Visualising VGI data with Leaflet

Dr. Peter Mooney

There's no need to be afraid but there WILL be computer code in this lecture

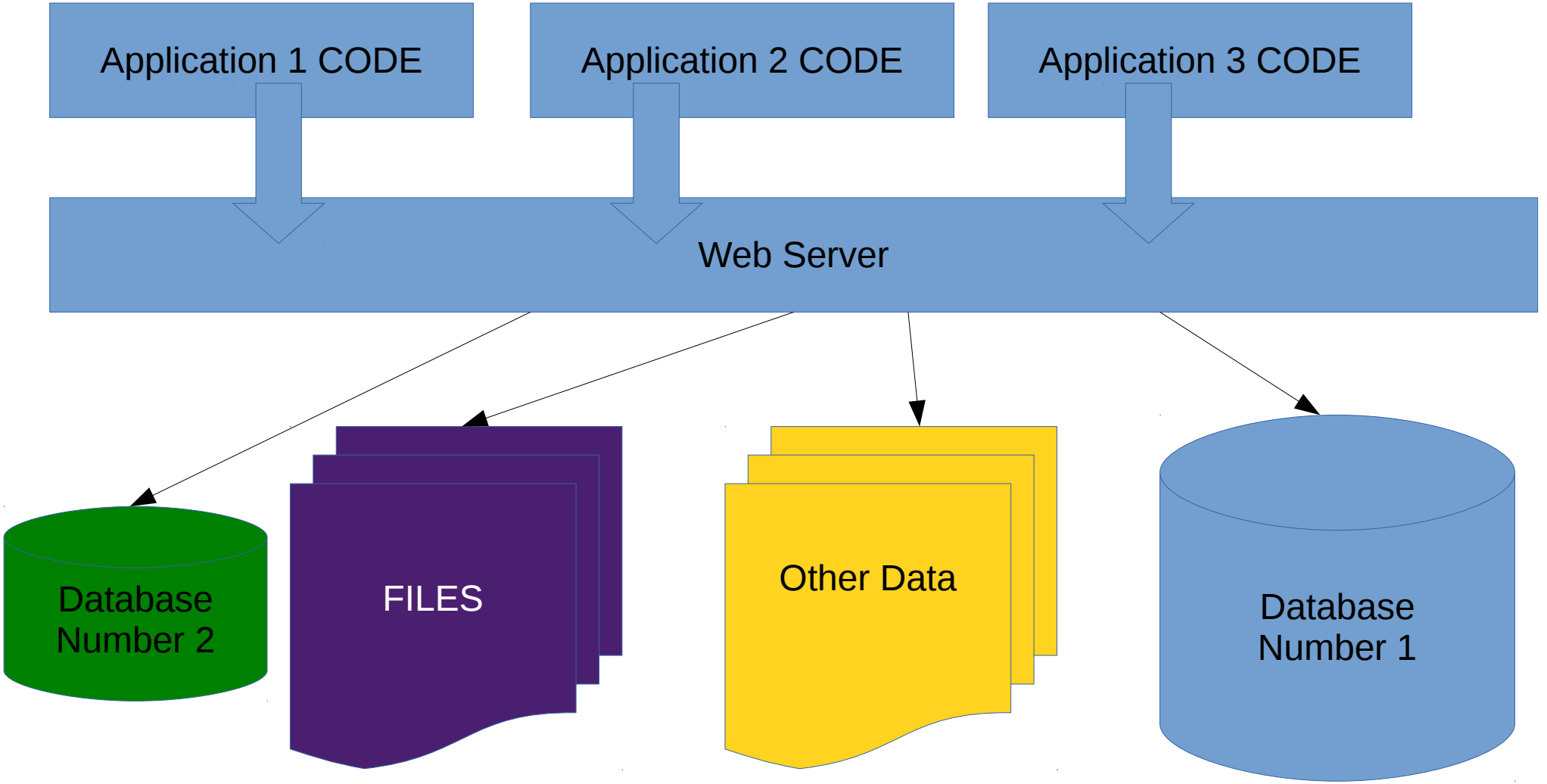
Aims of this lecture today

- To introduce you to some basic concepts of Visualising VGI data (spatial) data on a Web-based Map
- After the lecture you should be able to create simple visualisations of VGI data on web-based maps using Open Source Components
- You will see that much of the work in Visualisation of VGI comes from preparing the VGI for display on the web-based maps

We are going to need to have a web-server to run some of the examples today

- A web server is a special piece of software which provides the software technology to make data and content available on the Internet
- There are millions of web servers running around the world but very often we don't even think about them or what they do!

A very simplistic view of what a web-server actually does



Installing a webserver for this lecture

We will need a web server for some of the more advanced examples

- There are LOTS of choices for this.
- The BEST solution (in my opinion) would be to use Apache Server
- It is probably the best web-server available (and it is Open Source)
- But it can be tricky to install when we are dealing with lots of different machines and operating systems.

Download Mongoose Free

https://cesanta.com/mongoose.shtml

129825 downloads since 2014-03-05

Download Mongoose Free Edition

- Stable and mature: over 1 million downloads since 2004
- Cross-platform: works on Mac, Windows, UNIX/Linux
- No installation or configuration required: runs in one click
- IP-based ACL (deny access to certain IP addresses)
- File blacklist (hide certain files from serving)
- Resumed download support

-- download for free --
Windows Executable
Windows Mongoose+PHP Package
Windows Mongoose+Lua Package
MacOS Installer
Linux x86_64 Executable

-- download for free --

Download Mongoose Pro Edition

- All features of the Free edition, plus:
- CGI (ability to run sites written in PHP, Ruby or any other scripting language)
- Server Side Includes (include one HTML page into another)
- WebDAV (attach shared directory as a remote drive)
- Pre-packaged, ready-to-go Mongoose + PHP bundle for Windows (start PHP project in seconds)
- Life-time software updates

ersonal and Commercial use.
bution not allowed.
ccept Mongoose EULA
uy for $5 --

Download Mongoose Dev Edition

- All features of the Pro edition, plus:
- Ability to view and analyze all incoming and outgoing network packets

- Experimental HTTPS security (TLS1.2) support

For personal and Commercial use.
Distribution not allowed.

I accept Mongoose EULA
-- buy for $8 --

I accept Mongoose EULA
-- buy for $15 (with SSL) --

Documentation

Licensing

Mongoose – Windows Users

- Download the EXE file
- Move the EXE file to a new folder anywhere on your computer (*call the folder www or public_html*)
- Ensure that when you run the file that you do so as administrator (check that Windows has not prevented you from running it)
- Check in your browser that the URL http://127.0.0.1:8080 or http://localhost:8080 work

Mongoose – Linux Users (1)

- Download the Linux executable
- Move this file to a new folder within your workspace – call the folder www or public_html
- You will need to make this file executable
- You will need to type the command **sudo chmod ugo+x mongoose-lua-sqlite-ssl-static-x86_64-5.2** in a terminal window
- Click on the file to run it
- Check in your browser that the URL http://127.0.0.1:8080 or http://localhost:8080 work

Windows and Linux Users

- DO NOT DELETE the Mongoose file
- As this is just a TEST ENVIRONMENT it is OK here – but this technique should not be used in a production environment
- However that's a different set of lecture notes!

We cannot proceed beyond this point unless you have Mongoose running successfully!

Which browser should you use for the lecture today?

Safari

We are also going to need to have a nice text editor installed

- A text editor is one of the most important tools which a web developer uses
- Like web servers there are LOTS of options.
- We need a text editor which will give us some user friendly editing functionality for HTML and Javascript

Suggested Text Editors

- **ANY ONE OF .....**
- **The Geany**
- **Bluefish**
- **Notepad++**
- **Sublime**
- Perhaps Bracket or TextMate for MAC specifically
- The links are available on the GitHub page

What web technologies are we using today for the lecture?

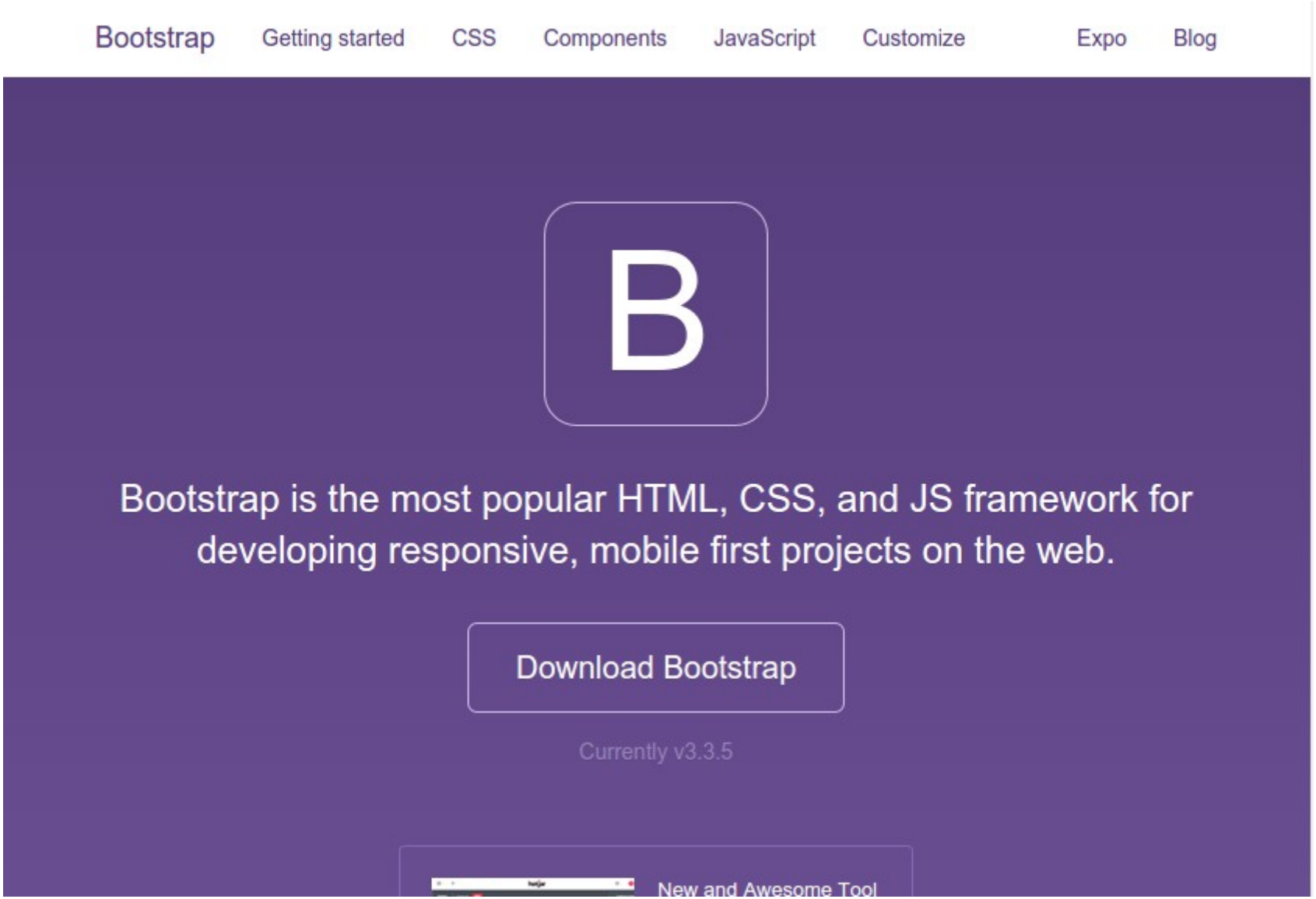
We are using Leaflet for our map container Javascript

an open-source JavaScript library
for mobile-friendly interactive maps

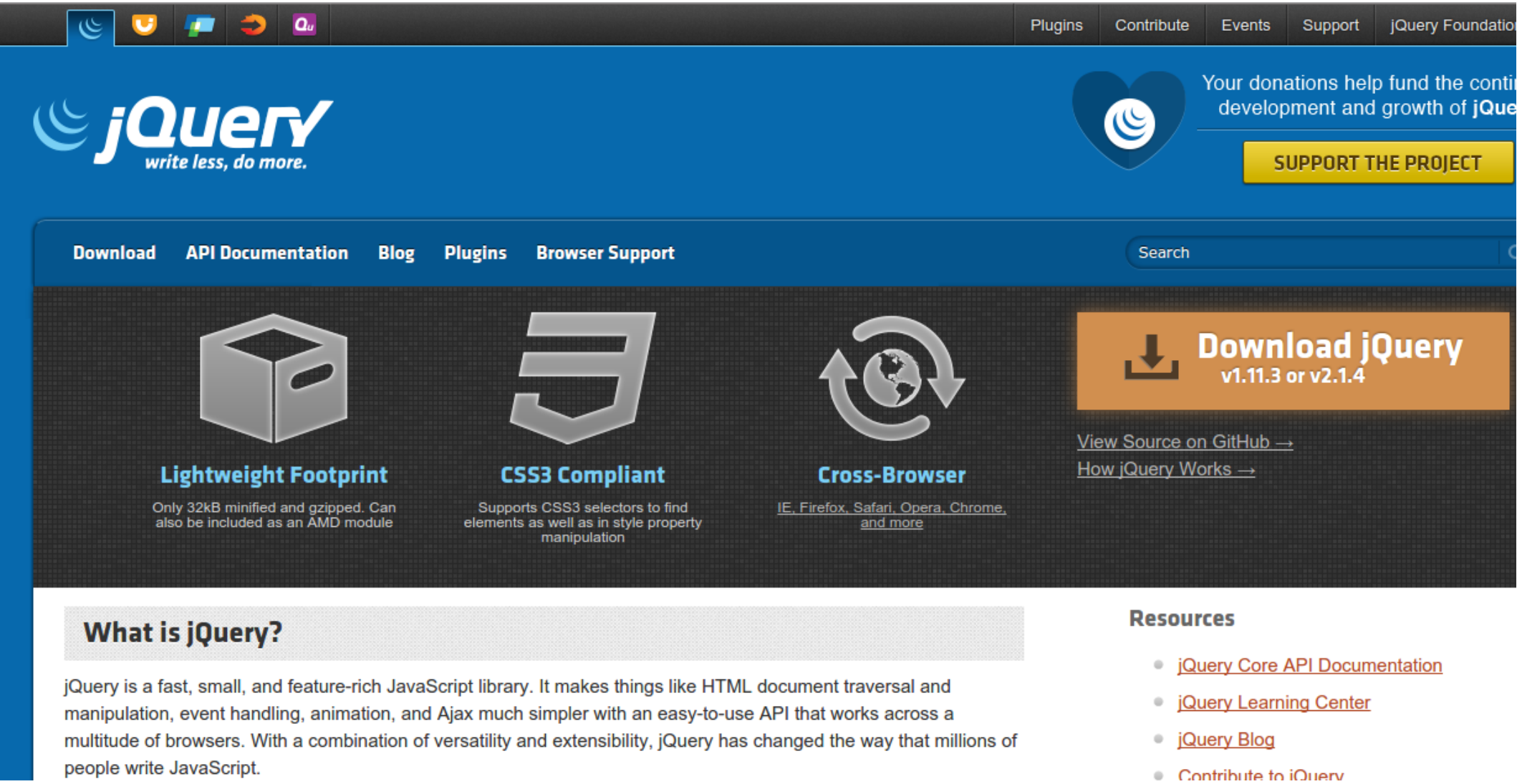
Overview Tutorials Docs Download Plugins Blog

Leaflet is the leading open-source JavaScript library for mobile-friendly interactive maps. Weighing just about 33 KB of JS, it has all the mapping features most developers ever need.

We are using Bootstrap to make our pages display in a responsive way



Jquery provides LOTS of very useful Javascript functionality

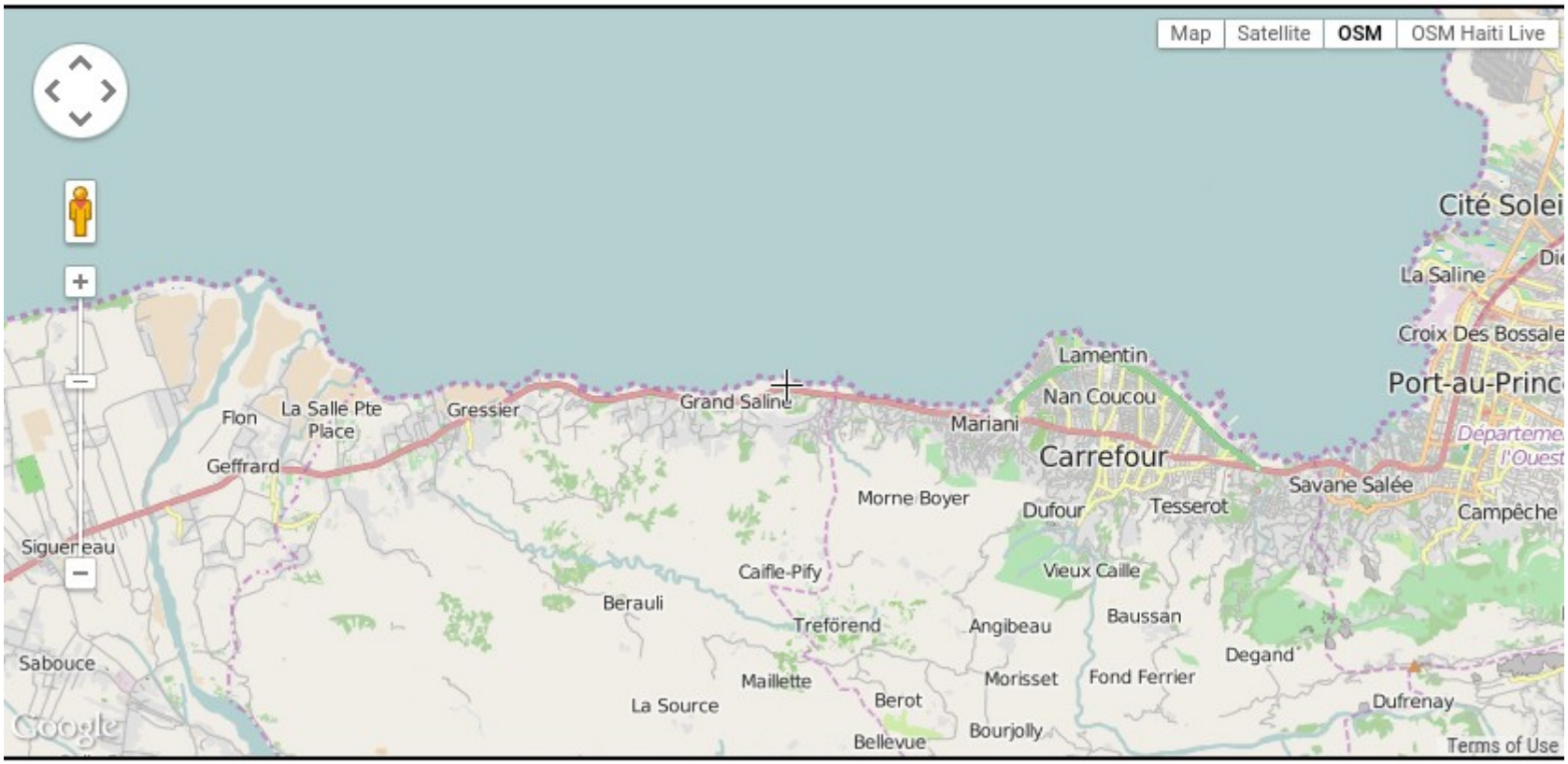


Finding the Latitude Longitude of any point will be very useful today

Get Lat Lon

Find the latitude and longitude of a point on a map.

Place name: Zoom to place

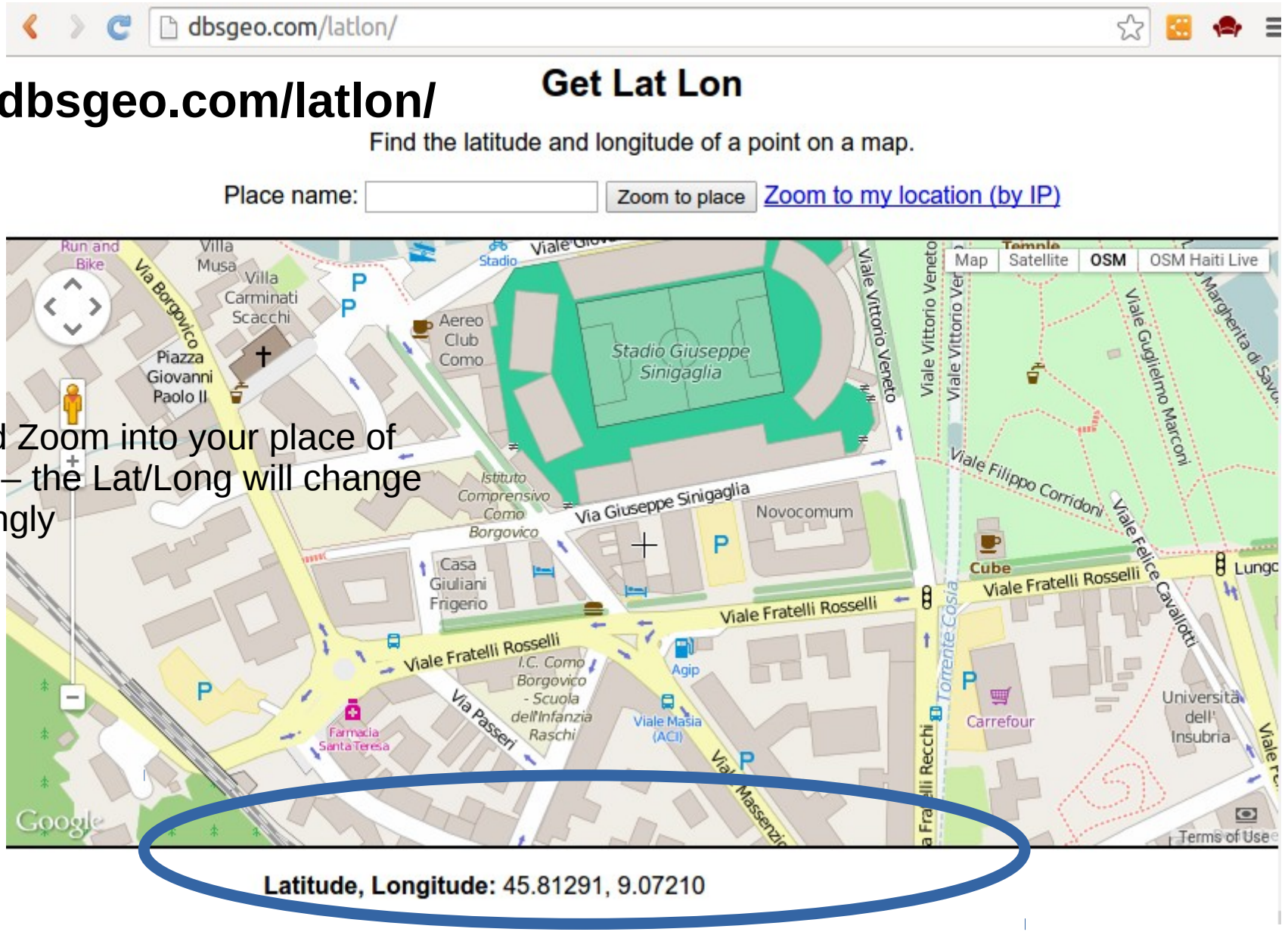


Latitude, Longitude: 18.54732, -72.46788

Getting Started

Get the Latitude Longitude of ANY point on the earth

http://dbsgeo.com/latlon/



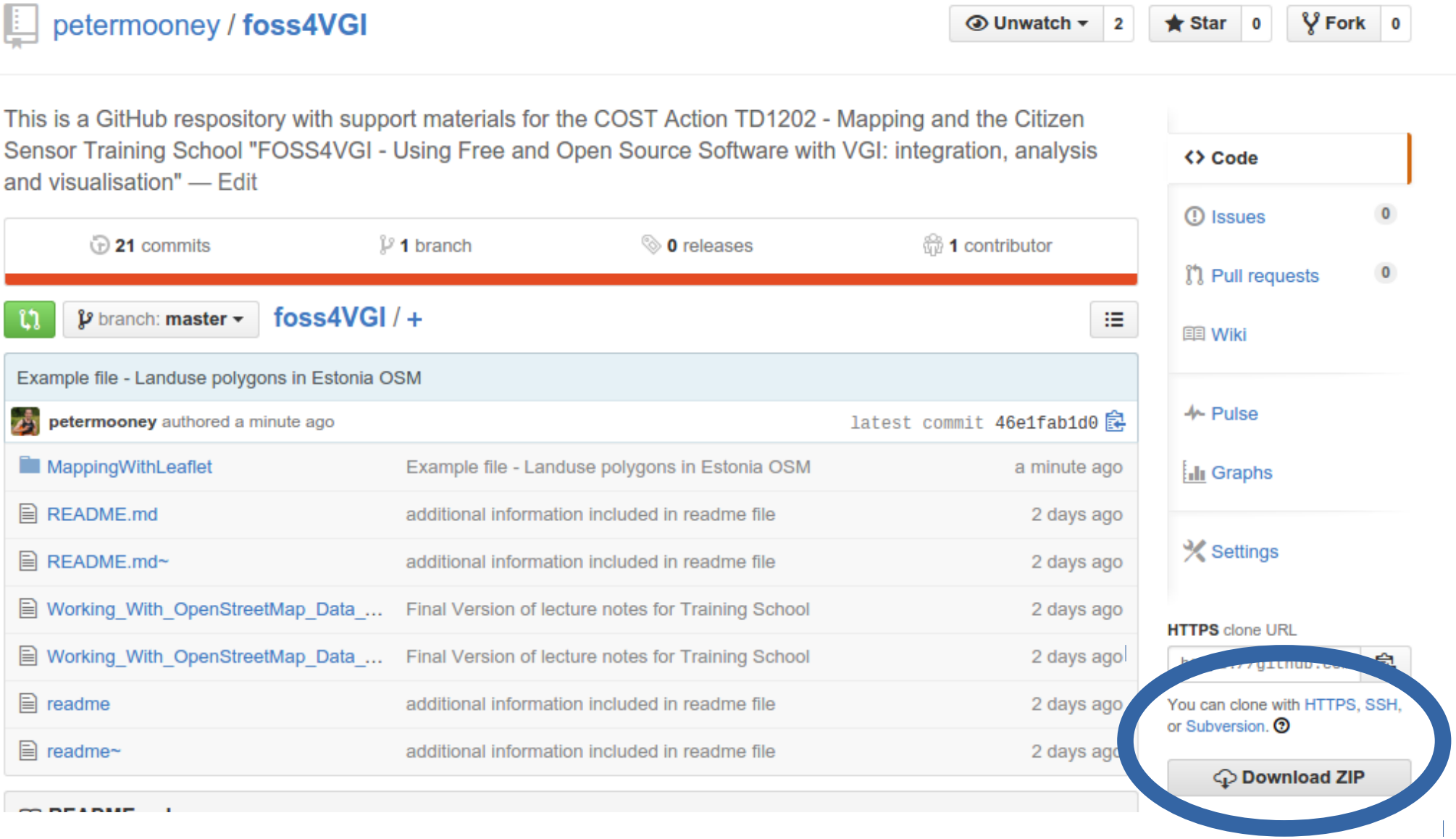
Pan and Zoom into your place of interest – the Lat/Long will change accordingly

Getting our material together

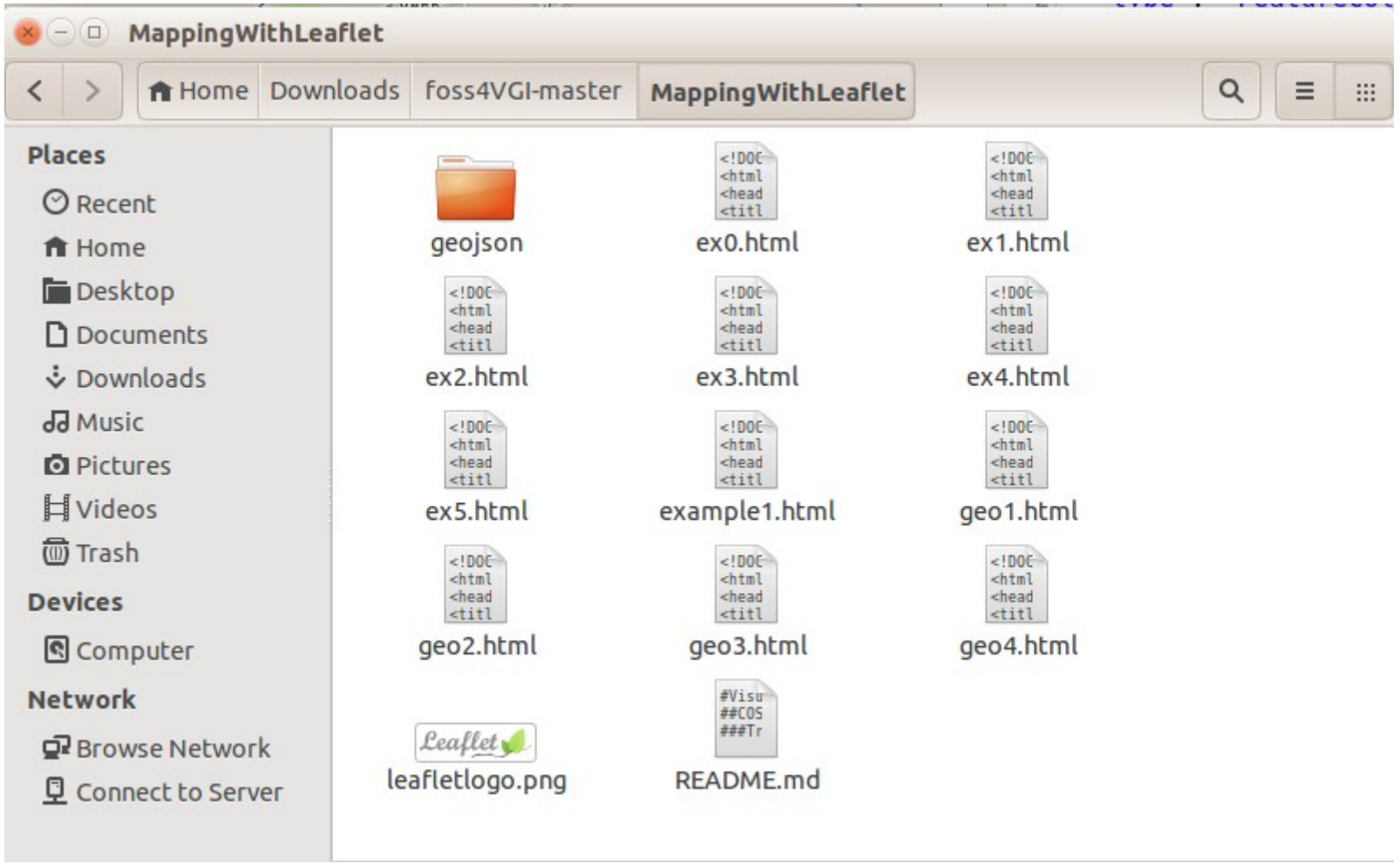
- We will need to download all of the materials from GitHub
- You will unzip these materials into a folder on your computer.
- You then need to copy the Mongoose executable into this folder.
- DO NOT CHANGE the names of any files or folders within this new folder.

Downloading from GITHUB

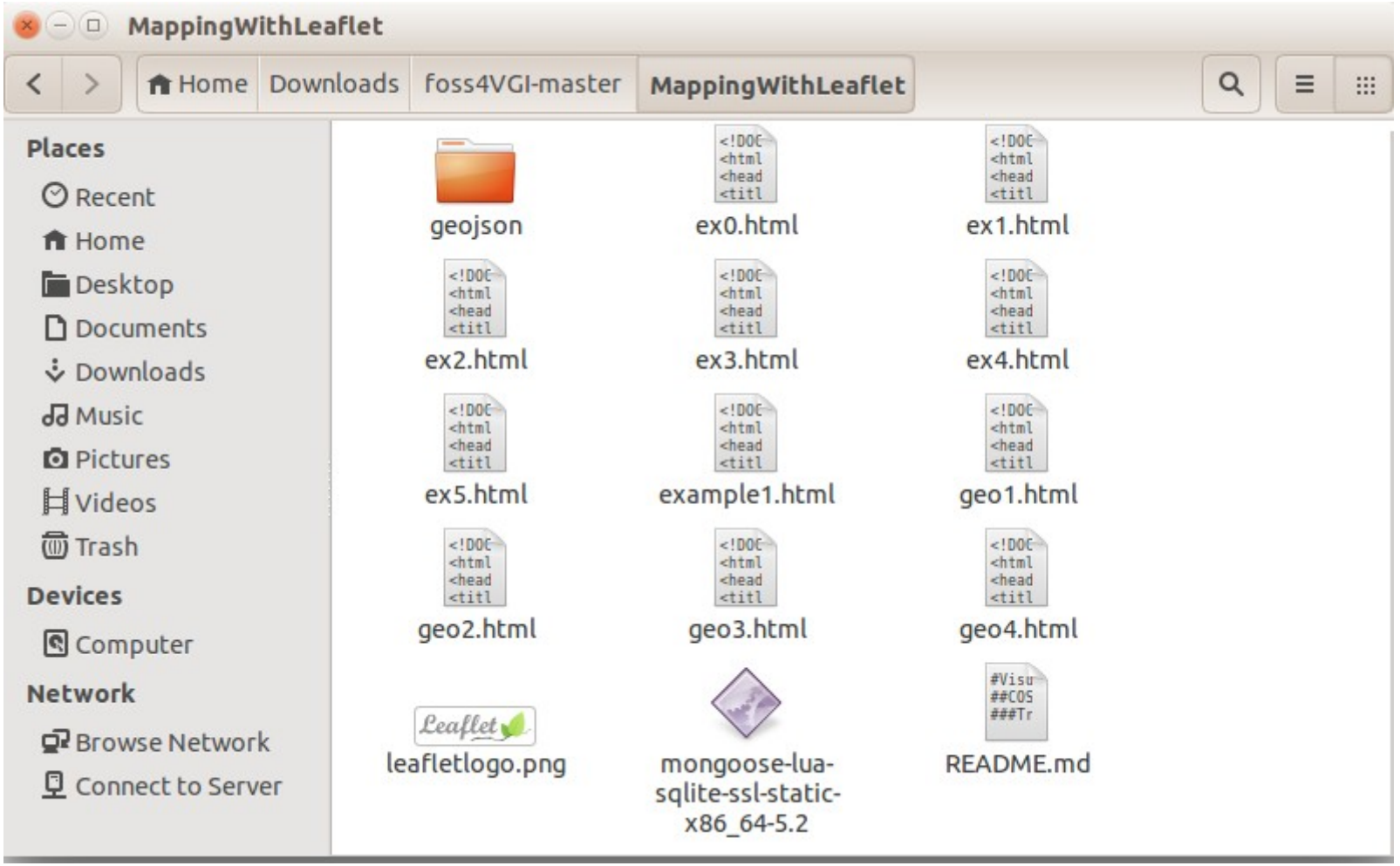
https://github.com/petermooney/foss4VGI



The Unzipped Files in a Folder

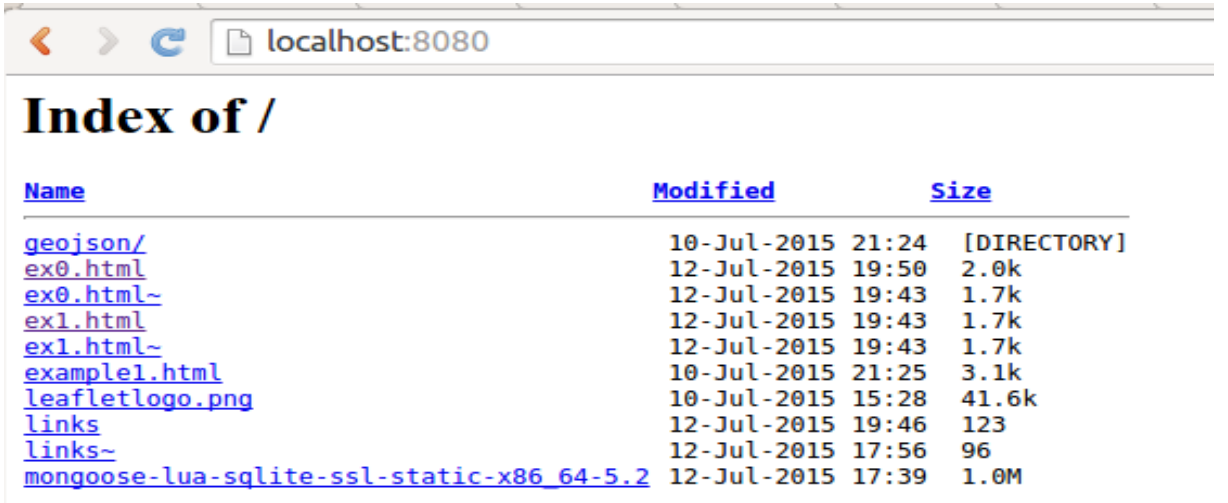


And look – there is the Mongoose Executable Included now!



We have a special web address now that Mongoose is operational

- We can now type in the address http://127.0.0.1:8080 or http://localhost:8080 into your browser
- You should be given a list of the files in the folder.
- Just click on “ex0.html” and wait for the lecture slides to catch up



Every webpage in the world has the same structure regardless of what the webpage actually does

A webpage at it's most fundamental has a <head> and has a <body>

The <body> is what you see in your web-browser. The <head> helps to support the technical details of how the <body> is managed and displayed.

Unfortunately – we do not have enought time to go into the deeper details of web page development

- So we are going to take a few things for granted and make some assumptions.
- In our examples – the <HEAD> of our pages WILL NOT CHANGE except for the <TITLE>
- In our examples – the <BODY> of our pages will change but we are not trying to design very fancy beautiful webpages.
- **The focus today is on DISPLAYING MAPS WITH GEODATA in a webpage and NOT webpage development.**

Let's open ex0.html in your text editor which you installed

- We are going to step through this very carefully so we can see the <HEAD> and the <TITLE> and the <BODY>

The <HEAD> of our Web Pages

```
2  <html>
3  <head>
4      <title>FOSS4VGI - Leaflet Lectures - Example 0</title>
5  <!-- This part of our web page should hardly change over the course of lecture today -->
6  <!-- This part of the web-page tells the web-browser it will need certain functionality
7  to make the page display/function as the programmer intended -->
8  <!-- There should be no need for you to change any of the links below -->
9
10 <meta charset="utf-8" />
11 <meta name="viewport" content="width=device-width, initial-scale=1">
12
13 <!-- Bootstrap CSS -->
14 <link rel="stylesheet" href=
   "https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css">
15
16 <!-- Bootstrap Optional theme -->
17 <link rel="stylesheet" href=
   "https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap-theme.min.css">
18 <!-- JQuery - which is needed by Bootstrap and Leaflet -->
19 <script src="https://code.jquery.com/jquery-1.11.3.min.js"></script>
20 <!-- Latest compiled and minified JavaScript for Bootstrap -->
21 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/js/bootstrap.min.js"></script>
22
23 <!-- Include Leaflet -->
24 <script src="http://cdnjs.cloudflare.com/ajax/libs/leaflet/0.7.3/leaflet.js"></script>
25 <link rel="stylesheet" href=
   "http://cdnjs.cloudflare.com/ajax/libs/leaflet/0.7.3/leaflet.css" />
26
27 </head>
```

The <BODY> of our Web Pages

```
<body>
<div class="container">
    <!-- start the row of content -->
<div class="row">
    <h1>This is Example 0</h1>  <!-- We can change this piece of text each time -->
    <div id="map" style="width: 100%; height: 600px"></div>
</div><!-- close the row of content-->

<script><!-- This is where our mapping code starts -->
// This is our base layer - you should not need to change this in many examples
var OpenStreetMap_Mapnik = L.tileLayer('http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png',
{maxZoom: 19, attribution: '&copy; <a
href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>'});

// Notice how the MAP is centered at a particular point - notice the ZOOM level
var map = L.map('map', {center: [45.81288, 9.07454], zoom: 19,layers: [OpenStreetMap_Mapnik
]});

var baseMaps = {"OpenStreetMap_Mapnik":  enStreetMap_Mapnik};

var overlayMaps = { };

L.control.layers(baseMaps, overlayMaps).addTo(map);

// This is a marker pin on a particular spot. It does not have to be the center
L.marker([45.81288, 9.07454]).addTo(map).bindPopup("<b>Hello Everyone</b><br/>This is the
Cube Cafe in Como.").  Popup();

</script><!-- This is  ere our mapping code ends -->
</div><!-- close the c tainer -->
</body>
</html>
```

Exercise A: Let's put a marker pin on our own house/appartment

- Copy the file ex0.html and rename the file exerciseA.html *(stay in the same folder)*
- Open exerciseA.html in your text editor.
- Using the 'get lat long' web page – find the lat/longitude of your home – change the code in your file so that the marker pin is changed.
- Change also the <TITLE> and the <h1> text
- SAVE the file and then look at http://localhost:8080/exerciseA.html

Let's look at Ex1.html in your text editor – multiple markers

- You can have multiple markers on your map – it's no problem. You can actually have hundreds of marker icons on your map if you need.
- **Decision 1:** When you have more than one marker – you will need to decide where to center the map
- **Decision 2:** You will also need to decide upon the appropriate zoom level
- **Decision 3:** You might need to use a different LAT/LONG for your center

The code for adding more markers to your map is very simple

```
42
43  // Notice how the MAP is centered at a particular point - notice the ZOOM level
44  var map = L.map('map', {center: [45.80880, 9.08367], zoom: 15,layers: [OpenStreetMap_Mapnik]});
45
46  var baseMaps = {"OpenStreetMap_Mapnik": OpenStreetMap_Mapnik};
47
48  var overlayMaps = { };
49
50  L.control.layers(baseMaps, overlayMaps).addTo(map);
51
52  // This is a marker pin on a particular spot. It does not have to be the center
53  L.marker([45.81288, 9.07454]).addTo(map).bindPopup("<b>Hello Everyone</b><br/>This is the Cube Cafe in Como.");
54
55  L.marker([45.81413, 9.08407]).addTo(map).bindPopup("<b>Hello Train Users</b><br/>This is the Como Lago Nord train
    station.");
56
57  L.marker([45.80168, 9.08913]).addTo(map).bindPopup("<b>Hello Shoppers</b><br/>This is the Carrefour Supermarket in
    the south of Como city.");
58
59
```

NOTICE that the CENTER of the map is different to the locations of the 3 markers which we have added

Exercise B: Let's put multiple marker pins on a map using our own data

- Copy the file ex1.html and rename the file exerciseB.html *(stay in the same folder)*
- Open exerciseB.html in your text editor
- Use Get Lat Long to find the locations of FOUR TRAIN STATIONS close to YOUR HOME
- You will need to also test and find an appropriate center for your map.
- You will need to fina an appropriate zoom
- Result at http://localhost:8080/exerciseB.html

Changing the Background Layer

- In the previous examples we have used the default OpenStreetMap MAPNIK layer
- BUT the great thing about Leaflet is that we can easily change the background layer

You will need to change 4 pieces of information

```
<script><!-- This is where our mapping code starts -->
// This is our base layer - you should not need to change this in many examples
var OpenStreetMap_Mapnik = L.tileLayer('http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {maxZoom:
19, attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>'});

// Notice how the MAP is centered at a particular point - notice the ZOOM level
var map = L.map('map', {center: [45.80880, 9.08367], zoom: 15,layers: [OpenStreetMap_Mapnik]});

var baseMaps = {"OpenStreetMap_Mapnik": OpenStreetMap_Mapnik};

var overlayMaps = { };

L.control.layers(baseMaps, overlayMaps).addTo(map);
```

1

2

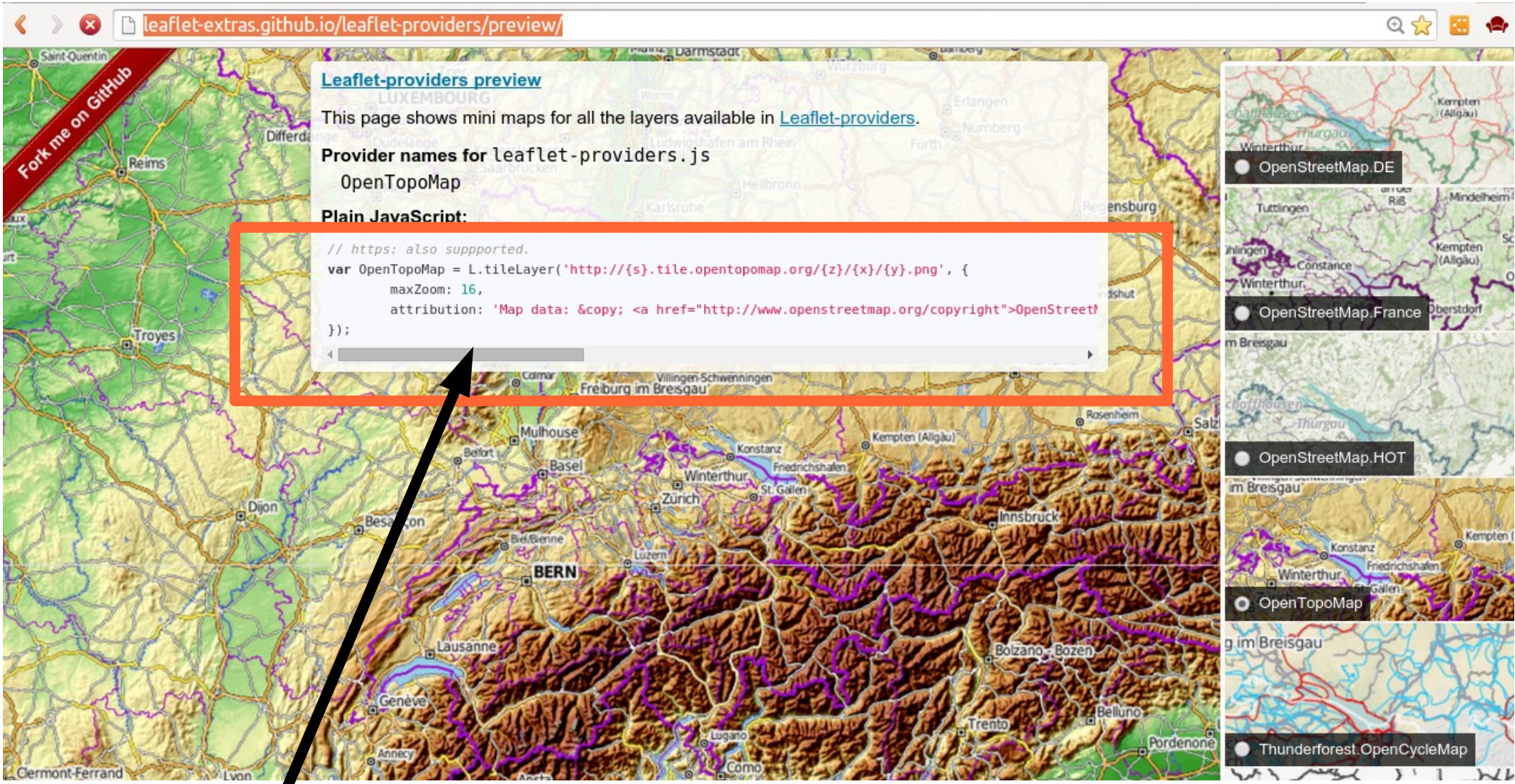
3

4

Choosing a background layer

- We are going to look at some choices of background layers
- REMEMBER
- You will choose a layer which is appropriate to your data – you need to make sure that you users can easily understand the data and understand the significance of the background layer
- GO TO http://leaflet-extras.github.io/leaflet-providers/preview/

Leaflet Layer Providers



YOU WILL NEED TO COPY THIS EXACTLY

Exercise C: Changing the background layer in Leaflet

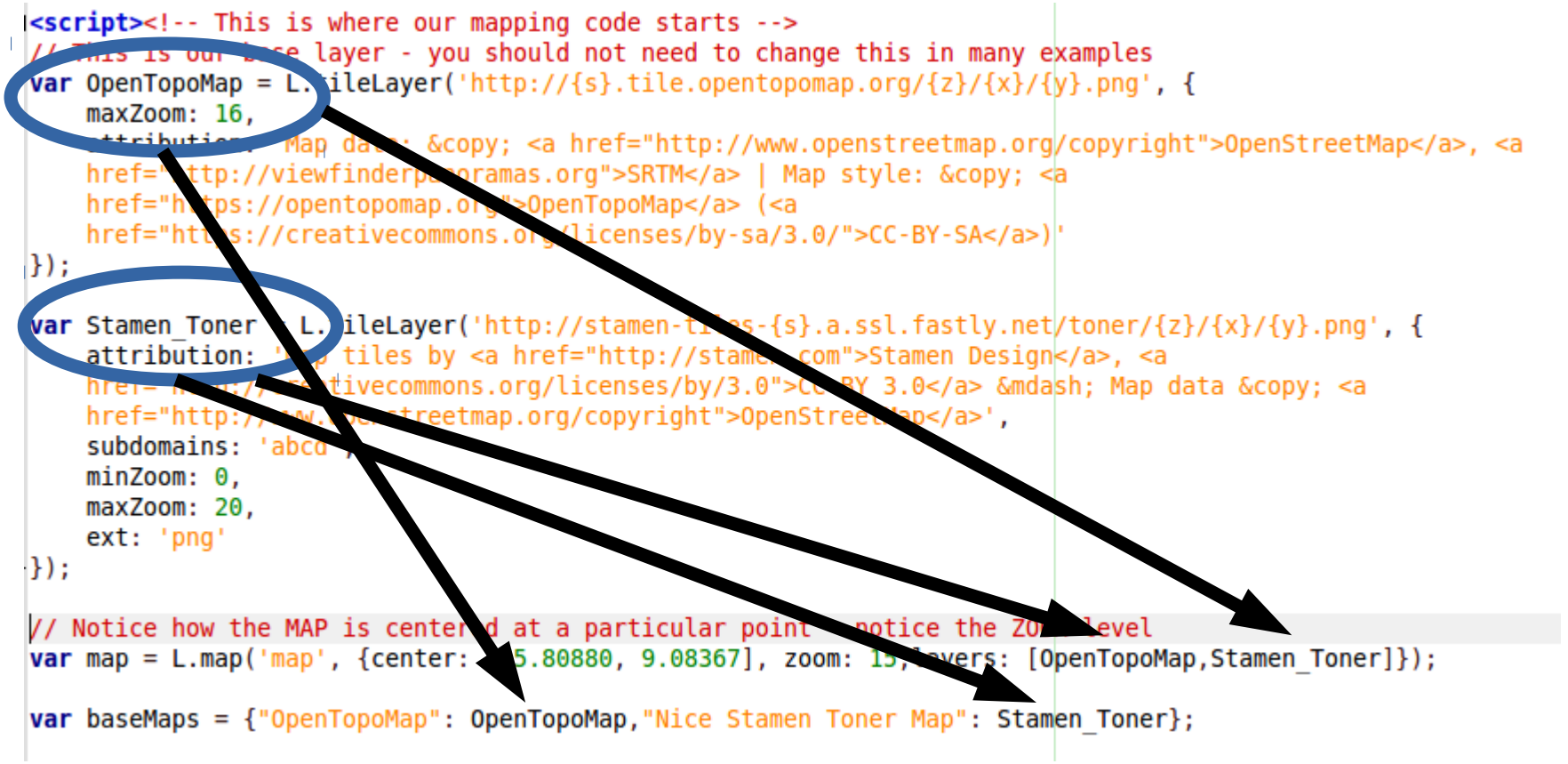
- Copy the file excerciseB.html and rename the file exerciseC.html *(stay in the same folder)*
- Open exerciseC.html in your text editor
- Pick your favourite layer from the Leaflet Layer Providers page
- Copy this code and REPLACE the Mapnik Definition. You will also need to change the line containing 'var baseMaps'

This is one example of a different background layer

Exercise D – Having Multiple Background Layers

- It's possible to have several different background layers which the user can easily switch between.
- Let's look at ex3.html in your text editor
- We just use Leaflet Layer Providers to get the code for our new layer
- Let's look and see where the new code fits in and then we'll test it.

Exercise D: Having Multiple Background Layers



Exercise D: Making a map with THREE background layers

- Copy the file ex3.html and rename the file exerciseD.html *(stay in the same folder)*
- Open exerciseD.html in your text editor
- Use Leaflet Map Providers page to choose THREE different background layers.
- BE CAREFUL – You will need to make the changes to your code as shown
- Look at the result in your browser at http://localhost:8080/exerciseD.html

Adding Polygons to Leaflet

It is easy to add polygons to Leaflet using similar code to that of points

```
var polygon1 = L.polygon([
    [51.509, -0.08],
    [51.503, -0.06],
    [51.502, -0.05],
    [51.51, -0.047],
    [51.509, -0.08]
]).addTo(map);
```

REMEMBER: A polygon is a closed object – so the start and end nodes must be the same!

Ex4: Adding a polygon

- We can use a tool such as 'Get Lat Long' to help us get the points we need to make up a polygon. Look at ex4.html in your text editor

```
// Notice how the MAP is centered at a particular point - notice the ZOOM level
var map = L.map('map', {center: [45.80880, 9.08367], zoom: 15,layers: [OpenStreetMap_BlackAndWhite,MapQuestOpen_OSM
]});

var baseMaps = {"OSM BW": OpenStreetMap_BlackAndWhite,"MapQuest OSM":

var overlayMaps = { };

L.control.layers(baseMaps, overlayMaps).addTo(map);

var polygon1 = L.polygon([
[45.81424, 9.07097],
[45.81460, 9.07350],
[45.81339, 9.07386],
[45.81307, 9.07144],
[45.81374, 9.07068],
[45.81424, 9.07097]
]).addTo(map);
polygon1.bindPopup("I am a polygon showing the outline polygon for the Stadio Giuseppe Sinigaglia in Como");
```

Exercise E: Adding more Polygons

- Copy ex4.html and rename it as exerciseE.html
- **AIM:**
- Use 'Get Lat Long' to help create TWO NEW POLYGONS (you can fully delete the polygon which is already in ex4.html)
- The TWO NEW POLYGONS will represent TWO Educational facilities in your town, city or Region. You will need to find an appropriate center for the map. You will need to provide text for the popup windows.

OK! Let's take a break from Polygons for a moment

Let's create our very own simple version of Get Lat Long for ourselves

Ex5.html – Open this file in your text editor and view it in the browser

- Open Ex5.html in your text editor
- Check out http://localhost:8080/ex5.html in the browser
- **We have added some code – but it is a really useful piece of code**

```
57  // This is the where Javascript is connected to the map when you 'click' on the map
58  map.on('click', onMapClick);
59
60  // THIS IS NEW. This is a special FUNCTION from Javascript.
61  // Leaflet has lots of functionality that we can use. This is very useful to display information about
    the map
62  function onMapClick(e) {
63      var theDateAndTime = new Date();
64      var mapCenter = map.getCenter();
65      $("#mycoordinates").html("<p class = 'lead'>MyGetLatLong Says:<br/> At " + theDateAndTime +
        "<br/>You clicked the map at Coordinates " + "(Lat, Lon) = " + e.latlng.lat + "," + e.latlng.lng +
        "<br/>Also the map is at Zoom level "+ map.getZoom() + "<br/>The CENTER of the map is currently at
        Coordinates " + "(Lat,Lon) = " + mapCenter.lat + "," + mapCenter.lng + "</p>");
66  }
67
```

This is the output from Ex5.html

localhost:8080/ex5.html

MyGetLatLong Says:

At Mon Jul 13 2015 17:10:54 GMT+0100 (IST)

You clicked the map at Coordinates (Lat, Lon) = 45.808823378560895,9.085154533313471

Also the map is at Zoom level 15

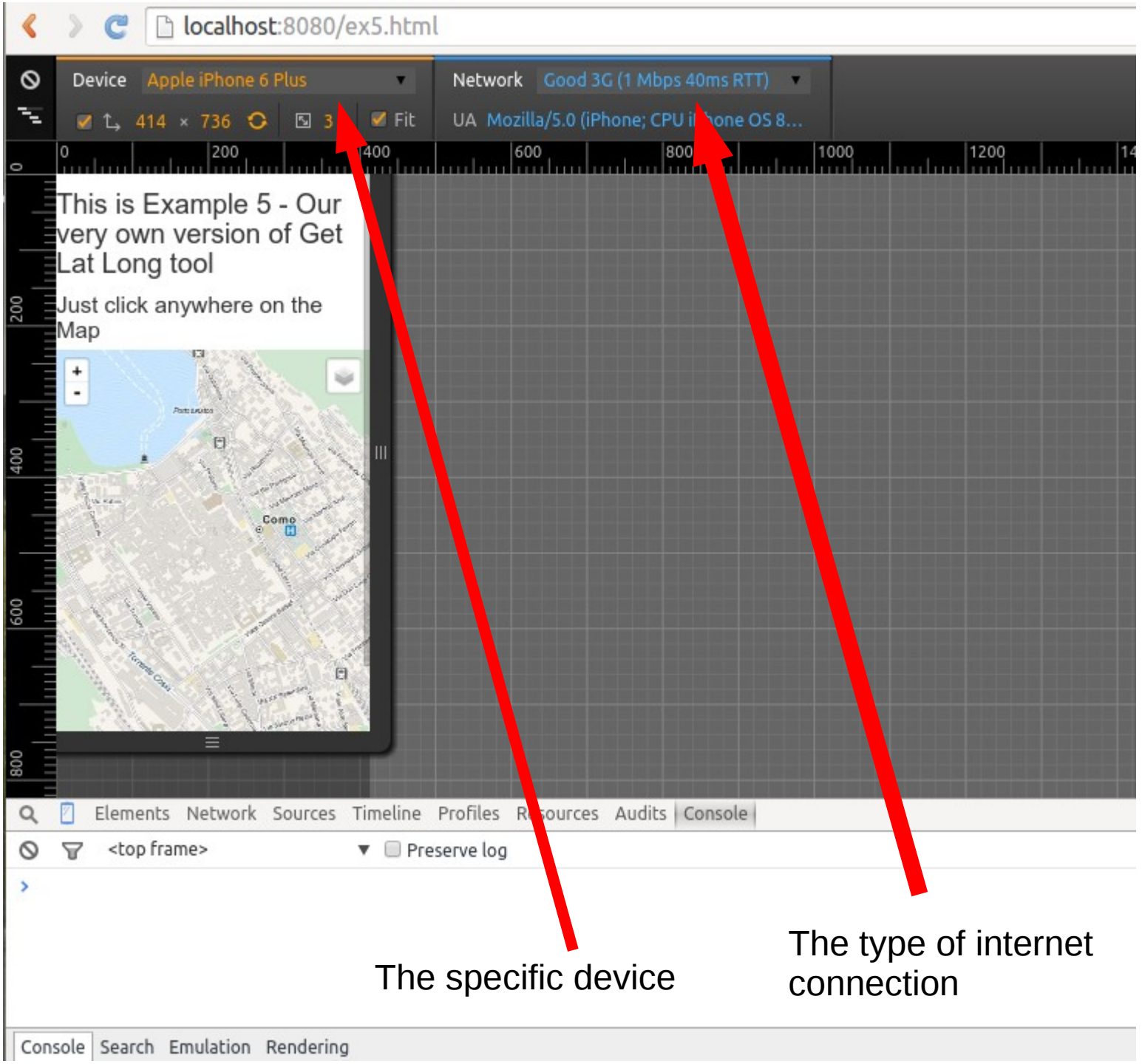
The CENTER of the map is currently at Coordinates (Lat,Lon) = 45.808803435510455,9.081478118932864

How does all this look on your mobile phone?

Can we test how this will all look on your smartphone or tablet?

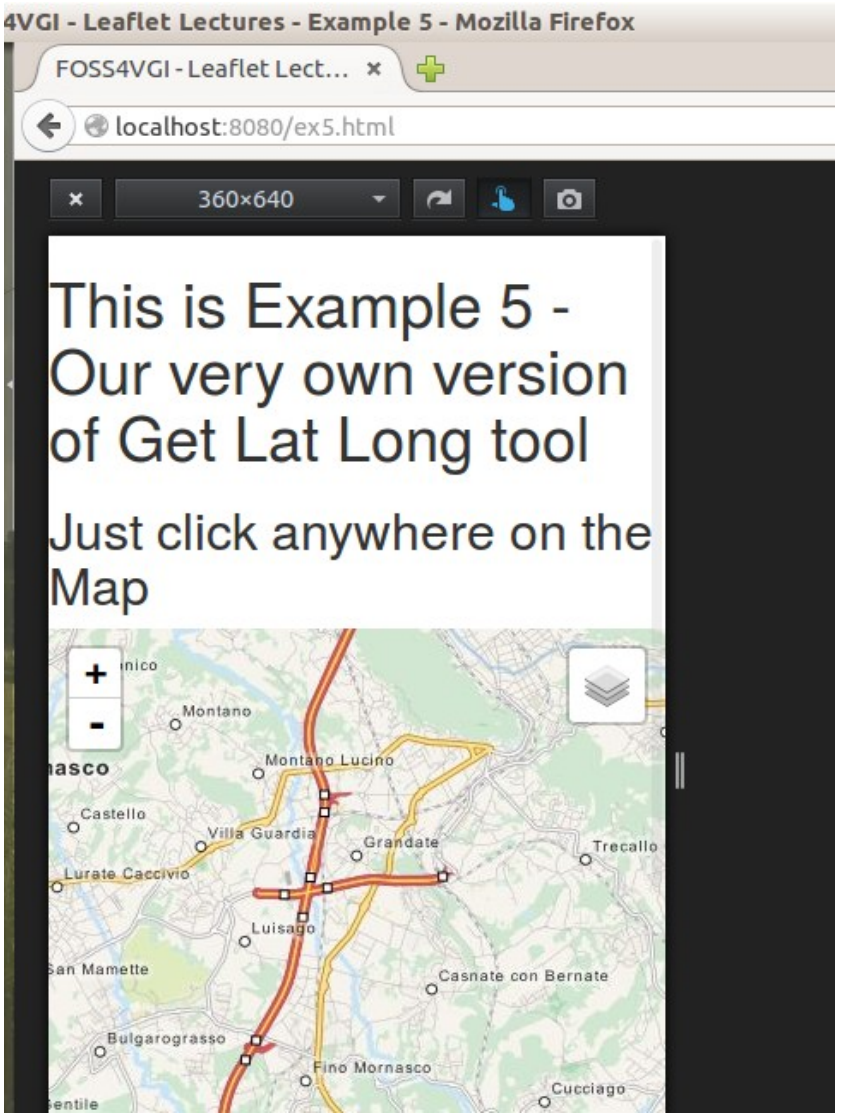
If you are using Google Chrome

- Open up ex5.html in your Google Chrome
- Go to the options menu
- Then go to More Tools
- Then go to Developer Tools
- Click on the little smartphone icon in the left hand corner.


localhost:8080/ex5.html
Device
Apple iPhone 6 Plus
414 × 736
Fit
Network
Good 3G (1 Mbps 40ms RTT)
UA Mozilla/5.0 (iPhone; CPU
This is Example 5 - Our very own version of Get Lat Long tool
Just click anywhere on the Map
Como
Elements
Network
Sources
Timeline
Profiles
Audits
Console
<top frame>
Preserve log
Console
Search
Emulation
Rendering
The specific device
The type of internet connection

If you are using Firefox

- Go to options
- Go to Developer
- Go to 'Responsive Design View”



Starting with GeoJSON and Leaflet

GeoJSON and Leaflet are made for each other :-)

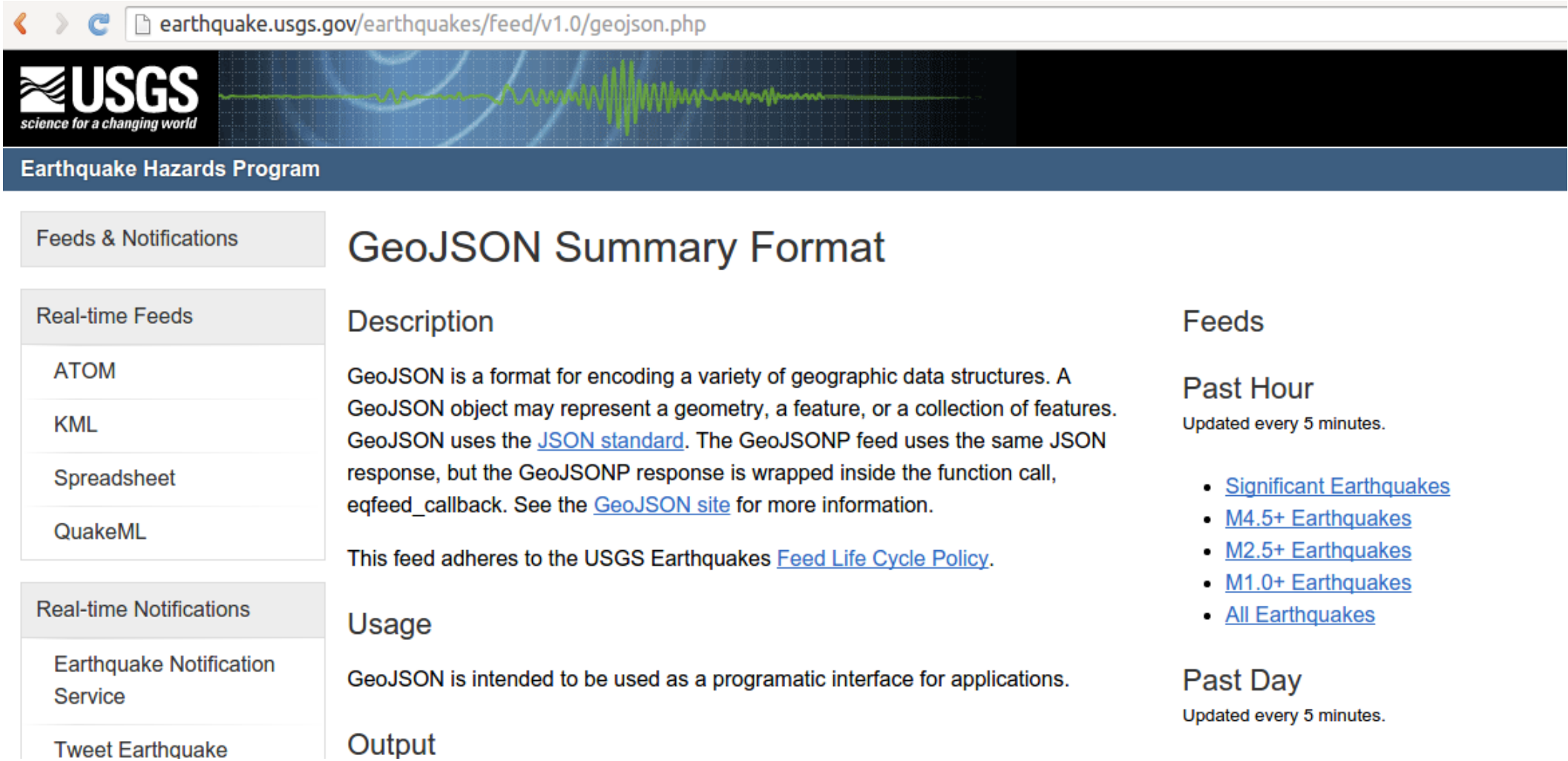
- **Leaflet is able to process, manipulate and display GeoJSON data**
- So it is in our best interests (because it is very efficient) to try to format our spatial data in GeoJSON format so that Leaflet can display it for us on a web map

Remember: Every GeoJSON file is a dataset in its own right

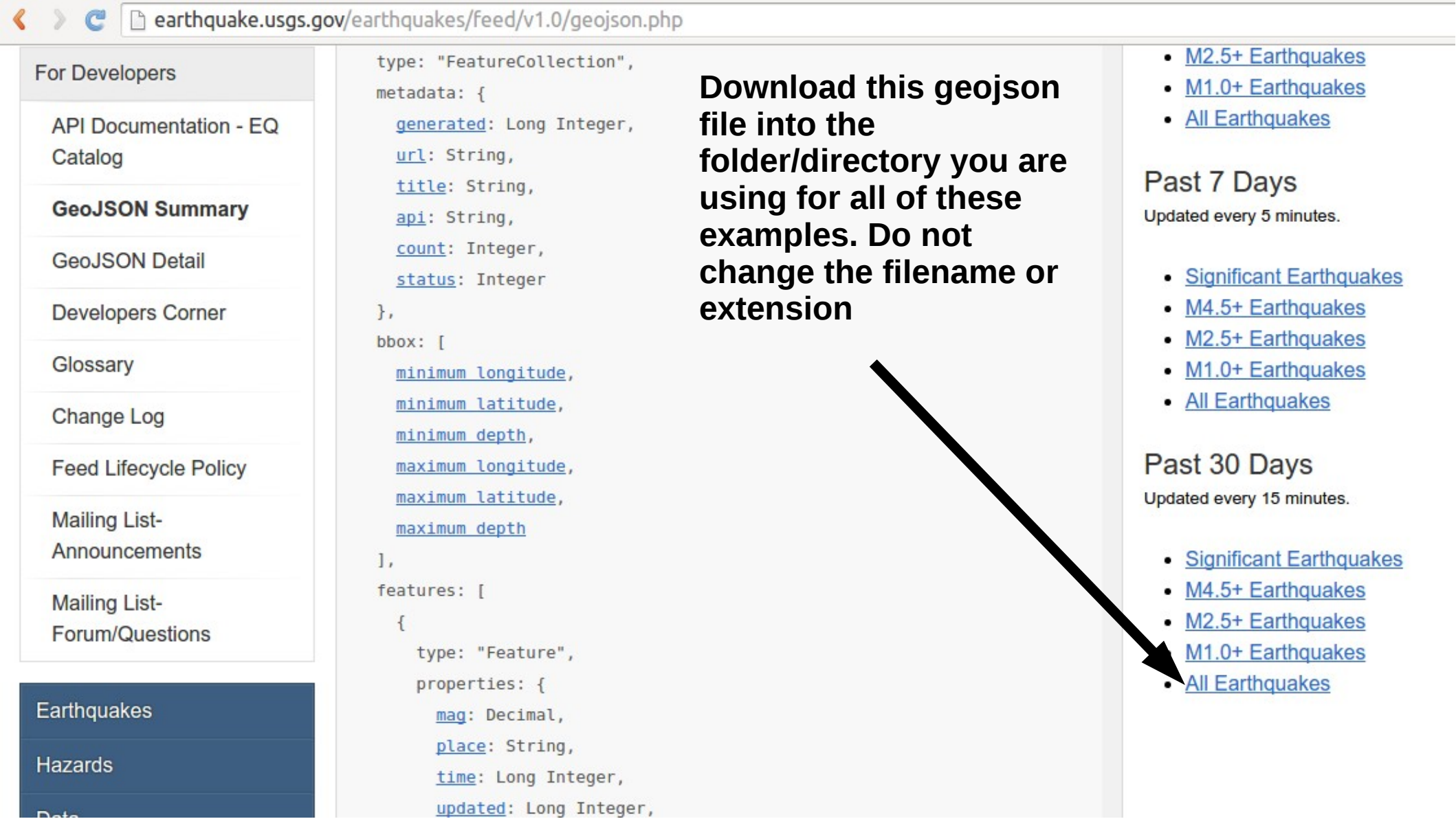
- This is an important consideration.
- We cannot just “throw” the GeoJSON file at Leaflet and expect Leaflet to display the file.
- **We will have to understand what the data contents of our GeoJSON file is in order to generate the best way to display our data**
- We shall start off with a GeoJSON file from the web and then proceed to OpenStreetMap data

geo1.html – US Geological Survey EarthQuake Data

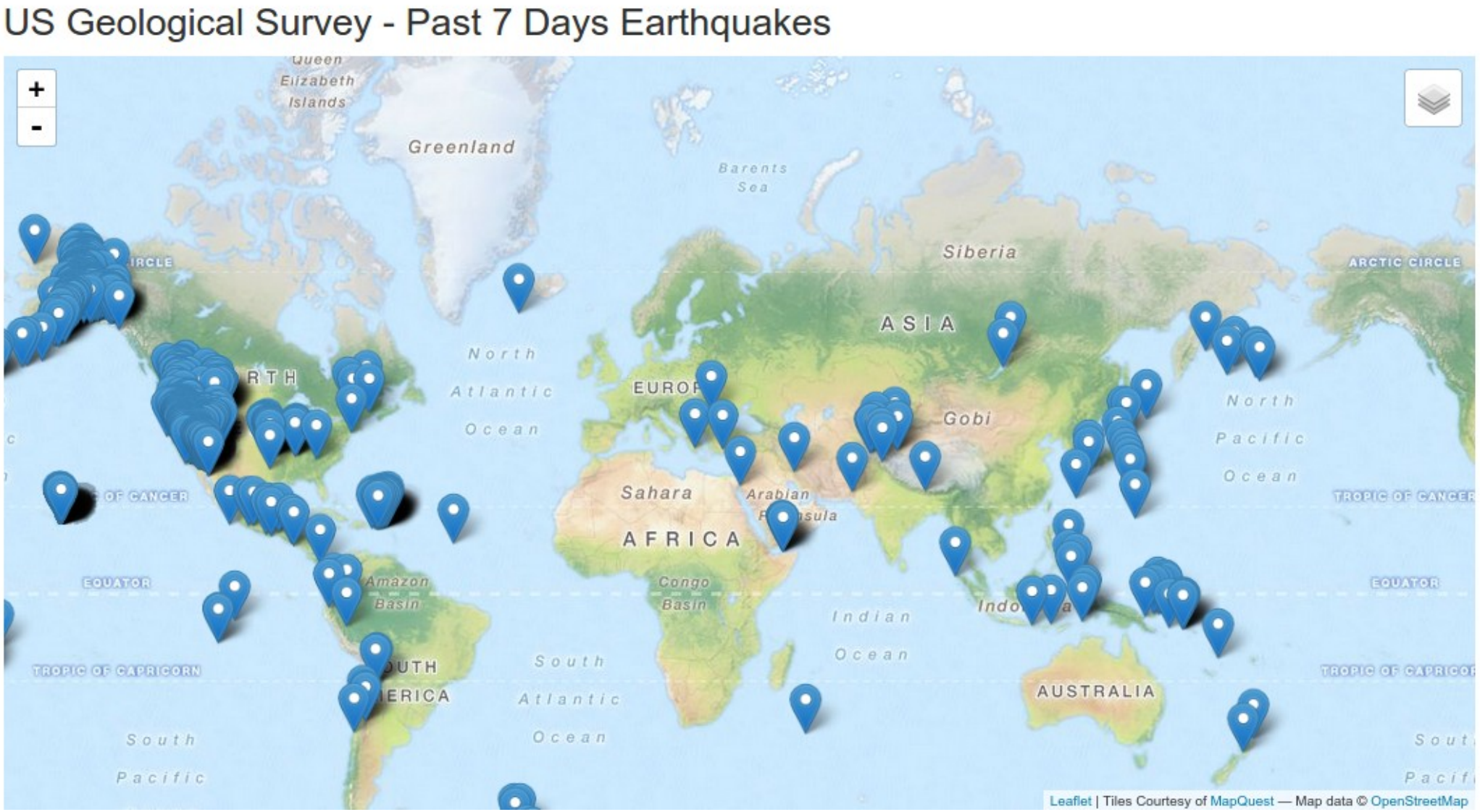
- http://earthquake.usgs.gov/earthquakes/feed/v1.0/geojson.php



geo1.html – please scroll to the bottom of this page and download the Past 7 Days 'All Earthquakes'



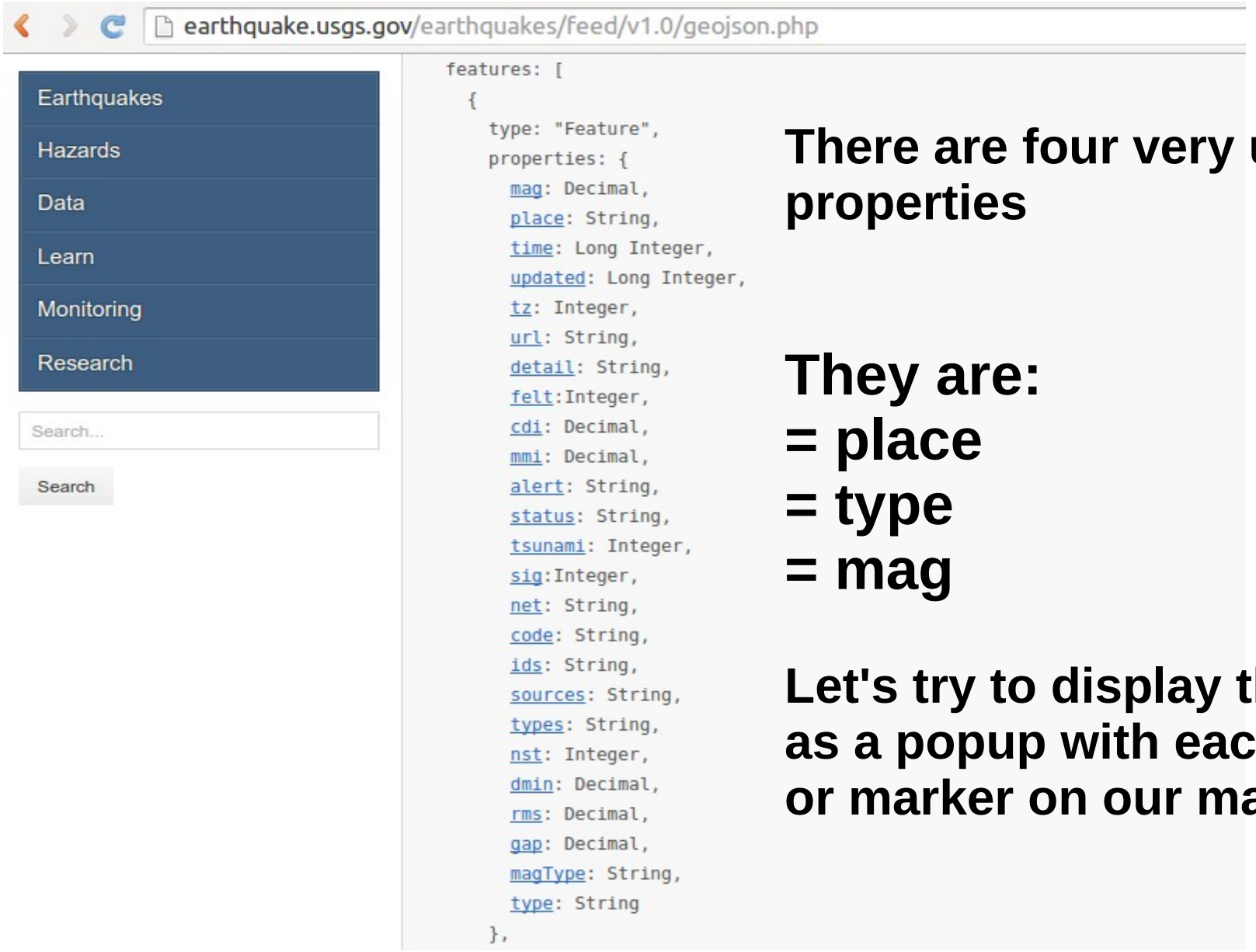
If you downloaded the data correctly – the link http://localhost:8080/geo1.html should look like this



Unfortunately there is one major problem with the map we have created

- If you look at the map – it is just a bunch of pins or markers
- YES – We state on the page that it is a USGS dataset
- We need to make those pins or markers 'clickable'
- We need to display some information about each pin.
- **We need to revisit the specification for the GeoJSON file on the USGS Website first.**

Here is the specification of the PROPERTIES of the GeoJSON data



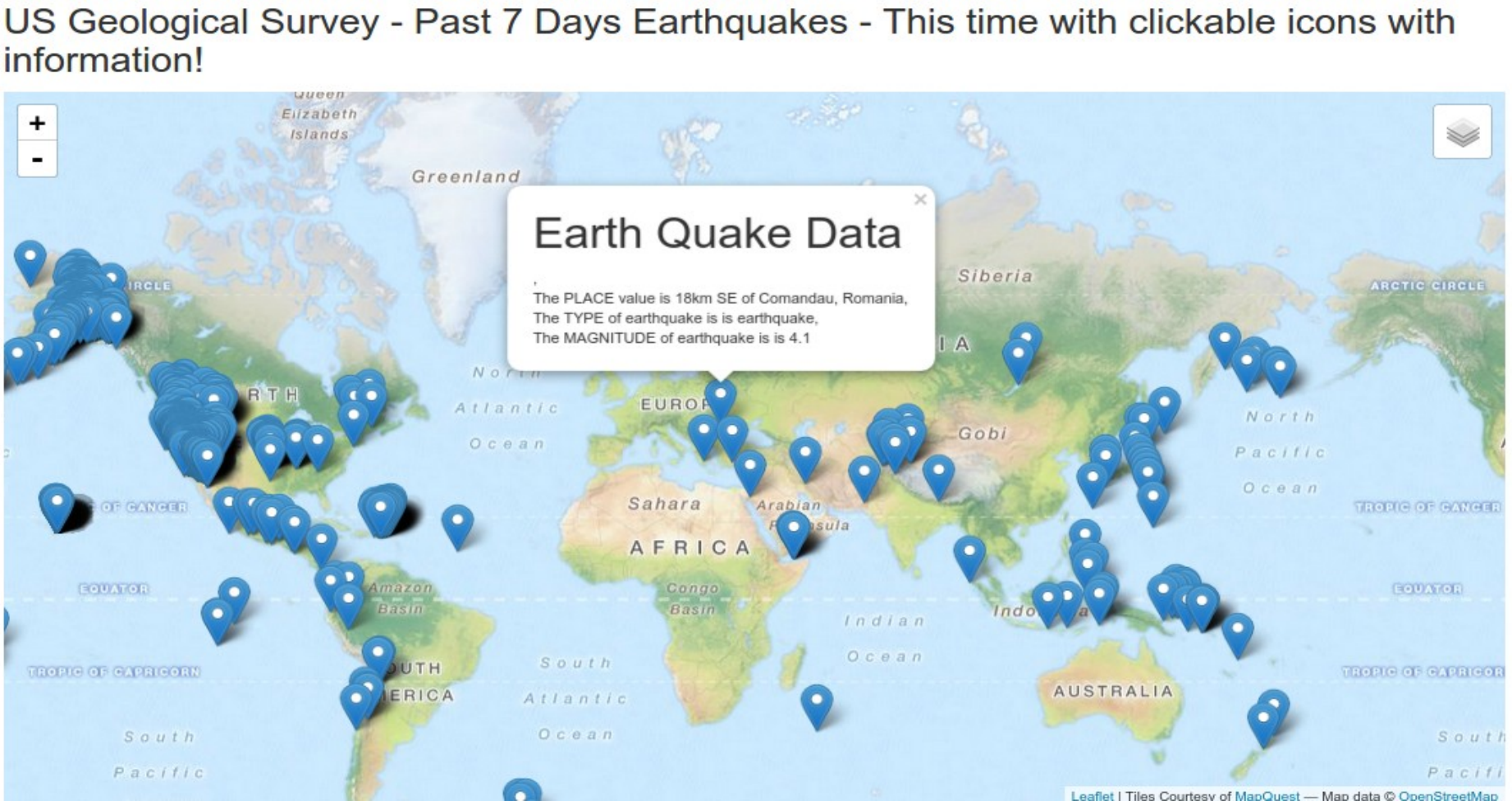
There are four very useful properties

They are:
= place
= type
= mag

Let's try to display these as a popup with each pin or marker on our map

Let's open up geo2.html in your text editor and look at it in the browser

- REMEMBER to view it at http://localhost:8080/geo2.html



It's very important for you to undertand the link between the Javascript in Leaflet and the PROPERTIES in the GeoJSON

- There is a VERY IMPORTANT new piece of code in the Javascript. We will REUSE this for all of our GeoJSON examples in the future!

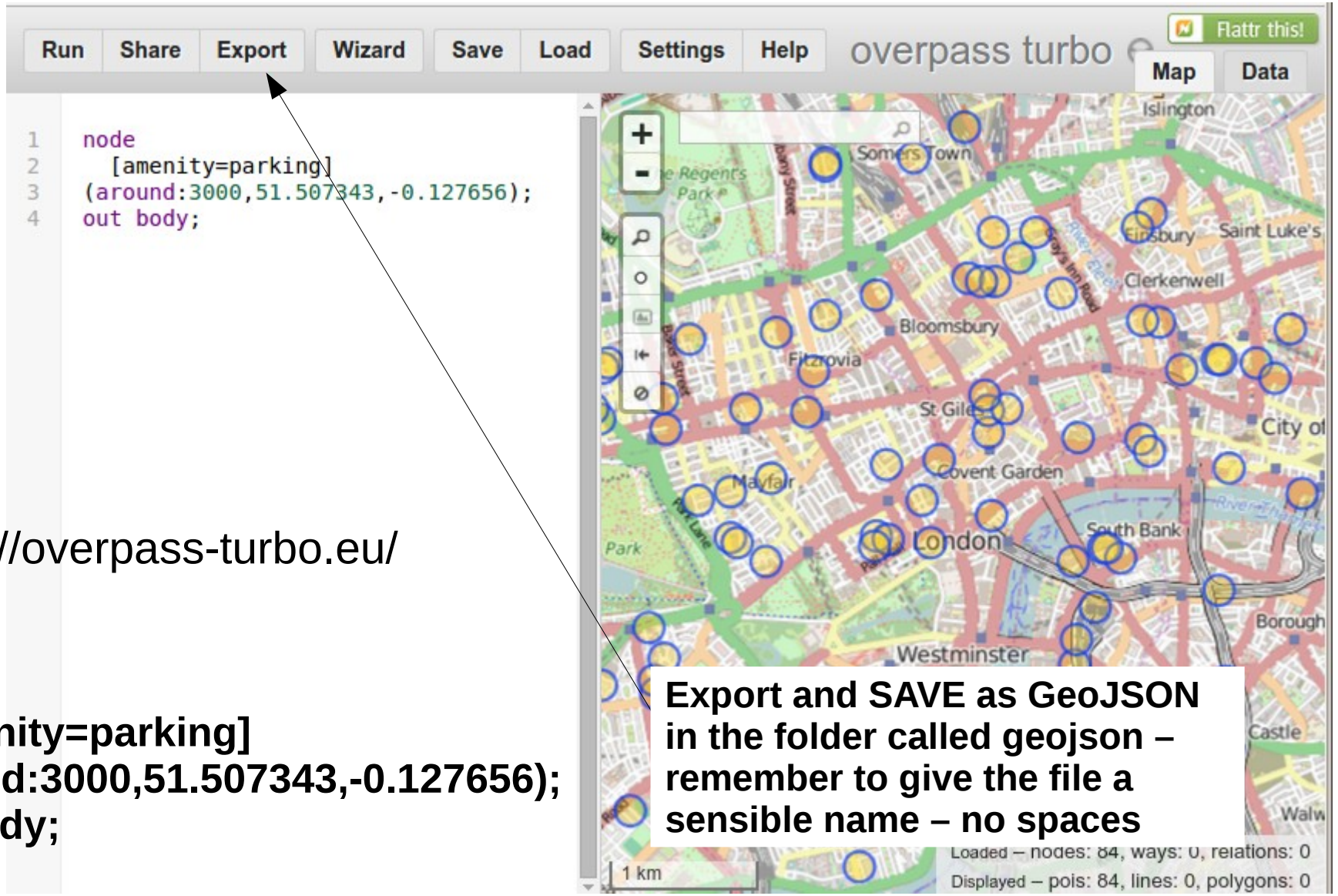
```
// This is where Leaflet will go if you click on any of the objects which are in the
// GeoJSON file
function action_To_Perform_When_Marker_Is_Clicked_On_The_Map(feature, layer) {

    // does this clicked feature have any properties in its GeoJSON?
    if (feature.properties)
    {
        // the text for the Popup
        var PopupText = [];

        PopupText.push("<h1>Earth Quake Data</h1>");

        if (feature.properties.place) // if this is a property called place
            PopupText.push("<br/>The PLACE value is " + feature.properties.place)

        if (feature.properties.type) // if this is a property called type
            PopupText.push("<br/>The TYPE of earthquake is is " + feature.properties.type)

        if (feature.properties.mag) // if this is a property called type
            PopupText.push("<br/>The MAGNITUDE of earthquake is is " + feature.properties.mag)

        // Join all the text to make the Popup
        layer.bindPopup("<p>" + PopupText.join() + "</p>");
    } // end if
} // end of function
```

Look at the PROPERTIES here
- these appear EXACTLY as they are typed in the GeoJSON file

= place
= type
= mag

Let's go back to get some OSM GeoJSON data from OVERPASS

Suppose we are feeling crazy and decide to drive our car into central London



http://overpass-turbo.eu/

node
[amenity=parking]
(around:3000,51.507343,-0.127656);
out body;

When downloaded and in correct folder – let's open it in a text editor


```
},
{
  "type": "Feature",
  "id": "node/368044118",
  "properties": {
    "@id": "node/368044118",
    "amenity": "parking",
    "capacity": "233",
    "capacity:disabled": "5",
    "covered": "yes",
    "fee": "yes",
    "name": "Baynard House",
    "operator": "Corporation of London",
    "website":
    "http://www.cityoflondon.gov.uk/services/transport-and-streets/parking/where-to-park/car-parks/Pages/Baynar
    d-House-car-park.aspx"
  },
  "geometry": {
    "type": "Point",
    "coordinates": [
      -0.1000953,
      51.5118468
    ]
  }
},
{
```


Look CAREFULLY at the properties of each Feature/Object/Node
Notice – each node has a different number of tags

Let's choose three: name operator capacity

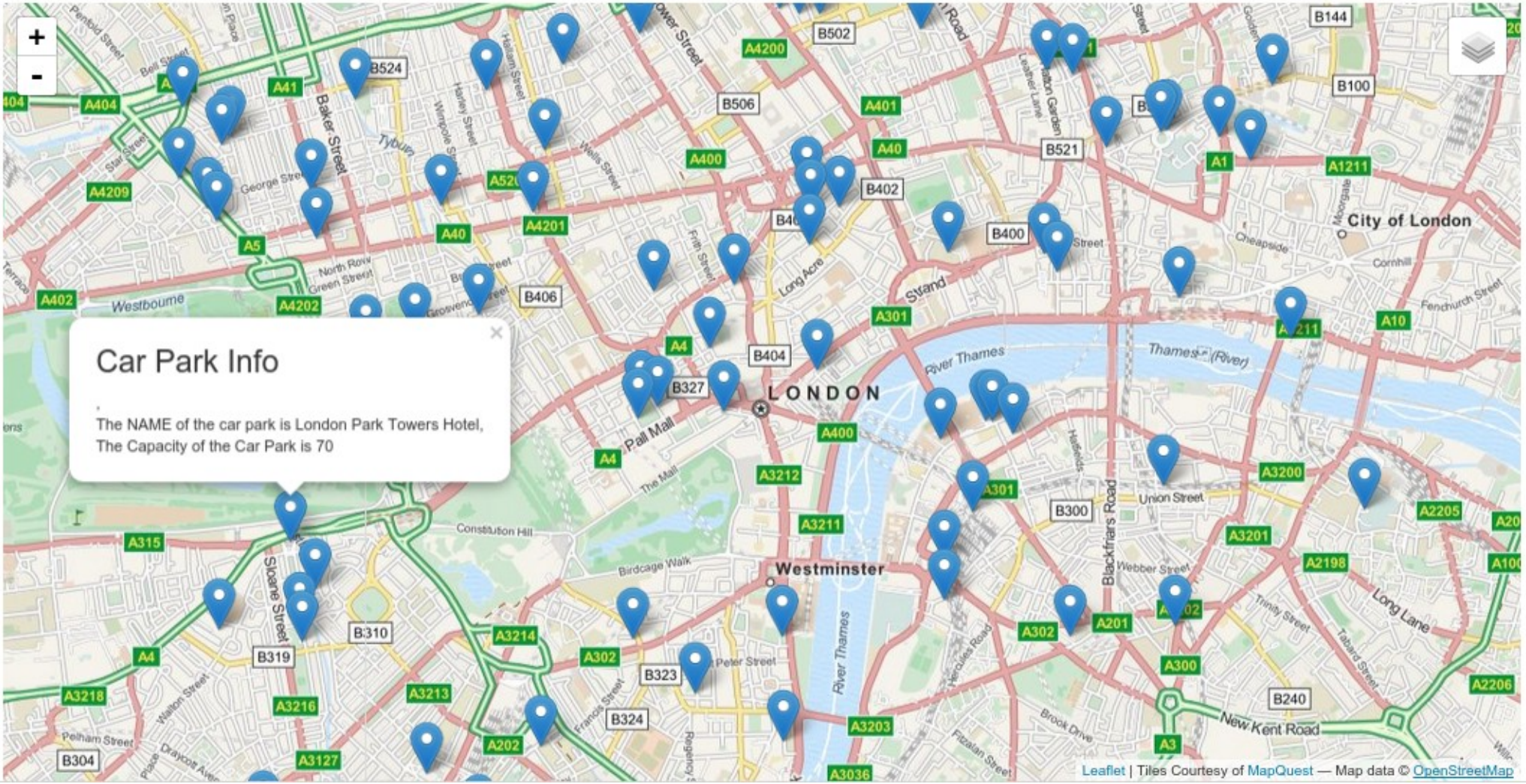
Check Map Features if you need more information

Open the file geo3.html in your text editor AND your browser

- You'll need to check out one very important detail
- You will need to change the filename of the geojson file to the name you used
- You will then save the file and refresh the page in your browser
- Otherwise ..... you won't see any car parks!!!

This is what http://localhost:8080/geo3.html should look like!

PARKING PLACES in Central London, UK. GeoJSON data from the Overpass Turbo service



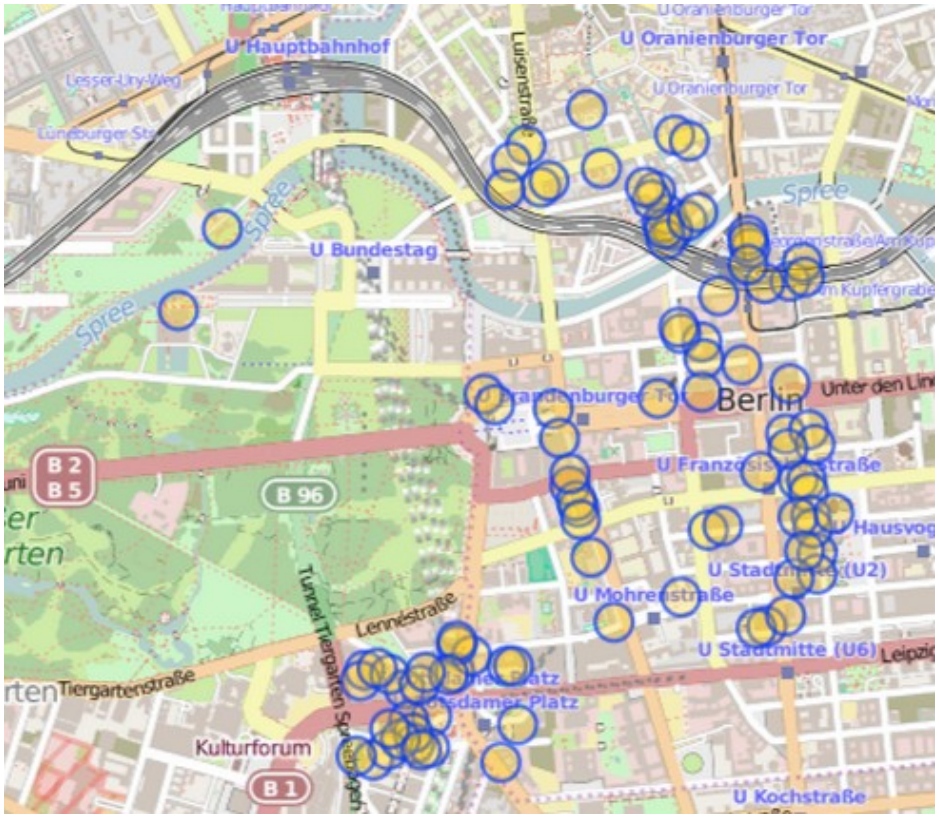
Look VERY carefully at the action_To_Perform_When_Marker_Is_Clicked_On_The_Map function

```
// This is where Leaflet will go if you click on any of the objects which are in the
// GeoJSON file
function action_To_Perform_When_Marker_Is_Clicked_On_The_Map(feature, layer) {

    // does this clicked feature have any properties in its GeoJSON?
    if (feature.properties)
    {
        // the text for the Popup
        var PopupText = [];

        PopupText.push("<h3>Car Park Info</h3>");

        if (feature.properties.name) // if this is a property called name
            PopupText.push("<br/>The NAME of the car park is " + feature.properties.name)

        if (feature.properties.type) // if this is a property called operator
            PopupText.push("<br/>The OPERATOR of the car park is " + feature.properties.operator)

        if (feature.properties.capacity) // if this is a property called capacity
            PopupText.push("<br/>The Capacity of the Car Park is " + feature.properties.capacity)

        // Join all the text to make the Popup
        layer.bindPopup("<p>" + PopupText.join() + "</p>");
    } // end if
} // end of function
```

NOTICE – How we had to change veyr little – we just change the text a little and we put in our own properties which we picked out a few minutes ago

GeoJSON Exercise on your own!

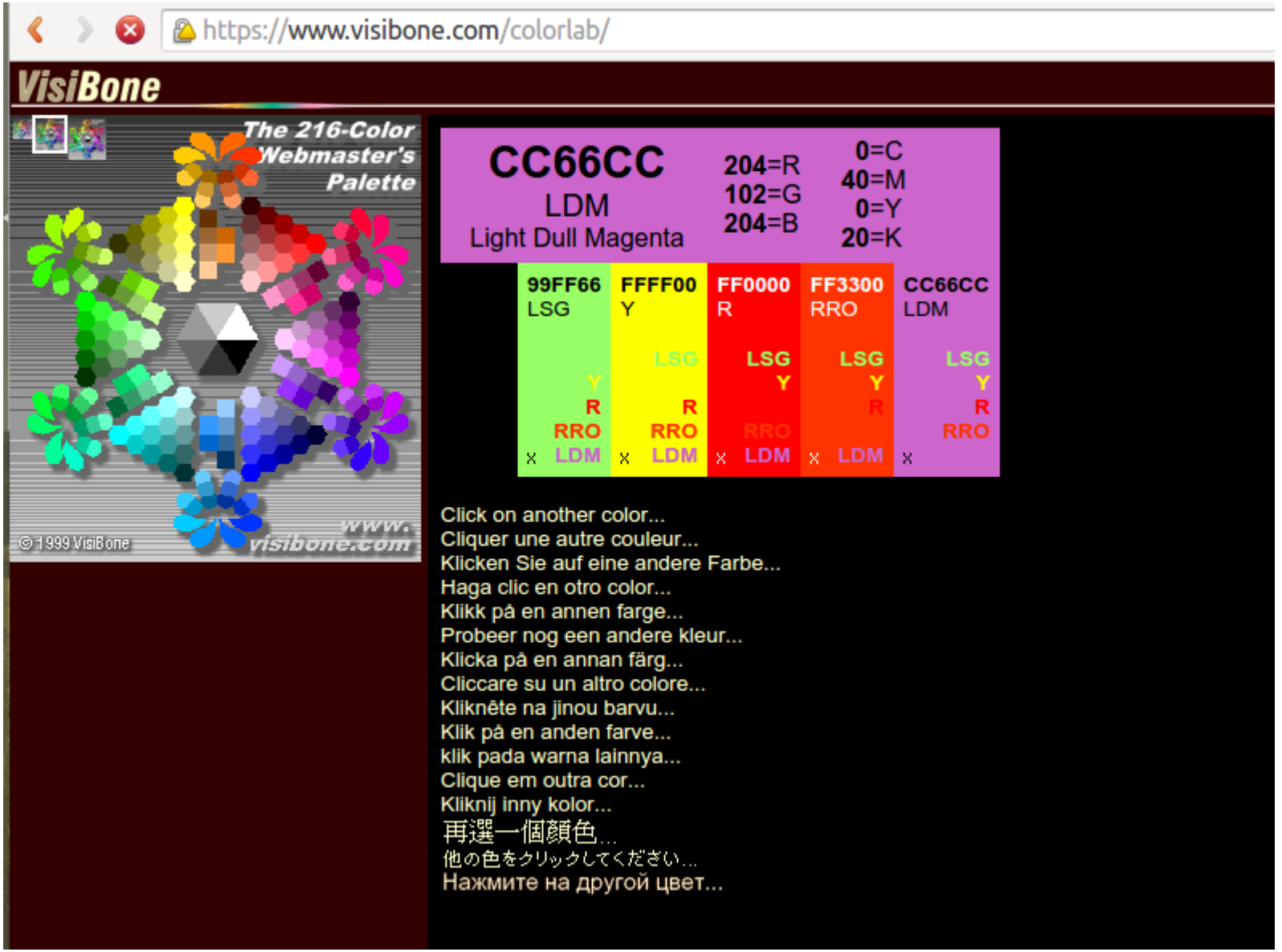
Exercise: Map of places to eat in Berlin, Germany



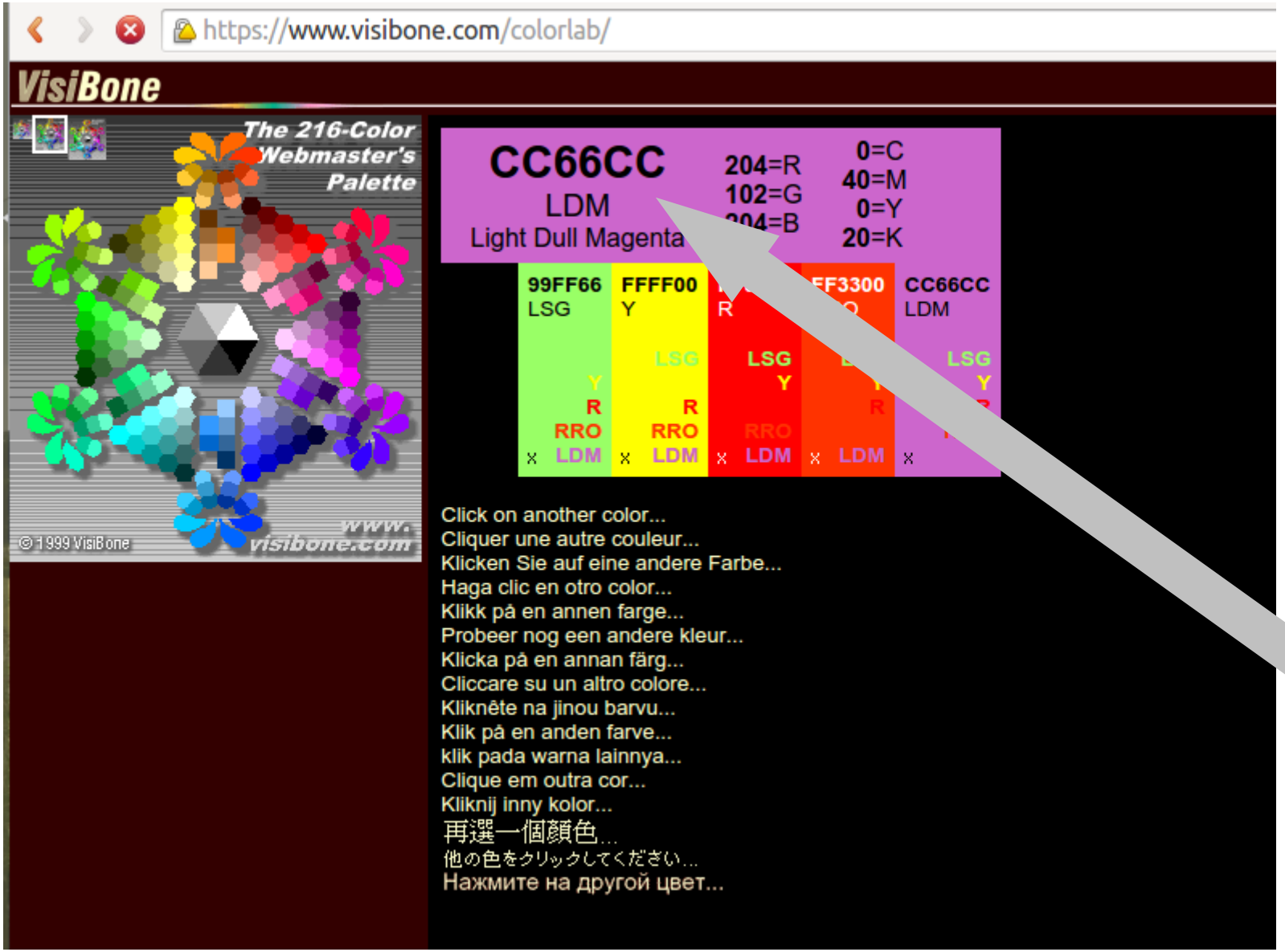
- Using the Overpass API generate GeoJSON representing all `amenity=restaurant` around 1000m radius of the Brandenburg Gate in Berlin
- Use `name, cuisine, opening_hours, wheelchair` as properties
- Center and zoom the map appropriately
- You can use geo3.html as your starting point

Colors for Polygons and other features in Javascript

Visibone Color Lab is very useful



It is very good practice to always specify colors using their HEX value – all browsers and Javascript understand HEX

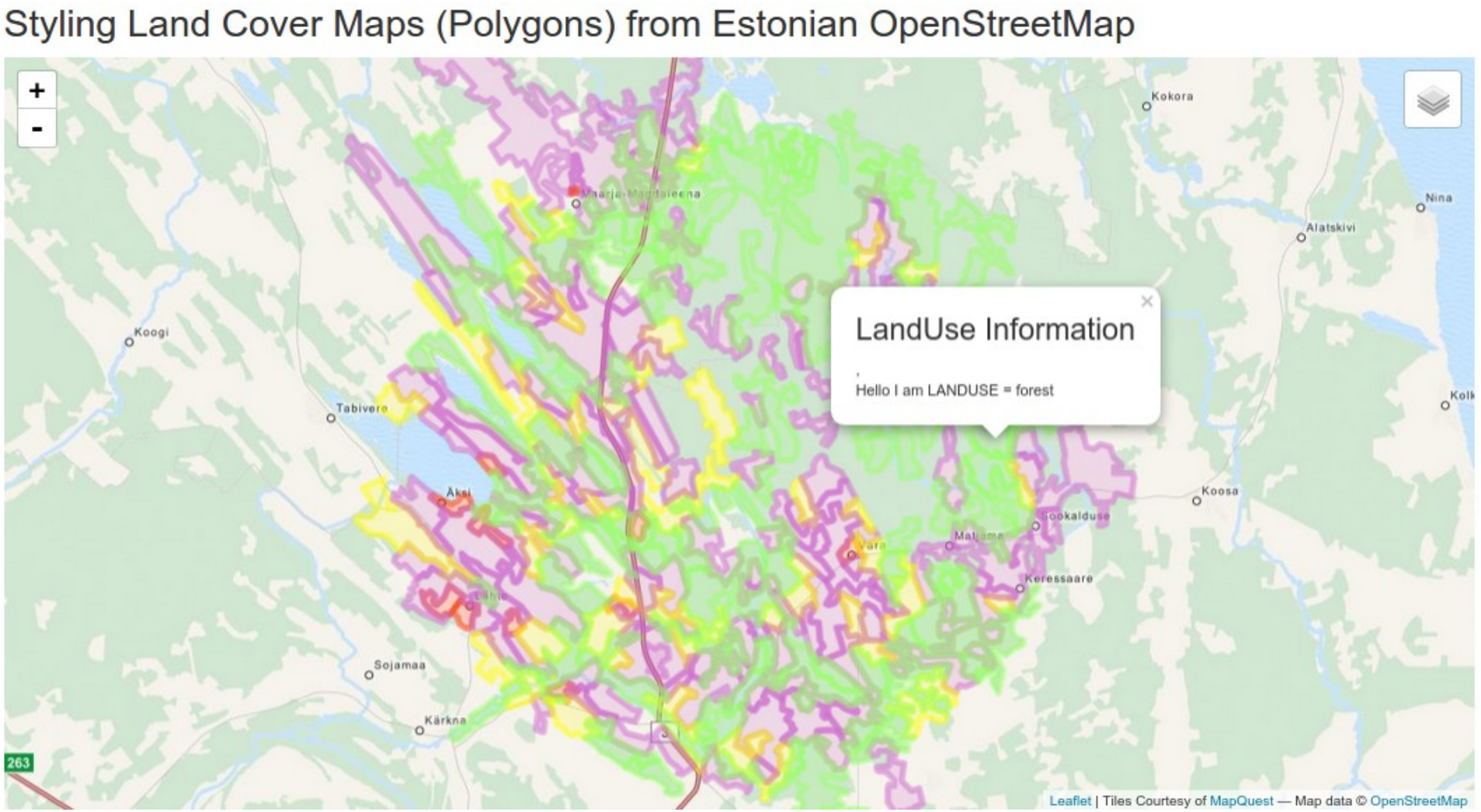


#CC66CC

Colouring/Styling Landuse Polygons in Estonia

- Open up geo4.html in your text editor and in your browser at http://localhost:8080/geo4.html
- This is an example where we have POLYGONS and not points.
- To make a map a bit more interesting we have decided to give the polygons a different colour depending on the value of the tag
- We are looking at LANDUSE tags only

The geo4.html web map should look something like this in your browser

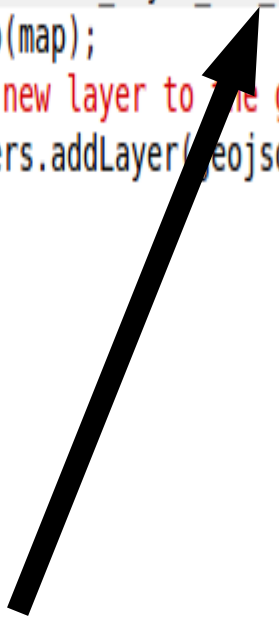


There are a few VERY important changes to our Javascript

- 1 – We tell LEAFLET that we want to put a specific STYLE on the polygons
- 2 – We must have code to tell LEAFLET what the colours in the specific STYLE actually are
- 3 – We must also adapt our code for when someone clicks on one of the polygons – in this case we just show the landcover tag

We tell LEAFLET that we want to put a specific STYLE on the polygons

```
$.getJSON("./geojson/estoniaLanduse.geojson",
    function(data) {
        // tell Leaflet to go to the function action_To_Perform_When_Marker_Is_Clicked_On_The_Map
        // when someone clicks on one of the objects in the GeoJSON layer
        var geojson = L.geoJson(data,{onEachFeature: action_To_Perform_When_Marker_Is_Clicked_On_The_Map,
            style: specific_style_for_ways});
        geojson.addTo(map);
        <!-- add our new layer to the group of top layers -->
        myGeoJSONLayers.addLayer(geojson);
    }
);
```



We must have code to tell LEAFLET what the colours in the specific STYLE actually are

```
// This is a VERY important function. This is where we say which colors will be
// displayed for specific values of the landuse tag
function specific_style_for_ways(feature) {
            switch (feature.properties.landuse) {
                case 'meadow': return {color: "#FFFF00"};
                case 'forest':   return {color: "#99FF66"};
                case 'farm':   return {color: "#CC66CC"};
                default: return {color: "#FF3300"}; // this is when we don't know the property value
                // or it is different to the ones above
            }
        }
```

We must also adapt our code for when someone clicks on one of the polygons – in this case we just show the landcover tag

```
// This is where Leaflet will go if you click on any of the objects which are in the
// GeoJSON file
function action_To_Perform_When_Marker_Is_Clicked_On_The_Map(feature, layer) {

    // does this clicked feature have any properties in its GeoJSON?
    if (feature.properties)
    {
        // the text for the Popup
        var PopupText = [];

        PopupText.push("<h3>LandUse Information</h3>");

        if (feature.properties.landuse) // if this is a property called landuse
            PopupText.push("<br/>Hello I am LANDUSE = " + feature.properties.landuse)

        // Join all the text to make the Popup
        layer.bindPopup("<p>" + PopupText.join() + "</p>");
    } // end if
} // end of function
```

Just a very simple popup for this example

Optional Exercise on Polygons

Visualising Amenity polygons in Como

- Go to the GitHUB page for today's lecture
- Copy the final piece of OVERPASS code into overpass online
- This is for amenity objects in COMO
- Save the output as GeoJSON into your geoJSON folder

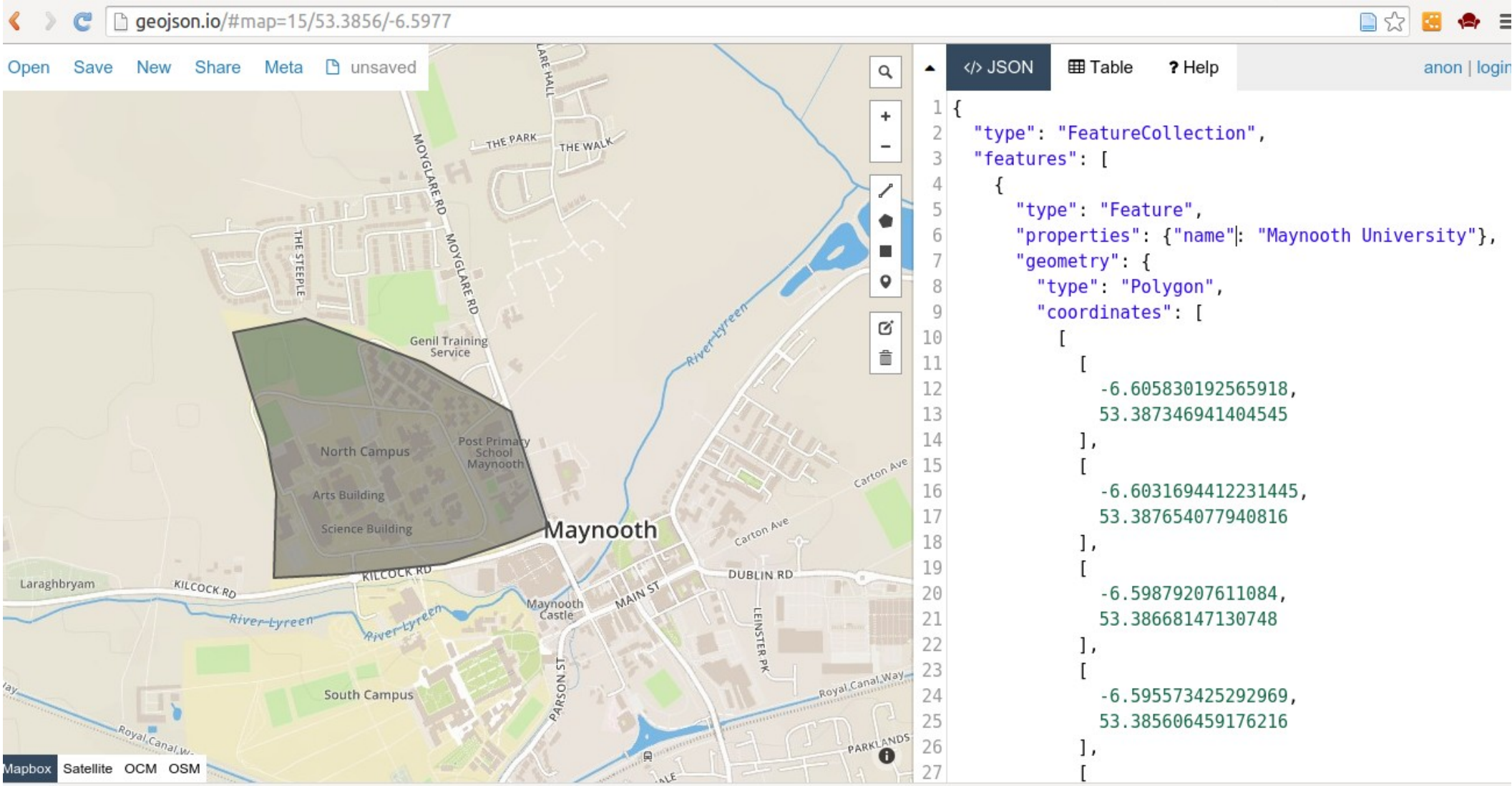
Exercise – Colouring amenity objects in Como

- Copy the geo4.html file and rename as geo5.html
- Make the necessary changes to center, zoom and name of geoJSON file
- Color all Amenity = School (#009246)
- Color all Amenity = Parking (#CE2B37)
- Color everything else (White #FFFFFF)
- In the POPUP information just give the AMENITY type as text

So now you can develop your own web-based maps using Open Source Software AND Open Data

You can easily make your own GeoJSON

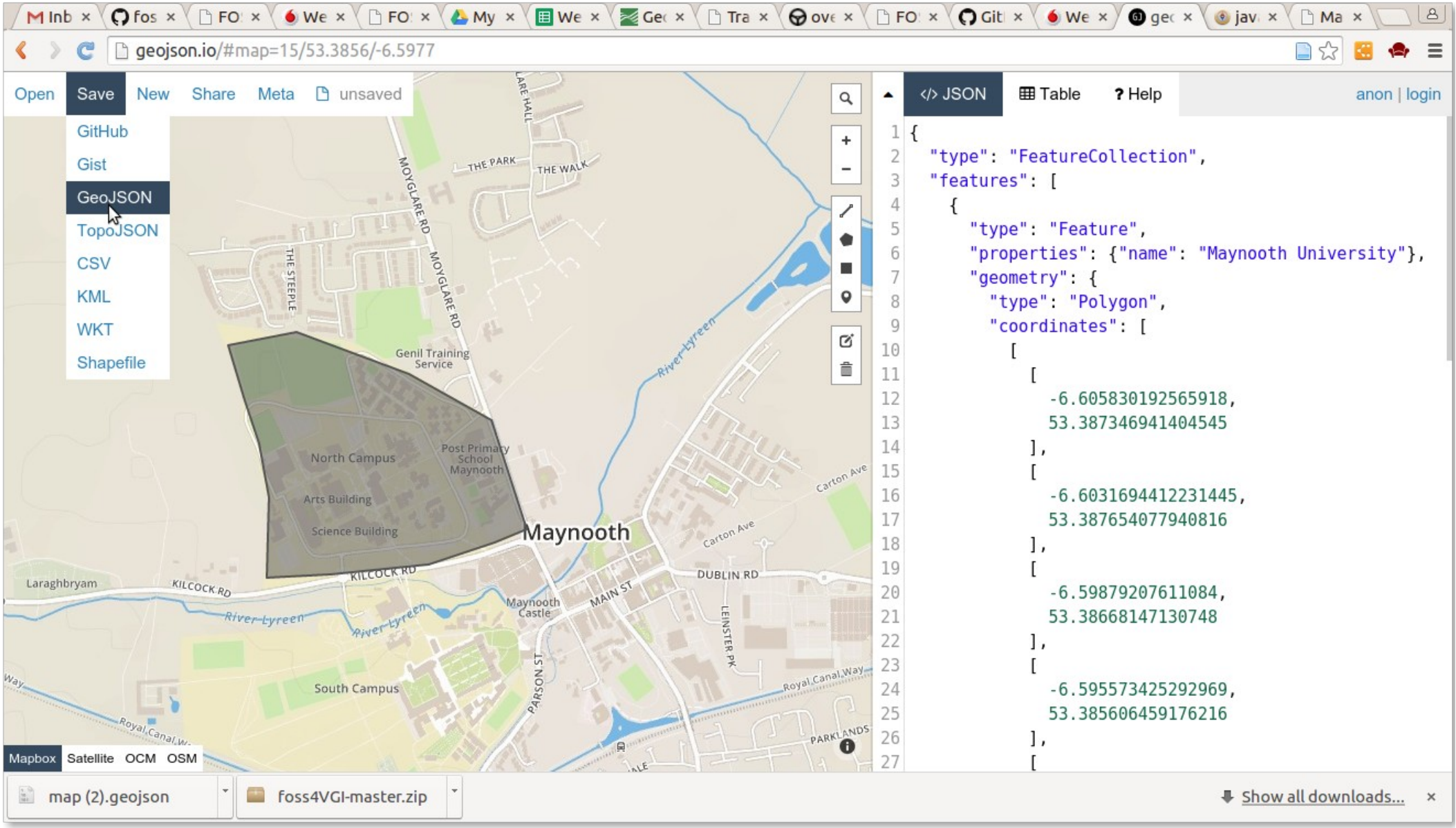
Go to GeoJSON.io



You can make really nice GeoJSON datasets here

- Remember PROPERTIES – the name of the property and the value MUST always have double quotes
- For example “population”: “200,000”
- You can have a mixture of polylines, polygons and points

You can easily SAVE the data you create as GeoJSON




```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "properties": {"name": "Maynooth University"},
      "geometry": {
        "type": "Polygon",
        "coordinates": [
          [
            [
              -6.605830192565918,
              53.387346941404545
            ],
            [
              -6.6031694412231445,
              53.387654077940816
            ],
            [
              -6.59879207611084,
              53.38668147130748
            ],
            [
              -6.595573425292969,
              53.385606459176216
            ],
            [
```

The final exercise.....

Exercise – on your own

- Create a simple GeoJSON dataset on GeoJSON.io
- Give each feature one or two PROPERTIES (remember the double quotes)
- Use the code you have written for LEAFLET today to make a web-based map of the data you have created.
- In particular geo2.html, geo3.html and geo4.html will be of help! (if you are adventurous)

I'm happy to help with questions at petermooney78@gmail.com