Introduction to Mapping APIs

Lecturer: Ms. Aneeqa Abrar
Email: aneeqa.abrar@gmail.com

1

Table of Content

2

Introducing Mapping APIs

3

Introducing Mapping APIs

OpenLayers

Google Maps

Leaflet

Cesium JS

4

OpenLayers

- Object-oriented JavaScript library version 2+ 3
  - OL 2.X using Prototype.js and Rico library)
  - OL 3 using Google's Closure Tools (compiler+ library)
- Lets you add maps to any web page by embedding OpenLayer.js
  - no server-side dependencies
  - Easily reusable component similar to Google Maps and BING Web Mapping APIs
- Input Formats
  - Bing, Open Street map, Google Maps, WMS, Vector layers, GeoRSS, WFS, KML
- Standard Tools
  - Google Like zoom bar, standard functions like zoom in/out pan

Clustering example

5

Google Maps

- The Google Maps JavaScript API lets you embed Google Maps in your app
- The latest version (v3) of this API is especially designed to be faster and more applicable to mobile devices
- The API provides a number of utilities for manipulating maps and adding content to the map through a variety of services
- Available both as free and paid services

6

Leaflet

A popular, lightweight (38 kb) JavaScript library

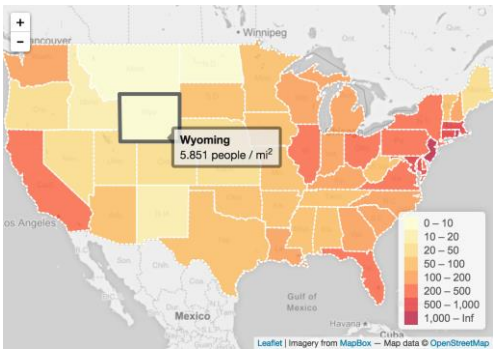
No external dependencies

Lots of features, plugins

Ideal for making mobile-friendly interactive maps

Can be used on all major desktop and mobile platforms

Uses JS, HTML5 and CSS



7

Cesium JS

An open-source JavaScript library for world-class 3D globes and maps in the browser

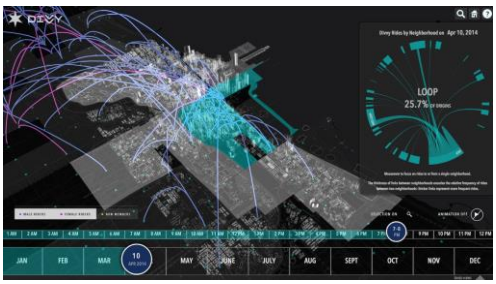
JavaScript Software with WebGL (Web Graphics Library) for displaying
- 3D virtual globe
- 2D map
- 2.5D Columbus View

Time-dynamic Scenes

Multiple terrain sources

Overlays :
- Raster : WMS, TMS, OSM, Bing
- Vector : glTF, CZML, KML, Shapefiles

Extensible with plugins



8

Let’s Start Coding!

9

Let’s Start Coding!

Preparing Page

Add a Map

Markers and Circles

Working with Popups

Dealing with Events

GeoJSON

Documentation & Plugins

10

Preparing Page

Leaflet CSS file in the head section:

```
<link rel="stylesheet" href="https://unpkg.com/leaflet@1.4.0/dist/leaflet.css"
integrity="sha512-
puBpdR0798OZvTTbP4A8Ix/l+A4dHDD0DGqYW6RQ+9jxkRFclaxxQb/SJAWZfWAkuyeQUytO7+7N4QKrDh+drA=="
crossorigin=""/>
```

Leaflet JavaScript after Leaflet's CSS:

```
<script src="https://unpkg.com/leaflet@1.4.0/dist/leaflet.js" integrity="sha512-
QVftwZFqvtRNi0ZyCtsznlKSWOStnDORoefr1enyq5mVL4tmKB3S/EnC3rRJcxCPavG10IcrVGSmPh6Qw5lwrg=="
crossorigin=""></script>
```

11

Div element with an id:

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

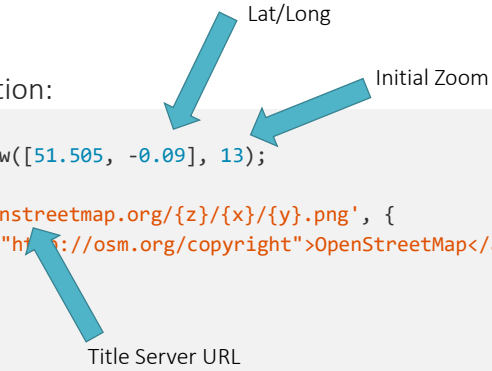
Ensure the map container has a defined height by setting it in CSS:

```
#mapid {
    height: 500px;
    width: 700px;
}
```

12

Add a Map

Leaflet CSS file in the head section:




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

L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="ht   ://osm.org/copyright">OpenStreetMap</a> contributors',
    maxZoom: 18,
    minZoom: 10
}).addTo(mymap);
```

13

Markers, Circles and Polygons

Add a Marker:

```
var marker = L.marker([51.5, -0.09]).addTo(mymap);
```

Add a Circle:

```
var circle = L.circle([51.508, -0.11], {
    color: 'red',
    fillColor: '#f03',
    fillOpacity: 0.5,
    radius: 500
}).addTo(mymap);
```

Add a Polygon:

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

14

Working with Popups

Default Popups

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

Custom Popups

```
var popup = L.popup()
    .setLatLng([51.5, -0.09])
    .setContent("I am a standalone popup.")
    .openOn(mymap);
```

15

Dealing with Events

A basic onClick event

```
function onMapClick(e) {

    alert("You clicked the map at " + e.latlng);
}

mymap.on('click', onMapClick);
```

16

A customized event

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

mymap.on('click', onMapClick);
```

17

GeoJSON

GeoJSON is a format for encoding a variety of geographic data structures […]. A GeoJSON object may represent a region of space (a Geometry), a spatially bounded entity (a Feature), or a list of Features (a FeatureCollection). GeoJSON supports the following geometry types: **Point**, **LineString**, **Polygon**, MultiPoint, MultiLineString, MultiPolygon, and GeometryCollection. Features in GeoJSON contain a Geometry object and additional properties, and a FeatureCollection contains a list of Features.

```
L.geoJSON(geoJSON).addTo(mymap);
```

18

```geojson
var geojsonFeature = {
    "type": "Feature",
    "properties": {
        "name": "Coors Field",
        "amenity": "Baseball Stadium",
        "popupContent": "This is where the Rockies play!"
    },
    "geometry": {
        "type": "Point",
        "coordinates": [-104.99404, 39.75621]
    }
};
```

```
L.geoJSON(geojsonFeature).addTo(mymap);
```

19

GeoJSON - Point

```geojson
var myLines = [{
    "type": "LineString",
    "coordinates": [[-100, 40], [-105, 45], [-110, 55]]
}, {
    "type": "LineString",
    "coordinates": [[-105, 40], [-110, 45], [-115, 55]]
}];
```

Add Style

```
var myStyle = {
    "color": "#ff7800",
    "weight": 5,
    "opacity": 0.65
};
```

Add Layer

```
L.geoJSON(myLines,(
    style: myStyle
}).addTo(mymap);
```

20

GeoJSON - Lines

```
var states = [{
    "type": "Feature",
    "properties": {"party": "Republican"},
    "geometry": { "type": "Polygon",
    "coordinates": [[ [-104.05, 48.99], [-97.22, 48.98], [-96.58, 45.94], [-104.03, 45.94],
    [-104.05, 48.99]
    ]]
    }
}, {
    "type": "Feature",
    "properties": {"party": "Democrat"},
    "geometry": { "type": "Polygon",
    "coordinates": [[ [-109.05, 41.00], [-102.06, 40.99], [-102.03, 36.99], [-109.04,
    36.99], [-109.05, 41.00]
    ]]
}
}];
```

GeoJSON - Polygon

21

```
L.geoJSON(states, {
    style: function(feature) {
                        } switch (feature.properties.party)
    {
            case 'Republican': return {color: "#ff0000"};
            case 'Democrat': return {color: "#0000ff"};

    }
}). addTo(mymap);
```

22

Documentation & Plugins

For more refer to the leaflet documentation:

https://leafletjs.com/reference-1.4.0.html

For plugins refer to:

https://leafletjs.com/plugins.html

23

Thank You!

24