Emerging Technologies: Mobile Development for Android Devices

Dalvik & ART

Introduction

- Dalvik
- ART
- Dalvik ART comparison
- The Dex file format
- Garbage collection

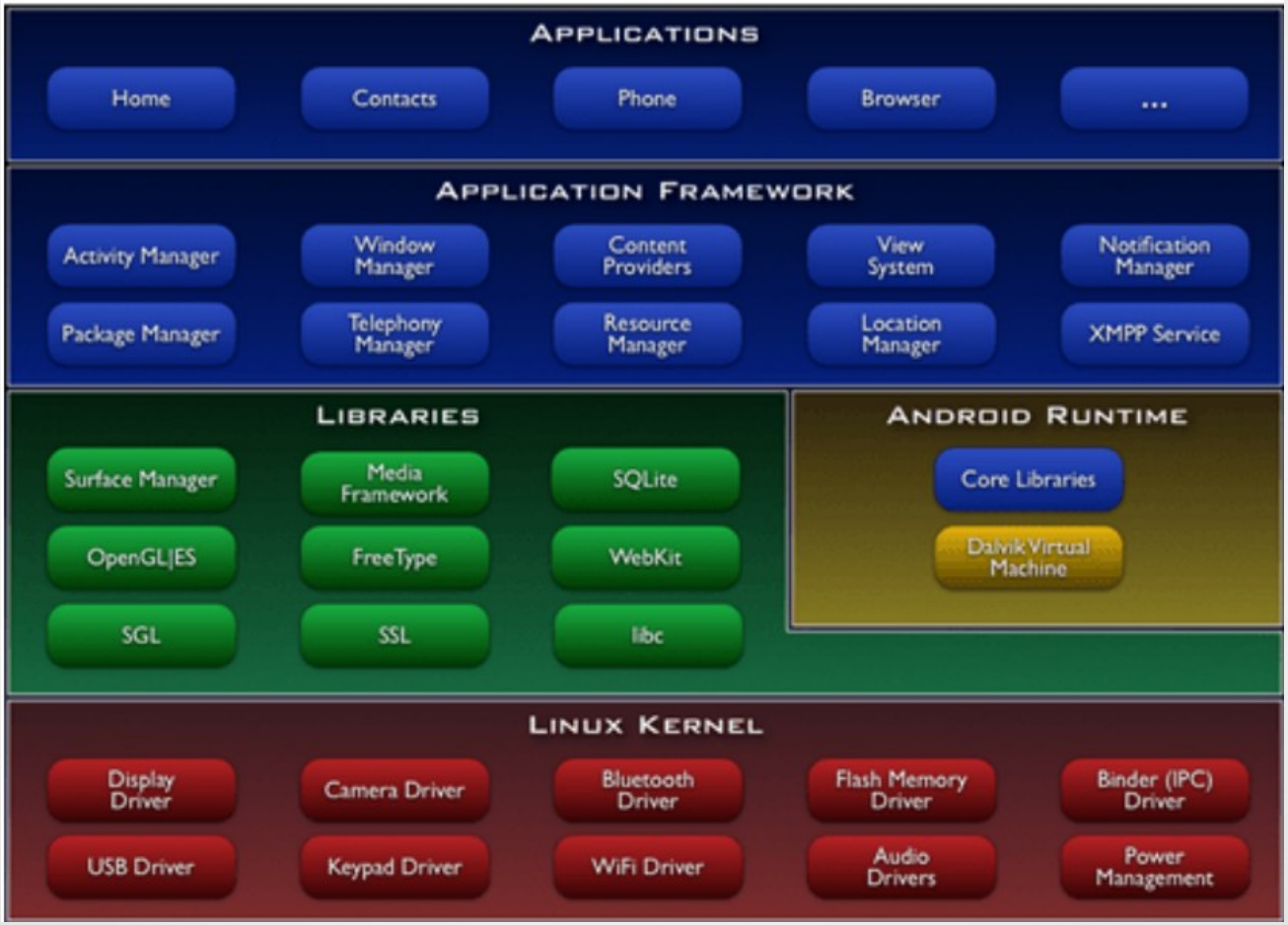
Dalvik

- Original runtime. Present since Version 1.0
- There are a significant number of devices that run Dalvik still in existence.
- Dalvik is a runtime code interpreter using a custom virtual machine similar to the JVM.
- A virtual machine hides hardware, instruction and architecture differences of the various machines on which it runs.
- There is no one specific architecture selected for Android.
- However, most devices are ARM based.

Dalvik

- Dalvik uses bytecode as an intermediate stage.
- If a new processor wants to use Android, then a version of Dalvik would have to be created for it.
- However, for applications that run on top of Dalvik, it looks the same to them all regardless of the underlying device architecture.
- Dalvik is tailored to each device but looks the same to all applications supported above it.
- But, as with other interpreters, there is a performance cost.

Dalvik


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource Manager
Location Manager
XMPP Service
Libraries
Surface Manager
Media Framework
SQLite
OpenGL|ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Flash Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

Dalvik

- Dalvik runs on top of an operating system (OS) with a linux kernel.
- The linux kernel manages process scheduling, threads, security etc.
- Running Dalvik on top of an existing OS with a proven kernel was more economical than designing everything from scratch. Creating an entire OS is a large, costly and time consuming task.
- Using a linux kernel takes advantage of open-source skills and bug-fixes as well as avoiding having to take on already solved problems.

Dalvik

- On Dalvik, each application is given its own process and its own instance of a virtual machine.
- Dalvik is able to switch between VMs quickly.
- Threading, memory management etc. is handed off to the kernel.
- Native libraries implement core functionality that will be used by all applications.
- Includes SQLite and OpenGLES for example.
- This functionality is made available through java wrapper classes.

Dalvik

- Libraries are stable and long-standing, many benefiting from open-source community contributions.
- The application framework consists of the whole Android API. This is what the programmer/developer interacts with.
- Single unified Java interface to access all components of Android.
- Hides low-level implementation details.
- Write once run anywhere.

Native Development Kit (NDK)

- Where performance is an absolute priority, this can be optimised by using native code.
- A NDK is provided for this purpose.
- C++ based.
- Only to be used in limited cases, eg. games, DSP etc.
- Is more limited in what it can do compared to the SDK.
- With the advent of ART, there is a much less compelling case for using native code.

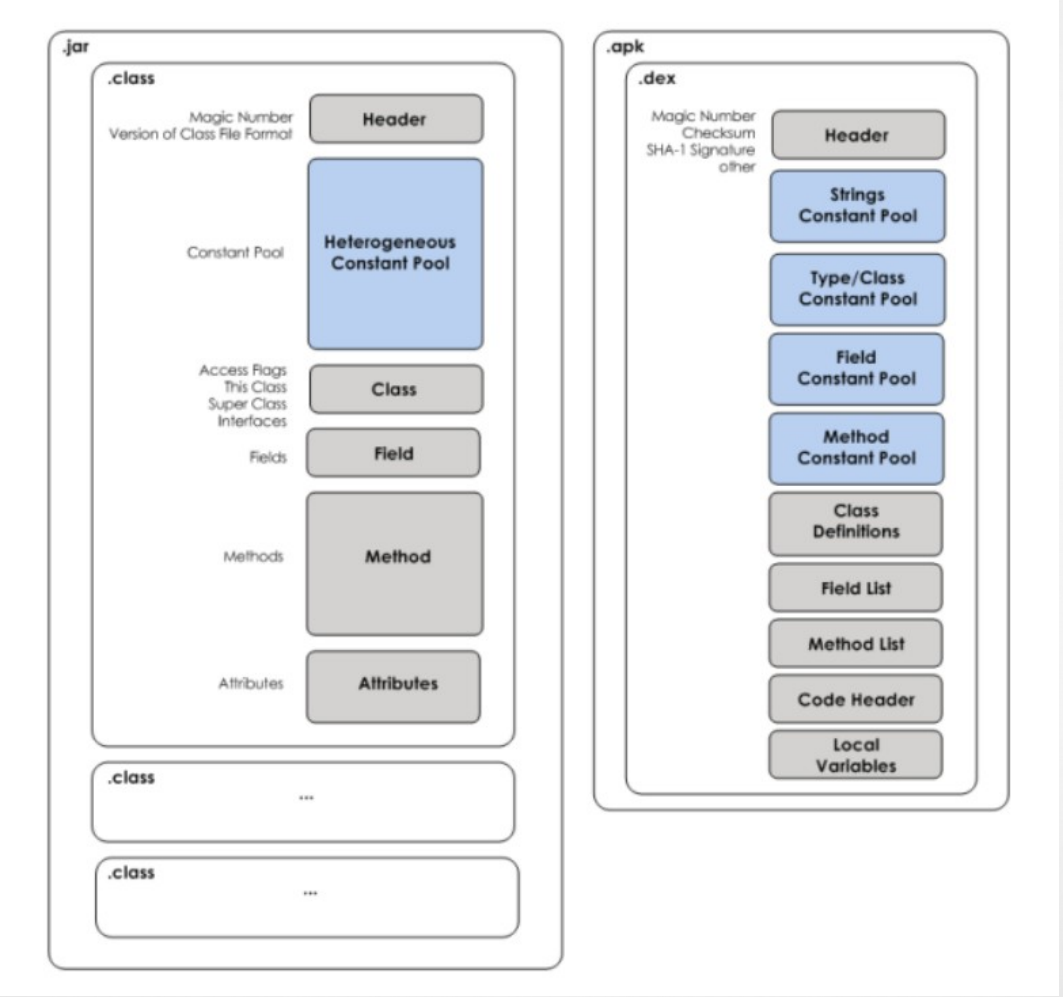
Android Run Time (ART)

- From Android Version 4.4 ART replaces Dalvik.
- However, useful to compare the two, particularly in the way that they treat code resources.
- While Dalvik is a code interpreter, ART introduces ahead of time (AOT) compilation.
- All Dex byte code is compiled to native machine code during installation.
- Application code runs with less instructions giving improved performance and better battery life.
- However, install time increases because the application is compiled at install time.

The Dex File Format

- Android does not use the .jar format for Java code.
- Android uses a custom format called dex which results in smaller files than jar.
- Jar files consist of a number of class files and an overall manifest.
- Each class is kept separate but share the same structure.
- Each class has a header containing format, type, magic numbers, version of Java required etc.
- Also a section indicating the relationship between this class and others.

Dex Jar Comparison



The Dex File Format

- Jar files also contain a list of class fields and class methods.
- There is also a heterogeneous constant pool.
- Contains constants of many types.
- Each constant stores type and value.
- However, the same constant may be repeated in many classes.
- Redundancy in the Jar format resulting in wasted space.
- Dex addresses this by not separating class files.
- Classes share the same space including all the methods, fields and constants related to the classes.
- By storing everything in this way, redundancy is removed.
- More efficient use of storage space.

The Dex File Format

- But it is Dex's use of a number of homogeneous constant pools along with a heterogeneous one that results in considerable space saving.
- Constants can be repeated in multiple class files.
- The use of homogeneous constant pools removes a lot of type information and repetition leading to better use of space.
- According to David Hringer's paper on Dex and Jar on average, the constant pool in a Jar makes up about 61% of the file size.
- Use of the better organized Dex structure can lead to considerably better space efficiency.

Garbage Collection: Dalvik

- The garbage collection mechanism used by Dalvik introduces two pauses in a running application.
- Enumeration: How many times an object is referenced in the current process.
- Marking: Mark all objects with at least one current reference. Those with none are removed.
- Each run of the garbage collector therefore has two ‘points of stutter’. The application momentarily becomes unresponsive.

Garbage Collection: ART

- ART moves some of the garbage collection task to the application itself.
- Less frequent traverses by the garbage collector and less pauses/stutters.
- Objects can very greatly in size with some very large, leading to a potential memory fragmentation problem. A large object space heap is used and is kept separate from the main heap.
- This reduces the amount of fragmentation in both heaps, better use of memory and a less pressing need for garbage collection calls.