MapML Experiments

OGC July 2024 Open Standards Code Sprint

Rui Cavaco Barrosa (rpcavaco@gmail.com)

Expectations, preparation, goals

- Planned presence: three days, in person, at the Open Standards code sprint
- Experimenting with MapML was a stated main goal
- Other goals, secondary:
    - experimenting with MapML on JavaScript
    - extend / understand the use of projected CRS (such as ‘national grids’)

Real work, day one

At first experiments it was clear that MapML itself, and the reference implementation polyfill, have several complexities which manifest themselves when one goes beyond the simplest OSM tiling examples.

Three examples:

- the map-extent element (and possibility of existing several for each layer)
- the combining of “projection” attribute and of “units” attribute
- the OSMTILES, CBTILES and other keywords for “projection” or “units”

So it became clear the necessity to go through several examples to fully grasp the MapML complexity. This effectively occupied the code sprint’s first day.

Day two, setting JavaScript aside

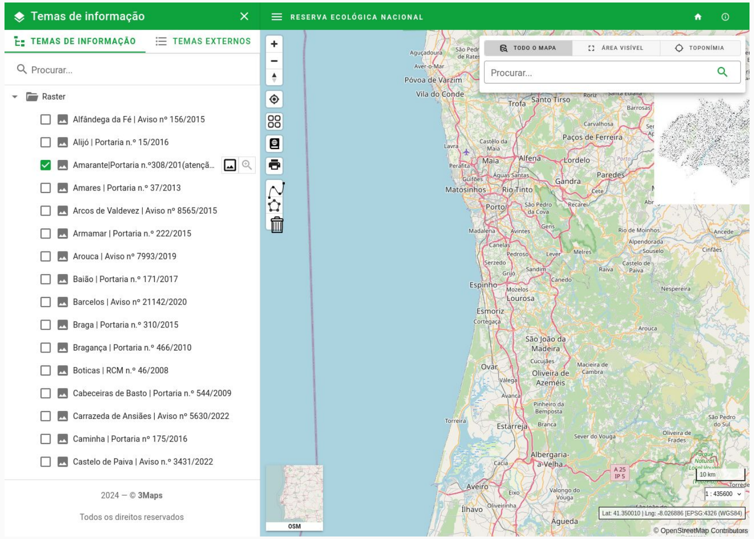
After insisting in go through the seemingly most relevant examples, I was curious on how to support “custom” CRS and on how to properly consume vector data such as GeoJSON or JSON-FG. Trying to explore this, I started peeking the MapML polyfill code.

But the interest on MapML is not around incremeting or extending the existing polyfill but on future support of the standard by browser vendors. So instead of experimenting in JS, I decided to explore DOM dynamics, and the ability to add and remove layers to a MapML map.

Day three, a MapML webapp

A real world problem to solve: this site from my home institution allows to navigate the “REN” (national ecological reserve) areas of Portuguese, one layer for each municipality.

The user must manually locate a given municipality in map and then activate the corresponding layer in layer list at left. The user must know how to locate the municipality in map (name search is not properly working).



Raster
- [ ] Alfândega da Fé | Aviso nº 156/2015
- [ ] Alijó | Portaria n.º 15/2016
- [✓] Amarante|Portaria n.º308/201(atençã...
- [ ] Amares | Portaria n.º 37/2013
- [ ] Arcos de Valdevez | Aviso nº 8565/2015
- [ ] Armamar | Portaria n.º 222/2015
- [ ] Arouca | Aviso nº 7993/2019
- [ ] Baião | Portaria n.º 171/2017
- [ ] Barcelos | Aviso nº 21142/2020
- [ ] Braga | Portaria n.º 310/2015
- [ ] Bragança | Portaria n.º 466/2010
- [ ] Boticas | RCM n.º 46/2008
- [ ] Cabeceiras de Basto | Portaria n.º 544/2009
- [ ] Carrazeda de Ansiães | Aviso nº 5630/2022
- [ ] Caminha | Portaria nº 175/2016
- [ ] Castelo de Paiva | Aviso n.º 3431/2022

2024 — © **3Maps**

Todos os direitos reservados

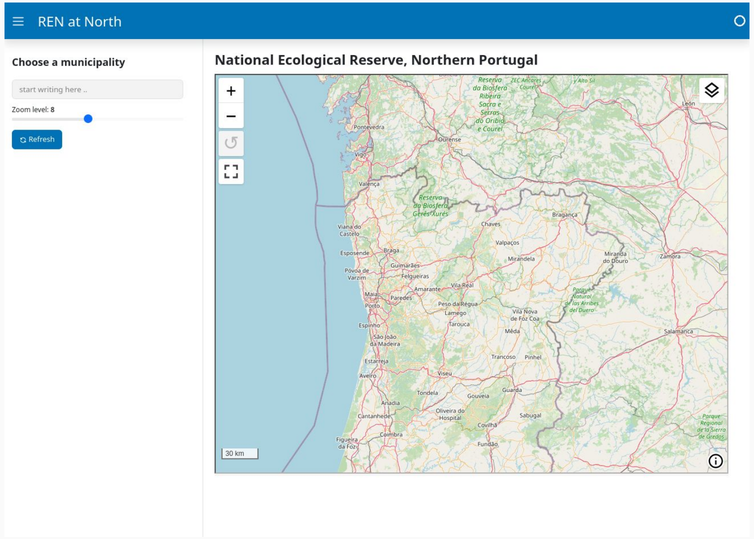
Lat: 41.350010 | Lng: -8.026886 [EPSG:4326 (WGS84)]

https://web-geonorte.ccdr-n.pt/#/app/REN

Day three, a MapML webapp

Using MapML, a web map client app was built, offering:

- a text search of muncipality by name (or part of it);
- on clicking over “refresh” button, occurs a zoom to the chosen municipality and, given time, a raster layer of “ecological reserve” is appended to map;
- when choosing another municipality, previous REN raster theme is removed;
- zoom level can be regulated with a slider.



Web app details

Base components:

- Containerization engine (podman was used, docker may be used instead)
- Holoviz Panel Python web app framework
- MapML Maps4HTML web map custom element polyfill (JavaScript reference implementation)

Other:

- Some JavaScript and Python code

Web app source code

Example source code can be found in GitHub at:

https://github.com/rpcavaco/panel_mapml

Container image preparation

Using podman, a container image can be created, running:

```
podman build -t rpcavaco/mapml_panel -f ./Dockerfile
```

Instead of ‘podman’, ‘docker’ can be used.
An image called **rpcavaco/mapml_panel** is created (other name can be chosen).

This image contains Python 3.11 and ‘panel’ package installed.

Container preparation

Using podman, a rootless container can be created running this command line without elevated permissions:

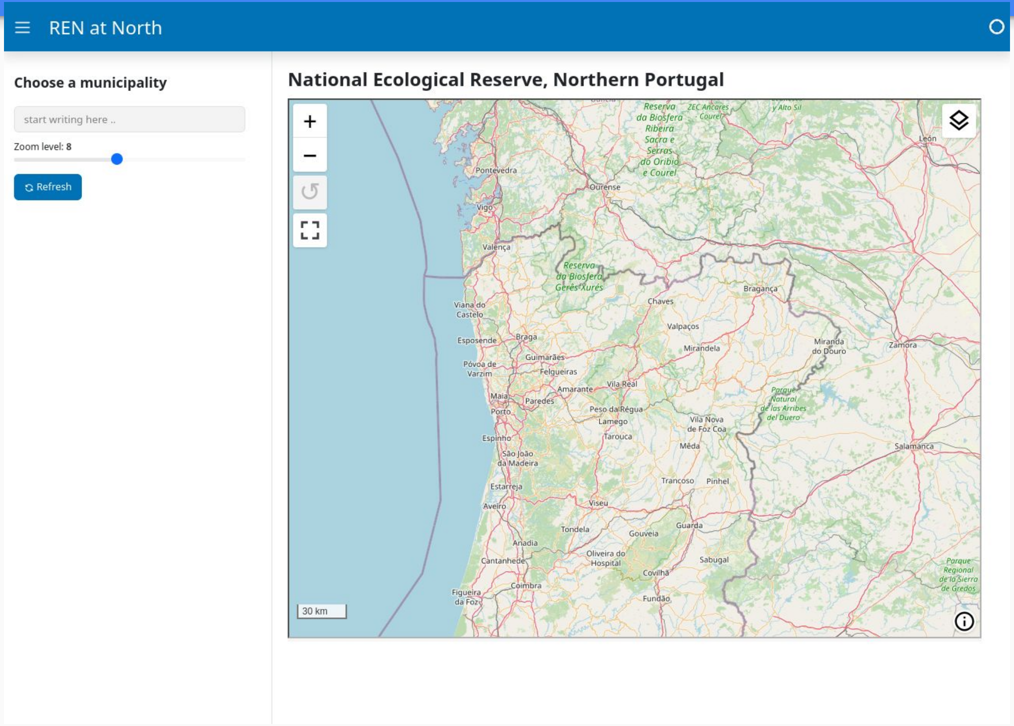
```
podman run \
   --name mapml_panel_cont \
   -p 8090:8090 \
   -v <some local directory>/app:/home/app:Z \
   --detach rpcavaco/mapml_panel
```

- Instead of ‘podman’, ‘docker’ can be used.
- Can user other names for container.
- <some local directory> must be substituted by a real local directory.
- ‘Z’ flag must be used on OSes with SELinux activated.

Web app running

When container is running, the app is available on

http://localhost:8090

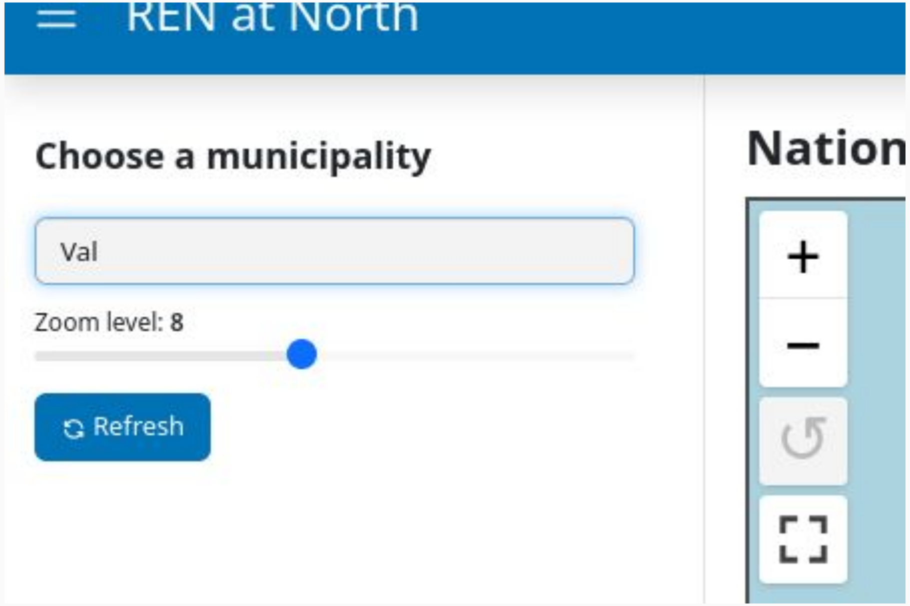


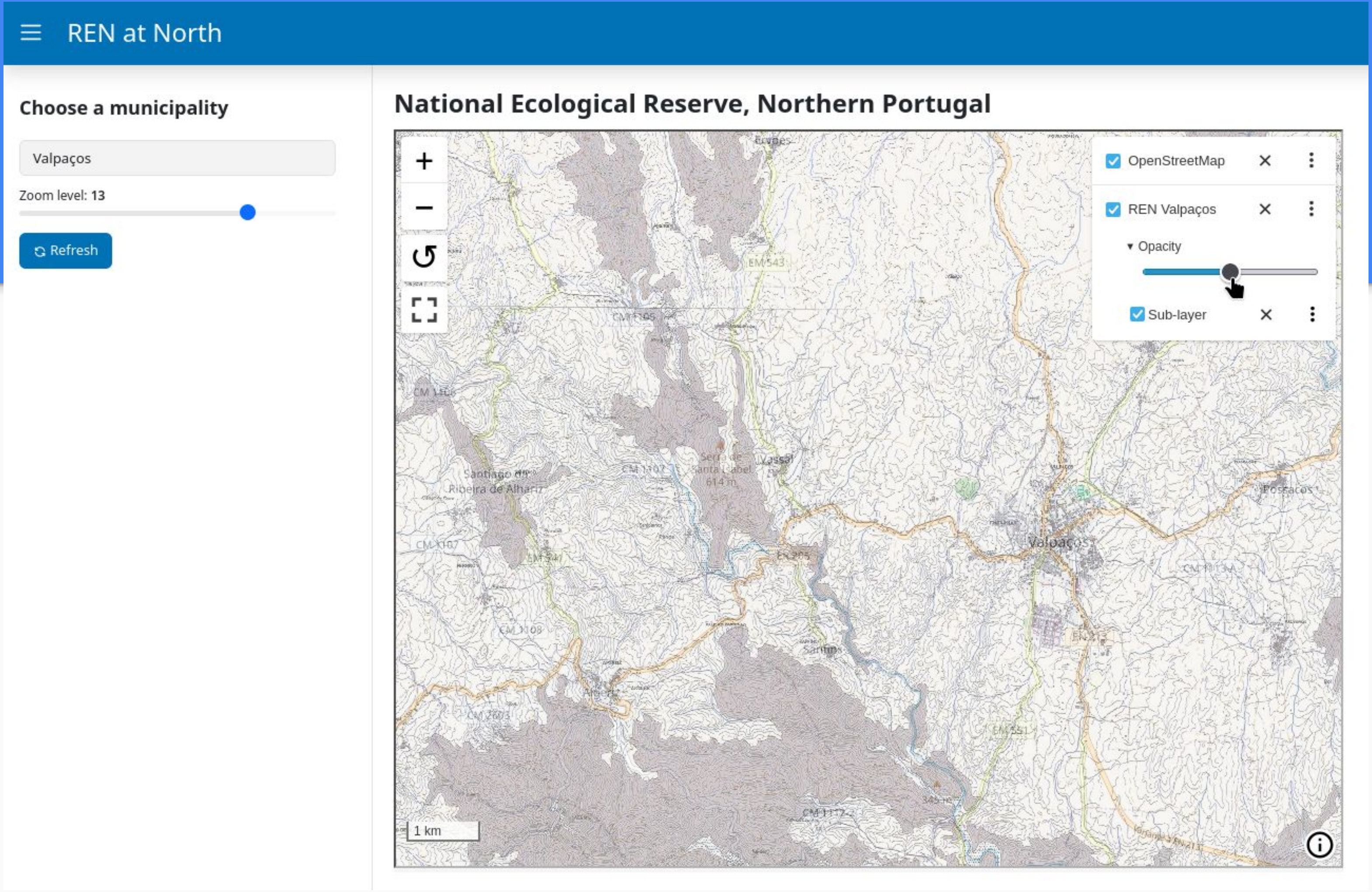
Web app running

Start entering a name in autocomplete text input.

Regulate desired zoom level in slider.

Press ‘Refresh’





REN at North

Choose a municipality

Valpaços

Zoom level: 13

Refresh

National Ecological Reserve, Northern Portugal

Base data

Municipality data is read from a JSON data file (municip_data_final.json).

File contains WMS layer name, lat long for zooming in, Web Mercator box values to delfine a new ‘map-meta’ element.

```
{
    "Vizela": {
        "acronym": "VIZ",
        "wmslayer": "Portaria_23_2016_10FEV" ,
        "lat": 41.3733196401036 ,
        "lon": -8.295616549197314 ,
        "xmin": -927078.8324369721 ,
        "ymin": 5062779.520456224 ,
        "xmax": -917955.9229061622 ,
        "ymax": 5072380.30066498
    },
    "Amarante": {
        "acronym": "AMT",
        "wmslayer": "REN_Amarante_Aviso_308_2017" ,
        "lat": 41.27253023228803 ,
        "lon": -8.042074401776963 ,
        "xmin": -914635.8298312958 ,
        "ymin": 5040324.512158425 ,
        "xmax": -878761.282353797 ,
        "ymax": 5067841.467180247
    },
    "Amares": {
        "acronym": "AMR",
        "wmslayer": "Ren_AMR",
        "lat": 41.65168397589452 ,
```

DOM elements dynamically added on each new layer

```
            <layer- checked>
            <!-- -967000,5011800,-940000,5036000 -->
                <map-meta name="extent" content="top-left-easting=-967000, top-left-northing=5036000, bottom-right-easting=-940000,
bottom-right-northing=5011800"></map-meta>
                <map-title>REN VNG</map-title>
                <map-extent units="OSMTILE"  checked>
                    <map-input name="z" type="zoom" value="18" min="4" max="18"></map-input>
                    <map-input name="w" type="width"></map-input>
                    <map-input name="h" type="height"></map-input>
                    <map-input name="xmin" type="location" units="pcrs" position="top-left" axis="easting" ></map-input>
                    <map-input name="ymin" type="location" units="pcrs" position="bottom-left" axis="northing" ></map-input>
                    <map-input name="xmax" type="location" units="pcrs" position="top-right" axis="easting" ></map-input>
                    <map-input name="ymax" type="location" units="pcrs" position="top-left" axis="northing" ></map-input>
                    <map-link rel="image"
tref="https://web-geoserver.ccdr-n.pt/geoserver/geoapp/wms?bbox={xmin},{ymin},{xmax},{ymax}&format=image%2Fpng&service=WMS&request=GetMap&sr
s=EPSG%3A3857&width={w}&height={h}&layers=REN_VNG&transparent=true&version=1.1.1&styles=/">
                </map-extent>
            </layer->
```

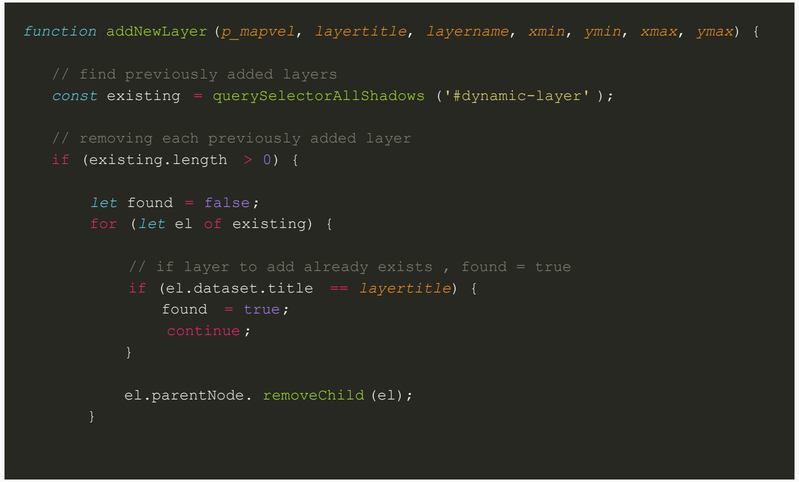
Javascript explanation (assets/app.js)

```
function querySelectorAllShadows(selector, el = document.body) {
   // recurse on childShadows
   const childShadows = Array.from(el.querySelectorAll('*')).
     map(el => el.shadowRoot).filter(Boolean);
    const childResults = childShadows.map(child => querySelectorAllShadows(selector, child));

   // fuse all results into singular, flat array
   const result = Array.from(el.querySelectorAll(selector));
   return result.concat(childResults).flat();
}
```

querySelectorAllShadows function is needed to find elements, because of extensive ‘shadow DOM’ use on Panel framework.

Javascript explanation (assets/app.js)



```
function addNewLayer (p_mapvel, layertitle, layername, xmin, ymin, xmax, ymax) {

   // find previously added layers
   const existing = querySelectorAllShadows ('#dynamic-layer' );

   // removing each previously added layer
   if (existing.length  > 0) {

       let found = false;
       for (let el of existing) {

           // if layer to add already exists , found = true
           if (el.dataset.title  == layertitle) {
               found  = true;
               continue;
           }

           el.parentNode. removeChild (el);
       }
```

function *addNewLayer*

- removes layer with id=dynamic-layer, if exists
- executes commands to add new MapML tags to DOM