

LINK 3 continued..

Create stretchable nine-patch bitmaps

- Problem:

If you use a bitmap as the background in a view that changes size, you will notice Android scales your images as the view grows or shrinks based on the size of the screen or content in the view. This often leads to visible blurring or other scaling artifacts.

- Solution:

Using nine-patch bitmaps, which are specially formatted PNG files that indicate which areas can and cannot be stretched.

- A **nine-patch bitmap** is basically a standard PNG file, but with an extra 1px border that indicates which pixels should be stretched (and with a `.9.png` extension instead of just `.png`). As shown in figure 5, the intersection between the black lines on the left and top edge is the area of the bitmap that can be stretched.
- Optionally, you can also define the safe region where content should go inside the view by similarly adding lines on the right and bottom edges.
- When you apply a nine-patch as the background to a view, the framework stretches the image correctly to accommodate the size of the button.

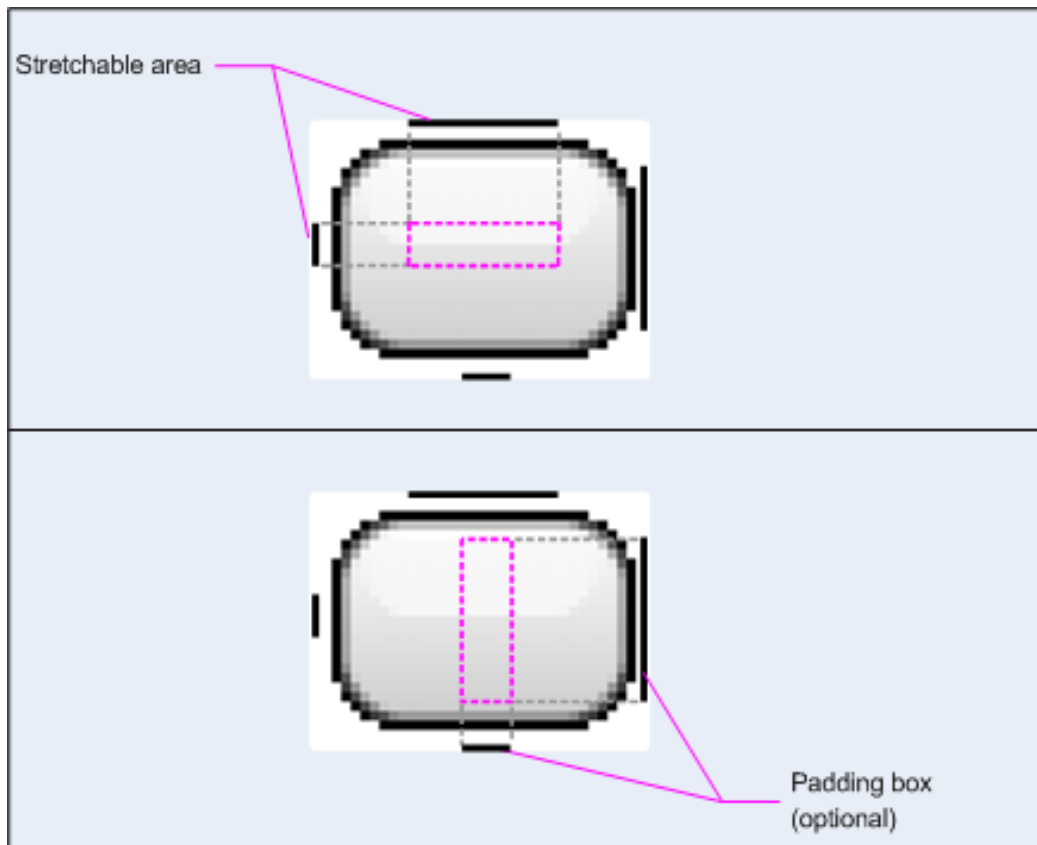


Figure 5. A nine-patch image (`button.9.png`)

Refer →

<https://www.youtube.com/watch?v=qd2dDUZ--bY>

-----Link 3 Finished-----

USER INTERFACE DESIGN

Chapter 3

[Android Application Development for Java Programmers;
By James C. Sheusi]

APPLICATION DESIGN

- Model-View-Controller (MVC) is a graphical user interface (GUI) application that allows the developer to consider three main areas when developing an application.

These areas can be kept as independent as possible.

1. The model, loosely analogous to the process in the traditional Input-Process-Output (IPO) model, represents what the application does and the coding behind what it is intended to do.
 2. The view, analogous to the output in the IPO model, is concerned with rendering the results on the display.
 3. The controller, analogous to the input in the IPO model, deals with how the user will interact with the application, including mouse movements, button clicks, and so on.
- When developing Android applications, we can easily isolate the view from the model and controller components. The view or layout of components is written in XML format in the main.xml file.
 - Once the programmer determines which controls are necessary for the application, such as lists, text fields, buttons, and so on, he can plan and code their arrangement, size, labels, fonts, and colors in the main.xml file.
 - Indeed, after the application is written, should the programmer decide he does not like the user interface's aesthetics, he can make changes without altering the Java classes and methods that produce the model and controller components of the MVC model.

For instance, Would two input fields work better side by side, or one atop the other?

The programmer can try both arrangements by altering the main.xml file only without having to find their references in Java code and making changes there.

Table 3.1 offers a quick refresher on the rules and syntax of XML files.

<code><some_element></some_element></code> or <code><some_element /></code>	Empty element
<code><some_element></code>	Opening tag
<code></some_element></code>	Closing tag
<code>some_attribute = "some value"</code>	Attribute
<code>some_attribute</code>	Name of the attribute
<code>"some value"</code>	Value of the attribute
<code><!-- comment --></code>	Comment

Following are the rules for XML files:

1. All XML elements must have opening and closing tags.
2. XML tags are case sensitive.
3. XML elements must be properly nested.

ex.

```
<tag1>  
<tag2>  
</tag2>  
</tag1>
```

4. XML documents must have a root element. A single tag pair must surround all other elements of the document. This is the root element.
5. XML attribute values must be quoted using either single or double quotes.

For any doubt contact 9873961590