# Geolocation

## Overview

In this lab, you'll see how to use the geolocation API to determine the user's current location as longitude/latitude values. You'll then see how to use this information in conjunction with the Google Maps API, to display the user's location on a map in the browser window.

Other mapping APIs are available (e.g. Bing Maps), but they all offer the same kind of functionality.

## Roadmap

There are 3 exercises in this lab, of which the last exercise is "if time permits". Here is a brief summary of the tasks you will perform in each exercise; more detailed instructions follow later:

1. Obtaining geolocation information
2. Displaying a map
3. Adding a marker to indicate the current location (if time permits)

## Server

HTML files can be opened from the file system, but for security reasons browsers will limit the possibilities. It is better to open a server in the project folders. Here are two ways to do that.

### Visual Studio Code

1. Open Visual Studio Code
2. Choose menu “File | Open…” and open the root folder of the course material. After opening you should see a folder called “.vscode” at the top of the tree in the Explorer on the left.
3. Choose menu “Tasks | Run Task…” and choose task “npm: install”.
4. Select/Open a file in the folder that you want to be the root of your server. This will generally be the homepage of your app.
5. Choose menu “Tasks | Run Build Task…” or choose the shortcut.
6. To close the server again, choose menu “Tasks | Terminate Task…”.

### Command line

1. Globally install “live-server” with command “npm install -g live-server” (see: https://www.npmjs.com/package/live-server)
2. Open the command line (Windows) or terminal (macOS) in the folder that you want to be the root of your server. This will generally be the homepage of your app.
3. Run “live-server” to open the server. Choose Ctrl+c to close the server again.

## Exercise 1: Getting geolocation information

In Visual Studio, open the *Start* project for this lab as follows:

- *Folder*:
  `\start`

The project contains an HTML page named `index.html`. Take a look at this page now. The page body contains a `<div>` named `mapContainer`, which will eventually display a map (obtained via the Google Maps API).

The first step is to obtain geolocation information. Follow these steps:

- In the `init()` function, test whether the browser supports geolocation. If it does, call the geolocation `getCurrentPosition()` function to get geolocation information.
- Write a callback function to receive the geolocation information. In this function, display the current longitude and latitude values in an alert box for now.
- You might also want to implement an "error callback" in case the geolocation request fails.

Run the Web application, to display `index.html` in the browser. As soon as the page opens, the browser should attempt to get geolocation information (it might ask you if you want to allow this request to happen, in which case you should say "yes" ☺). Verify that the application displays meaningful longitude/latitude values.

## Exercise 2: Displaying a map

The next step is to display a map showing the user's current location. The Google Maps API provides a rich programming model for getting map images, based on various coordinate systems.

To use the Google Map API, add a `<script>` tag to your web page (in the `<head>` section) to pull in the Google Map API JavaScript file (see the comment in the code for the appropriate URL).

Now add the following code in your geolocation callback function, to get a map from Google and to display it in the web page:

- Create a new `google.maps.LatLng` coordinate object. The constructor requires two parameters:
    - latitude
    - longitude
- Create a new JavaScript map options object (using the regular JavaScript object syntax, i.e. `{}`), and set the following properties on the object:
    - `zoom`: the zoom factor for the map (specify a value such as 15)
    - `center`: the coordinate object you created just now
    - `mapTypeControl`: true
    - `navigationControlOptions`: an object that configures the navigation icon. Set the object's `style` property`google.maps.NavigationControlStyle.SMALL`.
    - `mapTypeId`: the map type, e.g. `google.maps.MapTypeId.SATELLITE`
- Create a new `google.maps.Map` object. The constructor requires two parameters:
    - The element where you want to display the map – specify the `mapContainer` element here (note you must pass in the actual element, e.g. as obtained via the `document.getElementById()` function).
    - The map options object you just created.

That's it! Just by creating the `google.maps.Map` object, Google Maps will create a map and display it in the target element you specified. If you refresh the web page in the browser, you should see that it displays your current location in a map.

### Exercise 3 (If time permits): Adding a marker to indicate the current location

In your geolocation callback function, add code to display a marker on the map to indicate your current location. To do this, you must create a `google.maps.Marker` object – take a look at the Google Maps API documentation online to see what parameters are needed.

You might also want to experiment with other features in the Google Maps API – see the documentation and try them out in your web page.