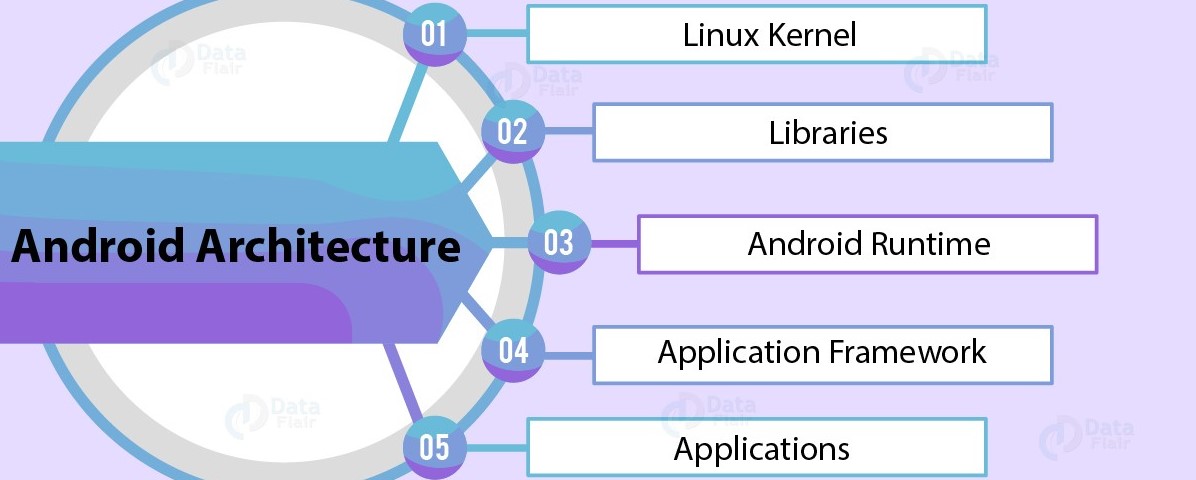
**ANDROID ARCHITECTURE**

**OVERVIEW**

Android is a mobile operating system that has an open-source framework and is based on [Linux](https://data-flair.training/blogs/linux-commands-tutorial/)which helps us to develop advanced and user-friendly applications, having a collection of a number of C/C++ libraries.

Android Architecture is divided into five components, **Applications, Application Framework, Android Runtime, Platform Libraries, Linux Kernel.**

****

**APPLICATIONS:**

It is the top-most layer of Android architecture. All the third-party applications downloaded from the play store like games, messenger, etc, and pre-installed applications like home, contacts, camera, etc, will be installed on this layer. It runs within the Android runtime with the help of the classes and services provided by the application framework.

**APPLICATION FRAMEWORK:**

The application framework built on top of the native library layer provides us with an Application programming interface and higher-level services. It provides the services with the help of which we can create a particular class and, Android developers use these high-level services to build applications.

It also consists of an Android **Hardware Abstraction Layer (HAL)** that allows the Android Application framework to communicate with hardware-specific device drivers. It acts as an interface for hardware vendors to implement. An android application uses HAL APIs to get commands from different hardware devices.

The application framework consists of the following key services:

* **Activity Manager:** The method in this class uses testing and debugging methods.
* **Content provider:** It provides data from the application to other layers.
* **Resource Manager:** It provides access to non-code resources as such string, color setting, and user interface layouts.
* **Notification Manager:** The users get notifications about all the actions happening in the background.
* **View System:** It acts as a base class for widgets and is responsible for event handling.

**ANDROID RUNTIME:**

This is the third section of the architecture and is available on the second layer from the bottom. It comprises DVM (Dalvik Virtual Machine). Just like Java Virtual Machine, Android uses DVM to optimize battery life, memory, and performance. It depends on the Linux kernel layer for threading and low-level memory management. The core libraries enable us to implement android applications using the standard JAVA or Kotlin programming languages.

**LIBRARIES:**

This layer consists of a set of Libraries and Android Runtime. The Android component is built using **native codes** and requires **native libraries**, which are written in C/C++ and most of the libraries are open-source libraries. Also, this layer handles data that is specific to the hardware.

A summary of some key core Android libraries available to the Android developer is as follows −

* **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.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.text** − Used to render and manipulate text on a device display.
* **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.

A summary of some key core Java-based libraries to provide support for android development is as follows –

* **Media** - Library provides support to play and record audio and video formats.
* **Surface manager** - Responsible for managing access to the display subsystem.
* **SGL** and **OpenGL** - Both cross-language, cross-platform application program interface (API) is used for 2D and 3D computer graphics.
* **SQLite** - Provides database support.
* **FreeType** - Provides font support.
* **Web-Kit** - This open-source web browser engine provides all the functionality to display web content and to simplify page loading.
* **SSL (Secure Sockets Layer)** – It is a security technology to establish an encrypted link between a web server and a web browser.

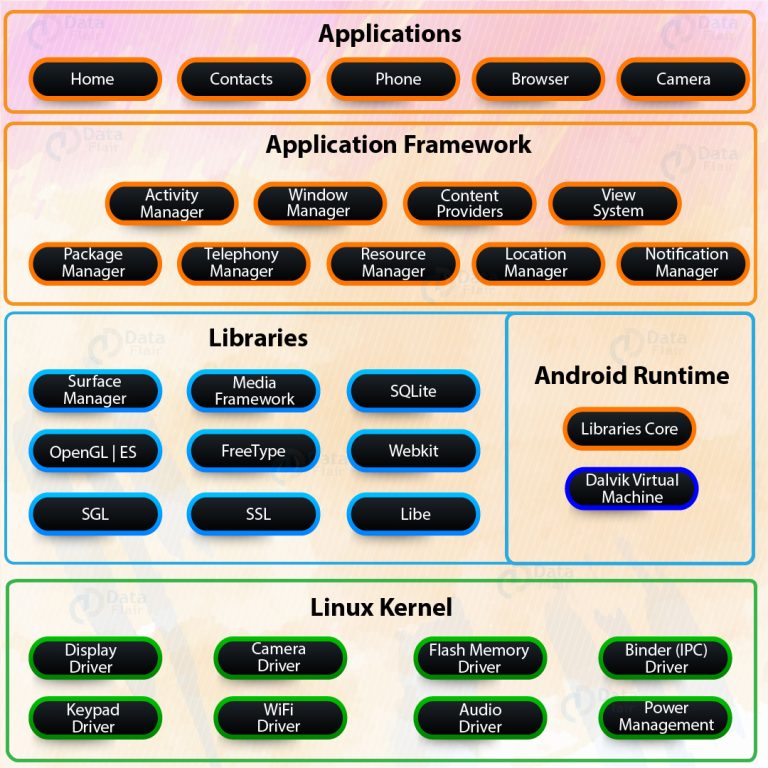
**LINUX:**

At the bottom of the layers is Linux. Among all the components Linux kernel provides the main functionality of operating system functions. It also provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display, etc.

It provides features such as:

* **Security:** It provides security between the application and the system.
* **Memory Management:** It manages the memory efficiently and thus provides the freedom to develop our apps.
* **Process Management:** It allocates resources to processes whenever they need them and manages the process well.
* **Network Stack:** It effectively handles network communication.
* **Driver Model:** It ensures that the application works properly on the device and hardware manufacturers responsible for building their drivers into the Linux build.

**SUMMARY**

****