Introduction – What is Android?

CS 2046
Mobile Application Development
Fall 2010
Everything you know is wrong...

• Most desktop/web applications:
  – Large screen size
Everything you know is wrong...

- Most desktop/web applications:
  - High-quality internet connections
Everything you know is wrong...

• Most desktop/web applications:
  – Take one or two types of input
Modern Computing

• Think – could you go without your computer for a day?
  – What if you had a smartphone that could:
    • Check email
    • Read news
    • Play games, music, and viral videos
  – What do we really use computers for?
Mobile Applications

- **Apple App Store**
  - 250,000 applications
  - Has paid over $1 billion to developers as of June

- **Android Market**
  - 80,000 applications
  - Android OS (in some form) installed on 1/3 of new phones (#1 in the U.S.)
Differences

• Could you sell a great desktop app?
  – Maybe if you’re Microsoft or Adobe
  – You’ll get ~100% of revenue!
    • Hooray!
  – But:
    • How to push your product?
      – Advertising
      – Media – CDs or bandwidth
    • How do people download, install?
    • How do they know it’s going to work?
Differences

• Typical mobile market (iOS, Android):
  – Many apps are free
    • But ad-supported – shareware is back!
  – For paid apps, developer gets ~70% of revenue
  – Tradeoff for:
    • One-click install for everyone (including payment!)
    • Central source for finding your applications
    • No worrying about handling financials, statistics...
Differences

• Screen size varies tremendously
  – Currently, for majority – small or very small
Differences

• Internet connection is extremely sporadic
  – Phones support Wi-Fi, GSM/EDGE, CDMA/EV-DO, 3G, 4G, etc., and switch often
  – But many applications rely on internet data
  – Need resilience for transient failures
Differences

• Replacements for typical types of input
  – Keyboard
    • Generally, still exists. But:
      – Might be a virtual keyboard
      – Might be hardware, but will be very small (so no Starcraft)
  – Mouse
    • Most devices have touch screens
      – But multi-touch is new compared to a mouse
    • Could also have a trackball, navigation buttons, etc.
Differences

• Plus, many more types:
  – Accelerometer
    • Labyrinth
  – GPS
    • Fine-grained local search
  – Compass
  – Vibration
    • BUMP - [http://bu.mp/](http://bu.mp/)
  – Sound
    • Speech recognition
  – Camera
Differences

• Plus, many more types:
  – Accelerometer
    • Labyrinth
  – GPS
    • Fine-grained local search
  – Compass
  – Vibration
    • BUMP - [http://bu.mp/](http://bu.mp/)
  – Sound
    • Speech recognition
  – Camera

Nothing really new!
But, can rely on almost every device supporting all of them.
Android

- “A software stack for mobile devices that includes an operating system, middleware, and key applications.”

- Essentially, a unified platform to develop apps that will run on many different phones and devices.

- A wrinkle:
  - Open source – get some modifications (MotoBLUR)
The Platform

- Code is written in Java
  - vs. Apple iOS: Objective-C
  - Exception – can compile most C code using the NDK
- Libraries available:
  - Essentially all of the standard java.* packages
  - android.* - Android-specific libraries
    - UI elements
    - SQLite data storage
    - Media support (audio/video)
    - Access to camera, GPS, accelerometer
Development

- Open-source platform, open-source tools
  - Eclipse IDE
  - Android SDK
- Emulator for running applications
  - If you have a real device, can use that too.
- See course webpage for directions.
- We will have a lab exercise in class to introduce the tools.
Software Stack

• Lowest level – Linux kernel
  – Android != Linux
    • Does not use glibc, X, or GTK
  – Used for security, memory management, threading...

• Dalvik Virtual Machine
  – Each app runs in its own VM instance (secure)
  – Sort of like Java VM on the desktop, but optimized for mobile

• Application Framework
  – Where we will be spending our time
What is an Android application?

• Four major components:
  – Activity
    • Main component of an application
    • Generally, visible to the user
  – ContentProvider
    • Stores (tabular) data and makes it available to other applications
  – Service
    • Handles background work and ongoing tasks
  – BroadcastReceiver
    • Receives events from the phone
Example – Music Player

• Activity
  – UI for browsing music, selecting songs

• ContentProvider
  – Store the music and metadata
  – Gives other apps access
    • Music recommendation

• Service
  – Actually plays music – supports backgrounding

• Broadcast Receiver
  – Pause music if headphones are unplugged

• Project: Tasks
What we will cover

• Building simple applications
  – DEMO – your project
• Mobile user interfaces
• Data storage
• Interacting with the Internet
• Multimedia
• Other topics if there’s interest
  – Let me know!
  – Intro of the Day
Course Logistics

• Instructor
  – Jeff Davidson (M.Eng)
  – jpd236@cornell.edu
  – Office Hours: MW 1:15-2:15pm (i.e. after class) or by appointment, starting next Monday, 10/25. See site for location.

• Teaching Assistant
  – Jae Yong Sung
  – js946@cornell.edu
  – Office Hours: TBA, probably Friday 11:00am-12:00pm.
Course Logistics

• Dates/Times
  – October 18th – November 12th
  – MWF 12:20-1:10pm, Upson 207
  – Add/Drop deadline: 10/25

• Websites
  – CMS: [http://cms.csuglab.cornell.edu](http://cms.csuglab.cornell.edu)
Grading

• 1 credit, S/U only

• Components:
  – 70%: Individual Assignments
  – 15%: Lab Exercise (next Monday, 10/25)
  – 15%: Participation
    • In-class
    • Newsgroup: cornell.class.cs2046
      – See website for instructions
    • Challenge problems

• Passing Grade: 60%
Course Format

• Lectures
  – Conceptual understanding, demos.
  – Only 4 weeks – not enough time to pore over lines of code.

• Assignments
  – There’s a reason they’re worth 70% of the grade...
  – You will learn most of the concrete material – how to code applications – from completing them.

• Come to office hours if you get stuck!
Intents

• On a desktop:
  – Open an application – double click it
  – Open a file with an application – drag file to shortcut
    • Alternatively, pass filename as an argument.
  – Can specify default handler for a certain filetype.
Intents

- Android uses Intents to accomplish these (and more) tasks.
- Intents: abstract description of an operation to be performed.
  - Action: The general action to be performed
  - Data: What should be operated on
  - Component: What will perform the action (optional)
- “The boy hits the ball”
  - What is the action? Data? Component?
Intent Examples

• **ACTION_VIEW** [http://www.cornell.edu](http://www.cornell.edu)
  – Launch an application to view this URL.

• **ACTION_DIAL** tel:5551234
  – Launch an application to call this phone number.

• Note – these intents do not specify the component which should handle them.
  – We call these *implicit* intents
  – *Explicit* intents include the target component
Intent Resolution

• Applies when no target is specified
• *Intent filters*
  – Associated with each component
  – Describe the Intents that the component can handle.

• Your app calls `startActivity()` with `ACTION_VIEW <url>`
  – For each Activity registered with the system:
    • Does it support the action `ACTION_VIEW`?
    • Does it support URI’s of the form http://*?
Intent Resolution

• If only one application matches, launch it.

• What if there are multiple possible handlers?
  – Example: Multiple browsers
Intents - Summary

• Intents – the “glue” connecting Android components.

• Can specify what should handle the Intent (explicit), or let the system figure it out (implicit).

• Standardized Intents make Android extensible:
  – Make an application that takes pictures as input? Gallery will automatically let you send pictures to it.