Microsoft Office 97-2007

Office Drawing 97-2007 Binary Format Specification

Contents

Introduction

This document describes the file format of the Office Drawing Layer (Escher). With this document and knowledge of the host application's file format, the reader should be able to construct and interpret an Escher file stream.

Object Container Hierarchy

Escher has an object containership hierarchy similar to other drawing programs. At the root of the hierarchy is a drawing group object. There is one drawing group per client document. Drawing groups contain drawings. Drawings in turn contain shapes that are the objects that actually mark a page. Next to the drawings in a drawing group is a collection that contains the images and pictures used by the drawings. Escher keeps pictures in a separate collection to be able to incrementally load and save them.

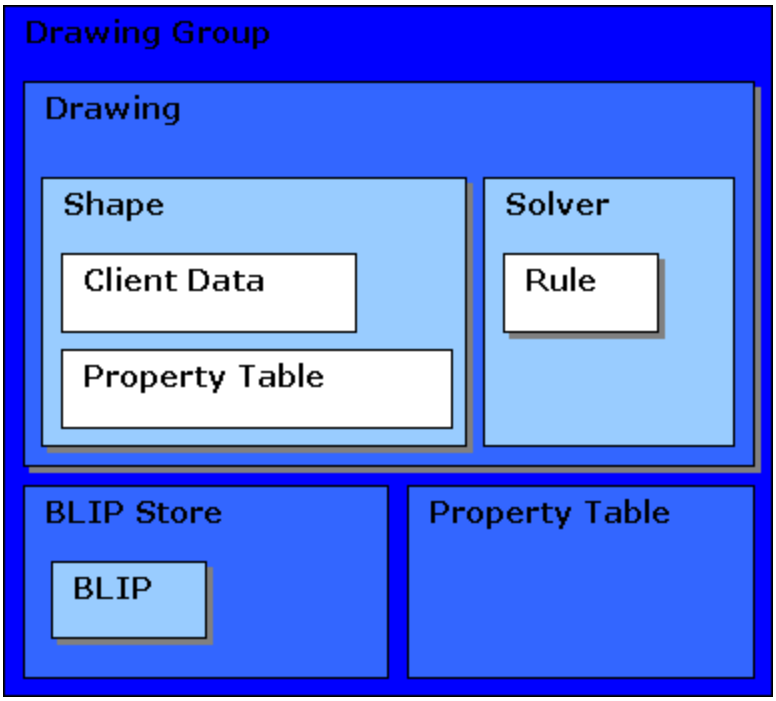


Figure 1 -- Containership Hierarchy

A few other points are worth noting

Associated with each shape is a piece of client data that keeps the shape's anchor, text and OLE data, as well as host specific properties. The format of this structure is host-defined.

Shapes store their properties in a separate structure called a property table. The property table is basically a sorted list of property id-value pairs.

Each drawing group has a shape property table that stores the defaults for new shapes.

Each drawing has a collection of rules that govern the shapes in the drawing.

This object hierarchy does **not** exactly correspond to the record hierarchy in the file format. In the file format, drawings are not saved inside drawing groups, but in separate top-level containers. In this scheme, hosts can save drawing group information with per-document information, and save drawing information with per-sheet, per-slide, or per-page information.

Use by the Host Application

Escher is part of the Office DLL and is used by PowerPoint, Word, FrontPage, Publisher, and Excel. FrontPage does not save the Office Drawing file format into files; it emits and consumes the Office Drawing file format only when interacting with the clipboard. The other client applications use the Office Drawing file format when saving data into their binary file format. When serializing Escher data into a file, each client application provides an OLE IStream interface to Escher. This document describes the records Escher writes to this interface. However, it doesn't describe how the actual bytes are saved to disk. The client controls this format, since Escher writes to a client provided interface. Excel, for example, places the bytes of the Escher stream into BIFF records.

Records

The Escher file stream is a series of records that share a common header structure. Records can be categorized in to two groups.

Atoms Records that contain information about an Escher object and are kept inside containers.

Containers Records that keep atoms and other containers in a logical and organized way.

Each record, whether it's an atom or a container, has a common header. Container records are just the common header, while the atom records are the common header followed by some record specific data. Escher uses the structure of container records containing atom records and other container records throughout its stream.

Common Header

The common record header is an 8-byte structure defined as follows:

```
typedef struct MSOFBH
   {
   struct
      {
      ULONG ver : 4;
      ULONG    inst: 12;
      ULONG    fbt : 16;
      };
   ULONG    cbLength;
   } MSOFBH;
```

The fields are:

Record Type (fbt) Indicates the signature or type of the record. Each record has a symbolic and a numeric signature. Escher uses values from 0xF000 to 0xFFFF. Clients may define their own records in other ranges. A description of each of the different types can be found in the following sections.

Record Instance (inst) Differentiates atoms. Depending on the instance a record's contents it can have different meanings. For example a list container can store a list of slides or a list of fonts, and its instance would vary accordingly. The instance of a record is useful for differentiating atoms when there is more than one atom of the same type in a particular container

Record Version (ver) Indicates the version if the record is an atom. If the record is a container, this field has a value of 0xFFFF.

Record Length (cbLength) Stores the length of the record in bytes. If the record is an atom, it refers to the length of the atom excluding the header. If the record is a container, it refers to the sum of the lengths of the atoms inside it, plus the length of the record headers.

Notes for Implementers

The common header specifies the length of each record. Consequently, it is possible to parse the Escher record stream without knowledge of the actual contents of each record. The Escher team intends to take advantage of this fact in future versions. As new features are added, Escher will define new record types. Readers of the Escher file format should skip over record types unknown to the reader. In addition, readers should not expect a record to come in a certain order in a container. They can, however, expect that the containership hierarchy will not change. For example, readers do not need to handle the case of a shape record containing a drawing record.

When Escher writes to a client file, it stores client-specific records in its stream to preserve the client features and behaviors. On the other hand, when Escher writes to a clipboard stream, it uses a client-independent form of the file format to allow interchange between applications.

Escher saves records in Intel byte-order even on the Macintosh. The Macintosh version of Escher byte-swaps the records as they are loaded and as they are saved. Records are tightly packed, without alignment. The LONG type is 32 bits in length.

Pointers

The general problem of saving pointers to objects in the file format is solved in ordinary fashion by giving objects unique identifiers, which are saved in the file format in place of the pointer values. At load time, these IDs are converted back into pointers.

The most common instances of this are pointers to shapes, which are saved as shape IDs, or SPIDs. SPIDs are unique per drawing group, and are parceled out by the drawing group to individual drawings in blocks of 1024. The drawing group keeps a table recording which drawing owns which block of SPIDs, so that, given a SPID, it is easy to determine which drawing the shape is in. That table makes up the bulk of the msofbtDgg record, and is the only place where pointers to drawings are saved (as DGIDs).

Record Name	Word	Excel	PorwerPoint	FBT value	Version	Instance	Contents
msofbtDggContainer	✓	✓	✓	F000			per-document data
msofbtDgg	✓	✓	✓	F006	0		an FDGG and several FIDCLs
msofbtCLSID	C	C	C	F016	0		the CLSID of the application that put the data on the clipboard
msofbtOPT	✓	✓	✓	F00B	3	count of properties	the document-wide default shape properties (Block 1)
msofbtTertiaryOPT	✓	✓	✓	F122	3	count of properties	the document-wide default shape properties (Block 3)
msofbtColorMRU	✓	✓	✓	F11A	0	count of colors	the colors in the MRU swatch
msofbtSplitMenuColors	✓	✓	✓	F11E	0	count of colors	the colors in the top-level split menus
msofbtBstoreContainer	✓	✓	✓	F001		count of BLIPs	all images in the document (JPEGs, metafiles, etc.)
msofbtBSE	✓	✓	✓	F007	2	BLIP type	an FBSE (one per BLIP)
msofbtBlip***	✓	✓	✓	F018 - F117			range of fbts reserved for various kinds of BLIPs
msofbtDgContainer	✓	✓	✓	F002			per-sheet/page/slide data
msofbtDg	✓	✓	✓	F008	0	drawing ID	an FDG
msofbtRegroupItems	✓	✓	✓	F118	0	count of regroup entries	several FRITs
msofbtColorScheme		C	C	F120	0	count of colors	the colors of the source host's color scheme
msofbtSpgrContainer	✓	✓	✓	F003			several SpContainers, the first of which is the group shape itself
msofbtSpContainer	✓	✓	✓	F004			a shape
msofbtSpgr	✓	✓	✓	F009	1		an FSPGR; only present if the shape is a group shape
msofbtSp	✓	✓	✓	F00A	2	shape type	an FSP
msofbtOPT	✓	✓	✓	F00B	3	count of properties	a shape property table (Block 1)
msofbtSecondaryOPT	✓	✓	✓	F121	3	count of properties	a shape property table (Block 2)
msofbtTertiaryOPT	✓	✓	✓	F122	3	count of properties	a shape property table (Block 3)

		msofbtTextbox	C	C	C	F00C	0		RTF text
		msofbtClientTextbox	✓	✓	✓	F00D		host-defined	the text in the textbox, in a host-defined format
		msofbtAnchor	C	C	C	F00E	0		a RECT, in 100000ths of an inch
		msofbtChildAnchor	✓	✓	✓	F00F	0		a RECT, in units relative to the parent group
		msofbtClientAnchor	✓	✓	✓	F010		host-defined	the location of the shape, in a host-defined format
		msofbtClientData	✓	✓	✓	F011		host-defined	host-specific data
		msofbtOleObject	C	C	C	F11F	0		a serialized IStorage for an OLE object
		msofbtDeletedPspl	✓			F11D	0		an FPSPL; only present in top-level deleted shapes
	msofbtSolverContainer		✓	✓	✓	F005		count of rules	the rules governing shapes
		msofbtConnectorRule		✓	✓	F012	1		an FConnectorRule
		msofbtAlignRule	✓	✓	✓	F013	0		an FAlignRule
		msofbtArcRule	✓	✓	✓	F014	0		an FARCRU
		msofbtClientRule				F015		host-defined	host-defined
		msofbtCalloutRule	✓	✓	✓	F017	0		an FCORU
msofbtSelection				✓		F119	0		an FDGSL followed by the SPIDs of the shapes in the selection

Drawing Group Container

msofbtDggContainer

Drawing Group Record

msofbtDgg

The drawing group record is a variable length record consisting of a fixed part followed by an array. The fixed part is defined as follows.

```
// FDGG - File DGG
typedef struct _FDGG
   {
   MSOSPID spidMax;  // The current maximum shape ID
   ULONG cidcl;      // The number of ID clusters (FIDCLs)
   ULONG cspSaved;   // The total number of shapes saved
         // (including deleted shapes, if undo
         // information was saved)
   ULONG cdgSaved;   // The total number of drawings saved
   } FDGG;
```

The fixed part is followed by an array of ID clusters. The ID clusters are used internally for the translation of shape ids (SPIDs) to shape handles (MSOHSPs).

```
// File ID Cluster - used to save IDCLs
typedef struct _FIDCL
   {
   MSODGID dgid;  // DG owning the SPIDs in this cluster
   ULONG cspidCur;  // number of SPIDs used so far
   } FIDCL;
```

Class ID Record

msofbtCLSID

The class ID record is only present in the clipboard format. It just contains an OLE CLSID record from the source application, and is used by the destination application to check where the clipboard data originated.

Default Property Table Records

msofbtOPT, msofbtTertiaryOPT

This describes the default properties of newly created shapes. Only the properties that differ from the per-property defaults are saved. The format of the record is the same as that of the property table in a shape, except that Block 2 (msofbtSecondaryOPT) is not allowed in the defaults. A discussion of that format is in the Shape Properties section.

Color MRU Record

msofbtColorMRU

The Color MRU record contains the colors in the most-recently-used-colors swatch that appears at the bottom of color dropdowns. The instance field contains the number of colors; the data of the record contains the colors in order from left to right.

Split Menu Colors Record

msofbtSplitMenuColors

The single MRU colors of the top-level Fill Color, Line Color, Shadow Color, and 3D Color split menus are saved to a SplitMenuColors record in that order, with the number of colors (currently always four) in the instance field.

BStore Container

msofbtBstoreContainer

The images and pictures in a drawing can dominate the size of a drawing. Consequently, Escher handles these objects in a special way. As an abstraction, Escher names these objects BLIPs for Big Large Image or Picture. In Office 2003, there are eight types of blips supported in Escher: Windows Metafiles, Enhanced Metafiles, JPEG Interchange Format, Device Independent Bitmap (DIB,) Tag Image File Format (TIFF,) Portable Network Graphics (PNG,) Graphic interchange format (GIF,) and Macintosh PICT. Implementers should note that some of these types cannot be processed by Office 2000, and even more cannot be processed by Office 97. Additional types may be permitted in future versions.

Escher stores all the BLIPs in a document in a separate container called the BStore. It reference counts the BLIPs, so that if a picture is inserted multiple times in a document it is only stored once in the BStore but is multiply referenced by different shapes.

The host may choose to store the blip data in a separate delay stream. If a delay stream is used, Escher can incrementally load the blips as they are displayed, not when the document is loaded. (As of Office 97, Word and PowerPoint use a delay stream, and Excel does not.)

The BStore container is just an array of Blip Store Entry (BSE) records. Each shape stores indices into the array for the BLIPs they use. BLIPs are used not only for inserted pictures, but also for the textured and pictures fills of the shape.

BLIP Store Entry Record

msofbtBSE

Each BLIP in the BStore is serialized to a File BLIP Store Entry (FBSE) record. The instance field encodes the type of the blip. A fixed size header contains the rest of the common information about the BLIP. If the cbName field in the FBSE is nonzero, a null-terminated Unicode string is written immediately after the FBSE in the file.

```
// FBSE - File Blip Store Entry
typedef struct _FBSE
   {
   BYTE     btWin32;    // Required type on Win32
   BYTE     btMacOS;    // Required type on Mac
   BYTE      rgbUid[16]; // Identifier of blip
   WORD     tag;          // currently unused
   ULONG    size;         // Blip size in stream
   ULONG    cRef;         // Reference count on the blip
   MSOFO    foDelay;   // File offset in the delay stream
   BYTE     usage;        // How this blip is used (MSOBLIPUSAGE)
   BYTE     cbName;    // length of the blip name
   BYTE     unused2;   // for the future
   BYTE     unused3;   // for the future
   } FBSE;
```

```
typedef enum
   {
   msoblipUsageDefault,  // All non-texture fill blips get this.
   msoblipUsageTexture,
   msoblipUsageMax = 255 // Since this is stored in a byte
   } MSOBLIPUSAGE;

typedef enum
   {                    // GEL provided types...
   msoblipERROR = 0,       // An error occured during loading
   msoblipUNKNOWN,      // An unknown blip type
   msoblipEMF,          // Windows Enhanced Metafile
   msoblipWMF,          // Windows Metafile
   msoblipPICT,         // Macintosh PICT
   msoblipJPEG,         // JFIF
   msoblipPNG,             // PNG or GIF
   msoblipDIB,             // Windows DIB
   msoblipTIFF = 17,    // TIFF
   msoblipCMYKJPEG = 18,// JPEG data in YCCK or CMYK color space
   msoblipFirstClient = 32,   // First client defined blip type
   msoblipLastClient  = 255   // Last client defined blip type
   } MSOBLIPTYPE;

typedef enum
   {
   msobiUNKNOWN = 0,
   msobiWMF  = 0x216,  // Metafile header then compressed WMF
   msobiEMF  = 0x3D4,  // Metafile header then compressed EMF
   msobiPICT = 0x542,  // Metafile header then compressed PICT
   msobiPNG  = 0x6E0,  // One byte tag then PNG data
   msobiJFIF = 0x46A,  // One byte tag then JFIF data
   msobiJPEG = msobiJFIF,
   msobiDIB  = 0x7A8,  // One byte tag then DIB data
   msobiCMYKJPEG = 0x6E2, // One byte tag then CMYK/YCCK JPEG data
   msobiTIFF = 0x6e4, // One byte tag then TIFF data
   msobiClient=0x800,  // Clients should set this bit
   }
MSOBI;                  // Blip signature as encoded in the MSOFBH.inst
```

The btWin32 and btMacOS fields store the MSOBLIPTYPE for the respective operating systems. When the OS blip type doesn't match the blip type of stored, Escher will attempt to convert the blip. For example, a PICT will be stored as a msoblipPICT with a btWin32 field of msoblipWMF and a btMacOS field of msoblipPICT. When the PICT blip is loaded on Windows, the stored field will not match the OS field, so PICTtoWMF filter will be called to create a msoblipWMF BLIP.

A few additional facts are worth noting. Clients can define their own BLIP types. When loading client defined blip types Escher calls the clients to load the blips. Each BSE contains a 16-byte checksum that is used to quickly compare a BLIP with other BLIPs in the store. Any algorithm could be used for this checksum. Escher uses the *RSA Data Security, Inc. MD4 Message-Digest Algorithm* for the checksums of its BLIP types. Finally, the cRef field can be 0, indicating an empty slot in the BStore.

If a delay stream is not used, then the BLIP data follows the BSE header in a separate record. (If a delay stream *is* being used, the BLIP's record header and data are both written there instead.)

msofbtBlip*

Here is the format of the BLIP data. The FBT (MSOFBH::fbt) of the BLIP record is the MSOBLIPTYPE plus msofbtBlipFirst (0xF018). The instance (MSOFBH::inst) contains a signature that varies by blip type (see the Metafile/PICT/Bitmap Blip sections below.) The data that follows the file block header varies by blip type (again, (see the Metafile/PICT/Bitmap Blip sections below.)

Metafile/PICT Blips

Those blips have one of the following values from the MSOBI enumeration in MSOFBH::inst: msobiEMF, msobiWMF, or msobiPICT. They are normally stored in a compressed format using the LZ compression algorithm in the format used by GNU Zip deflate/inflate with a 32k window. The format is zlib format [1]. The only metafile compression version number currently defined identifies this format and is analogous to the PNG compression type value in the PNG file format. The filter values (MSOBLIPFILTER) define pre-filtering of metafile data to give better compression. Currently no pre-filtering is done (it is likely that filtering on a per-record basis will give substantially better compression in the future).

However, if there is an exception due to out-of-memory or out-of-disk space when saving those blips, the compression operation is skipped and the blips are then saved in a non-compressed format- in this case the compressed bits are simply the original metafile data. When the blips are loaded back in memory, a check is performed based on a "compression status" flag (MSOBLIPCOMPRESSION) that follows the blip header encoded as follows:

```
typedef enum
   {
   msocompressionDeflate = 0,
   msocompressionNone = 254,     // Used only if compression fails
   msocompressionTest = 255,     // For testing only
   }
MSOBLIPCOMPRESSION;

typedef enum
   {
   msofilterAdaptive = 0,        // PNG type - not used/supported for metafile
```

[1] The formal documentation is as follows (note that this will almost certainly not be of interest, see the comment about code resources below.)

Zlib: http://www.ietf.org/rfc/rfc1950

deflate: http://www.ietf.org/rfc/rfc1951.txt

PNG: http://www.ietf.org/rfc/rfc2083.txt

JFIF: JPEG File Interchange Format version 1.02 (September 1992) by Eric Hamilton

see: ftp://ftp.uu.net/graphics/jpeg/jfif.txt.gz

Sorry, but for some reason, I can't add comments. Please check the "gz" at the end of the previous link—It appears to me to not belong, but I can't be sure.

JPEG: ISO/IEC 10918-1

see:
http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=41504

```
   msofilterNone = 254,
   msofilterTest = 255,         // For testing only
   }
MSOBLIPFILTER;

/* The secondary, or data, UID - should always be set. */
BYTE  m_rgbUid[16];
/* The primary UID - this defaults to 0, in which case the primary ID is
   that of the internal data. NOTE!: The primary UID is only saved to disk
   if (blip_instance ^ blip_signature == 1). Blip_instance is MSOFBH.inst and
   blip_signature is one of the values defined in MSOBI */
BYTE  m_rgbUidPrimary[16]; /  / optional based on the above check

/* Metafile Blip overhead = 34 bytes. m_cb gives the number of
   bytes required to store an uncompressed version of the file, m_cbSave
   is the compressed size.  m_mfBounds gives the boundary of all the
   drawing calls within the metafile (this may just be the bounding box
   or it may allow some whitespace, for a WMF this comes from the
   SetWindowOrg and SetWindowExt records of the metafile). */
int           m_cb;         // Cache of the metafile size
RECT          m_rcBounds;   // Boundary of metafile drawing commands
POINT         m_ptSize;     // Size of metafile in EMUs
int           m_cbSave;     // Cache of saved size (size of m_pvBits)
BYTE         m_fCompression; // MSOBLIPCOMPRESSION
BYTE         m_fFilter  ;     // always msofilterNone
void         *m_pvBits;     // Compressed bits of metafile.
```

Bitmap Blips

Those blips have one of the following values from the MSOBI enumeration in MSOFBH::inst: msobiJPEG, msobiPNG, msobiCMYKJPEG, msobiTIFF, or msobiDIB. They have the same UID header as described in the Metafile Blip case. The data after the header is just a single BYTE "tag" value and is followed by the compressed data of the bitmap in the relevant format (JFIF, TIFF, GIF or PNG, bytes as would be stored in a file). For the msobiDIB format, the data is in the standard DIB format as a BITMAPINFOHEADER, BITMAPCOREHEADER or BITMAPV4HEADER followed by the color map (DIB_RGB_COLORS) and the bits. This data is not compressed (the format is used for very small DIB bitmaps only).

To determine where the bits are located, refer to the following header:

```
/* The secondary, or data, UID - should always be set. */
BYTE  m_rgbUid[16];
/* The primary UID - this defaults to 0, in which case the primary ID is
   that of the internal data. NOTE!: The primary UID is only saved to disk
   if (blip_instance ^ blip_signature == 1). Blip_instance is MSOFBH.finst and
   blip_signature is one of the values defined in MSOBI*/
BYTE  m_rgbUidPrimary[16];    // optional based on the above check
BYTE  m_bTag;
void  *m_pvBits;     // raster bits of the blip.
```

Drawing Container

msofbtDgContainer

The drawing container contains all per-slide/sheet types of information, including the shapes themselves. With a few exceptions, shapes are stored hierarchically according to how they've been grouped (through use of the Draw/Group command). For normal shapes, there is a special parent group shape called the patriarch that contains all of the top-level shapes (which in turn may contain other shapes). The patriarch is always the first msofbtSpgrContainer in the drawing container.

A few kinds of shapes are stored separately from the patriarch. The background shape, if there is one, is saved in its own msofbtSpContainer after the patriarch and its children. Additionally, if undo information is being saved and there are deleted shapes that could be brought back via Undo, the deleted shapes are saved. Note that there is no patriarch for the deleted shapes, so the top-level deleted shapes are saved separately into the drawing container. (Deleted groups still contain their children, though.)

Record Type	Condition	Comments
msofbtDg	Always.	Basic drawing information.
msofbtRegroupItems	Shapes have been ungrouped.	Mappings to reconstitute groups.
msofbtSpgrContainer	Always.	Patriarch shape, with all non-background non-deleted shapes inside it.
msofbtSpContainer with fBackground bit set in the FSP (see below).	Application uses a background shape (currently Word and PowerPoint only).	Special shape used as background of the document, e.g. the background texture of a Web page.
Other msofbtSpContainers and msofbtSpgrContainers	Undo is being saved, and there are deleted shapes in the drawing.	Shapes that have been deleted but that could be brought back via Undo.
msofbtSolverContainer	There are rules in the drawing.	Rules governing shapes in the drawing.
msofbtColorScheme	The application uses a color scheme.	Only present in the clipboard format.

Drawing Record

msofbtDg

The drawing record is very simple, with just a count and MSOSPID seed. The attentive reader may expect to find the size of the drawing recorded here, but that information is stored elsewhere by the host application.

```
// FDG - File DG
typedef struct _FDG
   {
   ULONG csp;     // The number of shapes in this drawing
   MSOSPID spidCur;  // The last MSOSPID given to an SP in this DG
   } FDG;
```

Regroup

msofbtRegroupItems

Each shape in a drawing has a regroup ID (separate from the shape ID), so that the regroup command can find shapes that were once grouped. In order to handle nested cases (e.g. ungroup, ungroup, ungroup, regroup, regroup, regroup), there is a table logging changes to regroup IDs. Each entry has an old ID and a new ID and records the change of all instances of the old ID to the new ID.

The instance of an msofbtRegroupItems record contains the number of entries, and the record itself is just that many FRITs (File Regroup Items).

```
typedef struct _FRIT // File Regroup item
   {
   FRID fridNew;
   FRID fridOld;
   } FRIT;
```

Group Container

msofbtSpgrContainer

A group is a collection of other shapes. The contained shapes are placed in the coordinate system of the group. The group container contains a variable number of shapes (msofbtSpContainer) and other groups (msofbtSpgrContainer, for nested groups). The group itself is a shape, and always appears as the first msofbtSpContainer in the group container.

Shape Container

msofbtSpContainer

A shape is the elemental object that composes a drawing. All graphical figures on a drawing are shapes. Each shape has a list of properties, which is stored in an array. A shape container contains the following records:

Record Type	Condition	Comments
msofbtSpgr	Shape is a group shape.	Group-shape-specific information.
msofbtSp	Always.	A shape atom record.
msofbtOPT	Always.	Those properties of a shape that are stored in Block 1.
msofbtSecondaryOPT	Shape has properties from Block 2.	The Block 2 properties of a shape.
msofbtTertiaryOPT	Shape has properties from Block 3.	The Block 3 properties of a shape.
msofbtAnchor or msofbtChildAnchor or msofbtClientAnchor	Always, except for the background shape.	The anchor or location of the shape. If the shape is saved to a clipboard, a msofbtAnchor record is used. If the shape is a child of a group shape, a msofbtChildAnchor is used. Otherwise, for top-level shapes, a host anchor record is present.
msofbtClientData	Always.	A client data record, the content of which is up to the host.
msofbtClientTextbox or msofbtTextbox	Shape has attached text.	If the shape has text, a text record is written. For clipboard streams, a msofbtTextbox record is used. Otherwise, a msofbtClientTextbox record is used, the content of which is up to the host.

msofbtOleObject	Shape is an OLE object.	Used only in the clipboard format.
msofbtDeletedPspl	Shape is deleted.	Link to previous spot of object.

Group Shape Record

msofbtSpgr

This record is present only in group shapes (not shapes *in* groups, shapes that *are* groups). The group shape record defines the coordinate system of the shape, which the anchors of the child shape are expressed in. All other information is stored in the shape records that follow.

```
typedef struct _FSPGR
   {
   RECT  rcgBounds;
   } FSPGR;
```

Shape Record

msofbtSp

The instance field of the record header contains the shape type; the record itself contains the shape ID and a group of persistent flags:

```
typedef struct _FSP
   {
   MSOSPID  spid;       // The shape id
   ULONG grfPersistent;
   } FSP;
```

The flags for the shape are:

```
typedef struct
{
   ULONG fGroup : 1;    // This shape is a group shape
   ULONG fChild : 1;    // Not a top-level shape
   ULONG fPatriarch : 1;   // This is the topmost group shape.
                  // Exactly one of these per drawing.
   ULONG fDeleted : 1;     // The shape has been deleted
   ULONG fOleShape : 1; // The shape is an OLE object
   ULONG fHaveMaster : 1;  // Shape has a hspMaster property
   ULONG fFlipH : 1;    // Shape is flipped horizontally
   ULONG fFlipV : 1;    // Shape is flipped vertically
   ULONG fConnector : 1;   // Connector type of shape
   ULONG fHaveAnchor : 1;  // Shape has an anchor of some kind
   ULONG fBackground : 1;  // Background shape
   ULONG fHaveSpt : 1;     // Shape has a shape type property
   ULONG reserved : 20; // Not yet used
}
```

Property Table Records

msofbtOPT, msofbtSecondaryOPT, msofbtTertiaryOPT

A shape's properties are stored in a sorted array of property id-value pairs. Only the properties that differ from the per-shape-type defaults or the per-property defaults are saved. (Note that the per-property defaults are unrelated to the default property table stored in the drawing group container).

The format of a property table record is in the Shape Properties section of this document.

Anchor Record

msofbtAnchor

An anchor record is used for top-level shapes when the shape streamed to the clipboard. The content of the record is simply a RECT with a coordinate system of 100,000 units per inch and origin in the top-left of the drawing.

Child Anchor Record

msofbtChildAnchor

A child anchor record is used for all shapes that belong to a group. The content of the record is simply a RECT in the coordinate system of the parent group shape.

Textbox Record

msofbtTextbox

A textbox record is used when a shape with attached text is written to a clipboard stream. It just contains an RTF string.

OLE Object Record

msofbtOleObject

An OLE object record is present when a shape that is an OLE object is saved to the clipboard. It contains the OLE object's storage, serialized using OleConvertIStorageToOLESTREAM.

Deleted PSPL Record

msofbtDeletedPspl

Top-level deleted shapes save a pointer back into their former position in the shape tree, so that if they are undeleted via undo they can be easily put back into the main shape tree.

The record consists of a single FPSPL:

```
// FPSPL - File PSPL
typedef struct _FPSPL
   {
   union
      {
      ULONG lAll;
      struct
         {
         ULONG spid : 30; // The SPID of the shape PSPL points at.
         ULONG fFirst : 1;  // Is this a pointer to the m_splFirst?
         ULONG fLast  : 1;  // Is this a pointer to the m_splLast?
         };
      };
```

```
   } FPSPL;
```

Solver Container

msofbtSolverContainer

Rules give special behaviors to shapes. Rules can govern a single shape, like in the case of a callout shape, or multiple shapes, as in the case of connectors. Each drawing can have a list of rules associated with it.

Connector Rule Record

msofbtConnectorRule

Governs a connector shape.

```
typedef struct _FConnectorRule
   {
   ULONG ruid;    // rule ID
   MSOSPID spidA; // SPID of shape A
   MSOSPID spidB; // SPID of shape B
   MSOSPID spidC; // SPID of connector shape
   ULONG cptiA;      // Connection site Index of shape A
   ULONG cptiB;      // Connection site Index of shape B
   } FConnectorRule;
```

Align Rule Record

msofbtAlignRule

Aligns shapes. The FAlignRule record is followed by the SPIDs of the proxy shapes.

```
// FAlignRule
typedef struct _FAlignRule
   {
   ULONG ruid;    // rule ID
   ULONG align;      // alignment - see below
   ULONG cProxies;   // number of shapes governed by rule
   } FAlignRule;

// ALIGN == Shape alignment (Horz and vert can be or'ed together)
#define alignHorz  0x000F  // mask for horizontal component
#define alignLeft  0x0001  //  left edges
#define alignCenter   0x0002  //  horizontal center
#define alignRight    0x0003  //  right edges
#define alignVert  0x00F0  // mask for vertical component
#define alignTop   0x0010  //  top edges
#define alignMiddle   0x0020  //  vertical center
#define alignBottom   0x0030  //  bottom edges
#define alignRelative 0x0100// Relative to the page
```

Arc Rule Record

msofbtArcRule

One Arc rule per elliptical arc shape.

```
// FARCRU -- Arc Rule
typedef struct _FARCRU
   {
   ULONG    ruid; // rule ID
   MSOSPID  spid; // spid of arc shape
   } FARCRU;
```

Callout Rule Record

msofbtCalloutRule

One callout rule per callout shape.

```
// FCORU -- Callout Rule
typedef struct _FCORU
   {
   ULONG    ruid; // rule ID
   MSOSPID  spid; // spid of callout shape
   } FCORU;
```

Color Scheme

msofbtColorScheme

Hosts may define their own color scheme and store colors in shape properties that are an index to that scheme plus an indicating flag (see Appendix B). Since hosts' color schemes are independent of each other, the color scheme is saved when rendering to the clipboard. If the clipboard data is pasted back into the same application, the color scheme block is ignored; otherwise, it is used to translate scheme colors in properties into RGB values.

The data in the block is an array of RGB values, saved as LONGs in order of color scheme index.

Selections

msofbtSelection

Selections of shapes are saved as top-level file blocks; they are never placed in a container. (Note: As of Office 97, only Excel saves shape selections; Word and PowerPoint do not.) The selection record consists of an FDGSL followed by the SPIDs of the shapes in the selection.

```
// FDGSL - File Drawing Selection
typedef struct _FDGSL
   {
   ULONG       cpsp;     // number of shapes in the selection
   ULONG       dgslk;    // kind of selection (an MSODGSLK)
   MSOSPID     spidFocus; // SPID of the focus shape
   } FDGSL;

// DGSLK = DrawinG SeLection Kind.
typedef enum
   {
   msodgslkNormal,         // Normal Selection Mode.
   msodgslkRotate,         // Rotate selection mode
   msodgslkReshape,        // Reshape Selection Mode.
   msodgslkUnused,
   msodgslkWrapPolygon, // Display and edit of wrap polygons.
```

```
   msodgslkTextEdit        // Text Edit Mode.
   } MSODGSLK;
```

Shape Properties

msofbtOPT, msofbtSecondaryOPT, msofbtTertiaryOPT

The first part of an OPT record is an array of FOPTEs, consisting of ID-value pairs tightly packed:

```
typedef struct _FOPTE
   {
   struct
      {
      USHORT pid : 14; // Property ID
      USHORT fBid : 1; // value is a blip ID – only valid if fComplex is FALSE
      USHORT fComplex : 1; // complex property, value is length
      };
   ULONG op;  // Value
   } FOPTE;
```

The FOPTE array is sorted by property ID.

Some property values, such as Unicode strings, don't fit in 32 bits. For these properties, the fComplex bit is set in the FOPTE, and the length of the data is saved in the value slot. The data of the complex properties follows the FOPTE array in the file record (sorted by property ID).

BLIPs are usually saved in the BLIP Store, so, in most cases, BLIP properties just store a BLIP ID (basically an index into an array in the BLIP Store). This is signaled by the fBid flag; note however that this flag is only valid if fComplex is FALSE.

Boolean properties are grouped in bitfields by property set; note that the Boolean properties in each property set below are contiguous. They are saved under the property ID of the *last* Boolean property in the set, and are placed in the value field in reverse order starting with the last property in the low bit.

Notes on types and units:

MSOHSP properties are basically just pointers to shapes, and they are therefore saved as SPIDs.

WCHAR* properties are Unicode strings; char* properties are ASCII strings.

IMsoArray properties are arrays. They are always complex when non-NULL. The complex-data part is saved as three shorts (16 bits each) followed by the array data. The first short is the number of elements in the array; the second short is the number of elements allocated for the array in memory (always greater than or equal to the first short); and the third short is the size of each array element.

Absolute distances are specified in English Metric Units (EMUs), occasionally referred to as A units; there are 360000 EMUs per centimeter, 914400 EMUs per inch, 12700 EMUs per point.

A coordinate space relative to the size of the shape is specified with the geoLeft, geoTop, geoRight, and geoBottom properties; coordinates in this space are said to be in G units.

Many quantities are specified as fixed-point 16.16 numbers; that is, the quantity fits in a LONG, where the high word specifies the integer part and the low word specifies the fractional part. In this system, 1<<16 signifies 1, 1<<17 signifies 2, and 1<<15 signifies ½.

The property listings below contain only those properties which are saved in OPT records in the file. Properties which are never saved, or which appear elsewhere in the file format, have been omitted.

The "Ver" column in the property listings specifies the version of Office in which the property was added. The property is known to that version and all subsequent versions, but not the ones preceding that version:

97	Office 97 for Windows
98	Office 98 for Macintosh
2000	Office 2000 for Windows
XP	Office XP for Windows
2003	Office System 2003 for Windows
2007	2007 Microsoft Office System for Windows

"Block 1, Block 2, Block 3"

The property table for a single shape or default property table may be split into as many as three records, with each individual record in the OPT format described above. Block 1 properties go into a record of type msofbtOPT, Block 2 properties into msofbtSecondaryOPT, and Block 3 properties into msofbtTertiaryOPT.

Using the "Ver" column, the in implementer must infer which "block" a property should be written in as follows: if "Ver" is 97 or 98, the property must appear in block 1 unless its property ID is 274 (movie,) in which case it must appear in block 2. If "Ver" is anything else, the property must appear in block 3.

When emitting the Office Drawing file format, the implementer must partition property table entries among the three record types as defined above. However, at read time an implementation must be prepared to handle any property in a block of *any* of the three types.

Transform

Position, size, rotation, and flipping of the shape.

Property	PID	Type	Default	Ver	Description
left	0	LONG	0	97	Bounds of the unrotated shape expressed as top left and bottom right in drawing units.
top	1	LONG	0	97	
right	2	LONG	1	97	
bottom	3	LONG	1	97	
rotation	4	LONG	0	97	Rotation is about the top left. Fixed point: 16.16 degrees
gvPage	5	MSOGV	0	97	
fChangePage	61	BOOL	FALSE	97	
fFlipV	62	BOOL	FALSE	97	Flip vertically
fFlipH	63	BOOL	FALSE	97	Flip horizontally

Protection

Changes the behavior of a shape by restricting direct manipulation.

Property	PID	Type	Default	Ver	Description
fLockAgainstUngrouping	118	BOOL	FALSE	XP	Do not ungroup this shape
fLockRotation	119	BOOL	FALSE	97	No rotation
fLockAspectRatio	120	BOOL	FALSE	97	Don’t allow changes in aspect ratio
fLockPosition	121	BOOL	FALSE	97	Don’t allow the shape to be moved
fLockAgainstSelect	122	BOOL	FALSE	97	Shape may not be selected
fLockCropping	123	BOOL	FALSE	97	No cropping this shape
fLockVertices	124	BOOL	FALSE	97	Edit Points not allowed
fLockText	125	BOOL	FALSE	97	Do not edit text
fLockAdjustHandles	126	BOOL	FALSE	97	Do not adjust
fLockAgainstGrouping	127	BOOL	FALSE	97	Do not group this shape

Text

How text fits in a shape. Text is host-dependent, so some hosts may ignore some of these properties.

Property	PID	Type	Default	Ver	Description
lTxid	128	LONG	0	97	id for the text, value determined by the host
dxTextLeft	129	LONG	1/10 inch	97	margins relative to shape's inscribed text rectangle (in EMUs)
dyTextTop	130	LONG	1/20 inch	97	margins relative to shape's inscribed text rectangle (in EMUs)
dxTextRight	131	LONG	1/10 inch	97	margins relative to shape's inscribed text rectangle (in EMUs)
dyTextBottom	132	LONG	1/20 inch	97	margins relative to shape's inscribed text rectangle (in EMUs)
WrapText	133	MSOWRAPMODE	FALSE	97	Wrap text at shape margins
scaleText	134	LONG	0	97	Text zoom/scale (used if fFitTextToShape)
anchorText	135	MSOANCHOR	Top	97	How to anchor the text
txflTextFlow	136	MSOTXFL	HorzN	97	Text flow
cdirFont	137	MSOCDIR	msocdir0	97	Font rotation
hspNext	138	MSOHSP	NULL	97	ID of the next shape (used by Word for linked textboxes)
txdir	139	MSOTXDIR	LTR	97	Bi-Di Text direction

ccol	140	LONG	1	XP	Count of columns
dzColMargin	141	LONG	91440	XP	Column margin on both sides (in EMUs)
fSelectText	187	BOOL	TRUE	97	TRUE if single click selects text, FALSE if two clicks
fAutoTextMargin	188	BOOL	FALSE	97	use host's margin calculations
fRotateText	189	BOOL	FALSE	97	Rotate text with shape
fFitShapeToText	190	BOOL	FALSE	97	Size shape to fit text size
fFitTextToShape	191	BOOL	FALSE	97	Size text to fit shape size

GeoText

Effect text of the shape - this is what the WordArt tools use, and is separate from the attached text present in ordinary textboxes. Theoretically, a shape could have both (a WordArt with attached text), but this is not currently allowed by the UI. Note that font information is provided here. The default text size is in points, the text effect geometry interfaces require the device size of a point to interpret this. The default point size is a 16.16 fixed-point number. A text effect is present if the fGText boolean is set and either the gtextUNICODE (UNICODE) or gtextRTF (RTF) is present, the UNICODE string takes precedence, however it cannot include any additional font information (unlike the RTF).

Property	PID	Type	Default	Ver	Description
gtextUNICODE	192	WCHAR*	NULL	97	UNICODE text string
gtextRTF	193	char*	NULL	97	RTF text string
gtextAlign	194	MSOGEOTEXTALIGN	Center	97	alignment on curve
gtextSize	195	LONG	36<<16	97	default point size
gtextSpacing	196	LONG	1<<16	97	fixed point 16.16
gtextFont	197	WCHAR*	NULL	97	font family name
gtextCSSFont	198	WCHAR*	""	2000	To preserve CSS font selectors
gtextFReverseRows	240	BOOL	FALSE	97	By default multiple rows of text are laid out with the first at the top for horizontal text and with the first at the left for vertical text, this flag reverses that behavior (bottom to top or right to left)
fGtext	241	BOOL	FALSE	97	Has text effect
gtextFVertical	242	BOOL	FALSE	97	Rotate characters
gtextFKern	243	BOOL	FALSE	97	Kern characters

gtextFTight	244	BOOL	FALSE	97	Tightening or tracking
gtextFStretch	245	BOOL	FALSE	97	Stretch to fit shape
gtextFShrinkFit	246	BOOL	FALSE	97	Char bounding box
gtextFBestFit	247	BOOL	FALSE	97	Scale text-on-path
gtextFNormalize	248	BOOL	FALSE	97	Stretch char height
gtextFDxMeasure	249	BOOL	FALSE	97	Do not measure along path
gtextFBold	250	BOOL	FALSE	97	Bold font
gtextFItalic	251	BOOL	FALSE	97	Italic font
gtextFUnderline	252	BOOL	FALSE	97	Underline font
gtextFShadow	253	BOOL	FALSE	97	Shadow font
gtextFSmallcaps	254	BOOL	FALSE	97	Small caps font
gtextFStrikethrough	255	BOOL	FALSE	97	Strike through font

Blip

How a BLIP fits into a shape. This includes cropping information as well as picture display modifications such as brightness and contrast.

Property	PID	Type	Default	Ver	Description
cropFromTop	256	LONG	0	97	16.16 fraction times total image width or height, as appropriate.
cropFromBottom	257	LONG	0	97	16.16 fraction times total image width or height, as appropriate.
cropFromLeft	258	LONG	0	97	16.16 fraction times total image width or height, as appropriate.
cropFromRight	259	LONG	0	97	16.16 fraction times total image width or height, as appropriate.
Pib	260	IMsoBlip*	NULL	97	Blip to display
pibName	261	WCHAR*	NULL	97	Blip file name
pibFlags	262	MSOBLIPFLAGS	Comment	97	Blip flags
pictureTransparent	263	Extended Color	~0	97	transparent color (none if ~0UL)
pictureContrast	264	LONG	1<<16	97	contrast setting
pictureBrightness	265	LONG	0	97	brightness setting
pictureGamma	266	LONG	0	97	16.16 gamma
pictureId	267	LONG	0	97	Host-defined ID for OLE

					objects (usually a pointer)
pictureDblCrMod	268	MSOCLR	This	97	Modification used if shape has double shadow
pictureFillCrMod	269	MSOCLR	undefined	97	
pictureLineCrMod	270	MSOCLR	undefined	97	
pibPrint	271	IMsoBlip*	NULL	97	Blip to display when printing
pibPrintName	272	WCHAR*	NULL	97	Blip file name
pibPrintFlags	273	MSOBLIPFLAGS	Comment	97	Blip flags
movie	274	MOVIE	NULL	98	Movie data
pictureRecolor	282	Extended Color	NULL	XP	Recolor the picture to this color
picturePreserveGrays	313	BOOL	NULL	XP	When doing a color modification to a picture, leave grays unmodified
fRewind	314	BOOL	FALSE	98	Should we rewind the movie when done playing
fLooping	315	BOOL	FALSE	98	Is movie looping
pictureGray	317	BOOL	FALSE	97	Display picture in grayscale
pictureBiLevel	318	BOOL	FALSE	97	Display picture in pure black and white
pictureActive	319	BOOL	FALSE	97	Server is active (OLE objects only)

Geometry

The geometry of the shape. Typically, these properties reside in a shape type definition, and so are not written to the file. However, freeform shapes drawing using the polygon tools set the pVertices and pSegmentInfo properties to define their geometries.

Property	PID	Type	Default	Ver	Description
geoLeft	320	LONG	0	97	Defines the G (geometry) coordinate space.
geoTop	321	LONG	0	97	Defines the G (geometry) coordinate space.
geoRight	322	LONG	21600	97	Defines the G (geometry) coordinate space.
geoBottom	323	LONG	21600	97	Defines the G (geometry) coordinate space.
shapePath	324	MSOSHAPEPATH	msoshapeLinesClosed	97	

pVertices	325	IMsoArray	NULL	97	An array of points, in G units.
pSegmentInfo	326	IMsoArray	NULL	97	
adjustValue	327	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
adjust2Value	328	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
adjust3Value	329	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
adjust4Value	330	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
adjust5Value	331	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
adjust6Value	332	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
adjust7Value	333	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
adjust8Value	334	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
adjust9Value	335	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape

					type to shape type.
adjust10Value	336	LONG	0	97	Adjustment values corresponding to the positions of the adjust handles of the shape. The number of values used and their allowable ranges vary from shape type to shape type.
pConnectionSites	337	IMsoArray*	STD	97	Array of connection sites (points where connectors can attach) in G units. Array entries are of type POINT.
pConnectionSitesDir	338	IMsoArray*	STD	97	Array of angles specifying angle at which connectors should connect to the corresponding connection sites. Angles are fixed point, 16.16 degrees. If this property is omitted, angles are determined automatically based on the geometric center of the shape. Array entries are of type LONG.
xLimo	339	LONG	LONG_MIN	97	The point along the x dimension of the shape where it will limo stretch. Specified in G units.
yLimo	340	LONG	LONG_MIN	97	The point along the y dimension of the shape where it will limo stretch. Specified in G units.
pAdjustHandles	341	IMsoArray*	STD	97	Array of adjust handles for the shape. Array entries are of type ADJH.
pGuides	342	IMsoArray*	STD	97	Array of guide formula for the shape which specify how the geometry of the shape changes as the adjust handles are dragged. Array entries are of type SG.
pInscribe	343	IMsoArray*	STD	97	Array of inscribed rectangles. Array entries are of type RECT.
cxk	344	MSOCXK	msocxkSegments	97	Type of connection sites
pFragments	345	IMsoArray*	NULL	97	Array of fragment ids
fColumnLineOK	377	BOOL	FALSE	XP	Column style may be set
fShadowOK	378	BOOL	TRUE	97	Shadow may be set
f3DOK	379	BOOL	TRUE	97	3D may be set
fLineOK	380	BOOL	TRUE	97	Line style may be set
fGtextOK	381	BOOL	FALSE	97	Text effect (WordArt) supported
fFillShadeShapeOK	382	BOOL	FALSE	97	If true, a concentric gradient fill will be drawn based on the geometry of the shape. If false, a concentric gradient fill will be

					drawn rectangularly
fFillOK	383	BOOL	TRUE	97	OK to fill the shape through the UI or VBA?

For simple closed polygons, the pSegmentInfo property can be omitted. The data in the pVerticies array is interpreted as a list of vertices for the polygon. For a more complex path, the pSegmentInfo property specifies the types of segments and number of segments in the path and determines how you should interpret the list of values in the pVerticies array.

For example, let's take a look at the path data for the Heart shape. The pSegmentInfo array looks like

```
msopathMoveTo,  0
msopathCurveTo, 5
msopathLineTo,  2
msopathCurveTo, 5
msopathClose,   1
msopathEnd,     0
```

and the pVertices array looks like

```
10860 ,  2187
10451 ,  1746
 9529 ,  1018
 9015 ,   730
 7865 ,   152
 6685 ,     0
 5415 ,     0
 4175 ,   152
 2995 ,   575
 1967 ,  1305
 1150 ,  2187
  575 ,  3222
  242 ,  4220
    0 ,  5410
  242 ,  6560
  575 ,  7597
10860 , 21600
20995 ,  7597
21480 ,  6560
21600 ,  5410
21480 ,  4220
21115 ,  3222
20420 ,  2187
19632 ,  1305
18575 ,   575
17425 ,   152
16275 ,     0
15005 ,     0
13735 ,   152
12705 ,   730
12176 ,  1018
11254 ,  1746
```

```
10860 ,  2187
```

- The MoveTo,0 says to take the 1st vertex in pVertices, (10860,2187), and place the pen at that location.
- The CurveTo,5 says to draw 5 cubic bezier segments. The current pen location is used as the first vertex of the first cubic bezier segment, and the next 3 vertices from pVertices, (10451,1746), (9529,1018), and (9015,730), are used as the control points and endpoint of the first cubic bezier segments. This is repeated 4 more times using a total of 15 vertices from pVertices for the 5 cubic bezier segments.
- The LineTo,2 says to draw 2 straight line segments. One line segment is drawn from the current pen location to the next vertex in pVertices, (10860,21600), and another line drawn from that point to the next vertex in pVertices, (20995,7597).
- The CurveTo,5 says to draw 5 more cubic bezier segments using another 15 vertices in pVertices.
- The Close,1 says the draw 1 line segment from the current pen location back to first point in the path and make it a closed path. No vertices from pVertices are taken for this operation.
- The End,0 says to end the path. No vertices from pVertices are taken for this operation.

Each SegmentType,SegmentCount pair is stored in an unsigned short value. The segment type is stored in the upper 3 bits and segment count is stored in the lower 13 bits.

Segment type can be any of the follow enum values:

```
typedef enum
   {
   msopathLineTo,         // Draw a straight line (one point)
   msopathCurveTo,        // Draw a cubic Bezier curve (three points)
   msopathMoveTo,         // Move to a new point (one point)
   msopathClose,          // Close a sub-path (no points)
   msopathEnd,            // End a path (no points)
   msopathEscape,         // Escape code
   msopathClientEscape,   // Escape code interpreted by the client
   msopathInvalid         // Invalid - should never be found
   }
MSOPATHTYPE;
```

If the segment type is msopathEscape, the lower 13 bits are divided in a 5 bit escape code and 8 bit vertex count (not segment count!).

The freeform objects produced in Office will contain some msopathEscape. These store editing information (like whether or not to allow the control points to be adjusted on a bezier segment). But these are not needed to understand how the freeforms are rendered.

All CurveTo segments in a path are cubic beziers. The mathematical definition for a cubic bezier can be found in most computer graphics textbooks. In Office, some of the built-in AutoShapes have some cubic bezier curve segments in them. Shapes drawn with the "Curve" tool have only cubic bezier curve segments in them. Shapes drawn with the "Freeform" tool have cubic bezier curve segments in the smooth sections and straight-line segments in the straight sections.

Fill Style

Two main colors are defined - a foreground color and a background color. Different fillTypes use these values differently. In addition to the foreground and background any number of shade colors can be

defined. Each shade color is associated with a "position" which says how far into the shade the color appears – colors must be given in position order.

For a solid fill the foreground color is used and the background (and everything else except transparency) is ignored. For pattern and texture fills the fillBlip identifies a BLIP, which will be used for the fill.

fillWidth and fillHeight define the desired pattern/texture size in EMUs. The pattern/tile will be expanded to this size. If the pattern is a bitmap the actual size will be rounded to a close integer multiple of the original (pixel) size of the bitmap. If the size is 0 then the pattern/tile will not be expanded or contracted at all in pixel terms, so the fill will be device dependent - this may result in non-proportional scaling between devices (on devices with non-square pixels).

For a pattern the foreground and background colors define the colors to use when filling with a pattern, for a texture the colors are in the bitmap (this is the only difference).

For both pattern and texture fills the fill is registered with (0,0) on the view in which the effect appears unless fillShape is set to TRUE, in which case the pattern/texture is registered relative to the shape (so it moves with the shape).

For a picture fill the fillBlip is centered in the shape - not tiled.

For a shaded fill the colors define shade bands to use across the shade, shading between each pair of colors. The positions are the positions of the shade at which the given color appears - the shade ends with the first index that is >= 1 (in 16.16 notation), the indices must be in ascending order or the result is undefined. The interpretation of the shade start and shade end points varies according to the exact shade type.

The fillBackground fill indicates a fill inherited from a background object.

Property	PID	Type	Default	Ver	Description
fillType	384	MSOFILLTYPE	Solid	97	Type of fill
fillColor	385	Extended Color	white	97	Foreground color
fillOpacity	386	LONG	1<<16	97	Fixed 16.16
fillBackColor	387	Extended Color	white	97	Background color
fillBackOpacity	388	LONG	1<<16	97	Shades only
fillCrMod	389	MSOCLR	undefined	97	Modification for BW views
fillBlip	390	IMsoBlip*	NULL	97	Pattern/texture
fillBlipName	391	WCHAR*	NULL	97	Blip file name
fillBlipFlags	392	MSOBLIPFLAGS	Comment	97	Blip flags
fillWidth	393	LONG	0	97	How big (A units) to make a metafile texture.
fillHeight	394	LONG	0	97	How big (A units) to make a metafile texture.
fillAngle	395	LONG	0	97	Fade angle - degrees in 16.16
fillFocus	396	LONG	0	97	Linear shaded fill focus percent
fillToLeft	397	LONG	0	97	Fraction 16.16
fillToTop	398	LONG	0	97	Fraction 16.16

fillToRight	399	LONG	0	97	Fraction 16.16
fillToBottom	400	LONG	0	97	Fraction 16.16
fillRectLeft	401	LONG	0	97	For shaded fills, use the specified rectangle instead of the shape's bounding rect to define how large the fade is going to be.
fillRectTop	402	LONG	0	97	For shaded fills, use the specified rectangle instead of the shape's bounding rect to define how large the fade is going to be.
fillRectRight	403	LONG	0	97	For shaded fills, use the specified rectangle instead of the shape's bounding rect to define how large the fade is going to be.
fillRectBottom	404	LONG	0	97	For shaded fills, use the specified rectangle instead of the shape's bounding rect to define how large the fade is going to be.
fillDztype	405	MSODZTYPE	Default	97	
fillShadePreset	406	LONG	0	97	Special shades
fillShadeColors	407	IMsoArray	NULL	97	a preset array of colors
fillOriginX	408	LONG	0	97	
fillOriginY	409	LONG	0	97	
fillShapeOriginX	410	LONG	0	97	
fillShapeOriginY	411	LONG	0	97	
fillShadeType	412	MSOSHADETYPE	Default	97	Type of shading, if a shaded (gradient) fill.
fRecolorFillAsPicture	441	BOOL	FALSE	XP	Recolor the picture fill according to the recoloring properties from the Picture property set
fUseShapeAnchor	442	BOOL	TRUE	XP	Fit the fill to the shape anchor, not the bounds
fFilled	443	BOOL	TRUE	97	Is shape filled?
fHitTestFill	444	BOOL	TRUE	97	Should we hit test fill?
fillShape	445	BOOL	TRUE	97	Register pattern on shape
fillUseRect	446	BOOL	FALSE	97	Use the large rect?
fNoFillHitTest	447	BOOL	FALSE	97	Hit test a shape as though filled

Line Style

Lines are centered about the infinitely thin proto-line along which they are drawn. Complex dash effects are supported only for simple lines (e.g. changing the end cap) - defaults should be used for

other line styles. The line width is in EMUs; a line width of zero should not be used - there is no logical interpretation on a high-resolution printer.

Property	PID	Type	Default	Ver	Description
lineColor	448	Extended Color	black	97	Color of line
lineOpacity	449	LONG	1<<16	97	Not implemented
lineBackColor	450	Extended Color	white	97	Background color
lineCrMod	451	MSOCLR	undefined	97	Modification for BW views
lineType	452	MSOLINETYPE	Solid	97	Type of line
lineFillBlip	453	IMsoBlip*	NULL	97	Pattern/texture
lineFillBlipName	454	WCHAR*	NULL	97	Blip file name
lineFillBlipFlags	455	MSOBLIPFLAGS	Comment	97	Blip flags
lineFillWidth	456	LONG	0	97	How big (A units) to make a metafile texture.
lineFillHeight	457	LONG	0	97	How big (A units) to make a metafile texture.
lineFillDztype	458	MSODZTYPE	Default	97	How to interpret fillWidth/Height numbers.
lineWidth	459	LONG	9525	97	A units; 1pt == 12700 EMUs
lineMiterLimit	460	LONG	8<<16	97	ratio (16.16) of width
lineStyle	461	MSOLINESTYLE	Simple	97	Draw parallel lines?
lineDashing	462	MSOLINEDASHING	Solid	97	Can be overridden by:
lineDashStyle	463	IMsoArray	NULL	97	As Win32 ExtCreatePen
lineStartArrowhead	464	MSOLINEEND	NoEnd	97	Arrow at start
lineEndArrowhead	465	MSOLINEEND	NoEnd	97	Arrow at end
lineStartArrowWidth	466	MSOLINEENDWIDTH	MediumWidthArrow	97	Arrow at start
lineStartArrowLength	467	MSOLINEENDLENGTH	MediumLenArrow	97	Arrow at end
lineEndArrowWidth	468	MSOLINEENDWIDTH	MediumWidthArrow	97	Arrow at start
lineEndArrowLength	469	MSOLINEENDLENGTH	MediumLenArrow	97	Arrow at end
lineJoinStyle	470	MSOLINEJOIN	JoinRound	97	How to join lines
lineEndCapStyle	471	MSOLINECAP	EndCapFlat	97	How to end lines
fInsetPen	505	BOOL	FALSE	XP	Draw line inside the shape

fInsetPenOK	506	BOOL	TRUE	XP	Allow inset pen if prop. is set
fArrowheadsOK	507	BOOL	FALSE	97	Allow arrowheads if prop. is set
fLine	508	BOOL	TRUE	97	Any line?
fHitTestLine	509	BOOL	TRUE	97	Should we hit test lines?
lineFillShape	510	BOOL	TRUE	97	Register pattern on shape
fNoLineDrawDash	511	BOOL	FALSE	97	Draw a dashed line if no line

Shadow Style

The interpretation of the transform properties depends on the type of shadow:

msoshadowOffset, msoshadowDouble:

Only the offset is used. It is interpreted as an absolute offset expressed in EMUs. The default corresponds to 1/36” in both X and Y (2 or 3 pixels on screen depending on monitor resolution). The offset is relative to the drawing axes (as msoshadowDrawing below, not msoshadowRich) so a shadow offset to the bottom right of the drawing is still offset (by the same amount) to the bottom right if the shape is rotated. The “double” case causes two shadows to be drawn, the first (lower) at the second offset and in the shadowHighlightColor. If the second offset is 0,0, it defaults to being the inverse of the first.

msoshadowRich:

The offsets and transformation properties are in absolute units measured relative to the shape on the drawing - the shadow moves with the shape, but anisotropic scaling of the shape changes the proportions of the shadow, not its angles. Compare with the following where such scaling scales the shadow in proportion too, thus changes the angle between (e.g.) a vertical line in the shape and it’s shadow.

msoshadowShape:

The offsets and transformation properties are relative to the shape; 1.0 corresponds to the shape width/height as appropriate. The shadow is cast relative to the shape then scaled with the shape, so it moves with the shape. The units are simple numbers (ratios of the G unit space effectively). This transformation type is unnatural in real world terms, but behaves nicely in geometric terms. The offset elements of the property set are treated as fixed-point 16.16 values.

msoshadowDrawing:

A rich shadow cast onto a plane in drawing space. The transform is applied to the drawing coordinates of the shape and is thus expressed in EMUs. This shadow type enables creation of shadows from multiple objects; however the shadows may overlap higher (different) objects if the shadow plane and shape drawing planes overlap on the screen.

The shadowWeight parameter is used as in the perspective property set to apply additional scaling to the perspective parameters - these are divided by the weight.

Shadow transformations are independent of the perspective transformation applied to a shape - either the perspective transformation or the shadow transformation is used as appropriate.

Property	PID	Type	Default	Ver	Description
shadowType	512	MSOSHADOWTYPE	Offset	97	Type of effect
shadowColor	513	Extended Color	0x808080	97	Foreground color
shadowHighlight	514	Extended Color	0xCBCBCB	97	Embossed color
shadowCrMod	515	MSOCLR	undefined	97	Modification for BW views
shadowOpacity	516	LONG	1<<16	97	Fixed 16.16
shadowOffsetX	517	LONG	25400	97	Offset shadow
shadowOffsetY	518	LONG	25400	97	Offset shadow
shadowSecondOffsetX	519	LONG	0	97	Double offset shadow
shadowSecondOffsetY	520	LONG	0	97	Double offset shadow
shadowScaleXToX	521	LONG	1<<16	97	16.16
shadowScaleYToX	522	LONG	0	97	16.16
shadowScaleXToY	523	LONG	0	97	16.16
shadowScaleYToY	524	LONG	1<<16	97	16.16
shadowPerspectiveX	525	LONG	0	97	16.16 / weight
shadowPerspectiveY	526	LONG	0	97	16.16 / weight
shadowWeight	527	LONG	1<<8	97	scaling factor
shadowOriginX	528	LONG	0	97	
shadowOriginY	529	LONG	0	97	
fShadow	574	BOOL	FALSE	97	Any shadow?
fshadowObscured	575	BOOL	FALSE	97	Excel5-style shadow

Perspective Style

This is just a 2D transformation matrix (3x3). Specifying peculiar values will cause the shape to render completely outside its geometry - normally clients will constrain the values to get reasonable results. The transformation may be applied at various times as the geometry is processed, this affects the behavior of the perspective which results in the same way as the corresponding shadow perspective types.

msoxformAbsolute Equivalent to msoshadowRich

msoxformShape Equivalent to msoshadowShape

msoxformDrawing Equivalent to msoshadowDrawing

In the case of perspective the msoxformShape form is the default - the perspective will then scale proportionally with the overall shape scaling.

All parameters except the weight and offset elements are 16.16 fixed-point numbers. The offset is interpreted according to the perspective type, if the type is msoxformShape then the offset is assumed

to be a 16.16 fixed point value, otherwise it is assumed to be an integral value (effectively a number of EMUs). The weight acts as an additional divisor for the perspectivePerspectiveX/Y elements - the values in the transformation matrix are obtained by dividing the property values by the weight, the default value of 256 gives a useful range of values for the msoxformShape case, for the other cases the weight should normally be around 1 to 256 times the size of the shape in the coordinate space.

Property	PID	Type	Default	Ver	Description
perspectiveType	576	MSOXFORMTYPE	Shape	97	Where transform applies
perspectiveOffsetX	577	LONG	0	97	The LONG values define a transformation matrix, effectively, each value is scaled by the perspectiveWeight parameter.
perspectiveOffsetY	578	LONG	0	97	The LONG values define a transformation matrix, effectively, each value is scaled by the perspectiveWeight parameter.
perspectiveScaleXToX	579	LONG	1<<16	97	The LONG values define a transformation matrix, effectively, each value is scaled by the perspectiveWeight parameter.
perspectiveScaleYToX	580	LONG	0	97	The LONG values define a transformation matrix, effectively, each value is scaled by the perspectiveWeight parameter.
perspectiveScaleXToY	581	LONG	0	97	The LONG values define a transformation matrix, effectively, each value is scaled by the perspectiveWeight parameter.
perspectiveScaleYToY	582	LONG	1<<16	97	The LONG values define a transformation matrix, effectively, each value is scaled by the perspectiveWeight parameter.
perspectivePerspectiveX	583	LONG	0	97	The LONG values define a transformation matrix, effectively, each value is scaled by the perspectiveWeight parameter.
perspectivePerspectiveY	584	LONG	0	97	The LONG values define a transformation matrix, effectively, each value is scaled by the perspectiveWeight parameter.
perspectiveWeight	585	LONG	1<<8	97	Scaling factor
perspectiveOriginX	586	LONG	1<<15	97	
perspectiveOriginY	587	LONG	1<<15	97	
fPerspective	639	BOOL	FALSE	97	On/off

3D Object

Material properties of a 3D object. A 3D effect overrides the fill and line effects and corresponding colors. Extrusion depths are always specified in absolute units.

Property	PID	Type	Default	Ver	Description
c3DSpecularAmt	640	LONG	0	97	Fixed-point 16.16
c3DDiffuseAmt	641	LONG	65536	97	Fixed-point 16.16
c3DShininess	642	LONG	5	97	Default gives OK results
c3DEdgeThickness	643	LONG	12700	97	Specular edge thickness
C3DExtrudeForward	644	LONG	0	97	Distance of extrusion in EMUs
c3DExtrudeBackward	645	LONG	457200	97	Distance of extrusion in EMUs
c3DExtrudePlane	646	LONG	0	97	Extrusion direction
c3DExtrusionColor	647	Extended Color	FillThenLine	97	Basic color of extruded part of shape; the lighting model used will determine the exact shades used when rendering.
c3DCrMod	648	MSOCLR	undefined	97	Modification for BW views
f3D	700	BOOL	FALSE	97	Does this shape have a 3D effect?
fc3DMetallic	701	BOOL	0	97	Use metallic specularity?
fc3DUseExtrusionColor	702	BOOL	FALSE	97	
fc3DLightFace	703	BOOL	TRUE	97	

3D Style

Properties of a 3D view; note that distances are in drawing units.

Property	PID	Type	Default	Ver	Description
c3DYRotationAngle	704	LONG	0	97	degrees (16.16) about y axis
c3DXRotationAngle	705	LONG	0	97	degrees (16.16) about x axis
c3DRotationAxisX	706	LONG	100	97	These specify the rotation axis; only their relative magnitudes matter.
c3DRotationAxisY	707	LONG	0	97	These specify the rotation axis; only their relative magnitudes matter.
c3DRotationAxisZ	708	LONG	0	97	These specify the rotation axis; only their relative magnitudes matter.

c3DRotationAngle	709	LONG	0	97	degrees (16.16) about axis
c3DRotationCenterX	710	LONG	0	97	rotation center x (16.16 or g-units)
c3DRotationCenterY	711	LONG	0	97	rotation center y (16.16 or g-units)
c3DRotationCenterZ	712	LONG	0	97	rotation center z (absolute (emus))
c3DRenderMode	713	MSO3DRENDERMODE	FullRender	97	Full,wireframe, or bcube
c3DTolerance	714	LONG	30000	97	pixels (16.16)
c3DXViewpoint	715	LONG	1250000	97	X view point (emus)
c3DYViewpoint	716	LONG	-1250000	97	Y view point (emus)
c3DZViewpoint	717	LONG	9000000	97	Z view distance (emus)
c3DOriginX	718	LONG	32768	97	
c3DOriginY	719	LONG	-32768	97	
c3DSkewAngle	720	LONG	-8847360	97	degree (16.16) skew angle
c3DSkewAmount	721	LONG	50	97	Percentage skew amount
c3DAmbientIntensity	722	LONG	20000	97	Fixed point intensity
c3DKeyX	723	LONG	50000	97	Key light source direc-
c3DKeyY	724	LONG	0	97	tion; only their relative
c3DKeyZ	725	LONG	10000	97	magnitudes matter
c3DKeyIntensity	726	LONG	38000	97	Fixed point intensity
c3DFillX	727	LONG	-50000	97	Fill light source direc-
c3DFillY	728	LONG	0	97	tion; only their relative
c3DFillZ	729	LONG	10000	97	magnitudes matter
c3DFillIntensity	730	LONG	38000	97	Fixed point intensity
fc3DConstrainRotation	763	BOOL	TRUE	97	
fc3DRotationCenterAuto	764	BOOL	FALSE	97	
fc3DParallel	765	BOOL	1	97	Parallel projection?
fc3DKeyHarsh	766	BOOL	1	97	Is key lighting harsh?
fc3DFillHarsh	767	BOOL	0	97	Is fill lighting harsh?

Shape

Miscellaneous properties of a single shape which do not apply to group shapes.

Property	PID	Type	Default	Ver	Description

hspMaster	769	MSOHSP	NULL	97	master shape
cxstyle	771	MSOCXSTYLE	None	97	Type of connector
bWMode	772	MSOBWMODE	Automatic	97	Settings for modifications to be made when in different forms of black-and-white mode.
bWModePureBW	773	MSOBWMODE	Automatic	97	Settings for modifications to be made when in different forms of black-and-white mode.
bWModeBW	774	MSOBWMODE	Automatic	97	Settings for modifications to be made when in different forms of black-and-white mode.
idDiscussAnchor	775	LONG	STD	2000	
dgmLayout	777	MSODGMLO	msodgmlo Nil	XP	Node layout
dgmNodeKind	778	DGMNK	-1	XP	Kind of node
dgmLayoutMRU	779	MSODGMLO	msodgmlo Nil	XP	Most recently used layout for its child
wzEquationXML	780	char*	NULL	2007	This property is present if the shape represents an equation generated by Office 2007 or later. The property is a string of XML representing a Word 2003 XML document. The original equation is stored within the “oMathPara” tag within the document. Refer to the Office Open XML documentation for details on this XML representation of equations. If the document containing the shape is opened in Office 2007 or later, the shape is replaced with the equation in this document.
fPolicyLabel	822	BOOL	FALSE	2007	True if the shape is a policy label representing metadata about a document.
fPolicyBarcode	823	BOOL	FALSE	2007	True if the shape represents a barcode as part of a barcode policy for record management.
fFlipHQFE5152	824	BOOL	FALSE	XP	The value of this property should match the value of

					the fFlipH property (in the transform property set) if the pib property exists (in the blip property set.)
fFlipVQFE5152	825	BOOL	FALSE	XP	The value of this property should match the value of the fFlipV property (in the transform property set) if the pib property exists (in the blip property set.)
fPreferRelativeResize	827	BOOL	FALSE	97	For UI only. Prefer relative resizing.
fLockShapeType	828	BOOL	FALSE	97	Lock the shape type (don't allow Change Shape)
fInitiator	829	BOOL	NULL	97	Set by the solver
fDeleteAttachedObject	830	BOOL	FALSE	97	
fBackground	831	BOOL	FALSE	97	If TRUE, this is the background shape.

Callout

Properties of a callout shape.

Property	PID	Type	Default	Ver	Description
spcot	832	MSOSPCOT	TwoSegment	97	Callout type
dxyCalloutGap	833	LONG	1/12 inch	97	Distance from box to first point.(EMUs)
spcoa	834	MSOSPCOA	Any	97	Callout angle
spcod	835	MSOSPCOD	Specified	97	Callout drop type
dxyCalloutDropSpecified	836	LONG	9 points	97	if msospcodSpecified, the actual drop distance
dxyCalloutLengthSpecified	837	LONG	0	97	if fCalloutLengthSpecified, the actual distance
fCallout	889	BOOL	FALSE	97	Is the shape a callout?
fCalloutAccentBar	890	BOOL	FALSE	97	does callout have accent bar
fCalloutTextBorder	891	BOOL	TRUE	97	does callout have a text border
fCalloutMinusX	892	BOOL	FALSE	97	If true, callout tail is to the right of the box
fCalloutMinusY	893	BOOL	FALSE	97	If true, callout tail is

					below the box
fCalloutDropAuto	894	BOOL	FALSE	97	If true, then we occasionally invert the drop distance
fCalloutLengthSpecified	895	BOOL	FALSE	97	If true, we look at dxyCalloutLengthSpecified

Group Shape

Miscellaneous shape properties that *can* apply to group shapes.

Property	PID	Type	Default	Ver	Description
wzName	896	WCHAR*	NULL	97	Shape Name (present only if explicitly set)
wzDescription	897	WCHAR*	NULL	97	alternate text
pihlShape	898	IHlink*	NULL	97	The hyperlink in the shape.
pWrapPolygonVertices	899	IMsoArray	NULL	97	The polygon that text will be wrapped around (Word)
dxWrapDistLeft	900	LONG	1/8 inch	97	Left wrapping distance from text (Word)
dyWrapDistTop	901	LONG	0	97	Top wrapping distance from text (Word)
dxWrapDistRight	902	LONG	1/8 inch	97	Right wrapping distance from text (Word)
dyWrapDistBottom	903	LONG	0	97	Bottom wrapping distance from text (Word)
lidRegroup	904	LONG	0	97	Regroup ID
groupLeft	905	LONG	0	97	The group's coordinate rectangle
groupTop	906	LONG	0	97	
groupRight	907	LONG	20000	97	
groupBottom	908	LONG	20000	97	
wzTooltip	909	WCHAR*	NULL	2000	Tooltip for the hyperlink in the shape.
wzScript	910	WCHAR*	STD	2000	Script (JavaScript, VBScript etc) attached to shape
posh	911	MSOPH	msophAbs	2000	
posrelh	912	MSOPRH	msoprhText	2000	
posv	913	MSOPV	msopvAbs	2000	

posrelv	914	MSOPRV	msoprvText	2000	
pctHR	915	LONG	1000	2000	Percentage width for a horizontal rule
alignHR	916	LONG	0	2000	Alignment for an HR; left==0, center==1, right==2
dxHeightHR	917	LONG	0	2000	Height for an HR
dxWidthHR	918	LONG	0	2000	Width for an HR
wzScriptExtAttr	919	WCHAR*	STD	2000	Extended Script Attributes (other than Lang, Id) of script(VBScript etc) attached to shape
scriptLang	920	LONG	1	2000	Script Language of script attached to shape (JavaScript, VBScript or other)
wzScriptIdAttr	921	WCHAR*	STD	2000	Id Script Attribute of script(VBScript etc) attached to shape
wzScriptLangAttr	922	WCHAR*	STD	2000	Lang Script Attribute of script(VBScript etc) attached to shape
borderTopColor	923	COLORREF	MSOCOLORNONE	2000	Top border color (WORD)
borderLeftColor	924	COLORREF	MSOCOLORNONE	2000	Left border color (WORD)
borderBottomColor	925	COLORREF	MSOCOLORNONE	2000	Bottom border color (WORD)
borderRightColor	926	COLORREF	MSOCOLORNONE	2000	Right border color (WORD)
tableProperties	927	LONG	0	2000	Flags that indicate whether the group shape represents a PowerPoint table. Bit 1: Group shape is a table Bit 2: Group shape is a table placeholder Bit 3: The table is right-to-left ordered
tableRowProperties	928	IMsoArray*	NULL	2000	Row heights if the group shape is a PowerPoint table. Array entries are 32-bit integers, where each entry is a row height in PowerPoint master coordinates.
scriptHtmlLocation	929	LONG	2	2000	Script location
wzApplet	930	WCHAR*	NULL	2000	Applet Body - not really a shape - visual cue to indicate presence of

					an applet block
wzFrameTrgtUnused	932	WCHAR*	NULL	2000	Frame target for the hyperlink in the shape.
wzWebBot	933	WCHAR*	STD	2000	If shape is representing a frontpage webbot, this is content attached
wzAppletArg	934	WCHAR*	NULL	2000	Applet Tag arguments
wzAccessBlob	936	WCHAR*	NULL	XP	If shape is representing an access datapage HTML blob, this is the HTML
metroBlob	937	IByteStream*	NULL	2007	The shape's 2007 representation in Office Open XML format. The actual data is a package in Office XML format, which can simply be opened as a zip file. This zip file contains an XML file with the root element "sp". Refer to the publically available Office Open XML documentation for more information about this data. In case we lose any property when converting a 2007 Office Art shape to 2003 shape, we use this blob to retrieve the original Office Art property data when opening the file in 2007. See Appendix F for more information.
dhgt	938	DHGT	0	2007	The shape's unique z-order, ranging from 1 to 503316479. The higher the number, the closer to "in-front" the shape is.
fLayoutInCell	944	BOOL	TRUE	2000	Lay out inside nested table if TRUE
fIsBullet	945	BOOL	FALSE	2000	Is the shape a picture bullet
fStandardHR	946	BOOL	FALSE	2000	fTrue iff not a graphical HR
fNoshadeHR	947	BOOL	FALSE	2000	fTrue iff an HR with NOSHADE set
fHorizRule	948	BOOL	FALSE	2000	fTrue iff is HR
fUserDrawn	949	BOOL	FALSE	2000	UserDrawn shape on PPT master.
fAllowOverlap	950	BOOL	TRUE	2000	True if the shape is allowed to overlap other shapes in Web Layout view in Word.
fReallyHidden	951	BOOL	FALSE	2000	Has the fHidden flag really been set by user (used to hide script anchors in Office97)

fScriptAnchor	952	BOOL	FALSE	2000	Script Anchor - visual cue to indicate presence of script block

Relative Transform

Defines the size and location of the shape in the parent group or drawing. The coordinates are relative to the position of the parent group or drawing. The units are relative to the rcg of the parent. The Relative Transform and Transform property sets express the same information in different ways. For top-level shape they are equivalent.

Property	PID	Type	Default	Ver	Description
relLeft	0	LONG	0	97	Bounds of the unrotated shape expressed as top left and bottom right in drawing units.
relTop	1	LONG	0	97	
relRight	2	LONG	1	97	
relBottom	3	LONG	1	97	
relRotation	4	LONG	0	97	Rotation is about the top left. Fixed point: 16.16 degrees
gvRelPage	5	MSOGV	0	97	
fRelChangePage	61	BOOL	FALSE	97	
fRelFlipV	62	BOOL	FALSE	97	Flip vertically
fRelFlipH	63	BOOL	FALSE	97	Flip horizontally

Unknown HTML

ID properties from generic HTML.

Property	PID	Type	Default	Ver	Description
wzLineId	1026	WCHAR*	NULL	2000	XML line element ID
wzFillId	1027	WCHAR*	NULL	2000	XML fill element ID
wzPictureId	1028	WCHAR*	NULL	2000	XML picture element ID
wzPathId	1029	WCHAR*	NULL	2000	XML path element ID
wzShadowId	1030	WCHAR*	NULL	2000	XML shadow element ID
wzPerspectiveId	1031	WCHAR*	NULL	2000	XML perspective element ID
wzGtextId	1032	WCHAR*	NULL	2000	XML geotext element ID
wzFormulaeId	1033	WCHAR*	NULL	2000	XML formula element ID
wzHandlesId	1034	WCHAR*	NULL	2000	XML handle element ID
wzCalloutId	1035	WCHAR*	NULL	2000	XML callout element ID
wzLockId	1036	WCHAR*	NULL	2000	XML lock element ID
wzTextId	1037	WCHAR*	NULL	2000	XML text element ID

wzThreeDId	1038	WCHAR*	NULL	2000	XML threed element ID
FakeShapeType	1039	MSOSPT	msosptNil	2000	
fFakeMaster	1086	BOOL	FALSE	2000	This (master) shape is a fake from <shapetype>

Diagram

Properties related to Canvases and Diagrams

Property	PID	Type	Default	Ver	Description
dgmt	1280	MSODGMT	msodgmtNil	XP	Diagram/Canvas type
dgmStyle	1281	MSODGMSTYLE	msodgmstNil	XP	Diagram style
pRelationTbl	1284	IMsoArray	NULL	XP	Table of shape relationships. The array is a list of DGMRL structures. Each entry holds SPIDs for source, destination, and connector shapes
dgmScaleX	1285	LONG	1<<16	XP	Scale factor for width of diagram
dgmScaleY	1286	LONG	1<<16	XP	Scale factor for height of diagram
dgmDefaultFontSize	1287	LONG	-1	XP	Font size for new nodes
dgmConstrainBounds	1288	IMsoArray	NULL	XP	An array of four LONGs recording the left, top, right, bottom bounds that the nodes are constrained to.
dgmBaseTextScale	1289	LONG	1<<16	2003	Scale factor for base text size
fBorderlessCanvas	1338	BOOL	FALSE	2007	Indicates that the canvas is intended to be used for inking. The canvas does not have a border around it unless the user is actively inking. As soon as the user stops inking, the border goes away. All ink that was drawn over the canvas will stick to the canvas.
fNonStickyInkCanvas	1339	BOOL	FALSE	2007	A sub-property of fBorderlessCanvas; it won't exist on its own. A non-sticky canvas is a borderless canvas that ink does not necessarily bind to if the user inks over it. The ink should not crop if the user draws outside the canvas, and the canvas won't grow to accommodate the new stroke.
fDoFormat	1340	BOOL	FALSE	XP	True if auto format enabled
fReverse	1341	BOOL	FALSE	XP	True if diagram layout is reversed
fDoLayout	1342	BOOL	TRUE	XP	True if layout is needed

fPseudoInline	1343	BOOL	FALSE	XP	For Word’s Pseudo-inline feature

Line Left Style

On rectangular shapes, OfficeArt supports having different line styles of each of the four borders (left, top, right, bottom, and on the boundary between columns of text. When a rectangular shape has no line (fLine property is FALSE), the display engine will derive the line style for the four borders from the left, top, right and bottom property sets.

Thus there is a *Line Left Style* property set that controls the rendering of the left boundary of a rectangle, and is identical to the *Line Style* property set in every way, except that the properties count up from 1344 (lineLeftColor) instead of 448 (lineColor.)

Line Top Style

The *Line Top Style* property set controls the rendering of the top boundary of a rectangle, and is identical to the *Line Style* property set in every way, except that the properties count up from 1408 (lineTopColor) instead of 448 (lineColor.)

Line Right Style

The *Line Top Style* property set controls the rendering of the right boundary of a rectangle, and is identical to the *Line Style* property set in every way, except that the properties count up from 1472 (lineRightColor) instead of 448 (lineColor.)

Line Bottom Style

The *Line Bottom Style* property set controls the rendering of the bottom boundary of a rectangle, and is identical to the *Line Style* property set in every way, except that the properties count up from 1536 (lineBottomColor) instead of 448 (lineColor.)

Line Column Style

The *Line Column Style* property set controls the rendering of column borders between text, and is identical to the *Line Style* property set in every way, except that the properties count up from 1600 (lineLeftColor) instead of 448 (lineColor.)

Line Top Style

The *Line Top Style* property set controls the rendering of the top boundary of a rectangle, and is identical to the *Line Style* property set in every way, except that the properties count up from 1408 (lineLeftColor) instead of 448 (lineColor.)

Web Component

Properties related to Web Component functionality

Property	PID	Type	Default	Ver	Description
webComponentWzHtml	1664	WCHAR*	NULL	XP	HTML content of web comp.
webComponentWzName	1665	WCHAR*	NULL	XP	Friendly name of web comp.
webComponentWzUrl	1666	WCHAR*	NULL	XP	URL of web comp.

webComponentWzProperties	1667	WCHAR*	NULL	XP	Property string for web comp.
fIsWebComponent	1727	BOOL	FALSE	XP	TRUE if this shape is a web comp.

Clip

This property set defines an additional clipping path for a shape or group

Property	PID	Type	Default	Ver	Description
pVerticesClip	1728	IMsoArray	NULL	XP	Vertices of clipping path. Format is same as the of pVertices property in the Geometry property set
pSegmentInfoClip	1729	IMsoArray	NULL	XP	Segment info of clipping path. Format is same as the of pSegmentInfo property in the Geometry property set
shapePathClip	1730	MSOSHAPEPATH	msoshapeLinesClosed	XP	Type of clipping path. Format is same as the of shapePath property in the Geometry property set
fClipToWrap	1790	BOOL	FALSE	XP	If true, shape is clipped to its text tight wrap polygon
fClippedOK	1791	BOOL	FALSE	XP	If true, additional clipping is enabled

Ink

This property set defines properties for Tablet PC Ink data.

Property	PID	Type	Default	Ver	Description
pInkData	1792	IMsoInkData	NULL	2003	Ink stroke data for the shape. This data consists of **a.** 16 byte GUID (CLSID_InkDisp) followed by **b.** 4 byte unsigned integer, giving the size of the ink blob, followed by **c.** an ink blob. The ink blob is in the Tablet PC Ink Serialized Format (ISF), more details on the Ink Serialized Format can be found at http://download.microsoft.com/download/0/B/E/0BE8BDD7-E5E8-422A-ABFD-4342ED7AD886/InkSerializedFormat(ISF)Specification.pdf.
fInkAnnotation	1852	BOOL	FALSE	2003	TRUE if current shape is ink annotation
fHitTestInk	1853	BOOL	TRUE	2003	TRUE to allow hit testing on ink strokes
fRenderShape	1854	BOOL	FALSE	2003	TRUE to render geometry/path on a shape with ink (normally these are

					hidden for shapes with ink)
fRenderInk	1855	BOOL	FALSE	2003	TRUE to render ink

Signature

These properties specify that the shape is a signature line a document generated by Office 2007 or later. A signature line provides a visual representation of a signature in a document that is digitally signed. The properties in this set correspond to attributes of the "signatureline" element in the Office Open XML format. For more details on these properties, refer to the documentation of that element in the Office Open XML documentation.

Property	PID	Type	Default	Ver	Description
wzSigSetupId	1921	WCHAR*	NULL	2007	GUID representing a place to sign in the doc
wzSigSetupProvId	192	WCHAR*	NULL	2007	GUID representing the signature provider (e.g. Office default is GUID_NULL, Office stamp is well-defined – can look up if you need it, and third-party is CLSID)
wzSigSetupSuggSigner	1923	WCHAR*	NULL	2007	The first line to show in the signature line representing who is suggested as appropriate signer for the signature line.
wzSigSetupSuggSigner 2	1924	WCHAR*	NULL	2007	Like above, but the second line.
wzSigSetupSuggSigner Email	1925	WCHAR*	NULL	2007	Like above, but email address.
wzSigSetupSignInst	1926	WCHAR*	NULL	2007	Instructions to show the user in the signing ceremony.
wzSigSetupAddlXml	1927	WCHAR*	NULL	2007	generic string to store additional information with a signature line.
wzSigSetupProvUrl	1928	WCHAR*	NULL	2007	url to redirect user if the signature line is from a signature provider not installed.
fSigSetupShowSignDate	1980	BOOL	TRUE	2007	whether the sign date should be shown in the signed signature line.
fSigSetupAllowComments	1981	BOOL	FALSE	2007	whether the signing ceremony should allow comments to be entered.
fSigSetupSignInstSet	1982	BOOL	FALSE	2007	whether a suggested signer is set
fIsSignatureLine	1983	BOOL	FALSE	2007	whether the shape is a signature line object.

Group Shape 2

Relative position and size properties for shapes and textboxes in Word.

Property	PID	Type	Default	Ver	Description
pctHoriz	1984	LONG	msopctSizeNone	2007	Percentage width for a shape (Word)
pctVert	1985	LONG	msopctSizeNone	2007	Percentage height for a shape (Word)
pctHorizPos	1986	LONG	msopctPosNone	2007	Percentage horizontal position for a shape (Word)
pctVertPos	1987	LONG	mospctPosNone	2007	Percentage vertical position for a shape (Word)
sizerelh	1988	MSOSRH	msosrhPage	2007	relative size horizontal relation (Word)
sizerelv	1989	MSOSRV	msoprvPage	2007	relative size vertical relation (Word)
colStart	1990	LONG	0	2007	Starting column (Word)
colSpan	1991	LONG	0	2007	Number of columns to span (Word)

Appendix A: Change History

Date	Change
1/11/96	Created the document. Left work to be done in the BStore and Complex shape properties.
1/23/97	Extensively updated and reorganized document. Added overview and shape hierarchy information. Added more complete shape property information.
4/15/97	Add details on compression of Metafile Blips. See section "Metafile/PICT Blips"
5/21/99	Update document for Office 2000.
9/1/03	Update document for Office 2003.
12/15/06	Update document for Office 2007.

Appendix B: Colors

For each color in a property table there is a "main" property ID. A well formed file that specifies a value for a given color must always specify a four-byte LONG as the value for the "main" property table entry. The colors designated "Extended Color" also have *optional* "Extended" property IDs in the property table. The values assigned to these extended IDs can define the color more precisely in a different color model (say, CMYK.)

Thus, a color defined in the Office Drawing file format consists of one required "main" property ID and zero or more optional "extended" property IDs. For example, a fill color is defined by the "main" property 385 (fillColor) and the "extended" properties 414, 415, 416, 417, and 422 (fillllColorExt, fillColorExtCMY, fillColorExtMod, fillColorExtWzName, and fillColorExtK, respectively.) To reduce redundancy, the "extended" property table IDs are not listed in the tables in the main part of the

document. To see the complete listing of these IDs, refer to the section below discussing "extended" properties.

Interpreting the "main" property

Recall that this value is a four-byte LONG. If the high byte is zero, then the low three bytes contain a normal RGB COLORREF. Otherwise, the value of the high byte is used as a bitfield of flags (notice that the values below are the *indices* of the bits that are set).

```
typedef enum
   {
   msocolorFlagPaletteIndex,   // PALETTEINDEX macro
   msocolorFlagPaletteRGB,     // PALETTERGB macro
   msocolorFlagSystemRGB,      // MSOSYSTEMRGB
   msocolorFlagSchemeIndex,    // MSOSCHEMECOLOR
   msocolorFlagSysIndex,       // MSOSYSCOLOR
   } MSOCOLORINDEX;
```

Windows defines the first two. The third (SystemRGB) is a flag that it set on an otherwise normal RGB value to indicate to the rendering engine to bypass its normal halftone dithering process and just render the color directly using Windows.

The presence of either of the last two flags indicates that the low three bytes are an index into a predefined array of colors. For SchemeIndex colors, the host application provides the translation to RGB colors when necessary (PowerPoint and Excel both use this). SysIndex colors are indices into colors tracked by Escher itself:

```
typedef enum
   {
   msosyscolorButtonFace,          // COLOR_BTNFACE
   msosyscolorWindowText,          // COLOR_WINDOWTEXT
   msosyscolorMenu,                // COLOR_MENU
   msosyscolorHighlight,           // COLOR_HIGHLIGHT
   msosyscolorHighlightText,       // COLOR_HIGHLIGHTTEXT
   msosyscolorCaptionText,         // COLOR_CAPTIONTEXT
   msosyscolorActiveCaption,       // COLOR_ACTIVECAPTION
   msosyscolorButtonHighlight,     // COLOR_BTNHIGHLIGHT
   msosyscolorButtonShadow,        // COLOR_BTNSHADOW
   msosyscolorButtonText,          // COLOR_BTNTEXT
   msosyscolorGrayText,            // COLOR_GRAYTEXT
   msosyscolorInactiveCaption,     // COLOR_INACTIVECAPTION
   msosyscolorInactiveCaptionText, // COLOR_INACTIVECAPTIONTEXT
   msosyscolorInfoBackground,      // COLOR_INFOBK
   msosyscolorInfoText,            // COLOR_INFOTEXT
   msosyscolorMenuText,            // COLOR_MENUTEXT
   msosyscolorScrollbar,           // COLOR_SCROLLBAR
   msosyscolorWindow,              // COLOR_WINDOW
   msosyscolorWindowFrame,         // COLOR_WINDOWFRAME
   msosyscolor3DLight,             // COLOR_3DLIGHT
   msosyscolorMax,                 // Count of system colors

   msocolorFillColor =0xF0,  // Use the fillColor property
   msocolorLineOrFillColor,  // Use the line color only if there is a line
   msocolorLineColor,        // Use the lineColor property
   msocolorShadowColor,      // Use the shadow color
```

```
   msocolorThis,              // Use this color (only valid as described below)
   msocolorFillBackColor,     // Use the fillBackColor property
   msocolorLineBackColor,     // Use the lineBackColor property
   msocolorFillThenLine,      // Use the fillColor unless no fill and line
   msocolorIndexMask =0xFF,   // Extract the color index

   msocolorProcessMask      =0xFFFF00, // All the processing bits
   msocolorModificationMask =0x0F00,   // Just the function
   msocolorModFlagMask      =0xF000,   // Just the additional flags
   msocolorDarken      =0x0100,   // Darken color by parameter/255
   msocolorLighten     =0x0200,   // Lighten color by parameter/255
   msocolorAdd         =0x0300,   // Add grey level RGB(param,param,param)
   msocolorSubtract    =0x0400,   // Subtract grey level RGB(p,p,p)
   msocolorReverseSubtract =0x0500,   // Subtract from grey level RGB(p,p,p)
   /* In the following "black" means maximum component value, white minimum.
      The operation is per component, to guarantee white combine with
      msocolorGray */
   msocolorBlackWhite  =0x0600,   // Black if < uParam, else white (>=)
   msocolorInvert      =0x2000,   // Invert color (at the *end*)
   msocolorInvert128   =0x4000,   // Invert by toggling the top bit
   msocolorGray        =0x8000,   // Make the color gray (before the above!)
   msocolorBParamMask  =0xFF0000, // Parameter used as above
   msocolorBParamShift =16,       // To extract the parameter value
   }
MSOSYSCOLORINDEX;
```

Interpreting “extended” properties

For a given “Extended” color the following can be specified:

MSOCLR **color** – The “main” color property for this extended color.

- If none of the extended properties is set, it contains the full color definition which is either a straight RGB, or a simple scheme color, or a derived color. The format of this property is described in the section on “Interpreting the Main property”.
- If the other extended properties are set, this field must contain the ‘resolved’ RGB that is computed by applying *colorExtMod* to *colorExt*. When the main and extended values for a given color are inconsistent, (i.e. *colorExt* does not match the sRGB translation of *colorExtCMY* / *colorExtK* / *colorExtWzName*) an implementer should respect the main value and discard the extended values.
- **In either case, implementers interested only in the sRGB color space (the usual case) can use the value of this property as the color and ignore all the extended color properties.**

MSOCLR **colorExt** – this is base color. Resolving it (applying *colorExtMod* modification to it) we will get *color*. It’s used in the following situations:

- if *colorExtWzName* is set, this must contain the RGB approximation of the CMS color.
- if *colorExtCMY* and *colorExtK* are set, this must contain the RGB approximation of the CMYK or spot color they contain.
- Otherwise, contains either a ‘base’ RGB or ‘base’ scheme color. *colorExtMod* should be tint or shade color modification.

MSOCLR **colorExtMod** – if set, contains a tint or shade (color modification) to apply to *colorExt*. This should be set if *colorExt* is set, otherwise it should be **MSOCOLORMODUNDEFINED**.

LONG **colorExtCMY, colorExtK** – these two properties comprise the low-order (*colorExtCMY*) and high-order (*colorExtK*) bits of an **MSOINKCOLOR** (this type is defined below). They are actually mis-named as they can contain information other than CMYK. Used in the following cases:

- If *colorExtWzName* is set, these properties must contain the CMYK approximation of the CMS color.
- Otherwise, if set, they contain a CMYK color and/or spot color(s). (Again, see MSOINKCOLOR for full details.)

WCHAR ***colorExtWzName** - if set, contains a string that identifies a CMS (Color Management System) color. The format of the string is documented in the MSO headers, but only the UI code really cares.

Here are the sets of property IDs that comprise each extended color:

Property	color	colorExt	colorExtMod	colorExtCMY	colorExtK	colorExtWzName
pictureTransparent	263	277	279	278	281	280
pictureRecolor	282	283	285	284	287	286
fillColor	385	414	416	415	422	417
fillBackColor	387	418	420	419	423	421
lineColor	448	473	475	474	481	476
lineBackColor	450	477	479	478	482	480
shadowColor	513	530	532	531	538	533
shadowHighlight	514	534	536	535	539	536
c3DExtrusionColor	647	649	651	650	653	652

The main color properties were all added in Office 97 (hence they should be saved in Block 1.) The extended color properties first appeared in Office XP, and so should be written in Block 3.

MSOINKCOLOR

Finally, here is the definition of the 64-bit MSOINKCOLOR type.

- Bit 0 (lowest order) is an "overprinting" flag. If set it is a hint that when printing color separations, ink from objects underneath this object does not need to be removed.
- Bit 1 is reserved and should be zero.
- Bits 2-4 determine the format of the data stored in bits 6 and up. The possibilities are:

```
typedef enum
   {
   msoprocessNone,    // No process - color just contains spot/named inks
   msoprocesssRGB,    // sRGB64 - actually three 16 bit values, R,G,B.
   msoprocessCMYK,    // CMYK as defined by our canonical ICM profile
   }
MSOPROCESS;
```

- If the format is msoprocessNone, *spot ink values* (see below) follow starting at bit 5.

- If the format is msoprocesssRGB, a color in the sRGB64 color space follows, with the red channel in bits 5-20, green channel in bits 21-36, and blue channel in bits 37-52. No *spot ink values* will follow.
- If the format is msoprocessCMYK, a packed CMYK color follows. Bits 5-8 determine which of the Cyan (bit 8), Magenta (bit 7), Yellow (bit 6), and Black (bit 5) inks is present in the color. For each bit that is set, the amount of the corresponding ink present in this MSOINKCOLOR will appear as an 8 bit value in the range 0-255. No bits are used for inks that are not present. The order of the colors (from low-order bits to high-order bits) is C, M, Y, K (=black.) Zero or more *spot ink values* follow.

Spot ink values are stored as an *ink index increment* (low-order bits) followed by an *ink amount* (high-order bits.) The ink index increment's three lowest order bits are a number taking the following values:

0: end of list; no more spot ink values

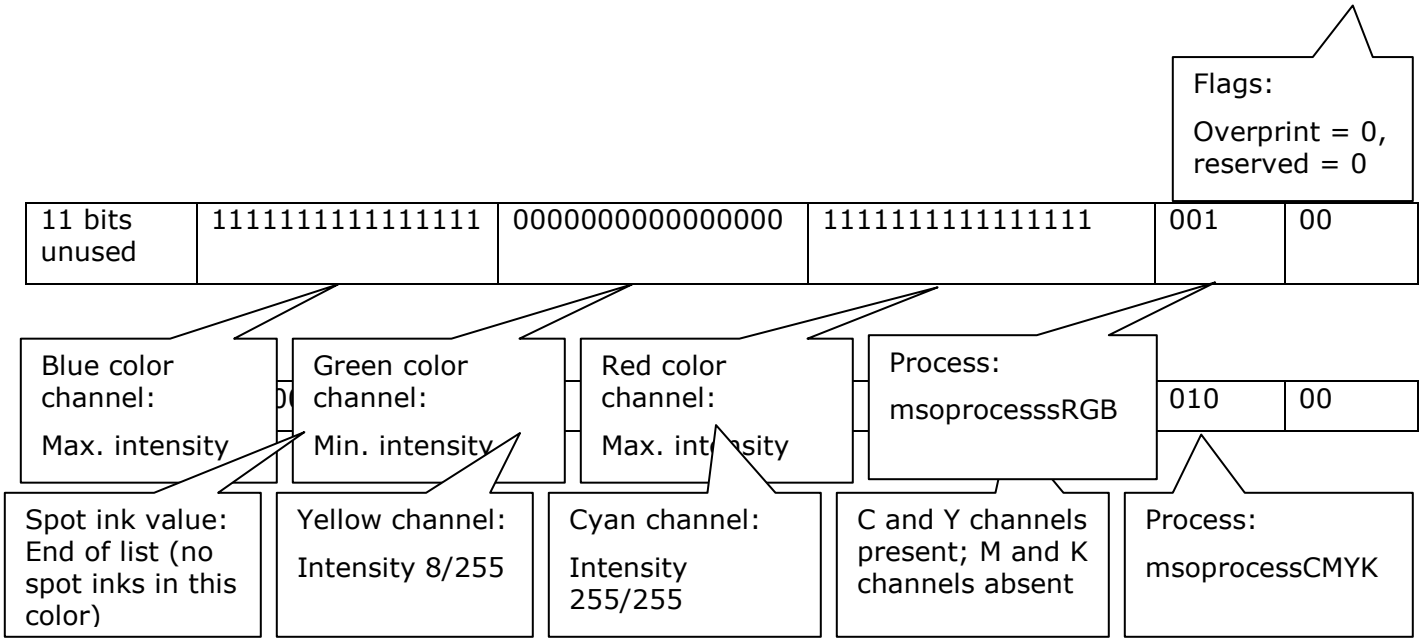
1..6: This value is the ink index increment

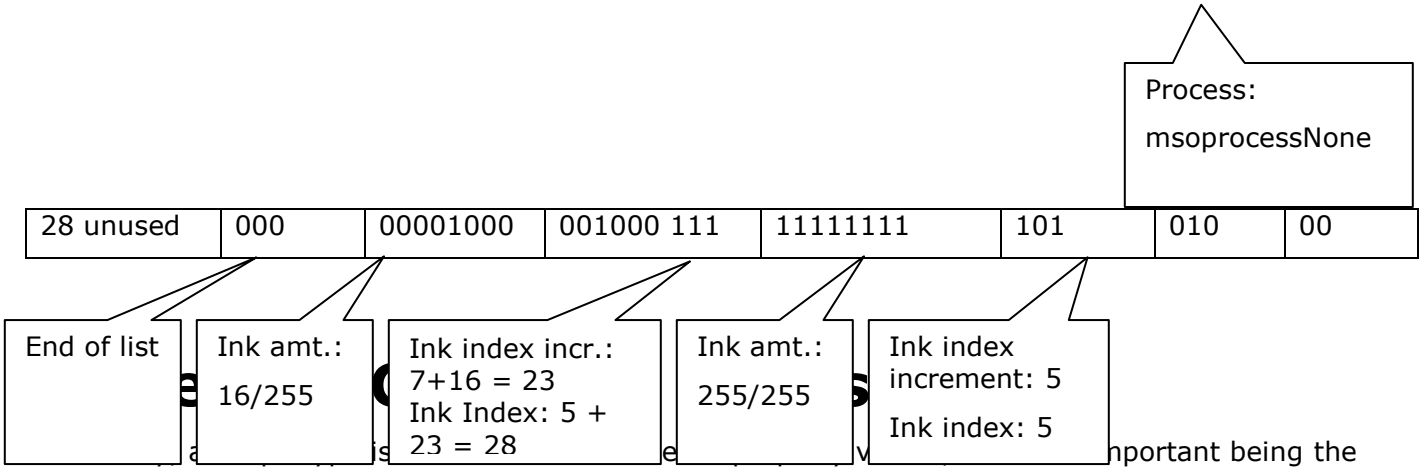
7: 6 more bits of ink index increment follow. The total ink index increment for this spot ink value is 7 + (the number in these 6 bits.) The maximum permissible value for an ink index increment is 63.

As an implementation scans an MSOINKCOLOR from low-order to high-order bits, it must maintain an **ink index**, initialized to zero. Each time it encounters an ink index *increment*, it should add the increment to the current ink index to obtain the next ink index. The ink index is an index into a table of inks stored by the host application (as of Office 2003, only Publisher supports spot ink values.) The format of this table of inks and its location in the binary file are external to OfficeArt and therefore are not described in this document.

The **ink amount** is an 8 bit number in the range 0-255, with 0 denoting no ink and 255 denoting maximum ink coverage.

Exempli Gratia: Here are three sample MSOINKCOLOR structures to illustrate the definitions above.





geometry of the shape (the pVertices property, etc.). Each shape stores in itself only those properties that differ from its shape type. When a shape is asked for a property that isn't in its local table, it looks in the shape type's table. If the shape type doesn't define a value for the property, then the property's default value is used.

In the file format, shapes store their shape types in the instance field of the msofbtSp record.

```
typedef enum
    {
    msosptMin = 0,
    msosptNotPrimitive = msosptMin,
    msosptRectangle = 1,
    msosptRoundRectangle = 2,
    msosptEllipse = 3,
    msosptDiamond = 4,
    msosptIsocelesTriangle = 5,
    msosptRightTriangle = 6,
    msosptParallelogram = 7,
    msosptTrapezoid = 8,
    msosptHexagon = 9,
    msosptOctagon = 10,
    msosptPlus = 11,
    msosptStar = 12,
    msosptArrow = 13,
    msosptThickArrow = 14,
    msosptHomePlate = 15,
    msosptCube = 16,
    msosptBalloon = 17,
    msosptSeal = 18,
    msosptArc = 19,
    msosptLine = 20,
    msosptPlaque = 21,
    msosptCan = 22,
    msosptDonut = 23,
    msosptTextSimple = 24,
    msosptTextOctagon = 25,
    msosptTextHexagon = 26,
    msosptTextCurve = 27,
    msosptTextWave = 28,
    msosptTextRing = 29,
    msosptTextOnCurve = 30,
    msosptTextOnRing = 31,
    msosptStraightConnector1 = 32,
    msosptBentConnector2 = 33,
    msosptBentConnector3 = 34,
    msosptBentConnector4 = 35,
    msosptBentConnector5 = 36,
    msosptCurvedConnector2 = 37,
    msosptCurvedConnector3 = 38,
    msosptCurvedConnector4 = 39,
    msosptCurvedConnector5 = 40,
    msosptCallout1 = 41,
    msosptCallout2 = 42,
    msosptCallout3 = 43,
    msosptAccentCallout1 = 44,
    msosptAccentCallout2 = 45,
    msosptAccentCallout3 = 46,
    msosptBorderCallout1 = 47,
    msosptBorderCallout2 = 48,
    msosptBorderCallout3 = 49,
    msosptAccentBorderCallout1 = 50,
    msosptAccentBorderCallout2 = 51,
    msosptAccentBorderCallout3 = 52,
    msosptRibbon = 53,
    msosptRibbon2 = 54,
    msosptChevron = 55,
    msosptPentagon = 56,
```

```
msosptNoSmoking = 57,
msosptSeal8 = 58,
msosptSeal16 = 59,
msosptSeal32 = 60,
msosptWedgeRectCallout = 61,
msosptWedgeRRectCallout = 62,
msosptWedgeEllipseCallout = 63,
msosptWave = 64,
msosptFoldedCorner = 65,
msosptLeftArrow = 66,
msosptDownArrow = 67,
msosptUpArrow = 68,
msosptLeftRightArrow = 69,
msosptUpDownArrow = 70,
msosptIrregularSeal1 = 71,
msosptIrregularSeal2 = 72,
msosptLightningBolt = 73,
msosptHeart = 74,
msosptPictureFrame = 75,
msosptQuadArrow = 76,
msosptLeftArrowCallout = 77,
msosptRightArrowCallout = 78,
msosptUpArrowCallout = 79,
msosptDownArrowCallout = 80,
msosptLeftRightArrowCallout = 81,
msosptUpDownArrowCallout = 82,
msosptQuadArrowCallout = 83,
msosptBevel = 84,
msosptLeftBracket = 85,
msosptRightBracket = 86,
msosptLeftBrace = 87,
msosptRightBrace = 88,
msosptLeftUpArrow = 89,
msosptBentUpArrow = 90,
msosptBentArrow = 91,
msosptSeal24 = 92,
msosptStripedRightArrow = 93,
msosptNotchedRightArrow = 94,
msosptBlockArc = 95,
msosptSmileyFace = 96,
msosptVerticalScroll = 97,
msosptHorizontalScroll = 98,
msosptCircularArrow = 99,
msosptNotchedCircularArrow = 100,
msosptUturnArrow = 101,
msosptCurvedRightArrow = 102,
msosptCurvedLeftArrow = 103,
msosptCurvedUpArrow = 104,
msosptCurvedDownArrow = 105,
msosptCloudCallout = 106,
msosptEllipseRibbon = 107,
msosptEllipseRibbon2 = 108,
msosptFlowChartProcess = 109,
msosptFlowChartDecision = 110,
msosptFlowChartInputOutput = 111,
msosptFlowChartPredefinedProcess = 112,
msosptFlowChartInternalStorage = 113,
msosptFlowChartDocument = 114,
msosptFlowChartMultidocument = 115,
msosptFlowChartTerminator = 116,
msosptFlowChartPreparation = 117,
msosptFlowChartManualInput = 118,
msosptFlowChartManualOperation = 119,
msosptFlowChartConnector = 120,
msosptFlowChartPunchedCard = 121,
msosptFlowChartPunchedTape = 122,
msosptFlowChartSummingJunction = 123,
msosptFlowChartOr = 124,
msosptFlowChartCollate = 125,
msosptFlowChartSort = 126,
msosptFlowChartExtract = 127,
msosptFlowChartMerge = 128,
msosptFlowChartOfflineStorage = 129,
msosptFlowChartOnlineStorage = 130,
msosptFlowChartMagneticTape = 131,
msosptFlowChartMagneticDisk = 132,
msosptFlowChartMagneticDrum = 133,
msosptFlowChartDisplay = 134,
msosptFlowChartDelay = 135,
msosptTextPlainText = 136,
msosptTextStop = 137,
msosptTextTriangle = 138,
msosptTextTriangleInverted = 139,
msosptTextChevron = 140,
msosptTextChevronInverted = 141,
msosptTextRingInside = 142,
msosptTextRingOutside = 143,
msosptTextArchUpCurve = 144,
msosptTextArchDownCurve = 145,
msosptTextCircleCurve = 146,
msosptTextButtonCurve = 147,
msosptTextArchUpPour = 148,
msosptTextArchDownPour = 149,
msosptTextCirclePour = 150,
msosptTextButtonPour = 151,
msosptTextCurveUp = 152,
msosptTextCurveDown = 153,
msosptTextCascadeUp = 154,
msosptTextCascadeDown = 155,
msosptTextWave1 = 156,
msosptTextWave2 = 157,
msosptTextWave3 = 158,
msosptTextWave4 = 159,
msosptTextInflate = 160,
msosptTextDeflate = 161,
msosptTextInflateBottom = 162,
msosptTextDeflateBottom = 163,
msosptTextInflateTop = 164,
```

```
    msosptTextDeflateTop = 165,
    msosptTextDeflateInflate = 166,
    msosptTextDeflateInflateDeflate = 167,
    msosptTextFadeRight = 168,
    msosptTextFadeLeft = 169,
    msosptTextFadeUp = 170,
    msosptTextFadeDown = 171,
    msosptTextSlantUp = 172,
    msosptTextSlantDown = 173,
    msosptTextCanUp = 174,
    msosptTextCanDown = 175,
    msosptFlowChartAlternateProcess = 176,
    msosptFlowChartOffpageConnector = 177,
    msosptCallout90 = 178,
    msosptAccentCallout90 = 179,
    msosptBorderCallout90 = 180,
    msosptAccentBorderCallout90 = 181,
    msosptLeftRightUpArrow = 182,
    msosptSun = 183,
    msosptMoon = 184,
    msosptBracketPair = 185,
    msosptBracePair = 186,
    msosptSeal4 = 187,
    msosptDoubleWave = 188,
    msosptActionButtonBlank = 189,
    msosptActionButtonHome = 190,
    msosptActionButtonHelp = 191,
    msosptActionButtonInformation = 192,
    msosptActionButtonForwardNext = 193,
    msosptActionButtonBackPrevious = 194,
    msosptActionButtonEnd = 195,
    msosptActionButtonBeginning = 196,
    msosptActionButtonReturn = 197,
    msosptActionButtonDocument = 198,
    msosptActionButtonSound = 199,
    msosptActionButtonMovie = 200,
    msosptHostControl = 201,
    msosptTextBox = 202,
    msosptMax,
    msosptNil = 0x0FFF,
} MSOSPT;
```

Appendix D: AutoShapes

Introduction

This document describes each of the AutoShapes available in Microsoft Office. It provides a detailed description of each shape, including its geometric properties. This document is intended for use in conjunction with the *Office Drawing File Format* specification, which documents the binary file format used to represent shapes.

The first section of this document describes the meaning of each shape property and provides a mapping to its representation in the binary file format. The section of this document lists each AutoShape and its individual properties.

AutoShapes

AutoShapes are drawing objects with a particular shape that may be customized through smart resizing and adjustments. Multiple disjoint paths and subpaths, and quadratic and cubic curves give shapes rich geometry. Multiple adjust handles which can adjust in two dimensions anywhere inside or outside of a shape, and a robust set of formulas allow very smart adjustment behaviors to be defined. In addition, AutoShapes may contain text sized to fit within the shape.

AutoShapes have been designed with the following consistency guidelines:

- Left or top perimeter. Perimeter adjust handles are used when only one degree of freedom needs to be adjusted, and the vertices that are adjusted are near the perimeter of the shape. Perimeter adjust handles are placed so that they never overlap resize handles. If the shape is symmetrical, the perimeter adjust handle is placed on the left or top perimeter instead of the right or bottom.

- Inner adjustments. For most shapes, the adjust handles do not change the bounding box of the shape; rather, they make an internal adjustment within the bounding box of the shape. Exceptions to this rule include the callout shapes, for which it is useful to place the point independent of the box, and the arc shape.

- Arrow adjust handle. Most of the block arrow shapes use a single adjust handle that controls both the length of the arrowhead and the width of the trunk. Some block arrow shapes have other adjust handles in addition to this one.

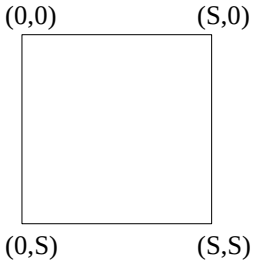
- Limo-stretch. Some shapes that have portions that should be constrained to a fixed aspect ratio are designed with limo-stretch to keep those portions at the fixed aspect ratio.

Description of Shape Properties

A table describing the properties of each AutoShape follows this section. In the table, AutoShapes are organized into six categories:

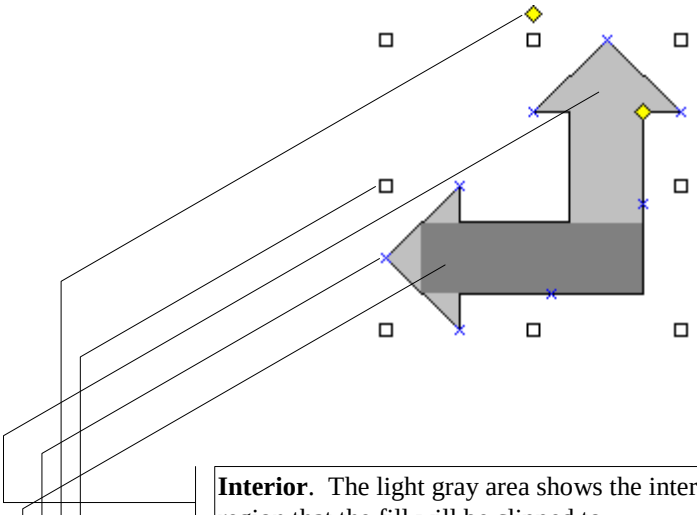
- Basic Shapes
- Block Arrows
- Flowchart
- Stars & Banners
- Callouts
- Action Buttons

This section describes each property that may be specified for each shape. All coordinates are given in shape coordinates: the origin is at the top-left, positive-x is to the right, positive-y is down. The region from (0,0) to (S,S) maps to the geometry box of the shape (S=21600 is a constant). Adjust values are "inputs" which are usually determined by the location of the adjust handles.



Shape Diagrams

For each shape, a diagram of the shape is shown, and a list of information about the shape is given. The diagram shows the following information:



Interior. The light gray area shows the interior of the shape. This is the region that the fill will be clipped to.

Text rectangle. The dark gray area shows the rectangular region in which the attached textbox is placed (this region isn't really filled differently; it is just shown in dark grey for this diagram).

Connection sites. The blue Xs show the locations at which a connector can attach to this shape. The connection sites adjust with the shape as the adjust handles are used. Connection sites are numbered counterclockwise starting from the top or left side.

Adjust handles. The yellow diamonds show the handles which you can drag to adjust the shape. Each adjust handle can control either one adjust value (horizontal or vertical) or two adjust values (both horizontal and vertical). The adjust handles are shown in their default location for new shapes.

Resize handles. The white squares show the handles which you can drag to resize the shape. The resize handles are shown around the "geometry box" of the shape (when you create a shape by dragging out, you are specifying the position and size of the geometry box). Most shapes are contained entirely within their geometry box, but some extend past it. Most shapes extend to each edge of their geometry box, but some do not.

Internal Name

The friendly name of the shape. This is name is used for the name of the command bar button (and the tooltip for it) and for the default value of the Name property in VBA.

Shaped Concentric Fill

Specifies whether or not this shape will use a shaped concentric fill (i.e. the shape is rendered at various different sizes to create a concentric gradient fill). If not, the shape will use a rectangular concentric fill.

This values corresponds to the `fFillShadeShapeOK` variable in the shape's geometry properties.

Joins

Specifies what join style the shape has. Since there is no UI for changing the join style, all shapes of this type will always have the specified join style.

This values corresponds to the `lineJoinStyle` variable in the shape's line style properties.

Endcaps

Specifies what endcap style the shape has. Since there is no UI for changing the endcap style (except for the implicit change in endcap style caused by changing the dash style), all shapes of this type will always have the specified endcap style (except when certain dash styles are used).

This values corresponds to the `lineEndCapStyle` variable in the shape's line style properties.

Adjustments

Describes the behavior of the adjustments including a brief description of what the adjust handles do, the range of the adjust handles, and whether or not the shape has limo-stretch behavior.

Path

This string describes a sequence of commands that define the shape's path. This string describes both the `pSegmentInfo` array and `pVertices` array in the shape's geometry properties.

The following rules apply to path strings:

- Commas or spaces delimit parameters for each command. Both "m 0,0" and "m0 0" are acceptable.
- A parameter that is omitted using commas is treated as having a value of zero. Thus, "c 10,10,0,0,25,13" and "c 10,10,,,25,13" are equivalent.
- Parameterized paths are also allowed. In this case, the shape must also have a list of guide formulas that are substituted into the path using the @ symbol followed by the number of the formula. The adj property of the shape contains the input parameters for these formulas. For example, "moveto @1@4". The evaluations of the formulas are substituted into the appropriate positions. Note that @ also serves as a delimiter.

The allowed commands are given below. An asterisk (*) indicates that the command is allowed to be repeated. For the `qb` command, the controlpoint parameter is also allowed to be repeated.

Command	Name	Parameters	Description
`m`	moveto	2	Start a new sub-path at the given (x,y) coordinate.

Command	Name	Parameters	Description
l	lineto	2*	Draw a line from the current point to the given (x,y) coordinate which becomes the new current point. Specifying a number of coordinate pairs forms a polyline.
c	curveto	6*	Draw a cubic bézier curve from the current point to the coordinate given by the final two parameters. The control points are given by the first four parameters.
x	close	0	Close the current sub-path by drawing a straight line from the current point to the original moveto point.
e	end	0	End the current set of sub-paths. A given set of sub-paths (as delimited by end) is filled. Subsequent sets of sub-paths are filled independently and superimposed on existing ones.
t	rmoveto	2*	Start a new sub-path at a coordinate relative to the current point, cp (cpx+x, cpy+y).
r	rlineto	2*	Draw a line from the current point to the given relative coordinate (cpx+x, cpy+y).
v	rcurveto	6*	Cubic bézier curve using the given coordinate relative to the current point.
nf	nofill	0	The current set of sub-paths (delimited by e) will not be filled.
ns	nostroke	0	The current set of sub-paths (delimited by e) will not be stroked.
ae	angleellipseto	6*	Draws a segment of an ellipse as described using these parameters. A straight line is drawn from the current point to the start point of the segment. The parameters are: center (x,y), size(w,h), start angle, end angle.
al	angleellipse	6*	Same as angleellipseto except that there is an implied moveto the starting point of the segment.

Command	Name	Parameters	Description
at	arcto	8*	A segment of the ellipse is drawn which starts at the angle defined by the start radius vector and ends at the angle defined by the end vector. A straight line is drawn from the current point to the start of the arc. The arc is always drawn in a counterclockwise direction. The parameters are: left, top, right, bottom, start(x,y), end(x,y). The first four values define the bounding box of an ellipse. The last four define two radial vectors.
ar	arc	8*	Same as arcto except there is an implied moveto the start point of the arc.
wa	clockwisearcto	8*	Same as arcto but the arc is drawn in a clockwise direction.
wr	clockwisearc	8*	Same as arc but the arc is drawn in a clockwise direction
qx	ellipticalqaudrantx	2*	A quarter ellipse is drawn from the current point to the given end point. The elliptical segment is initially tangential to a line parallel to the x-axis. (i.e. the segment starts out horizontal). The parameters are: end(x,y).
qy	ellipticalquadranty	2*	Same as ellipticalquadrantx except that the elliptical segment is initially tangential to a line parallel to the y-axis (i.e. the segment starts out vertical).
qb	quadraticbezier	2+2*	Defines one or more quadratic bézier curves by means of control points and an end point. Intermediate (on-curve) points are obtained by interpolation between successive control points as in the OpenType font specification. The sub-path need not be started in which case the sub-path will be closed. In this case the last point of the sub-path defines the start point of the quadratic bézier. The parameters are: controlpoint(x,y)*, end(x,y).

Each command corresponds to an entry in the `pSegmentInfo` array, while each parameter represents a set of (x,y) coordinates that correspond to an entry in the `pVertices` array in the shape's geometry properties.

Guide Formulas

This specifies a list of formulas whose calculated values are referenced by other properties. Each formula is listed on a separate line. Formulas are ordered, with the first formula having index 0. This section can be omitted if the shape doesn't need any guides.

The result of each calculation is referenced using @ followed by a number corresponding to the zero-based index for that formula in the list of formulas.

The list of formulas corresponds to the `pGuides` array in the shape's geometry properties. Individual formulas correspond to an instance of the `SG` (Shape Guide) structure.

The following list describes the set of possible formulas:

- `#` followed by an integer specifies the adjustment value with the given zero-basedindex in the list of adjustment values.
- `@` followed by an integer specifies another guide formula in the list by its zero-based index in the list.
- *`var`* is the name of a variable. Variables can be of the form `x`*`n`*, `y`*`n`*, or `t`*`n`* where *`n`* is an integer between `0` and `127`.
- *`const`* is a constant. Constants can be integers between `0` and `65535` (unsigned 16-bit integers) or of the form `SD`*`n`* or `SD`*`n`*`T`*`m`* where *`n`* and *`m`* are integers. `SD`*`n`* = `S/`*`n`* and `SD`*`n`*`T`*`m`* = `(S/`*`n`*`)*`*`m`*.
- *`val`* is a value. Values can be constants or variables:
  - Constants are described above.
  - Variables can be any of the variable names that are defined in the `guides` section. For values in the `guides` section, you can only reference variables that have been defined earlier in the section. If there is no `guides` section, you cannot use variables as values.
- In the guides section only, *`val`* can also be one of the following:
  - Adjust values (of the form `adjust`*`n`* where *`n`* is an integer between `1` and `8`). Must be ones that are defined in the `adjust handles` section.
  - `width` or `height`. Variables that represent the width and height of the shape in shape coordinates. For non-limo-stretch shapes, the constant `S` can be used instead, but for limo-stretch these variables may be needed.
- *`func`* is a function that can be used in defining the value of a variable in the `guides` section. The full list of functions that can be used is given in the following table:

Function	Name	Formula
Add and Subtract	`sum`	`a + b - c`
Multiply and Divide	`product`	`a * b / c`
Midpoint	`mid`	`(a + b) / 2`
Absolute Value	`absolute`	`abs(a)`
Minimum	`min`	`min(a,b)`
Maximum	`max`	`max(a,b)`
If	`if`	`a > 0 ? b : c`
Square Root	`sqrt`	`sqrt(a)`
Modulus	`mod`	$\texttt{sqrt}(a^2 + b^2 + c^2)$
Sine	`sin`	`a * sin(b)`
Cosine	`cos`	`a * cos(b)`
Tangent	`tan`	`a * tan(b)`
ArcTangent	`atan2`	`atan2(b,a)`
Sine of ArcTangent	`sinatan2`	`a * sin(atan2(c,b))`
Cosine of ArcTangent	`cosatan2`	`a * cos(atan2(c,b))`

Angle Add and Subtract	`sumangle`	`a + b° - c°`
Ellipse Intersection	`ellipse`	`c * sqrt(1-(a/b)²)`

Adjustment Values

Specifies a comma-delimited list of parameters, or adjustment values, used to define values for a parameterized formula. These values represent the location of an adjust handle and may be referenced by the geometry of an adjust handle or as a parameter guide function.

Values may be omitted. Each value is referenced using # followed by a number corresponding to the zero-based index for that value in the list of adjustment values.

These values correspond to `adjustValue`, `adjust2Value`, `adjust3Value`, `adjust4Value`, `adjust5Value`, `adjust6Value`, `adjust7Value`, `adjust8Value`, `adjust9Value`, and `adjust10Value` in the shape's geometry properties.

Connector Locations

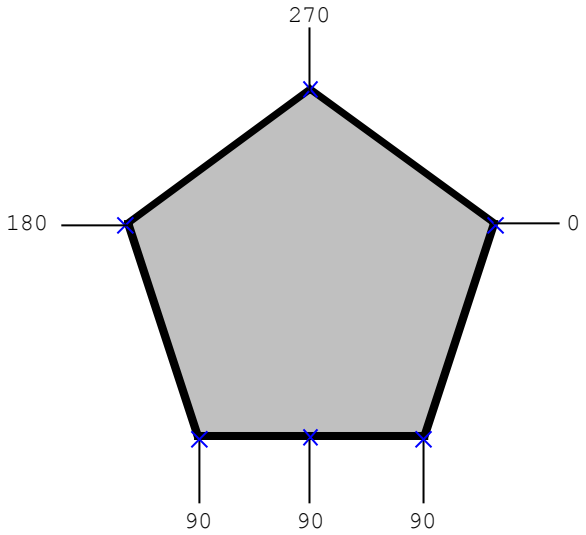
These values specify the location of connection points on the shape's path. The connection points are defined by a string consisting of pairs of x and y values, delimited by commas.

The value "Rectangle" specifies that the shape has four default connection sites; one at each midpoint of the shape's top, bottom, left, and right sides.

These values correspond to the `pConnectionSites` array in the shape's geometry properties.

Connector Angles

Associated with each connection site, there is a direction which specifies at what angle elbow and curved connectors should attach to it:



These are listed in the same order as the connection sites that they correspond to. Values are specified in degrees and separated by commas. If this property is omitted, the direction

for each site is computed automatically as the angle from the geometric center of the shape to the connection site.

These values correspond to the `pConnectionSitesDir` array in the shape's geometry properties.

Text Box Rectangle

Specifies one or more text boxes inscribed inside the shape. A textbox is defined by one or more sets of numbers specifying (in order) the left, top, right, and bottom points of the rectangle. Multiple sets are delimited by a semicolon. If omitted, the text box is the same as the geometry's bounding box.

For normal shapes, the same text rectangle is used for both horizontal and vertical text; for shapes with two text rectangles, the first text rectangle is used for horizontal text and the second is used for vertical text.

These values correspond to entries in the `pInscribe` array in the shape's geometry properties.

Handles

This section specifies the properties of each adjust handle on the shape. One adjust handle is specified per line. The properties for each handle correspond to values of the `ADJH` structure contained in the `pAdjustHandles` array in the shape's geometry properties. The following properties are specified if different than their default values:

Attribute	Description
position	Specifies the x and y position of the handle. If the polar attribute is present, defines the handle position using radius and angle values. Default is "0,0". Each values is one of the following: • constant • formula (e.g., @2) • adjustment value (e.g., #2) • center • topleft • bottomright Each of the above except for an adj value reference fixes the handle position for that dimension. Specifying an adjustment value allows the handle to move in that dimension and the handle position for that dimension is stored in the adjustment value. These values correspond to the `apX` and `apY` variables in the `ADJH` structure.
polar	Specifies the center position of a handle that uses polar coordinates. If specified, the position attribute is assumed to contain radius and angle values. If omitted, the position attribute is assumed to contain x and y positions. Default is 0,0. These values correspond to the `xRange` and `yRange` variables in the `ADJH` structure.

xrange	Specifies a range of minimum and maximum values that constrain the x position of a handle. Default is "0,0". Each value is either a constant or a formula reference. Omitting a value leaves that bound unconstrained. These values correspond to the `xMin` and `xMax` variables in the `ADJH` structure.
yrange	Specifies a range of minimum and maximum values that constrain the y position of a handle. Default is "0,0". Each value is either a constant or a formula reference. Omitting a value leaves that bound unconstrained. These values correspond to the `yMin` and `yMax` variables in the `ADJH` structure.
radiusrange	Specifies a range of minimum and maximum values that constrain the radius of a handle using polar coordinates. Default is "0,0". Each value is either a constant or a formula reference. Omitting a value leaves that bound unconstrained. These values correspond to the `xMin` and `xMax` variables in the `ADJH` structure, when the adjust handle type is `PolarPin`.
switch	Specifies whether the x and y dimensions of the handle are switched when the shape is taller than it is wide. Default is `false`. This is useful for shapes with limo stretch behavior. If true, the `SwitchPosition` property is added to the adjust handle type, specified by the variable `f` in the `ADJH` structure. For example, Pin becomes `Pin | SwitchPosition`.

The above values also determine the adjust handle type, specified by the `f` variable in the `ADJH` structure. By default, the type is `Pin`. If the polar attribute but the radiusrange attribute is not present, the type is `Polar`. If the polar attribute and radiusrange attributes are present, the type is `PolarPin`.

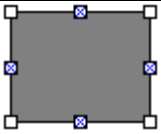
Limo

Specifies the (x,y) coordinates of the limo stretch point. Some shapes that have portions that should be constrained to a fixed aspect ratio, are designed with limo-stretch to keep those portions at the fixed aspect ratio. If omitted, the shape has no limo stretch point.

This value corresponds to the xLimo and yLimo variables in the shape's geometry properties.

Description of Shape Properties

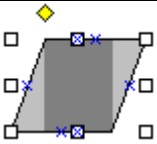
Basic Shapes



Rectangle
Internal Name: Rect
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

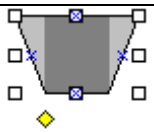
Path	m,l,21600r21600,l21600,xe
Connector Locations	Rectangle



Parallelogram
Internal Name: Parallelogram
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top perimeter adjust handle controls the amount of skew.

Geometric properties:

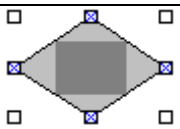
Path	m@0,l,21600@1,21600,21600,xe
Guide Formulas	val #0 sum width 0 #0 prod #0 1 2 sum width 0 @2 mid #0 width mid @1 0 prod height width #0 prod @6 1 2 sum height 0 @7 prod width 1 2 sum #0 0 @9 if @10 @8 0 if @10 @7 height
Adjustment Values	5400
Connector Locations	@4,0;10800,@11;@3,10800;@5,21600;10800,@12;@2,10800
Text Box Rectangle	1800,1800,19800,19800;8100,8100,13500,13500;10800,10800,10800,10800
Handles	position="#0,topLeft" xrange="0,21600"



Trapezoid
Internal Name: Trapezoid
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A bottom perimeter adjust handle controls the length of the bottom edge. The adjust handle can extend halfway across.

Geometric properties:

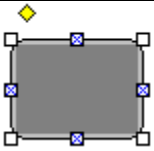
Path	m,l@0,21600@1,21600,21600,xe
Guide Formulas	val #0 sum width 0 #0 prod #0 1 2 sum width 0 @2 mid #0 width mid @1 0 prod height width #0 prod @6 1 2 sum height 0 @7 prod width 1 2 sum #0 0 @9 if @10 @8 0 if @10 @7 height
Adjustment Values	5400
Connector Locations	@3,10800;10800,21600;@2,10800;10800,0
Text Box Rectangle	1800,1800,19800,19800;4500,4500,17100,17100;7200,7200,14400,14400
Handles	position="#0,bottomRight"; xrange="0,10800"



Diamond
Internal Name: Diamond
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

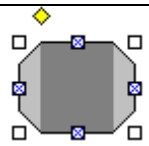
Path	m10800,l,10800,10800,21600,21600,10800xe
Connector Locations	Rectangle
Text Box Rectangle	5400,5400,16200,16200



Rounded Rectangle
Internal Name: RRect
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the roundedness of the corners. It limo-stretches both vertically and horizontally at the midpoint, so that the rounded corners are always circular. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend halfway across.

Geometric properties:

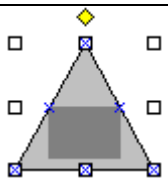
The rounded rectangle shape is a geometric primitive. It uses default rectangle connection points and its one adjust handle adjusts the radius of the corners.



Octagon
Internal Name: Octagon
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount cropped off of the corners. It limo-stretches both vertically and horizontally at the midpoint, so that the corners are always at 45°. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension.

Geometric properties:

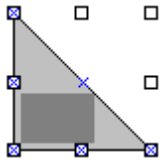
Path	m@0,l0@0,0@2@0,21600@1,21600,21600@2,21600@0@1,xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod @0 2929 10000 sum width 0 @3 sum height 0 @3 val width val height prod width 1 2 prod height 1 2
Adjustment Values	6326
Connector Locations	@8,0;0,@9;@8,@7;@6,@9
Text Box Rectangle	0,0,21600,21600;2700,2700,18900,18900;5400,5400,16200,16200
Handles	position="#0,topLeft" switch="true" xrange="0,10800"
Limo	10800,10800



Isosceles Triangle
Internal Name: IsoscelesTriangle
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top perimeter adjust handle controls the location of the top vertex.

Geometric properties:

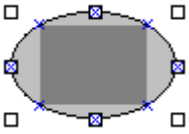
Path	m@0,l,21600r21600,xe
Guide Formulas	val #0 prod #0 1 2 sum @1 10800 0
Adjustment Values	10800
Connector Locations	@0,0;@1,10800;0,21600;10800,21600;21600,21600;@2,10800
Text Box Rectangle	0,10800,10800,18000;5400,10800,16200,18000;10800,10800,21600,18000;0,7200,7200,21600;7200,7200,14400,21600;14400,7200,21600,21600
Handles	position="#0,topLeft" xrange="0,21600"



Right Triangle
Internal Name: RightTriangle
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

Path	m,l,21600r21600,xe
Connector Locations	0,0;0,10800;0,21600;10800,21600;21600,21600;10800,10800
Text Box Rectangle	1800,12600,12600,19800



Oval
Internal Name: Oval
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

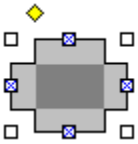
The oval is a geometric primitive with default values.

Hexagon

Internal Name: Hexagon
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top perimeter adjust handle controls the length of the pointed ends on both sides. The adjust handle can extend halfway across.

Geometric properties:

Path	m@0,l,10800@0,21600@1,21600,21600,10800@1,xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod @0 2929 10000 sum width 0 @3 sum height 0 @3
Adjustment Values	5400
Connector Locations	Rectangle
Text Box Rectangle	1800,1800,19800,19800;3600,3600,18000,18000;6300,6300,15300,15300
Handles	position="#0,topLeft" xrange="0,10800"

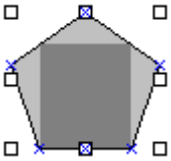


Cross

Internal Name: Plus
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount cut out of the corners. It limo-stretches both vertically and horizontally at the midpoint, so that the amount cut out of the corners is always square. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension.

Geometric properties:

Path	m@0,l@0@0,0@0,0@2@0@2@0,21600@1,21600@1@2,21600@2,21600 @0@1@0@1,xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod @0 2929 10000 sum width 0 @3 sum height 0 @3 val width val height prod width 1 2 prod height 1 2
Adjustment Values	5400
Connector Locations	@8,0;0,@9;@8,@7;@6,@9
Text Box Rectangle	0,0,21600,21600;5400,5400,16200,16200;10800,10800,10800,10800
Handles	position="#0,topLeft" switch="true" xrange="0,10800"
Limo	10800,10800



Regular Pentagon

Internal Name: Pentagon
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

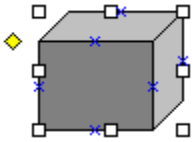
Path	m10800,l,8259,4200,21600r13200,l21600,8259xe
Connector Locations	10800,0;0,8259;4200,21600;10800,21600;17400,21600;21600,8259
Connector Angles	270,180,90,90,90,0
Text Box Rectangle	4200,5077,17400,21600

Can

Internal Name: Can
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: One vertical adjust handle along the center adjusts the tilt of the can. The adjust handle can extend halfway down.

Geometric properties:

Path	m10800,qx0@1l0@2qy10800,21600,21600@2l21600@1qy10800,xem0@1qy 10800@0,21600@1nfe
Guide Formulas	val #0 prod #0 1 2 sum height 0 @1
Adjustment Values	5400
Connector Locations	10800,@0;10800,0;0,10800;10800,21600;21600,10800
Connector Angles	270,270,180,90,0
Text Box Rectangle	0,@0,21600,@2
Handles	position="center,#0" yrange="0,10800"

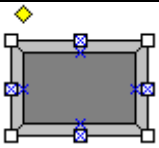


Cube

Internal Name: Cube
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of extrusion. It limo-stretches both vertically and horizontally at the midpoint, so that the extrusion is always at 45°. The adjust handle switches between the top and left sides depending on which dimension is smaller.

Geometric properties:

Path	m@0,l0@0,,21600@1,21600,21600@2,21600,xem0@0nfl@1@0,21600,em@ 1@0nfl@1,21600e
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 mid height #0 prod @1 1 2 prod @2 1 2 mid width #0
Adjustment Values	5400
Connector Locations	@6,0;@4,@0;0,@3;@4,21600;@1,@3;21600,@5
Connector Angles	270,270,180,90,0,0
Text Box Rectangle	0,@0,@1,21600
Handles	position="topLeft,#0" switch="true" yrange="0,21600"
Limo	10800,10800

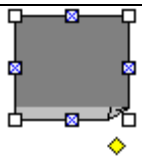


Bevel

Internal Name: Bevel
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension.

Geometric properties:

Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0e
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0
Adjustment Values	2700
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,10800"
Limo	10800,10800

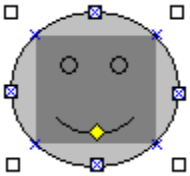


Folded Corner

Internal Name: FoldedCorner
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A bottom perimeter adjust handle controls the amount that the corner is folded.

Geometric properties:

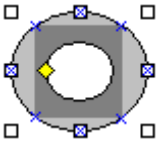
Path	m,l,21600@0,21600,21600@0,21600,xem@0,21600nfl@3@5c@7@9@11@ 13,21600@0e
Guide Formulas	val #0 sum 21600 0 @0 prod @1 8481 32768 sum @2 @0 0 prod @1 1117 32768 sum @4 @0 0 prod @1 11764 32768 sum @6 @0 0 prod @1 6144 32768 sum @8 @0 0 prod @1 20480 32768 sum @10 @0 0 prod @1 6144 32768 sum @12 @0 0
Adjustment Values	18900
Connector Locations	Rectangle
Text Box Rectangle	0,0,21600,@13
Handles	position="#0,bottomRight" xrange="10800,21600"



Smiley Face
Internal Name: SmileyFace
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: A vertical adjust handle along the center adjust the curvature of the mouth changing it from a smile to a frown.

Geometric properties:

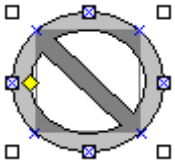
Path	m10800,qx,10800,10800,21600,21600,10800,10800,xem7340,6445qx6215,7570,7340,8695,8465,7570,7340,6445xnfem14260,6445qx13135,7570,14260,8695,15385,7570,14260,6445xnfem4960@0c8853@3,12747@3,16640@0nfe
Guide Formulas	sum 33030 0 #0 prod #0 4 3 prod @0 1 3 sum @1 0 @2
Adjustment Values	17520
Connector Locations	10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437,18437;21600,10800;18437,3163
Text Box Rectangle	3163,3163,18437,18437
Handles	position="center,#0" yrange="15510,17520



Donut
Internal Name: Donut
Shaped Concentric Fill: No.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: One horizontal adjust handle along the center adjusts the inner radius of the ring. The adjust handle can extend halfway across.

Geometric properties:

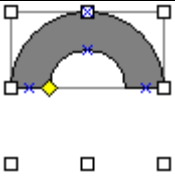
Path	m,10800qy10800,,21600,10800,10800,21600,,10800xm@0,10800qy10800@2@1,10800,10800@0@0,10800xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod @0 2929 10000 sum width 0 @3 sum height 0 @3
Adjustment Values	5400
Connector Locations	10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437,18437;21600,10800;18437,3163
Text Box Rectangle	3163,3163,18437,18437
Handles	position="#0,center" xrange="0,10800"



“No” Symbol
Internal Name: NoSmoking
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One horizontal adjust handle along the center adjusts the inner radius of the ring and thickness of the slash. The adjust handle can extend 1/3 of the way across.

Geometric properties:

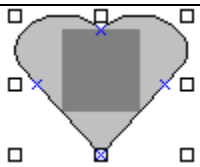
Path	m,10800qy10800,,21600,10800,10800,21600,,10800xar@0@0@16@16@12 @14@15@13xar@0@0@16@16@13@15@14@12xe
Guide Formulas	val #0 prod @0 2 1 sum 21600 0 @1 prod @2 @2 1 prod @0 @0 1 sum @3 0 @4 prod @5 1 8 sqrt @6 prod @4 1 8 sqrt @8 sum @7 @9 0 sum @7 0 @9 sum @10 10800 0 sum 10800 0 @10 sum @11 10800 0 sum 10800 0 @11 sum 21600 0 @0
Adjustment Values	2700
Connector Locations	10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437,18437;21600,10800;18437,3163
Text Box Rectangle	3163,3163,18437,18437
Handles	position="#0,center" xrange="0,7200"



Block Arc
Internal Name: BlockArc
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A polar adjust handle controls the starting angle and inner radius. The shape goes clockwise starting from the starting angle and is symmetrical about the vertical center.

Geometric properties:

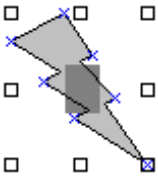
Path	al10800,10800@0@0@2@14,10800,10800,10800,10800@3@15xe
Guide Formulas	val #1 val #0 sum 0 0 #0 sumangle #0 0 180 sumangle #0 0 90 prod @4 2 1 sumangle #0 90 0 prod @6 2 1 abs #0 sumangle @8 0 90 if @9 @7 @5 sumangle @10 0 360 if @10 @11 @10 sumangle @12 0 360 if @12 @13 @12 sum 0 0 @14 val 10800 sum 10800 0 #1 prod #1 1 2 sum @18 5400 0 cos @19 #0 sin @19 #0 sum @20 10800 0 sum @21 10800 0 sum 10800 0 @20 sum #1 10800 0 if @9 @17 @25 if @9 0 21600 cos 10800 #0 sin 10800 #0 sin #1 #0 sum @28 10800 0 sum @29 10800 0 sum @30 10800 0 if @4 0 @31 if #0 @34 0 if @6 @35 @31 sum 21600 0 @36 if @4 0 @33 if #0 @38 @32 if @6 @39 0 if @4 @32 21600 if @6 @41 @33
Adjustment Values	11796480,5400
Connector Locations	10800,@27;@22,@23;10800,@26;@24,@23
Text Box Rectangle	@36,@40,@37,@42
Handles	position="#1,#0" polar="10800,10800" radiusrange="0,10800"



Heart
Internal Name: Heart
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

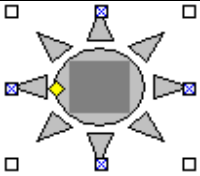
Path	m10860,2187c10451,1746,9529,1018,9015,730,7865,152,6685,,5415,,4175,152,2995,575,1967,1305,1150,2187,575,3222,242,4220,,5410,242,6560,575,7597l10860,21600,20995,7597v485,-1037,605,-2187,485,-3377c21115,3222,20420,2187,19632,1305,18575,575,17425,152,16275,,15005,,13735,152,12705,730v-529,288,-1451,1016,-1845,1457xe
Connector Locations	10860,2187;2928,10800;10860,21600;18672,10800
Connector Angles	270,180,90,0
Text Box Rectangle	5037,2277,16557,13677



Lightning Bolt
Internal Name: LightningBolt
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None. Note that the text rectangle extends past the interior of the shape.

Geometric properties:

Path	m8472,l,3890,7602,8382,5022,9705r7200,4192l10012,14915r11588,6685l14767,12877r1810,-870l11050,6797r1810,-717xe
Connector Locations	8472,0;0,3890;5022,9705;10012,14915;21600,21600;16577,12007;12860,6080
Connector Angles	270,270,180,180,90,0,0
Text Box Rectangle	8757,7437,13917,14277

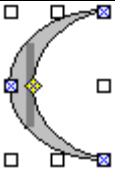


Sun

Internal Name: Sun
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A horizontal adjust handle along the center controls the radius of the core part of the sun.

Geometric properties:

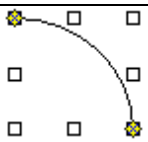
Path	m21600,10800l@15@14@15@18xem18436,3163l@17@12@16@13xem10800,l@14@10@18@10xem3163,3163l@12@13@13@12xem,10800l@10@18@10@14xem3163,18436l@13@16@12@17xem10800,21600l@18@15@14@15xem18436,18436l@16@17@17@16xem10800@19qx@19,10800,10800@20@20,10800,10800@19xe
Guide Formulas	sum 10800 0 #0 prod @0 30274 32768 prod @0 12540 32768 sum @1 10800 0 sum @2 10800 0 sum 10800 0 @1 sum 10800 0 @2 prod @0 23170 32768 sum @7 10800 0 sum 10800 0 @7 prod @5 3 4 prod @6 3 4 sum @10 791 0 sum @11 791 0 sum @11 2700 0 sum 21600 0 @10 sum 21600 0 @12 sum 21600 0 @13 sum 21600 0 @14 val #0 sum 21600 0 #0
Adjustment Values	5400
Text Box Rectangle	@9,@9,@8,@8
Handles	position="#0,center" xrange="2700,10125"



Moon
Internal Name: Moon
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A horizontal adjust handle along the center adjusts the thickness of the crescent.

Geometric properties:

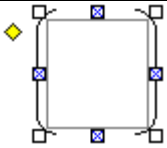
Path	m21600,qx,10800,21600,21600wa@0@10@6@11,21600,21600,21600,xe
Guide Formulas	val #0 sum 21600 0 #0 prod #0 #0 @1 prod 21600 21600 @1 prod @3 2 1 sum @4 0 @2 sum @5 0 #0 prod @5 1 2 sum @7 0 #0 prod @8 1 2 sum 10800 0 @9 sum @9 10800 0 prod #0 9598 32768 sum 21600 0 @12 ellipse @13 21600 10800 sum 10800 0 @14 sum @14 10800 0
Adjustment Values	10800
Connector Locations	21600,0;0,10800;21600,21600;@0,10800
Connector Angles	270,180,90,0
Text Box Rectangle	@12,@15,@0,@16
Handles	position="#0,center" xrange="0,18900



Arc
Internal Name: Arc
Shaped Concentric Fill: Yes (elliptical fill).
Joins: Rounded.
Endcaps: Rounded.
Adjustments: The first adjust handle controls the start angle and the second adjust handle controls the end angle.

Geometric properties:

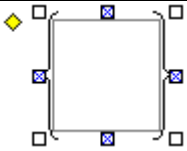
Path	wr-21600,,21600,43200,,,21600,21600nfewr-21600,,21600,43200,,,21600,21600l,21600nsxe
Guide Formulas	val #2 val #3 val #4
Adjustment Values	-5898240,,,21600,21600
Connector Locations	0,0;21600,21600;0,21600
Handles	position="@2,#0" polar="@0,@1 position="@2,#1" polar="@0,@1



Double Bracket
Internal Name: BracketPair
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Flat.
Adjustments: A top or left perimeter adjust handle controls the roundedness of the corners. It limo-stretches both vertically and horizontally at the midpoint, so that the rounded corners are always circular. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend halfway across.

Geometric properties:

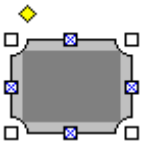
Path	m@0,nfqx0@0l0@2qy@0,21600em@1,nfqx21600@0l21600@2qy@1,21600em@0,nsqx0@0l0@2qy@0,21600l@1,21600qx21600@2l21600@0qy@1,xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod @0 2929 10000 sum width 0 @3 sum height 0 @3 val width val height prod width 1 2 prod height 1 2
Adjustment Values	3600
Connector Locations	@8,0;0,@9;@8,@7;@6,@9
Text Box Rectangle	@3,@3,@4,@5
Handles	position="#0,topLeft" switch="true" xrange="0,10800"
Limo	10800,10800



Double Brace
Internal Name: BracePair
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Flat.
Adjustments: A top or left perimeter adjust handle controls the roundedness of the corners. It limo-stretches both vertically and horizontally at the midpoint, so that the rounded corners are always circular. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend halfway across.

Geometric properties:

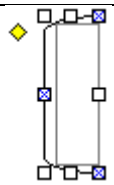
Path	m@9,nfqx@0@0l@0@7qy0@4@0@8l@0@6qy@9,21600em@10,nfqx@5@0l@5@7qy21600@4@5@8l@5@6qy@10,21600em@9,nsqx@0@0l@0@7qy0@4@0@8l@0@6qy@9,21600l@10,21600qx@5@6l@5@8qy21600@4@5@7l@5@0qy@10,xe
Guide Formulas	val #0 val width val height prod width 1 2 prod height 1 2 sum width 0 #0 sum height 0 #0 sum @4 0 #0 sum @4 #0 0 prod #0 2 1 sum width 0 @9 prod #0 9598 32768 sum height 0 @11 sum @11 #0 0 sum width 0 @13
Adjustment Values	1800
Connector Locations	@3,0;0,@4;@3,@2;@1,@4
Text Box Rectangle	@13,@11,@14,@12
Handles	position="topLeft,#0" switch="true" yrange="0,5400"
Limo	10800,10800



Plaque
Internal Name: Plaque
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the roundedness of the corners. It limo-stretches both vertically and horizontally at the midpoint, so that the rounded corners are always circular. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension.

Geometric properties:

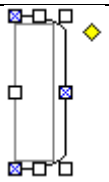
Path	m@0,qy0@0l0@2qx@0,21600l@1,21600qy21600@2l21600@0qx@1,xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod @0 7071 10000 sum width 0 @3 sum height 0 @3 val width val height prod width 1 2 prod height 1 2
Adjustment Values	3600
Connector Locations	@8,0;0,@9;@8,@7;@6,@9
Text Box Rectangle	@3,@3,@4,@5
Handles	position="#0,topLeft" switch="true" xrange="0,10800"
Limo	10800,10800



Left Bracket
Internal Name: LeftBracket
Shaped Concentric Fill: No.
Joins: Rounded.
Endcaps: Flat.
Adjustments: A left perimeter adjust handle controls the curvature of the corners. The adjust handle can extend halfway down.

Geometric properties:

Path	m21600,qx0@0l0@1qy21600,21600e
Guide Formulas	val #0 sum 21600 0 #0 prod #0 9598 32768 sum 21600 0 @2
Adjustment Values	1800
Connector Locations	21600,0;0,10800;21600,21600
Text Box Rectangle	6326,@2,21600,@3
Handles	position="topLeft,#0" yrange="0,10800"



Right Bracket
Internal Name: RightBracket
Shaped Concentric Fill: No.
Joins: Rounded.
Endcaps: Flat.
Adjustments: A right perimeter adjust handle controls the curvature of the corners. The adjust handle can extend halfway down.

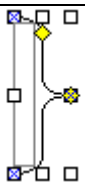
Geometric properties:

Path	m,qx21600@0l21600@1qy,21600e
Guide Formulas	val #0 sum 21600 0 #0 prod #0 9598 32768 sum 21600 0 @2
Adjustment Values	1800
Connector Locations	0,0;0,21600;21600,10800
Text Box Rectangle	0,@2,15274,@3
Handles	position="bottomRight,#0" yrange="0,10800"

Left Brace
Internal Name: LeftBrace
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Flat.
Adjustments: A vertical adjust handle along the center controls the curvature of the corners and cusp. The adjust handle can extend ¼ of the way down. A left perimeter adjust handle controls the position of the cusp.

Geometric properties:

Path	m21600,qx10800@0l10800@2qy0@11,10800@3l10800@1qy21600,21600e
Guide Formulas	val #0 sum 21600 0 #0 sum #1 0 #0 sum #1 #0 0 prod #0 9598 32768 sum 21600 0 @4 sum 21600 0 #1 min #1 @6 prod @7 1 2 prod #0 2 1 sum 21600 0 @9 val #1
Adjustment Values	1800,10800
Connector Locations	21600,0;0,10800;21600,21600
Text Box Rectangle	13963,@4,21600,@5
Handles	position="center,#0" yrange="0,@8 position="topLeft,#1" yrange="@9,@10

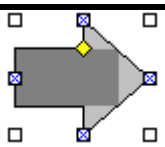


Right Brace
Internal Name: RightBrace
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A vertical adjust handle along the center controls the curvature of the corners and cusp. The adjust handle can extend ¼ of the way down. A right perimeter adjust handle controls the position of the cusp.

Geometric properties:

Path	m,qx10800@0l10800@2qy21600@11,10800@3l10800@1qy,21600e
Guide Formulas	val #0 sum 21600 0 #0 sum #1 0 #0 sum #1 #0 0 prod #0 9598 32768 sum 21600 0 @4 sum 21600 0 #1 min #1 @6 prod @7 1 2 prod #0 2 1 sum 21600 0 @9 val #1
Adjustment Values	1800,10800
Connector Locations	0,0;21600,@11;0,21600
Text Box Rectangle	0,@4,7637,@5
Handles	position="center,#0" yrange="0,@8 position="bottomRight,#1" yrange="@9,@10

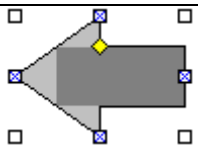
Block Arrows



Right Arrow
Internal Name: RightArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Flat.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. The adjust handle can extend all of the way across and halfway down.

Geometric properties:

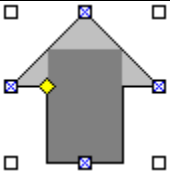
Path	m@0,l@0@1,0@1,0@2@0@2@0,21600,21600,10800xe
Guide Formulas	val #0 val #1 sum height 0 #1 sum 10800 0 #1 sum width 0 #0 prod @4 @3 10800 sum width 0 @5
Adjustment Values	16200,5400
Connector Locations	@0,0;0,10800;@0,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	0,@1,@6,@2
Handles	position="#0,#1" xrange="0,21600" yrange="0,10800"



Left Arrow
Internal Name: LeftArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. The adjust handle can extend all of the way across and halfway down.

Geometric properties:

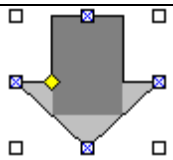
Path	m@0,l@0@1,21600@1,21600@2@0@2@0,21600,,10800xe
Guide Formulas	val #0 val #1 sum 21600 0 #1 prod #0 #1 10800 sum #0 0 @3
Adjustment Values	5400,5400
Connector Locations	@0,0;0,10800;@0,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	@4,@1,21600,@2
Handles	position="#0,#1" xrange="0,21600" yrange="0,10800"



Up Arrow
Internal Name: UpArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. The adjust handle can extend halfway across and all of the way down.

Geometric properties:

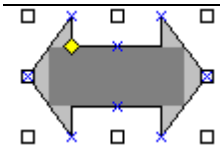
Path	m0@0l@1@0@1,21600@2,21600@2@0,21600@0,10800,xe
Guide Formulas	val #0 val #1 sum 21600 0 #1 prod #0 #1 10800 sum #0 0 @3
Adjustment Values	5400,5400
Connector Locations	10800,0;0,@0;10800,21600;21600,@0
Connector Angles	270,180,90,0
Text Box Rectangle	@1,@4,@2,21600
Handles	position="#1,#0" xrange="0,10800" yrange="0,21600"



Down Arrow
Internal Name: DownArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. The adjust handle can extend halfway across and all of the way down.

Geometric properties:

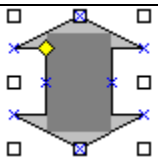
Path	m0@0l@1@0@1,0@2,0@2@0,21600@0,10800,21600xe
Guide Formulas	val #0 val #1 sum height 0 #1 sum 10800 0 #1 sum width 0 #0 prod @4 @3 10800 sum width 0 @5
Adjustment Values	16200,5400
Connector Locations	10800,0;0,@0;10800,21600;21600,@0
Connector Angles	270,180,90,0
Text Box Rectangle	@1,0,@2,@6
Handles	position="#1,#0" xrange="0,10800" yrange="0,21600"



Left-Right Arrow
Internal Name: LeftRightArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the arrowhead length is the same at both ends. The adjust handle can extend halfway across and halfway down.

Geometric properties:

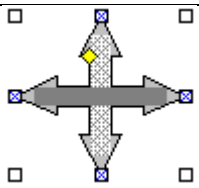
Path	m,10800l@0,21600@0@3@2@3@2,21600,21600,10800@2,0@2@1@0@1@0,xe
Guide Formulas	val #0 val #1 sum 21600 0 #0 sum 21600 0 #1 prod #0 #1 10800 sum #0 0 @4 sum 21600 0 @5
Adjustment Values	4320,5400
Connector Locations	@2,0;10800,@1;@0,0;0,10800;@0,21600;10800,@3;@2,21600;21600,10800
Connector Angles	270,270,270,180,90,90,90,0
Text Box Rectangle	@5,@1,@6,@3
Handles	position="#0,#1" xrange="0,10800" yrange="0,10800"



Up-Down Arrow
Internal Name: UpDownArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the arrowhead length is the same at both ends. The adjust handle can extend halfway across and halfway down.

Geometric properties:

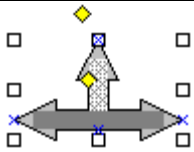
Path	m10800,l21600@0@3@0@3@2,21600@2,10800,21600,0@2@1@2@1@0,0 @0xe
Guide Formulas	val #1 val #0 sum 21600 0 #1 sum 21600 0 #0 prod #1 #0 10800 sum #1 0 @4 sum 21600 0 @5
Adjustment Values	5400,4320
Connector Locations	10800,0;0,@0;@1,10800;0,@2;10800,21600;21600,@2;@3,10800;21600,@0
Connector Angles	270,180,180,180,90,0,0,0
Text Box Rectangle	@1,@5,@3,@6
Handles	position="#0,#1" xrange="0,10800" yrange="0,10800"



Quad Arrow
Internal Name: QuadArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the width of the arrowhead. The second adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the arrowhead length is the same for all four arrowheads when the shape is scaled to a 1:1 aspect ratio. The first adjust handle can extend as far left as possible before the tips of the arrowheads cross, and as far right as the current position of the second adjust handle. The second adjust handle can extend as far left as the current position of the first adjust handle, as far right as halfway across, and as far down as possible before the arrowheads intersect the trunks. A second text rect is defined for vertical text.

Geometric properties:

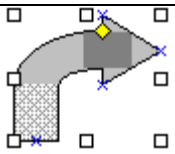
Path	m10800,l@0@2@1@2@1@1@2@1@2@0,,10800@2@3@2@4@1@4@1 @5@0@5,10800,21600@3@5@4@5@4@4@5@4@5@3,21600,10800@5 @0@5@1@4@1@4@2@3@2xe
Guide Formulas	val #0 val #1 val #2 sum 21600 0 #0 sum 21600 0 #1 sum 21600 0 #2 sum #0 0 10800 sum #1 0 10800 prod @7 #2 @6 sum 21600 0 @8
Adjustment Values	6480,8640,4320
Connector Locations	Rectangle
Text Box Rectangle	@8,@1,@9,@4;@1,@8,@4,@9
Handles	position="#0,topLeft" xrange="@2,@1 position="#1,#2" xrange="@0,10800" yrange="0,@0



Left-Right-Up Arrow
Internal Name: LeftRightUpArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the width of the arrowhead. The second adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the arrowhead length is the same for all three arrowheads when the shape is scaled such that the horizontal part of the trunk is the same width as the vertical part of the trunk. The first adjust handle can extend as far left as possible before the tips of the arrowheads cross, and as far right as the current position of the second adjust handle. The second adjust handle can extend as far left as the current position of the first adjust handle, as far right as halfway across, and as far down as possible before the arrowheads intersect the trunks. A second text rect is defined for vertical text.

Geometric properties:

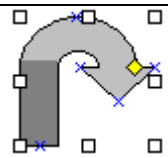
Path	m10800,l@0@2@1@2@1@6@7@6@7@5,0@8@7,21600@7@9@10@9@10,21600,21600@8@10@5@10@6@4@6@4@2@3@2xe
Guide Formulas	val #0 val #1 val #2 sum 21600 0 #0 sum 21600 0 #1 prod @0 21600 @3 prod @1 21600 @3 prod @2 @3 21600 prod 10800 21600 @3 prod @4 21600 @3 sum 21600 0 @7 sum @5 0 @8 sum @6 0 @8 prod @12 @7 @11 sum 21600 0 @13 sum @0 0 10800 sum @1 0 10800 prod @2 @16 @15
Adjustment Values	6480,8640,6171
Connector Locations	10800,0;0,@8;10800,@9;21600,@8
Connector Angles	270,180,90,0
Text Box Rectangle	@13,@6,@14,@9;@1,@17,@4,@9
Handles	position="#0,topLeft" xrange="@2,@1 position="#1,#2" xrange="@0,10800" yrange="0,@5



Bent Arrow
Internal Name: BentArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. The adjust handle can extend as far left as where the trunk begins to bend and halfway down the arrowhead. A second text rect is defined for vertical text.

Geometric properties:

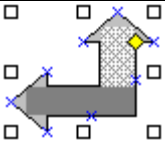
Path	m21600,6079l@0,0@0@1,12427@1qx,12158l,21600@4,21600@4,12158qy12427@2l@0@2@0,12158xe
Guide Formulas	val #0 val #1 sum 12158 0 #1 sum @2 0 #1 prod @3 32768 32059 prod @4 1 2 sum 21600 0 #0 prod @6 #1 6079 sum @7 #0 0
Adjustment Values	Connector Angles
Connector Locations	@0,0;@0,12158;@5,21600;21600,6079
Connector Angles	270,90,90,0
Text Box Rectangle	12427,@1,@8,@2;0,12158,@4,21600
Handles	position="#0,#1" xrange="12427,21600" yrange="0,6079"



U-Turn Arrow
Internal Name: UturnArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. The adjust handle can extend halfway across the arrowhead, as far up as where the trunk begins the bend, and as far down as the tip of the arrowhead.

Geometric properties:

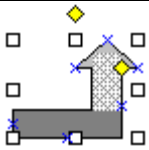
Path	m15662,14285l21600,8310r-2970,qy9250,,,8485l,21600r6110,l6110,8310qy8907,5842l9725,5842qx12520,8310l9725,8310xe
Connector Locations	9250,0;3055,21600;9725,8310;15662,14285;21600,8310
Connector Angles	270,90,90,90,0
Text Box Rectangle	0,8310,6110,21600



Left-Up Arrow
Internal Name: LeftUpArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the width of the arrowhead. The second adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the arrowhead length is the same for both arrowheads when the shape is scaled to a 1:1 aspect ratio. The first adjust handle can extend as far left as possible before the tips of the arrowheads cross, and as far right as twice the distance from the right perimeter as the current position of the second adjust handle. The second adjust handle can extend as far left as half the distance from the right perimeter as the current position of the first adjust handle, and as far down as possible before the arrowheads intersect the trunks. A second text rect is defined for vertical text.

Geometric properties:

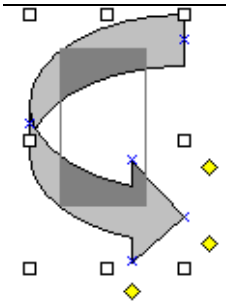
Path	m@4,l@0@2@5@2@5@5@2@5@2@0,0@4@2,21600@2@1@1@1@1@2,21600@2xe
Guide Formulas	val #0 val #1 val #2 prod #0 1 2 sum @3 10800 0 sum 21600 #0 #1 sum #1 #2 0 prod @6 1 2 prod #1 2 1 sum @8 0 21600 sum @5 0 @4 sum #0 0 @4 prod @2 @10 @11
Adjustment Values	9257,18514,6171
Connector Locations	@4,0;@0,@2;@2,@0;0,@4;@2,21600;@7,@1;@1,@7;21600,@2
Connector Angles	270,180,270,180,90,90,0,0
Text Box Rectangle	@12,@5,@1,@1;@5,@12,@1,@1
Handles	position="#0,topLeft" xrange="@2,@9 position="#1,#2" xrange="@4,21600" yrange="0,@0



Bent-Up Arrow
Internal Name: BentUpArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the width of the arrowhead. The second adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the lengths of the horizontal and vertical trunks are the same when scaled so that widths of the horizontal and vertical trunks are the same. The first adjust handle can extend as far left as possible before the arrowhead intersects the trunk, and as far right as twice the distance from the right perimeter as the current position of the second adjust handle. The second adjust handle can extend as far left as half the distance from the right perimeter as the current position of the first adjust handle, and as far down as possible before the arrowhead intersect the trunk. A second text rect is defined for vertical text.

Geometric properties:

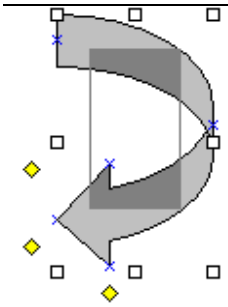
Path	m@4,l@0@2@5@2@5@12,0@12,,21600@1,21600@1@2,21600@2xe
Guide Formulas	val #0 val #1 val #2 prod #0 1 2 sum @3 10800 0 sum 21600 #0 #1 sum #1 #2 0 prod @6 1 2 prod #1 2 1 sum @8 0 21600 prod 21600 @0 @1 prod 21600 @4 @1 prod 21600 @5 @1 prod 21600 @7 @1 prod #1 1 2 sum @5 0 @4 sum @0 0 @4 prod @2 @15 @16
Adjustment Values	9257,18514,7200
Connector Locations	@4,0;@0,@2;0,@11;@14,21600;@1,@13;21600,@2
Connector Angles	270,180,180,90,0,0
Text Box Rectangle	0,@12,@1,21600;@5,@17,@1,21600
Handles	position="#0,topLeft" xrange="@2,@9 position="#1,#2" xrange="@4,21600" yrange="0,@0



Curved Right Arrow
Internal Name: CurvedRightArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a perimeter adjust handle that controls the curvature of the arrow. The second adjust handle controls both the length of the arrowhead, and the width of the trunk.

Geometric properties:

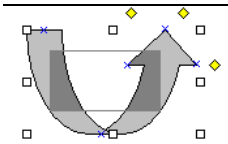
Path	r,0@23@3@22,,0@4,0@15@23@1,0@7@2@13l@2@14@22@8@2@12w a,0@23@3@2@11@26@17,0@15@23@1@26@17@22@15xear,0@23@3, 0@4@26@17nfe
Guide Formulas	val #0 val #1 val #2 sum #0 width #1 prod @3 1 2 sum #1 #1 width sum @5 #1 #0 prod @6 1 2 mid width #0 sum height 0 #2 ellipse @9 height @4 sum @4 @10 0 sum @11 #1 width sum @7 @10 0 sum @12 width #0 sum @5 0 #0 prod @15 1 2 mid @4 @7 sum #0 #1 width prod @18 1 2 sum @17 0 @19 val width val height prod height 2 1 sum @17 0 @4 ellipse @24 @4 height sum height 0 @25 sum @8 128 0 prod @5 1 2 sum @5 0 128 sum #0 @17 @12 ellipse @20 @4 height sum width 0 #0 prod @32 1 2 prod height height 1 prod @9 @9 1 sum @34 0 @35 sqrt @36 sum @37 height 0 prod width height @38 sum @39 64 0 prod #0 1 2 ellipse @33 @41 height sum height 0 @42 sum @43 64 0 prod @4 1 2 sum #1 0 @45 prod height 4390 32768 prod height 28378 32768
Adjustment Values	12960,19440,14400
Connector Locations	0,@17;@2,@14;@22,@8;@2,@12;@22,@16
Connector Angles	180,90,0,0,0
Text Box Rectangle	@47,@45,@48,@46
Handles	position="bottomRight,#0" yrange="@40,@29 position="bottomRight,#1" yrange="@27,@21 position="#2,bottomRight" xrange="@44,@22



Curved Left Arrow
Internal Name: CurvedLeftArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a perimeter adjust handle that controls the curvature of the arrow. The second adjust handle controls both the length of the arrowhead, and the width of the trunk.

Geometric properties:

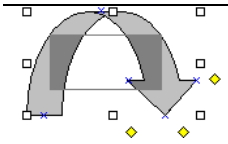
Path	wr@22,0@21@3,,0@21@4@22@14@21@1@21@7@2@12l@2@13,0@8 @2@11at@22,0@21@3@2@10@24@16@22@14@21@1@24@16,0@14x ear@22@14@21@1@21@7@24@16nfe
Guide Formulas	val #0 val #1 val #2 sum #0 width #1 prod @3 1 2 sum #1 #1 width sum @5 #1 #0 prod @6 1 2 mid width #0 ellipse #2 height @4 sum @4 @9 0 sum @10 #1 width sum @7 @9 0 sum @11 width #0 sum @5 0 #0 prod @14 1 2 mid @4 @7 sum #0 #1 width prod @17 1 2 sum @16 0 @18 val width val height sum 0 0 height sum @16 0 @4 ellipse @23 @4 height sum @8 128 0 prod @5 1 2 sum @5 0 128 sum #0 @16 @11 sum width 0 #0 prod @29 1 2 prod height height 1 prod #2 #2 1 sum @31 0 @32 sqrt @33 sum @34 height 0 prod width height @35 sum @36 64 0 prod #0 1 2 ellipse @30 @38 height sum @39 0 64 prod @4 1 2 sum #1 0 @41 prod height 4390 32768 prod height 28378 32768
Adjustment Values	12960,19440,7200
Connector Locations	0,@15;@2,@11;0,@8;@2,@13;@21,@16
Connector Angles	180,180,180,90,0
Text Box Rectangle	@43,@41,@44,@42
Handles	position="topLeft,#0" yrange="@37,@27 position="topLeft,#1" yrange="@25,@20 position="#2,bottomRight" xrange="0,@40



Curved Up Arrow
Internal Name: CurvedUpArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a perimeter adjust handle that controls the curvature of the arrow. The second adjust handle controls both the length of the arrowhead, and the width of the trunk.

Geometric properties:

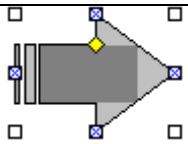
Path	ar0@22@3@21,,0@4@21@14@22@1@21@7@21@12@2l@13@2@8,0@11@2wa0@22@3@21@10@2@16@24@14@22@1@21@16@24@14,xewr@14@22@1@21@7@21@16@24nfe
Guide Formulas	val #0 val #1 val #2 sum #0 width #1 prod @3 1 2 sum #1 #1 width sum @5 #1 #0 prod @6 1 2 mid width #0 ellipse #2 height @4 sum @4 @9 0 sum @10 #1 width sum @7 @9 0 sum @11 width #0 sum @5 0 #0 prod @14 1 2 mid @4 @7 sum #0 #1 width prod @17 1 2 sum @16 0 @18 val width val height sum 0 0 height sum @16 0 @4 ellipse @23 @4 height sum @8 128 0 prod @5 1 2 sum @5 0 128 sum #0 @16 @11 sum width 0 #0 prod @29 1 2 prod height height 1 prod #2 #2 1 sum @31 0 @32 sqrt @33 sum @34 height 0 prod width height @35 sum @36 64 0 prod #0 1 2 ellipse @30 @38 height sum @39 0 64 prod @4 1 2 sum #1 0 @41 prod height 4390 32768 prod height 28378 32768
Adjustment Values	12960,19440,7200
Connector Locations	@8,0;@11,@2;@15,0;@16,@21;@13,@2
Connector Angles	270,270,270,90,0
Text Box Rectangle	"@41,@43,@42,@44
Handles	position="#0,topLeft" xrange="@37,@27 position="#1,topLeft" xrange="@25,@20 position="bottomRight,#2" yrange="0,@40



Curved Down Arrow
Internal Name: CurvedDownArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a perimeter adjust handle that controls the curvature of the arrow. The second adjust handle controls both the length of the arrowhead, and the width of the trunk.

Geometric properties:

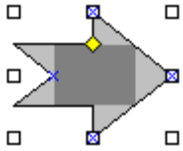
Path	wr,0@3@23,0@22@4,0@15,0@1@23@7,0@13@2l@14@2@8@22@12@2at,0@3@23@11@2@17@26@15,0@1@23@17@26@15@22xewr,0@3@23@4,0@17@26nfe
Guide Formulas	val #0 val #1 val #2 sum #0 width #1 prod @3 1 2 sum #1 #1 width sum @5 #1 #0 prod @6 1 2 mid width #0 sum height 0 #2 ellipse @9 height @4 sum @4 @10 0 sum @11 #1 width sum @7 @10 0 sum @12 width #0 sum @5 0 #0 prod @15 1 2 mid @4 @7 sum #0 #1 width prod @18 1 2 sum @17 0 @19 val width val height prod height 2 1 sum @17 0 @4 ellipse @24 @4 height sum height 0 @25 sum @8 128 0 prod @5 1 2 sum @5 0 128 sum #0 @17 @12 ellipse @20 @4 height sum width 0 #0 prod @32 1 2 prod height height 1 prod @9 @9 1 sum @34 0 @35 sqrt @36 sum @37 height 0 prod width height @38 sum @39 64 0 prod #0 1 2 ellipse @33 @41 height sum height 0 @42 sum @43 64 0 prod @4 1 2 sum #1 0 @45 prod height 4390 32768 prod height 28378 32768
Adjustment Values	12960,19440,14400
Connector Locations	@17,0;@16,@22;@12,@2;@8,@22;@14,@2
Connector Angles	270,90,90,90,0
Text Box Rectangle	@45,@47,@46,@48
Handles	position="#0,bottomRight" xrange="@40,@29 position="#1,bottomRight" xrange="@27,@21 position="bottomRight,#2" yrange="@44,@22



Striped Right Arrow
Internal Name: StripedRightArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. The adjust handle can extend as far left as where the stripes are and halfway down.

Geometric properties:

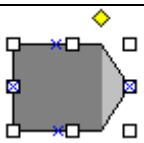
Path	m@0,l@0@1,3375@1,3375@2@0@2@0,21600,21600,10800xem1350@1l1 350@2,2700@2,2700@1xem0@1l0@2,675@2,675@1xe
Guide Formulas	val #0 val #1 sum height 0 #1 sum 10800 0 #1 sum width 0 #0 prod @4 @3 10800 sum width 0 @5
Adjustment Values	16200,5400
Connector Locations	@0,0;0,10800;@0,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	3375,@1,@6,@2
Handles	position="#0,#1" xrange="3375,21600" yrange="0,10800"



Notched Right Arrow
Internal Name: NotchedRightArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls both the length of the arrowhead, and the width of the trunk. The notch at the end stays adjusted so that it matches the shape of the arrowhead. The adjust handle can extend all of the way across and halfway down.

Geometric properties:

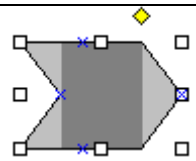
Path	m@0,l@0@1,0@1@5,10800,0@2@0@2@0,21600,21600,10800xe
Guide Formulas	val #0 val #1 sum height 0 #1 sum 10800 0 #1 sum width 0 #0 prod @4 @3 10800 sum width 0 @5
Adjustment Values	16200,5400
Connector Locations	@0,0;@5,10800;@0,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	@5,@1,@6,@2
Handles	position="#0,#1" xrange="0,21600" yrange="0,10800"



Pentagon
Internal Name: HomePlate
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top perimeter adjust handle controls the length of the pointed end.

Geometric properties:

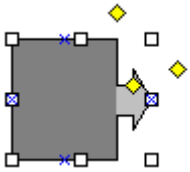
Path	m@0,l,,,21600@0,21600,21600,10800xe
Guide Formulas	val #0 prod #0 1 2
Adjustment Values	16200
Connector Locations	@1,0;0,10800;@1,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	0,0,10800,21600;0,0,16200,21600;0,0,21600,21600
Handles	position="#0,topLeft" xrange="0,21600"



Chevron
Internal Name: Chevron
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top perimeter adjust handle controls the length of the pointed end.

Geometric properties:

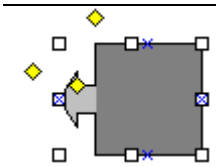
Path	m@0,l,0@1,10800,,21600@0,21600,21600,10800xe
Guide Formulas	val #0 sum 21600 0 @0 prod #0 1 2
Adjustment Values	16200
Connector Locations	@2,0;@1,10800;@2,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	0,0,10800,21600;0,0,16200,21600;0,0,21600,21600
Handles	position="#0,topLeft" xrange="0,21600"



Right Arrow Callout
Internal Name: RightArrowCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the width of the text rectangle area. The second adjust handle is a right perimeter adjust handle which controls the width the arrowhead. The third adjust handle controls both the length of the arrowhead, and the width of the trunk.

Geometric properties:

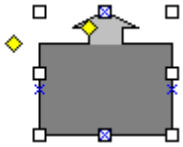
Path	m,l,21600@0,21600@0@5@2@5@2@4,21600,10800@2@1@2@3@0@3@0,x
Guide Formulas	val #0 val #1 val #2 val #3 sum 21600 0 #1 sum 21600 0 #3 prod #0 1 2
Adjustment Values	14400,5400,18000,8100
Connector Locations	@6,0;0,10800;@6,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	0,0,@0,21600
Handles	position="#0,topLeft" xrange="0,@2 position="bottomRight,#1" yrange="0,@3 position="#2,#3" xrange="@0,21600" yrange="@1,10800



Left Arrow Callout
Internal Name: LeftArrowCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the width of the text rectangle area. The second adjust handle is a left perimeter adjust handle which controls the width the arrowhead. The third adjust handle controls both the length of the arrowhead, and the width of the trunk.

Geometric properties:

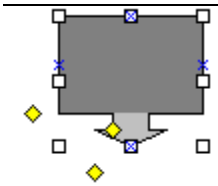
Path	m@0,l@0@3@2@3@2@1,,10800@2@4@2@5@0@5@0,21600,21600,21600,21600,xe
Guide Formulas	val #0 val #1 val #2 val #3 sum 21600 0 #1 sum 21600 0 #3 sum #0 21600 0
Adjustment Values	7200,5400,3600,8100
Connector Locations	@7,0;0,10800;@7,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	@0,0,21600,21600
Handles	position="#0,topLeft" xrange="@2,21600 position="topLeft,#1" yrange="0,@3 position="#2,#3" xrange="0,@0" yrange="@1,10800



Up Arrow Callout
Internal Name: UpArrowCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a left perimeter adjust handle which controls the height of the text rectangle area. The second adjust handle is a top perimeter adjust handle which controls the width the arrowhead. The third adjust handle controls both the length of the arrowhead, and the width of the trunk.

Geometric properties:

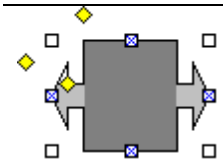
Path	m0@0l@3@0@3@2@1@2,10800,0@4@2@5@2@5@0,21600@0,21600,21600,,21600xe
Guide Formulas	val #0 val #1 val #2 val #3 sum 21600 0 #1 sum 21600 0 #3 sum #0 21600 0 prod @6 1 2
Adjustment Values	7200,5400,3600,8100
Connector Locations	10800,0;0,@7;10800,21600;21600,@7
Connector Angles	270,180,90,0
Text Box Rectangle	0,@0,21600,21600
Handles	position="topLeft,#0" yrange="@2,21600 position="#1,topLeft" xrange="0,@3 position="#3,#2" xrange="@1,10800" yrange="0,@0



Down Arrow Callout
Internal Name: DownArrowCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a left perimeter adjust handle which controls the width of the text rectangle area. The second adjust handle is a bottom perimeter adjust handle which controls the width the arrowhead. The third adjust handle controls both the length of the arrowhead, and the width of the trunk.

Geometric properties:

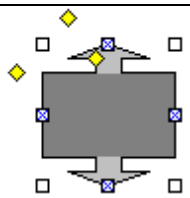
Path	m,l21600,,21600@0@5@0@5@2@4@2,10800,21600@1@2@3@2@3@0,0 @0xe
Guide Formulas	val #0 val #1 val #2 val #3 sum 21600 0 #1 sum 21600 0 #3 prod #0 1 2
Adjustment Values	14400,5400,18000,8100
Connector Locations	10800,0;0,@6;10800,21600;21600,@6
Connector Angles	270,180,90,0
Text Box Rectangle	0,0,21600,@0
Handles	position="topLeft,#0" yrange="0,@2 position="#1,bottomRight" xrange="0,@3 position="#3,#2" xrange="@1,10800" yrange="@0,21600



Left-Right Arrow Callout
Internal Name: LeftRightArrowCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the width of the text rectangle area. The second adjust handle is a right perimeter adjust handle which controls the width the arrowhead. The third adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the arrows at each end are the same size.

Geometric properties:

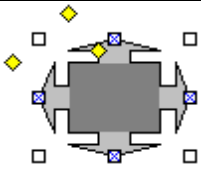
Path	m@0,l@0@3@2@3@2@1,,10800@2@4@2@5@0@5@0,21600@8,21600 @8@5@9@5@9@4,21600,10800@9@1@9@3@8@3@8,xe
Guide Formulas	val #0 val #1 val #2 val #3 sum 21600 0 #1 sum 21600 0 #3 sum #0 21600 0 prod @6 1 2 sum 21600 0 #0 sum 21600 0 #2
Adjustment Values	5400,5400,2700,8100
Connector Locations	10800,0;0,10800;10800,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	@0,0,@8,21600
Handles	position="#0,topLeft" xrange="@2,10800 position="topLeft,#1" yrange="0,@3 position="#2,#3" xrange="0,@0" yrange="@1,10800



Up-Down Arrow Callout
Internal Name: UpDownArrowCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a left perimeter adjust handle which controls the width of the text rectangle area. The second adjust handle is a top perimeter adjust handle which controls the width the arrowhead. The third adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the arrows at each end are the same size.

Geometric properties:

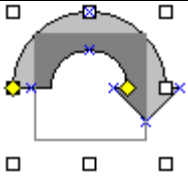
Path	m0@0l@3@0@3@2@1@2,10800,0@4@2@5@2@5@0,21600@0,21600@8@5@8@5@9@4@9,10800,21600@1@9@3@9@3@8,0@8xe
Guide Formulas	val #0 val #1 val #2 val #3 sum 21600 0 #1 sum 21600 0 #3 sum #0 21600 0 prod @6 1 2 sum 21600 0 #0 sum 21600 0 #2
Adjustment Values	5400,5400,2700,8100
Connector Locations	10800,0;0,10800;10800,21600;21600,10800
Connector Angles	270,180,90,0
Text Box Rectangle	0,@0,21600,@8
Handles	position="topLeft,#0" yrange="@2,10800 position="#1,topLeft" xrange="0,@3 position="#3,#2" xrange="@1,10800" yrange="0,@0



Quad Arrow Callout
Internal Name: CloudCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a left perimeter adjust handle which controls the width of the text rectangle area. The second adjust handle is a top perimeter adjust handle which controls the width the arrowhead. The third adjust handle controls both the length of the arrowhead, and the width of the trunk. It stays symmetric so that the arrows on each side stay the same size when the shape is scaled to a 1:1 aspect ratio.

Geometric properties:

Path	m@0@0l@3@0@3@2@1@2,10800,0@4@2@5@2@5@0@8@0@8@3@9@3@9@1,21600,10800@9@4@9@5@8@5@8@8@5@8@5@9@4@9,10800,21600@1@9@3@9@3@8@0@8@0@5@2@5@2@4,,10800@2@1@2@3@0@3xe
Guide Formulas	val #0 val #1 val #2 val #3 sum 21600 0 #1 sum 21600 0 #3 sum #0 21600 0 prod @6 1 2 sum 21600 0 #0 sum 21600 0 #2
Adjustment Values	5400,8100,2700,9450
Connector Locations	Rectangle
Text Box Rectangle	@0,@0,@8,@8
Handles	position="topLeft,#0" yrange="@2,@1 position="#1,topLeft" xrange="@0,@3 position="#3,#2" xrange="@1,10800" yrange="0,@0

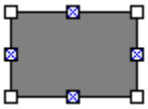


Circular Arrow
Internal Name: CircularArrow
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A polar adjust handle along the outer circle controls the starting angle. A second polar adjust handle controls the ending angle, and the inner radius. The shape goes clockwise from the starting angle to the ending angle.

Geometric properties:

Path	al10800,10800@8@8@4@6,10800,10800,10800,10800@9@7l@30@31@17 @18@24@25@15@16@32@33xe
Guide Formulas	val #1 val #0 sum #1 0 #0 val 10800 sum 0 0 #1 sumangle @2 360 0 if @2 @2 @5 sum 0 0 @6 val #2 sum 0 0 #0 sum #2 0 2700 cos @10 #1 sin @10 #1 cos 13500 #1 sin 13500 #1 sum @11 10800 0 sum @12 10800 0 sum @13 10800 0 sum @14 10800 0 prod #2 1 2 sum @19 5400 0 cos @20 #1 sin @20 #1 sum @21 10800 0 sum @12 @23 @22 sum @22 @23 @11 cos 10800 #1 sin 10800 #1 cos #2 #1 sin #2 #1 sum @26 10800 0 sum @27 10800 0 sum @28 10800 0 sum @29 10800 0 sum @19 5400 0 cos @34 #0 sin @34 #0 mid #0 #1 sumangle @37 180 0 if @2 @37 @38 cos 10800 @39 sin 10800 @39 cos #2 @39 sin #2 @39 sum @40 10800 0 sum @41 10800 0 sum @42 10800 0 sum @43 10800 0 sum @35 10800 0 sum @36 10800 0
Adjustment Values	-11796480,,5400
Connector Locations	@44,@45;@48,@49;@46,@47;@17,@18;@24,@25;@15,@16
Text Box Rectangle	3163,3163,18437,18437
Handles	position="@3,#0" polar="10800,10800" position="#2,#1" polar="10800,10800" radiusrange="0,10800

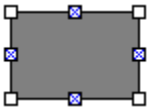
Flowchart



Flowchart: Process
Internal Name: FlowChartProcess
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

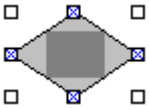
Path	m,l,21600r21600,l21600,xe
Connector Locations	Rectangle



Flowchart: Alternate Process
Internal Name: FlowChartAlternateProcess
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

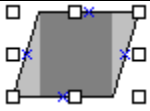
Path	m@0,qx0@0l0@2qy@0,21600l@1,21600qx21600@2l21600@0qy@1,xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod @0 2929 10000 sum width 0 @3 sum height 0 @3 val width val height prod width 1 2 prod height 1 2
Adjustment Values	2700
Connector Locations	@8,0;0,@9;@8,@7;@6,@9
Text Box Rectangle	@3,@3,@4,@5
Limo	10800,10800



Flowchart: Decision
Internal Name: FlowChartDecision
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

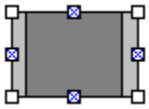
Path	m10800,l,10800,10800,21600,21600,10800xe
Connector Locations	Rectangle
Text Box Rectangle	5400,5400,16200,16200



Flowchart: Data
Internal Name: FlowChartInputOutput
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

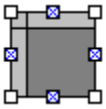
Path	m4321,l21600,,17204,21600,,21600xe
Connector Locations	12961,0;10800,0;2161,10800;8602,21600;10800,21600;19402,10800
Text Box Rectangle	4321,0,17204,21600



Flowchart: Predefined Process
Internal Name: FlowChartPredefinedProcess
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

Path	m,l,21600r21600,l21600,xem2610,nfl2610,21600em18990,nfl18990,21600e
Connector Locations	Rectangle
Text Box Rectangle	2610,0,18990,21600



Flowchart: Internal Storage
Internal Name: FlowChartInternalStorage
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

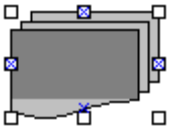
Geometric properties:

Path	m,l,21600r21600,l21600,xem4236,nfl4236,21600em,4236nfl21600,4236e
Connector Locations	Rectangle
Text Box Rectangle	4236,4236,21600,21600

Flowchart: Document
Internal Name: FlowChartDocument
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

Path	m,20172v945,400,1887,628,2795,913c3587,21312,4342,21370,5060,21597v2037,,2567,-227,3095,-285c8722,21197,9325,20970,9855,20800v490,-228,945,-400,1472,-740c11817,19887,12347,19660,12875,19375v567,-228,1095,-513,1700,-740c15177,18462,15782,18122,16537,17950v718,-113,1398,-398,2228,-513c19635,17437,20577,17322,21597,17322l21597,,,xe
Connector Locations	10800,0;0,10800;10800,20400;21600,10800
Text Box Rectangle	0,0,21600,17322



Flowchart: Multidocument
Internal Name: FlowChartMultidocument
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

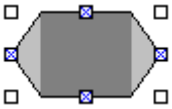
Geometric properties:

Path	m,20465v810,317,1620,452,2397,725c3077,21325,3790,21417,4405,21597v1620,,2202,-180,2657,-272c7580,21280,8002,21010,8455,20917v422,-135,810,-405,1327,-542c10205,20150,10657,19967,11080,19742v517,-182,970,-407,1425,-590c13087,19017,13605,18745,14255,18610v615,-180,1262,-318,1942,-408c16975,18202,17785,18022,18595,18022r,-1670l19192,16252r808,l20000,14467r722,-75l21597,14392,21597,,2972,r,1815l1532,1815r,1860l3675,,20465xem1532,3675nfl18595,3675r,12677em2972,1815nfl20000,1815r,12652e
Connector Locations	10800,0;0,10800;10800,19890;21600,10800
Text Box Rectangle	0,3675,18595,18022

Flowchart: Terminator
Internal Name: FlowChartTerminator
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

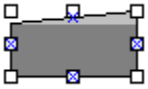
Path	m3475,qx,10800,3475,21600l18125,21600qx21600,10800,18125,xe
Connector Locations	Rectangle
Text Box Rectangle	1018,3163,20582,18437



Flowchart: Preparation
Internal Name: FlowChartPreparation
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

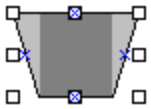
Path	m4353,l17214,r4386,10800l17214,21600r-12861,l,10800xe
Connector Locations	Rectangle
Text Box Rectangle	4353,0,17214,21600



Flowchart: Manual Input
Internal Name: FlowChartManualInput
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

Path	m,4292l21600,r,21600l,21600xe
Connector Locations	10800,2146;0,10800;10800,21600;21600,10800
Text Box Rectangle	0,4291,21600,21600



Flowchart: Manual Operation
Internal Name: FlowChartManualOperation
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

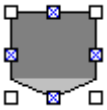
Geometric properties:

Path	m,l21600,,17240,21600r-12880,xe
Connector Locations	10800,0;2180,10800;10800,21600;19420,10800
Text Box Rectangle	4321,0,17204,21600

FlowChart: Connector
Internal Name: FlowChartConnector
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: None. *Issue: the name "connector" is confusing since Escher has another type of shape called a "connector", but this is the official name of this Flowchart shape.*

Geometric properties:

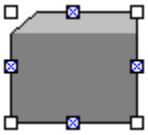
Path	m10800,qx,10800,10800,21600,21600,10800,10800,xe
Connector Locations	10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437,18437;21600,10800;18437,3163
Text Box Rectangle	3163,3163,18437,18437



FlowChart: Off-page Connector
Internal Name: FlowChartOffpageConnector
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

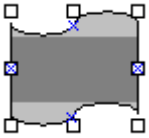
Path	m,l21600,r,17255l10800,21600,,17255xe
Connector Locations	Rectangle
Text Box Rectangle	0,0,21600,17255



FlowChart: Card
Internal Name: FlowChartPunchedCard
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

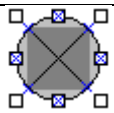
Path	m4321,l21600,r,21600l21600,,4338xe
Connector Locations	Rectangle
Text Box Rectangle	0,4321,21600,21600



FlowChart: Punched Tape
Internal Name: FlowChartPunchedTape
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

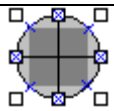
Path	m21597,19450v-225,-558,-750,-1073,-1650,-1545c18897,17605,17585,17347,16197,17260v-1500,87,-2700,345,-3787,645c11472,18377,10910,18892,10800,19450v-188,515,-750,1075,-1613,1460c8100,21210,6825,21425,5400,21597,3937,21425,2700,21210,1612,20910,675,20525,150,19965,,19450l,2147v150,558,675,1073,1612,1460c2700,3950,3937,4165,5400,4337,6825,4165,8100,3950,9187,3607v863,-387,1425,-902,1613,-1460c10910,1632,11472,1072,12410,600,13497,300,14697,85,16197,v1388,85,2700,300,3750,600c20847,1072,21372,1632,21597,2147xe
Connector Locations	10800,2147;0,10800;10800,19450;21600,10800
Text Box Rectangle	0,4337,21600,17260



FlowChart: Summing Junction
Internal Name: FlowChartSummingJunction
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

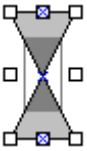
Path	m10800,qx,10800,10800,21600,21600,10800,10800,xem3163,3163nfl18437,18437em3163,18437nfl18437,3163e
Connector Locations	10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437,18437;21600,10800;18437,3163
Text Box Rectangle	3163,3163,18437,18437



FlowChart: Or
Internal Name: FlowChartOr
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

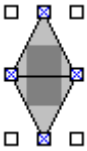
Path	m10800,qx,10800,10800,21600,21600,10800,10800,xem,10800nfl21600,10800em10800,nfl10800,21600e
Connector Locations	10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437,18437;21600,10800;18437,3163
Text Box Rectangle	3163,3163,18437,18437



FlowChart: Collate
Internal Name: FlowChartCollate
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

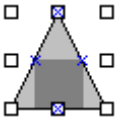
Path	m21600,21600l,21600,21600,,,xe
Connector Locations	10800,0;10800,10800;10800,21600
Text Box Rectangle	5400,5400,16200,16200



FlowChart: Sort
Internal Name: FlowChartSort
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

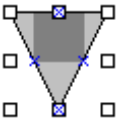
Path	m10800,l,10800,10800,21600,21600,10800xem,10800nfl21600,10800e
Connector Locations	Rectangle
Text Box Rectangle	5400,5400,16200,16200



FlowChart: Extract
Internal Name: FlowChartExtract
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

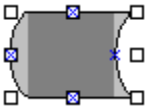
Path	m10800,l21600,21600,,21600xe
Connector Locations	10800,0;5400,10800;10800,21600;16200,10800
Text Box Rectangle	5400,10800,16200,21600



FlowChart: Merge
Internal Name: FlowChartMerge
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

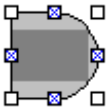
Path	m,l21600,,10800,21600xe
Connector Locations	10800,0;5400,10800;10800,21600;16200,10800
Text Box Rectangle	5400,0,16200,10800



FlowChart: Stored Data
Internal Name: FlowChartOnlineStorage
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

Path	m3600,21597c2662,21202,1837,20075,1087,18440,487,16240,75,13590,,1077 0,75,8007,487,5412,1087,3045,1837,1465,2662,337,3600,l21597,v-937,337,-1 687,1465,-2512,3045c18485,5412,18072,8007,17997,10770v75,2820,488,547 0,1088,7670c19910,20075,20660,21202,21597,21597xe
Connector Locations	10800,0;0,10800;10800,21600;17997,10800
Text Box Rectangle	3600,0,17997,21600



FlowChart: Delay
Internal Name: FlowChartDelay
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

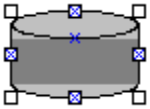
Geometric properties:

Path	m10800,qx21600,10800,10800,21600l,21600,,xe
Connector Locations	Rectangle
Text Box Rectangle	0,3163,18437,18437

FlowChart: Sequential Access Storage
Internal Name: FlowChartMagneticTape
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

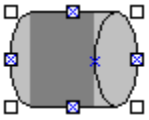
Path	ar,,21600,21600,18685,18165,10677,21597l20990,21597r,-3432xe
Connector Locations	Rectangle
Text Box Rectangle	3163,3163,18437,18437



FlowChart: Magnetic Disk
Internal Name: FlowChartMagneticDisk
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

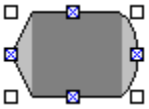
Path	m10800,qx,3391l,18209qy10800,21600,21600,18209l21600,3391qy10800,xe m,3391nfqy10800,6782,21600,3391e
Connector Locations	10800,6782;10800,0;0,10800;10800,21600;21600,10800
Connector Angles	270,270,180,90,0
Text Box Rectangle	0,6782,21600,18209



FlowChart: Direct Access Storage
Internal Name: FlowChartMagneticDrum
Shaped Concentric Fill: Yes.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

Path	m21600,10800qy18019,21600l3581,21600qx,10800,3581,l18019,qx21600,10 800xem18019,21600nfqx14438,10800,18019,e
Connector Locations	10800,0;0,10800;10800,21600;14438,10800;21600,10800
Connector Angles	270,180,90,0,0
Text Box Rectangle	3581,0,14438,21600

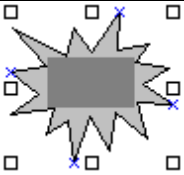


FlowChart: Display
Internal Name: FlowChartDisplay
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

Path	m17955,v862,282,1877,1410,2477,3045c21035,5357,21372,7895,21597,10827v-225,2763,-562,5300,-1165,7613c19832,20132,18817,21260,17955,21597r-14388,l,10827,3567,xe
Connector Locations	Rectangle
Text Box Rectangle	3567,0,17955,21600

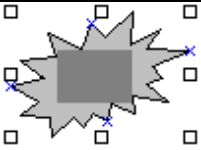
Stars & Banners



Explosion 1
Internal Name: IrregularSeal1
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

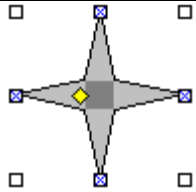
Path	m10800,5800l8352,2295,7312,6320,370,2295,4627,7617,,8615r3722,3160l135,14587r5532,-650l4762,17617,7715,15627r770,5973l10532,14935r2715,4802l14020,14457r4125,3638l16837,12942r4763,348l17607,10475,21097,8137,16702,7315,18380,4457r-4225,868l14522,xe
Connector Locations	14522,0;0,8615;8485,21600;21600,13290
Connector Angles	270,180,90,0
Text Box Rectangle	4627,6320,16702,13937



Explosion 2
Internal Name: IrregularSeal2
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

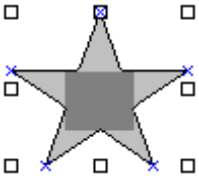
Path	m11462,4342l9722,1887,8550,6382,4502,3625r870,4192l1172,8270r2763,3322l,12877r3330,2493l1285,17825r3520,415l4917,21600,7527,18125r1173,1587l9872,17370r1740,1472l12180,15935r2762,1435l14640,14350r4237,1282l16380,12310r1890,-1020l16985,9402,21600,6645,16380,6532,18007,3172,14525,5777,14790,xe
Connector Locations	9722,1887;0,12877;11612,18842;21600,6645
Connector Angles	270,180,90,0
Text Box Rectangle	5372,6382,14640,15935



4-point Star
Internal Name: Seal4
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A horizontal adjust handle along the center adjusts the inner radius of the seal. The adjust handle can extend halfway across.

Geometric properties:

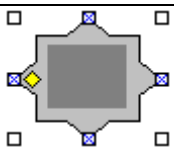
Path	m21600,10800l@2@3,10800,0@3@3,,10800@3@2,10800,21600@2@2xe
Guide Formulas	sum 10800 0 #0 prod @0 23170 32768 sum @1 10800 0 sum 10800 0 @1
Adjustment Values	8100
Connector Locations	Rectangle
Text Box Rectangle	@3,@3,@2,@2
Handles	position="#0,center" xrange="0,10800



5-point Star
Internal Name: Star
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: None.

Geometric properties:

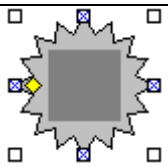
Path	m10800,l8280,8259,,8259r6720,5146l4200,21600r6600,-5019l17400,21600,14880,13405,21600,8259r-8280,xe
Connector Locations	10800,0;0,8259;4200,21600;17400,21600;21600,8259
Text Box Rectangle	6720,8259,14880,15628



8-point Star
Internal Name: Seal8
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A horizontal adjust handle along the center adjusts the inner radius of the seal. The adjust handle can extend halfway across.

Geometric properties:

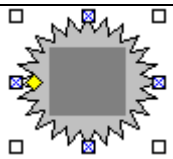
Path	m21600,10800l@3@6,18436,3163@4@5,10800,0@6@5,3163,3163@5@6,,10800@5@4,3163,18436@6@3,10800,21600@4@3,18436,18436@3@4xe
Guide Formulas	sum 10800 0 #0 prod @0 30274 32768 prod @0 12540 32768 sum @1 10800 0 sum @2 10800 0 sum 10800 0 @1 sum 10800 0 @2 prod @0 23170 32768 sum @7 10800 0 sum 10800 0 @7
Adjustment Values	2538
Connector Locations	Rectangle
Text Box Rectangle	@9,@9,@8,@8
Handles	position="#0,center" xrange="0,10800



16-point Star
Internal Name: Seal16
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A horizontal adjust handle along the center adjusts the inner radius of the seal. The adjust handle can extend halfway across.

Geometric properties:

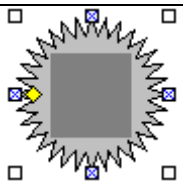
Path	m21600,10800l@5@10,20777,6667@7@12,18436,3163@8@11,14932,822 @6@9,10800,0@10@9,6667,822@12@11,3163,3163@11@12,822,6667@9 @10,,10800@9@6,822,14932@11@8,3163,18436@12@7,6667,20777@10 @5,10800,21600@6@5,14932,20777@8@7,18436,18436@7@8,20777,1493 2@5@6xe
Guide Formulas	sum 10800 0 #0 prod @0 32138 32768 prod @0 6393 32768 prod @0 27246 32768 prod @0 18205 32768 sum @1 10800 0 sum @2 10800 0 sum @3 10800 0 sum @4 10800 0 sum 10800 0 @1 sum 10800 0 @2 sum 10800 0 @3 sum 10800 0 @4 prod @0 23170 32768 sum @13 10800 0 sum 10800 0 @13
Adjustment Values	2700
Connector Locations	Rectangle
Text Box Rectangle	@15,@15,@14,@14
Handles	position="#0,center" xrange="0,10800



24-point Star
Internal Name: Seal24
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A horizontal adjust handle along the center adjusts the inner radius of the seal. The adjust handle can extend halfway across.

Geometric properties:

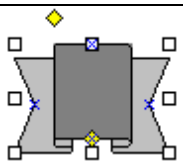
Path	m21600,10800l@7@14,21232,8005@9@16,20153,5400@11@18,18437,3163@12@17,16200,1447@10@15,13595,368@8@13,10800,0@14@13,8005,368@16@15,5400,1447@18@17,3163,3163@17@18,1447,5400@15@16,368,8005@13@14,,10800@13@8,368,13595@15@10,1447,16200@17@12,3163,18437@18@11,5400,20153@16@9,8005,21232@14@7,10800,21600@8@7,13595,21232@10@9,16200,20153@12@11,18437,18437@11@12,20153,16200@9@10,21232,13595@7@8xe
Guide Formulas	sum 10800 0 #0 prod @0 32488 32768 prod @0 4277 32768 prod @0 30274 32768 prod @0 12540 32768 prod @0 25997 32768 prod @0 19948 32768 sum @1 10800 0 sum @2 10800 0 sum @3 10800 0 sum @4 10800 0 sum @5 10800 0 sum @6 10800 0 sum 10800 0 @1 sum 10800 0 @2 sum 10800 0 @3 sum 10800 0 @4 sum 10800 0 @5 sum 10800 0 @6 prod @0 23170 32768 sum @19 10800 0 sum 10800 0 @19
Adjustment Values	2700
Connector Locations	Rectangle
Text Box Rectangle	@21,@21,@20,@20
Handles	position="#0,center" xrange="0,10800



32-point Star
Internal Name: Seal32
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A horizontal adjust handle along the center adjusts the inner radius of the seal. The adjust handle can extend halfway across.

Geometric properties:

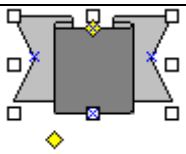
Path	m21600,10800l@9@18,21392,8693@11@20,20777,6667@13@22,19780,4800@15@24,18436,3163@16@23,16800,1820@14@21,14932,822@12@19,12907,208@10@17,10800,0@18@17,8693,208@20@19,6667,822@22@21,4800,1820@24@23,3163,3163@23@24,1820,4800@21@22,822,6667@19@20,208,8693@17@18,,10800@17@10,208,12907@19@12,822,14932@21@14,1820,16800@23@16,3163,18436@24@15,4800,19780@22@13,6667,20777@20@11,8693,21392@18@9,10800,21600@10@9,12907,21392@12@11,14932,20777@14@13,16800,19780@16@15,18436,18436@15@16,19780,16800@13@14,20777,14932@11@12,21392,12907@9@10xe
Guide Formulas	sum 10800 0 #0 prod @0 32610 32768 prod @0 3212 32768 prod @0 31357 32768 prod @0 9512 32768 prod @0 28899 32768 prod @0 15447 32768 prod @0 25330 32768 prod @0 20788 32768 sum @1 10800 0 sum @2 10800 0 sum @3 10800 0 sum @4 10800 0 sum @5 10800 0 sum @6 10800 0 sum @7 10800 0 sum @8 10800 0 sum 10800 0 @1 sum 10800 0 @2 sum 10800 0 @3 sum 10800 0 @4 sum 10800 0 @5 sum 10800 0 @6 sum 10800 0 @7 sum 10800 0 @8 prod @0 23170 32768 sum @25 10800 0 sum 10800 0 @25
Adjustment Values	2700
Connector Locations	Rectangle
Text Box Rectangle	@27,@27,@26,@26
Handles	position="#0,center" xrange="0,10800



Up Ribbon
Internal Name: Ribbon2
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the position of the fold. The second adjust handle is a vertical adjust handle along the center which controls the tilt of the ribbon. The first adjust handle can extend as far left as the notch in the side of the ribbon and as far right as where the folds meet in the center. The second adjust handle can extend 1/3 of the way up.

Geometric properties:

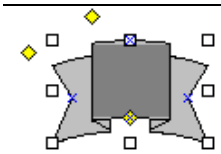
Path	m0@29l@3@29qx@4@19l@4@10@5@10@5@19qy@6@29l@28@29@26@22@28@23@9@23@9@24qy@8,l@1,qx@0@24l@0@23,0@23,2700@22xem@4@19nfqy@3@20l@1@20qx@0@21@1@10l@4@10em@5@19nfqy@6@20l@8@20qx@9@21@8@10l@5@10em@0@21nfl@0@23em@9@21nfl@9@23e
Guide Formulas	val #0 sum @0 675 0 sum @1 675 0 sum @2 675 0 sum @3 675 0 sum width 0 @4 sum width 0 @3 sum width 0 @2 sum width 0 @1 sum width 0 @0 val #1 prod @10 1 4 prod @10 1 2 prod @10 3 4 prod height 3 4 prod height 1 2 prod height 1 4 prod height 3 2 prod height 2 3 sum @11 @14 0 sum @12 @15 0 sum @13 @16 0 sum @17 0 @20 sum height 0 @10 sum height 0 @19 prod width 1 2 sum width 0 2700 sum @25 0 2700 val width val height
Adjustment Values	5400,18900
Connector Locations	@25,0;2700,@22;@25,@10;@26,@22
Connector Angles	270,180,90,0
Text Box Rectangle	@0,0,@9,@10
Handles	position="#0,topLeft" xrange="2700,8100 position="center,#1" yrange="14400,21600



Down Ribbon
Internal Name: Ribbon
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a bottom perimeter adjust handle which controls the position of the fold. The second adjust handle is a vertical adjust handle along the center which controls the tilt of the ribbon. The first adjust handle can extend as far left as the notch in the side of the ribbon and as far right as where the folds meet in the center. The second adjust handle can extend 1/3 of the way down.

Geometric properties:

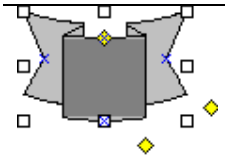
Path	m,l@3,qx@4@11l@4@10@5@10@5@11qy@6,l@21,0@19@15@21@16 @9@16@9@17qy@8@22l@1@22qx@0@17l@0@16,0@16,2700@15xem @4@11nfqy@3@12l@1@12qx@0@13@1@10l@4@10em@5@11nfqy@6 @12l@8@12qx@9@13@8@10l@5@10em@0@13nfl@0@16em@9@13nfl @9@16e
Guide Formulas	val #0 sum @0 675 0 sum @1 675 0 sum @2 675 0 sum @3 675 0 sum width 0 @4 sum width 0 @3 sum width 0 @2 sum width 0 @1 sum width 0 @0 val #1 prod @10 1 4 prod @11 2 1 prod @11 3 1 prod height 1 2 sum @14 0 @12 sum height 0 @10 sum height 0 @11 prod width 1 2 sum width 0 2700 sum @18 0 2700 val width val height
Adjustment Values	5400,2700
Connector Locations	@18,@10;2700,@15;@18,21600;@19,@15
Connector Angles	270,180,90,0
Text Box Rectangle	@0,@10,@9,21600
Handles	position="#0,bottomRight" xrange="2700,8100 position="center,#1" yrange="0,7200



Curved Up Ribbon
Internal Name: EllipseRibbon2
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a top perimeter adjust handle which controls the position of the fold. The second adjust handle is a vertical adjust handle along the center which controls the tilt of the ribbon. The third adjust handle is a left perimeter adjust handle which controls the curvature of the ribbon. The first adjust handle can extend as far left as the notch in the side of the ribbon and as far right as where the folds meet in the center.

Geometric properties:

Path	wr@9@34@8@35,0@24@0@23@9,0@8@11@0@22@19@22@9@34@8 @35@19@23@3@24l@7@36@3@4at@9@31@8@32@3@4@18@30@9 @1@8@33@18@28@17@28@9@31@8@32@17@30,0@4l@5@36xear@ 9@1@8@33@17@28@0@29nfl@17@30ewr@9@1@8@33@18@28@19 @29nfl@18@30em@0@23nfl@0@29em@19@23nfl@19@29e
Guide Formulas	val #0 val #1 val #2 val width val height prod width 1 8 prod width 1 2 prod width 7 8 prod width 3 2 sum 0 0 @6 prod #2 30573 4096 prod @10 2 1 sum @10 height #2 sum @10 #1 0 prod #1 1 2 sum @10 @14 0 sum @12 0 #1 sum #0 @5 0 sum width 0 @17 sum width 0 #0 sum @6 0 #0 ellipse @20 width @10 sum @10 0 @21 sum @22 @16 @10 sum #2 @16 @10 prod @10 2391 32768 sum @6 0 @17 ellipse @26 width @10 sum @10 #1 @27 sum @22 #1 0 sum @12 0 @27 sum height 0 #2 sum @10 @12 0 sum @32 @10 @16 sum @31 @10 @13 sum @32 @10 @13 sum @25 @12 @15 sum @16 0 @15 prod @37 2 3 sum @1 @38 0 sum #2 @38 0 max @40 675 prod width 3 8 sum @42 0 4
Adjustment Values	5400,16200,2700
Connector Locations	@6,0;@5,@36;@6,@1;@7,@36
Connector Angles	270,180,90,0
Text Box Rectangle	@0,@22,@19,@1
Handles	position="#0,topLeft" xrange="@5,@43 position="center,#1" yrange="@39,@31 position="topLeft,#2" yrange="@41,@24

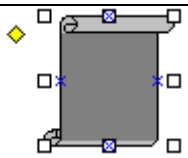


Curved Down Ribbon
Internal Name: EllipseRibbon
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The first adjust handle is a bottom perimeter adjust handle which controls the position of the fold. The second adjust handle is a vertical adjust handle along the center which controls the tilt of the ribbon. The third adjust handle is a left perimeter adjust handle which controls the curvature of the ribbon. The first adjust handle can extend as far left as the notch in the side of the ribbon and as far right as where the folds meet in the center.

Geometric properties:

Path	ar@9@38@8@37,0@27@0@26@9@13@8@4@0@25@22@25@9@38@8@37@22@26@3@27l@7@40@3,wa@9@35@8@10@3,0@21@33@9@36@8@1@21@31@20@31@9@35@8@10@20@33,,l@5@40xewr@9@36@8@1@20@31@0@32nfl@20@33ear@9@36@8@1@21@31@22@32nfl@21@33em@0@26nfl@0@32em@22@26nfl@22@32e
Guide Formulas	val #0 val #1 val #2 val width val height prod width 1 8 prod width 1 2 prod width 7 8 prod width 3 2 sum 0 0 @6 sum height 0 #2 prod @10 30573 4096 prod @11 2 1 sum height 0 @12 sum @11 #2 0 sum @11 height #1 sum height 0 #1 prod @16 1 2 sum @11 @17 0 sum @14 #1 height sum #0 @5 0 sum width 0 @20 sum width 0 #0 sum @6 0 #0 ellipse @23 width @11 sum @24 height @11 sum @25 @11 @19 sum #2 @11 @19 prod @11 2391 32768 sum @6 0 @20 ellipse @29 width @11 sum #1 @30 @11 sum @25 #1 height sum height @30 @14 sum @11 @14 0 sum height 0 @34 sum @35 @19 @11 sum @10 @15 @11 sum @35 @15 @11 sum @28 @14 @18 sum height 0 @39 sum @19 0 @18 prod @41 2 3 sum #1 0 @42 sum #2 0 @42 min @44 20925 prod width 3 8 sum @46 0 4
Adjustment Values	5400,5400,18900
Connector Locations	@6,@1;@5,@40;@6,@4;@7,@40
Connector Angles	270,180,90,0
Text Box Rectangle	@0,@1,@22,@25

Handles	position="#0,bottomRight" xrange="@5,@47 position="center,#1" yrange="@10,@43 position="topLeft,#2" yrange="@27,@45

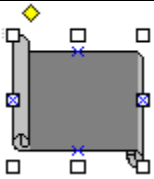


Vertical Scroll

Internal Name: VerticalScroll
Shaped Concentric Fill: No.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: A left perimeter adjust handle controls the size of the roll on both ends. It limo-stretches both vertically and horizontally, so that the roll is always circular at both ends.

Geometric properties:

Path	m@5,qx@1@2l@1@0@2@0qx0@7@2,21600l@9,21600qx@10@7l@10@1@11@1qx21600@2@11,xem@5,nfqx@6@2@5@1@4@3@5@2l@6@2em@5@1nfl@10@1em@2,21600nfqx@1@7l@1@0em@2@0nfqx@3@8@2@7l@1@7e
Guide Formulas	sum height 0 #0 val #0 prod @1 1 2 prod @1 3 4 prod @1 5 4 prod @1 3 2 prod @1 2 1 sum height 0 @2 sum height 0 @3 sum width 0 @5 sum width 0 @1 sum width 0 @2 val height prod height 1 2 prod width 1 2
Adjustment Values	2700
Connector Locations	@14,0;@1,@13;@14,@12;@10,@13
Connector Angles	270,180,90,0
Text Box Rectangle	@1,@1,@10,@7
Handles	position="topLeft,#0" yrange="0,5400
Limo	10800,10800

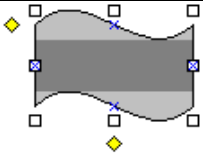


(*actually this image is flipped vertically*)

Horizontal Scroll
Internal Name: HorizontalScroll
Shaped Concentric Fill: No.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: A top perimeter adjust handle controls the size of the roll on both ends. It limo-stretches both vertically and horizontally, so that the roll is always circular at both ends.

Geometric properties:

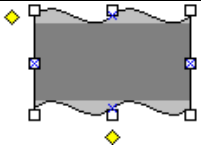
Path	m0@5qy@2@1l@0@1@0@2qy@7,,21600@2l21600@9qy@7@10l@1@10 @1@11qy@2,21600,0@11xem0@5nfqy@2@6@1@5@3@4@2@5l@2@6e m@1@5nfl@1@10em21600@2nfqy@7@1l@0@1em@0@2nfqy@8@3@7 @2l@7@1e
Guide Formulas	sum width 0 #0 val #0 prod @1 1 2 prod @1 3 4 prod @1 5 4 prod @1 3 2 prod @1 2 1 sum width 0 @2 sum width 0 @3 sum height 0 @5 sum height 0 @1 sum height 0 @2 val width prod width 1 2 prod height 1 2
Adjustment Values	2700
Connector Locations	@13,@1;0,@14;@13,@10;@12,@14
Connector Angles	270,180,90,0
Text Box Rectangle	@1,@1,@7,@10
Handles	position="#0,topLeft" xrange="0,5400
Limo	10800,10800



Wave
Internal Name: Wave
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A left perimeter adjust handle controls the amplitude of the wave. A bottom perimeter adjust handle controls the "skew" of the wave.

Geometric properties:

Path	m@28@0c@27@1@26@3@25@0l@21@4c@22@5@23@6@24@4xe
Guide Formulas	val #0 prod @0 41 9 prod @0 23 9 sum 0 0 @2 sum 21600 0 #0 sum 21600 0 @1 sum 21600 0 @3 sum #1 0 10800 sum 21600 0 #1 prod @8 2 3 prod @8 4 3 prod @8 2 1 sum 21600 0 @9 sum 21600 0 @10 sum 21600 0 @11 prod #1 2 3 prod #1 4 3 prod #1 2 1 sum 21600 0 @15 sum 21600 0 @16 sum 21600 0 @17 if @7 @14 0 if @7 @13 @15 if @7 @12 @16 if @7 21600 @17 if @7 0 @20 if @7 @9 @19 if @7 @10 @18 if @7 @11 21600 sum @24 0 @21 sum @4 0 @0 max @21 @25 min @24 @28 prod @0 2 1 sum 21600 0 @33 mid @26 @27 mid @24 @28 mid @22 @23 mid @21 @25
Adjustment Values	2809,10800
Connector Locations	@35,@0;@38,10800;@37,@4;@36,10800
Connector Angles	270,180,90,0
Text Box Rectangle	@31,@33,@32,@34
Handles	position="topLeft,#0" yrange="0,4459 position="#1,bottomRight" xrange="8640,12960

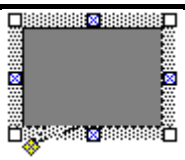


Double Wave
Internal Name: Wave
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A left perimeter adjust handle controls the amplitude of the wave. A bottom perimeter adjust handle controls the "skew" of the wave.

Geometric properties:

Path	m@43@0c@42@1@41@3@40@0@39@1@38@3@37@0l@30@4c@31@5@32@6@33@4@34@5@35@6@36@4xe
Guide Formulas	val #0 prod @0 41 9 prod @0 23 9 sum 0 0 @2 sum 21600 0 #0 sum 21600 0 @1 sum 21600 0 @3 sum #1 0 10800 sum 21600 0 #1 prod @8 1 3 prod @8 2 3 prod @8 4 3 prod @8 5 3 prod @8 2 1 sum 21600 0 @9 sum 21600 0 @10 sum 21600 0 @8 sum 21600 0 @11 sum 21600 0 @12 sum 21600 0 @13 prod #1 1 3 prod #1 2 3 prod #1 4 3 prod #1 5 3 prod #1 2 1 sum 21600 0 @20 sum 21600 0 @21 sum 21600 0 @22 sum 21600 0 @23 sum 21600 0 @24 if @7 @19 0 if @7 @18 @20 if @7 @17 @21 if @7 @16 #1 if @7 @15 @22 if @7 @14 @23 if @7 21600 @24 if @7 0 @29 if @7 @9 @28 if @7 @10 @27 if @7 @8 @8 if @7 @11 @26 if @7 @12 @25 if @7 @13 21600 sum @36 0 @30 sum @4 0 @0 max @30 @37 min @36 @43 prod @0 2 1 sum 21600 0 @48 mid @36 @43 mid @30 @37
Adjustment Values	1404,10800
Connector Locations	@40,@0;@51,10800;@33,@4;@50,10800
Connector Angles	270,180,90,0
Text Box Rectangle	@46,@48,@47,@49
Handles	position="topLeft,#0" yrange="0,2229 position="#1,bottomRight" xrange="8640,12960

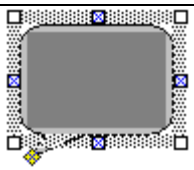
Callouts



Rectangular Callout
Internal Name: WedgeRectCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls the position of the point. The wedge extends from the center of the edge that is closest to the point. If the point is placed inside the rectangle, no wedge is drawn. Note that part of the shape extends past the geometry box of the shape.

Geometric properties:

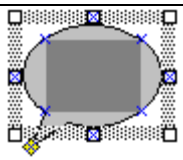
Path	m,l0@8@12@24,0@9,,21600@6,21600@15@27@7,21600,21600,21600,21600@9@18@30,21600@8,21600,0@7,0@21@33@6,xe
Guide Formulas	sum 10800 0 #0 sum 10800 0 #1 sum #0 0 #1 sum @0 @1 0 sum 21600 0 #0 sum 21600 0 #1 if @0 3600 12600 if @0 9000 18000 if @1 3600 12600 if @1 9000 18000 if @2 0 #0 if @3 @10 0 if #0 0 @11 if @2 @6 #0 if @3 @6 @13 if @5 @6 @14 if @2 #0 21600 if @3 21600 @16 if @4 21600 @17 if @2 #0 @6 if @3 @19 @6 if #1 @6 @20 if @2 @8 #1 if @3 @22 @8 if #0 @8 @23 if @2 21600 #1 if @3 21600 @25 if @5 21600 @26 if @2 #1 @8 if @3 @8 @28 if @4 @8 @29 if @2 #1 0 if @3 @31 0 if #1 0 @32 val #0 val #1
Adjustment Values	1350,25920
Connector Locations	10800,0;0,10800;10800,21600;21600,10800;@34,@35
Handles	position="#0,#1



Rounded Rectangular Callout
Internal Name: WedgeRRectCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls the position of the point. The wedge extends from the center of the edge that is closest to the point. If the point is placed inside the rectangle, no wedge is drawn. Note that part of the shape extends past the geometry box of the shape.

Geometric properties:

Path	m3600,qx,3600l0@8@12@24,0@9,,18000qy3600,21600l@6,21600@15@27@7,21600,18000,21600qx21600,18000l21600@9@18@30,21600@8,21600,3600qy18000,l@7,0@21@33@6,xe
Guide Formulas	sum 10800 0 #0 sum 10800 0 #1 sum #0 0 #1 sum @0 @1 0 sum 21600 0 #0 sum 21600 0 #1 if @0 3600 12600 if @0 9000 18000 if @1 3600 12600 if @1 9000 18000 if @2 0 #0 if @3 @10 0 if #0 0 @11 if @2 @6 #0 if @3 @6 @13 if @5 @6 @14 if @2 #0 21600 if @3 21600 @16 if @4 21600 @17 if @2 #0 @6 if @3 @19 @6 if #1 @6 @20 if @2 @8 #1 if @3 @22 @8 if #0 @8 @23 if @2 21600 #1 if @3 21600 @25 if @5 21600 @26 if @2 #1 @8 if @3 @8 @28 if @4 @8 @29 if @2 #1 0 if @3 @31 0 if #1 0 @32 val #0 val #1
Adjustment Values	1350,25920
Connector Locations	10800,0;0,10800;10800,21600;21600,10800;@34,@35
Text Box Rectangle	791,791,20809,20809
Handles	position="#0,#1



Oval Callout
Internal Name: WedgeEllipseCallout
Shaped Concentric Fill: No.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: One adjust handle controls position of the point. The wedge extends from the center of the ellipse. If the point is placed inside the ellipse, no wedge is drawn. Note that part of the shape extends past the geometry box of the shape.

Geometric properties:

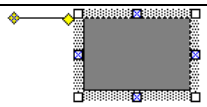
Path	wr,,21600,21600@15@16@17@18l@21@22xe
Guide Formulas	val #0 val #1 sum 10800 0 #0 sum 10800 0 #1 atan2 @2 @3 sumangle @4 11 0 sumangle @4 0 11 cos 10800 @4 sin 10800 @4 cos 10800 @5 sin 10800 @5 cos 10800 @6 sin 10800 @6 sum 10800 0 @7 sum 10800 0 @8 sum 10800 0 @9 sum 10800 0 @10 sum 10800 0 @11 sum 10800 0 @12 mod @2 @3 0 sum @19 0 10800 if @20 #0 @13 if @20 #1 @14
Adjustment Values	1350,25920
Connector Locations	10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437,18437;21600,10800;18437,3163;@21,@22
Text Box Rectangle	3163,3163,18437,18437
Handles	position="#0,#1

Cloud Callout
Internal Name: CloudCallout
Shaped Concentric Fill: No.
Joins: Rounded.
Endcaps: Rounded.
Adjustments: One adjust handle controls position of the point. The three bubbles extend from the perimeter of the cloud in-line with the center of the cloud. Note that part of the shape extends past the geometry box of the shape.

Geometric properties:

Path	ar,7165,4345,13110,1950,7185,1080,12690,475,11732,4835,17650,1080,12690,2910,17640,2387,9757,10107,20300,2910,17640,8235,19545,7660,12382,14412,21597,8235,19545,14280,18330,12910,11080,18695,18947,14280,18330,18690,15045,14822,5862,21597,15082,18690,15045,20895,7665,15772,2592,21105,9865,20895,7665,19140,2715,14330,,19187,6595,19140,2715,14910,1170,10992,,15357,5945,14910,1170,11250,1665,6692,650,12025,7917,11250,1665,7005,2580,1912,1972,8665,11162,7005,2580,1950,7185xear,7165,4345,13110,1080,12690,2340,13080nfear475,11732,4835,17650,2910,17640,3465,17445nfear7660,12382,14412,21597,7905,18675,8235,19545nfear7660,12382,14412,21597,14280,18330,14400,17370nfear12910,11080,18695,18947,18690,15045,17070,11475nfear15772,2592,21105,9865,20175,9015,20895,7665nfear14330,,19187,6595,19200,3345,19140,2715nfear14330,,19187,6595,14910,1170,14550,1980nfear10992,,15357,5945,11250,1665,11040,2340nfear1912,1972,8665,11162,7650,3270,7005,2580nfear1912,1972,8665,11162,1950,7185,2070,7890nfem@23@37qx@35@24@23@36@34@24@23@37xem@16@33qx@31@17@16@32@30@17@16@33xem@38@29qx@27@39@38@28@26@39@38@29xe
Guide Formulas	sum #0 0 10800 sum #1 0 10800 cosatan2 10800 @0 @1

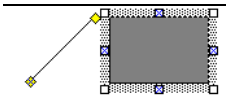
	sinatan2 10800 @0 @1 sum @2 10800 0 sum @3 10800 0 sum @4 0 #0 sum @5 0 #1 mod @6 @7 0 prod 600 11 1 sum @8 0 @9 prod @10 1 3 prod 600 3 1 sum @11 @12 0 prod @13 @6 @8 prod @13 @7 @8 sum @14 #0 0 sum @15 #1 0 prod 600 8 1 prod @11 2 1 sum @18 @19 0 prod @20 @6 @8 prod @20 @7 @8 sum @21 #0 0 sum @22 #1 0 prod 600 2 1 sum #0 600 0 sum #0 0 600 sum #1 600 0 sum #1 0 600 sum @16 @25 0 sum @16 0 @25 sum @17 @25 0 sum @17 0 @25 sum @23 @12 0 sum @23 0 @12 sum @24 @12 0 sum @24 0 @12 val #0 val #1
Adjustment Values	1350,25920
Connector Locations	67,10800;10800,21577;21582,10800;10800,1235;@38,@39
Text Box Rectangle	2977,3262,17087,17337
Handles	position="#0,#1



Line Callout 1
Internal Name: BorderCallout90
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line are adjustable.

Geometric properties:

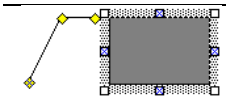
Path	m@0@1l@2@3nfem,l21600,r,21600l,21600xe
Guide Formulas	val #0 val #1 val #2 val #3
Adjustment Values	-1800,24300,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3



Line Callout 2
Internal Name: BorderCallout1
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line are adjustable.

Geometric properties:

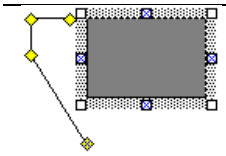
Path	m@0@1l@2@3nfem,l21600,r,21600l,21600xe
Guide Formulas	val #0 val #1 val #2 val #3
Adjustment Values	-8280,24300,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3



Line Callout 3
Internal Name: BorderCallout2
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line and the joint are adjustable.

Geometric properties:

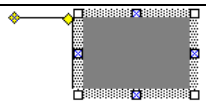
Path	m@0@1l@2@3@4@5nfem,l21600,r,21600l,21600xe
Guide Formulas	val #0 val #1 val #2 val #3 val #4 val #5
Adjustment Values	-10080,24300,-3600,4050,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3 position="#4,#5



Line Callout 4
Internal Name: BorderCallout3
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line and both joints are adjustable.

Geometric properties:

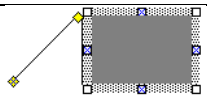
Path	m@0@1l@2@3@4@5@6@7nfem,l21600,r,21600l,21600xe
Guide Formulas	val #0 val #1 val #2 val #3 val #4 val #5 val #6 val #7
Adjustment Values	23400,24400,25200,21600,25200,4050,23400,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3 position="#4,#5 position="#6,#7



Line Callout 1 (Accent Bar)
Internal Name: AccentCallout90
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line are adjustable.

Geometric properties:

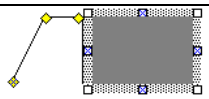
Path	m@0@1l@2@3nfem,l21600,r,21600l,21600nsxe
Guide Formulas	val #0 val #1 val #2 val #3
Adjustment Values	-1800,24300,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3



Line Callout 2 (Accent Bar)
Internal Name: AccentCallout1
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The positions of both ends of the line are adjustable.

Geometric properties:

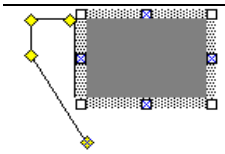
Path	m@0@1l@2@3nfem@2,l@2,21600nfem,l21600,r,21600l,21600nsxe
Guide Formulas	val #0 val #1 val #2 val #3
Adjustment Values	-8280,24300,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3



Line Callout 3 (Accent Bar)
Internal Name: AccentCallout2
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line and the joint are adjustable.

Geometric properties:

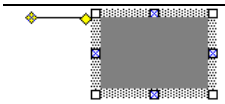
Path	m@0@1l@2@3@4@5nfem@4,l@4,21600nfem,l21600,r,21600l,21600nsxe
Guide Formulas	val #0 val #1 val #2 val #3 val #4 val #5
Adjustment Values	-10080,24300,-3600,4050,-1800,4050
Connector Locations	0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3 position="#4,#5



Line Callout 4 (Accent Bar)
Internal Name: AccentCallout3
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line and both joints are adjustable.

Geometric properties:

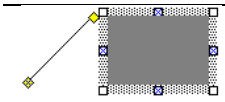
Path	m@0@1l@2@3@4@5@6@7nfem@6,l@6,21600nfem,l21600,r,21600l,21600nsxe
Guide Formulas	val #0 val #1 val #2 val #3 val #4 val #5 val #6 val #7
Adjustment Values	23400,24400,25200,21600,25200,4050,23400,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3 position="#4,#5 position="#6,#7



Line Callout 1 (No Border)
Internal Name: Callout90
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line are adjustable.

Geometric properties:

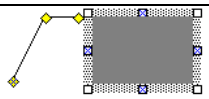
Path	m@0@1l@2@3nfem,l21600,r,21600l,21600nsxe
Guide Formulas	val #0 val #1 val #2 val #3
Adjustment Values	-1800,24300,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3



Line Callout 2 (No Border)
Internal Name: Callout1
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The positions of both ends of the line are adjustable.

Geometric properties:

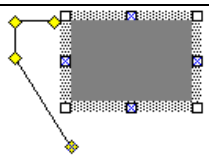
Path	m@0@1l@2@3nfem,l21600,r,21600l,21600nsxe
Guide Formulas	val #0 val #1 val #2 val #3
Adjustment Values	8280,24300,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3



Line Callout 3 (No Border)
Internal Name: Callout2
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line and the joint are adjustable.

Geometric properties:

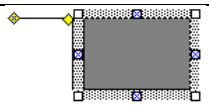
Path	m@0@1l@2@3@4@5nfem,l21600,r,21600l,21600nsxe
Guide Formulas	val #0 val #1 val #2 val #3 val #4 val #5
Adjustment Values	-10080,24300,-3600,4050,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3 position="#4,#5



Line Callout 4 (No Border)
Internal Name: Callout3
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line and both joints are adjustable.

Geometric properties:

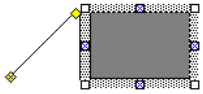
Path	m@0@1l@2@3@4@5@6@7nfem,l21600,r,21600l,21600nsxe
Guide Formulas	val #0 val #1 val #2 val #3 val #4 val #5 val #6 val #7
Adjustment Values	23400,24400,25200,21600,25200,4050,23400,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3 position="#4,#5 position="#6,#7



Line Callout 1 (Border and Accent Bar)
Internal Name: AccentBorderCallout90
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line are adjustable.

Geometric properties:

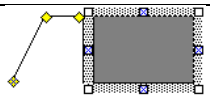
Path	m@0@1l@2@3nfem,l21600,r,21600l,21600xe
Guide Formulas	val #0 val #1 val #2 val #3
Adjustment Values	-1800,24300,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3



Line Callout 2 (Border and Accent Bar)
Internal Name: AccentBorderCallout1
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The positions of both ends of the line are adjustable.

Geometric properties:

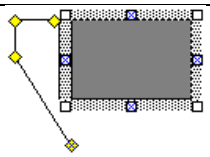
Path	m@0@1l@2@3nfem@2,l@2,21600nfem,l21600,r,21600l,21600xe
Guide Formulas	val #0 val #1 val #2 val #3
Adjustment Values	-8280,24300,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3



Line Callout 3 (Border and Accent Bar)
Internal Name: AccentBorder Callout2
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line and the joint are adjustable.

Geometric properties:

Path	m@0@1l@2@3@4@5nfem@4,l@4,21600nfem,l21600,r,21600l,21600xe
Guide Formulas	val #0 val #1 val #2 val #3 val #4 val #5
Adjustment Values	-10080,24300,-3600,4050,-1800,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3 position="#4,#5

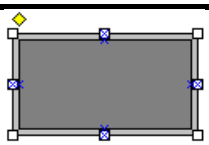


Line Callout 4 (Border and Accent Bar)
Internal Name: AccentBorderCallout3
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: The position of both ends of the line and both joints are adjustable.

Geometric properties:

Path	m@0@1l@2@3@4@5@6@7nfem@6,l@6,21600nfem,l21600,r,21600l,21600xe
Guide Formulas	val #0 val #1 val #2 val #3 val #4 val #5 val #6 val #7
Adjustment Values	23400,24400,25200,21600,25200,4050,23400,4050
Connector Locations	@0,@1;10800,0;10800,21600;0,10800;21600,10800
Handles	position="#0,#1 position="#2,#3 position="#4,#5 position="#6,#7

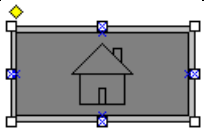
Action Buttons



Action Button: Custom
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0e
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800

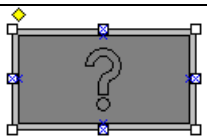


Action Button: Home
Internal Name: ActionButtonHome
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@3@9nfl@1 1@4@28@4@28@10@33@10@33@4@12@4@32@26@32@24@31@24 @31@25xem@31@25nfl@32@26em@28@4nfl@33@4em@29@10nfl@29 @27@30@27@30@10e
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @10 0 @9 prod @13 1 16 prod @13 1 8 prod @13 3 16 prod @13 5 16 prod @13 7 16 prod @13 9 16

	prod @13 11 16 prod @13 3 4 prod @13 13 16 prod @13 7 8 sum @9 @14 0 sum @9 @16 0 sum @9 @17 0 sum @9 @21 0 sum @11 @15 0 sum @11 @18 0 sum @11 @19 0 sum @11 @20 0 sum @11 @22 0 sum @11 @23 0 sum @3 @5 0 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0 sum @24 @5 0 sum @25 @5 0 sum @26 @5 0 sum @27 @5 0 sum @28 @5 0 sum @29 @5 0 sum @30 @5 0 sum @31 @5 0 sum @32 @5 0 sum @33 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800

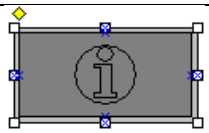


Action Button: Help
Internal Name: ActionButtonHelp
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@33@27nfq y@3@9@40@27@39@4@37@29l@37@30@36@30@36@29qy@37@28 @39@27@3@26@34@27xem@3@31nfqx@35@32@3@10@38@32@3@ 31xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @10 0 @9 prod @13 1 7

	prod @13 3 14 prod @13 2 7 prod @13 5 14 prod @13 11 28 prod @13 3 7 prod @13 4 7 prod @13 17 28 prod @13 9 14 prod @13 21 28 prod @13 11 14 prod @13 25 28 sum @9 @14 0 sum @9 @16 0 sum @9 @18 0 sum @9 @21 0 sum @9 @23 0 sum @9 @24 0 sum @9 @25 0 sum @11 @15 0 sum @11 @17 0 sum @11 @18 0 sum @11 @19 0 sum @11 @20 0 sum @11 @21 0 sum @11 @22 0 sum @11 @24 0 sum @3 @5 0 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0 sum @26 @5 0 sum @27 @5 0 sum @28 @5 0 sum @29 @5 0 sum @30 @5 0 sum @31 @5 0 sum @32 @5 0 sum @33 @5 0 sum @34 @5 0 sum @35 @5 0 sum @36 @5 0 sum @37 @5 0 sum @38 @5 0 sum @39 @5 0 sum @40 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,1080

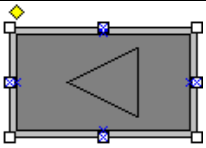


Action Button: Information
Internal Name: ActionButtonInformation
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@3@9nfqx @11@4@3@10@12@4@3@9xem@3@25nfqx@33@26@3@27@36@26@

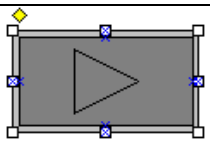
	3@25xem@32@28nfl@32@29@34@29@34@30@32@30@32@31@37@31@37@30@35@30@35@28xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @10 0 @9 prod @13 1 32 prod @13 5 32 prod @13 9 32 prod @13 5 16 prod @13 3 8 prod @13 13 32 prod @13 19 32 prod @13 5 8 prod @13 11 16 prod @13 13 16 prod @13 7 8 sum @9 @14 0 sum @9 @15 0 sum @9 @16 0 sum @9 @17 0 sum @9 @18 0 sum @9 @23 0 sum @9 @24 0 sum @11 @17 0 sum @11 @18 0 sum @11 @19 0 sum @11 @20 0 sum @11 @21 0 sum @11 @22 0 sum @3 @5 0 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0 sum @25 @5 0 sum @26 @5 0 sum @27 @5 0 sum @28 @5 0 sum @29 @5 0 sum @30 @5 0 sum @31 @5 0 sum @32 @5 0 sum @33 @5 0 sum @34 @5 0 sum @35 @5 0 sum @36 @5 0 sum @37 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800



Action Button: Back/Previous
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

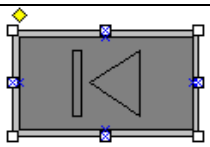
Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@12@9nfl@ 11@4@12@10xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800



Action Button: Forward/Next
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

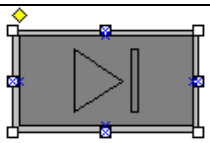
Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@11@9nfl@ 12@4@11@10xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800



Action Button: Beginning
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

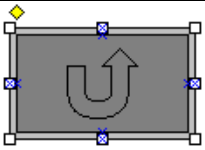
Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@12@9l@1 7@4@12@10xem@11@9l@16@9@16@10@11@10xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @10 0 @9 prod @13 1 8 prod @13 1 4 sum @11 @14 0 sum @11 @15 0 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0 sum @16 @5 0 sum @17 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800



Action Button: End
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

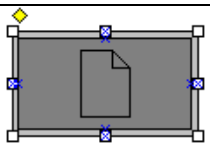
Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@11@9l@1 6@4@11@10xem@17@9l@12@9@12@10@17@10xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @10 0 @9 prod @13 3 4 prod @13 7 8 sum @11 @14 0 sum @11 @15 0 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0 sum @16 @5 0 sum @17 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800



Action Button: Return
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

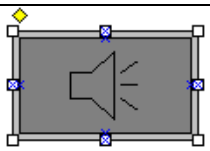
Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@12@21nfl @23@9@3@21@24@21@24@20qy@3@19l@25@19qx@26@20l@26@2 1@11@21@11@20qy@25@10l@3@10qx@22@20l@22@21xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @10 0 @9 prod @13 7 8 prod @13 3 4 prod @13 5 8 prod @13 3 8 prod @13 1 4 sum @9 @15 0 sum @9 @16 0 sum @9 @18 0 sum @11 @14 0 sum @11 @15 0 sum @11 @16 0 sum @11 @17 0 sum @11 @18 0 sum @3 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0 sum @19 @5 0 sum @20 @5 0 sum @21 @5 0 sum @22 @5 0 sum @23 @5 0 sum @24 @5 0 sum @25 @5 0 sum @26 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800



Action Button: Document
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

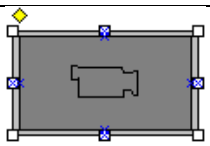
Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@12@9nfl@ 12@10@13@10@13@14@15@9xem@15@9nfl@15@14@13@14e
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 prod #0 3 4 sum @3 @11 6075 sum @3 6075 @11 sum @4 @5 4050 sum @13 @5 4050 sum @9 @5 0 sum @10 @5 0 sum @12 @5 0 sum @13 @5 0 sum @14 @5 0 sum @15 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch=“true” xrange="0,5400
Limo	10800,10800



Action Button: Sound
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@11@21nfl @11@22@24@22@25@10@25@9@24@21xem@26@21nfl@12@20em@ 26@4nfl@12@4em@26@22nfl@12@23e
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @10 0 @9 prod @13 1 8 prod @13 5 16 prod @13 5 8 prod @13 11 16 prod @13 3 4 prod @13 7 8 sum @9 @14 0 sum @9 @15 0 sum @9 @17 0 sum @9 @19 0 sum @11 @15 0 sum @11 @16 0 sum @11 @18 0 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0 sum @20 @5 0 sum @21 @5 0 sum @22 @5 0 sum @23 @5 0 sum @24 @5 0 sum @25 @5 0 sum @26 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800



Action Button: Movie
Internal Name: ActionButtonBlank
Shaped Concentric Fill: Yes.
Joins: Mitered.
Endcaps: Rounded.
Adjustments: A top or left perimeter adjust handle controls the amount of bevel. It limo-stretches both vertically and horizontally at the midpoint, so that the amount of bevel is the same horizontally and vertically. Also, the

image on the button face stays in proportion and centered. The adjust handle switches between the top and left sides depending on which dimension is smaller. The adjust handle can extend half the distance of the smaller dimension. When a hyperlink or Action Setting (PowerPoint) is attached to this shape, it will change to the second form when depressed.

Geometric properties:

Path	m,l,21600r21600,l21600,xem@0@0nfl@0@2@1@2@1@0xem,nfl@0@0em ,21600nfl@0@2em21600,21600nfl@1@2em21600,nfl@1@0em@11@39nfl @11@44@31@44@32@43@33@43@33@47@35@47@35@45@36@45@ 38@46@12@46@12@41@38@41@37@42@35@42@35@41@34@40@3 2@40@31@39xe
Guide Formulas	val #0 sum width 0 #0 sum height 0 #0 prod width 1 2 prod height 1 2 prod #0 1 2 prod #0 3 2 sum @1 @5 0 sum @2 @5 0 sum @0 @4 8100 sum @2 8100 @4 sum @0 @3 8100 sum @1 8100 @3 sum @10 0 @9 prod @13 1455 21600 prod @13 1905 21600 prod @13 2325 21600 prod @13 16155 21600 prod @13 17010 21600 prod @13 19335 21600 prod @13 19725 21600 prod @13 20595 21600 prod @13 5280 21600 prod @13 5730 21600 prod @13 6630 21600 prod @13 7492 21600 prod @13 9067 21600 prod @13 9555 21600 prod @13 13342 21600 prod @13 14580 21600 prod @13 15592 21600 sum @11 @14 0 sum @11 @15 0 sum @11 @16 0 sum @11 @17 0 sum @11 @18 0 sum @11 @19 0 sum @11 @20 0 sum @11 @21 0 sum @9 @22 0 sum @9 @23 0 sum @9 @24 0 sum @9 @25 0 sum @9 @26 0 sum @9 @27 0 sum @9 @28 0 sum @9 @29 0 sum @9 @30 0 sum @9 @31 0 sum @4 @5 0 sum @9 @5 0 sum @10 @5 0 sum @11 @5 0 sum @12 @5 0 sum @31 @5 0 sum @32 @5 0 sum @33 @5 0 sum @34 @5 0

	sum @35 @5 0 sum @36 @5 0 sum @37 @5 0 sum @38 @5 0 sum @39 @5 0 sum @40 @5 0 sum @41 @5 0 sum @42 @5 0 sum @43 @5 0 sum @44 @5 0 sum @45 @5 0 sum @46 @5 0 sum @47 @5 0
Adjustment Values	1350
Connector Locations	0,@4;@0,@4;@3,21600;@3,@2;21600,@4;@1,@4;@3,0;@3,@0
Text Box Rectangle	@0,@0,@1,@2
Handles	position="#0,topLeft" switch="true" xrange="0,5400
Limo	10800,10800

Appendix E: Animation Records

This appendix documents animation information records that can exist at the slide level of a document. For example, PowerPoint XP and later uses these records to store information about animations in presentations.

Many of these records represent the same type of information as an XML type, element, or attribute in the Office Open XML format. The publically available documentation for this format provides additional context and details for how animations are represented. Where appropriate, record descriptions contain references to their corresponding representation in the Office Open XML format.

F123 msofbtTimeNodeContainer

Same format as msofbtExtTimeNodeContainer.

F124 msofbtTimeConditionList

List of msofbtTimeCondition records.

The instance field of each condition record contains the condition type, with the values defined by the following enum:

```
enum ConditionType
{
   _NA,     // obsolete identifier; not used
   Begin,
   End,
   Next,
   Previous,
   EndSync,
   ConditionType_MaxEnumIDs
};
```

ConditionType indicates a type of condition that is equivalent to a condition type represented in the Office Open XML format:

ConditionType	Office Open XML equivalent
Begin	A “cond” in stCondLst
End	A “cond” in endCondLst
Next	A “cond” in nextCondLst
Previous	A “cond” in prevCondLst
EndSync	“endSync” element

F125 msofbtTimeConditionContainer

Record	Type

Base record	msofbtTimeCondition (FTC)
Visual element (optional)	msofbtClientVisualElement if triggerType == totVisualElement

F126 msofbtTimeModifierList

List of msofbtTimeModifier records.

F127 msofbtTimeNode (FTN)

Represents the same information as the CT_TLCommonTimeNodeData type in the Office Open XML format.

```
// FTN - File Time Node
struct FTN
{
   ULONG  masterID;           // OBSOLETE: id of the master time node in the timing tree
for a "subordinate" time node.
   ULONG  restart;              // TLTimeNodeRestartType
   ULONG  type;                 // TLTimeNodeType
   ULONG  fill;                 // TLTimeNodeFillType
   ULONG  syncBehavior;         // TLTimeNodeSyncType
   BYTE   fSyncMaster;          // OBSOLETE: set to zero
   LONG   duration;             // TIME, in milliseconds
   ULONG  propertiesSet;       // bit flag of which properties are used/set, as defined
below
};

enum TLTimeNodeType
{
   TLTimeNodeTypeParallel = 0,
   TLTimeNodeTypeSequential,
   TLTimeNodeTypeExclusive,
   TLTimeNodeTypeBehaviorType,
   TLTimeNodeTypeMediaType,
   TLTimeNodeType_MaxEnumIDs
};

enum TLTimeNodeRestartType
{
   TLR_NoRestartType = 0,
   TLR_AlwaysRestart = 1,
   TLR_RestartWhenOff = 2,
   TLR_NeverRestart = 3,
   TLTimeNodeRestartType_MaxEnumIDs = 4
};

enum TLTimeNodeFillType
{
   TLF_NoFillType = 0,
   TLF_FillRemove = 1,
   TLF_FillFreeze = 2,
```

```
   TLF_FillHold = 3,
   TLF_FillTransition = 4,
   TLTimeNodeFillType_MaxEnumIDs = 5
};

enum TLTimeNodeSyncType
{
   TLS_NoSyncType = 0,
   TLS_CanSlipSyncType = 1,
   TLS_LockedSyncType = 2,
   TLTimeNodeSyncType_MaxEnumIDs = 3
};

// property bit flag for propertiesSet
const int fillProperty         = (1 << 0);
const int restartProperty      = (1 << 1);
const int syncBehaviorProperty = (1 << 2);
const int groupingTypeProperty = (1 << 3);
const int durationProperty     = (1 << 4);
```

F128 msofbtTimeCondition (FTC)

Represents the same information as the CT_TLTimeCondition type in the Office Open XML format.

The instance field of the record contains the condition type, the same as described by the "ConditionType" for record F124.

```
// FTC - File Time Condition
struct FTC
{
   ULONG  triggerType;          // TriggerObjectType
   ULONG  event;                // TriggerEventType
   ULONG  id;                   // (1) RuntimeNodeReferenceType if triggerType ==
totRuntimeNodeRef; (2) index of timenode in tree where index is computed by depth-first
traversal of the tree
   LONG   delay;                // TIME, in milliseconds
};

{
   totNone,
   totVisualElement,
   totTimeNode,
   totRuntimeNodeRef,
   TriggerObjectType_MaxEnumIDs
};

enum TriggerEventType
{
   tetNone = 0,
   tetOnBegin,
   tetOnEnd,
   tetBegin,
   tetEnd,
```

```
   tetOnClick,
   tetOnDoubleClick,
   tetOnMouseOver,
   tetOnMouseOut,
   tetOnNext,             // PPT-specific
   tetOnPrev,             // PPT-specific
   tetOnStopAudio,
   TriggerEventType_MaxEnumIDs
};

enum RuntimeNodeReferenceType
{
   rnrtWithFirstChild,
   rnrtWithLastChild,
   rnrtAllChildren,
   RuntimeNodeReferenceType_MaxEnumIDs
};
```

F129 msofbtTimeModifier (FTM)

Time modifiers for a time node, used in a msofbtExtTimeNodeContainer.

The instance field of each record tells you the Modifier type:

```
   enum Type
   {
      RepeatCount = 0,  // The number of times to repeat
      RepeatDur,        // Duration of repeats
      Speed,            // Fraction indicating how much faster
                        // or slower to go than the normal duration.
      Accelerate,       // Fraction between 0 and 1 indicating
                        // portion of the duration over which to accelerate.
      Decelerate,       // Fraction between 0 and 1 indicating portion of the
                        // duration over which to decelerate.
      AutoReverse,      // The node should play forward then reverse.
      Type_MaxEnumIDs
   };


// FTM - File Time Modifier
struct FTM
{
   ULONG  type;                 // Type, as described above
   ULONG  value;                // a float, written out as a ULONG
};
```

F12A msofbtTimeBehaviorContainer

Represents the same information as the CT_TLCommonBehaviorData type in the Office Open XML format.

Record	Type

Base record	msofbtTimeBehavior FTB
Optional attribute names	msofbtTimeVariantList (should all be strings)
optional additional properties	msofbtTimePropertyList
optional "FROM"	msofbtTimeVariant
optional "TO"	msofbtTimeVariant
optional "BY"	msofbtTimeVariant
Optional target element	msofbtClientVisualElement

F12B msofbtTimeAnimateBehaviorContainer

Represents the same information as the CT_TLAnimateBehavior type in the Office Open XML format.

Record	Type
Base record	msofbtTimeAnimateBehavior (FTBA)
Optional animation value list	msofbtTimeAnimationValueList
"BY"	msofbtTimeVariant (instance 1)
"FROM"	msofbtTimeVariant (instance 2)
"TO"	msofbtTimeVariant (instance 3)
Base time behavior record	msofbtTimeBehaviorContainer (FTB)

F12C msofbtTimeColorBehaviorContainer

Represents the same information as the CT_TLAnimateColorBehavior type in the Office Open XML format.

Base record	msofbtTimeColorBehavior (FTBC)
Base time behavior record	msofbtTimeBehaviorContainer (FTB)

F12D msofbtTimeEffectBehaviorContainer

Represents the same information as the CT_TLAnimateEffectBehavior type in the Office Open XML format.

Base record	msofbtTimeColorBehavior (FTBE)
Type	msofbtTimeVariant (instance 1)
Progress	msofbtTimeVariant (instance 2)
Runtime context	msofbtTimeVariant (instance 3)
Base time behavior record	msofbtTimeBehaviorContainer (FTB)

F12E msofbtTimeMotionBehaviorContainer

Represents the same information as the CT_TLAnimateMotionBehavior type in the Office Open XML format.

Base record	msofbtTimeColorBehavior (FTBM)
Path	msofbtTimeVariant (instance 1)
Rotation	msofbtTimeVariant (instance 2)
Base time behavior record	msofbtTimeBehaviorContainer (FTB)

F12F msofbtTimeRotationBehaviorContainer

Represents the same information as the CT_TLAnimateRotationBehavior type in the Office Open XML format.

Base record	msofbtTimeRotationBehavior (FTBR)
Base time behavior record	msofbtTimeBehaviorContainer (FTB)

F130 msofbtTimeScaleBehaviorContainer

Represents the same information as the CT_TLAnimateScaleBehavior type in the Office Open XML format.

Base record	msofbtTimeScaleBehavior (FTBS)
Base time behavior record	msofbtTimeBehaviorContainer (FTB)

F131 msofbtTimeSetBehaviorContainer

Represents the same information as the CT_TLSetBehavior type in the Office Open XML format.

Base record	msofbtTimeSetBehavior (FTBSet)
"To"	msofbtTimeVariant (instance 1)
Base time behavior record	msofbtTimeBehaviorContainer (FTB)

F132 msofbtTimeCommandBehaviorContainer

Represents the same information as the CT_TLCommandBehavior type in the Office Open XML format.

Base record	msofbtTimeCommandBehavior (FTBCom)
Command	msofbtTimeVariant (instance 1)
Base time behavior record	msofbtTimeBehaviorContainer (FTB)

F133 msofbtTimeBehavior (FTB)

Represents part of the same information as the CT_TLBehavior type in the Office Open XML format.

```
// FTB - File Time Behavior
struct FTB
{
   ULONG       propertiesUsed;  // bit flag of properties set, as defined below by the
PropertyUsedFlag
   UINT        tbaddAdditive;   // Additive
   UINT        tbaccAccumulate; // Accumulate
   UINT        tbttTransformType; // TransformTYpe
};

enum PropertyUsedFlag
{
   PUF_NONE             = 0,
   PUF_Additive         = 1 << 0,
   PUF_Accumulate       = 1 << 1,
   PUF_AttributeNames   = 1 << 2,
   PUF_TransformType    = 1 << 3,
   PUF_FromFormula      = 1 << 4,
   PUF_ToFormula        = 1 << 5,
   PUF_ByFormula        = 1 << 6
};

enum Additive
```

```
{
   BaseAdditive,
   SumAdditive,
   ReplaceAdditive,
   MultiplyAdditive,
   NoAdditive,
   Additive_MaxEnumIDs
};

enum Accumulate
{
   NoAccumulate,
   AlwaysAccumulate,
   Accumulate_MaxEnumIDs
};

enum TransformType
{
   PropertyTransformType,
   ImageTransformType,
   TransformType_MaxEnumIDs
};
```

F134 msofbtTimeAnimateBehavior (FTBA)

Represents part of the same information as the CT_TLAnimateBehavior type in the Office Open XML format.

```
// FTBA - File Time Behavior
struct FTBA
{
   UINT        tabcmCalcMode;   // CalcMode
   ULONG       propertiesUsed;  // AnimatePropertyUsedFlag
   UINT        tabvtValueType;  // ValueType
};

enum CalcMode
{
   DiscreteMode,
   LinearMode,
   FormulaMode,
   CalcMode_MaxEnumIDs
};

enum AnimatePropertyUsedFlag
{
   APUF_NONE                = 0,
   APUF_By                  = 1 << 0,
   APUF_From                = 1 << 1,
   APUF_To                  = 1 << 2,
   APUF_CalcMode            = 1 << 3,
```

```
   APUF_AnimationValues     = 1 << 4,
   APUF_ValueType           = 1 << 5
};

enum ValueType
{
   StringType,
   NumberType,
   ColorType,
   ValueType_MaxEnumIDs
};
```

F135 msofbtTimeColorBehavior (FTBC)

Represents part of the same information as the CT_TLAnimateColorBehavior type in the Office Open XML format.

```
// FTBC - File Time Behavior
struct FTBC
{
   ULONG       propertiesUsed;  // AnimateColorPropertyUsedFlag
   UINT        tabcmBy;         // ColorModel
   UINT      nColor1By;      // range is -255 to 255; could be R or H depending on color
model specified by tabcmBy
   UINT        nColor2By;       // range is -255 to 255; G or S depending on tabcmBy
   UINT        nColor3By;       // range is -255 to 255; B or L depending on tabcmBy
   UINT        tabcmFrom;       // ColorModel
   UINT        nColor1From;     // range is 0 to 255
   UINT        nColor2From;     // range is 0 to 255
   UINT        nColor3From;     // range is 0 to 255
   UINT        tabcmTo;         // ColorModel
   UINT        nColor1To;       // range is 0 to 255
   UINT        nColor2To;       // range is 0 to 255
   UINT        nColor3To;       // range is 0 to 255
};

enum AnimateColorPropertyUsedFlag
{
   ACPUF_NONE                = 0,
   ACPUF_By                  = 1 << 0,
   ACPUF_From                = 1 << 1,
   ACPUF_To                  = 1 << 2,
   ACPUF_ColorSpace          = 1 << 3,
   ACPUF_Direction           = 1 << 4
};

enum ColorModel
{
   RGBColorModel   = 0,
   HSLColorModel   = 1,
   IndexColorModel = 2,
   ColorModel_MaxEnumIDs
};
```

F136 msofbtTimeEffectBehavior (FTBE)

Represents part of the same information as the CT_TLAnimateEffectBehavior type in the Office Open XML format.

```
// FTBE - File Time Behavior
struct FTBE
{
   ULONG       propertiesUsed;  // AnimateEffectPropertyUsedFlag
   UINT        taetTransition;  // Transition
};

enum AnimateEffectPropertyUsedFlag
{
   AEPUF_NONE                    = 0,
   AEPUF_Transition              = 1 << 0,
   AEPUF_Type                    = 1 << 1,
   AEPUF_Progress                = 1 << 2,
   AEPUF_RuntimeContext_Obsolete = 1 << 3 // Keep this for backward binary persistence
};

enum Transition
{
   TransitionIn,
   TransitionOut,
   TransitionNone,
   Transition_MaxEnumIDs
};
```

F137 msofbtTimeMotionBehavior (FTBM)

Represents part of the same information as the CT_TLAnimateMotionBehavior type in the Office Open XML format.

```
// FTBM - File Time Behavior Motion
struct FTBM
{
   ULONG       propertiesUsed;  // AnimateMotionPropertyUsedFlag
   ULONG       fXBy;            // a float, written out as a ULONG
   ULONG       fYBy;            // a float, written out as a ULONG
   ULONG       fXFrom;          // a float, written out as a ULONG
   ULONG       fYFrom;          // a float, written out as a ULONG
   ULONG       fXTo;            // a float, written out as a ULONG
   ULONG       fYTo;            // a float, written out as a ULONG
   UINT        tamboOrigin;     // Origin
};

enum AnimateMotionPropertyUsedFlag
```

```
{
   AMPUF_NONE                = 0,
   AMPUF_By                  = 1 << 0,
   AMPUF_From                = 1 << 1,
   AMPUF_To                  = 1 << 2,
   AMPUF_Origin              = 1 << 3,
   AMPUF_Path                = 1 << 4,
   AMPUF_Rotation            = 1 << 5,
   AMPUF_EditRotation        = 1 << 6,
   AMPUF_PointsTypes         = 1 << 7
};

enum Origin
{
   DefaultOrigin,
   ParentOrigin,
   LayoutOrigin,
   Origin_MaxEnumIDs
};
```

F138 msofbtTimeRotationBehavior (FTBR)

Represents part of the same information as the CT_TLAnimateRotationBehavior type in the Office Open XML format.

```
// FTBR - File Time Behavior Rotation
struct FTBR
{
   ULONG    propertiesUsed;     // AnimateRotationPropertyUsedFlag
   ULONG    fBy;                // a float, written out as a ULONG
   ULONG    fFrom;              // a float, written out as a ULONG
   ULONG    fTo;                // a float, written out as a ULONG
   UINT     tabrd;              // Direction
};

enum AnimateRotationPropertyUsedFlag
{
   ARPUF_NONE                = 0,
   ARPUF_By                  = 1 << 0,
   ARPUF_From                = 1 << 1,
   ARPUF_To                  = 1 << 2,
   ARPUF_Direction           = 1 << 3
};

enum Direction
{
   Clockwise,
   CounterClockwise,
   Direction_MaxEnumIDs
};
```

F139 msofbtTimeScaleBehavior (FTBS)

Represents part of the same information as the CT_TLAnimateScaleBehavior type in the Office Open XML format.

```
// FTBS - File Time Behavior Scale
struct FTBS
{
   ULONG    propertiesUsed;     // AnimateScalePropertyUsedFlag
   ULONG    fXBy;               // a float, written out as a ULONG
   ULONG    fYBy;               // a float, written out as a ULONG
   ULONG    fXFrom;             // a float, written out as a ULONG
   ULONG    fYFrom;             // a float, written out as a ULONG
   ULONG    fXTo;               // a float, written out as a ULONG
   ULONG    fYTo;               // a float, written out as a ULONG
   BYTE     fZoomContents;      // true or false
};

enum AnimateScalePropertyUsedFlag
{
   ASPUF_NONE                = 0,
   ASPUF_By                  = 1 << 0,
   ASPUF_From                = 1 << 1,
   ASPUF_To                  = 1 << 2,
   ASPUF_ZoomContents        = 1 << 3,
};
```

F13A msofbtTimeSetBehavior (FTBSet)

Represents part of the same information as the CT_TLSetBehavior type in the Office Open XML format.

```
// FTBSet - File Time Behavior Set
struct FTBSet
{
   ULONG    propertiesUsed;     // SetPropertyUsedFlag
   UINT     tabvtValueType;     // ValueType
};

enum SetPropertyUsedFlag
{
   SPUF_NONE               = 0,
   SPUF_To                 = 1 << 0,
   SPUF_ValueType          = 1 << 1
};

enum ValueType
{
   StringType,
   NumberType,
   ColorType,
   ValueType_MaxEnumIDs
};
```

F13B msofbtTimeCommandBehavior (FTBCom)

Represents part of the same information as the CT_TLCommandBehavior type in the Office Open XML format.

```
// FTBCom - File Time Behavior Command
struct FTBCom
{
   ULONG    propertiesUsed;     // CommandPropertyUsedFlag
   UINT     tcbtType;           // CommandType
};

enum CommandPropertyUsedFlag
{
   CPUF_NONE                = 0,
   CPUF_Type                = 1 << 0,
   CPUF_Command             = 1 << 1
};

enum CommandType
{
   EventType,
   CallType,
   OleVerbType,
   CommandType_MaxEnumIDs
};
```

F13C msofbtClientVisualElement

msofbtClientVisualElement is a container for either (1) PSR_VisualPageAtom or (2) PSR_VisualShapeAtom defined in serial.h. The PSR_VisualXXX records are documented in the PowerPoint file format documentation.

F13D msofbtTimePropertyList

A list of TimeVariant objects, defined by msofbtTimeVariant records. The instance field for each

TimeVariant record gives you the property ID for the property.

```
enum TLTimePropertyID
{
   tpidUnknown                  = 0,
   tpidID                       = 1, // string
   tpidDisplay                  = 2,
   tpidMasterPos                = 5,
   tpidSubNodeType              = 6,
   tpidParagraphLevel           = 7,
   tpidGraphLevel               = 8,
```

```
   tpidEffectID                = 9,
   tpidEffectDir               = 10,
   tpidEffectType              = 11,
   tpidAfterEffect             = 13,
   tpidDiagramLevel            = 14,
   tpidSlideCount              = 15, // integer
   tpidTimeFilter              = 16, // string
   tpidEventFilter             = 17, // string
   tpidHideWhenStopped         = 18, // boolean
   tpidGroupID                 = 19,
   tpidPPTType                 = 20,
   tpidPlaceholderNode         = 21,
   tpidMediaVolume             = 22,
   tpidMediaMute               = 23,
   tpidXMLUnknownAttribs       = 24,
   tpidXMLAttribsUnknownValues = 25,
   tpidZoomToFullScreen        = 26,
   tpidShowControls            = 27,
   tpidDVDTitle                = 28,
   tpidDVDStartTime            = 29,
   tpidDVDEndTime              = 30
};
```

F13E msofbtTimeVariantList

A list of msofbtTimeVariant records. Each record is a string value that is an attribute name that indicates an attribute that the Behavior modifies. The possible attribute names are the same as those listed in the Office Open XML Format for the “attrName” element.

F13F msofbtTimeAnimationValueList

A list of msofbtTimeAnimationValue records. Represents the same information as the CT_TLTimeAnimateValueList in the Office Open XML Format.

F140 msofbtTimeIterateData FTID

Represents the same information as the CT_TLIterateData in the Office Open XML Format.

```
// FTID - File Time Iterate Data
struct FTID
{
   ULONG interval;             // a float, written out as a ULONG
   ULONG type;                 // IterationType, defined below
   ULONG direction;            // Direction, defined below
   ULONG intervaltype;         // IntervalType, defined below
   ULONG used;                 // PropertyUsed, defined below
};

enum IterationType
{
   AllAtOnce,
   ByWord,
```

```
   ByLetter,
   IterationType_MaxEnumIDs
};

enum IntervalType
{
   Seconds,
   Percentage,
   IntervalType_MaxEnumIDs
};

enum Direction
{
   Backwards,
   Forwards,
   Direction_MaxEnumIDs
};

enum PropertyUsed
{
   DirectionProperty      = 1 << 0,
   IterationTypeProperty  = 1 << 1,
   IntervalProperty       = 1 << 2,
   IntervalTypeProperty   = 1 << 3,
};
```

F141 msofbtTimeSequenceDataFTSD

Represents part of the same information as the CT_TLTimeNodeSequence in the Office Open XML Format.

```
// FTSD - File Time Sequence Data
struct FTSD
{
   ULONG concurrency;           // ConcurrencyType
   ULONG nextAction;            // NextActionType
   ULONG previousAction;        // PreviousActionType
   ULONG enableNext;            // EnableNextType
   ULONG used;                  // see "TLTimeSequenceData" flags below
};

enum ConcurrencyType
{
   Disabled,
   Enabled,
   ConcurrencyType_MaxEnumIDs
};

enum NextActionType
{
   NoNextActionType,
   Seek,
```

```
   NextActionType_MaxEnumIDs
};

enum PreviousActionType
{
   NoPreviousActionType,
   SkipTimed,
   PreviousActionType_MaxEnumIDs
};

enum EnableNextType
{
   End,
   Begin,
   EnableNextType_MaxEnumIDs
};

// TLTimeSequenceData flags:
const int SF_Concurrency    = (1 << 0);
const int SF_NextAction     = (1 << 1);
const int SF_PreviousAction = (1 << 2);
const int SF_EnableNext     = (1 << 3);
```

F142 msofbtTimeVariant

Represents the same information as the CT_TLAnimVariant in the Office Open XML Format.

Type	Byte – values defined by AnimVariantType, defined below
Value	bool, int, float, or string depending on the Type

```
enum Type
{
   None     = -1,
   Bool     = 0,
   Int      = 1,
   Float    = 2,
   String   = 3,
};
```

F143 msofbtTimeAnimationValue

Represents the same information as the CT_TLTimeAnimateValue in the Office Open XML Format.

Time, in milliseconds	LONG
Value	msofbtTimeVariant
Formula	string

F144 msofbtExtTimeNodeContainer

Represents the same information as a CT_TLTimeNodeSequence, CT_TLMediaNodeAudio, or CT_TLMediaNodeVideo type in the Office Open XML Format.

Base record	msofbtTimeNode (FTN)
Behavior (optional)	Any of the msofbtTimeXXXBehavior records
Media (optional)	msofbtClientVisualElement
Iterate data (optional)	msofbtTimeIterateData
Sequence data (optional)	msofbtTimeSequenceData
Begin condition list	msofbtTimeConditionList
End condition list	msofbtTimeCondition
End sync	msofbtTimeConditionList
Time modifiers	msofbtTimeModifierList
Subordinate time nodes (0-n occurences)	msofbtSubNodeContainer
Child time nodes (0-n occurences)	msofbtExtTimeNodeContainer

F145 msofbtSubNodeContainer

Same format as msofbtExtTimeNodeContainer. This represents the same information as the “subTnLst” element of the CT_TLCommonTimeNodeData type in the Office Open XML Format.

Appendix F: Shape XML Data

The 2007 Microsoft Office System for Windows introduced a new XML-based file format. This format is fully documented in the publicly available Office Open XML documentation. While this is the default format for documents saved by Office 2007, Office 2007 also provides the capability to save files to the binary file format used in previous versions and described by this document.

Some new shape properties and property groups were added to the binary file format in Office 2007, and those properties are documented above. In addition, some XML information from the Office Open XML format is embedded in binary files. This is done to preserve data in the file, so that a file saved by Office 2007 to the binary format and later reopened in Office 2007 will retain document features available only in Office 2007 and later.

The shape property metroBlob (PID 937) stores a package in Office Open XML format. As described in the Office Open XML format, this data is equivalent to a zip file. In addition to storing the shape's Office Open XML representation, with root element "sp", the package may contain a second file named downrev.xml.

The downrev.xml file stores additional information about how the shape was converted from its native XML representation to the older binary representation. This information allows Office 2007 to determine what properties of the shape were changed when the file was saved by a previous version of Office. When re-opening the file in Office 2007, the application uses this information to determine whether to read the binary or XML representation of the shape. For example, a shape that was moved but not otherwise changed in Office 2003 could be rendered in Office 2007 using the full-fidelity XML data, but with adjusted coordinates.

This file does not store information affecting the appearance or meaning of a shape; it is used solely as a mechanism for converting shape information between the XML and binary representations. The following section provides the schema for this file and description of the relevant types, elements, and attributes.

Schema for downrev.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--============================================================================

   Module  : oartdownrev.xsd
   Owner   : [OFFICE]

   Copyright (c) Microsoft Corporation.  All rights reserved.

=============================================================================
   Schema definition for down-rev roundtriping
=============================================================================-->

<xsd:schema
targetNamespace="http://schemas.openxmlformats.org/drawingml/2006/main"
            elementFormDefault="qualified"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://schemas.openxmlformats.org/drawingml/2006/main">


   <!--==== Schema includes ================================================-->
   <xsd:include schemaLocation="oartbasetypes.xsd"/>
```

```
    <!--============================================================================
      Bounds Rect
      ============================================================================-->

    <xsd:complexType name="CT_BoundRect">
       <xsd:attribute name="l" type="ST_Coordinate" use="required"/>
       <xsd:attribute name="t" type="ST_Coordinate" use="required"/>
       <xsd:attribute name="r" type="ST_Coordinate" use="required"/>
       <xsd:attribute name="b" type="ST_Coordinate" use="required"/>
    </xsd:complexType>

    <!--============================================================================
      Precise Relative Rect
      ============================================================================-->

    <xsd:complexType name="CT_PreciseRelativeRect">
       <xsd:sequence>
          <xsd:element name="l" type="CT_Ratio" minOccurs="1" maxOccurs="1"/>
          <xsd:element name="t" type="CT_Ratio" minOccurs="1" maxOccurs="1"/>
          <xsd:element name="r" type="CT_Ratio" minOccurs="1" maxOccurs="1"/>
          <xsd:element name="b" type="CT_Ratio" minOccurs="1" maxOccurs="1"/>
       </xsd:sequence>
    </xsd:complexType>

    <!--============================================================================
      XL Anchor Point
      ============================================================================-->

    <xsd:complexType name="CT_XLLegacyAnchorPoint">
       <xsd:attribute name="row" type="xsd:int" use="required"/>
       <xsd:attribute name="col" type="xsd:int" use="required"/>
       <xsd:attribute name="rowOffset" type="ST_Coordinate" use="required"/>
       <xsd:attribute name="colOffset" type="ST_Coordinate" use="required"/>
    </xsd:complexType>

    <!--============================================================================
      XL Anchor
      ============================================================================-->

    <xsd:complexType name="CT_XLLegacyAnchor">
       <xsd:sequence>
          <xsd:element name="from" type="CT_XLLegacyAnchorPoint" minOccurs="1"
maxOccurs="1"/>
          <xsd:element name="to" type="CT_XLLegacyAnchorPoint" minOccurs="1"
maxOccurs="1"/>
       </xsd:sequence>
       <xsd:attribute name="fMove" type="xsd:boolean" use="optional"
default="false"/>
       <xsd:attribute name="fResize" type="xsd:boolean" use="optional"
default="false"/>
    </xsd:complexType>

    <!--============================================================================
      Down-rev save contents
      ============================================================================-->
    <xsd:complexType name="CT_DownRevStg">
```

```
        <xsd:sequence>
           <xsd:choice minOccurs="0" maxOccurs="1">
              <xsd:element name="bounds" type="CT_BoundRect"/>
              <xsd:element name="effectOffsets" type="CT_RelativeRect"/>
              <xsd:element name="preciseEffectOffsets"
   type="CT_PreciseRelativeRect"/>
              <xsd:element name="xlAnchor" type="CT_XLLegacyAnchor"/>
           </xsd:choice>
        </xsd:sequence>

        <xsd:attribute name="shapeCheckSum" type="xsd:string" use="required"/>
        <xsd:attribute name="textCheckSum" type="xsd:string" use="optional"/>
        <xsd:attribute name="shapeId" type="xsd:unsignedInt" use="optional"/>
        <xsd:attribute name="fHybridRaster" type="xsd:boolean" use="optional"/>
        <xsd:attribute name="rot" type="xsd:int" use="optional" default="0"/>
        <xsd:attribute name="ver" type="xsd:unsignedInt" use="optional" default="0"/>
        <xsd:attribute name="convChart" type="xsd:boolean" use="optional"
   default="false"/>
      </xsd:complexType>
   </xsd:schema>
```

Description of Types in Schema

Type	Description
CT_BoundsRect	Represents the transform bounds rectangle.
CT_PreciseRelativeRect	Represents a rectangle by the ratio of the offsets from the respective sides of some other rectangle. Left and right are the ratio of the rectangle width, and top and bottom are the ratio of the rectangle height. This is similar to RelativeRect, except we use Ratio instead of Percentage, so there won't be data loss after roundtrip.
CT_XLLegacyAnchorPoint	Represents the legacy Excel anchor point.
CT_XLLegacyAnchor	Represents the legacy Excel anchor.
CT_DownRevStg	Represents all information about how a shape was saved down-level.
Elements	
bounds	Represents the transform bounds of the shape.
effectOffsets	Obsolete; no longer written.
preciseEffectOffsets	Represents shape effect offsets.
xlAnchor	Represents shape anchoring if the shape is in an Excel document.
Attributes	
shapeCheckSum	Checksum computed on the shape. If the checksum doesn’t match the binary data when the file is loaded back into Office 2007, the XML data is considered out of date and the binary representation is used.
textCheckSum	Checksum computed on the text of the shape. If the checksum doesn’t match the binary data when the file is loaded back into Office 2007, the XML data is considered out of date and the binary representation of the text is used.

shapeId	Stores the shape ID; useful if the shape's full XML is not included.
fHybridRaster	Flag which is set on the group when rasterizing a shape into a legacy group of a picture and a textbox.
rot	Rotation of the shape; used to detect whether the shape's rotation has changed since this property is not computed in the shape checksum.
ver	Used to distinguish files saved by beta versions of Office. A value of 0 indicates beta 2 of Office 2007.
convChart	Whether a native chart converted to an OLE object when saved by Office 2007 should be converted back to a native object when reopened by Office 2007.

Schema for oartbasetypes.xsd

```
<?xml version="1.0" encoding="utf-8"?>
   <!--=========================================================================

      Module  : oartbasetypes.xsd
      Owner   : [OFFICE]

      Copyright (c) Microsoft Corporation.  All rights reserved.

   ==========================================================================-->

  <xsd:schema targetNamespace="http://schemas.openxmlformats.org/drawingml/2006/main"
elementFormDefault="qualified"
xmlns:r="http://schemas.openxmlformats.org/officeDocument/2006/relationships"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.openxmlformats.org/drawingml/2006/main">
    <xsd:import
namespace="http://schemas.openxmlformats.org/officeDocument/2006/relationships"
schemaLocation="shared-relationshipReference.xsd" />
    <xsd:simpleType name="ST_Coordinate">
      <xsd:annotation>
        <xsd:documentation>Coordinate</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:long">
        <xsd:minInclusive value="-27273042329600" />
        <xsd:maxInclusive value="27273042316900" />
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_Percentage">
      <xsd:annotation>
        <xsd:documentation>Percentage</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:int" />
    </xsd:simpleType>
    <xsd:complexType name="CT_Ratio">
      <xsd:attribute name="n" type="xsd:long" use="required">
        <xsd:annotation>
          <xsd:documentation>Numerator</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
```

```
      <xsd:attribute name="d" type="xsd:long" use="required">
        <xsd:annotation>
          <xsd:documentation>Denominator</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:complexType>
    <xsd:complexType name="CT_RelativeRect">
      <xsd:attribute name="l" type="ST_Percentage" use="optional" default="0">
        <xsd:annotation>
          <xsd:documentation>Left Offset</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="t" type="ST_Percentage" use="optional" default="0">
        <xsd:annotation>
          <xsd:documentation>Top Offset</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="r" type="ST_Percentage" use="optional" default="0">
        <xsd:annotation>
          <xsd:documentation>Right Offset</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="b" type="ST_Percentage" use="optional" default="0">
        <xsd:annotation>
          <xsd:documentation>Bottom Offset</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:schema>
```

Appendix G: Miscellaneous Enumerated Types and Structures

```
// MSOSHAPEPATH
typedef enum
    {
    msoshapeLines,        // A line of straight segments
    msoshapeLinesClosed,  // A closed polygonal object
    msoshapeCurves,       // A line of Bezier curve segments
    msoshapeCurvesClosed, // A closed shape with curved edges
    msoshapeComplex,      // pSegmentInfo must be non-empty
    } MSOSHAPEPATH;

// MSOWRAPMODE
typedef enum
    {
    msowrapSquare,
    msowrapByPoints,
    msowrapNone,
    msowrapTopBottom,
    msowrapThrough,
    } MSOWRAPMODE;

// MSOBWMODE
typedef enum
    {
    msobwColor,          // only used for predefined shades
    msobwAutomatic,      // depends on object type
    msobwGrayScale,      // shades of gray only
    msobwLightGrayScale, // shades of light gray only
    msobwInverseGray,    // dark gray mapped to light gray, etc.
    msobwGrayOutline,    // pure gray and white
    msobwBlackTextLine,  // black text and lines, all else grayscale
    msobwHighContrast,   // pure black and white mode (no grays)
    msobwBlack,          // solid black
    msobwWhite,          // solid white
    msobwDontShow,       // object not drawn
    msobwNumModes        // number of Black and white modes
    } MSOBWMODE;

// MSOANCHOR
typedef enum
    {
    msoanchorTop,
    msoanchorMiddle,
    msoanchorBottom,
    msoanchorTopCentered,
    msoanchorMiddleCentered,
    msoanchorBottomCentered,
    msoanchorTopBaseline,
    msoanchorBottomBaseline,
    msoanchorTopCenteredBaseline,
```

```
    msoanchorBottomCenteredBaseline
    } MSOANCHOR;

// MSOCDIR
typedef enum
    {
    msocdir0,       // Right
    msocdir90,      // Down
    msocdir180,     // Left
    msocdir270      // Up
    } MSOCDIR;

// MSOCXSTYLE - connector style
typedef enum
    {
    msocxstyleStraight = 0,
    msocxstyleBent,
    msocxstyleCurved,
    msocxstyleNone
    } MSOCXSTYLE;

// MSOCXK - the kind of connection sites
typedef enum
    {
    msocxkNone = 0,            // No connection sites
    msocxkSegments = 1,        // Connection sites at the segments
    msocxkCustom = 2,          // Sites defined by msopidPConnectionSites
    msocxkRect = 3,            // Use the connection sites for a rectangle
    } MSOCXK;

// MSOTXFL - text flow
typedef enum
    {
    msotxflHorzN,           //Horizontal non-@
    msotxflTtoBA,           //Top to Bottom @-font
    msotxflBtoT,            //Bottom to Top non-@
    msotxflTtoBN,           //Top to Bottom non-@
    msotxflHorzA,           //Horizontal @-font
    msotxflVertN,           //Vertical, non-@
    } MSOTXFL;

// MSOTXDIR - text direction (needed for Bi-Di support)
typedef enum
    {
    msotxdirLTR,           // left-to-right text direction
    msotxdirRTL,           // right-to-left text direction
    msotxdirContext,       // context text direction
    } MSOTXDIR;

// MSOSPCOT - Callout Type
typedef enum
    {
    msospcotRightAngle = 1,
    msospcotOneSegment = 2,
```

```
    msospcotTwoSegment = 3,
    msospcotThreeSegment = 4,
    } MSOSPCOT;

// MSOSPCOA - Callout Angle
typedef enum
    {
    msospcoaAny,
    msospcoa30,
    msospcoa45,
    msospcoa60,
    msospcoa90,
    msospcoa0,
    } MSOSPCOA;

// MSOSPCOD - Callout Drop
typedef enum
    {
    msospcodTop,
    msospcodCenter,
    msospcodBottom,
    msospcodSpecified,
    } MSOSPCOD;

// MSOGEOTEXTALIGN - WordArt alignment
typedef enum
    {
    msoalignTextStretch,      /* Stretch each line of text to fit width. */
    msoalignTextCenter,       /* Center text on width. */
    msoalignTextLeft,         /* Left justify. */
    msoalignTextRight,        /* Right justify. */
    msoalignTextLetterJust,   /* Spread letters out to fit width. */
    msoalignTextWordJust,     /* Spread words out to fit width. */
    msoalignTextInvalid       /* Invalid */
    } MSOGEOTEXTALIGN;

// MSOBLIPFLAGS - flags for pictures
typedef enum
    {
    msoblipflagDefault = 0,
    msoblipflagComment = 0, // Blip name is a comment
    msoblipflagFile,          // Blip name is a file name
    msoblipflagURL,           // Blip name is a full URL
    msoblipflagType = 3,      // Mask to extract type
    /* Or the following flags with any of the above. */
    msoblipflagDontSave   = 4, // A "dont" is the depression in the metal
                               // body work of an automobile caused when a
                               // cyclist violently thrusts his or her nose
                               // at it, thus a DontSave is another name for
                               // a cycle lane.
    msoblipflagDoNotSave  = 4, // For those who prefer English
    msoblipflagLinkToFile = 8,
    } MSOBLIPFLAGS;
```

```
// MSO3DRENDERMODE
typedef enum
    {
    msoFullRender,   // Generate a full rendering
    msoWireframe,    // Generate a wireframe
    msoBoundingCube, // Generate a bounding cube
    } MSO3DRENDERMODE;

// MSOXFORMTYPE
typedef enum
    {
    msoxformAbsolute,   // Apply transform in absolute space centered on shape
    msoxformShape,      // Apply transform to shape geometry
    msoxformDrawing     // Apply transform in drawing space
    } MSOXFORMTYPE;

// MSOSHADOWTYPE
typedef enum
    {
    msoshadowOffset,    // N pixel offset shadow
    msoshadowDouble,    // Use second offset too
    msoshadowRich,      // Rich perspective shadow (cast relative to shape)
    msoshadowShape,     // Rich perspective shadow (cast in shape space)
    msoshadowDrawing,   // Perspective shadow cast in drawing space
   msoshadowEmbossOrEngrave,
    } MSOSHADOWTYPE;

// MSODZTYPE - the type of a (length) measurement
typedef enum
    {
    msodztypeMin         = 0,
    msodztypeDefault     = 0,  // Default size, ignore the values
    msodztypeA           = 1,  // Values are in EMUs
    msodztypeV           = 2,  // Values are in pixels
    msodztypeShape       = 3,  // Values are 16.16 fractions of shape size
    msodztypeFixedAspect = 4,  // Aspect ratio is fixed
    msodztypeAFixed      = 5,  // EMUs, fixed aspect ratio
    msodztypeVFixed      = 6,  // Pixels, fixed aspect ratio
    msodztypeShapeFixed  = 7,  // Proportion of shape, fixed aspect ratio
    msodztypeFixedAspectEnlarge = 8,  // Aspect ratio is fixed, favor larger size
    msodztypeAFixedBig   = 9,  // EMUs, fixed aspect ratio
    msodztypeVFixedBig   = 10, // Pixels, fixed aspect ratio
    msodztypeShapeFixedBig= 11,// Proportion of shape, fixed aspect ratio
    msodztypeMax         = 11
    } MSODZTYPE;

// MSOFILLTYPE
typedef enum
    {
    msofillSolid,             // Fill with a solid color
    msofillPattern,           // Fill with a pattern (bitmap)
    msofillTexture,           // A texture (pattern with its own color map)
    msofillPicture,           // Center a picture in the shape
    msofillShade,             // Shade from start to end points
```

```
    msofillShadeCenter,       // Shade from bounding rectangle to end point
    msofillShadeShape,        // Shade from shape outline to end point
    msofillShadeScale,        // Similar to msofillShade, but the fillAngle
                                  // is additionally scaled by the aspect ratio of
                                  // the shape. If shape is square, it is the
                                  // same as msofillShade.
    msofillShadeTitle,        // special type - shade to title ---  for PP
    msofillBackground         // Use the background fill color/pattern
    } MSOFILLTYPE;

// MSOSHADETYPE - how to interpret the colors in a shaded fill.
typedef enum
    {
    msoshadeNone  = 0,        // Interpolate without correction between RGBs
    msoshadeGamma = 1,        // Apply gamma correction to colors
    msoshadeSigma = 2,        // Apply a sigma transfer function to position
    msoshadeBand  = 4,        // Add a flat band at the start of the shade
    msoshadeOneColor = 8,     // This is a one color shade

    /* A parameter for the band or sigma function can be stored in the top
        16 bits of the value - this is a proportion of *each* band of the
        shade to make flat (or the approximate equal value for a sigma
        function).  NOTE: the parameter is not used for the sigma function,
        instead a built in value is used.  This value should not be changed
        from the default! */
    msoshadeParameterShift = 16,
    msoshadeParameterMask  = 0xffff0000,

    msoshadeDefault = (msoshadeGamma|msoshadeSigma|
                           (16384<<msoshadeParameterShift))
    } MSOSHADETYPE;

//  MSOLINESTYLE - compound line style
typedef enum
    {
    msolineSimple,     // Single line (of width lineWidth)
    msolineDouble,     // Double lines of equal width
    msolineThickThin,  // Double lines, one thick, one thin
    msolineThinThick,  // Double lines, reverse order
    msolineTriple      // Three lines, thin, thick, thin
    } MSOLINESTYLE;

// MSOLINETYPE - how to "fill" the line contour
typedef enum
    {
    msolineSolidType,         // Fill with a solid color
    msolinePattern,           // Fill with a pattern (bitmap)
    msolineTexture,           // A texture (pattern with its own color map)
    msolinePicture            // Center a picture in the shape
    } MSOLINETYPE;

// MSOLINEDASHING - dashed line style
typedef enum
    {
```

```
    msolineSolid,            // Solid (continuous) pen
    msolineDashSys,          // PS_DASH system   dash style
    msolineDotSys,           // PS_DOT systemdash style
    msolineDashDotSys,       // PS_DASHDOT system dash style
    msolineDashDotDotSys,    // PS_DASHDOTDOT system dash style
    msolineDotGEL,           // square dot style
    msolineDashGEL,          // dash style
    msolineLongDashGEL,      // long dash style
    msolineDashDotGEL,       // dash short dash
    msolineLongDashDotGEL,   // long dash short dash
    msolineLongDashDotDotGEL // long dash short dash short dash
    } MSOLINEDASHING;

// MSOLINEEND - line end effect
typedef enum
    {
    msolineNoEnd,
    msolineArrowEnd,
    msolineArrowStealthEnd,
    msolineArrowDiamondEnd,
    msolineArrowOvalEnd,
    msolineArrowOpenEnd,
    } MSOLINEEND;

// MSOLINEENDWIDTH - size of arrowhead
typedef enum
    {
    msolineNarrowArrow,
    msolineMediumWidthArrow,
    msolineWideArrow
    } MSOLINEENDWIDTH;

//  MSOLINEENDLENGTH - size of arrowhead
typedef enum
    {
    msolineShortArrow,
    msolineMediumLenArrow,
    msolineLongArrow
    } MSOLINEENDLENGTH;

//  MSOLINEJOIN - line join style.
typedef enum
    {
    msolineJoinBevel,    // Join edges by a straight line
    msolineJoinMiter,    // Extend edges until they join
    msolineJoinRound     // Draw an arc between the two edges
    } MSOLINEJOIN;

//  MSOLINECAP - line cap style (applies to ends of dash segments too).
typedef enum
    {
    msolineEndCapRound,   // Rounded ends - the default
    msolineEndCapSquare,  // Square protrudes by half line width
    msolineEndCapFlat     // Line ends at end point
```

```
    } MSOLINECAP;

// ADJH - adjust handle
struct ADJH
    {
    ULONG   f;                    // type of adjust handle
    ADJPOS  apX;                  // X position of the handle
    ADJPOS  apY;                  // Y position of the handle
    LONG        xRange;           // Map values
    LONG        yRange;
    LONG        xMin;             // Pin values
    LONG        xMax;
    LONG        yMin;
    LONG        yMax;
    };

// SGF - Shape Guide Formula
enum SGF
    {
    // Arithmetic
    sgfSum,              // guide := value + parameter1 - parameter2
    sgfProduct,          // guide := value * parameter1 / parameter2 (MulDiv)
    sgfMid,              // guide := (value + param1)/2

    // Logical
    sgfAbsolute,         // guide := fabs(value)
    sgfMin,              // guide := min(value, param1)
    sgfMax,              // guide := max(value, param1)
    sgfIf,               // guide := vaule > 0 ? param1 : param2

    // Polar arithmetic - angles are 16.16 degrees
    sgfMod,              // guide := sqrt(value^2 + param1^2 + param2^2)
    sgfATan2,            // guide := atan2(param1, value) [param2 ignored]
    sgfSin,              // guide := value * sin(param1)  [param2 ignored]
    sgfCos,              // guide := value * cos(param1)  [param2 ignored]
    sgfCosATan2,         // guide := value * cos(atan2(param2, param1))
    sgfSinATan2,         // guide := value * sin(atan2(param2, param1))
    sgfSqrt,             // guide := sqrt(value) [param1 and param2 ignored]
    sgfSumAngle,
    sgfEllipse,
    sgfTan,              // guide := value * tan(param1)  [param2 ignored]
};

// SGVT - Shape guide parameter type
enum SGVT
    {
    sgvtV  = 0x2000,     // Value is adjust value or guide
    sgvtP1 = 0x4000,     // Parameter1 is adjust value or guide
    sgvtP2 = 0x8000      // Parameter2 is adjust value or guide
    };

// Value index: The first set define the value as an adjust handle property.
const USHORT sgvAdjust1 = msopidAdjustValue;
const USHORT sgvAdjust2 = msopidAdjust2Value;
```

```
const USHORT sgvAdjust3 = msopidAdjust3Value;
const USHORT sgvAdjust4 = msopidAdjust4Value;
const USHORT sgvAdjust5 = msopidAdjust5Value;
const USHORT sgvAdjust6 = msopidAdjust6Value;
const USHORT sgvAdjust7 = msopidAdjust7Value;
const USHORT sgvAdjust8 = msopidAdjust8Value;

// Geometry width and height
const USHORT sgvWidth   = msopidGeoRight;
const USHORT sgvHeight  = msopidGeoBottom;

// Other geometry properties
const USHORT sgvXCenter = msopidGeoLeft;
const USHORT sgvYCenter = msopidGeoTop;
const USHORT sgvXLimo   = msopidXLimo;
const USHORT sgvYLimo   = msopidYLimo;

// The guide properties
#define sgvGuide_(n) (USHORT(msopidLast + 1 + (n)))

/* View coordinate properties in V units (pixesl) and EMU, the EMU scaling
    is the average of the width/height scaling */
const USHORT sgvLineWidth = sgvGuide_(247);  // line width in pixels
const USHORT sgvDxvAnchor = sgvGuide_(248);  // width in pixels
const USHORT sgvDyvAnchor = sgvGuide_(249);  // height in pixels

const USHORT sgvDxeAnchor = sgvGuide_(252);  // width of a shape in the view in emus
const USHORT sgvDyeAnchor = sgvGuide_(253);  // Height of a shape in the view in emus
const USHORT sgvXeAnchorCenter = sgvGuide_(254);
const USHORT sgvYeAnchorCenter = sgvGuide_(255);

// SG - Shape Guide
struct SG
    {
    ULONG sgf:16;
    ULONG iValue:16;
    ULONG a:16;
    ULONG b:16;
    };

// MSOGV - Generic Value
typedef void *MSOGV;

// DiaGraM Node Kind
enum DGMNK
    {
    dgmnkMin = 0,
    dgmnkNode = dgmnkMin,
    dgmnkRoot,
    dgmnkAssistant,
    dgmnkCoWorker,
    dgmnkSubordinate,
    dgmnkAuxNode,
    dgmnkDefault, // used for the orgchart split bar
```

```
    dgmnkMax, dgmnkLast = dgmnkMax - 1,
    dgmnkNil = 0xFFFF,
    };

/* MSODGMLO = DiaGgraG LayOut */
typedef enum
        {
        msodgmloFirst = 0, msodgmloMin = msodgmloFirst, msodgmloMinLessOne = msodgmloMin - 1,

        // OrgChart layout
        msodgmloOrgChartMin,
        msodgmloOrgChartStd = 0,
        msodgmloOrgChartBothHanging,
        msodgmloOrgChartRightHanging,
        msodgmloOrgChartLeftHanging,
        msodgmloOrgChartMax, msodgmloOrgChartLast = msodgmloOrgChartMax - 1,

        // Cycle layout
        msodgmloCycleMin, msodgmloCycleMinLessOne = msodgmloCycleMin - 1,
        msodgmloCycleStd,
        msodgmloCycleMax, msodgmloCycleLast = msodgmloCycleMax - 1,

        // Radial layout
        msodgmloRadialMin, msodgmloRadialMinLessOne = msodgmloRadialMin - 1,
        msodgmloRadialStd,
        msodgmloRadialMax, msodgmloRadialLast = msodgmloRadialMax - 1,

        // Stacked layout
        msodgmloStackedMin, msodgmloStackedMinLessOne = msodgmloStackedMin - 1,
        msodgmloStackedStd,
        msodgmloStackedMax, msodgmloStackedLast = msodgmloStackedMax - 1,

        // Venn layout
        msodgmloVennMin, msodgmloVennMinLessOne = msodgmloVennMin - 1,
        msodgmloVennStd,
        msodgmloVennMax, msodgmloVennLast = msodgmloVennMax - 1,

        // BullsEye layout
        msodgmloBullsEyeMin, msodgmloBullsEyeMinLessOne = msodgmloBullsEyeMin - 1,
        msodgmloBullsEyeStd,
        msodgmloBullsEyeMax, msodgmloBullsEyeLast = msodgmloBullsEyeMax - 1,

        msodgmloMax, msodgmloLast = msodgmloMax -1,
        msodgmloNil = 0xFF,
        } MSODGMLO;

// MSODGMT -- DiaGraM Type
typedef enum
        {
        msodgmtMin                      = 0,
        msodgmtCanvas                   = msodgmtMin,
        msodgmtFirstDiagramType         = 1,
        msodgmtOrgChart                 = msodgmtFirstDiagramType,
        msodgmtRadial                   = 2,
```

```
        msodgmtCycle                    = 3,
        msodgmtStacked                  = 4,
        msodgmtVenn                     = 5,
        msodgmtBullsEye                 = 6,
        msodgmtMax, msodgmtLast         = msodgmtMax - 1,
        msodgmtNil                      = 0x0FFF,
        } MSODGMT;

/* MSODGMST = DiaGraM STyle */
typedef enum
        {
        /***WARNING: This is written out to the file format! ***/
        /***NOTE: If you add a new style enum, you need to update vrgdgmstdesc ***/

        msodgmstMin = 0, msodgmstFirst = msodgmstMin ,

        // OrgChart styles
        msodgmstOrgChartFirst = msodgmstFirst,
        msodgmstOrgChart2,
        msodgmstOrgChart3,
        msodgmstOrgChart4,
        msodgmstOrgChart5,
        msodgmstOrgChart6,
        msodgmstOrgChart7,
        msodgmstOrgChart8,
        msodgmstOrgChart9,
        msodgmstOrgChart10,
        msodgmstOrgChart11,
        msodgmstOrgChart12,
        msodgmstOrgChart13,
        msodgmstOrgChart14,
        msodgmstOrgChart15,
        msodgmstOrgChart16,
        msodgmstOrgChart17,
        msodgmstOrgChartMax,
        msodgmstOrgChartLast = msodgmstOrgChartMax - 1,

        // Radial styles
        msodgmstRadialFirst = msodgmstFirst,
        msodgmstRadial2,
        msodgmstRadial3,
        msodgmstRadial4,
        msodgmstRadial5,
        msodgmstRadial6,
        msodgmstRadial7,
        msodgmstRadial8,
        msodgmstRadial9,
        msodgmstRadial10,
        msodgmstRadialMax,
        msodgmstRadialLast = msodgmstRadialMax - 1,

        // Cycle styles
        msodgmstCycleFirst = msodgmstFirst,
        msodgmstCycle2,
```

```
        msodgmstCycle3,
        msodgmstCycle4,
        msodgmstCycle5,
        msodgmstCycle6,
        msodgmstCycle7,
        msodgmstCycle8,
        msodgmstCycle9,
        msodgmstCycle10,
        msodgmstCycle2First,
        msodgmstCycle11 = msodgmstCycle2First,
        msodgmstCycle12,
        msodgmstCycle13,
        msodgmstCycle14,
        msodgmstCycle15,
        msodgmstCycle16,
        msodgmstCycle17,
        msodgmstCycle18,
        msodgmstCycle19,
        msodgmstCycle20,
        msodgmstCycle21,
        msodgmstCycleMax,
        msodgmstCycleLast = msodgmstCycleMax - 1,

        // Stacked styles
        msodgmstStackedFirst = msodgmstFirst,
        msodgmstStacked2,
        msodgmstStacked3,
        msodgmstStacked4,
        msodgmstStacked5,
        msodgmstStacked6,
        msodgmstStacked7,
        msodgmstStacked8,
        msodgmstStacked9,
        msodgmstStacked10,
        msodgmstStackedMax,
        msodgmstStackedLast = msodgmstStackedMax - 1,

        // Venn styles
        msodgmstVennFirst = msodgmstFirst,
        msodgmstVenn2,
        msodgmstVenn3,
        msodgmstVenn4,
        msodgmstVenn5,
        msodgmstVenn6,
        msodgmstVenn7,
        msodgmstVenn8,
        msodgmstVenn9,
        msodgmstVenn10,
        msodgmstVennMax,
        msodgmstVennLast = msodgmstVennMax - 1,

        // BullsEyeChart styles
        msodgmstBullsEyeFirst = msodgmstFirst,
        msodgmstBullsEye2,
```

```
        msodgmstBullsEye3,
        msodgmstBullsEye4,
        msodgmstBullsEye5,
        msodgmstBullsEye6,
        msodgmstBullsEye7,
        msodgmstBullsEye8,
        msodgmstBullsEye9,
        msodgmstBullsEye10,
        msodgmstBullsEyeMax,
        msodgmstBullsEyeLast = msodgmstBullsEyeMax - 1,

        msodgmstNil = 0xFFFF,
        } MSODGMST;

//------------------------------------------------------------------------------
// DiaGraM ReLationship
//------------------------------------------------------------------------------
struct DGMRL
    {
    ULONG spidSrc;  // spid of the source node.
    ULONG spidDest; // spid of the destination node.
    ULONG spidCntr; // spid of the connector.
    };
```