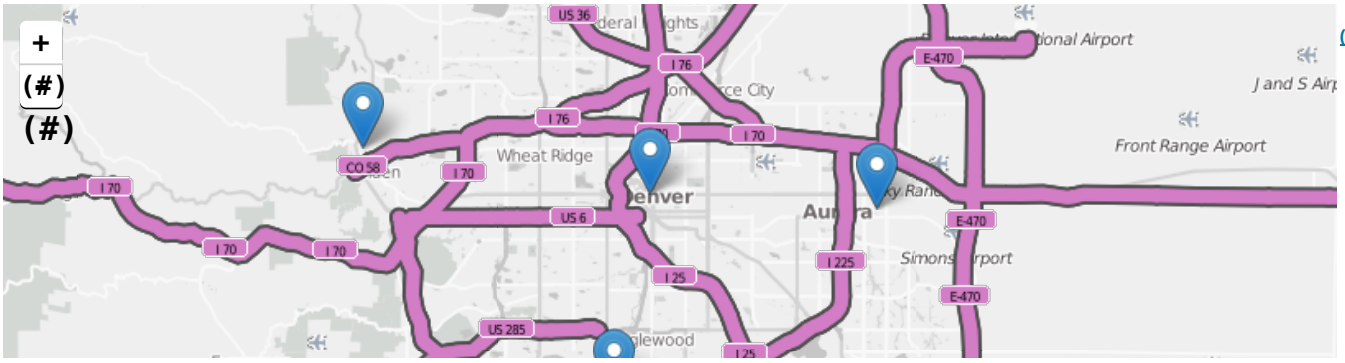
Leaflet (http://leafletjs.com)

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

Layer Groups and Layers Control

This tutorial will show you how to group several layers into one, and how to use the layers control to allow users to easily switch different layers on your map.



View example on a separate page → (layers-control-example.html)

Layer Groups

Lets suppose you have a bunch of layers you want to combine into a group to handle them as one in your code:

```
var littleton = L.marker([39.61, -105.02]).bindPopup('This is Littleton, CO.'),
    denver    = L.marker([39.74, -104.99]).bindPopup('This is Denver, CO.'),
    aurora    = L.marker([39.73, -104.8]).bindPopup('This is Aurora, CO.'),
    golden    = L.marker([39.77, -105.23]).bindPopup('This is Golden, CO.');
```

Instead of adding them directly to the map, you can do the following, using the LayerGroup (http://leafletjs.com/reference.html#layergroup) class:

```
var cities = L.layerGroup([littleton, denver, aurora, golden]);
```

Easy enough! Now you have a `cities` layer that combines your city markers into one layer you can add or remove from the map at once.

Layers Control

Leaflet has a nice little control that allows your users control what layers they want to see on your map. In addition to showing you how to use it, we'll show another handy use for layer groups.

There are two types of layers — base layers that are mutually exclusive (only one can be visible on your map), e.g. tile layers, and overlays — all the other stuff you put over the base layers. In this example, we want to have two base layers (minimal and night-style base map) to switch between, and two overlays to switch on and off — a pink motorways overlay and city markers (those we created earlier). Lets create those layers and add the default ones to the map:

```
var cloudmadeUrl = 'http://{s}.tile.cloudmade.com/API-key (http://account.cloudmade.com/register)/{styleId}/256/{z}/{x}/{y}.png',
```

```
    cloudmadeAttribution = 'Map data &copy; 2011 OpenStreetMap contributors, Imagery &copy; 2011 CloudMade';

var minimal   = L.tileLayer(cloudmadeUrl, {styleId: 22677, attribution: cloudmadeAttribution}),
    midnight  = L.tileLayer(cloudmadeUrl, {styleId: 999,   attribution: cloudmadeAttribution}),
    motorways = L.tileLayer(cloudmadeUrl, {styleId: 46561, attribution: cloudmadeAttribution});

var map = L.map('map', {
    center: new L.LatLng(39.73, -104.99),
    zoom: 10,
    layers: [minimal, motorways, cities]
});
```

Next, we'll create two objects. One will contain our base layers and one will contain our overlays. These are just simple objects with key/value pairs. The key is what sets the text for the layer in the control (e.g. "Night View"). The corresponding value is a reference to the layer (e.g. `midnight`).

```
var baseMaps = {
    "Minimal": minimal,
    "Night View": midnight
};

var overlayMaps = {
    "Motorways": motorways,
    "Cities": cities
};
```

Now, all that's left to do is to create a **Layers Control (../reference.html#control-layers)** and add it to the map. The first argument passed when creating the layers control is the base layers object. The second argument is the overlays object. Both arguments are optional — for example, you can pass just a base layers object by ommiting the second argument, or just an overlays objects by passing `null` as the first argument.

```
L.control.layers(baseMaps, overlayMaps).addTo(map);
```

Note that we added `minimal`, `motorways` and `cities` layers to the map but didn't add `midnight`. The layers control is smart enough to detect what layers we've already added and have corresponding checkboxes and radioboxes set.

Also note that when using multiple base layers, only one of them should be added to the map at instantiation, but all of them should be present in the base layers object when creating the layers control.

Now lets **view the result on a separate page → (layers-control-example.html)**

(http://github.com/Leaflet/Leaflet)