

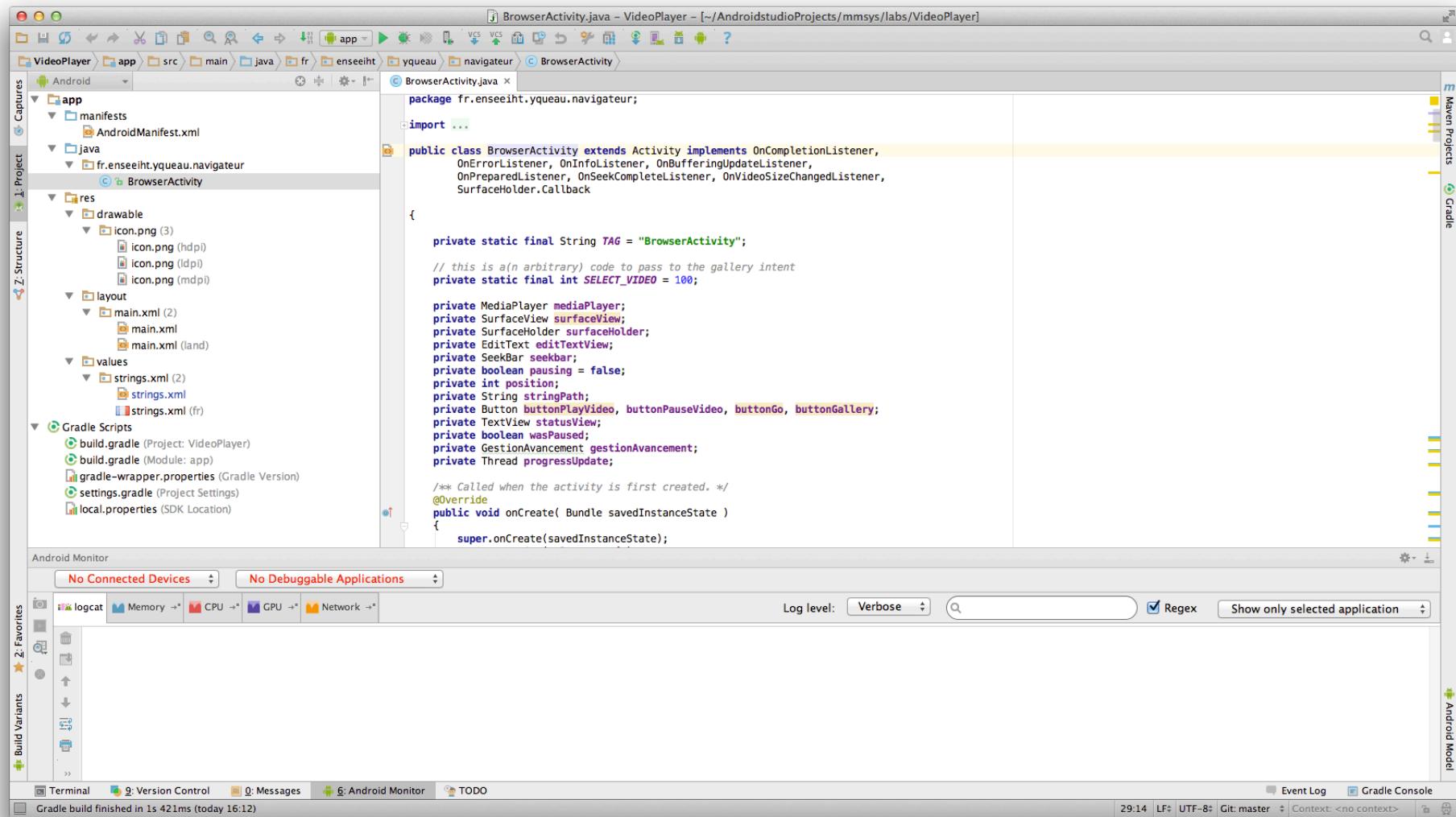
Plan

- Mobile devices
- Android overview
 - History
 - Architecture
 - Applications
- **The tools**
- Android Applications

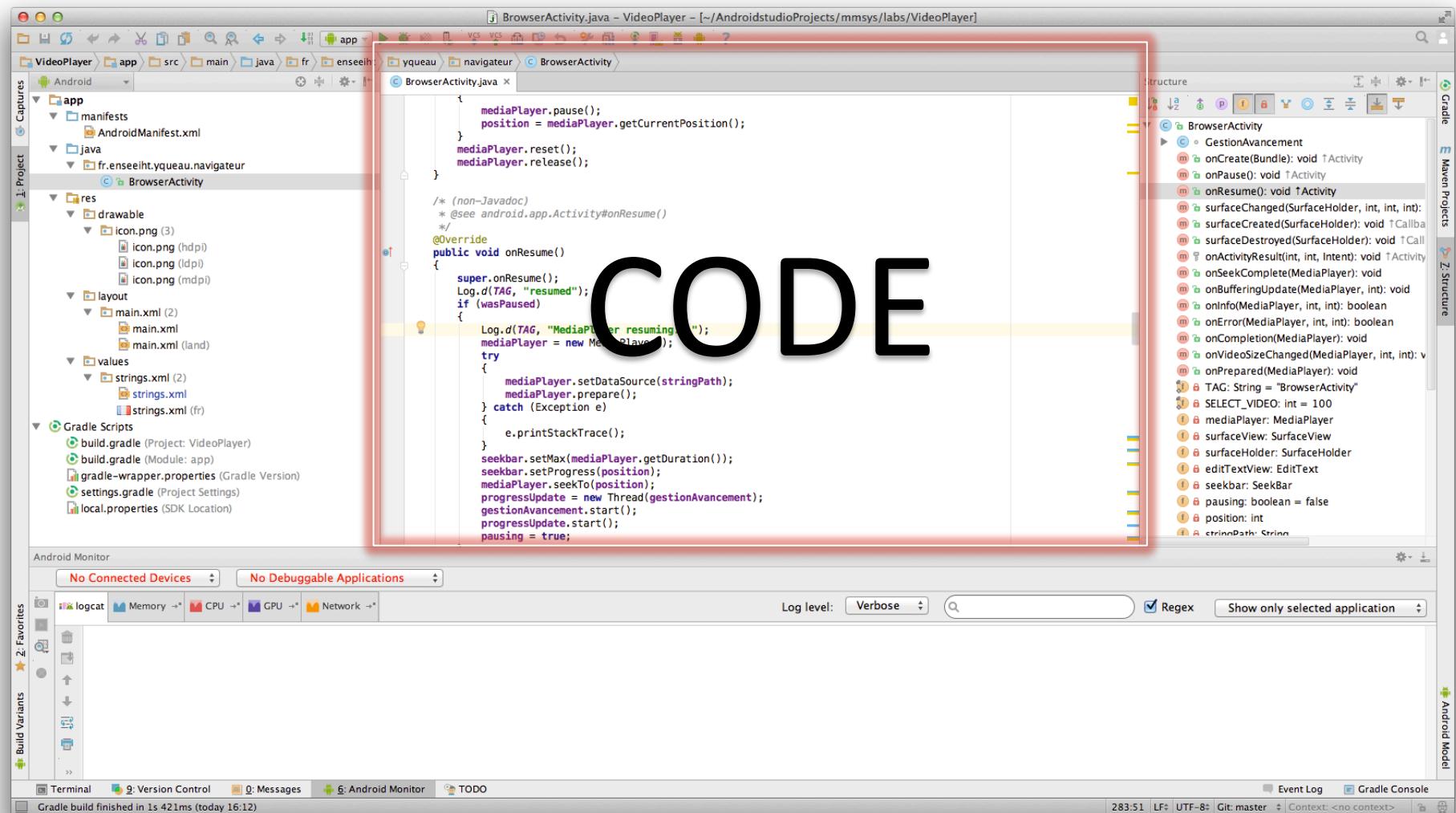
Development utilities

- **Android Studio**
 - Based on IntelliJ IDEA IDE for Java
 - Wizard for creating a new project
 - GUI editor (WYSIWYG)
 - Automated compilation process
 - Logging, debugging, code profiling ...
- **Android SDK**
 - API
 - Device emulators
 - Utilities for debugging, signing the code etc..
 - Sample code and Demos

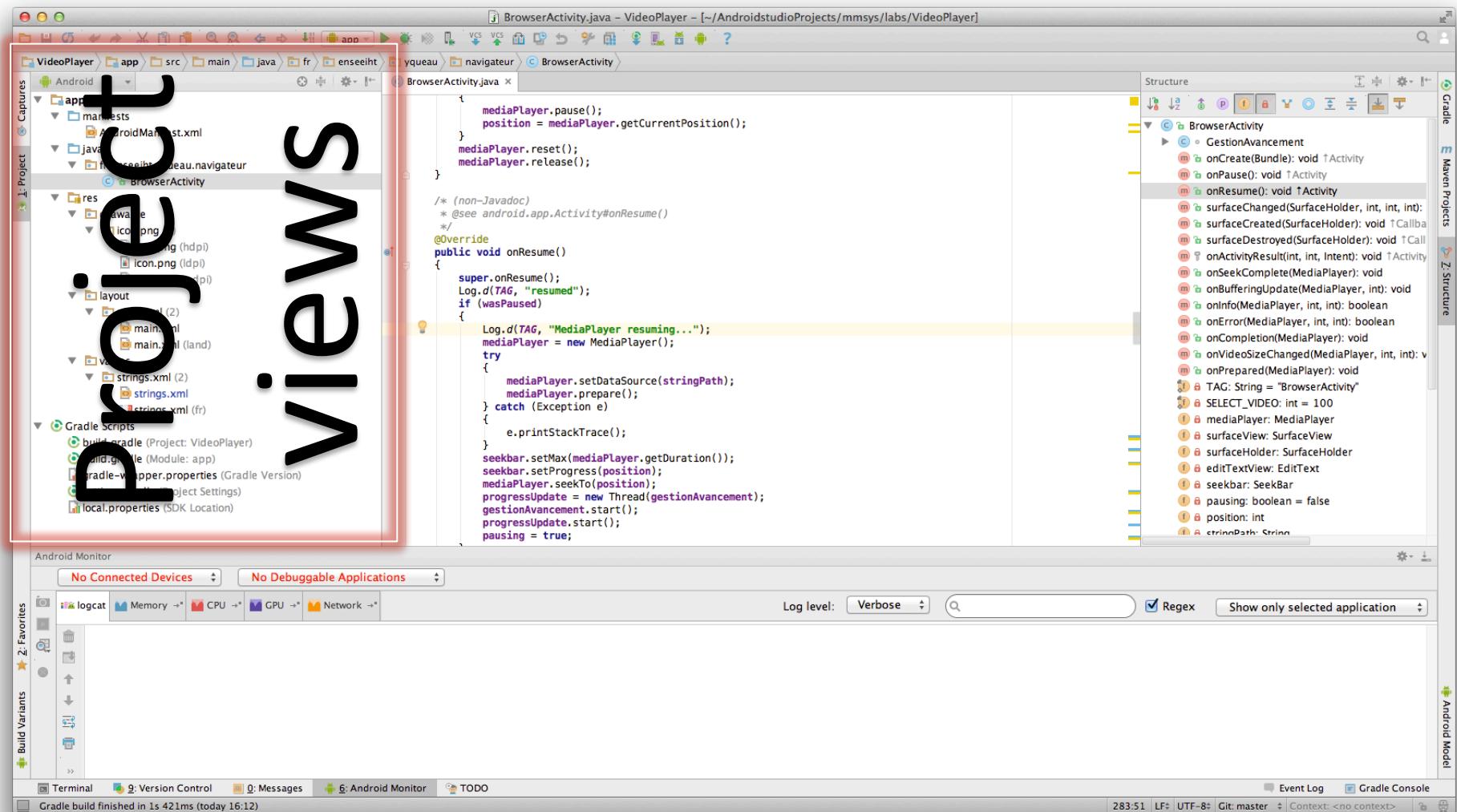
Android Studio



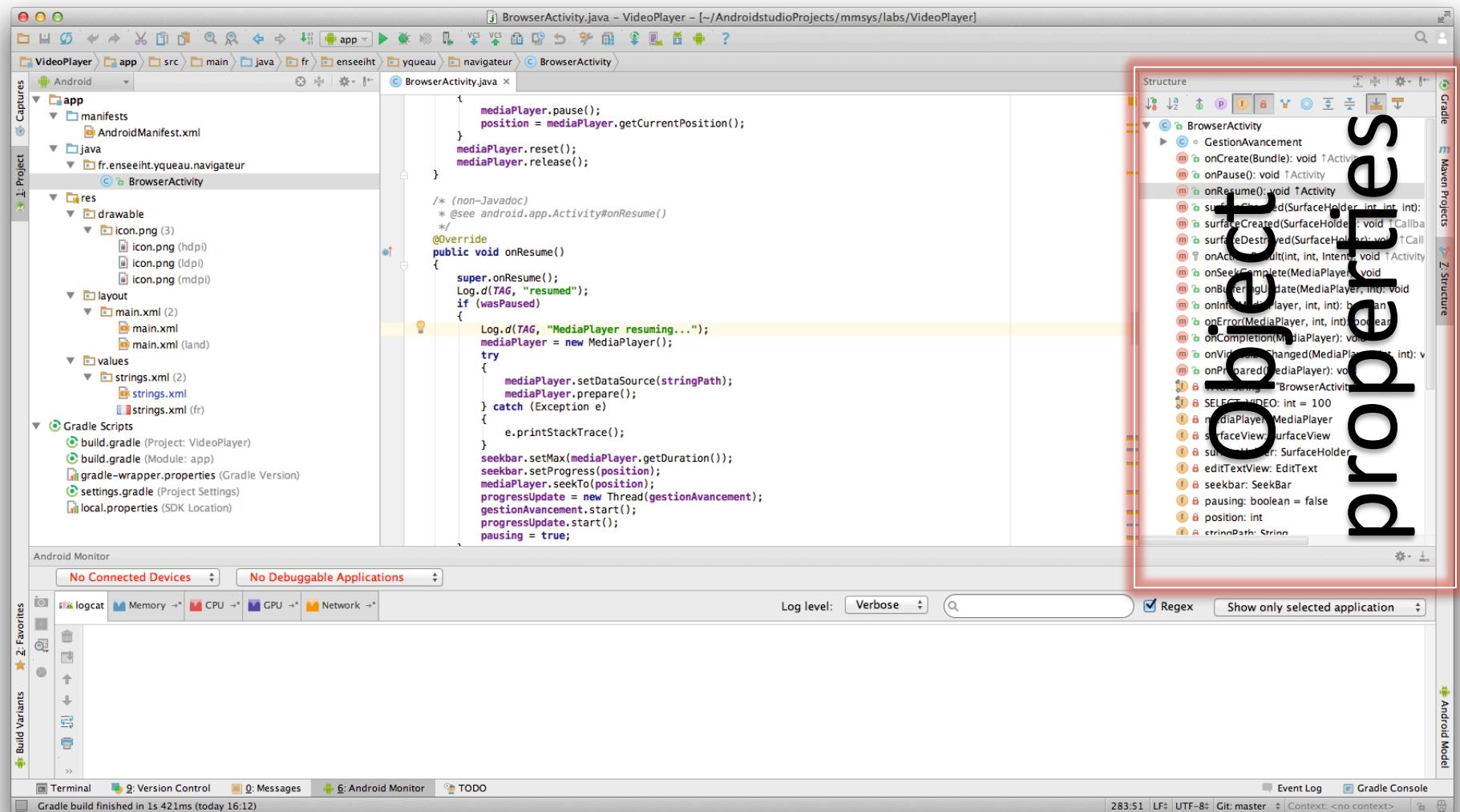
Android Studio



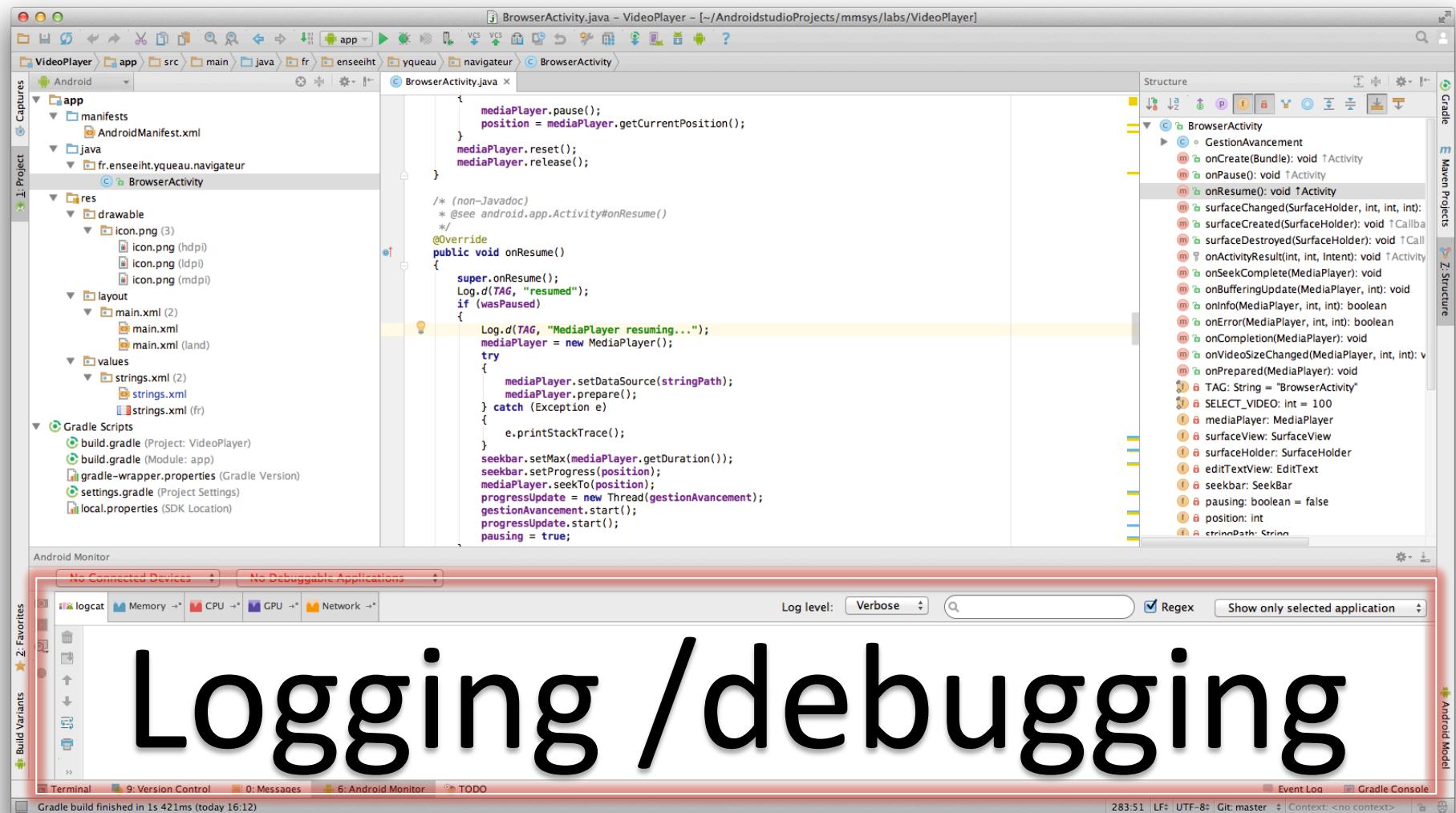
Android Studio



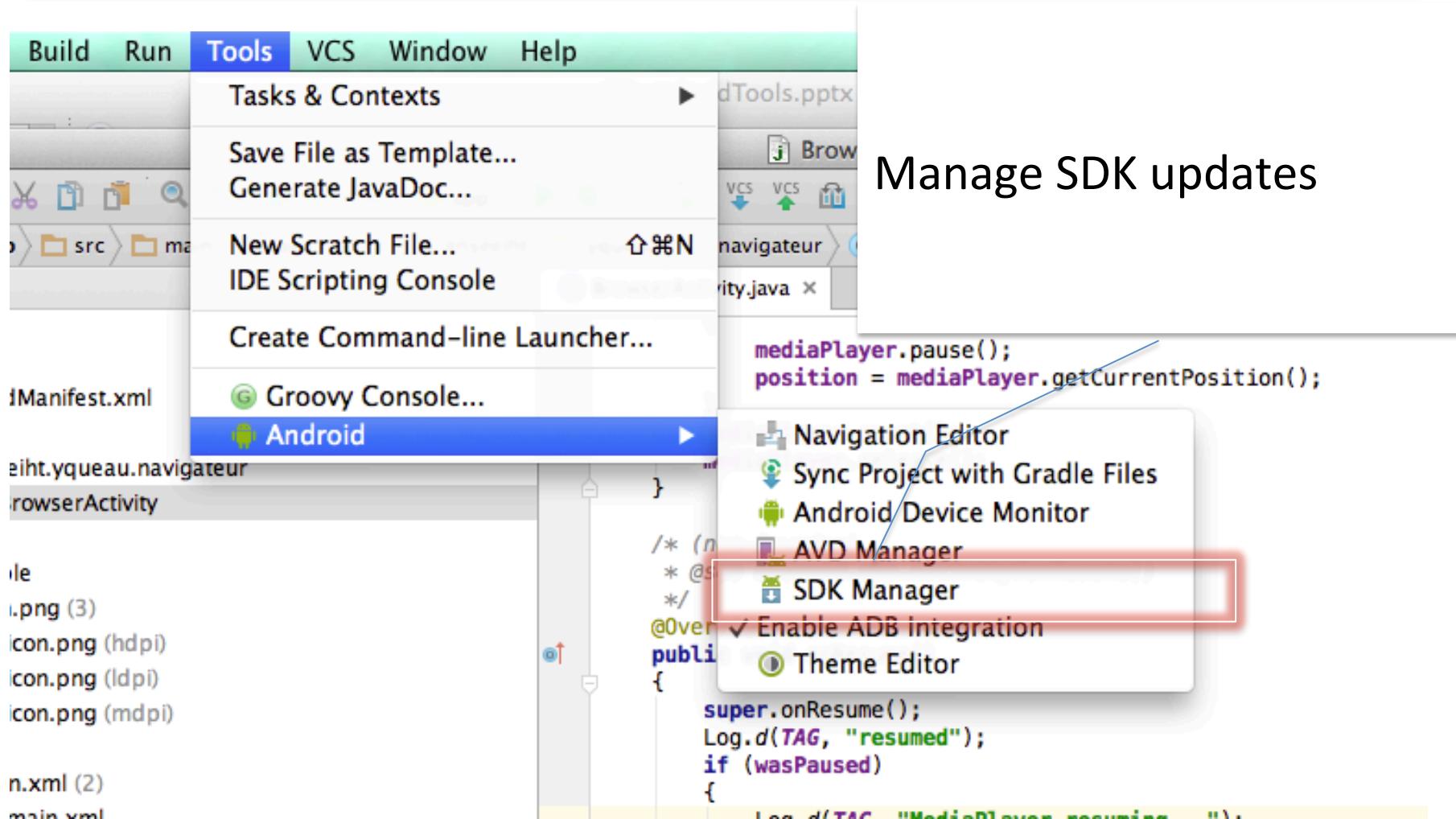
Android Studio



Android Studio

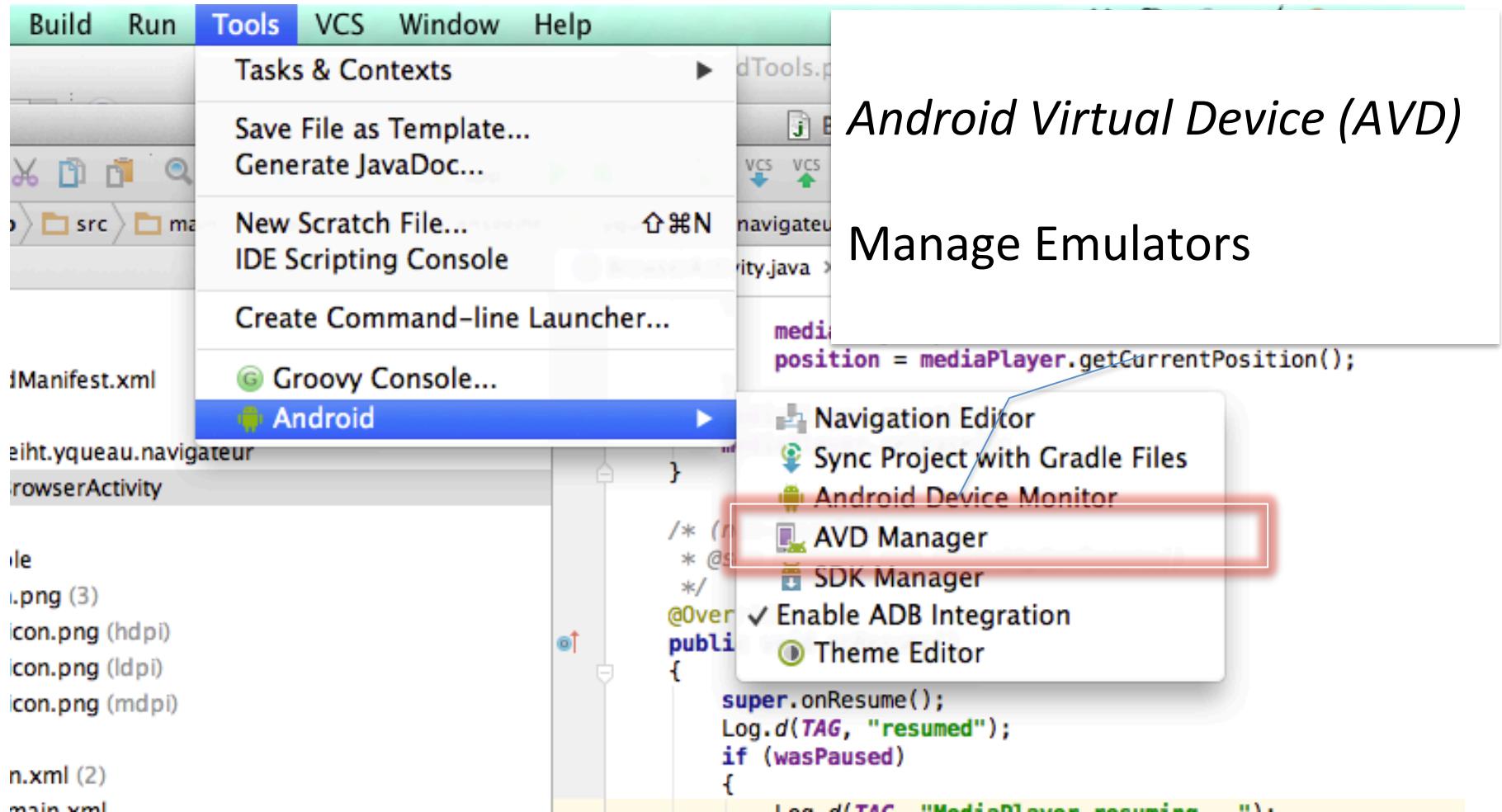


Android Studio

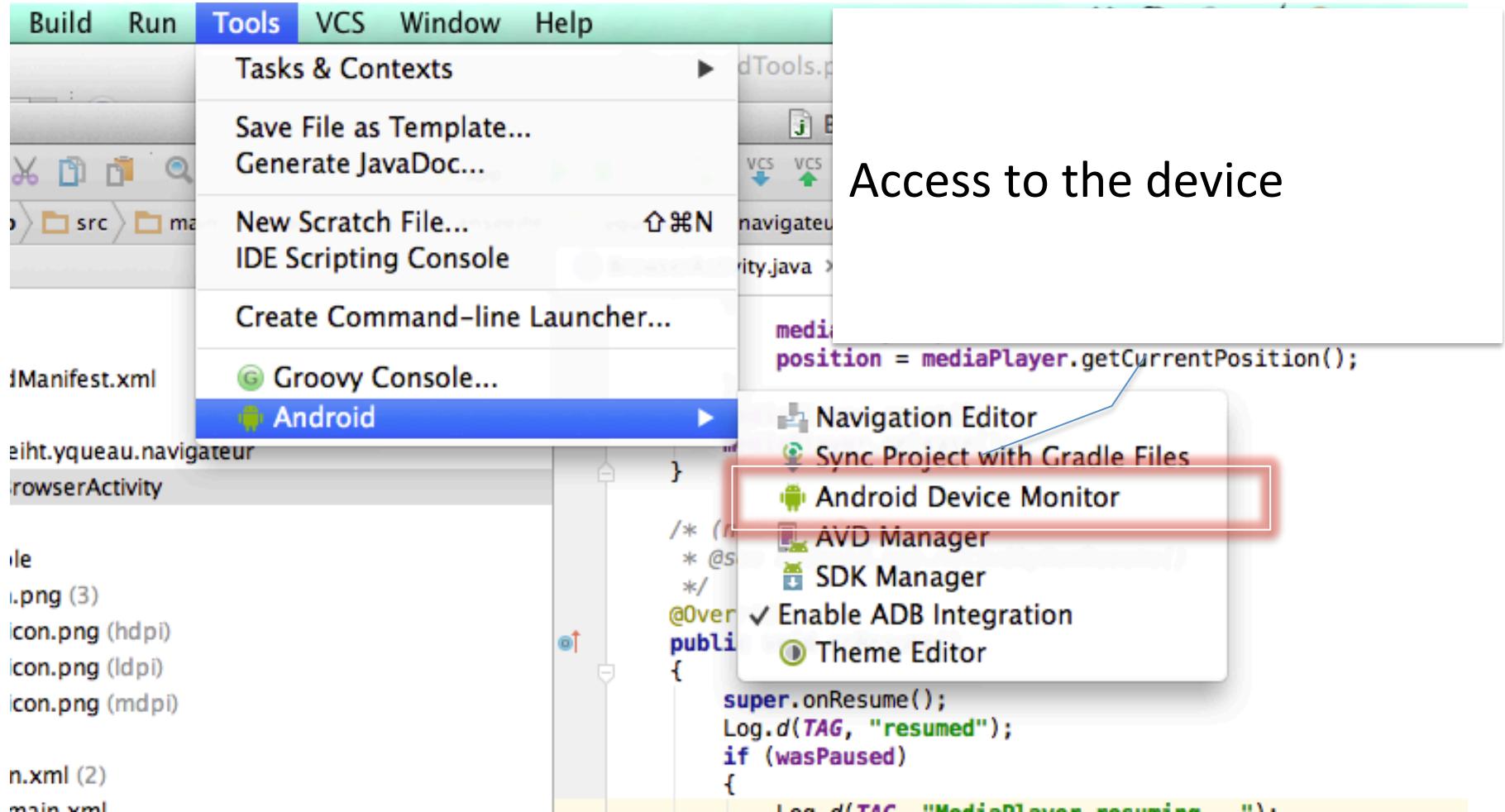


Manage SDK updates

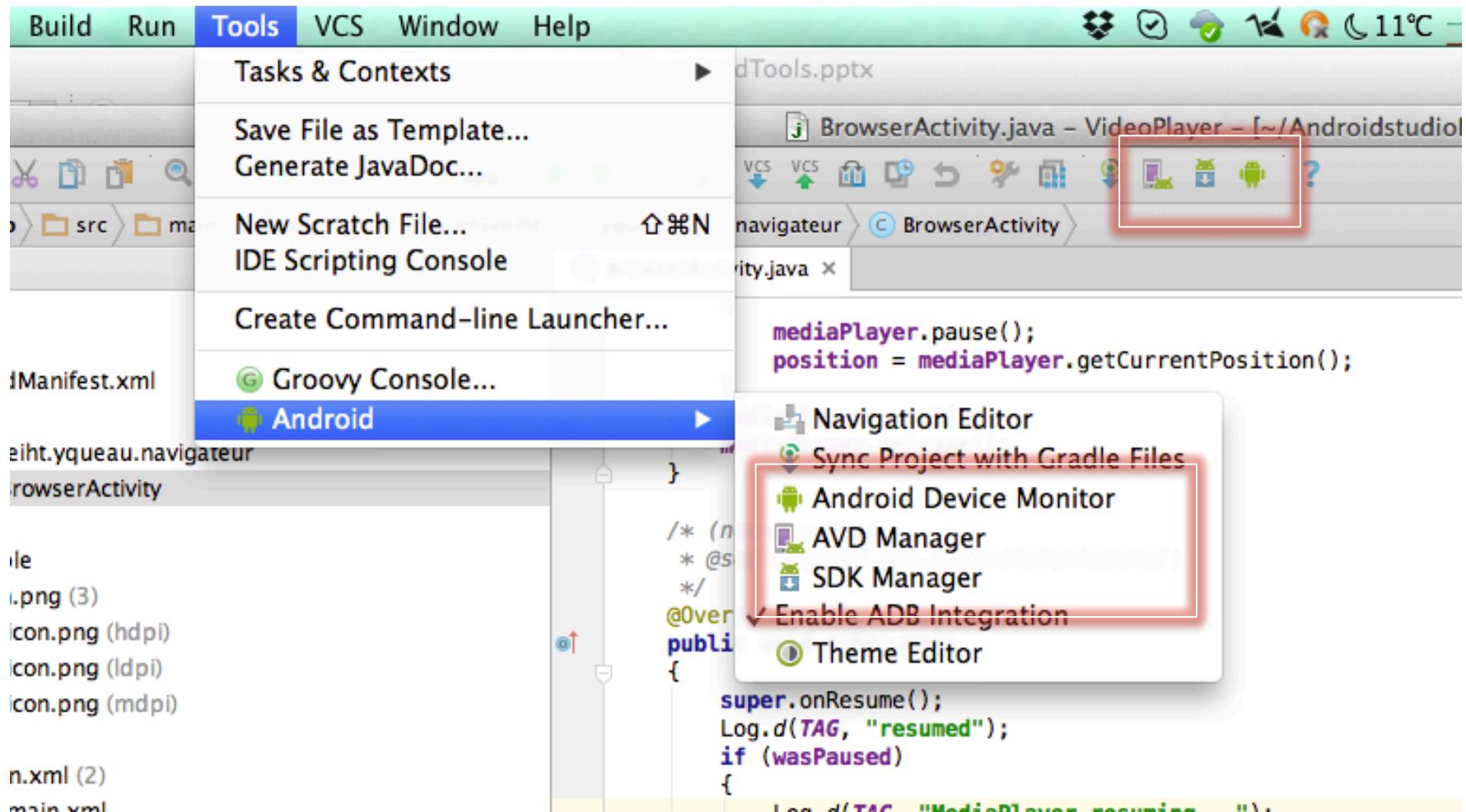
Android Studio



Android Studio



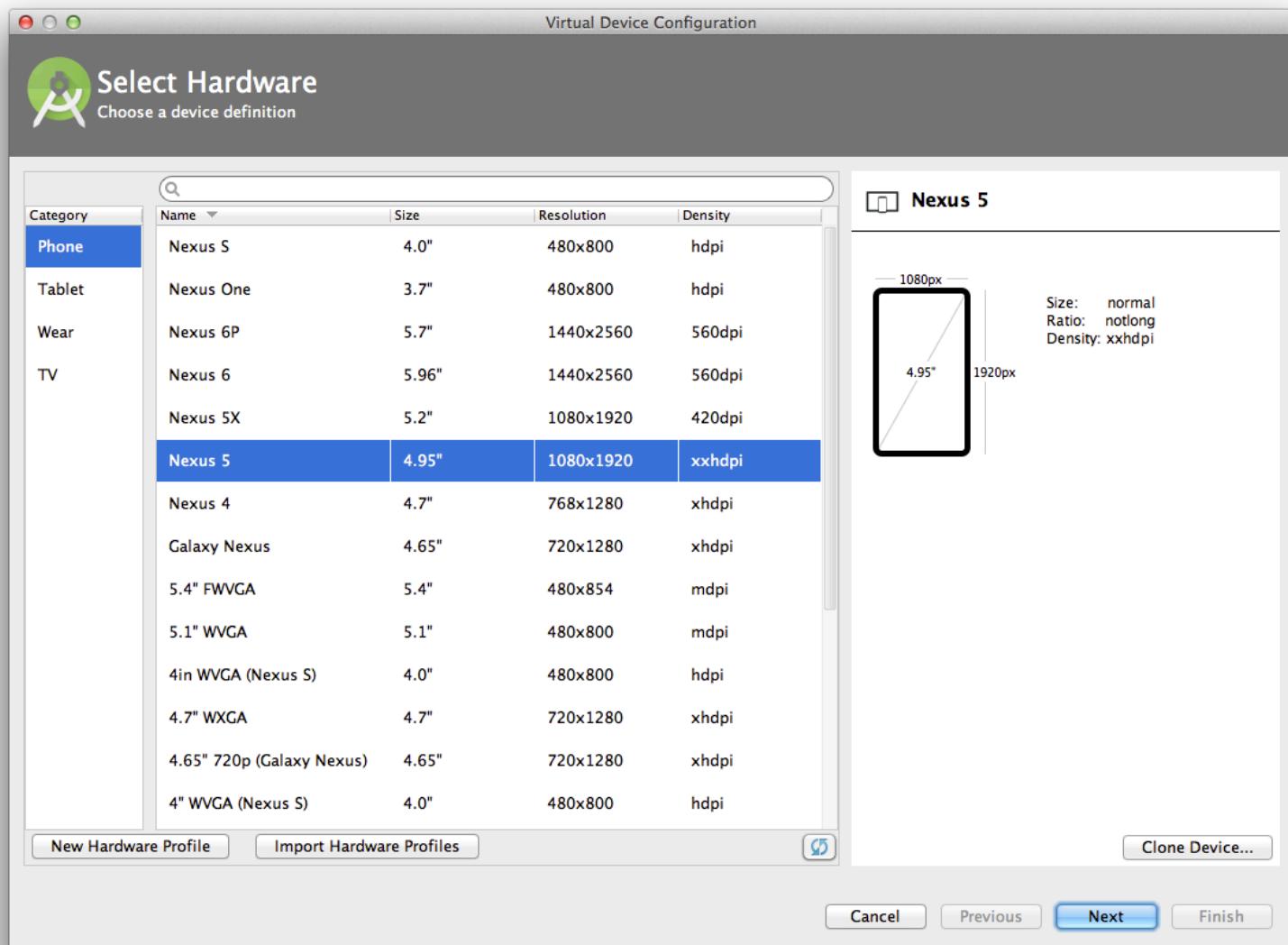
Android Studio



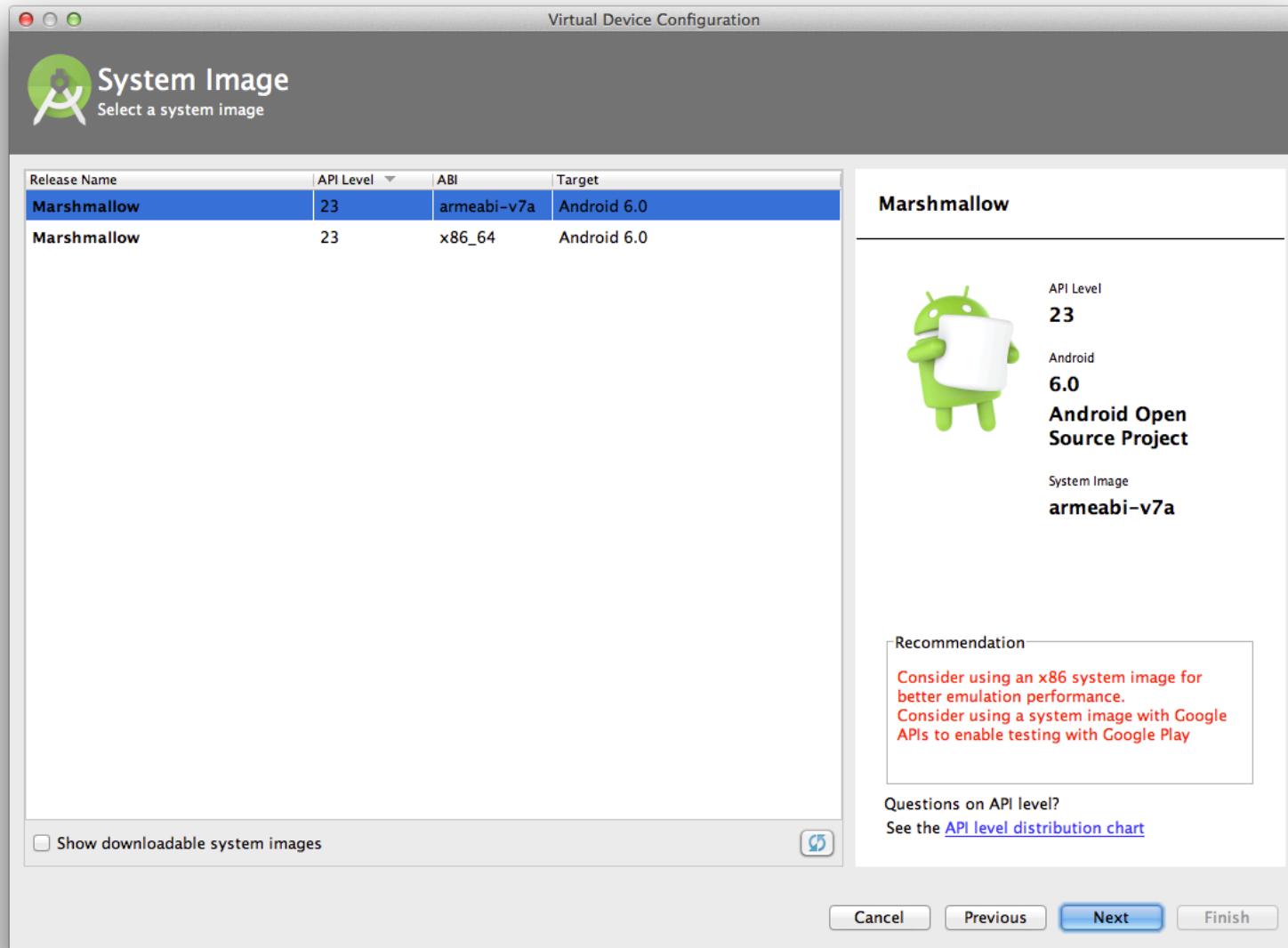
Android Studio – AVD Manager



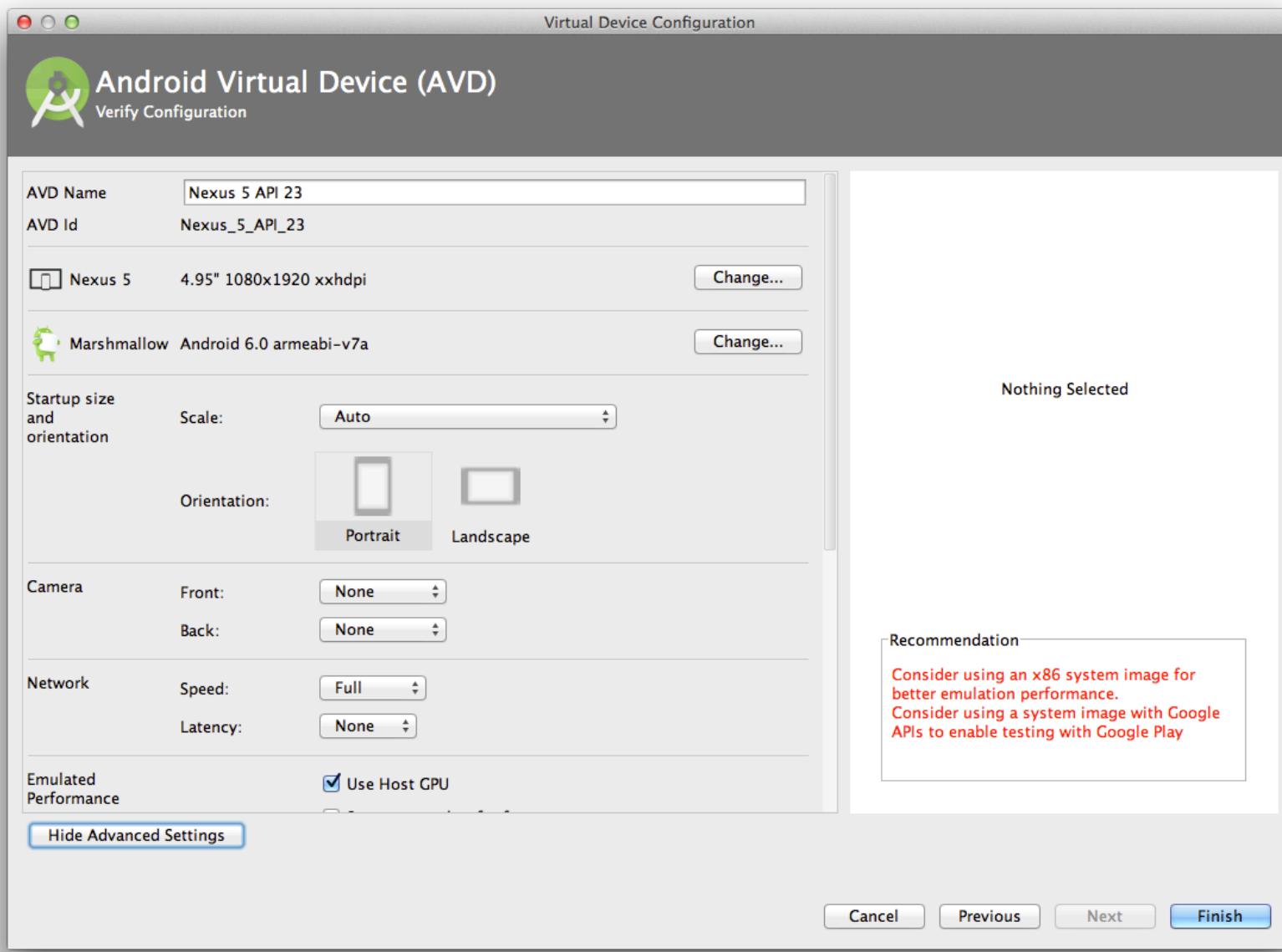
Android Studio – AVD



Android Studio – AVD



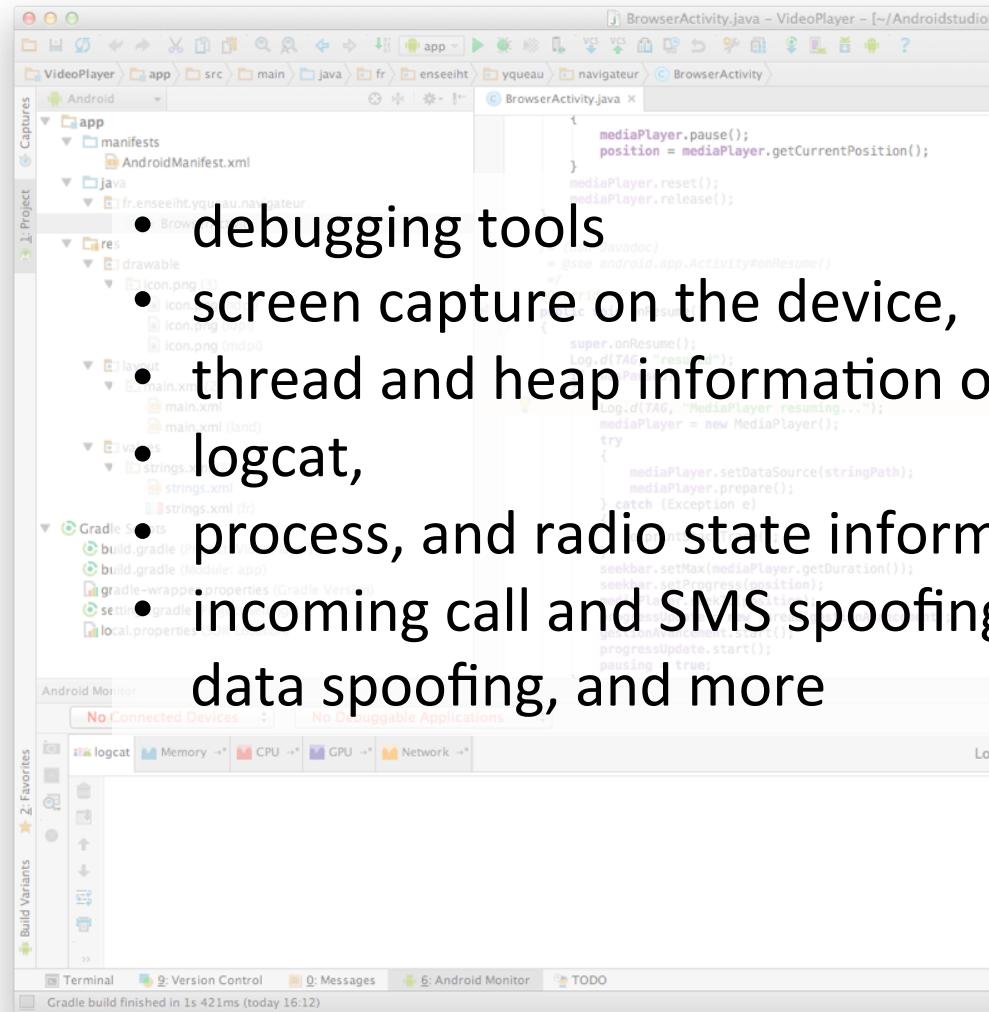
Android Studio – AVD



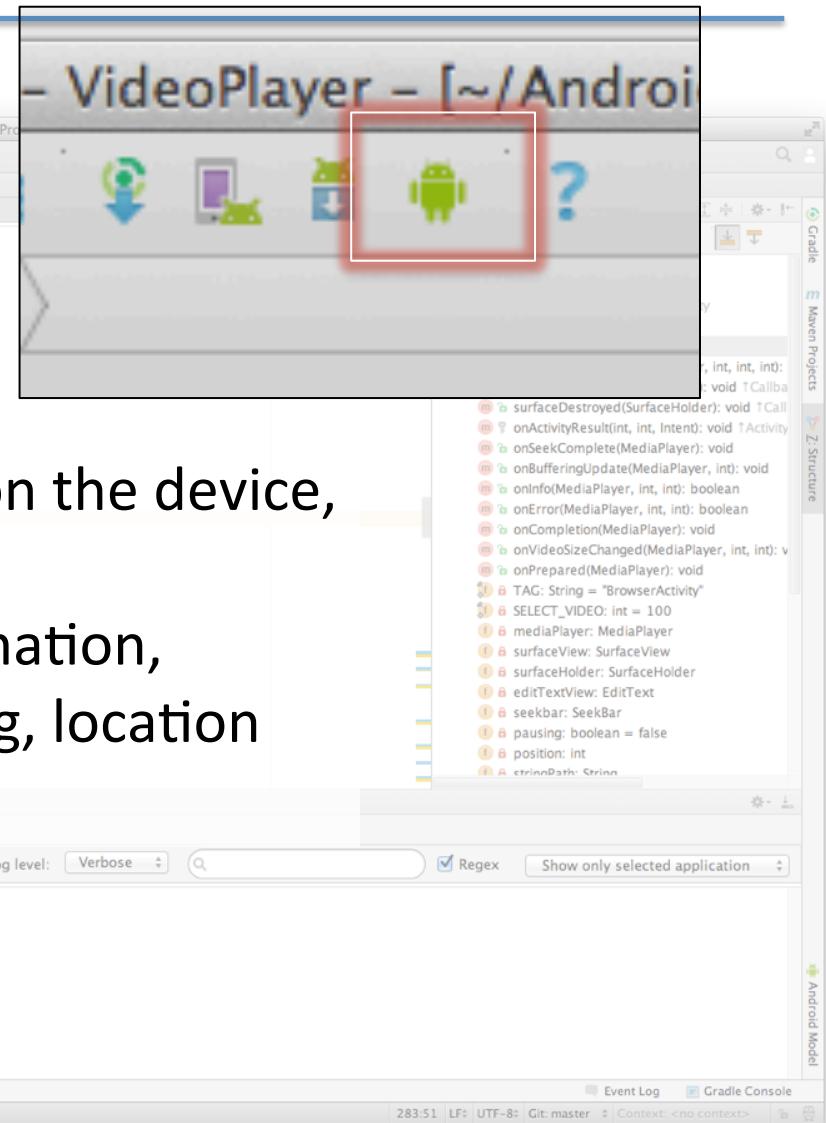
Android Studio – AVD



Android Device Monitor



- debugging tools
- screen capture on the device,
- thread and heap information on the device,
- logcat,
- process, and radio state information,
- incoming call and SMS spoofing, location data spoofing, and more



Android Device Monitor

The screenshot shows the Android Device Monitor interface. The top navigation bar includes 'Devices' (selected), 'Threads', 'Heap', 'Allocation Tracker', 'Network Statistics', 'File Explorer' (selected), 'Emulator Control', and 'System Information'. The main area has three tabs: 'File Explorer', 'LogCat' (selected), and 'Console'.

File Explorer Tab: Displays a file tree and a detailed list of files under the root directory. The list includes:

Name	Size	Date	Time	Permissions	Info
acct		2016-01-27	18:53	drwxr-xr-x	
cache		2016-01-15	14:41	drwxrwx---	
charger		1970-01-01	01:00	lrwxrwxrwx	-> /sbin/...
config		2016-01-27	18:53	dr-x-----	
d		2016-01-27	18:53	lrwxrwxrwx	-> /sys/k...
data		2016-01-14	16:18	drwxrwx--x	
default....	551	1970-01-01	01:00	-rw-r--r--	
dev		2016-01-27	18:53	drwxr-xr-x	
etc		2016-01-27	18:53	lrwxrwxrwx	-> /syste...
file_cont...	14591	1970-01-01	01:00	-rw-r--r--	
fstab.go...	943	1970-01-01	01:00	-rw-r-----	
fstab.ra...	962	1970-01-01	01:00	-rw-r----	
init	1397...	1970-01-01	01:00	-rwxr-x---	
init.envi...	852	1970-01-01	01:00	-rwxr-x---	
init.gold...	2465	1970-01-01	01:00	-rwxr-x---	
init.ranc...	1909	1970-01-01	01:00	-rwxr-x---	
init.rc	25026	1970-01-01	01:00	-rwxr-x---	
init.trace.rc	1921	1970-01-01	01:00	-rwxr-x---	
init.usb.rc	3885	1970-01-01	01:00	-rwxr-x---	
init.zygo...	301	1970-01-01	01:00	-rwxr-x---	
init.zygo...	531	1970-01-01	01:00	-rwxr-x---	

LogCat Tab: Shows log messages. The search bar says "Search for messages. Accepts Java regexes. Prefix with pid:, app:, tag: or text: to limit scope." The message list shows:

Level	Time	PID	TID	Application	Tag	Text
I	01-27 18:55:22.430	1942	1948	com.android.keych...	art	Debugger is no longer active
I	01-27 18:55:22.430	1962	1968	com.android.dialer	art	Debugger is no longer active
I	01-27 18:55:22.430	1979	1985	com.android.prov...	art	Debugger is no longer active
I	01-27 18:55:22.430	2011	2017	com.android.manag...	art	Debugger is no longer active
I	01-27 18:55:22.430	2024	2030	com.android.launc...	art	Debugger is no longer active
I	01-27 18:55:22.430	2046	2052	com.android.calen...	art	Debugger is no longer active
I	01-27 18:55:22.430	2071	2078	com.android.email	art	Debugger is no longer active
I	01-27 18:55:22.430	2086	2092	com.android.excha...	art	Debugger is no longer active
I	01-27 18:55:22.430	2105	2111	com.android.messa...	art	Debugger is no longer active

Bottom status bar: 100M of 492M

LogCat

The screenshot shows the Android LogCat interface. At the top, there are tabs for 'LogCat' (selected) and 'Console'. Below the tabs is a search bar with placeholder text: 'Search for messages. Accepts Java regexes. Prefix with pid:, app:, tag: or text: to limit scope.' To the right of the search bar are buttons for 'verbose', a filter icon, and a download icon.

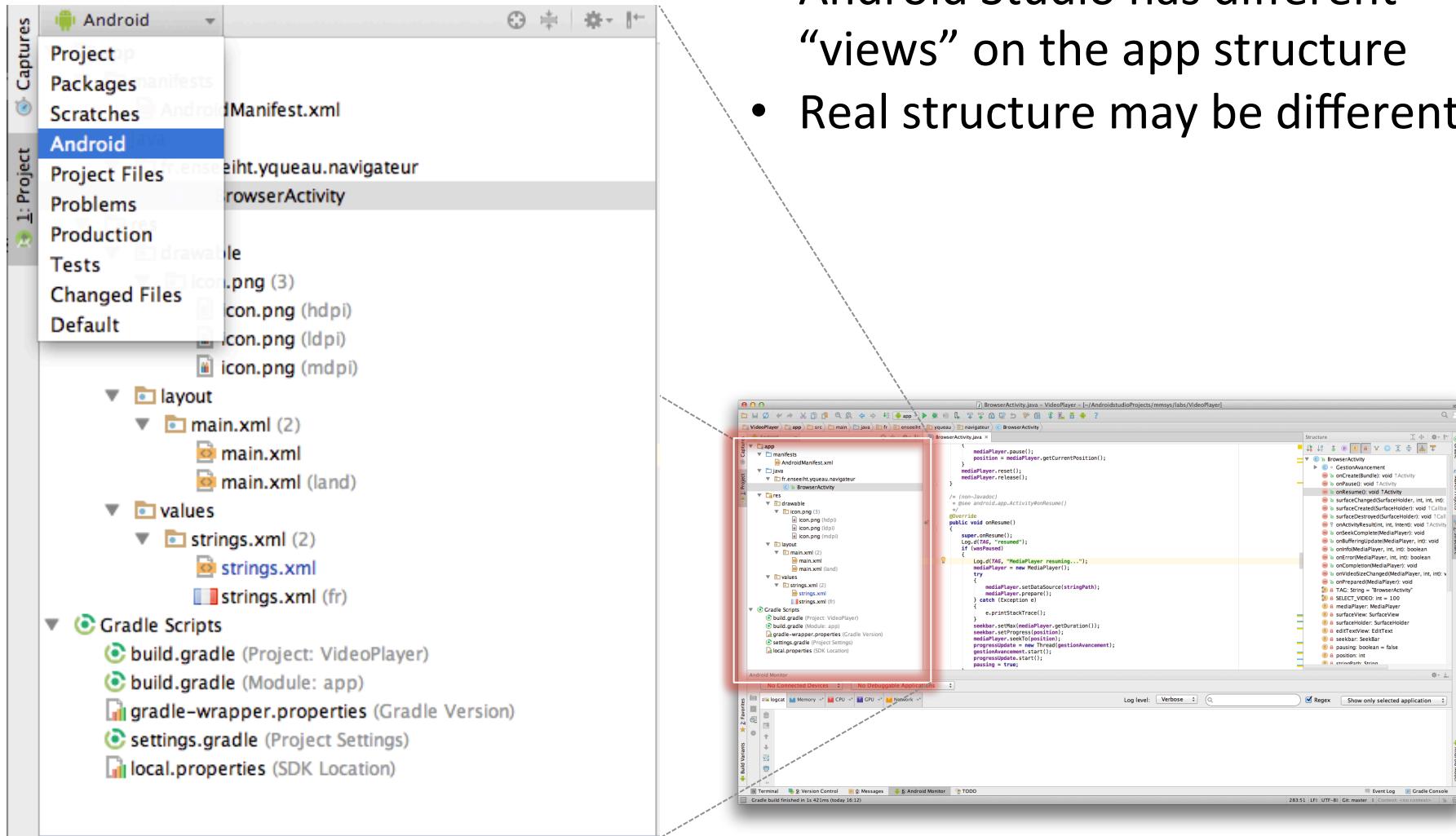
The main area displays a table of log messages. The columns are: Level (Lev), Time, PID, TID, Application, Tag, and Text. The 'Text' column contains the actual log entries. The log entries show various system and application logs, including messages from com.android.calendaring.AlertService, com.android.systemui.art, and com.android.phone.art. The log level is primarily 'I' (Info).

Lev	Time	PID	TID	Application	Tag	Text
I	01-27 18:54:50.520	2046	2159	com.android.calen...	GlobalDismiss...	no sender configured 87397Z0
D	01-27 18:54:50.520	2046	2159	com.android.calen...	AlertService	Beginning updateAlertNotification
D	01-27 18:54:50.520	2046	2159	com.android.calen...	AlertService	No fired or scheduled alerts
D	01-27 18:54:50.520	2046	2159	com.android.calen...	AlertService	Scheduling next alarm with AlarmScheduler. sEventReminderReceive null
D	01-27 18:54:50.530	2046	2159	com.android.calen...	AlarmScheduler	No events found starting within 1 week.
D	01-27 18:54:54.620	2046	2070	com.android.calen...	InitAlarmsSe...	Clearing and rescheduling alarms.
I	01-27 18:55:12.390	1355	1366	system_process	MediaFocusCo...	AudioFocus abandonAudioFocus() from android.media.AudioManager 03eafcom.android.music.MediaPlaybackService\$3@5de32bc
I	01-27 18:55:22.430	1459	1465	com.android.system...	art	Debugger is no longer active
I	01-27 18:55:22.430	1656	1662	com.android.input...	art	Debugger is no longer active
I	01-27 18:55:22.430	1714	1726	com.android.print...	art	Debugger is no longer active
I	01-27 18:55:22.430	1355	1361	system_process	art	Debugger is no longer active
I	01-27 18:55:22.430	1752	1758	android.process.a...	art	Debugger is no longer active
I	01-27 18:55:22.430	1693	1704	com.android.phone	art	Debugger is no longer active
I	01-27 18:55:22.430	1787	1790	com.android.music	art	Debugger is no longer active
I	01-27 18:55:22.430	1675	1682	android.process.m...	art	Debugger is no longer active
I	01-27 18:55:22.430	1480	1487	com.android.exter...	art	Debugger is no longer active
I	01-27 18:55:22.430	1856	1862	com.android.deskc...	art	Debugger is no longer active
I	01-27 18:55:22.430	1901	1903	com.android.quick...	art	Debugger is no longer active

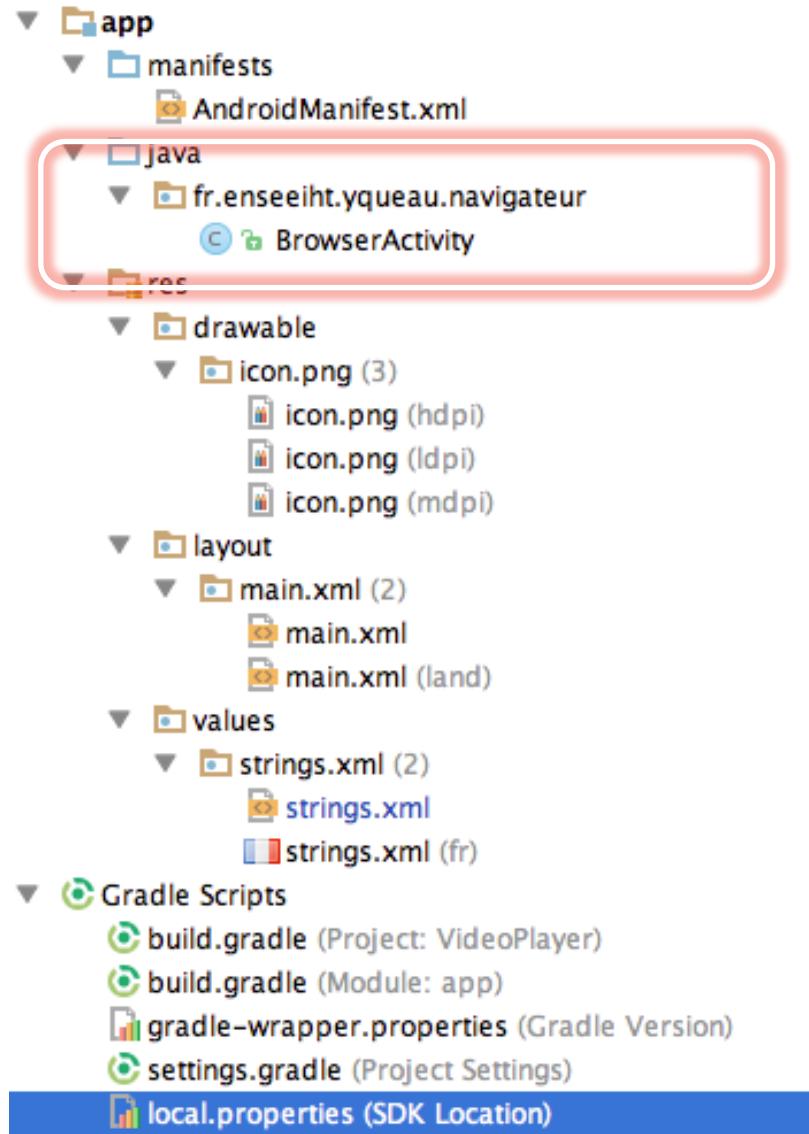
At the bottom right, there is a status bar indicating '134M of 492M' and a trash bin icon.

Android applications - Structure

- Android Studio has different “views” on the app structure
- Real structure may be different



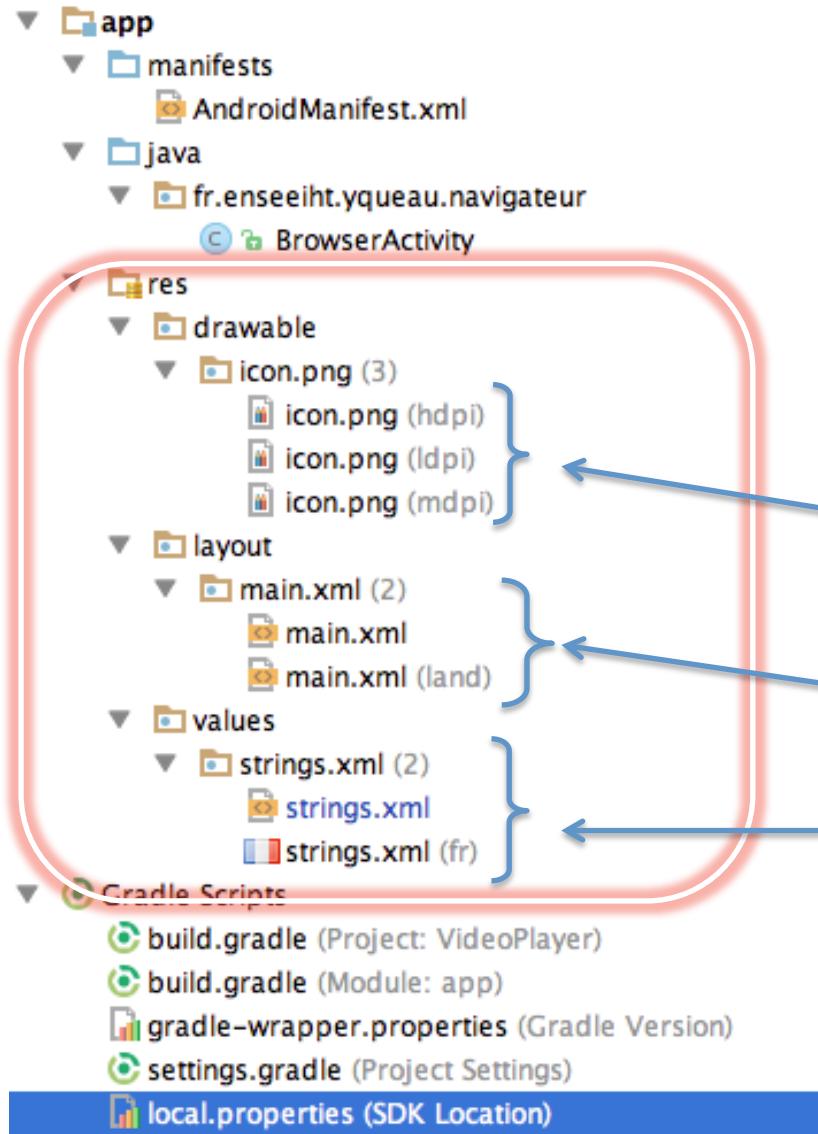
Android applications - Structure



Source code

- **java** folder
- Organize like any Java application
- Package(s)
- `.java` with sources

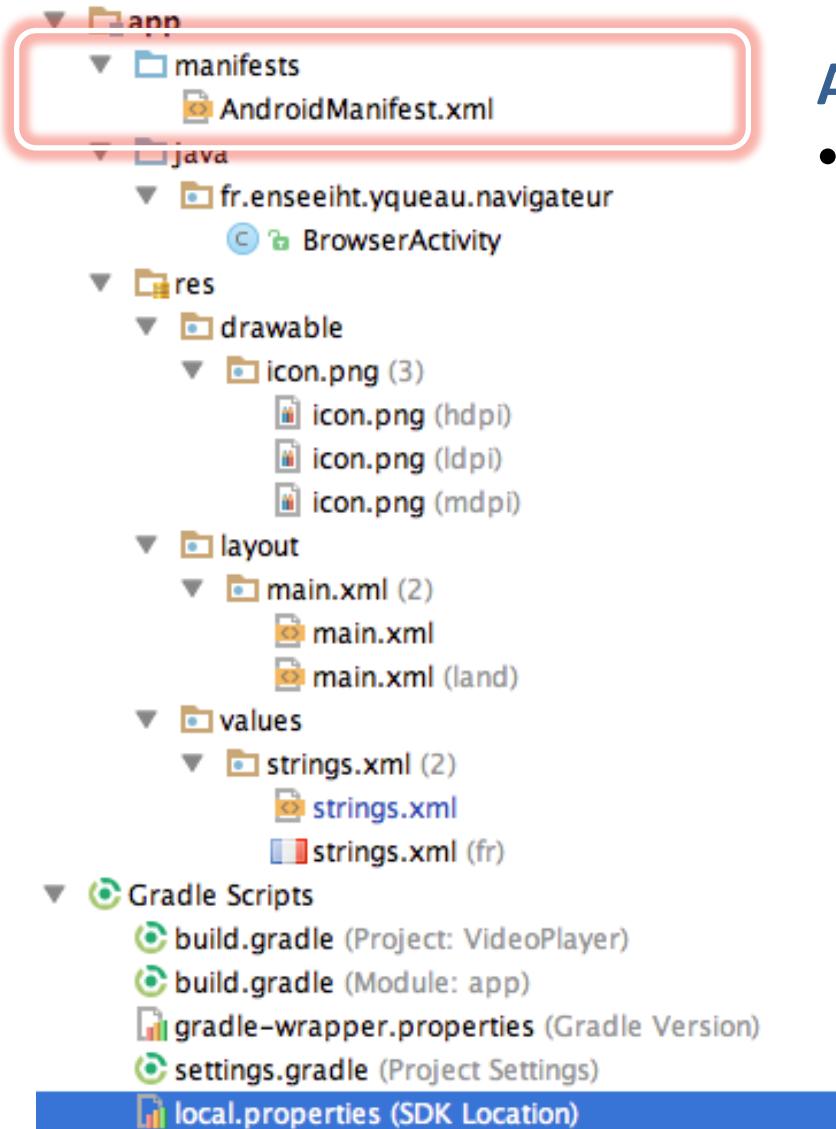
Android applications - Structure



Resource files

- All the GUI related files
 - Icons, text, layout GUI
- Support for multiple screen densities
- Support for multiple layout orientations
- Support for localisation

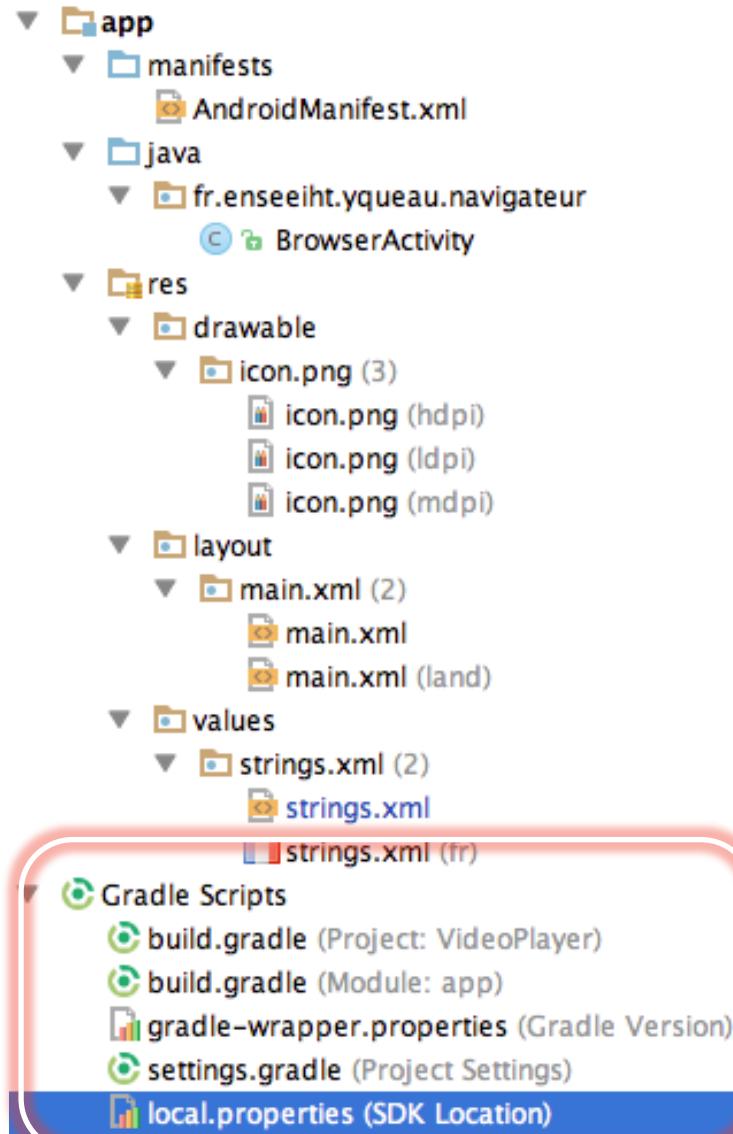
Android applications - Structure



Android Manifest File

- Contains information about the app
 - The name
 - The components (activities, services, broadcast receivers...)
 - Permissions needed to operate (access to sensors, to sdcard, to internet...)

Android applications - Structure



Gradle scripts

- Scripts and configuration file for the building system (Gradle)
- Dependencies
- Targeted version to build

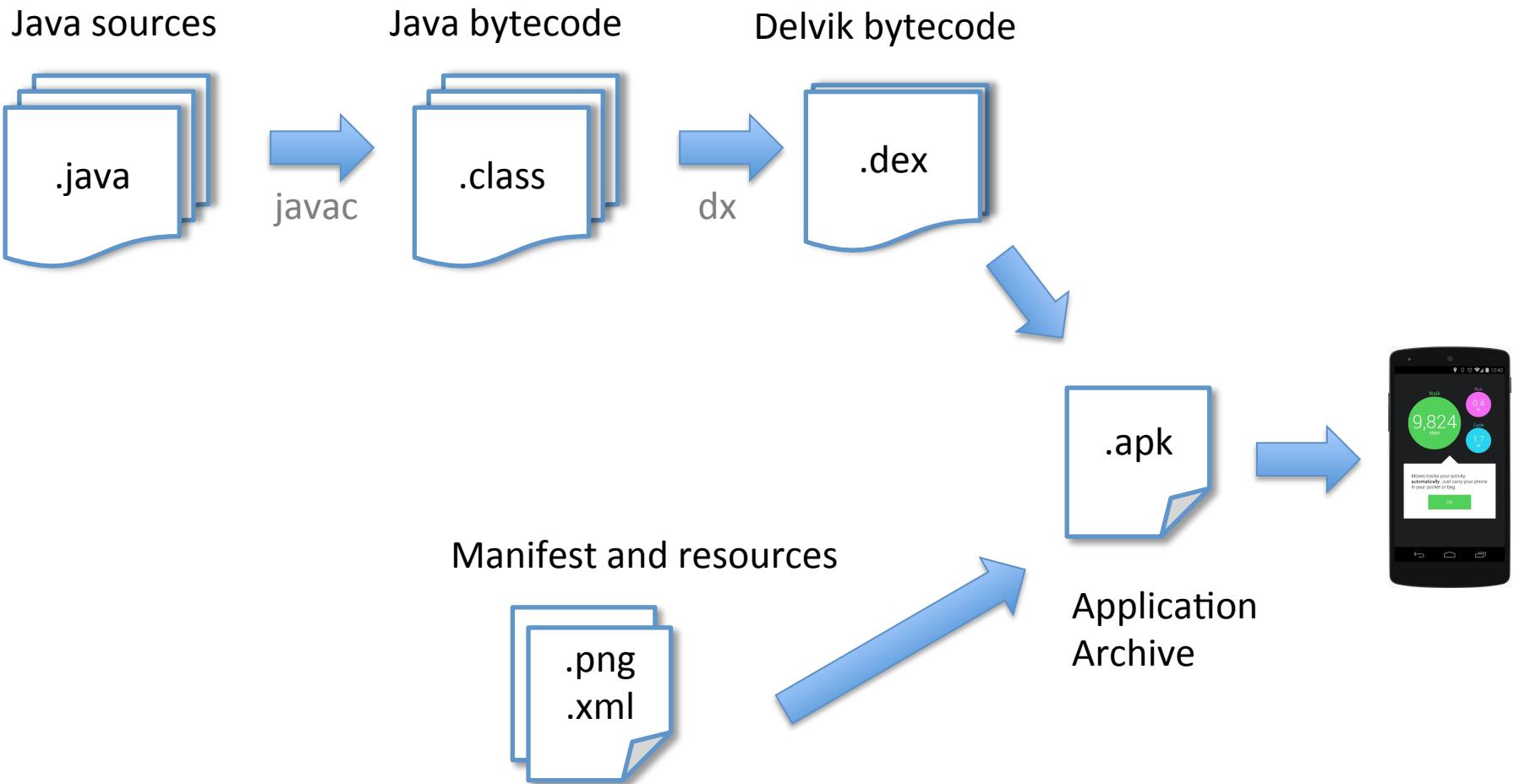
Plan

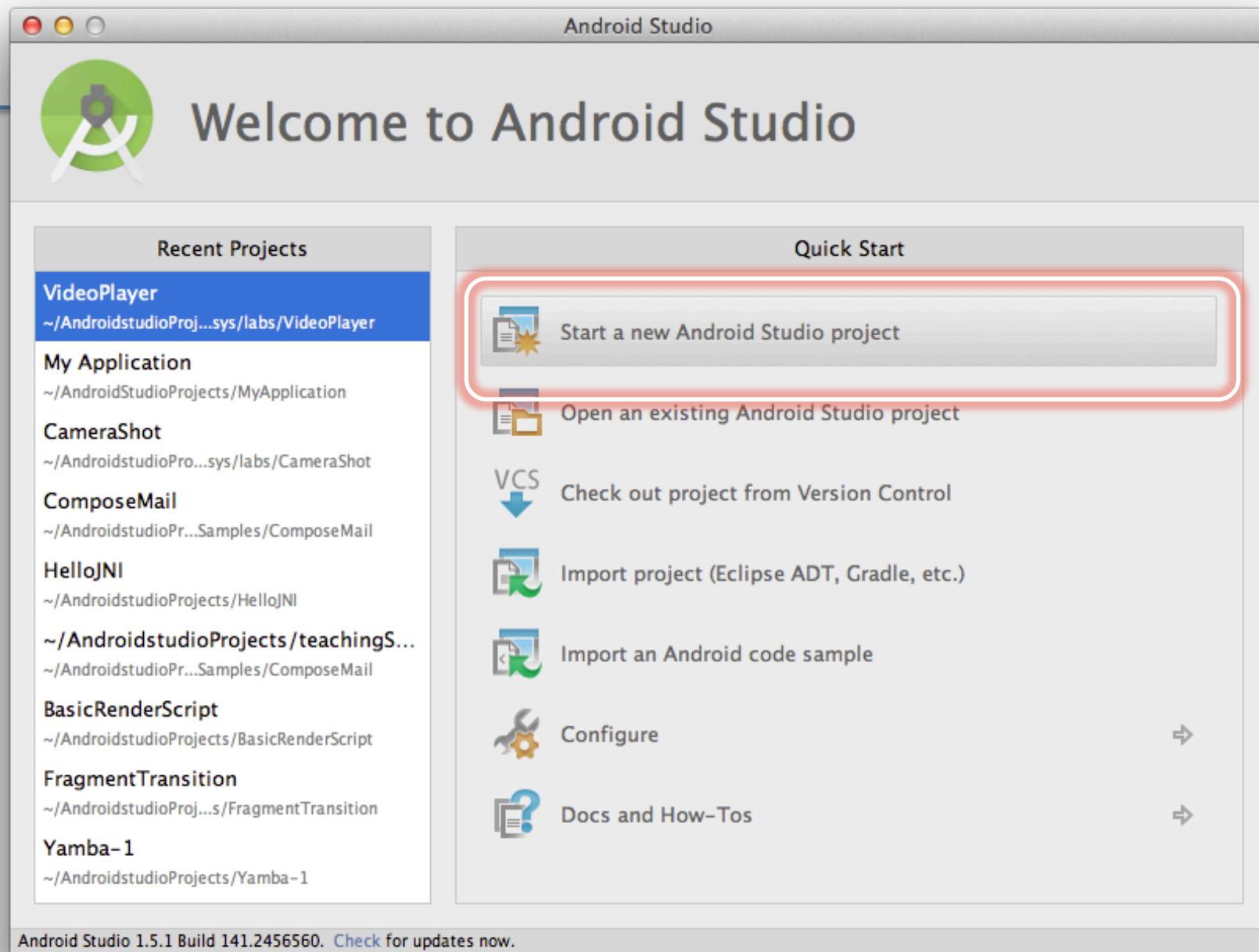
- Mobile devices
- Android overview
 - History
 - Architecture
 - Applications
- The tools
- **Android Applications**

Hello World! Application

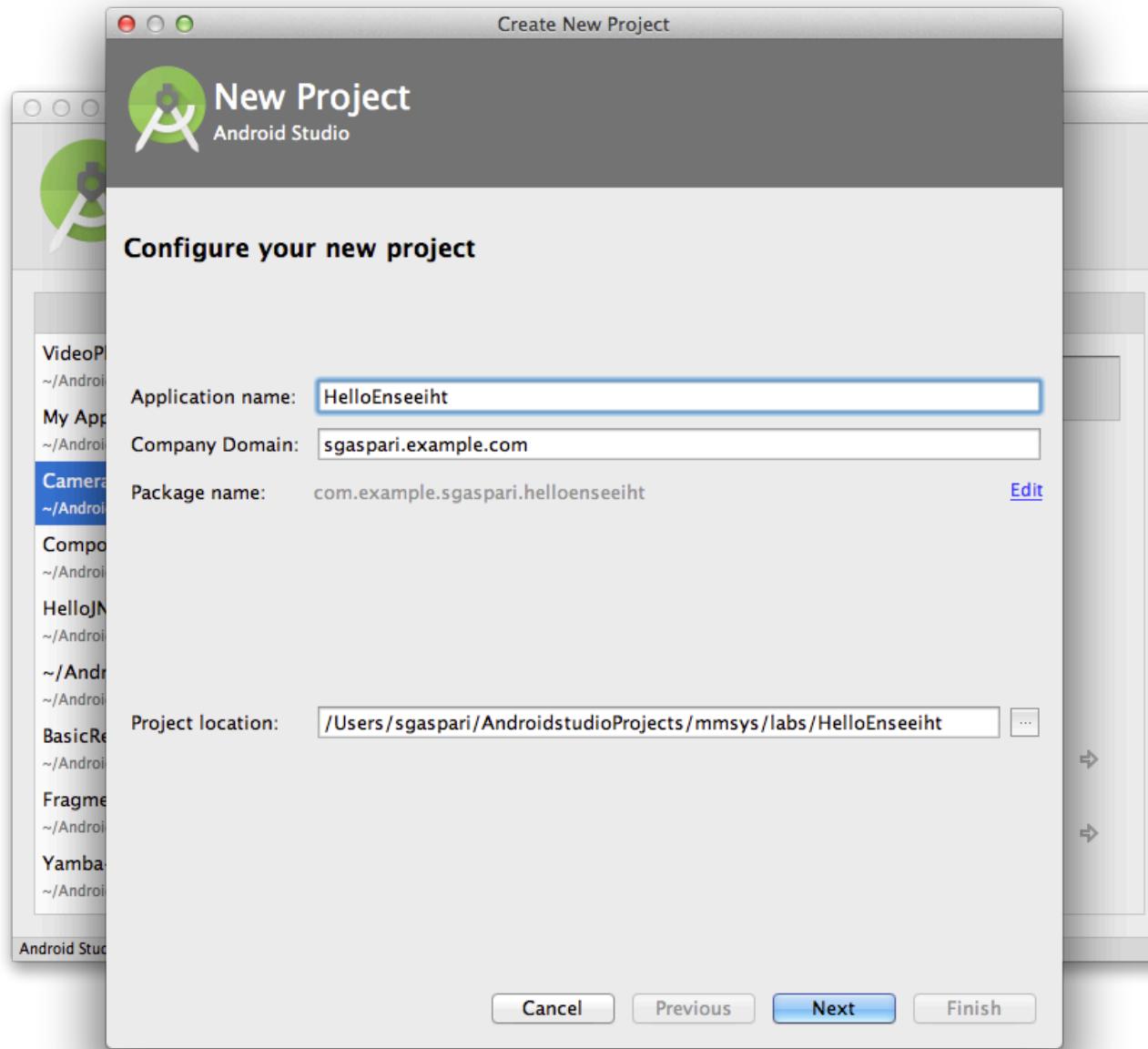
- Demo
- Activity class
 - Lifecycle
- The GUI
- Manifest file

Applications

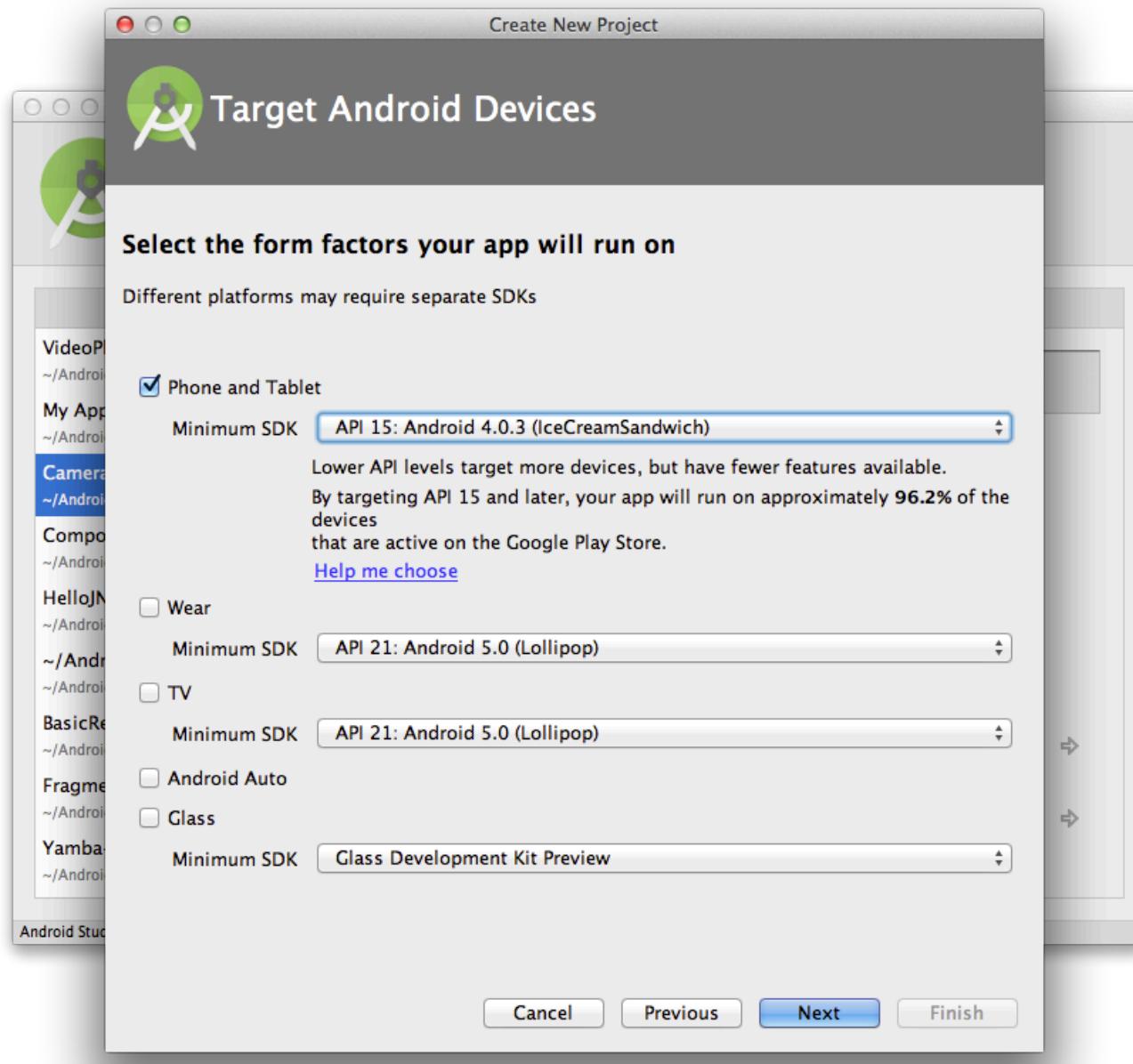




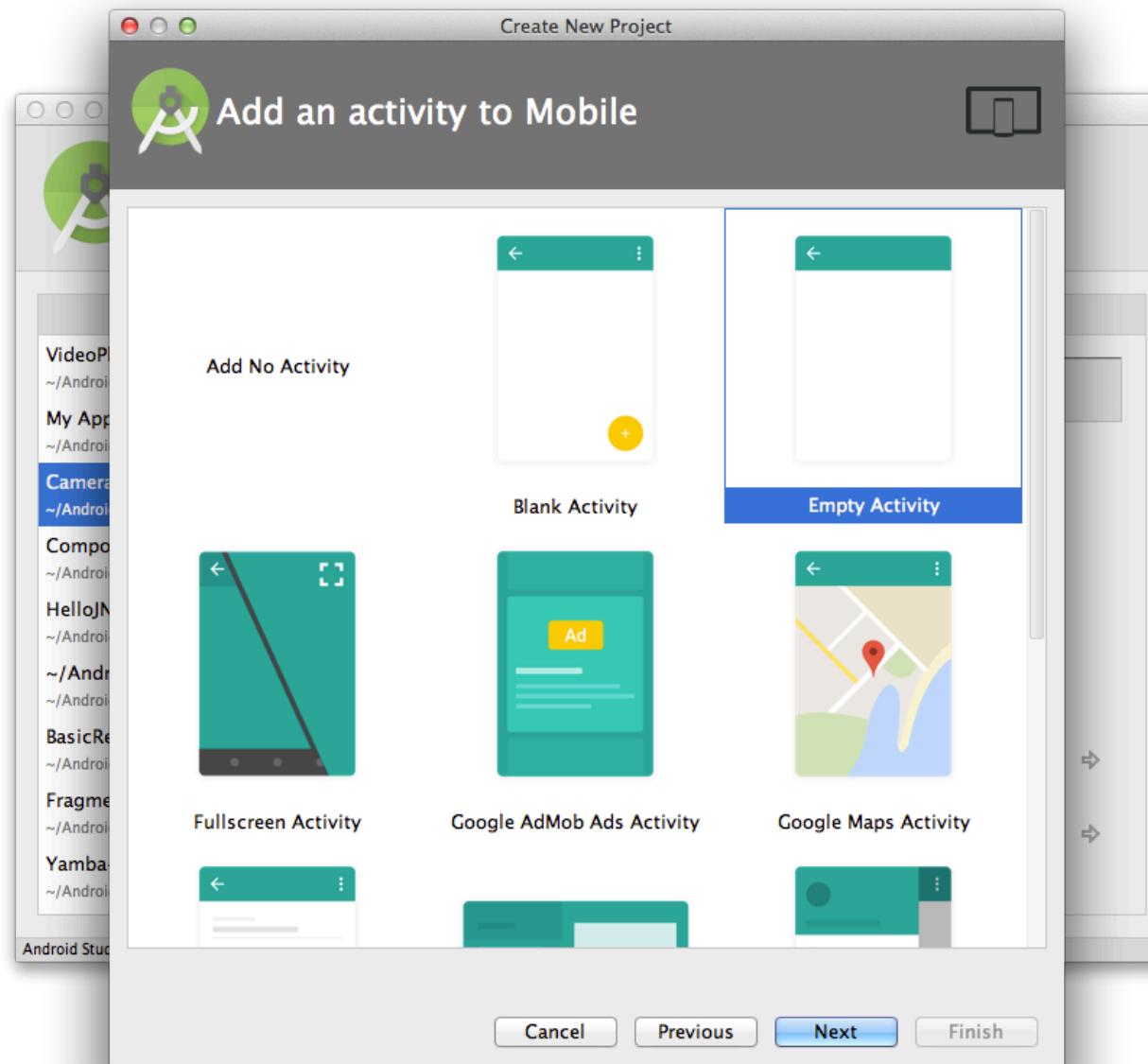
Hello World!



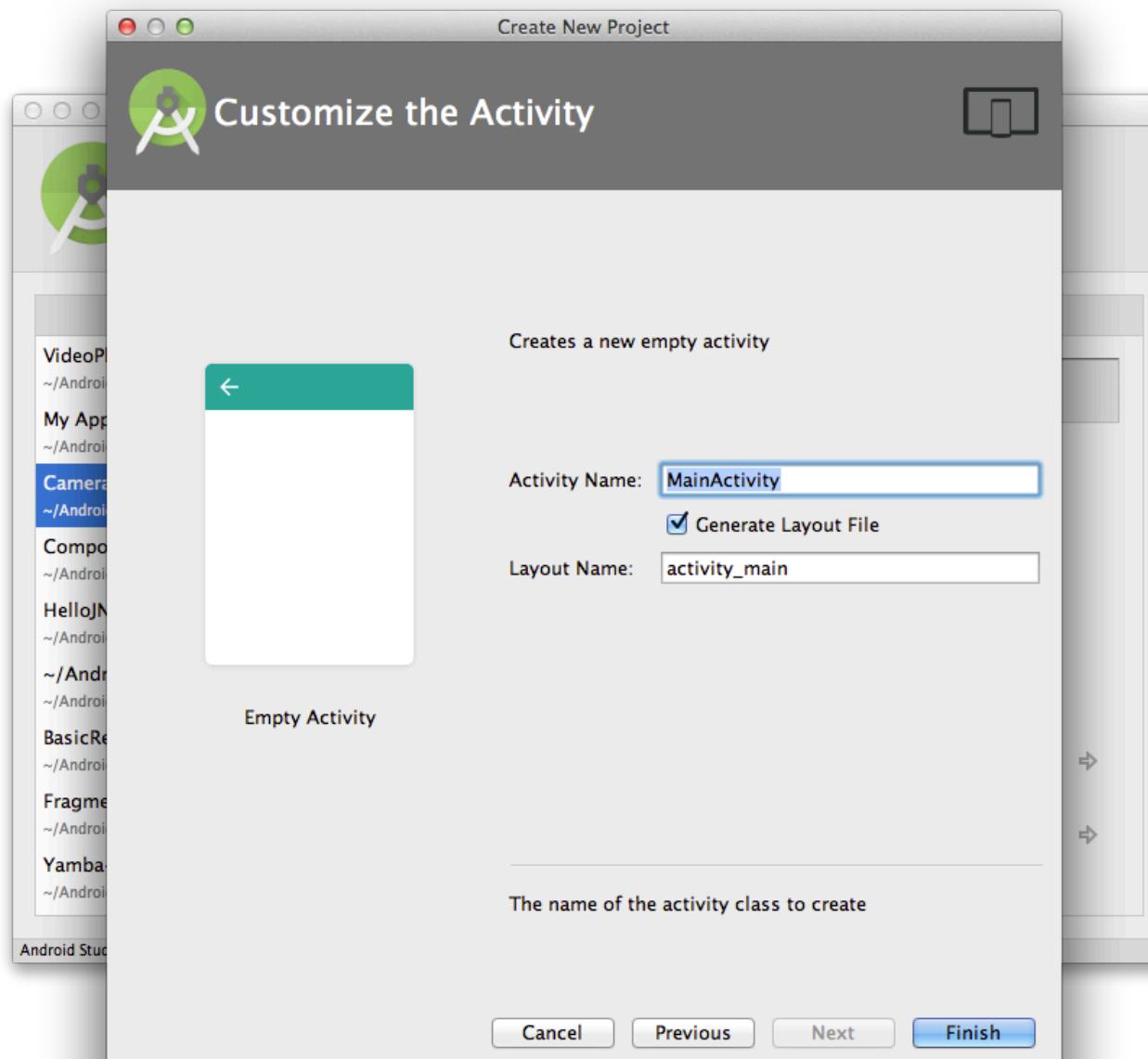
Hello World!



Hello World!



Hello World!



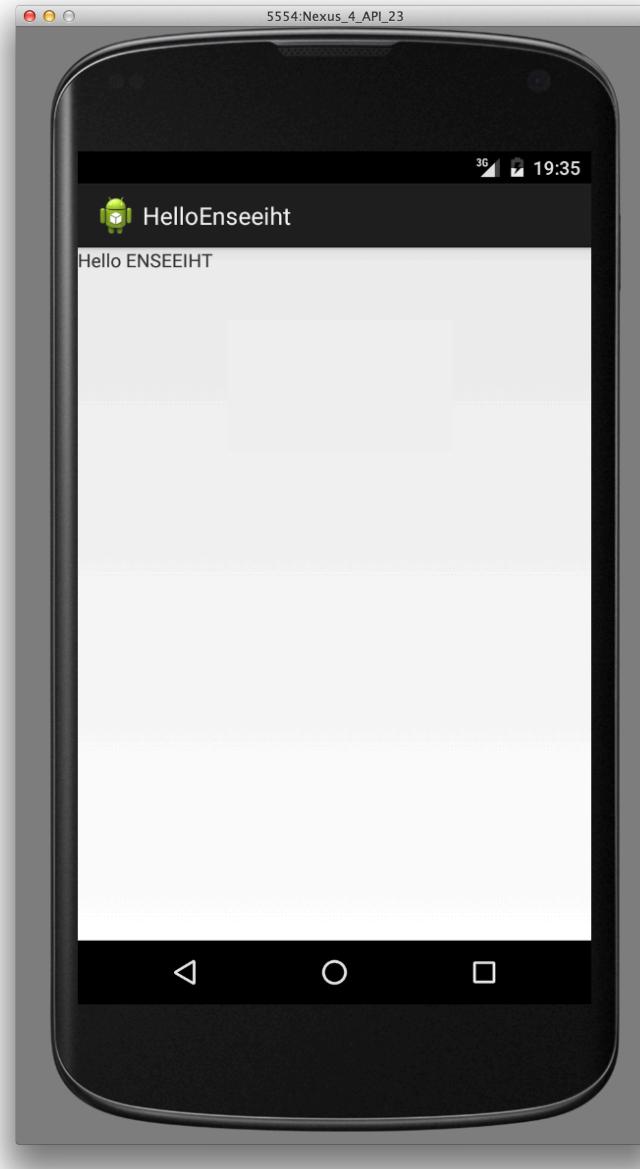
Hello World!

```
package com.example.helloworld;

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

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

Hello World!



Hello World!

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

Hello World! Application

- Demo
- Activity class
 - Lifecycle
- The GUI
- Manifest file

Activity lifecycle

3 main states for an activity according to the current Task

- **Resumed**
 - foreground and ready to interact (“running”)

Activity lifecycle

3 main states for an activity according to the current Task

- **Resumed**

- foreground and ready to interact (“running”)

- **Paused**

- Visible but with another activity in foreground

- i.e. under a dialog box or another activity is about to take focus

- It can be killed by the system in extremely low memory situations.

Activity lifecycle

3 main states for an activity according to the current Task

- **Resumed**

- foreground and ready to interact (“running”)

- **Paused**

- Visible but with another activity in foreground

- i.e. under a dialog box or another activity is about to take focus

- It can be killed by the system in extremely low memory situations.

- **Stopped**

- Not visible, another activity is in foreground

- It is still in memory but...

- ...it can be killed by the system when memory is needed elsewhere.

Activity lifecycle

Activity class provides callbacks for each state transition

- The callbacks can be overridden to perform custom operations

Activity lifecycle

Activity class provides callbacks for each state transition

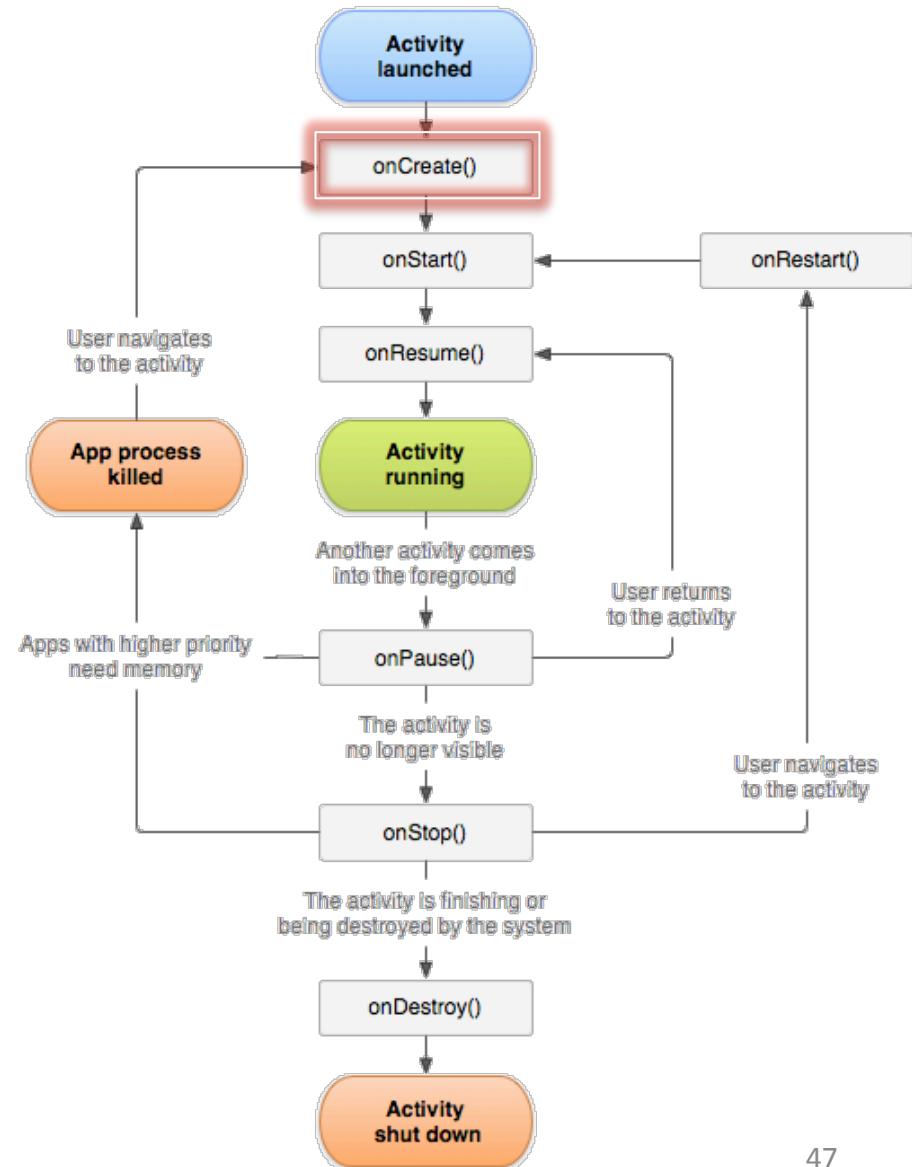
- The callbacks can be overridden to perform custom operations

```
public class ExampleActivity extends Activity {  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        // The activity is being created.  
    }  
    @Override  
    protected void onStart() {  
        super.onStart();  
        // The activity is about to become visible.  
    }  
    @Override  
    protected void onResume() {  
        super.onResume();  
        // The activity has become visible (it is now "resumed").  
    }  
    @Override  
    protected void onPause() {  
        super.onPause();  
        // Another activity is taking focus (this activity is about to be "paused").  
    }  
    @Override  
    protected void onStop() {  
        super.onStop();  
        // The activity is no longer visible (it is now "stopped")  
    }  
    @Override  
    protected void onDestroy() {  
        super.onDestroy();  
        // The activity is about to be destroyed.  
    }  
}
```

Activity Lifecycle callback methods

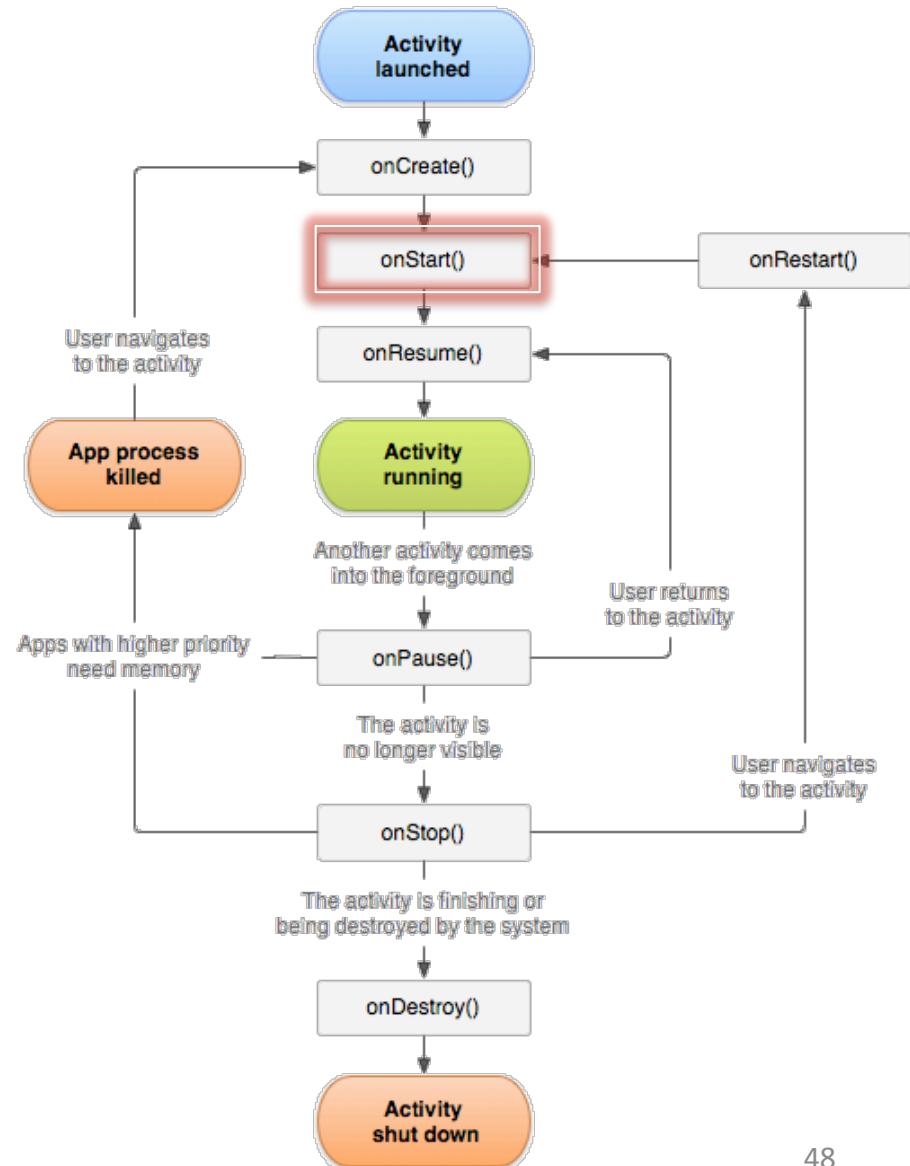
- **onCreate()** – called to initialize an Activity when it is first created

- Startup operations
- Ex. Load the GUI



Activity Lifecycle callback methods

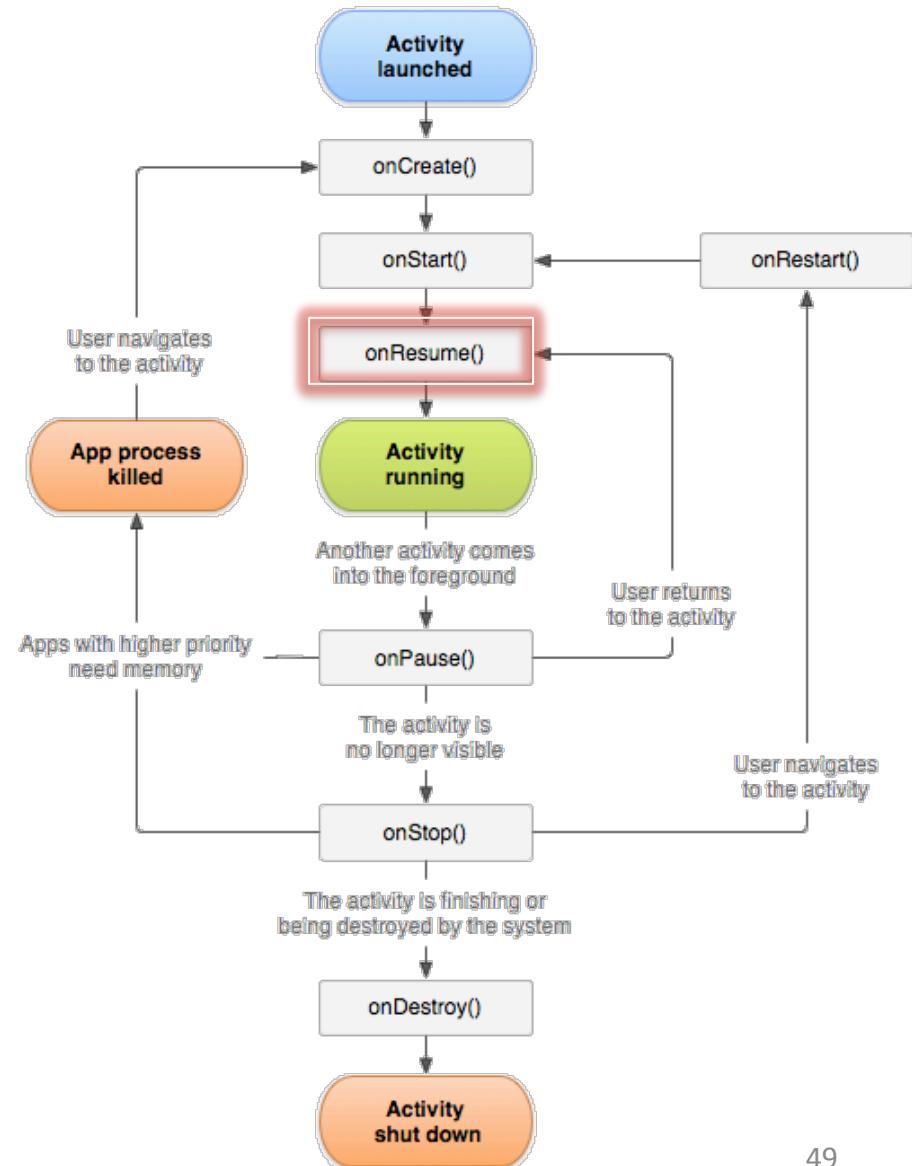
- `onCreate()` – called to initialize an Activity when it is first created
- `onStart()` – called when Activity is becoming visible to the user



Activity Lifecycle callback methods

- **onCreate()** – called to initialize an Activity when it is first created
- **onStart()** – called when Activity is becoming visible to the user
- **onResume()** – called when user returns to an Activity from another

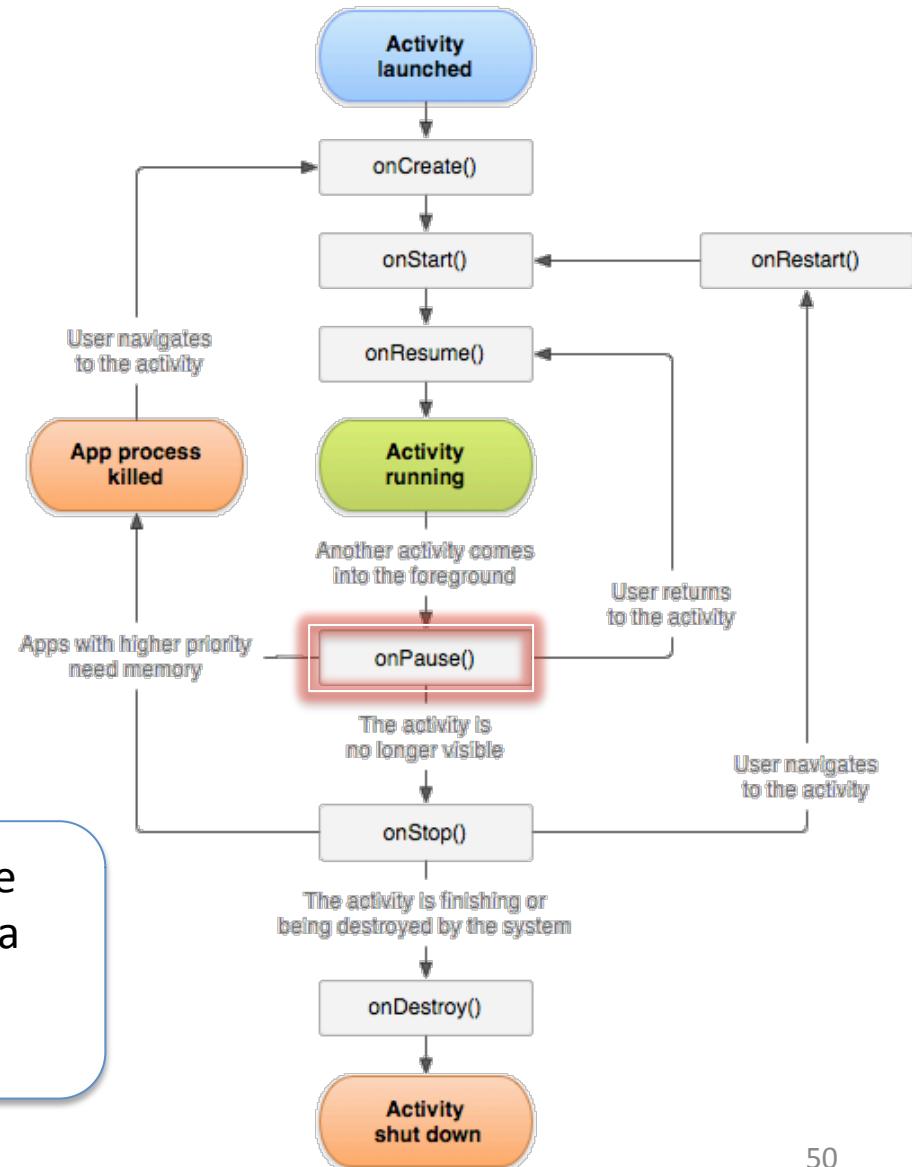
initialize components like sensors, camera etc



Activity Lifecycle callback methods

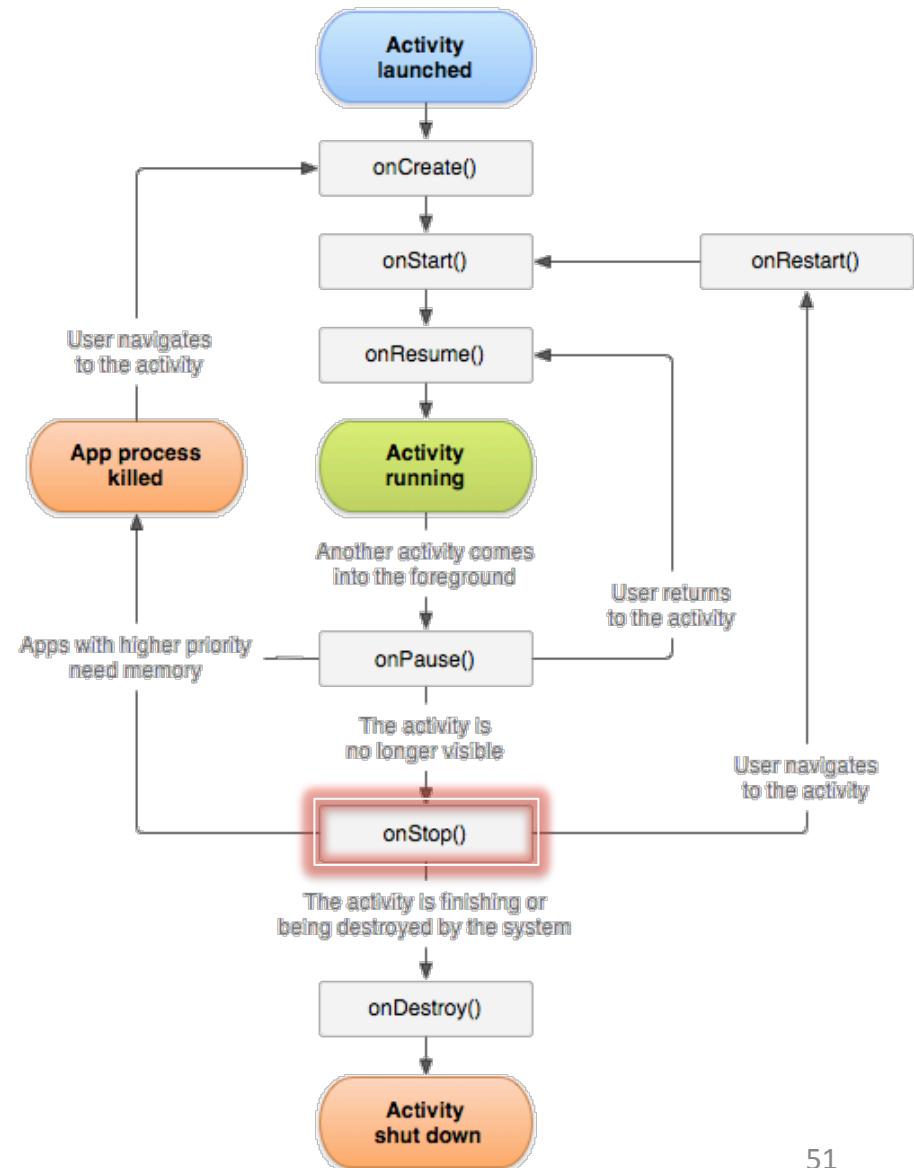
- **onCreate()** – called to initialize an Activity when it is first created
- **onStart()** – called when Activity is becoming visible to the user
- **onResume()** – called when user returns to an Activity from another
- **onPause()** – called when user leaves an Activity that's still visible in background

- Dismiss all the resources that consume computational power (sensors, camera etc)
- Save data if needed



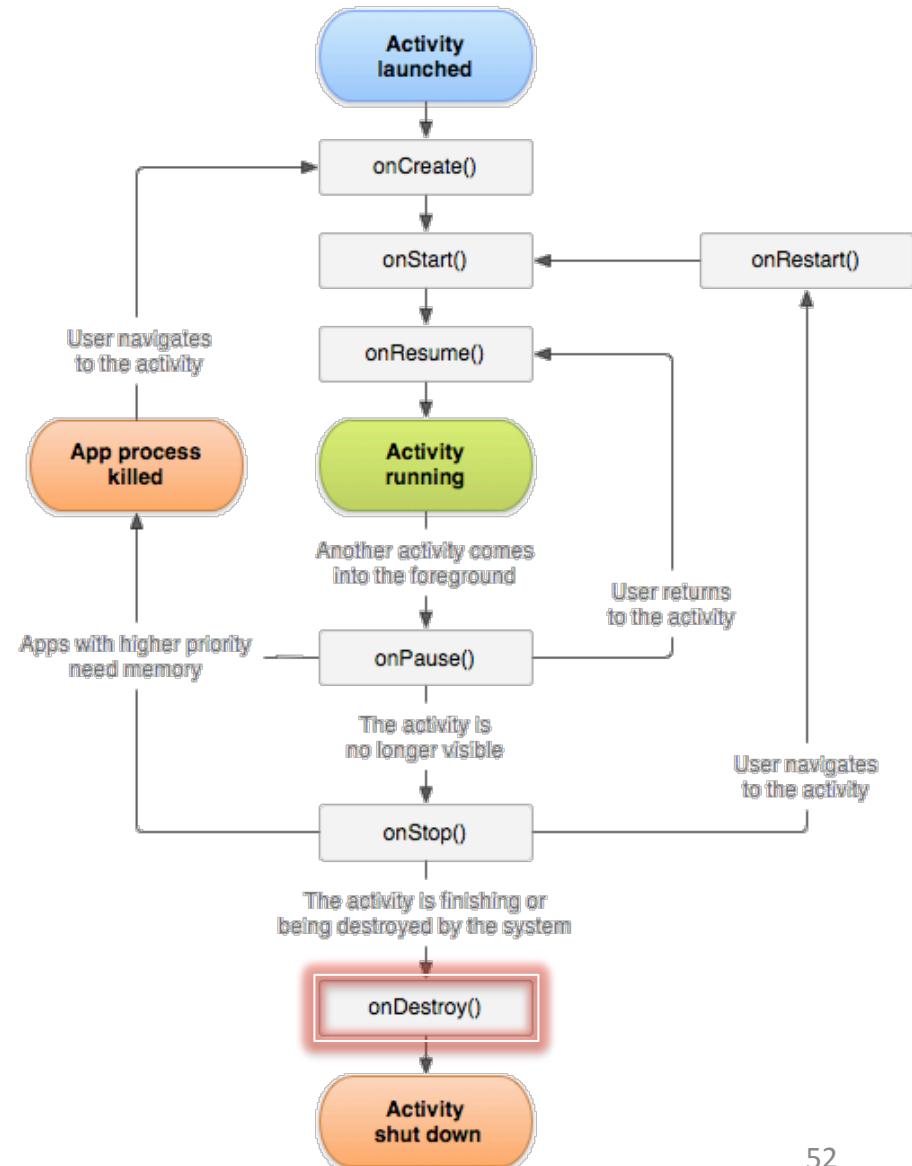
Activity Lifecycle callback methods

- **onCreate()** – called to initialize an Activity when it is first created
- **onStart()** – called when Activity is becoming visible to the user
- **onResume()** – called when user returns to an Activity from another
- **onPause()** – called when user leaves an Activity that's still visible in background
- **onStop()** – called when user leaves an Activity for another



Activity Lifecycle callback methods

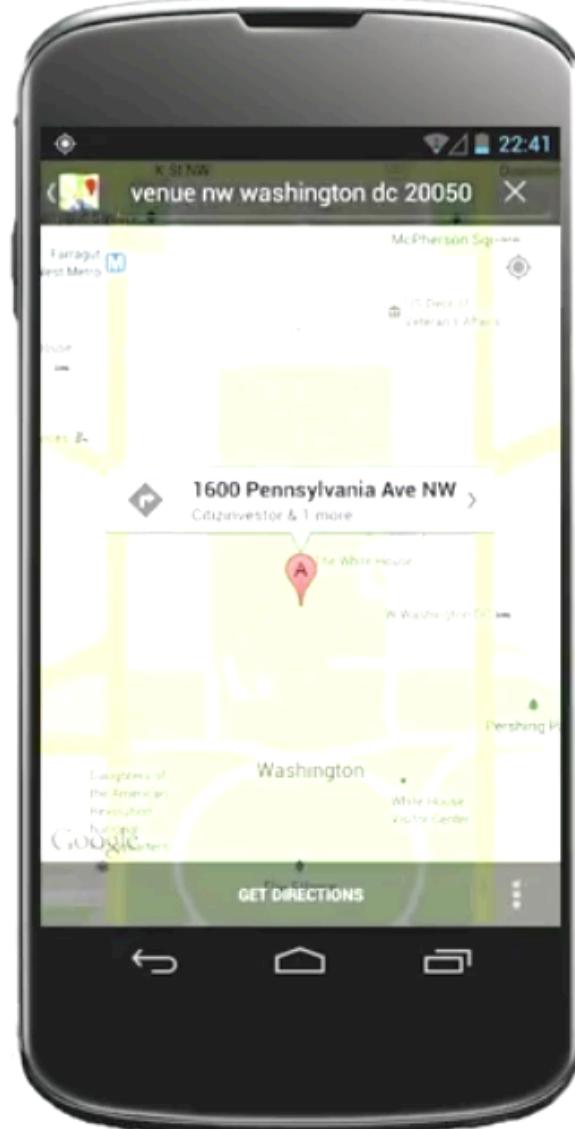
- **onCreate()** – called to initialize an Activity when it is first created
- **onStart()** – called when Activity is becoming visible to the user
- **onResume()** – called when user returns to an Activity from another
- **onPause()** – called when user leaves an Activity that's still visible in background
- **onStop()** – called when user leaves an Activity for another
- **onDestroy()** – called when Activity is being released & needs to clean up its allocated resources



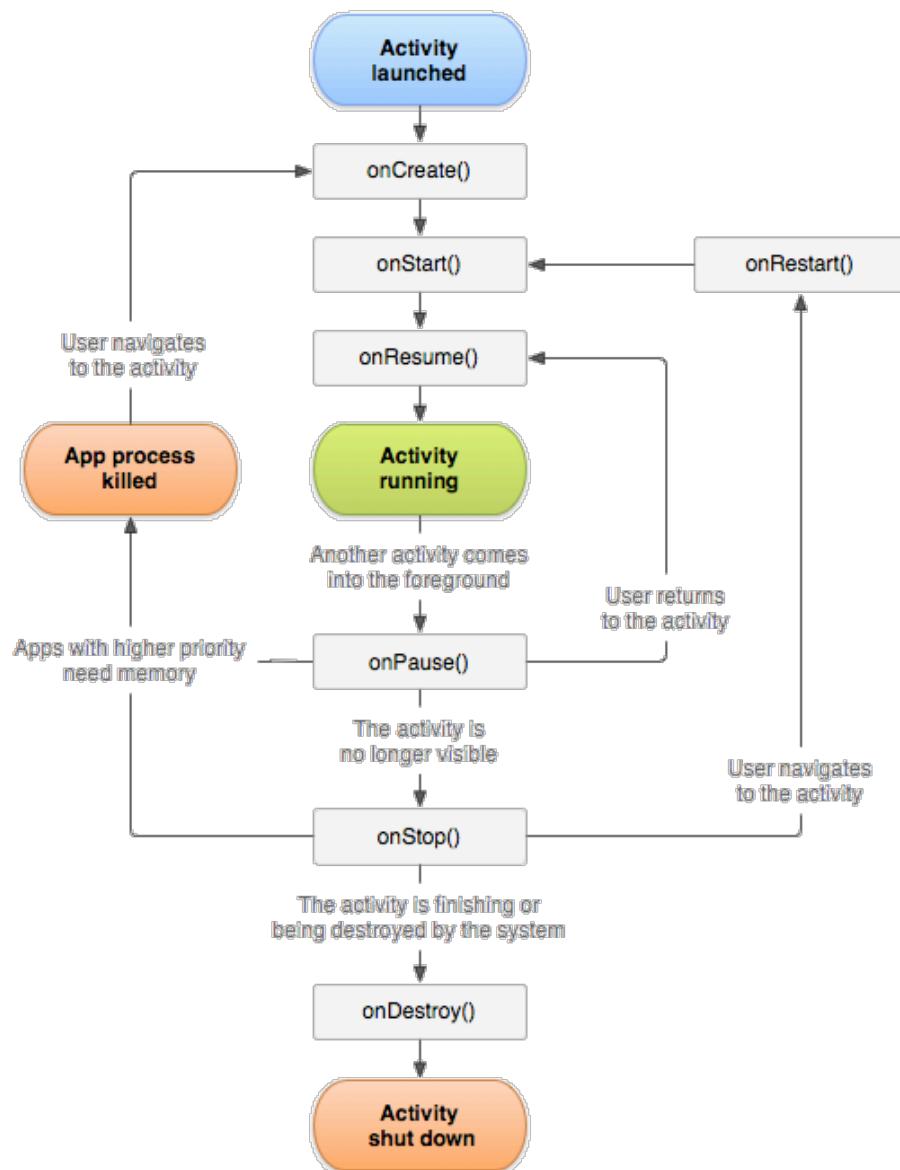
Example



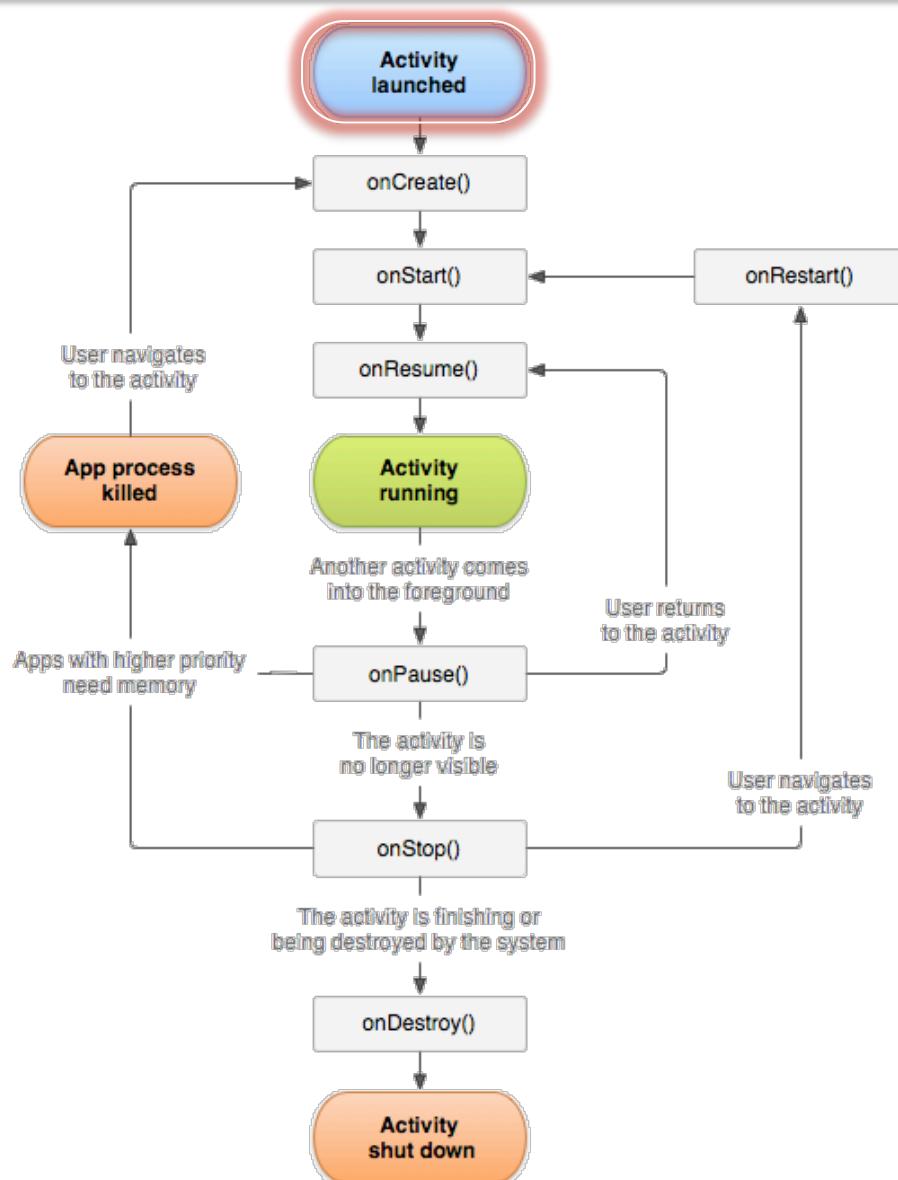
Main application
launch an **intent** to
show a location on
a map



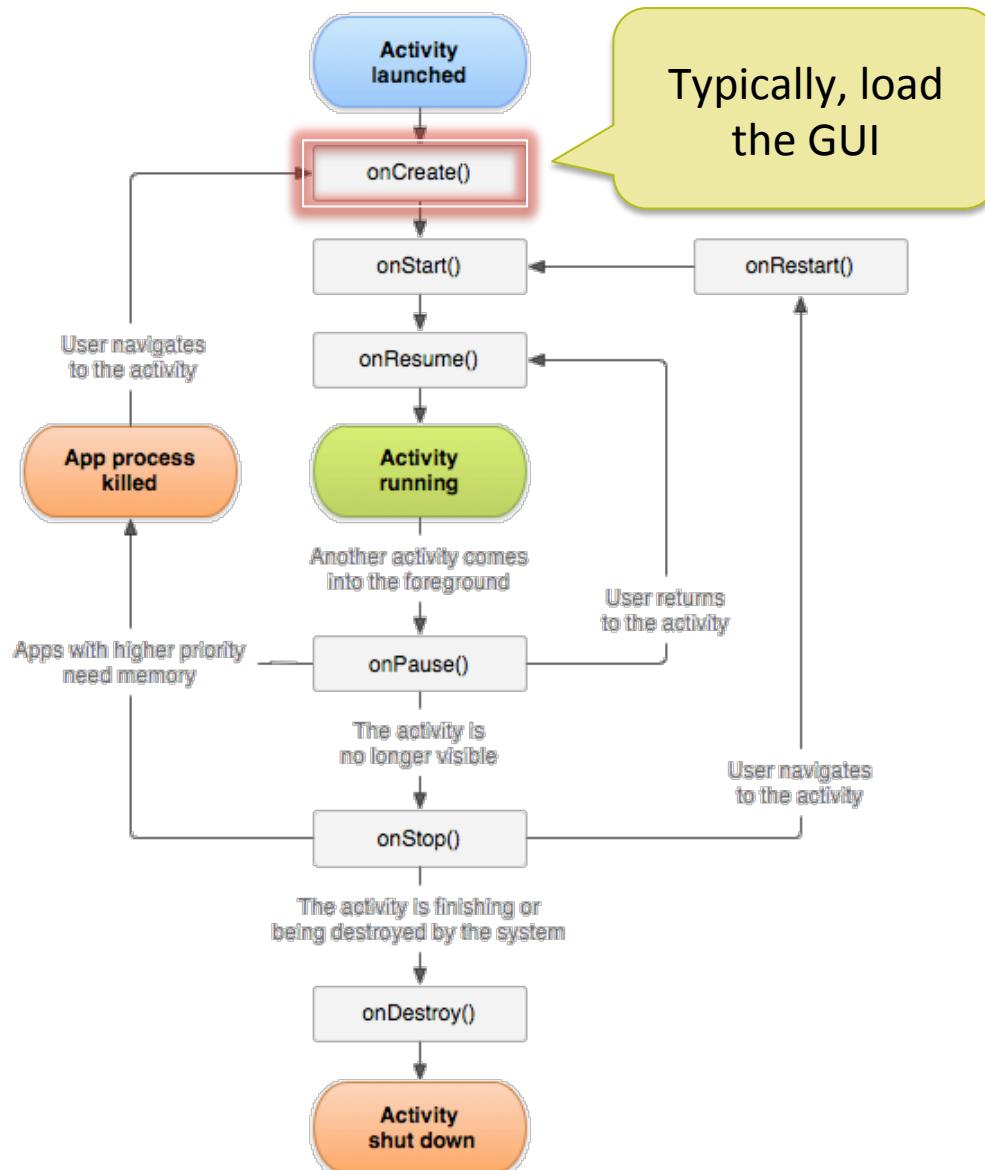
Example



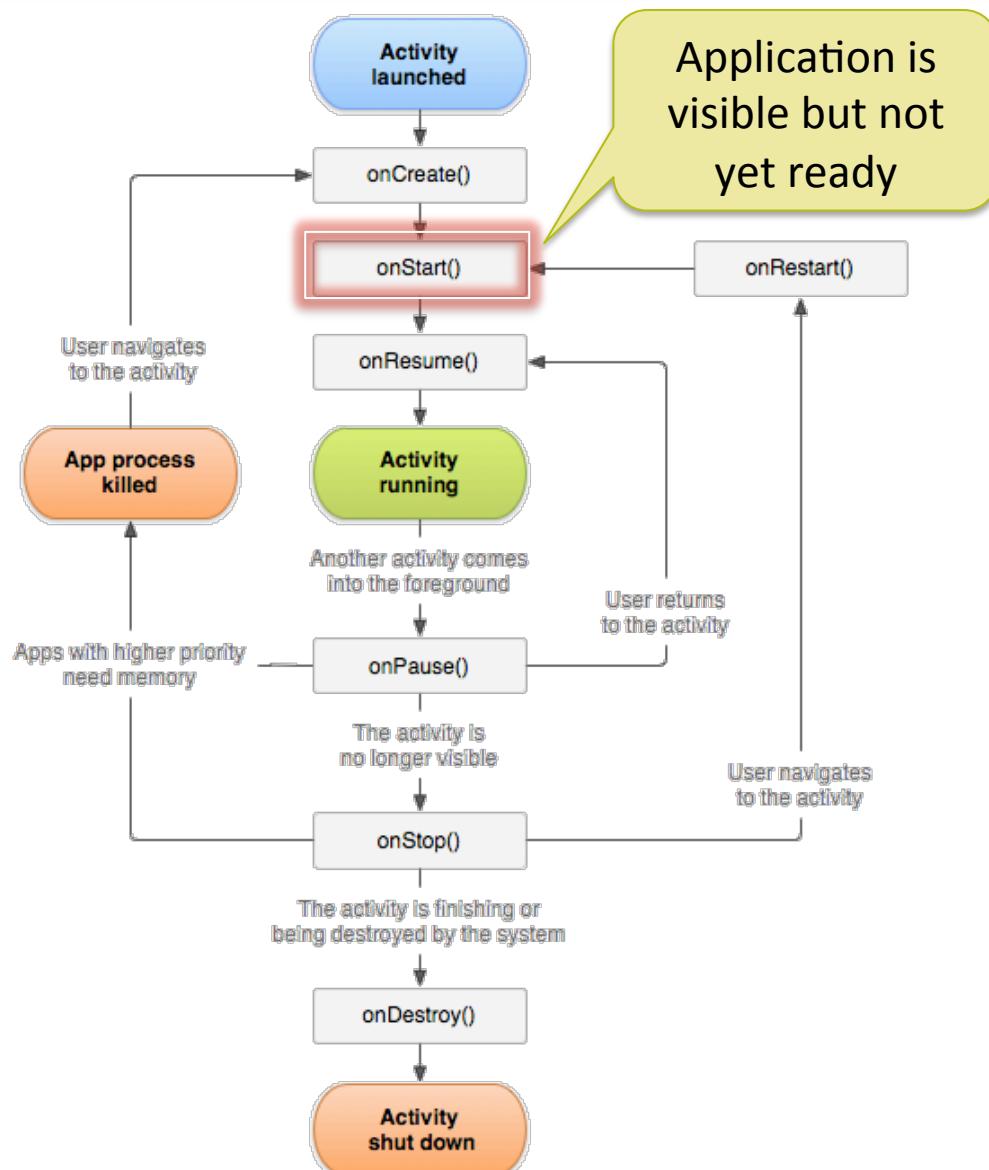
Example



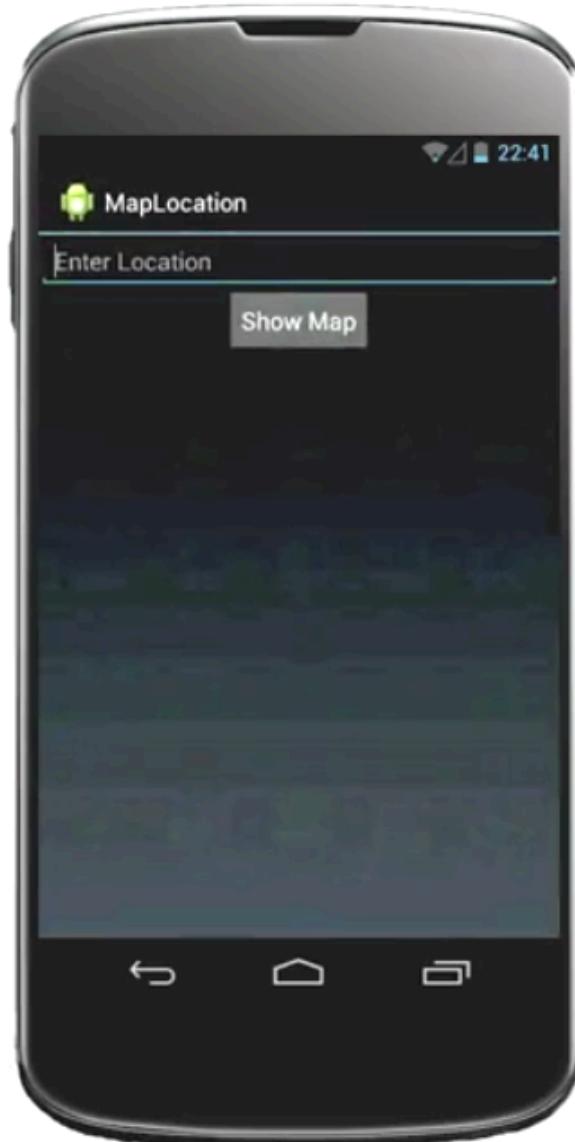
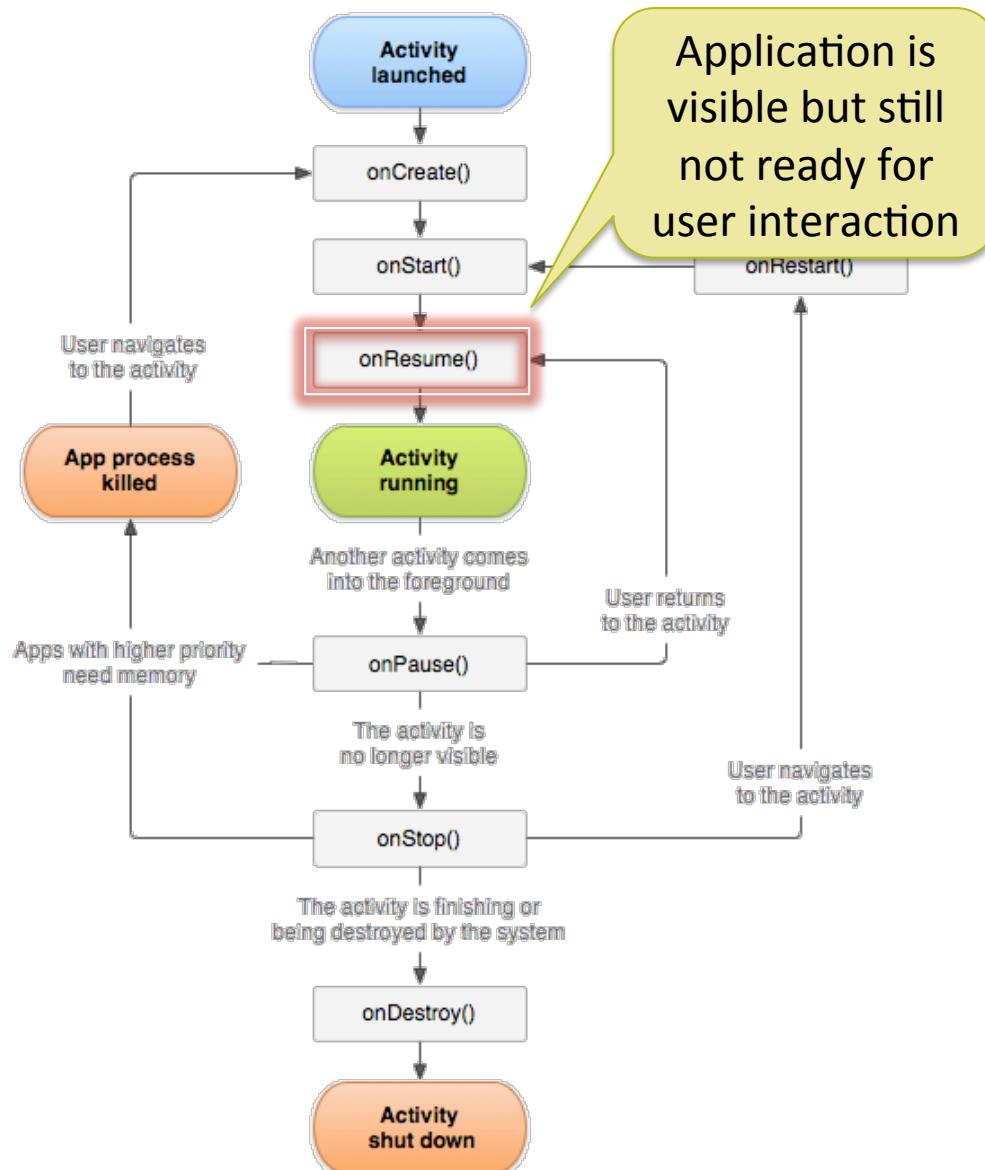
Example



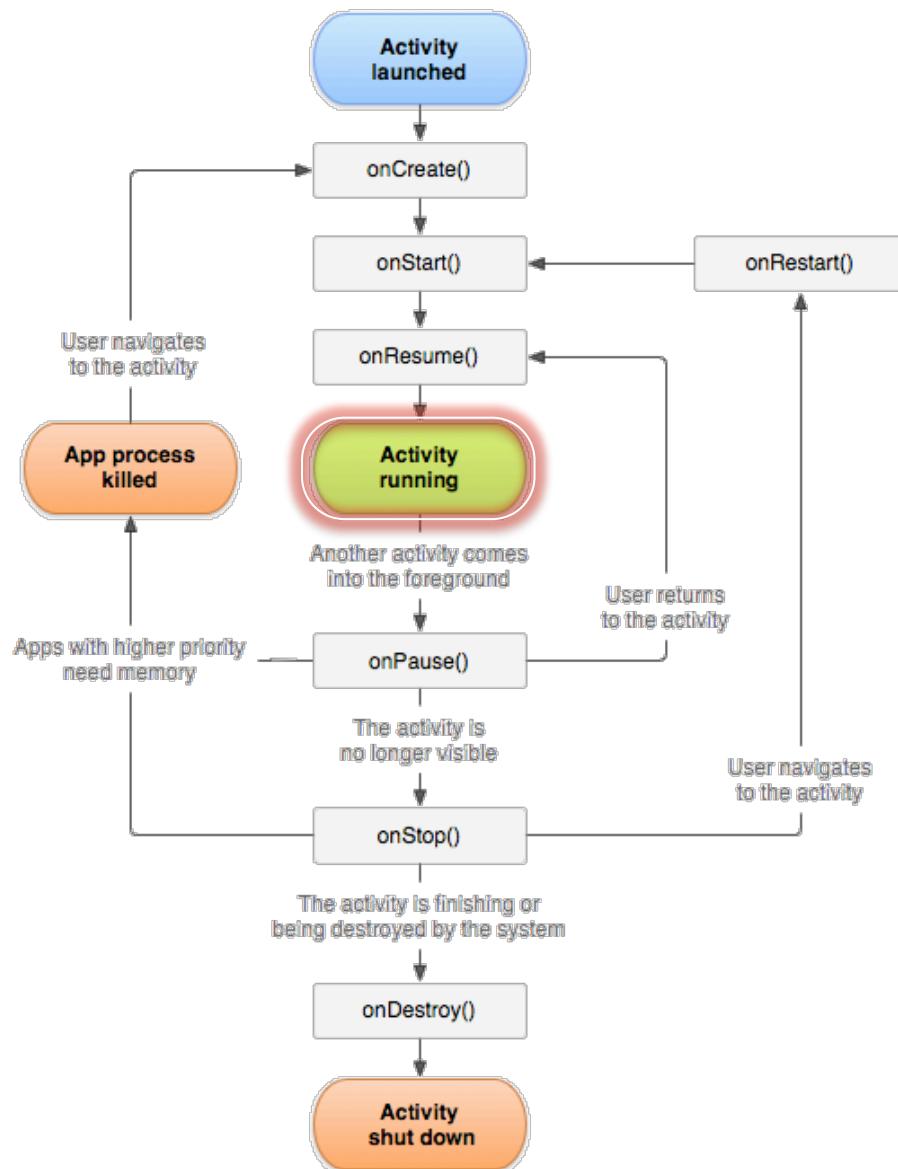
Example



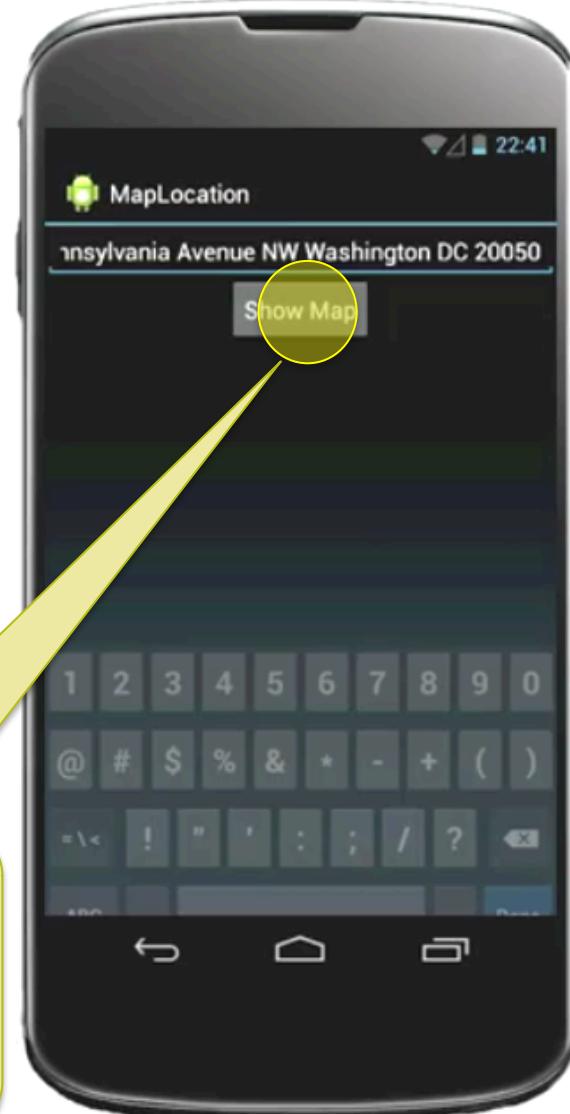
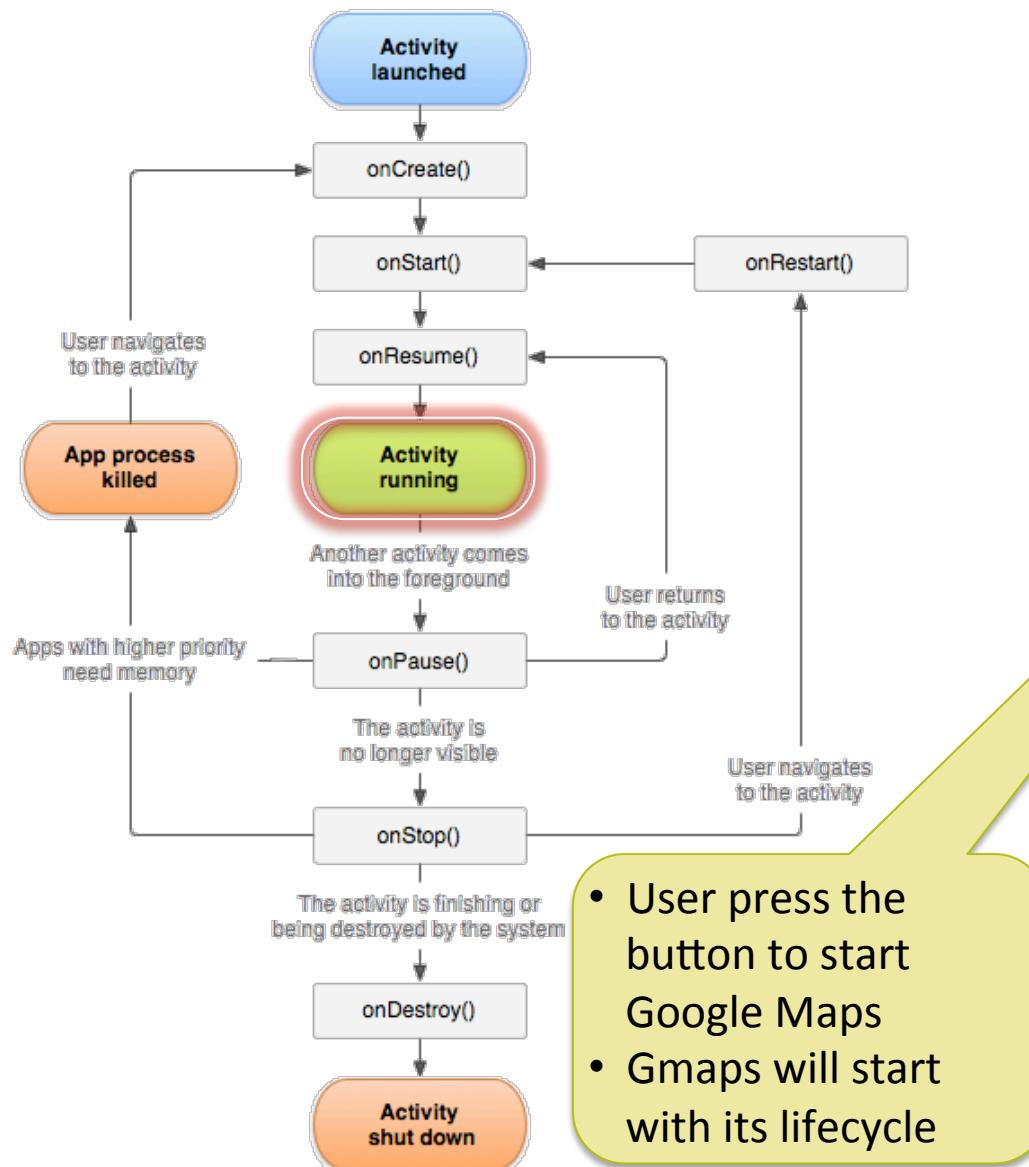
Example



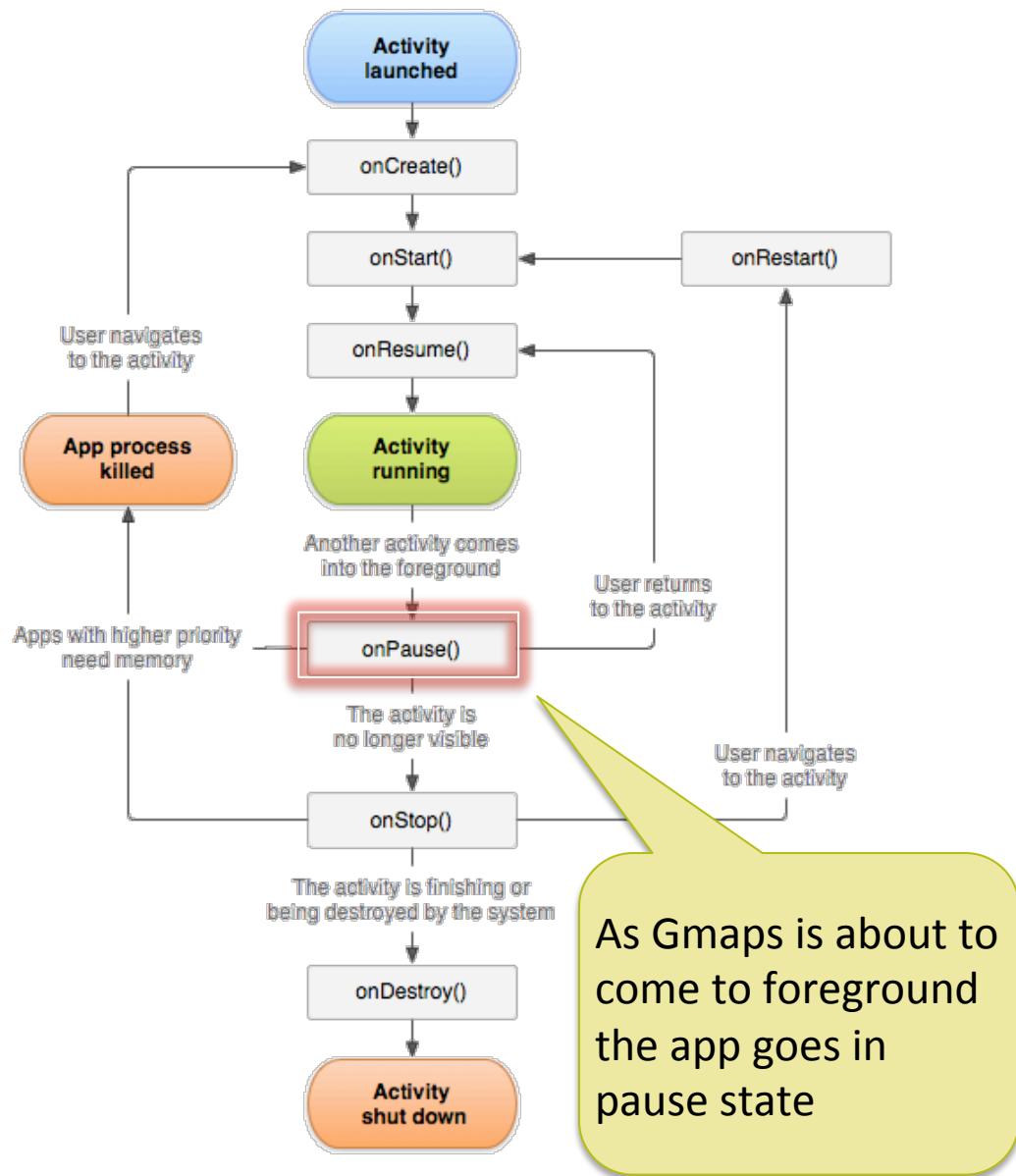
Example



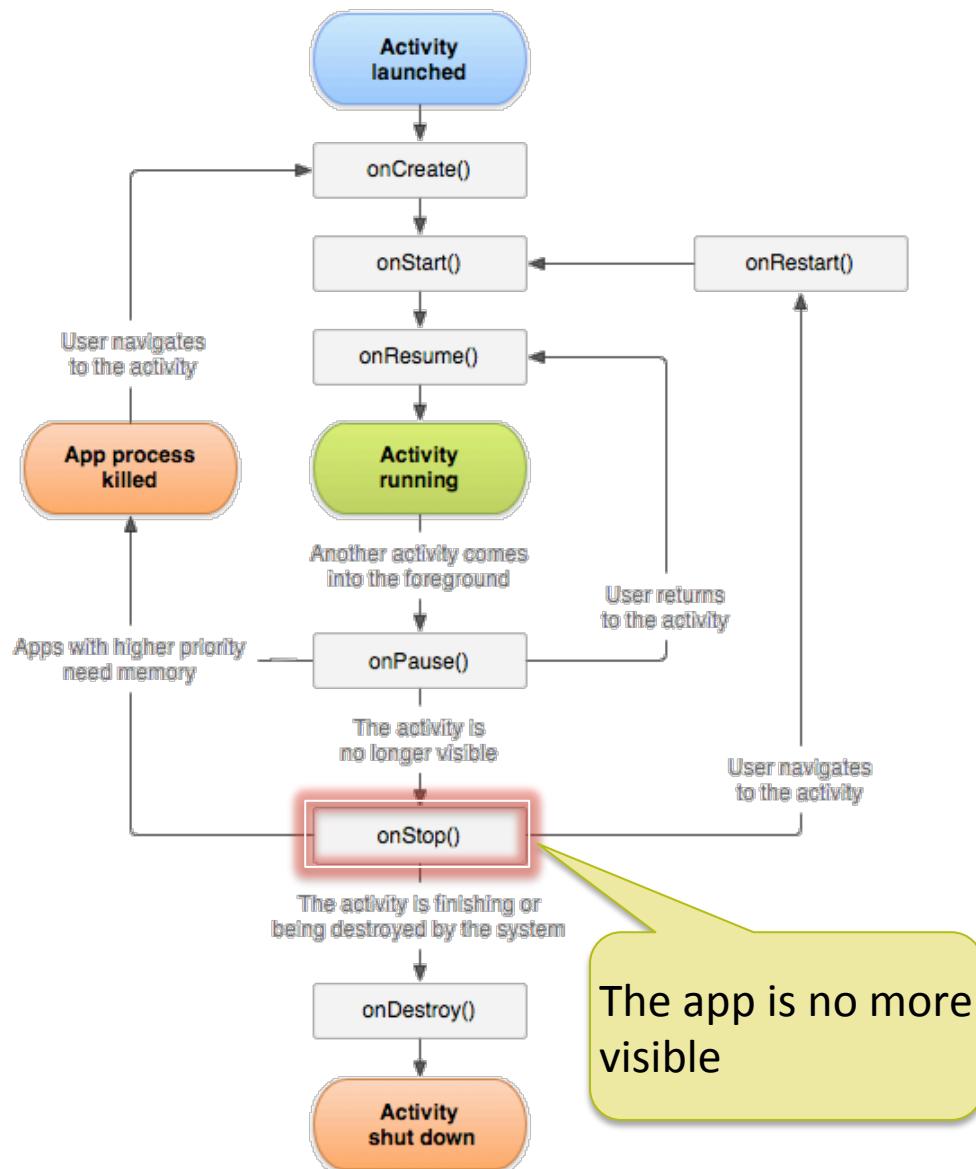
Example



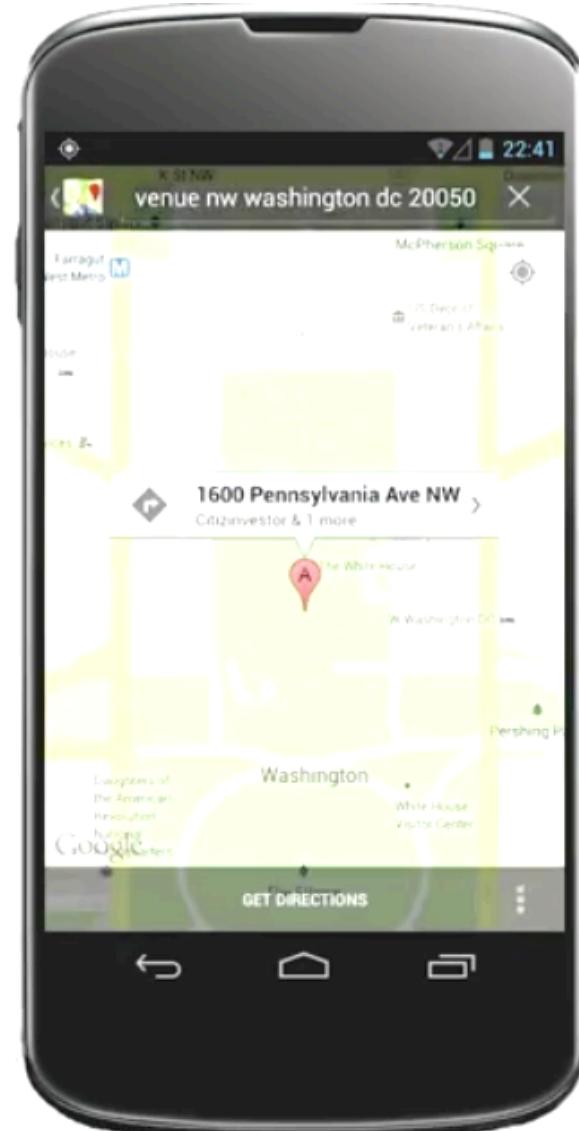
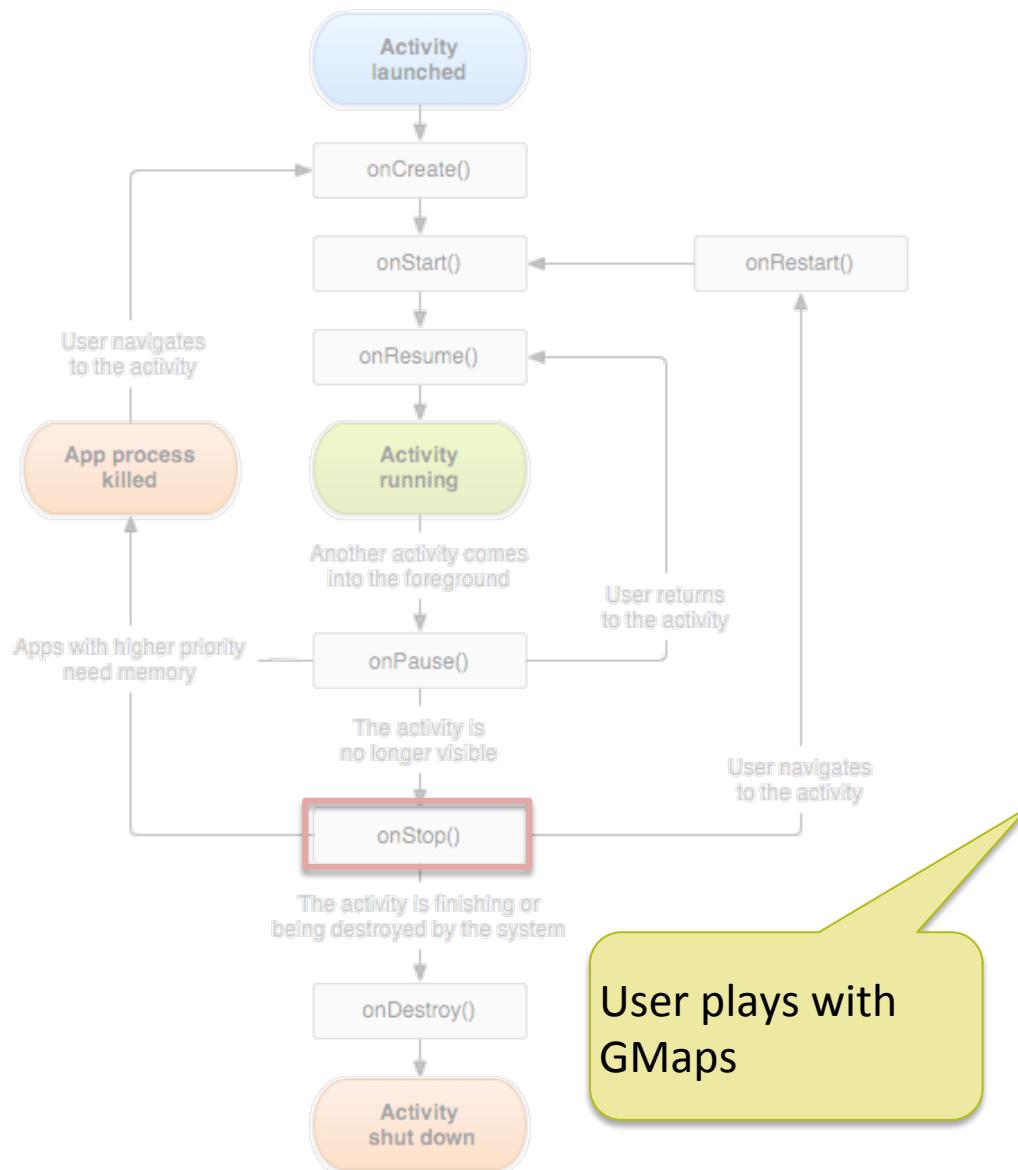
Example



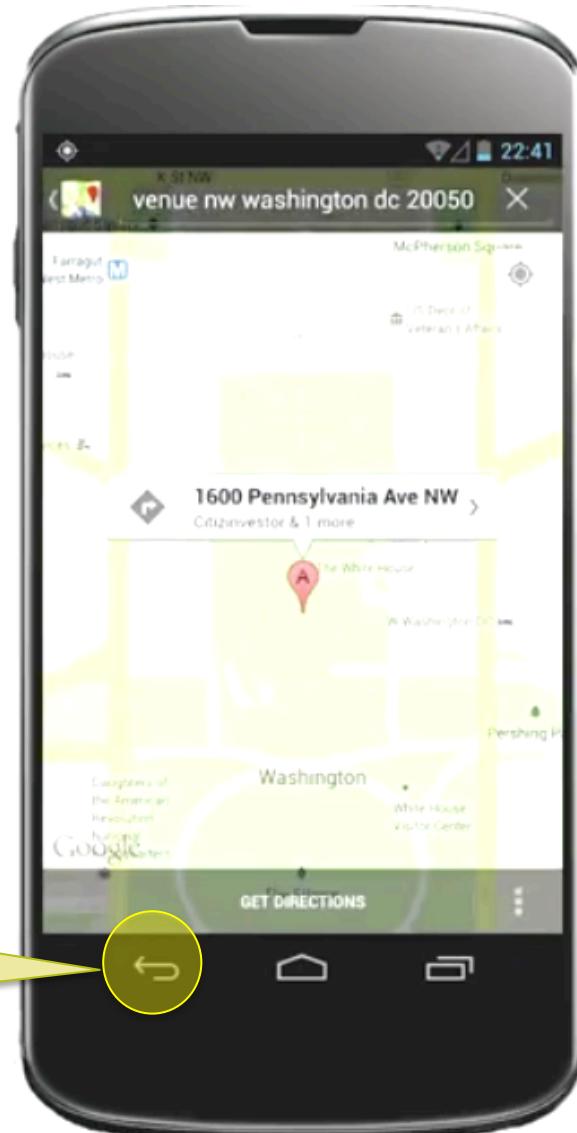
Example



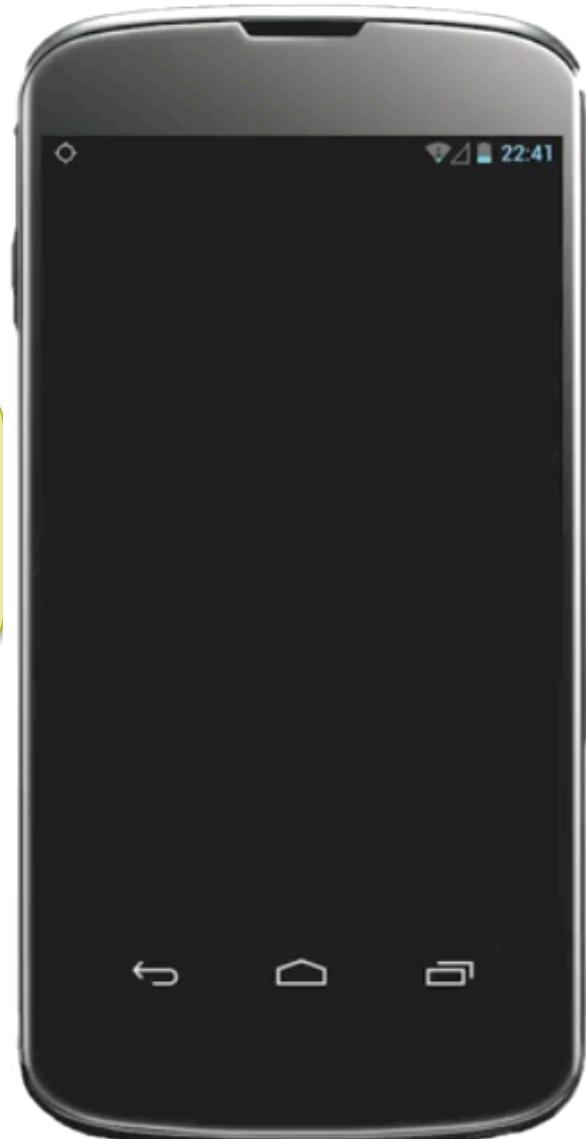
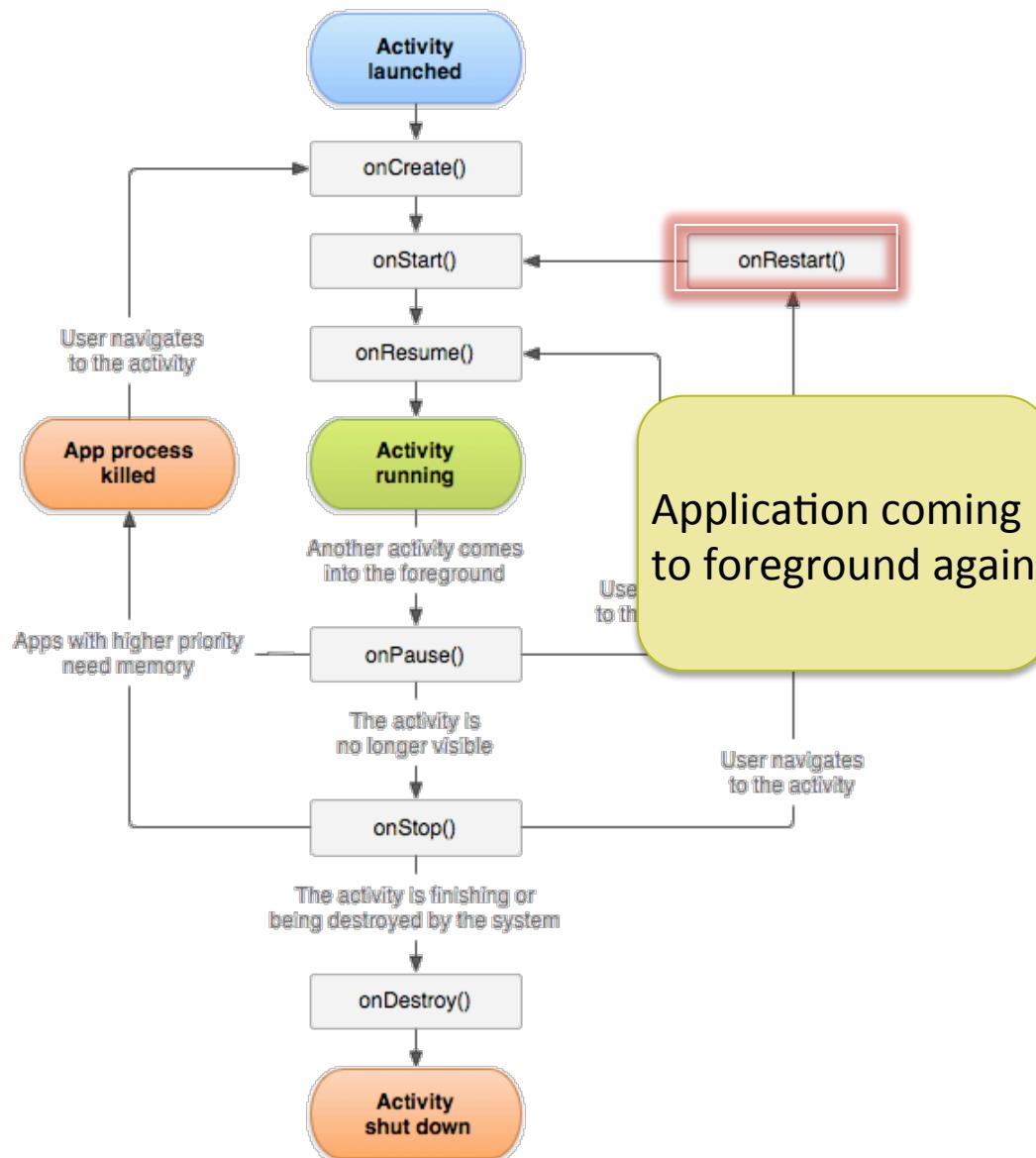
Example



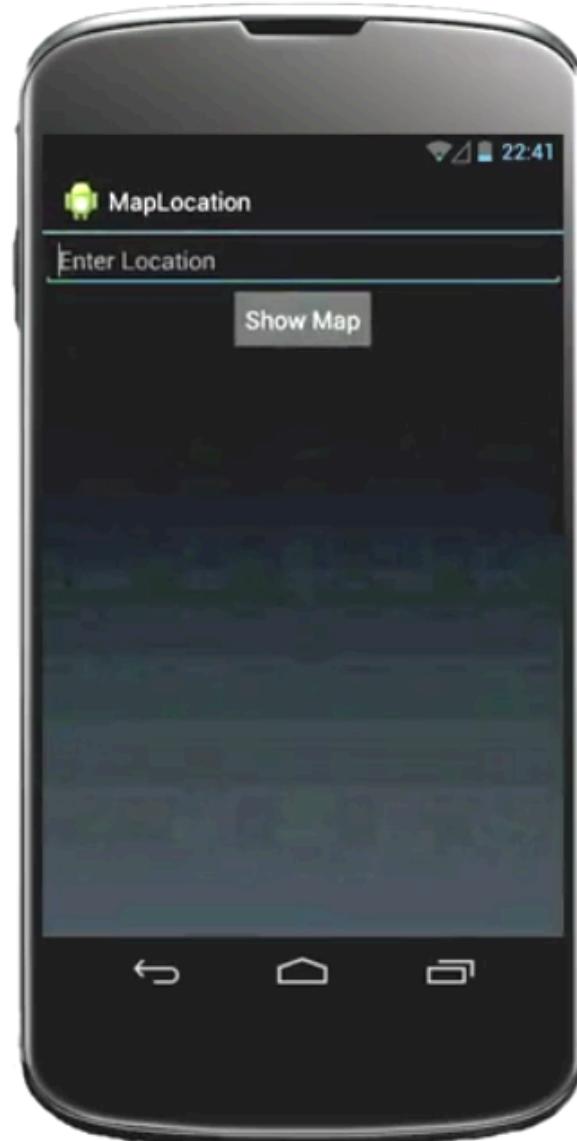
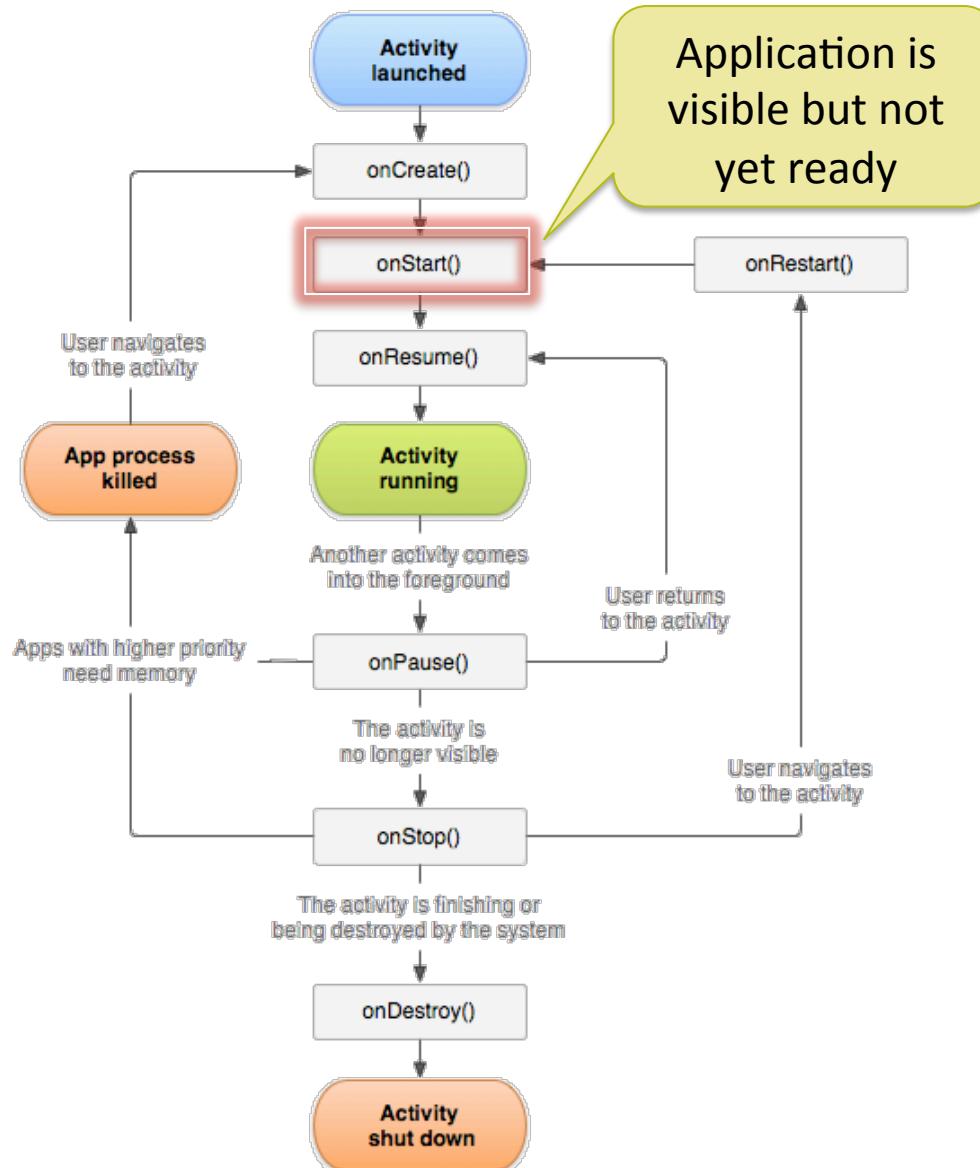
Example



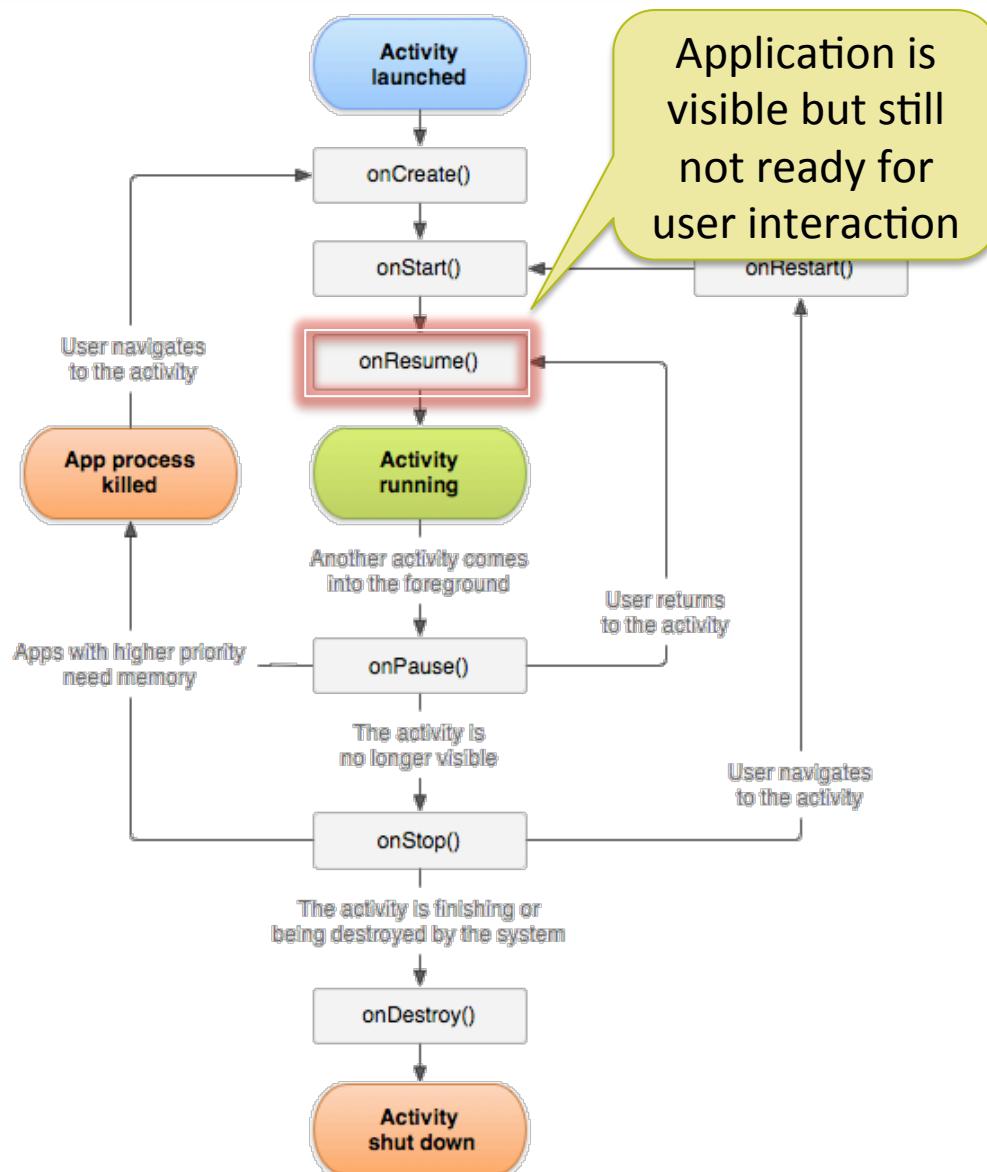
Example



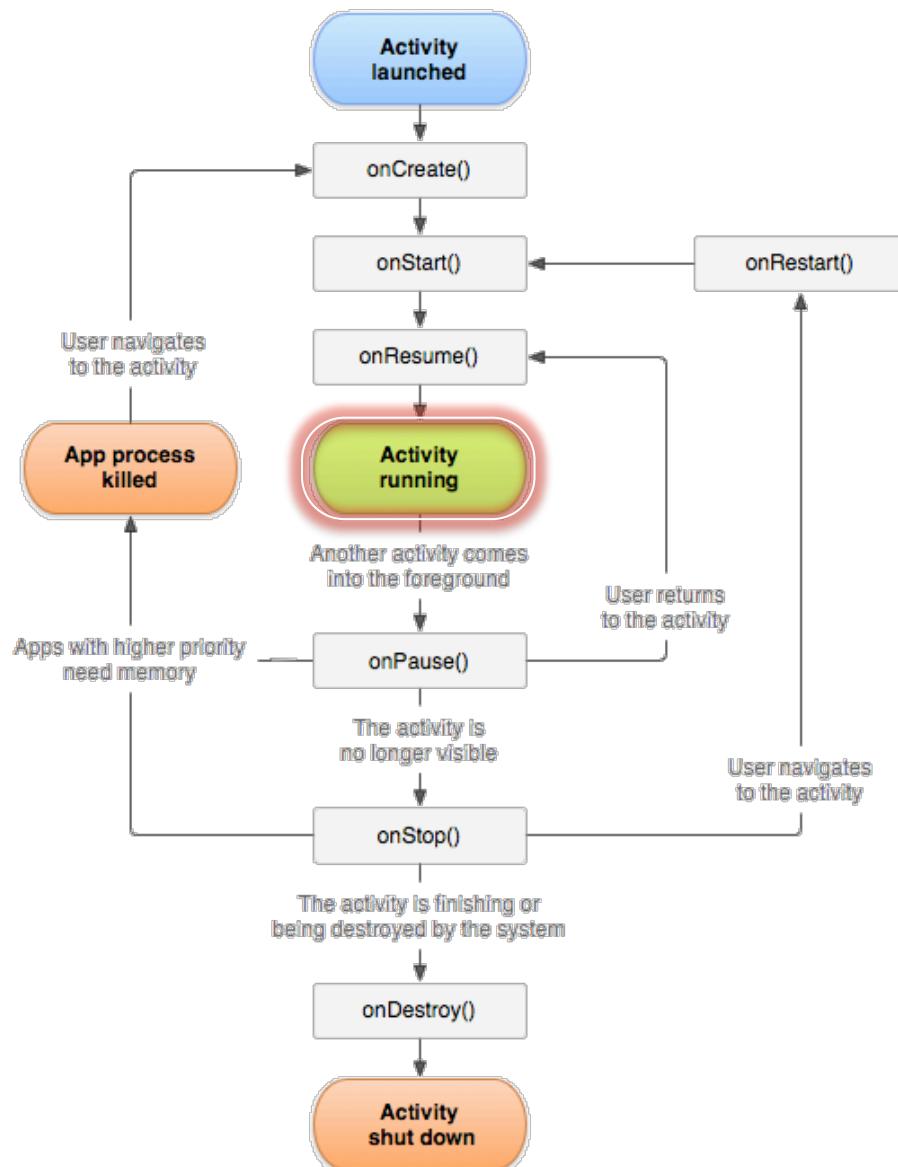
Example



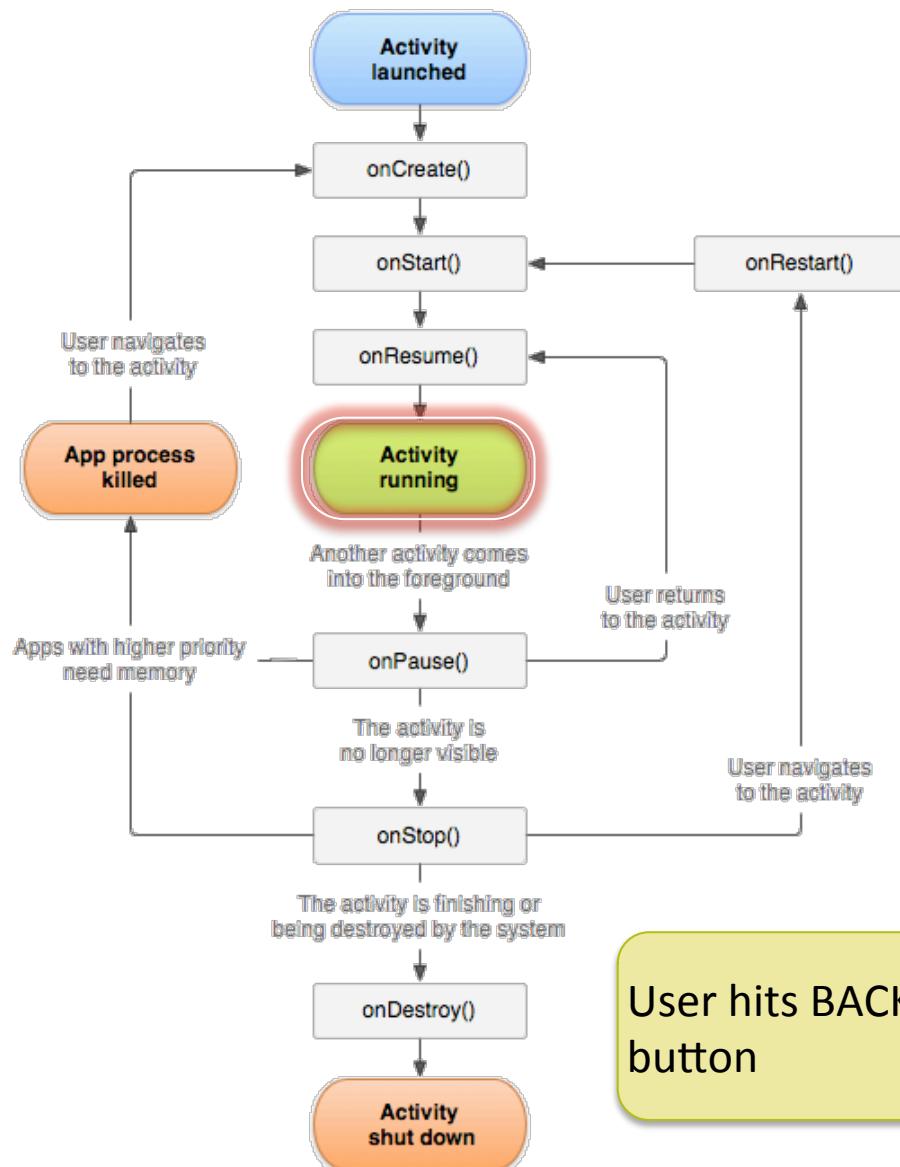
Example



Example

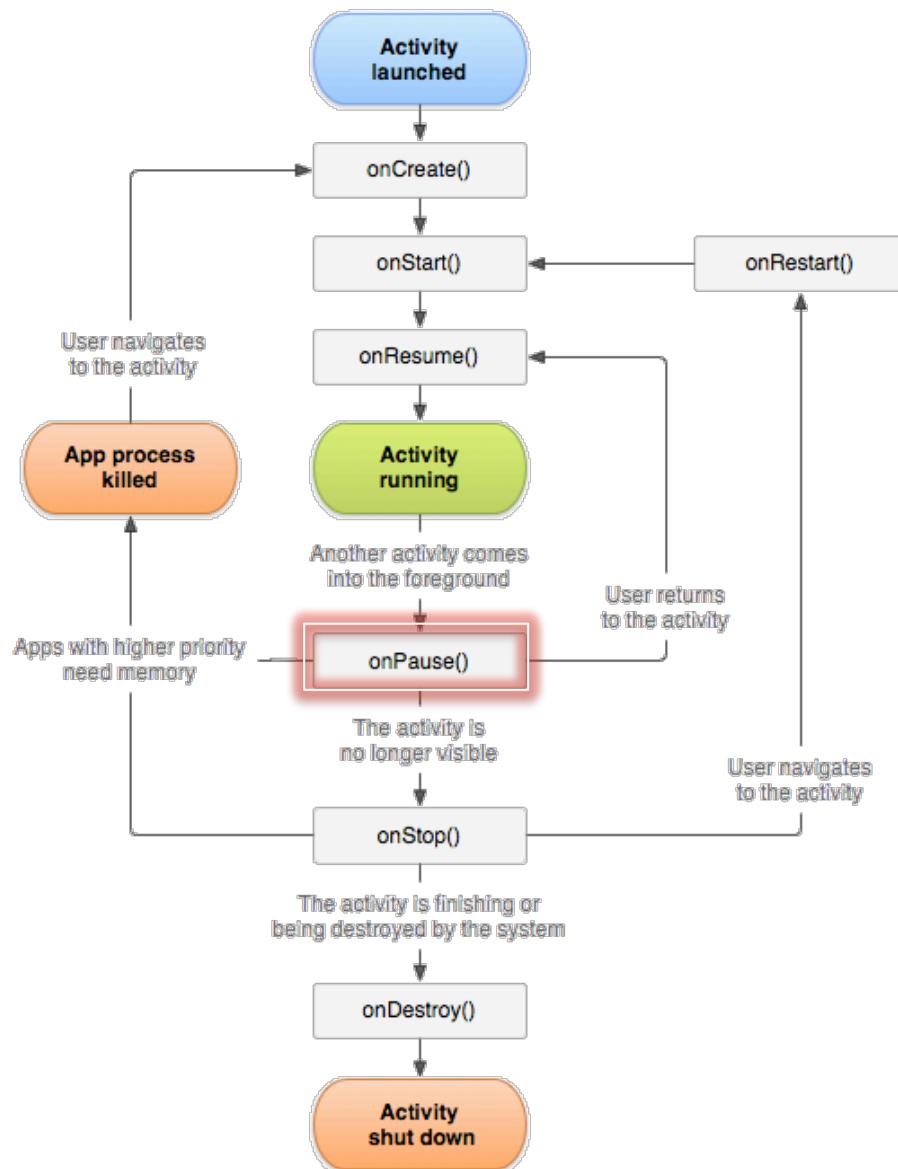


Example

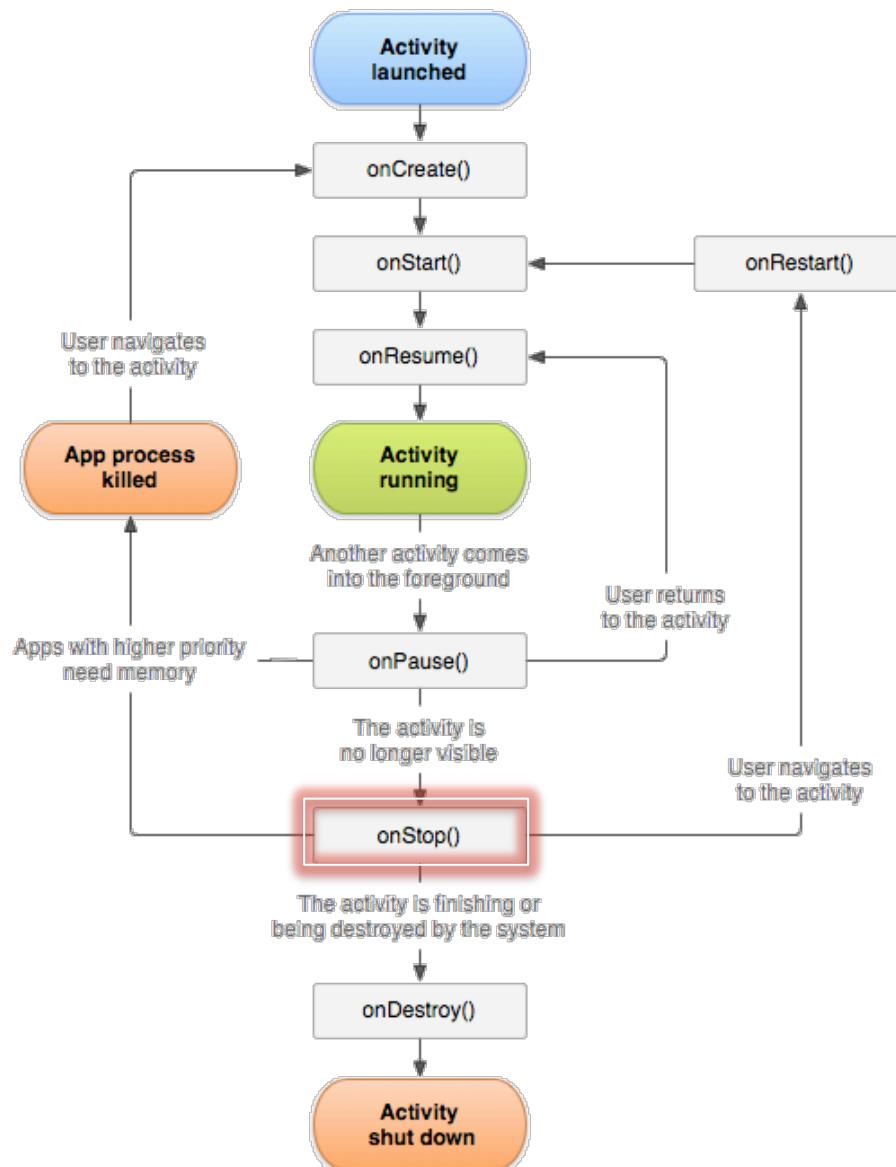


User hits BACK button

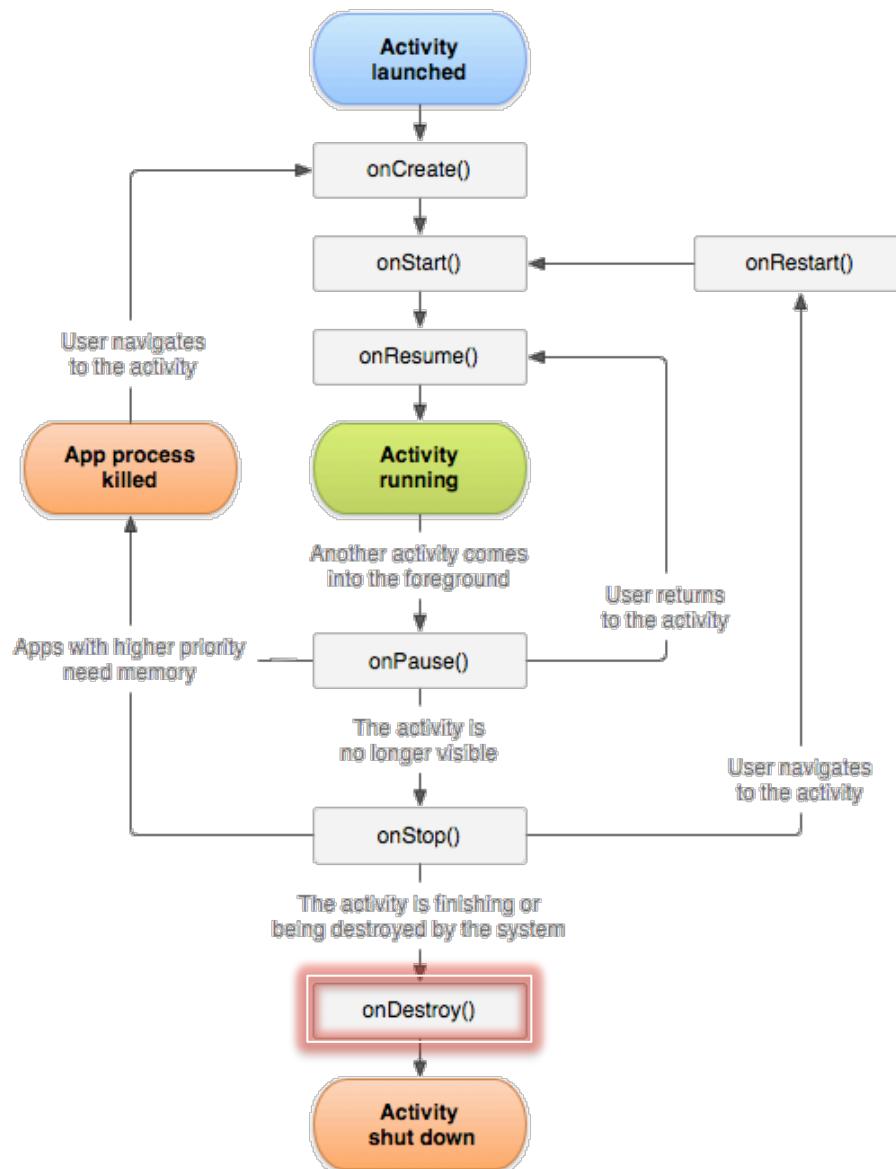
Example



Example



Example



Activity lifecycle

- Android attempts to keep application process alive for **as long as possible**.
- It will kill **less important processes before** killing more important processes

Order of importance

1. The **foreground activity** (the activity at the top of the screen)
Only as a last resort (ie application hanging)

Activity lifecycle

- Android attempts to keep application process alive for as long as possible.
- It will kill less important processes before killing more important processes

Order of importance

1. The **foreground activity** (the activity at the top of the screen)
Only as a last resort (ie application hanging)
2. A **visible activity** (eg. sitting behind a foreground dialog)
Not killed unless that is required to keep the foreground activity running.

Activity lifecycle

- Android attempts to keep application process alive for as long as possible.
- It will kill less important processes before killing more important processes

Order of importance

1. The **foreground activity** (the activity at the top of the screen)
Only as a last resort (ie application hanging)
2. A **visible activity** (eg. sitting behind a foreground dialog)
Not killed unless that is required to keep the foreground activity running.
3. A **background activity** (not visible to the user and paused)
Safely killed whenever memory is needed

Activity lifecycle

- Android attempts to keep application process alive for as long as possible.
- It will kill less important processes before killing more important processes

Order of importance

1. The **foreground activity** (the activity at the top of the screen)
Only as a last resort (ie application hanging)
2. A **visible activity** (eg. sitting behind a foreground dialog)
Not killed unless that is required to keep the foreground activity running.
3. A **background activity** (not visible to the user and paused)
Safely killed whenever memory is needed
4. An **empty process** (no activities, services or Broadcast Receivers)
Always killed → all background operation must always be associated to services