

ScrBuffer

COLLABORATORS

	<i>TITLE :</i> ScrBuffer		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		December 8, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	ScrBuffer	1
1.1	ScrBuffer.guide	1
1.2	author	1
1.3	sb_openscreen	2
1.4	sb_closescreen	3
1.5	sb_nextbuffer	3
1.6	sb_getbitmap	4
1.7	sb_getscreen	4
1.8	Information about the examples	4

Chapter 1

ScrBuffer

1.1 ScrBuffer.guide

Filled Vector Module for AmigaE 3.0+

© 1994 Michael Zucchi
All Rights Reserved

Not to be used for commercial software without express written permission from the author.

This document describes the functions available for creating and working with Workbench 3.0+ double buffered screen routines. They provide a general purpose high level suite of routines for accessing these system routines that can be used for any double buffering system.

Functions currently available are:

sb_OpenScreen()	to open a screen
sb_CloseScreen()	to close a screen
sb_NextBuffer()	waits for the next buffer to be displayed
sb_GetBitMap()	gives hidden bitmap pointer
sb_GetScreen()	gives base screen pointer

examples how to use it

Remember, use of this module requires that V39 libraries are present in the target system! Never use these functions without making sure this is the case.

1.2 author

I'm another one of these poor students, but i live in Australia, not Europe!

Presently, i study 'from time to time' :-) in order to obtain a Computer Systems Engineering degree from the Univerity Of South Australia. Now less than 6 months to go (at last!)

Other than this and other AmigaE modules, i've coded a couple of demos for FRONTIER as the coder 'Zed', along with some small utilities, esp ZGif.

I can be contacted in the following ways:

Internet email:

9107047w@lux.levels.unisa.edu.au
till the end of '94 at least - reliable

zucchi@hal9000.apana.org.au
if it works it works ...

'Real Mode' (tm) mail:

Michael Zucchi
PO BOX 824
Waikerie
South Australia 5330
slow, but very reliable

Michael Zucchi
110 Dunrobin Rd
Warradale
South Australia 5046
to my door - till i move (?)

1.3 sb_openscreen

scrbuffer.m/sb_OpenScreen

scrbuffer.m/sb_OpenScreen

SYNTAX

buffered screen := sb_OpenScreen (tags, type)

PURPOSE

To open multibuffered screen. Also allocates at least 1 more buffer to 'multibuffer' and a message port to receive notification events.

INPUTS

tags list of tags as found in "intuition/screens" that describe the screen you wish to open. All the usual tags such as SA_WIDTH, SA_HEIGHT, SA_DEPTH etc are used.
type the type of buffered screen to open. Only use 0 for now, may be enhanced in future to add multi-buffering.

OUTPUTS

buffered screen
a pointer to the buffered screen, or 0 if the screen or asociate memory could not be allocated. All fields of this structure are PRIVATE.

SEE ALSO
`sb_CloseScreen()`, `sb_NextBuffer()`, `sb_GetBitMap()`, `sb_GetScreen()`

1.4 sb_closescreen

`scrbuffer.m/sb_CloseScreen`

`scrbuffer.m/sb_CloseScreen`

SYNTAX

`sb_CloseScreen (buffered screen)`

PURPOSE

Closes a buffered screen. All associated resources (buffers, message ports etc) are returned to the system.

INPUTS

`screen` a buffered screen handle, as returned from `sb_OpenScreen()`
NOT a standard Amiga screen!

SEE ALSO
`sb_OpenScreen()`

1.5 sb_nextbuffer

`scrbuffer.m/sb_NextBuffer`

`scrbuffer.m/sb_NextBuffer`

SYNTAX

`bitmap = sb_NextBuffer(buffered screen)`

PURPOSE

Ask the operating system to change the bitmap of the screen associated with the double buffered screen, and wait for it to happen so that the other screen is hidden, ready for rendering. It does this by calling `ChangeScreenBuffer()` from the V39 Intuition library, then waiting for the reply message.

INPUTS

`screen` a buffered screen handle, as returned from `sb_OpenScreen()`

OUTPUTS

`bitmap` a standard Amiga bitmap, that represents the currently HIDDEN display area. This will usually need to be cleared and can then be rendered into.

SEE ALSO
`sb_OpenScreen()`, `sb_NextBuffer()`, `sb_GetBitMap()`, `sb_GetScreen()`

1.6 sb_getbitmap

scrbuffer.m/sb_GetBitMap

scrbuffer.m/sb_GetBitMap

SYNTAX

```
bitmap = sb_GetBitMap( buffered screen )
```

PURPOSE

Returns the currently hidden bitmap. This will be identical to that returned by `sb_NextBuffer()`.

INPUTS

`screen` a buffered screen handle, as returned from `sb_OpenScreen()`

OUTPUTS

`bitmap` a standard amiga bitmap, that represents the currently HIDDEN display area.

SEE ALSO

`sb_NextBuffer()`, `sb_OpenScreen()`, `sb_GetBitMap()`, `sb_GetScreen()`

1.7 sb_getscreen

scrbuffer.m/sb_GetScreen

scrbuffer.m/sb_GetScreen

SYNTAX

```
screen = sb_GetScreen( buffered screen )
```

PURPOSE

Returns the Amiga screen associated with the double buffered screen. This is to enable windows and other standard intuition operations to be performed on the screen.

INPUTS

`screen` a buffered screen handle, as returned from `sb_OpenScreen()`

OUTPUTS

`screen` a standard Amiga Screen

SEE ALSO

`sb_NextBuffer()`, `sb_OpenScreen()`, `sb_GetBitMap()`

1.8 Information about the examples

Only 1 example has been written so far.

TheBOX

A simple example that shows how to open a simple double buffered screen, and render into the offscreen bitmap using the system functions. It also shows how a window can be opened on the screen to get input events.