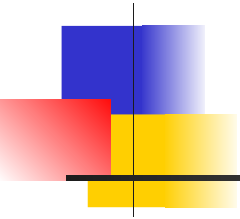


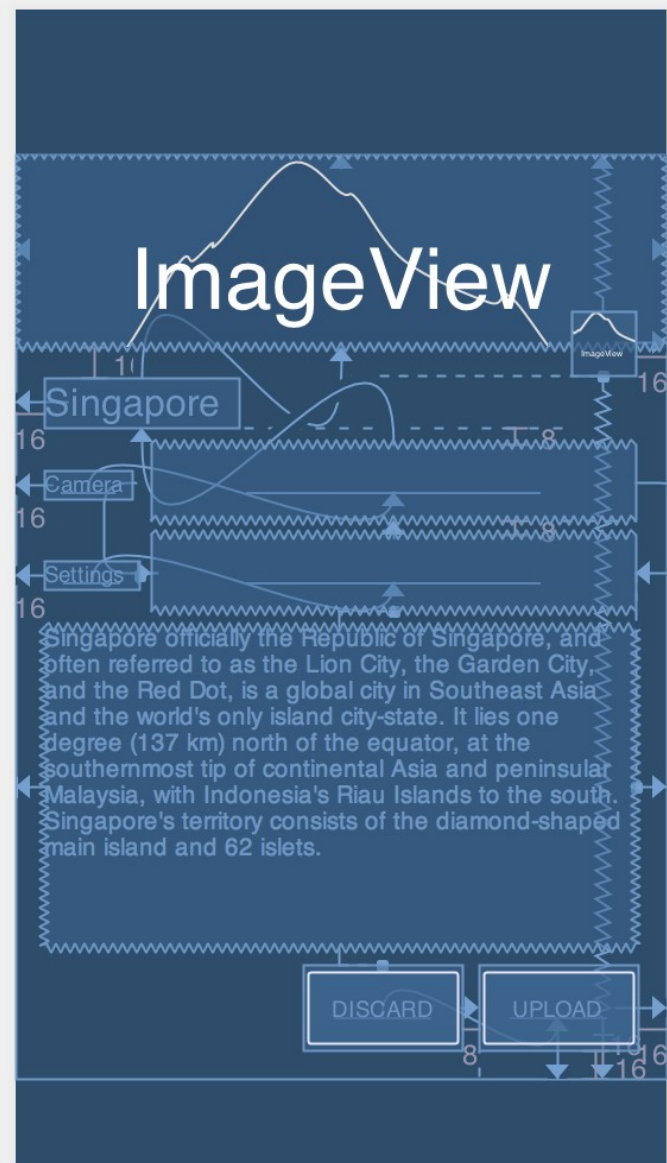
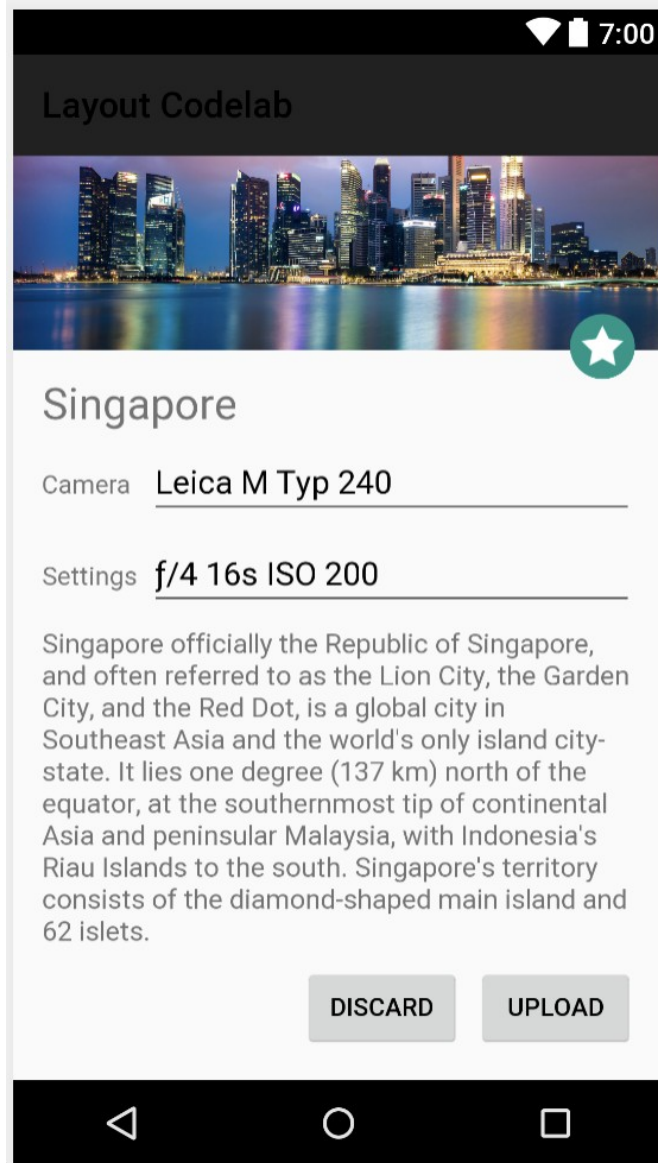
IT 4785. Phát triển ứng dụng cho thiết bị di động

Chapter 4. Graphical User Interfaces

ConstraintLayout



Ví dụ ConstraintLayout





ConstraintLayout - Ưu điểm

ConstraintLayout là một layout mới mà Google cung cấp để xây dựng giao diện cho các ứng dụng Android.

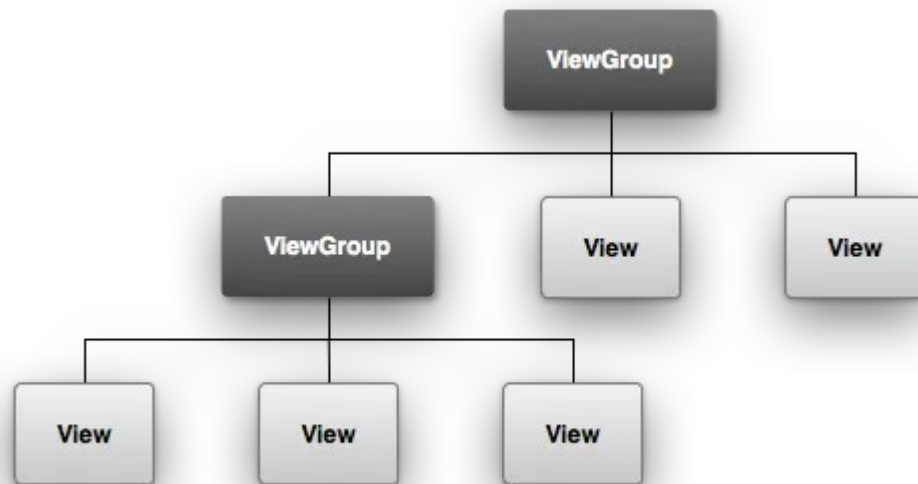
Trước đây, khi chưa có ConstraintLayout, các lập trình viên Android phải dùng đến các layout như RelativeLayout hay LinearLayout, TableLayout.

Trong số đó, RelativeLayout được sử dụng nhiều nhất. Tuy nhiên tạo ra một giao diện phức tạp thông thường phải kết hợp tất cả các dạng layout phổ biến đó lại với nhau.

Và một đặc điểm đặc trưng khi sử dụng các layout này là, chúng ta buộc phải code giao diện hoàn toàn bằng tay thông qua các dòng code XML. Công cụ kéo thả giúp thiết kế giao diện khi này cũng đã có nhưng chưa hoàn chỉnh,

ConstraintLayout - Ưu điểm

Bây giờ hầu như chỉ cần một layout duy nhất để thiết kế giao diện, đó chính là ConstraintLayout. Thêm nữa, do không cần phải kết hợp quá nhiều layout vào với nhau, nên sẽ không còn chuyện giao diện bị phân cấp như mô tả dạng cây, điều này có thể khiến hiệu năng của ứng dụng được cải thiện hơn.

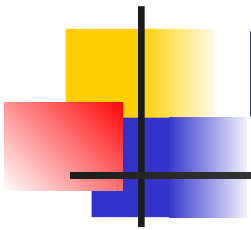




ConstraintLayout - Ưu điểm

Đặc biệt, công cụ thiết kế giao diện bằng cách kéo-thả dành cho ConstraintLayout đã hoàn thiện, mang đến khả năng tạo giao diện nhanh hơn và dễ dùng hơn so với kiểu code giao diện bằng XML truyền thống.

ConstraintLayout được hỗ trợ tương thích ngược đến API level 9 (Android 2.3) thông qua các gói thư viện support-vx. Vì vậy khi thiết kế giao diện với ConstraintLayout thì hầu hết các thiết bị Android trên thị trường đều có thể chạy được ứng dụng.



Layout ngầm định

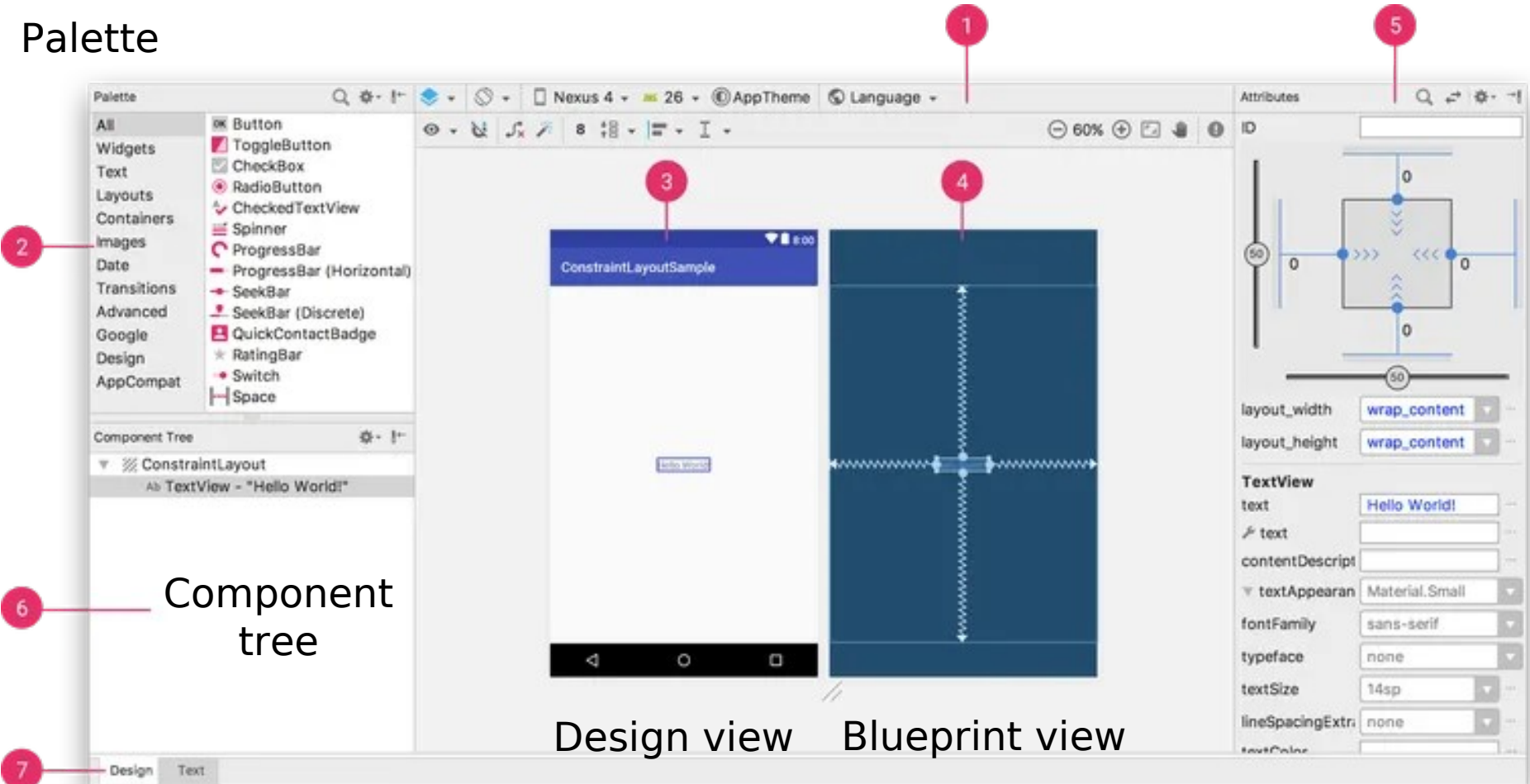
Với Android Studio 3.0 thì khi tạo mới project hay bất kỳ Activity nào khác thì `ConstraintLayout` sẽ là layout mặc định.

Giao diện thiết kế

Palette

Toolbar

Attribute



Design & Text



Các thành phần trong giao diện thiết kế

1. Toolbar: Thanh công cụ, nơi đây chứa một số công cụ quan trọng.
2. Palette: Bảng lựa chọn các view. Các widget, layout và nhiều thành phần giao diện khác ở đây.
3. Design view: Màn hình trực quan. Sử dụng màn hình này để thiết kế và xem trước kết quả thiết kế.
4. Blueprint view: Màn hình xanh đặc biệt dùng cho thiết kế.



Các thành phần trong giao diện thiết kế

5. Attribute: Nơi chứa đựng các thông số cho việc canh chỉnh giao diện, và một số các thuộc tính khác.
6. Component tree: Các view hiển thị ở Design view và Blueprint view cũng sẽ được hiển thị ở Component tree và được sắp xếp trực quan theo dạng cây để chúng ta dễ dàng theo dõi và quản lý.
7. Design & Text: Giúp thay đổi cách thức editor hiển thị giao diện ở dạng kéo thả (tab Design) hoặc code XML (tab Text).

Các constraint



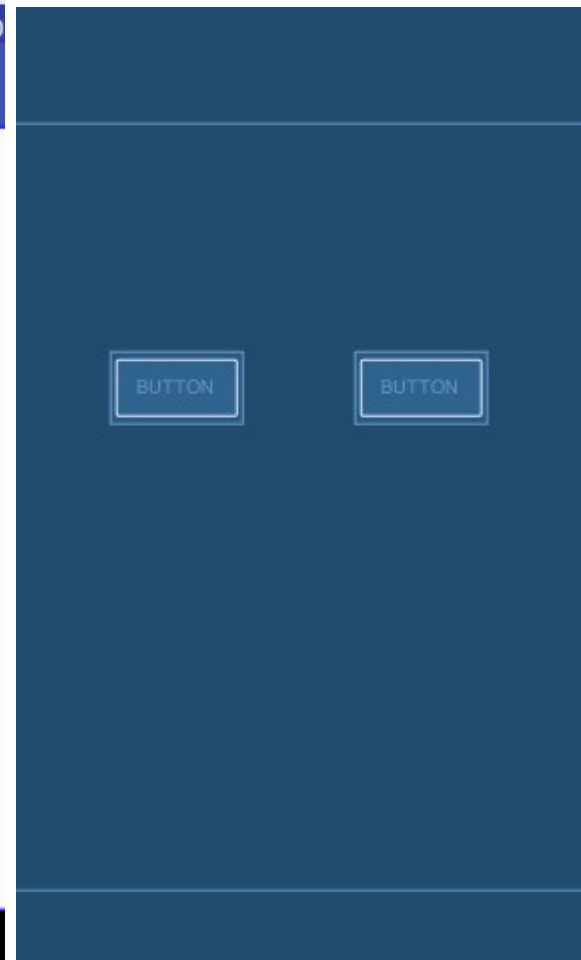
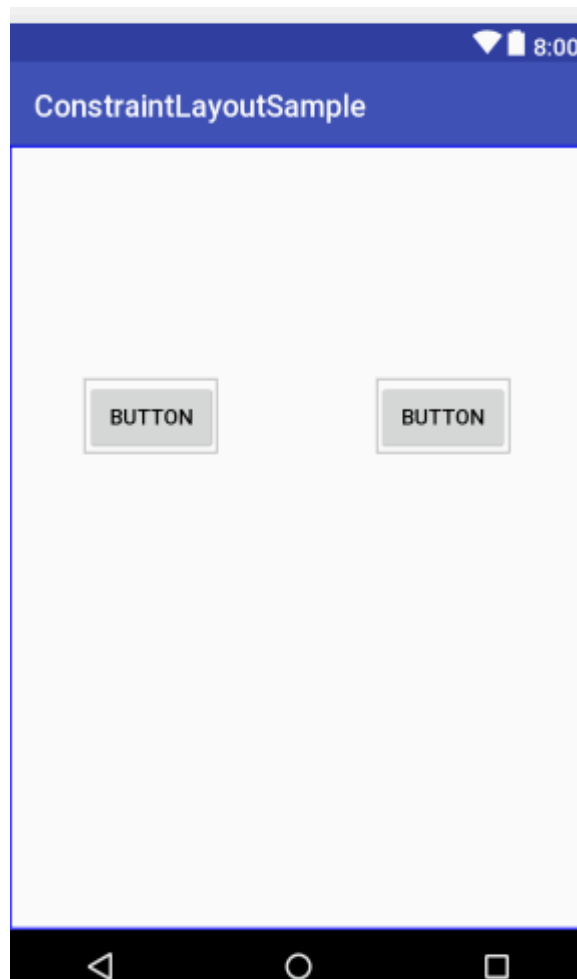
Constraint chính là các điểm neo của view vào các view nào đó khác.

Mỗi một view như vậy phải có ít nhất một điểm neo theo chiều ngang và một điểm neo theo chiều dọc

Constraint

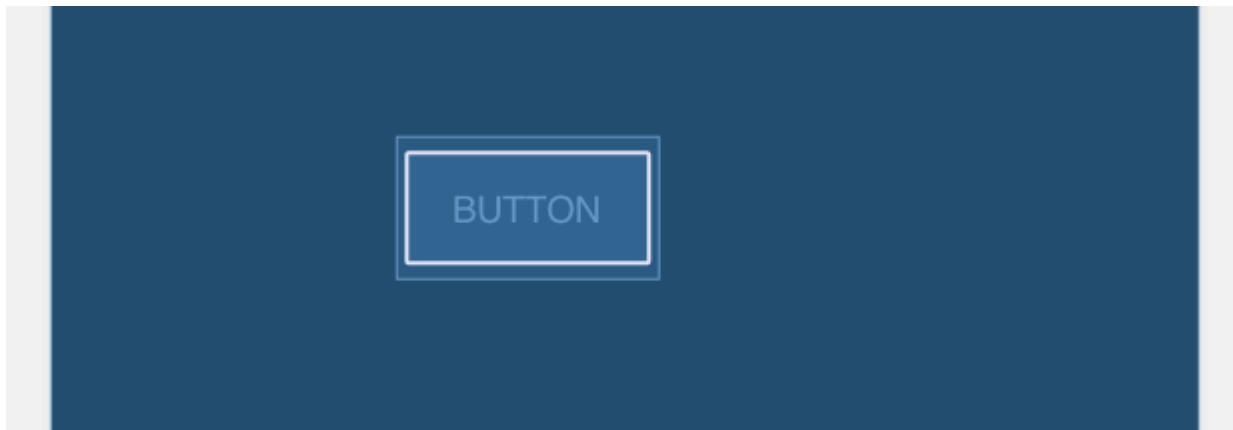
Sử dụng các chấm tròn ở mỗi cạnh của view.

Giữ chuột và kéo từ bất kỳ chấm tròn nào đến biên của màn hình, hay đến một chấm tròn của một view nào đó khác, sẽ thấy các view sẽ “xích lại” gần nhau sau khi tạo constraint theo cách này.



Tạo constraint cho view

Đường constraint của view sau khi bạn tạo ra sẽ có hai loại. Nếu neo một bên của view (trên, dưới, trái hay phải) vào màn hình hay một view nào đó, ta sẽ thấy một đường thẳng và có thông tin về khoảng cách của điểm neo đến view được neo.



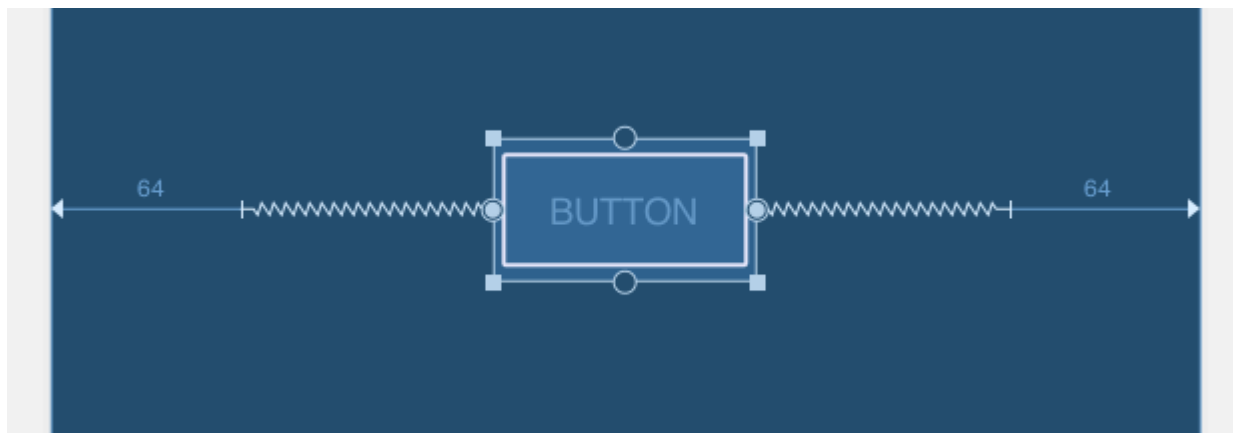
Tạo constraint cho view

Nhưng nếu thêm một điểm neo đối xứng với điểm neo đầu. Như ví dụ trên khi neo vào trái, nay thêm một neo bên phải, thì đường constraint lại hiển thị dạng lò xo, giống như thể view được cố định bằng một kết cấu gì đó khá động ở đây.



Xóa constraint

Đưa trỏ chuột đến chấm tròn ở cạnh đó cho đến khi xuất hiện màu đỏ, rồi nhấn.





Tạo Baseline Constraint Cho View

Nếu như constraint cho view trên kia là các điểm neo giữa view với view, thì Baseline constraint thì lại là sự canh chỉnh các text bên trong một view với nhau.

Việc canh chỉnh dựa trên baseline rất thích hợp cho các widget như TextView, EditText hay Button.

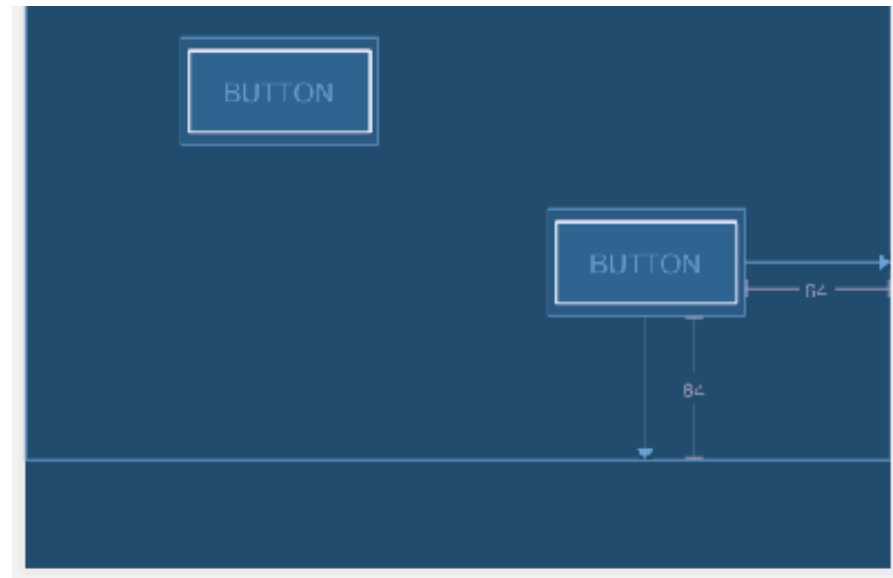
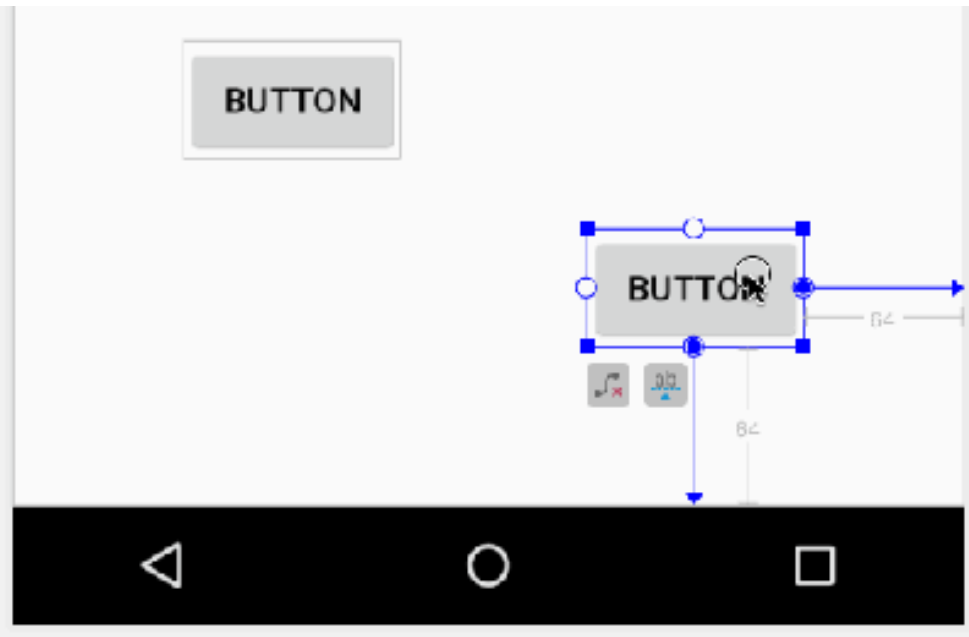
Để tạo ra baseline constraint, hãy nhấn vào một view, rồi nhấn vào nút có ký hiệu xuất hiện bên dưới view đó.



Baseline sẽ xuất hiện ngay dưới text của widget.

Kéo baseline của view này vào baseline của view khác để tạo ra một baseline constraint như constraint bình thường của của view vậy.

Tạo Baseline Constraint Cho View

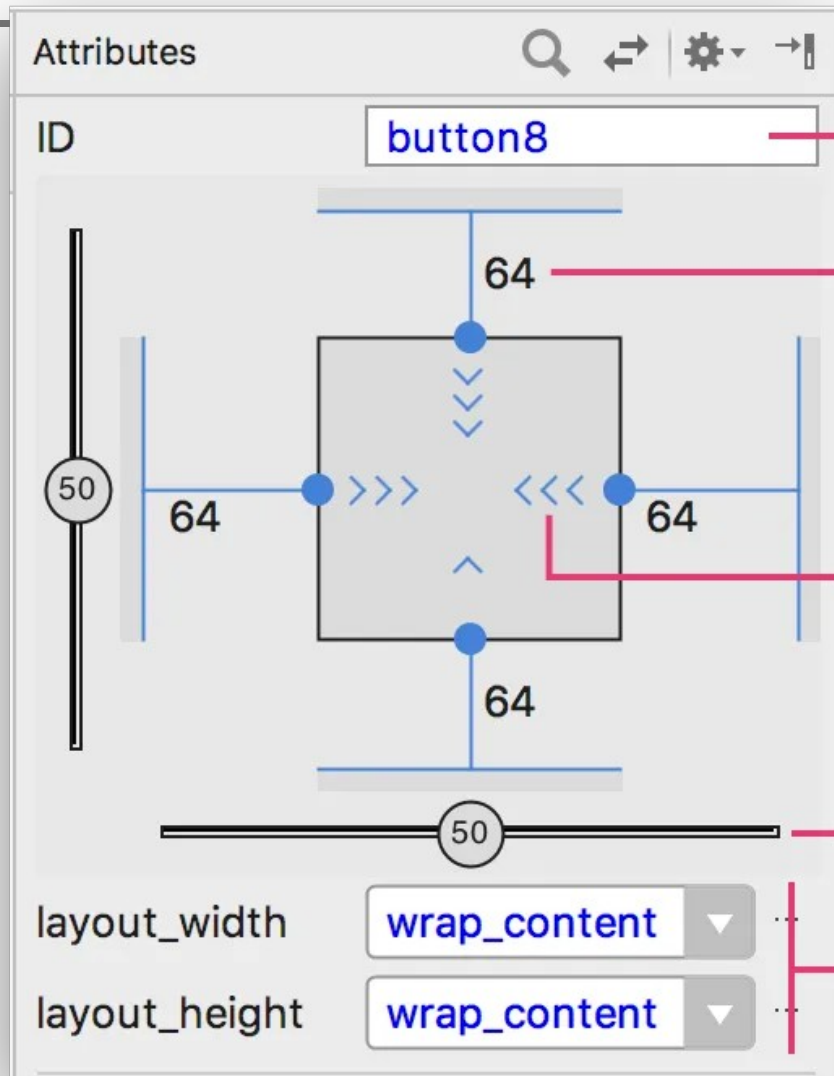


Các thuộc tính

1. ID Của View

2. Margin Của View

Margin chính là khoảng cách của một view đến các view khác.



3. Các Khoảng Cách Tới Các Thành Viên Bên Trong



Fixed: Nếu chỉ định khoảng cách kiểu này, có thể điền giá trị ở field `layout_width` và `layout_height`.

Match Constraint:

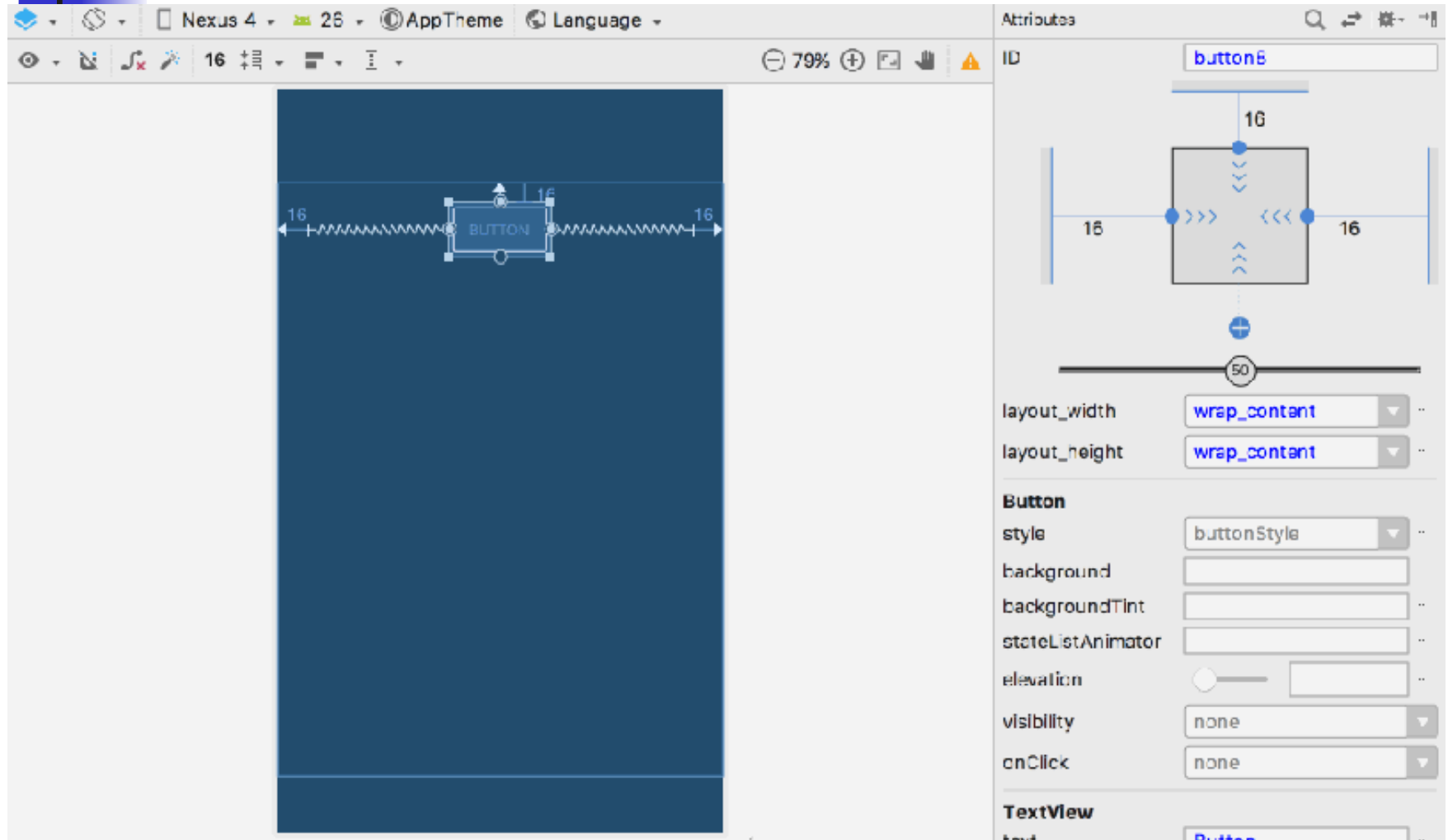


- Kiểu khoảng cách này gần giống với `match_parent` quen thuộc ở các layout khác.
- Chỉ khác ở chỗ Match Constraint không đến nỗi đẩy các view khác ra khỏi màn hình mà chiếm lấy hết không gian ở view cha.
- Vẫn chiếm không gian của view cha nhưng chiếm trong khả năng cho phép, tức là vẫn tuân thủ theo quy luật mà các constraint đã tạo, vẫn chừa không gian cho các view khác xuất hiện.
- Và khi chỉ định kiểu khoảng cách là Match Constraint thì code XML của view sẽ mang giá trị là ***0dp***.



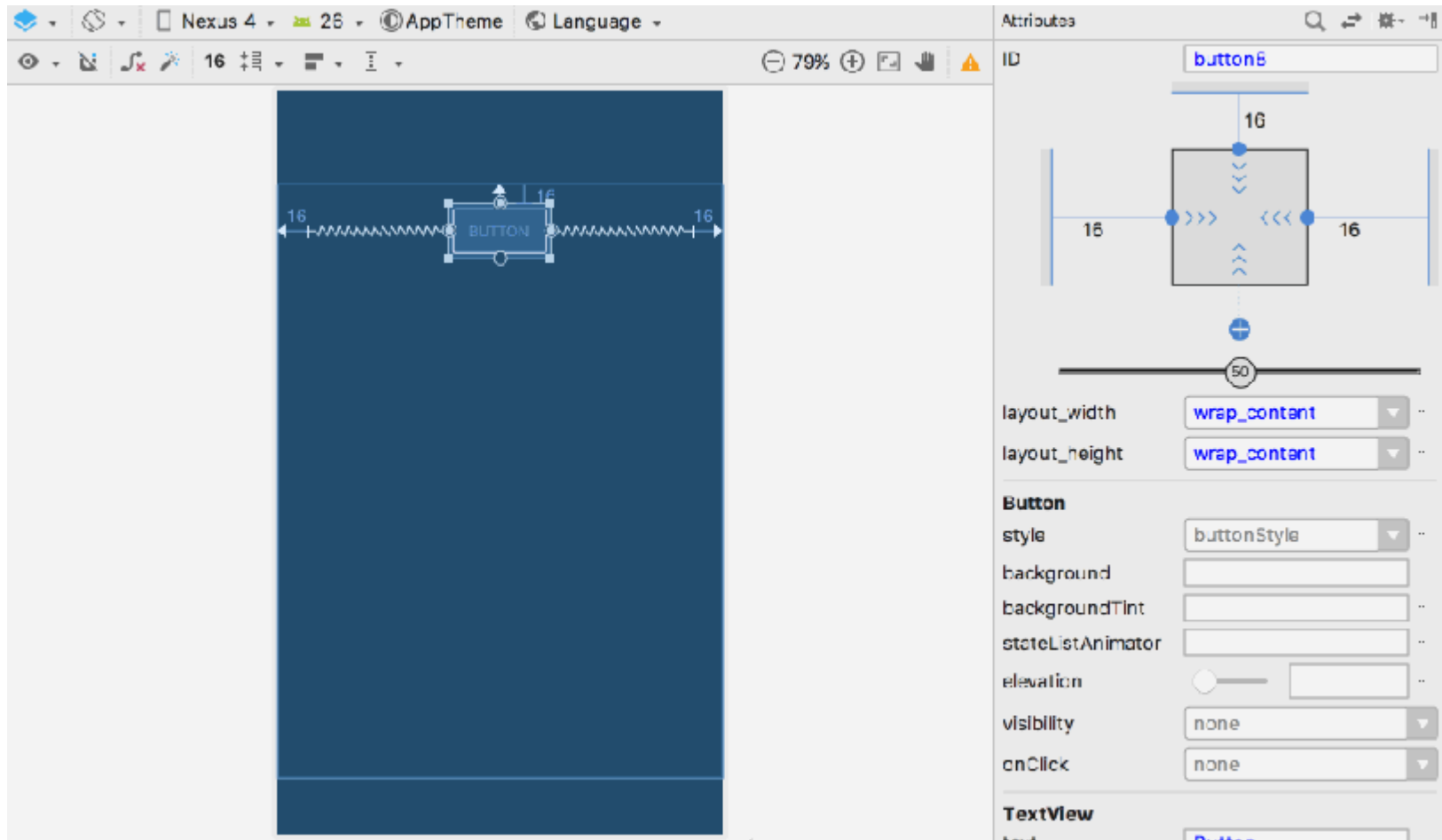
Wrap Content: khoảng cách này thì tương tự với `wrap_content`.

3. Các Khoảng Cách Tới Các Thành Viên Bên Trong

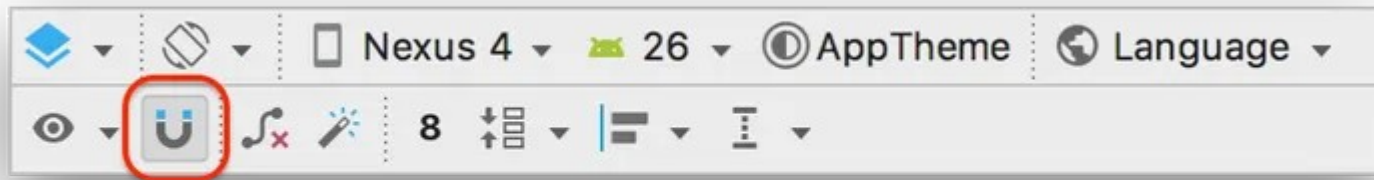


4. Bias

Giống như trọng số (weight) trong LinearLayout.

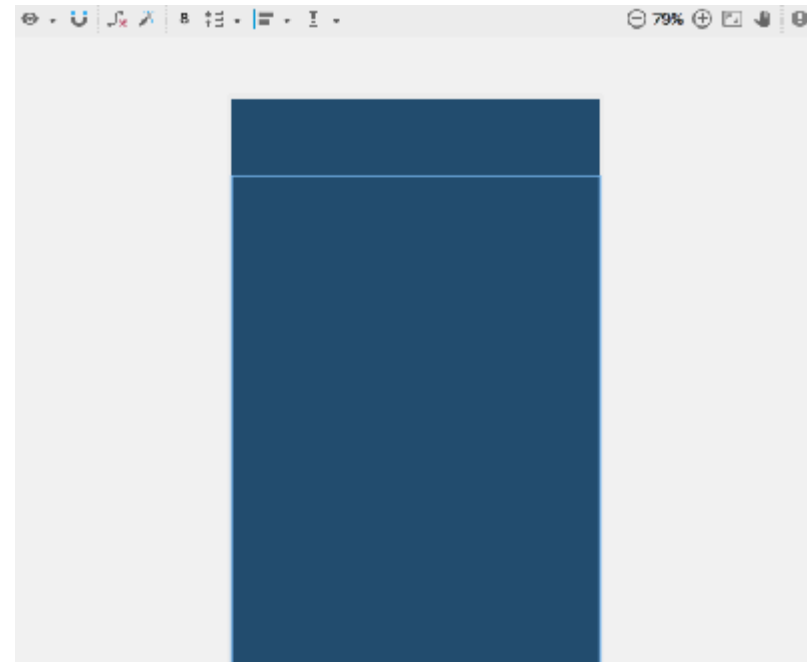


Chức Năng Autoconnect (Tạo Constraint Tự Động)

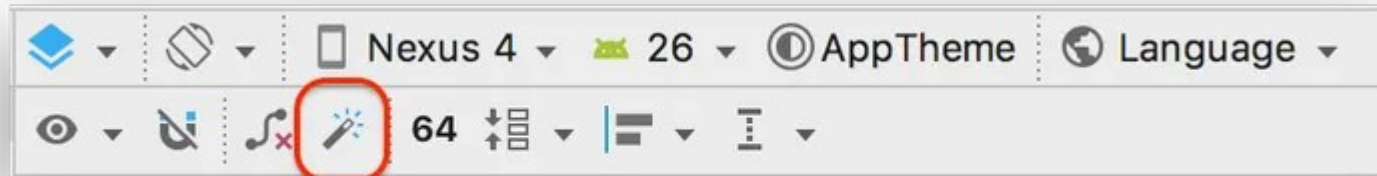


Nếu như ở phần trước, ta phải kéo, neo thì mới có được các constraint ưng ý.

Thì với một cú click chuột ở phần này, ta đã có thể nói cho hệ thống biết nên tự tạo các constraint ngay khi thả một view từ palette vào design view hoặc blueprint view.

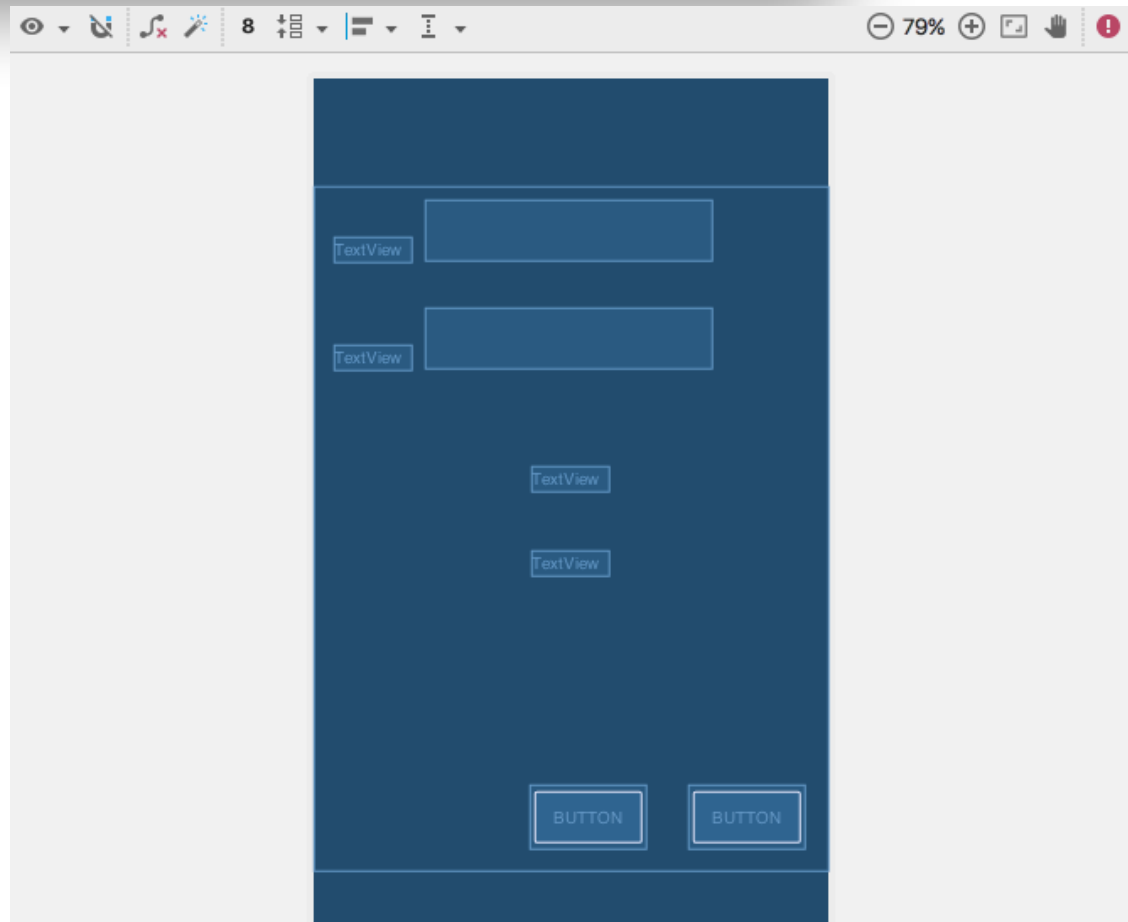


Chức Năng Infer Constraints (Tự Suy Luận Các Constraint)



Vì chức năng này khi được kích hoạt sẽ tự nó tính toán, suy luận ra các constraint để khớp với bố cục hiện thời của layout.

Để sử dụng tính năng này, bạn hãy tìm đến icon toolbar được khoanh đỏ như trên.



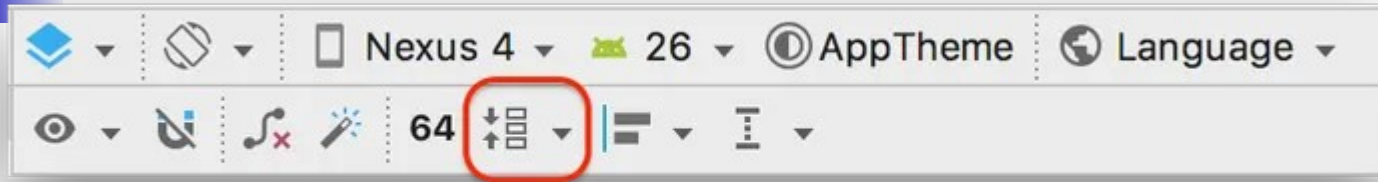


Chức Năng Infer Constraints (Tự Suy Luận Các Constraint)

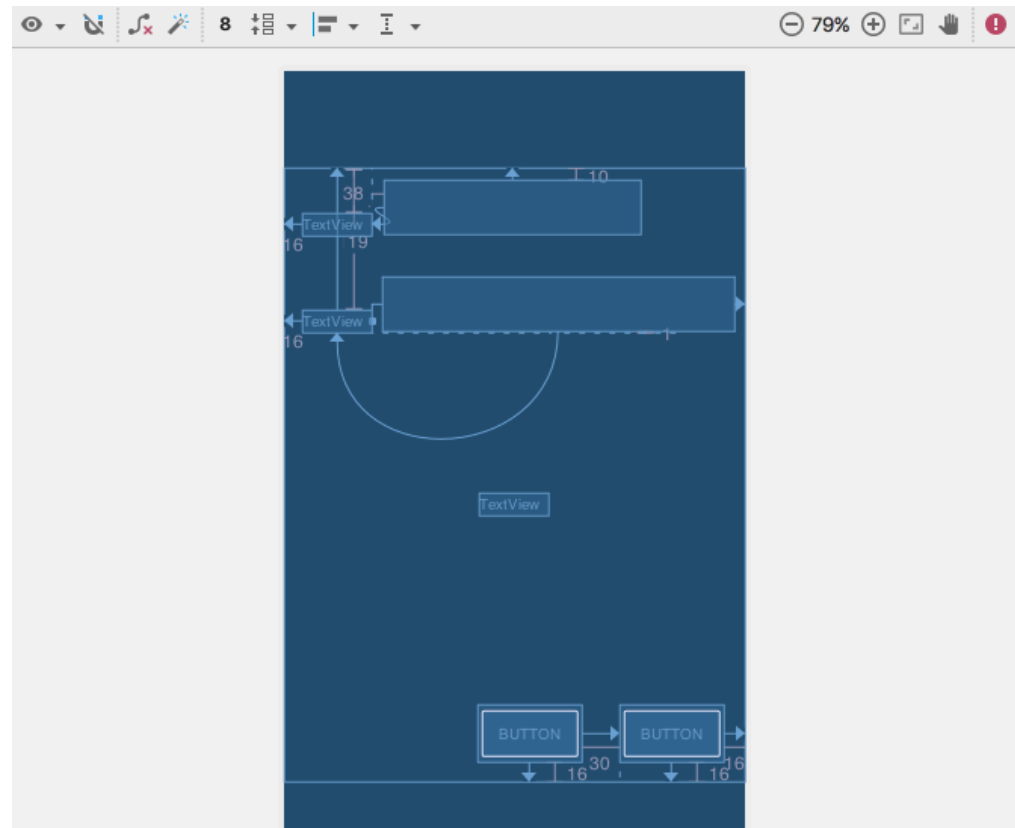
Sự khác nhau giữa tính năng Infer Constraints và Autoconnect:

- Infer Constraints sẽ suy luận ra tất cả các constraint cho tất cả các view trong layout, để đạt được một bố cục như ý muốn của lập trình viên,
- Autoconnect thì sẽ tạo các constraint cho bản thân view đang được tương tác.

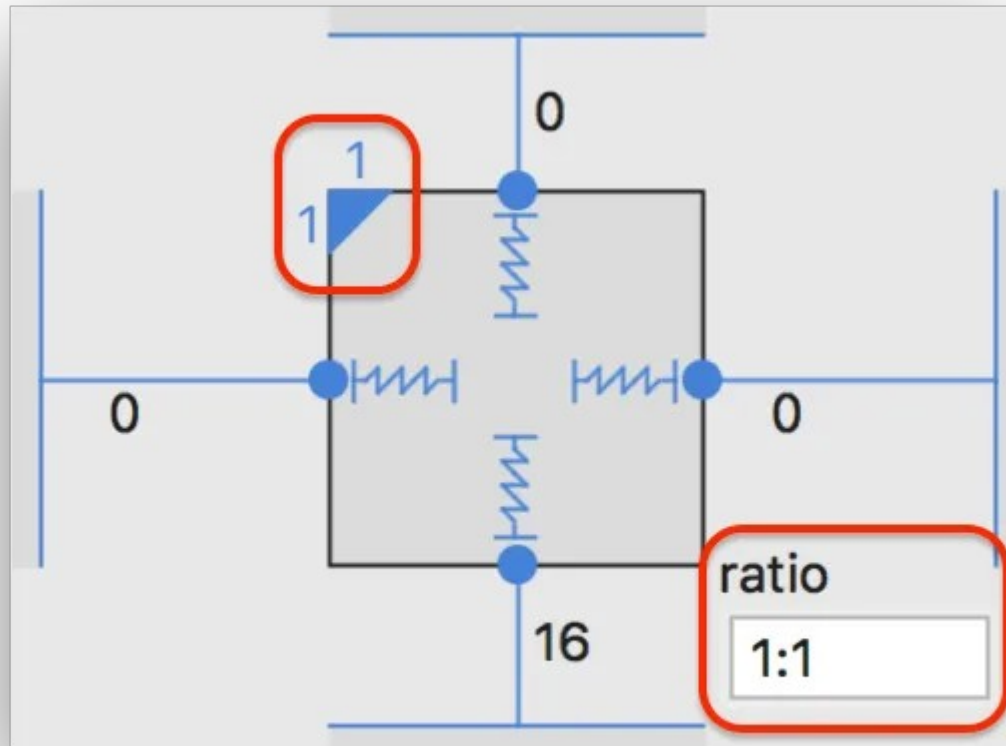
Chức Năng Pack



Kéo dẫn hết không gian của view theo chiều ngang hay dọc, sự kéo dẫn này không đẩy các view khác đi khỏi vị trí của chúng.

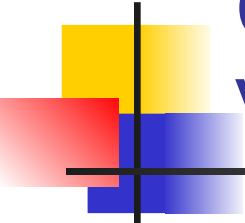


Chức Năng Ratio (Điều Chỉnh Kích Cỡ View Theo Tỷ Lệ)



Tạo các view bên trong ConstraintLayout với kích thước theo tỉ lệ ngang-dọc.

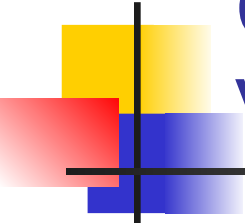
Chẳng hạn 16:9, hay 4:3.



Chức Năng Ratio (Điều Chỉnh Kích Cỡ View Theo Tỷ Lệ)

Khi đã chỉ định ratio cho view, thì kích thước của view sẽ được xác định bởi hệ thống.

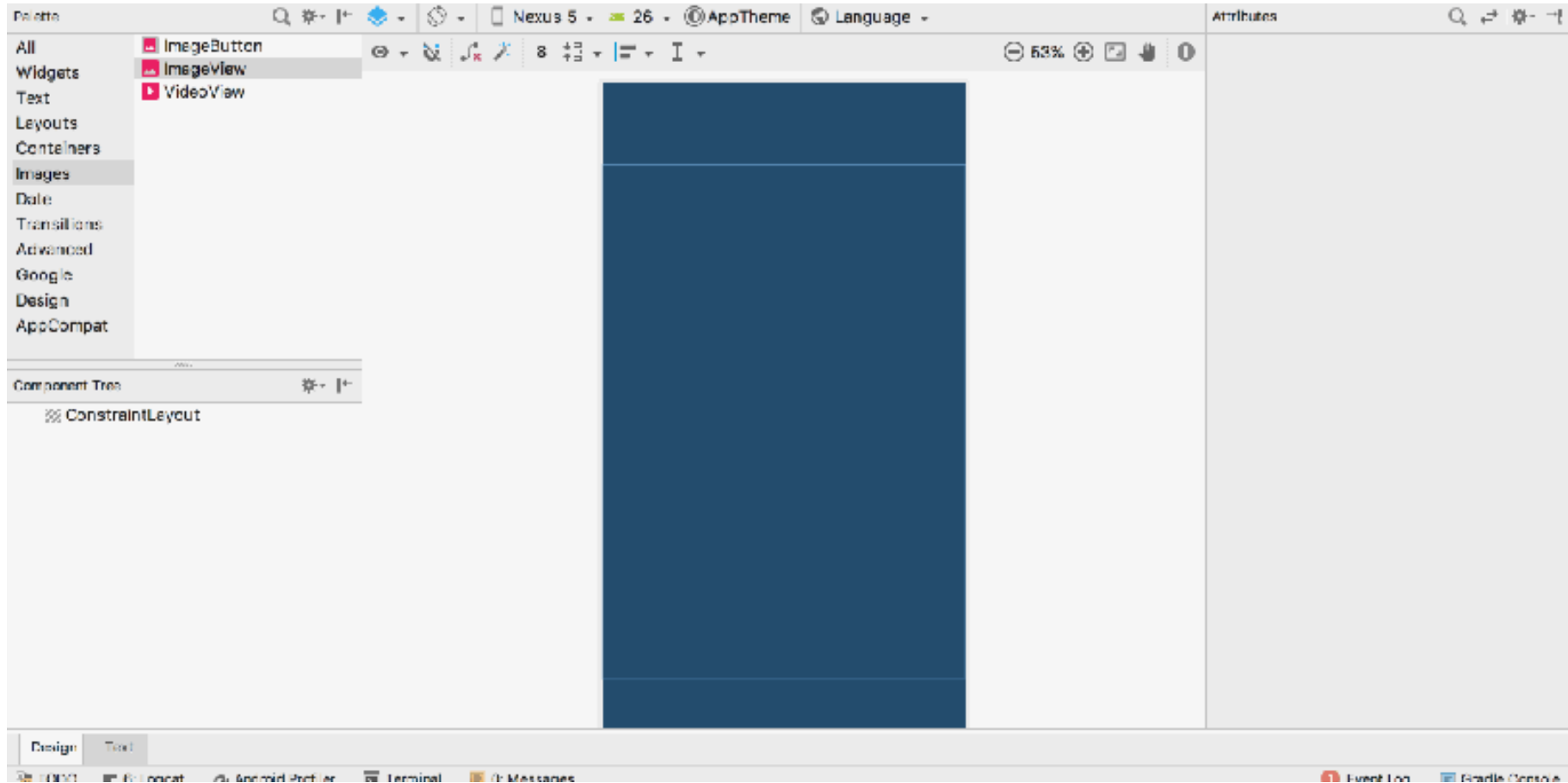
Để có thể thiết lập ratio cho view, phải chỉ định một trong hai giá trị (hoặc cả hai) `layout_width` và `layout_height` của view thành `match_constraint` (hoặc là `0dp`)



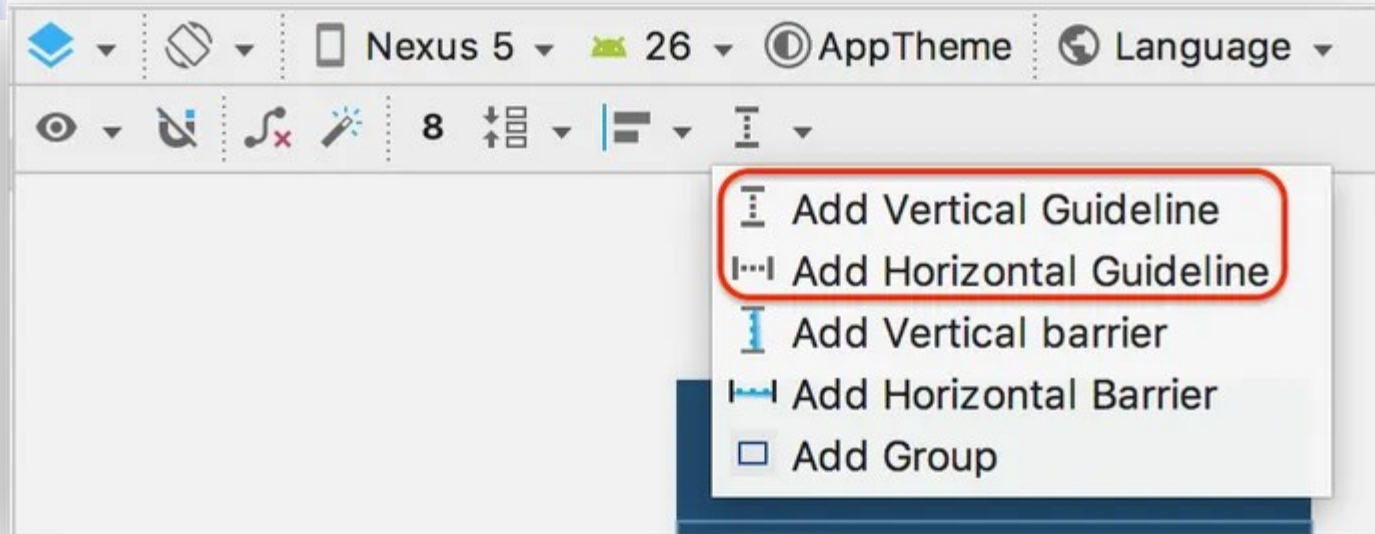
Chức Năng Ratio (Điều Chỉnh Kích Cỡ View Theo Tỷ Lệ)

Sau khi ta thiết lập một trong hai hướng (hoặc cả hai) của view là `match_constraint`, thì ở view inspector, chỗ hình vuông đại diện cho view trong cửa sổ này, sẽ xuất hiện một hình tam giác báo hiệu cho biết có thể click vào đó để thiết lập ratio cho view.

Chức Năng Ratio (Điều Chỉnh Kích Cỡ View Theo Tỷ Lệ)



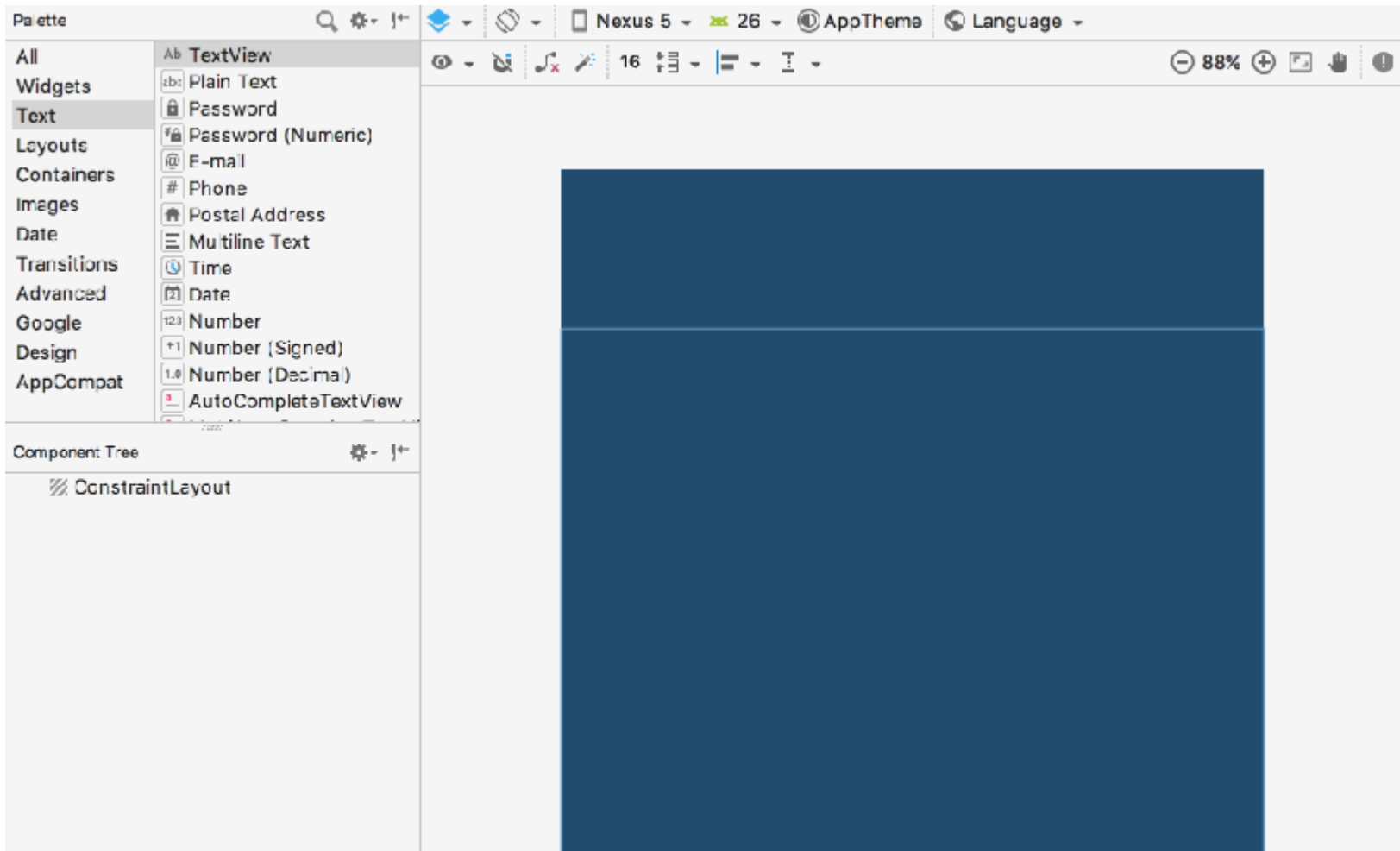
Chức Năng Guideline (Neo Constraint Vào Đường Biên)



Nếu có nhiều view được canh chỉnh theo một trật tự thẳng hàng nào đó (thẳng hàng ngang hay thẳng hàng dọc), ta có thể cân nhắc sử dụng guideline.

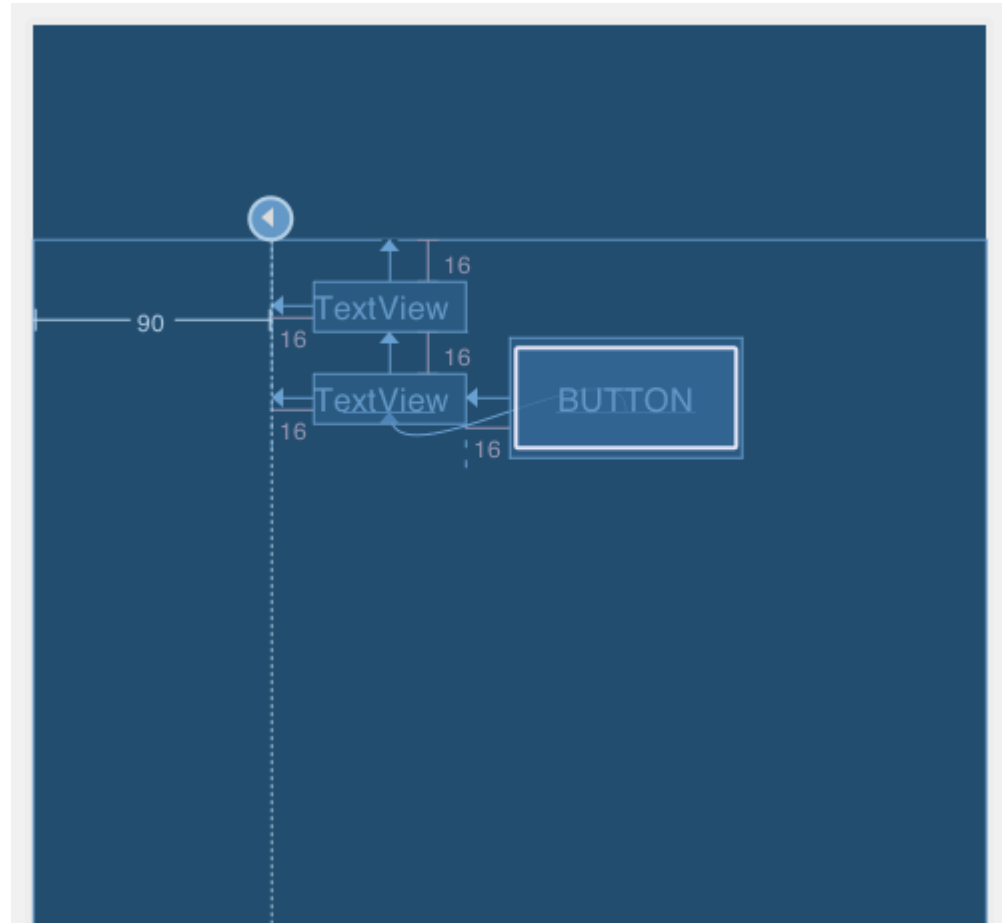
Các guideline này là các đường thẳng không hề hiển thị lên giao diện khi thực thi ứng dụng, nó chỉ được nhìn thấy khi thiết kế mà thôi, và nhờ vào các đường guideline ẩn này, ta có thể neo các view vào nó, để tạo ra một trật tự thẳng hàng nhất định.

Chức Năng Guideline (Neo Constraint Vào Đường Biên)

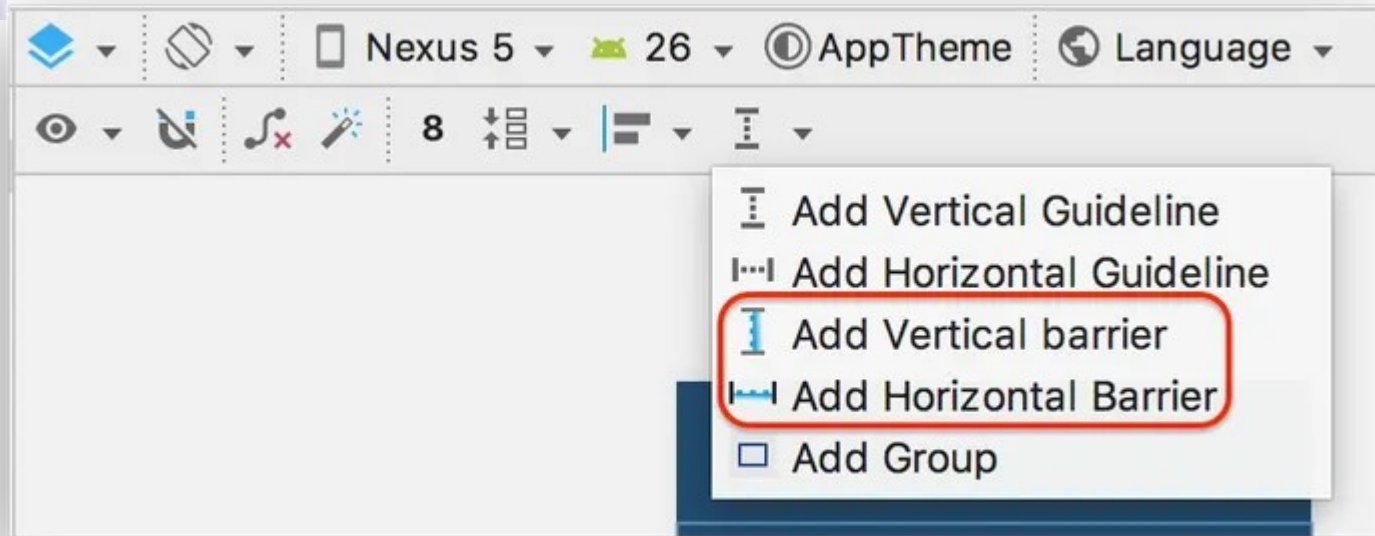


Chức Năng Guideline (Neo Constraint Vào Đường Biên)

Ngoài việc canh chỉnh guideline theo biên trái như trên, ta còn có thể nhấn vào hình tròn ở đầu guideline để thay đổi cách canh chỉnh theo biên phải, hay theo phần trăm.



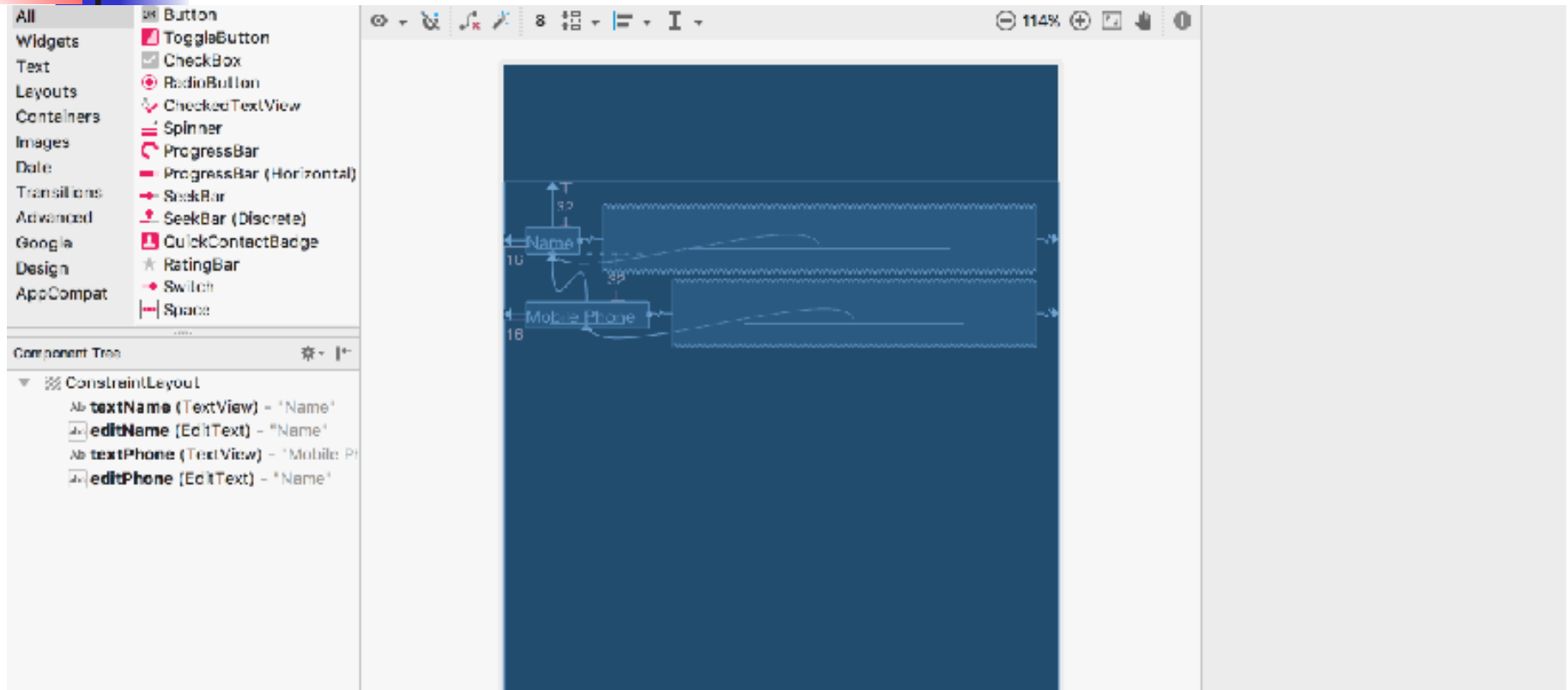
Chức Năng Barrier (Neo Constraint Vào Đường Biên Động)



Guideline giúp chúng ta tạo ra các đường biên, và canh chỉnh view dựa trên các đường biên đó.

Mặc dù ta có thể di chuyển được guideline như ví dụ trên. Nhưng guideline vẫn mang một công dụng “cứng”, tức là chỉ có thể chỉnh sửa các guideline khi nó còn đang ở dạng thiết kế, khi thực thi ứng dụng rồi thì guideline ở đâu view sẽ ở đó, mặc cho có sự chồng lấp giữa các view với nhau.

Chức Năng Barrier (Neo Constraint Vào Đường Biên Động)



Như đã nói, đầu tiên thêm barrier (dọc hay ngang) bằng cách chọn trên toolbar như hình trên kia.

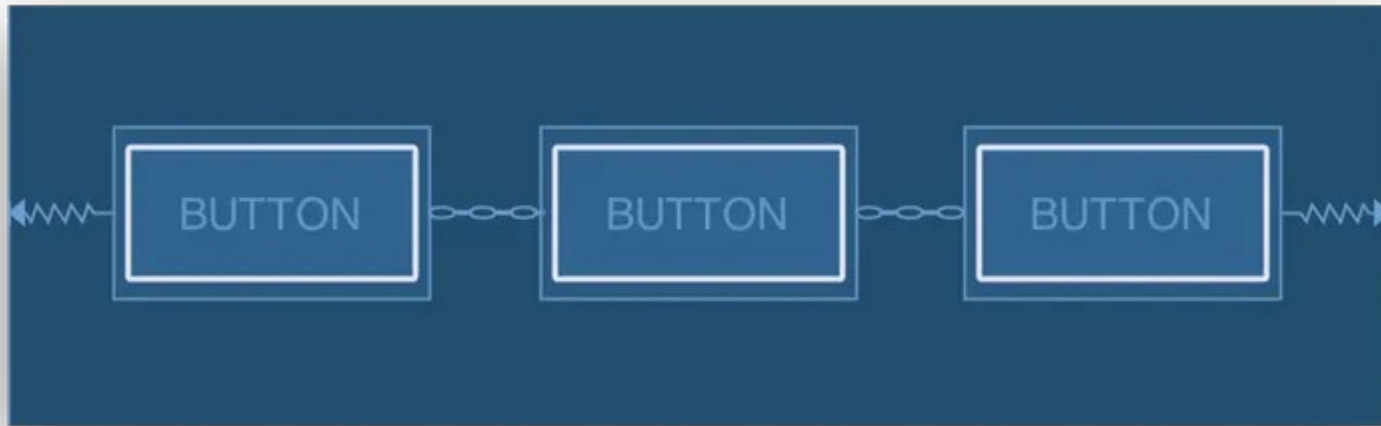
Hoặc click chuột phải vào blueprint view thì cũng sẽ thấy tùy chọn tương tự.



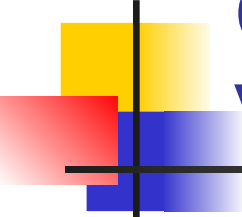
Chức Năng Barrier (Neo Constraint Vào Đường Biên Động)

- Sau đó phải kéo các view muốn “chặn” lại một cách cơ động, bằng cách kéo thả các view đó vào barrier vừa tạo ở component tree.
- Chỉ định thuộc tính barrierDirection là right, đây chính là phía mà barrier sẽ “chặn động” các view.
- Cuối cùng, có thể neo các view khác, như các EditText ở ví dụ trên, vào barrier này.
- Xong. Ta có thể thử nghiệm tính hiệu quả của barrier bằng cách thay đổi text của các TextView. Ta sẽ thấy barrier chạy động theo biên phải của các thành phần con của nó như thế nào. Và các view khác khi neo vào barrier cũng sẽ tự dịch chuyển như thế nào.

Chức Năng Chain (Xâu Chuỗi Các View Lại)

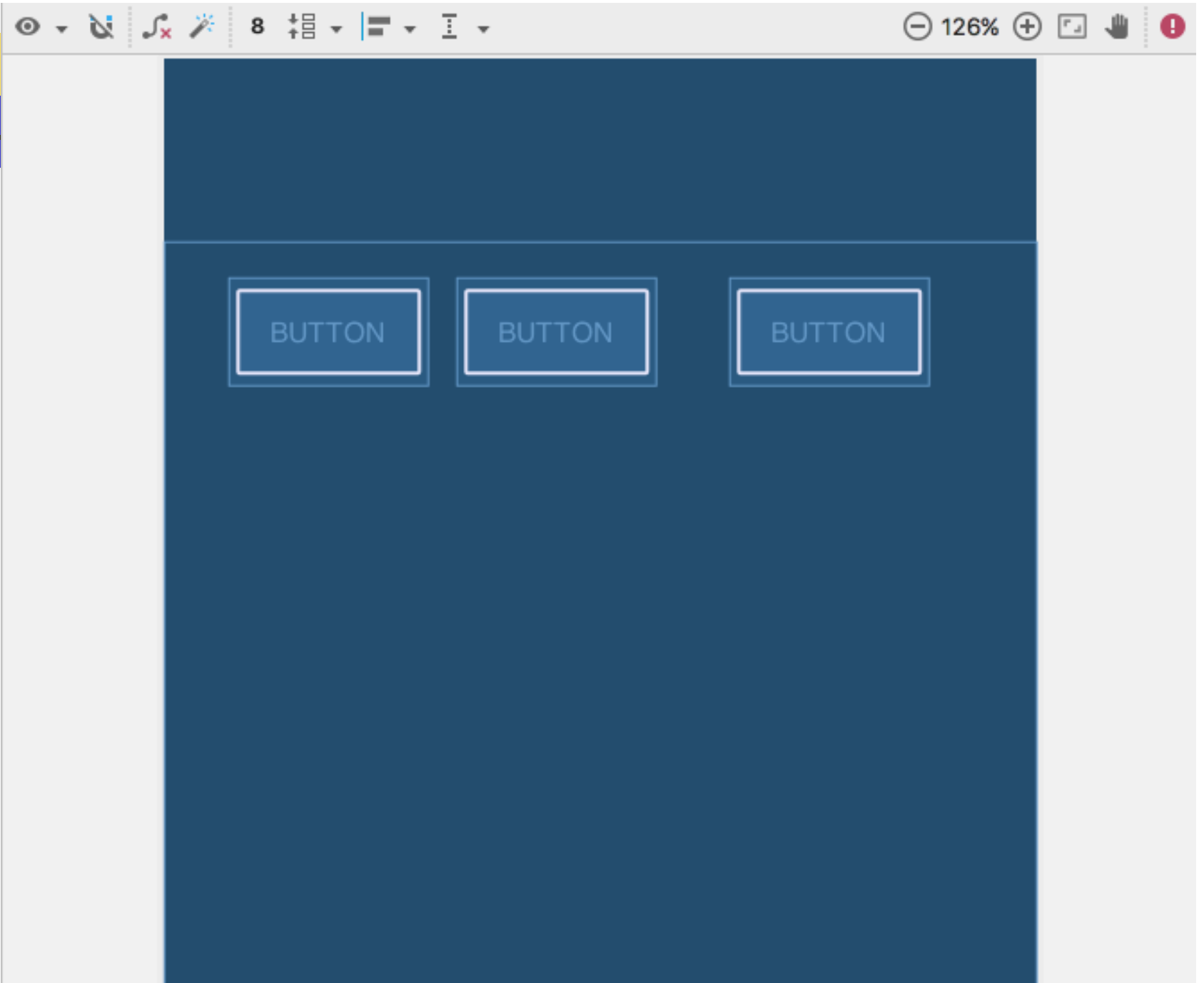


- Canh chỉnh vị trí của nhiều view một cách tự động.
- Việc canh chỉnh này giống như hệ thống đã dùng dây xích để cột chúng lại vậy, và sau đó hệ thống sẽ dàn các view đó ra hoặc co chúng lại tùy thuộc vào kích cỡ màn hình và các thiết lập của ta.
- Hệ thống gọi đây là chain.

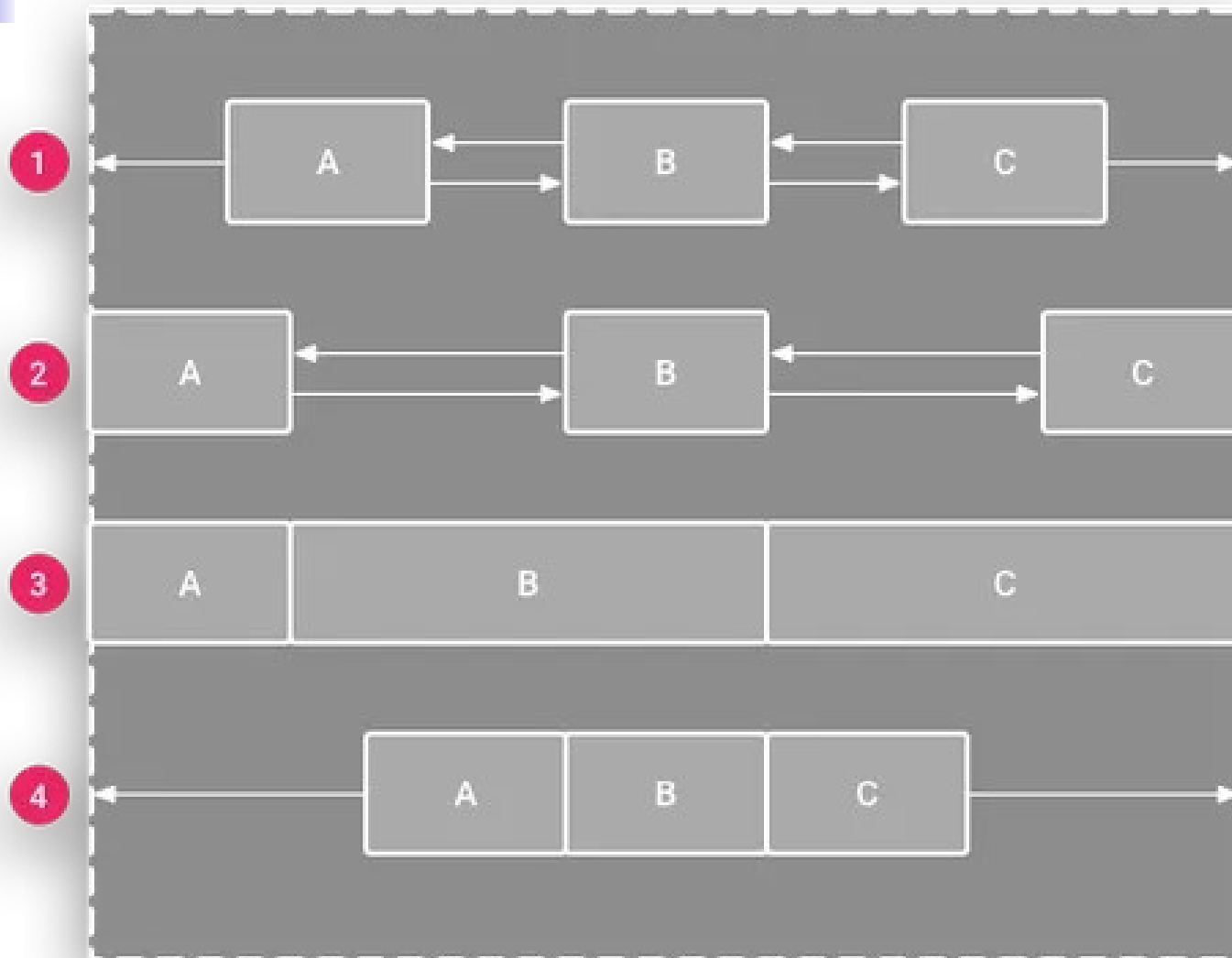


Chức Năng Chain (Xâu Chuỗi Các View Lại)

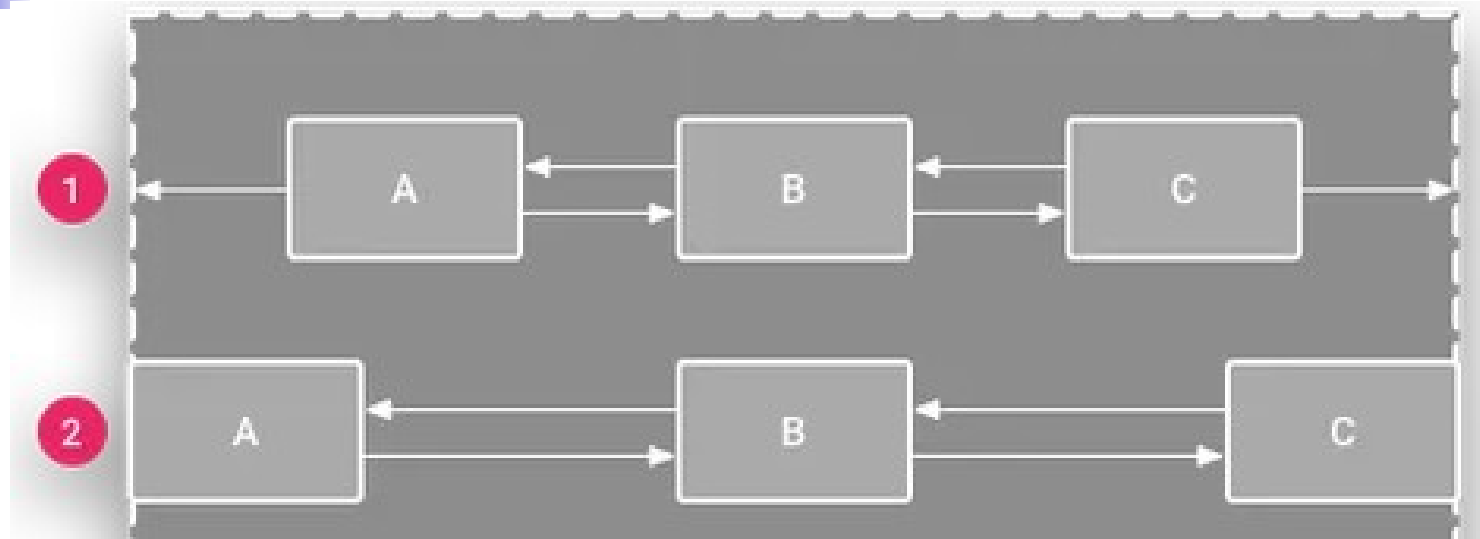
- Để gom các view vào trong một chain, bạn chỉ cần chọn hết các view muốn gom bằng cách nhấn giữ phím Shift trong lúc click chọn từng view.
- Rồi click phải lên bất kỳ view nào trong số chúng, và chọn Chain => Create Horizontal Chain (hay Create Vertical Chain).
- Sau khi gia nhập vào một chain, sẽ có một sợi xích ràng buộc các view lại với nhau, và khi đó vị trí các view sẽ dàn đều không gian của chúng.



Các loại chain



Các loại chain



1. Spread: Đây là kiểu dàn đều các view dựa vào không gian của chúng theo phương ngang hoặc dọc. Đây là kiểu sắp xếp mặc định khi bạn tạo mới một chain.
2. Spread inside: Kiểu này cũng sẽ dàn đều các view, nhưng nó sẽ tôn trọng constraint của view đầu và cuối trong một chain. Như trên hình, nếu các view A và C đều set margin ở các biên là 0dp thì chúng sẽ dính chặt vào biên như vậy.

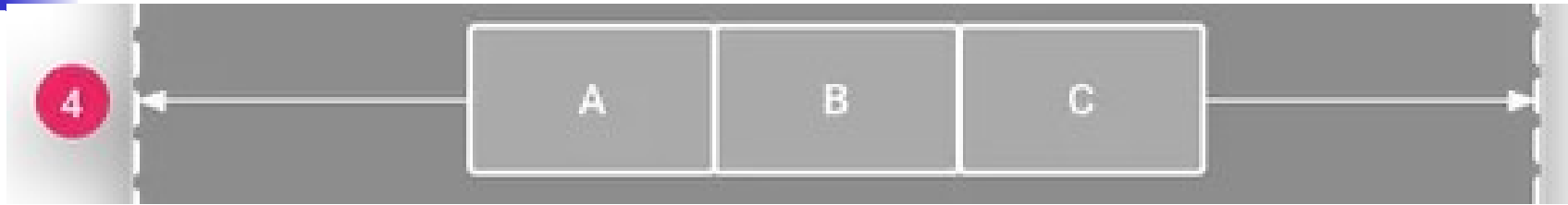
Các loại chain



3. Weight: Cách này tương tự như chỉ định trọng số `layout_weight` trong `LinearLayout` vậy.

Để sử dụng được `weight` trong `ConstraintLayout` thì phải chỉ định các view trong chain về `match_constraint`, rồi tìm đến thuộc tính `horizontal_weight` hoặc `vertical_weight` để thiết lập trọng số này cho từng view.

Các loại chain

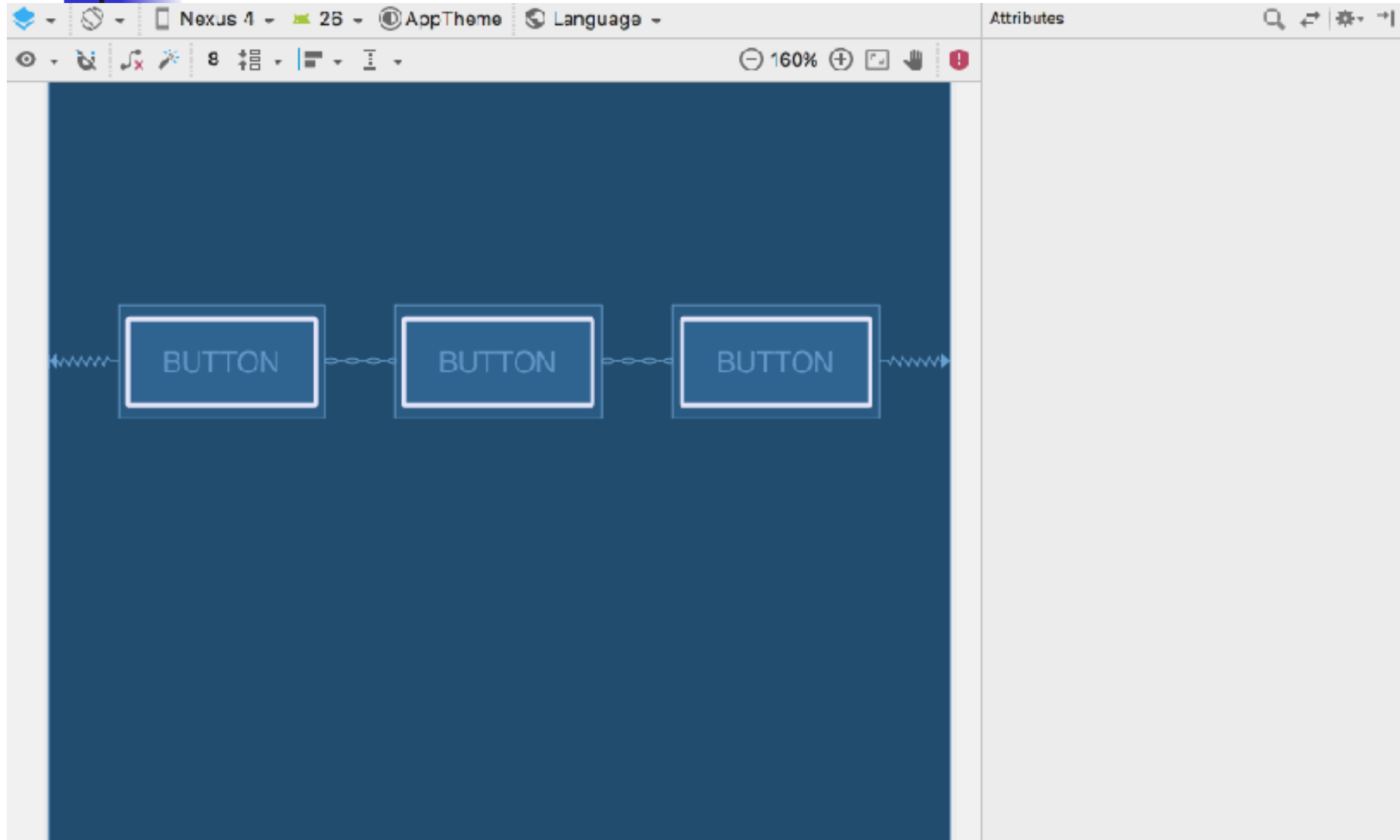


4. Packed: Kiểu “đóng gói” các view lại thành một “cục” sát vào nhau. Sau khi đóng gói các view lại xong, ta có thể sử dụng bias để thay đổi độ lệch theo chiều ngang hoặc dọc cho các gói này

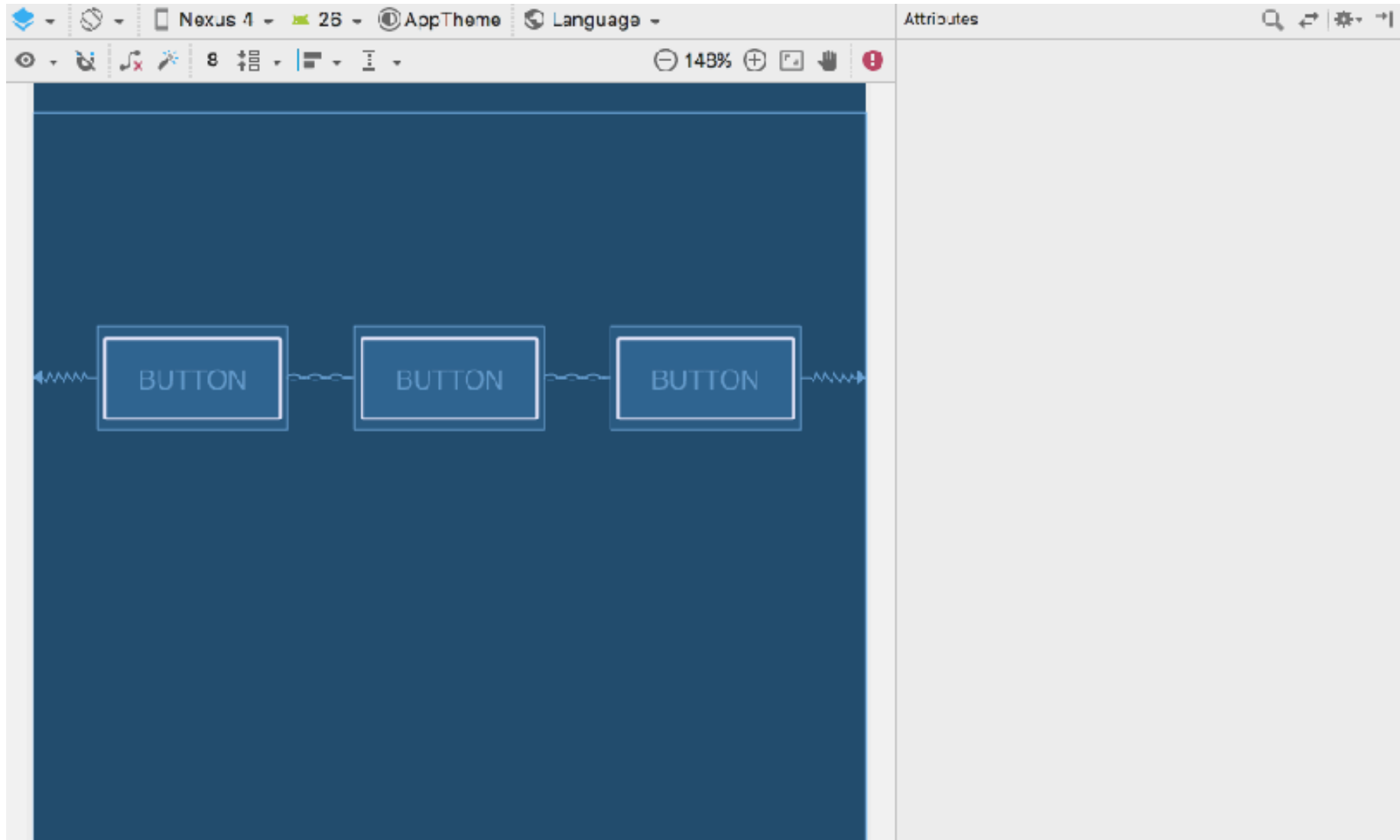
Để thay đổi từng loại chain đã nói đến trên đây, rất đơn giản, bạn hãy đưa trỏ chuột vào bất kỳ một view nào trong chain, khi thấy xuất hiện nút ở dưới view, ta hãy nhấn vào nó để lần lượt thay đổi qua các loại chain.



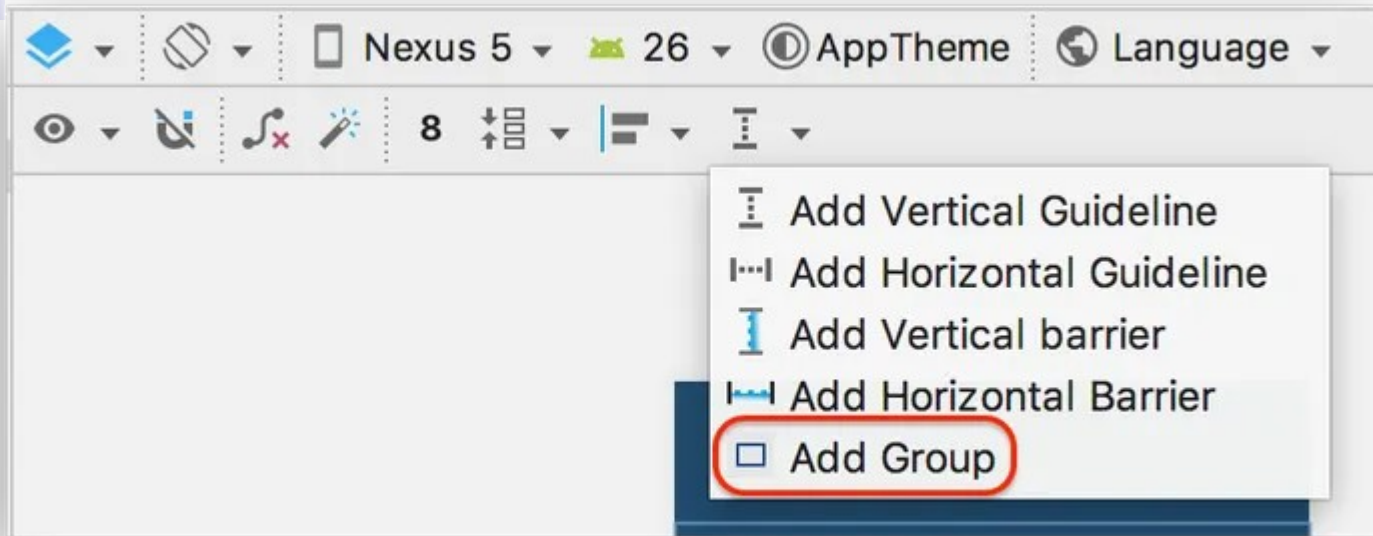
Chuyển đổi qua lại giữa các loại chain thứ 1 (Spread), 2 (Spread inside) và 4 (Packed)



Minh họa cho chain ở loại thứ 3 (weight) Set trọng số cho các view lần lượt là 1-2-1



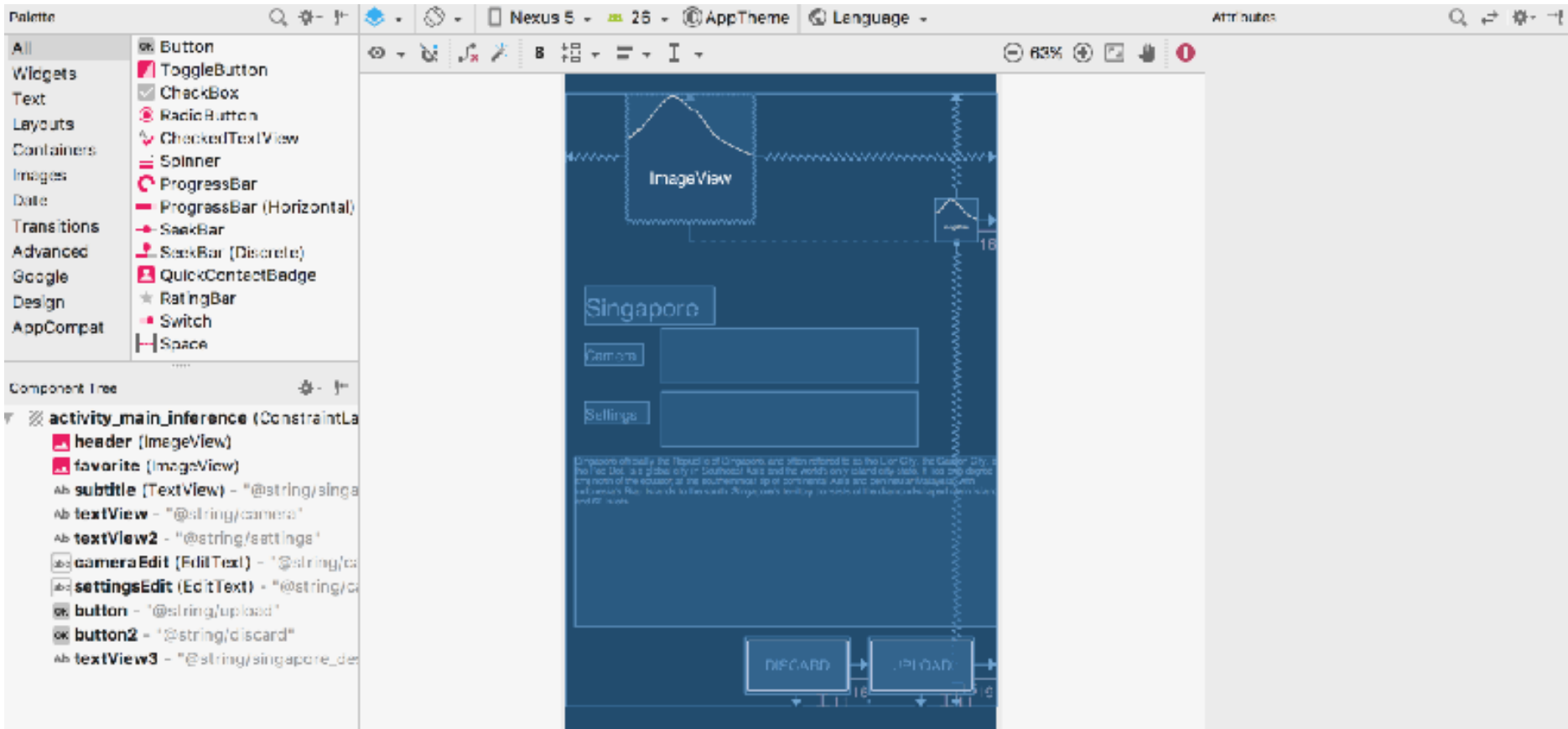
Chức Năng Group (Gom Nhóm)

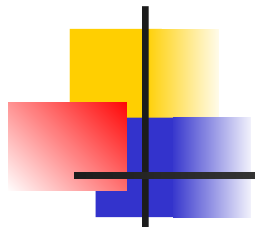


Đây không phải ViewGroup mà chúng ta từng biết. Bởi vì chẳng chứa đựng view nào bên trong cả.

Group ở ConstraintLayout chỉ chứa địa chỉ các view, để bạn dễ dàng thiết lập một số tham số chung cho các view mà nó đang chứa đựng

Chức Năng Group (Gom Nhóm)





Next: Thực Hành Xây Dựng Giao Diện
Bằng ConstraintLayout



Tài liệu tham khảo

Build a Responsive UI with ConstraintLayout

<https://developer.android.com/training/constraint-layout>

ConstraintLayout

<https://yellowcodebooks.com/2017/10/03/constraintlayout-phan-1-tim-hieu-cac-thanh-phan-co-ban/>

<https://yellowcodebooks.com/2017/10/06/constraintlayout-phan-2-cac-chuc-nang-nang-cao/>