**Leaflet JS**

Contents

[Introduction 2](#_Toc490236945)

[Features 2](#_Toc490236946)

[Leaflet Code Sample 2](#_Toc490236947)

[Create Map object 3](#_Toc490236948)

[Add Tile Layer 3](#_Toc490236949)

[Output 3](#_Toc490236950)

[Add Marker On Map 3](#_Toc490236951)

[Output/UI View 4](#_Toc490236952)

[Marker Options 4](#_Toc490236953)

[Popup 4](#_Toc490236954)

[Output 5](#_Toc490236955)

[Options 5](#_Toc490236956)

[Vector Layer 6](#_Toc490236957)

[Adding circle, polygon on map example 6](#_Toc490236958)

[Output/Map view 6](#_Toc490236959)

[Leaflet Events 7](#_Toc490236960)

[Kind of events 7](#_Toc490236961)

[Other Layers 8](#_Toc490236962)

[GeoJson Layer 8](#_Toc490236963)

[Choropleth Map 8](#_Toc490236964)

[JS Code 8](#_Toc490236965)

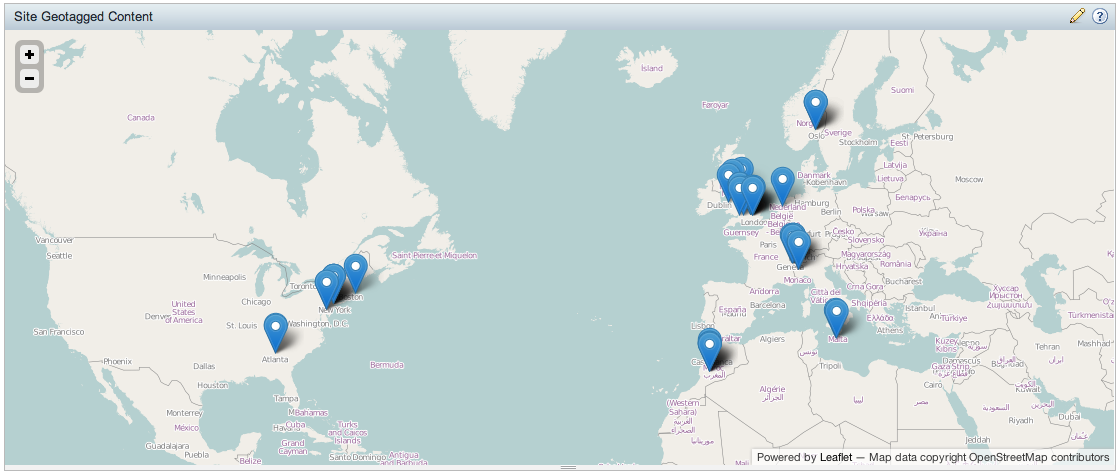
[Map View 9](#_Toc490236966)

[External Links 9](#_Toc490236967)

[Reference 9](#_Toc490236968)

# Introduction

* Leaflet is the leading open-source JavaScript library for mobile-friendly interactive maps.
* Weighing just about 34 KB of JS, it has all the mapping [features](http://leafletjs.com/#features) most developers ever need.
* Leaflet is designed with simplicity, performance and usability in mind.
* It works efficiently across all major desktop and mobile platforms, can be extended with lots of [plugins](http://leafletjs.com/plugins.html).



# Features

* Extremely lightweight — around 34 KB of gzipped JS code
* No external dependencies
* Tile layers
* Drag panning
* GeoJson Layers, Markers (custom markers support) , Popups , Pure CSS3 popups and controls
* Open Source
* Image Layers, Layer Groups, Scroll wheel zoom, Double click zoom, Zoom to area, Marker dragging
* Very nice default design for markers, popups and other map controls.

# Leaflet Code Sample

<html>

<head>

<link rel="stylesheet" href="http://cdn.leafletjs.com/leaflet-0.4/leaflet.css" />

<script src="http://cdn.leafletjs.com/leaflet-0.4/leaflet.js"></script>

<style type="text/css">

   #map {height: 180px; }

</style>

</head>

<body>

<div id="map">

</div>

<script type="text/javascript"></script>

</body>

</html>

## Create Map object

var map = L.map('map').setView([21.7679, 78.871], 13);

## Add Tile Layer

L.tileLayer("http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png", {

attribution: "Map data © OpenStreetMap contributors"

}).addTo(map);

## Output

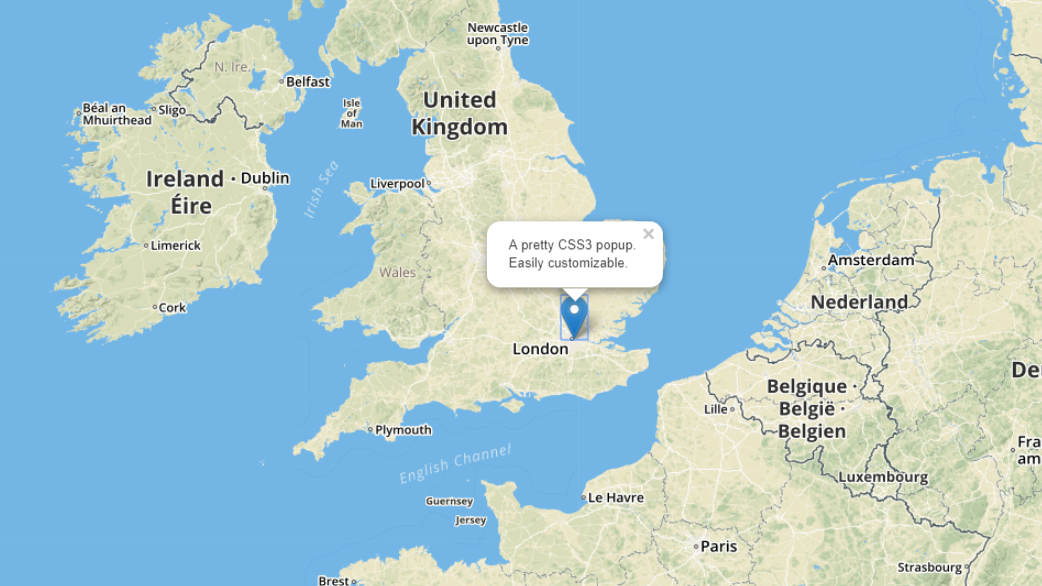


# Add Marker On Map

* L.Marker is used to display clickable/draggable icons on the map.

L.marker([51.5, -0.09]).addTo(map)  
 .bindPopup('A pretty CSS3 popup. <br> Easily customizable.').openPopup();

## Output/UI View



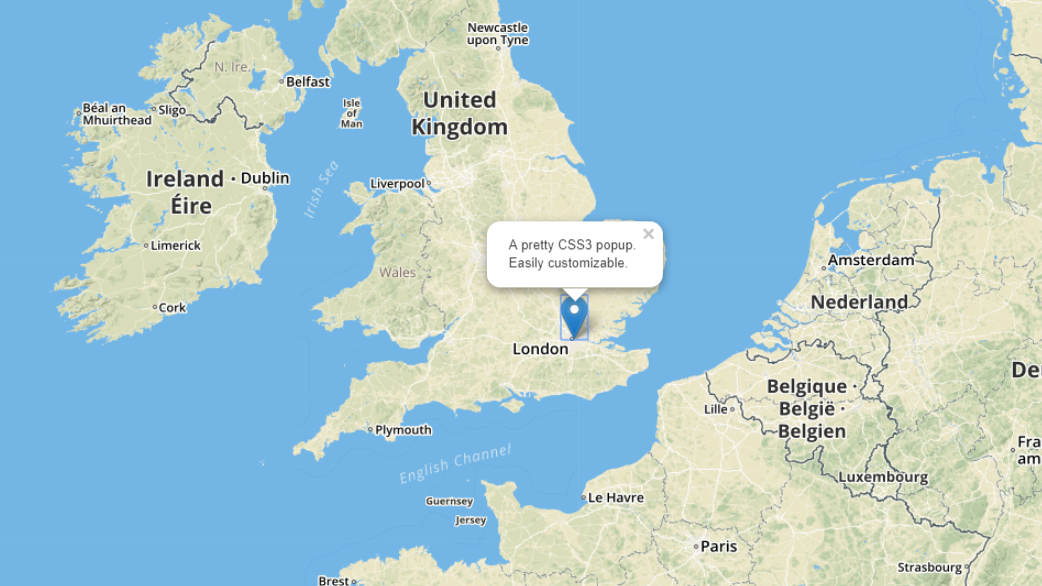
## Marker Options

|  |  |  |  |
| --- | --- | --- | --- |
| **Option** | **Type** | **Default** | **Description** |
| icon | [Icon](http://leafletjs.com/reference-1.2.0.html#icon) | \* | Icon instance to use for rendering the marker. See [Icon documentation](http://leafletjs.com/reference-1.2.0.html#icon) for details on how to customize the marker icon. If not specified, a common instance of [L.Icon.Default](http://leafletjs.com/reference-1.2.0.html" \l "icon-default) is used. |
| draggable | Boolean | false | Whether the marker is draggable with mouse/touch or not. |
| keyboard | Boolean | true | Whether the marker can be tabbed to with a keyboard and clicked by pressing enter. |
| title | String | '' | Text for the browser tooltip that appear on marker hover (no tooltip by default). |
| alt | String | '' | Text for the alt attribute of the icon image (useful for accessibility). |

# Popup

* Used to open popups in certain places of the map.
* Use [Map.openPopup](http://leafletjs.com/reference-1.2.0.html" \l "map-openpopup) to open popups while making sure that only one popup is open at one time (recommended for usability), or use [Map.addLayer](http://leafletjs.com/reference-1.2.0.html" \l "map-addlayer) to open as many as you want.
* If you want to just bind a popup to marker click and then open it, it's really easy:
* marker.bindPopup(popupContent).openPopup();
* Path overlays like polylines also have a bindPopup method. Here's a more complicated way to open a popup on a map:
* var popup = L.popup()
* .setLatLng(latlng)
* .setContent('A pretty CSS3 popup. <br> Easily customizable.')
* .openOn(map);

## Output



## Options

|  |  |  |  |
| --- | --- | --- | --- |
| **Option** | **Type** | **Default** | **Description** |
| maxWidth | Number | 300 | Max width of the popup, in pixels. |
| minWidth | Number | 50 | Min width of the popup, in pixels. |
| maxHeight | Number | null | If set, creates a scrollable container of the given height inside a popup if its content exceeds it. |
| autoPan | Boolean | true | Set it to false if you don't want the map to do panning animation to fit the opened popup. |
| autoPanPaddingTopLeft | [Point](http://leafletjs.com/reference-1.2.0.html#point) | null | The margin between the popup and the top left corner of the map view after autopanning was performed. |
| autoPanPaddingBottomRight | [Point](http://leafletjs.com/reference-1.2.0.html#point) | null | The margin between the popup and the bottom right corner of the map view after autopanning was performed. |
| autoPanPadding | [Point](http://leafletjs.com/reference-1.2.0.html#point) | Point(5, 5) | Equivalent of setting both top left and bottom right autopan padding to the same value. |
| keepInView | Boolean | false | Set it to true if you want to prevent users from panning the popup off of the screen while it is open. |
| closeButton | Boolean | true | Controls the presence of a close button in the popup. |
| autoClose | Boolean | true | Set it to false if you want to override the default behavior of the popup closing when another popup is opened. |

# Vector Layer

There are different kind of layer available in leaflet , are as follows-

* [Path](http://leafletjs.com/reference-1.2.0.html#path)
* [Polyline](http://leafletjs.com/reference-1.2.0.html#polyline)
* [Polygon](http://leafletjs.com/reference-1.2.0.html#polygon)
* [Rectangle](http://leafletjs.com/reference-1.2.0.html#rectangle)
* [Circle](http://leafletjs.com/reference-1.2.0.html#circle)
* [CircleMarker](http://leafletjs.com/reference-1.2.0.html#circlemarker)
* [SVG](http://leafletjs.com/reference-1.2.0.html#svg)
* [Canvas](http://leafletjs.com/reference-1.2.0.html#canvas)

## Adding circle, polygon on map example

var circle = L.circle([51.508, -0.11], {

color: 'red',

fillColor: '#f03',

fillOpacity: 0.5,

radius: 500

}).addTo(mymap);

var polygon = L.polygon([

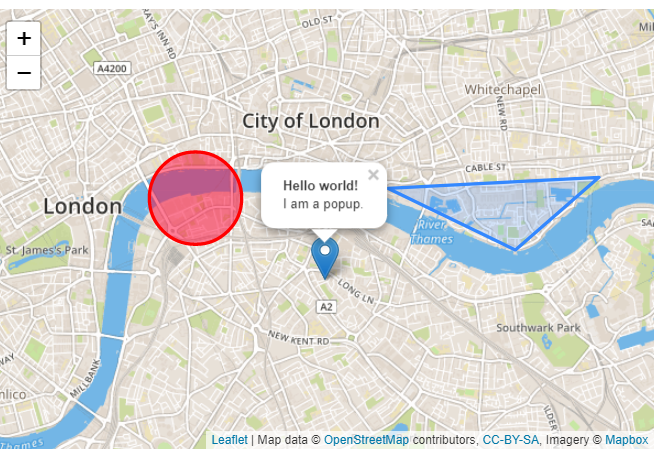
[51.509, -0.08],

[51.503, -0.06],

[51.51, -0.047]

]).addTo(mymap);

## Output/Map view



# Leaflet Events

* Every time something happens in Leaflet, e.g. user clicks on a marker or map zoom changes, the corresponding object sends an event which you can subscribe to with a function. It allows you to react to user interaction:

function onMapClick(e) {

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

}

map.on('click', onMapClick);

## Kind of events

* Layer Events (layerAdd, layerRemove etc)
* Map state change events (resize,load,unload,zoom,zoomstart,zoomend)
* Popup events (popupopen, popupclose)
* Tooltip events (tooltipopen,tooltipclose)
* Location events (locationerror, locationfound)
* Interaction events (click,dblclick,mousedown,mouseup,mousehover etc.)

# Other Layers

## GeoJson Layer

* Represents a GeoJSON object or an array of GeoJSON objects. Allows you to parse GeoJSON data and display it on the map. Extends [FeatureGroup](http://leafletjs.com/reference-1.2.0.html" \l "featuregroup).
* Usage Example -

L.geoJSON(data, {

style: function (feature) {

return {color: feature.properties.color};

}

}).bindPopup(function (layer) {

return layer.feature.properties.description;

}).addTo(map);

## Choropleth Map

### JS Code

* The GeoJSON with state shapes and all the related things are available in us-state.json and json data assigned to **statesData** JS variable.

function getColor(d) {

return d > 1000 ? '#800026' :

d > 500 ? '#BD0026' :

d > 200 ? '#E31A1C' :

d > 100 ? '#FC4E2A' :

d > 50 ? '#FD8D3C' :

d > 20 ? '#FEB24C' :

d > 10 ? '#FED976' :

'#FFEDA0';

}

function style(feature) {

return {

fillColor: getColor(feature.properties.density),

weight: 2,

opacity: 1,

color: 'white',

dashArray: '3',

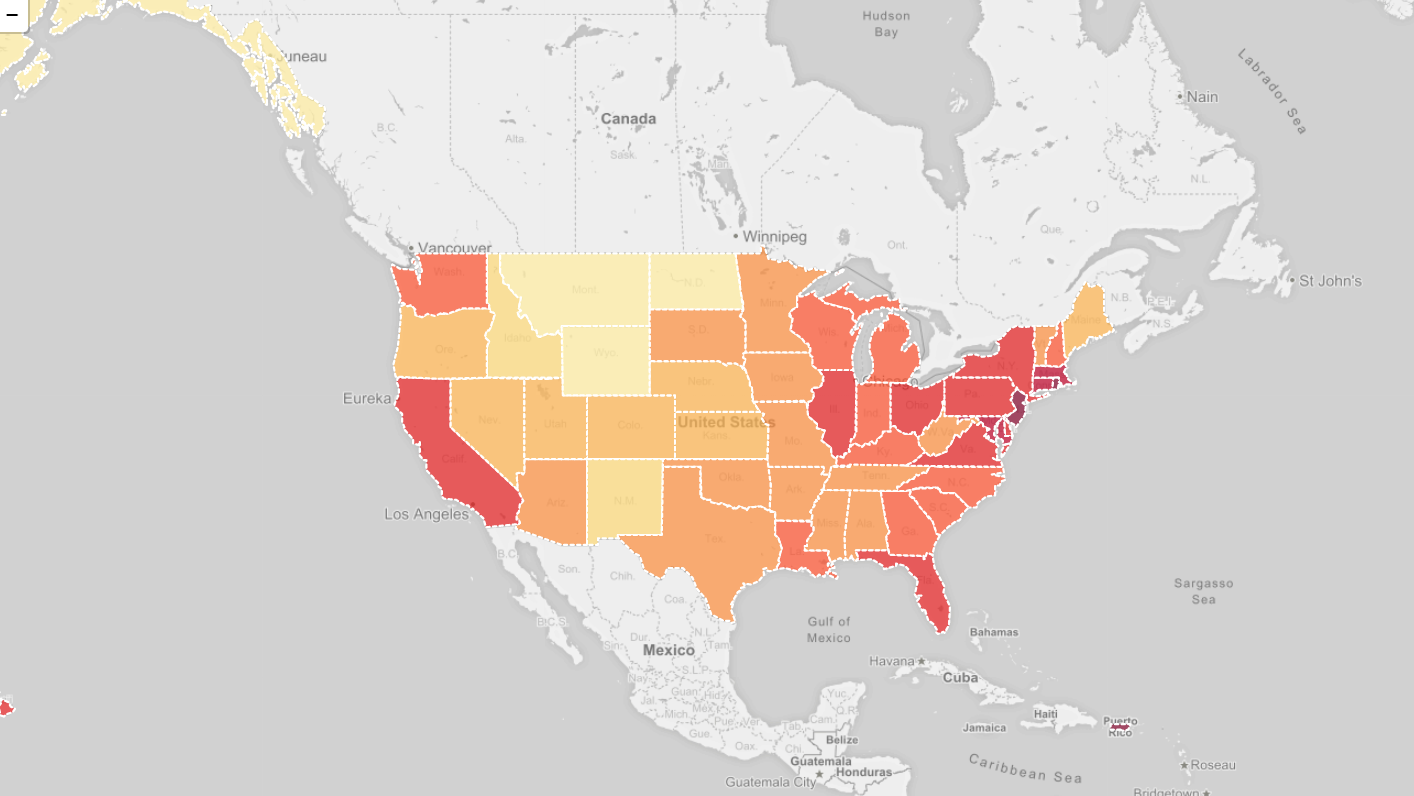
fillOpacity: 0.7

};

}

L.geoJson(statesData, {style: style}).addTo(map);

### Map View/Output



# External Links

* [openstreetmap:leaflet](https://wiki.openstreetmap.org/wiki/leaflet)
* [Leaflet Official Site](http://leafletjs.com/)
* [Leaflet Tutorials](http://www.digital-geography.com/category/leaflet-js/)

# Reference

* <http://leafletjs.com/index.html>
* <http://leafletjs.com/reference-1.1.0.html>