# Introduction to Mobile Computing with Android

**Prof. Manish Kumar Joshi,** Assistant Professor
IT & Computer Science

PU

# CHAPTER-2

# Getting Started with Android

# Overview

Android is a Linux based operating system it is designed primarily for touch screen mobile devices such as smart phones and tablet computers. The operating systems have developed a lot in last 15 years starting from black and white phones to recent smart phones or mini computers. One of the most widely used mobile OS these days is android. The android is software that was founded in Palo Alto of California in 2003.

PU

# Android Architecture

The android is a operating system and is a stack of software components which is divided into five sections and four main layers that is

- Linux Kernel
- Libraries
- Android Runtime
- Application Framework

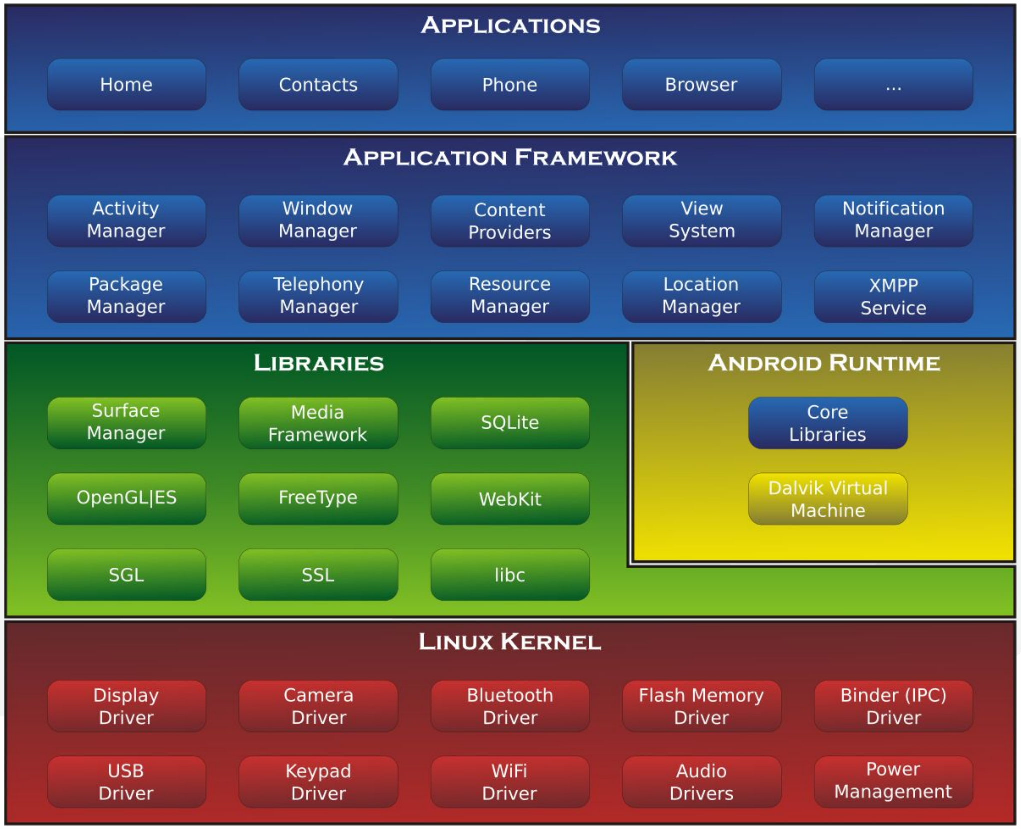
PU

# Linux Kernel

PU

The android uses the powerful Linux kernel and it supports wide range of hardware drivers. The kernel is the heart of the operating system that manages input and output requests from software. This provides basic system functionalities like process management, memory management, device management like camera, keypad, display etc the kernel handles all the things. The Linux is really good at networking and it is not necessary to interface it to the peripheral hardware. The kernel itself does not interact directly with the user but rather interacts with the shell and other programs as well as with the hard ware devices on the system.

# Application Framework



## Applications

Home | Contacts | Phone | Browser | ...

## Application Framework

Activity Manager | Window Manager | Content Providers | View System | Notification Manager

Package Manager | Telephony Manager | Resource Manager | Location Manager | XMPP Service

## Libraries

Surface Manager | Media Framework | SQLite

OpenGL|ES | FreeType | WebKit

SGL | SSL | libc

## Android Runtime

Core Libraries

Dalvik Virtual Machine

## Linux Kernel

Display Driver | Camera Driver | Bluetooth Driver | Flash Memory Driver | Binder (IPC) Driver

USB Driver | Keypad Driver | WiFi Driver | Audio Drivers | Power Management

# Libraries

The on top of a Linux kennel there is a set of libraries including open source web browser such as **webkit**, library **libc**. These libraries are used to play and record audio and video. The **SQLite** is a data base which is useful for storage and sharing of application data. The SSL libraries are responsible for internet security etc.

# Android Runtime

The android runtime provides a key component called Dalvik Virtual Machine which is a kind of java virtual machine. It is specially designed and optimized for android. The Dalvik VM is the process virtual machine in the android operating system. It is software that runs apps on android devices.

The Dalvik VM makes use of Linux core features like memory management and multithreading which is in java language. The Dalvik VM enables every android application to run it own process. The Dalvik VM executes the files in the .dex format.

# The Application Components

Application components are main building blocks of an Android application. The application manifest file loosely binds these components together.

There are mainly 4 components:

- Activities
- Services
- Broadcast Receivers
- Content Provider

PU

# Android SDK

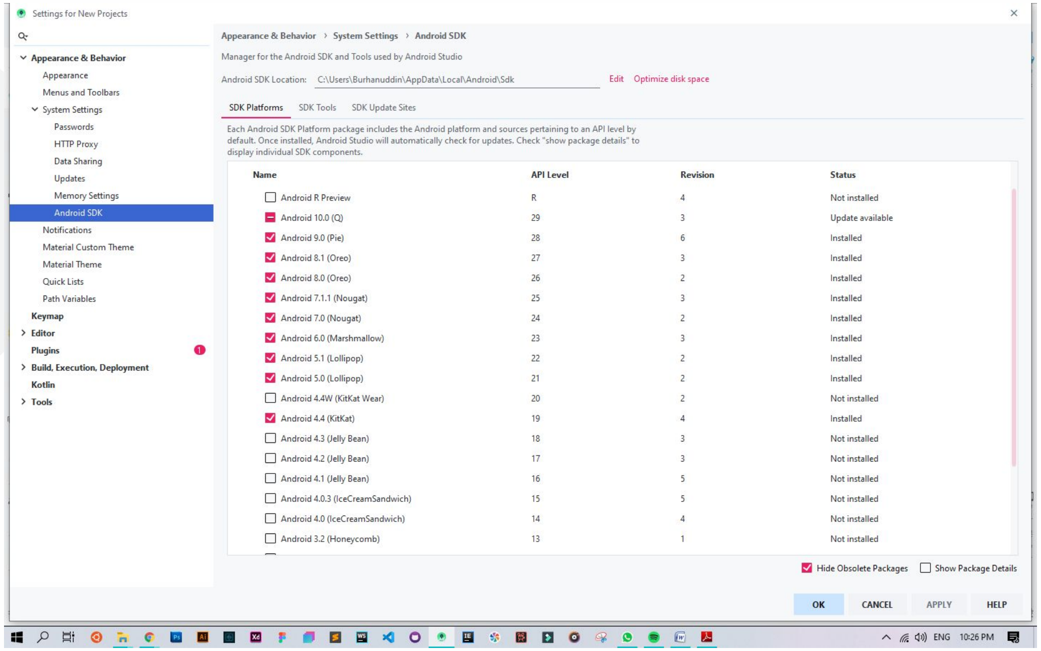
Android development starts with the Android SDK (Software Development Kit). While there are many different programming languages and a host of IDEs you can use to create an app, the SDK is a constant.

The Android SDK can be broken down into several components. These include:

- Platform-tools
- Build-tools
- SDK-tools
- The Android Debug Bridge (ADB)
- Android Emulator

# Using SDK Manager

While Android Studio will normally let you know when you need to update something, you can also manage updates to the SDK manually via the manager. You'll find this in Android Studio if you navigate to **Tools — Android — SDK Manager**. You'll see there are three tabs here for SDK Platforms, SDK Tools, and SDK Update Sites.



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: C:\Users\Burhanuddin\AppData\Local\Android\Sdk

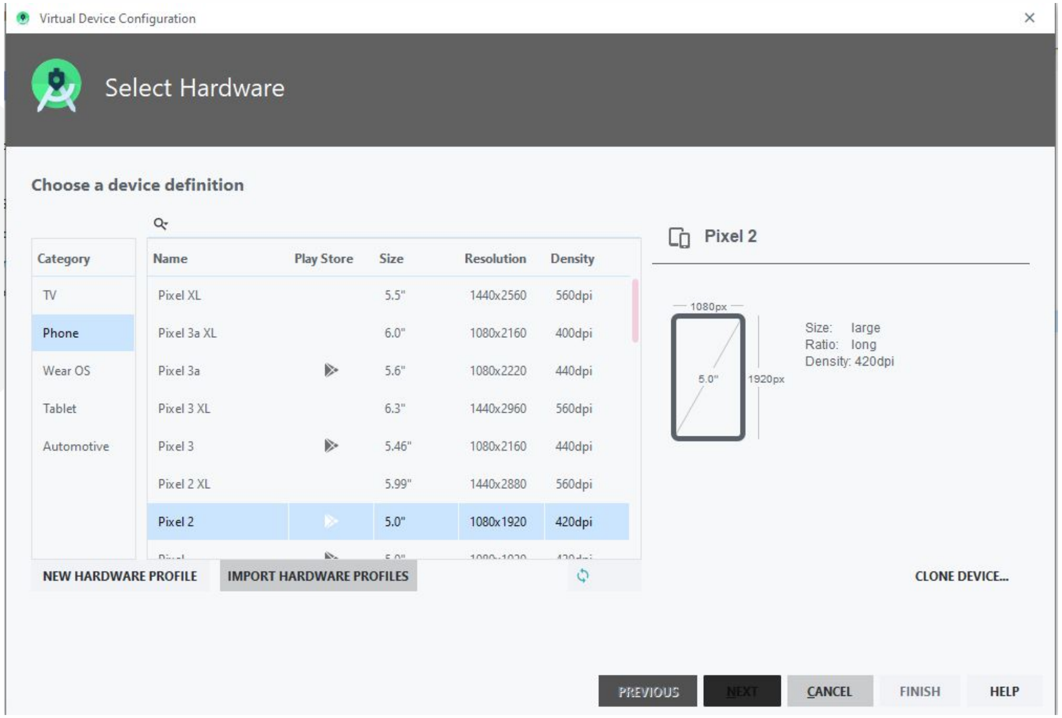
Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, Android Studio will automatically check for updates. Check "show package details" to display individual SDK components.

| Name | API Level | Revision | Status |
|---|---|---|---|
| ☐ Android R Preview | R | 4 | Not installed |
| ⊟ Android 10.0 (Q) | 29 | 3 | Update available |
| ☑ Android 9.0 (Pie) | 28 | 6 | Installed |
| ☑ Android 8.1 (Oreo) | 27 | 3 | Installed |
| ☑ Android 8.0 (Oreo) | 26 | 2 | Installed |
| ☑ Android 7.1.1 (Nougat) | 25 | 3 | Installed |
| ☑ Android 7.0 (Nougat) | 24 | 2 | Installed |
| ☑ Android 6.0 (Marshmallow) | 23 | 3 | Installed |
| ☑ Android 5.1 (Lollipop) | 22 | 2 | Installed |
| ☑ Android 5.0 (Lollipop) | 21 | 2 | Installed |
| ☐ Android 4.4W (KitKat Wear) | 20 | 2 | Not installed |
| ☑ Android 4.4 (KitKat) | 19 | 4 | Installed |
| ☐ Android 4.3 (Jelly Bean) | 18 | 3 | Not installed |
| ☐ Android 4.2 (Jelly Bean) | 17 | 3 | Not installed |
| ☐ Android 4.1 (Jelly Bean) | 16 | 5 | Not installed |
| ☐ Android 4.0.3 (IceCreamSandwich) | 15 | 5 | Not installed |
| ☐ Android 4.0 (IceCreamSandwich) | 14 | 4 | Not installed |
| ☐ Android 3.2 (Honeycomb) | 13 | 1 | Not installed |

☑ Hide Obsolete Packages ☐ Show Package Details

# Using SDK Manager

You'll likewise find the AVD Manager under **Tools — Android — AVD Manager**. This lets you build your own emulators. You'll choose the size of the device and some other specifications, and you'll be prompted to download the requisite x86 system image if it's not already installed.



Choose a device definition

Category
- TV
- Phone
- Wear OS
- Tablet
- Automotive

| Name | Play Store | Size | Resolution | Density |
|---|---|---|---|---|
| Pixel XL | | 5.5" | 1440x2560 | 560dpi |
| Pixel 3a XL | | 6.0" | 1080x2160 | 400dpi |
| Pixel 3a | ▶ | 5.6" | 1080x2220 | 440dpi |
| Pixel 3 XL | | 6.3" | 1440x2960 | 560dpi |
| Pixel 3 | ▶ | 5.46" | 1080x2160 | 440dpi |
| Pixel 2 XL | | 5.99" | 1440x2880 | 560dpi |
| Pixel 2 | ▶ | 5.0" | 1080x1920 | 420dpi |
| Pixel | ▶ | 5.0" | 1080x1920 | 420dpi |

Pixel 2

Size: large
Ratio: long
Density: 420dpi

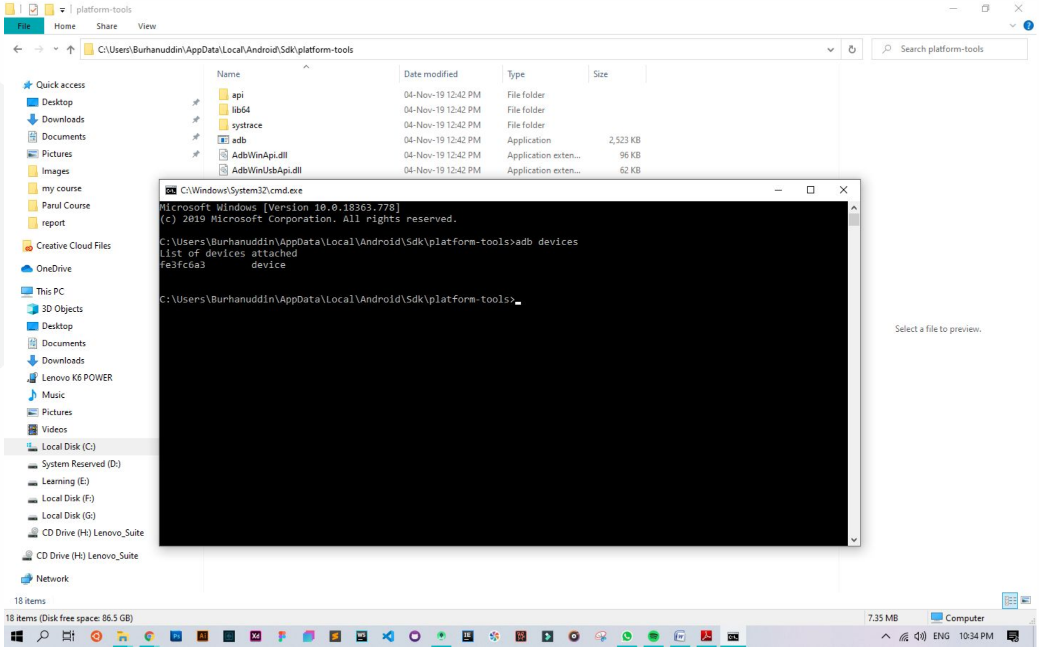
# Using The Android Device Monitor

The Android Device Monitor encapsulates DDMS and can be found under – you guessed it – **Tools — Android — DDMS**. This works with either an emulator or a connected device and will go a little deeper in monitoring the way your Android device and app are behaving.

PU

# Using ADB

Using ADB is a little different. To do this, you will need to find your Android SDK installation folder and navigate to the platform-tools directory. On Windows, hold shift and right click anywhere in the folder to open a command line. On Mac, just open Terminal from Launchpad (usually found in the Other folder).



| Name | Date modified | Type | Size |
|---|---|---|---|
| api | 04-Nov-19 12:42 PM | File folder | |
| lib64 | 04-Nov-19 12:42 PM | File folder | |
| systrace | 04-Nov-19 12:42 PM | File folder | |
| adb | 04-Nov-19 12:42 PM | Application | 2,523 KB |
| AdbWinApi.dll | 04-Nov-19 12:42 PM | Application exten... | 96 KB |
| AdbWinUsbApi.dll | 04-Nov-19 12:42 PM | Application exten... | 62 KB |

```
Microsoft Windows [Version 10.0.18363.778]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Burhanuddin\AppData\Local\Android\Sdk\platform-tools>adb devices
List of devices attached
fe3fc6a3        device


C:\Users\Burhanuddin\AppData\Local\Android\Sdk\platform-tools>_
```

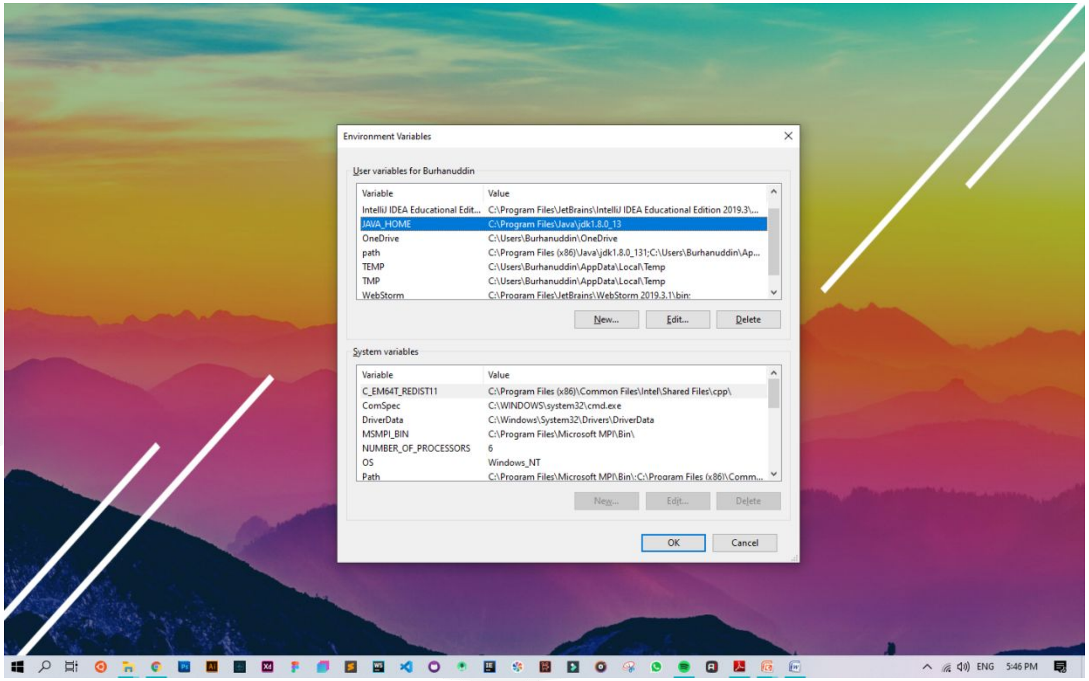
# Advantages of Android

- It supports wireless communication using 3G, 4G, 5G, Wi-Fi, EDGE and Bluetooth networks.
- It is Linux based and open-source.
- It has Media and Storage support.
- It supports all Google services like Gmail, Chrome, Location manager, Google maps, Drive, play Store etc.
- It supports multitasking and many more.

PU

# Android Studio Installation & Setup: Step 0

1. Installing Java Development Kit
2. Set environment variable “JAVA_HOME” to JDK installation directory.



Environment Variables

User variables for Burhanuddin

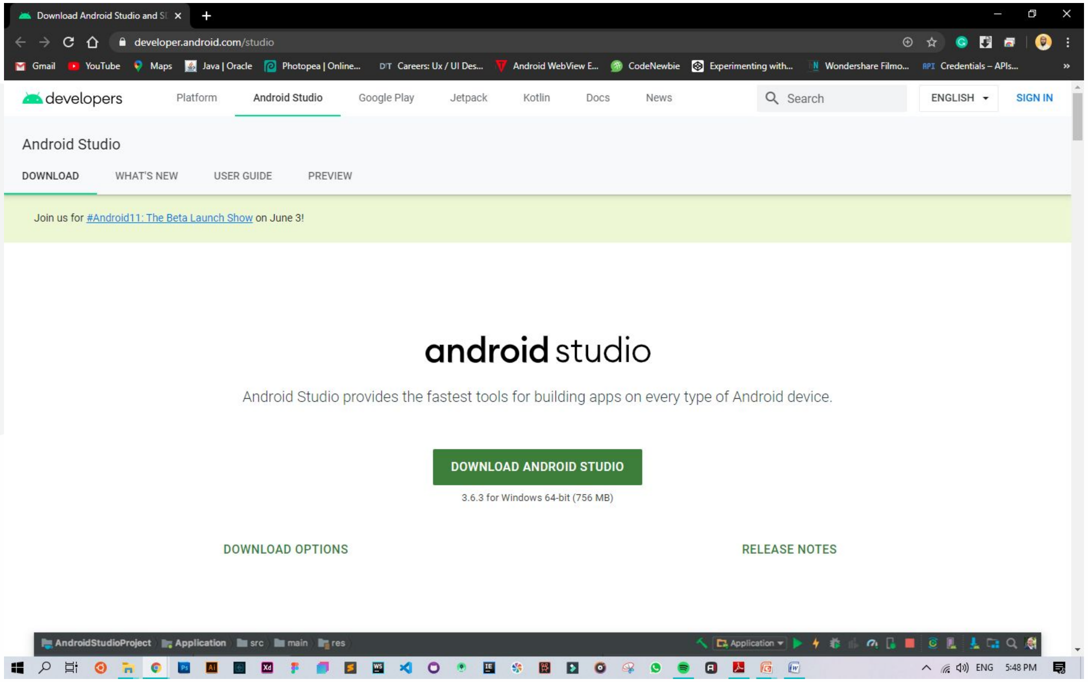
| Variable | Value |
|---|---|
| IntelliJ IDEA Educational Edit... | C:\Program Files\JetBrains\IntelliJ IDEA Educational Edition 2019.3\... |
| JAVA_HOME | C:\Program Files\Java\jdk1.8.0_13 |
| OneDrive | C:\Users\Burhanuddin\OneDrive |
| path | C:\Program Files (x86)\Java\jdk1.8.0_131;C:\Users\Burhanuddin\Ap... |
| TEMP | C:\Users\Burhanuddin\AppData\Local\Temp |
| TMP | C:\Users\Burhanuddin\AppData\Local\Temp |
| WebStorm | C:\Program Files\JetBrains\WebStorm 2019.3.1\bin; |

System variables

| Variable | Value |
|---|---|
| C_EM64T_REDIST11 | C:\Program Files (x86)\Common Files\Intel\Shared Files\cpp\ |
| ComSpec | C:\WINDOWS\system32\cmd.exe |
| DriverData | C:\Windows\System32\Drivers\DriverData |
| MSMPI_BIN | C:\Program Files\Microsoft MPI\Bin\ |
| NUMBER_OF_PROCESSORS | 6 |
| OS | Windows_NT |
| Path | C:\Program Files\Microsoft MPI\Bin\;C:\Program Files (x86)\Comm... |

# Step 1 (for Windows)

1. Goto developers.android.com/studio and download android studio
2. After downloading studio, launch android studio and continue to next steps.



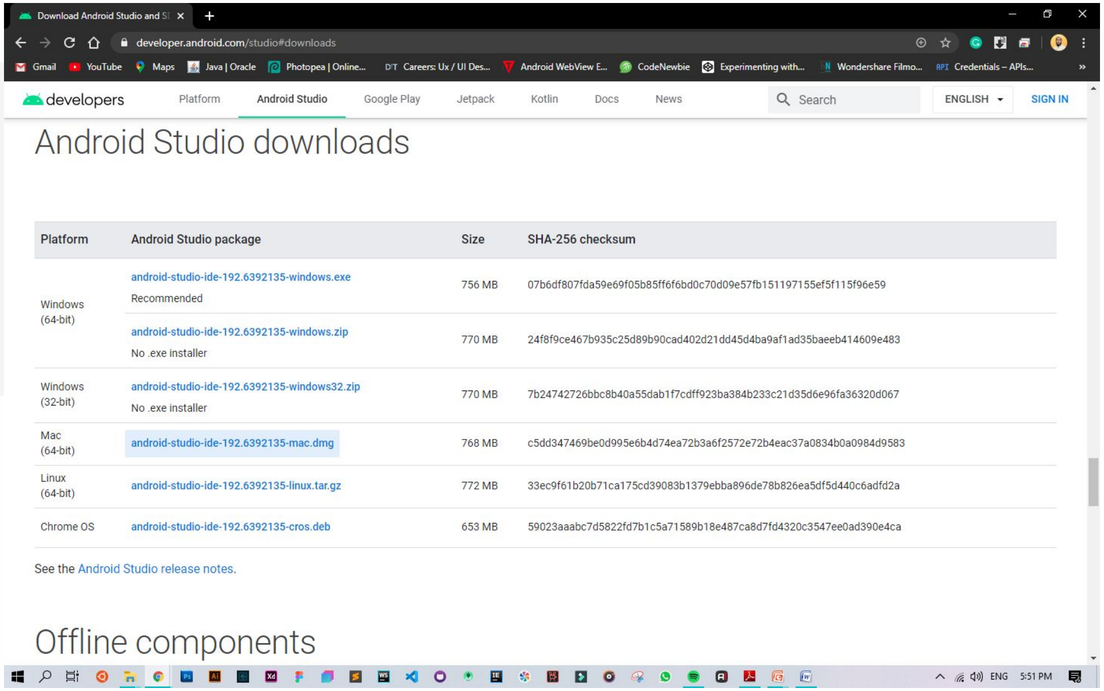
Join us for #Android11: The Beta Launch Show on June 3!

Android Studio provides the fastest tools for building apps on every type of Android device.

3.6.3 for Windows 64-bit (756 MB)

# Step 1 (for MacOS)

1. Goto developers.android.com and click on download options and select the android studio for macOS , the one with .dmg file extension.
2. Launch the installation file and continue to the next steps.



## Android Studio downloads

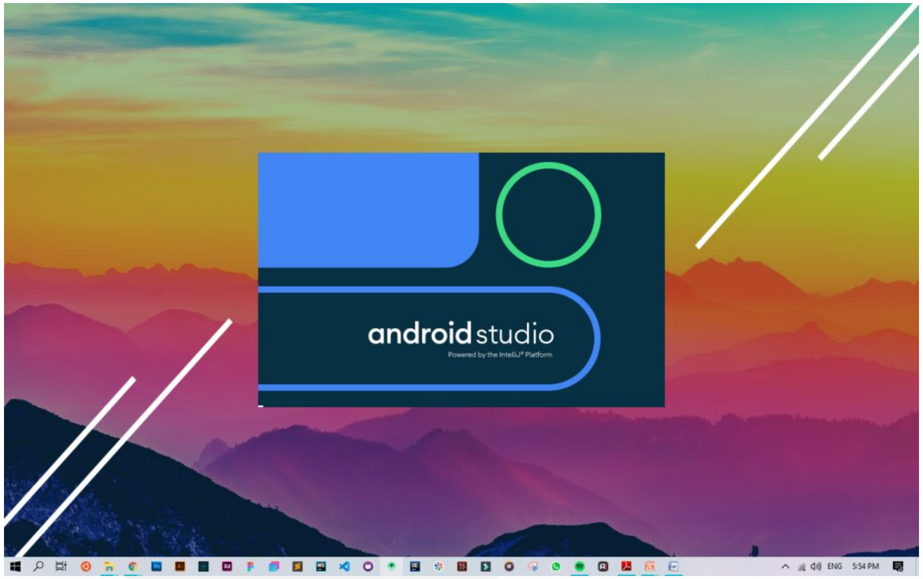
| Platform | Android Studio package | Size | SHA-256 checksum |
|---|---|---|---|
| Windows (64-bit) | android-studio-ide-192.6392135-windows.exe<br>Recommended | 756 MB | 07b6df807fda59e69f05b85ff6f6bd0c70d09e57fb151197155ef5f115f96e59 |
| | android-studio-ide-192.6392135-windows.zip<br>No .exe installer | 770 MB | 24f8f9ce467b935c25d89b90cad402d21dd45d4ba9af1ad35baeeb414609e483 |
| Windows (32-bit) | android-studio-ide-192.6392135-windows32.zip<br>No .exe installer | 770 MB | 7b24742726bbc8b40a55dab1f7cdff923ba384b233c21d35d6e96fa36320d067 |
| Mac (64-bit) | android-studio-ide-192.6392135-mac.dmg | 768 MB | c5dd347469be0d995e6b4d74ea72b3a6f2572e72b4eac37a0834b0a0984d9583 |
| Linux (64-bit) | android-studio-ide-192.6392135-linux.tar.gz | 772 MB | 33ec9f61b20b71ca175cd39083b1379ebba896de78b826ea5df5d440c6adfd2a |
| Chrome OS | android-studio-ide-192.6392135-cros.deb | 653 MB | 59023aaabc7d5822fd7b1c5a71589b18e487ca8d7fd4320c3547ee0ad390e4ca |

See the Android Studio release notes.

## Offline components

# Step 2

1. Launching android studio for the first time, it will run setup wizard
   a) Choose "don't import previous settings"
   b) In "welcome", choose "next"
   c) In "install type", choose "standard"
   d) Select UI theme.
2. In SDK setup make sure virtual device is selected and carry on to the installation steps.



# Step 3

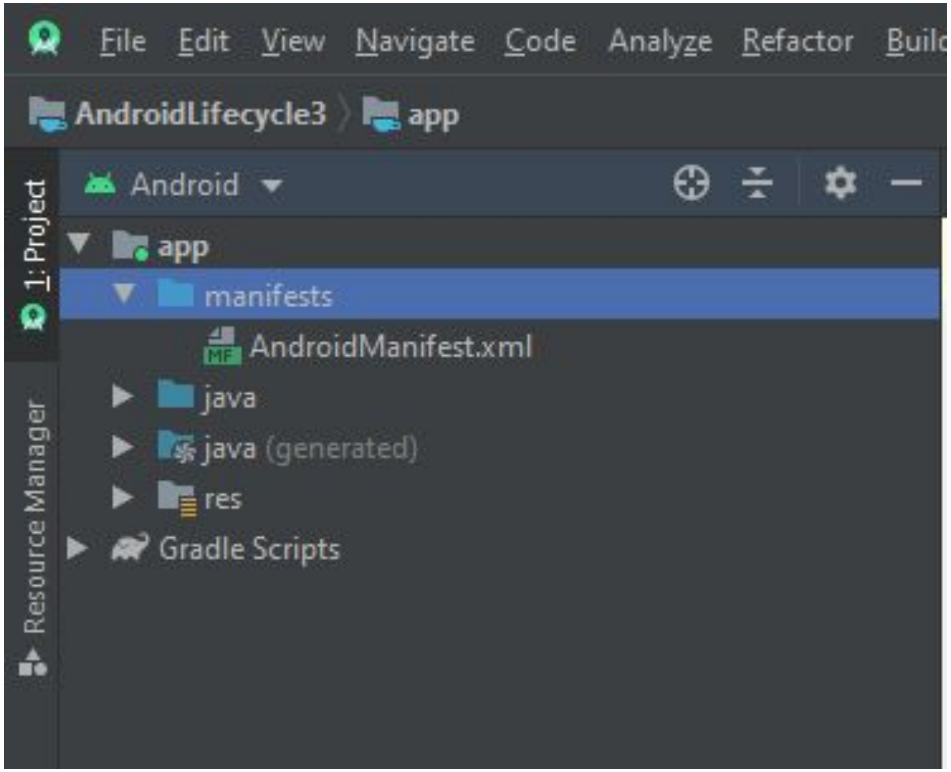
## Set up Emulator

1. In android studio select “Tools” -> Android -> AVD manager
2. Choose a device
3. Select and system image
4. Verify configuration, finish.

PU

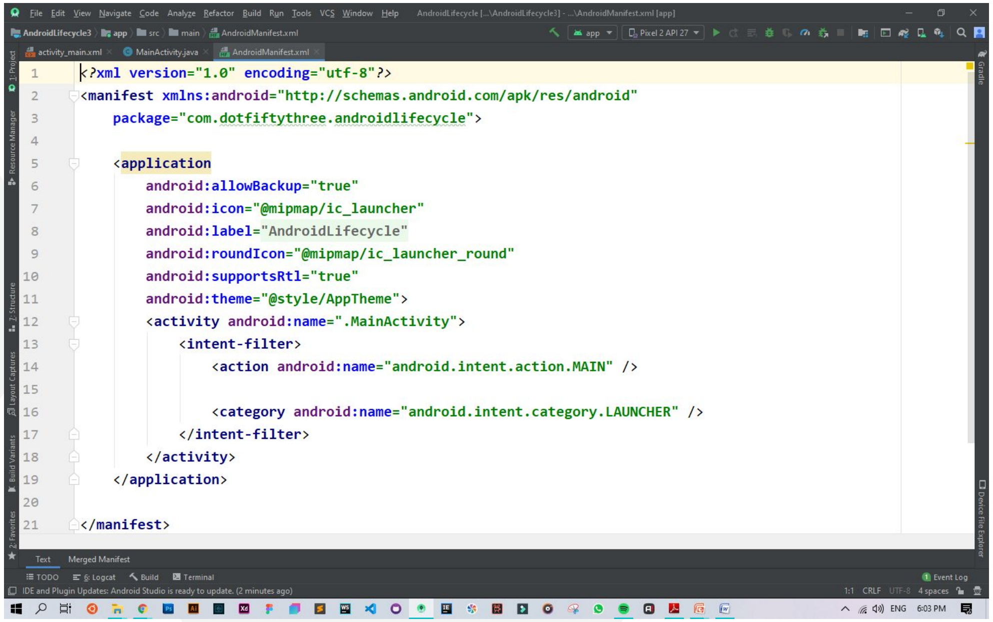
# Android Manifest File

Every project in Android includes a manifest file, which is AndroidManifest.xml, stored in the root directory of its project hierarchy. The manifest file is an important part of our app because it defines the structure and metadata of our application, its components, and its requirements.



# AndroidManifest.xml

This file includes nodes for each of the Activities, Services, Content Providers and Broadcast Receiver that make the application and using Intent Filters and Permissions, determines how they co-ordinate with each other and other applications.



```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.dotfiftythree.androidlifecycle">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="AndroidLifecycle"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

IDE and Plugin Updates: Android Studio is ready to update. (2 minutes ago)

# AndroidManifest.xml

A manifest file includes the nodes that define the application components, security settings, test classes and requirements that make up the application. Some of the manifest sub-node tags that are mainly used are:

- uses-sdk
- uses-configuration
- uses-features
- permission
- support-screens
- intent-filters
- application

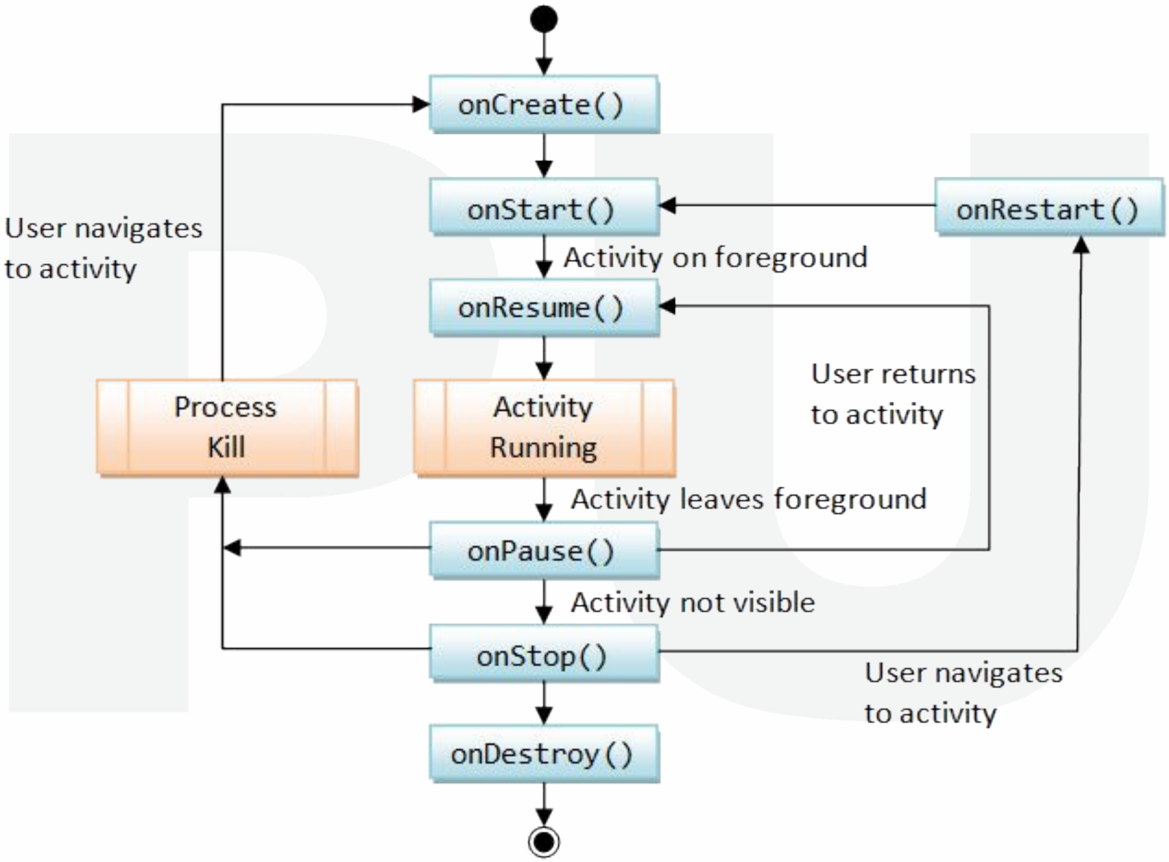
PU

# Android LifeCycle

As a user navigates through the app, Activity instances in your app transition through different stages in their life-cycle. The Activity class provides a number of callbacks that allow the activity to know that a state has changed: that the system is creating, stopping, or resuming an activity, or destroying the process in which the activity resides.

PU

# Android LifeCycle



# LifeCycle Methods

In general, activity lifecycle has seven callback methods:

1. onCreate()
2. onStart()
3. onResume()
4. onPause()
5. onStop()
6. onRestart()
7. onDestroy()

PU

# Android LifeCycle

**When you open the app it will go through below states:**

onCreate() –> onStart() –>  onResume()

**When you press the back button and exit the app**

onPaused() — > onStop() –> onDestory()

**When you press the home button**

onPaused() –> onStop()

**After pressing the home button, again when you open the app from a recent task list**

onRestart() –> onStart() –> onResume()

**After dismissing the dialog or back button from the dialog**

onResume()

PU

# Android LifeCycle

**If a phone is ringing and user is using the app**

onPause() –> onResume()

**After the call ends**

onResume()

**When your phone screen is off**

onPaused() –> onStop()

**When your phone screen is turned back on**

onRestart() –> onStart() –> onResume()

PU

# DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversity.ac.in