This document contains some information about a simplified interface to the OSF/Motif programming interface with respect to graphics.

It was written by Nils Schmeisser and is restricted by the GNU General Public License Agreement. You may not redistribute nor resale this document or parts of it. If you use the document or parts of it you should place the folloing in your index section:

"XGRAPH - simplified graphics interface to OSF/Motif programming interface"; Nils Schmeisser

Documentation XGRAPH

This module contains a certain number of graphics functions using X. The style is orientated to Borland's calling style, to make it easy to adapt PC C code to the X enviroment. There are some predefined constants:

xBLACK
xBLUE
xGREEN
xCYAN
xRED
xMAGENTA
xBROWN
xLIGHTGRAY
xDARKGRAY
xLIGHTBLUE
xLIGHTGREEN
xLIGHTCYAN
xLIGHTRED
xLIGHTMAGENTA
xYELLOW
xWHITE

To prevent the user from thinking about the colorindexes he should use these colorconstants for his application. If more colors are needed, the user may define colors with indexes from 0 up to the number given by getmaxcolor()-1 by using the setrgbpalette(index,r,g,b). If one wants to go deeper into the programming in X: X reserves about 40 colors for internal use. In this enviroment these internal colors got indexes from -40 to -1 the absolute colorindex is computed by Index+ColorOffset, where ColorOffset is set to 40. To have access to the internal colors the user should use the indexes -40,..,-1. But it is not recomended to do it.

How to program:

The programming follows the normal X programming style, that means the user has to initialize the display, the application context, the topshell and so on (Ref. XAPP Interface).
If one don't want to take care what's behind the XGRAPHICS he should start at step 2.
For those howknow how to handle the display the may start at step 1.
Initializing the X graphics enviroment:

1. setgraphicsdisplay(display)
2. initgraph()

Using the X graphics enviroment

1. Window window=open("Window name",x,y,sizex,sizey,parent)
2. call to graphics functions

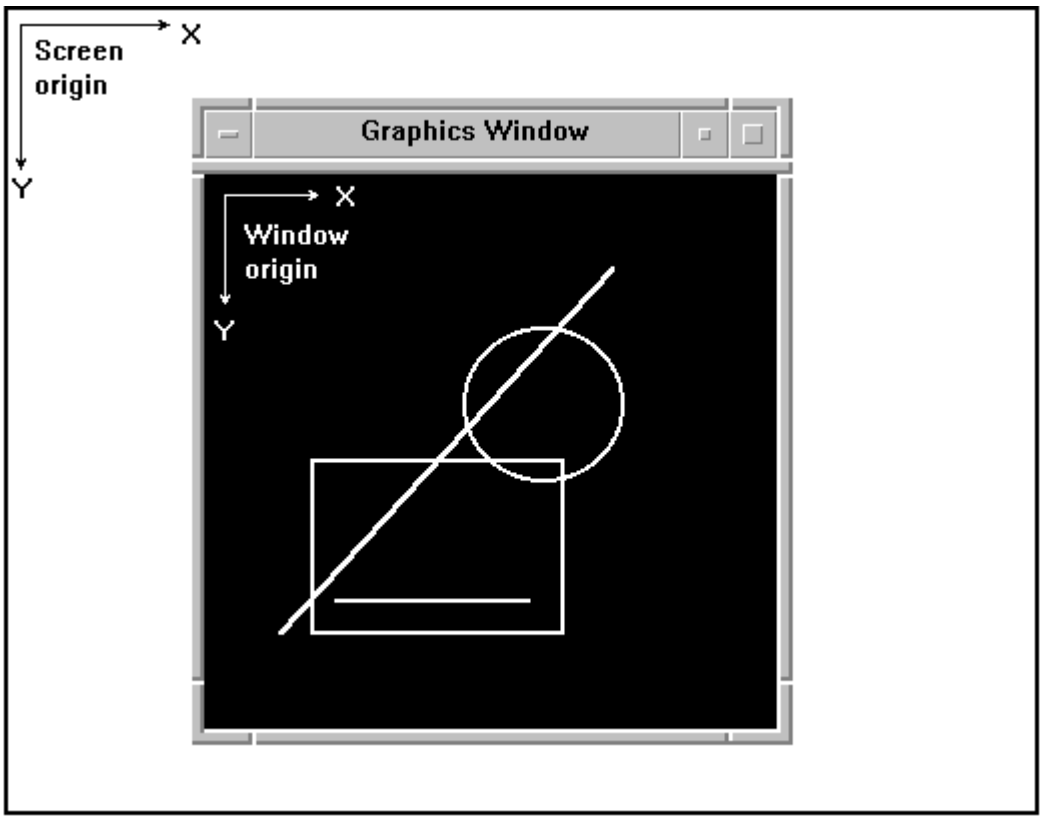
Closing the X graphics enviroment

1. closewindow(window)
2. closegraph()

Example 1:

```
#include "xapp.h"
#include "xgraph.h"

void draw_something_in_window(Window window) {
   int x,y;

   // draw a red line on top of the window

   for (x=0;x<getmaxx(window);x++) putpixel(window,x,0,xLIGHTRED);

   // draw some lines in blue and green

   setcolor(xLIGHTBLUE,window);
   for (y=0;y<getmaxy(window);y += 10) line(window,0,y,getmaxx(window),y);
   setcolor(xLIGHTGREEN);
   for (x=0;x<getmaxx(window);x += 20) line(window,1,x,getmaxy(window),x);
}

// insert here example 2 redraw_something

Display *display;
XtAppContext context;
Widget TopShell;
Window window;

void main() {
   int i;

   TopShell=xInitApplication("Test",&dpy,context);
   setgraphicsdisplay(dpy);
   initgraph();
   window=openwindow("Graphics Window",0,0,300,200,TopShell);

   // insert here exapmple 2: setredrawfunction

   draw_something_in_window(window);
   printf("Enter a number:");
   scanf("%d",&i);
   closegraph();
}
```

Coordinate system:



Types:

bitmap type

A bitmap consists of a header, holding information about the size of the bitmap and a data part. It is defined as:

```
typedef bitmap {
        word sizex;
        word sizey;
        byte data[sizex*sizey];
}
```

This is only a definition to show how the bitmap looks like. The reference given to the ***getimage*** and ***putimage*** function is only a reference pointing to an array of byte with the size of *sizex*sizey+4*

polypoint type

Is a structure according to the *XPoint* structure, which is defined as:

```
struct XPoint {
        int x;
        int y;
}
typedef XPoint polypoint;
```

XGraphicsRedrawFunction type

A redraw function of this type is defined as

```
typedef void (*XGraphicsRedrawFunction)(Window window,int reason);
```

If a function is assigned to a window via ***setredrawfunction*** it is always called if something happend to the window. The <reason> givs information about what happened. The window parameter holds the windowID for the window which caused the redraw function to be activated.

Exapmle 2:

```
.
.
void redraw_something(Window window,int reason) {
        printf("Window %d was ");
        switch (reason) {
                case ResizeRequest:     printf("resized.\n");
                                        break;
                case Expose:            printf("exposed.\n");
                                        break;
                case ButtonPress:       printf("clicked by mous.\n");
                                        break;
                case KeyPress:          printf("entered by keyboard.\n");
        }
        draw_something_in_window(window);
}
.
.
setredrawfunction(window,redraw_something);
.
.
```

Calling styles:

There are two different calling styles:

- giving the window, the action should effect on and
- using the ***default window*** for action

The first should be called the "***full calling style***", the second the "***default calling style***". For the default calling style it is not nessecary to give the *Window* parameter in the function arguments. To give a colorvalue to the drawing functions is not neccessary. An optional color argument is indicated as *...,int color = -MAXINT.*
Anyway, a default argument can be left out. A default argument is indicated by

<type> <argument> ***= <value>***

in the definition.

Functions:

void initgraph(Display *display = NULL)

Initializes the graphics enviroment for X. No additional parameters are needed. The call to initgraph with display argument unequal to NULL is the same as calling to setgraphicsdisplay and then calling to initgraph().

void closegraph()

Closes the graphics enviroment. The user should take care, that all graphics windows are closed bevor closegraph() is called.

Display *getgraphicsdisplay()

Returns the current graphics display.

void setgraphicsdisplay(Display *dpy);

Sets the current graphics display. Should be called only once bevor calling initgraph().

Window openwindow(char *name,int x,int y,int sizex,int sizey,Widget parent,
Widget *container = NULL)
Window openwindow(char *name,int x,int y,int sizex,int sizey,Widget parent,
XGraphicsRedrawfunction f,Widget *container = NULL);

Opens a window with the name <name> in the title bar. The windows origin is at (x,y) and the window has a size of <sizex> and <sizey>. The <parent> is the windows parent. The window is associated to a widget. The widget may be obtained by calling to Widget XtWindowToWidget(Display *display,Window window) or with adding a variable of type Widget* to the function call (which will return the container widget). The second call adds the ***redrawfunction***.

void setredrawfunction(Window window,XGraphicsRedrawFunction f)

Sets the redraw function for the window <window>. The redraw function is called if the window is resized, exposed or the mouse was clicked inside the window or if a key was pressed while the window was on top. The redraw function must be a function of ***XGraphicsRedrawFunction type***. The ***reason*** for calling the redraw function might be:

ResizeRequest	indicates, that the window was resized
Expose	indicates, that the window was exposed
ButtonPress	indicates, that a mouse button was pressed
KeyPress	indicates, that a key was pressed.

void setwindow(Window window)

Makes window the default window. (Ref. ***small calling style***)

void closewindow(Window window)

Closes the window <window>.

void keepratio(Window window,Boolean keepit)

Defines wether the ratio is to be kept or not.
Values for ***keepit*** are ***false*** or ***true***.

void setrgbpalette(int colornum, int red, int green, int blue)

Sets the RGB values for color number <colornum>. Normally the <colornum> parameter is in the range of 0,..,getmaxcolor(), but for special cases it might be also in the range of -40,..,-1 (see above). The RGB values are in the range of 0,..,255.

int getmaxcolor()

Returns the maximum number of user defined colors.

The following functions might be called in both styles:

void cleardevice(Window window)

Clears the window with the default background color.

void setcolor(Window window,int color)

Sets the default foreground color for the window <window>.

int getfgcolor(Window window)

Returns the foreground color value.

void setbkcolor(Window window,int color)

Sets the default background color for the window <window>.

int getbkcolor(Window window)

Returns the background color value.

int getmaxx(Window window)
Returns the width of the window.

int getmaxy(Window window)
Returns the height of the window <window>.

void getsize(Window window,int &w,int &h)
Returns both width and height in <w> and <h>.

void getposition(Window window,int &x,int &y)
Returns the origin of the window.

void moveto(Window window,int x,int y)
Moves the pen to (x,y) (ref. **coordinate system**).

void putpixel(Window window,int x,int y,int color = -MAXINT)
Puts a pixel at (x,y) in the window.

void putRGBpixel(Window window,int x,int y,int r,int g,int b)
Puts a pixel at (x,y) in the window and passes the color lookup table by direct setting the RGB values. Not recommended for slow machines.

void line(Window window,int x1,int y1,int x2,int y2,int color = -MAXINT)
Draws a line from (x1,y1) to (x2,y2) window.

void lineto(Window window,int x,int y,int color = -MAXINT)
Draws a line from the current pen position to (x,y) and sets the pen to (x,y).

void drawpoly(Window window,int num,polypoint *points,int color = -MAXINT)
Draws a polygon with <num> points defined in an array of ***polypoint type***.

void fillpoly(Window window,int num,polypoint *points,int =-MAXINT)
Draws a filled polygon using the foreground color.

void rectangle(Window window,int xa,int ya,int xb,int yb,int = -MAXINT)
Draws a rectangle, given by its upper left and its bottom right corner.

void bar(Window window,int xa,int ya,int xb,int yb,int = -MAXINT)
Draws a filled rectangle (bar). The bar is given by its upper left and bottom right corner.

unsigned long imagesize(int x1,int y1,int x2,int y2)
Returns the size in bytes needed to store the image returned by ***getimage***.

void getimage(Window window,int x1,int y1,int x2,int y2,void *bitmap)
Returns a bitmap of ***bitmap type*** in <bitmap>. The bitmap holds the area defined by (x1,y1) as upper left and (x2,y2) as bottom right corner.

void putimage(Window window,int x,int y,void *bitmap,int flag)
Puts a bitmap of ***bitmap type*** at (x,y) in the defined window. The ***flag*** indicates how to put the bitmap. Possible assignements are still not defined. Arguments have no effect.

Additionally:

void putimage(Window window,int x,int y,Impvaru *image)

Additionally using the ***Image Assistent types***.