CS E-76:
Building Mobile Applications

Android Primer

TF: Sophie Chang

Setup

- Required Software
  - JDK (http://www.oracle.com/technetwork/java/javase/downloads/index.html)
  - Eclipse (www.eclipse.org - Eclipse Classic)
  - Android SDK (http://developer.android.com/sdk/installing.html)
- More instructions
  - http://cdn.cs76.net/2012/spring/projects/android-setup/android-setup.pdf

Creating a New Project

- Open Eclipse: File > New > Project > Android Project
- Give your project a name (no spaces and start with a capital letter), Create a target, create a package name
  - Traditional java package naming conventions (eg com.sophiechang.section.android)
- Leave Min SDK Version blank or give an API Level number (eg 10, not 2.3.3)

System Architecture

- Application (the apps themselves including phone, email)
- Application Layer (views, content providers, resources)
- Dalvik VM and Java subset (core libraries)
- Native, C/C++ libraries (WebKit, OpenGL, SQLite)
- Linux kernel (memory/process management, drivers, etc)

Packaging an App

- Compile – javac (.java files to .class/Java bytecode)
- Convert - dx (Java bytecode -> Dalvik bytecode)
  - Converts between two instruction sets
- Create – aapt (creates an Android package (.apk))
  - Resulting archive is zip compatible
- Made easy with Eclipse ☺

Resources

- .apk package also includes non-code files (eg images)
- res folder – contains subdirectories for different types of files
  - res/drawable: images (png, jpg, etc)
  - res/layouts: layouts
  - res/values/strings.xml: defined constants to be used by the app
- All resources accessed via R class (R.java)

Application Components

- Activities – a single “screen” on the app
  - UI elements created within activities within views
- Services – code without a visual component that runs in the background
- Broadcast Receivers – respond to system events (ie low battery)
- Content Providers – make app data accessible to other apps

Lifecycles of an Activity

- Running
  - Currently at the foreground and has focus
- Paused
  - Lost focus but still visible to the user
- Stopped
  - No longer visible to the user
- Keep in mind!
  - Activities that are not running can be stopped at any time due to low memory

Lifestyles of an Activity (cont'd)

- onCreate() – called right when activity is being started, do all initialization here
- onStart() – about to be displayed to the user
- onRestart() – stopped -> running
- onPause() – running -> pause
- onResume() – paused -> running
- onStop() – paused -> stopped
- onDestroy() – process killed (from stopped/ paused)

Create an Activity

- Activity has a Java file to define behaviors and an associated XML file to define layout
  - Can define layout in Java file or in the XML
- Add new java file to package
  - Right click package -> New -> Class
    - Use Command(Ctrl)+Shift+O or wait for Eclipse error to import necessary Android packages and extend Activity class
- Add layout file
- Add activity to AndroidMainfest.xml

Intents

- Activities, services, broadcast receivers triggered by intents
  - Intent – object containing an operation to be performed
- Intent i = new Intent(Context context, Class class);
- Can start a new activity with an intent
  - startActivity(i);

Layout

- Every GUI element is represented by an XML element
  - Attributes specify properties of the element
- GUI elements must be encapsulated in layouts
  - FrameLayout – contains a single layout
  - LinearLayout – elements arranged horizontally or vertically
  - TableLayout – like HTML Table
  - RelativeLayout – elements arranged relative to others

Activities and Tasks

- App can consist of many activities
  - Users navigate among activities, app maintains internal stack of them
- Task – sequence of activities among potentially different apps
  - User can proceed from one app to another (emailing a photo)
  - Using intent filters – select app they want to use to complete a step in a task
  - Supports multitasking

AVDs

- AVD (Android virtual device) – Android emulator that allows your to simulate a specific device
  - Hardware Profile: Amount of memory on simulator
  - System Image: What version of Android
  - Dedicated Storage: Installed apps, settings
  - Other Options: Screen Dimensions
- From Eclipse: Window -> Android SDK and AVD Manager

AVDs (cont'd)

- Creation
  - Eclipse: Window -> Android SDK and AVD Manager -> New
  - Run android command from sdk/tools
- Specify which AVD to use when debugging in Eclipse
  - Eclipse: Run -> Run Configurations -> Android Application

Running on an AVD

- Running an Android project will start the emulator automatically
  - Start emulator by running emulator –avd <avd name>
- Simulator startup is very slow
  - Simulating the hardware of the phone (Slow process!)
- Eclipse will just use active simulator (if there is one)
  - Don’t quit the simulator unless you like to procrastinate and wait for it to load