**UNIT-I**

**Introduction**

**Android** is a complete set of software for mobile devices such as tablet computers, notebooks, smartphones, electronic book readers, set-top boxes etc. It contains a **Linux-based Operating System**, **middleware** and **key mobile applications**.

It can be thought of as a mobile operating system. However, it is not limited to mobile only. It is currently used in various devices such as mobiles, tablets, televisions etc.

Android was developed by the *Open Handset Alliance(*It's a consortium of 84 companies such as google, samsung, AKM, synaptics, KDDI, Garmin, Teleca, Ebay, Intel etc.), led by Google, and other companies.

It allows developers to write managed code in the Java language, controlling the device via Google-developed Java libraries.

**History of Android**

The code names of android ranges from A to J currently, such as **Aestro**, **Blender**, **Cupcake**, **Donut**, **Eclair**, **Froyo**, **Gingerbread**, **Honeycomb**, **Ice Cream Sandwitch**, **Jelly Bean**, **KitKat** and **Lollipop**. Let's understand the android history in a sequence.

* Initially, **Andy Rubin** founded Android Incorporation in Palo Alto, California, United States in October, 2003.
* In 17th August 2005, Google acquired android Incorporation. Since then, it is in the subsidiary of Google Incorporation.
* The key employees of Android Incorporation are **Andy Rubin**, **Rich Miner**, **Chris White** and **Nick Sears**.
* Originally intended for camera but shifted to smart phones later because of low market for camera only.
* Android is the nick name of Andy Rubin given by coworkers because of his love to robots.
* In 2007, Google announces the development of android OS.
* In 2008, HTC launched the first android mobile.

## Android Versions, Codename and API

Let's see the android versions, codenames and API Level provided by Google.

|  |  |  |
| --- | --- | --- |
| **Version** | **Code name** | **API Level** |
|  |  |  |
| 1.5 | Cupcake | 3 |
| 1.6 | Donut | 4 |
| 2.1 | Éclair | 7 |
| 2.2 | Froyo | 8 |
| 2.3 | Gingerbread | 9 and 10 |
| 3.1 and 3.3 | Honeycomb | 12 and 13 |
| 4.0 | Ice Cream Sandwitch | 15 |
| 4.1, 4.2 and 4.3 | Jelly Bean | 16, 17 and 18 |
| 4.4 | KitKat | 19 |
| 5.0 | Lollipop | 21 |
| 6.0 | Marshmallow | 23 |
| 7.0 | Nougat | 24-25 |
| 8.0 | Oreo | 26-27 |

**Core Components**

Android applications are composed of one or more of four types of core components.

**Activity: -** An Activity represents the screen on your Android application, which has its user interface. An application, for instance, an Email App can have many activities such as opening an email, composing an email, replying to an email – these all are different activities. Therefore, every Android application has more than one activity. When we start a new activity (like replying to an email), previous activity is pushed to the back stack and it is stopped until the new activity is finished.

#### **Services:**- The other important component of an Android application is the service. It performs running operations (long or short) in the background for the activity that you perform on your screen. For example, a push notification from an email. It is possible that service still runs while you have terminated the application or you are not using it currently. For example, when you get an email, you are notified while still, you are not using the application currently.

**Content provider: -** Content Providers manage the application Data and encapsulate it (Object Oriented Feature). This provides the data from one processor of an application to another one. The data might be stored in Database or in a file system or any other storage management systems. Android devices include several native Content Providers that expose useful databases such as the media store and contacts.

**Broadcast receiver: -** A component that responds to system conditions such as low battery or the screen being turned off. You can use broadcast receivers to initiate a response from a running application, such as if a picture has been taken. A broadcast receiver that a developer writes is implemented as a subclass of the Android BroadcastReceiver class.



**Features of Android**

Android is a powerful operating system competing with Apple 4GS and support great features.

* It is open-source.
* Anyone can customize the Android Platform.
* The consumer can choose many mobile applications.
* It provides many interesting features like weather details, opening screen, live RSS (Really Simple Syndication) feeds etc.
* It provides support for messaging services (SMS and MMS), web browser, storage (SQLite), connectivity (GSM, CDMA, Blue Tooth, and Wi-Fi etc.), media, handset layout etc.
* **Beautiful UI**: Android OS basic screen provides a beautiful and intuitive (Natural) user interface.
* **Storage**: SQLite, a lightweight relational database, is used for data storage purposes.
* **Multi-tasking**: User can jump from one task to another and same time various applications can run simultaneously.
* **Multi-touch**: Android has native support for multi-touch, which was initially made available in handsets such as the HTC Hero.
* **Messaging**: SMS and MMS
* **Wi-Fi Direct**: A technology that let apps discover and pair directly, over a high-bandwidth peer-to-peer connection.
* **Media support**: MPEG, AMR, MP3, WAV, JPEG, PNG, GIF and BMP.
* **Connectivity**: GSM/EDGE, CDMA, UMTS, Bluetooth, Wi-Fi and WiMAX.
* **Resizable widgets**: Widgets are resizable, so users can expand them to show more content or shrink them to save space.
* **Multi-Language** Support single direction and bi-directional text.

## Categories of Android applications

There are many android applications in the market. The top categories are:

* Entertainment
* Tools
* Communication
* Productivity
* Personalization
* Music and Audio
* Social
* Media and Video
* Travel and Local etc.

**Android Applications:** Android applications are usually developed in the Java language using the Android Software Development Kit. Once developed, Android applications can be packaged easily and sold out either through a store such as **Google Play** or the **Amazon Appstore**.

**Environment Setup**

Whenever we are going to start our Android application development on either of the following operating systems:

* Microsoft Windows XP or later version.
* Mac OS X 10.5.8 or later version with Intel chip.
* Linux including GNU C Library 2.7 or later.

**Android Development Tools**

To start Android Application we required the following tools. They are

* Java JDK 1.5 or 1.6
* Android SDK
* Android Studio (optional android SDK is inbuilt)
* Eclipse (optional)

**Android SDK**

The *Android Software Development Kit* (SDK) contains the necessary tools to create, compile and package Android application. Most of these tools are command line based.

The Android SDK also provides an Android device emulator, so that Android applications can be tested without a real Android phone. You can create *Android virtual devices* (AVD) via the Android SDK, which run in this emulator.

The Android SDK contains the *Android debug bridge* (adb) tool which allows to connect to an virtual or real Android device.

**Android Development Tools:** Google provides the *Android Development Tools* (ADT) to develop Android applications with Eclipse.

ADT contains all required functionalities to create, compile, debug and deploy Android applications from the Eclipse IDE. ADT also allows creating and starting AVDs.

The Android Development Tools (ADT) provides specialized editors for resources files, e.g. layout files. These editors allow switching between the XML representation of the file and a richer user interface via tabs on the bottom of the editor.

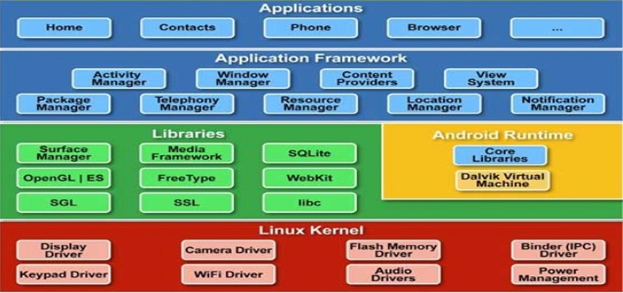
**Dalvik Virtual Machine**

The Android system uses a special virtual machine, i.e. the *Dalvik Virtual Machine* to run Java based applications. Dalvik uses an own byte code format which is different from Java byte code.

Therefore you cannot directly run Java class files on Android; they need to get converted in the Dalvik byte code format.

**Architecture:**

Android operating system is a stack of software components, which is roughly divided into five sections and four main layers. They are

****

The Android OS can be referred to as a software stack of different layers, where each layer is a group of several program components. Together it includes operating system, middleware and important applications. Each layer in the architecture provides different services to the layer just above it. We will examine the features of each layer in detail.

**Linux Kernel**

The basic layer is the Linux kernel. The whole Android OS is built on top of the Linux 2.6 Kernel with some further architectural changes made by Google. It is this Linux that interacts with the hardware and contains all the essential hardware drivers. Drivers are programs that control and communicate with the hardware. Inside the Linux kernel, it contains all the low-level device drivers meant for the various hardware components of an Android Device. (Extra Info: A driver is software which is meant to interconnect between Software System and the peripheral like camera, printer, Bluetooth, etc.) For example, consider the Bluetooth function. All devices have Bluetooth hardware in it. Therefore the kernel must include a Bluetooth driver to communicate with the Bluetooth hardware. The Linux kernel also acts as an abstraction layer between the hardware and other software layers. Android uses the Linux for all its core functionality such as Memory management, process management, networking, security settings etc.

**Libraries**

The next layer is the Android’s native libraries. It is this layer that enables the device to handle different types of data. These libraries are written in c or c++ language and are specific for a particular hardware.

Some of the important native libraries include the following:

**Surface Manager:**

The surface manager is responsible for composing different drawing surfaces on to the screen. So it's the surface manager that's responsible for taking different windows that are owned by different applications that are running in different processes and all drawing at different times and making sure the pixels end upon the screen when they are supposed to.

It is used for compositing window manager with off-screen buffering. Off-screen buffering means you can’t directly draw into the screen, but your drawings go to the off-screen buffer. There it is combined with other drawings and form the final screen the user will see. This off screen buffer is the reason behind the transparency of windows.

**Media framework:**

Media framework provides different media codec’s allowing the recording and playback of different media formats. Packet Video, one of the members of the open handset alliance, provided the Media Framework and that contains the entire codex that make up the core of the media experiences.

SQLite: SQLite is the database engine used in android for data storage purposes

WebKit: It is the browser engine used to display HTML content

OpenGL: Used to render 2D or 3D graphics content to the screen

**Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, on. One interesting thing about the Android graphics platform is that you can combine 3D and 2D graphics in the same application.

**Core Java Libraries:** These are different from Java SE and Java ME libraries. However these libraries provide most of the functionalities defined in the Java SE libraries. The core library contains all of the collection classes, utilities, IO, all the utilities and tools that you’ve come to expect to use.

**Application Framework**

Our application directly interacts with these blocks. These programs manage the basic functions of phone like resource management, voice call management etc. As a developer, you just consider these are some basic tools with which we are building our applications.

Important blocks of Application framework are:

* **Activity Manager**: Manages the activity life cycle of applications. It also maintains a common back stack so that application that is running in different processes can have a smoothly integrated navigation experience.
* PACKAGE MANAGER: The package manager is what keeps track of which applications are installed on your device. So, if you download new applications over the air or otherwise install apps, it's the package manager that's responsible for keeping track of what you have and what the capabilities of each of your applications are.
* **Content Providers**: Manage the data sharing between applications
* **Telephony Manager**: Manages all voice calls. We use telephony manager if we want to access voice calls in our application. The telephony manager contains the APIs that we use to build the phone application that's central to the phone experience.
* **Location Manager**: Location management, using GPS or cell tower
* **Resource Manager**: The resource manager is what they use to store local iStrings, bitmaps, and layout file descriptions, all of the external parts of an application that aren't code.

**Applications:** At the top of Android Architecture we have all the applications, which are used by the final user. By installing different applications, the user can turn his mobile phone into the unique, optimized and smart mobile phone. All applications are written using the Java programming language.

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

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