

Web Картографија

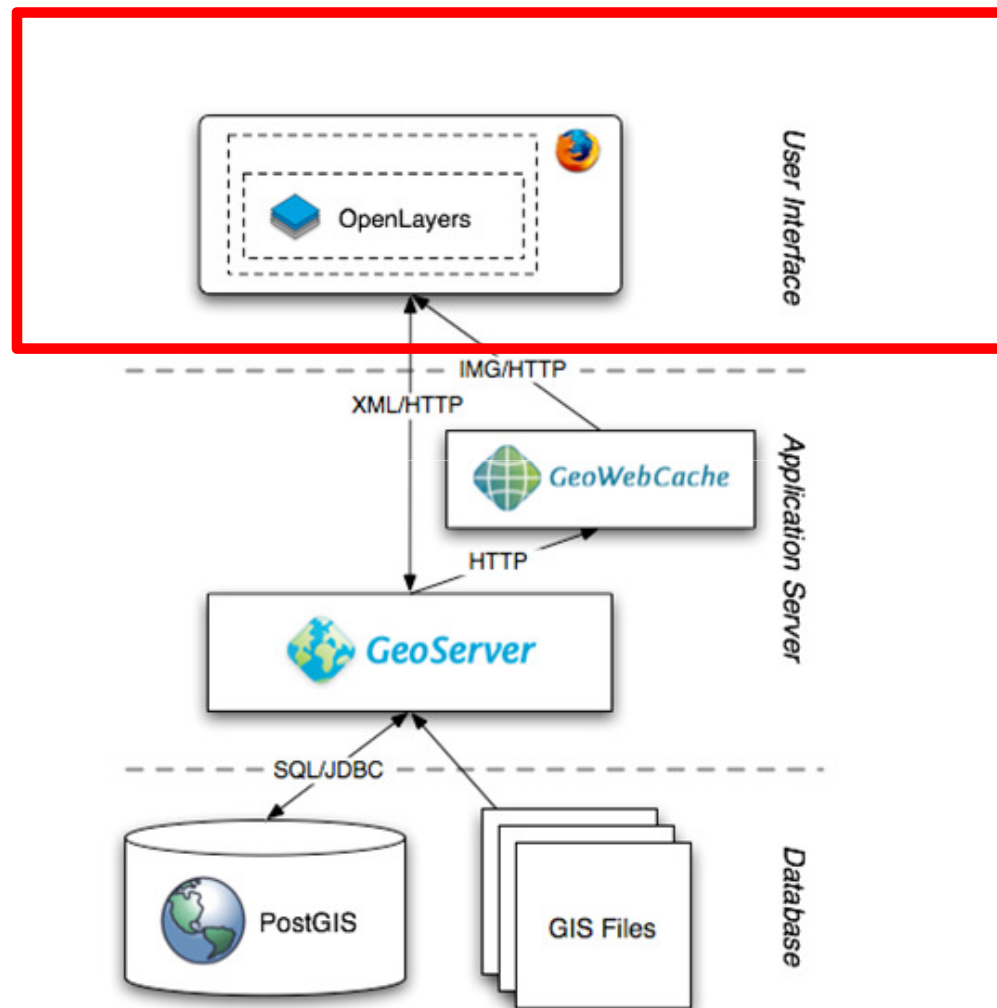
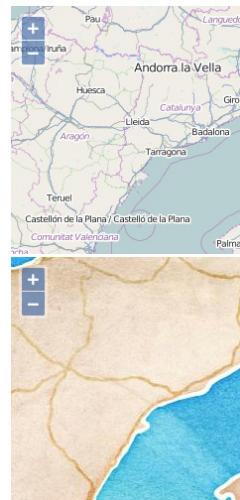
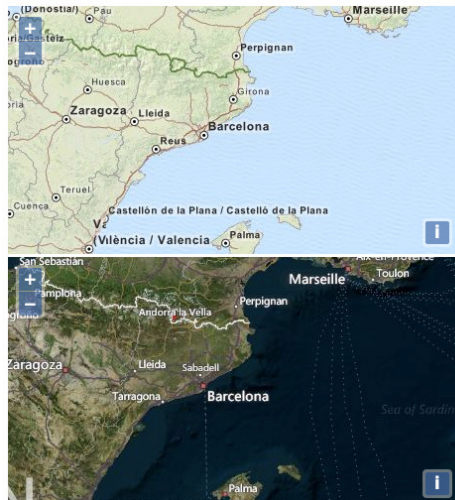
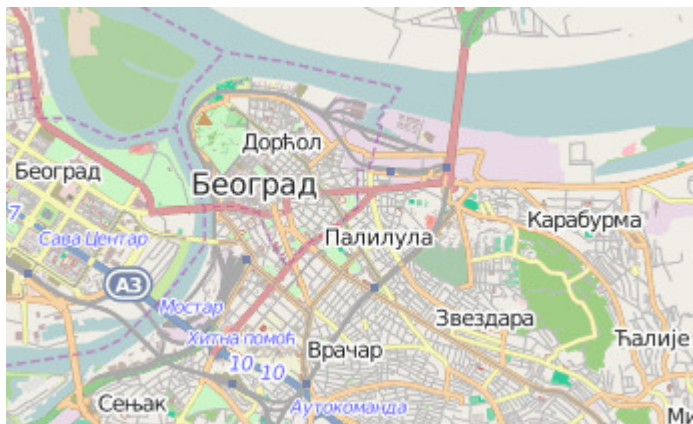
OpenLayers3 - 1.

Доц. др Милан Килибарда

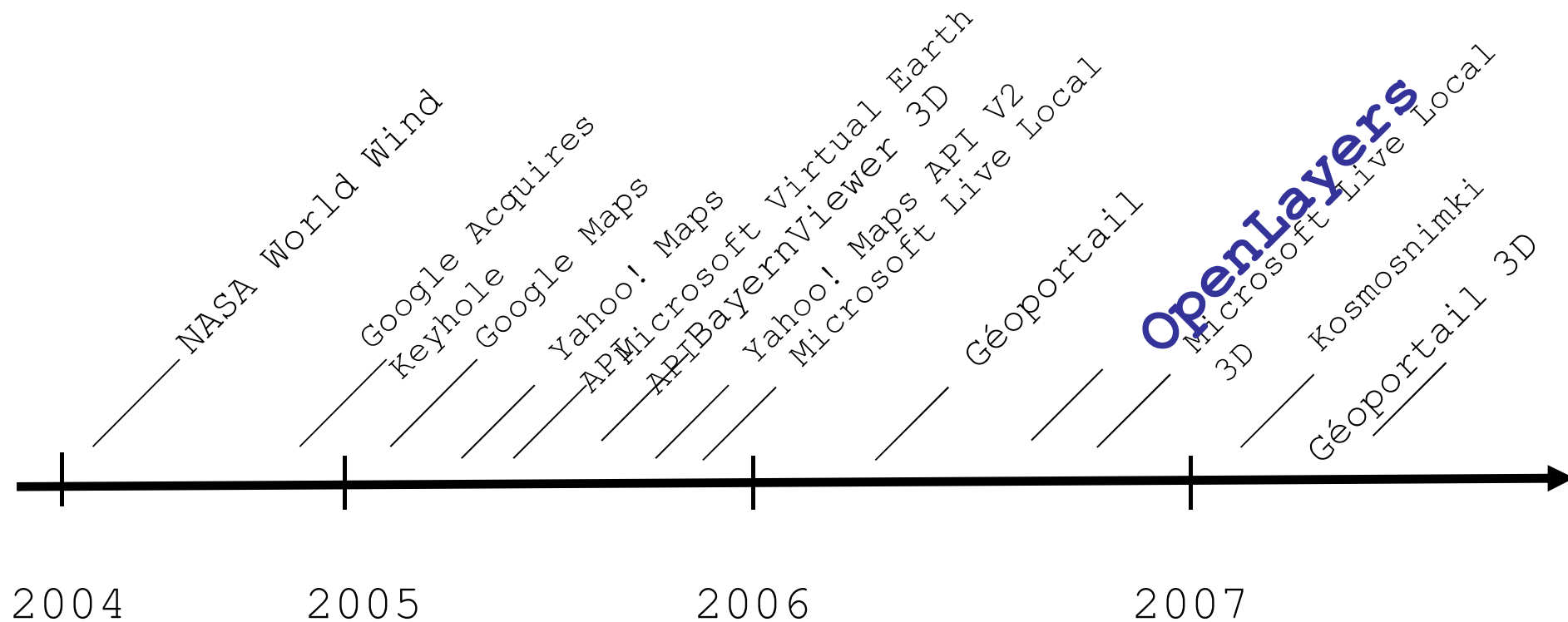
kili@grf.bg.ac.rs

Грађевински факултет,
Одсек за геодезију и геоинформатику,
Универзитет у Београду

Садржај



Историјски преглед



OL3 - 2014

<http://openlayers.org/>

OpenLayers

OpenLayers (OL) је open source JavaScript билиотека за приказ карата у браузерима. OL обезбеђује API за израду комплексних web базираних картографских апликација.

Подаци се могу комбиновати из различитих извора без захтева за процесирање са серверске стране, а да лејери буду картирани у браузеру.

OL клијент

OL може бити Web клијент за:

1. OGC web сервисе (WFS-T, WMS, WMST, WCS (XML, GML)), ...
2. Комерцијалне сервисе MSN Virtual Earth, Bing, Mapquest, Google Maps, ...
3. Open source стандарде као GeoJSON, GML, WKT ...

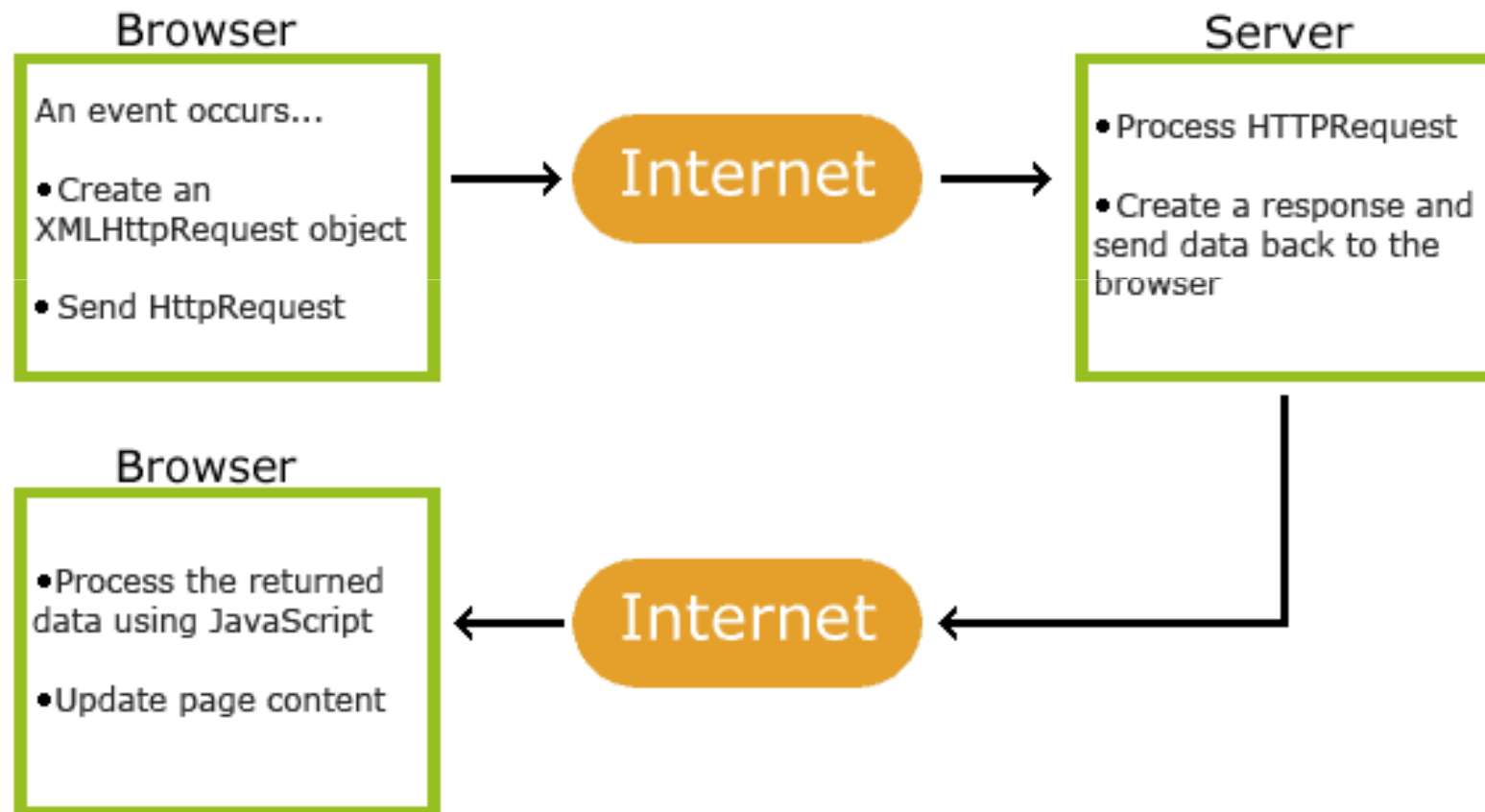
OpenLayers користи AJAX

Ајах је име додељено модерним веб апликацијама, раније познат под називом динамични HTML (DHTML) и удаљено скриптовање.

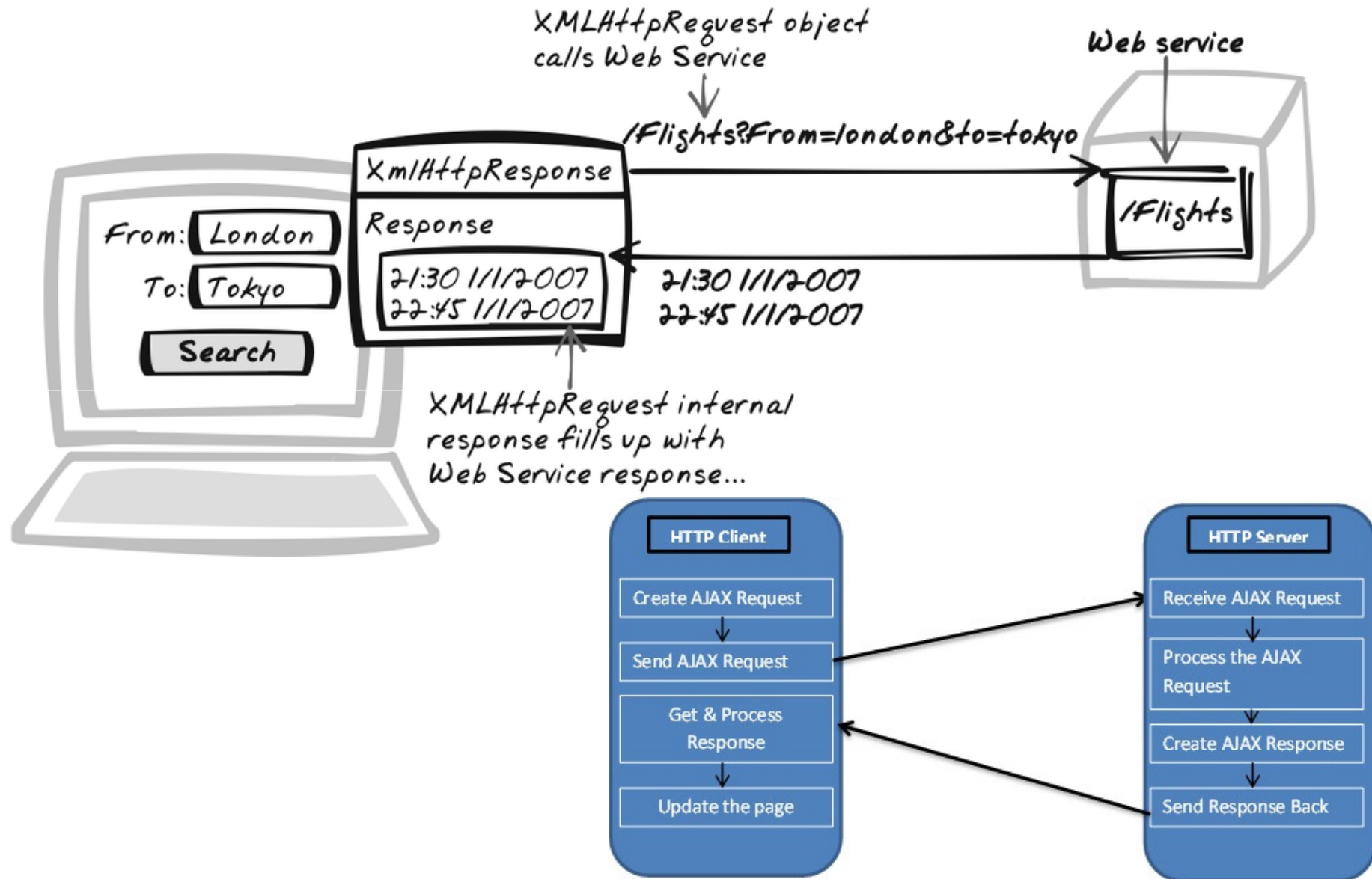
Суштина ове технологије је у томе што веб апликацији омогућује брзину сличну оној коју имају стандардне десктоп апликације.

Ајах технологија као темељ Web 2.0 концепту намењена је страницама које су динамичне, интерактивне и ефикасне у смислу брзине и удобности коришћења.

AJAX



XMLHttpRequest



AJAX философија

Ажурира се само део странице не цела страница, кад корисник захтева промену.

Архитектура Ajax базиране web апликације мање оптерећује сервер на коме су подаци, а већи део захтева се обрађује у машини која је садржана у прегледачу корисника, што знатно повећава ефикасност.

Интеракција са корисником карте се интерпретира у кратком времену, док се раније процес захтевао серверски рад најчешће користећи PHP скрипте, а онда се резултат враћао као модификована web страница која је онда цела морала бити изнова учитана.

AJAX компоненте

Кључне компоненте ове технологије су:

- стандарди базирани на XHTML и CSS (*Cascade Style Sheet*),
- динамичка презентација и интеракција података заснована на DOM (*Document Object Model*), размена и управљање подацима коришћењем XML (или JSON),
- асинхроно враћање подацима коришћењем XMLHttpRequest и
- JavaScript програмски језик који повезује све претходно побројане компоненте у целину.

OpenLayers3

Учитава се АПТИ и стилизација у Web апликацији

Include the JavaScript file, typically [ol.js](#), with the library implementation.

Include the CSS file, typically [ol.css](#), with the styles for some the elements of the library, like the map or the controls.

АПИ, CSS -wiki

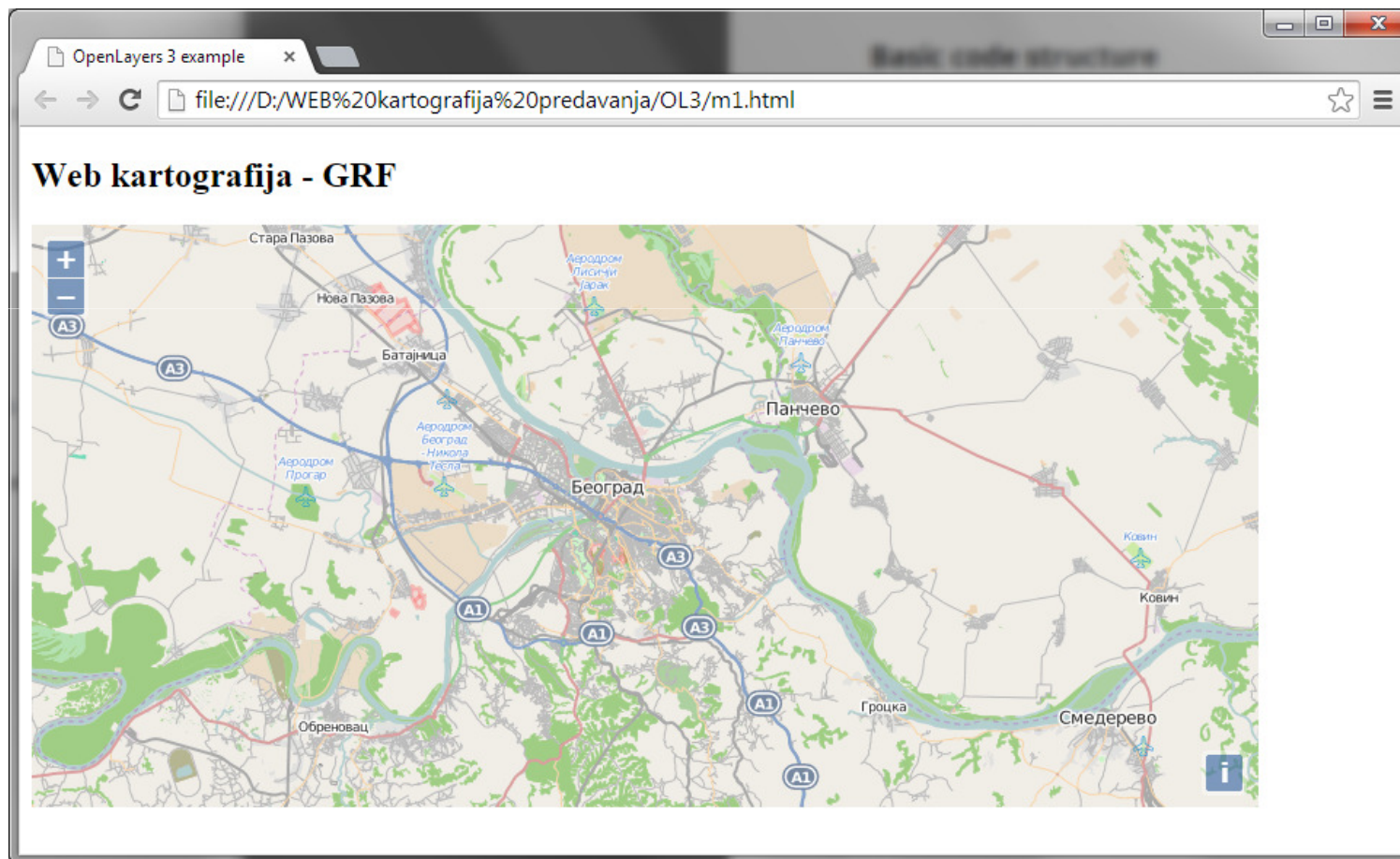
In computer programming, an **application programming interface (API)** is a set of routines, protocols, and tools for building software applications.

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language.

ОЛ пример

```
<!doctype html>
<html lang="en">
  <head>
    <link rel="stylesheet" href="http://openlayers.org/en/v3.0.0/css/ol.css" type="text/css">
    <style>
      .map {height: 400px; width: 90%;}
    </style>
    <script src="http://openlayers.org/en/v3.0.0/build/ol.js" type="text/javascript"></script>
    <title>OpenLayers 3 example</title>
  </head>
  <body>
    <h2>Web kartografija - GRF</h2>
    <div id="map" class="map"></div>
    <script type="text/javascript">
      var map = new ol.Map({
        target: 'map', // The DOM element that will contains the map
        layers: [
          // Add a new Tile layer getting tiles from OpenStreetMap source
          new ol.layer.Tile({
            source: new ol.source.OSM()
          })
        ],
        // Create a view centered on the specified location and zoom level
        view: new ol.View({
          center: ol.proj.transform([ 20.492420, 44.798095], 'EPSG:4326', 'EPSG:3857'),
          zoom: 10
        })
      });
    </script>
  </body>
</html>
```

ОЛ пример



Контејнер - секција за карту

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

ДОМ елемент у коме се смешта карта.
Секција XHTML документа.

Стил и димензије

`<style>`

```
.map {height: 400px;  
      width: 90%;}
```

`</style>`

или

`<style>`

```
#map {height: 400px;  
      width: 90%;}
```

`</style>`

Kapta

```
var map = new ol.Map({  
  target: 'map', // The DOM element that will contains the map  
  layers: [  
    // Add a new Tile layer getting tiles from OpenStreetMap source  
    new ol.layer.Tile({  
      source: new ol.source.OSM()  
    })  
  ],  
  // Create a view centered on the specified location and zoom level  
  view: new ol.View({  
    center: ol.proj.transform([ 20.492420, 44.798095],  
      'EPSG:4326', 'EPSG:3857'),  
    zoom: 10  
  })  
});
```

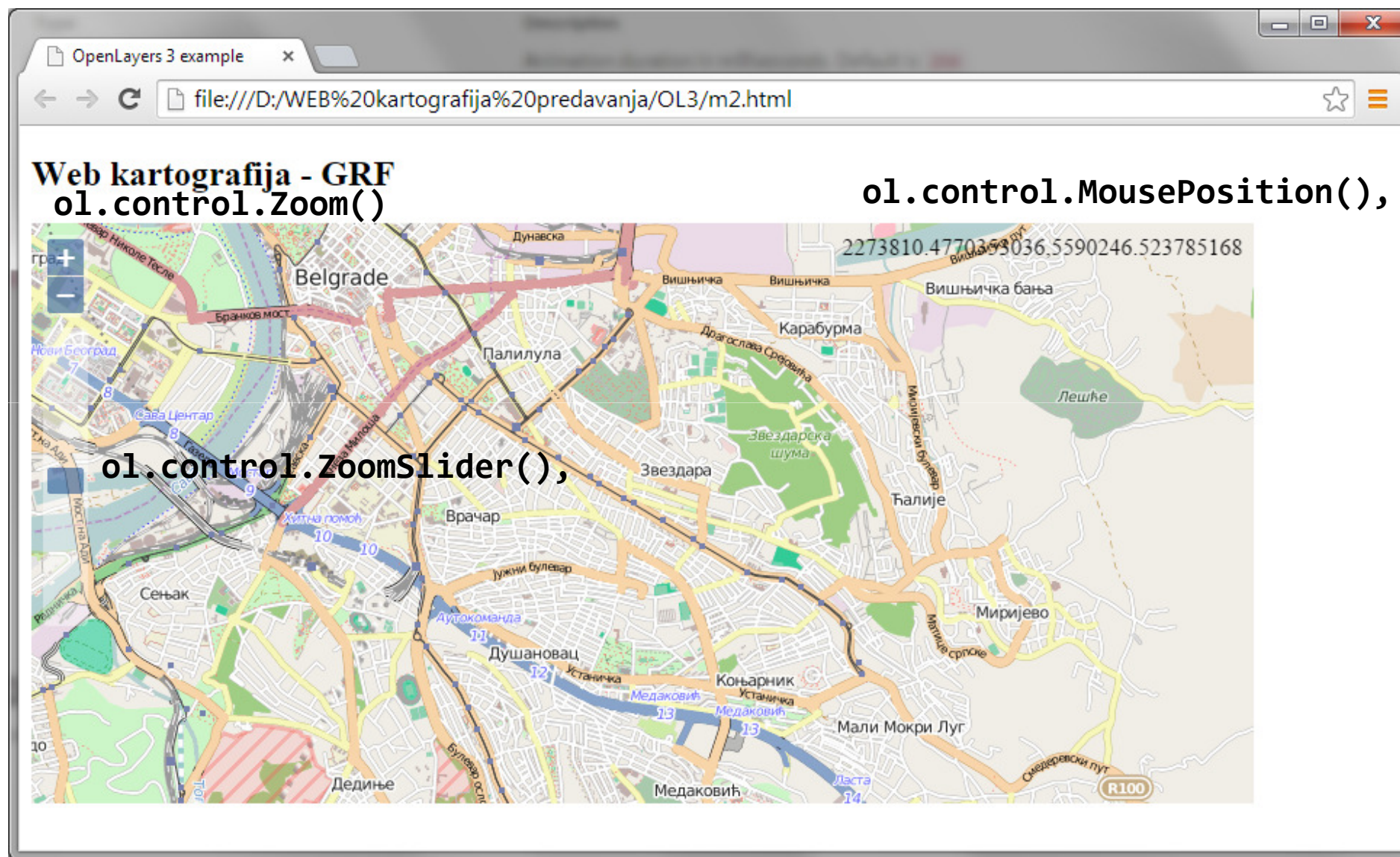
ol.Map <http://openlayers.org/en/v3.0.0/apidoc/ol.Map.html>

Name	Type	Description
controls	ol.Collection .< ol.control.Control > Array.< ol.control.Control > undefined	Controls initially added to the map. If not specified, ol.control.defaults() is used.
interactions	ol.Collection .< ol.interaction.Interaction > Array.< ol.interaction.Interaction > undefined	Interactions that are initially added to the map. If not specified, ol.interaction.defaults() is used.
layers	Array.< ol.layer.Base > ol.Collection .< ol.layer.Base > undefined	Layers. If this is not defined, a map with no layers will be rendered.
logo	boolean string olx.LogoOptions undefined	The map logo. A logo to be displayed on the map at all times. If a string is provided, it will be set as the image source of the logo. If an object is provided, the src property should be the URL for an image and the href property should be a URL for creating a link. To disable the map logo, set the option to false. By default, the OpenLayers 3 logo is shown.
overlays	ol.Collection .< ol.Overlay > Array.< ol.Overlay > undefined	Overlays initially added to the map. By default, no overlays are added.
renderer	ol.RendererType Array.<(ol.RendererType string)> string undefined	Renderer. By default, Canvas, DOM and WebGL renderers are tested for support in that order, and the first supported used. Specify a ol.RendererType here to use a specific renderer. Note that at present only the Canvas renderer supports vector data.
target	Element string undefined	The container for the map, either the element itself or the id of the element. If not specified at construction time, ol.Map#setTarget must be called for the map to be rendered.
view	ol.View undefined	The map's view. No layer sources will be fetched unless this is specified at construction time or through ol.Map#setView .

controls

```
var map = new ol.Map({
  target: 'map', // The DOM element that will contains the map
  controls: [
    new ol.control.ZoomSlider(),
    new ol.control.MousePosition(),
    new ol.control.Zoom()
  ],
  layers: [
    // Add a new Tile layer getting tiles from OpenStreetMap
    source
    new ol.layer.Tile({
      source: new ol.source.OSM()
    })
  ],
  // Create a view centered on the specified location and zoom
  level
  view: new ol.View({
    center: ol.proj.transform([ 20.492420, 44.798095],
    'EPSG:4326', 'EPSG:3857'),
    zoom: 10
  })
});
```

controls



ol.Map.addControl

```
var map = new ol.Map({
  target: 'map', // The DOM element that will contains the map
  layers: [
    // Add a new Tile layer getting tiles from OpenStreetMap
    new ol.layer.Tile({
      source: new ol.source.OSM()
    })
  ],
  // Create a view centered on the specified location and zoom
  view: new ol.View({
    center: ol.proj.transform([ 20.492420, 44.798095],
    'EPSG:4326', 'EPSG:3857'),
    zoom: 10
  })
});
zoomslider = new ol.control.ZoomSlider();
map.addControl(zoomslider);
```

Ol.Map неке методе

```
var size = map.getSize(); // [x,y] pixel size  
var view = map.getView();
```

Слично:

```
getTarget, setTarget, getView, setView,  
    getSize, setSize, getLayerGroup,  
    setLayerGroup
```

```
var group = map.getLayerGroup();  
// ol.Layer.Group instance  
var layers = map.getLayers();  
// [LayerA, LayerB, ...]
```

ol.Map методе

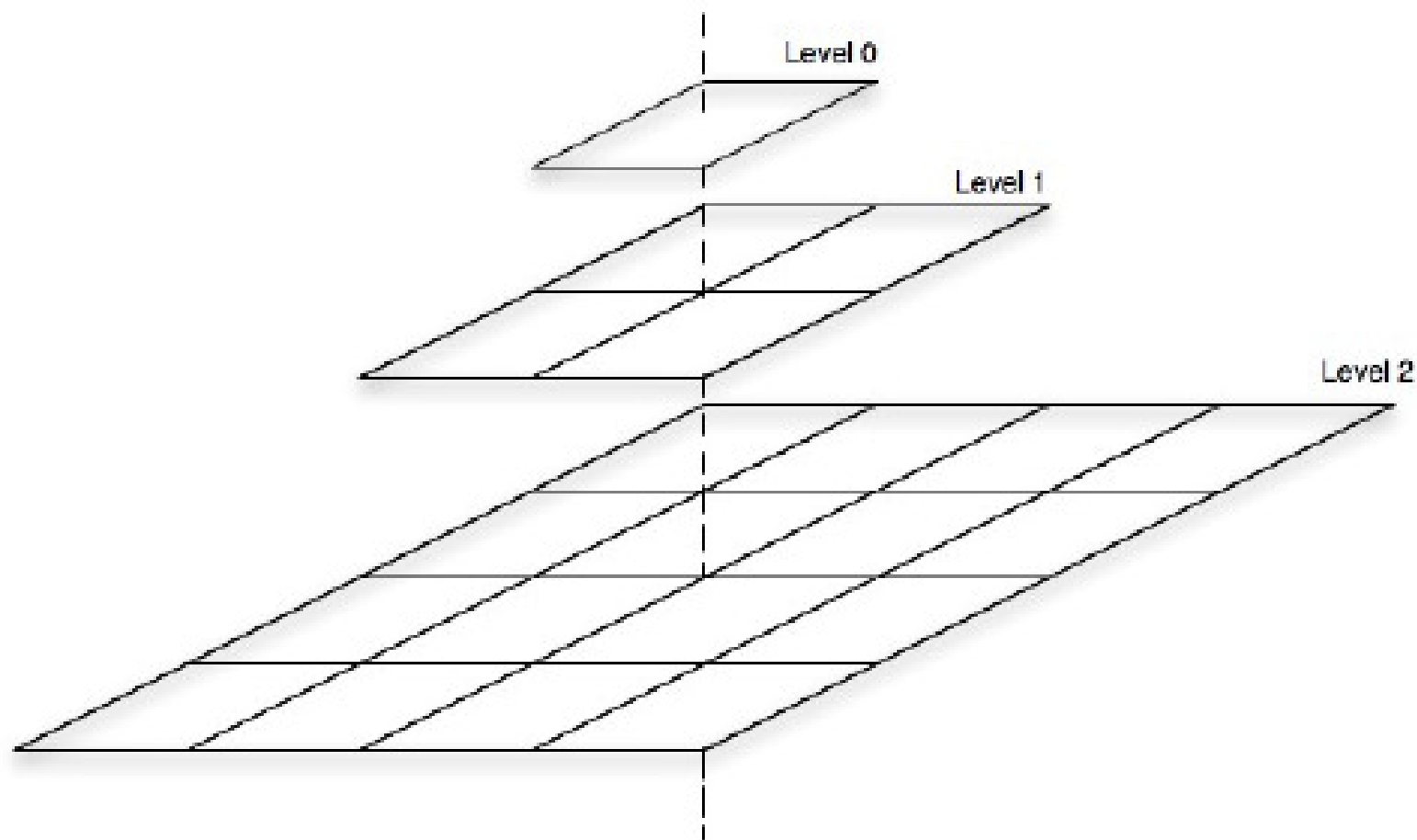
addControl, addInteraction, addLayer,
addOverlay, forEachFeatureAtPixel,
getControls, getCoordinateFromPixel,
getEventCoordinate, getEventPixel,
getInteractions, getLayerGroup, getLayers,
getOverlays, getPixelFromCoordinate, getSize,
getTarget, getView, getViewport, on, once,
removeControl, removeInteraction,
removeLayer, removeOverlay, render,
renderSync, setLayerGroup, setTarget,
setView, un, unByKey, updateSize

<http://openlayers.org/en/v3.0.0/apidoc/ol.Map.html>

ol.View

Name	Type	Description
center	ol.Coordinate undefined	The initial center for the view. The coordinate system for the center is specified with the projection option. Default is undefined, and layer sources will not be fetched if this is not set.
maxResolution	number undefined	The maximum resolution used to determine the resolution constraint. It is used together with minResolution (or maxZoom) and zoomFactor. If unspecified it is calculated in such a way that the projection's validity extent fits in a 256x256 px tile. If the projection is Spherical Mercator (the default) then maxResolution defaults to $40075016.68557849 / 256 = 156543.03392804097$.
minResolution	number undefined	The minimum resolution used to determine the resolution constraint. It is used together with maxResolution (or minZoom) and zoomFactor. If unspecified it is calculated assuming 29 zoom levels (with a factor of 2). If the projection is Spherical Mercator (the default) then minResolution defaults to $40075016.68557849 / 256 / \text{Math.pow}(2, 28) = 0.0005831682455839253$.
maxZoom	number undefined	The maximum zoom level used to determine the resolution constraint. It is used together with minZoom (or maxResolution) and zoomFactor. Default is 28. Note that if minResolution is also provided, it is given precedence over maxZoom.
minZoom	number undefined	The minimum zoom level used to determine the resolution constraint. It is used together with maxZoom (or minResolution) and zoomFactor. Default is 0. Note that if maxResolution is also provided, it is given precedence over minZoom.
projection	ol.proj.ProjectionLike	The projection. Default is EPSG:3857 (Spherical Mercator).
resolution	number undefined	The initial resolution for the view. The units are projection units per pixel (e.g. meters per pixel). An alternative to setting this is to set zoom. Default is undefined, and layer sources will not be fetched if neither this nor zoom are defined.
resolutions	Array.<number> undefined	Resolutions to determine the resolution constraint. If set the maxResolution, minResolution, minZoom, maxZoom, and zoomFactor options are ignored.
rotation	number undefined	The initial rotation for the view in radians (positive rotation clockwise). Default is 0.
zoom	number undefined	Only used if resolution is not defined. Zoom level used to calculate the initial resolution for the view. The initial resolution is determined using the <code>ol.View#constrainResolution</code> method.
zoomFactor	number undefined	The zoom factor used to determine the resolution constraint. Default is 2.

Нивои увећања



Zoom levels

ol.View резолуција и зоом

Цео Свет у једном тајлу од 256 пиксела. Цео Екватор се приказује у 256 пиксела.

$$6378137 * 2 * \pi / 256 = 156543.034 \text{ meters/pixel}$$

За зоом 1 са 2 тајла је приказан цео Свет.

78271.52 метара одговара једном пикселу

...

zoom	resolution, m/px
0	156543.03
1	78271.52
2	39135.76
3	19567.88
4	9783.94
5	4891.97
6	2445.98
7	1222.99
8	611.50
9	305.75
10	152.87
11	76.437
12	38.219
13	19.109
14	9.5546
15	4.7773
16	2.3887
17	1.1943
18	0.5972

setResolution, setZoom

```
view.setResolution(12000);
```

```
view.setZoom(7);
```

ol.View методе

getCenter

getProjection

getResolution

getRotation

getZoom

on

once

rotate

setCenter

setResolution

setRotation

setZoom

un

unByKey

OpenLayers 3 example

← → ↻

file:///D:/WEB%20kartografija%20predavanja/OL3/m4.html?

☆ ☰

Latitude:

44.79809499999999

Longitude:

20.492419999999996

Change

Degrees:

0

(OpenLayers3 requires you transform to radians)

Change

Level:

11

Change

+

-

Батајница

Летиште 13. ма

Аеродром Београд Никола Тесла

Београд

Панчево

Аеродром (Панчево)

13

10

10

26

12

13

6

8

9

9

1

1

2

2

A1

A3

A1

A3

A1

A3

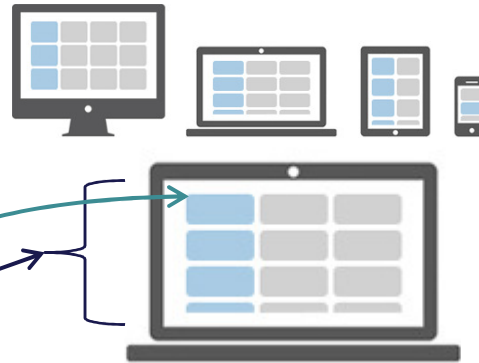
Openlayers, bootstrap, jquery

```
<!doctype html>
<html lang="en">
  <head>
    <link rel="stylesheet" href="http://openlayers.org/en/v3.0.0/css/ol.css"
type="text/css">
    <!-- bootswatch yeti theme -->
    <link href="bootYeti.css" rel="stylesheet">

  <style>
    body { font-family:arial,sans-serif !important; }
  </style>
  <script src="v3.0.0/build/ol.js" type="text/javascript"></script>
  <script src="jquery-2.1.1.min.js" type="text/javascript"></script>

  <title>OpenLayers 3 example</title>
</head>
<body>
  <h2>Web kartografija - GRF</h2>
  <p>Demonstrates how we can programmatically modify the view properties to change
center location, rotation or zoom level:</p>
```

bootstrap



Latitude:

Longitude:

```

<div class="row">
  <div class="col-md-4">
    <form role="form">
      <div class="form-group">
        <label for="lat">Latitude:</label>
        <input type="text" class="form-control" id="lat"
placeholder="longitude" value="0.0">
      </div>
      <div class="form-group">
        <label for="lon">Longitude:</label>
        <input type="text" class="form-control" id="lon"
placeholder="latitude" value="0.0">
      </div>
      <button type="button" class="btn btn-primary btn-xs"
id="changeCenter">Change</button>
    </form>
  </div>

```

Features Bootstrap 3 Grid System	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
Max container width	None (auto)	750px	970px	1170px
Class prefix	.col-xs-	.col-sm-	.col-md-	.col-lg-

```

        <div class="col-md-4">
            <form role="form">
                <div class="form-group">
                    <label for="angle">Degrees:</label>
                    <input type="text" class="form-control" id="angle"
placeholder="rotation angle" value="0.0">
                    <span class="help-block">(OpenLayers3 requires you transform to
radians)</span>
                </div>
                <button type="button" class="btn btn-primary btn-xs"
id="changeRotation">Change</button>
            </form>
        </div>

        <div class="col-md-4">
            <form role="form">
                <div class="form-group">
                    <label for="level">Level:</label>
                    <input type="text" class="form-control" id="level"
placeholder="zoom level" value="7">
                </div>
                <button type="button" class="btn btn-primary btn-xs"
id="changeZoom">Change</button>
            </form>
        </div>
    </div>
</div>
<div id="map" class="map"></div>

```

Openlayers

```
<script id="code">
    var map = new ol.Map({
        target: 'map', // The DOM element that will contains the map
        renderer: 'canvas', // Force the renderer to be used
        layers: [
            // Add a new Tile layer getting tiles from OpenStreetMap
source
            new ol.layer.Tile({
                source: new ol.source.OSM()
            })
        ],
        // Create a view centered on the specified location and zoom
level
        view: new ol.View({
            center: ol.proj.transform([20.492420, 44.798095],
'EPSG:4326', 'EPSG:3857'),
            zoom: 11
        })
    });
```

Openlayers, jquery

```
$(document).ready(function() {  
    // Once the document is ready retrieve the initial map view center  
    // location and set in the input fields.  
    var center = ol.proj.transform(map.getView().getCenter(), 'EPSG:3857',  
'EPSG:4326');  
    $('#lon').val(center[0]);  
    $('#lat').val(center[1]);  
    $('#angle').val(map.getView().getRotation());  
    $('#level').val(map.getView().getZoom());  
    // For each button click retrieve the values and apply on the  
    // map view.  
    $('#changeCenter').on('click', function() {  
        var center = [parseFloat($('#lon').val()),parseFloat($('#lat').val())];  
        map.getView().setCenter(ol.proj.transform(center, 'EPSG:4326',  
'EPSG:3857'));  
    });  
    $('#changeRotation').on('click', function() {  
        map.getView().setRotation($('#angle').val() * Math.PI / 180);  
    });  
    $('#changeZoom').on('click', function() {  
        map.getView().setZoom($('#level').val());  
    });  
});  
</script>  
  
</body>  
</html>
```

Литература

<http://openlayers.org/dev/examples>

<http://docs.openlayers.org/>

