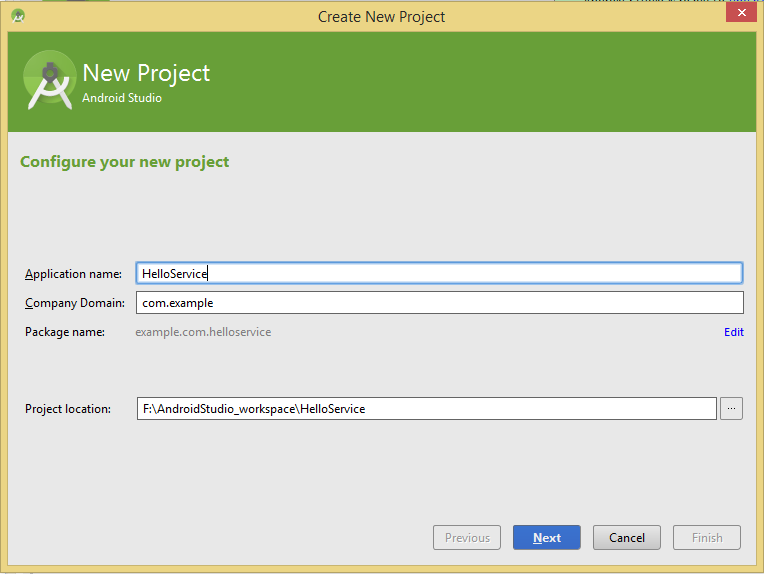
Lab 3: Android Service (Android Studio)

**Step 1:**

Create a simple Android Application using Android Studio. Follow the option **File -> NewProject**. Now name your application as HelloService using the wizard window as follows:



**Step 2:**

Following is the content of the modified main activity file java/com.example.helloservice/MainActivity.java. This file can include each of the fundamental lifecycle methods. We have added *startService()* and *stopService()* methods to start and stop the service.

package example.com.helloservice;

import android.os.Bundle;

import android.app.Activity;

import android.view.Menu;

import android.content.Intent;

import android.view.View;

public class MainActivity extends Activity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

//getMenuInflater().inflate(R.menu.activity\_main, menu);

return true;

}

// Method to start the service

public void startService(View view) {

startService(new Intent(getBaseContext(), MyService.class));

}

// Method to stop the service

public void stopService(View view) {

stopService(new Intent(getBaseContext(), MyService.class));

}

}

Create a new class in the com.example.helloservice package and named **MyService**. Following is the content of *java/com.example.helloservice/MyService.java*. This file can have implementation of one or more methods associated with Service based on requirements. For now we are going to implement only two methods *onStartCommand()* and *onDestroy()*:

**package** com.example.helloservice;

**import** android.app.Service;

**import** android.content.Intent;

**import** android.os.IBinder;

**import** android.widget.Toast;

**public** **class** MyService **extends** Service {

@Override

**public** IBinder onBind(Intent arg0) {

**return** **null**;

}

@Override

**public** **int** onStartCommand(Intent intent, **int** flags, **int** startId) {

// Let it continue running until it is stopped.

Toast.*makeText*(**this**, "Service Started", Toast.*LENGTH\_LONG*).show();

**return** *START\_STICKY*;

}

@Override

**public** **void** onDestroy() {

**super**.onDestroy();

Toast.*makeText*(**this**, "Service Destroyed", Toast.*LENGTH\_LONG*).show();

}

}

Following will the modified content of *AndroidManifest.xml* file. Here we have added <service.../> tag to include our service:

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="example.com.helloservice" >

<application

android:allowBackup="true"

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>

<service android:name=".MyService" />

</application>

</manifest>

Following will be the content of *res/layout/activity\_main.xml* file to include two buttons:

<RelativeLayout 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:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin" tools:context=".MainActivity">

<Button android:id="@+id/btnStartService"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="@string/start\_service"

android:onClick="startService"/>

<Button android:id="@+id/btnStopService"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="@string/stop\_service"

android:onClick="stopService"

android:layout\_marginTop="69dp"

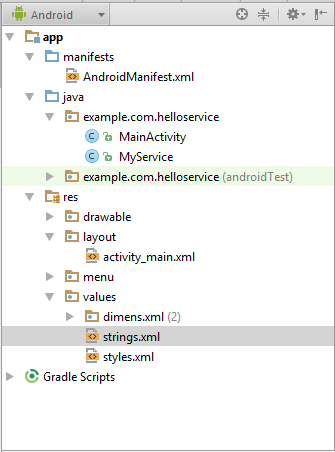
android:layout\_below="@+id/btnStartService"

android:layout\_alignParentLeft="true"

android:layout\_alignParentStart="true" />

</RelativeLayout>

Following will be the content of *res/values/strings.xml* to define two new constants:



<?xml version="1.0" encoding="utf-8"?>

<resources>

<string name="app\_name">HelloService</string>

<string name="hello\_world">Hello world!</string>

<string name="action\_settings">Settings</string>

<string name="title\_activity\_main">MainActivity</string>

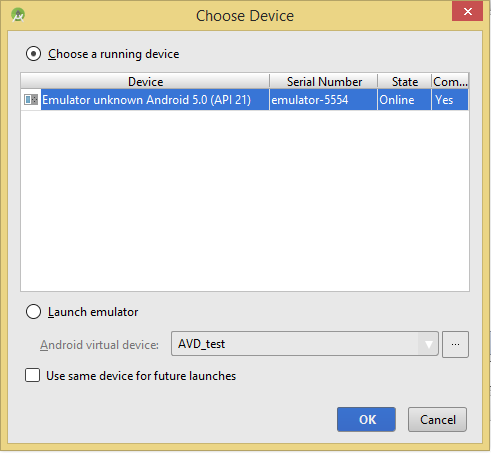
<string name="start\_service">Start Service</string>

<string name="stop\_service">Stop Service</string>

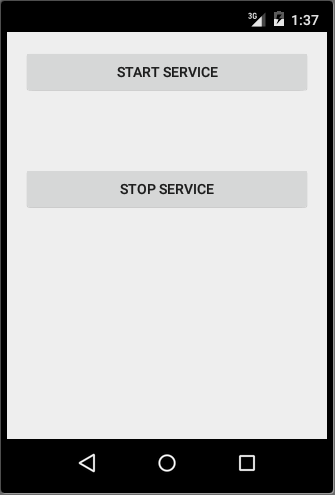
</resources>

Step 3:

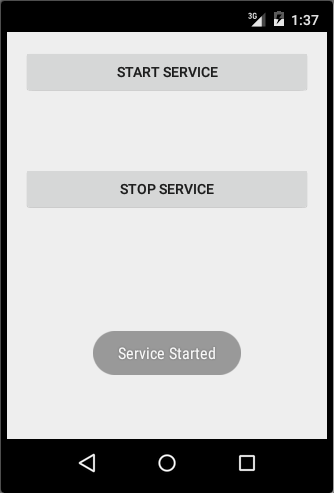
Let's try to run our modified **Hello Service application** we just modified. I assume you had created your AVD while doing environment setup. At first, start your existed **AVD**, and then click **Run** Eclipse Run Icon icon from the toolbar. Choose this AVD, and click “OK”:



Then the AVD will display like as following:



Now to start your service, let's click on **Start Service** button, this will start the service and as per our programming in *onStartCommand()* method, a message Service Started will appear on the bottom of the simulator as follows:



When you click on **Stop Service** button, this will start the service and as per our programming in *onDestroy()* method, a message Service Destroyed will appear on the bottom of the simulator as follows:

