**Giảng viên hướng dẫn : Lê Huỳnh Phước**

**Sinh viên thực hiện: Bùi Quang Trung**

**Bùi Thị Trọng**

**Lâm Vũ Thành Tài**

**Trương Hoài Phong**

**NHÓM 2**



**LẬP TRÌNH ANDROID**

Contents

[**I. Drawable và ảnh Bitmap** 3](#_Toc115865995)

[1. Ảnh bimap 3](#_Toc115865996)

[**2. Tổ Chức Ảnh Bitmap Trong res/** 3](#_Toc115865997)

[**3. Sử dụng ảnh bitmap** 4](#_Toc115865998)

[4. Công Cụ Image Asset 6](#_Toc115865999)

[5. Công Cụ Android Asset Studio 9](#_Toc115866000)

[**II. Sử dụng Drawable – ảnh 9Patch & ảnh Vector** 10](#_Toc115866001)

[1. Ảnh 9-Patch 10](#_Toc115866002)

[**1.1** **Cấu tạo của ảnh 9-Patch** 11](#_Toc115866003)

[**1.2. Tổ chức ảnh 9-patch trong res/** 13](#_Toc115866004)

[**1.3. Công cụ giúp chuyển ảnh PNG sang ảnh 9-Patch:** 14](#_Toc115866005)

[2 Ảnh Vector: 15](#_Toc115866006)

[**2.1.Sử dụng Support Library:** 16](#_Toc115866007)

[**2.2.Tổ chức ảnh Vector trong res/** 17](#_Toc115866008)

[**2.4.Sử dụng ảnh vector:** 20](#_Toc115866009)

[**2.5.Truy suất từ Java code:** 21](#_Toc115866010)

[**III. ẢNH SHAPE XML và LAYER LIST** 21](#_Toc115866011)

[1. ẢNH SHAPE XML 21](#_Toc115866012)

[- Tổ Chức Ảnh Shape XML Trong res/ 25](#_Toc115866013)

[2 .ẢNH LAYER LIST 25](#_Toc115866014)

[**IV. Drawable StateList và các Drawable khac** 26](#_Toc115866015)

[1. Drawable State List 27](#_Toc115866016)

[2. Color State List 28](#_Toc115866017)

[3. Các Resource Drawable còn lại 28](#_Toc115866018)

[**3.1 Ảnh dạng Level List** 28](#_Toc115866019)

[**3.2** **Transition Drawable** 28](#_Toc115866020)

[**3.3 Ảnh Inset** 29](#_Toc115866021)

[**3.4** **Ảnh Clip** 29](#_Toc115866022)

# **I. Drawable và ảnh Bitmap**

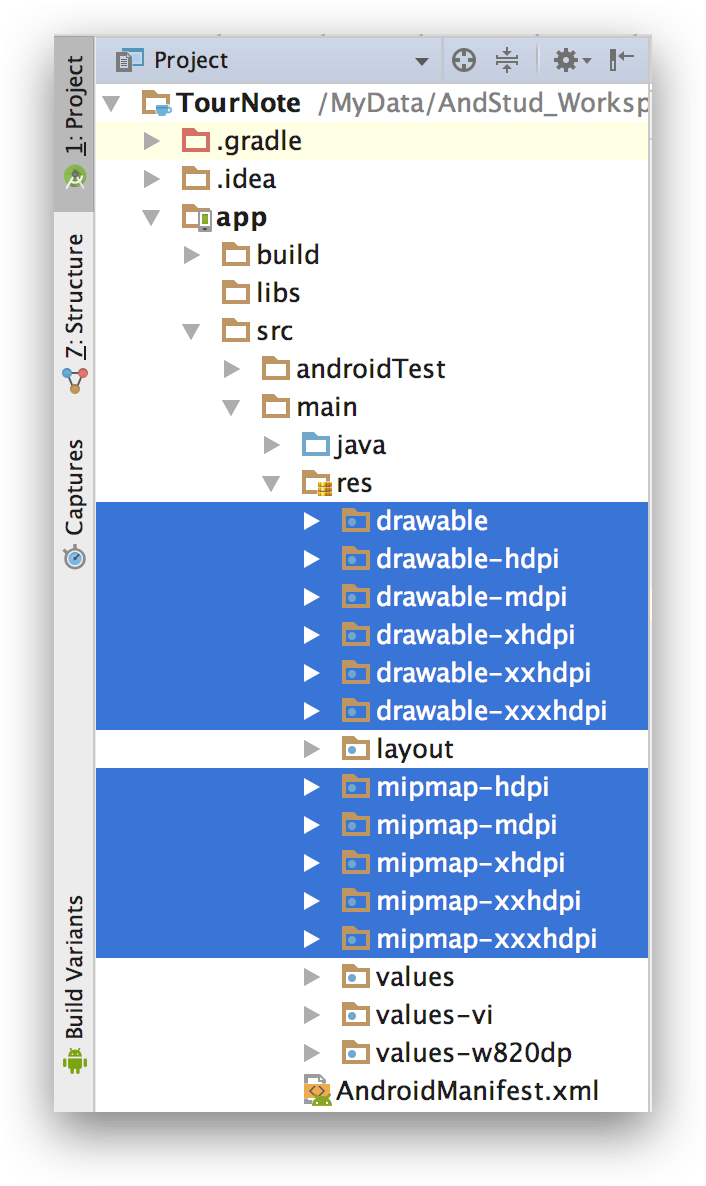
## 1. Ảnh bimap

- Bitmap là các dạng ảnh được tổ chức theo ma trận các điểm ảnh, như các ảnh PNG, JPEG, JPG, TIFF, PSD,…

Có nhiều các thể loại bitmap drawable nhưng Android chỉ hỗ trợ chúng ta sử dụng ba định dạng sau thôi, đó là PNG, JPG và GIF. Trong đó PNG được khuyến khích sử dụng, JPG thì chấp nhận được, và GIF thì nên hạn chế.

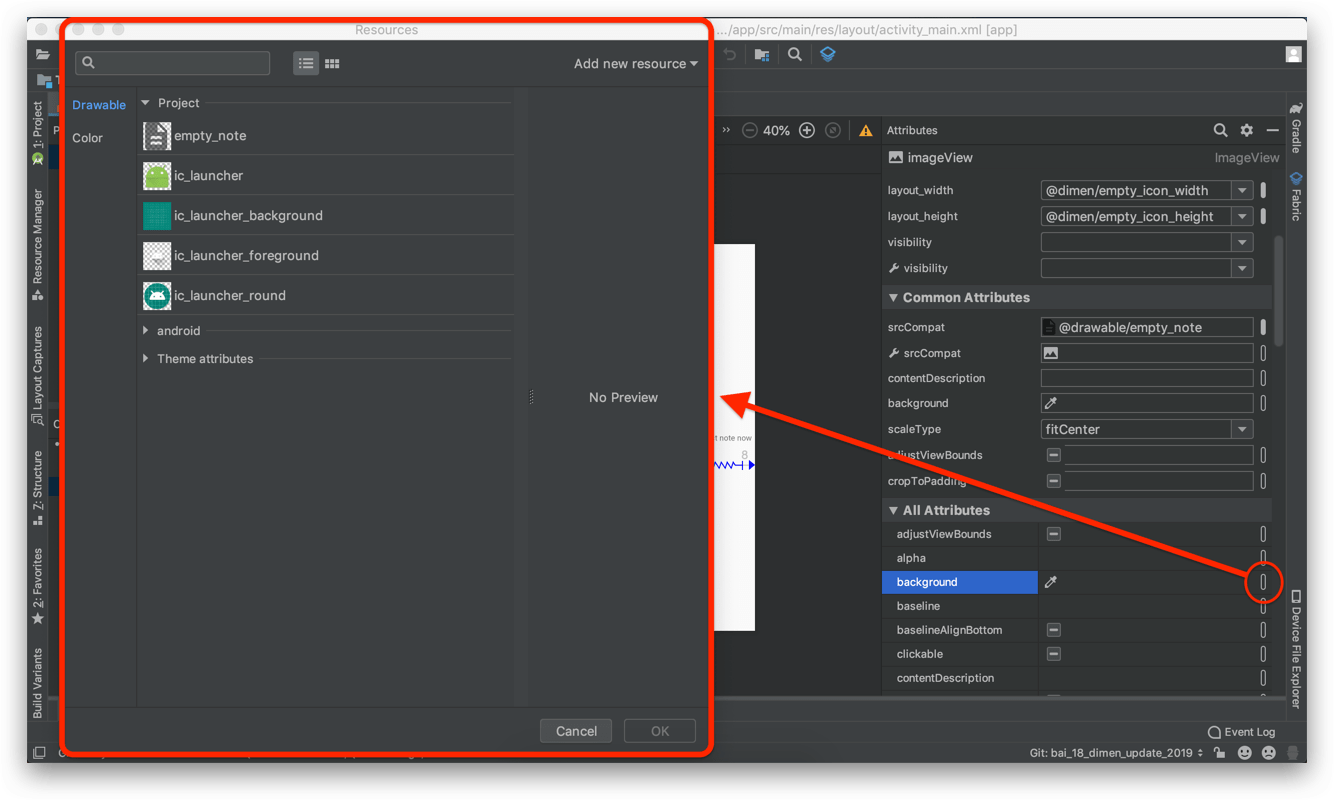
**2. Tổ Chức Ảnh Bitmap Trong res/**

- Có hai loại thư mục con của res/ mà bạn có thể dùng để tổ chức các ảnh bitmap này, đó chính là drawable-xxx/ và mipmap-xxx/. Chính là các thư mục được tô sáng như dưới đây:



**3. Sử dụng ảnh bitmap**

* Sử dụng ảnh Bitmap từ file XML: Từ bất kỳ file XML nào, như là các file layout hay file Manifest, nếu đang ở tab Design, thì ở các field liên quan đến hình ảnh, như field background ở hình dưới, có thể nhấn vào nơi được khoanh tròn trong hình để mở ra một cửa sổ liệt kê tất cả các hình ảnh, bạn chỉ việc chọn lựa hình ảnh đã khai báo sẵn mà thôi.



Còn nếu đang ở tab **Text**, bạn truy cập đến ảnh bitmap bằng cách gọi ***@drawable/tên\_file*** hoặc ***@mipmap/tên\_file*** tùy theo ảnh đó được để trong drawable hay mipmap.

Ví dụ:

Text

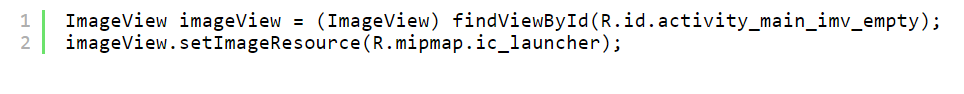
Description automatically generated

Với code trên, bạn có thể thấy rằng, khi truy xuất đến ảnh bitmap, sẽ không có thông tin về định dạng ảnh, như ***.png*** hay ***.jpg*** hay ***.gif*** gì cả. Tức là không có chuyện khai báo như này ***@drawable/empty\_note.png***, bạn chỉ cần truy xuất đến tên của file mà thôi, kiểu viết đúng luôn luôn là ***@drawable/empty\_note***

Và như vậy thì có nghĩa là, bạn sẽ không thể để hai ảnh có cùng tên nhưng khác định dạng, như ***empty\_note.png*** và ***empty\_note.jpg***, vào trong cùng một project, vì khi gọi đến chúng ở file XML, hệ thống sẽ không biết bạn đang ám chỉ loại ảnh nào.

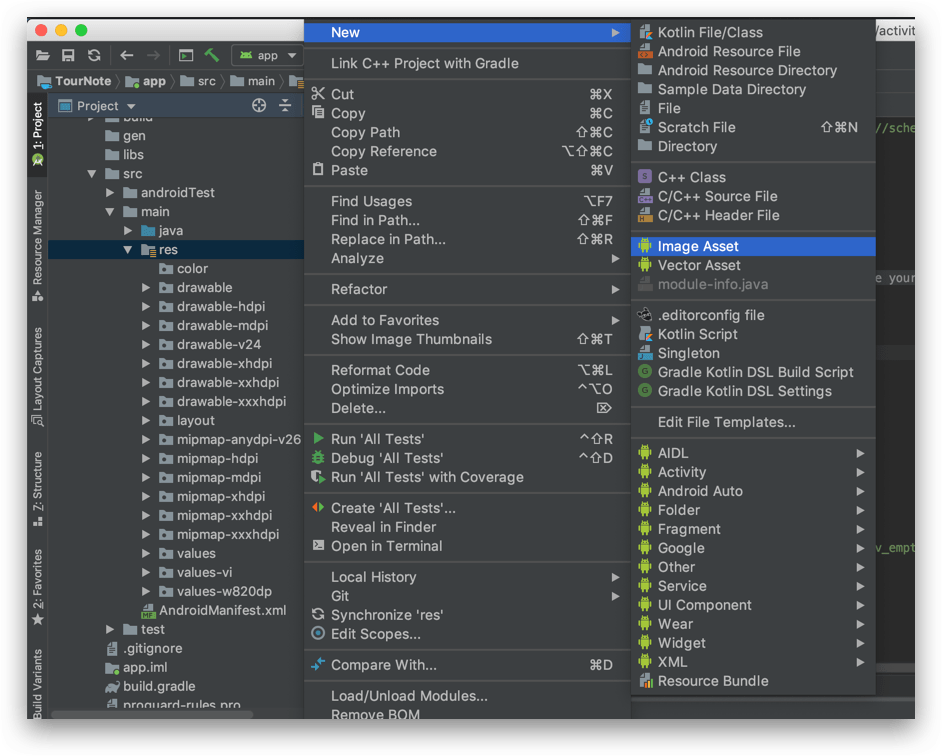
* Sử Dụng Ảnh Bitmap Từ Java Code

Từ Java code, để gọi đến ảnh bitmap này, bạn có thể truyền vào tham số ***R.drawable.tên\_file*** hoặc ***R.mipmap.tên\_file*** – Tùy theo ảnh đó được để trong drawable hay mipmap – Vào trong các hàm cần một resource ID.

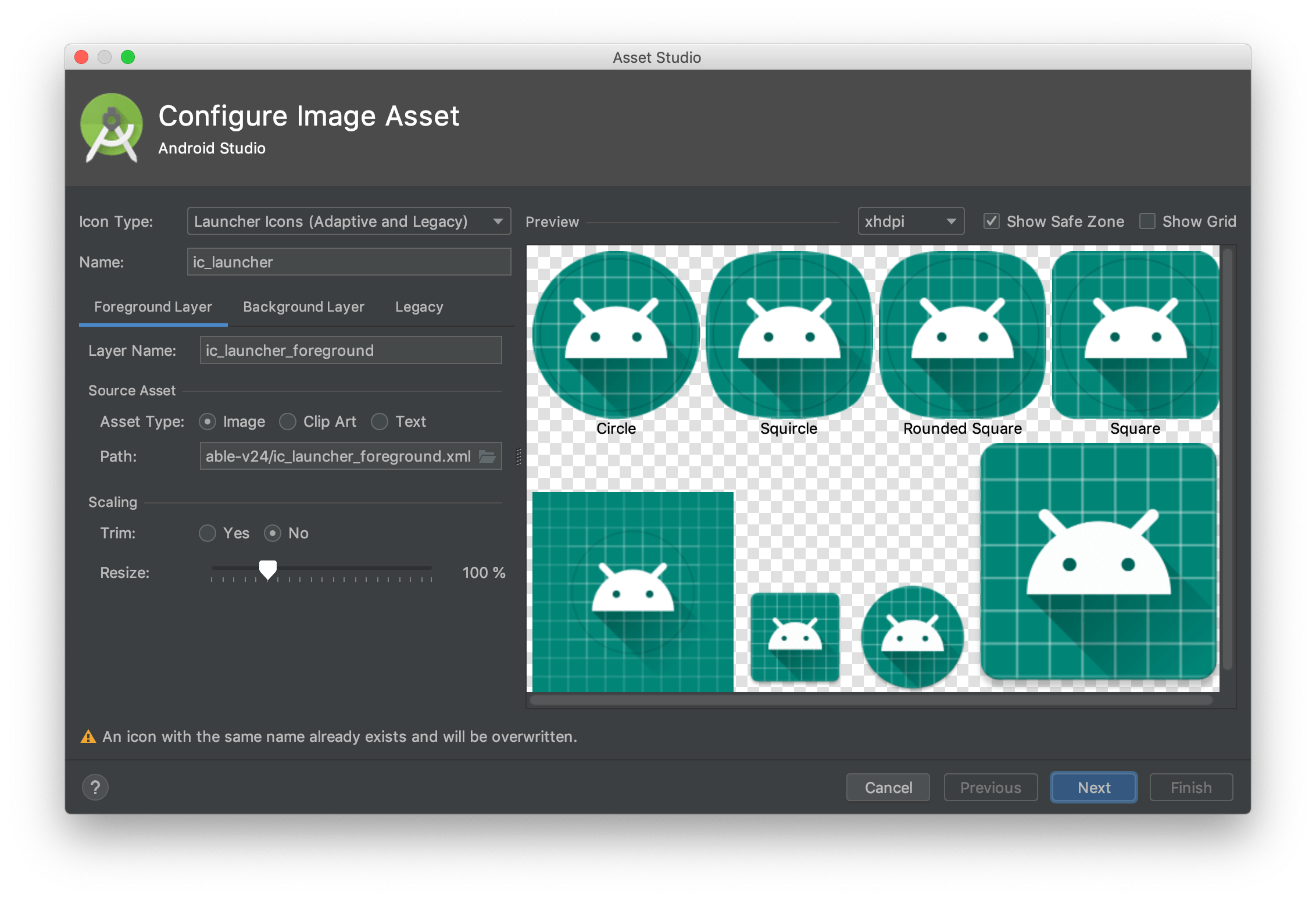
Ví dụ:

## 4. Công Cụ Image Asset

Để mở công cụ **Image Asset** thì từ Android Studio, bạn click chuột phải vào ***res/*** rồi chọn ***New > Image Asset***. Như hình.

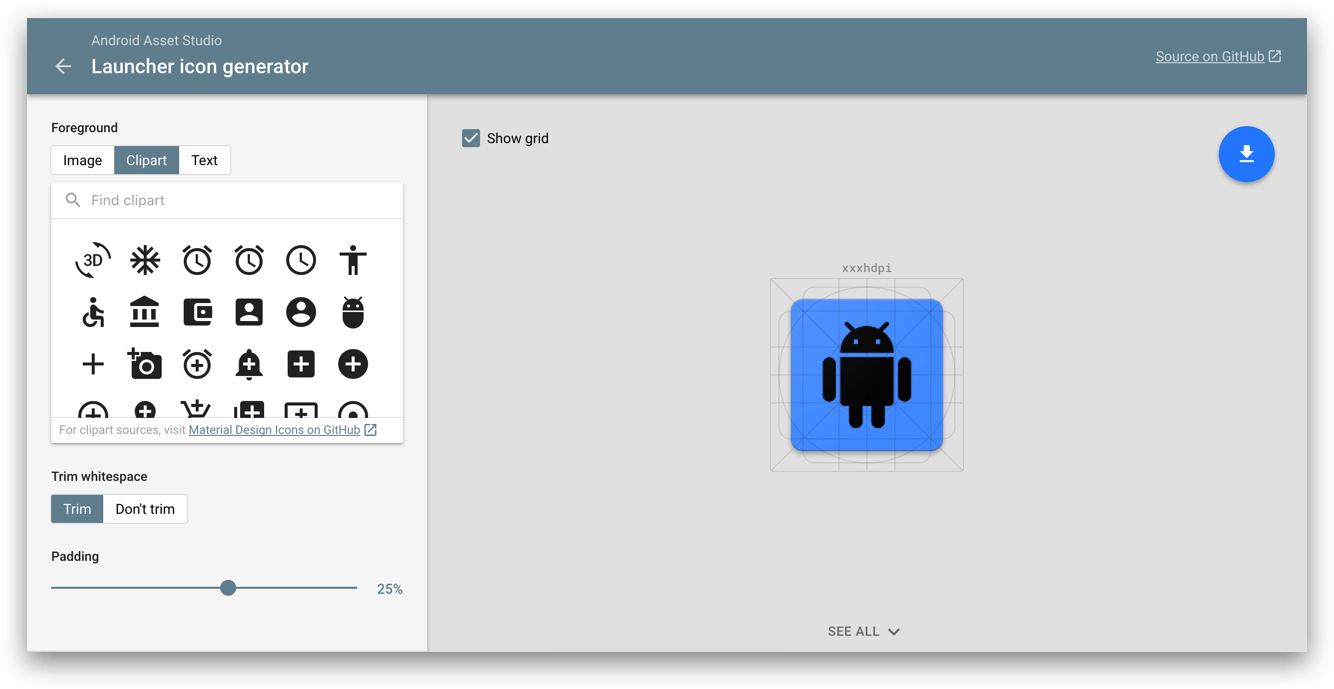


Popup xuất hiện sau đó sẽ trông như thế này.



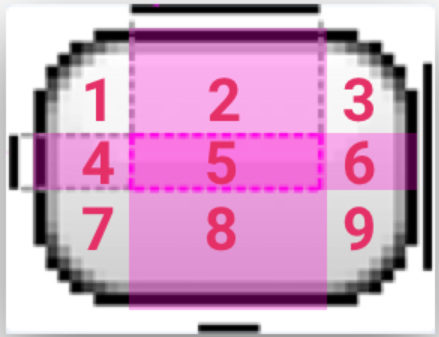
* ***Icon Type*** – Giúp bạn chọn lựa các bộ icon theo các chủ đề, các chủ đề này bao gồm.
  + ***Launcher Icons***, với tùy chọn này bạn sẽ tạo ra được bộ các icon cho app, với các kích cỡ icon ưng ý, và để sẵn vào trong thư mục ***res/mipmap-xxx/*** . Tùy chọn thứ nhất **Launcher Icons (Adaptive and Legacy)** hỗ trợ cả Adaptive icon và icon truyền thống ngày xưa trước Android 0. Tùy chọn thứ hai **Launcher Icons (Legacy only)** thì chỉ hỗ trợ bộ icon trước Android O mà thôi.
  + ***Action Bar and Tab Icons***, tùy chọn này giúp bạn tạo một bộ icon dùng cho action bar và tab, dĩ nhiên bạn cũng có thể tận dụng tùy chọn này để tạo các icon cho các thành phần khác, kết quả của tùy chọn này chính là bộ ảnh được sắp xếp *“ưng ý”* trong các thư mục ***res/drawable-xxx/*** tương ứng.
  + ***Notification Icons***, tùy chọn này giúp bạn tạo một bộ icon dùng cho notification
* ***Name*** –đặt tên cho icon.
* Nếu ở **Icon Type** mà bạn chọn **Launcher Icons (Adaptive and Legacy)** thì dưới mục **Name** sẽ có 3 tab tương ứng với 3 cài đặt cho các lớp của Adaptive icon:
  + Tab ***Foregaround Layer*** cho phép bạn chọn ảnh làm foreground. Đó có thể là một ảnh bitmap của bạn, hay ảnh từ thư viện Clip Art của Google, thậm chí là text.
  + Tab ***Background Layer*** cho phép bạn chọn ảnh làm background. Với 2 tùy chọn là màu sắc hoặc ảnh.
  + Tab ***Legacy*** cho phép bạn thiết lập các tùy chọn icon cho các hệ điều hàn Android khác nhau, thường thì mình để mặc định thông tin ở tab này.

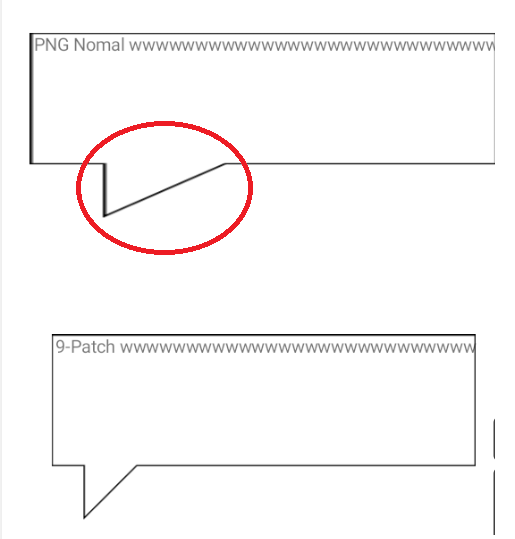
## 5. Công Cụ Android Asset Studio

Công cụ này khá tốt, nhưng nó cũng có một số khuyết điểm , chẳng hạn như các icon của nó không được nhóm lại theo từng chủ đề, bạn sẽ mất nhiều thời gian hơn mỗi lần tìm icon Clipart có sẵn. Hay sau khi chọn được icon ưng ý, bạn phải down về gói zip, tuy gói zip này có để sẵn các icon vào từng thư mục, nhưng cũng khiến bạn mất một ích thời gian để copy/paste chúng vào project, không được tự động như Image Asset trên kia.

# **II. Sử dụng Drawable – ảnh 9Patch & ảnh Vector**

## 1. Ảnh 9-Patch

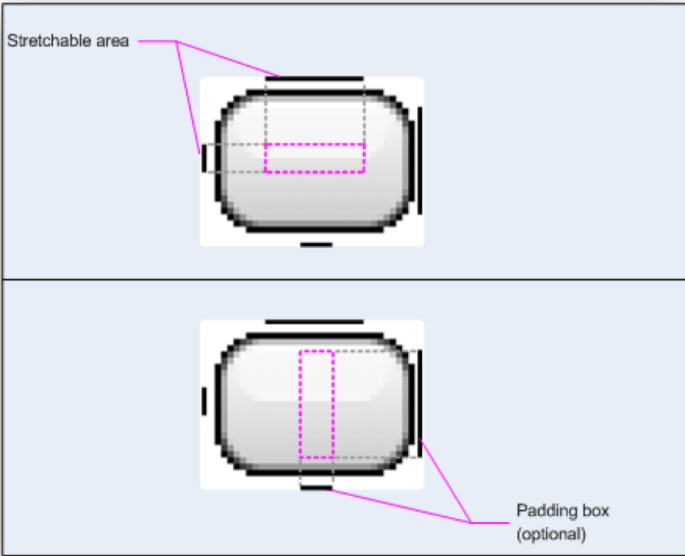


* Một dạng ảnh linh động hơn Bitmap, đúng như tên gọi ảnh được chia làm 9 miếng, là một ảnh PNG nhưng có khả năng co giãn theo chỉ định. 
* Với những ảnh PNG thông thường khi chúng ta kéo dãn thì đến một mức nào đó sẽ gây nên hiện tượng bể ảnh hoặc các chi tiết trong ảnh bị kéo dãn theo. Nhưng đối với 9-patch thì chúng ta có thể chỉ định khu vực nào của ảnh được phép co giản và khu vực nào giữ nguyên

### **1.1** **Cấu tạo của ảnh 9-Patch**

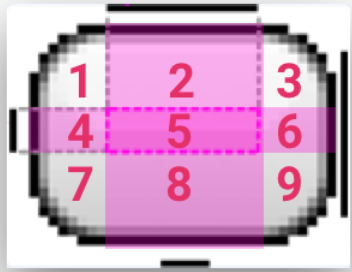
-Đầu tiên, một ảnh được gọi là ảnh 9-Patch nếu chúng được lưu trữ với đuôi là ***.9.png***. Bạn xem, ảnh này đúng là ảnh PNG, nhưng có thêm số 9 ở trước, nhìn vào phần đuôi này mà hệ thống Android sẽ hiểu bạn đang sử dụng ảnh PNG dạng 9-Patch chứ không phải PNG thường

-Để tạo ra ảnh***.9.png***, bạn (hay designer, hay Google cũng có tool cho bạn làm điều này) phải chừa ra 1 pixel cho mỗi cạnh của ảnh PNG gốc. 1 pixel này sẽ dùng để định nghĩa miếng nào có thể co giãn được, miếng nào không nên co giãn, rồi vẽ vào phần chừa ra đó một hay nhiều lằn gạch màu đen. Như hình minh họa sau của Google.

****

-**2 đường gạch đen trên cùng và bên trái** giúp điều chỉnh vùng có thể co giãn

-**2 đường gạch đen dưới cùng và bên phải** giúp điều chỉnh kích thước nội dung

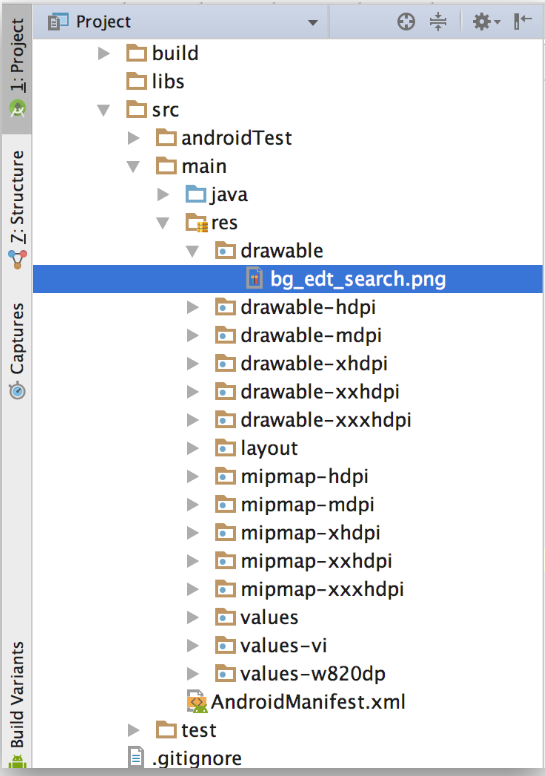
****

-Số (2,4,5,6,8) có thể được co giãn, còn số (1,3,7,9) sẽ chỉ định giữ nguyên

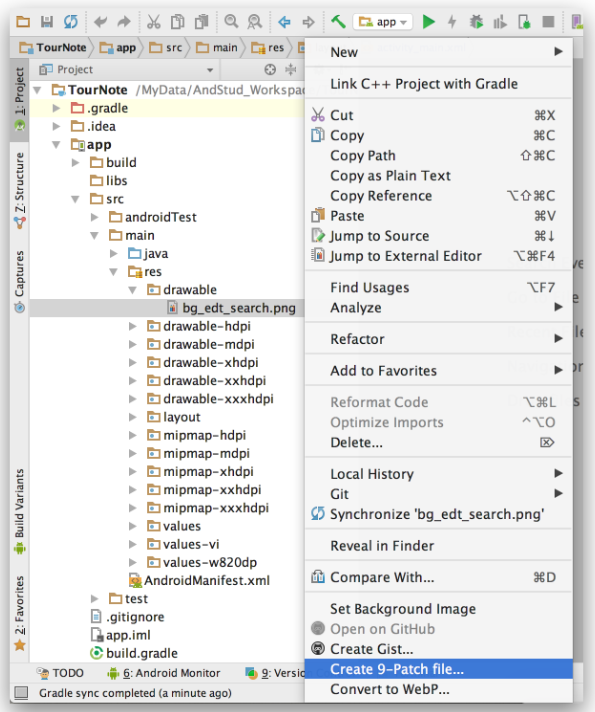
### **1.2. Tổ chức ảnh 9-patch trong res/**

-Ảnh 9-Patch thực chất cũng là một dạng ảnh trong nhóm drawable mà mình đang nói đến. Do đó giống như bên ảnh bitmap, bạn chỉ có thể để ảnh 9-Patch này vào thư mục ***res/drawable-xxx/***. Đặc biệt hơn, mình khuyên các bạn để thẳng vào ***res/drawable/*** luôn. Vì sao không cần chỉ định [alternative resource](https://yellowcodebooks.com/2016/10/05/android-bai-8-tim-hieu-android-resource/) cho chúng?

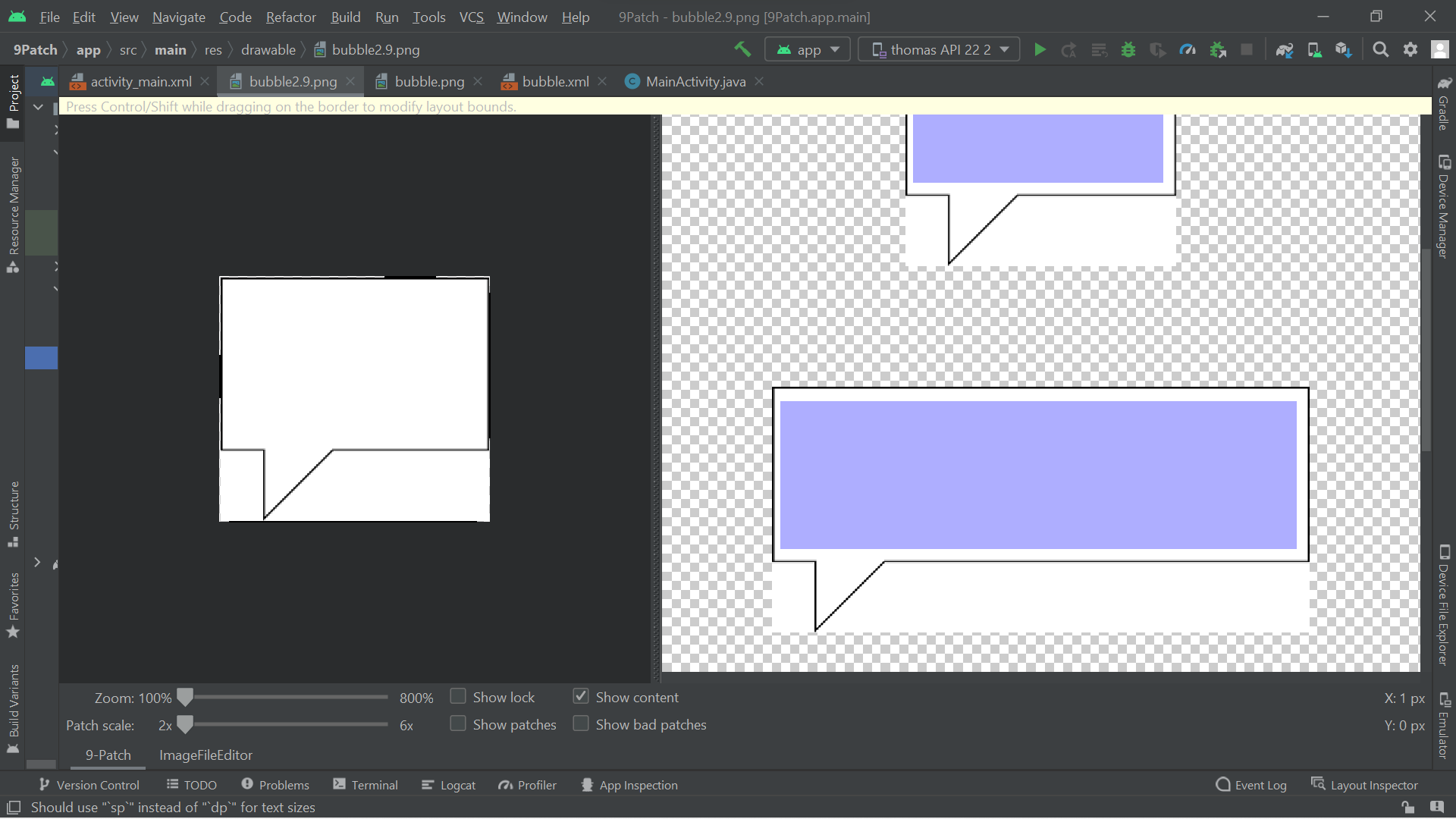
-Bởi đặc tính co giãn linh động của các ảnh 9-Patch, nên thường thì người ta chỉ cần tạo ra một ảnh, rồi chỉ định 9 miếng cho nó. Do chỉ có một ảnh thì để vào [default resource](https://yellowcodebooks.com/2016/10/05/android-bai-8-tim-hieu-android-resource/) luôn cho rồ

****

### **1.3. Công cụ giúp chuyển ảnh PNG sang ảnh 9-Patch:**

****

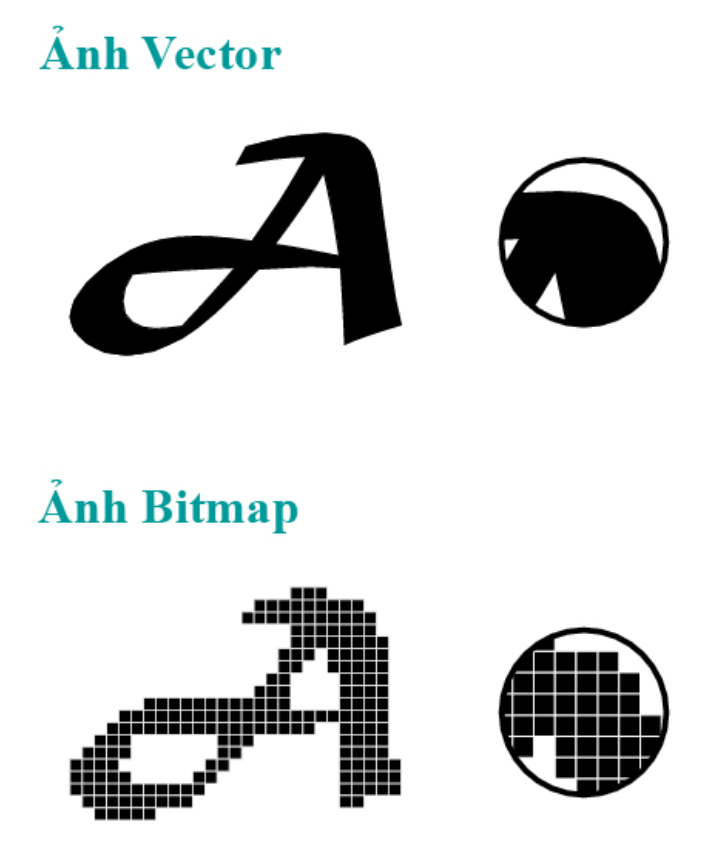
**-**Trong Android studio có hỗ trợ công cụ chuyển đổi PGN sang 9-Patch

****

-Ở đây chúng ta có thể chỉ định những vị trí sẽ co giãn cũng như nội dung vô cùng linh động và dễ dàng

**Thực hành ví dụ:**

## 2 Ảnh Vector:

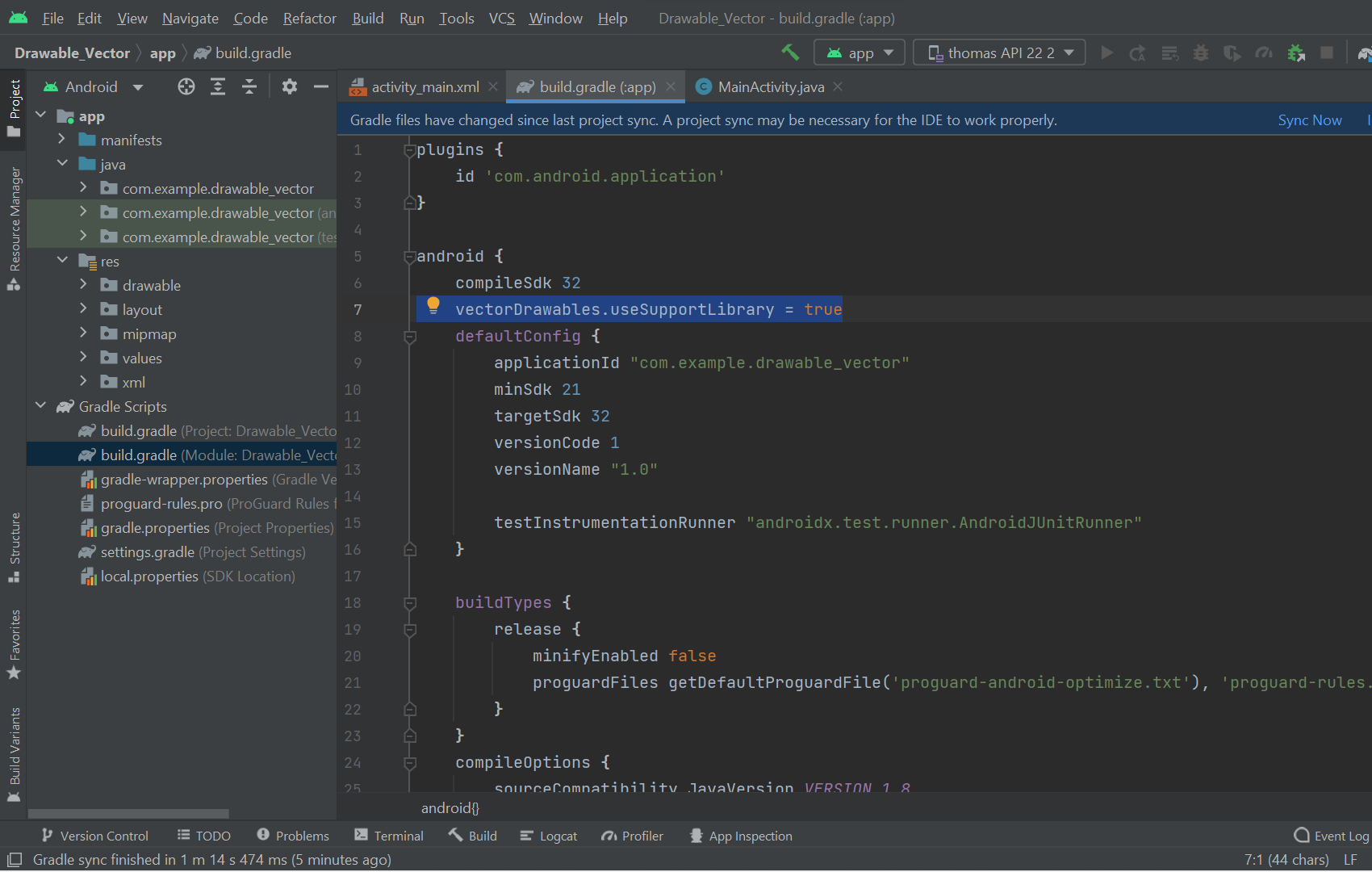
****

-Nếu như ảnh bitmap được cấu thành bởi ma trận các điểm ảnh (cố định). Bạn có thể tưởng tượng giả sử có một ảnh bitmap có kích cỡ ***100 x 100 pixel*** sẽ được biểu diễn bởi ma trận ***100 x 100*** điểm ảnh. Do đó nếu bạn muốn hiển thị ảnh này với kích cỡ lớn hơn, chẳng hạn như ***400 x 400 pixel***, thì các pixel này bị giãn nở ra, kết quả cho ta một ảnh bị vỡ. Chính vì đặc tính này của bitmap, mà bạn phải để nhiều kích thước ảnh vào trong các [alternative resource](https://yellowcodebooks.com/2016/10/05/android-bai-8-tim-hieu-android-resource/) khác nhau, để hạn chế tỉ lệ vỡ ảnh do sự co giãn này.

-Qua đến ảnh vector, các ảnh dạng này không dùng ma trận tĩnh các điểm ảnh nữa, mà chỉ lưu trữ các thông số chỉ định các đường thẳng, đường cong, màu sắc… cho mỗi ảnh. Đến khi sử dụng ảnh dạng này ở thực tế, thì hệ thống mới chính thức vẽ ảnh lên màn hình dựa vào các thông số đó. Chính vì đặc tính này mà ảnh Vector không bị vỡ dù cho được biểu diễn với bất kỳ kích thước nào.

### **2.1.Sử dụng Support Library:**

-Vấn đề này xảy ra nếu bạn có khai báo ***minSdkVersion*** trong file ***build.gradle*** nhỏ hơn API level ***21***. Đa số các project hiện nay vẫn còn hỗ trợ các hệ điều hành Android trước 5.0. Chỉ định có sử dụng Support Library trong file ***build.gradle***, thì thư viện này sẽ hỗ trợ cho bạn các hàm sử dụng hoàn toàn ảnh vector trong project

****

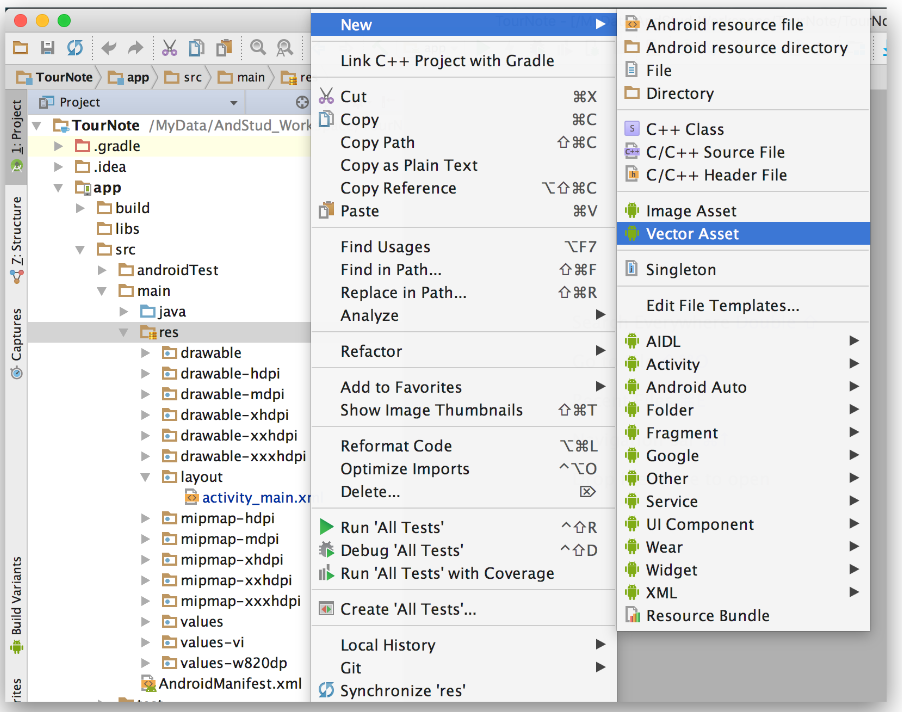
### **2.2.Tổ chức ảnh Vector trong res/**

Nên để các ảnh vector này trong thư mục ***res/drawable/***. Vì cũng giống như ảnh 9-Patch, các ảnh vector này sẽ tự co giãn khi hiển thị thực tế, nên chúng ta không cần tổ chức chúng theo [alternative resource](https://yellowcodebooks.com/2016/10/05/android-bai-8-tim-hieu-android-resource/) nữa.

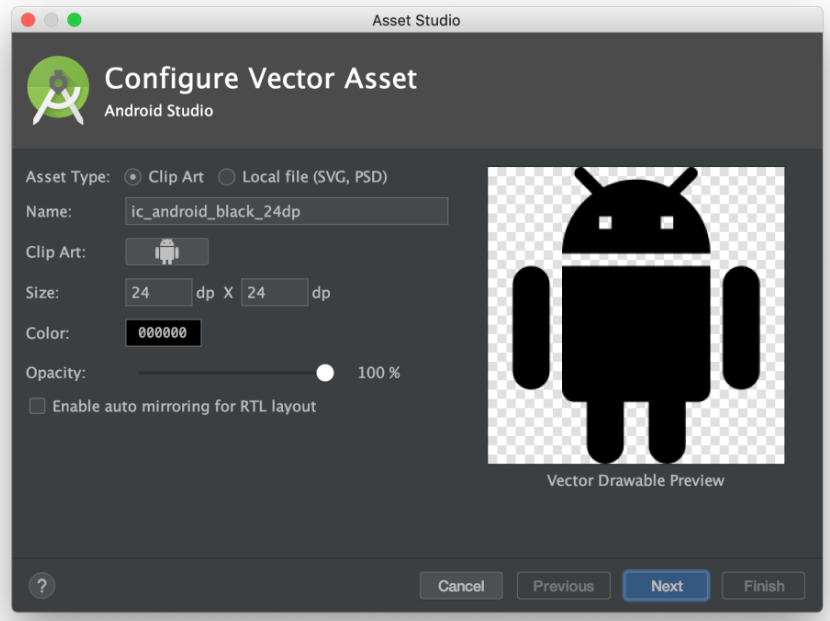
Có hai dạng ảnh vector được Android hỗ trợ, đó là ***SVG (Scalable Vector Graphic)*** và ***PSD (Adobe Photoshop Document)***.

**2.3.Công cụ Vector Asset Studio**

Để sử dụng Vector Asset, từ Android Studio, bạn click phải chuột vào thư mục ***res/*** và chọn ***New > Vector Asset***. Như hình.

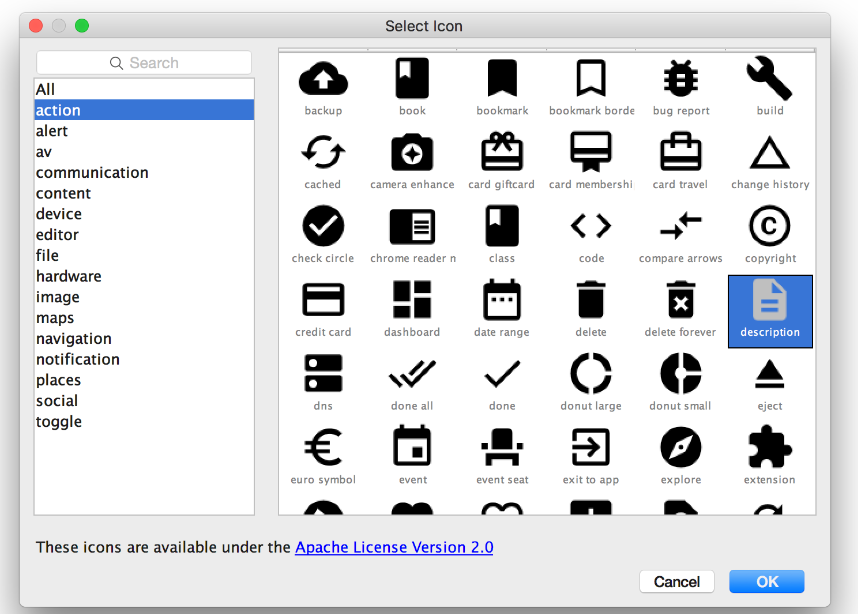


Popup xuất hiện sau đó trông như thế này.

****

* ***Asset Type*** – Giúp bạn chọn giữa bộ vector icon có sẵn của Google (nếu check ***Clip Art***), hay là từ file bạn upload lên (nếu check ***Local File***).
* ***Name*** – Chỉ là đặt tên cho icon thôi.
* ***Clip Art*** – Click vào hình con robot ở mục này sẽ dẫn bạn đến một bộ các icon có sẵn của Google. Nhưng nếu ở trên kia bạn chọn vào ***Local File*** thì field này đổi tên là ***Patch***, cho phép bạn chỉ định file vector của bạn.
* ***Size*** – Chỉ định kích thước của file, bạn yên tâm rằng với ảnh vector, bạn có thể co giãn thoải mái, nên kích thước này cũng chỉ là tương đối, nếu có ảnh hưởng chăng, thì sẽ ảnh hưởng đến cách tương thích ngược có dùng chung ảnh PNG với ảnh vector mà mình nói ở trên kia.
* ***Color*** – Thường các ảnh Vector là các ảnh đơn sắc. Nên chúng cho phép bạn chỉ định màu sắc chủ đạo của ảnh.
* ***Opacity*** – Chỉ định độ trong suốt của ảnh. Mặc định giá trị ***100%*** có nghĩa là không trong suốt.
* ***Enable auto mirroring for RTL layout*** – Check vào nếu bạn muốn ảnh hỗ trợ tự động đảo ngược cho các quốc gia sử dụng hệ chữ được viết từ phải qua trái.

Bạn click vào hình con robot ở mục ***Clip Art*** nhé, popup xuất hiện sau đó bạn hãy tìm đến và chọn hình sau đây, hoặc bất cứ hình nào bạn thích. Sau khi nhấn ***OK*** ở popup chọn hình, thì bạn hãy đặt lại tên là ***empty\_note\_vector*** và nhấn ***Next***.

****

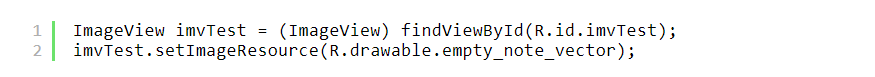
### **2.4.Sử dụng ảnh vector:**

**-Sử dụng ảnh Vector từ file.xml**

****

-Cách này sẽ không có ảnh PNG cho các hệ điều hành Android level 20 trở về trước, mà chỉ toàn các ảnh vector mà thôi. Do đó để các hệ điều hành Android cũ có thể hiểu được các ảnh vector này, bạn phải dùng đến thuộc tính ***app:srcCombat*** thay cho ***android:src*** ngày xưa. Như thế này.

### **2.5.Truy suất từ Java code:**



 Bằng cách truyền vào tham số ***R.drawable.tên\_file*** cho các hàm cần một resource ID, giống như đã từng làm với ảnh PNG bình thường vậy.

**Thực hành:**

# 

# **III. ẢNH SHAPE XML và LAYER LIST**

## 1. ẢNH SHAPE XML

- Shape XML hoặc Shape ảnh dạng này cho phép thiết kế các hình khối dựa trên XML

- Shape XML dựa trên các hình khối với Shape XML có thể code bằng tay

* ƯU ĐIỂM

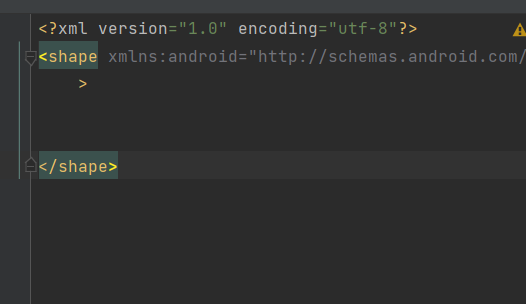
+ vẽ Shape XML không tốn CPU , giúp tốn ít không gian hệ thống lưu trữ các ảnh bitmap

* NHƯỢC ĐIỂM

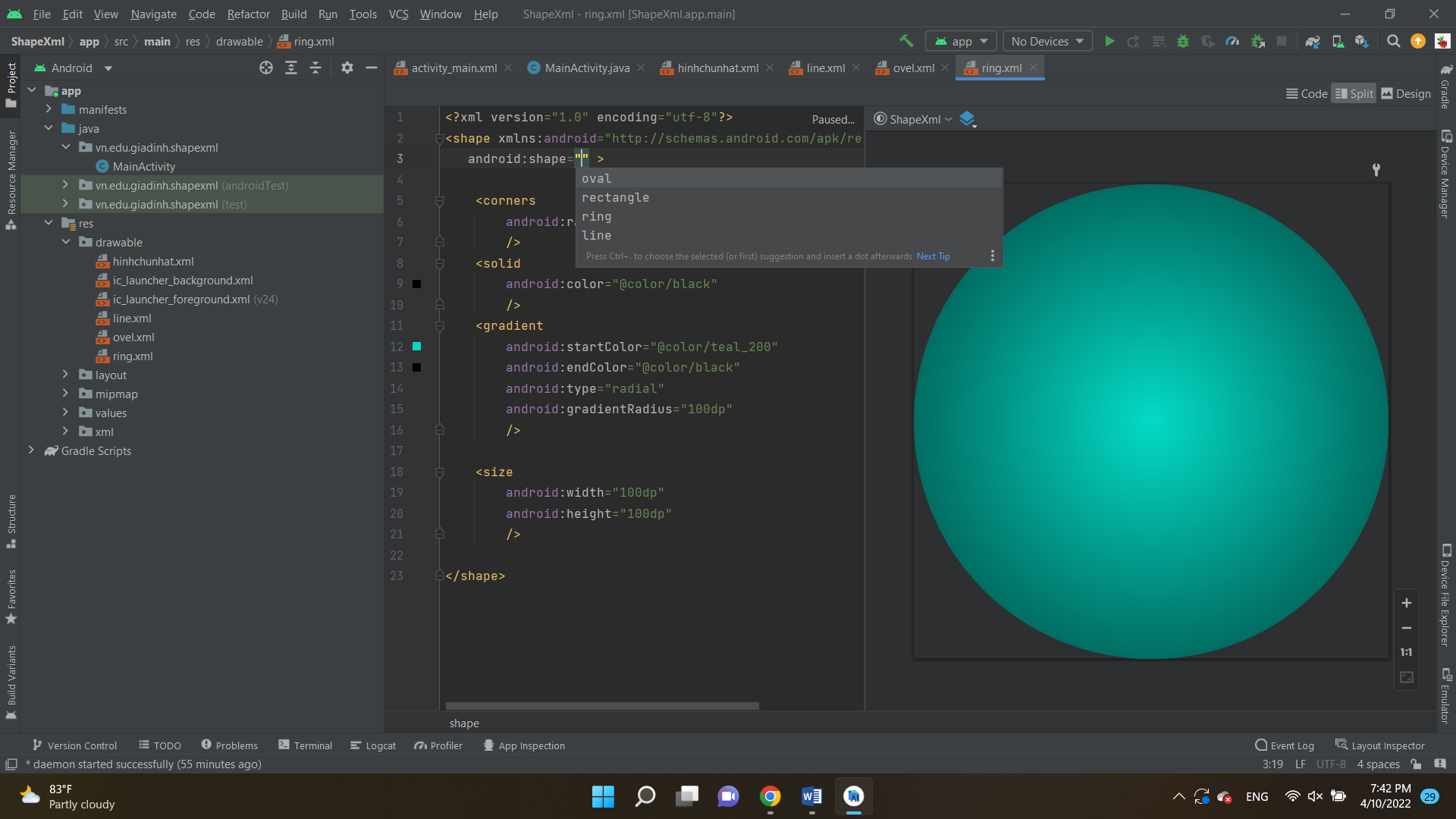
+ độ khó và độ đa dạng của nó chỉ có thể vẽ được các hình khối đơn giản như background cho button hay background cho text

+Cách Vẽ Shape

+ khai báo thẻ Shape



+ thuộc tính android:shape chỉ định hình khối muốn vẽ có tất cả là 4 loại:



Rectangle (hình chữ nhật) : hình này là mặc định , không khai báo vẫn hiện bình thường vì nó đã mặc định là hình chữ nhật

Ovel (hình bầu dục)

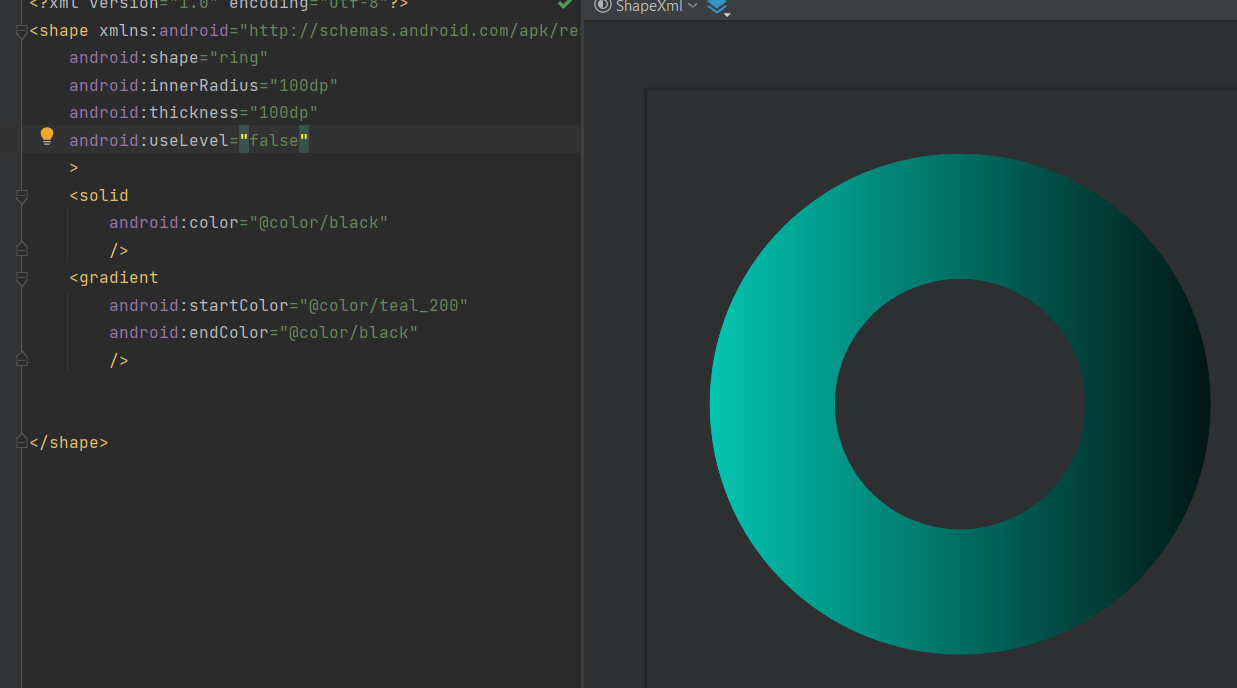
Line(đường thẳng)

Ring là một vành được xác định bởi 2 đường tròn , nếu dùng ring thì phải thêm thuộc tính

+ **android:innerRadius** hoặc **android:innerRadiusRatiođường kính đường tròn bên trong ở đây android:innerRadius** dùng độ lớn và **android:innerRadiusRatio dùng tỉ lệ**

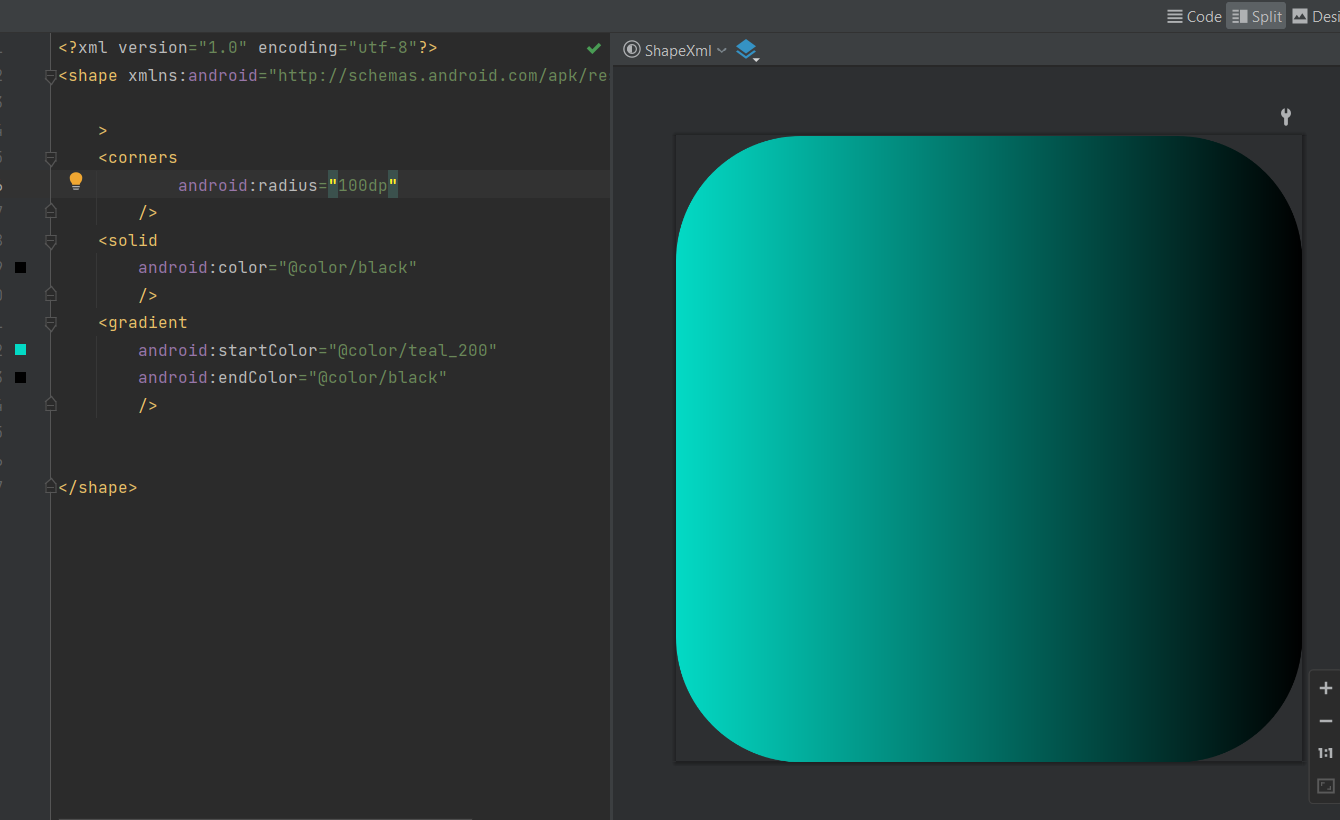
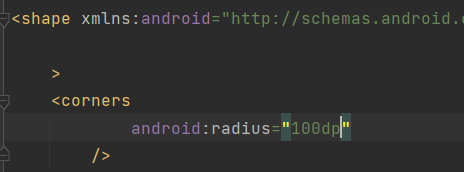
**+ android:thickness** hoặc **android:thicknessRatio** xác định khoảng cách giữa 2 đường tròn tương tự như trên **android:thickness** dùng độ lớn , **android:thicknessRatio**  dùng tỉ lệ

+ **android:useLevel**  cho biết ring có dùng drawable dạng LevelListDrawable hay không

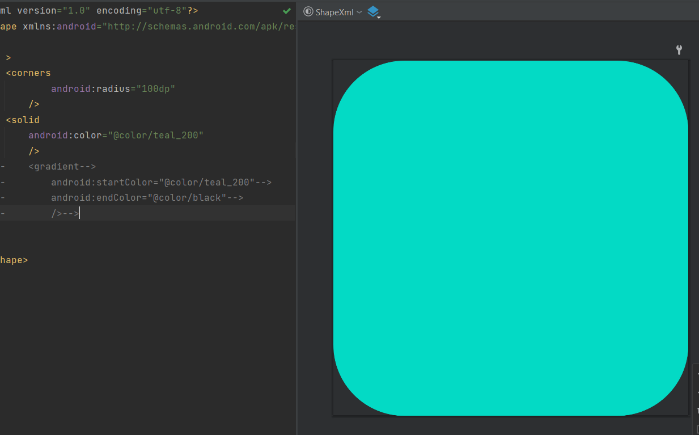


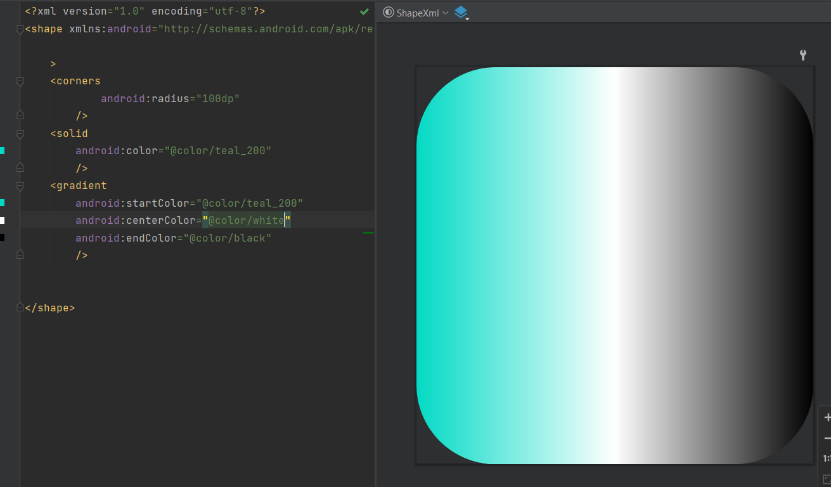
Trong thẻ Shape có các loại thẻ con:

+ corners : dùng để bo tròn hình khối chữ nhật bằng radius

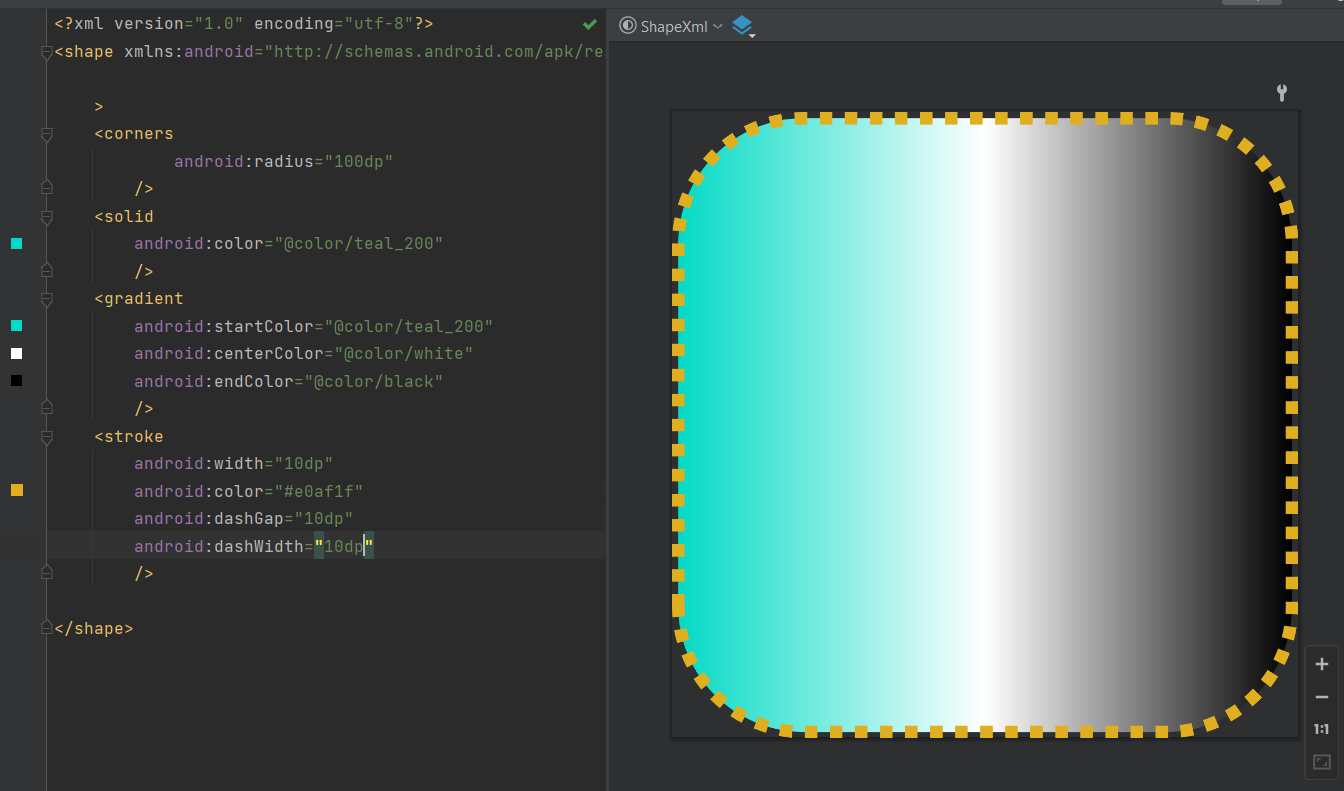
+ solid : tô màu background theo dạng một màu duy nhất



****** + gradient: tô màu background theo dạng trộn 3 điểm màu startColor , CenterColor, EndColor, trong gradient có thẻ chọn hình dạng mà màu có thể hiển thị bằng type : linear – tuyến tính , radial – kiểu ra da , sweep – kiểu kẻ quạt , ***android:gradientRadius : xác định bán kính hình tròn rada***

**+ size tùy chỉnh độ lớn của khối**

**+ stroke chỉnh độ rộng của đường biên thông qua thuộc tính width , color chỉnh màu đường biên, để tạo đường đứt nét bằng thuộc tính dashGap khoảng cách 2 nét và dashWidth độ dài giữa 2 nét**



## Tổ Chức Ảnh Shape XML Trong res/

Tương tự như ảnh ***9-Patch*** và ảnh ***Vector***  ảnh ***Shape XML*** hôm nay vẫn không cần thiết phải tổ chức theo thư mục [alternative resource](https://yellowcodebooks.com/2016/10/05/android-bai-8-tim-hieu-android-resource/), vậy bạn chỉ cần đặt tất cả các ***Shape XML*** này vào một thư mục duy nhất là ***res/drawable/*** là được.

## 2 .ẢNH LAYER LIST

LayerList giúp kết hợp các hình khối đơn giản thành một hình khối phức tạp đa dạng hơn

+ mỗi ảnh shape được đặt trong thẻ item thẻ item này nằm trong LayerList.



+ những thuộc tính của thẻ item bao gồm:

* + Drawable giúp chỉ định một resource drawable khác thay cho việc dùng thẻ ***shape***
  + **Id tạo id cho item hình được chỉ định**
  + **Top, bot , left, right : khoảng cách so với đường biên**
  + **trong rotate có thuộc tính fromDegrees có tác dụng xoay hình**
  + **ví ta thực hiện vẽ 1 hình trái tim bằng cách ta khai báo 3 item tương ứng với 3 hình : 1 hình vuông , 2 hình oval**

**+ hình vuông ta dùng rotate với tọa độ là 45**

# **IV. Drawable StateList và các Drawable khac**

* State List chính là chìa khóa để bạn tạo ra các Trạng thái khác nhau cho các widget (Widget là một thuật ngữ chung để chỉ một ứng dụng đặc biệt, có thể hiển thị toàn bộ hoặc một phần nội dung của các ứng dụng khác trên màn hình chính của thiết bị. Nói cách khác, Widget là một phím tắt để mở nhanh một ứng dụng khác. Không chỉ trên điện thoại, hàng ngày chúng ta có thể thấy các Widget này trên các trang web, [máy tính để bàn](https://www.thegioididong.com/may-tinh-de-ban), [máy tính bảng](https://www.thegioididong.com/may-tinh-bang))đó bằng cách định nghĩa vào một resource drawable.
* Có hai dạng State List mà bạn phải làm quen, đó là Drawable State List và Color State List.

## 1. Drawable State List

* Thẻ gốc của Drawable State list là thẻ **selector** có tác dụng chính là chứa các trạng thái của Widget bên trong nó
* Mỗi thẻ **item** chứa đựng các trạng thái như:

+ **android:state\_pressed** - định nghĩa trạng thái khi widget đó có được nhấn (touch) hay click không.

+ **android:state\_focused** - định nghĩa trạng thái khi widget đó có được focus vào không, như là với EditText ấy, khi user click vào EditText, thì khi đó trạng thái focus được kích hoạt.

+ **android:state\_hovered** - định nghĩa trạng thái khi con trỏ chuột có “lướt” qua Widget hay không, dùng cho một số thiết bị Android có điều khiển bằng trỏ chuột, trạng thái này nhiều khi được hành xử giống với state\_focused.

+ **android:state\_selected** - định nghĩa trạng thái khi dùng với cần gạt hay các console điều khiển. Khi các widget này được chọn đến sẽ được hệ thống thực thi trạng thái này.

+ **android:state\_checkable** - định nghĩa trạng thái khi mà widget đó được set là cho phép check hay không.

+ **android:state\_checked** - định nghĩa trạng thái check hay uncheck của widget, vừa được nhắc đến bên trên.

+ **android:state\_enabled** - tương tự như state\_checkable, state\_enabled này dùng để biểu diễn một widget có đang cho phép tương tác hay không.

+ **android:state\_activated** - định nghĩa trạng thái widget đó đang được kích hoạt hay không.

+ **android:state\_window\_focused** - định nghĩa trạng thái khi mà cửa sổ chứa đựng widget này có được focus hay không. Bạn nên biết là dù cho cửa sổ nào đó đang được hiển thị, thì không có nghĩa là nó đang được focus. Chẳng hạn như khi user đang tương tác với một cửa sổ, mà một notification show lên, sẽ làm cửa sổ đó đi vào trạng thái không được focus dù là vẫn đang được nhìn thấy bởi user.

+ **android:drawable** - đây không phải là một trạng thái, như mình có nhắc đến ở gạch đầu dòng trên kia, đây là nơi bạn khai báo resource drawable tương ứng cho mỗi item.

## 2. Color State List

* Color State List khá giống với Drawable State List về cách sử dụng cũng như cách tổ chức. Chỉ khác nhau ở chỗ một thằng thì dùng resource color, còn một thằng thì dùng resource drawable để biểu diễn các trạng thái. Khác nhau nữa đó là cách tổ chức chúng trong các thư mục res.
* Khác với Drawable State List ở trên, Color State List phải được tổ chức trong thư mục res/color/.

## 3. Các Resource Drawable còn lại

### **3.1 Ảnh dạng Level List**

- Ảnh Level List giúp chỉ định các level cho từng item.

- Thích hợp để biểu diễn cần đến các cấp độ.

- <level-list> : là phần tử gốc chứa một hoặc nhiều phần thử <item>.

- android:drawable – tài nguyên có thể vẽ (bắt buộc) ->tham chiếu đến 1 tài nguyên có thể lòng ghép

- android:maxlevel - cấp tối đa được cho phép sử dụng (số nguyên)

- android:minlevel - cấp tối thiểu được cho phép sử dụng (số nguyên)

### **3.2** **Transition Drawable**

**-** Là một đối tượng vẽ được có thể chuyển đổi ở dạng mờ dần giữa hai tài nguyên có thể vẽ.

**- <transition>** : phần tử gốc chứa 1 hoặc nhiều item

- android:drawable – tài nguyên có thể vẽ (bắt buộc) -tham chiếu đến 1 tài nguyên có thể lòng ghép

- android:id – mã nhậnd ạng tài nguyên

- android:top – độ lệch trên cùng tình là pixel (số nguyên)

- android:right – độ lệch bên phải

- android:left – độ lệch bên trái

- android:bottm – độ lệch dưới cùng

### **3.3 Ảnh Inset**

- Một đối tượng có thể vẽ được xác định trong tệp XML sẽ lồng ghép một đối tượng có thể vẽ khác theo khoảng cách đã chỉ định. Điều này rất hữu ích khi Chế độ xem cần nền nhỏ hơn giới hạn thực tế của Chế độ xem.

- <inset> phần tử gốc, xác định phần lòng ghép các đối tượng

- android:drawable tài nguyên vẽ tham chiếu đến một tài nguyên có thể vẽ để lồng

- android:insetTop phần lòng ghép trên cùng

- android:insetRight phần lòng ghép bên phải

- android:insetBottom phần lòng ghép dưới cùng

- android:insetLeft phần lồng ghép bên trái

### **3.4** **Ảnh Clip**

**-** Một đối tượng có thể vẽ được xác định trong tệp XML sẽ cắt bớt một đối tượng có thể vẽ khác dựa trên cấp độ hiện tại của đối tượng có thể vẽ này. Bạn có thể kiểm soát chiều rộng và chiều cao của đối tượng có thể vẽ được cắt bớt dựa trên cấp độ, cũng như trọng lực để kiểm soát vị trí đặt đối tượng trong vùng chứa tổng thể của đối tượng đó. Thường được dùng nhất để triển khai những nội dung như thanh tiến trình.

**-** <clip> phần tử gốc, xác định việc cắt bớt đối tượng có thể vẽ

**-** android:drawable tài nguyên có thể vẽ tham chiếu đến một tài nguyên có thể vẽ sẽ được cắt bớt

**-** android:clipOrientation hướng cắt, có 2 giá trị là horizontal và verical.

**-** android:gravity chỉ định vị trí cắt trong đối tượng có thể vẽ, có thể là 1 hoặc nhiều được phân tác bằng dấu |

|  |  |
| --- | --- |
| **Giá trị** | **Mô tả** |
| top | Đặt đối tượng lên đầu vùng chứa, không thay đổi kích thước của đối tượng. |
| bottom | Đặt đối tượng ở cuối vùng chứa, không thay đổi kích thước của đối tượng. |
| left | Đặt đối tượng ở cạnh bên trái của vùng chứa, không thay đổi kích thước của đối tượng. Đây là lựa chọn mặc định. |
| right | Đặt đối tượng ở cạnh bên phải của vùng chứa, không thay đổi kích thước của đối tượng. |
| center\_vertical | Đặt đối tượng ở giữa vùng chứa theo chiều dọc, không thay đổi kích thước của đối tượng. |
| fill\_vertical | Tăng kích thước của đối tượng theo chiều dọc nếu cần để đối tượng lấp đầy hoàn toàn vùng chứa. |
| center\_horizontal | Đặt đối tượng ở giữa vùng chứa theo chiều ngang, không thay đổi kích thước của đối tượng. |
| fill\_horizontal | Tăng kích thước của đối tượng theo chiều ngang nếu cần để đối tượng lấp đầy hoàn toàn vùng chứa. |
| center | Đặt đối tượng vào giữa vùng chứa của nó ở cả trục dọc và trục ngang, không thay đổi kích thước của đối tượng. |
| fill | Tăng kích thước của đối tượng theo chiều dọc và chiều ngang nếu cần để đối tượng lấp đầy hoàn toàn vùng chứa. |
| clip\_vertical | Bạn có thể đặt tuỳ chọn bổ sung để các cạnh trên và/hoặc dưới cùng của đối tượng con được cắt theo giới hạn của vùng chứa. Việc cắt dựa trên trọng lực dọc: trọng lực đỉnh cắt cạnh dưới, trọng lực đáy cắt cạnh trên, và không cắt cả hai cạnh. |