Chapter – I

Introduction to Leaflet

Building maps using Leaflet.js

Leaflet is an open-source JavaScript library used create interactive, mobile friendly web maps. It gives assists to Web-GIS developers in building web mapping applications.

This tutorial will help you how to create an easy and lightweight map using the leaflet.js library.

To build simple map requires below mentioned things:

- HTML (HyperText Markup Language)
- CSS (Cascading Style Sheet)
- JavaScript
- Code editors Sublime Text, Notepad ++, and Visual Studio code
- XAMMP is a web server helps to a local host or server to test website

Install XAMMP server

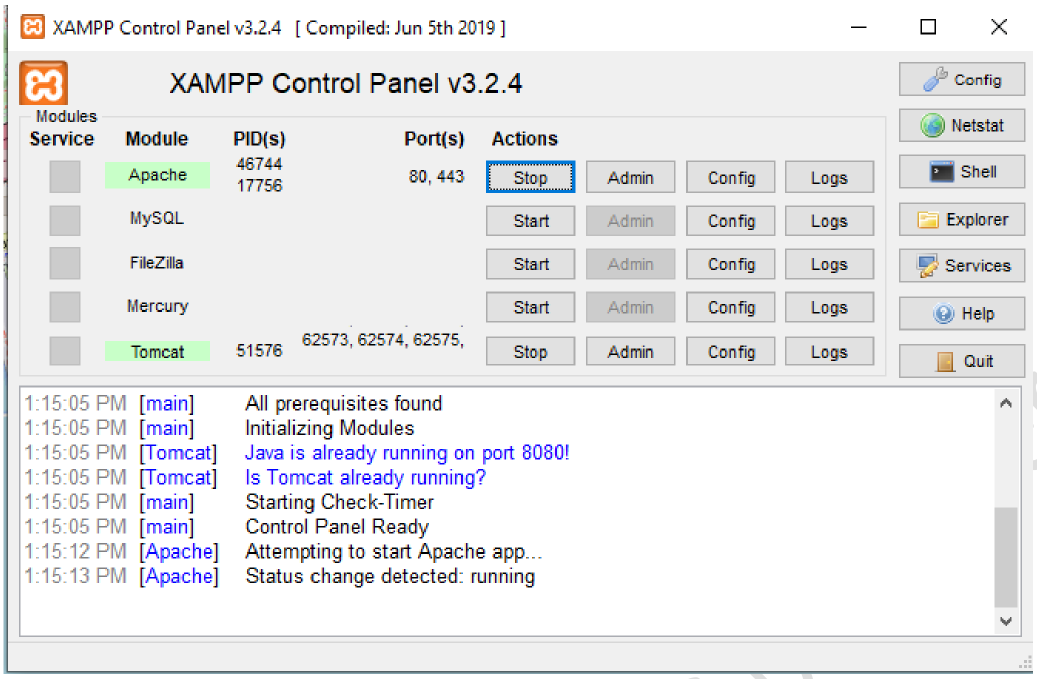
Step 1: Download xammp server and install

Step 2: After installed, go to windows C drive you can find xammp folder.

Windows (C) → xammp → htdocs → create a new folder 'Web_GIS'

Step 3: Inside project folder create index.html file

Step 4: Start Xammp server


XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]
XAMPP Control Panel v3.2.4
Modules
Config
Netstat
Shell
Explorer
Services
Help
Quit

Service	Module	PID(s)	Port(s)	Actions			
	Apache	46744 17756	80, 443	Stop	Admin	Config	Logs
	MySQL			Start	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat	51576	62573, 62574, 62575,	Stop	Admin	Config	Logs

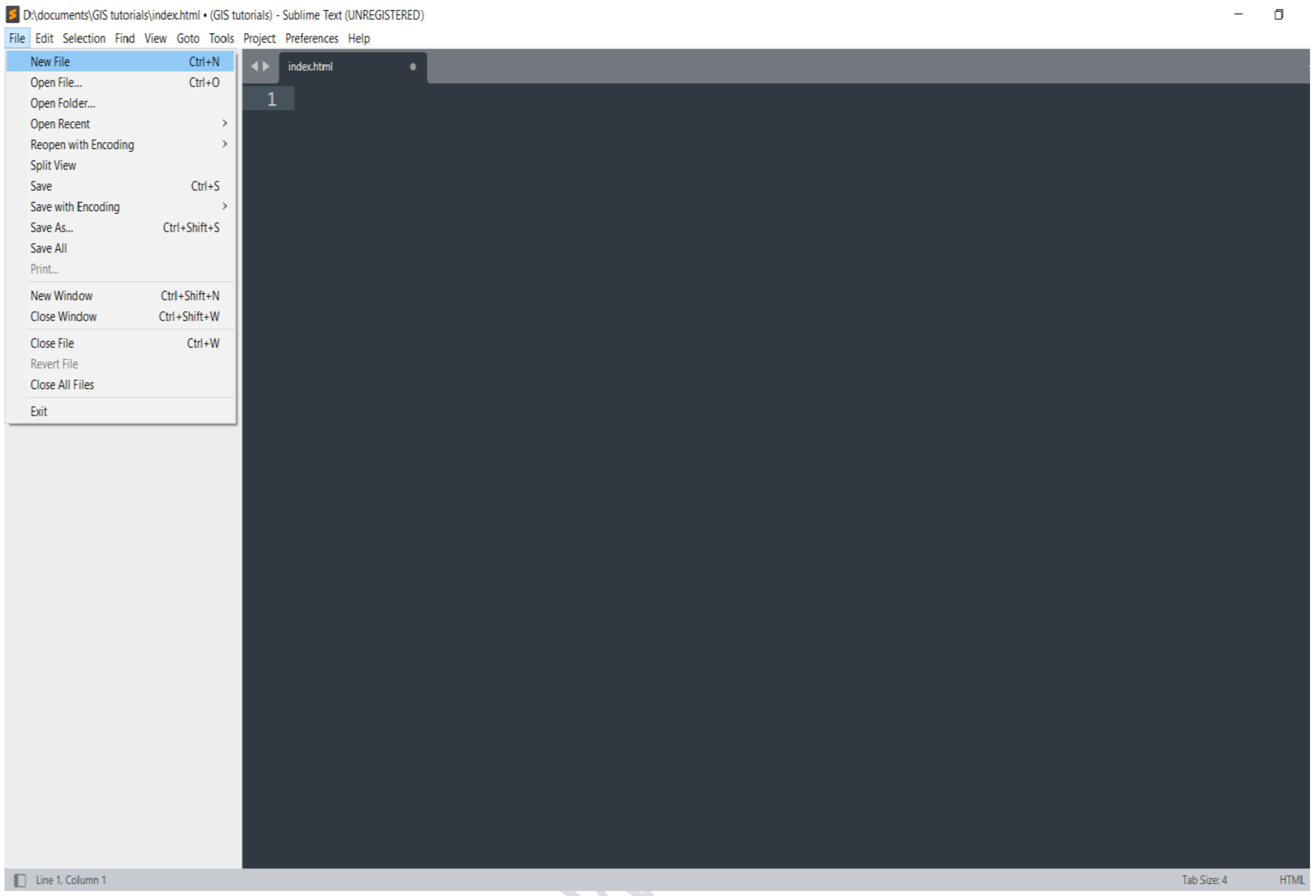
```
1:15:05 PM  [main]     All prerequisites found
1:15:05 PM  [main]     Initializing Modules
1:15:05 PM  [Tomcat]   Java is already running on port 8080!
1:15:05 PM  [Tomcat]   Is Tomcat already running?
1:15:05 PM  [main]     Starting Check-Timer
1:15:05 PM  [main]     Control Panel Ready
1:15:12 PM  [Apache]   Attempting to start Apache app...
1:15:13 PM  [Apache]   Status change detected: running
```

Create a basic HTML page

In this tutorial, we will use the Sublime text code editor.

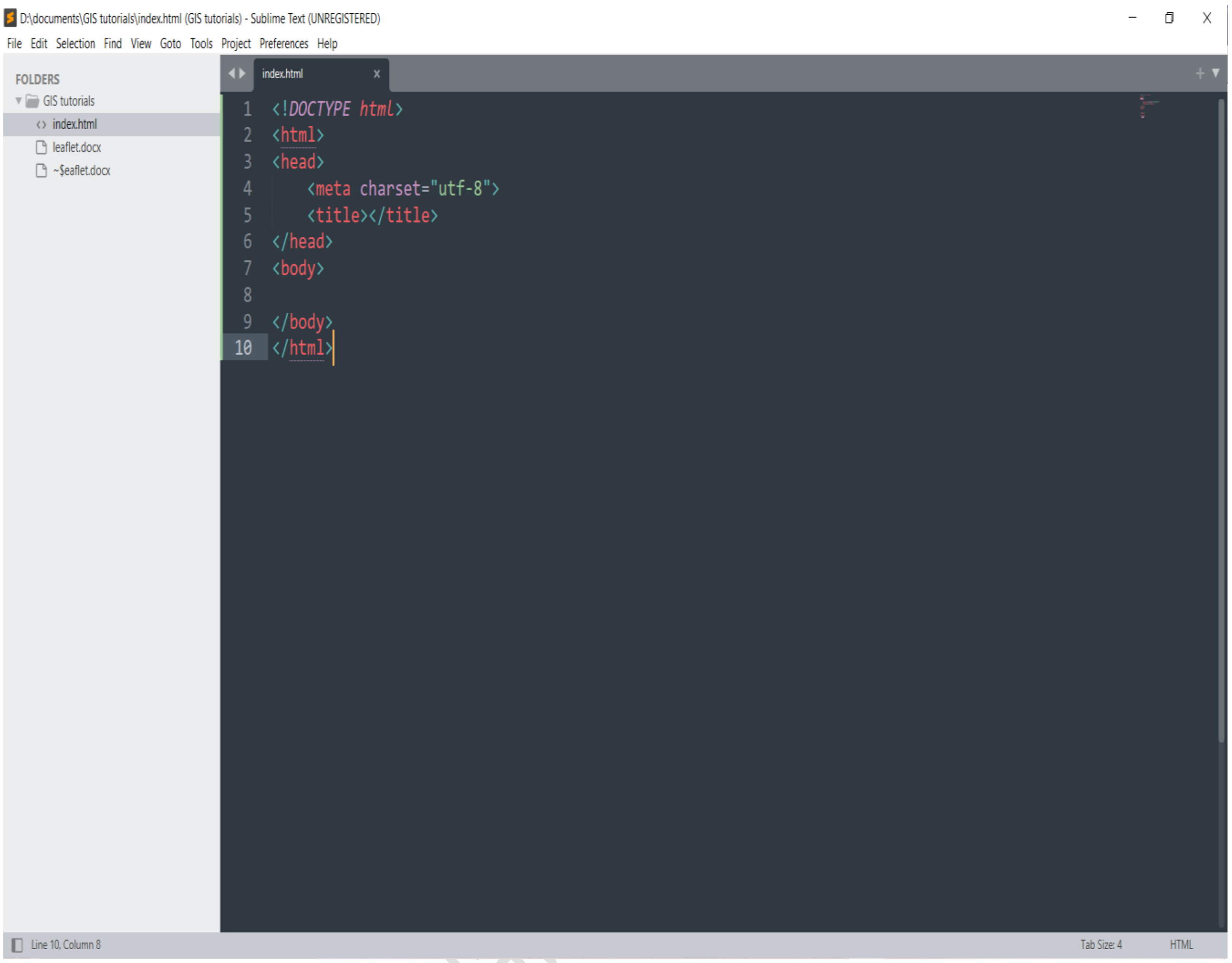
Open a sublime text editor → File → New → Save the file using a index.html extension.

;



Next, create a basic html page by adding the below html code.

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="utf-8">
        <title></title>
</head>
<body>
</body>
</html>
```


D:\documents\GIS tutorials\index.html (GIS tutorials) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
GIS tutorials
index.html
leaflet.docx
~$eaflet.docx
index.html
Line 10, Column 8
Tab Size: 4
HTML

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title></title>
</head>
<body>

</body>
</html>
```

Loading a remote script

Step 1: now include Leaflet CSS file in the head section of your index.html file

<link rel="stylesheet" href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css"/>

Step 2: Include Leaflet JavaScript CDN file after CSS CDN

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>

Loading a local script

When loading a script from a local file, we need to have an actual path of the file on your machine

<script src="leaflet.js"></script>

Create the container (Add map div)

To hold the map, we have to create a container element. Let's create a <div> element within the body to contain the map. Create a <div> element with the id of "map" to contain the map by adding the following to the HTML <body>:

<div id="map" style = "width:800px; height:550px;"></div>

Now create a map height, the <div> id element should have a defined height. In the head section, we create a style tag with the div id's name is map. Add below CSS style code in the head section with style tag.

```
<style>

#map {

      width: 100%;

      height: 100vh;

}

</style>
```

Initialize map

```
  var map = L.map('map',{

  center: [20.59, 78.96],

  zoom: 15

  });
```

Center option is specified in coordinates using an array of form [latitude, longitude] in the form of [Y, X] rather in the [X, Y]

Adding a tile layer

To add a tile layer to a leaflet web map, we use the L. tileLayer function. This function accepts:

- The URL of the tile layer
- An object with additional options

```
L.tileLayer(

  "https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png",

  {attribution: "&copy; OpenStreetMap"}

).addTo(map);
```

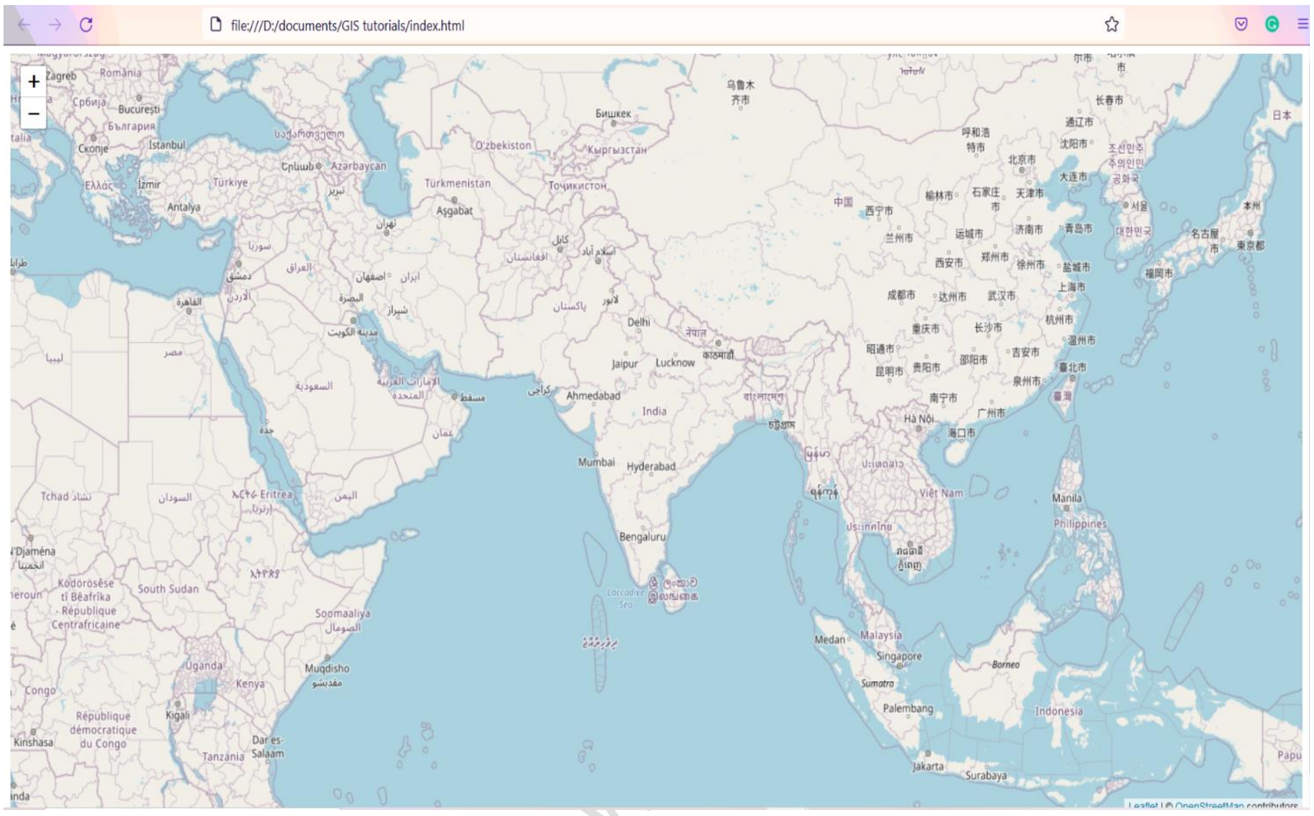
Raster tile server URL includes the {z}, {x}, and {y} placeholders, which are internally replaced by column, row and zoom levels each tile the Leaflet library loads a given tile.

The code is now complete so the file should now look like this:

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title>Document</title>
  <link rel="stylesheet" href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css"/>
  <script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
  <style>
    #map {
      width: 100%;
      height: 100vh;
    }
  </style>
</head>
<body>
  <div id="map"></div>
  <script>
  var map = L.map('map',{
  center: [20.59, 78.96],
  zoom: 4
  });
 var osm =  L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
  attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a>
contributors'
  }).addTo(map);
</script>
</body>
</html>
```

If you run the above HTML code in your browser, leaflet map like the image shown below

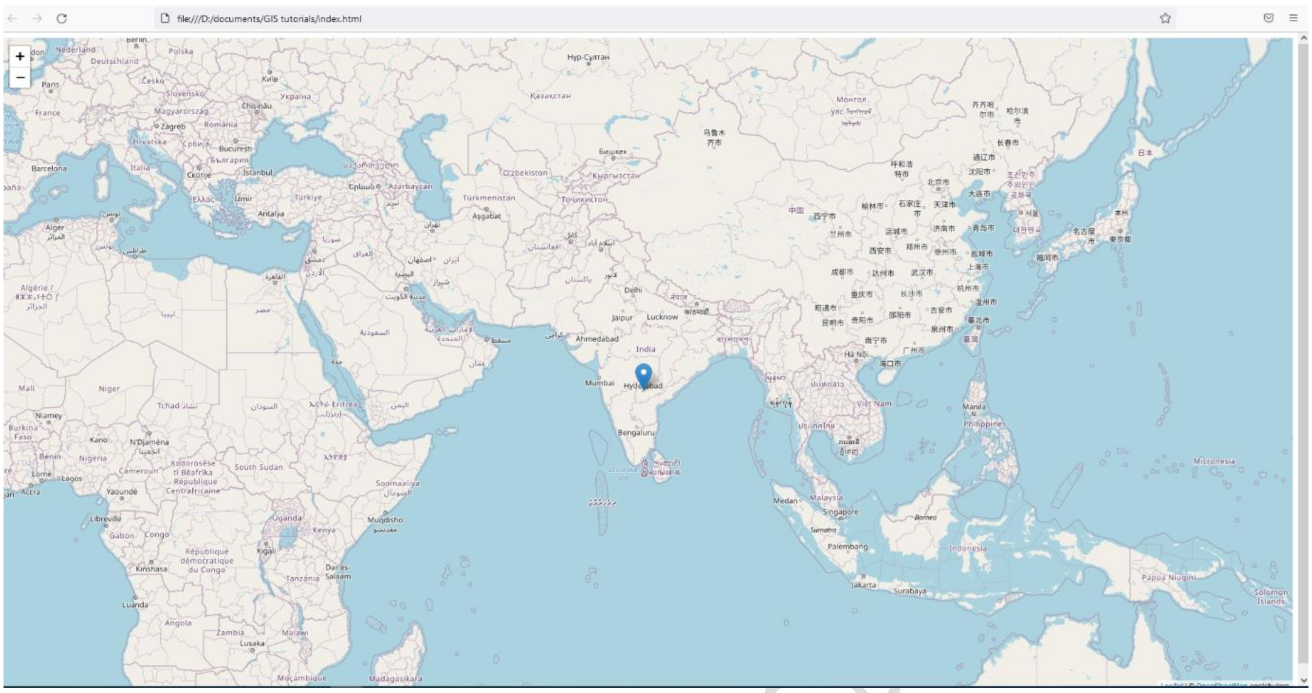
Open browser and type this url http://localhost/web_gis_tutorial/ → "URL/folder_name"


file:///D:/documents/GIS tutorials/index.html
Leaflet | © OpenStreetMap contributors

Chapter II - Add Overlayers

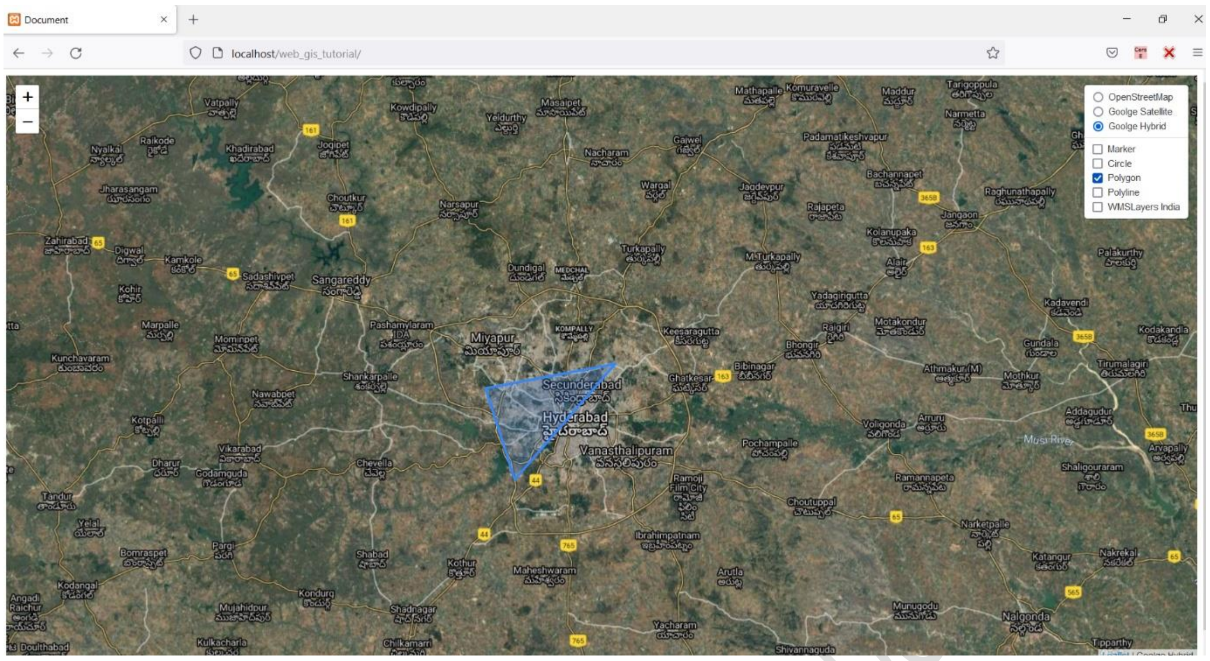
Add a marker

```
var marker = L.marker([17.3850, 78.4867]).addTo(map);
```


file:///D:/documents/GIS tutorials/index.html

Add Polygon

```
var polygonPoints = [
    [17.293129518, 78.381988260],
    [17.484305856, 78.556315685],
    [17.484305743, 78.55321569],
    [17.443212156, 78.33152321]];
var poly = L.polygon(polygonPoints).addTo(map);
```

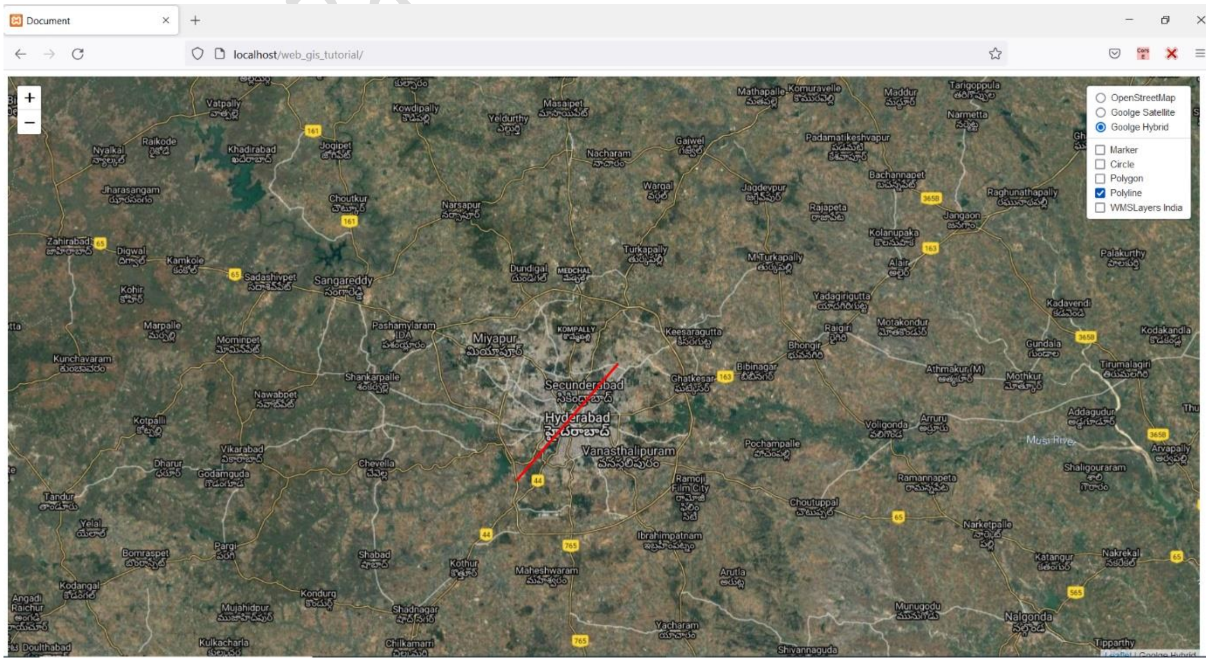

Document

localhost/web_gis_tutorial/

- [ ] OpenStreetMap
- [ ] Goolge Satellite
- [✓ Goolge Hybrid]
- [ ] Marker
- [ ] Circle
- [✓] Polygon
- [ ] Polyline
- [ ] WMSLayers India

Adding Polyline

```
var polyline = [
   [17.293129518, 78.381988260],
   [17.484305856, 78.556315685]];
var polyli = L.polyline(polyline, {color:'red', opacity:1}).addTo(map);
```

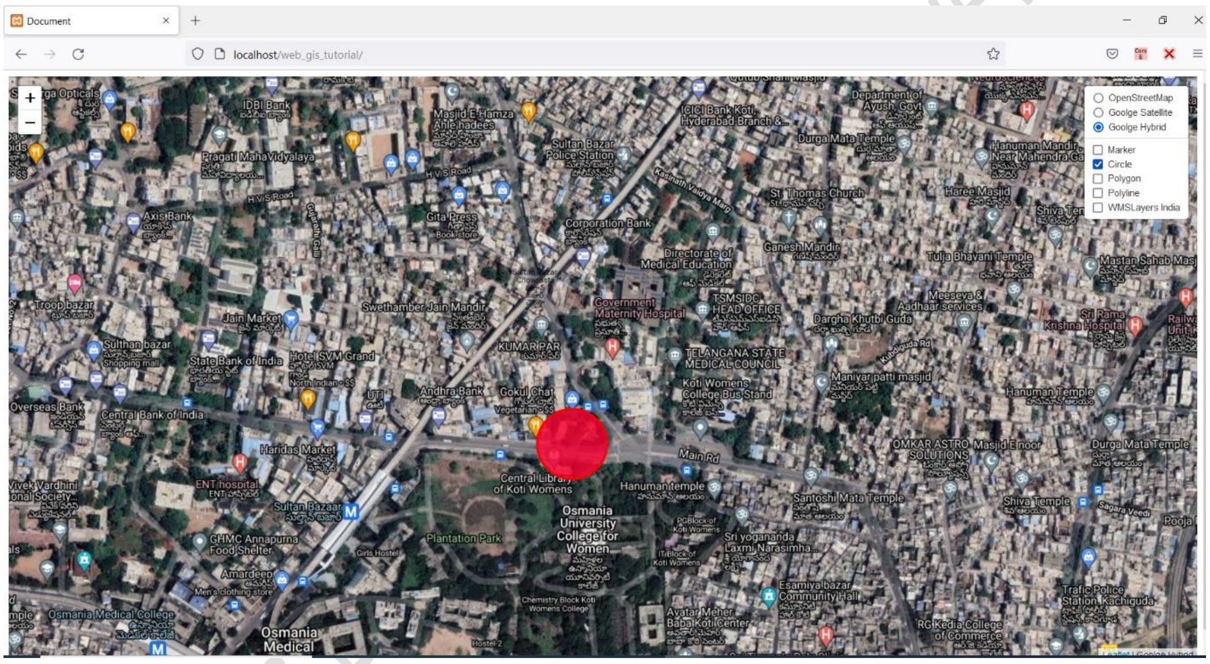

Document

localhost/web_gis_tutorial/

- [ ] OpenStreetMap
- [ ] Goolge Satellite
- [✓ Goolge Hybrid]
- [ ] Marker
- [ ] Circle
- [ ] Polygon
- [✓] Polyline
- [ ] WMSLayers India

Add Circle

```
var circle = L.circle([17.3850, 78.4867], {

color: "red",

fillColor: "#f03",

fillOpacity: 0.8,

radius: 50.0

}).addTo(map);
```



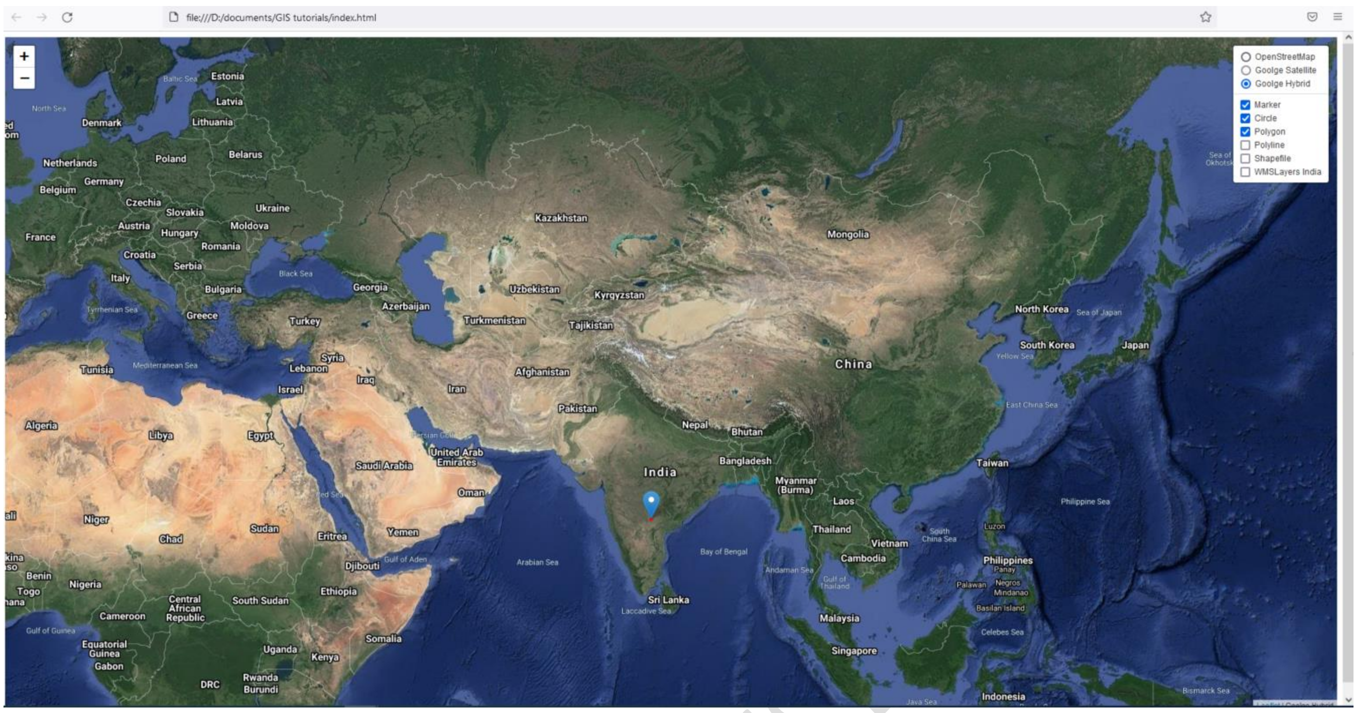
[ ] OpenStreetMap
[ ] Goolge Satellite
[✓ Goolge Hybrid]
[ ] Marker
[✓] Circle
[ ] Polygon
[ ] Polyline
[ ] WMSLayers India

Chapter III - Map Control Panel

Adding layer control panel

In this section, we will learn how to add a layer controller panel to the map. It is gives you a choice to select which layer you would like to view on map. To get this, first create a base map variables like osm, gsatellite,gHybird and another variable names for overlay maps, but anyway we already created overlay maps. Now we create a variable name called Basemaps and Overlyamaps.

```
var osm = L.tileLayer('http://tile.openstreetmap.org/{z}/{x}/{y}.png', {
  attribution: '&copy; <a
href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'
  })
  var gsatellite = L.tileLayer('http://mt0.google.com/vt/lyrs=s&hl=en&x={x}&y={y}&z={z}',{
  attribution:'Googel Satellite Imagery'
  })
  var gHybrid = L.tileLayer('https://mt1.google.com/vt/lyrs=y&x={x}&y={y}&z={z}',{
  attribution:'Goolge Hybrid'
  }).addTo(map)
var Basemaps = {
  "OpenStreetMap": osm,
  "Goolge Satellite": gsatellite,
  "Goolge Hybrid": gHybrid
}
var Overlayers = {
  "Marker": marker,
  "Circle": circle,
  "Polygon": poly
}
L.control.layers(Basemaps, Overlayers).addTo(map);
```

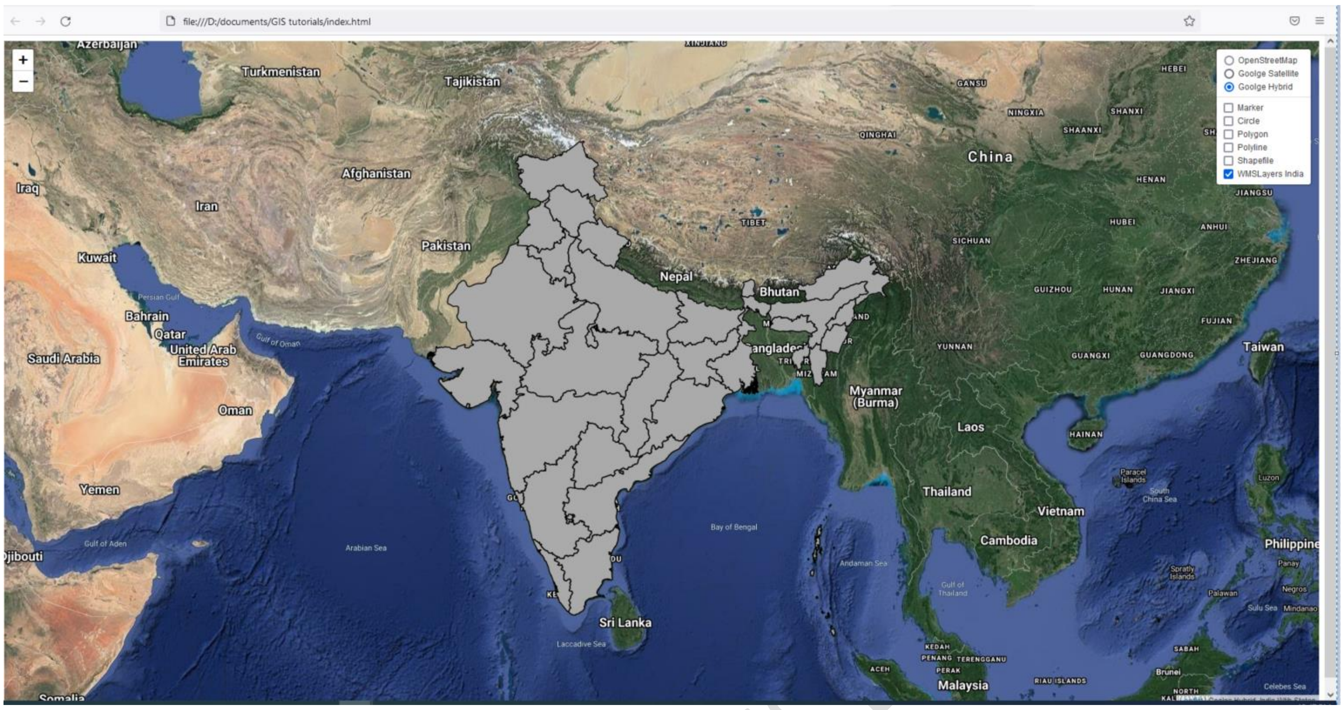


OpenStreetMap
Goolge Satellite
[✓ Goolge Hybrid]
[✓ Marker]
[✓ Circle]
[✓ Polygon]
Polyline
Shapefile
WMSLayers India

Adding WMS layer from geoserver

Used to display WMS layer as tile layers on the leaflet map. Using using L.tileLayer.wms() displays a WMS service as a tile layer on the map. Like the L.tilelayer() function, L.tilelayer.wms() is passed the url to the service provider and a set of options.

```
var ind = L.tileLayer.wms("http://localhost:8080/geoserver/gis_tutorials/wms",{

layers: 'gis_tutorials:India_WIth_States_Boundary',

format: 'image/png',

transparent: true,

attribution: "India Wtih States"

})
```


file:///D:/documents/GIS tutorials/index.html

( ) OpenStreetMap
( ) Goolge Satellite
(●) Goolge Hybrid

☐ Marker
☐ Circle
☐ Polygon
☐ Polyline
☐ Shapefile
☑ WMSLayers India

Adding geojson

GeoJSON is a very popular data format among many GIS technologies and services — it's simple, lightweight, straightforward, and Leaflet is quite good at handling it. (Source leaflet).

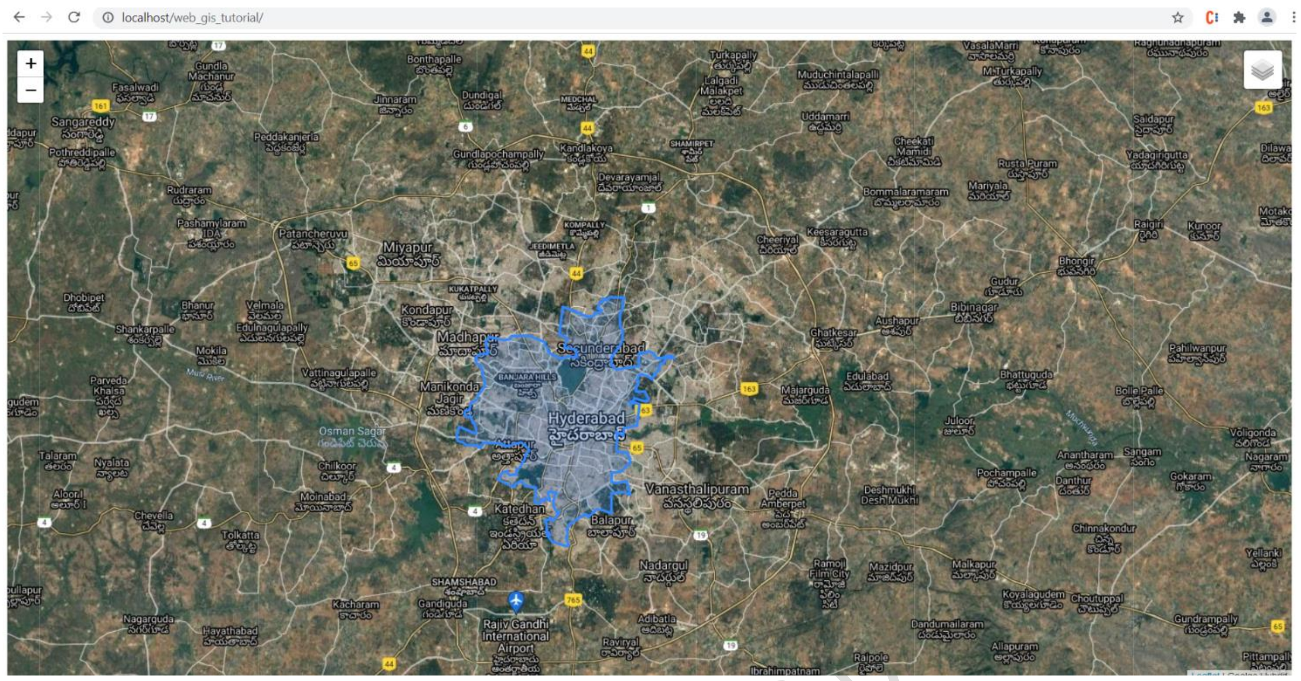
To do this we will use JQuery function getJSON(). Here we are supposed to loaded the JQuery library into the HTML document. Now this can be helps to load our external Geojson file.

Step 1: Export shapefile to geojson format in QGIS

Step 2: Save geojson file in your web gis project directory

Note: If you have geojson file keep in project directory// otherwise add your path address of geojson file.

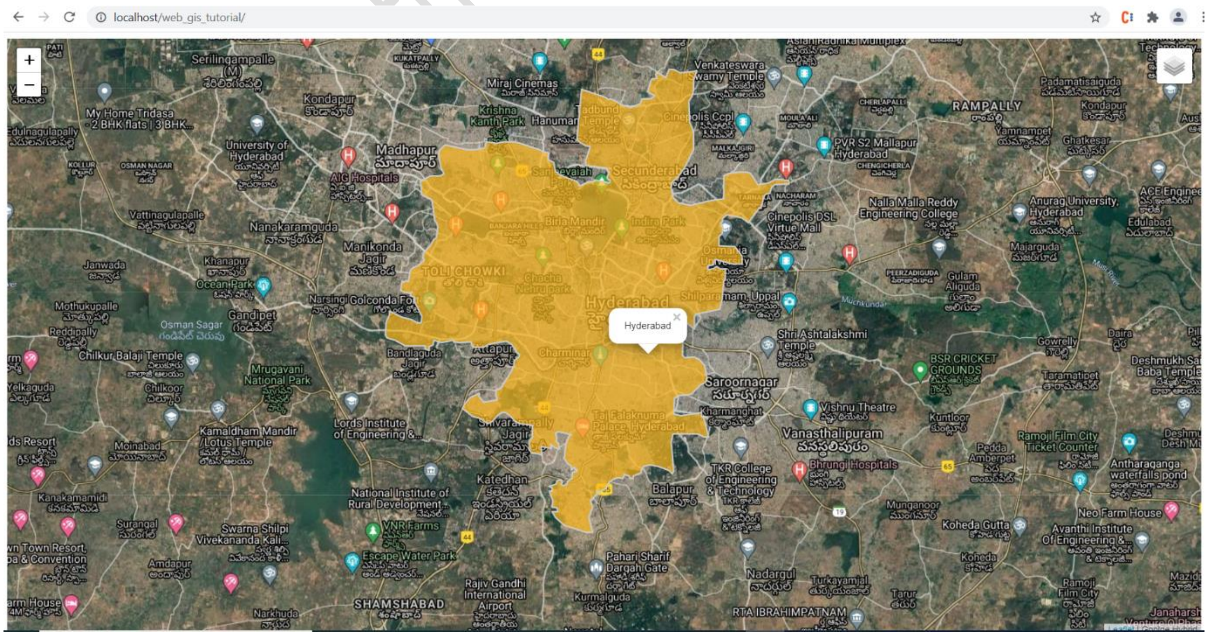
```
$.getJSON("hyderabad.geojson", function(data){

   L.geoJson(data).addTo(map);

})
```



Applying Popup and Style to geojson layer

Popups can be interactively opened and closed to get more information about the various features. When the user clicks on vector feature an information box shown on map. This is similar to the identify tool in QGIS software.

Style function that styles individual features based on their properties. This is similar to the symbology in QGI software.



```
function style(feature) {
 return {
   fillColor: "#FCB81E",
   weight: 2,
   opacity: 1,
   color: "#CCCCCC",
   fillOpacity: 0.7
 };
}
function onEachFeature(feature, layer) {
   // does this feature have a property named popupContent?
   if (feature.properties && feature.properties.Dist_Name) {
       layer.bindPopup(feature.properties.Dist_Name);
 }
}
$.getJSON("hyderabad.geojson", function(data){
   L.geoJson(data, {
  style:style,
       onEachFeature: onEachFeature
   }).addTo(map);
})
```

Finally, we have our final code as follows:

Final code attached in email