[Lesson 7.3: Let’s Map It! (secretly)](#h.d2m81n3o5v2j)

[Gem of the Day](#h.9hm1gvuv52ry)

[Hiding Secrets With Figaro](#h.fyk757t7fdxg)

[Google Maps API Key](#h.mgfv58sjmput)

[Implementing the Map With JavaScript](#h.it8kry35hih6)

[Basic Map](#h.cql1kiobmsi7)

[Center on User and Drop User Marker](#h.ren9i4o5ype)

[Add Bus Markers](#h.z0zc8deq6qqu)

[Change User Location Pin Icon](#h.uuny5f8rmqt4)

[Add Info Windows for Buses](#h.649umlgqnjaw)

[Required Homework - IMPORTANT!](#h.h6ongcli3r6n)

[Homework](#h.hw6hoy2g7g60)

# 

# Lesson 7.3: Let’s Map It! (secretly)

Our Bus Me app is great, but it would be even better if we could show a map with the user’s location and the buses near him or her.

## Gem of the Day

Pagination! <https://www.ruby-toolbox.com/categories/pagination>

* Will Paginate <https://github.com/mislav/will_paginate>
* Kaminari <https://github.com/amatsuda/kaminari>

## Hiding Secrets With Figaro

Figaro <https://github.com/laserlemon/figaro>

First, let’s create a new **branch** called “maps” where we will work for today.

Then, let’s add the Figaro gem:

gem 'figaro'

bundle

Then, we have to explicitly install per the readme:

figaro install

Now check your config folder - you should have a new file called application.yml which was also added to your .gitignore file so no one will ever see it - it won’t push anywhere else.

## Google Maps API Key

Now, let’s request an API key.

1. Go to <https://developers.google.com/maps/web/>
2. Click “get started with the javascript api”
3. Follow the directions to get your key.

Once you have your API key, add it to your **application.yml** file like so:

google\_maps\_api\_key: iAmnOtaReAlkEy12FrIdAy

Then, add the script link to the Google Maps API in the HEAD of your HTML:

<!DOCTYPE html>

<html>

<head>

<title>PreBusMe</title>

<script type="text/javascript" src="https://maps.googleapis.com/maps/api/js?key=<%= ENV['google\_maps\_api\_key'] %>"></script>

<%= stylesheet\_link\_tag 'application', media: 'all', 'data-turbolinks-track' => true %>

<%= javascript\_include\_tag 'application', 'data-turbolinks-track' => true %>

<%= csrf\_meta\_tags %>

</head>

When we deploy to Heroku, we will have to set this environment variable manually inside Heroku - instructions are [here](https://github.com/laserlemon/figaro#deployment) for future reference.

## Implementing the Map With JavaScript

### Basic Map

First, let’s attempt to put a basic map on our page - here is the simple example from the API documentation - put it in your show view above the loop, and see if it works:

<div id="map-canvas" style="height:500px;width:500px;margin:0;padding:0;"></div>

<script type="text/javascript">

function initialize() {

var mapOptions = {

center: { lat: -34.397, lng: 150.644},

zoom: 8

};

var map = new google.maps.Map(document.getElementById('map-canvas'),

mapOptions);

}

google.maps.event.addDomListener(window, 'load', initialize);

</script>

### Center on User and Drop User Marker

Cool! Now, let’s make it more interesting by making it centered on the user’s location and drop a marker for that user. While we are at it, let’s go ahead and put the script tag at the bottom of the file, after all the html and erb.

To do this, we need to set a new variable for the user’s latitude and longitude. You can see how to do this with the Google Maps API [here](https://developers.google.com/maps/documentation/javascript/tutorial#latitudes-and-longitudes). Then, we edit the mapOptions to center on that location. Finally, we drop a marker. See more about markers [here](https://developers.google.com/maps/documentation/javascript/markers).

function initialize() {

var myLatlng = new google.maps.LatLng(<%= @location.latitude %>, <%= @location.longitude %>);

var mapOptions = {

center: myLatlng,

zoom: 15

};

var map = new google.maps.Map(document.getElementById('map-canvas'),

mapOptions);

var myMarker = new google.maps.Marker({

position: myLatlng,

map: map,

animation: google.maps.Animation.DROP,

title: 'My Location'

});

}

google.maps.event.addDomListener(window, 'load', initialize);

### Add Bus Markers

How might we add buses to the map? With a loop. However, first we need the Ruby @nearby\_buses to be passed into a JavaScript array variable. Luckily, ERB and Rails have a fairly simple command for this:

var nearbyBuses = <%= raw @nearby\_buses.to\_json %>;

Now, we can loop through that buses object to grab the latitudes and longitudes, then drop markers, just like we did for the user location:

nearbyBuses.forEach(function (element){

var busLatlng = new google.maps.LatLng(element['LATITUDE'], element['LONGITUDE']);

var marker = new google.maps.Marker({

position: busLatlng,

map: map,

animation: google.maps.Animation.DROP,

title: element['VEHICLE']

});

});

### Change User Location Pin Icon

Nice! But, this is getting a little confusing since we can’t tell the difference between the user’s location and the bus locations. Let’s change the icon for the user. I’ll send you this icon to use for now, but feel free to change this icon in your homework.

var personIcon = "<%= image\_path 'man\_icon.png' %>";

var myMarker = new google.maps.Marker({

position: myLatlng,

map: map,

animation: google.maps.Animation.DROP,

title: 'My Location',

icon: personIcon

});

### Add Info Windows for Buses

You know what would be even better? If we could click on a bus marker and get that bus’ information. We can do that with [infowindows](https://developers.google.com/maps/documentation/javascript/infowindows).

First, we have to define the content for that window in html and put it in a variable:

var contentString = '<div id="content">'+

'<h2>Bus '+ element['VEHICLE']+' '+element['DIRECTION']+'</h2>'+

'<div id="bodyContent">'+

'<p>Next Stop: '+ element['TIMEPOINT'] +'</p>'+

'</div>'+

'</div>';

Then, we have to create the info window:

var infowindow = new google.maps.InfoWindow({

content: contentString

});

Finally, we have to add the event listener:

google.maps.event.addListener(marker, 'click', function() {

infowindow.open(map,marker);

});

Classroom challenge: Add an info window for the user marker that tells them their address.

## Required Homework - IMPORTANT!

1. Sign up for an account on <https://www.heroku.com/> and confirm your account through the email they send you.
2. Download the Heroku Toolbelt. You can get it from the Setup page of the Ruby tutorial for Heroku: <https://devcenter.heroku.com/articles/getting-started-with-ruby#set-up>
3. Test to make sure Heroku was installed correctly by opening your command line and typing heroku login and log in using your credentials. You should get something like this:  
   

## Homework

Continue optimizing your Bus Me app and making the Google Maps integration even better. Clean up the Show view especially. Add fake social media links, as if this were a real app. Also add a link to your repo on GitHub. We are going to share it with the world tomorrow!