

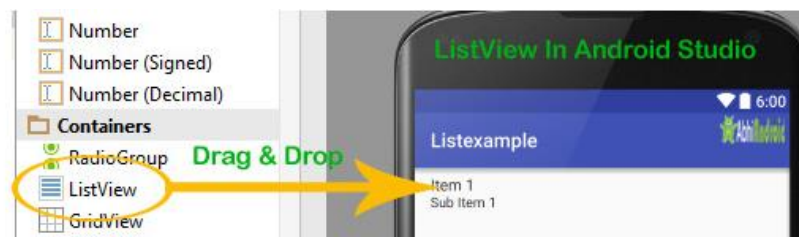
ListView Tutorial With Example In Android Studio

List of scrollable items can be displayed in Android using [ListView](#). It helps you to displaying the data in the form of a scrollable list. Users can then select any list item by clicking on it. [ListView](#) is default scrollable so we do not need to use scroll View or anything else with [ListView](#).

ListView is widely used in android applications. A very common example of ListView is your phone contact book, where you have a list of your contacts displayed in a ListView and if you click on it then user information is displayed.

Adapter: To fill the data in a ListView we simply use adapters. List items are automatically inserted to a list using an [Adapter](#) that pulls the content from a source such as an arraylist, array or database.

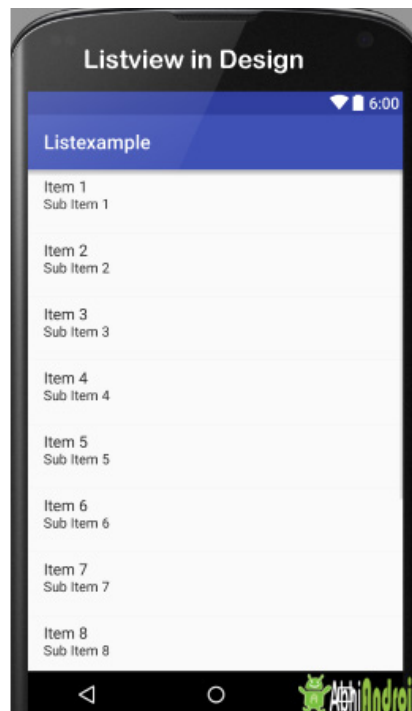
ListView in Android Studio: Listview is present inside Containers. From there you can drag and drop on virtual mobile screen to create it. Alternatively you can also [XML](#) code to create it.



Here is Android ListView XML Code:

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

Listview look in Design:



Attributes of ListView:

Lets see some different attributes of ListView which will be used while designing a custom list:

1. id: id is used to uniquely identify a ListView.

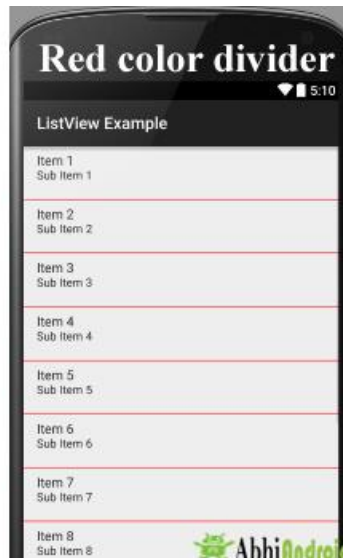
Below is the id attribute's example code with explanation included.

```
<!-- Id of a list view uniquely identify it-->
<ListView
    android:id="@+id/simpleListView"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
/>
```

2. divider: This is a drawable or color to draw between different list items.

Below is the divider example code with explanation included, where we draw red color divider between different views.

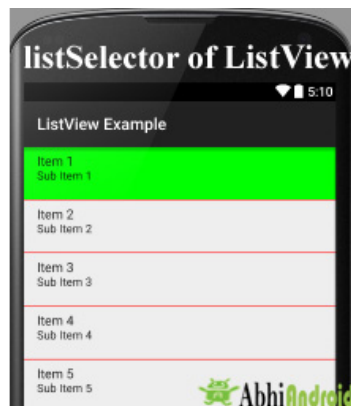
```
<!--Divider code in ListView-->
<ListView
    android:id="@+id/simpleListView"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:divider="#f00"
    android:dividerHeight="1dp"
/>
```



3. dividerHeight: This specifies the height of the divider between list items. This could be in dp(density pixel),sp(scale independent pixel) or px(pixel).

In above example of divider we also set the divider height 1dp between the list items. The height should be in dp,sp or px.

4. listSelector: listSelector property is used to set the selector of the listView. It is generally orange or Sky blue color mostly but you can also define your custom color or an image as a list selector as per your design.



Below is listSelector example code with explanation includes, where list selector color is green, when you select any list item then that item's background color is green .

```
<!-- List Selector Code in ListView -->
<ListView
    android:id="@+id/simpleListView"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:divider="#f00"
    android:dividerHeight="1dp"
    android:listSelector="#0f0"/> <!--list selector in green color-->
```

Adapters Use in ListView:

An [adapter](#) is a bridge between UI component and data source that helps us to fill data in UI component. It holds the data and send the data to [adapter](#) view then view can takes the data from the adapter view and shows the data on different views like as [list view](#), [grid view](#), [spinner](#) etc.

ListView is a subclass of AdapterView and it can be populated by binding to an Adapter, which retrieves the data from an external source and creates a View that represents each data entry.

In android commonly used adapters are:

1. Array Adapter
2. Base Adapter

Now we explain these two adapter in detail:

1.Array Adapter:

Whenever you have a list of single items which is backed by an array, you can use ArrayAdapter. For instance, list of phone contacts, countries or names.

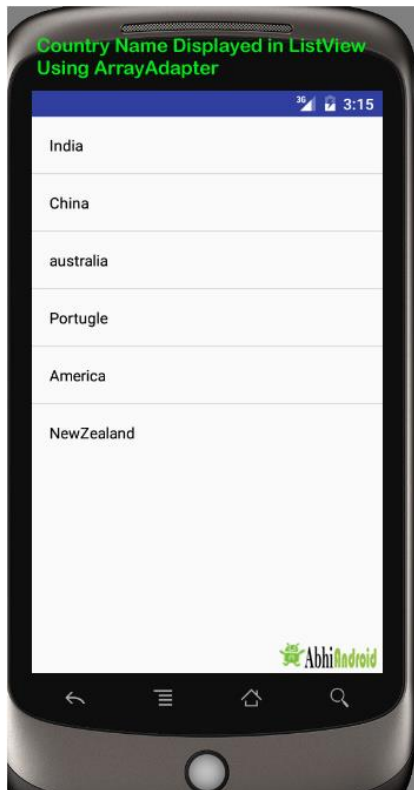
Important Note: By default, ArrayAdapter expects a Layout with a single [TextView](#), If you want to use more complex views means more customization in list items, please avoid ArrayAdapter and use custom adapters.

Below is Array Adapter code:

```
ArrayAdapter adapter = new  
ArrayAdapter<String>(this,R.layout.ListView,R.id.textView,StringArray);
```

Example of list view using Array Adapter:

In this example, we display a list of countries by using simple array adapter. Below is the final output we will create:



Step 1: [Create a new project](#) Listexample and activity Main Activity. Here we will create a ListView in LinearLayout. Below is the code of activity_main.xml or content_main.xml:

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

    <ListView
        android:id="@+id/simpleListView"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:divider="@color/material_blue_grey_800"
        android:dividerHeight="1dp" />
</LinearLayout>
```

Step 2: Create a new activity name Listview and below is the code of activity_listview.xml

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

    <TextView
        android:id="@+id/textView"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:padding="@dimen/activity_horizontal_margin"
        android:textColor="@color/black" />
</LinearLayout>
```

Step 3: Now in this final step we will use ArrayAdapter to display the country names in UI. Below is the code of MainActivity.java

```
package abhiandroid.com.listexample;

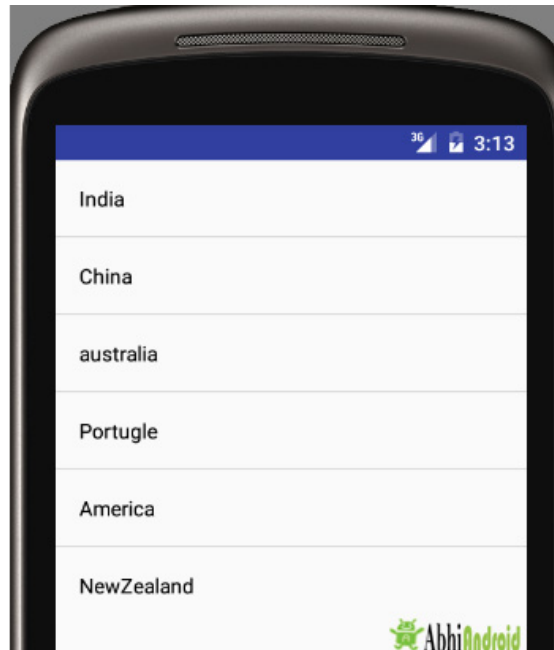
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.widget.ArrayAdapter;import android.widget.ListView;

public class MainActivity extends Activity
{
    // Array of strings...
    ListView simpleList;
    String countryList[] = {"India", "China", "australia", "Portugle",
    "America", "NewZealand"};

    @Override    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        simpleList = (ListView)findViewById(R.id.simpleListView);
        ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(this,
        R.layout.activity_listview, R.id.textView, countryList);
        simpleList.setAdapter(arrayAdapter);
    }
}
```

Output Screen:

Now run the App in Emulator. You will see the below output screen where list of country names will be printed:



2.Base Adapter:

BaseAdapter is a common base class of a general implementation of an Adapter that can be used in ListView. Whenever you need a customized list you create your own adapter and extend base adapter in that. Base Adapter can be extended to create a custom Adapter for displaying a custom list item. ArrayAdapter is also an implementation of BaseAdapter.

Example of list view using Custom adapter(Base adapter):

In this example we display a list of countries with flags. For this, we have to use custom adapter as shown in example:



Step 1: [Create a new project](#) Listbasexample and activity Main Activity. Here we will create a ListView in LinearLayout. **Below is the code of activity_main.xml or content_main.xml:**

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

    <ListView
        android:id="@+id/simpleListView"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:divider="@color/material_blue_grey_800"
        android:dividerHeight="1dp"
        android:footerDividersEnabled="false" />
</LinearLayout>
```

Step 2: Create a new activity name Listview and below is the code of activity_listview.xml

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

    <ImageView
        android:id="@+id/icon"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:src="@drawable/ic_launcher" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:padding="@dimen/activity_horizontal_margin"
        android:textColor="@color/black" />
</LinearLayout>
```

Step 3: In third step we will use custom adapter to display the country names in UI by coding MainActivity.[java](#). **Below is the code of MainActivity.java**

Important Note: Make sure flag images are stored in drawable folder present inside res folder with correct naming.

```
package com.abhiandroid.listbaseexample;

import android.app.Activity;
import android.os.Bundle;
import android.widget.ListView;

public class MainActivity extends Activity {

    ListView simpleList;
    String countryList[] = {"India", "China", "australia", "Portugle",
        "America", "NewZealand"};
    int flags[] = {R.drawable.india, R.drawable.china, R.drawable.australia,
        R.drawable.portugle, R.drawable.america, R.drawable.new_zealand};

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        simpleList = (ListView) findViewById(R.id.simpleListView);
        CustomAdapter customAdapter = new CustomAdapter(getApplicationContext(),
            countryList, flags);
        simpleList.setAdapter(customAdapter);
    }
}
```

Step 4: Now create another class Custom Adapter which will extend BaseAdapter. Below is the code of CustomAdapter.[java](#)

```
package com.abhiandroid.listbaseexample;

import android.content.Context;
import android.media.Image;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;

import java.util.zip.Inflater;

public class CustomAdapter extends BaseAdapter {
    Context context;
    String countryList[];
    int flags[];
    LayoutInflater inflater;

    public CustomAdapter(Context applicationContext, String[] countryList,
        int[] flags) {
        this.context = context;
        this.countryList = countryList;
        this.flags = flags;
        inflater = (LayoutInflater.from(applicationContext));
    }

    @Override
    public int getCount() {
        return countryList.length;
    }

    @Override
    public Object getItem(int i) {
        return null;
    }

    @Override
    public long getItemId(int i) {
        return 0;
    }

    @Override
    public View getView(int i, View view, ViewGroup viewGroup) {
        view = inflater.inflate(R.layout.activity_listview, null);
        TextView country = (TextView) view.findViewById(R.id.textView);
        ImageView icon = (ImageView) view.findViewById(R.id.icon);
        country.setText(countryList[i]);
        icon.setImageResource(flags[i]);
        return view;
    }
}
```

Output:

Now run the App in Emulator and it will show you name of countries along with flags. Below is the output screen:

