MIT
Global Startup Labs
México 2013

http://aiti.mit.edu
http://gsl.mit.edu
Coming Soon!

Lección 1 – Introducción a Android

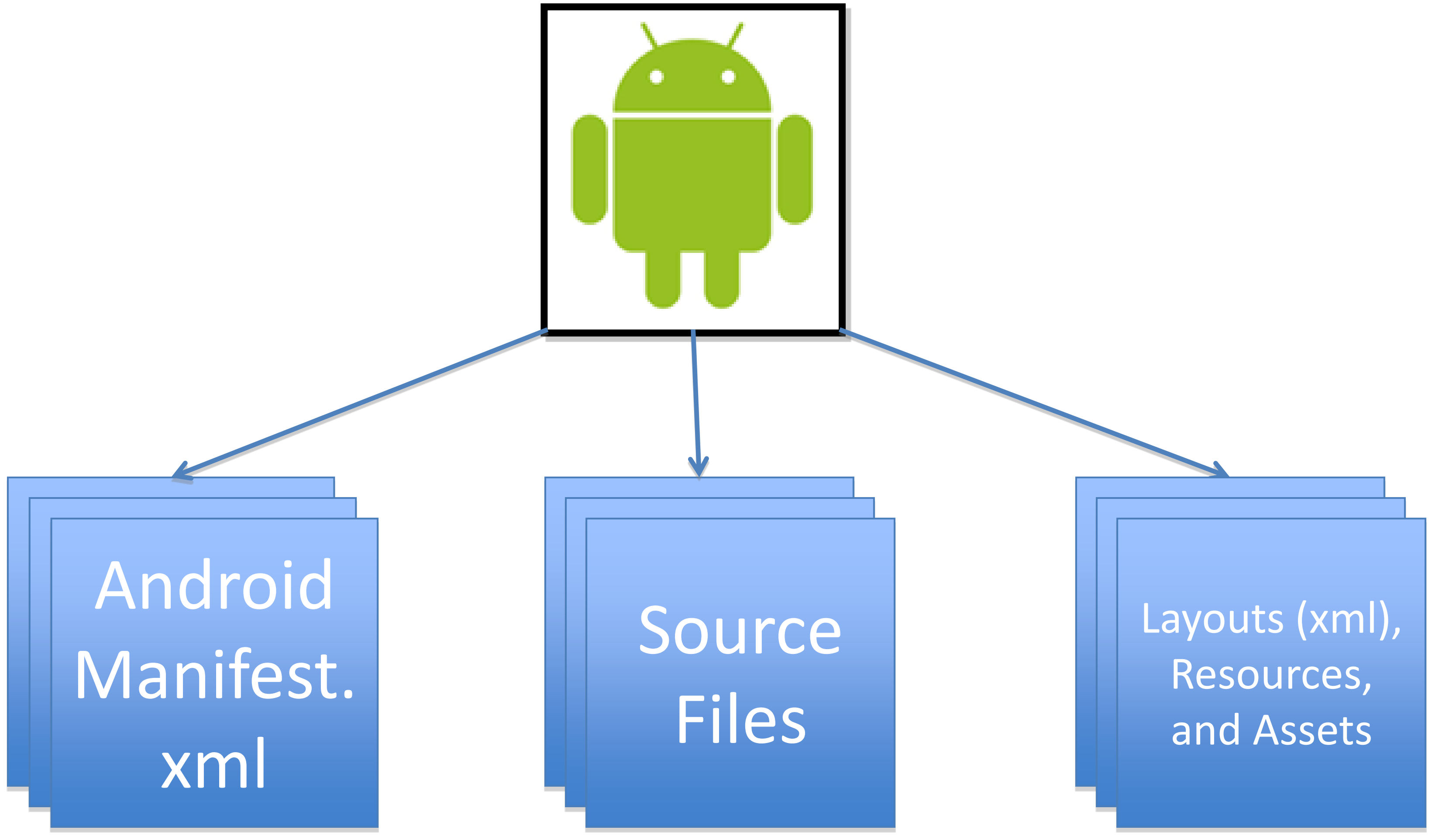
Agenda

- La Plataforma Android
- Estructura de un proyecto Android
- Android Build Process
- Android con Eclipse
- Anatomía de un proyecto Android
- Android Activity Lifecycle

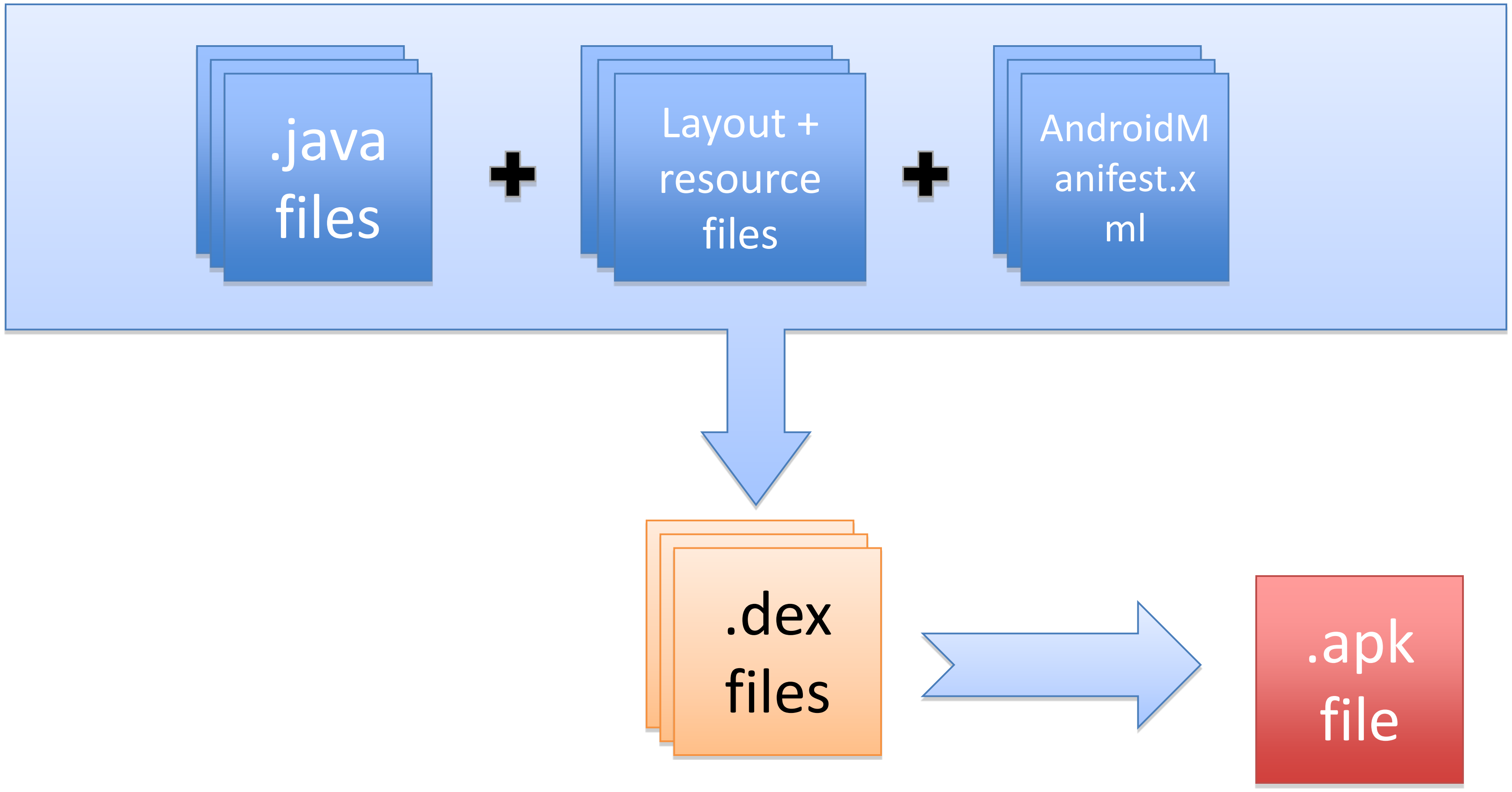
Platforma Android

- Sistema operativo basado en Linux
- Dalvik VM vs. JVM
- Base de datos SQLite (native)
- Aplicaciones integradas (Home, Contacts, Phone, Browser, Voice Recognition, Camera)
- Componentes: GPS, WiFi, Camera, Audio/Video recording + playback, Sensors

Estructura de un Android App



Android App Build Process

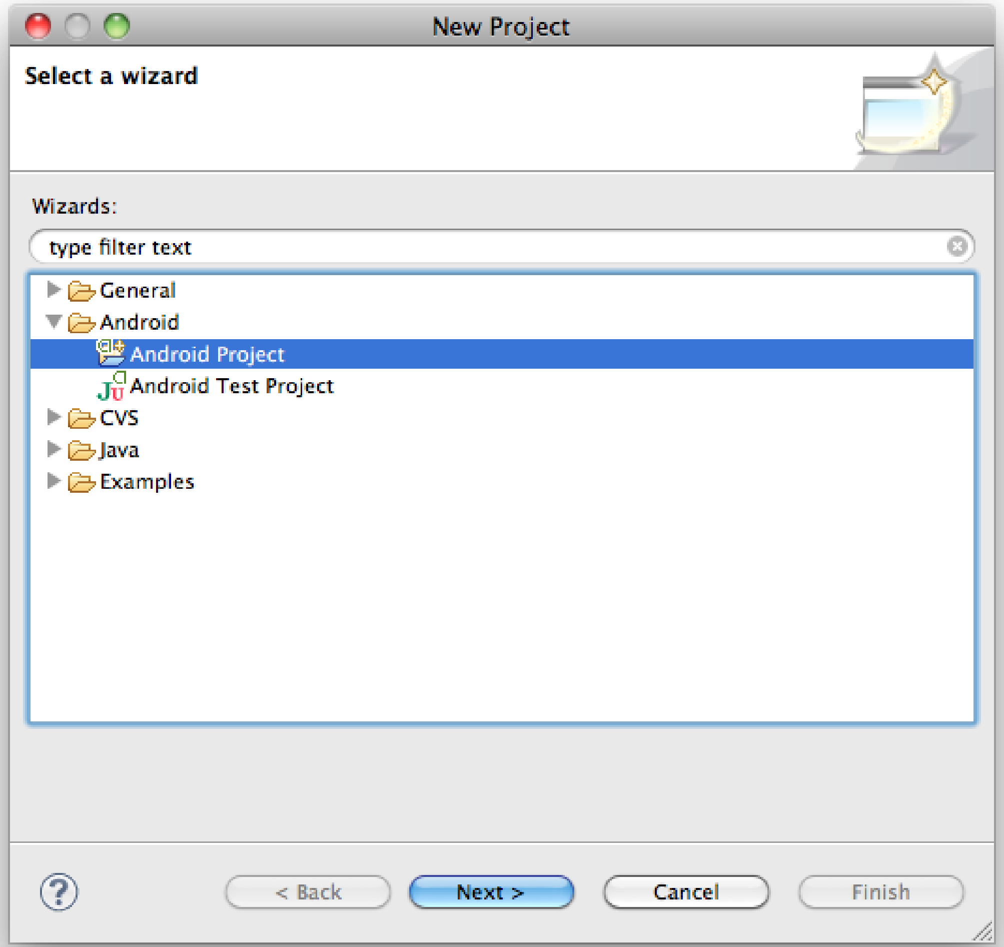


Android con Eclipse

Requisitos:

- Eclipse
  http://www.eclipse.org/downloads/
- Android SDK
  http://developer.android.com/sdk/index.html
- Eclipse ADT Plug-in
  http://developer.android.com/sdk/eclipse-adt.html
- Android API (Android 3.2, 4.0.3, etc.)
  - Latest version is Android v4.0.3 (API 15)
- USB Driver
- Emulator (Android Virtual Device)

Creating Android Projects, Step 1:


New Project

Select a wizard

Wizards:

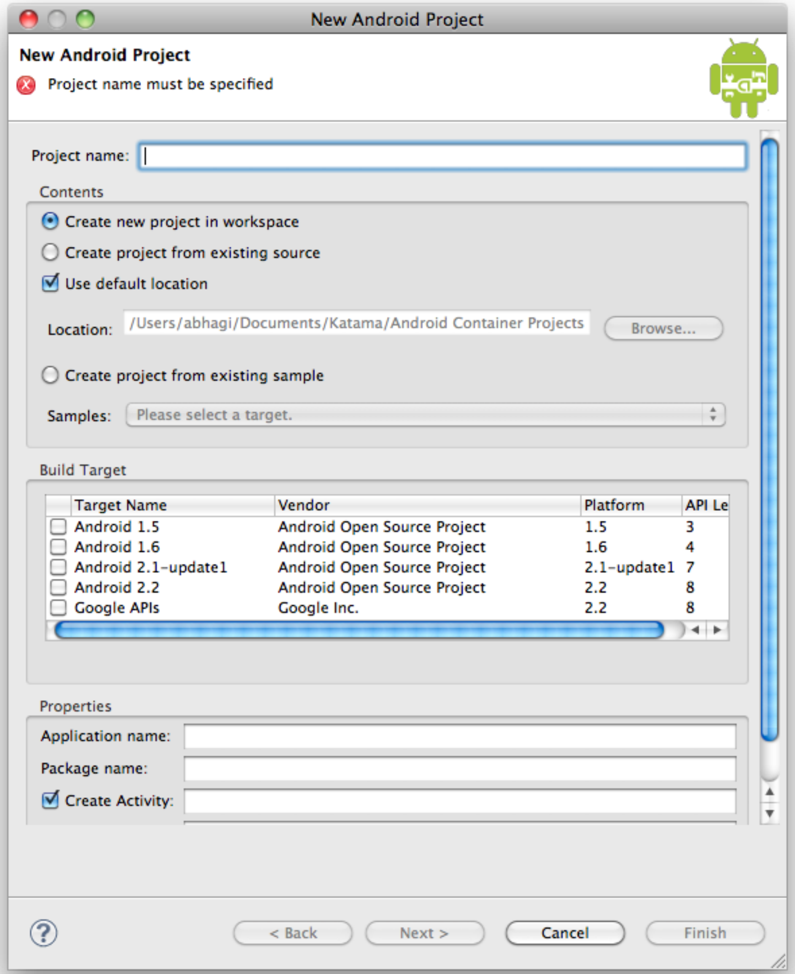
type filter text

- General
- Android
  - Android Project
  - Android Test Project
- CVS
- Java
- Examples

< Back | Next > | Cancel | Finish

Creating Android Projects, Step 2:

- – Project Name
- – Build Target
- – Application name
- – Package name
- – Activity name
- – Min SDK Version



New Android Project

New Android Project

Project name must be specified

Project name:

Contents

(●) Create new project in workspace

( ) Create project from existing source

[✓] Use default location

Location: /Users/abhagi/Documents/Katama/Android Container Projects

( ) Create project from existing sample

Samples: Please select a target.

Build Target

	Target Name	Vendor	Platform	API Le
☐	Android 1.5	Android Open Source Project	1.5	3
☐	Android 1.6	Android Open Source Project	1.6	4
☐	Android 2.1-update1	Android Open Source Project	2.1-update1	7
☐	Android 2.2	Android Open Source Project	2.2	8
☐	Google APIs	Google Inc.	2.2	8

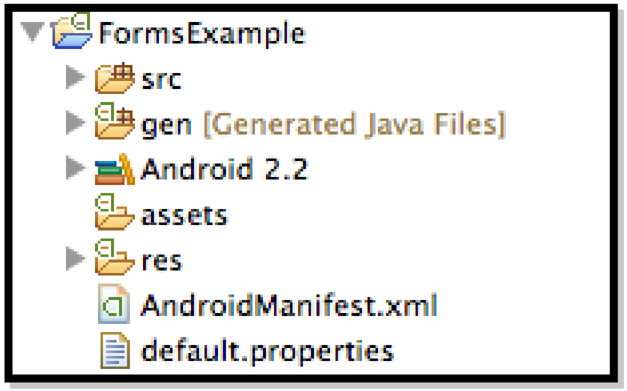
Properties

Application name:

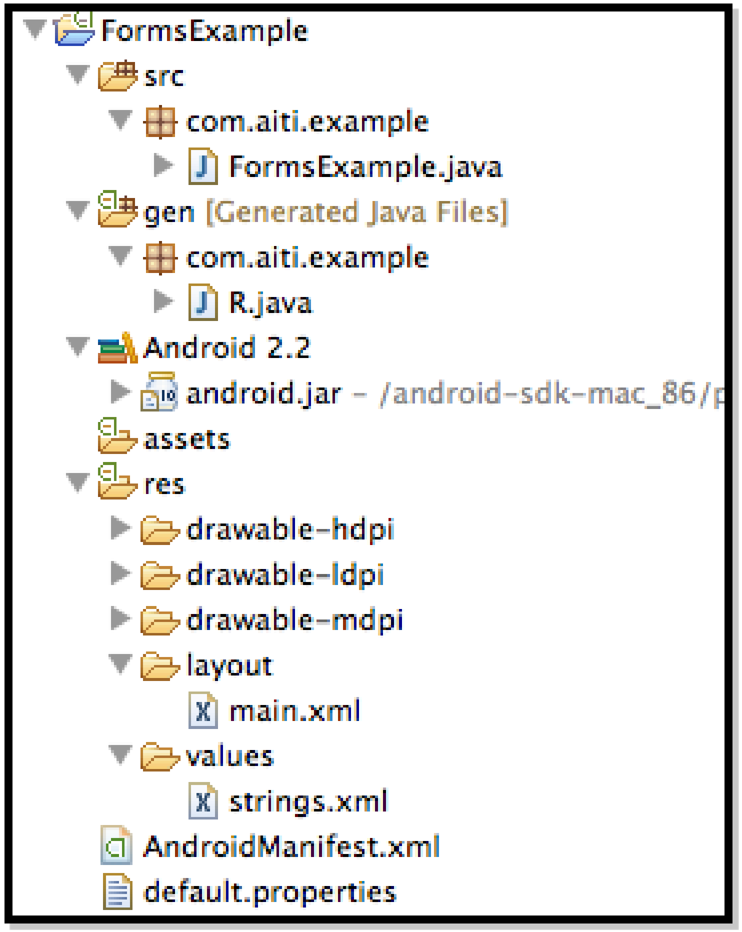
Package name:

[✓] Create Activity:

Anatomy of Android Project (in Eclipse)



```
▼ FormsExample
  ▶ src
  ▶ gen [Generated Java Files]
  ▶ Android 2.2
    assets
  ▶ res
    AndroidManifest.xml
    default.properties
```



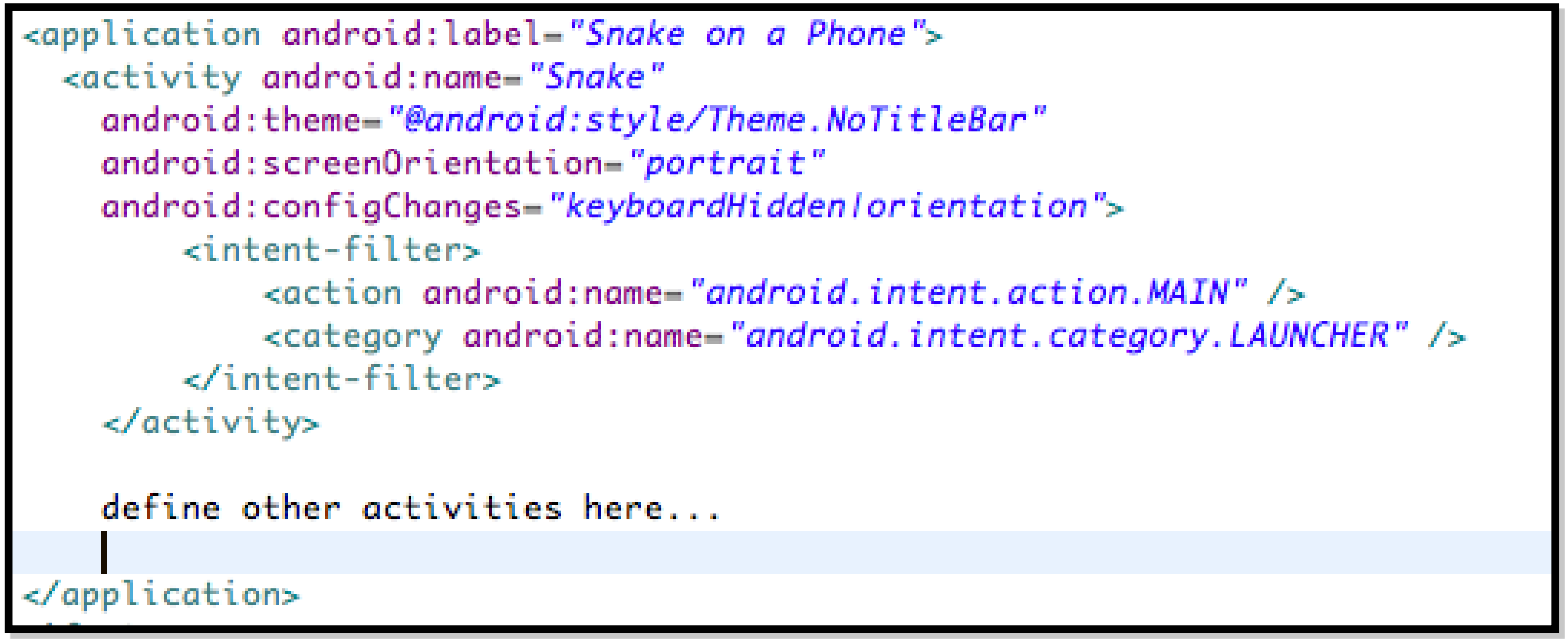
```
▼ FormsExample
  ▼ src
    ▼ com.aiti.example
      ▶ FormsExample.java
  ▼ gen [Generated Java Files]
    ▼ com.aiti.example
      ▶ R.java
  ▼ Android 2.2
    ▶ android.jar - /android-sdk-mac_86/p
    assets
  ▼ res
    ▶ drawable-hdpi
    ▶ drawable-ldpi
    ▶ drawable-mdpi
    ▼ layout
        main.xml
    ▼ values
        strings.xml
    AndroidManifest.xml
    default.properties
```

Actividades Android

- Apps contienen Activitidades
- Cada Actividad es una “página”, un interfaz usuario.
- Actividades están interconectadas, como páginas web contienen navegación
- Cada Actividad contiene un Layout con objectos específicos del View

Actividad Main

Solamente una Actividad puede ser el Main (i.e. el punto de entrada para el usuario)



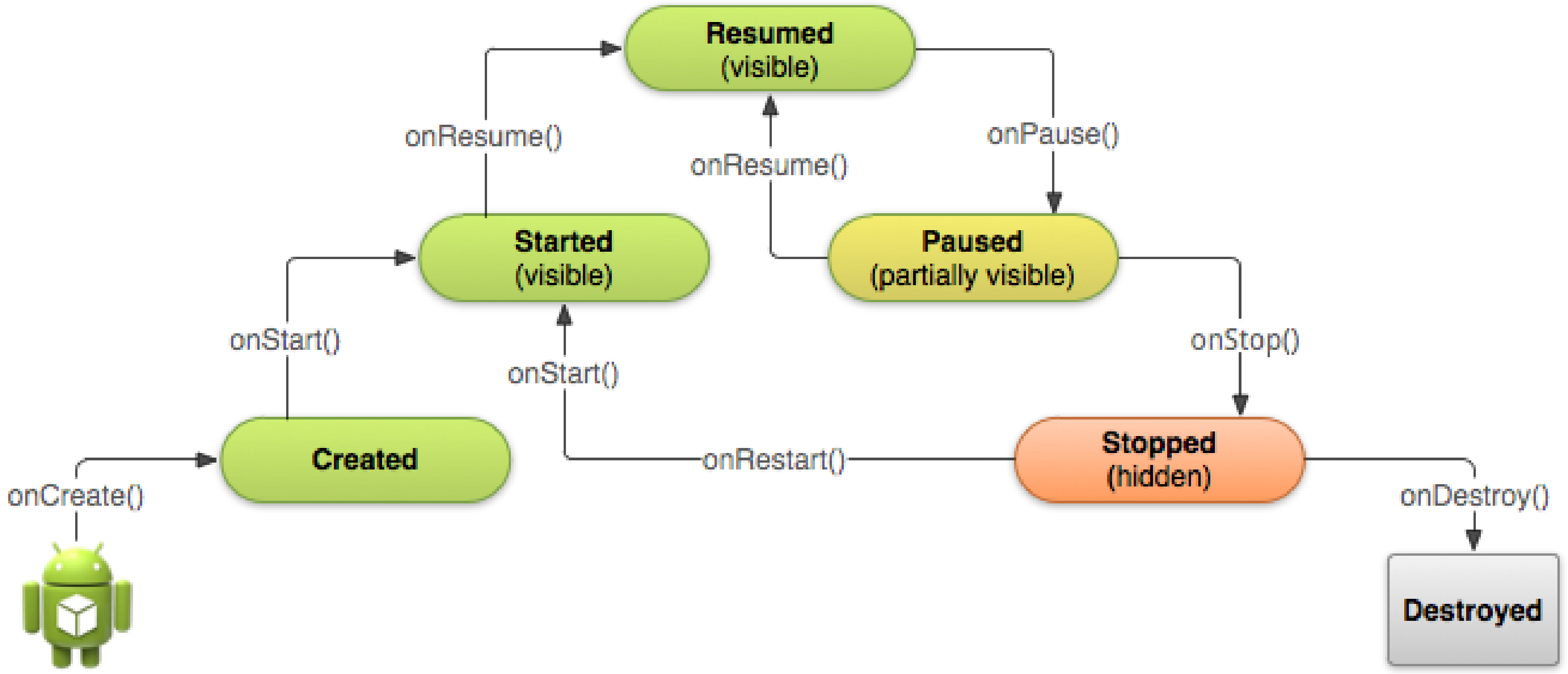
```
<application android:label="Snake on a Phone">
  <activity android:name="Snake"
    android:theme="@android:style/Theme.NoTitleBar"
    android:screenOrientation="portrait"
    android:configChanges="keyboardHidden|orientation">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    define other activities here...

</application>
```

MIT AITI

Basic Activity Lifecycle

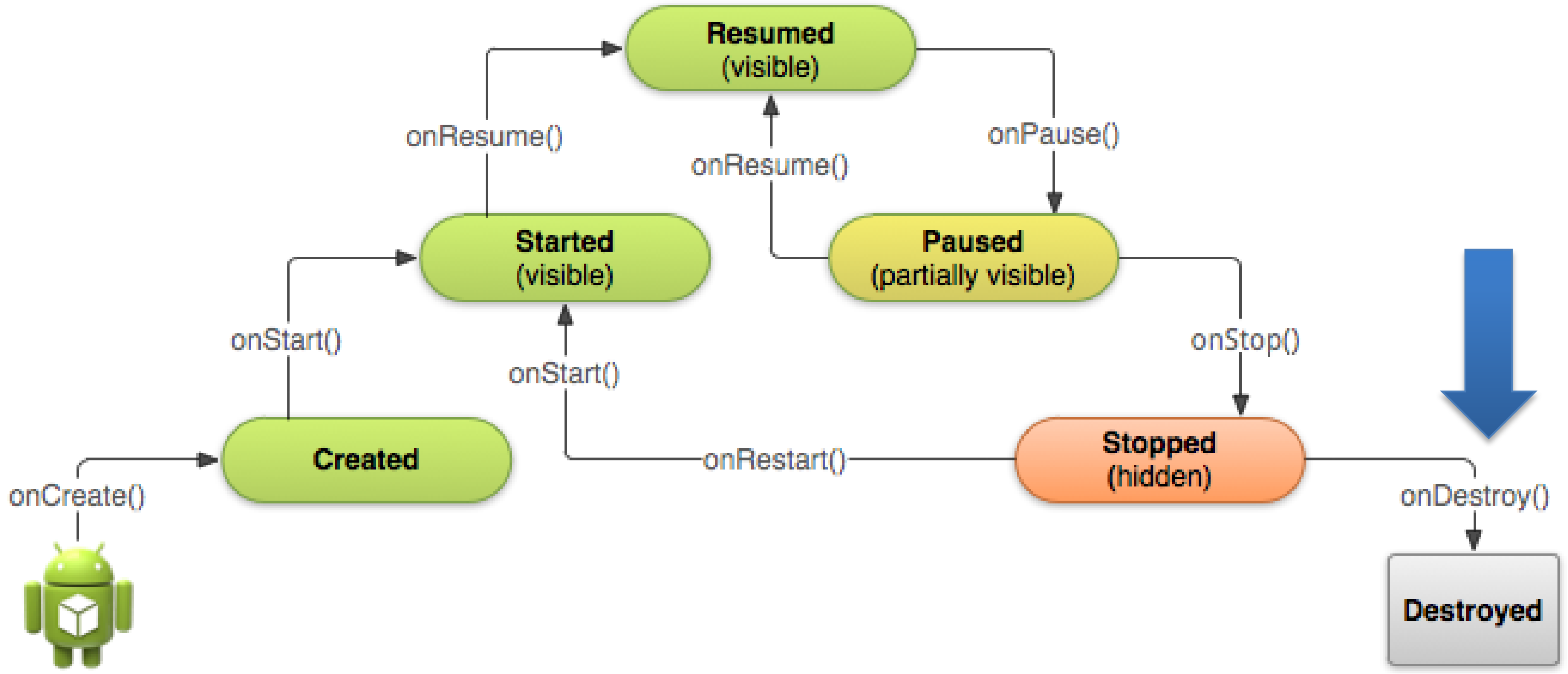


MIT AITI

Resources

- Managing the Activity Lifecycle
  - http://developer.android.com/training/basics/activity-lifecycle/index.html
- Images and source code examples taken from site above

Basic Activity Lifecycle



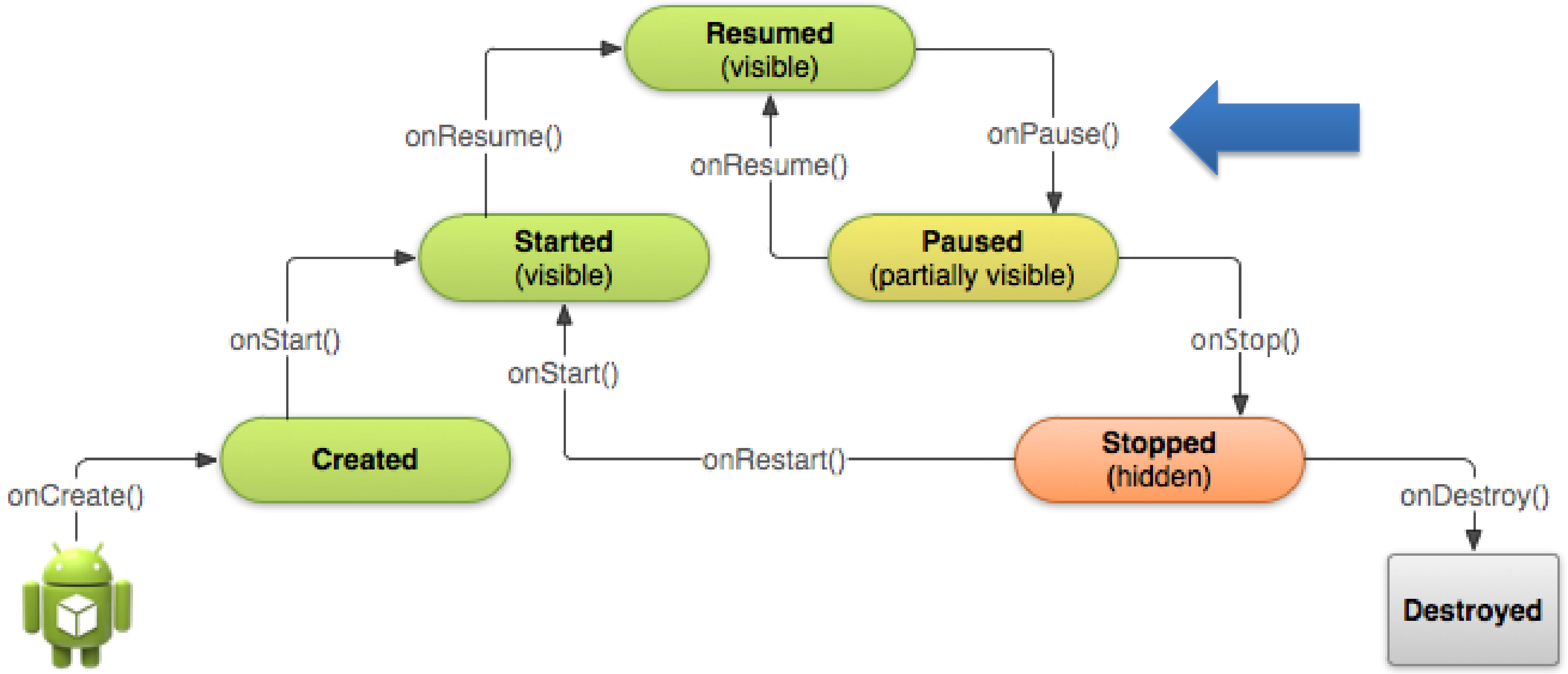
Destroy Application

- Two ways of destroying an application

- onPause() -> onStop() -> onDestroy()
  - Most common. Most cleanup done on onPause() and onStop()
  - Kill background threads in onDestroy()

- onCreate() -> finish() -> onDestroy()
  - Activity operates as a temporary decision maker. Destroy immediately after being created.

Basic Activity Lifecycle



Actions of onPause()

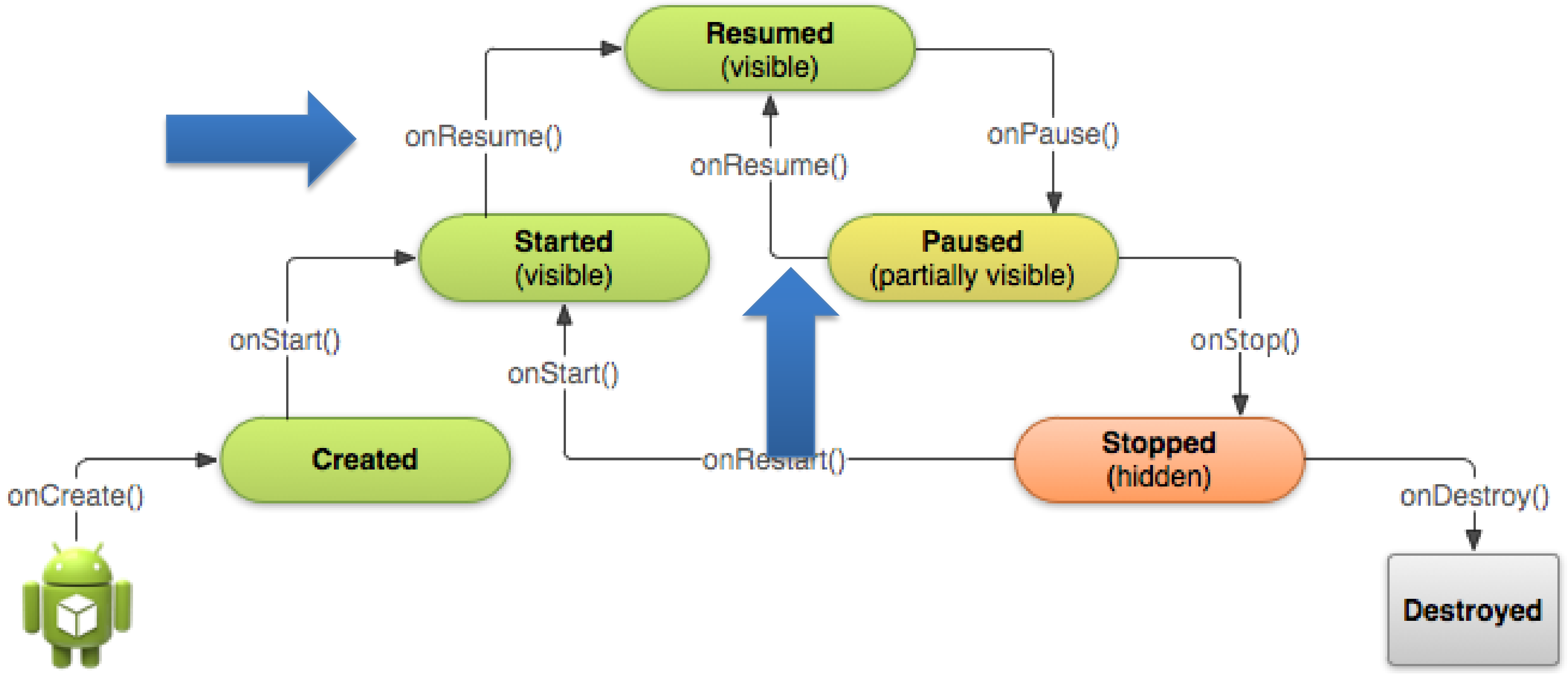
- Stop animations or other ongoing actions that could consume CPU.

- Commit unsaved changes, but only if users expect such changes to be permanently saved when they leave (such as a draft email).

- Release system resources, such as broadcast receivers, handles to sensors (like GPS), or any resources that may affect battery life while activity is paused and the user does not need them.

- Keep it simple – avoid intensive computation. User may come back to it soon.

Basic Activity Lifecycle

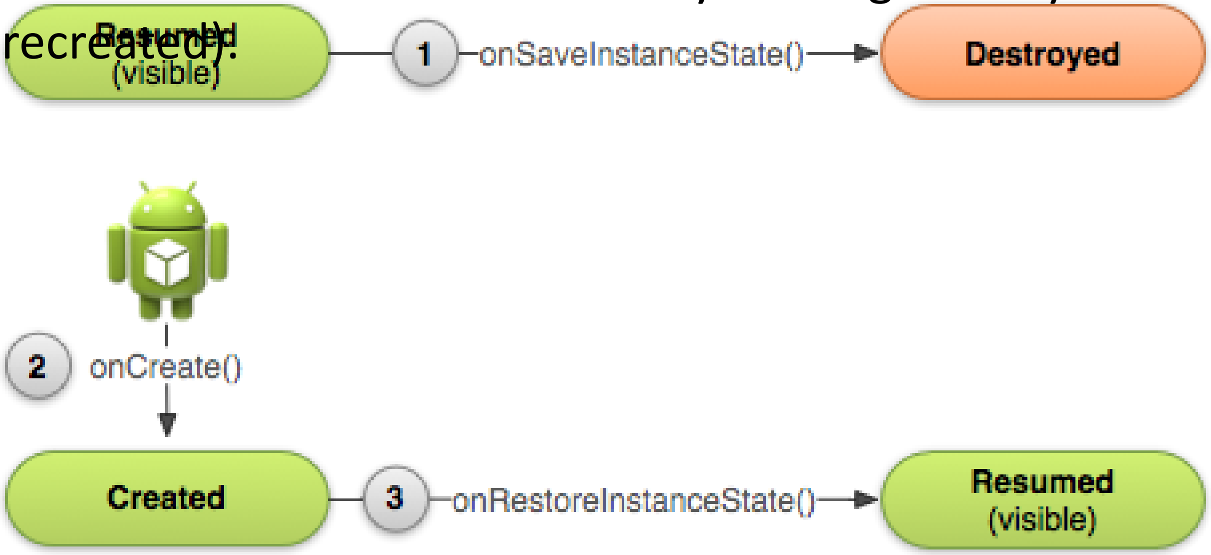


Actions of onResume()

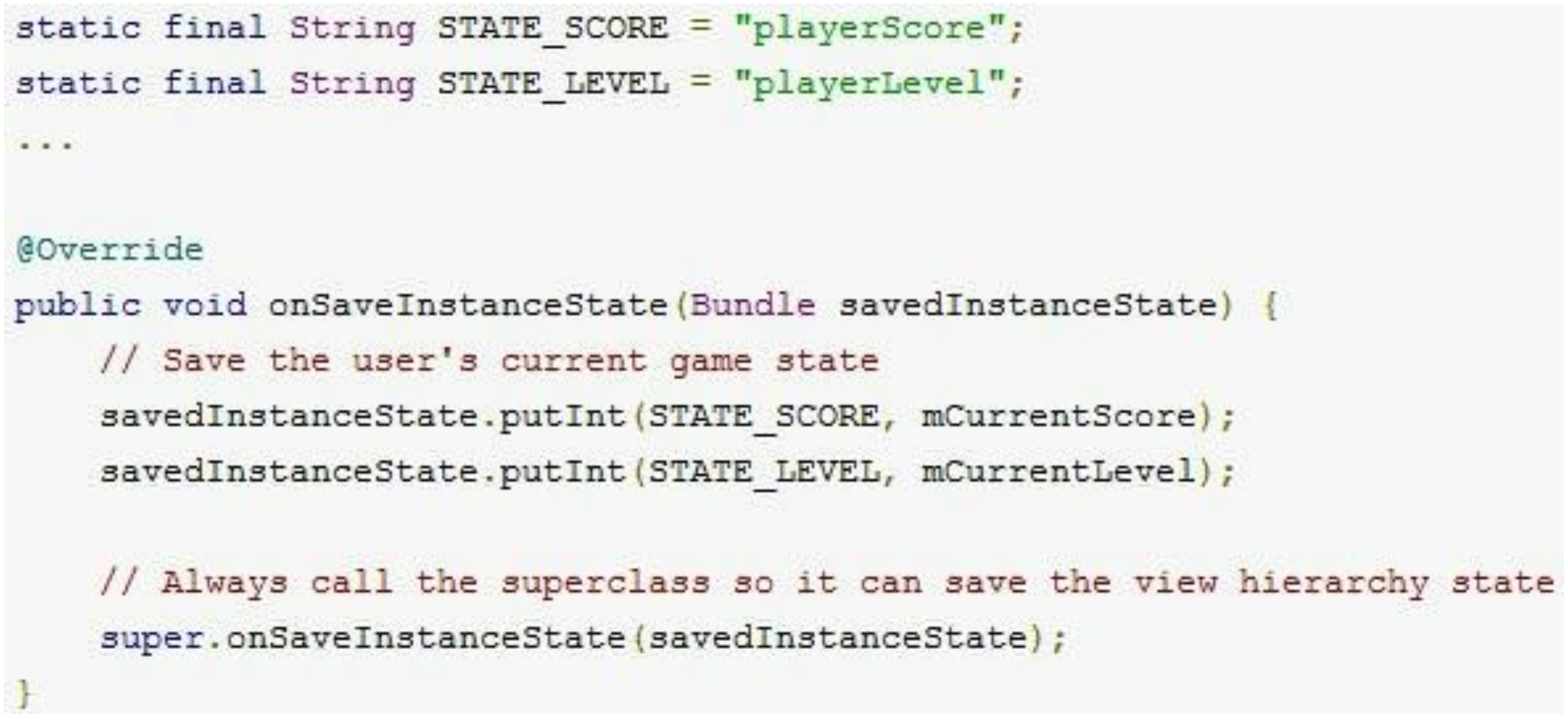
- Called from Started and Paused states
- Initialize components that you release during onPause() and perform any other initializations that must occur each time the activity enters the Resumed state (such as begin animations and initialize components only used while the activity has user focus).
- Counterpart of onPaused()

Save your Activity State

- Additional methods. Allows you to specify additional state data you would like to save in case the Activity instance must be recreated
- System calls these methods as an Activity is being destroyed or restored (recreated).



Example: onSaveInstanceState()



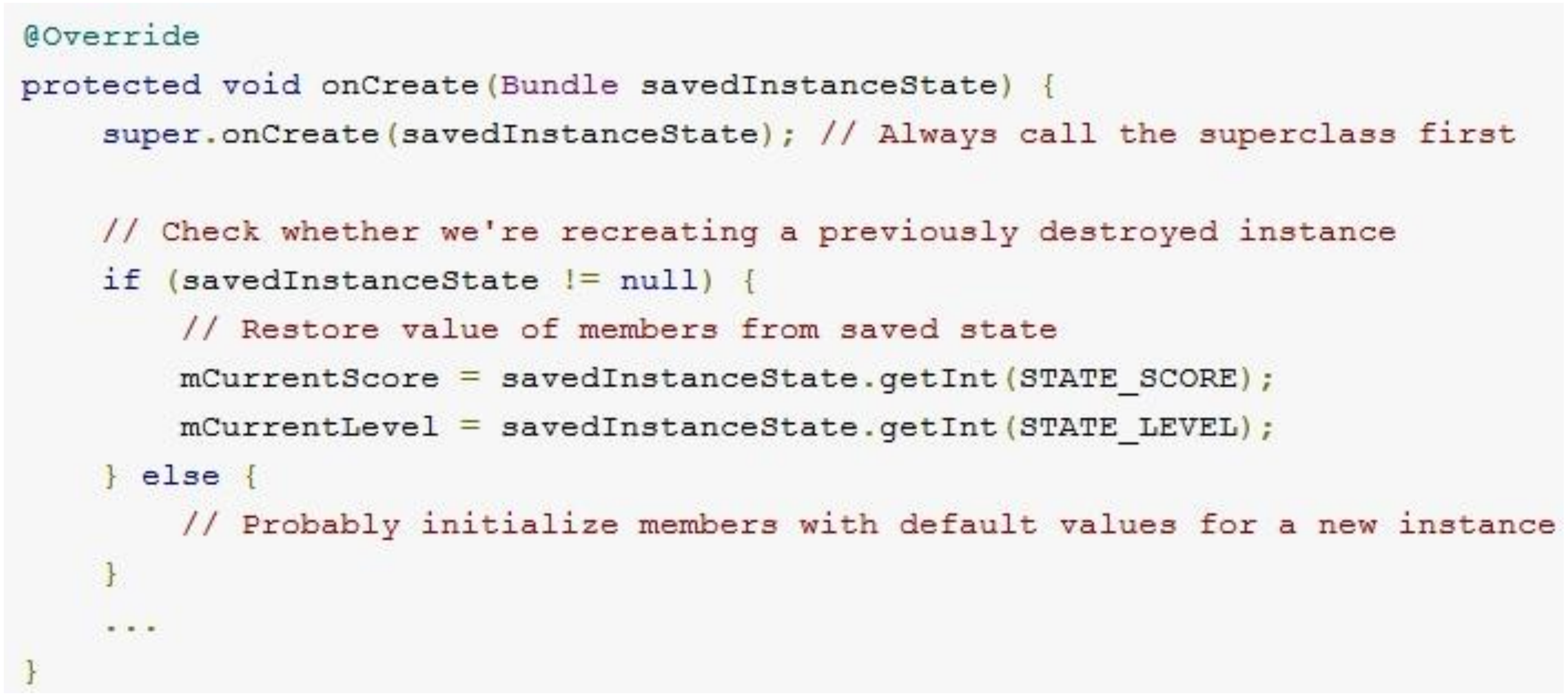
```java
static final String STATE_SCORE = "playerScore";
static final String STATE_LEVEL = "playerLevel";
...

@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
    // Save the user's current game state
    savedInstanceState.putInt(STATE_SCORE, mCurrentScore);
    savedInstanceState.putInt(STATE_LEVEL, mCurrentLevel);

    // Always call the superclass so it can save the view hierarchy state
    super.onSaveInstanceState(savedInstanceState);
}
```

Recreate an Activity

- Use onCreate() – check if bundle is not empty

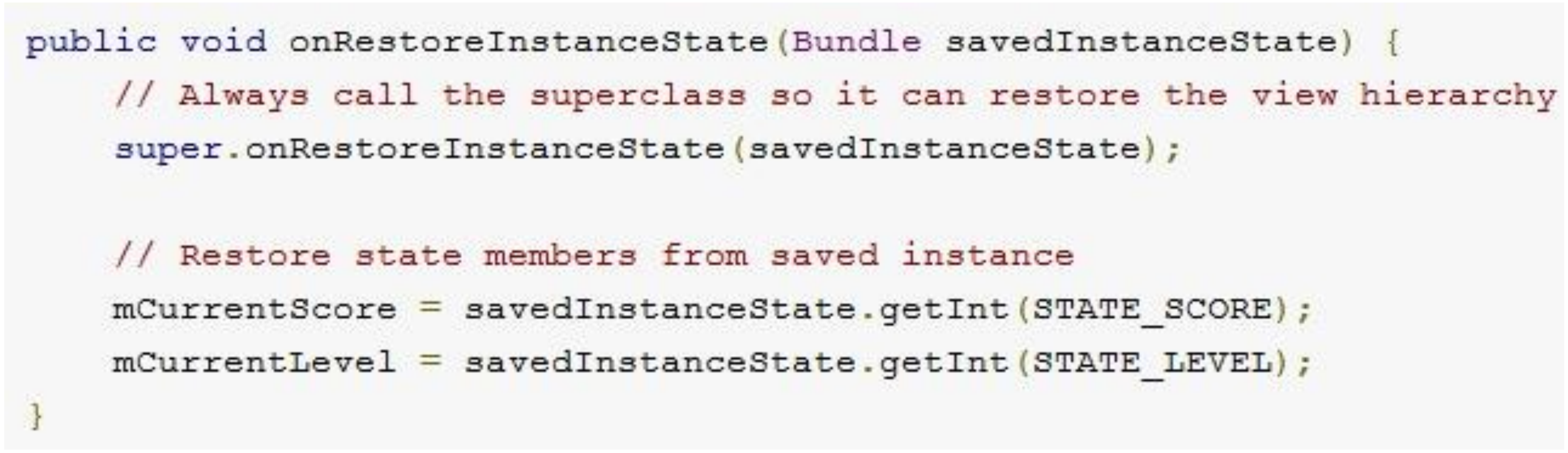


```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState); // Always call the superclass first

    // Check whether we're recreating a previously destroyed instance
    if (savedInstanceState != null) {
        // Restore value of members from saved state
        mCurrentScore = savedInstanceState.getInt(STATE_SCORE);
        mCurrentLevel = savedInstanceState.getInt(STATE_LEVEL);
    } else {
        // Probably initialize members with default values for a new instance
    }
    ...
}
```

Recreate an Activity

- Better way – use onRestoreInstanceState()
- No need to check if Bundle is empty
- Called by system after the onStart() method



```
public void onRestoreInstanceState(Bundle savedInstanceState) {
    // Always call the superclass so it can restore the view hierarchy
    super.onRestoreInstanceState(savedInstanceState);

    // Restore state members from saved instance
    mCurrentScore = savedInstanceState.getInt(STATE_SCORE);
    mCurrentLevel = savedInstanceState.getInt(STATE_LEVEL);
}
```