Introduction to Mobile Apps

CSE 5236: Mobile Application Development
Instructor: Adam C. Champion, Ph.D.
Course Coordinator: Dr. Rajiv Ramnath
Reading: Big Nerd Ranch Guide, Chapter 1
Info About Me

• Adam Champion
• Ph.D., OSU, 2017
• Research interests:
  – Mobile systems, networks, security, analytics
  – Computer networking, wireless communications
  – Parallel and distributed systems
• More:
  – http://www.cse.ohio-state.edu/~champion.17
More about you?

• Which OS do you prefer: iOS or Android?
• What’s your favorite mobile app?
  – What do you like most about it?
• What’s your least favorite mobile app?
  – What do you dislike about it?
• How do mobile apps connect to Internet-based services for end users? (Cost implications)
• How to capture data from on-device sensors (e.g., GPS, accelerometer)?
Course Objectives

- Understand characteristics of mobile apps
- Understand frameworks for mobile app dev.
  - Deep dive: Android framework
  - Device programming: Java, Kotlin
  - App components: design mobile UI, connect to Internet services, use on-device sensors
- Understand design principles for mobile apps
  - Design apps for ease of maintenance
  - Make apps fast, responsive (60 frames/sec goal)
Essence of A Mobile Device

• (Potentially) available to serve everywhere, any time.
• Interwoven into daily life – live, work, play, study
• Represents and intimately “knows” the user
  – Much more than just a small computer, it represents the user
• Brings in outside world: sensing, location, communication
• Now the dominant end-user device
  – 81% of Americans own a smartphone (Feb. 2019): https://www.pewinternet.org/fact-sheet/mobile/
Varied Shapes, Sizes, Capabilities

Sources: Apple, Google, Nintendo, Amazon
Mobile App. Development Challenges

- Competitive, fluid vendor landscape (Apple, Android consortium incl. Amazon, RIM, HP) means apps need to be multi-platform for wide adoption
- No “standard” device (iOS, Windows Phone devices?)
- Low bandwidth input (in most cases; what about tablets?)
- Limited screen size (tablets?)
- Unreliability in connectivity and device (network access, power, ambient light, noise, at least for now)
- Integration tradeoffs with cloud and enterprise services
Application Development Support

• Object-Oriented Languages
  – iOS: Swift
  – Android: Java/Kotlin

• Cross-platform frameworks:
  – Flutter, Xamarin, Titanium, PhoneGap, …
  – Scripting languages (JavaScript, Ruby)

• C and C++ (native code)

• Integrated into mobile app dev. frameworks
Android Framework Support

Applications use framework via API calls

Application Framework is based on runtime

Core libraries provide Java data structures, etc.

All Android apps run via ART/Dalvik VM

Runs atop native (C/C++) libraries

Exposes API to Android from hardware

Interacts with kernel

"Talks to" underlying hardware via kernel

Home Screen, Contacts, Dialer, SMS, MMS, Camera, Media Player, Web Browser, ...

Activity Manager, Window Manager, Content Providers, View System, Notification Manager, Package Manager, Telephony Manager, Resource Manager, Location Manager, ...

Android Runtime (ART)

Core Libraries

Java

ART/Dalvik VM

Native Libraries (C/C++)

Bionic libc, WebKit, SQLite, Media Framework, Surface Flinger, Audio Flinger, ...

Graphics, Audio, Camera, Bluetooth, GPS, Cellular Radio, Wi-Fi, ...

Linux Kernel

USB Driver, Display Driver, WiFi Driver, Bluetooth Driver, Audio Drivers, Camera Driver, Keypad Driver, Shared Mem. Driver, More drivers, functions

Google's Kernel Enhancements

Binder IPC, Power Management
Framework Capabilities, Add-Ons

• Built-in Services:
  – GUI
  – OS services (file I/O, threads, device mgmt.)
  – Graphics
  – Device access (GPS, camera, media players, sensors)
  – Networking
  – Standard language libraries (Java, Kotlin)

• Add-ons:
  – Google Play services (e.g. Google Maps, etc.)
  – Database support (SQLite)
  – Chromium WebView
# Tooling Support

<table>
<thead>
<tr>
<th>Tool Type</th>
<th>Android Support</th>
<th>iOS Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open-source Integrated Dev. Environments (IDEs)</td>
<td>Android Studio</td>
<td>–</td>
</tr>
<tr>
<td>Proprietary IDEs</td>
<td>IntelliJ IDEA</td>
<td>Xcode</td>
</tr>
<tr>
<td>Testing tools</td>
<td>JUnit, Espresso</td>
<td>XCUnit</td>
</tr>
<tr>
<td>Profiling tools</td>
<td>Android Profiler</td>
<td>Xcode Instruments</td>
</tr>
<tr>
<td>Source code management</td>
<td>Git, Subversion, CVS (for both Android, iOS)</td>
<td></td>
</tr>
<tr>
<td>Software emulators</td>
<td>Android Emulator, Intel HAXM, Genymotion</td>
<td>iPhone Simulator</td>
</tr>
<tr>
<td>Sensor injection tools</td>
<td>Built into emulators (for both Android, iOS)</td>
<td></td>
</tr>
</tbody>
</table>
IDE Support

- Open IDEs – Android Studio (Android)
- Proprietary (Xcode for iOS)
- Testing tools (unit tests, UI tests, test harnesses)
- Performance profiling tools
- SCM integration (Git, SVN, CVS)
- Software emulators
- Sensor injection (GPS, accelerometer)
Thank You

Questions and comments?