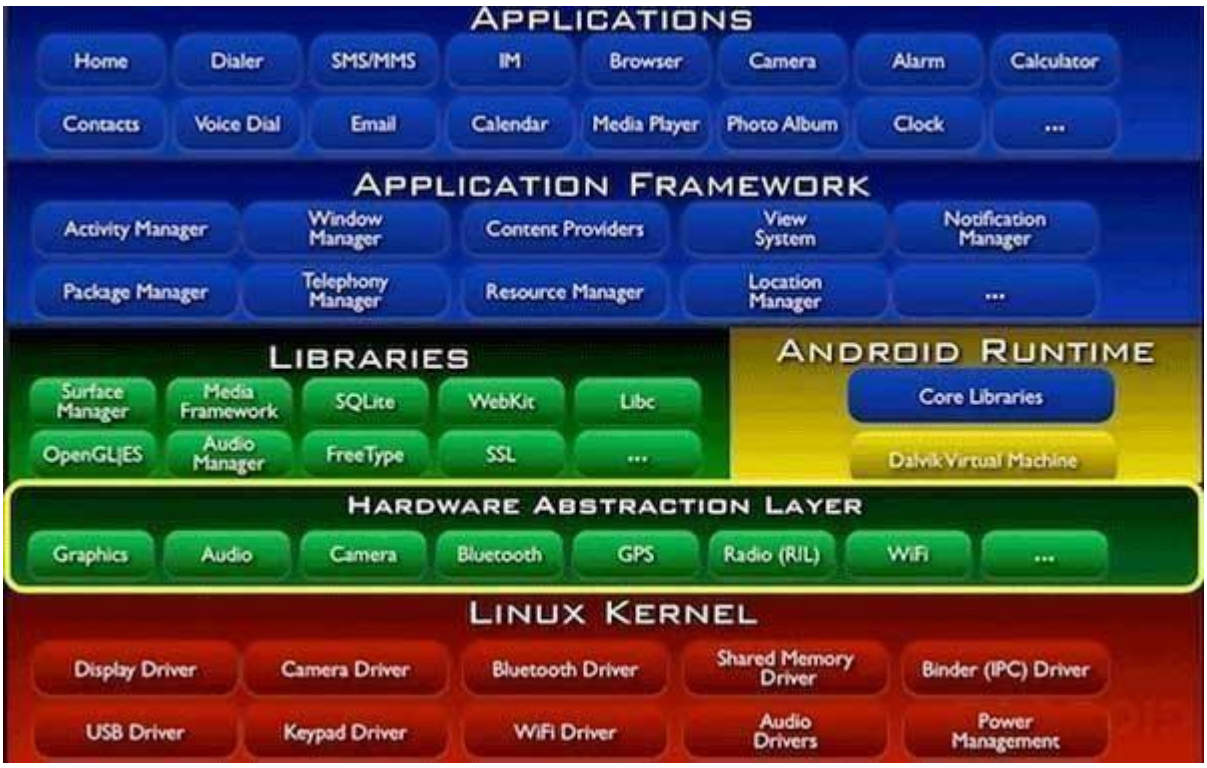
## Experiment No. 12

**Aim:** Case Study: Android Stack .

**Theory:**

### Introduction to Android Stack

The Android software stack generally consists of a Linux kernel and a collection of C/C++ libraries that are exposed through an application framework that provides services, and management of the applications and run time




The Android operating system is like a cake consisting of various layers. Each layer has its own characteristics and purpose— but the layers are not always cleanly separated and often seep into one another.

Although Android is based on Linux, it is not just another flavor of Linux, in the way that Ubuntu, Fedora, or Red Hat are.

Many things you'd expect from a typical Linux distribution aren't available in Android, such as the X11 window manager, the ability to add a person as a Linux user or even the glibc standard C library.

On the other hand, Android adds quite a bit to the Linux kernel, such as an improved power management that is well-suited for mobile battery-powered devices, • a very fast interprocess communication mechanisms

Mechanisms for sand‗ boxing applications so they are isolated from one another.

## Layers in the Android Stack

The Android stack, as the folks over at Google call it, has a number of layers, and each layer groups together several programs. In this tutorial I'll walk you through the various layers in Android stack and the functions they are responsible for. Following are the different layers in the Android stack:

- Linux Kernel Layer
- Native Layer
- Application Framework Layer
- Applications layer

Kernel Layer

At the bottom of the Android stack is the Linux Kernel. It never really interacts with the users and developers, but is at the heart of the whole system. Its importance stems from the fact that it provides the following functions in the Android system:

- Hardware Abstraction
- Memory Management Programs
- Security Settings
- Power Management Software
- Other Hardware Drivers (Drivers are programs that control hardware devices.)
- Support for Shared Libraries
- Network Stack

With the evolution of Android, the Linux kernels it runs on have evolved too.Here is a Table highlighting the different Kernel versions.

| Android Version | Linux Kernel Version |
|---|---|
| 1.0 | 2.6.25 |
| 1.5 (Cupcake) | 2.6.27 |
| 1.6 (Donut) | 2.6.29 |
| 2.2 (Froyo) | 2.6.32 |
| 2.3 (Gingerbread) | 2.6.35 |
| 3.0 (Honeycomb) | 2.6.36 |
| 4.0.x (Ice Cream Sandwich) | 3.0.1 |
| 4.1./4.2 (Jelly Bean) | 3.0.31 |

The Android system uses a binder framework for its Inter-Process Communication (IPC) mechanism. The binder framework was originally developed as OpenBinder and was used for IPC in BeOS.

## Hardware Abstraction Layer

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.

## Native Libraries Layer


The next layer in the Android architecture includes Android‘s native libraries. Libraries carry a set of instructions to guide the device in handling different types of data. For instance, the playback and recording of various audio and video formats is guided by the Media Framework Library.
**Open Source Libraries:**

- Surface Manager: composing windows on the screen
- SGL: 2D Graphics

- Open GL|ES: 3D Library
- Media Framework: Supports playback and recording of various audio, video and picture formats.
- Free Type: Font Rendering
- WebKit: Browser Engine
- libc (System C libraries)
- SQLite ● Open SSL

### Application Runtime Layer

Located on the same level as the libraries layer, the Android runtime layer includes a set of core Java libraries as well. Android application programmers build their apps using the Java programming language. It also includes the Dalvik Virtual Machine.


**What is Dalvik VM?**

Dalvik is open-source software. Dan Bornstein, who named it after the fishing village of Dalvík in Eyjafjörður, Iceland, where some of his ancestors lived, originally wrote Dalvic VM. It is the software responsible for running apps on Android devices.

- It is a Register based Virtual Machine.
- It is optimized for low memory requirements.
- It has been designed to allow multiple VM instances to run at once.
- Relies on the underlying OS for process isolation, memory management and threading support. ● Operates on DEX files.

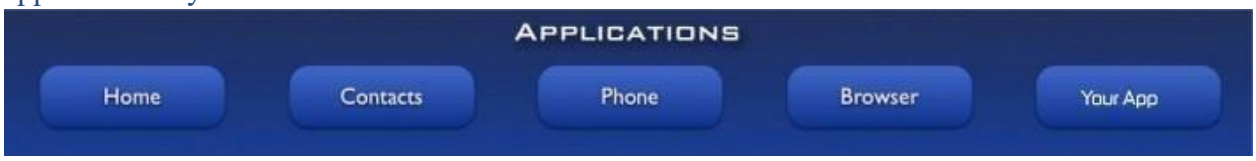
### Application Framework Layer

Our applications directly interact with these blocks of the Android architecture. These programs manage the basic functions of phones like resource management, voice call management etc.

**Important blocks of Application Framework:**

- **Activity Manager:** Manages the activity life cycle of applications. To understand the Activity component in Android in detail click here.
- **Content Providers:** Manage the data sharing between applications. Our Post on Content Provider component describes this in greater detail
- **Telephony Manager:** Manages all voice calls. We use a telephony manager if we want to access voice calls in our application.
- **Location Manager:** Location management, using GPS or cell tower
- **Resource Manager:** Manage the various types of resources we use in our Application

## Application Layer




The applications are at the topmost layer of the Android stack. An average user of the Android device would mostly interact with this layer (for basic functions, such as making phone calls, accessing the Web browser etc.). The layers further down are accessed mostly by developers, programmers and the likes.

Several standard applications come installed with every device, such as:

- SMS client app
- Dialer
- Web browser
- Contact manager

**Conclusion:**

We have successfully learned about the Android Stack. Also about layers of android stack.