A

ApplyOutline
ApplyPerspectiveEffect
ApplyPostscriptFill
ApplyPreset
ApplyPresetEnvelope
ApplyRotatedExtrude
ApplyStyle
ApplyTextureFill
ApplyToDuplicate
ApplyTwoColorFill
ApplyUniformFillColor

B

BeforeObject
BeginDrawArrow
BeginDrawBezier
BeginDrawCurve
BeginDrawFreehand
BeginEditObject
BreakApart

C

ChangeLayerName
ClearEffect
ClickedDialogButton
CloneObject
CloseCurve
Combine
ConvertColor
ConvertToCurves
CopyEffectFrom
CopyPropertiesFrom
CopyToClipboard
CopyToLayer
CreateArtisticText
CreateEllipse
CreateGuidelineUsingAngle
CreateGuidelineUsingTwoPoints
CreateRectangle
CreateSymPolygon
CreateTextString
CurrentPage

D

DeleteGuidelineByIndex
DeleteGuidelineUsingAngle
DeleteGuidelineUsingTwoPoints
DeleteLayer
DeleteObject
DeletePages
DeleteStyle

DetachBlendPath
DisplayFacingPages
DistributeObjects
DrawCurveClosePath
DrawCurveCurveTo
DrawCurveLineTo
DrawCurveMoveTo
DropSymbol
DuplicateObject

E

EditObjectCommand
EndDrawArrow
EndDrawBezier
EndDrawCurve
EndDrawFreehand
EndEditObject
EndOfRecording
ExtractText

F

FileClose
FileExport
FileImport
FileNew
FileOpen
FilePrint
FileSave
FindNextObjectOfStyle
FindObjectOfStyle
FitTextToPath
FullScreenPreview
FuseBlend

G

GetFillType
GetFountainFillColor
GetFountainFill
GetGuidelineInformation
GetNumberOfGuidelines
GetObjectsCDRStaticID
GetObjectType
GetOutlineColor
GetOutline
GetPageSize
GetPosition
GetSize
GetUniformFillColor
GetUserDataField
Group

I

InitBezierTool
InsertOLEObjectFromFile
InsertOLEObject
InsertPages
Intersection

L

LoadStyles
LockGuidelineByIndex

M

MenuCommand
MergeBackText
MoveBezierControl
MoveCenter
MoveGuidelineUsingAngleByIndex
MoveGuidelineUsingTwoPointsByIndex
MoveObject
MoveToLayer

N

NewLayer

O

OLEObjectDoVerb
OrderBackOne
OrderForwardOne
OrderReverseOrder
OrderToBack
OrderToFront
OverPrintFill
OverPrintOutline

P

PasteCustomClipboardFormat
PasteFromClipboard
PasteSystemClipboardFormat

R

RecorderApplyPerspective
RecorderBeginEditParaText
RecorderBeginEditText
RecorderEditParaTextChangeCase
RecorderEditParaTextCharAttributes
RecorderEditParaTextIndents
RecorderEditParaTextReplaceText
RecorderEditParaTextSpacing
RecorderEditTextChangeCase
RecorderEditTextCharAttributes
RecorderEditTextReplaceText
RecorderEndEditParaText

RecorderEndEditText
RecorderObjectScaleInfo
RecorderSelectObjectByIndex
RecorderSelectObjectsByIndex
RecorderSelectPreselectedObjects
RecorderStorePreselectedObjects
Redo
RedrawAllScreens
RedrawScreen
ReferencePoint
RemoveAllGuidelines
RemoveFountainFillColor
Repeat
ResetTransfo
ResumePainting
RevertToStyle
RotateObject

S

SaveStyleAs
SaveTemplate
SelectAllObjects
SelectLayer
SelectNextObject
SelectObjectAtPoint
SelectObjectOfCDRStaticID
SelectObjectsInRect
SelectPreviousObject
Separate
SetApplyToDuplicate
SetArtisticText
SetBullet
SetCharacterAttributes
SetColorOverride
SetCurrentPage
SetFountainFillColor
SetFrameColumn
SetFullScreenPreview
SetIndents
SetLayerLocked
SetLayerPrintable
SetLayerVisible
SetMultiLayer
SetOptionsForAllPages
SetOutlineArrow
SetOutlineColor
SetOutlineStyle
SetOutlineWidth
SetPageLayout
SetPageOrientation

SetPageSizeFromPrinter
SetPageSize
SetPaperColor
SetParagraphSpacing
SetPosition
SetReferencePoint
SetSize
SetTextString
SetToMasterLayer
SetUserDataField
SetVisible
ShareExtrudeVP
ShowPageBorder
SkewObject
SplitBlend
StartOfRecording
StraightenText
StretchObject
SuppressPainting

T

Trim

U

Undo
Ungroup
UnlockGuidelineByIndex
UnSelectAll

V

Visible

W

Weld

File commands

FileClose (DRAW)

ReturnValue = .FileClose(.PromptUser = *boolean*)

This command closes the current drawing.

Syntax	Description
.ReturnValue	Returns TRUE (-1) if the function was successful. Returns FALSE (0) if the file was not closed.
.PromptUser	Set to TRUE (-1) to prompt the user before closing the file. Set to FALSE (0) to close the file without prompting the user.

 **Note**

- This command must be preceded by the .FileSave command or changes will be lost.

Example

```
.FileClose TRUE
```

The above example prompts the user before closing the active CorelDRAW document.

```
.FileClose
```

The above example closes the active CorelDRAW document without prompting the user.

{button ,AL(`OVR1 File commands;',0,"Defaultoverview",)} Related Topics

FileExport (DRAW)

.FileExport .FileName = *string*, .FilterID = *long*, .Width = *long*, .Height = *long*, .XResolution = *long*, .YResolution = *long*, .ImageType = *long*

This command saves the current drawing in a format that other programs can read.

Syntax	Description
.FileName	Specifies the name of the file to export.
.FilterID	Specifies the type of file filter. 769 = Windows Bitmap (BMP) 770 = Paintbrush (PCX) 771 = Targa Bitmap (TGA) 772 = TIFF Bitmap (TIF) 773 = CompuServe Bitmap (GIF) 774 = JPEG Bitmap (JPG) 776 = Scitex CT Bitmap (SCT) 777 = Wavelet Compressed Bitmap (WVL) 787 = GEM Paint File (IMG) 790 = MACPaint Bitmap (MAC) 800 = CALS Compressed Bitmap (CAL) 1280 = Computer Graphics Metafile (CGM) 1281 = HPGL Plotter File (PLT) 1283 = Adobe Illustrator (AI) 1284 = GEM File (GEM) 1285 = IBM PIF (PIF) 1287 = WordPerfect Graphics (WPG) 1288 = Macintosh Pict (PCT) 1289 = Encapsulated PostScript (EPS) 1291 = OS/2 PM Metafile (MET) 1294 = Windows Metafile (WMF) 1296 = AutoCad (DXF) 1792 = Corel PHOTO-PAINT Image (CPT) Ver 5.0/6.0 1793 = Corel CMX 6.0 1794 = Corel CMX 5.0 1795 = CorelDRAW CDR 1796 = Corel CDX (CDR compressed) 1797 = Corel CPX (CMX compressed) 1799 = Corel PHOTO-PAINT Image (CPT) 7.0 1800 = CorelDRAW Template (CDT) 1801 = CorelDRAW Pattern (PAT)
.Width	Specifies the width of the image in pixels.
.Height	Specifies the height of the image in pixels.
.XResolution	Specifies the horizontal resolution of the image in dots per inch (dpi).
.YResolution	Specifies the vertical resolution of the image in dots per inch (dpi).
.ImageType	Specifies the image type. 1 = Monochrome bitmap 3 = 8-bit paletted color bitmap 4 = 24-bit RGB color bitmap 6 = 32-bit CMYK bitmap 10 = 4-bit, 16 colors (standard VGA palette)

Example

```
.FileExport "C:\COREL70\DRAW\TEMP1.BMP", 769, 320, 400, 72, 72, 4
```

The above example exports a CorelDRAW file to a Windows bitmap named "TEMP1.BMP".

{button ,AL(`OVR1 File commands;',0,"Defaultoverview",)} **Related Topics**

FileImport (DRAW)

.FileImport .FileName = *string*

This command brings graphics into CorelDRAW from other programs.

Syntax	Description
.FileName	Specifies the name of the file to import.

Example

```
.FileNew
.FileImport "C:\COREL70\DRAW\TEST1.BMP"
```

The above example imports a Windows bitmap file named "TEST1.BMP" into the document.

{button ,AL(`OVR1 File commands;',0,"Defaultoverview",)} **Related Topics**

FileNew (DRAW)

ReturnValue = .FileNew

This command creates a new drawing.

Syntax	Description
.ReturnValue	Returns TRUE (-1) if the function was successful. Returns FALSE (0) if the file was not created.

Note

- You cannot change the active DRAW document in a script except by using the .FileNew or .FileOpen command. Changing the active DRAW document with keyboard and mouse actions does not affect an executing script.

Example

```
.FileNew
```

The above example creates a new CorelDRAW document.

{button ,AL(`OVR1 File commands;',0,"Defaultoverview",)} **Related Topics**

FileOpen (DRAW)

.FileOpen .FileName = *string*

This command loads a drawing or Styles template into CorelDRAW.

Syntax	Description
.FileName	Specifies the name of the file to open.

Note

- You cannot change the active DRAW document in a script except by using the .FileNew or .FileOpen command. Changing the active DRAW document with keyboard and mouse actions does not affect an executing script.

Example

```
.FileOpen "C:\COREL70\DRAW\TEST1.CDR"
```

The above example opens a CorelDRAW file named "TEST1.CDR".

{button ,AL(`OVR1 File commands;',0,"Defaultoverview",)} Related Topics

FilePrint (DRAW)

.FilePrint

This command prints the active document.

Example

```
.FilePrint
```

The above example sends the active document to the printer.

{button ,AL(`OVR1 File commands;',0,"Defaultoverview",)} **Related Topics**

FileSave (DRAW)

.FileSave .FileName = *string*, .ThumbNailSize = *long*, .SaveSelectedOnly = *boolean*, .FileVersion = *long*, .IncludeCMXData = *boolean*

This command saves the active document.

Syntax	Description
.FileName	Specifies the name of the file to save.
.ThumbNailSize	Specifies the size of the thumbnail: 0 = Current 1 = None 2 = 1k (mono) 3 = 5k (color) 4 = 10k (color)
.SaveSelectedOnly	Set to TRUE (-1) to save selected items only. Set to FALSE (0) to save entire document.
.FileVersion	Specifies the file version of the document being saved. 0 = Version 7.0 1 = Version 6.0 2 = Version 5.0
.IncludeCMXData	Set to TRUE (-1) to include CMX data with the saved file. Set to FALSE (0) to disable this feature.

Example

```
.FileSave "C:\COREL70\DRAW\TEST1.CDR", 1, 2, 0, 0
```

The above example saves a version 7 CorelDRAW document named "TEST1.CDR", with a 1k thumbnail. CMX data is not saved.

{button ,AL(`OVR1 File commands;',0,"Defaultoverview",)} **Related Topics**

Edit commands

CopyPropertiesFrom (DRAW)

.CopyPropertiesFrom .lObjectID = *long*, .bOutlinePen = *boolean*, .bOutlineColor = *boolean*, .bFill = *boolean*, .bTextAttibutes = *boolean*

This command copies the properties from the object with the specified object ID to the selected object.

Syntax	Description
.lObjectID	Specifies the object ID of the source object. Use .GetObjectsCDRStaticID to get an object's ID.
.bOutlinePen	Set to TRUE (-1) to copy outline pen properties. Set to FALSE (0) to exclude outline pen properties.
.bOutlineColor	Set to TRUE (-1) to copy outline color properties. Set to FALSE (0) to exclude outline color properties.
.bFill	Set to TRUE (-1) to copy fill properties. Set to FALSE (0) to exclude fill properties.
.bTextAttibutes	Set to TRUE (-1) to copy text properties. Set to FALSE (0) to exclude text properties.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

CopyToClipboard (DRAW)

.CopyToClipboard

This command places a copy of the selected object(s) or text onto the Clipboard.

Example

```
.CreateRectangle 750000, -750000, 0, 0
.CopyToClipboard
.InsertPages 0, 2
.PasteFromClipboard
```

The above example copies a rectangle to the Clipboard, inserts 2 pages, then pastes the contents of the Clipboard to the third page.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

InsertOLEObject (DRAW)

.InsertOLEObject .ProgID = *string*

This command inserts an OLE object in a CorelDRAW document.

Syntax	Description
.ProgID	Specifies the OLE object's Windows registry name.

Example

```
.InsertOLEObject "CorelPhotoPaint.Image.6"
```

The above example inserts a Corel PHOTO-PAINT image into a CorelDRAW document.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

InsertOLEObjectFromFile (DRAW)

.InsertOLEObjectFromFile .FileName = *string*, .CreateLink = *boolean*

This command inserts an OLE object from a file into a CorelDRAW document.

Syntax	Description
.FileName	The filename.
.CreateLink	Set to TRUE (-1) to create a link. Set to FALSE (0) to disable this option.

Example

```
.InsertOLEObjectFromFile "C:\WINWORD\WORDFILE.DOC", -1
```

The above example inserts a Microsoft Word file in a CorelDRAW document.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} Related Topics

OLEObjectDoVerb (DRAW)

.OLEObjectDoVerb .Verb = *long*

This command performs the specified action on an OLE object.

Syntax	Description
.Verb	Specifies the OLE object action to perform. 0 = Primary 1 = Secondary 2 = Tertiary etc.

 **Note**

- Primary and secondary verbs depend on the object type.

Example

```
.InsertOLEObject "CorelPhotoPaint.Image.7"
.OLEObjectDoVerb 0
```

The above example inserts a Corel PHOTO-PAINT OLE object into a DRAW document and invokes in-place editing.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

PasteCustomClipboardFormat (DRAW)

.PasteCustomClipboardFormat .Format = *string*

This command specifies the custom format for pasting from the Clipboard.

Syntax	Description
.Format	Specifies the type of format. Options include: "Corel 32-bit Presentation Exchange Data" "Corel Presentation Exchange Data" "Corel Metafile" "Rich Text Format"

Example

```
.PasteCustomClipboardFormat "Rich Text Format"
```

The above example inserts the contents of the Clipboard into a CorelDRAW document as Rich Text.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

PasteFromClipboard (DRAW)

.PasteFromClipboard

This command places a copy of the object(s) on the Clipboard into your drawing.

Example

```
.CreateRectangle 750000, -750000, 0, 0, 0
.CopyToClipboard
.InsertPages 0, 2
.PasteFromClipboard
```

The above example copies a rectangle to the Clipboard, inserts 2 pages, then pastes the contents of the Clipboard in to the last page inserted.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

PasteSystemClipboardFormat (DRAW)

.PasteSystemClipboardFormat .Format = *long*

This command specifies the system format for pasting from the Clipboard.

Syntax	Description
.Format	Specifies the type of format. 1 = CF Text 2 = Bitmap 3 = Metafile Pict 8 = DIB 14 = Enhanced Metafile

Example

```
.PasteSystemClipboardFormat 2
```

The above example pastes a bitmap from the Clipboard into the active document.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} Related Topics

Redo (DRAW)

.Redo

This command restores changes reversed by the Undo command. Redo becomes available immediately after you select the Undo command.

Example

```
.Redo
```

The above command reverses the last .Undo command and reinstates the previous deletion or reversal of actions.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

Repeat (DRAW)

.Repeat

This command applies, if possible, the most recent command or action to selected object.

Example

```
.Repeat
```

The above example repeats the last command.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

Undo (DRAW)

.Undo

This command reverses actions performed during the current session. Use Undo after you have made a change you do not want to implement. Immediately after you select .Undo, the .Redo command becomes available, allowing you to restore what you just undid. You cannot undo the following operations: any change of view (e.g., Zoom-in or Zoom-out); any file operations (e.g., Open, Save, or Import); any selection operations (e.g., Marquee select or Node select).

Example

```
.Undo
```

The above example undoes the last command.

{button ,AL(`OVR1 Edit commands;',0,"Defaultoverview",)} **Related Topics**

View commands

FullScreenPreview (DRAW)

ReturnValue = ObjDRAW.FullScreenPreview
ObjDRAW.FullScreenPreview = Value

This function is a property and will only work when excuted in a programming language that supports properties (e.g.. Visual Basic). Corel SCRIPT doesn't support properties. Use the .SetFullScreenPreview command in Corel SCRIPT to remove everything but your drawing from the screen.

In a programming language that supports properties, this function either returns a value that indicates whether CorelDRAW is in Full Screen Preview mode or not (i.e.,**ReturnValue = ObjDRAW.FullScreenPreview**), or this function puts CorelDRAW into Full Screen Preview mode or not (**ObjDRAW.FullScreenPreview = Value**).

{button ,AL(`OVR1 View commands;',0,"Defaultoverview",)} **Related Topics**

RedrawAllScreens (DRAW)

.RedrawAllScreens

This command forces CorelDRAW to redraw all open document windows.

{button ,AL(`OVR1 View commands;',0,"Defaultoverview",)} **Related Topics**

RedrawScreen (DRAW)

.RedrawScreen

This command forces CorelDRAW to redraw the windows of the active document.

{button ,AL(`OVR1 View commands;',0,"Defaultoverview",)} **Related Topics**

ResumePainting (DRAW)

.ResumePainting

This command instructs CorelDRAW to resume screen updating. To stop screen updating, use the .SupressPainting command.

{button ,AL(`OVR1 View commands;',0,"Defaultoverview",)} **Related Topics**

SetFullScreenPreview (DRAW)

.SetFullScreenPreview .FullScreen = *boolean*

This command removes everything but your drawing from the screen. You cannot edit your drawing in this mode.

Syntax	Description
.FullScreen	Set to TRUE (-1) to remove everything but your drawing from the screen. Set to FALSE (0) to return to normal mode.

Example

```
.SetFullScreenPreview -1
```

The above example displays a full-screen preview of the active image.

{button ,AL(`OVR1 View commands;',0,"Defaultoverview",)} **Related Topics**

SetVisible (DRAW)

.SetVisible .Visible = *boolean*

This command makes the CorelDRAW application visible.

Syntax	Description
.Visible	Set to TRUE (-1) to show the CorelDRAW application.

Example

```
.SetVisible -1
```

The above example makes the CorelDRAW application visible.

{button ,AL(`OVR1 View commands;',0,"Defaultoverview",)} Related Topics

SuppressPainting (DRAW)

.SuppressPainting

This command instructs CorelDRAW to suppress screen updating. To resume screen updating, use the .ResumePainting command.

{button ,AL(`OVR1 View commands;',0,"Defaultoverview",)} **Related Topics**

Visible (DRAW)

ReturnValue = ObjDRAW.Visible
ObjDRAW.Visible = Value

This function is a property and will only work when executed in a programming language that supports properties (e.g.. Visual Basic). Corel SCRIPT doesn't support properties. Use the .SetVisible command in a Corel SCRIPT script to make CorelDRAW hidden or visible, and use GetVisible to determine if CorelDRAW is visible or not.

In a programming language that supports properties, this function either returns a value that indicates whether CorelDRAW is visible or not (i.e.,**ReturnValue = ObjDRAW.Visible**), or this function makes CorelDRAW visible or not (**ObjDRAW.Visible = Value**).

{button ,AL(`OVR1 View commands;',0,"Defaultoverview",)} **Related Topics**

Layout commands

AddPageFrame (DRAW)

.AddPageFrame

This command puts a printable background frame around the page.

Example

```
.AddPageFrame
```

The above example creates a frame around the new page.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

ChangeLayerName (DRAW)

.ChangeLayerName .LayerName = *string*

This command lets you assign a new name to the active layer.

Syntax	Description
.LayerName	Specifies the new name of the Layer.

Example

```
.ChangeLayerName "NewName"
```

The above example changes the layer name to "NewName."

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

CopyToLayer (DRAW)

.CopyToLayer .LayerName = *string*

This command places a copy of the selected object on the layer indicated in the LayerName.

Syntax	Description
.LayerName	Specifies the name of the destination layer.

Example

```
.CreateRectangle -200000, 200000, -900000, 900000, 0
.CopyToLayer "Layer2"
```

The above example creates a rectangle and copies it to "Layer2."

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

CurrentPage (DRAW)

ReturnValue& = ObjDRAW.CurrentPage
ObjDRAW.CurrentPage = Value&

This function is a property and will only work when excuted in a programming language that supports properties (e.g.. Visual Basic). Corel SCRIPT doesn't support properties. Use the .SetCurrentPage command in Corel SCRIPT to go to a specific page.

In a programming language that supports properties, this function either returns a value that indicates which page is the current page (i.e.,**ReturnValue& = ObjDRAW.CurrentPage**), or this function goes to a specific page (**ObjDRAW.CurrentPage = Value&**).

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

DeleteLayer (DRAW)

.DeleteLayer

This command deletes the active layer and any objects on it.

Example

```
.MoveToLayer "NewLayer1"
.DeleteLayer
```

The above example moves to the layer named "NewLayer 1" and deletes it.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

DeletePages (DRAW)

.DeletePages .BeforeCurrentPage = *boolean*, .NumberOfPages = *long*

This command deletes pages from the current drawing.

Syntax	Description
.BeforeCurrentPage	Set to TRUE (-1) to enable deletion before the current page. Set to FALSE (0) to enable deletion after the current page.
.NumberOfPages	Specifies the number of pages to delete. Note: The current page is included in the deletion.

Example

```
.CreateRectangle 750000, -750000, 0, 0, 0
.CopyToClipboard
.InsertPages 0, 4
.PasteFromClipboard
.DeletePages -1, 2
```

The above example inserts 4 pages after the current page, pastes the contents of the Clipboard on the fourth page, then deletes the current page and the two pages that precede it.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} Related Topics

DisplayFacingPages (DRAW)

.DisplayFacingPages .FacingPages = *boolean*, .LeftFirst = *boolean*

This command displays two consecutive pages on the screen at the same time.

Syntax	Description
.FacingPages	Set to TRUE (-1) to display two consecutive pages on the screen at the same time. Working in this view allows you to draw objects that lie partially on both pages at the same time. Set to FALSE (0) to disable this option.
.LeftFirst	Set to TRUE (-1) to display odd pages on the left. Set to FALSE (0) to display odd pages on the right.

Example

```
.FileNew
.DisplayFacingPages 0, -1 'Displays one page
```

The above example displays one page.

```
.FileNew
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.CreateRectangle 750000, -750000, 0, 0, 0
.CopyToClipboard
.InsertPages 0, 4
.PasteFromClipboard
.DisplayFacingPages -1, -1 'Displays two pages
```

The above example displays facing pages with the current page on the left.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

GetPageSize (DRAW)

.GetPageSize .lWidth = *long**, .lHeight = *long**

This command returns the width and height of the document page

Syntax	Description
.lWidth	Returns the width of the page in tenths of a micron.
.lHeight	Returns the height of the page in tenths of a micron.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} Related Topics

InsertPages (DRAW)

.InsertPages .BeforeCurrentPage = *boolean*, .NumberOfPages = *long*

This command inserts the specified number of pages into the current drawing.

Syntax	Description
.BeforeCurrentPage	Set to TRUE (-1) to position insertion point before the current page. Set to FALSE (0) to position insertion point after the current page.
.NumberOfPages	Specifies the number of pages to insert.

Example

```
.InsertPages 0, 4
```

The above example inserts 4 pages after the current page.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

MoveToLayer (DRAW)

.MoveToLayer .LayerName = *string*

This command moves the selected object to the layer selected in the Layers list.

Syntax	Description
.LayerName	Specifies the name of the destination layer.

Example

```
.MoveToLayer "NewLayer1"
```

The above example moves the selected object(s) to the layer named "NewLayer1."

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

NewLayer (DRAW)

.NewLayer .LayerName = *string*

This command lets you create a new layer and assign a name.

Syntax	Description
.LayerName	Specifies the name of the new layer.

Example

```
.NewLayer "NewLayer1"
```

The above example creates a new layer named "NewLayer1."

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

ReferencePoint (DRAW)

ReturnValue& = ObjDRAW.ReferencePoint
ObjDRAW.ReferencePoint = Value&

This function is a property and will only work when excuted in a programming language that supports properties (e.g.. Visual Basic). Corel SCRIPT doesn't support properties. Use the .SetReferencePoint command in Corel SCRIPT to go to a specific page.

In a programming language that supports properties, this function either returns a value that indicates what the current reference point is (i.e.,**ReturnValue& = ObjDRAW.ReferencePoint**), or this function goes to a specific reference point (**ObjDRAW.ReferencePoint = Value&**).

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SelectLayer (DRAW)

.SelectLayer .LayerName = *string*

This command lets you select a layer, making it the active layer.

Syntax	Description
.LayerName	Specifies the name of the selected layer.

Example

```
.SelectLayer "NewLayer1"
```

The above example selects the layer named "NewLayer1" and makes it the active layer.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetApplyToDuplicate (DRAW)

.SetApplyToDuplicate .ApplyToDuplicate = *boolean*

This command opens and closes a block of object-duplicating commands. An object must be selected to use this command. The duplicated object can be repositioned, resized, skewed, or rotated.

Syntax	Description
.ApplyToDuplicate	Set to TRUE (-1) to open a block of object-duplicating commands. Set to FALSE (0) to close the block.

Note

- The following commands can be used to duplicate objects within the .SetApplyToDuplicate block:

  .SetPosition
  .SkewObject
  .SetSize
  .RotateObject

  The duplicated object is selected.

Example

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.SetPosition 55555, 900000
.SetApplyToDuplicate TRUE
.SetPosition 0, 0 'Creates another object
.ApplyUniformFillColor 2, 255, 0, 0, 0
.SetPosition 55555, 100000 'Creates another object
.ApplyUniformFillColor 2, 0, 255, 0, 0
.SkewObject -15000000, 2000000, 3 'Creates another object
.SetSize 444444, 555555 'Creates another object
.RotateObject 45000000, 0, 0, 0 'Creates another object
.SetApplyToDuplicate FALSE
.SetPosition 0, 0
```

The above example creates an ellipse then creates 5 more ellipses in the SetApplyToDuplicate block.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetColorOverride (DRAW)

.SetColorOverride .Override = *boolean*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command outlines objects on a layer in the selected color. Objects on the selected layer will appear with a wireframe outline of the chosen color.

Syntax	Description
.Override	Set to TRUE (-1) to outline objects on a layer in the selected color. Set to FALSE (0) to disable this option.
.ColorModel	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click + for valid value ranges.

.Color2 Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click + for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click + for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color4 Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click + for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.SetColorOverride -1, 3, 255, 0, 0, 0
```

The above example sets the override color to cyan.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetCurrentPage (DRAW)

.SetCurrentPage .CurrentPage = *long*

This command makes the specified page the current page.

Syntax	Description
.CurrentPage	Specifies which page to make the current page.

Example

```
.SetCurrentPage 2
```

The above example sets the second page as the current page.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetLayerLocked (DRAW)

.SetLayerLocked .Locked = *boolean*

This command enables or disables selection of objects on a layer. Locking a layer prevents objects on it from being accidentally moved or changed in any way. You cannot add new objects to a locked layer.

Syntax	Description
.Locked	Set to TRUE (-1) to lock a layer, preventing objects on it from being accidentally moved or changed in any way. You cannot add new objects to a locked layer. Set to FALSE (0) to unlock a layer.

 **Note**

- If .SetOptionsForAllPages is set TRUE (-1), then the .SetLayerLocked command applies to all pages.

Example

```
.SetLayerLocked -1
```

The above example locks the current layer.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetLayerPrintable (DRAW)

.SetLayerPrintable .Printable = *boolean*

This command enables or disables printing of objects on the current layer.

Syntax	Description
.Printable	Set to TRUE (-1) to enable printing of the current layer. Set to FALSE (0) to disable printing of the current layer.

Note

- If .SetOptionsForAllPages is set TRUE (-1), then the .SetLayerPrintable command applies to all pages.

Example

```
.SetLayerPrintable 0
```

The above example disables printing of the current layer.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} Related Topics

SetLayerVisible (DRAW)

.SetLayerVisible .Visible = *boolean*

This command makes objects on a layer visible or invisible.

Syntax	Description
.Visible	Set to TRUE (-1) to make the current layer visible. Set to FALSE (0) to make the current layer invisible.

Note

- If .SetOptionsForAllPages is set TRUE (-1), then the .SetLayerVisible command applies to all pages.

Example

```
.SetLayerVisible -1
```

The above example makes the current layer visible.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetMultiLayer (DRAW)

.SetMultiLayer .MultiLayer = *boolean*

This command lets you select objects on all layers that are not locked or invisible.

Syntax	Description
.MultiLayer	Set to TRUE (-1) to enable selection of objects across all layers except those which are locked or invisible. Set to FALSE (0) to disable selection of objects across all layers — only objects on the current layer can be selected.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} Related Topics

SetOptionsForAllPages (DRAW)

.SetOptionsForAllPages .AllPages = *boolean*

This command enables CorelDRAW options to be set for all pages.

Syntax	Description
.AllPages	Set to TRUE (-1) to enable options to be set for all pages. Set to FALSE (0) to disable this option.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetPageLayout (DRAW)

.SetPageLayout .LayoutType = *long*

This command lets you specify a page layout.

Syntax	Description
.LayoutType	Specifies the style of the page layout: 1 = Full Page: Prints one full page per sheet. 2 = Book: Prints two pages per sheet, which you would cut down the middle. 3 = Booklet: Prints two pages per sheet, which you would fold vertically to obtain a side fold. 4 = Tent Card: Prints two pages per sheet, which you would fold horizontally to obtain a top fold. 5 = Side-Fold Card: Prints four pages per sheet, which you would fold first horizontally to create the top fold, then vertically to create the side fold. 6 = Top-Fold Card: Prints four pages per sheet, which you would fold first vertically to create the side fold, then horizontally to create the top fold.

Example

```
.SetPageLayout 3
```

The above example sets the page layout to booklet style.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetPageOrientation (DRAW)

.SetPageOrientation .lOrient = *long*

This command changes the orientation of the page.

Syntax	Description
.lOrient	0 = portrait 1 = landscape

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetPageSize (DRAW)

.SetPageSize .Width = *long*, .Height = *long*

This command lets you set the page size for the document.

Syntax	Description
.Width	Specifies the new page width in tenths of a micron.
.Height	Specifies the new page height in tenths of a micron.

Note

- You can use the LENGTHCONVERT function, or one of the FROM... or TO... functions to specify length measurements.

Example

```
.SetPageSize 1000000,1350000
```

The above example sets the page size to 1,000,000 microns wide by 1,350,000 microns high (or 3.94 inches by 5.31 inches).

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetPageSizeFromPrinter (DRAW)

.SetPageSizeFromPrinter

This command sets the page size and orientation of the current document to the current settings of the default printer.

Example

```
.SetPageSizeFromPrinter
```

The above example queries the printer to set the page size.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetPaperColor (DRAW)

.SetPaperColor .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command lets you color the Preview screen (and the Drawing Window, if you are working in the Editable Preview) to approximate the paper you plan to print it on.

Syntax	Description
.ColorModel	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.

.Color2 Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color4 Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.SetPaperColor 2, 0, 255, 0, 0
```

The above example sets the paper color to magenta.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetPosition (DRAW)

.SetPosition .XPos = *long*, .YPos = *long*

This command sets the position for placement of the selected object

Syntax	Description
.XPos	Specifies the X-coordinate of the new position in tenths of a micron.
.YPos	Specifies the Y-coordinate of the new position in tenths of a micron.

Example

```
.CreateRectangle 1350000, -1000000, 750000, -500000, 0
.CreateArtisticText "1"
.SetPosition -950000, 1250000
```

The above example creates a rectangle and positions a number '1' in its upper-left corner.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetReferencePoint (DRAW)

.SetReferencePoint .ReferencePoint = *long*

This command sets the specified Reference Point for a selected object. The reference point is used to set the object handle for subsequent commands such as .SetPosition.

Syntax	Description
.ReferencePoint	Specifies the reference point to set. 1 = Upper-left 2 = Upper-middle 3 = Upper-right 4 = Middle-right 5 = Lower-right 6 = Lower-middle 7 = Lower-left 8 = Middle-left 9 = Center

Example

```
.CreateRectangle 1250000, -1000000, 750000, -500000, 0
.SetReferencePoint 9
.SetPosition 0, 0
```

The above example creates a rectangle, sets its reference point to the center and positions it in the center of the page.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetSize (DRAW)

.SetSize .XSize = *long*, .YSize = *long*

This command lets you scale, mirror, or set the size of the selected object.

Syntax	Description
.XSize	Specifies the new horizontal size of the selected object, in tenths of a micron.
.YSize	Specifies the new vertical size of the selected object, in tenths of a micron.

Note

- To mirror an object, use negative values for the .XSize and .YSize parameters.

Example

```
.CreateRectangle 1000000, 750000, 500000, 100000, 0
id& = .GetObjectsCDRStaticID()
status& = .GetSize (XSize&, YSize&)
.SelectObjectOfCDRStaticID id&
.SetSize 2*XSize&, 3*YSize&
```

The above example gets the size of the selected rectangle and sets the width to twice the original size, and the height to three times the original size.

```
.CreateRectangle 1000000, 750000, 500000, 100000, 0
id& = .GetObjectsCDRStaticID()
status& = .GetSize (XSize&, YSize&)
.SelectObjectOfCDRStaticID id&
.SetSize -XSize&, YSize&
```

The above example horizontally mirrors the selected object, maintaining its original size.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

SetToMasterLayer (DRAW)

.SetToMasterLayer .Master = *boolean*

This command lets you set the selected object to a master layer. When you want the same element, for example, a company logo, to appear on every page of a document, use this command to set the "master layers" to contain the repeating elements.

Syntax	Description
.Master	Set to TRUE (-1) to enable, applying the Master Layer template to all layers. Set to FALSE (0) to disable this option.

Example

```
.CreateRectangle 1350000, -1000000, 750000, -500000, 0
.SetToMasterLayer -1
```

The above example sets the rectangle to the master layer.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} **Related Topics**

ShowPageBorder (DRAW)

.ShowPageBorder .ShowBorder = *boolean*

This command enables and disables the page border.

Syntax	Description
.ShowBorder	Set to TRUE (-1) to show the page border. Set to FALSE (0) to suppress the page border.

Example

```
.ShowPageBorder -1
```

The above example shows the page border.

```
.ShowPageBorder 0
```

The above example hides the page border.

{button ,AL(`OVR1 Layout commands;',0,"Defaultoverview",)} Related Topics

Styles commands

ApplyStyle (DRAW)

.ApplyStyle .Style = *string*

This command lets you apply a style to the selected object.

Syntax	Description
.Style	Specifies the name of the style.

Example

```
.SelectAllObjects
.ApplyStyle "Default Graphic"
```

The above example applies the 'Default Graphic' style to all selected objects.

{button ,AL(`OVR1 Styles commands;',0,"Defaultoverview",)} **Related Topics**

DeleteStyle (DRAW)

.DeleteStyle .Style = *string*

This command deletes styles. When you delete a style, objects with that style revert to the default style for that object type. The object's appearance does not change when it reverts to the default style.

Syntax	Description
.Style	Specifies the name of the style to delete.

Example

```
.DeleteStyle "Style 1"
```

The above example deletes the style named "Style 1."

{button ,AL(`OVR1 Styles commands;',0,"Defaultoverview",)} Related Topics

LoadStyles (DRAW)

.LoadStyles .StyleSheet = *string*

This command loads the styles from a template into the active drawing.

Syntax	Description
.StyleSheet	Specifies the name of the template to use.

Example

```
.LoadStyles "C:\COREL\Programs\mine.cdt"
```

The above example loads the styles from the template file "MINE.CDT" into the active document.

{button ,AL(`OVR1 Styles commands;',0,"Defaultoverview",)} **Related Topics**

RevertToStyle (DRAW)

.RevertToStyle

This command converts an object to its original style.

{button ,AL(`OVR1 Styles commands;',0,"Defaultoverview",)} **Related Topics**

SaveStyleAs (DRAW)

.SaveStyleAs .style = *string*, .bFill = *boolean*, .bOutline = *boolean*, .bTypeface = *boolean*, .bTypeStyle = *boolean*, .bSize = *boolean*, .bJustification = *boolean*, .bTabs = *boolean*, .bHyphenation = *boolean*, .bSpaceChar = *boolean*, .bSpaceWord = *boolean*, .bSpaceLine = *boolean*, .bBeforePara = *boolean*, .bAfterPara = *boolean*, .bUnderline = *boolean*, .bOverline = *boolean*, .bStrikeout = *boolean*, .bBulletIndent = *boolean*, .bFirstLineIndent = *boolean*, .bRestOfLinesIndent = *boolean*, .bRightMargin = *boolean*, .bSuperOrSubScript = *boolean*, .bCapitalize = *boolean*, .bBullet = *boolean*

This command saves the style of the current object as a new style.

Syntax	Description
.style	Specifies the name of the new style.
.bFill	Set to TRUE (-1) to include fill properties. Set to FALSE (0) to exclude these properties.
.bOutline	Set to TRUE (-1) to include outline properties. Set to FALSE (0) to exclude these properties.
.bTypeface	Set to TRUE (-1) to include typeface properties. Set to FALSE (0) to exclude these properties.
.bTypeStyle	Set to TRUE (-1) to include type style properties. Set to FALSE (0) to exclude these properties.
.bSize	Set to TRUE (-1) to include size properties. Set to FALSE (0) to exclude these properties.
.bJustification	Set to TRUE (-1) to include text justification properties. Set to FALSE (0) to exclude these properties.
.bTabs	Set to TRUE (-1) to include tab stop properties. Set to FALSE (0) to exclude these properties.
.bHyphenation	Set to TRUE (-1) to include hyphenation properties. Set to FALSE (0) to exclude these properties.
.bSpaceChar	Set to TRUE (-1) to include character spacing properties. Set to FALSE (0) to exclude these properties.
.bSpaceWord	Set to TRUE (-1) to include word spacing properties. Set to FALSE (0) to exclude these properties.
.bSpaceLine	Set to TRUE (-1) to include line spacing properties. Set to FALSE (0) to exclude these properties.
.bBeforePara	Set to TRUE (-1) to include paragraph spacing properties (before the paragraph). Set to FALSE (0) to exclude these properties.
.bAfterPara	Set to TRUE (-1) to include paragraph spacing properties (after the paragraph). Set to FALSE (0) to exclude these properties.
.bUnderline	Set to TRUE (-1) to include text underline properties. Set to FALSE (0) to exclude these properties.
.bOverline	Set to TRUE (-1) to include text overline properties. Set to FALSE (0) to exclude these properties.
.bStrikeout	Set to TRUE (-1) to include text strikeout properties. Set to FALSE (0) to exclude these properties.
.bBulletIndent	Set to TRUE (-1) to include bullet indentation properties. Set to FALSE (0) to exclude these properties.
.bFirstLineIndent	Set to TRUE (-1) to include indentation properties for the first line. Set to FALSE (0) to exclude these properties.
.bRestOfLinesIndent	Set to TRUE (-1) to include indentation properties for remaining lines (hanging indent). Set to FALSE (0) to exclude these properties.
.bRightMargin	Set to TRUE (-1) to include right margin properties. Set to FALSE (0) to exclude these properties.
.bSuperOrSubScript	Set to TRUE (-1) to include superscript or subscript properties. Set to FALSE (0) to exclude these properties.
.bCapitalize	Set to TRUE (-1) to include capitalization properties. Set to FALSE (0) to exclude these properties.

.bBullet	Set to TRUE (-1) to include bullet properties. Set to FALSE (0) to exclude these properties.

{button ,AL(`OVR1 Styles commands;',0,"Defaultoverview",)} **Related Topics**

SaveTemplate (DRAW)

.SaveTemplate .StyleSheet = *string*

This command lets you save the styles in the active document as a template.

Syntax	Description
.StyleSheet	Specifies the name of the Style Sheet to save.

Example

```
.SaveTemplate "C:\COREL70\DRAW\TMPLATE1.CDT"
```

The above example saves a template named "TMPLATE1.CDT" in the DRAW folder.

{button ,AL(`OVR1 Styles commands;',0,"Defaultoverview",)} **Related Topics**

Object selection commands

AfterObject (DRAW)

.AfterObject .lObjectID = *long*

This command selects the object that is after the reference object in the object tree. Use .lObjectID to specify the reference object.

Syntax	Description
.lObjectID	Specifies the object ID of the reference object. Use .GetObjectsCDRStaticID to get an object's ID.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} Related Topics

AppendObjectToSelection (DRAW)

.AppendObjectToSelection .lObjectID = *long*

This command adds the object with the specified object ID to the existing selection.

Syntax	Description
.lObjectID	Specifies the object ID of the object to append. Use .GetObjectsCDRStaticID to get an object's ID.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} Related Topics

BeforeObject (DRAW)

.BeforeObject .lObjectID = *long*

This command selects the object that is before the reference object in the object tree. Use .lObjectID to specify the reference object.

Syntax	Description
.lObjectID	Specifies the object ID of the reference object. Use .GetObjectsCDRStaticID to get an object's ID.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} Related Topics

FindNextObjectOfStyle (DRAW)

ReturnValue = .FindNextObjectOfStyle()

This function finds the next object with the current style.

Syntax	Description
ReturnValue	Returns TRUE (-1) if an object is found or FALSE (0) if no object is found.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} **Related Topics**

FindObjectOfStyle (DRAW)

ReturnValue = .FindObjectOfStyle(.pStyleName = *string*)

This function finds the next object with the specified style.

Syntax	Description
ReturnValue	Returns TRUE (-1) if an object is found or FALSE (0) if no object is found.
.pStyleName	Specifies the name of the style.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} **Related Topics**

GetObjectType (DRAW)

ReturnValue& = .GetObjectType()

This function returns a value that indicates the type of selected object. If more than one object is selected, the function returns the type of the last selected object.

Syntax	Description
ReturnValue&	0 = Reserved for future use 1 = Rectangle 2 = Ellipse 3 = Curve 4 = Text 5 = Bitmap 6 = Paragraph Text 7 = OLE 9 = Symmetrical Polygon 12 = Grouped objects

Example

```
objType& = .GetObjectType()
MESSAGE objType&
```

The above example displays a number that corresponds to the type of selected object in a message box.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} Related Topics

SelectAllObjects (DRAW)

.SelectAllObjects

This command selects every object in your drawing, including any not currently in view.

Example

```
.SelectAllObjects
```

The above example selects all objects in the active document.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} **Related Topics**

SelectNextObject (DRAW)

.SelectNextObject .SelectInsideGroup = *boolean*

This command lets you select the next object in the drawing. Repeat this command until the object you want is selected.

Syntax	Description
.SelectInsideGroup	Set to TRUE (-1) to permit object selection within a group of objects. Set to FALSE (0) to disable this option.

Example

```
.SelectNextObject -1
```

The above example selects the next object in the drawing. If that object is in a group, it can be selected.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} **Related Topics**

SelectObjectAtPoint (DRAW)

ReturnValue = .SelectObjectAtPoint(.XPos = *long*, .YPos = *long*, SelectInsideGroup = *boolean*)

This command toggles the selection of an object at the specified point. Using this command is the same as holding down SHIFT and clicking an object during a DRAW session.

Syntax	Description
.XPos	Specifies one of the X-coordinates of the selected object in tenths of a micron, relative to the center of the page.
.YPos	Specifies one of the Y-coordinates of the selected object in tenths of a micron, relative to the center of the page.
.SelectInsideGroup	Set to TRUE (-1) to permit object selection within a group of objects. Set to FALSE to disable this option.

Example

```
.CreateRectangle 1350000, -1000000, 1300000, 0, 0
.CreateRectangle 1000000, -750000, 500000, 100000, 0
.CreateRectangle 100000, -500000, -100000, 50000, 0
.CreateRectangle -750000, -500000, -250000, 50000, 0
.UnSelectAll
.SelectObjectAtPoint -750000, 500000, 0
.ApplyUniformFillColor 2, 255, 0, 0, 0
```

The above example creates four rectangles, then selects the second one and fills it with cyan.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} **Related Topics**

SelectObjectOfCDRStaticID (DRAW)

.SelectObjectOfCDRStaticID .CDRStaticID = *long*

This command selects the object with the specified CDRStaticID.

Syntax	Description
.CDRStaticID	Specifies the CDRStaticID number of the object to select.

Example

```
.CreateRectangle 750000, -600000, 250000, -100000, 0
IDRect& = .GetObjectsCDRStaticID()
.SelectObjectOfCDRStaticID IDRect&
```

The above example demonstrates object selection using the object's CDRStaticID.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} **Related Topics**

SelectObjectsInRect (DRAW)

.SelectObjectsInRect .Top = *long*, .Left = *long*, .Bottom = *long*, .Right = *long*, .IncludeIntersecting = *boolean*

This command selects all objects found within the defined rectangular area

Syntax	Description
.Top	Specifies the Y-coordinate of the upper-left corner of the distribution rectangle in tenths of a micron, relative to the center of the page.
.Left	Specifies the X-coordinate of the upper-left corner of the distribution rectangle in tenths of a micron, relative to the center of the page.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the distribution rectangle in tenths of a micron, relative to the center of the page.
.Right	Specifies the X-coordinate of the lower-right corner of the distribution rectangle in tenths of a micron, relative to the center of the page.
.IncludeIntersecting	Set to TRUE (-1) to included intersecting objects in the selection. Set to FALSE (0) to disable this option.

Example

```
.SelectObjectsInRect 1350000, -1000000, -1350000, 1000000, 0
```

The above example selects all objects within the specified rectangle.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} Related Topics

SelectPreviousObject (DRAW)

.SelectPreviousObject .SelectInsideGroup = *boolean*

This command lets you select the previously selected object in the drawing. Repeat this command until the object you want is selected. The objects are selected in the order in which they were created.

Syntax	Description
.SelectInsideGroup	Set to TRUE (-1) to permit object selection within a group of objects. Set to FALSE (0) to disable this option.

Example

```
.SelectPreviousObject -1
```

The above example selects the previous object in the group.

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} **Related Topics**

UnSelectAll (DRAW)

.UnSelectAll

This command deselects all objects.

Example

```
.UnSelectAll
```

The above example deselects all selected object(s).

{button ,AL(`OVR1 Object selection commands;',0,"Defaultoverview",)} **Related Topics**

Object creation commands

AddBezierPoint (DRAW)

.AddBezierPoint .lX = *long*, .lY = *long*, .bConstrain = *boolean*, .bCusp = *boolean*

This command creates the second point of a bezier segment created by using the .BeginDrawBezier command. The segment itself is not added until the .EndDrawBezier command is called.

Syntax	Description
.lX	Specifies the X-coordinate of the point in tenths of a micron, relative to the center of the page.
.lY	Specifies the Y-coordinate of the point in tenths of a micron, relative to the center of the page.
.bConstrain	Set to TRUE (-1) to use the constrain angle when positioning the point.
.bCusp	Set to TRUE (-1) to make the new node cusped. Set to FALSE (0) to make it symmetrical (except for end nodes).

Example

```
.InitBezierTool
.BeginDrawBezier -3085992, 163280, FALSE
.MoveBezierControl -1649128, 1142960, FALSE
.AddBezierPoint 146952, -277576, FALSE, FALSE
.MoveBezierControl -326560, -1453192, FALSE
.EndDrawBezier
```

The above example draws a simple bezier curve.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

AddFreehandPoint (DRAW)

.AddFreehandPoint .bConvertToDPCoords = *boolean*, .lX = *long*, .lY = *long*

This command adds a point to a freehand curve created by using the .BeginDrawFreehand command. The segment itself is not added until .EndDrawFreehand command is called.

Syntax	Description
.bConvertToDPCoords	Set to TRUE (-1) to convert the X and Y coordinates to physical coordinates on the screen. This parameter should always be set to true unless you know that the coordinates you are using are already physical coodinates on the screen.
.lX	Specifies the X-coordinate of the point in tenths of a micron, relative to the center of the page.
.lY	Specifies the Y-coordinate of the point in tenths of a micron, relative to the center of the page.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

AppendCurveLine (DRAW)

.AppendCurveLine .lX1 = *long*, .lY1 = *long*, .lX2 = *long*, .lY2 = *long*

This command adds a line segment to an existing curve. An open curve must be selected.

Syntax	Description
.lX1	Specifies the X-coordinate of the start point in tenths of a micron, relative to the center of the page.
.lY1	Specifies the Y-coordinate of the start point in tenths of a micron, relative to the center of the page.
.lX2	Specifies the X-coordinate of the end point in tenths of a micron, relative to the center of the page.
.lY2	Specifies the Y-coordinate of the end point in tenths of a micron, relative to the center of the page.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

BeginDrawBezier (DRAW)

.BeginDrawBezier .lX = *long*, .lY = *long*, .bCusp = *boolean*

This command creates the first point of a bezier segment. The second point is added by the .AddBezierPoint command. The segment itself is not added until the .EndDrawBezier command is called. This command must be preceeded by the .InitBezierTool command.

Syntax	Description
.lX	Specifies the X-coordinate of the point in tenths of a micron, relative to the center of the page.
.lY	Specifies the Y-coordinate of the point in tenths of a micron, relative to the center of the page.
.bCusp	Set to TRUE (-1) to make the new node cusped. Set to FALSE (0) to make it symmetrical (except for end nodes).

Note

- The bezier commands include:

  .AddBezierPoint
  .MoveBezierControl

Example

```
.InitBezierTool
.BeginDrawBezier -3085992, 163280, FALSE
.MoveBezierControl -1649128, 1142960, FALSE
.AddBezierPoint 146952, -277576, FALSE, FALSE
.MoveBezierControl -326560, -1453192, FALSE
.EndDrawBezier
```

The above example draws a simple bezier curve.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

BeginDrawCurve (DRAW)

.BeginDrawCurve .X = *long*, .Y = *long*

This command sets the coordinates of the starting node when drawing curves in Freehand mode.

Syntax	Description
.X	Specifies the X-coordinate of the starting node of the curve in tenths of a micron, relative to the center of the page.
.Y	Specifies the Y-coordinate of the starting node of the curve in tenths of a micron, relative to the center of the page.

 **Note**

- The .BeginDrawCurve command must be followed by a contiguous block of one or more DrawCurve commands, and one .EndDrawCurve command. The DrawCurve commands include:

  .DrawCurveClosePath
  .DrawCurveCurveTo
  .DrawCurveLineTo
  .DrawCurveMoveTo

Example

```
.BeginDrawCurve -500000, 1000000
.DrawCurveLineTo 500000, -500000
.EndDrawCurve
```

The above example demonstrates the DrawCurve commands.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

BeginDrawFreehand (DRAW)

.BeginDrawFreehand .bConvertToDPCoords = *boolean*, .lX = *long*, .lY = *long*

This commandcreates the first point of a freehand segment. Additional points are added to the segment by using the .AddFreehandPoint command. The curve itself is not added until the .EndDrawFreehand command is called.

Syntax	Description
.bConvertToDPCoords	Set to TRUE (-1) to convert the X and Y coordinates to physical coordinates on the screen. This parameter should always be set to true unless you know that the coordinates you are using are already physical coodinates on the screen.
.lX	Specifies the X-coordinate of the point in tenths of a micron, relative to the center of the page.
.lY	Specifies the Y-coordinate of the point in tenths of a micron, relative to the center of the page.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

CloneObject (DRAW)

.CloneObject

This command copies the selected object and offsets the copy from the original. Most changes applied to the original object (called the "master") are automatically applied to the copy (called the "clone"). For example, if you change the master's fill, the clone's fill will change as well. If you change the attributes of the clone, the attribute you change will no longer depend on the master's attributes. For example, after you change a clone's fill, its fill will no longer change when you change the master's fill. Likewise, if you stretch a clone, it will no longer stretch when you stretch its master.

Example

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.CloneObject
```

The above example creates an ellipse, then makes a clone.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

ConvertToCurves (DRAW)

.ConvertToCurves

This command converts the selected polygon, rectangle, ellipse, or text object to a series of curves you can shape with the Shape tool.

Example

```
.CreateRectangle 500000, -750000, -500000, 750000, 0
.ConvertToCurves
```

The above example converts the selected rectangle to a curve object.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

CreateEllipse (DRAW)

.CreateEllipse .Top = *long*, .Left = *long*, .Bottom = *long*, .Right = *long*, .StartAngle = *long*, .EndAngle = *long*, .Arc = *boolean*

This command is used to draws ellipses and circles.

Syntax	Description
.Top	Specifies the Y-coordinate of the upper-left corner of the bounding rectangle of the ellipse in tenths of a micron, relative to the center of the page.
.Left	Specifies the X-coordinate of the upper-left corner of the bounding rectangle of the ellipse in tenths of a micron, relative to the center of the page.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the bounding rectangle of the ellipse in tenths of a micron, relative to the center of the page.
.Right	Specifies the X-coordinate of the lower-right corner of the bounding rectangle of the ellipse in tenths of a micron, relative to the center of the page.
.StartAngle	If .CreateEllipse is used to create an arc, .StartAngle specifies the starting angle in degrees.
.EndAngle	If .CreateEllipse is used to create an arc, .EndAngle specifies the end angle, in degrees.
.Arc	Specifies whether to draw the ellipse as a pie or an arc. Set to TRUE (-1) to turn the ellipse into a pie. Set to FALSE (0) to draw the ellipse as an arc.

Note

- You can use the ANGLECONVERT function to specify angle measurements

Example

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
```

The above example creates an ellipse.

```
for count% = 1 to 4
.CreateEllipse 1500000-(250000 * count), -1200000 +( 200000* count), 750000 - ( 200000* count),
-500000+( 200000* count), 0, 0, 0
next count
```

The above example creates 4 ellipses.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

CreateRectangle (DRAW)

.CreateRectangle .Top = *long*, .Left = *long*, .Bottom = *long*, .Right = *long*, .CornerRadius = *long*

This command draws rectangles and squares.

Syntax	Description
.Top	Specifies the Y-coordinate of the upper-left corner of the rectangle in tenths of a micron, relative to the center of the page.
.Left	Specifies the X-coordinate of the upper-left corner of the rectangle in tenths of a micron, relative to the center of the page.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the rectangle in tenths of a micron, relative to the center of the page.
.Right	Specifies the X-coordinate of the lower-right corner of the rectangle in tenths of a micron, relative to the center of the page.
.CornerRadius	Specifies the radius used to create the rounded corners in tenths of a micron.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
```

The above example creates a rectangle.

```
FOR count% = 1 TO 8
.CreateRectangle 1500000-(250000 * count), -1200000 +( 200000* count), 750000 - ( 200000*
count), -500000+( 200000* count), 0
NEXT count
```

The above example creates 8 rectangles.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

CreateSymPolygon (DRAW)

.CreateSymPolygon .lTop = *long*, .lLeft = *long*, .lBottom = *long*, .lRight = *long*, .lSides = *long*, .lSubpaths = *long*, .lComplexity = *long*, .bStar = *boolean*, .lStarComplexity = *long*, .lMaxComplexity = *long*

This command creates a polygon or star.

Syntax	Description
.lTop	Specifies the coordinate of the top of the shape in tenths of a micron, relative to the center of the page.
.lLeft	Specifies the coordinate of the left of the shape in tenths of a micron, relative to the center of the page.
.lBottom	Specifies the coordinate of the bottom of the shape in tenths of a micron, relative to the center of the page.
.lRight	Specifies the coordinate of the right of the shape in tenths of a micron, relative to the center of the page.
.lSides	Specifies the number of sides (from 3 to 500).
.lSubpaths	Specifies the number of subpaths in a polygon. If both the number of subpaths and the complexity are set to 1 then the polygon will be a simple polygon. If either of these values are greater than 1 then the polygon becomes a star. The relationship between the number of subpaths and the complexity is represented by the Star/Polygon button and the Sharpness slider on the Polygon Property Bar. Appropriate values for each of these parameters change depending on the number of sides of the polygon.
.lComplexity	Specifies the complexity of a polygon. If both the number of subpaths and the complexity are set to 1 then the polygon will be a simple polygon. If either of these values are greater than 1 then the polygon becomes a star. The relationship between the number of subpaths and the complexity is represented by the Star/Polygon button and the Sharpness slider on the Polygon Property Bar. Appropriate values for each of these parameters change depending on the number of sides of the polygon. If there is only 1 subpath and the complexity is set to 2, then you will always create a simple star. For more complex stars, we recommend you experiment with the recorder to determine the appropriate values.
.bStar	Set to TRUE (-1) to enable the StarComplexity parameter. Set to FALSE (0) to ignore this parameter. This parameter must be TRUE if you want to create a star-shaped polygon.
.lStarComplexity	Specifies the distance that the start node is from the center of the shape. If this parameter is set to 0, the start node will remain at the outer edge of the shape. The closer this parameter's value is to the .lMaxComplexity value, the closer the start node is to the center of the shape. This parameter is equivalent to the Sharpness slider for polygons (as opposed to stars).
.lMaxComplexity	Specifies the maximum value for star complexity.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

DeleteObject (DRAW)

.DeleteObject

This command deletes selected objects.

Example

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.CreateRectangle 750000, -750000, 0, 0, 0
.DeleteObject
```

The above example deletes the selected object. Since the rectangle is the last object created, it is selected and gets deleted.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

DistributeObjects (DRAW)

.DistributeObjects .HorizontalDistribution = *long*, .VerticalDistribution = *long*, .ObjectOrPageExtents = *long*

This command distributes selected objects.

Syntax	Description
.HorizontalDistribution	Specifies the type of horizontal distribution. 0 = None 1 = Right edges of object 2 = Left edges of object 3 = Center edges of object 4 = Space between objects
.VerticalDistribution	Specifies the type of vertical distribution. 0 = None 1 = Top edges of object 2 = Bottom edges of object 3 = Center edges of object 4 = Space between objects
.ObjectOrPageExtents	Specifies the type of distribution. 0 = Extent of Selection 1 = Extent of Page

Example

```
.SelectAllObjects
.DistributeObjects 3, 3, 1
```

The above example distributes the selected objects to the center of the page.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

DrawCurveClosePath (DRAW)

.DrawCurveClosePath

This command closes the path on the last node when drawing curves in Freehand mode.

Note

- The .DrawCurveClosePath command must be in a contiguous block of one or more DrawCurve commands. The first DrawCurve command in the block must be preceded by the .BeginDrawCurve command, and the last must be followed by the .EndDrawCurve command. The DrawCurve commands include:

  .DrawCurveClosePath
  .DrawCurveCurveTo
  .DrawCurveLineTo
  .DrawCurveMoveTo

Example

```
.BeginDrawCurve -500000, 1000000
.DrawCurveCurveTo 500000, 500000, 1000000 ,-500000, -500000, -500000
.DrawCurveClosePath
.EndDrawCurve
```

The above example draws an object in the shape of an uppercase "D".

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

DrawCurveCurveTo (DRAW)

.DrawCurveCurveTo .X1 = *long*, .Y1 = *long*, .X2 = *long*, .Y2 = *long*, .XEnd = *long*, .YEnd = *long*

This command sets a node in a curve drawn in Freehand mode.

Syntax	Description
.X1	Specifies the X-coordinate for a control point in tenths of a micron, relative to the center of the page. The control point is used with the node that was created in a preceding BeginDrawCurve or DrawCurveCurveTo command.
.Y1	Specifies the Y-coordinate for a control point in tenths of a micron, relative to the center of the page. The control point is used with the node that was created in a preceding BeginDrawCurve or DrawCurveCurveTo command.
.X2	Specifies the X-coordinate for a control point in tenths of a micron, relative to the center of the page. The control point is used with the node that is specified in the current DrawCurveCurveTo command.
.Y2	Specifies the Y-coordinate for a control point in tenths of a micron, relative to the center of the page. The control point is used with the node that is specified in the current DrawCurveCurveTo command.
.XEnd	Specifies the X-coordinate of a node of the curve in tenths of a micron, relative to the center of the page.
.YEnd	Specifies the X-coordinate of a node of the curve in tenths of a micron, relative to the center of the page.

Note

- The .DrawCurveCurveTo command must be in a contiguous block of one or more DrawCurve commands. The first DrawCurve command in the block must be preceded by the .BeginDrawCurve command, and the last must be followed by the .EndDrawCurve command. The DrawCurve commands include:

  .DrawCurveClosePath
  .DrawCurveCurveTo
  .DrawCurveLineTo
  .DrawCurveMoveTo

Example

```
.BeginDrawCurve -500000, 1000000
.DrawCurveCurveTo 500000, 500000, 1000000 ,-500000, -500000, -500000
.DrawCurveCurveTo 600000, 600000, 1100000 ,-600000, -600000, -600000
.EndDrawCurve
```

The above example draws a curve.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

DrawCurveLineTo (DRAW)

.DrawCurveLineTo .X = *long*, .Y = *long*

This command sets the coordinates when drawing continuous curves in Freehand mode.

Syntax	Description
.X	Specifies the X-coordinate of the next node of the curve in tenths of a micron, relative to the center of the page.
.Y	Specifies the Y-coordinate of the next node of the curve in tenths of a micron, relative to the center of the page.

 **Note**

- The .DrawCurveLineTo command must be in a contiguous block of one or more DrawCurve commands. The first DrawCurve command in the block must be preceded by the .BeginDrawCurve command, and the last must be followed by the .EndDrawCurve command. The DrawCurve commands include:

  .DrawCurveClosePath
  .DrawCurveCurveTo
  .DrawCurveLineTo
  .DrawCurveMoveTo

Example

```
.BeginDrawCurve -500000, 1000000
.DrawCurveLineTo 500000, -500000
.EndDrawCurve
```

The above example demonstrates the DrawCurve commands.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

DrawCurveMoveTo (DRAW)

.DrawCurveMoveTo .X = *long*, .Y = *long*

This command sets the coordinates when drawing non-continuous curves in Freehand mode.

Syntax	Description
.X	Specifies the X-coordinate of the point to move to without drawing in tenths of a micron, relative to the center of the page.
.Y	Specifies the Y-coordinate of the point to move to without drawing in tenths of a micron, relative to the center of the page.

 **Note**

- The .DrawCurveMoveTo command must be in a contiguous block of one or more DrawCurve commands. The first DrawCurve command in the block must be preceded by the .BeginDrawCurve command, and the last must be followed by the .EndDrawCurve command. The DrawCurve commands include:

  .DrawCurveClosePath
  .DrawCurveCurveTo
  .DrawCurveLineTo
  .DrawCurveMoveTo

Example

```
.BeginDrawCurve -500000, 1000000
.DrawCurveLineTo 500000, -500000
.DrawCurveMoveTo -500000, -500000
.DrawCurveLineTo 500000, 1000000
.EndDrawCurve
```

The above example demonstrates the DrawCurve commands.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

DuplicateObject (DRAW)

.DuplicateObject

This command adds a copy of the selected object(s) to the current drawing. By default, the copy is placed on top of the original, offset up and to the right. It is also selected automatically.

Example

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.ApplyFountainFill 2, -50, -50, 900, 20, 20, 2, 0
.SetFountainFillColor 0, 5, 0, 255, 0, 0
.SetFountainFillColor 100, 5, 0, 0, 255, 0
.DuplicateObject
```

The above example creates an ellipse, fills it with a two color fountain fill, then duplicates it and fills the duplicate.

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.DuplicateObject
.ApplyFountainFill 2, -50, -50, 900, 20, 20, 2, 0
.SetFountainFillColor 0, 5, 0, 255, 0, 0
.SetFountainFillColor 100, 5, 0, 0, 255, 0
```

The above example creates an ellipse without filling it. The ellipse is then duplicated and the duplicate is filled with a two-color fountain fill.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

EndDrawBezier (DRAW)

.EndDrawBezier

This command ends a set of bezier creation commands that began with the .BeginDrawBezier command.

Note

- The bezier commands include:

  .AddBezierPoint
  .MoveBezierControl

Example

```
.InitBezierTool
.BeginDrawBezier -3085992, 163280, FALSE
.MoveBezierControl -1649128, 1142960, FALSE
.AddBezierPoint 146952, -277576, FALSE, FALSE
.MoveBezierControl -326560, -1453192, FALSE
.EndDrawBezier
```

The above example draws a simple bezier curve.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

EndDrawCurve (DRAW)

.EndDrawCurve

This command ends a set of curve creation commands that began with the .BeginDrawCurve command.

Note

- The DrawCurve commands include:

  .DrawCurveClosePath
  .DrawCurveCurveTo
  .DrawCurveLineTo
  .DrawCurveMoveTo

Example

```
.BeginDrawCurve -500000, 1000000
.DrawCurveLineTo 500000, -500000
.EndDrawCurve
```

The above example demonstrates the DrawCurve commands.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

EndDrawFreehand (DRAW)

.EndDrawFreehand .lStraightTightness = *long*, .lCornerTightness = *long*, .lCornerThreshold = *long*, .lSnapTightness = *long*

This command ends a set of frehand drawing commands that began with the .BeginDrawCurve command.

Syntax	Description
.lStraightTightness	Sets the amount the curve can vary from a straight path and still be treated as straight. The higher the value, the less accurate the line needs to be. The valid range is from 1 to 10 pixels.
.lCurveTightness	Determines how closely the curve will match the position of each point. The lower the number, the more accurate the match. The valid range is from 1 to 10 pixels.
.lCornerThreshold	Sets the limit at which each corner node is cusped (as opposed to smooth). A node is more likely to be cusped if the value is lower. The valid range is from 1 to 10 pixels.
.lSnapTightness	Determines how close two end nodes must be to join automatically. The valid range is from 1 to 10 pixels.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

GetObjectsCDRStaticID (DRAW)

ReturnValue& = .GetObjectsCDRStaticID()

This function returns the CDRStaticID of the selected object. If more than one object is selected, the function returns the CDRStaticID of the last selected object.

Syntax	Description
ReturnValue&	Returns the CDRStaticID of the selected object.

Note

- Every object you create has a unique CDRStaticID in a document.

Example

```
.CreateRectangle 750000, -600000, 250000, -100000, 0
IDRect& = .GetObjectsCDRStaticID()
.SelectObjectOfCDRStaticID IDRect&
```

The above example demonstrates object selection using the object's CDRStaticID.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

InitBezierTool (DRAW)

.InitBezierTool

This command must preceed the .BeginDrawBezier command.

Example

```
.InitBezierTool
.BeginDrawBezier -3085992, 163280, FALSE
.MoveBezierControl -1649128, 1142960, FALSE
.AddBezierPoint 146952, -277576, FALSE, FALSE
.MoveBezierControl -326560, -1453192, FALSE
.EndDrawBezier
```

The above example draws a simple bezier curve.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} **Related Topics**

MoveBezierControl (DRAW)

.MoveBezierControl .lX = *long*, .lY = *long*, .bConstrain = *boolean*

This command controls the position of bezier node control points created with the .BeginDrawBezier and .AddBezierPoint commands.

Syntax	Description
.lX	Specifies the X-coordinate of the control point's position in tenths of a micron, relative to the center of the page.
.lY	Specifies the Y-coordinate of the control point's position in tenths of a micron, relative to the center of the page.
.bConstrain	Set to TRUE (-1) to use the constrain angle when positioning the control points.

Example

```
.InitBezierTool
.BeginDrawBezier -3085992, 163280, FALSE
.MoveBezierControl -1649128, 1142960, FALSE
.AddBezierPoint 146952, -277576, FALSE, FALSE
.MoveBezierControl -326560, -1453192, FALSE
.EndDrawBezier
```

The above example draws a simple bezier curve.

{button ,AL(`OVR1 Object creation commands;',0,"Defaultoverview",)} Related Topics

Node editing commands

BeginEditObject (DRAW)

.BeginEditObject

This command initializes a block of node editing commands that includes one or more instances of the .EditObjectCommand and ends with the .EndEditObject command.

{button ,AL(`OVR1 Node editing commands;',0,"Defaultoverview",)} **Related Topics**

EditObjectCommand (DRAW)

.EditObjectCommand .lCmd = *long*, .lX = *long*, .lY = *long*, .lKey = *long*, .bAddToSelection = *boolean*

This command lets you change the shape of objects by editing their nodes. This command must be part of a block of commands that begins with the .BeginEditObject command and ends with the .EndEditObject command.

Syntax	Description
.lCmd	0 = This value is equivalent to releasing the mouse button. Use with the .lX parameter and the .lY parameter. 1 = This value is equivalent to pressing one of the nudge or super-nudge keys. Use the .lKey parameter to indicate which nudge key is used. 2 = This value is equivalent to pressing a key on the keyboard. Use the .lKey parameter to indicate which key is used. 3 = This value is equivalent to pressing down the mouse button. Use with the .lX parameter and the .lY parameter. 4 = This value is equivalent to the add node option. 5 = This value is equivalent to the delete node option. 6 = This value is equivalent to the join nodes option. 7 = This value is equivalent to the break path option. 8 = This value makes a node cusped. 9 = This value makes a node smooth. 10 = This value makes a segment straight. 11 = This value makes a segment curved. 12 = This value makes a node symmetrical. 13 = This value is equivalent to the auto-reduce nodes option. 14 = This value is equivalent to the extract subpath option. 16 = This value toggles the elastic option.
.lX	Specifies the X-coordinate of a node in tenths of a micron, relative to the center of the page.
.lY	Specifies the Y-coordinate of a node in tenths of a micron, relative to the center of the page.
.lKey	If .lCmd = 1 0 = nudge left 1 = nudge right 2 = nudge up 3 = nudge down 4 = super-nudge left 5 = super-nudge right 6 = super-nudge up 7 = super-nudge down If .lCmd = 2 0 = HOME 1 = END 2 = TAB 3 = ESC
.bAddToSelection	Set to TRUE (-1) to select multiple nodes (equivalent to pressing SHIFT).

{button ,AL(`OVR1 Node editing commands;',0,"Defaultoverview",)} Related Topics

EndEditObject (DRAW)

.EndEditObject

This command ends a block of node editing commands that includes one or more instances of the .EditObjectCommand and begins with the .BeginEditObject command.

{button ,AL(`OVR1 Node editing commands;',0,"Defaultoverview",)} **Related Topics**

CloseCurve (DRAW)

.CloseCurve

This command closes the selected open path.

{button ,AL(`OVR1 Node editing commands;',0,"Defaultoverview",)} **Related Topics**

Symbols commands

DropSymbol (DRAW)

.DropSymbol .SymbolLibrary = *string*, .SymbolNumber = *long*, .Tile = *boolean*, .XPosOrGridSize = *long*, .YPosOrGridSize = *long*, .ProportionalSizing = *boolean*, .SymbolSize = *long*

This command positions the specified symbol at the defined position or the specified grid position.

Syntax	Description
.SymbolLibrary	Specifies the name of the Symbol Library. Refer to the Symbols dialog box for more details.
.SymbolNumber	Specifies the Symbol Index Number, which identifies the selected symbol. Refer to the Symbols dialog box for more details.
.Tile	Set to TRUE (-1) to create a pattern from the selected symbol that fills the page. Set to FALSE (0) to disable this option. Note that the tiled symbols are clones of the upper left symbol.
.XPosOrGridSize	Specifies the X-coordinate or grid position at which to place the symbol, in tenths of a micron.
.YPosOrGridSize	Specifies the Y-coordinate or grid position at which to place the symbol, in tenths of a micron.
.ProportionalSizing	Set to TRUE (-1) to enable proportional sizing of the symbol. Set to FALSE (0) to disable this option.
.SymbolSize	Specifies the size of the symbol in tenths of a micron. The symbol can be resized after it's been added to your drawing.

Example

```
.DropSymbol "Animals 1", 42, 0, 0, 0, 0, 1000000
```

The above example places a kangaroo symbol in the center of the page.

Arrange commands

AlignObjects (DRAW)

.AlignObjects .HorizontalAlignment = *long*, .VerticalAlignment = *long*

This command aligns selected objects.

Syntax	Description
.HorizontalAlignment	Specifies the type of horizontal alignment. 0 = None 1 = Right 2 = Left 3 = Center
.VerticalAlignment	Specifies the type of vertical alignment. 0 = None 1 = Top 2 = Bottom 3 = Center

Example

```
.SelectAllObjects
.AlignObjects 2, 0
```

The above example horizontally aligns the selected objects to the left edge of the page.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

AlignToCenterOfPage (DRAW)

.AlignToCenterOfPage .HorizontalAlignment = *long*, .VerticalAlignment = *long*

This command aligns selected objects to the center of the page.

Syntax	Description
.HorizontalAlignment	Specifies the type of horizontal alignment. 0 = None 1 = Right 2 = Left 3 = Center
.VerticalAlignment	Specifies the type of vertical alignment. 0 = None 1 = Top 2 = Bottom 3 = Center

Example

```
.SelectAllObjects
.AlignToCenterOfPage 0, 3
```

The above example vertically aligns all objects to the center of the page.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

AlignToGrid (DRAW)

.AlignToGrid .HorizontalAlignment = *long*, .VerticalAlignment = *long*

This command aligns the selected objects to the gridpoint nearest to the edge of the selection.

Syntax	Description
.HorizontalAlignment	Specifies the type of horizontal alignment. 0 = None 1 = Right 2 = Left 3 = Center
.VerticalAlignment	Specifies the type of vertical alignment. 0 = None 1 = Top 2 = Bottom 3 = Center

Example

```
.SelectAllObjects
.AlignToGrid 1, 0
```

The above example horizontally aligns all objects to a gridpoint, nearest to the left edge of the selection.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

ApplyToDuplicate (DRAW)

ReturnValue = ObjDRAW.ApplyToDuplicate
ObjDRAW.ApplyToDuplicate = Value

This function is a property and will only work when excuted in a programming language that supports properties (e.g.. Visual Basic). Corel SCRIPT doesn't support properties. Use the .SetApplyToDuplicate command in Corel SCRIPT to open or close a block of object duplicating commands (commands that create a duplicate of the selected object if they occur after a **.SetApplyToDuplicate TRUE** command).

In a programming language that supports properties, this function either returns a value that indicates whether a block of object duplicating commands is open or not (i.e.,**ReturnValue = ObjDRAW.ApplyToDuplicate**), or this function opens or closes a block of object duplicating commands (**ObjDRAW.ApplyToDuplicate = Value**).

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

BreakApart (DRAW)

.BreakApart

This command converts an object made up of multiple subpaths into individual curve objects.

Example

```
.BreakApart
```

The above example breaks apart the selected object into individual curve objects.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

Combine (DRAW)

.Combine

This command combines the selected curve or line segments into a single object. If you use Combine on rectangles, ellipses, polygons, or text, CorelDRAW converts them to curves before converting them into a single curve object. However, when text is combined with other text it is not converted to curves; it is converted to larger blocks of text.

Example

```
for count% = 1 to 4
.CreateEllipse 1500000-(250000 * count), -1200000 +( 200000* count), 750000 - ( 200000* count),
-500000+( 200000* count), 0, 0, 0
next count
.SelectAllObjects
.Combine
.ApplyUniformFillColor 2, 0, 255, 0, 0
```

The above example creates 4 ellipses, then combines them before applying a fill.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

CreateGuidelineUsingAngle (DRAW)

.CreateGuidelineUsingAngle .lIntersectPointX = *long*, .lIntersectPointY = *long*, .lAngle = *long*, .bLocked = *boolean*

This command creates a guideline at a specific location using an intersection point and an angle to specify where to put the new guideline.

Syntax	Description
.lIntersectPointX	Specifies the X-coordinate of the point in tenths of a micron, relative to the center of the page.
.lIntersectPointY	Specifies the Y-coordinate of the point in tenths of a micron, relative to the center of the page.
.lAngle	Specifies the angle of the guideline in millionths of a degree (e.g., 5000000 = 5 degrees).
.bLocked	Set to TRUE (-1) to lock the guideline. Set to FALSE (0) to leave the guideline unlocked.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

CreateGuidelineUsingTwoPoints (DRAW)

.CreateGuidelineUsingTwoPoints .lPoint1X = *long*, .lPoint1Y = *long*, .lPoint2X = *long*, .lPoint2Y = *long*, .bLocked = *boolean*

This command creates a guideline at a specific location using two sets of coordinates to place the guidleine.

Syntax	Description
.lPoint1X	Specifies the X-coordinate of the first point in tenths of a micron, relative to the center of the page.
.lPoint1Y	Specifies the Y-coordinate of the first point in tenths of a micron, relative to the center of the page.
.lPoint2X	Specifies the X-coordinate of the second point in tenths of a micron, relative to the center of the page.
.lPoint2Y	Specifies the Y-coordinate of the second point in tenths of a micron, relative to the center of the page.
.bLocked	Set to TRUE (-1) to lock the guideline. Set to FALSE (0) to leave the guideline unlocked.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

DeleteGuidelineByIndex (DRAW)

.DeleteGuidelineByIndex .lIndex = *long*

This command deletes a specific guideline.

Syntax	Description
.lIndex	Specifies the guidline ID. Use the .GetNumberOfGuidelines command to get a guideline's ID.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

DeleteGuidelineUsingAngle (DRAW)

.DeleteGuidelineUsingAngle .lIntersectX = *long*, .lIntersectY = *long*, .lAngle = *long*

This command deletes a specific guideline based on a set of coordinates and an angle.

Syntax	Description
.lIntersectPointX	Specifies the X-coordinate of the point in tenths of a micron, relative to the center of the page.
.lIntersectPointY	Specifies the Y-coordinate of the point in tenths of a micron, relative to the center of the page.
.lAngle	Specifies the angle of the guideline in millionths of a degree (e.g., 5000000 = 5 degrees).

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

DeleteGuidelineUsingTwoPoints (DRAW)

.DeleteGuidelineUsingTwoPoints .lPoint1X = *long*, .lPoint1Y = *long*, .lPoint2X = *long*, .lPoint2Y = *long*

This command deletes a specific guideline using two sets of coordinates.

Syntax	Description
.lPoint1X	Specifies the X-coordinate of the first point in tenths of a micron, relative to the center of the page.
.lPoint1Y	Specifies the Y-coordinate of the first point in tenths of a micron, relative to the center of the page.
.lPoint2X	Specifies the X-coordinate of the second point in tenths of a micron, relative to the center of the page.
.lPoint2Y	Specifies the Y-coordinate of the second point in tenths of a micron, relative to the center of the page.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

GetGuidelineInformation (DRAW)

.GetGuidelineInformation .lIndex = *long*, .plPoint1X = *long**, .plPoint1Y = *long**, .plPoint2X = *long**, .plPoint2Y = *long**, .plInterceptX = *long**, .plInterceptY = *long**, .plAngle = *long**, .pbLocked = *boolean**

This command retrieves all the information available about a specific guideline. The .plPoint parameters are equivalent to the coordinates used in the .CreateGuidlineUsingTwoPoints command, and the .plIntercept and .plAngle parameters are equivalent to the parameters used in the .CreateGuidelineUsingAngle command.

Syntax	Description
.lIndex	Specifies the guidline ID. Use the .GetNumberOfGuidelines command to get a guideline's ID.
.plPoint1X	Returns the X-coordinate of the first point in tenths of a micron, relative to the center of the page.
.plPoint1Y	Returns the Y-coordinate of the first point in tenths of a micron, relative to the center of the page.
.plPoint2X	Returns the X-coordinate of the second point in tenths of a micron, relative to the center of the page.
.plPoint2Y	Returns the Y-coordinate of the second point in tenths of a micron, relative to the center of the page.
.plInterceptX	Returns the X-coordinate of the point that relates to .plAngle in tenths of a micron, relative to the center of the page.
.plInterceptY	Returns the Y-coordinate of the point that relates to .plAngle in tenths of a micron, relative to the center of the page.
.plAngle	Returns the angle of the guideline in millionths of a degree (e.g., 5000000 = 5 degrees).
.pbLocked	Returns TRUE (-1) if guidelines are locked. Returns FALSE (0) if the guidelines are unlocked.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

GetNumberOfGuidelines (DRAW)

ReturnValue& = .GetNumberOfGuidelines()

This function return the number of guidelines. You can use this function to determine the index of a guidline. The range of possible guideline index values is 0 to the number of guidelines minus 1.

Syntax	Description
ReturnValue&	Returns the number of guidelines.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

Group (DRAW)

.Group

This command groups all selected objects together to allow them to be selected and manipulated as a single object.

Example

```
for count% = 1 to 4
.CreateEllipse 1500000-(250000 * count), -1200000 +( 200000* count), 750000 - ( 200000* count),
-500000+( 200000* count), 0, 0, 0
next count
.SelectAllObjects
.Group
.ApplyUniformFillColor 5, 0, 0, 255, 0
```

The above example groups the four ellipses together so that they are treated as one object, and applies a blue uniform fill to all four.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

Intersection (DRAW)

.Intersection

This command creates a new object using the area common to two or more overlapping objects. Intersection joins their paths at the points where they intersect. The resulting curve object assumes the fill and outline attributes of the last selected object.

Example

```
.SelectAllObjects
.Intersection
```

The above example selects all objects and creates a new object(s) using the area common to overlapping objects.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

LockGuidelineByIndex (DRAW)

.LockGuidelineByIndex .lIndex = *long*

This command locks a specific guideline.

Syntax	Description
.lIndex	Specifies the guidline ID. Use the .GetNumberOfGuidelines command to get a guideline's ID.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

MoveGuidelineUsingAngleByIndex (DRAW)

.MoveGuidelineUsingAngleByIndex .lIndex = *long*, .lInterceptX = *long*, .lInterceptY = *long*, .lAngle = *long*, .bLocked = *boolean*

This command moves a specific guideline to a location using an intersection point and an angle to specify where to put the new guideline.

Syntax	Description
.lIndex	Specifies the guidline ID. Use the .GetNumberOfGuidelines command to get a guideline's ID.
.lInterseptPointX	Specifies the X-coordinate of the point in tenths of a micron, relative to the center of the page.
.lInterseptPointY	Specifies the Y-coordinate of the point in tenths of a micron, relative to the center of the page.
.lAngle	Specifies the angle of the guideline in millionths of a degree (e.g., 5000000 = 5 degrees).
.bLocked	Set to TRUE (-1) to lock the guideline. Set to FALSE (0) to leave the guideline unlocked.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

MoveGuidelineUsingTwoPointsByIndex (DRAW)

.MoveGuidelineUsingTwoPointsByIndex .lIndex = *long*, .lPoint1X = *long*, .lPoint1Y = *long*, .lPoint2X = *long*, .lPoint2Y = *long*, .bLocked = *boolean*

This command moves a specific guideline to a location using two sets of coordinates to place the guidleine.

Syntax	Description
.lIndex	Specifies the guidline ID. Use the .GetNumberOfGuidelines command to get a guideline's ID.
.lPoint1X	Specifies the X-coordinate of the first point in tenths of a micron, relative to the center of the page.
.lPoint1Y	Specifies the Y-coordinate of the first point in tenths of a micron, relative to the center of the page.
.lPoint2X	Specifies the X-coordinate of the second point in tenths of a micron, relative to the center of the page.
.lPoint2Y	Specifies the Y-coordinate of the second point in tenths of a micron, relative to the center of the page.
.bLocked	Set to TRUE (-1) to lock the guideline. Set to FALSE (0) to leave the guideline unlocked.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

OrderBackOne (DRAW)

.OrderBackOne

This command rearranges the stacking order by moving the selected object back one position.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyUniformFillColor 5,255,0,0,0
.CreateEllipse -450000, -700000, 450000, 700000, 0, 0, 0
.ApplyUniformFillColor 5,0,0,250,0
.OrderBackOne
```

The above example creates a rectangle and then creates an ellipse on top of the rectangle. The ellipse, still selected, is ordered back one position in the drawing.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

OrderForwardOne (DRAW)

.OrderForwardOne

This command rearranges the stacking order by moving the selected object up one position.

Example

```
.SelectObjectOfCDRStaticID Six&
.OrderForwardOne
```

The above example orders the selected object forward one position.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

OrderReverseOrder (DRAW)

.OrderReverseOrder

This command reverses the stacking order of the selected object(s).

Example

```
.SelectAllObjects
.OrderReverseOrder
```

The above example reverses the order of all the objects.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

OrderToBack (DRAW)

.OrderToBack

This command rearranges the stacking order by moving the selected object to the back of the screen. Areas of the object overlapped by other objects with fills are "knocked out" so that they will not print.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyUniformFillColor 5,255,0,0,0
.CreateEllipse -450000, -700000, 450000, 700000, 0, 0, 0
.ApplyUniformFillColor 5,0,0,250,0
.OrderToBack
```

The above example creates a rectangle and then creates an ellipse on top of the rectangle. The ellipse, still selected, is ordered to the back of the drawing.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

OrderToFront (DRAW)

.OrderToFront

This command rearranges the stacking order by moving the selected object to the front of the layer.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyUniformFillColor 5,255,0,0,0
.CreateEllipse -450000, -700000, 450000, 700000, 0, 0, 0
.ApplyUniformFillColor 5,0,0,250,0
.SelectPreviousObject 0
.OrderToFront
```

The above example creates a rectangle and then creates an ellipse on top of the rectangle. The rectangle is then selected and ordered to the front of the drawing.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

RemoveAllGuidelines (DRAW)

.RemoveAllGuidelines

This command removes all guidelines.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

Separate (DRAW)

.Separate

This command separates original objects from intermediate shapes.

Example

```
.Separate
```

The above example separates a combined object into its individual component object(s).

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

Trim (DRAW)

.Trim

This command lets you trim selected objects. Trimming two or more overlapping objects reshapes the last object selected. Trimming separates the paths at points where the objects overlap. Initially, the trimmed object may appear no different than it did before trimming. However, closer inspection will show that new nodes appear where the object was trimmed. Move the trimmed objects apart to see the full effect of the trim.

Example

```
.SelectAllObjects
.Trim
```

The above example trims the selected objects.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

Ungroup (DRAW)

.Ungroup

This command breaks up the selected group into its individual objects. If you have more than one sublevel of grouping, Ungroup breaks up one level of grouping at a time.

Example

```
.Ungroup
```

The above example breaks up the grouped object into its individual object components.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

UnlockGuidelineByIndex (DRAW)

.UnlockGuidelineByIndex .lIndex = *long*

This command unlocks a specific guideline.

Syntax	Description
.lIndex	Specifies the guidline ID. Use the .GetNumberOfGuidelines command to get a guideline's ID.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} Related Topics

Weld (DRAW)

.Weld

This command joins overlapping objects at points where their paths intersect. Although not necessarily apparent in editable preview, welding also removes sections of the path between those intersect points. The resulting curve object assumes the fill and outline attributes of the bottom object of the selected group of objects. If you marquee-select the objects, CorelDRAW will outline and fill the welded object with the attributes of the most recently created object.

Example

```
.SelectAllObjects
.Weld
```

The above example welds the selected object group.

{button ,AL(`OVR1 Arrange commands;',0,"Defaultoverview",)} **Related Topics**

Transformation commands

GetPosition (DRAW)

.GetPosition .XPos = *long**, .Ypos = *long**

This function returns the position coordinates of a selected object's reference point. If more than one object is selected, the function returns the position coordinates of the last selected object.

Syntax	Description
.XPos	Returns the X-coordinate of the selected object's reference point in tenths of a micron, relative to the center of the page. An object's default reference point is the upper-left corner.
.YPos	Returns the Y-coordinate of the selected object's reference point in tenths of a micron, relative to the center of the page. An object's default reference point is the upper-left corner.

Example

```
.CreateRectangle 1000000, 750000, 500000, 100005, 0
id& = .GetObjectsCDRStaticID()
.GetPosition XPos&, YPos&
MESSAGE "Horizontal"+STR(XPos&)
MESSAGE "Vertical"+STR(YPos&)
```

The above example creates a rectangle then displays the coordinates of the upper-left corner in message boxes.

```
.CreateRectangle 1000000, 750000, 500000, 100005, 0
id& = .GetObjectsCDRStaticID()
.SetReferencePoint 3
.GetPosition XPos&, YPos&
MESSAGE "Horizontal"+STR(XPos&)
MESSAGE "Vertical"+STR(YPos&)
```

The above example creates a rectangle then displays the coordinates of the upper-right corner in message boxes. The upper-right coordinates are used because the selected object's reference point was changed with the .SetReferencePoint command.

{button ,AL(`OVR1 Transformation commands;',0,"Defaultoverview",)} **Related Topics**

GetSize (DRAW)

.GetSize .XSize = *long**, .YSize = *long**

This function returns the size attributes of a selected object. If more than one object is selected, the function returns the size attributes of the last selected object.

Syntax	Description
.XSize	Returns the horizontal size of the selected object, in tenths of a micron.
.YSize	Returns the vertical size of the selected object, in tenths of a micron.

Example

```
.CreateRectangle 1000000, 750000, 450000, 100000, 0
id& = .GetObjectsCDRStaticID()
.GetSize XSize&, YSize&
MESSAGE "Horizontal"+STR(XSize&)
MESSAGE "Vertical"+STR(YSize&)
```

The above example returns the size of the selected rectangle and displays the width and height (in tenths of a micron) in message boxes.

{button ,AL(`OVR1 Transformation commands;',0,"Defaultoverview",)} Related Topics

MoveCenter (DRAW)

.MoveCenter .lAnchorID = *long*

This command moves the selected object's center of rotation.

Syntax	Description
.lAnchorID	Specifies the reference point of the object to be skewed. 1 = Upper-right 2 = Upper-middle 3 = Upper-left 4 = Middle-left 5 = Lower-left 6 = Lower-middle 7 = Lower-right 8 = Middle-right 9 = Center

{button ,AL(`OVR1 Transformation commands;',0,"Defaultoverview",)} **Related Topics**

MoveObject (DRAW)

.MoveObject .XDelta = *long*, .YDelta = *long*

This command repositions the selected object to the specified location.

Syntax	Description
.XDelta	Specifies the distance the object is to be moved along the X-axis in tenths of a micron.
.YDelta	Specifies the distance the object is to be moved along the Y-axis in tenths of a micron.

Example

```
.SetPageSize 2159000, 2794000
.CreateRectangle 500000, -750000, -500000, 750000, 0
.MoveObject 250000, -750000
```

The above example creates a rectangle, then moves it to the bottom right corner of an 8.5 by 11 inch page.

{button ,AL(`OVR1 Transformation commands;',0,"Defaultoverview",)} Related Topics

ResetTransfo (DRAW)

.ResetTransfo

This command clears all tranformations.

{button ,AL(`OVR1 Transformation commands;',0,"Defaultoverview",)} **Related Topics**

RotateObject (DRAW)

.RotateObject .Angle = *long*, .UseObjectsCenter = *boolean*, .XCenter = *long*, .YCenter = *long*

This command lets you rotate the selected object.

Syntax	Description
.Angle	Specifies the angle of rotation of the selected object, expressed in millionths of degrees. Negative values rotate the object clockwise from its current position; positive values rotate it counterclockwise. e.g., 45 degrees clockwise = -45000000
.UseObjectsCenter	Set to TRUE (-1) to enable rotation around the center of the object. Set to FALSE (0) to disable this option.
.XCenter	Specifies the logical X-coordinate of the center of the object to be rotated in tenths of a micron, relative to the center of the page.
.YCenter	Specifies the logical Y-coordinate of the center of the object to be rotated in tenths of a micron, relative to the center of the page.

Note

- You can use the ANGLECONVERT function to specify angle measurements.

Example

```
.CreateRectangle 500000, -750000, -500000, 750000, 0
.RotateObject 45000000, -1, 0,0
```

The above example rotates the rectangle 45 degrees counter clockwise.

```
.CreateRectangle 500000, -750000, -500000, 750000, 0
.RotateObject -45000000, 0, -500000, 500000
```

The above example rotates the rectangle 45 degrees clockwise about the specified point.

{button ,AL(`OVR1 Transformation commands;',0,"Defaultoverview",)} **Related Topics**

SkewObject (DRAW)

.SkewObject .XAngle = *long*, .YAngle = *long*, .Reference = *long*

This command lets you skew the selected object.

Syntax	Description
.XAngle	Specifies the amount of horizontal skew (skew along the X-axis), in millionths of degrees. Positive angles result in counter-clockwise skew. Negative angles result in clockwise skew.
.YAngle	Specifies the amount of vertical skew (skew along the Y-axis), in millionths of degrees. Positive angles result in counter-clockwise skew. Negative angles result in clockwise skew.
.Reference	Specifies the reference point of the object to be skewed. 1 = Upper-right 2 = Upper-middle 3 = Upper-left 4 = Middle-left 5 = Lower-left 6 = Lower-middle 7 = Lower-right 8 = Middle-right 9 = Center

Note

- You can use the ANGLECONVERT function to specify angle measurements.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.SkewObject -15000000, 20000000, 3
```

The above example creates a rectangle, horizontally skews it 15 degrees clockwise and vertically skews it 20 degrees counter-clockwise. The reference point for skewing is the upper-left position.

{button ,AL(`OVR1 Transformation commands;',0,"Defaultoverview",)} Related Topics

StretchObject (DRAW)

.StretchObject .XScaleNumerator = *long*, .XScaleDenominator = *long*, .YScaleNumerator = *long*, .YScaleDenominator = *long*, .bHMirror = *boolean*, .bVMirror = *boolean*, .ReferenceNum = *long*

This command stretches or mirrors the selected object.

Syntax	Description
.XScaleNumerator	Specifies the amount that the selected object is stretched along the X-axis. The final stretch value is determined by dividing this number by .XScaleDenominator. If the final value is less than 1, the object becomes smaller. If the final value is greater than 1, the object becomes larger.
.XScaleDenominator	Specifies the amount that the selected object is stretched along the X-axis. The final stretch value is determined by dividing .XScaleNumerator by this number. If the final value is less than 1, the object becomes smaller. If the final value is greater than 1, the object becomes larger.
.YScaleNumerator	Specifies the amount that the selected object is stretched along the Y-axis. The final stretch value is determined by dividing this number by .YScaleDenominator. If the final value is less than 1, the object becomes smaller. If the final value is greater than 1, the object becomes larger.
.YScaleDenominator	Specifies the amount that the selected object is stretched along the Y-axis. The final stretch value is determined by dividing .YScaleNumerator by this number. If the final value is less than 1, the object becomes smaller. If the final value is greater than 1, the object becomes larger.
.bHMirror	Set to TRUE (-1) to horizontally mirror the selected object.
.bVMirror	Set to TRUE (-1) to vertically mirror the selected object.
.ReferenceNum	Specifies the reference point of the object to be skewed. 1 = Upper-right 2 = Upper-middle 3 = Upper-left 4 = Middle-left 5 = Lower-left 6 = Lower-middle 7 = Lower-right 8 = Middle-right 9 = Center

{button ,AL(`OVR1 Transformation commands;',0,"Defaultoverview",)} **Related Topics**

Text commands

AddTabStop (DRAW)

.AddTabStop .FirstSelectedChar = *long*, .LastSelectedChar = *long*, .TabStop = *long*

This command adds tab stops to text.

Syntax	Description
.FirstSelectedChar	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
.LastSelectedChar	Specifies the ending character of the selected text.
.TabStop	Specifies the distance at which to apply tabs, in tenths of a micron.

Note

- The .CreateTextString and .SelectObjectsInRect functions must be called before this command.
- You can use the LENGTHCONVERT function, or one of the FROM... or TO... functions to specify length measurements.

Example

```
.CreateTextString 1000000, -1000000, -1000000, 1000000, "Specifies the type of underline. 0 =
None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double
thin 6 = Double thin words"
.SelectObjectsInRect 1000000, -1000000, -1000000, 1000000, 0
.AddTabStop 0, 0, 1270000
```

The above example adds a tab stop every 0.5 inch.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

AlignTextToBaseline (DRAW)

.AlignTextToBaseline .FirstSelectedChar = *long*, .LastSelectedChar = *long*

This command aligns text to the baseline.

Syntax	Description
.FirstSelectedChar	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
.LastSelectedChar	Specifies the ending character of the selected text.

Note

- The .CreateTextString and .SelectObjectsInRect functions must be called before this command.

Example

```
.CreateTextString 1000000, -1000000, -1000000, 1000000, "Specifies the type of underline. 0 =
None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double
thin 6 = Double thin words"
.SelectObjectsInRect 1000000, -1000000, -1000000, 1000000, 0
.AlignTextToBaseline 0, 0
```

The above example aligns the text to the baseline.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

CreateArticisticText (DRAW)

.CreateArtisticText .NewText = *string*, .X = *long*, .Y = *long*

This command allows you to create a text string with default text settings as Artistic Text. The left-most character of the Artistic Text is placed on the center of the page. Text created with the CreateArtisticText command can be modified using the SetArtisticText command.

Syntax	Description
.NewText	Specifies the name of the new text to create.
.X	Sets the X-coordinate of the artistic text's position in tenths of a micron, relative to the center of the page.
.Y	Sets the Y-coordinate of the artistic text's position in tenths of a micron, relative to the center of the page.

Example

```
.CreateArtisticText "CorelDRAW"
```

The above example displays the text string "CorelDRAW".

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

CreateTextString (DRAW)

.CreateTextString .Top = *long*, .Left = *long*, .Bottom = *long*, .Right = *long*, .Text = *string*

This command creates the text.

Syntax	Description
.Top	Specifies the Y-coordinate of the upper-left corner of the text's bounding box in tenths of a micron, relative to the center of the page.
.Left	Specifies the X-coordinate of the upper-left corner of the text's bounding box in tenths of a micron, relative to the center of the page.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the text's bounding box in tenths of a micron, relative to the center of the page.
.Right	Specifies the X-coordinate of the lower-right corner of the text's bounding box in tenths of a micron, relative to the center of the page.
.Text	Specifies the text. Maximum string length is 255 characters.

Note

- This function must be called first to create the text before any of the functions which manipulate the text.

Example

```
.CreateTextString 250000, -300000, -250000, 1100000, "COREL"
```

The above example creates the text "COREL".

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

ExtractText (DRAW)

.ExtractText .DestinationFile = *string*

This command extracts the Artistic Text to a text file which can then be edited in any text editor and merged back into the document with MergeTextBack.

Syntax	Description
.DestinationFile	Specifies the name of the destination file.

Example

```
.CreateArtisticText "COREL DRAW"
.ExtractText "C:\COREL70\DRAW\TEXTFILE.TXT"
```

The above example extracts the text "COREL DRAW" to a text file named "TEXTFILE.TXT".

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} Related Topics

FitTextToPath (DRAW)

.FitTextToPath .lTextOrientation = *long*, .lVertAlign = *long*, .lHorizAlign = *long*, .lCurveSideToFit = *long*, .bFitOtherSide = *boolean*, .lHorizOffset = *long*, .lDistFromPath = *long*

This command fits selected artistic text to the selected path.

Syntax	Description
.lTextOrientation	0 = Rotates individual characters to follow the contours of the path. 1 = The characters are not changed. 2 = Vertically skews each character, creating the impression that the text is standing upright on the path. 3 = Horizontally skews each character, creating the impression that the text is turning in toward the screen.
.lVertAlign	If you specify a distamce from the path, then this parameter has no effect. 0 = Variable. Allows you to move the text off the path by dragging with the mouse. 1 = Bottom. Aligns the descender line of the text with the path. 2 = Top. Aligns the ascender line of the text with the path. 3 = Center. Centers the text vertically on the path. 4 = Baseline. Aligns the baseline of the text with the path.
.lHorizAlign	1 = Aligns the text with the start node of the line or curve. 2 = Aligns the text with the end point of the line or curve. 3 = Centers the text on the path.
.lCurveSideToFit	1 = Aligns the text to the top of a closed object. 2 = Aligns the text to the left of a closed object. 3 = Aligns the text to the bottom of a closed object. 4 = Aligns the text to the right of a closed object.
.bFitOtherSide	Set to TRUE (-1) to place the text on the other side of the path.
.lHorizOffset	Specifies the distance the text is offset from the start node.
.lDistFromPath	Specifies the distance the text is from path.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

MergeBackText (DRAW)

.MergeBackText .SourceFile = *string*

This command merges the extracted text back into the DRAW document.

Syntax	Description
.SourceFile	Specifies the name of the source file to merge.

Example

```
.CreateArtisticText "COREL DRAW"
.ExtractText "C:\COREL70\DRAW\TEXTFILE.TXT"
.MergeBackText "C:\COREL70\DRAW\TEXTFILE.TXT"
```

The above example merges the extracted text from the file "TEXTFILE.TXT" back into the DRAW document.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

SetArtisticText (DRAW)

.SetArtisticText .NewText = *string*

This command allows you to change selected Artistic Text text strings.

Syntax	Description
.NewText	Specifies the name of the new text to set.

Example

```
.FileNew
.CreateArtisticText "1"
.FilePrint
FOR i%=2 TO 10 STEP 1
.SetArtisticText i%
.FilePrint
NEXT i%
```

The above example creates the string "1" as Artistic text and then prints the document. Within the FOR...NEXT loop, the Artistic text is changed from the numbers 2 to 10. After each change in the Artistic text, the document is printed.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

SetBullet (DRAW)

.SetBullet .FirstSelectedChar = *long*, .LastSelectedChar = *long*, .SymbolLibrary = *string*, .SymbolNumber = *long*, .PointSize = *long*, .BulletIndent = *long*, .VerticalShift = *long*

This command sets bullets for text.

Syntax	Description
.FirstSelectedChar	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
.LastSelectedChar	Specifies the ending character of the selected text.
.SymbolLibrary	Specifies the name of the symbol library. Refer to the Effects tab of the Paragraph dialog box for more details.
.SymbolNumber	Specifies the selected symbol number. Refer to the Effects tab of the Paragraph dialog box for more details.
.PointSize	Specifies the point size in tenths of a point.
.BulletIndent	Specifies the size of the bullet indentation in tenths of a micron.
.VerticalShift	Specifies the amount of baseline shift in tenths of a micron.

Note

- The .CreateTextString and .SelectObjectsInRect functions must be called before this command.

Example

```
.CreateTextString 1000000, -1000000, -1000000, 1000000, "Specifies the type of underline. 0 =
None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double
thin 6 = Double thin words"
.SelectObjectsInRect 1000000, -1000000, -1000000, 1000000, 0
.SetBullet 32, 123, "Animals 1", 55, 480, 400000, 0
```

The above inserts a camel bullet, indented 1.57 inches.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

SetCharacterAttributes (DRAW)

.SetCharacterAttributes .FirstSelectedChar = *long*, .LastSelectedChar = *long*, .FontName = *string*, .FontStyle = *long*, .PointSize = *long*, .Underline = *long*, .Overline = *long*, .StrikeOut = *long*, .Placement = *long*, .CharacterSpacing = *long*, .WordSpacing = *long*, .LineSpacing = *long*, .Alignment = *long*

This command sets the text character attributes.

Syntax	Description
.FirstSelectedChar	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
.LastSelectedChar	Specifies the ending character of the selected text.
.FontName	Specifies the font name.
.FontStyle	Specifies the style of the selected font. 7 = Normal 8 = Normal/Italic 13 = Bold 14 = Bold/Italic
.PointSize	Specifies the size of the selected font in tenths of a point.
.Underline	Specifies the type of underline. 0 = None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double thin 6 = Double thin words
.Overline	Specifies the type of overline. 0 = None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double thin 6 = Double thin words
.StrikeOut	Specifies the type of strikeout. 0 = None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double thin 6 = Double thin words
.Placement	Specifies the placement of the font. 0 = Normal 1 = Superscript 2 = Subscript
.CharacterSpacing	Specifies the character spacing in tenths of a percent.
.WordSpacing	Specifies the word spacing in tenths of a percent.
.LineSpacing	Specifies the line spacing in tenths of a percent.
.Alignment	Specifies the alignment. 0 = None 1 = Left 2 = Center 3 = Right 4 = Full justify 5 = Force justify

Example

```
.CreateTextString 250000, -300000, -250000, 1100000, "COREL"
.SelectObjectsInRect 250000, -300000, -250000, 1100000, 0
.SetCharacterAttributes 0, 4, "Arial", 13, 900, 0, 0, 0, 0, 0, 0, 0, 1
```

The above example creates the text "COREL", then sets the font to Arial, the font type to Bold, and the point size to 90.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

SetFrameColumn (DRAW)

.SetFrameColumn .ColumnNumber = *long*, .Width = *long*, .GutterWidth = *long*

This command formats columns for text.

Syntax	Description
.ColumnNumber	Specifies the column number.
.Width	Specifies the width of the column in tenths of a micron.
.GutterWidth	Specifies the width of the gutter in tenths of a micron.

Note

- The .CreateTextString and .SelectObjectsInRect functions must be called before this command. This command must be called twice.
- You can use the LENGTHCONVERT function, or one of the FROM... or TO... functions to specify length measurements.

Example

```
.CreateTextString 1000000, -1000000, -1000000, 1000000, "Specifies the type of underline. 0 =
None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double
thin 6 = Double thin words"
.SelectObjectsInRect 1000000, -1000000, -1000000, 1000000, 0
.SetFrameColumn 0, 500000, 50000
.SetFrameColumn 1, 500000, 50000
```

The above example formats the text into two columns, each 2 inches wide.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

SetIndents (DRAW)

.SetIndents .FirstSelectedChar = *long*, .LastSelectedChar = *long*, .FirstLine = *long*, .RestOfLines = *long*, .RightMargin = *long*

This command sets indents for text.

Syntax	Description
.FirstSelectedChar	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
.LastSelectedChar	Specifies the ending character of the selected text.
.FirstLine	Specifies the size of the first line indentation, in tenths of a micron.
.RestOfLines	Specifies the size of the remaining line indentation, in tenths of a micron.
.RightMargin	Specifies the size of the right margin indentation, in tenths of a micron.

Note

- The .CreateTextString and .SelectObjectsInRect functions must be called before this command.
- You can use the LENGTHCONVERT function, or one of the FROM... or TO... functions to specify length measurements.

Example

```
.CreateTextString 1000000, -1000000, -1000000, 1000000, "Specifies the type of underline. 0 =
None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double
thin 6 = Double thin words"
.SelectObjectsInRect 1000000, -1000000, -1000000, 1000000, 0
.SetIndents 0, 0, 0, 400000, 0
```

The above example indents all lines except the first by 1.57 inches.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} Related Topics

SetParagraphSpacing (DRAW)

.SetParagraphSpacing .FirstSelectedChar = *long*, .LastSelectedChar = *long*, .CharacterSpacing = *long*, .WordSpacing = *long*, .LineSpacing = *long*, .BeforeParagraph = *long*, .AfterParagraph = *long*, .Alignment = *long*, .AutoHyphenation = *boolean*, .HyphenHotZone = *long*

This command sets paragraph spacing.

Syntax	Description
.FirstSelectedChar	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
.LastSelectedChar	Specifies the ending character of the selected text.
.CharacterSpacing	Specifies the character spacing in tenths of a percent.
.WordSpacing	Specifies the word spacing in tenths of a percent.
.LineSpacing	Specifies the line spacing in tenths of a percent.
.BeforeParagraph	Specifies the spacing before paragraphs in tenths of a percent.
.AfterParagraph	Specifies the spacing after paragraphs in tenths of a percent.
.Alignment	Specifies the alignment. 0 = None 1 = Left 2 = Center 3 = Right 4 = Full justify 5 = Force justify
.AutoHyphenation	Set to TRUE (-1) to enable automatic hyphenation. Set to FALSE (0) to disable this option.
.HyphenHotZone	Specifies the size of the hyphen hot zone in tenths of a micron.

Note

- The .CreateTextString and .SelectObjectsInRect functions must be called before this command.

Example

```
.CreateTextString 1000000, -1000000, -1000000, 1000000, "Specifies the type of underline. 0 =
None 1 = Single thin 2 = Single thin words 3 = Single thick 4 = Single thick words 5 = Double
thin 6 = Double thin words"
.SelectObjectsInRect 1000000, -1000000, -1000000, 1000000, 0
.SetParagraphSpacing 0, 0, 900, 900, 900, 200, 200, 1, 0, 0
```

The above example creates a text string, selects the entire text and applies paragraph spacing to it.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

SetTextString (DRAW)

.SetTextString .FirstSelectedChar = *long*, .LastSelectedChar = *long*, .Text = *string*

This command changes the text in a selected text object.

Syntax	Description
.FirstSelectedChar	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
.LastSelectedChar	Specifies the ending character of the selected text.
.Text	Specifies the text. Maximum string length is 255 characters.

Note

- The .CreateTextString and .SelectObjectsInRect functions must be called before this command.

Example

```
.CreateTextString 250000, -300000, -250000, 1100000, "COREL"
.SelectObjectsInRect 250000, -300000, -250000, 1100000, 0
.SetCharacterAttributes 0, 4, "Arial", 13, 900, 0, 0, 0, 0, 0, 0, 0, 1
.SetTextString -1, -1, "RT"
.SetCharacterAttributes 5, 6, "Arial", 8, 900, 0, 0, 0, 1, 0, 0, 0, 0
```

The above example creates the text string "COREL", then appends a second text string "RT" to it. The appended string is italic and superscript.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

StraightenText (DRAW)

.StraightenText .FirstSelectedChar = *long*, .LastSelectedChar = *long*

This command resets the kerning angle to 0 for selected text.

Syntax	Description
.FirstSelectedChar	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
.LastSelectedChar	Specifies the ending character of the selected text.

Example

```
.StraightenText 0, 0
```

The above example straightens the first character of selected paragraph text.

{button ,AL(`OVR1 Text commands;',0,"Defaultoverview",)} **Related Topics**

Fill and outline commands

AddArrowPoint (DRAW)

.AddArrowPoint .lX = *long*, .lY = *long*, .bEnabled = *boolean*, .bLetter = *boolean*, .bUser = *boolean*, .bClosed = *boolean*, .lContinuity = *long*, .lNodeType = *long*

This command adds a node that is part of an arrowhead. This command must be part of a block of commands that begins with the .BeginDrawArrow command and ends with the .EndDrawArrow command. To add the arrowhead to a path, use the .SetOutlineArrow command.

Syntax	Description
.lX	Specifies the X-coordinate of the node in tenths of a micron, relative to the center of the arrowhead area.
.lY	Specifies the Y-coordinate of the node in tenths of a micron, relative to the center of the arrowhead area.
.bEnabled	Always set to FALSE (0)
.bLetter	Always set to FALSE (0)
.bUser	Set to TRUE (-1) to make this node selectable.
.bClosed	Set to TRUE (-1) to make the arrowhead closed.
.lContinuity	0 = cusp node 1 = smooth node 2 = symmetrical node
.lNodeType	1 = straight line segment 2 = curved line segment

Example

```
.BeginDrawArrow TRUE, 0, 7
.AddArrowPoint -1016000, 0, FALSE, FALSE, TRUE, TRUE, 0, 0
.AddArrowPoint -481838, -271272, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 571500, -271272, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 31750, 0, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 571500, 299974, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint -481838, 299974, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint -1016000, 0, FALSE, FALSE, FALSE, TRUE, 0, 1
.EndDrawArrow
.SetOutlineArrow 0
```

The above example adds an arrowhead to the selected line.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

ApplyFountainFill (DRAW)

.ApplyFountainFill .Type = *long*, .CenterX = *long*, .CenterY = *long*, .Angle = *long*, .Steps = *long*, .Padding = *long*, .Blend = *long*, .Rate = *long*

This command lets you apply a Fountain Fill to the selected object. If the Blend was Custom, then all intermediate colors will be lost unless the Blend applied is again Custom. If the existing fill is not fountain, the start color will be CMYK Black and the end color CMYK white.

Syntax	Description
.Type	Specifies the type of Fountain Fill to apply: 0 = Linear (default) 1 = Radial 2 = Conical 3 = Square
.CenterX	Specifies the horizontal offset of the center of the fill. Valid values range from -100 to 100 percent. A value of -50% will place the center on the left edge of your object; a value of 50% will place it on the right edge.
.CenterY	Specifies the vertical offset of the center of the fill. Valid values range from -100 to 100 percent. A value of -50% will place the center on the bottom edge of your object; a value of 50% will place it on the top edge.
.Angle	Specifies the angle at which the fill is applied in tenths of degrees. Positive values will rotate the fill counter-clockwise, negative values will rotate it clockwise.
.Steps	Specifies the number of steps you want. Lower values produce coarser fountains on screen which take less time to redraw. Valid values range from 2 to 256.
.Padding	Specifies the amount of padding to apply to the fill. Ignored for type 2. Valid values range from 0 to 45 percent.
.Blend	Specifies the type of blending to apply to the fill. 0 = Direct (default) 1 = Rainbow CW 2 = Rainbow CCW 3 = Custom
.Rate	Specifies the mid-point used to apply the fill. Valid values range from 1 to 99.

Note

- To apply a two-color fill:

  .ApplyFountainFill must be followed by two calls to the .SetFountainFillColor command.
- To apply a custom fill:

  .ApplyFountainFill must be followed by .SetFountainFillColor 'n' times, where 'n' is any integer between 1 and 101.
- The Horizontal and Vertical Offset options are not available for linear fountain fills; set parameters to 0.
- The Angle option is not available for circular fountain fills; set parameter to 0.
- You can use the ANGLECONVERT function to specify angle measurements

Example

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.ApplyFountainFill 1, -50, -50, 900, 20, 20, 2, 1
.SetFountainFillColor 100, 5, 255, 0, 0, 0
.SetFountainFillColor 0, 5, 0, 0, 255, 0
```

The above example fills the ellipse with a red to blue fountain fill.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

ApplyFullColorFill (DRAW)

.ApplyFullColorFill .FileName = *string*, .TileWidth = *long*, .TileHeight = *long*, .FirstTileOffsetX = *long*, .FirstTileOffsetY = *long*, .RowOffset = *boolean*, .RowColumnOffset = *long*, .SeamlessTiling = *boolean*, .ScaleWithObject = *boolean*, .lTop = *long*, .lBottom = *long*, .lLeft = *long*, .lRight = *long*

This command lets you apply a Full Color Fill to a selected object.

Syntax	Description
.FileName	Specifies the name of the Fill file.
.TileWidth	Specifies the width of the tile. If .ScaleWithObject is TRUE (-1), .TileWidth is expressed in tenths of a micron. If .ScaleWithObject is FALSE (0), .TileWidth is expressed in percent.
.TileHeight	Specifies the height of the tile. If .ScaleWithObject is TRUE (-1), .TileHeight is expressed in tenths of a micron. If .ScaleWithObject is FALSE (0), .TileHeight is expressed in percent.
.FirstTileOffsetX	Specifies the amount of offset applied to the first tile along the X-axis. Valid values range from 0 to 100 percent.
.FirstTileOffsetY	Specifies the amount of offset applied to the first tile along the Y-axis. Valid values range from 0 to 100 percent.
.RowOffset	Set to TRUE (-1) to enable row offset. Set to FALSE (0) to enable column offset.
.RowColumnOffset	Specifies the amount of row or column offsets. Valid values range from 0 to 100.
.SeamlessTiling	Set to TRUE (-1) to enable seamless tiling. Set to FALSE (0) to disable this option.
.ScaleWithObject	Set to TRUE (-1) to scale the pattern with the object. Set to FALSE (0) to disable this option.
.lTop	Specifies the top coordinate of the fill's bounding box in tenths of a micron, relative to the center of the page.
.lBottom	Specifies the bottom coordinate of the fill's bounding box in tenths of a micron, relative to the center of the page.
.lLeft	Specifies the left coordinate of the fill's bounding box in tenths of a micron, relative to the center of the page.
.lRight	Specifies the right coordinate of the fill's bounding box in tenths of a micron, relative to the center of the page.

Note

- You can use the LENGTHCONVERT function, or one of the FROM... or TO... functions to specify length measurements.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyFullColorFill "C:\COREL70\COLOR\MONTEMP.BMP", 500000, 500000, 100, 100, 0, 100, 0, 0
```

The above example applies a full color fill to a rectangle.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

ApplyNoFill (DRAW)

.ApplyNoFill

This command removes the fill from the selected object, allowing objects behind it to show through.

Example

```
.SelectAllObjects
.ApplyNoFill
```

The above example removes the fill from all objects.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

ApplyOutline (DRAW)

.ApplyOutline .Width = *long*, .Type = *long*, .EndCaps = *long*, .JoinType = *long*, .Aspect = *long*, .Angle = *long*, .DotDash = *long*, .RightArrow = *long*, .LeftArrow = *long*, .BehindFill = *boolean*, .OutlineType = *long*, .Preset = *long*, .ScalePen = *boolean*

This command lets you apply an Outline to the selected object.

Syntax	Description
.Width	Specifies the width of the outline to apply, in tenths of a micron.
.Type	Specifies the outline type: 0 = None 1 = Solid 2 = Dot - Dash
.EndCaps	Specifies the end caps to be applied to the outline: 0 = Butt 1 = Round 2 = Square
.JoinType	Specifies the outline join types: 0 = Miter 1 = Round 2 = Bevel
.Aspect	Specifies the stretch field which adjusts the width of the nib. Valid values range from 1 to 100 percent.
.Angle	Specifies the angle of the nib's edge, in tenths of degrees.
.DotDash	Specifies the type of dot/dash line. Dot/dash line types are listed in the Style drop-down list box of the Outline Pen Roll-Up. The types are numbered and identified according to their position in the list the first listed type is identified as 0, the second listed type is identified as 1, and so on.
.RightArrow	Specifies the style of right-arrow. Right-arrow types are listed in the right arrow drop-down list box of the Outline Pen Roll-Up. The types are numbered and identified according to their position in the list (from left-to-right) the first listed type is identified as 0, the second listed type is identified as 1, and so on.
.LeftArrow	Specifies the style of left-arrow. Left-arrow types are listed in the left arrow drop-down list box of the Outline Pen Roll-Up. The types are numbered and identified according to their position in the list (from left-to-right) the first listed type is identified as 0, the second listed type is identified as 1, and so on.
.BehindFill	Set to TRUE (-1) to position the outline behind the fill. Set to FALSE (0) to position the outline in front of the fill.
.OutlineType	Specifies the type of preset outline. 0 = Pen 1 = Outline 2 = PenOutline.
.Preset	Specifies the tint of the outline. Values range from 1 (0%) to 11 (100%). A value of 0 has no effect on the outline. This parameter is only used if the selected outline type supports preset tints.
.ScalePen	Set to TRUE (-1) to scale the outline when the object is scaled.

Note

- You can use the ANGLECONVERT function to specify angle measurements

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyOutline 50000, 2, 0, 1, 50, 250, 2, 0, 0, 0
```

The above example applies a dashed outline 50000 microns wide, with round corners to the rectangle.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

ApplyPostscriptFill (DRAW)

.ApplyPostscriptFill .PSFill = *string*, .NumParms = *long*, .Parm1 = *long*, .Parm2 = *long*, .Parm3 = *long*, .Parm4 = *long*, .Parm5 = *long*

This command lets you apply a PostScript Fill to a selected object.

Syntax	Description
.PSFill	Specifies the name of the postscript fill. The name must be preceded by an F/. For a listing of Postscript fills available, see the PostScript Texture dialog box. If you create custom PostScript fills, their definitions are placed in the USERPROC.PS file in the Custom folder of your Corel folder. The PostScript fills definitions supplied with DRAW are also in this file.
.NumParms	Specifies the number of parameters used for the selected PostScript Fill, an integer value between 1 and 5, inclusive. Refer to the PostScript Texture dialog box to determine the number of parameters for a full fill. Set to 2 for spot fills.
.Parm1	Specifies the first parameter for the selected PostScript Fill. This parameter varies depending on the Fill selected. Refer to the PostScript Texture dialog box for full fill parameter specifics. If you're using a spot fill, set Parm1 to a value between -1 and 1.
.Parm2	Specifies the second parameter for the selected PostScript Fill. This parameter varies depending on the Fill selected. Refer to the PostScript Texture dialog box for full fill parameter specifics. If the parameter is not used with the fill you selected, set it to 0. If you're using a spot fill, set Parm2 to a value between -1 and 1.
.Parm3	Specifies the third parameter for the selected PostScript Fill. This parameter varies depending on the Fill selected. Refer to the PostScript Texture dialog box for full fill parameter specifics. If the parameter is not used with the fill you selected, set it to 0. If you're using a spot fill, set this parameter to 0.
.Parm4	Specifies the fourth parameter for the selected PostScript Fill. This parameter varies depending on the Fill selected. Refer to the PostScript Texture dialog box for full fill parameter specifics. If the parameter is not used with the fill you selected, set it to 0. If you're using a spot fill, set this parameter to 0.
.Parm5	Specifies the fifth parameter for the selected PostScript Fill. This parameter varies depending on the Fill selected. Refer to the PostScript Texture dialog box for full fill parameter specifics. If the parameter is not used with the fill you selected, set it to 0. If you're using a spot fill, set this parameter to 0.

 **Note**

- If you create custom PostScript fills, their definitions are placed in the USERPROC.PS file in the Custom folder of your Corel folder. The PostScript fills definitions supplied with DRAW are also in this file.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyPostScriptFill "F/StoneWall", 4, 15, 100,0, 5, 0
```

The above example applies the StoneWall PostScript fill to the selected rectangle.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

ApplyPreset (DRAW)

.ApplyPreset .PresetFileName = *string*, .PresetName = *string*

This command lets you load and apply a Preset.

Syntax	Description
.PresetFileName	Specifies the name of the Preset File.
.PresetName	Specifies the name of the Preset.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyPreset "C:\COREL70\DRAW\CORELDRW.PST", "Button Blue"
```

The above example applies the specified preset fill to the rectangle.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

ApplyTextureFill (DRAW)

.ApplyTextureFill .TextureLibrary = *string*, .Texture = *string*, .Style = *string*

This command lets you apply one of the texture fills included in CorelDRAW.

Syntax	Description
.TextureLibrary	Specifies the name of the Texture Library.
.Texture	Specifies the name of the texture.
.Style	Specifies the name of the style. If you set .TextureLibrary to "Samples 5", the style name must be preceded by "CDR5:". For example, "CDR5:Blue Valley".

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyTextureFill "Styles", "Satellite Photography", "Satellite Photography"
```

The above example creates a rectangle, then applies the satellite photography fill to it.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

ApplyTwoColorFill (DRAW)

.ApplyTwoColorFill .FileName = *string*, .ColorModel1 = *long*, .Color11 = *long*, .Color12 = *long*, .Color13 = *long*, .Color14 = *long*, .ColorModel2 = *long*, .Color21 = *long*, .Color22 = *long*, .Color23 = *long*, .Color24 = *long*, .TileWidth = *long*, .TileHeight = *long*, .FirstTileOffsetX = *long*, .FirstTileOffsetY = *long*, .RowOffset = *boolean*, .RowColumnOffset = *long*, .SeamlessTiling = *boolean*, .ScaleWithObject = *boolean*

This command lets you apply a Two-Color fill to the selected object.

Syntax	Description
.FileName	Specifies the name of the two color fill file to use. See the Two-Color Bitmap Pattern dialog box for a list of valid file formats.
.ColorModel1	Specifies the Color Model to use for the first color: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color11	Specifies the first color component for .ColorModel1. For example, Hue is the first color component for HSB. Click for valid value ranges.
.Color12	Specifies the second color component for .ColorModel1. For example, Green is the second color component for RGB. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.Color13	Specifies the third color component for .ColorModel1. For example, Saturation is the third color component for HLS. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.Color14	Specifies the fourth color component for .ColorModel1. For example, Black is the fourth color component for CMYK. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.ColorModel2	Specifies the Color Model to use for the second color: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color21	Specifies the first color component for .ColorModel2. For example, Hue is the first color component for HSB. Click for valid value ranges.
.Color22	Specifies the second color component for .ColorModel2. For example, Green is the second color component for RGB. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.Color23	Specifies the third color component for .ColorModel2. For example, Saturation is the third color component for HLS. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.Color24	Specifies the fourth color component for .ColorModel2. For example, Black is the fourth color component for CMYK. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.TileWidth	Specifies the width of the tile, in tenths of a micron.
.TileHeight	Specifies the height of the tile, in tenths of a micron.
.FirstTileOffsetX	Specifies the amount of offset applied to the first tile along the X-axis. Valid values range from 0 to 100.
.FirstTileOffsetY	Specifies the amount of offset applied to the first tile along the Y-axis. Valid values

	range from 0 to 100.
.RowOffset	Set to TRUE (-1) to enable row offset. Set to FALSE (0) to enable column offset.
.RowColumnOffset	Specifies the amount of row or column offsets. Valid values range from 0 to 100.
.SeamlessTiling	Set to TRUE (-1) to enable seamless tiling. Set to FALSE (0) to disable this option.
.ScaleWithObject	Set to TRUE (-1) to enable seamless tiling. Set to FALSE (0) to disable this option.

Note

- You can use the LENGTHCONVERT function, or one of the FROM... or TO... functions to specify length measurements.

Example

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0
.ApplyTwoColorFill "mybitmap.bmp", 5, 255, 0, 0, 0, 5, 0, 0, 0, 0, 500000, 500000, 100, 100, 0,
100, 0, 0, 0
```

The above example applies a two-color bitmap fill from the MYBITMAP.BMP file to the rectangle.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

ApplyUniformFillColor (DRAW)

.ApplyUniformFillColor .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command lets you apply a Uniform Fill Color to a selected object.

Syntax	Description
.ColorModel	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.

.Color2 Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color4 Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.ApplyUniformFillColor 2, 100, 0, 0, 0
```

The above example creates an ellipse and uniformly fills it with cyan.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

BeginDrawArrow (DRAW)

.BeginDrawArrow .bLeftArrow = *boolean*, .lLineOffset = *long*, .NumOfPoints = *long*

This command initializes a block of arrowhead creation commands. This block must include one or more instances of the .AddArrowPoint command and this block must end with the .EndDrawArrow command. To add the arrowhead to a path, use the .SetOutlineArrow command.

Syntax	Description
.bLeftArrow	Set to TRUE (-1) to create a left-facing arrowhead. Set to FALSE (0) to create a right-facing arrowhead.
.lLineOffset	Specifies the distance between the end of the path and the arrowhead.
.NumOfPoints	Specifies the number of nodes in your arrowhead.

Example

```
.BeginDrawArrow TRUE, 0, 7
.AddArrowPoint -1016000, 0, FALSE, FALSE, TRUE, TRUE, 0, 0
.AddArrowPoint -481838, -271272, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 571500, -271272, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 31750, 0, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 571500, 299974, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint -481838, 299974, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint -1016000, 0, FALSE, FALSE, FALSE, TRUE, 0, 1
.EndDrawArrow
.SetOutlineArrow 0
```

The above example adds an arrowhead to the selected line.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

ConvertColor (DRAW)

.ConvertColor .lFromColorModel = *long*, .lFromV1 = *long*, .lFromV2 = *long*, .lFromV3 = *long*, .lFromV4 = *long*, .lToColorModel = *long*, .plToV1 = *long**, .plToV2 = *long**, .plToV3 = *long**, .plToV4 = *long**

This command converts one color model to another color model.

Syntax	Description
.lFromColorModel	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.lFromV1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click for valid value ranges.
.lFromV2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lFromV3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lFromV4	Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lToColorModel	Returns the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.plToV1	Returns the first color component
.plToV2	Returns the second color component
.plToV3	Returns the third color component
.plToV4	Returns the fourth color component

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

EndDrawArrow (DRAW)

.EndDrawArrow

This command ends a block of arrowhead creation commands. This block must include one or more instances of the .AddArrowPoint command and this block must begin with the .BeginDrawArrow command.

Example

```
.BeginDrawArrow TRUE, 0, 7
.AddArrowPoint -1016000, 0, FALSE, FALSE, TRUE, TRUE, 0, 0
.AddArrowPoint -481838, -271272, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 571500, -271272, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 31750, 0, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 571500, 299974, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint -481838, 299974, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint -1016000, 0, FALSE, FALSE, FALSE, TRUE, 0, 1
.EndDrawArrow
.SetOutlineArrow 0
```

The above example adds an arrowhead to the selected line.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

GetFillType (DRAW)

ReturnValue& = .GetFillType()

This function returns the Fill Type of a selected object. If more than one object is selected, the function returns the Fill Type of the last selected object.

Syntax	Description
ReturnValue&	0 = None 1 = Uniform 2 = Fountain 6 = PostScript 7 = Two-color 9 = ColorBitmap 10 = Vector 11 = Texture

Example

```
.SelectObjectOfCDRStaticID IDRect&
fillType& = .GetFillType()
Message fillType&
```

The above example displays a number corresponding to the fill type of the selected object in a message box.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

GetFountainFill (DRAW)

.GetFountainFill .Type = *long**, .CenterX = *long**, .CenterY = *long**, .Angle = *long**, .Steps = *long**, .Padding = *long**, .Blend = *long**, .Rate = *long**, .NumColors = *long**

This command returns the Fountain Fill attributes of a selected object. If more than one object is selected, the function returns the Fountain Fill attributes of the last selected object.

Syntax	Description
.Type	Returns the type of Fountain Fill: 0 = Linear (default) 1 = Radial 2 = Conical 3 = Square
.CenterX	Returns the Horizontal Offset of the center of the fill. Valid values range from -100 to +100 percent. A value of -50% will place the center on the left edge of your object; a value of 50% will place it on the right edge.
.CenterY	Returns the Horizontal Offset of the center of the fill. Valid values range from -100 to +100 percent. A value of -50% will place the center on the bottom edge of your object; a value of 50% will place it on the top edge.
.Angle	Returns the angle at which the fill is applied in degrees. Positive values will rotate the fill counter-clockwise, negative values will rotate it clockwise.
.Steps	Returns the number of stripes you want. Lower values produce coarser fountains on screen which take less time to redraw. Valid values range from 2 to 256.
.Padding	Returns the amount of padding to apply to the fill. Ignored for type 2. Valid values range from 0 to 45 percent.
.Blend	Returns the type of blending to apply to the fill. 0 = Direct (default) 1 = Rainbow CW 2 = Rainbow CCW 3 = Custom
.Rate	Returns the rate method used to apply the fill.
.NumColors	Returns the number of colors.

Note

- You can use the ANGLECONVERT function to specify angle measurements

Example

```
.GetFountainFill fillType&, CX&, CY&, Angle&, Steps&, Pad&, Blend&, Rate&, Num&
MESSAGE fillType&
```

The above example returns Fountain Fill attributes and displays a number corresponding to the fill type in a message box.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

GetFountainFillColor (DRAW)

.GetFountainFillColor .lIndex = *long*, .plPosition = *long**, .plColorModel = *long**, .plV1 = *long**, .plV2 = *long**, .plV3 = *long**, .plV4 = *long**

This command Retrieves a color from a fountain fill.

Syntax	Description
.lIndex	The index number of the color you want to get. Use the .GetFountainFill command to find out how many colors are in a fountain fill. Valid index numbers will range from 0 to the number of colors minus 1. If you use the value 100, you will always get the end color.
.plPosition	Return the position of the color within the fill.
.plColorModel	Returns the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.plV1	Returns the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click for valid value ranges.

.plV2 Returns the second color component for .ColorModel. For example, Green is the second color component for RGB. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.plV3 Returns the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.plV4	Returns the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

GetOutline (DRAW)

.GetOutline .Width = *long**, .Type = *long**, .EndCaps = *long**, .JoinType = *long**, .Aspect = *long**, .Angle = *long**, .DotDash = *long**, .RightArrow = *long**, .LeftArrow = *long**, .BehindFill = *boolean**, .ScalePen = *boolean**

This function returns the outline attributes of the selected object. If more than one object is selected, the function returns the outline attributes of the last selected object.

Syntax	Description
.Width	Returns the width of the outline, in tenths of a micron.
.Type	Returns the outline type: 0 = None 1 = Solid 2 = Dot - Dash
.EndCaps	Returns the End Caps applied to the outline: 0 = Butt 1 = Round 2 = Square
.JoinType	Returns the outline join types: 0 = Miter 1 = Round 2 = Bevel
.Aspect	Returns the stretch field which adjusts the width of the nib.
.Angle	Returns the angle of the nib's edge, in tenths of degrees.
.DotDash	Returns the type of dot/dash line. Refer to the Outline Pen dialog box for more details.
.RightArrow	Returns the style of right-arrow. Refer to the Outline Pen dialog box for more details.
.LeftArrow	Returns the style of left-arrow. Refer to the Outline Pen dialog box for more details.
.BehindFill	Returns the position of the outline fill. TRUE (-1) = Outline behind fill FALSE (0) = Outline in front of fill
.ScalePen	Returns the the scale pen setting. TRUE (-1) = Outline is scaled when object is scaled FALSE (0) = Outline is not scaled when object is scaled

Note

- You can use the ANGLECONVERT function to specify angle measurements

Example

```
.GetOutline Width&, outlineType&, EndCaps&, JoinType&, Aspect&, Angle&, DotDash&, RArrow&,
LArrow&, BehindFill&
```

The above example returns the outline attributes of the selected object.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

GetOutlineColor (DRAW)

.GetOutlineColor .ColorModel = *long**, .Color1 = *long**, .Color2 = *long**, .Color3 = *long**, .Color4 = *long**

This function returns the Outline Color attributes of a selected object. If more than one object is selected, the function returns the Outline Color attributes of the last selected object.

Syntax	Description
.ColorModel	Returns the Color Model: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color1	Returns the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click for valid value ranges.

.Color2 Returns the second color component for .ColorModel. For example, Green is the second color component for RGB. Click for valid value ranges.

.Color3	Returns the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click for valid value ranges.

.Color4 Returns the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click for valid value ranges.

Example

```
.GetOutlineColor Model&, C1&, C2&, C3&, C4&
MESSAGE Model&
```

The above example determines the outline color attributes of the selected object and displays a number corresponding to the color model in a message box.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

GetUniformFillColor (DRAW)

.GetUniformFillColor .ColorModel = *long**, .Color1 = *long**, .Color2 = *long**, .Color3 = *long**, .Color4 = *long**

This function returns the Uniform Fill color attributes of a selected object. If more than one object is selected, the function returns the Uniform Fill color of the last selected object.

Syntax	Description
.ColorModel	Returns the Color Model: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color1	Returns the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.

.Color2 Returns the second color component for .ColorModel. For example, Green is the second color component for RGB. Click  for valid value ranges.

.Color3	Returns the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click  for valid value ranges.

.Color4 Returns the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click  for valid value ranges.

Example

```
.GetUniformFillColor Model&, C1&, C2&, C3&, C4&
MESSAGE Model&
```

The above example determines the uniform fill color of the selected object and displays a number corresponding to the color model in a message box.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

OverPrintFill (DRAW)

.OverPrintFill

This command overprints the fill of the selected object.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

OverPrintOutline (DRAW)

.OverPrintOutline

This command overprints the outline of the selected object.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

RemoveFountainFillColor (DRAW)

.RemoveFountainFillColor .Position = *long*

This command removes the currently selected Fountain Fill Color.

Syntax	Description
.Position	Specifies the position of the color to be removed. 0 and 100 are invalid values. For any other value, the color at that position is removed, if one exists. Existing fill must be a Fountain and Blend must be custom.

Example

```
.ApplyFountainFill 2, -50, -50, 900, 20, 20, 2, 0
.SetFountainFillColor 75, 5, 0, 255, 0, 0
.SetFountainFillColor 75, 5, 0, 0, 255, 0
.RemoveFountainFillColor 75
```

The above example removes the color from the fountain fill, resulting in a black and white fountain fill.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

SetFountainFillColor (DRAW)

.SetFountainFillColor .Position = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command sets the fountain fill color.

Syntax	Description
.Position	Specifies the position at which to set the color. If position is 0, then the start color is set. If position is 100, then the end color is set. For other values, a color at that position is added (or changed if one already exists at that position. Note: If the position is not 0 or 100, Blend is forced to be custom.
.ColorModel	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click for valid value ranges.

.Color2 Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color3 Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color4	Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

Note

- To apply a two-color fill:

  .ApplyFountainFill must be followed by two calls to the .SetFountainFillColor command.

- To apply a custom fill:

  .ApplyFountainFill must be followed by .SetFountainFillColor 'n' times, where 'n' is any integer between 1 and 101.

Example

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.ApplyFountainFill 2, -50, -50, 900, 20, 20, 2, 0
.SetFountainFillColor 75, 5, 0, 255, 0, 0
```

The above example fills the ellipse with a green fountain fill.

```
.CreateEllipse -250000, -500000, 250000, 500000, 0, 0, 0
.ApplyFountainFill 2, -50, -50, 900, 20, 20, 2, 0
.SetFountainFillColor 0, 5, 0, 255, 0, 0
.SetFountainFillColor 100, 5, 0, 0, 255, 0
```

The above example fills the ellipse with a two color fountain fill- green and blue.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

SetOutlineArrow (DRAW)

.SetOutlineArrow .lArrowType = *long*

This command changes the arrowhead of the outline of the selected object. This command follows a block of arrowhead creation commands (see example).

Syntax	Description
.lArrowType	Specifies the arrow position. 0 = left arrow 1 = right arrow 2 = both arrows

Example

```
.BeginDrawArrow TRUE, 0, 7
.AddArrowPoint -1016000, 0, FALSE, FALSE, TRUE, TRUE, 0, 0
.AddArrowPoint -481838, -271272, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 571500, -271272, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 31750, 0, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint 571500, 299974, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint -481838, 299974, FALSE, FALSE, TRUE, FALSE, 0, 1
.AddArrowPoint -1016000, 0, FALSE, FALSE, FALSE, TRUE, 0, 1
.EndDrawArrow
.SetOutlineArrow 0
```

The above example adds an arrowhead to the selected line.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

SetOutlineColor (DRAW)

.SetOutlineColor .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command sets the color to be applied to the outline.

Syntax	Description
.ColorModel	Specifies the Color Model which indicates how each of the four colors (Color1-Color4) are to be interpreted. 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.Color1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.

.Color2 Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.Color4 Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.SetOutlineColor 2, 0, 0, 255, 0
```

The above example sets the outline color to yellow.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

SetOutlineStyle (DRAW)

.SetOutlineStyle .lDotDash = *long*

This command changes the style of the outline of the selected object.

Syntax	Description
.lDotDash	Specifies the outline style. This parameter can range from 0 to 27; 0 is a solid line; 1 to 27 correspond to the outline styles in CorelDRAW.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} Related Topics

SetOutlineWidth (DRAW)

.SetOutlineWidth .lWidth = *long*

This command changes the width of the outline of the selected object.

Syntax	Description
.lWidth	Specifies the width of the outline.

{button ,AL(`OVR1 Fill and outline commands;',0,"Defaultoverview",)} **Related Topics**

Special effects commands

ApplyBlend (DRAW)

.ApplyBlend .bSteps = *boolean*, .lNoOfSteps = *long*, .lAngleOfRotation = *long*, .bLoop = *boolean*, .lPathObjectID = *long*, .bFullPath = *boolean*, .bRotateAll = *boolean*, .lColorWheelMode = *long*, .lMapNodeStartObject = *long*, .lMapNodeEndObject = *long*, .bLinearBlend = *boolean*, .bLinearSpacing = *boolean*, .bLinkAcceleration = *boolean*, .bAccelShapes = *boolean*, .lBendLogBase = *long*, .lSpacingLogBase = *long*, .lBlendId = *long*, .lBlendType = *long*

This command applies a blend to two selected objects. The parameters below correspond to the controls in the Blend Roll-Up.

Syntax	Description
.bSteps	Set to TRUE (-1) to set the number of steps. Set to FALSE (0) if the blend is on a path and you want to use fixed spacing along that path.
.lNoOfSteps	Specifies the number of intermediate steps.
.lAngleOfRotation	Specifies the rotation of the intermediate steps in millionths of a degree (e.g., 5000000 = 5 degrees).
.bLoop	Set to TRUE (-1) to enable the loop option. Set to FALSE (0) to disable the loop option.
.lPathObjectID	Specifies the object ID of the path object. Use .GetObjectsCDRStaticID to get an object's ID.
.bFullPath	Set to TRUE (-1) to enable the blend along full path option. Set to FALSE (0) to disable the blend along full path option.
.bRotateAll	Set to TRUE (-1) to enable the rotate all objects option. Set to FALSE (0) to disable the rotate all objects option.
.lColorWheelMode	0 = straight 1 = clockwise 2 = counter-clockwise
.lMapNodeStartObject	Specifies a node on the start object to map to a specific node on the end object. The value can range from 0 (the first node) to the number of nodes minus 1.
.lMapNodeEndObject	Specifies a node on the end object to map to a specific node on the start object. The value can range from 0 (the first node) to the number of nodes minus 1.
.bLinearBlend	Set to TRUE (-1) to enable the rotate all objects option. Set to FALSE (0) to disable the rotate all objects option.
.bLinearSpacing	Set to TRUE (-1) to enable the rotate all objects option. Set to FALSE (0) to disable the rotate all objects option.
.bLinkAcceleration	Set to TRUE (-1) to link the blend acceleration options.
.bAccelShapes	Set to TRUE (-1) to accelerate the change in size between the start and end objects.
.lBendLogBase	Specifies the rate of color acceleration.
.lSpacingLogBase	Specifies the rate of spacing acceleration.
.lBlendId	Reserved for future use.
.lBlendType	Reserved for future use.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

ApplyContour (DRAW)

.ApplyContour .IContourType = *long*, .IOffset = *long*, .ISteps = *long*, .IColorWheelDirection = *long*, .IOutlineColorModel = *long*, .IOutlineV1 = *long*, .IOutlineV2 = *long*, .IOutlineV3 = *long*, .IOutlineV4 = *long*, .IFillFromColorModel = *long*, .IFillFromV1 = *long*, .IFillFromV2 = *long*, .IFillFromV3 = *long*, .IFillFromV4 = *long*, .IFillToColorModel = *long*, .IFillToV1 = *long*, .IFillToV2 = *long*, .IFillToV3 = *long*, .IFillToV4 = *long*

This command applies a contour to the selected object.

Syntax	Description
.IContourType	0 = To Center 1 = Inside 2 = Outside
.IOffset	Specifies the distance between contours in tenths of a micron.
.ISteps	Specifies the number of steps.
.IColorWheelDirection	0 = straight 1 = clockwise 2 = counter-clockwise
.IOutlineColorModel	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.IOutlineV1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click for valid value ranges.
.IOutlineV2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.IOutlineV3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.IOutlineV4	Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.IFillFromColorModel	Specifies the Color Model to use.
.IFillFromV1	Specifies the first color component.
.IFillFromV2	Specifies the second color component.
.IFillFromV3	Specifies the third color component.
.IFillFromV4	Specifies the fourth color component.
.IFillToColorModel	Specifies the Color Model to use.
.IFillToV1	Specifies the first color component.
.IFillToV2	Specifies the second color component.
.IFillToV3	Specifies the third color component.
.IFillToV4	Specifies the fourth color component.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

ApplyEnvelopeFrom (DRAW)

.ApplyEnvelopeFrom .lObjectID = *long*, .lMappingmode = *long*, .bKeepLines = *boolean*

This command applies an envelope to the selected object from the shape of another object.

Syntax	Description
.lObjectID	Specifies the object ID of the source object. Use .GetObjectsCDRStaticID to get an object's ID.
.lMappingmode	0 = Horizontal 1 = Original 2 = Putty 3 = Vertical
.bKeepLines	Set to TRUE (-1) to keep the lines of the source object. Set to FALSE (0) to exclude these lines.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

ApplyExtrude (DRAW)

.ApplyExtrude .lExtrudeType = *long*, .lVPProperties = *long*, .lCopyObjectID = *long*, .lDepth = *long*, .lVPHorizPos = *long*, .lVPVertPos = *long*, .bPageOrigin = *boolean*, .lLight1Pos = *long*, .lLight1Intensity = *long*, .lLight2Pos = *long*, .lLight2Intensity = *long*, .lLight3Pos = *long*, .lLight3Intensity = *long*, .lFillType = *long*, .lDrapeFillOrFillColorModel = *long*, .lFillV1 = *long*, .lFillV2 = *long*, .lFillV3 = *long*, .lFillV4 = *long*, .lFillFrontColorModel = *long*, .lFillFrontV1 = *long*, .lFillFrontV2 = *long*, .lFillFrontV3 = *long*, .lFillFrontV4 = *long*, .lFillBackColorModel = *long*, .lFillBackV1 = *long*, .lFillBackV2 = *long*, .lFillBackV3 = *long*, .lFillBackV4 = *long*

This command extrudes the selected object.

Syntax	Description
.lExtrudeType	0 = Small Back 1 = Small Front 2 = Big Back 3 = Big Front 4 = Back Parallel 5 = Front Parallel
.lVPProperties	0 = Vanishing Point locked to object 1 = Vanishing Point locked to page 2 = Copy VP from object (specified with .lCopyObjectID) 3 = Shared VP (specified with .lCopyObjectID)
.lCopyObjectID	Specifies the object ID of the source object for shared and copied vanishing points. Use .GetObjectsCDRStaticID to get an object's ID.
.lDepth	Specifies the depth of the extrusion.
.lVPHorizPos	Specifies the X-coordinate of the vanishing point in tenths of a micron, relative to the center of the page or the object depending on .bPageOrigin.
.lVPVertPos	Specifies the Y-coordinate of the vanishing point in tenths of a micron, relative to the center of the page or the object depending on .bPageOrigin.
.bPageOrigin	Set to TRUE (-1) to position the vanishing point using absolute page coordinates. Set to FALSE (0) to position the vanishing point relative to the object's position.
.lLight1Pos	Specifies the position of the light source. Valid values range from 0 to 16.
.lLight1Intensity	Specifies the intensity of the light source. Valid values range from 0 to 100.
.lLight2Pos	Specifies the position of the light source. Valid values range from 0 to 16.
.lLight2Intensity	Specifies the intensity of the light source. Valid values range from 0 to 100.
.lLight3Pos	Specifies the position of the light source. Valid values range from 0 to 16.
.lLight3Intensity	Specifies the intensity of the light source. Valid values range from 0 to 100.
.lFillType	0 = Object fill 1 = Solid fill 2 = Shade
.lDrapeFillOrFillColorModel	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.lFillV1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.
.lFillV2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lFillV3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.lFillV4 Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.lFillFrontColorModel	Specifies the Color Model to use.
.lFillFrontV1	Specifies the first color component.
.lFillFrontV2	Specifies the second color component.
.lFillFrontV3	Specifies the third color component.
.lFillFrontV4	Specifies the fourth color component.
.lFillBackColorModel	Specifies the Color Model to use.
.lFillBackV1	Specifies the first color component.
.lFillBackV2	Specifies the second color component.
.lFillBackV3	Specifies the third color component.
.lFillBackV4	Specifies the fourth color component.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

ApplyLensEffect (DRAW)

.ApplyLensEffect .lLensType = *long*, .bFrozen = *boolean*, .bRemoveFace = *boolean*, .bViewPoint = *boolean*, .lVpX = *long*, .lVpY = *long*, .lParam1 = *long*, .lColorModel1 = *long*, .lColor1V1 = *long*, .lColor1V2 = *long*, .lColor1V3 = *long*, .lColor1V4 = *long*, .lColorModel2 = *long*, .lColor2V1 = *long*, .lColor2V2 = *long*, .lColor2V3 = *long*, .lColor2V4 = *long*

This command adds a lens to the selected object.

Syntax	Description
.lLensType	0 = No Lens Effect 1 = Brighten 2 = Color Add 3 = Color Limit 4 = Custom Color Map 5 = Fish Eye 6 = Heat Map 7 = Invert 8 = Magnify 9 = Tinted Grayscale 10 = Transparancy 11 = Wireframe
.bFrozen	Set to TRUE (-1) to enable the Frozen option.
.bRemoveFace	Set to TRUE (-1) to enable the Remove Face option.
.bViewPoint	Set to TRUE (-1) to enable the View Point option.
.lVpX	Specifies the X-coordinate of the view point in tenths of a micron.
.lVpY	Specifies the Y-coordinate of the view point in tenths of a micron.
.lParam1	This value will vary depending on the selected lens. Refer to the Lens Roll-Up for more information
.lColorModel1	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.lColor1V1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click for valid value ranges.
.lColor1V2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lColor1V3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lColor1V4	Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lColorModel2	Specifies the Color Model to use.
.lColor2V1	Specifies the first color component.
.lColor2V2	Specifies the second color component.
.lColor2V3	Specifies the third color component.
.lColor2V4	Specifies the fourth color component.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

ApplyPerspectiveEffect (DRAW)

.ApplyPerspectiveEffect .lHandle = *long*, .lPosX = *long*, .lPosY = *long*

This command adds a perspective effect to the selected object.

Syntax	Description
.lHandle	Specifies the handle of the object that is repositioned to create the perspective effect. Valid values range from 1 to 5.
.lPosX	Specifies the X-coordinate of the new position of the handle, in tenths of a micron
.lPosY	Specifies the Y-coordinate of the new position of the handle, in tenths of a micron

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

ApplyPresetEnvelope (DRAW)

.ApplyPresetEnvelope .lPresetNumber = *long*, .lMappingmode = *long*, .bKeepLines = *boolean*

This command applies a preset envelope to the selected object.

Syntax	Description
.lPresetNumber	Specifies the preset envelope to use. Refer to the Preset Roll-Up to see which presets are available. Valid values range from 1 to 39.
.lMappingmode	0 = Horizontal 1 = Original 2 = Putty 3 = Vertical
.bKeepLines	Set to TRUE (-1) to keep the lines of the envelope. Set to FALSE (0) to exclude these lines.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

ApplyRotatedExtrude (DRAW)

.ApplyRotatedExtrude .lExtrudeType = *long*, .lDepth = *long*, .lXRotation = *long*, .lYRotation = *long*, .lZRotation = *long*, .lVPHorizPos = *long*, .lVPVertPos = *long*, .bPageOrigin = *boolean*, .lLight1Pos = *long*, .lLight1Intensity = *long*, .lLight2Pos = *long*, .lLight2Intensity = *long*, .lLight3Pos = *long*, .lLight3Intensity = *long*, .lFillType = *long*, .lDrapeFillOrFillColorModel = *long*, .lFillV1 = *long*, .lFillV2 = *long*, .lFillV3 = *long*, .lFillV4 = *long*, .lFillFrontColorModel = *long*, .lFillFrontV1 = *long*, .lFillFrontV2 = *long*, .lFillFrontV3 = *long*, .lFillFrontV4 = *long*, .lFillBackColorModel = *long*, .lFillBackV1 = *long*, .lFillBackV2 = *long*, .lFillBackV3 = *long*, .lFillBackV4 = *long*

This command applies a rotated extrusion to the selected object.

Syntax	Description
.lExtrudeType	0 = Small Back 1 = Small Front 2 = Big Back 3 = Big Front 4 = Back Parallel 5 = Front Parallel
.lDepth	Specifies the depth of the extrusion.
.lXRotation	Specifies the rotation value for the X-axis. Valid values range from 0 to 100.
.lYRotation	Specifies the rotation value for the Y-axis. Valid values range from 0 to 100.
.lZRotation	Specifies the rotation value for the Z-axis. Valid values range from 0 to 100.
.lVPHorizPos	Specifies the X-coordinate of the vanishing point, in tenths of a micron, relative to the center of the page or the object depending on .bPageOrigin.
.lVPVertPos	Specifies the Y-coordinate of the vanishing point, in tenths of a micron, relative to the center of the page or the object depending on .bPageOrigin.
.bPageOrigin	Set to TRUE (-1) to position the vanishing point using absolute page coordinates. Set to FALSE (0) to position the vanishing point relative to the object's position.
.lLight1Pos	Specifies the position of the light source. Valid values range from 0 to 16.
.lLight1Intensity	Specifies the intensity of the light source. Valid values range from 0 to 100.
.lLight2Pos	Specifies the position of the light source. Valid values range from 0 to 16.
.lLight2Intensity	Specifies the intensity of the light source. Valid values range from 0 to 100.
.lLight3Pos	Specifies the position of the light source. Valid values range from 0 to 16.
.lLight3Intensity	Specifies the intensity of the light source. Valid values range from 0 to 100.
.lFillType	0 = Object fill 1 = Solid fill 2 = Shade
.lDrapeFillOrFillColorModel	Specifies the Color Model to use: 1 = Pantone 2 = CMYK100 3 = CMYK255 4 = CMY 5 = RGB 6 = HSB 7 = HLS 8 = BW 9 = Gray 11 = YIQ255 12 = LAB
.lFillV1	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.
.lFillV2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lFillV3	Specifies the third color component for .ColorModel. For example, Saturation is the third color component for HLS. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.
.lFillV4	Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. Click  for valid value ranges. If this parameter is not available in the Color Model specified, set it to 0.

.lFillFrontColorModel	Specifies the Color Model to use.
.lFillFrontV1	Specifies the first color component.
.lFillFrontV2	Specifies the second color component.
.lFillFrontV3	Specifies the third color component.
.lFillFrontV4	Specifies the fourth color component.
.lFillBackColorModel	Specifies the Color Model to use.
.lFillBackV1	Specifies the first color component.
.lFillBackV2	Specifies the second color component.
.lFillBackV3	Specifies the third color component.
.lFillBackV4	Specifies the fourth color component.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

ClearEffect (DRAW)

.ClearEffect

This command removes a special effect from the selected object.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

CopyEffectFrom (DRAW)

.CopyEffectFrom .bClone = *boolean*, .lSourceObjectID = *long*

This command copies an effect from a specific object to the selected object.

Syntax	Description
.bClone	Set to TRUE (-1) to clone the effect instead of copying it. This creates a link between the two objects. When the effect is changed for one object, the other object also changes.
.lSourceObjectID	Specifies the object ID of the source object. Use .GetObjectsCDRStaticID to get an object's ID.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

DetachBlendPath (DRAW)

.DetachBlendPath

This command detaches a path from a blend group.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

FuseBlend (DRAW)

.FuseBlend .bEnd = *boolean*, .lPositionX = *long*, .lPositionY = *long*

This command fuses a split blend group.

Syntax	Description
.bEnd	Set to TRUE (-1) to fuse the top of the blend. Set to FALSE (0) to fuse the bottom of the blend.
.lPositionX	Specifies the X-coordinate of the point where you want the blend to be fused, in tenths of a micron, relative to the center of the page.
.lPositionY	Specifies the Y-coordinate of the point where you want the blend to be fused, in tenths of a micron, relative to the center of the page.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

SplitBlend (DRAW)

.SplitBlend .lPositionX = *long*, .lPositionY = *long*

This command splits the selected blend group.

Syntax	Description
.lPositionX	Specifies the X-coordinate of the point where you want the blend to be split, in tenths of a micron, relative to the center of the page.
.lPositionY	Specifies the Y-coordinate of the point where you want the blend to be split, in tenths of a micron, relative to the center of the page.

{button ,AL(`OVR1 Special effects commands;',0,"Defaultoverview",)} **Related Topics**

Object Data Manager commands

GetUserDataField (DRAW)

ReturnString$ = .GetUserDatdField(.FieldName = *string*)

This function returns a specified user-data field of a selected object. If more than one object is selected, the function returns the specified user-data field of the last selected object.

Syntax	Description
ReturnString$	Returns the user data field of the selected object.
.FieldName	Specifies the name of an object's user data field.

Example

```
u_d_f$="CDRStaticID"
data_field1$=.GetUserDataField (u_d_f)
data_field2$=.GetUserDataField ("Name")
```

The above example returns the value for the CDRStaticID and Name field of a selected object.

{button ,AL(`OVR1 Object Data Manager commands;',0,"Defaultoverview",)} Related Topics

SetUserDataField (DRAW)

.SetUserDataField .FieldName = *string*, .FieldValue = *string*

This command lets you set object data values for selected objects.

Syntax	Description
.FieldName	Specifies the name of the user data field to set.
.FieldValue	Specifies the value of the user data field to set.

Example

```
.CreateRectangle 1000000, 750000, 500000, 100000, 0
.SetUserDataField "Name", "MyObject"
```

The above example creates a rectangle and while it is still selected, sets its object name to "MyObject". Other common data fields for objects include cost and comments.

{button ,AL(`OVR1 Object Data Manager commands;',0,"Defaultoverview",)} Related Topics

Recorder commands

RecorderBeginEditText (DRAW)

.RecorderBeginEditText

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEndEditText (DRAW)

.RecorderEndEditText

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEditTextCharAttributes (DRAW)

.RecorderEditTextCharAttributes .lFirstSelectedChar = *long*, .lLastSelectedChar = *long*, .pszFontName = *string*, .lFontStyle = *long*, .lPointSize = *long*, .lUnderline = *long*, .lOverline = *long*, .lStrikeOut = *long*, .lPlacement = *long*, .lCharacterSpacing = *long*, .lWordSpacing = *long*, .lLineSpacing = *long*, .lAlignment = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEditTextChangeCase (DRAW)

.RecorderEditTextChangeCase .lCaseID = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEditTextReplaceText (DRAW)

.RecorderEditTextReplaceText .szNewText = *string*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderBeginEditParaText (DRAW)

.RecorderBeginEditParaText

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEndEditParaText (DRAW)

.RecorderEndEditParaText

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEditParaTextChangeCase (DRAW)

.RecorderEditParaTextChangeCase .lCaseID = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEditParaTextSpacing (DRAW)

.RecorderEditParaTextSpacing .lFirstSelectedChar = *long*, .lLastSelectedChar = *long*, .lCharacterSpacing = *long*, .lWordSpacing = *long*, .lLineSpacing = *long*, .lBeforeParagraph = *long*, .lAfterParagraph = *long*, .lAlignment = *long*, .bAutoHyphenation = *boolean*, .lHyphenHotZone = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEditParaTextIndents (DRAW)

.RecorderEditParaTextIndents .lFirstSelectedChar = *long*, .lLastSelectedChar = *long*, .lFirstLine = *long*, .lRestOfLines = *long*, .lRightMargin = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEditParaTextCharAttributes (DRAW)

.RecorderEditParaTextCharAttributes .lFirstSelectedChar = *long*, .lLastSelectedChar = *long*, .pszFontName = *string*, .lFontStyle = *long*, .lPointSize = *long*, .lUnderline = *long*, .lOverline = *long*, .lStrikeOut = *long*, .lPlacement = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderEditParaTextReplaceText (DRAW)

.RecorderEditParaTextReplaceText .szNewText = *string*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderApplyPerspective (DRAW)

.RecorderApplyPerspective .lType = *long*, .lFlags = *long*, .lBox0X = *long*, .lBox0Y = *long*, .lBox1X = *long*, .lBox1Y = *long*, .lBox2X = *long*, .lBox2Y = *long*, .lBox3X = *long*, .lBox3Y = *long*, .lVPHorizRef = *long*, .lVPHorizX = *long*, .lVPHorizY = *long*, .lVPVertRef = *long*, .lVPVertX = *long*, .lVPVertY = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderObjectScaleInfo (DRAW)

.RecorderObjectScaleInfo .ScaledSizeX = *long*, .ScaledSizeY = *long*, .DisplacementX = *long*, .DisplacementY = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderSelectObjectByIndex (DRAW)

.RecorderSelectObjectByIndex .bClearFirst = *boolean*, .lIndex = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderSelectObjectsByIndex (DRAW)

.RecorderSelectObjectsByIndex .bClearFirst = *boolean*, .lIndex1 = *long*, .lIndex2 = *long*, .lIndex3 = *long*, .lIndex4 = *long*, .lIndex5 = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderSelectPreselectedObjects (DRAW)

.RecorderSelectPreselectedObjects .bClearFirst = *boolean*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

RecorderStorePreselectedObjects (DRAW)

.RecorderStorePreselectedObjects .bConvertingPreset = *boolean*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

StartOfRecording (DRAW)

.StartOfRecording

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

EndOfRecording (DRAW)

.EndOfRecording

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

MenuCommand (DRAW)

.MenuCommand .lMenuID = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

ClickedDialogButton (DRAW)

.ClickedDialogButton .lDialogID = *long*, .lItemD = *long*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

ShareExtrudeVP (DRAW)

.ShareExtrudeVP .lExtrudeIndex = *long*, .lVPToShareIndex = *long*, .bSharedVP = *boolean*

This command is used by the Script And Preset Manager. You do not need to include this command in your script.

{button ,AL(`OVR1 Recorder commands;',0,"Defaultoverview",)} **Related Topics**

ID	Color Model	Color 1	Color 2	Color 3	Color 4
1	Pantone	Pantone ID number	Tint (0 - 100)	Ignored	Ignored
2	CMYK100	Cyan (0 - 100)	Magenta (0 - 100)	Yellow (0 - 100)	Black (0 - 100)
3	CMYK255	Cyan (0 - 255)	Magenta (0 - 255)	Yellow (0 - 255)	Black (0 - 255)
4	CMY	Cyan (0 - 255)	Magenta (0 - 255)	Yellow (0 - 255)	Ignored
5	RGB	Red (0 - 255)	Green (0 - 255)	Blue (0 - 255)	Ignored
6	HSB	Hue (0 - 360)	Saturation (0 - 255)	Brightness (0 - 255)	Ignored
7	HLS	Hue (0 - 360)	Lightness (0 - 255)	Saturation (0 - 255)	Ignored
8	Black and White	Black (0) or White (1)	Ignored	Ignored	Ignored
9	Grayscale	Black % (0-255)	Ignored	Ignored	Ignored
10	YIQ255	Y-luminance (0 - 255)	I-chromaticity (0 - 255)	Q-chromaticity (0 - 255)	Ignored
11	L*a*b*	L*-lightness (0 - 255)	a*-green to red (0 - 255)	b*-blue to yellow (0 - 255)	Ignored