Lab n°1

Learning Goals

The purpose of this lab is to get the prerequisites for Angular development installed on your machine and to prepare your first app for development using the Angular Command Line Interface or CLI.

Requirements

- Fork this repo : https://github.com/pr-elhajji/formation-angular
- Clone your fork into your ~/code/labs folder.
- Choose the branch "**lab1-first-app**"

Submission

Upon completion, run the following commands:

```
$ git add .
$ git commit -m "done"
$ git push origin "lab1-first-app"
```

Navigate to your repo and create a pull request from your **lab1-first-app** branch to the original repository's **lab1-first-app** branch.

In the pull request name, add your campus, name, and last name separated by a dash "-.

Deliverables

All the files, with the necessary code to satisfy the requirements described below.

Tutorial Labs

1. *Install Visual Studio Code*

   In this lab, you will learn how to install Visual Studio Code on your machine.

2. *Install Prerequisites*

In this lab, you will install Node, NPM, TypeScript, and the Angular CLI.

3. *Using the Angular CLI*

   In this lab, you will learn the basics of using the Angular CLI to generate components within your app, and to run your app locally.

   Angular in Visual Studio Code : https://code.visualstudio.com/docs/nodejs/angular-tutorial

   To Install Node.js and NPM

   Node.js - commonly referred to as Node - is a JavaScript runtime that allows you to run JavaScript on your computer without using a browser. This is the technology that enables JavaScript based application servers, including the one used by Angular.

   Node also includes a package manager for installing 3rd party modules within their environment called Node Package Manager, or NPM.

   In order to use the Angular CLI, we need to first install Node.js and NPM on your computer. Follow the instructions below based on which operating system you are currently using.

Mac OS X

1. If you do not have XCode installed, install it from the Apple App Store.

2. If you do not already have Homebrew installed, you will need to install it. Homebrew is a package manager specifically for OS X - you can find more about it on the Homebrew Website. To install it you run the following command in your terminal.

```
ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

3. With Homebrew installed, installing Node and NPM is easy. Just run the following:

```
brew install node
```

4. Follow the messages in the terminal to install Node.

Linux

To install Node on your Linux Machine, follow the instructions for your specific Linux distribution at Node's Linux Installation Instructions Page

Windows

1. Download the Node.js installer here. You must have administrative rights on your machine to install Node.

2. Run the installer.

3. Follow the prompts in the installer. Make sure that the option “Add to PATH” is selected. This allows you to run Node as a command in your command prompt.

4. Restart your computer.

To Install TypeScript

TypeScript is a superset of the JavaScript language that allows for stricter typing, along with a series of other features. Angular uses TypeScript instead of JavaScript. Now that we have Node and NPM installed, we can install TypeScript, the last prerequisite needed before installing the Angular CLI.

1. Installing TypeScript requires administrative privileges. For Windows, this means you need to open a command prompt or terminal with administrative rights. For Mac and Linux users, just open a terminal.

2. To install TypeScript, simply open your command prompt or terminal window and type:

```
npm install typescript -g
```

For Mac and Linux users you need sudo rights to install TypeScript, so type:

```
sudo npm install typescript -g
```

The -g flag tells NPM to install TypeScript as a global module, instead of to the directory your terminal is currently pointed it. This will enable you to be able to use the tsc command to compile TypeScript directly, and enable the Angular CLI to be able to use it.

Now that we have installed TypeScript, next we will install the Angular CLI.

To Install the Angular CLI

The Angular Command Line Interface is an optional component for Angular development that offers a series of generators and commands to make developing in Angular simpler and faster.

As you learned in Lesson 3, with the Angular CLI, you can create a scaffold for your app, generate components, modules, and also perform build tasks such as linting your code, serving your app locally, preparing your files for production usage, and using Angular's Ahead of Time compilation to optimize the performance of your app in production environments.

Let's go ahead and use NPM to install the Angular CLI on your machine.

1. Type the following into your command line or terminal window:

   ```
   npm install @angular/cli -g
   ```

   For Mac and Linux users you need sudo rights to install the Angular CLI, so type:

   ```
   sudo npm install @angular/cli -g
   ```

2. Press Enter, and observe the installation occurring (it will take some time).

   And that's it! Now we have everything installed to start creating our Angular app. In the next lab we will be using the Angular CLI to generate the scaffolding for our application.

To Generate an Application Scaffold using the Angular CLI

Now that we have the Angular CLI installed, we are going to use it to create a scaffold for our application. The term “Scaffold” refers to the process of generating a generic template of an application based on the standards of how an Angular application should be constructed. A basic folder hierarchy is created following layout and naming best-practices. The folders are then populated with the components necessary to create a basic Angular application. Creating a scaffold sets up the basic files and architecture required for your app to run, and gives you a good pattern to follow. Using the Angular CLI to build your scaffold also gives you a variety of useful tools, including:

- Linting
- Unit testing with Karma
- End to End testing with Protractor
- Ahead of Time Compilation

- A built in local server

Let's go ahead and instantiate a scaffold with the Angular CLI.

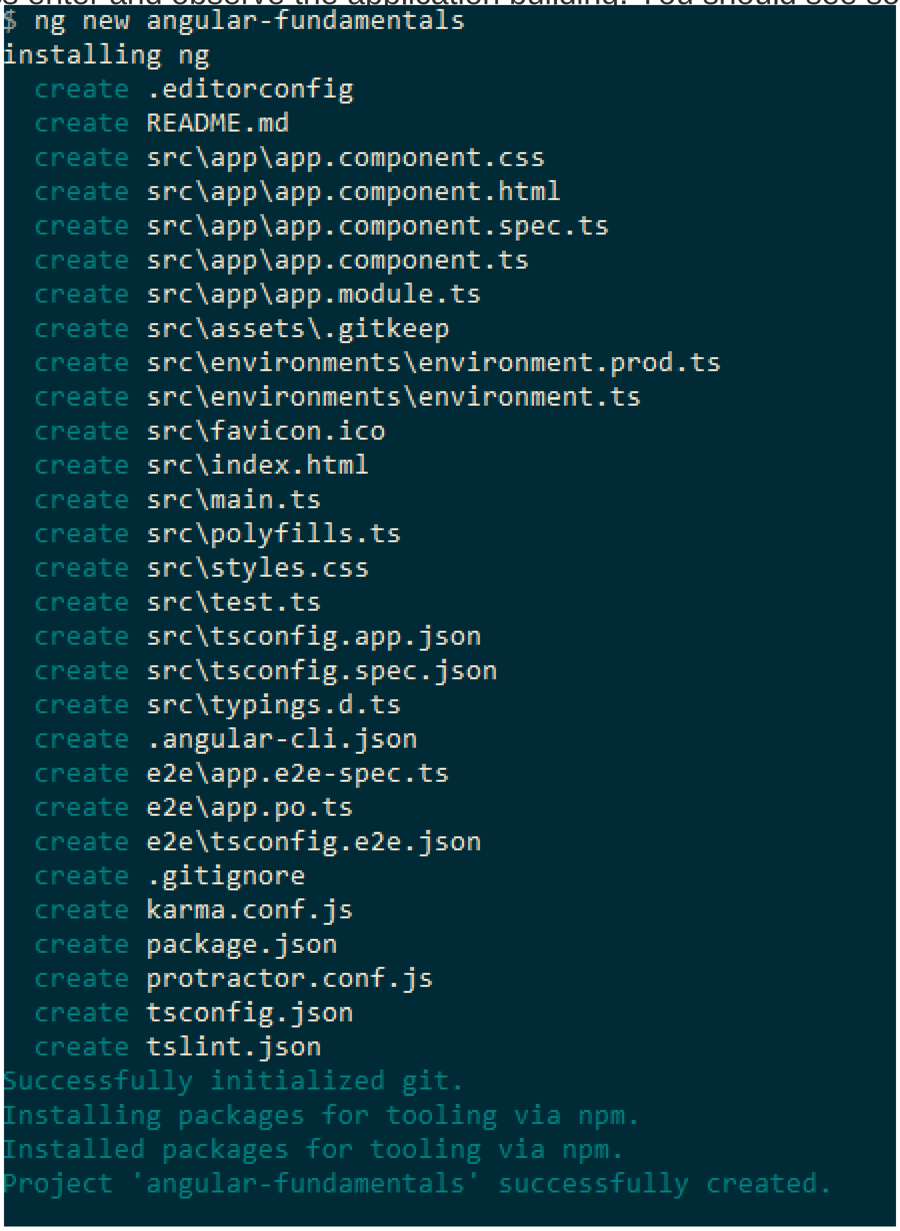
1. First we need a name for our application. It needs to be alphanumeric (excepting that it can use dash characters), but the dashes must be immediately succeeded by a letter. For this application we are going to call it angular-fundamentals.

2. Open your command prompt and navigate to the directory you want to generate your app in. Note that the CLI will generate a subdirectory with the name angular-fundamentals within that directory.

3. Type the following to instantiate your application.

```
ng new angular-fundamentals
```

4. Press enter and observe the application building. You should see something like this:



```
$ ng new angular-fundamentals
installing ng
  create .editorconfig
  create README.md
  create src\app\app.component.css
  create src\app\app.component.html
  create src\app\app.component.spec.ts
  create src\app\app.component.ts
  create src\app\app.module.ts
  create src\assets\.gitkeep
  create src\environments\environment.prod.ts
  create src\environments\environment.ts
  create src\favicon.ico
  create src\index.html
  create src\main.ts
  create src\polyfills.ts
  create src\styles.css
  create src\test.ts
  create src\tsconfig.app.json
  create src\tsconfig.spec.json
  create src\typings.d.ts
  create .angular-cli.json
  create e2e\app.e2e-spec.ts
  create e2e\app.po.ts
  create e2e\tsconfig.e2e.json
  create .gitignore
  create karma.conf.js
  create package.json
  create protractor.conf.js
  create tsconfig.json
  create tslint.json
Successfully initialized git.
Installing packages for tooling via npm.
Installed packages for tooling via npm.
Project 'angular-fundamentals' successfully created.
```

That's it! You now have your application scaffolded. Next, we're going to add an extra component.

Now that we have our app generated using the Angular CLI, we're going to use the Angular CLI's generator feature to generate a module. Using the built in generator command in the Angular CLI allows you to create components of your application easily without having to worry about properly including the code in your application. The Angular CLI generates the files in the appropriate format and modifies your files to include the newly generated component properly. Some of the different generators that the Angular CLI can run are:

- Modules ng g module my-module
- Components ng g component my-component
- Directives ng g directive my-directive
- Pipes ng g pipe my-pipe
- Services ng g service my-service
- Classes ng g class my-class
- Guards ng g guard my-guard
- Interfaces ng g interface my-interface
- Enums ng g enum my-enum

We will expand on many of these later in the course. For now, let's generate a module using the Angular CLI Generator function.

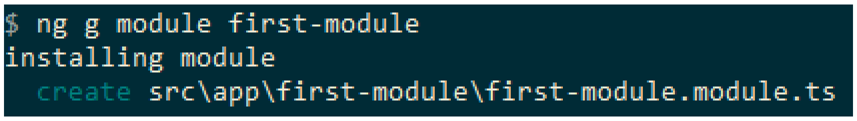
1. Open your command prompt or terminal and navigate to your angular-fundamentals directory that you created in the last section.

2. Run the following command:

   ng g module first-module

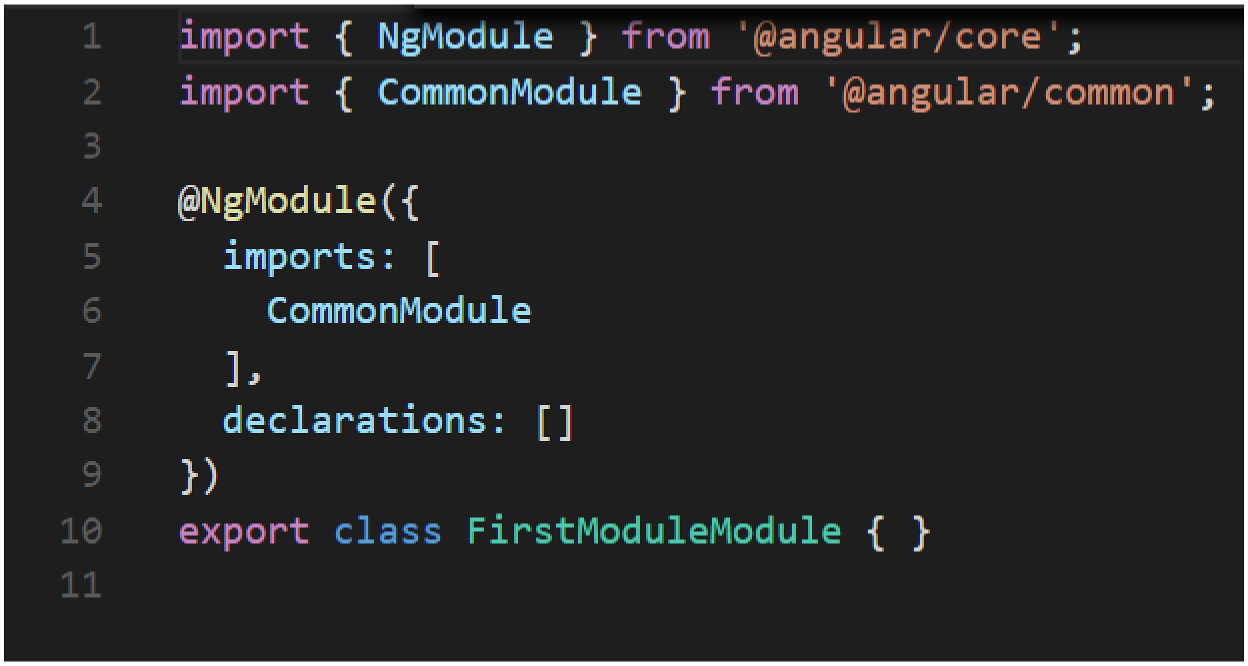
   Note that first-module is the name of the module you are creating.

3. You will see the following:



```
$ ng g module first-module
installing module
  create src\app\first-module\first-module.module.ts
```

4. If you open up the first-module.module.ts file you will see the following:



```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';

@NgModule({
  imports: [
    CommonModule
  ],
  declarations: []
})
export class FirstModuleModule { }

```

NOTE: You will find this file inside the first-module directory under the app directory in the src directory. src\app\first-module

That's it! You've generated a module. Next, we're going to learn how to serve your app using the Angular CLI.

Use the Angular CLI to Run your App Locally

One of the most useful features about using the Angular CLI is the ability to serve your app locally in order to test it in real time. Serving your app compiles your modules for use and runs a server on your computer. You can then access the app via a browser. In addition, the Angular CLI will listen to changes to your files and recompile your app, meaning you can edit code, and then immediately see the changes in your browser.

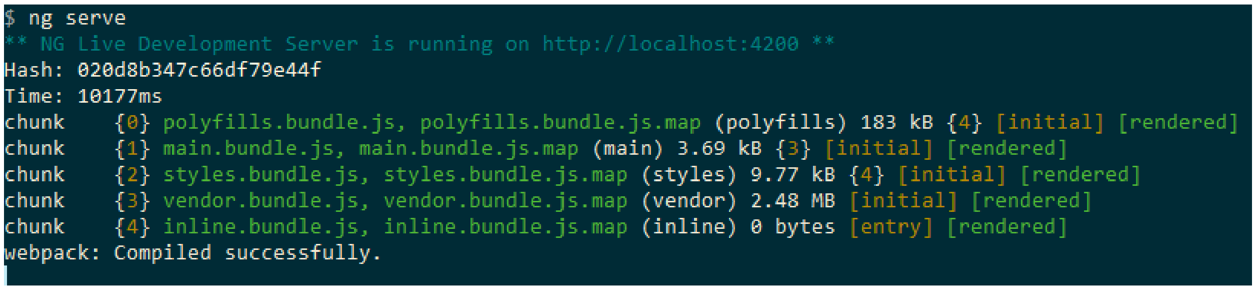
Let's serve the app we made now.

1. Open your command prompt and navigate to the angular-fundamentals app we generated earlier.

2. Type the following:

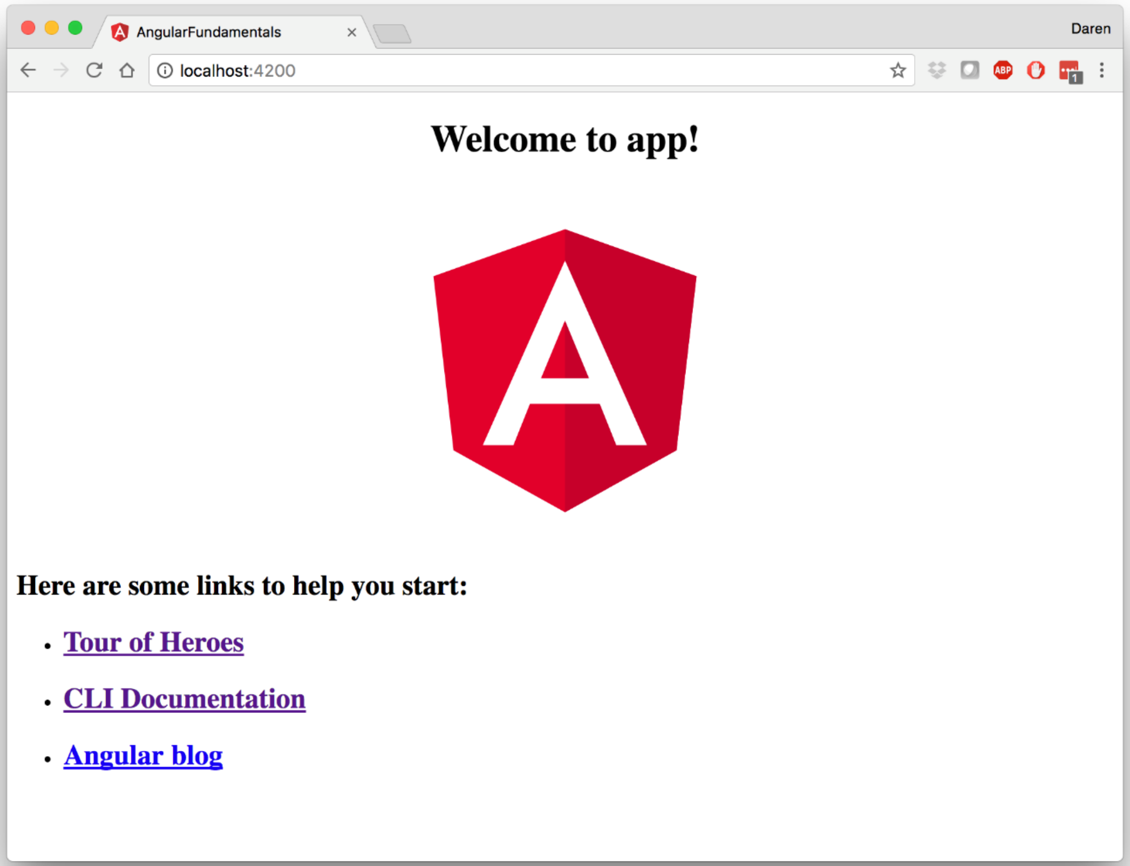
   ng serve

3. You should see the following:



```
$ ng serve
** NG Live Development Server is running on http://localhost:4200 **
Hash: 020d8b347c66df79e44f
Time: 10177ms
chunk    {0} polyfills.bundle.js, polyfills.bundle.js.map (polyfills) 183 kB {4} [initial] [rendered]
chunk    {1} main.bundle.js, main.bundle.js.map (main) 3.69 kB {3} [initial] [rendered]
chunk    {2} styles.bundle.js, styles.bundle.js.map (styles) 9.77 kB {4} [initial] [rendered]
chunk    {3} vendor.bundle.js, vendor.bundle.js.map (vendor) 2.48 MB [initial] [rendered]
chunk    {4} inline.bundle.js, inline.bundle.js.map (inline) 0 bytes [entry] [rendered]
webpack: Compiled successfully.
```

4. Open up a browser and navigate to http://localhost:4200. You should see this:



Welcome to app!

Here are some links to help you start:

- Tour of Heroes
- CLI Documentation
- Angular blog

5. Now, let's make a small change. Open up the file angular-fundamentals/src/app/app.component.ts in Visual Studio Code. Line 9 looks like this:

```
title = 'app';
```

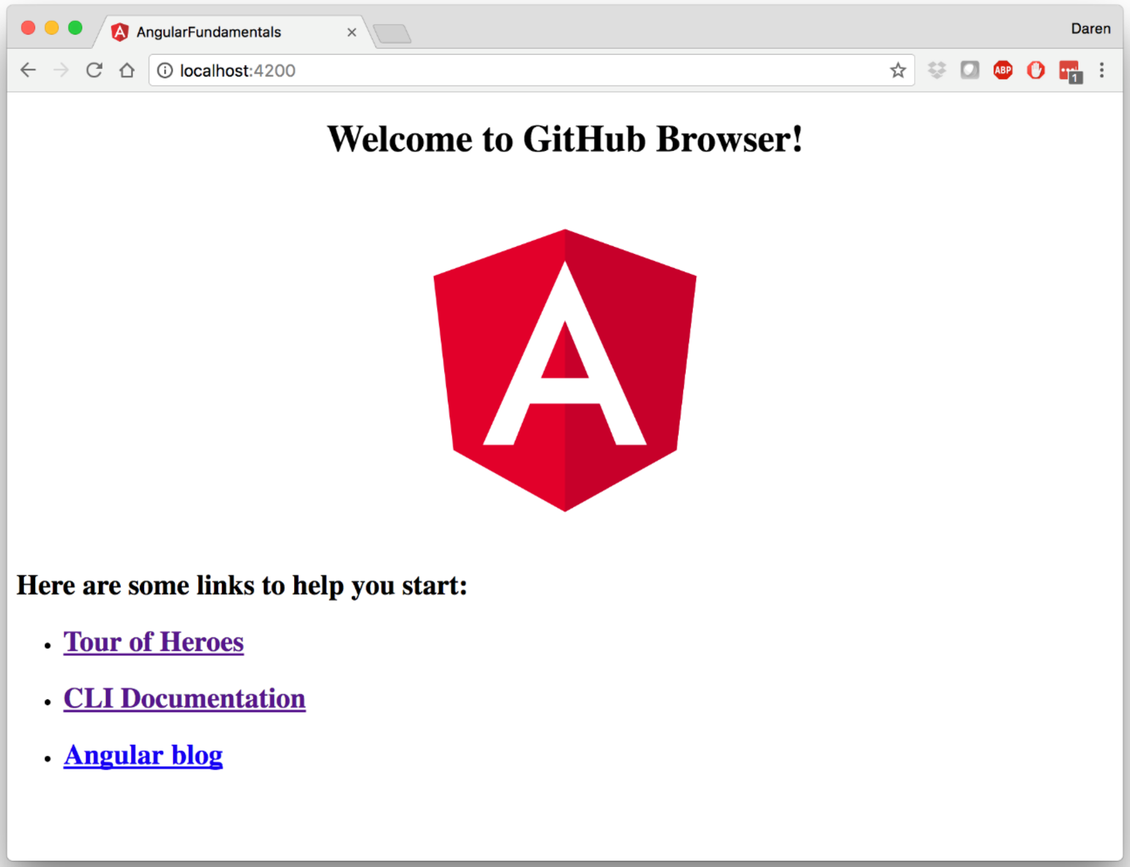
Let's change that to something else.

```
title = 'GitHub Browser';
```

6. Save the file, then go back to your console. You will something similar to the following after the original output:

```
webpack: Compiling...
Hash: 72146afddffbdc3352ac
Time: 416ms
chunk    {0} polyfills.bundle.js, polyfills.bundle.js.map (polyfills) 183 kB {4} [initial]
chunk    {1} main.bundle.js, main.bundle.js.map (main) 3.7 kB {3} [initial] [rendered]
chunk    {2} styles.bundle.js, styles.bundle.js.map (styles) 9.77 kB {4} [initial]
chunk    {3} vendor.bundle.js, vendor.bundle.js.map (vendor) 2.48 MB [initial]
chunk    {4} inline.bundle.js, inline.bundle.js.map (inline) 0 bytes [entry]
webpack: Compiled successfully.
```

7. And when you go back to your browser you will see the change, like this:



Welcome to GitHub Browser!

Here are some links to help you start:

- Tour of Heroes
- CLI Documentation
- Angular blog

You have now generated your first app using the Angular CLI and served it to your browser. In our next module, you will begin to build your app within the scaffold that you have created.