Map Tiles - Leaflet.js

InfoUma 2020-21 Andrea Marchetti

TMS - Tile Map Service

An efficient solution to publish maps on the web

Complexity in space (rather than in time)

Used by many map providers

Google Maps, Bing, Yahoo Maps, OpenStreetMaps, ...

Tile Map Service

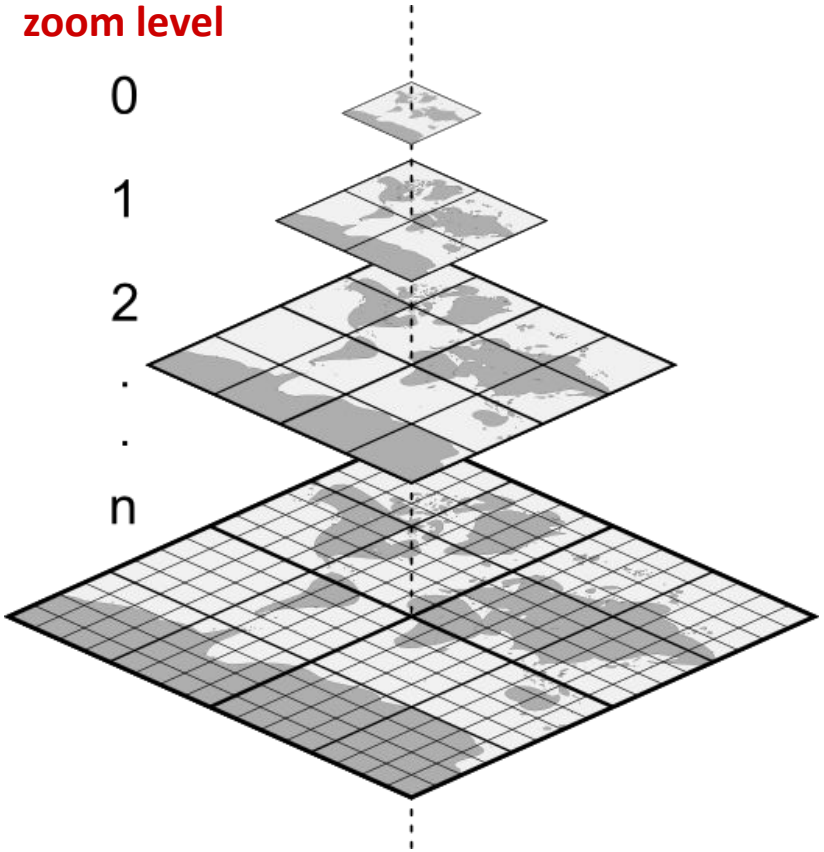
TMap Service or TMS, is a specification for tiled web maps, developed by the Open Source Geospatial Foundation.

The definition generally requires a URI structure which attempts to fulfill REST principles.

The TMS protocol

- fills a gap between the very simple standard used by OpenStreetMap and the complexity of the WMS (Web Map Service) standard,
- providing simple urls to tiles
- supporting alternate SRS (spatial referencing system)

TMS - Multi Resolution Image Pyramid



Maps are generated once for all level of zoom and then sliced into tiles

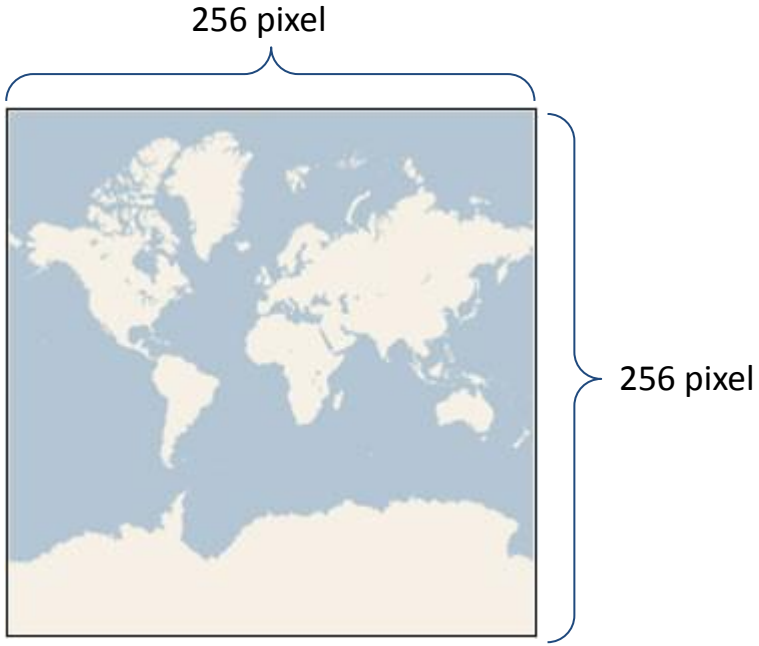
A map for a finite set of zoom levels

Each zoom level quadruples the number of tiles

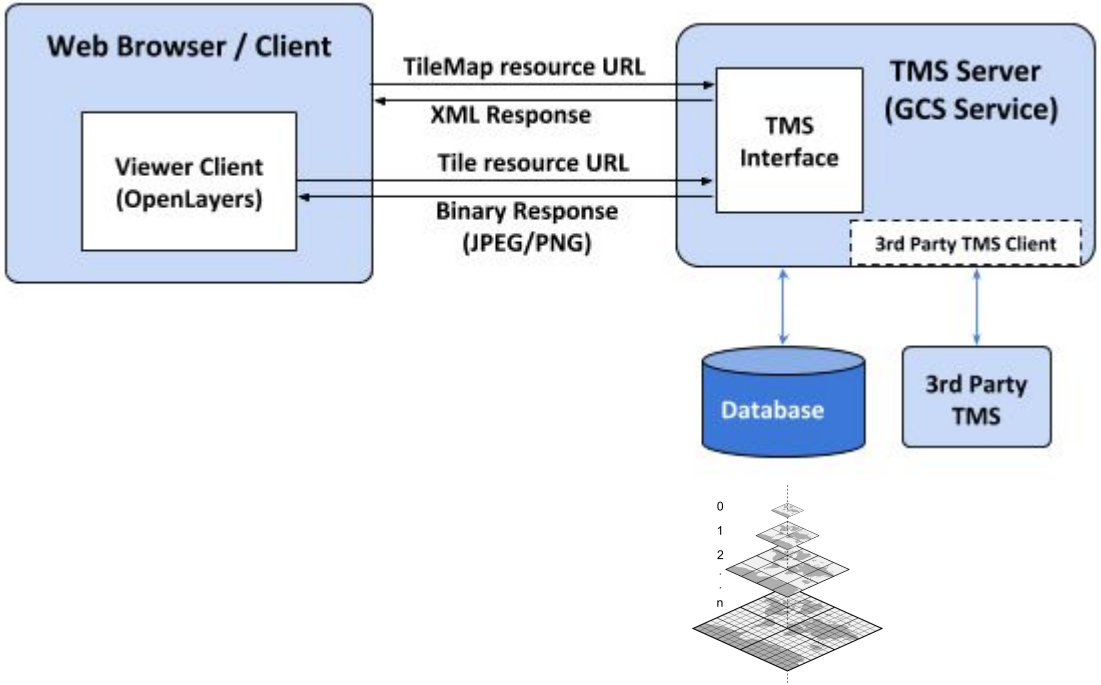
At zoom level **n** there are $4^n = 2^{n+1}$ tiles

TMS: tile

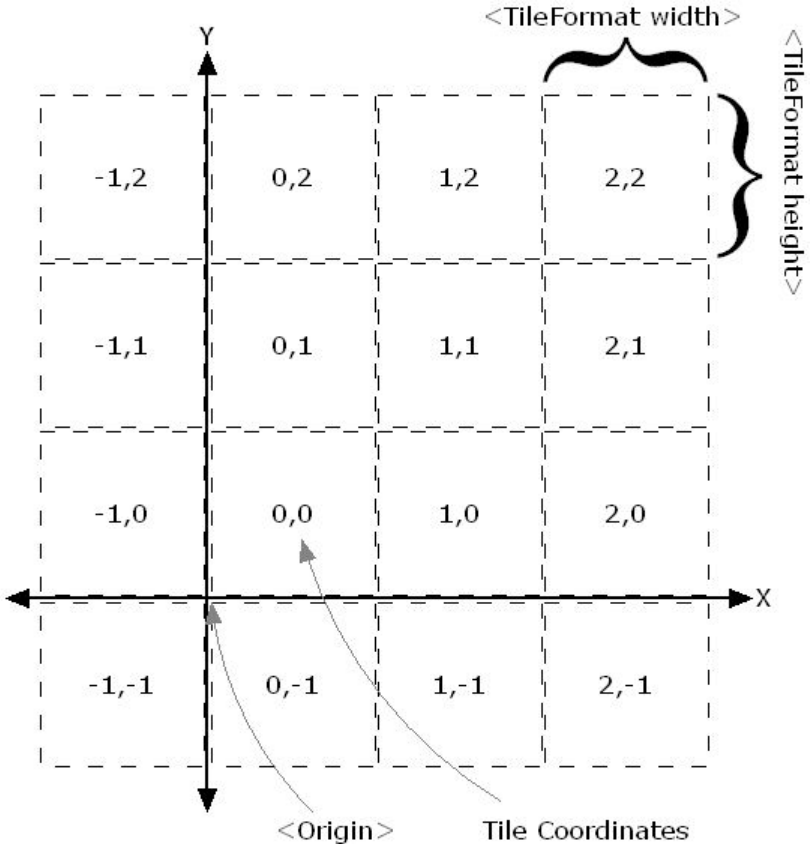
Every tile (any zoom) has a fixed dimension usually 256x256



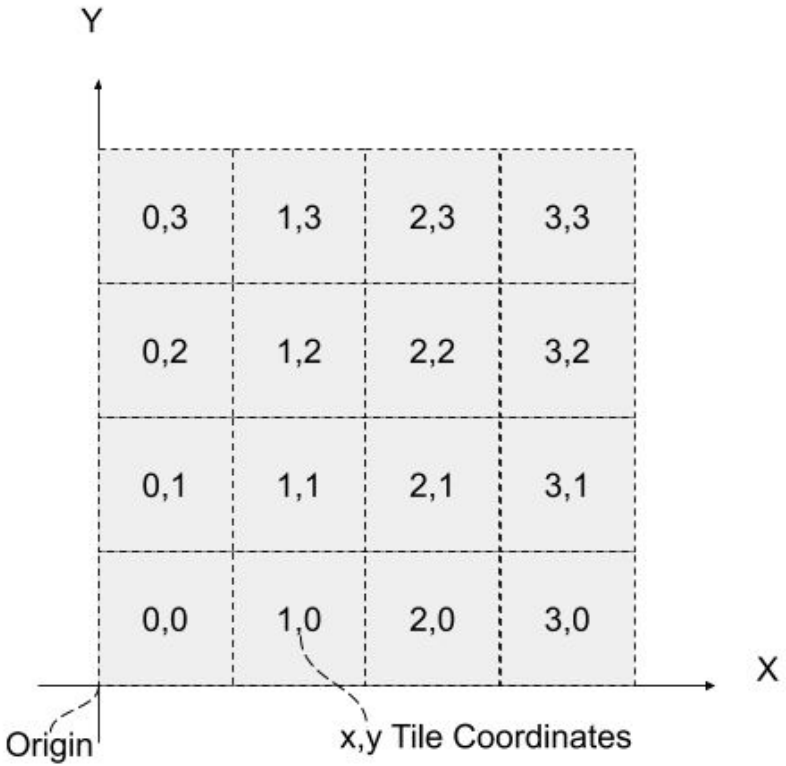
TMS Server


Web Browser / Client
Viewer Client
(OpenLayers)
TileMap resource URL
XML Response
Tile resource URL
Binary Response
(JPEG/PNG)
TMS Server
(GCS Service)
TMS
Interface
3rd Party TMS Client
Database
3rd Party
TMS
0
1
2
n

Tile coordinates

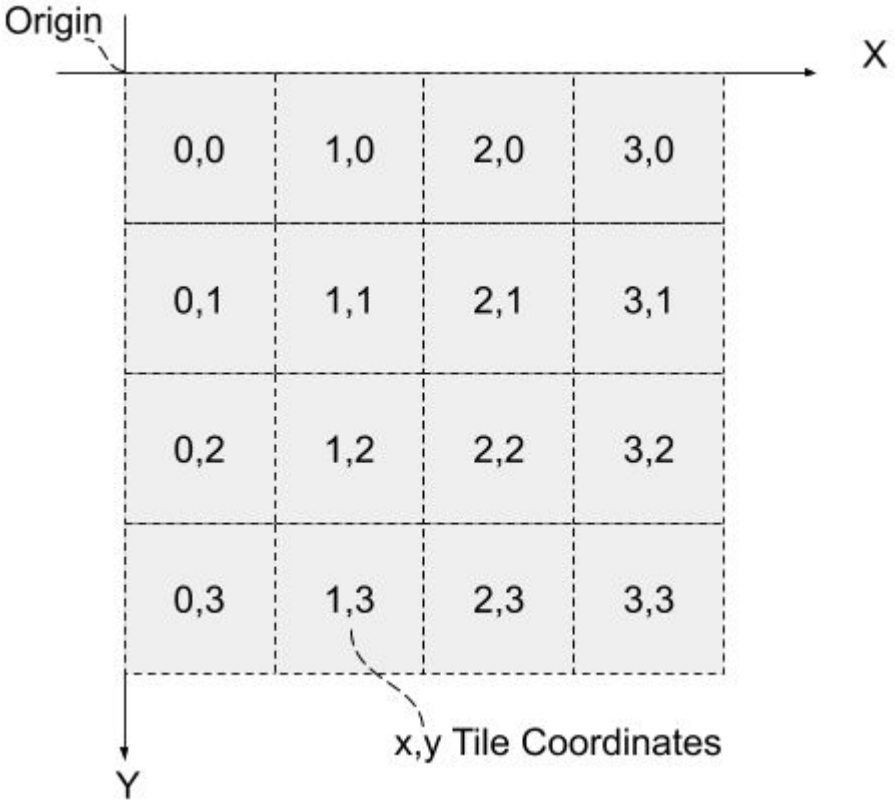


As depicted in the TMS specification—TileMap Diagram section, the Y-coordinates grow from south to north.



As depicted in the TMS specification—TileMap Diagram section, the Y-coordinates grow from south to north.

Tile Coordinates

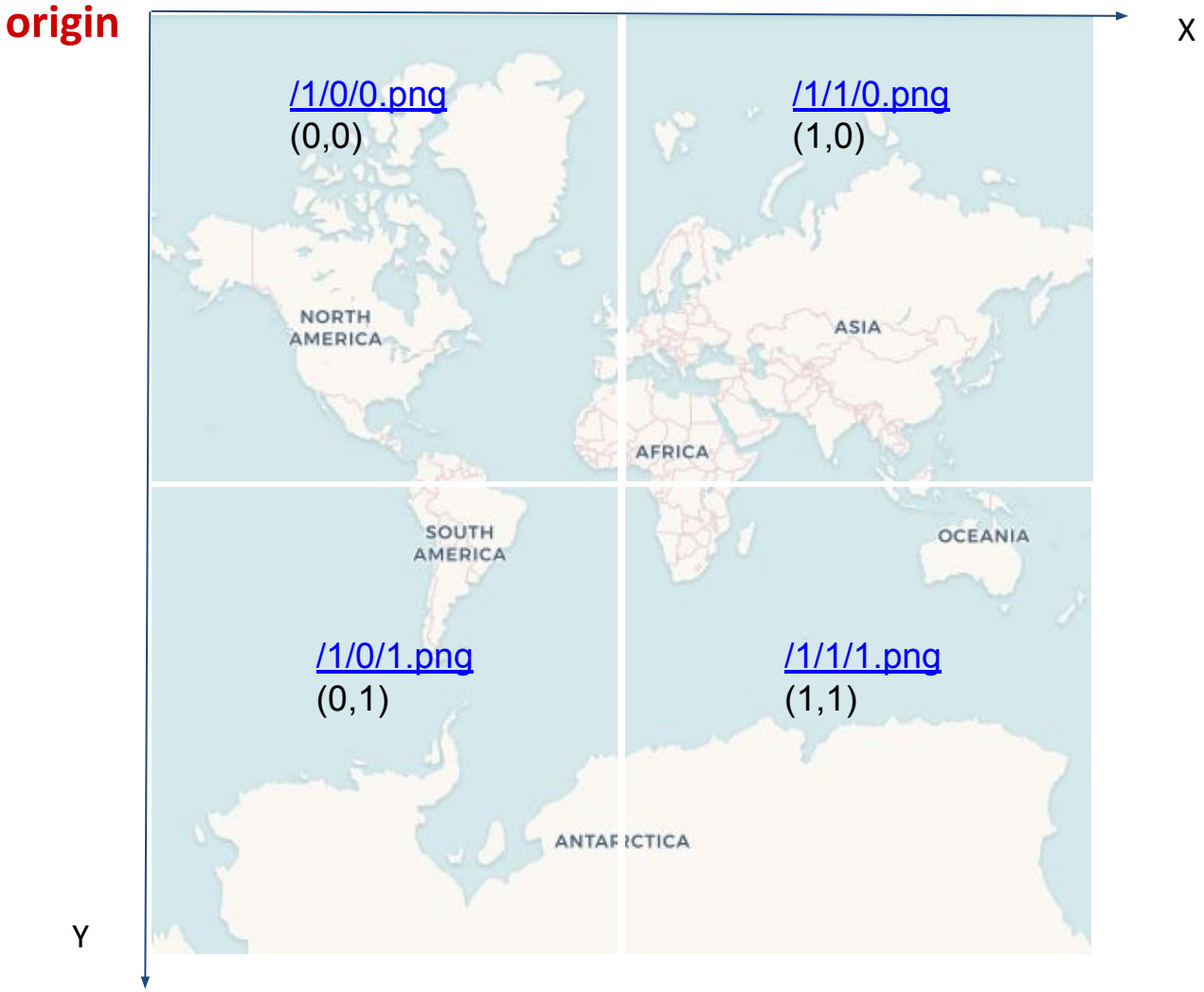


Some implementations have the opposite direction, with the grid origin at top left, and Y-coordinates numbered from north to south (e.g., OSM Tiles).
Depending on the implementation, it may be necessary to flip the y-coordinate

Tile Resource of CartoDB

https://a.basemaps.cartocdn.com/rastertiles/voyager/ 0/0/0.png

zoomLevel/X/Y.png



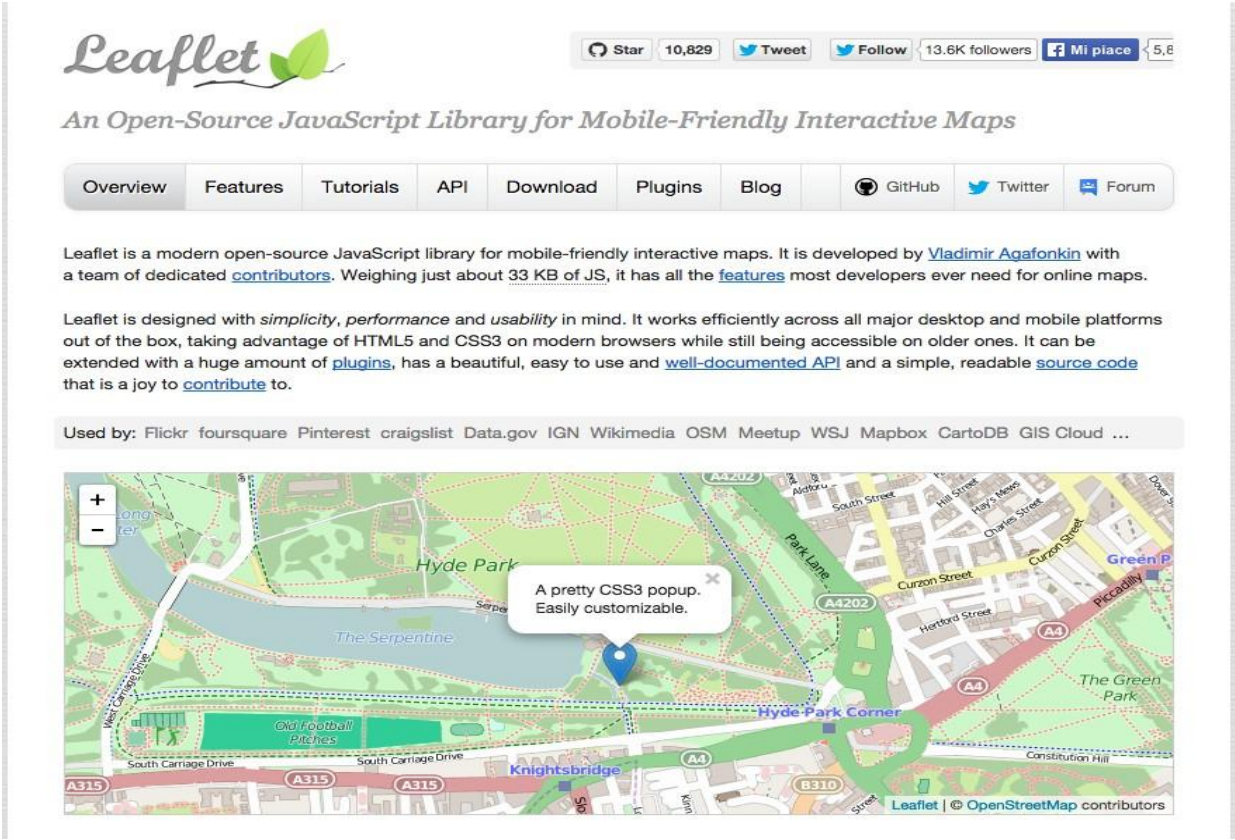
The 4 tiles at level 1 of CartoDB

Tile Map Server - SRS

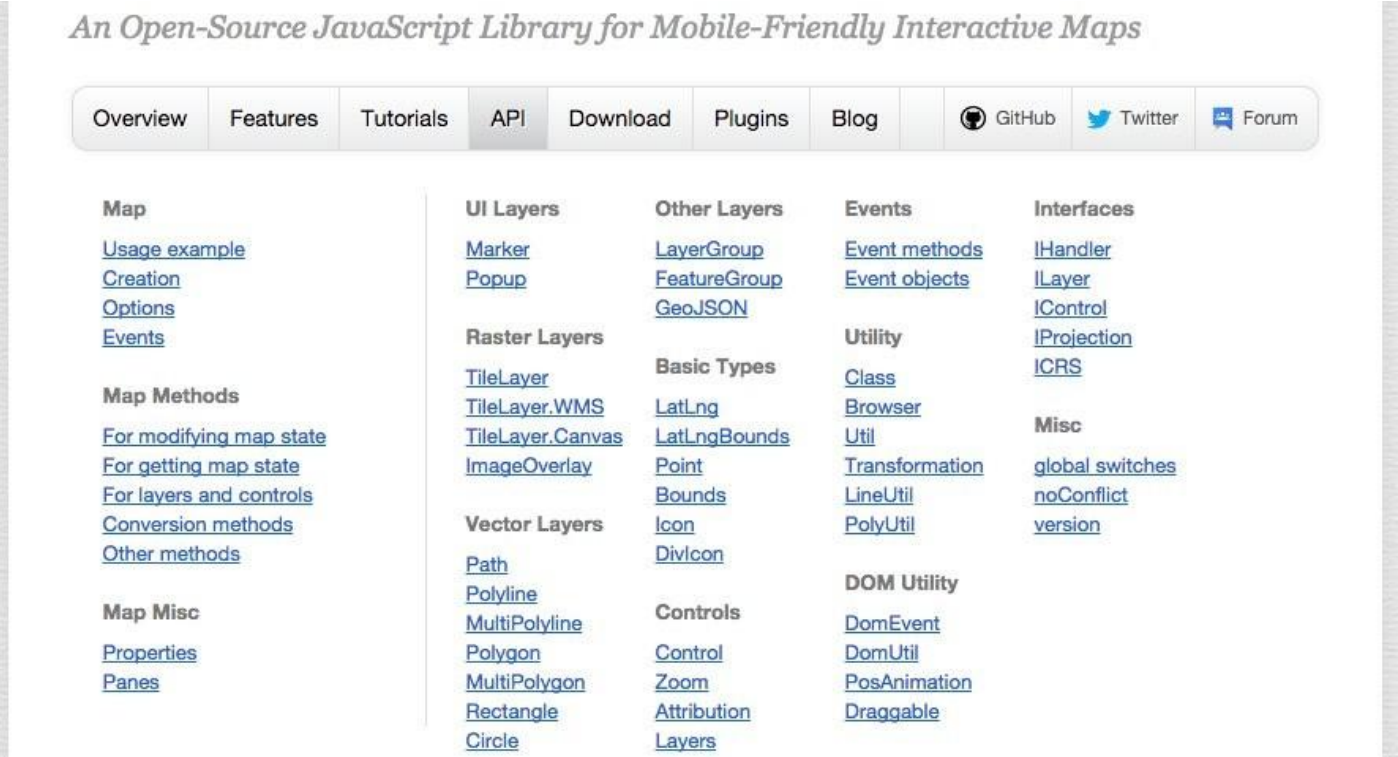
- To simplify coordinate mapping: cylindrical projection
- Two main reference systems:
  - Sphere Mercator (53004)
  - World Mercator (54004)
- Mercator Cyndric projection
  - Meridians are parallels
  - Conformal (preserves shapes)
  - Preserves directions

LeafLet

Leaflet.js 1.7.1



Leaflet.js - APIs



Leaflet.js

- A valid tool to provide tile-based maps
  - Open Source
  - Open Data ()
  - Free
- Easy to use API
- Lightweight lib (only 33kb)
- Support mobile applications

Free Tiles Providers

- OpenStreetMap
  - Some issues for high traffic services
- MapQuest Open License
  - Free, by attribution
  - Special configuration for heavy usage
- MapBox
  - Free tier
  - Customizable design (see next slide)
  - Same family as Leaflet.js

Commercial Tile Providers

- CloudMade
  - Mirror of OSM data till few years ago
  - Leaflet was born here
  - $30 per 1M tiles
- MapBox
  - Free for low traffic
  - $30 for 900k tiles
- ESRI

Tile map providers

[Leaflet Provider Demo]
[Leaflet-extras/leaflet-providers]
[Map Compare]
[27- reasons-not-to-use-google-maps]

Easy to install/use

// Insert link to CSS

```
<link rel="stylesheet"
href="http://cdnjs.cloudflare.com/ajax/libs/leaflet/0.7.3/leaflet.css" />
```

// Insert link to JS after link to CSS

```
<script src="http://cdnjs.cloudflare.com/ajax/libs/leaflet/0.7.3/leaflet.js ">
</script>
```

// Create a div element to contain the map

```
<div id="map"></div>
```

// Set height for the container

```
#map {height: 600px}
```

Easy to install/use

Create an object to handle the map

```
var map = L.map('map').setView([51.505, -0.09], 13);
```

Centre coordinates

Zoom Level

Select the tile provider

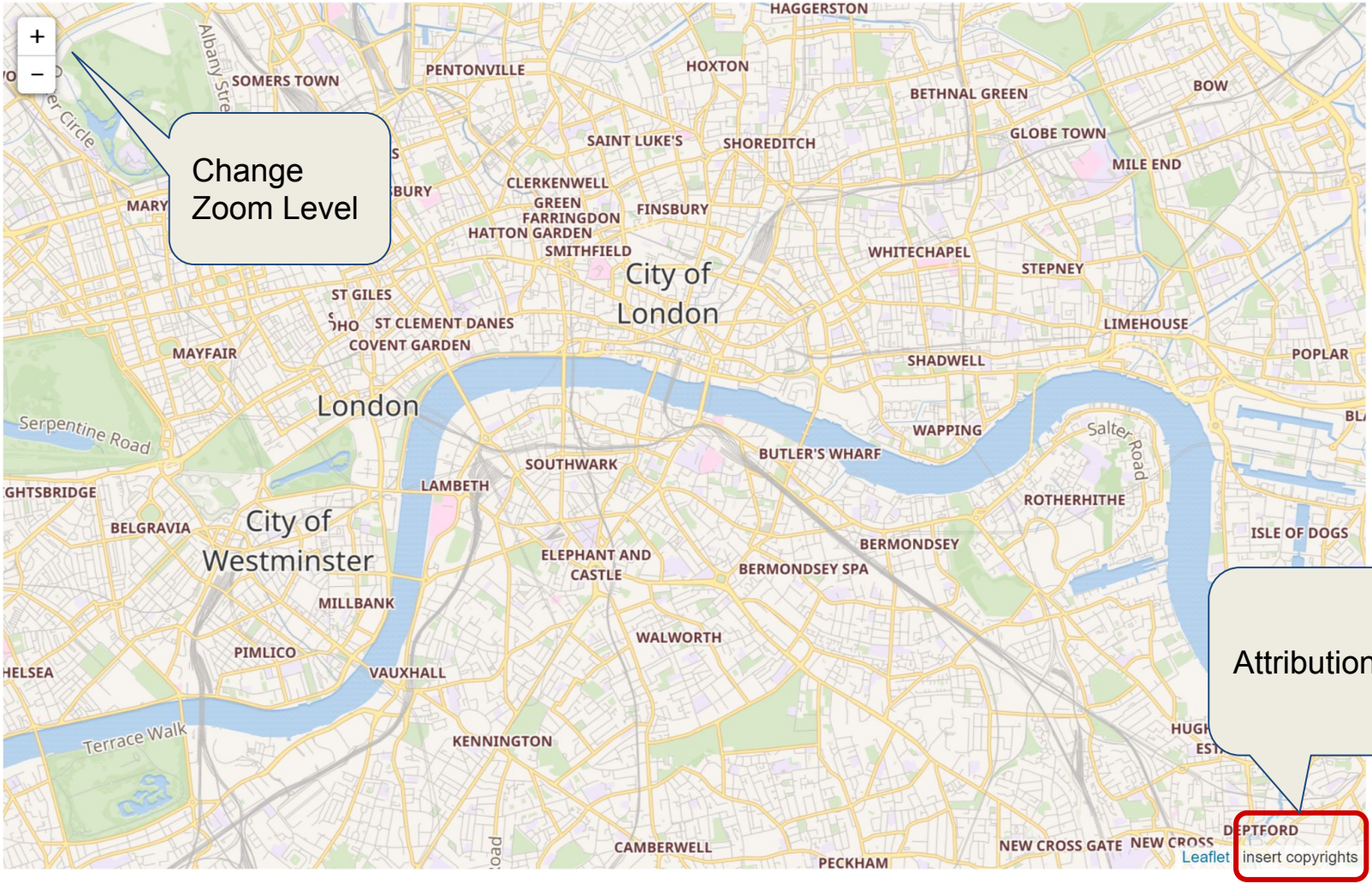
```
var tms = 'https://maps.wikimedia.org/osm-intl/{z}/{x}/{y}@2x.png';
```

Wikimedia tile map provider

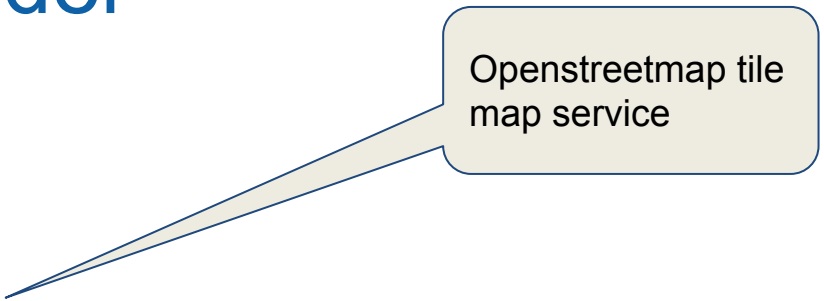
Connect the tiles to our map

```
L.tileLayer(tms,{attribution: ''}).addTo(map);
```

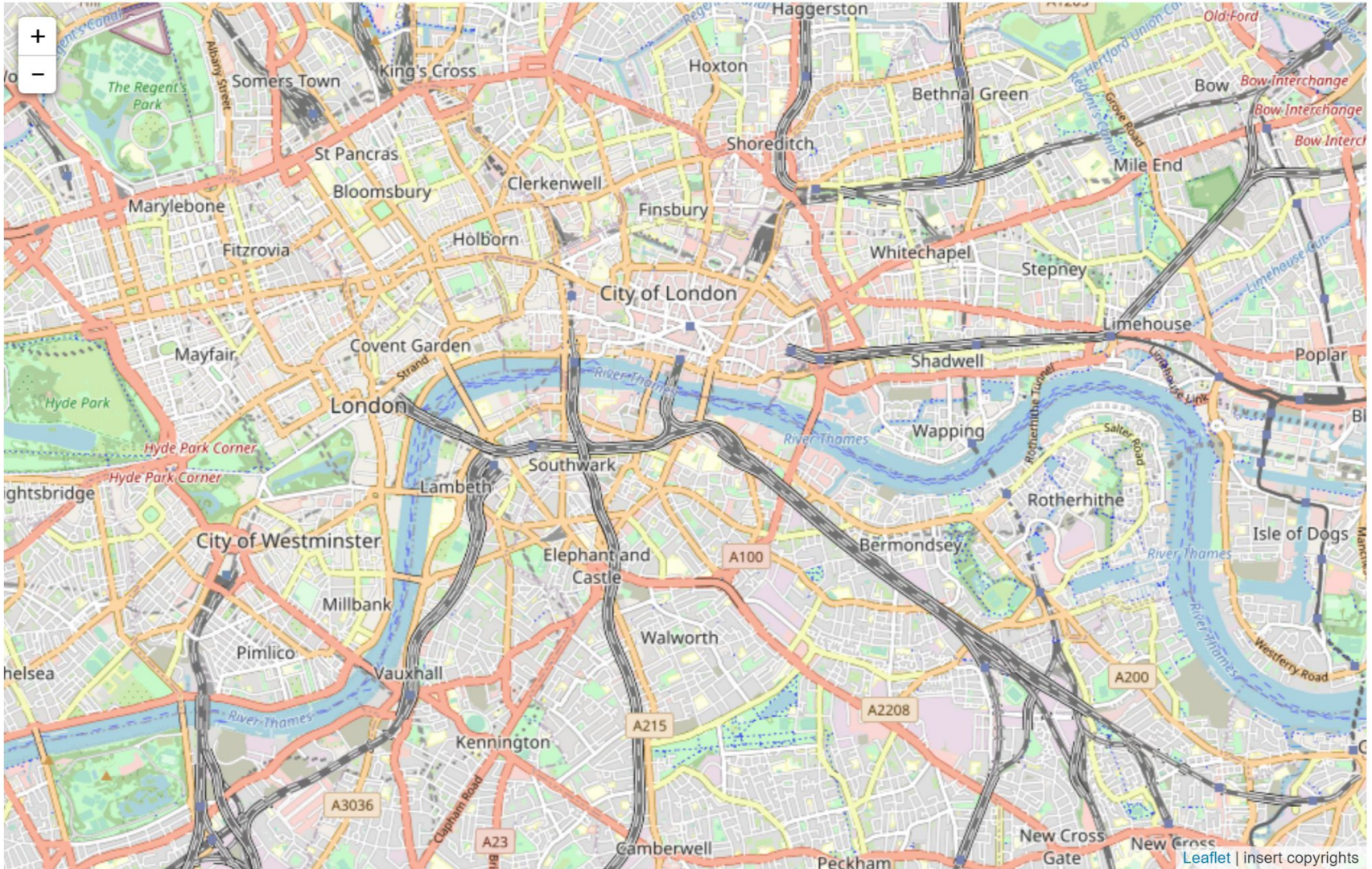
Copyright text as indicated by the tile provider


+
−
Change Zoom Level
HAGGERSTON
PENTONVILLE
HOXTON
SOMERS TOWN
BETHNAL GREEN
BOW
GLOBE TOWN
SAINT LUKE'S
SHOREDITCH
MILE END
CLERKENWELL
GREEN
FARRINGDON
FINSBURY
HATTON GARDEN
SMITHFIELD
WHITECHAPEL
STEPNEY
City of London
ST GILES
ST CLEMENT DANES
COVENT GARDEN
LIMEHOUSE
MAYFAIR
SHADWELL
POPLAR
London
Serpentine Road
WAPPING
Salter Road
BUTLER'S WHARF
SOUTHWARK
LAMBETH
ROTHERHITHE
BELGRAVIA
City of Westminster
BERMONDSEY
ISLE OF DOGS
ELEPHANT AND CASTLE
BERMONDSEY SPA
MILLBANK
WALWORTH
PIMLICO
VAUXHALL
Terrace Walk
KENNINGTON
CAMBERWELL
PECKHAM
NEW CROSS GATE
NEW CROSS
DEPTFORD
Leaflet | insert copyrights
Attribution

Change the tile provider




```
var tms = 'https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png'
```



Markers and geometries

```
// Marker
var marker = L.marker([51.5, -0.09]);

// Circle
var circleOptions = {radius:5, color:'red'}
var circle = L.circle([51.508, -0.11], circleOptions);


// Polygon
var polygon = L.polygon([[51.509,-0.08],[51.503,-0.06],[51.51,-0.04]]);
```

Layers

To display a marker on the map we have 2 options:

```
// add each one to the map
marker.addTo(map);
circle.addTo(map);
polygon.addTo(map);

// group inside a single layer and then add the layer to the map
myLayer = L.featureGroup([marker, circle, polygono]);
myLayer.addTo(map)
```

Interactions

```
marker.bindPopup("<b>Hello world!</b><br>I am a
popup for a Marker").openPopup();

circle.bindPopup("I am a Circle.");

polygon.bindPopup("I am a Polygon.");
```

Event handling

```
function onMapClick(e) {
    alert("You clicked the map at " + e.latlng);
}
map.on('click', onMapClick);
```

```
var popup = L.popup();
function onMapClick(e) {
    popup
    .setLatLng(e.latlng)
    .setContent("You clicked the map at " + e.latlng.toString())
    .openOn(map);
}
map.on('click', onMapClick);
```

Other Examples

- Mobile app
  - http://leafletjs.com/examples/mobile.html
- GeoJSON
  - http://leafletjs.com/examples/geojson.html
  - http://geojson.io/
- Tutorials
  - http://leafletjs.com/examples.html

Plugins

Tile & image layers

Basemap providers
Basemap formats
Non-map base layers
Tile/image display
Tile load
Vector tiles

Overlay data

Overlay data formats
Dynamic data loading
Synthetic overlays
Data providers

Overlay Display

Markers & renderers
Overlay animations
Clustering/decluttering
Heatmaps
DataViz

Overlay interaction

Edit geometries
Time & elevation
Search & popups
Area/overlay selection

Map interaction

Layer switching controls
Interactive pan/zoom
Bookmarked pan/zoom
Fullscreen
Minimaps & synced maps
Measurement
Mouse coordinates
Events
User interface
Print/export
Geolocation

Miscellaneous

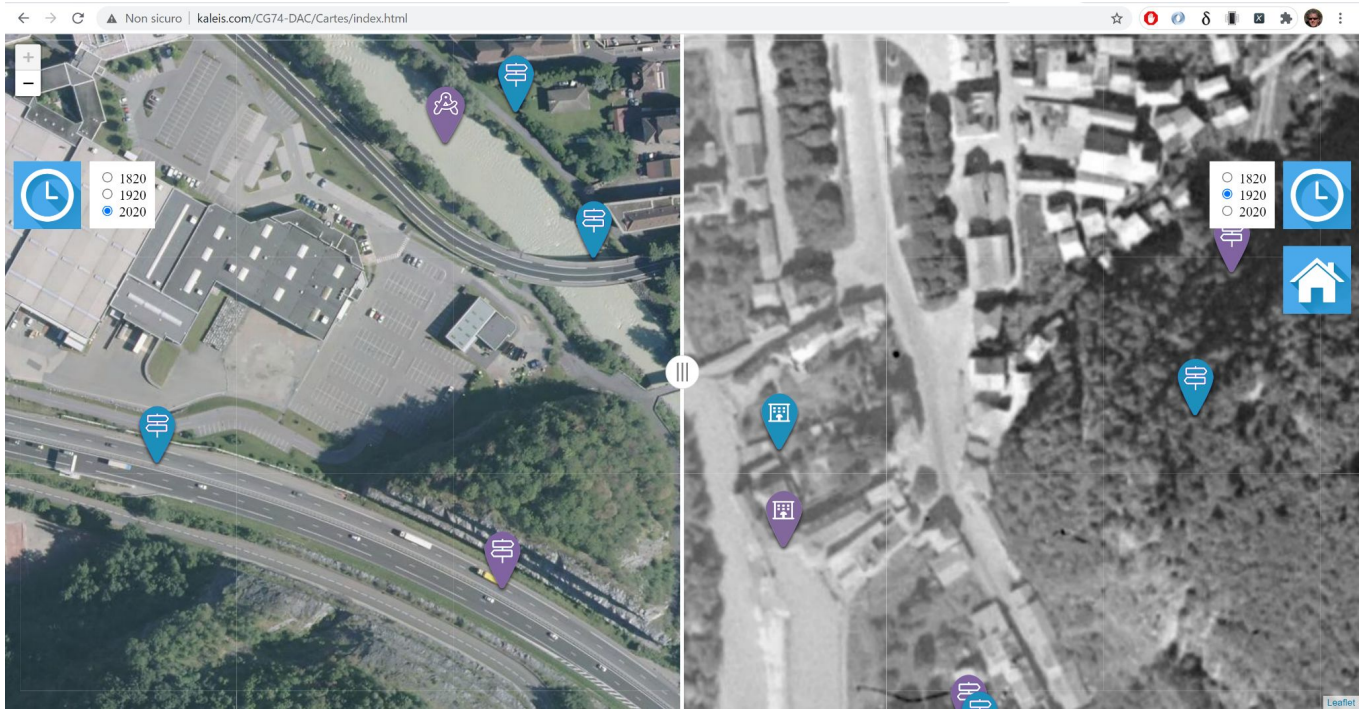
Geoprocessing
Routing
Geocoding
Plugin collections

Integration

Frameworks & build systems
3rd party

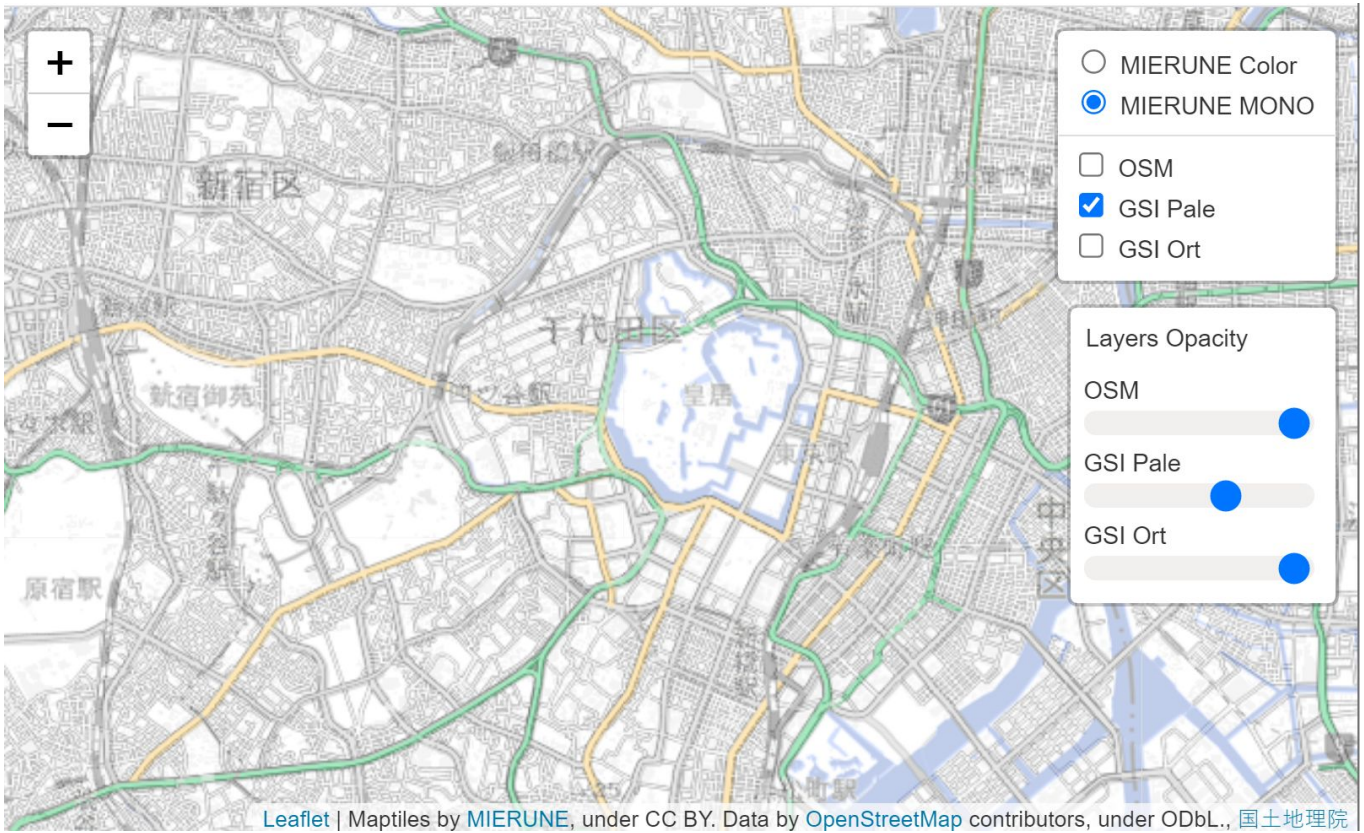
Develop your own

Plugin - Side by side

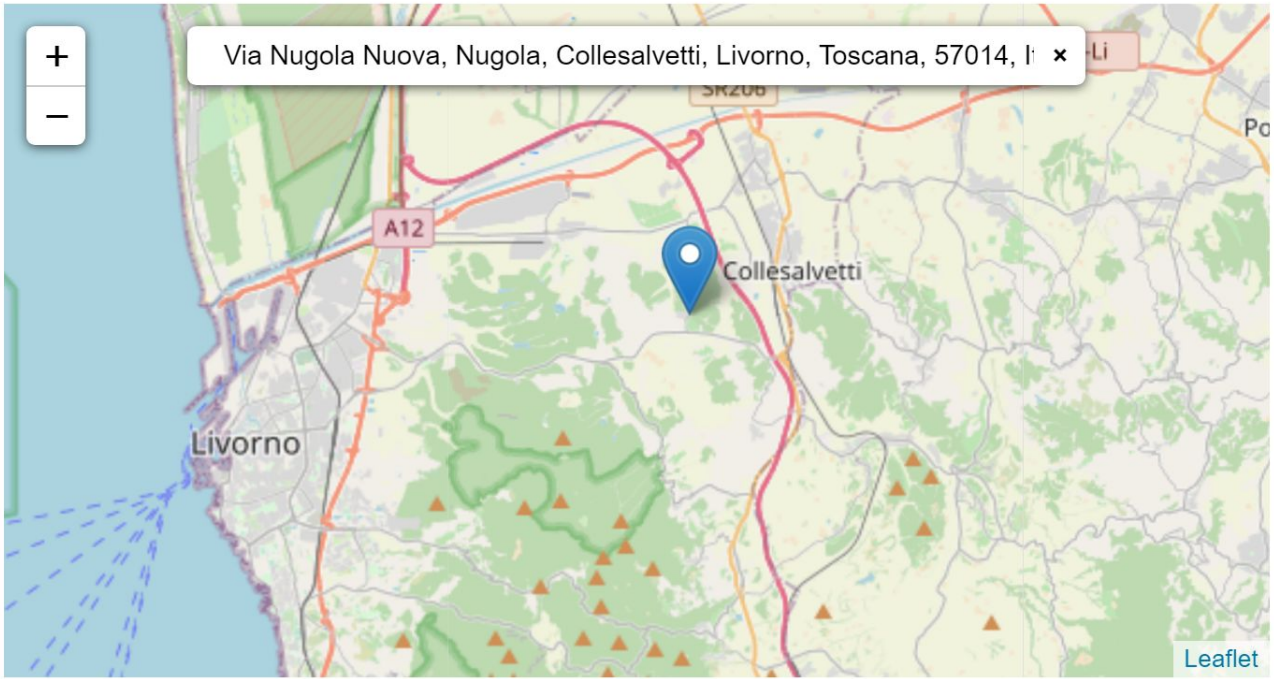


[A Leaflet control to add a split screen to compare two map overlays]

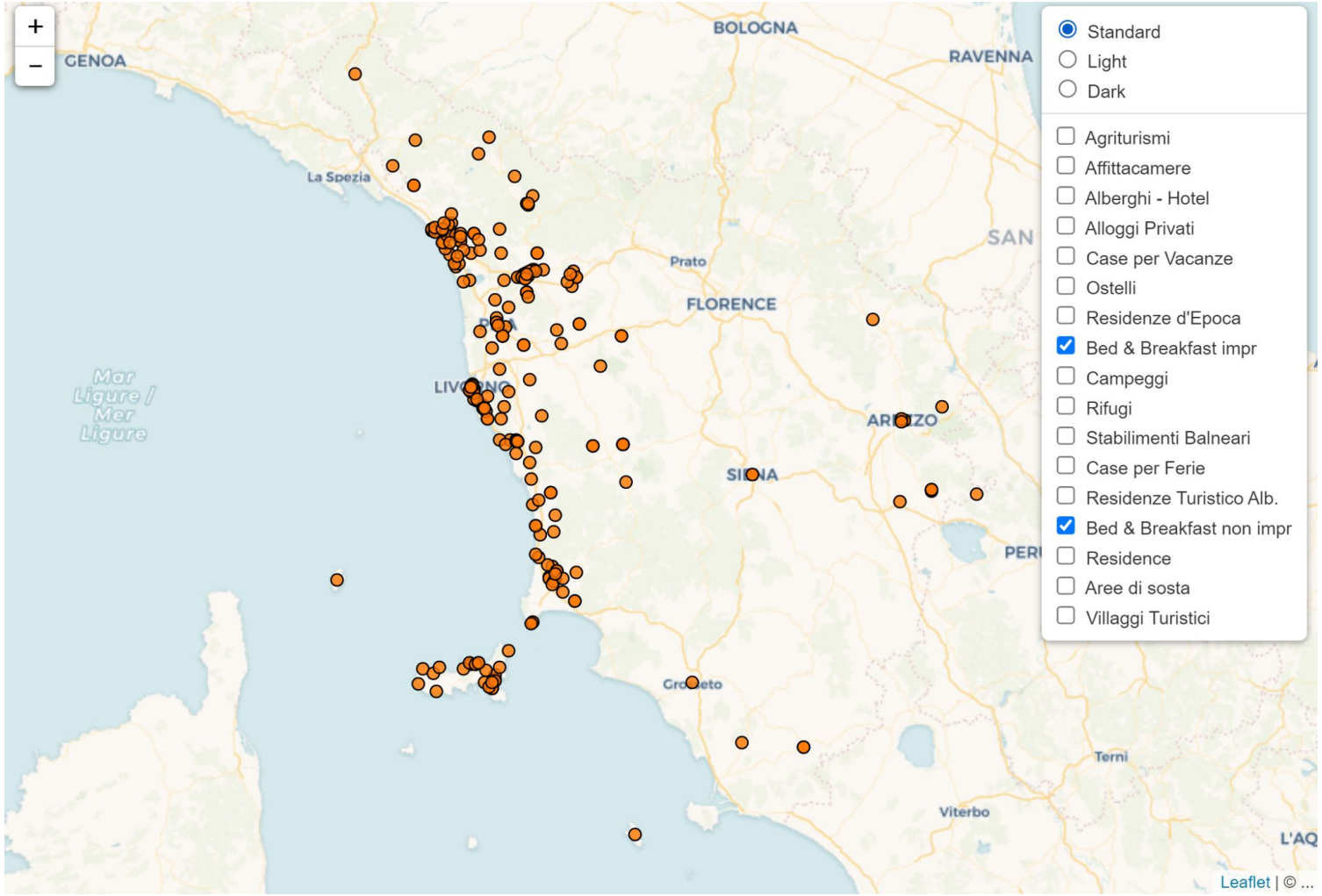
Plugin - Control layers opacity



Plugin - Geocoding



Un caso d’uso
caricare il file CSV delle strutture ricettive della Toscana interrogarlo con sintassi sql al fine di creare una mappa interattiva


+
−
GENOA
BOLOGNA
RAVENNA
La Spezia
Prato
FLORENCE
SAN
Mar Ligure / Mer Ligure
LIVORNO
AREZZO
SIENA
PERU
Grosseto
Terni
Viterbo
L'AQ
Standard
Light
Dark
Agriturismi
Affittacamere
Alberghi - Hotel
Alloggi Privati
Case per Vacanze
Ostelli
Residenze d'Epoca
Bed & Breakfast impr
Campeggi
Rifugi
Stabilimenti Balneari
Case per Ferie
Residenze Turistico Alb.
Bed & Breakfast non impr
Residence
Aree di sosta
Villaggi Turistici
Leaflet | © ...

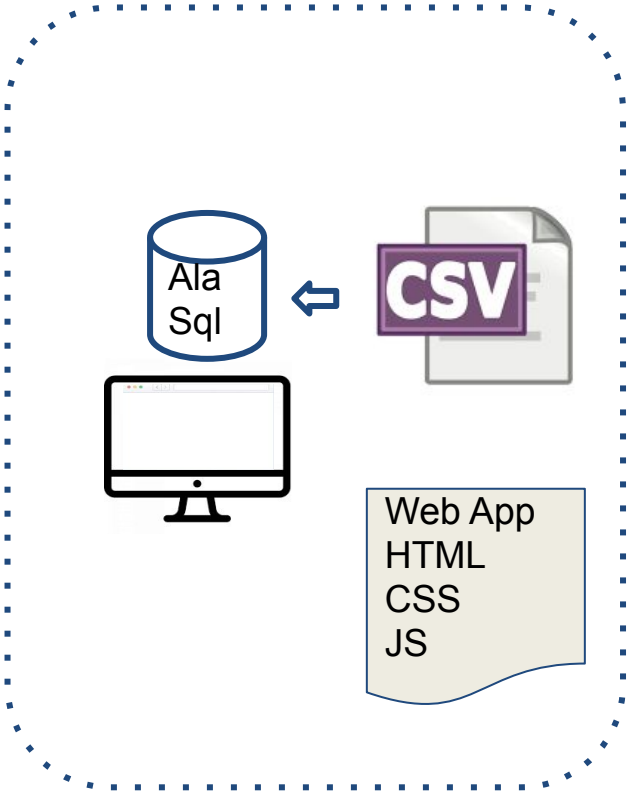
AlaSql Js

Javascript library
Client-side SQL and NoSQL Database
Very Fast Query for BI applications
Easy import from CSV, XLSx
Works in browser, node.js, mobile apps

Leaflet Js

Javascript library
For Interactive Maps
Version 1.7.1 September 2020

Client Tier



Questa semplificazione elimina la necessità di un DB. Utile in fase di sviluppo

Codice HTML

```
<!DOCTYPE html>
<html>
<head>
      <title>Strutture ricettive</title>
      <!-- AlaSql -->
      <script src="https://cdn.jsdelivr.net/alasql/0.3/alasql.min.js"></script>
      <!-- Leaflet -->
      <link rel="stylesheet" href="http://cdnjs.cloudflare.com/ajax/libs/leaflet/0.7.3/leaflet.css" />
      <script src="http://cdnjs.cloudflare.com/ajax/libs/leaflet/0.7.3/leaflet.js"></script>
</head>
<body>
      <h1>Strutture ricettive della Regione Toscana</h1>
      <div id="map" style="height: 600px"></div>
      <script>
      // Qui inserire il codice
      ...
      </script>
</body>
</html>
```

Caricamento delle 2 librerie javascript

Elemento div dove sarà inserita la mappa di altezza 600px

Codice Javascript

```
<script>
var map;

// Caricamento di una selezione del file CSV
alasql.promise('SELECT nome,lat,lon,tipologia FROM CSV("struttureRicettiveXall.csv",{headers:true,
quote:"\'",separator:"|"})')
      .then(function(data){
            console.log(data);
      })
      .catch(console.error);


</script>
```

```
var map;

// Caricamento di una selezione del file CSV
alasql.promise('SELECT nome,lat,lon,tipologia FROM CSV("struttureericettiveXall.csv",
               {headers:true, quote:"\'",separator:"|"})')
      .then(function(data){
            console.log(data);
            // si crea una tabella
            alasql("CREATE TABLE strutture (nome string, lat number, lon number, tipologia string)");
            alasql.tables.strutture.data = data; // si inseriscono nella tabella strutture
            init();
      })
      .catch(console.error);

function init(){
}
```

```
function init(){
    //Per ogni tipo di struttura carico le strutture e creo un layer
    var tipi = alasql("SELECT DISTINCT tipologia FROM strutture");
    console.log(tipi);

}
```

```
function init(){
   //Per ogni tipo di struttura carico le strutture e creo un layer
   var tipi = alasql("SELECT DISTINCT tipologia FROM strutture");
   var overlays = {};
   for (var tipo of tipi){
      //Carico le strutture di una certa tipologia
      var strutture = alasql("SELECT nome, lat, lon FROM strutture WHERE tipologia=?",tipo['tipologia']);
      ...
   }
   ...
}
```

```
function init(){
   //Per ogni tipo di struttura carico le strutture e creo un layer
   var tipi = alasql("SELECT DISTINCT tipologia FROM strutture");
   var overlays = {};
   for (var tipo of tipi){
      //Carico le strutture di una certa tipologia
      var strutture = alasql("SELECT nome, lat, lon FROM strutture WHERE
tipologia=?",tipo['tipologia']);
      //Creo un layer e lo inserisco in overlays
      var markerOptions = {radius: 4, fillColor: "#ff7800", color: "#000000",
                           weight: 1, opacity: 1,     fillOpacity: 0.8 }
      var markers = []; // array che ospita i markers delle strutture
      for (var struttura of strutture)
           markers.push(L.circleMarker(struttura,markerOptions).bindPopup(struttura['nome']));
      overlays[tipo['tipologia']] = L.featureGroup(markers);
   }
   ...
}
```

```
function init(){
   //Per ogni tipo di struttura carico le strutture e creo un layer
   var tipi = alasql("SELECT DISTINCT tipologia FROM strutture");
   var overlays = {};
   for (var tipo of tipi){
      //Carico le strutture di una certa tipologia
      var strutture = alasql("SELECT nome, lat, lon FROM strutture WHERE
tipologia=?",tipo['tipologia']);
      //Creo un layer e lo inserisco in overlays
      var markerOptions = {radius: 4, fillColor: "#ff7800", color: "#000000",
                           weight: 1, opacity: 1,     fillOpacity: 0.8 }
      var markers = []; // array che ospita i markers delle strutture
      for (var struttura of strutture)
           markers.push(L.circleMarker(struttura,markerOptions).bindPopup(struttura['nome']));
      overlays[tipo['tipologia']] = L.featureGroup(markers);
   }
   //Creazione della mappa
   map = L.map('map', {center:[43.4, 11], zoom: 8, layers: [cartoLayer, overlays['Agriturismi']] });
   //Creazione dei controllo dei layers
   L.control.layers (baseLayers, overlays).addTo(map);
}
```

```
var map;
// Base Layers
let cartoAttr       = '&copy; ...';
let carto           = 'https://{s}.basemaps.cartocdn.com/rastertiles/voyager/{z}/{x}/{y}.png';
let cartoLigth      = 'https://{s}.basemaps.cartocdn.com/light_all/{z}/{x}/{y}.png';
let cartoDark       = 'https://{s}.basemaps.cartocdn.com/dark_all/{z}/{x}/{y}.png';
let cartoLightLayer = L.tileLayer(cartoLigth, {attribution: cartoAttr });
let cartoDarkLayer  = L.tileLayer(cartoDark , {attribution: cartoAttr });
let cartoLayer      = L.tileLayer(carto     , {attribution: cartoAttr });
let baseLayers      = { "Standard": cartoLayer, "Light": cartoLightLayer, "Dark": cartoDarkLayer};
```