

# GML v2.1.2 (Geography Markup Language) Reader/Writer

The GML2 modules enables FME to read and write files in Geography Markup Language (GML) format.

This chapter assumes familiarity with GML.

## Additional GML Formats

Note that the following formats are GML formats, and their documentation is considered part of the GML Reader/Writer documentation:

| Format            | Reader/Writer |
|-------------------|---------------|
| OS (GB) MasterMap | Reader        |

## Overview

GML is an OpenGIS<sup>®</sup> Implementation Specification. The GML specification defines an XML encoding for the transport and storage of geographic information. This specification can be found at the Open GIS Consortium website [www.opengeospatial.org](http://www.opengeospatial.org).

GML documents must be instances of a conforming application schema. Conforming application schemas are to be defined with the W3C's XML Schema language.

## GML2 Quick Facts

|                               |  |
|-------------------------------|--|
| Format Type Identifier        | GML2   |
| Reader/Writer                 | Reader/Writer                                  |
| Licensing Level               | Professional                                   |
| Dependencies                  | None   |
| Dataset Type                  | File   |
| Feature Type                  | Varies depending on the GML application schema |
| Typical File Extensions       | .gml,.xml                                      |
| Automated Translation Support | Yes  |
| User-Defined Attributes       | Yes  |
| Coordinate System Support     | Yes  |
| Generic Color Support         | No   |
| Spatial Index                 | Never  |
| Schema Required               | Yes  |
| Transaction Support           | No   |
| Geometry Type                 | xml_type                                       |

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

## Reader Overview

### Reader Directives

The suffixes listed are prefixed by the current <ReaderKeyword> in a mapping file. By default, the <ReaderKeyword> for the GML2 reader is GML2.

#### DATASET

**Required/Optional:** *Required*

This directive specifies the location for the input GML instance document.

#### Example:

```
GMLSF_DATASET c:\gml_data\hydro.xml
```

## CACHE\_XSD

**Required/Optional:** *Optional*

This directive allows the XML Schema documents that are fetched from the internet to be cached locally, this reduces the number of network fetches when traversing the GML schema documents. The valid values of this directive are YES and NO, its default value is YES.

**Example:**

```
GML2_CACHE_XSD NO
```

## CACHE\_XSD\_EXPIRY\_TIME

**Required/Optional:** *Optional*

This directive is optional and takes effect only if the CACHE\_XSD directive is set to YES. The valid values for this directive are positive numbers denoting the number of seconds. The default value for this directive is 300.

**Example:**

```
GML2_CACHE_XSD_EXPIRY_TIME 600
```

## CACHE\_XSD\_DIRECTORY

**Required/Optional:** *Optional*

This optional directive takes effect when CACHE\_XSD directive is set to YES. The directive specifies the directory path for the location of the cache xsd directory, the directory name for the cache xsd directory is specified by the CACHE\_XSD\_NAME directive below. The default value for this directive is the user's temporary directory.

**Example:**

```
GML2_CACHE_XSD_DIRECTORY c:\tmp
```

## CACHE\_XSD\_NAME

**Required/Optional:** *Optional*

This optional directive specifies the xsd cache name. The cache name must also be a valid directory name, as this value is used as the subdirectory containing the cached schema documents within the CACHE\_XSD\_DIRECTORY. The default value for this directive is FME\_XSD\_CACHE.

**Example:**

```
GML2_CACHE_XSD_NAME gml_schema_cache
```

## DOCUMENT\_STREAM

**Required/Optional:** *Optional*

The entire GML document can be specified as the string value of this directive. This directive is optional and it overrides the DATASET directive if present.

## FEATURE\_ENCODING

**Required/Optional:** *Optional*

This directive specifies the encoding for the FME features. The default value for the directive is the system's encoding.

**Example:**

```
GML2_FEATURE_ENCODING Shift-JIS
```

## HTTP\_PROXY

**Required/Optional:** *Optional*

This directive specifies the HTTP proxy to be used for network fetches. The port number may be specified at the end of the proxy by appending `:[port number]` or through the `HTTP_PROXY_PORT` directive.

**Example:**

```
GML2_HTTP_PROXY www.someproxy.net
```

or

```
GML2_HTTP_PROXY www.someproxy.net:8081
```

---

**Note:** Users may bypass the `HTTP_PROXY` and `HTTP_PROXY_PORT` directives and still have http proxy support by specifying the `http_proxy` environment variable. The value for this environment variable should be of the form `[protocol://][user:password@]machine[:port]`, where components within `[]` are optional. An example value for the `http_proxy` environment variable is: `http://www.someproxy.net:8081`.

---

## HTTP\_PROXY\_PORT

**Required/Optional:** *Optional*

This directive is used if the HTTP proxy port was not specified in the `HTTP_PROXY` directive.

**Example:**

```
GML2_HTTP_PROXY_PORT 8081
```

## MAPPING\_FILE\_ENCODING

**Required/Optional:** *Optional*

This optional directive specifies the encoding for the FME mapping file. The default value for the directive is the system's encoding.

**Example:**

```
GML2_MAPPING_FILE_ENCODING ISO-8859-3
```

## READ\_DEFAULT\_GML\_PROPERTIES

**Required/Optional:** *Optional*

This directive specifies whether the default and optional GML feature properties, *fid*, *name*, and *description*, should be read. The valid values of this directive are **YES** and **NO**; its default value is **NO**.

### **XFMAP**

**Required/Optional:** *Optional*

This directive is not for general usage.

Rather than having the GML2 reader examine the GML application schema, it directs it to read the input dataset document with the specified *xfMap*. Alternatively, multiple *xfMaps* may be specified in a single value quoted **XFMAP** directive by separating each *xfMap* path with a semicolon. See the *XML Reader/Writer* chapter for a description of the *xfMap*.

#### **Example:**

```
GML2_XFMAP C:\tmp\data\features.xmp
```

or

```
GML2_XFMAP "C:\tmp\drainages.xmp;C:\tmp\pits_pipes.xmp"
```

or

```
GML2_XFMAP C:\tmp\drainages.xmp
```

```
GML2_XFMAP C:\tmp\pits_pipes.xmp
```

### **XFMAP\_STREAM**

**Required/Optional:** *Optional*

This directive is not for general usage.

Rather than having the GML2 reader examine the GML application schema, it directs it to read the input dataset document with the *xfMap* specified as the string value of this directive. See the *XML Reader/Writer* chapter for a description of the *xfMap*.

### **XFMAP\_SCHEMA**

**Required/Optional:** *Optional*

This directive is not for general usage.

Rather than having the GML2 reader examine the GML application schema, it directs it to read schema features from the input dataset document with the specified *xfMap*. Alternatively, multiple *xfMaps* may be specified in a single value quoted **XFMAP** directive by separating each *xfMap* path with a semicolon. See the *XML Reader/Writer* chapter for a description of the *xfMap*.

#### **Example:**

```
GML2_XFMAP C:\tmp\data\schema_features.xmp
```

or

```
GML2_XFMAP "C:\s_drainages.xmp;C:s_pits_pipes.xmp"
```

or

```
GML2_XFMAP C:\tmp\schema_drainages.xmp  
GML2_XFMAP C:\tmp\schema_pits_pipes.xmp
```

## XSD\_DOC

**Required/Optional:** *Optional*

A GML instance document specifies the namespace and the location of its application schema through its root element `xsi:schemaLocation` attribute. This directive allows the GML2 reader to use a different GML schema document from the one specified in the `xsi:schemaLocation` attribute.

The XML Schema specification states that the `xsi:schemaLocation` attribute value consists of a set of pairs: The first member each pair is the namespace for which the second member is the hint describing where to find an appropriate schema document. The presence of this hint does not require the processor to obtain or use the cited schema document, however, the processor is free to use other schemas obtained by other suitable means.

The `XSD_DOC` directive allows the usage of other schema documents on the instance besides the one stated in the instance's `xsi:schemaLocation` attribute.

---

**Note:** This directive only takes effect if the target namespace of the dataset is not in the Safe schema namespace `http://www.safe.com/xml/schemas/FMEFeatures`. The GML2 writer in `FIXED SCHEMA_MODE` writes out documents that belong to that namespace.

---

## Writer Overview

The GML2 writer currently writes out GML instance documents in two different modes. The mode of operation is controlled by the value of the `SCHEMA_MODE` directive:

- **FIXED:** In this mode, the GML2 writer produces instance documents that conform to the `FMEFeatures.xsd` schema. The `FMEFeatures.xsd` file can be found under the `{FME installed dir}/xfMap` directory.
- **CREATE:** In this mode, the GML2 writer creates two GML documents – a GML application schema and a GML instance document that conforms to this schema.
- **SCAN:** This mode is not yet implemented.

The character encoding for the output XML documents may also be specified with the `OUTPUT_ENCODING` directive. When this is not specified, the output encoding defaults to UTF-8.

## Writer Directives

The directives processed by the GML2 Writer are listed below. The suffixes shown are prefixed by the current `<WriterKeyword>` in a mapping file. By default, the `<WriterKeyword>` for the GML2 writer is `GML2`.

### DATASET

**Required/Optional:** *Required*

This directive specifies the location for the output GML instance document.

**Example:**

```
GMLSF_DATASET c:\gml\data.xml
```

**SCHEMA\_MODE****Required/Optional:** *Optional*

This controls the operation mode of the GML2 writer. The valid values are:

- **FIXED:** In this mode, the GML2 writer produces instance documents that conform to the `FMEFeatures.xsd` schema. The `FMEFeatures.xsd` file is under the `{FME installed dir}/xfMap` directory.
- **CREATE:** The GML2 writer creates two GML documents – a GML application schema and a GML instant document that conforms to this schema. In this mode, the GML2 writer `DEF` lines control the contents and format of the GML application schema. A third XML document may also be created. This document can be used by the XML Reader to read back into FME the GML instances that are created in this mode. The `xfMap` document is created in the same directory as the one that is specified with the `DATASET` directive. The document's filename extension is `.xmp` and its filename basename is the same as that specified in `DATASET` (see the `SUPPRESS_XFMAP_OUTPUT` directive). The `XSD_DOC`, `TARGET_NS_PREFIX`, and `TARGET_NS_URI` can be used in conjunction when the writer is in this mode.
- **SCAN:** This mode is not yet implemented.

The default value for this directive is `FIXED`.

**SUPPRESS\_XSD\_DOC\_OUTPUT****Required/Optional:** *Optional*

In `CREATE` mode, it enables the suppression for the output of the schema document. The valid values for this directive are `YES` and `NO`, with `NO` being the default value.

**Example:**

```
GML2_SUPPRESS_XSD_DOC_OUTPUT YES
```

**TARGET\_NS\_PREFIX****Required/Optional:** *Optional*

This directive is used only in `CREATE` mode. The GML application schema must declare a target namespace. All elements declared in the schema will reside in this namespace. The directive allows the specification of the target namespace prefix.

The default value for this directive is `gml2`.

**Example:**

```
GML2_TARGET_NS_PREFIX geo
```

**TARGET\_NS\_URI****Required/Optional:** *Optional*

This directive is used only in the `CREATE` mode. The GML application schema must declare a target namespace. All elements declared in the schema will reside in this namespace. The directive allows the specification of the target namespace URI.

The default value for this directive is `http://www.safe.com/gml2`.

**Example:**

```
GML2_TARGET_NS_URI http://www.geomaps.com/geo
```

### TARGET\_XSI\_SCHEMA\_LOCATION\_URL

**Required/Optional:** *Optional*

This directive is used only in the `CREATE` mode. This directive specifies the URL schema location for the target namespace pair in the GML instance's `xsi:schemaLocation` attribute.

**Example:**

```
GML2_TARGET_XSI_SCHEMALOCATION_URL http://www.sch.org/g.xsd
```

### XSD\_DOC

**Required/Optional:** *Required when SCHEMA\_MODE is SCAN; optional otherwise*

It specifies the location for the GML application schema. This directive is not required in `FIXED` mode, is optional in `CREATE` mode, and is required in `SCAN` mode.

- `CREATE` mode: The `XSD_DOC` directive is optional in this mode. Specifies the location of the GML application schema that is generated. If the `XSD_DOC` is not specified, then the GML application schema is generated in the directory that is specified by the `DATASET` directive.
- `SCAN` mode: The `XSD_DOC` directive is required in this mode. Given a GML application schema, the GML2 writer will output a conforming GML instance document.

### OUTPUT\_ENCODING

**Required/Optional:** *Optional*

This directive specifies the encoding that is to be used for the XML documents that GML2 writer outputs. The default value for this directive is `UTF-8`.

**Example:**

```
GML2_OUTPUT_ENCODING Shift-JIS
```

### FEATURE\_ENCODING

**Required/Optional:** *Optional*

This directive specifies the encoding for the FME feature data. When not specified, the feature data is assumed to be encoded with the system's encoding.

**Example:**

```
GML2_FEATURE_ENCODING Shift-JIS
```

## MAPPING\_FILE\_ENCODING

**Required/Optional:** *Optional*

This directive specifies the encoding of the FME mapping file. When not specified, the FME mapping file is assumed to be encoded in the system's encoding.

**Example:**

```
GML2_MAPPING_FILE_ENCODING ISO-8859-3
```

## USE\_WFS\_FEATURE\_COLLECTION

**Required/Optional:** *Optional*

Setting this directive to YES changes the root element in the output document `<wfs:FeatureCollection>`. The default value for this directive is NO.

**Example:**

```
GML2_USE_WFS_FEATURE_COLLECTION YES
```

## APPLY\_STYLESHEET

**Required/Optional:** *Optional*

This directive allows an XSLT stylesheet to be applied to the final output DATASET document. The `STYLESHEET_RESULT` directive may be used in conjunction with this directive to specify the location and filename of the resulting transformation. There are no default values for this directive.

**Example:**

```
GML2_APPLY_STYLESHEET c:\data\myTransform.xml
```

## STYLESHEET\_RESULT

**Required/Optional:** *Optional*

This directive only takes effect if `APPLY_STYLESHEET` is specified. When this directive is not present or its value is the empty string, then the resulting XSLT transformation will have the same location and filename as the output DATASET with the exception that the filename will be prefixed with "transformed\_".

**Example:**

```
GML2_STYLESHEET_RESULT c:\data\myTransformedDoc.xml
```

## SUPPRESS\_NULL\_ATTRS

**Required/Optional:** *Optional*

This directive suppresses printing of the attributes which are specified in the `DEF` lines but are not present in the FME features that are passed into the GML writer. The valid values for this directive are YES and NO. The default value is NO.

**Example:**

```
GML2_SUPPRESS_NULL_ATTRS YES
```

**XFMAP****Required/Optional:** *Optional*

In `CREATE` mode, the GML2 writer creates a third document, which is an `xfMap` that can be used by the XML reader to read back the GML documents output. This is an optional directive, and if it is not specified then the output `xfMap` will have the same filename as the specified output dataset but having an `.xmp` file extension. See the *XML Reader/Writer* chapter for a description of the `xfMap`.

**Example:**

```
GML2_XFMAP C:\tmp\test.xmp
```

**SUPPRESS\_XFMAP\_OUTPUT****Required/Optional:** *Optional*

In `CREATE` mode, this optional directive can be used to suppress the creation of the `xfMap` document. The valid values for this directive are `YES` and `NO`, with `YES` being its default value.

**Example:**

```
GML2_SUPPRESS_XFMAP_OUTPUT NO
```

**USE\_STYLESHEET\_RESULT\_AS\_DATASET****Required/Optional:** *Optional*

This directive is optional, but it only takes effect when it is used in conjunction with the `APPLY_STYLESHEET` directive. The valid values for this directive are `YES` and `NO`, with `NO` being its default value. When this directive is set to `YES`, then the file specified under the `DATASET` directive will be the resulting dataset after the stylesheet transformation; the original GML dataset will be written into a temporary file and deleted after the stylesheet transform. Additionally, the `SUPPRESS_XFMAP_OUTPUT` and the `SUPPRESS_XSD_DOC_OUTPUT` directive will be automatically set to `YES`, as the schema and `xfMap` document will possibly not conform to the output dataset after the transform.

**Example:**

```
GML2_USE_STYLESHEET_RESULT_AS_DATASET YES
```

**FME\_COORDINATE\_SYSTEM\_AS\_PROPERTY****Required/Optional:** *Optional*

This directive only takes effect in `CREATE` mode. The valid values for this directive are `YES` and `NO` with the default value being `NO`. If this directive is set to `YES`, then the GML2 writer will extract the FME coordinate system from a feature into a user-defined property named `fme_coordinate_system`, i.e., the application schema generated will con-

tain for each feature an extra property named `fme_coordinate_system` whose value will be the FME coordinate system short name.

**Example:**

```
GML2_FME_COORDINATE_SYSTEM_AS_PROPERTY YES
```

**LISTS\_IN\_MULTI\_VALUE\_ATTRIBUTES**

**Required/Optional:** *Optional*

This optional directive allows the simple list attributes, the ones specified in the GML2 `DEF` lines with a “{}” suffix, to be carried in the FME features a multi-value attributes. The valid values for this directive are `YES` and `NO`, with the default value being `YES`.

When this directive is set to `YES`, the GML2 writer will not expect the FME features to contain simple list attributes, instead every attribute that is defined in the `DEF` lines with a {} suffix is expected to be a single attribute such that its value is a token separated lists of values, a multi-value attribute.

**Example:**

```
GML2_LISTS_IN_MULTI_VALUE_ATTRIBUTES YES
```

**MULTI\_VALUE\_SEPARATOR**

**Required/Optional:** *Optional*

This optional directive is used when the `LISTS_IN_MULTI_VALUE_ATTRIBUTES` is set to `YES`. It specifies the token separator used for the multi-value attributes that are to be treated as simple lists. The default value for this directive is a comma, “,”.

**Example:**

```
GML2_MULTI_VALUE_SEPARATOR |
```

**SUPPRESS\_SCHEMA\_LOCATION\_ATTR**

**Required/Optional:** *Optional*

Setting this directive to `YES` suppresses the output of the `xsi:schemaLocation` attribute in the GML instance’s root element. The `xsi:schemaLocation` in an XML document instance is not a mandatory attribute – it is merely a hint which an XML processor may choose to ignore. Setting this keyword to `YES` suppresses the output of the `xsi:schemaLocation` attribute in the output GML instance. The default value for this directive is `NO`.

**Example:**

```
GML2_SUPPRESS_SCHEMA_LOCATION_ATTR YES
```

## DEF Lines

The `DEF` line controls the generation of the GML application schema when the GML2 writer is set to `FIXED` or `CREATE` mode.

The syntax of a GML2 `DEF` line is:

```
<WriterKeyword>_DEF <elementName>           \  
    xml_type<xml_type>                       \  
    [<attrName><attrType>]*
```

The interpretation of a `DEF` line is dependent on the GML2 writer's `SCHEMA_MODE`.

## FIXED Mode

The GML instance document output conforms to the `FMEFeatures.xsd` application schema (Safe Schema). Its namespace prefix is `fme` and its URL is `http://www.safe.com/xml/schemas/FMEFeatures`. A copy of the `FMEFeatures.xsd` schema document may be found under the `xfMap` directory of the FME installation.

This schema declares two GML feature collection elements – the `fme:schemaFeatures` and `fme:dataFeatures` collection elements. (Please refer to the `FMEFeatures.xsd` schema, in the `xfMap` directory of your FME installation, for the declaration of the GML elements that are in the `fme` namespace.) Each of the `fme` collection elements may contain 1 or more `fme:Feature` elements.

The `fme:Feature` can have a feature type, zero or more properties, and optionally one of the predefined geometric properties defined by the GML specification. (See the `fme:FeatureType` complex type definition in the `FMEFeatures.xsd` schema document.)

A `DEF` line specifies the contents of an `fme:Feature` in the `fme:schemaFeatures` container element. The `<elementName>` defines its feature type. Each `<attrName>` `<attrType>` pair defines an `fme:property` element. An additional `fme:property` with the name `xml_type` is defined by the value of the `<xml_type>`.

For example, the `DEF` line:

```
GML2_DEF Mexico                               \  
    xml_type          xml_area                \  
    State_Name        xml_char(35)           \  
    State_Code        xml_char(8)
```

causes the GML2 writer to generate, in the GML output instance, the following `fme:Feature`:

```
<fme:schemaFeatures>  
    ...  
    <gml:featureMember>  
        <fme:Feature>  
            <fme:featureType>Mexico</fme:featureType>  
            <fme:property name="xml_type">xml_area</fme:property>  
            <fme:property name="State_Name">xml_char(35)</fme:property>  
            <fme:property name="State_Code">xml_char(8)</fme:property>  
        </fme:Feature>  
    </gml:featureMember>
```

```

...
</fme:schemaFeatures>

```

When the GML2 Writer writes out an FME feature, it uses its feature type to determine how an `fme:Feature` element should be written in the `fme:dataFeature` container. The FME feature's feature type must equal one of the DEF lines `<elementName>` - through it, the writer determines the properties and geometry for the data `fme:Feature`. The geometry is determined by the DEF line's `<xml_type>`. It determines which GML geometric primitive is needed in the `fme:Feature`. The following table shows the correspondence:

| <b>xml_type</b> | <b>GML v2 geometric property element referenced in the fme:Feature</b> |
|-----------------|--|
| xml_no_geom     | none   |
| xml_point       | pointProperty, multiPointProperty                                      |
| xml_line        | lineStringProperty, multiLineStringProperty                            |
| xml_area        | polygonProperty, multiPolygonProperty                                  |
| xml_text        | pointProperty  |

## CREATE Mode

One GML2 DEF line specifies the following in the GML application schema:

- XML Schema Complex Type Definition
- XML Schema Global Element Declaration

### XML Schema Complex Type Definition

This complex type is named `<elementName>Type`. It resides in the GML application schema target namespace that is specified with the `TARGET_NS_PREFIX` and `TARGET_NS_URI` directives, and it derives by extension from the GML `AbstractFeatureType`.

The value of `<xml_type>` determines which GML geometric property will be included as an element reference in the definition of the `<elementName>Type` complex type. The following table shows this relation:

| <b>xml_type</b> | <b>GML v2 geometric property element referenced in the complex type</b> |
|-----------------|---|
| xml_no_geom     | none  |
| xml_point       | pointProperty, multiPointProperty                                       |
| xml_line        | lineStringProperty, multiLineStringProperty                             |
| xml_area        | polygonProperty, multiPolygonProperty                                   |
| xml_text        | currently not supported in CREATE mode                                  |

In the above table, if there is more than one GML geometric property element referenced, then the `<elementName>Type` complex type definition will reference these inside an XML Schema choice element.

Each `<attrName>` `<attrType>` in the DEF line specifies an element declaration in the `<elementName>Type` complex type. The element declaration name is `<attrName>` and its type is `<attrType>`. The valid values for `<attrType>` are: `xml_char(width)`, `xml_int32`, `xml_real32`, `xml_decimal(width,decimal)`, `xml_boolean`, and `xml_real64`.

An `<attrName>` may also be suffixed in the DEF line by `{ }` to indicate that the attribute is a simple list attribute. The GML property generated for this simple list attribute will have its `maxOccurs` set to `unbounded`.

### XML Schema Global Element Declaration

This global element is declared with the name `<elementName>`, its type is `<elementName>Type`, and it is assigned to the `GML_Feature` substitution group.

Example:

The following GML2 DEF line:

```
GML2_DEF roads                                     \
  xml_type xml_line                                \
  id      xml_char(4)                               \
  name    xml_char(16)                             \
  myList {}xml_char(64)
```

causes the GML2 Writer to generate the following for the GML2 application schema (assume that the GML application schema target namespace prefix is `gml2`, that the GML specification namespace prefix is `gml`, and that the XML Schema is in the default namespace [i.e., no prefix is required for the XML Schema elements]):

```
<complexType name="roadsType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element name="id">
          <simpleType>
            <restriction base="string">
              <maxLength value="4"/>
            </restriction>
          </simpleType>
        </element>
        <element name="name">
          <simpleType>
            <restriction base="string">
              <maxLength value="16"/>
            </restriction>
          </simpleType>
        </element>
        <element name="myList" maxOccurs="unbounded">
          <simpleType>
```

```

        <restriction base="string">
            <maxLength value="64"/>
        </restriction>
    </simpleType>
</element>
<choice>
    <element ref="gml:lineStringProperty"/>
    <element ref="gml:multiLineStringProperty"/>
</choice>
</sequence>
</extension>
</complexContent>
</complexType>

<!-- XML Schema global element declaration -->
<element name="roads" type="gml2:roadsType"
    substitutionGroup="gml:_Feature"/>

```

## Feature Representation

In addition to the generic FME feature attributes that FME Workbench adds to all features (see *About Feature Attributes* on page 7), this format adds the format-specific attributes described in this section.

The geometry of FME GML features may be identified by its `xml_type` attribute. The valid values for this attribute are:

| <b>xml_type</b>          | <b>Description</b>             |
|--------------------------|--------------------------------|
| <code>xml_no_geom</code> | FME Feature with no geometry.  |
| <code>xml_point</code>   | Point feature.                 |
| <code>xml_line</code>    | Linear feature.                |
| <code>xml_area</code>    | Areal feature, may be a donut. |
| <code>xml_text</code>    | Text feature.                  |

Other attributes, including the feature's feature type, are dependent on the mappings that are in an `xfMap` document.

### No Geometry

**xml\_type:** `xml_no_geom`

Features having their `xml_type` attribute set to `xml_no_geom` do not contain any geometry data.

### Points

**xml\_type:** `xml_point`

Features having their `xml_type` set to `xml_point` are single coordinate features.

## Lines

**xml\_type:** xml\_line

Features having their `xml_type` set to `xml_line` are polyline features and have at least two coordinates.

## Area

**xml\_type:** xml\_area

Features having their `xml_type` set to `xml_area` are either a single closed polyline feature (simple closed polygon), a donut, or an aggregate of donuts (and/or simple polygons). A simple closed polygon contains at least four coordinates, with the first and last coordinate being equal.

## Text

**xml\_type:** xml\_text

These are annotation features.