**9-Patch Images**

INTRODUCTION

* What is a 9 Patch Image?

 A 9 patch image is an image that has stretchable areas defined.These areas are defined in a special 1 pixel wide border around the area.Add black pixels to the top and left of this border to describe where the image can be stretched horizontally and vertically.

* Why Use 9 Patch Images?

 Nine patch images are especially useful when designing buttons. Custom drawn buttons can look distorted and pixelated when their bordersare stretched in addition to the rest ofthe image.Take a look at the Custom Drawn Button tutorial to see how using 9 Patch images

can be used to create a button, and what happens if they aren't used.

* Drawing the Patches

 You can think of the Draw 9 Patch program as a simple image editor,but you can only change the 1 pixel border and you can only use the color black.Draw along the top of the image to define the area that can be stretched horizontally. Draw along the left side of the image to define the area that can be stretched vertically. Draw along the bottom and the right of the image to define the padding. Holding down shift and clicking will erase pixels.

* .9.png File Extension

 Your nine patch images must be saved with the .9.png file extension in order for Android to recognize and handle them properly. Failure to do so may result in unpredictable results.When you save from the Draw 9-Patch program it will have the proper extension.

* Here's a quick guide to create a Nine-patch graphic using the Draw 9-patch tool.
1. From a terminal, launch the draw9patch application from your SDK /tools directory.
2. Drag your PNG image into the Draw 9-patch window (or **File** > **Open 9-patch...** to locate the file). Your workspace will now open.The left pane is your drawing area, in which you can edit the lines for the stretchable patches and content area. The right pane is the preview area, where you can preview your graphic when stretched.
3. Click within the 1-pixel perimeter to draw the lines that define the stretchable patches and (optional) content area. Right-click (or hold Shift and click, on Mac) to erase previously drawn lines.
4. When done, select **File** > **Save 9-patch...**

 **Your image will be saved with the .9.png file name.**

**Here is the image of Draw 9-patch**

 **(1)**

****

**(2)**

****

**Note:** A normal PNG file (\*.png) will be loaded with an empty one-pixel border added around the image, in which you can draw the stretchable patches and content area. A previously saved 9-patch file (\*.9.png) will be loaded as-is, with no drawing area added, because it already exists.

* Optional controls include:
* **Zoom**: Adjust the zoom level of the graphic in the drawing area.
* **Patch scale**: Adjust the scale of the images in the preview area.
* **Show lock**: Visualize the non-drawable area of the graphic on mouse-over.
* **Show patches**: Preview the stretchable patches in the drawing area (pink is a stretchable patch).
* **Show content**: Highlight the content area in the preview images (purple is the area in which content is allowed).
* **Show bad patches**: Adds a red border around patch areas that may produce artifacts in the graphic when stretched.
* **Nine-patch :**
	+ A NinePatchDrawable graphic is a stretchable bitmap image, which Android will automatically resize to accommodate the contents of the View in which you have placed it as the background. An example use of a NinePatch is the backgrounds used by standard Android buttons — buttons must stretch to accommodate strings of various lengths. A NinePatch drawable is a standard PNG image that includes an extra 1-pixel-wide border. It must be saved with the extension .9.png, and saved into the res/drawable/ directory of your project.
	+ The border is used to define the stretchable and static areas of the image. You indicate a stretchable section by drawing one (or more) 1-pixel-wide black line(s) in the left and top part of the border (the other border pixels should be fully transparent or white). You can have as many stretchable sections as you want: their relative size stays the same, so the largest sections always remain the largest.
	+ You can also define an optional drawable section of the image (effectively, the padding lines) by drawing a line on the right and bottom lines. If a View object sets the NinePatch as its background and then specifies the View's text, it will stretch itself so that all the text fits inside only the area designated by the right and bottom lines (if included). If the padding lines are not included, Android uses the left and top lines to define this drawable area.
	+ To clarify the difference between the different lines, the left and top lines define which pixels of the image are allowed to be replicated in order to stretch the image. The bottom and right lines define the relative area within the image that the contents of the View are allowed to lie within.

 **Here is a NinePatch file used to define a button:**



* This NinePatch defines one stretchable area with the left and top lines and the drawable area with the bottom and right lines. In the top image, the dotted grey lines identify the regions of the image that will be replicated in order to stretch the image. The pink rectangle in the bottom image identifies the region in which the contents of the View are allowed. If the contents don't fit in this region, then the image will be stretched so that they do.
* The Draw 9-patch tool offers an extremely handy way to create your NinePatch images, using a WYSIWYG graphics editor. It even raises warnings if the region you've defined for the stretchable area is at risk of producing drawing artifacts as a result of the pixel replication.

### Example XML :

* Here's some sample layout XML that demonstrates how to add a NinePatch image to a couple of buttons.

 **Code:**

<Button id="@+id/tiny"
        android:layout\_width="wrap\_content"
        android:layout\_height="wrap\_content"
        android:layout\_alignParentTop="true"
        android:layout\_centerInParent="true"
        android:text="Tiny"
        android:textSize="8sp"
        android:background="@drawable/my\_button\_background"/>

<Button id="@+id/big"
        android:layout\_width="wrap\_content"
        android:layout\_height="wrap\_content"
        android:layout\_alignParentBottom="true"
        android:layout\_centerInParent="true"
        android:text="Biiiiiiig text!"
        android:textSize="30sp"
        android:background="@drawable/my\_button\_background"/>

**Output:**

 ****