# Connection of the MapML initiative with OGC standards such as the new OGC API Tiles candidate or the old WMS

Joan Masó

Maps for the Web 2020

Findable

Accessible

Interoperable

Reusable

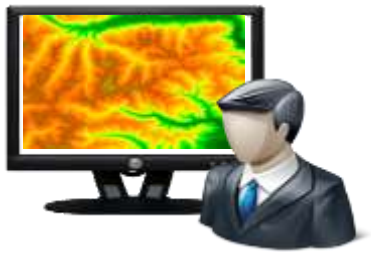
GRUMETS

CREAF

OGC®

# Who to send a map to a client

Client



Data Protocol

Server

| | | | | | |
|---|---|---|---|---|---|
| 123.0 | 120.0 | 121.5 | 120.6 | 127.1 | 129.0 |
| 123.0 | 120.0 | 121.5 | 120.6 | 127.1 | 129.0 |
| 123.0 | 120.0 | 121.5 | 120.6 | 999.9 | 999.9 |
| 123.0 | 120.0 | 121.5 | 999.9 | 999.9 | 999.9 |
| 123.0 | 120.0 | 121.5 | 999.9 | 999.9 | 999.9 |

JavaScript code

```
function CarregaDataViewsCapa(dv, i_nova_vista, i_data, valors)
{
	for (ver i_v=0; i_v<valors.length; i_v++)
	{
		if (i_nova_vista==-1)
			array_buffer=valors[i_v].arrayBuffer;
		else if (i_nova_vista==-2)
			array_buffer=valors[i_v].arrayBufferPrint;
		else if (i_nova_vista==-3)
			array_buffer=valors[i_v].capa_rodet[i_data].arrayBuffer;
		else if (i_nova_vista==-4)
			array_buffer=valors[i_v].capa_video[i_data].arrayBuffer;
		else
			array_buffer=(valors[i_v].nova_capa  && valors[i_v].nova_
	return n_v;
}
```

HTML MicroXML

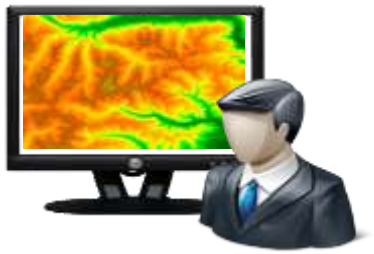
```
<map is="web-map" zoom="17" lat="45.398043" lon="-75.70683"
width="700" height="400" controls hidden>
    <layer id="osm" src="https://geogratis.gc.ca/mapml/en/osmtile/osm/"
            label="Open Street Map" checked hidden></layer>

    <area is="map-area"id="marker2" href="https://example.com/marker/"
            alt="Marker" coords="265,185" shape="marker">
</map>
```

# The common client-server diagram

Client

Server



① OWS OpenAPI services

② Protocol

=>

JavaScript code

| | | | | | |
|---|---|---|---|---|---|
| 123.0 | 120.0 | 121.5 | 120.6 | 127.1 | 129.0 |
| 123.0 | 120.0 | 121.5 | 120.6 | 127.1 | 129.0 |
| 123.0 | 120.0 | 121.5 | 120.6 | 999.9 | 999.9 |
| 123.0 | 120.0 | 121.5 | 999.9 | 999.9 | 999.9 |
| 123.0 | 120.0 | 121.5 | 999.9 | 999.9 | 999.9 |

```
function CarregaDataViewsCapa(dv, i_nova_vista, i_data, valors)
{
	for (ver i_v=0; i_v<valors.length; i_v++)
	{
		if (i_nova_vista==-1)
			array_buffer=valors[i_v].arrayBuffer;
		else if (i_nova_vista==-2)
			array_buffer=valors[i_v].arrayBufferPrint;
		else if (i_nova_vista==-3)
			array_buffer=valors[i_v].capa_rodet[i_data].arrayBuffer;
		else if (i_nova_vista==-4)
			array_buffer=valors[i_v].capa_video[i_data].arrayBuffer;
		else
			array_buffer=(valors[i_v].nova_capa  && valors[i_v].nova_
	return n_v;
}
```

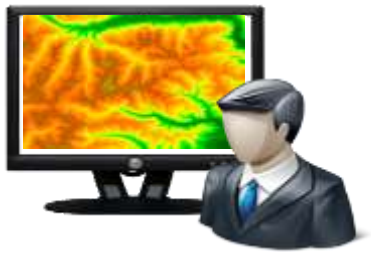
HTML MicroXML

```
<map is="web-map" zoom="17" lat="45.398043" lon="-75.70683"
width="700" height="400" controls hidden>
    <layer id="osm" src="https://geogratis.gc.ca/mapml/en/osmtile/osm/"
            label="Open Street Map" checked hidden></layer>

    <area is="map-area"id="marker2" href="https://example.com/marker/"
            alt="Marker" coords="265,185" shape="marker">
</map>
```

# The common client-server diagram

Client



①

②

Data Protocol

JavaScript Map API code

③

HTML MicroXML

Server

| | | | | | |
|---|---|---|---|---|---|
| 123.0 | 120.0 | 121.5 | 120.6 | 127.1 | 129.0 |
| 123.0 | 120.0 | 121.5 | 120.6 | 127.1 | 129.0 |
| 123.0 | 120.0 | 121.5 | 120.6 | 999.9 | 999.9 |
| 123.0 | 120.0 | 121.5 | 999.9 | 999.9 | 999.9 |
| 123.0 | 120.0 | 121.5 | 999.9 | 999.9 | 999.9 |

maps

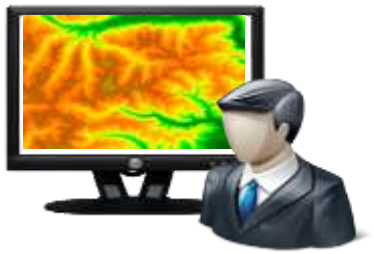
```
function CarregaDataViewsCapa(dv, i_nova_vista, i_data, valors)
{
        for (ver i_v=0; i_v<valors.length; i_v++)
        {
                if (i_nova_vista==-1)
                        array_buffer=valors[i_v].arrayBuffer;
                else if (i_nova_vista==-2)
                        array_buffer=valors[i_v].arrayBufferPrint;
                else if (i_nova_vista==-3)
                        array_buffer=valors[i_v].capa_rodet[i_data].arrayBuffer;
                else if (i_nova_vista==-4)
                        array_buffer=valors[i_v].capa_video[i_data].arrayBuffer;
                else
                        array_buffer=(valors[i_v].nova_capa  && valors[i_v].nova_
        return n_v;
}
```

```
<map is="web-map" zoom="17" lat="45.398043" lon="-75.70683"
width="700" height="400" controls hidden>
    <layer id="osm" src="https://geogratis.gc.ca/mapml/en/osmtile/osm/"
            label="Open Street Map" checked hidden></layer>

    <area is="map-area"id="marker2" href="https://example.com/marker/"
            alt="Marker" coords="265,185" shape="marker">
</map>
```

# The common client-server diagram

Client



① Data

② Protocol

CERTIFIED OGC COMPLIANT => Map API

③

Leaflet

Google maps

HTML
MapML
MicroXML
(OWS context)

Server

| | | | | | |
|---|---|---|---|---|---|
| 123.0 | 120.0 | 121.5 | 120.6 | 127.1 | 129.0 |
| 123.0 | 120.0 | 121.5 | 120.6 | 127.1 | 129.0 |
| 123.0 | 120.0 | 121.5 | 120.6 | 999.9 | 999.9 |
| 123.0 | 120.0 | 121.5 | 999.9 | 999.9 | 999.9 |
| 123.0 | 120.0 | 121.5 | 999.9 | 999.9 | 999.9 |

```
function CarregaDataViewsCapa(dv, i_nova_vista, i_data, valors)
{
        for (ver i_v=0; i_v<valors.length; i_v++)
        {
                if (i_nova_vista==-1)
                        array_buffer=valors[i_v].arrayBuffer;
                else if (i_nova_vista==-2)
                        array_buffer=valors[i_v].arrayBufferPrint;
                else if (i_nova_vista==-3)
                        array_buffer=valors[i_v].capa_rodet[i_data].arrayBuffer;
                else if (i_nova_vista==-4)
                        array_buffer=valors[i_v].capa_video[i_data].arrayBuffer;
                else
                        array_buffer=(valors[i_v].nova_capa  && valors[i_v].nova_
        return n_v;
}
```

```
<map is="web-map" zoom="17" lat="45.398043" lon="-75.70683"
width="700" height="400" controls hidden>
    <layer id="osm" src="https://geogratis.gc.ca/mapml/en/osmtile/osm/"
            label="Open Street Map" checked hidden></layer>

    <area is="map-area"id="marker2" href="https://example.com/marker/"
            alt="Marker" coords="265,185" shape="marker">
</map>
```

# 4 levels of complexity

①

- **Manually** build JavaScript code that
  - builds GET KVP request
  - interpret GML (XML)
  - extract and show the information
- Create a HTML 

②

Coming soon

- Automatically
  - create some code from the OpenAPI document
  - parse JSON
- **Manually** extract and show the information

③

- Call a javascript API:

```
var map = L.map(mapDiv, mapOptions);

var wmsLayer =
    L.tileLayer.wms('https://demo.boundlessgeo.com/geoserver/ows?', {
    layers: 'nasa:bluemarble' }).addTo(map);
```

④

- MapML
  - Add a MapML document into the <map> element of your HTML page

# The connection of MapML with OGC

- OGC Testbed-16: MapML Clients
  - MapML as an output of a Machine Learning process.

- OGC Testbed-15: Quebec Model MapML Engineering Report
  - 19-046r1 Scott Serich
  - Suggestion for OGC API Features to negotiate MapML content in the */items* response for retrieving <features>

- OGC Testbed-14: MapML Engineering Report
  - 18-023r1 Joan Masó
  - Suggestion for MapML to include URI templates instead of explicit references to tiles
    - Incorporated in the MapML standard draft

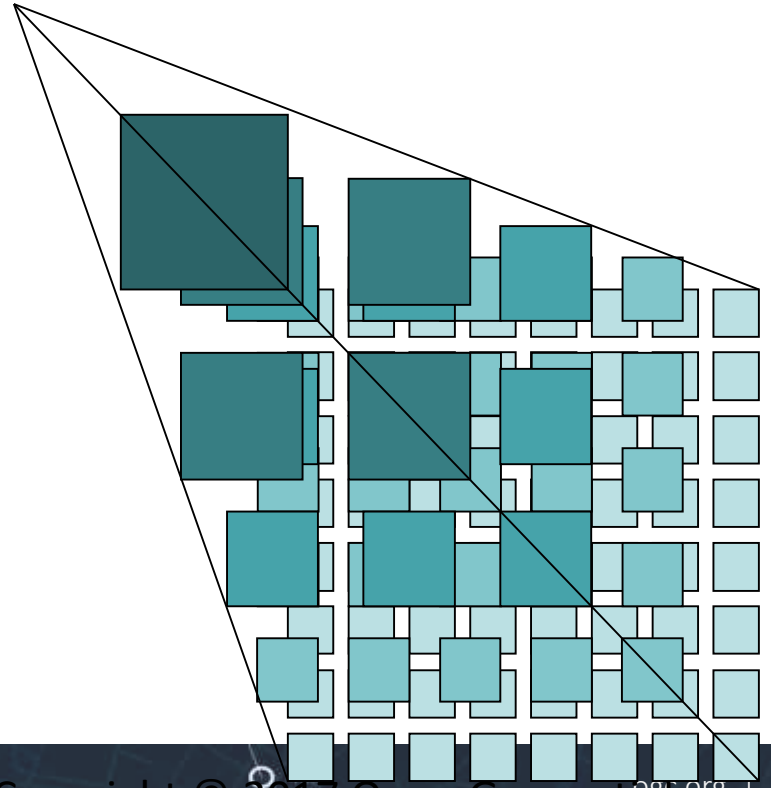
- OGC Testbed-13: MapML Engineering Report
  - 17-019 Joan Maso
  - Comparing MapML with OWS Context

# It is all about tiles (and features)

- OGC and MapML share exactly the same TileMatrixSet concept. MapML calls it *Tiled Coordinate Reference System*
- MapML defines 4 *Tiled Coordinate Reference Systems* that you can use. OGC allows for any *TileMatrixSet*
- 3 of the 4 "tcrs" in MapML are also defined in Annex D of the OGC Two Dimensional Tile Matrix Set standard

- OSMTILE
- CBMTILE
- APSTILE
- WGS84

- WebMercatorQuad
- CanadianNAD83_LCC
- *Not yet. DIY is possible*
- WorldCRS84Quad



# Tilesets

- Tilesets are represented by URI templates both in OGC API Tiles (and in the old WMTS) and in the MapML

```
"links": [
  {
    "href": "http://data.example.com/collections/buildings/map/{styleId}/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}.png",
    "rel": "tile",
    "type": "image/png"
  },
```

OGC API

Geospatial Resource Tiled Maps map (renderizations or vector tiles) partitioned into a hierarchy of tiles of a collection

GET /collections/{collectionId}/map/{styleId}/tiles/ WebMercatorQuad /{tileMatrix}/{tileRow}/{tileCol}

MapML

```
<title>tiles form USGS<title>
<extent units="OSMTILE">
    <input name="tilematrix" type="zoom" value="15" min="0" max="15">
    <input name="tileRow" type="location" units="tilematrix" axis="row">
    <input name="tileCol" type="location" units="tilematrix" axis="column">
    <link rel="tile" tref="https://.../tiles/WebMercatorQuad/{tileMatrix}/{tileRow}/{tileCol}/">
</extent>
```

# OGC API Tiles and Maps

- MapML need tiles but it does not specify who will produce them
- An API conformant to OGC API Tiles can generate tiles for a MapML.
  - OpenAPI and MapML will share the same URL template
- MapML does not specify how to produce a MApML document

- In OGC API - Tiles
  - Collections can be distributed as tiles.
    - Tiles are available as URI template
  - There is a tileset metadata document
    - MapML is a kind of tileset metadata document

New feature in OGC API Tiles

# OGC API Tiles

GET / (landing page)

- Provides the conformance and API description pages

GET /conformance

- List of conformance classes supported by the implementation

GET /api

- The API definition. Currently only in OpenAPI 3.0

GET /collections

- The list of geospatial data in the API

GET /collections/{collectionId}

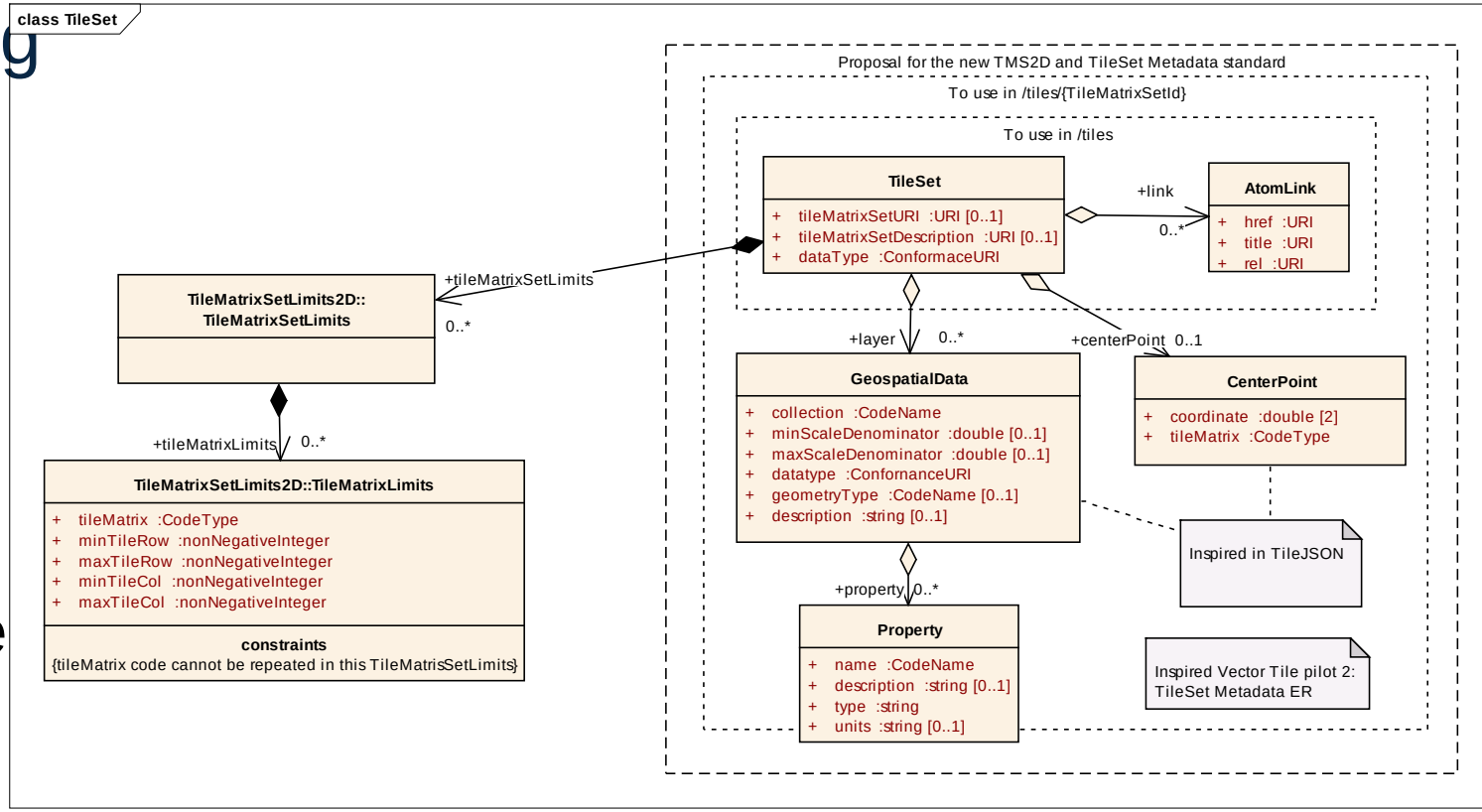
- A geospatial data [description]

GET /collections/{collectionId}/tiles

- List of tilesset (one for each TileMatrixSet) in the collection

# OGC API Tiles

- /collections/{collectionId}/tiles/{TileMatrixSetId}
- Returns a JSON file describing the tileset.
  - It provides a bit of metadata
  - It points to the TileMatrixSet definition
  - It states TileMatrixSetLimits
  - It points to the tile URI template
  - It provides a description of the content of the tiles



# OGC API Tiles

- \collections\{collectionId}\tiles\{TileMatrixSetId}

- Returns a JSON file describing the tileset.
  - It provides a bit of metadata
  - It points to the TileMatrixSet definition
  - It states TileMatrixSetLimits
  - It points to the tile URI template
  - It provides a description of the content of the tiles

- A MapML document is actually another way of describing a tileset.

```
<title>tiles form USGS<title>
<extent units="OSMTILE">
   <input name="z" type="zoom" value="15" min="0" max="15">
   <input name="y" type="location" units="tilematrix" axis="row">
   <input name="x" type="location" units="tilematrix" axis="column">
   <link rel="tile" tref="https://.../tile/{z}/{y}/{x}/">
</extent>
```

# OpenAPI for OGC API Tiles with a MapML

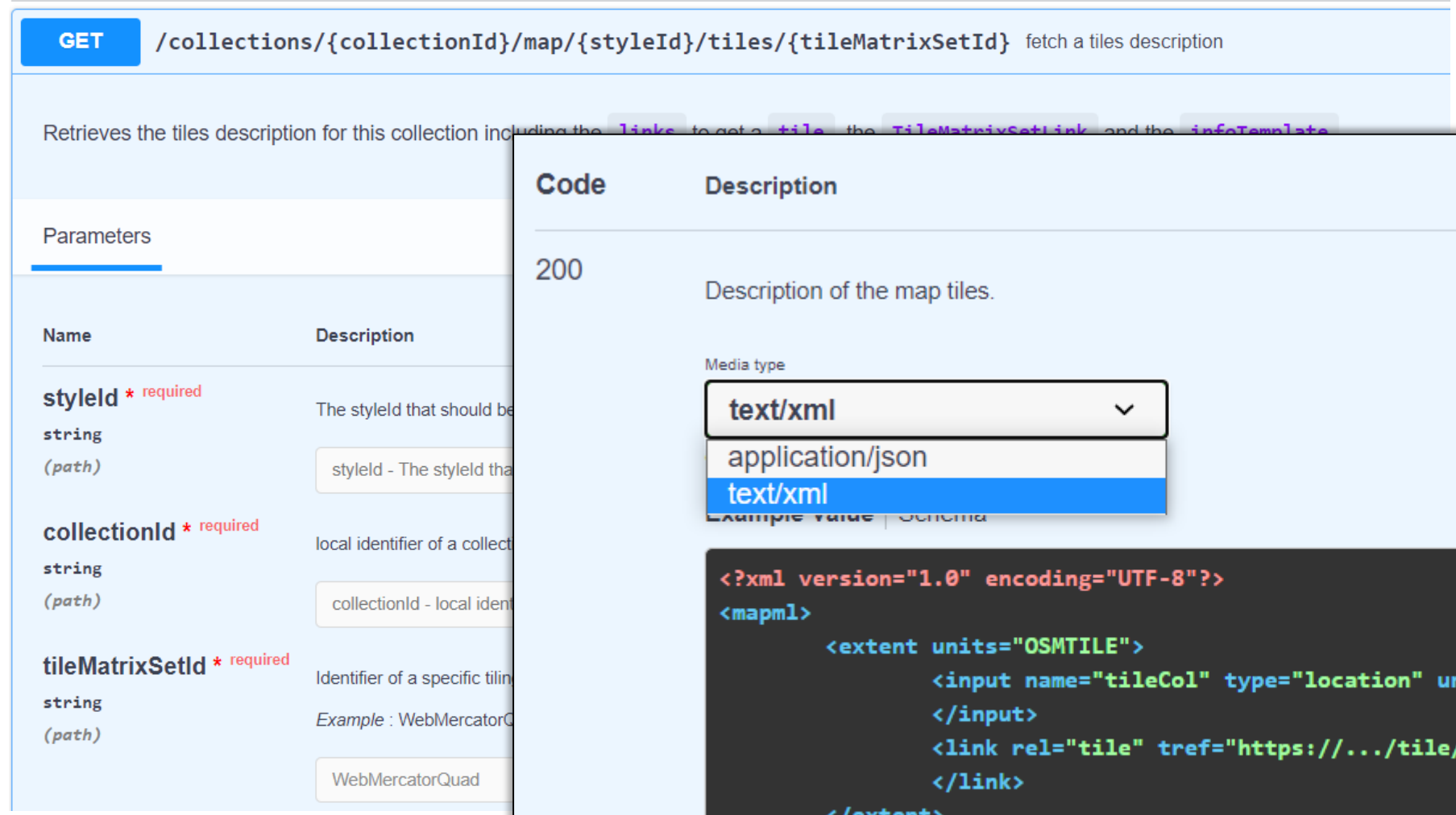
**GET** `/collections/{collectionId}/map/{styleId}/tiles/{tileMatrixSetId}` fetch a tiles description

Retrieves the tiles description for this collection including the `links` to get a `tile`, the `TileMatrixSetLink` and the `infoTemplate`

Parameters

| Name | Description |
|---|---|
| **styleId** * required<br>**string**<br>*(path)* | The styleId that should be included in the map. Each maps description includes a valid list of stylesIds.<br>styleId - The styleId that should be included in the map. Each |
| **collectionId** * required<br>**string**<br>*(path)* | local identifier of a collection<br>collectionId - local identifier of a collection |
| **tileMatrixSetId** * required<br>**string**<br>*(path)* | Identifier of a specific tiling scheme. It can be one of those specified in Annex D.1 of the OGC 17-083r2 standard or one defined in this service.<br>*Example* : WebMercatorQuad<br>WebMercatorQuad |

# OpenAPI for OGC API Tiles with a MapML



# Conclusions

- MapML is the simplest way of adding geospatial information in web browsers
  - MapML makes the use of *OGC API - Tiles* simpler in browsers
    - It provides a way to communicate two resources (tileset and tile URI template) of the *OGC API - Tiles* to browsers
  - *OGC API - Tiles* provides a way to automatically generate MapML documents that are connected to the OGC API tilesets
  - *OGC API – Tiles* is the *CGI* of the MapML documents.
- OGC should consider to describe how to generate MapML documents in the *OGC API – Tiles, part 1*
  - as an alternative format for a tileset metadata (as in OGC 2D TMS standard).
    - another alternative format considered is TileJSON (for "mapbox vector tiles")


# Thank You!

JoanMaso@uab.cat

GRUMETS CREAF