```
//
//	GRGradientFunctions
//
//	By Anders Bertelrud
//	Copyright (c) 1995-1996 Anders Bertelrud
//
//	This module contains two functions to draw NeXTSTEP 4.0 style gradients.
//	Both functions work by creating an 8-bits-per-sample bitmap on the fly and then drawing it
//	to the current context using NXDrawBitmap (which dithers it down to the screen depth). It
//	would almost certainly be more efficient to switch on the desired pixel depth and dither as
//	the pixels values are written. Since this is a quick hack to get blue gradient title bars
//	and gray gradient buttons to work, though, I didn't think it would be worth the time.
//	To make these functions reasonable efficient, fixed-point arithmetic is used (16 bits for
//	the whole number and 16 for the fraction). It's a simple linear interpolation, as in any
//	rasterizer, except that it's special-cased for the 45 degree diagonal gradients we're
//	interested in here. Rasterization 101. Simple stuff.
//	The functions are nowhere near optimized, though. About the only optimization made is the
//	fixed-point interpolation.
//	These functions can be used separately, without the other classes in this package.
//

#import <appkit/graphics.h>


void GRDrawHSBGradient (NXRect rectangle, float hue, float saturation, float startBrightness,
		float endBrightness);
	// Fills the given rectangle with a gradient in any hue. The gradient is NS4.0-style, i.e.
	// brightness is interpolated from the upper left corner to the lower right corner of the
	// rectangle. "Hue" and "saturation" remain constant throughout the gradient, and brightness
	// is interpolated from "startBrightness" (in the upper left corner) to "endBrightness" (in
	// the lower right corner). Lines perpendicular to the upper-left to lower-right diagonal
	// get constant colour.
	// The size of the input rectangle isn't checked. If you pass in a ten-thousand-by-ten-
	// thousand rectangle, you will fill up swap space. Remember, this is a hack. :-)

void GRDrawGrayGradient (NXRect rectangle, float startBrightness, float endBrightness);
	// Like GRDrawHSBGradient, except that the saturation is always zero (which makes the hue
	// irrelevant). Using this function is faster than calling GRDrawGrayGradient passing zero
	// for "saturation".
```