# SDL Console Reference Manual

Generated by Doxygen 1.3.2

Sun Aug 31 14:24:32 2003

# Contents

**1 SDL Console Main Page** **1**

1.1 Introduction . . . . . 1

1.2 Documentation . . . . . 1

1.3 Keyboard Reference . . . . . 1

**2 SDL Console Data Structure Index** **3**

2.1 SDL Console Data Structures . . . . . 3

**3 SDL Console File Index** **5**

3.1 SDL Console File List . . . . . 5

**4 SDL Console Data Structure Documentation** **7**

4.1 BitFont_td Struct Reference . . . . . 7

4.2 console_information_td Struct Reference . . . . . 9

**5 SDL Console File Documentation** **15**

5.1 documentation.h File Reference . . . . . 15

5.2 DT_drawtext.h File Reference . . . . . 16

5.3 internal.h File Reference . . . . . 17

5.4 SDL_console.h File Reference . . . . . 18

# Chapter 1

# SDL Console Main Page

## 1.1 Introduction

SDL_Console is a console that can be added to any SDL application. It is similar to Quake and other games consoles. A console is meant to be a very simple way of interacting with a program and executing commands. You can also have more than one console at a time.

## 1.2 Documentation

For a detailed description of all functions see SDL_console::h. Remark that functions that have the mark ”Internal” are only used internally. There’s not much use of calling these functions.

## 1.3 Keyboard Reference

| | |
|---|---|
| Up | move command history up |
| Down | move command history down |
| Left | move cursor left |
| Right | move cursor right |
| Ins | toggle overwrite mode (you can set the two different cursor in SDL_console.h) |
| Del | delete character above cursor |
| Backspace | delete character left of cursor |
| Home | move cursor to begin of command |
| End | move cursor to end of command |
| Ctrl-A | same as Home |
| Ctrl-E | same as End |
| Ctrl-C | clear commandline |
| Page-Up | move history up |
| Page-Down | move history down |
| Shift-Home | move to top of history |
| Shift-End | move to end of history |
| Ctrl-L | clear history |

Have Fun!

**Author:**

Garett Banuk <`mongoose@mongeese.org`> (Original Version)
Clemens Wacha <`reflex-2000@gmx.net`> (Version 2.x, Documentation)
Boris Lesner <`talanthyr@tuxfamily.org`> (Package Maintainer)

# Chapter 2

# SDL Console Data Structure Index

## 2.1 SDL Console Data Structures

Here are the data structures with brief descriptions:

BitFont_td . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
console_information_td . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

# Chapter 3

# SDL Console File Index

## 3.1 SDL Console File List

Here is a list of all files with brief descriptions:

documentation.h . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
DT_drawtext.h . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16
internal.h . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
SDL_console.h . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

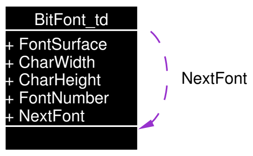
# Chapter 4

# SDL Console Data Structure Documentation

## 4.1 BitFont_td Struct Reference

`#include <DT_drawtext.h>`

Collaboration diagram for BitFont_td:



### Data Fields

- SDL_Surface * FontSurface
- int CharWidth
- int CharHeight
- int FontNumber
- BitFont_td * NextFont

### 4.1.1 Field Documentation

#### 4.1.1.1 SDL_Surface* BitFont_td::FontSurface

#### 4.1.1.2 int BitFont_td::CharWidth

#### 4.1.1.3 int BitFont_td::CharHeight

#### 4.1.1.4 int BitFont_td::FontNumber

#### 4.1.1.5 struct BitFont_td* BitFont_td::NextFont

The documentation for this struct was generated from the following file:

- DT_drawtext.h

## 4.2 console_information_td Struct Reference

```
#include <SDL_console.h>
```

### 4.2.1 Detailed Description

This is a struct for each consoles data

### Data Fields

- int Visible
- int WasUnicode
- int RaiseOffset
- int HideKey
- char ∗∗ ConsoleLines
- char ∗∗ CommandLines
- int TotalConsoleLines
- int ConsoleScrollBack
- int TotalCommands
- int FontNumber
- int LineBuffer
- int VChars
- int BackX
- int BackY
- char ∗ Prompt
- char Command [CON_CHARS_PER_LINE+1]
- char RCommand [CON_CHARS_PER_LINE+1]
- char LCommand [CON_CHARS_PER_LINE+1]
- char VCommand [CON_CHARS_PER_LINE+1]
- int CursorPos
- int Offset
- int InsMode
- SDL_Surface ∗ ConsoleSurface
- SDL_Surface ∗ OutputScreen
- SDL_Surface ∗ BackgroundImage
- SDL_Surface ∗ InputBackground
- int DispX
- int DispY
- unsigned char ConsoleAlpha
- int CommandScrollBack
- void(∗ CmdFunction )(struct console_information_td ∗console, char ∗command)
- char ∗(∗ TabFunction )(char ∗command)
- int FontHeight
- int FontWidth

### 4.2.2 Field Documentation

#### 4.2.2.1 int console_information_td::Visible

#### 4.2.2.2 int console_information_td::WasUnicode

enum that tells which visible state we are in CON_CLOSED, CON_OPEN, CON_CLOSING, CON_-OPENING

#### 4.2.2.3 int console_information_td::RaiseOffset

stores the UNICODE value before the console was shown. On Hide() the UNICODE value is restored.

#### 4.2.2.4 int console_information_td::HideKey

Offset used in show/hide animation

#### 4.2.2.5 char** console_information_td::ConsoleLines

the key that can hide the console

#### 4.2.2.6 char** console_information_td::CommandLines

List of all the past lines

#### 4.2.2.7 int console_information_td::TotalConsoleLines

List of all the past commands

#### 4.2.2.8 int console_information_td::ConsoleScrollBack

Total number of lines in the console

#### 4.2.2.9 int console_information_td::TotalCommands

How much the user scrolled back in the console

#### 4.2.2.10 int console_information_td::FontNumber

Number of commands that were typed in before (which are now in the CommandLines array)

#### 4.2.2.11 int console_information_td::LineBuffer

This is the number of the font for the console (DT_* specific; will hopefully disappear in future releases)

#### 4.2.2.12 int console_information_td::VChars

The number of visible lines in the console (autocalculated on CON_UpdateConsole())

#### 4.2.2.13 int console_information_td::BackX

The number of visible characters in one console line (autocalculated on CON_Init() and recalc. on CON_-Resize())

#### 4.2.2.14 int console_information_td::BackY

The number of visible characters in one console line (autocalculated on CON_Init() and recalc. on CON_-Resize())

#### 4.2.2.15 char∗ console_information_td::Prompt

Background image x and y coords

#### 4.2.2.16 char console_information_td::Command[CON_CHARS_PER_LINE+1]

Prompt displayed in command line

#### 4.2.2.17 char console_information_td::RCommand[CON_CHARS_PER_LINE+1]

current command in command line = lcommand + rcommand (Get's updated in AssembleCommand())

#### 4.2.2.18 char console_information_td::LCommand[CON_CHARS_PER_LINE+1]

left hand side of cursor

#### 4.2.2.19 char console_information_td::VCommand[CON_CHARS_PER_LINE+1]

right hand side of cursor

#### 4.2.2.20 int console_information_td::CursorPos

current visible command line

#### 4.2.2.21 int console_information_td::Offset

Current cursor position relative to the currently typed in command

#### 4.2.2.22 int console_information_td::InsMode

First visible character relative to the currently typed in command (used if command is too long to fit into console)

#### 4.2.2.23 SDL_Surface∗ console_information_td::ConsoleSurface

Boolean that tells us whether we are in Insert- or Overwrite-Mode

#### 4.2.2.24 SDL_Surface∗ console_information_td::OutputScreen

THE Surface of the console

#### 4.2.2.25 SDL_Surface∗ console_information_td::BackgroundImage

This is the screen to draw the console to (normally you VideoSurface)

#### 4.2.2.26 SDL_Surface∗ console_information_td::InputBackground

Background image for the console

#### 4.2.2.27 int console_information_td::DispX

Dirty rectangle that holds the part of the background image that is behind the commandline

#### 4.2.2.28 int console_information_td::DispY

Dirty rectangle that holds the part of the background image that is behind the commandline

#### 4.2.2.29 unsigned char console_information_td::ConsoleAlpha

The top left x and y coords of the console on the display screen

#### 4.2.2.30 int console_information_td::CommandScrollBack

The consoles alpha level

#### 4.2.2.31 void(∗ console_information_td::CmdFunction)(struct console_information_td ∗console, char∗ command)

How much the users scrolled back in the command lines

#### 4.2.2.32 char∗(∗ console_information_td::TabFunction)(char∗ command)

The Function that is executed if you press ’Return’ in the console

#### 4.2.2.33 int console_information_td::FontHeight

The Function that is executed if you press ’Tab’ in the console

#### 4.2.2.34 int console_information_td::FontWidth

The height of the font used in the console

The documentation for this struct was generated from the following file:

- SDL_console.h

# Chapter 5

# SDL Console File Documentation

## 5.1 documentation.h File Reference

## 5.2 DT_drawtext.h File Reference

### Data Structures

- struct BitFont_td

### Defines

- #define TRANS_FONT 1

### Typedefs

- typedef BitFont_td BitFont

### Functions

- void DT_DrawText (const char *string, SDL_Surface *surface, int FontType, int x, int y)
- int DT_LoadFont (const char *BitmapName, int flags)
- int DT_FontHeight (int FontNumber)
- int DT_FontWidth (int FontNumber)
- BitFont * DT_FontPointer (int FontNumber)
- void DT_DestroyDrawText ()

### 5.2.1 Define Documentation

#### 5.2.1.1 #define TRANS_FONT 1

### 5.2.2 Typedef Documentation

#### 5.2.2.1 typedef struct BitFont_td BitFont

### 5.2.3 Function Documentation

#### 5.2.3.1 void DT_DrawText (const char * *string*, SDL_Surface * *surface*, int *FontType*, int *x*, int *y*)

#### 5.2.3.2 int DT_LoadFont (const char * *BitmapName*, int *flags*)

#### 5.2.3.3 int DT_FontHeight (int *FontNumber*)

#### 5.2.3.4 int DT_FontWidth (int *FontNumber*)

#### 5.2.3.5 BitFont* DT_FontPointer (int *FontNumber*)

#### 5.2.3.6 void DT_DestroyDrawText ()

## 5.3 internal.h File Reference

### Defines

- #define PRINT_ERROR(X) fprintf(stderr, "ERROR in %s:%s(): %s\n", __FILE__, __FUNCTION__, X)

### Functions

- Uint32 DT_GetPixel (SDL_Surface ∗surface, int x, int y)
- void DT_PutPixel (SDL_Surface ∗surface, int x, int y, Uint32 pixel)

### 5.3.1 Define Documentation

#### 5.3.1.1 #define PRINT_ERROR(X) fprintf(stderr, "ERROR in %s:%s(): %s\n", __FILE__, __FUNCTION__, X)

### 5.3.2 Function Documentation

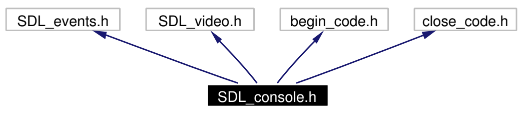
#### 5.3.2.1 Uint32 DT_GetPixel (SDL_Surface ∗ *surface*, int *x*, int *y*)

#### 5.3.2.2 void DT_PutPixel (SDL_Surface ∗ *surface*, int *x*, int *y*, Uint32 *pixel*)

## 5.4 SDL_console.h File Reference

```
#include "SDL_events.h"
#include "SDL_video.h"
#include "begin_code.h"
#include "close_code.h"
```

Include dependency graph for SDL_console.h:



### Data Structures

- struct console_information_td

### Defines

- #define CON_CHARS_PER_LINE 127
- #define CON_BLINK_RATE 500
- #define CON_CHAR_BORDER 4
- #define CON_DEFAULT_PROMPT ”]”
- #define CON_LINE_SCROLL 2
- #define CON_SCROLL_INDICATOR ”^”
- #define CON_INS_CURSOR ”_”
- #define CON_OVR_CURSOR ”|”
- #define CON_DEFAULT_HIDEKEY SDLK_ESCAPE
- #define CON_OPENCLOSE_SPEED 25

### Typedefs

- typedef console_information_td ConsoleInformation

### Enumerations

- enum { CON_CLOSED, CON_CLOSING, CON_OPENING, CON_OPEN }

### Functions

- DECLSPEC SDL_Event ∗SDLCALL CON_Events (SDL_Event ∗event)
- DECLSPEC void SDLCALL CON_Show (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_Hide (ConsoleInformation ∗console)
- DECLSPEC int SDLCALL CON_isVisible (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_UpdateOffset (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_DrawConsole (ConsoleInformation ∗console)

- DECLSPEC ConsoleInformation ∗SDLCALL CON_Init (const char ∗FontName, SDL_Surface ∗DisplayScreen, int lines, SDL_Rect rect)
- DECLSPEC void SDLCALL CON_Destroy (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_Free (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_Out (ConsoleInformation ∗console, const char ∗str,...)
- DECLSPEC void SDLCALL CON_Alpha (ConsoleInformation ∗console, unsigned char alpha)
- DECLSPEC void SDLCALL CON_AlphaGL (SDL_Surface ∗s, int alpha)
- DECLSPEC int SDLCALL CON_Background (ConsoleInformation ∗console, const char ∗image, int x, int y)
- DECLSPEC void SDLCALL CON_Position (ConsoleInformation ∗console, int x, int y)
- DECLSPEC int SDLCALL CON_Resize (ConsoleInformation ∗console, SDL_Rect rect)
- DECLSPEC int SDLCALL CON_Transfer (ConsoleInformation ∗console, SDL_Surface ∗new_-outputscreen, SDL_Rect rect)
- DECLSPEC void SDLCALL CON_Topmost (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_SetPrompt (ConsoleInformation ∗console, char ∗newprompt)
- DECLSPEC void SDLCALL CON_SetHideKey (ConsoleInformation ∗console, int key)
- DECLSPEC void SDLCALL CON_Execute (ConsoleInformation ∗console, char ∗command)
- DECLSPEC void SDLCALL CON_SetExecuteFunction (ConsoleInformation ∗console, void(∗CmdFunction)(ConsoleInformation ∗console2, char ∗command))
- DECLSPEC void SDLCALL CON_SetTabCompletion (ConsoleInformation ∗console, char ∗(∗Tab-Function)(char ∗command))
- DECLSPEC void SDLCALL CON_TabCompletion (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_NewLineConsole (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_NewLineCommand (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL CON_UpdateConsole (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Default_CmdFunction (ConsoleInformation ∗console, char ∗command)
- DECLSPEC char ∗SDLCALL Default_TabFunction (char ∗command)
- DECLSPEC void SDLCALL DrawCommandLine ()
- DECLSPEC void SDLCALL Cursor_Left (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Cursor_Right (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Cursor_Home (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Cursor_End (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Cursor_Del (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Cursor_BSpace (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Cursor_Add (ConsoleInformation ∗console, SDL_Event ∗event)
- DECLSPEC void SDLCALL Clear_Command (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Assemble_Command (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Clear_History (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Command_Up (ConsoleInformation ∗console)
- DECLSPEC void SDLCALL Command_Down (ConsoleInformation ∗console)

### 5.4.1 Define Documentation

#### 5.4.1.1 #define CON_CHARS_PER_LINE 127

Number of visible characters in a line. Lines in the history, the commandline, or CON_Out strings cannot be longer than this. Remark that this number does NOT include the '/0' character at the end of a string. So if we create a string we do this char∗ mystring[CON_CHARS_PER_LINE + 1];

#### 5.4.1.2 #define CON_BLINK_RATE 500

Cursor blink frequency in ms

#### 5.4.1.3 #define CON_CHAR_BORDER 4

Border in pixels from the left margin to the first letter

#### 5.4.1.4 #define CON_DEFAULT_PROMPT "]"

Default prompt used at the commandline

#### 5.4.1.5 #define CON_LINE_SCROLL 2

Scroll this many lines at a time (when pressing PGUP or PGDOWN)

#### 5.4.1.6 #define CON_SCROLL_INDICATOR "^"

Indicator showing that you scrolled up the history

#### 5.4.1.7 #define CON_INS_CURSOR "_"

Cursor shown if we are in insert mode

#### 5.4.1.8 #define CON_OVR_CURSOR "|"

Cursor shown if we are in overwrite mode

#### 5.4.1.9 #define CON_DEFAULT_HIDEKEY SDLK_ESCAPE

Defines the default hide key (that Hide()'s the console if pressed)

#### 5.4.1.10 #define CON_OPENCLOSE_SPEED 25

Defines the opening/closing speed when the console switches from CON_CLOSED to CON_OPEN

## 5.4.2 Typedef Documentation

#### 5.4.2.1 typedef struct console_information_td ConsoleInformation

This is a struct for each consoles data

### 5.4.3 Enumeration Type Documentation

#### 5.4.3.1 anonymous enum

**Enumeration values:**

**CON_CLOSED**

**CON_CLOSING** The console is closed (and not shown)

**CON_OPENING** The console is still open and visible but closing. After it has completely disappeared it changes to CON_CLOSED

**CON_OPEN** The console is open and visible

### 5.4.4 Function Documentation

#### 5.4.4.1 DECLSPEC SDL_Event∗ SDLCALL CON_Events (SDL_Event ∗ *event*)

Takes keys from the keyboard and inputs them to the console if the console isVisible(). If the event was not handled (i.e. WM events or unknown ctrl- or alt-sequences) the function returns the event for further processing. ∗∗∗The prototype of this function will change in the next major release to int CON_-Events(ConsoleInformation∗ console, SDL_Event ∗event) ∗∗

#### 5.4.4.2 DECLSPEC void SDLCALL CON_Show (ConsoleInformation ∗ *console*)

Makes the console visible

#### 5.4.4.3 DECLSPEC void SDLCALL CON_Hide (ConsoleInformation ∗ *console*)

Hides the console

#### 5.4.4.4 DECLSPEC int SDLCALL CON_isVisible (ConsoleInformation ∗ *console*)

Returns 1 if the console is opening or open, 0 else

#### 5.4.4.5 DECLSPEC void SDLCALL CON_UpdateOffset (ConsoleInformation ∗ *console*)

Internal: Updates visible state. This function is responsible for the opening/closing animation. Only used in CON_DrawConsole()

#### 5.4.4.6 DECLSPEC void SDLCALL CON_DrawConsole (ConsoleInformation ∗ *console*)

Draws the console to the screen if it is visible (NOT if it isVisible()). It get's drawn if it is REALLY visible ;-)

#### 5.4.4.7 DECLSPEC ConsoleInformation∗ SDLCALL CON_Init (const char ∗ *FontName*, SDL_Surface ∗ *DisplayScreen*, int *lines*, SDL_Rect *rect*)

Initializes a new console.

**Parameters:**

***FontName*** A filename of an image containing the font. Look at the example code for the image contents

***DisplayScreen*** The VideoSurface we are blitting to. ∗∗∗This was not a very intelligent move. I will change this in the next major release. CON_DrawConsole will then no more blit the console to this surface but give you a pointer to ConsoleSurface when all updates are done∗∗∗

***lines*** The total number of lines in the history

***rect*** Position and size of the new console

#### 5.4.4.8 DECLSPEC void SDLCALL CON_Destroy (ConsoleInformation ∗ *console*)

Frees DT_DrawText and calls CON_Free

#### 5.4.4.9 DECLSPEC void SDLCALL CON_Free (ConsoleInformation ∗ *console*)

Frees all the memory loaded by the console

#### 5.4.4.10 DECLSPEC void SDLCALL CON_Out (ConsoleInformation ∗ *console*, const char ∗ *str*, ...)

Function to send text to the console. Works exactly like printf and supports the same format

#### 5.4.4.11 DECLSPEC void SDLCALL CON_Alpha (ConsoleInformation ∗ *console*, unsigned char *alpha*)

Sets the alpha level of the console to the specified value (0 - transparent, 255 - opaque). Use this function also for OpenGL.

#### 5.4.4.12 DECLSPEC void SDLCALL CON_AlphaGL (SDL_Surface ∗ *s*, int *alpha*)

Internal: Sets the alpha channel of an SDL_Surface to the specified value. Preconditions: the surface in question is RGBA. 0 <= a <= 255, where 0 is transparent and 255 opaque

#### 5.4.4.13 DECLSPEC int SDLCALL CON_Background (ConsoleInformation ∗ *console*, const char ∗ *image*, int *x*, int *y*)

Sets a background image for the console

#### 5.4.4.14 DECLSPEC void SDLCALL CON_Position (ConsoleInformation ∗ *console*, int *x*, int *y*)

Changes current position of the console to the new given coordinates

#### 5.4.4.15 DECLSPEC int SDLCALL CON_Resize (ConsoleInformation ∗ *console*, SDL_Rect *rect*)

Changes the size of the console

#### 5.4.4.16 DECLSPEC int SDLCALL CON_Transfer (ConsoleInformation ∗ *console*, SDL_Surface ∗ *new_outputscreen*, SDL_Rect *rect*)

Beams a console to another screen surface. Needed if you want to make a Video restart in your program. This function first changes the OutputScreen Pointer then calls CON_Resize to adjust the new size. ∗∗∗Will disappear in the next major release. Instead i will introduce a new function called CON_ReInit or something that adjusts the internal parameters etc ∗∗∗

#### 5.4.4.17 DECLSPEC void SDLCALL CON_Topmost (ConsoleInformation ∗ *console*)

Give focus to a console. Make it the ”topmost” console. This console will receive events sent with CON_Events() ∗∗∗Will disappear in the next major release. There is no need for such a focus model ∗∗∗

#### 5.4.4.18 DECLSPEC void SDLCALL CON_SetPrompt (ConsoleInformation ∗ *console*, char ∗ *newprompt*)

Modify the prompt of the console. If you want a backslash you will have to escape it.

#### 5.4.4.19 DECLSPEC void SDLCALL CON_SetHideKey (ConsoleInformation ∗ *console*, int *key*)

Set the key, that invokes a CON_Hide() after press. default is ESCAPE and you can always hide using ESCAPE and the HideKey (2 keys for hiding). compared against event->key.keysym.sym !!

#### 5.4.4.20 DECLSPEC void SDLCALL CON_Execute (ConsoleInformation ∗ *console*, char ∗ *command*)

Internal: executes the command typed in at the console (called if you press ’Return’)

#### 5.4.4.21 DECLSPEC void SDLCALL CON_SetExecuteFunction (ConsoleInformation ∗ *console*, void(∗ *CmdFunction*)(ConsoleInformation ∗console2, char ∗command))

Sets the callback function that is called if a command was typed in. The function you would like to use as the callback will have to look like this:

**void my_command_handler(ConsoleInformation∗ console, char∗ command)**

You will then call the function like this:

**CON_SetExecuteFunction(console, my_command_handler)**

If this is not clear look at the example program

#### 5.4.4.22 DECLSPEC void SDLCALL CON_SetTabCompletion (ConsoleInformation ∗ *console*, char ∗(∗ *TabFunction*)(char ∗command))

Sets the callback function that is called if you press the ’Tab’ key. The function has to look like this:

**char∗ my_tabcompletion(char∗ command)**

The commandline on the left side of the cursor gets passed over to your function. You will then have to make your own tab-complete and return the completed string as return value. If you have nothing to complete you can return NULL or the string you got. ∗∗∗Will change in the next major release to char∗ mytabfunction(ConsoleInformation∗ console, char∗ command) ∗∗∗

#### 5.4.4.23 DECLSPEC void SDLCALL CON_TabCompletion (ConsoleInformation ∗ *console*)

Internal: Gets called when TAB was pressed and executes the function you have earlier registered with CON_SetTabCompletion()

#### 5.4.4.24 DECLSPEC void SDLCALL CON_NewLineConsole (ConsoleInformation ∗ *console*)

Internal: makes a newline (same as printf("\n") or CON_Out(console, "\n") )

#### 5.4.4.25 DECLSPEC void SDLCALL CON_NewLineCommand (ConsoleInformation ∗ *console*)

Internal: shift command history (the one you can switch with the up/down keys)

#### 5.4.4.26 DECLSPEC void SDLCALL CON_UpdateConsole (ConsoleInformation ∗ *console*)

Internal: updates console after resize, background image change, CON_Out() etc. This function draws the upper part of the console (that holds the history)

#### 5.4.4.27 DECLSPEC void SDLCALL Default_CmdFunction (ConsoleInformation ∗ *console*, char ∗ *command*)

Internal: Default Execute callback

#### 5.4.4.28 DECLSPEC char∗ SDLCALL Default_TabFunction (char ∗ *command*)

Internal: Default TabCompletion callback

#### 5.4.4.29 DECLSPEC void SDLCALL DrawCommandLine ()

Internal: draws the commandline the user is typing in to the screen. Called from within CON_DrawConsole() ∗∗∗ Will change in the next major release to void DrawCommandLine(ConsoleInformation∗ console) ∗∗∗

#### 5.4.4.30 DECLSPEC void SDLCALL Cursor_Left (ConsoleInformation ∗ *console*)

Internal: Gets called if you press the LEFT key (move cursor left)

#### 5.4.4.31 DECLSPEC void SDLCALL Cursor_Right (ConsoleInformation ∗ *console*)

Internal: Gets called if you press the RIGHT key (move cursor right)

#### 5.4.4.32 DECLSPEC void SDLCALL Cursor_Home (ConsoleInformation ∗ *console*)

Internal: Gets called if you press the HOME key (move cursor to the beginning of the line

#### 5.4.4.33 DECLSPEC void SDLCALL Cursor_End (ConsoleInformation $*$ *console*)

Internal: Gets called if you press the END key (move cursor to the end of the line

#### 5.4.4.34 DECLSPEC void SDLCALL Cursor_Del (ConsoleInformation $*$ *console*)

Internal: Called if you press DELETE (deletes character under the cursor)

#### 5.4.4.35 DECLSPEC void SDLCALL Cursor_BSpace (ConsoleInformation $*$ *console*)

Internal: Called if you press BACKSPACE (deletes character left of cursor)

#### 5.4.4.36 DECLSPEC void SDLCALL Cursor_Add (ConsoleInformation $*$ *console*, SDL_Event $*$ *event*)

Internal: Called if you type in a character (add the char to the command)

#### 5.4.4.37 DECLSPEC void SDLCALL Clear_Command (ConsoleInformation $*$ *console*)

Internal: Called if you press Ctrl-C (deletes the commandline)

#### 5.4.4.38 DECLSPEC void SDLCALL Assemble_Command (ConsoleInformation $*$ *console*)

Internal: Called if the command line has changed (assemles console->Command from LCommand and RCommand

#### 5.4.4.39 DECLSPEC void SDLCALL Clear_History (ConsoleInformation $*$ *console*)

Internal: Called if you press Ctrl-L (deletes the History)

#### 5.4.4.40 DECLSPEC void SDLCALL Command_Up (ConsoleInformation $*$ *console*)

Internal: Called if you press UP key (switches through recent typed in commands

#### 5.4.4.41 DECLSPEC void SDLCALL Command_Down (ConsoleInformation $*$ *console*)

Internal: Called if you press DOWN key (switches through recent typed in commands

# Index

Assemble_Command
    SDL_console.h, 25

BackgroundImage
    console_information_td, 12
BackX
    console_information_td, 11
BackY
    console_information_td, 11
BitFont
    DT_drawtext.h, 16
BitFont_td, 7
BitFont_td
    CharHeight, 8
    CharWidth, 8
    FontNumber, 8
    FontSurface, 8
    NextFont, 8

CharHeight
    BitFont_td, 8
CharWidth
    BitFont_td, 8
Clear_Command
    SDL_console.h, 25
Clear_History
    SDL_console.h, 25
CmdFunction
    console_information_td, 12
Command
    console_information_td, 11
Command_Down
    SDL_console.h, 25
Command_Up
    SDL_console.h, 25
CommandLines
    console_information_td, 10
CommandScrollBack
    console_information_td, 12
CON_Alpha
    SDL_console.h, 22
CON_AlphaGL
    SDL_console.h, 22
CON_Background
    SDL_console.h, 22
CON_BLINK_RATE
    SDL_console.h, 19
CON_CHAR_BORDER
    SDL_console.h, 20
CON_CHARS_PER_LINE
    SDL_console.h, 19
CON_CLOSED
    SDL_console.h, 21
CON_CLOSING
    SDL_console.h, 21
CON_DEFAULT_HIDEKEY
    SDL_console.h, 20
CON_DEFAULT_PROMPT
    SDL_console.h, 20
CON_Destroy
    SDL_console.h, 22
CON_DrawConsole
    SDL_console.h, 21
CON_Events
    SDL_console.h, 21
CON_Execute
    SDL_console.h, 23
CON_Free
    SDL_console.h, 22
CON_Hide
    SDL_console.h, 21
CON_Init
    SDL_console.h, 21
CON_INS_CURSOR
    SDL_console.h, 20
CON_isVisible
    SDL_console.h, 21
CON_LINE_SCROLL
    SDL_console.h, 20
CON_NewLineCommand
    SDL_console.h, 24
CON_NewLineConsole
    SDL_console.h, 24
CON_OPEN
    SDL_console.h, 21
CON_OPENCLOSE_SPEED
    SDL_console.h, 20
CON_OPENING
    SDL_console.h, 21
CON_Out

SDL_console.h, 22
CON_OVR_CURSOR
SDL_console.h, 20
CON_Position
SDL_console.h, 22
CON_Resize
SDL_console.h, 22
CON_SCROLL_INDICATOR
SDL_console.h, 20
CON_SetExecuteFunction
SDL_console.h, 23
CON_SetHideKey
SDL_console.h, 23
CON_SetPrompt
SDL_console.h, 23
CON_SetTabCompletion
SDL_console.h, 23
CON_Show
SDL_console.h, 21
CON_TabCompletion
SDL_console.h, 23
CON_Topmost
SDL_console.h, 23
CON_Transfer
SDL_console.h, 22
CON_UpdateConsole
SDL_console.h, 24
CON_UpdateOffset
SDL_console.h, 21
console_information_td, 9
BackgroundImage, 12
BackX, 11
BackY, 11
CmdFunction, 12
Command, 11
CommandLines, 10
CommandScrollBack, 12
ConsoleAlpha, 12
ConsoleLines, 10
ConsoleScrollBack, 10
ConsoleSurface, 11
CursorPos, 11
DispX, 12
DispY, 12
FontHeight, 12
FontNumber, 10
FontWidth, 12
HideKey, 10
InputBackground, 12
InsMode, 11
LCommand, 11
LineBuffer, 10
Offset, 11
OutputScreen, 12
Prompt, 11
RaiseOffset, 10
RCommand, 11
TabFunction, 12
TotalCommands, 10
TotalConsoleLines, 10
VChars, 10
VCommand, 11
Visible, 10
WasUnicode, 10
ConsoleAlpha
console_information_td, 12
ConsoleInformation
SDL_console.h, 20
ConsoleLines
console_information_td, 10
ConsoleScrollBack
console_information_td, 10
ConsoleSurface
console_information_td, 11
Cursor_Add
SDL_console.h, 25
Cursor_BSpace
SDL_console.h, 25
Cursor_Del
SDL_console.h, 25
Cursor_End
SDL_console.h, 24
Cursor_Home
SDL_console.h, 24
Cursor_Left
SDL_console.h, 24
Cursor_Right
SDL_console.h, 24
CursorPos
console_information_td, 11

Default_CmdFunction
SDL_console.h, 24
Default_TabFunction
SDL_console.h, 24
DispX
console_information_td, 12
DispY
console_information_td, 12
documentation.h, 15
DrawCommandLine
SDL_console.h, 24
DT_DestroyDrawText
DT_drawtext.h, 16
DT_DrawText
DT_drawtext.h, 16
DT_drawtext.h, 16
BitFont, 16

DT_DestroyDrawText, 16
DT_DrawText, 16
DT_FontHeight, 16
DT_FontPointer, 16
DT_FontWidth, 16
DT_LoadFont, 16
TRANS_FONT, 16
DT_FontHeight
DT_drawtext.h, 16
DT_FontPointer
DT_drawtext.h, 16
DT_FontWidth
DT_drawtext.h, 16
DT_GetPixel
internal.h, 17
DT_LoadFont
DT_drawtext.h, 16
DT_PutPixel
internal.h, 17

FontHeight
console_information_td, 12
FontNumber
BitFont_td, 8
console_information_td, 10
FontSurface
BitFont_td, 8
FontWidth
console_information_td, 12

HideKey
console_information_td, 10

InputBackground
console_information_td, 12
InsMode
console_information_td, 11
internal.h, 17
DT_GetPixel, 17
DT_PutPixel, 17
PRINT_ERROR, 17

LCommand
console_information_td, 11
LineBuffer
console_information_td, 10

NextFont
BitFont_td, 8

Offset
console_information_td, 11
OutputScreen
console_information_td, 12

PRINT_ERROR
internal.h, 17
Prompt
console_information_td, 11

RaiseOffset
console_information_td, 10
RCommand
console_information_td, 11

SDL_console.h
CON_CLOSED, 21
CON_CLOSING, 21
CON_OPEN, 21
CON_OPENING, 21
SDL_console.h, 18
Assemble_Command, 25
Clear_Command, 25
Clear_History, 25
Command_Down, 25
Command_Up, 25
CON_Alpha, 22
CON_AlphaGL, 22
CON_Background, 22
CON_BLINK_RATE, 19
CON_CHAR_BORDER, 20
CON_CHARS_PER_LINE, 19
CON_DEFAULT_HIDEKEY, 20
CON_DEFAULT_PROMPT, 20
CON_Destroy, 22
CON_DrawConsole, 21
CON_Events, 21
CON_Execute, 23
CON_Free, 22
CON_Hide, 21
CON_Init, 21
CON_INS_CURSOR, 20
CON_isVisible, 21
CON_LINE_SCROLL, 20
CON_NewLineCommand, 24
CON_NewLineConsole, 24
CON_OPENCLOSE_SPEED, 20
CON_Out, 22
CON_OVR_CURSOR, 20
CON_Position, 22
CON_Resize, 22
CON_SCROLL_INDICATOR, 20
CON_SetExecuteFunction, 23
CON_SetHideKey, 23
CON_SetPrompt, 23
CON_SetTabCompletion, 23
CON_Show, 21
CON_TabCompletion, 23
CON_Topmost, 23

CON_Transfer, 22
CON_UpdateConsole, 24
CON_UpdateOffset, 21
ConsoleInformation, 20
Cursor_Add, 25
Cursor_BSpace, 25
Cursor_Del, 25
Cursor_End, 24
Cursor_Home, 24
Cursor_Left, 24
Cursor_Right, 24
Default_CmdFunction, 24
Default_TabFunction, 24
DrawCommandLine, 24

TabFunction
console_information_td, 12
TotalCommands
console_information_td, 10
TotalConsoleLines
console_information_td, 10
TRANS_FONT
DT_drawtext.h, 16

VChars
console_information_td, 10
VCommand
console_information_td, 11
Visible
console_information_td, 10

WasUnicode
console_information_td, 10