Seneca
SCHOOL OF INFORMATION AND COMMUNICATIONS TECHNOLOGY

BTI425

Web Programming for Apps and Services

Angular introduction, new in-class app

This are the step-by-step instructions for re-creating the new app that we did in class.

Step-by-step instructions

Use two terminal windows. One will enable you to enter shell and Angular CLI commands. The other will enable you to start/run the app.

Generate a new app:

```
ng new XXXXX --routing -S -g
```

Navigate into that folder (in both terminal windows).

Open the app in the Visual Studio Code editor.

Start the app (which will open it in a browser tab/window):

```
ng serve --open
```

After you make and save an editing change, look at it in the browser. In the browser, it will also

be a good idea to open its console.

In the editor, locate and edit the app's entry point, which is `index.html`. Add the Bootstrap 3 CDN link.

Notice the contents of the `body` element. Don't change it, just notice its contained element, named `app-root`.

In the editor, locate and edit the "app root" component files, `app.component.html` (the UI / markup template), and `app.component.ts` (the program code). Replace some content, and notice that the browser-displayed app shows the changes.

Now, we will generate and display a new component. In the terminal, do this:

```
ng g c body --flat -S

# If your ng version is 9 or later, omit -S...
ng g c body --flat
```

Notice the generated files. Let's add that new component to the app root component. (Put it inside a `div`.) Notice that the browser-displayed app shows the changes.

Now, we will generate and display another new component:

```
ng g c io --flat -S
```

Add that component to the app root component, just below the recently-added body component.

Why did we name it "io"? It will do *input* and *output*, otherwise known as data binding.

First, let's do *output*. Locate and edit the component's program code. Declare a class property to hold your name. In TypeScript, this is done like this:

```
myName: String;
```

In the constructor, set its value:

```
this.myName = 'Peter';
```

How do we display the value? Locate and edit the component's markup template.

```
<p>Your name is...</p>
```

```
<p>{{myName}}</p>
```

Notice the double-curly-brace container, which is a bit different than what you've seen in React and in JavaScript string interpolation.

Next, let's do *input* (i.e. getting input from the user). Before continuing, comment out the code in the constructor (so that the displayed name is blank or empty).

Then, we must prepare the app for user input. We will use the Angular "template forms" capability (which you will learn more about later). Locate and open the `app.module.ts` source code file (which has app-wide settings). Near the top, add forms-handling to the app:

```
import { FormsModule } from "@angular/forms";
```

Further down, in the "imports" array, add the forms module, so that it looks like this:

```
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule
  ],
```

Next, return to editing the "io" markup template. Add this new code above the other code. Notice the value of the `[(ngModel)]` attribute - it MUST match the property name in the "io" program code. This technique performs two-way *data binding*.

```
<p>Enter your name:</p>
<p><input [(ngModel)]="myName" type="text"></p>
```

Happy coding!
