### **Audacious** Android Application Programming Workshop

Dr. Frank McCown
Computer Science Dept.
Harding University
Searcy, Arkansas, USA

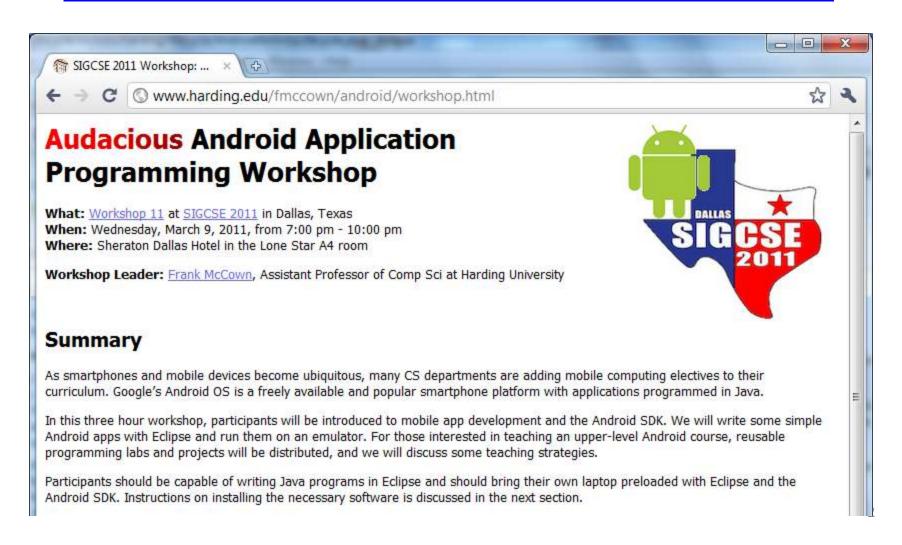
HARDING

March 9, 2011



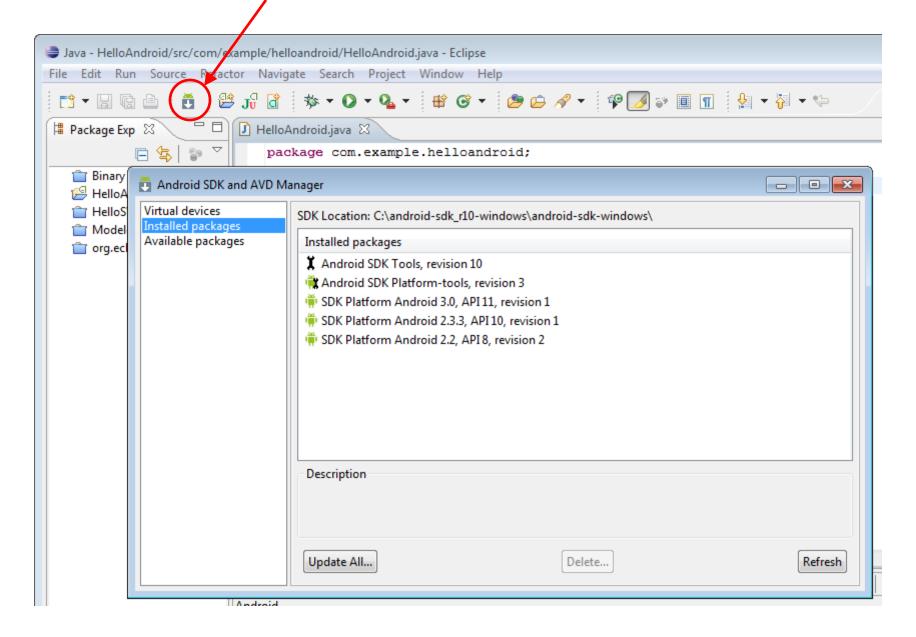
#### Workshop Website

http://www.harding.edu/fmccown/android/workshop.html



#### Setup Development Environment

- Install JDK 6
- Install Eclipse IDE for Java EE Developers (version 3.5 or 3.6)
- Download and unpack the Android SDK
- Install Android Development Tools (ADT) plugin for Eclipse
- Using the Android SDK and AVD Manager, add the SDK Platform Android 2.2, API 8 package
- Detailed install instructions available on Android site <a href="http://developer.android.com/sdk/installing.html">http://developer.android.com/sdk/installing.html</a>



#### Agenda

- Introduction to Android
- Android emulator basics
- Mobile devices and applications
- Simple Android app

#### 10 minute break

- Android activities and intents
- Tic-tac-toe example
- User interface issues
- Wrap-up

#### Why Mobile App Development?

- Mobile platform is the platform of the future
  - Double-digit growth in world-wide smartphone ownership<sup>3</sup>
- Job market is hot
  - Market for mobile software surges from \$4.1 billion in 2009 to \$17.5 billion by 2012<sup>1</sup>
  - 2010 Dice.com survey: 72% of recruiters looking for iPhone app developers, 60% for Android<sup>1</sup>
  - Dice.com: mobile app developers made \$85,000 in 2010 and salaries expected to rise<sup>2</sup>
- Students are naturally interested!

<sup>&</sup>lt;sup>1</sup> http://www.businessweek.com/technology/content/oct2010/tc20101020\_639668.htm

<sup>&</sup>lt;sup>2</sup> http://it-jobs.fins.com/Articles/SB129606993144879991/Mobile-App-Developers-Wanted-at-Ad-Agencies

<sup>3</sup>http://www.gartner.com/it/page.jsp?id=1466313



#### Why Android?

- A lot of students have them
  - 2010 survey by Univ of CO¹: 22% of college students have Android phone (26% Blackberry, 40% iPhone)
  - Gartner survey<sup>2</sup>: Android used on 22.7% of smartphones sold world-wide in 2010 (37.6% Symbian, 15.7% iOS)
- Students already know Java and Eclipse
  - Low learning curve
  - CSO students can use App Inventor for Android

#### Why Android?

- Transferring app to phone is trivial
  - Can distribute by putting it on the web
  - Android Market for wider distribution
- It's not 1984



Image: <a href="http://www.youtube.com/watch?v=Pm8iTUI-MvU">http://www.youtube.com/watch?v=Pm8iTUI-MvU</a>

# Types of Android Devices



#### HTC Droid Incredible

#### Motorola Droid





#### **Galaxy Tablet**

- Retail: \$250 with 2 year contract
- 7-inch 1024 x 600-pixel LCD screen
- 0.84 pounds
- 7.48 x 4.74 x 0.47 inches
- 1 GHz Coretx A8 processor
- 512 MB RAM
- 16 or 32 GB internal storage
- Front and rear cameras

#### Android-Powered Microwave



By Touch Revolution – at CES 2010

http://www.pocket-lint.com/news/30712/android-powered-microwave-cooking-google

#### Google Nexus One



- Retail: \$530
- Not tied to single provider
- 3.7-inch 800 x 400-pixel OLED screen
- No support for multitouch
- 512 MB of built-in flash memory
- Preloaded 4 GB SD card
- Ubiquitous voice recognition
- 5-megapixel camera with zoom and flash
- Navigation system using Google Maps and GPS

#### **Brief History**

#### • 2005

- Google acquires startup Android Inc. to start Android platform
- Work on Dalvik VM begins

#### • 2007

- Open Handset Alliance announced
- Early look at SDK

#### • 2008

- Google sponsors 1<sup>st</sup> Android Developer Challenge
- T-Mobile G1 announced
- SDK 1.0 released
- Android released open source (Apache License)
- Android Dev Phone 1 released

#### Brief History cont.

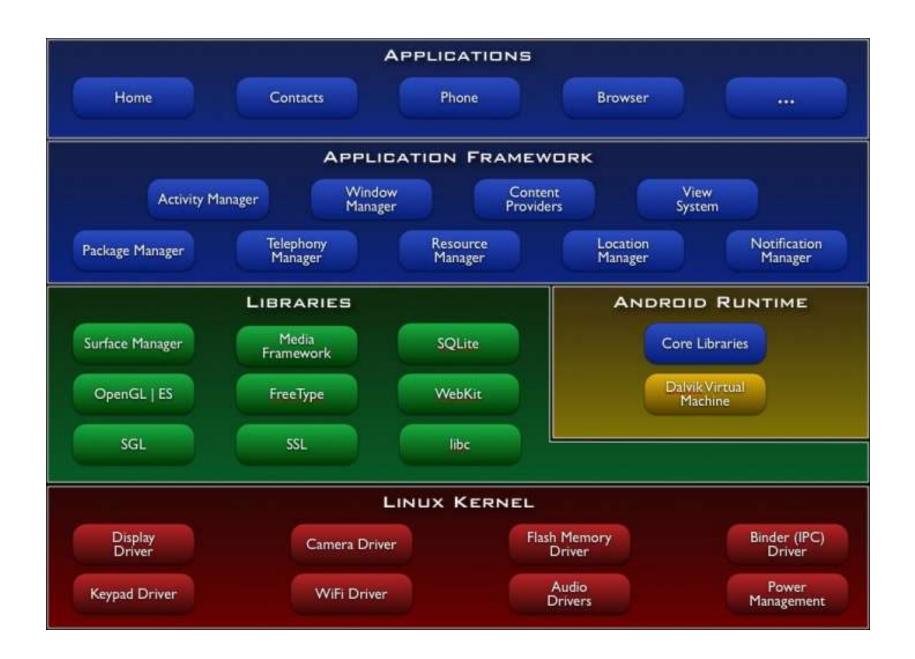
- 2009
  - SDK 1.5 (Cupcake)
    - New soft keyboard with "autocomplete" feature
  - SDK 1.6 (Donut)
    - Support Wide VGA
  - SDK 2.0/2.0.1/2.1 (Eclair)
    - Revamped UI, browser
- 2010
  - Nexus One released to the public
  - SDK 2.2 (Froyo)
    - Flash support, tethering
  - SDK 2.3 (Gingerbread)
    - UI update, system-wide copy-paste

#### Brief History cont.

- 2011
  - SDK 3.0 (Honeycomb) for tablets only
    - New UI for tablets, support multi-core processors

#### What is Google Android?

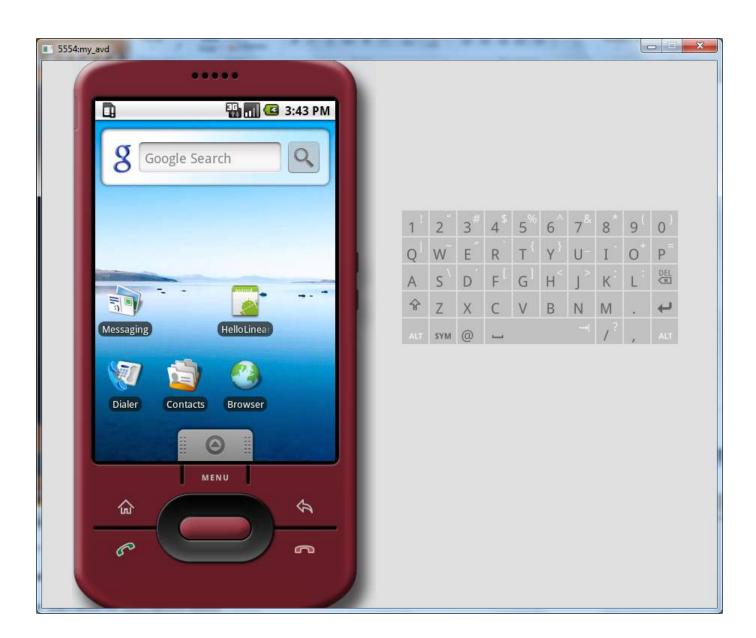
- A software stack for mobile devices that includes
  - An operating system
  - Middleware
  - Key Applications
- Uses Linux to provide core system services
  - Security
  - Memory management
  - Process management
  - Power management
  - Hardware drivers



#### Android Emulator or AVD

- Emulator is essential to testing app but is not a substitute for a real device
- Emulators are called Android Virtual Devices (AVDs)
- Android SDK and AVD Manager allows you to create AVDs that target any Android API level
- AVD have configurable resolutions, RAM, SD cards, skins, and other hardware

#### Android Emulator: 1.6 Device



#### Android Emulator: 2.2 Device



#### Android Emulator: 3.0 Device



#### **Emulator Basics**

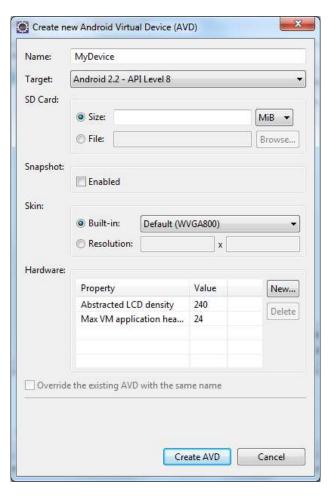
- Host computer's keyboard works
- Host's mouse works like finger
- Uses host's Internet connection
- Side buttons work: Home, Menu, Back, Search, volume up and down, etc.
- Ctrl-F11 toggle landscape → portrait
- Alt-Enter toggle full-screen mode
- More info at <a href="http://developer.android.com/guide/developing/devices/emulator.html">http://developer.android.com/guide/developing/devices/emulator.html</a>

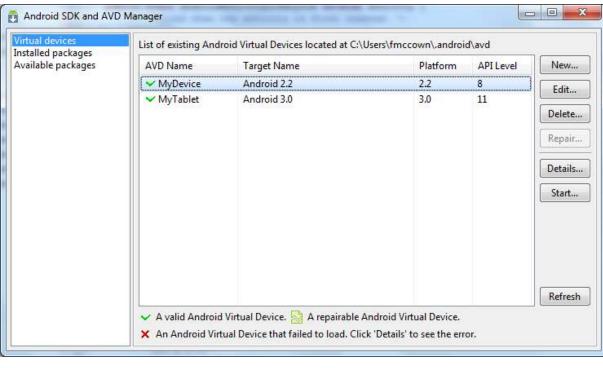
#### **Emulator Limitations**

- No support for placing or receiving actual phone calls
  - Simulate phone calls (placed and received) through the emulator console
- No support for USB connections
- No support for camera/video capture (input)
- No support for device-attached headphones
- No support for determining connected state
- No support for determining battery charge level and AC charging state
- No support for determining SD card insert/eject
- No support for Bluetooth
- No support for simulating the accelerometer
  - Use OpenIntents's Sensor Simulator

In other words, test your app on an actual device!

## Create an AVD using Android SDK and AVD Manager





#### Or From the Command Line

C:\android-sdk-windows\tools>android create avd -n MyDevice -t android-8

Android 2.2 is a basic Android platform.

Do you wish to create a custom hardware profile [no]

Created AVD 'MyDevice2' based on Android 2.2,

with the following hardware config:

hw.lcd.density=240

vm.heapSize=24

C:\android-sdk-windows\tools>emulator -avd MyDevice



More info:

http://developer.android.com/guide/developing/devices/managing-avds-cmdline.html

# Play Time: Get to know your Emulator

#### Mobile Devices: Advantages

- Always with the user
- Typically have Internet access
- Typically GPS enabled
- Typically have accelerometer & compass
- Many have cameras & microphones
- Many apps are free or low-cost

#### Mobile Devices: Disadvantages

- Limited screen size
- Limited battery life
- Limited processor speed
- Limited and sometimes slow network access
- Limited or awkward input: soft keyboard, phone keypad, touch screen, or stylus
- Limited web browser functionality
- Range of platforms & configurations across devices

#### Mobile Applications

- What are they?
  - Any application that runs on a mobile device
- Types
  - Web apps: run in a web browser
    - HTML, JavaScript, Flash, server-side components, etc.
  - Native: compiled binaries for the device
    - Often make use of web services

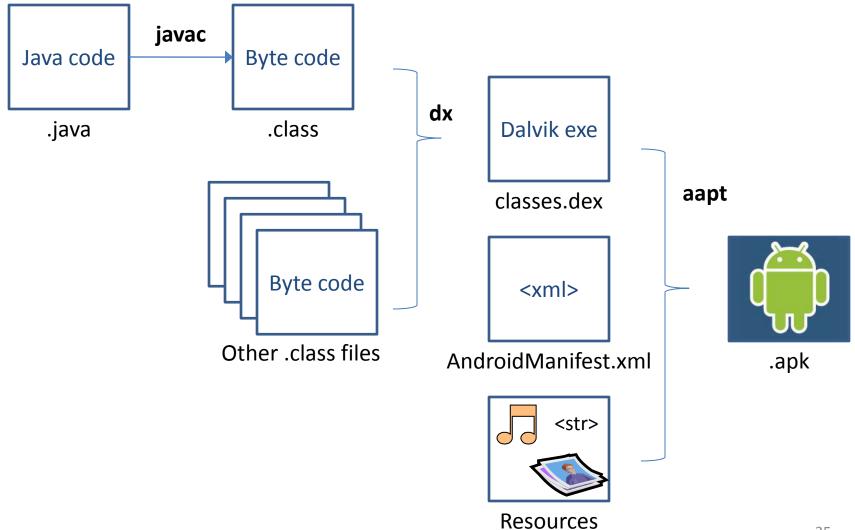
#### **Android Apps**

- Built using Java and new SDK libraries
  - No support for some Java libraries like Swing & AWT
  - Oracle currently suing Google over use
- Java code compiled into Dalvik byte code (.dex)
  - Optimized for mobile devices (better memory management, battery utilization, etc.)
- Dalvik VM runs .dex files

#### **Applications Are Boxed**

- By default, each app is run in its own Linux process
  - Process started when app's code needs to be executed
  - Threads can be started to handle time-consuming operations
- Each process has its own Dalvik VM
- By default, each app is assigned unique Linux ID
  - Permissions are set so app's files are only visible to that app

#### Producing an Android App



#### **Hello Android Tutorial**

HelloAndroid com.example.helloandroid MelloAndroid.java gen [Generated Java Files] tom.example.helloandroid R.java Android 1.1 📒 assets 🗸 造 res 🔣 icon.png layout

x main.xml

X strings.xml

AndroidManifest.xml

default.properties

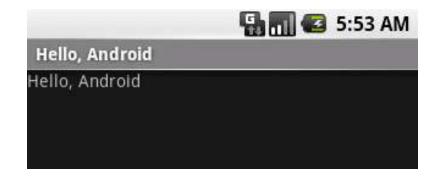
values

```
package com.example.helloandroid;

import android.app.Activity;
import android.os.Bundle;

public class HelloAndroid extends Activity {

    /** Called when the activity is first created. */
@Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```



# **Important Files**

- src/HelloAndroid.java
  - Activity which is started when app executes
- res/layout/main.xml
  - Defines & lays out widgets for the activity
- res/values/strings.xml
  - String constants used by app
- gen/R.java (Don't touch!)
  - Auto-generated file with identifiers from main.xml, strings.xml, and elsewhere
- AndroidManifest.xml
  - Declares all the app's components
  - Names libraries app needs to be linked against
  - Identifies permissions the app expects to be granted

# src/HelloAndroid.java

Activity which is started when app executes

```
package com.example.helloandroid;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
public class HelloAndroid extends Activity {
    private int mHelloCount = 0;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
```

# res/layout/main.xml

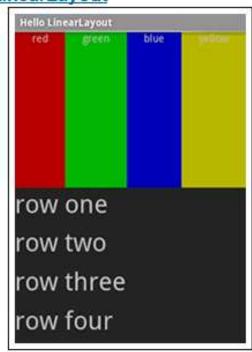
Declares layouts & widgets for the activity

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"</pre>
    android:orientation="vertical"
    android:layout width="fill parent"
    android:layout height="fill parent" >
    <EditText
                                                                              ViewGroup
        android:id="@+id/name"
        android:layout width="fill parent"
        android:layout height="wrap_content"
        android:text="@string/hello" />
                                                                      ViewGroup
                                                                                    View
                                                                                            View
    <Button
        android:id="@+id/hello button"
        android:layout height="wrap content"
                                                               View
                                                                        View
                                                                                 View
        android:layout width="wrap content"
        android:text="Press Me" />
</LinearLavout>
```

Tree from: http://developer.android.com/guide/topics/ui/index.html

## **Various Layouts**

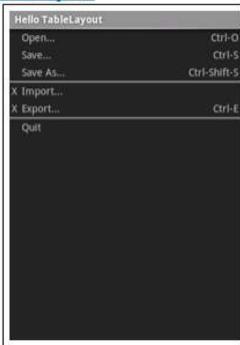
# LinearLayout



### RelativeLayout



### **TableLayout**



# Various Widgets

#### MapView



#### WebView



### **DatePicker**



#### Spinner



### **AutoComplete**



### ListView



# res/values/strings.xml

String constants used by app

- Used for supporting Localization
  - res/values-es/values/strings.xml to support Spanish
  - res/values-fr/values/strings.xml to support French
  - Etc.

# gen/R.java

 Auto-generated file with identifiers from main.xml, strings.xml, and elsewhere

```
public final class R {
    public static final class attr {
    public static final class drawable {
        public static final int icon=0x7f020000;
    public static final class id {
        public static final int hello button=0x7f050001;
        public static final int my button=0x7f050003;
        public static final int my check box=0x7f050002;
       public static final int name=0x7f050000;
   public static final class layout {
        public static final int main=0x7f030000;
       public static final int second=0x7f030001;
    public static final class string {
        public static final int app name=0x7f040001;
       public static final int hello=0x7f040000;
```

Do not modify!

## AndroidManifest.xml

- Declares all the app's components
- Names libraries app needs to be linked against
- Identifies permissions the app expects to be granted

# Four Application Components

### 1. Activities

- Presents a visual UI for a single endeavor
- Single app may be composed of several activities
- Examples: list of photos, buttons to start/stop a song

### 2. Services

- Performs background work (no UI)
- Examples: play background music, retrieve data over a network

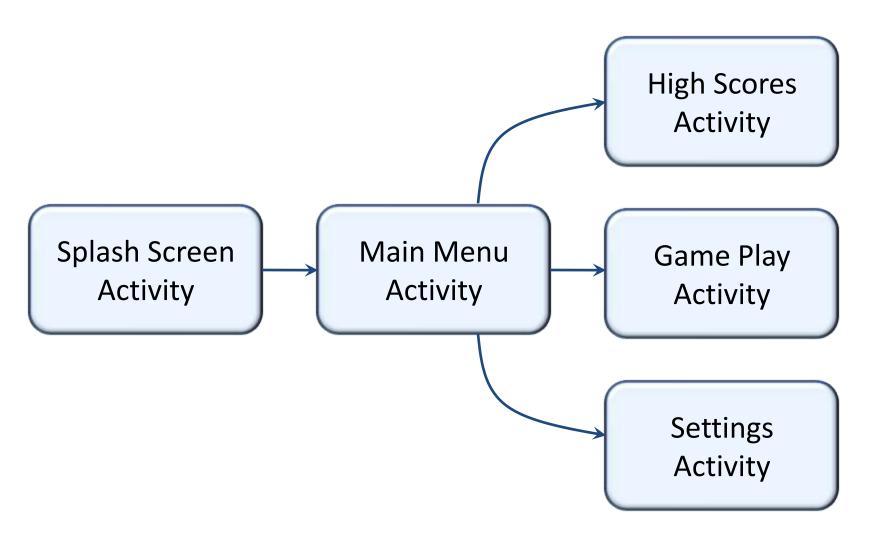
### 3. Broadcast Receivers

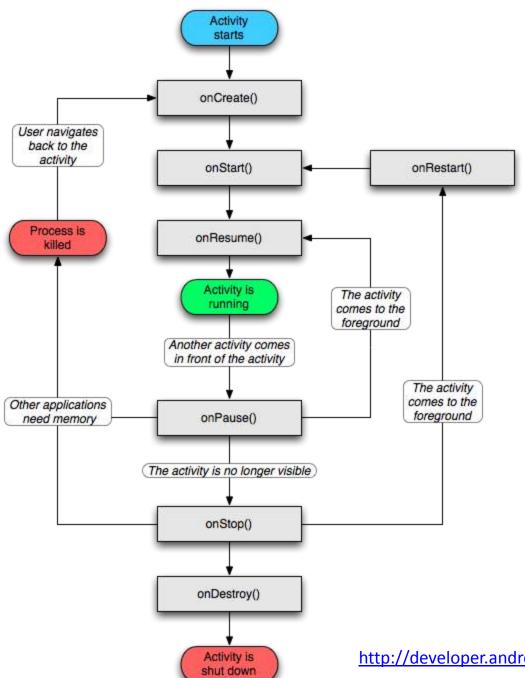
- Receives and reacts to broadcast announcements (no UI)
- Broadcast examples: battery is low, pic is taken, lang pref changed

### 4. Content Providers

- Provides app data to other applications (no UI)
- $-\,$  Examples: share contact info from SQLite, image from the file system  $_{45}$

# **Typical Game**





# Activity Lifecycle

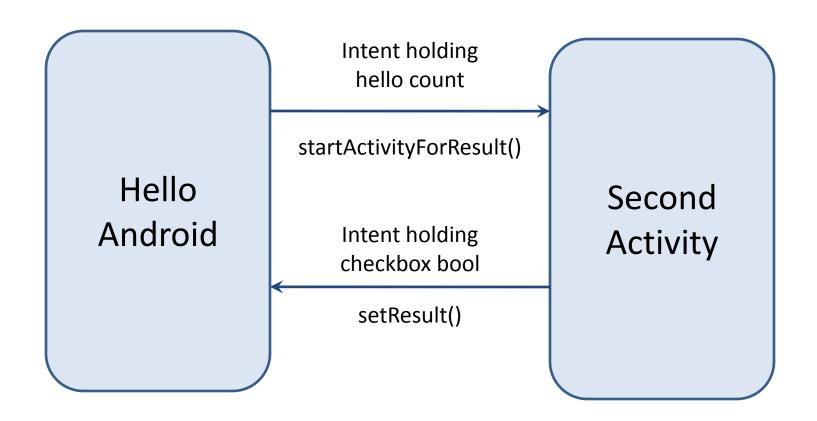
http://developer.android.com/reference/android/app/Activity.html

# Lifecycle Demo

### Intents

- Activities, services, & broadcast receivers are activated by intents
  - Intents are asynchronous messages
  - May include data to be passed between components
  - Example: A Service starts an Activity to pick a photo by using an intent. The photo is returned to the Service also using an intent.

## Intent Demo



## Data Storage

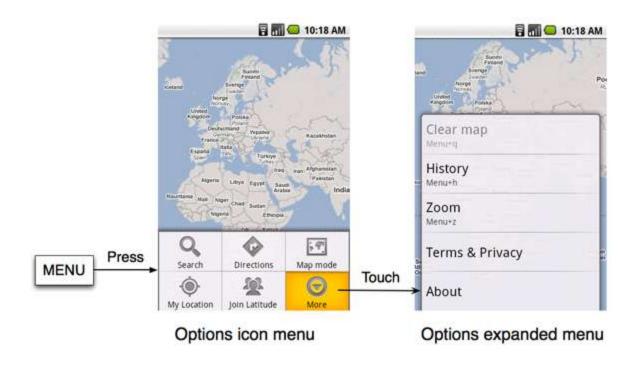
- App's data is private
  - Can be shared using content providers
- Four ways to store data:
  - 1. Preferences: Lightweight mechanism to store and retrieve key-value pairs of primitive data types
  - 2. Files: Store on mobile device or on a removable storage medium
  - 3. Databases: SQLite
  - 4. Network: Store/retrieve data stored elsewhere

### Common Interface Issues

- Menus
  - Options menu
  - Context menu
- Touch gestures
  - Tap/Touch
  - Long press
  - Double tap
  - Pinch & spread
  - Flick/swipe
  - Drag
- Responsiveness

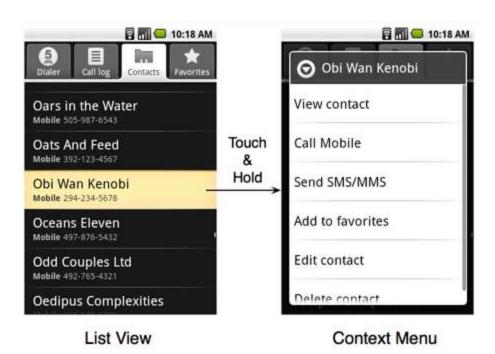
## **Options Menu**

- Activated by pressing MENU button
- More than 6 items show in expanded menu



### Context Menu

- Activated by long touch on item
- Most intuitive command should be listed first
- Many users never see or use them, so items should usually be visible elsewhere





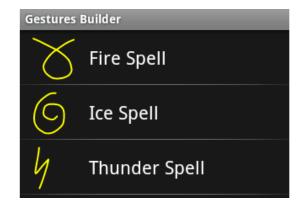
Context Menu for Text

54

## **Touch Gestures**

- Tap or touch
  - Most common action
  - Used to select item
- Long press
  - Launch context menu or other intuitive action
- Double tap
  - Toggle zoom-in and zoomout
- Pinch & spread
  - Zoom-in and out

- Flick or swipe
  - Switch between screens or scroll content
- Drag
  - Scroll, move items, or draw a path
- Custom gestures
  - android.gesture package



By Craig Villamor, Dan Willis, and Luke Wroblewski Last updated April 15, 2010

### CORE GESTURES Basic gestures for most touch commands

### Tap



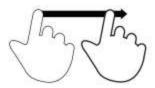
Briefly touch surface with fingertip

### Double tap



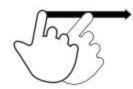
Rapidly touch surface twice with fingertip

### Drag



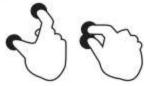
Move fingertip over surface without losing contact

### Flick

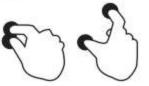


Quickly brush surface with fingertip

### Pinch



### Spread



### Press



### Press and tap



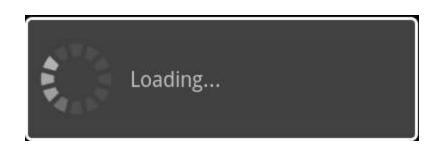
## Responsiveness

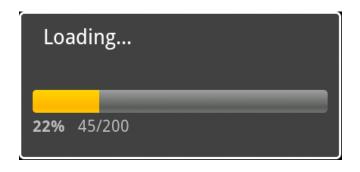
- Most important UI consideration is responsiveness
- If app ignores input event for 5 seconds, Android displays the dreaded Application Not Responding (ANR) dialog
- Often caused by performing lengthy operations like waiting for network response or AI to decide next move



# **Avoiding ANR**

- Most apps run on a single thread
- Any activity which may take a long time should be delegated to a child thread
- Use progress bars to give users feedback when lengthy activities are being performed





Short wait Long wait

58

## Tic-tac-toe



