

Programación WEB (PWEB)

Tema 4 LeafletJS

Miguel Ángel Manso
Ramón Alcarria



1

Contenido

- Introducción
- Referencias y documentación
- Nuestro primer mapa con LeafLet

Introducción

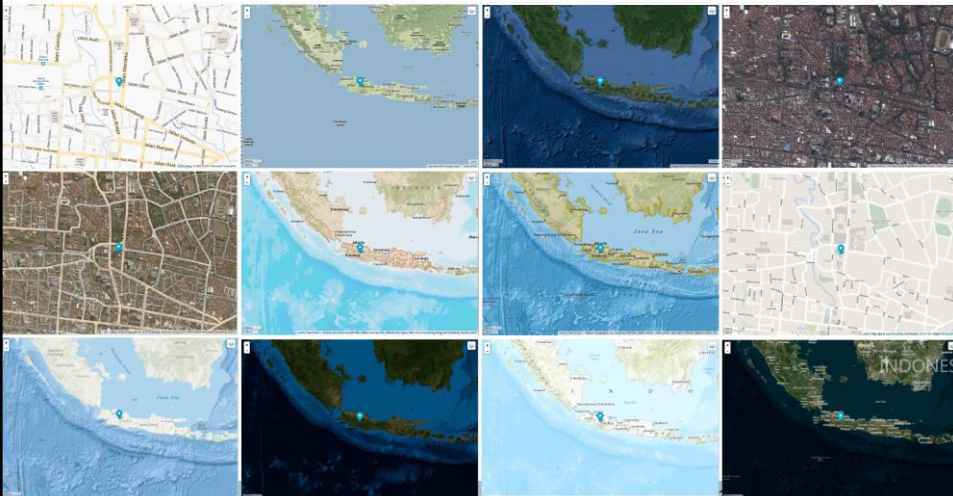
- Librería JavaScript de código abierto para la creación de mapas interactivos.
- Es utilizado por Flickr, Wikimedia, foursquare, OSM, cartoDB, GIS Cloud, Washington Post, Wall Street Journal, Geocaching.com, etc.
- Características:
 - Facilidad de aprendizaje y uso
 - Soporte móvil
 - HTML5 y CSS3
 - Compatibilidad con navegadores antiguos y modernos
 - Ampliable con plugins y API bien documentada
 - Web oficial: <http://leafletjs.com>

PWEB 2014-2015

3

Introducción

Mapas base disponibles



Introducción

Leaflet - a JavaScript library x

leafletjs.com

Star 10,446 Tweet Follow 13.1K followers Like 5.6k

Leaflet

An Open-Source JavaScript Library for Mobile-Friendly Interactive Maps

Overview Features Tutorials API Download Plugins Blog GitHub Twitter Forum

Leaflet is a modern open-source JavaScript library for mobile-friendly interactive maps. It is developed by [Vladimir Agafonkin](#) with a team of dedicated [contributors](#). Weighing just about 33 KB of JS, it has all the [features](#) most developers ever need for online maps.

Leaflet is designed with *simplicity, performance* and *usability* in mind. It works efficiently across all major desktop and mobile platforms out of the box, taking advantage of HTML5 and CSS3 on modern browsers while still being accessible on older ones. It can be extended with a huge amount of [plugins](#), has a beautiful, easy to use and [well-documented API](#) and a simple, readable [source code](#) that is a joy to [contribute](#) to.

Used by: Flickr foursquare Pinterest craigslist Data.gov IGN Wikimedia OSM Meetup WSJ Mapbox CartoDB GIS Cloud ...

A pretty CSS3 popup. Easily customizable.

PWEB 2014-2015 5

Referencias y documentación

- Características
 - <http://leafletjs.com/features.html>
- Ejemplos con código fuente:
 - <http://leafletjs.com/examples.html>
- Referencia (API Reference)
 - <http://leafletjs.com/reference.html>
- Creación de mapas con Leaflet.js:
 - <http://mappinggis.com/2013/06/como-crear-un-mapa-con-leaflet/>
 - <https://www.packtpub.com/books/content/getting-started-leaflet>

API Reference

The screenshot shows the Leaflet.js API Reference page for the `Map` class. The page is titled "Map" and includes a description, usage example, creation information, and options.

Map
The central class of the API – it is used to create a map on a page and manipulate it.

Usage example

```
// initialize the map on the "map" div with a given center and zoom
var map = L.map('map', {
  center: [51.505, -0.09],
  zoom: 15
});
```

Creation

Factory	Description
<code>L.map([HTMLElement String id, {Map options: options? }])</code>	Instantiates a map object given a div element (or its id) and optionally an object literal with map options described below.

Options

Map State Options

Option	Type	Default	Description
<code>center</code>	LatLng	null	Initial geographical center of the map.
<code>zoom</code>	Number	null	Initial map zoom.
<code>layers</code>	Layers[]	null	Layers that will be added to the map initially.
<code>minZoom</code>	Number	null	Minimum zoom level of the map. Overrides any <code>minZoom</code> set on map layers.
<code>maxZoom</code>	Number	null	Maximum zoom level of the map. This overrides any <code>maxZoom</code> set on map layers.
<code>maxBounds</code>	LatLngBounds	null	When this option is set, the map restricts the view to the given geographical bounds, bouncing the user back when he tries to pan outside the view. To set the restriction dynamically, use getMaxBounds method.
<code>crs</code>	CRS	L.CRS. EPSG:31466?	Coordinate Reference System to use. Don't change this if you're not sure what it means.

Interaction Options

Option	Type	Default	Description
--------	------	---------	-------------

Creando el primer mapa

- Incluiremos en la cabecera `<head>` de una página web la librería JavaScript `leaflet.js` y la hoja de estilo `leaflet.css`

```
<script src="http://cdn.leafletjs.com/leaflet-0.7/leaflet.js"></script>
```

```
<link rel="stylesheet" href="http://cdn.leafletjs.com/leaflet-0.7/leaflet.css" />
```

- Creamos un contenedor `html` con un identificador para que contenga el mapa

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

Creando el primer mapa

- Podemos crear un estilo para el mapa, de forma que tenga un tamaño definido. Dentro de `<style>`:

```
#map {
  height:70%;
  width:70%;
}
```

- Creamos el mapa:

```
var map = L.map('map'). setView([40.4193, -3.6931], 16);
```

- El mapa por defecto tiene dos controles, uno de zoom y otro de atribución.

Creando el primer mapa

- Añadimos un mapa base de OSM (teselado):
 - Debemos incluir URL, texto de la atribución, y el máximo nivel de zoom de la capa

```
L.tileLayer('http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
  attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, <a
href="http://creativecommons.org/licenses/by-sa/2.0/">CC-BY-SA</a>, Imagery © <a
href="http://cloudmade.com">CloudMade</a>',
  maxZoom: 18
}).addTo(map);
```

Creando el primer mapa

- Añadimos un control de escala

```
L.control.scale().addTo(map);
```

- Y por ejemplo un marcador:

```
L.marker([40.418, -3.684],{draggable:
true}).addTo(map).bindPopup('El estanque del Retiro');
```

Mapas avanzados

- Añadimos polígonos al mapa

```
var polygon = new L.Polygon([[40.421438, -3.680194], [40.411440, -3.676203],
[40.408270, -3.678478], [40.408956, -3.685816], [40.407322, -3.686632],
[40.407486, -3.688348], [40.419510, -3.688563]]).addTo(map);
```

- Creamos manejadores de eventos:

```
var popup = L.popup();

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

Mapas avanzados



- Iconos personalizados

```
var treeIcon = L.Icon({
  imageUrl: 'http://www.trees4life.ca/wp-content/uploads/2012/07/Tree-icon.png',
  iconSize: [60, 60], // tamaño del icono
  iconAnchor: [30, 50], // punto en el icono que se corresponde con la localización
  popupAnchor: [0, -45] // punto desde el que saldrá el popup
});
```

```
L.marker([40.41653, -3.6927], {icon: treeIcon}).addTo(map).bindPopup('Otro
parque');
```

Mapas avanzados

- Varias capas
 - Queremos juntar lo que hemos hecho con esto:

https://a.tiles.mapbox.com/v4/ramonalcarria.lm6go74l/page.html?access_token=pk.eyJ1IjoicmFtb25hbGNhcnJpYSIsImEiOiJ6djY2XzdVIno.2HPBM3VC_QherXzl-3epHw#4/40.41/-3.69

- Creamos un nuevo mapa base

```
var mapBoxLayer =
L.tileLayer('http://{s}.tiles.mapbox.com/v4/{mapId}/{z}/{x}/{y}.png?access_token={token}', {
  attribution: '<a href="http://mapbox.com">Mapbox</a>',
  maxZoom: 18,
  mapId: 'ramonalcarria.lm6go74l',
  token: 'pk.eyJ1IjoicmFtb25hbGNhcnJpYSIsImEiOiJ6djY2XzdVIno.2HPBM3VC_QherXzl-3epHw'
}).addTo(map);
```

Mapas avanzados

- Varias capas
 - Creamos una capa para los datos de ese mapa:

```
L.mapbox.accessToken =
'pk.eyJ1ljoicmFtb25hbGNhcnJpYSIsImEiOiJ6djY2XzdVIno.2HPBM3VC_QherXzl-3epHw';
```

```
var monumentos = L.mapbox.featureLayer('ramonalcarria.lm6g074l');
```

- Para ello necesitamos importar la librería de mapBox

```
<script src='https://api.tiles.mapbox.com/mapbox.js/v2.1.9/mapbox.js'></script>
```

Mapas avanzados

- Varias capas: Agrupamos los datos
 - Layer groups: Una capa con todas las geometrías

```
var parques = L.layerGroup([marker1, marker2, polygon]);
```

```
var overlayMaps = {
  "Parques": parques,
  "Monumentos": monumentos
};
```

- Layer switcher: Nos permite seleccionar el mapa base

```
var baseMaps = {
  "OpenStreetMaps": osmLayer,
  "Mapbox": mapBoxLayer
};
L.control.layers(baseMaps, overlayMaps).addTo(map);
```


Mapas avanzados

- Localización (para móviles)

- Se activa con:

```
map.locate({setView: true, maxZoom: 16});
```

- Definimos el manejador de la localización

```
function onLocationFound(e) {
  var radius = e.accuracy / 2;
  L.marker(e.latlng).addTo(map)
    .bindPopup("You are within " + radius + " meters from this point").openPopup();
  L.circle(e.latlng, radius).addTo(map);
}

map.on('locationfound', onLocationFound);
```

Mapas avanzados

- Localización (para móviles)

- Definimos el manejador del error en la localización

```
function onLocationError(e) {
  alert(e.message);
}
```

```
map.on('locationerror', onLocationError);
```