CS378 - Mobile Computing

Anatomy of an Android App and the App Lifecycle

Application Components

- five primary components
- different purposes and different lifecycles
- Activity
  - single screen with a user interface, app may have several activities, subclass of Activity
  - Most of early examples will be activities
- Intents
  - used to pass information between applications
- Service
  - Application component that performs long-running operations in background with no UI
  - example, an application that automatically responds to texts when driving

Application Components

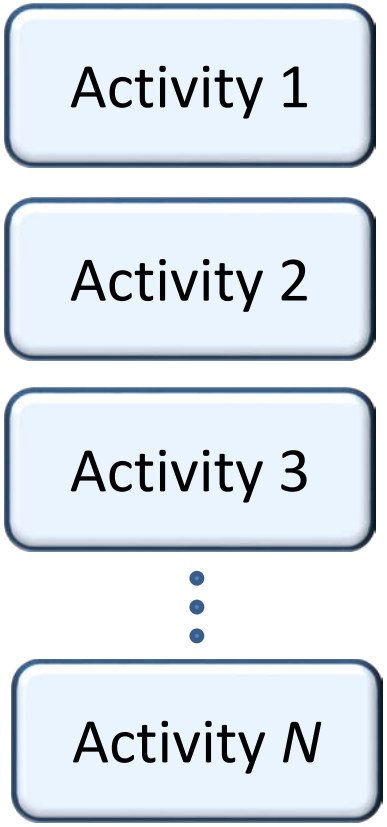
- Content Providers
  - a bridge between applications to share data
  - for example the devices contacts information
  - we tend to use these, but not create new ones
- Broadcast Receivers
  - component that responds to system wide announcements
  - battery low, screen off, date changed
  - also possible to initiate broadcasts from within an application

Hello Android

- Create an Activity
- Demonstrate resources created
- show the Activity lifecycle within the Android OS
- show the various debugging tools available
- show how to start one Activity from another

Activity Stack

Most recently created is at Top

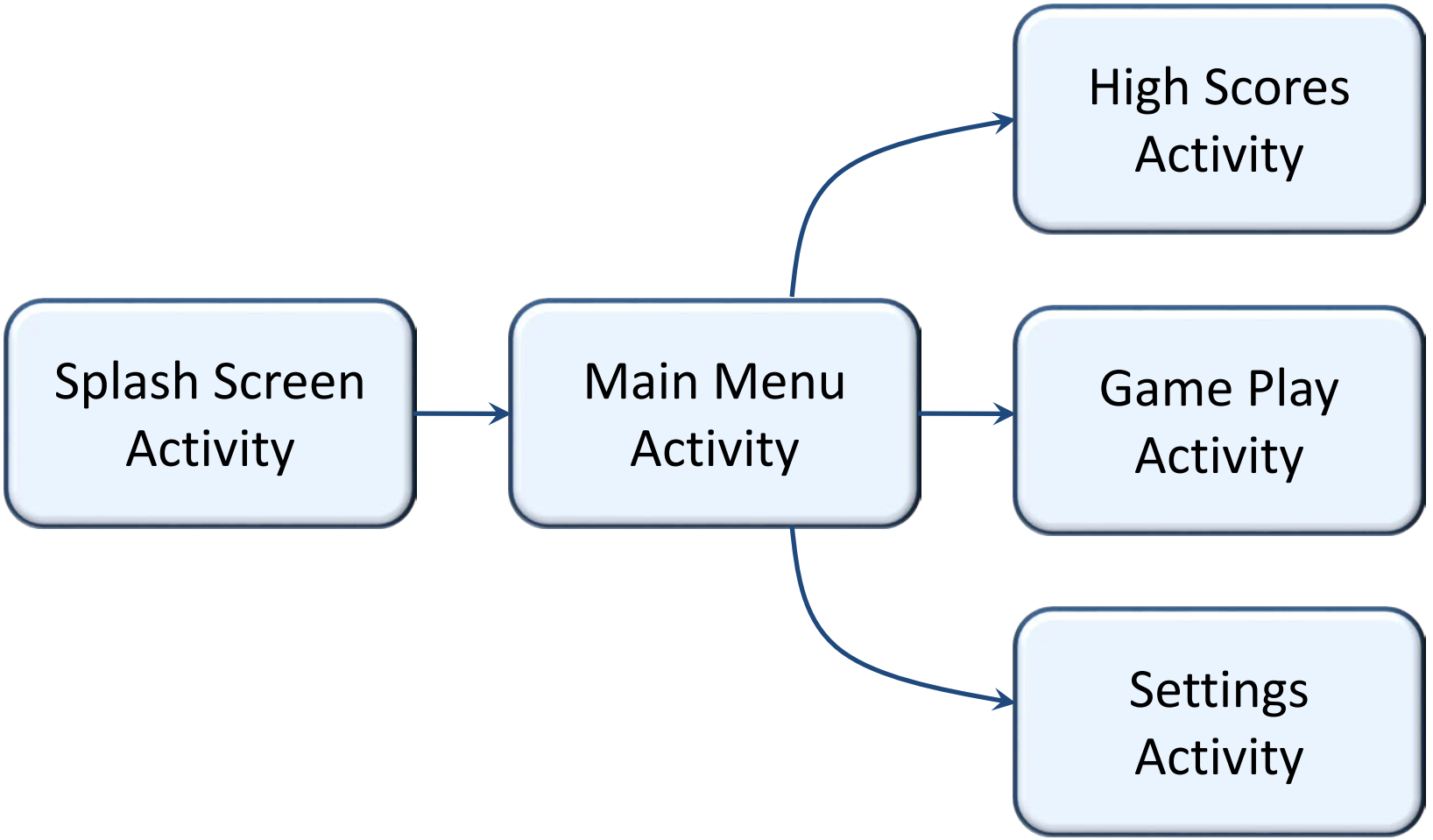


User currently interacting with me

Pressing Back or destroying A1 will bring me to the top

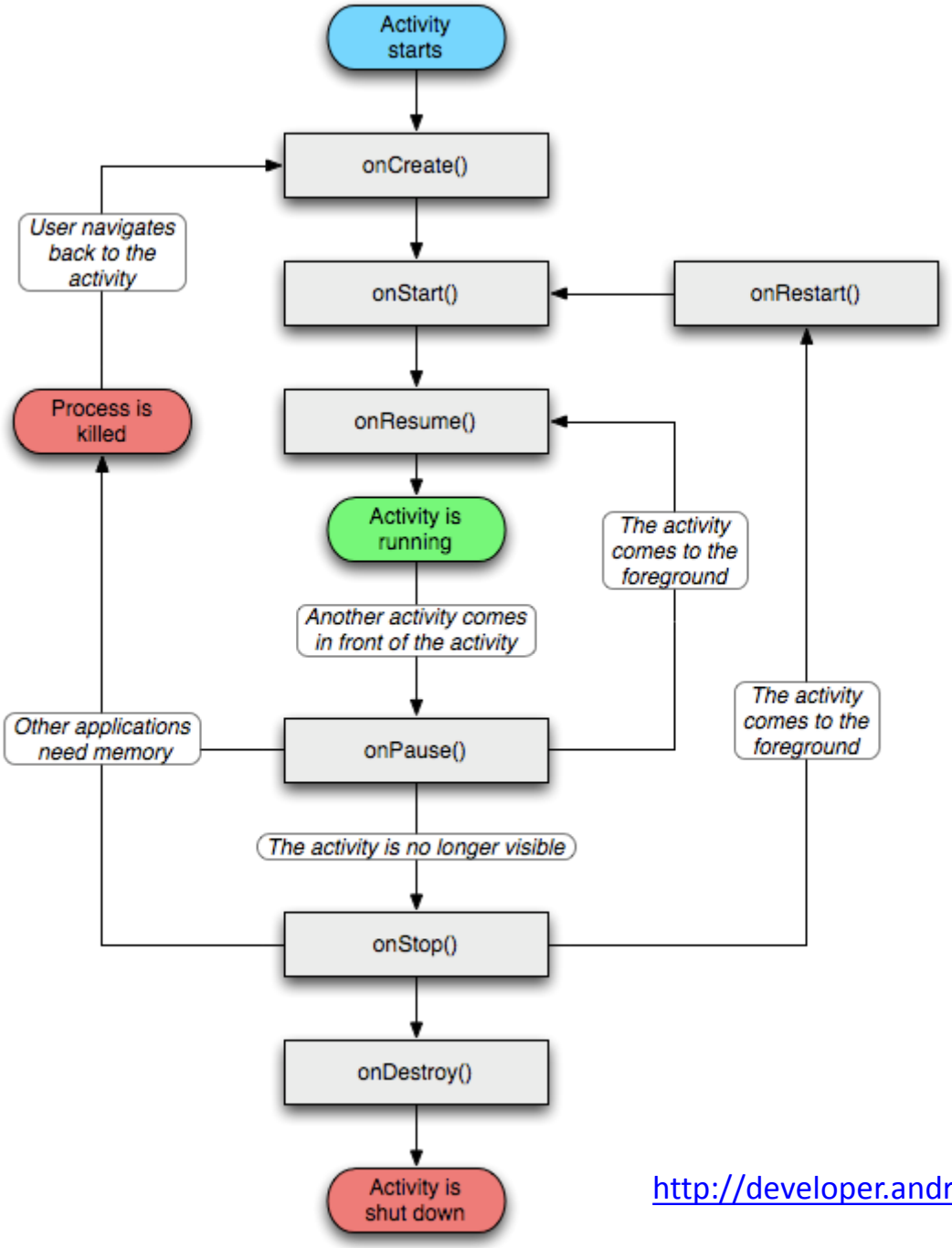
If Activities above me use too many resources, I'll be destroyed!

Typical Game



Conder & Darcey (2010), Fig 4.1, p. 74

Activity Lifecycle



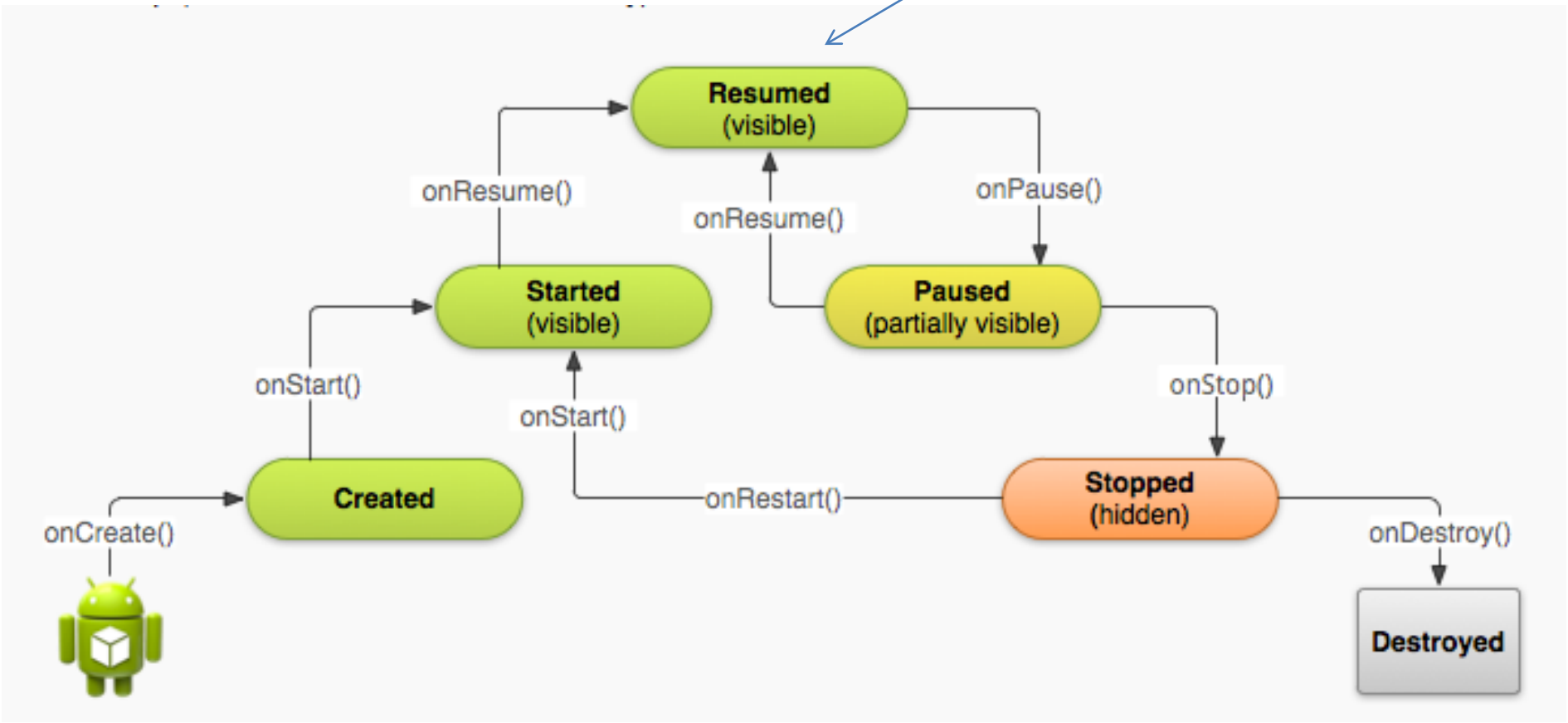
http://developer.android.com/reference/android/app/Activity.html

Starting Activities

- Android applications don't start with a call to main(String[])
- instead a series of callback methods are invoked
- each corresponds to specific stage of the Activity / application lifecycle
- callback methods also used to tear down Activity / application

Simplified Lifecycle Diagram

ready to interact with user



Understanding the Lifecycle

- Necessary to overload callback methods so your app behaves well:
- App should not crash if the user receives a phone call or switches to another app while using your app.
- App should not consume valuable system resources when the user is not actively using it.
- App should not lose the user's progress if they leave your app and return to it at a later time.
- App should not crash or lose the user's progress when the screen rotates between landscape and portrait orientation.

http://developer.android.com/training/basics/activity-lifecycle/starting.html

Primary States

- Active
  - activity is in the foreground and user can interact with it
- Paused
  - activity partially obscured by another activity and user cannot interact with it (for example when working with a menu or dialog)
- Stopped
  - activity completely hidden and not visible to user. It is in the background.
  - Activity instance and variables are retained but no code is being executed by the activity
- Dead, activity terminated (or never started)
- Two other states, Created and Started, but they are transitory onCreate -> onStart -> onResume

AndroidManifest.xml

All Activities that are part of application must be registered in Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="scott.examples.lifeCycleTest"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="10" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:name=".LifeCycleTestActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
            <activity
            android:name=".NameGetter"
            android:label="@string/getName"/>
    </application>
```

Specify Activity to start with

What is used for what?

- **Entire lifetime**: onCreate / onDestroy
  - Load UI
  - Could start and stop threads that should always be running
- **Visible lifetime**: onStart / onStop
  - Access or release resources that influence UI
- **Foreground lifetime**: onResume / onPause
  - Restore state and save state
  - Start and stop audio, video, animations

LifeCycleTest

- overload these methods from Activity:
  - onCreate(), onStart(), onResume(), onPause(), onStop(), onRestart(), onDestroy()
  - Use the Log class to log activity
  - methods: v, d, i, w, e
  - VERBOSE, DEBUG, INFO, WARN, ERROR
  - Create a TAG so we can filter

LifeCycleTest

- Run the app and open the Logcat view.
  - Eclipse Window-> Show View -> Other -> Android -> Logcat or via DDMS

```
protected void onStart() {
    super.onStart();
     Log.d(TAG, "in onStart Method");
}

protected void onRestart() {
    super.onRestart();
    Log.d(TAG, "in onRestart Method");
}

protected void onResume() {
    super.onResume();
    Log.d(TAG, "in onResume Method");
}

protected void onPause() {
    super.onPause();
    Log.d(TAG, "in onPause Method");
}

protected void onStop() {
    super.onStart();
    Log.d(TAG, "in onStop Method");
}

protected void onDestroy() {
    super.onDestroy();
    Log.d(TAG, "in onDestroy Method");
}
```

Logcat

- After app started

I	01-31 15:09:45.665	321		AndroidRuntime	NOTE: attach of threa
I	01-31 15:09:45.875	60	syste...	dalvikvm	Jit: resizing JitTabl
D	01-31 15:09:46.195	329	scott...	LIFECYCLE:	in onCreate Method
D	01-31 15:09:46.195	329	scott...	LIFECYCLE:	in onStart Method
D	01-31 15:09:46.195	329	scott...	LIFECYCLE:	in onResume Method
I	01-31 15:09:46.385	60	syste...	ActivityManager	Displayed scott.examp
I	01-31 15:09:46.385	60	syste...	ActivityManager	Displayed com.androic

Logcat

- Rotate emulator with CTRL+F-11

L...	Time	PID	Application	Tag	Text
D	01-31 15:09:46.195	329	scott...	LIFECYCLE:	in onCreate Method
D	01-31 15:09:46.195	329	scott...	LIFECYCLE:	in onStart Method
D	01-31 15:09:46.195	329	scott...	LIFECYCLE:	in onResume Method
I	01-31 15:09:46.385	60	syste...	ActivityManager	Displayed scott.examples.lifeCyc
I	01-31 15:09:46.385	60	syste...	ActivityManager	Displayed com.android.launcher/c
I	01-31 15:11:56.218	60	syste...	InputReader	Device reconfigured: id=0x0, nar
I	01-31 15:11:56.218	60	syste...	InputManager-Callbacks	No virtual keys found for device
I	01-31 15:11:56.625	60	syste...	ARMAssembler	generated scanline__00000177:035
I	01-31 15:11:56.675	60	syste...	ARMAssembler	generated scanline__00000077:035
I	01-31 15:11:56.745	60	syste...	ARMAssembler	generated scanline__00000177:035
I	01-31 15:12:00.491	60	syste...	WindowManager	Setting rotation to 1, animFlags
I	01-31 15:12:00.495	60	syste...	ActivityManager	Config changed: { scale=1.0 imsi
D	01-31 15:12:00.535	329	scott...	LIFECYCLE:	in onPause Method
D	01-31 15:12:00.535	329	scott...	LIFECYCLE:	in onStop Method
D	01-31 15:12:00.535	329	scott...	LIFECYCLE:	in onDestroy Method
D	01-31 15:12:00.565	329	scott...	LIFECYCLE:	in onCreate Method
D	01-31 15:12:00.565	329	scott...	LIFECYCLE:	in onStart Method
D	01-31 15:12:00.565	329	scott...	LIFECYCLE:	in onResume Method
D	01-31 15:12:02.844	60	syste...	dalvikvm	GC_EXPLICIT freed 341K, 47% free

Pausing - onPause method

- when activity paused you should
  - stop animations of other CPU intensive tasks
  - release resources such as broadcast receivers (app stops listening for broadcast info) and handles to sensors such as GPS device or handles to the camera
  - stop audio and video if appropriate

Stopping - onStop()

- Many scenarios cause activity to be stopped
- Well behaved apps save progress and restart seamlessly
- Activity stopped when:
  - user performs action in activity that starts another activity in the application
  - user opens Recent Apps window and starts a new application
  - user receives phone call
- use onStop to release all resources and save information (persistence)

How to stop an Activity yourself?

- Generally, don't worry about it!
- "**Note:** In most cases, you should not explicitly finish an activity using these methods. As discussed in the following section about the activity lifecycle, the Android system manages the life of an activity for you, so you do not need to finish your own activities. Calling these methods could adversely affect the expected user experience and should only be used when you absolutely do not want the user to return to this instance of the activity."
- methods: finish(), finishActivity()

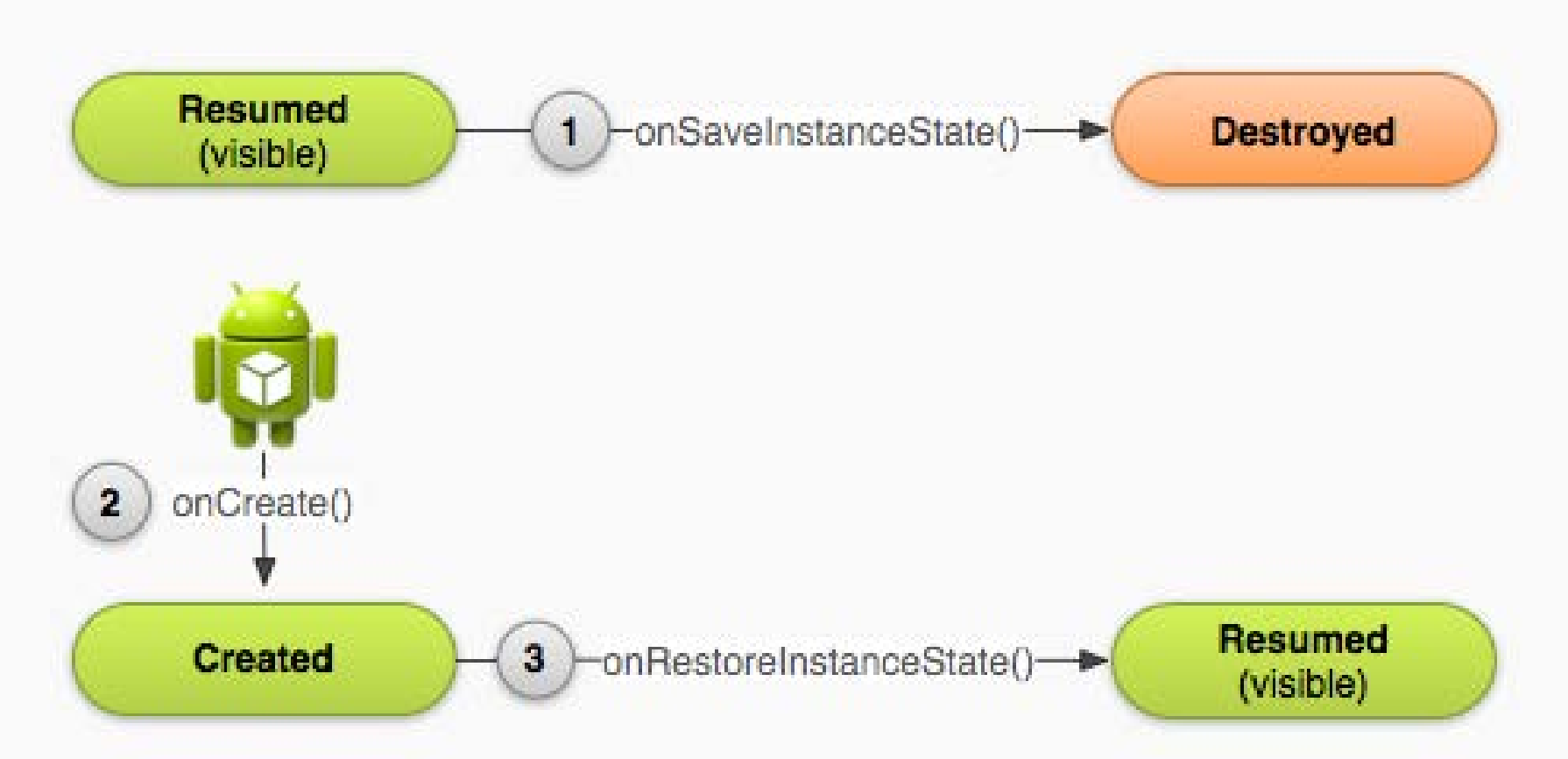
Saving State

- activities that are paused or stopped the state (instance vars) of the activity are retained
  - even if not in foreground
- When activity destroyed the Activity object is destroyed
  - can save information via onSaveInstanceState method. Write data to Bundle, Bundle given back when restarted

Activity Destruction

- app may be destroyed under normal circumstances
  - on its own by calling finish or user pressing the back button to navigate away from app
  - normal lifecycle methods handle this onPause() -> onStop() -> onDestroy
- If the system must destroy the activity (to recover resources or on an orientation change) must be able to recreate Activity

Activity Destruction



Activity Destruction

- If Activity destroyed with potential to be recreate later
- system calls the onSaveInstanceState (Bundle outState) method
- Bundle is a data structure like a Map
  - String keys
  - put methods for primitives, arrays, Strings, Serializables (Java), and Parcels (android)

onSaveInstanceState onRestoreInstanceState()

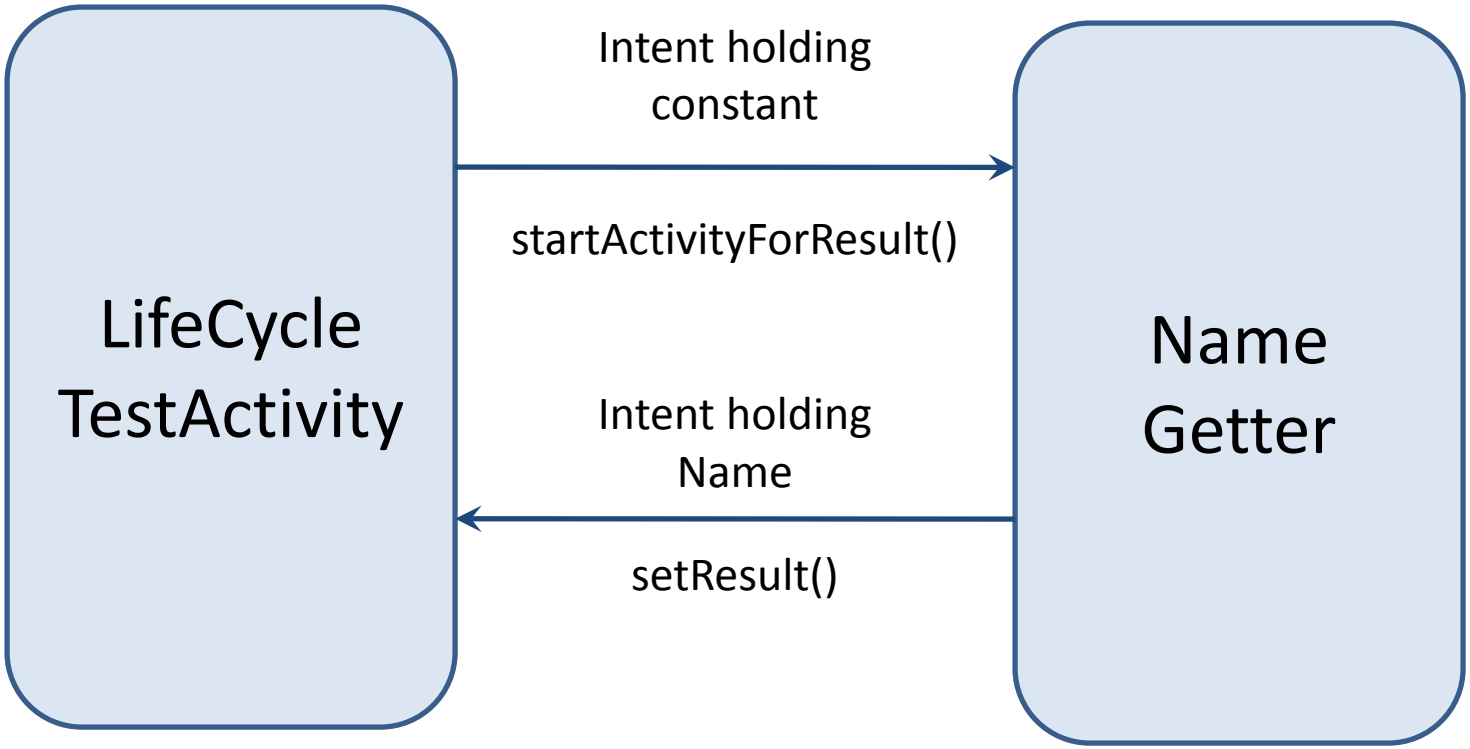
- systems write info about views to Bundle
- other information must be added by programmer
  - example, board state for mastermind
- When Activity recreated Bundle sent to onCreate and onRestoreInstanceState()
- use either method to restore state data / instance variables

Starting You Own Activities

- You will often start new Activities within your Activity
  - accomplish a task
  - get some data
- Click Button to get name
  - on button click (look at xml)
  - create an intent
  - call startActivityForResult
  - override onActivityResult()
  - add new Activity to Manifest
  - add data to intent, setResult, finish

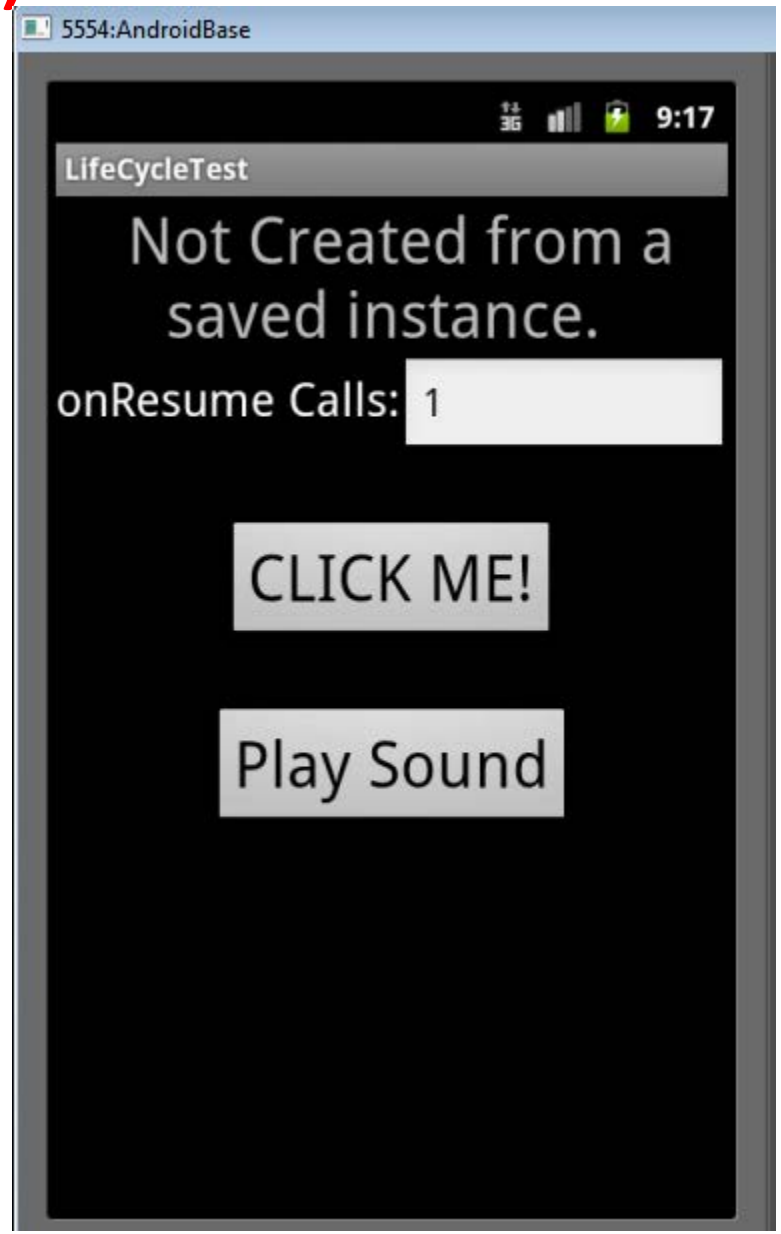
http://developer.android.com/guide/topics/fundamentals/activities.html#StartingAnActivity

Intent Demo


LifeCycle
TestActivity
Intent holding
constant
startActivityForResult()
Intent holding
Name
setResult()
Name
Getter

Playing Well (or not) With Others

- The Play Sound button causes a MediaPlayer to be created and plays a sound
- The Lifecycle app does not clean up after itself
- If app destroyed MediaPlayer keeps playing!!



References

- Android Introduction by Marko Gargenta, http://www.lecturemaker.com/2009/10/android-software-platform/
- Android Dev Guide http://developer.android.com/guide/topics/fundamentals.html http://developer.android.com/guide/topics/fundamentals/activities.html
- *Pro Android* by Hashimi & Komatineni (2009)
- Frank McCown, Harding University