Andhra Pradesh State Skill Development Corporation

Skill AP
APSSDC

ANDROID APPLICATION DEVELOPMENT

LIST VIEW

ListView

- Android ListView is a view which groups several items and display them in vertical scrollable list.
- The list items are automatically inserted to the list using an Adapter that pulls content from a source such as an array or database.
- It's one of the basic and most used UI components of android.
- The most common usages include displaying data in the form of a vertical scrolling list.

Using an Adapter

An adapter actually bridges between UI components and the data source that fill data into UI Component. Adapter holds the data and send the data to adapter view, the view can take the data from adapter view and shows the data on different views like as spinner, list view, grid view etc. The adapter pulls the items out of a data source, an array for example, and then converts each item into a view which it then inserts into the ListView.

The **ListView** and **GridView** are subclasses of **AdapterView** and they can be populated by binding them to an Adapter, which retrieves data from an external source and creates a View that represents each data entry. The common adapters are ***ArrayAdapter, BaseAdapter, CursorAdapter, SimpleCursorAdapter, SpinnerAdapter and WrapperListAdapter***.

Handling Android ListView Clicks

The **onListItemClick()** method is used to process the clicks on android ListView item. This method receives 4 parameters:

1. **ListView** : The ListView containing the item views
2. **View** : The specific item view that was selected
3. **Position** : The position of the selected item in the array. Remember that the array is zero indexed, so the first item in the array is at position 0
4. **Id** : The id of the selected item. Not of importance for our tutorial but is important when using data retrieved from a database as you can use the id (which is the id of the row containing the item in the database) to retrieve the item from the database.

Android ListView Example Project Structure

strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <string-array name="teams">
      <item>India</item>
      <item>South Africa</item>
      <item>Australia</item>
      <item>England</item>
      <item>New Zealand</item>
      <item>Sri Lanka</item>
      <item>Pakistan</item>
      <item>West Indies</item>
      <item>Bangladesh</item>
      <item>Ireland</item>
```

```
    </string-array>
</resources>
```

Each list view item will be represented by an xml layout,so lets define the xml layout comprising of a single textview as follows:

list_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--  Single List Item Design -->
<TextView xmlns:android="https://schemas.android.com/apk/res/android"
    android:id="@+id/textview"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:padding="10dip"
        android:textSize="16dip"
        android:textStyle="bold" >
</TextView>
```

Following snippet shows how to import the xml resources data and store them in data followed by binding them to the adapter:

```
// storing string resources into Array
String[] teams = getResources().getStringArray(R.array.teams);

// Binding resources Array to ListAdapter
this.setListAdapter(new ArrayAdapter<String>(this, R.layout.list_item, R.id.textview, teams));
```

In the following code we fetch the data value from the selected item and pass it as a bundle to the next activity using intents.

MainActivity.java

```
package com.example.android;


import android.app.ListActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;

public class MainActivity extends ListActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // storing string resources into Array
        String[] teams = getResources().getStringArray(R.array.teams);

        // Binding resources Array to ListAdapter
```

```
        this.setListAdapter(new ArrayAdapter<String>(this, R.layout.list_item, R.id.textview, team
s));

        ListView lv = getListView();

        // listening to single list item on click
        lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
          public void onItemClick(AdapterView<?> parent, View view,
                              int position, long id) {

            // selected item
            String team = ((TextView) view).getText().toString();

            // Launching new Activity on selecting single List Item
            Intent i = new Intent(getApplicationContext(), SecondActivity.class);
            // sending data to new activity
            i.putExtra("team", team);
            startActivity(i);

          }
        });
    }
}
```

The SecondActivity class retrieves the text label from the list item selected and displays it in a textview as shown in the following snippet.

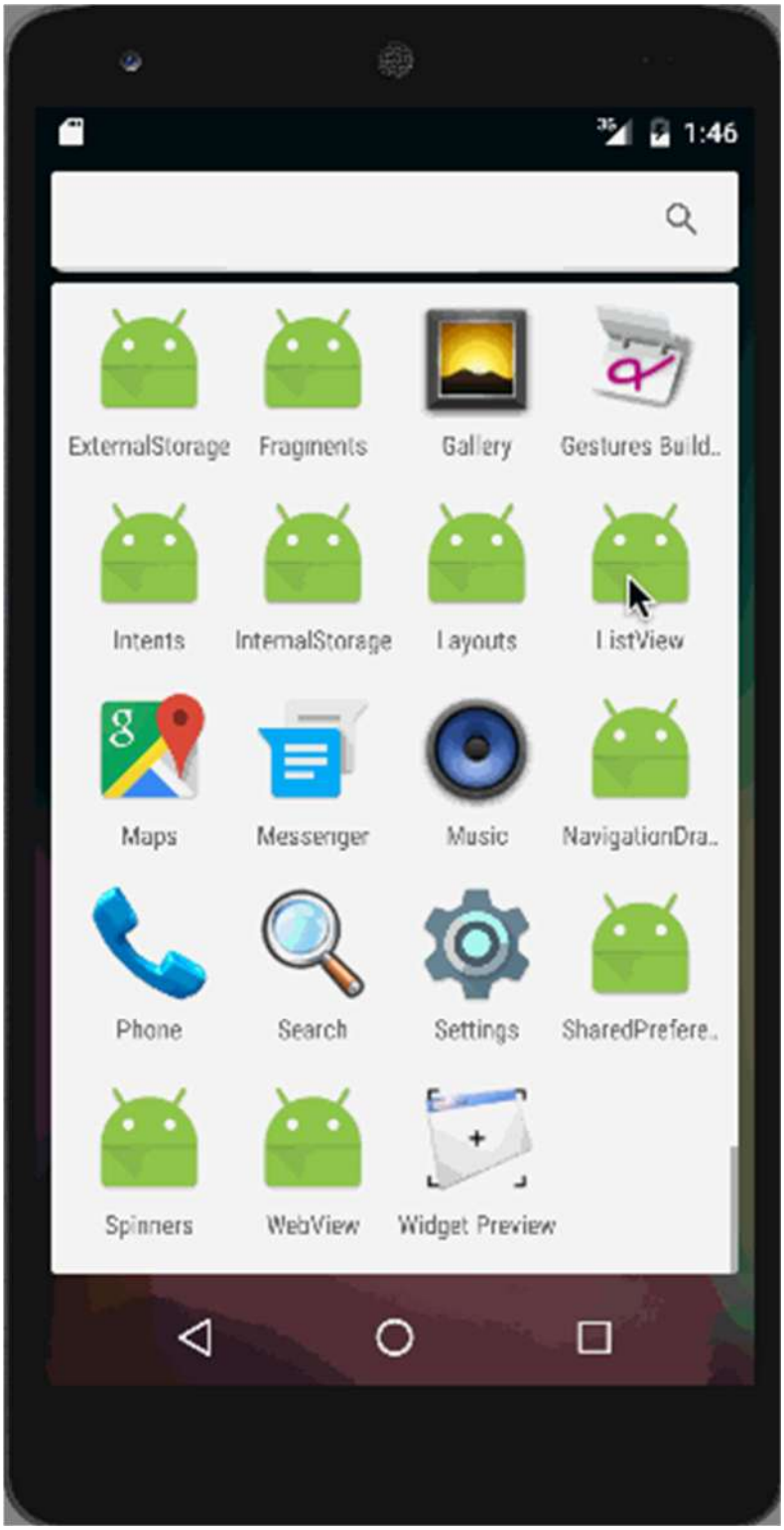
SecondActivity.java

```
package com.example.android;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.widget.TextView;

public class SecondActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        TextView txtProduct = (TextView) findViewById(R.id.team_label);

        Intent i = getIntent();
        // getting attached intent data
        String product = i.getStringExtra("team");
        // displaying selected product name
        txtProduct.setText(product);
    }
}
```



Custom ListView Android Tutorial

Creating a new Project

- As always let's start by creating a new Android Studio Project.
- I created a project named CustomListViewAndroid.
- Now first we will add all the images that we downloaded.

Adding Images

- On your project paste, all the images inside **res->drawable** that you downloaded.

Adding ListView in MainActivity

- Now come inside activity_main.xml and add a ListView here.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">

<ListView
   android:id="@+id/listView"
   android:layout_width="match_parent"
   android:layout_height="wrap_content" />

</RelativeLayout>
```

- So now we have a ListView inside our MainActivity.

Custom List Layout

- So inside res->layout create a new Layout Resource File named my_custom_list.xml. (You can give any name to the file).
- Inside this file, we will design the Layout for our List.
- we will write the following code inside my_custom_list.xml.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:padding="15dp"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical">

   <TextView
      android:id="@+id/textViewName"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_alignParentTop="true"
      android:layout_toEndOf="@+id/imageView"
      android:layout_toRightOf="@+id/imageView"
      android:paddingBottom="10dp"
      android:text="Spiderman"
      android:textAlignment="center"
      android:textSize="35dp"
```

```
    android:textStyle="bold" />

  <ImageView
    android:id="@+id/imageView"
    android:layout_width="150dp"
    android:layout_height="150dp" />

  <TextView
    android:id="@+id/textViewTeam"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/textViewName"
    android:layout_toEndOf="@+id/imageView"
    android:layout_toRightOf="@+id/imageView"
    android:paddingTop="10dp"
    android:text="Spiderman"
    android:textAlignment="center"
    android:textSize="25dp"
    android:textStyle="bold" />

</RelativeLayout>
```

Data Model for List Item

- To store the data of the List, we will use a data model class. The class will only contain the variables for all the List Items, a Constructor to initialize those attributes and getters to get those attribute values.
- So, you need to create the following class. It is named **Hero**.

```
package com.android.example.CustomListViewAndroid;

/**
 * Created by chaitanya on 10/2/2020.
 */

public class Hero {

  int image;
  String name, team;

  public Hero(int image, String name, String team) {
    this.image = image;
    this.name = name;
    this.team = team;
  }

  public int getImage() {
    return image;
  }

  public String getName() {
```

```
        return name;
    }

    public String getTeam() {
        return team;
    }
}
```

- You see in the above class we have **int** for the image. It is because we are going to use drawable resource for the image and every drawable resource has a unique id which the Android Studio creates automatically inside R.java file. And the type of the id generated is int. That is why to identify the image we have an **int** here.

Custom Adapter Class for ListView

- As we are building a customized ListView, we cannot use the predefined **ArrayAdapter**. Create a new class named **MyListAdapter.java** and write the following code.

```
package com.android.example.CustomListViewAndroid;

import android.content.Context;
import android.content.DialogInterface;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

import java.util.List;

/**
 * Created by Chaitanya on 9/14/2017.
 */

//we need to extend the ArrayAdapter class as we are building an adapter
public class MyListAdapter extends ArrayAdapter<Hero> {

    //the list values in the List of type hero
    List<Hero> heroList;

    //activity context
    Context context;

    //the layout resource file for the list items
    int resource;
```

```
    //constructor initializing the values
    public MyListAdapter(Context context, int resource, List<Hero> heroList) {
        super(context, resource, heroList);
        this.context = context;
        this.resource = resource;
        this.heroList = heroList;
    }

    //this will return the ListView Item as a View
    @NonNull
    @Override
    public View getView(final int position, @Nullable View convertView, @NonNull ViewGrou
p parent) {

        //we need to get the view of the xml for our list item
        //And for this we need a layoutinflater
        LayoutInflater layoutInflater = LayoutInflater.from(context);

        //getting the view
        View view = layoutInflater.inflate(resource, null, false);

        //getting the view elements of the list from the view
        ImageView imageView = view.findViewById(R.id.imageView);
        TextView textViewName = view.findViewById(R.id.textViewName);
        TextView textViewTeam = view.findViewById(R.id.textViewTeam);
        Button buttonDelete = view.findViewById(R.id.buttonDelete);

        //getting the hero of the specified position
        Hero hero = heroList.get(position);

        //adding values to the list item
        imageView.setImageDrawable(context.getResources().getDrawable(hero.getImage()));
        textViewName.setText(hero.getName());
        textViewTeam.setText(hero.getTeam());

        //finally returning the view
        return view;
    }
}
```

Custom ListView Android

- Now the last thing is making our Custom ListView. So com inside MainActivity.java and write the following code. ```Java package com.android.example.CustomListViewAndroid;

```
import android.support.v7.app.AppCompatActivity; import android.os.Bundle; import
android.widget.ListView;
import java.util.ArrayList; import java.util.List;
public class MainActivity extends AppCompatActivity {

//a List of type hero for holding list items
List<Hero> heroList;
```

```
//the listview
ListView listView;

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

    //initializing objects
    heroList = new ArrayList<>();
    listView = (ListView) findViewById(R.id.listView);

    //adding some values to our list
    heroList.add(new Hero(R.drawable.spiderman, "Spiderman", "Avengers"));
    heroList.add(new Hero(R.drawable.joker, "Joker", "Injustice Gang"));
    heroList.add(new Hero(R.drawable.ironman, "Iron Man", "Avengers"));
    heroList.add(new Hero(R.drawable.doctorstrange, "Doctor Strange", "Avengers"));
    heroList.add(new Hero(R.drawable.captainamerica, "Captain America", "Avengers"));
    heroList.add(new Hero(R.drawable.batman, "Batman", "Justice League"));

    //creating the adapter
    MyListAdapter adapter = new MyListAdapter(this, R.layout.my_custom_list, heroList);

    //attaching adapter to the listview
    listView.setAdapter(adapter);
}

} ```
```

CustomListView Output :