ITEC 4000

Christianne Huber

Winter 2016

Selected e-Learning Materials

for

Mobile Application Development

Table of Contents

1. Introduction 4

2. An Overview of the Ionic Framework and Cordova 4

In this Section 4

Recommended Reading: 5

3. Setting up the Development Environment 6

3.1. The Basic Development Environment 6

In this Section 6

Recommended Tutorial 7

Recommended Reading 7

3.2. GitHub 7

In this Section 7

Recommended Reading 9

Recommended Tutorial 9

4. Building a Mobile Application 10

4.1. Building a Mobile Application 10

In this Section 10

Recommended Reading 10

Recommended Tutorials 10

4.2. AngularJS and Organizing the Code 11

In this Section 11

Recommended Reading and Tutorials 12

Additional Tutorials 13

4.3. JSON 13

In this Section 13

Recommended Reading and Tutorials 14

Recommended Tutorial 14

5. The Backend 15

5.1. The Database 15

In this Section 15

Recommended Reading 16

5.2. Node.js - Connecting the Application to the Database 16

In this Section 16

Recommended Reading 17

6. Summary 18

7. Works Cited 19

1. Introduction

The scope of this document is to provide a set of selected electronic learning resources that can be used by students who are assigned to work on a mobile application. The training references provide information, Recommended Reading and tutorials for the technologies used. No prior experience in developing applications for mobile devices is assumed, however, the student is expected to be familiar with HTML, CSS, JavaScript and SQL. It is anticipated that the material can be covered in approximately one month.

2. An Overview of the Ionic Framework and Cordova

In this Section

- Introduction to hybrid mobile applications
- The benefits of creating a hybrid mobile application
- The Ionic framework and the benefits of using a framework

Coding native applications for each device type would be time consuming and difficult to maintain. Although a web application runs on a server and could be accessed by multiple devices, it would only be available when there is an Internet connection and the application would not have access to the device application programming interface (API) (Mayoral Baños). A hybrid application can be considered to be a native application with a browser inside – the browser renders the HTML5 while the JavaScript and plugins provide access to the device APIs (Mayoral Baños).

In the e-book, *Mobile Development for Web Developers*, Joshua Morony states that HTML5 results in a browser-based application, however, such applications do not have access to the device application programming interface (6-7). He explains a hybrid application as one that uses a native "wrapper" around an HTML application to provide access to the device application programming interfaces (9-12).

The Ionic framework is "targeted at building hybrid mobile apps" (Drifty Co, Welcome). A framework speeds up development by providing templates that can be used throughout the application, making it easier to maintain a consistent look (Drifty Co, Welcome). It uses Cordova as the wrapper and AngularJS for the functionality (Drifty Co, Welcome).

The Apache Cordova Architecture illustrates how an HTML5, CSS and JavaScript hybrid application can access the device components such as the camera, geolocation and local storage via the Cordova native API and plugins (The Apache Software Foundation).

Recommended Reading:

- http://ionicframework.com/docs/guide/preface.html (10 mins)
- http://blog.ionic.io/where-does-the-ionic-framework-fit-in/?_ga=1.266675467.428823918.1480993116 (10 mins)
- *Mobile Development for Web Developers* by Joshua Morony, pages 6 – 14, (pages 14 – 17 optional), pages 19-22
- http://cordova.apache.org/docs/en/latest/guide/overview/index.html (15 mins)

3. Setting up the Development Environment

3.1. The Basic Development Environment

In this Section

- Pre-requisites for Ionic
- Installation instructions
- Checklist for installation

The Ionic website provides all the installation instructions (Drifty Co, Installing). Make sure to review the “Linux Android note” or the “Windows note on Java, Ant and Android” if you are using Linux or Windows operating systems (Drifty Co, Installing)!

The prerequisites are Node.js and Cordova (Drifty Co, Installing). Cordova requires the Android Software Development Kit (Android SDK) for developing Android applications and the Android SKD requires the Java Development Kit (JDK). The FAQ page has a link to an excellent YouTube tutorial (Drifty Co, Android SDK) which demonstrates the installation procedure on the Windows operating system (Ionic).

Here is a checklist for the requirements as provided in the YouTube tutorial (Ionic):

1. Install the JDK
2. Install Apache Ant
3. Install Android SDK

   Notes:

   i. The instructions also work on Windows 10

   ii. Re: Android Distributions (Ionic 6:08)**:** Students should be advised to check what distributions are required as they can be very large!

4. Install Node.js
5. Install Apache Cordova
6. Install Ionic

Recommended Tutorial

- https://www.youtube.com/watch?v=RNrNIHQ9cWo

Recommended Reading

- http://ionicframework.com/docs/guide/installation.html (5 mins)

  Note: The "Platform Notes" in this Installation Guide suggest using "Git for Windows and optionally Console2" (Drifty Co, Installing). These are both optional as the Windows Command Prompt can be used instead.
- "Requirements and Support" and "Installing the Requirements" from the Android Platform Guide at http://cordova.apache.org/docs/en/latest/guide/platforms/android/index.html

3.2. GitHub

In this Section

- GIT and GitHub
- Setting up an account on GitHub
- Git workflow

- Creating branches/ “forking”
- Submitting pull requests and uploading/ “pushing” changes
- Checklist of process steps

According to Daniele Procida et al, “Git is a source code management system, designed to support collaboration” and “GitHub is a web-based service that hosts Git projects” (Procida et al). Lauren Orsini describes Git as “version control software” (Orsini, Part1).

Orsini provides instructions beginning with setting up an account on GitHub (Orsini, Part1) through to uploading (“pushing”) the changes to the master repository in the second part of her tutorial (Orsini, Part 2). However, she does not mention “branches” or how to manage them. Roger Dudler briefly describes the “workflow” and the difference between the “Working Directory”, the “Index” (staging area) and the “Head” (Dudler). The tutorial by Procida et al introduces the concept of “forking” to create a new branch and “making a Pull Request” (Procida et al). **Note**: I would suggest reading through all the Recommended Reading material before attempting the tutorials.

The GitHub repository for application may already have been created. When the developers prepare to work on a new set of updates, they will do the following:

1. Create a local folder for the application code within the Ionic folder (if it does not yet exist)
2. Initialize the folder as a Git folder (if it is not yet a Git folder)
3. Create a branch off the master in the remote repository

4. Clone his/her branch to the folder on his/her computer, making this the local master
5. Checkout a branch from the local master as the working branch
6. Once all the updates have been completed and tested, proceed as instructed; **either**:
    a. Commit and push to the **local** master so that changes can be pushed to the **developer's remote** branch, in preparation for updates to be merged into the remote master from there **or:**
    b. Commit and push to the **developer's remote working branch** and issue a pull request from there, then merge the upstream changes into the developer's remote master (Procida et al, "Git on the commandline").

Recommended Reading

- http://readwrite.com/2013/09/30/understanding-github-a-journey-for-beginners-part-1/
- http://readwrite.com/2013/10/02/github-for-beginners-part-2/
- http://rogerdudler.github.io/git-guide/

Recommended Tutorial

- http://dont-be-afraid-to-commit.readthedocs.io/en/latest/git/git.html
- http://dont-be-afraid-to-commit.readthedocs.io/en/latest/git/commandlinegit.html
- http://dont-be-afraid-to-commit.readthedocs.io/en/latest/git/remotes.html

4. Building a Mobile Application

4.1. Building a Mobile Application

In this Section

- Tutorial to build your first mobile application
- Testing the application
- Using the browser Developer Tools

The Ionic Framework Book (Drifty Co) shows how to create an application (Drifty Co, Installing), add menus to the application (Drifty Co, Starting), test it (Drifty Co, Testing), then add content to the menus (Drifty Co, Building). It is beneficial to have an understanding of AngularJS when completing this section of the tutorial. (Refer to 4.2 AngularJS and Organizing the Code).

When testing the application in the desktop browser, it is helpful to view the Developer Tools. These can be opened from the browser options available from the top right corner of the browser, just below the button to close the window: e.g. in Google Chrome, select More Tools >> Developer Tools (Mayoral Baños, 14 Oct 2016). In Microsoft Edge, select F12 Developer Tools.

Recommended Reading

- Ionic Book: http://ionicframework.com/docs/guide/

Recommended Tutorials

- http://ionicframework.com/docs/guide/starting.html

- http://ionicframework.com/docs/guide/testing.html
- http://ionicframework.com/docs/guide/building.html
- http://ionicframework.com/docs/guide/publishing.html. **Note**: This is optional since if working on a development team, only one person will be publishing the application.

4.2. AngularJS and Organizing the Code

In this Section

- AngularJS directives, controllers and services
- Organizing AngularJS code into modules

AngularJS handles the navigation between the different views within the application, as well as any logic and data that is stored in variables (Conceptual Overview). Having an understanding of the AngularJS concepts will help distinguish between directives, controllers and services. It will also help to learn in which files to place the code. The AngularJS tutorials available on the W3Schools website are a good place to start, since they are quick and easy to read. Each step can be tested in the "Try it Yourself" editor to see how a change in the AngularJS script is displayed in the HTML.

The Ionic templates automatically create multiple separate files for the AngularJS code. Although this style is mentioned in the W3Schools tutorials (Refnes, AngularJS Modules; Refnes, AngularJS Controllers), most of the examples either call one JavaScript file or have the AngularJS code with the HTML. Morony shows how to use different "modules" for organizing the code in his tutorial (46, 47).

Recommended Reading and Tutorials

- The AngularJS tutorials available on the W3Schools website http://www.w3schools.com/angular/default.asp. Proceed from the "Angular JS Tutorial" home in the link above, through the following:
  - "AngularJS Introduction" https://www.w3schools.com/angular/angular_intro.asp
  - "AngularJS Expressions" https://www.w3schools.com/angular/angular_expressions.asp
  - "AngularJS Modules" https://www.w3schools.com/angular/angular_modules.asp
  - "AngularJS Directives" https://www.w3schools.com/angular/angular_directives.asp
  - "AngularJS ng-model Directive" https://www.w3schools.com/angular/angular_model.asp
  - "AngularJS Data Binding" https://www.w3schools.com/angular/angular_databinding.asp
  - "AngularJS Controllers" https://www.w3schools.com/angular/angular_controllers.asp
  - "AngularJS Scope" https://www.w3schools.com/angular/angular_scopes.asp
  - "AngularJS Services" https://www.w3schools.com/angular/angular_services.asp and

  - "AngularJS AJAX -$http"

    https://www.w3schools.com/angular/angular_http.asp

- "A first example: Data binding", "Adding UI Logic: Controllers", "View-independent business logic: Services" in https://docs.angularjs.org/guide/concepts
- Some more information about scope is provided at the following link:
  - https://github.com/angular/angular.js/wiki/Understanding-Scopes

Additional Tutorials

- *Mobile Development for Web Developers* by Joshua Morony
  - Pages 39-49 and pages 50-56 to build the interface for the next tutorial
  - Pages 58-73 to create a list of photos and access them through data binding with the controllers

4.3. JSON

In this Section

- JSON for data transfer
- JSON tutorial
- JSON to access data in a mobile application

In "JSON vs XML", the W3Schools tutorial states that whereas they "both can be parsed and used by lots of programming languages" (Refsnes Data), JSON is quicker, easier to use and easier to parse (Refsnes Data).   The tutorial provides fundamental information on JSON, such as syntax, data types and conversion to/from a String, as well as how to send, receive, parse and store data with JSON. The tutorial is rather quick and can be completed in as little as an hour and a half.

Learning the syntax in the W3Schools tutorial will help with understanding the code in Morony's tutorial as you continue to develop the application to allow a user to add, display and delete pictures from the mobile device. Morony shows how the application uses Cordova to access the files and to store them locally on the mobile device (74-87). He also introduces JavaScript Object Notation (JSON) (77, 78).

Recommended Reading and Tutorials

- The JSON tutorials available on the W3Schools website http://www.w3schools.com/js/js_json_intro.asp. Proceed from the "JSON – Introduction" in the link above, through to the following:
  - "JSON Syntax" https://www.w3schools.com/js/js_json_objects.asp
  - "JSON Data Types" https://www.w3schools.com/js/js_json_datatypes.asp
  - "JSON Objects" https://www.w3schools.com/js/js_json_objects.asp
  - "JSON Arrays" https://www.w3schools.com/js/js_json_arrays.asp
  - "JSON.parse()" https://www.w3schools.com/js/js_json_parse.asp and
  - "JSON.stringify()" https://www.w3schools.com/js/js_json_stringify.asp

Recommended Tutorial

- *Mobile Development for Web Developers* by Joshua Morony
  - Pages 74-87 cover accessing the local storage.

5. The Backend

So far, the data that has been used in the tutorials has been stored locally in the device (Morony 77). The illustration of the architecture of an AngularJS mobile application in Williamson et al shows that although there is local storage, it connects to a remote source via JSON/HTTPs (Chapter 4 - Mobile Application Development Figure 4.2) and typically, the data is stored in a database (Chapter 5 - Mobile Enterprise - Beyond the Mobile End-Point). Therefore, a mobile application sends requests for data in JSON over HTTP to a web server which acts as a controller to access the data in a backend database.

5.1. The Database

In this Section

- NoSQL databases
- NoSQL database data models and DBMS
- Why NoSQL databases are becoming more popular

In “NoSQL Databases: An Overview”, Pramod Sadalage discusses why NoSQL databases are now being used. He illustrates the data models and provides some examples of the databases that are available (Sadalage). He states that the “impedance mismatch” and the need to store large quantities of data on clusters are two main reasons why NoSQL databases have become more popular (Sadalage). According to Hans Marggraff, the ability to store complex objects is one of the reasons for using a NoSQL database. “If you come from an SQL background you probably wonder, what the hype is about” (Marggraff).

Recommended Reading

- "Chapter 4 - Mobile Application Development". *Enterprise Class Mobile Application Development: A Complete Lifecycle Approach for Producing Mobile Apps* by Leigh Williamson et al.
- "Chapter 5 - Mobile Enterprise - Beyond the Mobile End-Point". *Enterprise Class Mobile Application Development: A Complete Lifecycle Approach for Producing Mobile Apps* by Leigh Williamson et al.
- http://searchdatamanagement.techtarget.com/definition/NoSQL-Not-Only-SQL
- https://www.thoughtworks.com/insights/blog/nosql-databases-overview
- http://www.reportsanywhere.com/pebble/2010/06/25/a_short_explanation_of_the_nosql_data_model.html

5.2. Node.js - Connecting the Application to the Database

In this Section

- AngularJS client, Node.js server
- Function passing, anonymous functions, asynchronous callbacks

Node.js is server-side code (Maddygoround). It acts as the Controller in the Model-View-Controller architecture. The AngularJS client and Node.js server side code is implemented as shown in "Call function in nodejs from angular application" (Basti1302; Spoike). This is a key difference between where the AngularJS code and Node.js code reside.

Manuel Kiessling has an excellent tutorial on creating a simple "Hello World" HTTP server in Node.js. In this tutorial, he explains function passing, anonymous functions, and asynchronous callbacks (Kiessling). These techniques are important since they are used throughout mobile applications. **Note**: Should you wish to learn more about setting up a web server to handle HTTP requests, the Node.js website also has some excellent tutorials on this topic.

Recommended Reading

- http://www.nodebeginner.org/
- http://stackoverflow.com/questions/32515171/common-method-for-calling-a-node-function-from-angular-controller
- http://stackoverflow.com/questions/18327039/call-function-in-nodejs-from-angular-application
- https://nodejs.org/dist/latest-v7.x/docs/api/synopsis.html (optional)
- https://nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/ (optional)

6. Summary

A mobile application makes use of several technologies. However, it can be described simply, as a Model-View-Controller system (Williamson et al, Chapter 4 - Mobile Application Development). The Ionic Framework is used to simplify the effort to create a consistent look for the user interface, throughout the application (Drifty Co, Welcome). As illustrated in Williamson et al, the AngularJS ngModel, ngDirectives and ngController provide the “MVC stack” (Chapter 4 - Mobile Application Development, Figure 4.2). AngularJS manages the view, stores data locally in the model, and controls the data as the user navigates through the application. The mobile application uses JSON to send and receive data to/from the mobile backend which controls access to the data services (Williamson, Chapter 4 - Mobile Application Development). The data services are typically NoSQL databases (Williamson, Chapter 5 - Mobile Enterprise - Beyond the Mobile End-Point), which store data “in a form suitable for rapid access by mobile apps” (Williamson, Chapter 4 - Mobile Application Development).

7. Works Cited

Basti1302. "Call function in nodejs from angular application." 20 Aug 2013. *StackOverflow.* Web. Accessed 22 Jan 2017. <stackoverflow.com/questions/18327039/call-function-in-nodejs-from-angular-application>.

Bradley, Adam. "Where does the Ionic Framework fit in?" 28 Oct 2013. *Ionic.* Web. Accessed 09 Dec 2016. <blog.ionic.io/where-does-the-ionic-framework-fit-in/?_ga=1.266675467.428823918.1480993116>.

Conceptual Overview. 2010-2017. *Angular JS.* Web. Accessed 27 Dec 2016. <docs.angularjs.org/guide/concepts>.

Drifty Co. "Building out your app." 2013-2016. *Ionic v1.* Web. Accessed 09 Dec 2016. <ionicframework.com/docs/guide/building.html>.

—. "Common Issues and FAQs." 2013-2016. *Ionic v1.* Web. <ionicframework.com/docs/ionic-cli-faq/#android-sdk>.

—. "Installing Ionic and its Dependencies." 2013 - 2016. *Ionic v1.* Web. Accessed 10 Dec 2016. <ionicframework.com/docs/guide/installation.html>.

—. *Ionic Book*. 2013-2016. Web. <ionicframework.com/docs/guide/>.

—. "Starting your app." 2013-2016. *Ionic v1.* Web. Accessed 10 Dec 2016. <ionicframework.com/docs/guide/starting.html>.

—. "Testing your app." 2013-2016. *Ionic v1.* Web. <ionicframework.com/docs/guide/testing.html>.

—. "Welcome to Ionic." 2013-2016. *Ionic v1.* Web. <ionicframework.com/docs/guide/preface.html>.

Dudler, Roger, et al. *git - the simple guide*. n.d. Web. Accessed 17 Dec 2016. <rogerdudler.github.io/git-guide/>.

Ionic. *Installing Android and Cordova dependencies for Ionic Development*. 30 Jun 2014. Web. <www.youtube.com/watch?v=RNrNIHQ9cWo>.

Kiessling, Manuel. "The Node Beginner Book." n.d. Web. Accessed 14 Feb 2017. <www.nodebeginner.org/>.

Kumar Garg, Gandharv, et al. "Understanding Scopes." 10 Oct 2016. *GitHub.* Web. Accessed 16 Dec 2016. <github.com/angular/angular.js/wiki/Understanding-Scopes>.

Maddygoround. "Common method for calling a node function from Angular controller." 11 Sep 2015. *Stackoverflow.* Web. Accessed 22 Jan 2017. <stackoverflow.com/questions/32515171/common-method-for-calling-a-node-function-from-angular-controller>.

Marggraff, Hans. "A Short Explanation of the NoSQL Data Model." 25 Jun 2010. *Reports Anywhere Articles.* Web. Accessed 09 Jan 2017. <http://www.reportsanywhere.com/pebble/2010/06/25/a_short_explanation_of_the_nosql_data_model.html>.

Mayoral Baños, Alejandro. York University, Toronto, 21 Sep 2016. Meeting.

—. York University, Toronto, 14 Oct 2016. Meeting.

Morony, Joshua. *Mobile Development for Web Developers*. n.d.

Node.js Foundation. "Anatomy of an HTTP Transaction." 2017. *Node.js.* Web. Accessed 25 Jan 2017. <nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/>.

"Node.js v 7.7.1 Documentation." n.d. *Node.js.* Web. Accessed 24 Jan 2017. <nodejs.org/dist/latest-v7.x/docs/api/synopsis.html>.

Orsini, Lauren. "GitHub for Beginners: Commit, Push And Go." 02 Oct 2013. *Adwrite.* Web. Accessed 08 Nov 2016. <readwrite.com/2013/10/02/github-for-beginners-part-2/>.

—. "GitHub for Beginners: Don't Get Scared, Get Started." 30 Sep 2013. *Adwrite.* Web. Accessed 08 Nov 2016. <readwrite.com/2013/09/30/understanding-github-a-journey-for-beginners-part-1/>.

Procida, Daniele, et al. "Git and GitHub." 2013. *Don't be afraid to commit.* Web. Accessed 24 Feb 2017. <dont-be-afraid-to-commit.readthedocs.io/en/latest/git/index.html>.

—. "Git on the commandline." 2013. *Don't be afraid to commit.* Web. Accessed 24 Feb 2017. <dont-be-afraid-to-commit.readthedocs.io/en/latest/git/commandlinegit.html>.

—. "Working with remotes." 2013. *Don't be afraid to commit.* Web. Accessed 24 Feb 2017. <dont-be-afraid-to-commit.readthedocs.io/en/latest/git/remotes.html>.

Refsnes Data. "AngularJS AJAX - $http." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_http.asp>.

—. "AngularJS Controllers." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_controllers.asp>.

—. "AngularJS Data Binding." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_databinding.asp>.

—. "AngularJS Directives." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_directives.asp>.

—. "AngularJS Expressions." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_expressions.asp>.

—. "AngularJS Introduction." 1999-2017. *W3Schools.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_intro.asp>.

—. "AngularJS Modules." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_modules.asp>.

—. "AngularJS ng-model Directive." 1999-2016. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_model.asp>.

—. "AngularJS Scope." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/angular/angular_scopes.asp>.

—. "AngularJS Services." 1999-2017. *W3Schools.com.* Web. Accessed 10 Dec 2016. <www.w3schools.com/angular/angular_services.asp>.

—. *AngularJS Tutorial*. 1999-2017. Web. Accessed Dec 2016. <www.w3schools.com/angular/default.asp>.

—. *JSON - Introduction*. 1999-2017. Web. Accessed 06 Dec 2016. <www.w3schools.com/js/js_json_intro.asp>.

—. "JSON Arrays." 1999-2017. *W3Schools.* Web. Accessed Dec 2016. <www.w3schools.com/js/js_json_arrays.asp>.

—. "JSON Data Types." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/js/js_json_datatypes.asp>.

—. "JSON Objects." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/js/js_json_objects.asp>.

—. "JSON Syntax." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/js/js_json_syntax.asp>.

—. "JSON vs XML." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/js/js_json_xml.asp>.

—. "JSON.parse()." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/js/js_json_parse.asp>.

—. "JSON.stringify()." 1999-2017. *W3Schools.com.* Web. Accessed Dec 2016. <www.w3schools.com/js/js_json_stringify.asp>.

Rouse, Margaret and Barney Beal. "NoSQL (Not Only SQL Database)." Oct 2011. *TechTarget.* Web. Accessed 09 Jan 2017. <searchdatamanagement.techtarget.com/definition/NoSQL-Not-Only-SQL>.

Sadalage, Pramod. "NoSQL Databases: An Overview." 02 Oct 2014. *Thoughtworks.* Web. Accessed 09 Jan 2017. <www.thoughtworks.com/insights/blog/nosql-databases-overview>.

Spoike. "Call function in nodejs from angular application." 20 Aug 2013. *StackOverflow.* Web. Accessed 22 Jan 2017. <stackoverflow.com/questions/18327039/call-function-in-nodejs-from-angular-application>.

The Apache Software Foundation. "Android Platform Guide." 2012, 2013, 2015. *Cordova.* Web. <cordova.apache.org/docs/en/latest/guide/platforms/android/index.html>.

—. *Overview*. 2012, 2013, 2015. Web. 2016. <cordova.apache.org/docs/en/latest/guide/overview/index.html>.

Webruster. "How to use two way data binding ionic & angularjs." 22 Mar 2016, 9:44. *Stackoverflow.* Web. Accessed 16 Dec 2016. <stackoverflow.com/questions/36148565/how-to-use-the-two-way-data-binding-ionic-angularjs>.

Williamson, Leigh, et al. "Chapter 4 - Mobile Application Development." *Enterprise Class Mobile Application Development: A Complete Lifecycle Approach for Producing Mobile Apps*. IBM Press, 2016. Web. Accessed 02 Dec 2016. <common.books24x7.com.ezproxy.library.yorku.ca/toc.aspx?bookid=104316>.

Williamson, Leigh, et al. "Chapter 5 - Mobile Enterprise - Beyond the Mobile End-Point." *Enterprise Class Mobile Application Development: A Complete Lifecycle Approach for Producing Mobile Apps.* IBM Press, 2016. Web. Accessed 03 Dec 2016. <common.books24x7.com.ezproxy.library.yorku.ca/toc.aspx?bookid=104316>.