Programación Móvil

Nombre: ***Adrian Eduardo Reynosa Flores***

#2010455

1. Configuración en `build.gradle`

Asegúrate de tener las dependencias necesarias en tu archivo `build.gradle` para utilizar `RecyclerView` y `CardView` (opcional, pero útil para diseño de elementos de lista):

```
implementation 'androidx.recyclerview:recyclerview:1.2.0'
implementation 'androidx.cardview:cardview:1.0.0'
```

2. Creación de un diseño de elemento de lista (`item_layout.xml`)

Crea un archivo XML que defina el diseño de cada elemento de la lista en `res/layout/item_layout.xml`. Este diseño contendrá los elementos visuales que quieres mostrar para cada elemento de la lista. Por ejemplo:

```
<!-- item_layout.xml -->
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="16dp">

    <TextView
        android:id="@+id/textViewTitle"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="18sp"
        android:textStyle="bold"/>

    <TextView
        android:id="@+id/textViewDescription"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="14sp"/>
</LinearLayout>
```

3. Creación del adaptador (`MyAdapter.java`)

Crea una clase adaptador que extienda `RecyclerView.Adapter`. Este adaptador se encargará de vincular los datos de tu lista con los elementos visuales en el `RecyclerView`.

```
public class MyAdapter extends
RecyclerView.Adapter<MyAdapter.MyViewHolder> {

    private List<MyDataModel> dataList; // Lista de datos a mostrar

    // Constructor para inicializar con los datos
    public MyAdapter(List<MyDataModel> dataList) {
        this.dataList = dataList;
    }

    // ViewHolder para los elementos de la lista
    public static class MyViewHolder extends RecyclerView.ViewHolder {
        public TextView textViewTitle, textViewDescription;

        public MyViewHolder(View itemView) {
            super(itemView);
            textViewTitle = itemView.findViewById(R.id.textViewTitle);
            textViewDescription =
itemView.findViewById(R.id.textViewDescription);
        }
    }

    @Override
    public MyViewHolder onCreateViewHolder(ViewGroup parent, int
viewType) {
        View v =
LayoutInflater.from(parent.getContext()).inflate(R.layout.item_layout,
parent, false);
        return new MyViewHolder(v);
    }

    @Override
    public void onBindViewHolder(MyViewHolder holder, int position) {
        MyDataModel data = dataList.get(position);
        holder.textViewTitle.setText(data.getTitle());
        holder.textViewDescription.setText(data.getDescription());
    }

    @Override
    public int getItemCount() {
        return dataList.size();
    }
}
```

4. Configuración del `RecyclerView` en la actividad o fragmento

En tu actividad o fragmento donde desees mostrar el `RecyclerView`, configura el `RecyclerView`, crea una lista dinámica de datos (`List<MyDataModel>` en este ejemplo) y asigna el adaptador al `RecyclerView`.

```
public class MainActivity extends AppCompatActivity {
```

```
    private RecyclerView recyclerView;
    private MyAdapter adapter;
    private List<MyDataModel> dataList;

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

        recyclerView = findViewById(R.id.recyclerView);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));

        // Inicialización de la lista de datos (puedes llenarla de alguna
fuente de datos)
        dataList = new ArrayList<>();
        dataList.add(new MyDataModel("Título 1", "Descripción 1"));
        dataList.add(new MyDataModel("Título 2", "Descripción 2"));
        // Añadir más datos si es necesario...

        adapter = new MyAdapter(dataList);
        recyclerView.setAdapter(adapter);
    }
}
```

5. Definición del modelo de datos (`MyDataModel.java`)

Define una clase para tu modelo de datos si no tienes una ya. Esta clase encapsula los datos que quieres mostrar en cada elemento de la lista.

```
public class MyDataModel {

    private String title;
    private String description;

    public MyDataModel(String title, String description) {
        this.title = title;
        this.description = description;
    }

    public String getTitle() {
        return title;
    }

    public String getDescription() {
        return description;
    }
}
```