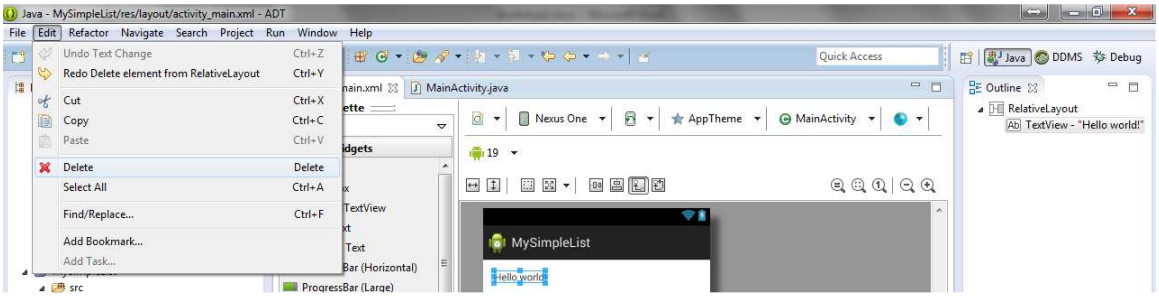
1. Simple List

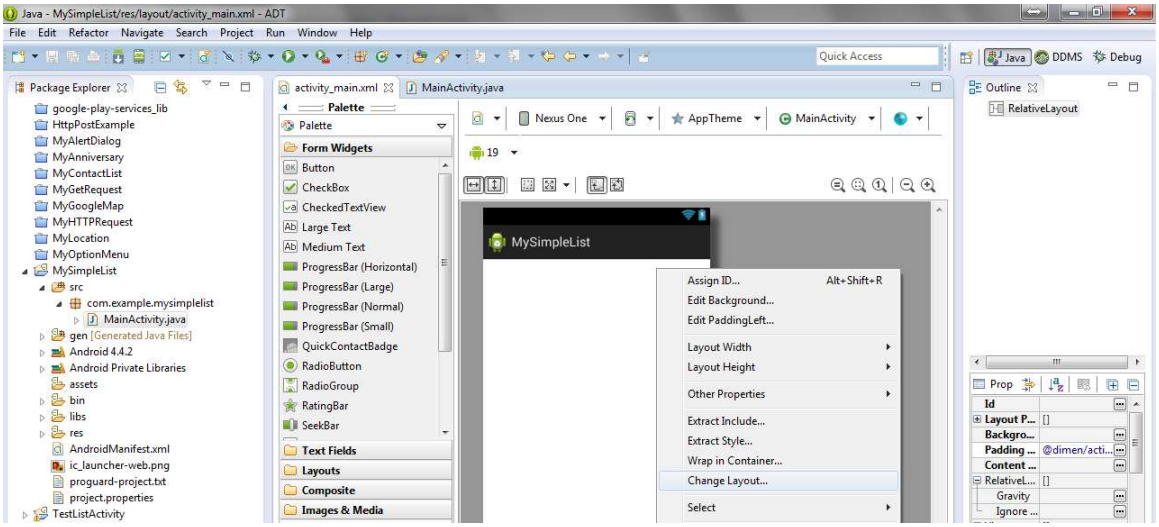
1.1 Simple List using simple_list_item_1

1. Create the Android application with the following attributes.
   - Application Name: **MySimpleList**
   - Project Name: **MySimpleList**
   - Package Name: **com.example.mysimplelist**

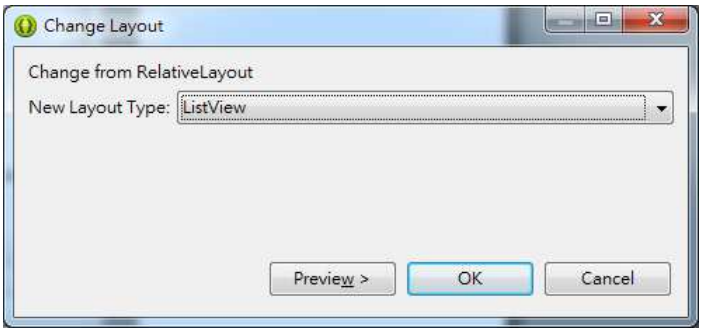
2. Remove the default text view “Hello World”



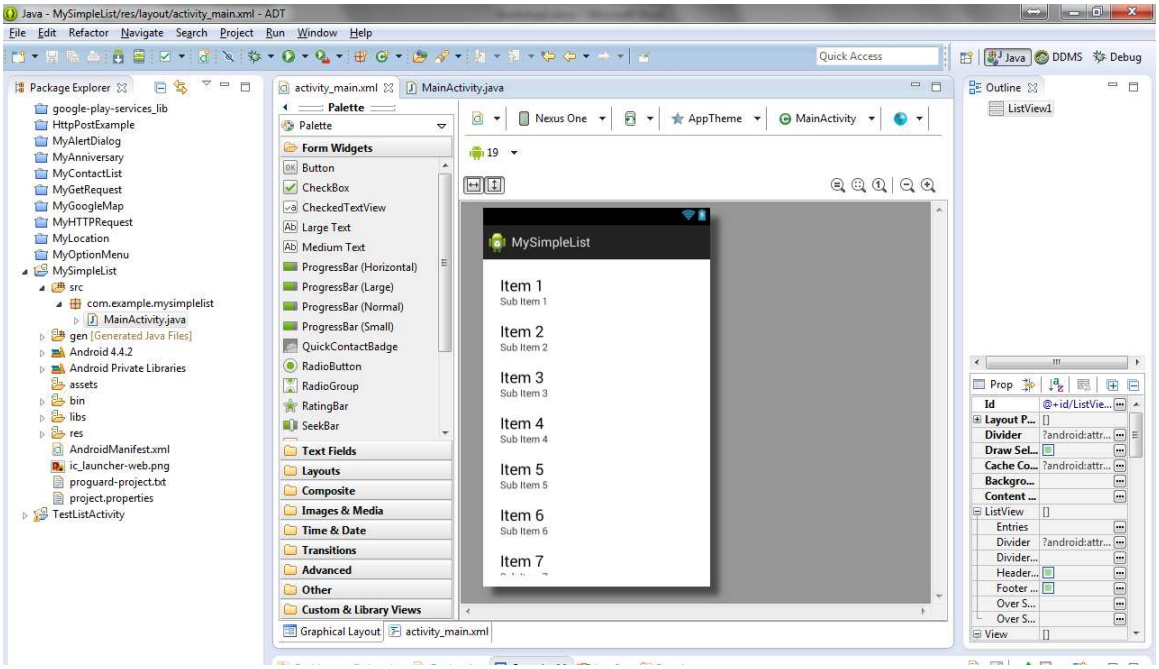
3. Right click the layout, and select **Change Layout**.



4. Change the layout to List View.



5. The layout will be changed to:



6. The XML for the layout look like:

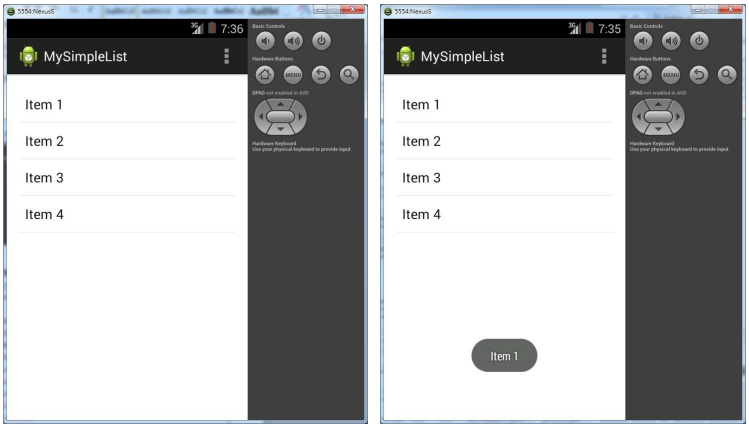
```
<ListView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/ListView1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >
</ListView>
```

7. Modify the source code for the file "**MainActivity.java**" as follow.

```
package com.example.mysimplelist;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
```

```
public class MainActivity extends Activity  {
    private ListView ListView1;
    private String ListItems[] = { "Item 1", "Item 2", "Item 3", "Item 4"};
    ArrayAdapter<String> adapter;

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

        // Define the visual style and set data for the list view
        ListView1 = (ListView)findViewById(R.id.ListView1);

        // Define a new Adapter
        adapter = new ArrayAdapter<String>
            (this,android.R.layout.simple_list_item_1, ListItems);

        // Assign adapter to ListView
        ListView1.setAdapter(adapter);

        // Register a callback to be invoked when an item in the list view clicked
        ListView1.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View view,
            int position, long id) {
                // Display a toast that show the content of the clicked list item
                Toast.makeText( getApplicationContext(),
                    ((TextView) view).getText(), Toast.LENGTH_SHORT).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.main, menu);
        return true;
    }
}
```

8. Save and execute the app, then select the list item to observe the result.



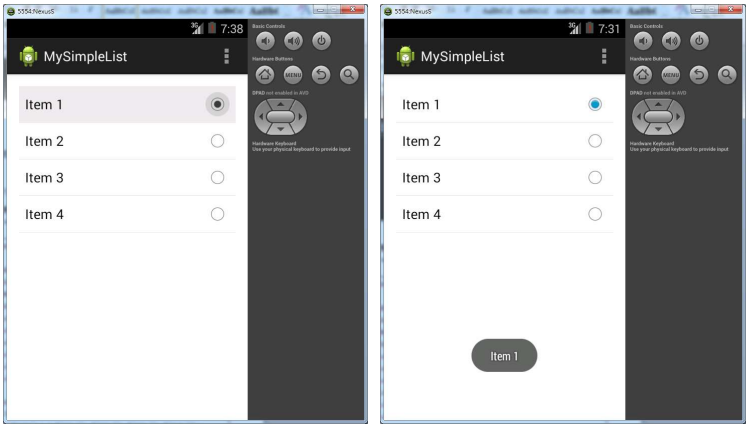
1.2 Simple List using simple_list_item_single_choice

1. Open the previous project and modify the source code for "**MainActivity.java**" as follow:

```
package com.example.mysimplelist;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity  {
    private ListView ListView1;
    private String ListItems[] = { "Item 1", "Item 2", "Item 3", "Item 4"};
    ArrayAdapter<String> adapter;

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

        // Define the visual style and set data for the list view
        ListView1 = (ListView)findViewById(R.id.ListView1);
```

```
        // Define a new Adapter
        adapter = new ArrayAdapter<String>
             (this,android.R.layout.simple_list_item_single_choice, ListItems);

        // Define the selection mode – Single choice
        ListView1.setChoiceMode(ListView.CHOICE_MODE_SINGLE);

        // Assign adapter to ListView
        ListView1.setAdapter(adapter);

        // Register a callback to be invoked when an item in the list view clicked
        ListView1.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View view,
            int position, long id) {
                 // Display a toast that show the content of the clicked list item
                Toast.makeText( getApplicationContext(),
                     ((TextView) view).getText(), Toast.LENGTH_SHORT).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.main, menu);
        return true;
    }
}
```

2. Save and execute the app, then select the list item to observe the result.



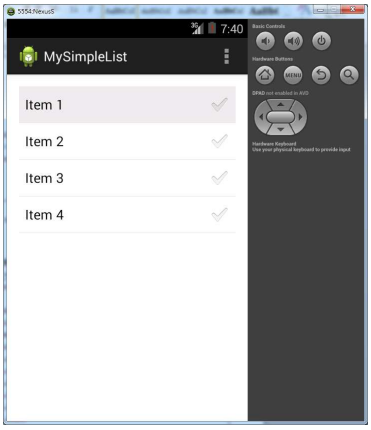
1.3 Simple List using simple_list_item_checked

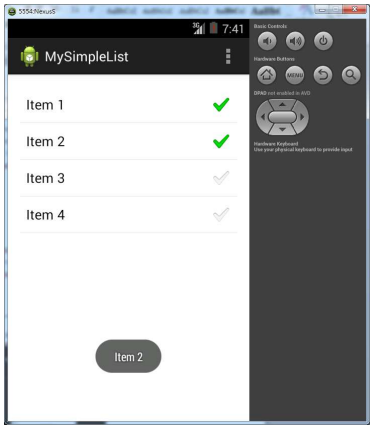
1. Open the previous project and modify the source code for "**MainActivity.java**" as follow:

```
package com.example.mysimplelist;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity  {
    private ListView ListView1;
    private String ListItems[] = { "Item 1", "Item 2", "Item 3", "Item 4"};
    ArrayAdapter<String> adapter;

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

        // Define the visual style and set data for the list view
        ListView1 = (ListView)findViewById(R.id.ListView1);

        // Define a new Adapter
        adapter = new ArrayAdapter<String>
            (this,android.R.layout.simple_list_item_checked, ListItems);

        // Define the selection mode – Multiple choice
        ListView1.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);

        // Assign adapter to ListView
        ListView1.setAdapter(adapter);

        // Register a callback to be invoked when an item in the list view clicked
```

```
        ListView1.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View view,
            int position, long id) {
                // Display a toast that show the content of the clicked list item
                Toast.makeText( getApplicationContext(),
                    ((TextView) view).getText(), Toast.LENGTH_SHORT).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.main, menu);
        return true;
    }
}
```

2. Save and execute the app, then select the list item to observe the result.





1.4 Simple List using simple_list_item_multiple_choice

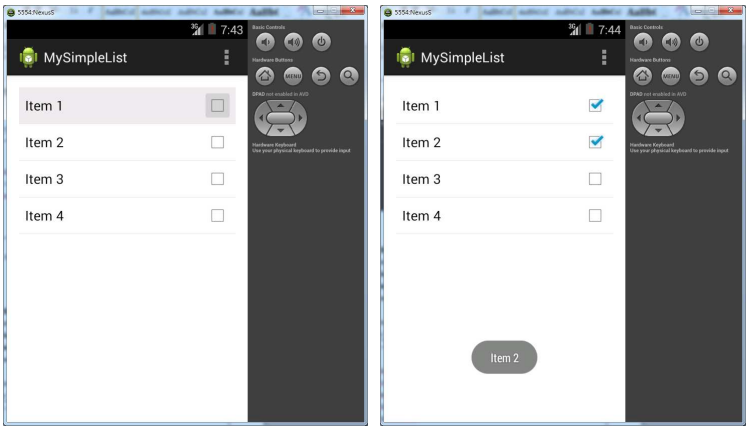
1. Open the previous project and modify the source code for "**MainActivity.java**" as follow.

```
package com.example.mysimplelist;


import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.AdapterView;
```

```
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity  {
    private ListView ListView1;
    private String ListItems[] = { "Item 1", "Item 2", "Item 3", "Item 4"};
    ArrayAdapter<String> adapter;

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

        // Define the visual style and set data for the list view
        ListView1 = (ListView)findViewById(R.id.ListView1);

        // Define a new Adapter
        adapter = new ArrayAdapter<String>
            (this,android.R.layout.simple_list_item_multiple_choice, ListItems);

        // Define the selection mode – Multiple choice
        ListView1.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);

        // Assign adapter to ListView
        ListView1.setAdapter(adapter);

        // Register a callback to be invoked when an item in the list view clicked
        ListView1.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View view,
            int position, long id) {
                // Display a toast that show the content of the clicked list item
                Toast.makeText( getApplicationContext(),
                    ((TextView) view).getText(), Toast.LENGTH_SHORT).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.main, menu);
        return true;
    }
}
```

2. Save and execute the app, then select the list item to observe the result.



2. Advanced List Technique

2.1 Create List using ListActivity

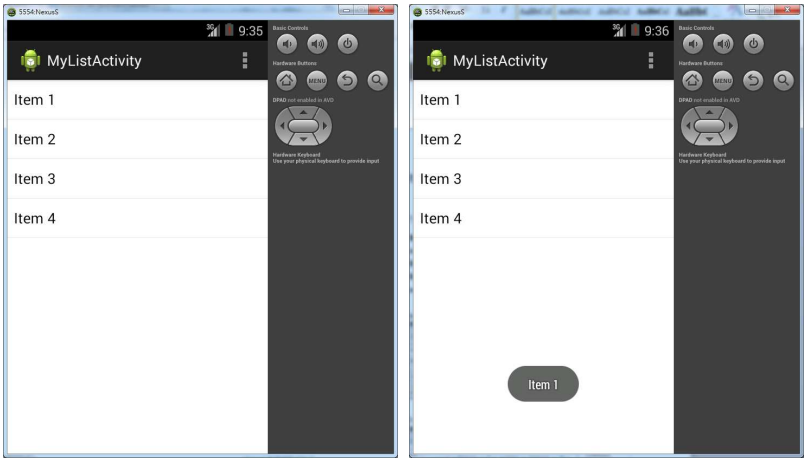
1. Create the Android application with the following attributes.
   - Application Name: **MyListActivity**
   - Project Name: **MyListActivity**
   - Package Name: **com.example.mylistactivity**

2. Modify the source code for "**MainActivity.java**" as follow.

```
package com.example.mylistactivity;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.app.ListActivity;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends ListActivity  {
    private String ListItems[] = { "Item 1", "Item 2", "Item 3", "Item 4"};
    private ArrayAdapter<String> adapter;

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

        // Define a new Adapter
        adapter = new ArrayAdapter<String>
            (this, android.R.layout.simple_list_item_1, ListItems);

        // Assign adapter to ListView
        setListAdapter(adapter);
    }
```

```
    // Handle Item click event
    protected void onListItemClick(ListView list, View view, int position, long id){
        Toast.makeText( getApplicationContext(),
            ((TextView) view).getText(), Toast.LENGTH_SHORT).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.main, menu);
        return true;
    }
}
```

3. Save and execute the app, then select the list item to observe the result.



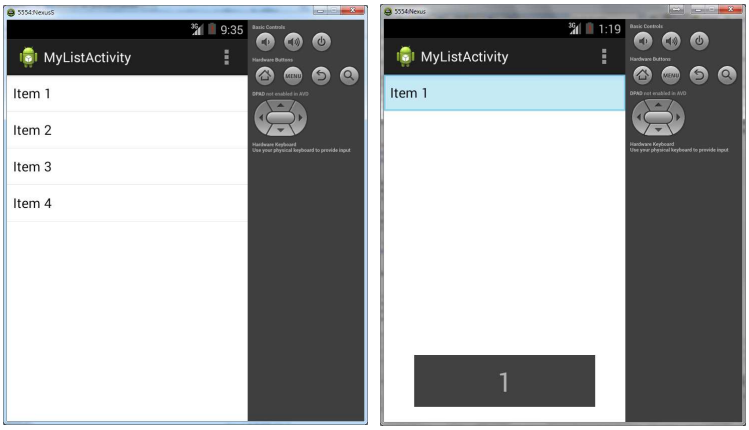
2.2 Switch on Text Filter

1. Open the previous project and modify the source code for "**MainActivity.java**" as follow.

```
package com.example.mylistactivity;


import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.app.ListActivity;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
```

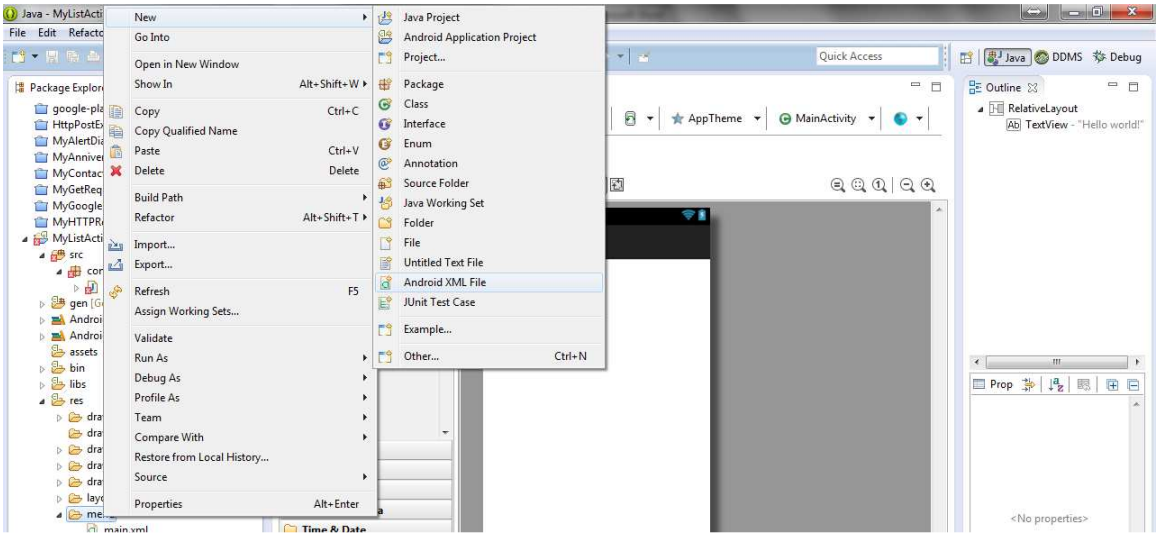
```
import android.widget.Toast;

public class MainActivity extends ListActivity  {
     private String ListItems[] = { "Item 1", "Item 2", "Item 3", "Item 4"};
     private ArrayAdapter<String> adapter;

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

          // Define a new Adapter
          adapter = new ArrayAdapter<String>
               (this, android.R.layout.simple_list_item_1, ListItems);

          // Assign adapter to ListView
          setListAdapter(adapter);

          // Enables or disables the type filter window
          getListView().setTextFilterEnabled(true);
     }

     // Handle Item click event
     protected void onListItemClick(ListView list, View view, int position, long id){
          Toast.makeText( getApplicationContext(),
               ((TextView) view).getText(), Toast.LENGTH_SHORT).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.main, menu);
          return true;
     }
}
```

2. Save and execute the app, then press “1” to observe the result.

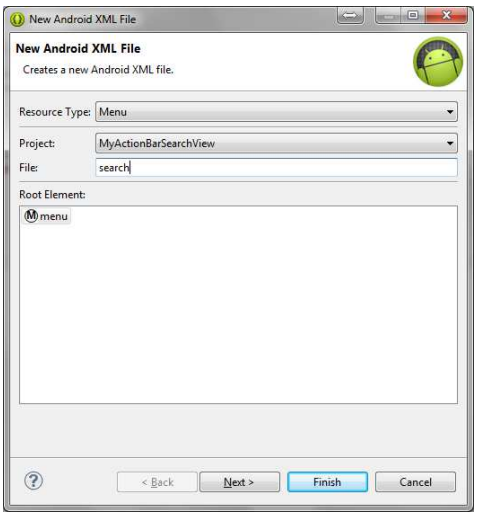


2.3 Enable Search View

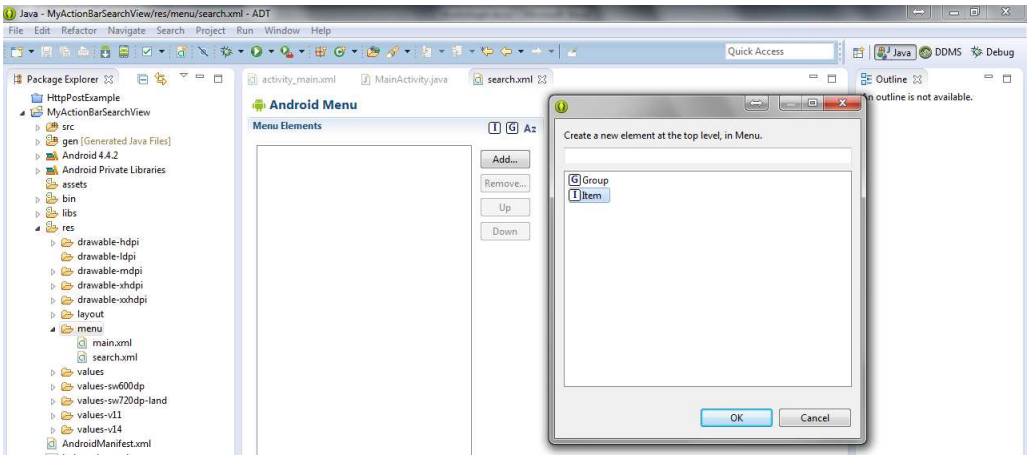
1. Open the preview project and select the "**res/menu**" folder. Then right click and select **New ➔ Android XML File**.



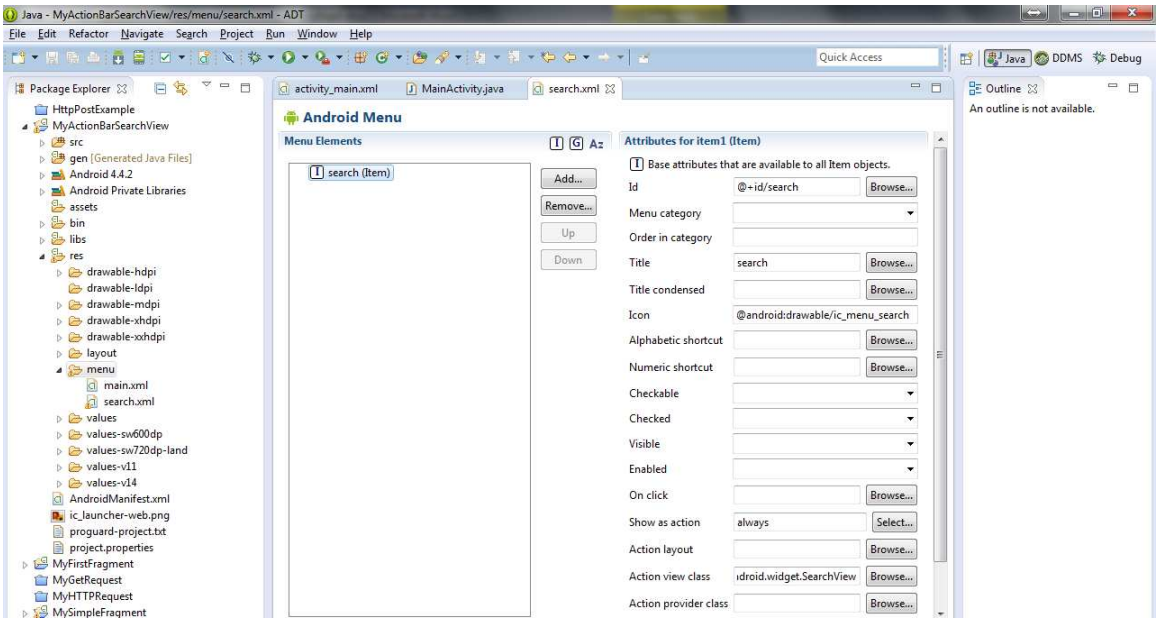
2. Set the File as “**search**” and press **[Finish]** button.



3. In order to create new menu item, press the **[Add]** button. Then select "**Item**" for "**Create a new element at the top level, in Menu**", press **[OK]** button to confirm.



4. Set the Id as "**@+id/search**", Icon as "**@android:drawable/ic_menu_search**", Title as "**search**", Show as action as "**always**" and Action view class as "**android.widget.SearchView**".



5. The XML code for the search view will look like:

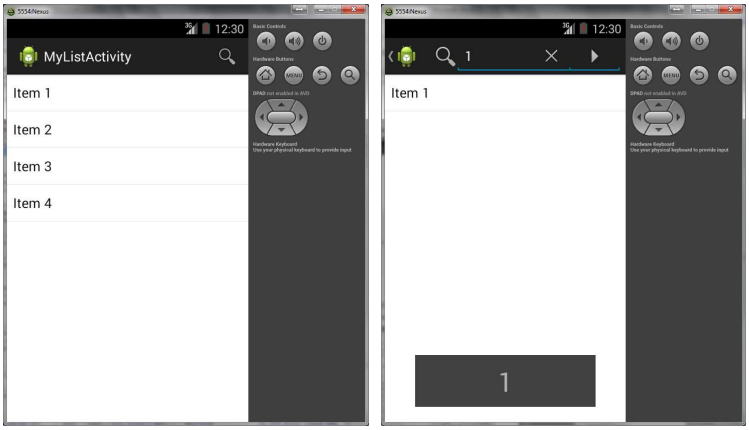
```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <item android:id="@+id/search"
          android:title="search"
          android:actionViewClass="android.widget.SearchView"
          android:showAsAction="always"
          android:icon="@android:drawable/ic_menu_search"/>
</menu>
```

6. Modify the source code for "**MainActivity.java**" as follow:

```
package com.example.mylistactivity;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.app.ListActivity;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.SearchView;
import android.view.MenuItem;

public class MainActivity extends ListActivity
                          implements SearchView.OnQueryTextListener {
    private String ListItems[] = { "Item 1", "Item 2", "Item 3", "Item 4"};
    private ArrayAdapter<String> adapter;
    private SearchView mSearchView;

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

         // Define a new Adapter
         adapter = new ArrayAdapter<String>
              (this, android.R.layout.simple_list_item_1, ListItems);

         // Assign adapter to ListView
         setListAdapter(adapter);

         // Enables or disables the type filter window
         getListView().setTextFilterEnabled(true);
    }

    private void setupSearchView(MenuItem searchItem) {
        // Allow the search item to be expanded and collapsible
```

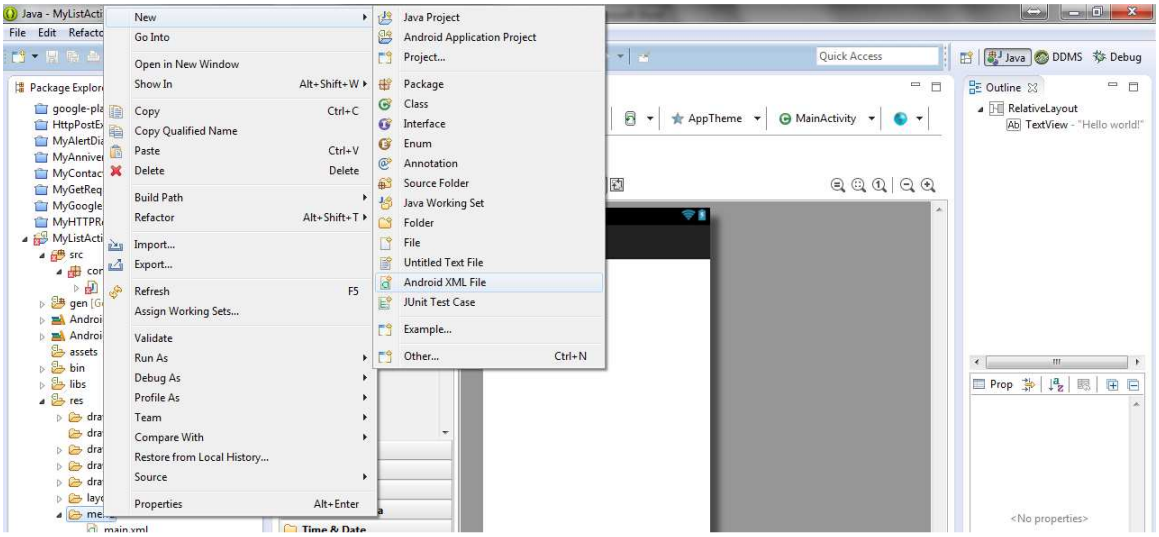
```
        searchItem.setShowAsActionFlags(MenuItem.SHOW_AS_ACTION_IF_ROOM |
                                        MenuItem.SHOW_AS_ACTION_COLLAPSE_ACTION_VIEW);

        // Define the search view attributes
        mSearchView.setIconifiedByDefault(false);
        mSearchView.setOnQueryTextListener(this);
        mSearchView.setSubmitButtonEnabled(true);
        mSearchView.setQueryHint("Search Here");
    }

    public boolean onQueryTextChange(String newText) {
        getListView().setFilterText(newText.toString());
        return true;
    }

    public boolean onQueryTextSubmit(String query) {
        return false;
    }

    // Handle Item click event
    protected void onListItemClick(ListView list, View view, int position, long id){
        Toast.makeText( getApplicationContext(),
             ((TextView) view).getText(), Toast.LENGTH_SHORT).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.search, menu);;

        // Correlate the menu item and search view
        MenuItem searchItem = menu.findItem(R.id.search);
        mSearchView = (SearchView) searchItem.getActionView();
        setupSearchView(searchItem);

//      getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

7. Save and execute the app, then press “1” in search view to observe the result:

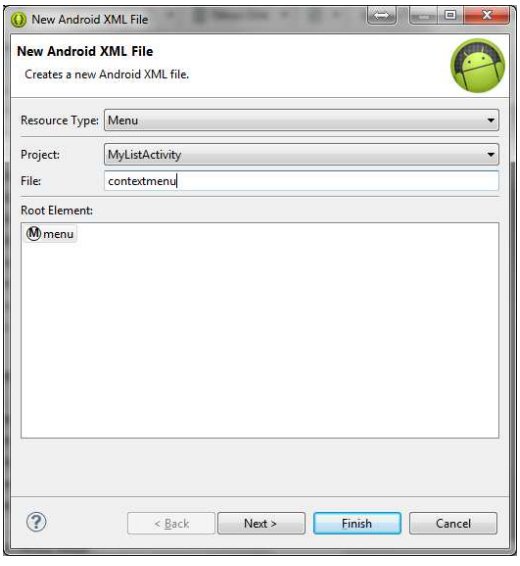


2.4 Attach a Context Menu

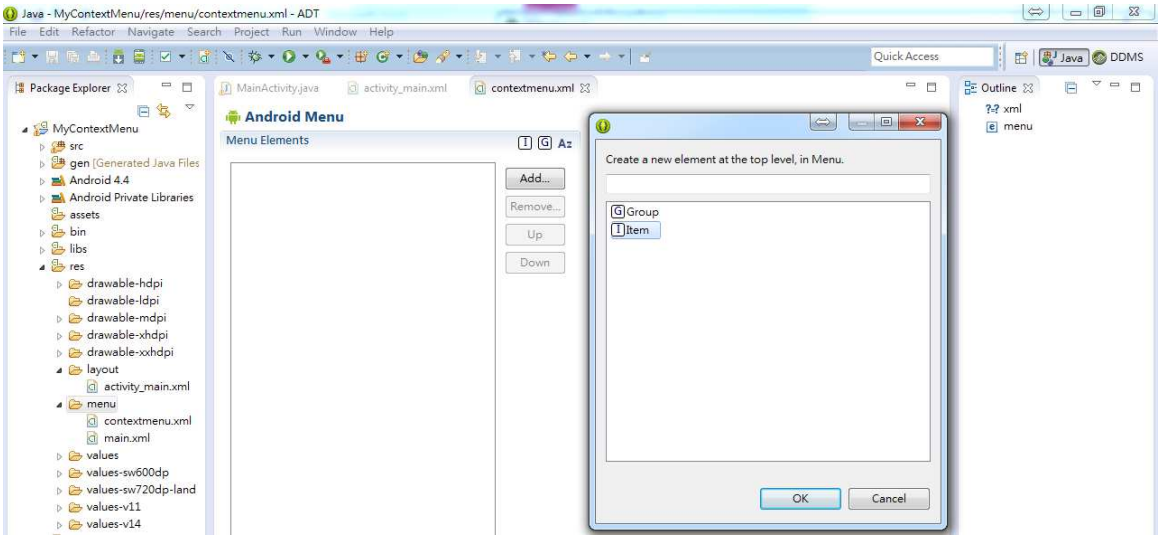
1. Open the preview project and select the "**res/menu**" folder. Then right click and select **New ➔ Android XML File**.



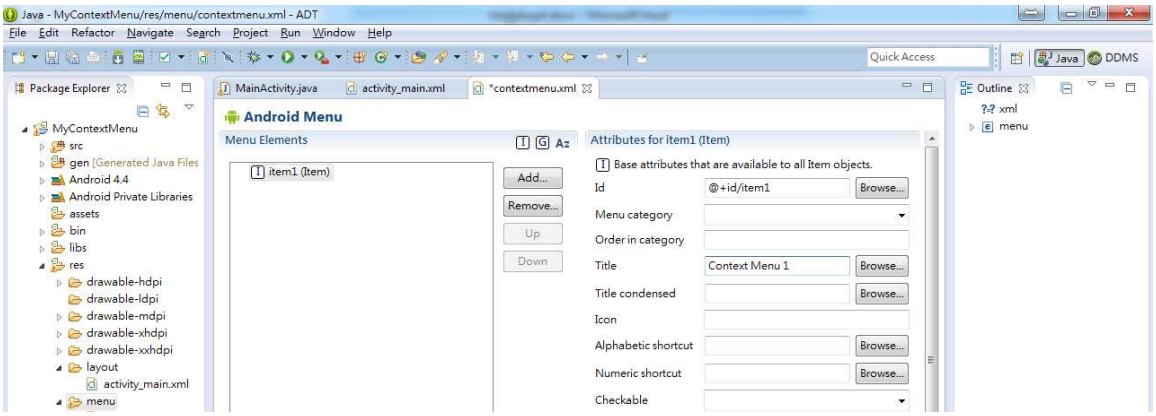
2. Set the File as “**contextmenu**” and press **[Finish]** button.



3. In order to create new context menu item, press the **[Add]** button. Then select "**Item**" in the dialog, press **[OK]** button to confirm.



4. Set the "**Title**" to "**Context Menu 1**" and save the menu. Repeat the steps to create "**Context Menu 2**".



5. The XML code for the content menu will look like:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <item android:id="@+id/item1" android:title="Context Menu 1"></item>
    <item android:id="@+id/item2" android:title="Context Menu 2"></item>
</menu>
```

6. Modify the source code for "**MainActivity.java**" as follow.

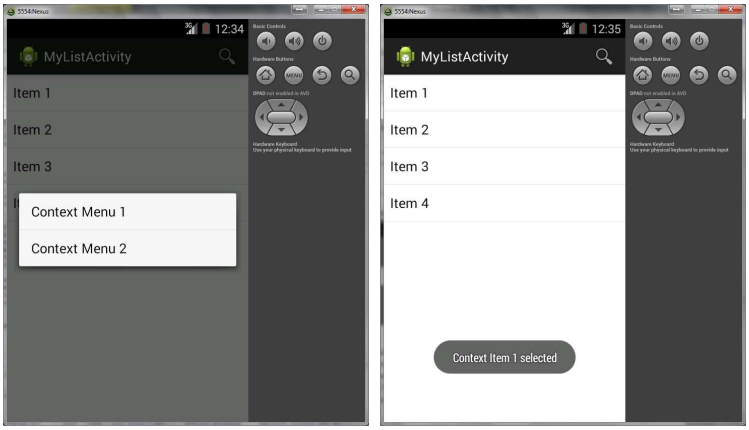
```
package com.example.mylistactivity;


import android.os.Bundle;
import android.app.Activity;
```

```
import android.view.Menu;
import android.app.ListActivity;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.SearchView;
import android.view.MenuItem;
import android.view.ContextMenu;
import android.view.ContextMenu.ContextMenuInfo;

public class MainActivity extends ListActivity
                          implements SearchView.OnQueryTextListener {
    private String ListItems[] = { "Item 1", "Item 2", "Item 3", "Item 4"};
    private ArrayAdapter<String> adapter;
    private SearchView mSearchView;

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

        // Define a new Adapter
        adapter = new ArrayAdapter<String>
            (this, android.R.layout.simple_list_item_1, ListItems);

        // Assign adapter to ListView
        setListAdapter(adapter);

        // Enables or disables the type filter window
        getListView().setTextFilterEnabled(true);
    }

    private void setupSearchView(MenuItem searchItem) {
       // Allow the search item to be expanded and collapsible
        searchItem.setShowAsActionFlags(MenuItem.SHOW_AS_ACTION_IF_ROOM |
                                        MenuItem.SHOW_AS_ACTION_COLLAPSE_ACTION_VIEW);
```

```
        // Define the search view attributes
        mSearchView.setIconifiedByDefault(false);
        mSearchView.setOnQueryTextListener(this);
        mSearchView.setSubmitButtonEnabled(true);
        mSearchView.setQueryHint("Search Here");
    }

    public boolean onQueryTextChange(String newText) {
        getListView().setFilterText(newText.toString());
        return true;
    }

    public boolean onQueryTextSubmit(String query) {
        return false;
    }

    // Handle Item click event
    protected void onListItemClick(ListView list, View view, int position, long id){
        Toast.makeText( getApplicationContext(),
            ((TextView) view).getText(), Toast.LENGTH_SHORT).show();

        // Register the content menu for the List item
        registerForContextMenu(view);
    }

    @Override
    public void onCreateContextMenu(ContextMenu contextMenu, View view,
                ContextMenuInfo menuInfo) {
        // Inflate the context menu
        super.onCreateContextMenu(contextMenu, view, menuInfo);
        getMenuInflater().inflate(R.menu.contextmenu, contextMenu);
    }

    public boolean onContextItemSelected (MenuItem item) {
        switch(item.getItemId()) {
            case R.id.item1:
                Toast.makeText(this, "Context Item 1 selected",
                            Toast.LENGTH_LONG).show();
                return true;
```

```
            case R.id.item2:
                Toast.makeText(this, "Context Item 2 selected",
                        Toast.LENGTH_LONG).show();
                return true;

            default:
                return super.onContextItemSelected(item);
        }
    }

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

        // Correlate the menu item and search view
        MenuItem searchItem = menu.findItem(R.id.search);
        mSearchView = (SearchView) searchItem.getActionView();
        setupSearchView(searchItem);

//      getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

7. Save and execute the app, then long click the list item to call up the context menu.



2.5 Simple List using simple_list_item_2

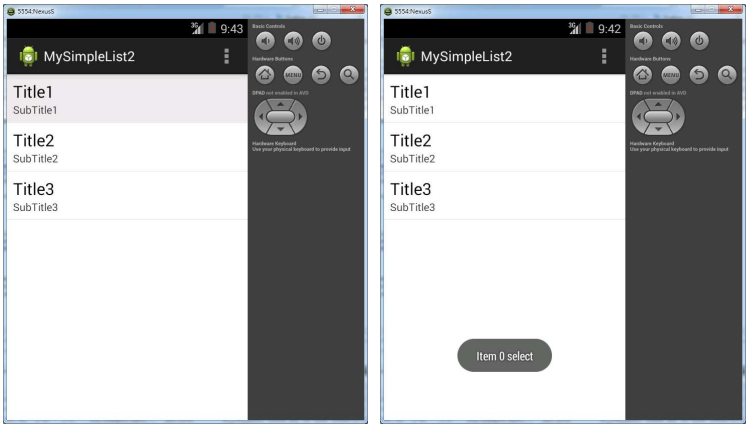
1. Create the Android application with the following attributes.
   - Application Name: **MySimpleList2**
   - Project Name: **MySimpleList2**
   - Package Name: **com.example.mysimplelist2**

2. Modify the source code for the file "**MainActivity.java**" as follow.

```
package com.example.mysimplelist2;


import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.app.ListActivity;
import java.util.ArrayList;
import java.util.HashMap;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.Toast;

public class MainActivity extends ListActivity {
    final String ID_TITLE = "TITLE", ID_SUBTITLE = "SUBTITLE";
    private SimpleAdapter adapter;

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

        // Define the array for title and subtitle
        String[] titles = new String[]{ "Title1" , "Title2", "Title3" };
        String[] subtitles = new String[]{ "SubTitle1" , "SubTitle2", "SubTitle3" };

        // Create the array for title and subtitle
        ArrayList<HashMap<String,String>> myListData = new
                ArrayList<HashMap<String,String>>();

        // Append the array item to the list
        for (int i=0; i<titles.length; ++i) {
```

```
            HashMap<String, String> item = new HashMap<String, String>();
            item.put(ID_TITLE, titles[i]);
            item.put(ID_SUBTITLE, subtitles[i]);
            myListData.add(item);
        }

        // Define a simple Adapter
        adapter = new SimpleAdapter( this, myListData,
            android.R.layout.simple_list_item_2,
            new String[] { ID_TITLE, ID_SUBTITLE },
            new int[] { android.R.id.text1, android.R.id.text2 });

        // Assign adapter to ListView
        setListAdapter(adapter);
    }

    // Handle Item click event
    protected void onListItemClick(ListView list, View view, int position, long id){
        Toast.makeText(getApplicationContext(), "Item " + position + " select",
                Toast.LENGTH_SHORT).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.main, menu);
        return true;
    }
}
```

3. Execute the app to see the lists.

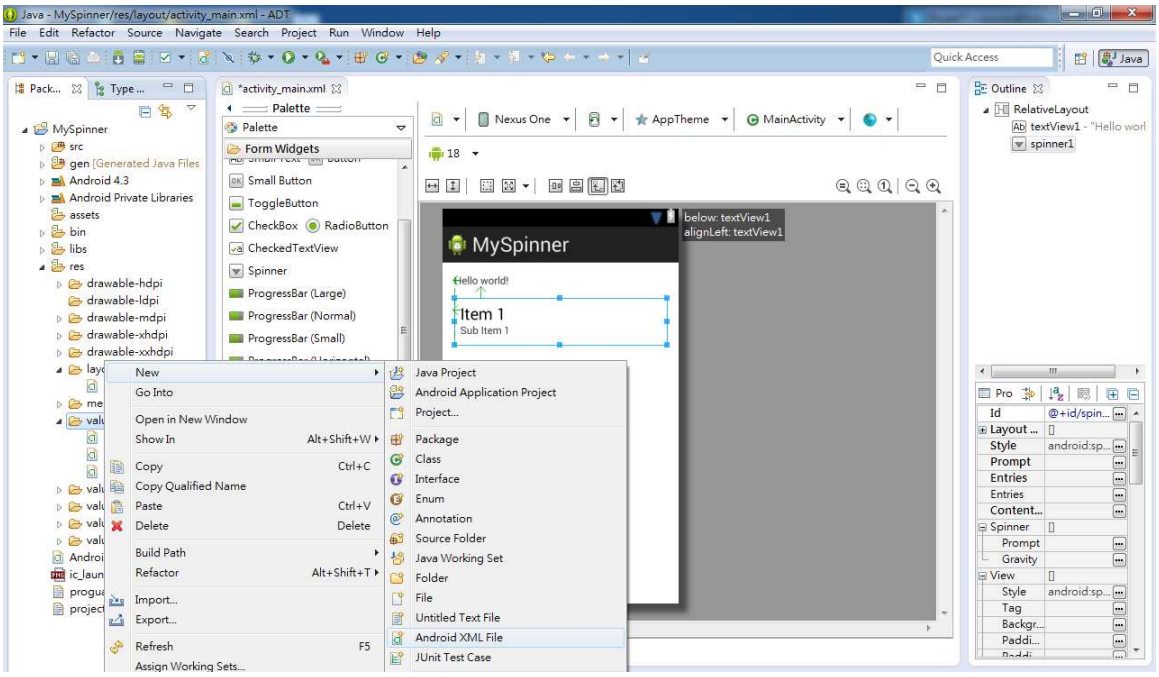


3. Advanced UI Components

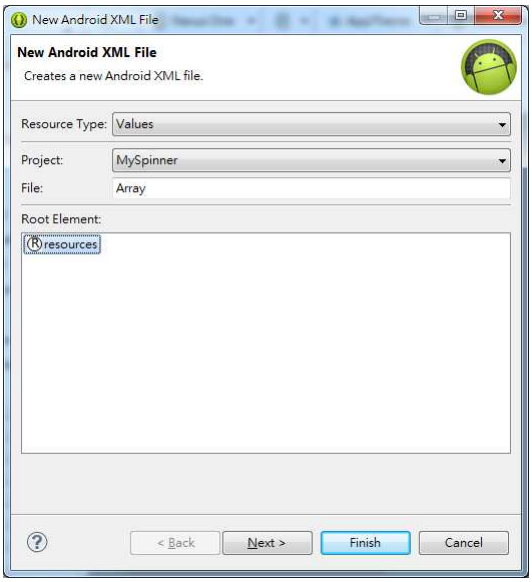
3.1 Spinner

1. Create the Android application with the following attributes.
   - Application Name: **MySpinner**
   - Project Name: **MySpinner**
   - Package Name: **com.example.myspinner**

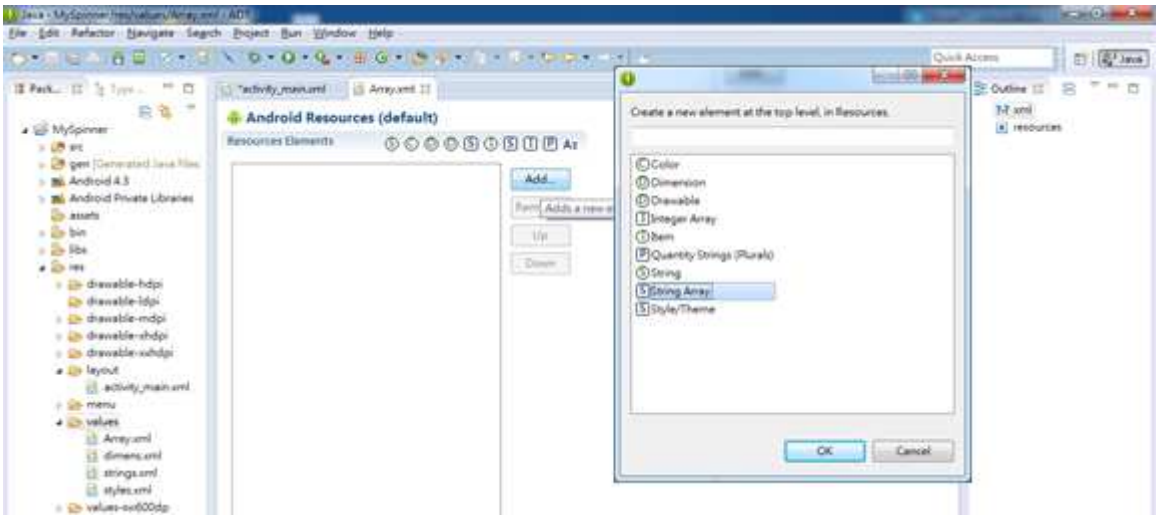
2. Drag a spinner into the layout. Right click the folder "**res\value**", then select **New ➔ Android XML File**.



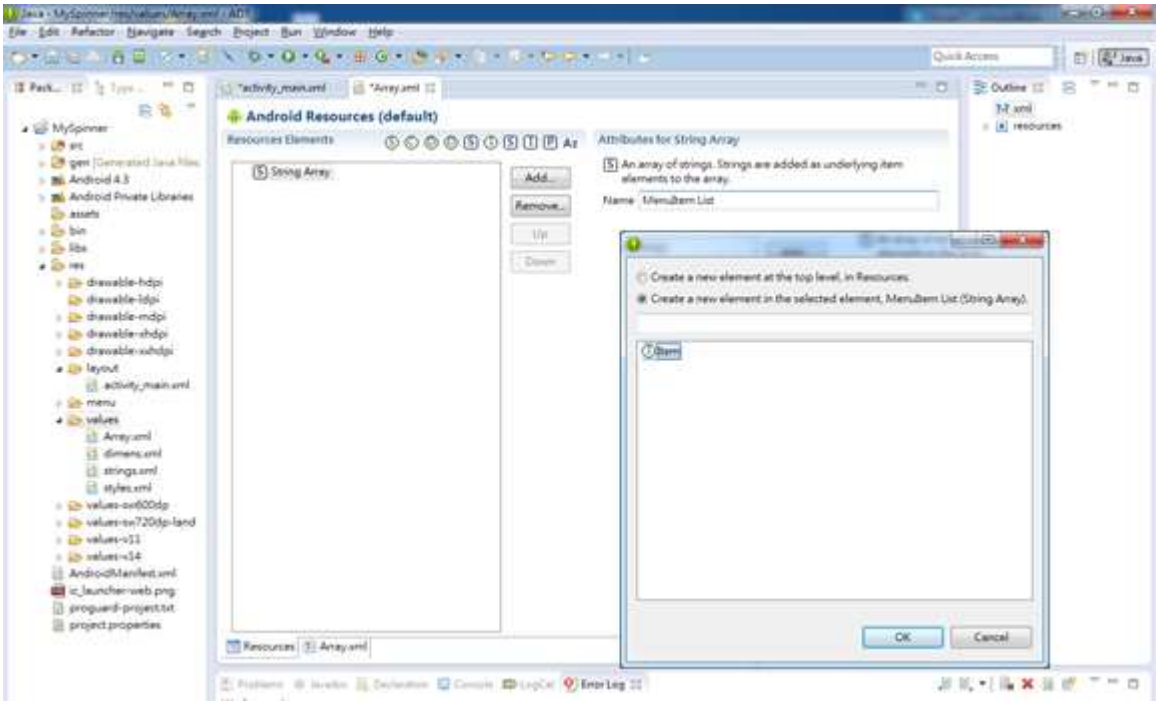
3. Input "**Array**" as the file name and click "**Finish**".



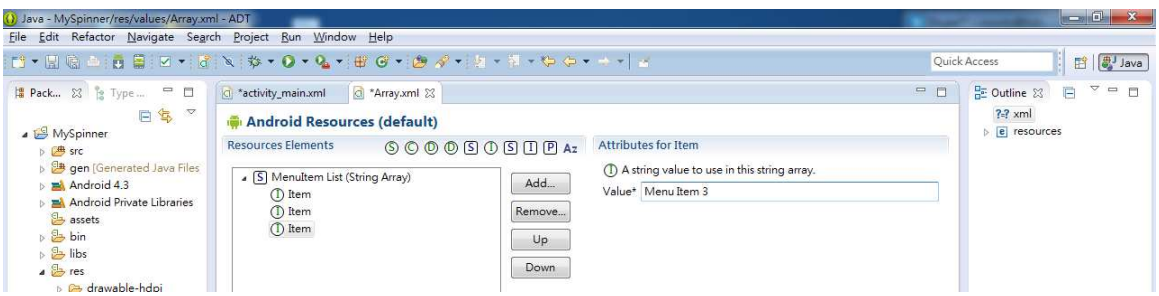
4. Click the "**Add**" button, and then select "**String Array**" and click "**OK**"



5. Input the name "**MenuItemList**" for the String Array, and then click "**Add**" button again. Select "**Item**" and press "**OK**".



6. Input the value "**Menu Item 1**". Then repeat the above step to create "**Menu Item 2**" and "**Menu Item 3**".



7. The XML source code should look like:

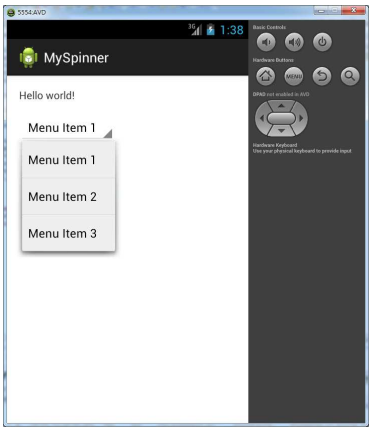
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="MenuItemList">
        <item >Menu Item 1</item>
        <item >Menu Item 2</item>
        <item >Menu Item 3</item>
    </string-array>
</resources>
```

8. Modify the source code for the file "**MainActivity.java**" as follow.

```
package com.example.myspinner;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemSelectedListener;
import android.widget.ArrayAdapter;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Finds a view that was identified by the ID attribute from the XML
        Spinner spinner1 = (Spinner) findViewById (R.id.spinner1);

        // Create an ArrayAdapter using the string array and a default spinner layout
        ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(
            this, R.array.MenuItemList, android.R.layout.simple_spinner_item);

        // Specify the layout to use when the list of choices appears
        adapter.setDropDownViewResource(
```

```
            android.R.layout.simple_spinner_dropdown_item);

        // Apply the adapter to the spinner
        spinner1.setAdapter(adapter);

        // Create an listener and handle the event
        spinner1.setOnItemSelectedListener(new OnItemSelectedListener() {
            public void onItemSelected(AdapterView<?> parent, View view,
                    int pos, long id) {
                Toast.makeText( getApplicationContext(),
                        ((TextView) view).getText(), Toast.LENGTH_SHORT).show();
            }

            @Override
            public void onNothingSelected(AdapterView<?> parent) {
                // Do something
            }
        });
    }

    @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;
    }
}
```

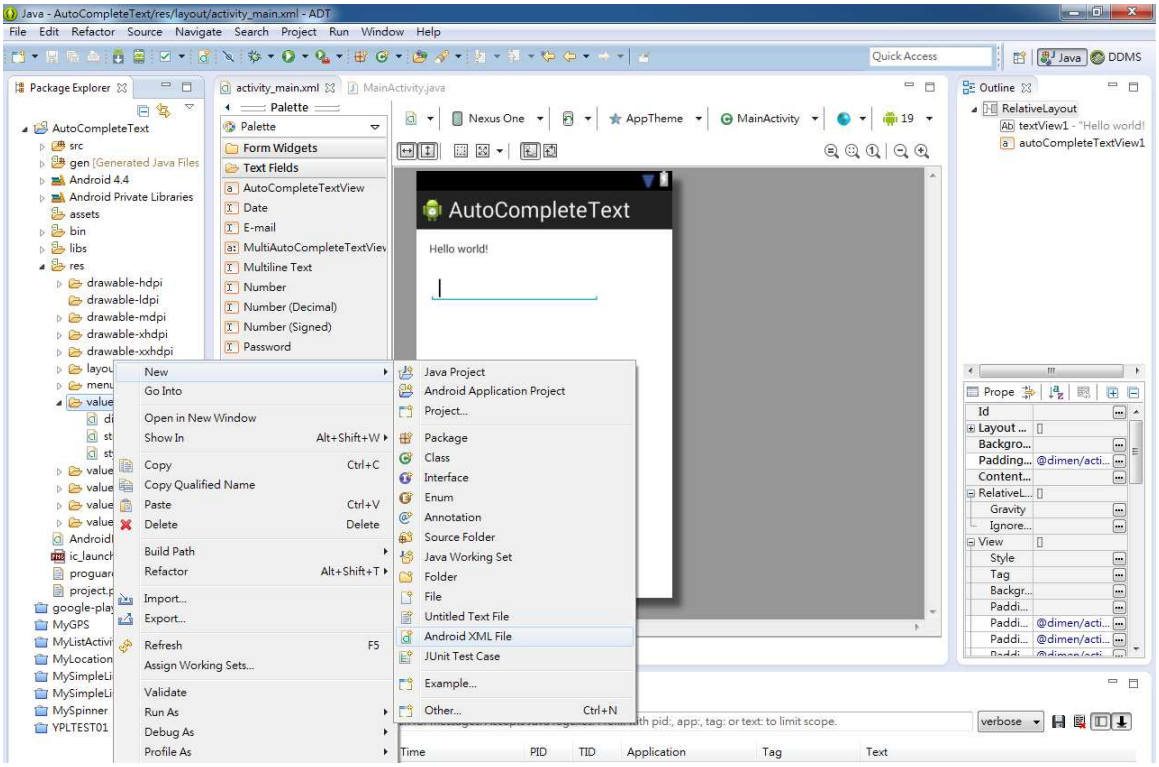
9. Execute the app, and press the spinner to display the menu.



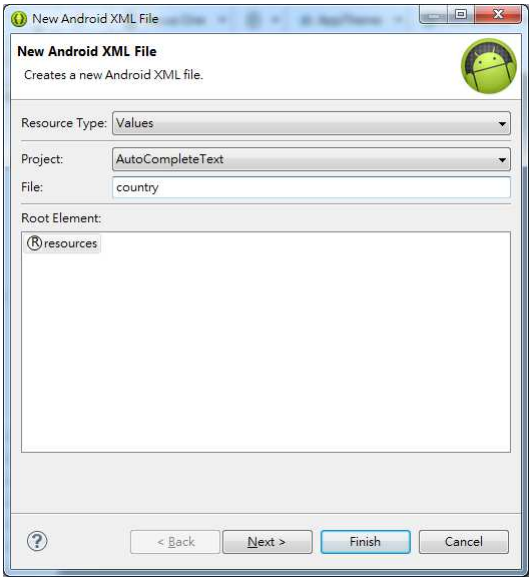
3.2 Auto Compete Text View

1. Create the Android application with the following attributes.
   - Application Name: **AutoCompleteText**
   - Project Name: **AutoCompleteText**
   - Package Name: **com.example.autocompletetext**

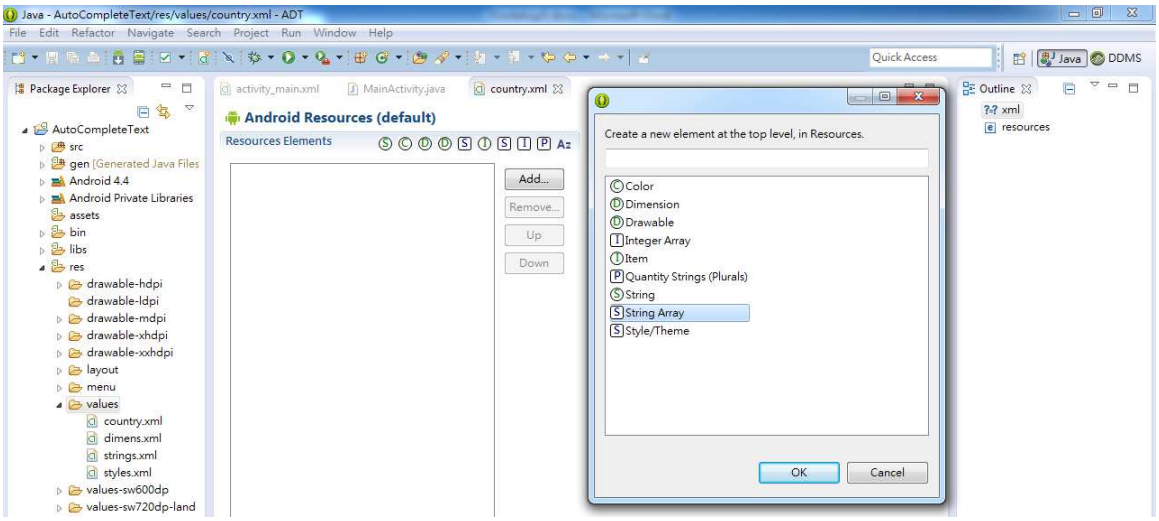
2. Add the AutoCompleteTextView to the layout, and remove the text content. Right click the folder "**res\value**", then select **New ➔ Android XML File**.



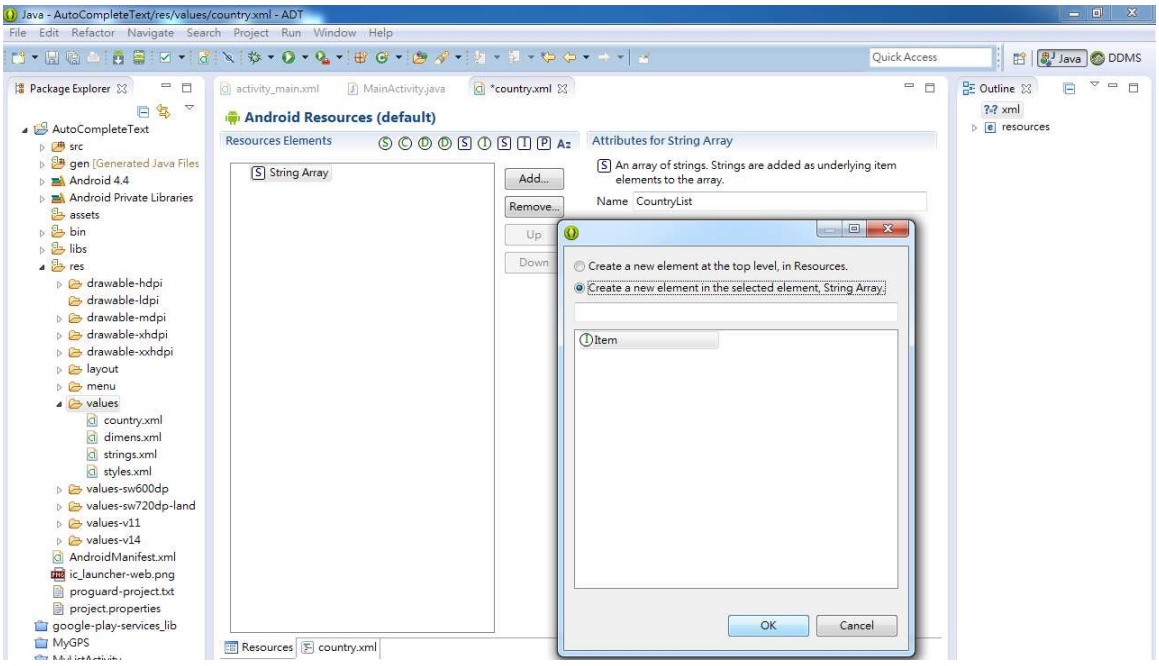
3. Input "**country**" as the file name and click "**Finish**".



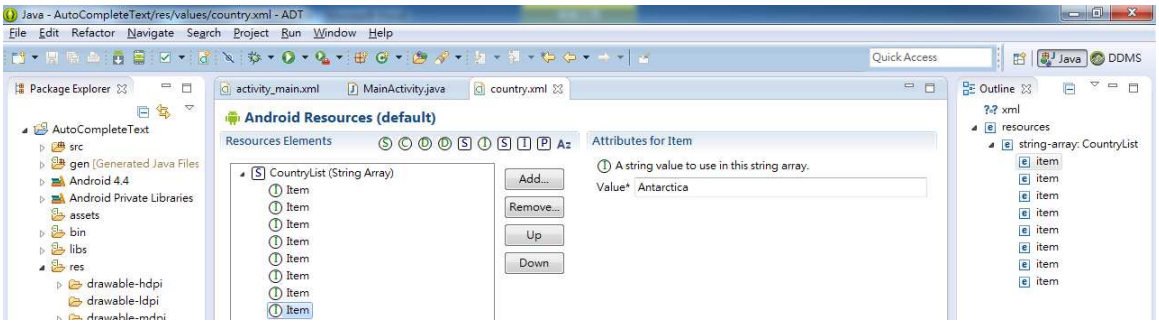
4. Click the "**Add**" button, and then select "**String Array**" and click "**OK**"



5. Input the name "**CountryList**" for the String Array, and then click "**Add**" button again. Select "**Item**" and press "**OK**".



6. Input the value "**Afghanistan**". Then repeat the above step to create "Afghanistan", "Albania", "Algeria", "American Samoa", "Andorra", "Angola", "Anguilla" and "Antarctica",



7. The XML source code should look like:

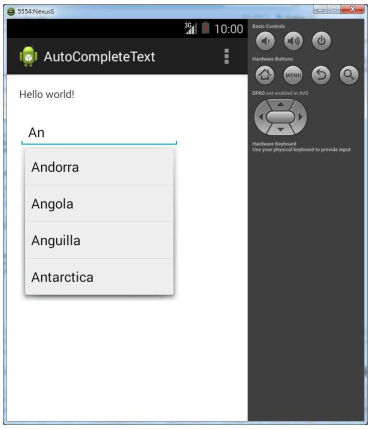
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="CountryList">
        <item>Afghanistan</item>
        <item>Albania</item>
        <item>Algeria</item>
        <item>American Samoa</item>
        <item>Andorra</item>
        <item>Angola</item>
        <item>Anguilla</item>
        <item>Antarctica</item>
    </string-array>
</resources>
```

8. Modify the source code for the file "**MainActivity.java**" as follow. Here, a new ArrayAdapter is initialized to bind each item in the country string array to a TextView that exists in the simple_list_item_1 layout.

```
package com.example.autocompletetext;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;

public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Get a reference to the AutoCompleteTextView in the layout
        AutoCompleteTextView textView = (AutoCompleteTextView)
            findViewById(R.id.autoCompleteTextView1);

        // Get the string array
        String[] countries = getResources().getStringArray(R.array.CountryList);
```

```
        // Create the adapter and set it to the AutoCompleteTextView
        ArrayAdapter<String> adapter = new ArrayAdapter<String> (this,
            android.R.layout.simple_dropdown_item_1line, countries);

        // Apply the adapter to the textview
        textView.setAdapter(adapter);
    }

    @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;
    }
}
```

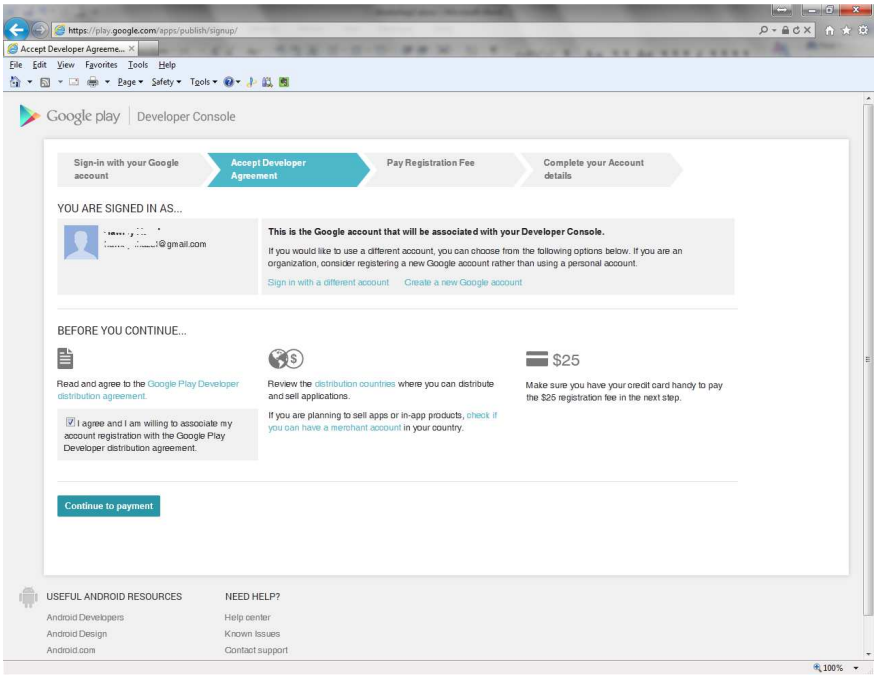
9. Execute the app and input the first several characters in the text box, the list will be displayed.



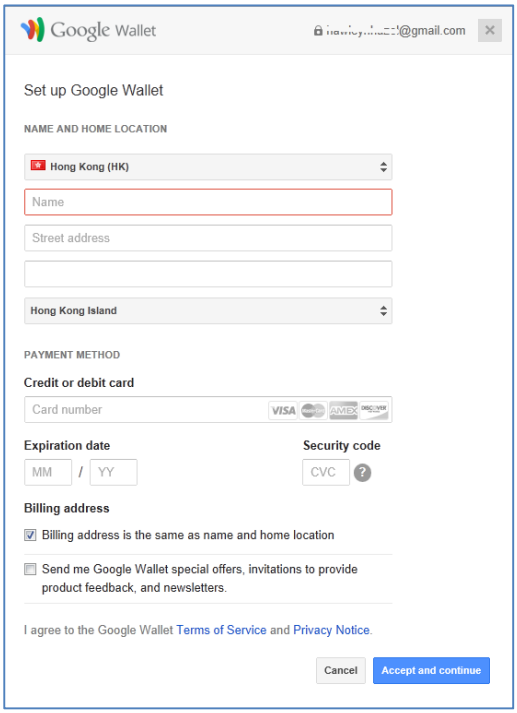
4. Deployment

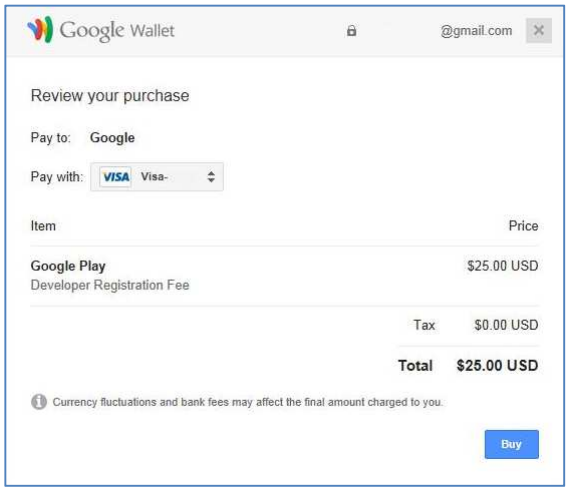
4.1 Register as Google Play Developer

1. Sign-in Google Play Developer Console (https://play.google.com/apps/publish) with your Google account. After read and agree to the Google Play Developer distribution agreement, press the "**Continue to payment**" button to process.

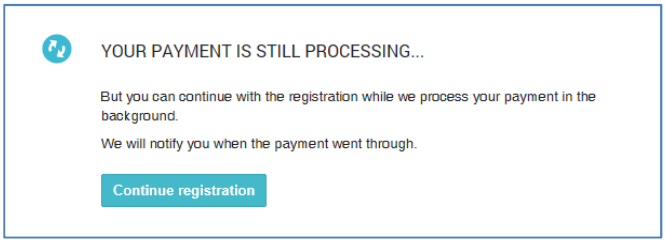


2. Signup the Google Wallet with your name, address and credit card. Then press "**Accept and Continue**" to proceed. Press "**Buy**" after your review the payment.

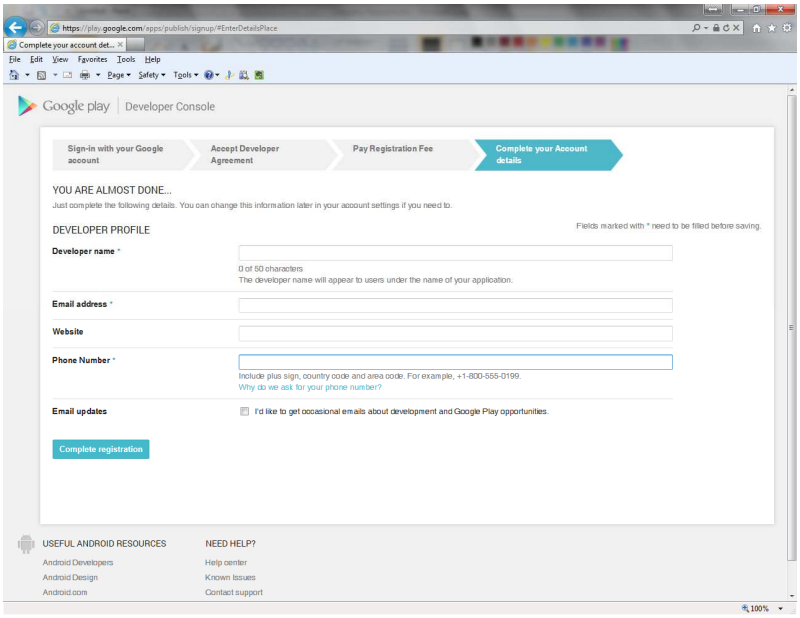




3. Your payment will be process, and US$25 will be deduced from your credit card. Press "**Continue registration**" to complete registration.



4. Fill up the developer name, contact email and mobile. Then press "**Complete registration**".



5. The developer registration is completed, and now you can publish your app.