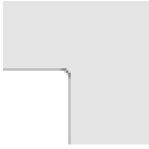
NSBezierPath drawing

Asked 12 years ago Modified 2 years, 8 months ago Viewed 5k times

14

I want to do a rounded rectangle outline on an NSImage and I figured that using NSBezierPath would be the best way. However, I ran into a problem: instead of drawing a nice curve, I get this:



For reasons I can't understand, NSBezierPath is drawing the rounded part with a darker color than the rest.

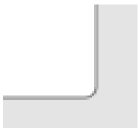
Here's the code I'm using (inside a drawRect: call on a custom view):

```
NSBezierPath* bp = [NSBezierPath bezierPathWithRoundedRect: self.bounds
xRadius: 5 yRadius: 5];
[[[NSColor blackColor] colorWithAlphaComponent: 0.5] setStroke];
[bp stroke];
```

Any ideas?

Edit:

If I inset the path by 0.5 everything draws just fine. But why is it that I get this when I offset the path by 10 pixels (for example)?



If I understand correctly, it should draw a thin line as well...

objective-c cocoa drawing nsbezierpath

Share

Improve this question

Follow

edited Dec 3, 2012 at 12:36

asked Dec 2, 2012 at 21:35

Alex

5,099 ● 4 ● 41 ● 74

The key is merely to inset by *at least* 0.5. That pulls you away from the edges of your clipping rect. You can go further, and get further away from the edge, but it won't have any additional effect. – Wade Tregaskis Dec 3, 2012 at 16:42

I know, but it seems that if I inset more than 0.5 I get a thicker line, and I don't understand why. It should draw exactly like when insetting by 0.5. – Alex Dec 4, 2012 at 7:51

2 Same reason as the answer explained - when you inset by 10.0, you're straddling to rows (or columns) of pixels, so you get a two-pixel wide grey line, instead of a one-pixel wide black line. Try insetting by 9.5, or 10.5, to see the difference. – Wade Tregaskis Dec 4, 2012 at 17:15

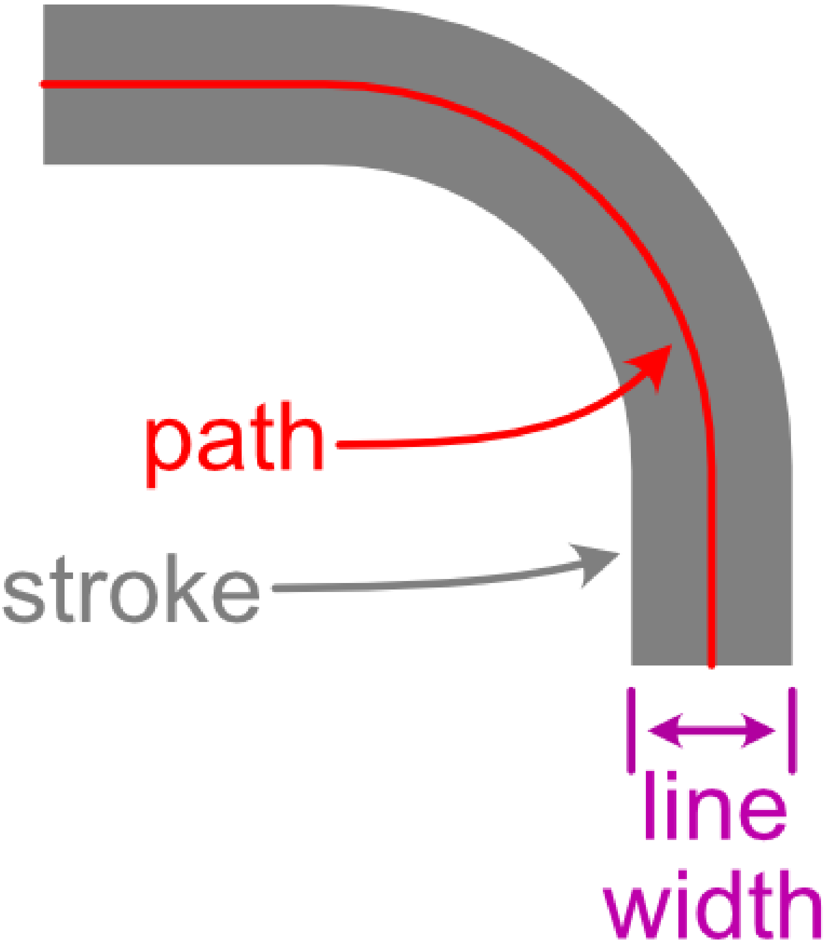
2 Answers

Sorted by: Highest score (default)

60

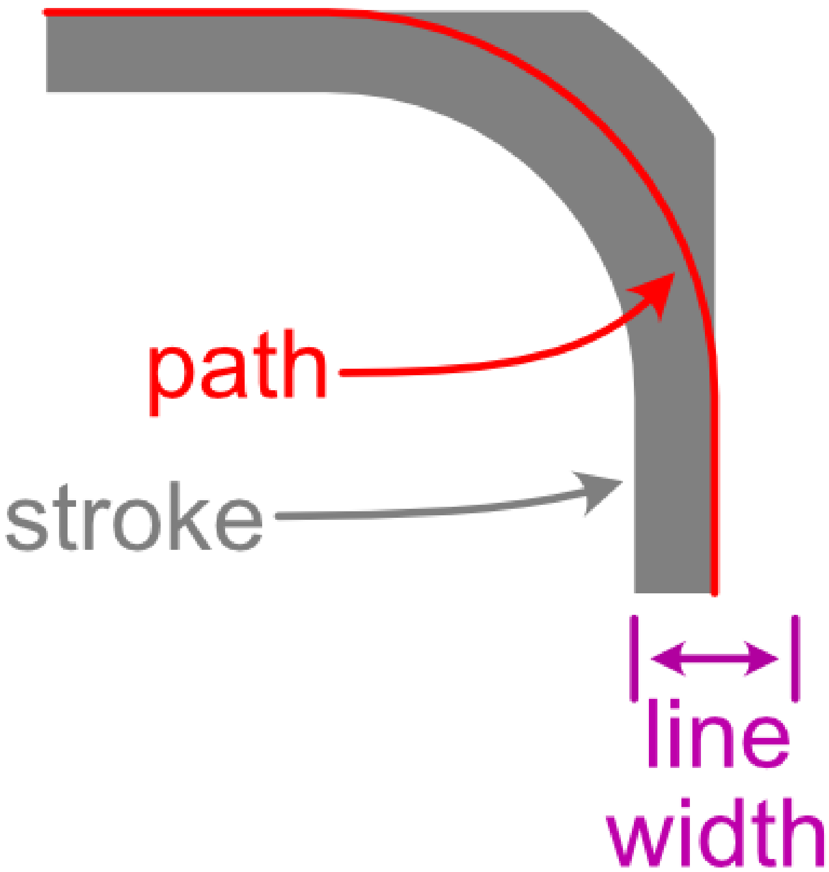
Many rendering systems are derived from the PostScript drawing model. Core Graphics is one of these derivative systems. (Here are some others: PDF, SVG, the HTML Canvas 2D Context, Cairo.)

All of these systems have the idea of stroking a path with a line of some fixed width. When you stroke the path, the line **straddles** the path: half of the line's width is on one side of the path, and half of the line's width is on the other side. Here's a diagram that may make this clearer:



Now, what happens when you stroke a path that lies along the boundary of your view? Half of the stroke falls outside of your view's bounds and is clipped away - not drawn. You only see the half of the stroke that falls inside the view's bounds.

When you use a rounded corner, that corner pulls away from the view's boundary, toward its center, so more of the stroke around the corner falls inside the view's boundary. So the stroke appears to get thicker around the rounded corner, like this:



To fix this, you need to inset your path by half the line width, so that the entire stroke falls inside your view's bounds along the entire path. The default line width is 1.0, so:

```
NSBezierPath* bp = [NSBezierPath bezierPathWithRoundedRect:
    NSRectInset(self.bounds, 0.5, 0.5) xRadius:5 yRadius:5];
```

Share

Improve this answer

Follow

edited Apr 13, 2022 at 14:50

answered Dec 2, 2012 at 22:10

rob mayoff

385k ● 68 ● 830 ● 878

Finally Me too got this amazing fact... Thanks Rob :) – Anoop Vaidya Dec 3, 2012 at 5:14

That... is an amazing answer. Thank you. I have one more question which has to do with your solution. Could you please see it? – Alex Dec 3, 2012 at 12:32

I can only see it if you post it. – rob mayoff Dec 4, 2012 at 22:28

2 1,000,000 hours and you are the only one who is able to bring to my attention "Half of the stroke will fall outside of your view's bounds and be clipped away - not drawn" no one else explicitly stated that and i was too dumb to make that connection after i knew the stroke was straddling the path, would +100 but i am limited to 1 – A'sa Dickens Jun 26, 2014 at 0:12

0

In iOS field, just minus the radius of the circle to prevent from being clipped.

```
UIBezierPath *roundPath = [UIBezierPath bezierPath];
[roundPath addArcWithCenter:
CGPointMake(self.frame.size.width / 2, self.frame.size.height / 2)
 radius:(self.frame.size.width / 2 - 0.5)
startAngle:M_PI_2 endAngle:M_PI * 3 / 2.f clockwise:YES];
```

Share Improve this answer Follow

answered Mar 13, 2019 at 2:25

tounaobun

14.9k ● 9 ● 57 ● 75