CS260 Intro to Java & Android 04.Android Intro

Winter 2015

Android - Getting Started

- Android SDK contains:
  - API Libraries
  - Developer Tools
  - Documentation
  - Sample Code

- Present development tools:
  - Eclipse with the Android Developer Tool (ADT) plugin which integrates developer tools
  - Android Studio

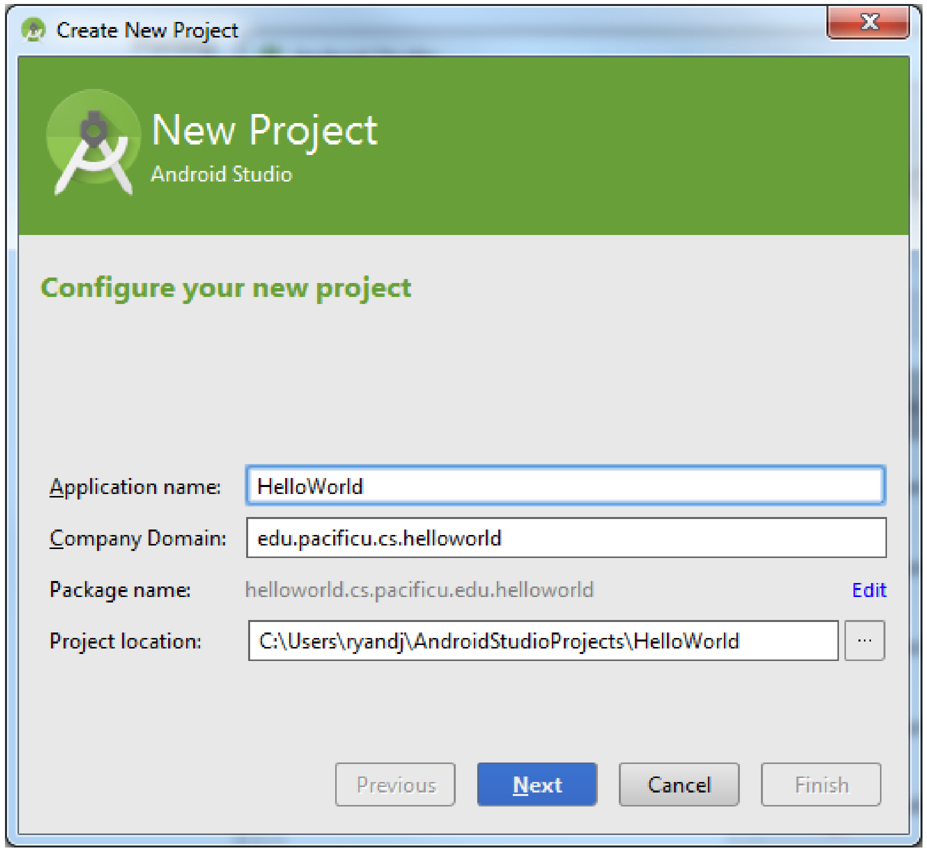
Android Portability

- Android applications run within the Dalvik virtual machine
- ART is a new Android runtime being introduced in 4.4

- Development Platforms:
  - Windows (XP, Windows, 7, 8)
  - Linux
  - Mac OS 10.4.8 or later (Intel chips only)

Android HelloWorld Application

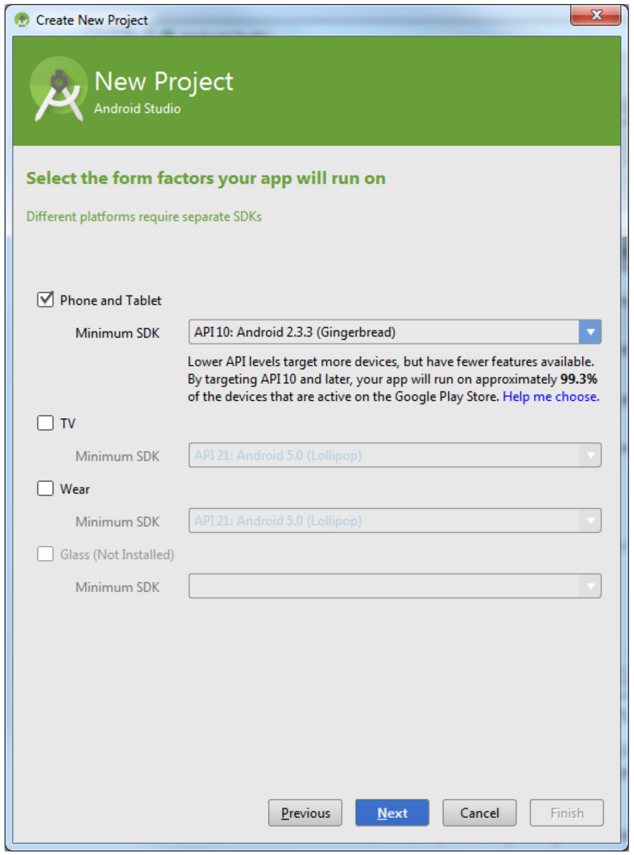
- Start Android Studio
- We will create our warm fuzzy HelloWorld



Configure your new project

Application name:	HelloWorld
Company Domain:	edu.pacificu.cs.helloworld
Package name:	helloworld.cs.pacificu.edu.helloworld Edit
Project location:	C:\Users\ryandj\AndroidStudioProjects\HelloWorld

New Android Project



Create New Project

New Project
Android Studio

Select the form factors your app will run on

Different platforms require separate SDKs

[✓] Phone and Tablet
Minimum SDK: API 10: Android 2.3.3 (Gingerbread)

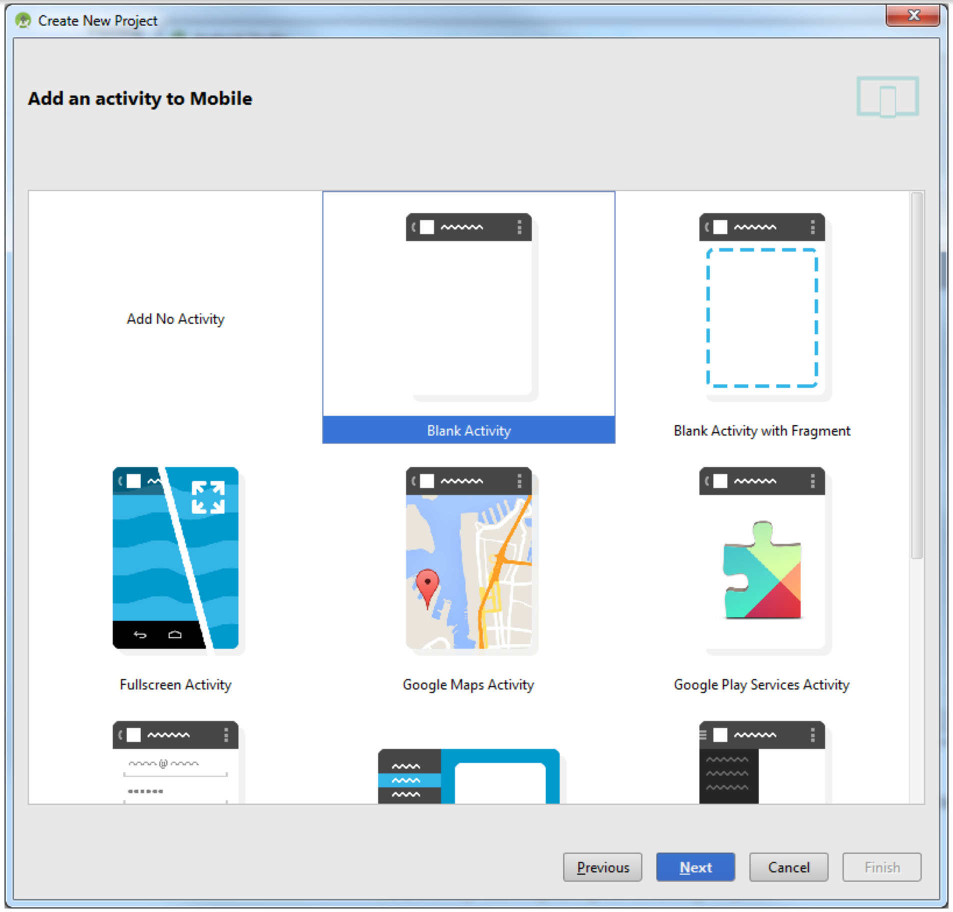
Lower API levels target more devices, but have fewer features available. By targeting API 10 and later, your app will run on approximately **99.3%** of the devices that are active on the Google Play Store. Help me choose.

[ ] TV
Minimum SDK: API 21: Android 5.0 (Lollipop)

[ ] Wear
Minimum SDK: API 21: Android 5.0 (Lollipop)

[ ] Glass (Not Installed)
Minimum SDK:

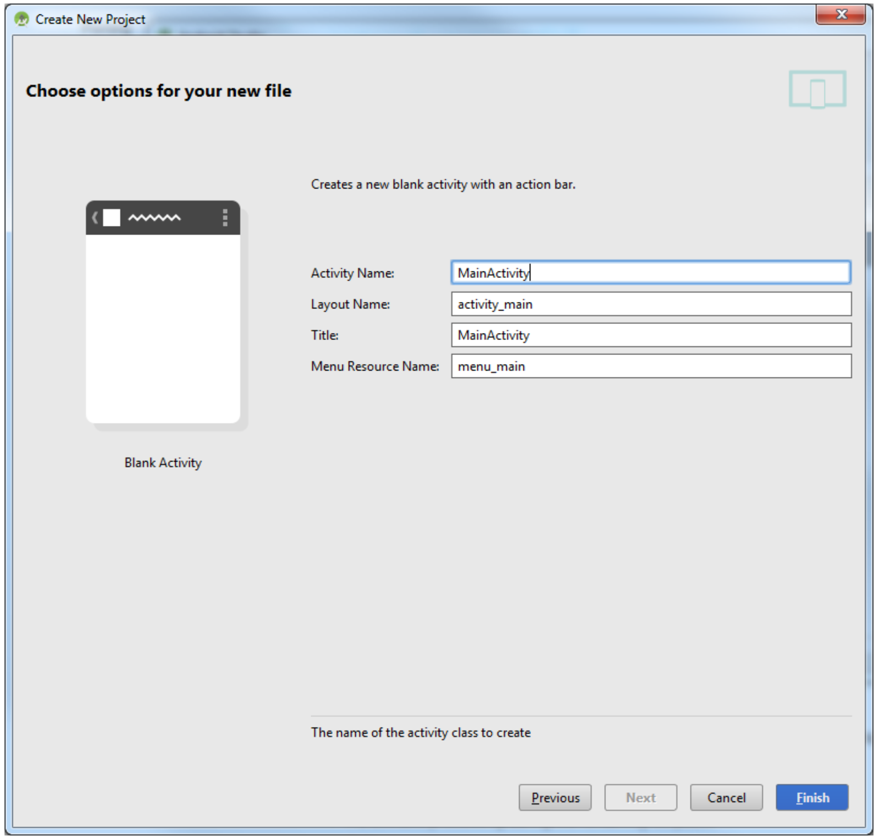
Click "Next" takes us to



Create New Project

Add an activity to Mobile

Click "Next" takes us to



Create New Project

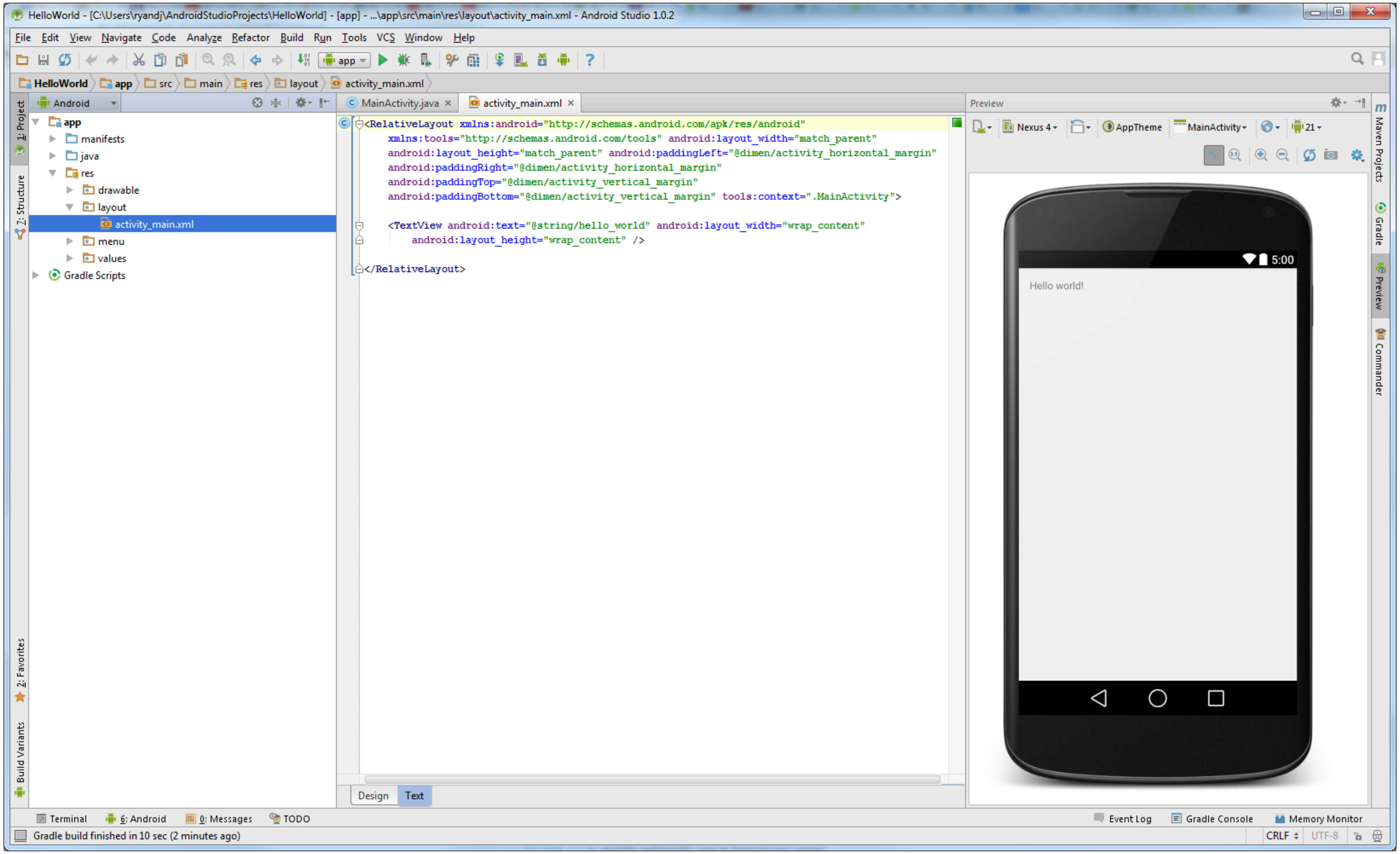
Choose options for your new file

Creates a new blank activity with an action bar.

Activity Name:	MainActivity
Layout Name:	activity_main
Title:	MainActivity
Menu Resource Name:	menu_main

The name of the activity class to create

Click “Finish” takes us to

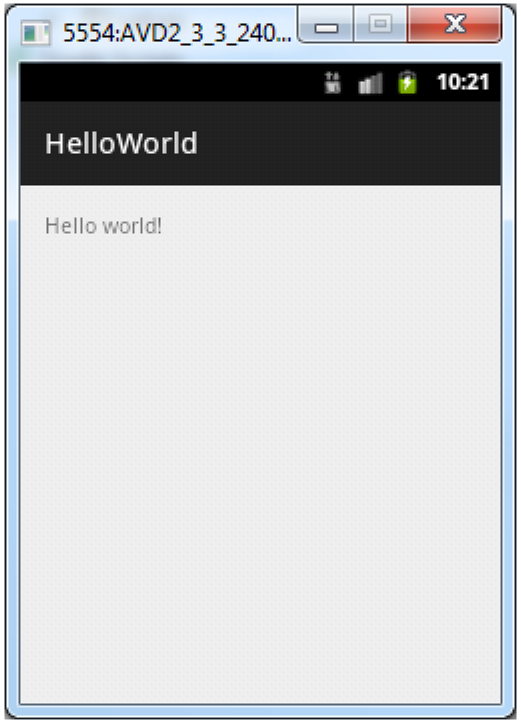


```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent" android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin" tools:context=".MainActivity">

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

</RelativeLayout>
```

Gradle build finished in 10 sec (2 minutes ago)

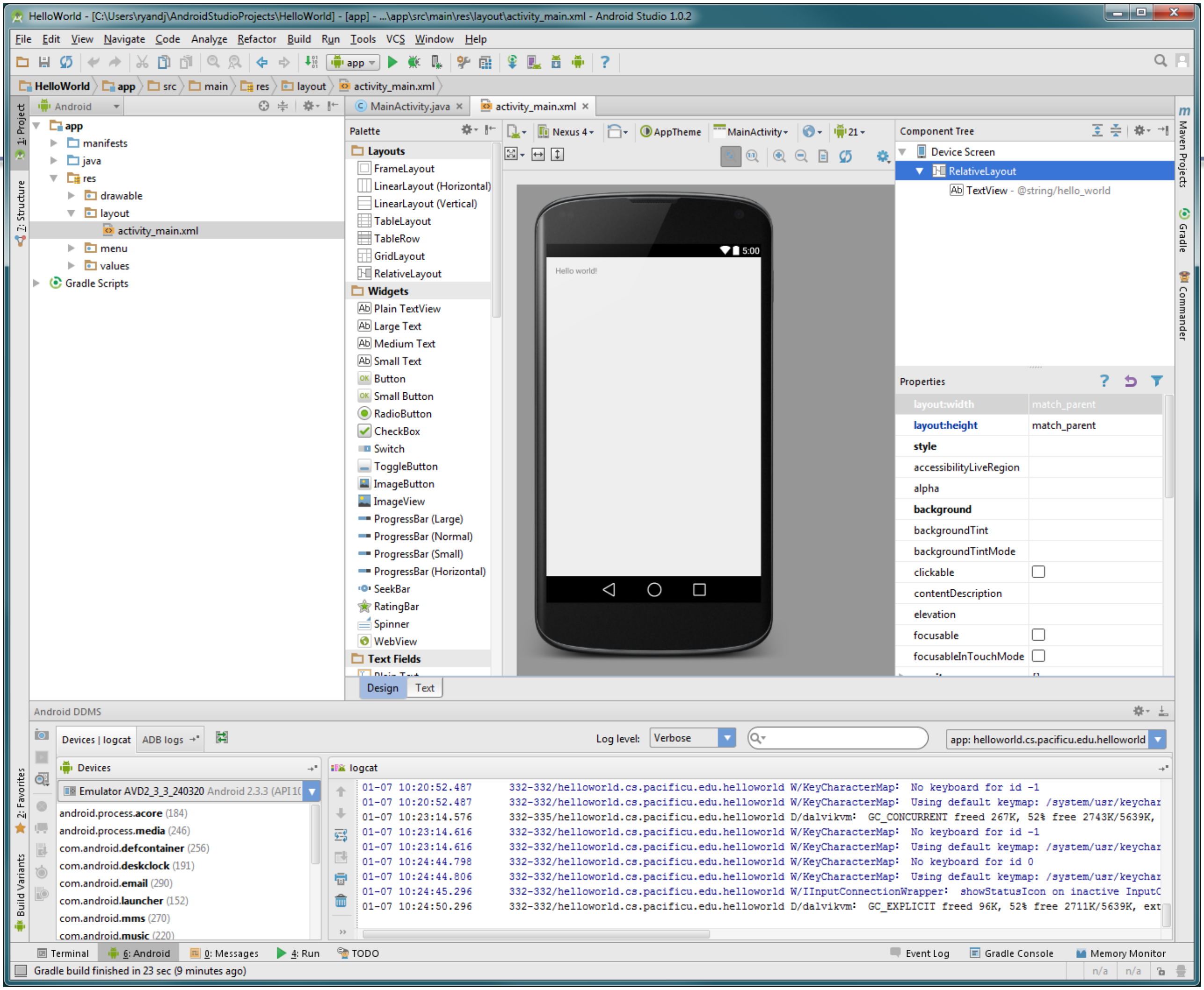
Run the Android Application

- Special keys
  - ➢ left ctl & F11 - landscape
  - ➢ Esc - back button
  - ➢ Home - Home
  - ➢ F3 - Call / Dial button
  - ➢ F4 - Hang up / end call
  - ➢ F5 - Search
- More Shortcuts
  http://www.shortcutworld.com/en/win/Android-Emulator.html



Hello world!

Design Mode

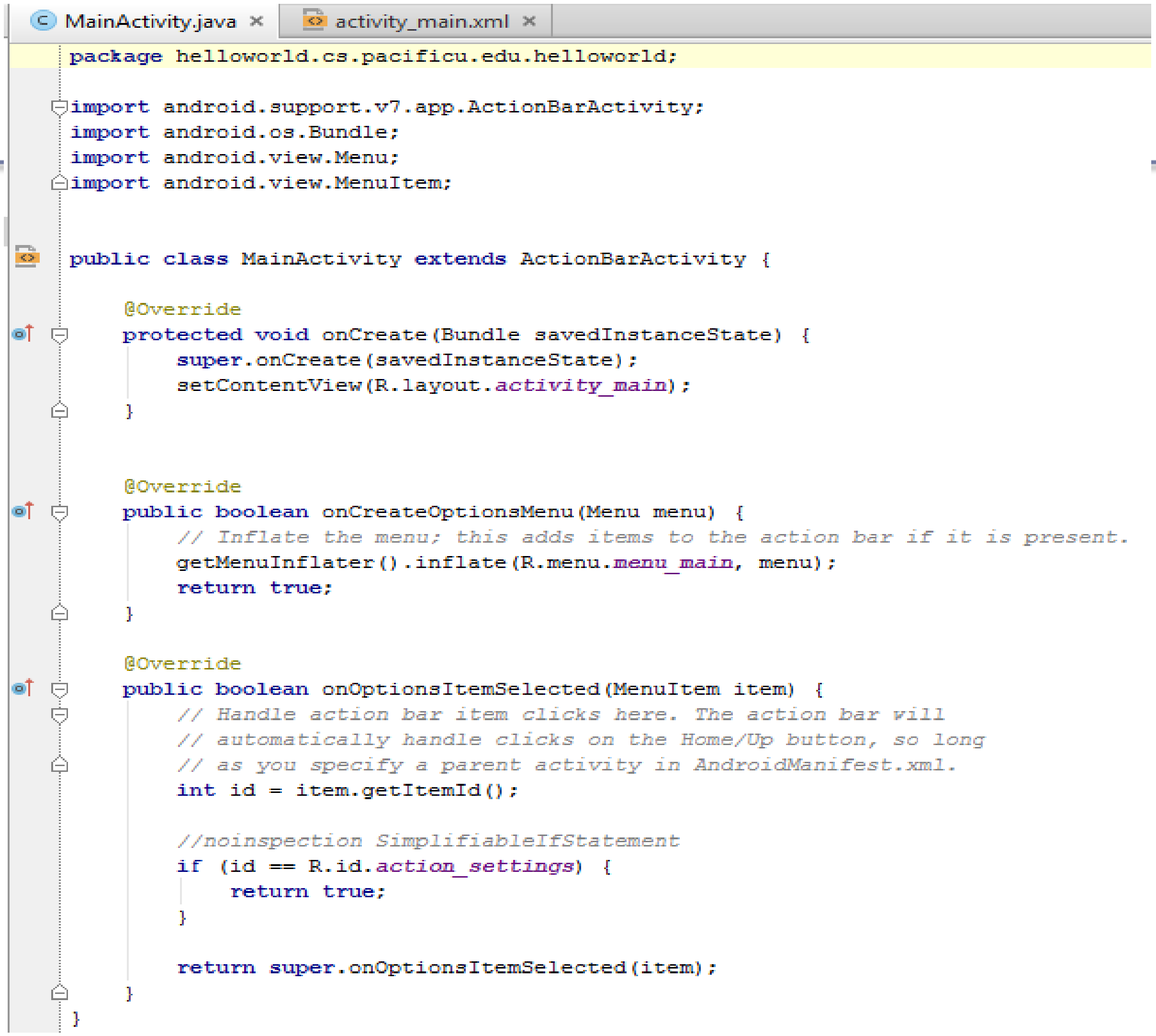


layout:width	match_parent
layout:height	match_parent
style	
accessibilityLiveRegion	
alpha	
background	
backgroundTint	
backgroundTintMode	
clickable	☐
contentDescription	
elevation	
focusable	☐
focusableInTouchMode	☐

```
android.process.acore (184)
android.process.media (246)
com.android.defcontainer (256)
com.android.deskclock (191)
com.android.email (290)
com.android.launcher (152)
com.android.mms (270)
com.android.music (220)
```

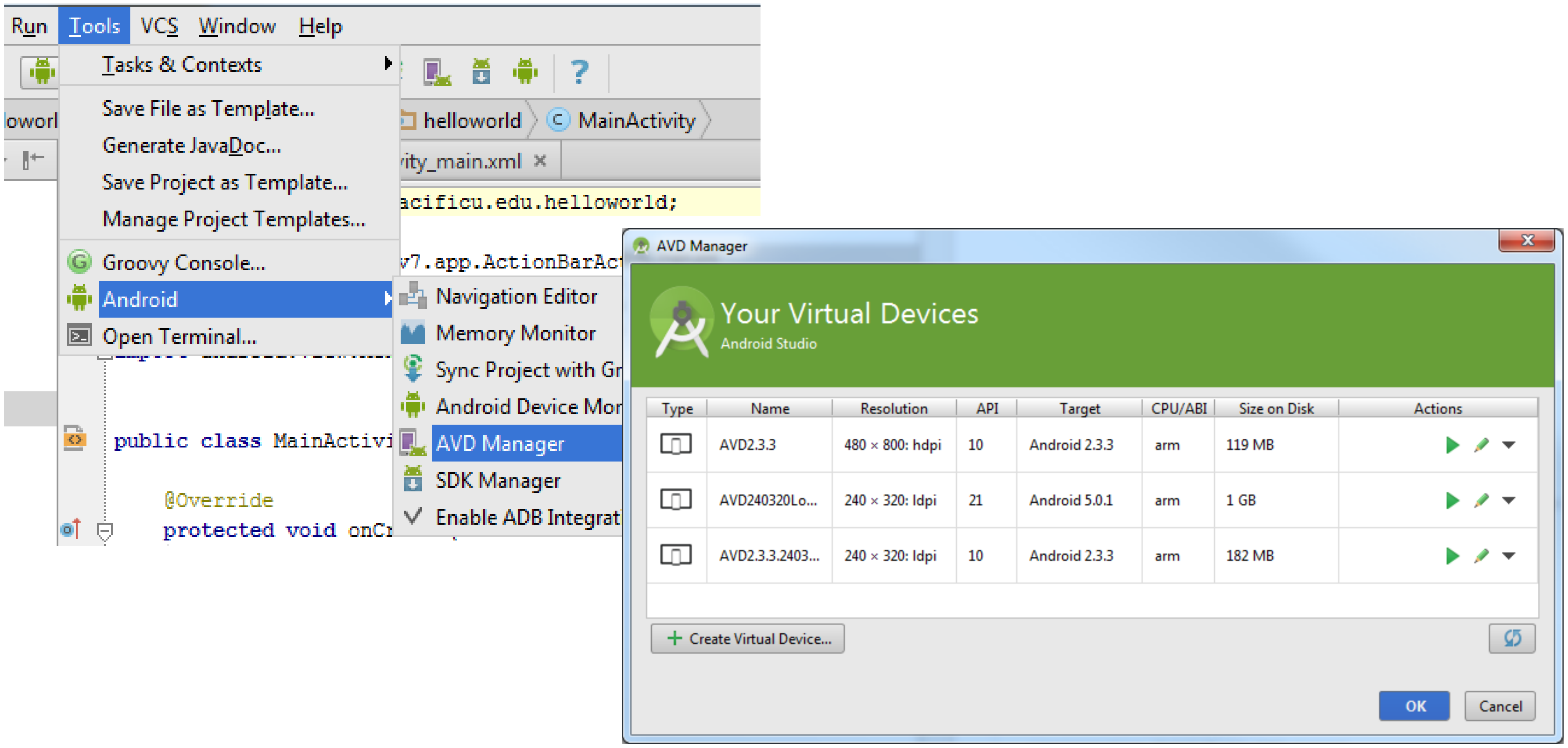
```
01-07 10:20:52.487     332-332/helloworld.cs.pacificu.edu.helloworld W/KeyCharacterMap:  No keyboard for id -1
01-07 10:20:52.487     332-332/helloworld.cs.pacificu.edu.helloworld W/KeyCharacterMap:  Using default keymap: /system/usr/keychar
01-07 10:23:14.576     332-335/helloworld.cs.pacificu.edu.helloworld D/dalvikvm:  GC_CONCURRENT freed 267K, 52% free 2743K/5639K,
01-07 10:23:14.616     332-332/helloworld.cs.pacificu.edu.helloworld W/KeyCharacterMap:  No keyboard for id -1
01-07 10:23:14.616     332-332/helloworld.cs.pacificu.edu.helloworld W/KeyCharacterMap:  Using default keymap: /system/usr/keychar
01-07 10:24:44.798     332-332/helloworld.cs.pacificu.edu.helloworld W/KeyCharacterMap:  No keyboard for id 0
01-07 10:24:44.806     332-332/helloworld.cs.pacificu.edu.helloworld W/KeyCharacterMap:  Using default keymap: /system/usr/keychar
01-07 10:24:45.296     332-332/helloworld.cs.pacificu.edu.helloworld W/IInputConnectionWrapper:  showStatusIcon on inactive InputC
01-07 10:24:50.296     332-332/helloworld.cs.pacificu.edu.helloworld D/dalvikvm:  GC_EXPLICIT freed 96K, 52% free 2711K/5639K, ext
```

HelloWorldAndroid Project



```java
package helloworld.cs.pacificu.edu.helloworld;

import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;


public class MainActivity extends ActionBarActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
}
```

Creating Virtual Devices



```
acificu.edu.helloworld;
v7.app.ActionBarAct
public class MainActivi
    @Override
    protected void onCr
```

Type	Name	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	AVD2.3.3	480 × 800: hdpi	10	Android 2.3.3	arm	119 MB	
	AVD240320Lo...	240 × 320: ldpi	21	Android 5.0.1	arm	1 GB	
	AVD2.3.3.2403...	240 × 320: ldpi	10	Android 2.3.3	arm	182 MB	

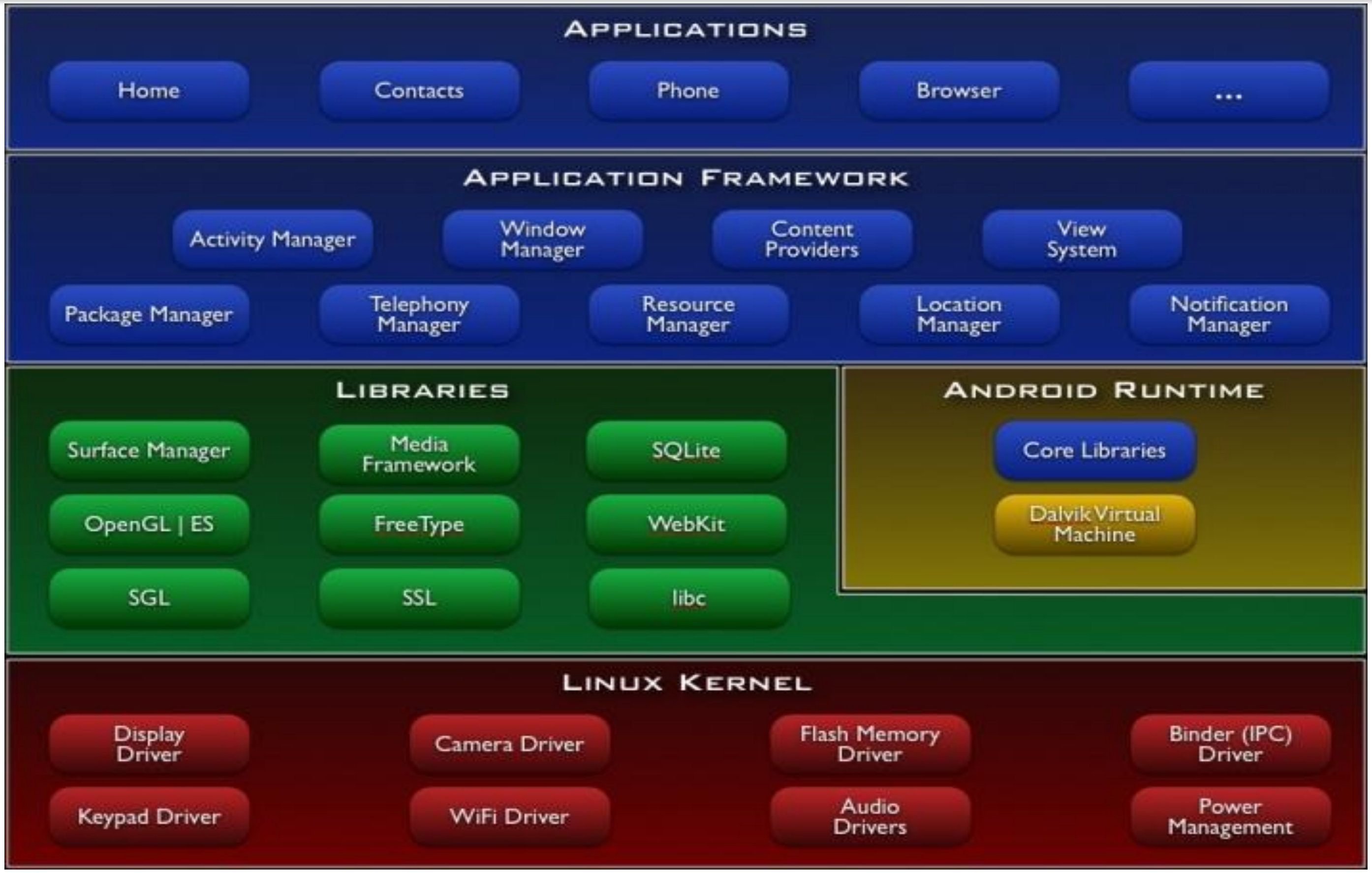
Important Android Dates

- Google acquires Android, August 2005
- Open Handset Alliance (OHA) announced, November 2007. OHA developed Android and is "…committed to commercially deploy handsets and services using the Android Platform." [10]
- First Android Phone, G1, October 2008
- Android SDK 1.0, October 2008

What is Android?

- Android is a software stack (set of programs working together) for mobile devices that includes:
  - an operating system
  - middleware
  - applications

Android Architecture

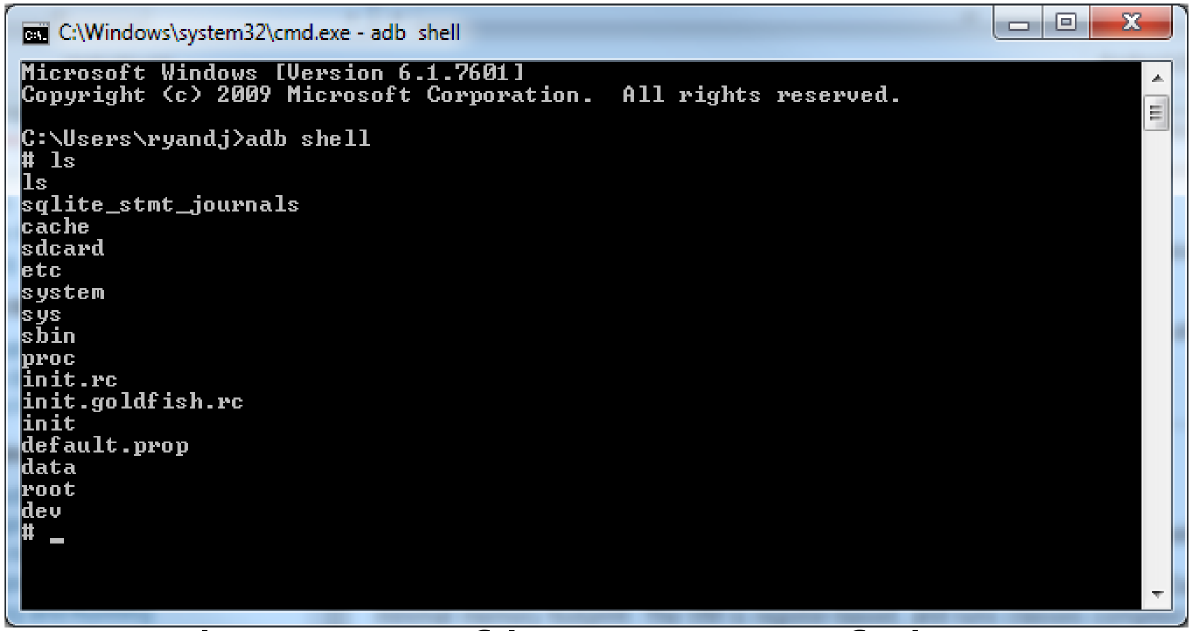


Linux Kernel

- Android relies on Linux version 2.6 (3.x from Android 4.0 Ice Cream Sandwich) for:
  - memory management
  - process management
  - security
  - networking
- You will not make Linux system calls
- Some utilities interact with Linux
  - e.g. adb shell

adb shell

- With an emulator running, open a Windows command shell
- Type adb shell
- Type ls



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\ryandj>adb shell
# ls
ls
sqlite_stmt_journals
cache
sdcard
etc
system
sys
sbin
proc
init.rc
init.goldfish.rc
init
default.prop
data
root
dev
# _
```

- Now you can examine the Linux file system of the phone which aids in in debugging

Native Libraries

- The native libraries are written in C & C++
- The libraries are exposed through the Application framework

Application Framework

- Android developers have access to the same framework APIs use by the core applications
- Services and systems for applications include:
  - **Views** – including lists, grids, buttons, ...
  - **Content Providers** – methods for accessing data
  - **Resource Manager** – organizes non-code resources such as strings and layout files
  - **Notification Manager** – displays custom alerts
  - **Activity Manager** – manages lifecycle of applications

Android Runtime

Every Application:

- Runs in its own process space
- Has a separate instance of the Dalvik VM
  - The Dalvik VM uses the Linux kernel for functionality such as threading and low-level memory management
  - Dalvik VM != JVM
- All Android code is written in Java and run within the Dalvik VM

What is Dalvik?

- Dalvik is a VM optimized for low memory requirements
- Android code is compiled into bytecodes executed by the Dalvik VM
- bytecodes are machine-independent instructions

Android Applications

- Apps are written in Java
- Code is compiled into Android package (.apk file)
- All code (including data & resource files) in .apk is one application

Android Application Specifics

- Android is a multi-user Linux system where each application is a user
- Only one application is visible at a time
- Each process has its own VM running an application in isolation
- Two or more applications can share data
- Applications consist of one or more activities

What is an Activity?

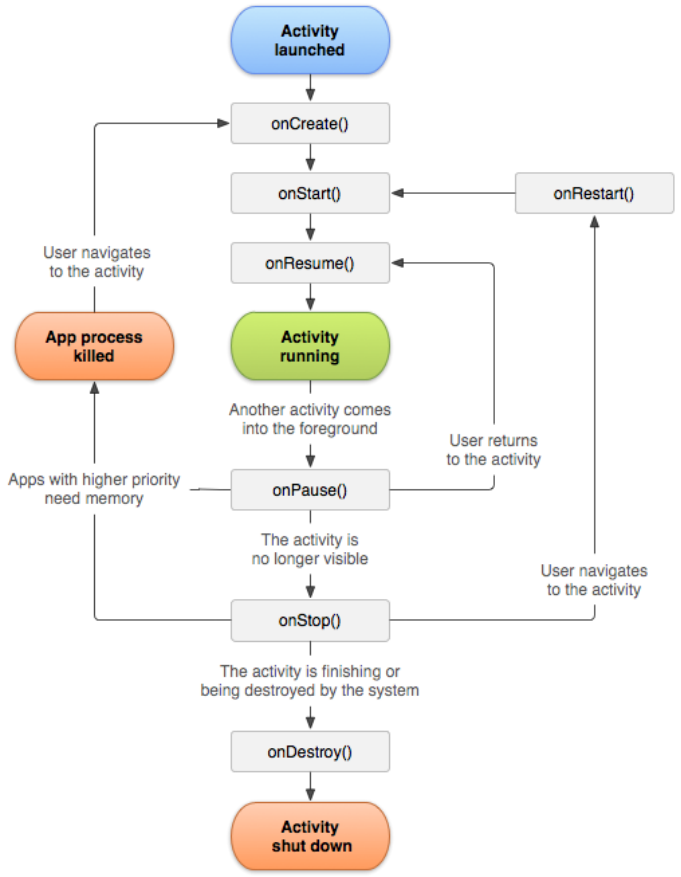
- An Activity represents a single screen with a UI
- Ex: Email Application consists of activities for
  - Showing list of emails
  - Composing an email
  - Reading an email
- Each activity is independent
- Other applications can use a particular activity if the email application gives permission to do so

Activity Lifecycle

Activity – a process that performs some specific action

- Every Android application is made up of one or more activities managed on an Activity Stack (AS) or the “back stack”.
- A new activity is always placed on top of the AS and then becomes the running activity.
- The AS is LIFO; therefore, when the Back button is pressed the current activity is popped and destroyed

Activity Lifecycle Visual

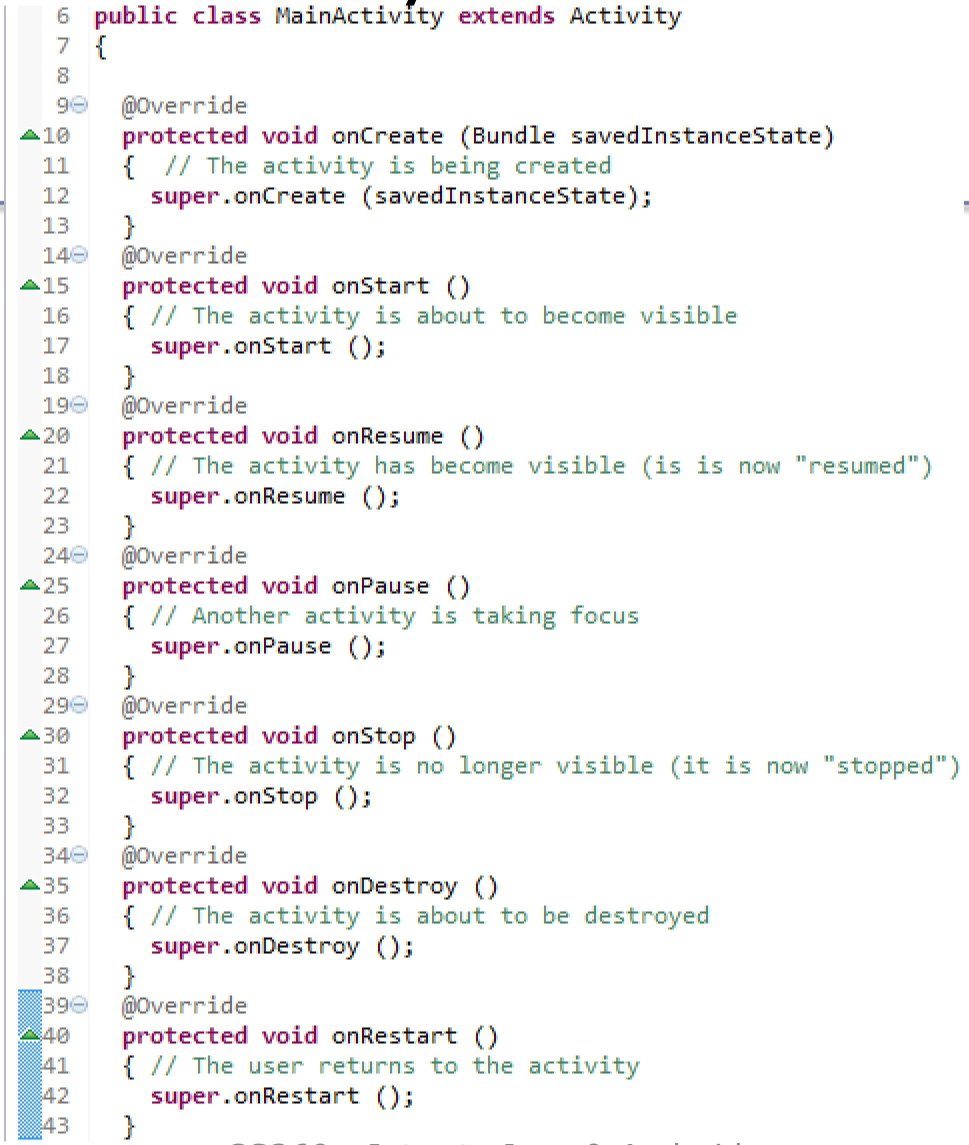

Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Activity States

An activity has essentially four states:

- **running** – in the foreground of the screen
- **paused** – lost focus but still visible with all state maintained
  - How? A new activity that is transparent or not full sized is running on top of the stack
- **stopped** – a new activity completely obscures another activity
  - The stopped activity is no longer visible
  - State is maintained
- **destroyed** – the activity must be completely restarted and the state information must be

Activity Skeleton



```
public class MainActivity extends Activity
{

  @Override
  protected void onCreate (Bundle savedInstanceState)
  {  // The activity is being created
    super.onCreate (savedInstanceState);
  }
  @Override
  protected void onStart ()
  { // The activity is about to become visible
    super.onStart ();
  }
  @Override
  protected void onResume ()
  { // The activity has become visible (is is now "resumed")
    super.onResume ();
  }
  @Override
  protected void onPause ()
  { // Another activity is taking focus
    super.onPause ();
  }
  @Override
  protected void onStop ()
  { // The activity is no longer visible (it is now "stopped")
    super.onStop ();
  }
  @Override
  protected void onDestroy ()
  { // The activity is about to be destroyed
    super.onDestroy ();
  }
  @Override
  protected void onRestart ()
  { // The user returns to the activity
    super.onRestart ();
  }
```

ActivityLifeCycleDemo Application

Copy the Android Project **ActivityLifeCycle** from CS260-01Public

1. Place the file in AndroidStudioProjects on your local machine
2. Let's take a look at the source code
3. Run the application

Q1: What is the difference between hitting the home button (HOME) and back button (ESC) ?

Q2: What is Log.v and how can it be used?