Platform Architecture

overview

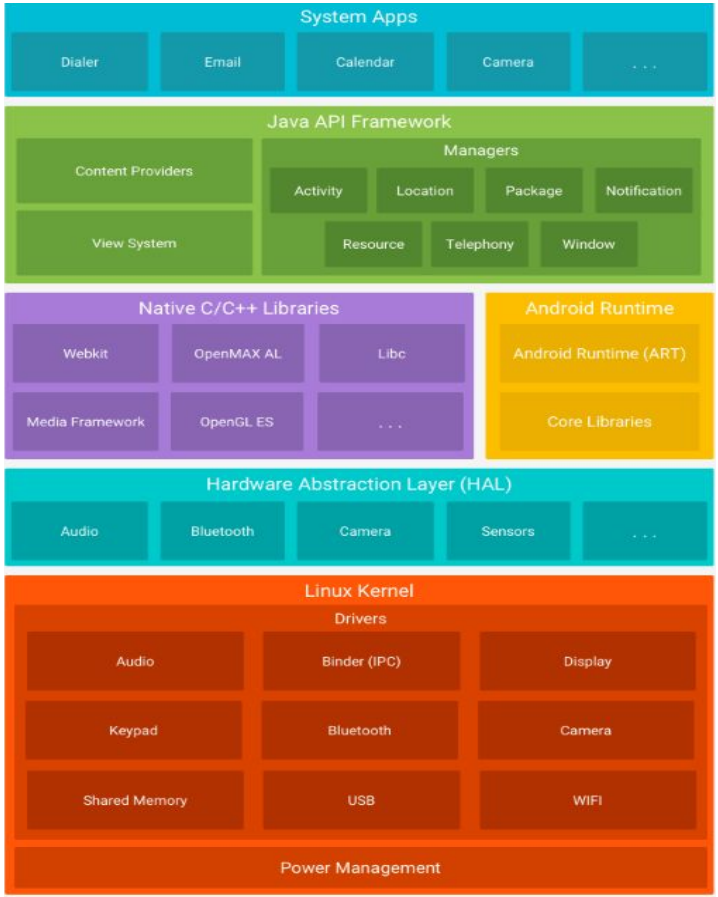
What is Android?

Android is an open source, Linux-based software stack created for a wide array of devices and form factors.

The major components of android stack includes :

- Linux Kernel
- Hardware Abstraction Layer
- Android Runtime
- Native C,C++ packages and Libraries
- System Apps

Android software Stack



Linux kernel:

The foundation of the Android platform is the Linux kernel. For example, the Android Runtime (ART) relies on the Linux kernel for underlying functionalities such as threading and low-level memory management. Using a Linux kernel lets Android take advantage of key security features and lets device manufacturers develop hardware drivers for a well-known kernel.

Hardware abstraction layer (HAL):

The Hardware abstraction layer (HAL) provides standard interfaces that expose device hardware capabilities to the higher-level Java API framework. The HAL consists of multiple library modules, each of which implements an interface for a specific type of hardware component, such as the camera or Bluetooth module. When a framework API makes a call to access device hardware, the Android system loads the library module for that hardware component.

Android Runtime:

Android Runtime (ART) is an application runtime environment used by the Android operating system. Replacing Dalvik, the process virtual machine originally used by Android.For devices running Android version 5.0 (API level 21) or higher, each app runs in its own process and with its own instance of the Android Runtime (ART). ART is written to run multiple virtual machines on low-memory devices by executing Dalvik Executable format (DEX) files, a bytecode format designed specifically for Android that's optimized for a minimal memory footprint. Build tools, such as d8, compile Java sources into DEX bytecode, which can run on the Android platform.

Some of the major features of ART include the following:

- Ahead-of-time (AOT) and just-in-time (JIT) compilation
- Optimized garbage collection (GC)
- On Android 9 (API level 28) and higher, conversion of an app package's DEX files to more compact machine code
- Better debugging support, including a dedicated sampling profiler, detailed diagnostic exceptions and crash reporting, and the ability to set watchpoints to monitor specific fields

Native C/C++ libraries:

Many core Android system components and services, such as ART and HAL, are built from native code that requires native libraries written in C and C++. The Android platform provides Java framework APIs to expose the functionality of some of these native libraries to apps. If you are developing an app that requires C or C++ code, you can use the Android NDK to access some of these native platform libraries directly from your native code.

Java API framework:

The entire feature-set of the Android OS is available to you through APIs written in the Java language. These APIs form the building blocks you need to create Android apps by simplifying the reuse of core, modular system components and services, which include the following:

- A rich and extensible view system you can use to build an app's UI, including lists, grids, text boxes, buttons, and even an embeddable web browser
- A resource manager, providing access to non-code resources such as localized strings, graphics, and layout files.

- A notification manager that enables all apps to display custom alerts in the status bar
- An activity manager that manages the lifecycle of apps and provides a common navigation back stack
- Content providers that enable apps to access data from other apps, such as the Contacts app, or to share their own data

System apps:

Android comes with a set of core apps for email, SMS messaging, calendars, internet browsing, contacts, and more. Apps included with the platform have no special status among the apps the user chooses to install. So, a third-party app can become the user's default web browser, SMS messenger, or even the default keyboard. Some exceptions apply, such as the system's Settings app.The system apps function both as apps for users and to provide key capabilities that developers can access from their own app. For example, if you want your app to deliver SMS messages, you don't need to build that functionality yourself. You can instead invoke whichever SMS app is already installed to deliver a message to the recipient you specify.