



GIS Data Preparation and Conversion for the Web

Ionuț Iosifescu

Data Preparation Workflow

Data Collection

- Data Sources
- GIS Data Formats

Data Check

- Projection
- Scale
- Generalization
- Attributes

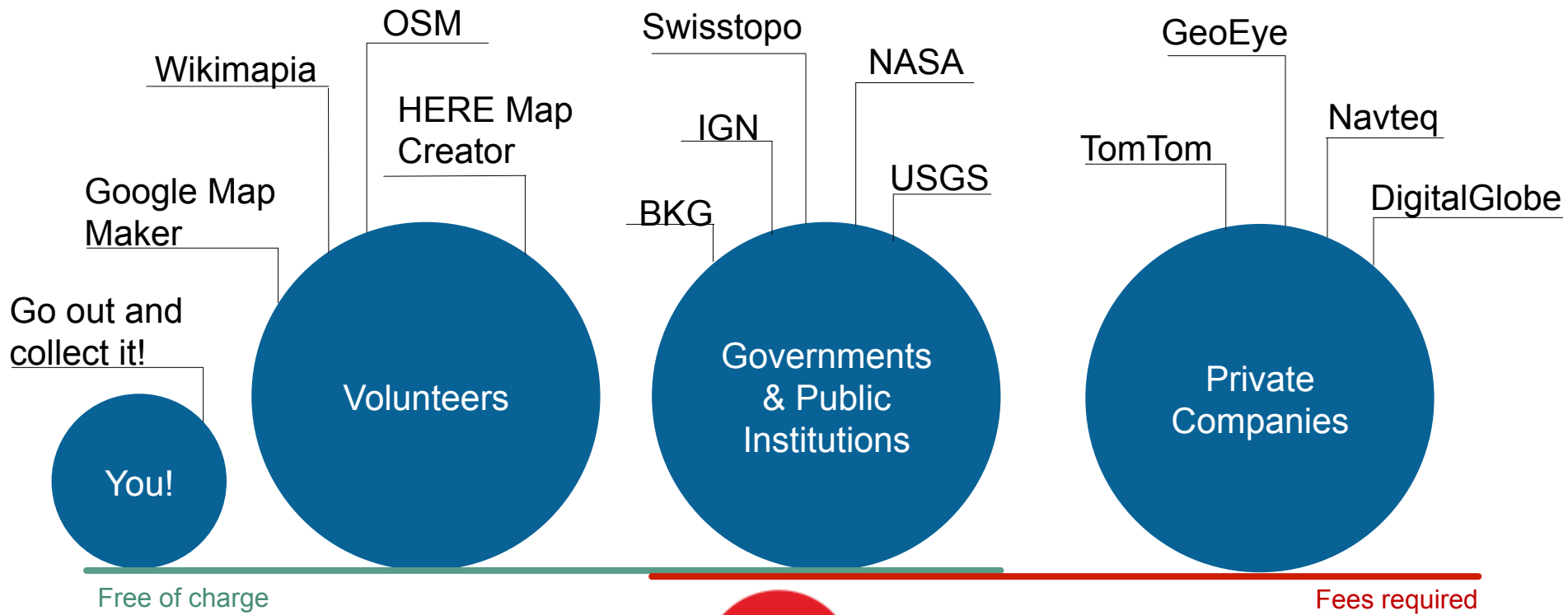
Convert Data

- Web Formats
- Web friendly formats

Visualize Data

- From Data to Maps

Data Sources



Traditional GIS Data Formats

Vector Data

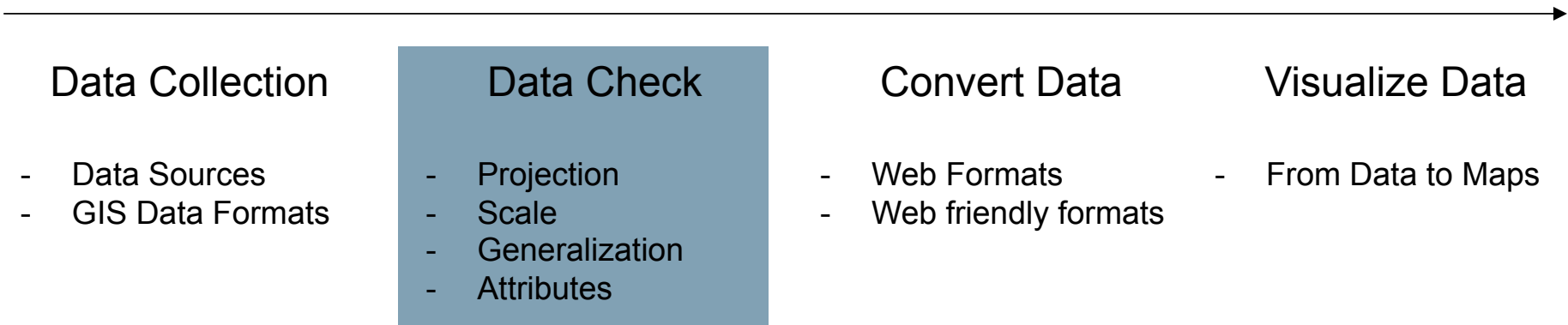
- Shapefiles (ESRI) - .shp, dbf, .prj, .shx
- Autocad files - .cad, .dxf
- MapInfo TAB format - .tab
- GPS format - .gpx

Raster Data

- GeoTIFF - .tif
- DEM - .dem
- ASCII - .asc

Good for GIS Desktop applications, but not very appropriate for the Web!

Data Preparation Workflow



Data Check

- Is it the right coordinate system?

Helps in choosing the right
coordinate system

<http://projectionwizard.org/>



Mercator



Plate Carrée

Data Check

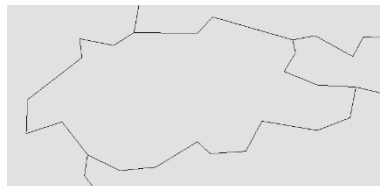
- How many zoom levels do you need? Is your data in the right scale?



Zoom level 0



Zoom level 9



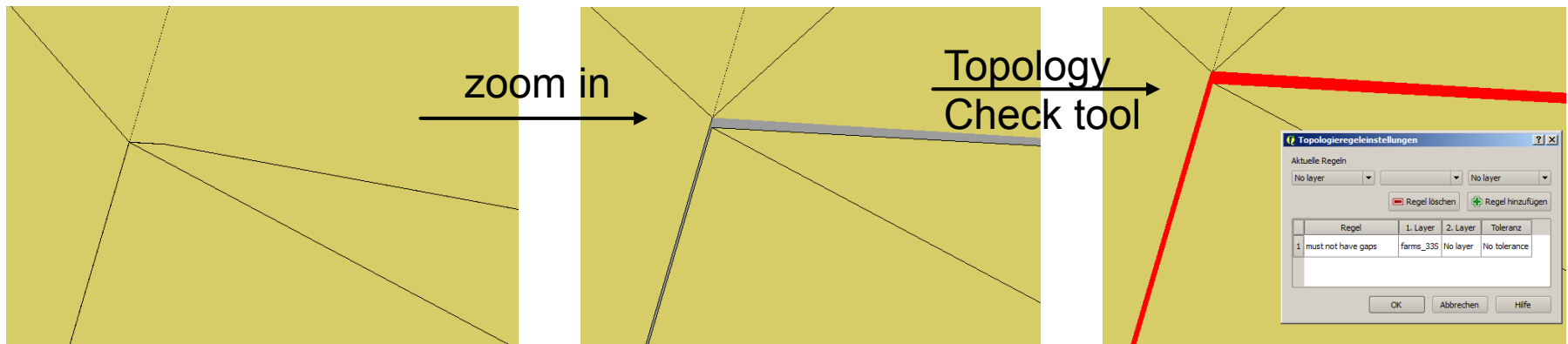
Zoom Level	Map Scale
0	1:591,657,550.500000
1	1:295,828,775.300000
2	1:147,914,387.600000
3	1:73,957,193.820000
4	1:36,978,596.910000
5	1:18,489,298.450000
6	1:9,244,649.227000
7	1:4,622,324.614000
8	1:2,311,162.307000
9	1:1,155,581.153000
10	1:577,790.576700
11	1:288,895.288400
12	1:144,447.644200
13	1:72,223.822090
14	1:36,111.911040
15	1:18,055.955520
16	1:9,027.977761
17	1:4,513.988880
18	1:2,256.994440
19	1:1,128.497220

Zoom levels
for Google
Maps

Reference: <http://blogs.esri.com/esri/arcgis/2009/03/19/how-can-you-tell-what-map-scales-are-shown-for-online-maps/>

Data Check

- Is the topology correct?



- Is the data complete? Should you add some attributes? Should you delete some unnecessary attributes?

	TYPE	NAME	ONEWAY	LANES
12	trunk	NULL	NULL	0
13	unclassified	NULL	NULL	0
14	residential	Kruin Crescent	NULL	0
15	residential	Waterkant Street	NULL	0
16	residential	Kruin Crescent	NULL	0

Data Check


- Are the attributes plausible?

Attributtabelle - farms_335 :: Objekte gesamt: 2008, gefiltert: 2008, gewählt: 0

	AG_CAD_ID	SG_CODE	FARM_NO	PORTION	MAJ_REGION	X	Y
0	6242406	C073000000000...	191	00000	SWELLENDAM RD	20.3795899200	-34.0060160700
1	6242407	C073000000000...	191	00000	SWELLENDAM RD	20.4063922800	-34.0227875700
2	6242408	C073000000000...	193	00000	SWELLENDAM RD	20.4034838400	-34.0418817000
3	6242409	C073000000000...	194	00000	SWELLENDAM RD	20.3763189600	-34.0359407100
4	6242539	C073000000000...	611	00000	SWELLENDAM RD	20.1572960400	-34.2226762200
5	6242541	C073000000000...	34	00000	SWELLENDAM RD	20.7856436400	-33.8297004000
6	6242542	C073000000000...	35	00000	SWELLENDAM RD	20.7366094800	-33.7903722000

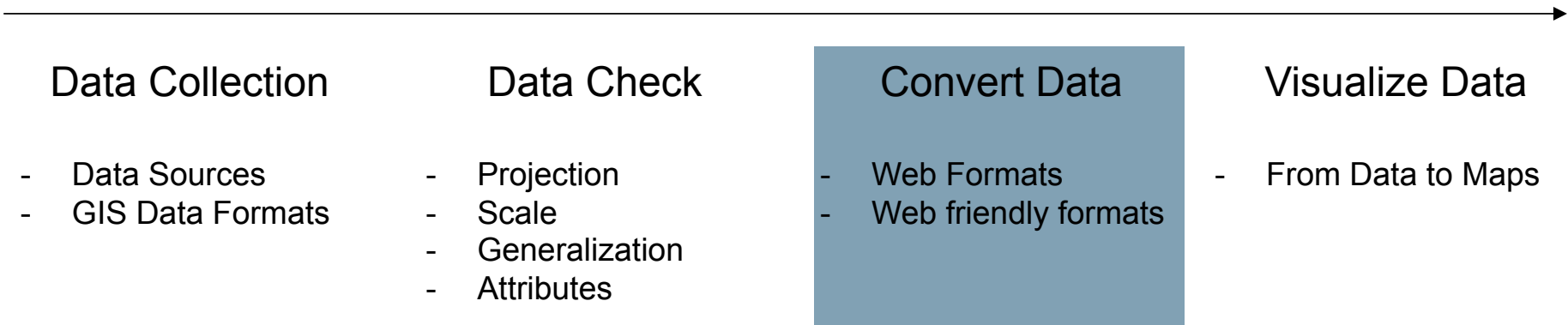
- Do you need all the data or do you need a selection of it?

`SELECT * WHERE "TYPE" = 'residential'`



18	residential	Van Zyl street	NULL	0
19	residential	Akasia Avenue	NULL	0
20	residential	Olien Avenue	NULL	0
21	residential	Akasia Avenue	NULL	0
22	residential	Karee Avenue	NULL	0
23	residential	NULL	NULL	0
24	residential	NULL	NULL	0
25	residential	Hoop Street	NULL	0
26	unclassified	Coetzee Street	NULL	0
27	residential	Rigg	NULL	0

Data Preparation Workflow



Data Formats for Web

Vector Data

Raster Data

Web Formats that can be visualized directly in the browser

- **SVG**

- GIF
- PNG
- JPG

Web friendly formats that need to be read/parsed with Javascript

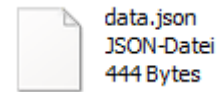
- **JSON / GeoJSON / TopoJSON**
- CSV
- **XML**
- GML
- KML

JSON – JavaScript Object Notation

A data exchange format

Based on Javascript, but language independent, can be understood by C, C++, C#, Java, Perl, Python...

```
{  
  "type": "city",  
  "collection": [  
    {  
      "properties": {  
        "population": 8615246,  
        "name": "London"  
      },  
      "id": "London"  
    },  
    {  
      "properties": {  
        "population": 2193031,  
        "name": "Paris"  
      },  
      "id": "Paris"  
    }  
  ]  
}
```



AJAX Request

```
▼ Object {type: "city", collection: Array[2]}  
  ▼ collection: Array[2]  
    ▼ 0: Object  
      id: "London"  
      ▼ properties: Object  
        name: "London"  
        population: 8615246  
        ► __proto__: Object  
    ► __proto__: Object  
    ▼ 1: Object  
      length: 2  
      ► __proto__: Array[0]  
type: "city"
```

GeoJSON – the Geo Version of JSON

Geometry object

```
{  
  "type": "Point",  
  "coordinates": [-0.117, 51.5]  
}
```

Feature object

```
{  
  "type": "Feature",  
  "geometry": {  
    "type": "Point",  
    "coordinates": [-0.117, 51.5]  
  },  
  "properties": {  
    "wikipedia": "London",  
    "city": "London"  
  },  
  "id": "London"  
}
```

FeatureCollection object

```
{  
  "type": "FeatureCollection",  
  "features": [{  
    "geometry": {  
      "type": "Point",  
      "coordinates": [-0.117, 51.5]  
    },  
    "type": "Feature",  
    "properties": {  
      "wikipedia": "London",  
      "city": "London"  
    },  
    "id": "London"  
  }, ...]  
}
```

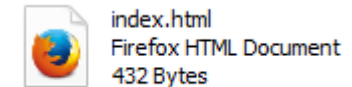
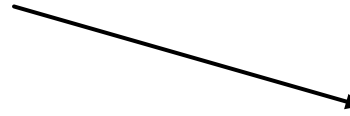
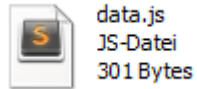
Must have this property

GeoJSON – how to read it?

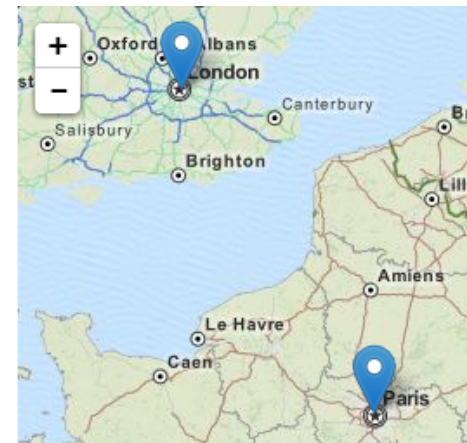
```

var cities = {
  "type": "FeatureCollection",
  "features": [{
    "geometry": {
      "type": "Point",
      "coordinates": [-0.117, 51.5]
    },
    "type": "Feature",
    "properties": {
      "wikipedia": "London",
      "city": "London"
    },
    "id": "London"
  }, {
    "geometry": {
      "type": "Point",
      "coordinates": [2.333, 48.867]
    },
    "type": "Feature",
    "properties": {
      "wikipedia": "Paris",
      "city": "Paris"
    },
    "id": "Paris"
  }
  ]
}

```



```
<script type="text/javascript" src="data.js"></script>
```



XML – eXtensible Markup Language



Another data exchange format

XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<type>city</type>
<collection>
  <properties>
    <population>8615246</population>
    <name>London</name>
  </properties>
  <id>London</id>
</collection>
<collection>
  <properties>
    <population>2193031</population>
    <name>Paris</name>
  </properties>
  <id>Paris</id>
</collection>
```

JSON

```
{
  "type": "city",
  "collection": [
    {
      "properties": {
        "population": 8615246,
        "name": "London"
      },
      "id": "London"
    },
    {
      "properties": {
        "population": 2193031,
        "name": "Paris"
      },
      "id": "Paris"
    }
  ]
}
```

GML – Geographic Markup Language

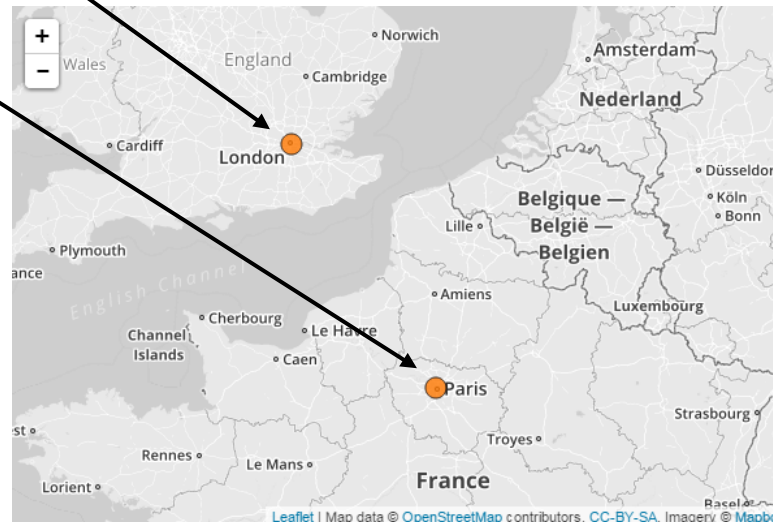
```
<gml:featureMember>
  <ms:polygon fid="2">
    <gml:boundedBy>
      <gml:Box srsName="EPSG:4326">
        <gml:coordinates>1.511919,47.088176 3.002191,47.882988</gml:coordinates>
      </gml:Box>
    </gml:boundedBy>
    <ms:msGeometry>
      <gml:Polygon srsName="EPSG:4326">
        <gml:outerBoundaryIs>
          <gml:LinearRing>
            <gml:coordinates>1.625463,47.357844 1.511919,47.741057 1.880938,47.882988
            2.420275,47.797830 2.789295,47.485582 3.002191,47.457196 2.874453,47.088176
            2.178993,47.343651 1.625463,47.357844 </gml:coordinates>
          </gml:LinearRing>
        </gml:outerBoundaryIs>
      </gml:Polygon>
    </ms:msGeometry>
    <ms:ogc_fid>2</ms:ogc_fid>
    <ms:name>My simple Polygon</ms:name>
    <ms:id>0</ms:id>
  </ms:polygon>
</gml:featureMember>
```


SVG – Scalable Vector Graphics

```

<svg width="625" height="417" viewBox="3373 167 625 417" style="transform:
translate3d(3373px, 167px, 0px);">
  <g>
    <path stroke="#000" fill="#ff7800" d="M3526,219A8,8,0,1,1,3525.9,219 z">
    </path>
  </g>
  <g>
    <path stroke="#000" fill="#ff7800" d="M3637,406A8,8,0,1,1,3636.9,406 z">
    </path>
  </g>
</svg>

```



Data Preparation Workflow

Data Collection

- Data Sources
- GIS Data Formats

Data Check

- Projection
- Scale
- Generalization
- Attributes

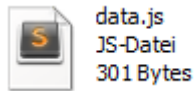
Convert Data

- Web Formats
- Web friendly formats

Visualize Data

- From Data to Maps

From our data to the map



data.js
JS-Datei
301 Bytes

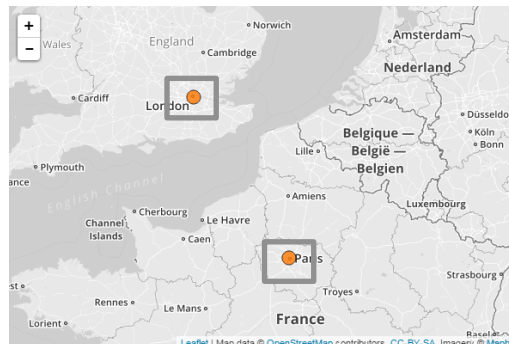
```
var cities = {
  "type": "FeatureCollection",
  "features": [{
    "geometry": {
      "type": "Point",
      "coordinates": [-0.117, 51.5]
    },
    "type": "Feature",
    "properties": {
      "wikipedia": "London",
      "city": "London"
    },
    "id": "London"
  }, {
    "geometry": {
      "type": "Point",
      "coordinates": [2.333, 48.867]
    },
    "type": "Feature",
    "properties": {
      "wikipedia": "Paris",
      "city": "Paris"
    },
    "id": "Paris"
  }
  ]
}
```



leaflet.js
JS-Datei
122 KB

is converted by Leaflet into SVG
and embedded into the HTML file

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body class>
    <div id="map" style="width: 600px; height: 400px">
      <div class="leaflet-map-pane" style="transform: rotate(0deg);">
        <div class="leaflet-tile-pane">...</div>
        <div class="leaflet-objects-pane">
          <div class="leaflet-shadow-pane"></div>
          <div class="leaflet-overlay-pane">
            <svg class="leaflet-zoom-animat" width="100%" height="100%">
              <g>
                <path stroke-linejoin="round" stroke="red" stroke-width="2px"
                  clickable"></path>
              </g>
            </svg>
          </div>
          <div class="leaflet-marker-pane"></div>
          <div class="leaflet-popup-pane"></div>
        </div>
      </div>
    </div>
  </body>
</html>
```



The browser natively interprets svg
and displays it

Questions



Exercise 2

- Basic Web Mapping with GeoJSON
 - Convert a GIS dataset to GeoJSON
 - Add the data to the web map
 - Further customize your map
 - Add interactivity (optional)