

## **.AddPageFrame (DRAW)**

This command puts a printable background frame the same size as the page on the screen.

### **Syntax**

**.AddPageFrame**

### **Example**

```
.AddPageFrame
```

The above example creates a frame around the new page.

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.AlignObjects (DRAW)**

This command aligns selected objects.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.HorizontalAlignment</b>	Specifies the type of horizontal alignment. 0 = None 1 = Right 2 = Left 3 = Center
<b>.VerticalAlignment</b>	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(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.AlignToCenterOfPage (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.HorizontalAlignment</b>	Specifies the type of horizontal alignment. 0 = None 1 = Right 2 = Left 3 = Center 4 = Width
<b>.VerticalAlignment</b>	Specifies the type of vertical alignment. 0 = None 1 = Top 2 = Bottom 3 = Center 4 = Height

### **Example**

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

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

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.AlignToGrid (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.HorizontalAlignment</b>	Specifies the type of horizontal alignment. 0 = None 1 = Right 2 = Left 3 = Center 4 = Width
<b>.VerticalAlignment</b>	Specifies the type of vertical alignment. 0 = None 1 = Top 2 = Bottom 3 = Center 4 = Height

### **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(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.ApplyFountainFill (DRAW)**

This command lets you apply a Fountain Fill to the selected object. If the existing fill is fountain, the existing start/end color will be retained. 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**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Type</b>	Specifies the type of Fountain Fill to apply: 0 = Linear (default) 1 = Radial 2 = Conical 3 = Square
<b>.CenterX</b>	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.
<b>.CenterY</b>	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.
<b>.Angle</b>	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.
<b>.Steps</b>	Specifies 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.
<b>.Padding</b>	Specifies the amount of padding to apply to the fill. Ignored for type 2. Valid values range from 0 to 45 percent.
<b>.Blend</b>	Specifies the type of blending to apply to the fill. 0 = Direct (default) 1 = Rainbow CW 2 = Rainbow CCW 3 = Custom
<b>.Rate</b>	Specifies the mid-point used to apply the fill. Valid values range from 0 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.

The Angle option is not available for circular fountain fills.

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, 0  
.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(`DRAW\_Fill\_Menu;;;;;,0,"Defaultoverview",)} [Related Topics](#)**

## **.ApplyFullColorFill (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.FileName</b>	Specifies the name of the Fill file.
<b>.TileWidth</b>	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.
<b>.TileHeight</b>	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.
<b>.FirstTileOffsetX</b>	Specifies the amount of offset applied to the first tile along the x-axis. Valid values range from 0 to 100 percent.
<b>.FirstTileOffsetY</b>	Specifies the amount of offset applied to the first tile along the y-axis. Valid values range from 0 to 100 percent.
<b>.RowOffset</b>	Set to TRUE (-1) to enable row offset. Set to FALSE (0) to enable column offset.
<b>.RowColumnOffset</b>	Specifies the amount of row or column offsets. Valid values range from 0 to 100.
<b>.SeamlessTiling</b>	Set to TRUE (-1) to enable seamless tiling. Set to FALSE (0) to disable this option.
<b>.ScaleWithObject</b>	Set to TRUE (-1) to scale the pattern with the object. 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  
.ApplyFullColorFill "C:\COREL60\COLOR\MONTEMP.BMP", 500000, 500000, 100, 100, 0, 100, 0, 0
```

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

---

{button ,AL(`DRAW\_Fill\_Menu;;;;;`,0,"Defaultoverview",)} [Related Topics](#)

## **.ApplyNoFill (DRAW)**

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

### **Syntax**

**.ApplyNoFill**

### **Example**

```
.SelectAllObjects  
.ApplyNoFill
```

The above example removes the fill from all objects.

---

**{button ,AL(`DRAW\_Fill\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.ApplyOutline (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Width</b>	Specifies the width of the outline to apply, in tenths of a micron.
<b>.Type</b>	Specifies the outline type: 0 = None 1 = Solid 2 = Dot - Dash
<b>.EndCaps</b>	Specifies the end caps to be applied to the outline: 0 = Butt 1 = Round 2 = Square
<b>.JoinType</b>	Specifies the outline join types: 0 = Miter 1 = Round 2 = Bevel
<b>.Aspect</b>	Specifies the stretch field which adjusts the width of the nib. Valid values range from 1 to 100 percent.
<b>.Angle</b>	Specifies the angle of the nib's edge, in tenths of degrees.
<b>.DotDash</b>	Specifies the type of dot/dash line. Refer to the Style list of the Outline Pen dialog box for more details.
<b>.RightArrow</b>	Specifies the style of right-arrow. Refer to the Arrows list of the Outline Pen dialog box for more details.
<b>.LeftArrow</b>	Specifies the style of left-arrow. Refer to the Arrows list of the Outline Pen dialog box for more details.
<b>.BehindFill</b>	Set to TRUE (-1) to position the outline behind the fill. Set to FALSE (0) to position the outline in front of the fill.

### **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, 5, 6, 0
```

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

---

**{button ,AL(`DRAW\_Outline\_Menu;;;;',0,"Defaultoverview",)} Related Topics**



## **.ApplyPostscriptFill (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.PSFill</b>	Specifies the name of the postscript fill. The name must be preceded by an F/ (indicating full fill) or an S/ (indicating spot fill). Refer to the PostScript Texture dialog box for more details.
<b>.NumParms</b>	Specifies the number of parameters used for the selected PostScript Fill, an integer value between 0 and 5.
<b>.Parm1</b>	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 more details.
<b>.Parm2</b>	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 more details.
<b>.Parm3</b>	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 more details.
<b>.Parm4</b>	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 more details.
<b>.Parm5</b>	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 more details.

### **Example**

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

The above example applies a stone wall PostScript fill to the rectangle.

---

**{button ,AL(`DRAW\_Fill\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

## **.ApplyPreset (DRAW)**

This command lets you load and apply a Preset.

### **Syntax**

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

### **Syntax**

### **Definition**

---

**.PresetFileName** Specifies the name of the Preset File.

**.PresetName** Specifies the name of the Preset.

### **Example**

```
.CreateRectangle 1000000, -500000, -1000000, 500000, 0  
.ApplyPreset "C:\COREL60\DRAW\CORELDRW.PST", "Button Blue"
```

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

---

**{button ,AL(`DRAW\_Fill\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.ApplyStyle (DRAW)**

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

### **Syntax**

**.ApplyStyle .Style=*string***

### **Syntax**

### **Definition**

---

**.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(`DRAW\_Styles\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

## **.ApplyTextureFill (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.TextureLibrary</b>	Specifies the name of the Texture Library.
<b>.Texture</b>	Specifies the name of the texture.
<b>.Style</b>	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(`DRAW\_Fill\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.ApplyToDuplicate (DRAW)**

In Visual Basic (or any other programming application that supports properties), use .ApplyToDuplicate to open and close a block of object duplicating commands.

### **Syntax**

**.ApplyToDuplicate**

### **Note**

The Corel SCRIPT programming language does not support properties. Use the .SetApplyToDuplicate command in a Corel SCRIPT script to open and close a block of object duplicating commands.

### **Example**

```
.ApplyToDuplicate
```

The above example applies the current style to the duplicated object(s).

---









**{button ,AL(`DRAW\_Fill\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.ApplyTwoColorFill (DRAW)**

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

### **Syntax**

**.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**

<b>Syntax</b>	<b>Definition</b>
<b>.FileName</b>	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.
<b>.ColorModel1</b>	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
<b>.Color11</b>	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.
<b>.Color12</b>	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.
<b>.Color13</b>	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.
<b>.Color14</b>	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.
<b>.ColorModel2</b>	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
<b>.Color21</b>	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.
<b>.Color22</b>	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.
<b>.Color23</b>	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.
<b>.Color24</b>	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.
<b>.TileWidth</b>	Specifies the width of the tile, in tenths of a micron.
<b>.TileHeight</b>	Specifies the height of the tile, in tenths of a micron.
<b>.FirstTileOffsetX</b>	Specifies the amount of offset applied to the first tile along the x-axis. Valid values range from 0 to 100.

<b>.FirstTileOffsetY</b>	Specifies the amount of offset applied to the first tile along the y-axis. Valid values range from 0 to 100.
<b>.RowOffset</b>	Set to TRUE (-1) to enable row offset. Set to FALSE (0) to enable column offset.
<b>.RowColumnOffset</b>	Specifies the amount of row or column offsets. Valid values range from 0 to 100.
<b>.SeamlessTiling</b>	Set to TRUE (-1) to enable seamless tiling. Set to FALSE (0) to disable this option.
<b>.ScaleWithObject</b>	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(`DRAW\_Fill\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.ApplyUniformFillColor (DRAW)**

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

### **Syntax**

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

### **Syntax**


### **Definition**


#### **.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(`DRAW\_Fill\_Menu;;;;;`,0,"Defaultoverview",)} [Related Topics](#)**



## **.BeginDrawCurve (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.X</b>	Specifies the X coordinate of the starting node of the curve in tenths of a micron, relative to the origin.
<b>.Y</b>	Specifies the Y coordinate of the starting node of the curve in tenths of a micron, relative to the origin.

### **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(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.BreakApart (DRAW)**

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

### **Syntax**

**.BreakApart**

### **Example**

.BreakApart

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

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;;'0,"Defaultoverview",)} Related Topics**

## **.ChangeLayerName (DRAW)**

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

### **Syntax**

**.ChangeLayerName .LayerName=*string***

### **Syntax**

### **Definition**

---

**.LayerName**

Specifies the new name of the Layer.

### **Example**

```
.ChangeLayerName "NewName"
```

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

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.CloneObject (DRAW)**

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 be dependent 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.

### **Syntax**

#### **.CloneObject**

### **Example**

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

The above example creates an ellipse, then a clone of it.

---

**{button ,AL(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.Combine (DRAW)**

This command combines the selected curve/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, but larger blocks of text.

### **Syntax**

#### **.Combine**

### **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 four ellipses, then combines them together before applying the fill.

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)}** [Related Topics](#)

## **.ConvertToCurves (DRAW)**

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

### **Syntax**

**.ConvertToCurves**

### **Example**

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

The above example converts the selected rectangle to a series of curves lines.

---

**{button ,AL(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.CopyToClipboard (DRAW)**

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

### **Syntax**

#### **.CopyToClipboard**

### **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 to the third page.

---

**{button ,AL(`DRAW\_Edit\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.CopyToLayer (DRAW)**

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

### **Syntax**

**.CopyToLayer .LayerName=*string***

### **Syntax**

### **Definition**

---

**.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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**



## **.CreateArtisticText (DRAW)**

This command allows you to change the default artistic, paragraph, or both, text styles.

### **Syntax**

**.CreateArtisticText .NewText=*string***

### **Syntax**

### **Definition**

---

#### **.NewText**

Specifies the name of the new text to create.

### **Example**

```
.CreateArtisticText "COREL DRAW"
```

The above example displays the text string "COREL DRAW".

---

**{button ,AL(^DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.CreateEllipse (DRAW)**

This command is used to draw ellipses and circles.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Top</b>	Specifies the Y coordinate of the upper-left corner of the bounding rectangle of the ellipse in tenths of a micron, relative to the origin.
<b>.Left</b>	Specifies the X coordinate of the upper-left corner of the bounding rectangle of the ellipse in tenths of a micron, relative to the origin.
<b>.Bottom</b>	Specifies the Y coordinate of the lower-right corner of the bounding rectangle of the ellipse in tenths of a micron, relative to the origin.
<b>.Right</b>	Specifies the X coordinate of the lower-right corner of the bounding rectangle of the ellipse in tenths of a micron, relative to the origin.
<b>.StartAngle</b>	If .CreateEllipse is used to create an arc, .StartAngle specifies the starting angle in degrees.
<b>.EndAngle</b>	If .CreateEllipse is used to create an arc, .EndAngle specifies the end angle, in degrees.
<b>.Arc</b>	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(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} [Related Topics](#)**

## **.CreateRectangle (DRAW)**

This command draws rectangles and squares.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Top</b>	Specifies the Y coordinate of the upper-left corner of the rectangle in tenths of a micron, relative to the origin.
<b>.Left</b>	Specifies the X coordinate of the upper-left corner of the rectangle in tenths of a micron, relative to the origin.
<b>.Bottom</b>	Specifies the Y coordinate of the lower-right corner of the rectangle in tenths of a micron, relative to the origin.
<b>.Right</b>	Specifies the X coordinate of the lower-right corner of the rectangle in tenths of a micron, relative to the origin.
<b>.CornerRadius</b>	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(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.CurrentPage (DRAW)**

In Visual Basic (or any other programming application that supports properties), use .CurrentPage to display the current page in the image window.

### **Syntax**

**.CurrentPage**

### **Note**

The Corel SCRIPT programming language does not support properties. Use the .SetCurrentPage command in a Corel SCRIPT script to display the current page in the image window.

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)}** [Related Topics](#)

## **.DeleteLayer (DRAW)**

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

### **Syntax**

**.DeleteLayer**

### **Example**

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

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

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.DeleteObject (DRAW)**

This command deletes selected objects.

### **Syntax**

**.DeleteObject**

### **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(`DRAW\_ObjectCreation\_Menu;;;;';0,"Defaultoverview",)} [Related Topics](#)

## **.DeletePages (DRAW)**

This command deletes pages from the current drawing.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.BeforeCurrentPage</b>	Set to TRUE (-1) to enable deletion before the current page. Set to FALSE (0) to enable deletion after the current page.
<b>.NumberOfPages</b>	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(`DRAW\_Layout\_Menu;,,,','0,"Defaultoverview",)} Related Topics**

## **.DeleteStyle (DRAW)**

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**

**.DeleteStyle .Style=*string***

### **Syntax**

### **Definition**

---

**.Style**

Specifies the name of the style to delete.

### **Example**

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

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

---

**{button ,AL(`DRAW\_Styles\_Menu;;;;';,0,"Defaultoverview",)} Related Topics**



## **.DisplayFacingPages (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.FacingPages</b>	Set to TRUE (-1) to display two consecutive pages on the screen at the same time. Working this view allows you to draw objects that lie partially on both pages at once. Set to FALSE (0) to disable this option.
<b>.LeftFirst</b>	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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.DistributeObjects (DRAW)**

This command distributes selected objects.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.HorizontalDistribution</b>	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
<b>.VerticalDistribution</b>	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
<b>.ObjectOrPageExtents</b>	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(`DRAW\_ObjectCreation\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

## **.DrawCurveClosePath (DRAW)**

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

### **Syntax**

**.DrawCurveClosePath**

### **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(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.DrawCurveCurveTo (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.X1</b>	Specifies the X coordinate of the first control point of the curve in tenths of a micron, relative to the origin.
<b>.Y1</b>	Specifies the Y coordinate of the first control point of the curve in tenths of a micron, relative to the origin.
<b>.X2</b>	Specifies the X coordinate of the second control point of the curve in tenths of a micron, relative to the origin.
<b>.Y2</b>	Specifies the Y coordinate of the second control point of the curve in tenths of a micron, relative to the origin.
<b>.XEnd</b>	Specifies the X coordinate of the end control point of the curve in tenths of a micron, relative to the origin.
<b>.YEnd</b>	Specifies the Y coordinate of the end control point of the curve in tenths of a micron, relative to the origin.

### **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  
.EndDrawCurve
```

The above example draws a curve.

---

**{button ,AL(`DRAW\_ObjectCreation\_Menu;;;;;0,"Defaultoverview",)} [Related Topics](#)**

## **.DrawCurveLineTo (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.X</b>	Specifies the X coordinate of the next node of the curve in tenths of a micron, relative to the origin.
<b>.Y</b>	Specifies the Y coordinate of the next node of the curve in tenths of a micron, relative to the origin.

### **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(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.DrawCurveMoveTo (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.X</b>	Specifies the X coordinate of the point to move to without drawing in tenths of a micron, relative to the origin.
<b>.Y</b>	Specifies the Y coordinate of the point to move to without drawing in tenths of a micron, relative to the origin.

### **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(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.DropSymbol (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.SymbolLibrary</b>	Specifies the name of the Symbol Library. Refer to the Symbols dialog box for more details.
<b>.SymbolNumber</b>	Specifies the Symbol Index Number, which identifies the selected symbol. Refer to the Symbols dialog box for more details.
<b>.Tile</b>	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 top left symbol.
<b>.XPosOrGridSize</b>	Specifies the X coordinate or grid position at which to place the symbol, in tenths of a micron.
<b>.YPosOrGridSize</b>	Specifies the Y coordinate or grid position at which to place the symbol, in tenths of a micron.
<b>.ProportionalSizing</b>	Set to TRUE (-1) to enable proportional sizing of the symbol. Set to FALSE (0) to disable this option.
<b>.SymbolSize</b>	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.

---

**{button ,AL(`DRAW\_Symbols\_Menu;;;;;'0,"Defaultoverview",)} Related Topics**

## **.DuplicateObject (DRAW)**

This command adds a copy of the selected object(s) to the current drawing.

### **Syntax**

#### **.DuplicateObject**

### **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, fills it with a two color fountain fill, then duplicates it without filling the duplicate.

---

**{button ,AL(`DRAW\_ObjectCreation\_Menu;;;;';0,"Defaultoverview",)} Related Topics**



## **.EndDrawCurve (DRAW)**

This command ends the current curve being drawn.

### **Syntax**

**.EndDrawCurve**

### **Note**

The .EndDrawCurve command must be preceded by a .BeginDrawCurve command and a contiguous block of one or more DrawCurve commands. 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(`DRAW\_ObjectCreation\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.ExtractText (DRAW)**

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**

**.ExtractText .DestinationFile=*string***

### **Syntax**

### **Definition**

---

**.DestinationFile**

Specifies the name of the destination file.

### **Example**

```
.CreateArtisticText "COREL DRAW"  
.ExtractText "C:\COREL60\DRAW\TEXTFILE.TXT"
```

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

---

**{button ,AL(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.FileClose (DRAW)**

This command closes the current drawing.

### **Syntax**

**.FileClose**

### **Note**

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

### **Example**

```
.FileClose
```

The above example closes the active CorelDRAW document.

---

**{button ,AL(`DRAW\_File\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## .FileExport (DRAW)

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

### Syntax

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

Syntax	Definition
<b>.FileName</b>	Specifies the name of the file to export.
<b>.FilterID</b>	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 Bitmaps (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 (CTP) 1794 = Corel CMX 5.0 1793 = Corel CMX 6.0
<b>.Width</b>	Specifies the width of the image in pixels.
<b>.Height</b>	Specifies the height of the image in pixels.
<b>.XResolution</b>	Specifies the horizontal resolution of the image in dots per inch (dpi).
<b>.YResolution</b>	Specifies the vertical resolution of the image in dots per inch (dpi).
<b>.ImageType</b>	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:\COREL60\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(`DRAW\_File\_Menu;;;;;' ,0,"Defaultoverview" ,)} [Related Topics](#)

## **.FileImport (DRAW)**

This command brings graphics into CorelDRAW from other programs.

### **Syntax**

**.FileImport .FileName=*string***

### **Syntax**

### **Definition**

---

**.FileName**

Specifies the name of the file to import.

### **Example**

```
.FileNew  
.FileImport "C:\COREL60\DRAW\TEST1.BMP"
```

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

---

**{button ,AL(`DRAW\_File\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.FileNew (DRAW)**

This command creates a new drawing. If you have a drawing open, the new drawing opens over the current drawing. The new drawing uses the same program settings that were in effect for the previous drawing (Page Setup, View options, New Object Fill and Outline attributes, etc.).

### **Syntax**

**.FileNew**

### **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(`DRAW\_File\_Menu;;;;';0,"Defaultoverview",)} Related Topics**

## **.FileOpen (DRAW)**

This command loads a drawing or Styles Template into CorelDRAW.

### **Syntax**

**.FileOpen .FileName=*string***

### **Syntax**

### **Definition**

---

#### **.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:\COREL60\DRAW\TEST1.CDR"
```

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

---

**{button ,AL(`DRAW\_File\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.FilePrint (DRAW)**

This command prints the currently displayed file.

### **Syntax**

**.FilePrint**

### **Example**

```
.FilePrint
```

The above example sends the current document to the printer.

---

**{button ,AL(`DRAW\_File\_Menu;;;;','0,"Defaultoverview",)} Related Topics**



## .FileSave (DRAW)

This command saves the active document.

### Syntax

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

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

### Example

```
.FileSave "C:\COREL60\DRAW\TEST1.CDR", 1, 0, 0, 0
```

The above example saves a version 6 CorelDRAW document named "TEST1.CDR", with a 1k thumbnail. CMX data is not saved.

---

{button ,AL(`DRAW\_File\_Menu;;;;;','0,"Defaultoverview",)} [Related Topics](#)

## **.GetFillType (DRAW)**

This function determines the Fill Type.

0 = None  
1 = Uniform  
2 = Fountain  
6 = PostScript  
7 = MonoBitmap1  
8 = MonoBitmap2  
9 = ColorBitmap  
10 = Vector  
11 = Texture

### **Syntax**

#### **.GetFillType**

### **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(`DRAW\_Fill\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

## **.GetFountainFill (DRAW)**

This function determines the Fountain Fill attributes.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Type</b>	Returns the type of Fountain Fill: 0 = Linear (default) 1 = Radial 2 = Conical 3 = Square
<b>.CenterX</b>	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.
<b>.CenterY</b>	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.
<b>.Angle</b>	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.
<b>.Steps</b>	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.
<b>.Padding</b>	Returns the amount of padding to apply to the fill. Ignored for type 2. Valid values range from 0 to 45 percent.
<b>.Blend</b>	Returns the type of blending to apply to the fill. 0 = Direct (default) 1 = Rainbow CW 2 = Rainbow CCW 3 = Custom
<b>.Rate</b>	Returns the rate method used to apply the fill.
<b>.NumColors</b>	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 obtains information about the current fountain fill and displays a number corresponding to the the fill type in a message box.

---

**{button ,AL(`DRAW\_Fill\_Menu;;;;',0,"Defaultoverview",)} [Related Topics](#)**

## **.GetObjectsCDRStaticID (DRAW)**

This function determines the CDRStaticID of the selected object.

### **Syntax**

**.GetObjectsCDRStaticID**

### **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(`DRAW\_ObjectSelection\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

## **.GetOutline (DRAW)**

This function determines the attributes of the active outline.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Width</b>	Returns the width of the outline, in tenths of a micron.
<b>.Type</b>	Returns the outline type: 0 = None 1 = Solid 2 = Dot - Dash
<b>.EndCaps</b>	Returns the End Caps applied to the outline: 0 = Butt 1 = Round 2 = Square
<b>.JoinType</b>	Returns the outline join types: 0 = Miter 1 = Round 2 = Bevel
<b>.Aspect</b>	Returns the stretch field which adjusts the width of the nib.
<b>.Angle</b>	Returns the angle of the nib's edge, in tenths of degrees.
<b>.DotDash</b>	Returns the type of dot/dash line. Refer to the Outline Pen dialog box for more details.
<b>.RightArrow</b>	Returns the style of right-arrow. Refer to the Outline Pen dialog box for more details.
<b>.LeftArrow</b>	Returns the style of left-arrow. Refer to the Outline Pen dialog box for more details.
<b>.BehindFill</b>	Returns the position of the outline fill. TRUE (-1) = Outline behind fill FALSE (0) = Outline in front of fill

### **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 determines the outline attributes of the selected object.

---

**{button ,AL(`DRAW\_Outline\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## .GetOutlineColor (DRAW)

This function determines the currently Outline Color.

### Syntax

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

### Syntax


### Definition


#### .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(`DRAW\_Outline\_Menu;;;;';0,"Defaultoverview",)} [Related Topics](#)**

## **.GetPosition (DRAW)**

This function determines the position of the selected object.

### **Syntax**

**.GetPosition .XPos=*long*, .YPos=*long***

<b>Syntax</b>	<b>Definition</b>
<b>.XPos</b>	Returns the X coordinate of the selected object in tenths of a micron, relative to the origin.
<b>.YPos</b>	Returns the Y coordinate of the selected object in tenths of a micron, relative to the origin.

### **Example**

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

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

---

**{button ,AL(`DRAW\_Transformation\_Menu;;;;',0,"Defaultoverview",)} [Related Topics](#)**

## **.GetSize (DRAW)**

This function determines the size of the selected object.

### **Syntax**

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

### **Syntax**

### **Definition**

#### **.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, 500000, 100000, 0  
id& = .GetObjectsCDRStaticID()  
status& = .GetSize (XSize&, YSize&)  
MESSAGE "Horizontal"+STR(XSize&)  
MESSAGE "Vertical"+STR(YSize&)
```

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

---

**{button ,AL(`DRAW\_Transformation\_Menu;;;;';0,"Defaultoverview",)} [Related Topics](#)**



## **.GetUniformFillColor (DRAW)**

This function determines the Uniform Fill color.

### **Syntax**

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

### **Syntax**


### **Definition**


#### **.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 current uniform fill color and displays a number corresponding to the color model in a message box.

---

**{button ,AL(`DRAW\_Fill\_Menu;;;;;','0,"Defaultoverview",)} [Related Topics](#)**

## **.GetUserDataField (DRAW)**

This function determines the name of the User Data File.

### **Syntax**

**.GetUserDataField .FieldName=string**

### **Syntax**

### **Definition**

---

**.FieldName** Returns the name of the user data field.

### **Example**

```
.GetUserDataField Field Name$
```

The above example determines the field name of the user data field.

---

**{button ,AL(`DRAW\_ObjectDataManager\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.Group (DRAW)**

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

### **Syntax**

#### **.Group**

### **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 to all four.

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.InsertPages (DRAW)**

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

### **Syntax**

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

### **Syntax**

### **Definition**

---

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

### **Example**

```
.InsertPages 0, 4
```

The above example inserts 4 pages after the current page.

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.Intersection (DRAW)**

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.

### **Syntax**

#### **.Intersection**

### **Example**

```
.SelectAllObjects  
.Intersection
```

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

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.LoadStyles (DRAW)**

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

### **Syntax**

**.LoadStyles .StyleSheet=*string***

### **Syntax**

### **Definition**

---

**.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 DRAW document.

---

**{button ,AL(`DRAW\_Styles\_Menu;;;;';0,"Defaultoverview",)} Related Topics**

## **.MergeBackText (DRAW)**

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

### **Syntax**

**.MergeBackText .SourceFile=*string***

### **Syntax**

### **Definition**

---

#### **.SourceFile**

Specifies the name of the source file to merge.

### **Example**

```
.CreateArtisticText "COREL DRAW"  
.ExtractText "C:\COREL60\DRAW\TEXTFILE.TXT"  
.MergeBackText "C:\COREL60\DRAW\TEXTFILE.TXT"
```

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

---

**{button ,AL(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.MoveObject (DRAW)**

This command repositions the selected object to the specified location.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.XDelta</b>	Specifies the distance the object is to be moved along the x-axis in tenths of a micron.
<b>.YDelta</b>	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(`DRAW\_Transformation\_Menu;;;;';0,"Defaultoverview",)} Related Topics**



## **.MoveToLayer (DRAW)**

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

### **Syntax**

**.MoveToLayer .LayerName=*string***

### **Syntax**

### **Definition**

---

**.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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.NewLayer (DRAW)**

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

### **Syntax**

**.NewLayer .LayerName=*string***

### **Syntax**

### **Definition**

---

**.LayerName**

Specifies the name of the new Layer.

### **Example**

```
.NewLayer "NewLayer1"
```

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

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.OrderBackOne (DRAW)**

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

### **Syntax**

**.OrderBackOne**

### **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(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.OrderForwardOne (DRAW)**

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

### **Syntax**

**.OrderForwardOne**

### **Example**

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

The above example orders the selected object forward one.

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.OrderReverseOrder (DRAW)**

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

### **Syntax**

**.OrderReverseOrder**

### **Example**

```
.SelectAllObjects  
.OrderReverseOrder
```

The above example reverses the order of all the objects.

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.OrderToBack (DRAW)**

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.

### **Syntax**

#### **.OrderToBack**

### **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(`DRAW\_Arrange\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.OrderToFront (DRAW)**

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

### **Syntax**

**.OrderToFront**

### **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(`DRAW\_Arrange\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.PasteFromClipboard (DRAW)**

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

### **Syntax**

#### **.PasteFromClipboard**

### **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 to last page inserted.

---

**{button ,AL(`DRAW\_Edit\_Menu;;;;','0,"Defaultoverview",)} Related Topics**



## **.Redo (DRAW)**

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

### **Syntax**

**.Redo**

### **Example**

.Redo

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

---

**{button ,AL(`DRAW\_Edit\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.ReferencePoint (DRAW)**

In Visual Basic (or any other programming application that supports properties), use .ReferencePoint to specify a fixed starting point for an operation.

### **Syntax**

**.ReferencePoint**

### **Note**

The Corel SCRIPT programming language does not support properties. Use the .SetReferencePoint command in a Corel SCRIPT script to set the reference point.

---

**{button ,AL(`DRAW\_ObjectDataManager\_Menu;;;;';0,"Defaultoverview",)} Related Topics**

## **.RemoveFountainFillColor (DRAW)**

This command removes the currently selected Fountain Fill Color.

### **Syntax**

**.RemoveFountainFillColor .Position=*long***

### **Syntax**

### **Definition**

#### **.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(`DRAW\_Fill\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.Repeat (DRAW)**

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

### **Syntax**

**.Repeat**

### **Example**

.Repeat

The above example repeats the last command.

---

**{button ,AL(`DRAW\_Edit\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.RotateObject (DRAW)**

This command lets you rotate the selected object.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Angle</b>	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
<b>.UseObjectsCenter</b>	Set to TRUE (-1) to enable rotation around the center of the object. Set to FALSE (0) to disable this option.
<b>.XCenter</b>	Specifies the logical X coordinate of the center of the object to be rotated in tenths of a micron, relative to the origin.
<b>.YCenter</b>	Specifies the logical Y coordinate of the center of the object to be rotated in tenths of a micron, relative to the origin.

### **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(`DRAW\_Transformation\_Menu;;;;';,0,"Defaultoverview",)} Related Topics**

## **.SaveTemplate (DRAW)**

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

### **Syntax**

**.SaveTemplate .StyleSheet=*string***

### **Syntax**

### **Definition**

---

**.StyleSheet**

Specifies the name of the Style Sheet to save.

### **Example**

```
.SaveTemplate "C:\COREL60\DRAW\TMPLATE1.CDT"
```

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

---

**{button ,AL(`DRAW\_Styles\_Menu;;;;';,0,"Defaultoverview",)} Related Topics**

## **.SelectAllObjects (DRAW)**

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

### **Syntax**

**.SelectAllObjects**

### **Example**

```
.SelectAllObjects
```

The above example selects all objects in the current image.

---

**{button ,AL(`DRAW\_ObjectSelection\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SelectLayer (DRAW)**

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

### **Syntax**

**.SelectLayer .LayerName=*string***

### **Syntax**

### **Definition**

---

**.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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**



## **.SelectNextObject (DRAW)**

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

### **Syntax**

**.SelectNextObject .SelectInsideGroup=*boolean***

### **Syntax**

### **Definition**

---

#### **.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 group.

---

**{button ,AL(`DRAW\_ObjectSelection\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.SelectObjectAtPoint (DRAW)**

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**

**.SelectObjectAtPoint .XPos=*long*, .YPos=*long*, .SelectInsideGroup=*boolean***

<b>Syntax</b>	<b>Definition</b>
<b>.XPos</b>	Specifies one of the X coordinates of the selected object in tenths of a micron, relative to the origin.
<b>.YPos</b>	Specifies one of the Y coordinates of the selected object in tenths of a micron, relative to the origin.
<b>.SelectInsideGroup</b>	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(`DRAW\_ObjectSelection\_Menu;;;;;','0,"Defaultoverview",)} [Related Topics](#)**

## **.SelectObjectOfCDRStaticID (DRAW)**

This command selects the object with the specified CDRStaticID.

### **Syntax**

**.SelectObjectOfCDRStaticID .CDRStaticID=*long***

### **Syntax**

### **Definition**

---

**.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(`DRAW\_ObjectSelection\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

## **.SelectObjectsInRect (DRAW)**

This command selects all objects found within the defined rectangular area

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Top</b>	Specifies the Y coordinate of the upper-left corner of the distribution rectangle in tenths of a micron, relative to the origin.
<b>.Left</b>	Specifies the X coordinate of the upper-left corner of the distribution rectangle in tenths of a micron, relative to the origin.
<b>.Bottom</b>	Specifies the Y coordinate of the lower-right corner of the distribution rectangle in tenths of a micron, relative to the origin.
<b>.Right</b>	Specifies the X coordinate of the lower-right corner of the distribution rectangle in tenths of a micron, relative to the origin.
<b>.IncludeIntersecting</b>	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(`DRAW\_ObjectSelection\_Menu;;;;',0,"Defaultoverview",)} [Related Topics](#)**

## **.SelectPreviousObject (DRAW)**

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**

**.SelectPreviousObject .SelectInsideGroup=*boolean***

### **Syntax**

### **Definition**

#### **.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(`DRAW\_ObjectSelection\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.Separate (DRAW)**

This command separates original objects from intermediate shapes.

### **Syntax**

**.Separate**

### **Example**

.Separate

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

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;;'0,"Defaultoverview",)} Related Topics**

## **.SetArtisticText (DRAW)**

This command sets the artistic text.

### **Syntax**

**.SetArtisticText .NewText=*string***

### **Syntax**

### **Definition**

---

**.NewText**

Specifies the name of the new text to set.

### **Example**

```
.CreateArtisticText "1"
```

The above example creates the number '1' with the default text settings.

---

**{button ,AL(`DRAW\_Text\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetColorOverride (DRAW)**

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**

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

### **Syntax**

### **Definition**

---

<b>.Override</b>	Set to TRUE (-1) to outline objects on a layer in the selected color. Set to FALSE (0) to disable this option.
<b>.ColorModel</b>	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
<b>.Color1</b>	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.
<b>.Color2</b>	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.
<b>.Color3</b>	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.
<b>.Color4</b>	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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**







## .SetFountainFillColor (DRAW)

This command sets the Fountain Fill Color.

### Syntax

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

Syntax	Definition
<b>.Position</b>	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 position is not 0 or 100, Blend is forced to be custom.
<b>.ColorModel</b>	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
<b>.Color1</b>	Specifies the first color component for .ColorModel. For example, Hue is the first color component for HSB. Click  for valid value ranges.
<b>.Color2</b>	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.
<b>.Color3</b>	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.
<b>.Color4</b>	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('DRAW\_Fill\_Menu;;;;','0,"Defaultoverview",)} [Related Topics](#)**

## **.SetLayerLocked (DRAW)**

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**

**.SetLayerLocked .Locked=*boolean***

### **Syntax**

### **Definition**

#### **.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(`DRAW\_Layout\_Menu;;;;';,0,"Defaultoverview",)} [Related Topics](#)**

## **.SetLayerPrintable (DRAW)**

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

### **Syntax**

**.SetLayerPrintable .Printable=*boolean***

### **Syntax**

### **Definition**

#### **.Printable**

Set to TRUE (-1) to enable current layer printing. 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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetLayerVisible (DRAW)**

This command makes objects on a layer visible or invisible.

### **Syntax**

**.SetLayerVisible .Visible=*boolean***

### **Syntax**

### **Definition**

#### **.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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetMultiLayer (DRAW)**

This command lets you select objects across all layers except those which are locked or invisible.

### **Syntax**

**.SetMultiLayer .MultiLayer=*boolean***

### **Syntax**

### **Definition**

#### **.MultiLayer**

Set to TRUE (-1) to select objects across all layers except those which are locked or invisible. Set to FALSE (0) to select all objects.

### **Example**

```
.SetMultiLayer -1
```

The above example sets multiple layers.

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetOptionsForAllPages (DRAW)**

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

### **Syntax**

**.SetOptionsForAllPages .AllPages=*boolean***

### **Syntax**

### **Definition**

#### **.AllPages**

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

### **Example**

```
.SetOptionsForAllPages -1
```

The above example applies the set options to all pages.

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetOutlineColor (DRAW)**

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

### **Syntax**

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

### **Syntax**


### **Definition**


#### **.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(`DRAW\_Outline\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetPageLayout (DRAW)**

This command lets you specify a page layout.

### **Syntax**

**.SetPageLayout .LayoutType=*long***

### **Syntax**

### **Definition**

#### **.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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**



## **.SetPageSize (DRAW)**

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

### **Syntax**

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

### **Syntax**

### **Definition**

---

#### **.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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetPageSizeFromPrinter (DRAW)**

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

### **Syntax**

**.SetPageSizeFromPrinter**

### **Example**

```
.SetPageSizeFromPrinter
```

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

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.SetPaperColor (DRAW)**

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**

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

### **Syntax**


### **Definition**


#### **.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(`DRAW\_Layout\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.SetPosition (DRAW)**

This command sets the position for placement of the selected object

### **Syntax**

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

### **Syntax**

### **Definition**

---

#### **.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(`DRAW\_Layout\_Menu;;;;';0,"Defaultoverview",)} Related Topics**

## **.SetSize (DRAW)**

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

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.XSize</b>	Specifies the new horizontal size of the selected object, in tenths of a micron.
<b>.YSize</b>	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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetToMasterLayer (DRAW)**

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**

**.SetToMasterLayer .Master=*boolean***

### **Syntax**

### **Definition**

#### **.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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetUserDataField (DRAW)**

This command lets you set object data values for selected objects.

### **Syntax**

**.SetUserDataField .FieldName=*string*, .FieldValue=*string***

### **Syntax**

### **Definition**

---

**.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(`DRAW\_ObjectDataManager\_Menu;;;;';0,"Defaultoverview",)} Related Topics**

## **.ShowPageBorder (DRAW)**

This command enables and disables the page border.

### **Syntax**

**.ShowPageBorder .ShowBorder=*boolean***

### **Syntax**

### **Definition**

---

#### **.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(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)} Related Topics**



## **.SkewObject (DRAW)**

This command lets you skew the selected object.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.XAngle</b>	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.
<b>.YAngle</b>	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.
<b>.Reference</b>	Specifies the reference point of the object to be skewed. 1 = Top-right 2 = Top-middle 3 = Top-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 top-left position.

---

**{button ,AL(`DRAW\_Transformation\_Menu;;;;;`,0,"Defaultoverview",)} Related Topics**

## **.Trim (DRAW)**

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.

### **Syntax**

**.Trim**

### **Example**

```
.SelectAllObjects  
.Trim
```

The above example trims the selected objects.

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.UnselectAll (DRAW)**

This command deselects all objects.

### **Syntax**

**.UnselectAll**

### **Example**

```
.UnselectAll
```

The above example deselects all selected object(s).

---

**{button ,AL(`DRAW\_ObjectSelection\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

## **.Undo (DRAW)**

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

### **Syntax**

**.Undo**

### **Example**

```
.Undo
```

The above example undoes the last command.

---

**{button ,AL(`DRAW\_Edit\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.Ungroup (DRAW)**

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

### **Syntax**

**.Ungroup**

### **Example**

.Ungroup

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

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.Weld (DRAW)**

This command joins overlapping objects at points where their paths intersect. Though 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.

### **Syntax**

**.Weld**

### **Example**

```
.SelectAllObjects  
.Weld
```

The above example welds the selected object group together.

---

**{button ,AL(`DRAW\_Arrange\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.FullScreenPreview (DRAW)**

In Visual Basic (or any other programming application that supports properties), use .FullScreenPreview to remove everything but your drawing from the screen. You cannot edit your drawing in this mode. Set to TRUE (-1) to remove everything but your drawing from the screen. Set to FALSE (0) to return to normal mode.

### **Syntax**

**.FullScreenPreview**

### **Note**

The Corel SCRIPT programming language does not support properties. Use the .SetFullScreenPreview command in a Corel SCRIPT script to remove everything but your drawing from the screen.

---

**{button ,AL(`DRAW\_View\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.Visible (DRAW)**

In Visual Basic (or any other programming application that supports properties), use .Visible to make DRAW visible.

### **Syntax**

**.Visible TRUE**

### **Note**

The Corel SCRIPT programming language does not support properties. Use the .SetVisible command in a Corel SCRIPT script to make CorelDRAW hidden or visible.

---

**{button ,AL(`DRAW\_View\_Menu;;;;','0,"Defaultoverview",)} Related Topics**



## **.SetCharacterAttributes (DRAW)**

This command sets the text character attributes.

### **Syntax**

**.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***

<b>Syntax</b>	<b>Definition</b>
<b>.FirstSelectedChar</b>	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
<b>.LastSelectedChar</b>	Specifies the ending character of the selected text.
<b>.FontName</b>	Specifies the font name.
<b>.FontStyle</b>	Specifies the style of the selected font. 7 = Normal 8 = Normal/Italic 13 = Bold 14 = Bold/Italic
<b>.PointSize</b>	Specifies the size of the selected font in tenths of a point.
<b>.Underline</b>	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
<b>.Overline</b>	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
<b>.StrikeOut</b>	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
<b>.Placement</b>	Specifies the placement of the font. 0 = Normal 1 = Superscript 2 = Subscript
<b>.CharacterSpacing</b>	Specifies the character spacing in tenths of a percent.
<b>.WordSpacing</b>	Specifies the word spacing in tenths of a percent.
<b>.LineSpacing</b>	Specifies the line spacing in tenths of a percent.
<b>.Alignment</b>	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(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.SetParagraphSpacing (DRAW)**

This command sets paragraph spacing.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.FirstSelectedChar</b>	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
<b>.LastSelectedChar</b>	Specifies the ending character of the selected text.
<b>.CharacterSpacing</b>	Specifies the character spacing in tenths of a percent.
<b>.WordSpacing</b>	Specifies the word spacing in tenths of a percent.
<b>.LineSpacing</b>	Specifies the line spacing in tenths of a percent.
<b>.BeforeParagraph</b>	Specifies the spacing before paragraphs in tenths of a percent.
<b>.AfterParagraph</b>	Specifies the spacing after paragraphs in tenths of a percent.
<b>.Alignment</b>	Specifies the alignment. 0 = None 1 = Left 2 = Center 3 = Right 4 = Full justify 5 = Force justify
<b>.AutoHyphenation</b>	Set to TRUE (-1) to enable automatic hyphenation. Set to FALSE (0) to disable this option.
<b>.HyphenHotZone</b>	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(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.AddTabStop (DRAW)**

This command adds tab stops to text.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.FirstSelectedChar</b>	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
<b>.LastSelectedChar</b>	Specifies the ending character of the selected text.
<b>.TabStop</b>	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(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.SetIndents (DRAW)**

This command sets indents for text.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.FirstSelectedChar</b>	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
<b>.LastSelectedChar</b>	Specifies the ending character of the selected text.
<b>.FirstLine</b>	Specifies the size of the first line indentation, in tenths of a micron.
<b>.RestOfLines</b>	Specifies the size of the remaining line indentation, in tenths of a micron.
<b>.RightMargin</b>	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(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.SetBullet (DRAW)**

This command sets bullets for text.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.FirstSelectedChar</b>	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
<b>.LastSelectedChar</b>	Specifies the ending character of the selected text.
<b>.SymbolLibrary</b>	Specifies the name of the symbol library. Refer to the Effects tab of the Paragraph dialog box for more details.
<b>.SymbolNumber</b>	Specifies the selected symbol number. Refer to the Effects tab of the Paragraph dialog box for more details.
<b>.PointSize</b>	Specifies the point size in tenths of a point.
<b>.BulletIndent</b>	Specifies the size of the bullet indentation in tenths of a micron.
<b>.VerticalShift</b>	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(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.SetFrameColumn (DRAW)**

This command formats columns for text.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.ColumnNumber</b>	Specifies the column number.
<b>.Width</b>	Specifies the width of the column in tenths of a micron.
<b>.GutterWidth</b>	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(`DRAW\_Text\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetTextString (DRAW)**

This command sets the text string.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.FirstSelectedChar</b>	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
<b>.LastSelectedChar</b>	Specifies the ending character of the selected text.
<b>.Text</b>	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(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} Related Topics**



## **.AlignTextToBaseline (DRAW)**

This command aligns text to the baseline.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.FirstSelectedChar</b>	Specifies the starting character of the selected text. Note: The first character in a string is equal to 0, not 1.
<b>.LastSelectedChar</b>	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(`DRAW\_Text\_Menu;;;','0,"Defaultoverview",)} Related Topics**

## **.StraightenText (DRAW)**

This command straightens text.

### **Syntax**

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

### **Syntax**

### **Definition**

#### **.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  
.StraightenText 35, 157
```

The above example straightens the text.

---

**{button ,AL(`DRAW\_Text\_Menu;;;','0,"Defaultoverview",)} Related Topics**

## **.CreateTextString (DRAW)**

This command creates the text.

### **Syntax**

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

<b>Syntax</b>	<b>Definition</b>
<b>.Top</b>	Specifies the Y coordinate of the upper-left corner of the text's bounding rectangle in tenths of a micron, relative to the origin.
<b>.Left</b>	Specifies the X coordinate of the upper-left corner of the text's bounding rectangle in tenths of a micron, relative to the origin.
<b>.Bottom</b>	Specifies the Y coordinate of the lower-right corner of the text's bounding rectangle in tenths of a micron, relative to the origin.
<b>.Right</b>	Specifies the X coordinate of the lower-right corner of the text's bounding rectangle in tenths of a micron, relative to the origin.
<b>.Text</b>	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(`DRAW\_Text\_Menu;;;;',0,"Defaultoverview",)} [Related Topics](#)**

## **.SetVisible (DRAW)**

This command makes the CorelDRAW application visible.

### **Syntax**

**.SetVisible .Visible=*boolean***

### **Syntax**

### **Definition**

---

#### **.Visible**

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

### **Example**

```
.SetVisible -1
```

The above example makes the CorelDRAW application visible.

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)}** [Related Topics](#)

## **.SetFullScreenPreview (DRAW)**

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

### **Syntax**

**.SetFullScreenPreview .FullScreen=*boolean***

### **Syntax**

### **Definition**

#### **.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 current image.

---

**{button ,AL(`DRAW\_View\_Menu;;;;','0,"Defaultoverview",)} Related Topics**

## **.SetCurrentPage (DRAW)**

This command makes the specified page the current page.

### **Syntax**

**.SetCurrentPage .CurrentPage=*long***

### **Syntax**

### **Definition**

---

**.CurrentPage**

Specifies the page number to make the current page.

### **Example**

```
.SetCurrentPage 2
```

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

---

**{button ,AL(`DRAW\_Layout\_Menu;;;;','0,"Defaultoverview",)}** [Related Topics](#)

## **.SetReferencePoint (DRAW)**

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**

**.SetReferencePoint .ReferencePoint=*long***

<b>Syntax</b>	<b>Definition</b>
<b>.ReferencePoint</b>	Specifies the reference point to set. 1 = Top-left 2 = Top-middle 3 = Top-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(`DRAW\_Layout\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.SetApplyToDuplicate (DRAW)**

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 re-positioned, resized, skewed, or rotated.

### **Syntax**

**.SetApplyToDuplicate .ApplyToDuplicate=*boolean***

### **Syntax**

### **Definition**

---

<b>.ApplyToDuplicate</b>	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(`DRAW\_Layout\_Menu;;;','0,"Defaultoverview",)} Related Topics**



## **.InsertOLEObject (DRAW)**

This command inserts an OLE object in a CorelDRAW document.

### **Syntax**

**.InsertOLEObject .ProgID=*string***

### **Syntax**

### **Definition**

---

#### **.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(`DRAW\_Edit\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.InsertOLEObjectFromFile (DRAW)**

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

### **Syntax**

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

### **Syntax**

### **Definition**

---

#### **.FileName**

The file name.

#### **.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(`DRAW\_Edit\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

## **.OLEObjectDoVerb (DRAW)**

This command performs the specified action on an OLE object.

### **Syntax**

**.OLEObjectDoVerb .Verb=*long***

### **Syntax**

### **Definition**

---

#### **.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.6"  
.OLEObjectDoVerb 0
```

The above example insets a Corel PHOTO-PAINT OLE object into a DRAW document and invokes inplace editing.

---

**{button ,AL(`DRAW\_Edit\_Menu;;;;';0,"Defaultoverview",)} Related Topics**

## **.PasteSystemClipboardFormat (DRAW)**

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

### **Syntax**

**.PasteSystemClipboardFormat .Format=*long***

<b>Syntax</b>	<b>Definition</b>
<b>.Format</b>	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(`DRAW\_Edit\_Menu;;;','0,"Defaultoverview",)} Related Topics**

## **.PasteCustomClipboardFormat (DRAW)**

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

### **Syntax**

**.PasteCustomClipboardFormat .Format=*string***

### **Syntax**

### **Definition**

#### **.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(`DRAW\_Edit\_Menu;;;;',0,"Defaultoverview",)} Related Topics**

## **.GetObjectType (DRAW)**

This function returns a value indicating the type of selected object.

0 = Other  
1 = Rectangle  
2 = Ellipse  
3 = Curve  
4 = Text  
5 = Bitmap  
6 = Paragraph Text  
7 = OLE  
9 = Symmetrical Polygon  
12 = A group of objects are selected

### **Syntax**

**.GetObjectType ( )**

### **Example**

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

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

---

**{button ,AL(`DRAW\_ObjectSelection\_Menu;;;;;','0,"Defaultoverview",)} Related Topics**

<b>ID</b>	<b>Color Model</b>	<b>Color 1</b>	<b>Color 2</b>	<b>Color 3</b>	<b>Color 4</b>
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

<b>ID</b>	<b>Color Model</b>	<b>Color 1</b>	<b>Color 2</b>	<b>Color 3</b>	<b>Color 4</b>
3	CMYK255	Cyan (0 - 255)	Magenta (0 - 255)	Yellow (0 - 255)	Black (0 - 255)
5	RGB	Red (0 - 255)	Green (0 - 255)	Blue (0 - 255)	Ignored
8	Black and White	Black (0) or White (1)	Ignored	Ignored	Ignored
9	Grayscale	Black % (0-255)	Ignored	Ignored	Ignored



