

Recording in Progress

This class is being recorded

Please turn off your video and/or video if you do not wish to be recorded

CMSC436: Programming Handheld Systems

The Activity Class

Today's Topics

The Activity class

The Task Backstack

The Activity lifecycle

Starting an Activity

Handling configuration changes

The Activity Class

Provides a visual interface for user interaction

Conceptually*, each Activity typically supports one focused thing a user can do, such as

- Viewing an email message

- Showing a login screen

* Often implemented with help of other Android components

Activities and Application

Applications can comprise multiple Activities

User interaction can result in navigating across these Activities

Android's Navigation Support

Tasks

The Task Backstack

Suspending and resuming Activities

Tasks

A collection of activities that users interact with when trying to do something in an app

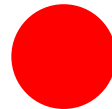
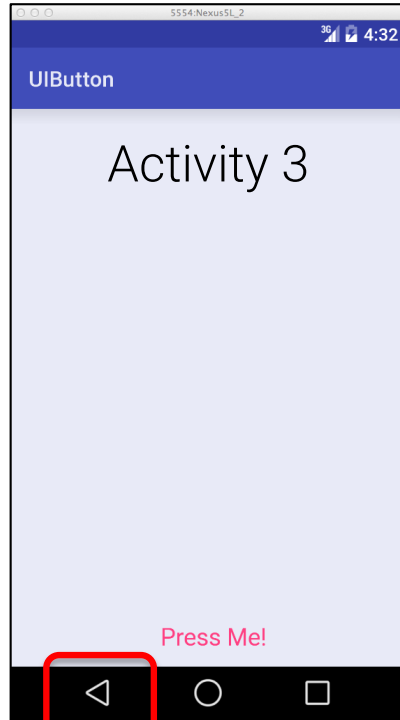
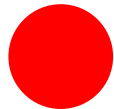
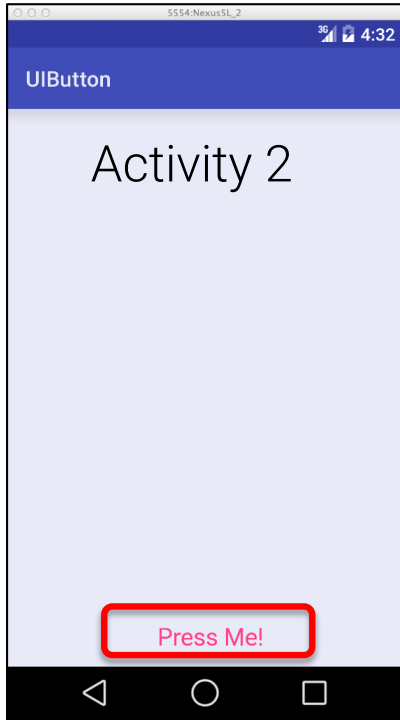
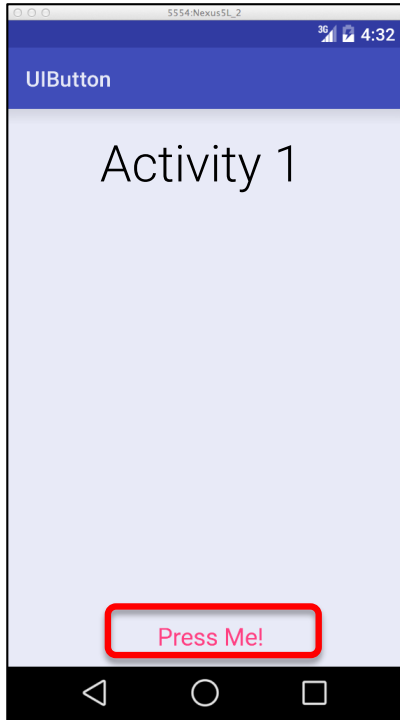
Can come from different applications

Most Tasks start at the home screen

Task Backstack

When an Activity is launched, it goes on top of the backstack

When the Activity is destroyed, it is popped off the backstack



Activity 3
Activity 2
Activity 1

Task Backstack

The Activity Lifecycle

Activities are created, suspended, resumed and destroyed as necessary when an application executes

Some of these actions depend on user behavior

e.g., User hits back button

Some depend on Android

e.g., Android can kill Activities when it needs their resources

Activity Lifecycle States

Resumed/Running—Visible, user interacting

Paused—Visible, user not interacting, can be terminated in older versions of Android

Stopped—Not visible, can be terminated

The Activity Lifecycle Methods

Android announces Activity lifecycle state changes to Activities by calling specific Activity methods

Known as Activity lifecycle callback methods

Some Activity Callback Methods

protected open fun onCreate(savedInstanceState: Bundle?): Unit

protected open fun onStart(): Unit

protected open fun onResume(): Unit

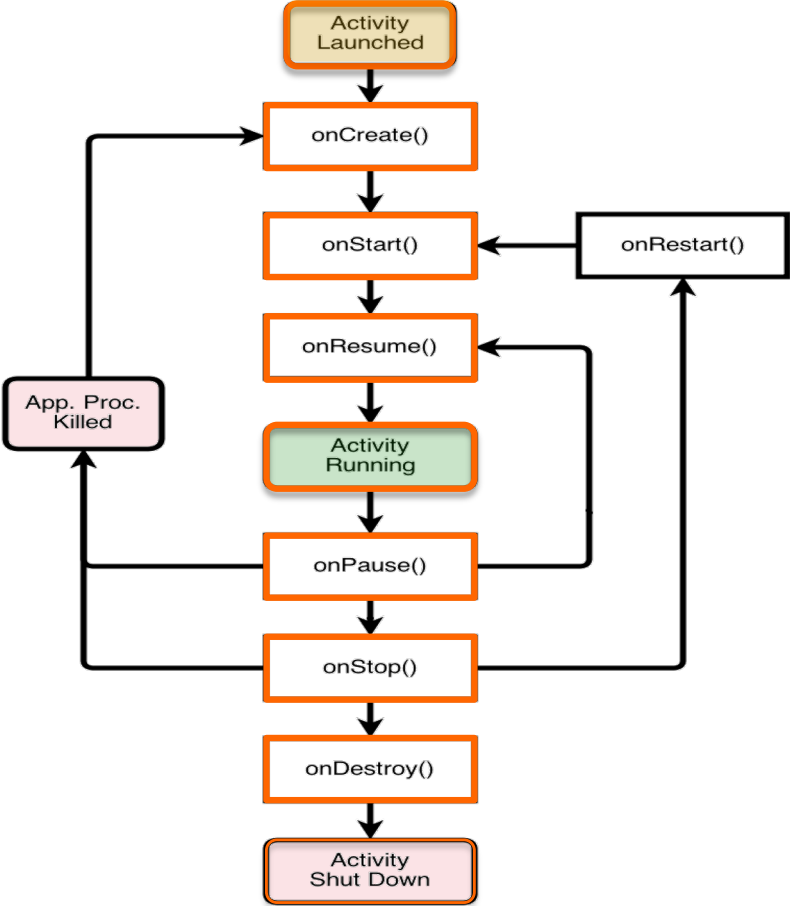
protected open fun onPause(): Unit

protected open fun onRestart(): Unit

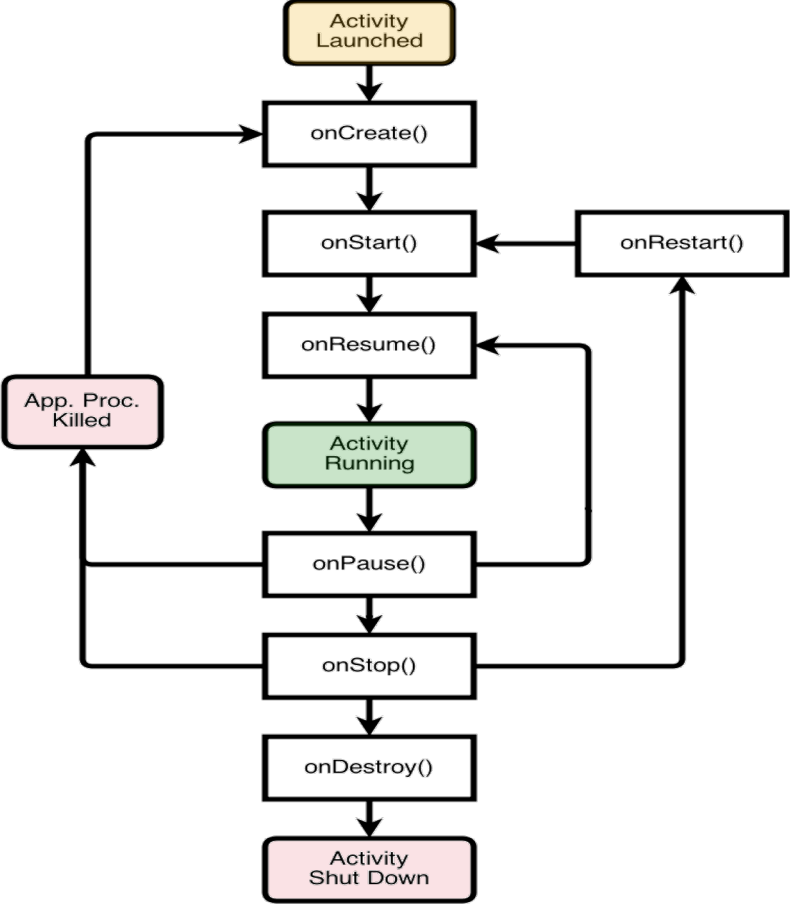
protected open fun onStop(): Unit

protected open fun onDestroy(): Unit

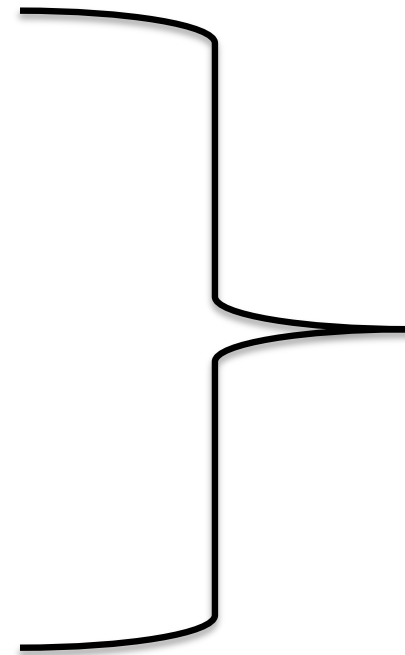
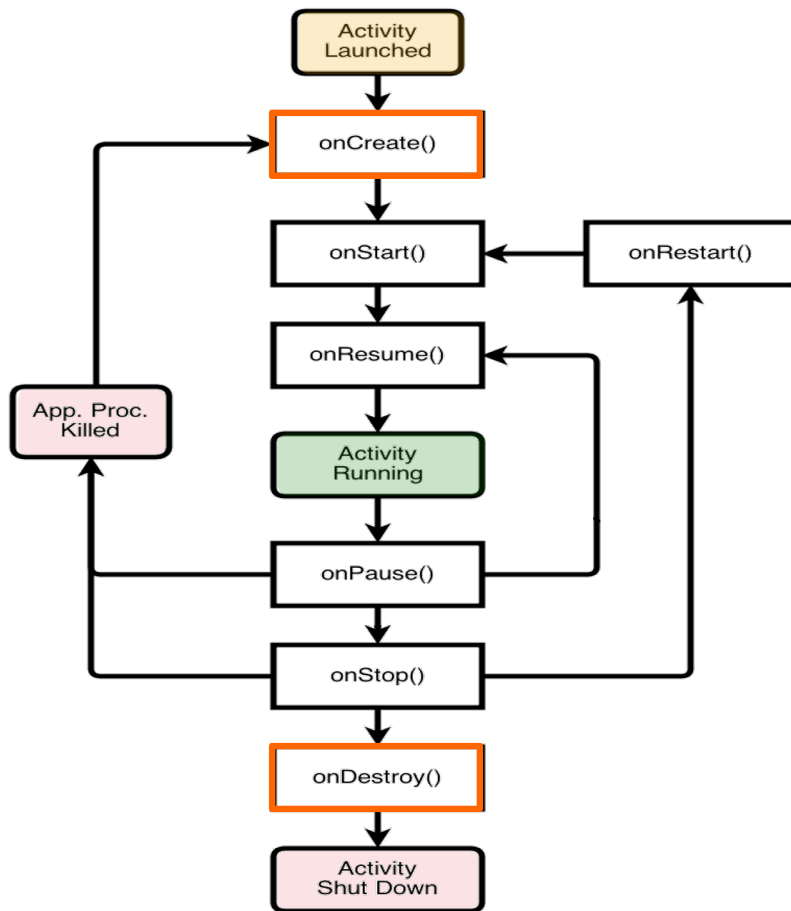
The Activity Lifecycle



The Activity Lifecycle

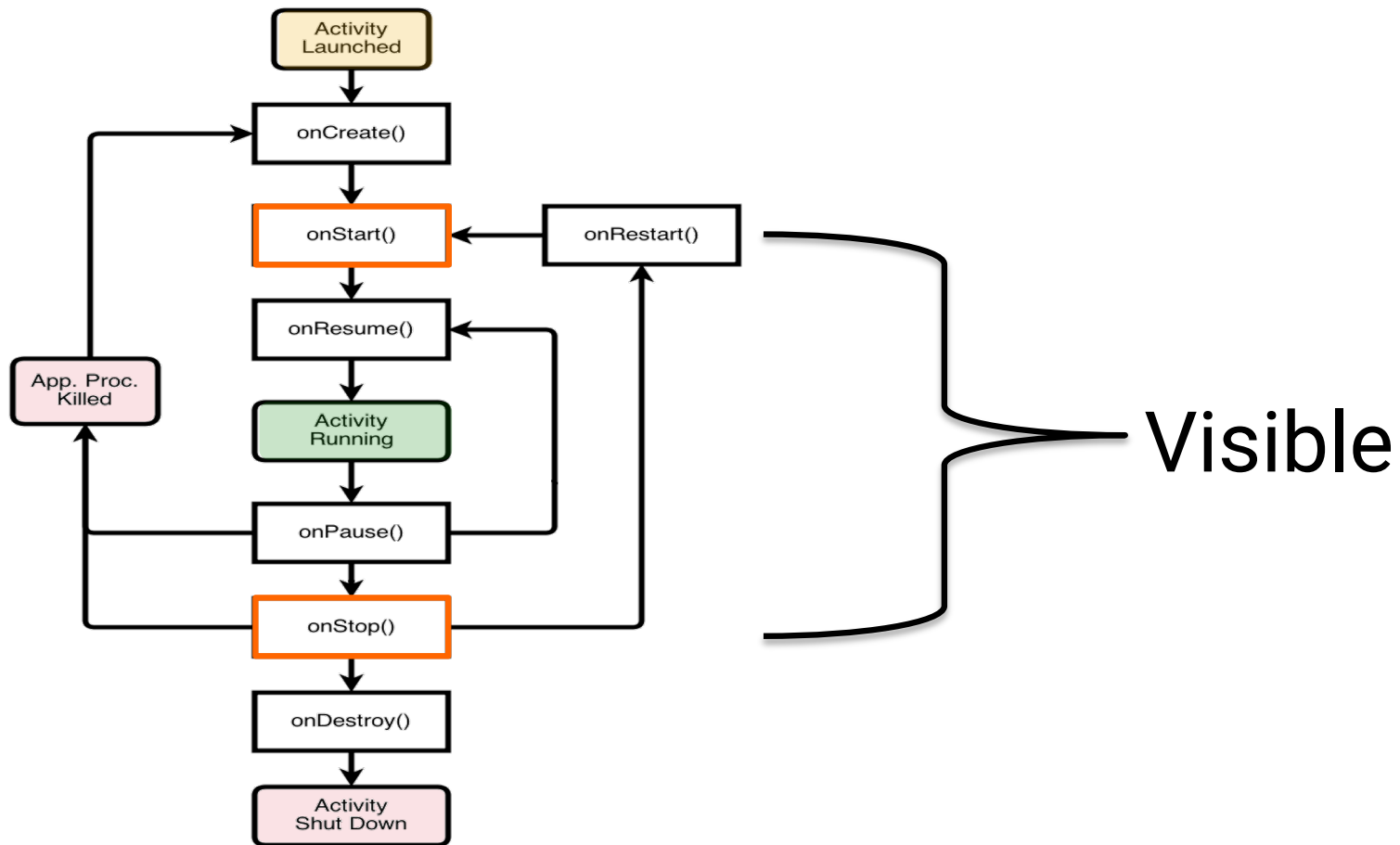


The Activity Lifecycle

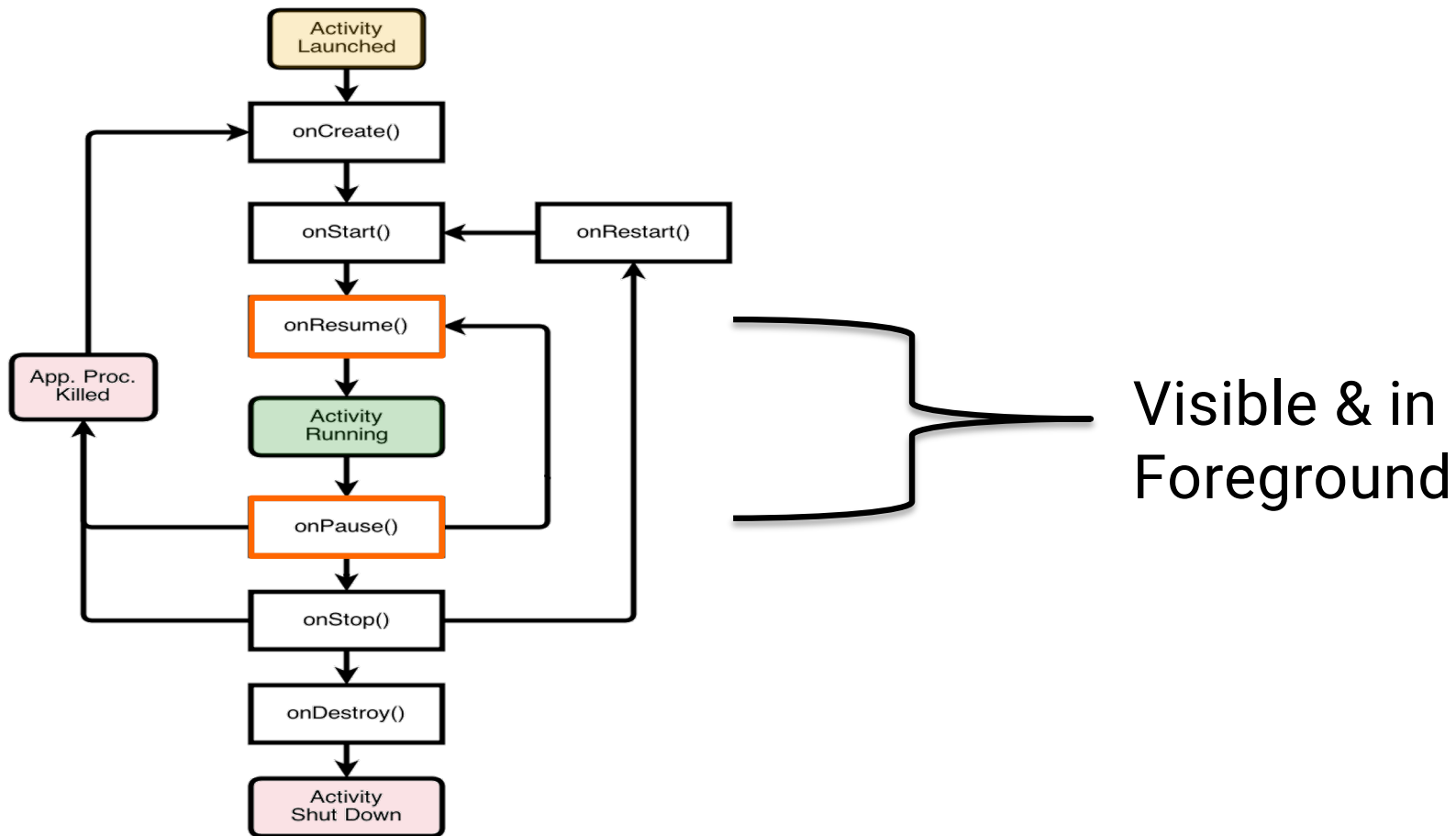


Entire
Lifetime

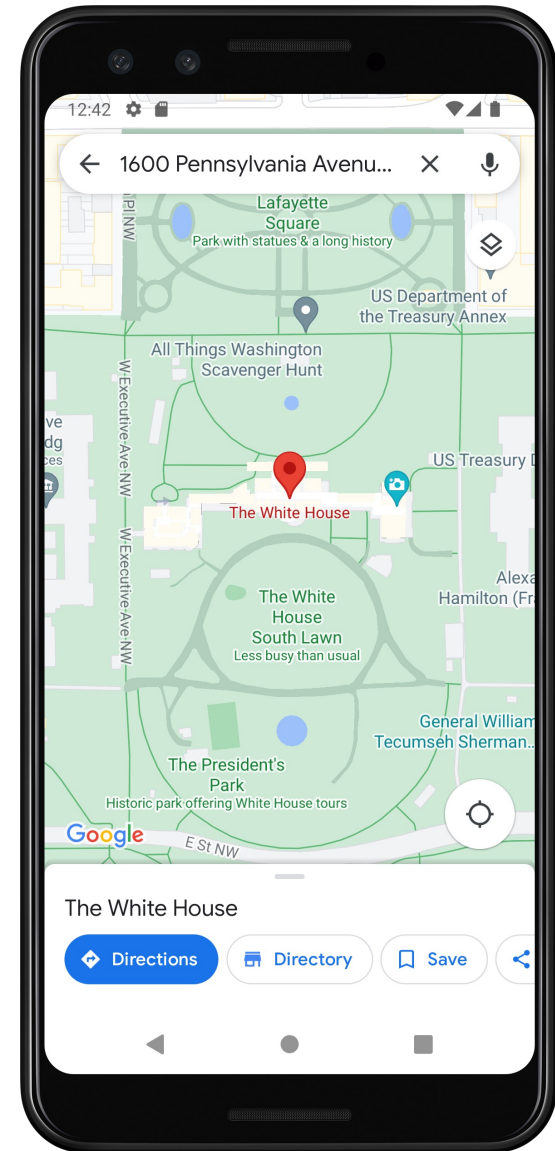
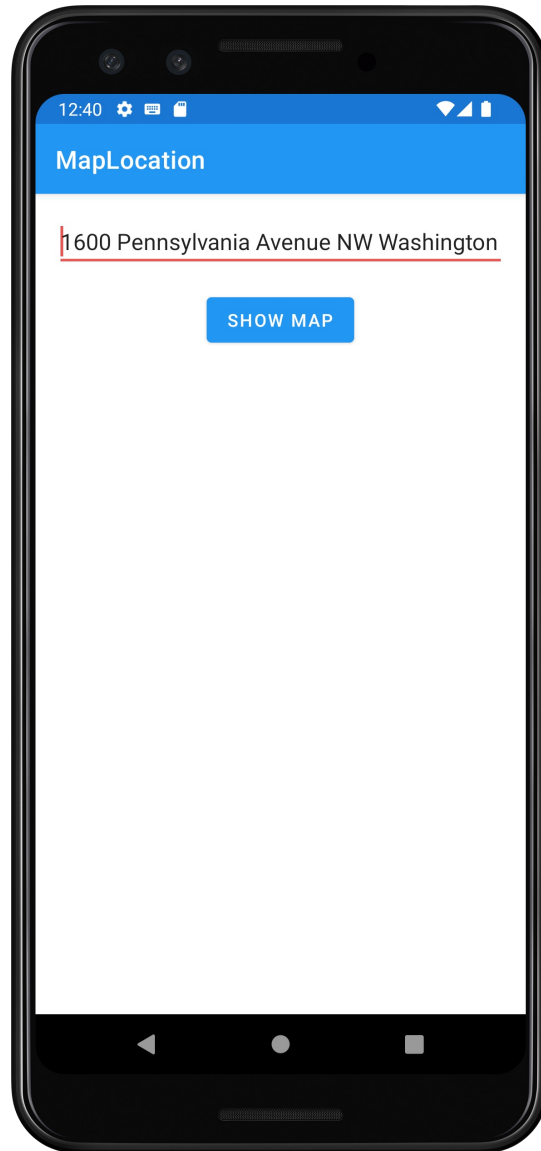
The Activity Lifecycle



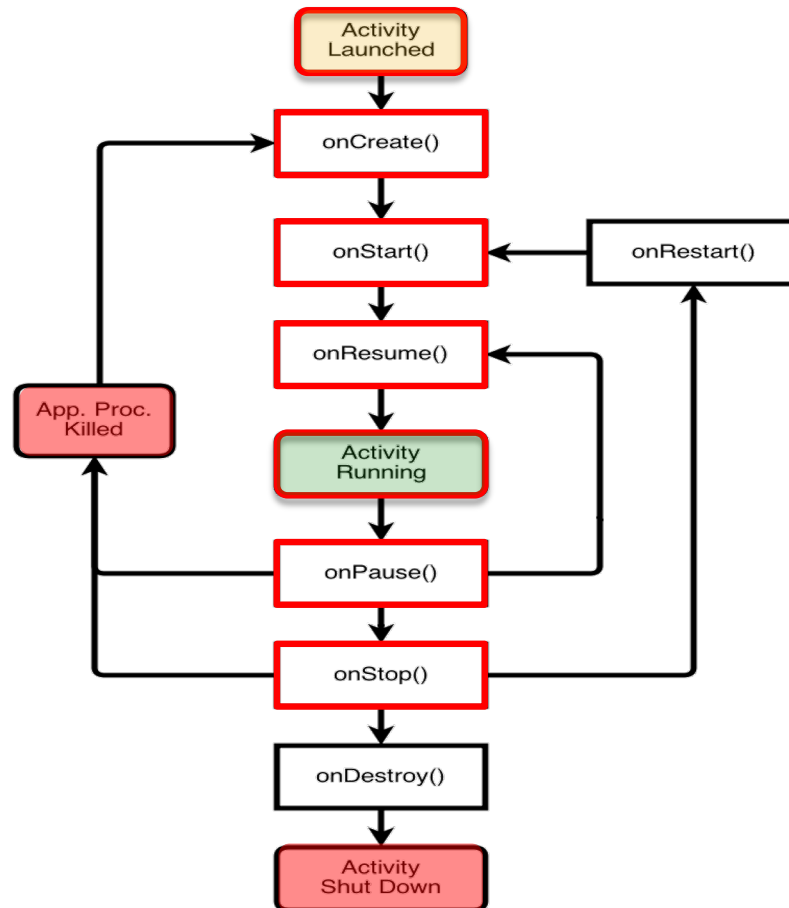
The Activity Lifecycle



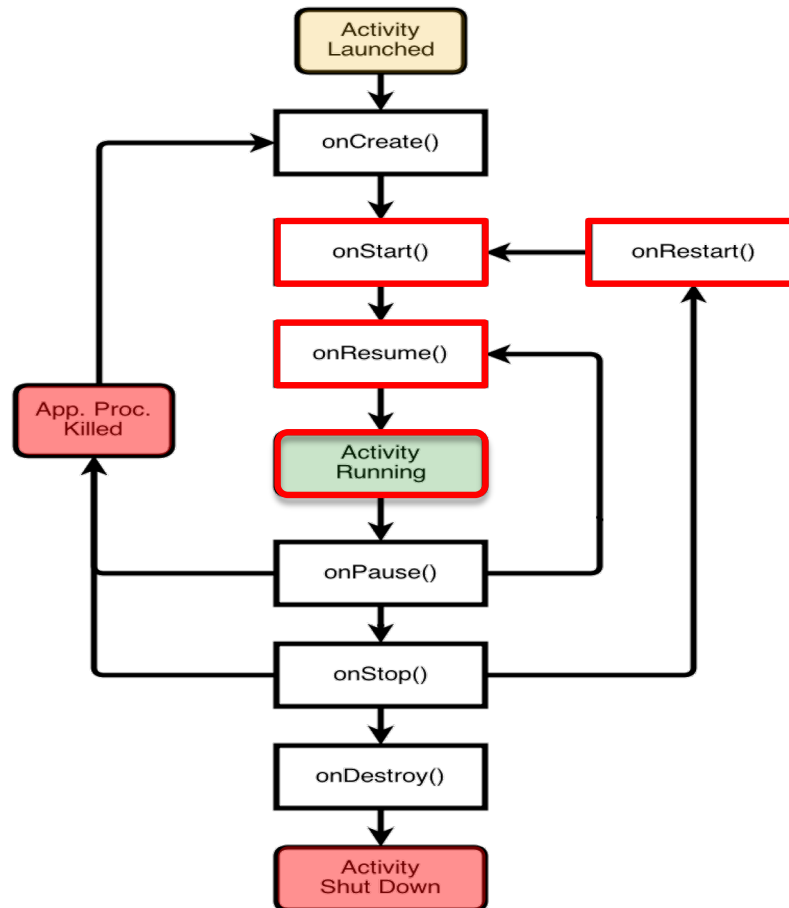
MapLocation



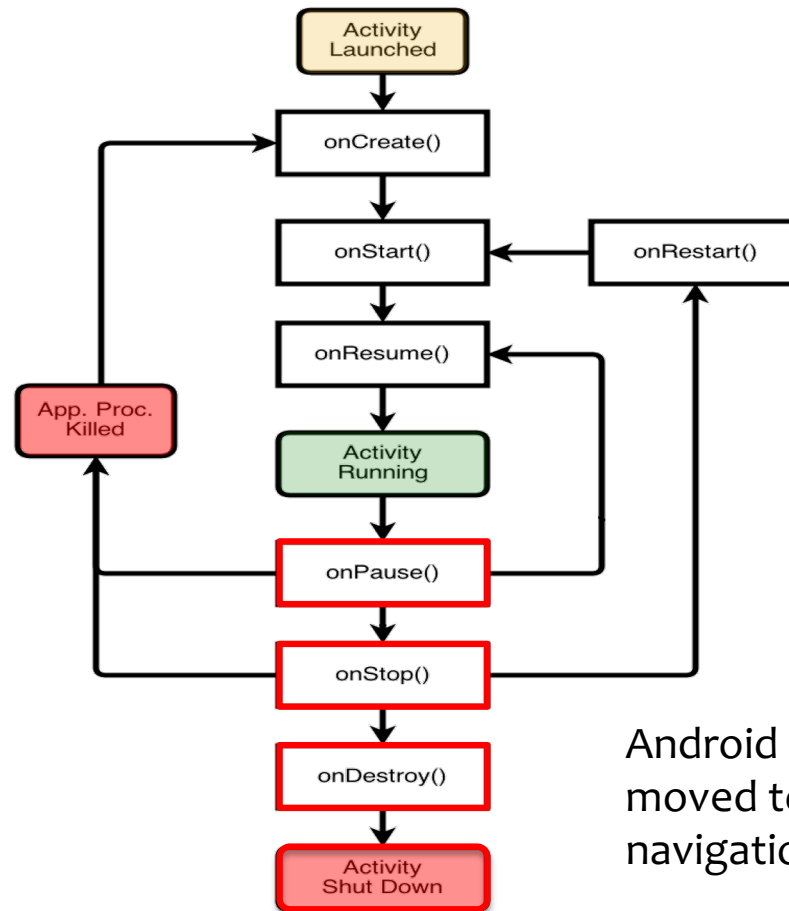
The Activity Lifecycle: MainActivity



The Activity Lifecycle: MainActivity



The Activity Lifecycle: MainActivity



Android 12+: Root launcher activities moved to background on back navigation

onCreate()

Called when Activity is created

Sets up initial state

- Call `super.onCreate()`

- Restore state if necessary

- Set the Activity's content view

- Retain references to UI views as necessary

- Configure views as necessary

onStart()

Activity is about to become visible

Typical actions

- Start visible-only behaviors

- Load persistent application state

onResume()

Activity is visible and about to start interacting with user

Typical actions

- Start foreground-only behaviors

onPause()

Focus about to switch to another Activity

Typical actions

- Shutdown foreground-only behaviors

- Save persistent state

onStop()

Activity is no longer visible to user

may be restarted later

Typical actions

Save persistent state

Do CPU-intensive save procedures

onRestart()

Called if the Activity has been stopped and is about to be started again

Typical actions

Special processing needed only after having been stopped

onDestroy()

Activity is about to be destroyed

Typical actions

- Release Activity-wide resources

Lifecycle Methods in MapLocation.kt

```
2022-09-12 10:47:02.018 12821-12821/course.examples.maplocation
I/MapLocation: Another activity is taking focus (this activity is about to
be "paused")
2022-09-12 10:47:04.145 12821-12821/course.examples.maplocation
I/MapLocation: The activity is no longer visible (it is now "stopped")
2022-09-12 10:47:19.454 12821-12821/course.examples.maplocation
I/MapLocation: The activity is visible and about to be restarted.
2022-09-12 10:47:19.454 12821-12821/course.examples.maplocation
I/MapLocation: The activity is visible and about to be started.
2022-09-12 10:47:19.455 12821-12821/course.examples.maplocation
I/MapLocation: The activity is visible and has focus (it is now "resumed")
```

Starting Activities

Create an Intent object matching the Activity to start

Pass Intent to methods, such as:

`Activity.startActivity()`

`ActivityResultCaller.registerForActivityResult()`

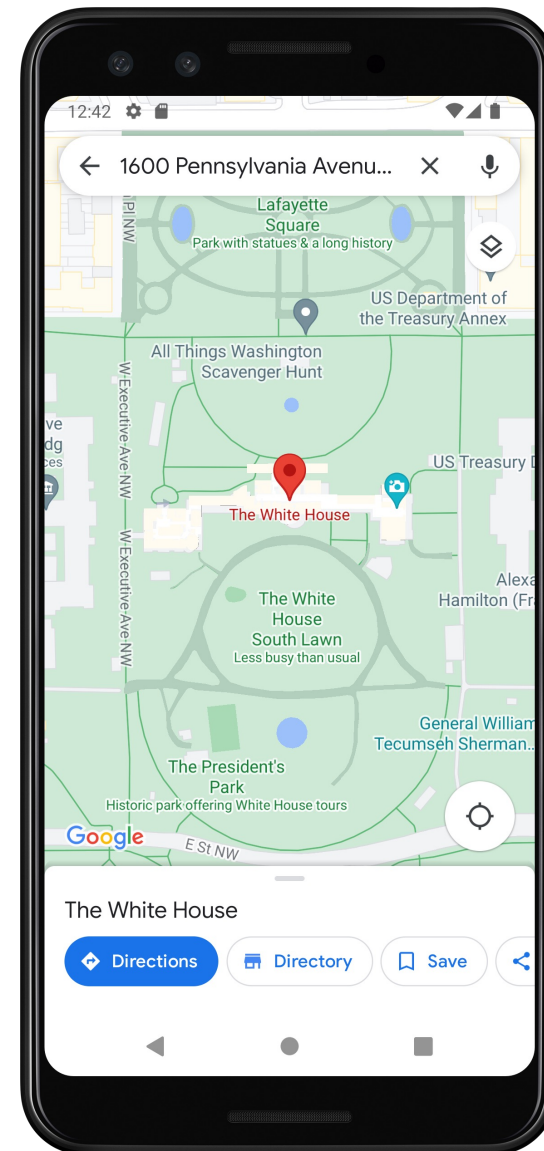
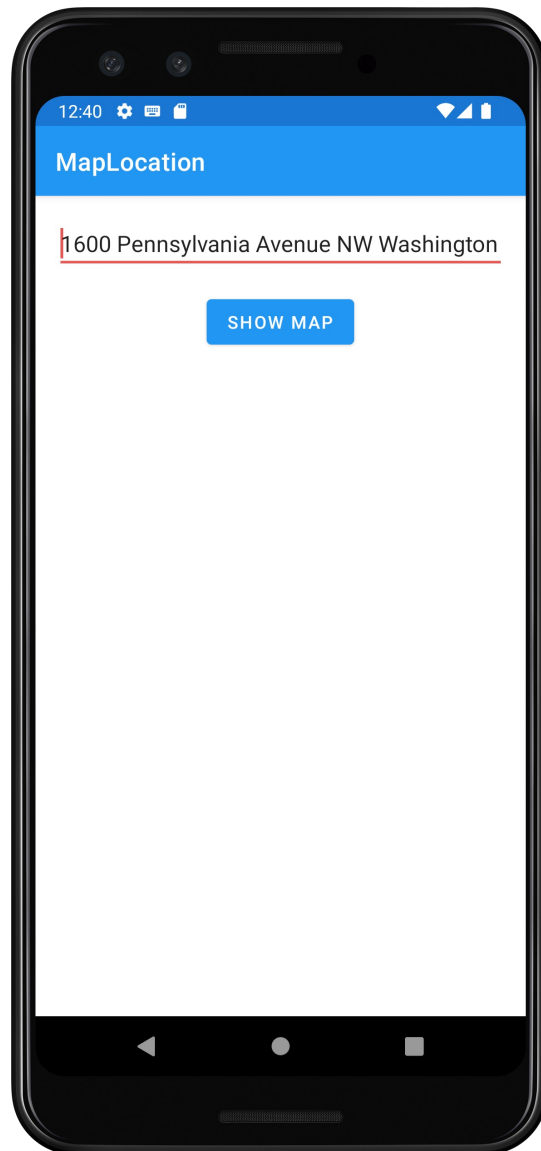
Activity.startActivity()

Create Intent

Check for presence of Intent handler

Call Activity.startActivity()

MapLocation

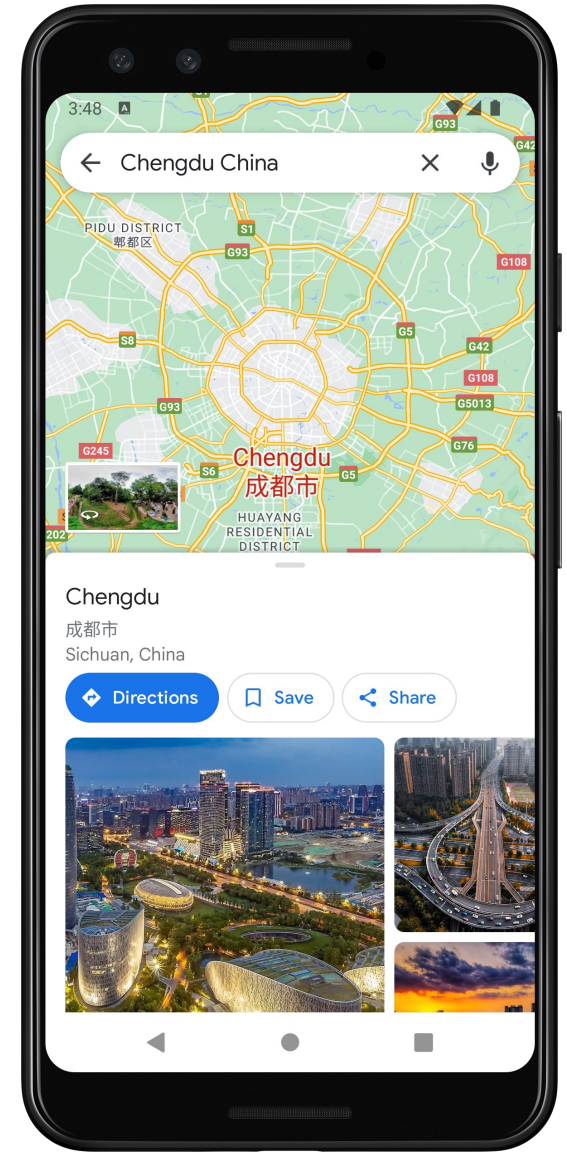
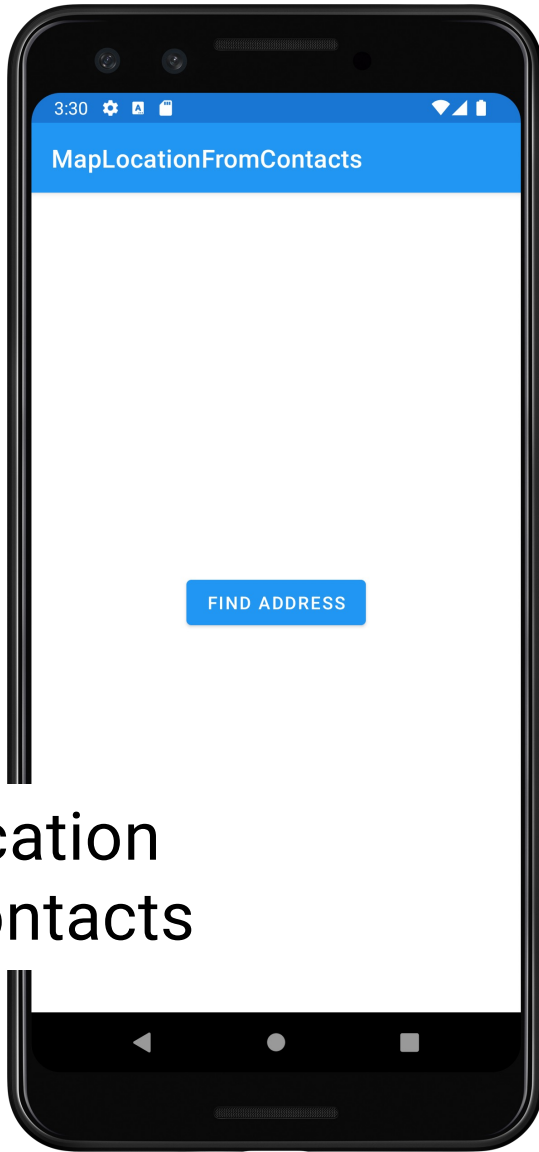


MapLocationFromContacts

Similar to MapLocation, but gets address from Contacts database

Make sure the Contacts app on your aVD has some contacts in it

MapLocation FromContacts



ActivityResultCaller.registerForActivityResult()

Example use case

Define `ActivityResultLauncher<Intent>` instance

This instance calls `registerForActivityResult()`, passing in necessary callback info

This info includes `ActivityResultContracts.StartActivityForResult()` contract interface instance

Call `ActivityResultLauncher<Intent>.launch(intent)` to start desired Activity

Registered callback is started when Activity returns

Configuration Changes

Keyboard, orientation, locale, etc.

Device configuration can change at runtime

On configuration changes, Android usually kills the current Activity & then restarts it

Configuration Changes

Activity restarting should be fast

Options

- Save Activity state in Bundle

- Use a separate Object (i.e., ViewModel)

- Manually handle the configuration change (not usually recommended)

Saving Activity State

Android automatically saves some information such as View state in a Bundle

You must save other state yourself

Modern Android best practices discourage this approach

Saving Activity State

Android calls `onSaveInstanceState(Bundle)`

after `onStop()` for API 28+

before `onStop()` for API <28

Save Activity instance state to system-provided
Bundle

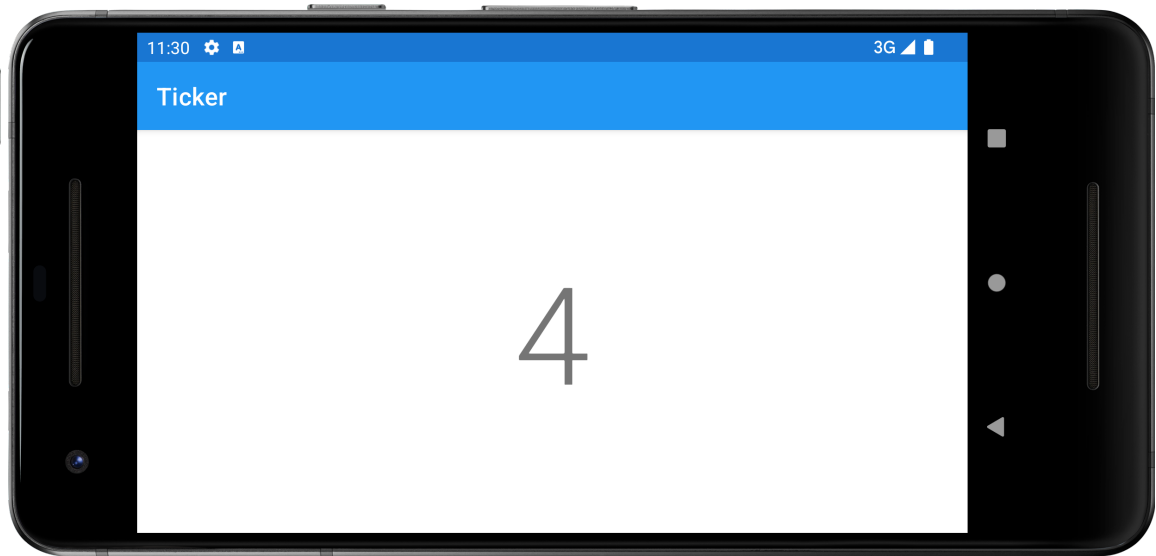
Saving Activity State

When Activity is restarted, you can restore Activity state from a system-provided Bundle in:

- `onCreate(Bundle)`

- `onRestoreInstanceState(Bundle)`, which is called between `onStart()` and `onPostCreate()`

Ticker



* Turn auto-rotate on in your AVD Display settings

Retaining an Object

Hard to recompute data can be cached to speed up handling of configuration changes

Current recommendation uses ViewModel class

We'll come back to this in a later lesson

Manual Reconfiguration

Can prevent system from restarting Activity

Declare the configuration changes your Activity handles in AndroidManifest.xml file, e.g.,

```
<activity android:name=".MyActivity"  
    android:configChanges=  
        "orientation|screenSize|keyboardHidden"...>
```

Manual Reconfiguration

When configuration changes, Activity' s
onConfigurationChanged() method is called

Passed a Configuration object specifying the new
device configuration

Manual Reconfiguration Caveat

Should generally avoid manual approach

- Hard to get right

- Fragile to system changes

Next

The Intent Class

Example Applications

MapLocation

MapLocationFromContacts

Ticker