11/21/2007
Let's look at solutions to the interclass problem.
Vote was 7 for ASCII, 10 for X11.
Do you have any questions about the assignment?
The ncurses library provides you with functions that allow you to write any ASCII character that you want to any point on the screen. This can allow you to do ASCII animations. It also includes functions for doing non-blocking input and input of non-character keys. Normal scanf and getchar functions only read from a terminal once enter has been pressed. Note that this isn't portable code. It won't work on a Windows box unless perhaps you have something like Cygwin.
In order to set up a window so that you can use curses function with it you need to initialize it. You do this with the WINDOW *initscr() function. Let's look at the man page for this function. The WINDOW* that is returned needs to be remembered because it has to be passed to most of the functions that will draw out characters for us.
There are number of different functions that can output to the screen. I'm not going to type them all up here, but I'll give a little list. We can look at a web page for a more complete list.

- `mvaddch(win,y,x,ch)`
- `mvwaddstr(win,y,x,str)`
- `mvwaddchstr(win,y,x,chstr)`

After calling one of these call `refresh(win)` to see the change.

You can use the See Also section of the man pages to find more related functions.
There are a number of functions for getting input in curses. The simplest is wgetch(win).

Unless you set other options this will work just like getchar() blocking the program and printing to screen.

The following lines will allow non-blocking, non-printed input that doesn't print.

- keypad(win,1);
- wtimeout(win,0);
- necho();
Linux and Unix boxes use a windowing system called X-windows. This is a very general graphical system that allows a lot of flexibility.

One of the most significant features of X-windows is the ability to display windows remotely.

Cygwin can give you X-windows under Windows, but by default you can't use it there.
There are several steps in getting an X window.
- Get a display. (XOpenDisplay)
- Get a screen number on the display. (XDefaultScreen)
- Create the window. (XCreateSimpleWindow)

Here again, the man pages can help us get more information.
Most GUI environments use events for user input. That's because you can't block the thread and there are many different event types (key presses, mouse actions, exposures, etc.)

The is a big union called XEvent that has the different events. Every structure in it shares certain elements at the beginning.

The XNextEvent function will return the next event to you. You switch on the type of the event to determine what you should do about it.

For X-programs the event handling likely belongs in a loop.
One type of event that is particular to the X system is exposures. They tell you that some part of the window has been exposed and that you need to redraw it. By default, X-windows don't remember what is drawn in them so the refresh has to be done manually.
The graphics context in X is a struct that tells X how to draw things.

You can create multiple instances of this structure and set them up to draw in different ways. You can think of each one as being like a pencil or brush with a different color, width, etc.

The struct is called XGCValues.
There are a bunch of drawing functions in X11 that begin with XDraw and XFill. They are fairly simply to master and running a command like “man -k XDraw” will list them.

Colors are more challenging. The XAllocNamedColor will give you a color from a set of “named” colors. For more variety you need to use XCreateColormap to make a full color map. This makes X applications more cross-platform compatible, but harder to work with.
What are you doing for Thanksgiving?
Interclass Problem – Write a little program to use one of these two libraries.