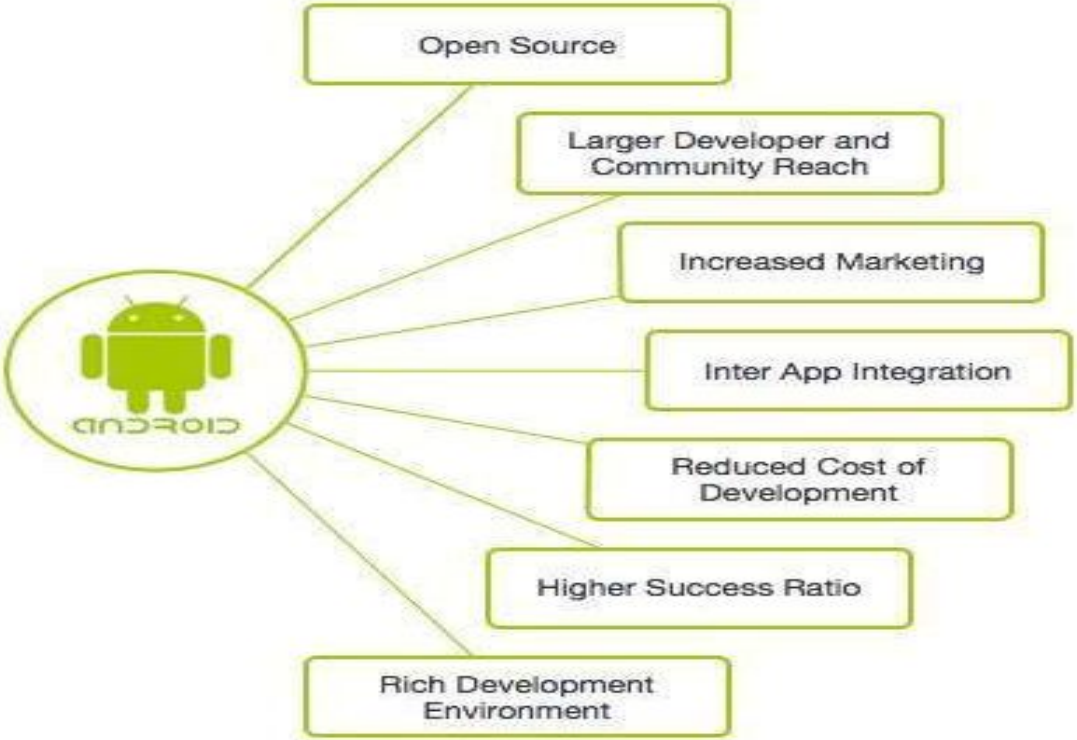
An Overview of the Android Architecture

An Overview of the Android Architecture

- An open source and Linux-based Operating System for mobile devices such as smartphones and tablet computers.
- Developed by the Open Handset Alliance, led by Google, and other companies.

An Overview of the Android Architecture



An Overview of the Android Architecture

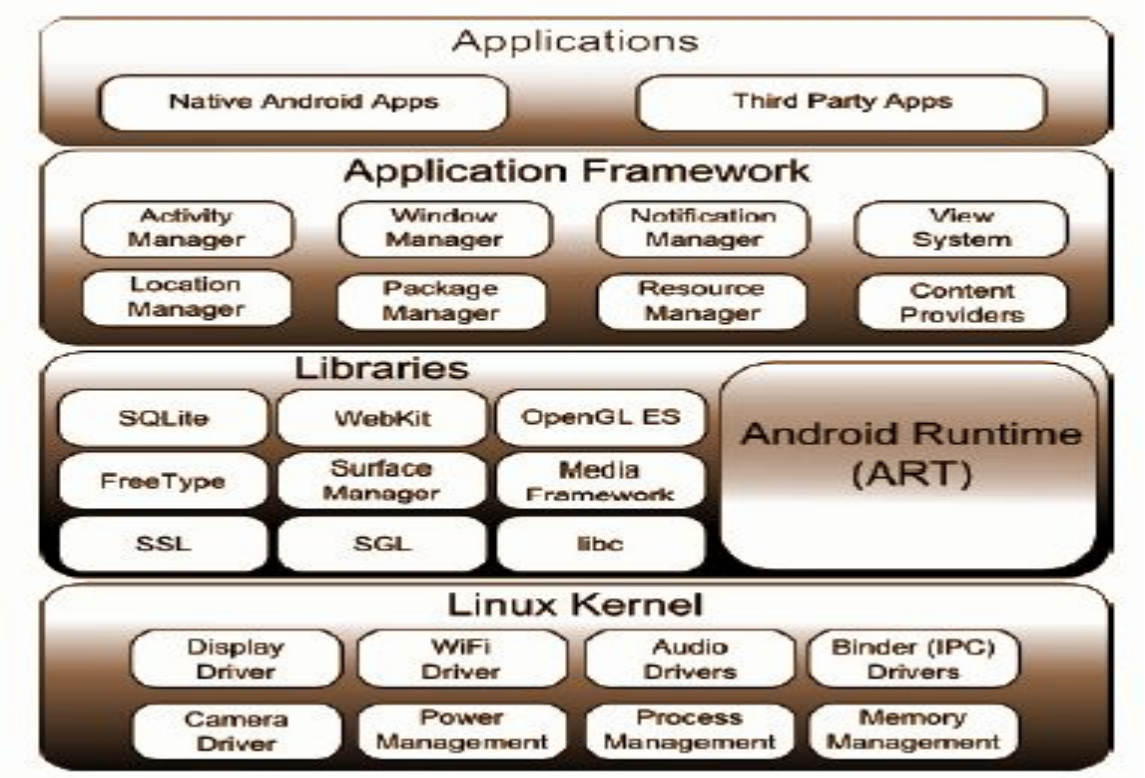
An Overview of the Android Architecture

- Android is an ecosystem made up of a combination of three components:
  - A free, open-source operating system for embedded devices
  - An open-source development platform for creating applications
  - Devices, particularly mobile phones, that run the Android operating system and the applications created for it

The Android Software Stack

- Android is structured in the form of a software stack comprising
  - Applications
  - An operating system
  - Run-time environment
  - Middleware
  - Services
  - Libraries.
- Each layer of the stack, and the corresponding elements within each layer, are tightly integrated and carefully tuned to provide the optimal application development and execution environment for mobile devices.

The Android Software Stack



The Linux Kernel

- Bottom of the Android software stack.
- Provides a level of abstraction between the device hardware and the upper layers of the Android software stack.
- Provides
  - preemptive multitasking
  - low-level core system services such as memory, process and power management
  - providing a network stack and device drivers for hardware such as the device display, Wi-Fi and audio.

Android Runtime – ART

- When an Android app is built within Android Studio it is compiled into an *intermediate bytecode format* (referred to as ***DEX** format*).
- When the application is subsequently loaded onto the device, the Android Runtime (ART) uses a process referred to as *Ahead-of-Time (AOT) compilation*.
  - It translates the bytecode down to the native instructions required by the device processor.
- This format is known as *Executable and Linkable Format (**ELF**)*.

Android Runtime – ART

- Each time the application is subsequently launched, the ELF executable version is run.
  - Results in faster application performance and improved battery life.
- This contrasts with the *Just-in-Time (JIT) compilation* approach used in older Android implementations.
  - Bytecode was translated within a virtual machine (VM) each time the application was launched.

Android Libraries

- Set of standard Java development libraries (providing support for such general purpose tasks as string handling, networking and file manipulation)
- Also includes the Android Libraries
  - These are a set of Java-based libraries that are specific to Android development.
  - Eg: application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access.

Android Libraries

- Some key core Android libraries available to the Android developer are:
  - **android.app** – Provides access to the application model and is the cornerstone of all Android applications.
  - **android.content** – Facilitates content access, publishing and messaging between applications and application components.
  - **android.database** – Used to access data published by content providers and includes SQLite database management classes.
  - **android.graphics** – A low-level 2D graphics drawing API including colors, points, filters, rectangles and canvases.

Android Libraries

- **android.hardware** – Presents an API providing access to hardware such as the accelerometer and light sensor.
- **android.opengl** – A Java interface to the OpenGL ES 3D graphics rendering API.
- **android.os** – Provides applications with access to standard operating system services including messages, system services and inter-process communication.
- **android.media** – Provides classes to enable playback of audio and video.
- **android.net –** A set of APIs providing access to the network stack. Includes android.net.wifi, which provides access to the device's wireless stack.

Android Libraries

- **android.print –** Includes a set of classes that enable content to be sent to configured printers from within Android applications.
- **android.provider –** A set of convenience classes that provide access to standard Android content provider databases such as those maintained by the calendar and contact applications.
- **android.text –** Used to render and manipulate text on a device display.
- **android.util –** A set of utility classes for performing tasks such as string and number conversion, XML handling and date and time manipulation.

Android Libraries

- **android.view –** The fundamental building blocks of application user interfaces.
- **android.widget -** A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
- **android.webkit –** A set of classes intended to allow web-browsing capabilities to be built into applications.

C/C++ Libraries

- Core libraries do not perform much of the actual work.
  - Essentially Java “wrappers” around a set of C/C++ based libraries.
- Eg: When making calls to the *android.opengl* library to draw 3D graphics on the device display, the library actually ultimately makes calls to the *OpenGL ES* C++ library which, in turn, works with the underlying Linux kernel to perform the drawing tasks.
- C/C++ Libraries functionalities:
  - 2D and 3D graphics drawing
  - Secure Sockets Layer (SSL) communication

C/C++ Libraries

- SQLite database management
- Audio and video playback
- Bitmap and vector font rendering
- Display subsystem and graphic layer management and an implementation of the standard C system library (libc).

- Access these libraries solely through the Java based Android core library APIs.
- Direct access to these libraries: Android Native Development Kit (NDK)
  - Call the native methods of non-Java or Kotlin programming languages (such as C and C++) from within Java code using the Java Native Interface (JNI).

Application Framework

- A set of services that collectively form the environment in which Android applications run and are managed.
- This framework implements the concept that Android applications are constructed from reusable, interchangeable and replaceable components.
- This concept is taken a step further in that an application is also able to publish its capabilities along with any corresponding data so that they can be found and reused by other applications.

Application Framework

- The Android framework includes the following key services:
  - **Activity Manager –** Controls all aspects of the application lifecycle and activity stack.
  - **Content Providers** – Allows applications to publish and share data with other applications.
  - **Resource Manager** – Provides access to non-code embedded resources such as strings, color settings and user interface layouts.
  - **Notifications Manager** – Allows applications to display alerts and notifications to the user.

Application Framework

- **View System** – An extensible set of views used to create application user interfaces.
- **Package Manager** – The system by which applications are able to find out information about other applications currently installed on the device.
- **Telephony Manager** – Provides information to the application about the telephony services available on the device such as status and subscriber information.
- **Location Manager** – Provides access to the location services allowing an application to receive updates about location changes.

Applications

- Located at the top of the Android software stack are the applications.
- These comprise both:
  - Native applications provided with the particular Android implementation (for example web browser and email applications).
  - Third party applications installed by the user after purchasing the device.

ANDROID SDK FEATURES

- GSM, EDGE, 3G, 4G, and LTE networks for telephony or data transfer, enabling you to make or receive calls or SMS messages, or to send and retrieve data across mobile networks
- Comprehensive APIs for location-based services such as GPS and network-based location detection
- Full support for applications that integrate map controls as part of their user interfaces
- Wi-Fi hardware access and peer-to-peer connections
- Full multimedia hardware control, including playback and recording with the camera and microphone

ANDROID SDK FEATURES

- Media libraries for playing and recording a variety of audio/video or still-image formats
- APIs for using sensor hardware, including accelerometers, compasses, and barometers
- Libraries for using Bluetooth and NFC hardware for peer-to-peer data transfer
- IPC message passing
- Shared data stores and APIs for contacts, social networking, calendar, and multi-media
- Background Services, applications, and processes
- Home-screen Widgets and Live Wallpaper
- The ability to integrate application search results into the system searches

ANDROID SDK FEATURES

- An integrated open-source HTML5 WebKit-based browser
- Mobile-optimized, hardware-accelerated graphics, including a path-based 2D graphics library and support for 3D graphics using OpenGL ES 2.0
- Localization through a dynamic resource framework
- An application framework that encourages the reuse of application components and the replacement of native applications

INTRODUCING THE DEVELOPMENT FRAMEWORK

- Android SDK includes everything you need to start developing, testing, and debugging Android applications:
  - The Android APIs — The core of the SDK is the Android API libraries that provide developer access to the Android stack.
  - Development tools — The SDK includes several development tools that let you compile and debug your applications so that you can turn Android source code into executable applications.

INTRODUCING THE DEVELOPMENT FRAMEWORK

- The Android Virtual Device Manager and emulator —
  - The Android emulator is a fully interactive mobile device emulator featuring several alternative skins.
  - The emulator runs within an Android Virtual Device (AVD) that simulates a device hardware configuration.
  - Using the emulator you can see how your applications will look and behave on a real Android device.
  - All Android applications run within the Dalvik VM

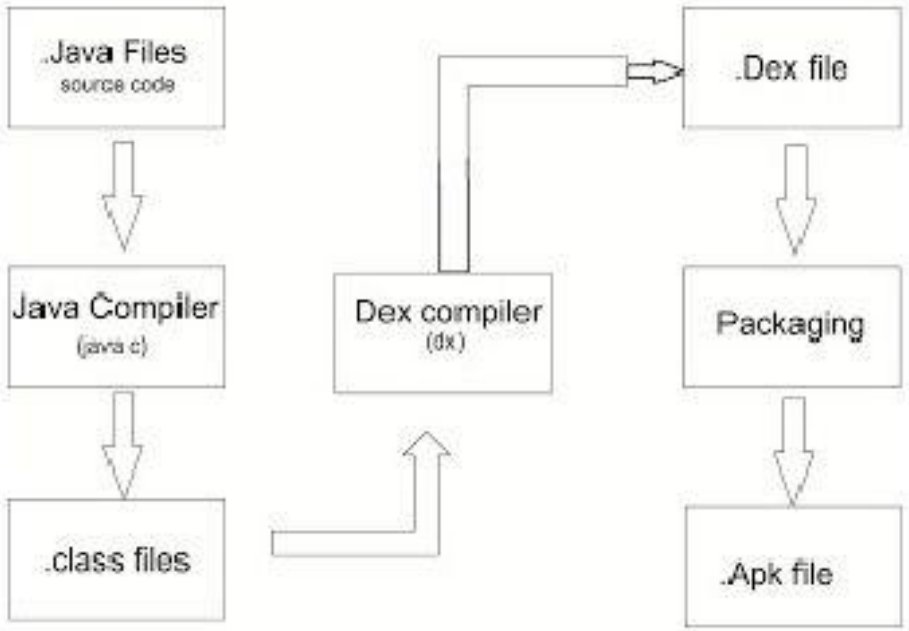
INTRODUCING THE DEVELOPMENT FRAMEWORK

- Full documentation — The SDK includes extensive code-level reference information detailing exactly what's included in each package and class and how to use them.
- Sample code
- Online support — Android has rapidly generated a vibrant developer community.

The Dalvik Virtual Machine

- Android uses its own custom VM designed to ensure that multiple instances run efficiently on a single device.
- The Dalvik VM executes Dalvik executable files, a format optimized to ensure minimal memory footprint.
- You create .dex executables by transforming Java language compiled classes using the tools supplied within the SDK
- Dalvik Virtual Machine uses its own byte-code and runs “.dex”(Dalvik Executable File) file.
- DDMS (Dalvik Debug Monitoring Service) :debug android applications

The Dalvik Virtual Machine



Study the Terms

- SDK manager
- Android Emulator
- Android Virtual Device (AVD)
- Android Debug Bridge (ABD)
- Android Developer Tool (ADT)

Thank You