

Geography Markup Language

From Wikipedia, the free encyclopedia

The **Geography Markup Language (GML)** is the XML grammar defined by the Open Geospatial Consortium (OGC) to express geographical features. GML serves as a modeling language for geographic systems as well as an open interchange format for geographic transactions on the Internet. Note that the concept of feature in GML is a very general one and includes not only conventional "vector" or discrete objects, but also coverages (see also GMLJP2) and sensor data. The ability to integrate all forms of geographic information is key to the utility of GML.

Contents

- 1 Standards
- 2 GML model
 - 2.1 Profile
 - 2.1.1 GML Simple Features Profile
 - 2.1.2 Subset tool
 - 2.2 Application schema
 - 2.3 GML and KML
 - 2.4 GML geometries
 - 2.5 Features
 - 2.6 Coordinates
 - 2.7 Coordinate Reference System
- 3 Examples
 - 3.1 Features using geometries
 - 3.2 Point Profile
- 4 See also
- 5 References

Standards

The OGC is an international voluntary consensus standards organization whose members maintain the *Geography Markup Language* standard. The OGC coordinates with the ISO TC 211 standards organization to maintain consistency between OGC and ISO standards work. GML is in the process of being adopted as an ISO standard (ISO 19136) and is expected to be released as an International Standard (<http://www.isotc211.org>) in 2007.

GML is the XML data standard for the GeoWeb (<http://www.geoweb.org/>) infrastructure, enabling Internet-connected devices to access geographical information, including, for example, merchant locations and traffic conditions.

GML can also be included in version 1.0 of the United States National Information Exchange Model.

GML model

The original GML model was based on the World Wide Web Consortium's Resource Description Framework (RDF). Subsequently, the OGC introduced XML schemas into GML's structure to help connect the various existing geographic databases, whose relational structure XML schemas more easily define. The resulting XML-schema-based GML retains many features of RDF, including the idea of child elements as properties of the parent object (RDFS) and the use of remote property references.

GML contains a rich set of primitives which are used to build application specific schemas or application languages. These primitives include:

- Feature
- Geometry
- Coordinate Reference System
- Time
- Dynamic feature
- Coverage (including geographic images)

- Unit of measure
- Map presentation styling rules

Profile

GML profiles are logical restrictions to GML, and may be expressed by a document, an XML schema or both. These profiles are intended to simplify adoption of GML, to facilitate rapid adoption of the standard. The following *profiles*, as defined by the GML specification, have been published or proposed for public use:

- A *Point Profile* for applications with point geometric data but without the need for the full GML grammar
- A *GML Simple Features profile* supporting vector feature requests and transactions, e.g. with a WFS
- A GML profile for GMJP2 (GML in JPEG 2000)
- A GML profile for RSS

Note that *Profiles* are distinct from application schemas. *Profiles* are part of GML namespaces (Open GIS GML (<http://www.opengis.net/gml>)) and define restricted subsets of GML. Application schemas are XML vocabularies defined using GML and which live in an application-defined target namespace. Application schemas can be built on specific GML profiles or use the full GML schema set.

Profiles are often created in support for GML derived languages (see application schemas) created in support of particular application domains such as commercial aviation, nautical charting or resource exploitation.

The GML Specification (Since GML v3.) contains a pair of XSLT scripts (usually referred to as the "subset tool") that can be used to construct GML profiles.

GML Simple Features Profile

The *GML Simple Features Profile* is a more complete profile of GML than the above *Point Profile* and supports a wide range of vector feature objects, including the following:

1. A reduced geometry model allowing 0d, 1d and 2d linear geometric objects (all based on linear interpolation) and the corresponding aggregate geometries (gml:MultiPoint, gml:MultiCurve, etc).
2. A simplified feature model which can only be one level deep (in the general GML model, arbitrary nesting of features and feature properties is not permitted).
3. All non-geometric properties must be XML Schema simple types – i.e. cannot contain nested elements.
4. Remote property value references (xlink:href) just like in the main GML specification.

Since the profile aims to provide a simple entry point, it does not provide support for the following:

- coverages
- topology
- observations
- value objects (for real time sensor data)
- nor support for dynamic features.

Nonetheless it supports a good variety of real world problems.

Subset tool

In addition, the GML specification provides a *subset tool* to generate GML profiles containing a user-specified list of components. The tool consists of a pair of XSLT scripts. The scripts generate a profile that a developer may extend manually or otherwise enhance through schema restriction. Note that as restrictions of the full GML specification, application schemas that a profile can generate must themselves be valid GML application schemas.

The subset tool can generate profiles for many other reasons as well. Listing the elements and attributes to include in the resultant profile schema and running the tool results in a single profile schema file containing only the user-specified items and all of the element, attribute and type declarations on which the specified items depend. Some Profile schemas created in this manner support other specifications including IHO S-57 and GML in JPEG 2000.

Application schema

In order to expose an application's geographic data with GML, a community or organization creates an XML schema specific to the application domain of interest (the *application schema*). This schema describes the object types whose data the community is interested in and which community applications must expose. For example, an application for tourism may define object types including monuments,

places of interest, museums, road exits, and viewpoints in its *application schema*. Those object types in turn reference the primitive object types defined in the GML standard.

A list of known publicly available GML Application Schemas is being assembled.

Some other markup languages for geography use schema constructs, but GML builds on the existing XML schema model instead of creating a new schema language.

GML and KML

Note that the KML language made popular by Google is a complementary to GML. Whereas GML is a language to encode geographic content, by describing a spectrum of application objects and their properties (e.g. bridges, roads, buoys, vehicles etc), KML is a language for the visualization of geographic information. KML can be used to carry GML content, and GML can be "styled" to KML for the purposes of presentation.

GML geometries

GML encodes the *GML geometries*, or *geometric characteristics*, of geographic objects as elements within GML documents. The geometries of those objects may describe, for example, roads, rivers, and bridges.

The key GML geometry object types in GML 1.0 and GML 2.0, are the following:

- Point
- LineString
- Polygon

Note that this geometry model is identical to the geometry model in KML.

Features

GML defines *features* distinct from *geometry objects*. A *feature* is an application object that represents a physical entity, e.g. a building, a river, or a person. A *feature* may or may not have geometric aspects. A *geometry object* defines a location or region instead of a physical entity, and hence is different from a *feature*. The distinction between *features* and *geometry objects* in GML contrasts with models used in other geographic information systems (GIS) that make no such distinction. That is, although some other GIS define *features* and *geometry objects* interchangeably as items on a map, GML maintains them as separate entity types.

In GML, a *feature* can have various geometry properties that describe geometric aspects or characteristics of the feature (e.g. the feature's *Point* or *Extent* properties). GML also provides the ability for *features* to share a geometry property with one another by using a *remote property reference* on the shared geometry property. Remote properties are a general feature of GML borrowed from RDF. An *xlink:href* attribute on a GML geometry property means that the value of the property is the resource referenced in the link.

For example, a *Building* feature in a particular GML application schema might have a position given by the primitive GML geometry object type *Point*. However, the *Building* is a separate entity from the *Point* that defines its position. In addition, a *feature* may have several geometry properties (or none at all), for example an *extent* and a *position*.

Coordinates

Coordinates in GML represent the coordinates of *geometry objects*. Coordinates can be specified by any of the following GML elements:

- `<gml:coordinates>`
- `<gml:pos>`
- `<gml:posList>`

GML has multiple ways to represent coordinates. For example, the `<gml:coordinates>` element can be used, as follows:

```
<gml:Point gml:id="p21" srsName="urn:ogc:def:crs:EPSG:6.6:4326">
  <gml:coordinates>45.67, 88.56</gml:coordinates>
</gml:Point>
```

Note that, when expressed as above, the individual coordinates (e.g. 88.56) are not separately accessible through the XML Document Object Model since the content of the `<gml:coordinates>` element is just a single string.

To make GML coordinates accessible through the XML DOM, GML 3.0 introduced the `<gml:pos>` and `<gml:posList>` elements. (Note

that although GML versions 1 and 2 had the `<gml:coord>` element, it is treated as a defect and is not used.) Using the `<gml:pos>` element instead of the `<gml:coordinates>` element, the same point can be represented as follows:

```
<gml:Point gml:id="p21" srsName="urn:ogc:def:crs:EPSG:6.6:4326">
  <gml:pos dimension="2">45.67 88.56</gml:pos>
</gml:Point>
```

The coordinates of a `<gml:LineString>` geometry object can be represented with the `<gml:coordinates>` element:

```
<gml:LineString gml:id="p21" srsName="urn:ogc:def:crs:EPSG:6.6:4326">
  <gml:coordinates>45.67, 88.56 55.56,89.44</gml:coordinates>
</gml:LineString >
```

The `<gml:posList>` element is used to represent a list of coordinate tuples, as required for linear geometries:

```
<gml:LineString gml:id="p21" srsName="urn:ogc:def:crs:EPSG:6.6:4326">
  <gml:posList dimension="2">45.67 88.56 55.56 89.44</gml:posList>
</gml:LineString >
```

For GML data servers (WFS) and conversion tools that only support GML 1 or GML 2 (i.e. only the `<gml:coordinates>` element), there is no alternative to `<gml:coordinates>`. For GML 3 documents and later, however, `<gml:pos>` and `<gml:posList>` are preferable to `<gml:coordinates>`.

For more information on the *srsName* attribute, see *Coordinate Reference System* below.

Coordinate Reference System

A **Coordinate Reference System** (CRS) determines the geometry of each geometry element in a GML document.

Unlike KML or GeoRSS, GML does not assume a single, fixed coordinate system. Instead, each coordinate system must be specified with a *Coordinate Reference System*. The elements whose coordinates are interpreted with respect to such a CRS include the following:

- `<gml:coordinates>`
- `<gml:pos>`
- `<gml:posList>`

An *srsName* attribute attached to a geometry object specifies the object's CRS, as shown in the following example:

```
<gml:Point gml:id="p1" srsName="#srs36">
  <gml:coordinates>100,200</gml:coordinates>
</gml:Point>
```

The value of the *srsName* attribute is a Uniform Resource Identifier (URI). It refers to a definition of the Coordinate Reference System that is used to interpret the coordinates in the geometry. The CRS definition may be in a document (i.e. a *flat file*) or in an online web service.

The *srsName* URI may also be a Uniform Resource Name (URN) for referencing a common CRS definition. The OGC has developed a URN structure and a set specific URNs to encode some common Coordinate Reference Systems. A URN resolver resolves those URNs to GML CRS definitions.

Examples

Polygons, Points, and *LineString* objects are encoded in GML 1.0 and 2.0 as follows:


```

<gml:Polygon>
  <gml:outerBoundaryIs>
    <gml:LinearRing>
      <gml:coordinates>0,0 100,0 100,100 0,100 0,0</gml:coordinates>
    </gml:LinearRing>
  </gml:outerBoundaryIs>
</gml:Polygon>
<gml:Point>
  <gml:coordinates>100,200</gml:coordinates>
</gml:Point>
<gml:LineString>
  <gml:coordinates>100,200 150,300</gml:coordinates>
</gml:LineString>

```

Note that *LineString* objects, along with *LinearRing* objects, assume linear interpolation between the specified points.

Features using geometries

The following GML example illustrates the distinction between *features* and *geometry objects*. The *Building* feature has several *geometry objects*, sharing one of them (the *Point* with identifier *p21*) with the *SurveyMonument* feature:

```

<abc:Building gml:id="SearsTower">
  <gml:name>Sears Tower</gml:name>
  <abc:height>52</abc:height>
  <abc:position>
    <gml:Point>
      <gml:coordinates>100,200</gml:coordinates>
    </gml:Point>
  </abc:position>
  <app:extent>
    <gml:Polygon>
      <gml:exterior>
        <gml:LinearRing>
          <gml:coordinates>100,200</gml:coordinates>
        </gml:LinearRing>
      </gml:exterior>
    </gml:Polygon>
  </app:extent>
</abc:Building>
<abc:Building gml:id="SearsTower">
  <abc:position xlink:type="Simple" xlink:href="#p21"/>
</abc:Building>
<abc:SurveyMonument gml:id="g234">
  <abc:position>
    <gml:Point gml:id="p21">
      <gml:coordinates>100,200</gml:coordinates>
    </gml:Point>
  </abc:position>
</abc:SurveyMonument>

```

Note that the reference is to the shared *Point* and not to the *SurveyMonument*, since any *feature* object can have more than one *geometry object* property.

Point Profile

The GML *Point Profile* contains a single GML geometry, namely a *<gml:Point>* object type. Any XML Schema can use the *Point Profile* by importing it and referencing the subject *<gml:Point>* instance:


```
<PhotoCollection xmlns="http://www.myphotos.org" xmlns:gml="http://www.opengis.net/gml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.myphotos.org
  MyGoodPhotos.xsd">
  <items>
    <Item>
      <name>Lynn Valley</name>
      <description>A shot of the falls from the suspension bridge</description>
      <where>North Vancouver</where>
      <position>
        <gml:Point srsDimension="2" srsName="urn:ogc:def:crs:EPSG:6.6:4326">
          <gml:pos>49.40 -123.26</gml:pos>
        </gml:Point>
      </position>
    </Item>
  </items>
</PhotoCollection>
```

Note that when using the *Point Profile*, the only geometry object is the '`<gml:Point>`' object. The rest of the geography is defined by the photo-collection schema.

See also

- GML Application Schemas
- CityGML
- Geographic Data Files (GDF)
- Well-known text

References

- GML 3.1 specification (http://portal.opengeospatial.org/files/?artifact_id=4700) (requires EULA to read)
- Digital Earth: GeoWeb (<http://www.dgeo.org/index.shtml>)
- GeoRSS - Geographically Encoded Objects for RSS Feeds (<http://www.georss.org>)
- *Recommended XMLGML encoding of common CRS definitions* (https://portal.opengeospatial.org/files/index.php?artifact_id=8837) , Open Geospatial Consortium
- *Demonstration of a Coordinate Reference System Registry* (<http://crs.opengis.org/crsportal>) , Open Geospatial Consortium
- C++ Data Binding for GML (<http://wiki.codesynthesis.com/Schemas/GML>)

Retrieved from "http://en.wikipedia.org/wiki/Geography_Markup_Language"

Categories: Markup languages | XML-based standards | Geographic Information Systems | GIS file formats

-
- This page was last modified 11:23, 29 May 2007.
 - All text is available under the terms of the GNU Free Documentation License. (See **Copyrights** for details.)
- Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a US-registered 501(c)(3) tax-deductible nonprofit charity.