

Programmer

David Swasbrook

Copyright © Copyright(C)1994 David Swasbrook. All Rights Reserved.

COLLABORATORS

	<i>TITLE :</i> Programmer		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	David Swasbrook	November 23, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	Programmer	1
1.1	Programmer (23.01.95)	1
1.2	BLANK	2
1.3	PREFS	3
1.4	Blanker ID String	4
1.5	Programming for locale...	5
1.6	index	5

Chapter 1

Programmer

1.1 Programmer (23.01.95)

**** This is a bit out of date (sorry)

SWAZBLANKER : Programming References Contents
=====

(C) 1992,93,94 David Swasbrook,
All Rights Reserved.

So you want to write a blanker module eh?

If you wish to look at some example source please feel free to have a look at the code provided. See [Example Blanker Source](#) .

Its pretty simple, there are a few guidelines however that should be followed:

- o Standard AmigaDos executable program
- o Exit on a SIGBREAK_CTRL_C signal
- o You must include in your program a blanker identification string . If this is omitted then SwazBlanker will not recognise it as valid module.
**** NOTE: Changed slightly. See example source.

- o Accept the following command line parameters:

BLANK - perform the blanking of the screen

PREFS - configure the blanker.

**** NOTE: This has changed, please see example source on using the SB_Init() function...

- o Tell SwazBlanker your screen and window. This enables SwazBlanker to perform Reblanking for your blanker.
SB_SetBlankerScreen(screen, window).
- o You must tell SwazBlanker your blanker is ready by calling the function SB_BlankerReady(). Otherwise SwazBlanker will wait until your blanker

exits.

- o It is also recommended (especially for modules with data files) that you change your current directory to the blanker module directory. The function SB_GotoBlankerHomeDir() and SB_ReturnBlankerHomeDir() are designed for this purpose.
- o Do not change your task priority .
- o You may also wish to localize your interface.
- o And please, please put some Waits in, we cant have a blanker taking all the available CPU time. There are other tasks running you know!
- o The preferences interface must quit upon receiving a SIGBREAKF_ABORT.
- o The preferences task should be registered with swazblanker.library using SB_AddPrefsTaskTagList().

Once all the above have been completed you have a finished blanker. yay!

1.2 BLANK

CLI Argument : BLANK

The screen and mouse pointer should be blanked.

To simply blank the screen all you need to do is open a one bitplane screen and set the two colors in the palette to black.

```
struct Screen *screen;
WORD CSpec[] = { 0,0,0,0, 1,0,0,0 -1 };

screen = SB_OpenScreenTags(
    SA_DisplayID,0,
    SA_Depth,1,
    SA_Colors, &CSpec,
    TAG_DONE);
```

To blank the mouse pointer you need to open a window on the screen and set up a blank pointer.

```
struct Window *window;

window = SB_OpenWindowTags(
    WA_CustomScreen,screen,
    TAG_DONE);
```

Once the screen and mouse pointer have been blanked you then setup anything else your blanker needs and then inform SwazBlanker that you are ready :

```
SB_SetBlankerScreen( screen, window ); /* Set the blankers screen and window ↵
*/
SB_BlankerReady(); /* Tell SwazBlanker we are working */
```

Then you can do whatever you like, SwazBlanker will make sure that your screen and window remain active. When it is time to unblank SwazBlanker immediately pushes your screen to the back and sends the blanker the signal SIGBREAKF_CTRL_C.

```
Wait (SIGBREAKF_CTRL_C);
```

When your blanker receives this signal it needs to quit.

```
SB_ClrBlankerScreen( screen, window ); /* Restore old screen */
```

1.3 PREFS

CLI Argument : PREFS

Configuring the blanker.

All blankers must have a configuration interface. If your blanker does not require configuration then you should inform the user that no configuration is required for your blanker.

Since under the Amiga's multitasking environment it is possible to request to configure the same blanker module several times, SwazBlanker library provides a simple function you can call to register your blankers interface so that only one configuration invocation of your blanker can exist.

Your blanker preferences routine should look something like the following:

```
void BlankerPreferences(void)
{
    struct BlankerPrefsNode *bpn;

    if( bpn = SB_AddPrefsTaskTagList( "Blanker Name", NULL ) )
    {
        /*
        ** Put your interface in here
        **
        ** Please set (struct Window *) bpn->window as soon
        ** as possible.
        */

        SB_RemPrefsTask( bpn );
    }
}
```

It is suggested that the configuration window for the blanker use the same font as chosen in blanker server preferences. This may be easily obtained from the SwazBlankerBase as:

```
(STRPTR) SwazBlankerBase->InterfaceFontName
```

This is a pointer to a string describing the font. Eg "Helvetica 11".

1.4 Blanker ID String

Blanker Identification String

This text string must be included in your program executable somewhere. Without this the blanker module will not be recognised as a valid module by SwazBlanker.

FORMAT

```
"$BLANKER:TITLE/K/A,AUTHOR/K/A,EMAIL/K,STACK/K/N,CPU/K/N,KS/K/N,LOAD/K, ←
CYEARS/K,COPYRIGHT/K,INFO/F"
```

ARGUMENTS

TITLE="<string>"

Breif description of the blanker module.

AUTHOR="<string>"

Name of the author of the blanker module.

EMAIL="<string>"

Optional email address of the author.

STACK=<number>

Some blankers may require large stacks, the default stack size for a module run by SwazBlanker is 4096 bytes. If your module requires large amounts of stack, you may specify the stack required which will override the default stack size set by SwazBlanker.

Eg. STACK=16384 (Allocate 16Kb of stack for the blanker).

CPU=<number>

This is the CPU that the blanker module is designed for.

Eg. CPU=68000 (Blanker requires a 68000 CPU or better)

CPU=68040 (Blanker requires a 68040 CPU or better)

Currently SwazBlanker does not look at this field, but it is intended to be used in the future.

KS=<number>

This is the minimum kickstart required to run the blanker.

Eg. KS=40 (Blanker requires kickstart V40 or higher)

Currently SwazBlanker does not look at this field, but it is intended to be used in the future.

LOAD="<string>"

Rough estimate of the CPU load the blanker module uses.

Eg. LOAD=Low or LOAD=High.

Values for LOAD are None, Low, Medium, High

CYEARS="<string>"

Copyright dates for the blanker module.

COPYRIGHT="<string>"

Alternative copyright message.

INFO=<rest of line>

The rest of the line contains a detailed description of what the blanker does and any other information you feel is important for the user to know. You may use the sequence ^M for a linefeed.

NOTE: You only need quotes around strings when

CPU=NUMBER - This is the minimum CPU required to run the blanker.

Example:

```
extern UBYTE ID[] = "$BLANKER:"
" CPU=68000"
" KS=39"
" LOAD=None"
" STACK=4096"
" AUTHOR=\"Example\" "
" EMAIL=\"example@example.co\" "
" TITLE=\"Example blanker module\" "
" CYEARS=\"1994\" "
"Just some example source."
"\n"; /* You MUST remember to put the linefeed at the end! */
```

1.5 Progrrmming for locale...

Locale

It is suggested that you localize the configuartion interface for your blanker. Please consult your system programming style guidelines for further information.

1.6 index

SWAZBLANKER : Programming References Index

=====

(C) 1992,93,94 David Swasbrook,
All Rights Reserved.

Programming Info

Programming A Blanker	Programming documentation
Blanker Identification String	Identifies the executable as a blanker

CLI Arg : BLANK
CLI Arg : PREFS
Locale

What to do when you get a BLANK argument
What to do when you get a PREFS argument
Programming for locale

Source

SwazBlankerBase
Example Blanker C Source

SwazBlanker.h
ExampleBlanker.c

SwazBlanker

Blanker Server Documentation
Reblanking
Task Priority

Documentation
Reblank when blanker not visible
Task priority management