IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 3, March 2022

DOI: 10.17148/IJARCCE.2022.11391

Android Operating System and Development Environment

Saji M.G.[1], Dipu Jose[2]

Lecturer in Computer Engineering, Govt Polytechnic College, Nedumangad[1]

Lecturer in Computer Engineering, Govt Polytechnic College, Nedumangad[2]

Abstract: A mobile operating system, also called a mobile OS, is an operating system that is specifically designed to run on mobile devices such as mobile phones, smartphones, PDAs, tablet computers and other handheld devices. Android is an open source and Linux-based Operating System. Android was developed by the Open Handset Alliance, led by Google, and other companies. Android Studio is the IDE only used for android application development, contains design tools, flexible build system, and emulator. Android studio versions are available for windows, linux, and Mac Os, etc.

Keywords: Android operating system, android studio, IDE

INTRODUCTION

Android is a mobile operating system developed by Google, with linux kernel and other open source software and designed for mobile devices. The Android OS was originally created by Android, Inc., which was bought by Google in 2005.The android is a powerful operating system and it supports large number of applications in Smart phones. Android studio is an integrated development environment exclusively for android application development. Versions of android studio with Android SDK, AVD are available for different operating systems. AVD is an emulator that allows developers to test the application by simulating the real device capabilities.

Types of Mobile Operating Systems

1. Android OS (Google Inc.)

The Android mobile operating system is Google's open and free software stack that includes an operating system, middleware and also key applications for use on mobile devices, including smartphones.

2. Bada (Samsung Electronics)

Bada is a proprietary Samsung mobile OS that was first launched in 2010. The Samsung Wave was the first smartphone to use this mobile OS. Bada provides mobile features such as multipoint-touch, 3D graphics and of course, application downloads and installation.

3. BlackBerry OS (Research In Motion)

The BlackBerry OS is a proprietary mobile operating system developed by research in motion for use on the company's popular BlackBerry handheld devices.

4. iPhone OS / iOS (Apple)

Apple's iPhone OS was originally developed for use on its iPhone devices. Now, the mobile operating system is referred to as iOS and is supported on a number of Apple devices including the iPhone, iPad, iPad 2 and iPod Touch. The iOS mobile operating system is available only on Apple's own manufactured devices

5. Symbian OS (Nokia)

Symbian is a mobile operating system (OS) targeted at mobile phones that offers a high-level of integration with communication and personal information management (PIM) functionality.

6. Windows Mobile (Windows Phone)

Windows Mobile is Microsoft's mobile operating system used in smartphones and mobile devices – with or without touchscreens. In 2010 Microsoft announced a new smartphone platform called Windows Phone 7.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 3, March 2022

DOI: 10.17148/IJARCCE.2022.11391

ANDROID OPERATING SYSTEM

Android is an open source and Linux-based Operating System for mobile devices such as smartphones and tablet computers. Android was developed by the Open Handset Alliance, led by Google, and other companies.

Features of Android

- **Messaging**

SMS and MMS are available forms of messaging, including threaded text messaging and Android Cloud To Device Messaging (C2DM)

- **Web browser**

The web browser available in Android is based on the open-source Blink (previously WebKit) layout engine, coupled with Chrome's V8 JavaScript engine.

- **Voice-based features**

Google search through voice has been available since initial release. Voice actions for calling, texting, navigation, etc. are supported on Android 2.2 onwards.

- **Multi-touch**

Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero.

- **Multitasking**

Multitasking of applications, with unique handling of memory allocation, is available.

- **Screen capture**

Android supports capturing a screenshot by pressing the power and home-screen buttons at the same time.

- **TV recording**

Android TV supports capturing video and replaying it.

- **Video calling**

Android does not support native video calling, but some handsets have a customized version of the operating system that supports it, either via the UMTS network (like the Samsung Galaxy S) or over IP.

- **Multiple language support**

Android supports multiple languages

- **Accessibility**

Built-in text-to-speech is provided by Talk Back for people with low or no vision.

- **Storage**

SQLite, a lightweight relational database, is used for data storage purposes.

- **Wi-Fi Direct**

A technology that let apps discover and pair directly, over a high-bandwidth peer-to-peer connection.

Android architecture

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

<table>
<tr><td colspan="2">Applications</td></tr>
<tr><td colspan="2">Applications Framework</td></tr>
<tr><td>Native Libraries</td><td>Android Runtime</td></tr>
<tr><td colspan="2">Linux Kernel</td></tr>
</table>

1. Linux kernel

At the bottom of the layers is Linux - Linux 2.6 with approximately 115 patches. This provides basic system functionality like process management, memory management, device management like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at, such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

2. Libraries

On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security, etc.

3. Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called Dalvik Virtual Machine which is a kind of Java Virtual Machine specially designed and

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

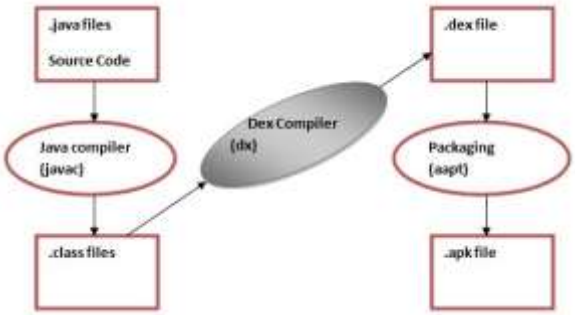
Impact Factor 7.39 ⁜ Vol. 11, Issue 3, March 2022

DOI: 10.17148/IJARCCE.2022.11391

optimized for Android. The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Dalvik Virtual Machine

The **Dalvik Virtual Machine (DVM)** is an android virtual machine optimized for mobile devices. It optimizes the virtual machine for memory, battery life and performance. The Dex compiler converts the class files into the .dex file that run on the Dalvik VM. Multiple class files are converted into one dex file. Below figure shows the compiling and packaging process from the source file:



The **javac tool** compiles the java source file into the class file.
The **dx tool** takes all the class files of your application and generates a single .dex file. It is a platform-specific tool.
The **Android Assets Packaging Tool (aapt)** handles the packaging process.

4. Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications. Applications You will find all the Android application at the top layer. You will write your application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, Games, etc.

ANDROID APPLICATION COMPONENTS

The **core building blocks** or **fundamental components** of android are activities, views, intents, services, content providers, fragments and AndroidManifest.xml.

- **Activity**

An activity is a class that represents a single screen. It is like a Frame in AWT.

- **View**

A view is the UI element such as button, label, text field etc. Anything that you see is a view.

- **Intent**

Intent is used to invoke components. It is mainly used to:

- Start the service
- Launch an activity
- Display a web page
- Display a list of contacts
- Broadcast a message
- Dial a phone call etc.

For example, you may write the following code to view the webpage.

```
Intent intent=new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("http://www.javatpoint.com"));
startActivity(intent);
```

- **Service**

Service is a background process that can run for a long time. There are two types of services local and remote. Local service is accessed from within the application whereas remote service is accessed remotely from other applications running on the same device.

- **Content Provider**

Content Providers are used to share data between the applications.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 3, March 2022

DOI: 10.17148/IJARCCE.2022.11391

- **Fragment**

Fragments are like parts of activity. An activity can display one or more fragments on the screen at the same time.

- **AndroidManifest.xml**

It contains information about activities, content providers, permissions etc. It is like the web.xml file in Java EE.

- **Android Virtual Device (AVD)**

It is used to test the android application without the need for mobile or tablet etc. It can be created in different configurations to emulate different types of real devices.

- **Android Emulator**

Android Emulator is used to run, debug and test the android application. If you don't have the real device, it can be the best way to run, debug and test the application.

Different IDEs for Android application development

- Eclipse
- Android SDK – Android studio
- Android Development Tools (ADT)

Eclipse

Eclipse is an integrated development environment (IDE) used in computer programming, and is the most widely used Java IDE. Eclipse is written mostly in Java and its primary use is for developing Java applications

Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system designed specifically for Android development. Android Studio was announced on May 16, 2013 at the Google I/O conference

Android SDK

The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android.

Android Development Tools (ADT)

It is a plugin for the Eclipse IDE that is designed to give you a powerful, integrated environment in which to build Android applications. ADT extends the capabilities of Eclipse to let you quickly set up new Android projects, create an application UI, add packages based on the Android Framework API, debug your applications using the Android SDK tools, and even export signed (or unsigned) .apk files in order to distribute your application.

Android Virtual Device

An Android Virtual Device (AVD) is an emulator configuration that allows developers to test the application by simulating the real device capabilities. We can configure the AVD by specifying the hardware and software options. AVD manager enables an easy way of creating and managing the AVD with its graphical interface. We can create as many AVDs as we need, based on the types of device we want to test for. This emulator is to provide a virtual device-specific environment in which to install and run Android apps.

Creating new project in Android Studio

1. Start your Android Studio. It will show you the default start window
2. Click on Start a new Android Studio project.
3. Provide project information or configuration.
   - Put Application Name - Without Space, Capitalized letter format
   - Company Domain,
   - Package Name,
   - Project Location,
   - choose platform where you want to run the android application
   - choose the activity – Blank Activity, you will get auto generated codes
   - Select layout name, activity name etc.
4. Click on Finish button

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 3, March 2022

DOI: 10.17148/IJARCCE.2022.11391

Folders and their files created by an android project

- **src** — Contains the .java source files for your project. You will write the code for your application in this file.
- Android library — This item contains one file, android.jar, which contains all the class libraries needed for an Android application.
- **gen** — Contains the R.java file, a compiler-generated file that references all the resources found in your project. You should not modify this file.
- **assets** — This folder contains all the assets used by your application, such as HTML, text files, databases, etc.
- **res** — This folder contains all the resources used in your application. It also contains a few other subfolders: drawable-<resolution> , layout , and values .
- **AndroidManifest.xml** — This is the manifest file for your Android application. Here you specify the permissions needed by your application, as well as other features (such as intent-filters, receivers, etc.).

Android project is now a basic "Hello World" app that contains the following default files.

- app/src/main/java/com.mycompany.myfirstapp/MainActivity.java – Class definition of the activity
- app/src/main/AndroidManifest.xml – Defines the components in the App
- app/src/main/res/layout/activity_main.xml - Activity seen in design mode and text mode

File: MainActivity.java

```
package com.example.helloandroid;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.widget.TextView;
public class MainActivity extends Activity {
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
TextView textview=new TextView(this);
textview.setText("Hello Android!");
setContentView(textview);
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.activity_main, menu);
return true;
}
}
```

- **Activity** is a java class that creates and default window on the screen where we can place different components such as Button, EditText, TextView, Spinner etc. It is like the Frame of Java AWT. It provides life cycle methods for activity such as onCreate, onStop, OnResume etc.
- The **onCreate method** is called when Activity class is first created.
- The **setContentView(R.layout.activity_main)** gives information about our layout resource. Here, our layout resources are defined in activity_main.xml file.

File: activity_main.xml

```
<RelativeLayout xmlns:androclass="http://schemas.android.com/apk/res/android"     xmlns:tools="http://schemas.and
roid.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity" >
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="true"
      android:text="@string/hello_world" />
</RelativeLayout>
```

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 3, March 2022

DOI: 10.17148/IJARCCE.2022.11391

The message for this string is defined in the strings.xml file. The **@string/hello_world** provides information about the textview message. The value of the attribute hello_world is defined in the strings.xml file.

File: strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">helloandroid</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
</resources>
```

We can change the value of the hello world attribute from this file.

Generated R.java file

It is the auto-generated file that contains IDs for all the resources of res directory. It is generated by apt (Android Asset Packaging Tool). Whenever you create any component on activity main, a corresponding ID is created in the R.java file which can be used in the Java Source file later.

APK File

An apt file is created by the framework automatically. If you want to run the android application on the mobile, transfer and install it.

Resources

It contains resource files including activity main, strings, styles etc.

AndroidManifest.xml

The AndroidManifest.xml file contains information of your package, including components of the application such as activities, services, broadcast receivers, content providers etc.

It performs some other tasks also:

- It is responsible to protect the application to access any protected parts by providing the permissions.
- It also declares the android api that the application is going to use.
- It lists the instrumentation classes. The instrumentation class provides profiling and other informations. These informations are removed just before the application is published etc.

This is the required xml file for all the android application and located inside the root directory.

A simple AndroidManifest.xml file is as follows:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.javatpoint.hello"
    android:versionCode="1"
    android:versionName="1.0" >
      <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />
      <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
 </manifest>
```

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 3, March 2022

DOI: 10.17148/IJARCCE.2022.11391

Elements of the AndroidManifest.xml file

The elements used in the above xml file are described below.

- **<manifest>**

manifest is the root element of the AndroidManifest.xml file. It has package attribute that describes the package name of the activity class.

- **<application>**

application is the subelement of the manifest. It includes the namespace declaration. This element contains several subelements that declares the application component such as activity etc. The commonly used attributes are of this element are **icon**, **label**, **theme** etc.

android:icon represents the icon for all the android application components.
android:label works as the default label for all the application components.
android:theme represents a common theme for all the android activities.

- **<activity>**

activity is the sub element of application and represents an activity that must be defined in the AndroidManifest.xml file. It has many attributes such as label, name, theme, launchMode etc.

android:label represents a label i.e. displayed on the screen.
android:name represents a name for the activity class. It is a required attribute.

- **<intent-filter>**

intent-filter is the sub-element of activity that describes the type of intent to which activity, service or broadcast receiver can respond to.

- **<action>**

It adds an action for the intent-filter. The intent-filter must have at least one action element.

- **<category>**

It adds a category name to an intent-filter.

Features of Android Studio

1. Gradle-based - is an open-source build automation system - process of automating the creation of a software build
2. Android-specific refactoring and quick fixes
3. Lint to ol that detects and flags errors in programming languages to catch performance, usability, version compatibility and other problems.
4. A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations.
5. Support for building Android Wear apps - a version of Google's Android operating system designed for smart watches and other wearables.
6. Android Virtual Device to run and debug apps in the Android studio.

CONCLUSION

This paper titled “Android Operating System and Development Environment” explains mobile operating systems, android OS and development of android applications using android studio. Android is based on linux kernel, but most of the operating systems are supporting android OS and android studio IDE. Mobile technology development is rapid, hence new versions of android studio is needed to support new features. Regular updation in this field is mandatory.

REFERENCES

[1] “Beginning Android Application Development”- Wei-Meng Lee- Wrox-First Edition
[2] “Android Cookbook: Problems and Solutions for Android Developers” - written by Ian F. Darwin
[3] “Enterprise Android: Programming Android Database Applications for the Enterprise” – by Zigurd Mednieks, G. Blake Meike, Laird Dornin and Zane Pan, Publisher: Wrox
[4] “Android Programming: Pushing the Limits” – Erik Hellman- Wiley- First Edition