CS 528 Mobile and Ubiquitous Computing
Lecture 01b: Introduction to Android

Emmanuel Agu

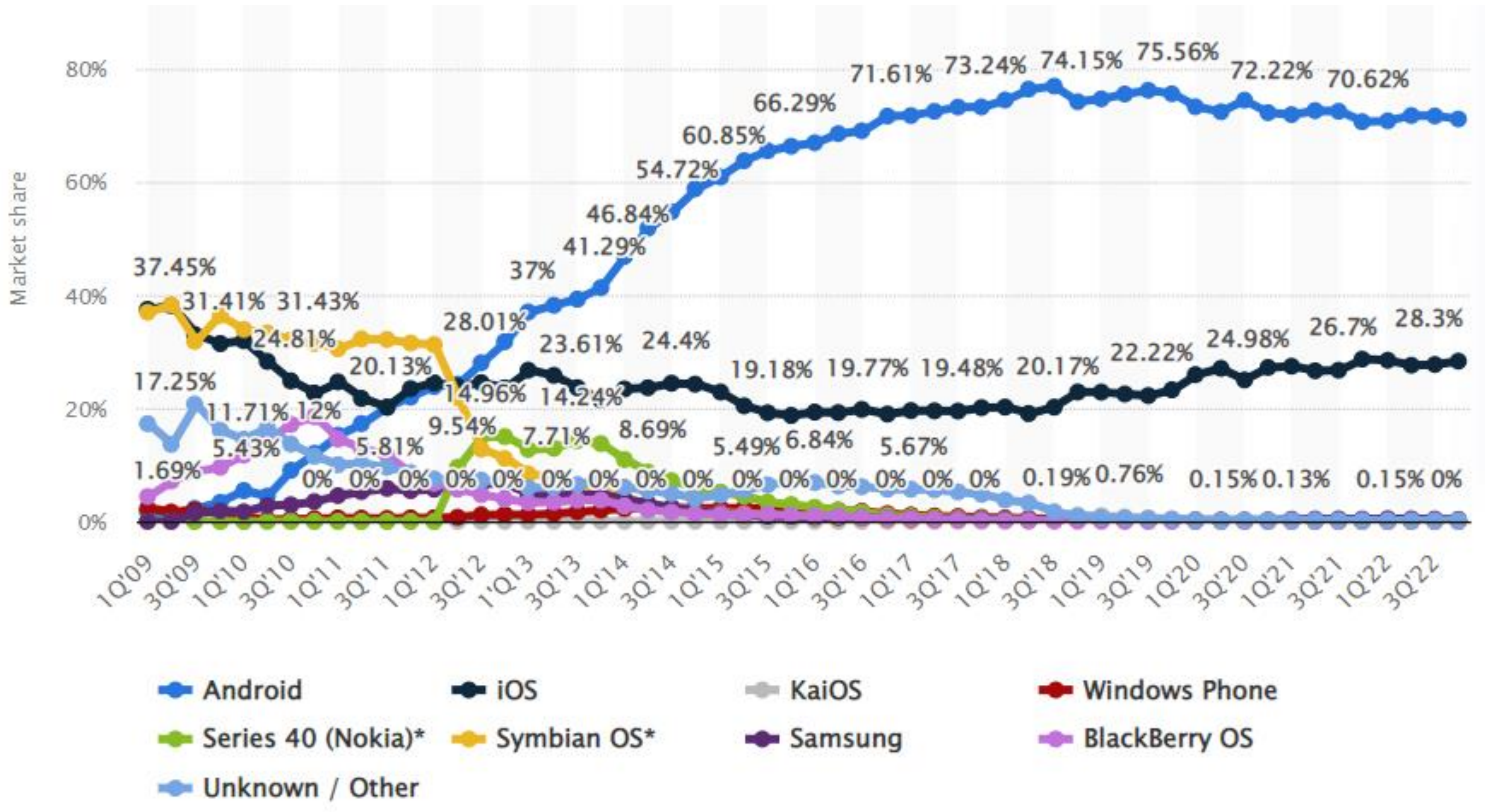
What is Android?

- Android is world’s leading mobile operating system
  - Open source (https://source.android.com/docs/setup)
  - Can download Android source code, explore, compile and customize it
  - 12 millions lines of code!!

- **Google:**
  - Owns Android, maintains it, extends it
  - Distributes Android OS, developer tools, free to use
  - Runs Android app market (https://play.google.com/store/apps)

SmartPhone OS

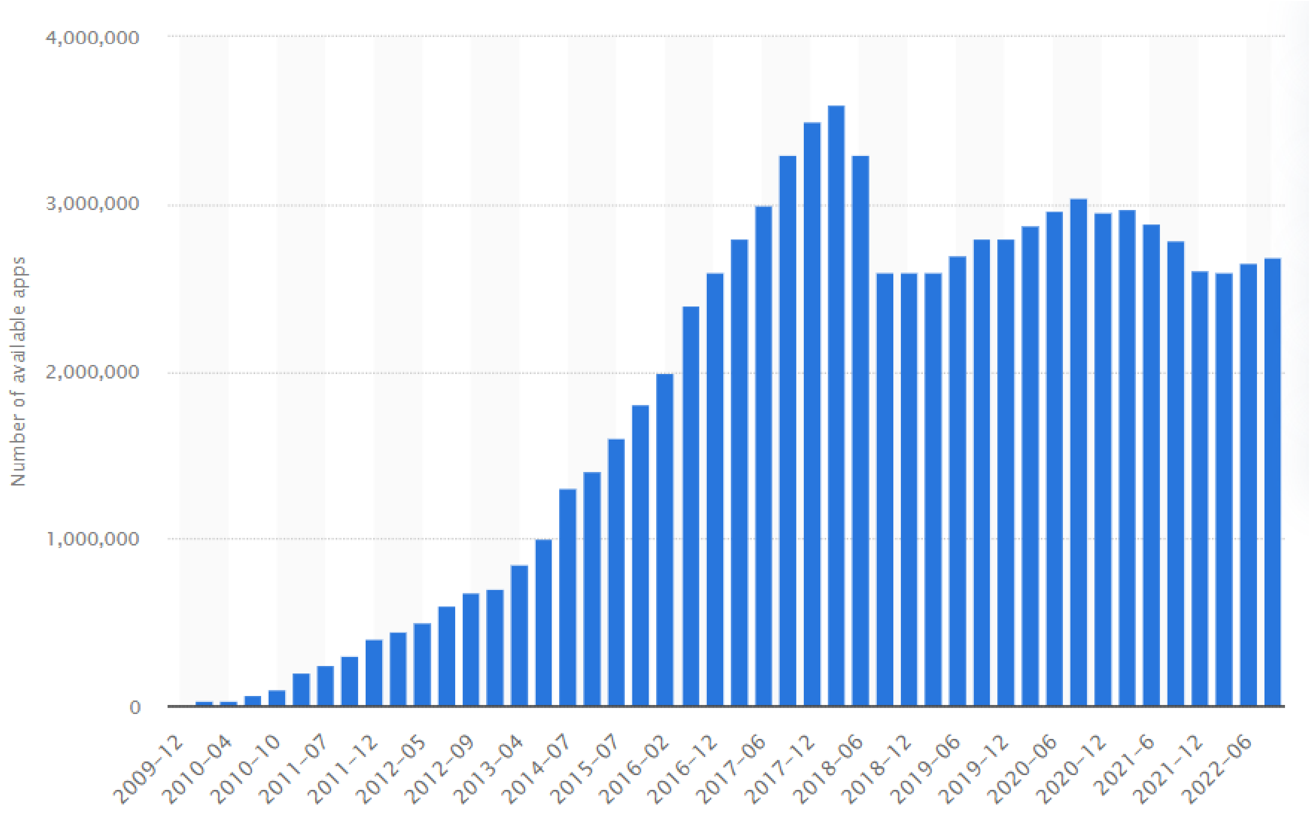
- Over 86% of all phones are smartphones
- Android OS on ~71% of phones worldwide



Source: Statista.com

Android Growth

- 3 billion active Android devices, May 2022 (ref: https://9to5google.com/)
- 2.7 million apps on Google Play Android app market (ref: statista.com)
  - Games, organizers, banking, entertainment, etc



Source: Statista.com

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!

Devices/Things
(e.g. Raspberry Pi)

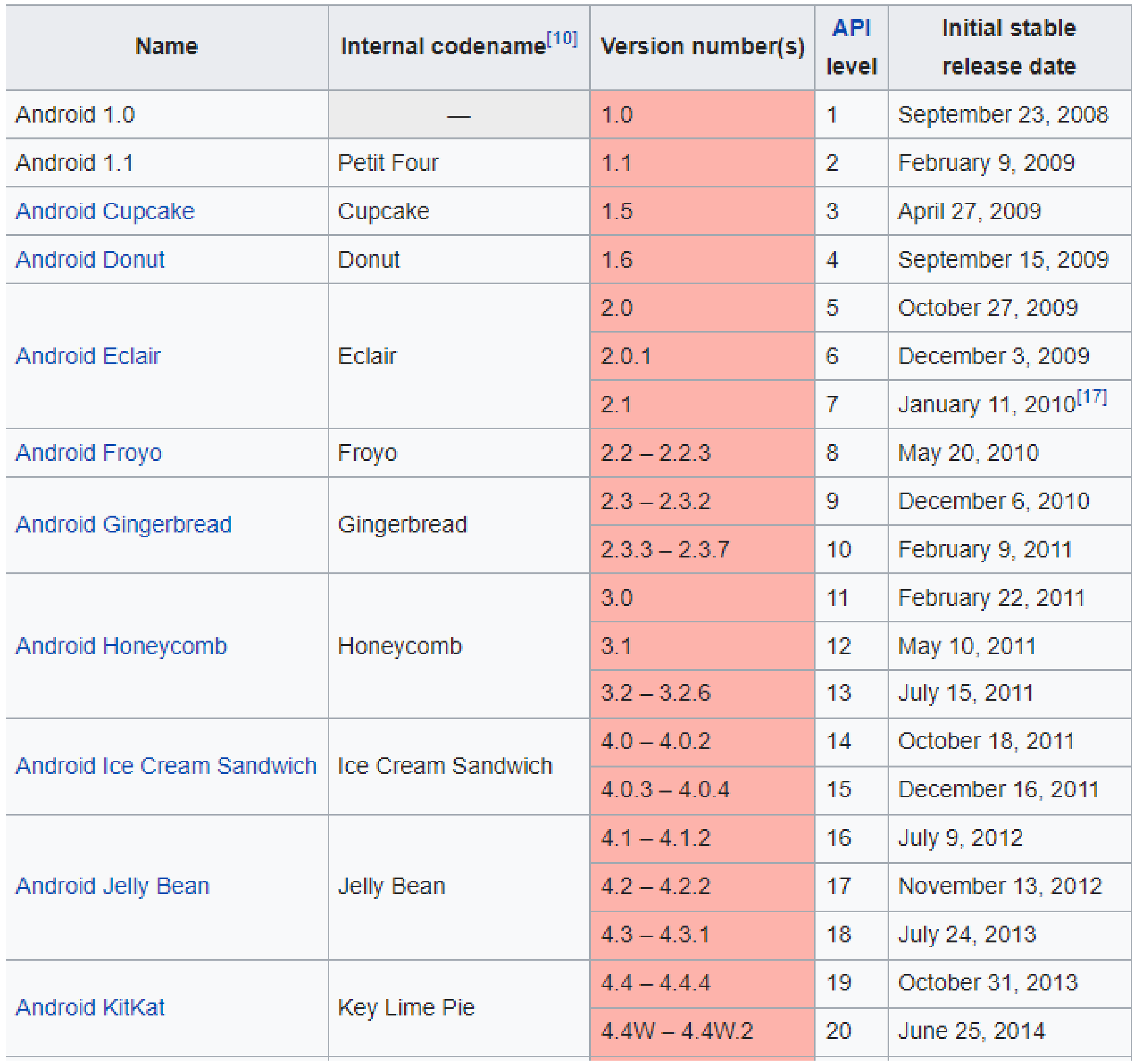
Television

Why Android? Already has many Mobile Computing and Ubicomp Modules

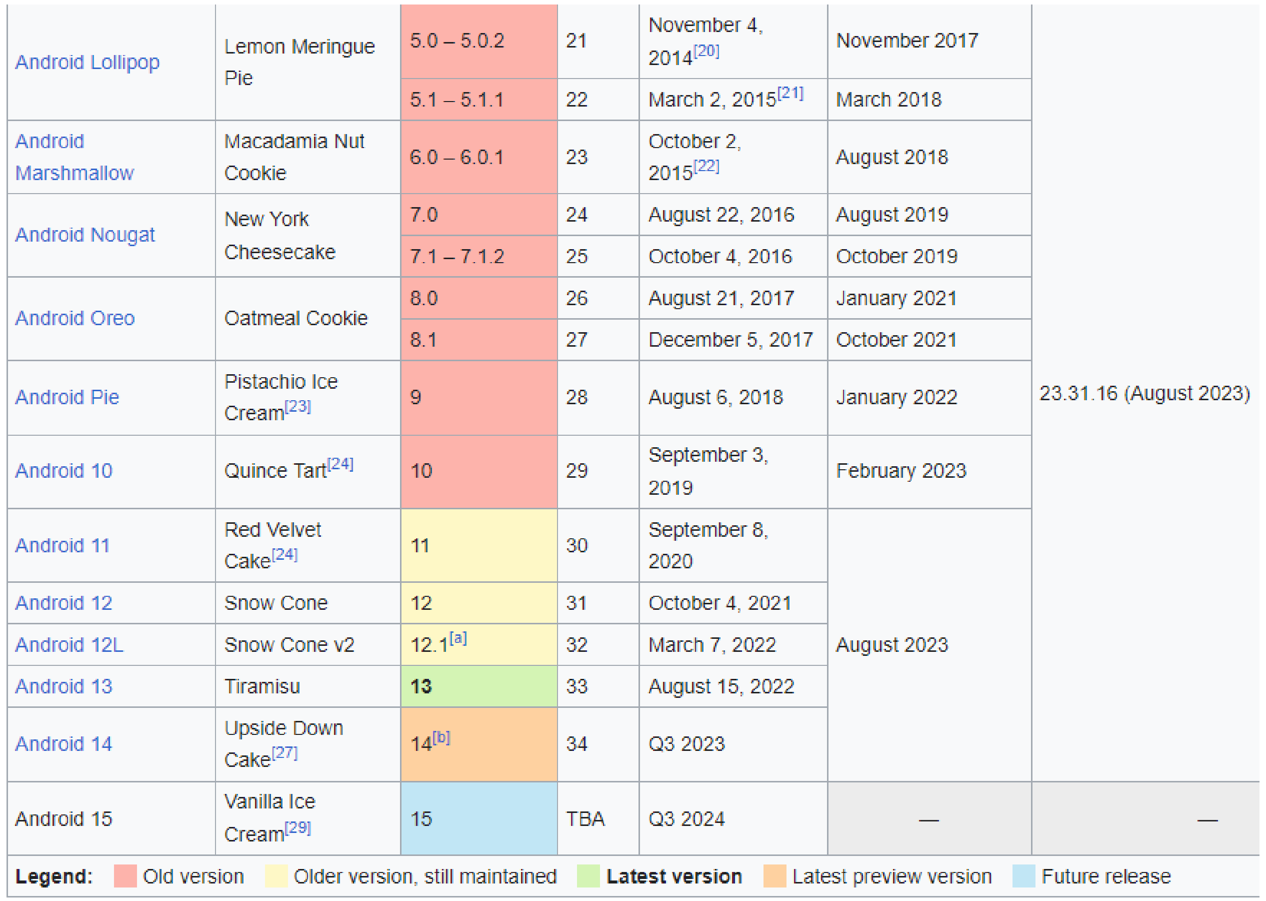
- Android’s Mobile programmable modules
  - Audio/video playback, taking pictures, database, location detection, maps

- Android’s 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 Version History



Name	Internal codename[10]	Version number(s)	API level	Initial stable release date
Android 1.0	—	1.0	1	September 23, 2008
Android 1.1	Petit Four	1.1	2	February 9, 2009
Android Cupcake	Cupcake	1.5	3	April 27, 2009
Android Donut	Donut	1.6	4	September 15, 2009
Android Eclair	Eclair	2.0	5	October 27, 2009
		2.0.1	6	December 3, 2009
		2.1	7	January 11, 2010[17]
Android Froyo	Froyo	2.2 – 2.2.3	8	May 20, 2010
Android Gingerbread	Gingerbread	2.3 – 2.3.2	9	December 6, 2010
		2.3.3 – 2.3.7	10	February 9, 2011
Android Honeycomb	Honeycomb	3.0	11	February 22, 2011
		3.1	12	May 10, 2011
		3.2 – 3.2.6	13	July 15, 2011
Android Ice Cream Sandwich	Ice Cream Sandwich	4.0 – 4.0.2	14	October 18, 2011
		4.0.3 – 4.0.4	15	December 16, 2011
Android Jelly Bean	Jelly Bean	4.1 – 4.1.2	16	July 9, 2012
		4.2 – 4.2.2	17	November 13, 2012
		4.3 – 4.3.1	18	July 24, 2013
Android KitKat	Key Lime Pie	4.4 – 4.4.4	19	October 31, 2013
		4.4W – 4.4W.2	20	June 25, 2014



Android Lollipop	Lemon Meringue Pie	5.0 – 5.0.2	21	November 4, 2014[20]	November 2017	23.31.16 (August 2023)
		5.1 – 5.1.1	22	March 2, 2015[21]	March 2018	
Android Marshmallow	Macadamia Nut Cookie	6.0 – 6.0.1	23	October 2, 2015[22]	August 2018	
Android Nougat	New York Cheesecake	7.0	24	August 22, 2016	August 2019	
		7.1 – 7.1.2	25	October 4, 2016	October 2019	
Android Oreo	Oatmeal Cookie	8.0	26	August 21, 2017	January 2021	
		8.1	27	December 5, 2017	October 2021	
Android Pie	Pistachio Ice Cream[23]	9	28	August 6, 2018	January 2022	
Android 10	Quince Tart[24]	10	29	September 3, 2019	February 2023	
Android 11	Red Velvet Cake[24]	11	30	September 8, 2020	August 2023	
Android 12	Snow Cone	12	31	October 4, 2021		
Android 12L	Snow Cone v2	12.1[a]	32	March 7, 2022		
Android 13	Tiramisu	**13**	33	August 15, 2022		
Android 14	Upside Down Cake[27]	14[b]	34	Q3 2023		
Android 15	Vanilla Ice Cream[29]	15	TBA	Q3 2024	—	—

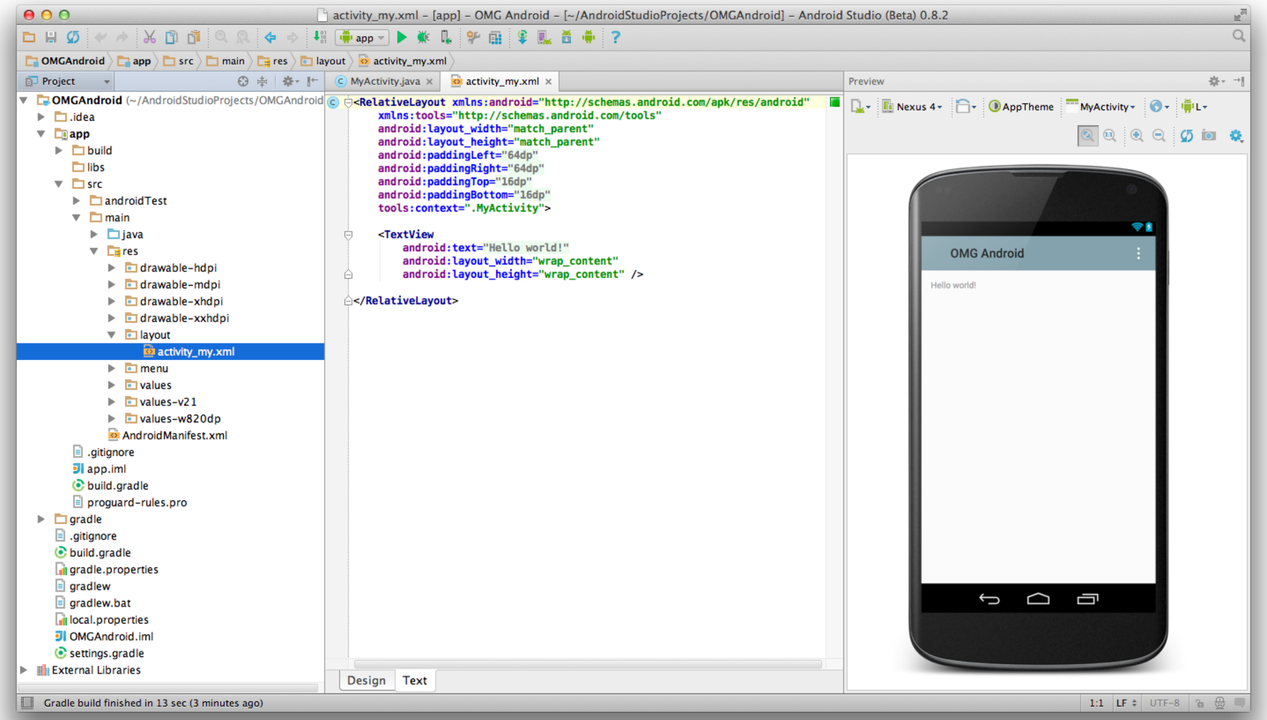
Legend: 🟥 Old version 🟨 Older version, still maintained 🟩 **Latest version** 🟧 Latest preview version 🟦 Future release

Source: https://en.wikipedia.org/wiki/Android_version_history

Android Developer Environment

Android IDE: Android Studio

- Developed by Google, announced in May 2013
- Replaced Eclipse IDE
- IDE specifically for just Android development, cleaner interface with drag and drop app design



activity_my.xml - [app] - OMG Android - [~/AndroidStudioProjects/OMGAndroid] - Android Studio (Beta) 0.8.2

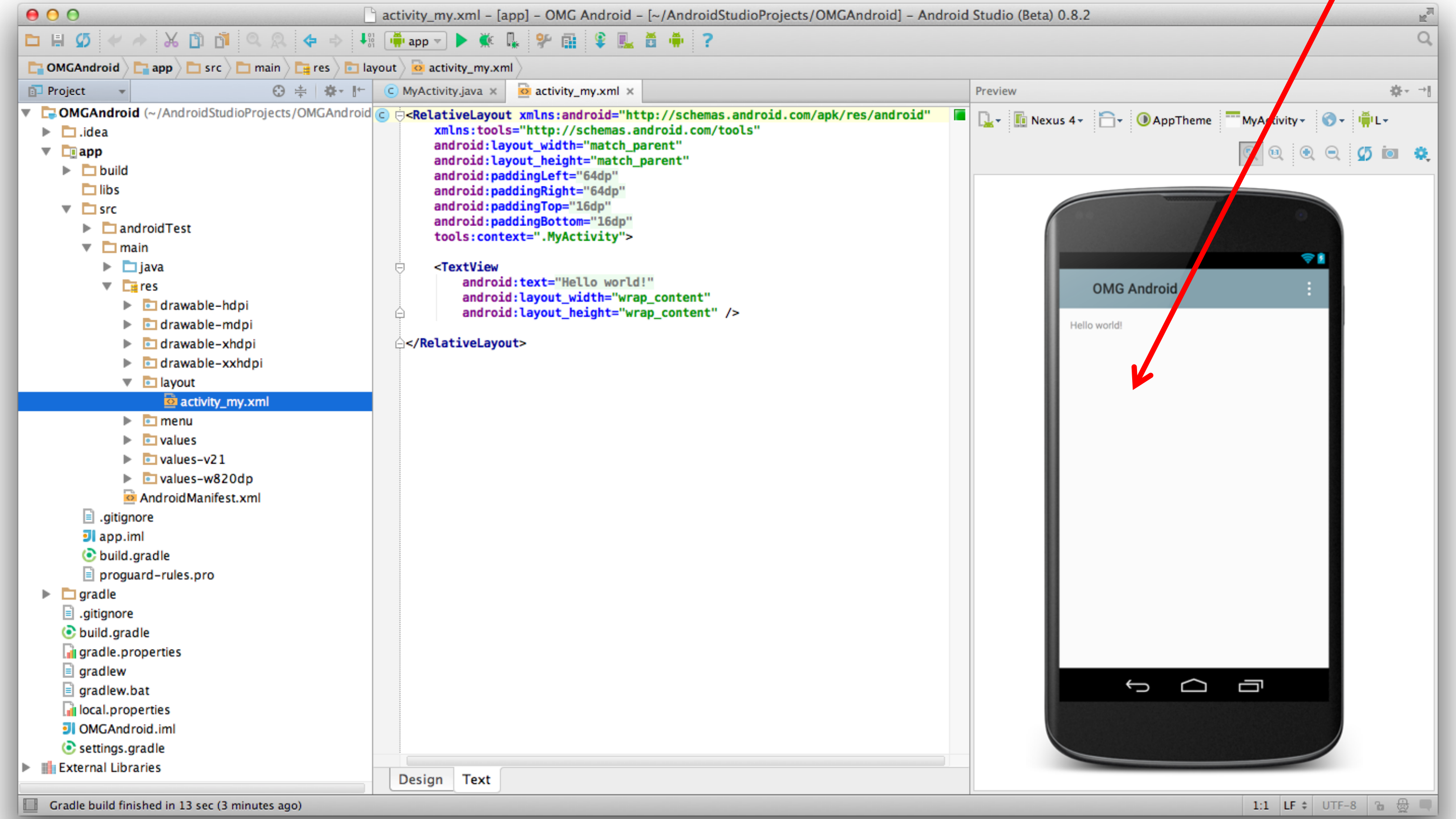
```xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="64dp"
    android:paddingRight="64dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp"
    tools:context=".MyActivity">

    <TextView
        android:text="Hello world!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

Gradle build finished in 13 sec (3 minutes ago)

Where to Run Android App

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

Emulated phone in Android Studio



```xml
<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"
    android:paddingLeft="64dp"
    android:paddingRight="64dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp"
    tools:context=".MyActivity">

    <TextView
        android:text="Hello world!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

Running Android App on Real Phone

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

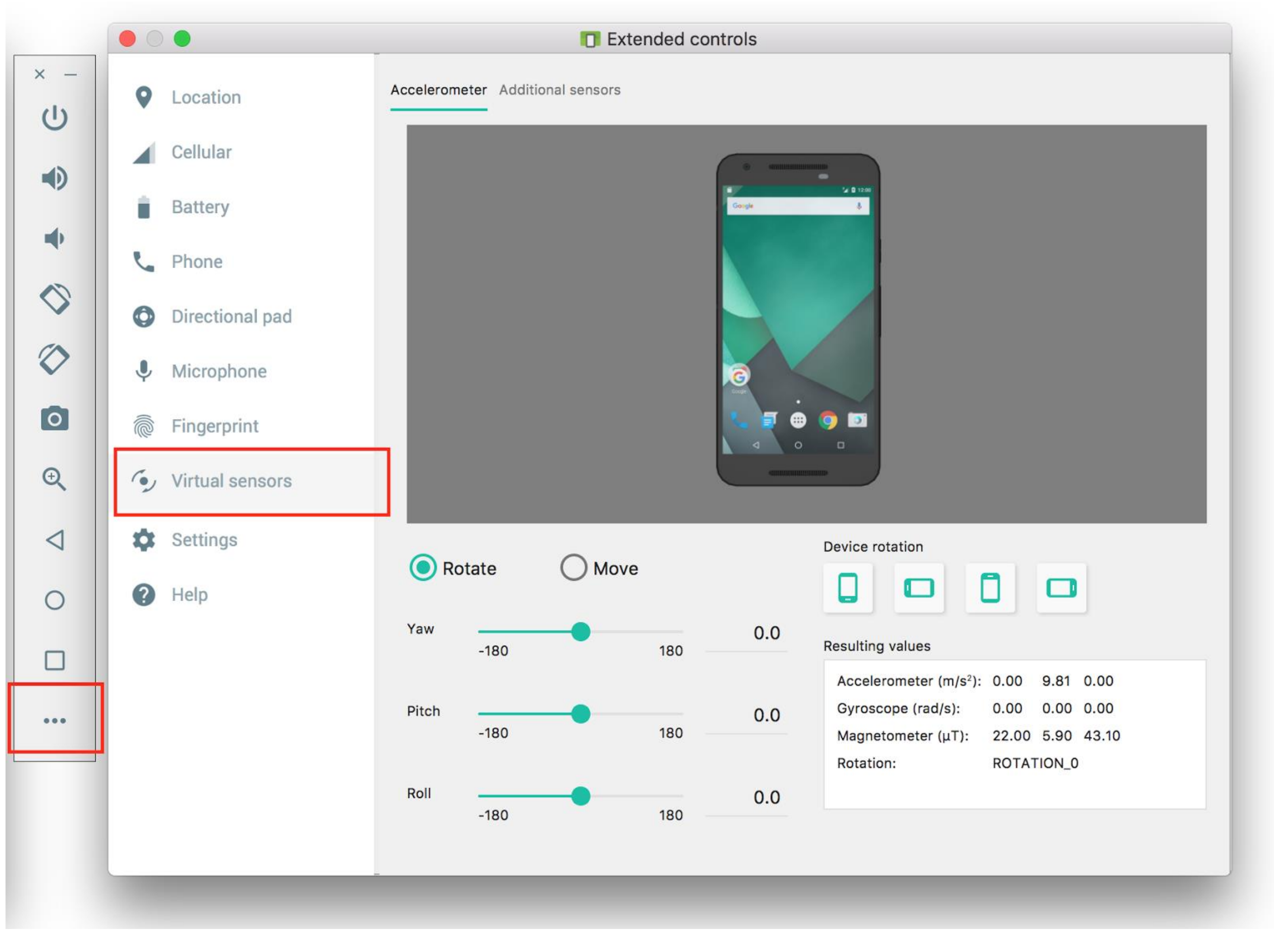
Emulator Pros and Cons (Vs Real Phone)

- Pros:
  - Conveniently test app on basic hardware by clicking in software
  - Easy to test app on many emulated devices (phones, tablets, TVs, etc), various screen sizes

- Cons:
  - Limited support, access to hardware, communications, sensors
  - E.g. GPS, camera, video recording, making/receiving phone calls, Bluetooth devices, USB devices, battery level, sensors, etc
  - Slower than real phone

Android Studio Supports Some Sensors

- Emulates **some** sensors (e.g. location, accelerometer), but still limited



Yaw -180 180 0.0
Pitch -180 180 0.0
Roll -180 180 0.0

```
Accelerometer (m/s²):  0.00   9.81   0.00
Gyroscope (rad/s):     0.00   0.00   0.00
Magnetometer (µT):     22.00  5.90   43.10
Rotation:              ROTATION_0
```

Demo: Android Studio

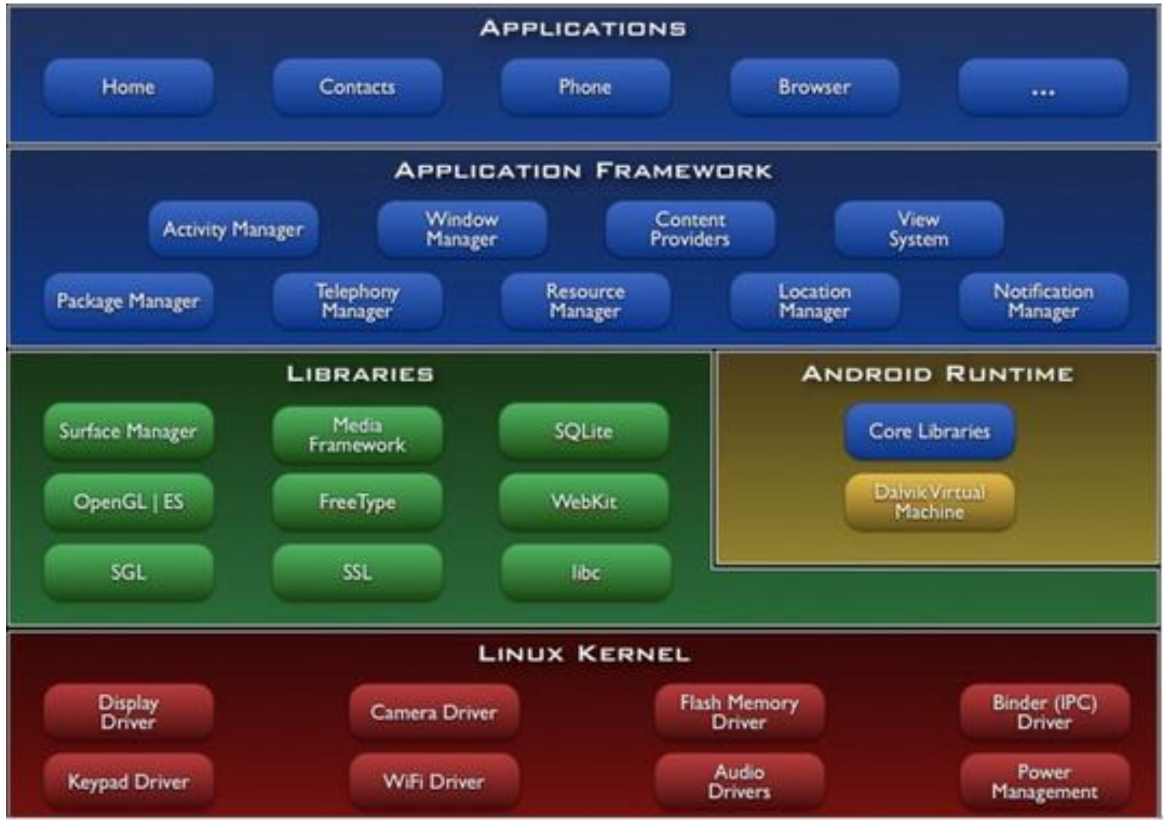
Android Software Framework

Android Functionality as Apps

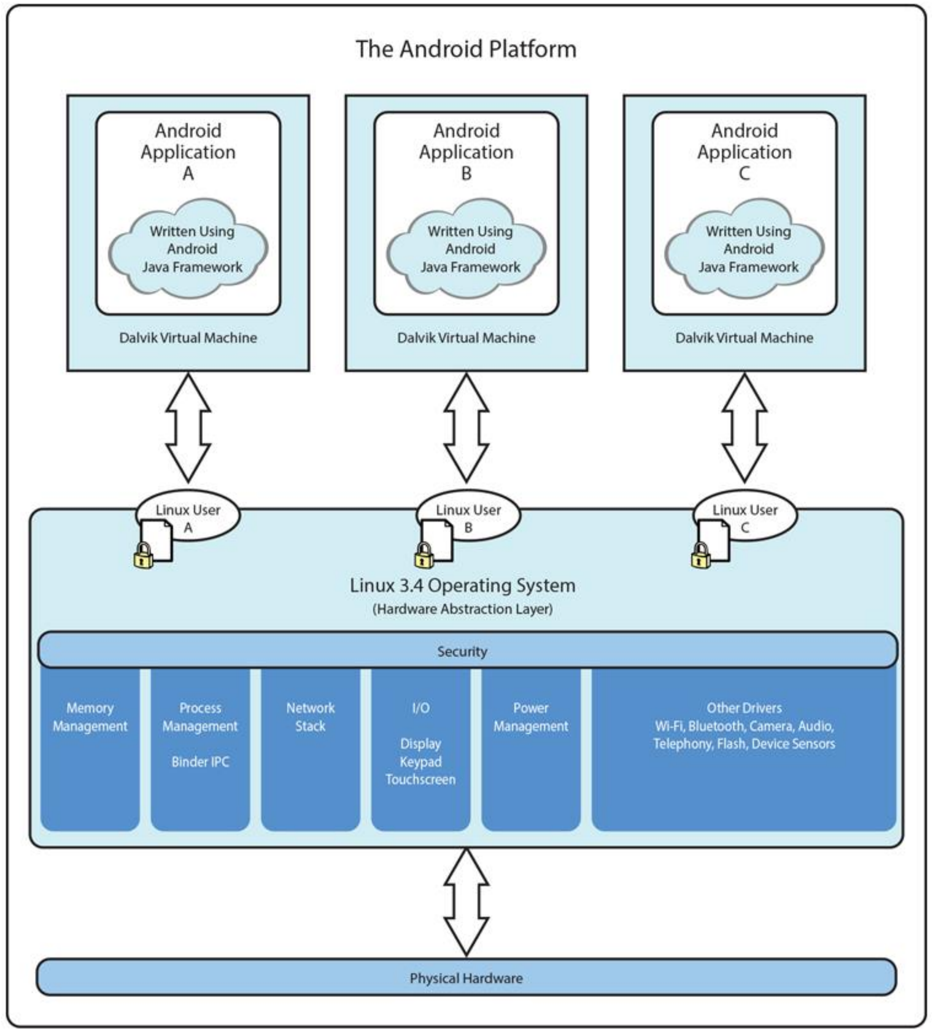
- Android functionality: collection of mini-applications (apps)
- Even phone “hardware” components dialer, keyboard, etc

Android Software Framework

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



Android Software Framework



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

- 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 processes with the app's user ID can access its files

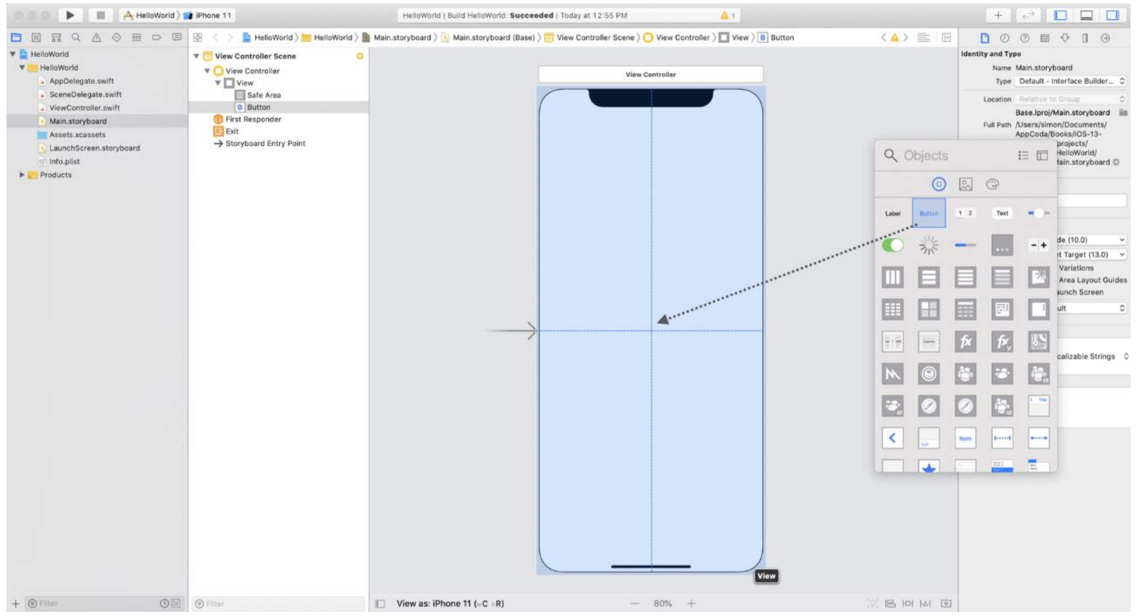
Android Programming Languages

- Two main languages to program Android
  1. **Kotlin:**
     - Newer, has become THE way to program Android. Higher level, easier?
     - Google encourages programmers to use kotlin instead of Java
  2. **Java-based (Native) programming + XML:**
     - Older way? Being used less than Kotlin?

Other Mobile Development Frameworks

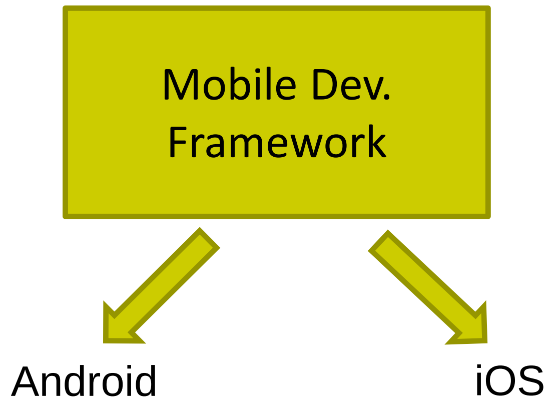
iOS App Development

- Download Xcode (iOS programming IDE)
- Free to program iOS apps
- But to publish app on app store, need to buy $99/yr membership
  - More regulated than Android
  - A human checks all iOS apps before publishing them
- iOS apps programmed in Swift language



Other Mobile Development Frameworks

- Lots of cross-platform frameworks
- Idea: write code in “some” language that gets compiled to Android or iOS



Xamarin: .NET Microsoft framework, code in C#

PhoneGap: Program mobile code in HTML, CSS

Other Mobile Development Frameworks

- Some framework just for UI development

Flutter

Ionic

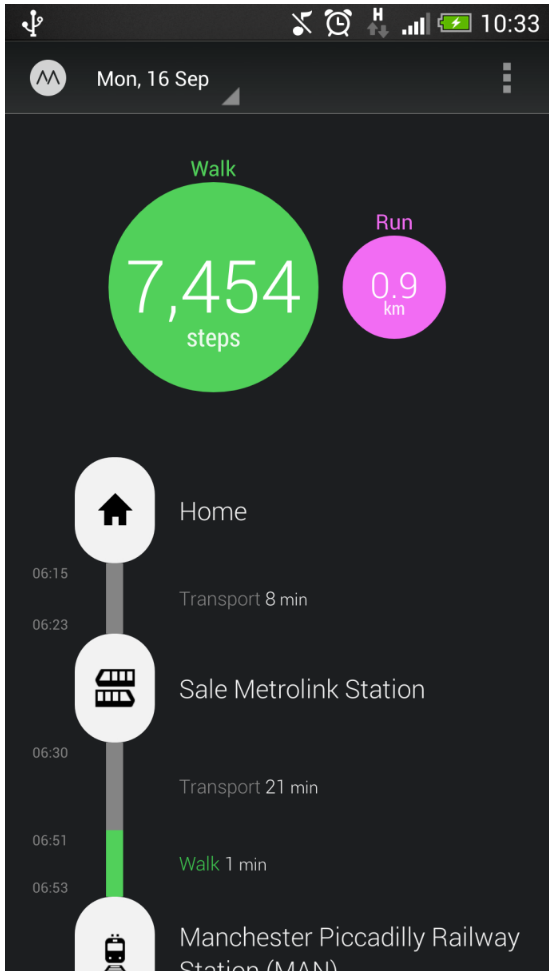
React

What other Mobile Dev frameworks do you know? Like?

Android Apps: Big Picture

UI Design using XML

- UI design code (XML) separate from the program (kotlin/Java)
- Why? Can modify UI without changing Kotlin/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**

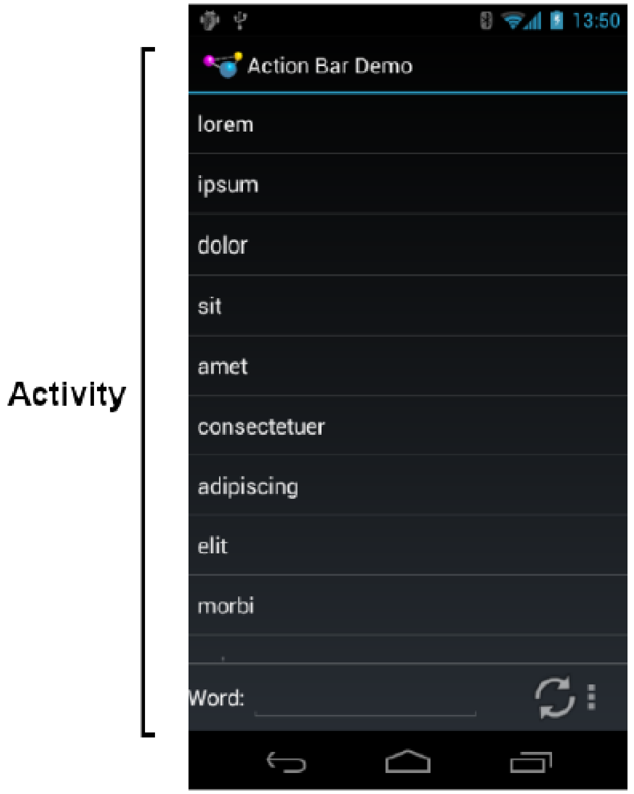
- Installation = installing **apk file**

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 (screens)

- 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

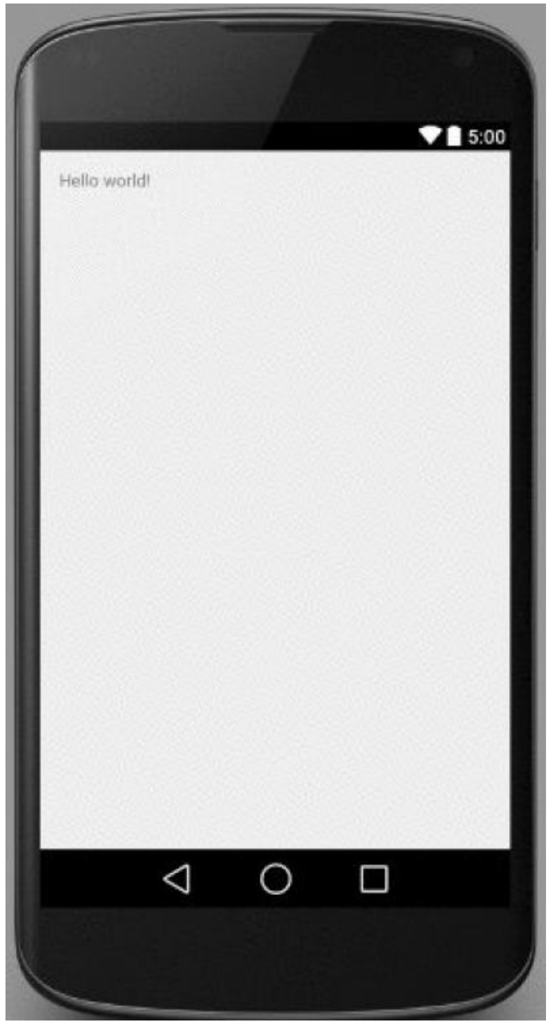
3 Files in “Hello World” Android Project

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

- **MainActivity.Java or MainActivity.kt:** Java or Kotlin 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 auto-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

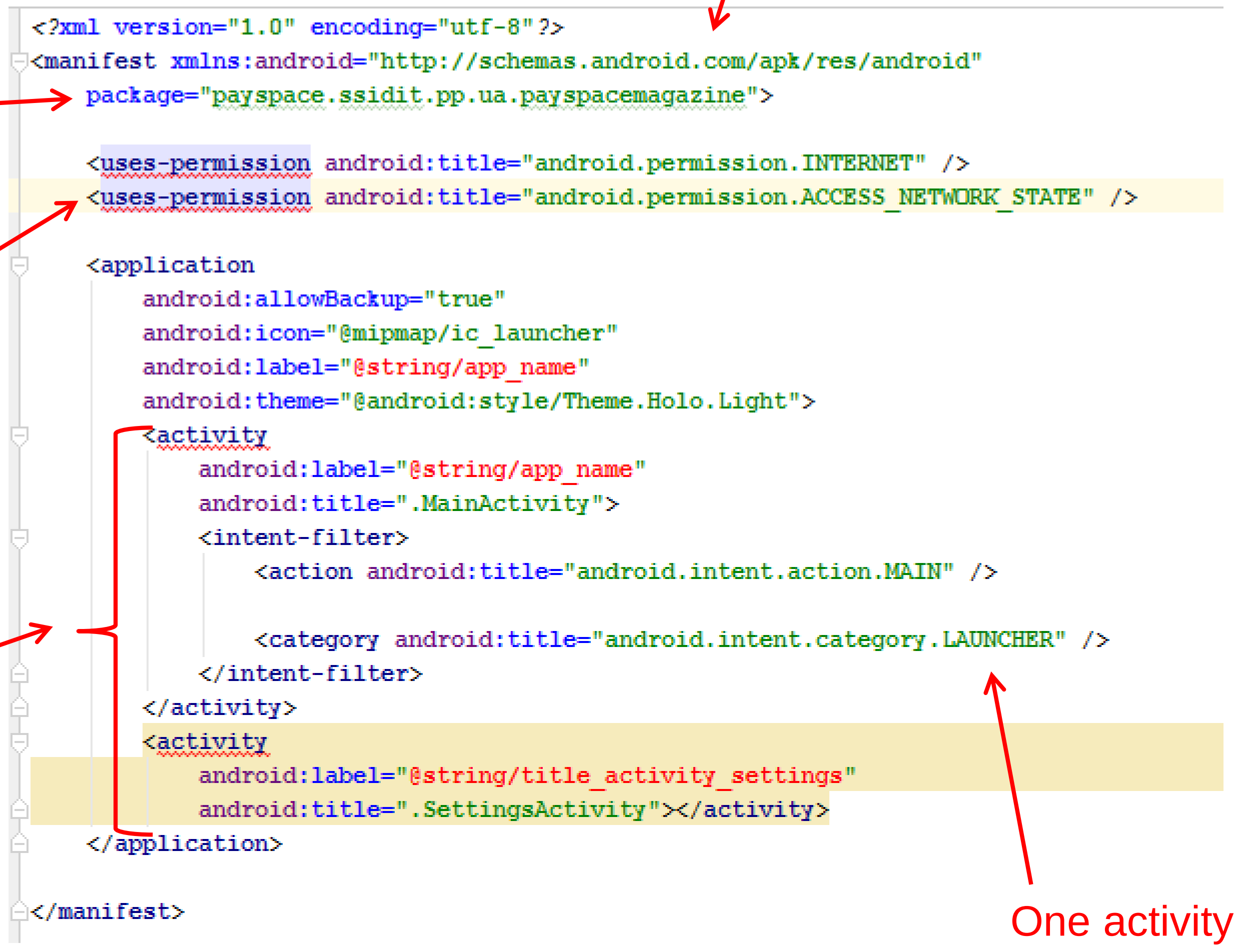
Create/execute activities
(declared in java or kotlin files)
E.g. MainActivity.Java or
MainActivity.kt

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

Hello world!

Inside "Hello World" AndroidManifest.xml

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



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="payspace.ssidit.pp.ua.payspacemagazine">

    <uses-permission android:title="android.permission.INTERNET" />
    <uses-permission android:title="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@android:style/Theme.Holo.Light">
        <activity
            android:label="@string/app_name"
            android:title=".MainActivity">
            <intent-filter>
                <action android:title="android.intent.action.MAIN" />

                <category android:title="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:label="@string/title_activity_settings"
            android:title=".SettingsActivity"></activity>
    </application>

</manifest>
```

Your package name

Permissions requested

List of activities (screens) in your app

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

Execution Order

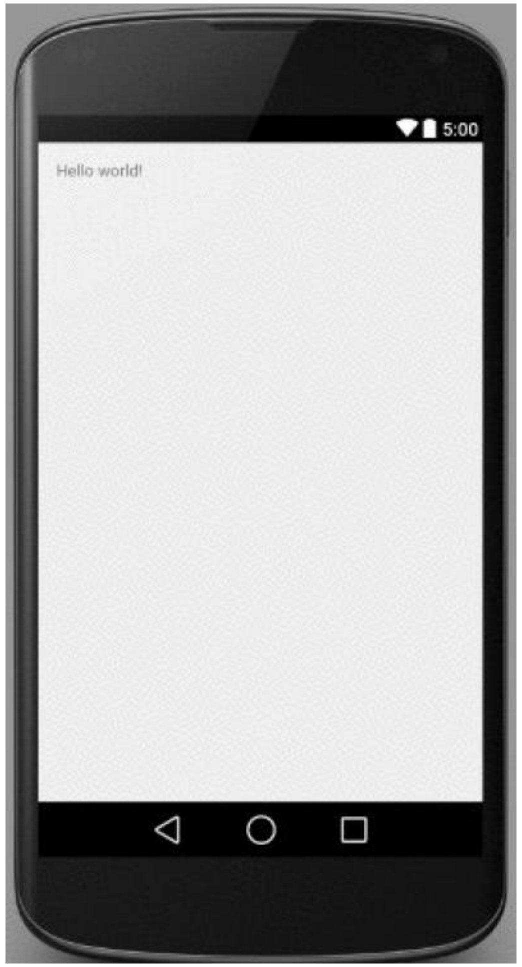
Next: Samples of AndroidManifest.xml
Hello World program

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

Next

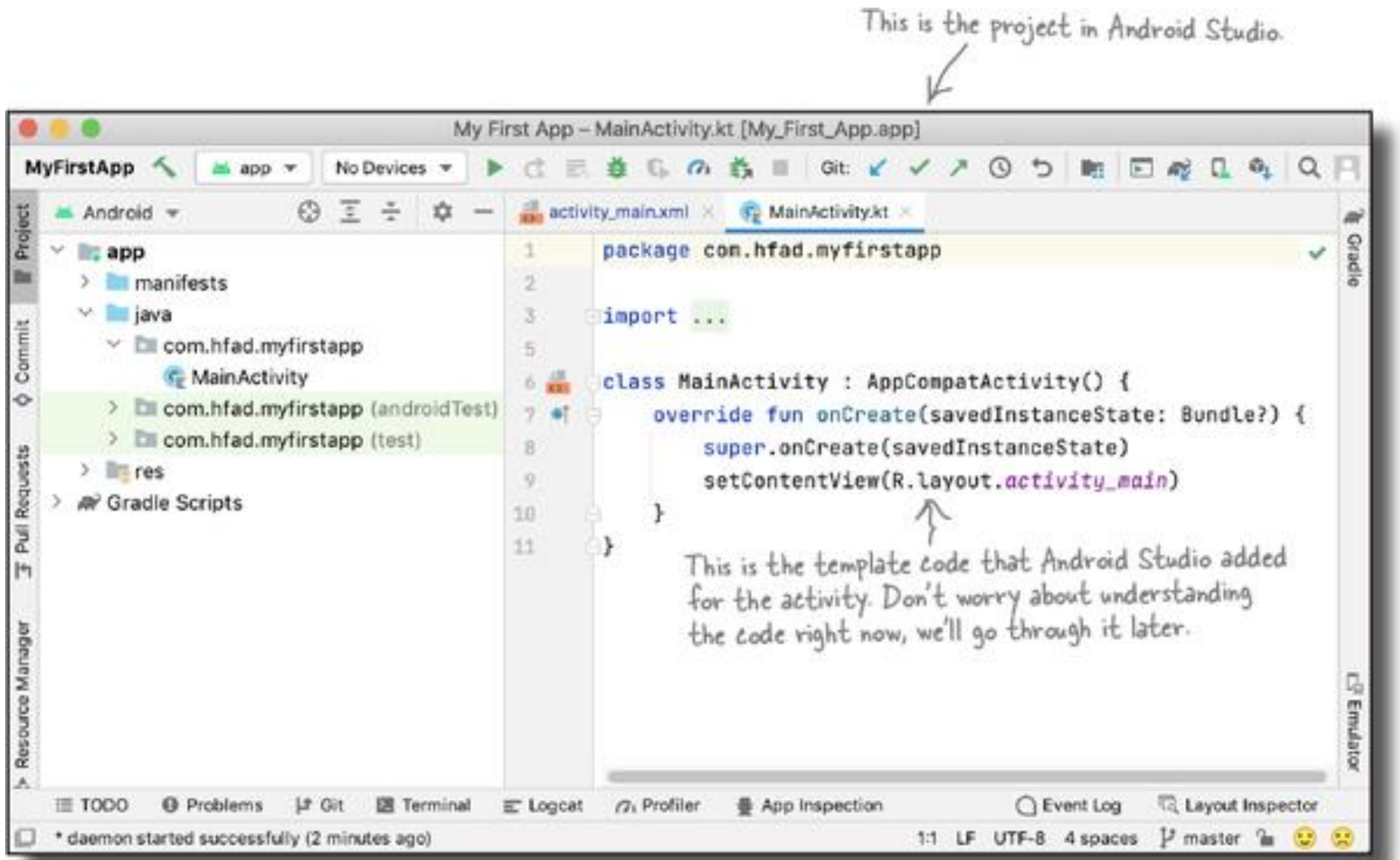
Create/execute activities
(declared in java or kotlin files)
E.g. MainActivity.Java or
MainActivity.kt

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



Example MainActivity.kt

- This sample was auto-generated by Android Studio
- Our app's MainActivity is derived from the AppCompatActivity class
- More later..


This is the project in Android Studio.
My First App – MainActivity.kt [My_First_App.app]
MyFirstApp
app
No Devices
Git:
Android
app
manifests
java
com.hfad.myfirstapp
MainActivity
com.hfad.myfirstapp (androidTest)
com.hfad.myfirstapp (test)
res
Gradle Scripts
activity_main.xml
MainActivity.kt
This is the template code that Android Studio added for the activity. Don't worry about understanding the code right now, we'll go through it later.
TODO
Problems
Git
Terminal
Logcat
Profiler
App Inspection
Event Log
Layout Inspector
daemon started successfully (2 minutes ago)
1:1 LF UTF-8 4 spaces master

```
package com.hfad.myfirstapp

import ...

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

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

Package declaration

```java
package com.commonsware.empublite;
```

Import needed classes

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

My class inherits from Android activity class

Initialize by calling onCreate( ) method of base Activity class

```java
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

Next: Samples of AndroidManifest.xml
Hello World program

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

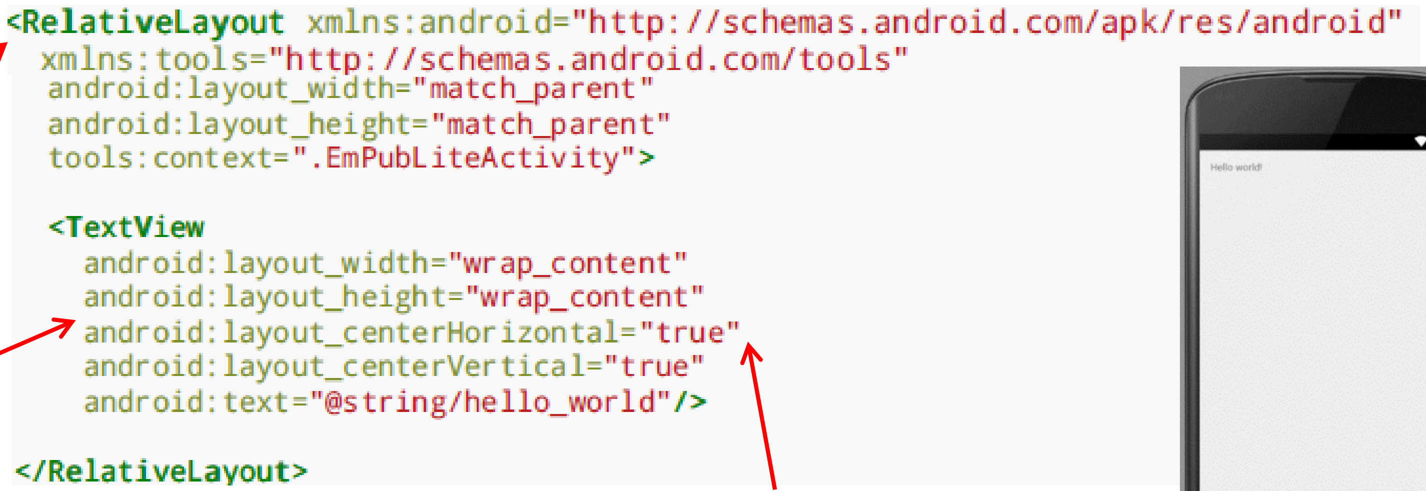
Create/execute activities (declared in java or kotlin files) E.g. MainActivity.Java or MainActivity.kt

Next

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

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**

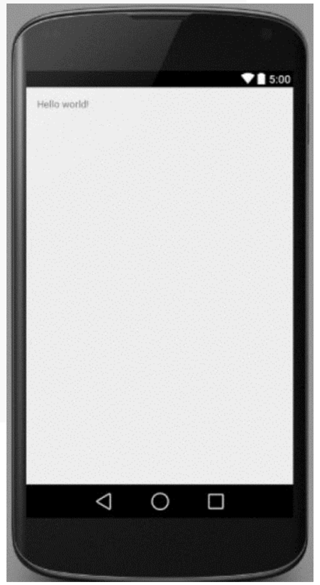


```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
 xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".EmPubLiteActivity">

  <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

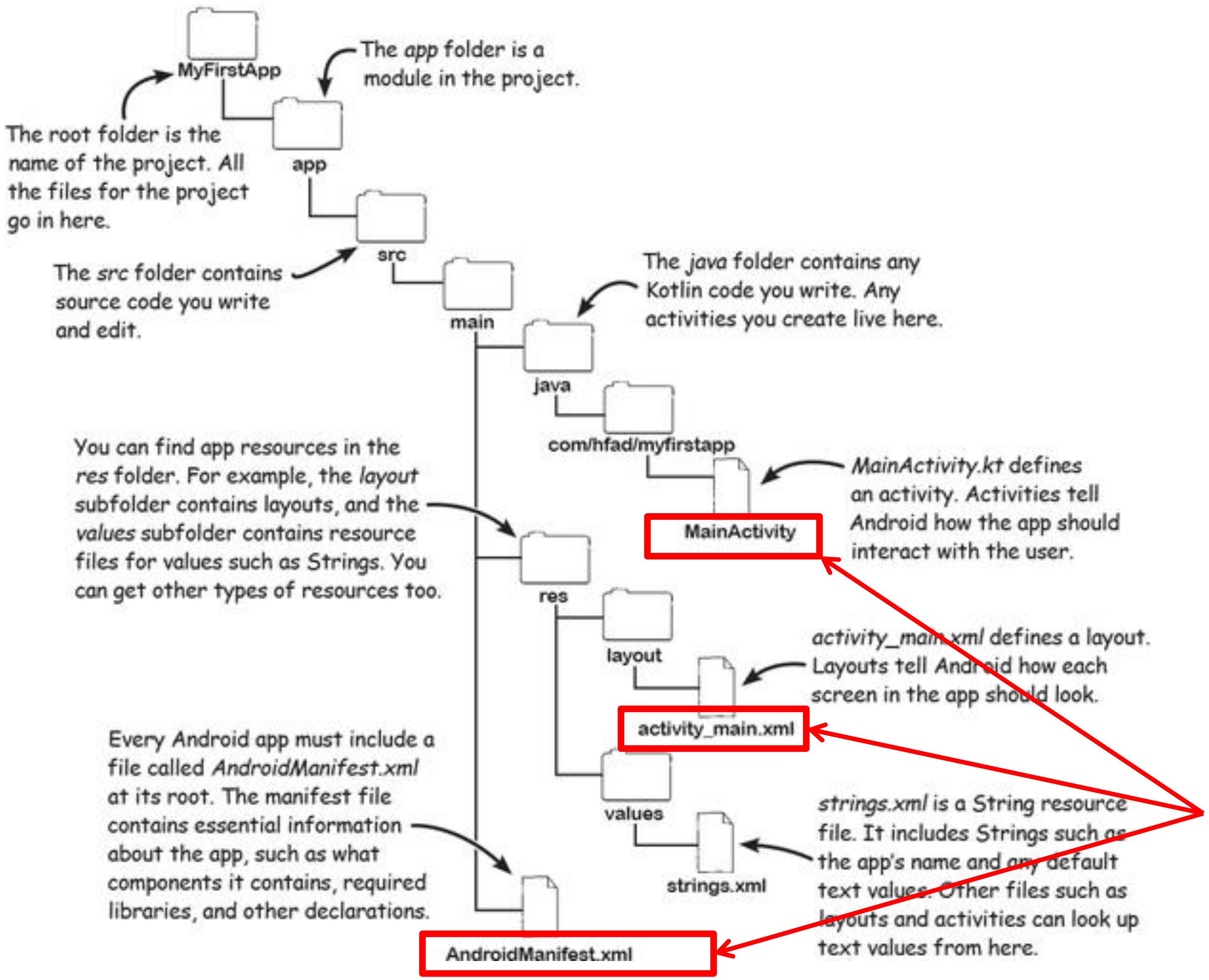
Add widgets

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



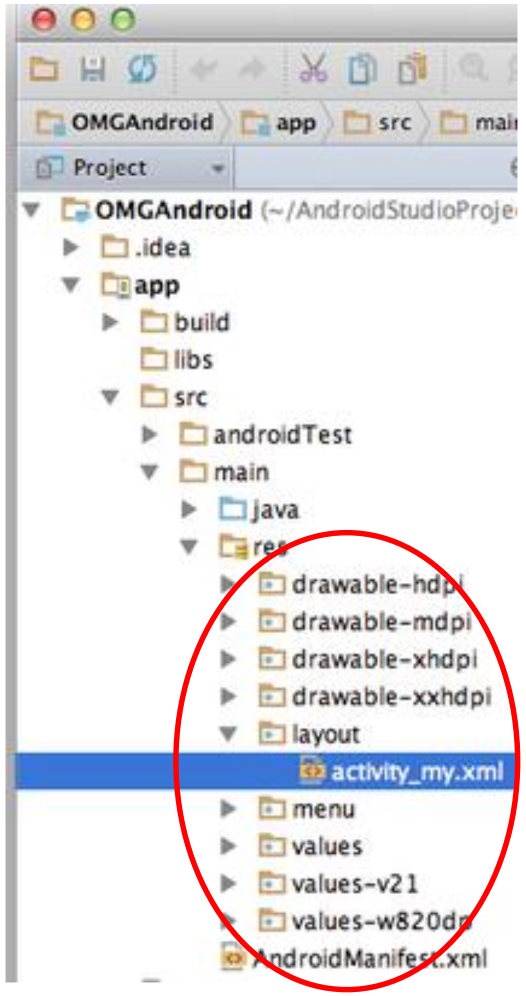
Android Files

Android Studio Project File Structure


MyFirstApp
The app folder is a module in the project.
The root folder is the name of the project. All the files for the project go in here.
app
src
The src folder contains source code you write and edit.
main
The java folder contains any Kotlin code you write. Any activities you create live here.
java
com/hfad/myfirstapp
MainActivity.kt defines an activity. Activities tell Android how the app should interact with the user.
MainActivity
You can find app resources in the res folder. For example, the layout subfolder contains layouts, and the values subfolder contains resource files for values such as Strings. You can get other types of resources too.
res
layout
activity_main.xml defines a layout. Layouts tell Android how each screen in the app should look.
activity_main.xml
values
Every Android app must include a file called AndroidManifest.xml at its root. The manifest file contains essential information about the app, such as what components it contains, required libraries, and other declarations.
strings.xml is a String resource file. It includes Strings such as the app's name and any default text values. Other files such as layouts and activities can look up text values from here.
strings.xml
AndroidManifest.xml
3 Main Files to Write Android app

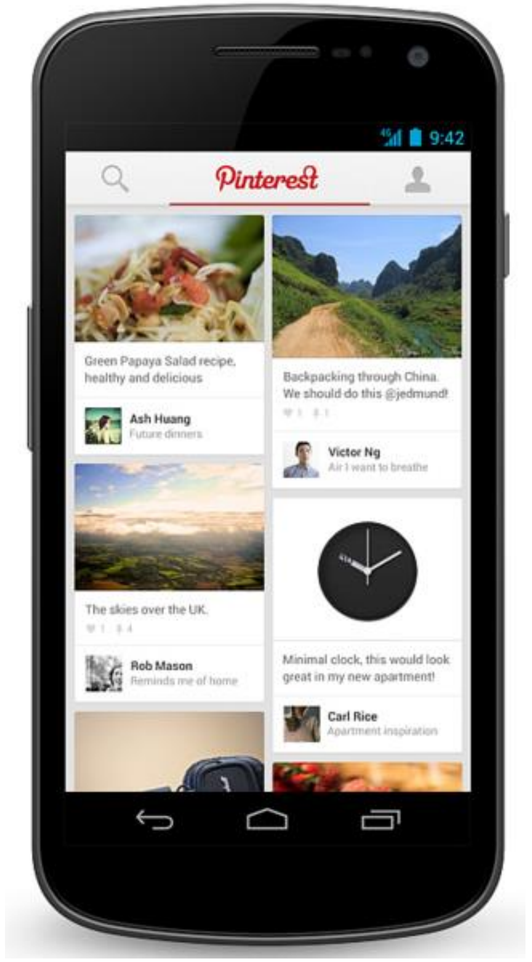
Files in an Android Project

- **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 different 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

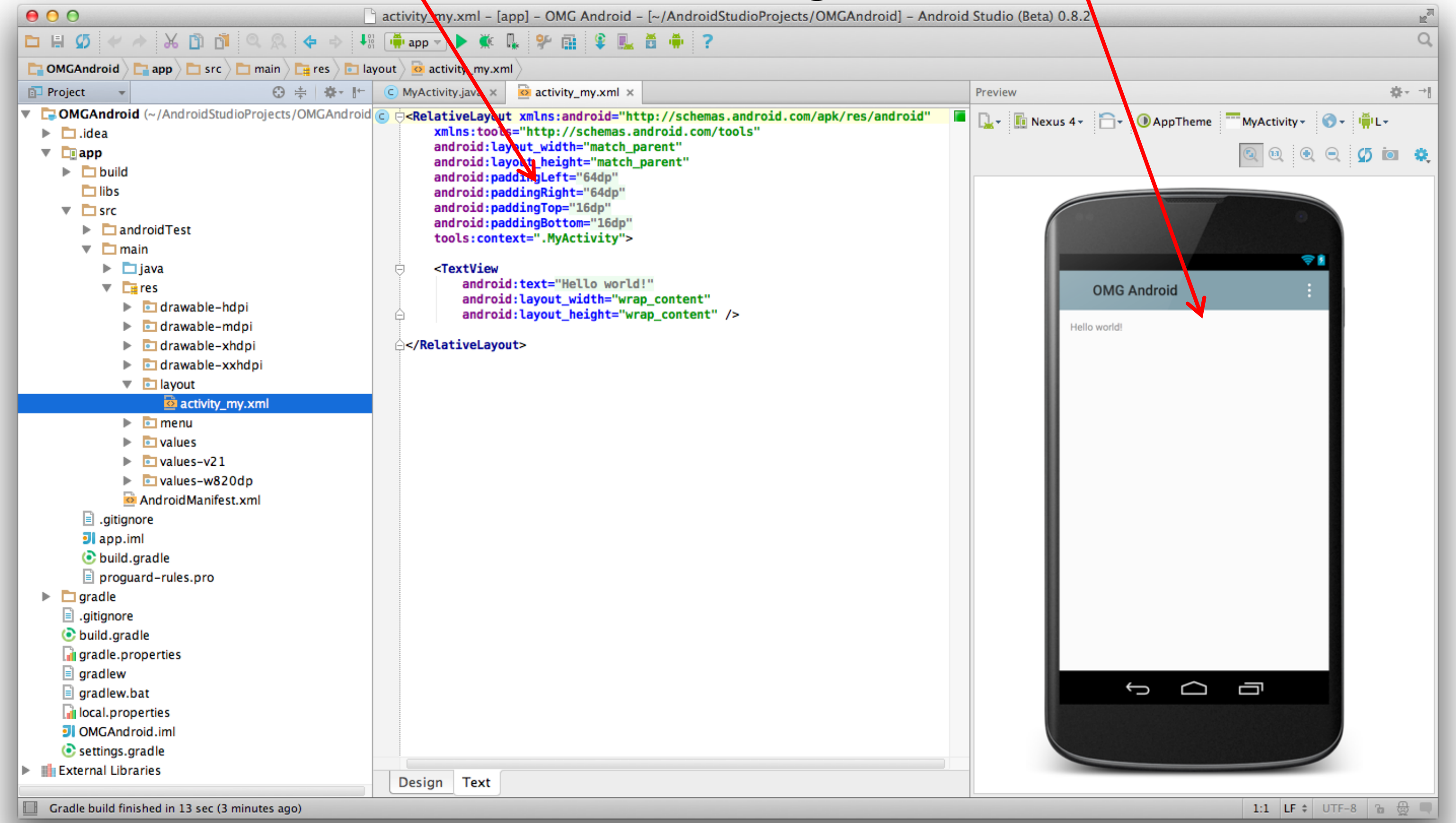
- **res/layout:** layout, dimensions (width, height) of screen cells are specified in XML file here
- **res/drawable-xyz/:** The images stored in jpg or other format here
- **java/:** App's response when user clicks on a selection is specified in java or kotlin file here
- **AndroidManifest.XML:** Contains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


9:42
Pinterest
Green Papaya Salad recipe, healthy and delicious
Ash Huang
Future dinners
Backpacking through China. We should do this @jedmund!
Victor Ng
Air I want to breathe
The skies over the UK.
Rob Mason
Reminds me of home
Minimal clock, this would look great in my new apartment!
Carl Rice
Apartment inspiration

Editting in Android Studio

Editting Android

- Can edit apps in:
  - **Text View:** edit XML directly, or
  - **Design View:** Drag and drop widgets unto emulated phone
  - **Split View:** Combines both Text View and Design View in one screen



activity_my.xml - [app] - OMG Android - [~/AndroidStudioProjects/OMGAndroid] - Android Studio (Beta) 0.8.2
OMGAndroid
app
src
main
res
layout
activity_my.xml
Project
MyActivity.java
Preview
Nexus 4
AppTheme
MyActivity
L
OMG Android
Hello world!
Design
Text
Gradle build finished in 13 sec (3 minutes ago)

```
OMGAndroid (~/AndroidStudioProjects/OMGAndroid)
  .idea
  app
    build
    libs
    src
      androidTest
      main
        java
        res
          drawable-hdpi
          drawable-mdpi
          drawable-xhdpi
          drawable-xxhdpi
          layout
            activity_my.xml
          menu
          values
          values-v21
          values-w820dp
        AndroidManifest.xml
    .gitignore
    app.iml
    build.gradle
    proguard-rules.pro
  gradle
  .gitignore
  build.gradle
  gradle.properties
  gradlew
  gradlew.bat
  local.properties
  OMGAndroid.iml
  settings.gradle
External Libraries
```

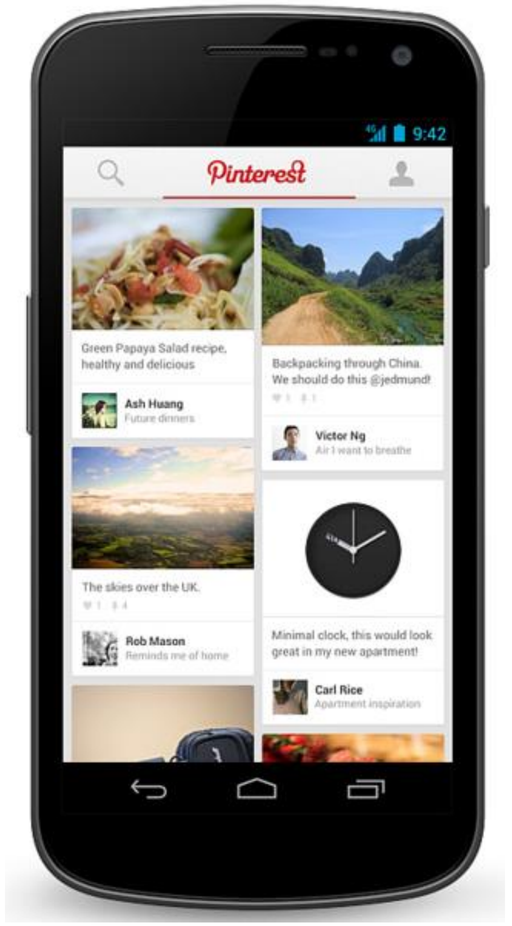
```xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="64dp"
    android:paddingRight="64dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp"
    tools:context=".MyActivity">

    <TextView
        android:text="Hello world!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

Android UI Design in XML

Recall: Files Hello World Android Project

XML file used to design Android UI

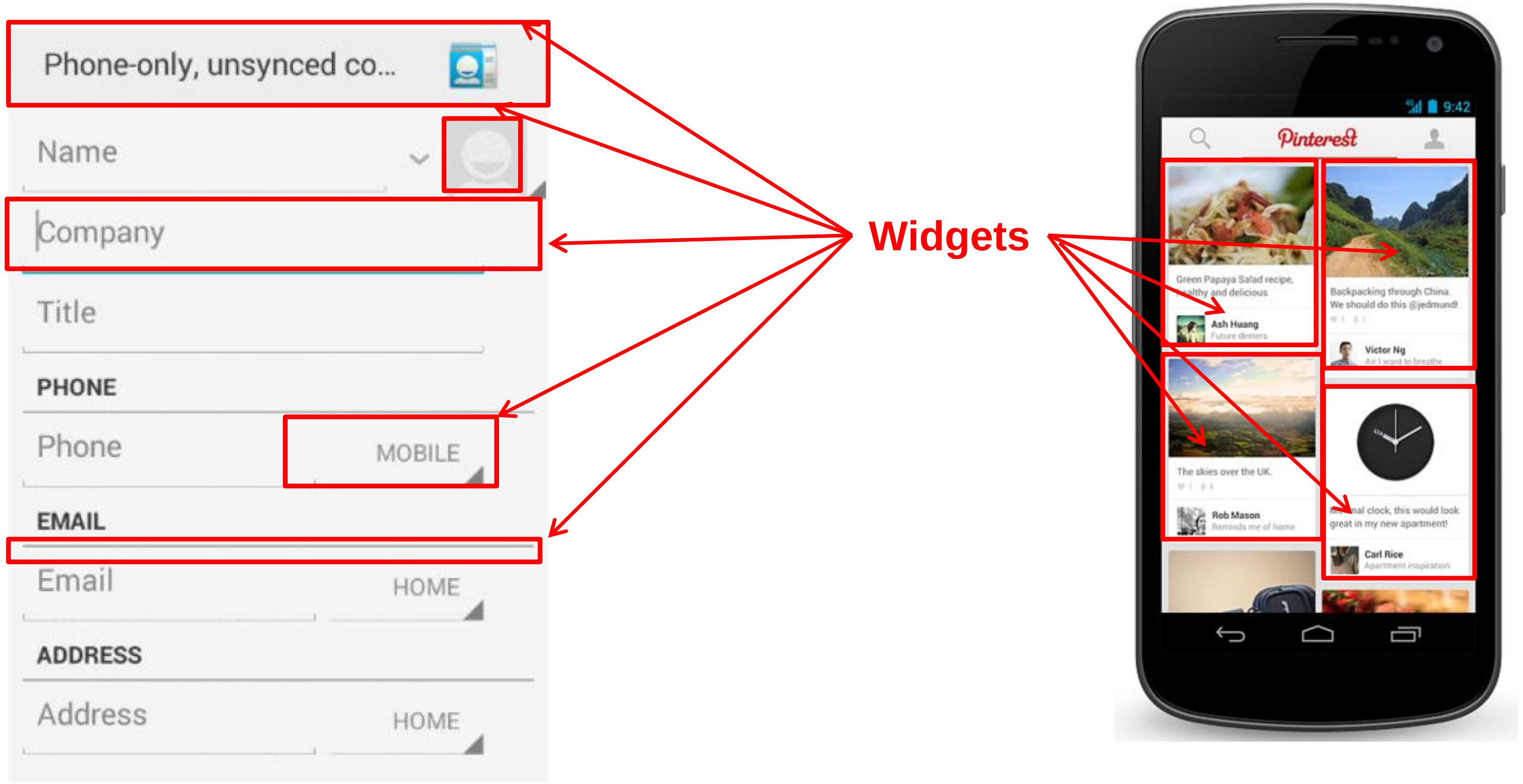
- 3 Files:
  - **Activity_main.xml:** XML file specifying screen layout
  - **MainActivity.Java or MainActivity.kt:** Java or kotlin code to define behavior, actions taken when button clicked (intelligence)
  - **AndroidManifest.xml:**
    - Lists all app components and screens
    - Like a table of contents for a book
    - E.g. Hello world program has 1 screen, so AndroidManifest.xml has 1 item listed
    - App starts running here (a bit like main( ) in C), launching activity with a tag "LAUNCHER"



9:42
Pinterest
Green Papaya Salad recipe, healthy and delicious
Ash Huang
Backpacking through China. We should do this @jedmund!
Victor Ng
The skies over the UK.
Rob Mason
Minimal clock, this would look great in my new apartment!
Carl Rice

Widgets

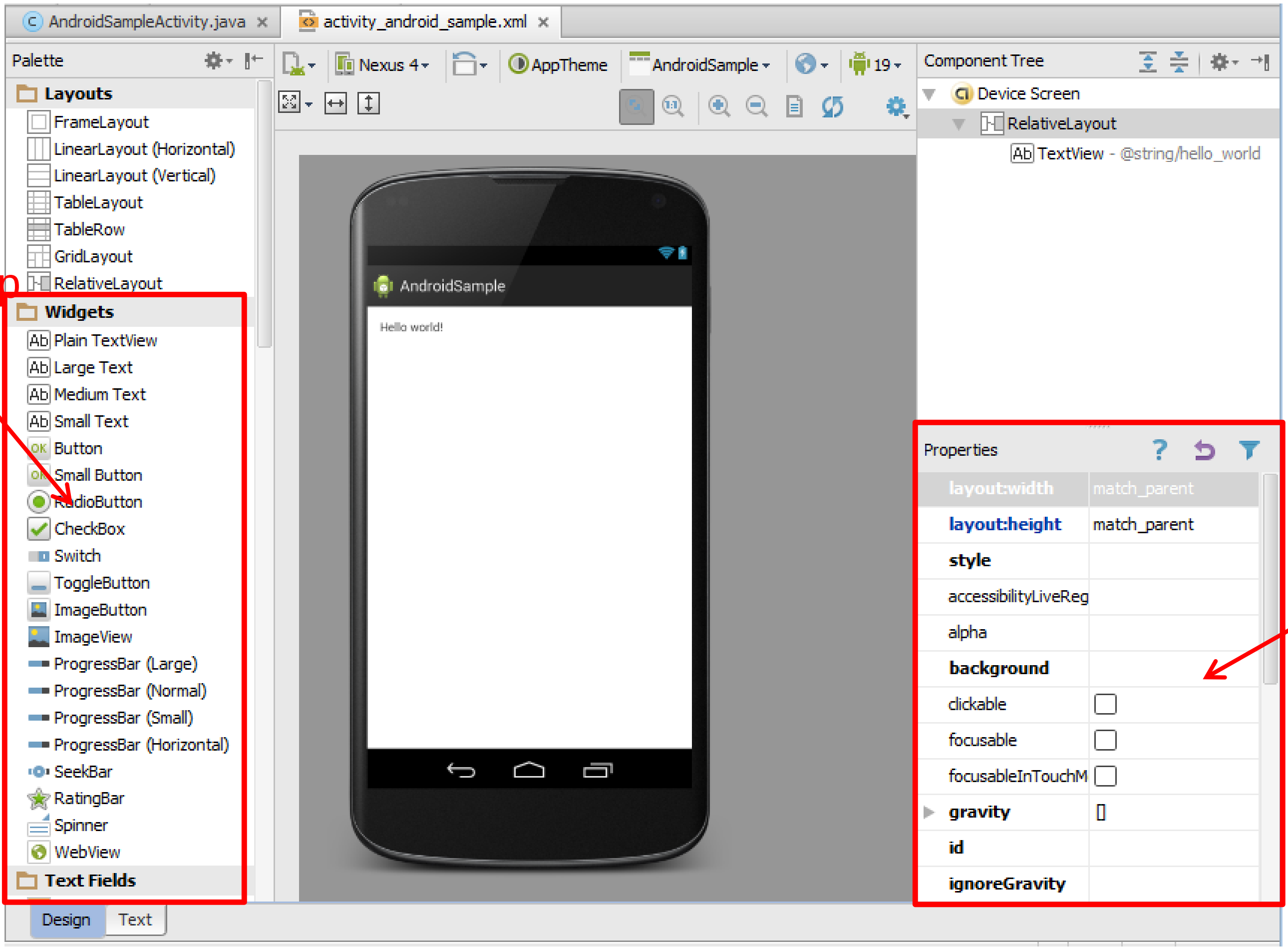
- ***Android UI design involves arranging widgets on a screen***
- **Widgets?** Rectangles containing texts, image, etc
- **Screen design:** Pick widgets, specify attributes (dimensions, margins, etc)



Design Option 1: Drag and Drop Widgets

- Drag and drop widgets in Android Studio Design View
- Edit widget properties (e.g., height, width, color, etc)

Drag and drop button or any other widget or view

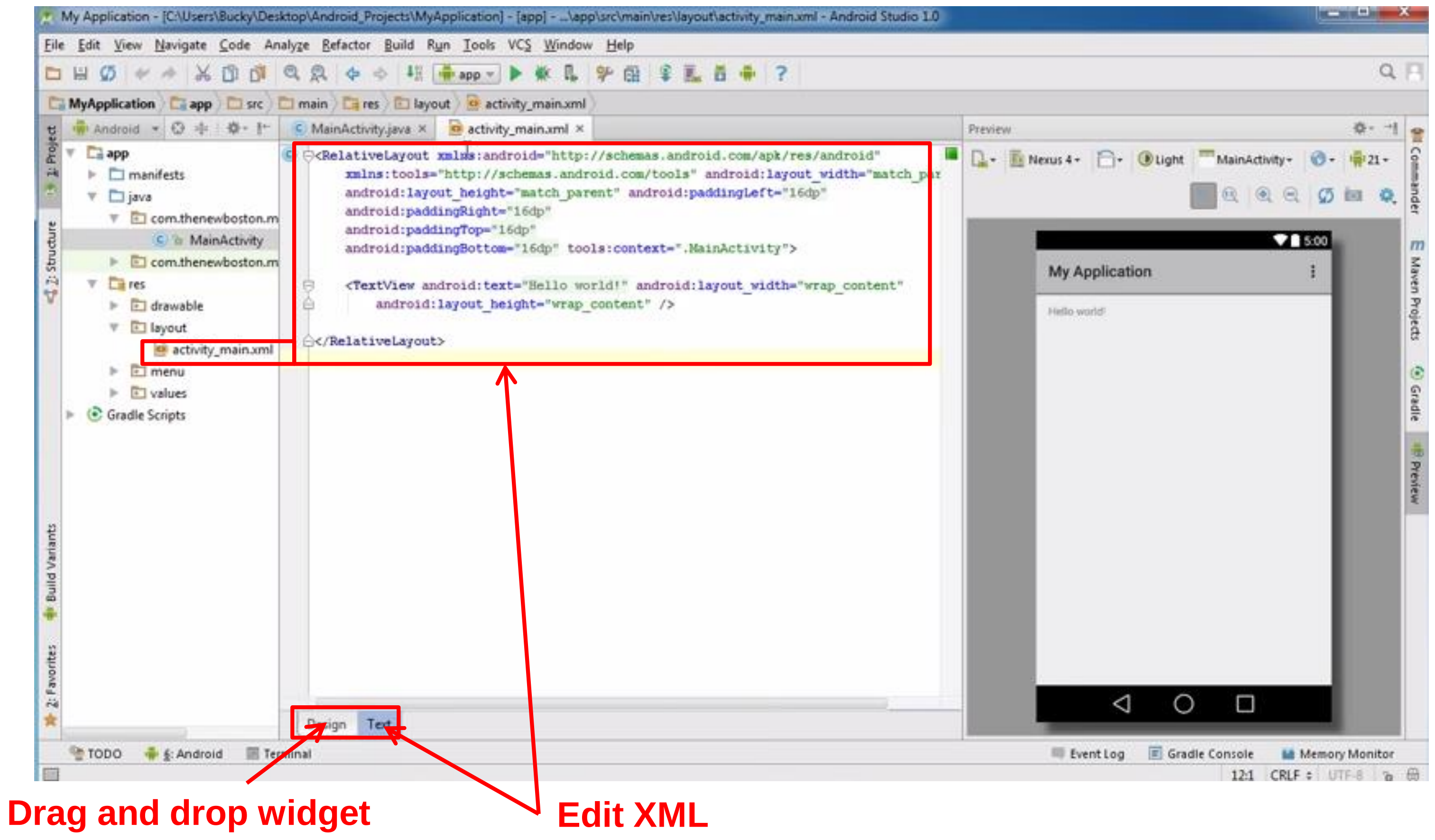


layout:width	match_parent
layout:height	match_parent
style	
accessibilityLiveReg	
alpha	
background	
clickable	☐
focusable	☐
focusableInTouchM	☐
gravity	[]
id	
ignoreGravity	

Edit widget properties

Design Option 2: Edit XML Directly

- **Text view:** Directly edit XML file defining screen (activity_main.xml)
- **Note:** dragging and dropping widgets in design view auto-generates corresponding XML in Text view


My Application - [C:\Users\Bucky\Desktop\Android_Projects\MyApplication] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 1.0
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main res layout activity_main.xml
MainActivity.java activity_main.xml
app
manifests
java
com.thenewboston.m
MainActivity
com.thenewboston.m
res
drawable
layout
activity_main.xml
menu
values
Gradle Scripts
Preview
Nexus 4
Light
MainActivity
21
My Application
Hello world!
Design
Text
TODO
6: Android
Terminal
Event Log
Gradle Console
Memory Monitor
Drag and drop widget
Edit XML

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_par
    android:layout_height="match_parent" android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp" tools:context=".MainActivity">

    <TextView android:text="Hello world!" android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

HW0: Android Setup/Getting Going

Project 0

- **Project 0:** available here: https://web.cs.wpi.edu/~emmanuel/courses/cs528/F23/projects/project0.html

- **Go through YouTube Android Studio tutorials (Parts 1-3):** by DJ Malone
- Create simple Android screens
- Due by class start time next Thursday, August 31, 2023

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