01 CO TO JEST GML

GML jest format tekstowy, który może odczytać zarówno człowiek jak i maszyna. Dla nas odczyt tego formatu może być łatwiejszy, gdy edytor tekstu podkoloruje składnię. Ja korzystam z Notepad++ z wtyczką XML Tools, którą używam do formatowania składni.

Co trzeba wiedzieć, żeby odczytać GML? Napewno trzeba znać jego strukturę, która oparta jest na XML (ang. Extensible Markup Language). Format XML został opracowany przez W3C (ang. World Wide Web Consortium) i jest jednym z wiądących formatów na świecie, jeśli chodzi o wymianę danych między różnymi systemami. Po pierwsze, na formacie XML powstał internet. Po drugie, jego dużą zaletą jest to, że mając schemat XSD możemy sprawdzić poprawność pliku. Mając tę wiedzę można się domyśleć dlaczego w geodezji użyto właśnie tego formatu do wymiany danych.

Skoro już napisałem, że trzeba znać strukturę XML, to zacznijmy od najmniej skomplikowanego elementu. Każdy plik GML składa się z zagnieżdżonych elementów, które zapisane są za pomocą znaczników.

Element

```
<geometria></geometria>
```

Teraz możemy powiedzieć, że zapisaliśmy element o nazwie 'geometria', który nie posiada wartości i atrybutów. Element ten składa się z dwóch części: otwierającej i zamykającej. Część otwierająca to '<geometria>', natomiast część zamykająca to '</geometria>'. Jeśli otworzyliśmy element to musimy go zamknąć!

Dla elementów, które nie posiadają wartości dopuszczalny jest poniży zapis (jest on tożsamy z tym powyżej):

```
<geometria/>
```

Element główny

Format GML wymaga, aby dokument zwierał wyłącznie jeden element główny (ang. root element) o nazwie FeatureCollection (wymagany jest jeszcze prefix, ale o tym będzie później):

```
<FeatureCollection></FeatureCollection>
```

Elementy zagnieżdżone

Kolejną ważną cechą jest to, że elementy mogą się zagnieżdżać.

```
<FeatureCollection>
    <featureMember/>
    <featureMember></featureMember>
</FeatureCollection>
```

W ten sposób budujemy strukturę drzewiastą. Z elementu głównego, z jednego pnia rozrastają się elementy zagnieżdżone. Elementem głównym jest tutaj FeatureCollection, natomiast pierwszym i drugim dzieckiem jest featureMember. Oba nie posiadają atrybutów i wartości. Jak juz wcześniej pisałem, są to takie same elementy, tylko zapisane na dwa sposoby. W pliku GML częściej można spotkać ten pierwszy zapis, bo zwyczajnie mniej miejsca zajmuje.

Atrybuty

W końcu nadszedł czas, żeby jakoś przekazać dane. Można to zrobić za pomocą atrybutów. Po pierwsze każdy obiekt może posiadać dowolną ilość atrybutów. Po drugie atrybut składa się z nazwy i wartości. Po trzecie atrybut może zwierać elementów zagnieżdżonych. Tylko wartość.

```
<FeatureCollection id="featureCollection">
    <featureMember/>
    <featureMember></featureMember>
</FeatureCollection>
```

W przykładzie atrybut ma nazwę 'id', którego wartość wynosi 'featureCollection'.

Przestrzeń nazw

Teraz przechodzimy do elementu, który w GML towarzyszy przy każdym elemencie. Jest nim przestrzeń nazw. Została wprowadzona po to, żeby tych samych nazw elementów w różnych bazach, można było bez wątpliwość zidentyfikować obiekt. Przestrzeń nazw stawia się przed nazwą elementu, oddzielając się od niego znakiem dwukropka ':'. Przestrzeń nazw możemy dodać do nazwy elementu lub nazwy atrybutu.

```
<FeatureCollection gml:id="featureCollection">
    <featureMember/>
    <featureMember></featureMember>
</FeatureCollection>
```

Jak widać w przykładzie przestrzeń nazw to 'gml', która dodana jest do nazwy atrybutu 'id', czyli atrybut 'id' należy do przestrzeni nazw 'gml', którego wartość wynosi 'featureCollection'.

W kolejnym przykładzie dodamy przestrzeń nazw do nazw elementów.

```
<gml:FeatureCollection gml:id="featureCollection">
    <gml:featureMember/>
    <gml:featureMember></gml:featureMember>
</gml:FeatureCollection>
```

Przestrzeń nazw 'gml' została dodana do wszystkich elementów, w tym do elementu głównego, oraz jego potomków. Ważne jest to, że jeśli w części otwierającej dodamy przestrzeń nazw, to również musi się ona znaleźć w części zamykającej.

Komentarze

Format GML pozwala umieszczać komentarze wewnątrz dokumentu. Komentarz to wszystko co znajduje się pomiędzy <!-- i -->. Znowu występuje zasada: to co zostało otwarte musi zostać zamknięte. W przeciwnym razie plik nie zostanie poprawnie zwalidowany. Komentarz może znajdować się pomiędzy elementami, wewnątrz elementu, przed elementem głównym lub na końcu dokumentu.

```
<!-- eksport z programu AcadGEO do pliku gml -->
<gml:FeatureCollection gml:id="featureCollection">
    <gml:featureMember>
    </gml:featureMember>
</gml:FeatureCollection>
```

Kometarzem w powyższym przykładzie jest to tekst 'eksport z programu AcadGEO do pliku gml'.
Mała uwaga! W poprzednim przykładzie były dwa elementy potomne elemntu głównego. W obecnym jest jeden element potomny. Nieco inaczej zapisany, bo otwarcie znacznika jest w jednej linii, natomiast jego zamknięcie jest w kolejnej. Z punktu widzenia standartu jest to też poprawne. Dlaczego? Ponieważ element składa się z części otwierającej i zamykającej, jednak nie jest powiedziane, że muszą obie być w jednej linii. Z punktu widzenia człowieka wygląda to inaczej, jednak dla maszyny odczytującej nie będzie różnicy, poza ilością danych, którą musi odczytać. Wynika to z powodu, że białe znaki pomiędzy elementami są ignorowane.
Białe znaki to: spacja, znak tabulacji, znak końca linii, lub dowolny inny znak niemający kształtu na ekranie.

Wracając do komentarzy to możemy go rozbudować i zapisać go w formie wielo liniowej.

```
<!--
;eksport z programu AcadGEO do pliku gml
;wersja-programu: 2022 v2.01.11.0
;wersja-pliku: gml 2021
;data: 2022-10-27T20:10:11
-->
<gml:FeatureCollection gml:id="featureCollection">
    <gml:featureMember>
    </gml:featureMember>
</gml:FeatureCollection>
```

W powyższym przykładzie pokazałem, ze również w komentarzach można zapisywać dane, które maszyna może interpretować. Nie jest to luźny zapis tylko forma zapisu za pomocą parametru i wartości. Parametr 'wersja-pliku' ma wartość 'gml 2022'.

Prolog

Zgodnie ze standardem, każdy dokument powinien zawierać prolog, którego częścią jest deklaracja:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Jest to informacja o wersji XML, której używamy, oraz określa kodowanie znaków, które zostało użyte w pliku. Wewnątrz prologu mogą znajdować się także inne deklaracje DTD (ang. Document Type Declaration) opisujące dokument. Dotychczas pisałem, że omawiam format GML, a de facto omawiam cały czas format XML. Dlaczego tak jest? Ponieważ format GML został stworzony w oparciu o XML, więc wszystko co opisuje standard XML, opisuje też standard GML. Format GML niczego nie modyfikuje w XML, tylko go rozbudowuje o dane przestrzenne. O tym będzie w innym blogu.
Prolog umieszcza się na początku:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
;eksport z programu AcadGEO do pliku gml
;wersja-programu: 2022 v2.01.11.0
;wersja-pliku: gml 2021
;data: 2022-10-27T20:10:11
-->
<gml:FeatureCollection gml:id="featureCollection">
    <gml:featureMember>
    </gml:featureMember>
</gml:FeatureCollection>
```

Część otwierająca prolog to '<?', a zamykająca to '?>'. Wszystko co mieści się pomiędzy to deklaracje. Cały czas obowiązuje zasada otwarcia i zamknięcia elementu.

To tyle jeśli chodzi o podstawową strukturę. W kolejnych artykułach zaprezentuję kolejne tematy związane z formatem GML.

Artykuł opracował: Szymon Szczerba
Data utworzenia: 9.11.2022r.