Constraint Layout

Professor Msc Romulo Beninca

Gerenciadores de *layout*
Constraint Layout

O *constraintLayout* as views são definidas de forma relativa, assim como no *relativeLayout*, a diferença é que somos capazes de indicar o posicionamento que queremos manter as *Views* por meio dos seus eixos.

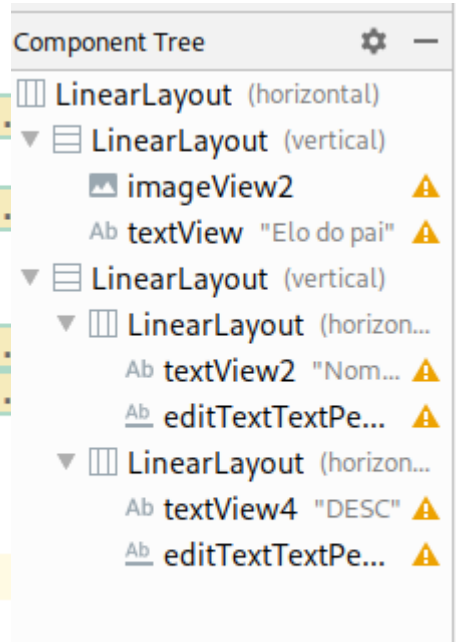
Constraint Layout como uma solução para definir layouts complexos sem a necessidade de uma hierarquia aninhada.

Evita necessidade de edição de XML.

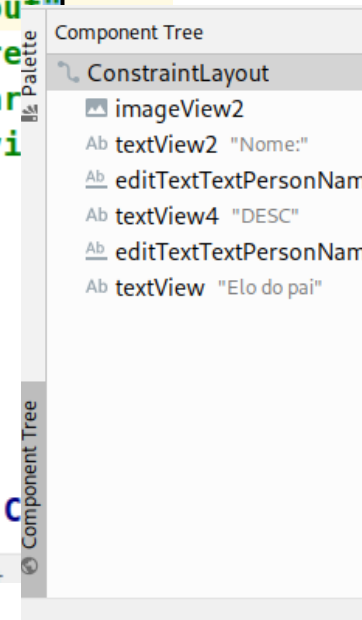
Exemplo de layout

A direita a declaração do layout utilização Constraint layout e a Esquerda utilizando linear layout


```
<LinearLayout
    <ImageView...>
    <TextView...>
</LinearLayout>
<LinearLayout
    <LinearLayout
        <TextView...
        <EditText...
    </LinearLayout>
    <LinearLayout
        <TextView...
        <EditText...
    </LinearLayout>
</LinearLayout>
</LinearLayout>
```

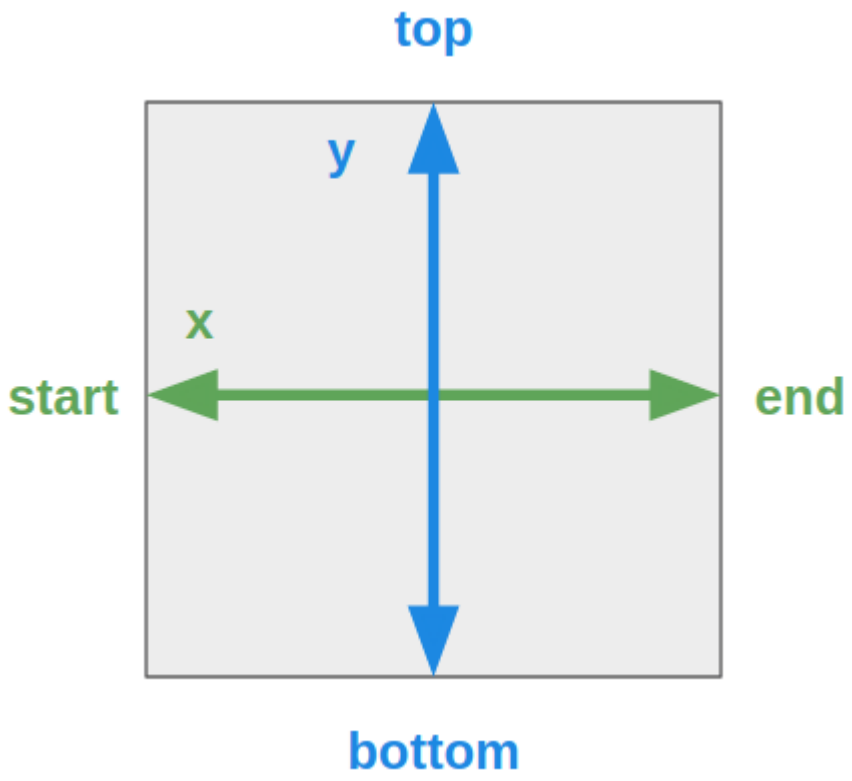



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.Constraint
    xmlns:app="http://schemas.android.com/ap
    xmlns:tools="http://schemas.android.com/
    android:id="@+id/ConstraintLayou
    android:layout_width="match_pare
    android:layout_height="match_par
    tools:context=".ConstraintActivi

    <ImageView...>
    <TextView...>
    <EditText...>
    <TextView...>
    <EditText...>
    <TextView...>
</androidx.constraintlayout.widget.C
```



Algumas propriedades do *constraintLayout*

Para posicionar as *views* no *constraint* layout deve-se sempre definir um regra para para cada eixo,Os eixos são definidos como X e Y, sendo X o eixo de ínicio (esquerdo) e fim (direito) e o Y o de topo (cima) e inferior (baixo).



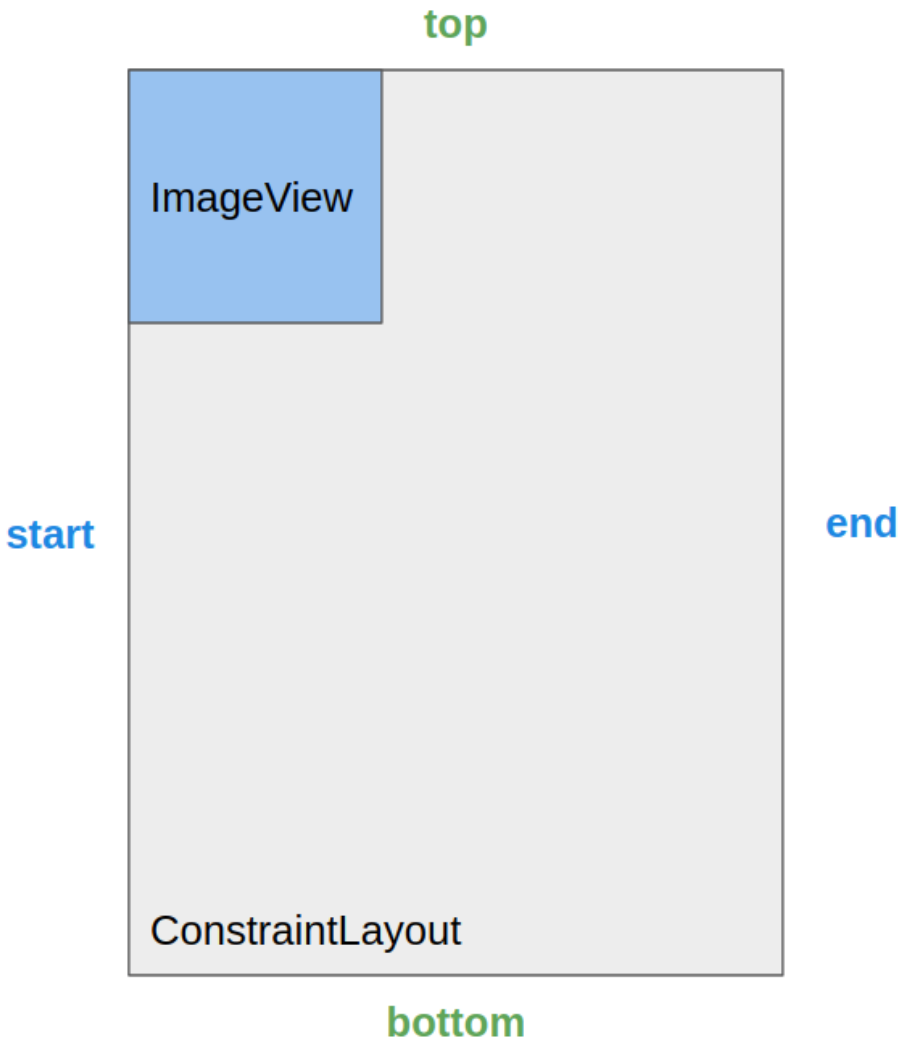
Representação teórica para cada View dentro do Constraint Layout

ConstraintLayout layout_constraint{Start_toStartOf |Top_toTopOf

O constraintLayout tem algumas propriedades :

- **layout_constraintStart_toStartOf**: alinha o início da *View* no início de outra *View*.
- **layout_constraintTop_to_topOf:** alinha o topo da *View* no topo de outra *View*.

```
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintStart_toTopOf="parent"
```



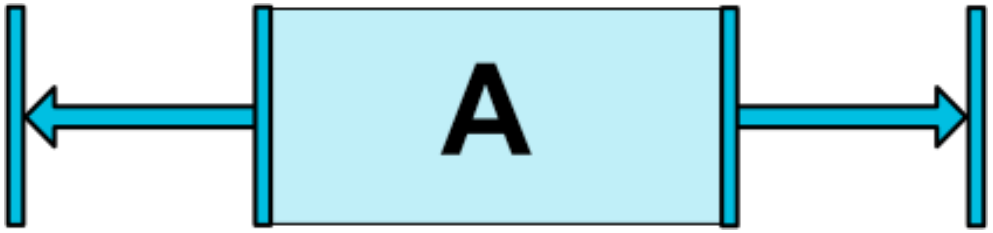
Algumas propriedades do constraintLayout

O constraintLayout tem algumas propriedades :

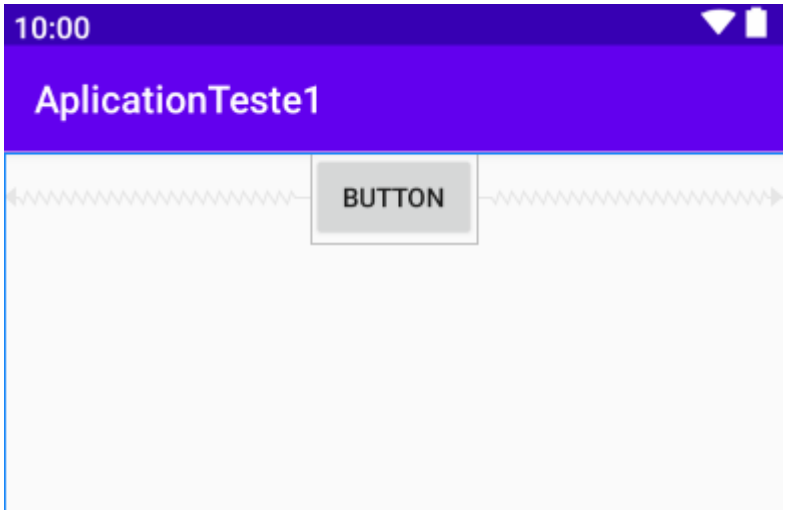
- **layout_constraintStart_toStartOf**: alinha o início da View no início de outra View.
- **layout_constraintTop_to_topOf:** alinha o topo da View no topo de outra View.
- layout_constraintLeft_toLeftOf
- ayout_constraintLeft_toRightOf
- layout_constraintRight_toLeftOf
- layout_constraintRight_toRightOf
- ayout_constraintTop_toTopOf
- layout_constraintTop_toBottomOf
- layout_constraintBottom_toTopOf
- layout_constraintBottom_toBottomOf
- layout_constraintBaseline_toBaselineOf
- layout_constraintStart_toEndOf
- layout_constraintStart_toStartOf
- layout_constraintEnd_toStartOf
- layout_constraintEnd_toEndOf
-

ConstraintLayout
Centralizando

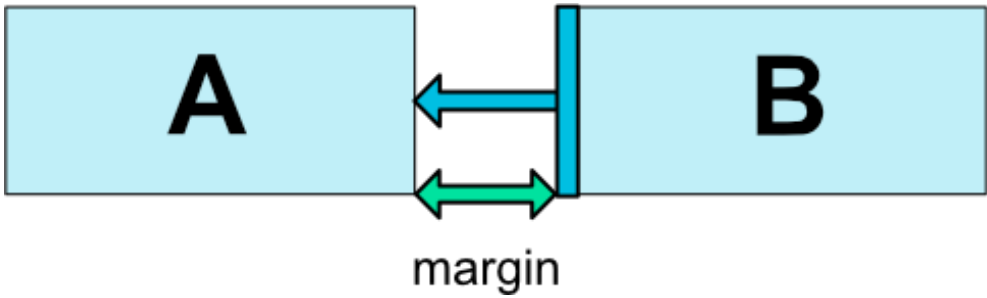
Para centralizar uma *view* utilizando o *constraintLayout* basta definir a posição de inicio e fim como as extremidades as quais deseja-se posicionar o elemento.



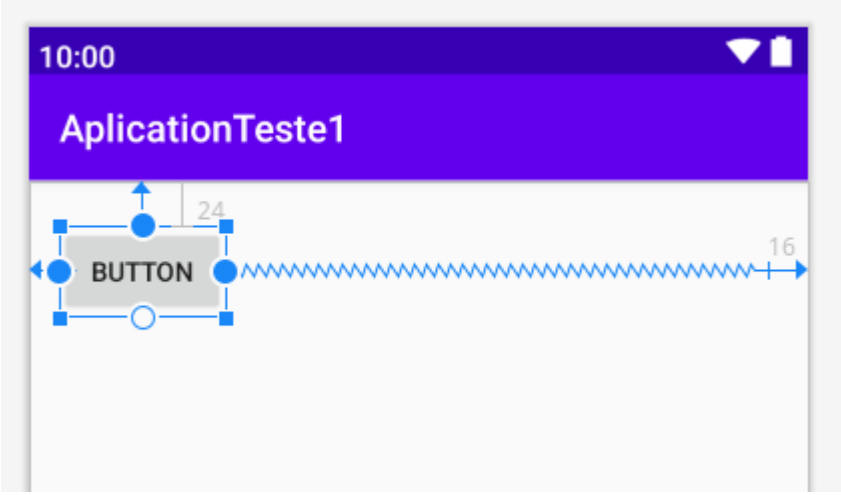

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    android:text="Button"

     />
```



ConstraintLayout
Margin

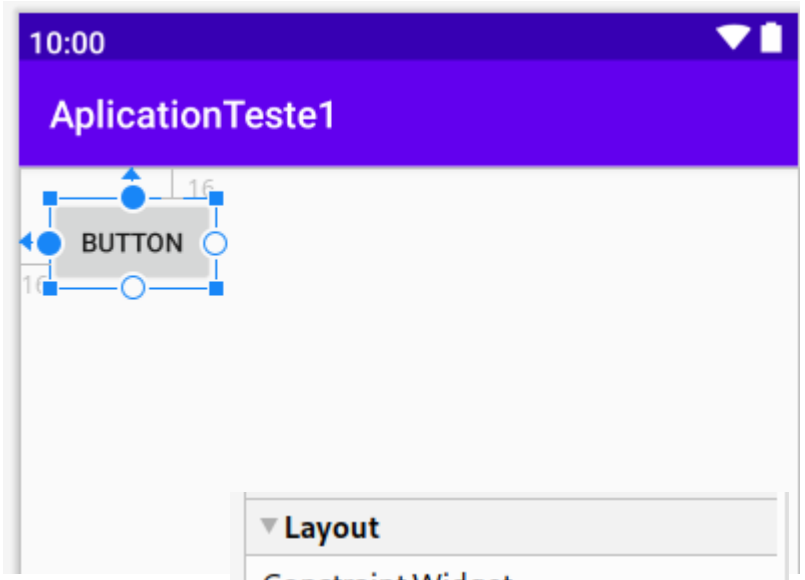


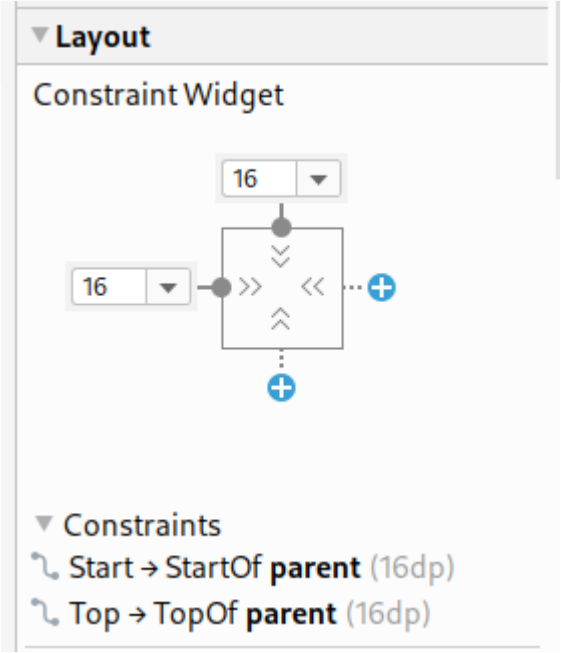
- android:layout_marginStart
- android:layout_marginEnd
- android:layout_marginLeft
- android:layout_marginTop
- android:layout_marginRight
- android:layout_marginBottom



Exemplo de uso de margin

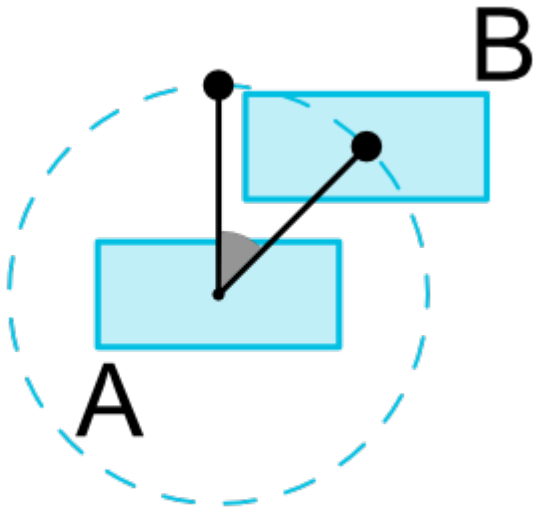
As propriedades de margin restrigem a próximidade da view de outros componentes.






```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="16dp"
    android:text="Button"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

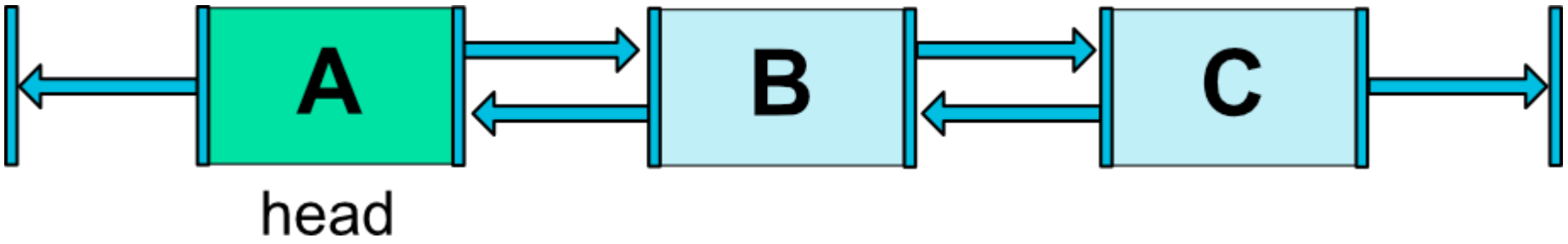
ConstraintLayout
Posicionamento radial



- **layout_constraintCircle** :faz refereccia ao ide do componente
- **layout_constraintCircleRadius :** distncia ao centro do componentes
- **layout_constraintCircleAngle :**defin e o angulo0 to 360)
-

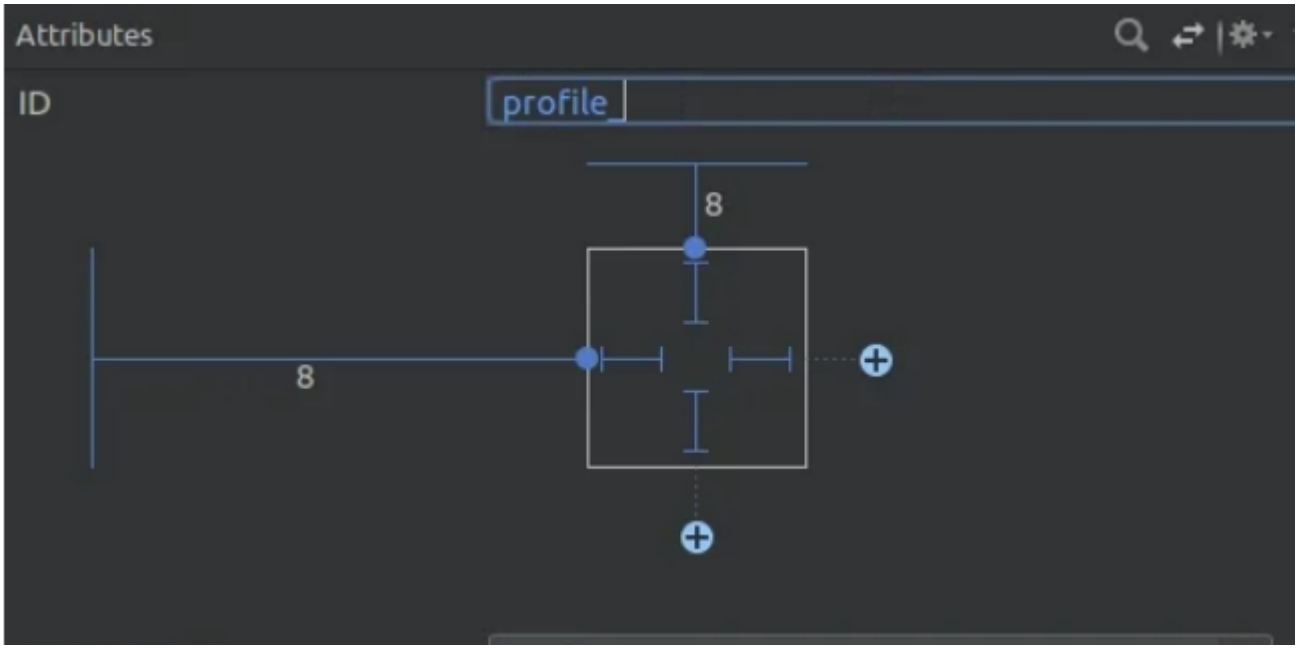
Correntes

O uso de correntes possibilita alinhar componentes de forma equidistante em uma mesma linha.

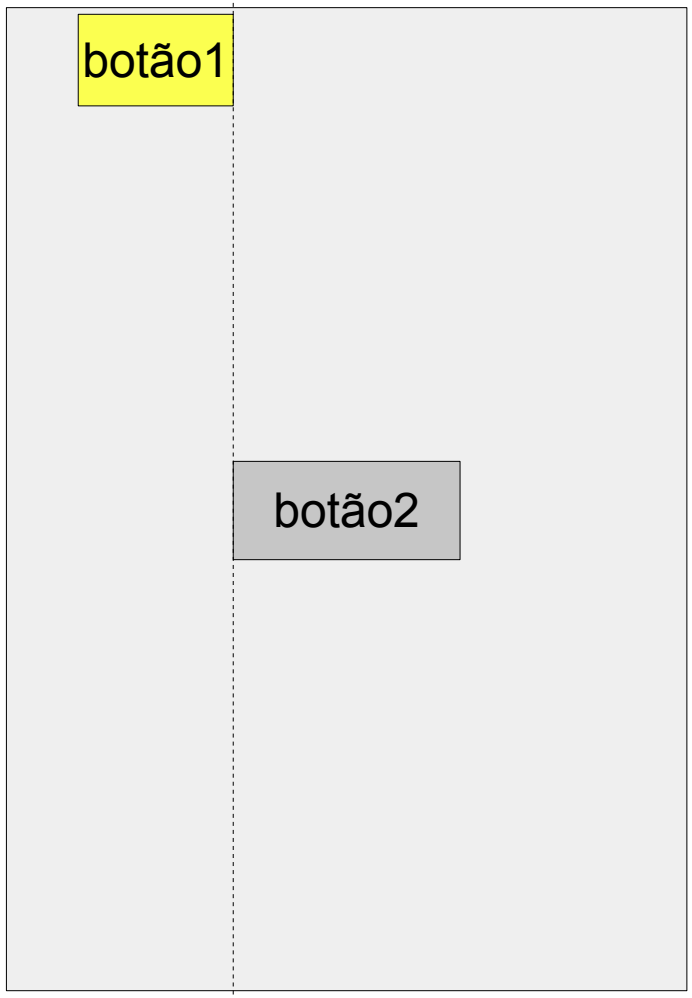


Margin

As propriedades de margin são individuais para cada view, e podem ser definidas pelo código xml, ou pela ferramenta de desing.



GuideLines são linhas verticais ou horizontais que podemos utilizar para posicionar nossos views, elas não tem visualização apenas nos auxiliarm




```xml
<?xml version="1.0" encoding="utf-8"?>
<ConstraintLayout .... >

  <Button
    android:id="@+id/botão1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="100dp"
    android:text="Button"
    app:layout_constraintEnd_toStartOf="@+id/guideline"
    app:layout_constraintTop_toTopOf="parent" />

<Button
    android:id="@+id/botão2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="340dp"
    android:text="Button"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="@+id/guideline" />

<androidx.constraintlayout.widget.Guideline
    android:id="@+id/guideline"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    app:layout_constraintGuide_begin="160dp" />
</ConstraintLayout>
```

Fonte:Autor (2018)

Referências

- Lecheta, R. Google Android: Aprenda a criar aplicações para dispositivos móveis. 3a Ed. Novatec, 2013.
- Androd Developer. Android Disponível em:<*https://developer.android.com*>
- Androd Developer, 2018. Relative Parameters Disponível em:<*https://developer.android.com/reference/android/widget/RelativeLayout.LayoutParams*>