Gerenciadores de layout
ViewGroups

Dispositivos Moveis

Professor Msc Romulo Beninca

Sumário

O que são Views?

- O que são ViewsGrups?
- AbsoluteLayout
- LinearLayout
- RelativeLayout
- ConstraintLayout
  -

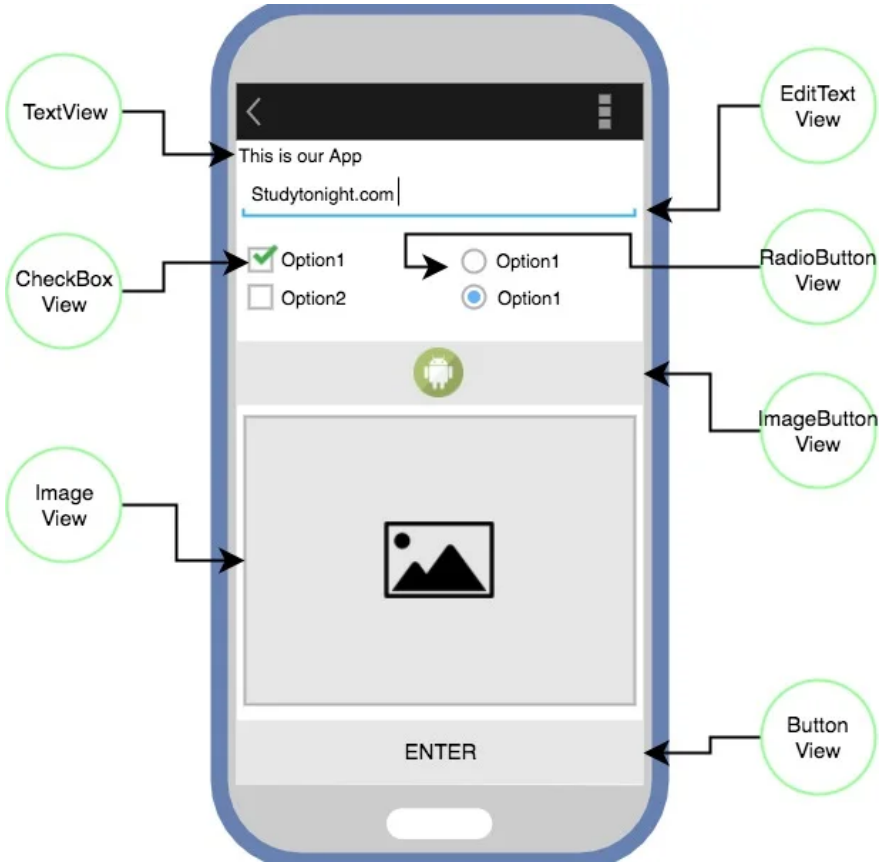
INSTITUTO FEDERAL
SANTA CATARINA

View Groups
(View Groups)

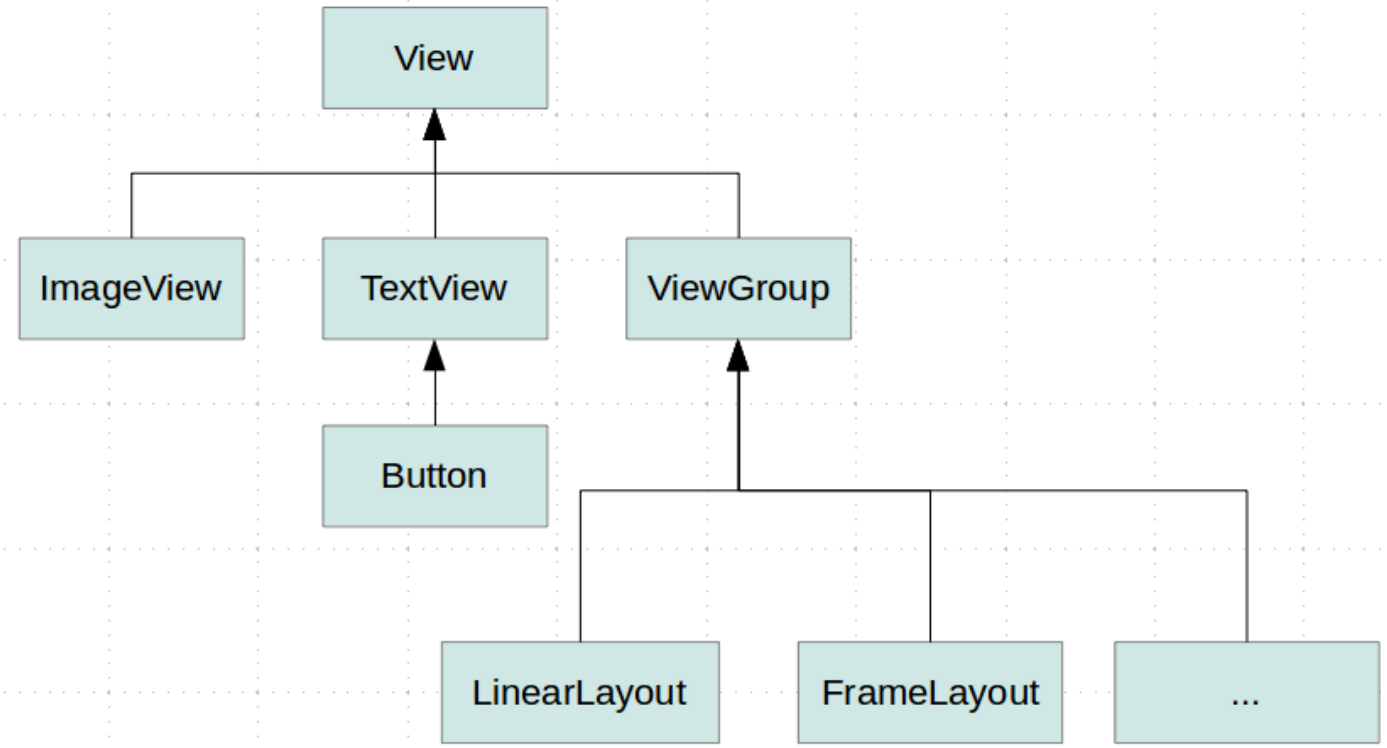
Obs: Slides são apenas guias para aulas, **é obrigatória** a leitura e resolução dos exercícios nele existentes.

View no Android

View no android é uma classe que implementa alguns métodos que possibilitam desenhar algo na tela, como o canvas. Todos widgets estendem de canvas como TextView, EditText , Button, ImageView , TougleButton , Spinner …

- 

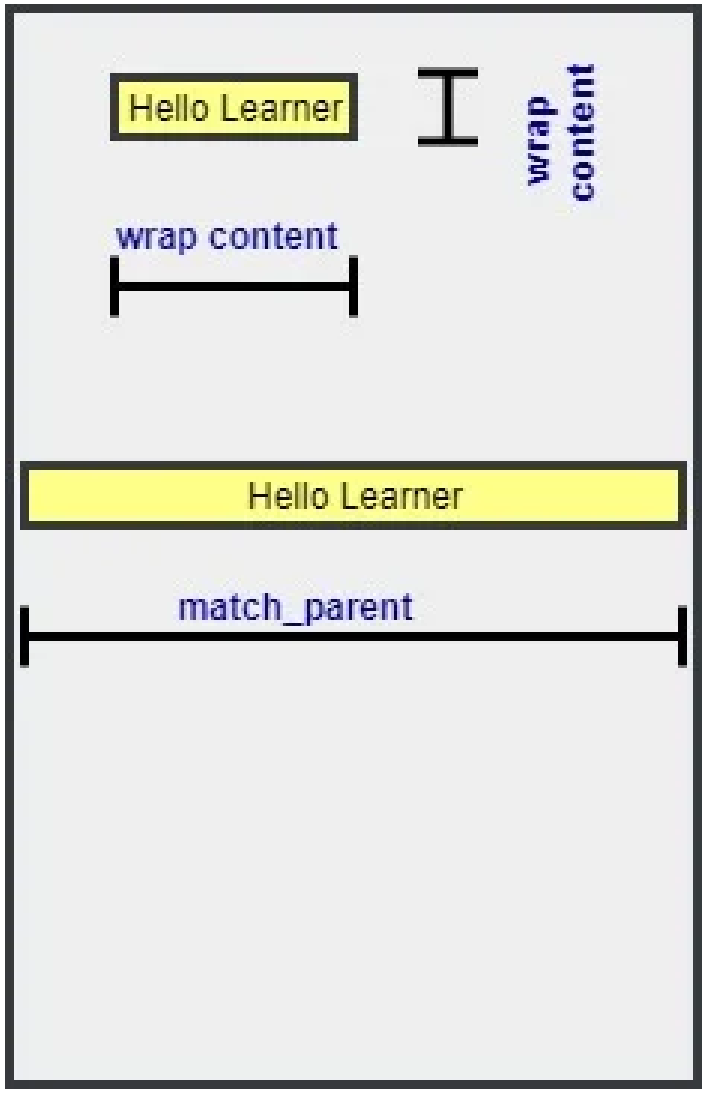
Hierarquia de classes



Basic LayoutParamns

Uma característica que deve ser definia em um View ao desenhar ele na tela é a altura e largura.

- Essa propriedades podem ter interpretações distintas de acordo com o gerenciador de layout Utilizado.



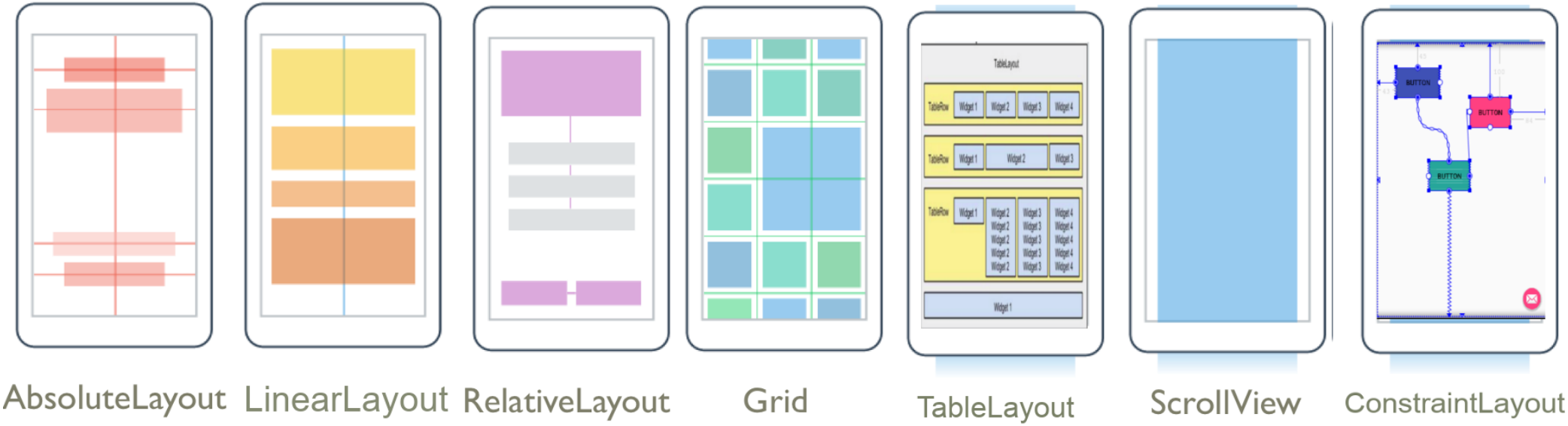
INSTITUTO FEDERAL
SANTA CATARINA

View Groups
(View Groups)

Obs: Slides são apenas guias para aulas, **é obrigatória** a leitura e resolução dos exercícios nele existentes.

Gerenciadores de *layout*
Tipos de gerenciadores disponíveis

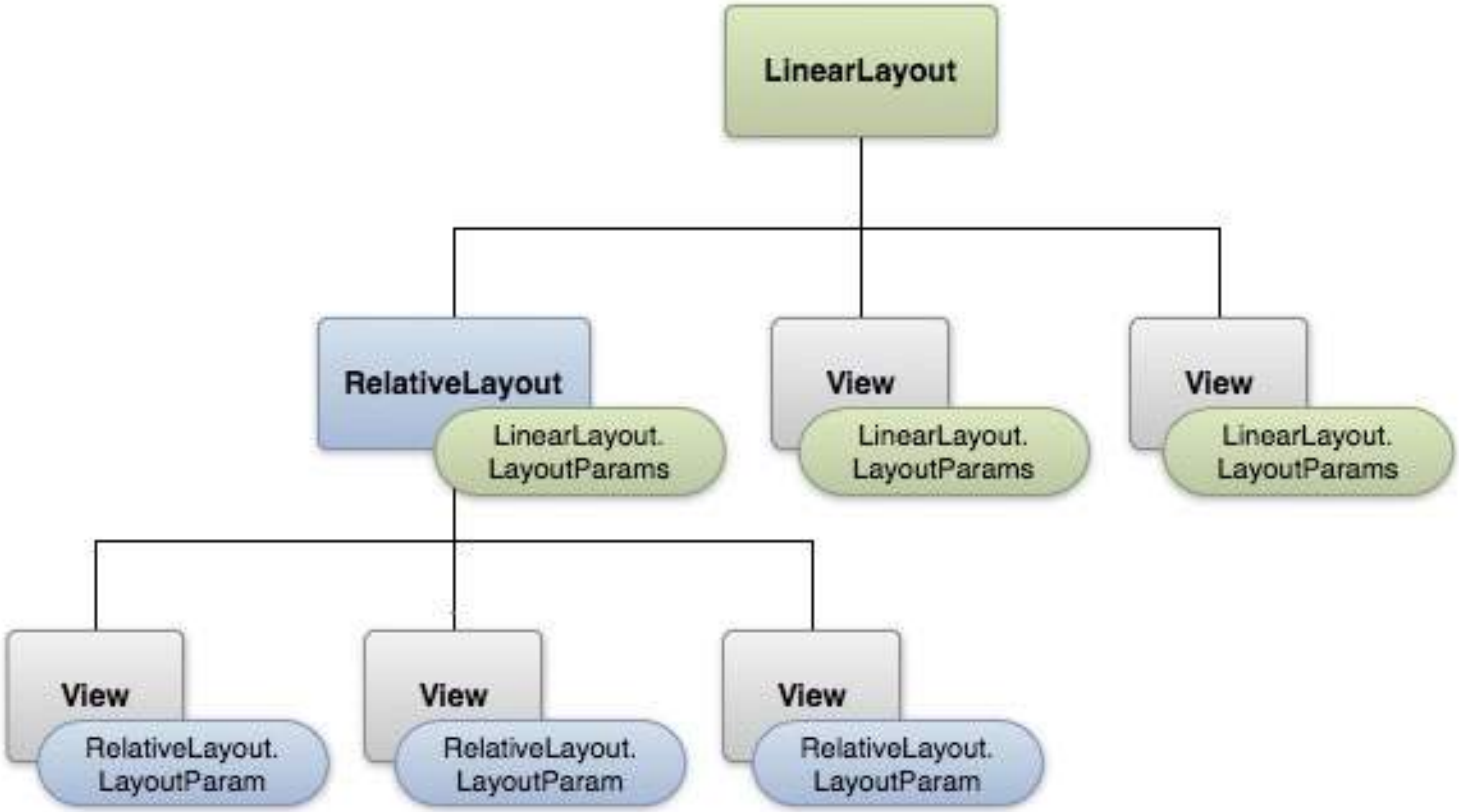
O Android prove diversos gerenciadores de *layout* diferentes, todos eles estendem da classe *ViewGroup*, e também permitem o posicionamento dos componentes *view* em um layout de acordo com algumas regras. Como por exemplo a distancia do topo, ou como uma lista de *views*.



Fonte:Autor adaptado de https://docs.microsoft.com/pt-br/xamarin/xamarin-forms/user-interface/layouts/absolute-layout

Propriedade de cada uma das view

Ao definirmos um determinado layout em uma tela as views que estão dentro desse layout carregam um objeto *LayoutParamns* que contem as configurações da View dentro do ViewGroup.

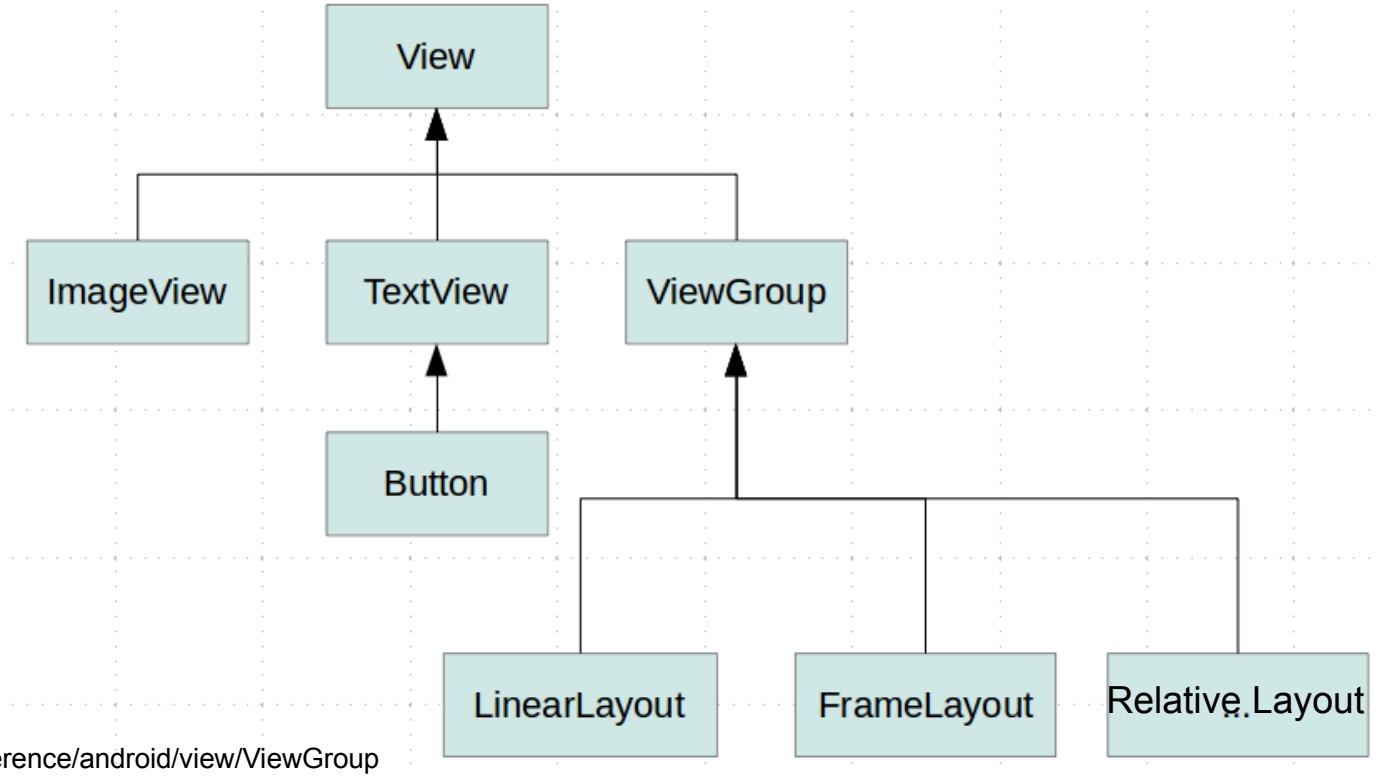


LayoutParams


```
mLinearLayout= new LinearLayout(getApplicationContext());
Toast.makeText(this, "bla", Toast.LENGTH_SHORT).show();
mTextView= new TextView(this);
mTextView.setHint("Entre com Valor");
mLinearLayout.addView(mTextView);

mButton= new Button(this);
mLinearLayout.addView(mButton);

mLinearLayout.setOrientation(LinearLayout.VERTICAL);
mLinearLayout.setHorizontalGravity(Gravity.CENTER_HORIZONTAL);

LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(
      LinearLayout.LayoutParams.WRAP_CONTENT,
      LinearLayout.LayoutParams.WRAP_CONTENT,
      LinearLayout.VERTICAL
      );
mLinearLayout.setLayoutParams(params);
params.weight=0.0f;
mButton.setLayoutParams(params);
mTextView.setLayoutParams(params);

mButton.setText("bla");
setContentView(mLinearLayout);
```

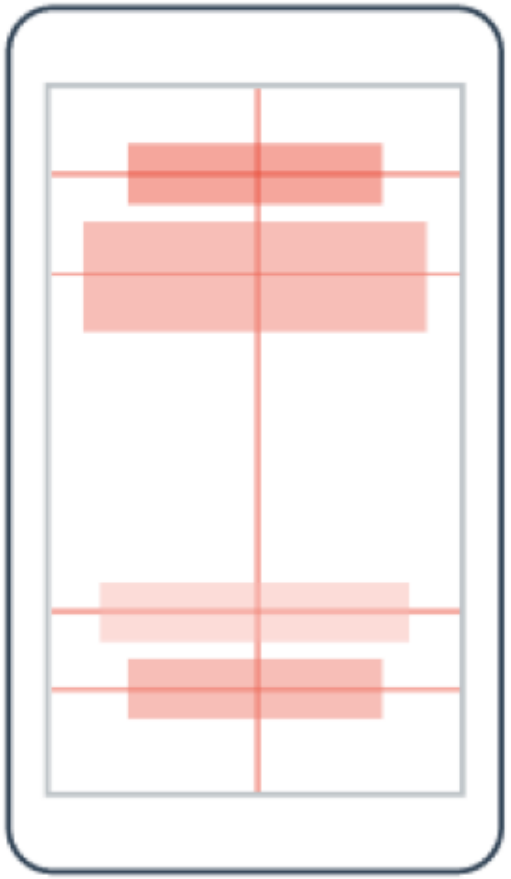
Gerenciadores de *layout*
Hierarquia da classe *ViewGroup*

Um *ViewGroup* é um componente *view* especial que pode conter outros modos de exibição (chamados de filhos). O grupo de visualizações é a classe base dos *layouts* e contêineres de visualizações. Essa classe também define a classe *ViewGroup*.*LayoutParams*, que serve como a classe base para os parâmetros dos layouts.



Fonte: https://developer.android.com/reference/android/view/ViewGroup

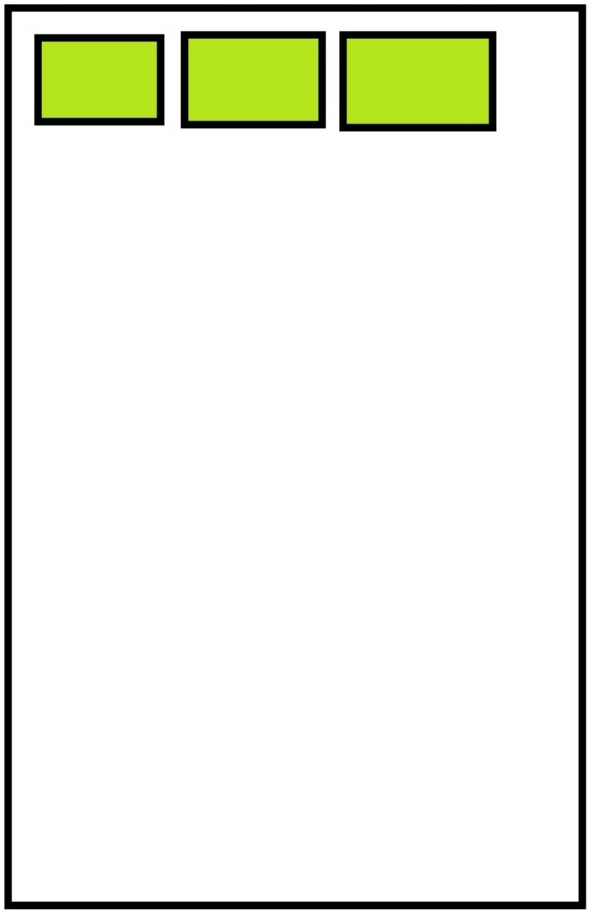
AbsoluteLayout é um gerenciador de layout em que os elementos visuais são posicionados de forma absoluta. Assim, a posição de cada view é definido por um par de coordenadas (X,Y) em relação a origem (0,0).



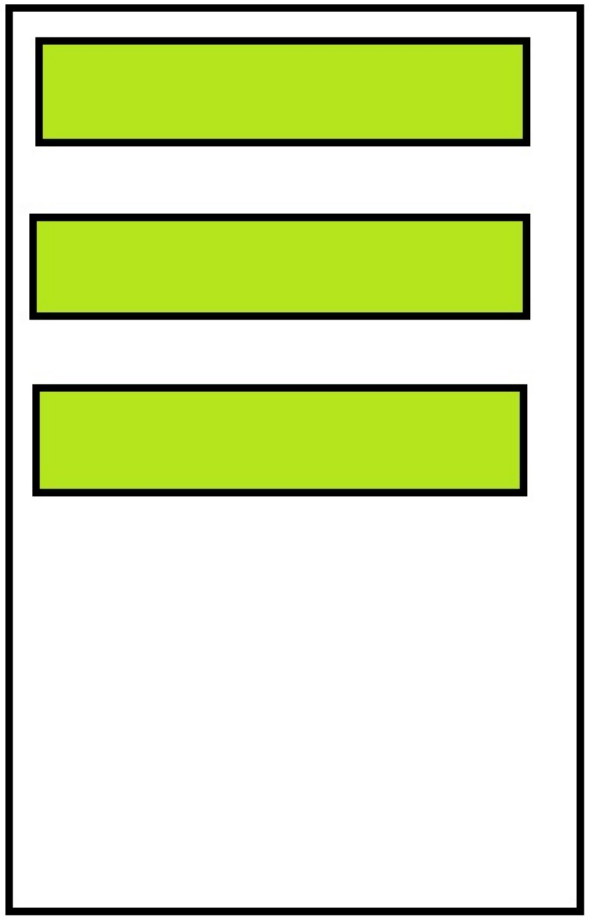
AbsoluteLayout

Gerenciadores de *layout*
Linear Layout

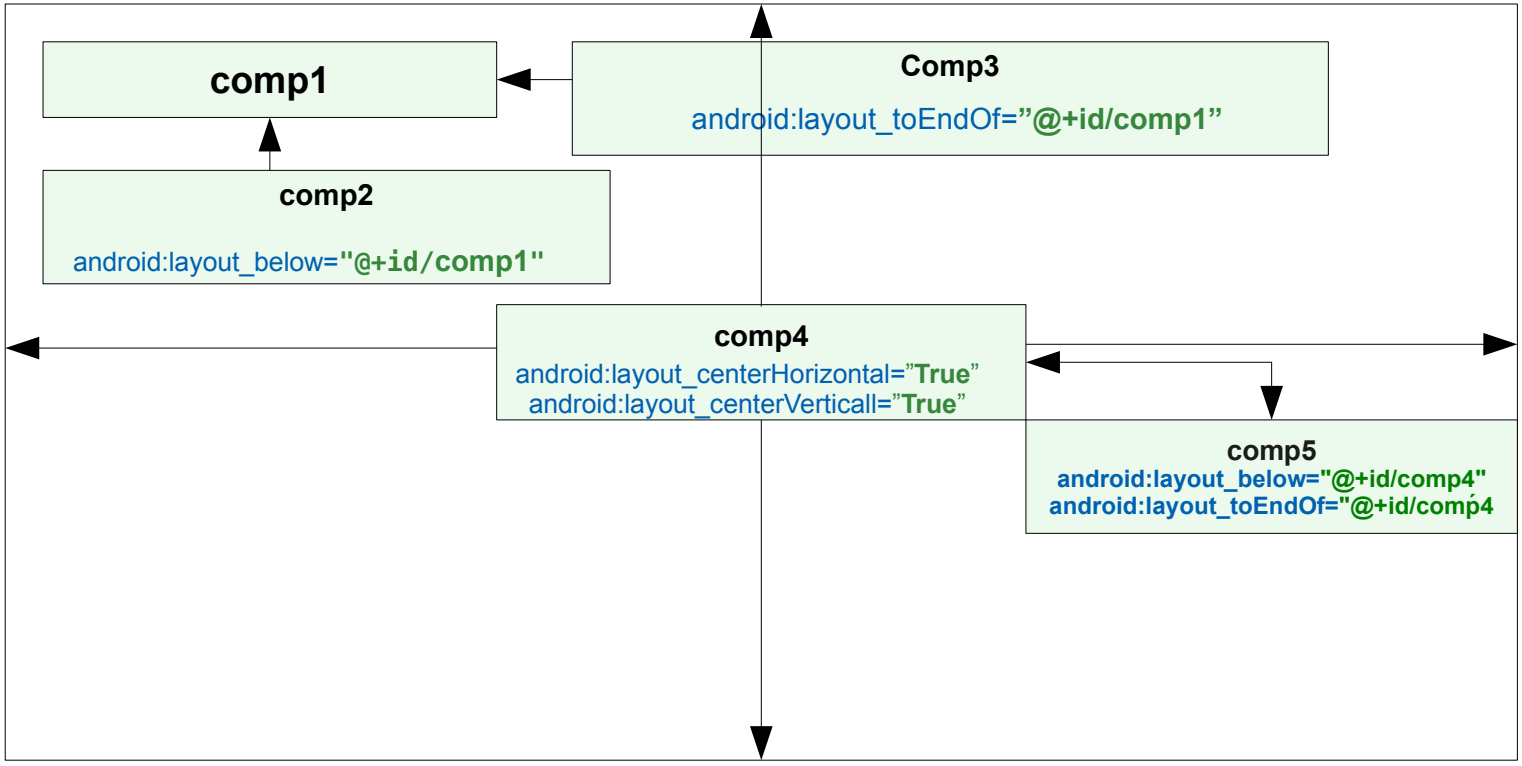
Linear Layout Horizontal



Linear Layout Vertical

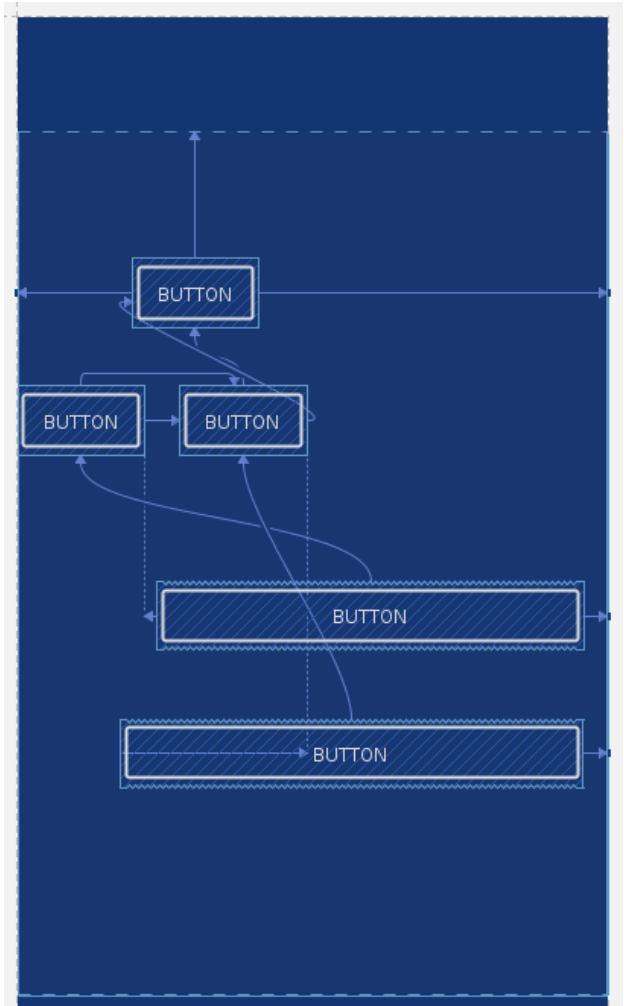


O RelativeLayout é um *ViewGroup* que exibe *views* em posições relativas. A posição de cada visualização pode ser especificada em relação as outros componentes vizinhos, a esquerda, direita, topo ou abaixo, ou ainda em relação a o pai que é o *RelativeLayout,*o que possibilita alinhar á parte inferior, esquerda ou central (DeveloperAndroid,2018).



Gerenciadores de *layout*
Constranit Layout

No *Constraint Layout* o posicionamento das *views* é feito de forma relativa, parecido com *RelativeLayout,* entretanto as regras são mais flexíveis com foco na criação de layouts responsivos.



Resumo sobre *Views Groups*

A SDK do *android* oferece diversos gerenciadores de layout, que possibilitam posicionar as *View* de acordo com regras pré-fixadas como é o caso do linear layout ou com relações e restrições.

Embora não existe uma regra para escolha do ViewGroup, a forma mais adequada de avaliar deve-se levar em consideração a responsividade e desempenho.

INSTITUTO FEDERAL
SANTA CATARINA

Declaração do Layout

Obs: Slides são apenas guias para aulas, **é obrigatória** a leitura e resolução dos exercícios nele existentes.

Declaração do *ViewGroup*

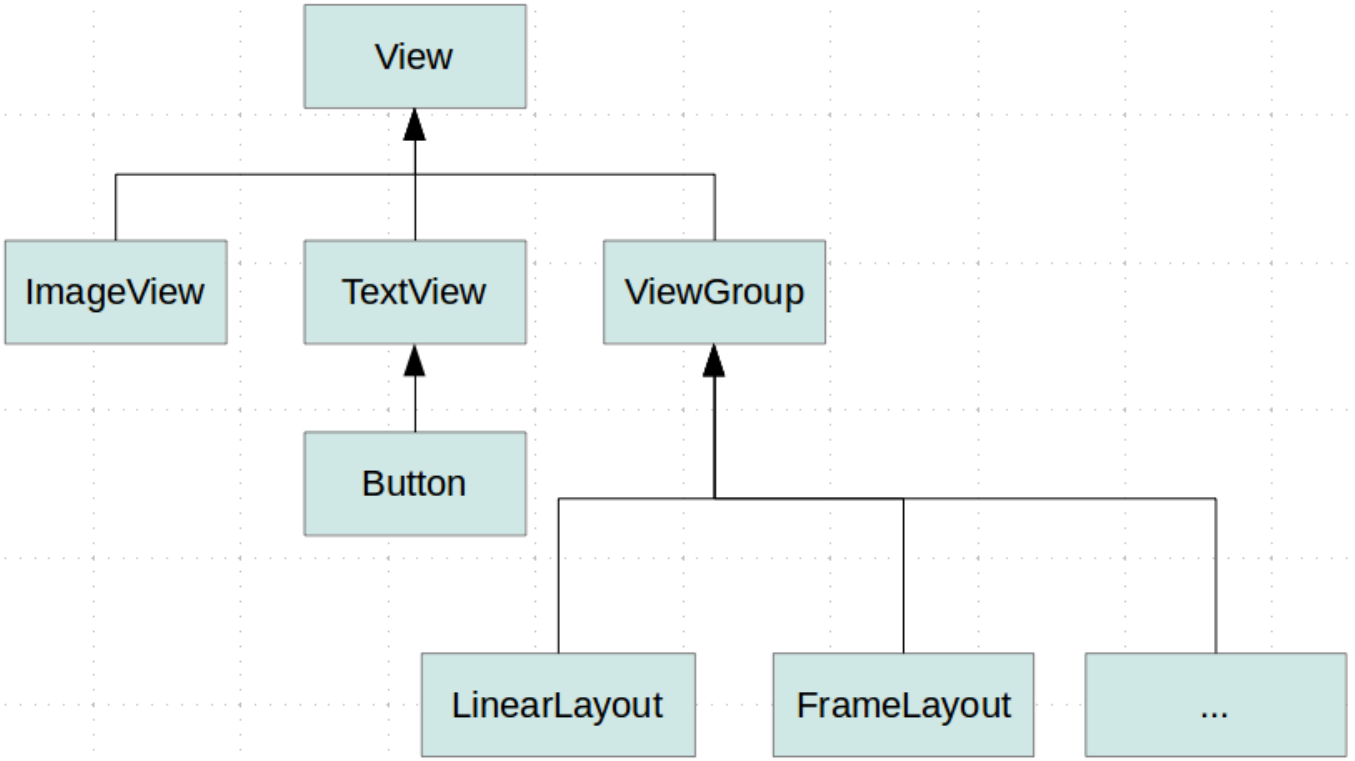
No Android a definição de um layout pode ser realizada de duas formas:

- Declarar elementos da IU em XML. O Android fornece um vocabulário XML direto que corresponde às classes e subclasses de View, como as de widgets e layouts.

- Instanciar elementos do layout em tempo de execução. O aplicativo pode criar objetos View e ViewGroup (e processar suas propriedades) programaticamente.

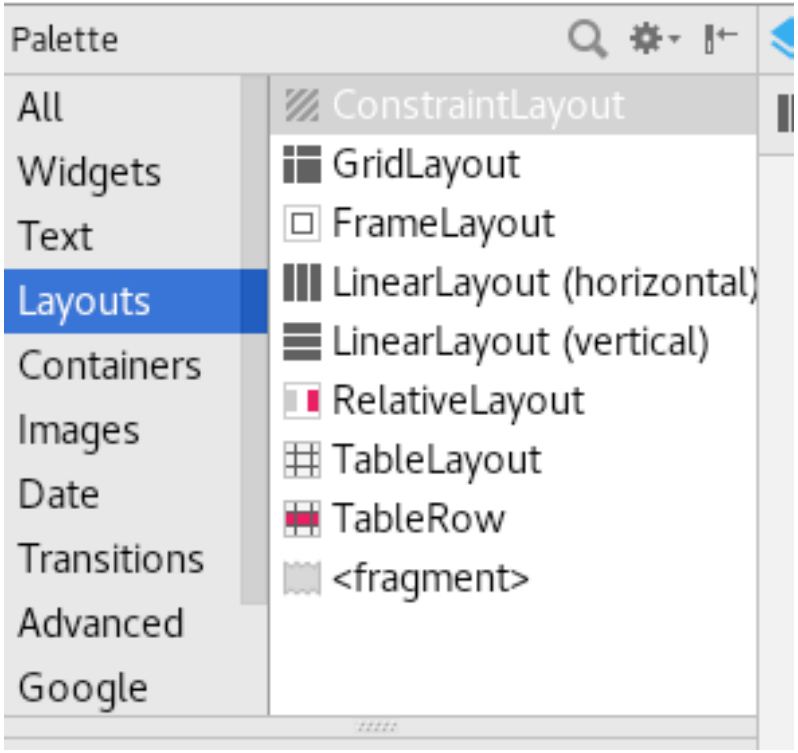
View e *ViewGroups* Estrutura de classes

- Um gerenciador de layout é utilizado para **organizar a disposição dos componentes** na tela automaticamente;
- Os gerenciadores de layout mais conhecidos são:
  - *AbsoluteLayout*;
  - *RelativeLayout*;
  - *ConstraintLayout*;
  - *LinearLayout*;
  - *FrameLayout*;
  - *TableLayout*;

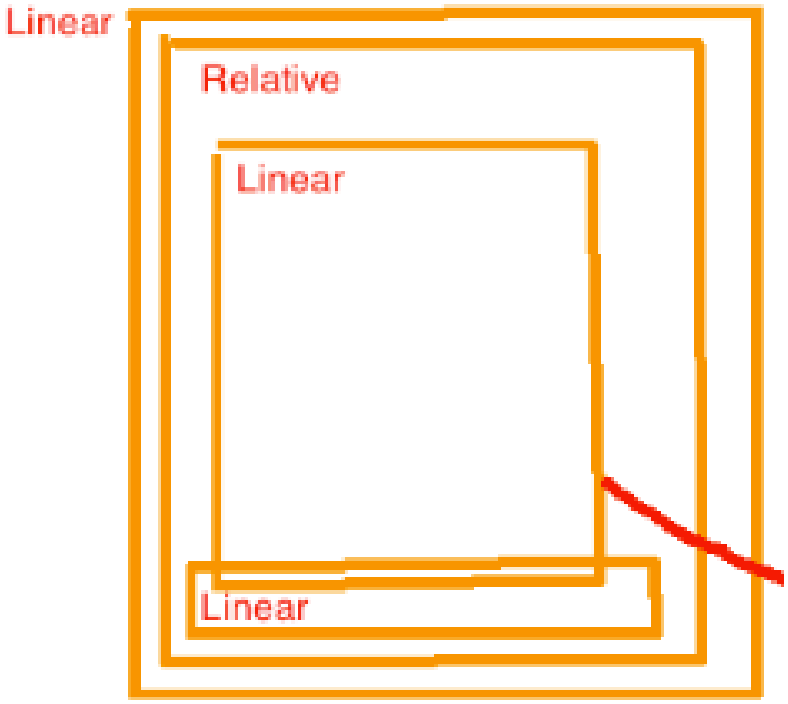


Quais layout minha SDK oferece suporte

Podemos verficiar os layout que são suportados pela versão da SDK Tools na IDE por meio da paleta de componentes, no item layout



ViewGroups



Como um *ViewGroup* é uma subclasse de víew, e ele contém *views* podemos ter um *viewGroup* dentro do outro.

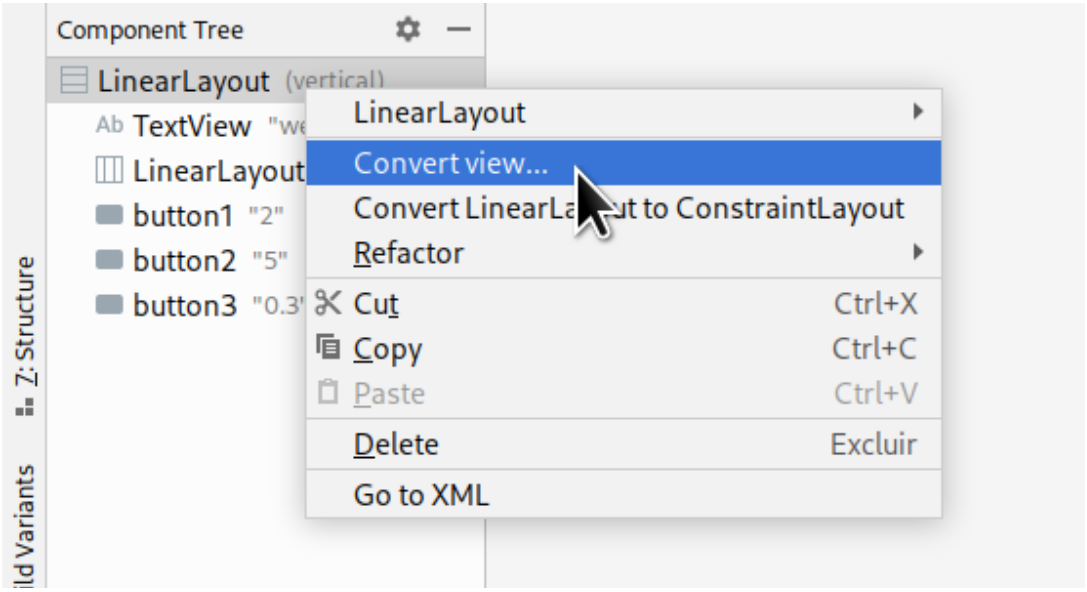
Altere a definição do tipo de layout

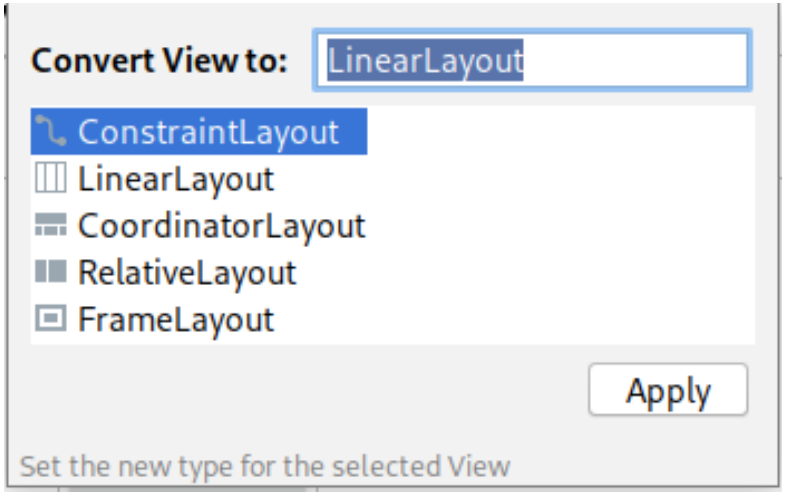
```
<AbsoluteLayout
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:id="@+id/AbsoluteLayout1"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
</AbsoluteLayout>
```

Observe que estamos definindo que o Layout ocupa todo espaço

Ferramenta para alteração de *Layout*

Podemos alterar o *viewgroup* de uma *layout* modificando o XML com a declaração, ou utilizando uma ferramenta que pode ser acessada a partir da ferramenta de *Desgn*.





Referências

Básica

Livro Deitel, P., Deitel, H., Deitel, A. e Morgano, M.. Android para Programadores.. 1a ed.. Bookman. 2012

Livro Darwin, I. F. Android Cookbook.. 1ª. Novatec,,. 2013

Complementar

Livro Saudate, A.. SOA Aplicado: Integrando com WebServices e além. 1ª. São Paulo: Casa do código. 2012

Livro Stark, J e Jepson, B.. Construindo Aplicativos Android com HTML, CSS e JavaScript.. 1ª. Novatec. 2012

Livro Querino Filho, L. C.. Desenvolvendo seu Primeiro Aplicativo Android.. 1ª. Novatec. 2013

Livro Lecheta, R. Google Android: Aprenda a criar aplicações para dispositivos móveis. 3a Ed. Novatec, 2013.

http://developer.android.com/