text2.Text="9.84"
```

```
EndSub

PrivateSubmText_UpdateTime(ByValdblJumpAsDouble)
Text2.Text=Str(Format(dblJump,"0"))
DoEvents
EndSub
```

剩下的代码在一个名为 TimerState 的类模块中。包括在该模块的命令中的是 RaiseEvent 语句。

```
OptionExplicit
PublicEventUpdateTime(ByValdblJumpAsDouble)
PublicEventChangeText()
PublicSubTimerTask(ByValDurationAsDouble)
DimdblStartAsDouble
DimdblSecondAsDouble
DimdblSoFarAsDouble
dblStart=Timer
dblSoFar=dblStart

DoWhileTimer<dblStart+Duration
IfTimer-dblSoFar>=1Then
dblSoFar=dblSoFar+1
```

```
RaiseEventUpdateTime(Timer-dblStart)
EndIf
Loop

RaiseEventChangeText

EndSub
```

Randomize 语句

初始化随机数生成器。

语法

Randomize [*number*]

可选的 *number* 参数是 Variant 或任何有效的数值表达式。

说明

Randomize 用 number 将 Rnd 函数的随机数生成器初始化，该随机数生成器给 number 一个新的种子值。如果省略 number，则用系统计时器返回的值作为新的种子值。

如果没有使用 Randomize，则（无参数的）Rnd 函数使用第一次调用 Rnd 函数的种子值。

注意若想得到重复的随机数序列，在使用具有数值参数的 Randomize 之前直接调用具有负参数值的 Rnd。使用具有同样 number 值的 Randomize 是不会得到重复的随机数序列的。

请参阅

Timer 函数，Rnd 函数

示例

本示例用 Randomize 语句初始化随机数生成器。由于忽略了数值参数，所以 Randomize 用 Timer 函数的返回值作为新的随机数种子值。

```
DimMyValue
Randomize  '对随机数生成器做初始化的动作。

MyValue=Int((6*Rnd)+1) '生成 1 到 6 之间的随机数值。
```

Rate 函数

返回一个 Double，指定每一期的年金利率。

语法

Rate(*nper*,*pmt*,*pv*[,*fv*[,*type*[,*guess*]]])

Rate 函数有下列`命名参数：

部分	描述
nper	必要。Double 指定一笔年金的付款总期数。例如，如果对一笔为期四年的汽车贷款选择按月付款，则贷款共有 4*12（或 48）个付款期
pmt	必要。Double，指定每一期的付款金额。付款金额通常包含本金和利息，且此付款金额在年金的有效期间不变
pv	必要。Double 指定未来一系列付款或收款的现值。例如，当贷款买一辆汽车时，向贷方所借贷的金额为将来每月偿付给贷方款项的现值
fv	可选。Variant 指定在付清贷款后所希望的未来值或现金结存。例如，贷款的未来值在贷款付清后为 0 美元。但是，如果想要在 18 年间存下 50，000 美元作为子女教育基金，那么 50，000 美元为未来值。如果省略的话，缺省值为 0
type	可选。Variant，指定贷款到期时间，如果贷款是在贷款周期结束时到期，则请使用 0。如果贷款是在周期开始时到期，则请使用 1。如果省略的话，缺省值为 0
guess	可选。Variant 指定 Rate 返回的估算值。如果省略，则 *guess* 为 0.1(10%)

说明

年金是在一段时间内的一系列固定现金支付，年金可以是贷款（如房屋抵押贷款）或是一笔投资（如按月储蓄计划）。

对所有参数，现金支出（如储蓄存款）用负数表示，而现金收入（如红利支票）用正数表示。

Rate 是叠代计算的。先从 *guess* 的值开始，Rate 反复循环计算，直到精确度达到 0.00001%。如果经过 20 次叠代测试还不能得到结果，则 Rate 计算失败。如果猜测是 10%而 Rate 计算失败，则请试用不同的 *guess* 值。

示例

本示例使用 Rate 函数计算某贷款的利率，计算时需给定付款总期数（TotPmts），每期偿还贷款之金额（Payment），贷款的现值或本金（PVal），贷款的未来值（FVal），付款方式，以数值表示期初或期末付款（PayType），及利率的约略估计值（Guess）。

```
DimFmt,FVal,Guess,PVal,Payment,TotPmts,PayType,APR
ConstENDPERIOD=0,BEGINPERIOD=1      '付款方式。
Fmt="##0.00"                        '定义百分比的显示格式。
FVal=0                              '对贷款而言通常为零。
Guess=.1                            '利率估计值为 10%。
PVal=InputBox("Howmuchdidyouborrow?")
Payment=InputBox("What'syourmonthlypayment?")
TotPmts=InputBox("Howmanymonthlypaymentsdoyouhavetomake?")
PayType=MsgBox("Doyoumakepaymentsattheendofthemonth?",_
vbYesNo)
IfPayType=vbNoThenPayType=BEGINPERIODElsePayType=ENDPERIOD
```

```
APR=(Rate(TotPmts,-Payment,PVal,FVal,PayType,Guess)*12)*100
MsgBox"Yourinterestrateis"&Format(CInt(APR),Fmt)&"percent."
```

Read 方法

从一个 TextStream 文件中读取指定数量的字符并返回得到的字符串。

应用于

TextStream 对象

语法

object.**Read**(*characters*)

Read 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 TextStream 对象的名字
characters	必需的。从文件中要读取的字符数

请参阅

Close 方法, ReadAll 方法, ReadLine 方法, Skip 方法, SkipLine 方法, Write 方法, Write/BlankLines 方法, WriteLine 方法

ReadAll 方法

读取整个的 TextStream 文件并返回得到的字符串。

应用于

TextStream 对象

语法

object.**ReadAll**

object 始终是一个 TextStream 对象的名字。

说明

对于大的文件，使用 ReadAll 方法浪费内存资源。应使用其它的技术去输入一个文件，比如一行一行地读取文件。

请参阅

Close 方法，Read 方法，ReadLine 方法，Skip 方法，SkipLine 方法，Write 方法，WriteBlankLines 方法，WriteLine 方法

ReadFromFile 方法

从用 SaveToFile 方法创建的数据文件中加载对象。不支持命名的参数。

应用于

OLEContainer 控件

语法

object.**ReadFromFile***filenumber*

ReadFromFile 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
filenumber	必要的。数值表达式，指定用于加载对象的文件号。这个号必须与一个打开的二进制文件相对应

说明

可以使用 SaveToFile 或 SaveToOle1File 方法将对象保存到数据文件中。

应用于

TextStream 对象

ReadLine 方法

从一个 TextStream 文件读取一整行（到换行符但不包括换行符）并返回得到的字符串。

应用于

TextStream 对象

语法

object.**ReadLine**

object 参数始终是一个 TextStream 对象的名字。

请参阅

Close 方法，Read 方法，ReadAll 方法，Skip 方法，SkipLine 方法，Write 方法，WritebBlankLines 方法，WriteLine 方法

ReadOnly 属性

返回或设置一个值，该值确定一个 FileListBox 控件是否包含具有只读属性的文件。

应用于

FileListBox 控件

语法

object.**ReadOnly**[=*boolean*]

ReadOnly 属性的语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
boolean	一个布尔表达式，确定控件显示文件是否具有只读属性，如“设置值”中描述的

设置值

boolean 的设置值为：

设置值	描述
True	文件列表中包含只读文件
False	控件中不列出只读文件

说明

与 FileListBox 控件一起使用 ReadOnly 属性，以确定是否在列表中显示只读属性文件。

请参阅

Text 属性，Locked 属性

ReadOnlyMode 属性

该属性返回或设置决定 VisualBasic 开发环境如何与只读文件交互作用的值。

语法

object.**ReadOnlyMode**[=*value*]

ReadOnlyMode 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	决定工程状态的长整数或常数，如“设置值”所描述

设置值

value 的设置值是：

值	描述
1	严格的。如果工程文件是磁盘上的只读文件，则“删除文件”和“添加文件”命令是无效的。窗口和窗体可以移动，但不保存更改。对于在磁盘上也是只读的那些文件，代码窗口是只读的；不添加或删除控件，锁定控件的位置，并且自定义的“属性”对话框不能使用
0	宽松的。如果文件是只读的，可以更改代码、窗体和工程，但所作的更改不能保存回原文件

请参阅

IconState 属性

ReadProperties 事件

当加载具有保存状态的对象的旧实例时，发生该事件。

应用于

User 控件对象，UserDocument 对象，Class

语法

Sub*object*_**ReadProperties**(*pb***AsPropertyBag**)

ReadProperties 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
pb	类型为 PropertyBag 类的对象，其中包含要加载的保存数据

说明

该事件发生时，可以从 pb 中加载处于保存状态的数据，在加载每个数值时调用 PropertyBag 对象的 ReadProperty 方法。该事件在 Initialize 事件之后发生。

处理 ReadProperties 事件时，无论何时都应包括错误捕获，保护控件免受用户使用文本编辑器编辑包含保存数据的文件时，可能输入的无效属性值的影响。然而不能在事件中产生错误，因为这样对容器是致命的，所以 ReadProperties 事件过程中的任何

错误捕获都不应产生错误。

请参阅

Initialize 事件

Read 属性

从 PropertyBag 类对象中返回保存的数值。

应用于

PropertyBag 对象

语法

object.**ReadProperty**(*DataName*[,*DefaultValue*])

ReadProperty 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
dataName	字符串表达式，它代表属性包中的数值
defaultValue	如果属性包中没有任何数值，则返回该数值

说明

ReadProperty 方法返回由字符串表达式 DataName 表示的保存数据值，没有保存数据时，该方法返回由字符串表达式 DefaultValue 表示的数值。DataName 应与属性包中用来保存数值的字符串表达式匹配。

注意指定缺省的数值可以减少属于控件容器的文件大小。只有当要写入的数值与缺省数值不同时，才将属性行写入文件。在可能

的情况下，初始化、保存和获取属性值时都应为控件的属性指定缺省值。

ReadProperty 函数（外接程序）

该方法返回一个字符串，该字符串或者来自工程的.Vbp 文件中的指定的用户自定义节和键，或者来自部件文件中的指定的用户自定义节和键。

VBComponent 对象：

VBProject 对象：

语法

object.**ReadProperty**(*key*As String)As String

object.**ReadProperty**(*section*As String,*key*As String)As String

ReadProperty 函数的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
section	字符串表达式，包含从中找到键的节的名称
key	字符串表达式，包含返回的键名

说明

如果文件的 *section* 或 *key* 的区域是空值或不存在，将产生运行时错误 5：“非法函数调用”。

ReDim 语句

在过程级别中使用，用于为动态数组变量重新分配存储空间。

语法

ReDim[**Preserve**]*varname*(*subscripts*)[**As***type*][,*varname*(*subscripts*)[**As***type*]]...ReDim **语句的语法包括以下几个部分：**

部分	**描述**
Preserve	**可选的。关键字，当改变原有数组最末维的大小时，使用此关键字可以保持数组中原来的数据**
varname	**必需的。变量的名称；遵循标准的变量命名约定**
subscripts	**必需的。数组变量的维数；最多可以定义** 60 **维的多维数组** subscripts **参数使用下面的语法：** [*lower*To]*upper*[,[*lower*To]*upper*]... **如果不显式指定** *lower*，**则数组的下界由** OptionBase **语句控制。如果没有** OptionBase **语句则下界为** 0
type	**可选的。变量的数据类型；可以是** Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal**（目前尚不支持）**、Date、String**（对变长的字符串）**、String***length***（对定长的字符串）**、Object、Variant、**用户定义类型或对象类型。所声明的每个变量都要有一个单独的** As*type* **子句。对于包含数组的** Variant **而言，***type* **描述的是该数组的每个元素的类型，不能将此** Variant **改为其它类型**

说明

ReDim 语句用来定义或重定义原来已经用带空圆括号（没有维数下标）的 Private、Public 或 Dim 语句声明过的动态数组的大小。

可以使用 ReDim 语句反复地改变数组的元素以及维数的数目，但是不能在将一个数组定义为某种数据类型之后，再使用 ReDim 将该数组改为其它数据类型，除非是 Variant 所包含的数组。如果该数组确实是包含在某个 Variant 中，且没有使用 Preserve 关键字，则可以使用 Astype 子句来改变其元素的类型，但在使用了此关键字的情况下，是不允许改变任何数据类型的。

如果使用了 Preserve 关键字，就只能重定义数组最末维的大小，且根本不能改变维数的数目。例如，如果数组就是一维的，则可以重定义该维的大小，因为它是最末维，也是仅有的一维。不过，如果数组是二维或更多维时，则只有改变其最末维才能同时仍保留数组中的内容。下面的示例介绍了如何在为已有的动态数组增加其最末维大小的同时而不清除其中所含的任何数据。

```
ReDimX(10,10,10)
...
ReDimPreserveX(10,10,15)
```

同样地，在使用 Preserve 时，只能通过改变上界来改变数组的大小；改变下界则会导致错误。

如果将数组改小，则被删除的元素中的数据就会丢失。如果按地

址将数组传递给某个过程，那么不要在该过程内重定义该数组的各维的大小。

在初始化变量时，数值变量被初始化为 0，变长的字符串被初始化为一个零长度的字符串(""),而定长的字符串则用 0 填充。Variant 变量被初始化为 Empty。用户自定义类型的变量的每个元素作为各自独立的变量进行初始化。在使用引用对象的变量之前，必须使用 Set 语句将一个已有的对象赋给该变量。在被赋值之前，所声明的对象变量有一个特定值 Nothing，这个值表示该变量没有指向任何对象的实例。

小心如果 ReDim 语句所使用的变量在模块级别或过程级别不存在，则该语句就相当于一个声明语句。如果此后在一个更广的范围内又创建了同名的变量，即使使用了 OptionExplicit，ReDim 也将使用后声明的这个变量，且不会导致编译错误。为了避免这种冲突，就不应把 ReDim 作为声明语句使用，而只应作为重定义数组大小的语句。

注意要改变 Variant 所包含的数组的大小，必须在试图改变其数组大小之前显式声明该 Variant 变量。

请参阅

Array 函数，OptionBase 语句，Private 语句，Public 语句，Set 语句，Static 语句

示例

该示例使用 ReDim 语句为动态数组变量分配和重分配存储空间。假设 OptionBase 是 1。

```
DimMyArray()AsInteger          '声明动态数组。
RedimMyArray(5)                '分配 5 个元素。
ForI=1To5                      '循环 5 次。
   MyArray(I)=I                '初始化数组。
NextI
```

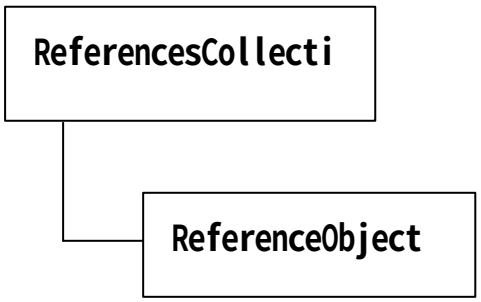
下一条语句重定义该数组的大小，并清除其中的元素。

```
RedimMyArray(10)               '大小重定为 10 个元素。
ForI=1To10                     '循环 10 次。
   MyArray(I)=I                '初始化数组。
NextI
```

下面的语句重定义该数组的大小，但没有清除其中的元素。

```
RedimPreserveMyArray(15)   '大小重定为 15 个元素。
```

Reference 对象



表示类型库或工程的引用。

说明

用 Reference 对象来检验一引用是否一直正确。

如果引用不再指向有效的引用，则 IsBroken 属性返回 True。如

果引用是一不能移动或删除的默认引用，则 BuiltIn 属性返回 True。可用 Name 属性决定，您所要添加或删除的引用是否正确。

属性

BuiltIn 属性，集合属性，Description 属性，FullPath 属性，GUID 属性，IsBroken 属性，Major 属性，Minor 属性，Name 属性(VBA 处接程序对象模型)，Type 属性(VBA 处接程序对象模型)，VBE 属性

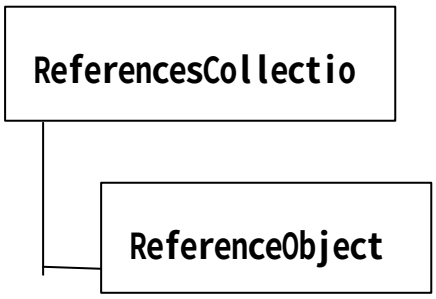
特别声明

References 集合

请参阅

AddFromFile 方法，AddFromGuid 方法，References 集合，

References 集合



表示工程中的引用集合。

说明

可用 References 集合添加或删除引用。References 集合与“引用”对话框中选项的引用集合相同。

属性

Count 属性 Parent 属性，VBE 属性

方法

AddFromFile 方法，AddFromGuid 方法，Item 方法(VBA 外接程序对象模型)，Remove 方法(VBA 外接程序对象模型)

请参阅

AddFromFile 方法，AddFfromGuid 方法，Item 方法(VBA 外接程序对象模型)，Remove 方法(VBA 外接程序对象模型)，Reference 对象，ReferencesEvents 对象，VBProject 对象

References 属性

返回一个工程中的引用的集合，该属性为只读。

应用于

VBProject 对象形字 Object

说明

References 属性是一个访问者属性(即，该属性返回的对象类型与属性名相同)。

ReferencesEvents 对象

由 ReferencesEvents 属性返回。

说明

ReferencesEvents 对象为一事件源，这些事件在添加或删除一

工程的引用时发生。将引用添加到工程后就会触发 ItemAdded 事件；从一工程中删除引用后就会触发 ItemRemoved 事件。

属性

ItemAdded 事件，ItemRemoved 事件

请参阅

Events 对象，References 集合，ReferencesEvents 属性

ReferencesEvents 属性

返回 ReferencesEvents 对象，此属性为只读。

应用于

Events 对象

设置值

传送给 ReferencesEvents 属性的参数设置可以是：

参数	描述
Vbproject	若 *vbproject* 指向 Nothing，则所返回的对象将可为 VBProjects 集合中的所有 VBProject 对象的 References 集合提供事件 若 *vbproject* 指向一个有效的 VBProject 对象，则所返回的对象将只为该工程的 References 集合提供事件

说明

ReferencesEvents 属性可接受一个参数且返回一个事件源对象。ReferencesEvents 对象是当引用被添加或删除时所触发的事件的来

源。

请参阅

Add 方法(VBA 处接程序对象模型)，Remove 方法(VBA 外接程序对象模型)，ReferencesEvents 对象，VBProject 对象，VBProjects 集合，CommandBarEvents 属性

示例

下列示例使用含有 ReferencesEvents 属性的代码，来支持在添加或删除引用时所用的代码。

```
PrivateWithEventsXAsReferencesEvents

SubTest()
SetX=Application.VBE.Events.ReferencesEvents
EndSub

PrivateSubX_ItemAdded(ByValReferenceAsVBIDE.Reference)
'将支持“添加”的代码写于此处
EndSub

PrivateSubX_ItemRemoved(ByValReferenceAsVBIDE.Reference)
'将支持“删除”的代码写于此处
EndSub
```

Refresh 方法

强制全部重绘一个窗体或控件。

应用于

DataReport 对象，Data 控件，PropertyPage 对象，User 控件对象，UserDocument 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件，Line 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，OLEContainer 控件

语法

object.**Refresh**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

在下列情况下使用 Refresh 方法：

在另一个窗体被加载时显示一个窗体的全部。

更新诸如 FileListBox 控件之类的文件系统列表框的内容。

更新 Data 控件的数据结构。

Refresh 方法不能用于 MDI 窗体，但能用于 MDI 子窗体。不能在 Menu 或 Timer 控件上使用 Refresh 方法。

通常，如果没有事件发生，窗体或控件的绘制是自动处理的。但

是，有些情况下希望窗体或控件立即更新。例如，如果使用文件列表框、目录列表框或者驱动器列表框显示当前的目录结构状态，当目录结构发生变化时可以使用 Refresh 更新列表。

可以在 Data 控件上使用 Refresh 方法来打开或重新打开数据库（如果 DatabaseName，ReadOnly，Exclusive 或 Connect 属性的设置值发生改变），并能重建控件的 Recordset 属性内的 dynaset。

请参阅

Paint 事件，MDIForm 对象，MDIChild 属性

示例

本例在创建测试文件时使用 Refresh 方法更新 FileListBox 控件。要试用此例，将以下代码粘贴至带有名为 File1 的 FileListBox 控件的窗体的声明部分，然后执行此例并单击窗体。

```
PrivateSubForm_Click()
   '声明变量。
   DimFilName,MsgasString,IasInteger
   File1.Pattern="TestFile.*"                    '设置文件模式。
   ForI=1To8                                  '执行八次。
      FilName="TESTFILE."&I
      '建立空文件。
      OpenFilNameForOutputAsFreeFile
      File1.Refresh                              '刷新文件列表框。
      Close                                      '关闭文件。
```

```
    NextI
    Msg="ChooseOKtoremovethecreatedtestfiles."
    MsgBoxMsg                          '显示消息。
    Kill"TESTFILE.*"                   '删除测试文件。
    File1.Refresh                          '更新文件列表框。
EndSub
```

ReleaseInstance 方法

释放一个 WebClass 对象。

应用于

WebClass 对象

语法

object.ReleaseInstance

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

如果 StateManagement 被设置成 wcRetainInstance，则 WebClass 对象在运行时释放 WebClass 对象。开发者应该在处理一个 HTTP 请求中，当 WebClass 已经完成了处理并将被终止时，调用 ReleaseInstance 方法。最终的 HTTP 响应一般包含一些 UI 元素，例如，一个超链接，它允许用户漫游到 WebClass 之外的其他 URL 上。如果没有这样的元素，则需要用户在浏览器中手工输入其他 URL，以便可以

漫游到其他地方。响应不应该包含该 WebClass 中其他 WebItems 的 URL，例如，使用 URLFor 创建的 WebItems。如果在响应中有这样的 URL，并且用户可以漫游到它，那么将创建一个新的 WebClass 实例。

Reload 方法

该方法从磁盘重载指定的部件，放弃任何未保存的变更。

应用于

VBComponent 对象

语法

object.**Reload**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

游标位置、代码窗口和窗体的可视性不受 Reload 方法的影响。Reload 不改变指示工程自上次保存以来是否被编辑过的设置值。

请参阅

FileCount 属性，FileNames 属性

Rem 语句

用来在程序中包含注释。

语法

Rem*comment*

也可以使用如下语法：

'*comment*

comment **参数是可选的，指要包括的任何注释文本。在** Rem **关键字与** *comment* **之间要加一个空格。**

说明

如果使用行号或行标签，则可以从 GoTo **或** GoSub **语句转到一个** Rem **语句行。程序会从该** Rem **语句下面的第一条可执行语句继续执行。如果在其它语句行后使用** Rem **关键字，则必须使用冒号(:)与语句隔开。**

可以用一个撇号(')来代替 Rem **关键字。若使用撇号，则在其它语句行使用时不必加冒号。**

示例

该示例说明用来在程序中包含注释的 Rem **语句的各种格式。**

```
DimMyStr1,MyStr2
MyStr1="Hello":Rem 注释在语句之后要用冒号隔开。
MyStr2="Goodbye"    '这也是一条注释；无需使用冒号。
```

Remove 方法（用于 VisualBasic 的应用程序）

把成员从 Collection **对象中删除。**

应用于

ExportFormats 集合，DataMembers 集合，Binding 集合对象，

StdDDataFormats 集合，集合对象，Dictionary 对象

语法

object.**Remove***index*

Remove 方法的语法具有下列对象限定符和部分：

部分	描述
Object	必需的。对象表达式，其值为“应用于”列表中的对象
Index	必需的。表达式，指定集合成员的位置。如果是数值表达式，则 *index* 必须是介于 1 和集合 Count 属性值之间的数。如果是字符串表达式，则在将被引用的成员添加到集合时，*index* 必须和 *key* 参数一致

说明

如果 index 的值与集合的现有成员不匹配，则会导致错误发生。

请参阅

ItemAdded 事件(VBA 处接程序对象模型)，ItemRemoved 事件(VBA 处接程序对象模型)，Add 方法（VBA 处接程序对象模型），Item 方法（VBA 处接程序对象模型）Add 方法，Item 方法

示例

本例证实 VBComponents 集合的特定数目是一个模块，然后使用 Remove 方法删除模块。

```
Debug.PrintApplication.VBE.ActiveVBProject.VBComponents(4).Name
Application.VBE.ActiveVBProject.VBComponents.Remove
Application.VBE.ActiveVBProject.VBComponents(4)
```

本示例说明如何使用 Remove 方法将 Collection 对象 MyClasses 中的对象删除。代码在每次循环中都将索引为 1 的对象删除。

```
DimNum,MyClasses
ForNum=1ToMyClasses.Count
   MyClasses.Remove1     '将第一个对象删除
                '直到删除所有对象为止。
NextNum
```

Remove 方法（VBA 外接程序对象模型）

从集合中删除一项。

应用于

LinkedWindows 集合，References 集合，VBComponents 集合

语法

object.**Remove**(*component*)

Remove 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
component	必需的。对于 LinkedWindows 集合来说，是一个对象。对于 References 集合来说，是一个对类型库，或者对工程的引用。对于 VBComponents 集合来说，是一个枚举型的常数，它代表一个类模块、一个窗体，或者是一个标准模块

说明

当在 LinkedWindows 集合使用时，Remove 方法可从当前链接窗口的集合中删除一个窗口。被删除的窗口可变成拥有属于自己的链接窗口框架的浮动窗口。

Remove 方法（FileSystemObject 对象）

从一个 Dictionary 对象中删除一个关键字和条目对。

语法

object.**Remove**(*key*)

Remove 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 Dictionary 对象的名字
key	必需的。*Key* 与要从 Dictionary 对象中删除的关键字和条目对相关联

说明

如果指定的关键字和条目对不存在，则发生一个错误。

下面的代码举例说明了 Remove 方法的使用：

```
Dima,d,i'创建一些变量
Setd=CreateObject("Scripting.Dictionary")
d.Add"a","Athens"'添加一些关键字和条目
d.Add"b","Belgrade"
d.Add"c","Cairo"
...
a=d.Remove()  '删除第二对
```

请参阅

Add 方法(Dictionary)，Exists 方法，Items 方法，Keys 方法，RemoveAll 方法

Remove 方法（VisualBasic 可扩展性）

该方法从集合中删除一项。

语法

object.**Remove**(*index*)

Item 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
index	必需的。variant 表达式，指定被访问对象在集合中的名称或索引

说明

对于 LinkedWindows 集合，从当前链接的几个窗口的集合中删除一个窗口。被删除的窗口变成一个有自己的 LinkedWindowFrame 的浮动窗口。此时创建 LinkedWindowFrame 窗口。

对于 VBProjects 集合，从该集合中删除指定的工程。

对于 References 集合，从该集合中删除指定的引用。

RemoveAddInFromToolbar 方法

在引用外接程序或向导的“外接程序”工具栏中删除按钮。

应用于

处接程序 Toolbar

语法

object.**RemoveAddInFromToolbar**(*saddinname*As**String**)

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
saddinname	必需的。字符串表达式，它指定了要从“外接程序”工具栏中删除的外接程序或者向导的名称（就象 AddToAddInToolbar 方法的 saddinname 参数所指定的那样）

请参阅

AddToAddInToolbar 方法

示例

该示例在“外接程序”工具栏中（该工具栏引用一个叫 MyAddInTitle 的假设的外接程序）删除已存在的按钮：

```
SubMain()
dimxasObject
Setx=CreateObject("AddInToolbar.Manager")
x.RemoveAddInFromToolbarsAddInName:="MyAddInTitle"
EndSub
```

RemoveAll 方法

RemoveAll 方法从 Dictionary 对象中删除所有关键字和条目对。

应用于

Dictionary 对象

语法

object.**RemoveAll**

object 始终是一个 Dictionary 对象的名字。

说明

下面的代码举例说明了 RemoveAll 方法的用法：

```
Dima,d,i'创建一些变量
Setd=CreateObject("Scripting.Dictionary")
d.Add"a","Athens"'添加一些关键字和条目
d.Add"b","Belgrade"
d.Add"c","Cairo"
...
a=d.RemoveAll'清除字典
```

请参阅

Add 方法词典，Exists 方法，Items 方法，Keys 方法，Remove 方法

RemoveItem 方法

用以从 ListBox 或 ComboBox 控件中删除一项，或从 MSFlexGrid 控件中删除一行。不支持命名参数。

应用于

ComboBox 控件，ListBox 控件

语法

object.**RemoveItem***index*

RemoveItem 方法的语法包含下列部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
index	必需的。一个整数，它表示要删除的项或行在对象中的位置。对于 ListBox 或 ComboBox 中的首项或 MSFlexGrid 控件中的首行，index=0

说明

被绑定到 Data 控件的 ListBox 或 ComboBox 不支持 RemoveItem 方法。

请参阅

ComboBox 控件，ListBox 控件

示例

本示例使用 Removeltem 方法将一个列表框中的输入项删除。要检验此示例，可将本例代码粘贴到一个带有名为 List1 的一个 ListBox 控件的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimEntry,I,Msg                    '声明变量。
   Msg="ChooseOKtoadd100itemstoyourlistbox."
   MsgBoxMsg                         '显示信息。
   ForI=1To100                       '计数值从 1~100。
       Entry="Entry"&I               '创建输入项。
```

```
        List1.AddItemEntry              '添加该输入项。
    NextI
    Msg="ChooseOKtoremoveeveryotherentry."
    MsgBoxMsg                       '显示信息。
    ForI=1To50                      '确定如何
        List1.RemoveItemI               '每隔一项
    NextI                           '删除。
    Msg="ChooseOKtoremoveallitemsfromthelistbox."
    MsgBoxMsg                       '显示信息。
    List1.Clear                         '清除列表框。
EndSub
```

Render 方法

在目标对象上绘制源图像的部分或整体。

应用于

Picture 对象

语法

object.**Render**(*hdc*,*xdest*,*ydest*,*destwid*,*desthgt*,*xsrc*,*ysrc*,*srcwid*,*srchgt*,*w*
bounds)

Render **方法的语法包含下面部分：**

部分	描述
object	必需的。对象表达式，其值是“应用于”列表中的一个对象
hdc	必需的。指向目标对象的设备描述体的句柄
xdest	必需的。目标对象中绘图区域左上角的 X 轴的坐标。该坐标用目标对象刻度单位
ydest	必需的。目标对象中绘图区域左上角的 Y 轴坐标。该坐标用目标对象
destwid	必需的。目标对象中绘图区域的宽度，其值是用目标对象刻度单位表示的
desthgt	必需的。目标对象中绘图区域的高度，其值是用目标对象刻度单位表示的
xsrc	必需的。源对象中绘图区域左上角的 X 轴的坐标。单位是 HIMETRIC
ysrc	必需的。源对象中绘图区域左上角的 Y 轴的坐标。单位是 HIMETRIC
srcwid	必需的。源对象中绘图区域的宽度，单位是 HIMETRIC
srchgt	必需的。源对象中绘图区域的高度，单位是 HIMETRIC
wbounds	必需的。元文件的世界边线。在绘制元文件时，该参数应传送一个与 RECTL 结构一致的用户定义类型，其它情况下应传送为 NULL

说明

在把图形的一部分绘制给一个目标时，推荐使用 PaintPicture 方法。

请参阅

PaintPicture 方法

Replace 函数

返回一个字符串，该字符串中指定的子字符串已被替换成另一子字符串，并且替换发生的次数也是指定的。

语法

Replace(*expression*,*find*,*replacewith*[,*start*[,*count*[,*compare*]]])

Replace 函数语法有如下几部分：

部分	描述
expression	必需的。字符串表达式，包含要替换的子字符串
find	必需的。要搜索到的子字符串
replacewith	必需的。用来替换的子字符串
start	可选的。在表达式中子字符串搜索的开始位置。如果忽略，假定从 1 开始
count	可选的。子字符串进行替换的次数。如果忽略，缺省值是–1，它表明进行所有可能的替换
compare	可选的。数字值，表示判别子字符串时所用的比较方式。关于其值，请参阅“设置值”部分

设置值

compare 参数的设置值如下：

常数	值	描述
vbUseCompareOption	–1	使用 OptionCompare 语句的设置值来执行比较
vbBinaryCompare	0	执行二进制比较
vbTextCompare	1	执行文字比较
vbDatabaseCompare	2	仅用于 MicrosoftAccess。基于您的数据库的信息执行比较

返回值

Replace 的返回值如下：

如果	Replace 返回值
expression 长度为零	零长度字符串(""
expression 为 Null	一个错误
find 长度为零	Expression 的复本
replacewith 长度为零	expression 的复本，其中删除了所有出现的 find 的字符串
start>Len(expression)	长度为零的字符串
countis0	Expression.的复本

说明

Replace 函数的返回值是一个字符串，但是，其中从 start 所指定的位置开始，到 expression 字符串的结尾处的一段子字符串已经

发生过替换动作。并不是原字符串从头到尾的一个复制。

请参阅

Filter 函数

ReplaceLine 方法

用特定的代码代替原代码。

应用于

CodeModule 对象

语法

object.**ReplaceLine**(*line*,*code*)

ReplaceLine 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
line	必需的。一个 Long 型数，用来指定所要代替的行
code	必需的。一个 String 型数，用来指定要插入的代码

请参阅

CreateEventProc 方法，DeleteLines 方法，InsertLines 方法，CountOfLines 属性，CountOfDeclarationLines 属性，CountOfVisibleLines 属性

示例

下列示例包含两个步骤。第一个步骤里的 For-Next 循环使用 InsertLines 方法将 26 个英文字母排成 26 行文本（第一行从只有一个字母“A”，第二行有两个字母“ab”，其余依此类推，最后一行包含“A”~“z”共 26 个字母），插入 CodePanes(1)。
第二个步骤里的 For…Next 循环使用 ReplaceLine 方法将偶数行的内容变成该行的最后一个字母，奇数行不变。

```
For I=1 to 26
    Application.VBE.CodePanes(1).CodeModule.InsertLines I,Mid$("abcdefghijklmnopqrstuvwxyz",1,I)
Next I
For I=1 to 13
    Application.VBE.CodePanes(1).CodeModule.ReplaceLine 2*I,Mid$("abcdefghijklmnopqrstuvwxyz",I,1)
Next I
```

ReportWidth 属性

以缇为单位返回或设置报表的宽度。

应用于

DataReport 对象

语法

object.ReportWidth[=*integer*]

ReportWidth 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
integer	可选的。一个数值表达式，以缇为单位指定报表的宽度

说明

DataReport 对象的 Height、Width、Top 和 Left 属性决定打印预览窗口的尺寸，而 ReportWidth 属性值指定实际的报表宽度。

Request 属性

返回活动服务器的 Request 对象。

应用于

WebClass 对象

语法

***object*.Request**

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

WebClass 使用 Request 对象读取客户请求的内容。Request 对象包含来自浏览器的、WebClass 能够用来响应请求的信息。其中包括用户输入到 HTML 窗体中的任何数据和 URL 中的参数。

如果WebClass对象使用URLData属性在URL中放置状态，则Request对象就被用于读取随后的HTTP请求的状态。

关于 Request 对象的属性、方法、和事件的详细信息，请参阅活动服务器页面文档。

RequestChangeFileName 事件

该事件发生在为部件或工程指定了新的文件名，且名字更改已完成之后。

应用于

FileControlEvents对象

语法

SubRequestChangeFileName(*vbproject***AsVBProject**,*filetype***Asvbext_FileType**,*newname***AsString**,*oldname***AsString**,*cancel***AsBoolean**)

RequestChangeFileName 事件语法包含下面部分：

部分	描述
vbproject	VBProject 对象，指出在其中添加新文件的工程名
filetype	数字值(vbext_FileType)，指出已写入的文件的类型，如“设置值”中列表所示
newname	字符串表达式，指出重新命名文件的名字
oldname	字符串表达式，指出文件在重新命名之前的名字
cancel	布尔表达式，确定缺省的 VisualBasic 动作，如“设置值”中的描述

设置值

filetype **的设置值是：**

常数	值	描述
vbext_ft_Form	0	**文件类型是窗体**
Vbext_ft_Module	1	**文件类型是基本模块**
Vbext_ft_Class	2	**文件类型是类模块**
Vbext_ft_Project	3	**文件类型是工程**
Vbext_ft_Exe	4	**文件类型是可执行文件**
Vbext_ft_Res	6	**文件类型是资源文件**
Vbext_ft_UserControl	7	**文件类型是** User **控件**
Vbext_ft_PropertyPage	8	**文件类型是** PropertyPage
Vbext_ft_DocObject	9	**文件类型是** UserDocument
Vbext_ft_Binary	10	**文件类型是二进制文件**
Vbext_ft_GroupProject	11	**文件类型是工程组**
Vbext_ft_Designer	12	**文件类型是设计器对象**

cancel **的设置值是：**

设置值	描述
True	**取消文件的重新命名。对任何后续的连接到** FileControl **对象的外接程序，都不触发该事件**
False	**对后续的连接到** FileControl **对象的外接程序继续触发该事件**

说明

该事件允许所有的外接程序检验准备添加到工程中的新文件名，并决定是否接受或取消名字更改。

该事件发生在所有连接到 FileControl 对象的外接程序中。外接程序不能阻止文件写盘，因为写盘操作已经完成。然而，可以用该事件执行其它任务，如：

记录关于该事件的信息。

更新有关该文件的信息。

备份该文件。

RequestEdit 属性

返回某个 Member 对象的 RequestEdit 属性，或者对其进行设置。

应用于

Member 对象

语法

object.**RequestEdit**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

RequestWriteFile 事件

该事件发生在保存任何未保存更改的工程部件之前。

应用于

FileControlEvents 对象

语法

SubRequestWriteFile(*vbproject***AsVBProject,***filename***AsString,***cancel***AsBoolean**)

RequestWriteFile **事件语法包含下面部分：**

部分	**描述**
vbproject	VBProject **对象，指出包含该部件的工程名**
filename	**字符串表达式，包含要保存的文件名**
cancel	**布尔表达式，用做取消该动作的标志，如“设置值”中的描述**

设置值

cancel **的设置值是：**

设置值	**描述**
True	**文件不写盘。对随后发生的任何连接到** FileControl **对象的外接程序，都不触发该事件**
False	**对随后发生的连接到** FileControl **对象的外接程序继续触发该事件**

说明

对每个保存的部件，该事件发生一次，对每个相关的二进制数据文件（如.Frx 或.Pgx 文件）也发生一次。

该事件允许外接程序来准备指定的写入文件。例如，用它可以使外接程序能在写入之前从源代码控件工程中签出文件。

该事件发生在所有连接到 FileControl **对象的外接程序中。外接程**

序不能阻止文件写盘，因为写盘操作已经完成。然而，可以用该事件执行其它任务，如：

记录关于该事件的信息。

更新有关该文件的信息。

备份该文件。

RescanReplacements 属性

设置或返回一个值，决定是否对包含在替换文本中的令牌重新扫描。至少扫描一次，且如果 RescanReplacement=TRUE，则包含令牌的替换文本将被连续扫描，直到没有剩下的令牌。

应用于

WebItem 对象

语法

object.**RescanReplacements**[=*boolean*]

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对
boolean	一个布尔表达式，指示是否重新扫描替换文本中的令牌

说明

请参阅 ProcessTag 事件，它允许用替换文本替换指定的字符串。

请参阅

ProcessTag 事件

Reset 语句

关闭所有用 Open 语句打开的磁盘文件。

语法

Reset

说明

Reset 语句关闭 Open 语句打开的所有活动文件，并将文件缓冲区的所有内容写入磁盘。

请参阅

End 语句，Close 语句，Open 语句

示例

本示例使用 Reset 语句关闭所有已打开的文件，并且将所有文件缓冲区内的数据写到磁盘中。请注意，示例中 Variant 变量 FileNumber 用来当作字符串及数值来使用。

```
DimFileNumber
ForFileNumber=1To5                    '循环五次。
   '打开输出文件。文件名为 TEST 加上文件代码 FileNumber，
   '跟在#之后它只是一个数值。
   Open"TEST"&FileNumberForOutputAs#FileNumber
   Write#FileNumber,"HelloWorld"          '将数据写入文件。
NextFileNumber
Reset                                 '关闭文件并将缓冲区内的数据写到磁
```

盘中。

Resize 事件

当一个对象第一次显示或当一个对象的窗口状态改变时该事件发生（例如，一个窗体被最大化、最小化或被还原）。

应用于

Data 控件，DataReport 对象，UserControl 对象，UserDocument 对象，Form 对象，Forms 集合，MDIForm 对象，PictureBox 控件，OLEContainer 控件

语法

PrivateSubForm_Resize()

PrivateSub*object*_**Resize**(*height***AsSingle**,*width***AsSingle**)

Resize 事件语法包括下列部分：

部分	描述
Object	一个对象表达式，其值是“应用于”列表中的一个对象
Height	指定控件新高度的数
Width	指定控件新宽度的数

说明

当父窗体调整大小时，可用 Resize 事件过程来移动控件或调整其大小。也可用此事件过程来重新计算那些变量或属性，如：ScaleHeight 和 ScaleWidth 等，它们取决于该窗体的尺寸。如果在调整大小时想要保持图形的大小与窗体的大小成比例，可在一个

Resize 事件中通过使用 Refresh 方法调用 Paint 事件。

任何时候只要 AutoRedraw 属性被设置为 False 而且窗体被调整大小，VisualBasic 也会按 Resize 和 Paint 的顺序调用相关的事件。当给这些相关事件附加过程时，要确保它们的操作不会互相冲突。

当一个 OLE 容器控件的 SizeMode 属性被设置为 2（自动调大小）时，该控件自动根据所显示的包含于该控件之中的对象的大小来调整其大小。如果所显示的对象的大小发生变化，则该控件自动重调其大小以适应该对象的变化。当这种情况出现时，为该对象调用 Resize 事件会在 OLE 容器控件被重调大小之前发生。height 和 width 部分指示该对象显示的最佳尺寸（这个尺寸由创建该对象的应用程序决定）。可通过在 Resize 事件中改变 height 和 width 部分的值来按不同的尺寸设定控件的大小。

请参阅

Paint 事件，Refresh 方法，Load 语句，Show 方法，AutoRedaw 属性

示例

本例在任何调整窗体大小的时候，都将自动调整一个 TextBox 控件的大小以填充该窗体。要尝试这个例子，可以将代码粘贴到包含 TextBox 的窗体声明部分。设置 TextBox 控件的 MultiLine 属性为 True，ScrollBars 属性为 3，BorderStyle 属性为 0，然后按 F5 键并调整窗体大小。

```
PrivateSubForm_Load()
   Text1.Text=""    '清除文本。
EndSub

PrivateSubForm_Resize()
   Text1.Move0,0,ScaleWidth,ScaleHeight
EndSub
```

Respond 事件

如果没有查找到与选定元素直接相关的事件，则当 WebItem 对象被用户请求激活时发生。

应用于

WebItem 对象

语法

PrivateSub*object*_**Respond**()

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

浏览器的请求经常与 WebItem 的 template 事件相对应。例如，当用户选择 Hyperlink1 时，webclass 就激发一个 Hyperlink1 事件。然而，如果在选定元素和 WebItem 的事件之间没有发现直接的相关性，那么系统就激发 Respond 事件。当使用 NextWebItem

方法从一个 WebItem 定位到另一个时，Respond 事件也会被激发。Respond 事件是 WebItem 对象的缺省事件。当该事件被激发时，代码必须使用 Response 对象创建一个要发送到浏览器的响应。

Response 属性

返回活动服务器页面的 Response 对象。

应用于

WebClass 对象

语法

object.Response

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

WebClass 使用 Response 对象发送信息给客户用于显示。关于针对此对象的属性、方法、和事件的详细信息，请参阅活动服务器页面文档。

Resume 语句

在错误处理程序结束后，恢复原先的运行。

语法

Resume[0]

ResumeNext

Resume*line*

Resume 语句的语法可以具有以下任何一种形式：

语句	描述
Resume	如果错误和错误处理程序出现在同一个过程中，则从产生错误的语句恢复运行。如果错误出现在被调用的过程中，则从最近一次调用包含错误处理程序的过程的语句处恢复运行
ResumeNext	如果错误和错误处理程序出现在同一个程序中，则从紧随产生错误的语句的下个语句恢复运行。如果错误发生在被调用的过程中，则对最后一次调用包含错误处理程序的过程的语句（或 OnErrorResumeNext 语句），从紧随该语句之后的语句处恢复运行
Resume*line*	在必要的 line 参数指定的 line 处恢复运行。line 参数是行标签或行号，必须和错误处理程序在同一个过程中

说明

在错误处理程序之外的任何地方使用 Resume 语句都会导致错误发生。

请参阅

End 语句，Exit 语句，OnError 语句，Err 对象

示例

本示例使用 Resume 语句来结束错误处理程序，然后将执行返回到产生错误的语句。以生成错误代号 55 的状况来示范如何使用 Resume 语句。

```
SubResumeStatementDemo()
    OnErrorGoToErrorHandler                 '打开错误处理程序。
    Open"TESTFILE"ForOutputAs#1         '打开输出文件。
    Kill"TESTFILE"                          '试图删除已打开的文件。
    ExitSub                                 '退出程序，以避免进入错误处理
程序。
ErrorHandler:                               '错误处理程序。
    SelectCaseErr.Number                '检查错误代号。
        Case55                              '发生“文件已打开”的错误。
            Close#1                         '关闭已打开的文件。
        CaseElse
            '处理其他错误状态...
    EndSelect
    Resume   '将执行返回到发生错误的语句。
EndSub
```

Resync 方法（远程数据）

取得当前行的批冲突值。

语法

object.**Resync**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

仅当使用客户批光标时，Resync 方法才是有效的。

Resync 使得光标库中当前行的各列跟服务器（该服务器对于事务来说是可见的）上的当前数据重新同步。如果尚未修改该行，则此方法将会改动 Value 及 OriginalValue 属性，以使其与当前在服务器上的数据相匹配。

如果已修改该行，则此方法将仅调整 OriginalValue 属性，这样就不会使编辑松散。在想要避免某个开放式并发冲突时，第二种情况是很有用的。

使用此方法的最后一种情况是：对于试图用 BatchUpdate 进行更新的行，在处理该行时，由于并发检查失败而出现冲突。在这种情况下，此方法将对 BatchConflictValue 进行调整，以反映服务器上列的最新版本。

Revision 属性

返回或设置该工程的修订版本号。该属性在运行时是只读的。

应用于

App 对象

语法

object.**Revision**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

Revision 属性的取值范围在 0~9999 之间。

该属性提供运行中的应用程序的版本信息。

在设计时，使用位于“工程属性”对话框中的“生成”选项卡上的“修订”框可设置该属性。

请参阅

Major 属性，Minor 属性

RGB 函数

返回一个 Long 整数，用来表示一个 RGB 颜色值。

语法

RGB(*red,green,blue*)

RGB 函数的语法含有以下这些命名参数：

部分	描述
red	必需的；Variant(Integer)。数值范围从 0~255，表示颜色的红色成份
green	必需的；Variant(Integer)。数值范围从 0~255，表示颜色的绿色成份
blue	必需的；Variant(Integer)。数值范围从 0~255，表示颜色的兰色成份

说明

可以接受颜色说明的应用程序的方法和属性期望这个说明是一个代表 RGB 颜色值的数值。一个 RGB 颜色值指定红、绿、蓝三原色的相对亮度，生成一个用于显示的特定颜色。

传给 RGB 的任何参数的值，如果超过 255，会被当作 255。

下面的表格显示一些常见的标准颜色，以及这些颜色的红、绿、蓝三原色的成份：

颜色	红色值	绿色值	兰色值
黑色	0	0	0
兰色	0	0	255
绿色	0	255	0
青色	0	255	255
红色	255	0	0
洋红色	255	0	255
黄色	255	255	0
白色	255	255	255

请参阅

QBColor 函数

示例

本示例示范如何使用 RGB 函数来返回代表 RGB 色彩值的完整数值。此函数通常用在和色彩有关的方法或属性。示例中的 MyObject 对象及其属性仅供示范帮助之用。如果 MyObject 并不存在，或者没有 Color 属性，则会发生错误。

```
DimRED,I,RGBValue,MyObject
Red=RGB(255,0,0)              '返回代表红色的值。
I=75                          '初始化偏移量。
RGBValue=RGB(I,64+I,128+I) '同 RGB(75,139,203)。
MyObject.Color=RGB(255,0,0)     '设置 MyObject 的 Color 属性为红色。
```

Right 函数

返回 Variant(String)，其中包含从字符串右边取出的指定数量的字符。

语法

Right(*string*,*length*)

Right 函数的语法具有下面的命名参数：

部分	说明
string	必要参数。字符串表达式，其中最右边的字符将被返回。如果 string 包含 Null，将返回 Null
length	必要参数；为 Variant(Long)。为数值表达式，指出想返回多少字符。如果为 0，返回零长度字符串("")。如果大于或等于 string 的字符数，则返回整个字符串

说明

要知 string 的字符数，用 Len 函数。

注意 RightB 函数作用于包含在字符串中的字节数据。所以 length 指定的是字节数，而不是指定返回的字符数。

请参阅

Left 函数，Len 函数，Mid 函数

示例

本示例使用 Right 函数来返回某字符串右边算起的几个字符。

```
DimAnyString,MyStr
AnyString="HelloWorld" '定义字符串。
```

```
MyStr=Right(AnyString,1)    '返回"d"。
MyStr=Right(AnyString,6)    '返回"World"。
MyStr=Right(AnyString,20)   '返回"HelloWorld"。
```

RightToLeft 属性

返回布尔值，它指示双向系统上的文本显示方向和控件可见的外观。

应用于

AmbientProperties 对象，PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，Printers 集合，CheckBox 控件，ComboBox 控件，CommandButton 控件，Data 控件，Form 对象，Forms 集合，Frame 控件，HscrollBar，VscrollBar 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

object.**RightToLeft**

RightToLeft 属性的语法包含下面部分：

部分	描述
Object	对象表达式，其值为“应用于”列表中的对象

设置值

RightToLeft 属性可能的布尔返回值为：

设置值	描述
True	控件运行在双向平台上（例如，阿拉伯语 Windows95 或者希伯来语 Windows95），文本从右向左显示。此时，控件应修改自身的行为，例如将垂直滚动条放置在文本框或列表框的左侧，将标签放置在文本框的右侧等等
False	控件的表现和运行在非双向平台（例如英语 Windows95）一样，文本从左向右显示。如果容器没有实现这种环境属性，这将是缺省的设置值

RmDir 语句

删除一个存在的目录或文件夹。

语法

RmDir*path*

必要的 path 参数是一个字符串表达式，用来指定要删除的目录或文件夹。path 可以包含驱动器。如果没有指定驱动器，则 RmDir 会在当前驱动器上删除目录或文件夹。

说明

如果想要使用 RmDir 来删除一个含有文件的目录或文件夹，则会发生错误。在试图删除目录或文件夹之前，先使用 Kill 语句来删除所有文件。

请参阅

ChDir 语句，CurDir 函数，Kill 语句，MkDir 语句

示例

本示例使用 RmDir 语句删除已存在的目录或文件夹。

```
'假设 MYDIR 为一空的目录或文件夹。
RmDir"MYDIR"  '将 MYDIR 删除。
```

Rnd 函数

返回一个包含随机数值的 Single。

语法

Rnd[(*number*)]

可选的 number 参数是 Single 或任何有效的数值表达式。

返回值

如果 number 的值	Rnd 生成
小于 0	每次都使用 number 作为随机数种子得到的相同结。
大于 0	序列中的下一个随机数
等于 0	最近生成的数
省略	序列中的下一个随机数

说明

Rnd 函数返回小于 1 但大于或等于 0 的值。

number 的值决定了 Rnd 生成随机数的方式。

对最初给定的种子都会生成相同的数列，因为每一次调用 Rnd 函数都用数列中的前一个数作为下一个数的种子。

在调用 Rnd 之前，先使用无参数的 Randomize 语句初始化随机数生成器，该生成器具有根据系统计时器得到的种子。

为了生成某个范围内的随机整数，可使用以下公式：

```
Int((upperbound-lowerbound+1)*Rnd+lowerbound)
```

这里，upperbound 是随机数范围的上限，而 lowerbound 则是随机数范围的下限。

注意若想得到重复的随机数序列，在使用具有数值参数的 Randomize 之前直接调用具有负参数值的 Rnd。使用具有同样 number 值的 Randomize 是不会得到重复的随机数序列的。

请参阅

Timer 函数，Randomize 语句

示例

本示例使用 Rnd 函数随机生成一个 1~6 的随机整数。

```
DimMyValue
MyValue=Int((6*Rnd)+1) '生成 1~6 之间的随机数值。
```

RootFolder 属性

返回一个 Folder 对象，该对象表示一个指定驱动器的根文件夹。只读属性。

应用于

Drive 对象

语法

object.**RootFolder**

object 总是一个 Drive 对象。

说明

驱动器上所包含的所有文件和文件夹都可以使用返回的 Folder 对象进行访问。

请参阅

AvailableSpace 属性，DriveLetter 属性，DriveType 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，Path 属性，SerialNumber 属性,ShareName 属性,TotalSize 属性,VolumeName 属性

Round 函数

返回一个数值，该数值是按照指定的小数位数进行四舍五入运算的结果。

语法

Round(*expression*[,*numdecimalplaces*])

Round 函数语法有如下几部分：

部分	描述
expression	必需的。要进行四舍五入运算的数值表达式
numdecimalplaces	可选的。数字值，表示进行四舍五入运算时，小数点右边应保留的位数。如果忽略，则 Round 函数返回整数

RowColChange 事件

在当前单元改变为一个不同的单元时该事件发生。

语法

PrivateSub*object*_**RowColChange**([*index***AsInteger**,*lastrow***AsString**,*lastcol* **AsInteger**])

RowColChange 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一标识一个在控件数组中的控件
lastrow	（用于 DataGrid 控件）是一个字符串表达式，它用来指定前一行的位置
lastcol	（用于 DataGrid 控件）是一个整数，它用来指定前一列的位置

说明

无论何时，只要单击当前单元之外的任何一个单元，或在一个选

择中用 Col 和 Row 属性有计划地改变当前单元时，此事件都会发生。

SelChange 事件也会在单击一个新单元时发生，却不会在不改变当前单元的前提下对所选范围作有计划的改变时发生。

对 DataGrid 控件来说，当前单元的位置是由 Bookmark 和 ColIndex 属性提供的。前一个单元位置由 lastrow 和 lastcol 指定。如果对数据进行编辑然后将当前单元位置移动到一个新行，则对原有行的更新事件在另一个单元成为当前单元之前完成。

请参阅

SelChange 事件，Col，Row 属性

RowHeight 属性

返回或设置 DataGrid 控件中所有行的高。RowHeight 总是使用与 DataGrid 控件容器相同的度量单位。

语法

object.**RowHeight** [=*value*]

RowHeight 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个指定高度的数值表达式

说明

在运行时，通过将鼠标指针放到在行间的网格线上并拖动鼠标能够改变任何行的 RowHeight。

请参阅

Col，Row 属性

示例

本例在单击窗体时将当前行的高设为 500 缇。要试用此例，创建一个新的工程，使用“部件”对话框添加一个 DataGrid 控件到工具箱中（从“工程”菜单中，选择“部件”，然后选择“Microsoft 数据绑定网格控件”），然后在窗体上画一个 DataGrid 控件。将下面的代码粘贴到窗体的声明部分，按 F5 键运行该程序，然后选择一个单元并单击窗体。

```
PrivateSubForm_Load()
   DataGrid1.Rows=5'设置行和列。
   DataGrid1.Cols=7
EndSub
PrivateSubForm_Click()
   DataGrid1.RowHeight(DataGrid1.Row)=500
EndSub
```

RSet 语句

在一字符串变量中将一字符串往右对齐。

语法

RSet*stringvar=string*

RSet 语句的语法有下面这些部分：

部分	说明
stringvar	必需的。为字符串变量名
string	必需的。在 stringvar 内想往右对齐的字符串表达式

defstringexpression@veendf98.chm

说明

如果 stringvar 比 string 长，Rset 会将 stringvar 中空余的字符以空白代替，直至字符串开头。

注意 RSet 不能用于用户定义类型。

请参阅

Lset 语句

示例

本示例使用 RSet 语句将某字符串插入到另一字符串的最右边。

```
DimMyString
MyString="0123456789"  '设置字符串初值。
RsetMyString="Right->" 'MyString 的内容为"Right->"。
```

SaveAs 方法

该方法用新文件名将部件或工程保存在给定的位置。

应用于

VBComponent 对象，VBProject 对象，VBProjects 集合

语法

object.**SaveAs**(*newfilename***AsString**)

SaveAs 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
newfilename	必需的。字符串表达式，指定要保存部件的新文件名

说明

如果给了新路径名，就用新路径名。否则，就使用老路径名。如果新文件名无效或引用了只读文件，便产生一个错误。

当保存一个窗体时，newfilename 指定窗体文件的新名。如果.Frx 文件是可用的话，则窗体被保存，并自动使用.Frx 扩展名。

注意成功地调用该方法会调用来自 FileControl 对象的关联事件。

Saved 属性

返回一个布尔型值，指示对象自上一次保存后是否编辑过，此属性为可读/写。

应用于

VBComponent 对象，VBProject 对象

返回值

Saved 属性可返回这些值：

值	描述
True	自上一次保存后对象没有编辑过
False	自上一次保存后对象编辑过

说明

SaveAs 方法设置 Saved 属性为 True。

注意如果在代码中把 Saved 属性设置为 False，就将返回 False，并且对象并标记为自上一次保存后编辑过。

请参阅

Protection 属性

示例

下列示例使用 Saved 属性返回一 Boolean 值，指示工程的当前状态是否已经保存过了。

```
Debug.PrintApplication.VBE.VBProjects(1).Saved
```

SavePicture 语句

从对象或控件（如果有一个与其相关）的 Picture 或 Image 属性中将图形保存到文件中。

语法

SavePicture*picture*,*stringexpression*

SavePicture 语句的语法包含下面部分：

参数	描述
picture	产生图形文件的 PictureBox 控件或 Image 控件
stringexpression	要保存的图形文件名

说明

无论在设计时还是运行时，图形从文件加载到对象的 Picture 属性，而且它是位图、图标、元文件或增强元文件，则图形将以原始文件同样的格式保存。如果它是 GIF 或 JPEG 文件，则将保存为位图文件。

Image 属性中的图形总是以位图的格式保存而不管其原始格式。

请参阅

LoadPicture 函数，Picture 属性，Image 属性

示例

本例使用 SavePicture 语句保存画在 Form 对象的 Picture 属性中的图形。要试用此例，可将以下代码粘贴到 Form 对象的声明部分，然后运行此例，单击 Form 对象。

```
PrivateSubForm_Click()
   '声明变量
   DimCX,CY,Limit,Radius    asInteger,MsgasString
   ScaleMode=vbPixels       '设置比例模型为像素
   AutoRedraw=True       '打开 AutoRedraw
   Width=Height          '改变宽度以便和高度匹配
   CX=ScaleWidth/2       '设置 X 位置
   CY=ScaleHeight/2      '设置 Y 位置
```

```
    Limit=CX                    '圆的尺寸限制
    ForRadius=0ToLimit          '设置半径
        Circle(CX,CY),Radius,RGB(Rnd*255,Rnd*255,Rnd*255)
        DoEvents                '转移到其它操作
    NextRadius
    Msg="ChooseOKtosavethegraphicsfromthisform"
    Msg=Msg&"toabitmapfile."
    MsgBoxMsg
    SavePictureImage,"TEST.BMP"  '将图片保存到文件。
EndSub
```

SaveSetting 语句

在 Windows 注册表中或(Macintosh 中)应用程序初始化文件中的信息保存或建立应用程序项目。

语法

SaveSettingappname,section,*key*,*setting*

SaveSetting **语句的语法具有下列命名参数：**

部分	描述
appname	必需的。字符串表达式，包含应用程序或工程的名称，对这些应用程序或工程使用设置在 Macintosh 中，这是 System 文件夹中 Preferences 文件夹中初始化文件的文件名
section	必需的。字符串表达式，包含区域名称，在该区域保存注册表项设置
key	必需的。字符串表达式，包含将要保存的注册表项设置的名称
setting	必需的。表达式，包含 key 的设置值

说明

如果无论如何也不能保存注册表项设置，则将导致错误发生。

请参阅

DeleteSetting 语句，GetAllSettings 函数，GetSetting 函数

示例

下列示例首先使用 SaveSetting 语句来建立 WindowsMacintosh 注册区（或 16 位 Windows 平台的.ini 档）里 MyApp 应用程序的项目，然后使用 DeleteSetting 语句来将之删除。

```
'在注册区中添加一些设置值
SaveSettingappname:="MyApp",section:="Startup",_
key:="Top",setting:=75
```

```
SaveSetting"MyApp","Startup","Left",50
'删除区段及所有的设置值
DeleteSetting"MyApp","Startup"
```

SaveToFile 方法

将对象保存到数据文件中。不支持命名的参数。

语法

object.**SaveToFile***filenumber*

SaveToFile 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
filenumber	必要的。数值表达式，指定用于保存对象的文件号。这个号必须与一个打开的二进制文件相对应

说明

使用这个方法可以保存 ActiveX 部件。为了将 ActiveX 部件按 OLEversion1.0 版本的格式保存，可用 SaveToOle1File 方法来代替。如果对象是链接的(OLEType=vbOLELinked,0)，只将链接信息及数据图像保存到指定的文件中。对象的数据则由创建该对象的应用程序来维护。如果对象是嵌入的(OLEType=vbOLEEmbedded,1)，则对象的数据由 OLE 容器控件维护，并可由 VisualBasic 应用程序保存。

可用 ReadFromFile 方法加载保存到数据文件中的对象。

请参阅

OLEType 属性，ReadFromFile 方法，SaveToOle1File 方法

SaveToOle1File 方法

将对象以 OLEversion1.0 版本的文件格式保存。不支持命名的参数。

应用于

OLE 包容器控件

语法

object.**SaveToOle1File***filenumber*

SaveToOle1File 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
filenumber	必要的。数值表达式，指定用于保存或加载对象的文件号。这个号必须与一个打开的二进制文件相对应

说明

如果对象是链接的(OLEType=vbOLELinked,0)，只将链接信息及数据图像保存到指定的文件中。对象的数据则由创建该对象的应用程序来维护。如果对象是嵌入的(OLEType=vbOLEEmbedded,1)，则对象的数据由 OLE 容器控件维护，并可由 VisualBasic 应用

程序保存。

对象如想以当前的 ActiveX 部件格式保存，可用 SaveToFile 方法来代替。

请参阅

ReadFormFile 方法，SaveToFile 方法

Scale 方法

用以定义 Form、PictureBox 或 Printer 的坐标系统。不支持命名参数。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，
Printers 集合，Form 对象，Forms 集合，PictureBox 控件

语法

object.**Scale**(*x1*,*y1*)-(*x2*,*y2*)

Scale 方法的语法包含下列部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象如果省略 object，则带有焦点的 Form 对象缺省为 object
x1,*y1*	可选的。均为单精度值，指示定义 object 左上角的水平（x-轴）和垂直（y-轴）坐标。这些值必须用括号括起。如果省略，则第二组坐标也必须省略
x2,*y2*	可选的。均为单精度值，指示定义 object 右下角的水平和垂直坐标。这些值必须用括号括起。如果省略，则第一组坐标也必须省略

说明

Scale 方法使您能够将坐标系统重置到所选择的任意刻度。Scale 对运行时的图形语句以及控件位置的坐标系统都有影响。

如果使用不带参数的 Scale（两组坐标都省略），坐标系统将重置为缇。

请参阅

TextHeight 方法，TextWidth 方法，ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性

示例

本示例使用 Move 方法设立一个自定义坐标系统，使得一个条形图可以在窗体上画出。要检验此示例，可将本例代码粘贴到一个

窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
    DimI,OldFontSize          '声明变量
    Width=8640:Height=5760    '按缇设置窗体大小
    Move100,100               '移动窗体起点
    AutoRedraw=-1                 '打开 AutoRedraw
    OldFontSize=FontSize      '保持旧的字体大小
    BackColor=QBColor(7)      '将背景设置为灰色
    Scale(0,110)-(130,0)          '设定自定义座标系统
    ForI=100To10Step-10
        Line(0,I)-(2,I)           '每隔 10 个单位划尺寸标记
        CurrentY=CurrentY+1.5     '移动光标位置
        PrintI                'Printscalemarkvalueonleft
        Line(ScaleWidth-2,I)-(ScaleWidth,I)
        CurrentY=CurrentY+1.5     '移动光标位置
        CurrentX=ScaleWidth-9
        PrintI                '将尺寸标记值打印在右边
    NextI
    '画条形图。
    Line(10,0)-(20,45),RGB(0,0,255),BF    '第一个蓝色条
    Line(20,0)-(30,55),RGB(255,0,0),BF    '第一个红色条
    Line(40,0)-(50,40),RGB(0,0,255),BF
```

```
    Line(50,0)-(60,25),RGB(255,0,0),BF
    Line(70,0)-(80,35),RGB(0,0,255),BF
    Line(80,0)-(90,60),RGB(255,0,0),BF
    Line(100,0)-(110,75),RGB(0,0,255),BF
    Line(110,0)-(120,90),RGB(255,0,0),BF
    CurrentX=18:CurrentY=100          '移动光标位置
    FontSize=14                       '放大标题尺寸
    Print"WidgetQuarterlySales"           '打印标题
    FontSize=OldFontSize              '还原字体大小
    CurrentX=27:CurrentY=93           '移动光标位置
    Print"PlannedVs.Actual"                '打印子标题
    Line(29,86)-(34,88),RGB(0,0,255),BF  '打印图例
    Line(43,86)-(49,88),RGB(255,0,0),BF
EndSub
```

ScaleHeight、ScaleWidth 属性

当使用图形方法或调整控件位置时，返回或设置对象内部的水平(ScaleWidth)或垂直(ScaleHeight)度量单位。对于 MDIForm 对象，在设计时是不可用的，并且在运行时是只读的。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，Printers 集合，Form 对象，Forms 集合，MDIForm

对象，PictureBox 控件

语法

object.**ScaleHeight**[=*value*]

object.**ScaleWidth**[=*value*]

ScaleHeight 和 ScaleWidth 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定水平或垂直度量的数值表达式

说明

能够使用这些属性来为绘图或打印创建一个自定义的坐标比例尺。例如，语句 ScaleHeight=100 将改变窗体实际内部高度的度量单位。取代当前高度为 *n* 个单位（缇、像素、...），高度将变为 100 个自定义单位。因而，50 个单位的距离就是对象的高度/宽度的一半，101 个单位的距离将超出对象 1 个单位。

为了定义基于标准度量单位的比例尺，例如缇、磅、像素、字符、英寸、毫米或厘米应使用 ScaleMode 属性。

这些属性设置为正值将使坐标从上向下及从左向右增加。它们设置为负值将使坐标从下向上及从右向左增加。

这些属性和相关的 ScaleLeft 与 ScaleTop 属性的使用，可以建立起一个完全的带有正、负坐标的坐标系统。所有这四个 Scale 属性与 ScaleMode 属性按下面的方式进行交互作用：

把其它任何 Scale 属性设置为任何值都将使 ScaleMode 自动设置

为 0。ScaleMode 等于 0 是用户定义。

把 ScaleMode 设置为一个大于 0 的数，将使 ScaleHeight 和 ScaleWidth 的度量单位发生改变，并将 ScaleLeft 和 ScaleTop 设置为 0。另外，CurrentX 和 CurrentY 的设置值将发生改变以反映当前点的新坐标。

也可以在语句中使用 Scale 方法设置 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 属性。

注意 ScaleHeight 和 ScaleWidth 属性与 Height 和 Width 属性是不一样的。

对于 MDIForm 对象，ScaleHeight 和 ScaleWidth 仅涉及窗体中未被 PictureBox 控件覆盖的区域。在 MDIForm 的 Resize 事件中应避免使用这些属性调整 PictureBox 的大小。

请参阅

Scale 方法，BackColor，ForeColor 属性，Height，Width 属性，CurrentX，CurrentY 属性，DrawMode 属性，DrawStyle 属性，FillColor 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性

示例

本例使用 ScaleHeight 和 ScaleWidth 属性来改变窗体的垂直和水平度量单位。要试用此例，先将下面的代码粘贴到窗体的声明部分，然后按 F5 键。要看到其效果，单击该窗体，改变它的大小，然后再次单击它。

```
PrivateSubForm_Click()
   DimRadiusAsInteger   '声明变量
   ScaleHeight=100      '设置高度的单位值
```

```
    ScaleWidth=100'设置宽度的单位值
    ForRadius=5to50Step5
        FillStyle=1
        Circle(50,50),Radius'画圆
    NextRadius
EndSub
```

ScaleLeft、ScaleTop 属性

当使用图形方法或调整控件位置时，返回或设置一个对象左边和上边水平(ScaleLeft)和垂直(ScaleTop)的坐标。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**ScaleLeft**[=*value*]

object.**ScaleTop**[=*value*]

ScaleLeft 和 ScaleTop 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定水平或垂直坐标的数值表达式。缺省设置值为 0

说明

这些属性和相关的 ScaleHeight 与 ScaleWidth 属性的使用，可以建立起一个完全的带有正、负坐标的坐标系统。这四个 Scale 属性与 ScaleMode 属性按下面的方式进行交互作用：

把其它任何 Scale 属性设置为任何值都将使 ScaleMode 自动设置为 0。ScaleMode 等于 0 是用户定义。

把 ScaleMode 设置为一个大于 0 的数，将使 ScaleHeight 和 ScaleWidth 的度量单位发生改变，并将 ScaleLeft 和 ScaleTop 设置为 0。另外，CurrentX 和 CurrentY 的设置值将发生改变以反映当前点的新坐标。

也可以在语句中使用 Scale 方法设置 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 属性。

注意 ScaleLeft 和 ScaleTop 属性与 Left 和 Top 属性是不一样的。

请参阅

Scale 方法，BackColor，ForeColor 属性，Left，Top 属性，DrawMode 属性，DrawStyle 属性，FillColor 属性，ScaleHeight，ScaleWidth 属性，ScaleMode 属性

示例

本例在 PictureBox 控件中创建一个网格，并将左上角的坐标设置为-1,-1 以代替 0,0。每隔 0.25 秒，从左上角到右下角随机地画些点。要试用此例，先将下面的代码粘贴到包含一个大的 PictureBox 和 Timer 控件的窗体的声明部分中，然后按 F5 键。

```
PrivateSubForm_Load()
   Timer1.Interval=250                    '设置计时器的间隔
   Picture1.ScaleTop=-1                   '为网格的顶部设置刻度
   Picture1.ScaleLeft=-1                  '为网格的左部设置刻度
   Picture1.ScaleWidth=2                  '设置刻度范围（-1 到 1）
   Picture1.ScaleHeight=2
   Picture1.Line(-1,0)-(1,0)                  '画水平线
   Picture1.Line(0,-1)-(0,1)                  '画垂直线
EndSub

PrivateSubTimer1_Timer()
   DimI                                   '声明变量。
   '在一个范围内随机地画些点。
   ForI=-1To1Step.05
       Picture1.PSet(I*Rnd,I*Rnd)         '画一个点。
   NextI
EndSub
```

ScaleMode 属性

当使用图形方法或调整控件位置时，返回或设置一个值，该值指示对象坐标的度量单位。

应用于

PropertyPage 对象，UserControl 对象，UserDocumetn 对象，Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**ScaleMode**[=*value*]

ScaleMode 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个指定度量单位的整数，“设置值”中有详细描述

设置值

value 的设置值为：

常数	设置值	描述
VbUser	0	指出 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 属性中的一个或多个被设置为自定义的值
VbTwips	1	（缺省值）缇（每逻辑英寸为 1440 缇；每逻辑厘米为 567 个缇）
VbPoints	2	磅（每逻辑英寸为 72 个磅）
VbPixels	3	像素（监视器或打印机分辨率的最小单位
VbCharacters	4	字符（水平每个单位=120 缇；垂直每个单位=240 缇。）
vbInches	5	英寸
vbMillimeters	6	毫米
vbCentimeters	7	厘米
vbHimetric	8	HiMetric
vbContainerPositio n	9	控件容器使用的单位，决定控件位置
vbContainerSize	10	控件容器使用的单位，决定控件的大小

说明

相关的 ScaleHeight、ScaleWidth、ScaleLeft 与 ScaleTop 属性的使用，可以建立起一个带有正、负坐标的自定义坐标系统。这四个 Scale

属性与 ScaleMode 属性按下面的方式进行交互作用：

把其它任何 Scale 属性设置为任何值都将 ScaleMode 自动地设置为 0。ScaleMode 等于 0 是用户定义。

把ScaleMode属性设置为一个大0的数,将使ScaleHeight和ScaleWidth的度量单位发生改变，并将 ScaleLeft 和 ScaleTop 设置为 0。CurrentX 和 CurrentY 的设置值将发生改变以反映当前点的新坐标。

请参阅

Scale 方法，BackColor，ForeColor 属性，DrawMode 属性，DrawStyle 属性，FillColor 属性，ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTo 属性

示例

本例显示不同的 ScaleMode 属性设置值是如何改变圆的大小的。要试用此例，先将下面的代码粘贴到窗体的声明部分，然后按 F5 键并单击窗体。单击窗体时，度量单位将改变为下一个 ScaleMode 设置值并在窗体上画一个圆。

```
PrivateSubForm_Click()
   '在七个“ScaleMode”设置值中循环。
   ScaleMode=((ScaleMode+1)Mod7)+1
   '在窗体的中心处画一个半径为 2 的圆。
   Circle(ScaleWidth/2,ScaleHeight/2),2
EndSub
```

ScaleUnits 属性

返回字符串值，该值是容器使用的坐标单位的名称。

应用于

AmbientProperties 对象

语法

object.**ScaleUnits**

ScaleUnits 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

该字符串代表控件的容器使用的坐标单位，例如“缇”。控件显示坐标值时，可以用此字符串作为单位指示标记。

如果容器没有实现这种环境属性，则缺省值是空字符串。

ScaleX、ScaleY 方法

用以将 Form，PictureBox 或 Printer 的宽度或高度值从一种 ScaleMode 属性的度量单位转换到另一种。不支持命名参数。

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox

控件

语法

object.**ScaleX**(*width*,*fromscale*,*toscale*)

object.**ScaleY**(*height*,*fromscale*,*toscale*)3

ScaleX 和 ScaleY 方法的语法包含如下部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象如果省略 object，则带有焦点的 Form 对象缺省为 object
width	必需的。为 object 指定被转换的度量单位的数量
height	必需的。为 object 指定被转换的度量单位的数量
fromscale	可选的。一个常数或数值，按照下列设置中的描述，指定 object 的 width 或 height 从哪一种坐标系统转换。fromscale 可取的数值与 ScaleMode 属性的数值加上 HiMetric 的新数值相同
toscale	可选的。一个常数或数值，按照下列“设置值”中的描述，指定 object 的 width 或 height 转换到哪一种坐标系统。toscale 可取的数值与 ScaleMode 属性的数值加上 HiMetric 的新数值相同

设置值

用于 fromscale 和 toscale 设置值有：

常数	值	描述
vbUser	0	用户定义：指示 object 的宽度和高度设置为自定义值
vbTwips	1	缇（每逻辑英寸 1440 缇；每逻辑厘米 567 缇）
vbPoints	2	磅（每逻辑英寸 72 点）
vbPixels	3	像素（显示器或打印机分辨率的最小单位）
vbCharacters	4	字符（水平=每单位 120 缇，垂直=每单位 240 缇）
vbInches	5	英寸
vbMillimeters	6	毫米
vbCentimeters	7	厘米
vbHimetric	8	HiMetric。如果省略 fromscale，则 HiMetric 为缺省值
vbContainerPositio n	9	决定控件位置
vbContainerSize	10	决定控件大小

说明

ScaleX 和 ScaleY 方法按 fromscale 指定的度量单位取值（width 或 height），并将它转换为 toscale 指定的度量单位下相应的值。也可以结合 PaintPicture 方法使用 ScaleX 和 ScaleY。

请参阅

ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性

Scope 属性

该属性返回一个公共的、私人的或者是朋友的成员。

应用于

Member 对象

语法

object.**Scope**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Screen 对象

根据窗体在屏幕上的布局而操作窗体，并在运行时控制应用程序窗体之外的鼠标指针。Screen 对象通过关键字 Screen 访问。

语法

Screen

说明

Screen 对象是指整个 Windows 桌面。当模式窗体被显示时，使用 Screen 对象可以设置 Screen 对象的 MousePointer 属性为沙漏指针(hourglasspointer)。

Screen 属性

返回一个 Screen 对象，该对象可以根据窗体在屏幕上的位置来对窗体进行操作，而且在运行时还能控制应用程序窗体之外的鼠标指针。Screen 对象是用关键字 Screen 来进行访问的。

属性

Height，Width 属性，FontCount 属性，Fonts 属性，MouseIcon 属性，MousePointer 属性，TwipsPerPixelX 属性，TwipsperpixeLY 属性，ActiveControl 属性，ActiveForm 属性

语法

Screen

说明

Screen 对象是整个 Windows 桌面。使用 Screen 对象就可以在显示一个模式窗体时，将 Screen 对象的 MousePointer 属性设置为沙漏指针。

应用于

Global 对象

请参阅

Global 对象，Screen 对象

Scroll 事件

当 ScrollBar 控件上的或包含一个滚动条对象的滚动框被重新定

位，或按水平方向或垂直方向滚动时，此事件发生。

语法

Private Sub *DataGrid*_**Scroll**([*cancel* **As Integer**])

Private Sub *object*_**Scroll**()

Scroll 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
cancel	如在说明中所描述的，它决定滚动操作是否成功以及 ScrollBar 或 DataGrid 是否被重绘

说明

对一个 DataGrid 控件来说，此事件当将网格水平或垂直地滚动，但在该网格被重绘之前发生。它用来显示滚动操作的结果。

对于一个 ComboBox 控件来说，此事件仅在位于该控件的下拉部分的滚动条被操作时发生。

将 cancel 设置为 True 会导致 DataGrid 滚动操作失败，并且不出现重绘操作。如果在此事件中调用 Refresh 方法则即使 cancel 被设置为 True，该网格也按其新的（滚动的）位置进行。然而，在这种情况下，由于滚动操作失败使它快速退回到它原来的位置，该网格被再一次定位。

可用此事件进行计算或操纵必须与滚动条中所进行的变动同步的控件。反之，当在 ScrollBar 控件变动之后想让更新只出现一次，可使用 Change 事件。

注意在此事件中应避免使用 MsgBox 语句或函数。

应用于

UseDocument 对象，ComboBox 控件，DirListBox 控件，Drive ListBox 控件 FileListBox 控件，HScrollBar，VScrollBar 控件，ListBox 控件

示例

本例在滚动条上拖动滚动框时，改变 Shape 控件的大小来响应一个水平滚动条(HScrollBar)的值。要尝试这个例子，可以将代码粘贴到包含一个 Shape 控件、一个 Label 控件和一个 HScrollBar 控件窗体的声明部分。将 Shape 控件 Index 属性设置为 0 以创建一个控件数组。然后按 F5 键并移动滚动条。

```
PrivateSubForm_Load()
   '移动第一个 Shape 控件并其大小。
   Shape1(0).MoveHScroll1.Left,HScroll1.Top*1.5,HScroll1.Width,H
Scroll1.Height
   Label1.Caption=""                          '设置 Label 的标题。
   LoadShape1(1)                              '创建第二个 Shape。
   '移动第二个 Shape 控件并定制其大小。
   Shape1(1).MoveShape1(0).Left,Shape1(0).Top,1,Shape1(0).Height
   Shape1(1).BackStyle=1                      ' 设 置  BackStyle  为
Opaque。
   Shape1(1).Visible=True                     '显示第二个 Shape。
```

```
    HScroll1.Min=1                          '设置滚动条的值
    HScroll1.Max=HScroll1.Width
EndSub

PrivateSubHScroll1_Change()
    Label1.Caption="Changed"                '显示改变后的信息
EndSub

PrivateSubHScroll1_Scroll()
    Shape1(1).BackColor=&HFF0000        '设置 Shape 的颜色为蓝色
    Label1.Caption="Changing"               '显示滚动时的信息
    Shape1(1).Width=HScroll1.Value          '按 ScrollValue 调整
Shape 的尺寸
EndSub
```

ScrollBars 属性

返回或设置一个值，该值指示一个对象是有水平滚动条还是有垂直滚动条。在运行时是只读的。

应用于

UserDocument 对象，MDIForm 对象，TextBox 控件

语法

object.**ScrollBars**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

设置值

对于 MDIForm 对象，ScrollBars 属性的设置值为：

设置值	描述
True	（缺省值）窗体有一个水平滚动条或垂直滚动条，或两者都有
False	窗体没有滚动条

对于 TextBox 控件，ScrollBars 属性的设置值为：

常数	设置值	描述
vbSBNone	0	（缺省值）无
vbHorizontal	1	水平
vbVertical	2	垂直
vbBoth	3	两种

说明

对于 ScrollBars 的属性设置值为 1（水平）、2（垂直）、或 3（两种）的 TextBox 控件，必须将 MultiLine 属性设置为 True。

在运行时，MicrosoftWindows 操作环境自动地实现一个标准键盘界面以允许在 TextBox 控件中使用方向（上箭头、下箭头、左箭头、和右箭头）、HOME 和 END 键等来定位。

滚动条只在对象的内容超过对象的边框时才被显示在对象上。例如，在 MDIForm 对象中，如果子窗体的一部分被隐藏在父 MDI 窗体的边框之后，那么一个水平滚动条（HScrollBar 控件）将被显

示。该情况的例外是总显示滚动条的 TextBox 控件。如果 ScrollBars 被设为 False，那么对象将没有滚动条，而不管其内容如何。

请参阅

WordWrap 属性

Second 函数

返回一个 Variant(Integer)，其值为 0～59 之间的整数，表示一分钟内的某个秒。

语法

Second(*time*)

必要的 time 参数，可以是任何能够表示时刻的 Variant、数值表达式、字符串表达式或它们的组合。如果 time 包含 Null，则返回 Null。

请参阅

Day 函数，Hour 函数，Minute 函数，Now 函数，Time 函数，Time 语句

示例

本示例使用 Second 函数转换指定的时间，得到分钟后面的秒数。在开发环境中，日期和时间原义会根据系统的地区设置，以短式日期和时间格式显示。

```
DimMyTime,MySecond
MyTime=#4:35:17PM# '指定一时间。
```

```
MySecond=Second(MyTime)      'MySecond 的值为包含 17。
```

Section 对象（数据报表设计器）

Section 对象代表数据报表设计器的一个部分。

语法

Section

说明

必须使用标准的集合语法检索对数据报表设计器中一个 Section 对象和任何控件的引用。下面的示例检索 Section 对象第二部分上第三个控件的名称：

```
Debug.PrintDataReport1.Sections(2).Controls(3).Name
```

属性

ForcePageBreak 属性，KeepTogether 属性，Controls 属性

请参阅

DataReport 对象，Sections 属性

Sections 集合（数据报表设计器）

Section 对象的集合。

语法

Sections

说明

必须使用标准的集合语法检索对数据报表设计器中一个 Section 对象和任何控件的引用。

请参阅

Sections 属性

示例

本例打印在数据报表设计程序的每个 Section 对象名称，也打印该节的每个控件名称。要试本例创建一个数据报表，把 CommandButton 控件放入报表中，把代码粘贴到代码模块的声明段，按 F5 运行项目并单击按钮。

```
PrivateSubCommand1_Click()
Dimsect, ct1
ForEachsectInDataReport1.Sections
Debug.Print"Section", sect.Name
ForEachctlInsect.Controls
Debuq.Print.ctl.Name
Nextctl
Nextsect
EndSub
```

示例

本例打印数据报表设计器中每一 Section 对象以及该部分中每一控件的名称。要试验该例，创建一个数据报表。在窗体上安置一个 CommandButton 控件，并把代码粘贴到代码模块的声明部分。按 F5 键运行工程，并单击该按钮。

```
PrivateSubCommand1_Click()
    Dimsect,ctl
    ForEachsectInDataReport1.Sections
        Debug.Print"Section",sect.Name
        ForEachctlInsect.Controls
            Debug.Print,ctl.Name
        Nextctl
    Nextsect
EndSub
```

Sections 属性

返回一个对 Sections 集合的引用。

应用于

DataReport 对象

语法

object.**Sections**

object.**Sections**(*index*)

Sections 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
index	可选的。一个部分的索引或名称

说明

这一属性使用集合语法指定要返回的是集合还是集合的一个成员。

请参阅

Section 对象(数据报表设计器)，Sections 集合(数据报表设计器)

示例

本例打印在数据报表设计程序的每个 Section 对象名称，也打印该节的每个控件名称。要试本例创建一个数据报表，把 CommandButton 控件放入报表中，把代码粘贴到代码模块的声明段，按 F5 运行项目并单击按钮。

```
PrivateSubCommand1_click()
Dimsect, ctl
ForEachsectInDataReport1.Sections
Debug.Print“Section”, sect.Name
ForEachctlInsect.Controls
Debug.Print, ctl.Name
```

Nextctl

Nextsect

EndSub

Seek 函数

返回一个 Long，在 Open 语句打开的文件中指定当前的读/写位置。

语法

Seek(*filenumber*)

必要的 filenumber 参数是一个包含有效文件号的 Integer。

说明

Seek 函数返回介于 1 和 2，147，483，647（相当于 2^31–1）之间的值。

对各种文件访问方式的返回值如下：

方式	返回值
Random	下一个读出或写入的记录号

Binary,Output 下一个操作发生的字节位置。文件的第一个字节在位置 1Append,Input 第二个字节在位置 2，等等

请参阅

Get 语句，Loc 函数，Open 语句，Put 语句，Seek 语句

示例

本示例使用 Seek 函数来返回当前文件位置。示例中假设 TESTFILE 文件内含有用户自定义数据类型 Record 的记录。

```
TypeRecord '定义用户自定义数据类型。
    IDAsInteger
    NameAsString*20
EndType

如果以随机方式打开文件，Seek 返回下一个记录的编号。
DimMyRecordAsRecord     '声明变量。
Open"TESTFILE"ForRandomAs#1Len=Len(MyRecord)
DoWhileNotEOF(1)              '循环至文件尾。
    Get#1,,MyRecord           '读入下一个记录。
    Debug.PrintSeek(1)            '在立即窗口中显示记录号。
Loop
Close#1                           '关闭文件。
```

如果不以 Random 方式打开文件，则 Seek 返回下一个操作会发生的位置。假设 TESTFILE 文件内含有文本数据。

```
DimMyChar
Open"TESTFILE"ForInputAs#1          '打开输入文件。
DoWhileNotEOF(1)                    '循环至文件尾。
    MyChar=Input(1,#1)                  '读入下一个字符。
    Debug.PrintSeek(1)                  '将下一字符的位置显示在立
即窗口。
```

```
Loop
Close#1                                    '关闭文件。
```

Seek 语句

在 Open 语句打开的文件中，设置下一个读/写操作的位置。

语法

Seek[#]*filenumber*,*position*

Seek 语句的语法具有以下几个部分：

部分	描述
filenumber	必需的。任何有效的文件号
position	必需的。介于 1–2，147，483，647 之间的数字，指出下一个读写操作将要发生的位置

说明

在 Get 及 Put 语句中指定的记录号将覆盖由 Seek 语句指定的文件位置。

若在文件结尾之后进行 Seek 操作，则进行文件写入的操作会把文件扩大。如果试图对一个位置为负数或零的文件进行 Seek 操作，则会导致错误发生。

请参阅

Get 语句，Loc 函数，Open 语句，Put 语句，Seek 函数

示例

本示例使用 Seek 语句在文件内设置下一次读写的位置。示例中假设 TESTFILE 文件内含有用户自定义数据类型 Record 的记录。

```
TypeRecord '定义用户自定义数据类型。
    IDAsInteger
    NameAsString*20
EndType
```

如果以随机方式打开文件，Seek 将读写位置设置到下一个记录。

```
DimMyRecordAsRecord,MaxSize,RecordNumber          '声明变量。
'以随机文件方式打开文件。
Open"TESTFILE"ForRandomAs#1Len=Len(MyRecord)
MaxSize=LOF(1)\Len(MyRecord)                      '取得文件中的记录的数。
'用循环读入所有记录，但是从最后的记录开始往前读。
ForRecordNumber=MaxSizeTo1Step-1
    Seek#1,RecordNumber                           '设置读写位置。
    Get#1,,MyRecord                               '读入一个记录。
NextRecordNumber
Close#1                                           '关闭文件。
```

如果不以 Random 打开文件，则 Seek 设置下一个操作发生的位置。假设 TESTFILE 文件内含有文本数据。

```
DimMaxSize,NextChar,MyChar
Open"TESTFILE"ForInputAs#1                        '打开输入文件。
MaxSize=LOF(1)                                    '取得文件的总字符数。
```

```
'用循环读入所有记录，但是从最后的记录开始往前读。
ForNextChar=MaxSizeTo1Step-1
   Seek#1,NextChar                          '设置读写位置。
   MyChar=Input(1,#1)                       '读入一字符。
NextNextChar
Close#1                                     '关闭文件。
```

SelChange 事件

当前范围改变为不同的单元或单元的范围时发生。

语法

```
PrivalteSubDataGrid_SelChange([cancelAsInteger])
PrivateSubObject_SelChange()
```

SelChange 事件语法有这几部分：

部分描述

object 一个对象表达式，在“应用于”列表中表示对象

cancel 决定是否选择恢复事件以前发生的位置

说明

当用户无论在何时单击一个单元而不是当前单元和当用户拖动选择单元的新范围时，SelChange 事件发生。用户也可按 Shift 键和箭头键来选择单元范围，可以使用 SelStartCol 和 SelEndCol 属性改变选择的区域来触发 DataGrid 控件代码中的事件。

当用户拖动选择跨过 DataGrid 控件或当用户不移动单元而编程

改变选择时，用户单击新单元但不发生时 RowColChane 事件也发生。

在 DataGrid 控件中设定 cancel 为 True,导致选择恢复到单元或以前事件发生的活动范围。

请参阅

RowColChange 事件，SelEndCol,SelStartCol,SelEndRow,SelStartRow 属性

SelCount 属性

返回在 ListBox 控件中被选中项的数量。

应用于

ListBox 控件

语法

object.**SelCount**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

如果没有项被选中，那么 SelCount 属性将返回 0 值。否则，它返回当前被选中的列表项的数量。该属性对能够作复选是尤其有用的。

请参阅

AddItem 方法，RemoveItem 方法，List 属性，ListIndex 属性，

MultiSelect

属性，NewIndex 属性，Selected 属性，TopIndex 属性

SelectCase 语句

根据表达式的值，来决定执行几组语句中的其中之一。

语法

SelectCase*testexpression*

Case*expressionlist-n*

[statements-*n*]]...

CaseElse

[*elsestatements*]]

EndSelect

SelectCase 语句的语法具有以下几个部分：

部分	描述
Testexpression	必要参数。任何数值表达式或字符串表达式
Expressionlist-n	如果有 Case 出现，则为必要参数。其形式为 expression ， expressionToexpression ， Iscomparisonoperatorexpression 的一个或多个组成的分界列表。To 关键字可用来指定一个数值范围。如果使用 To 关键字，则较小的数值要出现在 To 之前。使用 Is 关键字时，则可以配合比较运算符（除 Is 和 Like 之外）来指定一个数值范围。如果没有提供，则 Is 关键字会被自动插入
statements-n	可选参数。一条或多条语句，当 testexpression 匹配 expressionlist-n 中的任何部分时执行
elsestatements	可选参数。一条或多条语句，当 testexpression 不匹配 Case 子句的任何部分时执行

说明

如果 testexpression 匹配某个 Caseexpressionlist 表达式，则在 Case 子句之后，直到下一个 Case 子句的 statements 会被执行；如果是最后一个子句，则会执行到 EndSelect。然后控制权会转移到 EndSelect 之后的语句。如果 testexpression 匹配一个以上的 Case 子句中的 expressionlist 表达式，则只有第一个匹配后面的语句会被执行。

CaseElse 子句用于指明 elsestatements，当 testexpression 和所有的 Case 子句中的 expressionlist 都不匹配时，则会执行这些语句。虽然不是必要的，但是在 SelectCase 区块中，最好还是加上 CaseElse 语句来处理不可预见的 testexpression 值。如果没有 Caseexpressionlist 匹配 testexpression，而且也没有 CaseElse 语句，则程序会从 EndSelect 之后的语句继续执行。

可以在每个 Case 子句中使用多重表达式或使用范围，例如，下面的语句是正确的：

```
Case1To4,7To9,11,13,Is>MaxNumber
```

注意 Is 比较运算符和使用在 SelectCase 语句中的 Is 关键字并不相同。

也可以针对字符串指定范围和多重表达式。在下面的例子中，Case 所匹配的字符串为：等于 everything、按英文字母顺序落入从 nuts 到 soup 之间的字符串、以及 TestItem 所代表的当前值。

```
Case"everything","nuts"To"soup",TestItem
```

SelectCase 语句也可以是嵌套的。但每个嵌套的 SelectCase 语句必须要有相应的 EndSelect 语句。

请参阅

Choose 函数，End 语句，If...hen...lse 语句，On...GoSub，On...GoTo

示例

本示例使用 SelectCase 语句来判断变量的值。示例中第二个 Case 子句包含了变量值，故只有此区块内的语句会被完成。

```
DimNumber
Number=8                    '设置变量初值。
SelectCaseNumber                '判断 Number 的值。
Case1To5                    'Number 的值在 1～5 之间，包含 1 和 5。
   Debug.Print"Between1and5"
'下一个 Case 子句是本示例中唯一判断值为 True 的子句。
Case6,7,8                   'Number 的值在 6～8 之间。
   Debug.Print"Between6and8"
Case9 到 10                 'Number 的值为 9 或 10。
Debug.Print"Greaterthan8"
CaseElse                    '其他数值。
   Debug.Print"Notbetween1and10"
EndSelect
```

SelectAll 方法

选定某个窗体上包含的所有控件。

应用于

VBForm 对象

语法

object.**SelectAll**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Selected 属性

返回或设置在 FileListBox 或 ListBox 控件中的一个项的选择状态。该属性是一个与 List 属性一样、有相同项数的布尔值数组。在设计时是不可用的。

应用于

FileListBox 控件，ListBox 控件

语法

object.**Selected**(*index*)[=*boolean*]

Selected 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
index	控件中项的索引号
boolean	一个用来指定项是否被选中的布尔表达式，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
True	表示此项被选中
False	（缺省值）表示项没有被选中

说明

此属性对能够复选是尤其有用的。可以快速检查在列表中哪些项

已被选中。也可以从代码中使用该属性选中或取消选中列表中的一些项。

如果 MultiSelect 属性被设置为 0，那么可以使用 ListIndex 属性来获得选中项的索引尽管如此，在复选中，ListIndex 属性返回的是包含在焦点矩形框内的项的索引，而不管该项是否真正被选中。

如果 ListBox 控件的 Style 属性设置为 1（复选框），那么 Selected 属性只对其复选框被选中的项返回 True。Selected 属性对那些只是显示为高亮度的项不返回 True。

请参阅

AddItem 方法，Clear 方法(粘贴板，ComboBox，ListBox)，RemoveItem 方法，List 属性，ListCount 属性，ListIndex 属性，MultiSelect 属性，NewIndex 属性，TopIndex 属性

SelectedControls 集合

一个集合，允许对对象中当前选定的所有控件进行访问。

语法

SelectedControls(*index*)

index 所在处表示从 0 到 SelectedControls.Count–1 的整数。

请参阅

SelectedControls 属性

SelectedControls 属性

返回包含当前在窗体上选定的控件的集合。在创建属性页时，SelectedControls 属性不可用，在属性页运行时，该属性是只读的。

应用于

PropertyPage 对象

语法

object.**SelectedControls**

SelectedControls 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

当属性页要判断哪些控件是当前选定的，因此哪些控件可能需要改变属性时，此集合十分有用。有些容器只允许一次选定一个控件；在这种情况下 SelectedControls 将只包含一个控件。有些容器允许一次选定多个控件；这时可能有多个选定的控件，因此属性页必须在 SelectedControls 集合的控件中重复迭代，并试图设置变更的属性。应该编写适当的错误处理例程，以处理下面两种情况：集合中的某个控件没有变更的属性，或者设置属性时控件产生了错误。

请参阅

Controls 集合，Count 属性(VB 集合)

SelectedVBComponent 属性

返回一个被选择的单元，此属性为只读。

应用于

VBE 对象

说明

SelectedVBComponent 属性返回工程窗口中被选择的部件，如果在 Project 窗口中被选择的项不是一个部件，SelectedVBComponent 返回 Nothing。

请参阅

VBComponent 对象，VBComponents 集合，ActiveVBProject 属性

示例

下列示例使用 SelectedVBComponent 属性返回被选择的部件。

```
Debug.PrintApplication.VBE.SelectedVBComponent.Name
```

SelectedVBControls 集合

返回当前在部件上选定的控件集合。

语法

SelectedVBControls

说明

可用该集合访问窗体上所有当前选定的控件。代码可逐步遍历控件集合，或请求一个特定控件。

除了未实现 Add 方法这一点之外，对该集所作的说明与 VBControls 集合相同。对于 SelectedVBControls 集合，缺省方法是 Item 方法，并用整数索引。

该集合取代了 VisualBasic4.0 版中的 SelectedControlTemplates 集合。

属性

Count 属性(VB 集合)，Parent 属性，VBE 属性

方法

Item 方法，Copy 方法

请参阅

SelectedVBControlsEvents 对象，SelectedVBControlsEvents 属性

SelectedVBControls 属性

该属性返回一个窗体中选定控件的集合。

语法

object.**SelectedVBControls**

object 所在处代表对象表达式，其值是“应用于”列表中的对象

SelectedVBControlsEvents 对象

表示由所有当前选定的控件所支持的事件的源。

语法

SelectedVBControlsEvents

事件

ItemAdded 事件，ItemRemoved 事件

请参阅

SelectedVBControls 集合，SelectedVBControlsEvents 属性

SelectedVBControlsEvents 属性

该属性返回当前在窗体上被选择的控件所支持的全部事件。

应用于

Events 对象

语法

object.**SelectedVBControlsEvents**(*vbproject***As Variant**)

SelectedVBControlsEvents **属性的语法包含下面部分：**

部分	**描述**
object	**对象表达式，其值是“应用于”列表中的对象**
vbproject	**指定包含窗体和控件的工程的** variant **表达式**

说明

该属性返回类型 SelectedVBControlsEvents **的事件对象。这一事件源于** VBForm。

SelectionChanged 事件

当窗体上控件的选定变更时，发生该事件。

应用于

PropertyPage 对象

语法

Sub*object*_SelectionChanged()

SelectionChanged 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

该事件的发生通知属性页控件的选定已经变更，因此需要相应更新当前属性值的显示。此时，应读取 SelectedControls 属性以找出选定控件的新集合。

为控件第一次提出属性页时，也将产生 SelectionChanged 事件。

请参阅

SelectedContrlos 属性

SelEndCol、SelStartCol、SelEndRow、SelStartRow 属性

为一些连续的单元的范围返回或设置第一或最后的行或列。在设计时不可用。

SelEndCol一右边最后被选中的列。

SelStartCol一左边第一个被选中的列。

SelEndRow一最后被选中的行。

SelStartRow一第一个被选中的行。

语法

object.**SelEndCol**[=*value*]

object.**SelStartCol**[=*value*]

object.**SelEndRow**[=*value*]

object.**SelStartRow**[=*value*]

SelEndCol、SelStartCol、SelEndRow 和 SelStartRow 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定第一或最后一行或列的数值表达式

说明

可以从代码中使用这些属性来选择 DataGrid 控件的特定区域，或者在代码中返回所选择区域的大小。

SelStartCol 和 SelStartRow 一起使用可指定选中范围左上角处的单元。SelEndCol 和 SelEndRow 一起使用可指定选中范围右下角处的单元。

要指定一个单元而不改变当前的选择，使用 Col 和 Row 属性。

SelStartCol 和 SelEndCol 的缺省值是-1。

请参阅

Col，Row 属性

SelLength、SelStart、SelText 属性

SelLength—返回或设置所选择的字符数。

SelStart—返回或设置所选择的文本的起始点；如果没有文本被选中，则指出插入点的位置。

SelText—返回或设置包含当前所选择文本的字符串；如果没有字符被选中，则为零长度字符串("")。

这些属性在设计时是不可用的。

应用于

Slider 控件，DataCombo 控件

语法

object.**SelLength**[=*number*]

object.**SelStart**[=*index*]

object.**SelText**[=*value*]

SelLength、SelStart、和 SelText 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个用来指定被选择字符数的数值表达式。对于 SelLength 和 SelStart，设置值的有效范围是 0 到文本长度－在 ComboBox 或 TextBox 控件编辑区中字符的总数
index	一个用来指定所选择文本的起始点的数值表达式，“设置值”中有详细描述
value	包含所选择文本的字符串表达式

说明

为下面这些任务应使用这些属性，如设置插入点、建立插入范围、在控件中选择子串、或清除文本等。与 Clipboard 对象联合使用，这些属性对于复制、剪切、和粘贴操作是很有用的。

当使用这些属性时：

SelLength 的设置比 0 小会导致一个运行时错误。

SelStart 的设置比文本长度大，会使该属性设置为现有文本长度；

SelStart 的改变将使选择改变到插入点并将 SelLength 设置为 0。

SelText 的设置为新值，会将 SelLength 设置为 0 并用新字符串代替所选择的文本。

请参阅

Slider 控件，SelectRange 属性，ClearSel 方法，SelText 属性

(掩码编辑控制器)，Text 属性，ActiveControl 属性，ActiveForm 属性

示例

本例能够指定需要查找的文本，然后查找该文本并在找到后选中它。要试用此例，先将下面的代码粘贴到包含一个宽 TextBox 控件窗体的声明部分，然后按 F5 键并单击窗体。

```
PrivateSubForm_Load()
   Text1.Text="Twoofthepeakhumanexperiences"
   Text1.Text=Text1.Text&"aregoodfoodandclassicalmusic."
EndSub
PrivateSubForm_Click()
   DimSearch,Where                    '声明变量。
   '获取需要查找的字符串。
   Search=InputBox("Entertexttobefound:")
   Where=InStr(Text1.Text,Search)          '在文本中查找字符串。
   IfWhereThen                        '如果找到，
      Text1.SelStart=Where-1          '设置选定的起始位置并
      Text1.SelLength=Len(Search)        '设置选定的长度。
   Else
      MsgBox"Stringnotfound."         '给出通知。
   EndIf
EndSub
```

本例显示如何在剪切、复制、粘贴、和删除操作中使用 Clipboard 对象。要试用此例，创建一个带有 TextBox 控件的窗体并使用“菜单编辑器”来创建一个“编辑”菜单（对于每个命令，将其 Caption 属性分别设置为 Cut、Copy、Paste 和 Delete，将其 Name 属性分别设置为 EditCut、EditCopy、EditPaste 和 EditDelete）。

```
PrivateSubEditCut_Click()
    '清除剪贴板上的内容。
Clipboard.Clear
    '复制选中的文本到剪贴板上。
    ClipBoard.SetTextScreen.ActiveControl.SelText
    '删除选中的文本。
    Screen.ActiveControl.SelText=""
EndSub

PrivateSubEditCopy_Click()
    '清除剪贴板上的内容。
Clipboard.Clear
    '复制选中的文本到剪贴板上。
    ClipBoard.SetTextScreen.ActiveControl.SelText
EndSub

PrivateSubEditPaste_Click()
    '从剪贴板上将文本放置到活动控件中。
```

```
    Screen.ActiveControl.SelText=ClipBoard.GetText()
EndSub

PrivateSubEditDelete_Click()
'删除选中的文本。
Screen.ActiveControl.SelText=""
EndSub
```

SendKeys 语句

将一个或多个按键消息发送到活动窗口，就如同在键盘上进行输入一样。

语法

SendKeys*string*[,*wait*]

SendKeys **语句的语法具有以下几个命名参数：**

部分	**描述**
string	**必需的。字符串表达式，指定要发送的按键消息**
wait	**可选的。指定等待方式的值。如果为** False**（缺省值），则控件在按键发送出去之后立刻返回到过程。如果为** True**，则按键消息必须在控件返回到过程之前加以处理**

说明

每个按键由一个或多个字符表示。为了指定单一键盘字符，必须

按字符本身的键。例如，为了表示字母 A，可以用"A"作为 string。为了表示多个字符，就必须在字符后面直接加上另一个字符。例如，要表示 A、B 及 C，可用"ABC"作为 string。

对 SendKeys 来说，加号(+)、插入符(^)、百分比符号(%)、上划线(~)及圆括号()都具有特殊意义。为了指定上述任何一个字符，要将它放在大括号({})当中。例如，要指定正号，可用{+}表示。方括号([])对 SendKeys 来说并不具有特殊意义，但必须将它们放在大括号中。在其它应用程序中，方括号有特殊意义，在出现动态数据交换(DDE)的时候，它可能具有重要意义。为了指定大括号字符，请使用{{}及{}}。

为了在按下按键时指定那些不显示的字符，例如 ENTERRETURN 或 TAB 以及那些表示动作而非字符的按键，请使用下列代码：

按键	代码
BACKSPACE	{BACKSPACE},{BS},或{BKSP}
BREAK	{BREAK}
CAPSLOCK	{CAPSLOCK}
DELorDELETE	{DELETE}或{DEL}
DOWNARROW	{DOWN}
END	{END}
ENTERRETURN	{ENTER}或~
ESC	{ESC}
HELP	{HELP}
HOME	{HOME}

INSorINSERT	{INSERT}或{INS}
LEFTARROW	{LEFT}
NUMLOCK	{NUMLOCK}
PAGEDOWN	{PGDN}
PAGEUP	{PGUP}
PRINTSCREEN	{PRTSC}
RIGHTARROW	{RIGHT}
SCROLLLOCK	{SCROLLLOCK}
TAB	{TAB}
UP	{UP}
F1	{F1}
F2	{F2}
F3	{F3}
F4	{F4}
F5	{F5}
F6	{F6}
F7	{F7}
F8	{F8}
F9	{F9}
F10	{F10}
F11	{F11}
F12	{F12}

F13	{F13}
F14	{F14}
F15	{F15}
F16	{F16}

为了指定那些与 SHIFT、CTRL 及 ALT 等按键结合的组合键，可在这些按键码的前面放置一个或多个代码，这些代码列举如下：

按键	代码
SHIFT	+
CTRL	^
ALT	%

为了说明在按下其它按键时应同时按下 SHIFT、CTRL、及 ALT 的任意组合键，请把那些按键的码放在括号当中。例如，为了说明按下 E 与 C 的时候同时按下 SHIFT 键，请使用"+(EC)"。为了说明在按下 E 的时候同时按下 SHIFT 键，但接着按 c 而不按 SHIFT，则使用"+EC"。

为了指定重复键，使用{keynumber}的形式。必须在 key 与 number 之间放置一个空格。例如，{LEFT42}意指 42 次按下 LEFTARROW 键；{h10}则是指 10 次按下 h 键。

注意不能用 SendKeys 将按键消息发送到这样一个应用程序，这个应用程序并没有被设计成在 MicrosoftWindowsorMacintosh 中运行。Sendkeys 也无法将 PRINTSCREEN 按键{PRTSC}发送到任何应用程序。

请参阅

DoEvents 函数，AppActivate 语句

示例

本示例使用 Shell 函数来运行 MicrosoftWindows 所附的计算器程序；然后使用 SendKeys 语句来按下计算器的某些数字键，最后退出计算器。（若要观察示例运行过程，可将示例粘贴到过程中，再运行过程即可。因为 AppActivate 会将焦点转移到计算器应用程序，故本示例不能以单步方式来运行。）。在 Macintosh 上，使用 Macintosh 应用程序接受键盘输入而不是用 Windows 计算器。

```
DimReturnValue,I
ReturnValue=Shell("CALC.EXE",1)              '运行计算器。
AppActivateReturnValue                       '激活计算器。
ForI=1To100                                  '设置计数循环。
   SendKeysI&"{+}",True                      '按下按键给计算器
NextI                                        '将所有 I 值相加。
SendKeys"=",True                             '取得总合。
SendKeys"%{F4}",True                         '按 ALT+F4 关闭计算器。
```

SerialNumber 属性

返回用于唯一标识磁盘卷标的十进制序列号。

应用于

Drive 对象

语法

object.**SerialNumber**

object 总是一个 Drive 对象。

说明

可以使用 SerialNumber 属性确保正确的磁盘已插入到某个带有可删除媒体的驱动器中。

下面的代码举例说明了 SerialNumber 属性的用法：

```
SubShowDriveInfo(drvpath)
Dimfs,d,s,t
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(fs.GetDriveName(fs.GetAbsolutePathName(drvpath)
))
SelectCased.DriveType
Case0:t="Unknown"
Case1:t="Removable"
Case2:t="Fixed"
Case3:t="Network"
Case4:t="CD-ROM"
Case5:t="RAMDisk"
EndSelect
s="Drive"&d.DriveLetter&":-"&t
s=s&vbCrLf&"SN:"&d.SerialNumber
MsgBoxs
EndSub
```

请参阅

AvailableSpace 属性，DriveLetter 属性，DriveType 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，Path 属性(文件系统对象对象)，RootFolder 属性，ShareName 属性，TotalSize 属性，VolumeName 属性

Server 属性

返回活动服务器页面的 Server 对象。

应用于

WebClass 对象

语法

object.Server

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

WebClass 使用 Server 对象创建其他的对象，并决定可能影响其处理的指定服务器的属性。当创建要存储在 Session 或 Application 对象中的对象时，必须用 Server.CreateObject，而不用 VisualBasic 的 New 关键字或 CreateObject 函数。

关于 Server 对象的属性、方法、和事件的详细信息，请参阅活动服务器页面文档。

请参阅

《MicrosoftVisualBasic6.0 工具部件指南》第五部分“开发 IIS 应用程序”，第三章“ IIS 应用程序对象模型 ”的“ 构造 Internet 应用程序 ”。

Session 属性

返回活动服务器页面的 Session 对象。

应用于

WebClass 对象

语法

object.**Session**

object 所在处表示一个对象表达式，其值是“ 应用于 ”列表中的一个对象。

说明

WebClass 使用 Session 对象维护有关当前用户对话的信息，并有可能存储和检索状态信息。 Session 对象可以用来在不同的 WebClasses 以及 WebClasses 和活动服务器页面之间维护状态。

关于 Session 对象的属性、方法、和事件的详细信息，请参阅活动服务器页面文档。

请参阅

《MicrosoftVisualBasic6.0 部件工具指南》“构造 Internet 应用程序 ”第五部分“开发 IIS 应用程序”，第三章中“ IIS 应用程序对象模型 ”的“ 对象中的存储状态 ”。

Set 语句

将对象引用赋给变量或属性。

语法

Set*objectvar*={[**New**]*objectexpression*|**Nothing**}

Set **语句的语法包含下面部分：**

部分	描述
objectvar	必需的。变量或属性的名称，遵循标准变量命名约定
new	可选的。通常在声明时使用 New，以便可以隐式创建对象。如果 New 与 Set 一起使用，则将创建该类的一个新实例。如果 objectvar 包含了一个对象引用，则在赋新值时释放该引用。不能使用 New 关键字来创建任何内部数据类型的新实例，也不能创建从属对象
objectexpression	必需的。由对象名，所声明的相同对象类型的其它变量，或者返回相同对象类型的函数或方法所组成的表达式
nothing	可选的。断绝 objectvar 与任何指定对象的关联。若没有其它变量指向 objectvar 原来所引用的对象，将其赋为 Nothing 会释放该对象所关联的所有系统及内存资源

说明

为确保合法，objectvar 必须是与所赋对象相一致的对象类型。Dim、Private、Public、ReDim 以及 Static 语句都只声明了引用对象的变量。在用 Set 语句将变量赋为特定对象之前，该变量并没有引用任何实际的对象。

下面的示例说明了如何使用 Dim 来声明 Form1 类型的数组。Form1 实际上还没有实例。然后使用 Set 将新创建的 Form1 实例的引用赋给 myChildForms 变量。在 MDI 应用程序中可以使用这些代码来创建子窗体。

```
DimmyChildForms(1to4)AsForm1
SetmyChildForms(1)=NewForm1
SetmyChildForms(2)=NewForm1
SetmyChildForms(3)=NewForm1
SetmyChildForms(4)=NewForm1
```

通常，当使用 Set 将一个对象引用赋给变量时，并不是为该变量创建该对象的一份副本，而是创建该对象的一个引用。可以有多个对象变量引用同一个对象。因为这些变量只是该对象的引用，而不是对象的副本，因此对该对象的任何改动都会反应到所有引用该对象的变量。不过，如果在 Set 语句中使用 New 关键字，那么实际上就会新建一个该对象的实例。

请参阅

Dim 语句，Let 语句，Private 语句，Public 语句，ReDim 语句，Static 语句，=操作符

示例

该示例使用 Set 语句将对象引用赋给变量。假定 YourObject 指向一个具有 Text 属性的合法对象。

```
DimYourObject,MyObject,MyStr
SetMyObject=YourObject                    '对象引用赋值。
'MyObject 和 YourObject 引用同一个对象。
YourObject.Text="HelloWorld"              '初始化属性。
MyStr=MyObject.Text                       '返回 HelloWorld"。

'脱离关联。MyObject 不再引用 YourObject。
SetMyObject=Nothing                       '释放该对象。
```

SetAttr 语句

为一个文件设置属性信息。

语法

SetAttr*pathname,attributes*

SetAttr 语句的语法含有以下这些命名参数：

部分	描述
pathname	必需的。用来指定一个文件名的字符串表达式，可能包含目录或文件夹、以及驱动器
attributes	必需的。常数或数值表达式，其总和用来表示文件的属性

设置值

attributes 参数设置可为：

常数	值	描述
vbNormal	0	常规（缺省值）
vbReadOnly	1	只读
vbHidden	2	隐藏
vbSystem	4	系统文件在 Macintosh 中不可用
vbArchive	32	上次备份以后，文件已经改变

注意这些常数是由 VBA 所指定的，在程序代码中的任何位置，可以使用这些常数来替换真正的数值。

说明

如果想要给一个已打开的文件设置属性，则会产生运行时错误。

请参阅

GetAttr 函数，FileAttr 函数

示例

本示例使用 SetAttr 语句来设置文件属性。

```
SetAttr"TESTFILE",vbHidden    '设置隐含属性。
SetAttr"TESTFILE",vbHidden+vbReadOnly     '设置隐含并只读。
```

SetAutoServerSettings 方法

为了满足远程自动化的需求，设置远程自动化注册值，包括远程服务器访问的设置。

语法

object.**SetAutoServerSettings**(*remote*,[*progid*],[*clsid*],[*servername*],[*protocol*],[*authentication*])

SetAutoServerSettings 方法的语法有这些部分：

部分	描述
object	必需的。对象表达式，其值为应用于 ” 列表中的一个对象
remote	必需的。布尔值。服务器是远程的为 True，本地的为 False
progid	可选的。variant 表达式，指定服务器的 ProgID
clsid	可选的。variant 表达式，指定服务器的 CLSID
servername	可选的。variant 表达式，指定服务器的机器名
protocol	可选的。variant 表达式，指定使用的协议的 RPC 名
Authentication	可选的。variant 表达式，指定 RPC 身份验证级别

返回值

SetAutoServerSettings 方法返回下列错误代码：

值	描述
0	没有错误
1	未知的运行时错误发生
2	未指定协议
3	未指定服务器机器名
4	读注册表错误
5	写注册表错误
6	ProgID 和 CLSID 参数丢失
7	没有本地服务器（无论是进程中的，还是跨进程的；16 位的还是 32 位的）
8	有个寻找 ProxyDLLs 的错误,请检查它们是否安装的属性

说明

SetAutoServerSettings 方法是取 CLSID 值还是取 ProgID 值，是把注册表信息设置为本地还是远程，取决于 remote 参数。如果既有 CLSID 又有 ProgID，则先取 CLSID。

示例

这个示例将"Hello"服务器从本地注册切换到远程注册，再从远程注册切换到本地注册：

```
SubSwitchHello()
    DimoRegClassAsNewRegClass
    '将 Hello 注册到调用 Server1 的机器上远程运行。
```

```
    oRegClass.SetAutoServerSettingsTrue,_
"HelloProj.HelloClass",1_
    ServerName:="Server1",Protocol:="ncacn_ip_tcp"
    '再将 Hello 注册为本地运行。
    oRegClass.SetAutoServerSettingsFalse,_
"HelloProj.HelloClass"
EndSub
```

SetData 方法

用以使用指定的图形格式将图片放到 Clipboard 对象上。不支持命名参数。

应用于

Clipboard 对象

语法

object.**SetData***data*,*format*

SetData 方法的语法包含下列部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
data	必需的。被放置到 Clipboard 对象中的图形
format	可选的。一个常数或数值，按照下列“设置值”中的描述，指定 VisualBasic 识别的 Clipboard 对象格式。如果省略 format，则 SetData 自动决定图形格式

设置值

用于 format 的设置值有：

常数	值	描述
vbCFBitmap	2	位图（.bmp 文件）
vbCFMetafile	3	元文件（.wmf 文件）
vbCFDIB	8	与设备无关的位图(DIB)
vbCFPalette	9	调色板

说明

上述常数在对象浏览器中的 VisualBasic(VB)对象库里列出。

使用 LoadPicture 函数或 Form、Image 或 PictureBox 的 Picture 属性来建立将放置到 Clipboard 对象中的图形。

请参阅

GetData 方法，GetForm 方法，GetText 方法，SetText 方法

SetData 方法（DataObject 对象）

用指定的数据格式把数据插入 DataObject 对象。

应用于

DataObject 对象

语法

object.**SetData**[*data*],[*format*]

SetData 方法语法包含下面部分：

部分	描述
object	必需的。对象表达式，其值是“应用于”列表中的一个对象
data	可选的变体型，包含要传送到 DataObject 对象的数据
format	可选的常数或值，规定所传送数据的格式，如“设置值”中所述

设置值

format 设置如下：

常数	值	描述
vbCFText	1	文本（.txt 文件）
vbCFBitmap	2	位图（.bmp 文件）
vbCFMetafile	3	元文件（.wmf 文件）
vbCFEMetafile	14	增强元文件（.emf 文件）
vbCFDIB	8	设备无关位图(DIB)
vbCFPalette	9	调色板
vbCFFiles	15	文件列表
vbCFRTF	-16639	丰富文本格式（.rtf 文件）

说明

这些常数列于对象浏览器中的 VisualBasic(VB)对象库。

data 参数是可选的。这允许设置几个不同的、源部件支持的格式，而不必对每种格式都单独加载数据。通过多次调用 SetData 可设置多种格式，每次用不同的格式。若想启动刷新，用 Clear 方法清除 DataObject 的所有数据及格式信息。

Format 参数也是可选的，但是 data 或 format 参数二者之一必须确定。如果确定了 *data* 而不是 *format*，那么 VisualBasic 将试图确定数据的格式。若失败，就会产生一个错误。当目标需要数据且指定了格式，但未提供数据，则源 OLESetData 事件发生，进而源可以提供所需的数据类型。

对 GetData 和 SetData 方法而言，应用列于设置值表以外的数据格

式是可行的。包括通过 RegisterClipboardFormat()API 函数注册的自定义格式。然而，有几点需要注意：

当 SetData 方法不能识别指定的数据格式时，要求数据的格式为字节数组。

当 GetData 方法不能识别数据格式时，尽管 VisualBasic 能够透明地将返回的字节数组转化成其他数据类型，如字符串，但是它总是返回字节数组格式的数据。

在某些操作系统运行时，GetData 返回的字节数组将大于实际数据，并在数组末尾带有随机的字节。原因是 VisualBasic 不知道数据的格式，仅知道操作系统分配给该数据的内存数量。所分配的内存经常要大于数据实际所需要的，所以在分配的内存段的末尾附近有附加的字节。因此要以有意义的方式，采用合适的函数来解释返回的数据。（例如，数据是文本格式时，可用 Left 函数以特定的长度来截取字符串）。

SetFocus 方法

将焦点移至指定的控件或窗体。

应用于

ADOData 控件，TreeView 控件，ImageCombo 控件，ListView 控件，Slider
控件，TabStrip 控件，DateTimePicker 控件，Animation 控件，FlatScrollBar

控件,MonthView 控件,UpDown 控件,DataRepeater 控件,Datalist 控件, DBCombo 控件, Dblist 控件, TextBox 控件(Lightweight), MaskedEdit 控件, MSHFlexGrid 控件, MSFlexGrid 控件, RichTextBox 控件, ProtertyPage 对象, UserContrl 对象, UserDocument 对象, CheckBox 控件, ComboBox 控件, CommandButton 控件, DirListBox 控件, DriveListBox 控件, FileListBox 控件, Form 对象, Forms 集合, HscrollBar, VscrollBar 控件, ListBox 控件, MDIForm 对象, OptionButton 控件, PictureBox 控件, TextBox 控件, OLEContainer 控件

语法

object.**SetFocus**

object 所在处代表对象表达式，其值是“应用于”列表中的一个对象。

说明

对象必须是 Form 对象、MDIForm 对象或者能够接收焦点的控件。调用 SetFocus 方法以后，任何的用户输入将指向指定的窗体或控件。

焦点只能移到可视的窗体或控件。因为在窗体的 Load 事件完成前窗体或窗体上的控件是不可视的，所以如果不是在 Form_Load 事件过程完成之前首先使用 Show 方法显示窗体的话，是不能使用 SetFocus 方法将焦点移到正在自己的 Load 事件中加载的窗体的。

也不能把焦点移到 Enabled 属性被设置为 False 的窗体或控件。如

果已在设计时将 Enabled 属性设置为 False，必须在使用 SetFocus 方法使其接收焦点前将 Enabled 属性设置为 True。

请参阅

Load 语句，Show 方法，Enabled 属性，Enabled 属性(ActiveX 控件)

SetFocus 方法(VBAAdd-In 对象模型)

将焦点移动到指定窗口。

应用于

Window 对象

语法

object.**SetFocus**

object 为一个对象表达式，其值是“应用于”列表中的一个对象。

说明

在可见的窗口上使用 SetFocus 方法。

请参阅

SetSelection 方法，Show 方法(VBAAdd-In 对象模型)，CodePane 对象，Visible 属性

示例

下列示例使用 SetFocus 方法将焦点移到 Windows 集合的特定成员上，就象用鼠标单击了它的标题栏一样。

```
Application.VBE.Windows(9).SetFocus
```

SetSelection 方法

设置代码窗格中的选择。

应用于

CodePane 对象

语法

object.**SetSelection**(*startline*,*startcol*,*endline*,*endcol*)

SetSelection 语法有以下几个部分：

部分	描述
object	必需的。一个对象表达式，其值是“应用于”列表中的一个对象
startline	必需的。一个 Long 型数，用来指定在代码窗格中所选定的起始行
startcol	必需的。一个 Long 型数，用来指定在代码窗格中所选定的起始列
endline	必需的。一个 Long 型数，用来指定在代码窗格中所选定的最后一行
endcol	必需的。一个 Long 型数，用来指定在代码窗格中所选定的最后一列

请参阅

GetSelection 方法，SetFocus 方法，Window 对象

示例

下列示例使用 SetSelection 方法将 CodePanes(1)第二行第四个字符之后（即第五个字符）到第三行第十五个字符之间的文本选择起来。

```
Application.VBE.CodePanes(1).SetSelection2,4,3,15
```

SetText 方法

用以使用指定的 Clipboard 图像格式将文本字符串放到 Clipboard 对象中。不支持命名参数。

应用于

Clipboard 对象

语法

object.**SetText***data*,*format*

SetText 方法的语法包含下列部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
data	必需的。是被放置到剪贴板中的字符串数据
format	可选的。一个常数或数值，按照下列“设置值”中的描述，指定 VisualBasic 识别的剪贴板格式

设置值

用于 format 的设置值有：

常数	值	描述
vbCFLink	&HBF00	DDE 对话信息
vbCFRTF	&HBF01	RTF 格式
vbCFText	1	（缺省值）文本

说明

上述常数在对象浏览器中的 VisualBasic(VB)对象库里列出。

请参阅

GetData 方法，GetFormat 方法，GetText 方法，SetData 方法

示例

本示例使用 SetText 方法从一个文本框中复制文本到剪贴板。要检验此示例，可将本例代码粘贴到一个带有名为 Text1 的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
    ConstCF_TEXT=1                          '定义位图各种格式。
    DimI,Msg,Temp                           '声明变量。
    OnErrorResumeNext                       '设置错误处理。
    Msg="Typeanythingyoulikeintothetextboxbelow."
    Text1.Text=InputBox(Msg)                '取得用户正文。
    Msg="ChooseOKtocopythecontentsofthetextbox"
    Msg=Msg&"totheClipboard."
    MsgBoxMsg                               '显示信息。
```

```
    ClipBoard.Clear                         '清除剪贴板。
    Clipboard.SetTextText1.Text             '将正文放置在剪贴板
上。
    IfClipboard.GetFormat(CF_TEXT)Then
        Text1.Text=""                       '清除该正文框。
        Msg="ThetextisnowontheClipboard.ChooseOK"
        Msg=Msg&"tocopythetextfromtheClipboardback"
        Msg=Msg&"tothetextbox."
        MsgBoxMsg                           '显示信息。
        Temp=Clipboard.GetText(CF_TEXT)     '取得剪贴板正文。
        ForI=Len(Temp)To1Step-1             '使该正文反向。
            Text1.Text=Text1.Text&Mid(Temp,I,1)
        NextI
    Else
        Msg="ThereisnotextontheClipboard."
        MsgBoxMsg                           '显示错误信息。
    EndIf
EndSub
```

SetViewport 方法

对于在窗口中可见的 UserDocument，设置其左边和顶端坐标。

应用于

UserDocument 对象

语法

object.**SetViewPort***left*,*top*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
left	必需的。类型为 Single 的值，它指定了 UserDocument 的左边坐标
top	必需的。类型为 Single 的值，它指定了 UserDocument 的顶端坐标

请参阅

MinHeight，MinWidth 属性，ViewportHeight，ViewportLeft，ViewportTop，ViewportWidth 属性

示例

该示例使用 SetViewport 方法自动将带有焦点的 TextBox 控件放到容器的 Viewport 的左上角。要试用此例，可将包含有 3 个以上 TextBox 控件组成的数组放到 UserDocument 对象上。将下列代码粘贴到通用部分。按 F5 键来运行该工程，然后运行 InternetExplorer（3.0 或者更新版本）。在 InternetExplorer 中，将 ActiveX 文档(UserDocument1.vbd)的路径和文件名键入到“地址”框中（该文件与 VisualBasic 的可执行文件在同一目录下）。当显示 ActiveX 文档时，将任何可辨认的文本键入到第一个 TextBox 控件中。按 TAB 键，移动到下一个控件，看看 SetViewPort 方法的影

响。

```
PrivateSubText1_GotFocus(IndexAsInteger)
    UserDocument.SetViewportText1(Index).Left,_
        Text1(Index).Top
EndSub

PrivateSubUserDocument_Initialize()
    '容器对于所出现的滚动条来说必须足够小。
    '为了保证这一点，可将 MinHeight 属性
    '和 MinWidth 属性设置得比容器大一些。
    UserDocument.MinHeight=10000
    UserDocument.MinWidth=10000
EndSub
```

Sgn 函数

返回一个 Variant(Integer)，指出参数的正负号。

语法

Sgn(*number*)

必要的 number 参数是任何有效的数值表达式。

返回值

如果 number 为	Sgn 返回
大于 0	1
等于 0	0
小于 0	-1

说明

number 参数的符号决定了 Sgn 函数的返回值。

请参阅

Math 函数，Abs 函数

示例

本示例使用 Sgn 函数来判断某数的正负号。

```
DimMyVar1,MyVar2,MyVar3,MySign
MyVar1=12:MyVar2=-2.4:MyVar3=0
MySign=Sgn(MyVar1) '返回 1。
MySign=Sgn(MyVar2) '返回-1。
MySign=Sgn(MyVar3) '返回 0。
```

Shape 控件

Shape 控件是图形控件，显示矩形、正方形、椭圆、圆形、圆角矩形或者圆角正方形。

语法

Shape

说明

或者不调用运行时的 Circle 和 Line 方法而使用在设计时的 Shape 控件，或者二者同时使用。可以在容器中绘制 Shape 控件，但是不能把该控件当作容器。设置 BorderStyle 属性产生的效果取决于 BorderWidth 属性的设置。如果 BorderWidth 不是 1，并且 BorderStyle 不是 0 或者 6，则将 BorderStyle 设置成 1。

属性

BackColor，ForeColor 属性，BackStyle 属性，BorderColor 属性，BorderStyle 属性，BorderWidth 属性，Height，Width 属性，Left，Top 属性，Tag 属性，Visible 属性，DrawMode 属性，FillColor 属性，FillStyle 属性，Shape 属性，Index 属性(控件阵列)，Name 属性，Parent 属性，Container 属性

方法

Refresh 方法，Move 方法，Zorder 方法

请参阅

Frame 控件，PictureBox 控件

Shape 控件（数据报表设计器）

Shape 控件是一个图形控件，显示为一个矩形、正方形、椭圆形、圆形、圆角矩形或圆角正方形。

语法

Shape

说明

Shape 控件的数据报表设计器版可以在任何应用程序中显示确定的形状，在概念上类似标准的 VisualBasic Shape 控件。然而，数据报表设计器版不支持 BorderWidth、FillColor 或 FillStyle 属性。

属性

BackColor，ForeColor 属性，BackStyle 属性，BorderColor 属性，BorderStyle 属性，Height，Width 属性，Left，Top 属性，Visible 属性，Shape 属性，Name 属性

Shape 属性

返回或设置一个值，该值指示一个 Shape 控件的外观。

应用于

Shape 控件(数据报表设计器)，Shape 控件

语法

object.**Shape**[=*value*]

Shape 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	用来指定控件外观的整数，“设置值”中有详细描述

设置值

value 的设置值为：

常数	设置值	描述
VbShapeRectangle	0	（缺省值）矩形
VbShapeSquare	1	正方形
vbShapeOval	2	椭圆形
vbShapeOval	3	圆形
vbShapeRoundedRectangle	4	圆角矩形
vbShapeRoundedSquare	5	圆角正方形

请参阅

BorderStyle 属性，BorderStyle 属性(ActiveX 控件)

示例

本例说明 Shape 控件的六种可能的形状。要试用此例，将下面的代码粘贴到包含一个 OptionButton 控件和一个 Shape 控件的窗体的声明部分。对于 OptionButton，将 Index 属性设置为 0 来创建一个只有一个元素的控件数组，然后按 F5 键。每次单击 OptionButton 能看到一个不同的形状。

```
PrivateSubForm_Load()
   DimI                          '声明变量。
   Option1(0).Caption="Shape#0"
   ForI=1To5                     '创建“Option1”的五个实例。
       LoadOption1(I)
       '为新的选项按钮设置位置。
```

```
        Option1(I).Top=Option1(I-1).Top+Option1(0).Height+40
        '设置选项按钮的标题。
        Option1(I).Caption="Shape#"&I
        '显示新的选项按钮。
        Option1(I).Visible=True
    NextI
EndSub

PrivateSubOption1_Click(IndexAsInteger)
    Shape1.Shape=Index
EndSub
```

ShareName 属性

返回指定驱动器的网络共享名。

应用于

Drive 对象

语法

object.**ShareName**

object 总是一个 Drive 对象。

说明

如果 object 不是一个网络驱动器，则 ShareName 属性返回一个 0 字节长度的字符串("")。

下面的代码举例说明了 ShareName 属性的用法：

```
SubShowDriveInfo(drvpath)
Dimfs,d,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(fs.GetDriveName(fs.GetAbsolutePathName(drvpath)
))
s="Drive"&d.DriveLetter&":-"&d.ShareName
MsgBoxs
EndSub
```

请参阅

AvailableSpace 属性，DriverLetter 属性，DriverType 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，path 属性(文件系统对象对象)，RootFolder 属性，SeralNumber 属性，TotalSize 属性，VolumeName 属性

Shell 函数

执行一个可执行文件，返回一个 Variant(Double)，如果成功的话，代表这个程序的任务 ID，若不成功，则会返回 0。

语法

Shell(*pathname*[,*windowstyle*])

Shell 函数的语法含有下面这些命名参数：

部分	描述
pathname	必需的。Variant(String)，要执行的程序名，以及任何必需的参数或命令行变量，可能还包括目录或文件夹，以及驱动器。在 Macintosh 中，可以使用 MacID 函数来指定一个应用程序的署名而不是名称。下面的例子使用了 MicrosoftWord 的署名: ShellMacID("MSWD")
windowstyle	可需的。Variant(Integer)，表示在程序运行时窗口的样式。如果 windowstyle 省略，则程序是以具有焦点的最小化窗口来执行的。在 Macintosh（系统 7.0 或更高）中，windowstyle 仅决定当应用程序运行时是否获得焦点

windowstyle 命名参数有以下这些值：

常量	值	描述
vbHide	0	窗口被隐藏，且焦点会移到隐式窗口。常数 vbHide 在 Macintosh 平台不可用
vbNormalFocus	1	窗口具有焦点，且会还原到它原来的大小和位置
vbMinimizedFocus	2	窗口会以一个具有焦点的图标来显示
vbMaximizedFocus	3	窗口是一个具有焦点的最大化窗口
vbNormalNoFocus	4	窗口会被还原到最近使用的大小和位置，而当前活动的窗口仍然保持活动
vbMinimizedNoFocus	6	窗口会以一个图标来显示。而当前活动的的窗口仍然保持活动

说明

如果 Shell 函数成功地执行了所要执行的文件，则它会返回程序的任务 ID。任务 ID 是一个唯一的数值，用来指明正在运行的程序。如果 Shell 函数不能打开命名的程序，则会产生错误。

注意缺省情况下，Shell 函数是以异步方式来执行其它程序的。也就是说，用 Shell 启动的程序可能还没有完成执行过程，就已经执行到 Shell 函数之后的语句。

请参阅

AppActivate 语句

示例

本示例使用 Shell 函数来完成一个用户指定的应用程序。

```
'将第二个参数值设成 1，可让该程序以正常大小的窗口完成，并且拥有焦点。
DimRetVal
RetVal=Shell("C:\WINDOWS\CALC.EXE",1)  '运行 Calculator。
```

Shortcut 属性

设置一个值，该值为 Menu 对象指定一个快捷键。在运行时是不可用。

应用于

Menu 控件

说明

使用这个属性来为菜单命令提供键盘快捷方式。可使用“菜单编辑器”来设置该属性。关于可使用的快捷键的列表，请参阅“菜单编辑器”中的“快捷键”列表。

注意除了快捷键之外，还可以通过在 Caption 属性设置中使用&字符来为命令、菜单、和控件指定访问键。

请参阅

Caption 属性，Caption 属性(ActiveX 控件)

ShortName 属性

返回需要较早的 8.3 命名约定的程序所使用的短名字。

应用于

File 对象，Folder 对象

语法

object.**ShortName**

object 总是一个 File 或 Folder 对象。

说明

下面的代码用一个 File 对象举例说明了 ShortName 属性的用法：

```
SubShowShortName(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFile(filespec)
s="Theshortnamefor"&""&UCase(f.Name)
s=s&""&vbCrLf
s=s&"is:"&""&f.ShortName&""
MsgBoxs,0,"ShortNameInfo"
EndSub
```

请参阅

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性(文件系统对象对象)，ParentFolder 属性，Path

属性(文件系统对象)，ShortPath 属性，Size 属性(文件系统对象对象)，SubFolders 属性，Type 属性

ShortPath 属性

返回需要较早的 8.3 文件命名约定的程序所使用的短路径。

应用于

File 对象，Foler 对象

语法

object.**ShortPath**

object 总是一个 File 或 Folder 对象。

说明

下面的代码用一个 File 对象举例说明了 ShortName 属性的用法：

```
SubShowShortPath(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFile(filespec)
s="Theshortpathfor"&""&UCase(f.Name)
s=s&""&vbCrLf
s=s&"is:"&""&f.ShortPath&""
MsgBoxs,0,"ShortPathInfo"
EndSub
```

请参阅

Attributes 属性，DataCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性(文件系统对象对象)，ParentFolder 属性，Path 属性(文件系统对象对象)，ShortName 属性，Size 属性(文件系统对象对象)，SubFolders 属性，Type 属性

Show 事件（UserControl 对象）

当对象的 Visible 属性变为 True 时，发生该事件。

应用于

UserControl 对象

语法

Sub*object*_**Show**()

Show 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

要在 Windows 中绘制屏幕，每个对象都必须有一个窗口，无论是临时窗口还是永久窗口；VisualBasic 的 ActiveX 控件具有永久的窗口。将控件定位在窗体上之前，其窗口不在容器上。当添加窗口时，控件接收 Show 事件。

当控件的窗口位于窗体上时，控件的 Visible 属性变为 True 时控件接收 Show 事件。

如果窗体隐藏后又显示出来，或者窗体最小化后又还原，则控件不接收 Show 事件。在执行这些操作期间，控件的窗口仍保留在窗体上，而且其 Visible 属性没有改变。

如果在 internet 浏览器中显示控件时，用户返回包含该控件的页，会产生 Show 事件。

注意在版本早于 5.0 版的 VisualBasic 中使用控件时，控件在设计时不接收 Show 事件。这是因为早期版本的 VisualBasic 在设计时不将任何可见的窗口放到窗体上。

请参阅

Visible 属性

Show 事件（UserDocument 对象）

当对象的 Visible 属性变为 True 时，发生该事件。

应用于

UserDocument 对象

语法

Sub*object*_**Show**()

Show 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

说明

要在 Windows 中绘制屏幕，每个对象都必须有一个窗口，无论是临时窗口还是永久窗口；VisualBasic 的 ActiveX 文档具有永久的窗口。在将 object 定位在窗体上之前，其窗口不在容器上。添加窗口时，UserDocument 对象接收 Show 事件。

object 的窗口位于容器上时，object 的 Visible 属性变更为 True 时 object 接收 Show 事件.

如果容器隐藏后又显示出来，或者容器最小化后又还原，则 object 不接收 Show 事件。在进行这些操作期间，object 的窗口保留在容器上，而且其 Visible 属性没有改变。

如果在 internet 浏览器中显示 object 时，用户定位到该页，会产生 Show 事件。

注意在早于 5.0 版的 VisualBasic 中使用 object 时，在设计时 object 不接收 Show 事件。这是因为早期版本的 VisualBasic 在设计时不将任何可见的窗口放到窗体上。

请参阅

Visible 属性

Show 方法

用以显示 MDIForm 或 Form 对象。不支持命名参数。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

object.**Show***style*,*ownerform*

Show 方法的语法包含下列部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 object，则与活动窗体模块关联的窗体缺省为 object
style	可选的。一个整数，它用以决定窗体是模式还是无模式。如果 style 为 0，则窗体是无模式的；如果 style 为 1，则窗体是模式的
ownerform	可选的。字符串表达式，指出部件所属的窗体被显示。对于标准的 VisualBasic 窗体，使用关键字 Me

说明

如果调用 Show 方法时指定的窗体没有装载，VisualBasic 将自动装载该窗体。

当 Show 在显示无模式窗体时，随后遇到的代码则要执行。当 Show 在显示模式窗体(modalform)时，则随后的代码直到该窗体被隐藏或卸载时才能执行。

当 Show 在显示模式窗体时，除了模式窗体中的对象之外不能进行输入（键盘或鼠标单击）。对其它窗体进行输入前程序必须隐藏或卸载模式窗体（通常是处于响应用户某些操作状态）。MDIForm 不能是形式的。

在模式窗体显示时，虽然应用程序中的其它窗体失效，但其它应

用程序不会失效。

应用程序的启动窗体在其 Load 事件调用后会自动出现。

下面的例子说明如何使用 ownerform 参数：

```
PrivateSubcmdShowResults_Click()
   '显示模式窗体 frmResults.
   frmResults.ShowvbModal,Me
EndSub
```

请参阅

Activate，Deactivate 事件，Load 语句，Unload 语句，Hide 方法，Visible 属性，MAPISession 控件，MAPIMessages 控件

示例

本示例使用 Show 方法显示一个隐藏的窗体。要检验此示例，可将本例代码粘贴到一个非 MDI 的窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimMsg                                    '声明变量。
   Hide                                      '隐藏窗体。
   Msg="ChooseOKtomaketheformreappear."
   MsgBoxMsg                                 '显示信息。
   Show                                      '使窗体重显。
EndSub
```

Show 方法（VBA 外接程序对象模型）

使指定的代码窗格成为其窗口中可见的代码窗格。

应用于

CodePane 对象

语法

object.**Show**

object 为一个对象表达式，其值是“应用于”列表中的一个对象。

说明

Show 方法可使此指定的代码窗格成为该窗口中具有焦点的窗格。

请参阅

SetFocus 方法，CodePaneView 属性

示例

下列示例使用 Show 方法来将指定的代码窗格移至前景。

```
Application.VBE.CodePanes(2).Show
```

ShowColor 方法

显示 CommonDialog 控件的“颜色”对话框。

应用于

CommonDialog 控件

语法

object.**ShowColor**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

请参阅

ShowFont 方法，ShowHelp 方法，ShowOpen 方法，ShowPrinter 方法，ShowSave 方法

示例

本例使用 CommonDialog 控件和 ShowColor，ShowFont，ShowHelp，ShowOpen，ShowPrinter，和 ShowSav 等方法，显示公用对话框。要试用此例子，将代码粘贴到窗体的声明部分，该窗体带有 CommandButton，OptionButton（设置该选项按钮的 Index 属性为 0）、以及 CommonDialog 控件。按 F5 键，为所要的公用对话框选择选项按钮，再选取命令按钮。

```
PrivateSubForm_Paint()
   StaticFlagFormPaintedAsInteger
   '当第一次画窗体时，
   IfFlagFormPainted<>TrueThen
       Fori=1To5
           LoadOption1(i)              '给数组添加 5 个选项按钮。
           Option1(i).Top=Option1(i-1).Top+350
           Option1(i).Visible=True
       Nexti
       Option1(0).Caption="Open"         '在每个选项按钮上放置标题。
       Option1(1).Caption="Save"
```

```
        Option1(2).Caption="Color"
        Option1(3).Caption="Font"
        Option1(4).Caption="Printer"
        Option1(5).Caption="Help"
        Command1.Caption="ShowDlg"          '标签命令按钮。
        FlagFormPainted=True                '窗体已画完。
    EndIf
EndSub

PrivateSubCommand1_Click()
    IfOption1(0).ValueThen                  '如果选择打开选箱，
        CommonDialog1.ShowOpen              '显示打开公共对话框。
    ElseIfOption1(1).ValueThen              '否则，
        CommonDialog1.ShowSave              '显示 Save 公共对话框。
    ElseIfOption1(2).ValueThen              '否则，
        CommonDialog1.ShowColor             '显示 Color 公共对话框。
    ElseIfOption1(3).ValueThen              '否则，
        CommonDialog1.Flags=cdlCFBoth       '在使用 ShowFont 方法之
前，
                '必须给 cdlCFBoth，
    'cdlCFPrinterFonts，
                '或 cdlCFScreenFonts
```

```
    '置标识属性。
        CommonDialog1.ShowFont              '显示字体公共对话框。
    ElseIfOption1(4).ValueThen              '或，
        CommonDialog1.ShowPrinter       '显示打印机公共对话框。
    ElseIfOption1(5).ValueThen              '或
        CommonDialog1.HelpFile="VB5.hlp"
        CommonDialog1.HelpCommand=cdlHelpContents
        CommonDialog1.ShowHelp'显示 VisualBasic 帮助目录主题。
        EndIf
EndSub
```

ShowFont 方法

显示 CommonDialog 控件的“字体”对话框。

应用于

CommonDialog 控件

语法

object.**ShowFont**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

在使用 ShowFont 方法前，必须先设置 CommonDialog 控件的 Flags 属性为下列三个常数或值中的一个：cdlCFBoth 或&H3,cdlCFPrinterFonts 或&H2,以及 cdlCFScreenFonts 或&H1。如

果不置 Flags，将会显示一个信息框，提示“没有安装的字体。”并产生一个运行时错误。

请参阅

Flags 属性(字体对话框)，ShowColor 方法，ShowHelp 方法，ShowOpen 方法，ShowPrinter 方法，ShowSave 方法，Font 对象

示例

本例使用 CommonDialog 控件和 ShowColor，ShowFont，ShowHelp，ShowOpen，ShowPrinter，和 ShowSav 等方法，显示用共对话框。要试用此例子，将代码粘贴到窗体的声明部分，该窗体带有 CommandButton，OptionButton (设置该选项按钮的 Index 属性为 0)、以及 CommonDialog 控件。按 F5 键，为所要的公用对话框选择选项按钮，再选取命令按钮。

```
PrivateSubForm_Paint()
   StaticFlagFormPaintedAsInteger
   '当第一次画窗体时，
   IfFlagFormPainted<>TrueThen
      Fori=1To5
         LoadOption1(i)                    '给数组添加 5 个选项按钮。
         Option1(i).Top=Option1(i-1).Top+350
         Option1(i).Visible=True
      Nexti
      Option1(0).Caption="Open"          '在每个选项按钮上放置标
```

```
题。
        Option1(1).Caption="Save"
        Option1(2).Caption="Color"
        Option1(3).Caption="Font"
        Option1(4).Caption="Printer"
        Option1(5).Caption="Help"
        Command1.Caption="ShowDlg"          '标签命令按钮。
        FlagFormPainted=True                '窗体已画完。
    EndIf
EndSub

PrivateSubCommand1_Click()
    IfOption1(0).ValueThen                  '如果选择打开选箱，
        CommonDialog1.ShowOpen              '显示打开公共对话框。
    ElseIfOption1(1).ValueThen              '否则，
        CommonDialog1.ShowSave              '显示 Save 公共对话框。
    ElseIfOption1(2).ValueThen              '否则，
        CommonDialog1.ShowColor             '显示 Color 公共对话框。
    ElseIfOption1(3).ValueThen              '否则，
        CommonDialog1.Flags=cdlCFBoth       '在使用 ShowFont 方法之
前，                    '必须给 cdlCFBoth，
    'cdlCFPrinterFonts，
                    '或 cdlCFScreenFonts
```

```
    '置标识属性。
        CommonDialog1.ShowFont              '显示字体公共对话框。
    ElseIfOption1(4).ValueThen              '或，
        CommonDialog1.ShowPrinter       '显示打印机公共对话框。
    ElseIfOption1(5).ValueThen              '或
        CommonDialog1.HelpFile="VB5.hlp"
        CommonDialog1.HelpCommand=cdlHelpContents
        CommonDialog1.ShowHelp              '显示 VisualBasic 帮助目录
主题。
        EndIf
EndSub
```

ShowGrabHandles 属性

返回一个布尔值，它规定控件是否应当显示抓取处理。

应用于

AmbientProperties 对象

语法

object.**ShowGrabHandles**

ShowGrabHandles 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

设置值

ShowGrabHandles 属性可能的布尔返回值为：

设置值	描述
True	控件在需要时显示抓取处理。如果容器没有实现这种环境属性，这将是缺省的设置值
False	控件不显示抓取处理

说明

当控件处于设计模式（控件的运行模式）的容器中时，控件的缺省行为是自动显示抓取处理。然而，许多容器都不希望控件显示抓取处理，它们倾向于用其它方法处理控件大小的指示。容器使用 ShowGrabHandles 属性通知控件谁来显示大小指示。

注意所有已知的容器都倾向于自己处理控件大小指示，因此它们将 ShowGrabHandles 属性设置为 False。当 ShowGrabHandles 设置为 True 时，可能不需要进行实际处理。

ShowHatching 属性

返回一个布尔值，它规定控件是否在它周围显示阴影。

应用于

AmibentProperties 对象

语法

object.**ShowHatching**

ShowHatching 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

设置值

ShowHatching 属性可能的布尔返回值为：

设置值	描述
True	控件在需要时显示阴影标记。如果容器没有实现这种环境属性，这将是缺省的设置值
False	控件显示阴影标记

说明

当控件处于设计模式（控件的运行模式）的容器中并且获得焦点时，该控件缺省的行为是自动显示阴影。然而，许多容器都不希望控件显示阴影，它们倾向于用其它方法处理控件焦点的指示。容器使用 ShowHatching 属性通知控件谁来显示控件焦点的指示。注意 VisualBasic 窗体不实现这种环境属性，因此当控件放置在 VisualBasic 的窗体上时，ShowHatching 属性将设置为缺省值：True。然而，当 ShowHatching 属性值变为 True 时，VisualBasic 不希望控件在响应时执行任何操作，因此 ShowHatching 为 True 时，实际上可能不需要处理。

ShowHelp 方法

运行 WINHLP32.EXE 并显示指定的帮助文件。

应用于

CommonDialog 控件

语法

object.**ShowHelp**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

使用 ShowHelp 方法前，必须 CommonDialog 控件的 HelpFile 和 HelpCommand 属性设置为其相应的一个常数或值。否则，Winhlp32.exe 就不能显示帮助文件。

请参阅

HelpCommand 属性，ShowColor 方法，ShowFont 方法，ShowOpen 方法，ShowPrinter 方法，ShowSave 方法，HelpFile 属性(App，CommonDialog，MenuLine)

示例

本例使用 CommonDialog 控件和 ShowColor，ShowFont，ShowHelp，ShowOpen，ShowPrinter，和 ShowSav 等方法，显示公共对话框。要试用此例子，将代码粘贴到窗体的声明部分，该窗体带有 CommandButton，OptionButton（设置该选项按钮的 Index 属性为 0)、以及 CommonDialog 控件。按 F5 键，为所要的公共对话框选择选项按钮，再选取命令按钮。

```
PrivateSubForm_Paint()
   StaticFlagFormPaintedAsInteger
   '当第一次画窗体时，
```

```
    IfFlagFormPainted<>TrueThen
        Fori=1To5
            LoadOption1(i)                    '给数组添加 5 个选项按钮。
            Option1(i).Top=Option1(i-1).Top+350
            Option1(i).Visible=True
        Nexti
        Option1(0).Caption="Open"    '在每个选项按钮上放置标题。
        Option1(1).Caption="Save"
        Option1(2).Caption="Color"
        Option1(3).Caption="Font"
        Option1(4).Caption="Printer"
        Option1(5).Caption="Help"
        Command1.Caption="ShowDlg"            '标签命令按钮。
        FlagFormPainted=True                  '窗体已画完。
    EndIf
EndSub

PrivateSubCommand1_Click()
    IfOption1(0).ValueThen                    '如果选择打开选箱，
        CommonDialog1.ShowOpen                '显示打开公共对话框。
    ElseIfOption1(1).ValueThen                '否则，
        CommonDialog1.ShowSave                '显示 Save 公共对话框。
```

```
    ElseIfOption1(2).ValueThen                    '否则，
        CommonDialog1.ShowColor                   '显示 Color 公共对话框。
    ElseIfOption1(3).ValueThen                    '否则，
            CommonDialog1.Flags=cdlCFBoth         '在使用
ShowFont 方法之前，
      '必须给 cdlCFBoth，                         'cdlCFPrinterFonts,
      '或 cdlCFScreenFonts
    '置标识属性。
        CommonDialog1.ShowFont                    '显示字体公共对话框。
    ElseIfOption1(4).ValueThen                    '或，
        CommonDialog1.ShowPrinter                 '显示打印机公共对话框。
    ElseIfOption1(5).ValueThen                    '或
        CommonDialog1.HelpFile="VB5.hlp"
        CommonDialog1.HelpCommand=cdlHelpContents
        CommonDialog1.ShowHelp      '显示 VisualBasic 帮助目录主题。
        EndIf
EndSub
```

ShowInTaskbar 属性

返回或设置一个值，该值决定一个 Form 对象是否出现在 Windows95 任务栏中。该值在运行时为只读状态。

应用于

Form 对象，Forms 集合

语法

object.**ShowInTaskbar**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

设置值

ShowInTaskbar 属性的设置值为：

设置值	描述
True	（缺省值）该 Form 对象出现在任务栏中
False	该 Form 对象不出现在任务栏中

说明

应用程序中使用 ShowInTaskbar 属性可使对话框不出现在任务栏中。

ShowInTaskbar 属性的缺省值假定为该 Form 对象的 BorderStyle 属性的缺省设置(Sizable)。改变BorderStyle 属性将会改变ShowInTaskbar 属性的设置。

请参阅

BorderStyle 属性

ShowOpen 方法

显示 CommonDialog 控件的“打开”对话框。

应用于

CommonDialog 控件

语法

object.**ShowOpen**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

请参阅

ShowColor 方法，ShowFont 方法，ShowHelp 方法，ShowPrinter 方法，ShowSave 方法

示例

本例使用 CommonDialog 控件和 ShowColor，ShowFont，ShowHelp，ShowOpen，ShowPrinter，和 ShowSav 等方法，显示公共对话框。要试用此例子，将代码粘贴到窗体的声明部分，该窗体带有 CommandButton，OptionButton（设置该选项按钮的 Index 属性为 0）、以及 CommonDialog 控件。按 F5 键，为所要的公共对话框选择选项按钮，再选取命令按钮。

```
PrivateSubForm_Paint()
   StaticFlagFormPaintedAsInteger
   '当第一次画窗体时，
   IfFlagFormPainted<>TrueThen
       Fori=1To5
           LoadOption1(i)                    '给数组添加 5 个选项按钮。
           Option1(i).Top=Option1(i-1).Top+350
           Option1(i).Visible=True
       Nexti
```

```
        Option1(0).Caption="Open"    '在每个选项按钮上放置标题。
        Option1(1).Caption="Save"
        Option1(2).Caption="Color"
        Option1(3).Caption="Font"
        Option1(4).Caption="Printer"
        Option1(5).Caption="Help"
        Command1.Caption="ShowDlg"        '标签命令按钮。
        FlagFormPainted=True              '窗体已画完。
    EndIf
EndSub

PrivateSubCommand1_Click()
    IfOption1(0).ValueThen                '如果选择打开选箱，
        CommonDialog1.ShowOpen            '显示打开公共对话框。
    ElseIfOption1(1).ValueThen            '否则，
        CommonDialog1.ShowSave            '显示 Save 公共对话框。
    ElseIfOption1(2).ValueThen            '否则，
        CommonDialog1.ShowColor           '显示 Color 公共对话框。
    ElseIfOption1(3).ValueThen            '否则，
        CommonDialog1.Flags=cdlCFBoth'在使用 ShowFont 方法之前，
                '必须给 cdlCFBoth，
    'cdlCFPrinterFonts,
```

```
                    '或 cdlCFScreenFonts
    '置标识属性。
        CommonDialog1.ShowFont              '显示字体公共对话框。
    ElseIfOption1(4).ValueThen              '或，
        CommonDialog1.ShowPrinter       '显示打印机公共对话框。
        ElseIfOption1(5).ValueThen          '或
        CommonDialog1.HelpFile="VB5.hlp"
        CommonDialog1.HelpCommand=cdlHelpContents
        CommonDialog1.ShowHelp              '显示 VisualBasic 帮助目录
主题。
        EndIf
EndSub
```

ShowPrinter 方法

显示 CommonDialog 控件的“打印”对话框。

应用于

CommonDialog 控件

语法

object.**ShowPrinter**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

请参阅

ShowColor 方法，ShowFont 方法，ShowHelp 方法，ShowOpen 方

法，ShowSave 方法，Printer 对象，Printers 集合

示例

本例使用 CommonDialog 控件和 ShowColor，ShowFont，ShowHelp，ShowOpen，ShowPrinter，和 ShowSav 等方法，显示公共对话框。要试用此例子，将代码粘贴到窗体的声明部分，该窗体带有 CommandButton，OptionButton（设置该选项按钮的 Index 属性为 0）、以及 CommonDialog 控件。按 F5 键，为所要的公共对话框选择选项按钮，再选取命令按钮。

```
PrivateSubForm_Paint()
   StaticFlagFormPaintedAsInteger
   '当第一次画窗体时，
   IfFlagFormPainted<>TrueThen
       Fori=1To5
           LoadOption1(i)                    '给数组添加 5 个选项按钮。
           Option1(i).Top=Option1(i-1).Top+350
           Option1(i).Visible=True
       Nexti
       Option1(0).Caption="Open"    '在每个选项按钮上放置标题。
       Option1(1).Caption="Save"
       Option1(2).Caption="Color"
       Option1(3).Caption="Font"
       Option1(4).Caption="Printer"
```

```
        Option1(5).Caption="Help"
        Command1.Caption="ShowDlg"          '标签命令按钮。
        FlagFormPainted=True                '窗体已画完。
    EndIf
EndSub

PrivateSubCommand1_Click()
    IfOption1(0).ValueThen                  '如果选择打开选箱，
        CommonDialog1.ShowOpen              '显示打开公共对话框。
    ElseIfOption1(1).ValueThen              '否则，
        CommonDialog1.ShowSave              '显示 Save 公共对话框。
    ElseIfOption1(2).ValueThen              '否则，
        CommonDialog1.ShowColor             '显示 Color 公共对话框。
    ElseIfOption1(3).ValueThen              '否则，
        CommonDialog1.Flags=cdlCFBoth'在使用 ShowFont 方法之前，
                '必须给 cdlCFBoth，
    'cdlCFPrinterFonts，
                '或 cdlCFScreenFonts
    '置标识属性。
        CommonDialog1.ShowFont              '显示字体公共对话框。
    ElseIfOption1(4).ValueThen              '或，
        CommonDialog1.ShowPrinter           '显示打印机公共对话框。
    ElseIfOption1(5).ValueThen              '或
```

```
        CommonDialog1.HelpFile="VB5.hlp"
        CommonDialog1.HelpCommand=cdlHelpContents
        CommonDialog1.ShowHelp      '显示 VisualBasic 帮助目录主题。
        EndIf
    EndSub
```

ShowSave 方法

显示 CommonDialog 控件的“另存为”对话框。

应用于

CommonDialog 控件

语法

object.**ShowSave**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

请参阅

ShowColor 方法，ShowFont 方法，ShowHelp 方法，ShowOpen 方法，ShowPrinter 方法

示例

本例使用 CommonDialog 控件和 ShowColor，ShowFont，ShowHelp，ShowOpen，ShowPrinter，和 ShowSav 等方法，显示公共对话框。要试用此例子，将代码粘贴到窗体的声明部分，该窗体带有 CommandButton，OptionButton（设置该选项按钮的 Index 属性为 0）、以及 CommonDialog 控件。按 F5 键，为所要的公共对话框选择选

项按钮，再选取命令按钮。

```
PrivateSubForm_Paint()
   StaticFlagFormPaintedAsInteger
   '当第一次画窗体时，
   IfFlagFormPainted<>TrueThen
       Fori=1To5
           LoadOption1(i)   '给数组添加 5 个选项按钮。
           Option1(i).Top=Option1(i-1).Top+350
           Option1(i).Visible=True
       Nexti
       Option1(0).Caption="Open"          '在每个选项按钮上放置标
题。
       Option1(1).Caption="Save"
       Option1(2).Caption="Color"
       Option1(3).Caption="Font"
       Option1(4).Caption="Printer"
       Option1(5).Caption="Help"
       Command1.Caption="ShowDlg"         '标签命令按钮。
       FlagFormPainted=True               '窗体已画完。
   EndIf
EndSub

PrivateSubCommand1_Click()
```

```
    If Option1(0).Value Then                    '如果选择打开选箱，
        CommonDialog1.ShowOpen                  '显示打开公共对话框。
    ElseIf Option1(1).Value Then                '否则，
        CommonDialog1.ShowSave                  '显示 Save 公共对话框。
    ElseIf Option1(2).Value Then                '否则，
        CommonDialog1.ShowColor                 '显示 Color 公共对话框。
    ElseIf Option1(3).Value Then                '否则，
        CommonDialog1.Flags=cdlCFBoth     '在使用 ShowFont 方法之前，
                    '必须给 cdlCFBoth，
    'cdlCFPrinterFonts，
                    '或 cdlCFScreenFonts
    '置标识属性。
        CommonDialog1.ShowFont          '显示字体公共对话框。
    ElseIf Option1(4).Value Then            '或，
        CommonDialog1.ShowPrinter       '显示打印机公共对话框。
    ElseIf Option1(5).Value Then            '或
        CommonDialog1.HelpFile="VB5.hlp"
        CommonDialog1.HelpCommand=cdlHelpContents
        CommonDialog1.ShowHelp                  '显示 VisualBasic 帮助目录主题。
        EndIf
```

EndSub

ShowWhatsThis 方法

显示 Windows95Help 所提供的“这是什么”弹出式窗口使用的 Help 文件中选定的一个主题。

应用于

ADO 数据控件，TreeView 控件，ImageCombo 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，TextBox 控件(Lightweight)，Masked 编辑控件，MSChart 对象，MSHFlexGrid 控件，MSFlexGrid 控件，RemoteData 控件，RichTextBox 控件，Data 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Frame 控件，HscrollBar，VscrollBar 控件，Image 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 包容器控件

语法

object.**ShowWhatsThis**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

ShowWhatsThis 方法对于从应用程序上下文菜单中提供上下文敏感 Help 非常有用。该方法显示语法中指定对象的 WhatsThisHelpID 属性所标识的主题。

请参阅

WhatsThisHelpID 属性，WhatsThisButton 属性，WhatsThisHelp 属性，
WhatsThisMode 方法

示例

该例子显示通过在为该按钮创建的上下文菜单中选取一个菜单命令的某个 CommandButton 控件的“这是什么”的 Help 主题。
设置窗体的 WhatsThisHelp 属性为 True。将 CommandButton 控件放在窗体上，用“菜单编辑器”创建一个菜单，使它具有顶层不可见的项 mnuBtnContextMenu 和子菜单 mnuBtnWhatsThis，子菜单标题为“这是什么？”

```
PrivateThisControlAsControl

PrivateSubCommand1_MouseUp(ButtonAsInteger,ShiftAsInteger,XAsSingle,YAsSingle)
    IfButton=vbRightButtonThen
        SetThisControl=Command1
        PopupMenumnuBtnContextMenu
```

```
        EndIf
    SetThisControl=Nothing
EndSub

PrivateSubmnuBtnWhatsThis_Click()
    ThisControl.ShowWhatsThis
EndSub
```

Sin 函数

返回一 Double，指定参数的 sine（正弦）值。

语法

Sin(*number*)

必要的 number 参数是 Double 或任何有效的数值表达式，表示一个以弧度为单位的角。

说明

Sin 函数取一角度为参数值，并返回角的对边长度除以斜边长度的比值。

结果的取值范围在 -1～1 之间。

为了将角度转换为弧度，请将角度乘以 pi/180。为了将弧度转换为角度，请将弧度乘以 180/pi。

请参阅

Atn 函数，Cos 函数，Tan 函数，DerivedMath 函数

示例

本示例使用 Sin 函数来求出一个角的正弦值（sin()）。

```
DimMyAngle,MyCosecant
MyAngle=1.3                    '定义角度（以“弧度”为单位）。
MyCosecant=1/Sin(MyAngle)      '利用正弦来计算余割（csc()）。
```

Size 方法

改变 UserControl 对象的宽度和高度。

应用于

UserControl 对象

语法

object.**Size***width*,*heigh*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
width	必需的。用缇表示的对象的宽度
height	必需的。用缇表示的对象的高度

说明

UserControl 对象的 Width 和 Height 属性总是用缇来给出的，而不管 ScaleMode 是否如此。

请参阅

Height，Width 属性

Size 属性

对于文件来说，返回以字节为单位的指定文件大小。对于文件夹来说，返回以字节为单位的包含在文件夹中所有文件和子文件夹的大小。

应用于

File 对象，Folder 对象

语法

object.**Size**

object 总是一个 File 或 Folder 对象。

说明

下面的代码用一个 Folder 对象举例说明了 Size 属性的用法：

```
SubShowFolderSize(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFolder(filespec)
s=UCase(f.Name)&"uses"&f.size&"bytes."
MsgBoxs,0,"FolderSizeInfo"
EndSub
```

请参阅

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性(文件系统对象对象)，ParentFolder 属性，Path

属性(文件系统对象对象)，ShortName 属性，ShortPath 属性，SubFolders 属性，Type 属性

Size 属性（字体）

返回或设置 Font 对象中使用字体的大小。

应用于

Font 对象

语法

object.**Size**[=*number*]

Size 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个用来指定用点数表示的字体大小的数值表达式

说明

使用这个属性来将文本格式化为想要的字体大小。缺省的字体大小由操作系统所决定。可以指定用点数表示的字体大小来改变缺省值。Size 属性的最大值是 2048 个点。

Font 对象在设计时不能直接使用。取而代之的是，在“属性”窗口中通过选择控件的 Font 属性并单击“属性”按钮来设置 Size 属性。在“字体”对话框的“字形”框中，选择想要的大小。然而在运行时，通过为 Font 对象指定 Size 属性值可直接设置 Size。

请参阅

Bold 属性，FontTransparent 属性，Italic 属性，StrikeThrough 属性，Underline 属性，Weight 属性，Name 属性

示例

这个例子完成下面的功能：每次单击鼠标时在窗体上打印文本。要试用此例，可以先将下面的代码粘贴到一个窗体的声明部分中，然后按下 F5 键并双击此窗体。

```
PrivateSubForm_Click()
   Font.Bold=NotFont.Bold   '转换 Bold。
   Font.StrikeThrough=NotFont.StrikeThrough'转换 StrikeThrough。
   Font.Italic=NotFont.Italic   '转换 Italic。
   Font.Underline=NotFont.Underline'转换 Underline。
   Font.Size=16'设置 Size 属性。
   IfFont.BoldThen
       Print"Fontweightis"&Font.Weight&"(bold)."
   Else
       Print"Fontweightis"&Font.Weight&"(notbold)."
   EndIf
EndSub
```

SizeMode 属性

返回或设置一个值，它指定当包含对象时，OLE 容器控件如何调

整大小，或者如何显示图像。

应用于

OLE 包容器控件

语法

object.**SizeMode**[=*value*]

SizeMode 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	整数或常数，它指示控件如何调整大小或者它的图象如何显示，在“设置值”中有详细说明

设置值

value 的设置值是：

常数	值	描述
vbOLESizeClip	0	（缺省值）剪裁。对象按实际大小显示。如果对象比 OLE 容器控件大，它的图像被控件的边框剪裁掉
vbOLESizeStretch	1	伸展。调整对象图像的大小使其充满 OLE 容器控件。图象也许不能维持对象原来的比例
cbOLESizeAutoSize	2	自动。OLE 容器控件重新调整大小以显示整个对象
vbOLESizeZoom	3	缩放。重新调整对象的大小使其尽可能充满 OLE 容器控件，并且仍然维持该对象原来的比例

说明

当 SizeMode 置为 2（自动）时，如果显示对象的大小作了改变，OLE 容器控件自动调整大小。此时，在 OLE 容器控件自动调整大小之前，先调用 Resize 事件。在 Resize 事件过程中的 heightnew 和 widthnew 参数，指示显示对象的最佳大小（其大小由创建该对象的应用程序决定）。可以改变 Resize 事件过程中的 heightnew 和 widthnew 参数的值，来调整控件的大小。

请参阅

Resize 事件，Updated 事件

SizeMode 属性（RptImage 控件）

返回或设置一个值，指定如何调整数据报表设计器 Image 控件中一幅图画的尺寸。

应用于

Image 控件(数据报表设计器)

语法

object.**SizeMode**[=*integer*]

SizeMode 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
integer	可选的。一个数值表达式，指定图片的尺寸如何调整，如在设置值中所示

设置值

对 integer 的设置如下：

常数	值	描述
rptSizeClip	0	图画将被剪切
rptSizeStretch	1	图画将被拉伸以添满整个控件
rptSizeZoom	2	类似拉伸，只是图像 X:Y 的比例仍保持为常数

Skip 方法

当读一个 TextStream 文件时跳过指定数量的字符。

应用于

TextStream 对象

语法

object.**Skip**(*characters*)

Skip 方法语法有下面几部分：

部分	描述
object	必需的。始终是一个 TextStream 对象的名字
characters	必需的。当读文件时要跳过的字符的数量

说明

跳过的字符被放弃。

请参阅

Close 方法，Read 方法，ReadAll 方法，ReadLine 方法，SkipLine 方法，Write 方法，WriteBlankLines 方法，WriteLine 方法

SkipLine 方法

当读一个 TextStream 文件时跳过下一行。

应用于

TextStream 对象

语法

object.**SkipLine**

object 始终是一个 TextStream 对象的名字。

说明

跳过一行是指读取并放弃一行中的所有字符，一直到并包括该行的换行符。

如果读的文件没有打开，则产生一个错误。

请参阅

Close 方法，Read 方法，ReadAll 方法，ReadLine 方法，Skip 方法，Write
方法，WriteBlankLines 方法，WriteLine 方法

SLN 函数

返回一个 Double，在一期里指定一项资产的直线折旧。

语法

SLN(*cost*,*salvage*,*life*)

SLN 函数有下列命名参数：

部分	描述
cost	必需的。Double 指定资产的初始成本
salvage	必需的。Double 指定资产在可用年限结束后的价值
life	必需的。Double 指定资产的可用年限

说明

折旧期间必须用与 life 参数相同的单位表示。所有参数都必须是正数。

请参阅

DDB 函数，FV 函数，IPmt 函数，IRR 函数，MIRR 函数，Nper 函数，NPV 函数，Pmt 函数，PV 函数，Rate 函数，SYD 函数

示例

本示例使用 SLN 函数以“直线折旧”(straight-linedepreciation）法计算某项资产在一期之内的折旧，计算时须给定资产的初始成本（InitCost)，资产在可用年限退出后的价值（SalvageVal）及资产的可用年限（LifeTime)。

```
DimFmt,InitCost,SalvageVal,MonthLife,LifeTime,PDepr
ConstYEARMONTHS=12                    '一年之中的月份数。
Fmt="###,##0.00"    '定义金额格式。
InitCost=InputBox("What'stheinitialcostoftheasset?")
SalvageVal=InputBox("What'stheasset'svalueattheendofitsusefullife?")
MonthLife=InputBox("What'stheasset'susefullifeinmonths?")
DoWhileMonthLife<YEARMONTHS                    '确保计算期间大于等于一年。
   MsgBox"Assetlifemustbeayearormore."
   MonthLife=InputBox("What'stheasset'susefullifeinmonths?")
Loop
LifeTime=MonthLife/YEARMONTHS                    '将月份数转成年份数。
```

```
IfLifeTime<>Int(MonthLife/YEARMONTHS)Then
   LifeTime=Int(LifeTime+1)                    '四舍五入到最接近的年份。
EndIf
PDepr=SLN(InitCost,SalvageVal,LifeTime)
MsgBox"Thedepreciationis"&Format(PDepr,Fmt)&"peryear."
```

Sorted 属性

返回一个值，指定控件的元素是否自动按字母表顺序排序。

应用于

ComboBox 控件，ListBox 控件

语法

object.**Sorted**

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

返回值

Sorted 属性的返回值如下：

设置值	描述
True	列表中的项目按字符码顺序排序
False	（缺省值）列表中的项目不按字母表顺序排序

说明

当该属性为 True 时，VisualBasic 进行几乎所有的字符串处理，

以保持按字母表排顺，包括对添加或删除项目所需的索引序号的改变。

注意用 AddItem 方法向列表中的某一确定位置添加元素可能违反排序顺序，后面项目的添加可能不会正确地排序。

请参阅

ListView 控件，SortKey 属性(ListView 控件)，SortOrder 属性(ListView 控件)，AddItem 方法，RemoveItem 方法，List 属性，ListCount 属性，ListIndex 属性

Source 属性

返回一个错误的来源。

应用于

Error 对象(数据报表设计器)

语法

object.**Source**

object 所在处表示一个对象表达式，其值为“应用于”列表中的一个对象。

返回类型

String

Source 属性（用于 VB 的应用程序）

返回或设置一个字符串表达式，指明最初生成错误的对象或应用

程序的名称。可读/可写。

应用于

Err 对象

说明

Source 属性是字符串表达式，指定生成错误的对象；此表达式通常是这个对象的类名或程序设计的 ID。在程序代码无法处理被访问对象产生的错误时，请使用 Source 提供消息。例如，如果访问 MicrosoftExcel 时生成了一个“除以零”的错误，则 MicrosoftExcel 将 Err.Number 设置成代表此错误的错误代码，并将 Source 设置成 Excel.Application。

在错误生成时，Source 就是应用程序的程序设计 ID。对于类模块，Source 应该包含一个具有 project.class 窗体的名称。当代码中出现不可预料的错误时，Source 属性会自动填上数据。对于标准模块中的错误，Source 含有工程名称。对于类模块中的错误，Source 包含具有 project.class 窗体的名称。

请参阅

获取对象函数，HelpContextID 属性(VBA 外接程序对象模型)，OnError 语句，Err 对象，Description 属性，HelpContext 属性，HelpFile 属性，LastDLLError 属性，Number 属性

示例

本示例将某个由 VisualBasic 生成的 Automation 对象的程序设计标识符（ProgrammaticID）存到变量 MyObjectID 中，并在程序使

用 Raise 方法来生成一错误状态时，将之存入 Err 对象的 Source 属性中。在错误处理过程中，请不要在程序中使用 Source 属性（或除 Number 外的 Err 对象属性）。只有用户不能处理该错误时，才要使用 Number 以外的属性将详细的错误信息显示给用户。该示例假设 App 和 MyClass 是有效的引用。

```
DimMyClass,MyObjectID,MyHelpFile,MyHelpContext
'某个 MyClass 类型的对象生成一个错误，并设置 Err 对象
'的所有属性值，包括 Source（记录 MyObjectID－为 App 对象
'的 Title 属性及 MyClass 对象的 Name 属性之组合）。
MyObjectID=App.Title&"."&MyClass.Name
Err.Raise  Number:=vbObjectError+894,Source:=MyObjectID,_
    Description:="Wasnotabletocompleteyourtask",_
    HelpFile:=MyHelpFile,HelpContext:=MyHelpContext
```

SourceDoc 属性

返回或设置创建对象时使用的文件名。

注意设置 SourceDoc 属性是为了与早期版本的 Action 属性兼容。要获得目前的该功能，可使用 CreateEmbed 和 CreateLink 方法。

应用于

OLE 包容器控件

语法

object.**SourceDoc**[=*name*]

SourceDoc 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
name	一个指定文件名的字符串表达式

说明

当使用 Action 属性创建链接对象时，用 SourceDoc 属性指定要链接的文件。使用 SourceItem 属性指定在要链接的文件内的数据。

当使用 Action 属性创建内嵌对象时，如果 SourceDoc 属性被置为有效的文件名，使用指定的文件将嵌入的对象作为模板创建。

当创建链接对象时，SourceItem 属性与 SourceDoc 属性相连接。在运行时，SourceItem 属性返回一个零长度字符串("")，SourceDoc 属性返回链接文件的整个路径，之后是一个惊叹号(!)或反斜杠(\)，再之后是 SourceItem。例如：

```
"C:\WORK\QTR1\REVENUE.XLS!R1C1:R30C15"
```

请参阅

Class 属性，OLEType 属性，SourceItem 属性，OLETypeAllowed 属性，CreateEmbed 方法，CreateLink 方法，Action 属性(OLE 包容器控件)

SourceFile 属性

该属性指定了一个 HTML 文件的路径，该文件将被页面设计器装

入到设计器“细表”窗格中。它在运行时是不可使用的。

应用于

DHTMLPageDesigner 对象

说明

本属性既可以通过“页面属性”对话框设置，指定一个现有的文件作为页面源，也可以直接通过‘属性’窗口中的设计器 DHTMLPage 对象进行设置。系统将对其进行检查，以确定输入的文件路径是否代表一个有效的文件，然后将 HTML 页面的内容输入到设计器中。BuildFile 属性的初始路径设置值与 SourceFile 属性相同。

本属性中的源 HTML 文件必须是绝对路径，它需要指出完整的驱动器盘符和编译之后的文件所在的目录结构。如果将来需要与其他开发者共享自己的工程，而该开发者使用的目录结构或者驱动器名称与您同原来使用的不同，那么后来的开发者将需要修改 SourceFile 属性的值，使之反映他的计算机上的 HTML 文件的位置。运行时用户不会受到这种限制。

请参阅

《MicrosoftVisualBasic6.0 工具部件指南》中第五部分“开发 DHTML 应用程序”第二章的 BuildFile 属性的“构造 Internet 应用程序”。

SourceItem 属性

当创建链接对象时，返回或设置在要链接的文件内的数据。

应用于

OLE 包容器控件

语法

object.**SourceItem**[=*string*]

SourceItem 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	一个指定被链接数据的字符串表达式

说明

当使用这个属性时，必须将 LETypeAllowed 设置为 0（链接）或 2（均可）。使用 SourceDoc 属性指定要链接的文件。

每个对象都使用其自己的语法描述数据单元。为了设置属性，指定对象可识别的数据单元。例如，当与 MicrosoftExcel 链接时，使用象 R1C1 或 R3C4:R9C22 那样的单元格或单元格范围引用，或者象“税收”那样的命名范围，来指定 SourceItem。

为了确定描述对象的数据单元的语法，请参阅创建对象的应用程序的文档。

注意在设计时使用特殊粘贴命令（在 OLE 容器控件上单击鼠标右键），可以通过创建链接对象来确定这个语法。一旦创建了对象，在属性窗口选择 SourceDoc 属性，并查看在设置值框中的字符串。对于大多数对象，这个字符串包含链接文件的路径，其后是惊叹号(!)或反斜杠(\)及链接数据的语法。

当创建链接对象时，SourceItem 属性与 SourceDoc 属性相连接。在运行时，SourceItem 属性返回一个零长度字符串("")，SourceDoc 属性返回链接文件的整个路径，之后是一个惊叹号(!)或反斜杠(\)，再之后是 SourceItem。例如：

```
"C:\WORK\QTR1\REVENUE.XLS!R1C1:R30C15"
```

请参阅

Class 属性，OLEType 属性，SourceDoc 属性，OLETpyeAllowed 属性

Space 函数

返回特定数目空格的 Variant(String)。

语法

Space(*number*)

必要的 number 参数为字符串中想要的空格数。

说明

Space 函数在格式输出或清除固定长度字符串数据时很有用。

请参阅

Spc 函数，String 函数

示例

此例采用 Space 函数以返回特定长度的字符串

```
DimMystring
'返回占据 10 个空格的字符串
```

```
Mystring=Space(10)

'在两个字符串间插入 10 个空格
MyString=“Hello”&Space(10)&“World”
```

Spc 函数

与 Print#语句或 Print 方法一起使用，对输出进行定位。

语法

Spc(*n*)

必要的 n 参数是在显示或打印列表中的下一个表达式之前插入的空白数。

说明

如果 n 小于输出行的宽度，则下一个打印位置将紧接在数个已打印的空白之后。如果 n 大于输出行的宽度，则 Spc 利用下列公式计算下一个打印位置：

currentprintposition+(*n*Mod*width*)

例如，如果当前输出位置为 24，而输出行的宽度为 80，并指定了 Spc(90)，则下一个打印将从位置 34 开始（当前打印位置+90/80 的余数）。如果当前打印位置和输出行宽度之间的差小于 *n*（n 或 Mod*width*），则 Spc 函数会跳到下一行的开头，并产生数量为 n–(*width*–*urrentprintposition*)的空白。

注意要确保表格栏宽度足以容纳较宽的字符串。

当 Print 方法与间距字体一起使用时，使用 Spc 函数打印的空格字符的宽度总是等于选用字体内以磅数为单位的所有字符的平均宽度。但是，在已打印字符的个数与那些字符所占据的定宽列的数目之间不存在任何关系。例如，大写英文字母 W 占据超过一个定宽的列，而小写字母 i 则占据少于一个定宽的列。

请参阅

Print#语句，Tab 函数，Width#语句，Print 方法，Mod 操作符，Space 函数

示例

本示例使用 Spc 函数确定在文件或立即窗口中的输出位置。

```
'Spc 函数可以和 Print#语句一起使用。
Open"TESTFILE"ForOutputAs#1                    '打开输出文件。
Print#1,"10spacesbetweenhere"；Spc(10)；"andhere."
Close#1                                        '关闭文件。
```

下列语句在显示到调试窗口（使用 Print 方法）的文本之前加 30 个空格。

```
Debug.PrintSpc(30)；"Thirtyspaceslater..."
```

Split 函数

返回一个下标从零开始的一维数组，它包含指定数目的子字符串。

语法

Split(*expression*[,*delimiter*[,*count*[,*compare*]]])

Split 函数语法有如下几部分：

部分	描述
expression	必需的。包含子字符串和分隔符的字符串表达式。如果 expression 是一个长度为零的字符串("")，Split 则返回一个空数组，即没有元素和数据的数组
delimiter	可选的。用于标识子字符串边界的字符串字符。如果忽略，则使用空格字符("")作为分隔符。如果 delimiter 是一个长度为零的字符串，则返回的数组仅包含一个元素，即完整的 expression 字符串。
count	可选的。要返回的子字符串数，–1 表示返回所有的子字符串
compare	可选的。数字值，表示判别子字符串时使用的比较方式。关于其值，请参阅“设置值”部分

设置值

compare 参数的设置值如下：

常数	值	描述
vbUseCompareOption	–1	用 OptionCompare 语句中的设置值执行比较
vbBinaryCompare	0	执行二进制比较
vbTextCompare	1	执行文字比较
vbDatabaseCompare	2	仅用于 MicrosoftAccess。基于您的数据库的信息执行比较

请参阅

Join 函数

Sqr 函数

返回一个 Double，指定参数的平方根。

语法

Sqr(*number*)

必要的 number 参数 number 是 Double 或任何有效的大于或等于 0 的数值表达式。

请参阅

Derived 数学函数

示例

本示例使用 Sqr 函数来计算某数的平方根。

```
DimMySqr
```

```
MySqr=Sqr(4)    '返回 2。
MySqr=Sqr(23)   '返回 4.79583152331272。
MySqr=Sqr(0)    '返回 0。
MySqr=Sqr(-4)   '生成一个运行时错误（负数不能用此函数开平方根）。
```

StandardMethod 属性

返回某个 Member 对象的 StandardMethod 属性，或者对其进行设置。

应用于

Member 对象

语法

object.**StandardMethod**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

StandardSize 属性

将属性页设置为标准尺寸。

应用于

PropertyPage 对象

语法

object.**StandardSize**[=*value*]

StandardSize 属性的语法有下面这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
value	整数表达式，指定了所显示的鼠标指针类型，如下列“设置值”中所述

设置值

value 的设置值为：

常数	值	描述
custom	0	（缺省的）由对象所决定的尺寸
small	1	将 PropertyPageStandardSize 设置为 101 个像素高，375 个像素宽
large	2	将 PropertyPageStandardSize 设置为 179 个像素高，375 个像素宽

说明

当设为 1-小或 2-大时，属性页的属性可能不会给出期望的结果。该属性只影响被认可的显示器驱动程序和分辨率属性页的大小。其它显示器可能发生未预期的结果。

建议使用缺省 StandardSize 属性值（1-自定义），并象对待其它窗体一样对待属性页。

请参阅

PropertyPage 对象

Start 事件

如果在请求中没有指定 webItem 和事件，当用户从浏览器中发射一个 WebClass 对象时发生。在应用程序的存活期中，Start 事件在 BeginRequest 事件之后发生。

应用于

WebClass 对象

语法

PrivateSub*object*_Start()

object 所在处表示一个对象表达式，其值是“应用于”列表中的一个对象。

说明

在用户通过漫游构成应用程序基本 URL 的活动服务器页面来访问您的应用程序时，Start 事件发生。可以为 Start 事件编写代码，以使系统成功地运行您的应用程序。

注意如果用户在它们用来启动 webclass 的 URL 中包括了一个 webItem 时，Start 事件不会引发。运行时引发 BeginRequest 事件，然后启动指定的 webitem 及其缺省事件或用户指定的事件。

通常，应该使用 Start 事件定位到 WebClass 对象中的第一个 WebItem 上。Start 事件代码能够使用 NextItem 属性定位到 Webclass 中指定的 WebItem 上。

请参阅

第三章的“IIS 应用程序开发过程”、“WebClass 生存期”和“设置 StartEvent”，在《MicrosoftVisualBasic6.0 部件工具指南》第五部分的“开发 IIS 应用程序”的“构造 Internet 应用程序”中。

StartLogging 方法

对某个操作的日志目标及日志模式进行设置。

应用于

App 对象

语法

object.**StartLogging***logTarget*,*logMode*

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
logTarget	用来捕获 LogEvent 方法输出的文件的路径及其文件名
logMode	一个值，它决定如何记入日志（通过 LogEvent 方法）。请看下列“设置值”

设置值

logMode 的设置值是：

常数	值	描述
vbLogAuto	0	如果在 Windows95 上运行，则该选项将把日志消息记入到 LogFile 属性所指定的文件中。如果在 WindowsNT 上运行，则将日志消息连同应用程序记入到 NT 应用程序 EventLog 中。"VBRunTime"用作应用程序源 andApp.Titleappearinginthedescription
vbLogOff	1	关闭所有的记入日志。忽略并放弃来自 UI 分路以及 LogEvent 方法中的消息
vbLogToFile	2	强制记入到一个文件中。如果在 LogPath 中未出现有效文件名，那么忽略记入日志，而将属性设置为 vbLogOff
vbLogToNT	3	强制记入到 NT 事件日志中。如果不是在 WindowsNT 上运行，或者事件日志无效，则忽略记入日志，并将属性设置为 vbLogOff
vbLogOverwrite	16	指示每次启动应用程序时，应该重新创建日志文件。这个值可与使用 OR 操作符的其它模式选项结合使用。将记入日志的缺省动作追加到现存文件中。在 NT 事件记入日志的情况下，这个标记没有任何意义
vbLogThreadID	32	指示在窗体"[T:0nnn]"中，当前线程 ID 被考虑

到消息中。这个值可与使用 OR 操作符的其它模式选项结合使用。缺省动作表明仅当应用程序是多线程时（要么明确标记为线程安全的，要么作为不明确的多线程应用程序～比如一个其实例属性设置为单用户、多线程的本地服务器来实现），才显示线程 ID

StartMode 属性

该属性返回或设置工程的启动模式。

应用于

VBProject 对象

语法

object.StartMode

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

如果代码当前正在执行则返回 RunMode；如果代码在堆栈中但没执行则返回 BreakMode，否则返回 DesignMode。

StartMode 属性

返回或设置一个值，该值用来决定一个应用程序是被当作一个独立的工程还是一个 ActiveX 部件来启动。在运行时是只读的。

应用于

App 对象

语法

object.**StartMode**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

设置值

StartMode 属性的设置值为：

常数	值	描述
vbSModeStandalone	0	（缺省值）应用程序被当作一个独立的工程来启动
vbSModeAutomation	1	应用程序被当作一个 ActiveX 部件来启动

说明

这些常数在 VisualBasic(VB)中的对象浏览器中的对象库中被列出。

在设计时，可以通过在“工程选项”对话框中将 StartMode 设置为 1(vbSModeAutomation)来调试一个应用程序，把它当作好象是一个 ActiveX 部件被启动一样。

一旦一个工程被编译，StartMode 属性的值就决定于那个应用程序如何启动，而不是被“工程选项”对话框中其正常的设置值来决定。

当 StartMode 被设置为 1 并且在工程中没有公共的类时，必须使用 End 语句或从“运行”菜单或工具条上选择“结束”来结束该应用程序。如果从“系统”菜单中选择“关闭”来结束该应用程序，那么窗体虽然被关闭，但是工程仍然在运行。

请参阅

EXEName 属性，HelpFile 属性(App，CommonDialog，Menuline)，PrevInstance 属性，Title 属性，Path 属性

示例

该例子显示在设计时将 StartMode 属性设为 1(vbSModeAutomation)的一个可能的影响。创建一个 ActiveXEXE 工程。创建一个新的窗体。从“工程”菜单中选择“工程属性”命令。选中“部件”标签，然后在“启动模式”组中选中“ActiveX 部件”选项按钮。选择“确定”关闭“工程属性”对话框。要试用此例，可以先将该代码粘贴到窗体的声明部分中，然后按下 F5 键并双击窗体的标题栏左边的“控件”菜单。如果窗体不显示，则在立即窗口中输入 Form1.Show。

```
PrivateSubForm_QueryUnload(CancelAsInteger,UnloadModeAsInteger)
    IfUnloadMode=vbFormControlMenuAndApp.StartMode=vbSModeAutomat
ionThen

    Msg="Formwillclosebutapplicationwillstillberunning."&Chr(10)

    Msg=Msg+"Toterminateapplicationwithoutapublicclass,"&Chr(10)
```

```
        Msg=Msg+"youmustuseanEndstatement."
        MsgBoxMsg
    EndIf
EndSub
```

StartProject 属性

该属性返回或设置工程，该工程在从“运行”菜单中选择“启动”或按 F5 键时启动。

应用于

VBProjects 集合

语法

object.**StartProject**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

StartUpObject 属性

该属性返回或设置工程的启动部件。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

object.**StartUpObject**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

返回值

该返回的值是一个变体，它或者包含类型 vbext_StartupObject 的计数值，或者包含代表启动对象的 VBComponent 对象。

vbext_StartUpObject 的 StartUpObject 属性的设置值是：

常数	值	描述
vbext_so_SubMain	0	启动对象是 SubMain（子程序）
vbext_so_None	1	没有启动对象

说明

只有在运行时可见的工程项才能是启动对象。

StartUpPosition 属性

返回或设置一个值，指定对象首次出现时的位置，运行时不能使用。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

object.**StartUpPosition**=*position*

StartUpPosition 属性的语法有这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
startUpPosition	整数，规定当对象在“设置值”中显示时的位置

设置值

可用 StartUpPosition 四个设置值中的一个：

常数	值	描述
vbStartUpManual	0	没有指定初始设置值
vbStartUpOwner	1	UserForm 所属的项目中央
vbStartUpScreen	2	屏幕中央
vbStartUpWindowsDefaul t	3	屏幕的左上角

StateManagement 属性

设置或返回 WebClass 的状态类型。

应用于

WebClass 对象

语法

object.StateManagement[=*enum*]

部分	描述
Object	对象表达式，其值是“应用于”列表中的一个对象
Enum	决定 WebClass 如何维护状态的常数，如“设置值”中所描述的

设置值

enum 的设置值是:

设置值	描述
1-wcNoState	当接收到一个 HTTP 请求时，创建一个 WebClass 实例。该实例被用于 HTTP 请求期间，然后被破坏。任何 WebClass 需要的状态都必须在外部存储，例如，存储在一个数据库管理系统中。此外，该状态的状态或关键信息可以存储在 Session 对象中、使用 URLData 属性存储在 URL 自身或使用 Response.Cookies 集合存储在存储块
2-cRetainInstance	当接收到第一个 HTTP 请求时，创建 WebClass 对象实例。该实例直到 WebClass 对象调用 ReleaseInstance 或活动服务器页面对话超时时才被破坏。状态能够被安全地保留在 WebClass 对象的属性中；然而，这将增加应用程序的大小并降低应用程序的可缩放性

请参阅

第三章的“TheWebClass 生存期”和“IIS 应用程序状态管理”，在《MicrosoftVisualBasic6.0 部件工具指南》第五部分“IIS 应用程序”的“构造 Internet 应用程序”中。

Static 属性

该属性返回引用的变量或方法是否被声明为 Static。

应用于

Member 对象

语法

object.**Static**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

Static 语句

在过程级别中使用，用于声明变量并分配存储空间。在整个代码运行期间都能保留使用 Static 语句声明的变量的值。

语法

Static*varname*[([*subscripts*])][**As**[**New**]*type*][,*varname*[([*subscripts*])][**As**[**New**]*type*]]...

Static 语句的语法包含下面部分：

部分	描述
varname	必需的。变量的名称；遵循标准变量命名约定
subscripts	可选的。数组变量的维数；最多可以定义 60 维的多维数组 subscripts 参数使用下面的语法： [*lower*To]*upper*[,[*lower*To]*upper*]... 如果不显式指定 lower，则数组的下界由 OptionBase 语句控制。如果没有 OptionBase 语句则下界为 0
new	可选的。用它可以隐式地创建对象的关键字。如果使用 New 声明对象变量，则在第一次引用该变量时将新建该对象的实例，因此不必使用 Set 语句来对该对象引用赋值。New 关键字不能用来声明任何内部数据类型的变量，也不能用来声明从属对象的实例
type	可选的。变量的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（对变长的字符串）、String*length（对定长的字符串）、Object、Variant、用户定义类型或对象类型。所声明的每个变量都要有一个单独的 Astype 子句

说明

模块的代码开始运行后，使用 Static 语句声明的变量会一直保持其值，直至该模块复位或重新启动。可以在非静态的过程中使用

Static 语句显式声明只在该过程内可见，但具有与包含该过程定义的模块相同生命期的变量。
可以在过程中使用 Static 语句来声明在过程调用之间仍能保持其值的变量的数据类型。例如，下面的语句声明了一个定长的整型数组：

```
StaticEmployeeNumber(200)AsInteger
```

下面的语句为 worksheet 的新实例声明了一个变量：

```
StaticXAsNewWorksheet
```

如果在定义对象变量时没有使用 New 关键字，则在使用该变量之前，必须使用 Set 语句将一个已有的对象赋给这个引用对象的变量。在被赋值之前，所声明的这个对象变量有一个特定值 Nothing，这个值表示该变量没有指向任何对象的实例。若在声明中使用了 New 关键字，则在第一次引用对象时将新建一个该对象的实例。
如果不指定数据类型或对象类型，且在模块中没有使用 Deftype 语句，则按缺省情况，定义该变量为 Variant 类型。
注意 Static 语句与 Static 关键字很相似，但是针对不同的效果来使用的。如果使用 Static 关键字（如 StaticSubCountSales()）来声明一个过程，则该过程中的所有局部变量的存储空间都只分配一次，且这些变量的值在整个程序运行期间都存在。对非静态过程而

言，该过程每次被调用时都要为其变量分配存储空间，当该过程结束时都要释放其变量的存储空间。Static 语句则用来在非静态的过程中声明特定的变量，以使其在程序运行期间能保持其值。

在初始化变量时，数值变量被初始化为 0，变长的字符串被初始化为一个零长度的字符串("")，而定长的字符串则用 0 填充。Variant 变量被初始化为 Empty。用户自定义类型的变量的每个元素作为各自独立的变量进行初始化。

注意如果在过程中使用 Static 语句，应和其它的声明语句（如 Dim）一样将其放在过程的开始。

请参阅

Data 类型模式，Array 函数，Dim 语句，Function 语句，Option 基本语句，Private 语句，Public 语句，ReDim 语句，Sub 语句

示例

该示例使用 Static 语句，可以在模块代码运行期间一直保持变量的值。

```
'函数定义。
FunctionKeepTotal(Number)
    '只有 Accumulate 变量能在调用过程之间保持其值。
    StaticAccumulate
    Accumulate=Accumulate+Number
    KeepTotal=Accumulate
EndFunction
```

```
'Static 函数定义。
StaticFunctionMyFunction(Arg1,Arg2,Arg3)
   '所有局部变量在函数调用之间都能保持其值。
   Accumulate=Arg1+Arg2+Arg3
   Half=Accumulate/2
   MyFunction=Half
EndFunction
```

Status 属性（AsyncProperty 对象）

返回关于一个 AsyncRead 操作的特定协议的状态信息。

应用于

AsyncProperty 对象

语法

object.**Status**

Status 属性语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象

说明

Status 属性返回一个对 StatusCode 属性特定的字符串并且可能为空串。例如，对 vbAsyncStatusCodeBeginConnecting，字符串为 TCP/IP 的端口地址。

StatusCode 属性

返回一个常数，指出一个 AsyncRead 操作的当前状态。

应用于

AsyncProperty 对象

语法

object.**StatusCode**

StatusCode 属性语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象

设置值

StatusCode 属性可能的返回值列举如下：

常数	设置值	描述
vbAsyncStatusCodeError	0	在异步下载过程中发生一个错误。要查找错误，请检查 AsyncProp 对象的 Value 属性
vbAsyncStatusCodeFindingResource	1	AsyncRead 正在查找 Status 属性指定的拥有正被下载的

		存储资源
vbAsyncStatusCodeRedirecting	2	AsyncRead 正在与 Status 属性指定的拥有被下载的存储资源连接
vbAsyncStatusCodeRedirecting	3	AsyncRead 已经变更到 Status 属性指定的另外地方
vbAsyncStatusCodeBeginDownloadData	4	AsyncRead 已开始往 Status 属性指定的存储接收数据
vbAsyncStatusCodeDownloadingData	5	对 Status 属性指定的存储 AsyncRead.已收到足够多的数据
vbAsyncStatusCodeEndDownloadData	6	对 Status 属性指定的存储 AsyncRead 已完成接受数据
vbAsyncStatusCodeUsingCachedCopy	10	由于 Status 属性是空值 AsyncRead 正在从缓存复本中获取被请

		求的存储
vbAsyncStatusCodeSendingRequest	11	AsyncRead 正在请求
		Status
		属性指定的存储
vbAsyncStatusCodeMIMETypeAvailable	13	被请求存储的 MIME 类型在 Status 属性中指定
vbAsyncStatusCodeCacheFileNameAvailable	14	用于请求请求存储的本地高
		速缓存文件的文件名
		在 Status 属性中指定
vbAsyncStatusCodeBeginSyncOperation	15	AsyncRead 将同步操作
vbAsyncStatusCodeEndSyncOperation	16	AsyncRead 已经完成同步操作

说明

StatusCode 常数列表仅包含 VisualBasic 常数。其它不是 Visual Basic 或 AsyncRead 方法特有的常数可在 InternetExplorerSDK 的 BINDSTATUS 标志下找到。

StdDataFormat 对象

StdDataFormat 对象允许在从数据库中读出或向数据库中写入数据

时，对数据进行格式化。

语法

StdDataFormat

说明

StdDataFormat 对象是数据绑定所不可缺少的部分。无论您是否在代码中创建它，它都将为所有的绑定控件而创建。

属性

FalseValue 属性，FirstDayOfWeek 属性，FirstWeekOfYear 属性，Format 属性(StdDataFormat 对象)，NullValue 属性，TrueValue 属性，Type 属性

事件

Format 事件，Unformat 事件

请参阅

StdDataFormats 集合，StdDataValue 对象

StdDataFormats 集合

StdDataFormats 集合含有一个或多个 StdDataFormat 对象。

说明

复杂绑定对象可以被绑定到多个 StdDataFormat 对象上。例如，DataGrid 的每一列都有一个绑定。StdDataFormats 集合提供了一个顶层对象，通过它可以访问多个格式化对象。

属性

Count 属性(VB 集合)

方法

Add 方法(格式对象)，Clear 方法(格式对象)，Item 方法，Remove 方法

请参阅

StdDataFormat 对象，StdDataValue 对象

Stop 语句

暂停执行。

语法

Stop

说明

可以在过程中的任何地方放置 Stop 语句，使用 Stop 语句，就相当于在程序代码中设置断点。

Stop 语句会暂停程序的执行，但是它不像 End，因为 Stop 不会关闭任何文件，或清除变量，除非它是以编译后的可执行文件(.exe)方式来执行。

请参阅

End 语句

示例

本示例使用 Stop 语句来暂停 For...Next 循环里的每一次完成。

```
Dim I
```

```
For I=1 To 10 '开始 For...Next 循环。
    Debug.Print I '将 I 的值显示到“立即”窗口。
    Stop '每一次的完成都会在此暂停。
Next I
```

StrComp 函数

返回 Variant(Integer)，**为字符串比较的结果。**

语法

StrComp(*string1*,*string2*[,*compare*])

StrComp **函数的语法有下面的命名参数：**

部分	**说明**
String1	**必需的。任何有效的字符串表达式**
String2	**必需的。任何有效的字符串表达式**
Compare	**可选的。指定字符串比较的类型。如果** compare **参数是** Null，**将发生错误。如果省略** compare，OptionCompare **的设置将决定比较的类型**

设置

compare **参数设置为：**

常数	值	描述
vbUseCompareOption	-1	使用 OptionCompare 语句设置执行一个比较
vbBinaryCompare	0	执行一个二进制比较
vbTextCompare	1	执行一个按照原文的比较
vbDatabaseCompare	2	仅适用于 MicrosoftAccess，执行一个基于数据库信息的比较

返回值

StrComp 函数有下列返回值：

如果	StrComp 返回
string1 小于 *string2*	-1
string1 等于 *string2*	0
string1 大于 *string2*	1
string1 或 *string2* 为 *Null*	Null

请参阅

Option 比较语句，InStr 函数，CompareMode 属性

示例

本示例使用 StrComp 函数来比较两个字符串。如果第三个参数值为 1，字符串是以文本比较的方式进行比较；如果第三个参数值为 0 或是缺省，则以二进制比较的方式进行比较。

文本比较方式会将大小写字母视为一样，但二进制比较方式则视

为不同。

```
DimMyStr1,MyStr2,MyComp
MyStr1="ABCD":MyStr2="abcd"     '定义变量。
MyComp=StrComp(MyStr1,MyStr2,1)     '返回0。
MyComp=StrComp(MyStr1,MyStr2,0)     '返回-1。
MyComp=StrComp(MyStr2,MyStr1)  '返回1。
```

StrConv 函数

返回按指定类型转换的 Variant(String)。

语法

StrConv(*string,conversion,LCID***)**

StrConv 函数的语法有下面的命名参数：

部分	说明
string	必需的。要转换的字符串表达式
conversion	必需的。Integer。其值的和决定转换的类型
lcid	可选的。如果与系统 LocaleID 不同，则为 LocaleID（系统 LocaleID 为缺省值）

设置值

conversion 参数的设置值为：

常数	值	说明
vbUpperCase	1	将字符串文字转成大写
vbLowerCase	2	将字符串文字转成小写
vbProperCase	3	将字符串中每个字的开头字母转成大写
vbWide*	4*	将字符串中单字节字符转成双字节字符
vbNarrow*	8*	将字符串中双字节字符转成单字节字符
vbKatakana**	16**	将字符串中平假名字符转成片假名字符
vbHiragana**	32**	将字符串中片假名字符转成平假名字符
vbUnicode	64	根据系统的缺省码页将字符串转成 Unicode。(在 Macintosh 中不可用)
vbFromUnicode	128	将字符串由 Unicode 转成系统的缺省码页。(在 Macintosh 中不可用)

* 应用到远东国别。

** 仅应用到日本。

注意这些常数是由 VBA 指定的。可以在程序中使用它们来替换真正的值。其中大部分是可以组合的，例如 vbUpperCase+vbWide，互斥的常数不能组合，例如 vbUnicode+vbFromUnicode。当在不适用的国别使用常数 vbWide、vbNarrow、vbKatakana，和 vbHiragana 时，就会导致运行时错误。

下面是一些一般情况下的有效分界符：Null(Chr$(0))，水平制表符(Chr$(9))，换行(Chr$(10))，垂直制表符(Chr$(11))，换页

(Chr$(12))，回车(Chr$(13))，空白(SBCS)(Chr$(32))。在 DBCS 中，空白的实际值会随国家/地区而不同。

说明

在把 ANSI 格式的 Byte 数组转换为字符串时，您应该使用 StrConv 函数。当您转换 Unicode 格式的这种数组时，使用赋值语句。

请参阅

类型转换函数，String 数据类型，Chr 函数

Stretch 属性

返回或设置一个值，该值用来指定一个图形是否要调整大小，以适应与 Image 控件的大小。

应用于

Image 控件

语法

object.**Stretch**[=*boolean*]

Stretch 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
Boolean	一个用来指定是否要调整图形的大小的布尔表达式，按照设置值的描述

设置值

boolean 的设置值为：

设置值	描述
True	表示图形要调整大小以与控件相适合
False	（缺省值）表示控件要调整大小以与图形相适

说明

如果 Stretch 被设置为 True，那么，控件大小的调整使得它所包含的图形的大小也要调整。

请参阅

Picture 属性，Image 属性，Picture 属性(AciveX 控件)

示例

本例从一个图标目录将一个箭头图标装入 Image 控件。当 Stretch 属性设置为 True 时，该箭头将缓慢地移过窗体，而当 Stretch 属性设置为 False 时，该箭头将跳跃式通过窗体。要试用此例，先将下面的代码粘贴到包含一个 Image 控件、一个 CheckBox 控件、以及一个 Timer 控件的窗体的声明部分，然后按 F5 键并单击该窗体。应确认图标目录所在的路径，必要时可以改变。单击 CheckBox，然后再次单击窗体，就可以看到 Stretch 属性的影响。

```
DimImgW                                    '声明变量。
PrivateSubForm_Load()
    '将图标装入 Image 控件。
    Image1.Picture=LoadPicture("ICONS\ARROWS\ARW02RT.ICO")
    Image1.Left=0                          '将图象移动到左部边界处。
```

```
    ImgW=Image1.Width                    '保存图象的宽度。
    Timer1.Interval=300
    Timer1.Enabled=False                 '关闭计时器。
    Check1.Caption="StretchProperty"
EndSub

PrivateSubForm_Click()
    Timer1.Enabled=True                  '打开计时器。
EndSub

PrivateSubTimer1_Timer()
    StaticMoveIconAsInteger              '移动图标的标志。
    IfNotMoveIconThen
        Image1.MoveImage1.Left+ImgW,Image1.Top,ImgW*2
    Else
        '移动图象并使它返回原来的宽度。
        Image1.MoveImage1.Left+ImgW,Image1.Top,ImgW
    EndIf
    '如果图象在窗体的边界之外，那么将导致结束。
    IfImage1.Left>ScaleWidthThen
        Image1.Left=0
        Timer1.Enabled=False
```

```
    EndIf
    MoveIcon=NotMoveIcon          '复位标志。
EndSub

PrivateSubCheck1_Click()
    Image1.Stretch=Check1.Value
EndSub
```

StrikeThrough 属性

返回或设置 Font 对象的字形为删除线或无删除线。

应用于

Font 对象

语法

object.**StrikeThrough** [=*boolean*]

StrikeThrough 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定字形的布尔表达式，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
True	打开删除线格式化
False	（缺省值）关闭删除线格式化

说明

Font 对象在设计时不能直接使用。取而代之的是，在“属性”窗口中通过选择控件的 Font 属性并单击“属性”按钮来设置 StrikeThrough 属性。在“字体”对话框中，选择“删除线”复选框。在运行时，通过为 Font 对象指定 StrikeThrough 属性值可直接设置 StrikeThrough。

请参阅

Bold 属性，FontTransparent 属性，Italic 属性，Size 属性(字体)，Underline 属性，Weight 属性，Name 属性

示例

本例用鼠标单击打印表单中的文本。要试本例，把代码粘贴到表单的声明部分，然后按 F5 并击表单两次。

```
privateSubForm_Click()
Font.Bold=NotFont.Bold.Togglebold.
Font.StrickThrough=NotFont.StrikeThroughTogglestrikethrough.
Font.Italic=NotFont.Italic.Toggleitalic.
Font.Underline=NotFont.UnderLine.Toggleunderline.
```

```
Font.Size=16.SetSize 属性.
IfFont.BoldThen
Print“Fontweightis”&Font.Weight&“(Bold).”
Else
Print“Fontweightis”&Font.Weight&“(notbold).”
EndIf
EndSub
```

String 函数

返回 Variant(String)，**其中包含指定长度重复字符的字符串。**

语法

String(*number*,*character*)

String **函数的语法有下面的命名参数：**

部分	**说明**
number	**必需的。**Long。**返回的字符串长度。如果** number **包含** Null，**将返回** Null
character	**必需的。**Variant。**为指定字符的字符码或字符串表达式，其第一个字符将用于建立返回的字符串。如果** character **包含** Null，**就会返回** Null

说明

如果指定 character **的数值大于 255，**String **会按下面的公式将其转为有效的字符码：**

*character*Mod256

请参阅

String 数据类型，Mod 操作符，Space 函数

示例

本示例使用 String 函数来生成一指定长度，且只含单一字符的字符串。

```
DimMyString
MyString=String(5,"*") '返回"*****"。
MyString=String(5,42)  '返回"*****"。
MyString=String(10,"ABC")  '返回"AAAAAAAAAA"。
```

StrReverse 函数

返回一个字符串，其中一个指定子字符串的字符顺序是反向的。

语法

StrReverse(*string1*)

参数 string1 是一个字符串，它的字符顺序要被反向。如果 string1 是一个长度为零的字符串(""),则返回一个长度为零的字符串。如果 string1 为 Null，则产生一个错误。

Style 属性

返回或设置一个值，该值用来指示控件的显示类型和行为。在运

行时是只读的。

应用于

DBCombo 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，
ListBox 控件，OptionButton 控件

语法

object.**Style**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

设置值

对于 Checkbox、CommandButton 和 OptionButton 控件，Style 属性的设置值为：

常数	值	描述
vbButtonStandard	0	（缺省的）标准的。控件按它们在 VisualBasic 老版本中的样子显示。也就是，Checkbox 控件显示为在其身旁有一个标签的复选框，OptionButton 显示为在其身旁有一个标签的选项按钮，而 CommandButton 显示为标准的、没有相关图形的 CommandButton
vbButtonGraphical	1	图形的。控件用图形的样式显示。即，Checkbox 控件显示为类似按钮的 CommandButton，它能上下切换；OptionButton 显示为类似按钮的 CommandButton，它保持向上或向下的切换，直到它的选项群组内的另一个 OptionButton 被选中；而 CommandButton 显示为标准的、也能显示相关图形的 CommandButton

对于 ComboBox 控件，Style 属性值为：

常数	值	描述
vbComboDropDown	0	（缺省值）下拉式组合框。包括一个下拉式列表和一个文本框。可以从列表选择或在文本框中输入
vbComboSimple	1	简单组合框。包括一个文本框和一个不能下拉的列表。可以从列表中选择或在文本框中输入。简单组合框的大小包括编辑和列表部分。按缺省规定，简单组合框的大小调整在没有任何列表显示的状态。增加 Height 属性值可显示列表的更多部分
vbComboDrop-downList	2	下拉式列表。这种样式仅允许从下拉式列表中选择

对于 ListBox 控件，Style 属性值为：

常数	值	描述
vbListBoxStandard	0	（缺省值）标准的。ListBox 控件按它在 VisualBasic 老版本中的样子显示；即，象是文本项的列表
vbListBoxCheckbox	1	复选框。在 ListBox 控件中，每一个文本项的边上都有一个复选框。在 ListBox 中可以选择多项

说明

对于 ComboBox 控件，根据下面这些原则来决定选用哪种设置值：使用设置值 0（下拉式组合框）或设置值 1（简单组合框）来给用户一选择列表。每种风格都使用户能在文本框中输入一个选择。设置值 0 能节省窗体上的空间，因为列表部分在用户选择一个项时将关闭。

使用设置值 2（下拉式列表）能显示一个从中选择一个项的固定选择列表。列表部分在用户选择一个项时关闭。

请参阅

Change 事件, Click 事件, DropDown 事件, DblClick 事件, Change 事件(ActioveX 控件)，Click 事件(ActiveX 控件)，DblClick 事件(ActiveX 控件)

Sub 语句

声明子过程的名称，参数，以及构成其主体的代码。

语法

[**Private**|**Public**|**Friend**][**Static**]*Subname*[(*arglist*)]

[*statements*]

[**ExitSub**]

[*statements*]

EndSub

Sub 语句的语法包含下面部分：

部分	描述
public	可选的。表示所有模块的所有其它过程都可访问这个 Sub 过程。如果在包含 OptionPrivate 的模块中使用，则这个过程在该工程外是不可使用的
private	可选的。表示只有在包含其声明的模块中的其它过程可以访问该 Sub 过程
friend	可选的。只能在类模块中使用。表示该 Sub 过程在整个工程中都是可见的，但对对象实例的控制者是不可见的
static	可选的。表示在调用之间保留 Sub 过程的局部变量的值。Static 属性对在 Sub 外声明的变量不会产生影响，即使过程中也使用了这些变量
name	必需的。Sub 的名称；遵循标准的变量命名约定
arglist	可选的。代表在调用时要传递给 Sub 过程的参数的变量列表。多个变量则用逗号隔开
statements	可选的。Sub 过程中所执行的任何语句组

其中的 *arglist* 参数的语法以及语法各个部分如下：

[Optional][ByVal|ByRef][ParamArray]*varname*[()][As*type*][=*defaultvalue*]

部分	描述
Optional	可选的。表示参数不是必需的关键字。如果使用了该选项，则 arglist 中的后续参数都必须是可选的，而且必须都使用 Optional 关键字声明。如果使用了 ParamArray，则任何参数都不能使用 Optional
ByVal	可选的。表示该参数按值传递
ByRef	可选的。表示该参数按地址传递。ByRef 是 VisualBasic 的缺省选项
ParamArray	可选的。只用于 arglist 的最后一个参数，指明最后这个参数是一个 Variant 元素的 Optional 数组。使用 ParamArray 关键字可以提供任意数目的参数。ParamArray 关键字不能与 ByVal，ByRef，或 Optional 一起使用
varname	必需的。代表参数的变量的名称；遵循标准的变量命名约定
type	可选的。传递给该过程的参数的数据类型，可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（只支持变长）、Object 或 Variant。如果没有选择参数 Optional，则可以指定用户定义类型，或对象类型

defaultvalue	可选的。任何常数或常数表达式。只对 Optional 参数合法。如果类型为 Object，则显式的缺省值只能是 Nothing

说明

如果没有使用 Public、Private 或 Friend 显式指定，Sub 过程按缺省情况就是公用的。如果没有使用 Static，则在调用之后不会保留局部变量的值。Friend 关键字只能在类模块中使用。不过 Friend 过程可以被工程的任何模块中的过程访问。Friend 过程不会在其父类的类型库中出现，且 Friend 过程不能被后期绑定。

小心 Sub 过程可以是递归的；也就是说，该过程可以调用自己来完成某个特定的任务。不过，递归可能会导致堆栈上溢。通常 Static 关键字和递归的 Sub 过程不在一起使用。

所有的可执行代码都必须属于某个过程。不能在别的 Sub、Function 或 Property 过程中定义 Sub 过程。

ExitSub 语句使执行立即从一个 Sub 过程中退出。程序接着从调用该 Sub 过程的语句下一条语句执行。在 Sub 过程的任何位置都可以有 ExitSub 语句。

Sub 过程与 Function 过程的相似之处是：它们都是一个可以获取参数，执行一系列语句，以及改变其参数的值的独立过程。而与 Function 过程不同的是：带返回值的 Sub 过程不能用于表达式。

可以使用过程名并后跟相应的参数列表来调用一个 Sub 过程。请参阅 Call 语句关于如何调用 Sub 过程的详细的说明信息。

在 Sub 过程中使用的变量分为两类：一类是在过程内显式定义的，

另一类则不是。在过程内显式定义的变量（使用 Dim 或等效方法）都是局部变量。对于使用了但又没有在过程中显式定义的变量，除非其在该过程外更高级别的位置有显示地定义，否则也是局部的。

小心过程可以使用没有在过程内显式定义的变量，但只要有任何在模块级别定义的名称与之同名，就会产生名称冲突。如果过程中使用的未定义的变量与别的过程，常数，或变量的名称相同，则认为过程使用的是模块级的名称。显式定义变量就可以避免这类冲突。可以使用 OptionExplicit 语句来强制显式定义变量。

注意不能使用 GoSub、GoTo 或 Return 来进入或退出 Sub 过程。

请参阅

Call 语句，Dim 语句，Function 语句，Option 显式语句，Property 获得语句，Property 允许语句，Property 设置语句，AddressOf 操作符，Friend

示例

该示例使用 Sub 语句来定义子过程的名称、参数、以及构成子过程主体的代码。

```
'子过程的定义。
'子过程带有两个参数。
SubSubComputeArea(Length,TheWidth)
    DimAreaAsDouble                    '声明局部变量。
    IfLength=0OrTheWidth=0Then
```

```
    '如果有一个参数=0，
        ExitSub                    '就立即退出子过程。
    EndIf
    Area=Length*TheWidth           '计算矩形的面积。
    Debug.PrintArea                    '将面积显示在调试窗口。
EndSub
```

SubFolders 属性

返回包含所有文件夹的一个 Folders 集合，这些文件夹包含在某个特定的文件夹中，包括设置了隐藏和系统文件属性的那些文件夹。

应用于

Folder 对象

语法

object.**SubFolders**

object 总是一个 Folder 对象。

说明

下面的代码举例说明了 SubFolders 属性的用法：

```
SubShowFolderList(folderspec)
Dimfs,f,f1,s,sf
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFolder(folderspec)
```

```
Setsf=f.SubFolders
ForEachf1insf
s=s&f1.name
s=s&vbCrLf
Next
MsgBoxs
EndSub
```

请参阅

Folders 集合，Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Drives 属性，Files 属性，IsRootFolder 属性，Name 属性(FIleSystemObject 对象)，ParentFolder 属性，Path 属性(FileSystemObject 对象)，ShortName 属性，ShortPath 属性，Size 属性(FileSysTemObject 对象)，Type 属性

SupportsMnemonics 属性

返回一个布尔值，它规定控件的容器是否处理控件的访问键。

应用于

AmbientProperties 对象

语法

object.**SupportsMnemonics**

SupportsMnemonics 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

设置值

SupportsMnemonics 属性可能的布尔返回值为：

设置值	描述
True	控件的容器要处理访问键
False	控件的容器不处理访问键。如果容器不实现这种环境属性，这将是缺省设置值

说明

大多数控件的容器都能够处理它所包含的控件的所有访问键。这当中包括判断哪个控件要赋予特殊的访问键。如果容器不能处理访问键，SupportsMnemonics 属性会指出这一点，而且控件可以相应采取某些行动，例如不显示作为键盘加速键指示的下划线字符。

Switch 函数

计算一组表达式列表的值，然后返回与表达式列表中最先为 True 的表达式所相关的 Variant 数值或表达式。

语法

Switch(*expr*-1,*value*-1[,*expr*-2,*value*-2…[,*expr-n*,*value-n*]])

Switch 函数的语法具有以下几个部分：

部分	描述
expr	必需的。要加以计算的 Variant 表达式
value	必需的。如果相关的表达式为 True，则返回此部分的数值或表达式

说明

Switch 函数的参数列表由多对表达式和数值组成。表达式是由左至右加以计算的，而数值则会在第一个相关的表达式为 True 时返回。如果其中有部分不成对，则会产生一个运行时错误。如果 *expr-1* 为 True 则 Switch 返回 *value-1*，如果 *expr-1* 为 False，但 *expr-2* 为 True，则 Switch 返回 *value-2*，以此类推。

Switch 会返回一个 Null 值，如果：

没有一个表达式为 True。

第一个为 True 的表达式，其相对应的值为 Null。

虽然它只返回其中的一个值，但是 Switch 会计算所有的表达式。因此应该注意到所产生的副作用。例如，只要其中一个表达式导致被零除错误，就会发生错误。

请参阅

Choose 函数，IIf 函数，Select 事例语句

示例

本示例使用 Switch 函数返回和城市名称匹配的语言名称。

```
FunctionMatchUp(CityNameAsString)
```

```
Matchup=Switch(CityName="London","English",CityName_
                    ="Rome","Italian",CityName="Paris","French")
EndFunction
```

SYD 函数

返回一个 Double，指定某项资产在一指定期间用年数总计法计算的折旧。

语法

SYD(*cost*,*salvage*,*life*,*period*)

SYD 函数有下列命名参数：

部分	描述
cost	必需的。Double 指定资产的初始成本
salvage	必需的。Double 指定资产在可用年限结束后的价值
life	必需的。Double 指定资产的可用年限
period	必需的。Double 指定计算资产折旧所用的那一期间

说明

必须用相同的单位表示 life 和 period 参数。例如，如果 life 用月份表示，则 period 也必须用月份表示。所有参数都必须是正数。

请参阅

DDB 函数，FV 函数，IPmt 函数，IRR 函数，MIRR 函数，NPer 函数，NPV 函数，Pmt 函数，PPmt 函数，PV 函数，Rate 函数，SLN 函数

示例

本示例使用 SYD 函数计算某项资产在指定期间内的折旧，计算时需给定资产初始成本（InitCost），资产在可用年限退出后的价值（SalvageVal）及资产的可用年限（LifeTime）。计算折旧的以年为单位的期间为 PDepr。

```
DimFmt,InitCost,SalvageVal,MonthLife,LifeTime,DepYear,PDepr
ConstYEARMONTHS=12 '一年之中的月份数。
Fmt="###,##0.00"   '定义金额格式。
InitCost=InputBox("What'stheinitialcostoftheasset?")
SalvageVal=InputBox("What'stheasset'svalueattheendofitslife?")
MonthLife=InputBox("What'stheasset'susefullifeinmonths?")
DoWhileMonthLife<YEARMONTHS    '确保计算期间大于等于一年。
   MsgBox"Assetlifemustbeayearormore."
   MonthLife=InputBox("What'stheasset'susefullifeinmonths?")
Loop
LifeTime=MonthLife/YEARMONTHS  '将月份数转成年份数。
IfLifeTime<>Int(MonthLife/YEARMONTHS)Then
   LifeTime=Int(LifeTime+1) '四舍五入至最接近的年份。
EndIf
DepYear=CInt(InputBox("Forwhichyeardoyouwantdepreciation?"))
DoWhileDepYear<1OrDepYear>LifeTime
   MsgBox"Youmustenteratleast1butnotmorethan"&LifeTime
```

```
    DepYear=CInt(InputBox("Forwhatyeardoyouwantdepreciation?"))
Loop
PDepr=SYD(InitCost,SalvageVal,LifeTime,DepYear)
MsgBox"ThedepreciationforyearDepYear&isFormat(PDepr,Fmt)&".
"
```

Tab 函数

与 Print#语句或 Print 方法一起使用，对输出进行定位。

语法

Tab[(*n*)]

可选的 n 参数是在显示或打印列表中的下一个表达式之前移动的列数。若省略此参数，则 Tab 将插入点移动到下一个打印区的起点。这就使 Tab 可用来替换国别中的逗号，此处，逗号是作为十进制分隔符使用的。

说明

如果当前行上的打印位置大于 n，则 Tab 将打印位置移动到下一个输出行的第 n 列上。如果 n 小于 1，则 Tab 将打印位置移动到列 1。如果 n 大于输出行的宽度，则 Tab 函数使用以下公式计算下一个打印位置：

nModwidth

例如，如果 width 是 80，并指定 Tab(90)，则下一个打印将从列 10 开始（90/80 的余数）。如果 n 小于当前打印位置，则从下一行

中计算出来的打印位置开始打印。如果计算后的打印位置大于当前打印位置，则从同一行中计算出来的打印位置开始打印。

输出行最左端的打印位置总是 1。在使用 Print#语句将数据写入文件时，最右端的打印位置是输出文件的当前宽度，这一宽度可用 Width#语句设置。

注意要确保表格列的宽度足以容纳较宽的字符串。

当 Print 方法与 Tab 函数一起使用时，打印的外观将会被分割为均匀、定宽的列。各列的宽度等于选用字体内以磅数为单位的所有字符的平均宽度。但是，在已打印字符的个数与那些字符所占据的定宽列的数目之间不存在任何关系。例如，大写字母 W 占据超过一个定宽的列，而小写字母 i 则占据少于一个定宽的列。

请参阅

Print#语句，Spc 函数，Width#语句，Print 方法，Mod 操作符，Space 函数

示例

本示例使用 Tab 函数确定在文件或立即窗口的输出位置。

```
'Tab 函数可以和 Print#语句一起使用。
Open"TESTFILE"ForOutputAs#1      '打开输出文件。
'第二个字输出到第 20 列。
Print#1,"Hello":Tab:"World."
'如果省略参数，游标移至下一个输出区。
Print#1,"Hello":Tab:"World"
```

```
Close#1     '关闭文件。
```

Tab 函数也可与 Print 方法合用。下列语句显示从第 10 列开始的文本。

```
Debug.PrintTab(10):"10columnsfromstart."
```

TabIndex 属性

返回或设置父窗体中大部分对象的 tab 键次序

应用于

ADOData 控件，TreeView 控件，ImageCombo 控件，ListView 控件，ProgressBar 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，CoolBar 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，MSChart 控件，MSHFlexGrid 控件，SSTab 控件，RichTextBox 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Frame 控件，HScrpllBar，VScrollBar 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 包容器控件

语法

object.**TabIndex**[=*index*]

TabIndex 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
index	0 到(n-1)的整数，这里 n 是窗体中有 TabIndex 属性的控件的个数。给 TabIndex 赋一个小于 0 的值会产生错误

说明

缺省情况下，在窗体上画控件时 VisualBasic 会分配一个 tab 键顺序，但 Menu、Timer、Data、Image、Line 和 Shape 控件除外，这些控件不包括在 tab 键顺序中。运行时，不可见或无效的控件以及不能接收焦点的控件（Frame 和 Label 控件）仍保持在 tab 键顺序中，但在切换时要跳过这些控件。

每个新控件都放在 tab 键顺序的最后。如果改变控件的 TabIndex 属性值来调整缺省 tab 键顺序，VisualBasic 会自动对其它控件的 TabIndex 属性重新编号，以反映出插入和删除操作。可以在设计时用属性窗口或在运行时用代码来作改变。

Zorder 方法不会影响 TabIndex 属性。

注意控件的 tab 键顺序不会影响与其相关的访问键。对于 Frame 或 Label 控件，如果按下访问键，则焦点移到 tab 键顺序中能够接收焦点的下一个控件上。

当加载存为 ASCII 文本的窗体时，对于具有 TabIndex 属性但在窗体描述中没有列出的控件会自动地分配一 TabIndex 值。以后加载的控件，如果现有的 TabIndex 值与先前分配的值发生冲突，将给

该控件分配新值。

删除一个或多个控件时，可以用 Undo 命令恢复控件以及除 TabIndex 之外所有的属性，TabIndex 是不能恢复的。用 Undo 命令时 TabIndex 被重放在 tab 键顺序的结尾。

请参阅

ZOrder 方法，TabStop 属性

示例

这个例子通过改变命令按钮数组的 TabIndex 属性使一组按钮的标签顺序反向。要尝试这个例子，请将代码粘贴到包含四个 CommandButton 控件的窗体的声明部分。将每个按钮的 Name 属性设置为 CommandX 来创建控件数组，然后按 F5 键并单击窗体使按钮的标签顺序反向。

```
PrivateSubForm_Click()
   DimI,X   '声明变量。
   '通过设置 X 的起始值使标签顺序反向。
   IfCommandX(0).TabIndex=0ThenX=4ElseX=1
      ForI=0To3
         CommandX(I).Caption=X    '设置标题。
         CommandX(I).TabIndex=X-1'设置标签的顺序。
         IfCommandX(0).TabIndex=3Then
            X=X-1    'X 减一。
         Else
            X=X+1    'X 增一。
```

```
            EndIf
        NextI
EndSub
```

TabStop 属性

返回或设置一个值，该值用来指示是否能够使用 TAB 键来将焦点从一个对象移动到另一个对象。

应用于

ADOData 控件，TreeView 控件，ImageCombo 控件，ListViewVontrol，Slider 控件，TabStrip 控件，DateTimePicker 控件，Animation 控件，MonthView 控件，UpDown 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MaskedEdit 控件，MSChart 对象，MSHFLexGrid 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，HScrollBar，VScrollBar 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 包容器控件

语法

object.**TabStop**[=*boolean*]

TabStop 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定该对象是否能够被 tab 停止的布尔表达式，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
True	（缺省值）表示指定对象能够被 tab 停止
False	表示当用户按下 tab 键时，将跨越该对象，虽然该对象仍然在实际的 tab 键顺序中保持其位置，按照 TabIndex 属性的决定

说明

该属性能够在窗体的 tab 键次序上加入或删除一个控件。例如，如果你正在使用 PictureBox 控件画一个图形，那么将其 TabStop 属性设置为 False，则就不能使用 tab 键使焦点移动到 PictureBox 上。

请参阅

TabIndex 属性

Tag 属性

返回或设置一个表达式用来存储程序中需要的额外数据。与其它属性不同，Tag 属性值不被 VisualBasic 使用；可以用该属性来

标识对象。

应用于

ADOData 控件，TreeView 控件，ImageCombo 控件，DataRepeater 控件，DataList 控件，DBCombo 控件，DBList 控件，MSChart 对象，RemoteData 控件，OLEObject 对象，PropertyPage 对象，Data 控件，CheckBox 控件，CommonDialog 控件，PropertyPage 对象，UserControl 对象，UserDocument 对象，ComboBox 控件，COmmandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件，Line 控件，ListBox 控件，MDIForm 对象，Menu 控件，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，Timer 控件，OLEContainer 控件

语法

object.**Tag**[=*expression*]

Tag 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
expression	字符串表达式用来标识对象，缺省值为零长度字符串 ("")

说明

利用该属性可以给对象赋予一个标识字符串，而不会影响其任何其它属性设置值或引起副作用。当需要检查控件或作为变量传递

给过程的 MDIForm 对象的标识时，Tag 属性是有用的。

提示创建一个新的窗口实例时，给 Tag 属性赋予唯一值。

请参阅

Name 属性

示例

这个例子为每个被拖动的控件显示一个单独的图标。要尝试这个例子，请将代码粘贴到包含三个 PictureBox 控件的窗体的声明部分。将 Picture1 和 Picture2 的 DragMode 属性设置为一，然后按 F5 键。使用鼠标在 Picture3 上面拖曳 Picture1 和 Picture2。

```
PrivateSubForm_Load()
Picture1.Tag="ICONS\ARROWS\POINT03.ICO"
Picture2.Tag="ICONS\ARROWS\POINT04.ICO"
EndSub

PrivateSubPicture3_DragOver(SourceAsControl,XAsSingle,YAsSingle,
StateAsInteger)
    IfState=vbEnterThen
        '根据每个图片框的 Name 属性选择。
        SelectCaseSource.Name
        Case"Picture1"
            '加载 Picture1 的图标。
            Source.DragIcon=LoadPicture(Picture1.Tag)
Case"Picture2"
```

```
            '加载 Picture2 的图标。
            Source.DragIcon=LoadPicture(Picture2.Tag)
        EndSelect
    ElseIfState=vbLeaveThen
        '当 Source 不在 Picture3 之上时，卸载图标。
        Source.DragIcon=LoadPicture()
    EndIf
EndSub
```

TagPrefix 属性

设置或返回字符串，用于在 HTML 模板中为替换标记名称加前缀。第一个字符必须是字母。

语法

object.TagPrefix[=*string*]

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
string	标识用于识别 HTML 模板中的替换标记的 TagPrefix 的顺序

应用于

WebItem 对象

说明

可以在设计时或运行时设置 TagPrefix。运行时 WebClass 将在模板

文件中查找以指定的前缀开头的任意标记。当找到标记时，它将激发 WebClass 对象上的 ProcessTag 事件。

请参阅

第三章的“执行 WebClass 中的文本置换”，第五部分的“开发 IIS 应用程序”和《MicrosoftVisualBasic6.0 部件工具指南》的“构造 Internet 应用程序”。

Tan 函数

返回一个 Double 的值，指定一个角的正切值。

语法

Tan(*number*)

必要的 number 参数 Double 或任何有效的数值表达式，表示一个以弧度为单位的角度。

说明

Tan 取一角度为参数值，并返回直角的两条邻边的比值。该比值是角的对边长度除以角的邻边长度的商。

为了将角度转换为弧度，请将角度乘以/∏/180。为了将弧度转换为角度，请将弧度乘以 180/∏。

请参阅

Atn 函数，Cos 函数，Sin 函数，DerivedMath 函数

示例

本示例使用 Tan 函数来求出一个角的正切（tan()）。

```
DimMyAngle,MyCotangent
MyAngle=1.3     '定义角度（以“弧度”为单位）。
MyCotangent=1/Tan(MyAngle) '利用正切来计算余切（cot()）。
```

Target 属性

返回传递给 AsyncRead.的 Target 参数。

应用于

AsyncProperty 对象

语法

object.**Target**

Target 属性语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象

说明

Target 属性返回一个字符串，该字符串和传递给 AsyncRead 方法的第一个参数具有相同的值，通常是一个 URL。

TargetObject 属性

返回绑定对象、或绑定控件所决定的其它对象。在运行时为只读。

应用于

StdDataValue 对象

语法

object.**TargetObject**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

说明

TargetObject 属性提供了在 Format 和 Unformat 事件中对绑定控件的属性、方法和事件的访问。如果绑定的控件传递了一个对象而不是控件本身，您可以使用它的属性、方法和事件去操作数据，或者向上定位到控件对象模式层次中的其它对象。

TaskVisible 属性

返回或设置一个值，用来确定应用程序是否出现在窗口任务列表中。

应用于

App 对象

语法

object.**TaskVisible**[=*boolean*]

TaskVisible 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式用来确定应用程序是否出现在任务列表中，对布尔的描述如设置值中所示

设置值

boolean 的设置值如下：

设置	描述
True	（缺省）应用程序出现在窗口任务列表中
False	应用程序不出现在窗口任务列表中

说明

如果应用程序不显示接口，那么它的 TaskVisible 属性只能设置为 False，例如不包含或不显示 Form 对象的 ActiveX 部件。如果应用程序显示接口，那么 TaskVisible 属性被自动设置为 True。

请参阅

ShowInTaskbar 属性

Template 属性

返回或设置导出一个报表时使用的 HTML 模板。

应用于

ExportFormat 对象

语法

object.**Template**[=*string*]

Template 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
string	可选的。要用到的标头和/或注脚代码

说明

如果调用 ExportReport 方法时未指定模板，一个缺省的适合于报表类型的模板将被使用。提供的缺省模板有以下四种：

HTML 模板

UTF-8 编码的 HTML

文本

Unicode 文本

通过下面的代码，您可以打印用作创建自己模板启动点的缺省模板：

```
DimiAsInteger
Fori=1TorptNwind.ExportFormats.Count
    Debug.Print"Template"&i
    Debug.PrintrptNwind.ExportFormats(i).Template
    Debug.Print
Nexti
```

可以使用 HTML 标记（如<HTML>、<HEAD>、<TITLE>、<BODY>和它们的必要的关闭标记）创建自己的模板。为了指定报表主体的位置，使用 VisualBasic 常数 rptTagBody。例如，下面的代码片段

把报表主体放在<BODY>标记之间：

```
DimstrTemplateBodyAsString
strTemplateBody=_
"<BODY>"&vbCrLf&_
rptTagBody&vbCrLf&_
"</BODY>"
```

类似地，可以使用常数 rptTagTitle 把标题（由 Title 属性决定）放在模板的任何位置。下面的代码片段把标题恰恰放在主体上面：

```
DimstrTemplateBodyAndTitleAsString
strTemplateBodyAndTitle=_
rptTagTitle&vbCrLf&_
"<BODY>"&vbCrLf&_
rptTagBody&vbCrLf&_
"</BODY>"
```

相应地，文本导出的缺省模板仅简单地由 rptTagBody 常数组成。这样，就可以创建一个简单的文本模板，以作者的名字作为报表的开始，接着是 rptTagBody 常数。如下面的示例所示：

```
DimstrTemplateAsString
strTemplate=_
"Author:JohnSmith"&vbCrLf&_
rptTagBody
```

请参阅

ExportReport 方法，Add 方法(ExportFormats 集合)

示例

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
   DataReport1.Title="DailyReport"'本标题出现在报表中。
   DimstrTemplateAsString
   '创建模板
   strTemplate=_
   "<HTML>"&vbCrLf&_
   "<HEAD>"&vbCrLf&_
   "<TITLE>"&"MyCompany:"&rptTagTitle&_
   "</TITLE>"&vbCrLf&_
   "<BODY>"&vbCrLf&_
   rptTagBody&vbCrLf&_
   "<BODY>"&vbCrLf&_
   "</HTML>"

   '使用该模板添加一个新 ExportFormat 对象。
   DataReport1.ExportFormats.Add_
   Key:="DailyReport",_
   FormatType:=rptFmtHTML,_
   FileFormatString:="DailyReport(*.htm)",_
```

```
    FileFilter:="*.HTM",_
    Template:=strTemplate

    '使用新的 ExportFormat 对象导出报表。
    DataReport1.ExportReport_
    FormatIndexOrKey:="DailyReport",_
    FileName:="C:\Temp\DailyRpt",_
    Overwrite:=True,_
    ShowDialog:=False,_
    Range:=rptRangeFromTo,_
    Pagefrom:=1,_
    Pageto:=10
EndSub
```

TemplatePath 属性

该属性返回 VisualBasic 存储模板文件处的完整路径名。

应用于

VBE 对象

语法

object.**TemplatePath**

object 所在处代表对象表达式，其值是“应用于”列表中的一个对象。

Terminate 事件

通过设置所涉及对象的所有变量为 Nothing，Form、MDIForm、User 控件、PropertyPageWebclass、DHTMLPageDesigner 或类的实例的所有引用都被从内存删除，或当对象的最后一个引用失去范围时发生。

应用于

DataReport 对象，WebClass 对象，DHTMLPageDesigner 对象，PropertyPage 对象，UserControl 对象，UserDocument 对象，Class，Form 对象，Forms 集合，MDIForm 对象

语法

Private Sub *object*_**Terminate**()

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

除类之外所有的对象，Terminate 事件在 Unload 事件之后发生。如果窗体或类的实例从内存删除，因为应用程序是非正常结束，所以不会触发 Terminate 事件。例如，应用程序在从内存中删除所有存在的类或窗体的实例前，调用 End 语句，对该类或窗体，Terminate 事件不会触发。

请参阅

QueryUnload 事件，Unload 事件，Initialize 事件

Text 属性

ComboBox 控件（Style 属性设置为 0[下拉组合框]或为 1[简单组合框]）和 TextBox 控件～返回或设置编辑域中的文本。

ComboBox 控件（Style 属性设置为 2[下拉列表]）和 ListBox 控件～返回列表框中选择的项目；返回值总与表达式 List(ListIndex)的返回值相同。在设计时为只读；在运行时为只读。

应用于

ComboBox 控件，ListBox 控件，TextBox 控件

语法

object.**Text**[=*string*]

Text 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	字符串表达式，指定文本

说明

只在设计时，Text 属性的缺省值为：

ComboBox 和 Textbox 控件～该控件的 Name 属性。

ListBox 控件～零长度字符串("")。

对于 Style 属性设置为 0（下拉组合框）或为 1（简单组合框）的 ComboBox 或者对于 Textbox，本属性对读取控件编辑域内的字符串很有用。对于 Style 属性设置为 2（下拉列表）的 ComboBox 或

ListBox 控件，可以用 Text 属性来确定当前选择的项目。

Textbox 控件的 Text 设置值最多可以有 2048 个字符，但是如果 MultiLine 属性设置为 True，此时最大限制大约是 32K。

请参阅

Value 属性，SelLength，SelStart，SelText 属性，SelLength，SelStart，SelText 属性(ActiveX 控件)，MaskedEdit 控件，ClipMode 属性，FormattedText 属性，SelText 属性(MaskedEdit 控件)

示例

这个例子用来说明 Text 属性。要尝试这个例子，请将代码粘贴到包含三个 TextBox 控件和一个 CommandButton 控件的窗体的声明部分，然后按 F5 键并在 Text1 中输入文本。

```
PrivateSubText1_Change()
   Text2.Text=LCase(Text1.Text)'用小写的格式显示文本。
   Text3.Text=UCase(Text1.Text)'用大写的格式显示文本。
EndSub

PrivateSubCommand1_Click() '删除文本。
   Text1.Text=""
EndSub
```

TextAlign 属性

返回 TextAlignChoices 类型的枚举值，规定容器希望控件采用何种对齐方式。

应用于

AmbientProperties 对象

语法

object.**TextAlign**

TextAlign 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

设置值

TextAlign 属性可能返回的枚举值为：

设置值	描述
0-General	通用的对齐方式：文本左对齐，数字右对齐。如果容器不实现这种环境属性，这将是缺省的设置值
1-Left	左对齐
2-Center	居中对齐
3-Right	右对齐
4-FillJustify	充满

说明

容器通过这个环境属性告诉所含的控件如何对齐；这是一个暗

示，控件可以采用，也可以不采用。

TextBox 控件

TextBox 控件有时也称作编辑字段或者编辑控件，显示设计时输入的用户输入的、或运行时在代码中赋予控件的信息。

语法

TextBox

说明

为了在 TextBox 控件中显示多行文本，要将 MultiLine 属性设置为 True。如果多行 TextBox 没有水平滚动条，那么即使 TextBox 调整了大小，文本也会自动换行。为了在 TextBox 上定制滚动条组合，需要设置 ScrollBars 属性。

如果文本框的 MultiLine 属性设置为 True 而且它的 ScrollBars 没有设置为 None(0)，则滚动条总出现在文本框上。

如果将 MultiLine 属性设置为 True，则可以在 TextBox 内用 Alignment 属性设置文本的对齐。如果 MultiLine 属性是 False，则 Alignment 属性不起作用。

在 DDE 对话中，TextBox 控件还可以起接收端链接的作用。

属性

DataMember 属性，DataFormat 属性，RightToLeft 属性，OLEDragMode 属性，OLEDropMode 属性，BackColor，ForeColor 属性，BorderStyle 属性，FontBold，FontItalic，FontStrike

thru，FontUnderline 属性，FontName 属性，FontSize 属性，Height,Width 属性，Left,Top 属性，TabIndex 属性，Tag 属性，Text 属性，Visible 属性，Alignment 属性，DragIcon 属性，DragMode 属性，HideSelection 属性，hWnd 属性，LinkItem 属性，LinkMode 属性，LinkTimeout 属性，Locked 属性，LinkTopic 属性，MaxLength 属性，MouseIcon 属性，MousePointer 属性，MultiLine 属性，PassoworkChar 属性，ScrollBars 属性，SelLength，SelStart，SelText 属性，TabStop 属性，Appearance 属性，Enabled 属性，HelpContextID 属性，Index 属性(ControlArray)，Name 属性，Parent 属性，FontProperty，Container 属性，ToolTipText 属性，WhatsThisHelpID 属性，Textbox，DataChanged 属性，DataField 属性，DataSource 属性，OLEDropMode 属性(ActiveX 控件)

方法

Refresh 方法，SetFocus 方法，Drag 方法，LinkExecute 方法，LinkPoke 方法，LinkRequest 方法，LinkSend 方法，Move 方法，ZOrder 方法，OLEDrag 方法，ShowWhatsThis 方法

事件

Change 事件,Click 事件,DragDrop 事件,DragOver 事件,GotFocus 事件，KeyDown，KeyUp 事件，KeyPress 事件，LinkClose 事件，LinkError 事件，LinkNotify 事件，LinkOpen 事件，LostFocus 事件，MouseDown，MouseUp 事件，MouseMove 事件，Validate

事件，DblClick 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件，OLECompleteDrag 事件(ActiveX 控件)，OLEDragOver 事件(ActiveX 控件)

请参阅

Alignment 属性，MultiLine 属性，ScrollBars 属性

TextBox 控件（数据报表设计器）

数据报表设计器的 TextBox 控件是一个数据绑定控件，它只能在运行时显示数据库的文本。

语法

RptTextBox

数据报表设计器的 TextBox 控件不能显示设计时用户输入的信息，并且，它不能在运行时用于数据输入。

属性

CanGrow 属性，DataFormat 属性，BackColor，ForeColor 属性，BackStyle 属性，BorderColor 属性，Height，Width 属性，Left，Top 属性，Visible 属性，Alignment 属性，Name 属性，Font 属性，DataField 属性

TextHeight 方法

用以返回按 Form、PictureBox 或 Printer 的当前字体将被打印的文本字符串的高度。不支持命名参数.

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象，Printers 集合 Form 对象，Forms 集合，PictureBox 控件

语法

object.**TextHeight**(*string*)

TextHeight 方法的语法包含下列部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 object，则带有焦点的 Form 对象缺省为 object
string	必需的。一个字符串表达式，它用以计算确定其高度的字符串。必须用括号包括该字符串表达式

说明

字符串高度是以对 object 有效的 ScaleMode 属性设置或通过 Scale 方法的坐标系统来表示的。使用 TextHeight 可以确定文本显示需要的垂直空间高度。返回的高度包括文本上下的正常前导空间，因此，可以使用该高度来计算和定位 object 内的多行文本。

如果 string 含有嵌入的回车返回符，TextHeight 将返回各行的累加高度，包括每行上下的前导空间。

请参阅

Scale 方法，TextWidth 方法，FontSize 属性，ScaleMode 属性

示例

TextHeight 方法被用来在一个窗体上使正文行垂直居中。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimHalfWidth,HalfHeight,Msg '声明变量。
   AutoRedraw=-1    '打开 AutoRedraw。
   BackColor=QBColor(4)'设置背景颜色。
   ForeColor=QBColor(15)    '设置前景颜色。
   Msg="VisualBasic"    '创建信息。
   FontSize=48 '设置字体大小。
   HalfWidth=TextWidth(Msg)/2   '计算半宽。
   HalfHeight=TextHeight(Msg)/2'计算半高。
   CurrentX=ScaleWidth/2-HalfWidth '设置 X。
   CurrentY=ScaleHeight/2-HalfHeight    '设置 Y。
   PrintMsg'打印信息。
EndSub
```

TextStream 对象

加快对文件的顺序访问。

语法

TextStream.{*property*|*method*}

property 和 *method* 参数可以是和 TextStream 对象相关联的任何属性和方法。请注意，在实际应用中，TextStream 被一个变量占位符所替代，该变量占位符表示从 FileSystemObject 返回的 TextStream 对象。

说明

在下面的代码中，a 是由 FileSystemObject 的 CreateTextFile 方法返回的 TextStream 对象：

```
Setfs=CreateObject("Scripting.FileSystemObject")
Seta=fs.CreateTextFile("c:\testfile.txt",True)
a.WriteLine("Thisisatest.")
a.Close
```

WriteLine 和 Close 是 TextStream 对象的两个方法。

属性

AtEndOfLine 属性，AtEndOfStream 属性，Column 属性，Line 属性,Close 方法，Read 方法，ReadAll 方法，ReadLine 方法，Skip 方法，SkipLine 方法，Write 方法，WriteBlankLines 方法，

WriteLine 方法

请参阅

Dictionary 对象，FileSystemObject 对象

TextWidth 方法

用以返回按 Form，PictureBox 或 Printer 的当前字体被打印的文本字符串的宽度。不支持命名参数.

应用于

PropertyPage 对象，UserControl 对象，UserDocument 对象，Printer 对象,Printers 集合,Form 对象,Forms 集合,PictureBox 控件

语法

object.**TextWidth**(*string*)

TextWidth 方法的语法包含下列部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 object，则带有焦点的 Form 对象缺省为 object
string	必需的。一个字符串表达式，它用以计算确定其宽度的字符串。必须用括号包括该字符串表达式

说明

字符串宽度是以对 object 有效的 ScaleMode 属性设置或通过 Scale

方法的坐标系统来表示的。使用 TextWidth 可以确定文本显示需要的水平空间宽度。如果 string 含有嵌入的回车返回符，TextWidth 将返回最长行的宽度。

请参阅

Scale 方法，TextHeight 方法，FontSize 属性，ScaleMode 属性

示例

TextWidth 方法被用来在一个窗体上使正文行水平居中。要检验此示例，可将本例代码粘贴到一个窗体的声明部分，然后按 F5 键并单击该窗体。

```
PrivateSubForm_Click()
   DimHalfHeight,HalfWidth,Msg '声明变量。
   AutoRedraw=-1    '打开 AutoRedraw。
   BackColor=QBColor(4)'设置背景颜色。
   ForeColor=QBColor(15)    '设置前景颜色。
   Msg="VisualBasic"    '创建信息。
   FontSize=48 '设置字体大小。
   HalfWidth=TextWidth(Msg)/2'计算半宽。
   HalfHeight=TextHeight(Msg)/2'计算半高。
   CurrentX=ScaleWidth/2-HalfWidth '设置 X。
   CurrentY=ScaleHeight/2-HalfHeight    '设置 Y。
   PrintMsg'打印信息。
EndSub
```

ThreadID 属性

该属性返回执行线程的 Win32ID（用于 Win32API 调用）。

应用于

App 对象

语法

object.ThreadID

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

返回值类型

Long

Time 函数

返回表明当前系统时间的一个 Variant(Date)。

语法

Time

声明

设置系统时间，使用 Time 语句。

请参阅

Date 函数，Date 语句，Time 语句，Timer 函数

示例

本示例使用 Time 函数返回系统当前的时间。

```
DimMyTime
```

```
MyTime=Time      '返回系统当前的时间。
```

Time 语句

设置系统时间。

语法

Time=*time*

必要的 time 参数，可以是任何能够表示时刻的数值表达式、字符串表达式或它们的组合。

说明

如果 *time* 是一字符串，则 Time 会试着根据系统指定的时间，利用时间分隔符将其转换成一个时间。如果无法转换成一个有效的时间，则会导致错误发生。

请参阅

Date 函数，Date 语句，Time 函数

示例

本示例使用 Time 语句来设置系统时间。

```
DimMyTime
MyTime=#4:35:17PM# '指定一时间。
Time=MyTime      '将系统时间设置为 MyTime 的内容。
```

Timer 控件

通过引发 Timer 事件，Timer 控件可以有规律地隔一段时间执行一次代码。

语法

Timer

说明

Timer 控件用于背景进程中，它是不可见的。

对于 Timer 控件以外的其它控件的多重选择，不能设置 Timer 的 Enabled 属性。

在运行于 Windows95 或 WindowsNT 下的 VisualBasic5.0 中可以有多个活动的定时器控件，对此，实际上并没有什么限制。

属性

Interval 属性，Left，Top 属性，Tag 属性，Enabled 属性，Index 属性(控件排列)，Name 属性，Parent 属性

事件

Timer 事件

请参阅

Time 函数，Time 语句，Timer 事件，Enabled 属性

Timer 事件

在一个 Timer 控件的预定的时间间隔过去之后发生。该间隔的频

率储存于该控件的 Interval 属性中，它以千分之一秒为单位指定时间的长度。

应用于

Timer 控件

语法

PrivateSub*object*_**Timer**([*index***AsInteger**])

Timer 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
index	一个整数，用来唯一地标识一个在控件数组中的控件

说明

使用 Timer 事件时，可用此事件在每次 Timer 控件时间间隔过去之后通知 VisualBasic 应该做什么：

Interval 属性以千分之一秒为单位指定 Timer 事件之间的间隔。无论何时，只要 Timer 控件的 Enabled 属性被设置为 True 而且 Interval 属性大于 0，则 Timer 事件以 Interval 属性指定的时间间隔发生。

请参阅

Interval 属性，Enabled 属性，Enabled 属性(ActiveX 控件)

示例

这个例子演示一个数字时钟。要尝试这个例子，可以将代码粘贴到包含一个 Label 控件和一个 Timer 控件窗体的声明部分，然后按 F5。

```
PrivateSubForm_Load()
   Timer1.Interval=1000'设置计时器时间间隔。
EndSub

PrivateSubTimer1_Timer()
   Label1.Caption=Time '更新时间显示。
EndSub
```

本例在一个窗体内移动一个 PictureBox 控件。要尝试这个例子，可以将代码粘贴到一个包含 Timer 控件和一个 PictureBox 控件窗体的声明部分，然后按 F5。为了得到更好的视觉效果，可以将 PictureBox 的 Picture 属性赋值为一个位图。

```
DimDeltaX,DeltaYAsInteger '声明变量。
PrivateSubTimer1_Timer()
   Picture1.MovePicture1.Left+DeltaX,Picture1.Top+DeltaY
   IfPicture1.Left<ScaleLeftThenDeltaX=100
   IfPicture1.Left+Picture1.Width>ScaleWidth+ScaleLeftThen
      DeltaX=-100
   EndIf
   IfPicture1.Top<ScaleTopThenDeltaY=100
   IfPicture1.Top+Picture1.Height>ScaleHeight+ScaleTopThen
      DeltaY=-100
   EndIf
EndSub
```

```
PrivateSubForm_Load()
   Timer1.Interval=1000'设置时间间隔。
   DeltaX=100   '初始化变量。
   DeltaY=100
EndSub
```

Timer 函数

返回一个 Single，代表从午夜开始到现在经过的秒数。

请参阅

Time 函数，Randomize 语句

语法

Timer

说明

MicrosoftWindows 中，Timer 函数返回一秒的小数部分。在 Macintosh 上，计时器的精度是 1 秒。

示例

本示例使用 Timer 函数来暂停应用程序。同时用 DoEvents 在暂停期间将控制让给其他进程。

```
DimPauseTime,Start,Finish,TotalTime
If(MsgBox("PressYestopausefor5seconds",4))=vbYesThen
   PauseTime=5 '设置暂停时间。
```

```
    Start=Timer '设置开始暂停的时刻。
    DoWhileTimer<Start+PauseTime
        DoEvents '将控制让给其他程序。
    Loop
    Finish=Timer '设置结束时刻。
    TotalTime=Finish-Start  '计算总时间。
    MsgBox"Pausedfor"&TotalTime&"seconds"
Else
    End
EndIf
```

TimeSerial 函数

返回一个 Variant(Date)，**包含具有具体时、分、秒的时间。**

语法

TimeSerial(*hour*,*minute*,*second*)

TimeSerial **函数语法有下列的命名参数**:

部分	**描述**
Hour	**必要**；Variant(Integer)。**其值从** 0(12:00A.M.) **到** 23(11:00P.M.)，**或一数值表达式**
Minute	**必要**；Variant(Integer)。**任何数值表达式**
second	**必要**；Variant(Integer)。**任何数值表达式**

说明

为了指定一个时刻，如 11:59:59，TimeSerial 的参数取值应在正常范围内；也就是说，钟点应介于 0-23 之间，而分钟与秒应介于 0-59 之间。但是，当一个数值表达式表示某时刻之前或其后的时、分钟或秒数时，也可以为每个使用这个数值表达式的参数指定相对时间。以下示例中使用了表达式代替绝对时间数。TimeSerial 函数返回中午之前六小时(12-6)又十五分钟(-15)的时间，即 5:45:00A.M.

```
TimeSerial(12-6,-15,0)
```

当任何一个参数的取值超出正常范围时，它会适时进位到下一个较大的时间单位。例如，如果指定了 75（75 分钟），则这个时间被解释成一小时又十五分。如果一个参数值超出-32，768 到 32，767 的范围，就会导致错误发生。如果三个参数指定的时间会使日期超出可接受的日期范围，则亦会导致错误发生。

请参阅

DateSerial 函数，DateValue 函数，Hour 函数，Minute 函数，Now 函数，Second 函数，TimeValue 函数

示例

本示例使用 TimeSerial 函数来将已知的时分秒转换为时间。

```
DimMyTime
MyTime=TimeSerial(16,35,17)     'MyTime 的值为 4:35:17PM 之时间表达式。
```

TimeValue 函数

返回一个包含时间的 Variant(Date)。

语法

TimeValue(*time*)

必要的 time 参数，通常是一个字符串表达式，表示 0:00:00(12:00:00A.M.)到 23:59:59(11:59:59P.M.)之间的时刻。但是，time 也可以是表示在同一时间范围取值的任何其它表达式。如果 time 包含 Null，则返回 Null。

说明

可以使用 12 小时制或 24 小时制的时间格式。例如，“2:24PM”和"14:24"均是有效的 time 表达式。

如果 time 参数包含日期信息，TimeValue 将不会返回它。但是，若 time 包含无效的日期信息，则会导致错误发生。

请参阅

DateSerial 函数，DateValue 函数，Hour 函数，Minute 函数，Now 函数，Second 函数，TimeSerial 函数

示例

本示例使用 TimeValue 函数将字符串转换为时间。也可以使用日期原义直接给 Variant 或 Date 类型的变量赋值时间，例如 MyTime=#4:35:17PM#。

```
DimMyTime
MyTime=TimeValue("4:35:17PM")  '返回时间。
```

Title 属性（DataReport 对象）

返回或设置报表的标题。

应用于

DataReport 对象

语法

object.**Title**[=*string*]

Title 属性的语法包含如下部分：

部分	描述
object	必需的。一个对象表达式，其值为“应用于”列表中的一个对象
string	可选的。报表的标题

说明

Title 属性指定的标题使用在如下两个地方：

在标题设置为%i 的 TextBox 控件中。

在导出的报表中，当创建 ExportFormat 对象的一个模板（使用 Template 属性）时，标题用 rptTagTitle 常数表示。

示例

本例创建一个模板，并使用新模板把 ExportFormat 对象添加到 ExportFormats 集合，然后使用 ExportFormat 对象导出报表。

```
PrivateSubExportDailyReport()
    DataReport1.Title="DailyReport"'本标题出现在报表中。
```

```
DimstrTemplateAsString
'创建模板
strTemplate=_
"<HTML>"&vbCrLf&_
"<HEAD>"&vbCrLf&_
"<TITLE>"&"MyCompany:"&rptTagTitle&_
"</TITLE>"&vbCrLf&_
"<BODY>"&vbCrLf&_
rptTagBody&vbCrLf&_
"<BODY>"&vbCrLf&_
"</HTML>"

'使用该模板添加一个新 ExportFormat 对象。
DataReport1.ExportFormats.Add_
Key:="DailyReport",_
FormatType:=rptFmtHTML,_
FileFormatString:="DailyReport(*.htm)",_
FileFilter:="*.HTM",_
Template:=strTemplate

'使用新的 ExportFormat 对象导出报表。
DataReport1.ExportReport_
FormatIndexOrKey:="DailyReport",_
```

```
    FileName:="C:\Temp\DailyRpt",_
    Overwrite:=True,_
    ShowDialog:=False,_
    Range:=rptRangeFromTo,_
    Pagefrom:=1,_
    Pageto:=10
EndSub
```

Title 属性

返回或设置应用程序的标题，该标题要显示在 MicrosoftWindows 的任务列表中。如果在运行时发生改变，那么发生的改变不会与应用程序一起被保存。

应用于

App 对象

语法

object.**Title**[=*value*]

Title 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定应用程序的标题的字符串表达式。value 的最大长度为 40 个字符。在 DBCS（双字节字符集）系统中，这意味着最大长度为 40 个字节

说明

该属性在设计时在“工程”菜单上的“工程属性”命令的对话框中是可用的。

请参阅

Caption 属性，Caption 属性(ActiveX 控件)

ToolboxBitmap 属性

返回或设置位图，作为工具箱中的用图片表示。工具箱中位图的空间大小为 16x15 像素；在需要时，将把该属性指定的位图按这个尺寸缩放。在创建控件时，ToolboxBitmap 属性可读可写，在控件运行时，该属性不可用。

应用于

User 控件对象

说明

重点不要把图标赋值给 ToolboxBitmap 属性。图标不能很好地缩放到工具箱位图尺寸。

当用户将鼠标指针放在工具箱中某个图标上方时，VisualBasic 自动将控件的类名作为工具提示文本。
提示创建位图时，请记住对许多类型的色盲来说，同一亮度级的颜色看起来没有区别。为避免这种情况，位图的颜色只采用白色、黑色和灰色阴影，或者精心选择使用的颜色。

ToolTipText 属性

返回或设置一个工具提示.

应用于

CheckBox 控件, ComboBox 控件, CommandButton 控件, DirListBox 控件, DriveListBox 控件, FileListBox 控件, Frame 控件, Image 控件, Label 控件, ListBox 控件, OptionButton 控件, PictureBox 控件, TextBox 控件, ADOData 控件, TreeView 控件 ImageCombo 控件, ListView 控件, ProgressBar 控件, Slider 控件, StatusBar 控件, Panel 对象, TabSrtip 控件, Tab 对象, Toolbar 控件, Button 对象, DateTimePicker 控件, Animation 控件, FlatScrollBar 控件, MonthView 控件, UpDown 控件, CoolBar 控件, DataRepeater 控件, DataList 控件, DBCombo 控件, DBList 控件., MaskedEdit 控件, MultimediaMCI 控件, MSChart 对象, MSFlexGrid 控件, SSTab 控件, RemoteData 控件, RichTextBox 控件, Data 控件

语法

object.**ToolTipText**[=*string*]

ToolTipText 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
+	与“应用于”列表中的一个对象相联系的字符串。当运行时光标在对象上徘徊约一秒时，该字符串将显示在该对象下面的一个小矩形框中

说明

如果仅用图象作为对象的标签，那么能够使用此属性以较少的话解释每个对象。

在设计时仅可以在控件的属性对话框中设置 ToolTipText 属性字符串。

对于 Toolbar 和 TabStrip 控件，为了显示 ToolTips 必须设置 ShowTips 属性为 True。

请参阅

ShowTips 属性(ActiveX 控件)

Top 属性

返回或设置一个 Single，它以缇为单位指示该窗口上边缘在屏幕中的位置，此属性可读/写。

应用于

Windows 对象

说明

由 Top 属性所返回的值取决于此窗口是否为可连接窗口、链接窗口，或在连接查看中。

注意改变一个链接或连接窗口的 Top 属性设置不会产生任何效果，只要该窗口保持链接或连接。

请参阅

Height 属性(VBAAdd-In 对象 Model)，Left 属性，Width 属性

TopIndex 属性

返回或设置一个值，该值指定在 ComboBox、DirListBox、DriveListBox、FileListBox 或 ListBox 控件中的哪个项被显示在顶部的位置。在设计时是不可用。

语法

object.**TopIndex**[=*value*]

TopIndex 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	被显示在顶部位置的列表项的号码。缺省值是 0，或是列表中的第一项

说明

使用此属性可以不用选择项而在控件中进行滚动。

对于 ListBox 控件，如果把 Columns 属性设置为 0，那么如果在一

个项的下面还有足够的项来填充列表的可视部分时，则该项被显示在顶部位置。

对于 ListBox 控件，如果 Columns 属性设置值大于 0，那么该项所在的列将移动到最左位置而不改变其在列中的位置。

请参阅

AddItem 方法，Clear 方法(Clipboard，ComboBox，ListBox)，RemoveItem 方法，List 属性，ListCount 属性，Columns 属性(ListBox)，MultiSelect 属性，NewIndex 属性，Selected 属性

示例

本例用屏幕字体的名字来填充 ListBox 控件，然后当单击窗体时可滚动 ListBox。要试用此例，先将代码粘贴到包含 ListBox 控件的窗体的声明部分，然后按 F5 键并单击窗体。

```
PrivateSubForm_Load()
   DimI'声明变量。
   ForI=0ToScreen.FontCount-1   '填充列表框用
        List1.AddItemScreen.Fonts(I)'屏幕字体的名字。
   NextI
EndSub

PrivateSubForm_Click()
   DimX'声明变量。
   X=List1.TopIndex'获得当前索引。
   List1.TopIndex=List1.TopIndex+5'复位最上面的项。
```

```
    IfList1.TopIndex=XThenList1.TopIndex=0
EndSub
```

TopLine 属性

返回一个 Long 型数，它指定在代码窗格顶部的行号，或设置代码窗格顶部的行，此属性可读/写。

应用于

CodePane 对象

说明

使用 TopLine 属性可返回或设置显示在代码窗格程序最上方之行。譬如，若想要在代码窗格显示第 25 列，就把 TopLine 属性设为 25。

TopLine 属性必须设置为一个正数。若 TopLine 属性的设置大于代码窗格中的实际行数，则设置是代码窗口中的最后一行。

请参阅

GetSelection 方法，SetSelection 方法，CountOfVisibleLines 属性

示例

下列示例使用 TopLine 属性返回指定的代码窗格中第一行的行号。

```
Debug.PrintApplication.VBE.CodePanes(3).TopLine
```

TotalSize 属性

以字节为单位，返回驱动器或网络共享的总空间大小。

应用于

Drive 对象

语法

object.**TotalSize**

object 总是一个 Drive 对象。

说明

下面的代码举例说明了 TotalSize 属性的用法：

```
SubShowSpaceInfo(drvpath)
Dimfs,d,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(fs.GetDriveName(fs.GetAbsolutePathName(drvpath)
))
s="Drive"&d.DriveLetter&":"
s=s&vbCrLf
s=s&"TotalSize:"&FormatNumber(d.TotalSize/1024,0)&"Kbytes"
s=s&vbCrLf
s=s&"Available:"&FormatNumber(d.AvailableSpace/1024,0)&"Kbytes"
MsgBoxs
EndSub
```

请参阅

AvailableSpace 属性，DriveLetter 属性，DriveType 属性，FileSystem 属性，FreeSpace 属性，Path 属性(FileSystemObject 对象)，RootFolder 属性，SerialNumber 属性，ShareName 属性，VolumeName 属性

Trace 方法

将特定的字符串传递给 Win32OutputDebugStringAPI。字符串可能被适当的调试工具捕获，例如，DBMON。

语法

object.Trace(traceoutputAsString)

Trace 方法的语法有这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
traceoutput	写入到 Win32OutputDebugStringAPI 的字符串

说明

跟踪字符串一般包含开发过程中的调试信息。对一个产品 WebClass 来说，跟踪字符串可能包含错误消息，以及性能和统计数据。

TrackDefault 属性

返回或设置一个值，该值用来决定 Printer 对象是总指向一个相同的打印机，还是当在操作系统的控制面板中改变了缺省打印机时

改变它所指向的打印机。在设计时是不可用的。

应用于

Printer 对象，Printers 集合

语法

object.**TrackDefault** [=*boolean*]

TrackDefault 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定打印机 object 指向的布尔表达式，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
True	（缺省值）当操作系统控制面板中缺省打印机设置值改变时，Printer 对象改变它所指向的打印机
False	Printer 对象仍然指向相同的打印机，即使操作系统控制面板中缺省打印机的设置值改变

说明

当一个打印作业正在处理时改变 TrackDefault 属性的设置值，将发送一个隐含的 EndPage 语句给 Printer 对象。

请参阅

EndDoc 方法，Height，Width 属性，ColorMode 属性，Copies 属

性，PaperBin 属性，PrintQuality 属性

True

True 关键字的值等于-1。

请参阅

False，BooleanData 类型

TrueValue 属性

设置或返回一个值，用于格式化或去除格式化一个布尔 True 值。在设计时或运行时都可读写。

应用于

StdDataFormat 对象

语法

object.**TrueValue**[=*value*]

TrueValue 属性的语法有如下的部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
value	可选的变体型。在 Format 事件中，如果数据是一个布尔 True，则返回 value；如果 TrueValue 为空，则返回整型数 0。在 Unformat 事件中，如果 TrueValue 为空并且 value 为 0，或者数据与 value 相匹配，则布尔 True 将被写到数据库中。缺省值为 True

说明

除非 Type 属性设置为 fmtBoolean，否则它将被忽略。每次取数据时都将读 TrueValue 属性。

请参阅

FalseValue 属性，Format 事件(StdDataFormat 对象)，NullValue 属性，Type 属性，Unformat 事件

TwipsPerPixelX、TwipsPerPixelY 属性

返回水平(TwipsPerPixelX)或垂直(TwipsPerPixelY)度量的对象的每一像素中的缇数。

应用于

Screen 对象，Printer 对象，Printers 集合

语法

object.**TwipsPerPixelX**

object.**TwipsPerPixelY**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

WindowsAPI 例程一般需要以像素为度量单位。使用这些属性能够快速转换度量单位而不用改变对象的 ScaleMode 属性设置值。

请参阅

ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性

类型转换函数

每个函数都可以强制将一个表达式转换成某种特定数据类型。

语法

CBool(*expression*)

CByte(*expression*)

CCur(*expression*)

CDate(*expression*)

CDbl(*expression*)

CDec(*expression*)

CInt(*expression*)

CLng(*expression*)

CSng(*expression*)

CStr(*expression*)

CVar(*expression*)

CStr(*expression*)

必要的 expression **参数可以是任何字符串表达式或数值表达式。**

返回类型

函数名称决定返回类型，如下所示：

函数	**返回类型**	Expression **参数范围**
Cbool	Boolean	**任何有效的字符串或数值表达式**
Cbyte	Byte	0 **至** 255
Ccur	Currency	-922,337,203,685,477.5808 **至** 922,337,203,685,477.5807。
CDate	Date	**任何有效的日期表达式**
CDbl	Double	**负数从**-1.79769313486232E308 **至**-4.94065645841247E-324；**正数从** 4.94065645841247E-324 **至** 1.79769313486232E308
CDec	Decimal	**零变比数值，即无小数位数值，为**+/-79,228,162,514,264,337,593,543,950,335。 **对于** 28 **位小数的数值，范围则为**+/-7.9228162514264337593543950335；**最小的可能非零值是** 0.0000000000000000000000000001
CInt	Integer	-32,768 **至** 32,767，**小数部分四舍五入**

CLng	Long	-2,147,483,648 至 2,147,483,647，小数部分四舍五入
CSng	Single	负数为-3.402823E38 至-1.401298E-45；正数为 1.401298E-45 至 3.402823E3
CStr	String	依据 expression 参数返回 Cstr
CVar	Variant	若为数值，则范围与 Double 相同；若不为数值，则范围与 String 相同

说明

如果传递给函数的 expression 超过转换目标数据类型的范围，将发生错误。

通常，在编码时可以使用数据类型转换函数，来体现某些操作的结果应该表示为特定的数据类型，而不是缺省的数据类型。例如，当单精度、双精度或整数运算发生的情况下，使用 CCur 来强制执行货币运算。

应该使用数据类型转换函数来代替 Val，以使国际版的数据转换可以从一种数据类型转换为另一种。例如，当使用 Ccur 时，不同的小数点分隔符、千分位分隔符和各种货币选项，依据系统的国别设置都会被妥善识别。

当小数部分恰好为 0.5 时，CInt 和 CLng 函数会将它转换为最接近的偶数值。例如，0.5 转换为 0、1.5 转换为 2。Cint 和 CLng 函数不同于 Fix 和 Int 函数，Fix 和 Int 函数会将小数部分截断而不是四舍五入。并且 Fix 和 Int 函数总是返回与传入的数据类型相同

的值。

使用 IsDate 函数，可判断 date 是否可以被转换为日期或时间。Cdate 可用来识别日期文字和时间文字，以及落入可接受的日期范围内的数值。当转换一个数字成为日期时，是将整数部分转换为日期，小数部分转换为从午夜起算的时间。

CDate 依据系统上的国别设置来决定日期的格式。如果提供的格式为不可识别的日期设置，则不能正确判断年、月、日的顺序。另外，长日期格式，若包含有星期的字符串，也不能被识别。

CVDate 函数也提供对早期 VisualBasic 版本的兼容性。CVDate 函数的语法与 CDate 函数是完全相同的，不过，CVDate 是返回一个 Variant，它的子类型是 Date，而不是实际的 Date 类型。因为现在已有真正的 Date 类型，所以 CVDate 也不再需要了。转换一个表达式成为 Date，再赋值给一个 Variant，也可以达到同样的效果。也可以使用这种技巧将其他真正的数据类型转换为对等的 Variant 子类型。

注意 CDec 函数不能返回独立的数据类型，而总是返回一个 Variant，它的值已经被转换为 Decimal 子类型。

CStr 返回值:

如果 expression 是	CStr 返回
Boolean	含有 True 或 False 的字符串
Date	含有系统中短日期格式日期的字符串
Null	一个运行时错误
Empty	一个零长度字符串("")
Error	包含单词 Error 以及错误号的字符串
其他数值	含有数值的字符串

示例

本示例使用 CBool 函数来将一表达式转成 Boolean 值。如果表达式的结果为非零的值，CBool 返回 True；否则返回 False。

```
DimA,B,Check
A=5:B=5    '设置变量初值。
Check=CBool(A=B)    'Check 的值为 True。

A=0    '定义变量。
Check=CBool(A) 'Check 的值为 False。
```

本示例使用 CByte 函数将一表达式转成 Byte。

```
DimMyDouble,MyByte
MyDouble=125.5678  'MyDouble 为 Double（双精度）。
MyByte=CByte(MyDouble) 'MyByte 值为 126。
```

本示例使用 CDate 函数将字符串转换成 Date。一般说来，字符串格式的日期与时间硬编码（如示例中所示）并不好。较好的做法是使用日期原义表达式和时间的原义表达式（如#2/12/1969#，#4:45:23PM#）。

```
DimMyDate,MyShortDate,MyTime,MyShortTime
MyDate="February12,1969"    '定义日期。
MyShortDate=CDate(MyDate)   '转换为 Date 数据类型。

MyTime="4:35:47PM" '定义时间。
MyShortTime=CDate(MyTime)   '转换为 Date 数据类型。
```

本示例使用 CCur 函数将表达式转成 Currency。

```
DimMyDouble,MyCurr
MyDouble=543.214588      'MyDouble 为 Double 类型。
MyCurr=CCur(MyDouble*2)      '将 MyDouble*2 的结果
   ' (1086.429176) 转换为 Currency (1086.4292)。
```

本示例使用 CDbl 函数将表达式转换为 Double 类型。

```
DimMyCurr,MyDouble
MyCurr=CCur(234.456784)      'MyCurr 为 Currency 类型。
MyDouble=CDbl(MyCurr*8.2*0.01) '将结果转换为 Double 类型。
```

本示例使用 CInt 函数将一数值转换为 Integer。

```
DimMyDouble,MyInt
MyDouble=2345.5678 'MyDouble 为 Double 类型。
MyInt=CInt(MyDouble)    'MyInt 的值为 2346。
本示例使用 CLng 函数将一数值转换为 Long。
DimMyVal1,MyVal2,MyLong1,MyLong2
MyVal1=25427.45:MyVal2=25427.55      'MyVal1、MyVal2 均为 Double 类
型。
MyLong1=CLng(MyVal1)    'MyLong1 的值为 25427。
MyLong2=CLng(MyVal2)    'MyLong2 的值为 25428。

本示例使用 CSng 函数将一数值转换为 Single。
DimMyDouble1,MyDouble2,MySingle1,MySingle2
'MyDouble1、MyDouble2 均为 Double 类型。
MyDouble1=75.3421115:MyDouble2=75.3421555
MySingle1=CSng(MyDouble1)  'MySingle1 的值为 75.34211。
MySingle2=CSng(MyDouble2)  'MySingle2 的值为 75.34216。

本示例使用 CStr 函数将一数值转换为 String。
DimMyDouble,MyString
MyDouble=437.324    'MyDouble 为 Double 类型。
MyString=CStr(MyDouble)      'MyString 的内容为“437.324”。
```

本示例使用 CVar 函数将表达式转换为 Variant。

```
DimMyInt,MyVar
MyInt=4534 'MyInt 为 Integer 类型。
MyVar=CVar(MyInt&"000")      'MyVar 的内容为字符串“4534000”。
```

该示例使用 CDec 函数将数字值转换为 Decimal。

```
DimMyDecimal,MyCurr
MyCurr=10000000.0587    'MyCurr 是货币。
MyDecimal=CDec(MyCurr)      'MyDecimal 是二进制数。
```

Type 属性（Format 对象）

设置或返回 StdDataFormat 对象所应用的格式化类型。基于该设置值，对象的其它属性被用来格式化该值。在设计时或运行时都可读写。

应用于

StdDataFormat 对象

语法

object.**Type**=*formattype*

Type 属性的语法有如下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
formattype	必需的枚举整型。指定所希望的格式化类型，如设置值中所述

设置值

formattype 设置如下：

常数	设置值	描述
fmtGeneral	0	没有格式化被应用于该值
fmtCustom	1	Format 属性所包含的格式化字符串被用于格式化和去除格式化该值
fmtPicture	2	该值被作为图象处理。从数据库中读取的二进制对象被转换为图形对象，并且以二进制对象写回到数据库中
fmtObject	3	该值视为一个对象。StdDataFormat 对象希望从数据库中检索一个 CLSID。CLSID 用于该对象的实例化，它从 StdDataFormat 对象返回。当值返回数据库时，CLSID 从格式化的对象中读出
fmtCheckbox	4	该设置用于绑定一个复选框到一个数据库字段。该数据库值被处理为该复选框的 Value 属性。返回值本文的说明部分的表格

		中指定。已格式化的值仅对 Value 属性为 OLE_TRISTATE 类型的复选框控件有意义。对于普通的布尔类型，则使用 fmtBoolean
fmtBoolean	5	该值作为 Boolean 值处理。当选择了此类型时，TrueValue 和 FalseValue 属性，有时还有 NullValue 属性（要依据数据库类型）将被用于决定如何格式化值。如果以上的属性均未设置，缺省行为是从数据库中返回逻辑值

说明

如上所述，下表列出了与 fmtCheckbox 设置值相关的返回值。

数据库值	格式化值
LogicalFalse	0
LogicalTrue	1
Null	2

为了在运行时重新设置 Type 属性，必需首先解除绑定，然后设置该属性，最后重新绑定。

请参阅

FalseValue 属性, Format 属性(StdDataFormat 对象), NullValue 属性,
TrueValue 属性，CheckBox 控件

Type 属性

该属性返回当前选择的成员或工程的类型。

应用于

Member 对象，Reference 对象，BVComponent 对象，VBProject 对象，
Window 对象

语法

object.**Type**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

返回值

Member 对象(vbext_MemberType)的 Type 属性的设置值是：

常数	值	描述
vbext_mt_Method	1	成员是方法
vbext_mt_Property	2	成员是属性
vbext_mt_Variable	3	成员是变量
vbext_mt_Event	4	成员是事件
vbext_mt_Enum	5	成员是计数值
vbext_mt_Const	6	成员是常数
vbext_mt_EventSink	7	成员是事件吸收

Project 对象(vbext_ProjectType)的 Type 属性的设置值是：

常数	值	描述
vbext_pt_StandardExe	1	工程类型是标准可执行文件（StandardExe）
vbext_pt_ActiveXExe	2	工程类型是 ActiveX 可执行文件（ActiveXExe）
vbext_pt_ActiveXDll	3	工程类型是 ActiveX 动态链接库
vbext_pt_ActiveXControl	4	工程类型是 ActiveX 控件

Type 属性（VBA 外接程序对象模型）

返回一个数字或字符串值，其内容为对象的类型，此属性为只读。

应用于

Reference 对象，VBComponent 对象，Window 对象

返回值

Window 对象的 Type 属性设置如下表描述：

常数	值	描述
vbext_wt_CodeWindow	0	“代码窗口”window
vbext_wt_Designer	1	设计器
vbext_wt_Browser	2	“对象浏览器”
vbext_wt_Immediate	5	“立即”窗口
vbext_wt_ProjectWindow	6	工程窗口
vbext_wt_PropertyWindow	7	属性窗口
vbext_wt_Find	8	“查找”对话框
vbext_wt_FindReplace	9	“查找和替换”对话框
vbext_wt_LinkedWindowFrame	11	链接窗口框架
vbext_wt_MainWindow	12	主窗口
vbext_wt_Watch	3	监视窗格
vbext_wt_Locals	4	本地

VBComponent 对象的 Type 属性设置如下表描述：

常数	描述
Vbext_ct_ClassModule	类模块
Vbext_ct_MSForm	Microsoft 窗体
Vbext_ct_StdModule	标准模块

Reference 对象的 Type 属性设置如下表描述：

常数	描述
vbext_rk_TypeLib	类型库
vbext_rk_Project	工程

请参阅

IndexedValue 属性(VBA 外接程序对象模型)，Name 属性(VBA 外接程序对象模型)，Value 属性

示例

下列示例使用 Type 属性返回一个值，指示特定工程中 VBComponents 集合的指定成员的类型。返回的值是一事先定义好的常量，代表部件的类型。

```
Debug.PrintApplication.VBE.VBProjects(1).VBComponents(1).Type
```

Type 属性(FileSystemObject 对象)

返回关于某个文件或文件夹类型的信息。例如，对于以.TXT 结尾的文件来说，返回"TextDocument"。

应用于

File 对象

语法

object.**Type**

object 总是一个 File 或 Folder 对象。

说明

下面的代码举例说明了返回某个文件夹类型的 Type 属性的用法。在这个示例中，试图将 RecycleBin 的路径或其他唯一的文件夹提供给过程。

```
SubShowFileSize(filespec)
Dimfs,f,s
Setfs=CreateObject("Scripting.FileSystemObject")
Setf=fs.GetFolder(filespec)
s=UCase(f.Name)&"isa"&f.Type
MsgBoxs,0,"FileSizeInfo"
EndSub
```

请参阅

Attributes 属性，DateCreated 属性，DateLastAccessed 属性，DateLastModified 属性，Drive 属性，Files 属性，IsRootFolder 属性，Name 属性(FileSystemObject 对象)，ParentFolder 属性，Path 属性(FileSystemObject 对象)，ShortName 属性，ShortPath 属性，Size 属性(FileSystemObject 对象)，SubFolders 属性

Type 属性（图片）

返回 Picture 对象的图形格式。在设计时是不可用的；在运行时是只读的。

应用于

Picture 对象，AxisScale 对象

语法

object.**Type**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

返回值

Type 属性的返回值为：

常数	值	描述
vbPicTypeNone	0	图片为空。
vbPicTypeBitmap	1	位图（.bmpBMP 文件）
vbPicTypeMetafile	2	元文件（.wmfWMF 文件）
vbPicTypeIcon	3	图标(.icoICOfiles)
VbPicTypeEMetafile	4	增强的元文件(.emfEMF 文件）

说明

这些常数在中的对象浏览器中的 VisualBasic(VB)对象库中被列出。

请参阅

Height，Width 属性，PercentBasis 属性

示例

这个例子读取 PictureBox 控件中的 Picture 对象的 Type 和 Width 属性的设置。要试用此例，可以先将该代码粘贴到包含 PictureBox（其 Picture 属性被设为一个图标）的窗体的声明部分中，然后按下 F5 键并单击此窗体。

```
PrivateSubForm_Click()
    IfPicture1.Picture.Type=vbPicTypeIconThen
        Print"Thegraphicinthepictureboxisanicon."
    Else
        Print"ThePicturepropertyisn'tsettoanicon."
    EndIf
    Print"WidthofthegraphicinHiMetricsis"&Picture1.Picture.Width
    Print"Widthofpictureboxitselfintwipsis"&Picture1.Width
EndSub
```

Type 语句

在模块级别中使用，用于定义包含一个或多个元素的用户自定义的数据类型。

语法

[Private|Public]Type*varname*

elementname[([*subscripts*])]As*type*

[*elementname*[([*subscripts*])]As*type*]

...

EndType

Type 语句的语法包含下面部分：

部分	描述
Public	可选的。用于声明可在所有工程的所有模块的任何过程中使用的用户定义类型
Private	可选的。用于声明只能在包含该声明的模块中使用的用户自定义的类型
varname	必需的。用户自定义类型的名称；遵循标准的变量命名约定
Elementname	必需的。用户自定义类型的元素名称。除了可以使用的关键字，元素名称也应遵循标准变量命名约定
subscripts	可选的。数组元素的维数。当定义大小可变的数组时，只需圆括号。Subscripts 参数使用如下语法： [lowerTo]upper[,[lowerTo]upper]... 如果不显式指定 lower，则数组的下界由 OptionBase 语句控制。如果没有 OptionBase 语句则下界为 0
type	必需的。元素的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（对变长的字符串）、String*length（对定长的字符串）、Object、Variant、其它的用户自定义的类型或对象类型

说明

Type 语句只能在模块级使用。使用 Type 语句声明了一个用户自

定义类型后，就可以在该声明范围内的任何位置声明该类型的变量。可以使用 Dim、Private、Public、ReDim 或 Static 来声明用户自定义类型的变量。

在标准模块中，用户自定义类型按缺省设置是公用的。可以使用 Private 关键字来改变其可见性。而在类模块中，用户自定义类型只能是私有的，且使用 Public 关键字也不能改变其可见性。

在 Type...EndType 块中不允许使用行号和行标签。

用户自定义类型经常用来表示数据记录，记录一般由多个不同数据类型的元素组成。

下面的示例演示了一个用户自定义类型的大小固定的数组的用法：

```
TypeStateData
   CityCode(1To100)AsInteger     声明静态数组.
   CountyAsString*30
EndType

DimWashington(1To100)AsStateData
```

在上述示例中，StateData 中包括了一个 CityCode 静态数组，且记录 Washington 的结构与 StateData 相同。

当在用户自定义类型中声明大小固定的数组时，必须用数字文字或常数而不能用变量来声明数组的维数。

数组的下界由 OptionBase 语句的设置确定。

请参阅

Data 类型概述，Dim 语句，Enum 语句，OptionBase 语句，Private 语句，Public 语句，ReDim 语句，Static 语句

示例

该示例使用 Type 语句，定义用户自定义的数据类型。Type 语句只能在模块级使用。如果要在类模块中使用，则必须在 Type 语句前冠以关键字 Private。

```
TypeEmployeeRecord '创建用户自定义的类型。
   IDAsInteger '定义元素的数据类型。
   NameAsString*20
   AddressAsString*30
   PhoneAsLong
   HireDateAsDate
EndType
SubCreateRecord()
   DimMyRecordAsEmployeeRecord '声明变量。

   '对 EmployeeRecord 变量的赋值必须在过程内进行。
   MyRecord.ID=12003    '给一个元素赋值。
EndSub
```

TypeName 函数

返回一个 String，提供有关变量的信息。

语法

TypeName(*varname*)

必要的 varname 参数是一个 Variant，它包含用户定义类型变量之外的任何变量。

说明

TypeName 所返回的字符串可以是下面列举的任何一个字符串：

返回字符串	变量
对象类型	类型为 objecttype 的对象
Byte	位值
Integer	整数
Long	长整数
Single	单精度浮点数
Double	双精度浮点数
Currency	货币
Decimal	十进制值
Date	日期
String	字符串
布尔	布尔值
Error	错误值
Empty	未初始化
Null	无效数据
Object	对象
Unknown	类型未知的对象
Nothing	不再引用对象的对象变量

如果 varname 是一个数组，则返回的字符串可以是任何一个后面添加了空括号的可能的返回字符串（或 Variant）。例如，如果 varname 是一个整数数组，则 TypeName 返回"Integer()"。

请参阅

IsArray 函数，IsDate 函数，IsEmpty 函数，IsError 函数，IsMissing 函数，IsNull 函数，IsNumeric 函数，IsObject 函数，VarType 函数，Data 类型概述，VariantData 类型

示例

本示例使用 TypeName 函数返回有关变量的信息。

```
'声明变量。
DimNullVar,MyType,StrVarAsString,IntVarAsInteger,CurVarAsCurrenc
y
DimArrayVar(1To5)AsInteger
NullVar=Null   '设置变量值为 Null。
MyType=TypeName(StrVar)     '返回"String"。
MyType=TypeName(IntVar)     '返回"Integer"。
MyType=TypeName(CurVar)     '返回"Currency"。
MyType=TypeName(NullVar)    '返回"Null"。
MyType=TypeName(ArrayVar)   '返回"Integer()"。
```

UBound 函数

返回一个 Long 型数据，其值为指定的数组维可用的最大下标。

语法

UBound(*arrayname*[,*dimension*])

UBound 函数的语法包含下面部分：

部分	描述
arrayname	必需的。数组变量的名称，遵循标准变量命名约定
dimension	可选的；Variant(Long)。指定返回哪一维的上界。1 表示第一维，2 表示第二维，如此等等。如果省略 dimension，就认为是 1

说明

UBound 函数与 LBound 函数一起使用，用来确定一个数组的大小。LBound 用来确定数组某一维的上界。

对具有下述维数的数组而言，UBound 的返回值见下表：

DimA(1To100,0To3,-3To4)

语句	返回值
uBound(A,1)	100
uBound(A,2)	3
uBound(A,3)	4

请参阅

Dim 语句，LBound 函数，OptionBase 语句，Public 语句，ReDim 语句

示例

该示例使用 UBound 函数，确定数组的指定维的最大可用下标。

```
DimUpper
DimMyArray(1To10,5To15,10To20) '声明数组变量。
DimAnyArray(10)
```

```
Upper=UBound(MyArray,1)      '返回10。
Upper=UBound(MyArray,3)      '返回20。
Upper=UBound(AnyArray) '返回10。
```

UBound 属性

返回控件数组 array 中控件的最高有序值。

语法

object.**UBound**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

UBound 与控件数组中最后一个控件的 Index 属性值相等。

示例

该例子为控件数组打印以上提及的两个属性的值。在一个窗体上放一个 OptionButton 控件，并设置其 Index 属性为 0（用来创建一个控件数组）。要试用此例，可以将该代码粘贴到一个窗体的声明部分中，然后按下 F5 键并单击此窗体。

```
PrivateSubForm_Paint()
   StaticFlagFormPaintedAsInteger
   IfFlagFormPainted<>TrueThen'当窗体进行首次绘画时。
       Fori=1To3
           LoadOption1(i)   '在数组中加入三个选项按钮。
```

```
            Option1(i).Top=Option1(i-1).Top+350
            Option1(i).Visible=True
        NextI
        ForI=0to3    '在选项按钮上放置标题。
            Option1(i).Caption="Option#"&CStr(i)
        NextI
        Option1(0).Value=True    '选中第一个选项按钮。
        FlagFormPainted=True'窗体已完成绘画。
    EndIf
EndSub
PrivateSubForm_Click()
    Print"Controlarray'sCountpropertyis"&Option1().Count
    Print"Controlarray'sLBoundpropertyis"&Option1().LBound
    Print"Controlarray'sUBoundpropertyis"&Option1().UBound
EndSub
```

UCase 函数

返回 Variant(String)，其中包含转成大写的字符串。

语法

UCase(*string*)

必要的 string 参数为任何有效的字符串表达式。如果 string 包含 Null，将返回 Null。

说明

只有小写的字母会转成大写；原本大写或非字母之字符保持不变。

请参阅

LCase 函数

示例

本示例使用 UCase 函数来将某字符串转成全部大写。

```
DimLowerCase,UpperCase
LowerCase="HelloWorld1234" '要输送的字符串。
UpperCase=UCase(LowerCase) '返回"HELLOWORLD1234"。
```

UIDead 属性

返回一个布尔型数值，它指定控件是否应响应用户。

应用于

AmbientProperties 对象

语法

object.**UIDead**

UIDead 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

设置值

UIDead 属性可能的布尔返回值为：

设置值	描述
True	控件不响应用户。
False	控件响应用户。如果容器不实现这种环境属性，这将是缺省的设置值。

说明

该属性通常用来指示容器处于中断模式；这种模式时，控件不应响应任何用户输入。这就是说，控件应忽略鼠标单击和键盘输入，而且即使鼠标在控件窗口上方时也不改变鼠标游标。当程序员中止程序执行时，象 VisualBasic 窗体这样的容器会把这个标志设置为 TRUE——此时容器既不处在设计模式，也不处在运行模式；VisualBasic 只是希望控件处于不工作状态。

UIDefault 属性

返回某个 Member 对象的 UIDefault 属性，或者对其进行设置。

应用于

Member 对象

语法

object.**UIDefault**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

UnattendedApp 属性

返回一个值，该值决定应用程序是否将在没有任何用户接口的情况下运行。

应用于

App 对象

语法

object.**UnattendedApp**=*boolean*

UnattendedApp 属性的语法有这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
boolean	布尔表达式，它指定了应用程序是否将在没有任何用户接口的情况下运行

设置值

boolean **的设置值是：**

常数	值	描述
True	-1	应用程序没有任何用户接口
False	0	应用程序有用户接口

Underline 属性

返回或设置 Font 对象的字形为带下划线或不带下划线。

应用于

Font 对象

语法

object.**Underline**[=*boolean*]

Underline 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定字形的布尔表达式，“设置值”中有详细描述。

设置值

boolean 的设置值为：

设置值	描述
True	打开下划线格式
False	（缺省值）关闭下划线格式化

说明

Font 对象在设计时不能直接使用。取而代之的是，在“属性”窗口中通过选择控件的 Font 属性并单击“属性”按钮来设置 Underline 属性。在“字体”对话框中，选择“下划线”复选框。在运行时，通过为 Font 对象指定 Underline 属性值可直接设置 Underline。

请参阅

Bold 属性，FontTransparent 属性，Italic 属性，Size 属性 (Font)，StriketHrough 属性，Weight 属性，Name 属性

示例

这个例子完成下面的功能：每次单击鼠标时在窗体上打印文本。要试用此例，可以先将下面的代码粘贴到一个窗体的声明部分中，然后按下 F5 键并双击此窗体。

```
PrivateSubForm_Click()
   Font.Bold=NotFont.Bold   '转换 Bold。
   Font.StrikeThrough=NotFont.StrikeThrough'转换 StrikeThrough。
   Font.Italic=NotFont.Italic   '转换 Italic。
   Font.Underline=NotFont.Underline '转换 Underline。
   Font.Size=16'设置 Size 属性。
   IfFont.BoldThen
      Print"Fontweightis"&Font.Weight&"(bold)."
   Else
      Print"Fontweightis"&Font.Weight&"(notbold)."
   EndIf
EndSub
```

Unformat 事件

在 StdFormat 对象对数值解除格式化之后发生。

应用于

StdDataFormat 对象

语法

Sub*object*_**Unformat**(**ByRef** *datavalue* **As StdDataValue**)

Unformat 事件语法有如下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
datavalue	StdDataValue 对象

说明

该事件允许用户进行 StdDataFormat 对象标准设置所不能完成的解除格式化操作。

请参阅

Format 事件(StdDataFormat 对象)

Unload 事件

当窗体从屏幕上删除时发生。当那个窗体被重新加载时，它的所有控件的内容均被重新初始化。当使用在 Control 菜单中的 Close 命令或 Unload 语句关闭该窗体时，此事件被触发。

应用于

PropertyPage 对象，Form 对象，Forms 集合，MDIForm 对象

语法

Private Sub *object*_**Unload**(*cancel* **As Integer**)

Unload 事件语法包括下列部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
cancel	一个整数，用来确定窗体是否从屏幕删除。如果 cancel 为 0，则窗体被删除。将 cancel 设置为任何一个非零的值可防止窗体被删除

说明

将 cancel 设置为任何非零的值可防止窗体被删除，但不能阻止其它事件，诸如从 MicrosoftWindows 操作环境中退出等。可用 QueryUnload 事件阻止从 Windows 中的退出。

在窗体被卸载时，可用一个 Unload 事件过程来确认窗体是否应被卸载或用来指定想要发生的操作。也可在其中包括任何在关闭该窗体时也许需要的验证代码或将其中的数据储存到一个文件中。

QueryUnload 事件在 Unload 事件之前发生。Unload 事件在 Terminate 事件之前发生。

使用 Unload 语句或在一个窗体的“控件”菜单上选择“关闭”命令，用“任务窗口”列表上的“结束任务”按钮退出应用程序，在当前窗体为其一个子窗体的情况下关闭该 MDI 窗体，或当应用程序正在运行的时候退出 MicrosoftWindows 操作环境等情况都可引发 Unload 事件。

请参阅

Load 事件，QueryUnload 事件，Load 语句，Unload 语句示例

本例演示在用各种各样的信息框提示用户的同时，关闭一个窗体的简单过程。在实际应用中，可以往通用的 Sub 过程中添加仿真 VisualBasic 中 File 菜单上的 Exit、Save 和 SaveAs 命令的调用。要尝试这个例子，可以将代码粘贴到一个窗体的声明部分，然后按 F5 键。一旦窗体显示出来，可按 ALT+F4 关闭它。

```
PrivateSubForm_Unload(CancelAsInteger)
    DimMsg,Response  '声明变量。
    Msg="SaveDatabeforeclosing?"
    Response=MsgBox(Msg,vbQuestion+vbYesNoCancel,"SaveDialog")
    SelectCaseResponse
        CasevbCancel '不允许关闭。
            Cancel=-1
            Msg="Commandhasbeencanceled."
        CasevbYes
        '这里输入保护数据的代码。
            Msg="Datasaved."'
        CasevbNo
            Msg="Datanotsaved."
    EndSelect
    MsgBoxMsg,vbOKOnly,"Confirm" '显示信息。
EndSub
```

Unload 事件(DHTMLPage)

响应用户而漫游离开给定的 HTML 页面或关闭浏览器时发生。

应用于

DHTMLPage 对象

语法

PrivateSub*object*_**Unload**

object 所在处表示一个对象表达式，其值为“应用于”列表中的一个对象。

说明

可以使用 Unload 事件在代码执行终止前清除代码。当应用程序激发该事件，您可能希望通过关闭打开的文件、删除临时文件等方法来清除。当 Unload 事件激发时，所有在 HTML 页面上的对象仍然存在。

Unload 语句

从内存中卸载窗体或控件。

语法

Unload*object*

object 所在处是要卸载的 Form 对象或控件数组元素的名称。

说明

当所占内存另有它用，或需要重新设置窗体、控件的属性为初始

值时，就有必要卸载窗体或控件。

在卸载窗体前，会发生 Query_Unload 事件过程，然后是 Form_Unload 事件过程。在其中任一过程中设置 cancel 参数为 True 可防止窗体被卸载。若为 MDIForm 对象，先发生 MDIForm 对象的 Query_Unload 事件过程,接着是各 MDI 子窗体的 Query_Unload 事件过程和 Form_Unload 事件过程，最后是 MDIForm 对象的 Form_Unload 事件过程。

当窗体卸载之后，所有在运行时放到该窗体上的控件都不再是可访问的。在设计时放到该窗体上的控件将保持不变；但是，当窗体重新加载时，在运行时对这些控件及其属性的任何更改将丢失。所有对于窗体属性的更改也将丢失。对窗体上任何控件的访问会导致窗体重新加载。

注意在卸载窗体时，只有显示的部件被卸载。和该窗体模块相关联的代码还保持在内存中。

只有在运行时添加到窗体上的控件数组元素才能用 Unload 语句卸载。重新加载被卸载的控件时，其属性会被重新初始化。

请参阅

QueryUnload 事件，Load 语句，Hide 方法，Show 方法

示例

这个示例使用 Unload 语句来卸载 Form 对象。在运行此例前，在 Form 对象的声明部分粘贴以下代码，然后运行此例并单击该 Form 对象。

```
PrivateSubForm_Click()
```

```
    DimAnswer,Msg    '声明变量。
    UnloadForm1 '卸载窗体。
    Msg="Form1hasbeenunloaded.ChooseYestoloadand"
    Msg=Msg&"displaytheform.ChooseNotoloadtheform"
    Msg=Msg&"andleaveitinvisible."
    Answer=MsgBox(Msg,vbYesNo)   '获得用户响应。
    IfAnswer=vbYesThen   '测试应答。
        Show'如果回答 Yes，则显示窗体。
    Else
        LoadForm1    '如果回答 No，仅加载窗体。
        Msg="Form1isnowloaded.ChooseOKtodisplayit."
        MsgBoxMsg    '显示信息。
        Show'显示窗体。
    EndIf
EndSub
```

Update 方法

用和用户已经打开“外接程序管理器”对话框一样的方法，刷新来自列在 Vbaddin.ini 文件中的外接程序的 AddIns 集合的内容。

应用于

AddIns 集合

语法

object.**Update**

其中 *object* 为对象表达式，其值为“应用于”列表中的对象。

说明

所有列在 Vbaddin.ini 文件中的外接程序，在它们能在 VisualBasic 中使用之前，必须是在注册表中注册的 ActiveX 部件。

Update 方法（OLE 容器）

从支持对象的应用程序检索当前数据，并在 OLE 容器控件中将该数据作为图形显示。

应用于

OLEContainer 控件

语法

object.**Update**

object 是一个对象表达式，其值是“应用于”列表中的一个对象。

Updated 事件

当修改对象的数据时，发生该事件。

应用于

OLEContainer 控件

语法

Sub*object*_**Updated**(*code* **As Integer**)

Updated 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
code	整数，指定对象是如何更新的，在“设置值”中有详细说明

设置值

code 的设置值是：

常数	值	描述
vbOLEChanged	0	对象的数据已经改变
vbOLESaved	1	对象的数据已由创建该对象的应用程序保存
vbOLEClosed	2	含有链接对象数据的文件已被创建该对象的应用程序关闭
vbOLERenamed	3	含有链接对象数据的文件已被创建该对象的应用程序重命名

说明

这些常数列在“对象浏览器”的 VisualBasic(VB)对象库中。可以使用这个事件，确定在上次保存之后对象的数据是否已被修改。为此，在 Updated 事件中设置一个全局变量，来指示需要保存的数据。在保存对象之后，将该变量复位。

请参阅

Class 属性,OLEType 属性,UpdateOptions 属性,OLETypeAssowed

属性

UpdateMode 属性

返回或设置集合的更新方式。

应用于

BindingCollection 对象

语法

object.**UpdateMode**[=*integer*]

UpdateMode 属性的语法包含如下内容：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
integer	一个指定更新方式的值或常数，如在设置值中所示

设置值

对 integer 的设置如下：

常数	值	描述
vbUsePropertyAttributes	0	记录集根据“属性特性(propertyAttributes)”对话框的“数据绑定”窗格中找到的设置值进行更新
vbUpdateWhenPropertyChanges	1	记录集在属性值更改时更新。
vbUpdateWhenRowChanges	2	记录集仅在用户离开行时更新

说明

数据源是否是可更新的取决于数据供应程序。例如，如果某个数据源是一个由于 LockType 属性被设置为 adLockReadOnly 而打开的 ADO 记录集，那么，该数据源就不能更新。

UpdateOptions 属性

返回或设置一个值，指示当链接数据修改后如何更新对象。

应用于

OLE 包容器控件

语法

object.**UpdateOptions**[=*number*]

UpdateOptions 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	整数，指示如何更新对象，在“设置值”中有详细说明

设置值

number **的设置值是：**

常数	值	描述
vbOLEAutomatic	0	（缺省值）自动的。每次改变链接数据时均更新对象
vbOLEFrozen	1	冻结的。无论何时从生成对象的应用程序内保存链接数据，均更新对象
vbOLEManual	2	手动的。只有使用 Update 方法才更新对象

说明

对链接的对象来说，当其它用户或应用程序能够存取和修改链接数据时，这个属性是很有用的。

当改变对象的数据时，调用 Updated 事件。

请参阅

Class 属性，OLEType 属性，Updated 事件，OLETypeAllowed 属性，Update 方法(OLE 包容器)

URLData 属性

设置或返回一个参数，该参数追加到任何用 URLFor **方法生成、或在**

WriteTemplate 处理中发现的 URL 的末尾。

应用于

WebItem 对象

语法

object.URLData[=s*tring*]

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
string	追加到 URL 末端的信息，URL 通过 WebClass 生成

说明

该信息被自动追加到在用 WriteTemplate 处理 HTML 模板过程中或调用 URLFor 方法过程中发现的每一个 URL 上。可以在追加的参数中存储状态信息。通过使用 Request.QueryString 集合，可以在随后的调用中检索状态。

请参阅

第三章的“指定 WebItem 的 URL”、“IIS 应用程序中的状态管理”和“在浏览器和服务器之间的移动状态”，第五部分的“开发 IIS 应用程序”和《MicrosoftVisualBasic6.0 部件工具指南》的“构造 Internet 应用程序”

URLFor 方法

用于指定系统引用 webclass 的 HTML 模板或浏览器中 WebItem 所需要的统一资源定位程序(URL)。

应用于

WebClass 对象

语法

object.URLFor(*WebItemobject***AsWebItem,[*eventname*])**

URLFor 方法的语法有这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
WebItemobject	希望为其生成一个 URL 并激发一个事件的 WebItem 对象
eventname	引用 WebItem 中一个事件的字符串——如果忽略，则引用 Respond 事件。返回的 URL 可以根据客户响应被嵌入，并且在被引用时，将触发 WebClass 中相关的事件

说明

也可以用于生成一个运行时包含 URL 的字符串，该字符串引用 WebItem 并指定要激发的一个自定义事件。当接收到一个在设计时没有定义的事件名称时，URLFor 方法返回一个 URL，该 URL 在由客户处理时，激发相关的 WebItem 中的 UserEvent 事件。运行时定义的事件名称将传递给 UserEvent 事件。

请参阅

第三章的“指定 WebItem 的 URL”、第五部分的“开发 IIS 应用程序”和《MicrosoftVisualBasic6.0 部件工具指南》的“构造

Internet 应用程序”。

UseMaskColor 属性

返回或设置一个值，该值决定了赋值给 MaskColor 属性的颜色是否被用作一个“掩码”。（也就是说，用来创建透明区域。）

应用于

CheckBox 控件，CommandButton 控件，OptionButton 控件

语法

object.**UseMaskColor**[=*boolean*]

UseMaskColor 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个布尔表达式，用来指定赋值给 MaskColor 属性的颜色是否被用作掩码

设置值

boolean 的设置值为：

设置值	描述
True	赋值给 MaskColor 属性的颜色被用作一个掩码，在该颜色所在处创建透明
False	（缺省）赋值给 MaskColor 属性的颜色被忽略，并且该颜色仍然不透明

UseMnemonic 属性

返回或设置一个值，该值用来指定是否要在 Label 控件的 Caption 属性的文本中包含一个&字符来定义一个访问键。

应用于

Label 控件

语法

object.**UseMnemonic**[=*boolean*]

UseMnemonic 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	一个用来指定 Label 控件是否可使用一个访问键的布尔表达式，“设置值”中有详细描述

设置值

boolean 的设置值为：

设置值	描述
True	（缺省值）任何出现在 Caption 属性文本中的&字符将使得紧接&字符之后的字符成为一个访问键。&字符本身并不显示在 Label 控件的界面中
False	任何出现在 Caption 属性文本中的&字符将被当作&字符显示在 Label 控件的界面中

说明

在运行时，按下 ALT 加上在 Label 控件的 Caption 属性中定义的访问键，将把焦点移到在 tab 键次序中紧接 Label 控件之后的控件上。

请参阅

Caption 属性，Caption 属性(ActiveX 控件)

示例

这个例子读取 Label 控件的 UseMnemonic 属性的设置。要试用此例，可以先将该代码粘贴到包含 Label 窗体的声明部分中，然后按下 F5 键并单击此窗体。

```
PrivateSubForm_Click()
    IfLabel1.UseMnemonicAndInStr(Label1,"&")Then
        MsgBox"Thelabelhasanaccesskeycharacter."
    ElseIfLabel1.UseMnemonicAndNotInStr(Label1,"&")Then

    MsgBox"Thelabelsupportsanaccesskeycharacterbutdoesn'thaveanam
persand."
    Else
        MsgBox"Thelabeldoesn'tsupportanaccesskeycharacter."
    EndIf
EndSub
```

UserControl 对象

UserControl 对象是用于创建 ActiveX 控件的对象。

说明

用 VisualBasic 所创建的 ActiveX 控件总是由 UserControl 对象加上选中放到 UserControl 上的任何控件（称为组成控件）所组成。就象 VisualBasic 窗体一样，UserControl 对象具有代码模块以及可视化的设计器。将组成控件放到 UserControl 对象的设计器上，就象把控件放到窗体上一样。

属性

DataMembers 属性，AccessKeys 属性，Ambient 属性，BackStyle 属性(UserControl 对象)，ContainedControls 属性，EdiTAtDesignTime 属性,EventsFrozen 属性,Extender 属性,ParentControls 属性，PropertyPages 属性，RightToLeft 属性，ClipBehavior 属性，HitBehavior 属性，ContainerHWnd 属性，Control 属性，Hyperlink 属性，Palette 属性，PaletteMode 属性，OLEDragMode 属性，Parent 属性(UserControl 对象)，ParentControlsType 属性，MaskColor 属性(UserControl 对象)，MaskPicture 属性(UserControl 对象)，BackColor，ForeColor 属性，DrawWidth 属性，FontBold，FontItalic，FontStrikethru，FontUnderline 属性,FontName 属性,FontSize 属性,Height,Width 属性,Picture 属性,Tag 属性,AutoRedraw 属性,ClipControls 属性,CurrentX，CurrentY 属性，DrawMode 属性，DrawStyle 属性，FillColor 属

性,FillStyle 属性,hDC 属性,hWnd 属性,Image 属性,KeyPreview 属性，HasDC 属性，MouseIcon 属性，MousePointer 属性 ScaleHeight，ScaleWidth 属性，ScaleLeft，ScaleTop 属性，ScaleMode 属性，ActiveControl 属性，Appearance 属性，FontTransparent 属性,Count 属性(VB 集合),Enabled 属性,Name 属性，Font 属性.

方法

AsyncRead 方法，CancelAsyncRead 方法，CanPropertyChange 方法,PropertyChanged 方法,Size 方法,Refresh 方法,SetFocus 方法,Circle 方法,Line 方法,Pset 方法,Cls 方法,PaintPicture 方法，Point 方法，PopupMenu 方法，Scale 方法，ScaleX,ScaleY 方法，TextHeihgt 方法，TextWidth 方法，OLEDrag 方法，Print 方法

事件

AsyncReadComplete 事件，EnterFocux 事件，ExitFocus 事件，GotFocus 事件(UserContrlObjectandUserDocumentObject)，Hide 事件（UserControlObject），InitProperties 事件，LostFocus 事件（UseControlObjectandUserDocumentObject），ReadProperties 事件，Show 事件，WriteProperties 事件，Click 事件，DragDrop 事件，DragOver 事件，KeyDow,KeyUp 事件，KeyPress 事件，MouseDown,MouseUp 事件，MouseMove 事件，Paint 事件，Resize 事件，Scroll 事件，DblClick 事件，Initialize

事件，Terminate 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDragOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件

请参阅

Hyperlink 对象；UserDocument 对象

UserDocument 对象

ActiveX 文档的基类对象，UserDocument 对象类似于带有某些例外的标准 VisualBasic 的 Form 对象。

说明

UserDocument 对象具有大多数（但并非全部）可以在 Form 对象上找到的事件。出现在 Form 上（但并未在 UserDocument 上找到）的事件包括：Activate、Deactivate、LinkClose、LinkError、LinkExecute、LinkOpen、Load、QueryUnload 以及 Unload 等事件。

出现在 UserDocument 上，但不能在 Form 对象上找到的事件包括：AsyncReadComplete、EnterFocus、ExitFocus、Hide、InitProperties、ReadProperties、Scroll、Show 及 WriteProperties 等事件。

不能将内嵌对象（比如 Excel 或者 Word 文档）或 OLEContainer 控件放到 UserDocument 上。

属性

ContainedControls 属性, RightToLeft 属性, ContinuousScroll 属性, Controls 属性, HscrollSmallChange,VscrollSmallChange 属性, Hyperlink 属性, MinHeight,MinWidth 属性, Palette 属性, PaletteMode 属性, ViewportHeight,ViewportLeft,ViewportTop,ViewportWidth 属性, OLEDropMode 属性, BackColor, ForeColor 属性, DrawWidth 属性, FontBold,FontItalic,FontStrikethru,FontUnderline 属性, FontName 属性, FontSize 属性, Height,Width 属性, Picture 属性, Tag 属性, AutoRearaw 属性, ClipControls 属性, CurrentX,CurrentY 属性, DrawMode 属性, DrawStyle 属性, FillColor 属性, FillStyle 属性, hDC 属性, hWnd 属性, Image 属性, KeyPreview 属性, MouseIcon 属性, MousePointer 属性, ScaleLeft,ScaleTop 属性, ScaleMode 属性, ScrollBars 属性, ScaleHeihgt,ScaleWidth 属性, ActiveControl 属性, Appearance 属性, FontTransparent 属性, Count 属性 (VB 集合), Name 属性, Parent 属性, Font 属性, Appearance 属性 (ActiveX 控件)

方法

Asyncaraead 方法, CancelAsyncRead 方法, PropertyChanged 方法, SetViewport 方法, Refresh 方法, SetFocus 方法, Circle 方法, Line 方法, Pset 方法, Cls 方法, PaintPicture 方法, Point 方法, PopupMenu 方法, PrintForm 方法, Scale 方法, ScaleX,ScaleY 方法, TextHeihgt 方法, TextWidth 方法, OLEDrag

方法

事件

AsyncReadComplete 事件，EnterFocux 事件，ExitFocus 事件，GotFocus 事件，Hide 事件，InitProperties 事件，LostFocus 事件，ReadProperties 事件，Show 事件，WriteProperties 事件，Click 事件，DragDrop 事件，DragOver 事件，KeyDow,KeyUp 事件，KeyPress 事件，MouseDown,MouseUp 事件，MouseMove 事件，Paint 事件，Resize 事件，Scroll 事件，DblClick 事件，Initialize 事件，Terminate 事件，OLECompleteDrag 事件，OLEDragDrop 事件，OLEDagOver 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件，OLEGiveFeedback 事件，OLESetData 事件，OLEStartDrag 事件

请参阅

Hyperlink 对象，UserControl 对象，Form 对象，Forms 集合

UserEvent 事件

在对运行时定义事件的激发做出响应时发生。

语法

PrivateSub*object*_UserEvent(*eventname*AsString)

UserEvent 事件语法有这些部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
eventname	包含一个在运行时添加的自定义事件名称

说明

当接收到一个在设计时没有定义的事件名称时，URLFor 方法返回一个 URL，该 URL 在由客户处理时，激发相应 WebItem 上的 UserEvent 事件。运行时定义的事件名称将会传给 UserEvent 事件。使用 URLFor 方法可以生成一个 URL，该 URL 在响应用户操作，使用浏览器检索时，将调用 UserEvent 方法。WebClass 将按照事件的含义采取相应的操作。您要在 UserEvent 事件内的 WebClass 中为所有这样的事件编写事件过程。如果有一个包含多个用户事件的 WebClass，使用一个条件语句，例如 If 或 SelectCase，来指定系统针对每一个用户执行的操作。

请参阅

URLFor 方法;第三章的“Webclass 事件，”“标准 Webclass 事件，”和“在运行时定义 Webclass 事件，第五部分的“开发 IIS 应用程序”和《MicrosoftVisualBasic6.0 部件工具指南》的“构造 Internet 应用程序”。

UserMode 属性

返回一个布尔值，它指示控件正被窗体设计者使用，还是正被窗体用户使用。

应用于

AmbientProperties 对象

语法

object.**UserMode**

UserMode 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象

设置值

UserMode 属性可能的布尔返回值为：

设置值	描述
True	控件正被窗体用户使用。如果容器不实现这种环境属性，这将是缺省的设置值。在 VisualBasic 中，这就是运行模式
False	控件正在由窗体设计者（开发者）使用。在 VisualBasic 中，这就是“设计模式”

Val 数

返回包含于字符串内的数字，字符串中是一个适当类型的数值。

语法

Val(*string*)

必要的 string 参数可以是任何有效的字符串表达式.

说明

Val 函数，在它不能识别为数字的第一个字符上，停止读入字符串。那些被认为是数值的一部分的符号和字符，例如美元号与逗号，都不能被识别。但是函数可以识别进位制符号&O（八进制）和&H（十六进制）。空白、制表符和换行符都从参数中被去掉。

下面的返回值为 1615198：

```
Val("1615198thStreetN.E.")
```

在下面的代码中，Val 为所示的十六进制数值返回十进制数值-1。

```
Val("&HFFFF")
```

注意 Val 函数只会将句点（.）当成一个可用的小数点分隔符。当使用不同的小数点分隔符时,如在国际版应用程序中,代之以 CDbl 来把字符串转换为数字。

请参阅

类型转换函数，Str 函数

示例

本示例使用 Val 函数返回字符串中所含的数值。

```
DimMyValue
MyValue=Val("2457")      '返回 2457。
MyValue=Val("2457")      '返回 2457。
MyValue=Val("24and57") '返回 24。
```

Validate 事件

在焦点转换到一个（第二个）控件之前发生，此时该控件的 CausesValidation 属性值设置为 True。

应用于

DataRepeater 控件，DataList 控件，Extender 对象，ADOData 控件，TreeView 控件，ImageCombo 控件，ListView 控件，Slider 控件，TabStrip 控件，Button 控件，DateTimePicker 控件，Animation 控件，MonthView 控件，UpDown 控件，DataGrid 控件，DataCombo 控件，DBCombo 控件，CheckBox 控件(Lightweight)，ComboBox 控件(Lightweight)，CommandButton 控件(LiGhtweight)，HScrollBar，VScrollBar 控件(Lightweight)，ListBox 控件(Lightweight)，OptionButton 控件(Lightweight)，TextBox 控件(Lightweight)，MaskedEdit 控件，MultimediaMCI 控件，MSChart 对象，SSTab 控件，RichTextBox 控件，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，HScrollBar，VScrollBar 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件

语法

Private Sub *object*_**Validate**(*KeepFocus* **As Boolean**)

Validate 事件语法包含下面几部分：

部分	描述
object	一个对象表达式，其值为“应用于”列表中的一个对象
KeepFocus	确定控件是否失去焦点的值。KeepFocus 设置为 True 时，控件保持焦点

说明

Validate 事件和 CausesValidation 属性协同工作，防止控件失去焦点直到满足确定的准则。

重点只有在即将获得焦点的控件的 CausesValidation 属性值设置为 True 时，Validate 事件才发生。

示例

该示例使用三个控件来示范 Validate 事件和 CausesValidation 属性的使用。在缺省情况下，两个 TextBox 控件的 CausesValidation 属性设置为 True，这样当您想把焦点从一个 TextBox 转换到另一个时，Validate 事件发生。如果 Text1 没有包含日期或 Text2 没有包含一个大于 10 的数字，焦点的转换将被阻止。由于 Command1 控件的 CausesValidation 属性设置为 False，因此您无论何时都可以单击 Help 按钮。

要试验该示例，在窗体中放置一个 CommandButton 和两个 TextBox 控件，将代码粘接到窗体的“声明”部分并运行此工程。按 Tab 键尝试转换焦点。

```
PrivateSubForm_Load()
```

```
    '设置按钮的 CausesValidation 属性为 False。当用户
    '单击按钮时，Validate 事件不发生。
    '设置按钮的 Caption 属性为“帮助”。
    WithCommand1
        .CausesValidation=False
        .Caption="Help"
    EndWith

    Show
    WithText1'选择 Text1 的文本并为它设置焦点。
        .SelLength=Len(Text1.Text)
        .SetFocus
    EndWith
EndSub

PrivateSubCommand1_Click()
    '当单击此按钮时给出用户帮助信息。
    MsgBox_
    "Text1mustbesettoadate."&VbCrLF&_
    "Text2mustbeanumberlessthan10."
EndSub

PrivateSubText1_Validate(KeepFocusAsBoolean)
```

```
    '如果值不是一个日期，则保持焦点，除非用户
    '单击 Help。
    IfNotIsDate(Text1.Text)Then
        KeepFocus=True
        MsgBox"Pleaseinsertadateinthisfield.",, "Text1"
    Endif
EndSub

PrivateSubText2_Validate(KeepFocusAsBoolean)
    '如果值是一个大于 10 的数字，保持焦点。
    IfNotIsNumeric(Text2.Text)OrVal(Text2.Text)>10Then
        KeepFocus=True
MsgBox_
"Pleaseinsertanumberlessthanorequalto10.",,"Text2"
    EndIf
EndSub
```

Value 属性

CheckBox 和 OptionButton 控件——返回或设置控件的状态。

CommandButton 控件——返回或设置指示该按钮是否可选的值；在设计时不可用。

Field 对象——返回或设置字段的内容；在设计时不可用。

HScrollBar 和 VScrollBar 控件（水平和垂直滚动条）——返回或设置滚动条的当前位置，其返回值始终介于 Max 和 Min 属性值之间，包括这两个值。

应用于

Column 对象，Parameter 对象(VisualBasic)，AsyncProperty 对象，CheckBox 控件，CommandButton 控件，HScrollBar，VScrollBar 控件，OptionButton 控件

语法

object.**Value**[=*value*]

Value 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	该值指定控件的状态，内容或位置，“设置值”中有详细说明

设置值

value 的设置值为：

CheckBox 控件——0 是没有检查（缺省值），1 为已检查，和 2 为变灰（变暗）。

CommandButton 控件——True 表示已选择该按钮；False（缺省值）表示没有选择该按钮。如果在代码中设置 Value 属性值为 True 激活该按钮的 Click 事件。

Field 对象——只受字段数据类型的限制。

HScrollBar 和 VScrollBar 控件——设置介于-32,768 与 32,767 之间的值以定位滚动框。

OptionButton 控件——True 表示已选择了该按钮；False（缺省值）表示没有选择该按钮。

说明

采用对象的缺省属性，不必在代码中指定它。例如，Field 是任何 Recordset 的缺省属性，Value 是 Field 对象的缺省属性。这使得下面的两条语句等效：

```
Dn.Fields("PubID").Value=X
Dn.("PubID")=X
```

第一个语句指定了缺省属性；第二个语句假定了该属性。

请参阅

Max，Min 属性(滚动条)；第七章的“使用控件值”，《MicrosoftVisualBasic6.0 程序员指南》的“使用 VisualBasic 标准控件”。

示例

这个例子在 TextBox 控件中显示 HScrollBar（水平滚动条）控件的数值。要尝试这个例子，请将代码粘贴到包含一个 TextBox 控件和一个 HScrollBar 控件的窗体的声明部分。按 F5 键运行该程序，然后单击滚动条。

```
PrivateSubForm_Load()
   HScroll1.Min=0   '初始化滚动条。
   HScroll1.Max=1000
```

```
    HScroll1.LargeChange=100
    HScroll1.SmallChange=1
EndSub

PrivateSubHScroll1_Change()
    Text1.Text=Format(HScroll1.Value)
EndSub
```

Value 属性(VBA 外接程序对象模式)

返回或设置一个 Variant 型数，它指定该属性之值，此属性可读/写。

应用于

Property 对象

说明

由于 Value 属性返回的是一个 Variant，所以可以访问任何属性。要访问一个列表，请使用 IndexedValue 属性。

如果 Property 对象所代表的属性是可读写的，则 Value 属性便是可读写的。若该属性是只读的，则对 Value 属性进行设置，可导致错误。若该属性是只写的，则返回 Value 属性的操作，可导致错误。

Value 属性是 Property 对象的默认属性。

请参阅

IndexedValue 属性(VBA 外接程序对象模型)，Name 属性(VBA 外接程序对象模型)，Object 属性，Type 属性(VBA 外接程序对象模型)

示例

下列示例使用 Value 属性返回 VBComponents 集合中某个成员的指定属性的值。

```
Debug.PrintApplication.VBE.
ActiveVBProject.VBComponents(1).Properties("AcceptLabelsInFormul
as").Value
```

Value 属性（Format 对象）

设置或返回 StdDataValue 对象的值。仅能在运行时进行读写。

应用于

StdDataValue 对象

语法

object.**Value**[=*value*]

Value 属性语法有如下部分：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
value	可选的变体型。是被 StdDataFormat 对象格式化的数值

说明

设置值或返回值是一个恰当的 StdDataFormat 对象的 Type 属性值。

请参阅

StdDataFormat 对象，Type 属性

VarType 函数

返回一个 Integer，指出变量的子类 5 型。

语法

VarType(*varname*)

必要的 varname 参数是一个 Variant，包含用户定义类型变量之外的任何变量。

返回值:

常数	值	描述
vbEmpty	0	Empty（未初始化）
vbNull	1	Null（无有效数据）
vbInteger	2	整数
vbLong	3	长整数
vbSingle	4	单精度浮点数
vbDouble	5	双精度浮点数
vbCurrency	6	货币值
vbDate	7	日期
vbString	8	字符串
vbObject	9	对象
vbError	10	错误值
vbBoolean	11	布尔值
vbVariant	12	Variant（只与变体中的数组一起使用）
vbDataObject	13	数据访问对象
vbDecimal	14	十进制值
vbByte	17	位值
vbUserDefinedType	36	包含用户定义类型的变量
vbArray	8192	数组

注意这些常数是由 VisualBasic 为应用程序指定的。这些名称可以在程序代码中到处使用，以代替实际值。

说明

VarType 函数自身从不对 vbArray 返回值。VarType 总是要加上一些其他值来指出一个具体类型的数组。常数 vbVariant 只与 vbArray 一起返回，以表明 VarType 函数的参数是一个 Variant 类型的数组。例如，对一个整数数组的返回值是 vbInteger+vbArray，或 8194。如果一个对象有缺省属性，则 VarType(object)返回对象缺省属性的类型。

请参阅

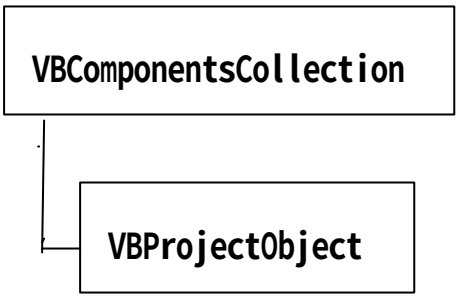
Data 类型概述，Variant 数据类型，IsArray 函数，IsDate 函数，IsEmpty 函数，IsError 函数，IsMissing 函数，IsNull 函数，IsNumeric 函数，IsObject 函数，TypeName 函数

示例

本示例使用 VarType 函数决定变量的次类型（subtype）。

```
DimIntVar,StrVar,DateVar,MyCheck
'初始化变量。
IntVar=459:StrVar="HelloWorld":DateVar=#2/12/69#
MyCheck=VarType(IntVar)      '返回2。
MyCheck=VarType(DateVar)     '返回7。
MyCheck=VarType(StrVar)      '返回8。
```

VBComponent 对象



代表一个包含在工程中的部件，例如类模块或标准模块。

说明

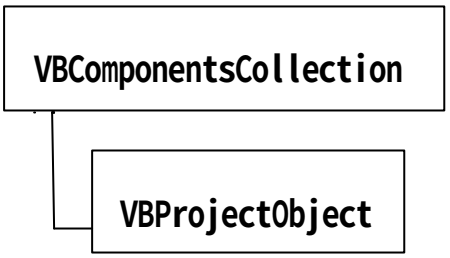
使用 VBComponent 对象访问与部件关联的代码模块或改变部件的属性设置。

可以使用 Type 属性找出 VBComponent 对象所引用的部件类型，使用 Collection 属性找出该部件在哪个集合中。

属性

IsDirty 属性，DesignerWindw 方法，CodeModule 属性，集合属性，Designer 属性，HasOpenDesigner 属性，Name 属性(VBA 外接程序对象模型)，Saved 属性，Type 属性(VBA 外接程序对象模型)，VBE 属性，FileCount 属性，FileNames 属性，IconState 属性，HelpContextID 属性，Description 属性

VBComponents 集合



代表工程中的部件。

说明

使用 VBComponents 集合在工程中访问、添加和删除部件。部件可以是窗体、模块、或类。VBComponents 集合是标准集合，可以在 ForEach 块中使用。

可以使用 Parent 属性返回 VBComponents 集合所在的工程。

在 VisualBasicforApplications 中，可以使用 Import 方法把文件中的部件添加到工程中。

属性

Count 属性，Parent 属性，VBE 属性，StartUpObject 属性

方法

Add 方法(VBA 外接程序对象模型)，Import，Item 方法(VBA 外接程序对象模型),Remove 方法(VBA 外接程序对象模型),AddCustom 方法，AddFile 方法，AddFromTemplate 方法

事件

ItemReloaded 事件，ItemAdded 事件(VBA 外接程序对象模型)，

ItemRemoved 事件，(VBA 外接程序对象模型)，ItemActivated 事件，ItemRenamed 事件，ItemSelected 事件

请参阅

CodeModule 对象，VBComponent 对象，VBProject 对象

VBComponents 属性

返回包含在一个工程中的部件的集合。

应用于

VBProject 对象

说明

使用 VBComponents 集合访问、增加或删除工程中的部件。部件可以是窗体、模块或类。VBComponents 集合是标准集合，可用 For…Each 块中。

可用 Parent 属性返回 VBComponents 集合所在的工程名。

在用 VisualBasic 编写应用程序时，可以用 Import 方法从文件往工程中增加部件。

VBComponentsEvents 对象

在 VisualBasic 工程中添加、删除、重新命名或激活对象时将产生事件，该对象就表示这些事件的源。

语法

VBComponentsEventsProperty

VBComponentsEvents 属性

该属性返回类型 VBComponentsEvents **的事件对象。**

应用于

Events 对象

语法

object.**VBComponentsEvents**(*vbproject***AsvbProject**)

VBComponentsEvents **属性的语法包含下面部分：**

部分	**描述**
object	**对象表达式，其值是“应用于”列表中的对象**
vbproject	**指定包含该部件工程类型的** vbProject **对象**

VBControl 对象

表示工程中的一个部件上的控件。

语法

VBControl

说明

程序可通过 VBForm **对象访问控件。用** VBForm **对象可以：**

访问控件的所有设计时属性。

标识控件的容器。

改变控件的 Z-顺序。

VBControl 对象取代了 VisualBasic4.0 版中的 ControlTemplate 对象。

属性

Properties 集合，ControlObject 属性，ControlType 属性，InSelection 属性，ProgID 属性 mContainer 属性，Collection 属性，VBE 属性

方法

ZOrder 方法

请参阅

VBControls 集合，VBControlsEvents 对象，VBControlsEvents 属性

VBControlExtender 对象

提供 VisualBasicVBControlExtender 的各种属性。

语法

VBControlExtender

说明

VBControlExtender 对象主要被用于动态地使用 Add 方法将控件添加到 Controls 集合这样的情况。在这一点上，VBControlExtender 对象对开发者是最有用的，因为它提供了一系列通用属性、事件和方法。对象的另一个功能是 ObjectEvent 事件，该事件的作用是

对动态添加的控件发出的所有事件进行解释。下面的例子中首先声明了一个 VBControlExtender 类型的对象变量，然后在添加控件的时候设置变量。该示例还说明了如何编写 ObjectEvent 事件处理程序。

```
OptionExplicit
DimWithEventsobjExtAsVBControlExtender' 使 用 WithEvents 声 明
'VBControlExtender 变量
PrivateSubLoadControl()
   Licenses.Add"Project1.Control1","ewrinvcmcoe"
   SetobjExt=Controls.Add("Project1.Control1","myCtl")
   objExt.Visible=True'该控件在缺省情况下为不可见的。
EndSub

PrivateSubextObj_ObjectEvent(InfoAsEventInfo)
   '使用 SelectCase 语句编写控件的事件处理程序。
   SelectCaseInfo.Name
   Case"Click"
        '在此处理 Click 事件。
   '在这里处理其他情况
   CaseElse'未知事件
        '在此处理各种未知的事件。
   EndSelect
EndSub
```

将引用值赋值给变量时的有关限制

将 VBControlExtender 对象设置到动态添加的控件时，需要注意的是：内部控件不能够被赋值给变量。

VBControls 集合

返回窗体上所有部件的集合。

语法

VBControls

说明

程序可通过 VBControls 集合访问多个控件。用 VBControls 集合可以：

访问部件中的所有控件。

逐步遍历控件集合。

返回特定控件。

向部件添加控件。

Item 方法决定 VBControls 集合的缺省值。

VBControls 集合取代了 VisualBasic4.0 版中的 ControlTemplates 集合。

属性

Count 属性(VB 集合)，Parent 属性，VBE 属性

方法

Add 方法，Item 方法，Remove 方法(VisualBasicExtensibility)

事件

ItemAdded 事件，ItemRemoved 事件，ItemRenamed 事件

请参阅

VBControl 对象，VbControlsEvents 对象，VBControlsEvents 属性

VBControls 属性

该属性返回一个包含窗体中所有控件的集合。

应用于

VBForm 对象

语法

object.**VBControls**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

VBControlsEvents 对象

在 VisualBasic 工程中添加、删除、重新命名或激活控件时将产生事件，该对象就表示这些事件的源。

语法

VBControlsEvents

事件

ItemAdded 事件，ItemRemoved 事件，ItemRenamed 事件

请参阅

VBControl 对象，VBControls 集合，VBControlsEvents 属性

VBControlsEvents 属性

该属性返回窗体上控件支持的全部事件。

应用于

Events 对象

语法

object.**VBControlsEvents**(*vbproject*As**Variant**,*vbform*As**VBForm**)

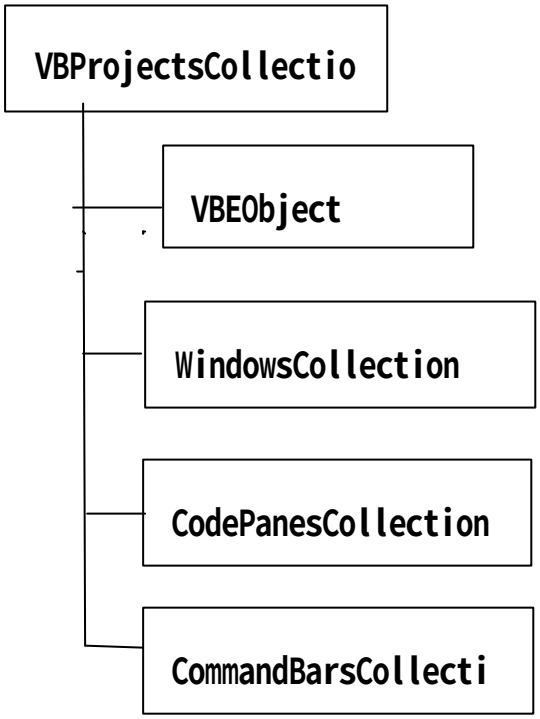
VBControlsEvents 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的对象
vbproject	指定包含这些控件的工程的 variant 表达式
vbform	包含这些控件的窗体

说明

返回类型 VBControlsEvents 的事件对象。该事件源于 VBForm 或可以包含控件的 VBForm 上的控件。

VBE 对象



根对象，它包含所有其它可在 VisualBasicforApplications 中表示的对象和集合。

说明

可用下列集合访问 VBE 对象中所包含的对象：

- 可用 VBProjects 集合访问工程的集合。
- 可用 Windows 集合访问窗口的集合。
- 可用 CodePanes 集合访问代码窗格的集合。
- 可用 CommandBars 集合访问命令栏的集合。

可用 Events 对象访问属性，该属性使外接程序能够连接到 VisualBasicforApplications 中的所有事件。

用 SelectedVBComponent 属性返回活动的部件。活动的部件是工程窗口中追踪的部件。如果在工程窗口中所选择的项目不是部件，则 SelectedVBComponent 返回 Nothing。

注意这个对象模块中的所有对象都具有指向 VBE 对象的 VBE 属性。

属性

ActiveCodePane 属性，ActiveVBProject 属性，ActiveWindow 属性，CodePanes 属性，MainWindow 属性，Name 属性(VBAAdd-In 对象模式)，SelectedVBComponent 属性，Version 属性，COmmandBars 属性，AddIns 属性，Events 属性，VBProjects 属性，Windows 属性，DisplayModel 属性，FullName 属性，LastUsedPath 属性，ReadOnlyMode 属性，TemplatePath 属性

方法

QuitMethod(Add-Ins)

请参阅

CommandBars 集合，CodePanes 集合，Events 对象，VBProjects 集合，Windows 集合，VBE 属性

VBE 属性

返回该 VBE 对象的根，此属性为只读。

应用于

CodeModule 对象,CodePane 对象,CodePanes 集合,LinkedWindows

集合，Property 对象，Properties 集合(VBAAdd-In 对象模式)，Reference 对象，References 集合，VBComponent 对象，VBComponents 集合，VBProject 对象，VBProjects 集合，Window 对象，Windows 集合

说明

所有的对象都有一个指向 VBE 对象根的 VBE 属性。

请参阅

Collection 属性，Parent 属性

示例

下列示例使用 VBE 以及 Name 属性返回活动工程名称。

```
Debug.PrintApplication.VBE.ActiveVBProject.Name
```

VBForm 对象

返回工程中的部件。

语法

VBForm

说明

ClassName 属性决定 VBForm 对象的缺省值。

VBForm 对象取代了 VisualBasic4.0 版中的 ControlTemplate 对象。

特别声明

VBProjects 集合

属性

VBForm 对象，ContainedVBControls 集合，SelectedVBControls 集合，CanPaste 属性，Parent 属性

方法

SelectAll 方法，Paste 属性

请参阅

Form 对象，Forms 集合

VBNewProjects 集合

表示开发环境中的所有新工程。

说明

使用 VBNewProjects 集合访问开发程序实例中的特定工程。VBNewProjects 是一个可以通过使用 For…Each 块迭代的标准集合。

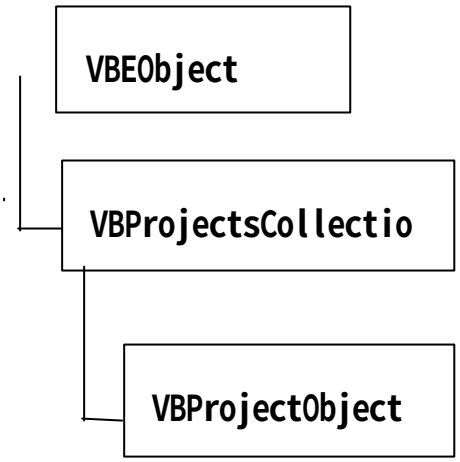
属性

Count 属性，VBE 属性

方法

Item 方法(VBAAdd-In 对象模式)

VBProject 对象



表示一个工程。

说明

可用 VBProject 对象设置工程的属性、访问 VBComponents 集合以及访问 References 集合。

属性

IsDIrty 属性，Collection 属性，Description 属性，HelpContextID 属性(VBAAdd-In 对象模式)，HelpFile 属性(VBAAdd-In 对象模式)，Mode 属性，Name 属性(VBAAdd-In 对象模式)，Protection 属性，Saved 属性，VBE 属性，BuildFileName 属性，Compatible OLEServer 属性，IconState 属性，StartMode 属性，Type 属性，FileName 属性，Name 属性

方法

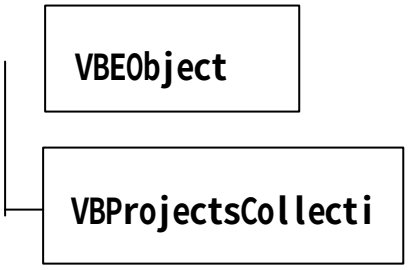
ReadProperty 方法，WriteProperty 方法，AddToolboxProgID

方法，MakeCompiledFile 方法，SaveAs 方法

请参阅

References 集合，VBComponents 集合

VBProjects 集合



表示开发环境中所有打开的工程。

说明

在开发环境的实例中可用 VBProjects 集合访问具体的工程。VBProjects 是一个可用于 ForEach 块的标准集合。

属性

IsDIrty 属性，Collection 属性，Description 属性，HelpContextID 属性(VBAAdd-In 对象模式)，HelpFile 属性(VBAAdd-In 对象模式),Mode 属性,Name 属性(VBAAdd-In 对象模式),Protection 属性，Saved 属性，VBE 属性，BuildFileName 属性，Compatible OLEServer 属性，IconState 属性，StartMode 属性，Type 属性，FileName 属性，Name 属性

方法

ReadProperty 方法，WriteProperty 方法，AddToolboxProgID 方法，MakeCompiledFile 方法，SaveAs 方法

请参阅

References 集合，VBComponents 集合

VBProjects 属性

返回 VBProjects 集合,它表示当前在 VisualBasicIDE 中打开的所有工程。

语法

object.VBProjects

object 所在处是一个对象表达式。

VBProjectsEvents 对象

在 VisualBasic 工程中添加、删除、重新命名或激活控件时将产生事件，该对象就表示这些事件的源。

语法

VBProjectsEvents

事件

ItemActivated 事件，ItemAdded 事件，ItemRemoved 事件，ItemRenamed 事件

请参阅

VBProjectsEvents 属性

VBProjectsEvents 属性

该属性返回类型 VBProjectsEvents 的事件对象。

应用于

Events 对象

语法

object.**VBProjectsEvents**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

说明

这和使用 VBProjects 集合事件是一样的。

Verb 属性

返回或设置一个值，当使用 Action 属性激活对象时，它指定执行的操作。

注意包含 Verb 属性是为了与早期版本的 Action 属性兼容。要获得目前的该功能，可使用 DoVerb 方法。

应用于

OLEContainer 控件

语法

object.**Verb**[=*number*]

Verb 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个指定执行操作的值

说明

每个对象都能支持它自己的谓词集合。使用 ObjectVerbs 和 ObjectVerbsCount 属性存取对象支持的谓词列表。设置 Verb=1 指定列表中的第一个谓词，设置 Verb=2 指定列表中的第二个谓词，以此类推。

将 AutoActivate 设置为 2（双击），当双击对象时，它自动激活。

设置 AutoVerbMenu=True，当使用鼠标右键单击对象时，显示含有对象谓词的弹出式菜单。

请参阅

OLEType 属性，ObjectVerbs 属性，ObjectVerbsCount 属性，AutoActivate 属性，AutoVerbMenu 属性，OLETypeAllowed 属性,DoVerb 方法，Action 属性(OLE 包容器)

Version 属性

返回一个 String，其内容为该应用程序正在使用的 VisualBasic forApplications 的版本号，此属性为只读。

应用于

VBE 对象

说明

Version 属性值是一个由一或两个数值字符开始、一个句号及两个数值字符所组成的字符串；字符串剩余的部分则没有定义，可以包含任何文本或数字。

请参阅

Major 属性，Minor 属性

示例

下列示例使用 Version 属性返回主应用程序的版本号。

```
Debug.PrintApplication.VBE.Version
```

ViewportHeight、ViewportLeft、ViewportTop、ViewportWidth 属性

分别返回视口的当前高度、左边、顶端或者宽度值。

应用于

UserDocument 对象

语法

object.**ViewportHeight**

object.**ViewportLeft**

object.**ViewportTop**

object.**ViewportWidth**

object 所在处代表对象表达式，其值是“应用于”列表中的对象。

返回值类型

Single

说明

用来查看 ActiveX 文档的应用程序将控制视口的尺寸大小。但是，也可用 MinHeight 以及 MinWidth 属性来重新调整 UserDocument 的尺寸。例如，下列代码根据 ViewportHeight 属性和 ViewportWidth 属性，对 PictureBox 控件的尺寸进行重新调整。

```
PrivateSubUserDocument_Resize()
   Picture1.Width=UserDocument.ViewportWidth-_
       Picture1.Left
   Picture1.Height=UserDocument.ViewportHeight-_
       Picture1.Top
EndSub
```

请参阅

MinHeight，MinWidth 属性，SetViewport 方法

Visible 属性

返回或设置一指示对象为可见或隐藏的值。

应用于

ADOData 控件，TreeView 控件，DataRepeater 控件，Column 对象，DataList 控件，RemoteData 控件，Function 控件(数据报表设计程序)，Image 控件(数据报表设计程序)，Label 控件(数

据报表设计程序)，Line 控件(数据报表设计程序)，Section 对象(数据报表设计程序)，Shape 控件(数据报表设计程序)，TextBox 控件(数据报表设计程序)，Extender 对象，Data 对象，CheckBox 控件，ComboBox 控件，CommandButton 控件，DirList BoxControl，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件，Line 控件，ListBox 控件，MDIForm 对象，Menu 控件，OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，OLE 包容器控件

语法

object.**Visible**[=*boolean*]

Visible 属性语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
boolean	布尔表达式指定对象是可见还是隐藏

设置值

boolean 的设置值为：

设置值	描述
True	（缺省值）对象是可见的
False	对象是隐藏的

说明

要在启动时隐藏一个对象，在设计时将 Visible 属性设置为 False。

在代码中设置该属性能够在运行时隐藏然后又重新显示控件以响应某特别事件。

对窗体用 Show 或 Hide 方法，和在代码中将 Visible 属性分别设置为 True 或 False 的效果是一样的。

请参阅

Load 语句，Hide 方法，Show 方法，MultimediaMCI 控件，ButtonVisible

属性(MultimediaMCI 控件)

示例

这个例子使用两个 PictureBox 控件创建动画。要尝试这个例子，请将代码粘贴到窗体的声明部分，这个窗体包含两个和图标大小相等的 PictureBox 控件。将两个 PictureBox 控件的 Name 属性设置为 FileCab 来创建数组，然后按 F5 键并单击图片查看动画。

```
PrivateSubForm_Load()
   DimI '声明变量。
   FileCab(0).BorderStyle=0 '设置 BorderStyle。
   FileCab(1).BorderStyle=0
   '将图标加载到图片框。
   FileCab(1).Picture=LoadPicture("ICONS\OFFICE\FILES03B.ICO")
   FileCab(0).Picture=LoadPicture("ICONS\OFFICE\FILES03A.ICO")
   ForI=0To1
       FileCab(I).Move400,400   '将图片放置到相同的位置。
   NextI
```

```
    FileCab(1).Visible=False '设置为不可视。
    FileCab(0).Visible=True  '设置为可视。
EndSub

PrivateSubFileCab_Click(IndexAsInteger)
    DimI '声明变量。
    ForI=0To1
        '切换两张图片的可视性。
        FileCab(I).Visible=NotFileCab(I).Visible
    NextI
EndSub
```

Visible 属性（VBAAdd-In 对象模式）

对于 Window 对象，返回或设置一个布尔型值，指定该窗口是否可见，此属性可读/写。对 CodePane 对象，返回一个 Boolean 值，指定窗口中的代码窗格是否可见，此属性为只读。

应用于

Window 对象

返回值

Visible 属性返回下列值：

值	描述
True	（默认）对象是可见的
False	对象被隐藏

请参阅

SetFocus 方法，Show 方法(VBAAdd-In 对象模式)，ActiveWindow 属性，
Caption 属性，MainWindow 属性，WindowState 属性

示例

下列示例使用 Visible 属性返回一个 Boolean 值，指示指定的窗口是否可见。

```
Debug.PrintApplication.VBE.Windows(9).Visible
```

VolumeName 属性

设置或返回指定驱动器的卷标名。读/写属性。

应用于

Drive 对象

语法

object.**VolumeName**[=*newname*]

VolumeName 属性有下列几部分：

部分	描述
object	必需的。总是一个 Drive 对象的名字
newname	可选的。如果提供的话，newname 是指定 object 的新名字

说明

下面的代码举例说明了 VolumeName 属性的用法：

```
SubShowVolumeInfo(drvpath)
Dimfs,d,sSetfs=CreateObject("Scripting.FileSystemObject")
Setd=fs.GetDrive(fs.GetDriveName(fs.GetAbsolutePathName(drvpath)
))
s="Drive"&d.DriveLetter&":-"&d.VolumeName
MsgBoxs
EndSub
```

请参阅

AvailableSpace 属性，DriveLetter 属性，DriveType 属性，FileSystem 属性，FreeSpace 属性，IsReady 属性，Path 属性 (FileSystemObject 对象)，RootFolder 属性，SerialNumber 属性，ShareName 属性，TotalSize 属性

WebClass 对象

包含对 HTTP 请求做响应时，要发送给 Web 浏览器的 WebItems (一般为 HTML 文档)。

语法

WebClass

说明

WebClass 对象驻留在 Web 服务器上。使用它截取 HTTP 请求，以便能处理关联的 VisualBasic 代码并返回一个 HTML 文档或其他 WebItems 给浏览器。

属性

Application 属性(WebClass)，BrowserType 属性，Request 属性，Response 属性，Server 属性，Session 属性，NextItem 属性,StateManagement 属性,ErrorPeoperty(WebClass 对象),Name 属性(WebClass)，WebItem)

方法

ReleaseInstance 方法，Trace 方法，URLFor 方法

事件

BeginRequest 事件，EndRequest 事件，FatalErrorResponse 事件，Start 事件

请参阅

第三章的“开发 IIS 应用程序项目”和“IIS 应用程序中的状态管理”,第五章的“开发 IIS 应用程序”，《MicrosoftVisualBasic6.0

部件工具指南》的“构建 Internet 应用程序”

WebClassError 对象

当处理一个 FatalErrorResponse 事件时，该对象指示发生了什么错误。

语法

WebClassError

可以通过 WebClass 对象直接访问该对象，如：

```
Me.Error
```

或更简单一些：

```
Error
```

属性

Error 属性(WebClass 对象)

WebItem 对象

表示一个附加到 WebClass 对象上的 HTML 模板或自定义的 WebItem。WebItem 处理并/或响应一个 HTTP 请求。

语法

WebItem

说明

可以使用以下代码，直接从 WebClass 对象访问 WebItem 对象：

Me.WebItem1

或更简单一些：

WebItem1

HTML 模板是引用了一个 HTML 文件的 WebItem。将作为 HTTP 响应被返回。一个自定义的 WebItem 在其创建 HTTP 响应的 Response 方法中包含了用户书写的代码。

属性

URLData 属性，RescanReplacements 属性，Properties 属性，TagPrefix 属性，Name 属性(WebClass，WebITem)

方法

WriteTemplate 方法

事件

ProcessTag 事件，UserEvent 事件，Respond 事件

WebItemProperties 对象

包含一个与 WebItem 对象相关联的用户定义属性的集合。

语法

WebItemProperties

说明

由您来定义该集合中的属性。WebItemProperties 对象是 WebItem 的一个属性。它是一个变体型的集合，可以用于在 wcRetainInstanceWebClass 中，以每一个 WebItem 为基础组织状态。

属性

Item 属性

方法

Remove 方法(ActiveX 控件)，Clear 方法(ActiveX 控件)

请参阅

Properties 属性

示例

```
DimUserNameAsString
WebItem1.Properties("UserName")="Someone"
```

Weekday 函数

返回一个 Variant(Integer)，包含一个整数，代表某个日期是星期几。

语法

Weekday(*date*,[***firstdayofweek***])

Weekday 函数语法有下列的命名参数：

部分	描述
date	必要。能够表示日期的 Variant、数值表达式、字符串表达式或它们的组合。如果 *date* 包含 Null，则返回 Null
Firstdayofweek	可选。指定一星期第一天的常数。如果未予指定，则以 vbSunday 为缺省值

设置:

firstdayofweek 参数有以下设定值：

常数	值	描述
vbUseSystem	0	使用 NLSAPI 设置
vbSunday	1	星期日（缺省值）
vbMonday	2	星期一
vbTuesday	3	星期二
vbWednesday	4	星期三
vbThursday	5	星期四
vbFriday	6	星期五
vbSaturday	7	星期六

返回值:

Weekday 函数可以返回以下诸值：

常数	值	描述
vbSunday	1	星期日
vbMonday	2	星期一
vbTuesday	3	星期二
vbWednesday	4	星期三
vbThursday	5	星期四
vbFriday	6	星期五
vbSaturday	7	星期六

请参阅

Date 函数，Date 语句，Day 函数，Month 函数，Now 函数，Year 函数，WeekdayName 函数

示例

本示例使用 Weekday 函数将日期转换为星期几。

```
DimMyDate,MyWeekDay
MyDate=#February12,1969#    '指定一日期。
MyWeekDay=Weekday(MyDate)   'MyWeekDay 的值为 4，因为 MyDate 是星期四。
```

WeekdayName 函数

返回一个字符串，表示一星期中的某天。

语法

WeekdayName(*weekday,abbreviate,firstdayofweek*)

WeekdayName **函数语法有如下几部分：**

部分	描述
weekday	**必需的。数字值，表示一星期中的某天。该数字值要依赖于** firstdayofweek **设置中的设置值来决定。**
abbreviate	**可选的。**Boolean **值，表示星期的名称是否被缩写。如果忽略该值，缺省值为** False**，表明星期的名称不能被缩写。**
firstdayofweek	**可选的。数字值，表示一星期中第一天。关于其值，请参阅“设置值”部分。**

设置值

firstdayofweek **参数值如下：**

常数	值	描述
vbUseSystem	0	**使用本国语言支持**(NLS)API **设置值。**
vbSunday	1	**星期日（缺省）。**
vbMonday	2	**星期一**
vbTuesday	3	**星期二**
vbWednesday	4	**星期三**
vbThursday	5	**星期四**
vbFriday	6	**星期五**
vbSaturday	7	**星期六**

请参阅

MonthName 函数

Weight 属性

返回或设置组成 Font 对象的字符的权重。权重指的是字符的宽度，或“粗体因素”。值越大，字符越粗。

应用于

Font 对象

语法

object.**Weight**[=*number*]

Weight 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个用来指定字体权重的数值表达式

说明

Font 对象在设计时不能直接使用。取而代之的是，在“属性”窗口中通过选择控件的 Font 属性并单击“属性”按钮来设置 Weight 属性。可以通过从“字体”对话框的“字形”框中选择一个项来隐含地设置 Weight 属性。正常和斜体设置值的 Weight 值是 400（缺省值），而粗体和斜粗体设置值的 Weight 值是 700。然而在运行时，通过为 Font 对象指定 Weight 属性值可直接设置 Weight。

如果在运行时将Font对象的Weight设置为不同于400或700的值，那么VisualBasic将把这个值转换为400或700，这取决于哪个值与设置的值更接近。精确的范围为：Weight>400并且<551则转换为400；Weight>550则转换为700。

请参阅

Bold属性，FontTransparent属性，Italic属性，Size属性(Font)，StrikeThrough属性，Underline属性，Name属性

示例

这个例子完成下面的功能：每次单击鼠标时在窗体上打印文本。要试用此例，可以先将下面的代码粘贴到一个窗体的声明部分中，然后按下F5键并双击此窗体。

```
PrivateSubForm_Click()
   Font.Bold=NotFont.Bold   '转换Bold。
   Font.StrikeThrough=NotFont.StrikeThrough'转换
StrikeThrough。
   Font.Italic=NotFont.Italic   '转换Italic。
   Font.Underline=NotFont.Underline'转换Underline。
   Font.Size=16'设置Size属性。
   IfFont.BoldThen
       Print"Fontweightis"&Font.Weight&"(bold)."
   Else
       Print"Fontweightis"&Font.Weight&"(notbold)."
   EndIf
```

EndSub

WhatsThisButton 属性

返回或设置一个值，该值决定某个 Form 对象标题栏里是否出现“这是什么”按钮。该值在运行时为只读状态。

应用于

Form 对象，Forms 集合

语法

object.**WhatsThisButton**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

设置值

WhatsThisButton 属性的设置值为：

设置值	描述
True	打开“这是什么”的显示按钮
False	（缺省值）关闭“这是什么”帮助的显示按钮

说明

WhatsThisButton 属性取值为 True 时，WhatsThisHelp 属性必须取值为 True。此外，下列属性也必须如下设置：

ControlBox 属性=True

BorderStyl 属性=FixedSingle 或 Sizable

MinButton 和 MaxButton=False

-或-

BorderStyle 属性=FixedDialog

请参阅

BorderStyle 属性，WhatsThisHelpID 属性，WhatsThisHelp 属性，ShowWhatsThis 方法，WhatsTHisMode 方法，BorderStyle 属性(ActiveX 控件)

WhatsThisHelp 属性

返回或设置一个值，该值决定上下文敏感的帮助是否使用由 Windows95Help 提供的“这是什么”弹出式窗口或主 Help 窗口。该值在运行时为只读状态。

应用于

Form 对象,Forms 集合,MDIForm 对象,Animation 控件,MSFlexGrid 控件

语法

object.**WhatsThisHelp**[=*boolean*]

WhatsThisHelp 属性的语法包括的部分有：

部分	描述
object	一个对象表达式，其值是“应用于”列表中的一个对象
boolean	一个值，该值决定 Help 是否使用“这是什么”弹出式窗口，参见“设置值”中的描述

设置值

boolean 的设置值为：

设置值	描述
True	该应用程序使用一种“这是什么”访问技术来启动 WindowsHelp，并加载一个由 WhatsThisHelpID 属性所标识的主题
False	（缺省值）该应用程序使用 F1 键来启动 WindowsHelp，并加载一个由 HelpContextID 属性所标识的主题

说明

有三种访问技术可用以在应用程序中提供“这是什么”的 Help。采用其中任何一种访问技术时，都必须将 WhatsThisHelp 属性设置为 True。

使用 WhatsThisButton 属性可在窗体的标题栏提供一个“这是什么”按钮。鼠标指针能改变成“这是什么”状态（带问号的箭头）。所显示的主题由用户所单击控件的 WhatsThisHelpID 属性所标识。

调用窗体的 WhatsThisMode 方法。这将导致不使用按钮而像单击“这是什么”按钮时同样的行为。例如，可从应用程序的菜单条的某个菜单命令中来调用该方法。

为某个特定的控件调用 ShowWhatsThis 方法，所显示的主题由该控件的 WhatsThisHelpID 属性来标识。

请参阅

WhatsThisHelpID 属性，EhatsThisButton 属性，ShowWhatsThis

方法，WhatsTHisMode 方法

WhatsThisHelpID 属性

返回或设置一个与对象关联的上下文编号。用以为应用程序提供使用 Windows95Help 中的“这是什么”弹出式窗口的上下文敏感的 Help。

应用于

CheckBox 控件，ComboBox 控件，CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Frame 控件，HscrollBar，VScrollBar 控件，Image 控件，Label 控件，ListBox 控件，OptionButton 控件，PictureBox 控件，TextBox 控件，OLE 包容器控件，ADO 数据控件，TreeView 控件，ImageCombo 控件，ListView 控件，ProgressBar 控件，Slider 控件，StatusBar 控件，TabStrip 控件，Toolbar 控件，DateTimePicker 控件，FlatScrollBar 控件，MonthView 控件 UpDown 控件，CoolBar 控件，DataRepeater 控件，DataList 控件，DBCCombo 控件，DBList 控件，MaskedEdit 控件，MSChart 控件，MSHFlexGrid 控件，SSTab 控件，RemoteData 控件，RichTextBox 控件，Data 控件

语法

object.**WhatsThisHelpID** [=*number*]

WhatsThisHelpID 属性的语法有三个部分：

部分	描述
Object	对象表达式，其值是“应用于”列表中的一个对象
Number	一个指定帮助上下文编号的数值表达式，参见“设置值”中的描述

设置值

number 的设置值为：

设置值	描述
0	（缺省值）无指定的上下文编号
>0	一个整数，该整数给与对象关联的“这是什么”主题指定有效的上下文编号

说明

Windows95 使用窗口右上角处的“这是什么”按钮来启动 WIndowsHelp，并加载一个由 WhatsThisHelpID 属性所标识的主题。

请参阅

HelpContextID 属性，WhatsThisButton 属性，WhatsThisHelp 属性，ShowWhatsTHis 方法，WhatsThisMode 方法

WhatsThisMode 方法

使鼠标指针改变为“这是什么”指针，并且为应用程序准备好在选定的对象上显示“这是什么”Help。

应用于

Form 对象，Forms 集合，MDIFrom 对象

语法

object.**WhatsThisMode**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

说明

执行 WhatsThisMode 方法将把应用程序放置在如同在标题栏中单击“这是什么”按钮时所得的相同状态。鼠标指针要改变为“这是什么”指针。当用户单击一个对象时，被击中的对象的 WhatsThisHelpID 属性用以调用上下文敏感的 Help。在需要从应用程序的菜单条的菜单上调用 Help 时，该方法特别有用。

请参阅

WhatsThisHelpID 属性，WhatsThisButton 属性，WhatsThisHelp 属性，ShowWhatsThis 方法

示例

该例子使用一个菜单命令来将鼠标指针改变为“这是什么”指针，并使上下文敏感 Help 有效。为试用此例，应创建一个菜单，并把这段代码粘贴进一个 Menu 控件的单击事件中。按 F5 键，然后再单击该菜单命令将使该应用程序转变为“这是什么”状态。

```
PrivateSubmnuContextHelp_Click()
   Form1.WhatsThisMode
EndSub
```

While...Wend 语句

只要指定的条件为 True，则会重复执行一系列的语句。

语法

While*condition*

[*statements*]

Wend

While...Wend 语句的语法具有以下几个部分：

部分	描述
condition	必要参数。数值表达式或字符串表达式，其计算结果为 True 或 False。如果 condition 为 Null，则 condition 会视为 False
Statements	可选参数。一条或多条语句，当条件为 True 时执行

说明

如果 condition 为 True，则所有的 statements 都会执行，一直执行到 Wend 语句。然后再回到 While 语句，并再一次检查 condition，如果 condition 还是为 True，则重复执行。如果不为 True，则程序会从 Wend 语句之后的语句继续执行。

While...Wend 循环也可以是多层的嵌套结构。每个 Wend 匹配最近的 While 语句。

提示 Do...Loop 语句提供了一种结构化与适应性更强的方法来执行循环。

请参阅

DO...Loop 语句，With 语句

示例

本示例使用 While...Wend 语句来增加计数变量的值。如果条件判断值为 True，则循环内的语句将一直执行下去。

```
DimCounter
Counter=0                    '设置变量初值。
WhileCounter<20              '测试计数器的值。
   Counter=Counter+1         '将计数器的值加一。
Wend                         '当 Counter>19 时循环终止。
Debug.PrintCounter           '在“立即”窗口中显示数字 20。
```

Width#语句

将一个输出行的宽度指定给用 Open 语句打开的文件。

语法

Width#*filenumber*,*width*

Width#语句的语法具有以下几个部分：

部分	描述
filenumber	必要。任何有效的文件号
width	必要。范围在 0–255 之间的数值表达式，在新的一行开始之前，指出在该行上可出现多少字符。如果 width 等于 0，则行的长度不受限制。width 的缺省值为 0

请参阅

Open 语句，Print 语句#语句

示例

本示例使用 Width#语句来设置文件的输出行宽。

```
DimI
Open"TESTFILE"ForOutputAs#1     '打开输出文件。
Width#1,5  '设置输出行宽为 5。
ForI=0To9  '循环 10 次。
   Print#1,Chr(48+I)；  '每行输出五个字符。
NextI
Close#1                    '关闭文件。
```

Width 属性

返回或设置一个 Single 型数，它以缇(twip)为单位指示该窗口的宽度，此属性可读/写。

应用于

Window 对象

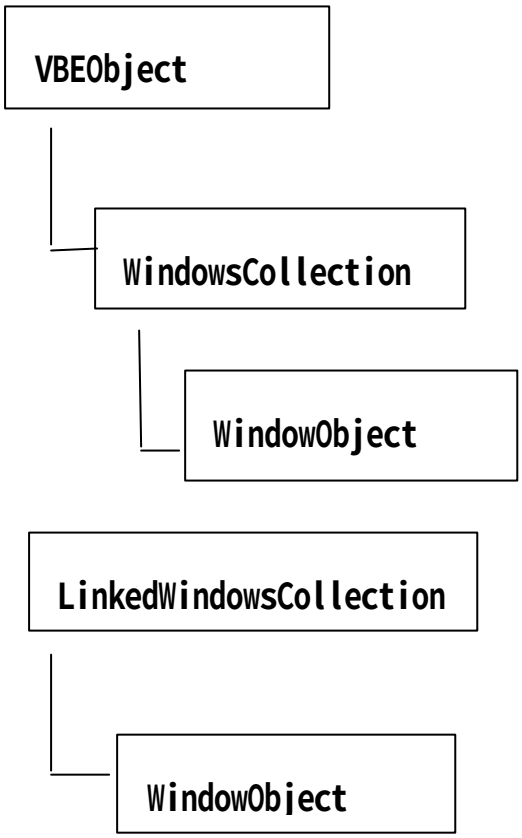
说明

改变一个链接窗口或可连接窗口的 Width 属性，只要该窗口保持链接或连接，就不会有任何效果。

请参阅

Height 属性(VBAAdd-In 对象模式)，Left 属性，Top 属性

Window 对象



表示开发环境中的窗口。

说明

用窗口对象显示、隐含窗口或将窗口定位。

可在 Windows 集合中用 Close 方法关闭窗口。Close 方法对不同类型窗口的影响如下：

窗口	使用 Close 方法的结果
代码窗口	将窗口从 Windows 集合中删除。
设计器	将窗口从 Windows 集合中删除。

链接窗口框架类型的 Window 对象窗口变为分离的无链接窗口。

注意对代码窗口和设计器使用 Close 方法，实际上就是关闭窗

口。把 Visible 属性设置成 False 将会隐含窗口，但不会关闭窗口。对开发环境窗口，比如工程窗口或属性窗口使用 Close 方法，这将等同于把 Visible 属性设置成 False。

可用 SetFocus 方法将焦点移动到窗口。

可用 Visible 属性返回或设置窗口的可见性。

可用 Type 属性来查知正在使用的窗口的类型。如果有多种窗口，则可用 Caption 属性来决定所使用的窗口。也可用 VBComponent 对象的 DesignerWindow 属性或 CodePane 对象的 Window 属性，来查知要使用的窗口。

属性

Caption 属性，Collection 属性，Height 属性(VBAAdd-In 对象模式)，Left 属性，LinkedWindowFrame 属性，LinkedWindows 属性，Top 属性，Type 属性(VBAAdd-In 对象模式)，VBE 属性，Visible 属性，Width 属性，WindowState 属性

方法

Close 方法（VBAAdd-In 对象模式)，SetFocus 方法

特别声明

Windows 集合

请参阅

DesignerWindow 方法，CodePane 对象，VBComponents 对象，Windows

集合，Window 属性

Window 属性

返回显示代码窗格的窗口，此属性为只读。

应用于

CodePane 对象

请参阅

SetFocus 方法，SetSelection 方法，Show 方法（VBAAdd-In 对象模式），CodePaneView 属性，LinkedWindowFrame 属性，TopLine 属性，Visible 属性

示例

下列示例使用 Window 以及 Caption 属性返回指定代码窗格的标题。

```
Debug.PrintApplication.VBE.CodePanes(1).Window.Caption
```

Windowless 属性

返回或设置一个值，它定义在运行时是否为 UserControl 指定一个窗口句柄（hWnd）。

语法

object.**Windowless**[=*boolean*]

Windowless 属性的语法包含如下部分：

部分	描述
object	一个 UserControl 对象
boolean	决定 UserControl 的 hWnd 属性是否返回一个有效的 hWnd

设置值

对 Windowless 的设置值如下：

设置值	描述
True	运行时不对 UserControl 指定 hWnd
False	（缺省的）运行时对 UserControl 指定一个 hWnd

说明

可以使用 Windowless 属性来减少控件的资源占用。按照缺省规定，UserControl 对象的任一实例都指定一个消耗系统资源的窗口句柄（在 hWnd 属性中引用）。如果 Windowless 属性被设置为 True，hWnd 属性将总是返回 0。

通常，除非需要访问要求一个 hWnd 参数的 API 调用或者是将控件作为控件容器，否则可以考虑设置 Windowless 属性为 True。

该属性只能在设计时设置。

注意并非所有的容器都支持 Windowless 控件。如果 WindowlessUserControl 处于一个不支持它的容器内，它将被指定一个 hWnd，并且 Windowless 属性将被忽略。

WindowList 属性

返回或设置一个值，该值用来决定 Menu 对象是否维护 MDIForm 对象中当前 MDI 子窗体的列表。在运行时是只读的。

应用于

Menu 控件

语法

object.**WindowList**

object 所在处代表一个对象表达式，其值是“应用于”列表中的一个对象。

设置值

WindowList 属性的设置值为：

设置值	描述
True	表示 Menu 对象维护一个已打开窗口的列表并显示一个对下一个活动窗口的复选标记。单击窗口名字可以激活那个窗口
False	（缺省值）Menu 不维护已打开窗口的列表

说明

许多多文档接口(MDI)应用程序，例如 MicrosoftExcel 和 MicrosoftWordforWindows，都有一个“窗口”菜单包含一个打开的 MDI 子窗口的列表。该属性能够在应用程序中加入这种机能。

在窗体上只有一个 Menu 对象，其 WindowList 属性能够设置为 True。

当在“菜单编辑器”中为 Menu 对象选择 WindowList 复选框时，正创建的菜单的打开的 MDI 子窗口的列表被显示出来。

请参阅

MDIForm 对象，Caption 属性，Caption 属性(ActiveX 控件)

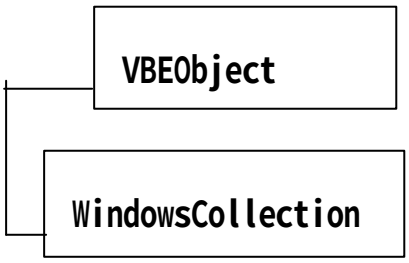
示例

本例创建一些菜单命令，说明 WindowList 菜单的功能，并显示如何能够在多文档接口(MDI)应用程序中加入新的窗体。要试用此例，使用‘工程’菜单上的‘添加 MDI 窗体’命令来创建一个 MDIForm 对象。在 Form1 上，将 MDIChild 属性设置为 True，并创建一个名为“文件”的菜单。为“文件”菜单选择“窗口列表”框。在“文件”菜单上，创建一个“新建”命令，将其 Name 属性设置为“文件菜单”，并将其 Index 属性设置为 0 以建立一个控件数组。将代码粘贴到窗体的声明部分，然后按 F5 键以运行该程序。选择在“文件”菜单上的“新建”命令以建立一个新的 MDI 子窗体。它们的名字被列在“文件”菜单的底部。

```
PrivateSubForm_Load()
   FileMenu(0).Caption="&New"   '在标题中设置访问键。
   LoadFileMenu(1)  '创建新的菜单项。
   FileMenu(1).Caption="-"  '设置分隔符。
   LoadFileMenu(2)  '创建新的菜单项。
   FileMenu(2).Caption="E&xit"  '设置标题及访问键。
EndSub
```

```
PrivateSubFileMenu_Click(IndexAsInteger)
    SelectCaseIndex
        Case0     '选择“新建”命令。
            DimNewFormAsNewForm1'创建 Form1 的副本。
            '装入 NewForm 和设置唯一的标题。
            NewForm.Caption="Untitled"&Forms.Count
        Case2     '选择“退出”命令。
            End  '结束程序。
    EndSelect
EndSub
```

Windows 集合



包含所有打开的窗口或永久窗口。

说明

用 Windows 集合访问 Window 对象。

Windows 集合有固定的窗口集合，在集合中总是可用的，比如在工程窗口、属性窗口中，在一组代表所有打开的代码窗口的窗口

中，在设计器窗口中就都是可用的。打开代码或设计器窗口，就可将一个新成员添加到 Windows 集合。而关闭代码或设计器窗口，就可从 Windows 集合删除一个成员。关闭一个永久开发环境窗口并不会导致从这个集合中删除对应的对象，但会导致此窗口不可见。

属性

Count 属性，Parent 属性，VBE 属性

方法

Item 方法(VBSAdd-In 对象模式

请参阅

CodePane 对象，CodePanes 集合，Window 对象

Windows 属性

返回 Window 对象，它表示 VisualBasicIDE 中的一个窗口。

应用于

VBE 对象

语法

object.**Window**

object 所在处是一个对象表达式，其值是“应用于”列表中的一个对象。

WindowState 属性

返回或设置一个值，该值用来指定在运行时窗体窗口的可视状态。

应用于

Form 对象，Forms 集合，MDIForm 对象

语法

object.**WindowState**[=*value*]

WindowState 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定对象状态的整数，“设置值”中有详细描述

设置值

value 的设置值为：

常数	值	描述
vbNormal	0	（缺省值）正常
vbMinimized	1	最小化（最小化为一个图标）
vbMaximized	2	最大化（扩大到最大尺寸）

说明

在窗体被显示之前，WindowState 属性常常被设置为正常(0)，而不管其初始设置值。这反映在 Height、Left、ScaleHeight、ScaleWidth、

Top 和 Width 属性设置值中。如果窗体在它已被显示后被隐藏，那么这些属性将反映以前的状态直到窗体被再次显示，而不管在此期间对 WindowState 属性所作的任何改变。

请参阅

Load 事件，Paint 事件，Resize 事件

示例

本例在父窗体(Form1)被最小化时隐藏对话框(Form2)，并在父窗体被还原为原来的状态或被最大化时重新显示对话框。要试用此例，将下面的代码粘贴到包含两个窗体的应用程序的 Form1 的声明部分。按 F5 键以启动这个例子。移动 Form1 以使能够看到两个窗体，然后最小化或最大化窗体并观察 Form2 的行为。

```
PrivateSubForm_Load()
   Form2.Show   '显示 Form2。
EndSub

PrivateSubForm_Resize()
   '如果父窗体被最小化...
   IfForm1.WindowState=vbMinimizedThen
'...隐藏 Form2。
      Form2.Visible=False
   '如果父窗体不再是最小化...
Else
```

```
        '...恢复 Form2。
    Form2.Visible=True
      EndIf
    EndSub
```

WindowState 属性(VBAAdd-In 对象模式)

返回或设置一个数值，它指定一个窗口的可视状态，此属性可读/写。

应用于

Window 对象

设置值

WindowState 属性可返回或设置下列值：

常数	值	描述
vbext_ws_Normal	0	（默认）正常
vbext_ws_Minimize	1	缩到最小（缩小至一图标）
vbext_ws_Maximize	2	放到最大（放大至最大的尺寸）

请参阅

SetFocus 方法，ActiveWindow 属性，MainWindow 属性，Visible 属性

示例

下列示例使用 WindowState 属性返回指定窗口的显示状态。返回的值为事先定义好的常量，代表窗口的显示状态。

```
Debug.PrintApplication.VBE.Windows(9).WindowState
```

With 语句

在一个单一对象或一个用户定义类型上执行一系列的语句。

语法

With*object*

[*statements*]

EndWith

With 语句的语法具有以下几个部分：

部分	描述
object	必要参数。一个对象或用户自定义类型的名称
statements	可选参数。要执行在 object 上的一条或多条语句

说明

With 语句可以对某个对象执行一系列的语句，而不用重复指出对象的名称。例如，要改变一个对象的多个属性，可以在 With 控制结构中加上属性的赋值语句，这时候只是引用对象一次而不是在每个属性赋值时都要引用它。下面的例子显示了如何使用 With 语句来给同一个对象的几个属性赋值。

```
WithMyLabel
    .Height=2000
    .Width=2000
```

```
    .Caption="ThisisMyLabel"
EndWith
```

注意当程序一旦进入 With 块，object 就不能改变。因此不能用一个 With 语句来设置多个不同的对象。

可以将一个 With 块放在另一个之中，而产生嵌套的 With 语句。但是，由于外层 With 块成员会在内层的 With 块中被屏蔽住，所以必须在内层的 With 块中，使用完整的对象引用来指出在外层的 With 块中的对象成员。

注意一般来说，建议您不要跳入或跳出 With 块。如果在 With 块中的语句被执行，但是 With 或 EndWith 语句并没有执行，则一个包含对该对象引用的临时变量将保留在内存中，直到您退出该过程。

请参阅

Do...Loop 语句

示例

本示例使用 With 语句对某单一对象执行一系列的语句。MyObject 对象及其属性均为示范目的而采用了通用名称。

```
With MyObject
    .Height=100          '和 MyObject.Height 一样等于 100。
    .Caption="HelloWorld"    ' 和 MyObject.Caption 一样等于
"HelloWorld"。
    With .Font
        .Color=Red       '和 MyObject.Font.Color 一样等于 Red。
```

```
        .Bold=True        '和 MyObject.Font.Bold 一样等于 True。
    EndWith
EndWith
```

WordWrap 属性

返回或设置一个值，该值用来指示一个 AutoSize **属性设置为** True **的** Label **控件，是否要进行水平或垂直展开以适合其** Caption **属性中指定的文本的要求。**

应用于

Label 控件

语法

object.**WordWrap**[=*boolean*]

WordWrap **属性的语法包含下面部分：**

部分	**描述**
Object	**对象表达式，其值是“应用于”列表中的一个对象**
Boolean	**一个用来指定** Label **是否要展开与其文本相适应的布尔表达式，“设置值”中有详细描述**

设置值

boolean **的设置值为：**

设置值	描述
True	文本卷绕；Label 控件垂直展开或缩短，以使其与文本和字体大小相适。水平大小不变
False	（缺省值）文本不卷绕；Label 水平地展开或缩短以使其与文本的长度相适，并且垂直地展开或缩短以使其与字体的大小和文本的行数相适应

说明

为了决定 Label 控件如何显示其内容，应使用此属性。例如，一个动态改变的图形可能具有一个包含的文本也发生改变的 Label。为了使 Label 保持水平方向尺寸不变并允许增加或减少文本，则应将 WordWrap 和 AutoSize 属性设置为 True。

如果希望 Label 控件只水平展开，则应将 WordWrap 设置为 False。

如果不希望 Label 改变大小，应将 AutoSize 设置为 False。

注意如果 AutoSize 被设置为 False，那么文本总是要卷绕，而不管 Label 控件的大小或 WordWrap 属性的设置如何。这可能使某些文本被隐藏，因为 Label 在任何方向上都不能展开。

如果 AutoSize 和 WordWrap 两者都设定为 True，Label 控件的尺寸没有增加文将换行，除非输入的单个字大于 Label 的宽度。在那种情况下，AutoSize 属性选优并增加 Label 的宽度来适应字长。

请参阅

AutoSize 属性，Caption 属性，Caption 属性(ActiveX 控件)

示例

本例将文本放入两个 Label 控件并使用 WordWrap 属性来说明它们不同的行为。要试用此例，将下面的代码粘贴到包含两个 Label 控件的窗体的声明部分，然后按 F5 键并单击窗体来转换 WordWrap 属性的设置值。

```
PrivateSubForm_Load()
   DimAuthor1,Author2,Quote1,Quote2 '声明变量。
   Label1.AutoSize=True'设置“自动调整大小”。
   Label2.AutoSize=True
   Label1.WordWrap=True     '设置“自动换行”。
   Quote1="Icouldn'twaitforsuccess,solwentonwithoutit."
   Author1="-JonathanWinters"
   Quote2="Logicisasystemwherebyonemaygowrongwithconfidence."
   Author2="-CharlesKettering"
   Label1.Caption=Quote1&Chr(10)&Author1
   Label2.Caption=Quote2&Chr(10)&Author2
EndSub

PrivateSubForm_Click()
   Label1.Width=1440    '将宽度设置为一英寸，以缇来表示。
   Label2.Width=1440
   Label1.WordWrap=NotLabel1.WordWrap'转换“自动换行”属性。
```

```
    Label2.WordWrap=NotLabel2.WordWrap
EndSub
```

Write#语句

将数据写入顺序文件。

语法

Write#*filenumber*,[*outputlist*]

Write#语句的语法具有以下几个部分：

部分	描述
filenumber	必要。任何有效的文件号
outputlist	可选。要写入文件的数值表达式或字符串表达式，用一个或多个逗号将这些表达式分界

说明

通常用 Input#从文件读出 Write#写入的数据。

如果省略 outputlist，并在 filenumber 之后加上一个逗号，则会将一个空白行打印到文件中。多个表达式之间可用空白、分号或逗号隔开。空白和分号等效。

用 Write#将数据写入文件时将遵循几个通用的约定，使得无论什么国别都可用 Input#读出并正确解释数据：

在写入数值数据时总使用句号作为十进制分隔符。

对于 Boolean 类型的数据，或者打印#TRUE#或者打印#FALSE#。

无论在什么地区，都不将 True 和 False 这两个关键字翻译出来。

使用通用的日期格式将 Date 类型的数据写入文件中。当日期或时间的部件丢失或为零时，只将现有部分写入文件中。

如果 outputlist 的数据为 Empty，则不将任何数据写入文件。但对 Null 数据，则要写入#NULL#。

如果 outputlist 数据为 Null 数据，则将#NULL#写入文件中。

对于 Error 类型的数据，输出看起来与#ERROR errorcode#一样。无论在什么地区，都不将关键字 Error 翻译出来。

与 Print#语句不同，当要将数据写入文件时，Write#语句会在项目和用来标记字符串的引号之间插入逗号。没有必要在列表中键入明确的分界符。Write#语句在将 outputlist 中的最后一个字符写入文件后会插入一个新行字符，即回车换行符(Chr(13)+Chr(10))。

请参阅

Input#语句，Open 语句，Print#语句

示例

本示例使用 Write#语句将行数据写入顺序文件。

```
Open"TESTFILE"ForOutputAs#1    '打开输出文件。
Write#1,"HelloWorld",234       '写入以逗号隔开的数据。
Write#1,                       '写入空白行。

DimMyBool,MyDate,MyNull,MyError
'赋值 Boolean、Date、Null 及 Error 等。
MyBool=False:MyDate=#February12,1969#:MyNull=Null
```

```
MyError=CVErr(32767)
'Boolean 数据以#TRUE#或#FALSE#的格式写入。
'日期以通用日期格式写入，例如：#1994-07-13#代表
'1994 年 1 月 13 日。Null 数据以#NULL#格式写入。
'Error 数据以#ERROR 错误代号#的格式写入。
Write#1,MyBool ;"isaBooleanvalue"
Write#1,MyDate ;"isadate"
Write#1,MyNull ;"isanullvalue"
Write#1,MyError ;"isanerrorvalue"
Close#1    '关闭文件。
```

Write 方法

写一个指定的字符串到一个 TextStream **文件。**

应用于

TextStream 对象

语法

object.**Write**(*string*)

Write **方法的语法有如下几部分：**

部分	描述
object	必需的。始终是一个 TextStream 对象的名字
string	必需的。要写到文件中去的正文

说明

指定的字符串被写入到文件中，在每个字符串之间没有插入空格或字符。使用 WriteLine 方法写入一个换行符或一个以换行符为结尾的字符串。

请参阅

Close 方法，Read 方法，ReadAll 方法，ReadLine 方法，Skip 方法，SkipLine 方法，WriteBlankLines 方法，WriteLine 方法

WriteBlankLines 方法

写入指定数量的换行符到一个 TextStream 文件中。

应用于

TextStream 对象

语法

object.**WriteBlankLines**(*lines*)

WriteBlankLines 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 TextStream 对象的名字
lines	必需的。要写入的换行符数量

请参阅

Close 方法，Read 方法，ReadAll 方法，ReadLine 方法，Skip 方法，SkipLine 方法，Write 方法，WriteLine 方法

WriteLine 方法

写入一个指定的字符串和换行符到一个 TextStream 文件中。

应用于

TextStream 对象

语法

object.**WriteLine**([*string*])

WriteLine 方法语法有如下几部分：

部分	描述
object	必需的。始终是一个 TextStream 对象的名字
string	可选的。要写入文件的正文。如果省略，一个换行符被写入文件中

请参阅

Close 方法，Read 方法，ReadAll 方法，ReadLine 方法，Skip 方法，SkipLine 方法，Write 方法，WriteBlankLines 方法

WriteProperties 事件

当保存对象的实例时，发生该事件。该事件通知对象此时需要保存对象的状态，以便将来可恢复该状态。大多数情况下，对象的状态仅包括属性值。

应用于

UserControl 对象，UserDocument 对象，Class

语法

Sub*object***t_WriteProperties**(*pb***AsPropertyBag**)

WriteProperties 事件的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
Pb	要写入数据的类型为 PropertyBag 类的对象

说明

WriteProperties 事件发生时，object 的创建者为每个要保存的数据调用 PropertyBag 对象的 WriteProperty 方法，来保存 object 的状态。

注意 pb 属性包可能与传递给最近的 ReadProperties 事件的 pb 不同。

在 object 实例的生存期内，WriteProperties 事件可能发生多次。

请参阅

ReadProperties 事件

WriteProperty 方法

将要保存的数值写入 PropertyBag 类对象。

应用于

PropertyBag 对象

语法

object.**WriteProperty**(*DataName*,*Value*[,*DefaultValue*])

WriteProperty 方法的语法包含下面部分：

部分	描述
object	对象表达式，其值为“应用于”列表中的对象
dataName	字符串表达式，它代表要放入属性包的数据值
value	要保存到属性包中的数据值
defaultValue	数据的缺省值

说明

WriteProperty 方法将在属性包中写入数据值，并将此数据值与 DataName 中的字符串值关联。当调用 ReadProperty 方法从属性包中检索保存的数据值时，将使用这个字符串值来访问数据值。

注意指定缺省值能够减少属于控件容器的文件大小。只有在要写入的值与缺省值不同时，才将属性行写入文件。在可能的情况下，初始化、保存和检索属性值时都应为控件的属性指定缺省值。

请参阅

ReadProperty 方法

WriteTemplate 方法

处理附加到 WebItem 对象上的一个 HTML 模板并将其返回到浏览器。

应用于

WebItem 对象语法

object.WriteTemplate(*Template*AsVariant)

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
template	可选的。指出要返回到浏览器的模板

object 所在处表示对象表达式，其值是“应用于”列表中的一个对象。

说明

使用该方法将 HTML 模板发送给客户。在模板的处理中，如果模板包含了替换令牌，则 ReplaceToken 事件被激发。模板被写到 Response 对象中。

WriteTemplate 方法不会激发关联的 WebItem 的 Respond 事件。建议最好只从 WebItem 的 Respond 事件内部调用 WriteTemplate 方法，并且通过设置 NextItem 属性来触发 Respond 事件。

请参阅

第三章的“发送 HTML 到浏览器执行文本”，“在 WebClass 中置换”和“IIS 应用程序中的状态管理”，“开发 IIS 应用程序”，

《MicrosoftVisualBasic6.0 部件工具指南》中“构建 Internet 应用程序”。

X1、Y1、X2、Y2 属性

返回或设置 Line 控件的起始点(X1,Y1)和终止点(X2,Y2)的坐标。水平坐标是 X1 和 X2；垂直坐标是 Y1 和 Y2。

应用于

Line 控件

语法

object.**X1**[=*value*]

object.**Y1**[=*value*]

object.**X2**[=*value*]

object.**Y2**[=*value*]

X1、Y1、X2 和 Y2 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
value	一个用来指定坐标的数值表达式

说明

在运行时为了动态地使 Line 控件从一个点扩展到另一个点，应使用这些属性。例如，你能够显示一个列表中的项与另一个列表中的项之间的关系，或将地图上的一些点连接起来。

请参阅

Left，Top 属性

示例

本例在单击窗体时显示一条生动的线条在窗体中慢慢下降。要试用此例，将下面的代码粘贴到包含一个 Timer 控件和一个 Line 控件的窗体的声明部分，然后按 F5 键并单击窗体。

```
PrivateSubForm_Load()
   Timer1.Interval=100 '设置计时器时间间隔。
   '将线定位在左上角附近。
'设置 Line1 的属性。
   WithLine1
.X1=100
   .Y1=100
   .X2=500
   .Y2=300
EndWith
Timer1.Enabled=False
EndSub

PrivateSubForm_Click()
   Timer1.Enabled=True '启动计时器。
EndSub
```

```
PrivateSubTimer1_Timer()
   StaticOdd    '声明变量。
   IfOddThen
       Line1.X2=Line1.X2+250
       Line1.Y2=Line1.Y2+600
   Else
       Line1.X1=Line1.X1+250
       Line1.Y1=Line1.Y1+600
   EndIf
   Odd=NotOdd   '转换值
   '如果线超出窗体，那么将导致结束。
   IfLine1.Y1>ScaleHeightThen
       Timer1.Enabled=False'等待另一次单击。
   WithLine1
.X1=100
   .Y1=100
   .X2=500
   .Y2=300
EndWith
       Odd=False
   EndIf
EndSub
```

Year 函数

返回 Variant(Integer)，包含表示年份的整数。

语法

Year(*date*)

必要的 date 参数，可以是任何能够表示日期的 Variant、数值表达式、字符串表达式或它们的组合。如果 date 包含 Null，则返回 Null。

请参阅

Date 函数，Date 语句，Day 函数，Month 函数，Now 函数，Weekday 函数

示例

本示例使用 Year 函数返回某个日期的年份。在开发环境中，日期原义会根据系统的地区设置，以短式日期格式显示。

```
DimMyDate,MyYear
MyDate=#February12,1969#    '指定一日期。
MyYear=Year(MyDate)      'MyYear 的值为 1969。
```

Zoom 属性

返回或设置用来扩大或缩小打印输出比例的百分比。它在设计时是不可用的。

应用于

Printer 对象，Printers 集合

语法

object.**Zoom**[=*number*]

Zoom 属性的语法包含下面部分：

部分	描述
object	对象表达式，其值是“应用于”列表中的一个对象
number	一个用来计算放大或缩小打印输出比例的百分比的数值表达式。缺省值是 0，它指定打印页按照其正常尺寸进行显示

说明

Zoom 属性设置将用因子 Zoom/100 对物理页的尺寸进行放大或缩小，从而成为打印输出的外观上的尺寸。例如，一个信件大小的页以 Zoom 设定为 50 进行打印，能够包含 17x22 英寸大小的页相同的数据，因为打印出的文本和图形被缩小为它们原来的高和宽的一半。

注意 Printer 对象属性的影响确定于打印机厂家所提供的驱动程序。某些属性设置可能不发生影响，或者一些不同的属性设置可能具有相同的影响。超出可接受范围的设置可能会或者不会产生错误。要获得更多的信息，请参阅厂家提供的具体的驱动程序的文档。

请参阅

Printer 对象，Printers 集合

ZOrder 方法

将指定的 MDIForm，Form 或控件放置在其图层的 z-顺序的前端或后端。不支持命名参数。

应用于

CheckBox 控件，ComboBox 控件 CommandButton 控件，DirListBox 控件，DriveListBox 控件，FileListBox 控件，Form 对象，Forms 集合，Frame 控件，HScrollBar，VScrollBar 控件，Image 控件，Label 控件,Line 控件,ListBox 控件,MDIForm 对象,OptionButton 控件，PictureBox 控件，Shape 控件，TextBox 控件，OLE 包容器控件，ADOData 控件,TreeView 控，ImageCombo 控件,ListView 控件,ProgressBar 控件,Slider 控件,StatusBar 控件,TabStrip 控件,ToolBar 控件，DateTimePicker 控件，Animation 控件，FlatScrollBar 控件，MonthView 控件，UpDown 控件，DateReoeater 控件，DataList 控件，DBCombo 控件，DBList 控件，TextBox 控件，MaskedEdit 控件，MSHFlexGrid 控件，MSFLexGrid 控件，RemoteData 控件，RichTextBox 控件，Data 控件

语法

object.**ZOrder***position*

ZOrder 方法的语法包含下列部分：

部分	描述
object	可选的。一个对象表达式，其值为“应用于”列表中的一个对象。如果省略 object，则具有焦点的 Form 对象缺省为 object
position	可选的。一个整数，它用以指示 object 相对于同一 object 其它实例的位置。如果 position 为 0 或被省略，则 object 定位在 Z-顺序前面。如果 position 为 1，则 object 定位在 Z-顺序后面

说明

在设计时选择“编辑”菜单中的“置前”或“置后”菜单命令，可以设置对象的 Z-顺序。

在 MDIForm 对象内，ZOrder 根据 position 的数值决定将 MDI 子窗体放置到 MDI 客户区的前面或后面。对于 MDIForm 或 Form 对象，Zorder 根据 position 的数值决定将窗体放置到屏幕的前面或后面。因此，窗体可以显示在其它运行中的应用程序的前面或后面。

有三个图层与窗体和容器相关联。后层是显示图形方法结果的绘图空间。下一个是中层，用来显示图形对象和 Label 控件。前层显示所有非图形控件，如 CommandButton、CheckBox 或 ListBox。靠近前层的层中包含的东西将覆盖该层后面的各层包含的东西。ZOrder 只对该对象在其中显示的那一层内的各个对象进行重排。

请参阅

附录 AANSI 字符集

字符集(0–127)

0		32	[space]	64	@	96	`
1	•	33	!	65	A	97	a
2	•	34	"	66	B	98	b
3	•	35	#	67	C	99	c
4	•	36	$	68	D	100	d
5	•	37	%	69	E	101	e
6	•	38	&	70	F	102	f
7	•	39	'	71	G	103	g
8	**	40	(	72	H	104	h
9	**	41	)	73	I	105	i
10	**	42	*	74	J	106	j
11	•	43	+	75	K	107	k
12	•	44	,	76	L	108	l
13	**	45	-	77	M	109	m
14	•	46	.	78	N	110	n

15	·	47	/	79	O	111	o
16	·	48	0	80	P	112	p
17	·	49	1	81	Q	113	q
18	·	50	2	82	R	114	r
19	·	51	3	83	S	115	s
20	·	52	4	84	T	116	t
21	·	53	5	85	U	117	u
22	·	54	6	86	V	118	v
23	·	55	7	87	W	119	w
24	·	56	8	88	X	120	x
25	·	57	9	89	Y	121	y
26		·	58	:	90	Z	122 z
27	·	59	;	91	[	123	{
28	·	60	<	92	\	124	|
29	·	61	=	93	]	125	}
30	·	62	>	94	^	126	~
31	·	63	?	95	_	127	·

*·MicrosoftWindows **不支持这些字符。**

***值** 8、9、10 **和** 13 **分别转换为退格、制表、换行和回车字符。它们并没有特定的图形显示，但会依不同的应用程序，而对文本显示有不同的影响。**

请参阅

Chr 函数，字符集(128-255)

字符集(128–255)

128	•	160	[space]	192	À	224	à
129	•	161	¡	193	Á	225	á
130	•	162	¢	194	Â	226	â
131	•	163	£	195	Ã	227	ã
132	•	164	¤	196	Ä	228	ä
133	•	165	¥	197	Å	229	å
134	•	166	¦	198	Æ	230	æ
135	•	167	§	199	Ç	231	ç
136	•	168	¨	200	È	232	è
137	•	169	©	201	É	233	é
138	•	170	ª	202	Ê	234	ê
139	•	171	«	203	Ë	235	ë
140	•	172	¬	204	Ì	236	ì
141	•	173	-	205	Í	237	í
142	•	174	®	206	Î	238	î
143	•	175	¯	207	Ï	239	ï
144	•	176	°	208	Đ	240	ð
145	’	177	±	209	Ñ	241	ñ
146	’	178	²	210	Ò	242	ò
147	•	179	³	211	Ó	243	ó

148	•	180	´	212	Ô	244	ô
149	•	181	µ	213	Õ	245	õ
150	•	182	¶	214	Ö	246	ö
151	•	183	·	215	×	247	÷
152	•	184	¸	216	Ø	248	ø
153	•	185	¹	217	Ù	249	ù
154	•	186	º	218	Ú	250	ú
155	•	187	»	219	Û	251	û
156	•	188	¼	220	Ü	252	ü
157	•	189	½	221	Ý	253	ý
158	•	190	¾	222	Þ	254	þ
159	•	191	¿	223	ß	255	ÿ

• MicrosoftWindows **并不支持这些字符。**

本表中的值是 Windows **的默认值。** ANSI **特征集中** 127 **以上的值是由与操作系统有关的代码页决定的。**

请参阅

字符集(0-127)，Chr 函数

附录 B 数据类型

数据类型概述

以下表格显示所支持的数据类型以及存储空间大小与范围。

数据类型	存储空间大小	范围
Byte	1 个字节	0 到 255
Boolean	2 个字节	True 或 False
Integer	2 个字节	-32,768 到 32,767
Long (长整型)	4 个字节	-2,147,483,648 到 2,147,483,647
Single (单精度浮点型)	4 个字节	负数时从 -3.402823E38 到 -1.401298E-45；正数时从 1.401298E-45 到 3.402823E38
Double (双精度浮点型)	8 个字节	负数时从 -1.79769313486232E308 到 -4.94065645841247E-324；正数时从 4.94065645841247E-324 到 1.79769313486232E308
Currency (变比整型)	8 个字节	从 -922,337,203,685,477.5808 到 922,337,203,685,477.5807

Decimal	14 个字节	没有小数点时为+/-79,228,162,514,264,337,593,543,950,335，而小数点右边有 28 位数时为+/-7.9228162514264337593543950335；最小的非零值为+/-.0000000000000000000000001
Date	8 个字节	100 年 1 月 1 日到 9999 年 12 月 31 日
Object	4 个字节	任何 Object 引用
String (变长)	10 字节加字符串长度	0 到大约 20 亿
String (定长)	字符串长度	1 到大约 65,400
Variant (数字)	16 个字节	任何数字值，最大可达 Double 的范围
Variant (字符)	22 个字节加字符串长度	与变长 String 有相同的范围
用户自定义（利用 Type）	所有元素所需数目	每个元素的范围与它本身的数据类型的范围相同

注意任何数据类型的数组都需要 20 个字节的内存空间，加上每一数组维数占 4 个字节，再加上数据本身所占用的空间。数据所占用的内存空间可以用数据元数目乘上每个元素的大小加以计

算。例如，以 4 个 2 字节之 Integer 数据元所组成的一维数组中的数据，占 8 个字节。这 8 个字节加上额外的 24 个字节，使得这个数组所需总内存空间为 32 个字节。

包含一数组的 Variant 比单独的一个数组需要多 12 个字节。

请参阅

类型转换函数，Boolean 数据类型，Byte 数据类型，Currency 数据类型，Date 数据类型，Decimal 数据类型，Double 数据类型，Integer 数据类型，Long 数据类型，对象数据类型，Single 数据类型，String 数据类型，User-Defined 数据类型，Variant 数据类型，Deftype 语句，Type 语句，Int 函数

Boolean 数据类型

Boolean 变量存储为 16 位 (2 个字节) 的数值形式，但只能是 True 或是 False。Boolean 变量的值显示为 True 或 False （在使用 Print 的时候），或者#TRUE#或#FALSE# （在使用 Write#的时候）。使用关键字 True 与 False 可将 Boolean 变量赋值为这两个状态中的一个。

当转换其他的数值类型为 Boolean 值时，0 会转成 False，而其他的值则变成 True。当转换 Boolean 值为其他的数据类型时，False 成为 0，而 True 成为-1。

请参阅

类型转换函数，数据类型概述，Integer 数据类型，Deftype 语句，Dim 语句

Byte 数据类型

Byte 变量存储为单精度型、无符号整型、8 位（1 个字节）的数值形式，范围在 0 至 255 之间。

Byte 数据类型在存储二进制数据时很有用。

请参阅

类型转换函数，数据类型概述，Integer 数据类型，Deftype 语句

Currency 数据类型

Currency 变量存储为 64 位（8 个字节）整型的数值形式，然后除以 10,000 给出一个定点数，其小数点左边有 15 位数字，右边有 4 位数字。这种表示法的范围可以从-922,337,203,685,477.5808 到 922,337,203,685,477.5807。Currency 的类型声明字符为 at 号(@)。

Currency 数据类型在货币计算与定点计算中很有用，在这种场合精度特别重要。

请参阅

类型转换函数，数据类型概述，Long 数据类型，Deftype 语句

Date 数据类型

Date 变量存储为 IEEE64 位（8 个字节）浮点数值形式，其可以表示的日期范围从 100 年 1 月 1 日到 9999 年 12 月 31 日，而时间可以从 0:00:00 到 23:59:59。任何可辨认的文本日期都可以赋值给 Date 变量。日期文字须以数字符号(#)扩起来，例如，#January1,1993#或#1Jan93#。

Date 变量会根据计算机中的短日期格式来显示。时间则根据计算机的时间格式（12 或 24 小时制）来显示。

当其他的数值类型要转换为 Date 型时，小数点左边的值表示日期信息，而小数点右边的值则表示时间。午夜为 0 而中午为 0.5。负整数表示 1899 年 12 月 30 日之前的日期。

请参阅

类型转换函数，数据类型概述，Double 数据类型，Variant 数据类型，Deftype 语句

Decimal 数据类型

Decimal 变量存储为 96 位（12 个字节）无符号的整型形式，并除以一个 10 的幂数。这个变比因子决定了小数点右面的数字位数，其范围从 0 到 28。变比因子为 0（没有小数位）的情形下，最大的可能值为+/-79,228,162,514,264,337,593,543,950,335。

而在有 28 个小数位的情况下，最大值为+/-7.9228162514264337593543950335，而最小的非零值为+/-0.0000000000000000000000000001。

注意此时，Decimal 数据类型只能在 Variant 中使用，也就是说，不能声明一个变量为 Decimal 的类型。不过可用 CDec 函数，创建一个子类型为 Decimal 的 Variant。

请参阅

类型转换函数，数据类型概述，Deftype 语句

Double 数据类型

Double（双精度浮点型）变量存储为 IEEE64 位（8 个字节）浮点数值的形式，它的范围在负数的时候是从-1.79769313486232E308 到-4.94065645841247E-324，而正数的时候是从 4.94065645841247E-324 到 1.79769313486232E308。Double 的类型声明字符是数字符号(#)。

请参阅

类型转换函数，数据类型概述，Single 数据类型，Deftype 语句

Integer 数据类型

Integer 变量存储为 16 位（2 个字节）的数值形式，其范围为-32,768 到 32,767 之间。Integer 的类型声明字符是百分比符号

(%)。

也可以用 Integer 变量来表示枚举值。枚举值可包含一个有限集合，该集合包含的元素都是唯一的整数，每一个整数都在它使用时的上下文当中有其特殊意义。枚举值为在已知数量的选项中做出选择提供了一种方便的方法，例如，black=0，white=1 等等。较好的编程作法是使用 Const 语句将每个枚举值定义成常数。

请参阅

类型转换函数，数据类型概述，Long 数据类型，Variant 数据类型，Deftype 语句

Long 数据类型

Long（长整型）变量存储为 32 位（4 个字节）有符号的数值形式，其范围从-2,147,483,648 到 2,147,483,647。Long 的类型声明字符为和号(&)。

请参阅

类型转换函数，数据类型概述，Integer 类型，Deftype 语句

Object 数据类型

Object 变量存储为 32 位（4 个字节）的地址形式，其为对象的引用。利用 Set 语句，声明为 Object 的变量可以赋值为任何对象的引用。

注意虽然以 Object 类型声明的变量足以适应包含对各种对象的引用，但是绑定到变量引用的对象总是在晚期（运行时）绑定。要强迫在早期（编译时间）绑定的话，须将对象的引用赋值给用特定类名称声明的变量。

请参阅

类型转换函数，数据类型概述，Variant 数据类型，Deftype 语句，Is 对象函数

Single 数据类型

Single（单精度浮点型）变量存储为 IEEE32 位（4 个字节）浮点数值的形式，它的范围在负数的时候是从-3.402823E38 到-1.401298E-45，而在正数的时候是从 1.401298E-45 到 3.402823E38。Single 的类型声明字符为感叹号(!)。

请参阅

类型转换函数，数据类型概述，Double 数据类型，Variant 数据类型，Deftype 语句

String 数据类型

字符串有两种：变长与定长的字符串。
变长字符串最多可包含大约 20 亿(2^31)个字符。
定长字符串可包含 1 到大约 64K(2^16)个字符。

注意 Public 定长字符串不能在类模块中使用。

String 之字符码的范围是 0 到 255。字符集的前 128 个字符（0 到 127）对应于标准的 U.S.键盘上的字符与符号。这前 128 个字符与 ASCII 字符集中所定义的相同。后 128 个字符（128 到 255）则代表特殊字符，例如国际字符，重音符号，货币符号及分数。

String 的类型声明字符为美元号($)。

请参阅

类型转换函数，数据类型概述，Variant 数据类型，Deftype 语句，String 函数

用户定义数据类型

可以是任何用 Type 语句定义的数据类型。用户自定义类型可包含一个或多个某种数据类型的数据元素、数组或一个先前定义的用户自定义类型。例如：

```
TypeMyType
MyNameAsString '定义字符串变量存储一个名字。
MyBirthDateAsDate  '定义日期变量存储一个生日。
MySexAsInteger '定义整型变量存储性别
EndType  ' (0 为女，1 为男)
```

请参阅

Data 类型 Summary，类型语句

Variant 数据类型

Variant 数据类型是所有没被显式声明（用如 Dim、Private、Public 或 Static 等语句）为其他类型变量的数据类型。Variant 数据类型并没有类型声明字符。

Variant 是一种特殊的数据类型，除了定长 String 数据及用户定义类型外，可以包含任何种类的数据。Variant 也可以包含 Empty、Error、Nothing 及 Null 等特殊值。可以用 VarType 函数或 TypeName 函数来决定如何处理 Variant 中的数据。

数值数据可以是任何整型或实型数，负数时范围从-1.797693134862315E308 到-4.94066E-324，正数时则从 4.94066E-324 到 1.797693134862315E308。通常，数值 Variant 数据保持为其 Variant 中原来的数据类型。例如，如果把一个 Integer 赋值给 Variant，则接下来的运算会把此 Variant 当成 Integer 来处理。然而，如果算术运数针对含 Byte、Integer、Long 或 Single 之一的 Variant 执行，并当结果超过原来数据类型的正常范围时，则在 Variant 中的结果会提升到较大的数据类型。如 Byte 则提升到 Integer，Integer 则提升到 Long，而 Long 和 Single 则提升为 Double。当 Variant 变量中有 Currency、Decimal 及 Double 值超过它们各自的范围时，会发生错误。

可以用 Variant 数据类型来替换任何数据类型，这样会更有适应性。如果 Variant 变量的内容是数字，它可以用字符串来表示数字或是用它实际的值来表示，这将由上下文来决定，例如：

```
DimMyVarAsVariant
MyVar=98052
```

在前面的例子中，MyVar 内有一实际值为 98052 的数值。像期望的那样，算术运算子可以对 Variant 变量运算，其中包含数值或能被解释为数值的字符串数据。如果用+运算子来将 MyVar 与其他含有数字的 Variant 或数值类型的变量相加，结果便是一算术和。

Empty 值用来标记尚未初始化（给定初始值）的 Variant 变量。内含 Empty 的 Variant 在数值的上下文中表示 0，如果是用在字符串的上下文中则表示零长度的字符串("")。

不应将 Empty 与 Null 弄混。Null 是表示 Variant 变量确实含有一个无效数据。

在 Variant 中，Error 是用来指示在过程中出现错误时的特殊值。然而，不像对其他种类的错误那样，程序并不产生普通的应用程序级的错误处理。这可以让程序员或应用程序本身，根据此错误值采取另外的行动。可以用 CVErr 函数将实数转换为错误值来产生 Error 值。

请参阅

CVErr 函数，类型转换函数，数据类型概述，Deftype 语句，Dim 语句，Private 语句，Public 语句，Static 语句，TypeName 函数，Nothing

附录C 运算符

在一个表达式中进行若干操作时，每一部分都会按预先确定的顺序进行计算求解，称这个顺序为运算符的优先顺序。

在表达式中，当运算符不止一种时，要先处理算术运算符，接着处理比较运算符，然后再处理逻辑运算符。所有比较运算符的优先顺序都相同；也就是说，要按它们出现的顺序从左到右进行处理。而算术运算符和逻辑运算符则必须按下列优先顺序进行处理：

算术	比较	逻辑
指数运算(^)	相等(=)	Not
负数(–)	不等(<>)	And
乘法和除法(*、/)	小于(<)	Or
整数除法(\)	大于(>)	Xor
求模运算(Mod)	小于或相等(<=)	Eqv
加法和减法(+、–)	大于或相等(>=)	Imp
字符串连接(&)Is	Like	

当乘法和除法同时出现在表达式中时，每个运算都按照它们从左到右出现的顺序进行计算。当乘法和除法同时出现在表达式中时，每个运算也都按照它们从左到右出现的顺序进行计算。可以

用括号改变优先顺序，强令表达式的某些部分优先运行。括号内的运算总是优先于括号外的运算。但是，在括号之内，运算符的优先顺序不变。

字符串连接运算符(&)不是算术运算符，但是，就其优先顺序而言，它在所有算术运算符之后，而在所有比较运算符之前。

Like 的优先顺序与所有比较运算符都相同，实际上是模式匹配运算符。

Is 运算符是对象引用的比较运算符。它并不将对象或对象的值进行比较，而只确定两个对象引用是否参照了相同的对象。

请参阅

运算符概述，Is 运算符，Like 运算符，=运算符

=操作符

用于对一个变量或属性赋值。

语法

variable=*value*=操作符的语法有如下几部分：

部分	描述
variable	任何变量或任何可写的属性。
value	任何数值型或字符串文字、常数或表达式。

说明

等号左边的名字可以是一个简单的标量变量或一个数组的元素。

等号左边的属性只能是运行时可写的属性。

^运算符

用来求一个数字的某次方，次方数为 exponent 值。

语法

result=number^exponent

^运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
number	必需的；任何数值表达式
exponent	必需的；任何数值表达式

说明

只有当 exponent 为整数值时，number 才可以为负数。在表达式中执行多个指数运算时，^运算符的计算顺序从左到右。

result 的数据类型通常是 Double 或包含 Double 的 Variant。但是，如果 number 或 exponent 中有一个是 Null 表达式，则 result 也是 Null。

示例

本示例使用^运算符来表示某数的乘方的指数值。

```
DimMyValue
MyValue=2^2      '返回 4。
MyValue=3^3^3    '返回 19683。
MyValue=(-5)^3 '返回-125。
```

+运算符

用来求两数之和。

语法

result=expression1+expression2

+运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
expression1	必需的；任何表达式
expression2	必需的；任何表达式

说明

在使用+运算符时有可能无法确定是做加法还是做字符串连接。为避免混淆，请使用&运算符进行连接，并且改进程序代码的可读性。

如果至少有一个表达式不是Variant，则可运用以下法则：

如果	则
两个表达式都是数值数据类型(Byte、Boolean、Integer、Long、Single、Double、Date、Currency 或是 Decimal)	相加
两个表达式都是 String	连接
一个表达式是数值数据类型而另一个是 Null 之外的任意 Variant。	相加
一个表达式是 String 而其它是 Null 之外的任意 Variant。	连接
一个表达式是 EmptyVariant	返回另一个不变的表达式作为 result
一个表达式是数值数据类型，而另一个是 String	产生一个类型不匹配错误
每个表达式都是 Null	result 是 Null

如果两个表达式都是 Variant 表达式，则可运用下列规则：

如果	则
两个 Variant 表达式都是数值	相加
两个 Variant 表达式都是字符串	连接
一个 Variant 表达式是数值而另一个是字符串	相加

对于只有数值数据类型表达式的单纯加法，result 的数据类型通常与其中最精确的表达式的数据类型相同。精确度由最低到最高的顺序是 Byte、

Integer、Long、Single、Double、Currency 和 Decimal。但下列情况例外：

如果	则 result 为
一个 Single 和一个 Long 相加，	一个 Double
result 的数据类型是 Long、Single 或 Date 变体，且越出正确范围，	转换成 Double 变体
result 的数据类型是 Byte 变体，且超过本身的正确范围时，	转换成 Integer 变体
result 的数据类型是 Integer 变体，且超过本身的正确范围时，	转换成 Long 变体
将一个 Date 加到任何数据类型上，	一个 Date

如果有一个或两个表达式是 Null 表达式，则 result 为 Null。如果两个表达式都是 Empty，则 result 是 Integer。但是，如果只有一个表达式是 Empty，则另一个表达式原封不动地作为 result 返回。

注意加法和减法用到的精确度等级与乘法用到的精确度等级不一样。

示例

本示例使用+运算符来计算数值的和。+运算符也可以用来做字符串的串接操作。不过，最好还是使用&运算符来做字符串的串接操作。如果+运算符两边的表达式中混着字符串及数值的话，其结果会是数值的求和。如果都是字符串作“相加”，则返回串接起来的字符串。

```
DimMyNumber,Var1,Var2
```

```
MyNumber=2+2    '返回 4。
MyNumber=4257.04+98112 '返回 102369.04。

Var1="34":Var2=6    '初始化混合变量的值。
MyNumber=Var1+Var2 '返回 40。

Var1="34":Var2="6" '用字符串初始化混合变量的值。
MyNumber=Var1+Var2 '返回"346"（字符串被串接起来）。
```

–运算符

用来求两数之差或表示数值表达式的负值。

语法 1

result=number1-number2

语法 2

-number

–运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
number	必需的；任何数值表达式
number1	必需的；任何数值表达式
number2	必需的；任何数值表达式

说明

在语法 1 中，–运算符是求两数之差的算术减法运算符。在语法 2 中，–运算符为一元负运算符，说明表达式的值为负值。

result 的数据类型通常与最精确的表达式的数据类型相同。精确度由最低到最高的顺序是 Byte、Integer、Long、Single、Double、Currency 和 Decimal。

下列情况例外：

如果	则 result 为
当减法运算中有一个 Single 和一个 Long，	转换成一个 Double。
result 的数据类型是一个 Long、Single 或 Date 变体，且超出正确范围，	转换成包含 Double 的 Variant。
result 的数据类型是一个 Byte 变体，且超出正确范围，	转换成一个 Integer 变体。
result 的数据类型是一个 Integer 变体，且超出正确范围，	转换成一个 Long 变体。
减法运算中有 Date 和其它任何数据类型，	一个 Date。
减法运算中有两个 Date 表达式，	一个 Double。

如果有一个或两个表达式是 Null 表达式，则 result 为 Null。如果一个表达式是 Empty，则作为 0 处理。

注意加法和减法用到的精确度等级与乘法用到的精确度等级不一样。

示例

本示例使用-运算符来计算两数值的差。

```
DimMyResult
MyResult=4-2    '返回 2。
MyResult=459.35-334.90 '返回 124.45。
```

*运算符

用来将两数相乘。

语法

result=*number1* **number2*

*运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
number1	必需的；任何数值表达式
number2	必需的；任何数值表达式

说明

result 的数据类型通常与最精确的表达式的数据类型相同。精确度由最低到最高的顺序是 Byte、Integer、Long、Single、Currency、Double 和 Decimal。下列情况是例外：

如果	则 result 为
乘法运算有一个 Single 和一个 Long，	转换成 Double。
result 的数据类型是 Long、Single 或 Date 变体，且超出正确范围，	转换成有 Double 的 Variant。
result 的数据类型是 Byte 变体，且超出正确范围，	转换成 Integer 变体。
result 的数据类型是 Integer 变体，且超出正确范围，	转换成 Long 变体。

如果有一个或两个表达式是 Null 表达式，则 result 为 Null。如果一个表达式是 Empty，则作为 0 处理。

注意乘法用到的精确度等级与加法和减法用到的精确度等级不一样。

示例

本示例使用*运算符来计算两数的乘积。

```
DimMyValue
MyValue=2*2      '返回 4。
MyValue=459.35*334.90  '返回 153836.315。
```

/运算符

用来进行两个数的除法运算并返回一个浮点数。

语法

result=number1/number2

/运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
number1	必需的；任何数值表达式
number2	必需的；任何数值表达式

说明

result 的数据类型通常是 Double 或 Double 变体。下列情况是例外：

如果	则 result 为
两个表达式都是 Byte、Integer 或 Single 表达式时，	一个 Single，除非超出正确范围；如果发生这种情况，则产生错误
两个表达式都是 Byte、Integer 或 Single 变体，	一个 Single 变体，除非已超出正确范围；一旦发生这种情况，*result* 是一个包含 Double 的 Variant
除法运算中有一个 Decimal 及其它任何数据类型，	一个 Decimal 数据类型

如果有一个或是两个表达式是 Null 表达式，则 result 为 Null。如果一个表达式是 Empty 则作为 0 处理。

示例

本示例使用/运算符来计算浮点数除法。

```
DimMyValue
```

```
MyValue=10/4    '返回 2.5。
MyValue=10/3    '返回 3.333333。
```

\运算符

用来对两个数作除法并返回一个整数。

语法

result=number1\number2

\运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
number1	必需的；任何数值表达式
number2	必需的；任何数值表达式

说明

在进行除法运算之前，数值表达式通过舍去小数部分转换成 Byte、Integer 或 Long 表达式。

通常，无论 result 是不是整数，result 的数据类型都是 Byte、Byte 变体、Integer、Integer 变体、Long 或 Long 变体。任何小数部分都被删除。但是，如果任何一个表达式是 Null，则 result 为 Null。如果一个表达式是 Empty，则作为 0 处理。

示例

本示例使用\运算符来计算整型除法。

```
DimMyValue
MyValue=11\4    '返回 2。
MyValue=9\3     '返回 3。
MyValue=100\3   '返回 33。
```

&运算符

用来强制两个表达式作字符串连接。

语法

result=expression1&expression2

&运算符的语法具有以下几个部分：

部分	描述
result	**必需的；任何** String **或** Variant **变量**
expression1	**必需的；任何表达式**
expression2	**必需的；任何表达式**

说明

如果 expression **不是字符串，则将其转换成** String **变体。如果两个表达式都是字符串表达式，则** result **的数据类型是** String **；否则** result **是** String **变体。如果两个表达式都是 Null，则** result **也是** Null**。但是，只要有一个** expression **是** Null**，那么在与其它表达式连接时，都将其作为长度为零的字符串("")处理。任何 Empty 类型表达式也作为长度为零的字符串处理。**

示例

本示例使用&运算符将字符串串接起来。

```
DimMyStr
MyStr="Hello"&"World"  '返回"HelloWorld"。
MyStr="Check"&123&"Check"  '返回"Check123Check"。
```

比较运算符

用来比较表达式。

语法

result=expression1comparisonoperatorexpression2

result=object1Isobject2

result=stringLikepattern

比较运算符具有以下几个部分：

部分	描述
result	必需的；任何数值变量
expression	必需的；任何表达式
comparisonoperator	必需的；任何比较运算符
object	必需的；任何对象名称
string	必需的；任何字符串表达式
pattern	必需的；任何字符串表达式或字符的范围

说明

下列表格中有一系列比较运算符以及判定 result 是 True、False 还

是 Null 的条件：

运算符	Trueif	Falseif	Nullif
<（小于）	Expression1<expression2	expression1>=expression2	expression1orexpression2=Null
<=（小于或等于）	Expression1<=expression2	expression1>expression2	expression1orexpression2=Null
>（大于）	Expression1>expression2	expression1<=expression2	expression1orexpression2=Null
>=（大于或等于）	Expression1>=expression2	expression1<expression2	expression1orexpression2=Null
=（等于）	Expression1=expression2	expression1<>expression2	expression1orexpression2=Null
<>（不等于）	Expression1<>expression2	expression1=expression2	expression1orexpression2=Null

注意 Is **和** Like **运算符有特定的比较功能，它们不同于表格中的运算符。**

在比较两个表达式的时候可能难以确定将要比较的表达式是数字还是字符串。下列表格将说明如何比较表达式，或说明当其中一个表达式不是 Variant 时，如何获得比较的结果。

如果	则
两个表达式都是数值数据类型(Byte、Boolean、Integer、Long、Single、Double、Date、Currency 或 Decimal)	进行数值比较
两个表达式都是 String	进行字符串比较
一个表达式是数值数据类型而另一个是数字 Variant 或可以为一数字	进行数值比较
一个表达式是数值数据类型而另一个是不能转换成数字的字符串 Variant	产生类型不匹配的错误
一个表达式是 String，而另一个是除了 Null 以外的任何 Variant	进行字符串比较
一个表达式是 Empty 而另一个是数值数据类型	进行数值比较，使用 0 作为 Empty 表达式
一个表达式是 Empty 而另一个是 String	进行字符串比较，使用长度为零的字符串 ("")作为 Empty 表达式

如果 expression1 和 expression2 都是 Variant 表达式，则它们的基本类型决定了比较方式。下列表格说明如何比较表达式，或指出通过比较得到的结果，此结果取决于 Variant 的基本类型：

如果	则
两个 Variant 表达式都是数值	进行数值比较
两个 Variant 表达式都是字符串	进行字符串比较
一个 Variant 表达式是数值而另一个是字符串	数值表达式小于字符串表达式
一个 Variant 表达式是 Empty 而另一个是数值	进行数值比较，使用 0 作为 Empty 表达式
一个 Variant 表达式是 Empty 而另一个是字符串	进行字符串比较，使用长度为零的字符串("")作为 Empty 表达式
两个 Variant 表达式都是 Empty	表达式相等

当一个 Single 与一个 Double 作比较时，Double 会进行舍入处理而与此 Single 有相同的精确度。

如果一个 Currency 与一个 Single 或 Double 进行比较，则 Single 或 Double 转换成一个 Currency。与此相似，当一个 Decimal 要与一个 Single 或 Double 进行比较时，Single 或 Double 会转换成一个 Decimal。对于 Currency，任何小于.0001 的小数将被舍弃，对于 Decimal，任何小于 1E-28 的小数将被舍弃，或者可能产生溢出错误。舍弃这样的小数部分会使原来不相等的两个数值经过比较后相等。

示例

本示例示范各种“比较运算符”的用法。这类运算符通常拿来做表达式结果的比较。

```
DimMyResult,Var1,Var2
MyResult=(45<35)    '返回 False。
MyResult=(45=45)    '返回 True。
MyResult=(4<>3)         '返回 True。
MyResult=("5">"4") '返回 True。

Var1="5":Var2=4         '设置变量初值。
MyResult=(Var1>Var2)    '返回 True。

Var1=5:Var2=Empty
MyResult=(Var1>Var2)    '返回 True。

Var1=0:Var2=Empty
MyResult=(Var1=Var2)    '返回 True。
```

And 运算符

用来对两个表达式进行逻辑连接。

语法

*result=expression1***And***expression2*

And **运算符的语法具有以下几个部分：**

部分	描述
result	必需的；任何数值变量
expression1	必需的；任何表达式
expression2	必需的；任何表达式

说明

如果两个表达式的值都是 True，则 result 是 True。如果其中一个表达式的值是 False，则 result 是 False。下列表格说明如何确定 result：

如果 expression1 为	且 expression2 为	则 result 为
True	True	True
True	False	False
True	Null	Null
False	True	False
False	False	False
False	Null	False
Null	True	Null
Null	False	False
Null	Null	Null

And 运算符还对两个数值表达式中位置相同的位进行逐位比较，并根据下表对 result 中相应的位进行设置：

如果在 expression1 的位为	且在 expression2 中的位为	result 为
0	0	0
0	1	0
1	0	0
1	1	1

示例

本示例使用 And 运算符来做两个表达式结果的逻辑合取（alogicalconjunction）。

```
DimA,B,C,D,MyCheck
A=10:B=8:C=6:D=Null    '设置变量初值。
MyCheck=A>BAndB>C  '返回 True。
MyCheck=B>AAndB>C  '返回 False。
MyCheck=A>BAndB>D  '返回 Null。
MyCheck=AAndB  '返回 8（位比较的结果）。
```

Eqv 运算符

用来对两个表达式进行逻辑等价运算。

语法

*result=expression1***Eqv***expression2*

Eqv **运算符的语法具有以下几个部分：**

部分	描述
result	必需的；任何数值变量
expression1	必需的；任何表达
expression2	必需的；任何表达式

说明

如果有一个表达式是 Null，则 result 也是 Null。如果表达式都不是 Null，则根据下表来确定 result：

如果 expression1 为	且 expression2 为	则 result 为
True	True	True
True	False	False
False	True	False
False	False	True

Eqv 运算符对两个数值表达式中位置相同的位进行逐位比较，并根据下表对 result 中相应的位进行设置：

如果在 expression1 的位为	且在 expression2 中的位为	result 为
0	0	1
0	1	0
1	0	0
1	1	1

示例

本示例使用 Eqv 运算符来做两个表达式结果的逻辑等价（logicalequivalence)。

```
DimA,B,C,D,MyCheck
A=10:B=8:C=6:D=Null    '设置变量初值。
MyCheck=A>BEqvB>C  '返回 True。
MyCheck=B>AEqvB>C  '返回 False。
MyCheck=A>BEqvB>D  '返回 Null。
MyCheck=AEqvB  '返回-3（位比较的结果)。
```

Imp 运算符

用来对两个表达式进行逻辑蕴涵运算。

语法

*result=expression1***Imp***expression2*

Imp 运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
expression1	必需的；任何表达式
expression2	必需的；任何表达式

说明

下列表格说明如何确定 result：

如果 expression1 为	且 expression2 为	则 result 为
True	True	True
True	False	False
True	Null	Null
False	True	True
False	False	True
False	Null	True
Null	True	True
Null	False	Null
Null	Null	Null

Imp 运算符对两个数值表达式中位置相同的位进行逐位比较，并根据下表对 result 中相应的位进行设置：

如果在 expression1 的位为	且在 expression2 中的位为	result 为
0	0	1
0	1	1
1	0	0
1	1	1

示例

本示例使用 Imp 运算符来做两个表达式结果的逻辑隐含式（logicalimplication）。

```
DimA,B,C,D,MyCheck
```

```
A=10:B=8:C=6:D=Null    '设置变量初值。
MyCheck=A>BImpB>C  '返回 True。
MyCheck=A>BImpC>B  '返回 False。
MyCheck=B>AImpC>B  '返回 True。
MyCheck=B>AImpC>D  '返回 True。
MyCheck=C>DImpB>A  '返回 Null。
MyCheck=BImpA  '返回-1（位比较的结果）。
```

Is 运算符

用来比较两个对象的引用变量。

语法

*result=object1***Is***object2*

Is **运算符的语法具有以下几个部分：**

部分	描述
result	**必需的；任何数值变量**
object1	**必需的；任何对象名称**
object2	**必需的；任何对象名称**

说明

如果 object1 **和** object2 **两者引用相同的对象，则** result **为** True **；否则，** result **为** False**。有很多方法使两个变量引用相同的对象。在以下示例中，A 和 B 已被设置成与 C 引用相同的对象：**

SetA=B

在下例中，A 和 B 引用的对象与 C 相同：

SetA=C

SetB=C

示例

本示例使用 Is 运算符来比较两个对象引用。示例中的对象变量名只是作说明用途的一般性名称而已。

```
DimMyObject,YourObject,ThisObject,OtherObject,ThatObject,MyCheck
SetYourObject=MyObject '指定对象引用。
SetThisObject=MyObject
SetThatObject=OtherObject
MyCheck=YourObjectIsThisObject '返回 True。
MyCheck=ThatObjectIsThisObject '返回 False。
'假设 MyObject<>OtherObject
MyCheck=MyObjectIsThatObject   '返回 False。
```

Like 运算符

用来比较两个字符串。

语法

result=*string***Like***pattern*

Like 运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
string	必需的；任何字符串表达式
pattern	必需的；任何字符串表达式，遵循“说明”中的模式匹配约定

说明

如果 string 与 pattern 匹配，则 result 为 True；如果不匹配，则 result 为 False。但是如果 string 或 pattern 中有一个为 Null，则 result 为 Null。

Like 运算符的特性随着 OptionCompare 语句而不同。每个模块的缺省字符串比较方法是 OptionCompareBinary。

在字符串比较中，OptionCompareBinary 的结果是根据字符的内部二进制表示法导出的排序顺序得到的。在 MicrosoftWindows 中，排序顺序由代码页决定。下例说明通常二进制的排序顺序：A<B<E<Z<a<b<e<z<À<Ê<Ø<à<ê<ø 在字符串比较中，OptionCompareText 的结果是字符串比较，它建立在不区分大小写的文本排序顺序基础上，而这一排序顺序是由系统的国别确定的。在对相同的字符使用 OptionCompareText 时会产生以下文本排序顺序：(A=a)<(À=à)<(B=b)<(E=e)<(Ê=ê)<(Z=z)<(Ø=ø) 内建的模式匹配功能提供了多种工具来进行字符串比较。有了模式匹配功能就可以使用通配符、字符串列表或字符区间的任何组合来匹配字符串。下列表格

指出 pattern 中允许的字符以及它们与什么进行匹配：

pattern 中的字符	符合 string 中的
?	任何单一字符
*	零个或多个字符
#	任何一个数字(0–9)
[charlist]	charlist.中的任何单一字符。
[!charlist]	不在 *charlist* 中的任何单一字符

在中括号([])中，可以用由一个或多个字符(charlist)组成的组与 string 中的任一字符进行匹配，这个组几乎包括任何一个字符代码以及数字。

注意为了与左括号([)、问号(?)、数字符号(#)和星号(*)等特殊字符进行匹配，可以将它们用方括号括起来。不能在一个组内使用右括号(])与自身匹配，但在组外可以作为个别字符使用。

通过在范围的上、下限之间用连字符(–)，charlist 可以指定字符的范围。例如，如果 string 中相应字符的位置包括 A–Z 之间的任意大写字母，则[A-Z]得到一个匹配。不需要分界符，方括号内就可以包括多个范围。

指定范围的意义取决于运行时的有效字符排序（正如 OptionCompare 和系统的国别设置所确定的，代码在运行之中）。使用 OptionCompareBinary 示例可以看到，范围[A–E]与 A、B 和 E 相匹配。通过 OptionCompareText 可以看到，[A–E] 与 A,a,_,_,B,b,E,e 相匹配。此范围与_或_不匹配，因为按照排序顺序，重音字符在非重音字符之后。

下面列举的是模式匹配的其它重要规则：

charlist 开头的惊叹号(!)意味着，如果在 string 中找到任何不属于 charlist 的字符，则存在一个匹配。如果在方括号之外使用惊叹号，则惊叹号与自身匹配。

连字符(–)可以出现在 charlist 的开头（如果使用惊叹号，则在惊叹号之后），也可以出现在 charlist 的结尾与自身匹配。在任何其它地方，连字符用来识别字符的范围。

当指定了字符范围时，这些字符必须按照升序（从最小到最大）顺序来显示。[A-Z]是有效模式，但[Z-A]不是。

字符序列[]被看作是长度为零的字符串("")。

某些语言的字母表中有一些特殊字符，它们表示两个分开的字符。例如，在一些语言中，当字符"a"与"e"同时出现时，用"_"来表示字符"a"与"e"。Like 运算符可以辨认出单一的特殊字符和两个个别字符是否相等。

当一个语言使用了一个特殊字符，而且这个字符在系统地区设置中已被指定的时候，在 pattern 或 string 中出现的单一特殊字符将与其他字符串中等价的两个字符相匹配。与此相似，由方括号括起来的 pattern 中的单一特殊字符（字符本身在列表或范围内）将与 string 中等价的两个字符序列匹配。

请参阅

OptionCompare 语句，比较运算符，InStr 函数，StrComp 函数

示例

本示例使用 Like 运算符做字符串的方式比较。

```
DimMyCheck
MyCheck="aBBBa"Like"a*a"    '返回 True。
MyCheck="F"Like"[A-Z]" '返回 True。
MyCheck="F"Like"[!A-Z]"     '返回 False。
MyCheck="a2a"Like"a#a" '返回 True。
MyCheck="aM5b"Like"a[L-P]#[!c-e]"   '返回 True。
MyCheck="BAT123khg"Like"B?T*"   '返回 True。
MyCheck="CAT123khg"Like"B?T*"   '返回 False。
```

Mod 运算符

用来对两个数作除法并且只返回余数。

语法

result=*number1* **Mod** *number2*

Mod 的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
number1	必需的；任何数值表达式
number2	必需的；任何数值表达式

说明

在进行 Mod 运算或求余数运算时，该运算符将 number1 用 number2 除（将浮点数字四舍五入成整数），并把余数作为 result 的值返

回。例如，在下列表达式中，A(result)等于 5。

A=19Mod6.7 一般说来，不管 *result* 是否为一个整数，result 的数据类型为 Byte，Byte 变体、Integer、Integer 变体、Long 或一个包含 Long 的 Variant。任何小数部分都被删除。但是，如果任何一个 Null，类型的表达式出现时，result 都将是 Null。任何 Empty 类型表达式都作为 0 处理。

示例

本示例使用 Mod 运算符来对两数作除法运算，但返回其余数而非商数。如果两数中有一数为浮点数，该数会先被四舍五入成整型后再进行运算。

```
DimMyResult
MyResult=10Mod5     '返回 0。
MyResult=10Mod3     '返回 1。
MyResult=12Mod4.3   '返回 0。
MyResult=12.6Mod5   '返回 3。
```

Not 运算符

用来对表达式进行逻辑否定运算。

语法

result=**Not***expression*

Not 运算符的语法具有以下几个部分：

部分	描述
result	必需的；任何数值变量
expression	必需的；任何表达式

说明

下表说明如何确定 result：

如果 expression 为	则 result 为
True	False
False	True
Null	Null

此外，Not 运算符改变任何变量的位值，并根据下表设置 *result* 中相应的位：

如果在 expression 的位为	则在 result 中的位为
0	1
1	0

示例

本示例使用 Not 运算符来做两个表达式结果的逻辑非（logicalnegation）。

```
DimA,B,C,D,MyCheck
A=10:B=8:C=6:D=Null    '设置变量初值。
MyCheck=Not(A>B)    '返回 False。
MyCheck=Not(B>A)    '返回 True。
```

```
MyCheck=Not(C>D)    '返回 Null。
MyCheck=NotA    '返回-11（位比较的结果）。
```

Or 操作符

用于执行两个表达式上的逻辑分离。

语法

*result=expression1***Or***expression2*

Or 操作符的语法有以下几部分：

部分	描述
result	必要的；任何数字变量
expression1	必要的；任何表达式
expression2	必要的；任何表达式

说明

如果表达式的两者之一或两者都为 True，则 result 为 True。下表表示了 resul 是怎样决定的：

如果 expression1 是	expression2 是	那么 result 是
True	True	True
True	False	True
True	Null	True
False	True	True
False	False	False
False	Null	Null
Null	True	True
Null	False	Null
Null	Null	Null

or 操作符也执行两个数字表达式相同位置的位的比较，并根据下表设定相关的位：

如果 expression1 的位是	expression2 的位是	那么 result 是
0	0	0
0	1	1
1	0	1
1	1	1

示例

本例使用 or 操作符执行两个表达式的逻辑分离：

```
DimA,B,C,D,MyCheck
A=10:B=8:C=6:D=Null    'Initializevariables.
```

```
MyCheck=A>BOrB>C  'ReturnsTrue.
MyCheck=B>AOrB>C  'ReturnsTrue.
MyCheck=A>BOrB>D  'ReturnsTrue.
MyCheck=B>DOrB>A  'ReturnsNull.
MyCheck=AOrB  'Returns10(bitwisecomparison).
```

Xor 运算符

用来对两个表达式进行逻辑互斥或运算。

语法

[*result*=]*expression1***Xor***expression2*

Xor **运算符的语法具有以下几个部分：**

部分	**描述**
result	**可选；任何数值变量**
expression1	**必需的；任何表达式**
expression2	**必需的；任何表达式**

说明

如果表达式中有一个而且只有一个值为 True**，则** result **为** True**。但是，如果表达式中有一个为** Null**，则** result **也为** Null**。当两个表达式都不为** Null**，则根据下表来确定** result**：**

如果 expression1 为	且 expression2 为	则 result 为
True	True	False
True	False	True
False	True	True
False	False	False

Xor 运算符既可作为逻辑运算符，也可作为位运算符。使用互斥或的逻辑进行的两个表达式的逐位比较，其结果通过下表说明：

如果 expression1 为	且 expression2 为	则 result 为
0	0	0
0	1	1
1	0	1
1	1	0

示例

本示例使用 Xor 运算符来做两个表达式结果的逻辑异或（logicalexclusion）。

```
DimA,B,C,D,MyCheck
A=10:B=8:C=6:D=Null    '设置变量初值。
MyCheck=A>BXorB>C  '返回 False。
MyCheck=B>AXorB>C  '返回 True。
MyCheck=B>AXorC>B  '返回 False。
MyCheck=B>DXorA>B  '返回 Null。
MyCheck=AXorB          '返回 2（位比较的结果）。
```

附录D 导出的数学函数

以下为非基本数学函数的列表，皆可由基本数学函数导出：

函数	**由基本函数导出之公式**
Secant **（正割）**	Sec(X)=1/Cos(X)
Cosecant **（余割）**	Cosec(X)=1/Sin(X)
Cotangent **（余切）**	Cotan(X)=1/Tan(X)
InverseSine **（反正弦）**	Arcsin(X)=Atn(X/Sqr(-X*X+1))
InverseCosine **（反余弦）**	Arccos(X)=Atn(-X/Sqr(-X*X+1))+2*Atn(1)
InverseSecant **（反正割）**	Arcsec(X)=Atn(X/Sqr(X*X-1))+Sgn((X)-1)*(2*Atn(1))
InverseCosecant **（反余割）**	Arccosec(X)=Atn(X/Sqr(X*X-1))+(Sgn(X)-1)*(2*Atn(1))
InverseCotangent **（反余切）**	Arccotan(X)=Atn(X)+2*Atn(1)
HyperbolicSine **（双曲正弦）**	HSin(X)=(Exp(X)–Exp(-X))/2
HyperbolicCosine	HCos(X)=(Exp(X)+Exp(-X))/2

（双曲余弦）	
HyperbolicTangent **（双曲正切）**	HTan(X)=(Exp(X)–Exp(-X))/(Exp(X)+Exp(-X))
HyperbolicSecant **（双曲正割）**	HSec(X)=2/(Exp(X)+Exp(-X))
HyperbolicCosecant **（双曲余割）**	HCosec(X)=2/(Exp(X)-Exp(-X))
HyperbolicCotangent **（双曲余切）**	HCotan(X)=(Exp(X)+Exp(-X))/(Exp(X)-Exp(-X))
InverseHyperbolicSine **（反双曲正弦）**	HArcsin(X)=Log(X+Sqr(X*X+1))
InverseHyperbolicCosine **（反双曲余弦）**	HArccos(X)=Log(X+Sqr(X*X-1))
InverseHyperbolicTangent **（反双曲正切）**	HArctan(X)=Log((1+X)/(1-X))/2
InverseHyperbolicSecant **（反双曲正割）**	HArcsec(X)=Log((Sqr(-X*X+1)+1)/X)
InverseHyperbolicCosecant	HArccosec(X)=Log((Sgn(X)*Sqr(X*X+1)+1)/X)
InverseHyperbolicCotangent **（反双曲余切）**	HArccotan(X)=Log((X+1)/(X-1))/2
以 N **为底的对数**	LogN(X)=Log(X)/Log(N)

请参阅

Atn 函数，Cos 函数，Exp 函数，Log 函数，Sin 函数，Sqr 函数，Tan 函数

附录 EAsc 函数

返回一个 Integer，代表字符串中首字母的字符代码。

语法

Asc(*string*)

必要的 string 参数可以是任何有效的字符串表达式。如果 string 中没有包含任何字符，则会产生运行时错误。

说明

在非 DBCS 系统下，返回值范围为 0–255。在 DBCS 系统下，则为 -32768–32767。

注意 AscB 函数作用于包含在字符串中的字节数据，AscB 返回第一个字节的字符代码，而非字符的字符代码。AscW 函数返回 Unicode 字符代码，若平台不支持 Unicode，则与 Asc 函数功能相同。

请参阅

类型转换函数，Chr 函数

示例

本示例使用 Asc 函数返回字符串首字母的字符值（ASCII 值）。

```
DimMyNumber
MyNumber=Asc("A")   '返回 65。
```

```
MyNumber=Asc("a")   '返回 97。
MyNumber=Asc("Apple")   '返回 65。
```

CVErr 函数

返回 Error 子类型的 Variant，其中包含指定的错误号。

语法

CVErr(*errornumber*)

必要的 errornumber 参数可以是任何有效的错误号代码。

说明

可以在过程中，使用 CVErr 函数来创建用户自定义错误。例如，如果创建一个函数，它可以接受若干个参数，且正常返回一个字符串，则可以让函数来判断输入的参数，确认它们是在可接受的范围内。如果不是的话，此函数将不会返回所要的字符串。在这种情况下，CVErr 可以返回一个错误号，并告知应该采取的行动。注意，Error 的隐式转换是不允许的，例如，不能直接把 CVErr 的返回值赋值给一个非 Variant 的变量。然而，可以对 CVErr 的返回值进行显式转换（使用 CInt、CDbl 等等），并赋值给适当的数据类型变量。

请参阅

类型转换函数，IsError 函数，数据类型概述

示例

本示例使用 CVErr 函数返回一 Variant 类型的值，其 VarType 为

vbError(10)。如果传进去的参数不是一个数字，用户自定义函数 CalculateDouble 将会返回错误信息。可用 CVErr 返回来自用户自定义过程的自定义错误，或改变处理运行时错误的方式。IsError 函数可用来测试返回值是否代表错误状态。

```
'调用 CalculateDouble，且传入一个会产生错误的参数。
SubTest()
    Debug.PrintCalculateDouble("345.45robert")
EndSub
'定义 CalculateDouble 函数过程。
FunctionCalculateDouble(Number)
    IfIsNumeric(Number)Then
  CalculateDouble=Number*2     '返回结果。
    Else
    CalculateDouble=CVErr(2001)       '返回一自定义错误码。
  EndIf
   EndFunction
```

Str 函数

返回代表一数值的 Variant(String)。

语法

Str(*number*)

必要的 number 参数为一个 Long，其中可包含任何有效的数值表

达式。

说明

当一数字转成字符串时，总会在前头保留一空位来表示正负。如果 number 为正，返回的字符串包含一前导空格暗示有一正号。

使用 Format 函数可将数值转成必要的格式，如日期、时间、货币或其他用户自定义格式。与 Str 不同的是，Format 函数不包含前导空格来放置 number 的正负号。

注意 Str 函数只视句点(.)为有效的小数点。如果使用不同的小数点（例如，国际性的应用程序），可使用 CStr 将数字转成字符串。

请参阅

Val 函数，类型转换函数，Format 函数

示例

本示例使用 Str 函数来将一个数字转成字符串。当数字转成字符串时，字符串的第一个位一定是空格或是正负号。

```
DimMyString
MyString=Str(459)        '返回"459"。
MyString=Str(-459.65)    '返回"-459.65"。
MyString=Str(459.001)    '返回"459.001"。
```

类型转换函数

每个函数都可以强制将一个表达式转换成某种特定数据类型。

语法

CBool(*expression*)

CByte(*expression*)

CCur(*expression*)

CDate(*expression*)

CDbl(*expression*)

CDec(*expression*)

CInt(*expression*)

CLng(*expression*)

CSng(*expression*)

CStr(*expression*)

CVar(*expression*)

必要的 *expression* **参数可以是任何字符串表达式或数值表达式。**

返回类型

函数名称决定返回类型，如下所示：

函数	**返回类型**	Expression **参数范围**
CBool	Boolean	**任何有效的字符串或数值表达式**
CByte	Byte	0 **至** 255
CCur	Currency	-922,337,203,685,477.5808 **至** 922,337,203,685,477.5807
CDate	Date	**任何有效的日期表达式**

CDbl	Double	负数从-1.79769313486232E308 至-4.94065645841247E-3E-324；正数从4.9406564584124724 至 1.79769313486232E308
CDec	Decimal	零变比数值，即无小数位数值，为+/-79,228,162,514,264,337,593,543,950,335。对于 28 位小数的数值，范围则为+/-7.9228162514264337593543950335；最小的可能非零值是0.0000000000000000000000000001
CInt	Integer	-32,768 至 32,767，小数部分四舍五入
CLng	Long	-2,147,483,648 至 2,147,483,647，小数部分四舍五入
CSng	Single	负数为-3.402823E38 至-1.401298E-45；正数为 1.401298E-45 至 3.402823E38
CStr	String	依据 expression 参数返回 Cstr
CVar	Variant	若为数值，则范围与 Double 相同；若不为数值，则范围与 String 相同

说明

如果传递给函数的 expression 超过转换目标数据类型的范围，将发生错误。

通常，在编码时可以使用数据类型转换函数来体现某些操作的结果应该表示为特定的数据类型，而不是缺省的数据类型。例如，当单精度、双精度或整数运算发生的情况下，使用 CCur 来强制执行货币运算。

应该使用数据类型转换函数来代替 Val，以使国际版的数据转换可以从一种数据类型转换为另一种。例如，当使用 CCur 时，不同的小数点分隔符、千分位分隔符和各种货币选项，依据系统的国别设置都会被妥善识别。

当小数部分恰好为 0.5 时，CInt 和 CLng 函数会将它转换为最接近的偶数值。例如，0.5 转换为 0、1.5 转换为 2。CInt 和 CLng 函数不同于 Fix 和 Int 函数，Fix 和 Int 函数会将小数部分截断而不是四舍五入。并且 Fix 和 Int 函数总是返回与传入的数据类型相同的值。

使用 IsDate 函数，可判断 date 是否可以被转换为日期或时间。CDate 可用来识别日期文字和时间文字，以及落入可接受的日期范围内的数值。当转换一个数字成为日期时，是将整数部分转换为日期，小数部分转换为从午夜起算的时间。

CDate 依据系统上的国别设置来决定日期的格式。如果提供的格式为不可识别的日期设置，则不能正确判断年、月、日的顺序。另外，长日期格式，若包含有星期的字符串，也不能被识别。

CVDate 函数也提供对早期 VisualBasic 版本的兼容性。CVDate 函数的语法与 CDate 函数是完全相同的，不过，CVDate 是返回一个 Variant，它的子类型是 Date，而不是实际的 Date 类型。因为现在已有真正的 Date 类型，所以 CVDate 也不再需要了。转换一个表达式成为 Date，再赋值给一个 Variant，也可以达到同样的效果。也可以使用这种技巧将其他真正的数据类型转换为对等的 Variant 子类型。

注意 CDec 函数不能返回独立的数据类型，而总是返回一个 Variant，它的值已经被转换为 Decimal 子类型。

CStr 返回值

如果 expression 是	CStr 返回
Boolean	含有 True 或 False 的字符串
Date	含有系统中短日期格式日期的字符串
Null	一个运行时错误
Empty	一个零长度字符串("")
Error	包含单词 Error 以及错误号的字符串
其他数值	含有数值的字符串

示例

本示例使用 CBool 函数来将一表达式转成 Boolean 值。如果表达式的结果为非零的值，CBool 返回 True；否则返回 False。

```
DimA,B,Check
A=5:B=5    '设置变量初值。
```

```
Check=CBool(A=B)    'Check 的值为 True。

A=0     '定义变量。
Check=CBool(A) 'Check 的值为 False。
```

本示例使用 CByte 函数将一表达式转成 Byte。

```
DimMyDouble,MyByte
MyDouble=125.5678  'MyDouble 为 Double（双精度）。
MyByte=CByte(MyDouble) 'MyByte 值为 126。
```

本示例使用 CCur 函数将表达式转成 Currency。

```
DimMyDouble,MyCurr
MyDouble=543.214588     'MyDouble 为 Double 类型。
MyCurr=CCur(MyDouble*2)     '将 MyDouble*2 的结果
   '（1086.429176）转换为 Currency（1086.4292）。
```

本示例使用 CDate 函数将字符串转换成 Date。一般说来，字符串格式的日期与时间硬编码（如示例中所示）并不好。较好的做法是使用日期原义表达式和时间的原义表达式（如#2/12/1969#，#4:45:23PM#）。

```
DimMyDate,MyShortDate,MyTime,MyShortTime
MyDate="February12,1969"    '定义日期。
MyShortDate=CDate(MyDate)  '转换为 Date 数据类型。
```

```
MyTime="4:35:47PM" '定义时间。
MyShortTime=CDate(MyTime)  '转换为 Date 数据类型。
```

本示例使用 CDbl 函数将表达式转换为 Double 类型。

```
DimMyCurr,MyDouble
MyCurr=CCur(234.456784)      'MyCurr 为 Currency 类型。
MyDouble=CDbl(MyCurr*8.2*0.01) '将结果转换为 Double 类型。
```

该示例使用 CDec 函数将数字值转换为 Decimal。

```
DimMyDecimal,MyCurr
MyCurr=10000000.0587                'MyCurr 是货币。
MyDecimal=CDec(MyCurr)       'MyDecimal 是二进制数。
```

本示例使用 CInt 函数将一数值转换为 Integer。

```
DimMyDouble,MyInt
MyDouble=2345.5678 'MyDouble 为 Double 类型。
MyInt=CInt(MyDouble)   'MyInt 的值为 2346。
```

本示例使用 CLng 函数将一数值转换为 Long 类型。

```
DimMyVal1,MyVal2,MyLong1,LyLong2
MyVal1=25427.45;MyVal2=25427.55,MyVal1,MyVal2areDoubles.
MyLong1=CLng(MyVal1),MyLong1Contaon2527.
```

```
MyLong2=CLng(MyVal2),MyLong2Contains25428.
```

本示例使用 CSng 函数将一数值转换为 Single 类型。

```
DimMyDouble1,MyDouble2,LySingle1,MySingle2
'MyDouble1,MyDouble2areDpubles.
MyDouble=75.3421115;MyDouble2=75.3421555.
MySingle1=CSng(MyDouble1)'MySingle1Contains75.34211.
MySingle2=CSng(MyDouble2)'MySingle2Contains75.34216.
```

本示例使用 CStr 函数将数字值转换为 String 类型。

```
DimMyDouble,MyString
MyDouble=437.324'MyDOubleisaDouble.
MyString=CStr(MyDouble)'MyStringcontains"437.324"
```

本示例使用 CVar 函数将表达式转换为 Variant 类型。

```
DimMyInt.MyVar
MyInt=4534'MyIntisanInteger.
MyVar=Cvar(MyInt&"000")'MyVarcontainsthestring
'4534000.
```

Val 函数

返回包含于字符串内的数字，字符串中是一个适当类型的数值。

语法

Val(*string*)

必要的 *string* 参数可以是任何有效的字符串表达式。

说明

Val 函数，在它不能识别为数字的第一个字符上，停止读入字符串。那些被认为是数值的一部分的符号和字符，例如美圆号与逗号，都不能被识别。但是函数可以识别进位制符号&O（八进制）和&H（十六进制）。空白、制表符和换行符都从参数中被去掉。

下面的返回值为 1615198：

```
Val("1615198thStreetN.E.")
```

在下面的代码中，Val 为所示的十六进制数值返回十进制数值-1。

```
Val("&HFFFF")
```

注意 Val 函数只会将句点（.）当成一个可用的小数点分隔符。当使用不同的小数点分隔符时,如在国际版应用程序中,代之以CDbl来把字符串转换为数字。

请参阅

类型转换函数，Str 函数

示例

本示例使用 Val 函数返回字符串中所含的数值。

```
DimMyValue
MyValue=Val("2457")      '返回 2457。
MyValue=Val("2457")      '返回 2457。
```

```
MyValue=Val("24and57") '返回24。
```