Comp 322/422 - Software Development for Wireless and Mobile Devices

Fall Semester 2018 - Week 1

Dr Nick Hayward

Course Details

Lecturer

- Name: Dr Nick Hayward
- Office: Doyle 307 (LSC)
- Office hours
  - *Tuesday, Thursday afternoon by appointment (WTC or LSC)*
- Faculty Page

Course Schedule

Important dates for this semester

- Project outline and mockup - presentation & demo
  - *25th September 2018 @ 2.30pm*
  - *27th September 2018 @ 2.30pm*
- Mid-semester break
  - ***n.b.*** *no formal class: 9th October 2018*
- DEV week: 23rd to 30th October 2018
- DEV week - presentation & demo
  - *30th October 2018 @ 2.30pm*
  - *1st November 2018 @ 2.30pm*
- Thanksgiving break: 21st to 25th November 2018
  - ***n.b.*** *no formal class: 21st November 2018*
- Final classes: 4th & 6th December 2018
- Final presentation & demo
  - *4th December 2018 @ 2.30pm*
  - *6th December 2018 @ 2.30pm*
- Exam week: 10th December to 15th December 2018
  - *Final assessment due on 13th December 2018 by 2.30pm*

Initial Course Plan - Part I

Up to ~ DEV Week

- begin development of a mobile application
  - *group project up to and including DEV week*
- cross-platform options
  - *Apache Cordova & Onsen UI &c.*
  - *React Native*
- UI and UX design considerations for mobile apps
- prototypes and tests
- data store options and usage
- API usage and integration
- lots of examples...

Initial Course Plan - Part 2

DEV Week to the end of the semester

- continue to develop your app concept and prototypes
  - *group project augmented and improved from DEV week*
- continue React Native
  - *more examples and advanced options*
- consider recent mobile development options
  - *Flutter by Google*
- comparison of mobile development options
  - *hybrid vs React Native vs native SDK*
  - *Swift and iOS*
- many options for mobile development...

Assignments and Coursework

Course will include

- weekly bibliography and reading (where applicable)
- weekly notes, examples, extras...

Coursework will include

- exercises and discussions (Total = 20%)
  - *various individual or group exercises and discussions*
- project outline & mockup (Total = 15%)
  - *brief group presentation of initial concept and mockup*
- DEV week assessment (Total = 25%)
  - *DEV week: 23rd to 30th October 2018*
  - *demo due 30th October or 1st November 2018 @ 2.30pm*
- end of semester final assessment (Total = 40%)
  - *demo due 4th or 6th December 2018 @ 2.30pm*
  - *report due 13th December 2018 @ 2.30pm*

Exercises & discussions

Course total = 20%

- exercises
  - *help develop course project*
  - *test course knowledge at each stage*
  - *get feedback on project work*
- discussions
  - *sample mobile applications, games, services...*
  - *design topics, UI and UX concepts*
- extras
  - *code and application reviews*
  - *various other assessments*
  - *peer review of demos*

Development and Project Assessment

Course total = 80% (Parts 1, 2 and 3 combined)

Initial overview

- combination project work
  - *part 1 = project outline & mockup (15%)*
  - *part 2 = DEV Week development & demo (25%)*
  - *part 3 = final demo and report (40%)*
- group project (max. 4 persons per group)
- development of a mobile application
  - ***NOT** a responsive website viewed on a mobile device*
  - *Apache Cordova, React Native, Flutter...*
- purpose, scope, and target audience is group's choice
  - ***no** to-do lists, note-taking, flashlights &c.*
  - *chosen project topic needs approval*
  - *data store, APIs, underlying structure &c. is group's choice...*

Project outline & mockup assessment

Course total = 15%

- begin outline and design of a mobile application
  - *built from scratch*
  - ***NOT** a responsive website viewed on a mobile device*
  - *builds upon examples, technology outlined during first part of semester*
  - *purpose, scope &c. is group's choice*
  - *chosen topic requires approval*
  - *presentation should include mockup designs and concepts*

Project mockup demo

Assessment will include the following:

- brief presentation or demonstration of current project work
  - *~ 5 to 10 minutes per group*
  - *analysis of work conducted so far*
  - *presentation and demonstration*
  - *outline current state of mobile app concept and design*
  - *show prototypes and designs*

DEV week assessment

Course total = 25%

- begin development of a mobile application from scratch
  - ***NOT*** *a responsive website viewed on a mobile device*
  - *must apply technologies taught up to and including DEV week, e.g.*
  - *Apache Cordova, React Native, &c.*
  - *combine technologies taught to fit your mobile app...*
- can be platform agnostic (cross-platform) or specific targeted OS, e.g.
  - *cross-platform app that builds for Android and iOS*
  - *targeted build for Android or iOS*
  - *consider choice, and explain why?*
- outline concept, research conducted to date
- consider applicable design patterns
- are you using any sensors etc?
  - *how, why?*
- prototyping
  - *demo current prototypes*
  - *any working tests or models etc*
- anything else to help explain your mobile app...

DEV Week Demo

DEV week assessment will include the following:

- brief presentation or demonstration of current project work
  - *~ 10 minutes per group*
  - *analysis of work conducted so far*
  - *e.g. during semester & DEV week*
  - *presentation and demonstration...*
  - *outline mobile app*
  - *show prototypes and designs*
  - *explain what does & does not work*
  - *...*

Final Assessment

Course total = 40%

- continue to develop your app concept and prototypes
  - *develop application using any of the technologies taught during the course*
  - *again, combine technologies to best fit your mobile app*
- produce a working app
  - *as far as possible try to create a fully working app*
  - *explain any parts of the app not working...*
- explain choice of technologies for mobile app development
  - *e.g. data stores, APIs, modules, &c.*
- explain design decisions
  - *outline what you chose and why?*
  - *what else did you consider, and then omit? (again, why?)*
- which concepts could you abstract for easy porting to other platform/OS?
- describe patterns used in design of UI and interaction

Goals of the course

An overview and demonstration of building cross-platform applications for mobile and wireless devices.

Course will provide

- guide to developing and implementing mobile applications from scratch
- UI and UX design for mobile apps
- cross-platform design and development
  - *using Apache Cordova & UI options, React Native, Flutter...*
- best practices and guidelines for cross-platform development
- outline of example mobile design patterns
- comparisons with native SDKs and development
- guide to deploying and publishing final mobile app
- ...

Course resources - part 1

Website

Course website is available at https://csteach422.github.io

- timetable
- course overview
- course blog
- weekly assignments & coursework
- bibliography
- links & resources
- notes & material

NO Sakai

Course resources - part 2

GitHub

Course repositories available at https://github.com/csteach422

- weekly notes
- examples
- source code (where applicable)

Trello group

Group for weekly assignments, DEV week posts, &c.

- Trello group - COMP 422
  - *https://trello.com/csteach422*

Slack group

Group for class communication, weekly discussions, questions, &c.

- Slack group - COMP 422 - 2018
  - *https://csteach422-2018.slack.com/*

Group projects

- add project details to course's Trello group, *COMP 422 - 2018 @ LUC*
  - *Week 1 - Project Details*
  - *https://trello.com/b/044XRD8x*
- create channels on Slack for group communication
- start working on an idea for your project
- start planning weekly development up to *Project Outline & Mockup*
  - *demo on Tuesday 25th & Thursday 27th September 2018 @ 2.30pm*

A few questions...

What is mobile?

- what exactly do we mean by **mobile**?
- may seem like a simple question to answer
  - *do we categorise mobile based on the OS*
  - *is it Android, iOS, Windows Phone...*
- where do we draw the line for software development?
- 2010 - Mark Zuckerberg & mobile computing
  - *'iPad is not a mobile device, it is a computer...'*
- Wired Magazine - Defining Mobile

Merging technologies

- merging of technology and traditional environments and interactions
  - *definition of mobile will alter and update as well*
- will we perceive in-car devices as mobile?
  - *e.g. touchscreen panels and consoles*
  - *same as phones, tablets?*
- these differences are important
  - *they help us consider designs, UIs, interactions*
  - *different motivations for development*
- currently best to consider *mobile* relative to OS
  - *e.g. associated with phones and tablets*

Usage stats

- usage stats are also v.interesting for developers
- e.g. many users now use smartphones for less frivolous activities, including
  - *62% have used their smartphones to query information about their health or a medical condition*
  - *57% have used their smartphones to complete online banking*
  - *44% have used their smartphones to search real estate listings or other housing information*
  - *43% searched for job listings and availability*
  - *40% to view and check government listings and information*
  - *30% to take an online course or class*
  - *18% to actually submit a job application*

Video - Android Go

Source - YouTube - Android Go

Different types of mobile

- we need to be clear about the differences between mobile types
  - *mobile web (now includes Progressive Web Apps - PWA)*
  - *native mobile*
  - *hybrid mobile*
- each has its place in mobile development
- each has its own particular advantages and disadvantages

Mobile web

- apps viewed and run using a web browser
  - *usually, but not exclusively, a mobile device web browser*
- designed as *responsive* web apps or sites
  - *new generation of progressive apps becoming available*
- in this context *responsive* understood as adaptive views
  - *enables correct rendering on different resolutions of mobile and tablet devices*
- apps normally require user to be online with active data connection
- not true mobile apps
  - *may reflect same look and feel as native mobile OS app*
- apps not uploaded to mobile app stores
- unable to interact at the native, low-level of the mobile OS
- Progessive Web Apps (PWAs)
  - *help to reduce many non-native issues with web apps*
  - *Google Developers - Progressive Web Apps*

Native mobile

- native mobile app development originally perceived as *real deal*
  - *rightly or wrongly dependent upon your perspective*
- development of apps using SDKs and APIs for specific mobile OS
  - *Java for Android*
  - *Swift (& Objective C) for iOS*
  - *C# & .Net for Windows 10 Universal Platform*
- learn and develop different SDK &c. for each native OS
- developer will need to implement code and logic for each platform
  - *both mobile OS implementation and desktop development*
- issue with modified app design and logic
  - *need to meet requirements and restrictions*
  - *limits imposed by each mobile OS...*

Hybrid mobile - Part I

- hybrid mobile apps share a lot with native mobile apps
  - *e.g. characteristics, design traits, functionality*
- however, they are developed using different tools, technologies, methods...
- many options, including
  - *Apache Cordova*
  - *React Native*
  - *Flutter by Google*
- Apache Cordova apps developed using common web technologies
  - *HTML5 (HyperText Markup Language)*
  - *CSS (Cascading Style Sheet)*
  - *JS (JavaScript)*
    - many supported libraries & frameworks
    - many options for UI design and development

Hybrid mobile - Part 2

- attempt to leverage ease and speed of development
  - *due to web technologies*
  - *larger developer base for web development*
- and power of native functionality and hardware
  - *using plugins*
- benefit compared to native mobile
  - *option to use same code base for single app*
  - *same code across multiple mobile OSs*
- inherent benefit and grace of web stack for mobile app development
  - *ability to code once, run across multiple mobile platforms*
- still need to make changes to port an app from platform to platform
  - *often minor and trivial changes*
  - *in particular when compared with native OS development*
- other benefit is use of same languages across multiple platforms
  - *until development of custom plugins...*

Summary of options

Here is a useful table summarising your options for mobile development.

Technology	App Store	Technologies	Cross-platform	Native support	Performance (best practices)
Mobile web	No	HTML, CSS, & JS	Yes	Partial at best	Very good (most of the time)
Hybrid	Yes	HTML, CSS, JS, & APIs	Yes (modifications)	Full (using plugins)	Very good to Excellent (depends on task)
Native	Yes	Native SDK & APIs	No (requires porting)	Full	Very good to Excellent (depends on developer)

Cross-platform - intro

- inexorable rise in popularity of mobile devices
  - *rise in number of mobile OSs*
  - *each competing for market space*
  - *in particular in the consumer space*
- each OS offers similar options and features
- many mobile OS options, including
  - *Android*
  - *iOS*
  - *Windows 10 Universal platform*
  - *LG webOS*
  - *BlackBerry 10*
  - *Firefox OS*
  - *Fire OS*
  - *various OSs for wearables*
  - *...*
- some persist, others are now specialised or deprecated...

Cross-platform - issues and concerns

- mobile market largely dominated by big two
  - *Android and iOS*
- reduced field still introduces issues and concerns for developers
- each mobile OS implements their own
  - *SDK (software development kit)*
  - *API (application program/programming interface)*
- similarities exist but
  - *they use different programming languages*
  - *whilst achieving the same end goals*
  - *Java or Kotlin for Android & Swift (Objective-C) for iOS*
- each mobile OS has its own peculiarities
  - *differing design philosophies &c.*

Cross-platform - common issues and solutions

- common issues might include
  - *permissions*
  - *access to underlying services within an OS*
  - *e.g. SMS rights and logic for different mobile OSs*
- cross-platform alternatives allows us consider unified development environment
  - *access and harness native device*
  - *leverage native functionality, performance, features...*
- leverage common tools and technologies
  - *HTML5, CSS, JavaScript - Apache Cordova and React Native*
  - *C# and Xamarin*
  - *create easier cross-platform apps*

Image - Apache Cordova

Source - Apache Cordova

Apache Cordova - what can it do?

- designed to offer a simple, powerful set of API calls
  - *calls to JavaScript functions*
  - *functions map native OS code to plugins and code in Cordova*
  - *enables access to core functionality for a device*
- allows us to transfer, manipulate, control
  - *data and resources from the native OS and device*
  - *moves it to the web view in our Cordova app*
- allows us to provide same user experience as native app
  - *minus a few base caveats*
- cross-platform support

Apache Cordova - platform support

support has included following mobile OSs,

- Android
- iOS
- Windows 10 Universal platform
- Ubuntu
- LG webOS
- BlackBerry 10
- ...

now supports,

- Android
- iOS
- OS X
- Windows 10

Apache Cordova - functionality and plugins

- allows us to create native mobile applications using a set of common web technologies
  - *including HTML5, CSS, and JavaScript*
- a set of JavaScript APIs
  - *provides access to natively built core plugins*
- currently offers many core APIs
- includes some of the following native functionality,
  - *access the device's microphone for recording &c.*
  - *capture photos using the device's camera*
  - *photo retrieval from the OSs gallery/photo album*
  - *retrieve device information*
    - locale
    - various sensors such as motion, location, connection information, compass...
  - *retrieve device data, contact information...*
  - *process files from/to storage*
  - ...

Apache Cordova - documentation and APIs

official *Cordova* API documentation is currently available at the following URL,

- Apache Cordova API
- Apache Cordova GitHub
- Android API
- iOS API
- ... & many others

Apache Cordova - why choose it?

- potential to develop once, re-use with ease
- Cordova helps us solve some of the following mobile development issues,

1. different programming languages for different mobile OSs
   - *different programming philosophies, conventions, best practices, guidelines...*
2. unique problems inherent to each given mobile OS
   - *e.g. handling and routing SMS requests, data storage, privacy features...*
3. developing, testing, and maintaining applications across multiple mobile platforms
4. ...

Apache Cordova - overview of APIs

Platform APIs (cordova-plugin)	Android	iOS	Windows Universal Platform
BatteryStatus (battery-status)	Yes	Yes	Yes
Camera (camera)	Yes	Yes	Yes
Capture (media-capture)	Yes	Yes	Yes
Connection (network-information)	Yes	Yes	Yes
Device (device)	Yes	Yes	Yes
Events	Yes	Yes	Yes
File (file)	Yes	Yes	Yes
Geolocation (geolocation)	Yes	Yes	Yes
Globalization	Yes	Yes	Yes
InAppBrowser (inappbrowser)	Yes	Yes	Yes
Media (media)	Yes	Yes	Yes
Notification (dialogs)	Yes	Yes	Yes
Splashscreen (splashscreen)	Yes	Yes	Yes
Status Bar (statusbar)	Yes	Yes	Yes
Storage	Yes	Yes	Yes (localStorage & indexedDB)
Vibration (vibration)	Yes	Yes	Yes

Apache Cordova Documentation - Platform Support

Apache Cordova - API details

- many of these mobile native function APIs self explanatory
- a few examples,
- **capture**
  - *record various media directly from the native device*
- **connection**
  - *provides information about the device's wifi and cellular connections*
- **device**
  - *provides useful information on a device's hardware and software*
  - *native device model, the current platform and its version...*
- **events**
  - *particularly important, and useful, API*
  - *e.g.* `deviceready`, `backbutton`, `batterystatus`, *volume events...*
- **file** api to help process device files
- receive the device's location using GPS or network signals
- many more...
- Apache Cordova Documentation - Platform Support

Cordova App - intro

- working with Cordova - start building basic app from scratch
- helps introduce initial concepts underpinning Cordova app development
- look at some of the initial tools we'll be using
  - *to create, run, and deploy our applications*
- focusing on working with Cordova relative to Android and its native SDK
- setting up and configuring our Android development environment
- installing and configuring Apache Cordova, Node.js...
- getting familiar with the Cordova CLI (command line interface)
- develop and test some code

Cordova App - getting started

- initial focus will fall upon working with the CLI for Cordova
- Cordova CLI helps us create, develop, build, and test our first Cordova application
- Cordova CLI enables us to create our new project/s
- helps us build and target deployment on OSs such as Android, iOS, and Windows...

Cordova App - Android - SDK setup

- setup and configure Cordova development environment
- reference point is the Cordova Documentation section on **Platform Guides**
  - *Cordova Platform Guides*
- start work with the Android SDK
  - *download, install, and setup JDK*
  - *Java - JDK*
- Android SDK options include Android Stand-Alone SDK Tools or the recent Android Studio 3.1.4
- use **Android Studio** to help build custom Android plugins for Cordova...
- **Android Stand-Alone SDK Tools** sufficient to build and deploy an Android app
  - *command-line and Studio will both provide necessary SDK, build tools...*
  - *SDK Manager - https://developer.android.com/studio/command-line/sdkmanager*

Cordova App - Node.js Setup

- installed, setup, and configured Android SDK, and associated virtual device
- we also need to install and setup the **Cordova CLI**
- a few additional tools required including
  - *Node.js*
  - *Git*
- Node.js provides access to Node's JavaScript library and NPM
  - *NPM - Node Package Manager*
- Git is used by Cordova CLI to install various assets for new projects

Cordova App - CLI setup

- setup and configured Android SDK, initial Android Virtual Devices, Node.js, Git client...
- now ready to install and setup the **Cordova CLI** itself using **NPM**
- *OS X*

```
npm install -g cordova
```

- *Windows*

```
C:\>npm install -g cordova
```

- –g flag tells NPM to set package, Cordova as global
- default NPM behaviour is to install the package in working directory
- if necessary, update system's **PATH**
- *OS X (Unix)*

```
/usr/local/share/npm
```

- *Windows*

```
C:\Users\username\AppData\Roaming\npm
```

- test Cordova install, e.g. version 8.0.0

```
cordova -v
```

Cordova App - iOS Cross-platform

- start by installing the latest version of Xcode
  - *available from Mac App Store*
  - *follow install prompts*
  - *install additional components*
- check Xcode command line tools are installed

```
xcode-select --install
```

- then update required packages for Cordova and iOS development
  - *update `ios-deploy` if needed for the local system*

```
npm install -g ios-deploy
```

- update `ios-sim` if needed for the local system

```
npm install -g ios-sim
```

- then follow Cordova patterns for creating a new app

Cordova App - development paths - intro

Since the advent of version 3.0 of Apache Cordova, it's been possible to develop apps using two distinct workflows and paths. These workflows include,

- Cross-platform CLI
- Platform-centric

Cordova App - development paths - cross-platform CLI workflow

- CLI allows us to develop cross-platform apps
- offers a broad, wide-ranging collection of mobile OSs
  - *support for many native devices*
- workflow focused upon the `cordova` utility
  - *become known as the Cordova CLI*
- CLI is considered a high-level tool
  - *designed to abstract cross-platform development considerations*
  - *designed to abstract functionality of lower-level shell scripts*

e.g.

```
cordova platform add android --save
```

Cordova App - development paths - platform-centric workflow

- *platform-centric* focuses development on one native platform
  - *choose required native SDK - Android, iOS...*
- augment Cordova app with native components developed in the SDK
- workflow has been tailored for each platform
- relies on a set of lower level shell scripts
- features a custom plugin utility designed to help us apply plugins

Cordova App - development paths - careful choice

- start Cordova development using the *cross-platform* workflow
  - *using CLI*
- then option to migrate to the platform-centric workflow
  - *useful for lower-level, specific OS targeted apps*
- one way from CLI to *platform-centric*
  - *CLI keeps a common set of cross-platform source code*
  - *uses this code to ensure broad support and compliance per build*
  - *overrides any platform-specific modifications and code*
- lower-level specific apps should use *platform-centric* shell tools

Cordova App - creating a new project

- basic outline for creating a template for our project is as follows

```
cordova create basic com.example.basic 422Basic
cd basic
cordova platform add android --save
cordova build
```

- then launch this new Cordova project application, e.g. in the emulator

```
cordova emulate android
```

Cordova App - anatomy of a template - part I

```
cordova create basic com.example.basic 422Basic
```

- first parameter of this represents the path of our project
- creating a new directory in the current working directory called **basic**
- second and third parameters are initially optional
- helps to define at least the third parameter
  - *the visible name of the project,* `422Basic`
- edit either of the last two parameters in the projects `config.xml` file
- at the root level of our newly created project

Cordova App - anatomy of a template - part 2

- new project includes the following default structure
- default parts for our development...

```
config.xml
hooks
  - README.md
package.json
platforms
  - android
  - platforms.json
plugins
  - android.json
  - cordova-plugin-whitelist
  - fetch.json
res
  - icon
  - screen
www
  - css
  - img
  - index.html
  - js
```

- initially, our main focus will be the www directory

Cordova App - anatomy of a template - part 3

- the `www` directory will be the initial primary focus
- three primary child directories
  - *css*
  - *img*
  - *js*
- important `index.html` file at the root level
- three primary files, which include
  - *`index.css`*
  - *`index.js`*
  - *`logo.png`*
- `config.xml` file stores configuration settings for the application

Cordova App - anatomy of a template - part 4

- three important directories to help manipulate and configure our Cordova application
  - *platforms*
  - *plugins*
  - *hooks*
- `platforms` includes an application's currently supported native platforms
  - *e.g. Android*
- `plugins` includes all of the application's used plugins
- `hooks` contains a set of scripts used to customise commands in Cordova
  - *customise scripts that execute before or after a Cordova command runs*

References

- Android API
- Android SDK Manager
- Apache Cordova API
- Apache Cordova Documentation - Platform Support
- Apache Cordova GitHub
- Carmody, Tim., *Fighting Words: Defining "Mobile" and "Computer"* Wired. 11.08.2010. http://www.wired.com/2010/11/fighting-words-defining-mobile-and-computer/
- Google Developers - Progressive Web Apps
- iOS API
- Node.js
- Git