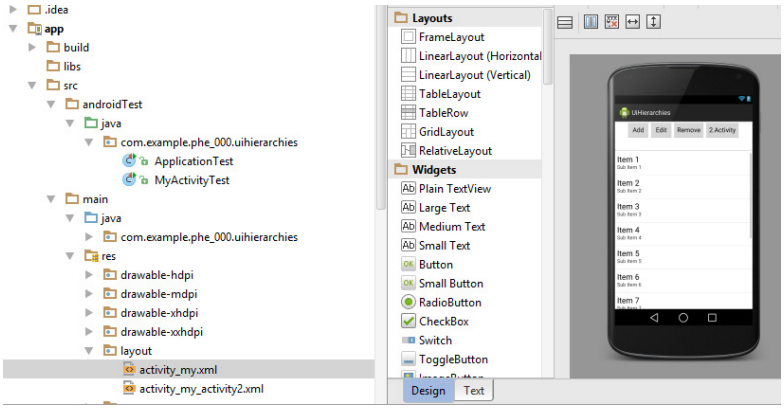
Exercise 1. List on Android

1. Create a new project by Android Studio, using SDK 17.
2. Open the res/layout and the xml-file (activity) in Design mode:

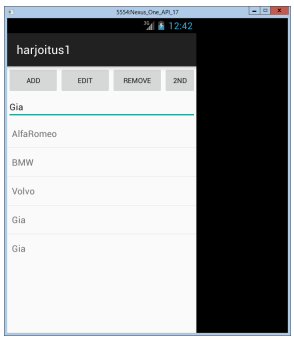


3. Compose the Linear layout of the emulator screen like the picture (above). At the top is 4 buttons: *Add*, *Edit*, *Remove* and *2nd Activity*. For those below the line for text editing: *EditText* (*PlainText*). Rest of the screen is covered by a list (*ListView*).

4. After that Activity xml file looks like this in a text mode:

```
▼<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="fill_parent"
 android:orientation="vertical">
 ▼<LinearLayout android:id="@+id/linearLayout1" android:layout_width="match_parent" android:layout_height="wrap_content" android:layout_gravity="center">
    <Button android:id="@+id/button1" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Add"/>
    <Button android:id="@+id/button3" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Edit"/>
    <Button android:id="@+id/button2" android:layout_width="wrap_content" android:layout_height="wrap_content" android:layout_gravity="fill_vertical" android:text="Remove"/>
    <Button android:id="@+id/button4" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="2nd"/>
  </LinearLayout>
  <EditText android:id="@+id/editText1" android:layout_width="match_parent" android:layout_height="wrap_content"/>
  <ListView android:id="@+id/myListView" android:layout_width="fill_parent" android:layout_height="wrap_content"/>
</LinearLayout>
```

5. Test now if this functions by the emulator. In this phase there is nothing to show on a list.



6. Make up a list of the code below:

```
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;

import java.util.ArrayList;

import static android.widget.AdapterView.*;

public class MyActivity extends Activity {

    static ArrayList<String> carList = new ArrayList<String>();

    private static void getCars(){
        carList.add("AlfaRomeo");
        carList.add("BMW");
        carList.add("Corvette");
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_my);

        getCars();

        final ListView myListView = (ListView)findViewById(R.id.myListView);
        final ArrayAdapter<String> aa;
        aa = new ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,carList);
        myListView.setAdapter(aa);
```

7. Test by the emulator.

8. The functionality of the *Add button* is here below; the new mark (of the car for the list) is done and is defined by *EditText line as* follows:

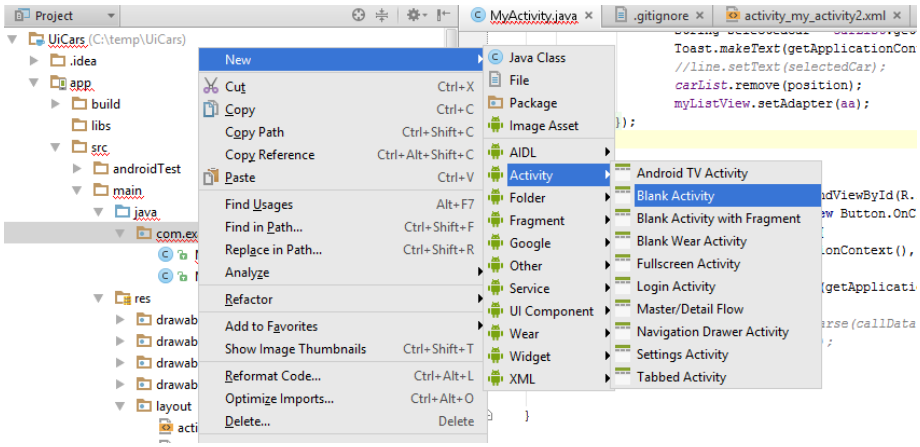
```
//for add button
Button addCar = (Button) findViewById(R.id.button1);
final EditText line = (EditText)findViewById(R.id.editText1);

addCar.setOnClickListener(new Button.OnClickListener() {
    public void onClick(View arg0) {
        String mark = line.getText().toString();
        carList.add(mark);
        myListView.setAdapter(aa);
    }
});
```

9. Do similarly the functionality for the *Edit button*:

```
//edit button
Button editCar = (Button) findViewById(R.id.button3);
editCar.setOnClickListener(new Button.OnClickListener() {
    public void onClick(View arg0) {
        // register onClickListener to handle click events on each item
        myListView.setOnItemClickListener(new OnItemClickListener() {
            // argument position gives the index of item which is clicked
            public void onItemClick(AdapterView<?> arg0, View v, int position, long arg3) {
                String selectedCar = carList.get(position);
                Toast.makeText(getApplicationContext(), "Car Selected : " + selectedCar, Toast.LENGTH_LONG).show();
                line.setText(selectedCar);
            }
        });
    }
});
```

10. Now, I think you are able to do the *Remove button* functionality by your self ..

11. Next, add a new *activity* by clicking mouse right over the name of the package (under java) *New* ..



12. The functionality for the *2nd Activity button* is implemented using the *Intent as follows*:

```
//Second activity button
Button secondActivity = (Button) findViewById(R.id.button4);
secondActivity.setOnClickListener(new Button.OnClickListener() {
    public void onClick(View arg0) {
        Toast.makeText(getApplicationContext(), "Second Activity Selected !", Toast.LENGTH_LONG).show();
        //--intent
        Intent intent = new Intent(getApplicationContext(), MyActivity2.class);
        startActivity(intent);
        //callIntent.setData(Uri.parse(callData));
        //startActivity(callIntent);
    }
});
```

13. Test the 2nd activity, the second screen, and return back to the first screen (original activity) by back button of the emulator.

14. Test the application one more time by the real phone.