# Geo tips & tricks

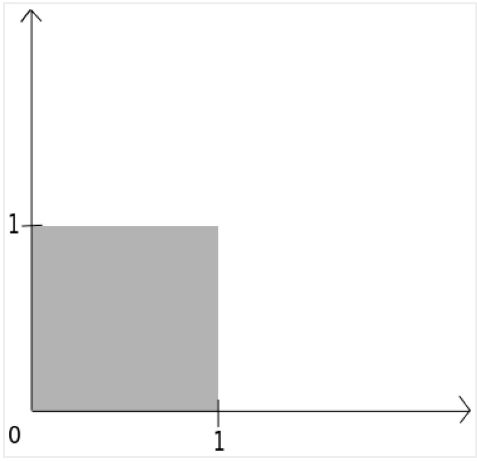
OSGeo, GDAL and other GIS fun. Find me on Twitter, GitHub or on my company Spatialys website

dimanche 6 avril 2014

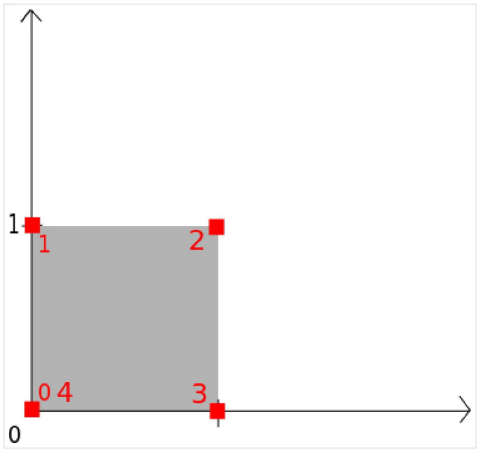
## GML madness

I am convinced that most people wonder "how many ways are there to encode a polygon in GML ?" If you have never considered that before, you might be interested in reading the following lines.

To start gently, let us consider the following grey shape :




Mathematicians call it a square, which is a particularly case of a rectangle, which is itself a polygon. A simple way of describing a polygon is to list the coordinates of its corners :




Corner 0 coordinates are (0,0)
Corner 1 coordinates are (0,1)
Corner 2 coordinates are (1,1)
Corner 3 coordinates are (1,0)
And Corner 4 = Corner 0

One of the most compact way of describing that polygon in GML 3.2 is the use of the gml:Polygon element :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
  <gml:exterior>
```





Qui êtes-vous ?

Even Rouault

Afficher mon profil complet

```
      <gml:LinearRing>
        <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
      </gml:LinearRing>
    </gml:exterior>
< /gml:Polygon>
```

We can forget the XML namespaces declaration and just concentrate on the fact that a Polygon is made of an exterior ring described by a list of positions. For those who wonder why we need to specify the "exterior", you must know that polygons may have holes in them, and those holes are called "interior rings", but we will not explore that level of complexity.

The documentation of the gml:LinearRing element shows that there are other ways of expressing the coordinates. We can isolate each corner in a separate gml:pos element :

```
< ?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://www.opengis.net/gml/3.2
                        http://schemas.opengis.net/gml/3.2.1/gml.xsd"
          gml:id="ID1">
    <gml:exterior>
      <gml:LinearRing>
        <gml:pos>0 0</gml:pos>
        <gml:pos>0 1</gml:pos>
        <gml:pos>1 1</gml:pos>
        <gml:pos>1 0</gml:pos>
        <gml:pos>0 0</gml:pos>
      </gml:LinearRing>
    </gml:exterior>
< /gml:Polygon>
```

Or we can use a gml:Point inside a gml:pointProperty :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://www.opengis.net/gml/3.2
                        http://schemas.opengis.net/gml/3.2.1/gml.xsd"
          gml:id="ID1">
    <gml:exterior>
      <gml:LinearRing>
        <gml:pointProperty>
          <gml:Point gml:id="ID2">
            <gml:pos>0 0</gml:pos>
          </gml:Point>
        </gml:pointProperty>
        <gml:pointProperty>
          <gml:Point gml:id="ID3">
            <gml:pos>0 1</gml:pos>
          </gml:Point>
        </gml:pointProperty>
        <gml:pointProperty>
          <gml:Point gml:id="ID4">
            <gml:pos>1 1</gml:pos>
          </gml:Point>
        </gml:pointProperty>
        <gml:pointProperty>
          <gml:Point gml:id="ID5">
            <gml:pos>1 0</gml:pos>
          </gml:Point>
        </gml:pointProperty>
        <gml:pointProperty>
          <gml:Point gml:id="ID6">
            <gml:pos>0 0</gml:pos>
          </gml:Point>
        </gml:pointProperty>
      </gml:LinearRing>
    </gml:exterior>
< /gml:Polygon>
```

Those who carefully look at the above snippet realize that the content of the last pointProperty is the same as the first one. So we can use xlink:href power to optimize that a bit :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:xlink="http://www.w3.org/1999/xlink"
          xsi:schemaLocation="http://www.opengis.net/gml/3.2
                        http://schemas.opengis.net/gml/3.2.1/gml.xsd"
          gml:id="ID1">
    <gml:exterior>
      <gml:LinearRing>
        <gml:pointProperty>
          <gml:Point gml:id="ID2">
```

```
            <gml:pos>0 0</gml:pos>
          </gml:Point>
        </gml:pointProperty>
        <gml:pointProperty>
          <gml:Point gml:id="ID3">
            <gml:pos>0 1</gml:pos>
          </gml:Point>
        </gml:pointProperty>
        <gml:pointProperty>
          <gml:Point gml:id="ID4">
            <gml:pos>1 1</gml:pos>
          </gml:Point>
        </gml:pointProperty>
        <gml:pointProperty>
          <gml:Point gml:id="ID5">
            <gml:pos>1 0</gml:pos>
          </gml:Point>
        </gml:pointProperty>
        <gml:pointProperty xlink:href="#ID2"/>
      </gml:LinearRing>
    </gml:exterior>
< /gml:Polygon>
```

People nostalgic of the GML 2.1.2 era will probably want to use the now deprecated (but still valid) gml:coordinates element :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                    http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:LinearRing>
      <!-- deprecated -->
        < gml:coordinates>0,0 0,1 1,1 1,0 0,0</gml:coordinates>
      </gml:LinearRing>
    </gml:exterior>
< /gml:Polygon>
```

We could play with cs (coordinate separator) and ts (tuple separator) attributes of gml:coordinates to generate alternate encoding for the coordinate list, but we will not do that. Enough with deprecated features ! Let us concentrate on modernity.

Our shape is a gml:Rectangle, isn't it ?

```
<?xml version="1.0"?>
< gml:Rectangle xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                    http://schemas.opengis.net/gml/3.2.1/gml.xsd">
    <gml:exterior>
      <gml:LinearRing>
        <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
      </gml:LinearRing>
    </gml:exterior>
< /gml:Rectangle>
```

Careful observers will notice that we have not simply substituted Polygon by Rectangle, but we have also removed the gml:id attribute. Why so ? Because a Polygon is a first citizen GML object deriving from gml:AbstractGMLType, whereas Rectangle just derives from gml:AbstractSurfacePatchType. Poor gml:Rectangle... We will come back to it later.

Until now, we have restricted the interior to be a LinearRing. But a LinearRing is a particular case of a gml:Ring :

```
<gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                    http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:LineString gml:id="ID2">
            <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
          </gml:LineString>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

As before we can use a series of gml:pos instead of gml:posList :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
```

```
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:LineString gml:id="ID2">
            <gml:pos>0 0</gml:pos>
            <gml:pos>0 1</gml:pos>
            <gml:pos>1 1</gml:pos>
            <gml:pos>1 0</gml:pos>
            <gml:pos>0 0</gml:pos>
          </gml:LineString>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

But we could also use several gml:curveMember with a simple 2-point gml:LineString :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:LineString gml:id="ID2">
            <gml:posList>0 0 0 1</gml:posList>
          </gml:LineString>
        </gml:curveMember>
        <gml:curveMember>
          <gml:LineString gml:id="ID3">
            <gml:posList>0 1 1 1</gml:posList>
          </gml:LineString>
        </gml:curveMember>
        <gml:curveMember>
          <gml:LineString gml:id="ID4">
            <gml:posList>1 1 1 0</gml:posList>
          </gml:LineString>
        </gml:curveMember>
        <gml:curveMember>
          <gml:LineString gml:id="ID5">
            <gml:posList>1 0 0 0</gml:posList>
          </gml:LineString>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

Instead of a single gml:LineString, we could use a more powerful gml:Curve :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:Curve gml:id="ID2">
            <gml:segments>
              <gml:LineStringSegment>
                <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
              </gml:LineStringSegment>
            </gml:segments>
          </gml:Curve>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

But it is a bit of a shame to use a single gml:LineStringSegment inside a gml:segments. Let us fix that :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
```

```
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:Curve gml:id="ID2">
            <gml:segments>
              <gml:LineStringSegment>
                <gml:posList>0 0 0 1</gml:posList>
              </gml:LineStringSegment>
              <gml:LineStringSegment>
                <gml:posList>0 1 1 1</gml:posList>
              </gml:LineStringSegment>
              <gml:LineStringSegment>
                <gml:posList>1 1 1 0</gml:posList>
              </gml:LineStringSegment>
              <gml:LineStringSegment>
                <gml:posList>1 0 0 0</gml:posList>
              </gml:LineStringSegment>
            </gml:segments>
          </gml:Curve>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

Of course we can still use gml:pointProperty to avoid repeating the same coordinate tuples :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:Curve gml:id="ID2">
            <gml:segments>
              <gml:LineStringSegment>
                <gml:pointProperty>
                  <gml:Point gml:id="ID3">
                    <gml:pos>0 0</gml:pos>
                  </gml:Point>
                </gml:pointProperty>
                <gml:pointProperty>
                  <gml:Point gml:id="ID4">
                    <gml:pos>0 1</gml:pos>
                  </gml:Point>
                </gml:pointProperty>
              </gml:LineStringSegment>
              <gml:LineStringSegment>
                <gml:pointProperty xlink:href="#ID4"/>
                <gml:pointProperty>
                  <gml:Point gml:id="ID5">
                    <gml:pos>1 1</gml:pos>
                  </gml:Point>
                </gml:pointProperty>
              </gml:LineStringSegment>
              <gml:LineStringSegment>
                <gml:pointProperty xlink:href="#ID5"/>
                <gml:pointProperty>
                  <gml:Point gml:id="ID6">
                    <gml:pos>1 0</gml:pos>
                  </gml:Point>
                </gml:pointProperty>
              </gml:LineStringSegment>
              <gml:LineStringSegment>
                <gml:pointProperty xlink:href="#ID5"/>
                <gml:pointProperty xlink:href="#ID3"/>
              </gml:LineStringSegment>
            </gml:segments>
          </gml:Curve>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

Another child element of gml:curveMember is a gml:CompositeCurve :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
```

```
                    http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:CompositeCurve gml:id="ID2">
            <gml:curveMember>
              <gml:LineString gml:id="ID3">
                <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
              </gml:LineString>
            </gml:curveMember>
          </gml:CompositeCurve>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

But, you may have noticed that the child of a CompositeCurve is a curveMember, which is also the parent of the CompositeCurve. So we may put a CompositeCurve inside a CompositeCurve :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:CompositeCurve gml:id="ID2">
            <gml:curveMember>
              <gml:CompositeCurve gml:id="ID3">
                <gml:curveMember>
                  <gml:LineString gml:id="ID4">
                    <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
                  </gml:LineString>
                </gml:curveMember>
              </gml:CompositeCurve>
            </gml:curveMember>
          </gml:CompositeCurve>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

You have probably understood now that we could nest CompositeCurve as many times as wished. So we have now the answer to the initial question : there is an **infinity** of ways of expressing a polygon in GML 3.2 !

Another child element of gml:curveMember is a gml:OrientableCurve :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:OrientableCurve gml:id="ID2">
            <gml:baseCurve>
              <gml:LineString gml:id="ID3">
                <gml:pos>0 0</gml:pos>
                <gml:pos>0 1</gml:pos>
                <gml:pos>1 1</gml:pos>
                <gml:pos>1 0</gml:pos>
                <gml:pos>0 0</gml:pos>
              </gml:LineString>
            </gml:baseCurve>
          </gml:OrientableCurve>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

But the full power of OrientableCurve is to be able to express the orientation of the curve. So let us split the ring into 2 pieces, one with positive orientation and one with negative orientation :

```
<?xml version="1.0"?>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
```

```
        gml:id="ID1">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:OrientableCurve gml:id="ID2">
            <gml:baseCurve>
              <gml:LineString gml:id="ID3">
                <gml:pos>0 0</gml:pos>
                <gml:pos>0 1</gml:pos>
                <gml:pos>1 1</gml:pos>
              </gml:LineString>
            </gml:baseCurve>
          </gml:OrientableCurve>
        </gml:curveMember>
        <gml:curveMember>
          <gml:OrientableCurve gml:id="ID4" orientation="-">
            < gml:baseCurve>
              <gml:LineString gml:id="ID5">
                <gml:pos>0 0</gml:pos>
                <gml:pos>1 0</gml:pos>
                <gml:pos>1 1</gml:pos>
              </gml:LineString>
            </gml:baseCurve>
          </gml:OrientableCurve>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
< /gml:Polygon>
```

Enough with polygons. A polygon is just a particular case of a gml:Surface :

```
<?xml version="1.0"?>
< gml:Surface xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
            http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:patches>
      <gml:PolygonPatch>
        <gml:exterior>
          <gml:LinearRing>
            <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
          </gml:LinearRing>
        </gml:exterior>
      </gml:PolygonPatch>
    </gml:patches>
< /gml:Surface>
```

Instead of a gml:PolygonPatch as a child of a gml:patches, we can use the gml:Rectangle we have used before :

```
<?xml version="1.0"?>
< gml:Surface xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:patches>
      <gml:Rectangle>
        <gml:exterior>
          <gml:LinearRing>
            <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
          </gml:LinearRing>
        </gml:exterior>
      </gml:Rectangle>
    </gml:patches>
< /gml:Surface>
```

A Surface seems to be too simple. Why not using a gml:CompositeSurface ?

```
<?xml version="1.0"?>
< gml:CompositeSurface xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
    <gml:surfaceMember>
      <gml:Surface gml:id="ID2">
        <gml:patches>
          <gml:PolygonPatch>
            <gml:exterior>
              <gml:LinearRing>
                <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
```

```
                </gml:LinearRing>
              </gml:exterior>
            </gml:PolygonPatch>
          </gml:patches>
        </gml:Surface>
      </gml:surfaceMember>
    < /gml:CompositeSurface>
```

But it looks a bit dumb to use only one gml:surfaceMember. Let us divide our square into 2 triangles :

```
<?xml version="1.0"?>
< gml:CompositeSurface xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
  <gml:surfaceMember>
    <gml:Surface gml:id="ID2">
      <gml:patches>
        <gml:PolygonPatch>
          <gml:exterior>
            <gml:LinearRing>
              <gml:posList>0 0 0 1 1 1 0 0</gml:posList>
            </gml:LinearRing>
          </gml:exterior>
        </gml:PolygonPatch>
      </gml:patches>
    </gml:Surface>
  </gml:surfaceMember>
  <gml:surfaceMember>
    <gml:Surface gml:id="ID3">
      <gml:patches>
        <gml:PolygonPatch>
          <gml:exterior>
            <gml:LinearRing>
              <gml:posList>0 0 1 1 1 0 0 0</gml:posList>
            </gml:LinearRing>
          </gml:exterior>
        </gml:PolygonPatch>
      </gml:patches>
    </gml:Surface>
  </gml:surfaceMember>
< /gml:CompositeSurface>
```

Instead of a gml:CompositeSurface, why not using a gml:MultiSurface ?

```
<?xml version="1.0"?>
< gml:MultiSurface xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
  <gml:surfaceMember>
    <gml:Surface gml:id="ID2">
      <gml:patches>
        <gml:PolygonPatch>
          <gml:exterior>
            <gml:LinearRing>
              <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
            </gml:LinearRing>
          </gml:exterior>
        </gml:PolygonPatch>
      </gml:patches>
    </gml:Surface>
  </gml:surfaceMember>
< /gml:MultiSurface>
```

or maybe you prefer to use gml:surfaceMembers (with a final 's') instead of a gml:surfaceMember :

```
<?xml version="1.0"?>
< gml:MultiSurface xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
  <gml:surfaceMembers>
    <gml:Surface gml:id="ID2">
      <gml:patches>
        <gml:PolygonPatch>
          <gml:exterior>
            <gml:LinearRing>
              <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
```

```
                    </gml:LinearRing>
                  </gml:exterior>
                </gml:PolygonPatch>
              </gml:patches>
            </gml:Surface>
          </gml:surfaceMembers>
        < /gml:MultiSurface>
```

Similarly to gml:CompositeCurve, we can arbitrary nest as many gml:CompositeSurface as wished :

```
<?xml version="1.0"?>
< gml:CompositeSurface xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                    http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
  <gml:surfaceMember>
    <gml:CompositeSurface gml:id="ID2">
      <gml:surfaceMember>
        <gml:Surface gml:id="ID3">
          <gml:patches>
            <gml:PolygonPatch>
              <gml:exterior>
                <gml:LinearRing>
                  <gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
                </gml:LinearRing>
              </gml:exterior>
            </gml:PolygonPatch>
          </gml:patches>
        </gml:Surface>
      </gml:surfaceMember>
    </gml:CompositeSurface>
  </gml:surfaceMember>
< /gml:CompositeSurface>
```

So we have now two different kind of infinities ! That we could combine together. But, do not hope to have discovered more ways of expressing polygons. The cardinality of the set of natural numbers times the set of natural numbers ( N x N ) is just the cardinality of the set of natural numbers...

To conclude, we should mention that the authors of the GML specification have admitted that encoding polygons was a bit too complicated. So they have invented a "compact encoding" in the extended schemas of GML 3.3 :

```
<?xml version="1.0"?>
< gmlce:SimplePolygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:gmlce="http://www.opengis.net/gml/3.3/ce"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
            http://schemas.opengis.net/gml/3.2.1/gml.xsd
            http://www.opengis.net/gml/3.3/ce
            http://schemas.opengis.net/gml/3.3/geometryCompact.xsd"
        gml:id="ID1">
  <!-- we may, or not, close the ring -->
  < gml:posList>0 0 0 1 1 1 1 0 0 0</gml:posList>
< /gmlce:SimplePolygon>
```

But our SimplePolygon is indeed a SimpleRectangle. So let us use instead :

```
<?xml version="1.0"?>
< gmlce:SimpleRectangle xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:gmlce="http://www.opengis.net/gml/3.3/ce"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
            http://schemas.opengis.net/gml/3.2.1/gml.xsd
            http://www.opengis.net/gml/3.3/ce
            http://schemas.opengis.net/gml/3.3/geometryCompact.xsd"
        gml:id="ID1">
  <!-- caution : we should NOT clause with the first vertex ! -->
  < gml:posList>0 0 0 1 1 1 1 0</gml:posList>
< /gmlce:SimpleRectangle>
```

You can found the above 25 snippets at the following URL : http://even.rouault.free.fr/gml/
They are all valid GML 3.2 snippets that validate the XML schemas and pass the GML 3.2 Conformance Test Suite (except gml4.xsd which uses the deprecated gml:coordinates element).

Oh, final fun, as GML is XML, we can also use XML substitutable entities ...

```
<?xml version="1.0"?>
< !DOCTYPE points [
 < !ENTITY pt0 "0 0">
```

```
< !ENTITY pt1 "0 1">
< !ENTITY pt2 "1 1">
< !ENTITY pt3 "1 0">
]>
< gml:Polygon xmlns:gml="http://www.opengis.net/gml/3.2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gml/3.2
                    http://schemas.opengis.net/gml/3.2.1/gml.xsd"
        gml:id="ID1">
  <gml:exterior>
    <gml:LinearRing>
      <gml:posList>&pt0; &pt1; &pt2; &pt3; &pt0;</gml:posList>
    </gml:LinearRing>
  </gml:exterior>
< /gml:Polygon>
```

Those who wonder why I decided to write this article might want to have a look at the following simplified GML sample of a real-world use case where a PolygonPatch has only an interior ring (a hole), but no exterior ring... Standalone holes : interesting concept, isn't it ?

Publié par Even Rouault à 07:47 G+

## 5 commentaires:

**vncbmerter** 6 avril 2014 à 12:07

Even,
this is probably why shapefiles are still used so much...
Is the OGR GML driver able to process all this cases ?
Fred.

Répondre

Réponses

**Even Rouault** 6 avril 2014 à 13:48

Not all of them, but most of them.

**Répondre**

**Carl Reed (Geospatial Standards)** 15 mai 2014 à 10:46

Even - Great information!

Your example of a polygon is why communities develop profiles or application schemas for using GML in their specific domains. For example, the IETF uses a GML application schema with a single allowed approach to defining a polygon. This way, the IETF community has removed the "infinity" of possible approaches and simplified life for developers. Also, have you considered the compact form for polygon that is defined in GML 3.3? Section 7: Compact Encodings of Commonly Used GML Geometries. OASIS (another standards organization) has defined a GML profile using the compact encodings for use in many of their standards).

Cheers!

Répondre

**Carl Reed (Geospatial Standards)** 15 mai 2014 à 10:47

Whoops - should have read all the way to the end. I see you covered the compact encoding!

Répondre

**janh** 21 mai 2014 à 08:26

Thanks for the education..
Apart from the standards issues, the standalone hole is interesting.
I would guess that the concept has potential use.
Thinking geometrically (rather than geographically) at first, you could define the hole with an implicit exterior ring at infinity, which makes it an all, except definition. The standalone hole can of course contain (positive) rings.
Going geographical, the all-except notion still has validity, although the domain-definition becomes leading (instead of infinity), e.g. 2.5D earth or 3D earth.
Questions about the limits of the 3D domain remind me of the origin of fractals: the attempt to define and measure the circumference of England..
cheers, I hope you don't mind my loose off-topicness..,
Jan

Répondre

Saisissez votre commentaire…

Article plus récent Accueil Article plus ancien

Inscription à : Publier les commentaires (Atom)