Android Exercise (Part2)

Set up the android development environment

If you did not install the SDK, please refer to previous document or this link
https://developer.android.com/studio

Task 1	Hello world version 1

Knowledge learn in this task:

Function: “onCreate”	1) The first function called in the program 2) All coding in “onCreate” should be placed after the **super.onCreate(savedInstanceState);**

Procedure of the task:

Step 1
Create project in android studio

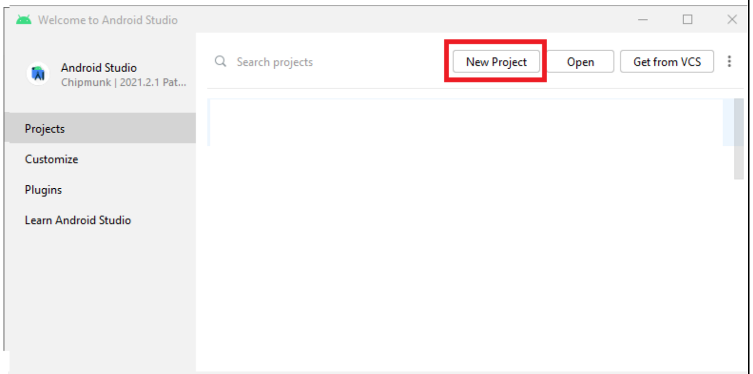
01) Click “All program->Android Studio->Android Studio”
➔ “Start a new Android Studio Project”

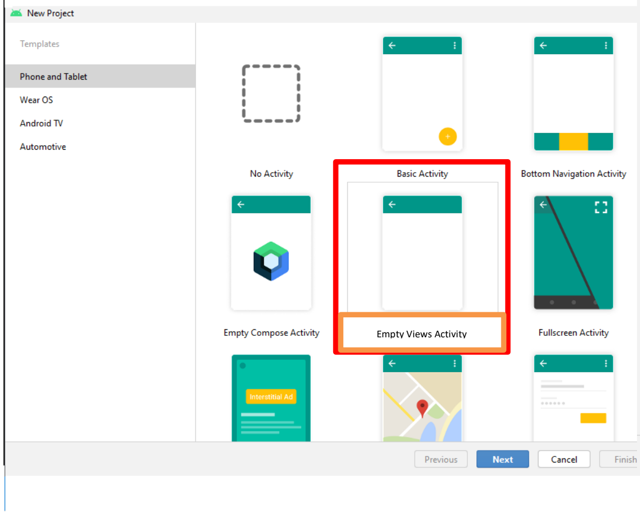
02) Enter the following information

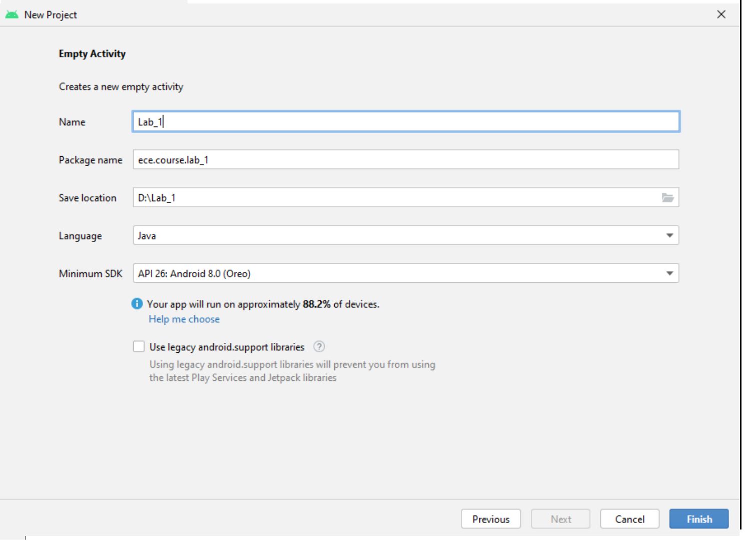
1) Choose an **Empty Views Activity**

2) Enter Application name
(You can use your own or using the “lab_1”) Now, package name is “**ece.course.lab_1**”

3) Select Minimum SDK to
“API 26: Android 8.0 (Oreo)”.
Put Language as **“Java”** Press “Finish>”







Empty Activity

Creates a new empty activity

Name: Lab_1

Package name: ece.course.lab_1

Save location: D:\Lab_1

Language: Java

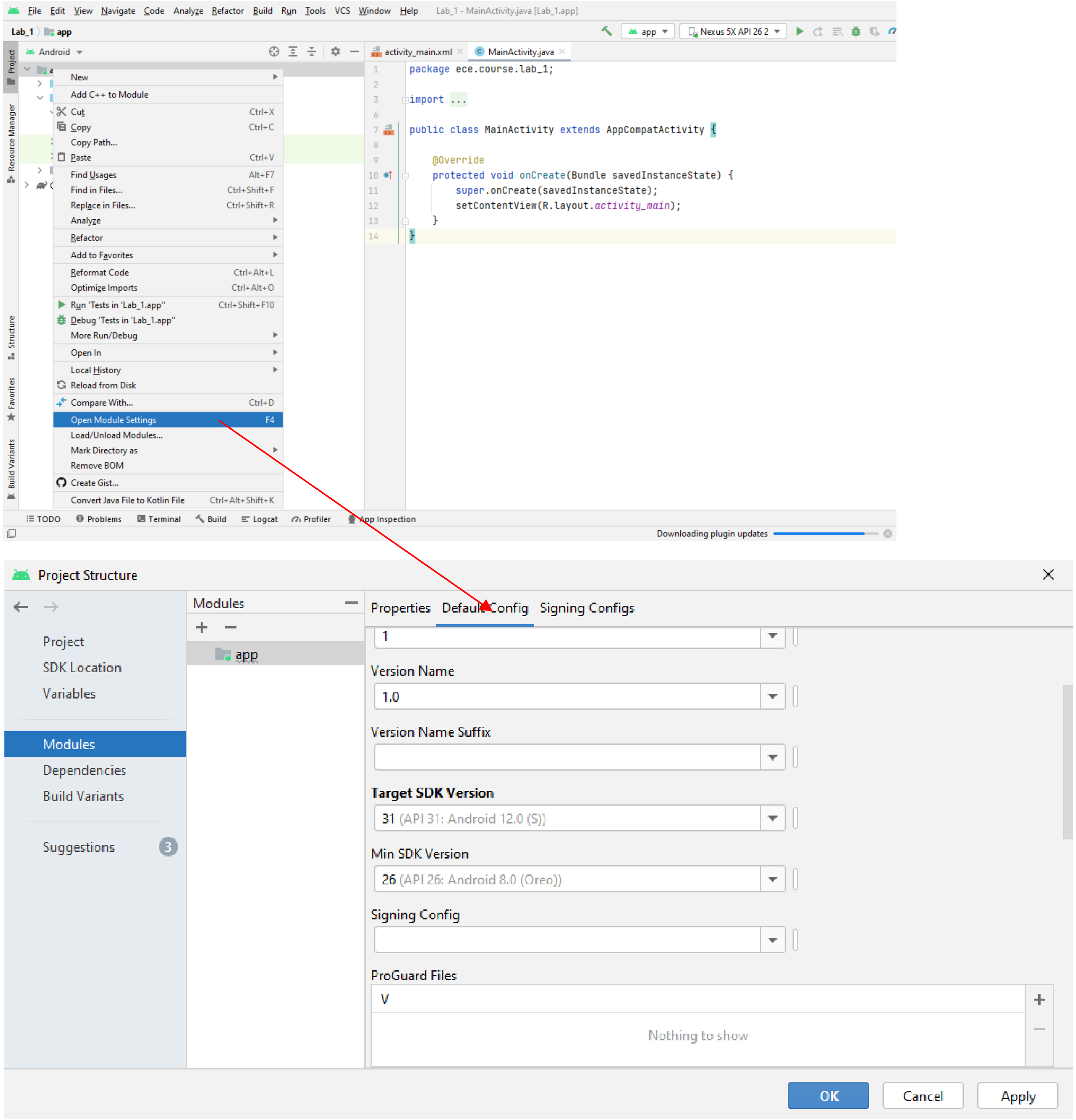
Minimum SDK: API 26: Android 8.0 (Oreo)

Your app will run on approximately **88.2%** of devices.
Help me choose

☐ Use legacy android.support libraries
Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

4) Configure build target version:

In app, Right Click->Open Module Settings; In Default Config tab, -> Select Android 12.0



```
package ece.course.lab_1;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Version Name: 1.0

Version Name Suffix:

Target SDK Version: 31 (API 31: Android 12.0 (S))

Min SDK Version: 26 (API 26: Android 8.0 (Oreo))

Signing Config:

ProGuard Files: Nothing to show

Choose “OK” and close it

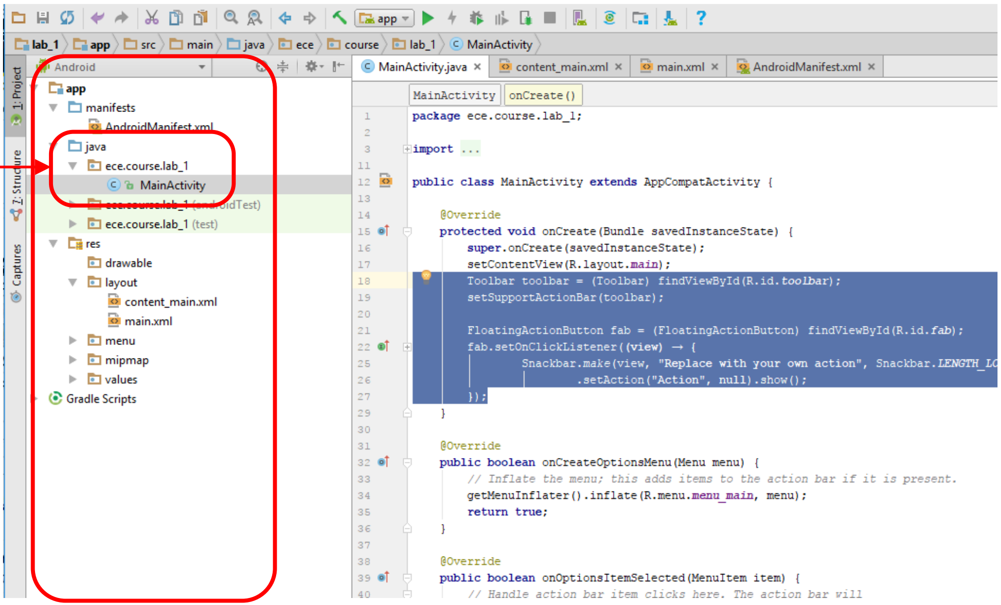
Step 2

Set display by coding

01) Open the java file of the created activity

*you can find the file in "Package Explorer"

**The file is in <Project name>
➔ "src"
➔ <package name>
➔ <activity name>



```
package ece.course.lab_1;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
        fab.setOnClickListener((view) → {
                Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH_LC
                        .setAction("Action", null).show();
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
```

***You can remove all the highlighted Toolbar code(if there is any)

02) Set up new "TextView" for displaying the message.(Refer * in P.4)

a) Create a new TextView

Code:
TextView textView = new TextView(this);

Class
Variable name
Constructing variable

b) Set the TextView's text

Code:
textView.setText("Hello world!!");

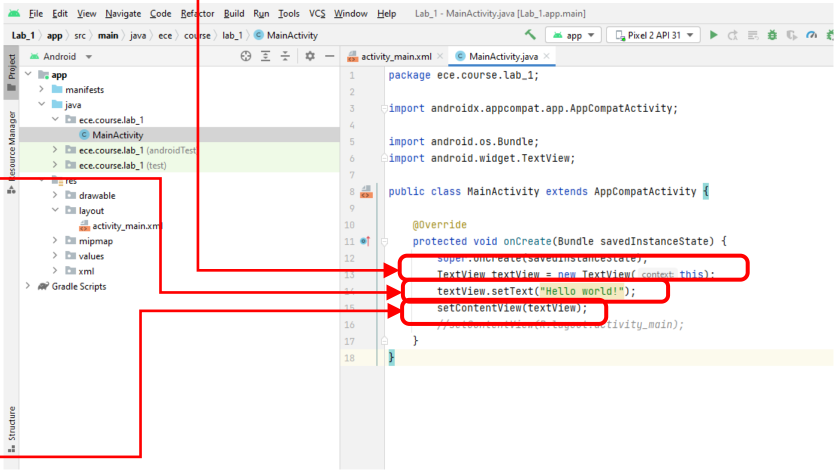
TextView name
Text need to display

c) Change the display to showing the text

Code change:

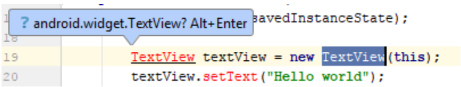
1) setContentView(R.layout.activiy_main);
➔ setContentView(textView);

TextView name



```
package ece.course.lab_1;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TextView textView = new TextView( context: this);
        textView.setText("Hello world!");
        setContentView(textView);
        //setContentView(R.layout.activity_main);
    }
}
```

2) In res->layout, we only use activity_main.xml, remove others(if there is any)



```
                                savedInstanceState);
        TextView textView = new TextView(this);
        textView.setText("Hello world");
```

*If you see error picture like this , click on it and select "Alt+Enter" to import the library.

Notice:

3) main.xml or activity_main.xml, the exact name doesn't matter

4) You may see some other codes like "onCreateOptionMenu","onOptionsItemSelected" in the java file, these codes are not being used in this lab and you can ignore it

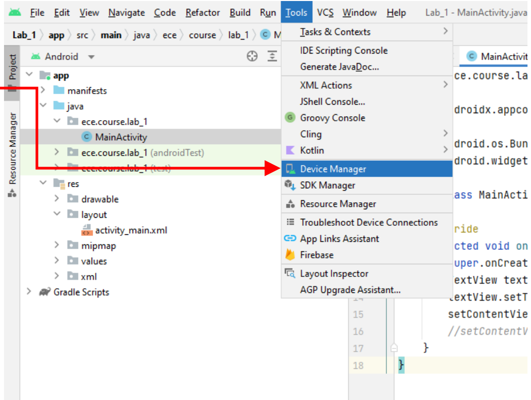
5) The toolbar code also not being used, you can ignore it or just delete the toolbar code

Step 3

Test the apps by emulator

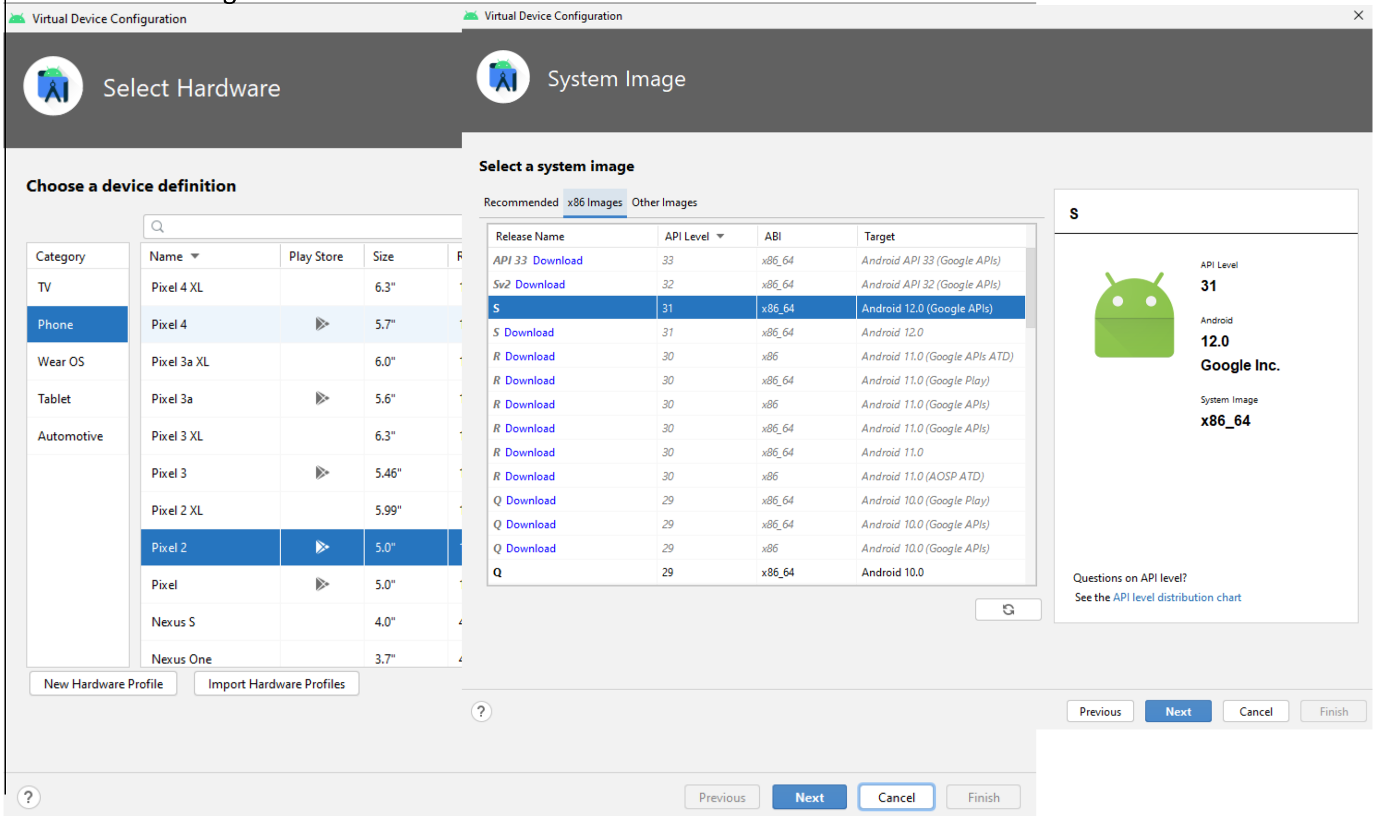
For the first time to use emulator, you need to do some configuration.

a) Open Device Manager



```
ce.course.la
droidx.appco
droid.os.Bun
droid.widget
ass MainActi
ride
cted void on
uper.onCreat
extView text
extView.setT
setContentVie
//setContentV
    }
}
```

b) Click "Create Device" in Device Manager



Choose a device definition

Category: TV, Phone, Wear OS, Tablet, Automotive

Name	Play Store	Size
Pixel 4 XL		6.3"
Pixel 4	▶	5.7"
Pixel 3a XL		6.0"
Pixel 3a	▶	5.6"
Pixel 3 XL		6.3"
Pixel 3	▶	5.46"
Pixel 2 XL		5.99"
Pixel 2	▶	5.0"
Pixel	▶	5.0"
Nexus S		4.0"
Nexus One		3.7"

Select a system image

Release Name	API Level	ABI	Target
API 33 Download	33	x86_64	Android API 33 (Google APIs)
Sv2 Download	32	x86_64	Android API 32 (Google APIs)
S	31	x86_64	Android 12.0 (Google APIs)
S Download	31	x86_64	Android 12.0
R Download	30	x86	Android 11.0 (Google APIs ATD)
R Download	30	x86_64	Android 11.0 (Google Play)
R Download	30	x86	Android 11.0 (Google APIs)
R Download	30	x86_64	Android 11.0 (Google APIs)
R Download	30	x86_64	Android 11.0
R Download	30	x86	Android 11.0 (AOSP ATD)
Q Download	29	x86_64	Android 10.0 (Google Play)
Q Download	29	x86_64	Android 10.0 (Google APIs)
Q Download	29	x86	Android 10.0 (Google APIs)
Q	29	x86_64	Android 10.0

S

API Level: 31

Android: 12.0

Google Inc.

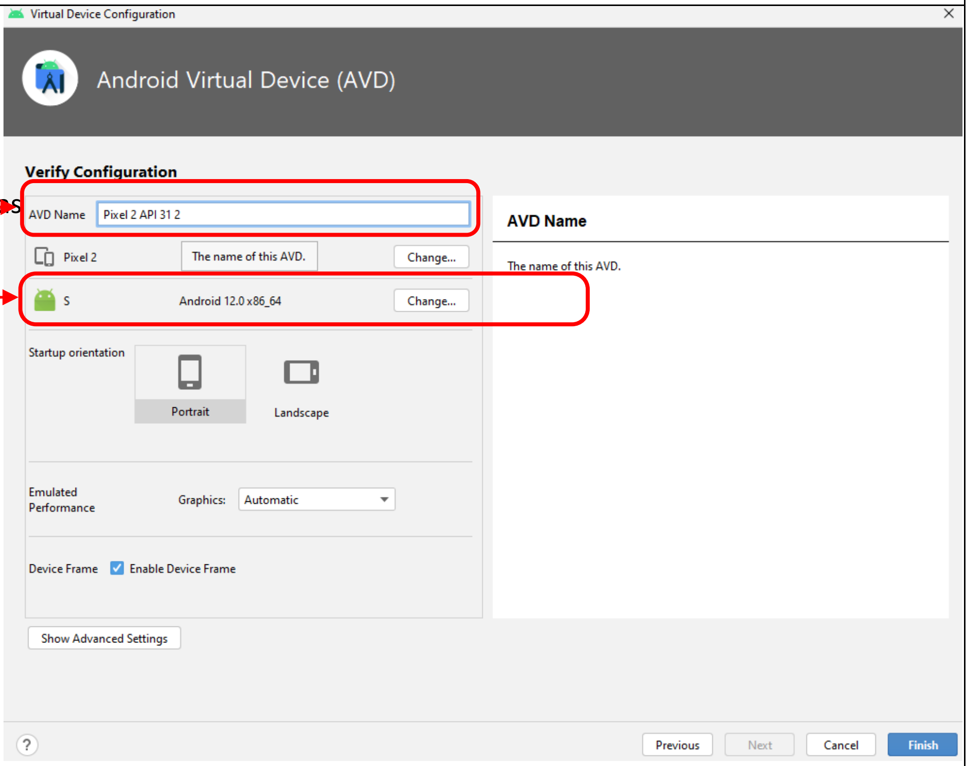
System Image: x86_64

c) Enter the name

(You can use any name)

d) Select the Target

(Choose "Android 12.0", since the lab is use Android 12.0 as the build target)



Virtual Device Configuration

Android Virtual Device (AVD)

Verify Configuration

AVD Name: Pixel 2 API 31 2

Pixel 2 — The name of this AVD. — Change...

S — Android 12.0 x86_64 — Change...

Startup orientation: Portrait, Landscape

Emulated Performance — Graphics: Automatic

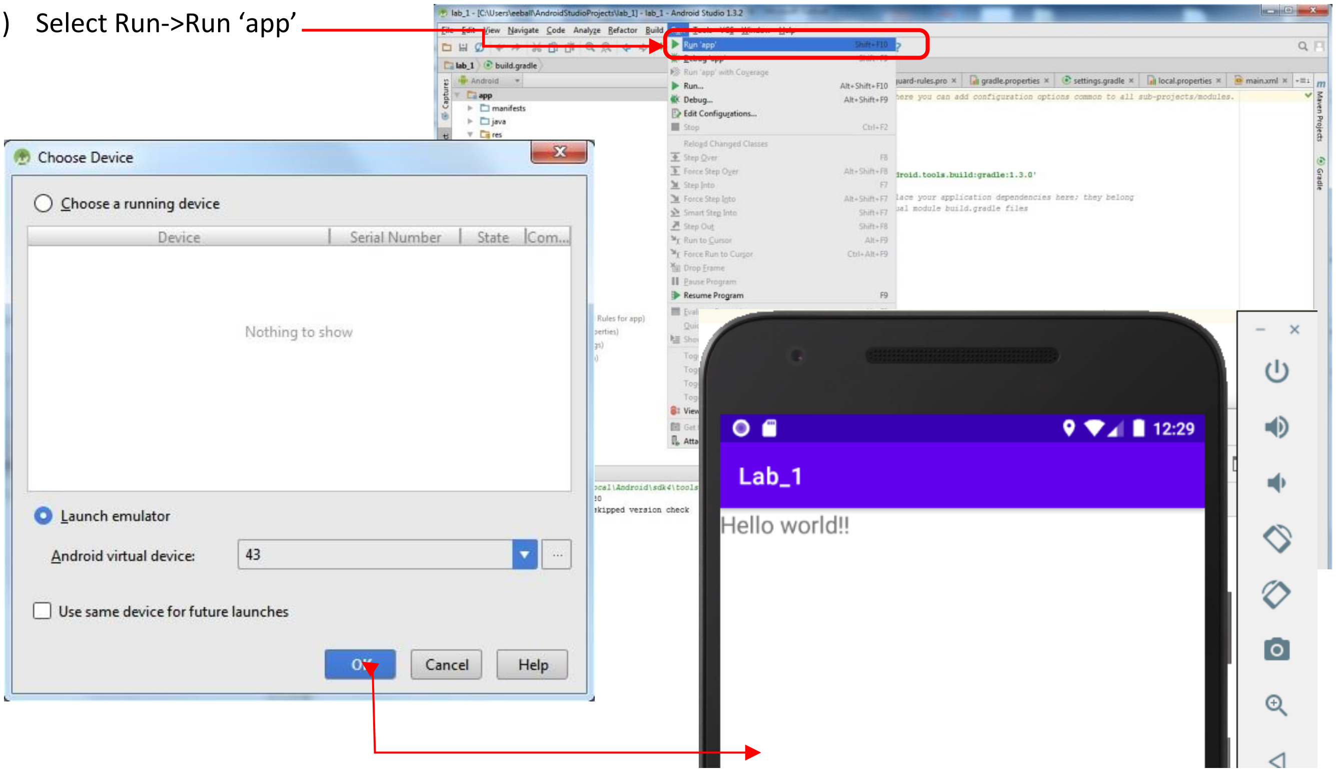
Device Frame — ☑ Enable Device Frame

AVD Name

The name of this AVD.

Select run/debug on the emulator

a) Select Run->Run 'app'



lab_1 - [C:\Users\eeball\AndroidStudioProjects\lab_1] - lab_1 - Android Studio 1.3.2

Choose Device

○ Choose a running device

Device	Serial Number	State	Com...

Nothing to show

● Launch emulator

Android virtual device: 43

☐ Use same device for future launches

Lab_1

Hello world!!

Task 2 | Hello world version 2

Knowledge learn in this task:

"res" ➔ "values" ➔ "strings.xml"	1) It is a file for storing some useful values. 2) It is an editable file.

Procedure of the task:

Step 1
Create project in android studio

Use the method in Task 1 to create a new Project

Project name: Lab_1_2 (or your own one)

Target SDK: Android 12.0

Package name: ece.course.lab_1_2

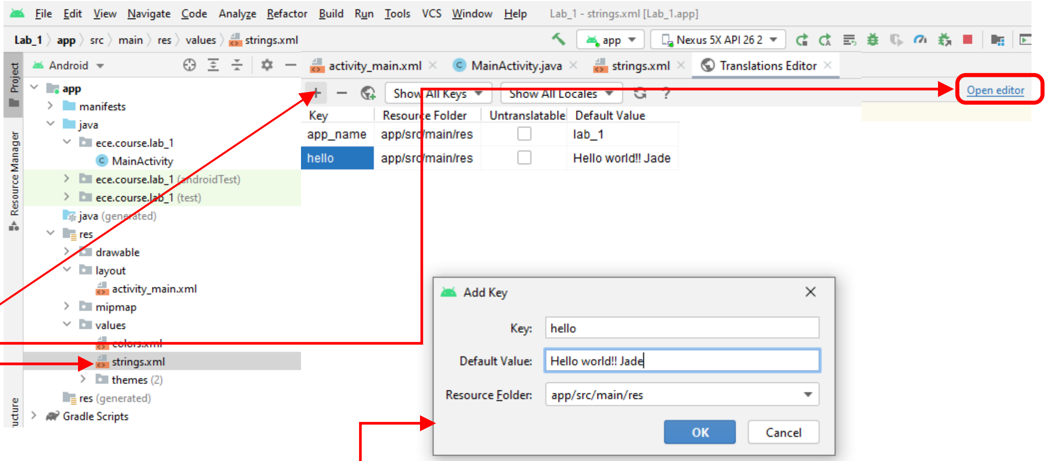
Step 2
Set display by xml

01) Open the "strings.xml"

*The file is in <Project name>
➔ "res"
➔ "values"
➔ "strings.xml"

02) Open Editor. Add an element "hello"

03) Change value to "Hello world!!" + your name



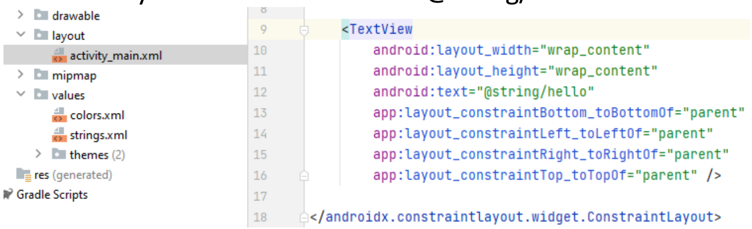
Key	Resource Folder	Untranslatable	Default Value
app_name	app/src/main/res	□	lab_1
hello	app/src/main/res	□	Hello world!! Jade

Key: hello

Default Value: Hello world!! Jade

Resource Folder: app/src/main/res

04) Make sure you have android:text="@string/hello" added in activity_main.xml



```
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

(Notes: Make sure you are using setContentView(R.layout.activity_main) instead of setting focus to textview)

Step 3

Test the apps by emulator

Task 3	Set up UI and try to use different views

Knowledge learn in this task:

"res" ➔ "layout"	1) A place to store the user's layout 2) All layout is stored in xml format
"res" ➔ "drawable"	1) A place to store the image
ImageView	1) A kind of View, which use to display image 2) You can custom what image it will display.

Procedure of the task:

Step 1

Open the project of last task

Step 2

Set the UI

01) Set the resources used in the layout

a) Open “strings.xml”

b) Add new string

1) Click “Open editor”

2) Click “Add…”

3) Enter “Key”

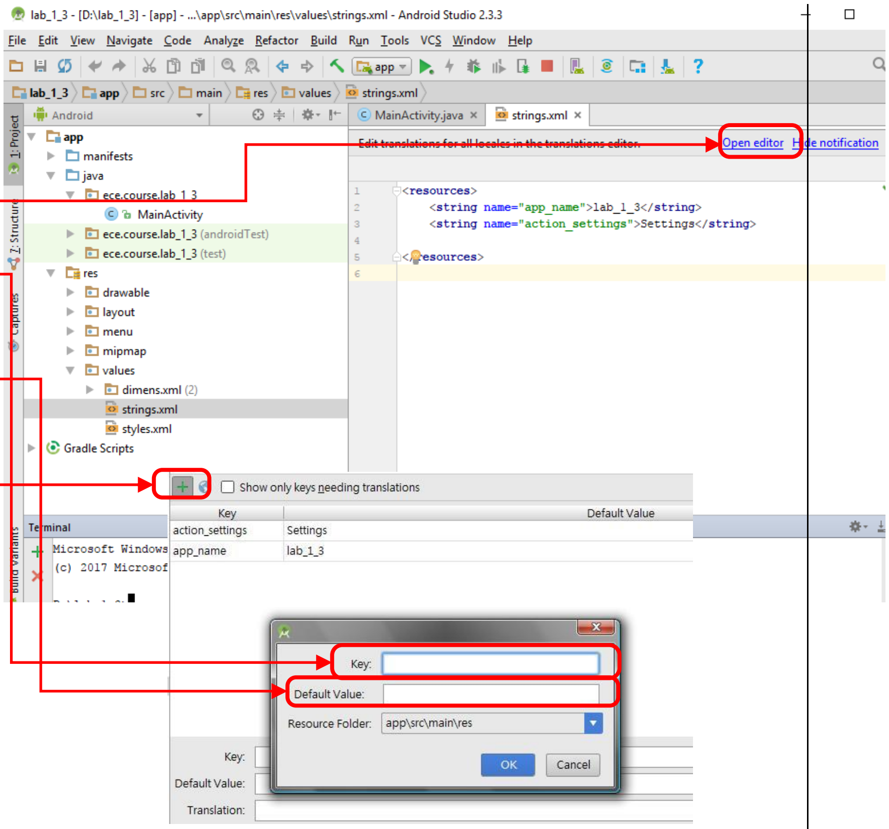
(This time enter “nameBoy” as the “Key”)

4) Enter “Default Value”

(This time enter “A boy” as the “Default Value”)

c) Add one more String
Name: nameGirl
Value: A girl

d) Save the file



Edit translations for all locales in the translations editor.

```
<resources>
    <string name="app_name">lab_1_3</string>
    <string name="action_settings">Settings</string>

</resources>
```

□ Show only keys needing translations

Key	Default Value
action_settings	Settings
app_name	lab_1_3

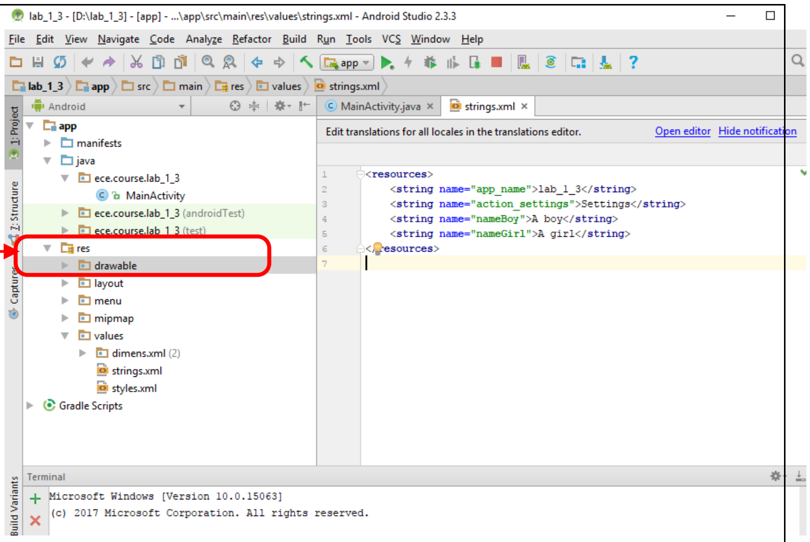
```
Microsoft Windows
(c) 2017 Microsof
```

Key:
Default Value:
Resource Folder: app\src\main\res

02) Open the drawable folder(~~Use Open~~ in -> Explorer)

a) Dig the picture "boy.jpg" into the folder (drawable, but not drawable-v24)

b) Dig the picture "girl.jpg" into the folder



lab_1_3 - [D:\lab_1_3] - [app] - ...\app\src\main\res\values\strings.xml - Android Studio 2.3.3

Edit translations for all locales in the translations editor.

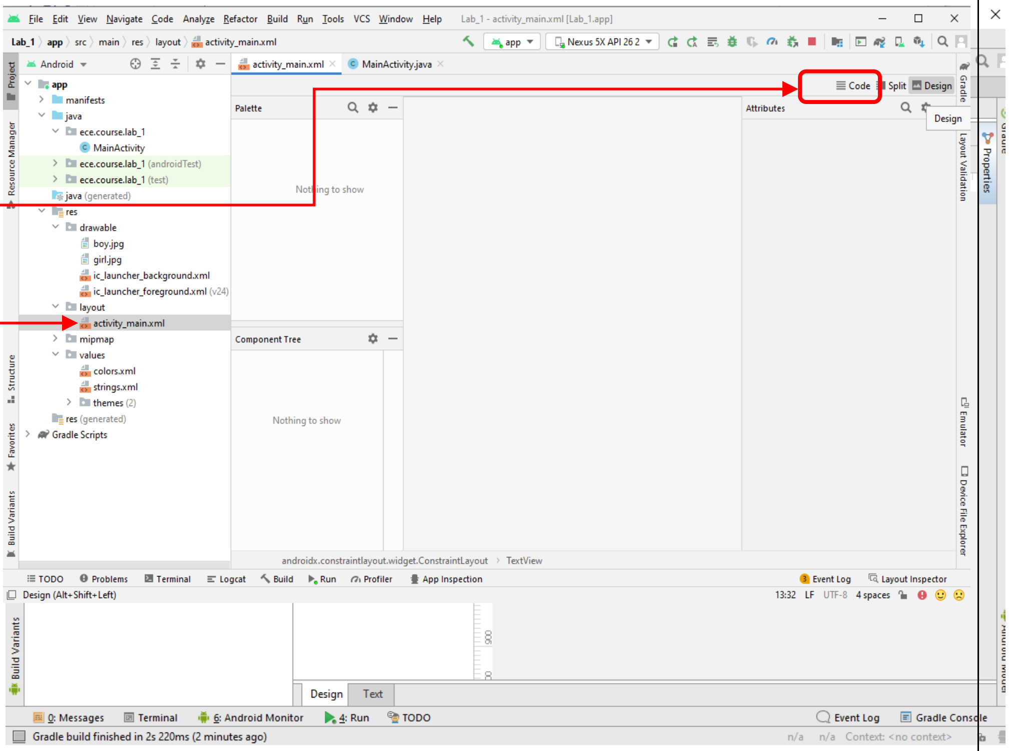
```
<resources>
    <string name="app_name">lab_1_3</string>
    <string name="action_settings">Settings</string>
    <string name="nameBoy">A boy</string>
    <string name="nameGirl">A girl</string>
</resources>
```

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.
```

03) Set the layout

a) Open the "activity_main.xml" under "layout"

b) Open the text mode by click "Code"



Gradle build finished in 2s 220ms (2 minutes ago)

c) You will see many tags in activity_main.xml, for example: hello tag, **delete all of them**

```
<TextView
        android:layout_width=”wrap_content”
        android:layout_height=”wrap_content”
        android:text=”@string/hello”
        />
```

d) Add a new TextView tag in the LinearLayout tag

```
<TextView android:id=”@+id/tvName”
        android:layout_width=”fill_parent”
        android:layout_height=”wrap_content”
        android:textSize=”20sp”
        android:text=”@string/nameBoy” />
```

e) Add a ImageView tag under the TextView tag in the LinearLayout tag

```
<ImageView android:id=”@+id/ivPicture”
        android:layout_width=”fill_parent”
        android:layout_height=”wrap_content”
        android:layout_weight=”1”
        android:src=”@drawable/boy” />
```

f) Add a LinearLayout tag under the ImageView tag in the LinearLayout tag

```
<LinearLayout android:layout_width=”fill_parent”
        android:layout_height=”wrap_content” >
</LinearLayout>
```

g) Add a EditText tag inside the LinearLayout tag (the inner one created by you)

```
<EditText android:id=”@+id/etName”
        android:maxLines=”1”
        android:layout_width=”wrap_content”
        android:layout_height=”wrap_content”
        android:layout_weight=”1” />
```

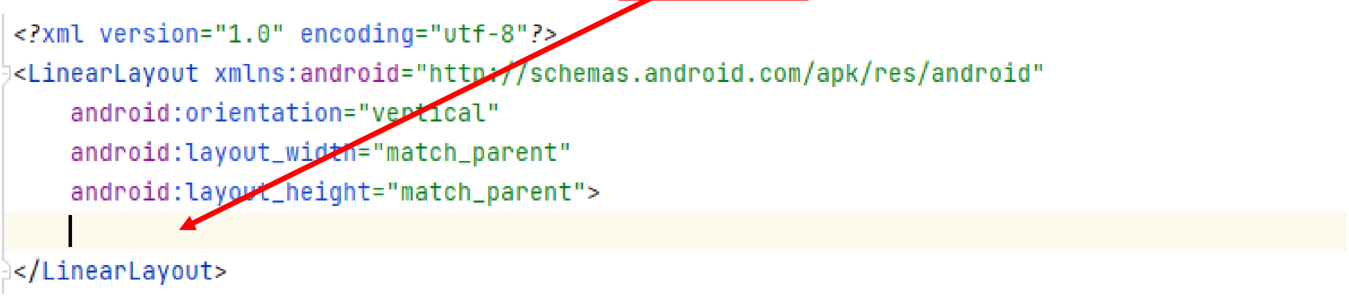
h) Add a Button tag inside the LinearLayout tag (the inner one created by you)

```
<Button android:id=”@+id/btnName”
        android:text=”Change”
        android:layout_width=”wrap_content”
        android:layout_height=”wrap_content” />
```

i) Add a Button tag under the LinearLayout tag

```
<Button android:id=”@+id/btnPicture”
        android:text=”Change Picture”
        android:layout_width=”fill_parent”
        android:layout_height=”wrap_content” />
```

Make sure item d) to i) above should be put in the cursor position as below:

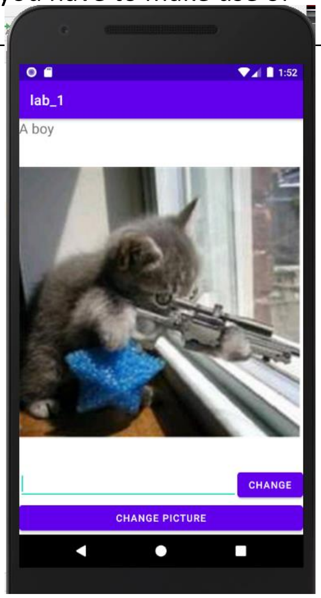


```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    |
</LinearLayout>
```

Notice that we are using LinerLayout, if you are using Relativelayout, you have to make use of android:layout_below properties

Step 3

Test the apps by emulator



Task 4	An app that can change picture by one button and can change the name of the picture.

Knowledge learn in this task:

Button	1) A kind of View, that design act as a button 2) It uses with the OnClickListener
EditText	1) A kind of View, that design for user to input text 2) You can use it to obtain text input from the user.
OnClickListener	1) It mainly uses with button. 2) What will perform in the button click is the coding under the function onClick in it.

Procedure of the task:

Step 1

Open the project of last task

Step 2**(This part is required)**

Set the link between the coding and the display

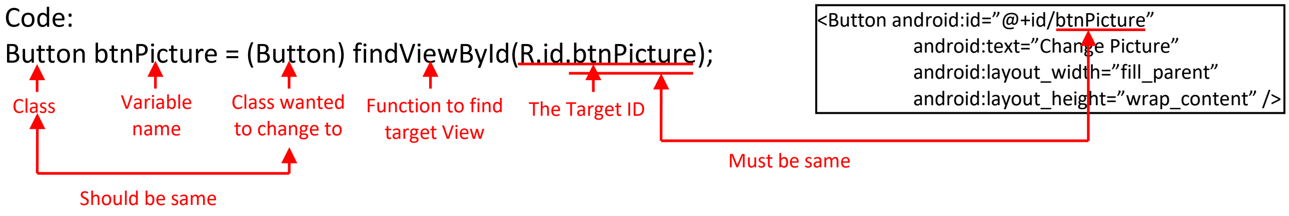
01) Open java file of the activity

02) Start coding in the function “onCreate”

03) Implements the “Change Picture” button

a) Link the Button “btnPicture”

Code:

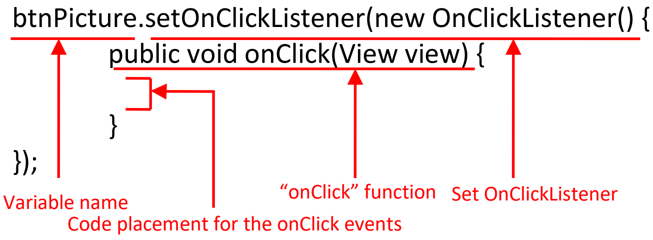


```
Button btnPicture = (Button) findViewById(R.id.btnPicture);
```

```
<Button android:id=”@+id/btnPicture”
        android:text=”Change Picture”
        android:layout_width=”fill_parent”
        android:layout_height=”wrap_content” />
```

b) Implements the onClick events, the logic need to be implemented by you.

Code:



```
btnPicture.setOnClickListener(new OnClickListener() {
        public void onClick(View view) {

        }
});
```

c) Implements a variable for determinant what should display

Code:

boolean isBoy = true;

Class Variable name Initialize value

```
public class ******* extends Activity {
        boolean isBoy = true;
        @Override
        Public void onCreate(Bundle savedInstanceState) {
                Super.onCreate(savedInstanceState);
                setContentView(R.layout.main);
        }
}
```

d) Link the TextView “tvName” (Put in the onClick Events)

Code:

TextView tvName = (TextView) findViewById(R.id.tvName);

e) Change the text of “tvName” (Put in the onClick Events)

Code:

For display boy case:

tvName.setText(R.string.nameBoy);

For display girl case:

tvName.setText(R.string.nameGirl);

Must be same

String for boy
String name: nameBoy

String for girl
String name: nameGirl

Must be same

f) Link the ImageView “ivPicture” (Put in the onClick Events)

Code:

ImageView ivPicture = (ImageView) findViewById(R.id.ivPicture);

g) Change the picture of “ivPicture” (Put in the onClick Events)

Code:

For display boy case:

ivPicture.setImageResource(R.drawable.boy);

For display girl case:

ivPicture.setImageResource(R.drawable.girl);

Must be same

Picture for boy
Picture name: boy.jpg

Picture for girl
Picture name: girl.jpg

Must be same

*It is better, if the coding is placed after **setContentView(R.layout.main);**

04) Implements the “Change Name” button

a) Link the Button “btnName”

b) Implements the onClick events

c) Link the EditText “etName”

Code:

EditText etName = (EditText) findViewById(R.id.etName);

d) Get the Text in “etName”

Code:

String name = etName.getText().toString();

e) Link the TextView

f) Change the “tvName” text

Step 3

Test the apps by emulator

Hand-in method:

N/A

Troubleshot method:

1) "adb kill-server" and "adb start-server" if you encounter terminal problem
2) If emulator is slow response, you can consider using an android phone to show the result

Useful link:

https://developer.android.com/guide