

# ESRI ArcInfo Generate Reader/Writer

The ESRI® ArcInfo Generate File Reader and Writer allows the Feature Manipulation Engine (FME) to access the simple ASCII format used by the ArcInfo Generate command. There are several types of Generate files, and each has its own syntax. Currently, point, line, and text Generate files are supported. Both two-dimensional and three-dimensional data can be imported and exported to line and point Generate files. Three-dimensional (3D) Generate files are often used to input data into ArcInfo's TINning package. Text Generate Files are defined to accept only two-dimensional (2D) coordinates.

## ARCGEN Quick Facts

Format Type Identifier	ARCGEN
Reader/Writer	Both
Licensing Level	Base
Dependencies	None
Dataset Type	Directory or File
Feature Type	.gen
Typical File Extensions	Yes
Automated Translation Support	File base name
User-Defined Attributes	No
Coordinate System Support	No
Generic Color Support	No
Spatial Index	Never
Schema Required	Yes
Transaction Support	No
Geometry Type	ARCGEN_GEOMETRY

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

## Overview

FME considers a Generate dataset to be a collection of Generate files in a single directory. All Generate file names are required to end with a `.gen` extension. The type of each Generate file must be defined in the mapping file before it can be read or written.

---

**Tip:** The very simple format of Generate files makes them useful for testing purposes or for transferring data between FME and other unsupported systems.

---

## Reader Overview

The ARCGEN Reader produces FME features for all feature data held in Generate files residing in a given directory.

The ARCGEN Reader first scans the directory it is given for all Generate files defined in the mapping file. For each Generate file it finds, it checks to see if that file is requested by looking at the list of IDs specified in the mapping file. If a match is made or if no IDs were specified in the mapping file, the Generate file is opened for read. The Generate reader extracts features from the file one at a time and passes them on to the rest of the FME for further processing. When the file is exhausted, the Generate reader starts on the next file in the directory. Optionally, a single Generate file can be provided as the dataset. In this case, only that Generate file is read.

## Reader Directives

The suffixes listed are prefixed by the current `<ReaderKeyword>` in a mapping file. By default, the `<ReaderKeyword>` for the Generate reader is `ARCGEN`.

### DATASET

**Required/Optional:** *Required*

Contains the directory name of the input Generate files, or a single Generate file.

**Workbench Parameter:** [<WorkbenchParameter>](#)

### DEF

**Required/Optional:** *Required*

Each Generate file must be defined before it can be read. The definition contains the file's base name (without the `.gen` extension), the type of geometry it contains, and optionally a delimiter between fields. There may be many DEF lines, one for each file to be read.

The syntax of a Generate DEF line is:

```
<ReaderKeyword>_DEF <baseName> \
  ARCGEN_GEOMETRY (arcgen_point|arcgen_line| \
    arcgen_text) \
  [ARCGEN_DELIMITER "<delimiter char>"]
```

The file name of the Generate file is the basename plus the `.gen` extension.

The mapping file fragment below defines two Generate files—one containing point features and the other containing linear features:

```
ARCGEN_DEF nodes          ARCGEN_GEOMETRY arcgen_point
ARCGEN_DEF boundaries ARCGEN_GEOMETRY arcgen_line
```

If no delimiter clause is specified, a comma ( , ) is used as the delimiter.

**Workbench Parameter:** [<WorkbenchParameter>](#)

## IDs

**Required/Optional:** *Optional*

This directive is used to limit which available and defined Generate files will be read. If no IDs are specified, then all defined and available Generate files will be read. The syntax of the IDs keyword is:

```
<ReaderKeyword>_IDs <baseName1> \
                    <baseName1> ... \
                    <baseNameN>
```

The basenames must match those used in DEF lines.

### Example:

The example below selects only the nodes Generate file for input during a translation:

```
ARCGEN_IDS nodes
```

**Workbench Parameter:** [<WorkbenchParameter>](#)

## CLOSED\_LINES\_AS\_POLYS

**Required/Optional:** *Optional*

This directive specifies how to determine the type of closed lines which may be indistinguishable from polygons.

### Example:

```
ARCGEN_CLOSED_LINES_AS_POLYS no
```

**Value:** YES | NO

**Workbench Parameter:** [<WorkbenchParameter>](#)

## Writer Overview

The ARCGEN Writer creates and writes feature data to Generate files in the directory specified by the DATASET directive.

As with the reader, the directory must exist before the translation occurs. Any old Generate files in the directory are overwritten with the new feature data. As features are routed to the Generate writer by FME, it determines which file to write to and outputs them according to the type of the file. Many Generate files can be written during a single FME session.

## Writer Directives

The Generate writer processes the `DATASET` and `DEF` directives as described in the *Reader Directives* section. It does not make use of the `IDS` directive.

## Feature Representation

In addition to the generic FME feature attributes that FME Workbench adds to all features (see the chapter *About Feature Attributes*), this format adds the format-specific attributes described in this section.

All Generate features contain a numeric ID field. The numeric ID is stored in the `arcgen_id` attribute of an FME feature read from a Generate file or destined to be written to a Generate file.

---

**Tip:** Features being written to an `ARCGEN` file *must* have an `arcgen_id` attribute.

---

FME considers the basename of the Generate file to be the *FME feature type* of a Generate feature. The feature type of a Generate feature must match the basename of a Generate file defined by a Generate `DEF` line. Each feature read from a Generate file has an `ARCGEN_GEOMETRY` attribute added to it that indicates if the feature came from an `arcgen_point`, `arcgen_line`, or `arcgen_text` file. The writer can also handle homogeneous aggregate features of points, lines or text, which also have an `ARCGEN_GEOMETRY` attribute.

## Points

Generate files containing only points consist of a series of lines that follow this syntax:

```
<idNumber>, <x>, <y> [, <z>]
```

---

**Tip:** By using the `idNumber` associated with each Generate feature as a key into a comma separated value file, the `@Relate` command can be used to attach attributes to Generate features.

---

The `<idNumber>` is any numeric value, and need not be unique within a file. As well, the `<z>` value is optional and, if it is not specified, the point is considered to be two-dimensional. The file is terminated by a line containing only the word `END`. A two-dimensional point Generate file example is shown below:

```
601, 3, 7
602, 53, 21
603, 19, 20
END
```

## Lines

Generate files containing only linear features consist of a series of lines that follow this syntax:

```
<idNumber>
<x0>, <y0> [, <z0>]
<x1>, <y1> [, <z1>]
...
<xN>, <yN> [, <zN>]
END
```

---

**Tip:** Polygonal features may also be input and output using linear Generate files. In such cases, the first point and the last point of the *line* are identical.

---

As with point files, the <idNumber> is any numeric value, and need not be unique within a file. As well, the <z> value is optional and, if it is not specified, the linear feature is considered to be two-dimensional. The end of each linear feature is marked by a line containing only the word `END`. A linear Generate file is terminated with two consecutive lines containing only the word `END`. A three-dimensional linear Generate file example, containing two features, is shown below:

```
101
32, 52, 1
33, 56, 2
36, 59, 6
31, 70, 3
END
102
52, 32, 3
53, 56, 5
56, 29, 1
61, 73, 14
END
END
```

## Text

Generate files containing only text (annotation) features, consist of a series of lines that follow this syntax:

```
<idNumber>, <x>, <y>, <angle>, <size>, <text>
```

As with point files, the <idNumber> is any numeric value and need not be unique within a file. A Text generate file is terminated with the word `END`. A text Generate file example, containing two features, is shown below:

```
101, 32, 52, 0, 20, Arnet Maves
102, 52, 32, 90, 30, Barnie Maves
END
```

The attributes used by the generate reader and writer are described in the following table.

Attribute Name	Value
arcgen_rotation	Specifies the rotation of the text in degrees measured counterclockwise from horizontal. <b>Range:</b> -360.0 . . . 360.0
arcgen_text_size	The size of the annotation in ground units. <b>Range:</b> Float > 0
arcgen_text_string	The text string to be placed at the annotation location. <b>Range:</b> Any text string

The example below shows an FME mapping file used to translate some points and linear features from the Generate format into Shape files. The mapping file defines the dataset location and gives the Generate definitions for the two files to be read. At run time, the Generate reader goes out to the directory, reads the files, and produces an FME feature for each Generate feature it finds.

## Example

Sample Generate to Shape Mapping File:

```
# -----
# Define the location of the Generate files
ARCGEN_DATASET /usr/data/generate/92i080
# -----
# Define the type of each of the Generate files
ARCGEN_DEF nodes          ARCGEN_GEOMETRY arcgen_point

# -----
# This second file uses a space as the delimiter
ARCGEN_DEF boundaries ARCGEN_GEOMETRY arcgen_line          \
                    ARCGEN_DELIMITER " "

# -----
# Now define the location of the Shape files
# which will be created
SHAPE_DATASET /usr/data/shape/92i080
# -----
# Define each of the Shape files.

SHAPE_DEF markers SHAPE_GEOMETRY shape_point          \
                    MARKER_ID number(6,0)
SHAPE_DEF edges   SHAPE_GEOMETRY shape_polyline       \
                    EDGE_ID number(6,0)

# -----
# Now define transfer specifications
ARCGEN nodes arcgen_id %nodeNum
SHAPE markers MARK_ID %nodeNum
ARCGEN boundaries arcgen_id %boundNum
SHAPE edges   EDGE_ID %boundNum
```

---

**Tip:** Notice the Shape file definitions create a field to store the identifier associated with each generate feature.

---

If the /usr/data/generate/92i080 directory contained the following files:

nodes.gen	boundaries.gen
601,7,7	101
602,53,21	32 52 1
603,19,20	33 56 2
END	36 59 6
	31 70 3
	END
	102
	52 32 3
	53 56 5
	56 29 1
	61 73 14
	END
	END

...then the FME features shown below would be created by the Generate reader.



Feature Type: nodes	
Attribute Name	Value
ARCGEN_GEOMETRY	arcgen_point
arcgen_id	601
Coordinates: 37,7	



Feature Type: nodes	
Attribute Name	Value
ARCGEN_GEOMETRY	arcgen_point
arcgen_id	602
Coordinates: 53,21	



Feature Type: nodes	
Attribute Name	Value
ARCGEN_GEOMETRY	arcgen_point
arcgen_id	603
Coordinates: 19,20	



Feature Type: boundaries	
Attribute Name	Value
ARCGEN_GEOMETRY	arcgen_line




---

**Feature Type: boundaries**


---

Attribute Name	Value
arcgen_id	101
Coordinates: (32,52,1),(33,56,2),(36,59,6),(31,70,3)	

---




---

**Feature Type: boundaries**


---

Attribute Name	Value
ARCGEN_GEOMETRY	arcgen_line
arcgen_id	102
Coordinates: (52,32,3),(53,56,5),(56,29,1),(61,73,14)	

---




---

**Feature Type: boundaries**


---

Attribute Name	Value
ARCGEN_GEOMETRY	arcgen_line
arcgen_id	101
Coordinates: (32,52,1),(33,56,2),(36,59,6),(31,70,3)	

---




---

**Feature Type: boundaries**


---

Attribute Name	Value
ARCGEN_GEOMETRY	arcgen_line
arcgen_id	102
Coordinates: (52,32,3),(53,56,5),(56,29,1),(61,73,14)	

---

These features would then be transformed by the FME and output to their destination Shape files by the Shape writer.