

```

/* boopsi1.c - Execute me to compile me with Lattice 5.10a
LC -bl -cflstg -v -y -j73 boopsi1.c
Blink FROM LIB:c.o,boopsi1.o TO boopsi1 LIBRARY LIB:LC.lib,LIB:Amiga.lib
quit
*/

/* boopsi example showing simple creation of boopsi
** gadgets and using system images.
*/

#include <intuition/intuition.h>
#include <intuition/classusr.h>
#include <intuition/imageclass.h>
#include <intuition/gadgetclass.h>
#include <clib/exec_protos.h>
#include <clib/intuition_protos.h>
#include <clib/graphics_protos.h>

#ifdef LATTICE
int CXBRK(void) { return(0); } /* Disable Lattice CTRL/C handling */
int chkabort(void) { return(0); }
#endif

UBYTE *vers = "\0$VER: boopsi1 1.0";

void ProcessWindow(void);

#define MYPROP      1L
#define MYSTRING    2L
#define MYLEFTBUT   3L
#define MYRIGHTBUT  4L

#define GADTOP      10L
#define STRLEFT     20L
#define STRWIDTH    30L
#define STRHEIGHT   11L
#define INITVALUE   25L
#define PROPWIDTh  100L
#define PROPVIS     10L
#define BUTGADWID   11L
#define BUTGADHEI   11L

struct Screen *screen;
struct Window *window;
struct DrawInfo *drinfo;
struct Library *IntuitionBase;
struct Gadget *prop, *string, *leftbut, *rightbut, *mygadgets;
struct Image *rightbimage, *leftbimage;

void main(void)
{
    struct Gadget *tmpgad;
    WORD gadtop;

    tmpgad = (struct Gadget *)&mygadgets

    if (IntuitionBase = OpenLibrary("intuition.library", 36L))
    {
        if (screen = LockPubScreen(NULL))
        {
            /* need my screen's DrawInfo for the system gadget images */
            drinfo = GetScreenDrawInfo(screen);

            gadtop = screen->Font->ta_YSize + GADTOP;
            if (window = OpenWindowTags(NULL,
                WA_Title, (ULONG *) "AMail boopsi1",
                WA_Height, gadtop+STRHEIGHT+10L,
                WA_Width, STRLEFT+STRWIDTH+PROPWIDTh+2*BUTGADWID+15,
                WA_Flags, WFLG_DEPTHGADGET | WFLG_DRAGBAR | WFLG_CLOSEGADGET,
                WA_IDCMP, IDCMP_CLOSEWINDOW | IDCMP_GADGETUP,
                TAG_END))
            {
                /* Create a boopsi string gadget */

```

```

if (string = (struct Gadget *)NewObject(NULL, "strgclass",
/* All the normal Gadget fields */
GA_ID, MYSTRING,
GA_Immediate, TRUE,
GA_RelVerify, TRUE,
GA_Top, gadtop,
GA_Left, STRLEFT,
GA_Width, STRWIDTH,
GA_Height, STRHEIGHT,
STRINGA_MaxChars, 3,
STRINGA_LongVal, INITVALUE,
STRINGA_Justification, STRINGRIGHT,

/* Boopsi makes it easy to link gadgets together.
** The GA_Previous tag accepts a (struct Gadget **)
** to the previous gadget in the list, then changes
** this value to point to the gadget being
** created (in this case, "string").
*/
GA_Previous, tmpgad,
TAG_END))

{
    /* create the prop gadget */
    if (prop = (struct Gadget *)NewObject(NULL, "propgclass",
GA_Immediate, TRUE,
GA_RelVerify, TRUE,
PGA_Freedom, FREEHORIZ,
PGA_Borderless, FALSE,
GA_Left, STRLEFT + STRWIDTH,
GA_Top, gadtop,
GA_Height, BUTGADHEI,
PGA_Top, INITVALUE,
PGA_Visible, PROPVIS,
PGA_Total, PROPWIDTh,
GA_ID, MYPROP,

/* link prop to string and
** make tmpgad point to prop. */
GA_Previous, tmpgad,
TAG_END))

{
    /* Ask the system for a left arrow image
    ** for the left arrow gadget. */
    if (leftbimage = (struct Image *)NewObject(NULL,
"sysiclass",

/* boopsi needs this screen's DrawInfo
** structure to get the right image. */
SYSIA_DrawInfo, drinfo,
SYSIA_Which, LEFTIMAGE,

/* this will give us 11 x 11 buttons */
SYSIA_Size, SYSISIZE_MEDRES,
TAG_END))

{
    /* Now ask for a right arrow */
    if (rightbimage = (struct Image *)NewObject(NULL,
"sysiclass",
SYSIA_DrawInfo, drinfo,
SYSIA_Which, RIGHTIMAGE,
SYSIA_Size, SYSISIZE_MEDRES,
TAG_END))

{
        /* Create the left button */
        if (leftbut = (struct Gadget *)NewObject(NULL,
"buttongclass",
GA_ID, MYLEFTBUT,
GA_Immediate, TRUE,
GA_RelVerify, TRUE,
GA_Image, leftbimage,
GA_Top, gadtop,
GA_Left, STRLEFT+STRWIDTH+PROPWIDTh-15,
GA_Width, BUTGADWID,
GA_Height, BUTGADHEI,
GA_Previous, tmpgad,
TAG_END))

{
            /* Create the right button */

```

```

        if (rightbut = (struct Gadget *)NewObject(NULL,
            "buttongclass",
            GA_ID,          MYRIGHTBUT,
            GA_Immediate, TRUE,
            GA_RelVerify, TRUE,
            GA_Image,       rightbimage,
            GA_Top,         gadtop,
            GA_Left,        STRLEFT+STRWIDTH+PROPWID+10,
            GA_Width,       BUTGADWID,
            GA_Height,      BUTGADHEI,
            GA_Previous,    tmpgad,
            TAG_END))
        {
            /* All of the gadgets have been created
            ** and linked together. Add them to the
            ** display and display them.
            */
            AddGList(window, mygadgets, -1, -1, NULL);
            RefreshGList(mygadgets, window, NULL, -1);
            ProcessWindow();
            RemoveGList(window, mygadgets, -1);
            DisposeObject(rightbut);
        }
        DisposeObject(leftbut);
    }
    DisposeObject(rightbimage);
}
DisposeObject(leftbimage);
}
DisposeObject(prop);
}
DisposeObject(string);
}
CloseWindow(window);
}
FreeScreenDrawInfo(screen, drinfo);
UnlockPubScreen(NULL, screen);
}
CloseLibrary(IntuitionBase);
}
}

void ProcessWindow(void)
{
    struct IntuiMessage *imsg;

```

```

    BOOL returnvalue = TRUE;
    ULONG class;
    LONG currval = INITVALUE;
    struct Gadget *g;

    while (returnvalue)
    {
        WaitPort(window->UserPort);
        while (imsg = (struct IntuiMessage *)GetMsg(window->UserPort))
        {
            g = (struct Gadget *)imsg->IAddress;
            class = imsg->Class;
            ReplyMsg((struct Message *)imsg);
            switch (class)
            {
                case IDCMP_CLOSEWINDOW:
                    returnvalue = FALSE;
                    break;
                case IDCMP_GADGETUP:
                    switch (g->GadgetID)
                    {
                        case MYLEFTBUT:
                            currval--;
                            break;
                        case MYRIGHTBUT:
                            currval++;
                            break;
                        case MYPROP:
                            /* read the prop gadget's value */
                            GetAttr(PGA_TOP, prop, &currval);
                            break;
                        case MYSTRING:
                            /* read the string gadget's value */
                            GetAttr(STRINGA_LongVal, string, &currval);
                            break;
                    }
                    /* make sure the value is between 0 and 90 */
                    if (currval < 0L)
                        currval = 0L;
                    else
                        if (currval > PROPWID - PROPVIS)
                            currval = PROPWID - PROPVIS;

                    /* Update the values of the prop and string.
                    ** gadgets (Intuition takes care of the refresh). */

                    SetGadgetAttrs( prop, window, NULL,
                                    PGA_TOP, currval,
                                    TAG_END );

                    SetGadgetAttrs( string, window, NULL,
                                    STRINGA_LongVal, currval,
                                    TAG_END );

                    break;
            }
        }
    }
}

```