Lab 1 - Understand the Ionic application and learn how to customize it.

In this short lab (10min) we are going to review the Ionic application artifacts and code and then customized it to a customer.

Steps

1. Start a command line terminal (i.e. `cmd` on Windows or `terminal` on OS X and Linux).
2. Change context to the IBMEmployeeApp directory:

   ```
   cd IBMEmployeeApp
   ```

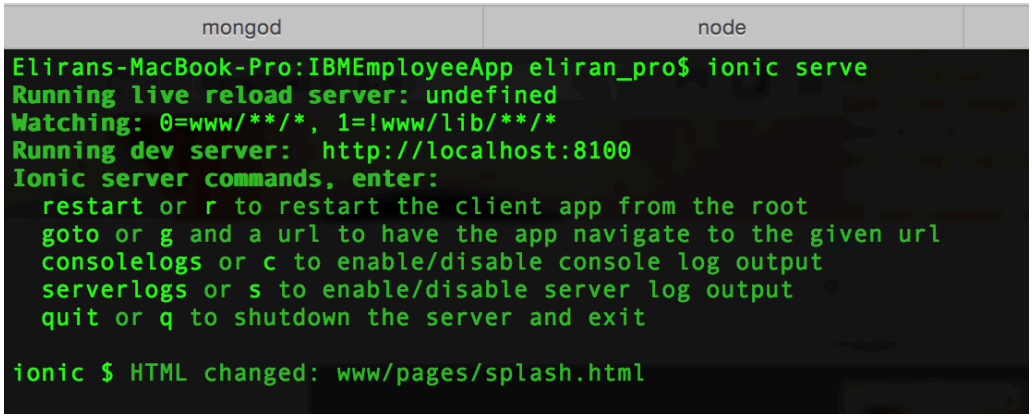
3. In order to start from a known point for the first lab run the following command:

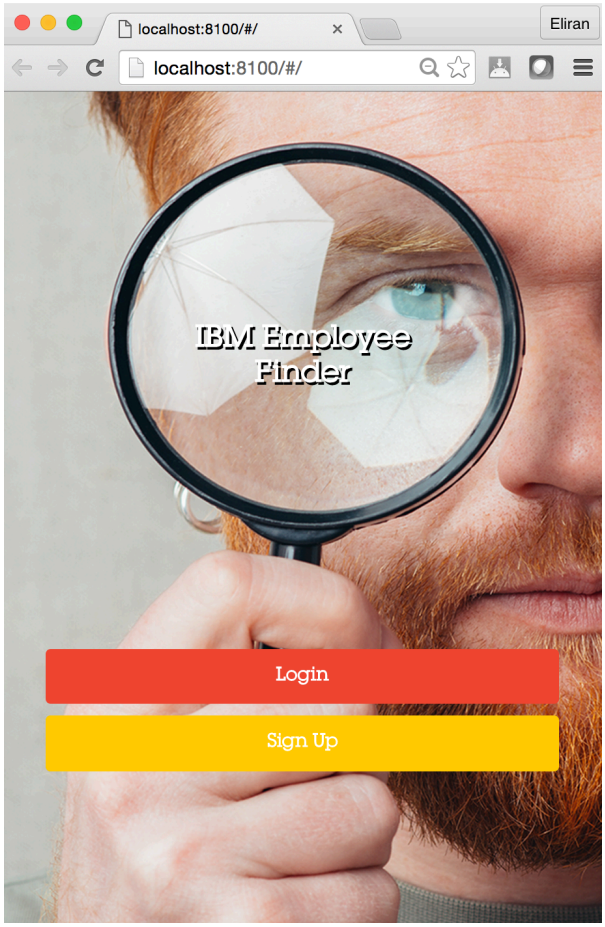
   ```
   git checkout -f step-0
   ```

   (this checkout will also add the **/snippets** and **/extras** folders for use in the editing steps later on)

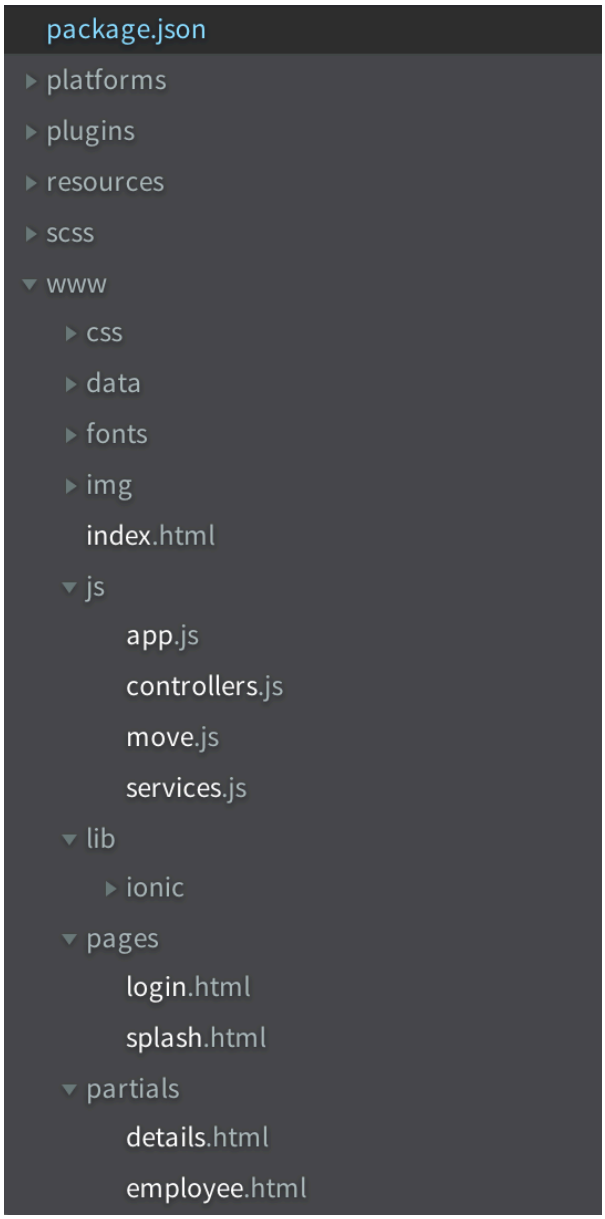
4. Run the application using the Ionic serve command, that will start your default browser and point to **localhost:8100/#/** it will also track your application code change and will reload the browser with your latest code changes.

   ```
   ionic serve
   ```





5. Lets look at the project structure on the left side



- **package.json** - contain all your application dependencies.
- **platfroms directory** - contain the platform that you add to your project (iOS, Android etc)
- **plugins directory** - contain the different cordova plugin you add to your projects.
- **resource directory** - contain the application splash screen and the app icon for the different env (default to Ionic)
- **scss directory** - contain your application color theme definition and will be applied if you using scss.
- **www directory** - contain your application assets like : css files, images, data files, html files, javascript files etc.
- **lib directory** - contain the ionic libs and any additional libs you might add.
- **index.html** - the main html file for our application.
- **app.js** - the main javascript file for our application
- **controllers.js** - hold the controllers we defined for each page.

- **services.js** - hold the different network calls or any other services that your application might use.

6. Lets take a look at the index.html file

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
    <title></title>
    <link href="lib/ionic/css/ionic.css" rel="stylesheet">
    <link href="css/ionicons.css" rel="stylesheet">
    <link href="css/style.css" rel="stylesheet">
    <!-- ionic/angularjs js -->
    <script src="lib/ionic/js/ionic.bundle.js"></script>
    <!-- cordova script (this will be a 404 during development) -->
    <script src="cordova.js"></script>
    <!-- your app's js -->
    <script src="js/app.js"></script>
    <script src="js/controllers.js"></script>
    <script src="js/services.js"></script>
  </head>
  <body ng-app="ibmApp" ng-controller="appCtrl">
      <!--  enable-menu-with-back-views="true" - fix the issue with the side-menu button not showing. -->
      <ion-side-menus enable-menu-with-back-views="true" ng-cloak>
        <!-- right menu -->
        <ion-side-menu side="right">
            <ul class="list menuList">
                <a nav-clear menu-close class="item item-icon-left" ui-sref="splash">
                    <i class="icon ion-key"></i>Home
                </a>
                <a menu-close class="item item-icon-left" ui-sref="main">
                    <i class="icon"></i> Employee List
                </a>
                <a nav-clear menu-close class="item item-icon-left" ui-sref="splash" on-tap="logout()">
                    <i class="icon ion-locked"></i> Logout
                </a>
            </ul>
        </ion-side-menu>
        <ion-side-menu-content>
        <!-- Main content, usually <ion-nav-view>  nav-title-slide-ios7 -->
            <ion-nav-bar class="bar-positive" align-title="center" ng-cloak>
                <ion-nav-back-button class="button-clear" side="primary">
                    <i class="ion-arrow-left-c"></i>Back
                </ion-nav-back-button>
                <ion-nav-buttons side="secondary">
                    <!-- Toggle left side menu -->
                    <button menu-toggle="right" class="button button-icon icon ion-navicon"></button>
                </ion-nav-buttons>
            </ion-nav-bar>
            <ion-nav-view></ion-nav-view>
        </ion-side-menu-content>
      </ion-side-menus>
  </body>
  <!-- animation js -->
  <script src='js/move.js'></script>

```

As you can see we have 3 different sections:

- First part add reference to the Ionic framework css and javascript files along with reference to our custom css style. style.css
- Second part reference to our app.js, controllers.js and services.js which include the application

business logic.

- Third part is the actual bootstrapping of our application we are using the **ng-app** directive to auto bootstrap the app and **ng-controller** directive which attaches a controller class to the view. This is a key aspect of how angular supports the principles behind the Model-View-Controller design pattern.

To learn more you can visit the link below: https://docs.angularjs.org/api/ng/directive/ngApp
https://docs.angularjs.org/api/ng/directive/ngController

Easy to customized

The demo asset was built in a way that will allow you easily customized it to match the most common use cases for you demos or POCs. The splash and the login screen are easy to change, you just need to choose your own background and company logo, and choose the color scheme that match the customer themes. The master detail list can represent anything from list of employee and their details or list of bank accounts and account details, or car insurance policies or insured car and their details. **Take a look at the image below:**

```
<ion-view class="" title="" hide-nav-bar="true" cache-view="false" hide-back-button="true">
  <ion-content scroll="false">
    <div class="item item-image">
         A <!-- Replace with customer/custom background -->
           <img src="img/login/bg7.png" height="730px" class="bgImg">
    </div>
    <div class="col col-center splashMsg" id="splashMsg">
     B <!-- Replace with customer logo -->
       <!--<img ng-src="img/fpl_logo.png">-->
       <!-- Replace with your application name or slogo -->
       <br><p class="splashSlogo">IBM Employee<br>Finder</p>
    </div>
    <center>
      <div class="list overlay box" id="splash-next-box" style="display:none;">
       C <button class="button button-block button-assertive rFont" on-tap="doShowLogin()">Login</button>
         <button class="button button-block button-energized rFont" ui-sref="auth.signup">Sign Up</button>
      </div>
      <div class="list overlay lBox" id="login-box" style="display:none;">
          <div class="loginOpacity">
              <label class="item item-input rFont">
                <input type="text" placeholder="Enter your username" ng-model="user.username" class="rFont">
              </label>
              <label class="item item-input">
                <input type="password" placeholder="Enter your password" ng-model="user.password" class="rFont">
              </label>
          </div>
          <button class="button icon-left ion-locked button-block button-energized rFont rFont-login" type="submit" on-
          tap="doLogin()">Login</button>
          <button class="button button-clear button-light rFont rFront-forgot">Forgot Password?</button>
      </div>
    </center>
  </ion-content>
</ion-view>
```

- To change the **splash/login background** refer to section A
- To change/add **splash/login customer logo** or change the slogo/application name refer to section B
- To change the **buttons color scheme** refer to section C , you can find all the default Ionic color scheme in the link below: http://ionicframework.com/docs/components/#buttons

7. Open the **splash.html** page under /pages/splash.html and customize your splash to the color scheme and the logo/application name of the last customer you have met.

8. If you look again in your browser you will see the ionic automatically apply and load your changes

Summary

You reviewed the Ionic application source code and now you can easily customized the look and feel to match you future POC or custom demos.

If you were unable to complete this lab, you can catch up by running this command:

```
git checkout -f step-1
```