Documentación

Proyecto recycler type

acivity_main.xml

El layout principal de la aplicación será una lista donde se visualizarán todos los elementos cargados. Contiene un **RecyclerView** que contendrá cada uno de los elementos de la lista.

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

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/lista"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:clipToPadding="false"/>

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

MainActivity.java

Para poder desarrollar el java del layout principal necesitaremos los siguiente imports:

```
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;
import java.util.ArrayList;
```

Estás son las variables de clase:

```
public class MainActivity extends AppCompatActivity implements
View.OnLongClickListener, View.OnClickListener {

    public static final int TYPE_ITEM_1 = 1;
```

```
    public static final int TYPE_ITEM_2 = 2;
    public static final int TYPE_ITEM_3 = 3;

    public ArrayList<Dato> datos;
    private RecyclerView recyclerView;
}
```

Las variables estáticas las utilzaremos para identificar cada uno de los tipos distintos de vista que tendremos dependiendo del tipo de dato. También tendremos una variable pública de datos, que serán donde guardaremos todos los datos que posteriormente utilizaremos en la lista del layout principal de la aplicación. Podemos cargar los datos que deseamos, en este caso estos son los que usaremos.

```
private void cargarDatos() {
        datos = new ArrayList<>();
        datos.add(new Dato(
            "Cohete espacial",
            "Soy un cohete espacial que viajo por el espacio interestela",
            BitmapFactory.decodeResource(this.getResources(),
R.drawable.cohete_flat),
            TYPE_ITEM_1));
        datos.add(new Dato(
            "Coordillera de noche",
            "No hay nada como una noche plácida en la montaña, ¿verdad?",
            BitmapFactory.decodeResource(this.getResources(),
R.drawable.material_flat),
            TYPE_ITEM_2));
        datos.add(new Dato(
            "London city",
            "No hay nada como pasear por la orilla del Támesis en una mañana con
niebla",
            BitmapFactory.decodeResource(this.getResources(),
R.drawable.london_flat),
            TYPE_ITEM_3));
        datos.add(new Dato(
            "Discovery en la noche",
            "Viajar al epacio, recorrer la vía láctea, y volver a casa con mi
Discovery...",
            BitmapFactory.decodeResource(this.getResources(),
R.drawable.moon_flat),
            TYPE_ITEM_3));
    }
```

Implemenación de las interfaces **View.OnClickListener** y **View.OnLongClickListener** Es mostrar simplemente un toast indicando que se ha realizado.

```
@Override
    public void onClick(View v) {
        Toast.makeText(this, "Pulsación corta sobre " +
```

```
datos.get(recyclerView.getChildAdapterPosition(v)).getTextoCorto(),
Toast.LENGTH_LONG).show();
    }

    @Override
    public boolean onLongClick(View v) {
        Toast.makeText(this, "Pulsación larga sobre " +

datos.get(recyclerView.getChildAdapterPosition(v)).getTextoCorto(),
Toast.LENGTH_LONG).show();
        return false;
    }
```

Finalmente, el método principal **onCreate**

1. Realizamos la carga de datos
2. Inicializamos el **RecyclerView**
3. Creamos el adaptador y le asignamos los manejadores de eventos
    1. También se le asigna un manejador para el click de las imágenes internas de cada elemento de la lista
4. Finalmente asignamos el adaptador al **RecyclerView** y le asignamos un **LayoutManager**

```
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        cargarDatos();
        recyclerView = findViewById(R.id.lista);
        Adaptador adaptador = new Adaptador(this);
        adaptador.setClickListener(this);
        adaptador.setLongClickListener(this);
        adaptador.setImageClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(MainActivity.this,
                "Has pulsado el elemento interno de la CardView",
                Toast.LENGTH_LONG).show();
            }
        });
        recyclerView.setAdapter(adaptador);
        recyclerView.setLayoutManager(new LinearLayoutManager(this,
RecyclerView.VERTICAL, false));
    }
```

Dato.java

El siguiente paso será la creación del POJO que manejara los datos

```
package com.danito.p_recyclertype;

import android.graphics.Bitmap;

public class Dato {
    String textoCorto;
    String textoLargo;
    Bitmap foto;
    int tipo;

    public Dato(String textoCorto, String textoLargo, Bitmap foto, int tipo) {
        this.textoCorto = textoCorto;
        this.textoLargo = textoLargo;
        this.foto = foto;
        this.tipo = tipo;
    }

    public String getTextoCorto() {
        return textoCorto;
    }

    public void setTextoCorto(String textoCorto) {
        this.textoCorto = textoCorto;
    }

    public String getTextoLargo() {
        return textoLargo;
    }

    public void setTextoLargo(String textoLargo) {
        this.textoLargo = textoLargo;
    }

    public Bitmap getFoto() {
        return foto;
    }

    public void setFoto(Bitmap foto) {
        this.foto = foto;
    }

    public int getTipo() {
        return tipo;
    }

    public void setTipo(int tipo) {
        this.tipo = tipo;
    }
}
```

Adaptador.java

Imports

```
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
```

Variables de clase

```
public class Adaptador extends RecyclerView.Adapter implements
View.OnClickListener, View.OnLongClickListener{
    Context context;
    View.OnClickListener clickListener;
    View.OnLongClickListener longClickListener;
    View.OnClickListener imageClickListener;
}
```

1. `context`, para almacenar el contexto y datos de la activity principal.
2. `clickLister`, que será el encargado de gestionar la pulsación corta sobre cada elemento
3. `longClickListener`, que será el encargado de gestionar la pulsación larga sobre cada elemento
4. `imageClickListener`, que será el encargado de gestionar la pulsación corta sobre cada la imagen

Constructor para inicializar el contexto

```
public Adaptador(Context context) {
        this.context = context;
    }
```

El siguiente método se utiliza de manera interna para obtener la posición del elemento en la lista

```
@Override
    public int getItemViewType(int position) {
        return ((MainActivity)context).datos.get(position).getTipo();
    }
```

El método que enlaza una vista al holder dependiendo del tipo de vista Dependiendo del tipo que devuelva la variable **viewType**, asignaremos al holder una vista u otra Después de establecer que vista se enlazará al

Holder, le asigaremos los listener correspondientes

```
@NonNull
    @Override
    public RecyclerView.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent,
int viewType) {
        View v = null;
        switch (viewType) {
            case MainActivity.TYPE_ITEM_1:
                v =
LayoutInflater.from(parent.getContext()).inflate(R.layout.item_type_1, parent,
false);
                break;
            case MainActivity.TYPE_ITEM_2:
                v =
LayoutInflater.from(parent.getContext()).inflate(R.layout.item_type_2, parent,
false);
                break;
            case MainActivity.TYPE_ITEM_3:
                v =
LayoutInflater.from(parent.getContext()).inflate(R.layout.item_type_3, parent,
false);
                break;
        }
        Holder holder = new Holder(v);
        holder.setImageClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (imageClickListener != null) {
                    imageClickListener.onClick(v);
                }
            }
        });
        v.setOnClickListener(clickListener);
        v.setOnLongClickListener(longClickListener);
        return holder;
    }
```

El método que enlaza los datos a un Holder

```
@Override
    public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int
position) {
        ((Holder) holder).bind(((MainActivity) context).datos.get(position));
    }
```

El método que es necesario implementar para el correcto funcinamiento del adaptador

```
@Override
    public int getItemCount() {
        return ((MainActivity) context).datos.size();
    }
```

Métodos para establecer los listener distintos y para decidir que ocurrirá al provocar el evento Hay dos métodos por cada uno, menos para el `setImageClickListener`, para el que decidimos que hacer mediante un listener anónimo

```
public void setImageClickListener(View.OnClickListener listener) {
        if (listener != null) {
            imageClickListener = listener;
        }
    }

    public void setClickListener(View.OnClickListener listener) {
        if (listener != null){
            clickListener = listener;
        }
    }

    @Override
    public void onClick(View v) {
        if (clickListener != null) {
            clickListener.onClick(v);
        }
    }

    public void setLongClickListener(View.OnLongClickListener listener) {
        if (listener != null) {
            longClickListener = listener;
        }
    }

    @Override
    public boolean onLongClick(View v) {
        if (longClickListener != null) {
            longClickListener.onLongClick(v);
        }
        return false;
    }
```

Holder.java

Imports

```

```

```
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import androidx.recyclerview.widget.RecyclerView;
```

Variables de clase

```
class Holder extends RecyclerView.ViewHolder implements View.OnClickListener {

    private ImageView imageView, imageView1, imageView2, imageView3;
    private Button bShare, bExplore;
    private TextView textView, textViewL;
    private View.OnClickListener imageClickListener;
}
```

Las variables de clase mantendrán los elementos de la vista de cada uno de los elementos de la lista

Constructor de la clase

1. Inicializamos los elementos dependiendo de la vista que sea
2. El id que comprobaremos será en de la vista que le pasamos en el constructor, lo comparamos con el contenedor principal de cada uno de los layouts
3. Les asignamos los listener correspondientes
4. Los últimos dos elementos están fuera del switch porque son comunes a todas las vistas

```
public Holder(View v) {
        super(v);
        switch (v.getId()) {
            case R.id.cadview1:
                break;
            case R.id.cadview2:
                imageView1 = v.findViewById(R.id.imageView1);
                imageView1.setOnClickListener(this);
                imageView2 = v.findViewById(R.id.imageView2);
                imageView2.setOnClickListener(this);
                imageView3 = v.findViewById(R.id.imageView3);
                imageView3.setOnClickListener(this);
                break;
            case R.id.cadview3:
                bShare = v.findViewById(R.id.bShare);
                bShare.setOnClickListener(this);
                bExplore = v.findViewById(R.id.bExplore);
                bExplore.setOnClickListener(this);
                textViewL = v.findViewById(R.id.textViewL);
                break;
```

```
        }
        textView = v.findViewById(R.id.txt_title);
        imageView = v.findViewById(R.id.imageView);
    }
```

Inicialización de los elementos dependiendo del tipo de dato que sea Como `imageView` y `textView` son comunes a todos, se pueden inicializar fuera del switch

```
public void bind(Dato dato) {
        switch (dato.getTipo()) {
            case MainActivity.TYPE_ITEM_1:
                break;
            case MainActivity.TYPE_ITEM_2:
                break;
            case MainActivity.TYPE_ITEM_3:
                textViewL.setText(dato.getTextoLargo());
                break;
        }
        imageView.setImageBitmap(dato.getFoto());
        textView.setText(dato.getTextoCorto());
    }
```

Asignación de listener e implementación del evento

```
public void setImageClickListener(View.OnClickListener listener) {
        if (listener != null) {
            imageClickListener = listener;
        }
    }

    @Override
    public void onClick(View v) {
        if (imageClickListener != null) {
            imageClickListener.onClick(v);
        }
    }
```