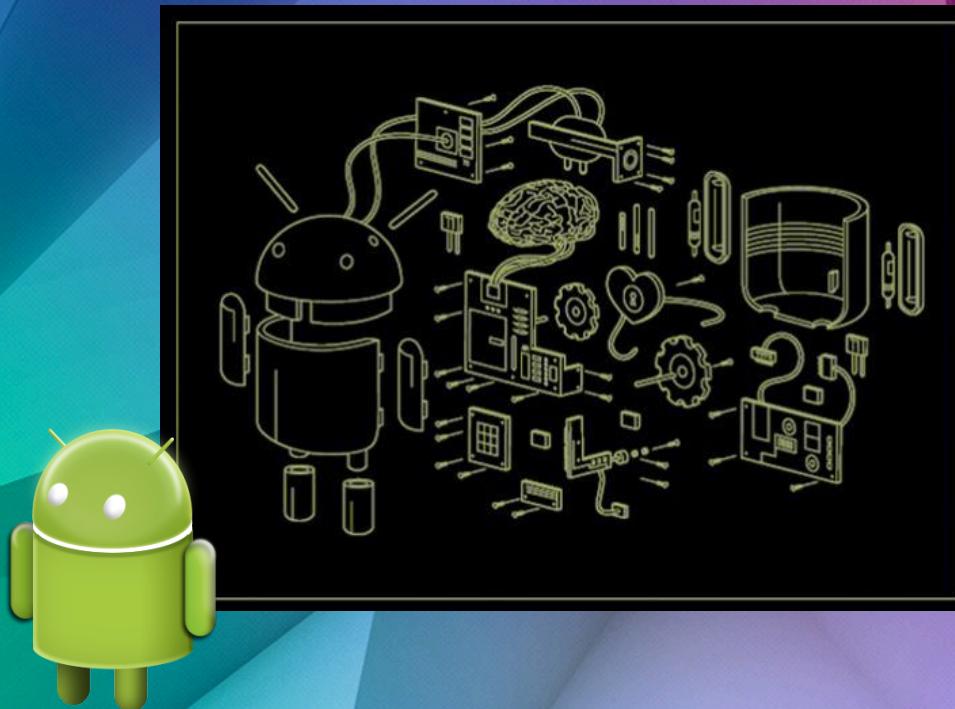


THE ANATOMY OF AN APP



Outline

- Project structure
- Manifest
- Build system
- Core components
- Application
- Activity
- Development Flow





PROJECT STRUCTURE



The screenshot shows the Android Studio interface with the 'Project' tab selected in the top-left corner. The left pane displays the project structure, including the app module with its build, libs, src (containing androidTest and main), res (with drawable, layout, menu, mipmap folders, and values/strings.xml), and various build-related files like .gitignore, app.iml, build.gradle, and proguard-rules.pro. The right pane shows the code editor with the 'AndroidManifest.xml' file open. The manifest file defines a package 'com.leafcastelabs.lifecycletest' with an application component containing an activity named 'MainActivity'. The activity is set to handle the main intent and is categorized as a launcher. The code editor has syntax highlighting and line numbers.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.leafcastelabs.lifecycletest" >

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="LifeCycleTest"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="LifeCycleTest" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Project Structure (Project view)



The screenshot shows the Android Studio interface with the project 'LifeCycleTest' open. The left sidebar displays the project structure, including the app module with its manifest, Java files (MainActivity), resources (res folder with drawable, layout, menu, mipmap, values), and Gradle scripts. The right pane shows the code editor for 'AndroidManifest.xml'. The XML code defines the application package, MainActivity, and its intent filter for the launcher.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.leafcastlelabs.lifecycletest" >

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="LifeCycleTest"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="LifeCycleTest" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Project Structure (Android view)



Modules

- **Android Application Modules**
- Test Modules
- Library Modules
- App Engine Modules (**Cloud backend**)



App Module Files

app/

 build/

 libs/

 src/

 androidTest/

 main/

 java/

 <package name>/

 ... code files ...

 res/

 ... resources files and xml ...

 manifest.xml

 build.gradle

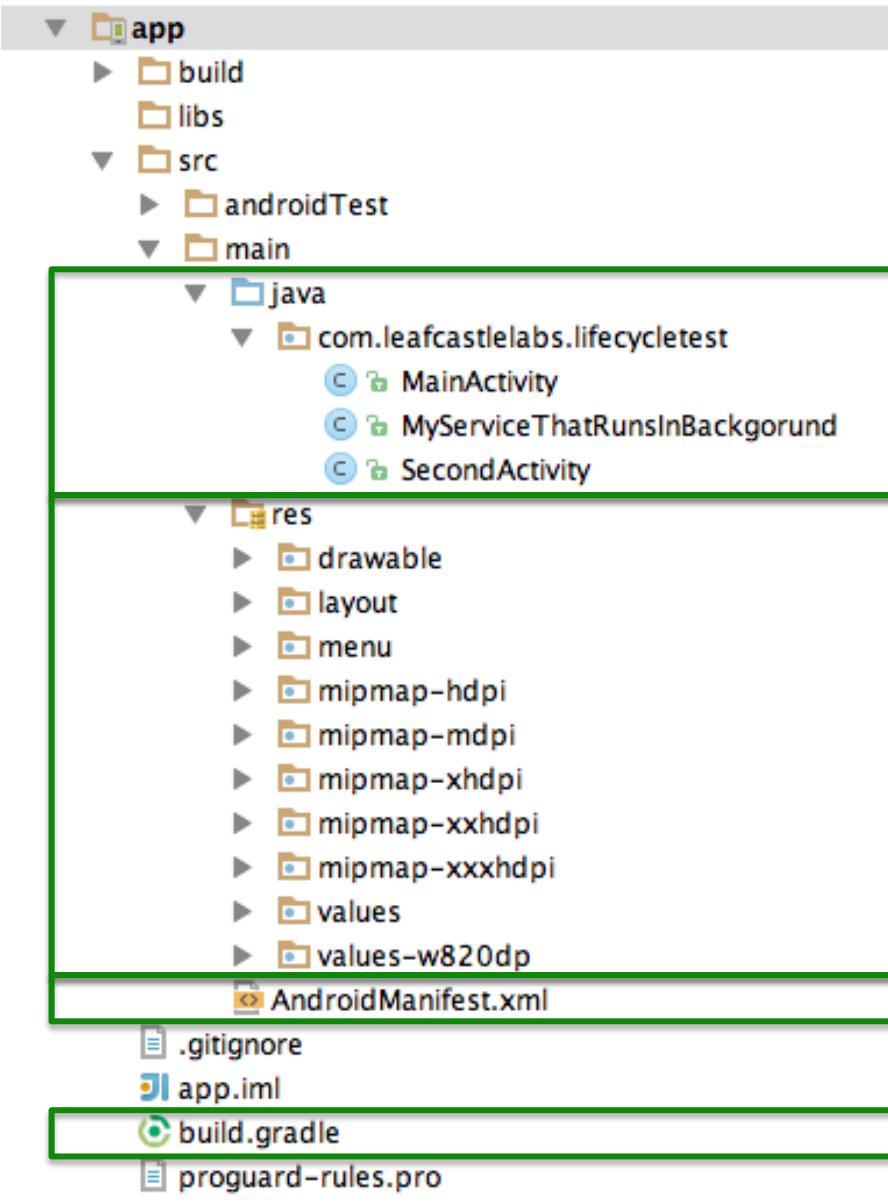


App Module Files

Code

Resources

Manifest
Gradle





THE MANIFEST



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.leafcastlelabs.lifecycletest" >
4
5      <uses-permission android:name="android.permission.INTERNET" />
6      <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
7
8      <application
9          android:allowBackup="true"
10         android:icon="@mipmap/ic_launcher"
11         android:label="@string/app_name"
12         android:theme="@style/AppTheme" >
13             <activity
14                 android:name=".MainActivity"
15                 android:label="@string/app_name" > This is the app name
16                 <intent-filter>
17                     <action android:name="android.intent.action.MAIN" />
18
19                     <category android:name="android.intent.category.LAUNCHER" />
20             </intent-filter>
21         </activity>
22         <activity
23             android:name=".SecondActivity"
24             android:label="@string/title_activity_second" >
25         </activity>
26         <service
27             android:name=".MyServiceThatRunsInBackgorund"
28             android:enabled="true"
29             android:exported="true" >
30         </service>
31     </application>
32
33 </manifest>
34
```

Manifest.xml



Core Components

- Application There will only be one Application per. app
- Activity "Unlimited amounts" of these
- Intents and Intent-Filters Like Signals and Slots
- Service Can work in the background
- Content Provider
- Broadcast Receiver
- Widget



Manifest Elements

<action>	<permission>
< activity >	<permission-group>
<activity-alias>	<permission-tree>
< application >	< provider >
<category>	< receiver >
<data>	< service >
<grant-uri-permission>	<supports-screens>
<instrumentation>	<uses-configuration>
<intent-filter>	<uses-feature>
< manifest >	<uses-library>
<meta-data>	<uses-permission>
	<uses-sdk>

<http://developer.android.com/guide/topics/manifest/manifest-intro.html>





THE BUILD SYSTEM





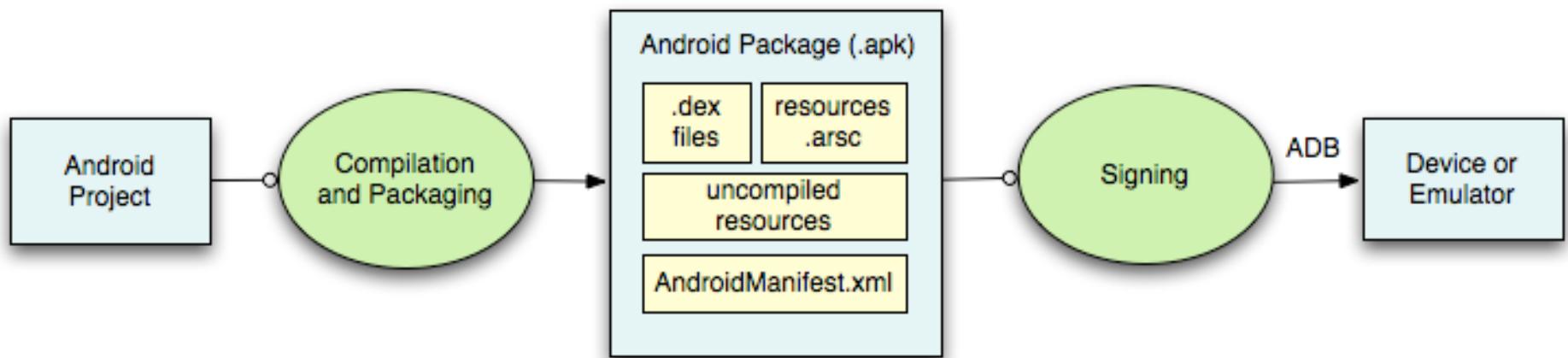
The app can be build with another compiler with the manifest file



```
1 apply plugin: 'com.android.application'  
2  
3     android {  
4         compileSdkVersion 22  
5         buildToolsVersion "22.0.1"  
6  
7             defaultConfig {  
8                 applicationId "com.leafcastlelabs.lifecycletest"  Package name -> Has to be Unique  
9                 minSdkVersion 14  
10                targetSdkVersion 22  Here SDK versions can be edited -> Target should always be the newest version.  
11                versionCode 1  
12                versionName "1.0"  
13            }  
14            buildTypes {  
15                release {  
16                    minifyEnabled false  
17                    proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'  
18                }  
19            }  
20        }  
21  
22        dependencies {  
23            compile fileTree(dir: 'libs', include: ['*.jar'])  
24            compile 'com.android.support:appcompat-v7:22.0.0'  
25        }  
26    }
```



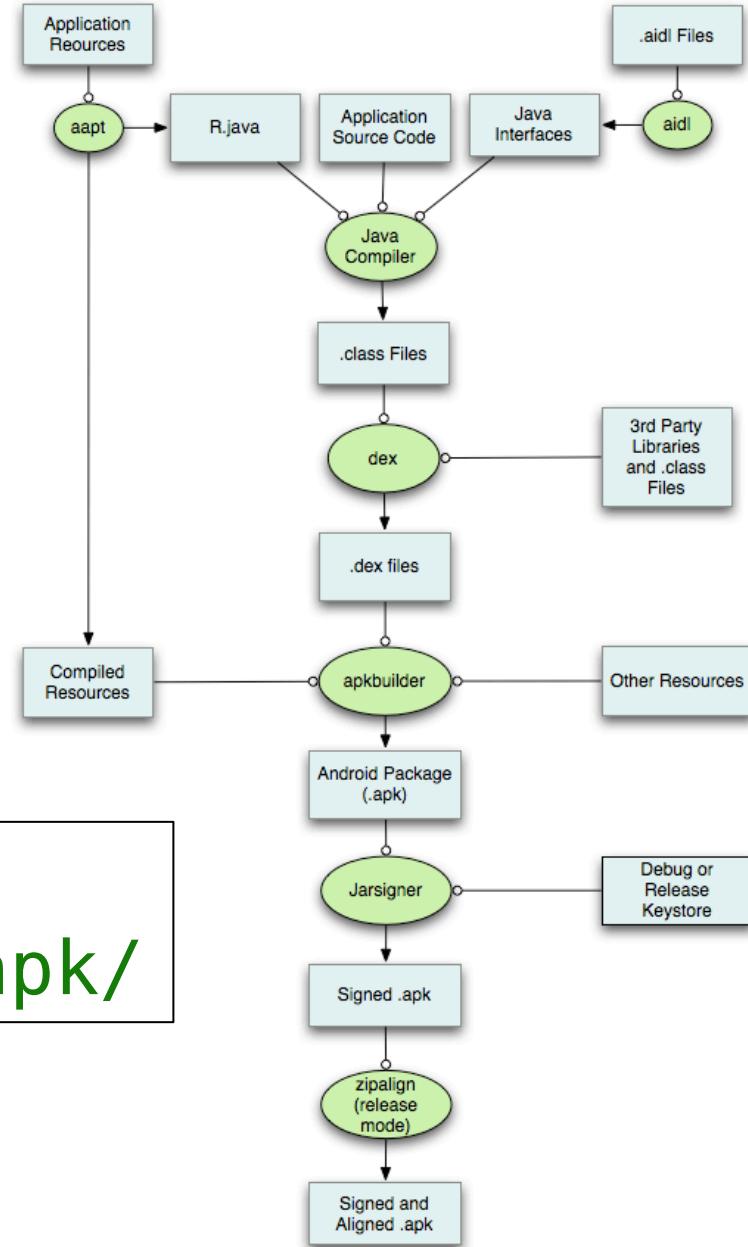
Building Apps



<https://developer.android.com/tools/building/>



Build System under the Hood



APK files output:
app/build/outputs/apk/

Source: <http://developer.android.com/>



Resources in `res/`

- Layouts `layout/`
- Simple values e.g. strings, string arrays, dimensions `values/`
- App icon
- Images `mipmap/`
- Menus `drawable/`
- Colors `menu/`
- Animations `color/`
- Arbitrary raw assets e.g. media files `anim/`
- Misc. XML `raw/`
- Misc. XML `xml/`



App Icon

- Support various dpi levels
- New standard is: **mipmap**
- Can be generated by e.g. **Android Asset Studio**



<http://romannurik.github.io/AndroidAssetStudio/icons-launcher.html>



ProGuard

“shrinks, optimizes, and **obfuscates** your code by **removing unused code** and **renaming** classes, fields, and methods with semantically obscure names.

The result is a **smaller sized .apk file** that is **more difficult to reverse engineer**”

<http://developer.android.com/tools/help/proguard.html>



Project and Manifest





ACTIVITIES



Tasks and Activities

- Task

“A task is a **collection of activities** that users interact with when **performing a certain job**”

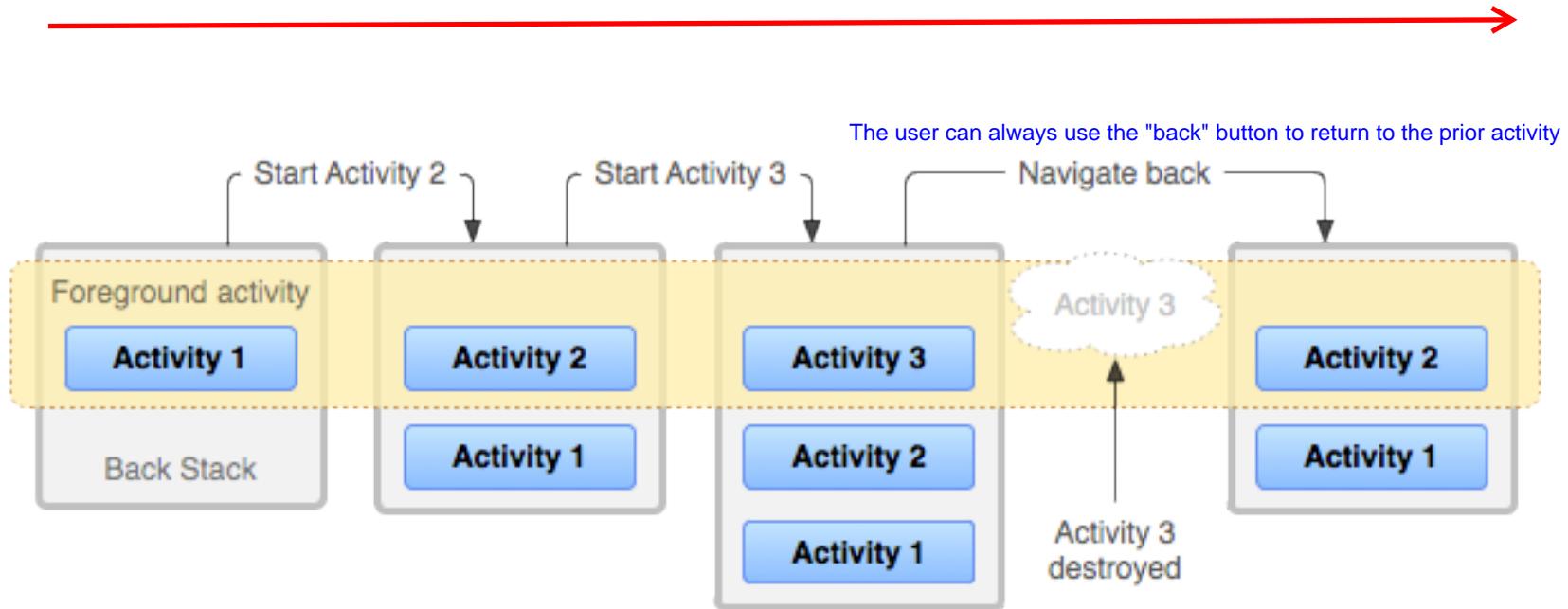
- Activity

“Each activity should be designed around a **specific kind of action** the user can perform and can start other activities”

<http://developer.android.com/guide/components/tasks-and-back-stack.html>



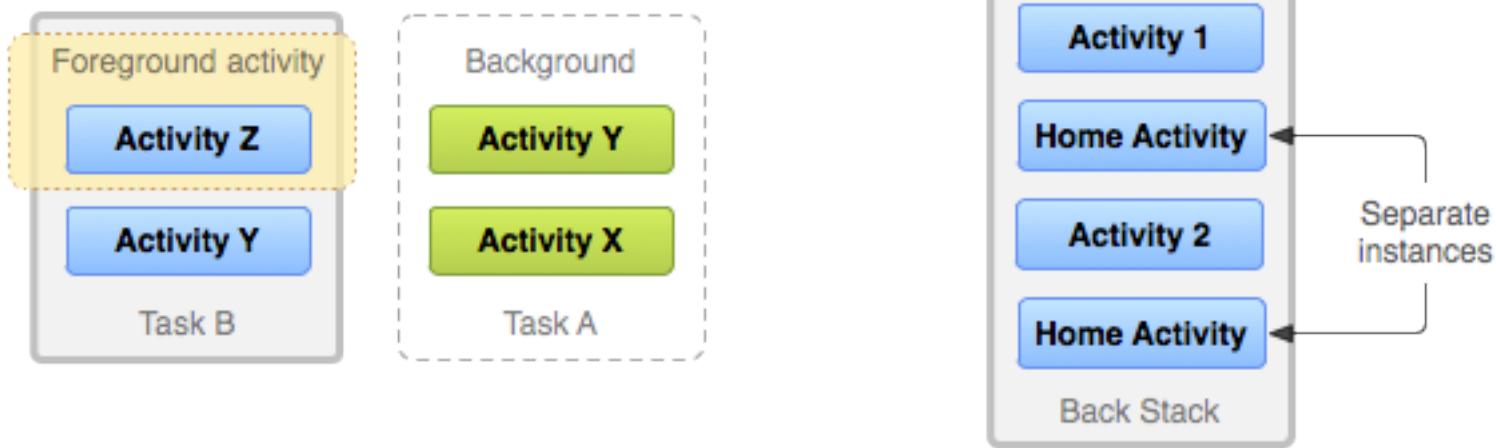
Back Stack



<http://developer.android.com/guide/components/tasks-and-back-stack.html>



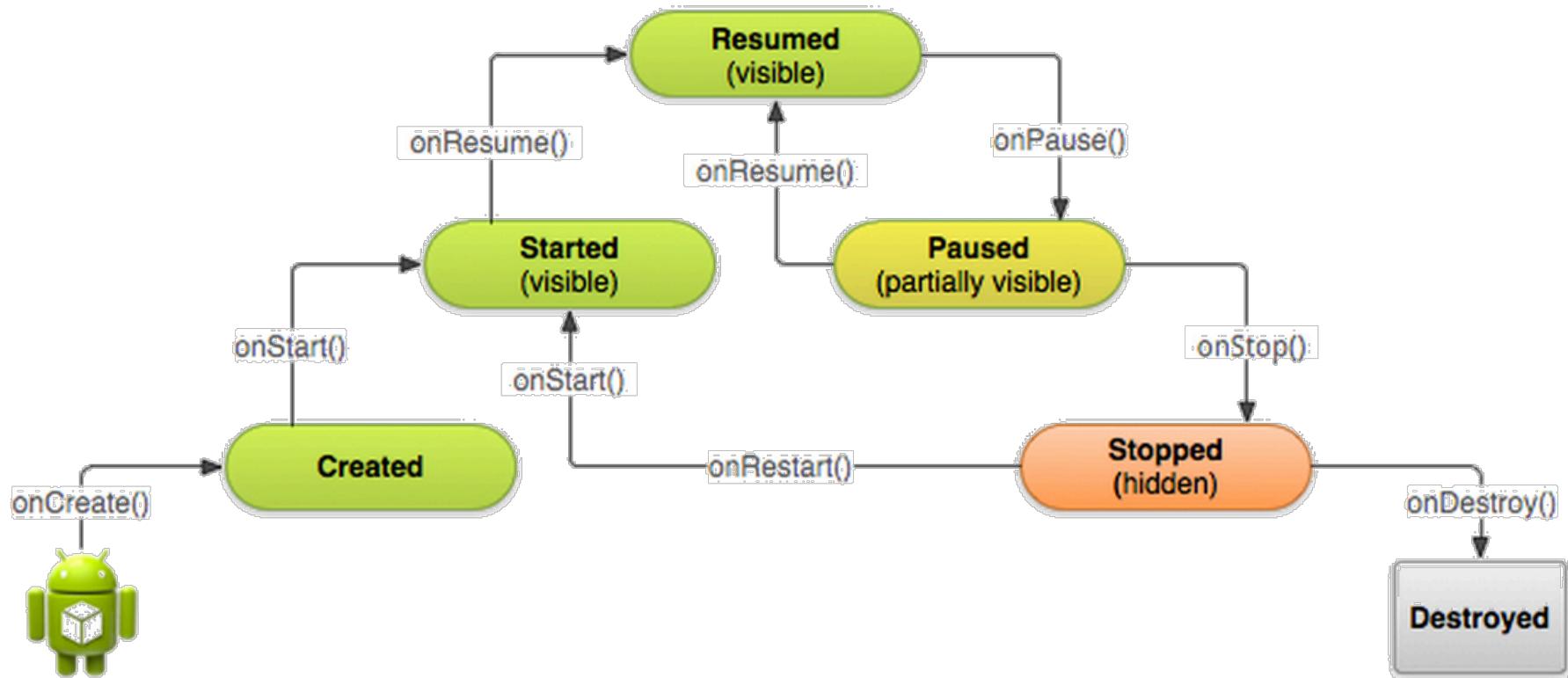
Multitasking vs Multi-instance



<http://developer.android.com/guide/components/tasks-and-back-stack.html>



Activity Life Cycle Diagram



<http://developer.android.com/>

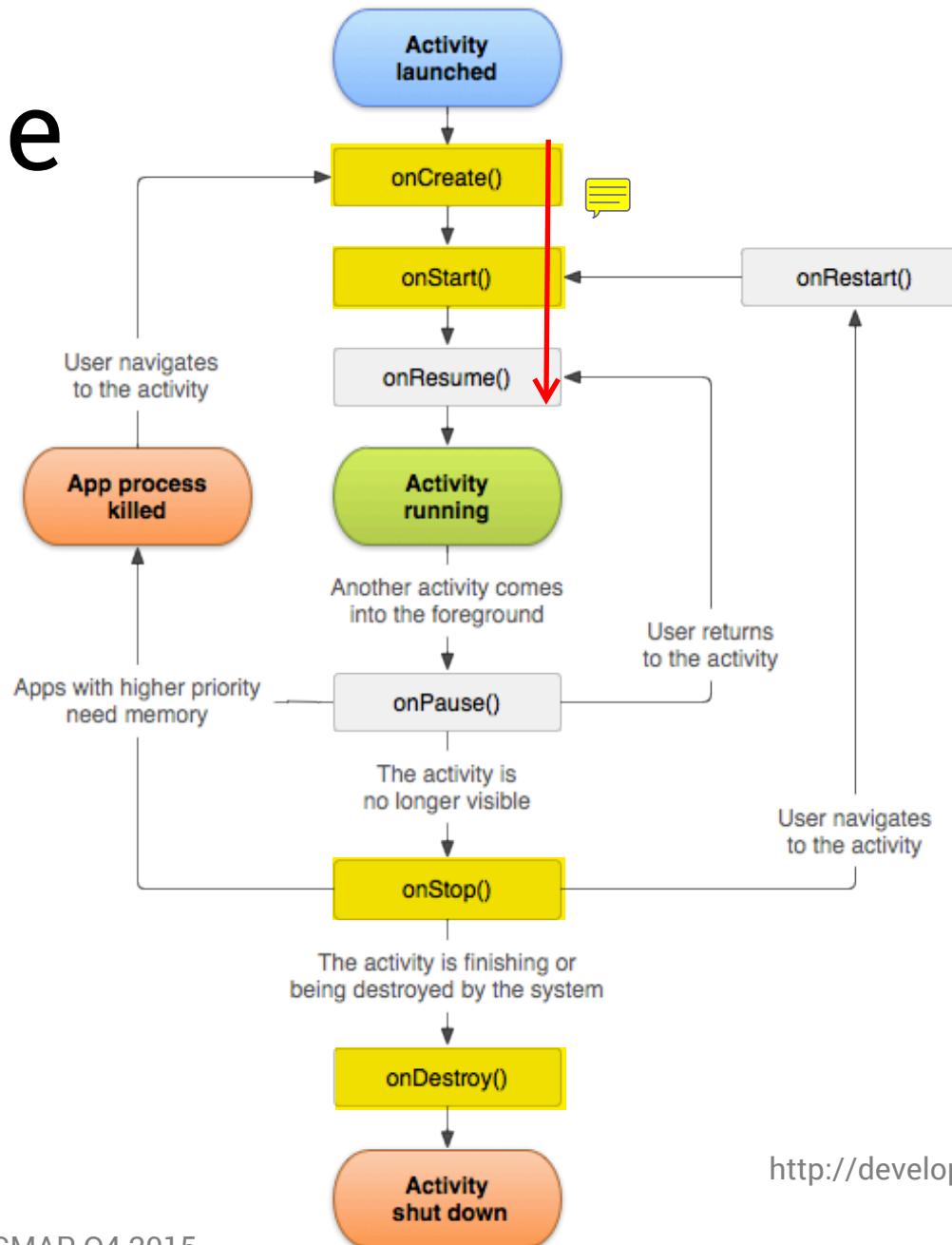


“Paranoid Android”

- Android is in charge!
- Once your activity is not in the foreground it can potentially be shutdown e.g. to free system memory
- Android can also intervene if your app is too slow on the main thread / UI



Life Cycle



<http://developer.android.com/>



Activity Life Cycle Methods

- @Override these methods to handle the main state changes
 - onCreate()
 - onDestroy()
 - onStart()
 - onStop()
 - onPause()
 - onResume()



savedInstanceState

- Activities used a **Bundle** to persist information about the instance state
- Parsed to the **Activity** on creation as the **Bundle savedInstanceState** parameter
- Android does a lot of saving automatically, but ultimately it is **YOUR responsibility** as a programmer
 - Think of the user experience!





APPLICATION



Application

- An app always has a “root level” singleton from the base class **Application**
- Activities can share global data and state by extending this, but in most cases it is more clean to use intents and/or static singletons



Context

- “**Interface to global information about an application environment.** This is an abstract class whose implementation is provided by the Android system. It **allows access to application-specific resources and classes, as well as up-calls for application-level operations** such as launching activities, broadcasting and receiving intents, etc.”

<http://developer.android.com/reference/android/content/Context.html>

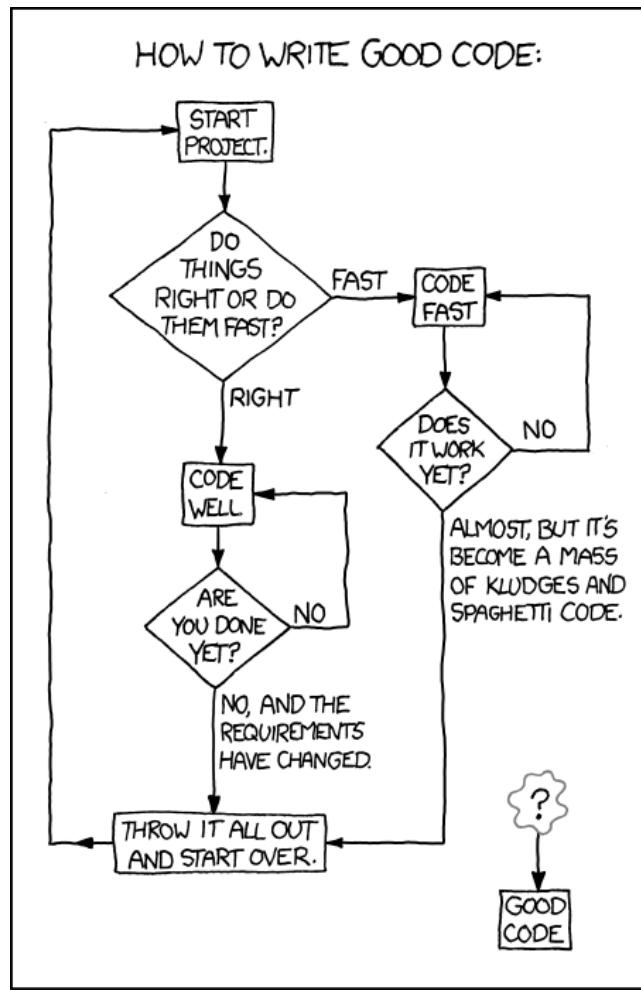




DEVELOPMENT FLOW



Android Development Flow



<https://xkcd.com/844/>



Pro Tip

- Beware of the tools
 - Features like code completion and project wizards are awesome for general productivity
 - But sometimes they “outsmart” you and do things without you noticing
 - Importing the wrong library (version)
 - Using unexpected derived classes
 - Can be really hard to track down!
- The code will never lie!
 - (comments might)



When Things go Wrong...

- ...and they will!
 - LogCat
 - Toasts
 - Debugger
 - DDMS / ADM





Debugger



Logging and Debugging



EXERCISE 1

