Comp 388/422 - Software Development for Wireless and Mobile Devices

Fall Semester 2015 - Week 1

Dr Nick Hayward

Course Details

Lecturer

- Name: Dr Nick Hayward
- Office: 316 Loyola Hall (LSC) & 531 Lewis Towers (WTC)
- Office hours
    - *Thursday afternoon by appointment (WTC)*
    - *Friday afternoon by appointment (LSC)*
- Faculty Page

TA

- Name: Tyler Bobella
- Email: tbobella@luc.edu

Course Schedule

Important dates for this semester

- Friday @ 2.45pm to 5.15pm (5pm with no break)
  - *Cuneo Hall, Room 117, LSC*
- Labor Day weekend: 4th to 7th September 2015
  - *No class: 4th September 2015*
- DEV week: 5th to 9th October 2015
  - *No class: 9th October 2015*
  - *Project Part 1*
  - *Demo due 16th October 2015 @ 2.45pm*
- Thanksgiving break: 25th to 28th November 2015
  - *No class: 27th November 2015*
- Final class: 4th December 2015
  - *Project Part 2*
  - *Demonstration of final assessment @ 2.45pm*
- Exam week: 7th December to 12th December 2015
  - *Final assessment report due 11th December 2015 by 5.15pm*

Initial Course Plan - Part I

Up to Week 7 - 9th October 2015

- Build a cross-platform mobile application from scratch
    - ***not*** *a responsive website*
    - *runs natively on local device*
        - Android, iOS, Windows Phone...
    - *developed using Apache Cordova & applicable UI (JQuery Mobile)*
    - *access device features*
        - camera, sound, geolocation...
    - *examine Cordova API*
    - *consider design patterns, examples...*

Initial Course Plan - Part 2

Up to Week 16 - 11th December 2015

- Continue and improve upon initial development
- Develop custom plugins for Cordova
    - *choose plugin for preferred device platform (Android, iOS etc)*
- Testing and feature prototyping
- Complete Cordova app
- Consider Android and iOS native development
    - *explore SDK*
    - *build test app*
- IF TIME
    - *combining Cordova with AngularJS - using Ionic*
    - *consider alternatives such as React Native*
    - *beyond phones, tablets etc*
    - *wearables, IoT...*

Assignments and Coursework

Course will include

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

Coursework will include

- quizzes or group exercises at the end of each section (Total = 30%)
    - *based on course notes, reading, and examples*
- mid-semester assessment (Total = 30%)
    - *end of DEV week*
    - *demo due*
- end of semester assessment (Total = 40%)
    - *demo due 4th December 2015 @ 2.45pm*
    - *report due 11th December 2015 @ 5.15pm*

Quizzes, group exercises...

Course total = 30%

- at least one week notice before quiz
    - *average time ~30 minutes (can be extended...)*
    - *taken towards the end of class*
- group exercises
    - *help develop course project*
    - *test course knowledge at each stage*
    - *get feedback on project work*

Development and Project Assessment

Course total = 70% (Parts 1 and 2 combined)

Initial overview

- project developed throughout semester
    - *part 1 includes DEV week (30%)*
    - *part 2 is from DEV week to final assessment (40%)*
- development can be individual or group (max 5 persons per group)
- design and develop a cross-platform mobile application
    - *develop using Apache Cordova and UI (JQuery Mobile...)*
    - *may use Ionic as well*
    - *purpose, scope, and target audience is group's choice*
    - ***no** to-do lists, note-taking, flashlights etc*
    - *chosen project topic needs approval*
    - *data, structure etc is group's choice...*

DEV Week Assessment

- cross-platform mobile app from scratch
    - *can be basic demo of intended final app*
    - *build using Apache Cordova and UI (JQuery Mobile...)*
- demo and project report
    - *week 8 - 16th October 2015*
    - *app assessed for functionality, implementation of Cordova API, design, aesthetics...*

Final Assessment

- continued development of DEV week project
  - *must work, ie: I need to be able to test and use the application*
- need to develop and implement custom plugin for Cordova
  - *what, how, why is the user interacting with your app?*
  - *clearly explain how and why you developed this plugin*
- how did you respond to DEV week feedback?
- outline design choices and influences
- presentation can be a live demo, video, storyboard...
  - *week 15 - 4th December 2015*
- final report
  - *due week 16 - 11th December @ 5.15pm*

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
- cross-platform design and development
- 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

Website

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

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

GitHub

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

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

Getting started

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 Wired magazine interview with Mark Zuckerberg
    - *iPad is not a mobile device, it is a computer*

Video - iPad not mobile

funnylog.kr - "iPad isn't mobile~ It's a computer" by MarkZ...

Source - YouTube - iPad isn't 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?
    - *eg: 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
    - *eg: associated with phones and tablets*

Mobile considerations

- surge in popularity for mobile devices, apps
  - *associated interactions and usage patterns*
- concept of **mobile first** entered broader lexicon
  - *developers and designers think in terms of* ***mobile first***
- encouraged to think in terms of mobile use cases, scenarios...
- think beyond standard desktop app or website

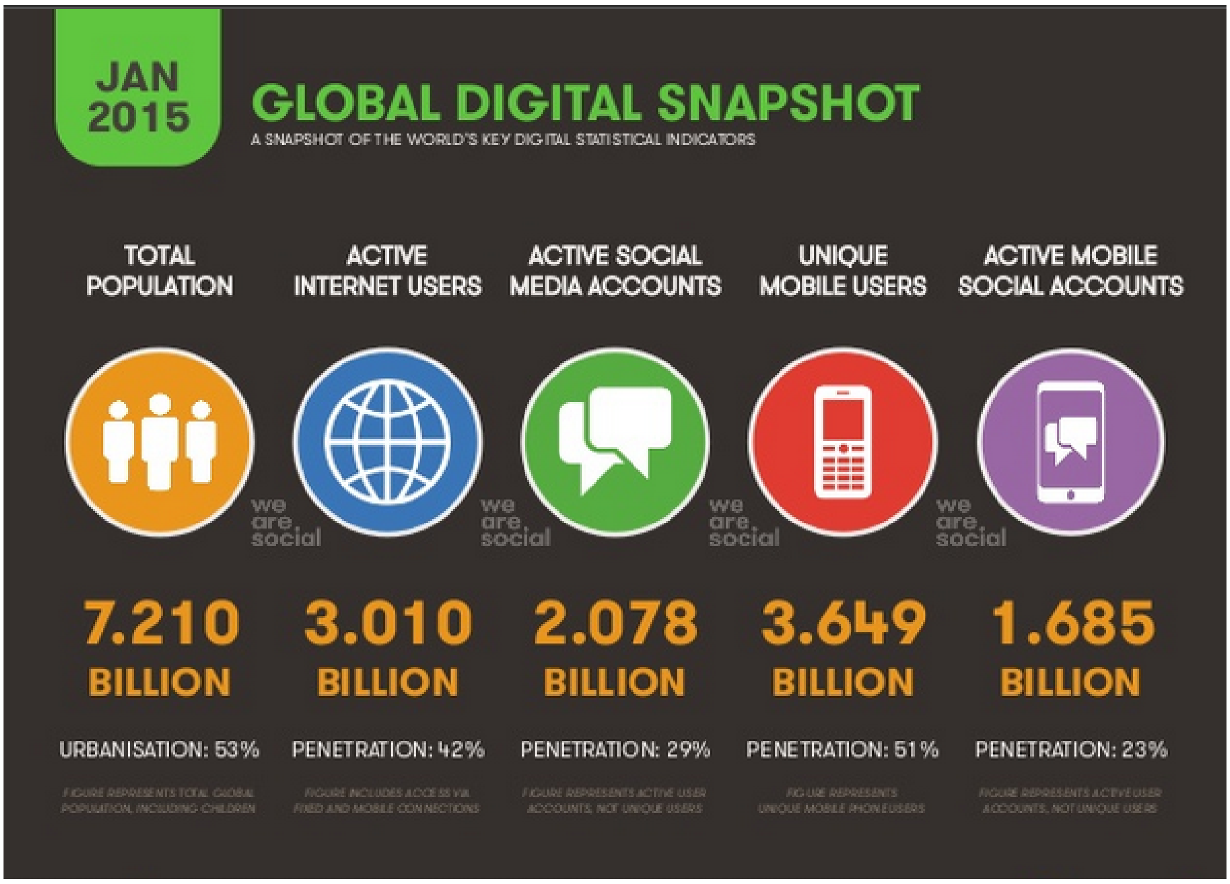
A few facts and figures

- by spring 2015 smartphone ownership in the US
  - *had hit ~64% of all adults*
  - *a rise from 35% in Spring 2011*
- research published by Pew Research Center, Washington
  - *at least 19% of US adults rely on smartphones*
  - *to access online services and information*
  - *due to lack of other broadband options*
  - *or they simply do not own an alternative device*
  - *perceived sub-class of 7%*
  - *solely reliant on smartphones for online access...*
  - *high level of smartphone ownership amongst younger Americans*
  - *at least 15% of young Americans between 18 and 29 yrs old*
    - heavily dependent on a smartphone for online access

Usage stats

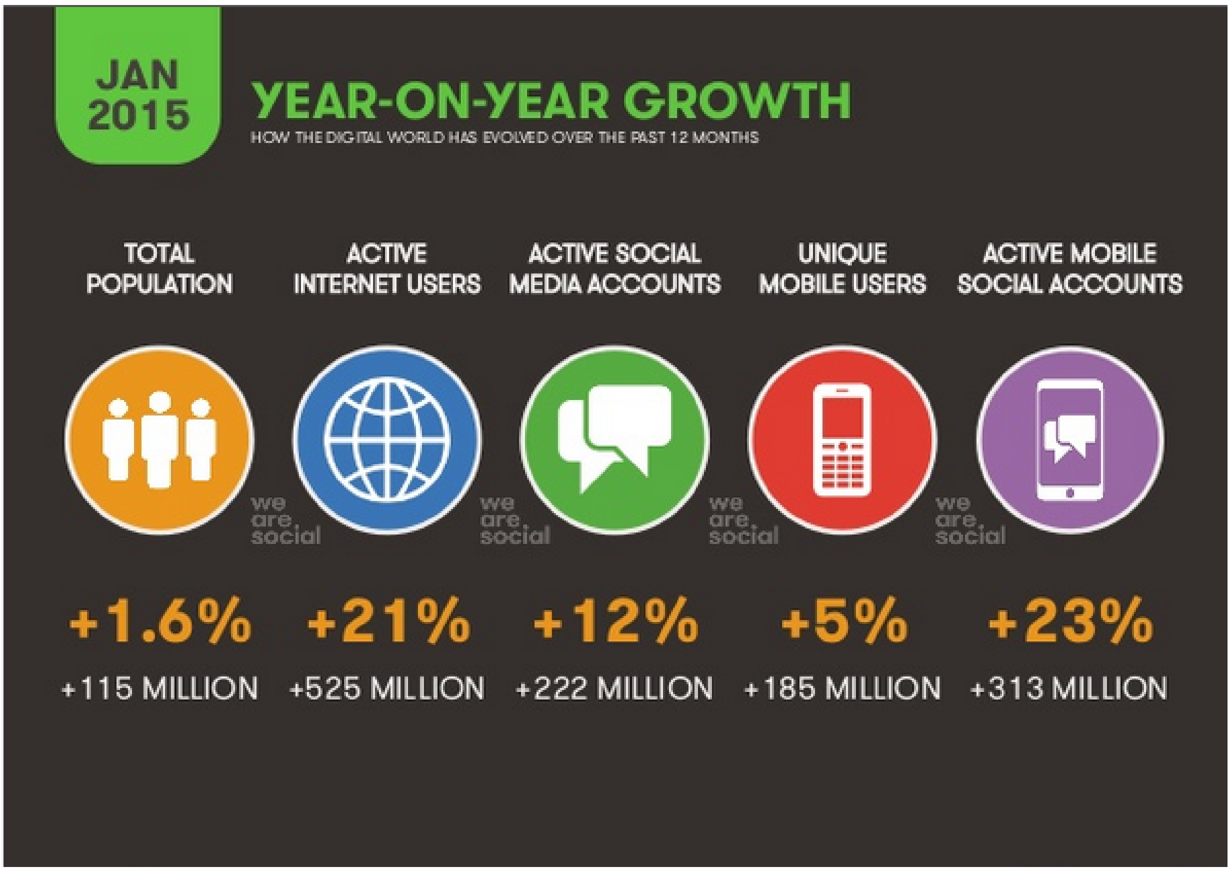
- usage stats are also v.interesting for developers
- eg: many users now use smartphones for less frivolous activities, including
  - *62% have used their smartphones to query information about their health or a medical condition*
  - *57% have used their smartphones to complete online banking*
  - *44% 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*

Image - Global Digital Snapshot



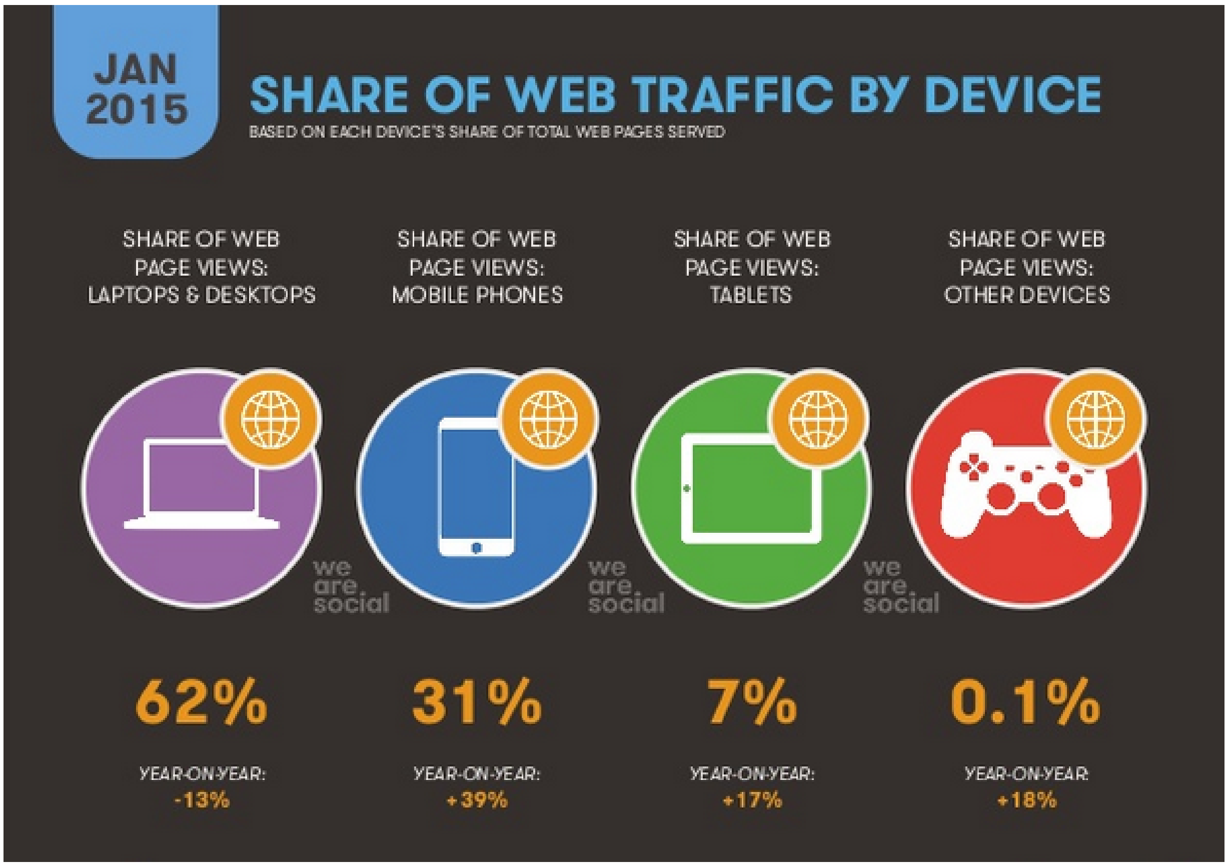
Source - We Are Social - Singapore

Image - Year on Year Growth



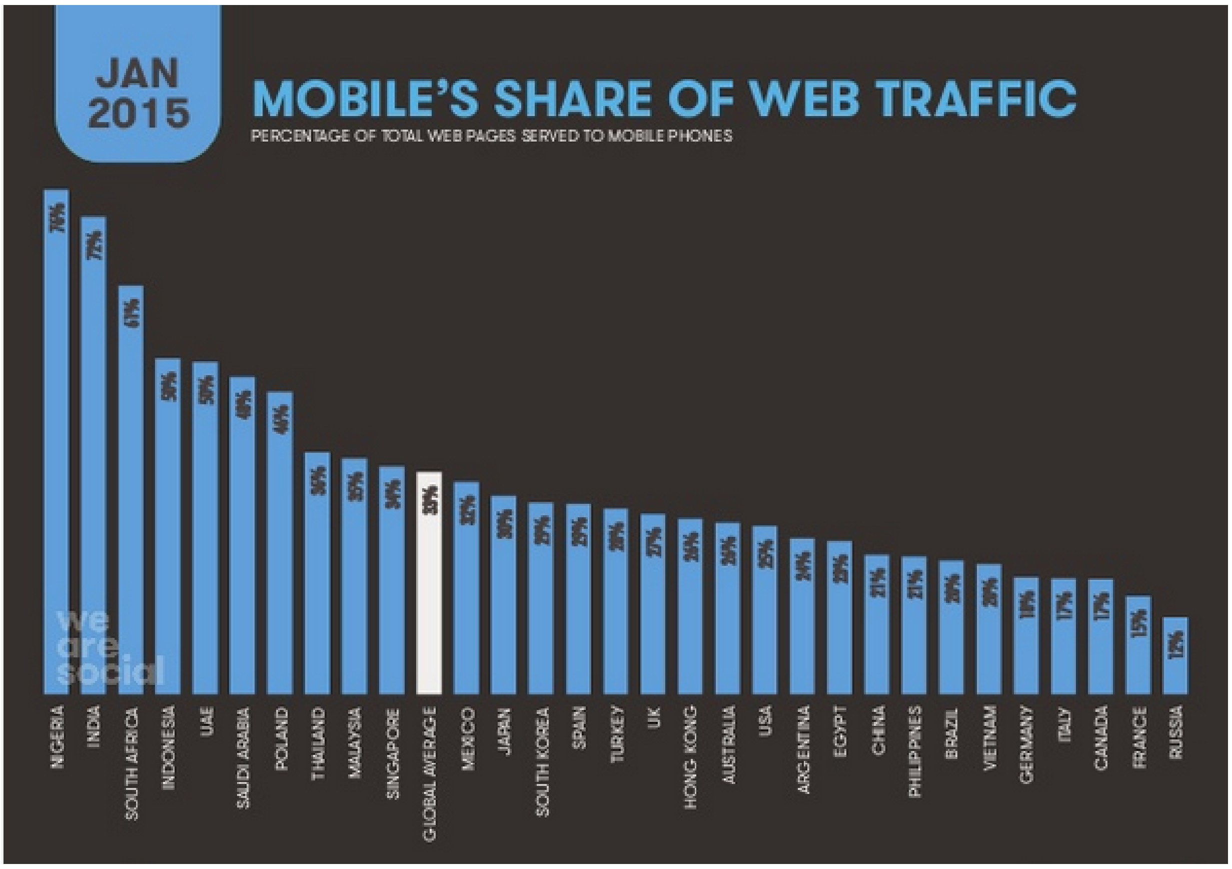
Source - We Are Social - Singapore

Image - Share of Web Traffic By Device



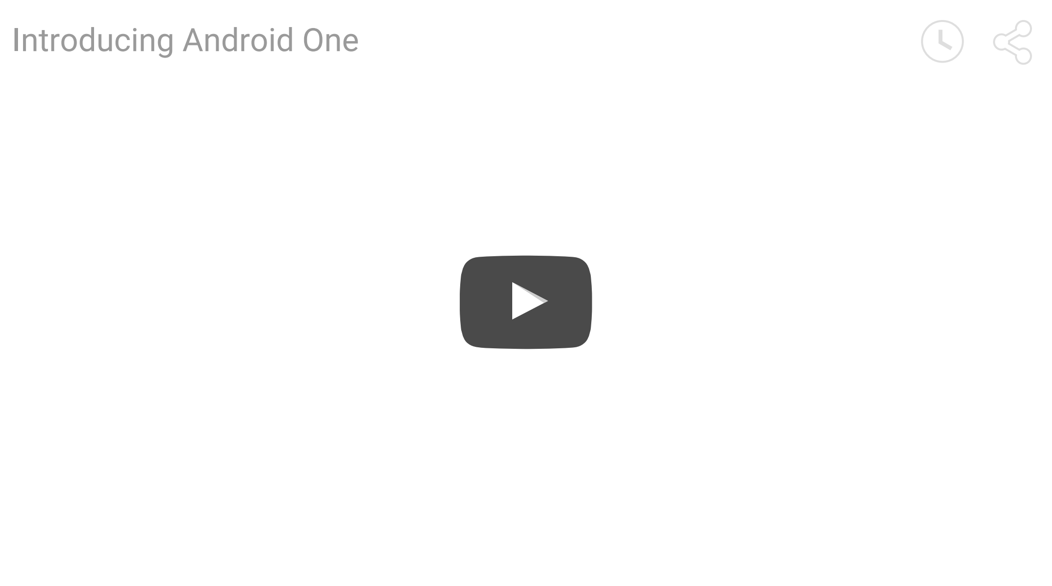
Source - We Are Social - Singapore

Image - Mobile's Share of Web Traffic



Source - We Are Social - Singapore

Video - Android One



Source - YouTube - Android One

Growing market

- optimistic point for developers
  - *growing market for mobile devices, apps, and services*
- developer's job to fill this need with apps...
- apps need to be
  - *useful, easy to use apps, aesthetically pleasing...*
- developers need to be able to develop apps quickly
  - *develop for multiple OSs and devices*
  - *largest markets*

Different types of mobile

- we need to be clear about the differences between mobile types
  - *mobile web*
  - *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
- 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

Native mobile

- native mobile app development 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*
    - *Objective-C (& Swift) for iOS*
    - *.Net for Windows Phone (Mobile...)*
- learn and develop different SDK etc 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 1

- hybrid mobile apps share a lot with native mobile apps
    - *eg: characteristics, design traits, functionality*
- however, they are developed using different tools, technologies, methods...
- Apache Cordova apps developed using common web technologies
    - *HTML (HyperText Markup Language)*
    - *CSS (Cascading Style Sheets)*
    - *JS (JavaScript)*

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

Considerations for mobile web

- many benefits to native app development
- obvious benefit is optimised nature of compiled code
- native apps will often be slightly faster than hybrid apps
- choice of development route will depend upon many factors
  - *time*
  - *cost*
  - *development expertise and experience*
  - *chosen platform(s)*
  - *scale of application*
- often a case of personal development preference

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)
Native	Yes	Native SDK & APIs	No (requires porting)	Full	Excellent
Hybrid	Yes	HTML, CSS, & JS	Yes (modifications)	Full (using plugins)	Very good to excellent

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 Phone (Mobile/10 ??)*
    - *BlackBerry*
    - *Tizen*
    - *Firefox OS*
    - *Ubuntu*
    - *...*

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 for Android & Objective-C (Swift) for iOS*
- each mobile OS has its own peculiarities
    - *differing design philosophies etc*

Cross-platform - common issues and solutions

- common issues might include
  - *permissions*
  - *access to underlying services within an OS*
  - *eg: 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 web technologies
  - *HTML, CSS, JavaScript*
  - *create easier cross-platform apps*

Image - Apache Cordova

Source - Apache Cordova

Apache Cordova - intro

- helps us develop cross-platform mobile apps
- Cordova leverages HTML, CSS, and JavaScript
- uses a powerful set of APIs
    - *allows access to native mobile functionality*
- Cordova started at a company called Nitobi in 2008
    - *project called* ***PhoneGap***
    - *simple goal to create easy to use cross-platform development*
    - *originally supported only iPhone*
    - *later added Android and BlackBerry support*
- in 2011 project acquired by Adobe
    - *donated open source core to Apache Software Foundation*
- original project goal continues with Apache's Cordova project

NB: PhoneGap still continues to be developed and promoted as an Adobe product.

Apache Cordova and PhoneGap

- initial differences between Cordova and PhoneGap were minimal
- Adobe has continued to add proprietary services to PhoneGap
  - *now developed ecosystem to fit PhoneGap*
- decide whether to choose
  - *proprietary Adobe PhoneGap or*
  - *open Apache Cordova and associated options*
- fun to note
  - *many other projects use Cordova at their core*
  - *eg; Ionic uses a combination of Cordova and AngularJS*

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*

Apache Cordova - limitations

- primary limitation to previous assertion
  - *reliance on nature of Cordova plugins*
  - *no plugin, no native functionality*
  - *either don't use or create our own plugin*
- real-terms limitation is lack of plugins
  - *or aptitude to develop a custom plugin*
- project such as *Ionic* are trying plug this gap
- goal of Ionic is to provide broader set of generic Cordova plugins
  - *help create more complex custom apps*
- Cordova provides some excellent and simple plugins
  - *perceived need for plugins to allow greater freedom*
  - *expose data from native layer to Cordova JavaScript*

Apache Cordova - functionality and plugins

- allows us to create native mobile applications using a set of common web technologies
    - *including HTML, 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 etc
- photo capture 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 - platform support

support includes following mobile OSs

- Android
- iOS
- Windows Platform
- Windows Phone 7 (to be deprecated as of version 3.7)
- Windows Phone 8 and Windows 8
- BlackBerry
- Tizen
- Firefox OS
- Ubuntu
- ...

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

NB: above repositories require knowledge of GitHub.

Apache Cordova - why choose it?

- potential to develop once, re-use with ease
- Cordova helps us solve some of the following mobile development issues
  - *different programming languages for different mobile OSs*
  - *different programming philosophies, conventions, best practices, guidelines...*
  - *unique problems inherent to each given mobile OS*
  - *eg: handling and routing SMS requests, data storage, privacy features...*
  - *developing, testing, and maintaining applications across multiple mobile platforms*

Apache Cordova - porting applications

- porting a mobile app from one native OS to another
    - *creates additional challenges for developers*
    - *handling incompatible behaviours of disparate mobile platforms*
    - *eg: restrictive nature of SMS integration in different OSs*
    - *not difficult issues to resolve but will require modified logic in the application*
    - *applications often require considerable re-development from platform to platform*
    - *different toolsets per OS create additional barrier to entry*
    - *different IDEs, OSs to develop for a given mobile OS*
- benefit of single based with Cordova
    - *gives developers a lot of flexibility*
    - *helps us handle vagaries and challenges of multiple mobile OSs*
    - *code is centralised making it easier to read, update, maintain...*

Apache Cordova - architecture - part 1

- Cordova relies on web technologies at its core
  - *HTML*
  - *CSS*
  - *JavaScript (JS)*
- core architecture for app development using Cordova
- supplement this core with additional helper files
  - *eg: JSON (JavaScript Object Notation) resource files*
- to enable access to a device's native functionality
  - *JS application objects (or functions) call Cordova APIs*
  - *Cordova APIs for different native mobile OSs*
  - *eg: use Cordova Android API for native Android functionality...*
- develop our own custom plugins as necessary

Image - Apache Cordova architecture

```
|--------------WebView-------------------|
|  --------------   -------------------- |
|  | HTML files |   | JavaScript files | |
|  --------------   -------------------- |
|                                        |
|  --------------   ----------------     |
|  | CSS files  |   | Helper files |     |
|  --------------   ----------------     |
|----------------------------------------|
```

Source - Apache Cordova

Apache Cordova - architecture - part 2

- core architecture creates a single screen in the native app
- single screen contains a **WebView**
  - *uses all of the device's available screen space (real estate)*
  - *native WebView used to enable loading app's HTML, CSS, JS...*
- WebView is a native view in each mobile OS
  - *allows us to display HTML based content*
  - *allows us to leverage power and functionality of a mobile browser*
  - *working within a contained native app*

Apache Cordova - WebView - part 1

- using this WebView in our app
    - *Cordova loads the app's default startup page*
    - *in essence its **`index.html`** page*
    - *passes control of the app to the native WebView*
    - *allows user to control the app as normal*
    - *user can interact with app in native manner*
    - *user get a native app experience*
- user interaction can include standard native interaction patterns and options
- user is not aware of difference between Cordova or native developed app

Apache Cordova - WebView - part 2

- WebView has an implementation in all of the major mobile OSs
- Android has a class called

`android.webkit.WebView`

- iOs references the **`UIWebView`**
  - *part of the UIKit framework*
- Window Phone refers to a WebView class called

`Windows.UI.Xaml.Controls`

Apache Cordova - native functionality - part 1

- provides access to many types of native functionality, including
  - *sound and audio*
  - *recording*
  - *camera capture*
  - *photo access*
  - *geolocation*
  - *sensors…*
- Cordova leverages JavaScript APIs to provide native functionality

Image - Apache Cordova - Native Functionality

```
|-----------------------------------------------------------------------|
|  |-----------------------------------------------------------------|  |
|  |                            WebView                              |  |
|  |                                                                 |  |
|  |  |-------------|    |---------------|          |-----------------|  |  |
|  |  | HTML files  |    |   JS files    | -------> |                 |  |  |
|  |  |-------------|    |---------------|          |                 |  |  |
|  |                                                |   Cordova JS    |  |  |
|  |  |-------------|    |---------------|          |      APIs       |  |  |
|  |  | CSS files   |    | Helper files  |          |                 |  |  |
|  |  |-------------|    |---------------|          |-----------------|  |  |
|  |                                                         |         |  |
|  |---------------------------------------------------------|---------|  |
|                                                            |            |
|                                                            v            |
|  |-----------------------------------------------------------------|  |
|  |                      Native Device APIs                         |  |
|  |-----------------------------------------------------------------|  |
|                                                                       |
|-----------------------------------------------------------------------|
```

Source - Apache Cordova

Apache Cordova - native functionality - part 2

- architecture is an elegant approach to solving cross-platform issues
- allows developers to leverage unified API interface
    - *perform specific native functions*
    - *calls to native functionality transparent across platforms*
    - *strength of using JavaScript APIs*
- Cordova JavaScript APIs
    - *call the required native OS API*
    - *eg: Cordova's Android or iOS API*
- plugins give Cordova its power and flexibility

References

- Carmody, Tim., *Fighting Words: Defining "Mobile" and "Computer"* Wired. 11.08.2010. http://www.wired.com/2010/11/fighting-words-defining-mobile-and-computer/
- Smith, Aaron., *U.S. Smartphone Use in 2015* PewResearchCenter. 04.01.2015.http://www.pewinternet.org/2015/04/01/us-smartphone-use-in-2015/
- Various., *Digital, Social & Mobile in 2015* We Are Social Singapore. 01.20.2015. http://www.slideshare.net/wearesocialsg/digital-social-mobile-in-2015