LẬP TRÌNH ANDROID CĂN BẢN

Bài 3: GUI & Layout

Ths. Trần Xuân Thanh Phúc | Trường Đại học Công Nghiệp Thực Phẩm

Mục tiêu

- Hiểu rõ về GUI và các Layout.
- Tự thiết kế giao diện ứng dụng

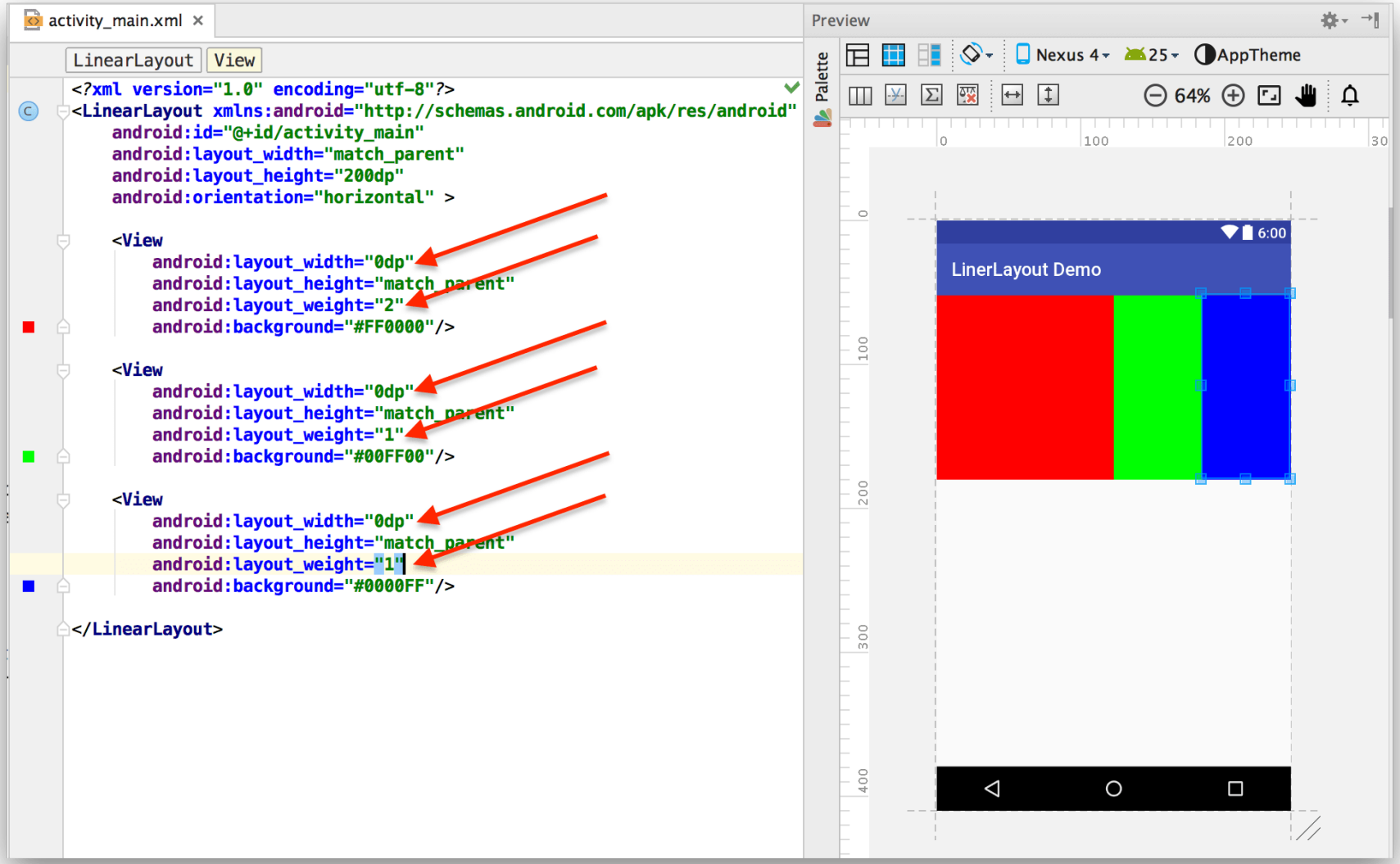
Nội dung

- GUI
- Layout

1.1 GUI (Graphical User Interface)

- Android GUI thường được implement bởi các XML file (hoặc từ Java/Kotlin code) (View)
- View class là thành phần cơ bản nhất của Android để tạo ra các đối tượng giao diện người dùng
- View là 1 một vùng hình chữ nhật trên màn hình, chịu trách nhiệm việc vẽ và xử lý sự kiện.
- Widget là các sub-class của View. Thường là các đối tượng có thể tương tác với người dùng như button, checkbox…
- Layout là một vật chứa (container) dùng để chứa các View hoặc Layout con.

1.2 XML Layout

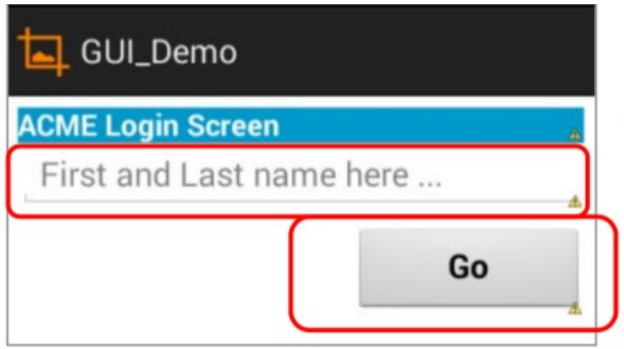


1.2 XML Layout

- Một XML view file thường bao gồm 1 layout để sắp xếp các thành phần con bên trong nó
- Activity dùng phương thức **setContentView(R.layout.xml_file_name)** để render 1 view lên màn hình thiết bị

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal" >

    Widgets and other nested layouts

</LinearLayout>
```

1.2 XML Layout – Retrieve view instance




```xml
<!- XML LAYOUT -->
<LinearLayout
     android:id="@+id/myLinearLayout"
     ... >

    <TextView
        android:text="ACME Login Screen"
        ...  />

    <EditText
        android:id="@+id/edtUserName"
        ... />

    <Button
        android:id="@+id/btnGo"
        ...  />

</LinearLayout>
```

res/layout/activity_main.xml

Java code

```java
package edu.csc.gui;
import android…;

public class MainActivity extends Activity {

    EditText edtUserName;

    Button   btnGo;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);

     setContentView(R.layout.activity_main);

     edtUserName = (EditText) findViewById(R.id.edtUserName);

     btnGo = (Button) findViewById(R.id.btnGo);
     ...

   }

 ...
}
```

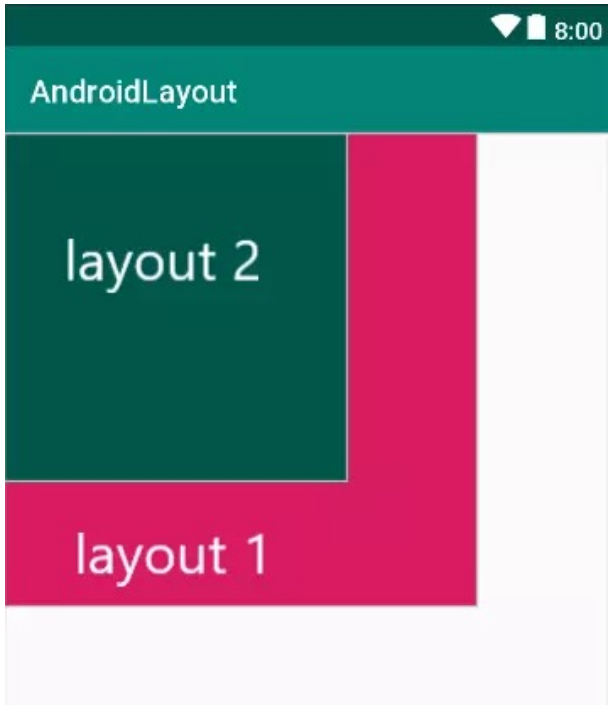
MainActivity.java

1.3 Layout

- Layout là thành phần định nghĩa cấu trúc giao diện người dùng hay nói cách khác là thành phần quyết định đến giao diện của một màn hình trong ứng dụng Android. Layout hỗ trợ việc căn chỉnh các widget (Ví dụ: TextView, Button, hay EditText…) theo một quy luật nhất định.
  - Framelayout
  - LinearLayout
  - RelativeLayout
  - GridLayout
  - TableLayout
  - ConstraintLayout

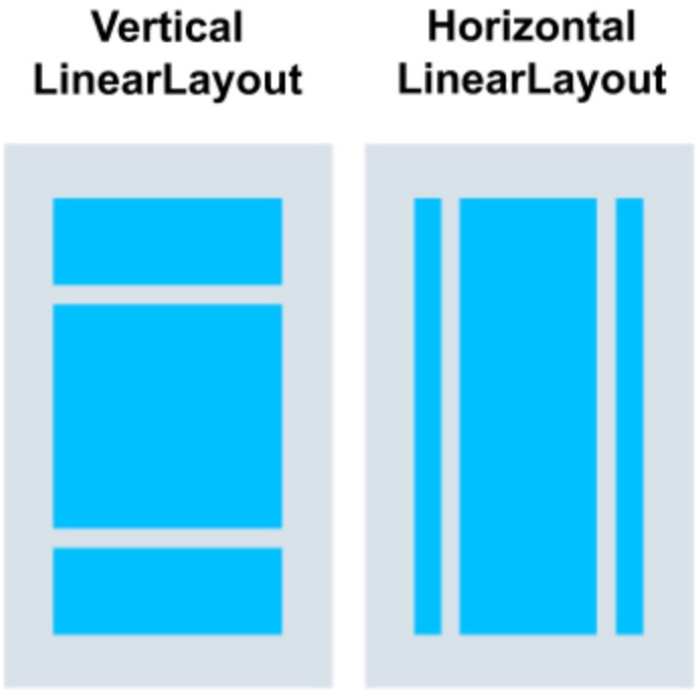
1.3.1 FrameLayout

- Là dạng layout cơ bản nhất khi gắn các view lên layout này thì nó sẽ luôn giữ các view này ở phía góc trái màn hình và không cho chúng ta thay đổi vị trí của chúng, các view đưa vào sau sẽ đè lên view ở trước trừ khi bạn thiết lập **transparent** cho view sau đó.



1.3.2 LinearLayout

- LinearLayout là loại layout sẽ sắp xếp các view theo chiều dọc hoặc ngang theo thứ tự của các view.



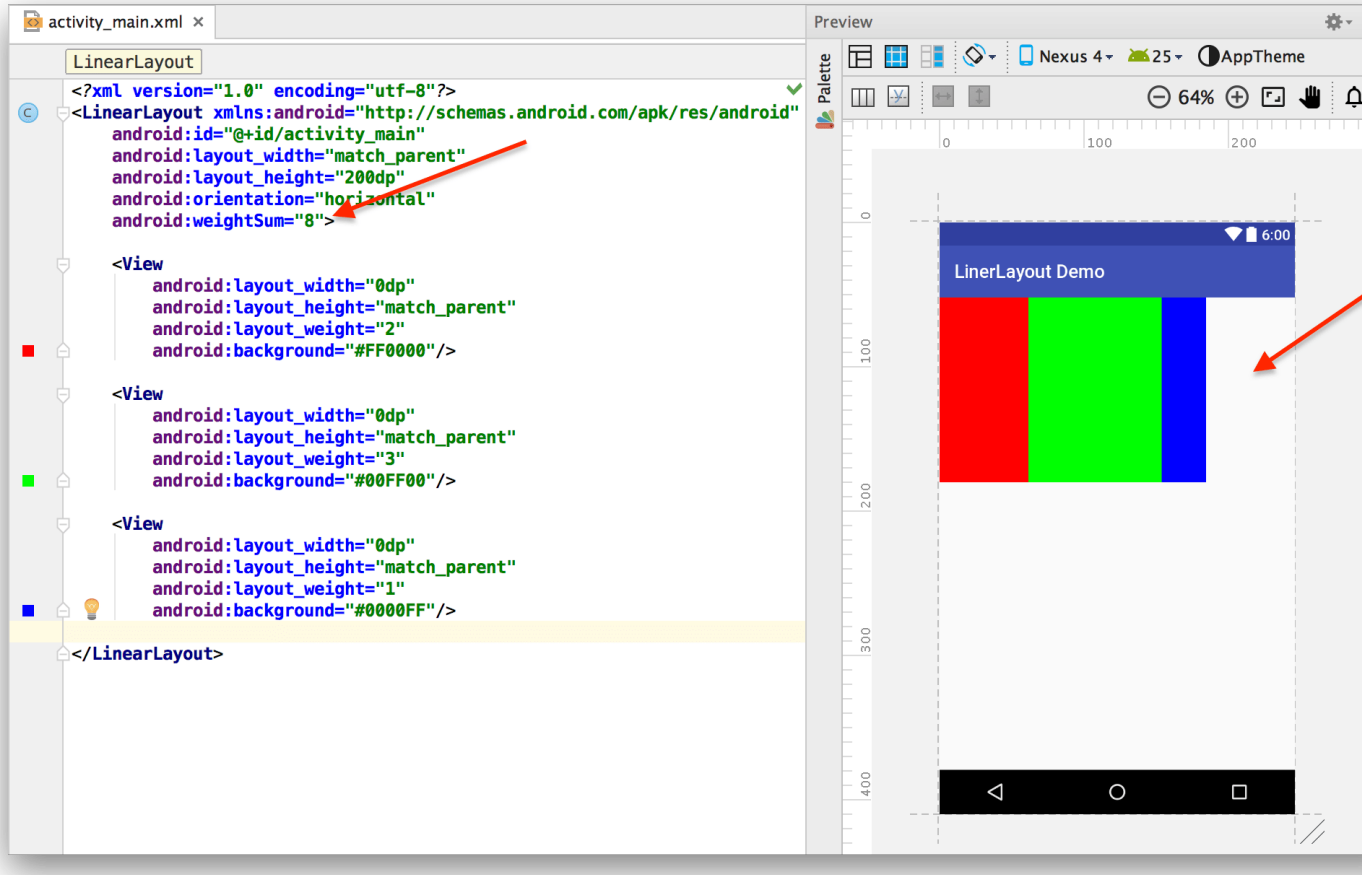
1.3.2 LinearLayout – Gravity

- Thuộc tính android:gravity để căn chỉnh các View nằm ở vị trí nào trong LinearLayout (mặc định là start-top). Giá trị có thể tổ hợp bằng |

center	Căn ở giữa
top	Ở phần trên
bottom	Phần dưới
center_horizontal	Ở giữa theo chiều ngang
center_vertical	Ở giữa theo chiều đứng
left	Theo cạnh trái
right	Theo cạnh phải
bottom	Cạnh dưới

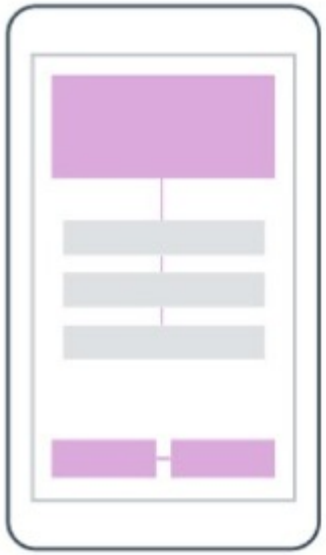
1.3.2 LinearLayout – Weight

- Các View con trong LinearLayout có thể gán cho nó một giá trị trọng số bằng thuộc tính **android:layout_weight**. Nếu View không gán giá trị này coi như nó có trọng số bằng không.



1.3.3 RelativeLayout

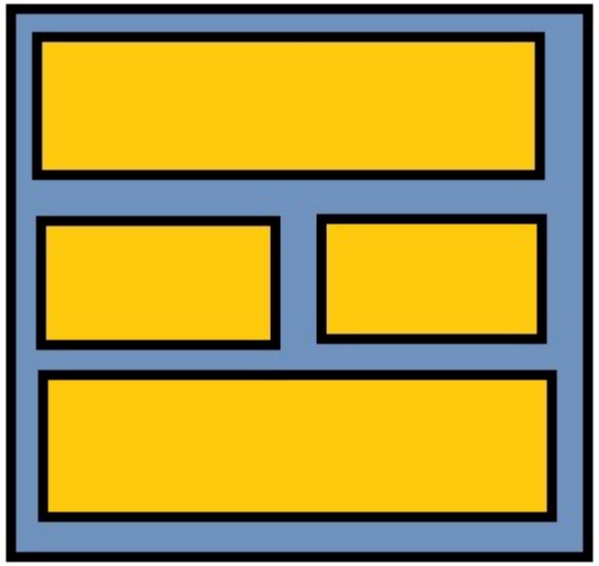
- RelativeLayout là một ViewGroup có hiển thị các View con ở các vị trí tương đối. Vị trí của mỗi View có thể được quy định liên quan đến các View anh em (như bên trái của hoặc bên dưới một View khác) hoặc ở các vị trí tương đối với khu vực cha RelativeLayout(chẳng hạn như sắp xếp ngay phía dưới, bên trái hoặc trung tâm).



1.3.3 RelativeLayout - Gravity

- Các View con có thể dịch chuyển tới những vị trí nhất định trong RelativeLayout bằng thuộc tính: android:gravity, nó nhận các giá trị (có thể tổ hợp lại với ký hiệu |)

Giá trị	**Ý nghĩa**
center	Căn ở giữa
top	Ở phần trên
bottom	Phần dưới
center_horizontal	Ở giữa theo chiều ngang
center_vertical	Ở giữa theo chiều đứng
left	Theo cạnh trái
right	Theo cạnh phải
bottom	Cạnh dưới

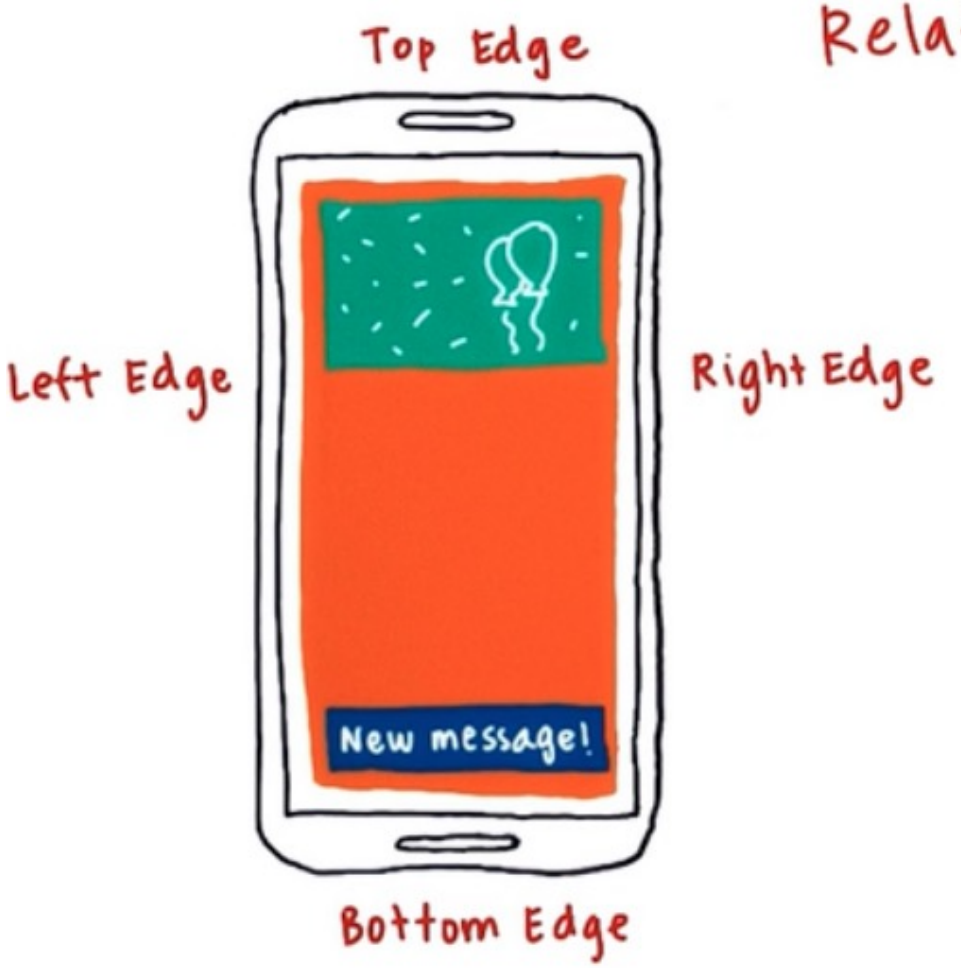


1.3.3 RelativeLayout - Định vị view con bằng view cha

- Vị trí của View con trong RelativeLayout có thể thiết lập bằng cách chỉ ra mối liên hệ vị trí với view cha, như căn thẳng cạnh trái, căn thẳng cạnh phải View cha…

Thuộc tính	**Ý nghĩa**
android:layout_alignParentBottom	true căn thẳng cạnh dưới view con với cạnh dưới View cha
android:layout_alignParentLeft	true căn thẳng cạnh trái view con với cạnh trái View cha
android:layout_alignParentRight	true căn thẳng cạnh phải view con với cạnh phải View cha
android:layout_alignParentTop	true căn thẳng cạnh trên view con với cạnh trên View cha
android:layout_centerInParent	true căn view con vào giữa View cha
android:layout_centerHorizontal	true căn view con vào giữa View cha theo chiều ngang
android:layout_centerVertical	true căn view con vào giữa View cha theo chiều đứng

1.3.3 RelativeLayout - Định vị view con bằng view cha



Relative to Parent

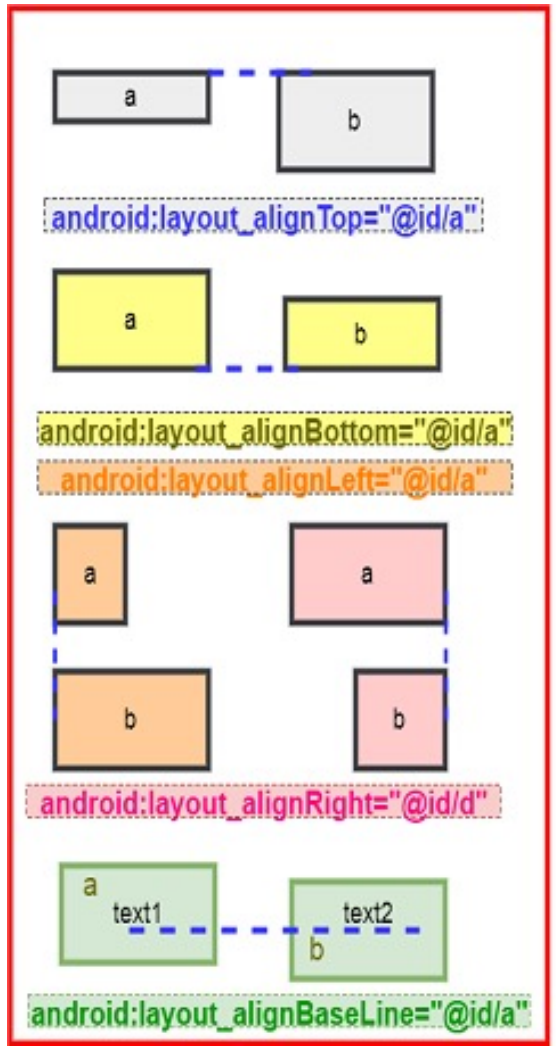
TextView attributes:

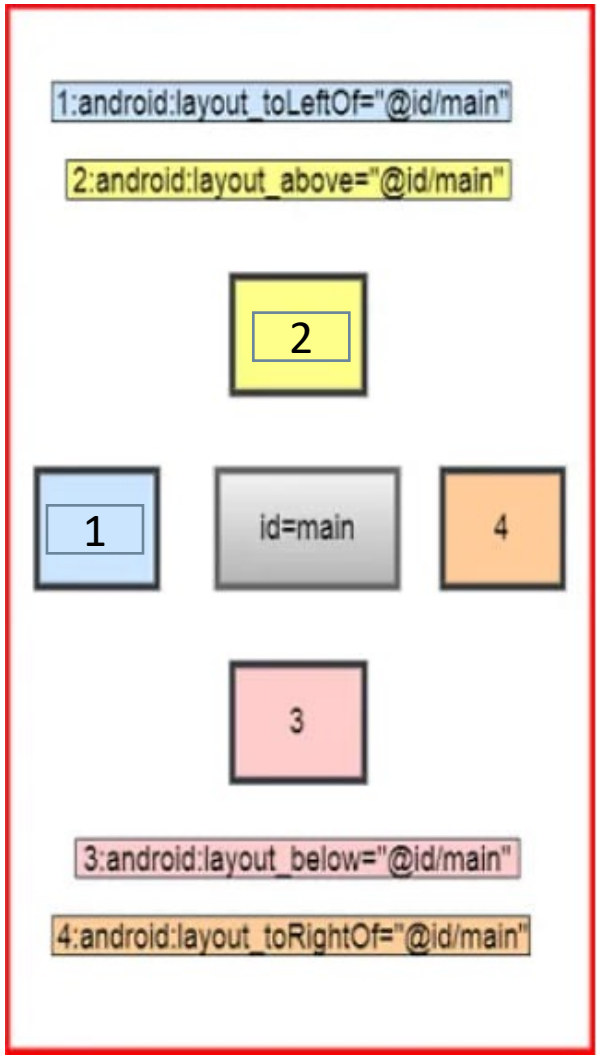
```
android:layout_alignParentTop = "false"
android:layout_alignParentBottom = "true"
android:layout_alignParentLeft = "true"
android:layout_alignParentRight = "true"
```

1.3.3 RelativeLayout - Định vị các view con với nhau

Thuộc tính	Ý nghĩa
android:layout_alignTop	Chỉ định đỉnh của thành phần này sẽ được canh theo đỉnh của thành phần gọi đến bằng ID
android:layout_alignBottom	Chỉ định đáy của thành phần này sẽ được canh theo đáy của thành phần gọi đến bằng ID.
android:layout_alignStart	Chỉ định cạnh start của thành phần này sẽ được canh theo cạnh start của thành phần gọi đến bằng ID.
android:layout_alignEnd	Chỉ định cạnh end của thành phần này sẽ được canh theo cạnh end của thành phần gọi đến bằng ID.
android:layout_alignBaseline	Chỉ định baseline của thành phần này sẽ được canh theo baseline của thành phần gọi đến bằng ID (thường dung cho TextView)
android:layout_above	Chỉ định thành phần này sẽ nằm ở trên so với thành phần gọi đến bằng ID.
android:layout_below	Chỉ định thành phần này sẽ nằm dưới so với thành phần gọi đến bằng ID.
android:layout_toStartOf	Chỉ định thành phần này sẽ nằm bên phía start so với thành phần gọi đến bằng ID.
android:layout_toEndOf	Chỉ định thành phần này sẽ nằm bên phía end so với thành phần gọi đến bằng ID.
android:layout_toLeftOf	Chỉ định thành phần này sẽ nằm bên phía trái so với thành phần gọi đến bằng ID.
android:layout_toRightOf	Chỉ định thành phần này sẽ nằm bên phía phải so với thành phần gọi đến bằng ID.

1.3.3 RelativeLayout - Định vị các view con với nhau

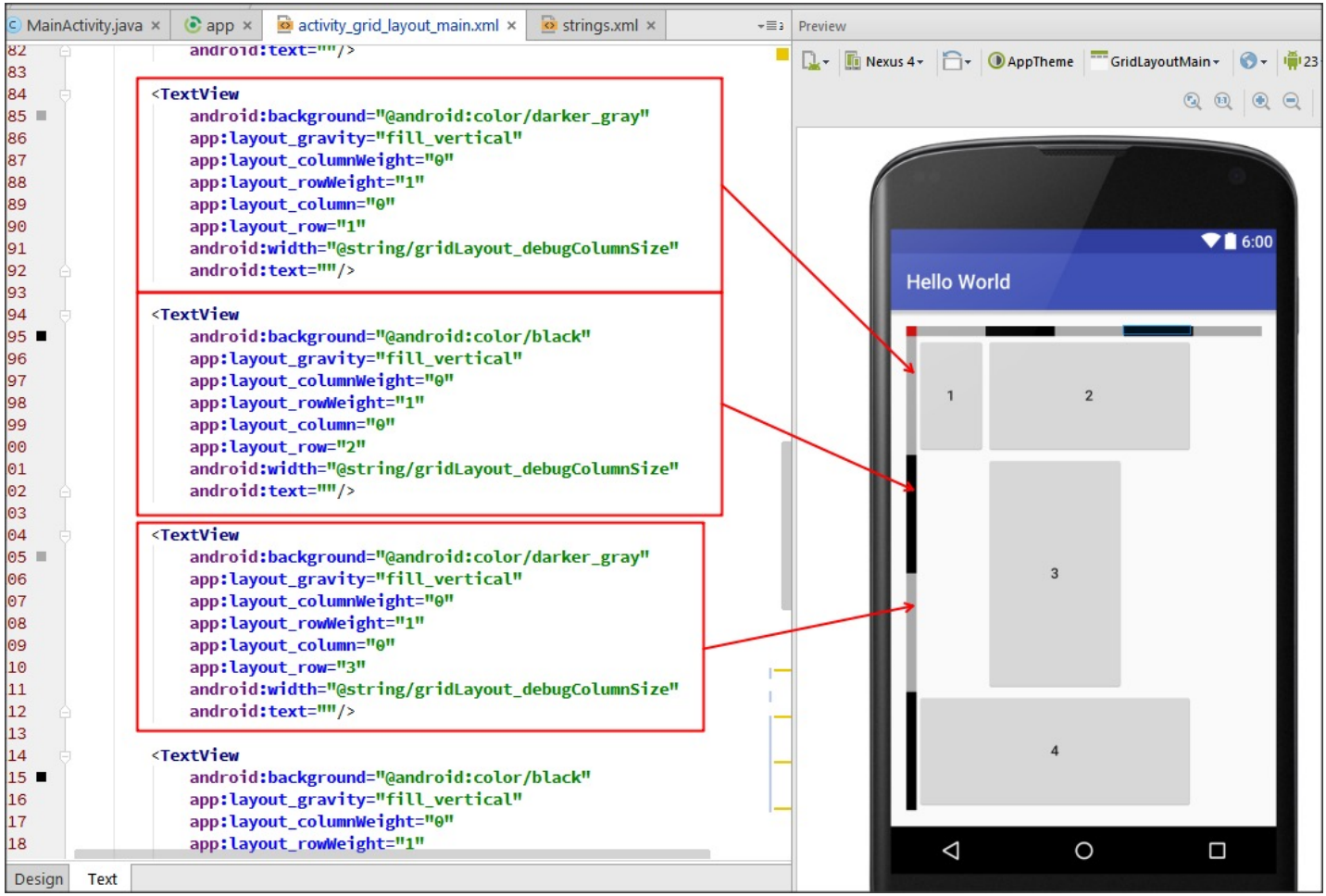




1.3.4 GridLayout

- GridLayout sử dụng một mạng lưới để tách khu vực của nó thành: các hàng, các cột, và các ô (cell). Nó hỗ trợ cả việc bắc qua (span) các hàng và các cột để chứa một View.

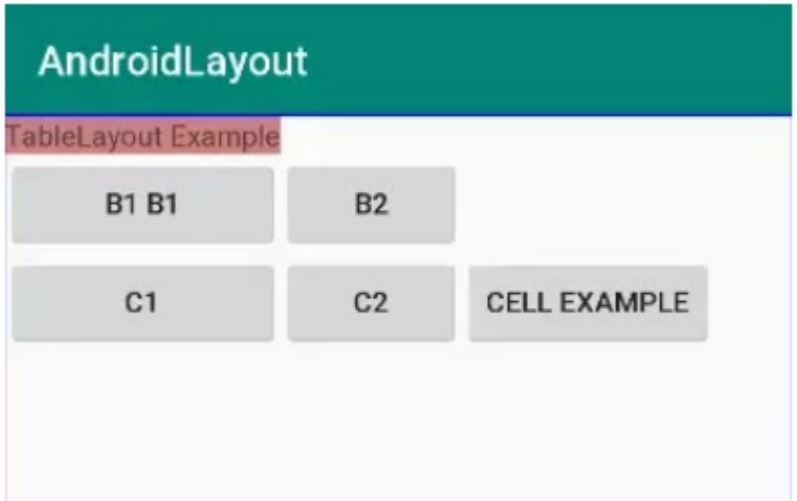
```
android:columnCount="6"
android:rowCount="5"
```



1.3.5 TableLayout

- Sắp xếp các View con bên trong thành dạng bảng. Mỗi hàng là một đối tượng view **TableRow** bên trong **TableRow** chứa các View con, mỗi View con này nằm ở vị trí một ô bảng (cell)

```
<TableLayout android:layout_height="match_parent"
    android:layout_width="match_parent"
    xmlns:android="http://schemas.android.com/apk/res/android">
    <TableRow>
        <TextView
            android:text="TableLayout Example"
            android:background="#c98282"
            android:gravity="center"/>
    </TableRow>

    <TableRow>
        <Button android:text="B1 B1" />
        <Button android:text="B2"/>
    </TableRow>

    <TableRow>
        <Button android:text="C1" />
        <Button android:text="C2" />
        <Button android:text="Cell example" />
    </TableRow>
</TableLayout>
```



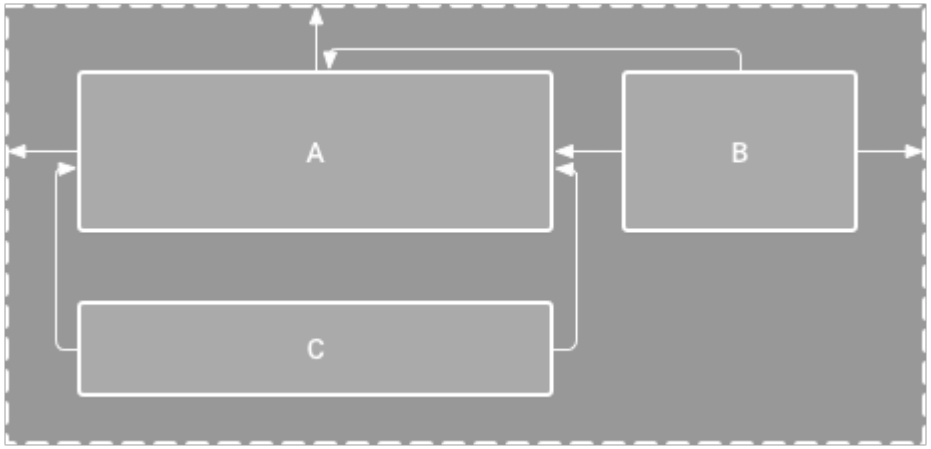
1.3.6 ConstraintLayout

- Giúp cho việc thiết kế các layouts phức tạp trở nên đơn giản hơn bằng cách cho phép các views kết nối với nhau thông qua các ràng buộc dựa trên mối quan hệ giữa các views khác nhau.
- Hướng tới việc thiết kếgiảm các views lồng nhau ,điều này sẽ làm tăng hiệu suất thực thi cho các tập tin layout

```
implementation 'com.android.support:appcompat-{{version}}’
implementation 'com.android.support.constraint:constraint-layout:{{version}}'
```

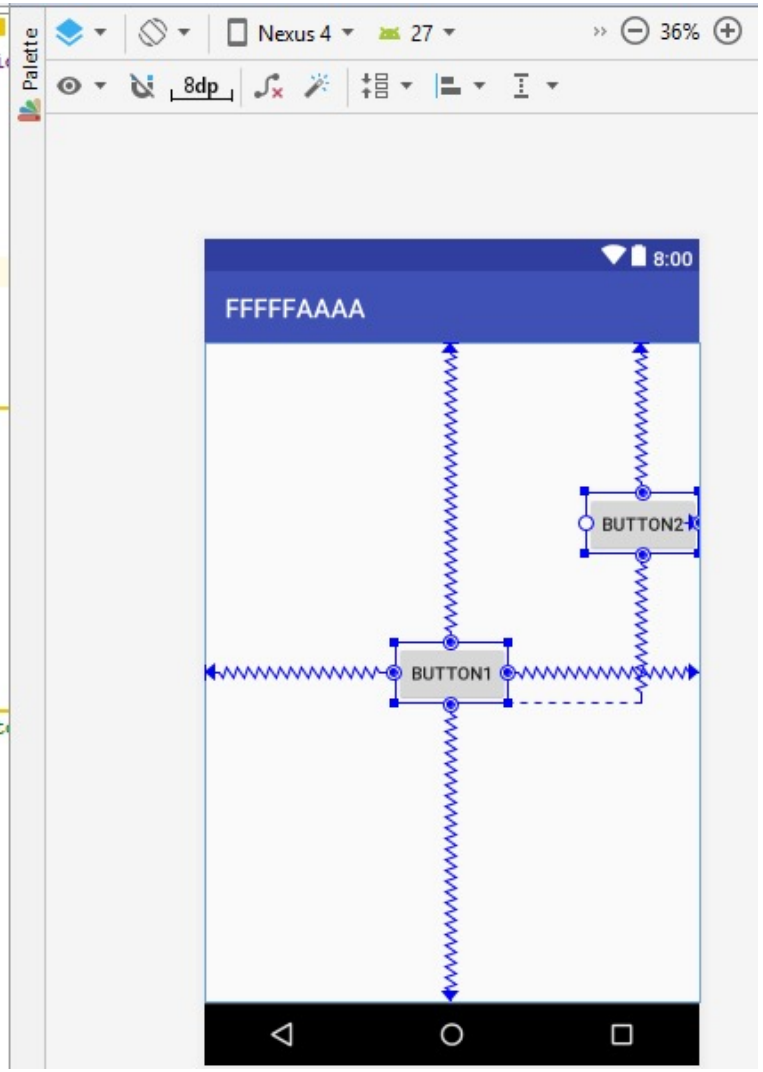
1.3.6 ConstraintLayout

- ConstraintLayout cung cấp các thuộc tính cho phép chúng ta định vị view hiện tại một cách dễ dàng. Các thuộc tính được mô tả như sau: ***layout_constraint<PointFrom>_to<PointTo>Of***
- *Các giá trị <Point> có thể là Top, Bottom, Left, Right*
- ***layout_constraintHorizontal_bias:*** Định vị view theo trục ngang màn hình.
- ***layout_constraintVertical_bias:*** Định vị view theo trục dọc màn hình.



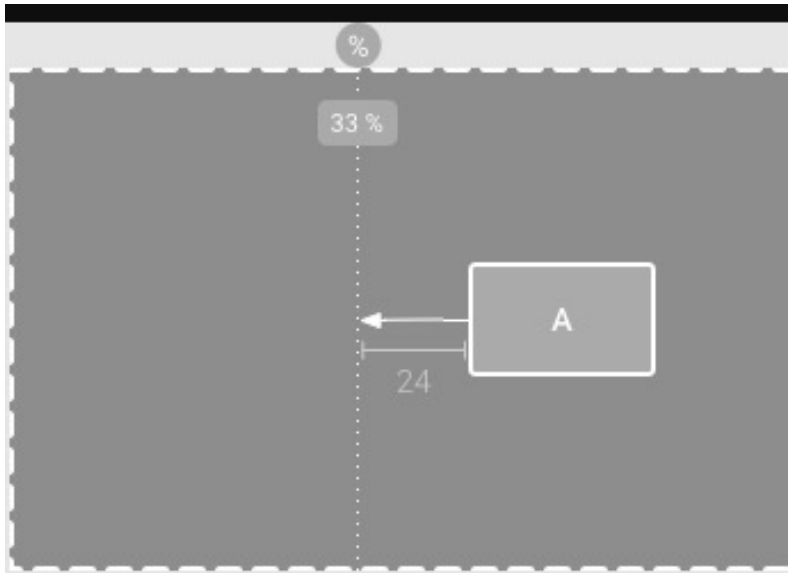
1.3.6 ConstraintLayout

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:androi
    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"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button1"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="BUTTON2"
        app:layout_constraintBottom_toBottomOf="@+id/butt
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```



1.3.6 ConstraintLayout - Guideline

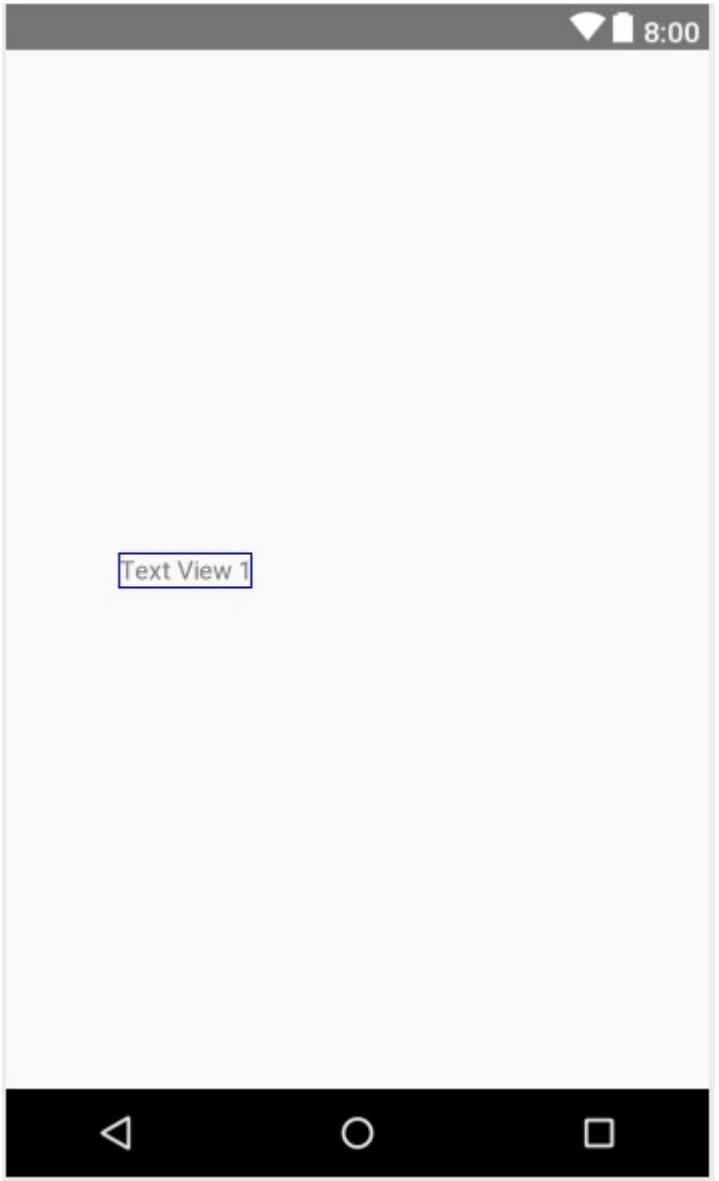
- Là một đường kẻ ẩn trong ConstraintLayout nằm ngang hoặc đứng nó như là một View con để các View khác ràng buộc đến nếu muốn
- Thiết lập đó là được kẻ ngang bằng thuộc tính: android:orientation="horizontal" đường kể đứng android:orientation="vertical“
- Vị trị của Guideline có thể thiết lập nó cách cạnh trái (hoặc trên nếu là Guideline ngang) bằng thuộc tính app:layout_constraintGuide_percent



1.3.6 ConstraintLayout - Bias

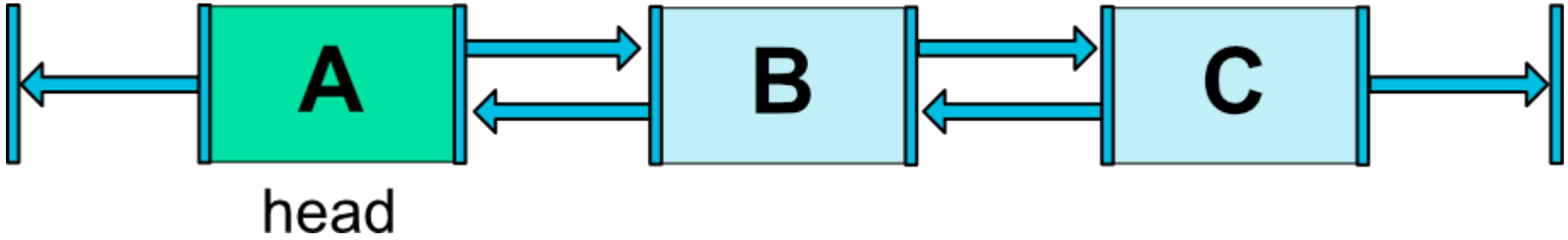
- Khi muốn sắp xếp View này thiên về bên nào hơn. Thuộc tính này chỉ có tác dụng khi View đang neo 2 cạnh đối diện hoặc cả 4 cạnh. ConstraintLayout cung cấp thuộc tính này cho cả chiều ngang (horizontal) và chiều dọc (vertical).

app:layout_constraintHorizontal_bias="0.2"



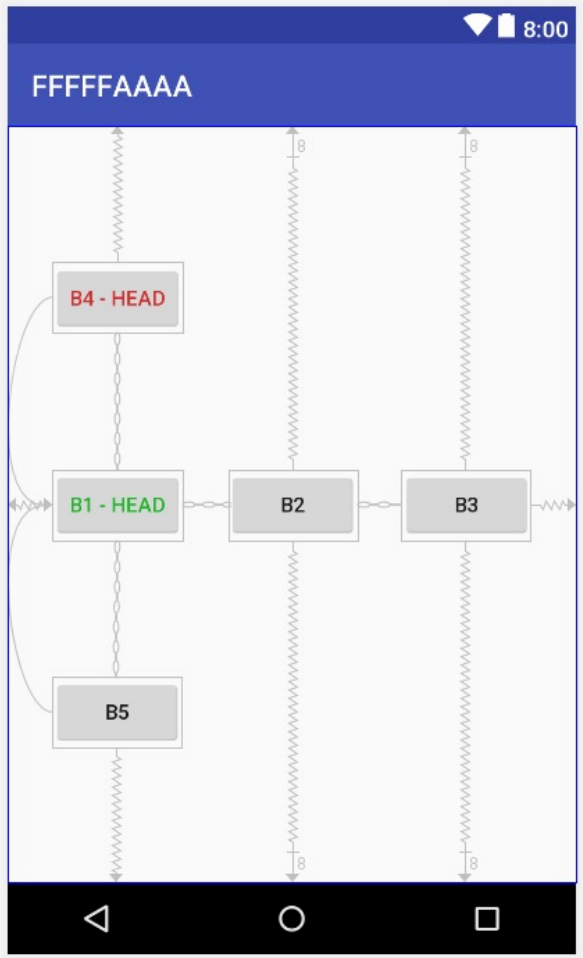
1.3.6 ConstraintLayout – Chain (xích)

- Các View ràng buộc qua lại các cạnh tiếp giáp nhau sẽ tạo thành một xích các View. Có hai loại xích các phần tử theo phương ngang và theo phương đứng. Lúc đó, phần tử đầu tiên có chức năng thiết lập chung một số thông số về xích.

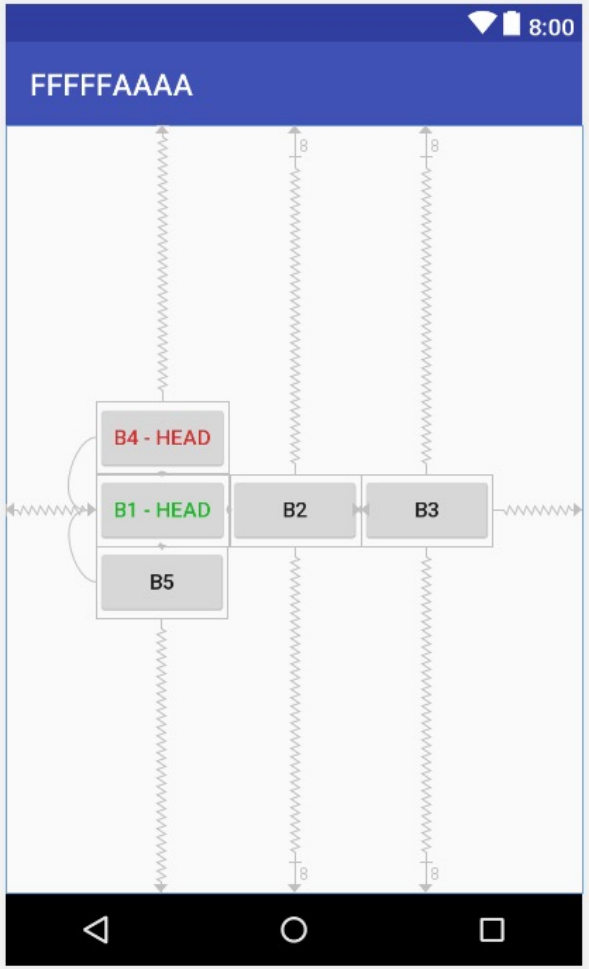


Phần tử đầu của xích thiết lập kiểu xích bằng thuộc tính: app:layout_constraintHorizontal_chainStyle và app:layout_constraintVertical_chainStyle tùy theo xích đứng hay ngang, mặc định xích có kiểu spread

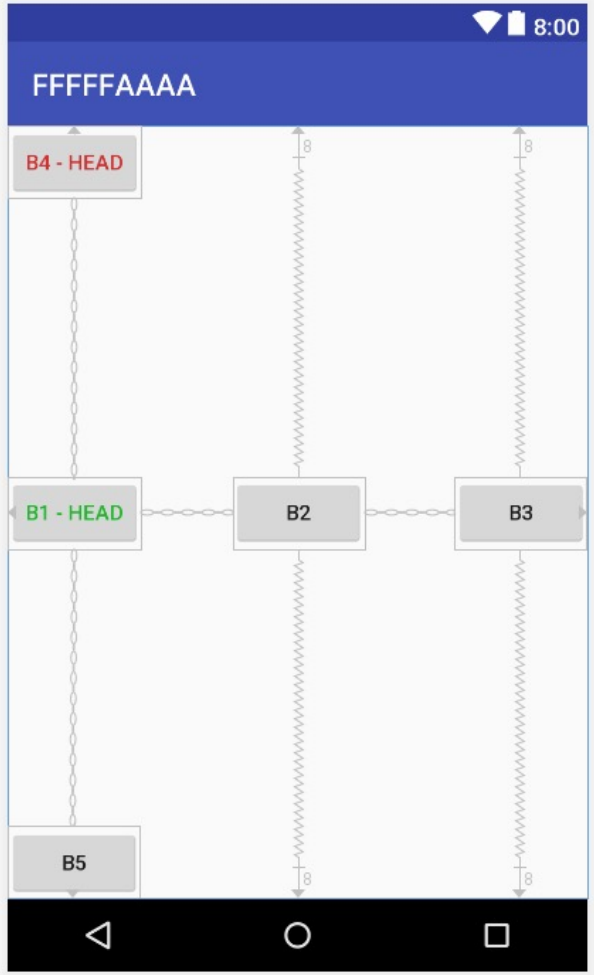
1.3.6 ConstraintLayout – Chain (xích)



xích “spread”



xích "packed”



xích “spread_inside”

LẬP TRÌNH ANDROID CĂN BẢN

Kết thúc ☺️

Ths. Trần Xuân Thanh Phúc | Trường Đại học Công Nghiệp Thực Phẩm