Android Life Cycle

CS328

Dick Steflik

Life Cycle

- The steps that an application goes through from starting to finishing
- Slightly different than normal Java life cycle due to :
  - the difference in the way Android application are defined
  - the limited resources of the Android hardware platform

Android Applications

- Applications are defined to Android via the android manifest file, located in the root of the Eclipse project definition (AndroidManifest.xml)
- Double clicking on the AndroidManifest.xml file in the Eclipse project will open the Manifest editor.
- The manifest editor is the normal way of creating and modifying the manifest file (defining the app to the system)

Android Applications

- An Android application is a collection of activities, an activity correlates to a screen or form that is presented to the user.
- The HelloAndroid is a simple one screen app that is essentially the same as a Java app run in a terminal/command window. Its AndroidManisest.xml file reflects this :

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloandroid"
    android:versionCode="1"
    android:versionName="1.0">
  <application android:icon="@drawable/icon" android:label="@string/app_name">
    <activity android:name=".HelloAndroid"
              android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
</manifest>
```

<manifest>

- The manifest tag has the following attributes:
  - xmlns ; the name of the namespace (android) and where the DTD for the xml parser is located
  - package ; the name of the java package for this application (must have at least two levels)
  - android:version ; the version code for this version of the app
  - android:versionName ; The version name (for publishing)

<activity>

- child tag of <manifest>
- need one <activity> tag for each activity of the application
- attributes:
  - android:name; the name of the activity, this will be used as the name of the Java file and the resulting class
  - android:label; a string that we will be able to programatically retrieve the activity name at run time.

<intent-filter>

- Child tag of <activity>
- First, what's an intent? In OO-speak an intent is a message sent from one program to another (message dispatcher) to tell the system what to do next. Typically an intent consists of two parts; an action and the data that that action is supposed to use to do it.
- When you select an icon on the main page the intent is to run the app associated with that icon
- The tag is used to construct an android.content.IntentFilter object to handle a particular android.content.Intent

<action>

- child of <intent-filter>
- the action we want done:
  - Predefined actions of the intent class of android.content ; see the api at: http://developer.android.com/reference/android/content/Intent.html

<category>

- child of <intent-filter>
- additional attributes that can be supplied
- LAUNCHER – indicates that it should apper in the launcher as a top level application
- see the api documentation for more on intent resolution.

Intents

- Commonly used Google application intents
  http://d.android.com/guide/appendix/g-app-intents.html
- Registry of 3rd party application Intents
  http://www.openintents.org/en/intentstable

Whew!

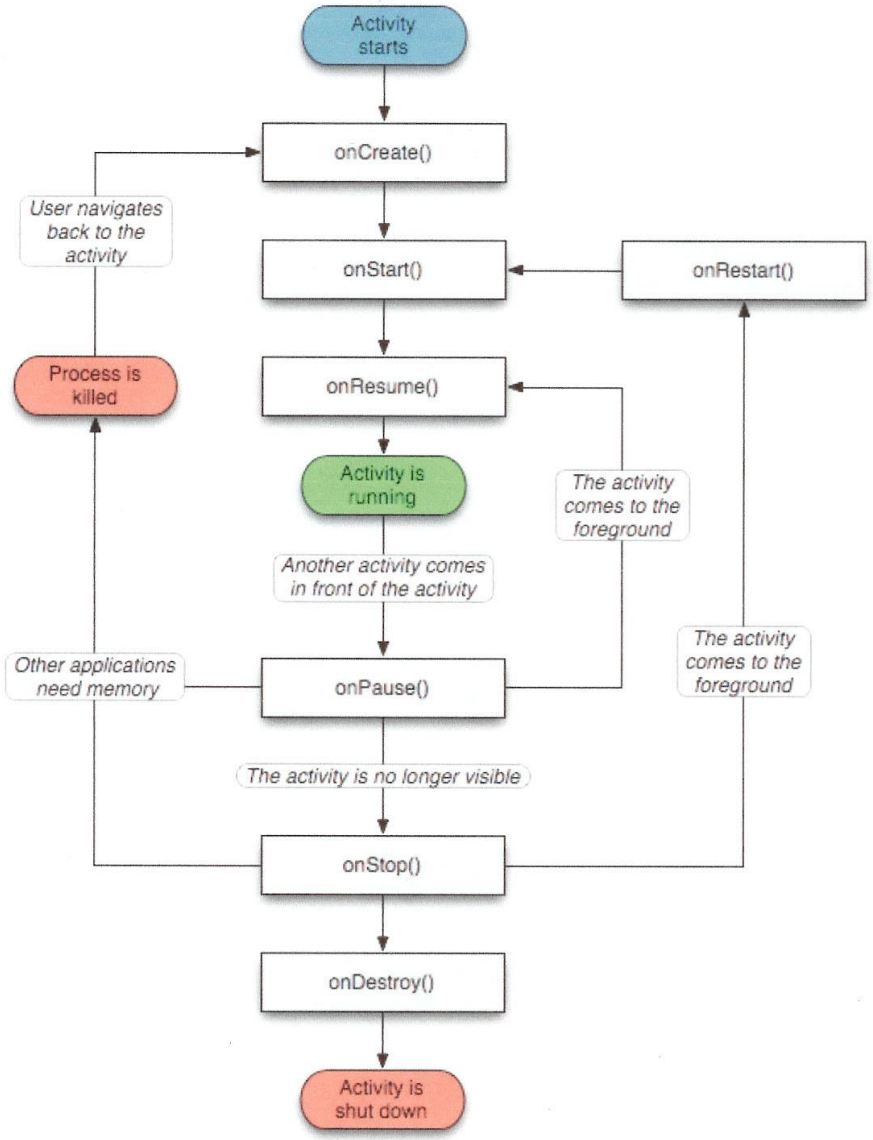
- we’ve explained the HelloAndroid manifest file, on to Life Cycle and Life cycle management.

Life Cycle

- Each application runs in its own process.
- Each activity of an app is run in the apps process
- Processes are started and stopped as needed to run an apps components.
- Processes may be killed to reclaim needed resources.
- Killed apps may be restored to their last state when requested by the user

Management

- Most management of the life cycle is done automatically by the system via the activity stack.
- The activity class has the following method callbacks to help you manage the app:
  - onCreate()
  - onStart()
  - onResume()
  - onPause()
  - onStop()
  - onRestart()
  - onDestroy()


Activity starts
onCreate()
User navigates back to the activity
onStart()
onRestart()
Process is killed
onResume()
Activity is running
The activity comes to the foreground
Another activity comes in front of the activity
Other applications need memory
onPause()
The activity comes to the foreground
The activity is no longer visible
onStop()
onDestroy()
Activity is shut down

using the callbacks

- To use a callback just overload it in your activity java file.
- The lifecycle is explained very well here:

  http://developer.android.com/videos/index.html#v=fL6gSd4ugSI
- The use of the callbacks is explained in the api documentation for the activity class:

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