Introduction to the Android Platform

Jaeho Shin
2011-02-25
ROPAS Show&Tell
Introduction
Introduction to the Android Platform

App Developers

Device Makers

Market Operators

- Makers port Android platform to their devices
  - The platform is open source: source.android.com
  - It is mostly written in C/C++ with small Java glue codes

- Application Framework
- Native Services, Libraries
- Linux kernel
- Dalvik VM

Users
App Developers

Java code
- Android developers write their apps in Java language.
- Android developers can benefit from the rich tools support in the Java community, e.g., refactoring features in Eclipse.
- However, Android is not Java SE nor Java ME.

C/C++ code

SDK
- Android SDK contains:
  - Android Platform libraries
  - Android emulator
  - Dalvik Executable Compiler (dex)
  - SDK command line tools
  - Android Asset Packaging Tool (aapt)
  - Android Debug Bridge (adb)
  - and more...

NDK
- Native Development Kit
  - Allows developers to mix C/C++ code in their apps
  - Uses JNI (Java Native Interface)

Android app is packaged into an .apk which contains:
- Dalvik executable binaries
- native binaries
- AndroidManifest.xml
- Other resources, e.g., images, data files
Java code

- Android developers write their apps in Java language.

- Android developers can benefit from the rich tools support in the Java community, e.g. refactoring features in Eclipse.

- However, Android is not Java SE nor Java ME.
App

Android app is packaged into an .apk which contains:

- Dalvik executable binaries
- native binaries
- AndroidManifest.xml
- Other resources, e.g. images, data files
Android SDK comprises

- Android Platform libraries w/ system images
- Android Emulator
- Dalvik Executable Compiler (dx)
- AIDL compiler (aidl)
- Android Asset Packaging Tool (aapt)
- Android Debug Bridge (adb)
- and more...

ADT is a Eclipse plugin that helps using the SDK
Java code

- Android developers write their apps in Java language.
- Android developers can benefit from the rich tools support in the Java community, e.g., refactoring features in Eclipse.
- However, Android is not Java SE nor Java ME.

C/C++ code

NDK

Native Development Kit

- Allows developers to mix C/C++ code in their apps
- Uses JNI (Java Native Interface)

App

Android app is packaged into an .apk which contains:
- Dalvik executable binaries
- native binaries
- AndroidManifest.xml
- Other resources, e.g., images, data files
NDK

Native Development Kit

• Allows developers to mix C/C++ code in their apps

• Uses JNI (Java Native Interface)
App Developers

Java code
- Android developers write their apps in the Java language.
- Android developers can benefit from the rich tools support in the Java community, e.g., refactoring features in Eclipse.
- However, Android is not Java SE nor Java ME.

C/C++ code

Android SDK components
- Android Platform libraries/system image
- Android Emulator
- Dalvik Executable Compiler side
- NDK compiler (-ldl)
- Android Asset Packaging Tool (aapt)
- Android Debug Bridge (adb)
- and more...

NDK
- Native Development Kit
- Allows developers to mix C/C++ code in their apps
- Uses JNI (Java Native Interface)

App
- Android app is packaged into an .apk which contains:
  - Dalvik executable binaries
  - native binaries
  - AndroidManifest.xml
  - Other resources, e.g., images, data files
App Fundamentals

Android Apps are built with four components:

**Activity**
- Activity corresponds to a single screen in the UI.
- It starts when a matching Intent was passed to startActivity() or onCreate()
- Intent filters declare which intents are handled by an Activity
- Different apps can start each other's activities

**Service**
- Service runs long operations in the background, e.g., play music, background download, or background tasks
- It starts when a matching Intent was passed to startService() or @Continue
- Different apps can start each other's services

**Broadcast Receiver**
- Broadcast Receiver is used for handling system-wide announcements, e.g., battery is low, roaming status change, or incoming message
- Intent filters declare when the receiver is handled
- Intent filters can declare whether the receiver should handle the request
- Intent filters can declare which receiver handles the request

**Content Provider**
- Content Provider abstracts the way of reading writing apps data
- Apps can access other app's data through Content Providers
- Content Provider can be created by invoking a target app on a ContentResolver

Components are declared in **AndroidManifest.xml** as well as their capabilities and app requirements.
Activity corresponds to a single screen in the UI.

It starts when some matching Intent was passed to `startActivity()`.

Intent Filters declare which Intents are handled by an Activity.

Different apps can start each others' activities.
Service

Service runs long operations in the background.
e.g. play music in the background,
or download data over the network

It starts when some matching Intent was passed to startService().

Different apps can start each others' services.
Broadcast Receiver

Broadcast Receiver is used for handling system-wide announcements, e.g. battery is low, screen turned off, incoming message, ...

Many broadcasts originate from the system, but apps can also initiate them.

Intents are broadcasted by sendBroadcast(), sendOrderedBroadcast(), or sendStickyBroadcast().
Content Provider

Content Provider abstracts the way of reading/writing app's data.

Apps can access other apps' data through Content Providers.

It can be queried by invoking a targeted query() on a ContentResolver.
App Fundamentals

Android Apps are built with four components:

**Activity**
- Activity corresponds to a single screen in the UI.
- It starts when some matching intent was passed to startActivity().
- Intent filters decide which intents are handled by an Activity.
- Different apps can start each other’s Activities.

**Service**
- Service runs long operations in the background, e.g., play music in the background, or download data over the network.
- It starts when some matching intent was passed to startService().
- Different apps can start each other’s Services.

**Broadcast Receiver**
- Broadcast Receiver is used for handling system-wide announcements, e.g., battery is low, screen orientation change, incoming message.
- Many broadcasts originate from the system, but apps can also initiate them.
- Intents are broadcast by sendBroadcast(), sendOrderedBroadcast(), or sendMulticastBroadcast().

**Content Provider**
- Content Provider abstracts the way of reading/writing app’s data.
- App can access other app’s data through Content Providers.
- It can be opened by invoking a targeted query() on a ContentResolver.

Components are declared in `AndroidManifest.xml` as well as their capabilities and app requirements.
Introduction to the Android Platform

App Developers

Device Makers

Market Operators

App

Java code

SDK

C/C++ code

NDK

Users

Makers port Android platform to their devices
- The platform is open source: source.android.com
- It is mostly written in C/C++ with small Java glue codes

Prezi.com
Device Makers

Makers port Android platform to their devices
- The platform is open source: source.android.com.
- It is mostly written in C/C++ with small Java glue codes.

Linux kernel
- Android uses a Modified version of Linux 2.6 kernel
  - BPF Compiler
  - Out Of Memory Killer
  - Ashmem
  - Wake Locks for Power Management
  - RAM Console and Log Device
  - Android Debug Bridge
  - Flash Filesystems
  - ...

Dalvik VM
- Optimized for
  - Slow CPU
  - Low RAM
  - No swap space
  - Battery powered
  - Fast Spawning
- Each app runs in a separate Dalvik process
- Alternate instruction set to Java
- Smaller code size
- Multiple class into single .dex to share strings and constants
- Virtual registers instead of stack
- JVM + Stack machine
- Dalvik = Register-based arch
- JIT since Android 2.2

Application Framework
- Mostly Java + JNI glue code
- Sun Java SE
  - Java language + JVM + Java Class libraries
- Android Java
  - Java language + Dalvik + Apache Harmony
- No AWT, Swing, nor Java ME Class Library (MIDP, CLDC, ...)

Native Services, Libraries
- Mostly written in C/C++
  - Service Manager
  - Media Server
  - RILD
  - Zygote
  - System Server
  - [Surface,Audio]Finger
  - [Activity,Power,Package,...] Manager
  - ...
  - Bionic
  - WebKit
  - SQLite
  - OpenGL ES
  - openCORE
  - ...

Prezi.com
Linux kernel

Android uses a Modified version of Linux 2.6 kernel

- IPC Binder
- Out Of Memory Killer
- Ashmem
- Wake Locks for Power Management
- RAM Console and Log Device
- Android Debug Bridge
- Flash Filesystems
- ...
Dalvik VM

Optimized for
• Slow CPU
• Low RAM
• No swap space
• Battery powered
• Fast Spawning

Each app runs in a separate dalvik process

Alternate instruction set to Java
Smaller code size
• Multiple .class into single .dex to share strings and constants

Virtual registers instead of stack
• JVM = Stack machine
• Dalivk = Register-based arch.

JIT since Android 2.2
Native Services, Libraries

Mostly written in C/C++

- Service Manager
- Media Server
- RILD
- Zygote
- System Server
- {Surface,Audio}Flinger
- {Activity,Power,Package,...} Manager
- ...

- Bionic
- WebKit
- SQLite
- OpenGL ES
- openCORE
- ...

prezi.com
Application Framework

Mostly Java + JNI glue code

Sun Java SE
= Java language + JVM + Java Class libraries
Android Java
= Java language + Dalvik + Apache Harmony

No AWT, Swing, nor Java ME Class Library (MIDP, CLDC, ...)

Prezi.com
Device Makers

Makers port Android platform to their devices
- The platform is open source: source.android.com.
- It is mostly written in C/C++ with small Java glue codes.

Linux kernel

- Android uses a Modified version of Linux 2.6 kernel
  - IPC Binder
  - Out Of Memory Killer
  - Ashmem
  - Wake Locks for Power Management
  - RAM Console and Log Device
  - Android Debug Bridge
  - Flash Filesystems
  - ...

Dalvik VM

- Optimized for
  - Slow CPU
  - Low RAM
  - No swap space
  - Battery powered
  - Fast Spawning
  - Each app runs in a separate Dalvik process

Application Framework

- Mostly Java + JNI glue code
- Sun Java SE
  - Java language + JVM + Java Class libraries
  - Android Java
    - Java language + Dalvik + Apache Harmony
  - No AWT, Swing, nor Java ME Class Library (MIDP, CLDC, ...)

Native Services, Libraries

- Mostly written in C/C++
- Service Manager
- Media Server
- RILD
- Zygote
- System Server
- [Surface,Audio]Binder
- [Activity,Power,Package,...] Manager
- ...

- Bionic
- WebKit
- SQLite
- OpenGL ES
- openCORE
- ...
Binder IPC

IPC is crucial since Android apps & services run in separated processes. Data are sent through "Parcel"s in "transaction"s with Kernel support. Passing Intent objects between apps, calling native system services, ... all rely on Binder IPC.

AIDL is used for generating IPC stub and proxy in Java, but C++ codes are all hand-crafted.
Introduction to the Android Platform

App Developers

Device Makers

Market Operators

Java code
SDK
App
C/C++ code
NDK

Makers port Android platform to their devices
- The platform is open source: source.android.com.
- It is mostly written in C/C++ with small Java glue codes.

Linux kernel
Dalvik VM
Application Framework
Native Services, Libraries

Samsung Apps
Android Apps
App Store

Users
Conclusion

• Android platform is complex.
• So, there will be various analysis problems.
• We need to understand it properly to apply program analysis techniques at the right level.