# Leaflet Draw API reference



This documentation is has been transcribed from the original README.MD to jsdoc's or natural docs style for use with Leafdoc. If you identify a typo or have a suggestion for this documentation, please feel free toedit the js comment blocks in the src directory, build with 'jake docs' and submit a pull request.

## Leaflet 1.0+ Examples

- Full Demo
- Popup
- Snapping
- Edit Handlers

## Leaflet 0.7+ Examples

- Full Demo
- Popup
- Snapping
- Edit Handlers

# L.Draw

To add the draw toolbar set the option drawControl: true in the map options.

## Usage example

```
var map = L.map('map', {drawControl: true}).setView([51.505, -0
L.tileLayer('http://{s}.tile.osm.org/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="http://osm.org/copyright">Ope
}).addTo(map);
```

## Adding the edit toolbar

To use the edit toolbar you must initialise the Leaflet.draw control and manually add it to the map.

```
var map = L.map('map').setView([51.505, -0.09], 13);
L.tileLayer('http://{s}.tile.osm.org/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="http://osm.org/copyright">Ope
}).addTo(map);
// FeatureGroup is to store editable layers
var drawnItems = new L.FeatureGroup();
map.addLayer(drawnItems);
var drawControl = new L.Control.Draw({
    edit: {
        featureGroup: drawnItems
    }
});
map.addControl(drawControl);
```

The key here is the featureGroup option. This tells the plugin which FeatureGroup contains the layers that should be editable. The featureGroup can contain 0 or more features with geometry types Point, LineString, and Polygon. Leaflet.draw does not work with multigeometry features such as MultiPoint, MultiLineString, MultiPolygon, or GeometryCollection. If you need to add multigeometry features to the draw plugin, convert them to a FeatureCollection of non-multigeometries (Points, LineStrings, or Polygons).

# L.drawLocal

The core toolbar class of the API — it is used to create the toolbar ui

## Usage example

```
var modifiedDraw = L.drawLocal.extend({
    draw: {
        toolbar: {
            buttons: {
                polygon: 'Draw an awesome polygon'
            }
        }
    }
});
```

The default state for the control is the draw toolbar just below the zoom control. This will allow map users to draw vectors and markers. **Please note the edit toolbar is not enabled by default.**

# L.Draw.Toolbar

The toolbar class of the API — it is used to create the ui This will be depreciated

## Usage example

```
var toolbar = L.Toolbar();
toolbar.addToolbar(map);
```

## Disabling a toolbar

If you do not want a particular toolbar in your app you can turn it off by setting the toolbar to false.

```
var drawControl = new L.Control.Draw({
    draw: false,
    edit: {
        featureGroup: editableLayers
```

```
    }
});
```

### Disabling a toolbar item

If you want to turn off a particular toolbar item, set it to false. The following disables drawing polygons and markers. It also turns off the ability to edit layers.

```
var drawControl = new L.Control.Draw({
    draw: {
        polygon: false,
        marker: false
    },
    edit: {
        featureGroup: editableLayers,
        edit: false
    }
});
```

### Methods

Methods for modifying the toolbar

| Method | Returns | Description |
|---|---|---|
| `initialize(options)` | `void` | Toolbar constructor |
| `enabled()` | `boolean` | Gets a true/false of whether the toolbar is enabled |
| `disable()` | `void` | Disables the toolbar |
| `addToolbar(map)` | `L.DomUtil` | Adds the toolbar to the map and returns the toolbar dom element |
| `removeToolbar()` | `void` | Removes the toolbar and drops the handler event listeners |

## L.Draw.Event

Use `L.Draw.Event.EVENTNAME` constants to ensure events are correct.

## Usage example

```
map.on(L.Draw.Event.CREATED; function (e) {
   var type = e.layerType,
       layer = e.layer;
   if (type === 'marker') {
       // Do marker specific actions
   }
   // Do whatever else you need to. (save to db; add to map etc)
   map.addLayer(layer);
});


     map.on('draw:edited', function (e) {
         var layers = e.layers;
         layers.eachLayer(function (layer) {
             //do whatever you want; most likely save back to db
         });
     });
```

## Events

| Event | Data | Description |
|---|---|---|
| `draw:created` | `PolyLine` | Polygon; Rectangle; Circle; Marker \| String Layer that was just created. The type of layer this is. One of: `polyline`; `polygon`; `rectangle`; `circle`; `marker` Triggered when a new vector or marker has been created. |
| `draw:edited` | `LayerGroup` | List of all layers just edited on the map. Triggered when layers in the FeatureGroup; initialised with the plugin; have been edited and saved. |
| `draw:deleted` | `LayerGroup` | List of all layers just removed from the map. Triggered when layers have been removed (and saved) from the FeatureGroup. |
| `draw:drawstart` | `String` | The type of layer this is. One of:`polyline`; `polygon`; `rectangle`; `circle`; `marker` Triggered when the user has chosen to draw a particular vector or marker. |

| Event | Data | Description |
|---|---|---|
| `draw:drawstop` | `String` | The type of layer this is. One of: `polyline`; `polygon`; `rectangle`; `circle`; `marker` Triggered when the user has finished a particular vector or marker. |
| `draw:drawvertex` | `LayerGroup` | List of all layers just being added from the map. Triggered when a vertex is created on a polyline or polygon. |
| `draw:editstart` | `String` | The type of edit this is. One of: `edit` Triggered when the user starts edit mode by clicking the edit tool button. |
| `draw:editmove` | `ILayer` | Layer that was just moved. Triggered as the user moves a rectangle; circle or marker. |
| `draw:editresize` | `ILayer` | Layer that was just moved. Triggered as the user resizes a rectangle or circle. |
| `draw:editvertex` | `LayerGroup` | List of all layers just being edited from the map. Triggered when a vertex is edited on a polyline or polygon. |
| `draw:editstop` | `String` | The type of edit this is. One of: `edit` Triggered when the user has finshed editing (edit mode) and saves edits. |
| `draw:deletestart` | `String` | The type of edit this is. One of: `remove` Triggered when the user starts remove mode by clicking the remove tool button. |
| `draw:deletestop` | `String` | The type of edit this is. One of: `remove` Triggered when the user has finished removing shapes (remove mode) and saves. |
| `draw:toolbaropened` | `String` | Triggered when a toolbar is opened. |
| `draw:toolbarclosed` | `String` | Triggered when a toolbar is closed. |
| `draw:markercontext` | `String` | Triggered when a marker is right clicked. |

# L.Draw.Feature

### Methods

| Method | Returns | Description |
|---|---|---|
| `initialize()` | `void` | |
| `enable()` | `void` | Enables this handler |
| `disable()` | `void` | |
| `addHooks()` | `void` | Add's event listeners to this handler |
| `removeHooks()` | `void` | Removes event listeners from this handler |
| `setOptions(`*`object`*`)` | `void` | Sets new options to this handler |

## L.Draw.SimpleShape

### Methods

| Method | Returns | Description |
|---|---|---|
| `initialize()` | `void` | |
| `addHooks()` | `void` | Add listener hooks to this handler. |
| `removeHooks()` | `void` | Remove listener hooks from this handler. |

▶ Methods inherited from L.Draw.Feature

## L.Draw.Marker

### Methods

| Method | Returns | Description |
|---|---|---|
| `initialize()` | `void` | |
| `addHooks()` | `void` | Add listener hooks to this handler. |
| `removeHooks()` | `void` | Remove listener hooks from this handler. |

▶ Methods inherited from L.Draw.Feature

# L.Draw.CircleMarker

## Methods

| Method | Returns | Description |
|---|---|---|
| `initialize()` | `void` | |

▶ Methods inherited from L.Draw.Marker

▶ Methods inherited from L.Draw.Feature

# L.Draw.Circle

## Methods

| Method | Returns | Description |
|---|---|---|
| `initialize()` | `void` | |

▶ Methods inherited from L.Draw.SimpleShape

▶ Methods inherited from L.Draw.Feature

# L.Draw.Polyline

## Methods

| Method | Returns | Description |
|---|---|---|
| `initialize()` | `void` | |
| `addHooks()` | `void` | Add listener hooks to this handler |

| Method | Returns | Description |
| --- | --- | --- |
| `removeHooks()` | `void` | Remove listener hooks from this handler. |
| `deleteLastVertex()` | `void` | Remove the last vertex from the polyline, removes polyline from map if only one point exists. |
| `addVertex()` | `void` | Add a vertex to the end of the polyline |
| `completeShape()` | `void` | Closes the polyline between the first and last points |

▶ Methods inherited from L.Draw.Feature

# L.Draw.Rectangle

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `initialize()` | `void` | |
| `disable()` | `void` | |

▶ Methods inherited from L.Draw.SimpleShape

▶ Methods inherited from L.Draw.Feature

# L.Draw.Polygon

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `initialize()` | `void` | |

▶ Methods inherited from L.Draw.Polyline

▶ Methods inherited from L.Draw.Feature

# L.Edit.SimpleShape

## Methods

| Method | Returns | Description |
|---|---|---|
| `intialize()` | `void` | |
| `addHooks()` | `void` | Add listener hooks to this handler |
| `removeHooks()` | `void` | Remove listener hooks from this handler |
| `updateMarkers()` | `void` | Remove the edit markers from this layer |

# L.Edit.Marker

## Methods

| Method | Returns | Description |
|---|---|---|
| `initialize()` | `void` | |
| `addHooks()` | `void` | Add listener hooks to this handler |
| `removeHooks()` | `void` | Remove listener hooks from this handler |

# L.Edit.CircleMarker

## Methods

▶ Methods inherited from L.Edit.SimpleShape

# L.Edit.Circle

## Methods

▶ Methods inherited from L.Edit.SimpleShape

# L.Edit.Polyline

## Methods

| Method | Returns | Description |
|---|---|---|
| `initialize()` | `void` | |
| `addHooks()` | `void` | Add listener hooks to this handler |
| `removeHooks()` | `void` | Remove listener hooks from this handler |
| `updateMarkers()` | `void` | Fire an update for each vertex handler |

# L.Edit.Rectangle

## Methods

▶ Methods inherited from L.Edit.SimpleShape

# L.EditToolbar.Edit

## Methods

| Method | Returns | Description |
|---|---|---|
| `intialize()` | `void` | |
| `enable()` | `void` | Enable the edit toolbar |
| `disable()` | `void` | Disable the edit toolbar |
| `addHooks()` | `void` | Add listener hooks for this handler |
| `removeHooks()` | `void` | Remove listener hooks for this handler |

| Method | Returns | Description |
| --- | --- | --- |
| `revertLayers()` | `void` | Revert each layer's geometry changes |
| `save()` | `void` | Save the layer geometries |

# L.EditToolbar.Delete

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `intialize()` | `void` | |
| `enable()` | `void` | Enable the delete toolbar |
| `disable()` | `void` | Disable the delete toolbar |
| `addHooks()` | `void` | Add listener hooks to this handler |
| `removeHooks()` | `void` | Remove listener hooks from this handler |
| `revertLayers()` | `void` | Revert the deleted layers back to their prior state. |
| `save()` | `void` | Save deleted layers |
| `removeAllLayers()` | `void` | Remove all delateable layers |

# L.GeometryUtil

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `geodesicArea()` | `number` | |
| `formattedNumber(`*`n`*`, `*`precision`*`)` | `string` | Returns n in specified number format (if defined) and precision |

| Method | Returns | Description |
| --- | --- | --- |
| `readableArea(area, isMetric, precision)` | `string` | Returns a readable area string in yards or metric. The value will be rounded as defined by the precision option object. |
| `readableDistance(distance, units)` | `string` | Converts a metric distance to one of [ feet, nauticalMile, metric or yards ] string |
| `readableDistance(distance, isMetric, useFeet, isNauticalMile, precision)` | `string` | Converts metric distance to distance string. The value will be rounded as defined by the precision option object. |
| `isVersion07x()` | `boolean` | Returns true if the Leaflet version is 0.7.x, false otherwise. |

# L.LatLngUtil

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `cloneLatLngs()` | `L.LatLngs[]` | Clone the latLng point or points or nested points and return an array with those points |
| `cloneLatLng(LatLng)` | `L.LatLng` | Clone the latLng and return a new LatLng object. |

# L.LineUtil

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `segmentsIntersect()` | `boolean` | Checks to see if two line segments intersect. Does not handle degenerate cases. http://compgeom.cs.uiuc.edu/~jeffe/teaching/373/notes/x06-sweepline.pdf |

# L.Polygon

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `intersects()` | `boolean` | Checks a polygon for any intersecting line segments. Ignores holes. |

# L.Polyline

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `intersects()` | `boolean` | Check to see if this polyline has any linesegments that intersect. NOTE: does not support detecting intersection for degenerate cases. |
| `newLatLngIntersects()` | `boolean` | Check for intersection if new latlng was added to this polyline. NOTE: does not support detecting intersection for degenerate cases. |
| `newPointIntersects()` | `boolean` | Check for intersection if new point was added to this polyline. newPoint must be a layer point. NOTE: does not support detecting intersection for degenerate cases. |

# L.Map.TouchExtend

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `initialize()` | `void` | Sets TouchExtend private accessor variables |
| `addHooks()` | `void` | Adds dom listener events to the map container |
| `removeHooks()` | `void` | Removes dom listener events from the map container |

# L.Control.Draw

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `initialize()` | `void` | Initializes draw control, toolbars from the options |
| `onAdd()` | `container` | Adds the toolbar container to the map |
| `onRemove()` | `void` | Removes the toolbars from the map toolbar container |
| `setDrawingOptions(`*`options`*`)` | `void` | Sets options to all toolbar instances |

# L.Draw.Tooltip

The tooltip class — it is used to display the tooltip while drawing This will be depreciated

## Usage example

```
var tooltip = L.Draw.Tooltip();
```

### Methods

#### Methods for modifying draw state

| Method | Returns | Description |
| --- | --- | --- |
| `initialize(map)` | `void` | Tooltip constructor |
| `dispose()` | `void` | Remove Tooltip DOM and unbind events |
| `updateContent(labelText)` | `this` | Changes the tooltip text to string in function call |
| `updatePosition(latlng)` | `this` | Changes the location of the tooltip |
| `showAsError()` | `this` | Applies error class to tooltip |
| `removeError()` | `this` | Removes the error class from the tooltip |

## L.DrawToolbar

### Methods

| Method | Returns | Description |
| --- | --- | --- |
| `initialize()` | `void` | |
| `getModeHandlers()` | `object` | Get mode handlers information |
| `getActions()` | `object` | Get action information |
| `setOptions()` | `void` | Sets the options to the toolbar |

## L.EditToolbar

### Methods

| Method | Returns | Description |
| --- | --- | --- |
| `intialize()` | `void` | |
| `getModeHandlers()` | `object` | Get mode handlers information |
| `getActions()` | `object` | Get actions information |
| `addToolbar(`*`map`*`)` | `L.DomUtil` | Adds the toolbar to the map |
| `removeToolbar()` | `void` | Removes the toolbar from the map |
| `disable()` | `void` | Disables the toolbar |

# L.Edit.PolyVerticesEdit

## Methods

| Method | Returns | Description |
| --- | --- | --- |
| `intialize()` | `void` | |
| `addHooks()` | `void` | Add listener hooks to this handler. |
| `removeHooks()` | `void` | Remove listener hooks from this handler. |
| `updateMarkers()` | `void` | Clear markers and update their location |

# L.Marker.Touch

This isn't full Touch support. This is just to get markers to also support dom touch events after creation

#TODO: find a better way of getting markers to support touch.

# Install

You have four methods of installing Leaflet.Draw, copy the leaflet.draw.js, css, and images from dist and embed directly into your application.

### npm

To install the plugin run `npm install leaflet-draw` via command line in your project. You must also require this in your project like so: `var leafletDraw = require('leaflet-draw');`

### bower

To install the plugin run `bower install leaflet-draw`.

### Builder

There is a custom builder at ../build/build.html to help make a custom package of Leaflet.Draw with the command line.

## CDN's

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/le
<script src="https://cdnjs.cloudflare.com/ajax/libs/leaflet.draw/0.4.2
```

## Options

You can configure the plugin by using the different options listed here. If you identify a typo or have a suggestion for this section of the documentation, please edit docs-misc.leafdoc in the build directory.

## Control.Draw

| Option | Type | Default | Description |
|---|---|---|---|
| position | String | `'topleft'` | The initial position of the control (one of the map corners). See control positions. |

| Option | Type | Default | Description |
|---|---|---|---|
| draw | DrawOptions | `{}` | The options used to configure the draw toolbar. |
| edit | EditPolyOptions | `false` | The options used to configure the edit toolbar. |

## DrawOptions

| Option | Type | Default | Description |
|---|---|---|---|
| polyline | PolylineOptions | `{ }` | Polyline draw handler options. Set to `false` to disable handler. |
| polygon | PolygonOptions | `{ }` | Polygon draw handler options. Set to `false` to disable handler. |
| rectangle | RectangleOptions | `{ }` | Rectangle draw handler options. Set to `false` to disable handler. |
| circle | CircleOptions | `{ }` | Circle draw handler options. Set to `false` to disable handler. |
| marker | MarkerOptions | `{ }` | Marker draw handler options. Set to `false` to disable handler. |
| circlemarker | CircleMarkerOptions | `{ }` | Circle marker draw handler options. Set to `false` to disable handler. |

## PolylineOptions

| Option | Type | Default | Description |
|---|---|---|---|
| allowIntersection | Bool | `true` | Determines if line segments can cross. |
| drawError | Object | See code | Configuration options for the error that displays if an intersection is detected. |
| guidelineDistance | Number | `20` | Distance in pixels between each guide dash. |
| shapeOptions | Leaflet Polyline | See code | The options used when drawing the polyline/polygon on the map. |

| Option | Type | Default | Description |
|---|---|---|---|
| | options | | |
| showLength | Bool | `true` | Show the length of the drawn line. **The area is only approximate and become less accurate the larger the polygon is.** |
| metric | Bool | `true` | Use the metric measurement system. |
| feet | Bool | `true` | Use feet instead of yards and miles, when not using the metric measurement system. |
| nautic | Bool | `false` | Use nautic miles instead of yards and miles, when not using the metric measurement system nor feet. |
| zIndexOffset | Number | `2000` | This should be a high number to ensure that you can draw over all other layers on the map. |
| repeatMode | Bool | `false` | Determines if the draw tool remains enabled after drawing a shape. |

## PolygonOptions

| Option | Type | Default | Description |
|---|---|---|---|
| showArea | Bool | `false` | Show the area of the drawn polygon in m², ha or km². **The area is only approximate and become less accurate the larger the polygon is.** |
| showLength | Bool | `false` | Show the length of the drawn line. **The area is only approximate and become less accurate the larger the polygon is.** |
| metric | Object | `true` | Use the metric measurement system. Can be a boolean value, but can also be an array to specify which units to use. Possible units are `km` (kilometers), `ha` (hectares), `m` (metres). So a value of `['km', 'm']` means that the length will be shown in metres and, when more than a 1000 metres, in kilometers, and the area will be shown in m² or km² and acres will not be used. |

| Option | Type | Default | Description |
|---|---|---|---|
| feet | Bool | `true` | Use feet instead of yards and miles, when not using the metric measurement system. |
| nautic | Bool | `false` | Use nautic miles instead of yards and miles, when not using the metric measurement system nor feet. |
| precision | Object | `{km: 2, ha: 2, m: 0, mi: 2, ac: 2, yd: 0, ft: 0, nm: 2}` | Defines the precision to use for numbers of each type of unit. Possible units are `km` (kilometers), `ha` (hectares), `m` (metres), `mi` (miles), `ac` (acres), `ya` (yards), `ft` (feet), `nm` (nautical miles). For example `{km: 1}` changes the default precision for km and km² to one which gives values like `1.5 km` and `15.0 km²` in stead of `1.53 km` and `15.01 km²`. |

# RectangleOptions

| Option | Type | Default | Description |
|---|---|---|---|
| shapeOptions | Leaflet Path options | See code | The options used when drawing the rectangle on the map. |
| repeatMode | Bool | `false` | Determines if the draw tool remains enabled after drawing a shape. |
| showRadius | Bool | `true` | Show the area of the drawn circle in m², ha or km². **The area is only approximate and become less accurate the larger the circle is.** |

# CircleOptions

| Option | Type | Default | Description |
|---|---|---|---|
| shapeOptions | Leaflet Path options | See code | The options used when drawing the circle on the map. |

| Option | Type | Default | Description |
|---|---|---|---|
| repeatMode | Bool | `false` | Determines if the draw tool remains enabled after drawing a shape. |

## MarkerOptions

| Option | Type | Default | Description |
|---|---|---|---|
| icon | Leaflet Icon | `L.Icon.Default()` | The icon displayed when drawing a marker. |
| zIndexOffset | Number | `2000` | This should be a high number to ensure that you can draw over all other layers on the map. |
| repeatMode | Bool | `false` | Determines if the draw tool remains enabled after drawing a shape. |

## EditPolyOptions

| Option | Type | Default | Description |
|---|---|---|---|
| featureGroup | Leaflet FeatureGroup | `null` | This is the FeatureGroup that stores all editable shapes. **THIS IS REQUIRED FOR THE EDIT TOOLBAR TO WORK** |
| edit | EditHandlerOptions | `{ }` | Edit handler options. Set to `false` to disable handler. |
| remove | DeleteHandlerOptions | `{ }` | Delete handler options. Set to `false` to disable handler. |
| poly | EditPolyOptions | `{ }` | Set polygon editing options |
| allowIntersection | Bool | `true` | Determines if line segments can cross. |

# EditHandlerOptions

| Option | Type | Default | Description |
|---|---|---|---|
| selectedPathOptions | Leaflet Path options | See code | The path options for how the layers will look while in edit mode. If this is set to null the editable path options will not be set. |

**Note:** To maintain the original layer color of the layer use `maintainColor: true` within `selectedPathOptions`. E.g. The edit options below will maintain the layer color and set the edit opacity to 0.3.

```
{
    selectedPathOptions: {
        maintainColor: true,
        opacity: 0.3
    }
}
```

# version

A constant that represents the Leaflet.Draw version in use.

```
L.drawVersion; // contains "0.4.2" (or whatever version is current
```

© 2012 - 2017 Leaflet and Leaflet Organization
Thanks to all our Contributors
Documentation generated with Leafdoc