

Cross-platform Using Ionic-09

In this assignment, you will use various technologies to accomplish a starter cross-platform app. We will be using Visual Studio Code as our development environment, along with Cordova for cross-platform plugins, Node.js to help manage the plugins, Angular as our framework, and Ionic for the UI. Visual Studio Code is already installed so we can jump right to installing the dependencies required to build cross-platform apps.

1. First, we will download and install Node.js. We are using Node.js for its package manager which makes it really easy to download/import useful tools and code to use in our app. Go to <https://nodejs.org/en/> and download the LTS version.
2. When the download is complete, run the downloaded setup file (when the setup wizard opens, keep all the default settings and click next until the install starts).
3. When the installation finishes, click the Windows button and find/run Node.js (once you run it you can close the window it opens, this just ensures the Node.js service is running).
4. You are now ready to start setting up the development environment. Click the Windows button in the bottom right of your screen and then find/run Visual Studio Code (VS Code).
5. To install the tools/code for our cross-platform app we will be using commands at the command line (a.k.a. terminal). When VS Code opens, click "View" from the top menu, then "Terminal". This will open a terminal in the bottom of your VS Code development environment where we can type commands.
6. We are going to use Ionic and Cordova to build a cross-platform app. Run the following command in the terminal:

```
npm install -g cordova ionic
```

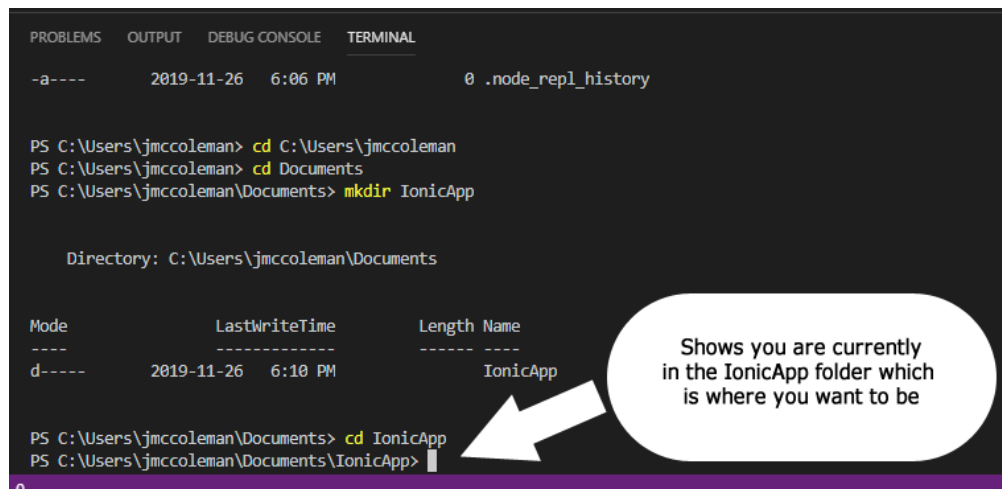
7. Once that is done, we will make a folder to put our app in. You should by default be in your home directory (if you are not, run the command **cd C:\Users\<your username>/student number>**). From your home directory run the following commands:

```
cd Documents  
mkdir IonicApp
```

8. Now let's go into our IonicApp folder by running the command:

```
cd IonicApp
```

9. Your terminal should now look something like this:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
-a----      2019-11-26   6:06 PM           0 .node_repl_history

PS C:\Users\jmccoleman> cd C:\Users\jmccoleman
PS C:\Users\jmccoleman> cd Documents
PS C:\Users\jmccoleman\Documents> mkdir IonicApp

Directory: C:\Users\jmccoleman\Documents

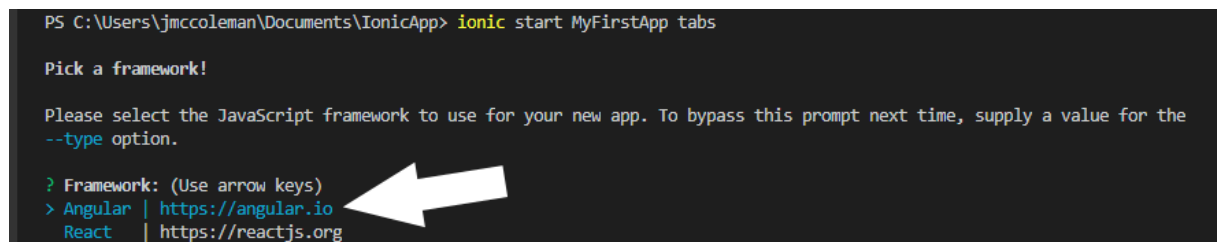
Mode                LastWriteTime         Length Name
----                -
d-----      2019-11-26   6:10 PM             IonicApp

PS C:\Users\jmccoleman\Documents> cd IonicApp
PS C:\Users\jmccoleman\Documents\IonicApp> |
```

10. Now, let's create our actual app. Ionic provides templates to start from which are small apps with a common starting point. We are going to use the tabs template which will create a app with three tabs in it. To do that, run the command

ionic start MyFirstApp tabs

11. The first thing you will have to do is select a JavaScript framework, ensure "Angular" is selected and press the enter key.



```
PS C:\Users\jmccoleman\Documents\IonicApp> ionic start MyFirstApp tabs

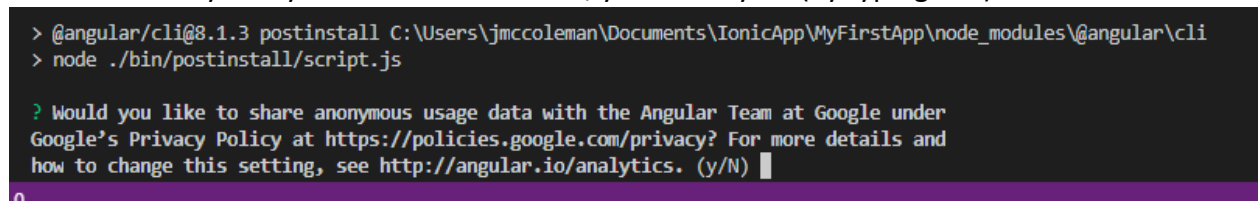
Pick a framework!

Please select the JavaScript framework to use for your new app. To bypass this prompt next time, supply a value for the
--type option.

? Framework: (Use arrow keys)
> Angular | https://angular.io
  React   | https://reactjs.org
```

12. After you select the framework, be patient as it can take some time for everything to get setup.

13. It will then ask you if you want to share data, you can say no (by typing "N")



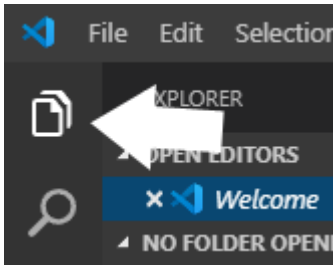
```
> @angular/cli@8.1.3 postinstall C:\Users\jmccoleman\Documents\IonicApp\MyFirstApp\node_modules\@angular\cli
> node ./bin/postinstall/script.js

? Would you like to share anonymous usage data with the Angular Team at Google under
Google's Privacy Policy at https://policies.google.com/privacy? For more details and
how to change this setting, see http://angular.io/analytics. (y/N) |
```

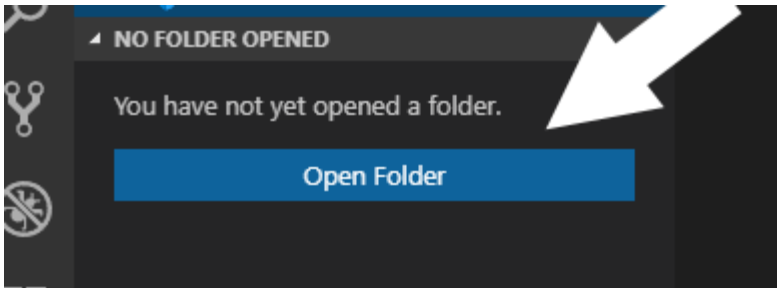
14. When it completes you will have successfully installed a starter app. Let's run it, first go into the directory npm created for you by running the

command cd MyFirstApp

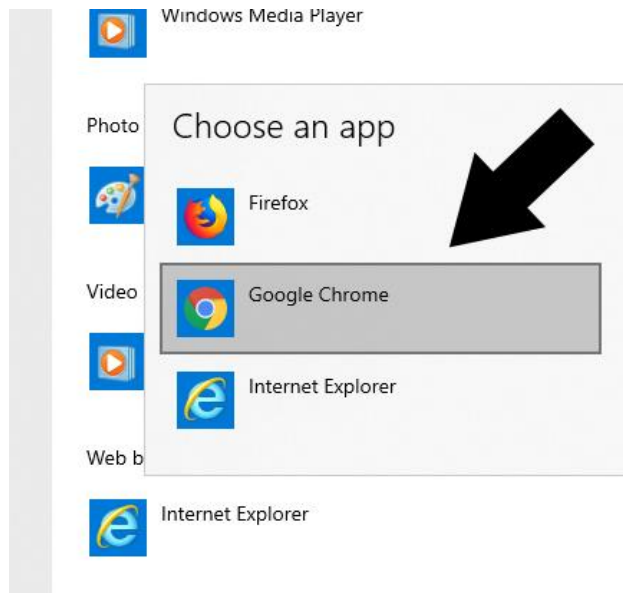
15. Now that our app has been created, let's open it up in VS Code so we can run it and change some code. Click the show folders/files button.



16. Next, click the "Open Folder" button.



17. In the open folder pop up, navigate to your MyFirstApp folder, highlight it, and click "Select Folder". This will open your app files into the file explorer section. There are many files, some you will edit and others you do not have to touch
18. Before we run our app, let's ensure Google Chrome is our default browser. Click the Windows key and type "Default Browser".
19. If you see "Internet Explorer" or "Firefox" under "Web browser", click it and select "Google Chrome"

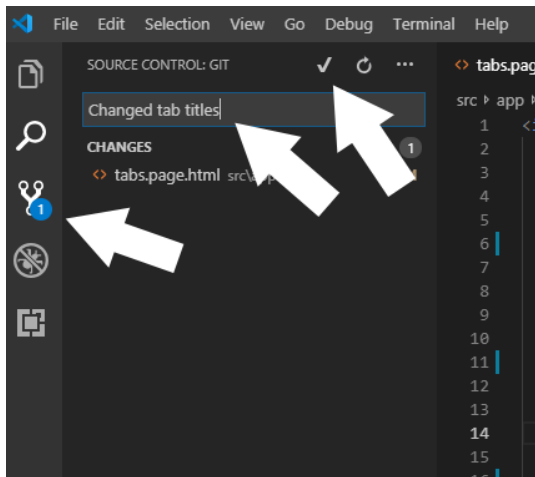


20. Let's run our app. To do that, run the command

ionic serve

21. Your app will open in a web browser (when asked which web browser you want to use, ensure you select Google Chrome). This is a great part of developing using cross-platform apps, you can test it in the browser first (however when you build apps that use device hardware, you sometimes have to run it on a device in order to test everything).
22. Click the tabs at the bottom of your app to see that the starter template we used created an app with three tabs by default. It also added some components/content to your first tab.
23. Let's check out some of the code and change the names of our tabs. Go back to VS Code and in the files/folder section, expand the folder "src", then "app", then "tabs".
24. Open the "tabs.page.html" file and change the name of your tabs (this is done by changing the content between the <ion-label></ion-label> tags).
25. Now save tabs.page.html and go back to your browser. Notice the tab buttons automatically change! If you closed your browser you can run the command **ionic serve** to get your app running again.
26. Time to commit and push!!! First let's stop our app. Go to the terminal section of VS Code and press Ctrl + C. This will stop our app and bring us back to the terminal prompt.
27. Let's go over the GitHub and create a repository for our app and copy the repository URL.

28. Go back to VS Code and click the version control button, enter a message for your commit, then click the check mark to commit.



29. Before we can push we have to add our remote repository. Let's add our remote using the command line. In the terminal windows, run the command

git remote add origin https://github.com/<your repository>

30. Now you will be able to push. Using the source control section, click the three dots at the top right and select "Push".

