

Q: How do I make a window resize from a corner other than the upper right?

Q: I have a view that I want to resize. However, what I really want is for the window to be resized such that my view becomes the given size using the autosizing attributes that I have set in my window. How do I do that?

A: The following window method, `sizeWindow:byCorner:`, resizes a window to the given size by moving the given corner. The view method below it, `sizeTo::byWindowCorner:`, resizes a view by growing its window such that the view becomes the given size with respect to its autosizing settings. The two methods are implemented as categories to `Window` and `View` respectively.

```
// Format is 80 columns with 4 space tabs.
@interface Window(Sizing)
- sizeWindow:(NXCoord)width :(NXCoord)height byCorner:(int)corner
@end

@implementation Window(Sizing)

#define CORNER_UPPER_LEFT      0
#define CORNER_LOWER_LEFT     1
#define CORNER_UPPER_RIGHT    2
#define CORNER_LOWER_RIGHT    3

// Keep 'a' between x and y
#define CLAMP(a,x,y) (MAX((x), MIN((y), (a))))

/*****
This Method resizes the receiving window as if it was dragged with the given corner. This method is useful when you want the window to
resize by a corner other that the default upper right.
*****/
- sizeWindow:(NXCoord)width :(NXCoord)height byCorner:(int)corner
{
    NXRect newFrame;
```

```

NXSize minSize, maxSize;

// Clamp width and height to their respective minimum and maximum values
[self getMinSize:&minSize]; [self getMaxSize:&maxSize];
width = CLAMP(width, minSize.width, maxSize.width);
height = CLAMP(height, minSize.height, maxSize.height);

// Set newFrame from the old frame and the new sizes
NXSetRect(&newFrame, NX_X(&frame), NX_Y(&frame), width, height);

// Move the respective corner by the amount of growth and set newFrame
switch(corner) {
    case CORNER_UPPER_LEFT:
        NX_X(&newFrame) -= width - NX_WIDTH(&frame);
        break;
    case CORNER_LOWER_LEFT:
        NX_X(&newFrame) -= width - NX_WIDTH(&frame);
        NX_Y(&newFrame) -= height - NX_HEIGHT(&frame);
        break;
    case CORNER_UPPER_RIGHT:
        break;
    case CORNER_LOWER_RIGHT:
        NX_Y(&newFrame) -= height - NX_HEIGHT(&frame);
        break;
}
[self placeWindowAndDisplay:&newFrame];
return self;
}
@end

```

```

@interface View(Sizing)
- sizeTo:(NXCoord)width :(NXCoord)height byWindowCorner:(int)corner;
@end

```

```

@implementation View(Sizing)

```

/\*\*\*\*\*\*

This Method resizes the receiving view to the given width and height by resizing the window by the appropriate amount with respect to autosizing. This method is useful for those occasions when you know what size a view should be, but don't know how big to make the window to hold it. If you ask for a new width, it assumes the view is width sizable. The same goes for height. If the hierarchy contains a ClipView (ie, in a ScrollView) it assumes that you want the ClipView's subview to be fully exposed. The window size will not exceed the set maximum.

\*\*\*\*\*/

- sizeTo:(NXCoord)width :(NXCoord)height byWindowCorner:(int)corner

{

```
int autosizing = [self autosizing];
float widthGrowth = width - NX_WIDTH(&bounds);
float heightGrowth = height - NX_HEIGHT(&bounds);
float stretchingWidth = NX_WIDTH(&bounds);
float stretchingHeight = NX_HEIGHT(&bounds);
float newSuperWidth, newSuperHeight;
NXRect superFrame;
```

// If we are a contentView we simply need to grow window by our growth

```
if(self==[window contentView]) {
    [window setFrame:&superFrame];
    [window sizeWindow:NX_WIDTH(&superFrame) + widthGrowth
        :NX_HEIGHT(&superFrame) + heightGrowth byCorner:corner];
}
```

else {

```
[superview setFrame:&superFrame];
```

// Add margins to stretching lengths if they have been turned on in IB

```
if(autosizing & NX_MINXMARGINSIZABLE) stretchingWidth += NX_X(&frame);
if(autosizing & NX_MAXXMARGINSIZABLE)
    stretchingWidth += NX_WIDTH(&superFrame) - NX_MAXX(&frame);
if(autosizing & NX_MINYMARGINSIZABLE)
    stretchingHeight += NX_Y(&frame);
if(autosizing & NX_MAXYMARGINSIZABLE)
    stretchingHeight += NX_HEIGHT(&superFrame) - NX_MAXY(&frame);
```

// Add growth times a ratio of stretching length::view length to Super

```
newSuperWidth = NX_WIDTH(&superFrame) +
    widthGrowth*stretchingWidth/NX_WIDTH(&bounds);
```

```
newSuperHeight = NX_HEIGHT(&superFrame) +  
    heightGrowth*stretchingHeight/NX_HEIGHT(&bounds);
```

```
// Resize the Superview  
[superview sizeTo:newSuperWidth :newSuperHeight  
    byWindowCorner:corner];
```

```
// If we are a ClipView, bring the docview up to size  
if([self isKindOfClass:[ClipView class]])  
    [[(ClipView *)self docView] sizeTo:width :height];
```

```
    }  
    return self;
```

```
}
```

QA886

Valid for 1.0, 2.0, 3.0, 3.1