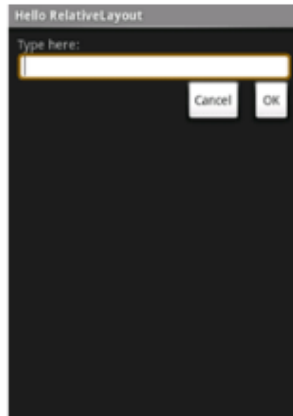


# Views & Layouts



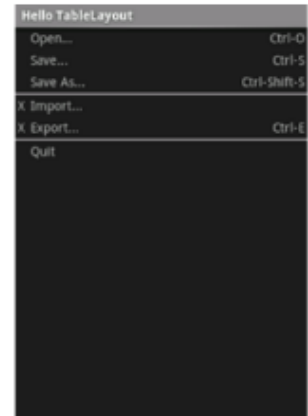
## Linear Layout

A LinearLayout is a ViewGroup that will lay child View elements vertically or horizontally.



## Relative Layout

A RelativeLayout is a ViewGroup that allows you to layout child elements in positions relative to the parent or siblings elements.



## Table Layout

A TableLayout is a ViewGroup that will lay child View elements into rows and columns.

## I. LinearLayout:

- Orientation
  - Horizontal (dạng hàng)
  - Vertical (dạng cột)

Có thể thiết lập thuộc tính: `android:orientation="..."` trong file layout xml hoặc có thể thay đổi `setOrientation()` bằng mã khi chương trình đang chạy

- Fill Model
  - Các view trong LinearLayout đều phải thiết lập thuộc tính kích thước `android:layout_width` và `android:layout_height`
  - Giá trị kích thước có thể là:
    - Một giá trị cụ thể - như 200 dip
    - **wrap\_content** – vừa đủ nội dung bên trong
    - **Fill\_parent** hoặc **match\_parent** – bằng độ lớn của đối tượng cha chứa chúng

- **Weight**
  - **android:layout\_weight** – trọng số để xác định tỉ lệ tương ứng phân không gian còn trống dành cho đối tượng (view)
  - Giá trị có thể là 1,2,3,.. Mặc định là 0
- **Gravity** được dùng để xác định cách căn lề của các đối tượng trên màn hình.
  - Mặc định thì các đối tượng sẽ căn lề phía trên, bên trái
  - Khi căn căn lề, ta dùng thuộc tính XML:
    - **android:layout\_gravity**="..."
    - **android:gravity**="..."

Giá trị có thể là: left, center, right, top, bottom, vv

### **Margin & Padding**

- **android:layout\_margin**=".." : Cách lề bên ngoài
- **android:padding**="..." : Cách lề bên trong

### **Padding & Margin**

- **android:padding**="..." : Cách lề bên trong view
- **android:layout\_margin**=".." : Cách lề bên ngoài view

Padding: Là khoảng không gian bên trong viền của LinearLayout

Margins: Thiết lập khoảng không gian giữa các View con của Viewgroup

## **II. RelativeLayout:**


Tên thuộc tính	Mô tả
<b>android:layout_above</b>	Đặt phần tử hiện tại nằm kế sau phần tử có id được chỉ ra
<b>android:layout_alignBaseline</b>	Đặt phần tử này lên cùng dòng với phần tử có id được chỉ ra
<b>android:layout_alignBottom</b>	Canh sao cho đáy của phần tử hiện thời trùng với đáy của phần tử có id được chỉ ra
<b>android:layout_alignLeft</b>	Đặt cạnh trái của phần tử hiện thời trùng với cạnh trái của phần tử có id được chỉ ra
<b>android:layout_alignParentBottom</b>	Nếu thiết lập là true thì phần tử hiện thời sẽ được canh xuống đáy của phần tử chứa nó
<b>android:layout_alignParentLeft</b>	Nếu được thiết lập là true thì phần tử hiện thời sẽ canh trái so với phần tử chứa nó

<b>android:layout_alignParentRight</b>	Nếu được thiết lập là true thì phần tử hiện thời sẽ canh phải so với phần tử chứa nó
<b>android:layout_alignParentTop</b>	Nếu được thiết lập là true thì phần tử hiện thời sẽ canh lên đỉnh phần tử chứa nó
<b>android:layout_alignRight</b>	Canh cạnh phải của phần tử hiện thời trùng với cạnh phải của phần tử có id được chỉ ra
<b>android:layout_alignTop</b>	Canh đỉnh của phần tử hiện thời trùng với đỉnh của phần tử có id được chỉ ra
<b>android:layout_alignWithParentIf Missing</b>	Nếu thiết lập là true, thì phần tử sẽ được canh theo phần tử chứa nó nếu các thuộc tính canh của phần tử không có.
<b>android:layout_below</b>	Đặt phần tử hiện thời ngay sau phần tử có id được chỉ ra.
<b>android:layout_centerHorizontal</b>	Nếu thiết lập là true thì phần tử hiện thời sẽ được canh giữa theo chiều ngang phần tử chứa nó.

<b>android:layout_centerInParent</b>	Nếu thiết lập là true thì phần tử hiện thời sẽ được canh chính giữa theo chiều phải trái và trên dưới so với phần tử chứa nó.
<b>android:layout_centerVertical</b>	Nếu thiết lập là true thì phần tử hiện thời sẽ được canh chính giữa theo chiều dọc phần tử chứa nó.
<b>android:layout_toLeftOf</b>	Đặt cạnh phải của phần tử hiện thời trùng với cạnh trái của phần tử có id được chỉ ra.
<b>android:layout_toRightOf</b>	Đặt cạnh trái của phần tử hiện thời trùng với cạnh phải của phần tử có id được chỉ ra.

### III. AbsoluteLayouts:

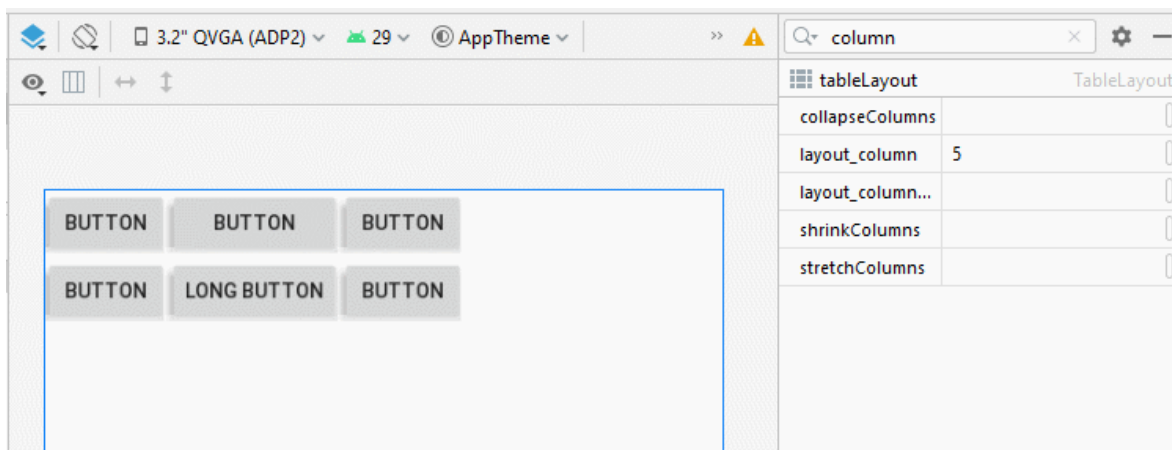
```
<AbsoluteLayout xmlns:android="
    http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Element One"
    />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Element Two"
        android:layout_x="30px"
        android:layout_y="30px"
    />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Element Three"
        android:layout_x="50px"
        android:layout_y="50px"
    />
</AbsoluteLayout>
```



Chỉ định vị trí các view con thông qua tọa độ (x,y)

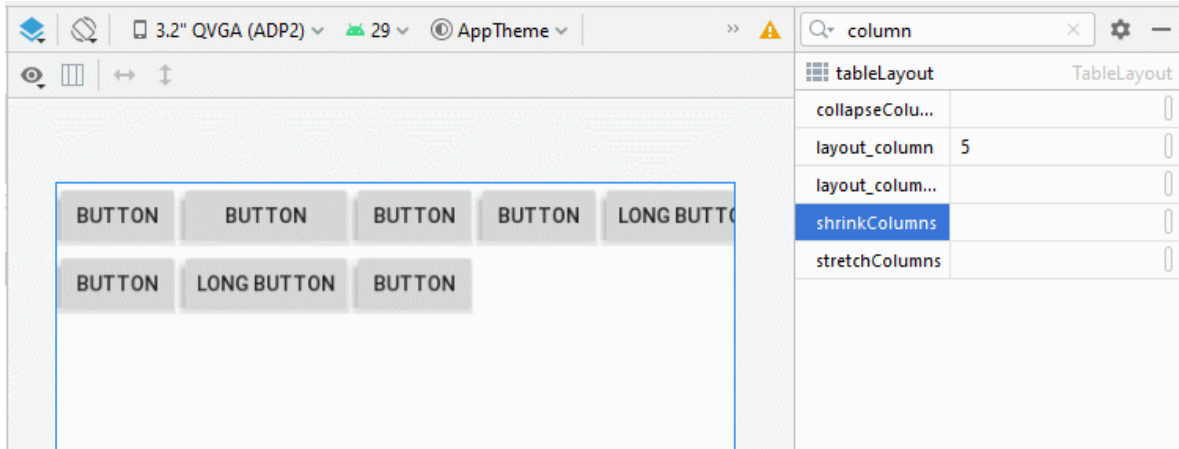
### IV. TableLayouts:

**android:stretchColumns** cho phép chỉ định các cột sẽ được kéo dài (stretched) để lấp đầy không gian còn trống



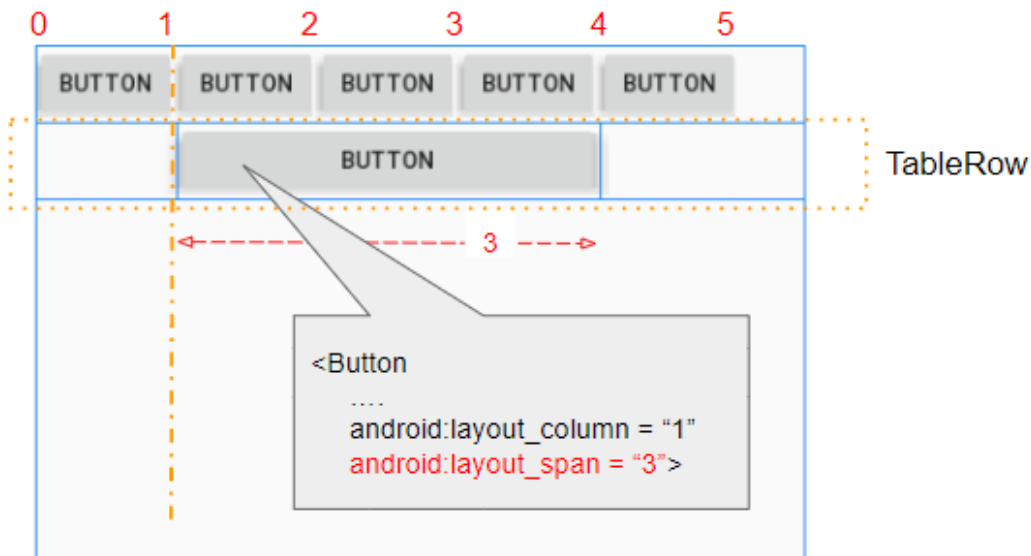
column	
collapseColumns	
layout_column	5
layout_column...	
shrinkColumns	
stretchColumns	

**android:shrinkColumns** chỉ định các cột sẽ bị co lại (shrunk) để tránh việc các View con tràn ra ngoài TableLayout



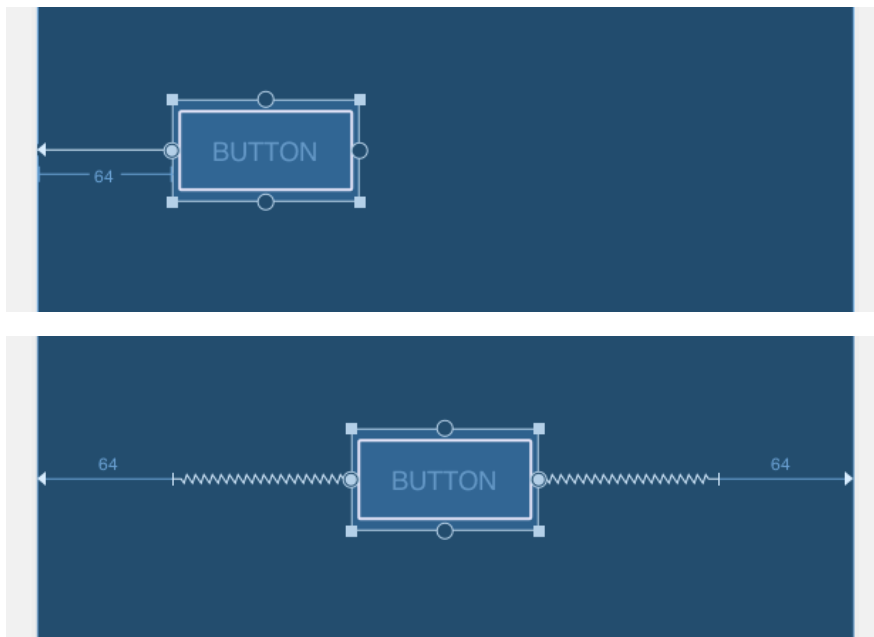
**android:layout\_span** áp dụng cho View con để chỉ định số ô liên tiếp trong một TableRow sẽ được hợp nhất với nhau.

**android:layout\_column** được áp dụng cho một View con trong một TableRow để chỉ định vị trí



## V. ConstraintLayouts:

- Tương tự `RelativeLayout` nhưng hỗ trợ kéo thả mạnh mẽ



View Margin – layout\_width:

- **Fixed:** chỉ định cứng
- **match\_constraint:** tương tự match\_parent trong RelativeLayout
- **wrap\_content**

