A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z

A

AdjustEffectInfo (PHOTO-PAINT)
AlphaOrder (PHOTO-PAINT)

B

BindToActiveDocument (PHOTO-PAINT)
BrushDabSettings (PHOTO-PAINT)
BrushOrbitSettings (PHOTO-PAINT)
BrushTextureSettings (PHOTO-PAINT)
BrushTool (PHOTO-PAINT)

C

ChannelNew (PHOTO-PAINT)
ChannelProperties (PHOTO-PAINT)
CloneTool (PHOTO-PAINT)
ColorMaskColor (PHOTO-PAINT)
ColorMaskCreateChannel (PHOTO-PAINT)
ColorMaskCreateMask (PHOTO-PAINT)
ColorMaskReset (PHOTO-PAINT)
ColorReplace (PHOTO-PAINT)
ColorReplacerTool (PHOTO-PAINT)
ContinueDraw (PHOTO-PAINT)

CreateBackground (PHOTO-PAINT)

D

DuotoneHandle (PHOTO-PAINT)
DuotoneInfo (PHOTO-PAINT)

E

EditCheckpoint (PHOTO-PAINT)
EditClear (PHOTO-PAINT)
EditClearClipboard (PHOTO-PAINT)
EditCopy (PHOTO-PAINT)
EditCopyToFile (PHOTO-PAINT)
EditCopyVisible (PHOTO-PAINT)
EditCut (PHOTO-PAINT)
EditCutIntoSelection (PHOTO-PAINT)
EditCutMask (PHOTO-PAINT)
EditCutSelection (PHOTO-PAINT)
EditFill (PHOTO-PAINT)
EditPasteDocument (PHOTO-PAINT)
EditPasteFromFile (PHOTO-PAINT)
EditPasteIntoSelection (PHOTO-PAINT)
EditPasteObject (PHOTO-PAINT)
EditPasteSelection (PHOTO-PAINT)
EditRestoreCheckpoint (PHOTO-PAINT)
EditSingleObject (PHOTO-PAINT)
Effect3DRotate (PHOTO-PAINT)
Effect3DStereoNoise (PHOTO-PAINT)
EffectAdaptiveUnsharp (PHOTO-PAINT)
EffectAddNoise (PHOTO-PAINT)
EffectAdjustBlur (PHOTO-PAINT)
EffectAdjustNoise (PHOTO-PAINT)
EffectAdjustSharpness (PHOTO-PAINT)
EffectBandPass (PHOTO-PAINT)
EffectBitPlanes (PHOTO-PAINT)
EffectCanvas (PHOTO-PAINT)
EffectDiffuse (PHOTO-PAINT)
EffectDirectionalSharpen (PHOTO-PAINT)
EffectDirectionalSmooth (PHOTO-PAINT)
EffectDisplace (PHOTO-PAINT)
EffectDustScratch (PHOTO-PAINT)
EffectEdgeDetect (PHOTO-PAINT)
EffectEmboss (PHOTO-PAINT)
EffectFindEdges (PHOTO-PAINT)
EffectGaussianBlur (PHOTO-PAINT)
EffectGlass (PHOTO-PAINT)
EffectGlassBlock (PHOTO-PAINT)
EffectHalftone (PHOTO-PAINT)
EffectHighPass (PHOTO-PAINT)
EffectImpressionist (PHOTO-PAINT)
EffectJaggyDespeckle (PHOTO-PAINT)
EffectLensFlare (PHOTO-PAINT)

EffectLighting (PHOTO-PAINT)
EffectLowPass (PHOTO-PAINT)
EffectMapToObject (PHOTO-PAINT)
EffectMaximum (PHOTO-PAINT)
EffectMedian (PHOTO-PAINT)
EffectMeshPoint (PHOTO-PAINT)
EffectMeshWarp (PHOTO-PAINT)
EffectMinimum (PHOTO-PAINT)
EffectMotionBlur (PHOTO-PAINT)
EffectOffset (PHOTO-PAINT)
EffectPageCurl (PHOTO-PAINT)
EffectPerspective (PHOTO-PAINT)
EffectPinchPunch (PHOTO-PAINT)
EffectPixelate (PHOTO-PAINT)
EffectPlugin (PHOTO-PAINT)
EffectPsychedelic (PHOTO-PAINT)
EffectPuzzle (PHOTO-PAINT)
EffectRadialBlur (PHOTO-PAINT)
EffectRemoveMoire (PHOTO-PAINT)
EffectRemoveNoise (PHOTO-PAINT)
EffectRipple (PHOTO-PAINT)
EffectSharpen (PHOTO-PAINT)
EffectShear (PHOTO-PAINT)
EffectShearTable (PHOTO-PAINT)
EffectSmokedGlass (PHOTO-PAINT)
EffectSmooth (PHOTO-PAINT)
EffectSoften (PHOTO-PAINT)
EffectSolarize (PHOTO-PAINT)
EffectSwirl (PHOTO-PAINT)
EffectTheBoss (PHOTO-PAINT)
EffectTile (PHOTO-PAINT)
EffectTool (PHOTO-PAINT)
EffectTraceContour (PHOTO-PAINT)
EffectUnsharpMask (PHOTO-PAINT)
EffectUserDefined (PHOTO-PAINT)
EffectUserDefinedPoint (PHOTO-PAINT)
EffectVignette (PHOTO-PAINT)
EffectWetPaint (PHOTO-PAINT)
EffectWhirlpool (PHOTO-PAINT)
EffectWind (PHOTO-PAINT)
EffectZigZag (PHOTO-PAINT)
Ellipse (PHOTO-PAINT)
EllipseTool (PHOTO-PAINT)
EndAdjustEffect (PHOTO-PAINT)
EndColorEffect (PHOTO-PAINT)
EndColorMask (PHOTO-PAINT)
EndColorTable (PHOTO-PAINT)
EndConvertDuotone (PHOTO-PAINT)
EndConvertPaletted (PHOTO-PAINT)
EndDraw (PHOTO-PAINT)

EndEditFill (PHOTO-PAINT)
EndEffectLighting (PHOTO-PAINT)
EndEffectMeshWarp (PHOTO-PAINT)
EndEffectShear (PHOTO-PAINT)
EndEffectUserDefined (PHOTO-PAINT)
EndImageEqualize (PHOTO-PAINT)
EndImageSTBalance (PHOTO-PAINT)
EndImageToneCurve (PHOTO-PAINT)
EndMovieFrameRate (PHOTO-PAINT)
EndObjectColorTransparencyTool (PHOTO-PAINT)
EndObjectTagWWWURL (PHOTO-PAINT)
Eraser (PHOTO-PAINT)

F

FadeLastCommand (PHOTO-PAINT)
FileAcquireWithFile (PHOTO-PAINT)
FileClose (PHOTO-PAINT)
FileNew (PHOTO-PAINT)
FileOpen (PHOTO-PAINT)
FilePrint (PHOTO-PAINT)
FileRevert (PHOTO-PAINT)
FileSave (PHOTO-PAINT)
FileSelectPartialArea (PHOTO-PAINT)
Fill (PHOTO-PAINT)
FillBitmap (PHOTO-PAINT)
FillFountain (PHOTO-PAINT)
FillFountainApply (PHOTO-PAINT)
FillFountainColor (PHOTO-PAINT)
FillSolid (PHOTO-PAINT)
FillTexture (PHOTO-PAINT)
FillTextureSettings (PHOTO-PAINT)
FillTool (PHOTO-PAINT)
FilterGIF (PHOTO-PAINT)
FilterJPG (PHOTO-PAINT)
FilterOS2 (PHOTO-PAINT)
FilterPNG (PHOTO-PAINT)
FilterTGA (PHOTO-PAINT)
FilterWVL (PHOTO-PAINT)

G

GetChannelCount (PHOTO-PAINT)
GetChannelName (PHOTO-PAINT)
GetCurrentMovieFrame (PHOTO-PAINT)
GetDocumentCount (PHOTO-PAINT)
GetDocumentHeight (PHOTO-PAINT)
GetDocumentIsMovie (PHOTO-PAINT)
GetDocumentIsPartial (PHOTO-PAINT)
GetDocumentModified (PHOTO-PAINT)
GetDocumentName (PHOTO-PAINT)
GetDocumentType (PHOTO-PAINT)
GetDocumentWidth (PHOTO-PAINT)

GetDocumentXdpi (PHOTO-PAINT)
GetDocumentYdpi (PHOTO-PAINT)
GetFillColor (PHOTO-PAINT)
GetFrameCount (PHOTO-PAINT)
GetMaskPresent (PHOTO-PAINT)
GetMaskRectangle (PHOTO-PAINT)
GetObjectCount (PHOTO-PAINT)
GetObjectIsEditable (PHOTO-PAINT)
GetObjectIsSelected (PHOTO-PAINT)
GetObjectIsVisible (PHOTO-PAINT)
GetObjectMergeMode (PHOTO-PAINT)
GetObjectName (PHOTO-PAINT)
GetObjectOpacity (PHOTO-PAINT)
GetObjectProperties (PHOTO-PAINT)
GetObjectRectangle (PHOTO-PAINT)
GetPaintColor (PHOTO-PAINT)
GetPaintVersion (PHOTO-PAINT)
GetPaperColor (PHOTO-PAINT)
GetPartialDocumentHeight (PHOTO-PAINT)
GetPartialDocumentWidth (PHOTO-PAINT)
GetPhotoPaintDir (PHOTO-PAINT)
GetSelectedObjectsRectangle (PHOTO-PAINT)
Gradient (PHOTO-PAINT)
GradientPoint (PHOTO-PAINT)
GradientTool (PHOTO-PAINT)

I

ImageAutoEqualize (PHOTO-PAINT)
ImageBCI (PHOTO-PAINT)
ImageColorBalance (PHOTO-PAINT)
ImageColorCrop (PHOTO-PAINT)
ImageColorTable (PHOTO-PAINT)
ImageColorTone (PHOTO-PAINT)
ImageConvert (PHOTO-PAINT)
ImageConvertDuotone (PHOTO-PAINT)
ImageConvertPaletted (PHOTO-PAINT)
ImageConvertVideoNTSC (PHOTO-PAINT)
ImageCrop (PHOTO-PAINT)
ImageCropToMask (PHOTO-PAINT)
ImageDeInterlace (PHOTO-PAINT)
ImageDesaturate (PHOTO-PAINT)
ImageDeskew (PHOTO-PAINT)
ImageDeskewCrop (PHOTO-PAINT)
ImageDuplicate (PHOTO-PAINT)
ImageEqualize (PHOTO-PAINT)
ImageEqualizeChannel (PHOTO-PAINT)
ImageFlipHorizontal (PHOTO-PAINT)
ImageFlipVertical (PHOTO-PAINT)
ImageGamma (PHOTO-PAINT)
ImageHSL (PHOTO-PAINT)

ImageHSLChannel (PHOTO-PAINT)
ImageInvert (PHOTO-PAINT)
ImageLevelThreshold (PHOTO-PAINT)
ImagePapersize (PHOTO-PAINT)
ImagePosterize (PHOTO-PAINT)
ImageReplaceColors (PHOTO-PAINT)
ImageResample (PHOTO-PAINT)
ImageRotate (PHOTO-PAINT)
ImageSelectiveColor (PHOTO-PAINT)
ImageSelectiveColorChannel (PHOTO-PAINT)
ImageSetChannel (PHOTO-PAINT)
ImageSplit (PHOTO-PAINT)
ImageSprayerSettings (PHOTO-PAINT)
ImageSprayerTool (PHOTO-PAINT)
ImageSTBalance (PHOTO-PAINT)
ImageSTColor (PHOTO-PAINT)
ImageToneChannel (PHOTO-PAINT)
ImageToneCurve (PHOTO-PAINT)
ImageToneCurve (PHOTO-PAINT)
ImageTonePoint (PHOTO-PAINT)
ImageToneTable (PHOTO-PAINT)

L

LensCreateFromMask (PHOTO-PAINT)
LensEdit (PHOTO-PAINT)
LensNew (PHOTO-PAINT)
LineTool (PHOTO-PAINT)
LocalUndo (PHOTO-PAINT)

M

MaskAffineDistort (PHOTO-PAINT)
MaskAlign (PHOTO-PAINT)
MaskBorder (PHOTO-PAINT)
MaskBrush (PHOTO-PAINT)
MaskChannelAdd (PHOTO-PAINT)
MaskChannelDelete (PHOTO-PAINT)
MaskChannelLoad (PHOTO-PAINT)
MaskChannelName (PHOTO-PAINT)
MaskChannelSave (PHOTO-PAINT)
MaskChannelToMask (PHOTO-PAINT)
MaskCreate (PHOTO-PAINT)
MaskCreateFromPath (PHOTO-PAINT)
MaskDeFloat (PHOTO-PAINT)
MaskDistort (PHOTO-PAINT)
MaskEllipse (PHOTO-PAINT)
MaskExpand (PHOTO-PAINT)
MaskFeather (PHOTO-PAINT)
MaskFlipHorizontal (PHOTO-PAINT)
MaskFlipVertical (PHOTO-PAINT)
MaskFloaterMoveTo (PHOTO-PAINT)
MaskFreehand (PHOTO-PAINT)

MaskGrow (PHOTO-PAINT)
MaskInvert (PHOTO-PAINT)
MaskLasso (PHOTO-PAINT)
MaskLoad (PHOTO-PAINT)
MaskMagicWand (PHOTO-PAINT)
MaskPaint (PHOTO-PAINT)
MaskRectangle (PHOTO-PAINT)
MaskReduce (PHOTO-PAINT)
MaskRemove (PHOTO-PAINT)
MaskRemoveHoles (PHOTO-PAINT)
MaskRotate (PHOTO-PAINT)
MaskSave (PHOTO-PAINT)
MaskScissors (PHOTO-PAINT)
MaskSelectAll (PHOTO-PAINT)
MaskSimilar (PHOTO-PAINT)
MaskSkew (PHOTO-PAINT)
MaskSmooth (PHOTO-PAINT)
MaskStretch (PHOTO-PAINT)
MaskStroke (PHOTO-PAINT)
MaskThreshold (PHOTO-PAINT)
MaskToMaskChannel (PHOTO-PAINT)
MaskTranslate (PHOTO-PAINT)
MovieBackOne (PHOTO-PAINT)
MovieCreate (PHOTO-PAINT)
MovieDeleteFrame (PHOTO-PAINT)
MovieForward (PHOTO-PAINT)
MovieForwardOne (PHOTO-PAINT)
MovieFrameDelay (PHOTO-PAINT)
MovieFrameRate (PHOTO-PAINT)
MovieGotoFrame (PHOTO-PAINT)
MovieInsertFile (PHOTO-PAINT)
MovieInsertFrame (PHOTO-PAINT)
MovieMoveFrame (PHOTO-PAINT)
MovieRewind (PHOTO-PAINT)
MovieSelectPartial (PHOTO-PAINT)

N

NibSettings (PHOTO-PAINT)

O

ObjectAddClipMask (PHOTO-PAINT)
ObjectAddClipMaskFromMask (PHOTO-PAINT)
ObjectAddClipMaskFromTransparency (PHOTO-PAINT)
ObjectAffineDistort (PHOTO-PAINT)
ObjectAlign (PHOTO-PAINT)
ObjectClip (PHOTO-PAINT)
ObjectClipToParent (PHOTO-PAINT)
ObjectColorTransparencyTool (PHOTO-PAINT)
ObjectCombine (PHOTO-PAINT)
ObjectCreate (PHOTO-PAINT)
ObjectCreateFromBackground (PHOTO-PAINT)

ObjectDefringe (PHOTO-PAINT)
ObjectDelete (PHOTO-PAINT)
ObjectDistort (PHOTO-PAINT)
ObjectDropShadow (PHOTO-PAINT)
ObjectDuplicate (PHOTO-PAINT)
ObjectEdit (PHOTO-PAINT)
ObjectEditAll (PHOTO-PAINT)
ObjectEditNone (PHOTO-PAINT)
ObjectEditSelected (PHOTO-PAINT)
ObjectEditTransparency (PHOTO-PAINT)
ObjectFeather (PHOTO-PAINT)
ObjectFlipHorizontal (PHOTO-PAINT)
ObjectFlipVertical (PHOTO-PAINT)
ObjectGroup (PHOTO-PAINT)
ObjectMerge (PHOTO-PAINT)
ObjectMergeMode (PHOTO-PAINT)
ObjectName (PHOTO-PAINT)
ObjectNew (PHOTO-PAINT)
ObjectOpacity (PHOTO-PAINT)
ObjectOrder (PHOTO-PAINT)
ObjectOrderChange (PHOTO-PAINT)
ObjectProperties (PHOTO-PAINT)
ObjectRemoveClipMask (PHOTO-PAINT)
ObjectRemoveMatte (PHOTO-PAINT)
ObjectRotate (PHOTO-PAINT)
ObjectSelect (PHOTO-PAINT)
ObjectSelectAll (PHOTO-PAINT)
ObjectSelectNone (PHOTO-PAINT)
ObjectSkew (PHOTO-PAINT)
ObjectStretch (PHOTO-PAINT)
ObjectTagWWWURL (PHOTO-PAINT)
ObjectThreshold (PHOTO-PAINT)
ObjectToggleClipMask (PHOTO-PAINT)
ObjectToggleLinkClipMask (PHOTO-PAINT)
ObjectTranslate (PHOTO-PAINT)
ObjectTransparencyTool (PHOTO-PAINT)
ObjectUngroup (PHOTO-PAINT)
ObjectURLInfo (PHOTO-PAINT)
ObjectVisible (PHOTO-PAINT)
OverprintColor (PHOTO-PAINT)

P

PaletteColor (PHOTO-PAINT)
PathCreate (PHOTO-PAINT)
PathCreateFromMask (PHOTO-PAINT)
PathDelete (PHOTO-PAINT)
PathEnd (PHOTO-PAINT)
PathImportVector (PHOTO-PAINT)
PathLoad (PHOTO-PAINT)
PathNew (PHOTO-PAINT)

PathNode (PHOTO-PAINT)
PathSave (PHOTO-PAINT)
PathStroke (PHOTO-PAINT)
PolygonTool (PHOTO-PAINT)
PressureSettings (PHOTO-PAINT)

R

RandomSeed (PHOTO-PAINT)
Rectangle (PHOTO-PAINT)
RectangleTool (PHOTO-PAINT)
RepeatSettings (PHOTO-PAINT)

S

SelectionMoveTo (PHOTO-PAINT)
SetDocumentInfo (PHOTO-PAINT)
SetDocVisible (PHOTO-PAINT)
SetPaintColor (PHOTO-PAINT)
SetPaperColor (PHOTO-PAINT)
SetVisible (PHOTO-PAINT)
StartCloneDraw (PHOTO-PAINT)
StartDraw (PHOTO-PAINT)
SymmetrySettings (PHOTO-PAINT)

T

TextSettings (PHOTO-PAINT)
TextTool (PHOTO-PAINT)
ToleranceSettings (PHOTO-PAINT)
TransparencyBrushTool (PHOTO-PAINT)

File commands (PHOTO-PAINT)

BindToActiveDocument (PHOTO-PAINT)

.BindToActiveDocument

This command creates a link between the executing script and the active PHOTO-PAINT document. This command is used when the user manually changes the active document during script execution.

Example

```
.FileNew 300, 300, 1, 100, 100, 0, 0, 0, -1, -1, -1, -1, 255, 0, 0, 0
.FileNew 300, 300, 1, 100, 100, 0, 0, 0, -1, -1, -1, -1, 0, 255, 255, 0
.FileNew 300, 300, 1, 100, 100, 0, 0, 0, -1, -1, -1, -1, 255, 0, 255, 0
MESSAGE "Select the image where you want to draw an ellipse"
.BindToActiveDocument
```

This example creates three new documents, prompts the user to select one of the three, then creates a link to the selected document.

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

FileAcquireWithFile (PHOTO-PAINT)

.FileAcquireWithFile .FileName = *string*, .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .LoadType = *long*

This command lets you open a scanned image from a file and apply color correction to it. The color correction applied depends on the scanner originally used to scan the image.

Syntax	Definition
.FileName	Specifies the name of the file.
.Left	Specifies the X-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.
.LoadType	Indicates the image load type of the file: 0 = All (coordinates are not used) 1 = Partial 2 = Resample 3 = Crop

Example

```
.FileAcquireFromFile "TEST.1.CPT", 0, 0, 0, 0, 0
```

This example opens the scanned image named "TEST.1.CPT".

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

FileClose (PHOTO-PAINT)

.FileClose

This command closes the active image.

Example

```
.FileClose
```

This example closes the active image.

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

FileNew (PHOTO-PAINT)

.FileNew .Width = *long*, .Height = *long*, .Type = *long*, .HRes = *long*, .VRes = *long*, .PartialFile = *boolean*, .MovieFile = *boolean*, .NumberFrames = *long*, .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long,* .Backgroundless = *boolean*

This command defines the characteristics of a new image.

Syntax	Definition
.Width	Specifies the width of the new image in pixels.
.Height	Specifies the height of the new image in pixels.
.Type	Indicates the image type: 1 = RGB 2 = 256 Grayscale 3 = Black and white 4 = Paletted 5 = CMYK 7 = LAB
.HRes	Specifies the horizontal resolution of the image in dots per inch (dpi).
.VRes	Specifies the vertical resolution of the image in dots per inch (dpi).
.PartialFile	Set to TRUE (-1) to load a partial area of an image. This lets you work on separate areas of an image without opening the entire file. Otherwise set to FALSE (0) Note: If this option is selected, the new file cannot be a movie file.
.MovieFile	Set to TRUE (-1) to open the new image as a movie file. Otherwise set to FALSE (0) Note: If the new image is opened as a movie file, the .PartialFile option is disabled.
.NumberFrames	If the movie file option is enabled (i.e. set to TRUE) enter the number of frames you would like the movie file to contain.
.Left	Specifies the X-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin. (Applies only if the new image is a partial file.).
.Top	Specifies the Y-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin. (Applies only if the new image is a partial file.).
.Right	Specifies the X-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin. (Applies only if the new image is a partial file.).
.Bottom	Specifies the Y-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin. (Applies only if the new image is a partial file.).
.Color1	Specifies the first color component for .Type. For example, Red is the first color component for RGB. Use RGB for 16 Color and 256 Color images.
.Color2	Specifies the second color component for .Type. For example, Green is the second color component for RGB. If this parameter is not available in the Color Model specified, set it to 0. Use RGB for 16 Color and 256 Color images.
.Color3	Specifies the third color component for .Type. For example, Blue is the third color component for RGB. If this parameter is not available in the Color Model specified, set it to 0. Use RGB for 16 Color and 256 Color images.
.Color4	Specifies the fourth color component for .Type. For example, Black is the fourth color component for CMYK. If this parameter is not available in the Color Model specified, set it to 0. Use RGB for 16 Color and 256 Color images.
.Backgroundless	Set to TRUE (-1) to create an image without a background. Set to FALSE (0) to create an image with a background.

Example

```
.FileNew 500, 500, 3, 100, 100, FALSE, FALSE, 1, 0, 0, 0, 0, 0, 0, 0, 0, FALSE
```

This example creates a new file that is 500 pixels by 500 pixels in size and has a black background.

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

FileOpen (PHOTO-PAINT)

.FileOpen .FileName = *string*, .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .LoadType = *long,* .StartFrame = *long,* .EndFrame = *long*

This command opens an existing file and loads it into the main Image window.

Syntax	Definition
.FileName	Specifies the path and name of the file to be opened.
.Left	Specifies the X-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.
.LoadType	Indicates the image load type of the file: 0 = All (coordinates are not used) 1 = Partial 2 = Resample 4 = Crop
.StartFrame	Specifies the starting index of the frame range to load from a movie file.
.EndFrame	Specifies the ending index of the frame range to load from a movie file.

Example

```
.FileOpen "TEST.1.CPT", 0, 0, 0, 0, 0, 1, 1
```

This example opens the Corel PHOTO-PAINT file named TEST.1.CPT.

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

FilePrint (PHOTO-PAINT)

.FilePrint

This command sends the current document to the printer.

Example

```
.FilePrint
```

This example sends the active document to the printer.

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

FileRevert (PHOTO-PAINT)

.FileRevert

This command undoes changes made to the image since it was last saved. Use this command when the Undo command is unavailable or when you want to undo more than one action.

Example

```
.FileRevert
```

This example undoes changes made to the image since it was last saved.

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

FileSave (PHOTO-PAINT)

.FileSave .FileName = *string*, .FilterID = *long*, .Compression = *long*

This command saves the current image.

Syntax	Definition
.FileName	Specifies the path and name of the file to be saved.
.FilterID	Specifies the type of file filter 769 = Windows Bitmap (BMP) 770 = Paintbrush (PCX) 771 = Targa Bitmap (TGA) 772 = TIFF Bitmap (TIF) 773 = CompuServe Bitmap (GIF) 774 = JPEG Bitmap (JPG) 775 = Kodak Photo CD Image (PCD) 776 = Scitex CT Bitmap (SCT) 777 = Wavelet Compressed Bitmap (WVL) 787 = GEM Paint File (IMG) 784 = Windows 3.x/NT Cursor Resource (CUR) 788 = Adobe Photoshop (PSD) 785 = Windows 3.x/NT Icon Resource (ICO) 786 = Windows 3.x/NT Bitmap Resource (EXE) 789 = Picture Publisher 4 (PB4) 790 = MACPaint Bitmap (MAC) 792 = OS/2 Bitmap (BMP) 793 = Wavelet Compressed Bitmap (WI) 800 = CALS Compressed Bitmap (CAL) 802 = Portable Network Graphic (PNG) 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) 1290 = PostScript Interpreted (PS) 1291 = OS/2 PM Metafile (MET) 1294 = Windows Metafile (WMF) 1295 = Corel Metafile (CMF) 1296 = AutoCad (DXF) 1536 = Video for Windows (AVI) 1539 = CorelMOVE (CMV) 1540 = CorelSHOW (SHW) 1541 = CorelCHART (CCH) 1543 = AutoDesk FLIC (FLI) 1548 = MicroSoft PowerPoint (PPT) 1549 = Lotus Freelance (PRE) 1551 = MPEG Animation (MPG) 1792 = Corel PHOTO-PAINT Image (CTP) 1793 = Corel CMX 6.0 1794 = Corel CMX 5.0 1795 = CorelDRAW (CDR)
.Compression	Specifies the type of file compression to apply: 0 = None 1 = LZW 2 = Packbits 3 = JPEG

Example

```
.FileSave "TEST1.BMP", 769, 0
```

This example saves the file named TEST1.BMP in Windows Bitmap format with no compression applied.

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

FileSelectPartialArea (PHOTO-PAINT)

.FileSelectPartialArea .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*

This command defines a specific area of an image to open.

Syntax	Definition
.Left	Specifies the X-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.

Note

The image must have been created or opened as a partial file to use this command. The specified rectangle must be within the image.

Example

```
.FileSelectPartialArea 120, 168, 335, 239
```

This example opens the specified area of the image.

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

FilterGIF (PHOTO-PAINT)

.FilterGIF .InvertMask = *boolean*, .Interlace = *boolean*, .Transparent = *long*, .Index = *long*, .Delay = *long*, .Red = *long*, .Green = *long*, .Blue=*long*

This command sets the CompuServe Bitmap (GIF) filter information for the FileSave command.

Syntax	Description
.InvertMask	Set to TRUE (-1) to invert the transparency mask.
.Interlace	Set to TRUE (-1) for an interlaced GIF.
.Transparent	Specifies the transparency method: 0 = None 1 = Sets the color specified by the .Index parameter to be transparent 2 = Uses a transparency mask
.Index	Specifies the color in the current palette to make transparent. Valid values range from 0 to 255.
.Delay	Specifies the delay, in hundredths of a second, between successive frames of an animated GIF.
.Red	Specifies the red channel of the area outside the transparency mask. Valid values range from 0 to 255.
.Green	Specifies the green channel of the area outside the transparency mask. Valid values range from 0 to 255.
.Blue	Specifies the blue channel of the area outside the transparency mask. Valid values range from 0 to 255.

Example

```
.FilterGIF 0, 1, 1, 200, 0, 0, 0, 0
.FileSave "test.gif", 773, 0
```

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

FilterJPG (PHOTO-PAINT)

.FilterJPG .Compression= *long*, .Progressive = *long,* .Smoothing = *long,* .SubFormat = *long,* .Optimized = *long*

This command sets the JPEG filter information for the FileSave command.

Syntax	Description
.Compression	Specifies the compression factor of the bitmap. Valid values range from 2 (large, high-quality file) to 255 (small, low-quality file).
.Progressive	Set to TRUE (-1) to create a progressive jpeg.
.Smoothing	Specifies a smoothing value for the image.
.SubFormat	Specifies an encoding method sub format. You can select Standard or Optional compression formats.
.Optimized	Specifies an optimal encoding method.

Example

```
.FilterJPG 51, TRUE, 46, 0 FALSE
```

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

FilterOS2 (PHOTO-PAINT)

.FilterOS2 .Format = *long*

This command sets the OS/2 Bitmap (BMP) filter information for the FileSave command.

Syntax	Description
.Format	Sets the export format: 0 = Standard format (OS/2 version 1.3) 1 = Enhanced format (OS/2 version 2.0 or later)

Example

```
.FilterOS2 1
.FileSave "test.bmp", 792, 0
```

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

FilterPNG (PHOTO-PAINT)

.FilterPNG .Interlace = *long*

This command sets the Portable Network Graphic (PNG) filter information for the FileSave command.

Syntax	Description
.Interlace	Set to TRUE (-1) to create an interlaced PNG.

Example

```
.FilterPNG 1
.FileSave "Test.png", 802, 0
```

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

FilterTGA (PHOTO-PAINT)

.FilterTGA .Format = *long*

This command sets the Targa Bitmap (TGA) filter information for the FileSave command.

Syntax	Description
.Format	Specifies the bitmap format: 0 = Normal 1 = Enhanced

Example

```
.FilterTGA 1
.FileSave "test.tga", 771, 0
```

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

FilterWVL (PHOTO-PAINT)

.FilterWVL .Compression = *long,* .Speed = *long,* .Path = *long,* .Contrast = *long,* .Edge = *long*

This command sets the Wavelet Compressed Bitmap (WI) filter information for the FileSave command.

Syntax	Description
.Compression	Specifies the compression factor of the bitmap. Valid values range from 1 (large, high-quality file) to 100 (small, low-quality file).
.Speed	Specifies the speed of the encoding method 0=Normal 1=Fast
.Path	Specifies the encoding path 0=One 1=Two 2=Three
.Contrast	Sets the level of contrast. Valid values range from 0 to 8.
.Edge	Sets the edging of the image. Valid values range from 0 to 8.

Example

```
.FilterWVL 41, 0, 0, 0, 0
```

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

SetDocumentInfo (PHOTO-PAINT)

.SetDocumentInfo .Width = *long*, .Height = *long*

This command defines an image size to use as a reference for performing the operations in the script. If you define the image size using SetDocumentInfo, you can run your script on images of any size and the script will scale its operations to match.

Syntax	Description
.Width	Specifies the reference image width.
.Height	Specifies the reference image height.

Example

```
.SetDocumentInfo 640, 480
.MaskRectangle 20, 20, 620, 460, 0, 0
```

This example can create a MaskRectangle on any image size with the same relative size as the original.

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

SetDocVisible (PHOTO-PAINT)

.SetDocVisible .Show = *boolean*

This command controls whether the document is visible in PHOTO-PAINT.

Syntax	Definition
.Show	Set to TRUE (-1) to display the active document. Set to FALSE (0) to hide the active document.

Example

```
.SetDocVisible -1
```

This example displays the active Corel PHOTO-PAINT document.

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

SetVisible (PHOTO-PAINT)

.SetVisible .Show = *boolean*

This command controls whether the PHOTO-PAINT application is visible or hidden.

Syntax	Definition
.Show	Set to TRUE (-1) to show the Corel PHOTO-PAINT application. Set to FALSE (0) to hide the application.

Example

```
.SetVisible -1
```

This example shows the Corel PHOTO-PAINT application.

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

Edit commands (PHOTO-PAINT)

EditCheckpoint (PHOTO-PAINT)

.EditCheckpoint

This command saves a copy of the image to a temporary file. Additions or edits to the image that are performed after the checkpoint can be removed by using the .EditRestoreCheckpoint command. This will reverse all previous changes made to the image since this command was selected.

Note

Issuing this command clears the undo list.

Example

```
.EditCheckpoint
```

This example saves the image at its current state.

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

EditClear (PHOTO-PAINT)

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

If an object is selected , this command deletes the object. If a mask is present, this command clears the masked area to the current paper color. If no mask is present, this command clears the entire image to the current paper color. If an object is selected and a mask is present, the object is deleted and the mask is ignored.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EditClear 5, 255, 255, 0, 0
```

This example clears the current document and removes any editing that has been performed.

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

EditClearClipboard (PHOTO-PAINT)

.EditClearClipboard

This command clears the Clipboard of all information, which conserves memory and reduces execution time.

Example

```
.EditClearClipboard
```

This example removes all items previously placed on the Clipboard.

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

EditCopy (PHOTO-PAINT)

.EditCopy

This command copies an object or masked area from the image and places it on the Clipboard. If no mask is present or no object is selected, the entire image is copied to the Clipboard.

Example

```
.EditCopy
```

This example copies the selected object to the Clipboard.

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

EditCopyToFile (PHOTO-PAINT)

.EditCopyToFile .FileName = *string*, .FilterID = *long*, .Compression = *long*

This command saves a copy of an object or masked area to an existing or new file. If no mask is present or no object is selected, the entire image is copied to the file.

Syntax	Definition
.FileName	The path and file name of the destination file.
.FilterID	Specifies the type of file filter 769 = Windows Bitmap (BMP) 770 = Paintbrush (PCX) 771 = Targa Bitmap (TGA) 772 = TIFF Bitmap (TIF) 773 = CompuServe Bitmap (GIF) 774 = JPEG Bitmap (JPG) 775 = Kodak Photo CD Image (PCD) 776 = Scitex CT Bitmap (SCT) 777 = Wavelet Compressed Bitmap (WVL) 787 = GEM Paint File (IMG) 784 = Windows 3.x/NT Cursor Resource (CUR) 788 = Adobe Photoshop (PSD) 785 = Windows 3.x/NT Icon Resource (ICO) 786 = Windows 3.x/NT Bitmap Resource (EXE) 790 = MACPaint Bitmap (MAC) 789 = Picture Publisher 4 (PB4) 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) 1290 = PostScript Interpreted (PS) 1291 = OS/2 PM Metafile (MET) 1294 = Windows Metafile (WMF) 1295 = Corel Metafile (CMF) 1296 = AutoCad (DXF) 1536 = Video for Windows (AVI) 1539 = CorelMOVE (CMV) 1540 = CorelSHOW (SHW) 1541 = CorelCHART (CCH) 1543 = AutoDesk FLIC (FLI) 1548 = MicroSoft PowerPoint (PPT) 1549 = Lotus Freelance (PRE) 1551 = MPEG Animation (MPG) 1792 = Corel PHOTO-PAINT Image (CTP) 1794 = Corel CMX 5.0 1793 = Corel CMX 6.0 1795 = CorelDRAW (CDR)
.Compression	Specifies the type of file compression to apply: 0 = None 1 = LZW 2 = Packbits 3 = JPEG

Example

```
.EditCopyToFile "TEST1.CPT", 1792, 0
```

This example copies the selected object to the PHOTO-PAINT file named TEST1.CPT.

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

EditCopyVisible (PHOTO-PAINT)

.EditCopyVisible

This command allows you to copy all visible objects, masked selections, or floating selections in the active image to the Clipboard.

Example

```
.EditCopyVisible
```

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

EditCut (PHOTO-PAINT)

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

This command cuts an object or masked area from the image and places it on the Clipboard. If no mask is present or no object is selected, the entire image is cut and placed on the Clipboard.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Note

This command is only available when a mask or object is active and selected in the main Image window.

Example

```
.EditCut 5, 255, 255, 0, 0
```

This example removes the selected object from the document and places it on the Clipboard.

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

EditCutIntoSelection (PHOTO-PAINT)

.EditCutIntoSelection

This command cuts a selection from inside another selection.

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

EditCutMask (PHOTO-PAINT)

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

This command cuts the area inside a mask and copies it to the Clipboard. The space behind the cut section reverts to the background color.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EditCutMask 5, 255, 255, 0, 0
```

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

EditCutSelection (PHOTO-PAINT)

.EditCutSelection

This command cuts a floating selection onto the Clipboard. Unlike the EditCutMask command, the area behind the selection is unchanged.

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

EditFill (PHOTO-PAINT)

.EditFill .MergeMode = *long*, .StartTransparency = *long*, .EndTransparency = *long*, .GradientType = *long*, .Handles = *long*, .X1 = *long*, .Y1 = *long*, .X2 = *long*, .Y2 = *long*, .X3 = *long*, .Y3 = *long*

This command applies a fill to the current image using settings specified in a block of commands. EditFill command blocks must end with an EndEditFill command.

Syntax	Description
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.StartTransparency	Specifies the starting transparency value for a gradient fill.
.EndTransparency	Specifies the ending transparency value for a gradient fill.
.GradientType	Specifies the gradient type: 0 = None 1 = Flat 2 = Linear 3 = Elliptical 4 = Radial 5 = Rectangular 6 = Square 7 = Conical
.Handles	Specifies the number of handles used to define the gradient. Valid values range from 1 to 3, depending on the type of gradient.
.X1	Specifies the horizontal coordinate of the first gradient handle.
.Y1	Specifies the vertical coordinate of the first gradient handle.
.X2	Specifies the horizontal coordinate of the second gradient handle.
.Y2	Specifies the vertical coordinate of the second gradient handle.
.X3	Specifies the horizontal coordinate of the third gradient handle.
.Y3	Specifies the vertical coordinate of the third gradient handle.

Example

```
.EditFill 0, 0, 100, 3, 3, 294, 240, 79, 164, 114, 61
    .FillFountainColor 5, 2, 74, 123, 0, 0, 0
    .FillFountainColor 5, 255, 255, 255, 0, 50, 1
    .FillFountainColor 5, 63, 125, 122, 0, 100, 2
    .FillFountainApply 3, 3, 256, 135, 0, 0, 0, 50
    .EndEditFill
```

This example creates an Elliptical Fountain Fill with a starting transparency of 0 and an ending transparency of 100.

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

EditPasteDocument (PHOTO-PAINT)

.EditPasteDocument

This command creates a new document and inserts the contents of the Clipboard as an object. The size of the new document is the same as the pasted object. The background paper color is the current paper color.

Example

```
.EditPasteDocument
```

This example inserts a copy of the object most recently placed on the Clipboard into a new document.

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

EditPasteFromFile (PHOTO-PAINT)

.EditPasteFromFile .FileName = *string*, .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .LoadType = *long*, .ptX = *long*, .ptY = *long*

This command lets you select an image from a file to paste into the active image.

Syntax	Definition
.FileName	Specifies the path and name of the source file.
.Left	Specifies the X-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.
.LoadType	Specifies the image loading method: 0 = All 2 = Resample 4 = Crop
.ptX	Specifies the X-coordinate in the active image to place the center of the pasted image.
.ptY	Specifies the Y-coordinate in the active image to place the center of the pasted image.

Example

```
.EditPasteFromFile "CLAMSHEL.BMP", 0, 0, 0, 0, 0, 180, 255
```

This example pastes the bitmap file named CLAMSHEL.BMP into the open image.

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

EditPasteIntoSelection (PHOTO-PAINT)

.EditPasteIntoSelection

This command pastes the Clipboard contents into the current selection.

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

EditPasteObject (PHOTO-PAINT)

.EditPasteObject .ptX = *long*, .ptY = *long*, .FileName = *string*

This command pastes the current Clipboard contents into the active image as a new object.

Syntax	Description
.ptX	Specifies the X-coordinate in the active image to place the center of the pasted image.
.ptY	Specifies the Y-coordinate in the active image to place the center of the pasted image.
.FileName	If filename is "" (Null) then the image is pasted from the Clipboard, instead of from the "FileName".

Example

```
.EditPasteObject 160, 120, ""
```

This example pastes a copy of the object most recently placed in the Clipboard into the current document.

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

EditPasteSelection (PHOTO-PAINT)

.EditPasteSelection .ptX = *long*, .ptY = *long*, .FileName = *string*

This command pastes the contents of the Clipboard into the active image as a selection.

Syntax	Description
.ptX	Specifies the X-coordinate in the active image to place the center of the pasted image.
.ptY	Specifies the Y-coordinate in the active image to place the center of the pasted image.
.FileName	If filename is "" (Null) then the image is pasted from the Clipboard, instead of from the "FileName".

Example

```
.EditPasteSelection 100, 200, ""
```

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

EditRestoreCheckpoint (PHOTO-PAINT)

.EditRestoreCheckpoint

This command returns an image to the state it was at when the .EditCheckpoint command was last used.

Example

```
.EditRestoreCheckpoint
```

This example returns the active image to the state it was at when the .EditCheckpoint command was last used.

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

EditSingleObject (PHOTO-PAINT)

.EditSingleObject .Mode = *long*

This command switches between the three editing modes.

Syntax	Description
.Mode	Specifies the editing mode: 0 = Multiple object 1 = Single object 3 = Layer object

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

EndEditFill (PHOTO-PAINT)

.EndEditFill

This command ends an EditFill command block.

Example

```
.EditFill 0, 0, 100, 3, 3, 294, 240, 79, 164, 114, 61
    .FillFountainColor 5, 2, 74, 123, 0, 0, 0
    .FillFountainColor 5, 255, 255, 255, 0, 50, 1
    .FillFountainColor 5, 63, 125, 122, 0, 100, 2
    .FillFountainApply 3, 3, 256, 135, 0, 0, 0, 50
    .EndEditFill
```

This example creates an Elliptical Fountain Fill with a starting transparency of 0 and an ending transparency of 100.

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

FadeLastCommand (PHOTO-PAINT)

.FadeLastCommand .Percent = *long,* .MergeMode = long

This command reduces the effect of the most recent operation gradually.

Syntax	Description
.Percent	Specifies the amount by which the operation is faded. Values range from 0 to 100. If you set the percent to zero, the last command is not faded. If you set the percent to 100, the last command is made transparent.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black

Example

```
.FadeLastCommand 77, 0
```

This example fades the last recorded command by 77% in Normal mode.

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

LocalUndo (PHOTO-PAINT)

.LocalUndo .Width = *long*, .Flatten = *long*, .Rotate = *long*, .NibShape = *long*, .Transparency = *long*, .SoftEdge = *long*

This command reverts an area described by a series of Draw commands to the state it was in before the last operation. A LocalUndo command block must contain a series of StartDraw and ContinueDraw commands and end with an EndDraw command.

Syntax	Description
.Width	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Flatten	Specifies the flatness of the nib. Valid values range from 0 to 100%.
.Rotate	Specifies the rotation of the nib. Valid values range from 0 to 360 degrees.
.NibShape	Specifies the shape of the nib: 0 = Round 1 = Square
.Transparency	Specifies the transparency of the nib. Valid values range from 0 to 100%.
.SoftEdge	Specifies the transparency of the nib's edges. As you increase the value of this setting, the soft edge expands to eventually reach the center of the stroke. Low values affect only the rim of the stroke. Valid values range from 0 to 100%.

Example

```
.LocalUndo 20, 100, 0, 0, 0, 0
    .StartDraw 40320, 8832, 0, 0
    .ContinueDraw 39040, 8832, 0, 0
    ...
    .ContinueDraw 59678, 18166, 0, 0
    .EndDraw
```

This example replaces the portion of your image defined by the Draw commands with the defined paper color.

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

SelectionMoveTo (PHOTO-PAINT)

.SelectionMoveTo .Left = *long*, .Bottom = *long*

This command moves the current selection to the given coordinates.

Syntax	Description
.Left	Specifies the horizontal displacement of the bottom-left corner of the selection, in pixels. The displacement is relative to the original position of the selection, rather than to the origin.
.Bottom	Specifies the vertical displacement of the bottom-left corner of the selection, in pixels. The displacement is relative to the original position of the selection, rather than to the origin.

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

Get commands (PHOTO-PAINT)

GetChannelCount (PHOTO-PAINT)

ReturnValue& = .GetChannelCount()

This function returns a value indicating the number of mask channels saved.

Syntax	Definition
.ReturnValue&	Indicates the number of mask channels saved.

Example

```
.MaskChannelAdd "Mask 1"
.MaskChannelAdd "Mask 2"
cnt& = .GetChannelCount()
MESSAGE cnt&
```

This example displays the number of mask channels. The variable cnt is assigned a value of 2.

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

GetChannelName (PHOTO-PAINT)

ReturnValue& = .GetChannelName(long ChannelID)

This function returns a string containing the name of the specified channel.

Syntax	Definition
.ReturnValue&	Returns a string containing the name of the specified channel.
.ChannelID	Specifies the channel. Channels for the active document are listed in the Channels Roll-Up. The channels are numbered and identified according to their position in the list — the first listed channel is identified as 0, the second listed channel is identified as 1, and so on.

Example

```
.MaskChannelName 1, "Mask 5"
cname$ = .GetChannelName(1)
```

This example sets and then returns the name of the second channel in the channel list.

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

GetCurrentMovieFrame (PHOTO-PAINT)

ReturnValue& = .GetCurrentMovieFrame()

This function returns the number of frames in the current movie.

Syntax	Description
ReturnValue&	Returns a value indicating the number of frames in the current movie.

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

GetDocumentCount (PHOTO-PAINT)

ReturnValue& = .GetDocumentCount()

This function returns the number of open documents

Syntax	Description
ReturnValue&	Returns a value indicating the number of open documents.

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

GetDocumentHeight (PHOTO-PAINT)

ReturnValue& = .GetDocumentHeight()

This function returns the height of the active image in pixels.

Syntax	Definition
.ReturnValue&	Indicates the height of the active image.

Example

```
h& = .GetDocumentHeight()
MESSAGE h&
```

This example returns the height of the active image.

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

GetDocumentModified (PHOTO-PAINT)

ReturnValue& = .GetDocumentModified()

This command specifies whether the document has been modified since it was loaded.

Syntax	Description
ReturnValue&	Returns a boolean value which determines whether the document has been modified since it was loaded. TRUE (-1) indicates that the document has been modified. FALSE (0) indicates that the document has not been modified since it was loaded.

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

GetDocumentIsMovie (PHOTO-PAINT)

ReturnValue = .GetDocumentIsMovie()

This function returns a value indicating whether the active document is a movie file
TRUE (-1) = Document is a movie file
FALSE (0) = Document is not a movie file.

Example

```
status& = .GetDocumentIsMovie()
MESSAGE status&
```

This example determines whether the document is a movie file.

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

GetDocumentIsPartial (PHOTO-PAINT)

ReturnValue = .GetDocumentIsPartial()

This function returns a value indicating whether the active document is a partial file
TRUE (-1) = Document is a partial file
FALSE (0) = Document is not a partial file.

Example

```
status& = .GetDocumentIsPartial()
MESSAGE status&
```

This example determines whether the document is a partial file.

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

GetDocumentName (PHOTO-PAINT)

ReturnValue& = .GetDocumentName()

This function returns a string containing the name and file extension of the active image.

Syntax	Definition
.ReturnValue&	Returns a string containing the name and extension of the active image.

Example

```
name$ = .GetDocumentName()
MESSAGE name$
```

This example determines the name of the active image.

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

GetDocumentType (PHOTO-PAINT)

ReturnValue& = .GetDocumentType()

This function returns a value indicating the PHOTO-PAINT image type of the active document:

Syntax	Definition
.ReturnValue&	Indicates the image type: 1 = RGB 2 = 256 Grayscale 3 = Black and white 4 = Paletted 5 = CMYK 6 = Duotone 7 = LAB

Example

```
docType& = .GetDocumentType()
```

This example determines the paint image type of the active document.

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

GetDocumentWidth (PHOTO-PAINT)

ReturnValue& = .GetDocumentWidth()

This function returns the width of the active image in pixels.

Syntax	Definition
.Returnvalue&	Indicates the width of the active image.

Example

```
w& = .GetDocumentWidth()
MESSAGE status&
```

This example returns the width of the active image.

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

GetDocumentXdpi (PHOTO-PAINT)

ReturnValue& = .GetDocumentXdpi()

This function returns the horizontal resolution of the active image in dots per inch (dpi).

Syntax	Definition
.ReturnValue&	Indicates the horizontal resolution of the active image.

Example

```
xdpi& = .GetDocumentXdpi()
MESSAGE xdpi&
```

This example returns the horizontal resolution of the active image.

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

GetDocumentYdpi (PHOTO-PAINT)

ReturnValue& = .GetDocumentYdpi()

This function returns the vertical resolution of the active image in dots per inch (dpi).

Syntax	Definition
.returnvalue&	Indicates the vertical resolution of the active image.

Example

```
ydpi& = .GetDocumentYdpi()
MESSAGE ydpi&
```

This example returns the vertical resolution of the active image.

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

GetFillColor (PHOTO-PAINT)

ReturnValue = .GetFillColor(.ColorModel = *long**, .Color1 = *long**, .Color2 = *long**, .Color3 = *long**, long* Color4)

This function returns the fill color of the selected object. The number and type of .Color variables returned depends on the current Color Model. The function will return FALSE values (0) if the selected object is filled with anything other than a solid color.

Syntax	Description
.ColorModel	Returns the currently active Color Model: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Returns the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Returns the second color component for .ColorModel. For example, Green is the second color component for RGB.
.Color3	Returns the third color component for .ColorModel. For example, Blue is the third color component for RGB.
.Color4	Returns the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK.

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

GetFrameCount (PHOTO-PAINT)

ReturnValue& = .GetFrameCount()

This function returns the number of frames in the active image.

Example

```
cnt& = .GetFrameCount()
MESSAGE cnt&
```

This example determines the frame count of the active image.

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

GetMaskPresent (PHOTO-PAINT)

ReturnValue& = .GetMaskPresent()

This function returns a value indicating whether a mask is present.

Syntax	Definition
.ReturnValue&	Indicates whether a mask is present: 1 = Mask is present 0 = Mask is not present

Example

```
status& = .GetMaskPresent()
MESSAGE status&
```

This example determines whether a mask is present.

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

GetMaskRectangle (PHOTO-PAINT)

.GetMaskRectangle .Left = *long**, .Top = *long**, .Right = *long**, .Bottom = *long**

This function determines the coordinates of the active mask.

Syntax	Definition
.Left	Returns the X-coordinate of the upper-left corner of the mask's bounding box in pixels, relative to the origin.
.Top	Returns the Y-coordinate of the upper-left corner of the mask's bounding box in pixels, relative to the origin.
.Right	Returns the X-coordinate of the lower-right corner of the mask's bounding box in pixels, relative to the origin.
.Bottom	Returns the Y-coordinate of the lower-right corner of the mask's bounding box in pixels, relative to the origin.

Example

```
.GetMaskRectangle l&, t&, r&, b&
MESSAGE l&
MESSAGE t&
MESSAGE r&
MESSAGE b&
```

This example returns the coordinates of the mask's bounding box.

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

GetObjectCount (PHOTO-PAINT)

ReturnValue& = .GetObjectCount()

This function returns the number of objects in the active image.

Note

The background is not included in the object count.

Example

```
cnt& = .GetObjectCount()
MESSAGE cnt&
```

This example determines the number of objects in the active image.

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

GetObjectIsEditable (PHOTO-PAINT)

ReturnValue = .GetObjectIsEditable(long ObjectID)

This function returns a value indicating whether the specified object is editable
TRUE (-1) = Editable
FALSE (0) = Not editable

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) — the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.

Example

```
status& = .GetObjectIsEditable(2)
MESSAGE status&
```

This example determines whether the third object is editable.

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

GetObjectIsSelected (PHOTO-PAINT)

ReturnValue = .GetObjectIsSelected(long ObjectID)

This function returns a value indicating whether the specified object is selected:
TRUE (-1) = Selected
FALSE (0) = Not selected

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) — the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.

Note

Since the background cannot be selected, .ObjectID must be greater than 0.

Example

```
status& = .GetObjectIsSelected(2)
MESSAGE status&
```

This example determines whether the third object is selected.

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

GetObjectIsVisible (PHOTO-PAINT)

ReturnValue = .GetObjectIsVisible(long ObjectID)

This function returns a value indicating whether the specified object is visible or hidden:
TRUE (-1) = Visible
FALSE (0) = Hidden

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.

Example

```
status& = .GetObjectIsVisible(4)
MESSAGE status&
```

This example determines whether the fifth object is visible.

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

GetObjectMergeMode (PHOTO-PAINT)

ReturnValue& = .GetObjectMergeMode(long ObjectID)

This function returns a value indicating the Merge Mode for the specified object:

Syntax	Definition
.returnvalue&	Indicates the merge mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.

Note

Since the background merge mode cannot be queried, .ObjectID must be greater than 0.

Example

```
mode& = .GetObjectMergeMode(2)
MESSAGE mode&
```

This example determines the merge mode of the third object.

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

GetObjectName (PHOTO-PAINT)

ReturnValue& = .GetObjectName(long ObjectID)

This function returns a string containing the name of the specified object.

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.

Example

```
name$ = .GetObjectName(2)
MESSAGE name$
```

This example determines the name of the third object.

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

GetObjectOpacity (PHOTO-PAINT)

ReturnValue& = .GetObjectOpacity(long ObjectID)

This function returns a value (1 to 100) indicating the opacity of the specified object.

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.

Note

Since the background merge mode cannot be queried, .ObjectID must be greater than 0.

Example

```
opacity& = .GetObjectOpacity(2)
MESSAGE opacity&
```

This example determines the opacity of the third object.

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

GetObjectProperties (PHOTO-PAINT)

.GetObjectProperties .ObjectID = *long*, .Name = *string*, .Opacity = *long**, .MergeMode = *long**, .Visible = *boolean**, .Clipped = *boolean**, .Enabled = *boolean**, .Linked = *boolean**

This function returns the attributes or properties of the specified object.

Syntax	Description
.ObjectID	Specifes the index number of the object.
.Name	Specifies the name of the object.
.Opacity	Specifies the opacity of the object.*
.MergeMode	.MergeMode 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation Specifies the Merge Mode: 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.Visible	Specifies whether the object is visible or hidden. If the parameter is set to TRUE (-1), the object is visible. If the parameter is set to FALSE (0), the object is invisible.
.Clip	Specifies whether the object is clipped to its parent object. If the parameter is set to TRUE (-1), the object is clipped to its parent object. Otherwise, the parameter is set to FALSE (0).
.Enable	Specifies whether the object's clip mask is enabled. If the parameter is set to TRUE (-1), the object's clip mask is enabled. If the parameter is set to FALSE (0), the object's clip mask is disabled.
.Link	Specifies whether the object is linked to its clip mask. If the parameter is set to TRUE (-1), the object is linked to its clip mask. Otherwise, the parameter is set to FALSE (0).

Example

```
.GetObjectProperties
```

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

GetObjectRectangle (PHOTO-PAINT)

.GetObjectRectangle .ObjectID = *long*, .Left = *long**, .Top = *long**, .Right = *long**, .Bottom = *long**

This function determines the coordinates of the bounding box of the specified object.

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.
.Left	Returns the X-coordinate of the upper-left corner of the object's bounding box in pixels, relative to the origin.
.Top	Returns the Y-coordinate of the upper-left corner of the object's bounding box in pixels, relative to the origin.
.Right	Returns the X-coordinate of the lower-right corner of the object's bounding box in pixels, relative to the origin.
.Bottom	Returns the Y-coordinate of the lower-right corner of the object's bounding box in pixels, relative to the origin.

Example

```
.GetObjectRectangle 5, l&, t&, r&, b&
MESSAGE l&
MESSAGE t&
MESSAGE r&
MESSAGE b&
```

This example determines the coordinates of the sixth object's bounding box.

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

GetPaintColor (PHOTO-PAINT)

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

This function returns the color that is assigned to the Brush tool. The number and type of .Color variables returned depends on the current Color Model.

Syntax	Description
.ColorModel	Returns the currently active Color Model: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Returns the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Returns the second color component for .ColorModel. For example, Green is the second color component for RGB.
.Color3	Returns the third color component for .ColorModel. For example, Blue is the third color component for RGB.
.Color4	Returns the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK.

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

GetPaintVersion (PHOTO-PAINT)

ReturnValue& = .GetPaintVersion()

This function returns a string containing the version information. The string includes the word "Version" followed by the PHOTO-PAINT version number. For example, "Version 6.00.118".

Example

```
ver$ = .GetPaintVersion()
```

This example returns the version information.

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

GetPaperColor (PHOTO-PAINT)

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

This function returns the current paper (background) color. The number and type of .Color variables returned depends on the current Color Model.

Syntax	Description
.ColorModel	Returns the currently active Color Model: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Returns the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Returns the second color component for .ColorModel. For example, Green is the second color component for RGB.
.Color3	Returns the third color component for .ColorModel. For example, Blue is the third color component for RGB.
.Color4	Returns the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK.

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

GetPartialDocumentHeight (PHOTO-PAINT)

ReturnValue& = .GetPartialDocumentHeight()

This function returns the height of that portion of an image you choose to load using the Partial Load option in the Open dialog box, or the FileOpen command. Use the GetPartialDocumentWidth function to get the width of a partial image.

Syntax	Description
ReturnValue&	Indicates the height of the current partial image, in pixels.

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

GetPartialDocumentWidth (PHOTO-PAINT)

ReturnValue& = .GetPartialDocumentWidth()

This function returns the width of that portion of an image you choose to load using the Partial Load option in the Open dialog box, or the FileOpen command. Use the GetPartialDocumentHeight function to get the height of a partial image.

Syntax	Description
ReturnValue&	The width of the current partial image, in pixels.

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

GetPhotoPaintDir (PHOTO-PAINT)

ReturnString$ = .GetPhotoPaintDir()

This function returns the name of the folder where Corel PHOTO-PAINT is installed.

Syntax	Description
ReturnString$	Returns the name of the folder where Corel PHOTO-PAINT is installed.

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

GetSelectedObjectsRectangle (PHOTO-PAINT)

.GetSelectedObjectsRectangle .Left = *long**, .Top = *long**, .Right = *long**, .Bottom = *long**

This function determines the coordinates of the bounding box of the selected object(s).

Syntax	Definition
.Left	Returns the X-coordinate of the upper-left corner of the object's bounding box in pixels, relative to the origin.
.Top	Returns the Y-coordinate of the upper-left corner of the object's bounding box in pixels, relative to the origin.
.Right	Returns the X-coordinate of the lower-right corner of the object's bounding box in pixels, relative to the origin.
.Bottom	Returns the Y-coordinate of the lower-right corner of the object's bounding box in pixels, relative to the origin.

Example

```
.MaskRectangle 82, 146, 270, 248, 0, 0
.ObjectCreate 0
.GetSelectedObjectsRectangle l&, t&, r&, b&
MESSAGE l&
MESSAGE t&
MESSAGE r&
MESSAGE b&
```

This example returns the coordinates of the selected object's bounding box.

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

Image commands (PHOTO-PAINT)

DuotoneHandle (PHOTO-PAINT)

.DuotoneHandle .ToneNumber = *long*, .HandleNumber = *long*, .X = *long*, .Y = *long*

This command specifies the coordinates of the control handles on the tone curve of an ImageConvertDuotone command. This command must appear in an ImageConvertDuotone command block.

Syntax	Description
.ToneNumber	Specifies the tone to change. Valid values range from 0 to 3.
.HandleNumber	Specifies the tone curve handle you want to change.
.X	Specifies the horizontal coordinate of the tone curve handle. Valid values range from 0 to 255.
.Y	Specifies the vertical coordinate of the tone curve handle. Valid values range from 0 to 100.

Example

```
.ImageConvertDuotone 2, TRUE
    .OverprintColor 0, 2, 26, 97, 0
    .OverprintColor 1, 0, 100, 0, 100
    .OverprintColor 2, 100, 0, 0, 100
    ...
    .OverprintColor 10, 100, 100, 100, 100

    .DuotoneInfo 0, 3, 0, 0, 0, 255
    .DuotoneHandle 0, 0, 0, 0
    .DuotoneHandle 0, 1, 117, 66
    .DuotoneHandle 0, 2, 255, 100

    .DuotoneInfo 1, 4, 0, 0, 255, 0
    .DuotoneHandle 1, 0, 0, 0
    .DuotoneHandle 1, 1, 85, 73
    .DuotoneHandle 1, 2, 166, 40
    .DuotoneHandle 1, 3, 255, 100
    .EndConvertDuotone
```

This example converts a grayscale image to a duotone with black and yellow channels.

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

DuotoneInfo (PHOTO-PAINT)

.DuotoneInfo .ToneNumber = *long*, .Handles = *long*, .Cyan = *long*, .Magenta = *long*, .Yellow = *long*, .Black = *long*

This command sets the attributes of the individual tones for the ImageConvertDuotone command. This command must appear in an ImageConvertDuotone command block.

Syntax	Description
.ToneNumber	Specifies the tone to change. Valid values range from 0 to 3.
.Handles	Specifies the number of handles in the tone curve.
.Cyan	Specifies the Cyan channel of the tone.
.Magenta	Specifies the Magenta channel of the tone.
.Yellow	Specifies the Yellow channel of the tone.
.Black	Specifies the Black channel of the tone.

Example

```
.ImageConvertDuotone 2, TRUE
    .OverprintColor 0, 2, 26, 97, 0
    .OverprintColor 1, 0, 100, 0, 100
    .OverprintColor 2, 100, 0, 0, 100
    ...
    .OverprintColor 10, 100, 100, 100, 100

    .DuotoneInfo 0, 3, 0, 0, 0, 255
    .DuotoneHandle 0, 0, 0, 0
    .DuotoneHandle 0, 1, 117, 66
    .DuotoneHandle 0, 2, 255, 100

    .DuotoneInfo 1, 4, 0, 0, 255, 0
    .DuotoneHandle 1, 0, 0, 0
    .DuotoneHandle 1, 1, 85, 73
    .DuotoneHandle 1, 2, 166, 40
    .DuotoneHandle 1, 3, 255, 100
    .EndConvertDuotone
```

This example converts a grayscale image to a duotone with black and yellow channels.

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

EndColorTable (PHOTO-PAINT)

.EndColorTable

This command ends an ImageColorTable command block.

Example

```
.ImageColorTable 256
    .PaletteColor 5, 255, 0, 255, 0, 0
    .PaletteColor 5, 249, 0, 255, 0, 1
    ...
    .PaletteColor 5, 255, 255, 255, 0, 255
    .EndColorTable
```

This example changes the existing palette to a new palette containing 256 colors. A PaletteColor command is needed for each color in the new palette.

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

EndConvertDuotone (PHOTO-PAINT)

.EndConvertDuotone

This command ends an ImageConvertDuotone command block.

Example

```
.ImageConvertDuotone 2, TRUE
    .OverprintColor 0, 2, 26, 97, 0
    .OverprintColor 1, 0, 100, 0, 100
    .OverprintColor 2, 100, 0, 0, 100
    ...
    .OverprintColor 10, 100, 100, 100, 100

    .DuotoneInfo 0, 3, 0, 0, 0, 255
    .DuotoneHandle 0, 0, 0, 0
    .DuotoneHandle 0, 1, 117, 66
    .DuotoneHandle 0, 2, 255, 100

    .DuotoneInfo 1, 4, 0, 0, 255, 0
    .DuotoneHandle 1, 0, 0, 0
    .DuotoneHandle 1, 1, 85, 73
    .DuotoneHandle 1, 2, 166, 40
    .DuotoneHandle 1, 3, 255, 100
    .EndConvertDuotone
```

This example converts a grayscale image to a duotone with black and yellow channels.

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

EndConvertPaletted (PHOTO-PAINT)

.EndConvertPaletted

This command ends an ImageConvertPaletted command block.

Example

```
.ImageConvertPaletted 1, 216
    .PaletteColor 5, 0, 0, 0, 0, 0
    .PaletteColor 5, 51, 0, 0, 0, 1
    .PaletteColor 5, 102, 0, 0, 0, 2
    .PaletteColor 5, 153, 0, 0, 0, 3
    ...
    .PaletteColor 5, 153, 255, 255, 0, 213
    .PaletteColor 5, 204, 255, 255, 0, 214
    .PaletteColor 5, 255, 255, 255, 0, 215
    .EndConvertPaletted
```

This example converts an image to a paletted image and maps the existing colors to the new palette colors using a Uniform render.

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

EndImageEqualize (PHOTO-PAINT)

.EndImageEqualize

This command ends an ImageEqualize command block.

Example

```
.ImageEqualize 0, 5, 5, FALSE
    .ImageEqualizeChannel 0, 27, 255, 8, 249, 208
    .ImageEqualizeChannel 1, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 2, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 3, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 4, 0, 255, 0, 255, 100
    .EndImageEqualize
```

This example equalizes the overall RGB channel without enhancing any of the channels individually.

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

EndImageSTBalance (PHOTO-PAINT)

.EndImageSTBalance

This command ends a ImageSTBalance command block.

Example

```
.ImageSTBalance 0, TRUE, TRUE, TRUE, TRUE
    ' Source low color
    .ImageSTColor 0, 5, 249, 249, 153, 0
    ' target low color
    .ImageSTColor 1, 5, 249, 249, 153, 0
    ' source mid color
    .ImageSTColor 2, 5, 249, 249, 222, 0
    ' target mid color
    .ImageSTColor 3, 5, 249, 249, 153, 0
    ' source high color
    .ImageSTColor 4, 5, 222, 189, 222, 0
    ' target high color
    .ImageSTColor 5, 5, 222, 189, 222, 0
    .EndImageSTBalance
```

This example maps each of the low, mid, and high colors to the designated target colors.

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

EndImageToneCurve (PHOTO-PAINT)

.EndImageToneCurve

This command ends a ImageToneCurve command block.

Example

```
.ImageToneCurve
    .ImageToneTable 0, 0, 0, 0, 0
    .ImageToneTable 1, 1, 0, 1, 0
    .ImageToneTable 2, 1, 0, 1, 1
    .ImageToneTable 3, 2, 0, 2, 1
    ...
    .ImageToneTable 254, 255, 252, 255, 253
    .ImageToneTable 255, 255, 255, 255, 255
    .EndImageToneCurve
```

This example defines a new tone curve for the current image. Each of the 256 ImageToneTable commands defines one tone in the new tone curve.

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

ImageAutoEqualize (PHOTO-PAINT)

.ImageAutoEqualize .AutoBlack = *long*, .AutoWhite = *long*

This command performs a flat equalization on your image by automatically redistributing the significant pixel values of your image through the tonal range.

Syntax	Description
.AutoBlack	Specifies the current percentage of outlying pixels at the light end of the tonal range that Corel PHOTO-PAINT will ignore when performing an auto-equalization. The value ranges in steps from 1 to 1000; each step represents 0.05%.
.AutoWhite	Specifies the current percentage of outlying pixels at the dark end of the tonal range that Corel PHOTO-PAINT will ignore when performing an auto-equalization. The value ranges in steps from 1 to 1000; each step represents 0.05%.

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

ImageBCI (PHOTO-PAINT)

.ImageBCI .Brightness = *long*, .Contrast = *long*, .Intensity = *long*
.EndColorEffect

This command adjusts the brightness, contrast, and intensity of the tones in your image. The Brightness parameter shifts all pixel values up or down the tonal range. When you adjust the brightness, you lighten or darken all colors equally. The Contrast parameter adjusts the distance between your lightest and darkest pixels. When you increase the intensity, you brighten the lighter areas of your image without washing out the dark areas. Contrast and intensity usually go hand-in-hand. An increase in contrast sometimes washes out detail in shadows and highlights, and an increase in intensity can bring it back.

Syntax	Description
.Brightness	Specifies the Brightness of the current image. The value can range from -100 to 100.
.Contrast	Specifies the Contrast of the current image. The value can range from -100 to 100.
.Intensity	Specifies the Intensity of the current image. The value can range from -100 to 100.

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

ImageColorBalance (PHOTO-PAINT)

.ImageColorBalance .Red = *long*, .Green = *long*, .Blue = *long*, .Shadows = *boolean*, .Midtones = *boolean*, .Highlights = *boolean*, .Luminance = *boolean*
.EndColorEffect

This command shifts the colors in your image. This is useful for correcting color casts in your image. For example, if someone's face is too red in your photograph, you could shift values from red to cyan. You can also use the Color Balance filter to change the hue values for your entire image.

Syntax	Description
.Red	Specifies the balance of red in your image. The value can range from -100 to 100.
.Green	Specifies the balance of green in your image. The value can range from -100 to 100.
.Blue	Specifies the balance of blue in your image. The value can range from -100 to 100.
.Shadows	Set to TRUE (-1) to apply the color changes to the darkest pixels in the tonal range. Set to FALSE (0) to preserve the color of the dark pixels.
.Midtones	Set to TRUE (-1) to apply the color changes to the midtones of your image. Set to FALSE (0) to preserve the color of the midtone pixels.
.Highlights	Set to TRUE (-1) to apply the color changes to the lightest pixels in the tonal range. Set to FALSE (0) to preserve the color of the light pixels.
.Luminance	Set to TRUE (-1) to maintain the brightness values of your image. If you set this parameter to FALSE (0), the overall lightness or darkness of your image may be affected by color correction.

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

ImageColorCrop (PHOTO-PAINT)

.ImageColorCrop .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*, .ToleranceMode = *long*, .Normal = *long*, .Hue = *long*, .Saturation = *long*, .Brightness = *long*

This command crops a specific color border surrounding an image to the point where a different colored pixel is encountered. The cropping produces a new image with as much of the border removed without affecting the principle image.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.
.ToleranceMode	Specifies the tolerance mode: 0 = Normal 1 = HSB
.Normal	Specifies the tolerance as a percentage. This parameter is used only if .ToleranceMode is set to 0.
.Hue	Specifies the hue tolerance as a percentage. In the HSB color model, hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. This parameter is used only if .ToleranceMode is set to 1.
.Saturation	Specifies the saturation tolerance as a percentage. Saturation is the purity of a color. The more colors used to mix a color, the duller the color looks. This parameter is used only if .ToleranceMode is set to 1.
.Brightness	Specifies the brightness tolerance as a percentage. In the HSB color model, the component that determines the amount of black in a color. This parameter is used only if .ToleranceMode is set to 1.

Example

```
.ImageColorCrop 5, 0, 0, 204, 0, 0, 0, 0, 0, 0
```

This example crops out a blue color border, using a normal tolerance of 0.

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

ImageColorTable (PHOTO-PAINT)

.ImageColorTable .Colors = *long*

This command changes the palette of a paletted image, using a series of PaletteColor commands inside a command block. ImageColorTable command blocks must end with an EndColorTable command.

Syntax	Description
.Colors	Specifies the number of colors in the palette. Valid values range from 1 to 256.

Example

```
.ImageColorTable 256
    .PaletteColor 5, 255, 0, 255, 0, 0
    .PaletteColor 5, 249, 0, 255, 0, 1
    ...
    .PaletteColor 5, 255, 255, 255, 0, 255
    .EndColorTable
```

This example changes the existing palette to a new palette containing 256 colors. A PaletteColor command is needed for each color in the new palette.

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

ImageColorTone (PHOTO-PAINT)

.ImageColorTone .Hue = *long*, .Saturation = *long*, .Lightness = *long*, .Brightness = *long*, .Contrast = *long*, .Intensity = *long*

This command changes the color tone of your image using one or both of the HSL and BCI color models.

Syntax	Description
.Hue	Specifies the hue of the current image.
.Saturation	Specifies the saturation of the current image.
.Lightness	Specifies the lightness of the current image.
.Brightness	Specifies the Brightness of the current image.
.Contrast	Specifies the Contrast of the current image.
.Intensity	Specifies the Intensity of the current image.

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

ImageConvert (PHOTO-PAINT)

.ImageConvert .Type = *long*, .RenderType = *long*, .PaletteType = *long*, .Threshold = *short*, .HalftoneType = *long*, .Angle = *long*, .Width = *long*, .Number = *long*

This command converts the loaded image to another graphic format.

Syntax	Definition
.Type	Specifies the image type: 1 = RGB 2 = 256 Grayscale 3 = Black and white 4 = Paletted 5 = CMYK 7 = LAB
.RenderType	Specifies the type of rendering to apply 1 = None 2 = Error Diffusion 3 = Ordered
.PaletteType	Specifies the palette type used (only applies when converting 256 Color images; parameter value ignored in other cases): 100 = Optimized 101 = Adaptive 102 = Uniform 104 = StdVGA
.Threshold	Darkens the image. Valid threshold values range from 1 to 255, and are only used to convert Black and white with a .RenderType equal to 1.
.HalftoneType	Specifies the Halftone types of the image (only applies when converting Black and white images; parameter value ignored in other cases): 0 = None 1 = Square 2 = Round 3 = Line 4 = Cross
.Angle	Specifies the angle of conversion, in degrees, to apply to the image (only applies when .HalftoneType does not equal 0; parameter value ignored in other cases). Positive numbers result in counter-clockwise rotation; negative numbers result in clockwise rotation.
.Width	Specifies the halftone width in pixels (only applies when .HalftoneType does not equal 0; parameter value ignored in other cases).
.Number	Specifies the number of colors in the palette for conversion to a paletted image. Values are 1 to 256.

Note

When converting with a custom palette, the user should use .ImageConvertPaletted. Use the ImageConvertDuotone command to convert to a duotone.

Example

```
.ImageConvert 4, 2, 100, 0, 0, 0, 0, 64
```

This example converts an image to a Paletted image using an optimized palette and error diffusion with 64 colors.

```
.ImageConvert 2, 1, 0, 0, 0, 0, 0, 0
```

This example converts an image to grayscale.

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

ImageConvertDuotone (PHOTO-PAINT)

.ImageConvertDuotone .Style = *long*, .UseOverprints = *boolean*

This command converts a grayscale image to a duotone image. An ImageConvertDuotone command block contains several commands that set attributes of the conversion process, and must end with an EndConvertDuotone command.

Syntax	Description
.Style	Specifies the type of image to create: 0 = Monotone 1 = Duotone 2 = Tritone 3 = Quadtone
.UseOverprints	Set to TRUE (-1) to use overprints.

Example

```
.ImageConvertDuotone 2, TRUE
    .OverprintColor 0, 2, 26, 97, 0
    .OverprintColor 1, 0, 100, 0, 100
    .OverprintColor 2, 100, 0, 0, 100
    ...
    .OverprintColor 10, 100, 100, 100, 100

    .DuotoneInfo 0, 3, 0, 0, 0, 255
    .DuotoneHandle 0, 0, 0, 0
    .DuotoneHandle 0, 1, 117, 66
    .DuotoneHandle 0, 2, 255, 100

    .DuotoneInfo 1, 4, 0, 0, 255, 0
    .DuotoneHandle 1, 0, 0, 0
    .DuotoneHandle 1, 1, 85, 73
    .DuotoneHandle 1, 2, 166, 40
    .DuotoneHandle 1, 3, 255, 100
    .EndConvertDuotone
```

This example converts a grayscale image to a duotone with black and yellow channels.

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

ImageConvertPaletted (PHOTO-PAINT)

.ImageConvertPaletted .RenderType = *long*, .Colors = *long*

This command converts the current document to a paletted image using the palette colors specified inside the command block. An ImageConvertPaletted command block must end with an EndConvertPaletted command.

Syntax	Description
.RenderType	Specifies the palette type: 1 = None 2 = Error Diffusion 3 = Ordered
.Colors	Specifies the number of colors to include in an Adaptive or Optimized palette. Additional colors will not be added if you select more colors than are used in the image. Black and white images are the exception: a palette with 256 shades of grays will be created on conversion. Valid values range from 1 to 256.

Example

```
.ImageConvertPaletted 1, 216
    .PaletteColor 5, 0, 0, 0, 0, 0
    .PaletteColor 5, 51, 0, 0, 0, 1
    .PaletteColor 5, 102, 0, 0, 0, 2
    .PaletteColor 5, 153, 0, 0, 0, 3
    ...
    .PaletteColor 5, 153, 255, 255, 0, 213
    .PaletteColor 5, 204, 255, 255, 0, 214
    .PaletteColor 5, 255, 255, 255, 0, 215
    .EndConvertPaletted
```

This example converts an image to a paletted image, mapping the existing colors to the new palette colors using a Uniform render.

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

ImageConvertVideoNTSC (PHOTO-PAINT)

.ImageConvertVideoNTSC

This command converts the current document to the NTSC RGB video color mode, using the colors in the NTSC RGB gamut. Convert your 24-bit RGB images to the video color mode to create images with colors that are suitable for television reproduction. This prevents oversaturation and retains the integrity of your image when it is broadcast.

Example

```
.ImageConvertVideoNTSC
```

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

ImageCrop (PHOTO-PAINT)

.ImageCrop .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*

This command crops the current image to the specified rectangle.

Syntax	Definition
.Left	Specifies the X-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the image's bounding box in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the image's bounding box in pixels, relative to the origin.

Example

```
.ImageCrop 41, 154, 173, 75
```

This example crops the image to the specified rectangle.

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

ImageCropToMask (PHOTO-PAINT)

.ImageCropToMask

This command crops the selected image to the bounding box of the current mask.

Note

A mask must be present before this command can be used.

Example

```
.ImageCropToMask
```

This example crops the selected image to the size of the current mask.

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

ImageDeInterlace (PHOTO-PAINT)

.ImageDeInterlace .ReplaceMode = *long*

This command removes even or odd horizontal lines from scanned or interlaced video images. You can fill the spaces left by the discarded lines using either of two methods: duplication fills in the spaces with copies of the adjacent lines of pixels, while interpolation fills them in with colors created by averaging the surrounding pixels.

Syntax	Description
.ReplaceMode	Specifies the deinterlace fill method: 0 = Replaces even lines using the duplication method 1 = Replaces even lines using the interpolation method 2 = Replaces odd lines using the duplication method 3 = Replaces odd lines using the interpolation method

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

ImageDesaturate (PHOTO-PAINT)

.ImageDesaturate
.EndColorEffect

This command reduces the saturation of each color in your image to 0, which converts each color to its grayscale equivalent. This makes your image appear to be grayscale without having to convert it.

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

ImageDeskew (PHOTO-PAINT)

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

This command adjusts a skewed or imperfectly positioned image and places it squarely on the screen. This command is especially valuable when you are working with scanned images.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Note

The color parameters specify the paper or border color around the image, which is used to the find the image edges

Example

```
.ImageDeskew 3, 0, 119, 211,0
```

This example uses the CMYK Color mode with Magenta and Yellow colors being applied.

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

ImageDeskewCrop (PHOTO-PAINT)

.ImageDeskewCrop .Angle = *double*, .Width = *long*, .Height = *long*, .PointX = *long*, .PointY = *long*

This command crops an image. If the angle is set to a value other than 0, the image is also deskewed.

Syntax	Description
.Angle	Specifies the angle of rotation. Values range from -360 - 360 degrees.
.Width	Specifies the width of the crop marquee.
.Height	Specifies the height of the crop marquee.
.PointX	Specifies the horizontal coordinate for the left edge of the cropped area.
.PointY	Specifies the vertical coordinate for the top of the cropped area.

Example

```
.ImageDeskewCrop -11.7, 200, 200, 264, 140
```

This example crops a rectangle to 200 pixels by 200 pixels and skews it 11.7 degrees. The center point is at 264,140.

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

ImageDuplicate (PHOTO-PAINT)

.ImageDuplicate .FileName = *string*, .MergeObjects = *boolean*

This command creates a copy of the current image and assigns it the specified name.

Syntax	Definition
.FileName	Specifies the new name of the duplicated image.
.MergeObjects	Set to TRUE (-1) to enable objects to be merged. Set to FALSE (0) to disable merging of objects.

Example

```
.ImageDuplicate "NEW2.CPT", FALSE
```

This example duplicates the file named NEW2.CPT, without merging all objects in the duplicated image.

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

ImageEqualize (PHOTO-PAINT)

.ImageEqualize .Method = *long*, .AutoBlack = *long*, .AutoWhite = *Long*, .AutoAdjust = *boolean*

This command equalizes the current image. An ImageEqualize command block must end with an EndImageEqualize command.

Syntax	Description
.Method	Specifies the equalization method: 0 = Proportional 1 = Nonproportional
.AutoBlack	Specifies the percentage of outlying pixels at the dark end of the tonal range that Corel PHOTO-PAINT will ignore when performing an auto-equalization. Valid values range from 0 to 100%.
.AutoWhite	Specifies the percentage of outlying pixels at the light end of the tonal range that Corel PHOTO-PAINT will ignore when performing an auto-equalization. Valid values range from 0 to 100%.
.AutoAdjust	Set to TRUE (-1) to let Corel PHOTO-PAINT determine the .AutoBlack and .AutoWhite parameters.

Example

```
.ImageEqualize 0, 5, 5, FALSE
    .ImageEqualizeChannel 0, 27, 255, 8, 249, 208
    .ImageEqualizeChannel 1, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 2, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 3, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 4, 0, 255, 0, 255, 100
    .EndImageEqualize
```

This example equalizes the overall RGB channel without enhancing any of the channels individually.

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

ImageEqualizeChannel (PHOTO-PAINT)

.ImageEqualizeChannel .Index = *long*, .InLow = *long*, .InHigh = *long*, .OutLow = *long*, .OutHigh = *long*, .Gamma = *long*

This command sets the channel equalization attributes for the ImageEqualize command. This command must appear inside an ImageEqualize command block.

Syntax	Description
.Index	Specifies the equalization channel: 0 = RGB channels 1 = Red channel 2 = Green channel 3 = Blue channel
.InLow	Specifies a clipping range for the darkest pixels in your image. All pixels that fall between this value and the .OutLow value will map to the darkest pixel value. Valid values range from 0 to 254.
.InHigh	Specifies a clipping range for the brightest pixels in your image. All pixels that fall between this value and the .OutHigh value will map to the brightest pixel value. Valid values range from 1 to 255.
.OutLow	Specifies the output brightness value of the darkest pixels in your image. Valid values range from 0 to 254.
.OutHigh	Specifies the output brightness value of the lightest pixels in your image. Valid values range from 1 to 255.
.Gamma	Specifies the gamma curve value. Adjusting the gamma curve value allows you to pick up detail in a low contrast image without significantly affecting the shadows or highlights. It does affect all the values in your image, but is curve-based so the changes are weighted toward the midtones. Valid values range from 1 to 1000.

Example

```
.ImageEqualize 0, 5, 5, FALSE
    .ImageEqualizeChannel 0, 27, 255, 8, 249, 208
    .ImageEqualizeChannel 1, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 2, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 3, 0, 255, 0, 255, 100
    .ImageEqualizeChannel 4, 0, 255, 0, 255, 100
    .EndImageEqualize
```

This example equalizes the overall RGB channel without enhancing any of the channels individually.

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

ImageFlipHorizontal (PHOTO-PAINT)

.ImageFlipHorizontal

This command flips the image horizontally.

Example

```
.ImageFlipHorizontal
```

This example flips the active image horizontally.

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

ImageFlipVertical (PHOTO-PAINT)

.ImageFlipVertical

This command flips the image vertically.

Example

```
.ImageFlipVertical
```

This example flips the active image vertically.

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

ImageGamma (PHOTO-PAINT)

.ImageGamma .Value = *long*
.EndColorEffect

This command picks out detail in low contrast images without significantly affecting shadows or highlights. It does affect all the values in your image, but is curve-based so that the changes are weighted toward the midtones.

Syntax	Description
.Value	Specifies the degree of gamma correction. Valid values range from 0.1 to 10.

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

ImageHSL (PHOTO-PAINT)

.ImageHSL .Hue = *long*, .Saturation = *long*, .Lightness = *long*
.EndColorEffect

This command adjust the colors in your image using HLS (Hue, Lightness, and Saturation) values

Syntax	Description
.Hue	Specifies the Hue tolerance. In the HSB color model, hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. Valid values range from -100 to 100.
.Saturation	Specifies the Saturation tolerance. Saturation is the purity of a color. The HSB color model uses Saturation as a component that determines the purity or intensity of a color. The more colors used to mix a color, the duller the color looks. Valid values range from -100 to 100.
.Lightness	Specifies the Lightness tolerance. Lightness is the amount of black or white in a color. Valid values range from -100 to 100.

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

ImageHSLChannel (PHOTO-PAINT)

.ImageHSLChannel .Channel = *long*, .Hue = *long*, .Saturation = *long*, .Lightness = *long*

This command specifies the HSL values for each channel in an image. The .ImageHSL command sets the values for the master channel.

Syntax	Description
.Channel	Specifies the channel 1 = red 2 = yellow 3 = green 4 = cyan 5 = blue 6 = magenta 7 = Grayscale
.Hue	Specifies the Hue tolerance. In the HSB color model, hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. Valid values range from -180 to 180.
.Saturation	Specifies the Saturation tolerance. Saturation is the purity of a color. The HSB color model uses Saturation as a component that determines the purity or intensity of a color. The more colors used to mix a color, the duller the color looks. Valid values range from -100 to 100.
.Lightness	Specifies the Lightness tolerance. Lightness is the amount of black or white in a color. Valid values range from -100 to 100.

Example

```
.ImageHSL -46, 48, -29
        .ImageHSLChannel 1, 180, -100, -100
        .ImageHSLChannel 2, 0, -18, 25
        .ImageHSLChannel 3, 0, 34, 47
        .ImageHSLChannel 4, 69, -32, 0
        .ImageHSLChannel 5, 0, 19, 40
        .ImageHSLChannel 6, -125, 0, 24
        .ImageHSLChannel 7, -110, 22, -31
        .EndColorEffect
```

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

ImageInvert (PHOTO-PAINT)

.ImageInvert
.EndColorEffect

This command inverts the colors in your image, producing an effect much like a photographic negative.

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

ImageLevelThreshold (PHOTO-PAINT)

.ImageLevelThreshold .Channel = *long*, .Low = *long*, .Threshold = *long*, .High = *Long*, .BiLevel = *long*
.EndColorEffect

This command converts certain shades of each color in an image to black or white. In bi-level mode, the command can convert shades to both black and white at the same time.

Syntax	Description
.Channel	Specifies the channel to convert: 0 = All three channels (RGB) at once 1 = Red channel 2 = Green channel 3 = Blue Channel
.Low	Specifies the low-level value. Valid values range from 0 to 255.
.Threshold	Specifies the threshold value. Valid values range from 0 to 255.
.High	Specifies the high-level value. Valid values range from 0 to 255.
.BiLevel	Specifies the conversion method: 0 = To black 1 = To White 2 = Bi-level

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

ImagePapersize (PHOTO-PAINT)

.ImagePapersize .Width = *long*, .Height = *long*, .Xoffset = *long*, .Yoffset = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command adjusts the size of the paper behind an image using absolute width and height values. The image can be repositioned on the paper by setting a placement position.

Syntax	Definition
.Width	Specifies the paper width in pixels.
.Height	Specifies the paper height in pixels.
.Xoffset	Specifies the placement of the image on the paper offset from the x-axis (expressed in pixels).
.Yoffset	Specifies the placement of the image on the paper offset from the y-axis (expressed in pixels).
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.ImagePapersize 408, 528, 0, 0, 5, 255, 200, 100, 0
```

This example sets the paper size to 408 pixels (width) by 528 pixels (height) with no offset applied.

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

ImagePosterize (PHOTO-PAINT)

.ImagePosterize .Level = *long*
.EndColorEffect

This command transforms the color range of your image to solid blocks of color, reducing gradual blends to hard edges between areas of color.

Syntax	Description
.Level	Specifies the intensity of the posterization effect (i.e. the number of colors in the final image). Valid values range from 2 to 32.

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

ImageReplaceColors (PHOTO-PAINT)

.ImageReplaceColors .OrgH = *long*, .OrgS = *long*, .OrgL = *long*, .DestH = *long*, .DestS = *long*, .DestL = *long*, .Range = *long*, .IgnoreGrayscale = *bool*
.EndColorEffect

This command replaces one color in your image with another color. Depending on the .Range value, the command will replace a single color, or shift the entire image from one range of color to another.

Syntax	Description
.OrgH	The Hue value of the original color. Valid values range from 0 to 6144.
.OrgS	The Saturation value of the original color. Valid values range from 0 to 1024.
.OrgL	The Lightness value of the original color. Valid values range from 0 to 1024.
.DestH	The Hue value of the destination color. Valid values range from -180 to 180.
.DestS	The Saturation value of the destination color. Valid values range from -100 to 100.
.DestL	The Lightness value of the destination color. Valid values range from -100 to 100.
.Range	The range of color that will be replaced by the command. Valid values range from 1 to 100.
.IgnoreGrayscale	Determines whether grayscale values are ignored.

Example

```
.ImageReplaceColors 3460, 562, 795, -150, 100, 75, 34, 1
.EndColorEffect
```

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

ImageResample (PHOTO-PAINT)

.ImageResample .Width = *long*, .Height = *long*, .HRes = *long*, .VRes = *long*, .AntiAlias = *boolean*

This command adjusts the dimensions and resolution of an image.

Syntax	Definition
.Width	Specifies the new width of the image in pixels.
.Height	Specifies the new height of the image in pixels.
.HRes	Specifies the new horizontal resolution of the image in dots per inch (dpi).
.VRes	Specifies the new vertical resolution of the image in dots per inch (dpi).
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.

Example

```
.ImageResample 410, 256, 75, 75, -1
```

This example displays the original image with a width of 410 pixels, a height of 256 pixels, horizontal and vertical resolutions of 75 dpi, applying anti-aliasing.

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

ImageRotate (PHOTO-PAINT)

.ImageRotate .Angle = *double*, .Clip = *boolean*, .AntiAlias = *boolean*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command rotates the image a specified angle.

Syntax	Definition
.Angle	Specifies the image rotation expressed in degrees. Positive numbers result in counter-clockwise rotation, negative numbers result in clockwise rotation.
.Clip	Set to TRUE (-1) to clip the image to original size. Set to FALSE (0) to increase the image size.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.ImageRotate 315, TRUE, TRUE, 5, 255, 204, 53, 0
```

This example rotates the image 45 degrees clockwise, maintaining the original size and applying anti-aliasing.

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

ImageSelectiveColor (PHOTO-PAINT)

.ImageSelectiveColor .Mode = *long*

This command applies the Selective Color effects.

Syntax	Description
.Mode	Specifies the mode 1 = Relative 0 = Absolute

Example

```
.ImageSelectiveColor 1
        .ImageSelectiveColorChannel 0, 0, 30, 42, -20
        .ImageSelectiveColorChannel 1, 19, 15, 17, 3
        .ImageSelectiveColorChannel 2, 30, 0, 28, 0
        .ImageSelectiveColorChannel 3, -42, 0, 29, 29
        .ImageSelectiveColorChannel 4, 50, 33, 0, -44
        .ImageSelectiveColorChannel 5, -47, -46, 0, 30
        .ImageSelectiveColorChannel 6, 0, 0, 0, 10
        .ImageSelectiveColorChannel 7, 0, 0, 0, -29
        .ImageSelectiveColorChannel 8, 0, 0, 0, 28
        .EndColorEffect
```

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

ImageSelectiveColorChannel (PHOTO-PAINT)

.ImageSelectiveColorChannel .Channel = *long*, .Cyan = *long*, .Magenta = *long*, .Yellow = *long*, .Black = *long*

This command specifies the color channel used to apply the Selective Color effect.

Syntax	Description
.Channel	Specifies the channel 0 = red 1 = yellow 2 = green 3 = cyan 4 = blue 5 = magenta 6 = shadow 7 = midtone 8 = highlight
.Cyan	Specifies the cyan color value. Values range from -100 - 100.
.Magenta	Specifies the magenta color value. Values range from -100 - 100.
.Yellow	Specifies the yellow color value. Values range from -100 - 100.
.Black	Specifies the black color value. Values range from -100 - 100.

Example

```
.ImageSelectiveColor 1
.ImageSelectiveColorChannel
.EndColorEffect
```

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

ImageSetChannel (PHOTO-PAINT)

.ImageSetChannel .Channel = *long*

This command selects the image channel or channels to be edited. Valid only for RGB and CMYK images. A channel is similar to a plate in the printing process.

Syntax	Definition
.Channel	Specifies the channel(s) to be set For RGB images: 0 = Blue 1 = Green 2 = Red -1 = All channels For CMYK images: 0 = Cyan 1 = Magenta 2 = Yellow 3 = Black -1 = All channels

Example

```
.ImageSetChannel 2
```

This example selects the red image channel to be edited.

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

ImageSplit (PHOTO-PAINT)

.ImageSplit .Type = *long*

This command separates an image into the color channels corresponding to the specified color model.

Syntax	Definition
.Type	Specifies how to split the image into channels: 0 = split the image into RGB channels 1 = split the image into CMYK channels 2 = split the image into HSB channels 3 = split the image into HLS channels 4 = split the image into YIQ channels 5 = split the image into LAB channels

Note

- RGB images can be split into all color models except LAB. LAB images can only be split into LAB channels, and CMYK images can only be split to CMYK channels.

Example

```
.ImageSplit 4
```

This example splits an RGB image into YIQ channels.

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

ImageSprayerSettings (PHOTO-PAINT)

.ImageSprayerSettings .Dabs = *long*, .Spacing = *long*, .Spread = *long*, .FadeOut = *long*, .Type = *long*, .From = *long*, .To = *long*, .Start = *long*

This command sets the brush stroke attributes of the Image Sprayer tool. This command must appear in an ImageSprayerTool command block.

Syntax	Description
.Dabs	Specifies the number of dabs in the brush stroke. Valid values range from 1 to 25.
.Spacing	Specifies the spacing between dabs along the length of the stroke. Valid values range from 1 to 999 pixels.
.Spread	Specifies the spacing between dabs along the width of the stroke. Valid values range from 1 to 999 pixels.
.FadeOut	Specifies the length of the brush stroke before it fades out. Valid values range from 0 to 100.
.Type	Specifies the order in which the images will be sprayed in each brush stroke: 0 = Random 1 = Sequential 2 = Directional
.From	The number of the first image in the sequence.
.To	The number of the last image in the sequence.
.Start	Specifies the image in the sequence with which to start painting.

Example

```
.ImageSprayerTool "fruit.cpt", 1, 5, 87, 0, 0
    .ImageSprayerSettings 1, 11, 19, 0, 0, 1, 5, 1
    .RandomSeed 552585248
    .StartDraw 11904, 8320, 0, 0
    .ContinueDraw 14343, 9096, 0, 0
    ...
    .ContinueDraw 61761, 18445, 0, 0
    .EndDraw
```

This example draws a brush stroke containing a variety of fruits.

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

ImageSprayerTool (PHOTO-PAINT)

.ImageSprayerTool .FileName = *string*, .Row = *long*, .Column = *long*, .Size = *long*, .MergeMode = *long*, .Transparency = *long*

This command paints a variety of images from an image file along a brush stroke. An ImageSprayerTool command block must contain a series of StartDraw and ContinueDraw commands, and end with an EndDraw command.

Syntax	Description
.FileName	Specifies the name of the image list to use.
.Row	Specifies the number of rows of images.
.Column	Specifies the number of columns of images.
.Size	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.Transparency	Specifies the transparency level of the nib. Valid values range from 0 to 99%.

Example

```
.ImageSprayerTool "fruit.cpt", 1, 5, 87, 0, 0
    .ImageSprayerSettings 1, 11, 19, 0, 0, 1, 5, 1
    .RandomSeed 552585248
    .StartDraw 11904, 8320, 0, 0
    .ContinueDraw 14343, 9096, 0, 0
    ...
    .ContinueDraw 61761, 18445, 0, 0
    .EndDraw
```

This example draws a brush stroke containing a variety of fruits.

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

ImageSTBalance (PHOTO-PAINT)

.ImageSTBalance .Channel = *long*, .UseLow = *boolean*, .UseMid = *boolean*, .UseHigh = *boolean*, .UseAll = *boolean*

This command changes a specific color in your image to a target color. An ImageSTBalance command block must contain a series of ImageSTColor commands, and end with an EndImageSTBalance.

Syntax	Description
.Channel	Specifies the channel you want to change: 0 = All three channels (RGB) at once 1 = Red channel 2 = Green channel 3 = Blue Channel
.UseLow	Set to TRUE (-1) to use the low point color values set with the ImageSTColor commands in the command block.
.UseMid	Set to TRUE (-1) to use the mid point color values set with the ImageSTColor commands in the command block.
.UseHigh	Set to TRUE (-1) to use the high point color values set with the ImageSTColor commands in the command block.
.UseAll	Set to TRUE (-1) to use all of the color values set with the ImageSTColor commands in the command block.

Example

```
.ImageSTBalance 0, TRUE, TRUE, TRUE, TRUE
    ' Source low color
    .ImageSTColor 0, 5, 249, 249, 153, 0
    ' target low color
    .ImageSTColor 1, 5, 249, 249, 153, 0
    ' source mid color
    .ImageSTColor 2, 5, 249, 249, 222, 0
    ' target mid color
    .ImageSTColor 3, 5, 249, 249, 153, 0
    ' source high color
    .ImageSTColor 4, 5, 222, 189, 222, 0
    ' target high color
    .ImageSTColor 5, 5, 222, 189, 222, 0
    .EndImageSTBalance
```

This example maps each of the low, mid, and high colors to the designated target colors.

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

ImageSTColor (PHOTO-PAINT)

.ImageSTColor .Index = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command sets the source and target colors for the ImageSTBalance command. This command must occur in an ImageSTBalance command block.

Syntax	Description
.Index	Specifies the source or target colors to define: 0 = Source low color 1 = Target low color 2 = Source mid color 3 = Target mid color 4 = Source high color 5 = Target high color
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Note

- The order of the ImageSTColor commands is important. The definitions of source and target commands must occur in pairs for the command to work.

Example

```
.ImageSTBalance 0, TRUE, TRUE, TRUE, TRUE
    ' Source low color
    .ImageSTColor 0, 5, 249, 249, 153, 0
    ' target low color
    .ImageSTColor 1, 5, 249, 249, 153, 0
    ' source mid color
    .ImageSTColor 2, 5, 249, 249, 222, 0
    ' target mid color
    .ImageSTColor 3, 5, 249, 249, 153, 0
    ' source high color
    .ImageSTColor 4, 5, 222, 189, 222, 0
    ' target high color
    .ImageSTColor 5, 5, 222, 189, 222, 0
    .EndImageSTBalance
```

This example maps each of the low, mid, and high colors to the designated target colors.

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

ImageToneCurve (PHOTO-PAINT)

.ImageToneCurve

This command adjusts the tone curve of the current image.

If you want to apply the adjustment directly to the image pixels, add the ImageToneTable command to it, and end the command block with the EndImageToneCurve command.

On the other hand, if you want to apply the adjustment to a lens object, you must add the ImageToneChannel and ImageTonePoint commands, and end the command block with the EndColorEffect command. Examples are provided for both options.

Example when applying to the image

```
.ImageToneCurve
    .ImageToneTable 0, 0, 0, 0, 0
    .ImageToneTable 1, 1, 0, 1, 0
    .ImageToneTable 2, 1, 0, 1, 1
    .ImageToneTable 3, 2, 0, 2, 1
    ...
    .ImageToneTable 254, 255, 252, 255, 253
    .ImageToneTable 255, 255, 255, 255, 255
    .EndImageToneCurve
```

This example defines a new tone curve for the current image. Each of the 256 ImageToneTable commands defines one tone in the new tone curve.

Example when applying to a lens

```
.ImageToneCurve
    .ImageToneChannel 0, 2, 100, 256, 0, 1
    .ImageTonePoint 0, 0, 0, 0
    .ImageTonePoint 0, 1, 255, 255
    .ImageToneChannel 1, 2, 100, 256, 0, 1
    .ImageTonePoint 1, 0, 0, 0
    .ImageTonePoint 1, 1, 255, 255
    .ImageToneChannel 2, 2, 100, 256, 0, 1
    .ImageTonePoint 2, 0, 0, 0
    .ImageTonePoint 2, 1, 255, 255
    .ImageToneChannel 3, 2, 100, 256, 0, 1
    .ImageTonePoint 3, 0, 0, 0
    .ImageTonePoint 3, 1, 255, 255
    .ImageToneChannel 4, 4, 100, 256, 0, 1
    .ImageTonePoint 4, 0, 0, 0
    .ImageTonePoint 4, 1, 142, 72
    .ImageTonePoint 4, 2, 166, 173
    .ImageTonePoint 4, 3, 255, 255
    .LensNew "Tone Curve Lens 5"
    .EndColorEffect
```

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

ImageToneTable (PHOTO-PAINT)

.ImageToneTable .Number = *long*, .Curve1 = *long*, .Curve2 = *long*, .Curve3 = *long*, .Curve4 = *long*

This command sets the color of a single tone on the tone curve of the current image. This command must appear in an ImageToneCurve command block.

Syntax	Description
.Number	Specifies the index number of the individual tone to change.
.Curve1	Specifies the first color component of the tone using the current color model. For example, Red is the first color component for RGB.
.Curve2	Specifies the first color component of the tone using the current color model. For example, Green is the second color component for RGB. If this parameter is not available in the current color model, set it to 0.
.Curve3	Specifies the first color component of the tone using the current color model. For example, Blue is the third color component for RGB. If this parameter is not available in the current color model, set it to 0.
.Curve4	Specifies the first color component of the tone using the current color model. For example, Black is the fourth color component for CMYK. If this parameter is not available in the current color model, set it to 0.

Example

```
.ImageToneCurve
    .ImageToneTable 0, 0, 0, 0, 0
    .ImageToneTable 1, 1, 0, 1, 0
    .ImageToneTable 2, 1, 0, 1, 1
    .ImageToneTable 3, 2, 0, 2, 1
    ...
    .ImageToneTable 254, 255, 252, 255, 253
    .ImageToneTable 255, 255, 255, 255, 255
    .EndImageToneCurve
```

This example defines a new tone curve for the current image. Each of the 256 ImageToneTable commands defines one tone in the new tone curve.

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

ImageToneChannel (PHOTO-PAINT)

.ImageToneChannel .Channel=*long*, .Nodes=*long*, .Gamma=*long*, Length=*long*, .DrawMode=*long*, .Orientation=*long*

This command sets the overall attributes of the tone curve for a specific channel when applying the Tone Curve to a lens object. This command must appear in the ImageToneCurve command block when a new Tone Curve is used on a lens.

Syntax	Description
.Channel	Specifies the channel to which the next parameters apply 0 = Red or Cyan channel 1 = Green or Magenta channel 2 = Blue or Yellow channel 3 = Black 4 = All channels
.Nodes	Specifies the number of nodes on the tone curve for the selected channel Maximum number: 256
.Gamma	Specifies the gamma value for the channel. Gamma values range from 0 to 1000
.Length	Must always be 256
.DrawMode	Specifies the Edit Style selected for the tone curve 0 = Curve style 1 = Linear style 2 = Freehand style 3 = Gamma style
.Orientation	Determines the orientation of the horizontal and vertical axes of the tone curve graph 0 = the brightness value of the shadows are near the origin point 1 = the brightness value of the highlights are near the origin point

Example

```
.ImageToneCurve
    .ImageToneChannel 0, 2, 100, 256, 0, 1
    .ImageTonePoint 0, 0, 0, 0
    .ImageTonePoint 0, 1, 255, 255
    .ImageToneChannel 1, 2, 100, 256, 0, 1
    .ImageTonePoint 1, 0, 0, 0
    .ImageTonePoint 1, 1, 255, 255
    .ImageToneChannel 2, 2, 100, 256, 0, 1
    .ImageTonePoint 2, 0, 0, 0
    .ImageTonePoint 2, 1, 255, 255
    .ImageToneChannel 3, 2, 100, 256, 0, 1
    .ImageTonePoint 3, 0, 0, 0
    .ImageTonePoint 3, 1, 255, 255
    .ImageToneChannel 4, 4, 100, 256, 0, 1
    .ImageTonePoint 4, 0, 0, 0
    .ImageTonePoint 4, 1, 142, 72
    .ImageTonePoint 4, 2, 166, 173
    .ImageTonePoint 4, 3, 255, 255
    .LensNew "Tone Curve Lens 5"
    .EndColorEffect
```

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

ImageTonePoint (PHOTO-PAINT)

.ImageTonePoint .Channel=*long*, .Index=*long*, .X=*long*, .Y=*long*

This command identifies the precise location of each node found on a tone curve that will be applied to a lens object. This command must appear in the ImageToneCurve command block when a new Tone Curve is used on a lens.

Syntax	Description
.Channel	Specifies the channel for which you are setting the node location 0 = Red or Cyan channel 1 = Green or Magenta channel 2 = Blue or Yellow channel 3 = Black 4 = All channels
.Index	Specifies the number of the node (the maximum number is 256)
.X	Sets the location of the node on the horizontal axis (from 0 to 256)
.Y	Sets the location of the node on the vertical axis (from 0 to 256)

Example

```
.ImageToneCurve
    .ImageToneChannel 0, 2, 100, 256, 0, 1
    .ImageTonePoint 0, 0, 0, 0
    .ImageTonePoint 0, 1, 255, 255
    .ImageToneChannel 1, 2, 100, 256, 0, 1
    .ImageTonePoint 1, 0, 0, 0
    .ImageTonePoint 1, 1, 255, 255
    .ImageToneChannel 2, 2, 100, 256, 0, 1
    .ImageTonePoint 2, 0, 0, 0
    .ImageTonePoint 2, 1, 255, 255
    .ImageToneChannel 3, 2, 100, 256, 0, 1
    .ImageTonePoint 3, 0, 0, 0
    .ImageTonePoint 3, 1, 255, 255
    .ImageToneChannel 4, 4, 100, 256, 0, 1
    .ImageTonePoint 4, 0, 0, 0
    .ImageTonePoint 4, 1, 142, 72
    .ImageTonePoint 4, 2, 166, 173
    .ImageTonePoint 4, 3, 255, 255
    .LensNew "Tone Curve Lens 5"
    .EndColorEffect
```

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

PaletteColor (PHOTO-PAINT)

.PaletteColor .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*, .Index = *long*

This command sets a color in a custom palette. This command must appear as part of a block, and is used by such commands as ImageColorTable and ImageConvertPaletted.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.
.Index	Specifies the position of the color in the palette. Since .Index is 0-based, the first color is 0, the second color is 1, and so on.

Example

```
.ImageColorTable 256
    .PaletteColor 5, 255, 0, 255, 0, 0
    .PaletteColor 5, 249, 0, 255, 0, 1
    ...
    .PaletteColor 5, 255, 255, 255, 0, 255
    .EndColorTable
```

This example changes the existing palette to a new palette containing 256 colors. A PaletteColor command is needed for each color in the new palette.

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

Color Mask commands (PHOTO-PAINT)

ColorMaskColor (PHOTO-PAINT)

.ColorMaskColor .Number = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*, .Normal = *long*, .Hue = *long*, .Saturation = *long*, .Brightness = *long*

This command sets color attributes for the ColorMaskCreateMask and ColorMaskCreateChannel commands. This command must appear in a ColorMaskCreateMask or ColorMaskCreateChannel command block.

Syntax	Definition
.Number	Identifies the colors in the Color Mask. Valid values range from 1 to 10.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.
.Normal	Specifies the Normal-mode tolerance. Valid values range from 0 to 100%.
.Hue	Specifies the hue tolerance. In the HSB color model, hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. Valid values range from 0 to 100%.
.Saturation	Specifies the saturation tolerance. Saturation is the purity of a color. The HSB color model uses Saturation as a component that determines the purity or intensity of a color. The more colors used to mix a color, the duller the color looks. Valid values range from 0 to 100%.
.Brightness	Specifies the brightness tolerance. In the HSB color model, the component that determines the amount of black in a color. Valid values range from 0 to 100%.

Example

```
.ObjectColorTransparencyTool 0, TRUE, 27
    .ColorMaskColor 0, 5, 62, 155, 0, 0, 14, 30, 50, 10
    .ColorMaskColor 1, 5, 62, 155, 0, 0, 14, 30, 50, 10
    .ColorMaskColor 2, 5, 112, 102, 28, 0, 14, 30, 50, 10
    .EndObjectColorTransparencyTool
```

This example makes the selected colors transparent in the currently selected object using a clip mask.

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

ColorMaskCreateChannel (PHOTO-PAINT)

.ColorMaskCreateChannel .MaskMode = *long*, .Smoothing = *long*, .ToleranceMode = *long*, .Gamut = *boolean,* .Threshold = *long,* .BrightnessModel = *long,* .Name = *string*

This command creates a new color channel using colors defined with a series of ColorMaskColor commands. A ColorMaskCreateMask command block must end with an EndColorMask command.

Syntax	Description
.MaskMode	Specifies the mode of the channel 0 = Protect 1 = Modify
.Smoothing	Specifies the amount of smoothing to be used in the channel. Valid values range from 0 to 100.
.ToleranceMode	Specifies the tolerance mode to set: 0 = Normal 1 = HSB 2 = Hue 3 = Saturation 4 = Brightness
.Gamut	Set to TRUE (-1) to enable gamut control.
.Threshold	Converts selected colors to black. This command uses black to display the pixels with a brightness value that is below the threshold that you set. Masked areas are always displayed in black however, black may translate differently depending on the preview method you specify. Values range from 0 to 255.
.BrightnessModel	Converts selected colors to white. This command uses white to display the pixels with a brightness value that is above the threshold you set. Selected areas on an image are always displayed in white however, this may translate differently depending on the preview method you specify. 0=Black 1=White
.Name	Specifies the name of the color channel that is created.

Example

```
.ColorMaskCreateChannel 1, 100, 2, FALSE, 0, 0, "Alpha 1"
        .ColorMaskColor 0, 5, 0, 0, 0, 0, 18, 18, 18, 18
        .EndColorMask
```

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

ColorMaskCreateMask (PHOTO-PAINT)

.ColorMaskCreateMask .DrawMode = *long*, .MaskMode = *long*, .Smoothing = *long*, .ToleranceMode = *long*, .Gamut = *boolean,* .Threshold = *long,* .BrightnessModel = *long*

This command creates a mask from a color mask, using colors defined with a series of ColorMaskColor commands. A ColorMaskCreateMask command block must end with an EndColorMask command.

Syntax	Definition
.DrawMode	Specifies the draw mode: 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.MaskMode	Specifies the mode of the Color Mask: 0 = Protect 1 = Modify
.Smoothing	Specifies the amount of smoothing to be applied by the Color Mask. Valid values range from 0 to 100.
.ToleranceMode	Specifies the tolerance mode to set: 0 = Normal 1 = HSB 2 = Hue 3 = Saturation 4 = Brightness
.Gamut	Set to TRUE (-1) to enable gamut control.
.Threshold	Converts selected colors to black. This command uses black to display the pixels with a brightness value that is below the threshold that you set. Masked areas are always displayed in black however, black may translate differently depending on the preview method you specify. Values range from 0 to 255.
.BrightnessModel	Converts selected colors to white. This command uses white to display the pixels with a brightness value that is above the threshold you set. Selected areas on an image are always displayed in white however, this may translate differently depending on the preview method you specify. 0=Black 1=White

Example

```
.ColorMaskCreateMask 0, 1, 100, 2, FALSE, 0, 0
    .ColorMaskColor 0, 5, 0, 0, 0, 0, 18, 18, 18, 18
    .ColorMaskColor 1, 5, 0, 0, 0, 0, 20, 20, 20, 20
    .ColorMaskColor 2, 5, 0, 0, 0, 0, 20, 20, 20, 20
    .EndColorMask
```

This example creates a color mask.

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

ColorMaskReset (PHOTO-PAINT)

.ColorMaskReset

This command resets the color and tolerance settings to be used in the next call to the ColorMaskCreateChannel or ColorMaskCreateMask commands.

Example

```
.ColorMaskReset
```

This example resets the color mask settings for the next call to the ColorMaskCreateChannel or ColorMaskCreateMask commands.

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

EndColorMask (PHOTO-PAINT)

.EndColorMask

This command ends a ColorMaskCreateMask command block.

Example

```
.ColorMaskCreateMask 0, 0, 0, 0, FALSE
    .ColorMaskColor 0, 5, 245, 149, 180, 0, 10, 10, 10, 10
    .ColorMaskColor 1, 5, 245, 232, 180, 0, 10, 10, 10, 10
    .ColorMaskColor 2, 5, 245, 149, 180, 0, 10, 10, 10, 10
    .EndColorMask
```

This example creates a color mask.

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

OverprintColor (PHOTO-PAINT)

.OverprintColor .Number = *long*, .Cyan = *long*, .Magenta = *long*, .Yellow = *long*, .Black = *long*

This command sets the overprint colors for the ImageConvertDuotone command. This command must appear in an ImageConvertDuotone command block.

Syntax	Description
.Number	Specifies the index number of the overprint color.
.Cyan	Specifies the cyan color channel value of the overprint color. Valid values range from 0 to 100.
.Magenta	Specifies the magenta color channel value of the overprint color. Valid values range from 0 to 100.
.Yellow	Specifies the yellow color channel value of the overprint color. Valid values range from 0 to 100.
.Black	Specifies the black color channel value of the overprint color. Valid values range from 0 to 100.

Example

```
.ImageConvertDuotone 2, TRUE
    .OverprintColor 0, 2, 26, 97, 0
    .OverprintColor 1, 0, 100, 0, 100
    .OverprintColor 2, 100, 0, 0, 100
    ...
    .OverprintColor 10, 100, 100, 100, 100

    .DuotoneInfo 0, 3, 0, 0, 0, 255
    .DuotoneHandle 0, 0, 0, 0
    .DuotoneHandle 0, 1, 117, 66
    .DuotoneHandle 0, 2, 255, 100

    .DuotoneInfo 1, 4, 0, 0, 255, 0
    .DuotoneHandle 1, 0, 0, 0
    .DuotoneHandle 1, 1, 85, 73
    .DuotoneHandle 1, 2, 166, 40
    .DuotoneHandle 1, 3, 255, 100
    .EndConvertDuotone
```

This example converts a grayscale image to a duotone with black and yellow channels, using the ten defined overprint colors.

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

Mask channel commands (PHOTO-PAINT)

AlphaOrder (PHOTO-PAINT)

.AlphaOrder .CurrentIndex = *long*, .NewIndex = *long*

This command reorders the mask or alpha channels in the Channels Docker window.

Syntax	Description
.CurrentIndex	Specifies the position of the current mask or alpha channel. 0 is the position of the first mask or alpha channel.
.NewIndex	Specifies the new position of the mask or alpha channel.

Example

```
.AlphaOrder 2, 0
```

This example reorders the alpha channels in the Channels Docker window.

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

ChannelNew (PHOTO-PAINT)

.ChannelNew .Name = *string*, .Opacity = *long*, .Invert = *long*, .FillWhite = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command creates a new alpha channel.

Syntax	Description
.Name	Specifies the name of the channel.
.Opacity	Specifies the opacity of the mask overlay that is applied to the channel.
.Invert	Specifies whether to invert the mask overlay that is applied to the channel.
.FillWhite	Set to TRUE (-1) to fill the mask selection with white. Set to FALSE (0) to fill the mask selection with black.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.ChannelNew "Alpha1", 50, 1, 0, 5, 255, 0, 0, 0
```

This example creates an alpha channel named "Alpha1". The overlay is red, inverted, and has 50% opacity.

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

ChannelProperties (PHOTO-PAINT)

.ChannelProperties .ChannelID = *long*, .Name = *string*, .Opacity = *long*, .Invert = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command sets the properties of an alpha channel

Syntax	Description
.ChannelID	Specifies the index number of the channel
.Name	Specifies the name of the channel.
.Opacity	Specifies the opacity of the mask overlay that is applied to the channel.
.Invert	Specifies whether to invert the mask overlay that is applied to the channel.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.ChannelProperties 2, "Sample", 25, 0, 5, 153, 255, 0, 0
```

This example changes the properties of an alpha channel named "Sample".

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

MaskChannelAdd (PHOTO-PAINT)

.MaskChannelAdd .MaskName = *string*

This command saves a mask as a mask channel.

Syntax	Definition
.MaskName	Specifies the mask channel name to save.

Example

```
.MaskChannelAdd "Mask1"
```

This example saves the current mask as a channel named Mask1 .

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

MaskChannelDelete (PHOTO-PAINT)

.MaskChannelDelete .MaskID = *long*

This command deletes the specified mask channel.

Syntax	Definition
.MaskID	Identifies the mask channel to delete. Channels for the active document are listed in the Channels Roll-Up. The channels are numbered and identified according to their position in the list the first listed channel is identified as 0, the second listed channel is identified as 1, and so on.

Example

```
.MaskChannelDelete 5
```

This example deletes the sixth mask channel.

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

MaskChannelLoad (PHOTO-PAINT)

.MaskChannelLoad .FileName = *string*, .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .LoadType = *long*

This command loads a mask as a mask channel.

Syntax	Definition
.FileName	Specifies the file name containing the mask to load.
.Left	Specifies the X-coordinate of the upper-left corner of the mask in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the mask in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the mask in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-left corner of the mask in pixels, relative to the origin.
.LoadType	Specifies the image loading method: 0 = All 1 = Partial 2 = Resample 3 = Crop

Example

```
.MaskChannelLoad "PIXELATE.CPT", 0, 0, 0, 0, 0
```

This example loads the mask channel named PIXELATE.CPT.

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

MaskChannelName (PHOTO-PAINT)

.MaskChannelName .MaskID = *long*, .Name = *string*

This command sets the name of an existing mask.

Syntax	Definition
.MaskID	Identifies the mask channel. Channels for the active document are listed in the Channels Roll-Up. The channels are numbered and identified according to their position in the list the first listed channel is identified as 0, the second listed channel is identified as 1, and so on.
.Name	Specifies the new name of the mask channel.

Example

```
.MaskChannelName 5, "Mask1"
```

This example names the sixth mask channel.

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

MaskChannelSave (PHOTO-PAINT)

.MaskChannelSave .MaskID = *long*, .FileName = *string*, .FilterID = *long*, .Compression = *long*

This command saves the mask in the specified channel to a file.

Syntax	Definition
.MaskID	Identifies the mask channel. Channels for the active document are listed in the Channels Roll-Up. The channels are numbered and identified according to their position in the list the first listed channel is identified as 0, the second listed channel is identified as 1, and so on.
.FileName	Specifies the name of the file.
.FilterID	Specifies the type of file filter 769 = Windows Bitmap (BMP) 770 = Paintbrush (PCX) 771 = Targa Bitmap (TGA) 772 = TIFF Bitmap (TIF) 773 = CompuServe Bitmap (GIF) 774 = JPEG Bitmap (JPG) 776 = Scitex CT Bitmap (SCT) 787 = GEM Paint File (IMG) 792 = OS/2 Bitmap (BMP) 1289 = Encapsulated PostScript (EPS) 1536 = Video for Windows (AVI) 1792 = Corel PHOTO-PAINT Image (CPT) 1794 = Corel CMX 5.0 1793 = Corel CMX 6.0 1795 = CorelDRAW (CDR)
.Compression	Specifies the type of file compression to apply: 0 = None 1 = LZW 2 = Packbits 3 = JPEG

Example

```
.MaskChannelSave 0, "MASK2.CPT", 1792, 0
```

This example saves the mask as MASK2.CPT.

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

MaskChannelToMask (PHOTO-PAINT)

.MaskChannelToMask .MaskID = *long*, .DrawMode = *long*

This command makes the mask in the specified channel the current mask.

Syntax	Definition
.MaskID	Identifies the mask channel. Channels for the active document are listed in the Channels Roll-Up. The channels are numbered and identified according to their position in the list the first listed channel is identified as 0, the second listed channel is identified as 1, and so on.
.DrawMode 0 = Normal 1 = Add 2 = Subtract 3 = XOR	Specifies the draw mode

Example

```
.MaskChannelToMask 0, 0
```

This example makes the mask in the specified channel the current mask.

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

MaskToMaskChannel (PHOTO-PAINT)

.MaskToMaskChannel .MaskID = *long*

This command stores the active mask in an existing mask channel.

Syntax	Definition
.MaskID	Identifies the mask channel. Channels the active document are listed in the Channels Roll-Up. The channels are numbered and identified according to their position in the list the first listed channel is identified as 0, the second listed channel is identified as 1, and so on.

Example

```
.MaskToMaskChannel 0
```

This example stores the active mask in the first mask channel.

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

Mask commands (PHOTO-PAINT)

MaskAffineDistort (PHOTO-PAINT)

.MaskAffineDistort .XOffset = *long*, .YOffset = *long*, .D11 = *double*, .D12 = *double*, .D21 = *double*, .D22 = *double*, .AntiAlias = *boolean*

This command is used by Corel SCRIPT to record manual manipulation of a mask area. If you want to distort a mask in Corel SCRIPT, use the MaskDistort command.

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

MaskAlign (PHOTO-PAINT)

.MaskAlign .Horizontal = *long*, .Vertical = *long*, .Center = *boolean*, .Grid = *boolean*, .ActObj = *boolean*, .SelObj = *boolean*, .Bkgnd = *boolean*

This command aligns the current mask on an image.

Syntax	Description
.Horizontal	Specifies the type of horizontal alignment: 0 = None 1 = Left 2 = Right 3 = Center
.Vertical	Specifies the type of vertical alignment: 0 = None 1 = Top 2 = Bottom 3 = Center
.Center	Set to TRUE (-1) to align objects to the center of the page. Set to FALSE (0) to align objects to the grid, to the document, to the selected object, or to the active object.
.Grid	Set to TRUE (-1) to align objects to the grid. Set to FALSE (0) to align objects to the center of the page, to the document, to the selected object, or to the active object.
.ActiveObject	Set to TRUE (-1) to align objects to the active object in the image. Set to FALSE (0) to align objects to the center of the page, to the grid, to the selected object, or to the document.
.SelectedObject	Set to TRUE (-1) to align selected objects to the document. Set to FALSE (0) to align objects to the center of the page, to the grid, to the selected object, or to the active object.
.Background	Set to TRUE (-1) to align selected objects to the background. Set to FALSE (0) to align objects to the center of the page, to the grid, to the selected object, or to the active object.

Example

```
.MaskAlign 3, 3, FALSE, FALSE, FALSE, FALSE, TRUE
```

This example aligns the mask marquee to the center of the document.

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

MaskBorder (PHOTO-PAINT)

.MaskBorder .Width = *long*, .Edges = *long*

This command creates a new mask border along the edges of the original mask.

Syntax	Definition
.Width	Specifies the width of the mask border, in pixels.
.Edges	Specifies the type of feather: 0 = Soft 1 = Medium 2 = Hard

Example

```
.MaskBorder 10, 0
```

This example creates a mask border with a width of 10 pixels and a soft border edge.

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

MaskBrush (PHOTO-PAINT)

.MaskBrush .DrawMode = *long*, .Width = *long*, .Flatten = *long*, .Rotate = *long*, .NibShape = *long*, .Transparency = *long*, .SoftEdge = *long*

This command defines a mask using a brush stroke following a path defined by Draw commands. A MaskBrush command block must contain a series of StartDraw and ContinueDraw commands and end with an EndDraw command.

Syntax	Description
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.Width	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Flatten	Specifies the flatness of the nib. Valid values range from 0 to 100%.
.Rotate	Specifies the rotation of the nib. Valid values range from 0 to 360 degrees.
.NibShape	Specifies the shape of the nib: 0 = Round 1 = Square
.Transparency	Specifies the transparency of the nib. Valid values range from 0 to 100%.
.SoftEdge	Specifies the transparency of the nib's edges. As you increase the value of this setting, the soft edge expands to eventually reach the center of the stroke. Low values affect only the rim of the stroke. Valid values range from 0 to 100%.

Example

```
.MaskBrush 0, 8, 100, 0, 0, 0, 43, 0
    .StartDraw 16768, 11136, 0, 0
    .ContinueDraw 17082, 11540, 0, 0
     ...
    .ContinueDraw 49366, 54742, 0, 0
    .EndDraw
```

This example creates a mask outlined by the brush stroke defined by the StartDraw and ContinueDraw commands.

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

MaskCreate (PHOTO-PAINT)

.MaskCreate .PreserveImage = *boolean*, .DrawMode = *long*

This command creates a mask from the selected object(s).

Syntax	Definition
.PreserveImage	Set to TRUE (-1) to preserve the image. Set to FALSE (0) to disable this option.
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR

Example

```
.MaskCreate -1, 0
```

This example creates a mask from the selected object, preserving the image and using normal draw mode.

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

MaskCreateFromPath (PHOTO-PAINT)

.MaskCreateFromPath .AntiAlias = *boolean*, .DrawMode = *long*

This command creates a mask selection that has the shape of the current path.

Syntax	Description
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR

Example

```
.MaskCreateFromPath -1, 1
```

This example creates a mask from an existing path using the Additive mask mode.

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

MaskDeFloat (PHOTO-PAINT)

.MaskDeFloat

This command is used after the MaskFloaterMoveTo command to draw the floating mask contents on the image and convert the mask to a non-floating mask.

Example

```
.MaskDeFloat
```

This example defloats the current mask.

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

MaskDistort (PHOTO-PAINT)

.MaskDistort .Corner1X = *long*, .Corner1Y = *long*, .Corner2X = *long*, .Corner2Y = *long*, .Corner3X = *long*, .Corner3Y = *long*, .Corner4X = *long*, .Corner4Y = *long*, .AntiAlias = *boolean*

This command distorts the shape of the current mask.

Syntax	Definition
.Corner1X	Specifies the X-coordinate of the new location of the upper-left corner of the mask's original bounding box in pixels, relative to the origin.
.Corner1Y	Specifies the Y-coordinate of the new location of the upper-left corner of the mask's original bounding box in pixels, relative to the origin.
.Corner2X	Specifies the X-coordinate of the new location of the upper-right corner of the mask's original bounding box in pixels, relative to the origin.
.Corner2Y	Specifies the Y-coordinate of the new location of the upper-right corner of the mask's original bounding box in pixels, relative to the origin.
.Corner3X	Specifies the X-coordinate of the new location of the lower-right corner of the mask's original bounding box in pixels, relative to the origin.
.Corner3Y	Specifies the Y-coordinate of the new location of the lower-right corner of the mask's original bounding box in pixels, relative to the origin.
.Corner4X	Specifies the X-coordinate of the new location of the lower-left corner of the mask's original bounding box in pixels, relative to the origin.
.Corner4Y	Specifies the Y-coordinate of the new location of the lower-left corner of the mask's original bounding box in pixels, relative to the origin.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.

Example

```
.MaskDistort 100, 100, 100, 200, 300, 450, 200, 200
```

This example distorts the selected mask by the specified parameters.

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

MaskEllipse (PHOTO-PAINT)

.MaskEllipse .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .DrawMode = *long*, .Feather = *long*, .AntiAlias = *boolean*

This command creates elliptical or circular masks.

Syntax	Definition
.Left	Specifies the X-coordinate of the upper-left corner of the ellipse's mask in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the ellipse's mask in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the ellipse's mask in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the ellipse's mask in pixels, relative to the origin.
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.Feather	Specifies the feathering width in pixels. Valid values range from 0 to 100.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.

Example

```
.MaskEllipse 61, 444, 314, 203, 0, 0, 0
```

This example creates an elliptical mask with the given coordinates, with no feathering, and no anti-aliasing.

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

MaskExpand (PHOTO-PAINT)

.MaskExpand .Width = *long*

This command extends transparent areas of the mask into opaque areas by a given number of pixels.

Syntax	Definition
.Width	Specifies the amount of stretching. Valid values range from 1 to 200 pixels.

Example

```
.MaskExpand 15
```

This example expands the mask by 15 pixels.

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

MaskFeather (PHOTO-PAINT)

.MaskFeather .Width = *long*, .Direction = *long*, .Type = *long*

This command feathers the edges of the mask, creating a smoothing effect.

Syntax	Definition
.Width	Controls the number of pixels that are used in the feathering transition. A large number produces a wider transition area.
.Direction	Controls the direction of feathering 0 = Inside 1 = Middle 2 = Outside 3 = Average
.Type	Specifies the type of feather edge: 0 = Linear 1 = Curved

Example

```
.MaskFeather 10, 0, 0
```

This example uses 10 pixels in the feathering transition, an inside edge transition between a mask and the background, and a hard feather edge.

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

MaskFlipHorizontal (PHOTO-PAINT)

.MaskFlipHorizontal

This command flips the active mask horizontally.

Example

```
.MaskFlipHorizontal
```

This example flips the current mask horizontally.

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

MaskFlipVertical (PHOTO-PAINT)

.MaskFlipVertical

This command flips the active mask vertically.

Example

```
.MaskFlipVertical
```

This example flips the current mask vertically.

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

MaskFloaterMoveTo (PHOTO-PAINT)

.MaskFloaterMoveTo .Left = *long*, .Bottom = *long*, .Copy = *boolean*

This command moves the active mask and moves or copies the part of the image within the mask area.

Syntax	Definition
.Left	Specifies the new X-coordinate of the lower-left corner of the mask's bounding box, in pixels.
.Bottom	Specifies the new Y-coordinate of the lower-left corner of the mask's bounding box, in pixels.
.Copy	Set to TRUE (-1) to make a copy of the image inside the mask. Set to FALSE (0) to move the image in the mask.

Example

```
.MaskFloaterMoveTo 46, 30, 0
```

This example moves the mask up 30 pixels from the bottom edge and 46 pixels to the right from the left edge.

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

MaskFreehand (PHOTO-PAINT)

.MaskFreehand .DrawMode = *long*, .Feather = *long*, .AntiAlias = *long*

This command defines a mask using a path defined by Draw commands. A MaskFreehand command block must contain a series of StartDraw and ContinueDraw commands and end with an EndDraw command.

Syntax	Description
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.Feather	Specifies the number of pixels you want to use along the mask selection's edge to apply feathering. Feathered pixels gradually become more opaque as you get closer to the protected area of the mask so that changes applied to the selection blend gradually towards the rest of the image. Valid values range from 0 to 100.
.AntiAlias	Set to 1 to apply anti-aliasing. Set to 0 to disable anti-aliasing.

Example

```
.MaskFreehand 0, 0, 1
    .StartDraw 166, 43, 0, 0
    .ContinueDraw 165, 43, 0, 0
     ...
    .ContinueDraw 221, 99, 0, 0
    .EndDraw
```

This example creates a mask defined by the StartDraw and ContinueDraw commands.

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

MaskGrow (PHOTO-PAINT)

.MaskGrow .AntiAlias = *boolean,* .MaskVisible = *boolean,* .ToleranceMode = *long,* .Normal = *long,* .Hue = *long,* .Saturation = *long,* .Brightness = *long*

This command expands a mask to include areas of the image with similar pixel colors. The mask continues to expand until all of the adjacent colors that meet the selection criteria are included.

Syntax	Definition
.Antialias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.MaskVisible	Set to TRUE (-1) to include all visible objects. Set to FALSE (0) to include the active object only.
.ToleranceMode	Specifies the tolerance mode: 0 = Normal 1 = HSB
.Normal	Specifies the tolerance as a percentage. This parameter is used only if .ToleranceMode is set to 0.
.Hue	Specifies the hue tolerance as a percentage. In the HSB color model, hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. This parameter is used only if .ToleranceMode is set to 1.
.Saturation	Specifies the saturation tolerance as a percentage. Saturation is the purity of a color. The more colors used to mix a color, the duller the color looks. This parameter is used only if .ToleranceMode is set to 1.
.Brightness	Specifies the brightness tolerance as a percentage. In the HSB color model, the component that determines the amount of black in a color. This parameter is used only if .ToleranceMode is set to 1.

Note

This command uses the current Magic Wand Tool tolerance settings.

Example

```
.MaskGrow TRUE, FALSE, 0,10,10,10,10
```

This example expands the mask to include areas of the image with similar pixel colors.

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

MaskInvert (PHOTO-PAINT)

.MaskInvert

This command reverses the area included in the mask.

Example

```
.MaskInvert
```

This example inverts the current mask.

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

MaskLasso (PHOTO-PAINT)

.MaskLasso .DrawMode = *long*, .AntiAlias = *long*, .MaskVisible = *boolean*

This command defines a mask using a bounding area defined by Draw commands. The mask will snap to conform to the edges of the objects within the bounding area. A MaskLasso command block must contain a series of StartDraw and ContinueDraw commands and end with an EndDraw command.

Syntax	Description
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.AntiAlias	Set to 1 to apply anti-aliasing. Set to 0 to disable anti-aliasing.
.MaskVisible	Set to TRUE (-1) to include all visible objects in the selection. Set to FALSE (0) to include the active object only.

Example

```
.MaskLasso 0, TRUE, TRUE
    .ToleranceSettings 0, 10, 0, 0, 0
    .StartDraw 195, 156, 0, 0
    .ContinueDraw 194, 156, 0, 0
    ...
    .ContinueDraw 246, 217, 0, 0
    .EndDraw
```

This example creates a mask from the objects in the area defined by the StartDraw and ContinueDraw commands.

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

MaskLoad (PHOTO-PAINT)

.MaskLoad .Filename = *string*, .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .LoadType = *long*

This command loads a previously saved mask or any importable image. PHOTO-PAINT converts the imported image to a grayscale mask.

Syntax	Definition
.Filename	Specifies the name of the file.
.Left	Specifies the X-coordinate of the upper-left corner of the mask to be loaded in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the mask to be loaded in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the mask to be loaded in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the mask to be loaded in pixels, relative to the origin.
.LoadType	Specifies the image loading method: 0 = All 2 = Resample 3 = Crop

Note

A mask or image cannot be loaded as a partial file. The mask image is scaled to have the same dimensions as the current image.

Example

```
.MaskLoad "MASK1.CPT", 0, 0, 0, 0, 0
```

This example loads all of the mask named MASK1.CPT in to the upper-left corner of the image.

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

MaskMagicWand (PHOTO-PAINT)

.MaskMagicWand .ptX = *long*, .ptY = *long*, .DrawMode = *long*, .AntiAlias = *boolean*, .MaskVisible = *boolean,* .ToleranceMode = *long*, .Normal = *long*, .Hue = *long*, .Saturation = *long*, .Brightness = *long*

This command creates a contiguous mask that includes image pixels having colors within the given tolerance of the color at the starting position.

Syntax	Definition
.ptX	Specifies the X-coordinate of the position at which the Magic Wand is applied, in pixels, relative to the origin.
.ptY	Specifies the Y-coordinate of the position at which the Magic Wand is applied, in pixels, relative to the origin.
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.MaskVisible	Set to TRUE (-1) to affect all visible objects. Set to FALSE (0) to affect the active object only.
.ToleranceMode	Specifies the tolerance mode: 0 = Normal. The .Normal parameter is used to set the tolerance level 1 = HSB The HSB parameters are used to set the tolerance level
.Normal	Specifies the Normal tolerance level. Valid values range from 0 to 100%.
.Hue	Specifies the Hue tolerance. In the HSB color model, Hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. Valid values range from 0 to 100%.
.Saturation	Specifies the Saturation tolerance. Saturation is the purity of a color. The more colors used to mix a color, the duller the color looks. Valid values range from 0 to 100%.
.Brightness	Specifies the Brightness tolerance. In the HSB color model, the component that determines the amount of black in a color. Valid values range from 0 to 100%.

Example

```
.MaskMagicWand 285, 108, 0, TRUE, 0, 10, 10, 10, 10
```

This example creates a Magic Wand mask at the point (285, 108), applying anti-aliasing. The coordinates are expressed in pixels.

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

MaskPaint (PHOTO-PAINT)

.MaskPaint .Mask = *long*

This command displays a grayscale copy of the selected mask, allowing you to switch between editing an image or its mask.

Syntax	Definition
.Mask	Specifies whether to paint on the mask or its image 1 = Paint on the mask 2 = Paint on the image

Example

```
.MaskPaint 1
```

This example paints on the mask.

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

MaskRectangle (PHOTO-PAINT)

.MaskRectangle .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .DrawMode = *long*, .Feather = *long*

This command defines rectangular masks.

Syntax	Definition
.Left	Specifies the X-coordinate of the upper-left corner of the mask's rectangle in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the mask's rectangle in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the mask's rectangle in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the mask's rectangle in pixels, relative to the origin.
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.Feather	Specifies the feathering width in pixels. Valid values range from 0 to 100.

Example

```
.MaskRectangle 59, 175, 210, 91, 0, 0
```

This example creates a rectangular mask with the upper-left corner at the point (59, 175) and the lower-right corner at the point (210, 91). These coordinates are expressed in pixels.

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

MaskReduce (PHOTO-PAINT)

.MaskReduce .Width = *long*

This command extends opaque areas of the mask into transparent areas by a given number of pixels.

Syntax	Definition
.Width	Specifies the amount by which the bounding box is inset, in pixels. Valid values range from 1 to 200.

Example

```
.MaskReduce 50
```

This example reduces the width of the mask by 50 pixels

```
.MaskReduce 60
```

This example reduces the width of the mask by 60 pixels.

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

MaskRemove (PHOTO-PAINT)

.MaskRemove

This command deletes the current mask .

Example

```
.MaskRemove
```

This example deletes the current mask.

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

MaskRemoveHoles (PHOTO-PAINT)

.MaskRemoveHoles

This command removes holes in the mask.

Example

```
.MaskRemoveHoles
```

This example removes holes created by masking tools.

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

MaskRotate (PHOTO-PAINT)

.MaskRotate .XCenter = *long*, .YCenter = *long*, .Angle = *double*, .AntiAlias = *boolean*

This command rotates the current mask by the specified angle, about the given rotation point.

Syntax	Definition
.XCenter	Specifies the X-coordinate of the center of rotation in pixels, relative to the origin.
.YCenter	Specifies the Y-coordinate of the center of rotation in pixels, relative to the origin.
.Angle	Specifies the angle of the mask's rotation in tenths of degrees. Positive numbers result in counter-clockwise rotation, negative numbers result in clockwise rotation for example, 45 degrees clockwise = -450.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.

Example

```
.MaskRotate 268, 363, 450, 0
```

This example rotates the mask 45 degrees around the center point located at X,Y coordinates 268, 363.

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

MaskSave (PHOTO-PAINT)

.MaskSave .Filename = *string*, .FilterID = *long*, .Compression = *long*

This command saves the active mask to a file.

Syntax	Definition
.Filename	Specifies the path and name of the file.
.FilterID	Specifies the type of file filter 769 = Windows Bitmap (BMP) 770 = Paintbrush (PCX) 771 = Targa Bitmap (TGA) 772 = TIFF Bitmap (TIF) 773 = CompuServe Bitmap (GIF) 774 = JPEG Bitmap (JPG) 775 = Kodak Photo CD Image (PCD) 776 = Scitex CT Bitmap (SCT) 777 = Wavelet Compressed Bitmap (WVL) 787 = GEM Paint File (IMG) 784 = Windows 3.x/NT Cursor Resource (CUR) 788 = Adobe Photoshop (PSD) 785 = Windows 3.x/NT Icon Resource (ICO) 786 = Windows 3.x/NT Bitmap Resource (EXE) 790 = MACPaint Bitmap (MAC) 789 = Picture Publisher 4 (PB4) 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) 1290 = PostScript Interpreted (PS) 1291 = OS/2 PM Metafile (MET) 1294 = Windows Metafile (WMF) 1295 = Corel Metafile (CMF) 1296 = AutoCad (DXF) 1536 = Video for Windows (AVI) 1539 = CorelMOVE (CMV) 1540 = CorelSHOW (SHW) 1541 = CorelCHART (CCH) 1543 = AutoDesk FLIC (FLI) 1548 = MicroSoft PowerPoint (PPT) 1549 = Lotus Freelance (PRE) 1551 = MPEG Animation (MPG) 1792 = Corel PHOTO-PAINT Image (CTP) 1794 = Corel CMX 5.0 1793 = Corel CMX 6.0 1795 = CorelDRAW (CDR)
.Compression	Specifies the type of file compression to apply: 0 = None 1 = LZW 2 = Packbits 3 = JPEG

Example

```
.MaskSave "MASK1.CPT", 1792, 0
```

This example saves the current mask as a file named MASK1.CPT.

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

MaskScissors (PHOTO-PAINT)

.MaskScissors .DrawMode = *long*, .Feather = *long*, .AntiAlias = *boolean,* .MaskVisible = *boolean*

This command defines a straight-edged mask using a bounding area defined by Draw commands. The mask will snap to approximately conform to the edges of the objects within the bounding area. A MaskScissors command block must contain a series of StartDraw and ContinueDraw commands and end with an EndDraw command.

Syntax	Description
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.Feather	Specifies the number of pixels you want to use along the mask selection's edge to apply feathering. Feathered pixels gradually become more opaque as you get closer to the protected area of the mask so that changes applied to the selection blend gradually towards the rest of the image. Valid values range from 0 to 100.
.Antialias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.MaskVisible	Set to TRUE (-1) to include all visible objects. Set to FALSE (0) to include the active object only.

Example

```
.MaskScissors 0, 0, FALSE, FALSE
    .StartDraw 78, 69, 0, 0
    .ContinueDraw 89, 76, 0, 0
    ...
    .ContinueDraw 87, 76, 0, 0
    .EndDraw
```

This example creates a scissors mask from the active object in the area defined by the StartDraw and ContinueDraw commands.

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

MaskSelectAll (PHOTO-PAINT)

.MaskSelectAll

This command creates a mask including the entire image area.

Example

```
.MaskSelectAll
```

This example encompasses the entire active image in a mask.

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

MaskSimilar (PHOTO-PAINT)

.MaskSimilar .AntiAlias = *boolean,* .MaskVisible = *boolean,* .ToleranceMode = *long,* .Normal = *long,* .Hue = *long,* .Saturation = *long,* .Brightness = *long*

This command adds to the current mask area any areas of the image having colors similar to the image color within the current mask area.

Syntax	Definition
.Antialias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.MaskVisible	Set to TRUE (-1) to include all visible objects. Set to FALSE (0) to include the active object only.
.ToleranceMode	Specifies the tolerance mode: 0 = Normal 1 = HSB
.Normal	Specifies the tolerance as a percentage. This parameter is used only if .ToleranceMode is set to 0.
.Hue	Specifies the hue tolerance as a percentage. In the HSB color model, hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. This parameter is used only if .ToleranceMode is set to 1.
.Saturation	Specifies the saturation tolerance as a percentage. Saturation is the purity of a color. The more colors used to mix a color, the duller the color looks. This parameter is used only if .ToleranceMode is set to 1.
.Brightness	Specifies the brightness tolerance as a percentage. In the HSB color model, the component that determines the amount of black in a color. This parameter is used only if .ToleranceMode is set to 1.

Note

This command uses the current Magic Wand Tool tolerance settings.

Example

```
.MaskSimilar TRUE, FALSE, 0,10,10,10,10
```

This example expands the active mask to include all similarly masked colors.

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

MaskSkew (PHOTO-PAINT)

.MaskSkew .HorzAngle = *double*, .VertAngle = *double*, .AntiAlias = *boolean*

This command skews the current mask.

Syntax	Definition
.HorzAngle	Specifies the horizontal skew angle in tenths of degrees. Positive numbers result in counter-clockwise skew, negative numbers result in clockwise skew.
.VertAngle	Specifies the vertical skew angle tenths of degrees. Positive numbers result in counter-clockwise skew, negative numbers result in clockwise skew.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.

Example

```
.MaskSkew 3.5, 11.5, TRUE
```

This example skews the mask by 3.5 degrees horizontally and 11.5 degrees vertically.

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

MaskSmooth (PHOTO-PAINT)

.MaskSmooth .Radius = *long*

This command smoothes over or rounds off the sharp angles of a mask.

Syntax	Definition
.Radius	Specifies the radius of mask edge smoothing. Valid values range from 1 to 50.

Example

```
.MaskSmooth 5
```

This example applies a smoothing radius of 5 to the mask.

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

MaskStretch (PHOTO-PAINT)

.MaskStretch .Left = *long*, .Top = *long*, .XScale = *double*, .YScale = *double*, .AntiAlias = *boolean*

This command stretches the active mask.

Syntax	Definition
.Left	Specifies the X-coordinate of the new upper-left corner of the stretched mask in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the new upper-left corner of the stretched mask in pixels, relative to the origin.
.XScale	Specifies the degree of horizontal stretch to apply to the mask.
.YScale	Specifies the degree of vertical stretch to apply to the mask.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.

Example

```
.MaskStretch 100, 478, 436, 244, 0
```

This example stretches the mask to the new coordinates shown.

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

MaskStroke (PHOTO-PAINT)

.MaskStroke .Mode = *long*

This command ends a BrushTool, ImageSprayerTool, or Eraser command block.

Syntax	Description
.Mode	Specifies the position of the brush stroke: 0 = Middle 1 = Inside 2 = Outside

Example

```
.BrushTool 1, 0, 0, 20, 0, 40, 75, 0, 100, 100
    .BrushTextureSettings "", 0, 0, 0, 0, 1, TRUE, FALSE
    .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
    .SetPaintColor 5, 186, 159, 106, 0
    .MaskStroke 0
```

This example strokes the middle of a mask using the Brush tool.

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

MaskThreshold (PHOTO-PAINT)

.MaskThreshold .Level = *long*

This command converts grayscale to a black and white (or bi-level) mask using the specified threshold value.

Syntax	Definition
.Level	Specifies the threshold level. Values greater than or equal to the threshold are set to white, values less than the threshold are set to black. Valid threshold levels range from 1 to 255.

Example

```
.MaskThreshold 128
```

This example specifies a threshold of 128 and converts grayscale to a black and white mask using the specified value.

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

MaskTranslate (PHOTO-PAINT)

.MaskTranslate .ptX = *long*, .ptY = *long*

This command repositions the active mask.

Syntax	Definition
.ptX	Specifies the number of pixels to horizontally translate the mask. A positive value moves to the right; a negative value moves to the left.
.ptY	Specifies the number of pixels to vertically translate the mask. A positive value moves down; a negative value moves up.

Example

```
.MaskTranslate 100, 14
```

This example translates the mask 100 pixels to the right and 14 pixels down.

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

Object commands (PHOTO-PAINT)

CreateBackground (PHOTO-PAINT)

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

This command creates a background in an image that has no background.

Syntax	Description
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.CreateBackground 5, 0, 0, 0, 0
```

This example creates a black background.

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

EndObjectTagWWWURL (PHOTO-PAINT)

.EndObjectTagWWWURL

This command ends an ObjectTagWWWURL command block.

Example

```
.ObjectTagWWWURL
    .ObjectURLInfo 1, 0, "www.object1.com", "Link to www.Object1.com"
    .ObjectURLInfo 2, 1, "www.sdfsdf.com", "Link to www.sdfsdf.com"
    .ObjectURLInfo 3, 2, "www.3.com", "Link to www.3.com"
    .EndObjectTagWWWURL
```

This example assigns URL tags to three objects in the current image. The first has a rectangular bounding box, the second, an oval bounding box, and the third, a polygonal bounding box.

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

ObjectAddClipMask (PHOTO-PAINT)

.ObjectAddClipMask .Level = *long*

This command creates a clip mask that shows or hides objects in the Image Window.

Syntax	Description
.Level	Specifies the transparency of the clip mask. Set to 0 to hide the object in the Image Window. Set to 255 to show the object in the Image Window.

Example

```
.ObjectAddClipMask 255
```

This example adds a clip mask to all selected objects.

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

ObjectAddClipMaskFromMask (PHOTO-PAINT)

.ObjectAddClipMaskFromMask .Inverted = *boolean*

This command creates a clip mask on the selected objects in an image. Creating a clip mask from a regular mask hides all pixels outside the selection.

Syntax	Description
.Inverted	Set to TRUE (-1) to invert the regular mask on the image before creating the clip mask. This allows you to create a clip mask from the area on the image that was originally masked or protected. Set to FALSE (0) to create a clip mask from the current selection.

Example

```
.ObjectAddClipMaskFromMask TRUE
```

This example inverts the mask and then creates a clip mask.

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

ObjectAddClipMaskFromTransparency (PHOTO-PAINT)

.ObjectAddClipMaskFromTransparency

This command adds a clip mask to an object, using the object's shape. The clip mask is added to all selected objects.\

Example

```
.ObjectAddClipMaskFromTransparency
```

This example adds a clip mask to an object, based on the transparency values of the active object in an image.

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

ObjectAffineDistort (PHOTO-PAINT)

.ObjectAffineDistort .XOffset = *long*, .YOffset = *long*, .D11 = *double*, .D12 = *double*, .D21 = *double*, .D22 = *double*, .AntiAlias = *boolean*

This command is used by Corel SCRIPT to record manual manipulation of an object. If you want to distort an object in Corel SCRIPT, use the ObjectDistort command.

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

ObjectAlign (PHOTO-PAINT)

.ObjectAlign .Horizontal = *long*, .Vertical = *long*, .Center = *boolean*, .Grid = *boolean,* Active = *boolean,* SelToDoc = *boolean,* DistrBounds = *boolean,* HorizAlign = *boolean,* VertAlign = *boolean*

This command aligns selected objects to the grid, the page center, or to the topmost selected object.

Syntax	Definition
.Horizontal	Specifies the type of horizontal alignment: 0 = None 1 = Left 2 = Right 3 = Center 4 = Width
.Vertical	Specifies the type of vertical alignment: 0 = None 1 = Top 2 = Bottom 3 = Center 4 = Height
.Center	Set to TRUE (-1) to align objects to the center of the page. Set to FALSE (0) to align objects to the grid, to the document, or to the active object.
.Grid	Set to TRUE (-1) to align objects to the grid. Set to FALSE (0) to align objects to the center of the page, to the document, or to the active object.
.Active	Set to TRUE (-1) to align objects to the active object in the image. Set to FALSE (0) to align objects to the center of the page, to the grid, or to the document.
.SelToDoc	Set to TRUE (-1) to align selected objects to the document. Set to FALSE (0) to align objects to the center of the page, to the grid, or to the active object.
.DistrBounds	Set to TRUE (-1) to distribute objects across the boundary of the image. Otherwise, set to FALSE (0).
.HorizAlign	Set to TRUE (-1) to align objects horizontally. Set to FALSE (0) to distribute objects horizontally.
.VertAlign	Set to TRUE (-1) to align objects vertically. Set to FALSE (0) to distribute objects vertically.

Example

```
.ObjectAlign 4, 2, FALSE, FALSE, FALSE, TRUE, TRUE, FALSE, TRUE
```

This example aligns objects along the bottom, distributing them horizontally by inter-object spacing (i.e., width) across the selected object's bounding box.

```
.ObjectAlign 3, 1, TRUE, FALSE, FALSE, FALSE, TRUE, TRUE, FALSE
```

This example aligns objects along the document's vertical, center axis, distributing them vertically according to their top edges across the selected object's bounding box.

```
.ObjectAlign 3, 1, FALSE, FALSE, TRUE, FALSE, FALSE, FALSE, TRUE
```

This example aligns objects along the top edge of the active object, distributing them horizontally according to their center points (across the entire document).

```
.ObjectAlign 2, 4, FALSE, FALSE, FALSE, TRUE, TRUE, TRUE, FALSE
```

This example aligns objects along the right edge of the document, distributing them vertically according to even inter-object spacing (i.e., height) across the selected object's bounding box.

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

ObjectClip (PHOTO-PAINT)

.ObjectClip

This command clips selected objects to the current mask.

Example

```
.ObjectClip
```

This example clips the active object.

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

ObjectClipToParent (PHOTO-PAINT)

.ObjectClipToParent .ObjectID = *long*, .Clip = *boolean*

This command clips the pixels of a child object to the shape of its parent object. An object is always the parent to the objects above it in the Objects Docker window.

Syntax	Description
.ObjectID	Specifies the index number of the object to be clipped.
.Clip	Set to TRUE (-1) to clip an object to its parent object. Otherwise, set to FALSE (0).

Example

```
.ObjectClipToParent 3 TRUE
```

This example clips the third object in an image to its parent object.

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

ObjectColorTransparencyTool (PHOTO-PAINT)

.ObjectColorTransparencyTool .ToleranceMode = *long*, .ApplyToClipMask = *boolean*, .Smoothing = *long*

This command applies the color transparency to the active object.

Syntax	Description
.ToleranceMode	Specifies the tolerance mode. Normal HSB
.ApplyToClipMask	Set to TRUE (-1) to apply the transparency to the active object's clip mask. Otherwise, set to FALSE (0).
.Smoothing	Specifies a smoothing value for the transparency. Valid values range from 0 to 100.

Example

```
.ObjectColorTransparencyTool 0, TRUE, 27
    .ColorMaskColor 0, 5, 62, 155, 0, 0, 14, 30, 50, 10
    .ColorMaskColor 1, 5, 62, 155, 0, 0, 14, 30, 50, 10
    .ColorMaskColor 2, 5, 112, 102, 28, 0, 14, 30, 50, 10
    .EndObjectColorTransparencyTool
```

This example makes the selected colors transparent in the currently selected object using a clip mask.

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

ObjectCombine (PHOTO-PAINT)

.ObjectCombine

This command combines two or more selected objects into a single object.

Example

```
.ObjectCombine
```

This example combines the selected objects together.

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

ObjectCreate (PHOTO-PAINT)

.ObjectCreate .PreserveImage = *boolean*

This command creates an object using the shape and contents of the active mask; which effectively transforms a mask into an object.

Syntax	Definition
.PreserveImage	Set to TRUE (-1) to copy the image inside the mask. Set to FALSE (0) to cut the image inside the mask.

Note

This command clears the original mask.

Example

```
.MaskRectangle 95, 46, 260, 225, 0, 0
.ObjectCreate 0
```

This example creates an object from a mask.

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

ObjectCreateFromBackground (PHOTO-PAINT)

.ObjectCreateFromBackground

This command creates an object from the image background. The background becomes an object in the image and is given an object name in the Objects Docker window. After you convert the background to an object, you can transform it with the transformation handles, move it, and edit it in a number of ways you were not able to before.

Example

```
.ObjectCreateFromBackground
```

This example creates an object from the image background.

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

ObjectDefringe (PHOTO-PAINT)

.ObjectDefringe .Amount = *long*

This command spreads the interior colors of the selected objects out to their edges.

Syntax	Definition
.Amount	Specifies the width of the Defringe effect, expressed in pixels. Valid values range from 1 to 100.

Example

```
.ObjectDefringe 10
```

This example applies a Defringe to the object with a width of 10 pixels.

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

ObjectDelete (PHOTO-PAINT)

.ObjectDelete

This command deletes the selected object(s).

Example

```
.ObjectDelete
```

This example deletes the selected object(s).

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

ObjectDistort (PHOTO-PAINT)

.ObjectDistort .Corner1X = *long*, .Corner1Y = *long*, .Corner2X = *long*, .Corner2Y = *long*, .Corner3X = *long*, .Corner3Y = *long*, .Corner4X = *long*, .Corner4Y = *long*, .AntiAlias = *boolean*

This command distorts the shape of the selected object.

Syntax	Definition
.Corner1X	Specifies the X-coordinate of the new location of the upper-left corner of the object's original bounding box in pixels, relative to the origin.
.Corner1Y	Specifies the Y-coordinate of the new location of the upper-left corner of the object's original bounding box in pixels, relative to the origin.
.Corner2X	Specifies the X-coordinate of the new location of the upper-right corner of the object's original bounding box in pixels, relative to the origin.
.Corner2Y	Specifies the Y-coordinate of the new location of the upper-right corner of the object's original bounding box in pixels, relative to the origin.
.Corner3X	Specifies the X-coordinate of the new location of the lower-right corner of the object's original bounding box in pixels, relative to the origin.
.Corner3Y	Specifies the Y-coordinate of the new location of the lower-right corner of the object's original bounding box in pixels, relative to the origin.
.Corner4X	Specifies the X-coordinate of the new location of the lower-left corner of the object's original bounding box in pixels, relative to the origin.
.Corner4Y	Specifies the Y-coordinate of the new location of the lower-left corner of the object's original bounding box in pixels, relative to the origin.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.

Example

```
.ObjectDistort 47, 180, 187, 141, 203, 15, 47, 15, 0
```

This example distorts the selected object by the specified parameters.

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

ObjectDropShadow (PHOTO-PAINT)

.ObjectDropShadow .Mode = *boolean,* .Direction = *long*, .Distance = *long,* .Feather = *long*, .Type = *long*, .Edges = *long*, .Opacity = *long*, .Relative = *boolean,* .LightAngle = *long,* .Fade = *long,* .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command creates an object that has the same shape as the selected object and places it behind the original, producing a drop shadow effect. Optionally, you can use the Color parameters to alter the shadow's color.

Syntax	Description
.Mode	Set to TRUE (-1) to create a flat drop shadow. Set to FALSE (0) to add perspective to the drop shadow that you are creating.
.Direction	Specifies the position of the shadow relative to the original object. Values range from 0 to 360, where 0 is E and moving counterclockwise increases the angle.
.Distance	Specifies the horizontal and vertical distance between the edge of the original object and the outside edge of the shadow object.
.Feather	the width, in pixels, of the shadow's feathered edge. A feathered edge makes the shadow object blend gradually from its color to the image background. This makes the shadow object's edges less noticeable. Valid values range from 0 (sharpest) to 100.
.Type	Specifies the location of the feathered pixels, relative to the shadow object 0 = Inside. Places the feathered portion inside the shadow, along its edges 1 = Middle. Places approximately as many feathered pixels inside the edge as outside 2 = Outside. Adds pixels just outside the shadow's edges 3 = Average. Samples all pixels in the defined width and assigns a color value to each one individually. This results in some pixels being inside, some outside, and creates a more gradual transition in color between the shadow object and the background, much like a gradient.
.Edges	Specifies the edge type for the feathered portion of the shadow object 0 = Linear. Uses the sharp bends in the object to produce the feathered section 1 = Curved. Rounds the sharp edges.
.Opacity	Specifies the shadow object's opacity. Zero is completely transparent, 100 is completely opaque.
.Relative	Set to TRUE (-1) to measure the drop shadow's offset and feather width in percent. Set to FALSE (0) to measure the drop shadow's offset and feather width in the default units of measurement.
.LightAngle	Specifies the angle of the light source used to add perspective to the object's drop shadow. The angle of the light source is measured in degrees. Values range from 0 to 90.
Fade	Determines the percent by which the drop shadow fades as it extends away from the object.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.ObjectDropShadow FALSE, 0, 23, 30, 3, 1, 100, TRUE, 0, 0, 5, 255, 51, 0, 0
```

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

ObjectDuplicate (PHOTO-PAINT)

.ObjectDuplicate

This command duplicates the selected object(s). The new objects are placed in front of existing objects and remain selected. The original objects are deselected.

Example

```
.ObjectDuplicate
```

This example creates a duplicate of the selected object.

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

ObjectEdit (PHOTO-PAINT)

.ObjectEdit .ObjectID = *long*, .Edit = *boolean*

This command makes the specified object editable or uneditable.

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.
.Edit	Set to TRUE(-1) to make the object editable. Set to FALSE (0) to make the object uneditable.

Note

Making an object editable also makes it visible if it is hidden.

Example

```
.ObjectEdit 3, -1
```

This example makes the third object editable.

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

ObjectEditAll (PHOTO-PAINT)

.ObjectEditAll

This command makes all objects editable, including the background.

Example

```
.ObjectEditAll
```

This example makes all objects editable including the background.

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

ObjectEditNone (PHOTO-PAINT)

.ObjectEditNone

This command deselects the current selection.

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

ObjectEditSelected (PHOTO-PAINT)

.ObjectEditSelected

This command allows you to perform editing commands on all selected objects.

Example

```
.ObjectEditSelected
```

This example makes all selected objects editable.

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

ObjectEditTransparency (PHOTO-PAINT)

.ObjectEditTransparency .Mode = *long*

This command locks and unlocks an object's transparency.

Syntax	Description
.Mode	Set to 0 to lock the object's transparency. Set to 1 to unlock an object's transparency.

Example

```
.ObjectEditTransparency 0
```

This example locks object transparency.

```
.ObjectEditTransparency 1
```

This example unlocks object transparency.

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

ObjectFeather (PHOTO-PAINT)

.ObjectFeather .Width = *long*, .Type = *long*

This command feathers the edges of the selected object(s). Feathering is the blending of the edge of an object with an underlying image.

Syntax	Definition
.Width	Specifies the number of pixels that are used in the feathering transition. Valid values range from 1 to 200.
.Type	Specifies the intensity of the feathered edge. A soft edge produces a more gradual blending between the object and the background than a hard edge 0 = Linear 1 = Curved

Example

```
.ObjectFeather 10, 0
```

This example sets the feathering width to 10 pixels, applying a hard edge.

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

ObjectFlipHorizontal (PHOTO-PAINT)

.ObjectFlipHorizontal

This command flips the selected object(s) horizontally.

Example

```
.ObjectFlipHorizontal
```

This example flips the selected object horizontally.

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

ObjectFlipVertical (PHOTO-PAINT)

.ObjectFlipVertical

This command flips the selected object(s) vertically.

Example

```
.ObjectFlipVertical
```

This example flips the selected object vertically.

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

ObjectGroup (PHOTO-PAINT)

.ObjectGroup

This command groups all selected objects together so that they can be selected and manipulated as a single object.

Example

```
.ObjectGroup
```

This example groups selected objects together.

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

ObjectMerge (PHOTO-PAINT)

.ObjectMerge .All = *boolean*

This command merges the selected object(s) or all objects with the background. The object(s) become(s) part of the image and cannot be selected again.

Syntax	Definition
.All	Set to TRUE (-1) to merge all objects. Set to FALSE (0) to merge selected objects only.

Example

```
.ObjectMerge -1
```

This example merges all objects.

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

ObjectMergeMode (PHOTO-PAINT)

.ObjectMergeMode .MergeMode = *long*

This command sets the merge mode of the selected object(s).

Syntax	Definition
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black

Note

The modes that can be used depend on the image type.

Example

```
.ObjectMergeMode 11
```

This example sets the object merge mode to Saturation.

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

ObjectName (PHOTO-PAINT)

.ObjectName .ObjectID = *long*, .Name = *string*

This command sets the name of the specified object.

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.
.Name	Specifies the new name of the object.

Example

```
.ObjectName 1, "Capricorn"
```

This example names the second object "Capricorn".

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

ObjectNew (PHOTO-PAINT)

.ObjectNew .Name = *string,* .Opacity = *long,* .MergeMode = *long*

This command creates a new object.

Syntax	Definition	
.Name	Specifies the name of the object you are creating.	
.Opacity	Specifies the object's opacity. Valid values range from 0 to 100.	
.MergeMode	.MergeMode 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation	Specifies the Merge Mode: 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black

Example

```
ObjectNew "Object1", 100, 0
```

This example creates an object with 100% opacity using the Normal merge mode.

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

ObjectOpacity (PHOTO-PAINT)

.ObjectOpacity .Amount = *long*

This command sets the opacity of the selected object(s).

Syntax	Definition
.Amount	Specifies the opacity. Valid values range from 0 to 99 percent.

Example

```
.ObjectOpacity 50
```

This example sets the object's opacity to 50%.

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

ObjectOrder (PHOTO-PAINT)

.ObjectOrder .Order = *long*

This command controls the stacking order of selected object(s) in the image.

Syntax	Definition
.Order	Specifies the stacking order of selected object(s): 0 = To front 1 = To back 2 = Forward 3 = Back 4 = Reverse

Note

If multiple objects are selected, they retain their relative order (except in the case of reverse ordering which reverses the order of the selected objects).

For reverse ordering, at least two objects must be selected.

Example

```
.ObjectOrder 0
```

This example orders the selected object to the front of the document

```
.ObjectOrder 3
```

This example orders the selected object back one.

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

ObjectOrderChange (PHOTO-PAINT)

.ObjectOrderChange .Source = *long*, .Destination = *long*

This command alters the order of the objects designated by the Source and Destination IDs.

Syntax	Description
.Source	The Object ID of the object to be moved.
.Destination	The final position of the Source object.

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

ObjectProperties (PHOTO-PAINT)

.ObjectProperties .ObjectID = *long*, .Name = *string*, .Opacity = *long*, .MergeMode = *long*, .Visible = *boolean*, .Clip = *boolean*, .Enable = *boolean*, .Link = *boolean*

This command changes an object's properties.

Syntax	Description
.ObjectID	Specifies the index number of the object.
.Name	Specifies the name of the object.
.Opacity	Specifies the opacity of the object.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.Visible	Set to TRUE (-1) to make the object visible. Set to FALSE (0) to make the object invisible.
.Clip	Set to TRUE (-1) to clip the object to its parent object. Otherwise, set to FALSE (0).
.Enable	Set to TRUE (-1) to enable the object's clip mask. Set to FALSE (0) to disable the object's clip mask.
.Link	Set to TRUE (-1) to link the object to its clip mask. Otherwise, set to FALSE (0).

Example

```
.ObjectProperties 1, "Object 1", 100, 0, TRUE, FALSE, FALSE, FALSE
```

This example changes the properties of an object named "Object 1.

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

ObjectRemoveClipMask (PHOTO-PAINT)

.ObjectRemoveClipMask .Apply = *boolean*

This command removes a clip mask from the selected objects.

Syntax	Description
.Apply	Set to TRUE (-1) to combine the clip mask with the object before removing it. Set to FALSE (0) to remove the clip mask without applying it to the object.

Example

```
.ObjectRemoveClipMask FALSE
```

This example removes the clip mask without applying it.

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

ObjectRemoveMatte (PHOTO-PAINT)

.ObjectRemoveMatte .White = *boolean*

This command removes the black or white matte (border pixels) sometimes found along the edges of an object when an object is cut away from a black or white background.

Syntax	Definition
.White	Set to TRUE (-1) to remove white matting. Set to FALSE (0) to remove black matting.

Example

```
.ObjectRemoveMatte -1
```

This example removes the black or white matting from an object.

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

ObjectRotate (PHOTO-PAINT)

.ObjectRotate .XCenter = *long*, .YCenter = *long*, .Angle = *double*, .AntiAlias = *boolean*, .Copy = *boolean*

This command rotates the selected object(s) by the specified angle, about the given rotation point.

Syntax	Definition
.XCenter	Specifies the X-coordinate of the center of rotation in pixels, relative to the origin.
.YCenter	Specifies the Y-coordinate of the center of rotation in pixels, relative to the origin.
.Angle	Specifies the angle of rotation in tenths of degrees, relative to the center of the object. Positive numbers result in counter-clockwise rotation, negative numbers result in clockwise rotation e.g., 45 degrees clockwise = -45.0
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.Copy	Set to TRUE (-1) to apply the rotate command to a copy of the object. Set to FALSE (0) to apply the command to the original object without making a copy.

Example

```
.ObjectRotate 180, 252, -45.0, -1
```

This example rotates the selected object 45 degrees clockwise relative to the object's center, applying anti-aliasing.

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

ObjectSelect (PHOTO-PAINT)

.ObjectSelect .ObjectID = *long*, .Selected = *boolean*

This command selects or deselects the specified object.

Syntax	Definition
.ObjectID	Specifies the object. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.
.Selected	Set to TRUE (-1) to select the object. Set to FALSE (0) to deselect the object.

Note

Selecting an object makes it visible and editable. Selecting or deselecting an object in a group selects or deselects all objects in the group.

Example

```
.ObjectSelect 2, -1
```

This example selects object number 2.

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

ObjectSelectAll (PHOTO-PAINT)

.ObjectSelectAll

This command selects all objects.

Example

```
.ObjectSelectAll
```

This example selects all objects.

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

ObjectSelectNone (PHOTO-PAINT)

.ObjectSelectNone

This command deselects any selected objects.

Example

```
.ObjectSelectNone
```

This example deselects all selected objects.

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

ObjectSkew (PHOTO-PAINT)

.ObjectSkew .HorzAngle = *double*, .VertAngle = *double*, .AntiAlias = *boolean*, .Copy = *boolean*

This command skews the selected object(s).

Syntax	Definition
.HorzAngle	Specifies the horizontal skew angle in tenths of degrees. Positive numbers result in counter-clockwise skew, negative numbers result in clockwise skew.
.VertAngle	Specifies the vertical skew angle in tenths of degrees. Positive numbers result in counter-clockwise skew, negative numbers result in clockwise skew.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.Copy	Set to TRUE (-1) to apply the skew command to a copy of the object. Set to FALSE (0) to apply the command to the original object without making a copy.

Example

```
.ObjectSkew -300, -300, TRUE, TRUE
```

This example skews the selected object by 30 degrees horizontally and 30 degrees vertically.

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

ObjectStretch (PHOTO-PAINT)

.ObjectStretch .Left = *long*, .Top = *long*, .XScale = *double*, .YScale = *double*, .AntiAlias = *boolean*, .Copy = *boolean*

This command stretches the selected object(s).

Syntax	Definition
.Left	Specifies the X-coordinate of the new upper-left corner of the object's bounding box in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the new upper-left corner of the object's bounding box in pixels, relative to the origin.
.XScale	Specifies the degree of horizontal stretch to apply to the object.
.YScale	Specifies the degree of vertical stretch to apply to the object.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.Copy	Set to TRUE (-1) to apply the stretch command to a copy of the object. Set to FALSE (0) to apply the command to the original object without making a copy.

Example

```
.ObjectStretch 51, 461, 282, 151, -1
```

This example stretches the bottom edge of the object to the new specified coordinate, applying anti-aliasing.

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

ObjectTagWWWURL (PHOTO-PAINT)

.ObjectTagWWWURL

This command assigns a URL tag to objects in your image. An ObjectTagWWWURL command block must contain one or more ObjectURLInfo commands, and end with an EndObjectTagWWWURL command.

Example

```
.ObjectTagWWWURL
    .ObjectURLInfo 1, 0, "www.object1.com", "Link to www.Object1.com"
    .ObjectURLInfo 2, 1, "www.sdfsdf.com", "Link to www.sdfsdf.com"
    .ObjectURLInfo 3, 2, "www.3.com", "Link to www.3.com"
    .EndObjectTagWWWURL
```

This example assigns URL tags to three objects in the current image. The first has a rectangular bounding box, the second, an oval bounding box, and the third, a polygonal bounding box.

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

ObjectThreshold (PHOTO-PAINT)

.ObjectThreshold .Threshold = *long*

This command alters the edges of mask selections or objects, and is accessible from both the Mask and Object menus. It is used to remove the smooth transition between a selection, or an object, and the underlying image. Such transitions are created by processes such as feathering. Setting a threshold value results in crisp and obvious edges for the current mask selection or selected object.

The object or mask marquee will be placed along the pixels in the feathered portion that have the grayscale value specified. The grayscale value of the pixels located on either side of the marquee are assigned a value of 0 or 255. This makes them part of the selection or object, or excludes them. A low threshold value includes more pixels in the selection or object than a high value.

Syntax	Description
.Threshold	Specifies the grayscale value of the pixels you want the mask selection edge, or object edge, to be located on. Valid values range from 0 (black) to 255 (white).

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

ObjectToggleClipMask (PHOTO-PAINT)

.ObjectToggleClipMask .Clip = *boolean*

This command enables or disables a clip mask.

Syntax	Description
.Clip	Set to TRUE (-1) to enable the clip mask. Set to FALSE (0) to disable the clip mask again.

Example

```
.ObjectToggleClipMask FALSE
```

This example disables the clip mask.

```
.ObjectToggleClipMask TRUE
```

This example enables the clip mask.

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

ObjectToggleLinkClipMask (PHOTO-PAINT)

.ObjectToggleLinkClipMask .ObjectID = *long*, .Link = *boolean*

This command specifies whether the object and its associated clip mask are linked together.

Syntax	Description
.ObjectID	Specifies the object to which the clip mask will be linked.
.Link	Set to TRUE (-1) to link the object and its associated clip mask. Otherwise, set to FALSE (0).

Example

```
.ObjectToggleLinkClipMask 3 TRUE
```

This example links the object and its associated clip mask.

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

ObjectTranslate (PHOTO-PAINT)

.ObjectTranslate .ptX = *long*, .ptY = *long*, .Copy = *boolean*

This command repositions selected object(s).

Syntax	Definition
.ptX	Specifies the number of pixels to horizontally translate the object(s) or their copies. A positive value moves or copies to the right; a negative value moves or copies to the left.
.ptY	Specifies the number of pixels to vertically translate the object(s) or their copies. A positive value moves or copies downwards; a negative value moves or copies upwards.
.Copy	Set to TRUE (-1) to copy the object(s). Set to FALSE (0) to translate the object(s).

Note

If the objects are copied, the copies remain selected and the originals are de-selected when the command complete execution.

Example

```
.ObjectTranslate 15, -10, 0
```

This example moves the selected object(s) 15 pixels to the right and 10 pixels up.

```
.ObjectTranslate 103, 25, -1
```

This example copies the selected object(s), and places the copied object(s) 103 pixels to the right of and 25 pixels below the selected object(s)'s position.

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

ObjectTransparencyTool (PHOTO-PAINT)

.ObjectTransparencyTool .Start = *long*, .End = *long*, .UseOriginal = *boolean,* .Handles = *long,* .ApplyToClipMask = *boolean*

This command changes the transparency of objects in your image.

Syntax	Description
.Start	Specifies the transparency value for the start point of the object's transparency blend with the rest of the image. Valid values range from 0 to 100%.
.End	Specifies the transparency value for the end point of the object's transparency blend with the rest of the image. Valid values range from 0 to 100%.
.UseOriginal	Set to TRUE (-1) to add the transparency value you set to the transparency value of the pixels in the object. Set to FALSE (0) to replace the existing transparency values of the object's pixels with the values set for this tool.
.Handles	Specifies the number of handles used to apply the transparency.
.ApplyToClipMask	Set to TRUE (-1) to apply the transparency to the object's clip mask. Otherwise, set to FALSE (0).

Example

```
.ObjectTransparencyTool 0, 100, TRUE, 4, FALSE
        .GradientPoint 0, -17, 112, 9, 0, 0, 0, 0, 255
        .GradientPoint 1, -22, 113, 9, 167, 0, 0, 0, 88
        .GradientPoint 2, -28, 115, 9, 76, 0, 0, 0, 179
        .GradientPoint 3, -33, 115, 9, 255, 0, 0, 0, 0
        .Gradient 2, 50
```

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

ObjectUngroup (PHOTO-PAINT)

.ObjectUngroup

This command breaks up the selected group of objects into its component objects.

Example

```
.ObjectUngroup
```

This example reverses the ObjectGroup command, and ungroups the grouped objects.

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

ObjectURLInfo (PHOTO-PAINT)

.ObjectURLInfo .ObjectID = *long*, .Region = *long*, .Address = *string*, .Comments = *string*

This command sets the URL attributes for the ObjectTagWWWURL command. This command must appear inside an ObjectTagWWWURL command block.

Syntax	Description
.ObjectID	Specifies the index number of the object to be tagged.
.Region	Specifies the type of bounding region: 0 = Rectangular 1 = Oval 2 = Polygonal 3 = Circular
.Address	Specifies the address string of the URL.
.Comments	Specifies the comments that appear on the command line when you place your cursor over the URL bounding region.

Example

```
.ObjectTagWWWURL
    .ObjectURLInfo 1, 0, "www.object1.com", "Link to www.Object1.com"
    .ObjectURLInfo 2, 1, "www.sdfsdf.com", "Link to www.sdfsdf.com"
    .ObjectURLInfo 3, 2, "www.3.com", "Link to www.3.com"
    .EndObjectTagWWWURL
```

This example assigns URL tags to three objects in the current image. The first has a rectangular bounding box, the second, an oval bounding box, and the third, a polygonal bounding box.

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

ObjectVisible (PHOTO-PAINT)

.ObjectVisible .StartIndex = *long*, .EndIndex = *long,* .Visible = *boolean*

This command hides or shows the specified object.

Syntax	Definition
.StartIndex	Specifies the first object in the range. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.
.EndIndex	Specifies the last object in the range. Objects are numbered and identified according to their image order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.
.Visible	Set to TRUE (-1) to make object visible. Set to FALSE (0) to hide object.

Note

Hiding an object deselects it (if it was selected) and makes it uneditable.

Example

```
.ObjectVisible 1, 4, TRUE
```

This example makes objects 1 to 4 visible.

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

.LensNew (PHOTO-PAINT)

.LensNew .Name=*string*

This command creates a new lens object that covers the entire image and to which an ImageEffect command is applied. All parameters assigned to the ImageEffect command are applied to the lens object instead of being applied to the image pixels themselves.

Syntax	Definition
.Name	Determines the name to assign to the lens. This name is displayed in the Objects Roll-Up next to the lens's thumbnail.

Example

```
.ImageBCI 13, -22, 11
    .LensNew "B-C-I Lens 1"
    .EndColorEffect
```

This example creates a Brightness/Contrast/Intensity lens that covers the entire image and that has the name "B-C-I Lens 1" displayed in the Objects Roll-Up.

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

.LensCreateFromMask (PHOTO-PAINT)

.LensCreateFromMask .Name=*string*

This command creates a new lens by converting the mask selection displayed in the Image Window into a lens. This command must be preceeded by an ImageEffect command. All parameters assigned to the ImageEffect command are applied to the lens object instead of being applied to the image pixels themselves.

Syntax	Definition
.Name	Determines the name to assign to the lens. This name is displayed in the Objects Roll-Up next to the lens's thumbnail.

Example

```
.ImageHSL 20, -20, 30
    .LensCreateFromMask "HSL Lens 2"
    .EndColorEffect
```

This example converts the current mask selection to a Hue/Saturation/Lightness lens "HSL Lens 2" displayed in the Objects Roll-Up.

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

.LensEdit (PHOTO-PAINT)

.LensEdit .ObjectID=*long,* Name=*string*

This command changes the properties of an existing lens. You can also use this command to change the ImageEffect command used in the lens. This command must be preceeded by an ImageEffect command which determines the effect and properties to assign to the existing lens you decide to edit.

Syntax	Definition
.ObjectID	Specifies the lens object. All objects are numbered and identified according to their position in the stacking order (from back to front) the first object is identified as 0, the second from the back is identified as 1, and so on. The background is considered an object, and because no other object can be placed below the background, it is always identified as 0.

.Name Determines the name to assign to the lens. This name is displayed in the Objects Roll-Up next to the lens's thumbnail. You can keep the current name of the lens or change it as you change its properties.

Example

```
.ImageLevelThreshold 0, 179,255, 255, 0
    .LensEdit 1, "Threshold Lens 3"
    .EndColorEffect
```

This example modifies the lens that is just above the image background and makes that lens use the ImageLevelThreshold command and changes the lens's name to "Threshold Lens 3".

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

Movie commands (PHOTO-PAINT)

MovieBackOne (PHOTO-PAINT)

.MovieBackOne

This command backs up the movie one frame and displays it in the main image window.

Example

```
.MovieBackOne
```

This example moves the movie back by one frame.

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

MovieCreate (PHOTO-PAINT)

.MovieCreate

This command creates a new movie document using the current image.

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

MovieDeleteFrame (PHOTO-PAINT)

.MovieDeleteFrame .FromFrame = *long*, .ToFrame = *long*

This command deletes the specified number of frames from a movie.

Syntax	Definition
.FromFrame	Specifies the first frame to be deleted from the movie.
.ToFrame	Specifies the last frame to be deleted from the movie.

Example

```
.MovieDeleteFrame 7, 9
```

This example deletes movie frames number 7, 8, and 9.

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

MovieForward (PHOTO-PAINT)

.MovieForward

This command advances the movie to its final frame and displays it in the main image window.

Example

```
.MovieForward
```

This example advances the movie to the last frame.

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

MovieForwardOne (PHOTO-PAINT)

.MovieForwardOne

This command advances the movie one frame and displays it in the main image window.

Example

```
.MovieForwardOne
```

This example advances the movie by one frame.

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

MovieFrameDelay (PHOTO-PAINT)

.MovieFrameDelay .StartFrame = *long*, .EndFrame = *long*, .Delay = *long*

This command specifies the delay, in milliseconds from the first frame in a movie to the last.

Syntax	Description
.StartFrame	Specifies the first frame the sequence.
.EndFrame	Specifies the last frame in the sequence.
.Delay	Specifies the delay in milliseconds.

Example

```
.MovieFrameRate 25
.MovieFrameDelay 1, 5, 200
.MovieFrameDelay 6, 10, 300
.MovieFrameDelay 11, 25, 200
.EndMovieFrameRate
```

This example sets the frame rate for each frame in the movie. Frames 1 to 5 are set to 200 milliseconds. Frames 6 to 10 are set to 300 milliseconds. Frames 11 to 25 are set to 200 milliseconds.

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

MovieFrameRate (PHOTO-PAINT)

.MovieFrameRate .Frames = *long*

This command sets the frame delay, in milliseconds, for each specified movie frame. Frame rates are run-length encoded, which means that you can specify the delay for each frame or a range of frames in the movie.

Syntax	Description
.Frames	Specifies the number of frames to change.

Example

```
.MovieFrameRate 25
.MovieFrameDelay 1, 5, 200
.MovieFrameDelay 6, 10, 300
.MovieFrameDelay 11, 25, 200
.EndMovieFrameRate
```

This example sets the frame rate for each frame in the movie. Frames 1 to 5 are set to 200 milliseconds. Frames 6 to 10 are set to 300 milliseconds. Frames 11 to 25 are set to 200 milliseconds.

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

EndMovieFrameRate (PHOTO-PAINT)

.EndMovieFrameRate

This command ends the movie frame rate sequence.

Example

```
.MovieFrameRate 25
.MovieFrameDelay 1, 5, 200
.MovieFrameDelay 6, 10, 300
.MovieFrameDelay 11, 25, 200
.EndMovieFrameRate
```

This example sets the frame rate for each frame in the movie. Frames 1 to 5 are set to 200 milliseconds. Frames 6 to 10 are set to 300 milliseconds. Frames 11 to 25 are set to 200 milliseconds.

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

MovieGotoFrame (PHOTO-PAINT)

.MovieGotoFrame .Frame = *long*

This command lets you go to a particular movie frame. The selected frame is immediately displayed in the main image window.

Syntax	Definition
.Frame	Specifies the frame number to which you want to advance.

Example

```
.MovieGotoFrame 5
```

This example makes movie frame number 5 the active image

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

MovieInsertFile (PHOTO-PAINT)

.MovieInsertFile .FileName = *string*, .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*, .LoadType = *long*, .StartFrame = *long*, .Before = *boolean*

This command inserts any image file into the active movie file.

Syntax	Definition
.FileName	Specifies the name of the file to be inserted.
.Left	Specifies the X-coordinate of the upper-left corner of the file to be inserted in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the file to be inserted in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the file to be inserted in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the file to be inserted in pixels, relative to the origin.
.LoadType	Specifies the image loading method: 0 = All 2 = Resample 3 = Crop
.StartFrame	Specifies the insertion point of a frame or series of frames.
.Before	Set to TRUE (-1) to insert the selected frame(s) before the Start frame. Set to FALSE (0) to insert the selected frame(s) after the Start frame.

Note

The loaded image will be scaled to the movie frame dimensions.

Example

```
.FileNew 640, 480, 1, 72, 72, 0, -1, 10, 0, 0, 0, 0, 255, 255, 255, 0
.MovieInsertFile "TILES\TOWELF.PCX", 0, 0, 0, 0, 0, 1, 0
```

This example inserts the file named TOWELF.PCX, load type of all, after frame number 1.

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

MovieInsertFrame (PHOTO-PAINT)

.MovieInsertFrame .StartFrame = *long*, .NumberOfFrames = *long*, .Before = *boolean*, .CopyCurrent = *boolean*

This command inserts one or more frames into a movie.

Syntax	Definition
.StartFrame	Specifies the insertion point of a frame or series of frames.
.NumberOfFrames	Specifies the number of frames to be inserted at the designated insertion point.
.Before	Set to TRUE (-1) to insert the selected frame(s) before the Start frame. Set to FALSE (0) to insert the selected frame(s) after the Start frame.
.CopyCurrent	Set to TRUE (-1) to copy the current frame. Set to FALSE (0) to use the paper color .

Example

```
.MovieInsertFrame 5, 6, 0, 0
```

This example inserts 6 movie frames after movie frame number 5, using the paper color

```
.MovieInsertFrame 2, 8, -1, -1
```

This example inserts 8 movie frames before movie frame number 2, copying the current frame.

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

MovieMoveFrame (PHOTO-PAINT)

.MovieMoveFrame .FromFrame = *long*, .ToFrame = *long*, .MoveToFrame = *long*, .Before = *boolean*

This command rearranges frames in a movie. You can move single or multiple frames to any point in the sequence of the movie.

Syntax	Definition
.FromFrame	Specifies the first frame included in the frame move.
.ToFrame	Specifies the last frame included in the frame move.
.MoveToFrame	Specifies the insertion point of the frame move. The frames selected for the move will be inserted before or after this frame depending on the status of the Before parameter.
.Before	Set to TRUE (-1) to move the selected frame(s) before the insertion point. Set to FALSE (0) to move the selected frame(s) after the insertion point.

Example

```
.MovieMoveFrame 5, 12, 1, 0
```

This example moves frames number 5 through 12 and repositions them after frame number 1

```
.MovieMoveFrame 4, 6, 8, -1
```

This example moves frames number 4, 5, and 6 and repositions them before frame number 8.

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

MovieRewind (PHOTO-PAINT)

.MovieRewind

This command rewinds the movie to the first frame and displays it in the main image window.

Example

```
.MovieRewind
```

This example rewinds the movie back to the beginning.

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

MovieSelectPartial (PHOTO-PAINT)

.MovieSelectPartial .StartFrame = *long*, .EndFrame = *long*

This command loads selected frames from a movie file.

Syntax	Description
.StartFrame	Specifies the first frame to load.
.EndFrame	Specifies the last frame in the sequence.

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

Drawing commands (PHOTO-PAINT)

ContinueDraw (PHOTO-PAINT)

.ContinueDraw .ptX = *long*, .ptY = *long*, .Timer = *long*, .Pressure = *long,* .Tilt = *long,* .Rotate = *long*

This command continues drawing operations started with the .StartDraw command such as drawing, masking, and erasing.

Syntax	Definition
.ptX	Specifies the X-coordinate of the point to continue drawing. For Brush, Effects and Clone tools, this value is expressed in 1/256 of a pixel, relative to the origin. For other tools, this value is expressed in pixels, relative to the origin.
.ptY	Specifies the Y-coordinate of the point to continue drawing. For Brush, Effects and Clone tool, this value is expressed in 1/256 of a pixel, relative to the origin. For other tools, this value is expressed in pixels, relative to the origin.
.Timer	Used for tools like the airbrush, where the nib size changes with time. Values start at 0 and increment by 1 for every clock cycle.
.Pressure	Used for a Pen Tablet that has pressure sensitivity. Values range from 0 to 9999.
.Tilt	Specifies the tilt attribute of the clone brush tool. Values range from 0-90.
.Rotate	Specifies the rotation angle of the clone brush tool. Values range from 0-360.

Example

```
.ImageSprayerTool "fruit.cpt", 1, 5, 87, 0, 0
    .ImageSprayerSettings 1, 11, 19, 0, 0, 1, 5, 1
    .RandomSeed 552585248
    .StartDraw 11904, 8320, 0, 0
    .ContinueDraw 14343, 9096, 0, 0
    ...
    .ContinueDraw 61761, 18445, 0, 0
    .EndDraw
```

This example draws a brush stroke containing a variety of fruits.

```
.Eraser 20, 100, 0, 0, 0, 0, 0
    .SetPaperColor 5, 255, 255, 255, 0
    .StartDraw 40320, 8832, 0, 0
    .ContinueDraw 39040, 8832, 0, 0
    ...
    .ContinueDraw 59678, 18166, 0, 0
    .EndDraw
```

This example erases a portion of your image defined by the draw path.

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

Ellipse (PHOTO-PAINT)

.Ellipse .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*

This command draws ellipses.

Syntax	Definition
.Left	Specifies the X-coordinate of the upper-left corner of the ellipse's bounding box in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the ellipse's bounding box in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the ellipse's bounding box in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the ellipse's bounding box in pixels, relative to the origin.

Note

The .Ellipse command must be preceded by the appropriate tool settings commands:
.SetPaintColor
.PenSettings
.ShapeSettings

Example

```
.EllipseTool 0, 0, 0, TRUE, TRUE, TRUE
    .FillBitmap "d:\Fill.cpt", 254000, 254000, 0, 0, FALSE, 0, FALSE, FALSE, TRUE
    .Ellipse 51, 199, 232, 356
```

This example creates an ellipse with the upper-left corner of the ellipse's bounding box at the point (58, 46) and the lower-right corner of the ellipse's bounding box at the point (263, 130). These coordinates are expressed in pixels.

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

EndDraw (PHOTO-PAINT)

.EndDraw

This command ends a sequence of drawing commands used with drawing, masking, and erasing tools.

Example

```
.ImageSprayerTool "fruit.cpt", 1, 5, 87, 0, 0
    .ImageSprayerSettings 1, 11, 19, 0, 0, 1, 5, 1
    .RandomSeed 552585248
    .StartDraw 11904, 8320, 0, 0
    .ContinueDraw 14343, 9096, 0, 0
    ...
    .ContinueDraw 61761, 18445, 0, 0
    .EndDraw
```

This example draws a brush stroke containing a variety of fruits.

```
.Eraser 20, 100, 0, 0, 0, 0, 0
    .SetPaperColor 5, 255, 255, 255, 0
    .StartDraw 40320, 8832, 0, 0
    .ContinueDraw 39040, 8832, 0, 0
    ...
    .ContinueDraw 59678, 18166, 0, 0
    .EndDraw
```

This example erases a portion of your image defined by the draw path..

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

Eraser (PHOTO-PAINT)

.Eraser .Width = *long*, .Flatten = *long*, .Rotate = *long*, .NibShape = *long*, .Transparency = *long*, .SoftEdge = *long*

This command erases a portion of your image defined by a series of Draw commands. An Eraser command block must contain a series of StartDraw and ContinueDraw commands and end with an EndDraw command.

Syntax	Description
.Width	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Flatten	Specifies the flatness of the nib. Valid values range from 0 to 100%.
.Rotate	Specifies the rotation of the nib. Valid values range from 0 to 360 degrees.
.NibShape	Specifies the shape of the nib: 0 = Round 1 = Square
.Transparency	Specifies the transparency of the nib. Valid values range from 0 to 100%.
.SoftEdge	Specifies the transparency of the nib's edges. As you increase the value of this setting, the soft edge expands to eventually reach the center of the stroke. Low values affect only the rim of the stroke. Valid values range from 0 to 100%.

Example

```
.Eraser 20, 100, 0, 0, 0, 0
    .SetPaperColor 5, 255, 255, 255, 0
    .StartDraw 40320, 8832, 0, 0
    .ContinueDraw 39040, 8832, 0, 0
    ...
    .ContinueDraw 59678, 18166, 0, 0
    .EndDraw
```

This example replaces the portion of your image defined by the Draw commands with the defined paper color.

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

Fill (PHOTO-PAINT)

.Fill .Left = *long*, .Top = *long*, .AntiAlias = *boolean*

This command fills areas based on the color similarities of adjacent pixels.

Syntax	Definition
.Left	Specifies the X-coordinate of the point where filling begins in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the point where filling begins in pixels, relative to the origin.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.

Note

The .Fill command must be preceded by the appropriate tool settings commands:
.FillTool
.ToleranceSettings

The .Fill command must be preceded by appropriate setup commands from the following group:
.FillSolid
.FillFountainColor and .FillFountainApply
.FillBitmap
.FillTexture

Example

```
FillTool 0, 0
    .ToleranceSettings 0, 100, 10, 10, 10
    .FillSolid 5, 0, 0, 255, 0
    .Fill 73, 53, TRUE
```

This example applies a solid blue fill beginning at the point (73, 53) with no anti-aliasing. These coordinates are expressed in pixels.

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

FillFountain (PHOTO-PAINT)

.FillFountain

This command creates a fountain fill in the current image.

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

Gradient (PHOTO-PAINT)

.Gradient .Type = *long*, .MarkerPos = *long*

This command creates a gradient fill in the current image, ending a GradientTool command block.

Syntax	Description
.Type	Specifies the type of gradient: 1 = Flat 2 = Linear 3 = Elliptical 4 = Radial 5 = Rectangular 6 = Square 7 = Conical 8 = Bitmap 9 = Texture
.MarkerPos	Marks the halfway point of the transparency range in the color gradient. Drag the slider to move the halfway point of the transparency range to a new position on the gradient.

Example

```
.GradientTool 5, 0, 0, 4
    .SetPaintColor 5, 102, 255, 0, 0
    .SetPaperColor 8, 0, 0, 0, 0
    .GradientPoint 0, 23, 169, 5, 255, 255, 255, 0, 255
    .GradientPoint 1, 65, 133, 5, 153, 255, 0, 0, 255
    .GradientPoint 2, 124, 82, 5, 255, 51, 0, 0, 255
    .GradientPoint 3, 173, 39, 8, 0, 0, 0, 0, 255
    .Gradient 2, 50
```

This example applies a gradient fill to the active objects.

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

GradientPoint (PHOTO-PAINT)

.GradientPoint .Index = *long*, .PtX = *long*, .PtY = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*, .Transparency = *long*

This command sets the gradient point for an image.

Syntax	Description
.Index	References the nodes created on the gradient. The start node is always 0.
.PtX	Specifies the horizontal coordinate (x) of the gradient.
.PtY	Specifies the vertcial coordinate (x) of the gradient.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.
.Transparency	Specifies the transparency of each node applied to the gradient. Valid values range from 0 to 100.

Example

```
.GradientTool 5, 0, 0, 4
    .SetPaintColor 5, 102, 255, 0, 0
    .SetPaperColor 8, 0, 0, 0, 0
    .GradientPoint 0, 23, 169, 5, 255, 255, 255, 0, 255
    .GradientPoint 1, 65, 133, 5, 153, 255, 0, 0, 255
    .GradientPoint 2, 124, 82, 5, 255, 51, 0, 0, 255
    .GradientPoint 3, 173, 39, 8, 0, 0, 0, 0, 255
    .Gradient 2, 50
```

This example applies a gradient fill to the active objects.

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

Rectangle (PHOTO-PAINT)

.Rectangle .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*

This command draws rectangles and rounded rectangles.

Syntax	Definition
.Left	Specifies the X-coordinate of the upper-left corner of the rectangle in pixels, relative to the origin.
.Top	Specifies the Y-coordinate of the upper-left corner of the rectangle in pixels, relative to the origin.
.Right	Specifies the X-coordinate of the lower-right corner of the rectangle in pixels, relative to the origin.
.Bottom	Specifies the Y-coordinate of the lower-right corner of the rectangle in pixels, relative to the origin.

Note

The .Rectangle command must be preceded by the appropriate tool settings commands:

.RectangleTool
.FillSolid
.FillBitmap
.FillTexture

Example

```
.RectangleTool 0, 0, 0, 0, -1, -1, -1
    .FillSolid 3, 0, 0, 255, 0
    .Rectangle 20, 20, 200, 200
```

This example creates a solid blue rectangle.

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

StartCloneDraw (PHOTO-PAINT)

.StartCloneDraw .SrcPtX = *long*, .SrcPtY = *long*, .DestPtX = *long*, .DestPtY = *long,* .Tilt = *long,* .Rotate = *long*

This command begins a stroke of the clone tool. A StartCloneDraw command block must contain one or more ContinueDraw commands and end with an EndDraw command.

Syntax	Description
.SrcPtX	Specifies the horizontal coordinate of the point in your drawing you want to clone. Expressed in normal coordinates.
.SrcPtY	Specifies the vertical coordinate of the point in your drawing you want to clone. Expressed in normal coordinates.
.DestPtX	Specifies the horizontal coordinate of the point you want to clone to. Expressed in 1/256 of a pixel.
.DestPtY	Specifies the vertical coordinate of the point you want to clone to. Expressed in 1/256 of a pixel.
.Tilt	Specifies the tilt attribute of the clone brush tool. Values range from 0-90.
.Rotate	Specifies the rotation angle of the clone brush tool. Values range from 0-360.

Example

```
.CloneTool 0, 1, 0, 20, 0, 25, 0, 0, 100, 25
    .BrushTextureSettings "", 0, 0, 0, 0, 10, TRUE, FALSE
    .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
    .SetPaintColor 5, 255, 255, 0, 0
    .RandomSeed 1905306265
    .StartCloneDraw 235, 100, 73528, 30890, 0, 0
    .ContinueDraw 71846, 30273, 0, 0
    .....
    .ContinueDraw 71408, 29783, 0, 0
    .EndDraw
```

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

StartDraw (PHOTO-PAINT)

.StartDraw .ptX = *long*, .ptY = *long*, .Timer = *long*, .Pressure = *long,* .Tilt = *long,* .Rotate = *long*

This command begins a freehand tool command such as drawing, masking, and erasing.

Syntax	Definition
.ptX	Specifies the X-coordinate of the point to start drawing. For Brush, Effects and Clone tool, this value is expressed in 1/256 of a pixel, relative to the origin. For other tools, this value is expressed in pixels, relative to the origin.
.ptY	Specifies the Y-coordinate of the point to start drawing. For Brush, Effects and Clone tool, this value is expressed in 1/256 of a pixel, relative to the origin. For other tools, this value is expressed in pixels, relative to the origin.
.Timer	Used for tools like the airbrush, where the nib size changes with time. Values start at 0 and increment by 1 for every clock cycle.
.Pressure	Used for a Pen Tablet that has pressure sensitivity. Values range from 0 to 9999.
.Tilt	Specifies the tilt attribute of the clone brush tool. Values range from 0-90.
.Rotate	Specifies the rotation angle of the clone brush tool. Values range from 0-360.

Example

```
.ImageSprayerTool "fruit.cpt", 1, 5, 87, 0, 0
    .ImageSprayerSettings 1, 11, 19, 0, 0, 1, 5, 1
    .RandomSeed 552585248
    .StartDraw 11904, 8320, 0, 0
    .ContinueDraw 14343, 9096, 0, 0
    ...
    .ContinueDraw 61761, 18445, 0, 0
    .EndDraw
```

This example draws a brush stroke containing a variety of fruits.

```
.Eraser 20, 100, 0, 0, 0, 0, 0
    .SetPaperColor 5, 255, 255, 255, 0
    .StartDraw 40320, 8832, 0, 0
    .ContinueDraw 39040, 8832, 0, 0
    ...
    .ContinueDraw 59678, 18166, 0, 0
    .EndDraw
```

This example erases a portion of your image defined by the draw path.

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

Tool commands (PHOTO-PAINT)

BrushDabSettings (PHOTO-PAINT)

.BrushDabSettings .Dabs = *long*, .Spacing = *long*, .Spread = *long*, .FadeOut = *long*, .Hue = *long*, .Saturation = *long*, .Luminance = *long*

This command sets the attributes for brush dabs.

Syntax	Definition
.Dabs	Specifies the number of parallel strokes drawn. Valid values range from 1 to 25.
.Spacing	Specifies the spacing between dabs as a percentage of brush size. Valid values range from 1 to 999 pixels.
.Spread	Specifies maximum random deviation of the dab positions as a percentage of brush size. Valid values range from 0 to 999. Its effect changes depending whether the .Dabs parameter is equal to 1 or greater than 1. If the .Dabs parameter equals 1, each time a dab is drawn its position is varied randomly. If the .Dabs parameter is greater than 1, the initial positions of the strokes are selected randomly, and they maintain their relative positions for the rest of the stroke.
.FadeOut	Specifies the length of the brush stroke before it fades out entirely. A value of zero sets the fade out to none. Valid values range from 0 to 100.
.Hue	Specifies the random variation in the hue (a particular color) when each dab is drawn, as a percentage. Valid values range from 0 to 100.
.Saturation	Specifies the random variation in the saturation (amount of a color) when each dab is drawn, as a percentage. Valid values range from 0 to 100.
.Luminance	Specifies the random variation of color brightness when each dab is drawn, as a percentage. Valid values range from 0 to 100.

Example

```
.SetPaintColor 5, 0, 51, 204, 0
.BrushSettings 25, 0, 1, 0, 20, 0, 20, 0, 0, 100, 80
.BrushTextureSettings , 0, 0, 0, 0, 0, -1
.BrushDabSettings 4, 20, 90, 50, 40, 60, 50
.StartDraw 13, 16
    .ContinueDraw 14, 21
    .ContinueDraw 16, 26
     ...
     ...
    .ContinueDraw 46, 24
    .ContinueDraw 42, 20
.EndDraw
```

This example sets the brush dab settings to 4 parallel strokes, with a spacing of 20, a spread of 90, fade out of 50, hue variation 40%, saturation variation 50%, and luminance variation 60%.

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

BrushOrbitSettings (PHOTO-PAINT)

.BrushOrbitSettings .Points = *long*, .Radius = *long*, .Rotation = *long*, .GrowRate = *long*, .GrowScale = *long*, .Center = *boolean*, .Clockwise = *boolean*, .ColorHue = *long*, .ColorSaturation = *long*, .ColorLightness = *long*, .Hue = *long*, .Saturation = *long*, .Lightness = *long*

This command...

Syntax	Description
.Points	*long*
.Radius	*long*
.Rotation	*long*
.GrowRate	*long*
.GrowScale	*long*
.Center	*boolean*
.Clockwise	*boolean*
.ColorHue	*long*
.ColorSaturation	*long*
.ColorLightness	*long*
.Hue	*long*
.Saturation	*long*
.Lightness	*long*

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

BrushTextureSettings (PHOTO-PAINT)

.BrushTextureSettings .TextureFile = *string*, .BrushTexture = *long*, .EdgeTexture = *long*, .Bleed = *long*, .SustainColor = *long*, .Smoothing = *long*, .AntiAlias = *boolean*, .Cumulative = *boolean,* .Range = *long,* .MergedSource = *boolean*

This command sets the attributes for the brush texture.

Syntax	Definition
.TextureFile	Specifies the name of the Texture file and its path (optional). If the default texture is used, TextureFile does not need to be set.
.BrushTexture	Specifies the amount of texture of the brush. Increase the value to make the brush stroke more coarse. Valid values range from 0 to 100.
.EdgeTexture	Specifies the amount of texture of the brush. Valid values range from 0 to 100.
.Bleed	Specifies the amount of bleed, which controls the amount of color diffusion when two or more colors are combined (as when red paint, for example, contacts blue). Bleed is especially useful when working with watercolors where interesting effects may be achieved by blending two or more colors together. Valid values range from 0 to 100.
.SustainColor	The Sustain Color control works in tandem with the Bleed control. The Bleed control works in tandem with the Transparency control. Consequently, you must enter a Bleed and Transparency when entering a Sustain Color value. Sustain Color retains brush paint color when painting over a colored background while applying bleed to the brush. Typically, using Bleed, the brush will eventually (during the course of an extended brush stroke) run out of paint and simple smear the background color with the brush. With Sustain Color, traces of the paint color remain throughout the brush stroke. Valid values range from 0 to 100.
.Smoothing	Specifies the amount of brush stroke smoothing, in pixels. Smoothing helps to create a more flowing and fluid paint stroke by smoothing out the sharper angles while you paint. Valid values range from 0 to 25.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.Cumulative	Set to TRUE (-1) to make the effects of brush strokes cumulative. Set to FALSE (0) if you want each brush stroke to "max out" after a certain point. For example, if you are applying a tint to an area and want it to appear uniform.
.Range	Specifies the range of effect for the Dodge/Burn tool, where 0 = Highlights 1 = Midtones 2 = Shadows
.MergedSource	Set to TRUE (-1) to use all objects. Set to FALSE (0) to use the active objects only.

Example

```
.SetPaintColor 5, 0, 204, 0, 0
.BrushSettings 25, 4, 0, 0, 20, 2, 2, 6, 10, 95, 30
.BrushTextureSettings , 3, 0, 0, 0, 24, -1
.BrushDabSettings 1, 11, 9, 0, 0, 0, 0
.NibSettings "PNTBRBR.MSK", 19
.StartDraw 89, 26
    .ContinueDraw 89, 27
    .ContinueDraw 90, 28
     ...
     ...
    .ContinueDraw 93, 29
    .ContinueDraw 94, 30
.EndDraw
```

This example sets the brush texture to the default texture with coarseness set to 3, 24 percent smoothing and anti-aliasing enabled.

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

BrushTool (PHOTO-PAINT)

.BrushTool .BrushID = *long*, .TypeID = *long*, .MergeMode = *long*, .Amount = *long*, .NibShape = *long*, .Size = *long*, .Transparency = *long*, .Rotate = *long*, .Flatten = *long*, .SoftEdge = *long*

This command creates a brush stroke with one of the predefined brushes. A BrushTool command block must end with a PathStroke command.

Syntax	Description
.BrushID	Specifies the brush to use by index number.
.TypeID	Specifies the type of brush stroke by index number.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.Amount	Specifies the rate at which the effect or paint is applied to the image, ranging from 1 to 100. A higher value results in a more pronounced effect or heavier application of paint.
.NibShape	Specifies the shape of the nib: 0 = Round 1 = Square
.Size	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Transparency	Specifies the transparency level of the nib. Valid values range from 0 to 99%.
.Rotate	Specifies the angle at which the nib is rotated. Valid values range from 0 to 360 degrees.
.Flatten	Specifies the flatness of the nib. Valid values range from 0 to 99.
.SoftEdge	Specifies the transparency of the nib's edges. As you increase the value of this setting, the soft edge expands to eventually reach the center of the stroke. Low values affect only the rim of the stroke. Valid values range from 0 to 100%.

Example

```
.BrushTool 0, 0, 0, 20, 0, 5, 0, 0, 100, 0
    .BrushTextureSettings "", 0, 0, 0, 0, 0, TRUE, FALSE
    .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
    .SetPaintColor 5, 186, 159, 106, 0
    .PathStroke
```

This example draws a brush stroke.

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

CloneTool (PHOTO-PAINT)

.CloneTool .BrushID = *long*, .TypeID = *long*, .MergeMode = *long*, .Amount = *long*, .NibShape = *long*, .Size = *long*, .Transparency = *long*, .Rotate = *long*, .Flatten = *long*, .SoftEdge = *long*

This command copies a part of your image to another part, following a path defined by StartCloneDraw and ContinueDraw commands. A CloneTool command block must end with an EndDraw command.

Syntax	Description
.BrushID	Specifies the brush to use by index number.
.TypeID	Specifies the type of brush stroke by index number.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.Amount	Specifies the rate at which the effect or paint is applied to the image, ranging from 1 to 100. A higher value results in a more pronounced effect or heavier application of paint.
.NibShape	Specifies the shape of the nib: 0 = Round 1 = Square
.Size	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Transparency	Specifies the transparency of the nib. Valid values range from 0 to 100%.
.Rotate	Specifies the rotation of the nib. Valid values range from 0 to 360 degrees.
.Flatten	Specifies the flatness of the nib. Valid values range from 0 to 100%.
.SoftEdge	Specifies the transparency of the nib's edges. As you increase the value of this setting, the soft edge expands to eventually reach the center of the stroke. Low values affect only the rim of the stroke. Valid values range from 0 to 100%.

Example

```
.CloneTool 0, 0, 0, 20, 1, 33, 0, 0, 100, 80
    .BrushTextureSettings "", 0, 0, 0, 0, 10, TRUE, FALSE
    .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
    .SetPaintColor 5, 186, 159, 106, 0
    .RandomSeed 721390233
    .StartCloneDraw 28, 250, 38016, 23168, 0, 0
    .ContinueDraw 39046, 25228, 0, 0
    ...
    .ContinueDraw 55430, 26365, 0, 0
    .EndDraw
```

This example clones one part of your image to another part.

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

ColorReplace (PHOTO-PAINT)

.ColorReplace

This command modifies all pixels similar or the same to the current paint color in the image to the active paper color.

Syntax	Description
.ToleranceMode	Specifies the tolerance mode: 0 = Normal 1 = HSB
.Normal	If ToleranceMode is set to 0, specifies the tolerance as a percentage (0-100). If ToleranceMode is set to 1 (HSB mode).
.Hue	Specifies the hue tolerance as a percentage (0-100). In the HSB color model, hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. This parameter is used only if ToleranceMode is set to 1.
.Saturation	Specifies the saturation tolerance as a percentage (0-100). Saturation is the purity of a color. The HSB color model uses Saturation as a component that determines the purity or intensity of a color. The more colors used to mix a color, the duller the color looks. This parameter is used only if ToleranceMode is set to 1.
.Brightness	Specifies the brightness tolerance as a percentage (0-100). In the HSB color model, the component that determines the amount of black in a color. This parameter is used only if ToleranceMode is set to 1.

Example

```
.ColorReplace 0, 10, 10, 10, 10
```

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

ColorReplacerTool (PHOTO-PAINT)

.ColorReplacerTool .Width = *long*, .Flatten = *long*, .Rotate = *long*, .NibShape = *long*, .Transparency = *long*, .SoftEdge = *long*

This command sets the attributes of the Color Replacer tool.

Syntax	Description
.Width	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Flatten	Specifies the flatness of the nib. Valid values range from 0 to 100%.
.Rotate	Specifies the rotation of the nib. Valid values range from 0 to 360 degrees.
.NibShape	Specifies the shape of the nib: 0 = Round 1 = Square
.Transparency	Specifies the transparency of the nib. Valid values range from 0 to 100%.
.SoftEdge	Specifies the transparency of the nib's edges. As you increase the value of this setting, the soft edge expands to eventually reach the center of the stroke. Low values affect only the rim of the stroke. Valid values range from 0 to 100%.

Example

```
.ColorReplacerTool 20, 100, 0, 0, 0, 80, 0
    .SetPaintColor 5, 255, 255, 0, 0
    .SetPaperColor 5, 255, 153, 204, 0
    .ToleranceSettings 0, 10, 0, 0, 0
    .StartDraw 50773, 36295, 0, 0
    .EndDraw
```

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

EndObjectColorTransparencyTool (PHOTO-PAINT)

.EndObjectColorTransparencyTool

This command completes the .ObjectColorTransparencyTool operation.

Example

```
.ObjectColorTransparencyTool 0, TRUE, 27
    .ColorMaskColor 0, 5, 62, 155, 0, 0, 14, 30, 50, 10
    .ColorMaskColor 1, 5, 62, 155, 0, 0, 14, 30, 50, 10
    .ColorMaskColor 2, 5, 112, 102, 28, 0, 14, 30, 50, 10
    .EndObjectColorTransparencyTool
```

This example makes the selected colors transparent in the currently selected object using a clip mask.

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

EffectPlugin (PHOTO-PAINT)

.EffectPlugin .GroupName = *string*, .EffectName = *string*, .MemoryType = *long*, .MemorySize = *long*, .Parameters = *string*

This command activates the specified plug-in effect.

Syntax	Definition
.GroupName	Specifies the name of the effect group. Refer to the Effects menu for a complete list of effect groups.
.EffectName	Specifies the name of the effect. Refer to the Effects menu drop-down menu for a list of available effect names.
.MemoryType	Specifies the memory type 0 = None 1 = Handle 2 = Pointer
.MemorySize	Specifies the size of memory in bytes.
.Parameters	Specifies the string of hexadecimal characters.

Note

The group and effect names may be different in different languages, which might result in non-portability of scripts using this command.

Example

```
.EffectPlugin "3D Effects", "Emboss ...", 1, 56,
"db0000000100000003000000000000000100000001000000320000003200000007f9099050000000000000000000000
000000000000000000000"
```

This example activates the Emboss effect.

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

EllipseTool (PHOTO-PAINT)

.EllipseTool .Width = *long*, .Transparency = *long*, .MergeMode = *long*, .AntiAlias = *boolean*, .RenderObject = *boolean*, .Fill = *boolean*

This command sets the attributes of the Ellipse tool.

Syntax	Description
.Width	Specifies the width of the shape's outline. Valid values range from 1 to 999 pixels.
.Transparency	Specifies the transparency of the shape's outline. Valid values range from 0 to 100%.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.RenderObject	Set to TRUE (-1) to convert the shape to an object.
.Fill	Set to TRUE (-1) to fill the shape with the current fill. Set to FALSE (0) to leave shape unfilled.

Example

```
.EllipseTool 0, 0, 0, TRUE, TRUE, TRUE
    .FillBitmap "d:\Fill.cpt", 254000, 254000, 0, 0, FALSE, 0, FALSE, FALSE, TRUE
    .Ellipse 51, 199, 232, 356
```

This example applies the specified bitmap fill inside an ellipse.

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

FillBitmap (PHOTO-PAINT)

.FillBitmap .BitmapName = *string*, .Width = *long*, .Height = *long*, .XOffset = *long*, .YOffset = *long*, .TileColumn = *boolean*, .TileOffset = *long*, .MaintainAspect = *boolean*, .Scale = *boolean*, .OriginalSize = *boolean,* .SkewAngle = *long,* .RotateAngle = *long*

This command specifies bitmap fill settings to be used by a following Fill, Rectangle, Ellipse or Polygon command.

Syntax	Definition
.BitmapName	Specifies the name of the bitmap file.
.Width	Specifies the width of the bitmap in pixels.
.Height	Specifies the height of the bitmap in pixels.
.XOffset	Specifies the amount of offset applied to the first tile along the x-axis. Valid values range from 0 to 100.
.YOffset	Specifies the amount of offset applied to the first tile along the y-axis. Valid values range from 0 to 100.
.TileColumn	Set to TRUE (-1) to enable column offset. Set to FALSE (0) to enable row offset.
.TileOffset	Specifies the amount of row or column offsets. Valid values range from 0 to 100.
.MaintainAspect	Set to TRUE (-1) to maintain the aspect. Set to FALSE (0) to alter the aspect.
.Scale	Set to TRUE (-1) to scale the bitmap. Set to FALSE (0) to disable scaling.
.OriginalSize	Set to TRUE (-1) to maintain the original size. Set to FALSE (0) to alter the original size.
SkewAngle	Specifies the skew angle. This is equivalent to dragging the skew handles onscreen.
RotateAngle	Specifies the angle of rotation for the bitmap fill. This is equivalent to dragging the rotation handles onscreen.

Example

```
.FillTool 0, 0
        .ToleranceSettings 0, 6, 10, 10, 10
        .FillBitmap "d:\corel\graphics8\custom\tiles\breadm.cpt", 338666, 338666, 0, 0, FALSE, 0, FALSE, FALSE, TRUE, 24, 36
        .Fill 82, 99, TRUE
```

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

FillFountainApply (PHOTO-PAINT)

.FillFountainApply .Type = *long*, .Colors = *long*, .Steps = *long*, .Angle = *long*, .EdgePad = *long*, .HorizontalOffset = *long*, .VerticalOffset = *long*, .Midpoint = *long*

This command is used with the FillFountainColor command to specify fountain-fill settings in an EditFill command block.

Syntax	Definition
.Type	Specifies the type of Fountain Fill to apply: 0 = Linear 1 = Radial 2 = Conical 3 = Square
.Colors	Specifies the number of colors used for the fill. This number will correspond to the number of calls to the FillFountainColor command.
.Steps	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.
.Angle	Specifies the angle at which the fill is applied in tenths of degrees. Positive values will rotate the fill counter-clockwise, negative values will rotate it clockwise.
.EdgePad	Specifies the amount of padding to apply to the fill Ignored for type 2. Valid values range from 0 to 45 percent.
.HorizontalOffset	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.
.VerticalOffset	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.
.Midpoint	Specifies the location in the fill of the color midway between the two endpoint colors. It is specified as a percentage (1-to-99%). It only applies to 2-color fountain fills.

Example

```
.EditFill 0, 0, 100, 3, 3, 294, 240, 79, 164, 114, 61
    .FillFountainColor 5, 2, 74, 123, 0, 0, 0
    .FillFountainColor 5, 255, 255, 255, 0, 50, 1
    .FillFountainColor 5, 63, 125, 122, 0, 100, 2
    .FillFountainApply 3, 3, 256, 135, 0, 0, 0, 50
    .EndEditFill
```

This example creates an Elliptical Fountain Fill with a starting transparency of 0 and an ending transparency of 100.

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

FillFountainColor (PHOTO-PAINT)

.FillFountainColor .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*, .Position = *long*, .Index = *long*

This command sets fountain fill colors for an EditFill command block.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.
.Position	Specifies the position at which to set the color. If position is 0, then the start color is set. If position is 100, then the end color is set. For other values, a color at that position is added (or changed if one already exists at that position Note: If position is not 0 or 100, Blend is forced to be custom.
.Index	Specifies the position of the color in the palette. Valid values range from 0 to 255.

Example

```
.EditFill 0, 0, 100, 3, 3, 294, 240, 79, 164, 114, 61
    .FillFountainColor 5, 2, 74, 123, 0, 0, 0
    .FillFountainColor 5, 255, 255, 255, 0, 50, 1
    .FillFountainColor 5, 63, 125, 122, 0, 100, 2
    .FillFountainApply 3, 3, 256, 135, 0, 0, 0, 50
    .EndEditFill
```

This example creates an Elliptical Fountain Fill with a starting transparency of 0 and an ending transparency of 100.

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

FillSolid (PHOTO-PAINT)

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

This command specifies a solid fill color to be used by an EditFill or object tool command block.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.RectangleTool 0, 0, 0, 0, TRUE, TRUE, TRUE
    .FillSolid 5, 255, 255, 255, 0
    .Rectangle 299, 115, 510, 280
```

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

FillTexture (PHOTO-PAINT)

.FillTexture .LibraryName = *string*, .TextureName = *string*, .StyleName = *string*

This command specifies a texture-fill preset to be used by an EditFill or object tool command block.

Syntax	Definition
.LibraryName	Specifies the name of the Texture Library.
.TextureName	Specifies the name of the texture.
.StyleName	Specifies the name of the style.

Example

```
.PolygonTool 0, 0, 0, 0, TRUE, FALSE, TRUE
        .FillTexture "", "Putty 2C", "Cdr5:Putty 2C"
        .StartDraw 88888, 79246, 0, 0
        .ContinueDraw 66133, 102855, 0, 0
        .ContinueDraw 124728, 109966, 0, 0
        .ContinueDraw 125582, 82090, 0, 0
        .ContinueDraw 101688, 81521, 0, 0
        .EndDraw
```

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

FillTextureSettings (PHOTO-PAINT)

.FillTextureSettings .Index = *long*, .Value1 = *long*, .Value2 = *long*, Value3 = *long*, Value4 = *long*

This command is used in conjunction with the Texture Fill commands. The parameters set up an internal structure, therefore the command is provided only for advanced users.

Syntax	Definition
.Index	Specifies the index of the control for the desired texture fill.
.Value1	Specifies the numbox value, or R value, depending on the defined texture fill.
.Value2	Specifies the G value (if one is required for the defined texture fill).
.Value3	Specifies the B value (if one is required for the defined texture fill).
.Value4	This value is usually 0.

Note

If this command is not included with a texture fill, the you get the default Fill for that Texture.

Example

```
.FillTool 0, 0
    .ToleranceSettings 0, 100, 10, 10, 10
    .FillTexture "", "Putty 2C", "Cdr5:Putty 2C"
    .FillTextureSettings 0, 28314, 0, 0, 0
    .FillTextureSettings 1, 25, 0, 0, 0
    .FillTextureSettings 2, 75, 0, 0, 0
    .FillTextureSettings 3, 51, 0, 0, 0
    .FillTextureSettings 4, 87, 0, 0, 0
    .FillTextureSettings 5, 52, 0, 0, 0
    .FillTextureSettings 6, 98, 0, 0, 0
    .FillTextureSettings 7, 12, 0, 0, 0
    .FillTextureSettings 8, 73, 40, 68, 0
    .FillTextureSettings 9, 78, 100, 74, 0
    .Fill 338, 177, TRUE
```

This example has ten texture parameters.

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

FillTool (PHOTO-PAINT)

.FillTool .Transparency = *long*, .MergeMode = *long*

This command creates a fill in the current image.

Syntax	Description
.Transparency	Specifies the transparency of the fill. Valid values range from 0 to 100%.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black

Example

```
FillTool 0, 0
    .ToleranceSettings 0, 100, 10, 10, 10
    .FillSolid 5, 0, 0, 255, 0
    .Fill 73, 53, TRUE
```

This example applies a solid blue fill beginning at the point (73, 53) with no anti-aliasing. These coordinates are expressed in pixels.

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

GradientTool (PHOTO-PAINT)

.GradientTool .Style = *long*, .MergeMode = *long*, .Transparency = *long,* .Handles = *long*

This command draws a gradient in the current image. A GradientTool command block must end with a Gradient command.

Syntax	Description
.Style	Specifies the type of transparency pattern: 0 = Paint to Paper 1 = Paint to Transparency 2 = Transparent to Paint 3 = Clockwise Spectrum 4 = Counterclockwise Spectrum
.MergeMode	Specifies the Merge Mode: 0 = Normal — 12 = Lum 1 = Add — 13 = Invert 2 = Subtract — 14 = And 3 = Difference — 15 = Or 4 = Multiply — 16 = Xor 5 = Divide — 17 = Red 6 = Lighter — 18 = Green 7 = Darker — 19 = Blue 8 = Texturize — 20 = Cyan 9 = Color — 21 = Magenta 10 = Hue — 22 = Yellow 11 = Saturation — 23 = Black
.Transparency	Sets the transparency level of the fill. Valid values range from 0 to 100%.
.Handles	Specifies the number of handles used to define the gradient.

Example

```
.GradientTool 5, 0, 0, 4
    .SetPaintColor 5, 102, 255, 0, 0
    .SetPaperColor 8, 0, 0, 0, 0
    .GradientPoint 0, 23, 169, 5, 255, 255, 255, 0, 255
    .GradientPoint 1, 65, 133, 5, 153, 255, 0, 0, 255
    .GradientPoint 2, 124, 82, 5, 255, 51, 0, 0, 255
    .GradientPoint 3, 173, 39, 8, 0, 0, 0, 0, 255
    .Gradient 2, 50
```

This example applies a gradient fill to the active objects.

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

LineTool (PHOTO-PAINT)

.LineTool .Width = *long*, .Transparency = *long*, .Joints = *long*, .MergeMode = *long*, .AntiAlias = *boolean*, .RenderObject = *boolean*

This command draws a line in your drawing. A LineTool command block must contain StartDraw and ContinueDraw commands, and end with an EndDraw command.

Syntax	Description
.Width	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Transparency	Specifies the transparency level of the nib. Valid values range from 0 to 99%.
.Joints	Specifies the type of corner joint: 0 = Butt 1 = Filled 2 = Round 3 = Point
.MergeMode	Specifies the Merge Mode: 0 = Normal — 12 = Lum 1 = Add — 13 = Invert 2 = Subtract — 14 = And 3 = Difference — 15 = Or 4 = Multiply — 16 = Xor 5 = Divide — 17 = Red 6 = Lighter — 18 = Green 7 = Darker — 19 = Blue 8 = Texturize — 20 = Cyan 9 = Color — 21 = Magenta 10 = Hue — 22 = Yellow 11 = Saturation — 23 = Black
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.RenderObject	Set to TRUE (-1) to convert the line to an object.

Example

```
.LineTool 20, 0, 0, 0, TRUE, FALSE
    .SetPaintColor 5, 255, 255, 0, 0
    .StartDraw 63004, 87779, 0, 0
    .ContinueDraw 127004, 94037, 0, 0
    .EndDraw
```

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

NibSettings (PHOTO-PAINT)

.NibSettings .FileName = *string*, .NibIndex = *long*

This command is used to select custom nibs.

Syntax	Definition
.FileName	Specifies the name of the Nib file.
.NibIndex	Identifies the Nib to load. Refer to the Nibs roll-up for more details.

Example

```
.NibSettings "PNTBRUSH.MSK", -1
```

This example selects a custom nib from the specified file.

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

PolygonTool (PHOTO-PAINT)

.PolygonTool .Width = *long*, .Transparency = *long*, .Joints = *long*, .MergeMode = *long*, .AntiAlias = *boolean*, .RenderObject = *boolean*, .Fill = *boolean*

This command sets the attributes of the Polygon tool.

Syntax	Description
.Width	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Transparency	Specifies the transparency level of the nib. Valid values range from 0 to 99%.
.Joints	Specifies the type of corner joint: 0 = Butt 1 = Filled 2 = Round 3 = Point
.MergeMode	Specifies the Merge Mode: 0 = Normal — 12 = Lum 1 = Add — 13 = Invert 2 = Subtract — 14 = And 3 = Difference — 15 = Or 4 = Multiply — 16 = Xor 5 = Divide — 17 = Red 6 = Lighter — 18 = Green 7 = Darker — 19 = Blue 8 = Texturize — 20 = Cyan 9 = Color — 21 = Magenta 10 = Hue — 22 = Yellow 11 = Saturation — 23 = Black
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.RenderObject	Set to TRUE (-1) to convert the polygon to an object.
.Fill	Set to TRUE (-1) to fill the shape with the current fill. Set to FALSE (0) to leave shape unfilled.

Example

```
.PolygonTool 0, 0, 0, 0, TRUE, FALSE, TRUE
        .FillTexture "", "Putty 2C", "Cdr5:Putty 2C"
        .StartDraw 88888, 79246, 0, 0
        .ContinueDraw 66133, 102855, 0, 0
        .ContinueDraw 124728, 109966, 0, 0
        .ContinueDraw 125582, 82090, 0, 0
        .ContinueDraw 101688, 81521, 0, 0
        .EndDraw
```

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

PressureSettings (PHOTO-PAINT)

.PressureSettings .Size = *long*, .Transparency = *long*, .Softness = *long*, .Hue = *long*, .Saturation = *long*, .Brightness = *long*, .Texture = *long*, .Bleed = *long*, .Resaturate = *long*, .Mask = *long,* .Elongation = *long*

This command adjusts the behavior of a drawing tool depending on the pressure applied to a pen tablet.

Syntax	Description
.Size	Specifies the ratio of nib size to pen pressure in pixels.
.Transparency	Specifies the ratio of transparency to pen pressure. Valid values range from 0 to 100%.
.Softness	Specifies the ratio of softness to pen pressure. Valid values range from 0 to 100%.
.Hue	Specifies the ratio of hue tolerance to pen pressure. Valid values range from -360 to 360 degrees.
.Saturation	Specifies the ratio of saturation tolerance to pen pressure. Valid values range from 0 to 100%.
.Brightness	Specifies the ratio of brightness tolerance to pen pressure. Valid values range from 0 to 100%.
.Texture	Specifies the ratio of texture density to pen pressure when painting with a texture fill. Valid values range from 0 to 100%.
.Bleed	Specifies the ratio of bleed to pen pressure. Valid values range from 0 to 100%.
.Resaturate	Specifies the sustain color value, such as found in the tool settings dialog. This is the rate at which your pen runs out of ink near the end of a stroke.
.Mask	Specifies which of the above settings to apply. Add any two index numbers together to apply both settings: 0 = None 1 = Size 2 = Transparency 4 = Softness 8 = Hue 16 = Saturation 32 = Brightness 64 = Texture 128 = Bleed 256 = Resaturate
.Elongation	Specifies the tilt and rotation attributes for the pressure-sensitive pen. Values range from 0-999.

Example

```
BrushTool 0, 1, 0, 20, 0, 20, 0, 0, 100, 80
    .BrushTextureSettings "", 0, 0, 0, 0, 0, TRUE, FALSE
    .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
    .SetPaintColor 5, 0, 0, 0, 0
    .PressureSettings 94, -77, 27, 36, 199, 153, 48, 83, 100, 511
    .RandomSeed 450724147
    .StartDraw 11904, 22400, 0, 180
    .ContinueDraw 13104, 22844, 0, 336
    .ContinueDraw 14424, 23333, 0, 508
    ......
    .EndDraw
```

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

RandomSeed (PHOTO-PAINT)

.RandomSeed .Seed = *long*

This command sets the random seed for the tools.

Syntax	Description
.Seed	Specifies the value of the random seed. Values range from 0 to 2, 147, 483. 647.

Example

```
.BrushTool 0, 0, 0, 20, 0, 19, 50, 0, 100, 77
        .BrushTextureSettings "", 0, 0, 0, 0, 12, TRUE, FALSE, 0, FALSE
        .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
        .BrushOrbitSettings 0, 25, 10, 15, 0, TRUE, FALSE, 100, 200, 200, 0, 0, 0
        .SymmetrySettings 251657, 275250, 770, 9
        .SetPaintColor 5, 102, 255, 0, 0
        .RandomSeed 523388952
        .StartDraw 18112, 16960, 0, 0
        .ContinueDraw 16880, 17305, 0, 0
        ......
        .ContinueDraw 22257, 23507, 0, 0
        .ContinueDraw 23444, 23030, 0, 0
        .ContinueDraw 24722, 22978, 0, 0
        .EndDraw
```

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

RectangleTool (PHOTO-PAINT)

.RectangleTool .Width = *long*, .Transparency = *long*, .Roundness = *long*, .MergeMode = *long*, .AntiAlias = *boolean*, .RenderObject = *boolean*, .Fill = *boolean*

This command draws a rectangle in the current image.

Syntax	Description
.Width	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Transparency	Specifies the transparency level of the nib. Valid values range from 0 to 99%.
.Roundness	Specifies the curvature of the rectangle's corners. Valid values range from 0 to 100.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.RenderObject	Set to TRUE (-1) to convert the rectangle to an object.
.Fill	Set to TRUE (-1) to fill the shape with the current fill. Set to FALSE (0) to leave shape unfilled.

Example

```
.RectangleTool 0, 0, 0, 0, -1, -1, -1
    .FillSolid 3, 0, 0, 255, 0
    .Rectangle 20, 20, 200, 200
```

This example creates a solid blue rectangle.

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

RepeatSettings (PHOTO-PAINT)

.RepeatSettings .Repeat = *long,* .Rotate = *long*, .RotateVar = *long*, .ColorFromImage = *long*, .Hue = *long*, .Lightness = *long*, .Saturation = *long*, .AccumulateAngle = *long*, .TangentToPath = *long*, .Scale = *long*, .ScaleVar = *long*, .StrokePath = *long*, .FromMask = *boolean,* .Left = *long*, .Top = *long*, .Right = *long*, .Bottom = *long*,

This command repeats the currently loaded path as a brush stroke.

Syntax	Description
.Repeat	Specifies the number of times to repeat the brush stroke.
.Rotate	Specifies the rotation of the brush stroke. Valid values range from 0 to 360 degrees.
.RotateVar	Specifies the variation of each successive rotation. Valid values range from 0 to 360 degrees.
.ColorFromImage	Specifies the color to use for the brush stroke: 0 = Uses the current paint color 1 = Takes the color that appears in the image at the starting point of the brush stroke
.Hue	Specifies the variation in the Hue color channel of each brush stroke. Valid values range from 0 to 100%.
.Lightness	Specifies the variation in the Lightness color channel of each brush stroke. Valid values range from 0 to 100%.
.Saturation	Specifies the variation in the Saturation color channel of each brush stroke. Valid values range from 0 to 100%.
.AccumulateAngle	Specifies the rotation setting: 0 = Rotates each successive brush stroke by the .Rotate value 1 = Increments the rotation by the .Rotate value each time
.TangentToPath	Specifies the direction of the brush stroke: 0 = Strokes with the path 1 = Strokes perpendicular to the path
.Scale	Specifies the size of the brush stroke relative to the original path.
.ScaleVar	Specifies how much each brush stroke can vary in size.
.StrokePath	Specifies whether to use the stroke path: 0 = Applies the brush stroke within the current bounding box only 1 = Strokes along the current path
.FromMask	Set to TRUE (-1) to apply paint strokes to the edge of the current mask. Otherwise, set to FALSE (0).
.Left	Specifies the horizontal coordinate of the top-left corner of the bounding box.
.Top	Specifies the vertical coordinate of the top-left corner of the bounding box.
.Right	Specifies the horizontal coordinate of the bottom-right corner of the bounding box.
.Bottom	Specifies the vertical coordinate of the bottom-right corner of the bounding box.

Example

```
.BrushTool 0, 0, 0, 20, 0, 20, 0, 0, 100, 25
    .BrushTextureSettings "", 0, 0, 0, 0, 0, TRUE, FALSE
    .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
    .SetPaintColor 5, 255, 255, 0, 0
    .RandomSeed 1090138539
    .RepeatSettings 1, 0, 0, 0, 0, 0, 0, 0, 0, 100, 0, 0, 0, 127, 281, 313, 409
    .StartDraw 78080, 71936, 0, 0
    .ContinueDraw 78080, 70656, 0, 0

    .ContinueDraw 77090, 50207, 0, 0
    .EndDraw
```

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

SetPaintColor (PHOTO-PAINT)

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

This command lets you set the paint color used by the Paint tool.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.SetPaintColor 5, 102, 102, 255, 0
```

This example uses the RGB color mode and sets the drawing color to blue.

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

SetPaperColor (PHOTO-PAINT)

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

This command sets the paper color used when clearing or erasing the image.

Syntax	Definition
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.SetPaperColor 5, 0, 0, 255, 0
```

This example sets the paper color to blue.

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

SymmetrySettings (PHOTO-PAINT)

.SymmetrySettings .CenterX = *long*, .CenterY = *long*, .Mode = *long*, .NumPoints = *long*

This command sets symmetry properties.

Syntax	Description
.CenterX	Sets the x coordinate of the symmetry center.
.CenterY	Sets the y coordinate of the symmetry center.
.Mode	Specifies the mode 0 = None 1 = Radial 2 = Mirror If you want to set vertical symmetry only, add 512 to the mode value. If you want to set horizontal symmetry only, add 256 to the mode value.
.NumPoints	Specifies the number of radial points.

Example

```
.BrushTool 0, 0, 0, 20, 0, 19, 50, 0, 100, 77
        .BrushTextureSettings "", 0, 0, 0, 0, 12, TRUE, FALSE, 0, FALSE
        .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
        .BrushOrbitSettings 0, 25, 10, 15, 0, TRUE, FALSE, 100, 200, 200, 0, 0, 0
        .SymmetrySettings 251657, 275250, 770, 9
        .SetPaintColor 5, 102, 255, 0, 0
        .RandomSeed 523388952
        .StartDraw 18112, 16960, 0, 0
        .ContinueDraw 16880, 17305, 0, 0
        ......
        .ContinueDraw 22257, 23507, 0, 0
        .ContinueDraw 23444, 23030, 0, 0
        .ContinueDraw 24722, 22978, 0, 0
        .EndDraw
```

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

TextSettings (PHOTO-PAINT)

.TextSettings .Bold = *long*, .Italic = *boolean*, .Underline = *boolean*, .Alignment = *long*, .FontName = *string*, .FontSize = *long*, .AntiAlias = *boolean*, .CharSpacing = *long*, .LineSpacing = *long*, .DrawMode = *long*, .RenderMask = *boolean*

This command specifies the settings for text to be drawn with the Text tool.

Syntax	Definition
.Bold	Specifies the text's weight setting. The minimum value is 0 and the maximum value is 1000. The setting for a normal weight is 400. The setting for a bold weight is 700.
.Italic	Set to TRUE (-1) to change text to italics.
.Underline	Set to TRUE (-1) to underline text.
.Alignment	Specifies the text alignment: 0 = Left 1 = Center 2 = Right
.FontName	A string specifying the font name. Installed fonts vary depending on your system.
.FontSize	Specifies the font size, in points.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.CharSpacing	Specifies the spacing between characters. Valid values range from 0 to 100%.
.LineSpacing	Specifies the spacing between lines. Valid values range from 0 to 100%.
.DrawMode	Specifies the draw mode 0 = Normal 1 = Add 2 = Subtract 3 = XOR
.RenderMask	Set to TRUE (-1) to render the text as a mask. Set to FALSE (0) to render the text as a normal object.

Example

```
.TextTool 65, 94, "An example"
    .SetPaintColor 5, 186, 159, 106, 0
    .TextSettings 700, FALSE, FALSE, 0, "News701 BT", 44, TRUE, 26, 85, 0, FALSE
```

This example writes the words "An example" into your image.

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

TextTool (PHOTO-PAINT)

.TextTool .ptX = *long*, .ptY = *long*, .Text = *string*

This command inserts a text object at the specified point in your graphic.

Syntax	Description
.ptX	Specifies the X-coordinate of the upper-left corner for the new text object in pixels, relative to the origin.
.ptY	Specifies the Y-coordinate of the upper-left corner for the new text object in pixels, relative to the origin.
.Text	A string specifying the contents of the text object.

Example

```
.TextTool 65, 94, "An example"
    .SetPaintColor 5, 186, 159, 106, 0
    .TextSettings 700, FALSE, FALSE, 0, "News701 BT", 44, TRUE, 26, 85, 0, FALSE
```

This example writes the words "An example" into your image.

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

ToleranceSettings (PHOTO-PAINT)

.ToleranceSettings .ToleranceMode = *long*, .Normal = *long*, .Hue = *long*, .Saturation = *long*, .Brightness = *long*

This command specifies the color tolerance to be used by the Magic Wand and Fill tools.

Syntax	Definition
.ToleranceMode	Specifies the tolerance mode: 0 = Normal 1 = HSB
.Normal	If ToleranceMode is set to 0, specifies the tolerance as a percentage (0-100). If ToleranceMode is set to 1 (HSB mode).
.Hue	Specifies the hue tolerance as a percentage (0-100). In the HSB color model, hue is the main attribute in a color that distinguishes it from other colors. Blue, green and red, for example, are all hues. This parameter is used only if ToleranceMode is set to 1.
.Saturation	Specifies the saturation tolerance as a percentage (0-100). Saturation is the purity of a color. The HSB color model uses Saturation as a component that determines the purity or intensity of a color. The more colors used to mix a color, the duller the color looks. This parameter is used only if ToleranceMode is set to 1.
.Brightness	Specifies the brightness tolerance as a percentage (0-100). In the HSB color model, the component that determines the amount of black in a color. This parameter is used only if ToleranceMode is set to 1.

Example

```
.ToleranceSettings 1, 10, 7, 8, 9
```

This example sets the mode to HSB, the Hue to 7, Saturation to 8, and Brightness to 9.

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

TransparencyBrushTool (PHOTO-PAINT)

.TransparencyBrushTool .Width = *long*, .Flatten = *long*, .Rotate = *long*, .NibShape = *long*, .Transparency = *long*, .SoftEdge = *long*, .Opacity = *long*, .UseOriginal = *boolean,* AntiAlias = *boolean,* ApplyToClipMask = *boolean*

This command makes a part of your drawing more transparent, following a brush stroke defined by a series of Draw commands. A TransparencyBrushTool command block must contain a series of StartDraw and ContinueDraw commands, and end with an EndDraw command.

Syntax	Description
.Width	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Flatten	Specifies the flatness of the nib. Valid values range from 0 to 100%.
.Rotate	Specifies the rotation of the nib. Valid values range from 0 to 360 degrees.
.NibShape	Specifies the shape of the nib: 0 = Round 1 = Square
.Transparency	Specifies the transparency of the nib. Valid values range from 0 to 100%.
.SoftEdge	Specifies the transparency of the nib's edges. As you increase the value of this setting, the soft edge expands to eventually reach the center of the stroke. Low values affect only the rim of the stroke. Valid values range from 0 to 100%.
.Opacity	Specifies the maximum opacity level to apply to pixels repeatedly brushed with the Transparency Brush tool.
.UseOriginal	Set to TRUE (-1) to add the transparency value you set to the transparency value of the pixels in the object. Set to FALSE (0) to replace the existing transparency values of the object's pixels with the values set for this tool.
.AntiAlias	Set to TRUE (-1) to apply anti-aliasing. Set to FALSE (0) to disable anti-aliasing.
.ApplyToClipMask	Set to TRUE (-1) to apply the transparency to the object's clip mask. Otherwise, set to FALSE (0).

Example

```
.TransparencyBrushTool 20, 100, 0, 0, 0, 80, 76, FALSE, TRUE, FALSE
        .StartDraw 24640, 19264, 0, 0
        .ContinueDraw 24556, 20541, 0, 0
        .ContinueDraw 24512, 21820, 0, 0
        .ContinueDraw 24512, 23100, 0, 0
        .....
        .ContinueDraw 33652, 22907, 0, 0
        .ContinueDraw 34928, 23005, 0, 0
        .EndDraw
```

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

Effect commands (PHOTO-PAINT)

AdjustEffectInfo (PHOTO-PAINT)

.AdjustEffectInfo .EffectID = *long*, .Value1 = *long*, .Value2 = *long*, .Value3 = *long*

This command sets the individual effect attributes for an AdjustEffect command. This command must appear in a block starting with one of the AdjustEffect commands.

Syntax	Description
.EffectID	Specifies the effect to adjust: None = 0 Dirsmooth = 1 Smooth = 2 Soften = 3 Gaussblur = 4 Motionblur = 5 Gaussnoise = 6 Spikenoise = 7 Uniformnoise = 8 Diffuse = 9 Jagdespeck = 10 Removenoise = 11 Minimum = 12 Median = 13 Maximum = 14 Aunsharp = 15 Dirsharpen = 16 Edgeenhance = 17 Sharpen = 18 Unsharpmask = 19 Motionblurangle = 20
.Value1	Specifies the first value of the adjusted effect. These values vary with the effect being adjusted.
.Value2	Specifies the second value of the adjusted effect. These values vary with the effect being adjusted.
.Value3	Specifies the third value of the adjusted effect. These values vary with the effect being adjusted.

Example

```
.EffectAdjustBlur
    .AdjustEffectInfo 4, 10, 0, 0
    .AdjustEffectInfo 20, 10, 0, 0
    .AdjustEffectInfo 20, 10, 69, 0
    .AdjustEffectInfo 1, 10, 69, 0
    .EndAdjustEffect
```

This example adjusts the Blur effect.

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

Effect3DRotate (PHOTO-PAINT)

.Effect3DRotate .Horizontal = *long*, .Vertical = *long*, .Face = *long*, .BestFit = *boolean*

This command rotates the image horizontally and vertically according to the horizontal and vertical parameters you set. The image is rotated as if it were one side of a three-dimensional box.

Syntax	Description
.Horizontal	Specifies the degree of horizontal rotation. Valid values range from -75 to 75.
.Vertical	Specifies the degree of vertical rotation. Valid values range from -75 to 75.
.Face	Specifies the face of the rotation cube that faces forward. Valid values range from 0 to 4.
.BestFit	Set to TRUE (-1) if you want to ensure that all parts of your image remain within the Image Window.

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

Effect3DStereoNoise (PHOTO-PAINT)

.Effect3DStereoNoise .Depth = *long*, .Dots = *boolean*

This command generates a dithered noise pattern. The result is an image that has the appearance of 3D depth when viewed a certain way.

Syntax	Description
.Depth	Specifies the depth of the stereogram image. Valid values range from 1 to 9.
.Dots	Set to TRUE (-1) to display dots to help focus on the stereogram image.

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

EffectAdaptiveUnsharp (PHOTO-PAINT)

.EffectAdaptiveUnsharp .Percentage = *long*

This command accentuates edge detail by analyzing the pixel value of neighboring pixels.

Syntax	Description
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectAddNoise (PHOTO-PAINT)

.EffectAddNoise .Level = *long*, .Density = *long*, .ColorNoise = *boolean*, .NoiseType = *long*

This command creates a granular effect that adds a texture to a flat or overly blended image.

Syntax	Description
.Level	Specifies how much noise to add. Valid values range from 0 to 100.
.Density	Specifies the density of noise addition. Valid values range from 0 to 100.
.ColorNoise	Set to TRUE (-1) to apply randomly colored noise to the image. Set to FALSE (0) for monochrome noise.
.NoiseType	Specifies the type of noise: 0 = Gaussian. Prioritizes colors along a Gaussian curve 1 = Spike. Uses colors that are distributed around a narrow curve 2 = Uniform. Provides an overall granular appearance

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

EffectAdjustBlur (PHOTO-PAINT)

.EffectAdjustBlur

This command blurs the current image using the blur settings specified by AdjustEffectInfo commands. An EffectAdjustBlur command block must end with an EndAdjustEffect command.

Example

```
.EffectAdjustBlur
    .AdjustEffectInfo 4, 10, 0, 0
    .AdjustEffectInfo 20, 10, 0, 0
    .AdjustEffectInfo 20, 10, 69, 0
    .AdjustEffectInfo 1, 10, 69, 0
    .EndAdjustEffect
```

This example adjusts the Blur effect.

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

EffectAdjustNoise (PHOTO-PAINT)

.EffectAdjustNoise

This command adds noise to the current image using the settings specified by AdjustEffectInfo commands. An EffectAdjustNoise command block must end with an EndAdjustEffect command.

Example

```
.EffectAdjustNoise
    .AdjustEffectInfo 6, 10, 10, 0
    .AdjustEffectInfo 13, 10, 10, 0
    .AdjustEffectInfo 14, 10, 10, 0
    .AdjustEffectInfo 8, 10, 10, 0
    .AdjustEffectInfo 6, 10, 10, 0
    .AdjustEffectInfo 9, 10, 17, 0
    .EndAdjustEffect
```

This example adds noise to the current image.

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

EffectAdjustSharpness (PHOTO-PAINT)

.EffectAdjustSharpness

This command sharpens the current image using the settings specified by AdjustEffectInfo commands. An EffectAdjustSharpness command block must end with an EndAdjustEffect command.

Example

```
.EffectAdjustSharpness
    .AdjustEffectInfo 19, 10, 50, 0
    .AdjustEffectInfo 17, 10, 50, 0
    .AdjustEffectInfo 16, 10, 50, 0
    .AdjustEffectInfo 18, 19, 50, 0
    .EndAdjustEffect
```

This example sharpens the current image.

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

EffectBandPass (PHOTO-PAINT)

.EffectBandPass .InRadius = *long*, .OutRadius = *long*, .InBand = *long*, .MidBand = *long*, .OutBand = *long*

This command adjusts the balance of sharp and smooth areas in an image.

Syntax	Description
.InRadius	Specifies the size of the inner band radius. The values range from 1 to 256.
.OutRadius	Specifies the size of the outer band radius. The values range from 1 to 256.
.InBand	Specifies the weighting of the inner band. To eliminate the sharp or smooth areas within a band, set the weighting to 0. Experiment with different weightings to see which provide the best results. The values range from 0 to 100%.
.MidBand	Specifies the weighting of the middle band. The values range from 0 to 100%.
.OutBand	Specifies the weighting of the outer band. The values range from 0 to 100%.

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

EffectBitPlanes (PHOTO-PAINT)

.EffectBitPlanes .Red = *long*, .Green = *long*, .Blue = *long*

This command reduces the image to basic RGB color components and emphasizes tonal changes. For example, certain areas appear as solid blocks because there is little change in tone.

Syntax	Description
.Red	Specifies the color sensitivity in the red channel. Valid values range from 0 to 7.
.Green	Specifies the color sensitivity in the green channel. Valid values range from 0 to 7.
.Blue	Specifies the color sensitivity in the blue channel. Valid values range from 0 to 7.

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

EffectCanvas (PHOTO-PAINT)

.EffectCanvas .Filename = *string*, .Transparency = *long*, .Emboss = *long*, .X = *long*, .Y = *long*, .Mode = *long*, .Offset = *long*

This command applies a textured surface over top of an image.

Syntax	Description
.Filename	Specifies the name of the file to use as the canvas surface.
.Transparency	Specifies the transparency of the canvas effect. Valid values range from 0 to 100%.
.Emboss	Specifies the degree of embossing. Embossing gives the canvas a raised, relief effect. Valid values range from 0 to 100%.
.X	Specifies the horizontal offset of the canvas map. Valid values range from 0 to 100%.
.Y	Specifies the vertical offset of the canvas map. Valid values range from 0 to 100%.
.Mode	Specifies the offset mode: 0 = Rows. Offsets rows of tiles 1 = Columns. Offsets columns of tiles 2 = Stretch To Fit
.Offset	Specifies the degree of offset. Valid values range from 0 to 100%.

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

EffectDiffuse (PHOTO-PAINT)

.EffectDiffuse .Level = *long*

This command spreads out the pixels of your image to fill in blank spaces and remove noise. Depending on the level you select, the effect can appear smooth, blurry, or produce a soft, double-edged look as if the image were being seen through a photographer's diffusion lens.

Syntax	Description
.Level	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectDirectionalSharpen (PHOTO-PAINT)

.EffectDirectionalSharpen .Percentage = *long*

This command analyzes pixels of similar shades to determine the direction in which to apply the greatest amount of sharpening.

Syntax	Description
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectDirectionalSmooth (PHOTO-PAINT)

.EffectDirectionalSmooth .Percentage = *long*

This command analyzes the value of pixels of similar tonal values to determine the direction in which to apply the greatest amount of smoothing. This subtly smoothes edges and surfaces, giving them anti-aliased edges without distorting the image.

Syntax	Description
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectDisplace (PHOTO-PAINT)

.EffectDisplace .Filename = *string*, .Displacement = *long*, .Edges = *long*, .Horizontal = *long*, .Vertical = *long*

This command alters an image using a displacement map. Corel PHOTO-PAINT includes a number of sample displacement maps you can use; however, you can load any bitmap image as a displacement map. The Displace filter evaluates the color value of pixels in both images, and then shifts the active image according to the values of the displacement map. The result is that values from the displacement map appear as forms, colors, and warp patterns in your image.

Syntax	Description
.Filename	Specifies the name of the image file to use as a displacement map.
.Displacement	Specifies the scaling mode of the displacement: 0 = Tiles the displacement map over the image 1 = Stretches the displacement map to cover the entire image
.Edges	Specifies the method of filling empty areas created by the displacement: 0 = Stretches the edge areas 1 = Wraps the opposite edge around to the empty areas
.Horizontal	Specifies the horizontal shift. Valid values range from 0 to 100.
.Vertical	Specifies the vertical shift. Valid values range from 0 to 100.

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

EffectDustScratch (PHOTO-PAINT)

.EffectDustScratch .Level = *long*, .Radius = *long*

This command reduces image noise by averaging pixel values. This works something like adding water to a dry watercolor painting; adjacent colors bleed into each other. As the name implies, this command is extremely useful for eliminating dust and scratch faults in an image.

Syntax	Description
.Level	Specifies how great a change in value must occur to any pixel before the effect is applied. Valid values range from 0 to 255.
.Radius	Specifies the range of the effect. Larger values increase the number of pixels that are successively selected and evaluated when you apply the effect. Valid values range from 1 to 20.

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

EffectEdgeDetect (PHOTO-PAINT)

.EffectEdgeDetect .Sensitivity = *long,* .ColorModel = *long,* .Color1 = *long,* .Color2 = *long,* .Color3 = *long,* .Color4 = *long*

This command finds the edges of elements in your image, then converts them to lines on a background of a single color, allowing you to add a variety of outline effects to your image.

Syntax	Description
.Sensitivity	Specifies the intensity of the effect. Valid values range from 1 to 10.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EffectEdgeDetect 7, 5, 0, 0, 0, 0
```

This example uses black as the color for the edge effect.

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

EffectEmboss (PHOTO-PAINT)

.EffectEmboss .Depth = *long*, .Level = *long*, .Direction = *long*, .Color = *long,* .ColorModel = *long,* .Color1 = *long,* .Color2 = *long,* .Color3 = *long,* .Color4 = *long*

This command transforms your image into a relief, making the details appear as ridges and crevices on a flat surface.

Syntax	Description
.Depth	Specifies the depth of the ridges and crevices in the relief.
.Level	Specifies the amount of background color the relief will contain.
.Direction	Specifies the angle at which the light hits the relief. Valid values range from 0 to 360.
.Color	Specifies the emboss color: 0 = The original color 1 = Gray 2 = Black 3 = The current paper color
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EffectEmboss 1, 500, 318, 1, 5, 255, 255, 255, 0
```

This example uses gray color options with the Emboss effect.

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

EffectFindEdges (PHOTO-PAINT)

.EffectFindEdges .Level = *long*, .EdgeType = *long*

This command detects the outlines of forms in your image and converts them to soft or solid lines.

Syntax	Description
.Level	Specifies the intensity of the effect. Valid values range from 0 to 100%.
.EdgeType	Specifies the type of edge: 0 = Soft, blurry outline 1 = Sharp, crisp outline

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

EffectGaussianBlur (PHOTO-PAINT)

.EffectGaussianBlur .Radius = *double*

This command produces a hazy effect, blurring the image according to a Gaussian distribution, which spreads the pixel information outward using bell-shaped curves.

Syntax	Description
.Radius	Specifies the intensity of the effect. Valid values range from 1 to 50.

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

EffectGlass (PHOTO-PAINT)

.EffectGlass .Angle = *long*, .BevelWidth = *long*, .Brightness = *long*, .Direction = *long*, .Dropoff = *long*, .Opacity = *long*, .Refraction = *long*, .Sharpness = *long*, .Smooth = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command applies the Glass effect to an image. You must create a mask on the image before applying this effect.

Syntax	Description
.Angle	Specifies the angle at which the light bounces off the bevel.
.BevelWidth	Specifies the width of the bevel.
.Brightness	Specifies the intensity of the light source.
.Direction	Specifies the angle at which the light hits the bevel.
.Dropoff	Specifies the current edge style of the bevel. Gaussian Flat Mesa
.Opacity	Specifies the opacity of the sheet of glass.
.Refraction	Specifies the amount of refraction.
.Sharpness	Specifies the amount of fading at the edge of the light shaft. A lower value results in a concentrated light source (like a flashlight). A higher value results in a softer, larger light source (like a ceiling light).
.Smooth	Specifies the smoothness of the bevel.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EffectGlass 32, 59, 83, 328, 0, 50, 70, 71, 62, 9, 128, 0, 0, 0
```

This example applies the Glass effect to an image.

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

EffectGlassBlock (PHOTO-PAINT)

.EffectGlassBlock .Width = *long*, .Height = *long*

This command mimics the effect of viewing an image through a number of thick glass blocks.

Syntax	Description
.Width	Specifies the width of the glass blocks. Valid values range from 1 to 100.
.Height	Specifies the height of the glass blocks. Valid values range from 1 to 100.

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

EffectHalftone (PHOTO-PAINT)

.EffectHalftone .Radius = *long*, .Cyan = *long*, .Magenta = *long*, .Yellow = *long*, .Black = *long*

This command gives your image the appearance of a color halftone

Syntax	Description
.Radius	Specifies the dot radius. Valid values range from 2 to 10.
.Cyan	Specifies the Cyan channel angle. Valid values range from 0 to 359 degrees.
.Magenta	Specifies the Magenta channel angle. Valid values range from 0 to 359 degrees.
.Yellow	Specifies the Yellow channel angle. Valid values range from 0 to 359 degrees.
.Black	Specifies the Black channel angle. Valid values range from 0 to 359 degrees.

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

EffectHighPass (PHOTO-PAINT)

.EffectHighPass .Radius = *long*, .Percentage = *long*

This command removes low-frequency detail and shading. The effect can give an image an ethereal, glowing quality. It emphasizes the highlights and luminous areas of an image. At higher settings, the High Pass effect removes most of the image detail, leaving only the edge details clearly visible. If you only want to emphasize highlights, use lower percentage settings.

Syntax	Description
.Radius	Specifies the range of the effect. Larger values increase the number of pixels that are successively selected and evaluated when you apply the effect. Valid values range from 1 to 20.
.Percentage	Specifies the intensity of the effect. Larger values remove more shadow detail. Valid values range from 0 o 100%.

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

EffectImpressionist (PHOTO-PAINT)

.EffectImpressionist .Horizontal = *long*, .Vertical = *long*

This command gives your image the look of an impressionist painting by converting your image to dabs of solid color.

Syntax	Description
.Horizontal	Specifies the amount of pixel displacement that occurs along the horizontal axis. Valid values range from 1 to 100.
.Vertical	Specifies the amount of pixel displacement that occurs along the horizontal axis. Valid values range from 1 to 100.

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

EffectJaggyDespeckle (PHOTO-PAINT)

.EffectJaggyDespeckle .Width = *long*, .Height = *long*

This command scatters colors in an image creating a soft, blurred effect with minimal distortion. It is most effective for removing the jagged edges that can appear in line art or high-contrast images.

Syntax	Description
.Width	Specifies the intensity of horizontal color scattering. Valid values range from 1 to 5.
.Height	Specifies the intensity of vertical color scattering. Valid values range from 1 to 5.

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

EffectLensFlare (PHOTO-PAINT)

.EffectLensFlare .X = *double*, .Y = *double*, .Brightness = *long*, .LensType = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command produces rings of light on your image that simulate the flare that appears on a photograph when the camera is aimed toward a direct bright light.

Syntax	Description
.X	Specifies the horizontal coordinate of the flare's center.
.Y	Specifies the vertical coordinate of the flare's center.
.Brightness	Specifies the intensity of the lens flare. The effect of the brightness setting varies with different lens types.
.LensType	Specifies the type of lens, which will affect the appearance of the flare: 0 = 50-300mm zoom 1 = 35mm prime 2 = 105mm prime
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EffectLensFlare 0.73, 0.77, 118, 0, 5, 255, 255, 255, 0
```

This example creates a white lens flare effect.

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

EffectLighting (PHOTO-PAINT)

.EffectLighting .Sources = *long*

This command adds the effect of one or more light sources to your drawing. An EffectLighting command block must contain at least one EffectLightSource command, and end with an EndEffectLighting command.

Syntax	Description
.Sources	Specifies the number of light sources to add.

Example

```
.EffectLighting 4
    .EffectLightSource 0, 0.5, 0.5, 0, 75, 180, 90, 40, 180, 100, 0, 100, 0, 0, 0, 255, 0, 0
    .EffectLightSource 1, 0.097166, 1.09717, 1, 60, 15, 25, 95, 45, 80, 0, 100, 0, 0, 0, 255,
255, 51
    .EffectLightSource 2, 0.433, 1.1, 1, 60, 15, 25, 95, 45, 80, 0, 100, 0, 0, 0, 204, 102, 255
    .EffectLightSource 3, 0.766, 1.1, 1, 60, 15, 25, 95, 45, 80, 0, 100, 0, 0, 0, 255, 255, 255
    .EndEffectLighting
```

This example adds four light sources to your drawing.

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

EffectLowPass (PHOTO-PAINT)

.EffectLowPass .Radius = *long*, .Percentage = *long*

This command removes sharp edges and detail from an image, leaving smooth gradients and low frequency detail.

Syntax	Description
.Radius	Specifies the number of pixels that are successively selected and evaluated when you apply the effect. Valid values range from 1 to 20.
.Percentage	Specifies the intensity of the effect. Larger values reduce harsh transitions between shadows and highlights. Valid values range from 0 to 100%.

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

EffectMapToObject (PHOTO-PAINT)

.EffectMapToObject .Mode = *long*, .Percentage = *long, .*Quality = *long*

This command creates the illusion that the image has been wrapped around a sphere, or a horizontal or vertical cylinder.

Syntax	Description
.Mode	Specifies the mapping mode: 0 = Spherical 1 = Horizontal cylinder 2 = Vertical cylinder
.Percentage	Specifies the direction and amount of wrapping. Negative percentage values wrap the image toward the back; positive percentage values wrap the image toward the front. Valid values range from -100 to 100%.
.Quality	Provides a list of preset quality levels that you can use when applying the filter. 0=Draft 1=Better 2=Best

Example

```
.EffectMapToObject 0, 19, 2
```

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

EffectMaximum (PHOTO-PAINT)

.EffectMaximum .Radius = *long*, .Percentage = *long*

This command removes noise by adjusting pixel values based on the maximum pixel value of neighboring pixels. The command also causes a mild blurring effect if applied in large percentages or more than once. The highest setting will completely obscure your image.

Syntax	Description
.Radius	Specifies the number of pixels that are successively selected and evaluated when you apply the effect. Valid values range from 1 to 20.
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectMedian (PHOTO-PAINT)

.EffectMedian .Radius = *long*, .Percentage = *long*

This command removes noise and detail by averaging the colors of adjacent pixels in the image.

Syntax	Description
.Radius	Specifies the number of pixels that are successively selected and evaluated when you apply the effect. Valid values range from 1 to 20.
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectMeshPoint (PHOTO-PAINT)

.EffectMeshPoint .Row = *long*, .Column = *long*, .X = *double*, .Y = *double*

This command defines the grid points for the EffectMeshWarp command. An EffectMeshWarp command block must contain an EffectMeshPoint command for each point in the warp grid.

Syntax	Description
.Row	Specifies the row number of the point you are defining.
.Column	Specifies the column number of the point you are defining.
.X	Specifies the horizontal coordinate of the point. Valid values range from 0.00 to 1.00.
.Y	Specifies the vertical coordinate of the point. Valid values range from 0.00 to 1.00.

Example

```
.EffectMeshWarp 5, 5
    .EffectMeshPoint 0, 0, 0, 0
    .EffectMeshPoint 0, 1, 0.25, 0
    .EffectMeshPoint 0, 2, 0.5, 0
    .EffectMeshPoint 0, 3, 0.75, 0
    .EffectMeshPoint 0, 4, 1, 0
    .EffectMeshPoint 1, 0, 0, 0.25
    .EffectMeshPoint 1, 1, 0.340067, 0.345946
    .EffectMeshPoint 1, 2, 0.52862, 0.297297
    .EffectMeshPoint 1, 3, 0.861953, 0.145946
    .EffectMeshPoint 1, 4, 1, 0.25
    .EffectMeshPoint 2, 0, 0, 0.5
    .EffectMeshPoint 2, 1, 0.25, 0.5
    .EffectMeshPoint 2, 2, 0.579125, 0.556757
    .EffectMeshPoint 2, 3, 0.75, 0.5
    .EffectMeshPoint 2, 4, 1, 0.5
    .EffectMeshPoint 3, 0, 0, 0.75
    .EffectMeshPoint 3, 1, 0.121212, 0.859459
    .EffectMeshPoint 3, 2, 0.5, 0.75
    .EffectMeshPoint 3, 3, 0.818182, 0.881081
    .EffectMeshPoint 3, 4, 1, 0.75
    .EffectMeshPoint 4, 0, 0, 1
    .EffectMeshPoint 4, 1, 0.25, 1
    .EffectMeshPoint 4, 2, 0.5, 1
    .EffectMeshPoint 4, 3, 0.75, 1
    .EffectMeshPoint 4, 4, 1, 1
    .EndEffectMeshWarp
```

This example distorts the current image using a 5 x 5 mesh warp grid.

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

EffectMeshWarp (PHOTO-PAINT)

.EffectMeshWarp .Width = *long*, .Height = *long*

This command distorts the current image using a grid of points that are displaced to produce the desired warping effect. An EffectMeshWarp command block must contain an EffectMeshPoint command for each point in the warp grid, and the block must end with an EndEffectMeshWarp command.

Syntax	Description
.Width	Specifies the number of columns in the warp grid. Valid values range from 5 to 11.
.Height	Specifies the number of rows in the warp grid. Valid values range from 5 to 11.

Example

```
.EffectMeshWarp 5, 5
    .EffectMeshPoint 0, 0, 0, 0
    .EffectMeshPoint 0, 1, 0.25, 0
    .EffectMeshPoint 0, 2, 0.5, 0
    .EffectMeshPoint 0, 3, 0.75, 0
    .EffectMeshPoint 0, 4, 1, 0
    .EffectMeshPoint 1, 0, 0, 0.25
    .EffectMeshPoint 1, 1, 0.340067, 0.345946
    .EffectMeshPoint 1, 2, 0.52862, 0.297297
    .EffectMeshPoint 1, 3, 0.861953, 0.145946
    .EffectMeshPoint 1, 4, 1, 0.25
    .EffectMeshPoint 2, 0, 0, 0.5
    .EffectMeshPoint 2, 1, 0.25, 0.5
    .EffectMeshPoint 2, 2, 0.579125, 0.556757
    .EffectMeshPoint 2, 3, 0.75, 0.5
    .EffectMeshPoint 2, 4, 1, 0.5
    .EffectMeshPoint 3, 0, 0, 0.75
    .EffectMeshPoint 3, 1, 0.121212, 0.859459
    .EffectMeshPoint 3, 2, 0.5, 0.75
    .EffectMeshPoint 3, 3, 0.818182, 0.881081
    .EffectMeshPoint 3, 4, 1, 0.75
    .EffectMeshPoint 4, 0, 0, 1
    .EffectMeshPoint 4, 1, 0.25, 1
    .EffectMeshPoint 4, 2, 0.5, 1
    .EffectMeshPoint 4, 3, 0.75, 1
    .EffectMeshPoint 4, 4, 1, 1
    .EndEffectMeshWarp
```

This example distorts the current image using a 5 x 5 mesh warp grid.

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

EffectMinimum (PHOTO-PAINT)

.EffectMinimum .Radius = *long*, .Percentage = *long*

This command darkens an image by adjusting pixel values based on the minimum pixel value of neighboring pixels.

Syntax	Description
.Radius	Specifies the number of pixels that are successively selected and evaluated when you apply the effect. Valid values range from 1 to 20.
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectMotionBlur (PHOTO-PAINT)

.EffectMotionBlur .Speed = *long*, .Direction = *long*, .Method = *long*

This command creates the illusion of movement in an image.

Syntax	Description
.Speed	Specifies the degree of image blurring. Valid values range from 1 to 50.
.Direction	Specifies the direction of blurring. Valid values range from 0 to 360 degrees.
.Method	Specifies the method of off-image sampling: 0 = Ignores pixels outside the image 1 = Uses paper color 2 = Samples nearest edge pixel

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

EffectOffset (PHOTO-PAINT)

.EffectOffset .Horizontal = *long*, .Vertical = *long*, .Shift = *boolean*, .Edges = *long,* .ColorModel = *long,* .Color1 = *long,* .Color2 = *long,* .Color3 = *long,* .Color 4 = *long*

This command corrects image positioning by shifting the image.

Syntax	Description
.Horizontal	Specifies the amount of horizontal shifting. Valid values range from -100 to 100.
.Vertical	Specifies the amount of vertical shifting. Valid values range from -100 to 100.
.Shift	Set to TRUE (-1) to coordinate the horizontal and vertical shift values with the size of the object. With a vertical shift value of 50, the image will shift along the vertical plane a distance corresponding to exactly one-half the size of the image. Set to FALSE (0) for absolute offsetting.
.Edges	Specifies the method used to fill the empty area left behind by the offset: 0 = Wraps the opposite edge around to the empty area 1 = Stretches the edges of the image to fill the empty area 2 = Uses the background paper color
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EffectOffset 71, -33, TRUE, 2, 5, 0, 0, 255, 0
```

This example applies the Offset effecct with a blue background.

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

EffectPageCurl (PHOTO-PAINT)

.EffectPageCurl .Corner = *long*, .Direction = *boolean*, .Width = *long*, .Height = *long*, .Opaque = *boolean*, .CurlColorModel = *long*, .CurlColor1 = *long*, .CurlColor2 = *long*, .CurlColor3 = *long*, .CurlColor4 = *long*, .BackColorModel = *long*, .BackColor1 = *long*, .BackColor2 = *long*, .BackColor3 = *long*, .BackColor4 = *long*

This command gives the impression that a corner of your image has rolled in on itself.

Syntax	Description
.Corner	Specifies the corner to curl: 0 = Top-left 1 = Top-right 2 = Bottom-left 3 = Bottom-right
.Direction	Set to TRUE (-1) to have the page curl begin along the top or bottom edge of the image, depending on the .Corner location. Set to FALSE (0) to have the page curl begin along the left or right edge of the image.
.Width	Specifies the width of the page curl. Increase the value to extend the page curl along the vertical edge of the image.
.Height	Specifies the height of the page curl. Increase the value to extend the page curl along the horizontal edge of the image.
.Opaque	Set to TRUE (-1) to make the curl completely opaque. Set to FALSE (0) to have some of the image show through the curl.
.CurlColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.CurlColor1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.CurlColor2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. If this parameter is not available in the Color Model specified, set it to 0.
.CurlColor3	Specifies the third color component for .ColorModel. For example, Blue is the third color component for RGB. If this parameter is not available in the Color Model specified, set it to 0.
.CurlColor4	Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. If this parameter is not available in the Color Model specified, set it to 0.
.BackColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.BackColor1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.BackColor2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. If this parameter is not available in the Color Model specified, set it to 0.
.BackColor3	Specifies the third color component for .ColorModel. For example, Blue is the third color component for RGB. If this parameter is not available in the Color Model specified, set it to 0.
.BackColor4	Specifies the fourth color component for .ColorModel. For example, Black is the fourth color component for CMYK. If this parameter is not available in the Color Model specified, set it to 0.

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

EffectPerspective (PHOTO-PAINT)

.EffectPerspective .X1 = *double*, .Y1 = *double*, .X2 = *double*, .Y2 = *double*, .X3 = *double*, .Y3 = *double*, .X4 = *double*, .Y4 = *double*, .BestFit = *boolean*

This command gives your image a sense of three-dimensional depth, as if it were on a flat plane receding into the distance.

Syntax	Description
.X1	The horizontal coordinate of the top-left handle of the distortion rectangle. Valid values range from 0 to 48.
.Y1	The vertical coordinate of the top-left handle of the distortion rectangle. Valid values range from 0 to 48.
.X2	The horizontal coordinate of the top-right handle of the distortion rectangle. Valid values range from 0 to 48.
.Y2	The vertical coordinate of the top-right handle of the distortion rectangle. Valid values range from 0 to 48.
.X3	The horizontal coordinate of the bottom-right handle of the distortion rectangle. Valid values range from 0 to 48.
.Y3	The vertical coordinate of the bottom-right handle of the distortion rectangle. Valid values range from 0 to 48.
.X4	The horizontal coordinate of the bottom-left handle of the distortion rectangle. Valid values range from 0 to 48.
.Y4	The vertical coordinate of the bottom-left handle of the distortion rectangle. Valid values range from 0 to 48.
.BestFit	Set to TRUE (-1) to keep two nodes equidistant at all times.

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

EffectPinchPunch (PHOTO-PAINT)

.EffectPinchPunch .Level = *long,* .CenterX = *long,* CenterY = *long*

This command warps your image by either "pinching" the image away from you or "punching" it toward you. Negative values apply a punch effect; positive values apply a pinch effect.

Syntax	Description
.Level	Specifies the intensity of the pinch or punch effect. Positive values apply a pinch effect, while negative values apply a punch effect. Valid values range from -100 to 100.
.CenterX	Specifies the horizontal position of the pinch or punch's center.
.CenterY	Specifies the vertical position of the pinch or punch's center.

Example

```
.EffectPinchPunch -54, 158, 126
```

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

EffectPixelate (PHOTO-PAINT)

.EffectPixelate .Width = *long*, .Height = *long*, .Opacity = *long*, .Mode = *long*

This command breaks up your image into square, rectangular, or circular cells.

Syntax	Description
.Width	Specifies the width of the cells. Valid values range from 1 to 100.
.Height	Specifies the height of the cells. Valid values range from 1 to 100.
.Opacity	Specifies the opacity of the cells. Valid values range from 1 to 100%.
.Mode	Specifies the pixelation mode: 0 = Square 1 = Rectangular 2 = Circular

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

EffectPsychedelic (PHOTO-PAINT)

.EffectPsychedelic .Level = *long*

This command changes the colors in your image to bright, electric colors such as orange, hot pink, cyan, and lime green

Syntax	Description
.Level	Specifies the intensity of the effect. Valid values range from 0 to 255.

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

EffectPuzzle (PHOTO-PAINT)

.EffectPuzzle .Width = *long*, .Height = *long*, .Offset = *long*, .Fill = *long,* .RandSeed = *long,* .ColorModel = *long,* .Color1 = *long,* .Color2 = *long,* .Color 3 = *long,* .Color4 = *long*

This command breaks down the image into puzzle-like pieces or blocks, resembling a jigsaw puzzle.

Syntax	Description
.Width	Specifies the width of the puzzle blocks. Valid values range from 1 to 100.
.Height	Specifies the height of the puzzle blocks. Valid values range from 1 to 100.
.Offset	Specifies the amount of shifting that occurs. Valid values range from 0 to 200%.
.Fill	Specifies the method used to fill the empty area behind the puzzle pieces: 0 = Black fill 1 = White fill 2 = Fill using the current paint color 3 = Use the original image 4 = Use a negative of the original image
.RandSeed	Provides a seed for a random number. You can type any value in this parameter.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EffectPuzzle 10, 10, 28, 1, 51665343, 5, 0, 0, 0, 0
```

This example applies the Puzzle effect with white background options.

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

EffectRadialBlur (PHOTO-PAINT)

.EffectRadialBlur .Mode = *long*, .Radius = *long*, .CenterX = *long*, .CenterY = *long*

This command creates a blurring effect that radiates outward from a central point.

Syntax	Description
.Mode	Specifies the radial mode: 0 = Spin 1 = Zoom
.Radius	Specifies the radius of the blur. Valid values range from 0 to 100.
.CenterX	Specifies the horizontal position of the blur's center.
.CenterY	Specifies the vertical position of the blur's center.

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

EffectRemoveMoire (PHOTO-PAINT)

.EffectRemoveMoire .Amount = *long*, .Quality = *long*, .FinalResolution = *long*, .OriginalResolution = *long*

This command applies the Remove Moire effect to an image.

Syntax	Description
.Amount	Specifies the intensity of the effect. Valid values range from 0 to 10.
.Quality	Specifies the level of the effect. Better = 0 Faster = 1
.FinalResolution	Specifies the resolution of the image after the filter is applied
.OriginalResolution	Specifies the resolution of the image before the filter is applied.

Example

```
.EffectRemoveMoire 6, 1, 75, 75
```

This example applies the Remove Moire effect to an image.

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

EffectRemoveNoise (PHOTO-PAINT)

.EffectRemoveNoise .Threshold = *long*, .Auto = *boolean*

This command softens the image and reduces the speckled effect that can occur during the scanning or video capturing process. The command compares each pixel to surrounding pixels, and calculates an average. Each pixel whose brightness value exceeds that of the threshold you set are removed.

Syntax	Description
.Threshold	Specifies how great a change in value must occur to any pixel before the effect is applied.
.Auto	Set to TRUE (-1) to have Corel PHOTO-PAINT automatically calculate the noise reduction level required to improve image quality. Set to FALSE (0) to use the threshold value above.

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

EffectRipple (PHOTO-PAINT)

.EffectRipple .Period = *long*, .Amplitude = *long*, .Angle = *long*, .Distort = *boolean,* .Mode = *long*

This command creates vertical or horizontal rippled waves throughout the image.

Syntax	Description
.Period	Specifies the distance between each wave cycle. Valid values range from 1 to 100.
.Amplitude	Specifies the amount of displacement created by each wave. Valid values range from 1 to 100.
.Angle	Specifies the direction of the ripple effect. Valid values range from 0 to 180 degrees.
.Distort	Set to TRUE (-1) to apply distortion to the ripple.
.Mode	Specifies the Ripple mode to use for the effect 0=Single Wave 1=Dual Wave 1:1 2=Dual Wave 2:1

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

EffectSharpen (PHOTO-PAINT)

.EffectSharpen .EdgeLevel = *long*, .Background = *long*

This command accentuates the edges in the image by finding the edges and increasing the contrast between adjacent pixels.

Syntax	Description
.EdgeLevel	Specifies the amount of edge sharpening. Valid values range from 0 to 100%.
.Background	Specifies how great a change in value must occur to any pixel before the effect is applied.

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

EffectShear (PHOTO-PAINT)

.EffectShear .Scale = *long*, .Border = *long*, .Orientation = *long,* ColorModel = *long,* Color1 = *long,* Color2 = *long,* Color3 = *long,* Color4 = *long*

This command distorts the current image in a manner defined by the table values set by the EffectShearTable command. An EffectShear command block must contain an EffectShearTable command for each of the 1024 points in the shear table, and must end with an EndEffectShear command.

Syntax	Description
.Scale	Specifies the degree to which the image conforms to the curve. Set the value at 100% to have the image conform completely to the curve. Valid values range from 0 to 100%.
.Border	Specifies the method to use when filling the empty space left by the shear command: 0 = Wraps the opposite end of the image around to the empty space 1 = Stretches the image to fill the space 2 = Fills empty areas with the current paint color
.Orientation	Specifies the direction of the shear curve: 0 = Horizontal 1 = Vertical
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EffectShear 50, 2, 1, 5, 255, 51, 0, 0
        .EffectShearTable 0, 512
        .EffectShearTable 1, 517
        ....
        .EffectShearTable 1023, 512
        .EndEffectShear
```

This example applies a shear effect to the active image and specifies red for the undefined areas.

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

EffectShearTable (PHOTO-PAINT)

.EffectShearTable .Number = *long*, .Value = *long*

This command sets the shear table values for the EffectShear command. An EffectShear command block must contain an EffectShearTable command for each of the 1024 points in the shear table.

Syntax	Description
.Number	Specifies the index number of the point on the shear curve. Valid values range from 0 to 1023.
.Value	Specifies the displacement of the shear point. Valid values range from 0 to 1023.

Example

```
.EffectShear 50, 0, 1
    .EffectShearTable 0, 264
    .EffectShearTable 1, 265
    .EffectShearTable 2, 267
    .EffectShearTable 3, 269
    ...
    .EffectShearTable 1021, 779
    .EffectShearTable 1022, 780
    .EffectShearTable 1023, 781
    .EndEffectShear
```

This example applies a shear effect to the active image.

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

EffectSmokedGlass (PHOTO-PAINT)

.EffectSmokedGlass .Tint = *long*, .Percent = *long*, .ColorModel = *long*, .Color1 = *long*, .Color2 = *long*, .Color3 = *long*, .Color4 = *long*

This command applies a transparent, colored tint over the image.

Syntax	Description
.Tint	Specifies the opacity of the tint. Valid values range from 0 to 100%.
.Percent	Specifies the amount of blurring to use to create the glass effect. Valid values range from 0 to 100%.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

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

EffectSmooth (PHOTO-PAINT)

.EffectSmooth .Percentage = *long*

This command tones down differences in adjacent pixels resulting in only a slight loss of detail, while smoothing the overall image or selected area.

Syntax	Description
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectSoften (PHOTO-PAINT)

.EffectSoften .Percentage = *long*

This command smoothes and tones down harsh edges with only minimal loss of image detail.

Syntax	Description
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.

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

EffectSolarize (PHOTO-PAINT)

.EffectSolarize .Level = *long*

This command transforms colors to appear like those of a negative photographic image.

Syntax	Description
.Level	Specifies the intensity of the effect. Valid values range from 0 to 255.

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

EffectSwirl (PHOTO-PAINT)

.EffectSwirl .Angle = *long,* .CenterX = *long,* .CenterY = *long*

This command creates a swirling vortex of distortion on your image.

Syntax	Description
.Angle	Specifies the angle through which the swirl occurs. Values range from -3600 to 3600.
.CenterX	Specifies the horizontal position of the swirl's center.
.CenterY	Specifies the vertical position of the swirl's center.

Example

```
.EffectSwirl -719, 233, 121
```

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

EffectTheBoss (PHOTO-PAINT)

.EffectTheBoss .Angle = *long*, .BevelWidth = *long*, .BevelHeight = *long*, .Brightness = *long*, .Direction = *long*, .Dropoff = *long*, .Sharpness = *long*, .Smooth = *long*

This command applies The Boss effect to an image. You must create a mask before applying this effect.

Syntax	Description
.Angle	Specifies the angle at which the light bounces off the bevel.
.BevelWidth	Specifies the width of the bevel.
.BevelHeight	Specifies the height of the bevel.
.Brightness	Specifies the intensity of the light source.
.Direction	Specifies the angle at which the light hits the bevel.
.Dropoff	Specifies the edges style of the bevel.
.Sharpness	Specifies the amount of fading at the edge of the light shaft. A lower value results in a concentrated light source (like a flashlight). A higher value results in a softer, larger light source (like a ceiling light).
.Smooth	Specifies the smoothness of the bevel.

Example

```
.EffectTheBoss 45, 20, 30, 60, 135, 0, 0, 30
```

This example applies The Boss effect to an image.

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

EffectTile (PHOTO-PAINT)

.EffectTile .Horizontal = *long*, .Vertical = *long*

This command reduces the dimensions of your image and reproduces the image as a series of tiles on a grid.

Syntax	Description
.Horizontal	Specifies the number of times the image appears along the horizontal axis.
.Vertical	Specifies the number of times the image appears along the vertical axis.

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

EffectTool (PHOTO-PAINT)

.EffectTool .BrushID = *long*, .TypeID = *long*, .MergeMode = *long*, .Amount = *long*, .NibShape = *long*, .Size = *long*, .Transparency = *long*, .Rotate = *long*, .Flatten = *long*, .SoftEdge = *long*

This command applies an effect stroke along a path defined by a series of StartDraw and ContinueDraw commands. An EffectTool command block must end with an EndDraw command.

Syntax	Description
.BrushID	Specifies the brush to use by index number.
.TypeID	Specifies the type of brush stroke by index number.
.MergeMode	Specifies the Merge Mode: 0 = Normal 1 = Add 2 = Subtract 3 = Difference 4 = Multiply 5 = Divide 6 = Lighter 7 = Darker 8 = Texturize 9 = Color 10 = Hue 11 = Saturation 12 = Lum 13 = Invert 14 = And 15 = Or 16 = Xor 17 = Red 18 = Green 19 = Blue 20 = Cyan 21 = Magenta 22 = Yellow 23 = Black
.Amount	Specifies the rate at which the effect or paint is applied to the image, ranging from 1 to 100. A higher value results in a more pronounced effect or heavier application of paint.
.NibShape	Specifies the shape of the nib: 0 = Round 1 = Square
.Size	Specifies the width of the nib. Valid values range from 1 to 999 pixels.
.Transparency	Specifies the transparency of the brush stroke. Valid values range from 0 to 99%.
.Rotate	Specifies the angle at which the nib is rotated. Valid values range from 0 to 360 degrees.
.Flatten	Specifies the flatness of the nib. Valid values range from 0 to 99.
.SoftEdge	Specifies the transparency of the nib's edges. As you increase the value of this setting, the soft edge expands to eventually reach the center of the stroke. Low values affect only the rim of the stroke. Valid values range from 0 to 100%.

Example

```
.EffectTool 7, 0, 0, 51, 0, 30, 0, 0, 100, 50
    .BrushTextureSettings "", 0, 0, 0, 0, 1, TRUE, FALSE
    .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
    .SetPaintColor 5, 186, 159, 106, 0
    .RandomSeed 1691418496
    .StartDraw 27776, 12160, 0, 0
    .ContinueDraw 28779, 13945, 0, 0
    ...
    .ContinueDraw 36293, 64568, 0, 0
    .EndDraw
```

This example applies an effect stroke along the defined path.

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

EffectTraceContour (PHOTO-PAINT)

.EffectTraceContour .Level = *long*, .EdgeType = *long*

This command creates edges of different intensity by tracing image elements using the 16 colors of the standard VGA palette.

Syntax	Description
.Level	Specifies the brightness threshold for outlining. Valid values range from 1 to 255.
.EdgeType	Specifies the type of edges to trace: 0 = Traces the areas of your image where the brightness levels of the pixels fall below the .Level value 1 = Traces the areas of your image where the brightness level of the pixels exceeds the .Level value

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

EffectUnsharpMask (PHOTO-PAINT)

.EffectUnsharpMask .Radius = *long*, .Percentage = *long, .*Threshold = *long*

This command accentuates edge detail as well as focusing some blurred areas in the image.

Syntax	Description
.Radius	Specifies the range of the effect. Larger values increase the number of pixels that are successively selected and evaluated when you apply the effect. Valid values range from 1 to 20.
.Percentage	Specifies the intensity of the effect. Valid values range from 0 to 100%.
.Threshold	Specifies the change in value that is required by a pixel before the Unsharp Mask effect is applied.

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

EffectUserDefined (PHOTO-PAINT)

.EffectUserDefined .Divisor = *long*, .Offset = *long*

This command applies a custom convolution to the current image. An EffectUserDefined command block must contain an EffectUserDefinedPoint command for each of the 25 points in the convolution matrix, and must end with an EndEffectUserDefined command.

Syntax	Description
.Divisor	Specifies the divisor value. After the command multiplies each matrix value by the brightness value of the corresponding pixel, it adds the products together, and then divides the sum by the value you type in the Divisor box.
.Offset	Specifies the offset value. This is the value that will be added to the final pixel values just before the effect is applied.

Note

.EffectUserDefined starts the command block, .EffectUserDefinedPoint fills the user array, and .EndEffectUserDefined ends the command.

Example

```
.EffectUserDefined 1, 0
    .EffectUserDefinedPoint 0, 1
    .EffectUserDefinedPoint 1, 1
    .EffectUserDefinedPoint 2, 1
    .EffectUserDefinedPoint 3, 1
    ...
    .EffectUserDefinedPoint 24, 1
    .EndEffectUserDefined
```

This example fills the user array with values of 1, which will give the effect of smoothing the image.

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

EffectUserDefinedPoint (PHOTO-PAINT)

.EffectUserDefinedPoint .Index = *long*, .Value = *long*

This command defines a point in the convolution matrix of an EffectUserDefined command. An EffectUserDefined command block must contain an EffectUserDefinedPoint command for each of the 25 points in the convolution matrix.

Syntax	Description
.Index	Specifies the index of the matrix point to define. Valid values range from 0 to 24.
.Value	Specifies the value of the specified matrix entry. Valid values range from -999 to 999.

Note

.EffectUserDefined starts the command block, .EffectUserDefinedPoint fills the user array, and .EndEffectUserDefined ends the command.

Example

```
.EffectUserDefined 1, 0
    .EffectUserDefinedPoint 0, 1
    .EffectUserDefinedPoint 1, 1
    .EffectUserDefinedPoint 2, 1
    .EffectUserDefinedPoint 3, 1
    ...
    .EffectUserDefinedPoint 24, 1
    .EndEffectUserDefined
```

This example fills the user array with values of 1, which will give the effect of smoothing the image.

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

EffectVignette (PHOTO-PAINT)

.EffectVignette .Shape = *long*, .Offset = *long*, .Fade = *long,* .ColorModel = *long,* .Color1 = *long,* .Color2 = *long,* .Color3 = *long,* .Color4 = *long*

This command creates a frame around your image.

Syntax	Description
.Shape	Specifies the frame shape: 0 = Elliptical 1 = Circular 2 = Rectangular 3 = Square
.Offset	Specifies the size of the frame. Valid values range from 0 to 140.
.Fade	Specifies the fade rate between the image and the frame. Valid values range from 0 to 100.
.ColorModel	Specifies the Color Model to use: 3 = CMYK (Cyan, Magenta, Yellow, Black) 5 = RGB (Red, Green, Blue) 8 = Black and White 9 = Grayscale
.Color1	Specifies the first color component for .ColorModel. For example, Red is the first color component for RGB.
.Color2	Specifies the second color component for .ColorModel. For example, Green is the second color component for RGB. 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, Blue is the third color component for RGB. 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. If this parameter is not available in the Color Model specified, set it to 0.

Example

```
.EffectVignette 1, 109, 33, 5, 0, 255, 0, 0
```

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

EffectWetPaint (PHOTO-PAINT)

.EffectWetPaint .Wetness = *long*, .Percentage = *long*

This command creates the illusion that your image is a painting that is still wet. The effects can range from subtle changes in the luminescence of colors to streaks of wet paint dripping down your image

Syntax	Description
.Wetness	Specifies the range of colors that drip. Negative values cause the dark colors to drip, positive values cause the light colors to drip. Valid values range from -50 to 50.
.Percentage	Specifies the size of the paint drip. Valid values range from 0 to 100%.

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

EffectWhirlpool (PHOTO-PAINT)

.EffectWhirlpool .Spacing = *long*, .Smear = *long*, .Twist = *long*, .Streak = *long*, .Warp = *boolean*

This command applies a pattern of fluid streamlines over your image.

Syntax	Description
.Spacing	Specifies the spacing between swirls. Valid values range from 5 to 200.
.Smear	Specifies the length of the swirls. Valid values range from 3 to 30.
.Twist	Specifies the degree of curvature. Valid values range from 0 to 90.
.Streak	Specifies the intensity of the swirls. Valid values range from 0 to 100.
.Warp	Set to TRUE (-1) if you want the whirlpool effect to distort the image. Set to FALSE (0) if you want the effect to overlay the image.

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

EffectWind (PHOTO-PAINT)

.EffectWind .Strength = *long*, .Opacity = *long*, .Direction = *long*

This command blurs your image in a specific direction, creating the effect of wind blowing across your image.

Syntax	Description
.Strength	Specifies the intensity of the effect. Valid values range from 0 to 100%.
.Opacity	Specifies the opacity of the effect. Valid values range from 1 to 100%.
.Direction	Specifies the direction of the blur. Valid values range from 0 to 360 degrees.

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

EffectZigZag (PHOTO-PAINT)

.EffectZigZag .Period = *long*, .Strength = *long*, .Damping = *long*, .Type = *long,* .CenterX = *long,* .CenterY = *long*

This command distorts an image by bending the image lines that run from the center of the image to its edge. This effect produces waves of straight lines and angles which seem to twist the image from its center outwards.

Syntax	Description
.Period	Specifies the period of the waves. Valid values range from 1 to 100.
.Strength	Specifies the intensity of the distortion. Valid values range from 1 to 100.
.Damping	Specifies the degree of damping in successive waves. Large values cause the distortion waves to phase out toward the edges of your image, Smaller values cause the waves to extend toward the edges. Valid values range from 1 to 100.
.Type	Specifies the type of wave: 0 = Pond ripples 1 = Out from center 2 = Around center
.CenterX	Specifies the horizontal position of the zig zag effect.
.CenterY	Specifies the vertical position of the zig zag effect.

Example

```
.EffectZigZag 3, -73, 20, 1, 99, 100
```

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

EndAdjustEffect (PHOTO-PAINT)

.EndAdjustEffect

This command ends an AdjustEffectInfo command block.

Example

```
.EffectAdjustBlur
    .AdjustEffectInfo 4, 10, 0, 0
    .AdjustEffectInfo 20, 10, 0, 0
    .AdjustEffectInfo 20, 10, 69, 0
    .AdjustEffectInfo 1, 10, 69, 0
    .EndAdjustEffect
```

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

EndColorEffect (PHOTO-PAINT)

.EndColorEffect

This command ends nine effects commands whether they are applied to the image itself or to a lens object (lenses are only available in the Corel PHOTO-PAINT 7 Plus standalone program). The effect commands that use this command are: .ImageReplaceColors, .ImagePosterize, .ImageLevelThreshold, ImageInvert, ImageColorBalance, .ImageBCI, .ImageHSL, .ImageDesaturate, and .ImageGamma.

Example

```
.ImageLevelThreshold 0, 179, 255, 255,0
    .EndColorEffect
```

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

EndEffectLighting (PHOTO-PAINT)

.EndEffectLighting

This command ends an EffectLighting command block.

Example

```
.EffectLighting 4
    .EffectLightSource 0, 0.5, 0.5, 0, 75, 180, 90, 40, 180, 100, 0, 100, 0, 0, 0, 255, 0, 0
    .EffectLightSource 1, 0.097166, 1.09717, 1, 60, 15, 25, 95, 45, 80, 0, 100, 0, 0, 0, 255,
255, 51
    .EffectLightSource 2, 0.433, 1.1, 1, 60, 15, 25, 95, 45, 80, 0, 100, 0, 0, 0, 204, 102, 255
    .EffectLightSource 3, 0.766, 1.1, 1, 60, 15, 25, 95, 45, 80, 0, 100, 0, 0, 0, 255, 255, 255
    .EndEffectLighting
```

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

EndEffectMeshWarp (PHOTO-PAINT)

.EndEffectMeshWarp

This command ends an EffectMeshWarp command block.

Example

```
.EffectMeshWarp 5, 5
    .EffectMeshPoint 0, 0, 0, 0
    .EffectMeshPoint 0, 1, 0.25, 0
    .EffectMeshPoint 0, 2, 0.5, 0
    .EffectMeshPoint 0, 3, 0.75, 0
    .EffectMeshPoint 0, 4, 1, 0
    .EffectMeshPoint 1, 0, 0, 0.25
    .EffectMeshPoint 1, 1, 0.340067, 0.345946
    .EffectMeshPoint 1, 2, 0.52862, 0.297297
    .EffectMeshPoint 1, 3, 0.861953, 0.145946
    .EffectMeshPoint 1, 4, 1, 0.25
    .EffectMeshPoint 2, 0, 0, 0.5
    .EffectMeshPoint 2, 1, 0.25, 0.5
    .EffectMeshPoint 2, 2, 0.579125, 0.556757
    .EffectMeshPoint 2, 3, 0.75, 0.5
    .EffectMeshPoint 2, 4, 1, 0.5
    .EffectMeshPoint 3, 0, 0, 0.75
    .EffectMeshPoint 3, 1, 0.121212, 0.859459
    .EffectMeshPoint 3, 2, 0.5, 0.75
    .EffectMeshPoint 3, 3, 0.818182, 0.881081
    .EffectMeshPoint 3, 4, 1, 0.75
    .EffectMeshPoint 4, 0, 0, 1
    .EffectMeshPoint 4, 1, 0.25, 1
    .EffectMeshPoint 4, 2, 0.5, 1
    .EffectMeshPoint 4, 3, 0.75, 1
    .EffectMeshPoint 4, 4, 1, 1
    .EndEffectMeshWarp
```

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

EndEffectShear (PHOTO-PAINT)

.EndEffectShear

This command ends an EffectShear command block.

Example

```
.EffectShear 50, 0, 1
    .EffectShearTable 0, 264
    .EffectShearTable 1, 265
    .EffectShearTable 2, 267
    .EffectShearTable 3, 269
    ...
    .EffectShearTable 1021, 779
    .EffectShearTable 1022, 780
    .EffectShearTable 1023, 781
    .EndEffectShear
```

This example applies a shear effect to the active image.

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

EndEffectUserDefined (PHOTO-PAINT)

.EndEffectUserDefined

This command ends an EffectUserDefined command block.

Note

.EffectUserDefined starts the command block, .EffectUserDefinedPoint fills the user array, and .EndEffectUserDefined ends the command.

Example

```
.EffectUserDefined 1, 0
    .EffectUserDefinedPoint 0, 1
    .EffectUserDefinedPoint 1, 1
    .EffectUserDefinedPoint 2, 1
    .EffectUserDefinedPoint 3, 1
    ...
    .EffectUserDefinedPoint 24, 1
    .EndEffectUserDefined
```

This example fills the user array with values of 1, which will give the effect of smoothing the image.

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

Path commands (PHOTO-PAINT)

PathCreate (PHOTO-PAINT)

.PathCreate .Nodes = *long*

This command creates a new path using nodes defined by a series of PathNode commands. A PathCreate command block must end with a PathEnd command.

Syntax	Description
.Nodes	Specifies the number of nodes used to define the path.

Example

```
.PathCreate 10
    .PathNode 36, 32, FALSE, 0, 0
    .PathNode 36, 32, FALSE, 0, 3
    .PathNode 15, 86, FALSE, 0, 3
    .PathNode 88, 118, FALSE, 2, 2
    .PathNode 161, 150, FALSE, 1, 3
    .PathNode 110, 108, FALSE, 1, 3
    .PathNode 179, 136, FALSE, 2, 2
    .PathNode 248, 164, FALSE, 1, 3
    .PathNode 237, 219, FALSE, 1, 3
    .PathNode 237, 219, FALSE, 1, 2
    .PathEnd
```

This example creates a ten-node path.

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

PathCreateFromMask (PHOTO-PAINT)

.PathCreateFromMask .Tightness = *long*, .Threshold = *long*

This command creates a path that has the shape of the current mask marquee.

Syntax	Description
.Tightness	Specifies the number of nodes for the new path.
.Threshold	Specifies the angle size required between sections of the selection's boundary for a node top be placed at the intersection of the sections.

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

PathDelete (PHOTO-PAINT)

.PathDelete .FromDisk = *boolean*

This command deletes the selected path.

Syntax	Description
.FromDisk	Set to TRUE (-1) to also delete the file containing a saved path.

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

PathEnd (PHOTO-PAINT)

.PathEnd

This command ends a PathCreate command block.

Example

```
.PathCreate 10
    .PathNode 36, 32, FALSE, 0, 0
    .PathNode 36, 32, FALSE, 0, 3
    .PathNode 15, 86, FALSE, 0, 3
    .PathNode 88, 118, FALSE, 2, 2
    .PathNode 161, 150, FALSE, 1, 3
    .PathNode 110, 108, FALSE, 1, 3
    .PathNode 179, 136, FALSE, 2, 2
    .PathNode 248, 164, FALSE, 1, 3
    .PathNode 237, 219, FALSE, 1, 3
    .PathNode 237, 219, FALSE, 1, 2
    .PathEnd
```

This example creates a ten-node path.

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

PathImportVector (PHOTO-PAINT)

.PathImportVector .FileName = *string,* .ScaleX = *double,* .ScaleY = *double*

This command imports vector images as paths. This command can only load vector file formats.

Syntax	Description
.FileName	Specifies the name of the vector to import.
.ScaleX	Specifies the horizontal scaling factor. A vector image can be scaled to any size on import.
.ScaleY	Specifies the vertical scaling factor. A vector image can be scaled to any size on import.

Example

```
.PathImportVector sample.cdr 1.5, 2
```

This example imports a CorelDRAW vector image named "sample" as a path.

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

PathLoad (PHOTO-PAINT)

.PathLoad .PathName = *string,* ScaleX = *double,* ScaleY = *double*

This command opens a path that has been saved to disk.

Syntax	Description
.PathName	The name of the file containing the saved path.
.ScaleX	Specifies the horizontal scaling factor. A vector image can be scaled to any size on import.
.ScaleY	Specifies the vertical scaling factor. A vector image can be scaled to any size on import.

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

PathNew (PHOTO-PAINT)

.PathNew

This command deletes the existing path so that you may create a new one.

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

PathNode (PHOTO-PAINT)

.PathNode .ptX = *long*, .ptY = *long*, .Closed = *boolean*, .Continuity = *long*, .Type = *long*

This command defines a single node in a path for the PatchCreate command.

Syntax	Description
.ptX	Specifies the horizontal coordinate of the node point.
.ptY	Specifies the vertical coordinate of the node point.
.Closed	Set to TRUE (-1) if you want the current point to close the path.
.Continuity	Specifies how the curve is derived: 0 = First order integral 1 = Second order integral
.Type	Specifies the node type: 0 = Line 1 = Curve 2 = Normal 3 = Control point

Example

```
.PathCreate 10
    .PathNode 36, 32, FALSE, 0, 0
    .PathNode 36, 32, FALSE, 0, 3
    .PathNode 15, 86, FALSE, 0, 3
    .PathNode 88, 118, FALSE, 2, 2
    .PathNode 161, 150, FALSE, 1, 3
    .PathNode 110, 108, FALSE, 1, 3
    .PathNode 179, 136, FALSE, 2, 2
    .PathNode 248, 164, FALSE, 1, 3
    .PathNode 237, 219, FALSE, 1, 3
    .PathNode 237, 219, FALSE, 1, 2
    .PathEnd
```

This example creates a ten-node path.

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

PathSave (PHOTO-PAINT)

.PathSave .PathName = *string*

This command saves the existing path to disk so that you may use it in the future in any image. Paths are given a .PTH file extension.

Syntax	Description
.PathName	Specifies the filename of the path you want to save.

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

PathStroke (PHOTO-PAINT)

.PathStroke

This command ends any of the CloneTool, BrushTool, or EffectTool command blocks.

Example

```
.BrushTool 0, 0, 0, 20, 0, 5, 0, 0, 100, 0
    .BrushTextureSettings "", 0, 0, 0, 0, 0, TRUE, FALSE
    .BrushDabSettings 1, 25, 0, 0, 0, 0, 0
    .SetPaintColor 5, 186, 159, 106, 0
    .PathStroke
```

This example applies a brush stroke to the current image.

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

No related topics were found.

No topics were found.

This area is presently under construction