



Adding To The Leaf Pile

# Leaflet.js Plugin Development

*Michael Moore*  
*Flat Rock Geographics*

<https://github.com/stuporglue/leaflet-spiders>



## Leaflet in 30 Seconds

- JavaScript web mapping Library
- Free
- Lightweight
  - 9k LOC vs. OpenLayer's 85k
- Many plugins for other use cases
  - Some things are still missing!



1. Follow a tutorial on [leafletjs.com](http://leafletjs.com)
2. Change `leaflet.js` to `leaflet-src.js`
3. Get as close as you can without customizing



## Finding What To Change: API

- Good API docs. Start there.
  - Find objects & methods related to task

### UI Layers

[Marker](#)

[Popup](#)

### Raster Layers

[TileLayer](#)

[TileLayer.WMS](#)

[TileLayer.Canvas](#)

[ImageOverlay](#)

### Vector Layers

[Path](#)

[Polyline](#)

[MultiPolyline](#)

[Polygon](#)

[MultiPolygon](#)

[Rectangle](#)

[Circle](#)

[CircleMarker](#)

### Other Layers

[LayerGroup](#)

[FeatureGroup](#)

[GeoJSON](#)

### Basic Types

[LatLng](#)

[LatLngBounds](#)

[Point](#)

[Bounds](#)

[Icon](#)

[DivIcon](#)

### Controls

[Control](#)

[Zoom](#)

[Attribution](#)

[Layers](#)

[Scale](#)

### Events

[Event methods](#)

[Event objects](#)

### Utility

[Class](#)

[Browser](#)

[Util](#)

[Transformation](#)

[LineUtil](#)

[PolyUtil](#)

### DOM Utility

[DomEvent](#)

[DomUtil](#)

[PosAnimation](#)

[Draggable](#)

### Interfaces

[IHandler](#)

[ILayer](#)

[IControl](#)

[IProjection](#)

[ICRS](#)

### Misc

[global switches](#)

[noConflict](#)

[version](#)

- Find those objects or in leaflet-src.js
- “L.Point = “
- “L.Point.prototype = ”
- “L.point = “
- “methodName: function(“

## L.Transformation

Represents an affine transformation: a set of... and doing the reverse. Used by L...

```
var tr = L.Transformation(0, 0);
```

### Methods

Method	Return
<code>transform( <a href="#">&lt;Point&gt;</a> point, <a href="#">&lt;Number&gt;</a> scale? )</code>	<a href="#">Point</a>
<code>untransform( <a href="#">&lt;Point&gt;</a> point, <a href="#">&lt;Number&gt;</a> scale? )</code>	<a href="#">Point</a>

- L.Object – the object itself
  - Initialization defined here
- L.Object.prototype – the object definition
  - This is what we actually want to modify
  - May be inherited, or may be defined explicitly
    - If inherited, you won't find it in the code
- L.object – convenience method
  - May have extra initialization options



## Special functions to know

- L.Class
  - Most Leaflet things inherit from this
- L.Class.extend
  - Create a new class by extending another with additional functions
- L.Class.include
  - Set functions in existing class
- L.Class.mergeOptions
  - Set new default option values

```
L.NewClass = L.Marker.extend({funcName: function(){}})
```

- Explore objects in debugger
  - Expand to see methods and variables
  - Use tab-completion!
- Use debugger on-next to figure out what gets called.
  - Step through to find which function you want to change





# Debugger Crash Course

The screenshot shows the Firebug JavaScript debugger interface. The main window displays the source code for `leaflet-src.js`, with the `latLngToPoint` function selected. The call stack on the right shows the following frames:

- `latLngToPoint(latlng=Object { lat=9.795677582829743, lng=-108.54186829743}, zoom=15, scale=1)`
- `project(latlng=Object { lat=9.795677582829743, lng=-108.54186829743}, zoom=15, scale=1)`
- `latLngToLayerPoint(latlng=Object { lat=9.795677582829743, lng=-108.54186829743}, zoom=15, scale=1)`
- `crawl() method.js (line 5)`
- `(?)() method.js (line 30)`

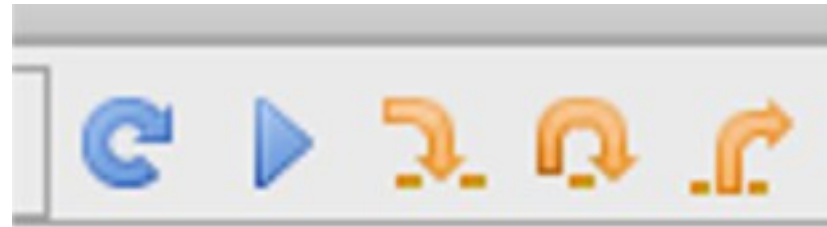
Annotations in the image include:

- A red circle around the `Break on Next` button (a blue square with a white lightning bolt) in the top toolbar.
- A red circle around the `Debugger Tools` button (a blue square with a white lightning bolt) in the top toolbar.
- A red box around the call stack area on the right.
- A red arrow pointing from the `Break on Next` button to the `JavaScript Console` area at the bottom.

The `JavaScript Console` at the bottom shows the following output:

```
>>> self
Window method.html
>>> this
Object { options={...}, _container=div#map.leaflet-container, _panes={...}, more... }
```

1. Reload
2. Continue
3. Step In To
4. Step Over
5. Step Out



1 2 3 4 5

Let's add some Halloween spiders to the map and make them dance

We will use 3 different methods!





## Ways To Bend Leaflet To Your Will

- Add new methods
- Override existing methods
- Create new objects by extending existing
  
- Not covered
  - New Objects From Scratch
  - Modifying Leaflet Itself
  
- Not recommended
  - Reaching Inside Leaflet Objects



## Static Spiders

### Demo

```
// Customization Code
/* NONE*/

// Add some spider code
var spiders = [];
var spider,lat,lng;
var spiderIcon = new L.Icon({iconUrl:'img/spider.png'});

for(var i = 0;i<10;i++){
  lat = Math.random()*180 - 90;
  lng = Math.random()*360 - 180;
  spider = new L.Marker([lat,lng],{icon:spiderIcon});
  spider.addTo(map);
}
```



## Adding a New Method

### DEMO

// Customization Code

// We add a new function called \*crawl\* to the Marker object

```
L.Marker.include({crawl: function(){
  // Convert to a pixel and jiggle it up to 5 pixels in any direction
  var pixel = this._map.latLngToLayerPoint(this._latlng);
  pixel.x += Math.round(Math.random()*10) - 5;
  pixel.y += Math.round(Math.random()*10) - 5;

  this._latlng = this._map.layerPointToLatLng(pixel);
  this.update();
}});
```



## New Method (cont)

```
// Add some spiders code
var spiders = [];
var spider,lat,lng;
var spiderIcon = new L.Icon({iconUrl:'img/spider.png'});

for(var i = 0;i<10;i++){
  lat = Math.random()*180 - 90;
  lng = Math.random()*360 - 180;
  spider = new L.Marker([lat,lng],{icon:spiderIcon});
  spider.addTo(map);
  spiders.push(spider);
}

// Every 200ms make each spider crawl
setInterval(function(){
  for(var i = 0;i<spiders.length;i++){
    spiders[i].crawl();
  }
},200);
```



## Override Method

### DEMO

// Customization Code

```
L.Marker.include({
```

```
  // Save off the original update function so we can use it later
  _origUpdate: L.Marker.prototype.update,
```

```
  // Define our new update function
```

```
  update: function(){
```

```
    // Convert to a pixel and jiggle it up to 5 pixels in any direction
```

```
    var pixel = this._map.latLngToLayerPoint(this._latlng);
```

```
    pixel.x += Math.round(Math.random()*10) - 5;
```

```
    pixel.y += Math.round(Math.random()*10) - 5;
```

```
    this._latlng = this._map.layerPointToLatLng(pixel);
```

```
    return this._origUpdate();
```

```
  }
```

```
});
```





## Override Method (cont)

```
// Add some spiders code
var spiders = [];
var spider,lat,lng;
var spiderIcon = new L.Icon({iconUrl:'img/spider.png'});

for(var i = 0;i<10;i++){
  lat = Math.random()*180 - 90;
  lng = Math.random()*360 - 180;
  spider = new L.Marker([lat,lng],{icon:spiderIcon});
  spider.addTo(map);
  spiders.push(spider);
}

setInterval(function(){
  for(var i = 0;i<spiders.length;i++){
    spiders[i].update();
  }
},200);
```



## Object Extending

### DEMO

```
L.Spider = L.Marker.extend({
  // Anything we don't define will call up to the L.Marker object
  options: {
    icon: new L.Icon({iconUrl:'img/spider.png'})
  },

  // Called when object created
  initialize: function(){
    var lat = Math.random()*180 - 90;
    var lng = Math.random()*360 - 180;
    this._latlng = L.latLng(lat,lng);

    var self = this;
    setInterval(function(){self.crawl();}, 200);
  },

  // Move the spider
  crawl: function(){
    var pixel = this._map.latLngToLayerPoint(this._latlng);
    pixel.x += Math.round(Math.random()*10) - 5;
    pixel.y += Math.round(Math.random()*10) - 5;

    this._latlng = this._map.layerPointToLatLng(pixel);
    return this.update();
  }
});

L.spider = function(latlng,options){
  return new L.Spider();
};
```



## Object Extending (cont)

```
for(var i = 0;i<10;i++){  
  new L.Spider().addTo(map);  
}
```



Get the Code on GitHub!

<https://github.com/stuporglue/leaflet-spiders>

Demo of different ways to extend Leaflet, using silly moving spiders — Edit

9 commits   2 branches   0 releases   1 contributor

branch: gh-pages   leaflet-spiders

This branch is 3 commits ahead and 5 commits behind master   Pull Request   Compare

leaflet-src.js		
	stuporglue authored 2 days ago	latest commit 6562f3a4ce
css	Initial files for demo	6 days ago
img	Basemap for the examples	6 days ago
js	leaflet-src.js	2 days ago
extend.html	leaflet-src.js	2 days ago
index.html	leaflet-src.js	2 days ago
method.html	leaflet-src.js	2 days ago
override.html	leaflet-src.js	2 days ago

We recommend adding a README to this repository to help give people an overview of your project.   Add a README

<> Code

Issues 0

Pull Requests 0

Pulse

Graphs

Network

Settings

HTTPS clone URL

<https://github.com/>

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

Clone in Desktop

Download ZIP





- [Leaflet Releases Source Code](#)
- [Leaflet API](#)
- [Leaflet Plugin Authoring Guide](#)
- [Leaflet-Spiders on GitHub](#)
  - [Leaflet-Spiders Demo](#)