

COMP4336/9336

Lab - Android Studio Preliminary

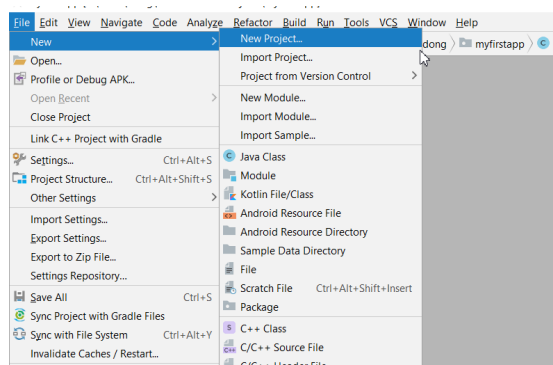
Lab Objectives

1. Learn how to run and create a new project with Android Studio.
2. Develop a simple Java program running on a Samsung android-based smart phone.
3. Learn how to deploy the program to both emulator and real device.
4. Learn how to copy your program to and run it on the smart phone.
5. Learn how to switch to and stop programs running on the smart phone.
6. Learn how to debug your program

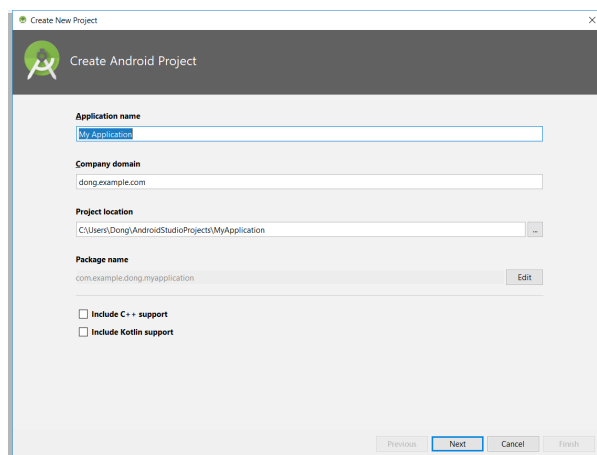
Lab Tasks

TASK1: “Hello World” Program with Android Studio

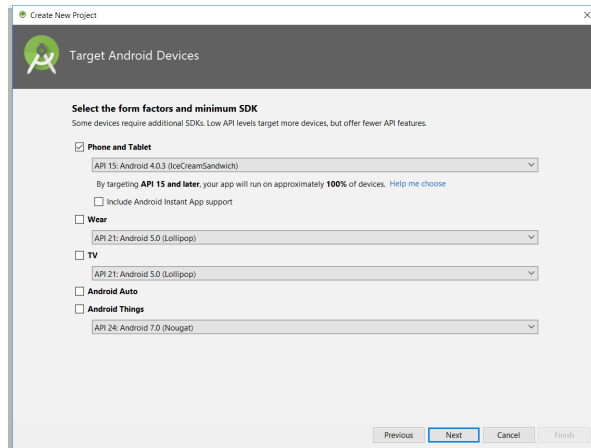
1. Connect the smart phone to your computer.
2. Run Android Studio.
3. Click **File -> New -> New Project**



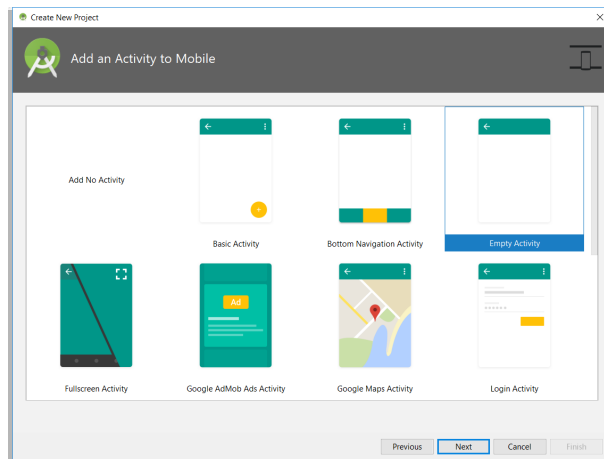
4. In the window that appears, edit the **Application name**. For others, you can just leave them as default. Then, click **Next**.



5. In the window that appears, select the form factors and minimum SDK. Then, click **Next**.

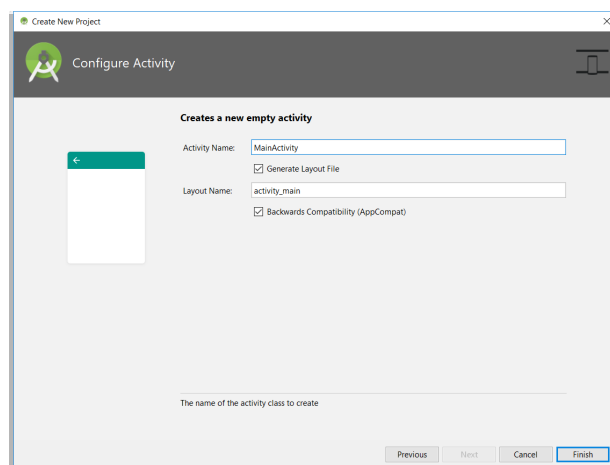


6. In the window that appears, select an activity (use default value). Then, click **Next**.



7. In the window that appears, configure the activity (use default value). Then, click **Finish**.

Your Android project is now set up with some default files and you're ready to begin building the app. Continue to the next part.



How you run your app depends on two things: whether you have a real Android-powered device and whether you're using Android Studio. This tutorial shows you how to install and run your app on a real device and on the Android emulator, and in both cases with either Android Studio or the command line tools.

Before you run your app, you should be aware of a few directories and files in the Android project.


Run on a Real Device

If you have a real Android-powered device, here's how you can install and run your app:

1. Plug in your device to your development machine with a USB cable. If you're developing on Windows, you might need to install the appropriate USB driver for your device. For help installing drivers, see the [OEM USB Drivers](#) document.
2. Enable **USB debugging** on your device.
 - On most devices running Android 3.2 or older, you can find the option under **Settings > Applications > Development**.
 - On Android 4.0 and newer, it's in **Settings > Developer options**.

Note: On Android 4.2 and newer, **Developer options** is hidden by default. To make it available, go to **Settings > About phone** and tap **Build number** seven times. Return to the previous screen to find **Developer options**.

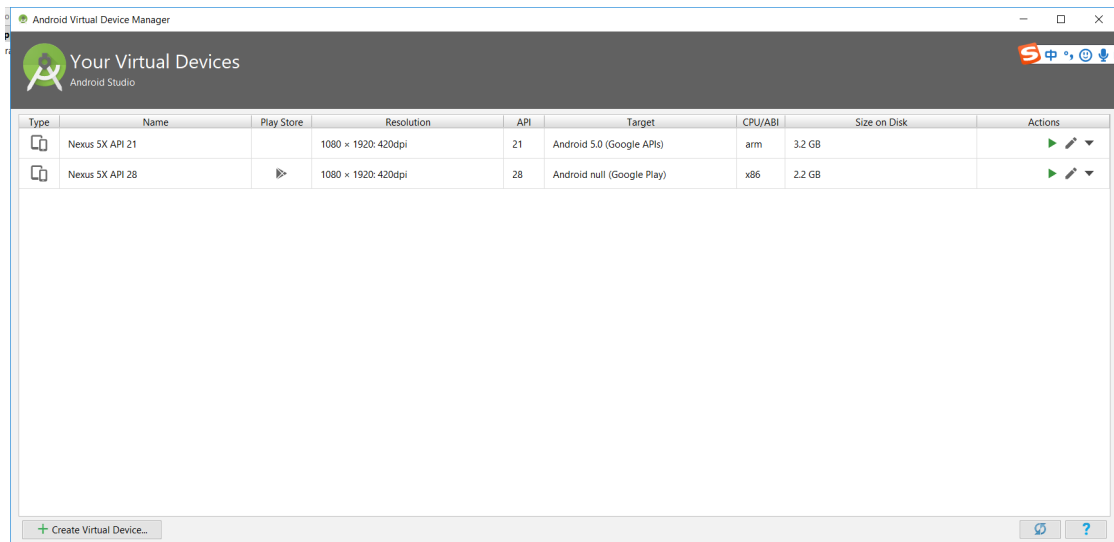
To run the app from Android Studio:

1. Open one of your project's files and click **Run**  from the toolbar.

Android Studio installs the app on your connected device and starts it.

Run on the Emulator

You need to first create an Android Virtual Device (AVD). An AVD is a device configuration for the Android emulator that allows you to model different devices.



The AVD Manager showing a few virtual devices.

To create an AVD:

1. Launch the Android Virtual Device Manager:
 - a. In Android Studio, click Android Virtual Device Manager from the toolbar.
2. In the *Android Virtual Device Manager* panel, click **Creat Virtual Device**.
3. Select a hardware for the AVD, or you can design a new hardware from **New Hardware Profile**. Then click **Next**.
4. Select a system image for the AVD (you may need to download a API in the first time). Then click **Next**.
5. Configure detail for the AVD. Then click **Finish**.

To run the app from Android Studio:

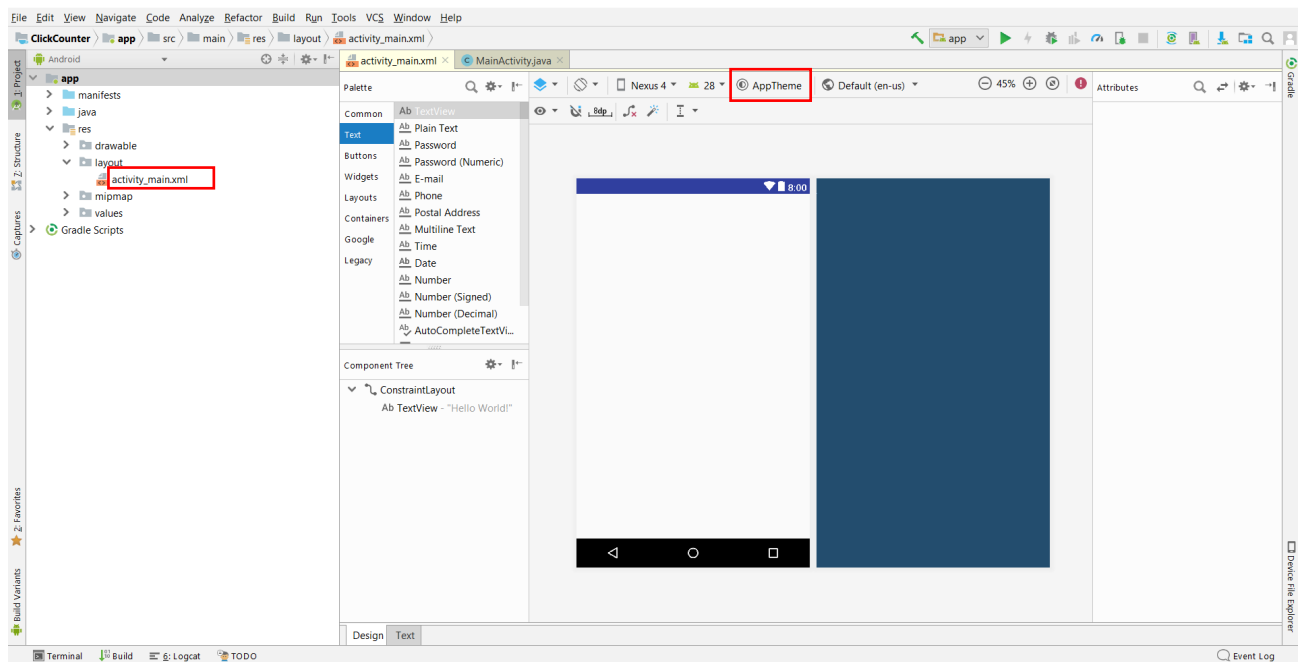
1. Open one of your project's files and click **Run** from the toolbar.
2. In the window that appears, select the AVD you created and click **OK**.

TASK2: “Click Counter” Program AIMS:

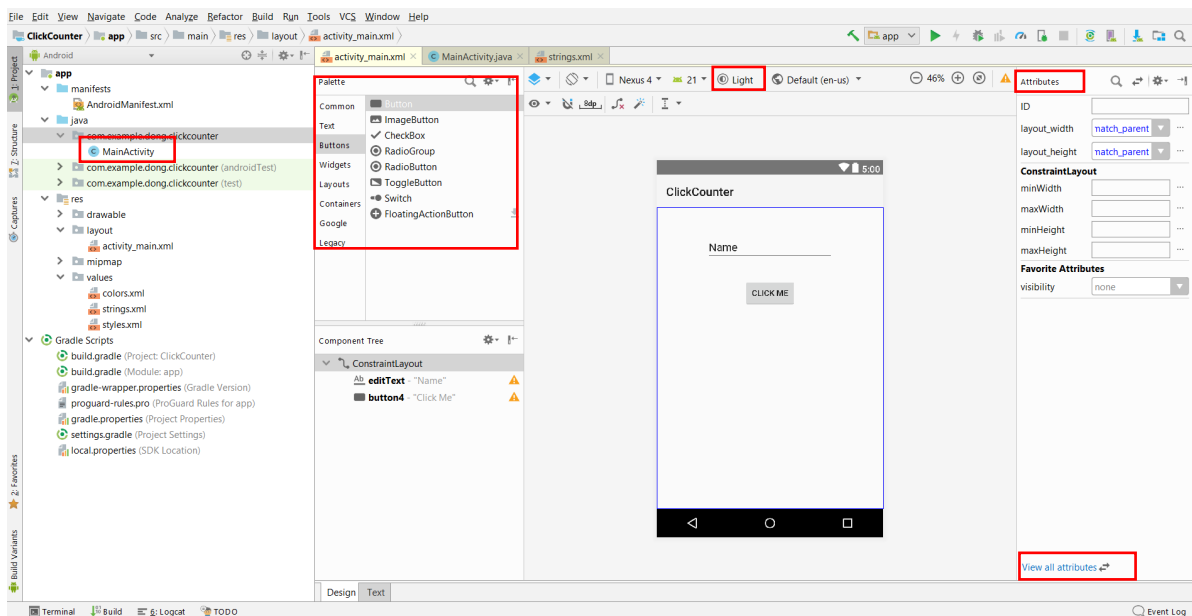
1. Study the code of your *HelloWorld* program.
2. Develop the *Click Counter* program so that it records and shows the number of times *Click Me* button has been clicked

Instruction:

1. Create a new Android Application Project named *ClickCounter*. Leave all options as their default values except the project name.
2. Go to the layout designer by double click on `res\layout\activity_main.xml` such as below picture:



3. Select a **Theme** for you app and then you can design the graphical layout of your project by drag-and-drop the component. By adding a Button and a Large Text, create the following layout.
 - You can change the Text of your button using **Attributes** window in the right side of your IDE such as you can see in the picture.



- By clicking on each object within the graphical layout (such as the button), you can change the details of the object in **Attributes** window. These attributes are stored in an XML file which is named “activity_main.xml” in this project.
 - You don’t need to edit the xml file directly.
 - For each graphical object, you need to know the id name which is necessary for manipulating its properties from your Java classes.
4. In order to manage click activity for the button in Android, you can do as below:
- Open MainActivity class from the src folder in Package Explorer window (at the left side of your IDE)
 - Then, declare your button in **onCreate** method of the class like

```
final Button btn = (Button) findViewById(R.id.button1);
```

- Then use the button variable as below:

```
btn.setOnClickListener(new View.OnClickListener() {  
  
    public void onClick(View v) {  
        myClick(v); /* my method to call new intent or activity */  
    }  
});
```

- As an example, you can handle the click event as :

```
public void myClick(View v) {  
    TextView txCounter = (TextView) findViewById(R.id.textView1);  
    txCounter.setText("yourtext");  
}
```

- Now, the following is the source code of class MainActivity for implementing our simple *ClickCounter* program. You should complete the **myClick** method by your own code.

Note: As this is the first lab, the below code has been provided for you but for next lab exercises you should be able to create the project and manage all methods and codes by yourself.

```
package com.example.clickcounter;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // set default counter for the text box

        Button clickMeBtn = (Button) findViewById(R.id.button1);
        clickMeBtn.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                myClick(v); /* my method to call new intent or
activity */
            }
        });
    }

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

    public void myClick(View v) {

        // Write your own code

    }
}
```