Below is a comprehensive example that demonstrates many common input controls and UI features in Android. This example covers:

* **Buttons**
* **Checkboxes**
* **Radio Buttons (inside a RadioGroup)**
* **Toggle Buttons**
* **Spinners**
* **Handling Input Events**
* **Menus**
* **Toast Messages**
* **Dialogs**
* **Using Styles & Themes**
* **Creating a List (ListView) with a Custom Adapter (Custom List)**

You can use these examples as a starting point for your own projects.

**1. Project Structure Overview**

pgsql

CopyEdit

app/

├─ src/

│ └─ main/

│ ├─ java/com/example/inputcontrols/

│ │ ├─ MainActivity.java

│ │ ├─ CustomListAdapter.java

│ │ └─ ListItem.java

│ ├─ res/

│ │ ├─ layout/

│ │ │ ├─ activity\_main.xml

│ │ │ └─ custom\_list\_item.xml

│ │ ├─ menu/

│ │ │ └─ main\_menu.xml

│ │ └─ values/

│ │ ├─ styles.xml

│ │ └─ colors.xml

│ └─ AndroidManifest.xml

└─ build.gradle

**2. Layout Files**

**2.1 Main Layout (activity\_main.xml)**

This layout includes a Button, CheckBox, RadioGroup (with RadioButtons), ToggleButton, Spinner, another Button (to trigger a Dialog), and a ListView for a custom list.

xml

CopyEdit

<?xml version="1.0" encoding="utf-8"?>

<!-- res/layout/activity\_main.xml -->

<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<LinearLayout

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="16dp">

<!-- Button to show a Toast -->

<Button

android:id="@+id/btnShowToast"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Show Toast" />

<!-- CheckBox -->

<CheckBox

android:id="@+id/checkboxSample"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Check me" />

<!-- RadioGroup with RadioButtons -->

<RadioGroup

android:id="@+id/radioGroup"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="horizontal"

android:layout\_marginTop="8dp">

<RadioButton

android:id="@+id/radioButton1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Option 1" />

<RadioButton

android:id="@+id/radioButton2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Option 2" />

</RadioGroup>

<!-- ToggleButton -->

<ToggleButton

android:id="@+id/toggleButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textOff="OFF"

android:textOn="ON"

android:layout\_marginTop="8dp" />

<!-- Spinner -->

<Spinner

android:id="@+id/spinner"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp" />

<!-- Button to show a Dialog -->

<Button

android:id="@+id/btnShowDialog"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Show Dialog"

android:layout\_marginTop="8dp" />

<!-- ListView for Custom List -->

<ListView

android:id="@+id/listView"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="16dp" />

</LinearLayout>

</ScrollView>

**2.2 Custom List Item Layout (custom\_list\_item.xml)**

This is a simple layout for each item in our custom list. (You can add your own images and text.)

xml

CopyEdit

<?xml version="1.0" encoding="utf-8"?>

<!-- res/layout/custom\_list\_item.xml -->

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal"

android:padding="8dp">

<ImageView

android:id="@+id/customImage"

android:layout\_width="40dp"

android:layout\_height="40dp"

android:src="@mipmap/ic\_launcher" />

<TextView

android:id="@+id/customText"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginStart="8dp"

android:textSize="16sp"

android:text="Custom Item" />

</LinearLayout>

**2.3 Menu Resource (main\_menu.xml)**

A simple menu with two items.

xml

CopyEdit

<?xml version="1.0" encoding="utf-8"?>

<!-- res/menu/main\_menu.xml -->

<menu xmlns:android="http://schemas.android.com/apk/res/android">

<item

android:id="@+id/menu\_settings"

android:title="Settings" />

<item

android:id="@+id/menu\_about"

android:title="About" />

</menu>

**3. Java Code**

**3.1 MainActivity.java**

This activity wires up all the UI elements, handles events, shows Toast messages, opens a Dialog, and inflates the options menu.

java

CopyEdit

// src/main/java/com/example/inputcontrols/MainActivity.java

package com.example.inputcontrols;

import android.app.AlertDialog;

import android.content.DialogInterface;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.widget.\*;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

Button btnShowToast, btnShowDialog;

CheckBox checkBox;

RadioGroup radioGroup;

ToggleButton toggleButton;

Spinner spinner;

ListView listView;

@Override

protected void onCreate(Bundle savedInstanceState) {

// Set the theme defined in styles.xml via the manifest (if needed)

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

// Initialize UI elements

btnShowToast = findViewById(R.id.btnShowToast);

btnShowDialog = findViewById(R.id.btnShowDialog);

checkBox = findViewById(R.id.checkboxSample);

radioGroup = findViewById(R.id.radioGroup);

toggleButton = findViewById(R.id.toggleButton);

spinner = findViewById(R.id.spinner);

listView = findViewById(R.id.listView);

// Button event to show a Toast with CheckBox and RadioButton status

btnShowToast.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// Get CheckBox state

String checkState = checkBox.isChecked() ? "Checkbox is checked" : "Checkbox is unchecked";

// Get selected RadioButton text

int selectedId = radioGroup.getCheckedRadioButtonId();

RadioButton selectedRadio = findViewById(selectedId);

String radioText = (selectedRadio != null) ? selectedRadio.getText().toString() : "No option selected";

// Get ToggleButton state

String toggleState = toggleButton.isChecked() ? "Toggle is ON" : "Toggle is OFF";

String message = "Toast Message:\n" + checkState + "\n" + radioText + "\n" + toggleState;

Toast.makeText(MainActivity.this, message, Toast.LENGTH\_LONG).show();

}

});

// Setup the Spinner

final String[] spinnerItems = {"Spinner Item 1", "Spinner Item 2", "Spinner Item 3"};

ArrayAdapter<String> spinnerAdapter = new ArrayAdapter<>(this,

android.R.layout.simple\_spinner\_item, spinnerItems);

spinnerAdapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

spinner.setAdapter(spinnerAdapter);

spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {

@Override

public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {

Toast.makeText(MainActivity.this, "Selected: " + spinnerItems[position],

Toast.LENGTH\_SHORT).show();

}

@Override

public void onNothingSelected(AdapterView<?> parent) { }

});

// Button event to show an AlertDialog

btnShowDialog.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) { showAlertDialog(); }

});

// Create and set a Custom Adapter for the ListView

ArrayList<ListItem> customItems = new ArrayList<>();

customItems.add(new ListItem("Custom Item 1", R.mipmap.ic\_launcher));

customItems.add(new ListItem("Custom Item 2", R.mipmap.ic\_launcher));

customItems.add(new ListItem("Custom Item 3", R.mipmap.ic\_launcher));

CustomListAdapter adapter = new CustomListAdapter(this, customItems);

listView.setAdapter(adapter);

}

// Method to show an AlertDialog

private void showAlertDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setTitle("Dialog Title")

.setMessage("This is a dialog message.")

.setPositiveButton("OK", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

Toast.makeText(MainActivity.this, "OK clicked", Toast.LENGTH\_SHORT).show();

}

})

.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

dialog.dismiss();

}

});

AlertDialog dialog = builder.create();

dialog.show();

}

// Inflate the Options Menu

@Override

public boolean onCreateOptionsMenu(Menu menu) {

MenuInflater inflater = getMenuInflater();

inflater.inflate(R.menu.main\_menu, menu);

return true;

}

// Handle Menu Item selections

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()){

case R.id.menu\_settings:

Toast.makeText(this, "Settings selected", Toast.LENGTH\_SHORT).show();

return true;

case R.id.menu\_about:

Toast.makeText(this, "About selected", Toast.LENGTH\_SHORT).show();

return true;

default:

return super.onOptionsItemSelected(item);

}

}

}

**3.2 CustomListAdapter.java**

A simple custom adapter to populate our ListView with custom list items.

java

CopyEdit

// src/main/java/com/example/inputcontrols/CustomListAdapter.java

package com.example.inputcontrols;

import android.content.Context;

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.ArrayList;

public class CustomListAdapter extends BaseAdapter {

private Context context;

private ArrayList<ListItem> items;

public CustomListAdapter(Context context, ArrayList<ListItem> items) {

this.context = context;

this.items = items;

}

@Override

public int getCount() { return items.size(); }

@Override

public Object getItem(int position) { return items.get(position); }

@Override

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

@Override

public View getView(int position, View convertView, ViewGroup parent) {

if(convertView == null) {

convertView = LayoutInflater.from(context).inflate(R.layout.custom\_list\_item, parent, false);

}

ImageView imageView = convertView.findViewById(R.id.customImage);

TextView textView = convertView.findViewById(R.id.customText);

ListItem item = items.get(position);

imageView.setImageResource(item.getImageRes());

textView.setText(item.getText());

return convertView;

}

}

**3.3 ListItem.java**

A simple model class for items in the custom list.

java

CopyEdit

// src/main/java/com/example/inputcontrols/ListItem.java

package com.example.inputcontrols;

public class ListItem {

private String text;

private int imageRes;

public ListItem(String text, int imageRes) {

this.text = text;

this.imageRes = imageRes;

}

public String getText() { return text; }

public int getImageRes() { return imageRes; }

}

**4. Styles and Themes**

You can define your styles and themes in **res/values/styles.xml**. For example:

xml

CopyEdit

<?xml version="1.0" encoding="utf-8"?>

<!-- res/values/styles.xml -->

<resources>

<style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">

<!-- Customize your theme colors -->

<item name="colorPrimary">@color/colorPrimary</item>

<item name="colorPrimaryDark">@color/colorPrimaryDark</item>

<item name="colorAccent">@color/colorAccent</item>

</style>

</resources>

Make sure your theme is applied in your **AndroidManifest.xml**:

xml

CopyEdit

<application

android:theme="@style/AppTheme"

... >

<!-- Activities here -->

</application>

**5. Running the Example**

1. **Add Dependencies:**  
   Ensure you have the required dependencies in your build.gradle (for AppCompat, etc.).
2. **Resources:**  
   Use your own images or the default launcher icon for the list item image (referenced as @mipmap/ic\_launcher).
3. **Build & Run:**  
   Run the app on your emulator or device. Interact with the various controls:
   * Tap the “Show Toast” button to see a Toast message incorporating the states of the CheckBox, RadioButtons, and ToggleButton.
   * Use the Spinner to select an item.
   * Tap the “Show Dialog” button to see an AlertDialog.
   * Use the options menu (in the ActionBar) to see menu item selection handling.
   * Scroll through the ListView to see your custom list items.