LISTVIEW ANDROID

ListView Description:-

- Android ListView is a **ViewGroup** , that is used to display list of items in multiple rows & contains an adapter that automatically inserts the items the list.

- The main purpose of **Adapter** is to fetch data from an array **(or)** database and insert each item that placed into the list for the desired result.

How to implement Listview in Andriod:-

Step 1: - First we need to create MainActivity.java,

MainActivity.java

- For beginners, here you guys may have a doubt what is meant by Activity.
- Activity is nothing but a single UI representation of Android Application.
- (or) by means it allows or helps user to interact with application in simple terms.

Step 2: - We need to create activity_Main.xml,

activity_Main.xml,

- Insert ListView widget

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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="match_parent"/>


</androidx.constraintlayout.widget.ConstraintLayout>
```

Step 3: - We need to create Model Class: -

A Short intro about model class,

- Model Class is typically used to "**model**" data in your application.
- For example, you can write a model class that mirrors the database table (or) a JSON.
- Here, we create a class as "UserModel.java", a model class for ListViewAdapter.

Usermodel. Java: -

```
package com.example.listview_java;

public class UserModel {


    public  int userId;
    public String userName;

    public UserModel(int userId, String userName) {
        this.userId = userId;
        this.userName = userName;
    }


    public int getUserId() {
        return userId;
    }

    public void setUserId(int userId) {
        this.userId = userId;
    }


    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }
}
```

Step 4:- Create an Base Adapter :-

A Short intro about Base Adapter ,

- Base Adapter means, as its name implies, it is the base class for so many concrete adapter implementations in Android.

- It is Abstract and therefore it cannot be directly instantiated.
- If your data source is an arrayList (or) array, we can also use the ArrayAdapter constructor as an alternative.

Predefined Methods used in Base Adapter: -

- **Int getItemCount()**
- **Object getItem(int position)**
- **Long getItemId(int position)**
- **View getView(int position, View convertView, ViewGroup parent)**

Detail Explanation about each Methods in BaseAdapter :-

```
@Override
    public int getCount() {
        return arrayList.size();
    }
```

Description :- It returns that size , total of the items in the list .

```
@Override
    public Object getItem(int position) {
        return arrayList.get(position);
    }
```

Description :- It returns list item in specified position .

```
@Override
    public long getItemId(int position) {
        return position;
    }
```

Description:- It will return the underlying widget ID's field for the item in position.

```
@Override
    public View getView(int position, View view, ViewGroup parent) {
        if (view == null){
            view = LayoutInflater.from(context).inflate(R.layout.item_lt,
parent, false);
        }

        TextView textView = (TextView) view.findViewById(R.id.txt);

        //Instatiate the UserModel
```

```
        UserModel model = (UserModel) getItem(position);

        textView.setText(model.getUserId()+" -- "+model.getUserName());

        return view;
    }
}
```

ExampleListViewAdapter.java

```
public class ExampleListViewAdapter extends BaseAdapter {


    MainActivity context;
    ArrayList<UserModel> arrayList;


    public ExampleListViewAdapter(MainActivity context,
ArrayList<UserModel> arrayList) {
        this.context = context;
        this.arrayList = arrayList;
    }

    @Override
    public int getCount() {
        return arrayList.size();
    }

    @Override
    public Object getItem(int position) {
        return arrayList.get(position);
    }

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

    @Override
    public View getView(int position, View view, ViewGroup parent) {
        if (view == null){
            view = LayoutInflater.from(context).inflate(R.layout.item_lt,
parent, false);
        }

        TextView textView = (TextView) view.findViewById(R.id.txt);
```

```
        //Instatiate the UserModel
        UserModel model = (UserModel) getItem(position);

        textView.setText(model.getUserId()+" -- "+model.getUserName());

        return view;
    }
}
```

Step 5:- Set Base Adapter For Android ListView :-

MainActivity.java

```
public class MainActivity extends AppCompatActivity {


    ListView listView;
    ArrayList<UserModel> arrayList;

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

        listView = (ListView) findViewById(R.id.listView);

        //implement Adapter and Connect to Adapter
        /*ExampleListViewAdapter adapter = new
ExampleListViewAdapter(MainActivity.this, generateArrayList());
        listView.setAdapter(adapter);  */


        listView.setAdapter(new ExampleListViewAdapter(MainActivity.this,
generateArrayList()));

    }

    private ArrayList<UserModel> generateArrayList(){

        arrayList = new ArrayList<UserModel>();
        arrayList.add(new UserModel(1, "praveen"));
        arrayList.add(new UserModel(2, "kumar"));
        arrayList.add(new UserModel(3, "sathish"));
        arrayList.add(new UserModel(4, "Santhosh"));
        arrayList.add(new UserModel(5, "senthil"));
        arrayList.add(new UserModel(6, "Dheeman"));
        arrayList.add(new UserModel(7, "prathiba"));
        arrayList.add(new UserModel(8, "ramya"));
```

```
        arrayList.add(new UserModel(9, "Shyamili"));
        return  arrayList;

    }


}
```

ListView PRO's and CON's :-

Adavantages :-

- Easy to implement
- OnItemClick Listener

Dis-Advantage :-

- Bad performance in huge List of Items.
- Complicate way to use View Holder Pattern (but can use it).
- Vertical List only

**********END**********