**Angular Overview**

Angular is a framework for building modern single-page applications. The official docs and tutorials for Angular can be found at [www.angular.io](http://www.angular.io).

How are single-page applications different then traditional applications?

* Traditional Application
  + A traditional application will have each user action result in a full HTML page load from a server.
* Single-Page Application
  + A single-page application is a web application that is composed of a single page.
  + Based on user actions, the application is updated instead of a full HTML page load.
  + It normally will perform a partial update instead of a full-page re-load.

**Angular History**

I have listed out the timeline below for the Angular history. It is important to note that AngularJS 1.0 is legacy. The Angular team did a complete re-write with Angular 2. So, Angular 2, and above is a completely different framework. Angular 2 and higher follow the same framework approach with incremental improvements.

* Angular was released in October 2010 and the first version was AngularJS 1.0
* Angular 2 was released in October 2016
* Angular 4 was released in March 2017
* Angular 5 was released in November 2017
* Angular 6 was released in May 2018
* Angular 7 was released in October 2018
* Angular 8 was released in May 2019
* Angular 9 was released February 2020

**Getting Started with Angular:**

* Angular Features
  + Component-based framework
  + Clean separation of template coding and application logic
  + Built-in support for databinding and dependency injection
  + Supports responsive web design and modern frameworks
* Key Terms
  + Component
    - Main player in an Angular application. It has two parts:
      * View for user interface
      * Class that contains application logic / event handling for the view
  + View Template
    - The user interface for the component
    - Static HTML with dynamic elements
  + Directive
    - Adds custom behavior to HTML elements
    - Used for looping, conditional and so on
  + Service
    - Helper class that provides desired functionality
    - Retrieving data from a server, performing a calculation, validation and so on
  + Module
    - A collection of related components, directive, services, and so on

**What is Angular CLI?**

* Angular CLI is a command-line interface (CLI) to automate your development workflow. It allows you to: create a new Angular application.

**Installing Angular CLI**

The first thing you will need to do is to pull up the command line and then run the commands below. The first one will install Angular CLI. Then you can run the second one to check to make sure Angular CLI installed correctly.

* npm install -g @angular/cli
* ng version

**Creating a New Project with Angular CLI**

Now we will be creating a new Angular project using Angular CLI. This is like what we did with React, but we will be using the Angular CLI to create a new Angular project. Below are the commands to create a new project.

* ng new <your-project-name>
* ng new my-first-angular-project
* After you run the ng new <your-project-name> command you will be asked as series of questions. I am listing out the questions with the answers beside the question.
  + Would you like to add Angular routing? No
  + Which stylesheet format would you like to use? CSS

**How to run the newly created project**

Ok, first I want to point out that your project will be running on port 4200. So after you run your project you will need to use: <http://localhost:4200> in the browser to locate your application. This is like React where we used <http://localhost:3000>. I have listed out the commands below:

* After creating your new project
  + cd into your project
  + then type out the command: ng serve
    - This will start your application. It might not open in the browser window with your application running. So, you might have to go to <http://localhost:4200>
    - You can also use the command: ng server --open
      * This will run your application and open it in your default browser.
  + Again, your server will be running by default on port 4200
  + Note you can change your server port if you wish by using the command:
    - ng server --port 5100
    - You can use whatever port you wish

**Open your project in Visual Studio Code or whatever you like**

* Open Visual Studio Code
* Locate the file tab in the top left corner and then select it
* Then move down to open folder
* Now locate your new project folder and open it
* You are now ready for Wednesday

**Project Files Overview**

* angular.json
  + Angular workspace configuration
  + List of execution targets
* e2e folder
  + End to End testing
* node\_modules
  + Local repo for node modules
* package.json
  + Project meta data
  + List of node dependencies
  + This is like Mavens pom.xml file
* src
  + Main source code directory
* App Folder
  + App components, template, and so on
* Assets folder
  + Images and so on
* Index.html
  + Main launch page
* Link to Angular file-structure
  + <http://www.angular.io/guide/file-structure>

**Create a new Component**

* Control C to stop your project
* Then run the command:
  + ng generate component <your-new-component>
  + In the demo I created:
    - ng generate component sales-person-list
* Next you will run the command to generate a typescript class for your new component
  + ng generate class <componentName/ClassName>
  + For the demo sales-person-list component this is how I created the class below:
    - ng generate class sales-person-list/SalesPerson

**Development Process for new Sales List Component**

* Generate a new component called sales-person-list
* Add new component selector to the app.component.html template page
* Generate a Salesperson class
  + Add a constructor with parameters.
  + The parameter properties are declared by prefixing the constructor with the public access modifier in the constructor arguments
* In the sales-person-list.component.ts, create some sample data
  + An array of sales person objects
* In the sales-person-list template file, build the the HTML table by looping over the data
  + Use \*ngFor that will loop over the array and create a table row for each array element.
* <h1 class="teamData">Team Data</h1>
* <table>
* <thead>
* <tr>
* <th>First Name</th>
* <th>Last Name</th>
* <th>Email</th>
* <th>Sales Volume</th>
* </tr>
* </thead>
* <tbody>
* <tr \*ngFor="let tempSalesPerson of salesPersonList">
* <td>{{tempSalesPerson.firstName}}</td>
* <td>{{tempSalesPerson.lastName}}</td>
* <td>{{tempSalesPerson.email}}</td>
* <td>{{tempSalesPerson.salesVolume}}</td>
* </tr>
* </tbody>
* </table>

**Add Bootstrap to your project**

* Get links for remote Bootstrap files
* Add links to index.html
* Apply Bootstrap CSS styles in the component HTML file
* Apply Bootstrap CSS styles in component HTML table
* Update TypeScript component file to reference Bootstrap HTML template

**For event binding:**

Below is a link that you can look at for a reference for all the events you can do and then bind them.

* <https://developer.mozilla.org/en-US/docs/Web/Events>

**Link for bootstrap**

You will put this link in the index.html file inside your header.

* <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">