Blanker

COLLABORATORS			
	TITLE : Blanker		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		December 6, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Blanker

1.1 Blanker Documentation

```
Blanker, A System Friendly Solution to Blanking Screens

  Select from the following topics:

  @{ " Introduction               " link introduction }
  @{ " Disclaimer                 " link disclaimer }
  @{ " Copyright and Distribution " link copyright }
  @{ " Requirements               " link requirements }
  @{ " Installation               " link installation }
  @{ " Using Blanker              " link usage }
  @{ " Module Usage Notes         " link moduse }
  @{ " Module Creation            " link modules }
  @{ " Credits                    " link credits }
  @{ " Plans                      " link plans }
  @{ " History                    " link history }
  @{ " F.A.Q.                     " link faq }
```

1.2 Introduction

```
Blanker is an attempt to provide the Amiga community with a future-
compatible, easily expandable screen blanker.

I would like to provide a platform for others to write custom screen
blanker modules when the whim strikes and not have to worry about the
difficulties associated with setting up a Commodities interface and
dealing with concurrency problems.
```

1.3 Disclaimer

```
The author cannot be held liable for the suitability or  accuracy  of this
manual  and/or  the  program(s)  it  describes.  Any  damage  directly  or
indirectly caused by the use or misuse of this manual  and/or the  program
it describes is the sole responsibility of the user her/him self.
```

1.4 Copyright and Distribution

Blanker is free software. I retain rights to this software to the extent that no one else can claim rights to the software and change its public domain status.

I release this program and its corresponding source and documentation with the following terms:

- You may use any part of this software in any venture of your own, provided that your software is released for free use as well. This means that Freeware or Shareware with no disabled features may freely use or include any part of the Blanker software package.

Note: Nico François' ReqTools library is included in this distribution, but these terms do not apply to that portion of the distribution. Please read the documentation in the ReqTools distribution for information regarding such.

Unlike most people, I regard computer programming as an art form. Certainly I put as much love and effort into a project like this as any artist would into a painting, sculpture, or piece of music. There may seem to be more 'right answers' in computing than in art, but I contend that this only seems to be the case and that people who have received art degrees could lecture on and on about the 'right' way to do things artistically as well.

I love to program and I'm going to program regardless of whether I can make money from my work. So when I release this software, I do it as the musician who plays on the side of the street. If you just happen to be walking by and enjoy the song, feel free to drop a little cash in the saxophone case or even simply take the time to say "Nice song, man". So if you're so inclined, my e-mail and regular addresses are listed in the 'Credits' section and I'd be happy to hear from you.

1.5 Requirements

This software requires that you be running AmigaDOS(tm) 2.04 or higher.

Additionally, reqtools.library is used by the program and is subsequently included in the distribution archive.

Note: reqtools.library is Copyright (c) Nico François

Justification: ReqTools is included because the ASL library does not include a screen mode requester under AmigaDOS 2.04. Faced with three choices: deny users of less than 2.1 access to my program, use ReqTools, or write my own proprietary screen mode requester, I opted for the ReqTools solution because many people already have and use ReqTools and most people in the U.S. don't have AmigaDOS 2.1 yet anyhow. (Political interjection: If C= made the upgrade freely distributable then it wouldn't matter.)

1.6 Installation

Installing Blanker is a very simple process. Double click on the installation icon and click on the pretty buttons.

1.7 Using Blanker

The Blanker interface is a fairly simple one. I will describe the important aspects as follows:

Screen mode button: Selecting this brings up a requester prompting you to select a screen mode. Here you can specify which type of screen you want the blanker to come up on and the number of colors that screen has. This information is displayed in the two text fields to the right of the screen mode button like so:

```
+-----------+ +-----------------------------------+ +-------------+
| Screen... | | Screen mode description           | | # of Colors |
+-----------+ +-----------------------------------+ +-------------+
```

Module preferences button: Selecting this brings up the preferences panel for the currently selected blanker module. Adjusting these parameters and pressing the OK button on this panel will change the behavior of the screen blanker module's graphical display. If you press CANCEL on this panel, the changes you made will be ignored. To the right of the module preferences button is a description of the active screen blanking module:

```
+-----------+ +-------------------------------------------------+
| Prefs...  | | Description of the currently selected module    |
+-----------+ +-------------------------------------------------+
```

Timeout integer gadget: This field allows you to change the length of time the blanker waits before blanking the screen. This value is in seconds.

Blank key text gadget: This field allows you to change the key that is pressed to blank the screen. This key is described in the standard Commodites way (for more information see page 5-29 of the "Using the System Software" manual supplied with your computer).

Pop key text gadget: This is similar to the blank key gadget. It allows you to set the key that is used to pop up the Blanker window. Additionally, the Commodities exchange program can be used to pop up the Blanker interface (refer to page 4-27 of your "Using the System Software" manual for more info on the Commodities exchange program).

Memory resident/Load on request gadget: When this cycle gadget is in memory resident mode, the current module is kept in memory and signaled when blanking occurs. If it is in load on request mode then the module is loaded from disk whenever a blank occurs or when the preferences are edited. I included this for users with little memory and for users running on one-floppy no-HD systems, this way you can run Blanker on startup and you won't be required to insert the disk that the module is on when the blanking or preferences editing occurs.

Load button: This button pops up a requester in which you can select a different blanker module to use. This new blanker will be loaded and its information will be displayed in the module description box. If the module cannot be found, the internal (simple black screen) blanker will become active. (Note: Be sure what you select is a blanker module if you specify something that is not a Blanker module here, the program will hang. I concluded that searching the actual binary for some sort of cookie (identifying string) took too long and was intolerable for floppy users, so Blanker only checks to see if a file exists before trying to use it as a Blanker. Also note, Blanker WON'T crash the system. It will simply lock up quietly, so don't be concerned about losing data or otherwise having Blanker wreak any havoc around your system.)

Hide button: This button hides the blanker interface from view. Blanker is still running when its interface is hidden and you can pop the interface back up with the Pop key or the Commodities exchange program. Also note that the close button in the upper left hand corner of the screen does the same thing as the Hide button. It doesn't actually quit the program.

Save button: The save button saves the current preferences so that when you reboot your computer, the settings will be the same. Additionally, the preferences for the selected module are also saved.

Quit button: This tells blanker to exit and frees up any resources it was using at the current time.

Additionally, Blanker has a number of menu options:

Project menu:
Open...: This allows you to open a preferences file directly. These files are generally stored in the Presets subdirectory of your Preferences directory. A file requester will appear and you can click on the file you wish to load.

Save as...: This allows you to save the current set-up to a file that can later be opened from the Open... menu item. This will bring up a file requester and allow you to enter the name and location of the file.

Load module: Here is where you can load different blanker modules. It has the same function as the 'Load' button.

About: This pops up a panel with a little blurb about Blanker.

Quit: This is the same as pressing the Quit button in the interface window.

Edit menu:
Last saved: If you make modifications to the current set-up and then decide you would like to start over, this menu item will load the Blanker preferences file again and restore the set-up from your last save.

Finally, there are two standard tooltypes supported by Blanker. They are:

CX_POPUP: If you set CX_POPUP=YES in the tool types section of the Info panel for the Blanker icon, Blanker will popup its interface when it is run. If it is set to CX_POPUP=NO, Blanker will remain hidden on startup and can be popped up with the Pop key or Commodities exchange.

CX_PRIORITY: This will probably not need to be changed for normal use. It is implemented for completeness. CX_PRIORITY is set to a number that tells the Commodities brokers who to send input events to first. This way if two Commodities are monitoring the same key (the Pop key, for example) the one with the higher priority gets that key-press first.

1.8 Module Usage

There is only one thing I think needs explaining with the modules. In an effort to keep the text module from getting any bigger, I didn't put a file requester in it for selection of a text file from which to pick quotes. So, when you put it in "Files" mode, you just type the full (or at least relative to the directory that Blanker was started from) path name to the text file and Blanker will pick a random quote from that text file.

Also the format is:

```
 Number of quotes in file     <-- No extra returns
 Font_Name Font_Size Quote    <-/ between any lines.
 ...
```

See the example, Text.quotes.

1.9 Module Creation

I've tried to make writing your own Blanker module as simple a task as possible. So let me elucidate the few things that you must keep in mind when writing your own module:

Your module must have the following three functions:

```
void blank( struct bMessage *bMsg );
void prefs( UBYTE *prefData );
void defs( UBYTE *prefData );
```

The bMessage structure is defined in the @{ "defs.h" LINK "Source/Blankers/TextD/defs.h/MAIN" } file (which you'll have to include in your source). Here it is for reference:

```
struct bMessage {
  struct  Message bMess;
  ULONG sMod;     /* Screen mode */
  ULONG sDep;     /* Screen depth */
  ULONG bm_Type;    /* Command type (used in main.c) */
  ULONG bm_Valid;   /* Validity of preferences data */
  UBYTE *prefData;    /* Module specific preferences data */
};
```

I have provided for you a @{ "main.c" LINK "Source/Blankers/TextD/main.c/MAIN" } file which you can link with your
blank(), prefs(), and defs() functions. It sets up a message port and
communicates with the Blanker program taking care of all the details.

The defs() function simply copies valid default preferences information
into the prefData space. Blanker and @{ "main.c" LINK "Source/Blankers/TextD/main.c/MAIN" } will take care of checking
whether or not the prefs are valid and will call defs() to get valid prefs
when necessary. This guarantees that blank() and prefs() will always be sent
valid preferences information when they are called.

The prefs() function will have to open a window, get the preferences settings
from the user, copy those into prefData space, close the window, and return.
This is all left up to the module for flexibility. The only limitation is that
prefData is only 512 bytes, and therefore no more than 512 bytes of information can be stored in prefData.

The blank() function will be called and sent the bMessage pointer so that it
can extract the additional information of screen depth and screen mode. These
are the modes that the user selects in the main Blanker panel. Please try to
use this information intelligently and open the closest screen mode that your
module supports. These modes will span the range of original/ECS/AGA screen
modes so be prepared for this contingency. However, if it doesn't make sense
to use more than one bitplane for example, then you have the option to ignore
that field. Also, I leave the overscan settings to the module. So that can be
set to whatever seems fitting for your module. The blank() function has to be
a bit more intelligent than the other functions. It can blank in a repeating
loop and should quit when it is sent the SIGBREAKF_CTRL_C signal. The best way
to understand how that is done is to simply look at the main loops in the
supplied example blanking functions. It is a fairly simple method and works
quite well. After the CTRL_C is received, you must close down any opened resources and return.

That's about all the help I think you should need in writing your own
blankers. Like I said before, I think the best thing to do is to have a look
at the existing blanker modules and fill in your own specific information
therein. I will also make the suggestion that using GadToolsBox for your preferences interface allows you to very easily create a preferences panel.

1.10 Credits

First, let me express my extreme gratitude to Nico François for writing
his wonderful ReqTools library. It makes it very easy to have a complete
professional look without having to write a great deal of "requester"
code. I applaud the quality of the ReqTools package.

Second, I also express gratitude to Jan van den Baard for writing his
amazing GadToolsBox interface toolkit. I also 'borrowed' his disclaimer
because it sounded nice and disclaiming.

Author Information:

Michael D. Bayne, Sophomore CS Student at Rose-Hulman Institute of Tech.

E-Mail address:

baynemd@nextwork.rose-hulman.edu (bug reports, etc. here)

Mailing address:

Box 216 (mail bombs, etc. here)
Rose-Hulman Institute of Technology
5500 Wabash Ave.
Terre Haute, IN 47803

1.11 Plans

Why to write more modules of course!

Additionally, I plan to implement an AREXX port. Mostly I will do this to appease a particular AREXX freak who beta tests my program, however I could conceive of some use for an AREXX input to a screen blanker and I will certainly make an AREXX module because the uses for that are obvious and diverse.

1.12 History

Version 1.0 -- Initial public release of Blanker

Version 1.1 -- Fixed bug in Moire module X-Speed gadget
-- Main window wasn't 'locked' when pref window was open
(ie. busy pointer, no mouse input), fixed it.

Version 2.0 -- Implemented modules and modularity.
Wrote the Text/Clock module.

Version 2.1 -- Replaced CHECKBOX_KIND Resident button with CYCLE_KIND
because GadTools.library v39 is so 'bug free' that the
checkbox gadget caused enforcer hits and wouldn't render.
Separated the Text/Clock module into a Text module and a
Clock module and added more features to each.

Version 2.2 -- Blanked the mouse as well as the screen for people who
aren't using a separate mouse blanker.

Version 2.3 -- Removed mouse blanking code because system friendly mouse
blanking for all versions of the OS is a kludge, and its
better for each person to use a standalone mouseblanker
(QMouse) that suites their needs.
Added Copper list options to the existing modules.
Removed delays before blanking after discovery of 'Upstroke'
keyword to commodities HotKey objects.

1.13 F.A.Q.

Q: The mouse gets really screwed up after the screen blanks. What's the deal?

A: I was blanking the mouse by clearing the pointer sprite. That works under 2.04 but not 3.0. I took out the mouseblanking code because its much easier and more convenient for you to use a different mouseblanking commodity along with Blanker.

Q: The mouse doesn't blank when the screen blanks. Why not?

A: See above question.

Q: The blanker seems 'jittery'. It unblanks right after I press the blank key. How can I fix this?

A: The problem is that the blanker is catching the key-up event of you releasing the key because the screen blanker is already blanking by the time you release the key. Simply put the word 'Upstroke' before the word for the key in your 'Blank Key' field.
Examples:
Alt Upstroke Help
Ctrl Shift Upstroke B
This will cause the blanker to blank on the upstroke of your keypress and that problem will go away.