$SPAD/src/lib xshade.c

The Axiom Team

December 3, 2016

Abstract

Contents

1 License

```
/*
Copyright (c) 1991-2002, The Numerical ALgorithms Group Ltd.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:

    - Redistributions of source code must retain the above copyright
      notice, this list of conditions and the following disclaimer.

    - Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
      the documentation and/or other materials provided with the
      distribution.

    - Neither the name of The Numerical ALgorithms Group Ltd. nor the
      names of its contributors may be used to endorse or promote products
      derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER
OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*/


            —— * ——


#ifndef MSYSplatform

#include <stdio.h>
#if !defined(BSDplatform)
#include <malloc.h>
#endif
#include <stdlib.h>

#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/Xos.h>
```

```
#include <X11/Intrinsic.h>
#include <X11/StringDefs.h>
#include <X11/cursorfont.h>

#define XShadeWidth 4
#define XShadeMax  17

char XShadeBits[] = {
   0x00, 0x00, 0x00, 0x00,
   0x01, 0x00, 0x00, 0x00,
   0x01, 0x00, 0x04, 0x00,
   0x05, 0x00, 0x04, 0x00,
   0x05, 0x00, 0x05, 0x00,
   0x05, 0x02, 0x05, 0x00,
   0x05, 0x02, 0x05, 0x08,
   0x05, 0x0a, 0x05, 0x08,
   0x05, 0x0a, 0x05, 0x0a,
   0x07, 0x0a, 0x05, 0x0a,
   0x07, 0x0a, 0x0d, 0x0a,
   0x0f, 0x0a, 0x0d, 0x0a,
   0x0f, 0x0a, 0x0f, 0x0a,
   0x0f, 0x0b, 0x0f, 0x0a,
   0x0f, 0x0b, 0x0f, 0x0e,
   0x0f, 0x0f, 0x0f, 0x0e,
   0x0f, 0x0f, 0x0f, 0x0f};

#include "xshade.h1"

Pixmap XShade[XShadeMax];
GC  TileGC;
unsigned int INIT = 1;

/*
 * This routine has the function of returning the number of characters needed
 * to store a bitmap. It first calculates the number of bits needed per line.
 * Then it finds the closest multiple of 8 which is bigger than the number of
 * bits. Once that is done, it multiplies this number by the number of bits
 * high the bitmap is.
 */
int
char_bitmap(void)
{
    int bits_line;
    int total_chars;

    for (bits_line = 8, total_chars = 1; bits_line < XShadeWidth; total_chars++)
        bits_line += 8;

    total_chars = total_chars * XShadeWidth;
```

```
    return total_chars;
}

int
XInitShades(Display *display, int screen)
{
    char *bits;
    int count;
    int chars_bitmap = char_bitmap();
    int bit;

    bits = (char *) malloc(chars_bitmap * sizeof(char));

    for (count = 0; count < XShadeMax; count++) {

        /* Load in the next bitmap */

        for (bit = 0; bit < chars_bitmap; bit++)
            bits[bit] = XShadeBits[count * chars_bitmap + bit];

        /* Create it and put it into the Pixmap array */

        XShade[count] = XCreatePixmapFromBitmapData(display,
                                                RootWindow(display, screen),
                                                    bits,
                                                    XShadeWidth, XShadeWidth,
                                                BlackPixel(display, screen),
                                                WhitePixel(display, screen),
                                            DisplayPlanes(display, screen));
    }
    TileGC = XCreateGC(display, RootWindow(display, screen), 0, NULL);
    XSetFillStyle(display, TileGC, FillTiled);
    XSetTile(display, TileGC, XShade[XShadeMax / 2]);
    return XShadeMax;
}


int
XChangeShade(Display *display, int shade)
{
    if (shade >= XShadeMax || shade < 0) {
        fprintf(stderr, "Shade %d, out of range\n",shade);
        return (-1);
    }
    XSetTile(display, TileGC, XShade[shade]);
    return (1);
}

int
XQueryShades(unsigned int *shades)
```

```
{
    *shades = XShadeMax;
    return 1;
}


void
XShadeRectangle(Display *display, Drawable drawable, int x,int y,
                unsigned int width, unsigned int height)
{
    if (!INIT) {
        fprintf(stderr, "xshade Error: Tried to fill before INIT called\n");
        exit(-1);
    }
    XFillRectangle(display, drawable, TileGC, x, y, width, height);
}


void
XShadeRectangles(Display *display, Drawable drawable,
                 XRectangle *rectangles, int nrectangles)
{
    if (!INIT) {
        fprintf(stderr, "xshade Error: Tried to fill before INIT called\n");
        exit(-1);
    }
    XFillRectangles(display, drawable, TileGC,
                          rectangles, nrectangles);
}


void
XShadePolygon(Display *display, Drawable drawable, XPoint * points,
              int npoints, int  shape, int mode)
{
    if (!INIT) {
        fprintf(stderr, "xshade Error: Tried to fill before INIT called\n");
        exit(-1);
    }

    XFillPolygon(display, drawable, TileGC,
                        points, npoints, shape, mode);
}

void
XShadeArc(Display *display, Drawable drawable, int x, int y,
          unsigned int width, unsigned int height, int angle1, int angle2)
{
    if (!INIT) {
        fprintf(stderr, "xshade Error: Tried to fill before INIT called\n");
```

```
        exit(-1);
    }
    XFillArc(display, drawable, TileGC, x, y, width,
                    height, angle1, angle2);
}


void
XShadeArcs(Display *display, Drawable drawable, XArc *arcs, int narcs)
{
    if (!INIT) {
        fprintf(stderr, "xshade Error: Tried to fill before INIT called\n");
        exit(-1);
    }
    XFillArcs(display, drawable, TileGC, arcs, narcs);
}

#endif /* MSYSplatform */
```

References

[1] nothing