Android Software Architecture

Android Software Architecture

Application framework ➔ Java classes
Primary support for developing apps
Android Software Architecture

Libraries ➞ C/C++
Implemented functionality supporting application framework
Android Software Architecture

Android Runtime ➔ C/C++
Support program execution at runtime
Android Software Architecture

Linux Kernel ➔ C/C++
Interface with Hardware
OS Services – memory management, scheduling, ...
```java
public static void main()
{
    // initialize
    // work
    // end
}
```
Traditional Java Application

```java
public class BankProgram {
    public static void main(String[] args) {
        BankAccount account1 = new BankAccount("William Turkett", 75290, 100.00);
        Transaction transaction1 = new Transaction("Deposit", 200.00);
        Transaction transaction2 = new Transaction("Withdrawal", 50.00);
        account1.handleTransaction(transaction1);
        account1.handleTransaction(transaction2);
        System.out.println("Balance: " + account1.getBalance());
    }
}
```
Simple Android Application: Design

An application with two screens. Each screen is composed of 1 or more GUI elements.
Simple Android Application: Design and Example

An application with two screens. Main map screen Settings

Each screen is composed of several GUI elements.
Simple Android Application: Our Initial Goal

Our goal: An Application with one Activity with one View (a big textbox)
Application Priority

- The Android OS allows multiple applications to be executing at a time
- To service an application, the OS can reclaim resources by killing another application.
- Lowest priority = most likely to be killed
- Within a level ➔ least recently seen = first to be killed

Priority Hierarchy

- Active applications
- Visible applications
- Service applications
- Background applications
- “Empty” (completed) applications
Activity States

- Within an application, there can be multiple Activities (screens)

- Activities are maintained in a stack

Activity States

**Active**
Foreground – receiving input

**Paused**
Visible but obscured

**Stopped**
No longer visible, still in memory

**Inactive**
Not visible, not in memory (terminated)

*Activities can be killed, just like applications, based on priority; states above are in priority order
Applications and Activities

Applications and Activities are just Java classes
android.app.Application
android.app.Activity

We will interact with them using member functions

Initially, we will use a default Application setup and focus on building Activities.

When an Application is selected on the device, it will trigger a first Activity to be instantiated.
Android App Lifecycle

As an Activity moves through its possible different states, functions are automatically called on the Activity, triggering different parts of our code.

[Remember this happens for every screen!]

The first such function is called `onCreate` and is triggered when an Activity is first requested.

- **Active** (state: active)
- **Visible** (state: active or paused)
- **Alive** (active, paused, stopped)
Traditional Java Application

Class A

public static void main()
{
    // initialize
    // work
    // end
}
Creating a first app

1. Create a new Android project (a collection of source code and resources for the app) from the Eclipse file menu

2. Choose a project name (can be anything)

3. Application specifics:
   1. Target platform
   2. Application name
   3. Package name
   4. Initial activity to launch
   5. Absolute minimum platform

4. Finish
Creating a first app

1. Expand the project, src folder, and your chosen package

2. Choosing your Activity file will reveal a default implementation of the `onCreate` function
   1. Calls the `onCreate` of the Activity parent class
   2. Sets the content of this screen to be an XML specified layout (we’ll come back to this)
Creating a first app

3. Replace pre-generated code with your own TextView code

4. Run the app from Eclipse

5. Emulator should start, and open your app
Applications and Activities

• How does the Application know the initial Activity to call?
  – Stored in application manifest: AndroidManifest.xml
    • Managed by Eclipse for us
Applications and Activities

• A manifest for an Application with two Activity components

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="turkett.android.ridethewake"
    android:versionCode="3"
    android:versionName="1.2">
    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".StartActivity"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".SettingsActivity"
            android:label="@string/settings_name" />
        <uses-library android:name="com.google.android.maps" android:required="true"/>
    </application>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-sdk android:minSdkVersion="8" android:targetSdkVersion="8" />
</manifest>
```
Important Java Concepts

• Packages:
  – packages of classes = directories of files
    • Importing in Java
    • Your own

• Inheriting from Activity/super

• Becoming familiar with the Android API