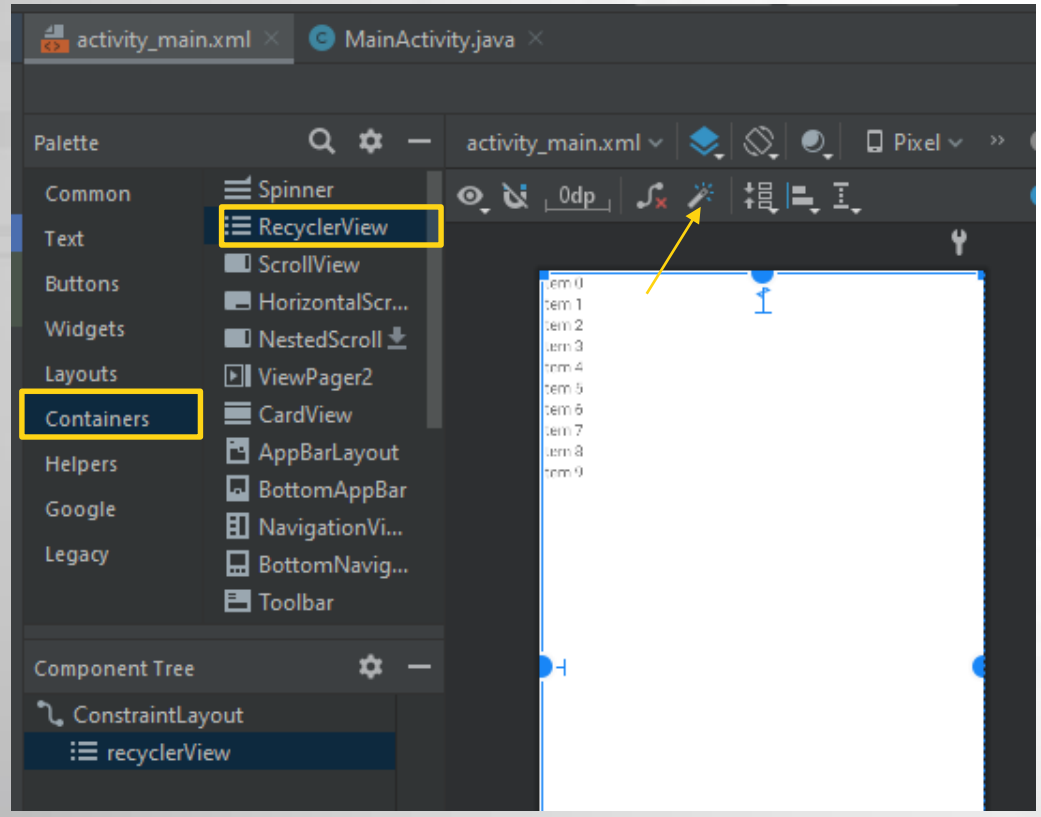
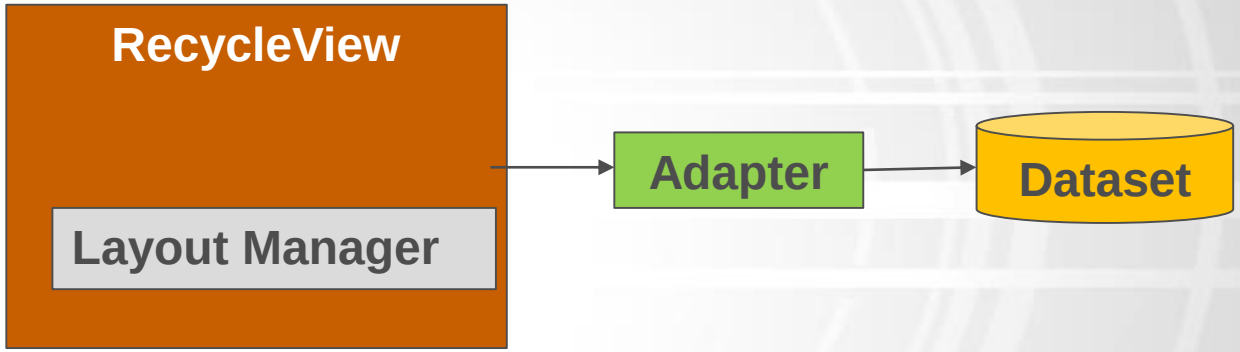
Listagem com RecycleView

Hercules Santos

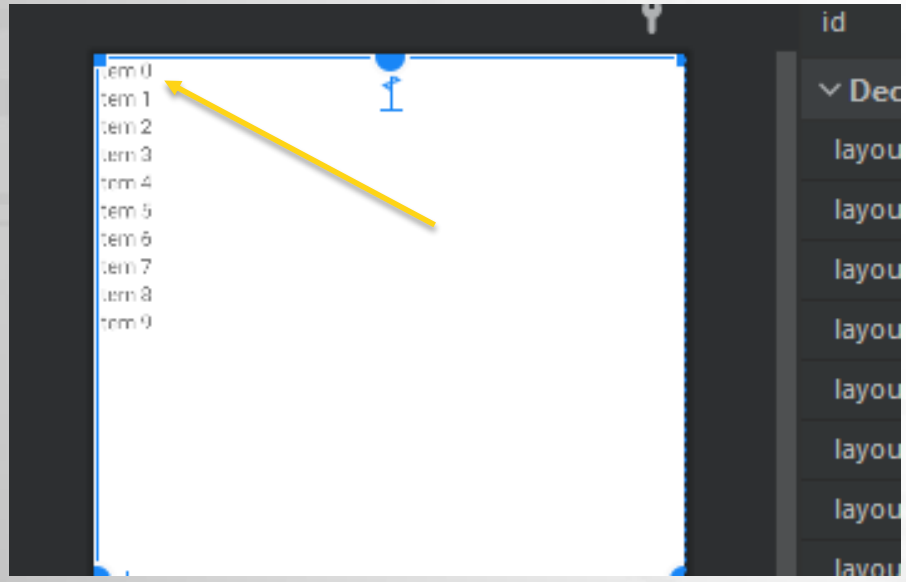
2023.1






```
public class MainActivity extends AppCompatActivity {

    RecyclerView recicle;

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

        recicle = findViewById(R.id.recyclerView);

        //configurar adapter

        //configurar RecycleView

    }
}
```

https://developer.android.com/guide/topics/ui/layout/recyclerview?hl=pt-br#java



O **Adapter** é quem recebe os dados, formatar o layoute e utilizar No **RecyclerView**

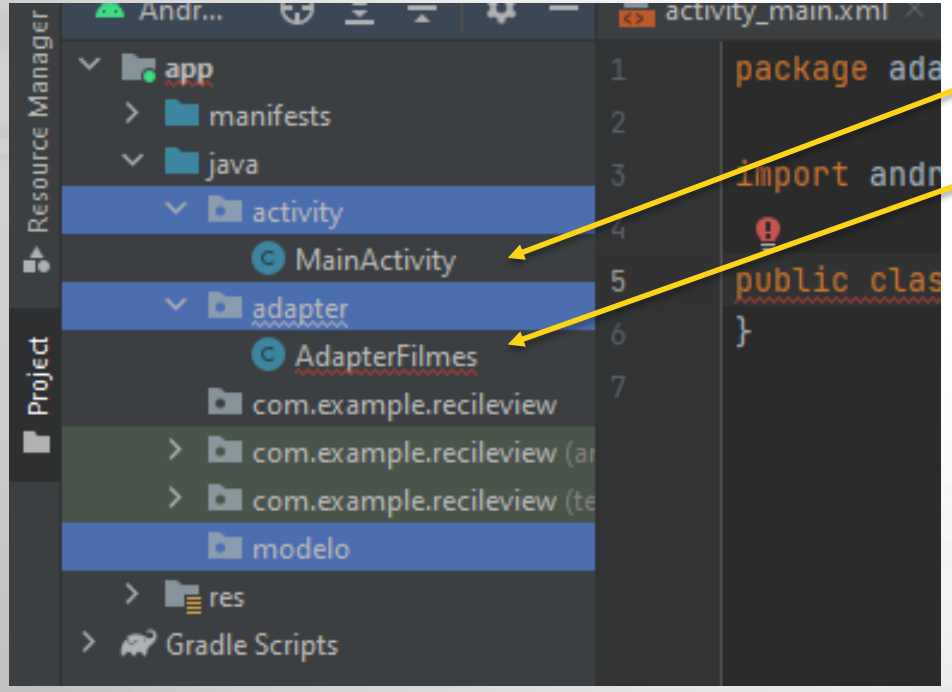
O RecyclerView cria cada um dos itens
Cada item pode ter um layoute e formatos diferentes

O **Adapter** retorna o formato que você quer exibir, linha a linha.

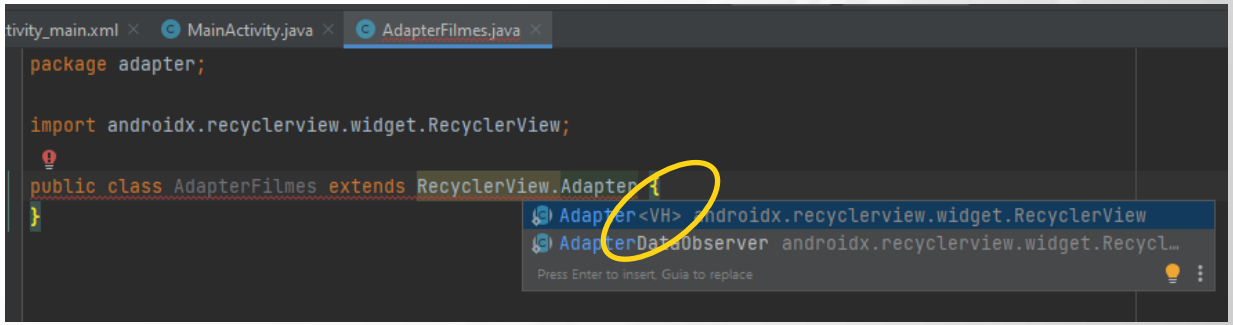
```
public class MainActivity extends AppCompatActivity {

    RecyclerView recicle;

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

        recicle = findViewById(R.id.recyclerView);

        //configurar adapter

        //configurar RecycleView
        RecyclerView.LayoutManager meuLayout = new LinearLayoutManager(getApplicationContext());
        recicle.setLayoutManager(meuLayout);
        recicle.setHasFixedSize(true);//tamanho fixo
    }
}
```

Criar um Adapter

Dentro de **java**, crie os seguintes package: **activity**, **adapter**, **modelo**



- Mova, MainActivity para dentro de **activity**
- Crie uma classe AdapterFimes, detro de **adapter**



VH – View Holder - descreve uma exibição de item e metadados sobre seu local no RecyclerView

Cria visualizações disponíveis e reaproveita, mudando apenas os dados.

VH – View Holder

Vamos criar uma nova **classe** para criar os itens da lista: Titulo, nome, data e etc.
Para isso vamos criar uma classe, dentro da classe AdapterFilmes.

```
public class AdapterFilmes extends RecyclerView.Adapter<> {

    public class MeuViewHoldeer extends RecyclerView.ViewHolder{
    }
}
```

```
public class AdapterFilmes extends RecyclerView.Adapter<> {

    public class MyViewHoldeer extends RecyclerView.ViewHolder{
```

Create constructor matching super

Press Ctrl+Q to open preview

```
package adapter;

import android.view.View;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

public class AdapterFilmes extends RecyclerView.Adapter<> {

    public class MeuViewHoldeer extends RecyclerView.ViewHolder{
        public MeuViewHoldeer(@NonNull View itemView) {
            super(itemView);
        }
    }
}
```

```
package adapter;

import android.view.View;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

public class AdapterFilmes extends RecyclerView.Adapter<AdapterFilmes.MeuViewHoldeer> {
    public class MeuViewHoldeer extends RecyclerView.ViewHolder{
        TextView titulo;
        TextView genero;
        TextView ano;
        public MeuViewHoldeer(@NonNull View itemView) {
            super(itemView);
        }
    }
}
```

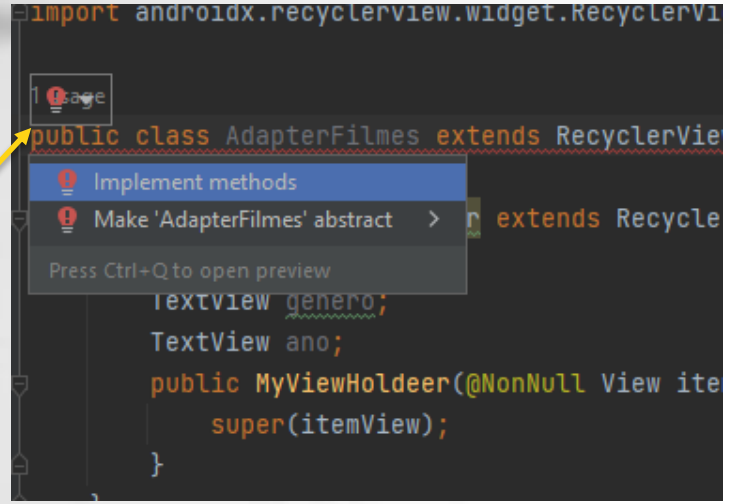
Agora você pode utilizar essa classe que acabou de criar.

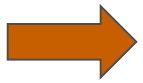
<AdapterFilmes.MeuViewHoldeer>

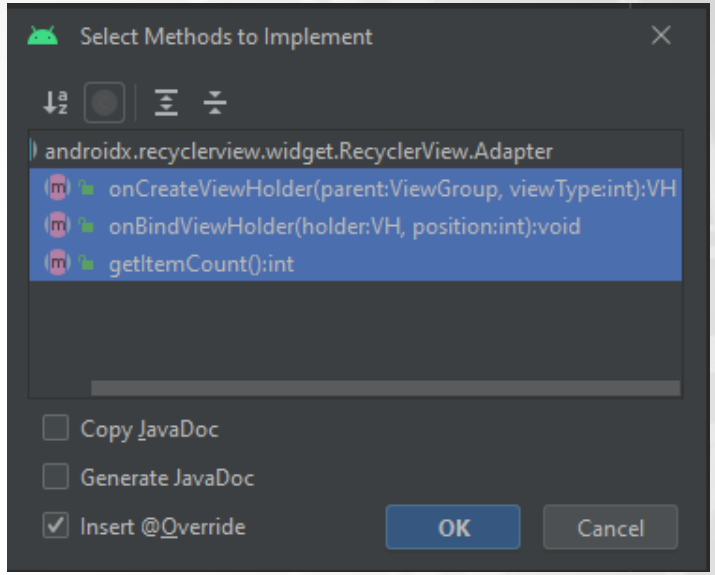
Essa classe será responsável de guardar os dados antes dos dados serem exibidos na tela.

Podemos definir os dados a serem guardados:

- Nome do Filme
- Genero
- Data lançamento







Agora vamos implementar os métodos desta classe, para isso, clique na lâmpada e em seguida selecione os três métodos.

```
@Override
public MeuViewHoldeer onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    return null;
}
@Override
public void onBindViewHolder(@NonNull MeuViewHoldeer holder, int position) {

}
@Override
public int getItemCount() {
    return 0;
}
```

```
@Override
public MeuViewHoldeer onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    return null;
}
@Override
public void onBindViewHolder(@NonNull MeuViewHoldeer holder, int position) {

}
@Override
public int getItemCount() {
    return 0;
}
```

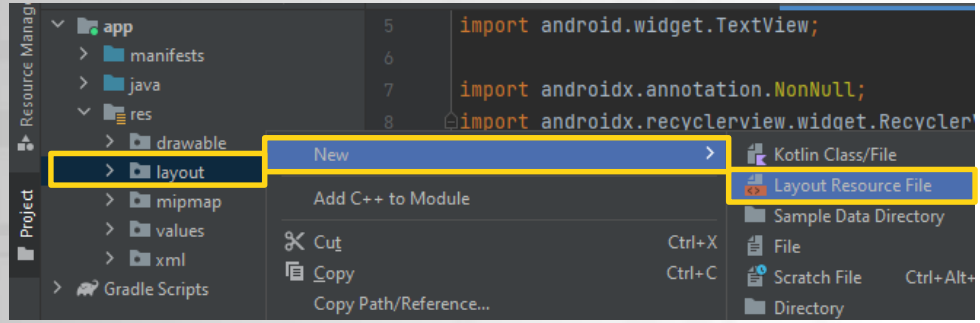
onCreateViewHolder
Chamado para criar uma nova exibição, que define a interface do usuário do item da lista

onBindViewHolder
Exibe cada elemento do seu conjunto de dados e substitue o conteúdo da exibição por outro elemento

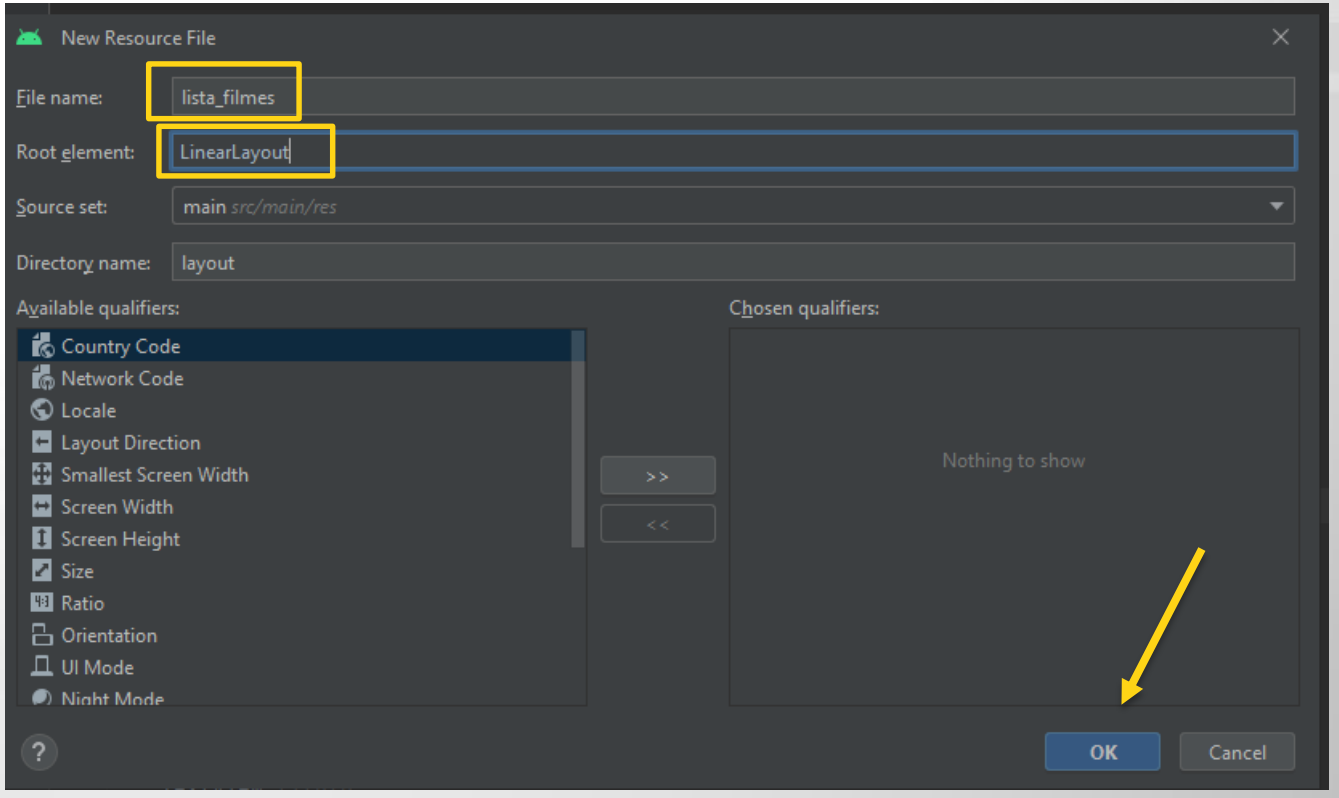
getItemCount
retorna a quantidade de Itens que serão exibidos na minha lista

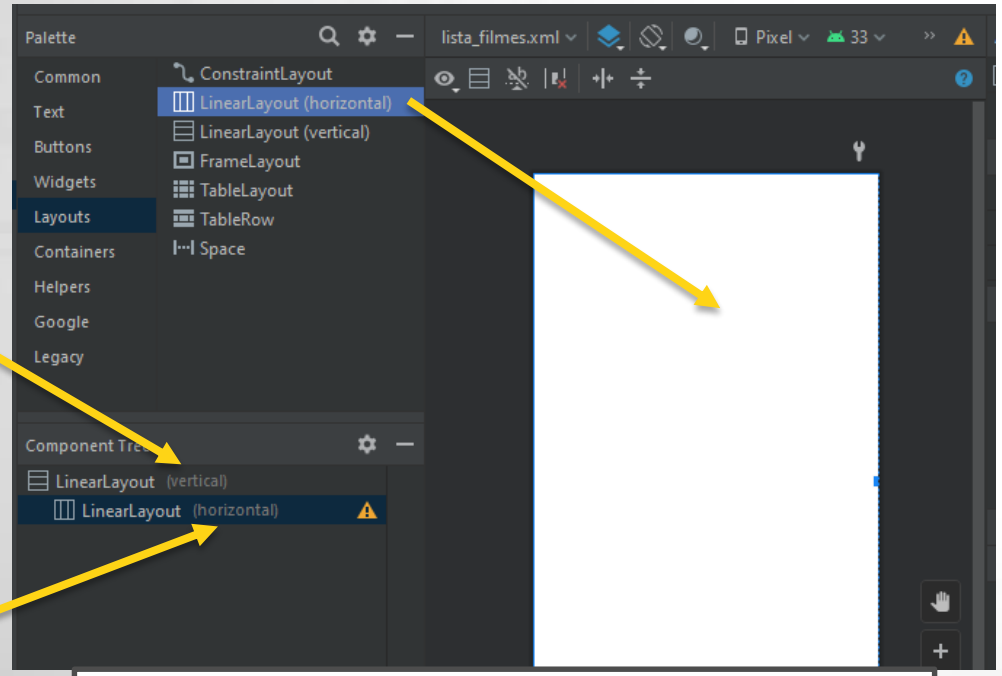
O layout de cada item de visualização é definido em um arquivo de layout XML

Vamos criar o layout XML



→ res\layout\New**Layout Resource File**

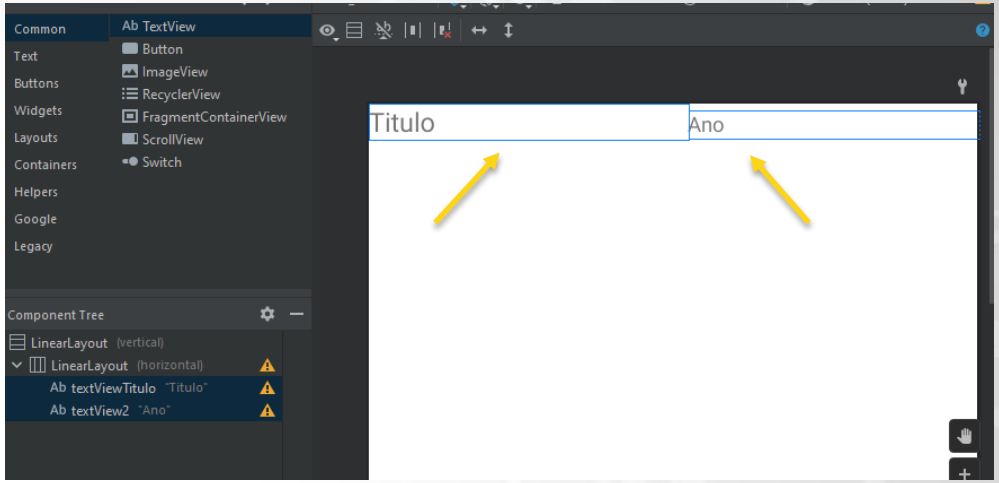




Vertical

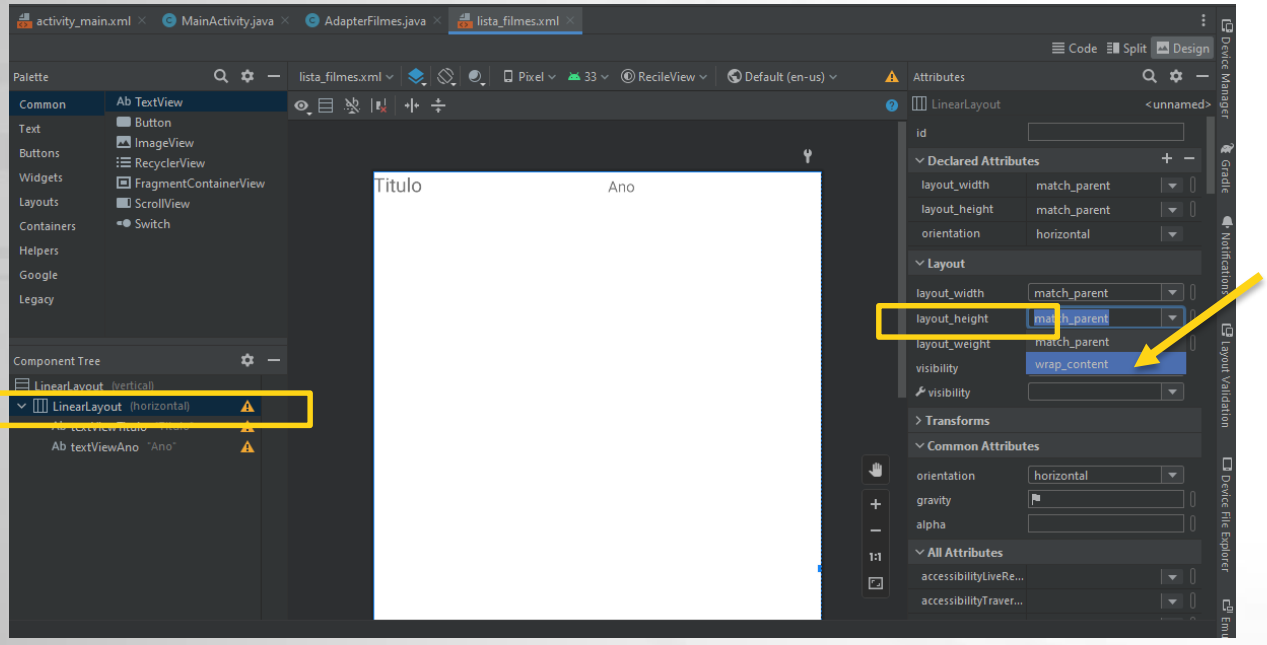
Horizontal

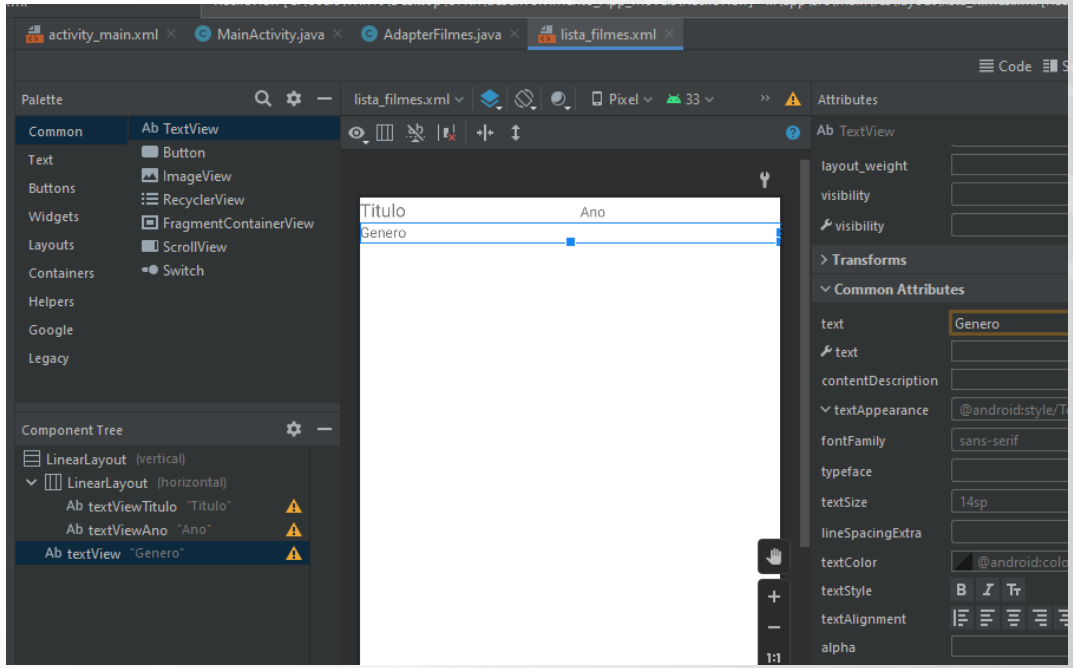
Adicione um LinearLayout (Horizontal)




```
<TextView
    android:id="@+id/textViewTitulo"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="5"
    android:text="Titulo"
    android:textSize="18sp" />

<TextView
    android:id="@+id/textViewAno"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="Ano" />
```

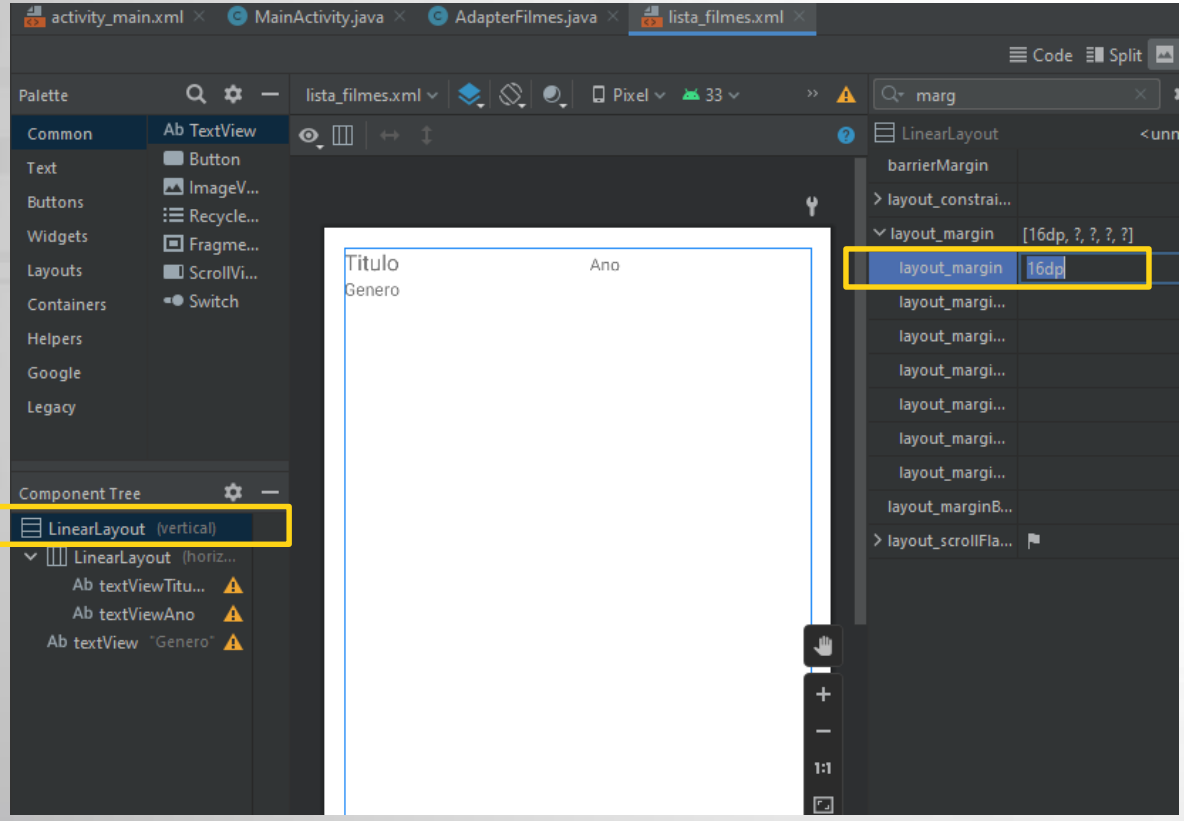
Adicione duas TextView com as seguintes configurações



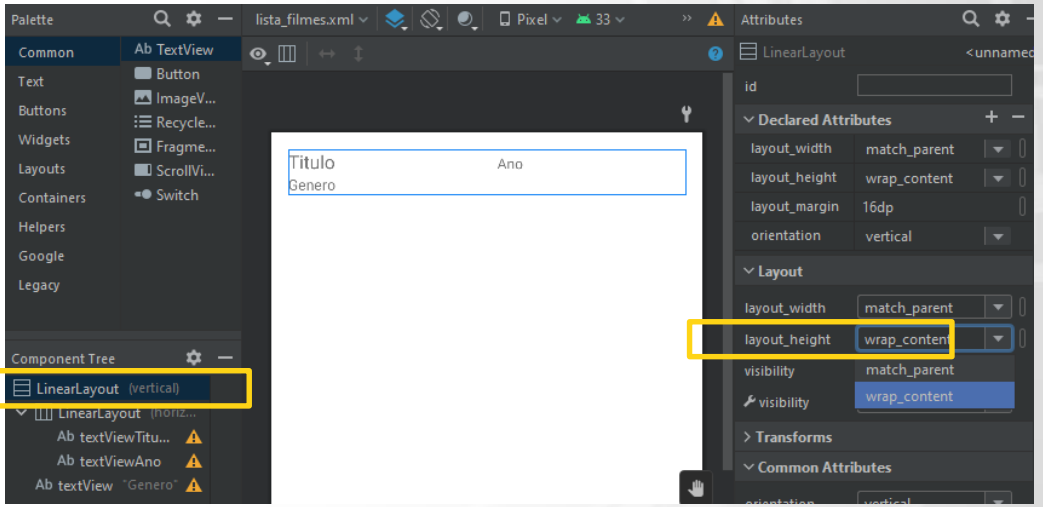


No componente LinearLayout, altere o Layout height para wrap_contente.

Em seguida adicione mais um TextView, Gênero



Altere as margens do LinearLayout para 16dp



No componente LinearLayout, altere o Layout height para wrap_contente.

AdapterFilmes

A classe OnCreate retorna um tipo, chamado de MeuViiewHolder que é o nome da classe criada logo a baixo.

```
1 usage
public class AdapterFilmes extends RecyclerView.Adapter<AdapterFilmes.MeuViewHoldeer> {
    @NonNull
    @Override
    public MeuViewHoldeer onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        return null;
    }
    @Override
    public void onBindViewHolder(@NonNull MeuViewHoldeer holder, int position) {

    }
    @Override
    public int getItemCount() { return 0; }

    3 usages
    public class MeuViewHoldeer extends RecyclerView.ViewHolder{
        TextView titulo;
        TextView genero;
        TextView ano;
        public MeuViewHoldeer(@NonNull View itemView) { super(itemView); }
    }
}
```

Perceba que é nesta classe que estão os dados da View: Titulo,gênero e ano.

Para que possamos chamar esse objeto dentro de OnCreat, primeiramente temos que converter o XML (AdapterFIlmes) em um objeto do tipo **View**.

```
public MeuViewHoldeer onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    View itensLista = LayoutInflater.from(parent.getContext()).
            inflate(R.layout.lista_filmes, parent, false);

    return new MeuViewHoldeer(itensLista);
}
```

itemLista: Nome do Objeto criado
LayoutInflater: converte o XML em uma visualização
from: recebe o Context. Para recuperar o contexto baseado no **itemLista**
parente: Componente em que intemLista está dentro
Inflater: recebe três parâmetros (XML, ViewGroup, viewType)

Implementando o método MeuViewHolder

```
public class MeuViewHoldeer extends RecyclerView.ViewHolder{
    TextView titulo;
    TextView genero;
    TextView ano;
    public MeuViewHoldeer(@NonNull View itemView) {

        super(itemView);
        titulo = itemView.findViewById(R.id.textViewTitulo);
        genero = itemView.findViewById(R.id.textViewGenero);
        ano = itemView.findViewById(R.id.textViewAno);
    }
}
```

Aqui você vai criar os objetos dentro do método construtor da classe MeuViewHolder.

Aqui você irá implementar os objetos criados anteriormente.

```
public void onBindViewHolder(@NonNull MeuViewHoldeer holder, int position) {

    holder.titulo.setText("Titulo Teste");
    holder.genero.setText("Comédia");
    holder.ano.setText("2023");
}

@Override
public int getItemCount() {return 5;}
```

Quantidade de itens a serem exibidos

MainActivity

activity_main.xml | MainActivity.java | AdapterFilmes.java | lista_filmes.xml

```
public class MainActivity extends AppCompatActivity {

    RecyclerView recicle;

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

        recicle = findViewById(R.id.recyclerView);

        //configurar adapter
        AdapterFilmes adapter = new AdapterFilmes();

        //configurar RecycleView
        RecyclerView.LayoutManager mlayoutManager = new LinearLayoutManager(getApplicationContext());
        recicle.setLayoutManager(mlayoutManager);
        recicle.setHasFixedSize(true);//tamnho fixo
        recicle.setAdapter(adapter);
    }
}
```

Agora podemos voltar a MainActivity e configurar o AdapterFilmes que acabamos de criar.

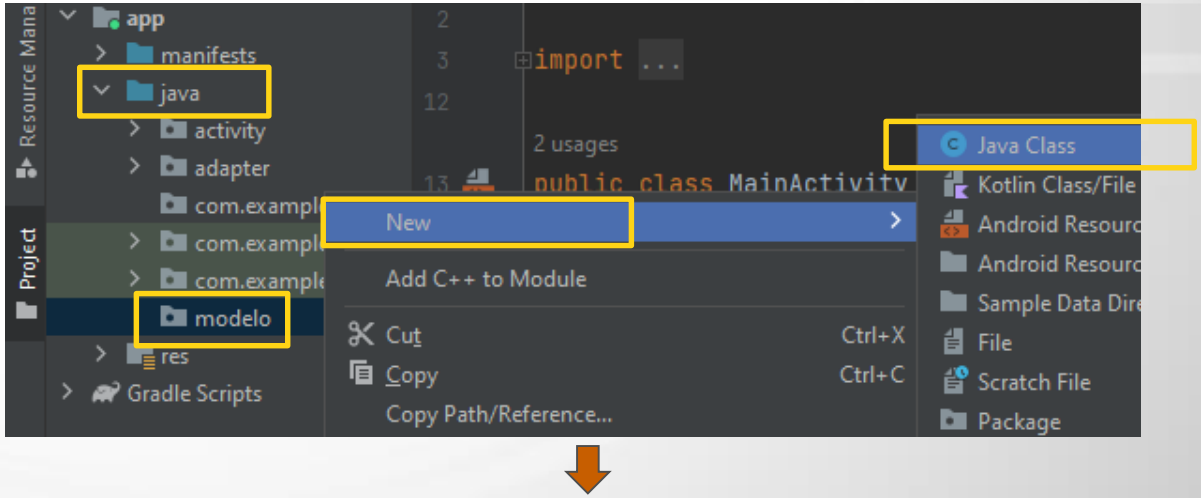
Instanciar dentro de Oncreadte e em seguida
Setar em **recicle**.

Agora você pode executar para visualizar o resultado.

Criando Lista Dinâmica

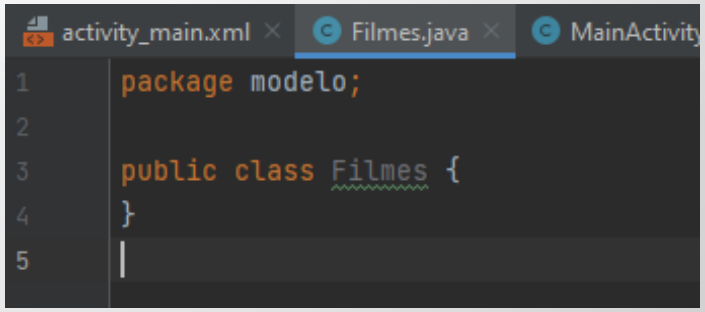
```
public void onBindViewHolder(@NonNull MeuViewHoldeer holder, int position) {
    holder.titulo.setText("Titulo Teste");
    holder.genero.setText("Comédia");
    holder.ano.setText("2023");
}
@Override
public int getItemCount() {return 5;
}
```

Para isso vamos criar um array list e dentro dele colocar os objetos filme.
Vamos em dentro do packege modelo e criamos uma classe filme como modelo para modeloar o nosso sistema.



Para criar uma lista de de itens precisamos passar uma lista dentro do AdapterFilmes como parametr. Que fica no MainActivity.

```
AdapterFilmes adapter = new AdapterFilmes(      );
```



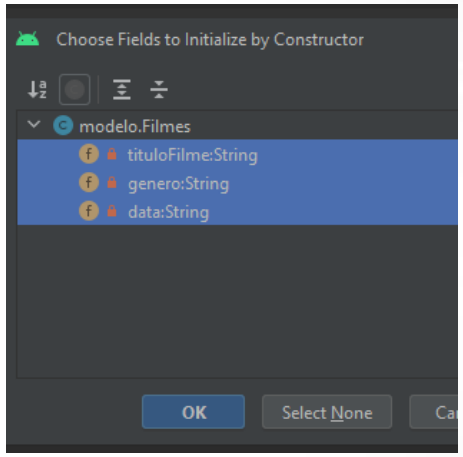
Criando o método Construtor

```
package modelo;

public class Filmes {
    private String tituloFilme;
    private String genero;
    private String data;

}
```


Click com o botão direito e escolha **Generate**

```
package modelo;

public class Filmes {
    private String tituloFilme;
    private String genero;
    private String data;

}
```

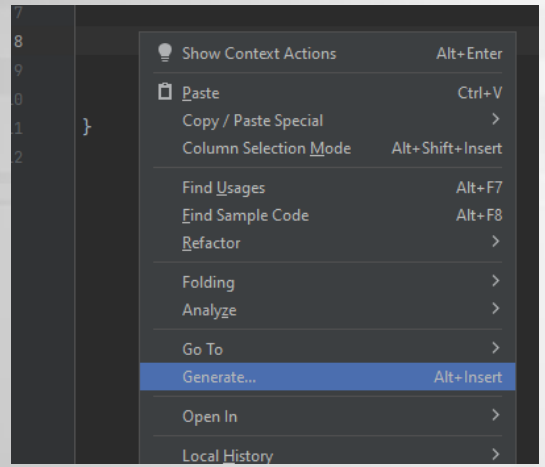

Em seguida escolha **Constructor**



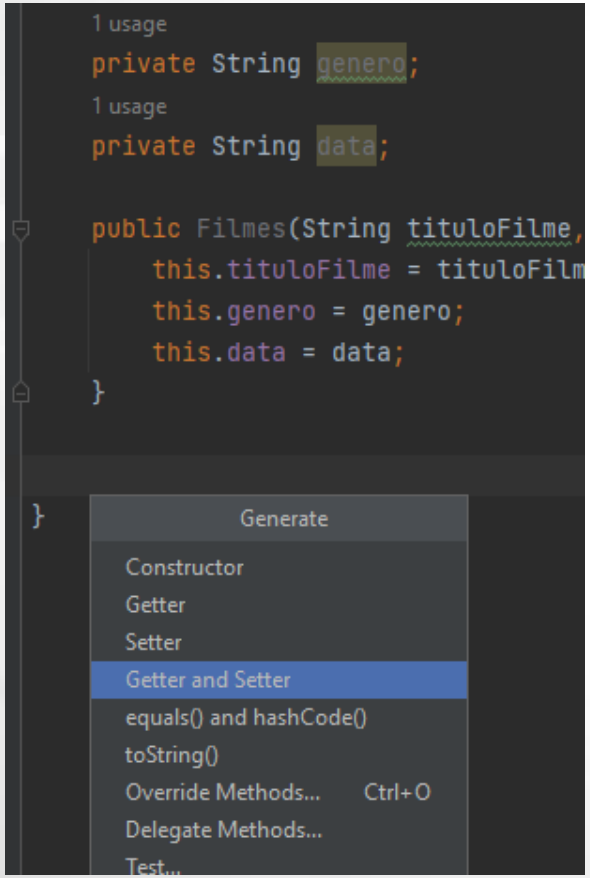
Selecione os três **itens** e click em **ok**

```
public Filmes(String tituloFilme, String genero, String data) {

    this.tituloFilme = tituloFilme;
    this.genero = genero;
    this.data = data;

}
```

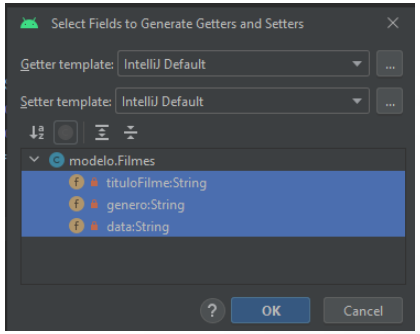
Criando os sets e gets



Click novamente com o botão direito e escolha **Generate**



Em seguida escolha **Getter and Setter**



Selecione os três **itens** e click em **ok**

```
public String getTituloFilme() {
    return tituloFilme;
}

public void setTituloFilme(String tituloFilme) {
    this.tituloFilme = tituloFilme;
}

public String getGenero() {
    return genero;
}

public void setGenero(String genero) {
    this.genero = genero;
}

public String getData() {
    return data;
}

public void setData(String data) {
    this.data = data;
}
```

Agora os métodos get e set já podem ser implementados para configurar e recuperar os valores dos atributos

Classe modelo

```
package modelo;

public class Filmes {
    private String tituloFilme;
    private String genero;
    private String data;

    public Filmes(String tituloFilme, String genero, String data) {
        this.tituloFilme = tituloFilme;
        this.genero = genero;
        this.data = data;
    }

    public String getTituloFilme() {
        return tituloFilme;
    }
    public void setTituloFilme(String tituloFilme) {
        this.tituloFilme = tituloFilme;
    }
    public String getGenero() {
        return genero;
    }
    public void setGenero(String genero) {
        this.genero = genero;
    }
    public String getData() {
        return data;
    }
    public void setData(String data) {
        this.data = data;
    }
}
```

Criando a listagem de itens

```
vity_main.xml | MainActivity.java | Filmes.java | AdapterFilmes.java
public class MainActivity extends AppCompatActivity {

    4 usages
    private RecyclerView recicle;
    1 usage
    private List<Filmes> lista_de_filmes = new ArrayList<>();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

Dentro de MainActivity crie uma lista lista_de_filmes para armazenar os dados dos filmes.

```
public void criarFilmes(){

    Filmes filme = new Filmes("titulo", "genero", "2023");
    this.lista_de_filmes.add(filme);

}
```

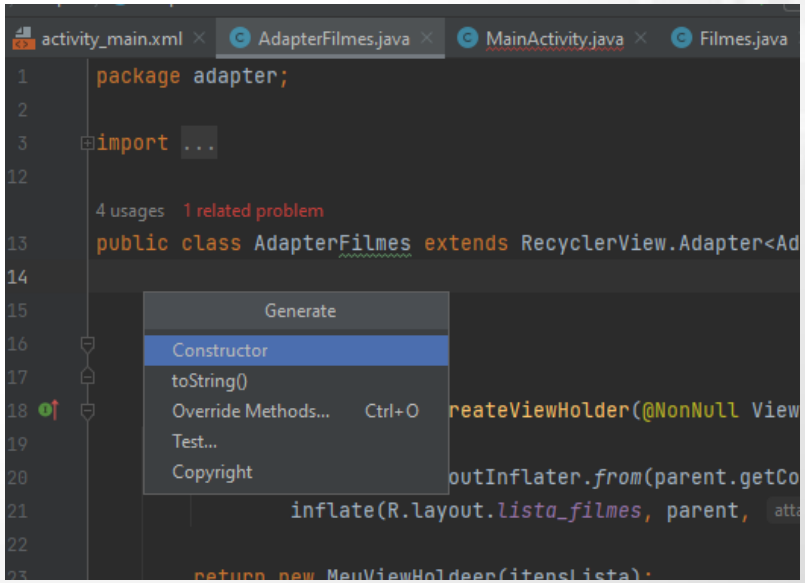
Ainda no MainActivity, crie um método para criar os filmes.

Agora você pode criar quantos filmes quiser, dentro desse método.

```
ain.xml | MainActivity.java | Filmes.java | AdapterFilmes.java | lista_filmes.xml
}

1 usage
public void criarFilmes(){

    Filmes filme = new Filmes( tituloFilme: "Homem Aranha - De Volta ao Lar", genero: "Aventura ", data: "2023");
    this.lista_de_filmes.add(filme);

    filme = new Filmes( tituloFilme: "Mulher Maravilha", genero: "Fantasia ", data: "2023");
    this.lista_de_filmes.add(filme);

    filme = new Filmes( tituloFilme: "Capitão américa - Guerra Cilvil", genero: "Ficção", data: "2023");
    this.lista_de_filmes.add(filme);

    filme = new Filmes( tituloFilme: "Wakanda pra Sempre", genero: "Ficção", data: "2023");
    this.lista_de_filmes.add(filme);

    filme = new Filmes( tituloFilme: "Alvo Perfeiro", genero: "Ação", data: "2023");
    this.lista_de_filmes.add(filme);

    filme = new Filmes( tituloFilme: "Assassino a Preço Fixo", genero: "Ação", data: "2023");
    this.lista_de_filmes.add(filme);

    filme = new Filmes( tituloFilme: "Policial em Apuros", genero: "Poicial", data: "2023");
    this.lista_de_filmes.add(filme);
}
```

```
private List<Filmes> lista_de_filmes = new ArrayList<>();

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

    recicle = findViewById(R.id.recyclerView);

    //listar Filmes
    this.criarFilmes();

    //configurar adapter
    AdapterFilmes adapter = new AdapterFilmes(lista_de_filmes);

    //configurar RecycleView
```

Mas, antes é preciso criar um construtor que permita a classe AdapterFilmes receber um lista de filmes.



Agora podemos chamar o método criarFilmes dentro do método construtor de mainActivity

```
public class AdapterFilmes extends RecyclerView.Adapter<AdapterFilmes.MeuViewHoldeer> {

    public AdapterFilmes(List<Filmes> lista) {

    }
```

Vamos criar um objeto do tipo List<Filmes> que irá receber o parâmetro do construtor do AdapterFilmes

```
public class AdapterFilmes extends RecyclerView.Adapter<AdapterFilmes.MeuViewHoldeer> {

    private List<Filmes> listagemF;

    public AdapterFilmes(List<Filmes> lista) {

        this.listagemF = lista;

    }
```

Agora vamos alterar os métodos onBindViewHolder e getItemCount

```java
@Override
public void onBindViewHolder(@NonNull MeuViewHoldeer holder, int position) {

    holder.titulo.setText("Titulo Teste");
    holder.genero.setText("Comédia");
    holder.ano.setText("2023");
}
@Override
public int getItemCount() {

    return 5;
}
```

Criaremos o objetos filme e passar o parâmetro position.
Em seguida substitua o parâmetro dos setText por esse objeto get*

AdapterFilmes — app — Xiaomi Redmi N

_main.xml | AdapterFilmes.java | MainActivity.java | Filmes.java | lista_filmes.xml

```java
        return new MeuViewHoldeer(itensLista);
    }
    @Override
    public void onBindViewHolder(@NonNull MeuViewHoldeer holder, int position) {
        Filmes filme = listagemF.get(position);

        holder.titulo.setText(filme.getTituloFilme());
        holder.genero.setText(filme.getGenero());
        holder.ano.setText(filme.getData());
    }
    @Override
    public int getItemCount() {
        return listagemF.size();
    }
```

Agora você pode executar seu App

Linha divisória

activity_main.xml | MainActivity.java | AdapterFilmes.java | Filmes.java | lista_filmes.xml

```
30
31          //listar Filmes
32          this.criarFilmes();
33
34          //configurar adapter
35          AdapterFilmes adapter = new AdapterFilmes(lista_de_filmes);
36
37          //configurar RecycleView
38          RecyclerView.LayoutManager mlayoutManager = new LinearLayoutManager(getApplicationContext());
39          recicle.setLayoutManager(mlayoutManager);
40          recicle.setHasFixedSize(true);//tamnho fixo
41          recicle.setAdapter(adapter);
42          recicle.addItemDecoration(new DividerItemDecoration( context: this, LinearLayout.VERTICAL));
43      }
44
        1 usage
        public void criarFilmes(){
```

Ainda no método construtor, você pode adicionar uma linha divisória para a lista de filmes.