# libaescprintf Documentation

***Release latest***

**Sep 14, 2017**

# Contents:

CHAPTER 1

# libaescprintf (ANSI Escape Sequence Color Print Format).

**library** libaescprintf

**version** latest

**author** Brüggemann Eddie

**contact** <mrcyberfighter@gmail.com>

**license** LGPLv3

**website** <http://www.open-source-projects.net/libaescprintf/libaescprintf>

**release** Sep 14, 2017

## Introduction

**`libaescprintf`** is a little library based on the `ANSI escape sequences` and only compatible with terminals *ANSI escape sequences* compliant terminals.

Most of terminal that I have tested works well but not all.

But in addition **`libaescprintf`** provides some styles features like **bold**, **strike-trough**, **italic**, and so on...

---

**Note:** You can insert the *ANSI escapes sequences* into an **`echo`** command or using the binary **`tput`** but with this library you can use it into a source code.

---

**note** *The goal* of this library is to provide a lightweight textual pretty output.

## libaescprintf

- **libaescprintf** can be divided into 2 different output functions types.
  - `printf(...)()`

    For simply formatting the output.

  and
  - `fprintf(...)()`

    For using the `FILE *` **stream** you want.

    And why not serializing to a file...
- **libaescprintf** can be divided into *background and foreground* colors.

  The colors can be divided into Light and Dark colors.
- **libaescprintf** provides styles features.

# CHAPTER 2

## Types of libaescprintf

**libaescprintf** make usage of different types of colors and styles:

```
// Building a color and a style type. (Used into library functions).

enum style_enum { normal = 0 , bold = 1 , dim = 2 , italic = 3, underline =␣
↪4 , blink = 5, inverted = 7, hidden = 8, strikethrough = 9 } ;

//typedef enum color_enum  color_t ;
typedef enum style_enum  style_t ;

// Building a foreground color type. (Used into library functions).

                    // Foreground Normal >= 30   <= 37   (8 colors)
                    // Foreground Light  >= 90   <= 97   (8 colors)
enum fg_color_enum { fg_black     = 30,     fg_darkgray     = 90,
                     fg_red       = 31,     fg_lightred     = 91,
                     fg_green     = 32,     fg_lightgreen   = 92,
                     fg_yellow    = 33,     fg_lightyellow  = 93,
                     fg_blue      = 34,     fg_lightblue    = 94,
                     fg_pink      = 35,     fg_lightpink    = 95,
                     fg_cyan      = 36,     fg_lightcyan    = 96,
                     fg_lightgray = 37,     fg_white        = 97,

                     fg_default   = 39 } ;


// Building a background color type. (Used into library functions).

                    // Background Normal >= 40   <= 47   (8 colors)
                    // Background Light  >= 100  <= 107  (8 colors)
enum bg_color_enum { bg_black     = (30+10),     bg_darkgray    = (90+10),
                     bg_red       = (31+10),     bg_lightred    = (91+10),
                     bg_green     = (32+10),     bg_lightgreen  = (92+10),
                     bg_yellow    = (33+10),     bg_lightyellow = (93+10),
                     bg_blue      = (34+10),     bg_lightblue   = (94+10),
```

```
                    bg_pink      = (35+10),     bg_lightpink   = (95+10),
                    bg_cyan      = (36+10),     bg_lightcyan   = (96+10),
                    bg_lightgray = (37+10),     bg_white       = (97+10),

                    bg_default   = (39+10) } ;

typedef enum fg_color_enum  fg_color_t ;
typedef enum bg_color_enum  bg_color_t ;
```

*Which you can use in various functions from this library.*

# Styles

- normal
- bold
- dim
- italic
- underline
- blink
- inverted
- hidden
- strikethrough

# Colors

## Foreground colors values

- fg_black
- fg_darkgray
- fg_red
- fg_lightred
- fg_green
- fg_lightgreen
- fg_yellow
- fg_lightyellow
- fg_blue
- fg_lightblue
- fg_pink
- fg_lightpink
- fg_cyan

- fg_lightcyan
- fg_lightgray
- fg_white
- fg_default (Your current terminal foreground color).

### Background colors values

- bg_black
- bg_darkgray
- bg_red
- bg_lightred
- bg_green
- bg_lightgreen
- bg_yellow
- bg_lightyellow
- bg_blue
- bg_lightblue
- bg_pink
- bg_lightpink
- bg_cyan
- bg_lightcyan
- bg_lightgray
- bg_white
- bg_default (Your current terminal background color).

# CHAPTER 3

## Defines of libaescprintf

The ANSI escape sequences are defines as well into **libaescprintf**:

```
/** You can use this defines into raw string formatting. **/

#define ESC   "\033"  // Escape sequence (octal format).

// Dark colors defines.
#define DB    ESC "[30m"  // Dark Black.
#define DR    ESC "[31m"  // Dark Red.
#define DG    ESC "[32m"  // Dark Green.
#define DY    ESC "[33m"  // Dark Yellow.
#define DS    ESC "[34m"  // Dark Sky. (Dark blue)
#define DP    ESC "[35m"  // Dark Pink.
#define DC    ESC "[36m"  // Dark Cyan.
#define DW    ESC "[37m"  // Dark white. (Light Gray)

// Light colors defines.
#define LB    ESC "[90m"  // Light Black.
#define LR    ESC "[91m"  // Light Red.
#define LG    ESC "[92m"  // Light Green.
#define LY    ESC "[93m"  // Light Yellow.
#define LS    ESC "[94m"  // Light Sky.  (Light blue)
#define LP    ESC "[95m"  // Light Pink.
#define LC    ESC "[96m"  // Light Cyan.
#define LW    ESC "[97m"  // Light White.

#define DD    ESC "[39m"  // Default color.

// Styles defines
#define RESET           ESC "[0m"   // Reset.
#define BOLD            ESC "[1m"   // Bold.
#define DIM             ESC "[2m"   // DIM
#define ITALIC          ESC "[3m"   // Italic.
#define UNDERSTRIKE     ESC "[4m"   // Under-strike.
#define BLINK           ESC "[5m"   // Blink (Doesn't work on all terminals␣
↪look at xterm it work's on it !).
```

```
#define INVERTED        ESC "[7m"   // Inverted (Invert foreground and␣
↪background).
#define HIDDEN          ESC "[8m"   // Hidden (Print only background color).
#define STRIKETROUGHT   ESC "[9m"   // Strike-through.
```

You can simply make usage of them into `printf()` family functions.

CHAPTER 4

# The styled colored `printf()` functions:

## Styled Print Format:

int **`s_printf`** (style_t *style*, const char **fmt*, ...)

**Parameters**

- **`style`** – the *style* to apply to the text.
- **`fmt`** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## Color Foreground Print Format:

int **`cfg_printf`** (fg_color_t *color*, const char **fmt*, ...)

**Parameters**

- **`fg_color`** – The *foreground color* to apply.
- **`fmt`** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## Color Background Print Format:

int **`cbg_printf`** (bg_color_t *bgcolor*, const char **fmt*, ...)

**Parameters**

- **bg_color** – The *foreground color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## Styled Foreground Color Print Format:

int **scfg_printf** (style_t *style*, fg_color_t *fg_color*, const char **fmt*, ...)

**Parameters**

- **style** – the *style* to apply to the text.
- **fg_color** – The *foreground color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## Styled Background Color Print Format:

int **scbg_printf** (style_t *style*, bg_color_t *bg_color*, const char **fmt*, ...)

**Parameters**

- **style** – the *style* to apply to the text.
- **bg_color** – The *background color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## Color Foreground Background Print Format:

int **cfgbg_printf** (fg_color_t *fg_color*, bg_color_t *bg_color*, const char **fmt*, ...)

**Parameters**

- **fg_color** – The *foreground color* to apply.
- **bg_color** – The *foreground color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## Styled Color Foreground Background Print Format:

int **scfgbg_printf**(style_t *style*, fg_color_t *fg_color*, bg_color_t *bg_color*, const char **fmt*, ...)

**Parameters**

- **style** – the *style* to apply to the text.
- **fg_color** – The *foreground color* to apply.
- **bg_color** – The *foreground color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

CHAPTER 5

# The styled colored `fprintf()` functions

## File Styled Print Format:

int **`s_fprintf`** (FILE **fp*, style_t *style*, const char **fmt*, ...)

**Parameters**

- **`fp`** – A `FILE` pointer.
- **`style`** – the *style* to apply to the text.
- **`fmt`** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## File Color Foreground Print Format:

int **`cfg_fprintf`** (FILE **fp*, fg_color_t *color*, const char **fmt*, ...)

**Parameters**

- **`fp`** – A `FILE` pointer.
- **`fg_color`** – The *foreground color* to apply.
- **`fmt`** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## File Color Background Print Format:

int **cbg_fprintf** (FILE **fp*, bg_color_t *bgcolor*, const char **fmt*, ...)

**Parameters**

- **fp** – A FILE pointer.
- **bg_color** – The *foreground color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** int.

## File Styled Foreground Color Print Format:

int **scfg_fprintf** (FILE **fp*, style_t *style*, fg_color_t *fg_color*, const char **fmt*, ...)

**Parameters**

- **fp** – A FILE pointer.
- **style** – the *style* to apply to the text.
- **fg_color** – The *foreground color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** int.

## Styled Background Color Print Format:

int **scbg_printf** (FILE **fp*, style_t *style*, bg_color_t *bg_color*, const char **fmt*, ...)

**Parameters**

- **fp** – A FILE pointer.
- **style** – the *style* to apply to the text.
- **bg_color** – The *background color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** int.

## File Color Foreground Background Print Format:

int **cfgbg_fprintf** (FILE **fp*, fg_color_t *fg_color*, bg_color_t *bg_color*, const char **fmt*, ...)

**Parameters**

- **fp** – A FILE pointer.

- **fg_color** – The *foreground color* to apply.
- **bg_color** – The *foreground color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

## File Styled Color Foreground Background Print Format:

int **scfgbg_fprintf** (FILE **fp*, style_t *style*, fg_color_t *fg_color*, bg_color_t *bg_color*, const char **fmt*, ...)

**Parameters**

- **fp** – A `FILE` pointer.
- **style** – the *style* to apply to the text.
- **fg_color** – The *foreground color* to apply.
- **bg_color** – The *foreground color* to apply.
- **fmt** – The format string for the output.

**Returns** The number of characters from the output string or -1 in case of error.

**Return type** `int`.

CHAPTER 6

# About **libaescprintf**

At first I often try myself to find the generic function into termcap to get the colors and styles codes for building a library compliant with any terminal and I fail to find it...
So I often try to get some help from a terminal dinosaur which master termcap or termios to make a more generic library.
But nobody **HOME**...

**Note:** I can do more with terminal Sequences as displaying a True Color image (RGB) into a terminal (xterm, terminals build with libvte-2.91 on GNU/Linux)
But I must reduce the cursor to the minimum size
for doing this but I can't restore a normal cursor size...

**enjoy** So enjoy using **libaescprintf** for ANSI art as I enjoy to write this little library.

## Compatibility

**libaescprintf** was tested on as many terminals I could find into the repositories, sometimes some terminals either don't blink or don't accept **ANSI** escape sequences at all.
But most of the time it works well.
It works in most of the terminals: konsole as gnome-terminal and many others.
But the old bug concern the background and the linefeed appears on some terminals:
if you set the background color other than default and set an linefeed the entire line is background highlighted with the chosen color per example the Apple-Terminal do this one

(but the blinking effect works well with it).

## Note from the author

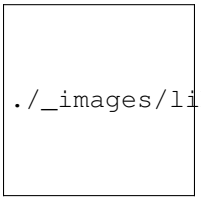
I don't try to reinvent the wheel but simply provide this minimalist library in the following goals:

1. At first to provide a lightweight library for only textual effect.
2. To catch your attention so that someone can help me to make the library universal, using termcap (or termios).

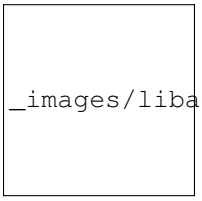
CHAPTER 7

# libaescprintf screen-shots:

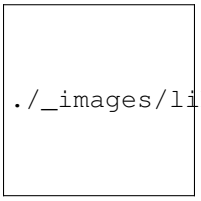
## Matrix output

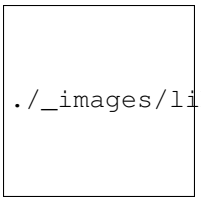



## Text output




## Flags output




./_images/libaescprintf_vertical_flag_example_screenshot.png

CHAPTER 8

# Indices and tables

- genindex
- modindex
- search

# Index

## C



## S