Workshop 3 – Web mapping application

Objectives

1. Create some points with local information for the visitors of you study area
2. Export data in .GeoJSON format from GIS
3. Create a basic Leaflet web map with your own data, parameters and content

Data

Use own data (from workshop 1) if you have mapped enough or the geomorphological data that you'll find in the resource folder. In the resource folder, there is an example of the shapefile for the new point features.

Tutorial

From GIS to Web application

Create GeoJSON files

GeoJSON is a Javascript format to store geographical data along with other information. Specification can be found here: http://geojson.org/

For this exercice, you will need three shapefiles:

1. geomorphological surfaces (polygons) with PROCESS and FORM attributes
2. geomorphological landforms (lines) with PROCESS and FORM attributes
3. points of interests (points) with DESCR and ICON attributes. DESCR stores information about the point (write simple text or HTML); ICON stores the name of the symbol you want to use for this point.

Export each layers (Layer/Save as) with GeoJSON format and CRS **WGS84**. As Leaflet uses a Mercator projection system by default, data must be reprojected from CH1903/LV3 to WGS84 (EPSG:4326).

Add the GeoJSON layers to web application

Just put the 3 .geojson files in the /geodata folder. In the script.js file, you have to replace the GP_something.geojson by the name of your three files. For example $.getJSON('geodata/**point_GP**.geojson' will become $.getJSON('geodata/**name-of-point-layer-file**.geojson'

Give specific icons to your point markers

Find your own .PNG picture or use the proposed ones. Every icon .PNG file should be put in the /icons folder. The name of the icon is the same as the ICON attribute value in the shapefile. Of cours you can use different icons.

Adapt map's parameters

The Leaflet map object is created in script.js file using L.map functions. The functions uses different parameters. Let's make your own map with some personalization...

```
var map = L.map('map', {
    center: [46.288, 7.234], //coordinates of map center when loaded (decimal degrees - WGS84)
    zoom: 12, //zoom level when map is loaded
    maxZoom: 16, //maximum zoom level
    minZoom: 11, //minimum zoom level
```

```
    maxBounds: [[46.2, 7.1], [46.35, 7.4]], //add limits to your map: the user cannot drag the map outside those limits
    layers : [couche_osm] //define the default background you want to show when map is loaded
});
```

Just change the numerical values by your own coordinates and zoom levels. You can find WGS84 coordinates in decimal degrees on QGIS or on Google Earth.

Add different background layers

Connect to https://leaflet-extras.github.io/leaflet-providers/preview/ to find other background layers. You just have to copy the given code to your script.js file (on the top of the page).

Code will be like:

```
var my_new_layer = L.tileLayer('http://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}', {
            attribution: '&copy; Esri &mdash; <a href="http://www.arcgis.com/home/item.html?id=c1c2090ed8594e0193194b750d0d5f83" target="_blank">Références complètes des sources</a>'
});
```

my_new_layer is the name of the layer. It can be changed as you like.

L.tileLayer is the function to create the background layer and **attribution** is the references of this layer.

Then you have to add this new layer to the background layer selector:

```
var selecteur_couches = {
    "Base map": couche_osm,
    "Satellite": couche_sat,
    “nice name for the layer”:  my_new_layer
};
```

You can give a nice name to this layer (it will be the name the user will see on the web page). Don't forget the “ and the , !

Adapt your web page

You can add text, links, pictures and so on In the index.html file in the **header** or in the **footer** elements.