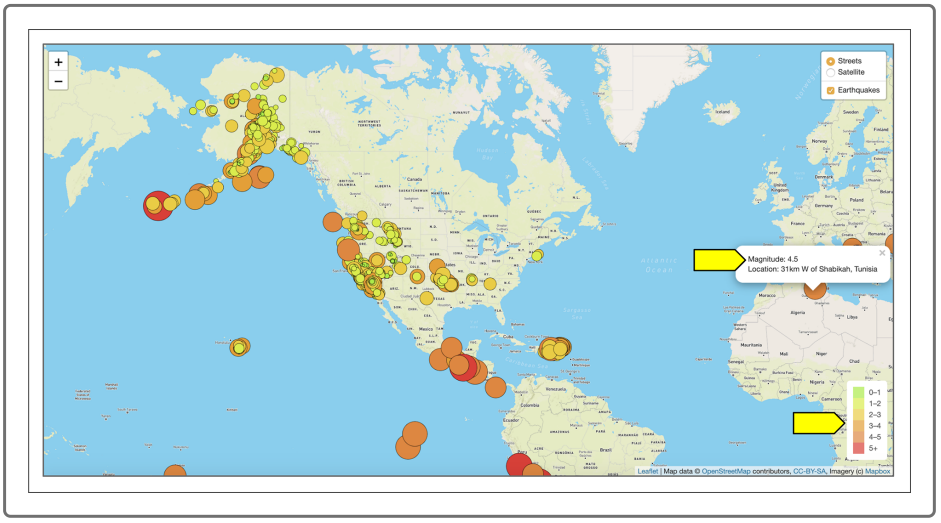
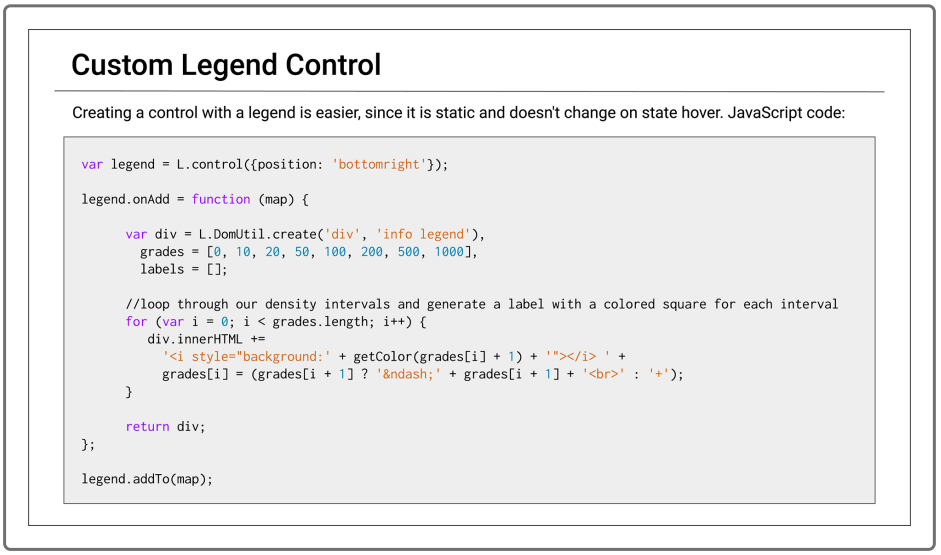
13.6.5 Add a Legend to the Map

There is one final piece to add to the map: a legend for the color range of the earthquakes. Basil and Sadhana think a legend will provide information needed for the colors of the earthquakes to make sense to the viewer without having to click on each marker.

After we add a legend, our earthquake map should look similar to the following map, with the legend at the bottom right of the map:



Sadhana tells you that we can use the Leaflet **choropleth examples page (https://leafletjs.com/examples/choropleth/)** to help us write the code for the legend. On the **choropleth examples page (https://leafletjs.com/examples/choropleth/)** , scroll down to the "Custom Legend Control" section:



Custom Legend Control

Creating a control with a legend is easier, since it is static and doesn't change on state hover. JavaScript code:

```
var legend = L.control({position: 'bottomright'});

legend.onAdd = function (map) {

      var div = L.DomUtil.create('div', 'info legend'),
        grades = [0, 10, 20, 50, 100, 200, 500, 1000],
        labels = [];

      //loop through our density intervals and generate a label with a colored square for each interval
      for (var i = 0; i < grades.length; i++) {
         div.innerHTML +=
           '<i style="background:' + getColor(grades[i] + 1) + '"></i> ' +
           grades[i] = (grades[i + 1] ? '&ndash;' + grades[i + 1] + '<br>' : '+');
      }

      return div;
};

legend.addTo(map);
```

Before we write the code to create this map, make a copy of the `logicStep4.js` file and name it `logicStep5.js`. Now let's edit the file.

On the choropleth examples page, copy the code for the Custom Legend Control and paste it below the `L.geoJSON()` layer, where we add the earthquake layer to the map, `earthquakes.addTo(map)`. Now, we'll edit the legend control object to suit our needs.

First, edit the code for the Leaflet `control()` object to look like the following. With this code, we'll place the legend at the indicated position—the bottom right:

```
// Create a legend control object.
let legend = L.control({
  position: "bottomright"
});
```

Next, remove the argument "map" from the legend function to look like the following:

```
// Then add all the details for the legend.
legend.onAdd = function() {
  let div = L.DomUtil.create("div", "info legend");
};
```

With this code, we're going to add a legend to the map with `legend.onAdd`. The legend will be added to a `div` element on the `index.html` file using the

`DomUtil` utility function.

Next, we're going to change the `grades` array in the Leaflet documentation to a `magnitudes` array, and we'll add a `colors` array that holds the colors for our magnitudes. Add the following code inside our `legend.onAdd` function:

```
const magnitudes = [0, 1, 2, 3, 4, 5];
const colors = [
  "#98ee00",
  "#d4ee00",
  "#eecc00",
  "#ee9c00",
  "#ea822c",
  "#ea2c2c"
];
```

The final piece is to edit the `for` loop. The `for` loop will add the color choices from our `colors` array as a small box for the color of earthquakes and place the text of the magnitude range next to the box. Edit the `for` loop code to look like the following:

```
// Looping through our intervals to generate a label with a colored square f
   for (var i = 0; i < magnitudes.length; i++) {
     console.log(colors[i]);
     div.innerHTML +=
       "<i style='background: " + colors[i] + "'></i> " +
       magnitudes[i] + (magnitudes[i + 1] ? "&ndash;" + magnitudes[i + 1] +
  }
   return div;
 };

 legend.addTo(map);
```

Let's review what's happening in this `for` loop:

1. After we iterate through the `magnitudes`, we'll add the color and text to the `div` element using `div.innerHTML +=`.
2. For each iteration, we'll add a color from the `colors` array by styling the background of an `<i>` tag with color options.
3. Next, we'll add the interval between earthquake magnitudes for our colors with the following code: `magnitudes[i] + (magnitudes[i + 1] ? "&ndash;" + magnitudes[i + 1] + "<br>" : "+")`.

This code is quite complex. For a deeper explanation of this code, watch the following video:

NOTE

> If that seemed a bit complex, that's good! Encountering, unpacking, and using other people's complex code is a critical part of being a developer.

The last thing we need to do is style the legend using CSS. Below the JavaScript code for the legend is the CSS code. Copy the CSS code and add it our `style.css` file:

```
.legend {
  line-height: 18px;
  color: #555;
}
.legend i {
  width: 18px;
  height: 18px;
  float: left;
  margin-right: 8px;
  opacity: 0.7;
}
```

We won't edit the `<i>` tag of the `.legend` class (`.legend i`) in the `style.css` file, but let's edit our `.legend` class to create some padding

around the legend, add a white background, and add a border radius. Edit your `.legend` class to look like the following:

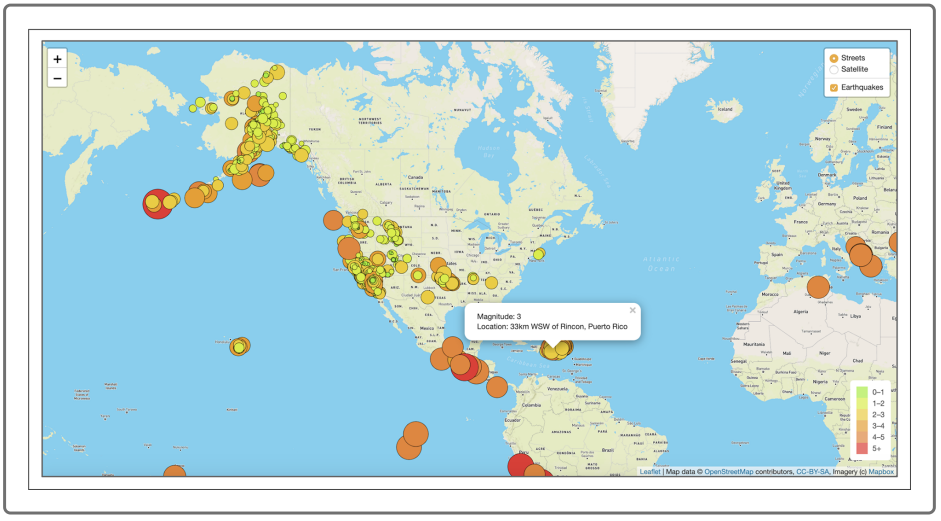
```
.legend {
  padding: 10px;
  line-height: 18px;
  color: #555;
  background-color: #fff;
  border-radius: 5px;
}
```

Our `style.css` file should now look like the following:

```
html,
body,
#mapid {
  width: 100%;
  height: 100%;
  padding: 0;
  margin: 0;
}
.legend {
  padding: 10px;
  line-height: 18px;
  color: #555;
  background-color: #fff;
  border-radius: 5px;
}

.legend i {
  float: left;
  width: 18px;
  height: 18px;
  margin-right: 8px;
  opacity: 0.7;
}
```

Save your `logicStep5.js` and `style.css` files. When you open `index.html` in your browser, your map should have a legend on the bottom right:



Congratulations on completing your earthquake map!

ADD/COMMIT/PUSH

Add, commit, and push your changes to your Earthquakes_past7days branch.