

Gerenciadores de layout do Android

Alisson G. Chiquitto

¹Instituto Federal do Mato Grosso do Sul

Naviraí, 2018

Outline - Seção

- 1 Introdução
- 2 Classe View
- 3 Classe ViewGroup
- 4 Referências

Gerenciadores de layout

No Android, existem diversos tipos de gerenciadores de layout.

Alguns podem organizar os componentes na horizontal e vertical, outros podem organizar os componentes em uma tabela com linhas e colunas.

Outline - Seção

1 Introdução

2 Classe View

- Widgets
- Gerenciadores de layout

3 Classe ViewGroup

4 Referências

A classe `android.view.View` é a classe mãe de todos os componentes visuais (*views*) do Android.

As subclasses são utilizadas para criar a interface gráfica do aplicativo.

Cada subclasse de `View` precisa implementar o método `onDraw(canvas:Canvas)`.

Tipos de Views

Existem dois tipos de views:

- *widgets*;
- gerenciadores de layout;

2 Classe View

- Widgets
- Gerenciadores de layout

Um *widget* é um componente que herda diretamente da classe `View`.

São chamados muitas vezes de componentes.

Exemplos: `Button`, `ImageView` e `TextView`.

2 Classe View

- Widgets
- Gerenciadores de layout

Gerenciadores de layout

São subclasses de `android.view.ViewGroup`.

Popularmente chamados de *layouts*.

Utilizado para controlar a disposição dos elementos na tela.

1 Introdução

2 Classe View

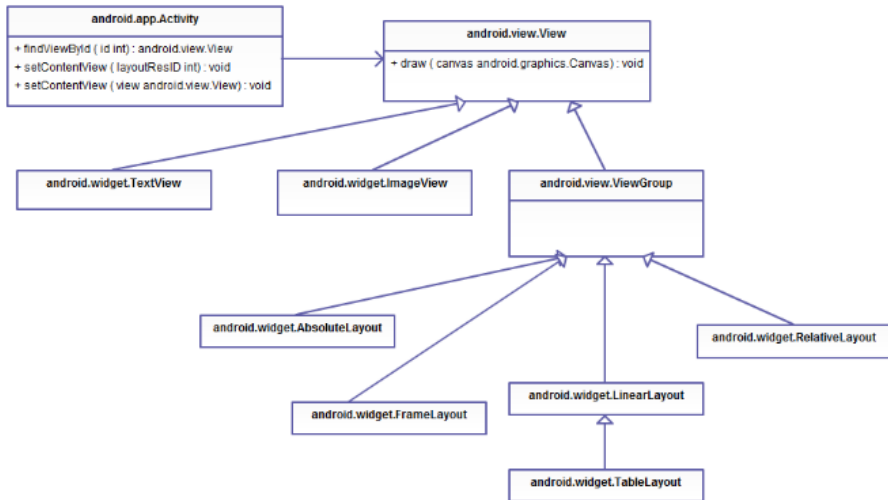
3 Classe ViewGroup

- Layouts
- AbsoluteLayout
- FrameLayout
- LinearLayout
- TableLayout
- RelativeLayout
- GridLayout
- ConstraintLayout

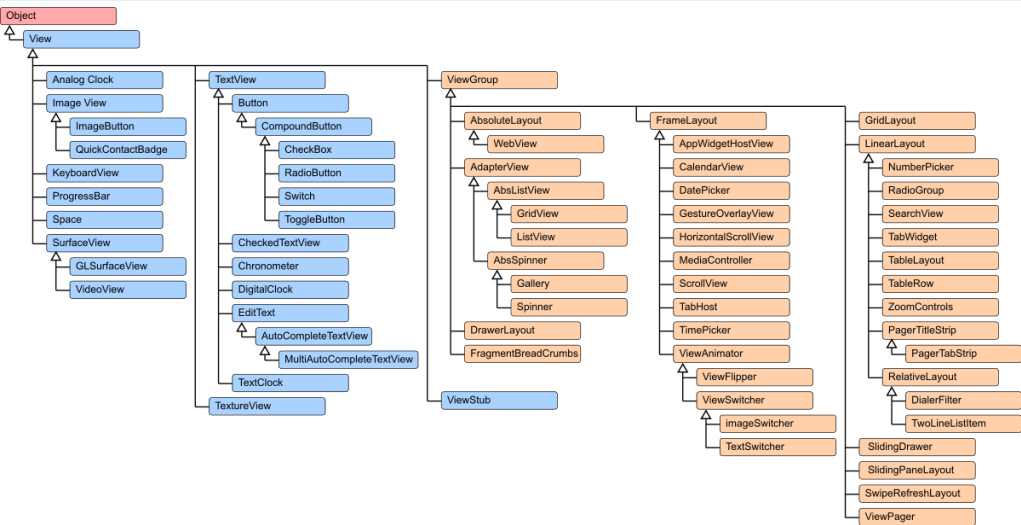
A classe `android.view.ViewGroup` é a classe mãe de todos os gerenciadores de layout do Android.

Os gerenciadores de layout são utilizados para organizar a disposição dos componentes (*views*) na tela.

Hierarquia classes View



Hierarquia classes View



3 Classe ViewGroup

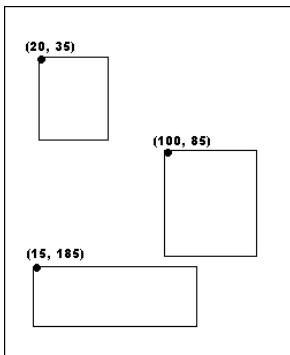
- Layouts
 - AbsoluteLayout
 - FrameLayout
 - LinearLayout
 - TableLayout
 - RelativeLayout
 - GridLayout
 - ConstraintLayout

- `AbsoluteLayout`
- `FrameLayout`
- `LinearLayout`
- `TableLayout`
- `RelativeLayout`
- `GridLayout`
- `ConstraintLayout`

3 Classe ViewGroup

- Layouts
- **AbsoluteLayout**
- FrameLayout
- LinearLayout
- TableLayout
- RelativeLayout
- GridLayout
- ConstraintLayout

AbsoluteLayout



- permite especificar a posição exata (coordenadas x e y) dos seus filhos;
- são menos flexíveis e mais difíceis de manter;
- está *deprecated* e por isso não deve ser utilizada.

- [AbsoluteLayout.LayoutParams | Android Developers](#)
- [Android Absolute Layout](#)

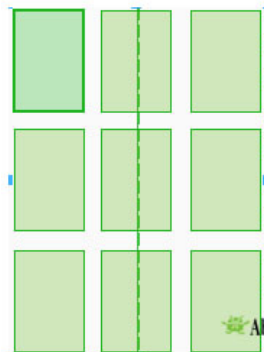
AbsoluteLayout

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"
3   android:layout_width="fill_parent"
4   android:layout_height="fill_parent">
5
6   <Button
7     android:layout_width="wrap_content"
8     android:layout_height="wrap_content"
9     android:text="Clique-me"
10    android:layout_x="100dp"
11    android:layout_y="100dp" />
12 </AbsoluteLayout>
```

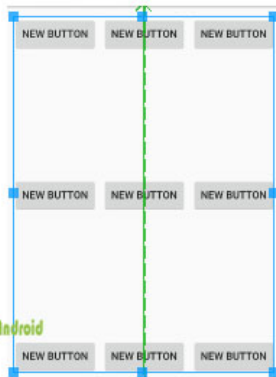
3 Classe ViewGroup

- Layouts
- AbsoluteLayout
- **FrameLayout**
- LinearLayout
- TableLayout
- RelativeLayout
- GridLayout
- ConstraintLayout

FrameLayout



Frame Layout



Button Placed in
Frame Layout

FrameLayout

FrameLayout cria uma área onde os filhos são posicionados em forma de pilha, e ficam “ancorados” em um lado do layout.

Geralmente é utilizado para apenas um componente filho, porque pode se tornar difícil para organizar os filhos de uma forma sem que um sobreponha os outros em diferentes tamanhos de telas.

- [FrameLayout.LayoutParams | Android Developers](#)
- [Android Frame Layout](#)

FrameLayout

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="fill_parent"
4     android:layout_height="fill_parent">
5
6     <Button
7         android:layout_width="wrap_content"
8         android:layout_height="wrap_content"
9         android:text="Clique-me"
10        android:layout_gravity="left|center_vertical"/>
11
12    <Button
13        android:layout_width="wrap_content"
14        android:layout_height="wrap_content"
15        android:text="Clique-me"
16        android:layout_gravity="right|bottom"/>
17 </FrameLayout>
```

3 Classe ViewGroup

- Layouts
- AbsoluteLayout
- FrameLayout
- **LinearLayout**
- TableLayout
- RelativeLayout
- GridLayout
- ConstraintLayout

LinearLayout



Alinha todos os filhos em uma única direção (vertical ou horizontal).

- [LinearLayout.LayoutParams | Android Developers](#)
- [Android Linear Layout](#)

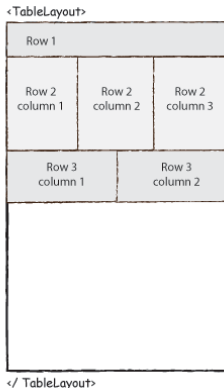
LinearLayout

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="fill_parent"
4     android:layout_height="fill_parent"
5     android:orientation="vertical"
6     android:gravity="left|top">
7
8     <Button
9         android:layout_width="wrap_content"
10        android:layout_height="wrap_content"
11        android:text="Clique-me"/>
12
13    <Button
14        android:layout_width="wrap_content"
15        android:layout_height="wrap_content"
16        android:text="Clique-me"/>
17 </LinearLayout>
```

3 Classe ViewGroup

- Layouts
- AbsoluteLayout
- FrameLayout
- LinearLayout
- **TableLayout**
- RelativeLayout
- GridLayout
- ConstraintLayout

TableLayout



- subclasse de `LinearLayout`;
- organiza as *views* em formato tabela, com linhas e colunas.

- `TableLayout.LayoutParams` | Android Developers
- Android Table Layout

TableLayout

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
3   android:layout_width="fill_parent" android:layout_height="fill_parent">
4
5   <TableRow>
6     <Button android:text="Clique-me" />
7     <Button android:text="Clique-me" android:layout_span="2" />
8   </TableRow>
9
10  <TableRow>
11    <Button android:text="Clique-me" />
12    <Button android:text="Clique-me" />
13    <Button android:text="Clique-me" />
14  </TableRow>
15
16  <TableRow>
17    <Button android:layout_column="2" android:text="Clique-me" />
18  </TableRow>
19
20 </TableLayout>
```

3 Classe ViewGroup

- Layouts
- AbsoluteLayout
- FrameLayout
- LinearLayout
- TableLayout
- **RelativeLayout**
- GridLayout
- ConstraintLayout

RelativeLayout



RelativeLayout exhibe as *views* em posições relativas. A posição de cada *view* pode ser especificada em relação aos elementos de irmãos (acima ou abaixo por exemplo) ou em posições relativas ao container do layout (ex.: alinhado à esquerda ou fixado no rodapé).

- [RelativeLayout.LayoutParams | Android Developers](#)
- [Android Relative Layout](#)

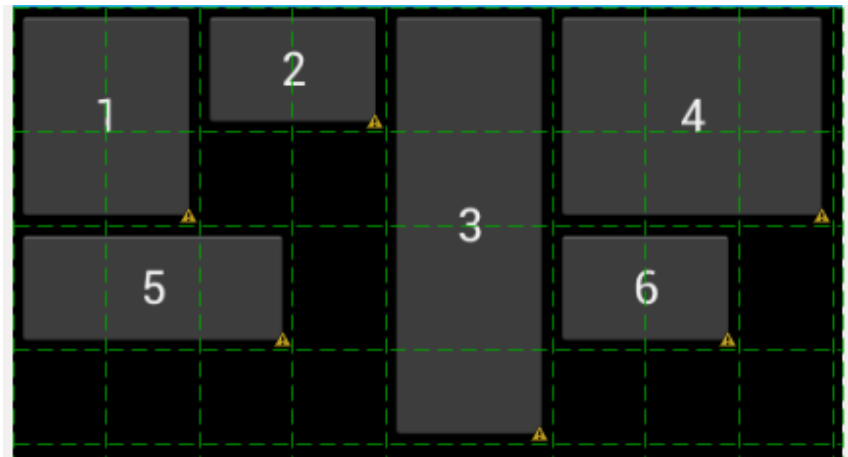
RelativeLayout

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="fill_parent"
4     android:layout_height="fill_parent">
5
6     <TextView
7         android:layout_width="wrap_content"
8         android:layout_height="wrap_content"
9         android:text="Texto no topo"
10        android:id="@+id/textView"
11        android:layout_alignParentTop="true"
12        android:layout_centerHorizontal="true"
13        android:layout_marginTop="40dp" />
14    <Button
15        android:layout_width="wrap_content"
16        android:layout_height="wrap_content"
17        android:layout_below="@id/textView"
18        android:text="Clicar"
19        android:layout_centerHorizontal="true"
20        android:layout_marginTop="20dp"/>
21
22 </RelativeLayout>
```

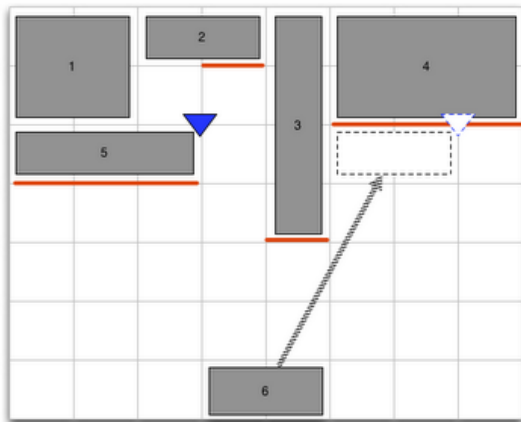

3 Classe ViewGroup

- Layouts
- AbsoluteLayout
- FrameLayout
- LinearLayout
- TableLayout
- RelativeLayout
- **GridLayout**
- ConstraintLayout

GridLayout



GridLayout



GridLayout

Usa uma grade de linhas (invisíveis) para criar áreas virtuais: linhas, colunas e células.

É possível mesclar células, fazendo que uma *view* ocupe um intervalo de células vizinhas.

Alocação automática: os filhos são alocados automaticamente nas células vazias.

- [GridLayout.LayoutParams | Android Developers](#)
- [New Layout Widgets: Space and GridLayout](#)
- [Android Grid Layout](#)

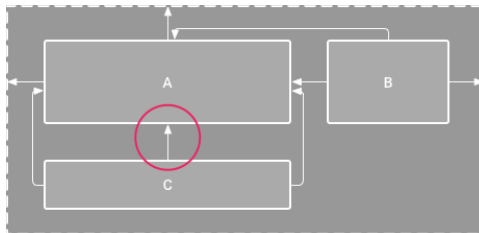
GridLayout

```
1 <GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
2   android:layout_width="match_parent"
3   android:layout_height="match_parent"
4   android:columnCount="3"
5   android:orientation="horizontal">
6
7   <Button android:text="Botao1" />
8
9   <Button android:text="Botao2" />
10
11  <Button android:text="Botao3"
12    android:layout_columnSpan="2" />
13
14  <Button android:text="Botao4" />
15
16 </GridLayout>
```

3 Classe ViewGroup

- Layouts
- AbsoluteLayout
- FrameLayout
- LinearLayout
- TableLayout
- RelativeLayout
- GridLayout
- **ConstraintLayout**

ConstraintLayout







Define como uma view será posicionada em relação a outras *views*.
Pode-se definir uma *constraint* para um ou mais *views*, por exemplo, ligando a *view* a um:

- ponto de ancoragem de outra *view*;
- lateral do layout;
- linha guia.





Outline - Seção

- 1 Introdução
- 2 Classe View
- 3 Classe ViewGroup
- 4 Referências**

Referências |

-  Layouts | Android Developers
<https://developer.android.com/guide/topics/ui/declaring-layout.html>
-  Build a Responsive UI with ConstraintLayout | Android Developers
<https://developer.android.com/training/constraint-layout/index.html#constraints-overview>
-  Como usar o Constraint Layout no Android - Imasters
<https://imasters.com.br/mobile/android/como-usar-o-constraint-layout-no-android/>
-  Android - UI Layouts
https://www.tutorialspoint.com/android/android_user_interface_layouts.htm

Referências II

-  Como dominar os Android Layouts em 07 passos
<https://www.androidpro.com.br/android-layouts-viewgroups-intro/>
-  Android UI Layouts Tutorial
<https://o7planning.org/en/10423/android-ui-layouts-tutorial>
-  Understanding Android Views, View Groups and Layouts
http://www.techotopia.com/index.php/Understanding_Android_Views,_View_Groups_and_Layouts
-  Graphics Programming - GUIs with Java (Android)
http://www.mathematik.uni-marburg.de/~thormae/lectures/graphics1/graphics_2_3_eng_web.html#1