

# RecyclerView

## Construindo uma lista diferente

- Para que seja possível utilizar o **RecyclerView** e o **CardView** é necessário a inclusão das referências as respectivas implementações no **build.gradle** da aplicação.

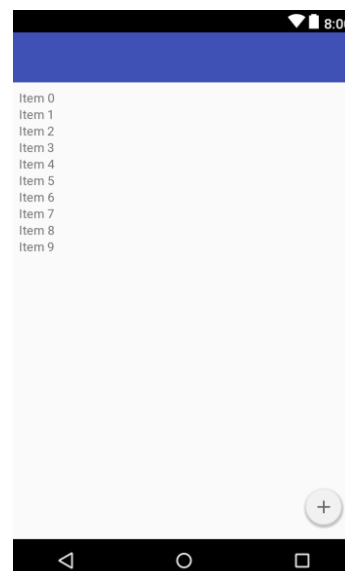
```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 26
    defaultConfig {
        applicationId "br.senai.sp.informatica.albunsmusicais"
        minSdkVersion 19
        targetSdkVersion 26
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
        android.defaultConfig.vectorDrawables.useSupportLibrary = true
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(include: ['*.jar'], dir: 'libs')
    implementation 'com.android.support:appcompat-v7:26.1.0'
    implementation 'com.android.support.constraint:constraint-layout:1.0.2'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.1'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'
    implementation 'com.android.support:design:26.1.0'
    implementation 'com.android.support:recyclerview-v7:26.1.0'
    implementation 'com.android.support:cardview-v7:26.1.0'
}
```

- Após a configuração e o sincronismo do projeto, basta declarar o **RecyclerView** no arquivo de layout que desejar.

```
<android.support.v7.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginBottom="8dp"
    android:layout_marginLeft="8dp"
    android:layout_marginRight="8dp"
    android:layout_marginTop="8dp"
    android:scrollbars="vertical"
    android:visibility="visible"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```



- Para criar o **CardView** é necessário um novo layout onde deverão ser ajustados os tamanho e formado deste e a inclusão dos elementos que irão compor este catálogo.

```
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">

    <android.support.v7.widget.CardView
        xmlns:card_view="http://schemas.android.com/apk/res-auto"
        android:id="@+id/card_view"
        android:layout_width="100dp"
        android:layout_height="150dp"
        android:layout_gravity="center"
        android:layout_margin="8dp"
        android:elevation="6dp"
        card_view:cardCornerRadius="4dp"
        card_view:layout_constraintBottom_toBottomOf="parent"
        card_view:layout_constraintEnd_toEndOf="parent"
        card_view:layout_constraintStart_toStartOf="parent"
        card_view:layout_constraintTop_toTopOf="parent">

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical">
            <ImageView
                android:id="@+id/ivFoto"
                android:layout_width="80dp"
                android:layout_height="80dp"
                android:layout_gravity="center_horizontal"
                android:layout_marginEnd="8dp"
                android:layout_marginStart="8dp"
                android:layout_marginTop="10dp"
                android:background="@android:color/darker_gray" />

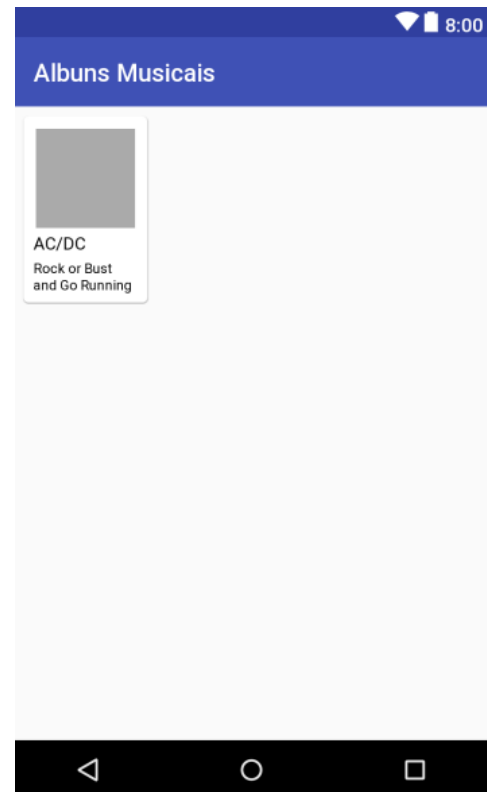
            <TextView
                android:id="@+id/tvBanda"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginEnd="8dp"
                android:layout_marginStart="8dp"
                android:layout_marginTop="3dp"
                android:text="AC/DC"
                android:textColor="@android:color/black"
                android:textSize="14sp" />

            <TextView
                android:id="@+id/tvAlbum"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginEnd="8dp"
                android:layout_marginStart="8dp"
                android:layout_marginTop="3dp"
                android:maxLines="2"
                android:text="Rock or Bust and Go Running alive"
                android:textColor="@android:color/black"
                android:textSize="11sp" />
        </LinearLayout>

        <CheckBox
            android:id="@+id/checkBox"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="8dp"
            android:button="@drawable/new_checkbox"
            android:visibility="invisible" />

    </android.support.v7.widget.CardView>

</android.support.constraint.ConstraintLayout>
```



- Da mesma forma que no **ListView** o **RecyclerView** necessita de um *Adapter*, neste caso deve ser utilizado **RecyclerView.Adapter** como classe base para a sua construção.

```
public class AlbumRecycledAdapter
    extends RecyclerView.Adapter<AlbumRecycledAdapter.AlbumViewHolder>
    implements AdapterInterface {

    private AlbumDao dao = AlbumDao.instance;
    private SparseLongArray mapa;
    private Activity activity;
    private OnItemClickListener listener;
    private boolean editar = false;

    public AlbumRecycledAdapter(Activity activity, OnItemClickListener listener) {
        this.activity = activity;
        this.listener = listener;
        criarMapa();
    }

    @Override
    public void setEditar(boolean value) {
        editar = value;
        notificaAtualizacao();
    }

    @Override
    public void notificaAtualizacao() {
        criarMapa();
        notifyDataSetChanged();
    }

    private void criarMapa() {
        // Obtém a identificação da preferência para Ordenação
        String ordemPreference = "ORDEM_DA_LISTA";
        // Obtém o valor padrão para a Ordenação
        String ordemDefault = "Banda";
        // Obtém o recurso de leitura de preferências
        SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(activity);
        // Localiza a configuração selecionada para Ordenação de Albuns
        String ordem = preferences.getString(ordemPreference, ordemDefault);

        mapa = new SparseLongArray();
        List<Long> ids = dao.listarIds(ordem);
        for (int linha = 0; linha < ids.size(); linha++) {
            mapa.put(linha, ids.get(linha));
        }
    }

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

    @Override
    public AlbumViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        LayoutInflater svc = LayoutInflater.from(parent.getContext());
        View layout = svc.inflate(R.layout.adapter_card, parent, attachToRoot: false);
        return new AlbumViewHolder(layout);
    }

    @Override
    public void onBindViewHolder(AlbumViewHolder viewHolder, int linha) {
        Album obj = dao.localizar(mapa.get(linha));
        viewHolder.setView(obj);
    }


    public class AlbumViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {...}
}
```

- Na implementação do exemplo temos o construtor declarando a necessidade dos argumentos: **Activity** e **OnItemClickListener**.

```
public AlbumRecycledAdapter(Activity activity, OnItemClickListener listener) {  
    this.activity = activity;  
    this.listener = listener;  
    criarMapa();  
}
```


- A **Activity** é utilizada para o acesso das preferências:

```
private void criarMapa() {  
    // Obtém a identificação da preferência para Ordenação  
    String ordemPreference = "ORDEM_DA_LISTA";  
    // Obtém o valor padrão para a Ordenação  
    String ordemDefault = "Banda";  
    // Obtém o recurso de leitura de preferências  
    SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(activity);  
    // Localiza a configuração selecionada para Ordenação de Albuns  
    String ordem = preferences.getString(ordemPreference, ordemDefault);  
  
    mapa = new SparseLongArray();  
    List<Long> ids = dao.listarIds(ordem);  
    for (int linha = 0; linha < ids.size(); linha++) {  
        mapa.put(linha, ids.get(linha));  
    }  
}
```



- A interface **OnItemClickListener** foi criada para possibilitar a **MainActivity** receber o evento de *click* num cartão que seja selecionado.

```
@Override  
public void onClick(View view) {  
    Long id = (Long)view.getTag();  
    if(view instanceof CheckBox) {  
        Album album = dao.localizar(id);  
        album.setDel(!album.isDel());  
        dao.salvar(album);  
    } else {  
        listener.onItemClick(id);  
    }  
}
```



- O **RecyclerView** necessita utilizar uma classe separada para manter a construção da **View**, esta classe deve utilizar **RecyclerView.ViewHolder** em sua implementação.

```
public class AlbumViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {
    private ImageView capa;
    private TextView banda;
    private TextView album;
    private CheckBox apagar;
    private View view;

    public AlbumViewHolder(View itemView) {
        super(itemView);
        this.view = itemView;
        banda = itemView.findViewById(R.id.tvBanda);
        album = itemView.findViewById(R.id.tvAlbum);
        capa = itemView.findViewById(R.id.ivFoto);
        apagar = itemView.findViewById(R.id.checkBox);
    }

    public void setView(final Album obj) {
        banda.setText(obj.getBanda());
        album.setText(obj.getAlbum());

        byte[] foto = obj.getCapa();
        if(foto != null) {
            // Transforma o vetor de bytes de base64 para bitmap
            Bitmap bitmap = Utilitarios.bitmapFromBase64(foto);
            // Cria uma foto circular e atribui à foto
            capa.setImageBitmap(bitmap);
        } else {
            // Obtem a 1ª letra do nome da pessoa e converte para Maiuscula
            String letra = obj.getBanda().substring(0, 1).toUpperCase();
            // Cria um bitmap contendo a letra
            Bitmap bitmap = Utilitarios.circularBitmapAndText(
                Color.parseColor("colorString: "#936A4D"), width: 200, height: 200, letra);
            // atribui à foto
            capa.setBackgroundColor(Color.TRANSPARENT);
            capa.setImageBitmap(bitmap);
        }

        apagar.setTag(obj.getId());
        apagar.setChecked(obj.isDel());
        apagar.setOnClickListener(this);

        if(editar) {
            apagar.setVisibility(View.VISIBLE);
        } else {
            apagar.setVisibility(View.INVISIBLE);
        }

        Log.d("tag: "AlbumViewHolder", msg: "album: " + obj);

        view.setTag(obj.getId());
        view.setOnClickListener(this);
    }

    @Override
    public void onClick(View view) {
        Long id = (Long)view.getTag();
        if(view instanceof CheckBox) {
            Album album = dao.localizar(id);
            album.setDel(!album.isDel());
            dao.salvar(album);
        } else {
            listener.onItemClick(id);
        }
    }
}
```



- Uma instancia do **RecyclerView** é criada e configurada na classe **MainActivity**, porém agora existirão duas formas diferentes para apresentar os Álbuns Musicais: um no **ListView** e outro no **RecyclerView**, assim para simplificar a forma como a interação entre a **MainActivity** e as listas deveremos tratar ambas de uma forma comum e para isto deveremos utilizar uma **interface** para definir todos os métodos que serão utilizados nesta comunicação. Esta interface será chamada de **AdapterInterface**.

```
public interface AdapterInterface {
    public void setEditor(boolean value);
    public void notificaAtualizacao();
}
```

- Uma vez criada a interface **AdapterInterface**, As classes **AlbumAdapter** e **AlbumRecycledAdapter** deverão implementar esta interface.

```
public class AlbumAdapter
    extends BaseAdapter
    implements View.OnClickListener, AdapterInterface {
```

```
public class AlbumRecycledAdapter
    extends RecyclerView.Adapter<AlbumRecycledAdapter.AlbumViewHolder>
    implement AdapterInterface {
```

- Os métodos implementados em ambas as classes **AlbumAdapter** e **AlbumRecycledAdapter** são idênticos.

```
@Override
public void setEditor(boolean value) {
    editor = value;
    notificaAtualizacao();
}

@Override
public void notificaAtualizacao() {
    criarMapa();
    notifyDataSetChanged();
}
```


- Além desta interface será necessário a criação de outra para tratar o evento de *click* nos cartões do **RecyclerView**, esta interface será chamada de **OnItemClickListener**.

```
public interface OnItemClickListener {
    void onItemClick(long id);
}
```

- Uma vez criada a interface **OnItemClickListener**, A classe **MainActivity** deverá implementar esta interface.

```
public class MainActivity
    extends AppCompatActivity
    implements AdapterView.OnItemClickListener, View.OnClickListener,
        NavigationView.OnNavigationItemSelectedListener,
        OnItemClickListener {
```

- O método implementado na classe **MainActivity** será utilizado para iniciar a *activity* **EditActivity** que apresenta o detalhe do Álbum Musical.



```
@Override
public void onItemClick(long id) {
    editAlbum(id);
}

private void editAlbum(long id) {
    Intent tela = new Intent(getBaseContext(), EditActivity.class);
    tela.putExtra( name: "id", id);
    startActivityForResult(tela, EDIT_ACTION);
}
```

- Para a inicialização do **ListView** e do **RecyclerView** será necessário o ajuste dos seguintes atributos na classe **MainActivity**.

```
private ListView listView;
private RecyclerView recyclerView;

private AdapterInterface adapter;
private AlbumAdapter albumAdapter;
private AlbumRecycledAdapter albumRecycledAdapter;

private RecyclerView.LayoutManager layoutManager3;
private RecyclerView.LayoutManager layoutManager5;
```

- Também será necessário a alteração no método **onCreate** na classe **MainActivity**.

```
// Inicializa o ListView
albumAdapter = new AlbumAdapter( activity: this);

listView = findViewById(R.id.listView);
listView.setOnItemClickListener(this);

// Inicializa o RecyclerView
albumRecycledAdapter = new AlbumRecycledAdapter( activity: this, listener: this);

recyclerView = (RecyclerView) findViewById(R.id.recycleView);
layoutManager3 = new GridLayoutManager( context: this, spanCount: 3);
layoutManager5 = new GridLayoutManager( context: this, spanCount: 5);

recyclerView.setHasFixedSize(true);
setGridOrientation(recyclerView);
```

- Devem ser acrescentados os métodos **onConfigurationChanged** e **setGridOrientation** na classe **MainActivity**, para tratar do evento de mudança da orientação do telefone de *Portrait* para *Landscape* e vice-versa.

```
private void setGridOrientation(RecyclerView recyclerView) {
    int orientacao = getResources().getConfiguration().orientation;
    if(orientacao == Configuration.ORIENTATION_PORTRAIT) {
        recyclerView.setLayoutManager(layoutManager3);
    } else {
        recyclerView.setLayoutManager(layoutManager5);
    }
}

@Override
public void onConfigurationChanged(Configuration newConfig) {
    super.onConfigurationChanged(newConfig);
    if(adapter instanceof RecyclerView)
        setGridOrientation(recyclerView);
}
```

- Por fim, o método **onStart** da classe **MainActivity** deverá ser modificado para poder identificar a alteração do modo de apresentação da lista pelo usuário.

```
// Obtém a identificação da preferência para Ordenação
String listaPreference = "TIPO_DA_LISTA";
// Obtém o valor padrão para a Ordenação
String listaDefault = "Em Lista";
// Localiza a configuração selecionada para Ordenação de Albuns
String tipo = preferences.getString(listaPreference, listaDefault);

if(adapter != null)
    resetEditor();

if(tipo.equals(listaDefault)) {
    listView.setVisibility(View.VISIBLE);
    listView.setAdapter(albumAdapter);

    recyclerView.setVisibility(View.INVISIBLE);
    recyclerView.setAdapter(null);

    adapter = albumAdapter;
} else {
    listView.setVisibility(View.INVISIBLE);
    listView.setAdapter(null);

    recyclerView.setVisibility(View.VISIBLE);
    recyclerView.setAdapter(albumRecycledAdapter);

    adapter = albumRecycledAdapter;
}
adapter.notificaAtualizacao();
```