Adobe Flash (SWF) Writer

The Adobe Flash® (SWF) Writer provides FME with the ability to write Flash export files. Flash is a movie creation and display application designed to deliver vector graphics and animation over the Internet.

Overview

Flash files are stored in compressed, binary, tag-formatted `.swf` files. The MIME type is `application/x-shockwave-flash`.

Flash files are compressed so they can be used as web content, but this means that file creation takes longer due to the extensive compression techniques that are employed. Writing the movie to the `.swf` file can take up to three times as long as processing the features. Processing the features themselves is also more time-consuming than with some other formats due to the conversions and formatting of features that takes place (see notes on particular features).

In addition, since Flash was not originally designed to handle maps or large numbers of features, performance declines according to the number of features. Therefore, in order to improve performance, we recommend that the features be generalized before being sent to the Flash writer.

A Flash movie may consist of several frames, however, the Flash Writer uses only one frame at a time to hold the converted image if the animate draw option is turned off; otherwise, each feature is stored in a frame and then they are displayed incrementally, which results in the map drawing itself bit by bit.

Flash Quick Facts

Format Type Identifier	FLASH
Reader/Writer	Writer
Licensing Level	Base
Dependencies	None
Dataset Type	File
Feature Type	"FLASH"
Typical File Extensions	.swf
Automated Translation Support	Yes
User-Defined Attributes	No
Coordinate System Support	No
Generic Color Support	No
Spatial Index	Not applicable
Schema Required	Yes
Transaction Support	No
Geometry Type Attribute	flash_type

Geometry Support			
Geometry	**Supported?**	**Geometry**	**Supported?**
aggregate	no	polygon	yes
circles	no	donut polygon	yes
circular arc	yes	line	yes
elliptical arc	yes	point	yes
ellipses	yes	text	yes
none	no	z values	no
surface	no	solid	no

SWF File Structure

An SWF file is made up of a header, followed by a number of tags. There are two types of tags:

- *Definition tags* define the objects known as characters, which are stored in the dictionary.
- *Control tags* manipulate characters, and control the flow of the movie.

All tags share a common format, so any program parsing an SWF file can skip over blocks it does not understand. Tags can be removed, inserted, or modified by tools that process an SWF file.

The Flash writer will be primarily concerned with the definition tags for the shapes and will use the control tags for only simple movie flow control such as the begin and end tags, and placing the objects in the frames.

All Flash frames in a movie are contained in a single file with an .swf extension.

Filename Extension	Contents
.swf	All vector geometric data in movie frames

Shapes

Shape representation is similar to most vector formats in that shapes are defined by a list of edges called a path. A path may be closed (where the start and end of the path meet to close the figure), or open (where the path forms an open-ended stroke). A path may contain a mixture of straight edges, curved edges, and "pen up and move" commands. The latter allows multiple disconnected figures to be described by a single shape structure.

A fill style defines the appearance of an area enclosed by a path. Fill styles supported by SWF include a solid color or two gradient fills.

A line style defines the appearance of the outline of a path. The line style may be a stroke of any thickness and color.

Most vector formats only allow one fill and line style per path. SWF extends this concept by allowing each edge to have its own line and fill style. This can have unpredictable results when fill styles change in the middle of a path. Flash also supports two fill styles per edge, one for each side of the edge. Note that this capability is not utilized by the Flash writer and all edges of a polygon or line will have the same line and fill styles.

Fonts and Text

Flash text is supported by FME but is represented as line segments, not as real text glyphs. This is for simplicity and because of incompatibility between the Hershey fonts used and the Flash glyph format.

Writer Overview

The Flash writer creates and writes feature data to a SWF file specified by the DATASET keyword. The writer searches the mapping file for the <WriterKeyword>_DATASET keyword in the mapping file. This keyword is required to be in the mapping file. An old SWF file in the directory with the same file name is overwritten with the new feature data. A typical mapping file fragment specifying the output SWF file looks like:

```
FLASH_DATASET /usr/data/flash/myfile.swf
```

The <WriterKeyword>_MAINTAIN_ASPECT keyword is another optional keyword. It should be followed by a value of YES or NO. By default, the value is set to YES. A YES indicates that the original map aspect will be maintained to fit within the destination defined bounding box. This means that the entire destination bounding box defined may not used. Alternatively, the value NO causes the original map to be clipped by the box if it is too large. Note that smaller frames are not expanded to fill the bounding box.

The <WriterKeyword>_BACKGROUND_COLOR keyword is another optional keyword. It should be followed by a string with three comma separated values from 0 to 255 indi-

cating the intensity values of red, green and blue respectively. This color is then set as the background color of the movie. The default color is white.

The <WriterKeyword>_ANIMATE_DRAW keyword is another optional keyword. It should be followed by a value of YES or NO. By default, the value is set to YES. A YES indicates that the features processed are placed into separate frame so that at drawing time the image draws features to the screen progressively and can be observed by the user. Alternatively, the value NO causes the features to be drawn on the same frame which is loaded all at once, with no display to the user until it is complete, at which point it is displayed.

Keyword	Description
DATASET	The file that will be written to **Range:** Valid File Name **Default:** None **Optional:** No
MAINTAIN_ASPECT	Specifies whether or not the source map dimensions will be kept or stretched to fit to the output bounding box. **Range:** YES or NO **Default:** YES **Optional:** Yes
RESOLUTION_X	Specifies the maximum Flash units (1 unit = 1/20 pixel) for the x dimension of the output map **Range:** 2000 -24000 **Default:** 16000 **Optional:** Yes
RESOLUTION_Y	Specifies the maximum Flash units (1 unit = 1/20 pixel) for the y dimension of the output map **Range:** 2000 - 18000 **Default:** 12000 **Optional:** Yes
BACKGROUND_COLOR	Specifies the color of the background of the movie. **Range:** String(0..255, 0..255, 0..255) **Default:** White **Optional:** Yes
ANIMATE_DRAW	Specifies whether to animate the drawing process of just make the user wait for it to be entirely done before displaying it on the screen. **Range:** YES or NO **Default:** YES **Optional:** Yes

Feature Representation

Flash features consist of geometric shapes which have associated with them attributes as part of their definition.

In addition to the generic FME feature attributes that FME Workbench adds to all features (see *About Feature Attributes* on page 7), all Flash features contain a flash_type attribute, which identifies the geometric type. All element types also have a color associated with it. All coordinate points and line widths are in *twips* instead of pixels,

where 20 twips = 1 pixel. Depending on the geometric type, the feature contains additional attributes specific to the geometric type. Note that points are a subset of area, circles are a subset of ellipse and most polygons and closed shapes fall into the area category. Additional attributes are described in subsequent sections. Here are the common attributes of all flash features.

Note that for closed areas that can hold a URL, the URL can only be active if the area has not been rotated, so it both are specified, the rotation takes precedence.

Attribute Name	Contents
flash_type	The Flash geometric type of this entity. **Range:** flash_point| flash_line| flash_text| flash_area| flash_arc| flash_ellipse **Default:** No default
flash_linecolor	This is a string that represents the color intensities of the element. It is formatted as red, green, blue intensities which range between 0..255 Note that if this attribute is not found, then fme_color will be used. **Range:** String. (0..255, 0..255, 0..255) **Default:** String (0,0,0)
fme_linecolor	This is a string that represents the color intensities of the element. It is formatted as red, green, blue intensities which range between 0..1 This is used as a backup to flash_color. **Range:** String. (0..1, 0..1, 0..1) **Default:** String(0,0,0)
flash_linewidth	This is an integer value representing the width of the line in twips. **Range:** Integer>=0 **Default:** 20
flash_rotation	This attribute specifies and optional rotation for the shape where the clockwise direction is positive. **Range:** -360 to 360 degrees **Default:** 0

Text

flash_type: flash_text

Flash text is supported through the drawing of Hershey fonts in FME. Basically the string is broken into letters which are then broken into lines and drawn as such. Attributes line text_size may be specified but the font is fixed due to availability of Her-

shey fonts. Essentially, though made up of lines, the text has no linewidth or color that is available for setting.

Attribute Name	Contents
flash_text_string	This is the string that will be written to the flash movie. **Range:** Any String value? **Default:** "No String Value" **Optional:** No
flash_text_size	This is the size for the text height and if specified, the text will have this height in ground units multiplied by a Hershey height factor to result in a reasonable size of text that if hopefully readable. **Range:** Integer **Default:** 20 **Optional:** Yes

Points

flash_type: flash_point

Flash point features specify a single set of coordinates which is converted. Each point has a colour associated with it. If not defined, default parameters are used, black for the color. URL is the only additional attribute available to control the Flash point settings other than the common ones for all feature types listed. Note: the concept of a line width is absent in the point feature.

Attribute Name	Contents
flash_URL	This is the URL for the point and if specified, the point will have a small circle around it which is linked to the given URL. **Range:** Any String value? **Default:** None

Lines

flash_type: flash_line

Flash line features specify linear features defined by a starting point and then a series of delta x and y values specifying the distance to travel from the last point, thus a line consists of 1 or more line segments. Each line has a colour and line width used when the line is drawn. If not defined, default parameters are used. The no special attributes are required to control the Flash line settings other than the common ones for all feature types listed. Since Flash does not support a line class in its swf API, lines are broken into strictly monotone segments and then buffered into simple polygons.

Arcs

flash_type: flash_arc

The flash_arcs are really just converted to a series of line segments to be drawn as a line since in Flash they are defined in terms of a delta x and delta y value pair for the

distances from the starting point to the control point and a control x and control y value pair for the distance from the control point to the end point. This does not match the FME definition of an arc, thus to avoid gruesome math, the arc is stroked into a series of line segments and drawn as a line.

Areas

flash_type: flash_area

Flash area features specify various polygon features. The areas that make up a single feature may or may not be disjoint. Note that the flash linewidth and linecolor now refer to the border lines of the closed polygons. Any closed area can be used as a button for events so long is it has not undergone any transformations, i..e., rotation, scaling or translation.

The following table lists the special FME attribute names used to control the flash area settings.

Attribute Name	Contents
flash_fill_color	This is a string that represents the color intensities of the element. It is formatted as red, green, blue intensities which range between 0..255. If this attribute is not found, then the polygon will not be filled. **Range:** String. (0..255, 0..255, 0..255) **Default:** None
fme_fillcolor	This is a string that represents the color intensities of the element. It is formatted as red, green, blue intensities which range between 0..1 This is used as a backup to flash_fill_color. **Range:** String. (0..1, 0..1, 0..1) **Default:** String(0,0,0)
flash_fillstyle	This is the value that determines which of the five available fill styles will be used when filling a closed area: nofill,solidfill, left to right linearfill, right to left linear fill, and radialfill. Note this requires a fill color to be specified. **Range:** 0,1,2,3,4 **Default:** 0 (nofill)
flash_URL	This is the URL for the area and if specified, the area will have a URL linked to it its shape unless rotated, scaled, or translated, any of which will negate the URL property. **Range:** Any String value? **Default:** None

Ellipses

flash_type: flash_ellipse

The flash_ellipse features are simply closed arc features and thus are handled in the same manner. The ellipse is stroked into a line segment and drawn as a line. Note that the flash linewidth and linecolor now refer to the border lines of the ellipse. Any ellipse can be used as a button for events so long is it has not undergone any transformations, i..e. rotation, scaling or translation. From the flash ellipse we also can arrive at circles since they are just ellipses with equal axis.

The following table lists the additional FME attribute names used to control the flash ellipse settings. on top of the common ones.

Attribute Name	Contents
flash_fill_color	This is a string that represents the color intensities of the element. It is formatted as red, green, blue intensities which range between 0..255. If this attribute is not found, then the ellipse will not be filled. **Range:** String. (0.255, 0..255, 0..255) **Default:** None
fme_fillcolor	This is a string that represents the color intensities of the element. It is formatted as red, green, blue intensities which range between 0..1 This is used as a backup to flash_fill_color. **Range:** String. (0..1, 0..1, 0..1) **Default:** String(0,0,0)
flash_fillstyle	This is the value that determines which of the five available fill styles will be used when filling an ellipse: nofill,solidfill, left to right linearfill, right to left linear fill, and radialfill. Note this requires a fill color to be specified. **Range:** 0,1,2,3,4 **Default:** 0 (nofill)
flash_URL	This is the URL for the ellipse and if specified, the ellipse will have a URL linked to it its shape unless rotated, scaled, or translated, any of which will negate the URL property. **Range:** Any String value? **Default:** None