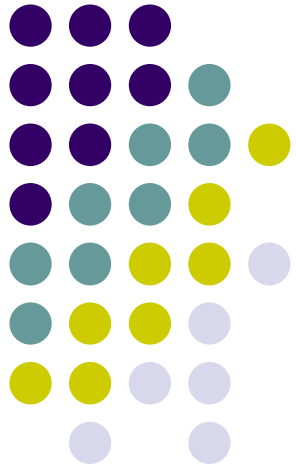
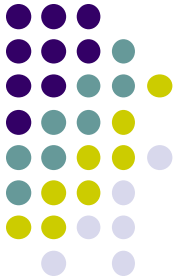
CS 4518 Mobile and Ubiquitous Computing
Lecture 2: Introduction to Android

Emmanuel Agu



What is Android?



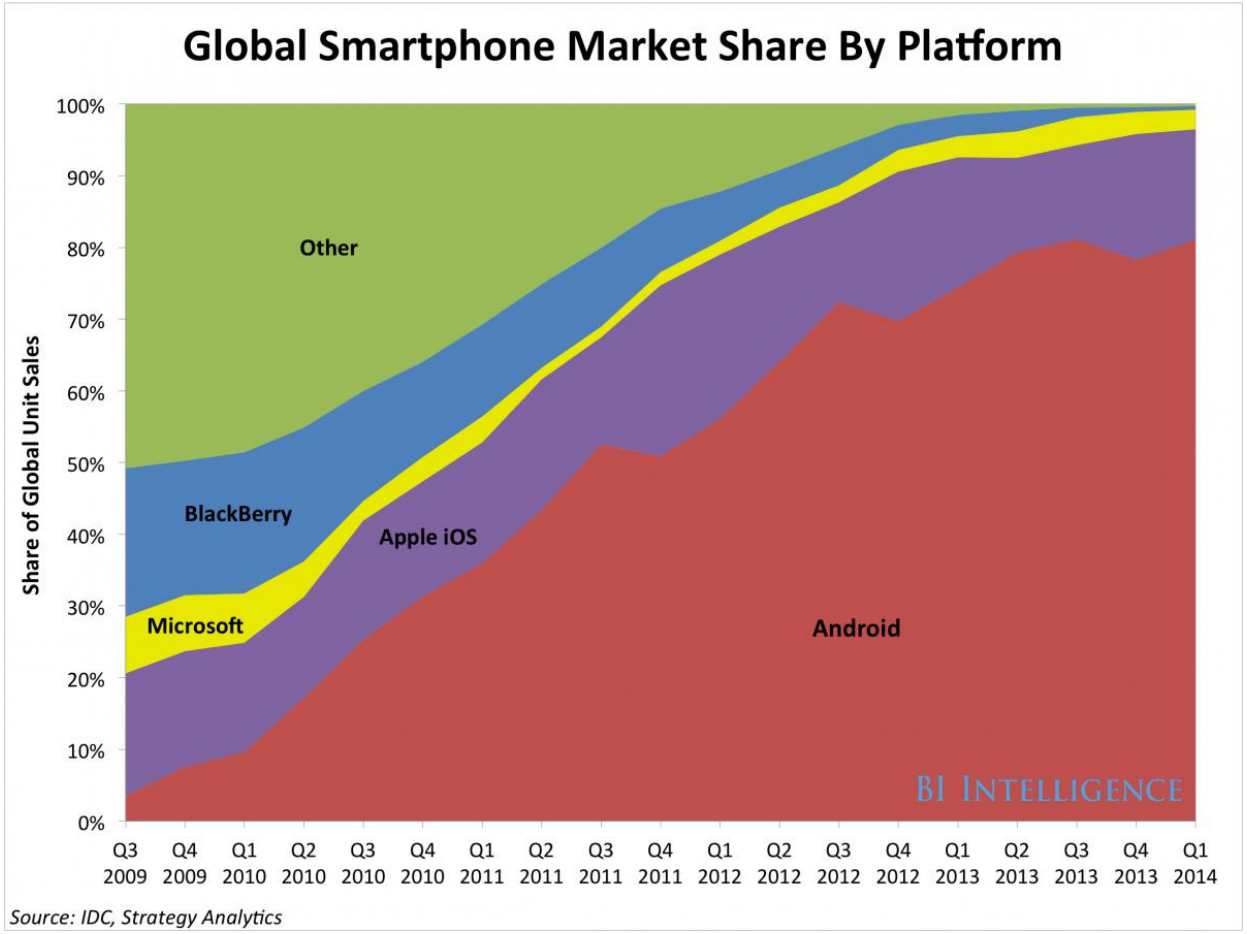
- Android is world's leading mobile operating system
  - Open source

- **Google:**
  - Owns Android, maintains it, extends it
  - Distributes Android OS, developer tools, free to use
  - Runs Android app market

SmartPhone OS

- Over 80% of all phones sold are smartphones
- Android share 86% worldwide

Source: IDC, Strategy Analytics



Android Growth

- October 2015, 1.4 billion Android users (ref: WSJ)
- 1.6 million apps on the Android app market (ref: statista.com)
  - Games, organizers, banking, entertainment, etc

Android is Multi-Platform

Google Glass
(being redone)

In-car console

Smartwatch

Android runs on
all these devices

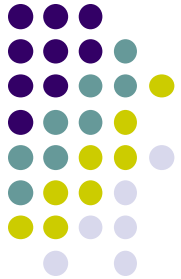
Tablet

Smartphone

This Class: Focuses
Mostly on Smartphones!

Television

Android for Mobile Computing and Ubicomp



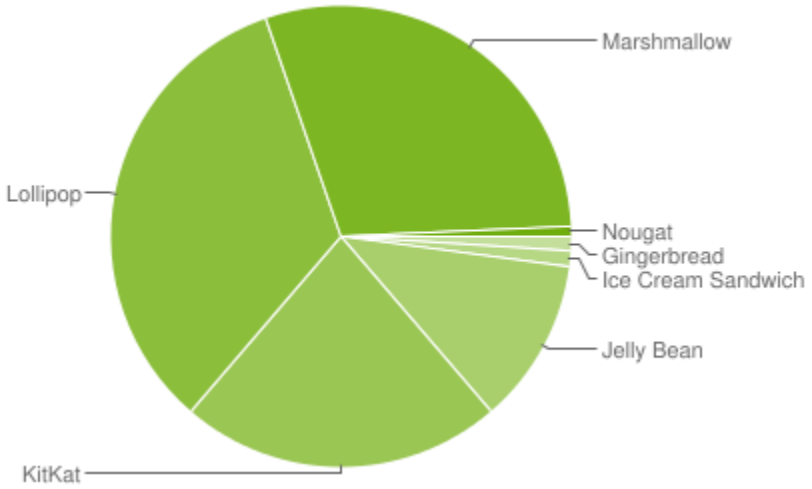
- Android for Mobile programmable modules
  - Audio/video playback, taking pictures, database, location detection, maps, enhanced User Interface

- Android for Ubicomp programmable modules
  - Sensors (temperature, humidity, light, etc), proximity
  - Face detection, activity recognition, place detection, speech recognition, speech-to-text, gesture detection, place type understanding, etc
  - Machine learning, deep learning

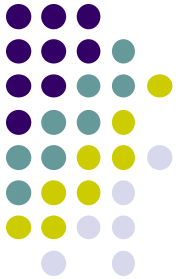
Android Versions

- Most recent Android version is Android (7.1.1) or “Nougat”
- Officially released December 5, 2016
- This class will use Android 5.0 (lollipop)
- Below is Android version distribution as at January 9, 2016

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	1.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.1%
4.1.x	Jelly Bean	16	4.0%
4.2.x		17	5.9%
4.3		18	1.7%
4.4	KitKat	19	22.6%
5.0	Lollipop	21	10.1%
5.1		22	23.3%
6.0	Marshmallow	23	29.6%
7.0	Nougat	24	0.5%
7.1		25	0.2%



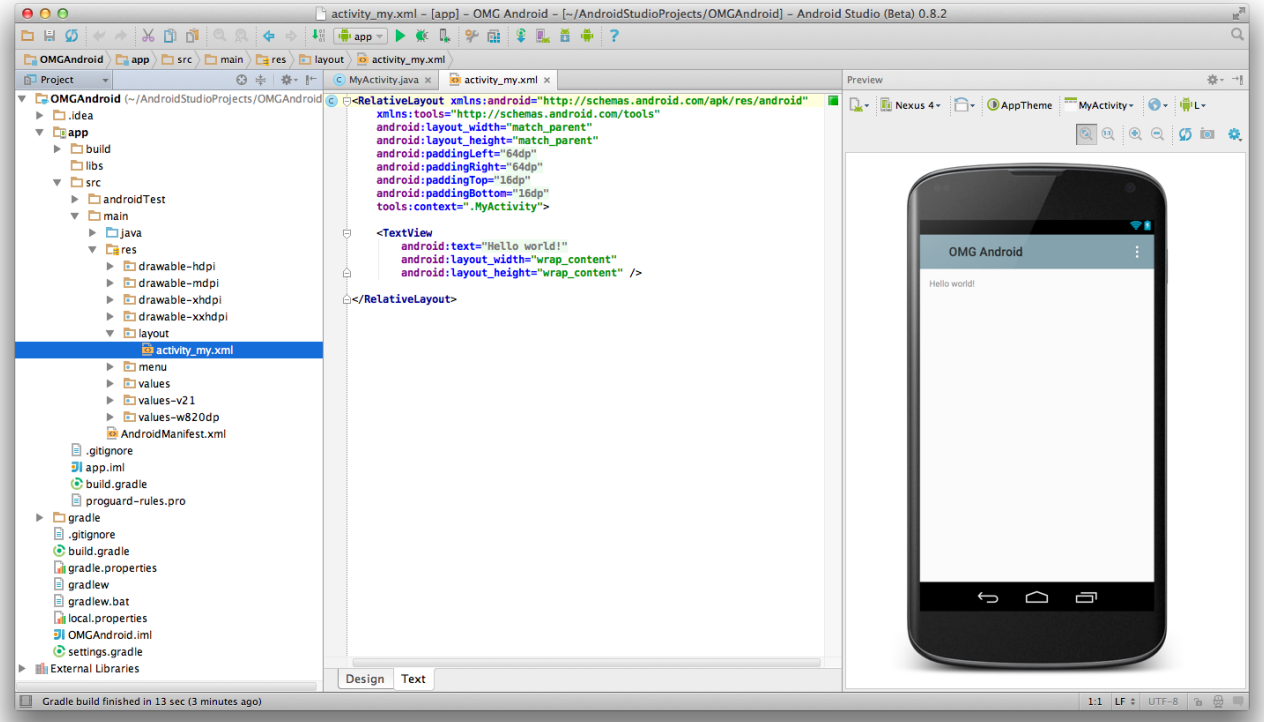
Source: http://developer.android.com/about/dashboards/index.html



Android Developer Environment

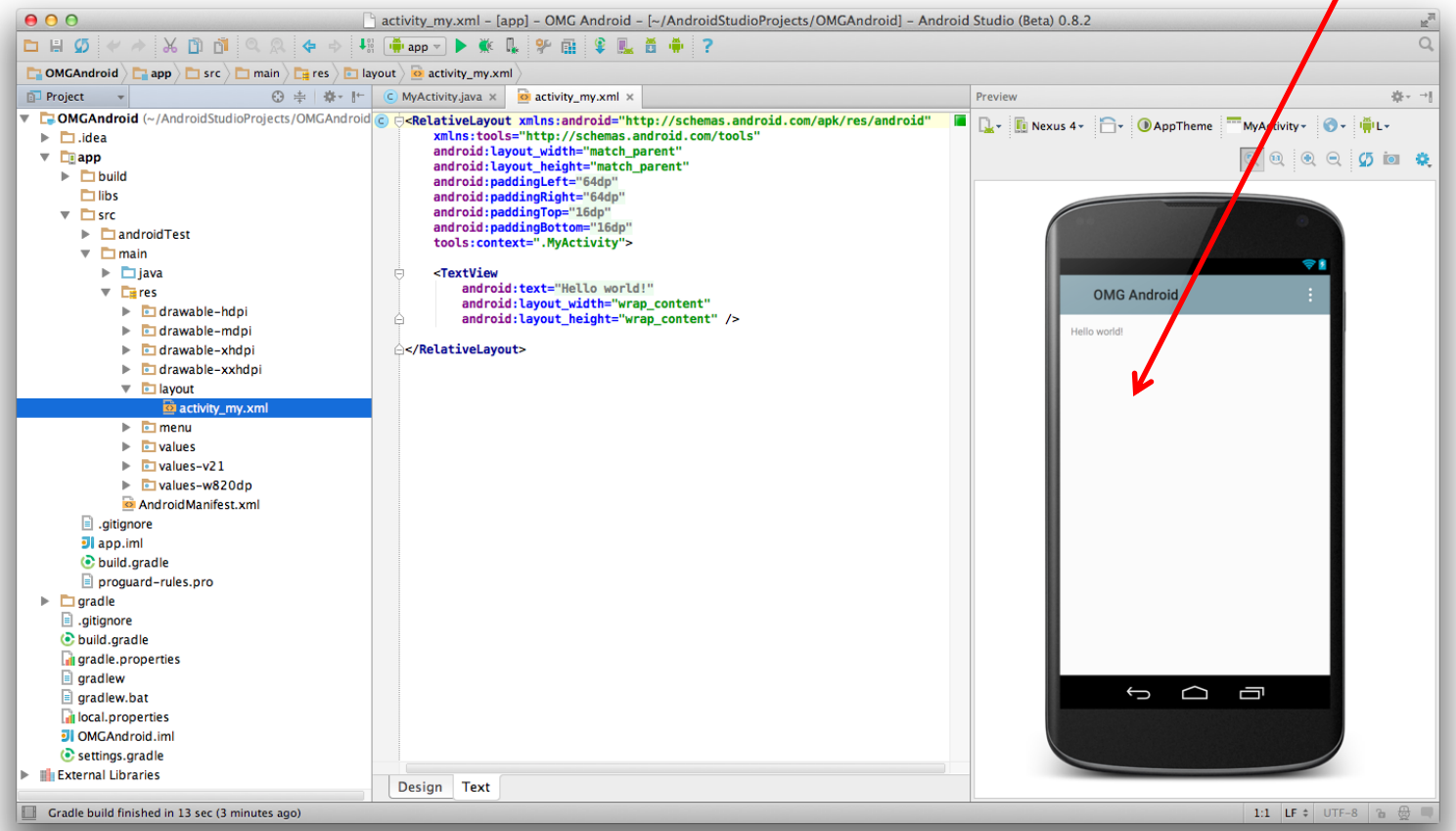
New Android Environment: Android Studio

- Old Android dev environment used **Eclipse + plugins**
- Google developed it's own IDE called **Android Studio**
- Integrated development environment, cleaner interface, specifically for Android Development (e.g. drag and drop app design)
- In December 2014, Google announced it will stop supporting Eclipse IDE



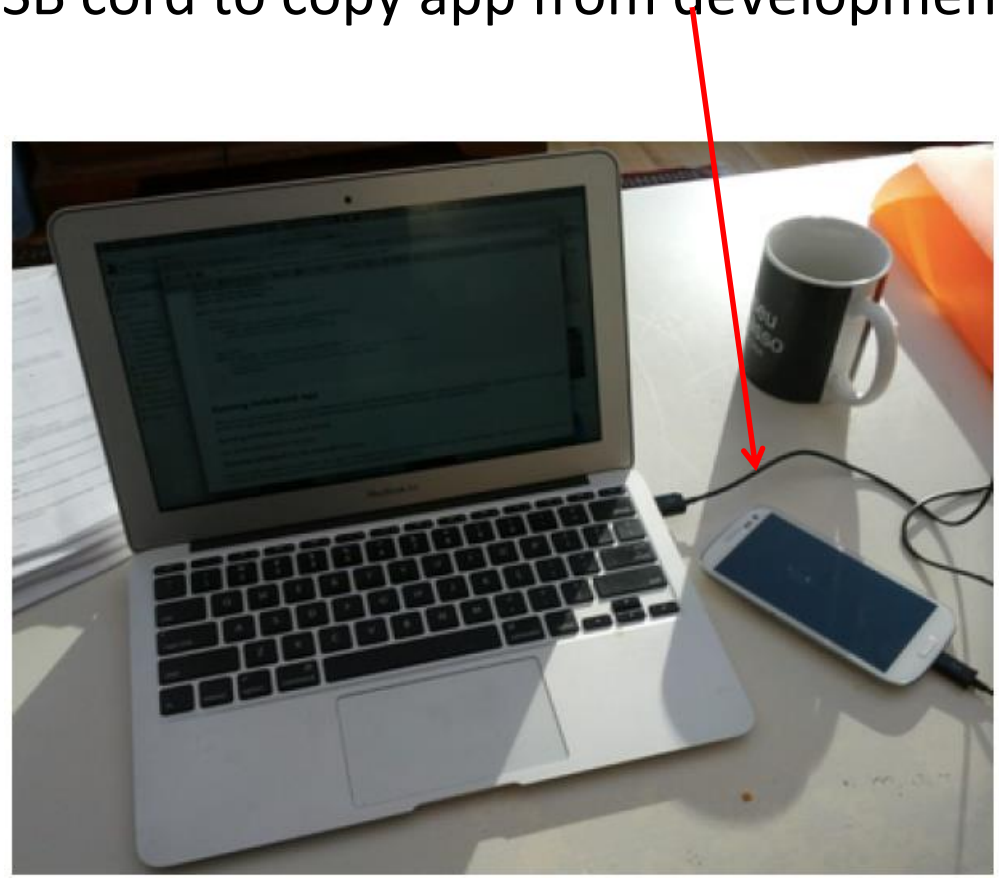
Where to Run Android App

- Android app can run on:
  - Real phone (or device)
  - Emulator (software version of phone)

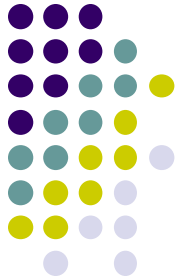


Running Android App on Real Phone

- Need USB cord to copy app from development PC to phone

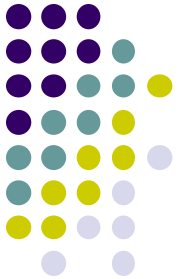


Emulator Pros and Cons (Vs Real Phone)

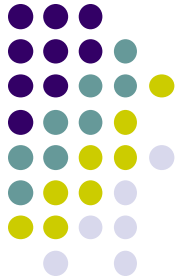


- Pros:
  - Conveniently test app on basic hardware by clicking in software
  - Easy to test app on various emulated devices (phones, tablets, TVs, etc), various screen sizes
- Cons:
  - Access to certain hardware, communications, sensors missing
  - E.g. GPS, camera, video recording, making/receiving phone calls, Bluetooth devices, USB devices, battery level, sensors, etc
  - Slower than real phone

HW0: Tutorials from YouTube Android Development Tutorials 1-8 by Bucky Roberts



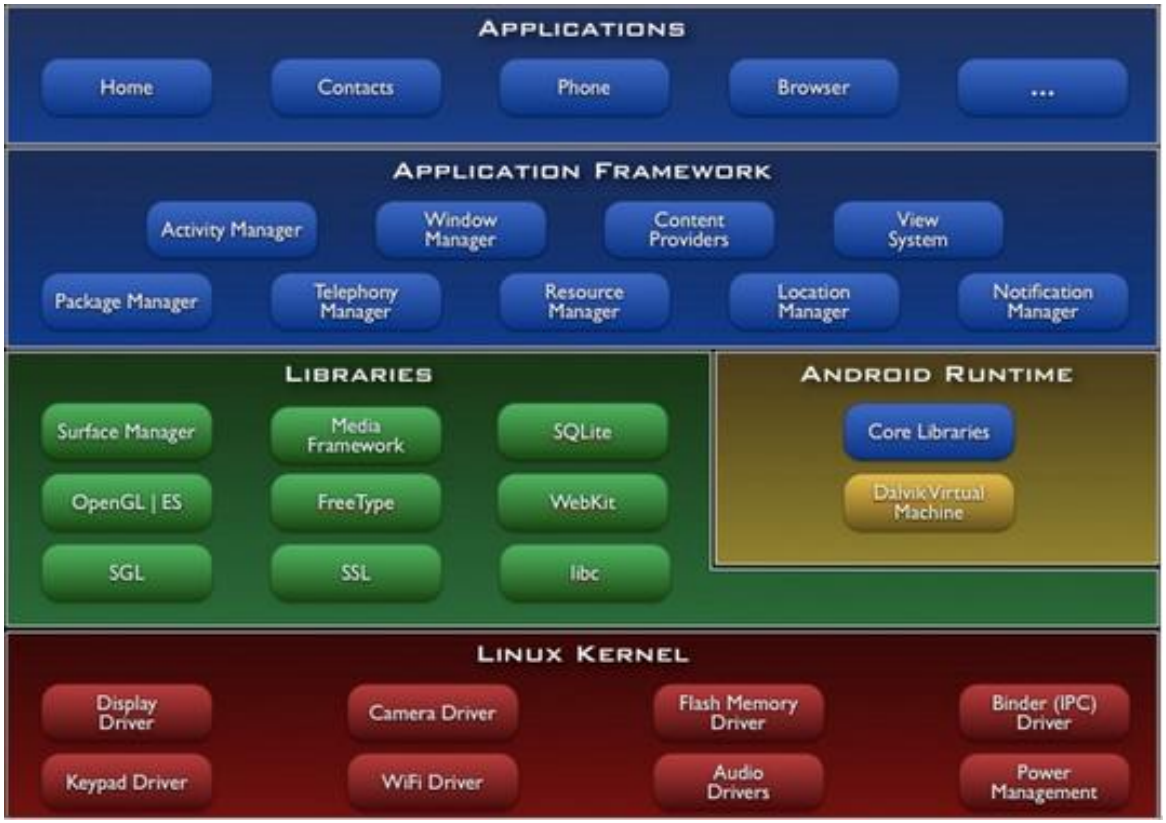
- **Tutorials 1 & 2 (Optional):** Installing Java, Android Studio on your own machine
  - **Tutorial 1:** Install Java (Android studio needs this at least ver. 1.8)
  - **Tutorial 2:** Install Android Studio
- **Tutorial 3:** Setting up your project
  - How to set up a new Android Project, add new Activity (App screen)
- **Tutorial 4:** Running a Simple App
  - How to select, run app on a virtual device (AVD)
- **Tutorial 5:** Tour of Android Studio Interface
  - Intro to Android Studio menus, toolbars and Drag-and-drop widget palette



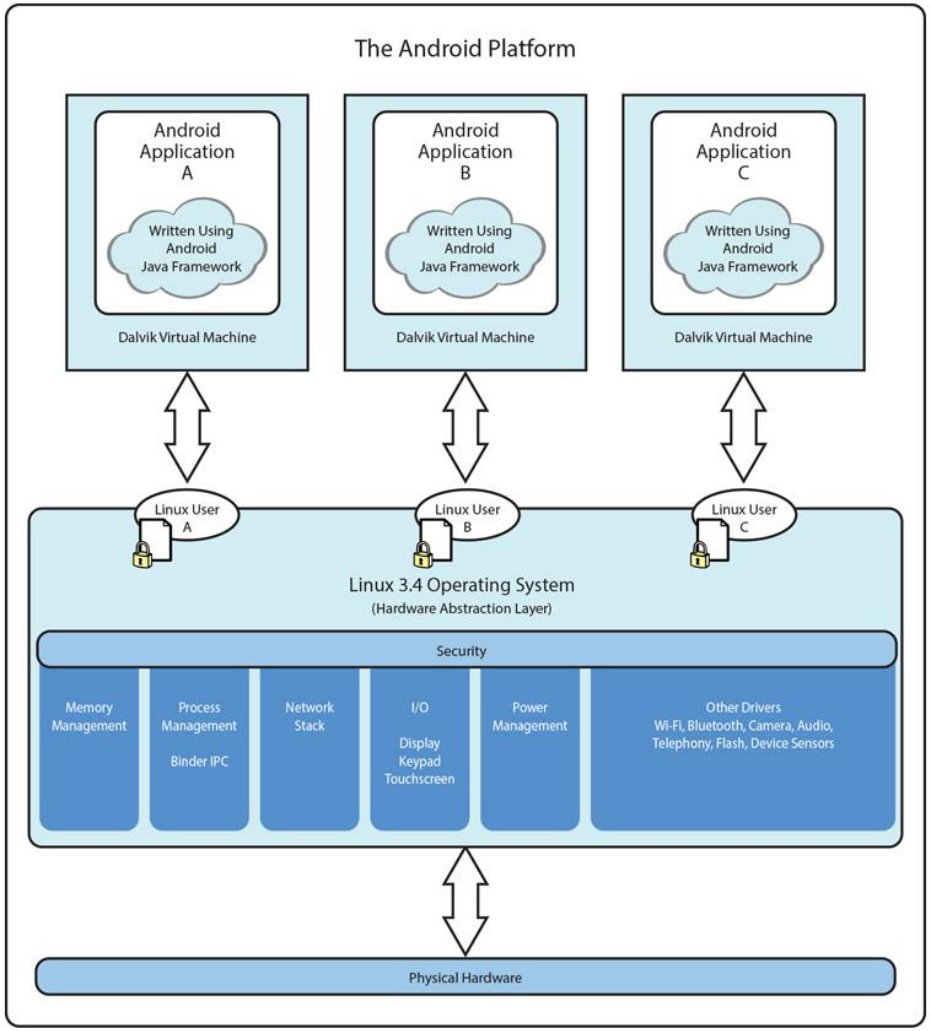
Android Software Framework

Android Software Framework

- **OS:** has Linux kernel, drivers
- **Apps:** programmed in Java
- **Libraries:** OpenGL ES (graphics), SQLite (database), etc

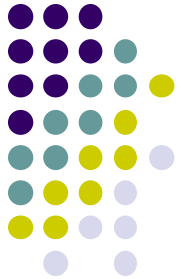


Android Software Framework



- Each Android app runs in its own security sandbox (VM, minimizes complete system crashes)
- Android OS multi-user Linux system
- Each app is a different user (assigned unique Linux ID)
- Access control: only process with the app's user ID can access its files

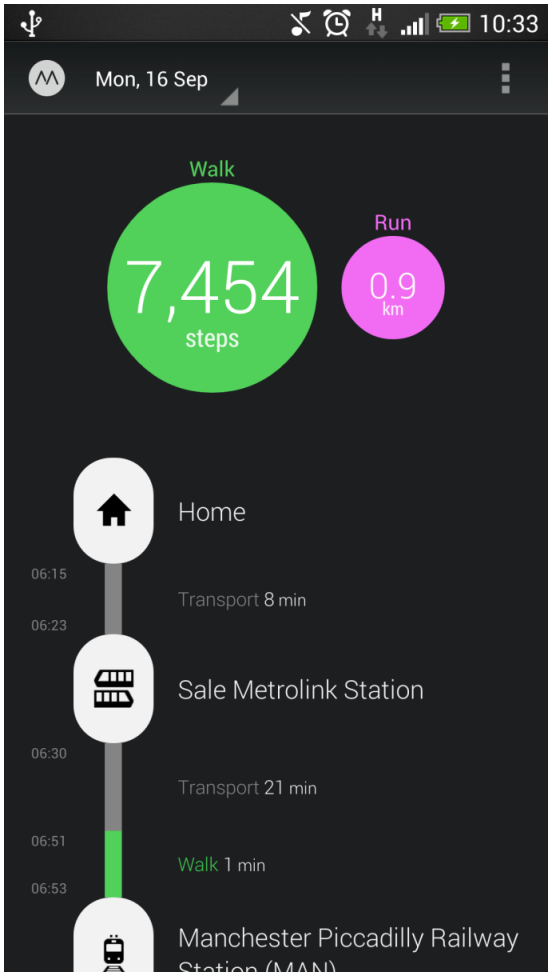
Ref: Introduction to Android Programming, Annuzzi, Darcey & Conder



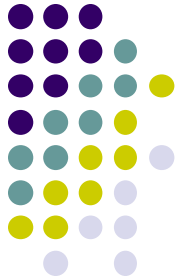
Android Apps: Big Picture

UI Design using XML

- UI design code (XML) separate from the program (Java)
- Why? Can modify UI without changing Java program
- **Example:** Shapes, colors can be changed in XML file without changing Java program
- UI designed using either:
  - Drag-and drop graphical (WYSIWYG) tool or
  - Programming Extensible Markup Language (XML)
- **XML:** Markup language, both human-readable and machine-readable''



Android App Compilation



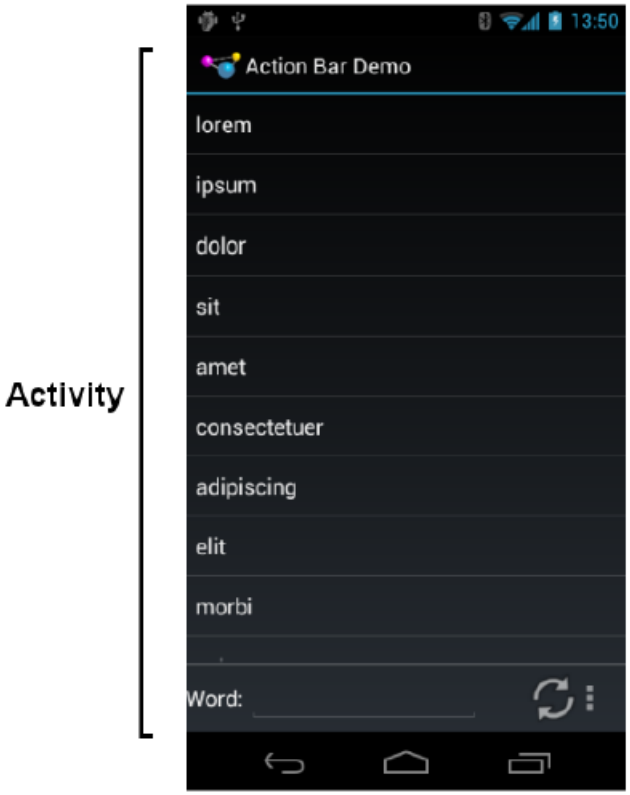
- Android Studio compiles code, data and resource files into **Android PacKage (filename.apk)**.
  - .apk is similar to .exe on Windows
- Apps download from Google Play, or copied to device as **filename.apk**
- Installation = installing **apk file**

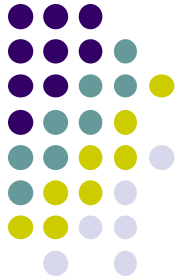
Activities

- Activity? 1 Android screen or dialog box
- Apps
  - Have at least 1 activity that deals with UI
  - Entry point, similar to **main( )** in C
  - Typically have multiple activities

- Example: A camera app
  - **Activity 1:** to focus, take photo, launch activity 2
  - **Activity 2:** to view photo, save it

- Activities
  - independent of each other
  - E.g. Activity 1 can write data, read by activity 2
  - App Activities derived from Android's **Activity** class





Our First Android App

3 Files in “Hello World” Android Project

- **Activity_my.xml:** XML file specifying screen layout

- **MainActivity.Java:** Java code to define behavior, actions taken when button clicked (intelligence)

- **AndroidManifest.xml:**
  - Lists all screens, components of app
  - Analogous to a table of contents for a book
  - E.g. Hello world program has 1 screen, so AndroidManifest.xml has 1 item listed
  - App starts running here (like main( ) in C)

- **Note:** Android Studio creates these 3 files for you

Execution Order

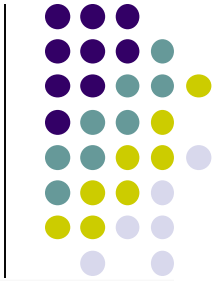
Next: Samples of AndroidManifest.xml
Hello World program

Start in AndroidManifest.xml
Read list of activities (screens)
Start execution from Activity tagged Launcher

Create/execute activities (declared in java files)
E.g. MainActivity.Java

Format each activity using layout In XML file (e.g. Activity_my.xml)

Inside “Hello World” AndroidManifest.xml



This file is written using xml namespace and tags and rules for android

Your package name

Android version

List of activities (screens) in your app

```
<?xml version="1.0"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.commonsware.android.skeleton"
  android:versionCode="1"
  android:versionName="1.0">

  <application>
    <activity
      android:name="Now"
      android:label="Now">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>

</manifest>
```

One activity (screen) designated LAUNCHER. The app starts running here

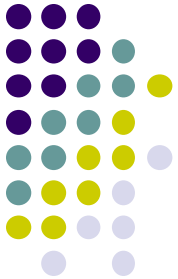
Execution Order

Start in AndroidManifest.xml
Read list of activities (screens)
Start execution from Activity tagged Launcher

Next → Create/execute activities (declared in java files)
E.g. MainActivity.Java

Format each activity using layout In XML file (e.g. Activity_my.xml)

Example Activity Java file (E.g. MainActivity.java)



Package declaration →

Import needed classes →

My class inherits from Android activity class →

Initialize by calling onCreate( ) method of base Activity class →

```
package com.commonsware.empublite;

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

public class EmPubLiteActivity extends Activity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
  }
}
```

Note: Android calls your Activity's onCreate method once it is created

Use screen layout (design) declared in file main.xml

Execution Order

Start in AndroidManifest.xml
Read list of activities (screens)
Start execution from Activity tagged Launcher

Create/execute activities (declared in java files)
E.g. MainActivity.Java

Format each activity using layout In XML file (e.g. Activity_my.xml)

Next

Simple XML file Designing UI

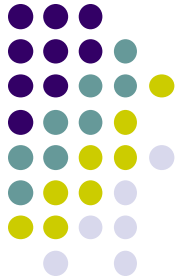
- After choosing the layout, then widgets added to design UI
- XML Layout files consist of:
  - UI components (boxes) called **Views**
  - Different types of views. E.g
    - **TextView:** contains text,
    - **ImageView:** picture,
    - **WebView:** web page
  - **Views** arranged into layouts or **ViewGroups**

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".EmPubLiteActivity">

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:text="@string/hello_world"/>

</RelativeLayout>
```

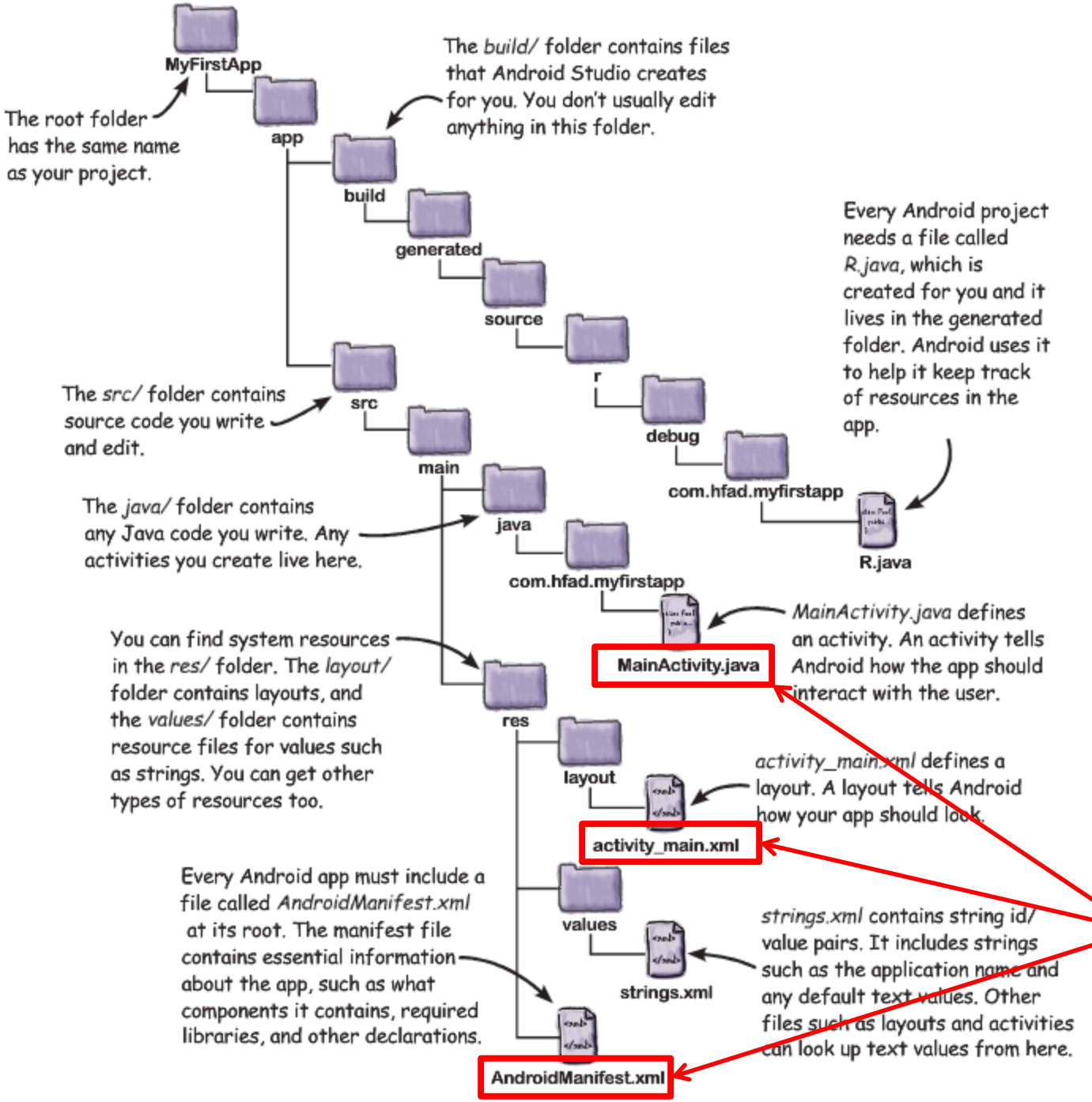
Declare Layout

Add widgets

Widget properties
(e.g. center contents
horizontally and vertically)



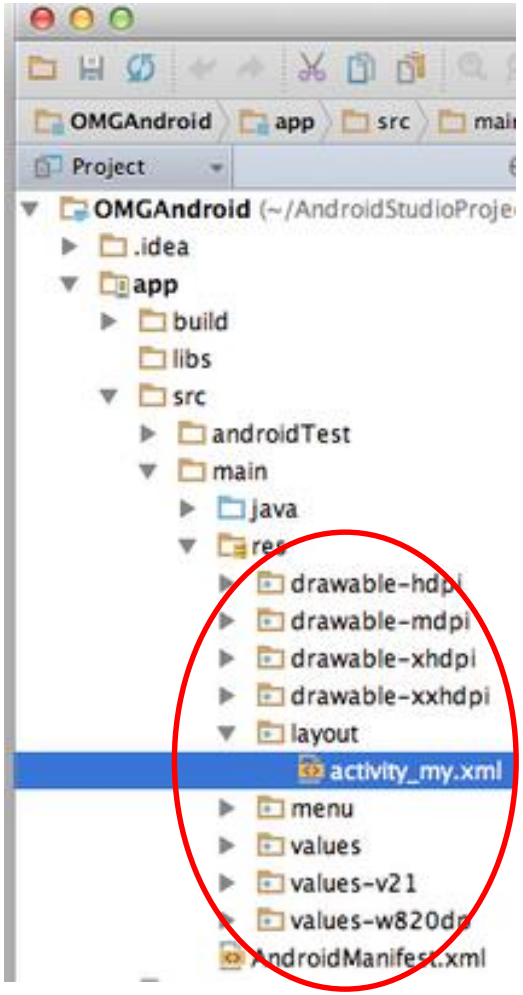
Android Files



Android Project File Structure

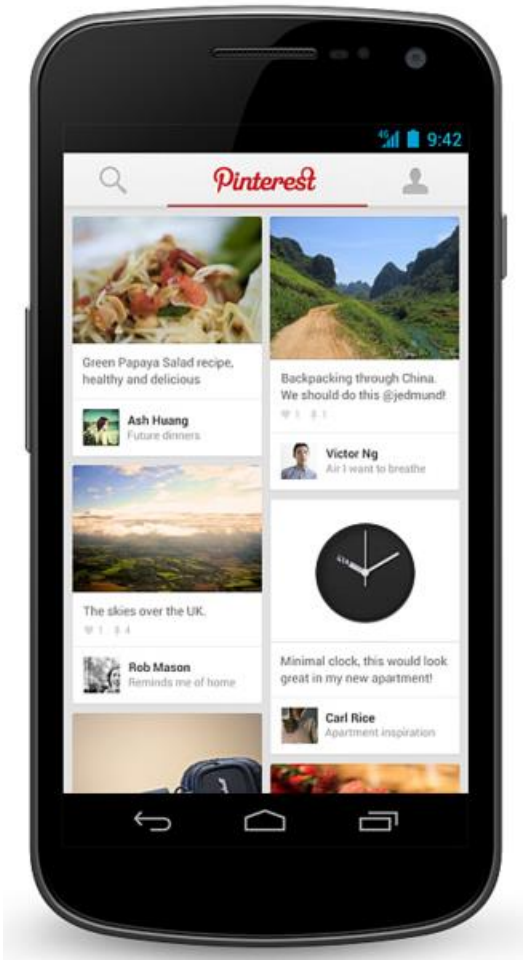
Files in an Android Project

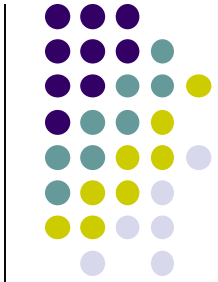
- **res/** (resources) folder contains static resources you can embed in Android screen (e.g. pictures, string declarations, etc)
- **res/menu/:** XML files for menu specs
- **res/drawable-xyz/:** images (PNG, JPEG, etc) at various resolutions
- **res/raw:** general-purpose files (e.g. audio clips, mpeg, video files, CSV files
- **res/values/:** strings, dimensions, etc



Concrete Example: Files in an Android Project

- **res/layout:** layout, dimensions (width, height) of screen cells are specified in XML file here
- **res/drawable-xyz/:** The images stored in jpg or other format here
- **java/:** App's response when user clicks on a selection is specified in java file here
- **AndroidManifext.XML:** Contains app name (Pinterest), list of app screens, etc

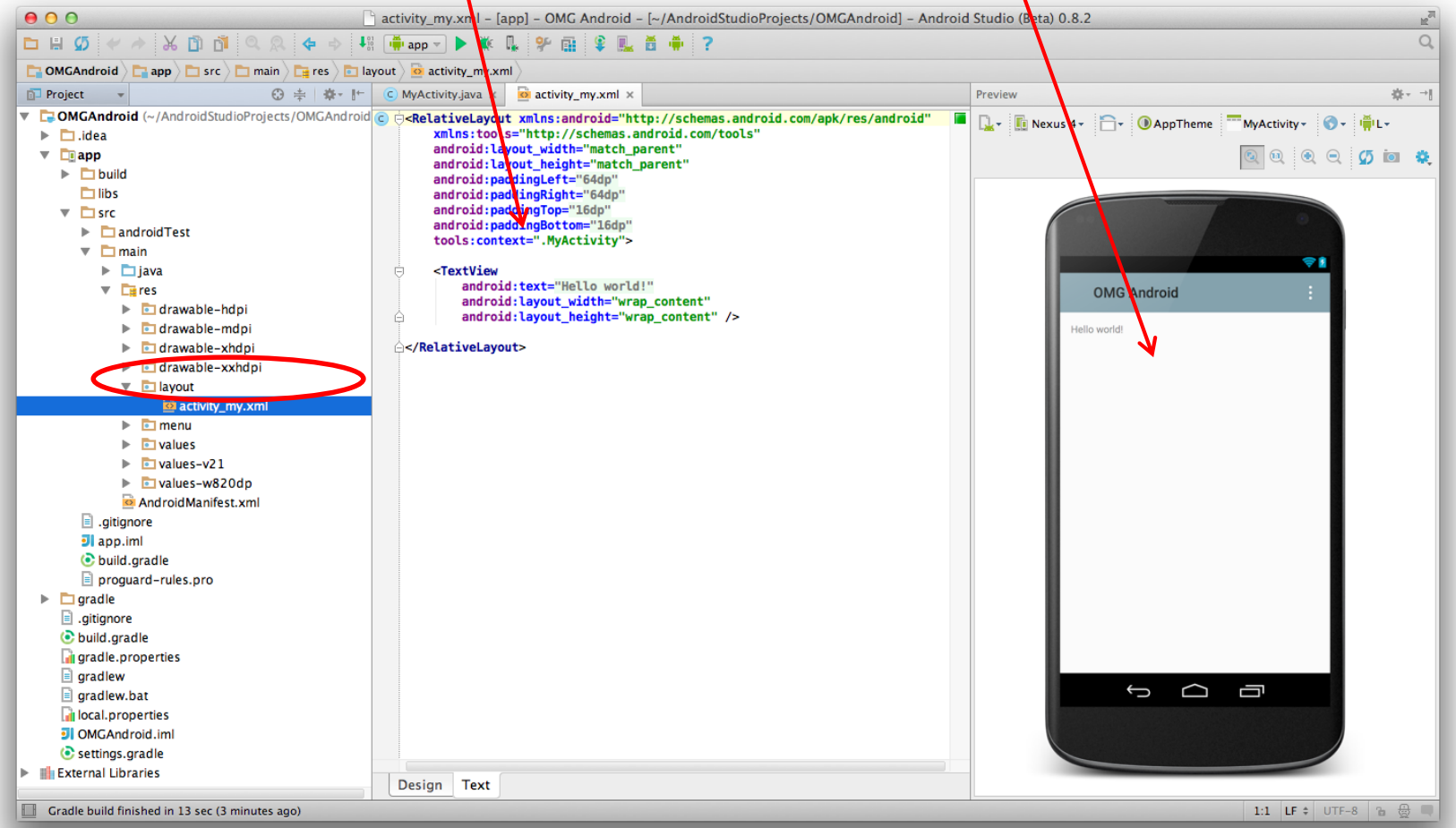


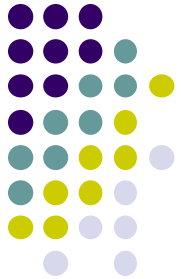


Editting in Android Studio

Editting Android

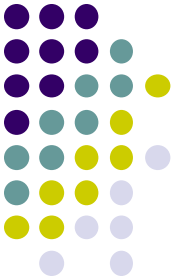
- Can edit apps in:
  - **Text View:** edit XML directly
  - **Design View:** or drag and drop widgets unto emulated phone





Resources

Declaring Strings in Strings.xml



- Can declare all strings in strings.xml

String declaration in strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">EmPubLite</string>
    <string name="hello_world">Hello world!</string>

</resources>
```

- Then reference in any of your app's xml files

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".EmPubLiteActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="@string/hello_world"/>

</RelativeLayout>
```

Strings in AndroidManifest.xml

- Strings declared in strings.xml can be referenced by all other XML files (activity_my.xml, AndroidManifest.xml)

String declaration in strings.xml

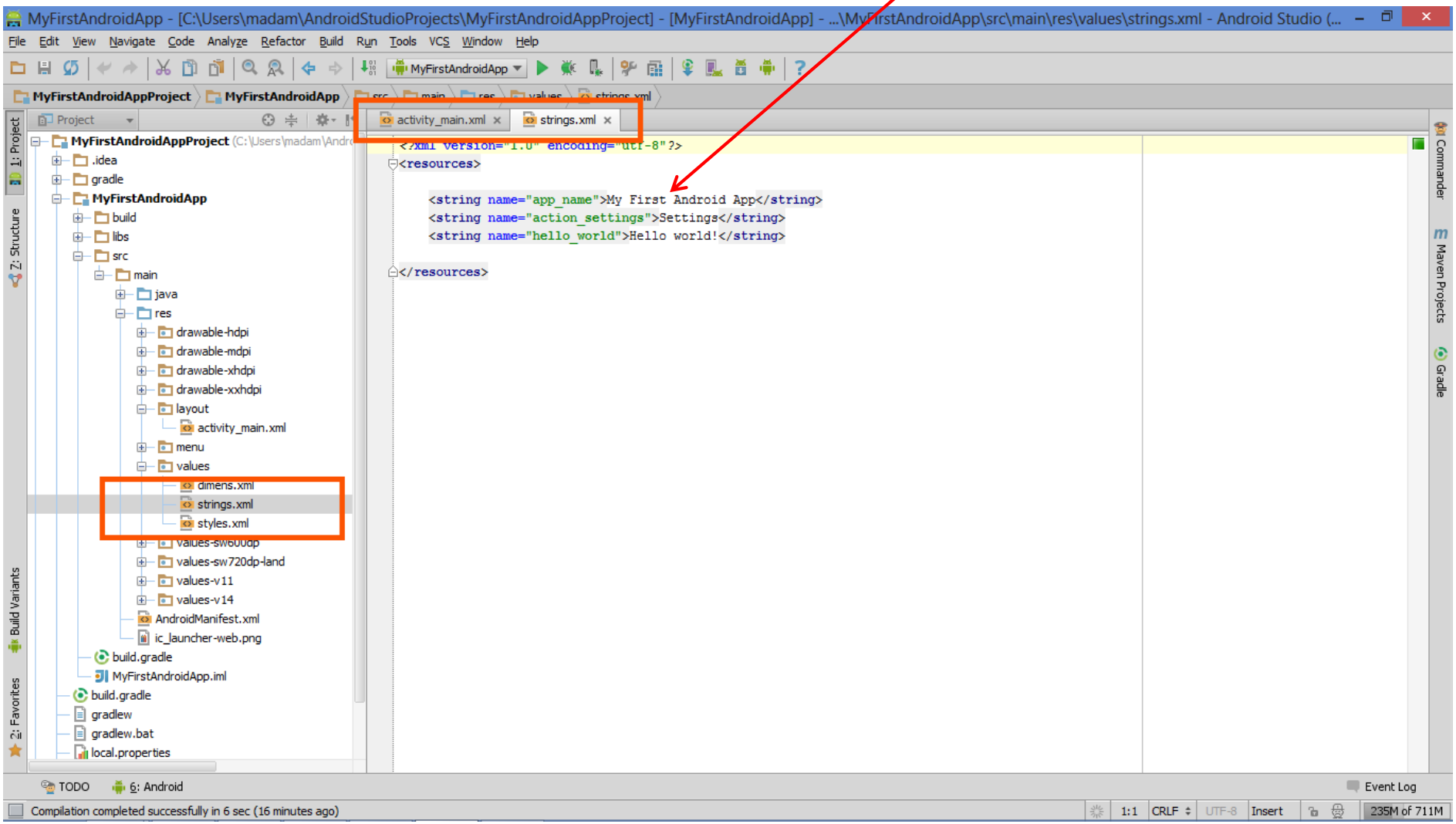
```
<?xml version="1.0" encoding="utf-8"?>
<resources>

  <string name="app_name">EmPubLite</string>
  <string name="hello_world">Hello world!</string>

</resources>
```

String usage in AndroidManifest.xml

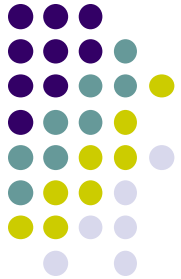
```
  <application
    android:allowBackup="false"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme">
    <activity
      android:name="EmPubLiteActivity"
      android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN"/>

        <category android:name="android.intent.category.LAUNCHER"/>
      </intent-filter>
    </activity>
  </application>

</manifest>
```

Where is strings.xml in Android Studio?

Editting any string in strings.xml changes it wherever it is displayed



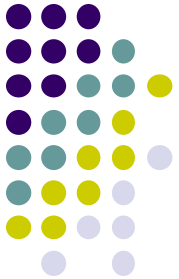
Styled Text



- In HTML, tags can be used for italics, bold, etc
  - E.g. Hello makes text *Hello*
  - <b> Hello <b> makes text **Hello**

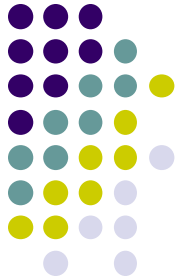
- Can use the same HTML tags to add style (italics, bold, etc) to Android strings

```
<resources>
  <string name="b">This has <b>bold</b> in it.</string>
  <string name="i">Whereas this has <i>italics</i>!</string>
</resources>
```



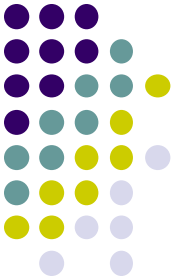
Quiz

Quiz 1



- No class next Monday (Martin Luther King holiday)
- Project 0: due next Thursday (Jan 19)
- Quiz in class, first 10 mins of class on Monday, Jan 23, 2016
- Quiz will be short, review style questions on
  - Concepts, definitions on today's slides
  - Hands-on Android question (e.g. from HW0)

References



- Android App Development for Beginners videos by Bucky Roberts (thenewboston)
- Ask A Dev, Android Wear: What Developers Need to Know, https://www.youtube.com/watch?v=zTS2NZpLyQg
- Ask A Dev, Mobile Minute: What to (Android) Wear, https://www.youtube.com/watch?v=n5Yjzn3b_aQ
- Busy Coder's guide to Android version 4.4
- CS 65/165 slides, Dartmouth College, Spring 2014
- CS 371M slides, U of Texas Austin, Spring 2014