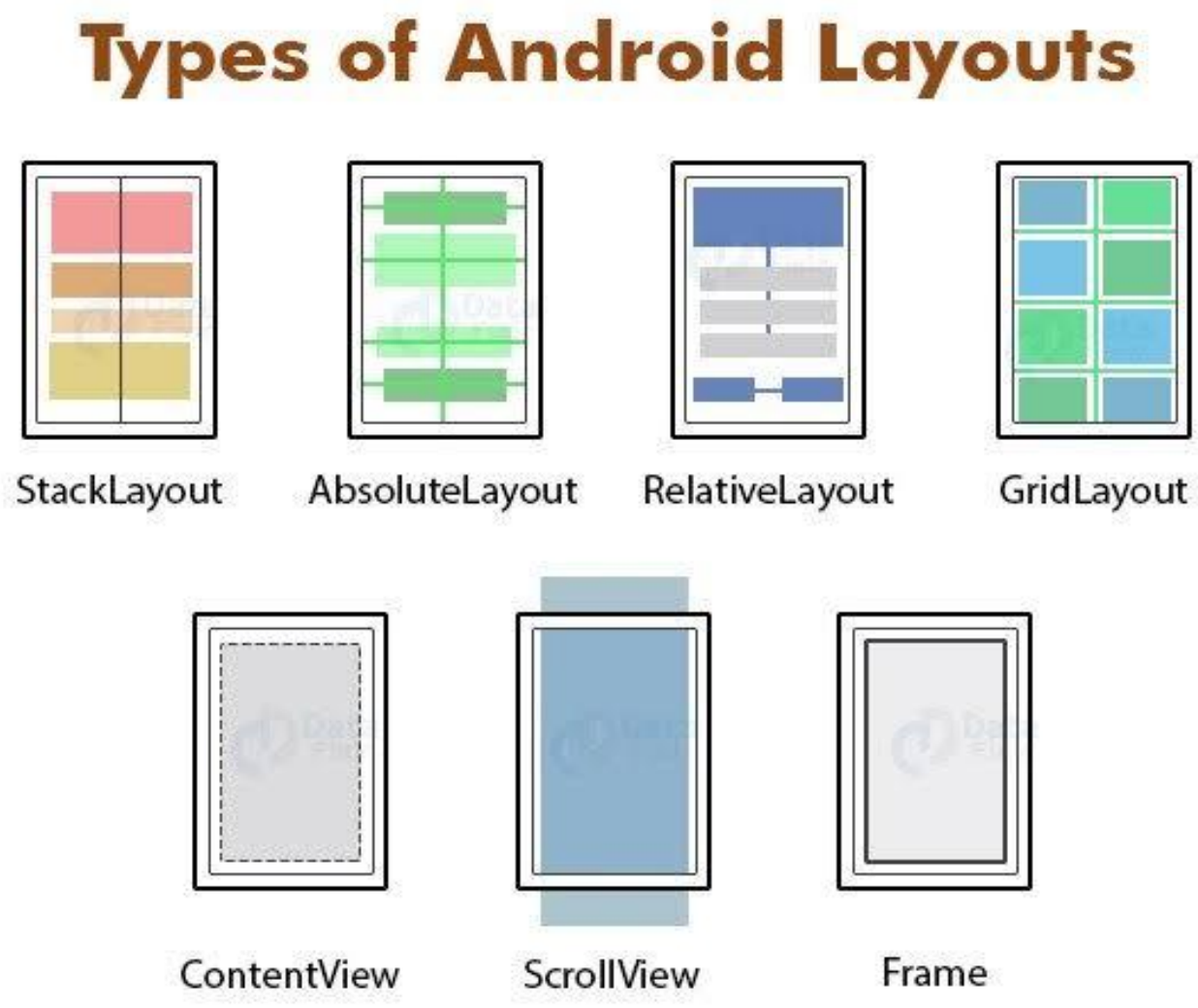


1. **StackLayout**
   A StackLayout is **a layout that organizes its children in a one-dimensional stack, either horizontally or vertically**. By default, a StackLayout is oriented vertically.

2. **Absolute Layout**
   An Absolute Layout allows you to specify the exact location . i.e., X and Y coordinates of its children with respect to the origin at the top left corner of the layout. The absolute layout is less flexible and harder to maintain.

3. **Relative Layout**

RelativeLayout is **a view group that displays child views in relative positions**. The position of each view can be specified as relative to sibling elements (such as to the left-of or below another view) or in positions relative to the parent RelativeLayout area (such as aligned to the bottom, left or center).

4. **Content Layout**
   Stay organized with collections **Save and categorize content based on your preferences**. Annotation that can be attached to a constructor with a single LayoutRes parameter to denote what layout the component intends to inflate and set as its content.

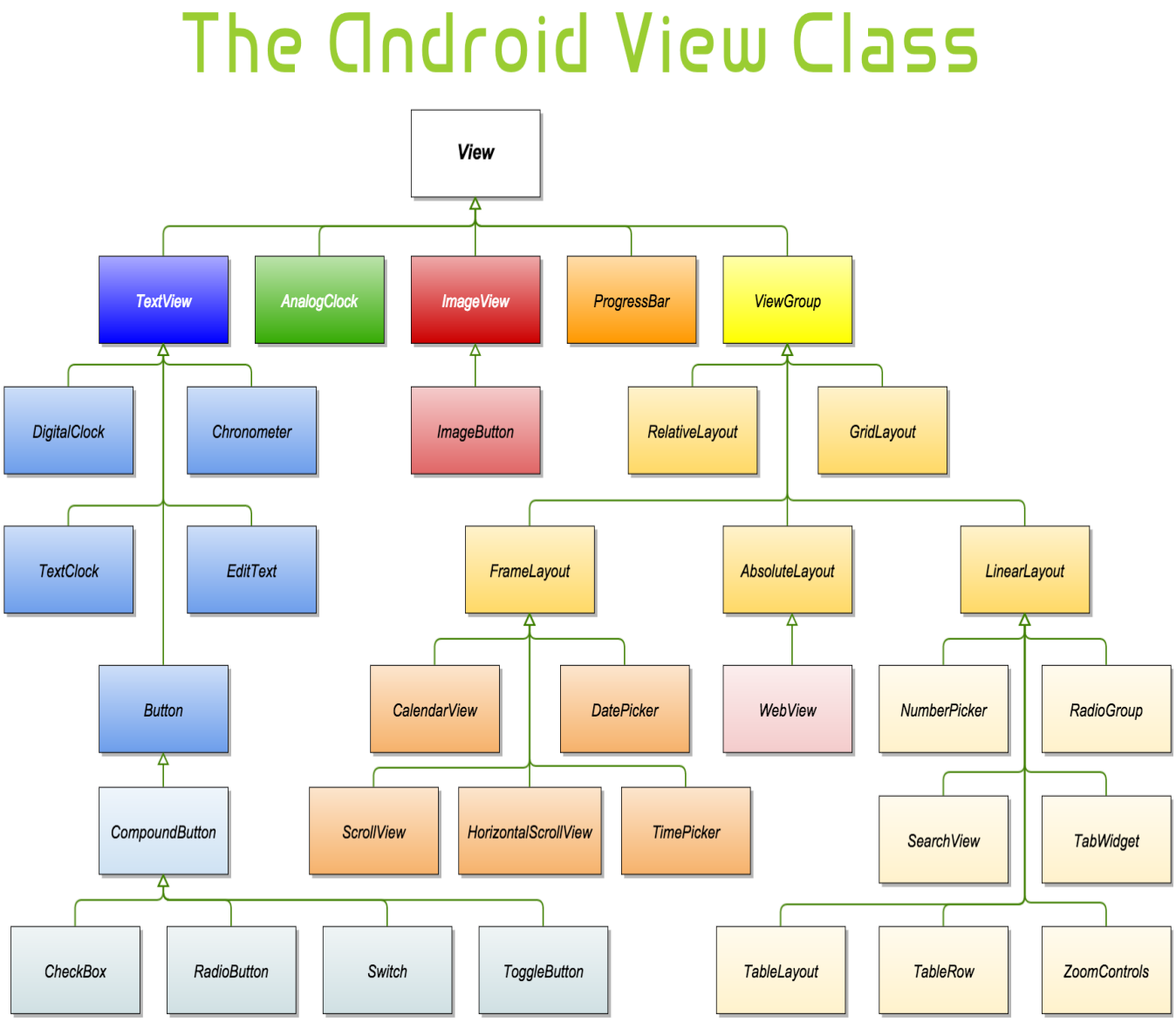
5. **Grid Layout**

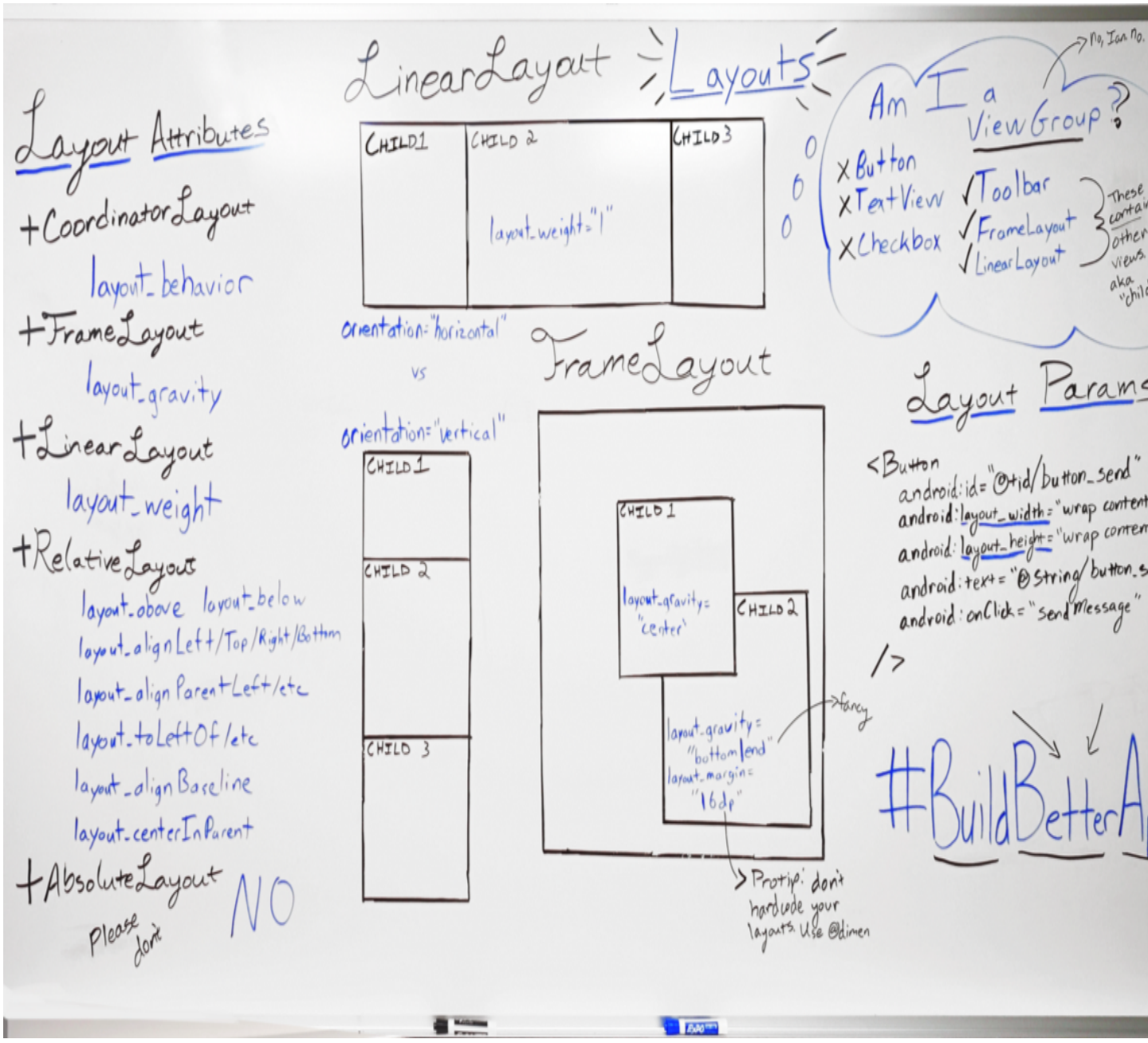
A GridView is a type of AdapterView that displays items in a two-dimensional scrolling grid. Items are inserted into this grid layout from a database or from an array. The adapter is used for displaying this data, **the setAdapter()** method is used to join the adapter with GridView. The main function of the adapter in GridView is to fetch data from a database or array and insert each piece of data in an appropriate item that will be displayed in GridView.

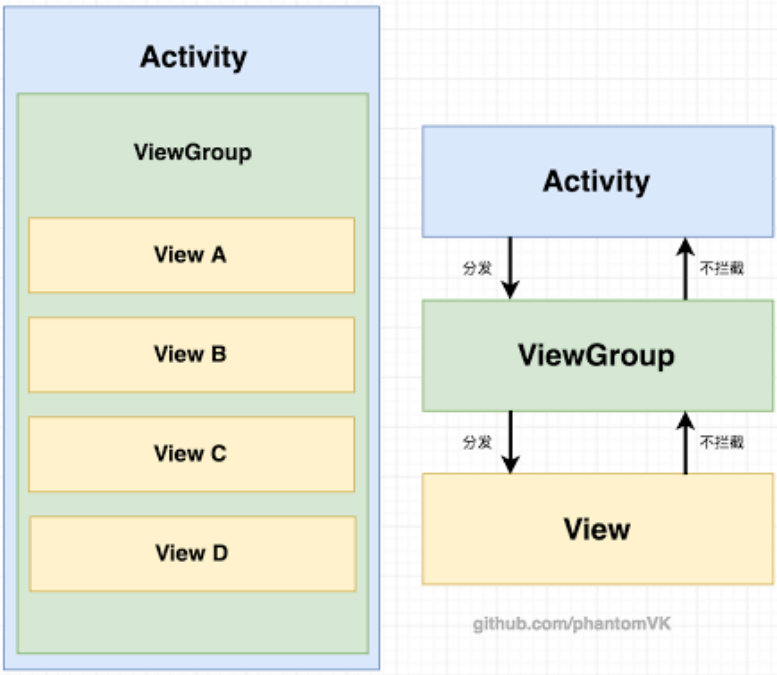
6. **Scroll View**
   a ScrollView is **a view group that is used to make vertically scrollable views**. A scroll view contains a single direct child only. In order to place multiple views in the scroll view, one needs to make a view group(like LinearLayout) as a direct child and then we can define many views inside it.

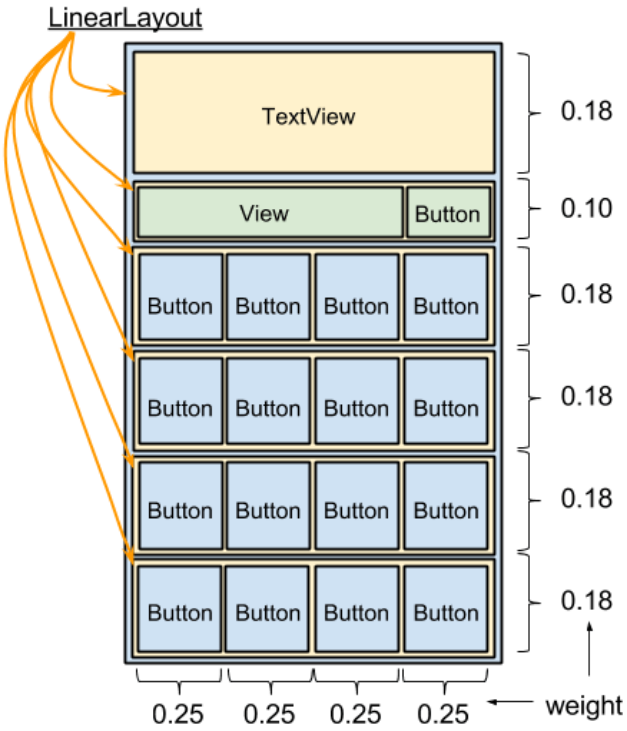
7. **Fram**
   Frame Layout is **designed to block out an area on the screen to display a single item**. Generally, FrameLayout should be used to hold a single child view, because it can be difficult to organize child views in a way that's scalable to different screen sizes without the children overlapping each other.

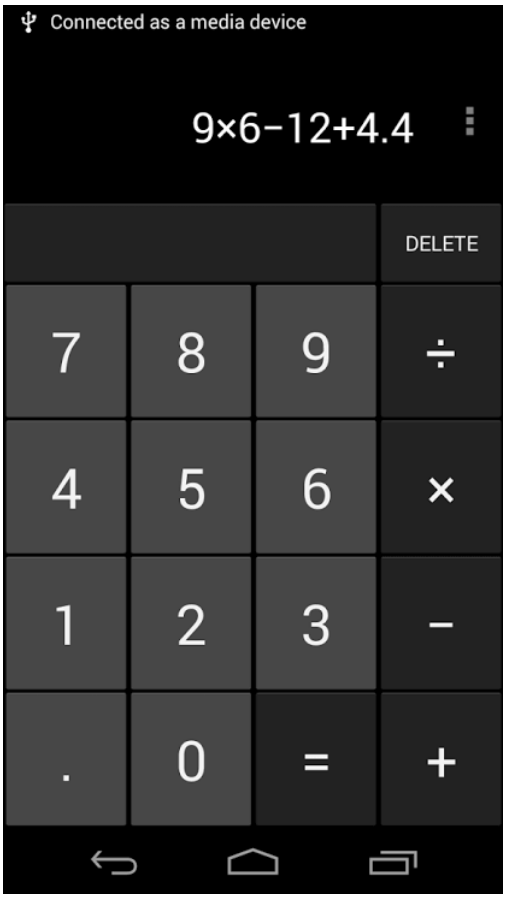
8. **Difference between Relative Layout and Absolute Layout**
   RelativeLayout : is a ViewGroup that displays child views in relative positions. **AbsoluteLayout : allows us to specify the exact location of the child views and widgets**.




Layout Attributes
+CoordinatorLayout
layout_behavior
+FrameLayout
layout_gravity
+LinearLayout
layout_weight
+RelativeLayout
layout_above layout_below
layout_alignLeft/Top/Right/Bottom
layout_alignParentLeft/etc
layout_toLeftOf/etc
layout_alignBaseline
layout_centerInParent
+AbsoluteLayout
please don't
NO
LinearLayout
CHILD1
CHILD 2
CHILD 3
layout_weight="1"
orientation="horizontal"
vs
orientation="vertical"
CHILD 1
CHILD 2
CHILD 3
FrameLayout
CHILD 1
layout_gravity="center"
CHILD 2
layout_gravity="bottom|end"
layout_margin="16dp"
fancy
Protip: don't hardcode your layouts. Use @dimen
Layouts
Am I a ViewGroup?
No, I am no.
X Button
X TextView
X Checkbox
√ Toolbar
√ FrameLayout
√ LinearLayout
These contain other views. aka "chil
Layout Params
<Button
android:id="@+id/button_send"
android:layout_width="wrap content
android:layout_height="wrap conten
android:text="@string/button_s
android:onClick="sendMessage"
/>
#BuildBetterA


Activity
ViewGroup
View A
View B
View C
View D
Activity
分发
不拦截
ViewGroup
分发
不拦截
View
github.com/phantomVK


LinearLayout
TextView
View
Button
0.18
0.10
0.18
0.18
0.18
0.18
0.25
0.25
0.25
0.25
weight


Connected as a media device
9×6−12+4.4
DELETE
7
8
9
÷
4
5
6
×
1
2
3
−
.
0
=
+