



FPT POLYTECHNIC



www.poly.edu.vn

LẬP TRÌNH ANDROID 1

CÁC LAYOUT CƠ BẢN TRONG ANDROID

❑ Các widget cơ bản trong Android

- ❖ LinearLayout, RelativeLayout
- ❖ ConstraintLayout



LẬP TRÌNH ANDROID 1

BÀI 3.1: CÁC LAYOUT CƠ BẢN TRONG ANDROID (P1)

MỤC TIÊU

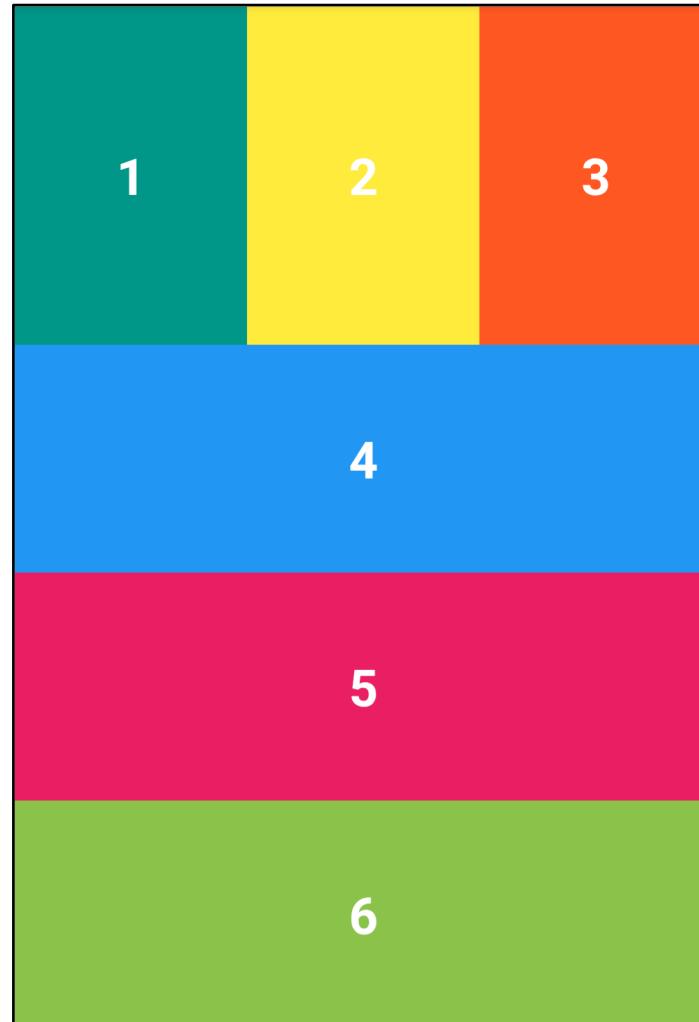
- NẮM ĐƯỢC CÁCH SỬ DỤNG CÁC LAYOUT CƠ BẢN TRONG ANDROID
- LINEARLAYOUT, RELATIVELAYOUT
- CONSTRAINTLAYOUT





LINEARLAYOUT

- ❑ **LinearLayout** là một layout rất tiện lợi trong thiết kế giao diện, nó sắp xếp các View con một cách liên tục theo tùy chọn xếp theo chiều ngang (một hàng các view) hay chiều đứng (một cột các view)



Nhập tên bạn

Bạn đang học hệ nào

Cao Đẳng Đại Học

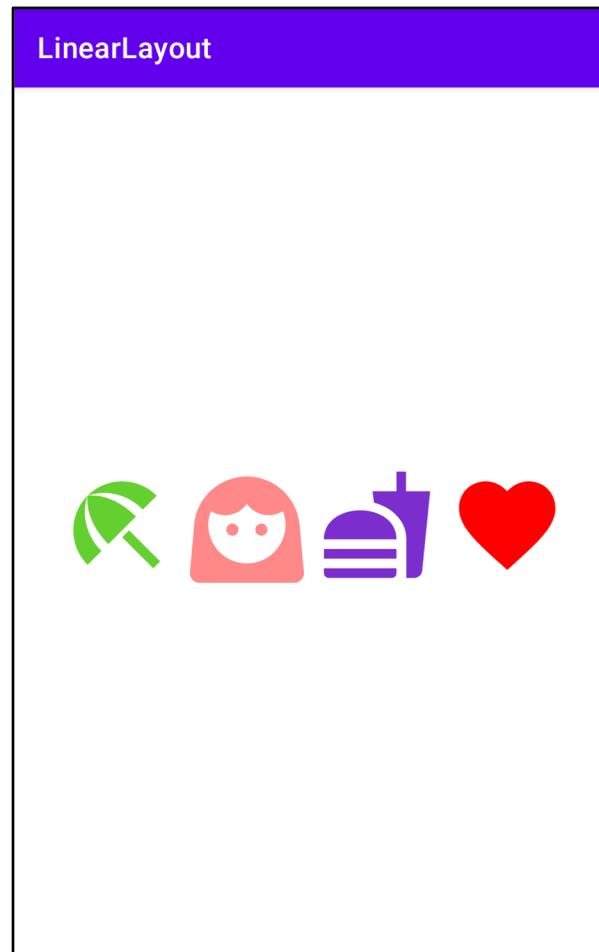
Bạn thích học ngôn ngữ nào

Java C++ JavaScript

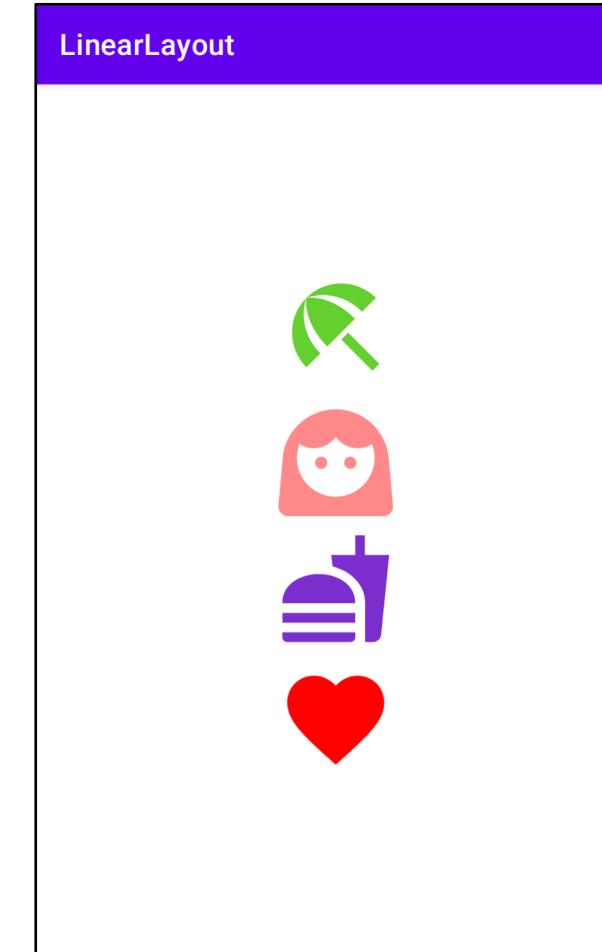
NHẤN VÀO ĐÂY ĐỂ NHẬP DỮ LIỆU

MỘT SỐ THUỘC TÍNH TRONG LINEARLAYOUT

- **android:orientation** dùng để thiết lập cách sắp xếp phần tử.



android:orientation="horizontal"
xếp theo chiều ngang



android:orientation="vertical"
xếp theo chiều dọc

MỘT SỐ THUỘC TÍNH TRONG LINEARLAYOUT

- ☐ **android:gravity** dùng để căn chỉnh các View nằm ở vị trí nào trong LinearLayout, nó nhận các giá trị bên dưới (có thể tổ hợp lại với ký hiệu |)

Giá trị	Ý nghĩa
center	Căn ở giữa
top	Căn ở phần trên
bottom	Căn ở phần dưới
center_horizontal	Căn ở giữa theo chiều ngang
center_vertical	Căn ở giữa theo chiều đứng
start (left)	Căn theo cạnh trái
end (right)	Căn theo cạnh phải

MỘT SỐ THUỘC TÍNH TRONG LINEARLAYOUT

❖ Ví dụ:

android:gravity="center"



android:gravity="end|center"



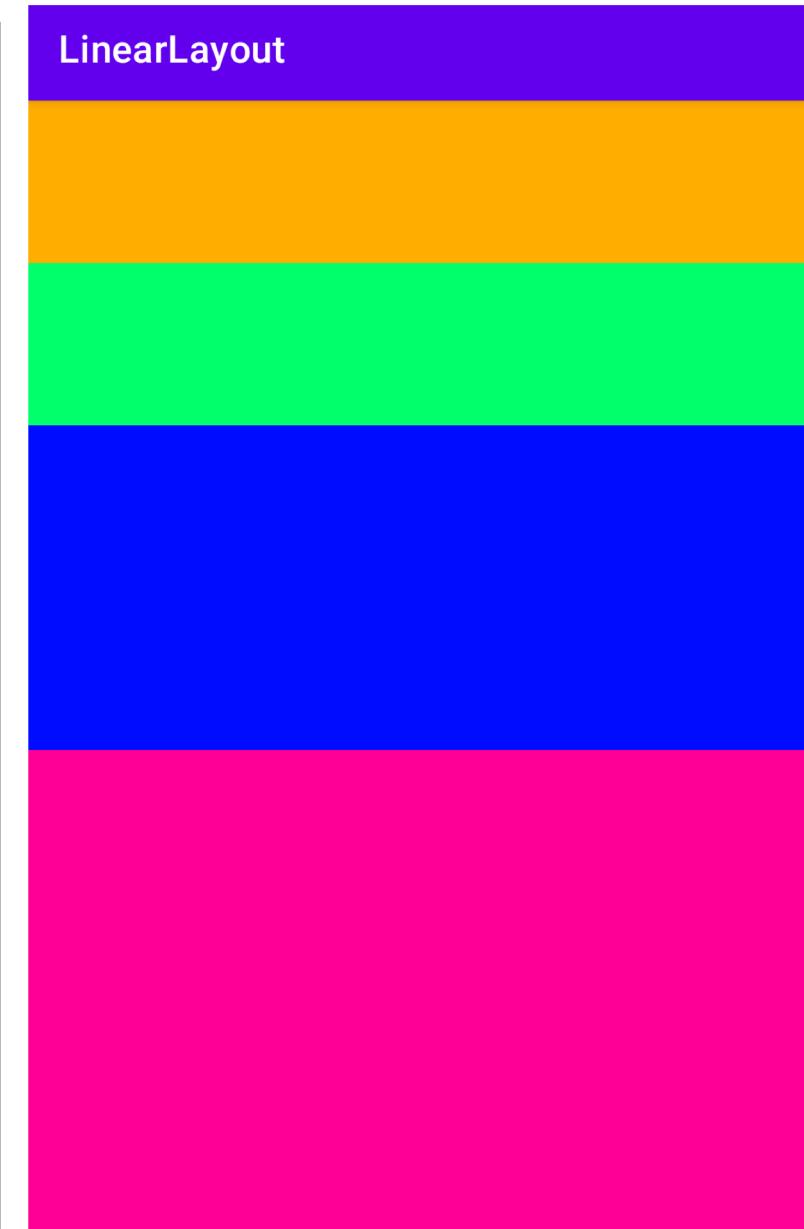
❑ android:layout_weight

- ❖ Ngoài cách sử dụng LinearLayout thông thường thì LinearLayout thường được sử dụng để phân chia tỉ lệ layout bằng cách sử dụng thuộc tính **layout_weight** và **weightSum**.
- ❖ **weightSum**: Xác định trọng số của LinearLayout hiện tại.
- ❖ **layout_weight**: Trọng số mà view con trong LinearLayout chiếm giữ.

MỘT SỐ THUỘC TÍNH TRONG LINEARLAYOUT

Chia tỉ lệ không
sử dụng
weightSum

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical">  
  
    <View  
        android:layout_width="match_parent"  
        android:layout_height="0dp"  
        android:layout_weight="1"  
        android:background="#00FF6A" />  
  
    <View  
        android:layout_width="match_parent"  
        android:layout_height="0dp"  
        android:layout_weight="2"  
        android:background="#000cff" />  
  
    <View  
        android:layout_width="match_parent"  
        android:layout_height="0dp"  
        android:layout_weight="3"  
        android:background="#FF0095" />  
    </LinearLayout>
```

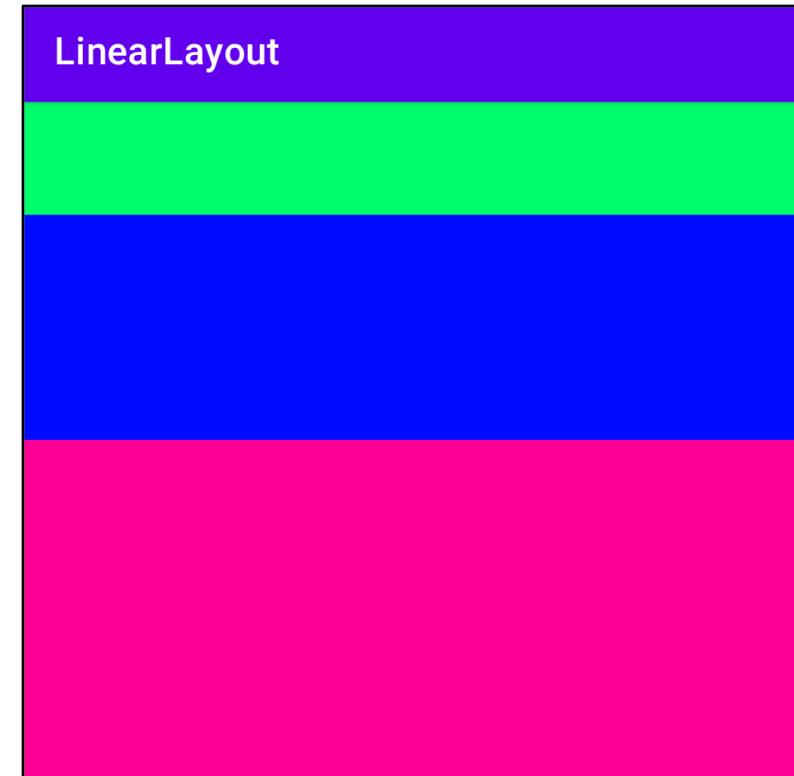


- ❑ Chúng ta không đánh trọng số **weightSum** cho LinearLayout nên tổng trọng số của nó sẽ được tính bằng tổng trọng số (**layout_weight**) các view con cộng lại.
- ❑ Ở đây **weightSum = 1 + 2 + 3 = 6**. Có nghĩa là LinearLayout này sẽ chia thành 6 phần:
 - ❑ View thứ nhất có **android:layout_weight="1"** sẽ chiếm **1/6**
 - ❑ View thứ hai có **android:layout_weight="2"** sẽ chiếm **2/6**
 - ❑ View thứ ba có **android:layout_weight="3"** sẽ chiếm **3/6**

MỘT SỐ THUỘC TÍNH TRONG LINEARLAYOUT

Chia tỉ lệ sử dụng
weightSum

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:weightSum="10"  
    android:orientation="vertical">  
  
<View  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:background="#00FF6A" />  
  
<View  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="2"  
    android:background="#000cff" />  
  
<View  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="3"  
    android:background="#FF0095" />  
</LinearLayout>
```



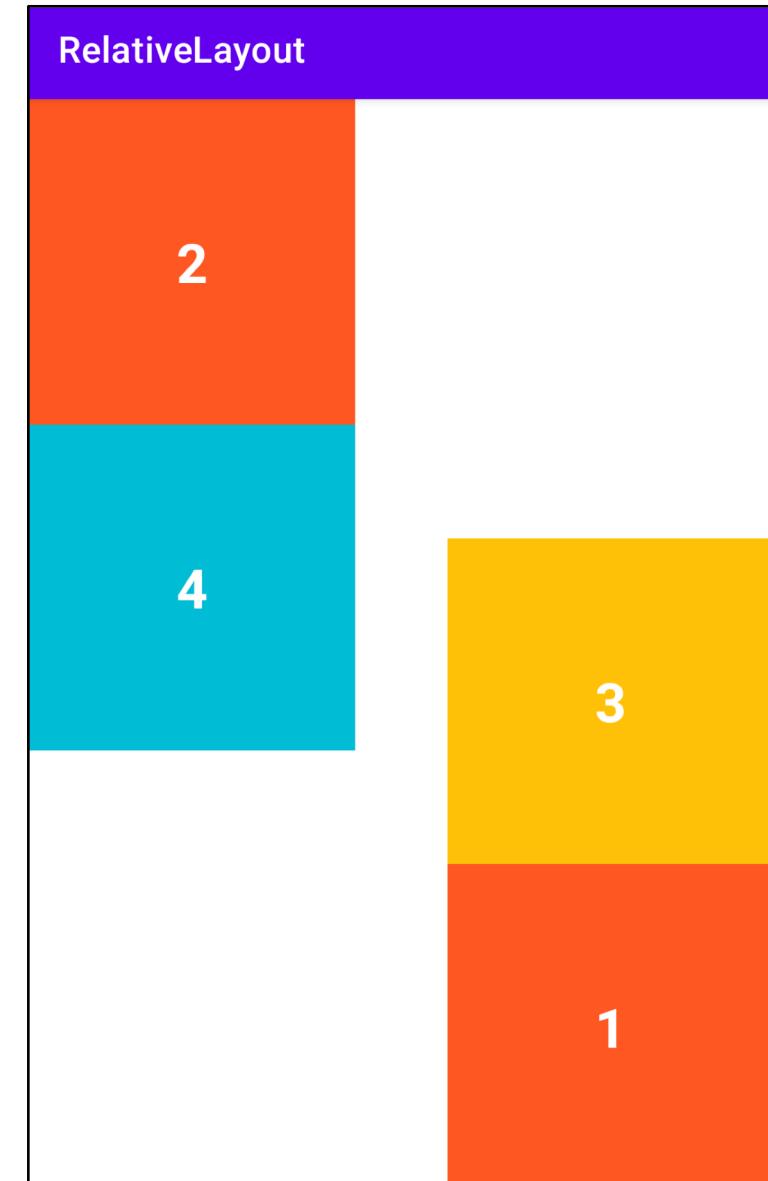
MỘT SỐ THUỘC TÍNH TRONG LINEARLAYOUT

- ❑ Khác với cách trên **weightSum** được tính toán bằng tổng của cách **layout_weight** của các view con. Thì chúng ta có thể xác định giá trị **weightSum** cụ thể
- ❑ Trong ví dụ trên, LinearLayout ta có thuộc tính **android:weightSum="10"**
- ❑ Chúng ta xác định **weightSum** có giá trị là **10**, có nghĩa là chia layout này thành 10 phần
 - ❖ View thứ nhất chiếm **1/10**
 - ❖ View thứ hai chiếm **2/10**
 - ❖ View thứ ba chiếm **3/10**
 - ❖ Còn lại **4/10** sẽ là khoảng trắng.



RELATIVELAYOUT

- ❑ **RelativeLayout** cho phép sắp xếp các widget theo vị trí tương đối giữa các widget khác trên giao diện (kể cả widget chứa nó). Thường nó dựa vào Id của các widget khác để sắp xếp theo vị trí tương đối



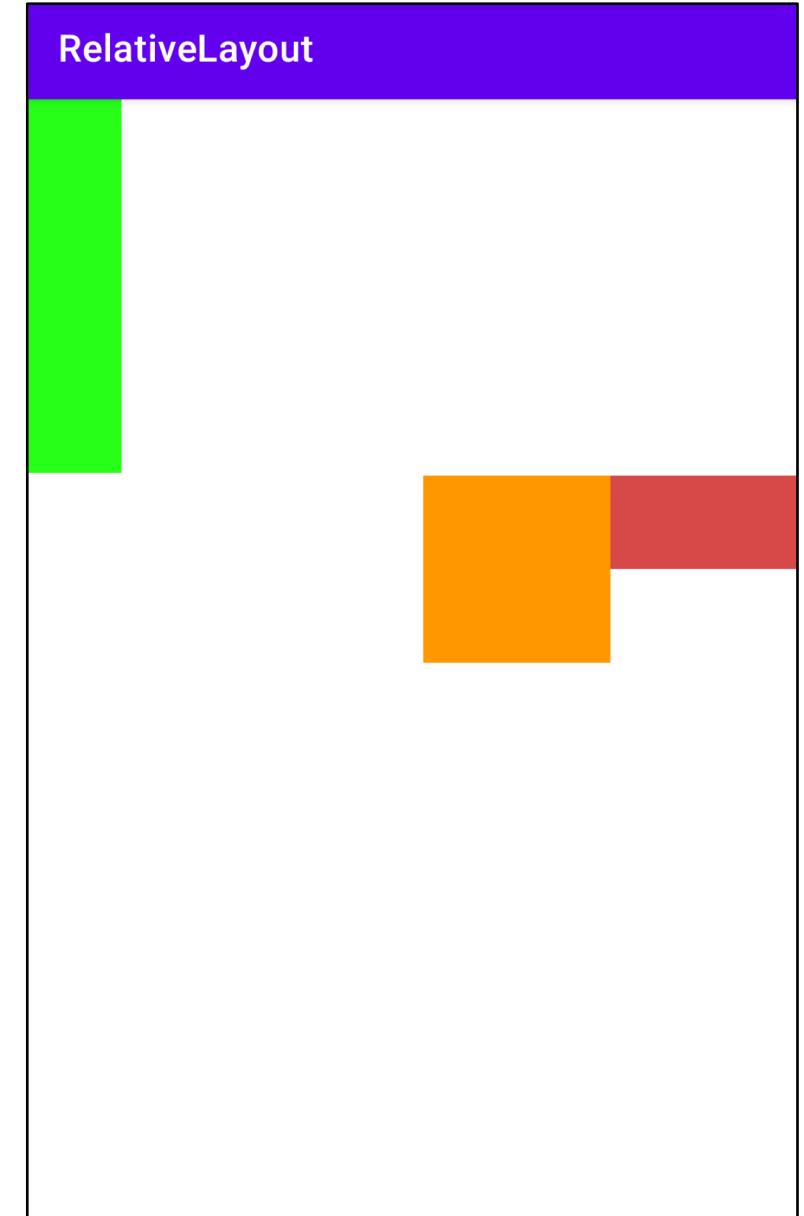
MỘT SỐ THUỘC TÍNH TRONG RELATIVELAYOUT

- **android:gravity** - dùng để thiết lập cách sắp xếp phần tử.

Giá trị	Ý nghĩa
center	Căn ở giữa
top	Căn ở phần trên
bottom	Căn ở phần dưới
center_horizontal	Căn ở giữa theo chiều ngang
center_vertical	Căn ở giữa theo chiều đứng
start (left)	Căn theo cạnh trái
end (right)	Căn theo cạnh phải

MỘT SỐ THUỘC TÍNH TRONG RELATIVELAYOUT

- ☐ **android:ignoreGravity** - RelativeLayout hỗ trợ một View con có thể không bị ảnh hưởng bởi gravity bằng thuộc tính **android:ignoreGravity="id-view-con"**



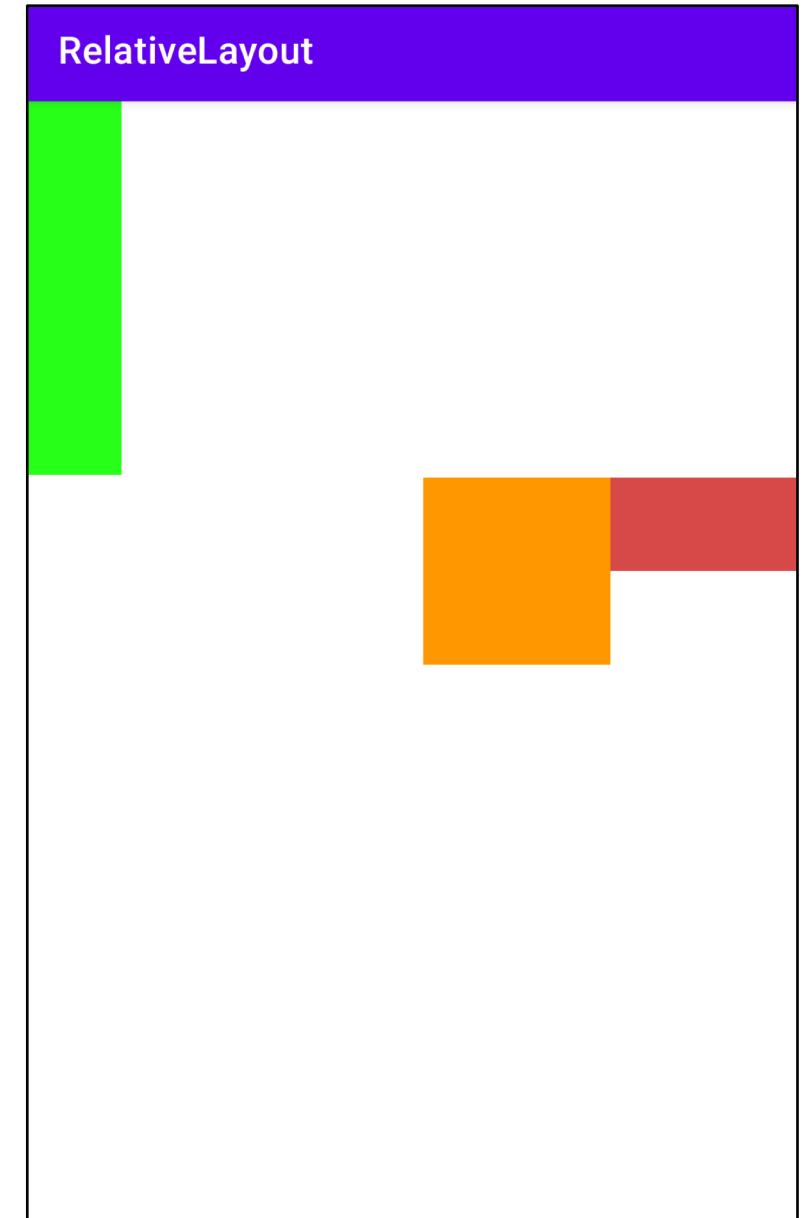
MỘT SỐ THUỘC TÍNH TRONG RELATIVELAYOUT

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center|end"
    android:ignoreGravity="@+id/view2">

    <View
        android:id="@+id/view1"
        android:layout_width="200dp"
        android:layout_height="50dp"
        android:background="#e8d33636" />

    <View
        android:id="@+id/view2"
        android:layout_width="50dp"
        android:layout_height="200dp"
        android:background="#E711FF00" />

    <View
        android:id="@+id/view3"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:background="#FF9800"/>
</RelativeLayout>
```



☐ Định vị View con bằng liên hệ với View cha RelativeLayout

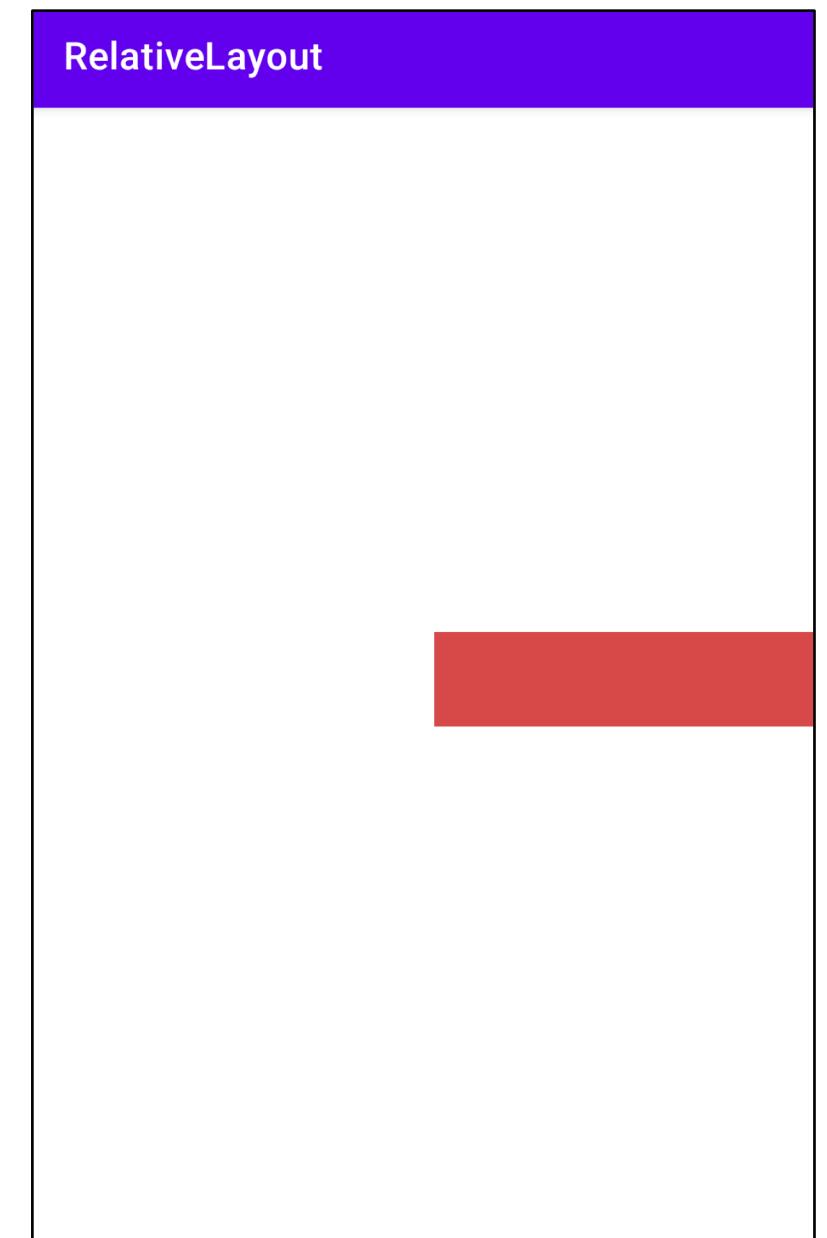
- ❖ Vị trí của View con trong RelativeLayout có thể thiết lập bằng cách chỉ ra mối liên hệ vị trí với view cha như căn thẳng cạnh trái View cha với View con, căn thẳng cạnh phải View cha với View con,...

Thuộc tính	Ý nghĩa
<code>android:layout_alignParentBottom</code>	căn thẳng cạnh dưới view con với cạnh dưới View cha
<code>android:layout_alignParentStart</code>	căn thẳng cạnh trái view con với cạnh trái View cha
<code>android:layout_alignParentEnd</code>	căn thẳng cạnh phải view con với cạnh phải View cha
<code>android:layout_alignParentTop</code>	căn thẳng cạnh trên view con với cạnh trên View cha
<code>android:layout_centerInParent</code>	căn view con vào giữa View cha
<code>android:layout_centerHorizontal</code>	căn view con vào giữa View cha theo chiều ngang
<code>android:layout_centerVertical</code>	căn view con vào giữa View cha theo chiều đứng

MỘT SỐ THUỘC TÍNH TRONG RELATIVELAYOUT

```
<RelativeLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
  
<View  
    android:id="@+id/view1"  
    android:layout_width="200dp"  
    android:layout_height="50dp"  
    android:layout_alignParentEnd="true"  
    android:layout_centerInParent="true"  
    android:background="#e8d33636" />  
  
</RelativeLayout>
```

RelativeLayout



☐ Định vị View con bằng liên hệ giữa chúng với nhau

- ❖ View con trong RelativeLayout ngoài mối liên hệ với View cha như trên, chúng có thể thiết lập liên hệ với nhau ví dụ như View con này nằm phía trên một View con khác, nằm phía dưới một view con khác,...

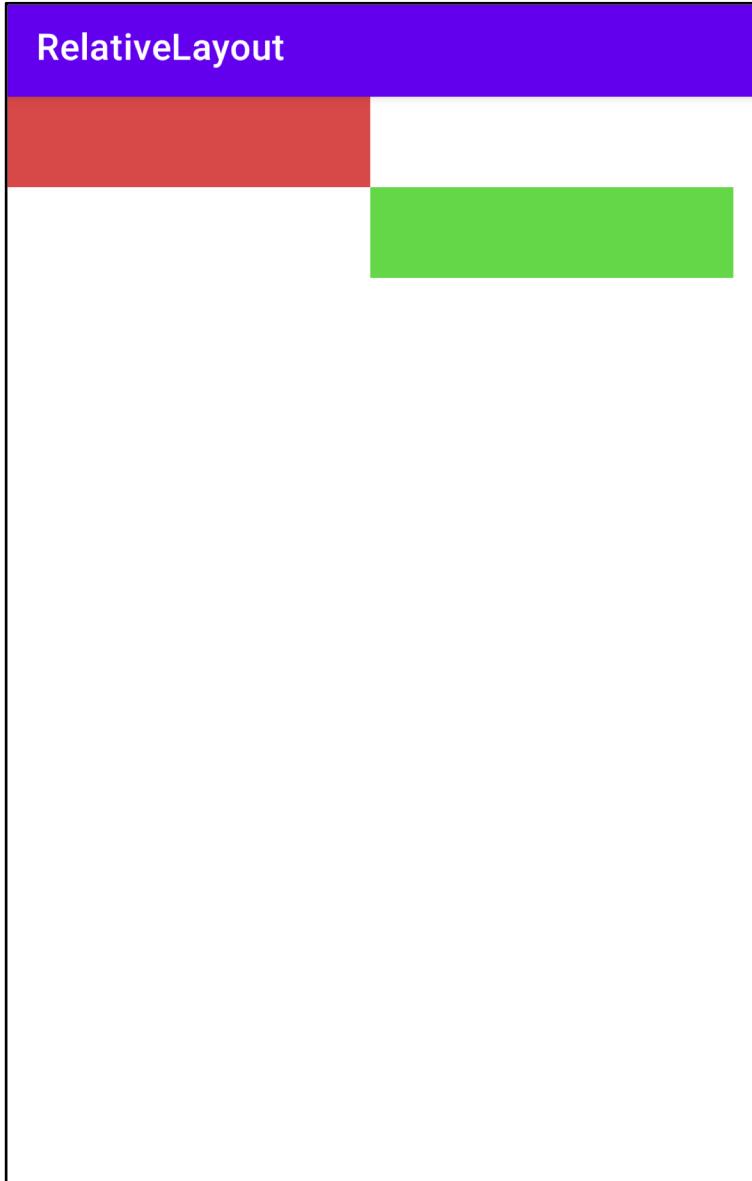
Thuộc tính	Ý nghĩa
<code>android:layout_below</code>	Nằm phía dưới View có ID được chỉ ra
<code>android:layout_above</code>	Nằm phía trên View có ID được chỉ ra
<code>android:layout_toStartOf</code>	Nằm phía trái View có ID được chỉ ra
<code>android:layout_toEndOf</code>	Nằm phía phải View có ID được chỉ ra
<code>android:layout_alignBottom</code>	Căn thẳng cạnh dưới với cạnh dưới của View có ID được chỉ ra
<code>android:layout_alignStart</code>	Căn thẳng cạnh trái với cạnh trái của View có ID được chỉ ra
<code>android:layout_alignEnd</code>	Căn thẳng cạnh phải với cạnh phải của View có ID được chỉ ra
<code>android:layout_alignTop</code>	Căn thẳng cạnh trên với cạnh trên của View có ID được chỉ ra

MỘT SỐ THUỘC TÍNH TRONG RELATIVELAYOUT

```
<RelativeLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">
```

```
<View  
    android:id="@+id/view1"  
    android:layout_width="200dp"  
    android:layout_height="50dp"  
    android:background="#e8d33636" />
```

```
<View  
    android:id="@+id/view2"  
    android:layout_width="200dp"  
    android:layout_height="50dp"  
    android:layout_below="@+id/view1"  
    android:layout_toEndOf="@+id/view1"  
    android:background="#E853D336" />  
</RelativeLayout>
```



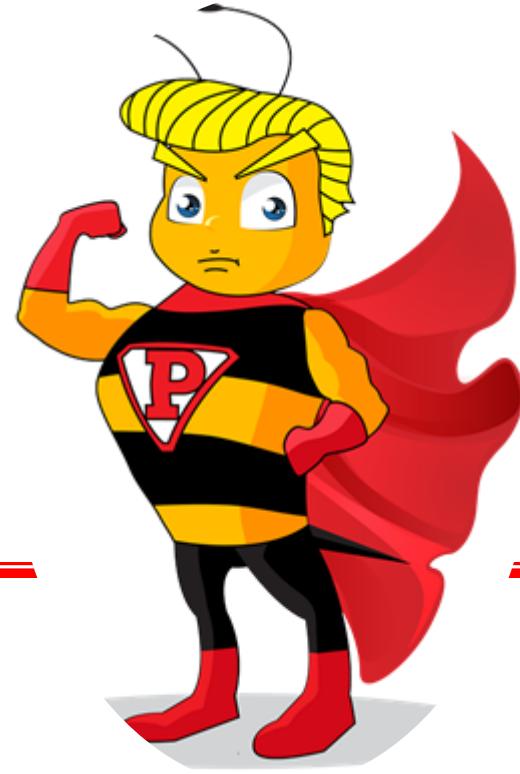


FPT POLYTECHNIC



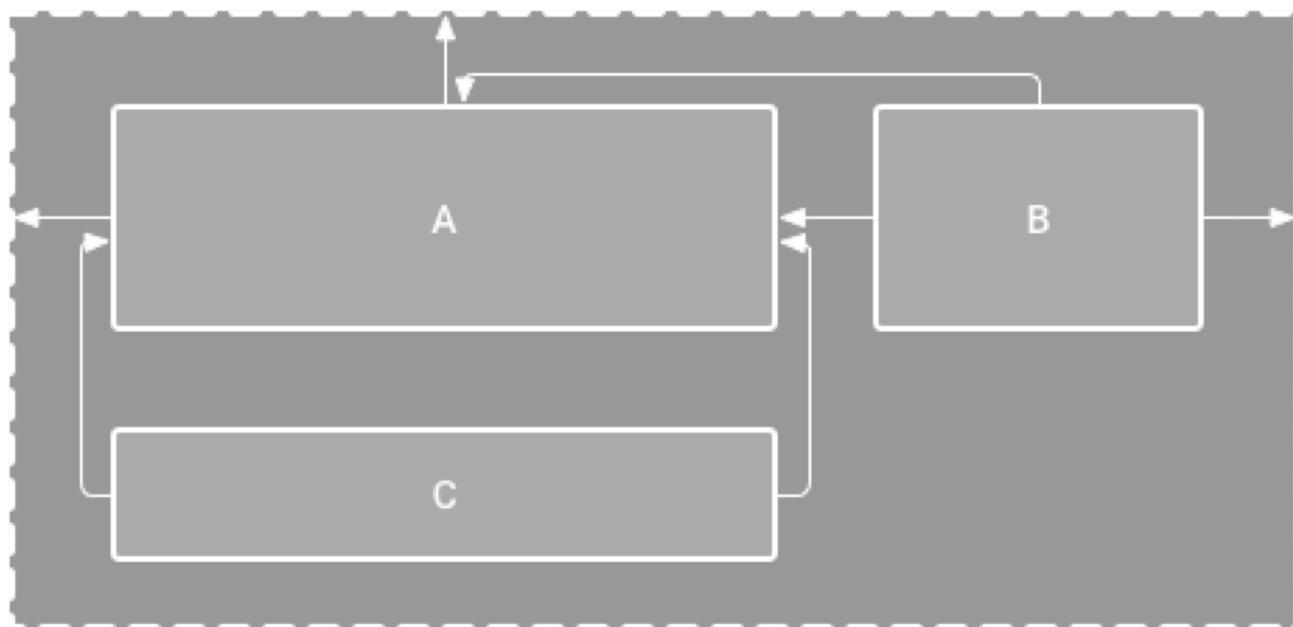
www.poly.edu.vn

BÀI 3.2: CÁC LAYOUT CƠ BẢN TRONG ANDROID (P2)

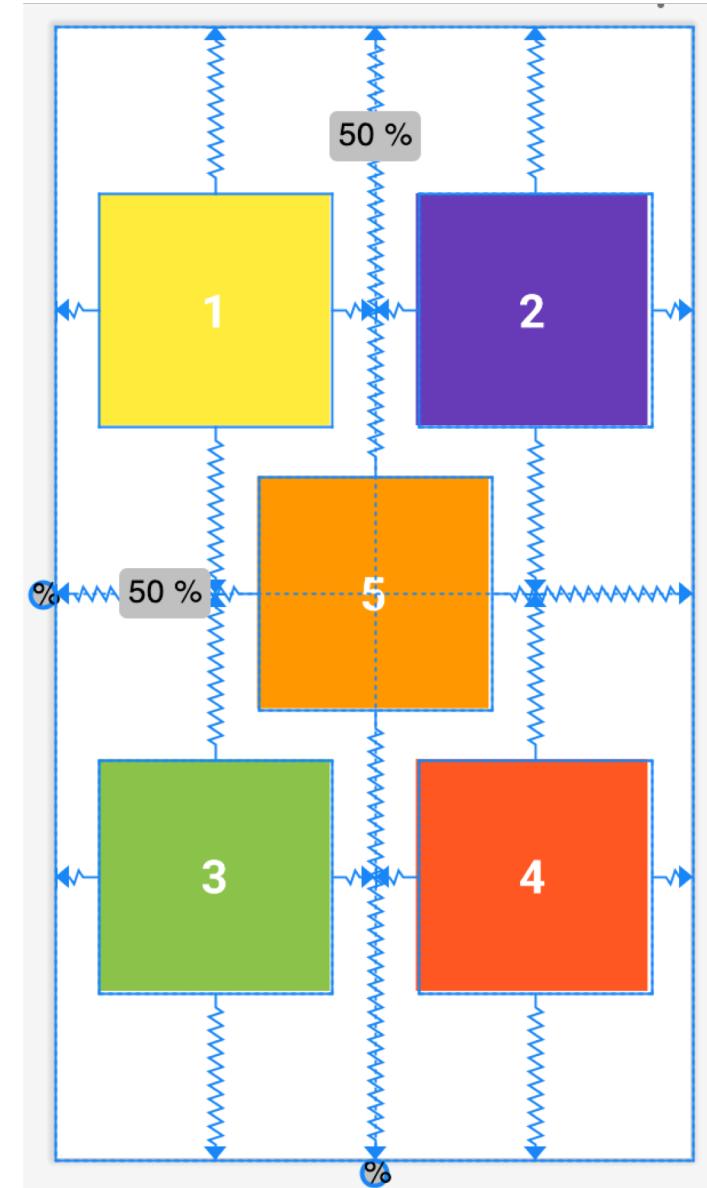


CONSTRAINTLAYOUT

- ❑ ConstraintLayout là một layout mạnh, nó giúp tạo ra các giao diện phức tạp, mềm dẻo (hạn chế tối đa sử dụng các layout lồng nhau). Giúp định vị, sắp xếp các View con dựa trên sự ràng buộc liên hệ của các View con với View cha và sự liên hệ ràng buộc giữa các View con với nhau, với cơ chế tạo xích các View, gán trọng số hay sử dụng trợ giúp giao diện với Guideline.



- ❑ Mỗi view trong **ConstraintLayout** để định vị được chính xác cần **tối thiểu 2 ràng buộc**, một theo phương ngang (X) và một theo phương đứng (Y)
- ❑ Các View không có ràng buộc sẽ định vị ở góc trái - trên (tọa độ 0,0).



❑ Các thuộc tính về ràng buộc layout_constraint...

- ❖ Các thuộc tính ràng buộc sử dụng với **namespace:app**, giá trị gán vào là một ID của phần tử khác để kết nối ràng buộc hoặc là phần tử đó bằng hằng số "**parent**"
- ❖ Ví dụ:

app:layout_constraintBottom_toBottomOf='parent'

MỘT SỐ THUỘC TÍNH TRONG CONSTRAINT LAYOUT

Ràng buộc	Ý nghĩa ràng buộc
<code>layout_constraintLeft_toLeftOf</code>	Ràng buộc cạnh trái của phần tử tới phần tử chỉ ra trong giá trị (gán ID)
<code>layout_constraintLeft_toRightOf</code>	Bên trái với bên phải của phần tử chỉ ra
<code>layout_constraintRight_toLeftOf</code>	Bên phải với bên trái
<code>layout_constraintRight_toRightOf</code>	Phải với phải
<code>layout_constraintTop_toTopOf</code>	Cạnh trên với cạnh trên
<code>layout_constraintTop_toBottomOf</code>	Cạnh trên nối với cạnh dưới
<code>layout_constraintBottom_toTopOf</code>	Dưới với trên
<code>layout_constraintBottom_toBottomOf</code>	Dưới với dưới
<code>layout_constraintBaseline_toBaselineOf</code>	Trùng Baseline
<code>layout_constraintStart_toEndOf</code>	Bắt đầu - Kết húc
<code>layout_constraintStart_toStartOf</code>	Bắt đầu - Bắt đầu
<code>layout_constraintEnd_toStartOf</code>	Cuối với bắt đầu
<code>layout_constraintEnd_toEndOf</code>	Cuối với cuối

MỘT SỐ THUỘC TÍNH TRONG CONSTRAINTLAYOUT

The screenshot shows the Android Studio interface with the XML code for a ConstraintLayout and its corresponding layout preview.

XML Code:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    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">

    <View
        android:id="@+id/view1"
        android:layout_width="100dp"
        android:layout_height="50dp"
        android:background="#FFEB3B"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <View
        android:id="@+id/view2"
        android:layout_width="100dp"
        android:layout_height="50dp"
        android:background="#673AB7"
        app:layout_constraintBottom_toBottomOf="@+id/view1"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Layout Preview:

The layout preview shows a blue rectangular container representing the ConstraintLayout. Inside, there are two views: a yellow view at the bottom-left and a purple view at the top-right. Both views have constraints relative to each other and the parent layout. The yellow view is constrained to the bottom, end, start, and top of the parent. The purple view is constrained to the bottom of the yellow view, the end of the parent, and the top of the parent. Spring lines indicate the active constraints between the views and the parent layout.

□ Phản tử Guideline

- ❖ Ta có thể 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. Thêm vào bằng cách:

```
<android.support.constraint.Guideline  
    android:id="@+id/guideline_1"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:orientation="vertical"  
    app:layout_constraintGuide_percent="0.3" />
```

- ❖ Thiết lập **đường kẻ ngang** bằng thuộc tính: **android:orientation="horizontal"** và **đường kẻ dọc** **android:orientation="vertical"**

□ Vị trí phân chia ConstraintLayout của Guideline

- ❖ Vị trí của Guideline có thể thiết lập cách cạnh trái (hoặc cạnh trên nếu là Guideline ngang) bằng thuộc tính **app:layout_constraintGuide_percent**
- ❖ Sử dụng tỷ lệ phần trăm để làm giá trị của thuộc tính này: **0.5 (50%), 0.2 (20%)**
Ví dụ: **app:layout_constraintGuide_percent="0.1"**
- ❖ Giá trị của thuộc tính này được tính bằng tỉ lệ của chiều ngang (hoặc chiều dọc) của ConstraintLayout

MỘT SỐ THUỘC TÍNH TRONG CONSTRAINTLAYOUT

The screenshot shows the Android Studio interface with the XML code for a ConstraintLayout and its corresponding visual preview.

XML Code:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    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">

    <androidx.constraintlayout.widget.Guideline
        android:id="@+id/guideline"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        app:layout_constraintGuide_percent="0.3" />

    <View
        android:id="@+id/view1"
        android:layout_width="200dp"
        android:layout_height="50dp"
        android:background="#363EDF"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintStart_toStartOf="@+id/guideline"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Visual Preview:

The preview shows a blue rectangular view labeled "view1" with a width of 200dp and a height of 50dp, colored "#363EDF". It is positioned at the bottom center of the parent layout. A vertical guideline is placed 30% from the top edge. The view is constrained to the bottom, end, left, and start edges relative to the guideline.

❑ Bias

- ❖ Khi hai cạnh đối diện nhau của View có ràng buộc, thì hai ràng buộc này ứng xử như một liên kết lò xo mặc định nó có độ mềm (độ cứng) bằng nhau dẫn đến View sẽ nằm giữa 2 điểm neo của ràng buộc. Nếu muốn điều chỉnh độ cứng này thì sử dụng thuộc tính:
 - ❖ **app:layout_constraintVertical_bias** thiết lập độ mềm của ràng buộc đầu (ngang).

Với tổng độ mềm là 1 thì khi **app:layout_constraintVertical_bias="0.1"** thì độ mềm ràng buộc thứ hai sẽ là **0.9**

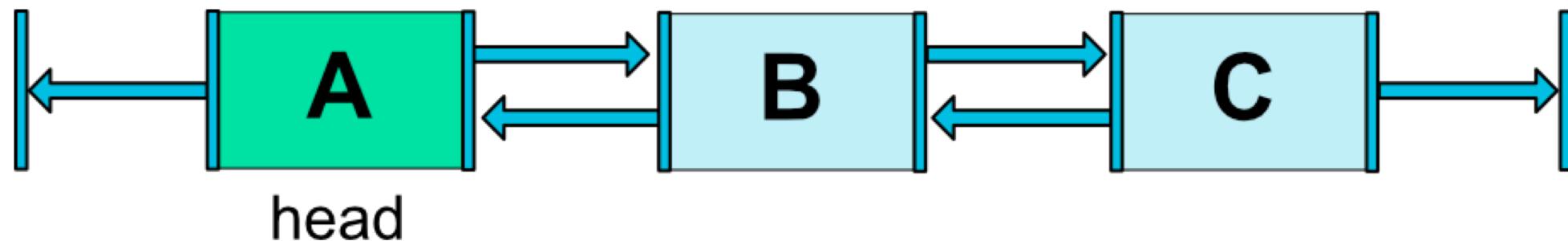
- ❖ **layout_constraintHorizontal_bias** để thiết lập độ mềm hai ràng buộc theo phương đứng

□ Tỷ lệ các cạnh của View

- ❖ Khi View con có thiết lập tối thiểu một kích thước là "**0dp**" thì kích thước đó có thể tự động điều chỉnh bằng cách lấy theo tỷ lệ với cạnh còn lại, thuộc tính **app:layout_constraintDimensionRatio** cho phép gán tỷ lệ giữa chiều rộng và chiều cao.
- ❖ Ví dụ: **app:layout_constraintDimensionRatio="2:1"** chiều rộng gấp đôi chiều cao

❑ Chain - xích các View lại

- ❖ 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



❑ Chain - xích các View lại

- ❖ 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**

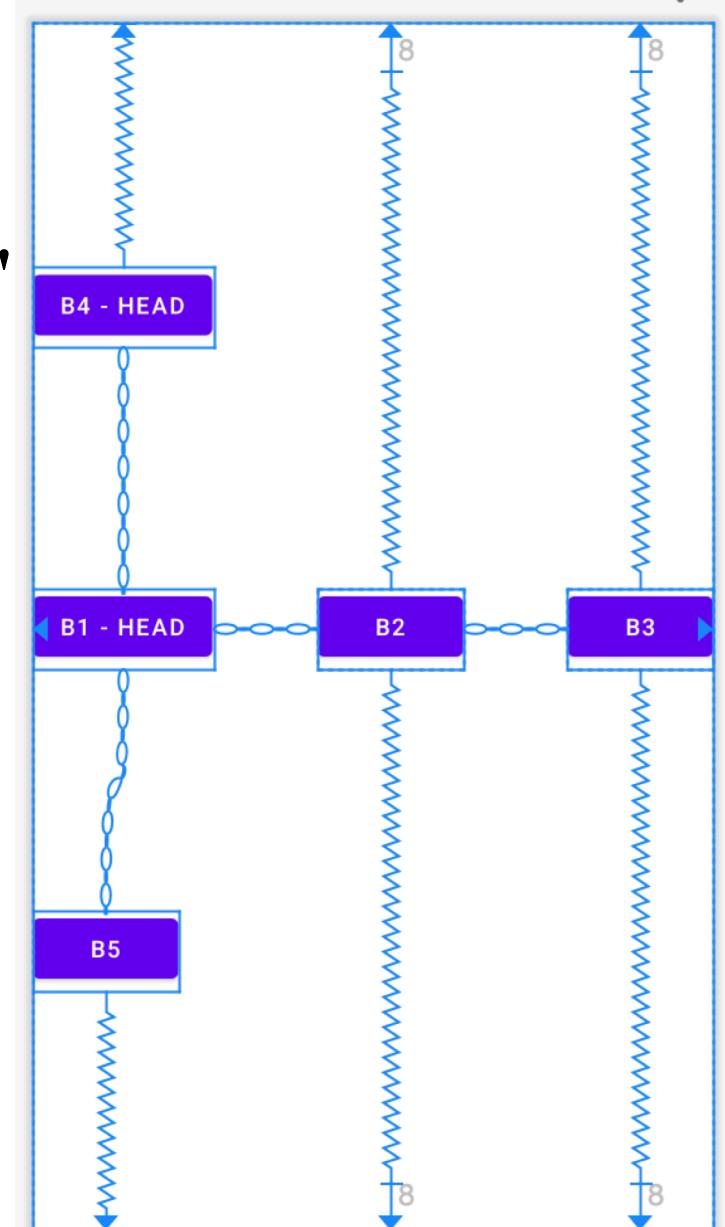
MỘT SỐ THUỘC TÍNH TRONG CONSTRAINT LAYOUT

❑ Chain - xích "spread":

`app:layout_constraintHorizontal_chainStyle="spread"`

hoặc

`app:layout_constraintVertical_chainStyle="spread"`



MỘT SỐ THUỘC TÍNH TRONG CONSTRAINT LAYOUT

```
<androidx.constraintlayout.widget.ConstraintLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
  
    <Button  
        android:id="@+id/b1"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:text="B1 - HEAD"  
        app:layout_constraintBottom_toBottomOf="parent"  
        app:layout_constraintHorizontal_chainStyle="spread"  
        app:layout_constraintRight_toLeftOf="@id/b2"  
        app:layout_constraintStart_toStartOf="parent"  
        app:layout_constraintTop_toTopOf="parent" />
```

1

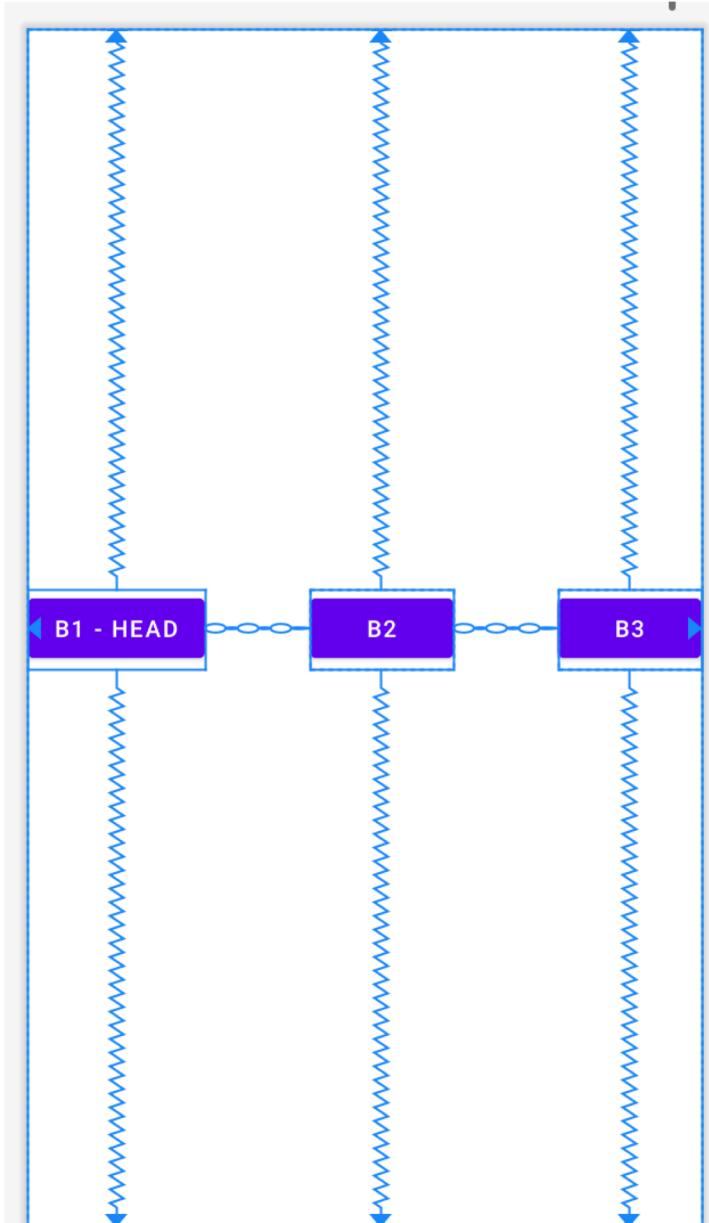
```
<Button  
    android:id="@+id/b2"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="B2"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintLeft_toRightOf="@id/b1"  
    app:layout_constraintRight_toLeftOf="@id/b3"  
    app:layout_constraintTop_toTopOf="parent" />  
  
<Button  
    android:id="@+id/b3"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="B3"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintLeft_toRightOf="@id/b2"  
    app:layout_constraintTop_toTopOf="parent" />
```

2

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

MỘT SỐ THUỘC TÍNH TRONG CONSTRAINT LAYOUT

app:layout_constraintHorizontal_chainStyle="spread"



MỘT SỐ THUỘC TÍNH TRONG CONSTRAINT LAYOUT

```
<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
        xmlns:app="http://schemas.android.com/apk/res-auto"  
        android:layout_width="match_parent"  
        android:layout_height="match_parent">  
  
<Button  
    android:id="@+id/b4"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="B4\nHEAD"  
    app:layout_constraintBottom_toTopOf="@id/b1"  
    app:layout_constraintLeft_toLeftOf="@id/b1"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_chainStyle="spread" />
```

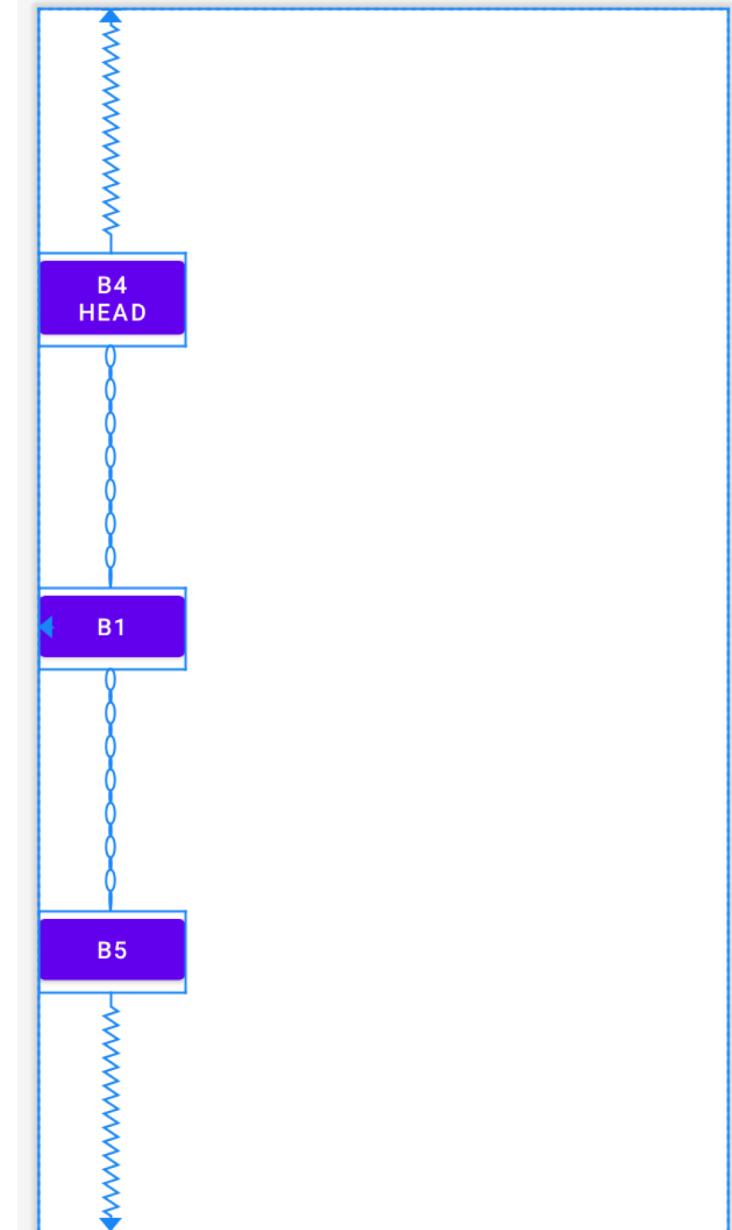
1

```
<Button  
    android:id="@+id/b1"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="B1"  
    app:layout_constraintBottom_toTopOf="@id/b5"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toBottomOf="@id/b4" />  
  
<Button  
    android:id="@+id/b5"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="B5"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintLeft_toLeftOf="@id/b1"  
    app:layout_constraintTop_toBottomOf="@id/b1" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>
```

2

MỘT SỐ THUỘC TÍNH TRONG CONSTRAINT LAYOUT

app:layout_constraintVertical_chainStyle="spread"

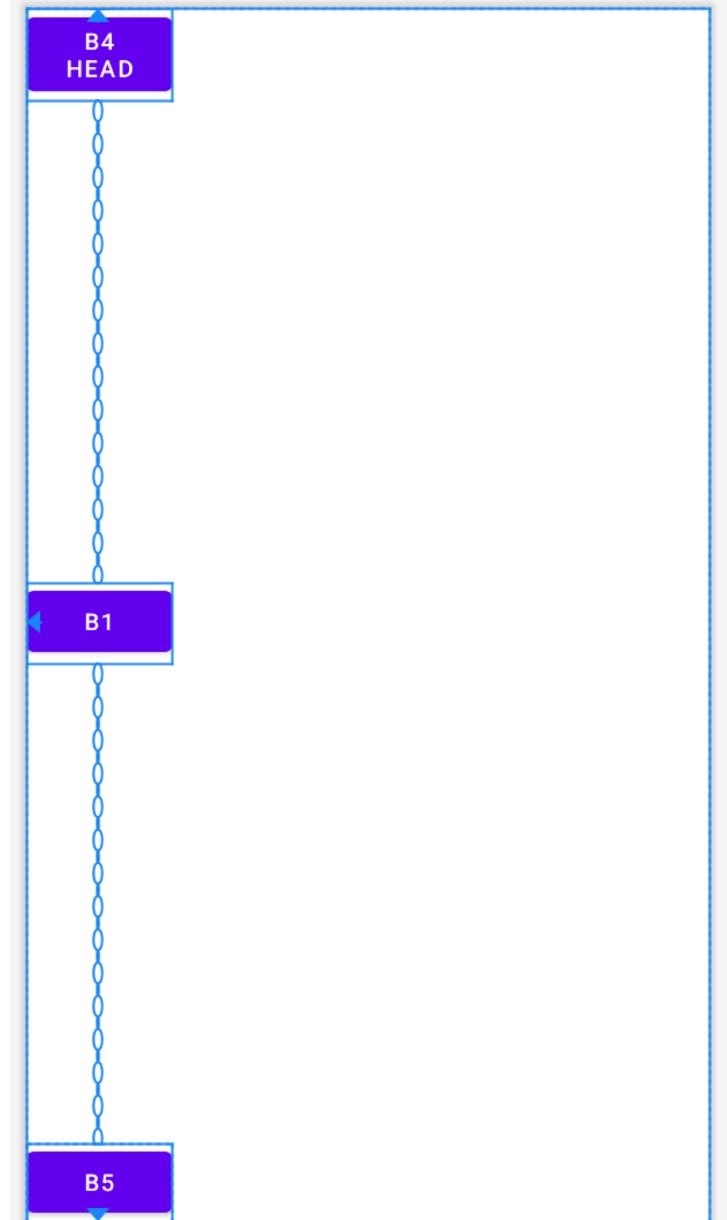


MỘT SỐ THUỘC TÍNH TRONG CONSTRAINT LAYOUT

❑ Chain - xích "spread_inside":

app:layout_constraintVertical_chainStyle="spread_inside"

tương tự với **layout_constraintHorizontal_chainStyle**

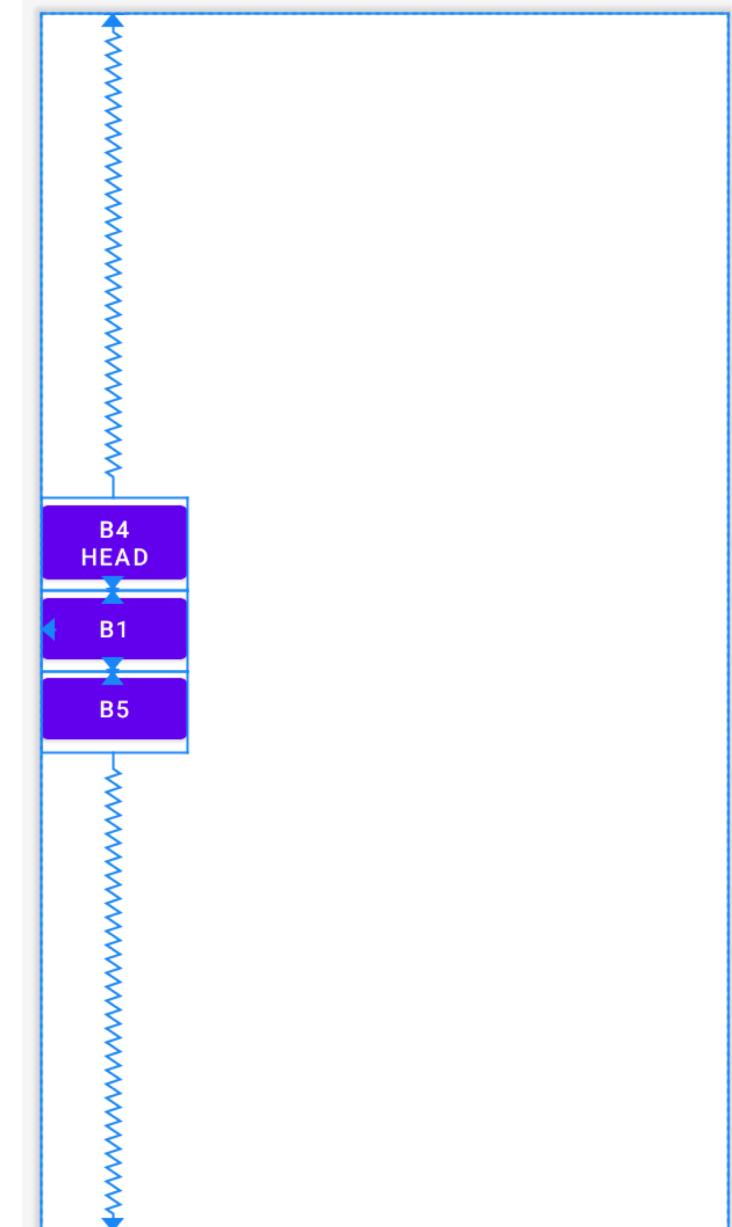


MỘT SỐ THUỘC TÍNH TRONG CONSTRAINT LAYOUT

❑ Chain - xích "packed":

app:layout_constraintVertical_chainStyle= "packed"

tương tự với **layout_constraintHorizontal_chainStyle**



- Với loại **spread_inside** và **spread** theo phương của xích các phần tử có thể gán kích thước bằng **0** và điều chỉnh lại theo trọng số weight với ý nghĩa tương tự như **LinearLayout**, thuộc tính thiết lập trọng số là:
 - ❖ **app:layout_constraintHorizontal_weight**
 - ❖ **app:layout_constraintVertical_weight**

MỘT SỐ THUỘC TÍNH TRONG CONSTRAINTLAYOUT

The screenshot shows the Android Studio interface with the XML code editor and the Layout Editor.

XML Code:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/b4"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:text="B4\nHEAD"
        app:layout_constraintBottom_toTopOf="@+id/b1"
        app:layout_constraintLeft_toLeftOf="@+id/b1"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_chainStyle="packed"
        app:layout_constraintVertical_weight="1" />

    <Button
        android:id="@+id/b1"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:text="B1"
        app:layout_constraintBottom_toTopOf="@+id/b5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/b4"
        app:layout_constraintVertical_weight="2" />

```

Component Tree:

- B4 HEAD (Purple)
- B1 (Blue)
- B5 (Yellow)

The components are arranged vertically. B4 is at the top, followed by B1, and then B5 at the bottom. B4 has its bottom constrained to the top of B1, and B1 has its top constrained to the bottom of B4. B1 also has its start constrained to the parent's start. B1 has a weight of 2, while B4 has a weight of 1, causing B1 to occupy more vertical space.



FPT Education

FPT POLYTECHNIC

Thank you