Overview of CS 282 & Android

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA

CS 282 Principles of Operating Systems II
Systems Programming for Android
Topics Covered in this Part of the Module

• Course goals & logistics
Topics Covered in this Part of the Module

• Course goals & logistics

• Present an overview of the Android software architecture
Course Goals

• Learn about
  • Mobile devices
  • Systems programming for mobile devices
  • The Android platform
• Develop interesting Android systems programming applications
  • Expect lots of programming
  • Each student will do multiple projects
  • There may also be a group project at the end
Administrivia
Logistics

- Douglas C. Schmidt
  - d.schmidt@vanderbilt.edu
  - Office: FGH #226
  - Office hours: M. 1-3pm & W. 1-3pm
    - Nearly always reachable by email
- TAs/graders
  - Nick King <nicholas.b.king@vanderbilt.edu>
  - Nolan Smith <nolan.m.smith@vanderbilt.edu>
  - Lane Kelly <lane.m.kelly@vanderbilt.edu>
- Course URL: www.dre.vanderbilt.edu/~schmidt/cs282
Course Work

• There will be 5-6 programming assignments written in Java
  • Can use Windows, Linux, Mac, etc.
  • \textit{Must} be done individually
• Programs will be graded as follows:
  • 40\% execution correctness
  • 30\% structure (e.g., modularization, information hiding, etc.)
  • 10\% insightful programming (e.g., developing reusable class components, etc.)
  • 10\% Consistent style (e.g., capitalization, indenting, etc.)
  • 10\% appropriate commenting style
Course Work

• There will be a 5 point deduction (out of a possible 100 points) for each day that your program is late
  • Programs turned in later than two calendar days after the due date will receive a zero
• There will be weekly quizzes & a comprehensive final exam
• The relative weighting of each portion of the course is:
  • 40% Programming projects
  • 40% Quizzes
  • 10% Final Exam
  • 10% Class participation
Ground Rules

• Assignments must be submitted on time
• Work **must** be your own (see www.owen.vanderbilt.edu/vanderbilt/about-us/honor-code.cfm)
• *No* laptops open in class unless explicitly allowed
• You will be called upon to answer questions
  • 10% class participation grade, so be involved & attend class
• You’ll get out of this course what you put into it, so be prepared to work hard & learn a lot
• Be prepared for weekly quizzes & occasional guest lectures
• Make sure to avail yourself of available help, e.g., office hours, TAs, mailing list, etc.
Class Organization

• Mix of lecture & programming exercises
  • ½ presentation
  • ½ laboratory exercises & semester project
• Organization will remain flexible
  • Will change as needed
Why Mobile Devices & Android?
Mobile Devices are the Next Computing Platform

Smartphone Sales To Beat PC Sales By 2011

Source: RBC Capital Markets estimates
Why Android?

- Android has > 50% of the smartphone market (#1)
- iPhone has < 30% of the smartphone market (#2)
- Blackberry, Windows Mobile, & etc. are rapidly losing market share since their platforms not nearly as interesting to develop for as Android/iPhone

### Top Smartphone Platforms
3 Month Avg. Ending Feb. 2012 vs. 3 Month Avg. Ending Nov. 2011
Total U.S. Smartphone Subscribers Ages 13+
Source: comScore MobiLens

<table>
<thead>
<tr>
<th></th>
<th>Share (%) of Smartphone Subscribers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Nov-11</td>
</tr>
<tr>
<td>Total Smartphone Subscribers</td>
<td>100.0%</td>
</tr>
<tr>
<td>Google</td>
<td>46.9%</td>
</tr>
<tr>
<td>Apple</td>
<td>28.7%</td>
</tr>
<tr>
<td>RIM</td>
<td>16.6%</td>
</tr>
<tr>
<td>Microsoft</td>
<td>5.2%</td>
</tr>
<tr>
<td>Symbian</td>
<td>1.5%</td>
</tr>
</tbody>
</table>
Why Android?

Android is:
- the fastest growing smartphone platform
  - open-source & works on multiple platforms
  - no need to own a Mac
  - no need to join a developer program
- Easy to learn for Java (& C++) programmers
  - Much easier to transition to than Objective-C
Getting Started with Android
Developing Android Apps

- Android is a software stack for mobile devices that provides an operating system, middleware, & key services/applications
- The Android SDK contains libraries & development tools for creating applications
Developing Android Apps

- Android is a software stack for mobile devices that provides an operating system, middleware, & key services/applications
- Android uses the Eclipse Integrated Development Environment (IDE)
Developing Android Apps

- Android is a software stack for mobile devices that provides an operating system, middleware, & key services/applications.
- Android uses the Eclipse Integrated Development Environment (IDE).
- Android Eclipse Plugins provide:
  - wizards for creating new apps
  - a visual editor for creating GUIs
  - editors for manipulating Android XML descriptors needed for your app
  - an emulator for testing your apps on your PC
  - a debugger for finding errors in the emulator or on a device
Developing Android Apps

• Android is a software stack for mobile devices that provides an operating system, middleware, & key services/applications
• The Android SDK contains libraries & development tools for creating applications
• Android uses the Eclipse Integrated Development Environment (IDE)
• Android Eclipse Plugins provide:
  • wizards for creating new apps
  • a visual editor for creating GUIs
  • editors for manipulating Android XML descriptors needed for your app
  • an emulator for testing your apps on your PC
  • a debugger for finding errors in the emulator or on a device
Setting Up an Android Development Environment

• Follow the instructions for Lab1 at http://www.dre.vanderbilt.edu/~schmidt/cs282/Lab1.pdf
Figuring Out Android

• Android is well documented
• The Android javadoc references will be critical reference material for your projects:
• The Android developer guide is another important resource:
• We recommend “The Busy Coder’s Guide to Android Development” e-book
  • [http://commonsware.com/warescription](http://commonsware.com/warescription)
Overview of Android
What is Android?

- Android provides a \textit{layered} software stack for mobile devices, including
- A variant of the Linux OS optimized for power conservation & local IPC
What is Android?

- Android provides a layered software stack for mobile devices, including:
  - A variant of the Linux OS optimized for power conservation & local IPC
  - An optimized Java Virtual Machine (Dalvik), a subset of Java libraries running on Dalvik, native C/C++ libraries, & a hardware abstraction layer
What is Android?

- Android provides a *layered* software stack for mobile devices, including:
  - A variant of the Linux OS optimized for power conservation & local IPC
  - An optimized Java Virtual Machine (Dalvik), a subset of Java libraries running on Dalvik, native C/C++ libraries, & a hardware abstraction layer
- Middleware, including:
  - GUIs
  - Telephony services
  - Camera
  - Multimedia
  - App frameworks
  - App Distribution
  - etc.
What is Android?

- Android provides a *layered* software stack for mobile devices, including:
  - A variant of the Linux OS optimized for power conservation & local IPC
  - An optimized Java Virtual Machine (Dalvik), a subset of Java libraries running on Dalvik, native C/C++ libraries, & a hardware abstraction layer
  - Middleware, including:
    - GUIs
    - Telephony services
    - Camera
    - Multimedia
    - App frameworks
    - App Distribution
    - etc.
  - Common set of apps

See [developer.android.com/guide/basics/what-is-android.html](http://developer.android.com/guide/basics/what-is-android.html) for more
Linux Kernel

- Provides infrastructure mechanisms to manage mobile device resources
  - Memory, process, & thread management
  - Network & inter-process communication stack
- Device driver framework
- Security
Linux Kernel

- Provides infrastructure mechanisms to manage mobile device resources
  - Memory, process, & thread management
  - Network & inter-process communication stack
  - Device driver framework
  - Security

- Android-specific enhancements
  - Binder – optimized inter-process communication (IPC)
  - Android shared memory
  - Power management
  - Alarm driver
  - Low memory killer
  - Kernel debugger & Logger

en.wikipedia.org/wiki/Android_(operating_system)#Linux has more info
Hardware Abstraction Layer (HAL)

- User space C/C++ library layer that defines the interface Android requires hardware “drivers” to implement
- The HAL helps to decouple
  - Android platform logic from hardware interface
  - Android frameworks from Linux kernel
Hardware Abstraction Layer (HAL)

- User space C/C++ library layer that defines the interface Android requires hardware “drivers” to implement
- The HAL helps to decouple
  - Android platform logic from hardware interface
  - Android frameworks from Linux kernel

- Motivation for a user-space HAL
  - Not all components have standardized kernel driver interfaces
    - Android has specific requirements for hardware drivers
  - Kernel drivers are GPL, which exposes proprietary intellectual property of Android
    - Implementations of HAL components are often *not* open-source

See tidsp.es.ncku.edu.tw/cinfon/resource/slides/11102012_02_final.pdf
Native C/C++ Libraries

- System C library
  - bionic libc
- Surface Manager
  - display management
- Media Framework
  - audio/video streaming
- FreeType
  - library for rendering fonts
- Webkit
  - web browser engine
- OpenGL ES, SGL
  - graphics engines
- SQLite
  - relational database engine
- SSL
  - secure sockets layer
Native C/C++ Libraries

- System C library
  - bionic libc
- Surface Manager
  - display management
- Media Framework
  - audio/video streaming
- FreeType
  - library for rendering fonts
- Webkit
  - web browser engine
- OpenGL ES, SGL
  - graphics engines
- SQLite
  - relational database engine
- SSL
  - secure sockets layer

Android Runtime

• Support services for executing apps & frameworks

• **Dalvik Virtual Machine (VM)**
  
  • Android apps typically written in Java, but don’t run in a standard Java VM
Android Runtime

- Support services for executing apps & frameworks

  - **Dalvik Virtual Machine (VM)**
    - Android apps typically written in Java, but don’t run in a standard Java VM
    - Bytecodes executed in Dalvik VM “register machine”
    - **dx** program transforms java classes into .dex-formatted bytecodes
    - Just-in-time (JIT) compiler available
Android Runtime

- Support services for executing apps & frameworks

- **Dalvik Virtual Machine (VM)**
  - Android apps typically written in Java, but don’t run in a standard Java VM
  - Bytecodes executed in Dalvik VM “register machine”
    - `dx` program transforms java classes into .dex-formatted bytecodes
    - Just-in-time (JIT) compiler now available
  - Apps typically run in their own processes, inside their own Dalvik VM instance

See [en.wikipedia.org/wiki/Dalvik_(software)](en.wikipedia.org/wiki/Dalvik_(software)) for more on Dalvik.
Overview of CS 282 and Android

D. C. Schmidt

Android Runtime

- Support services for executing apps & frameworks
  - Dalvik Virtual Machine (VM)
    - Android apps typically written in Java, but don’t run in a standard Java VM
    - Bytecodes executed in Dalvik VM “register machine”
    - dx program transforms java classes into .dex-formatted bytecodes
  - Just-in-time (JIT) compiler now available
  - Apps typically run in their own processes, inside their own Dalvik VM instance

- Core Libraries
  - Core Java classes
    - android.*
    - java.*, javax.*
    - junit.*
    - org.apache.*, org.json.*, org.xml.*
Android Runtime

• Support services for executing apps & frameworks

• Dalvik Virtual Machine (VM)
  • Android apps typically written in Java, but don’t run in a standard Java VM
  • Bytecodes executed in Dalvik VM “register machine”
  • `dx` program transforms java classes into .dex-formatted bytecodes
  • Just-in-time (JIT) compiler now available
  • Apps typically run in their own processes, inside their own Dalvik VM instance

• Core Libraries
  • Core Java classes
    • android.*
    • java.*, javax.*
    • junit.*
    • org.apache.*, org.json.*, org.xml.*
  • Doesn’t include all standard Java SDK classes

en.wikipedia.org/wiki/Comparison_of_Java_and_Android_API has more info
Application Frameworks

- Provide services that are essential to the Android platform

- **Window Manager**
  - Manages top-level window’s look & behavior

- **View System**
  - Lists, grids, text boxes, buttons, etc.

- **Content Providers**
  - Inter-application data sharing

- **Activity Manager**
  - Application lifecycle & common navigation stack

- **Package Manager**
  - Manages application packages

- **Telephony Manager**
  - State of telephony services

- **Resource Manager**
  - Manages non-code resources: strings, graphics, & layout files

- **Location Manager**
  - Access to system location services

- **Notification Manager**
  - Notify users when events occur
Application Frameworks

- Provide services that are essential to the Android platform

- **Window Manager**
  - Manages top-level window’s look & behavior

- **View System**
  - Lists, grids, text boxes, buttons, etc.

- **Content Providers**
  - Inter-application data sharing

- **Activity Manager**
  - Application lifecycle & common navigation stack

- **Package Manager**
  - Manages application packages

- **Telephony Manager**
  - State of telephony services

- **Resource Manager**
  - Manages non-code resources: strings, graphics, & layout files

- **Location Manager**
  - Access to system location services

- **Notification Manager**
  - Notify users when events occur

[links to more info] sites.google.com/site/io/inside-the-android-application-framework
Applications

• Some standard apps include:
  • **Home**
    • main screen
  • **Contacts**
    • contacts database
  • **Calendar**
    • track schedules
  • **Camera**
    • take photos & videos
  • **Phone**
    • dial phone numbers
  • **Browser**
    • view web pages
  • **Email reader**
    • Gmail & others
  • **Media player**
    • Play songs & watch movies
  • **SMS/ MMS**
    • Instant messaging

All apps written using Java (Android frameworks use many JNI calls to C/C++)
Key Types of Android Components

• **Activity**
  • Represents a single screen with a user interface

1: Activity calls `downloadImage()` with image URL

See [www.dre.vanderbilt.edu/~schmidt/cs282](http://www.dre.vanderbilt.edu/~schmidt/cs282) for info on this app
Key Types of Android Components

• **Activity**
  • Represents a single screen with a user interface
  • Can be started by creating an Intent object & passing it to startActivity()
  • Parameters can be passed as “extras” to the Intent used to start the Service
Key Types of Android Components

**Activity**

- Represents a single screen with a user interface
- Can be started by creating an Intent object & passing it to startActivity()
- Parameters can be passed as “extras” to the Intent used to start the Service
- Apps can have multiple Activities

See developer.android.com/guide/components/activities.html for more info
Key Types of Android Components

• **Activity**
  • Represents a single screen with a user interface

• **Service**
  • Runs in background to perform long-running operations or to access remote resources

![Download Service Diagram]

1: Activity calls `downloadImage()` with image URL
2: Sends GET request to web server

[Image of Android device with ThreadedDownload app interface]
Key Types of Android Components

• **Activity**
  • Represents a single screen with a user interface

• **Service**
  • Runs in background to perform long-running operations or to access remote resources
    • *Started Service* – Often performs a single operation & usually doesn’t return a result to the caller directly
    • Parameters can be passed as “extras” to the Intent used to start the Service

---

**Download Service**

1: Activity calls `downloadImage()` with image URL

2: Sends GET request to web server
Key Types of Android Components

• **Activity**
  - Represents a single screen with a user interface

• **Service**
  - Runs in background to perform long-running operations or to access remote resources
    - *Started Service* – Often performs a single operation & usually doesn’t return a result to the caller directly
      - Parameters can be passed as “extras” to the Intent used to start the Service
    - *Bound Service* – Offers a client-server interface that allows components to interact with the Service
      - e.g., via the Android Interface Definition Language (AIDL) & Binder RPC

See [developer.android.com/guide/components/services.html](https://developer.android.com/guide/components/services.html) for more info
Key Types of Android Components

- **Activity**
  - Represents a single screen with a user interface

- **Service**
  - Runs in background to perform long-running operations or to access remote resources

- **Content Provider**
  - Manages a shared set of application data

Download Service
2: Sends GET request to web server
3: Stores downloaded image in filesystem & metadata in Content Provider

1: Activity calls `downloadImage()` with image URL

Image Metadata
Content Provider
Image Files
Key Types of Android Components

- **Activity**
  - Represents a single screen with a user interface

- **Service**
  - Runs in background to perform long-running operations or to access remote resources

- **Content Provider**
  - Manages a shared set of application data
    - Data typically stored persistently in an SQLite database
Key Types of Android Components

- **Activity**
  - Represents a single screen with a user interface

- **Service**
  - Runs in background to perform long-running operations or to access remote resources

- **Content Provider**
  - Manages a shared set of application data
    - Data typically stored persistently in an SQLite database
    - Never accessed directly, but via a Content Resolver

---

developer.android.com/guide/topics/providers/content-providers.html has more
Key Types of Android Components

- **Activity**
  - Represents a single screen with a user interface

- **Service**
  - Runs in background to perform long-running operations or to access remote resources

- **Content Provider**
  - Manages a shared set of application data

- **Broadcast Receiver**
  - A component that responds to system-wide Intent broadcast announcements

---

**Download Service**

2: Sends GET request to web server
3: Stores downloaded image in filesystem & metadata in Content Provider
4. Broadcasts intent containing image URI
5: Activity receives intent & displays image
Key Types of Android Components

- **Activity**
  - Represents a single screen with a user interface

- **Service**
  - Runs in background to perform long-running operations or to access remote resources

- **Content Provider**
  - Manages a shared set of application data

- **Broadcast Receiver**
  - A component that responds to system-wide Intent broadcast announcements
    - Supports complex Intent filtering

---

See developer.android.com/reference/android/content/BroadcastReceiver.html
Summary

- Android defines a *layered* software stack for mobile devices

[Diagram of Android software stack]

sites.google.com/site/io/anatomy--physiology-of-an-android has Android overview
Summary

• Android defines a layered software stack for mobile devices

• Apps are developed using framework components that Android can instantiate & run as needed

See developer.android.com/guide/components/fundamentals.html for more info
Summary

- Android defines a *layered* software stack for mobile devices.
- Apps are developed using framework components that Android can instantiate & run as needed.
- Most parts of Android are available in open-source format.

See [source.android.com](http://source.android.com) for instructions on how to obtain Android source code.