

Desenvolvimento Para Dispositivos móveis



Prof. Me. Clênio Silva
E-mail: clenio.silva@uniube.br

CheckBox

- As caixas de seleção permitem que o usuário selecione uma ou mais opções de um conjunto. Normalmente, cada opção é apresentada em uma lista vertical;
- Ele vai te permitir selecionar ou não um valor de uma caixa de seleção;
- Esse elemento permite setar texto para representar a informação da caixa de seleção.

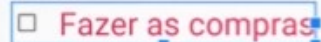
CheckBox

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".ElementosActivity">

    <CheckBox
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/colorAccent"
        android:textSize="32sp"
        android:layout_margin="16dp"
        android:paddingStart="16dp"
        android:text="Fazer as compras" />

</LinearLayout>
```

A propriedade:
textSize permite
aumentar o tamanho
do texto



☐ Fazer as compras

A propriedade: **paddingStart**
permite deslocar o texto do
quadrado de check

ConstraintLayout

- Primeira versão liberada em 2018 juntamente com Android Studio 2.3;
- Recomendação atual pela Google;
- Criado com a intenção de deixar a hierarquia plana e melhorar a performance da aplicação;
- Melhor renderização dos elementos;
- Melhorias significativas que permitem construir layout usando Design do Android além de XML;
- Disponível a partir da API 9.

ConstraintLayout

Por quê usar o ConstraintLayout?

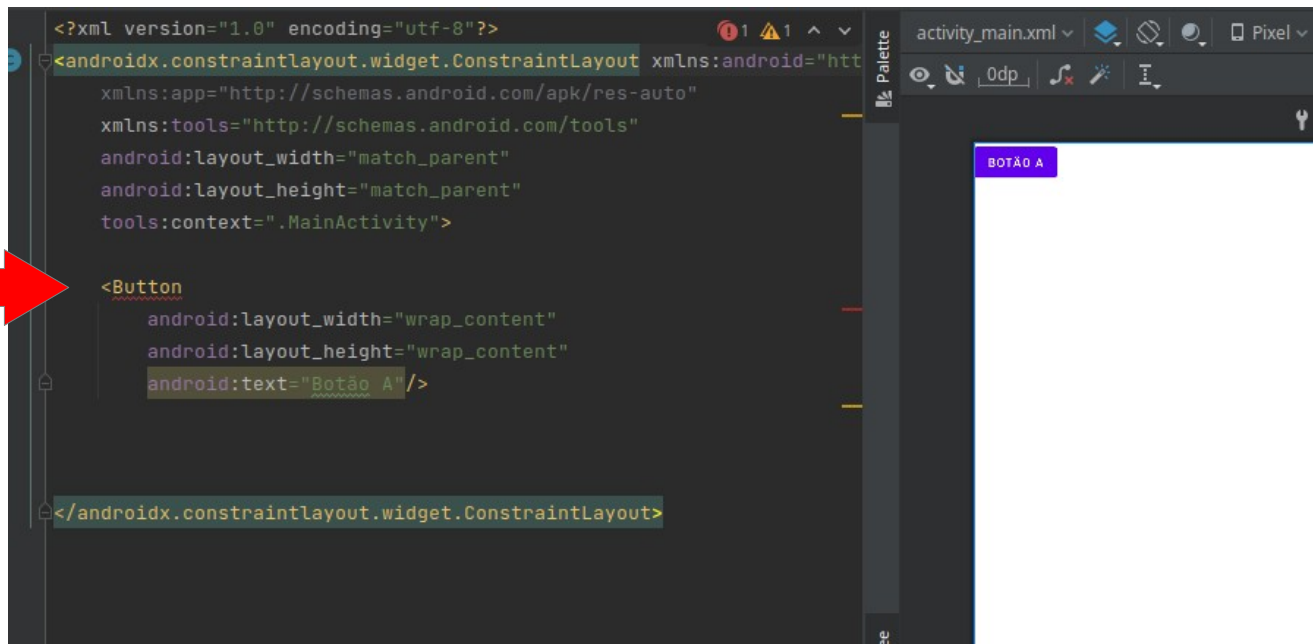
- 1) Porque o ConstraintLayout tem mais customizações que o RelativeLayout e vai tentar deixar sua hierarquia plana, vai tentar deixar o seu layout flat (achatado)
- 2) Tem uma melhor performance na renderização da sua tela.

Diferença entre RelativeLayout e ConstraintLayout...



ConstraintLayout

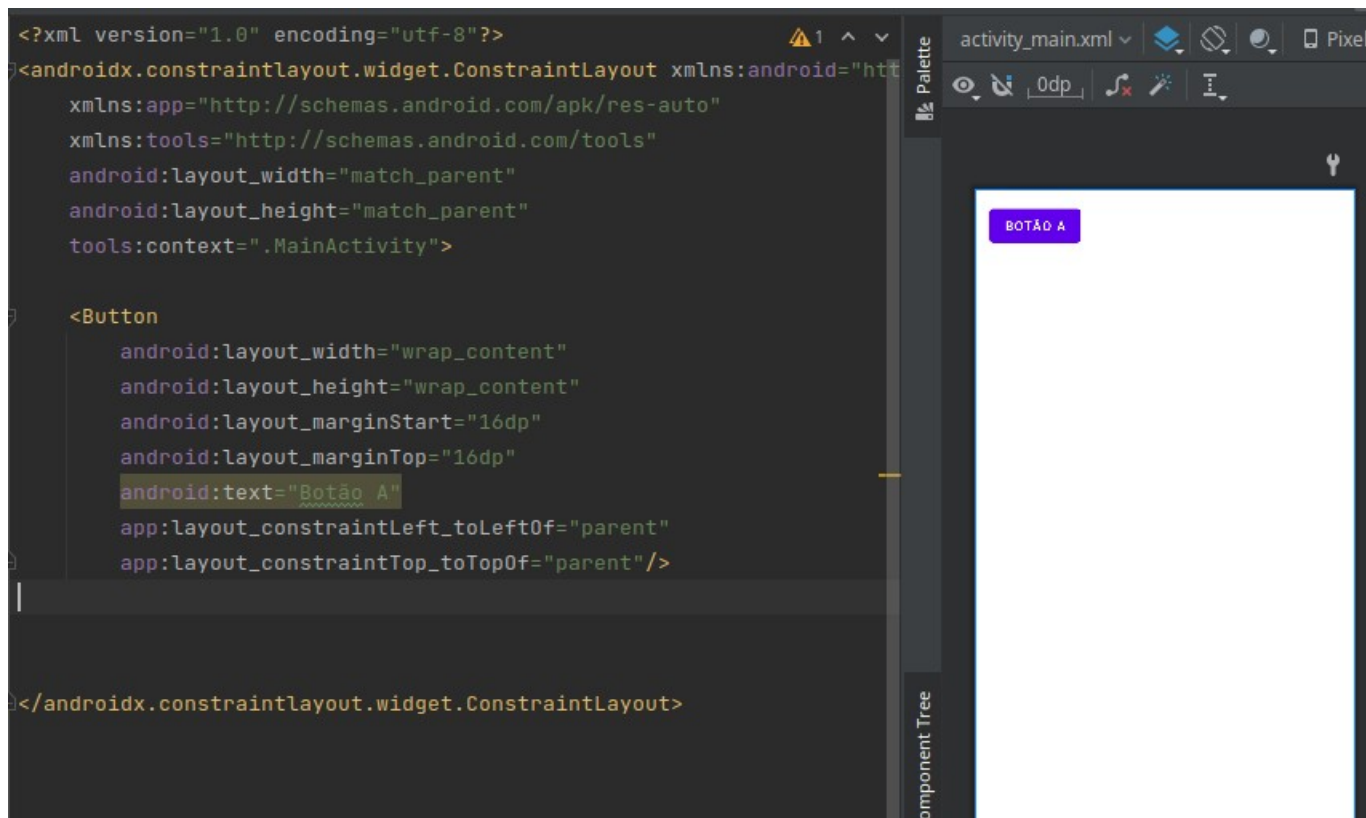
- Mas por quê o elemento Button esta sublinhado como se estivesse com erro?



ConstraintLayout

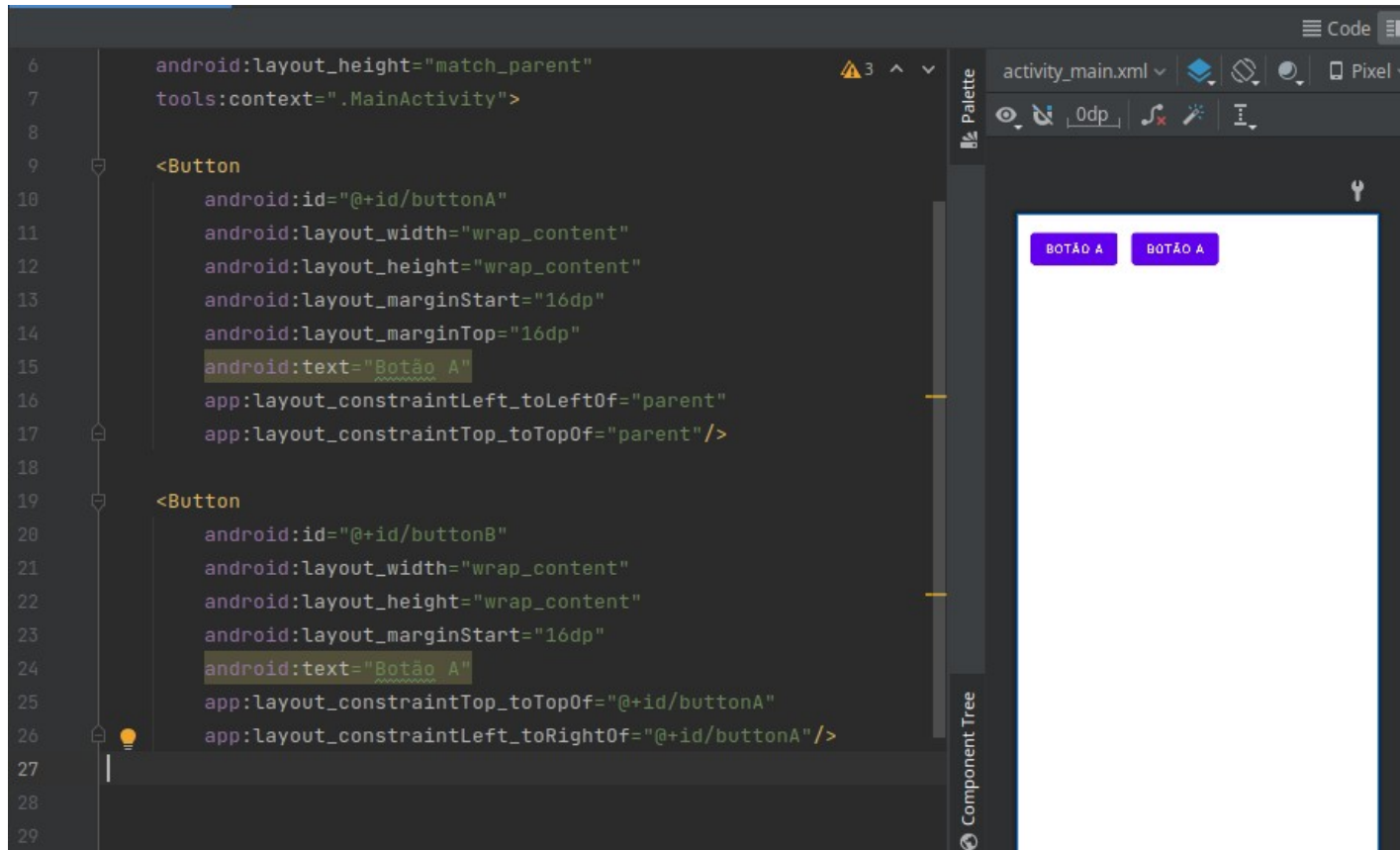
- Mas por quê o elemento Button esta sublinhado como se estivesse com erro?
 - Isso acontece porque ao utilizar uma ViewGroup do tipo `ConstraintLayout` é necessário definirmos constraints que irão definir como os elementos se comportam;

ConstraintLayout



O uso das propriedades `layout_constraintLeft_toLeftOf` e `layout_constraintTop_toTopOf` estão definindo o comportamento do elemento `Button` em relação ao elemento pai. Essas propriedades são **constraints**.

ConstraintLayout



No exemplo anterior adicionamos **constraints** para definir o comportamento do **buttonA** em relação ao pai. Nesse exemplo foi adicionado **constraints** para definir o comportamento do **buttonB** em relação ao **buttonA**.

Para referenciar um elemento é necessário definir um **id**

ConstraintLayout

- Vamos praticar?
 - Adicione mais 3 botões no código anterior e defina constraints para ajustar cada elemento Button em relação ao elemento anterior.

Referências

LECHETA, Ricardo. R. Google Android: aprenda a criar aplicações para dispositivos móveis com Android SDK. 5ª ed. São Paulo, SP: Novatec, 2016.