UNIT-1
INTRODUCTION TO ANDROID

Introduction to Android:

- Android is an Open source mobile Operating system which is a modified version of Linux Kernel.
- Android was developed by Andy Rubin and his Team members(Rich Miner, Nick sears, Chris white) in October 2003 at California (Palo Ato)
- Google acquired the startup company Android Inc. in 2005 to start the development of the Android Platform.
- In late 2007, a group of industry leaders came together around the Android Platform to form the Open Handset Alliance (http://www.openhandsetalliance.com).
- The Android SDK was first issued as an "early look" release in November 2007. September 2008 T-Mobile announced the availability of the T-Mobile G1, the first smartphone based on the Android Platform.
- A few days after that, Google announced the availability of Android SDK Release Candidate 1.0.
- In October 2008, Google made the source code of the Android Platform available under Apache's open source license.
- Android is a rich application Framework that allows you to develop innovative apps and games for your Mobile devices.
- **Java** is the Official Development for Android, but in Google I/O 2017 Google introduced another language **Kotlin** as an Official language.
- Anyone can be the App developer and the tools needed for the App development is freely available on the web.
- Android uses a special virtual machine to run your apps called **Dalvik Virtual Machine(DVM).**
- You can develop Android apps and make it available for users through Google Play Store.
- You can earn money from your apps through in App purchases and by placing Ads on your App.
- Every year Google releases a new version of Android.
- The latest Version of the Android is Oreo(Version 8.0, API level 26).

Advantages of Android:

- The ability for anyone to customize the Google Android platform
- It gives you better notification.
- It lets you choose your hardware.
- It has better app market(1,80,000 application)
- A more mature platform
- With the support of many applications, the user can change the screen display.
- With Google chrome you can open many window at once.
- Supports all Google services: Android operating system supports all of Google services ranging from Gmail to Google reader. all Google services can you have with one operating system, namely Android.

Disadvantages of Android:

- Android Market is less control of the manager, sometimes there are malware.
- Wasteful Batteries, This is because the OS is a lot of "process" in the background causing the battery quickly drains.
- Sometimes slow device company issued an official version of Android your own .
- Extremely inconsistence in design among apps.
- Very unstable and often hang or crash.

The Android 4.1 Jelly Bean SDK

The Android 4.1 Jelly Bean SDK was released with new features for developers in July 2012. It improves the beauty and simplicity of Android 4.0 and is a major platform release that adds a variety of new features for users and app developers. A few of the big features of this release include the
following:

• **Project Butter**—Makes the Jelly Bean UI faster and more responsive. Also CPU Touch Responsiveness is added, which increases CPU performance whenever the screen is touched. It uses the finger’s speed and direction to predict where it will be located after some milliseconds, hence making the navigation faster.

• **Faster speech recognition**—Speech recognition is now faster and doesn’t require any network to convert voice into text. That is, users can dictate to the device without an Internet connection.

• **Improved notification system**—Besides text, the notifications include pictures and lists too. Notifications can be expanded or collapsed through a variety of gestures, and users can block notifications if desired. The notifications also include action buttons that enable users to call directly from the notification menu rather replying to email.

• **Supports new languages**—Jelly Bean includes support for several languages including Arabic, Hebrew, Hindi, and Thai. It also supports bidirectional text.

• **Predictive keyboard**—On the basis of the current context, the next word of the message is automatically predicted.

• **Auto-arranging Home screen**—Icons and widgets automatically resize and realign as per the existing space.

• **Helpful for visually impaired users**—The Gesture Mode combined with voice helps visually impaired users to easily navigate the user interface.

• **Improved Camera app**—The Jelly Bean Camera app includes a new review mode of the captured photos. Users can swipe in from the right of the screen to quickly view the captured photos. Also, users can pinch to switch to a new film strip view, where they can swipe to delete photos.

• **Better communication in Jelly Bean**—Two devices can communicate with Near Field Communication (NFC); that is, two NFC-enabled Android devices can be tapped to share data. Also, Android devices can be paired to Bluetooth devices that support the Simple Secure Pairing standard by just tapping them together.

• **Improved Google Voice search**—Jelly Bean is equipped with a question and answer search method that helps in solving users' queries similar to Apple's popular Siri.

• **Face Unlock**—Unlocks the device when the user looks at it. It also prevents the screen from blacking out. Optionally "blink" can be used to confirm that a live person is unlocking the device instead of a photo.

• **Google Now**—Provides users "just the right information at just the right time." It displays cards to show desired information automatically. For example, the Places card displays nearby restaurants and shops while moving; the Transit card displays information on the next train or bus when the user is near a bus stop or railway station; the Sports card displays live scores or upcoming game events; the Weather card displays the weather conditions at a user's current location, and so on.

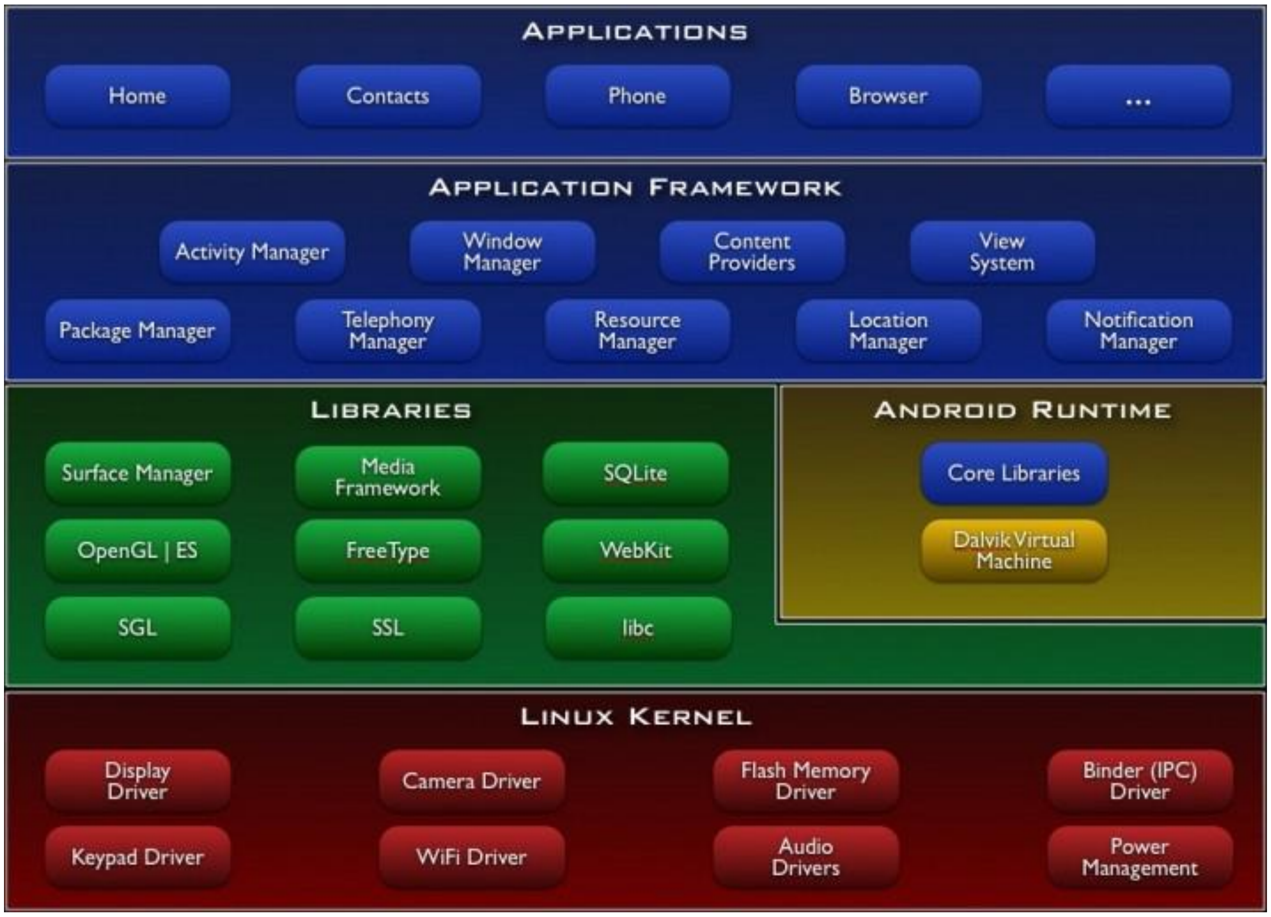
• **Google Play Widgets**—Provides quick and easy access to movies, games, magazines, and other media on the device. It also suggests new purchases on Google Play.

• **Faster Google Search**—Google Search can be opened quickly, from the lock screen and from the system bar by swiping up and also by tapping a hardware search key if it is available on the device.

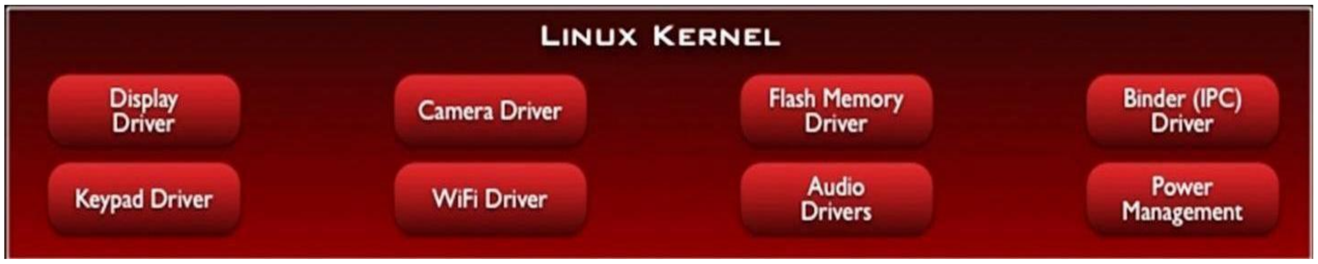
• **Supports antipiracy**—This feature supports developers in the sense that the applications are encrypted with a device-specific key making it difficult to copy and upload them to the Internet.

Understanding the Android Software Stack

The Android software stack consists of a Linux kernel and a collection of C/C++ libraries that are exposed through an application framework for application development. The Android software stack consists of four main layers, as shown below.



1. LINUX KERNEL LAYER

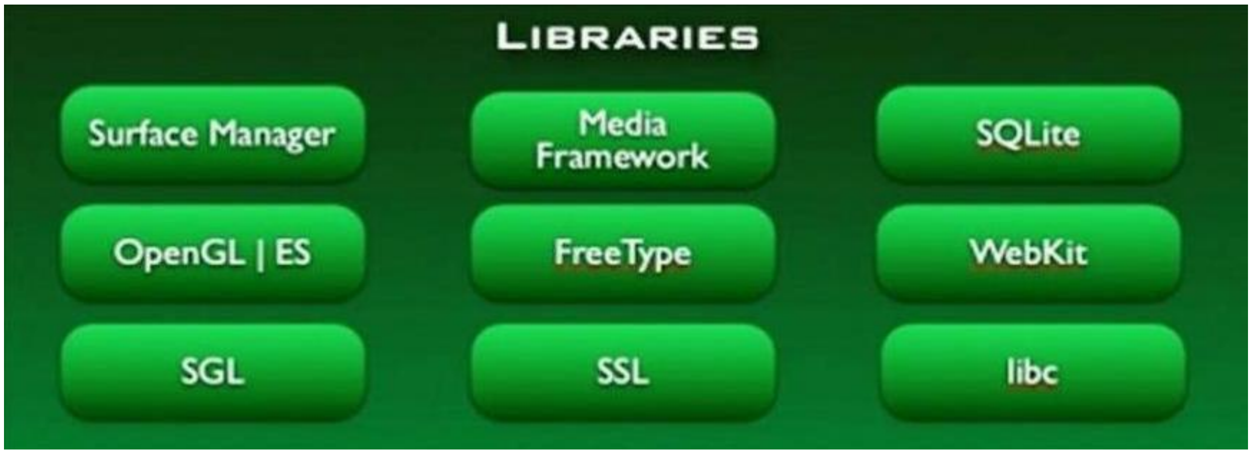


The architecture is based on the Linux2.6 kernel.

• This layer is core of android architecture. It provides service like power management, memory management,security etc.

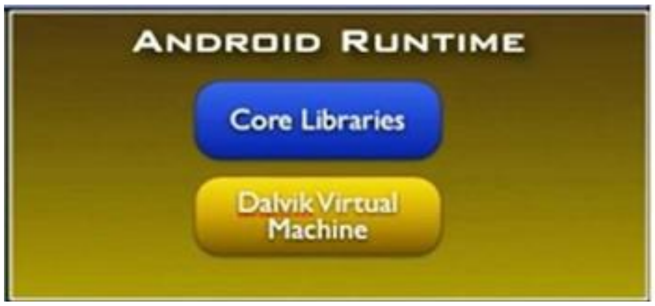
• It helps in software or hardware binding for better communication.

2. LIBRARIES AND ANDROID RUNTIME LAYER



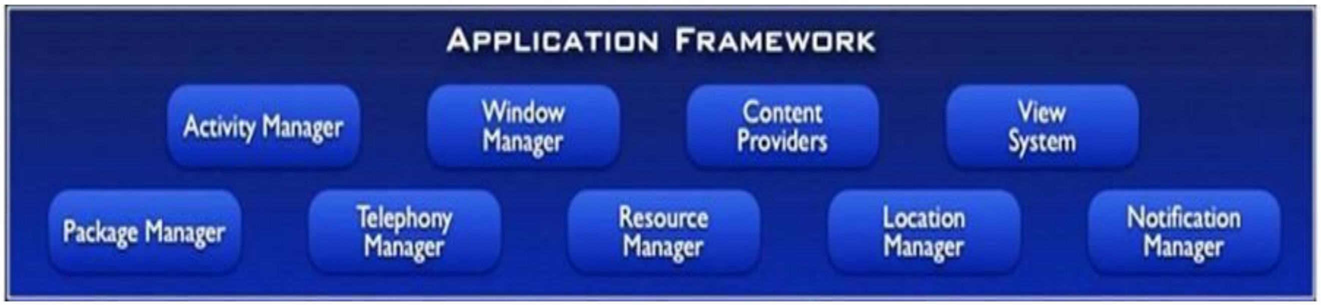
Android has its own libraries, which is written in C/C++. These libraries cannot be accessed directly. With the help of application framework, we can access these libraries. There are many libraries like :

- **WebKit library**—Responsible for browser support.
- **FreeType library**—Responsible for font support.
- **SQLite library**—Provides database support.
- **Media libraries**—Responsible for recording and playback of audio and video formats.
- **Surface Manager library**—Provides graphics libraries that include SGL and OpenGL for 2D and 3D graphics support.



- The Android Runtime was designed specifically for Android to meet the needs of running in an embedded environment where you have limited battery, limited memory, limited CPU.
- Dalvik is the process virtual machine in Google's android operating system. It is the software that runs the apps on android devices. Dalvik is thus an integral part of android ,which is typically used on mobile devices such as mobile phones and tablet computers.
- Programs are commonly written in java and compiled to byte code.
- Core Libraries are written in the Java programming language.
- The core library contains all of the collection classes, utilities, IO, all the utilities and tools that you've come to expected to use.

3. APPLICATION FRAMEWORK LAYER



- This is all written in a Java programming language and the application framework is the toolkit that all applications use.
- These applications include the ones that come with a phone like the home applications, or the phone application.

- It includes applications written by Google, and it includes apps that will be written by you. So, all apps use the same framework and the sameAPIs.
- **Activity manager:-**It manages the lifecycle of applications. It enable proper management of all the activities. All the activities are controlled by activity manager.
- **Resource manager:-**It provides access to non-code resources such as graphics etc.
- **Notification manager:-**It enables all applications to display custom alerts in status bar.
- **Location manager:-** It fires alerts when user enters or leaves a specified geographical location.
- **Package manager:-**It is use to retrieve the data about installed packages on device.
- **Window manager:-**It is use to create views and layouts.
- **Telephony manager:-**It is use to handle settings of network connection and all information about services on device.

4. APPLICATIONS LAYER

APPLICATIONS
Home
Contacts
Phone
Browser
...

The final layer on top is Applications.

- It includes the home application the contacts application , the browser, and apps.
- It is the most upper layer in android architecture.
- All the applications like camera, Google maps, browser,sms,calendars,contacts are native applications. These applications works with end user with the help of application framework to operate.

Installation of Android SDK:

Android SDK is installed in two phases.

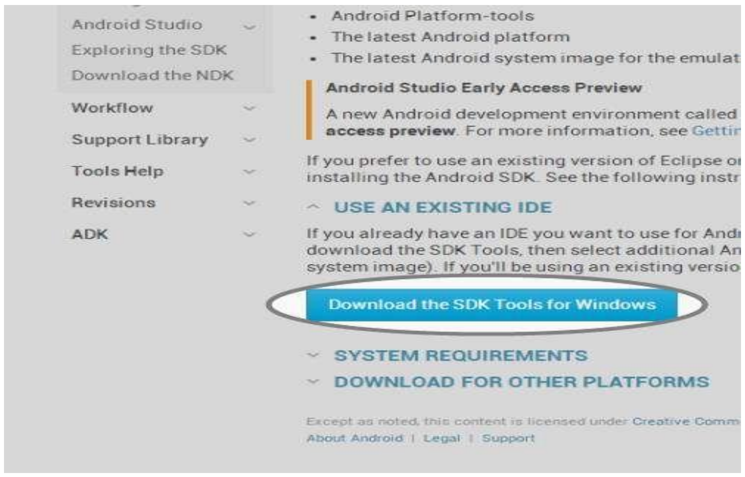
- The first phase is the installation of the SDK, which installs the Android SDK Tools.
- The second phase is installation of the Android platforms and other components.

To get and setup the Android SDK provided by Google, You need to install the following four application:

The Java Development Kit (JDK) can be downloaded from http://oracle.com/technetwork/java/javase/downloads/index.html.
The Eclipse IDE can be downloaded from http://www.eclipse.org/downloads/.
The Android Platform SDK Starter Package can be downloaded fromhttp://developer.android.com/sdk/index.html.
The Android Development Tools (ADT) Plug-in can be downloaded from http://developer.android.com/sdk/eclipse-adt.html.

Step1: To Obtain the Android SDK

- Visit the website http://developer.android.com/sdk/index.html



- Android Platform-tools
- The latest Android platform
- The latest Android system image for the emulat

Android Studio Early Access Preview

A new Android development environment called
access preview. For more information, see Gettir

If you prefer to use an existing version of Eclipse or
installing the Android SDK. See the following instr

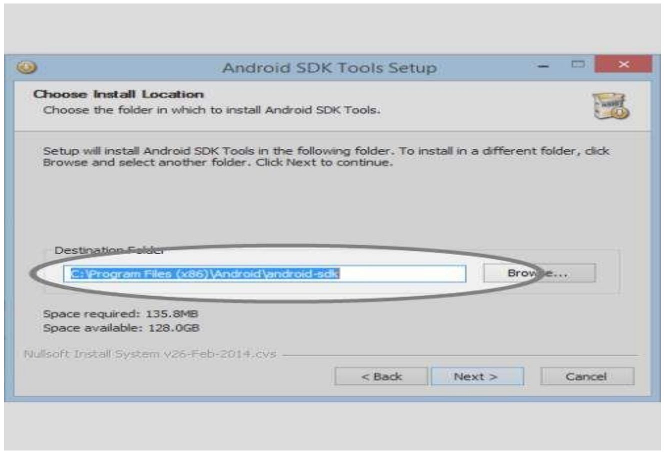
USE AN EXISTING IDE

If you already have an IDE you want to use for Andr
download the SDK Tools, then select additional An
system image). If you'll be using an existing versio

Except as noted, this content is licensed under Creative Comm

- Scroll to the bottom of the webpage and select the 'Download the SDK Tools' button.
- This will open a file to be saved somewhere.
- Then select the Next button to continue.
- The next dialog box asks you to specify the Start Menu folder where you want the program's shortcuts to appear.
- A default folder name appears called Android SDK Tools.
- If you do not want to make a Start Menu folder, select the Do not create shortcuts check box.

Step2: Install the Android SDK



Choose Install Location
Choose the folder in which to install Android SDK Tools.

Setup will install Android SDK Tools in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder: C:\Program Files (x86)\Android\android-sdk

Space required: 135.8MB
Space available: 128.0GB

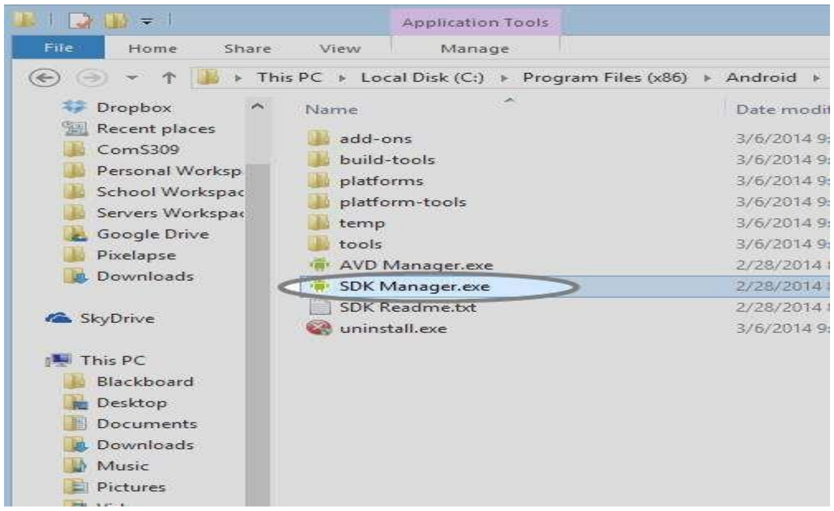
Nullsoft Install System v26-Feb-2014.cvs

- Open the file we just downloaded.
- This will open an executable which will ask you about the installation process.
- When you get to the install location screen, make sure you choose a location you remember

Step3:Open Android SDK Manager

We'll select the most recent version along with the extra SDK tools so you can build applications for any Android version.

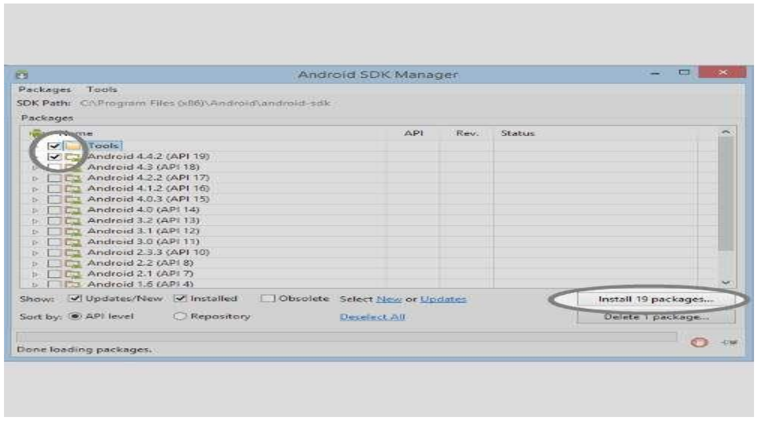
- Open the folder that we installed the SDK into.



Name	Date modi
add-ons	3/6/2014 9:
build-tools	3/6/2014 9:
platforms	3/6/2014 9:
platform-tools	3/6/2014 9:
temp	3/6/2014 9:
tools	3/6/2014 9:
AVD Manager.exe	2/28/2014
SDK Manager.exe	2/28/2014
SDK Readme.txt	2/28/2014
uninstall.exe	3/6/2014 9:

- You'll see a executable called 'SDK Manager'. Open it.
- You'll see a window where you can select different versions of Android to develop for.

Step 4: Install Android Version and Extras for SDK



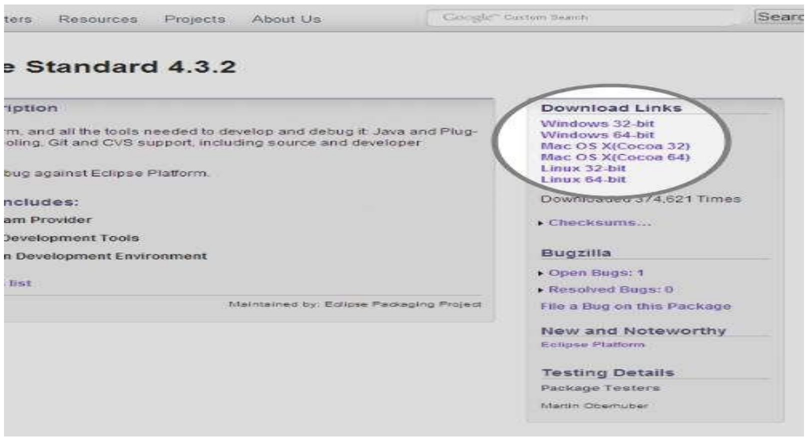
- Select the 'Tools' and 'Android 4.4.2 (API 19)' check boxes.
- If you would like to have extra Android tools you can choose them from the 'Extras' selection.
- You will then be prompted to accept the Android conditions and then the SDK will install. Then installation may take a while depending on your internet connection.

- After all the files have been downloaded and installed on the computer, select the Next button.
- The next dialog box tells you that the Android SDK Tools Setup Wizard is complete and the Android SDK Tools have successfully installed on the computer.
- Select Finish to exit the wizard.

Step 5: Obtain Eclipse IDE

Eclipse is the tool we'll be using to develop in. It is the most popular Android development environment and has officially supported tools from Google.

- Download Eclipse from the website below.
- http://www.eclipse.org/downloads/packages/eclipse-...
- Find the link for your operating system and 32/64 bit version.
- Save the compressed download file.



ters Resources Projects About Us

e Standard 4.3.2

ription

rm, and all the tools needed to develop and debug it: Java and Plug-
oling, Git and CVS support, including source and developer

bug against Eclipse Platform.

ncludes:
am Provider
Development Tools
n Development Environment

list

Maintained by: Eclipse Packaging Project

Download Links
Windows 32-bit
Windows 64-bit
Mac OS X(Cocoa 32)
Mac OS X(Cocoa 64)
Linux 32-bit
Linux 64-bit

Downloaded 374,621 Times

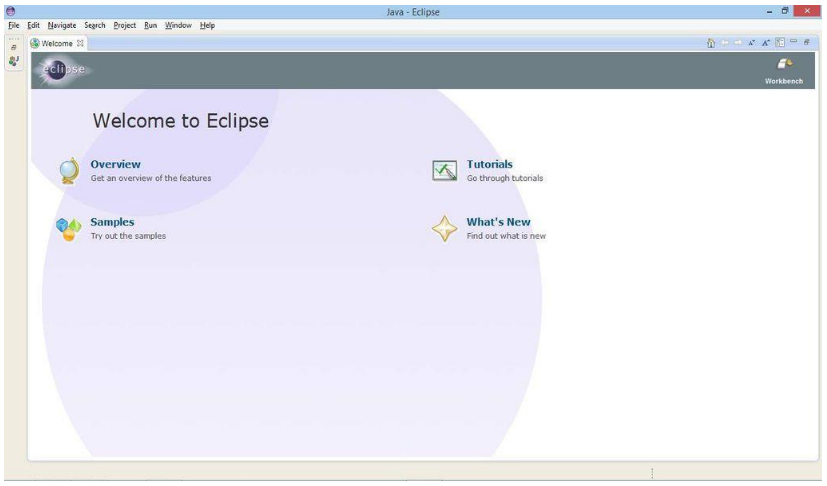
Checksums...

Bugzilla
Open Bugs: 1
Resolved Bugs: 0
File a Bug on this Package

New and Noteworthy
Eclipse Platform

Testing Details
Package Testers
Martin Oberhuber

Step 6: Run Eclipse for First Time



Eclipse does not require installation. It's a folder with all the necessary files and settings. You can run it directly from the Eclipse folder. It's recommended you put it in a safe place with other applications.

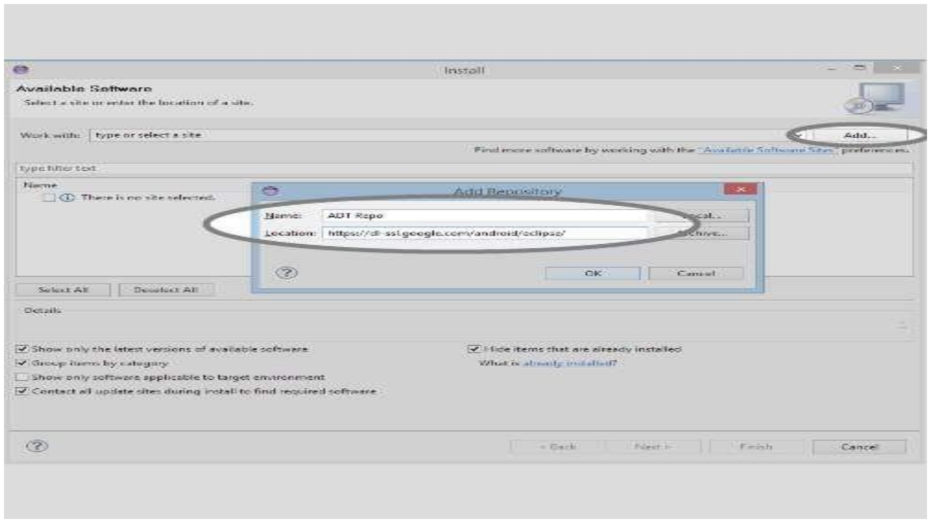
- Extract the downloaded Eclipse file into a safe place where you can keep the program.
- Open the extracted folder and open the 'eclipse' executable.

You should see a screen similar to the one attached if all went well.

Step 7: Add ADT Plugin Repository

The ADT (Android Development Tool) Plugin was made specifically for Eclipse to increase productivity and integration with your Android work environment. To use it, we first add the Eclipse plugin repository so it knows where to find it along with updates.

- In the Eclipse application menu, go to 'Help' and then 'Install New Software'.
- Click on the 'Add...' button and you'll see a window appear.



Work with: type or select a site

Find more software by working with the 'Available Software Sites' preferences.

There is no site selected.

Name: ADT Repo

Location: https://dl-ssl.google.com/android/eclipse/

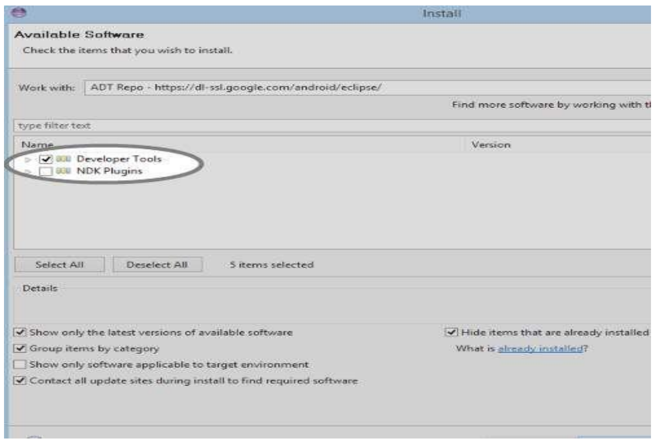
☑ Show only the latest versions of available software
☑ Group items by category
☐ Show only software applicable to target environment
☑ Contact all update sites during install to find required software
☑ Hide items that are already installed

- Give the repository a name like 'ADT Repo'.
- Give it the location http://dl-ssl.google.com/android/eclipse/.
- Click 'OK' button.

Step 8: Install ADT Plugin

Now that we have the plugin repo setup we need to install the plugin from it.

- On the 'Install Software' screen, select the repo you just created from the 'Work with' selector.
- Select the 'Developer Tools' option from the listed below options.



Work with: ADT Repo - https://dl-ssl.google.com/android/eclipse/

Find more software by working with t

☑ Developer Tools
☐ NDK Plugins

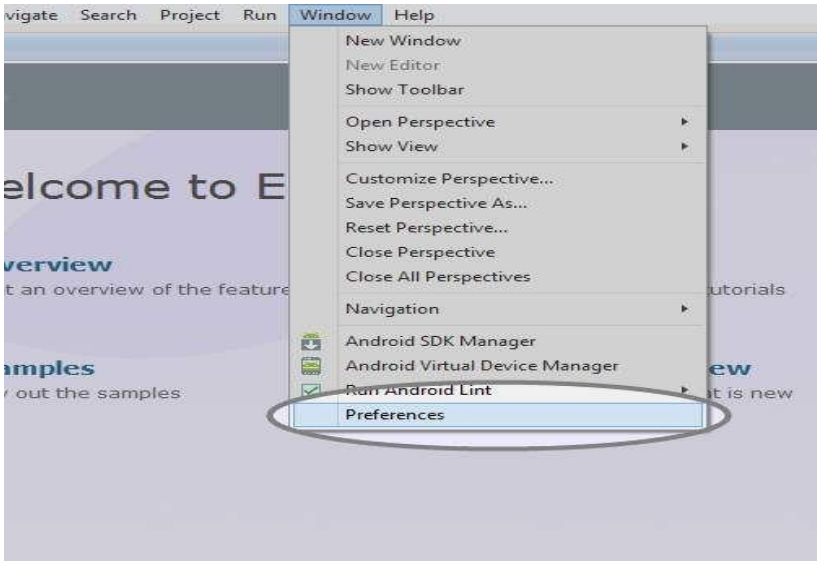
5 items selected

☑ Show only the latest versions of available software
☑ Group items by category
☐ Show only software applicable to target environment
☑ Contact all update sites during install to find required software
☑ Hide items that are already installed

- Click 'Next' and accept the agreements.
- Click 'Finish' and let it install. It might take a while depending on your internet speed.

Step 9: Access ADT Plugin Preferences

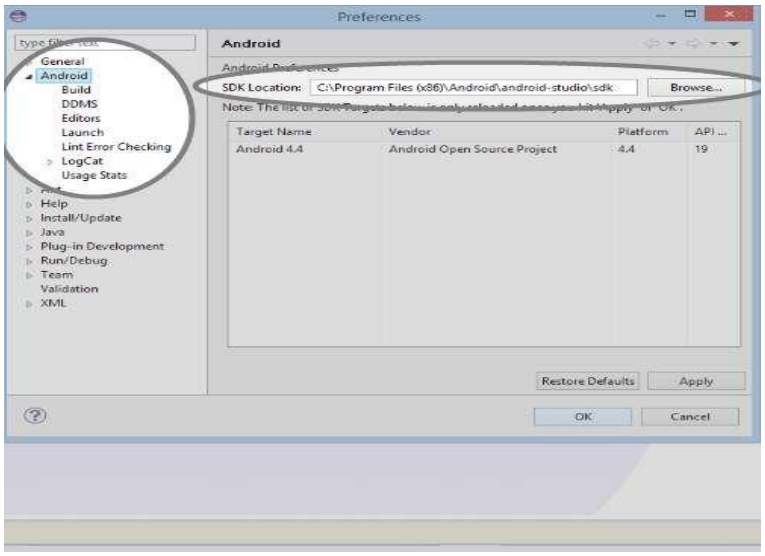
You'll most likely have to restart Eclipse after you install the ADT Plugin so do that before you continue.



- Mac/Linux: After Eclipse has restarted, click 'Eclipse' in the application menu.
- Windows: After Eclipse has restarted, click 'Window' in the application menu.
- Then select 'Preferences'.
- In the Preferences window, select the Android tab on the left side and it's corresponding drop down menu.

This is the ADT Preferences screen. It will allow you to change setting, remove the SDK and make editor preferences for your development.

Step 10: Setup ADT Plugin



SDK Location: C:\Program Files (x86)\Android\android-studio\sdk

Note: The list of SDK Targets below is only reloaded once you hit 'Apply' or 'OK'.

Target Name	Vendor	Platform	API ...
Android 4.4	Android Open Source Project	4.4	19

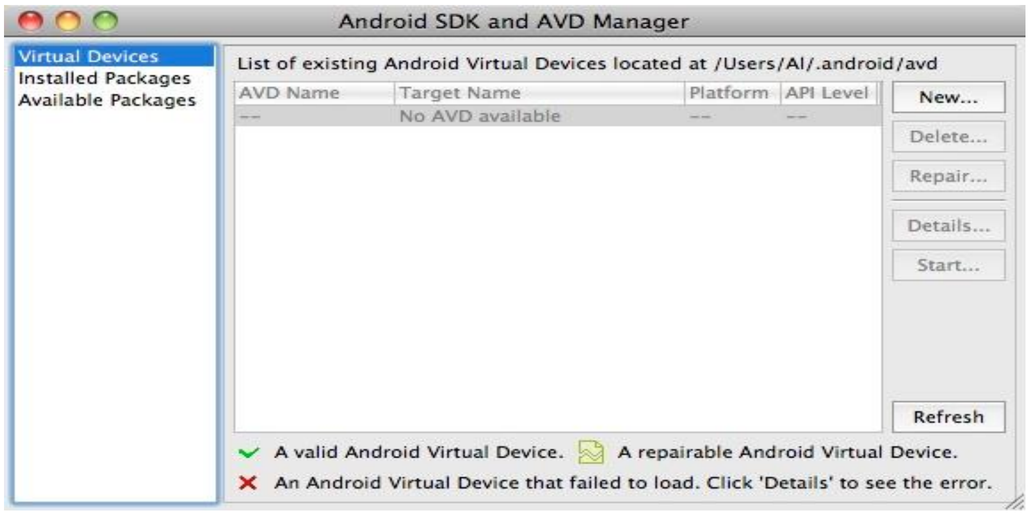
- Click 'Browse' on the right side of the screen.
- Search for the folder in which you installed the Android SDK into and select it.
- Hit the 'Apply' button on the Preferences screen.
- You should see the Android version you installed early to show up if all went well.
- If not, try reselecting the folder. (Make sure the folder contains the folders 'build-tools','platform','extras' and etc.
- Hit the 'OK' button and restart Eclipse.

Creating Android Virtual Devices:

- An Android Virtual Device (AVD) represents a device configuration.
- To test our applications we need an Android Virtual Device (AVD). It is an Android smartphone emulator, where Eclipse can install and launch our applications.
- To create AVDs in Eclipse

1) In Eclipse, click the Window menu, then click Android SDK and AVD Manager.

2) On the Android SDK and AVD Manager dialog that pops up, you'll see a list of any Android AVDs you have already created:



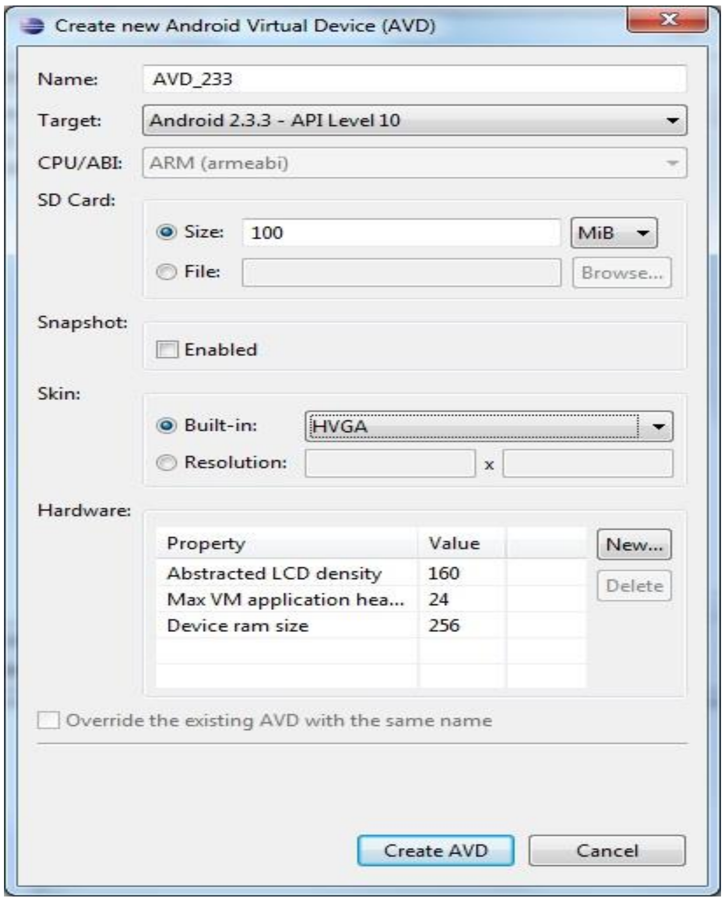
List of existing Android Virtual Devices located at /Users/Al/.android/avd

AVD Name	Target Name	Platform	API Level
--	No AVD available	--	--

✓ A valid Android Virtual Device. A repairable Android Virtual Device.
✗ An Android Virtual Device that failed to load. Click 'Details' to see the error.

3) Just click the "New..." button here to start creating a new AVD.

4) Then “Create new Android Virtual device” dialog box appears with some fields**.**



Create new Android Virtual Device (AVD)

Name: AVD_233

Target: Android 2.3.3 - API Level 10

CPU/ABI: ARM (armeabi)

SD Card: [✓ Size:] 100 MiB; File:

Snapshot: Enabled

Skin: [✓ Built-in:] HVGA; Resolution: x

Hardware:

Property	Value
Abstracted LCD density	160
Max VM application hea...	24
Device ram size	256

Override the existing AVD with the same name

The fields are as follows:

Name: Used to specify the name of the AVD.

Target: Used to specify the target API level.

CPU/ABI: Determines the processor that we want to emulate on our device.

SD Card: Used for extending the storage capacity of the device.

Snapshot: Used to start the Android quickly.

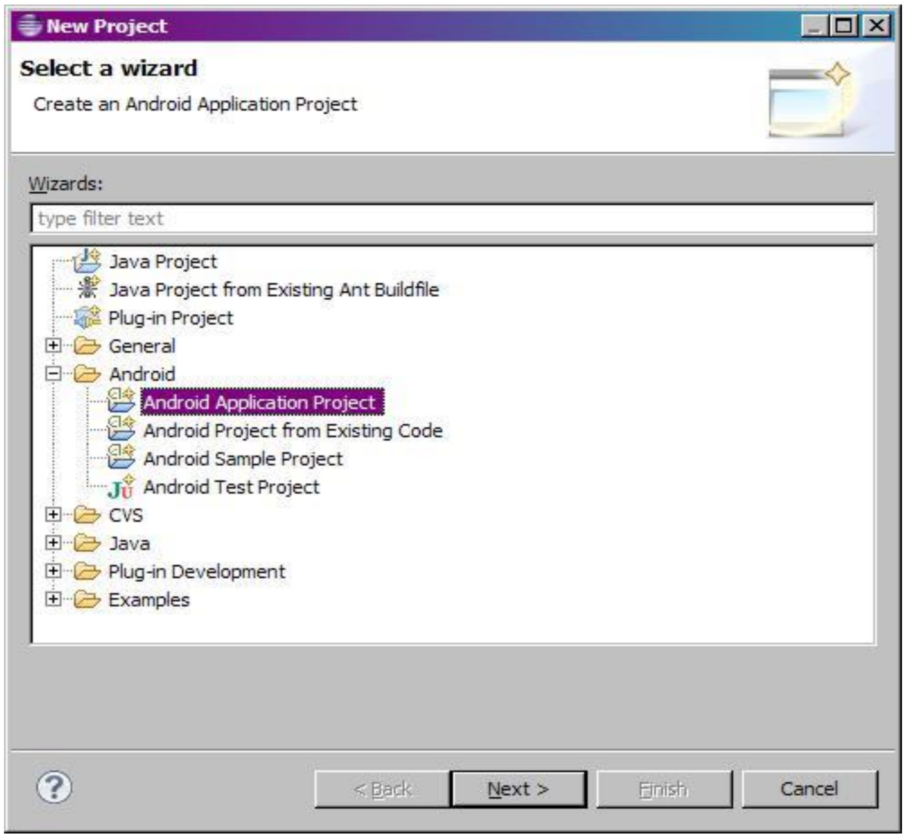
Skin: Used for setting the screen size.

Hardware: Used to set properties representing various optional hardware that may be present in the target device.

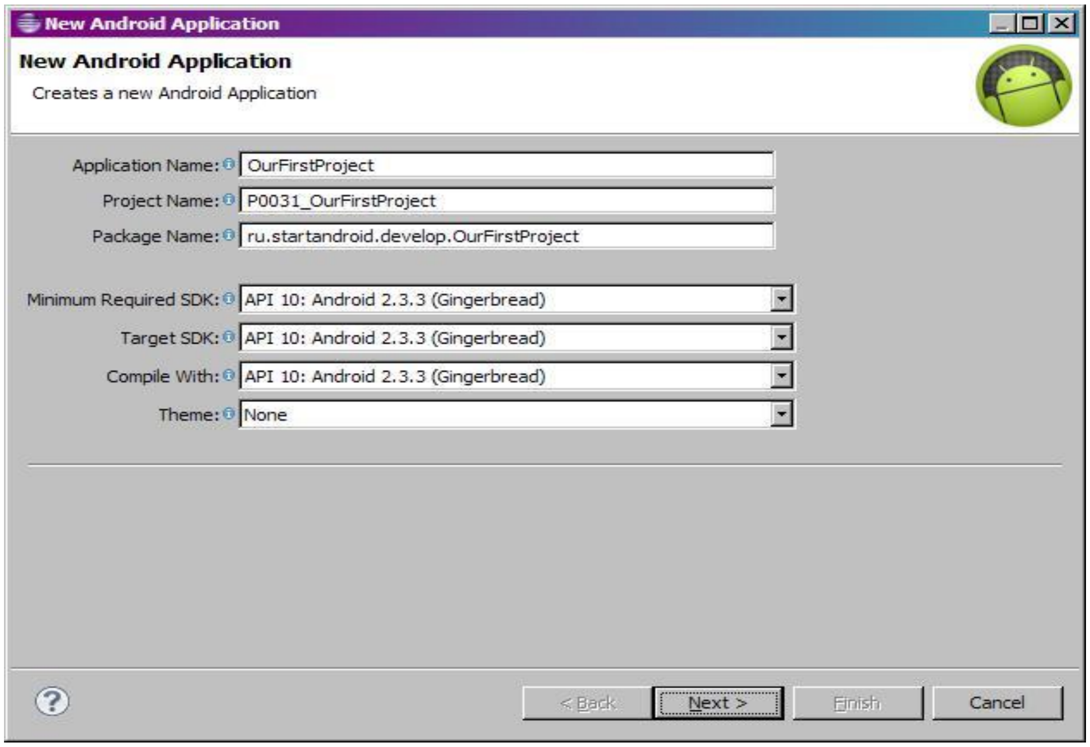
- After filling out all the parameters, just click the "Create AVD" button, and your new AVD will be created.
- You can now use this AD in your Eclipse/Android development projects.

Creating Android Project in Eclipse.

- In Eclipse go to File > New > Project
- Choose the project type - Android > Android Application Project, click Next

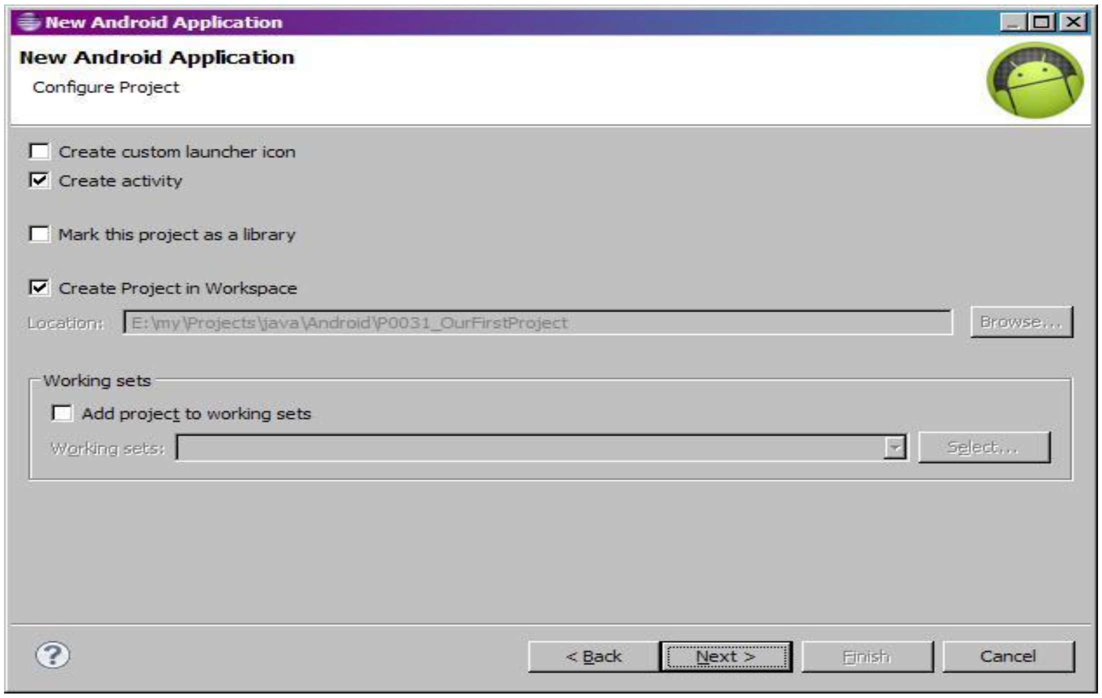


The window for creating a project appears.



Application Name: OurFirstProject
Project Name: P0031_OurFirstProject
Package Name: ru.startandroid.develop.OurFirstProject
Minimum Required SDK: API 10: Android 2.3.3 (Gingerbread)
Target SDK: API 10: Android 2.3.3 (Gingerbread)
Compile With: API 10: Android 2.3.3 (Gingerbread)
Theme: None

- Click Next



New Android Application

Configure Project

- [ ] Create custom launcher icon
- [x] Create activity
- [ ] Mark this project as a library
- [x] Create Project in Workspace

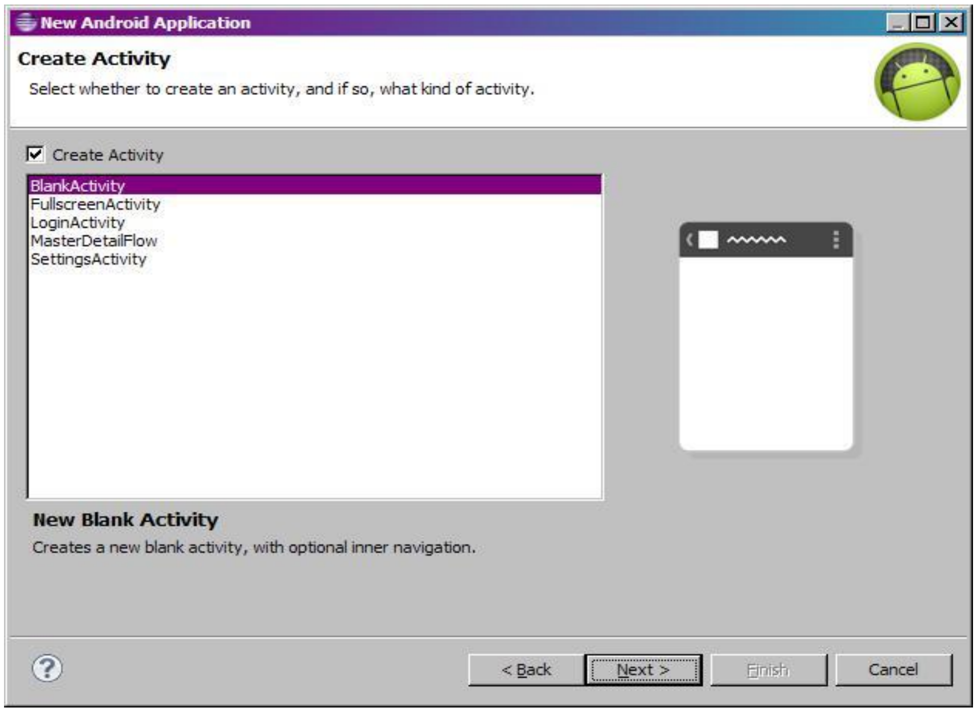
Location: E:\my\Projects\java\Android\P0031_OurFirstProject Browse...

Working sets

- [ ] Add project to working sets

Working sets: Select...

- Create Activity - immediately after the project creation the Activity will be created, not to create it manually.
- Create Project in Workspace. Project will be created and saved in the default workspace.
- Click Next



Create Activity

Select whether to create an activity, and if so, what kind of activity.

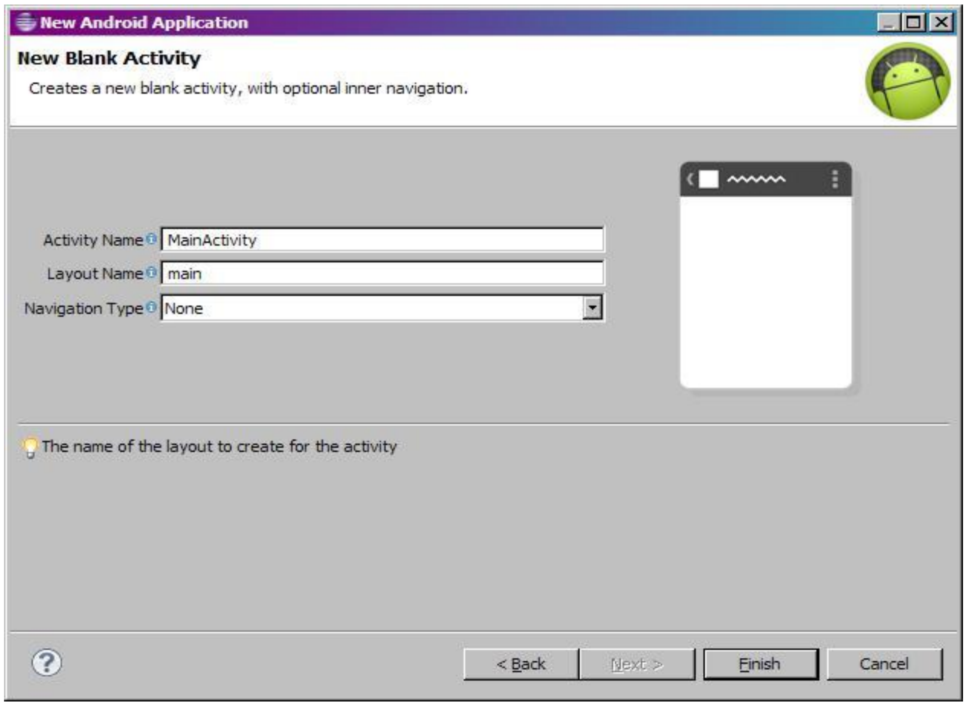
- [x] Create Activity

BlankActivity
FullscreenActivity
LoginActivity
MasterDetailFlow
SettingsActivity

New Blank Activity

Creates a new blank activity, with optional inner navigation.

- Choose BlankActivity
- Click Next


New Android Application

New Blank Activity

Creates a new blank activity, with optional inner navigation.

Activity Name: MainActivity

Layout Name: main

Navigation Type: None

The name of the layout to create for the activity

- Click Finish

Creating First Android Project using the Text view control:

In Android, TextView displays text to the user and optionally allows them to edit it programmatically. TextView is a complete text editor, however basic class is configured to not allow editing but we can edit it.

In android, we can create a TextView control in two ways either in XML layout file or create it in Activity file programmatically.

TextView code in XML:

```
<TextView android:id="@+id/simpleTextView"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:text="AbhiAndroid" />
```

TextView code in JAVA:

```
TextView textView = (TextView) findViewById(R.id.textView);

textView.setText("AbhiAndroid"); //set text for text view
```

Attributes of TextView:

1. id: id is an attribute used to uniquely identify a text view.

```
<TextView
android:id="@+id/simpleTextView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"/>
```

2. gravity: The gravity attribute is an optional attribute which is used to control the alignment of the text like left, right, center, top, bottom, center_vertical, center_horizontal etc.

```
<TextView
android:id="@+id/simpleTextView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="AbhiAndroid"
android:textSize="20sp"
android:gravity="center_horizontal"/> <!--center horizontal gravity-->
```

3. text: text attribute is used to set the text in a text view. We can set the text in xml as well as in the java class.

```
<TextView
android:id="@+id/simpleTextView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:textSize="25sp"
android:text="AbhiAndroid"/><!--Display Text as AbhiAndroid-->
```

In Java class:

Below is the example code in which we set the text in a textview programmatically means in java class.

```
TextView textView = (TextView)findViewById(R.id.textView);
textView.setText("AbhiAndroid"); //set text for text view
```

4. textColor: textColor attribute is used to set the text color of a text view. Color value is in the form of "#argb", "#rgb", "#rrggbb", or "#aarrggbb".

```
<TextView
```

```
android:id="@+id/simpleTextView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="AbhiAndroid"
android:layout_centerInParent="true"
android:textSize="25sp"
android:textColor="#f00"/><!--red color for text view-->
```

In Java class:

Below is the example code in which we set the text color of a text view programmatically means in java class.

```
TextView textView = (TextView)findViewById(R.id.textView);
textView.setTextColor(Color.RED); //set red color for text view
```

5. textSize: textSize attribute is used to set the size of text of a text view. We can set the text size in sp(scale independent pixel) or dp(density pixel).

Below is the example code in which we set the 20sp size for the text of a text view.

```
<TextView
    android:id="@+id/simpleTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="AbhiAndroid"
    android:layout_centerInParent="true"
    android:textSize="40sp" /><!--Set size-->
```

In Java class:

Below is the example code in which we set the text size of a text view programmatically means in java class.

```
TextView textView = (TextView)findViewById(R.id.textView);
textView.setTextSize(20); //set 20sp size of text
```

6. textStyle: textStyle attribute is used to set the text style of a text view. The possible text styles are bold, italic and normal. If we need to use two or more styles for a text view then "|" operator is used for that.

Below is the example code with explanation included in which we set the bold and italic text styles for text.

```
<TextView
    android:id="@+id/simpleTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="AbhiAndroid"
    android:layout_centerInParent="true"
    android:textSize="40sp"
    android:textStyle="bold|italic"/><!--bold and italic text style of
text-->
```

7. background: background attribute is used to set the background of a text view. We can set a color or a drawable in the background of a text view.

8. padding: padding attribute is used to set the padding from left, right, top or bottom. In above example code of background we also set the 10dp padding from all the side's of text view.

Below is the example code with explanation included in which we set the black color for the background, white color for the displayed text and set 10dp padding from all the side's for text view.

```
<TextView
android:id="@+id/simpleTextView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="AbhiAndroid"
android:layout_centerInParent="true"
android:textSize="40sp"
android:padding="10dp"
android:textColor="#fff"
android:background="#000"/> <!--red color for background of text view-->
```

In Java class:
Below is the example code in which we set the background color of a text view programmatically means in java class.

```
TextView textView = (TextView)findViewById(R.id.textView);
```

```
textView.setBackgroundColor(Color.BLACK);//set background color
```

Example of TextView:

Step 1: Create a new project and name it textViewExample.

```
Select File -> New -> New Project. Fill the forms and click "Finish"
button.
```

Step 2: Open res -> layout -> xml (or) activity_main.xml and add following code. Here we will create a buttonand a textview in Linear Layout.

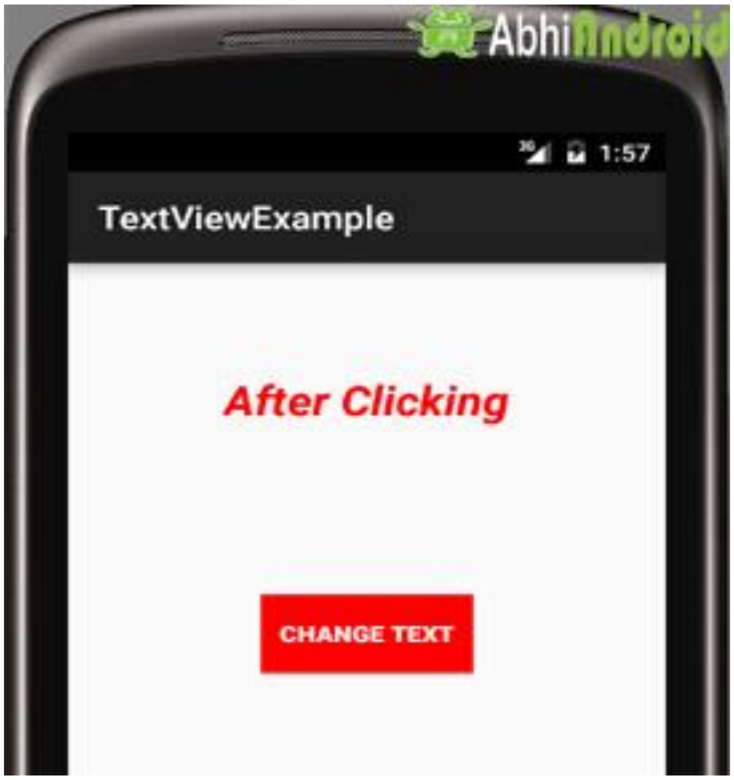
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/simpleTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:text="Before Clicking"
        android:textColor="#f00"
        android:textSize="25sp"
        android:textStyle="bold|italic"
        android:layout_marginTop="50dp"/>

    <Button
        android:id="@+id/btnChangeText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:background="#f00"
        android:padding="10dp"
        android:text="Change Text"
        android:textColor="#fff"
        android:textStyle="bold" />
</LinearLayout>
```

Step 3: Open app -> java -> package and open MainActivity.java and add the following code. Here we will change the text of TextView after the user click on Button.

```
package example.abhiandriod.textviewexample;


import android.graphics.Color;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;


public class MainActivity extends AppCompatActivity {


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main); //set the layout
        final TextView simpleTextView = (TextView)
findViewById(R.id.simpleTextView); //get the id for TextView
        Button changeText = (Button) findViewById(R.id.btnChangeText);
//get the id for button
        changeText.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                simpleTextView.setText("After Clicking"); //set the text
after clicking button
            }
        });
    }



}
```

Output:

Now run the app in Emulator and click on the button. You will see text will change "After Clicking".



Using the Android Emulator

The Android emulator is used for testing and debugging applications before they are loaded onto a real handset.

Limitations of the Android Emulator

The Android emulator is useful to testing and Debugging Android applications for compatibility with devices of different configurations which is present in AVD. But still, this Emulator is a piece of software and not an actual device and has several limitations:

1. Emulators no doubt help in knowing how an application may operate within a given environment, but they still don't provide the actual environment to an application.
2. Emulators just simulate certain handset behavior. Features such as GPS, sensors, battery, power settings, and network connectivity can be easily simulated on a computer.
3. SMS messages are also simulated and do not use a real network.
4. Phone calls cannot be placed or received but are simulated.
5. No support for device-attached headphones is available.
6. Peripherals such as camera/video capture are not fully functional.
7. No USB or Bluetooth support is available.

The emulator provides some facilities too.

- You can use the mouse and keyboard to interact with the emulator when it is running.
- You can use your computer keyboard to input text into UI controls and to execute specific emulator commands. Some of the most commonly used commands are
    - Back [ESC button]
    - Call [F3]
    - End [F4]
    - Volume Up [KEYPAD_PLUS, Ctrl-5]
    - Volume down [KEYPAD_MINUS, Ctrl-F6]
    - Switching orientations [KEYPAD_7, Ctrl-F11/KEYPAD_9, CtrlF12]

The Android Debug Bridge:

Android Debug Bridge (adb) is a versatile command-line tool that lets you communicate with a device. The adb command facilitates a variety of device actions, such as installing and debugging apps, and it provides access to a Unix shell that you can use to run a variety of commands on a device. It is a client-server program that includes three components:

- **A client,** which sends commands. The client runs on your development machine. You can invoke a client from a command-line terminal by issuing an adb command.

- **A daemon (adbd),** which runs commands on a device. The daemon runs as a background process on each device.

- **A server,** which manages communication between the client and the daemon.The server runs as a background process on your development machine.

When you install the Android SDK, the Android Debug Bridge is also automatically installed along with it. When the Android Debug Bridge is active, you can issue adb commands to interact with one or more emulator instances.To access ADB through Windows, open the command prompt and navigate to the folder where adb.exe is located by using the cd command. By default, adb.exe is installed in C:\Program Files (x86)\Android\android-sdk\platform-tools. When you are in the folder where adb is found, you can issue the following commands to interact with the device or emulator:

- **adb devices**—Displays the list of devices attached to the computer.
- **adb push**—Copies files from your computer to the device/emulator.

  Syntax:

  ```
  adb push source destination
  ```

where *source* refers to the file along with its path that you want to copy, and *destination* refers to the place in the device or emulator where you want to copy the file.

- **adb pull**—Copies files from the device/emulator to your computer.

  Syntax:

  ```
  adb pull source [destination]
  ```

- **adb shell**—Displays the shell prompt where you can issue Unix commands. You can see the names of different files and folders of the emulator after issuing an ls command.You can also issue the commands to list, rename, and delete applications from the emulator. For example, to delete the file song1.mp3 that you pushed into the emulator, you issue the rm command,
- **adb install**—Installs an application from your computer to the device/emulator.

  Syntax:

  ```
  adb install appname.apk
  ```

Launching Android Applications on a Handset

To load an application onto a real handset, you need to plug a handset into your computer, using the USB data cable. You first confirm whether the configurations for debugging your application are correct and then launch the application as described here:

1. In Eclipse, choose the Run, Debug Configurations option.

2. Select the configuration HelloWorldApp_configuration, which you created for the

HelloWorldApp application.

3. Select the Target tab, set the Deployment Target Selection Mode to Manual. The Manual option

allows us to choose the device or AVD to connect to when using this launch configuration.

4. Apply the changes to the configuration file by clicking the Apply button.

5. Plug an Android device into your computer, using a USB cable.

6. Select Run, Debug in Eclipse or press the F11 key. A dialog box appears, showing all available configurations for running and debugging your application. The physical device(s) connected to the computer are also listed. Double-click the running Android device. Eclipse now installs the Android application on the handset, attaches a debugger, and runs the application.

•

.