

Layout



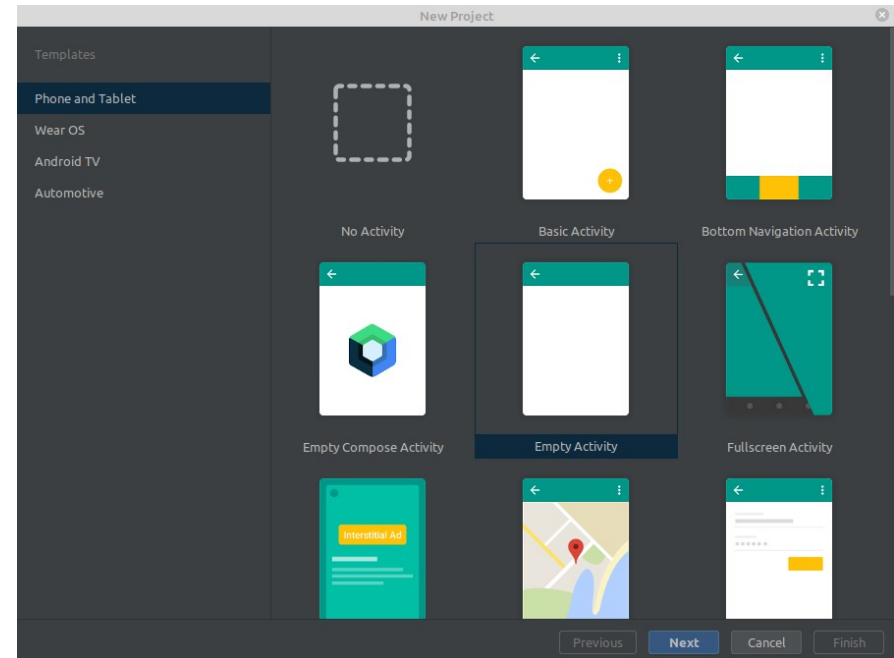
Prof. Dr. João Paulo Lemos Escola
Copyright© 2022

Conteúdo

- Layouts no Android Studio;
- Ferramentas para layout;
- XML;
- Tipos de layout;

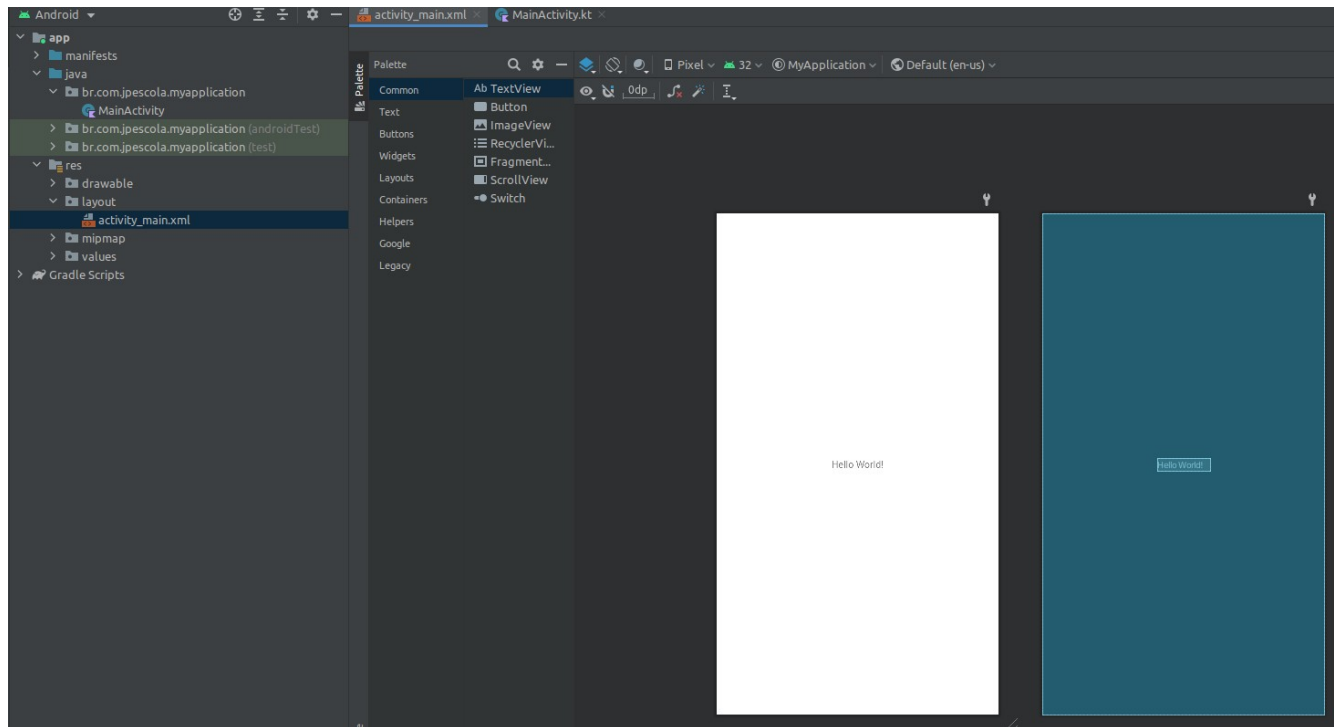
Novo projeto

- Vamos criar um novo projeto;
- Escolha “Empty Activity”.



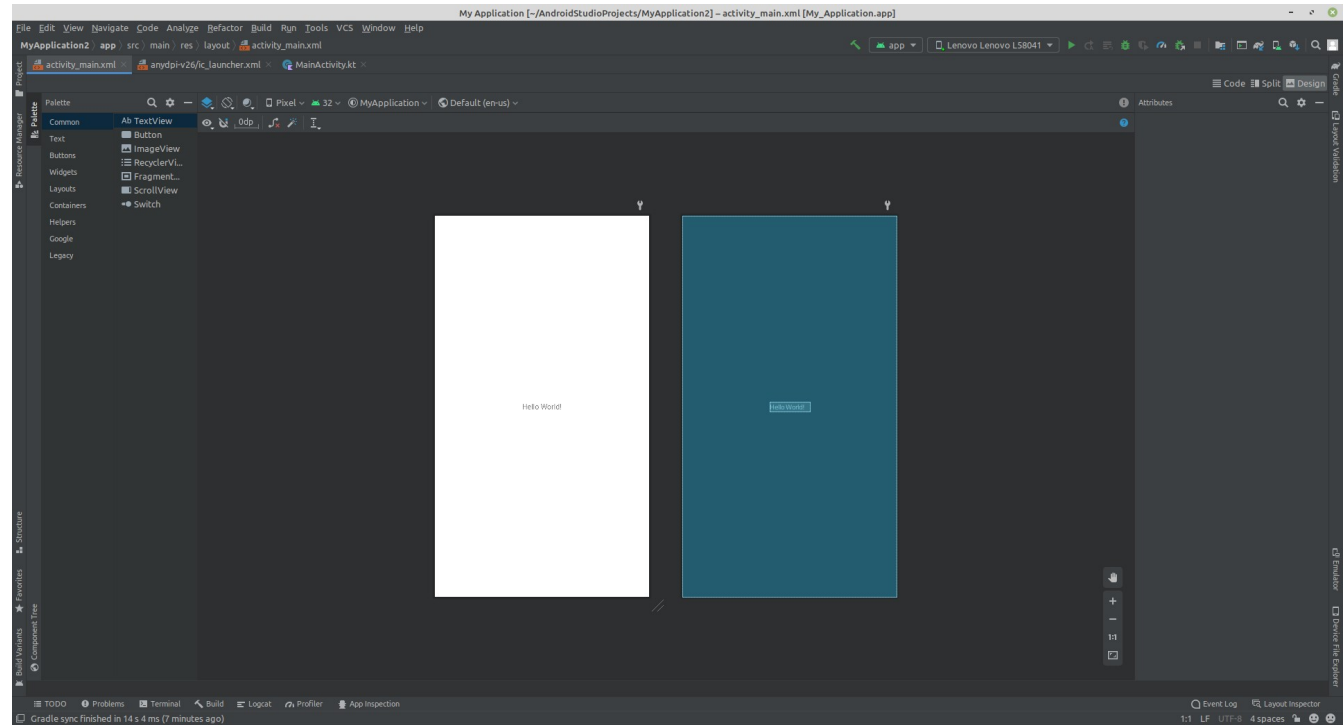
activity_main.xml

- Dê um duplo clique na pasta “res”, depois na pasta “layout” e por último no arquivo “activity_main.xml”;
- Esse arquivo armazena o código XML da activity principal da nossa aplicação;
- Por padrão, cada arquivo da pasta “layout” representa uma tela da aplicação;
- Normalmente, cada um desses arquivos está vinculado a uma classe do tipo Activity, representando uma “atividade”, uma tela, da nossa aplicação;
- Estudaremos activities no futuro.



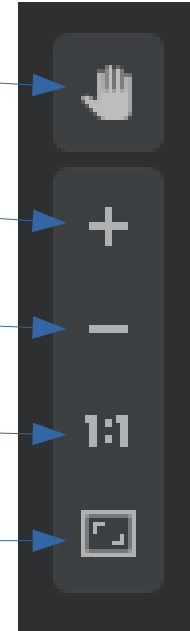
Maximizando a interface

- Aqui temos a tela de design de UI (user interface);
- Para maximizar a janela dentro da IDE, dê um duplo clique na aba superior que tem o nome do arquivo.



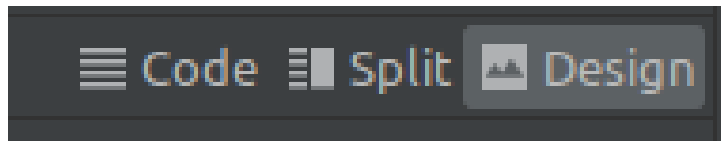
Ajuste da área de design

- Barra no canto inferior direito da tela;
- Pan (mover design);
- Aumentar zoom;
- Diminuir zoom;
- Tamanho máximo;
- Ajustar à tela (melhor opção);

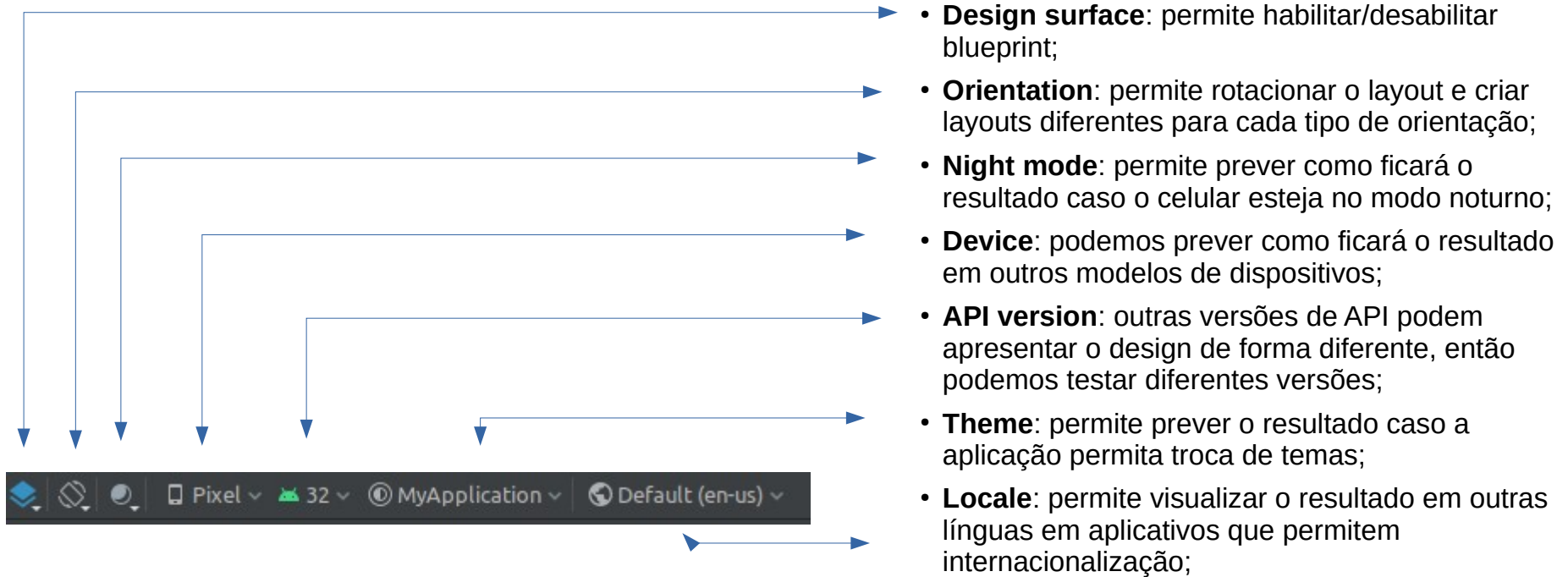


Alternar entre design e código

- Barra no canto superior direito;
- **Code**: mostra o código fonte XML gerado automaticamente;
- **Split**: mostra código e design lado a lado;
- **Design** (padrão): mostra somente o design.

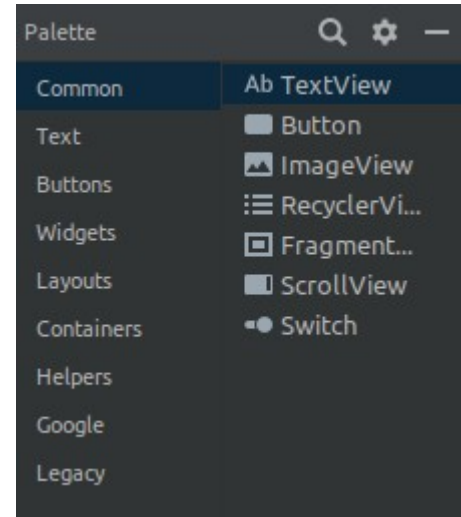


Barra de ferramentas



Paleta

- Componentes da UI, como botões, rótulos, imagens etc;
 - **Common**: componentes mais utilizados;
 - **Text**: componentes especializados para múltiplos tipos de entrada de texto;
 - **Buttons**: componentes de botões diversos, incluindo radio, check e switch;
 - **Widgets**: componentes avançados, como calendário, rating e vídeo;
 - **Layouts**: componentes muito importantes que permitem organizar os objetos dentro da tela de forma profissional;
 - **Containers**: permitem agrupar componentes;
 - **Helpers**: permitem organizar componentes;
 - **Google**: componentes vinculados à Google, como mapas e AD;
 - **Legacy**: componentes descontinuados.



Código XML

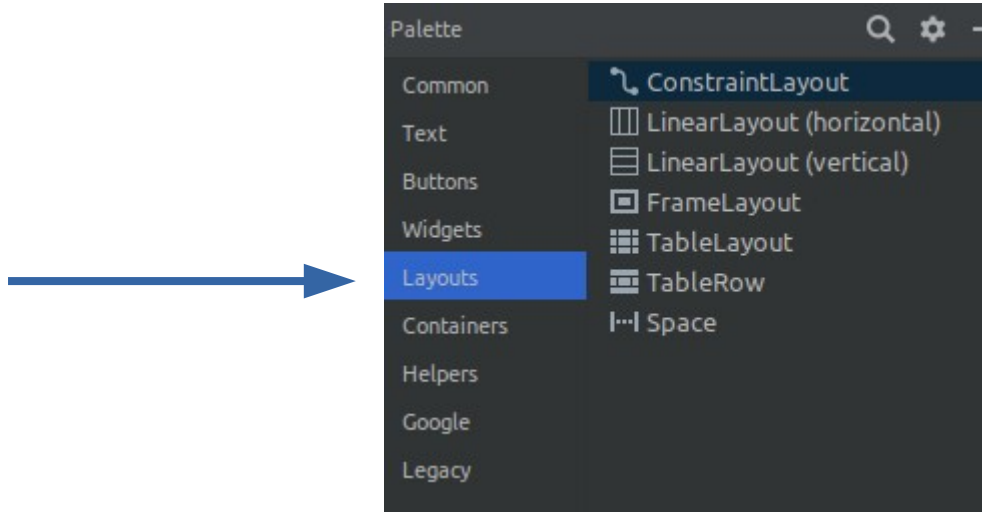
- Esta tela utiliza o ConstraintLayout;
- Largura e altura ajustadas ao componente pai (match_parent);
- Tela contém um único TextView (rótulo):
 - Largura e altura ajustadas ao conteúdo (wrap_content);
 - Texto do rótulo;
 - Margens nos quatro lados ajustadas ao componente pai (parent).

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res-auto"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MainActivity">
8
9      <TextView
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:text="Hello World!"
13         app:layout_constraintBottom_toBottomOf="parent"
14         app:layout_constraintLeft_toLeftOf="parent"
15         app:layout_constraintRight_toRightOf="parent"
16         app:layout_constraintTop_toTopOf="parent" />
17
18  </androidx.constraintlayout.widget.ConstraintLayout>
```

View

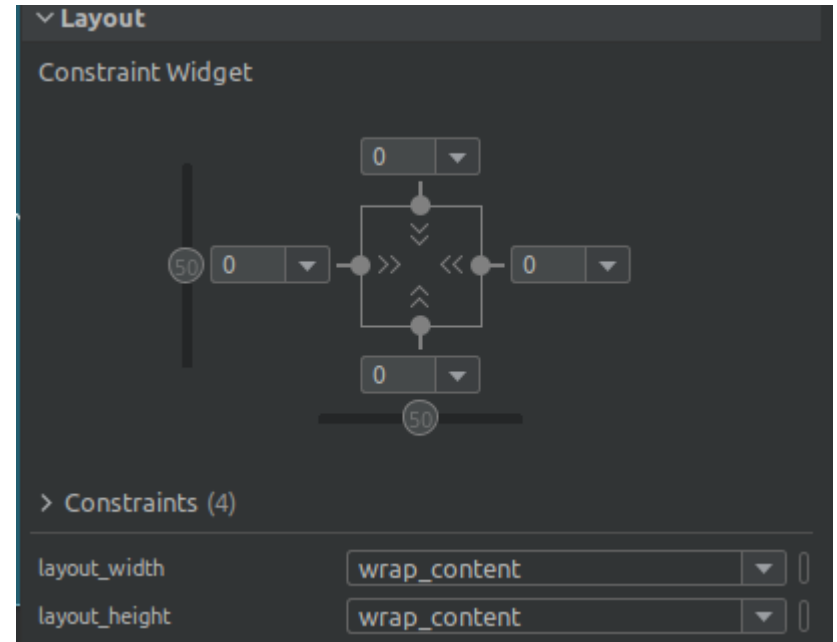
- Uma view é um componente visual do App;
- Tipos de views:
 - Layouts: LinearLayout, ConstraintLayout;
 - Button;
 - EditText;
 - TextView
 - etc

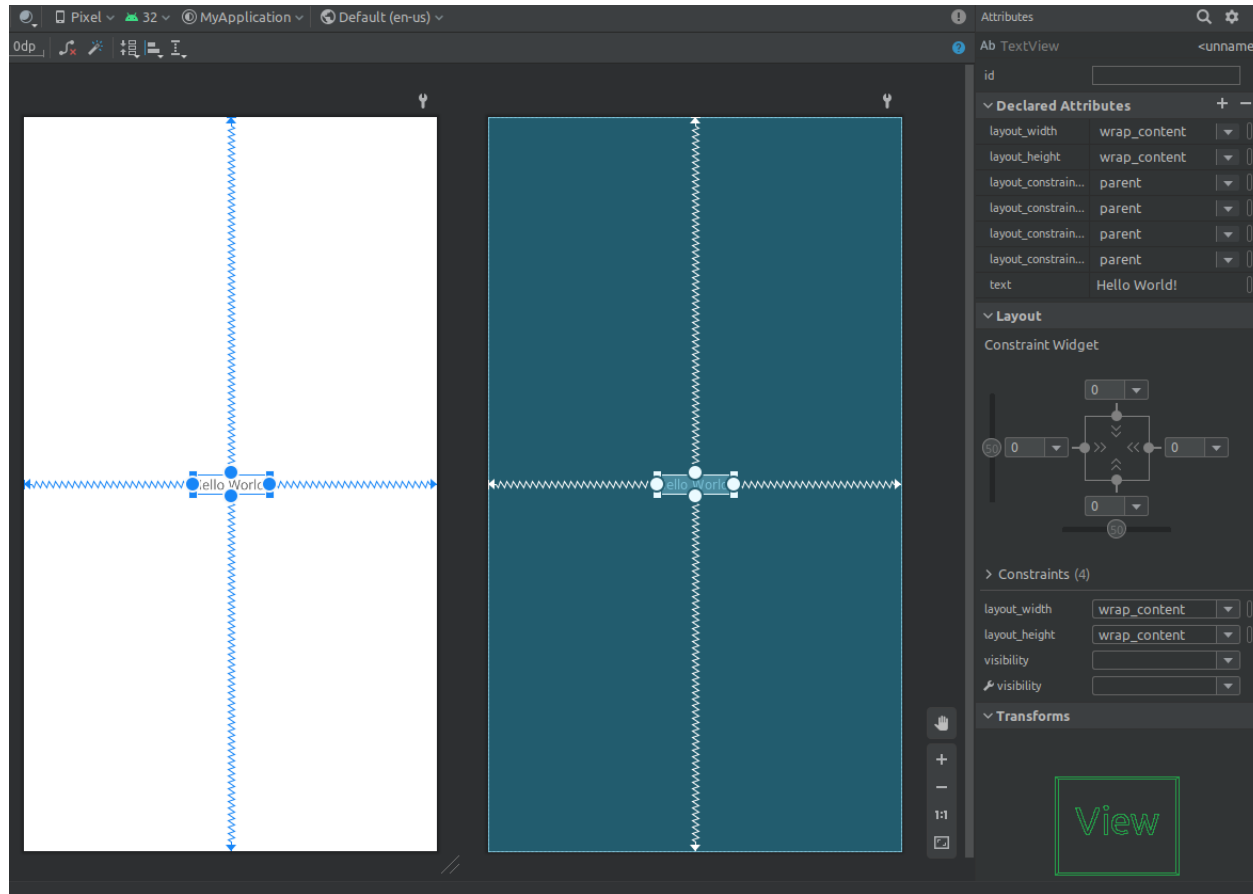
Tipos de Layout



Constraint Layout

- Constraint = restrição;
- Layout baseado em regras;
- Devemos definir as constraints do componente em relação aos demais componentes da tela.





- Atributos do componente selecionado;
- Restrições do componente selecionado;
- Largura e altura do componente selecionado;

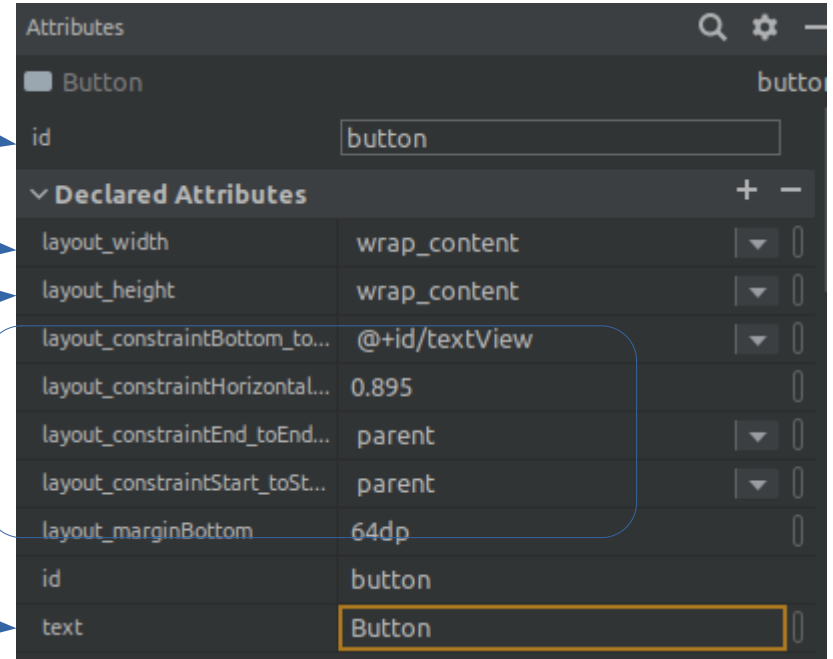
Constraints

layout_constraintBottom_toBottomOf	parent	▼	0
layout_constraintLeft_toLeftOf	parent	▼	0
layout_constraintRight_toRightOf	parent	▼	0
layout_constraintTop_toTopOf	parent	▼	0

- Parent = componente pai;

Atributos de um botão

- Definir seu ID;
- Largura;
- Altura;
- Constraints;
- Texto do botão.

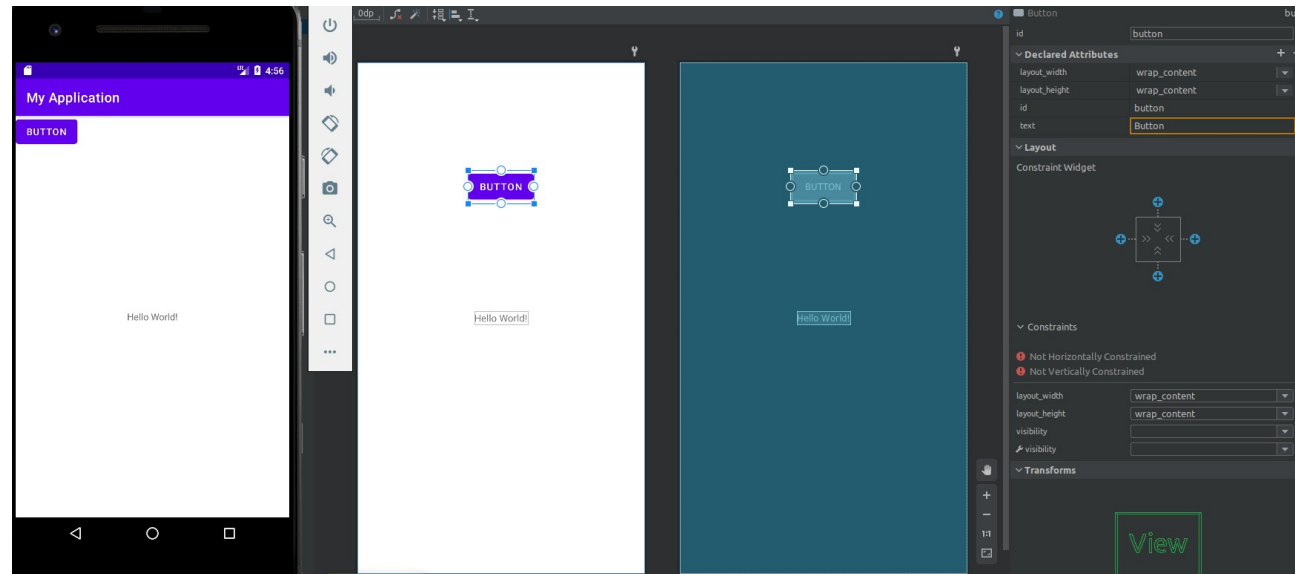


Termos importantes

- `wrap_content`: envolver conteúdo
 - Significa que o tamanho do componente vai corresponder ao seu conteúdo.
- `parent`: componente pai
- `match_parent`: corresponder pai
 - Significa que será considerado o componente pai como referência.

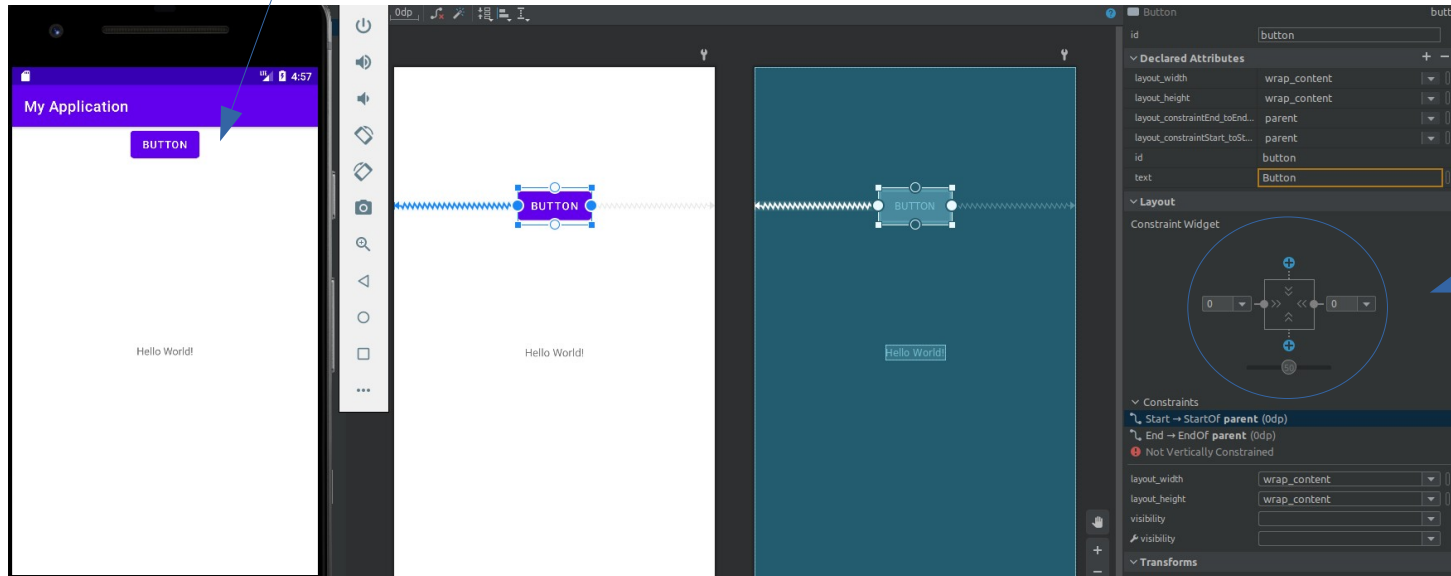
Adicionando um botão

- Mesmo adicionando o botão no centro, ao executar no emulador o botão fica à esquerda;
- Para resolver, precisamos adicionar as restrições (constraints);



Centralizando o botão

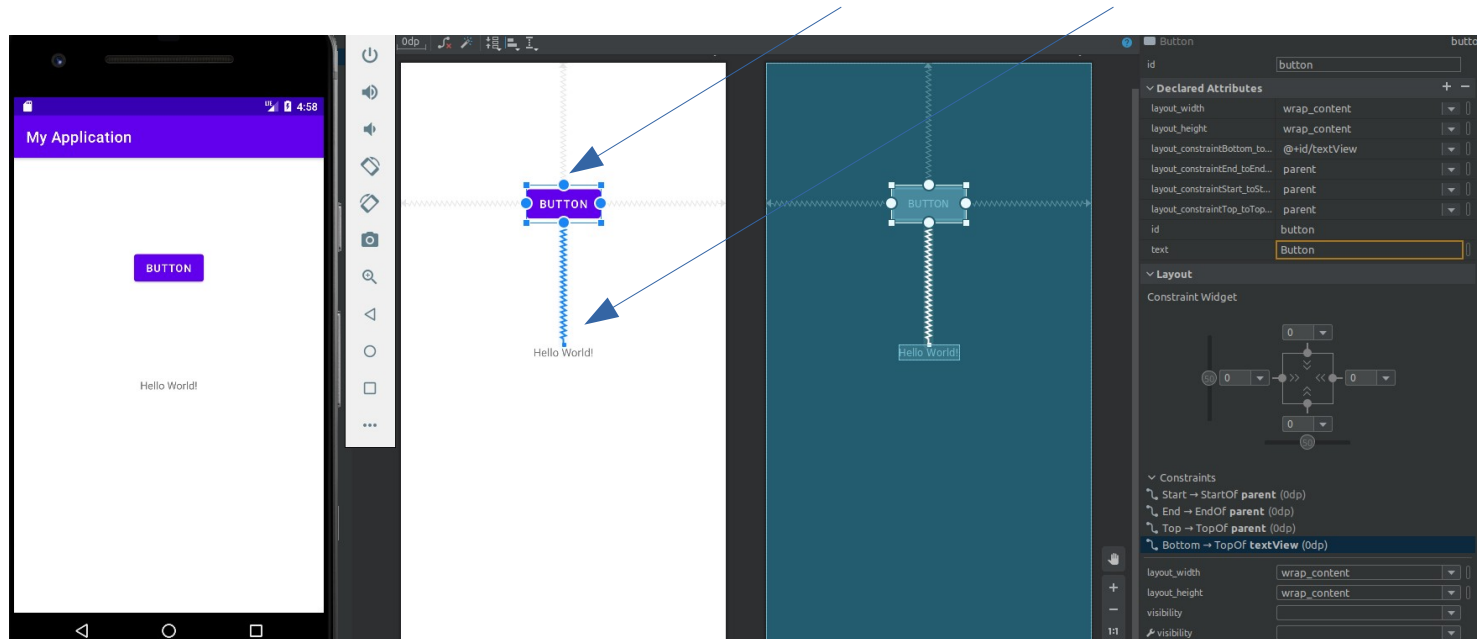
- Vamos definir as constraints para centralizar o botão;
- Clique na bolinha da esquerda ao lado do botão e arraste até a borda da janela;
- Faça o mesmo com a bolinha da direita;
- Veja como ficou.



Agora
temos
duas
constraints
aqui

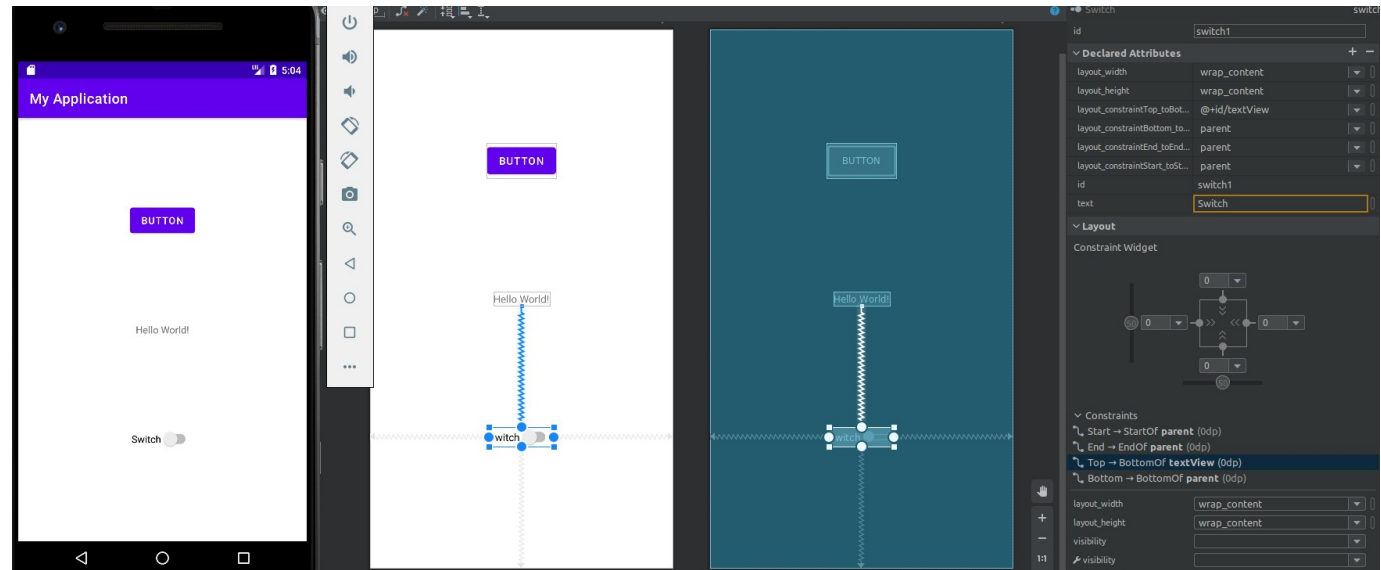
Centralizando verticalmente

- Vamos centralizar o botão verticalmente em relação ao espaço que sobrou entre o rótulo e a borda superior;
- Para isso, adicione constraints nas bolinhas de cima e de baixo do botão.



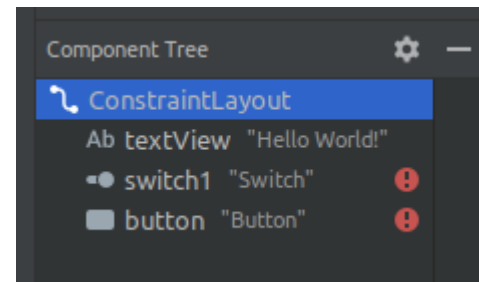
A3e1

- Adicione um “switch” centralizado abaixo do rótulo:



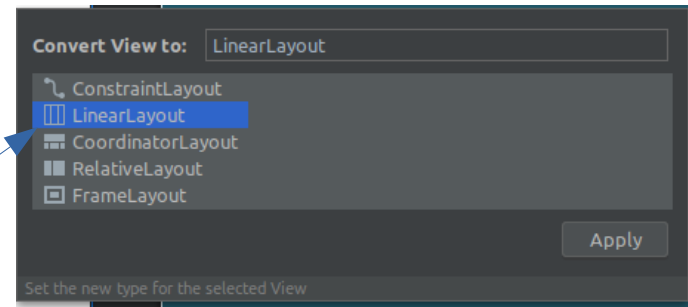
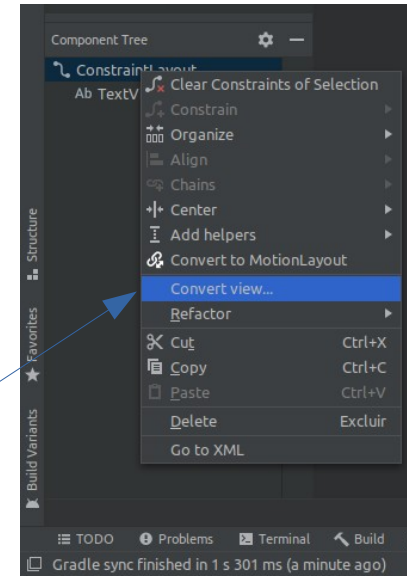
Árvore de componentes

- Hierarquia de componentes;
- Canto inferior esquerdo;
- Permite selecionar mais facilmente os componentes para configurá-los.



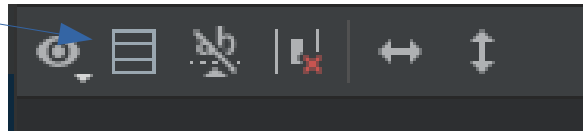
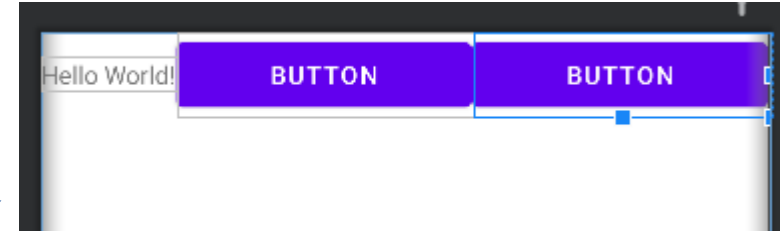
Linear Layout

- Ao criar um novo projeto na IDE, o padrão é utilizar o Constraint Layout;
- Para trocar o layout principal, vamos clicar com o botão direito sobre ConstraintLayout na árvore de componentes e escolher “Convert view...”;
- Em seguida escolha LinearLayout.

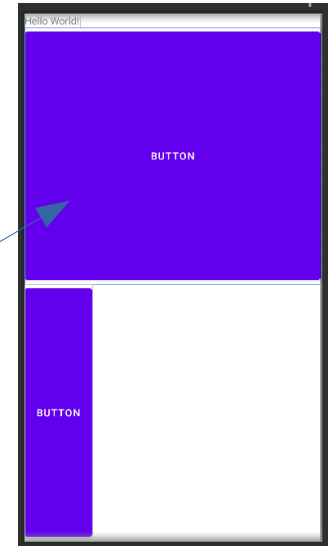


Linear Layout: Horizontal ou Vertical

- Vamos adicionar alguns botões;
- Podemos ver que eles foram organizados horizontalmente, ou seja, estão lado a lado;
- Esse é o formato padrão do LinearLayout;
- Para converter para o **modo vertical**, utilize o segundo botão da barra superior:

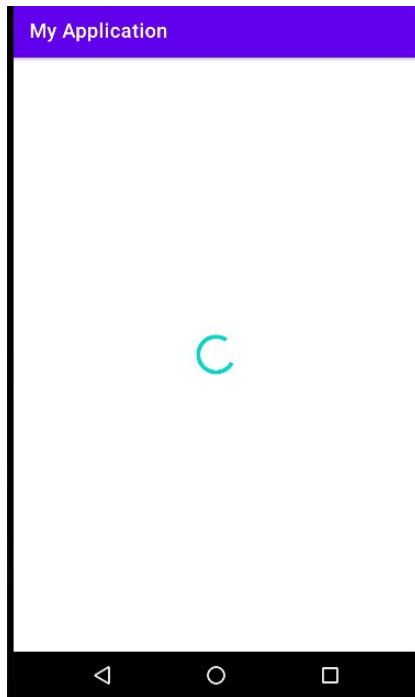


- Para o botão utilizar a largura total da tela, altere seu `layout_width` para "match_parent".

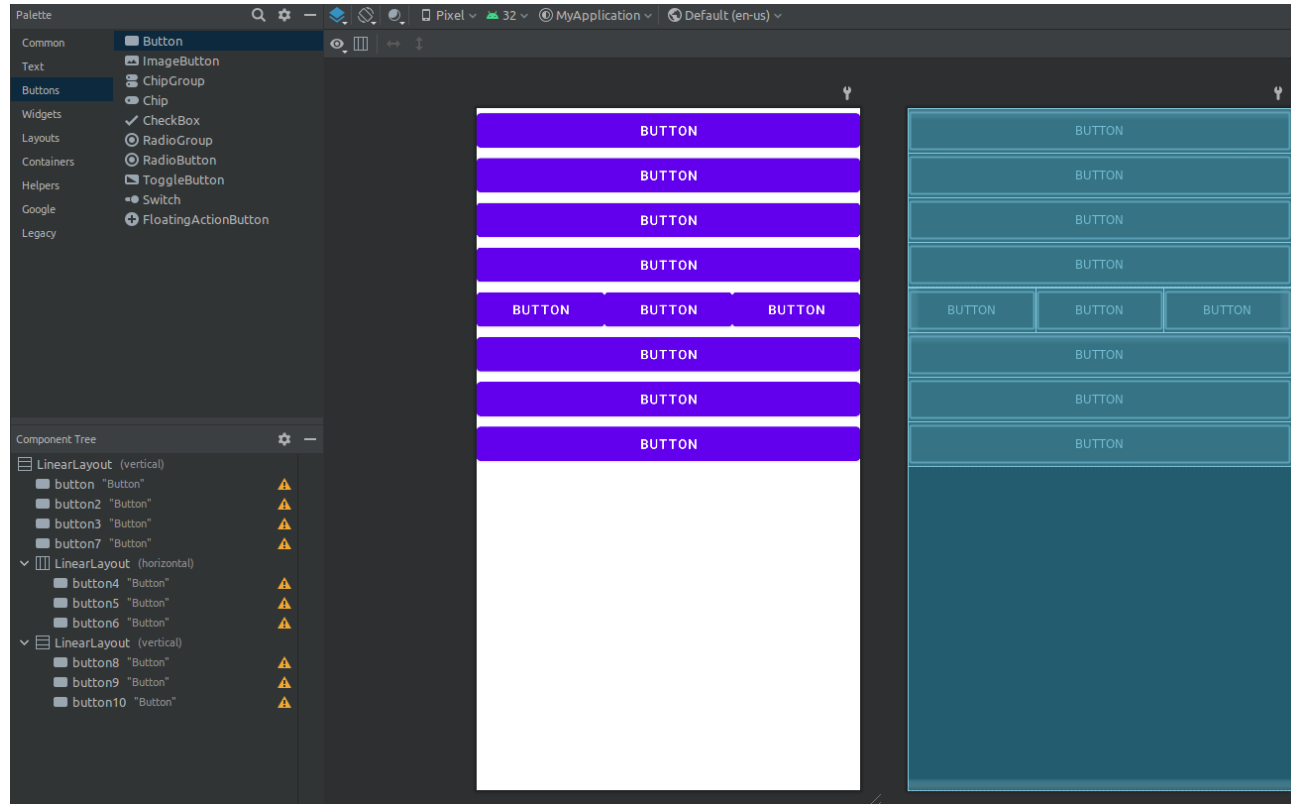


Frame Layout

- Permite inclusão de uma única view;
- Utilizado quando se deseja organizar um componente único na tela.
Ex: tela de inicialização;
- Pode ser utilizado em conjunto com outros layouts a fim de separar a view principal das demais componentes da tela.

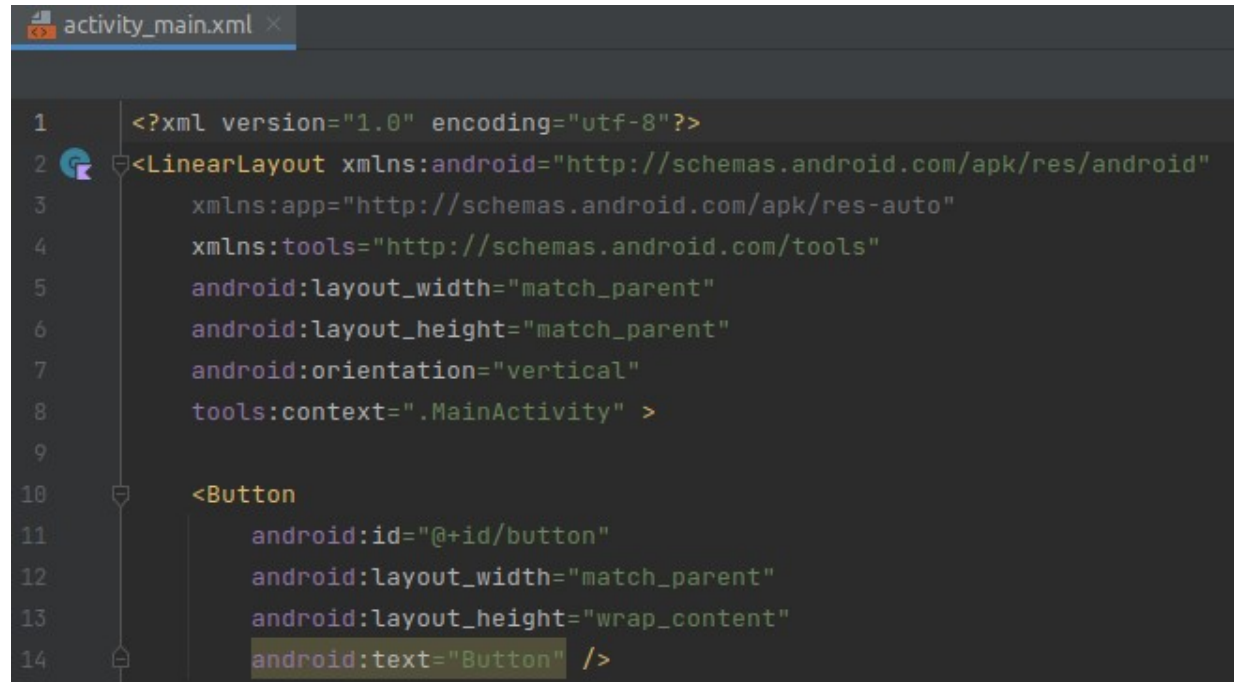


A3ex2



A3ex3

- Desenvolva o Layout anterior unicamente por meio do código XML.



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:orientation="vertical"
8      tools:context=".MainActivity" >
9
10     <Button
11         android:id="@+id/button"
12         android:layout_width="match_parent"
13         android:layout_height="wrap_content"
14         android:text="Button" />
```

O que aprendemos?

- Aprendemos para que servem os layouts;
- Quais os principais layouts para Android;
- Ferramentas da IDE para manipular layouts.

Na próxima aula...

- Iniciaremos nossa jornada pela linguagem Kotlin.

Referências

- <https://developer.android.com/guide/topics/ui/declaring-layout?hl=pt-br>
- https://www.tutorialspoint.com/android/android_user_interface_layouts.htm