Bash tips: Colors and formatting (ANSI/VT100 Control sequences)

The **ANSI/VT100** terminals and terminal emulators are not just able to display black and white text ; they can display **colors** and formatted texts thanks to **escape sequences**. Those sequences are composed of the **Escape character** (often represented by "^[" or "<Esc>") followed by some other characters: "<Esc>[*FormatCode*m".

In Bash, the <Esc> character can be obtained with the following syntaxes:

- \e
- \033
- \x1B

Examples:

Code (Bash)	Preview
`echo -e "\e[31mHello World\e[0m"`	Hello World
`echo -e "\033[31mHello\e[0m World"`	Hello World

NOTE[1]: The -e option of the echo command enable the parsing of the escape sequences.

NOTE[2]: The "\e[0m" sequence removes all attributes (formatting and colors). It can be a good idea to add it at the end of each colored text. ;)

NOTE[3]: The examples in this page are in **Bash** but the **ANSI/VT100** escape sequences can be used in every programming languages.

Formatting

Here are the most commonly supported control sequences for formatting text. Their support depends on the used terminal (see the compatibility list).

Set

Code	Description	Example	Preview
1	Bold/Bright	`echo -e "Normal \e[1mBold"`	Normal **Bold**
2	Dim	`echo -e "Normal \e[2mDim"`	Normal Dim
4	Underlined	`echo -e "Normal \e[4mUnderlined"`	Normal Underlined
5	Blink [1)]	`echo -e "Normal \e[5mBlink"`	Normal
7	Reverse (invert the foreground and background colors)	`echo -e "Normal \e[7minverted"`	Normal inverted
8	Hidden (useful for passwords)	`echo -e "Normal \e[8mHidden"`	Normal

Reset

Code	Description	Example	Preview
0	Reset all attributes	`echo -e "\e[0mNormal Text"`	Normal Text
21	Reset bold/bright	`echo -e "Normal \e[1mBold \e[21mNormal"`	Normal **Bold** Normal
22	Reset dim	`echo -e "Normal \e[2mDim \e[22mNormal"`	Normal Dim Normal
24	Reset underlined	`echo -e "Normal \e[4mUnderlined \e[24mNormal"`	Normal Underlined Normal
25	Reset blink	`echo -e "Normal \e[5mBlink \e[25mNormal"`	Normal Normal
27	Reset reverse	`echo -e "Normal \e[7minverted \e[27mNormal"`	Normal inverted Normal
28	Reset hidden	`echo -e "Normal \e[8mHidden \e[28mNormal"`	Normal Normal

8/16 Colors

The following colors works with most terminals and terminals emulators [2], see the compatibility list for more informations.

NOTE: The colors can vary depending of the terminal configuration.

Foreground (text)

Code	Color	Example	Preview
39	Default foreground color	`echo -e "Default \e[39mDefault"`	Default Default
30	Black	`echo -e "Default \e[30mBlack"`	Default Black
31	Red	`echo -e "Default \e[31mRed"`	Default Red
32	Green	`echo -e "Default \e[32mGreen"`	Default Green
33	Yellow	`echo -e "Default \e[33mYellow"`	Default Yellow
34	Blue	`echo -e "Default \e[34mBlue"`	Default Blue
35	Magenta	`echo -e "Default \e[35mMagenta"`	Default Magenta
36	Cyan	`echo -e "Default \e[36mCyan"`	Default Cyan
37	Light gray	`echo -e "Default \e[37mLight gray"`	Default Light gray
90	Dark gray	`echo -e "Default \e[90mDark gray"`	Default Dark gray
91	Light red	`echo -e "Default \e[91mLight red"`	Default Light red
92	Light green	`echo -e "Default \e[92mLight green"`	Default Light green
93	Light yellow	`echo -e "Default \e[93mLight yellow"`	Default Light yellow
94	Light blue	`echo -e "Default \e[94mLight blue"`	Default Light blue
95	Light magenta	`echo -e "Default \e[95mLight magenta"`	Default Light magenta
96	Light cyan	`echo -e "Default \e[96mLight cyan"`	Default Light cyan
97	White	`echo -e "Default \e[97mWhite"`	Default White

Background

Code	Color	Example	Preview
49	Default background color	`echo -e "Default \e[49mDefault"`	Default Default
40	Black	`echo -e "Default \e[40mBlack"`	Default Black
41	Red	`echo -e "Default \e[41mRed"`	Default Red
42	Green	`echo -e "Default \e[42mGreen"`	Default Green
43	Yellow	`echo -e "Default \e[43mYellow"`	Default Yellow
44	Blue	`echo -e "Default \e[44mBlue"`	Default Blue
45	Magenta	`echo -e "Default \e[45mMagenta"`	Default Magenta
46	Cyan	`echo -e "Default \e[46mCyan"`	Default Cyan
47	Light gray	`echo -e "Default \e[47mLight gray"`	Default Light gray
100	Dark gray	`echo -e "Default \e[100mDark gray"`	Default Dark gray
101	Light red	`echo -e "Default \e[101mLight red"`	Default Light red
102	Light green	`echo -e "Default \e[102mLight green"`	Default Light green
103	Light yellow	`echo -e "Default \e[103mLight yellow"`	Default Light yellow
104	Light blue	`echo -e "Default \e[104mLight blue"`	Default Light blue
105	Light magenta	`echo -e "Default \e[105mLight magenta"`	Default Light magenta
106	Light cyan	`echo -e "Default \e[106mLight cyan"`	Default Light cyan
107	White	`echo -e "Default \e[107mWhite"`	Default

88/256 Colors

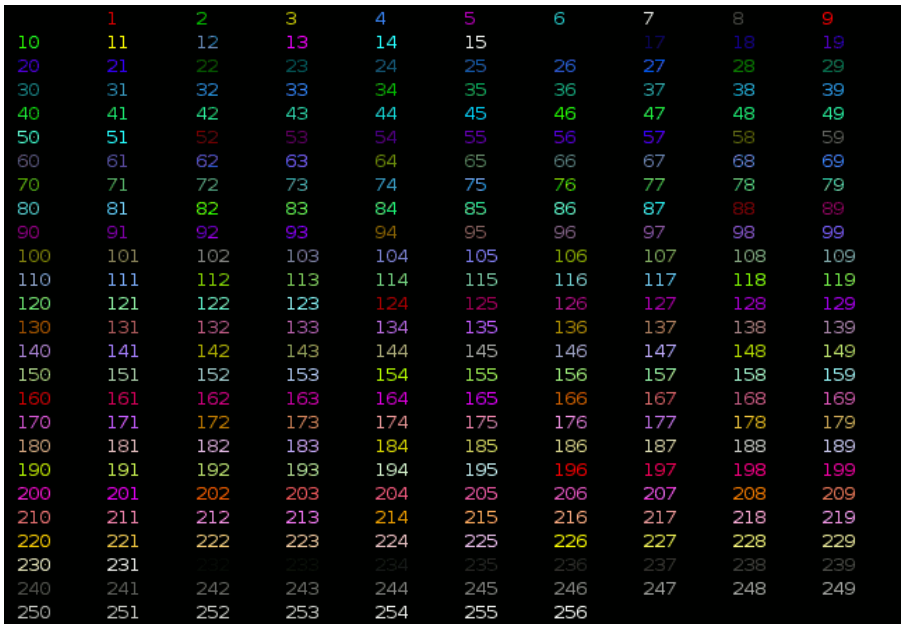
Some terminals (see the compatibility list) can support 88 or 256 colors. Here are the control sequences that permit you to use them.

NOTE[1]: The colors number `256` is only supported by **vte** (GNOME Terminal, XFCE4 Terminal, Nautilus Terminal, Terminator,…).

NOTE[2]: The 88-colors terminals (like **rxvt**) does not have the same color map that the 256-colors terminals. For showing the 88-colors terminals color map, run the "256-colors.sh" script in a 88-colors terminal.

Foreground (text)

For using one of the 256 colors on the foreground (text color), the control sequence is "`<Esc>[38;5;`*ColorNumber*`m`" where `ColorNumber` is one of the following colors:

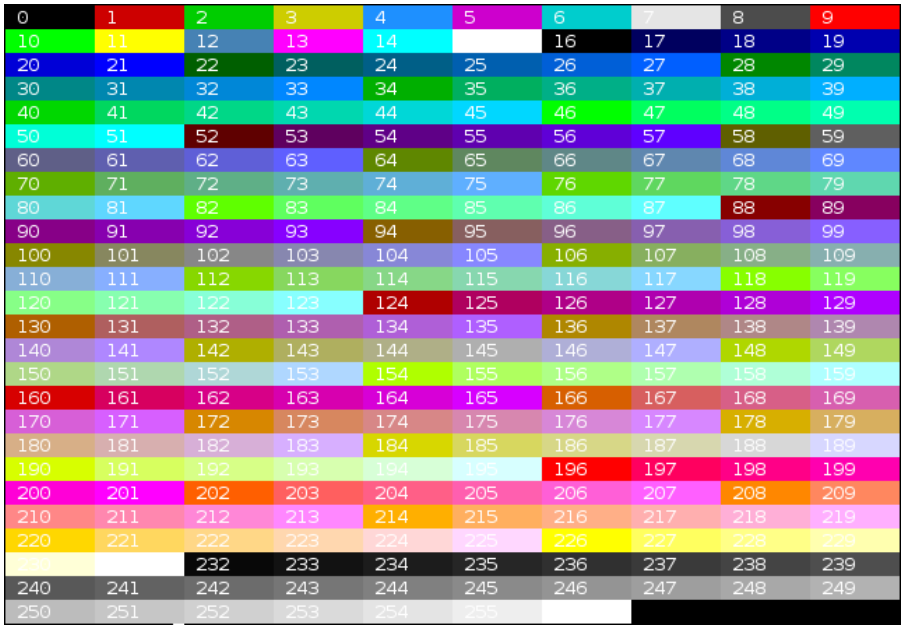


Examples:

Code (Bash)	Preview
`echo -e "\e[38;5;82mHello \e[38;5;198mWorld"`	Hello World
`for i in {16..21} {21..16} ; do echo -en "\e[38;5;${i}m#\e[0m" ; done ; echo`	##########

Background

For using one of the 256 colors on the background, the control sequence is "`<Esc>[48;5;`*ColorNumber*`m`" where `ColorNumber` is one of the following colors:



Examples:

Code (Bash)	Preview
`echo -e "\e[40;38;5;82m Hello \e[30;48;5;82m World \e[0m"`	Hello World

Code (Bash)	Preview
`for i in {16..21} {21..16} ; do echo -en "\e[48;5;${i}m \e[0m" ; done ; echo`	

Attributes combination

Terminals allow attribute combinations. The attributes must be separated by a semicolon (";").

Examples:

Description	Code (Bash)	Preview
Bold + Underlined	`echo -e "\e[1;4mBold and Underlined"`	Bold and Underlined
Bold + Red forground + Green background	`echo -e "\e[1;31;42m Yes it is awful \e[0m"`	Yes it is awful

Terminals compatibility

Terminal	Formatting						Colors				Comment
	Bold	Dim	Underlined	Blink	invert	Hidden	8	16	88	256	
aTerm	ok	-	ok	-	ok	-	ok	~	-	-	Lighter background instead of blink.
Eterm	~	-	ok	-	ok	-	ok	~	-	ok	Lighter color instead of Bold. Lighter background instead of blink. Can overline a text with the "^[[6m" sequence.
GNOME Terminal	ok	ok	ok	ok	ok	ok	ok	ok	-	ok	Strikeout with the "^[[9m" sequence.
Guake	ok	ok	ok	ok	ok	ok	ok	ok	-	ok	Strikeout with the "^[[9m" sequence.
Konsole	ok	-	ok	ok	ok	-	ok	ok	-	ok	
Nautilus Terminal	ok	ok	ok	ok	ok	ok	ok	ok	-	ok	Strikeout with the "^[[9m" sequence.
rxvt	ok	-	ok	~	ok	-	ok	ok	ok	-	If the background is not set to the default color, Blink make it lighter instead of blinking. Support of italic text with the "^[[3m" sequence.
Terminator	ok	ok	ok	-	ok	ok	ok	ok	-	ok	Strikeout with the "^[[9m" sequence.
Tilda	ok	-	ok	ok	ok	-	ok	ok	-	-	Underline instead of Dim. Convert 256-colors in 16-colors.
XFCE4 Terminal	ok	ok	ok	ok	ok	ok	ok	ok	-	ok	Strikeout with the "^[[9m" sequence.
XTerm	ok	-	ok	ok	ok	ok	ok	ok	-	ok	
xvt	ok	-	ok	-	ok	-	-	-	-	-	
Linux TTY	ok	-	-	-	ok	-	ok	~	-	-	Specials colors instead of Dim and Underlined. Lighter background instead of Blink, Bug with 88/256 colors.
VTE Terminal [3)]	ok	ok	ok	ok	ok	ok	ok	ok	-	ok	Strikeout with the "^[[9m" sequence.

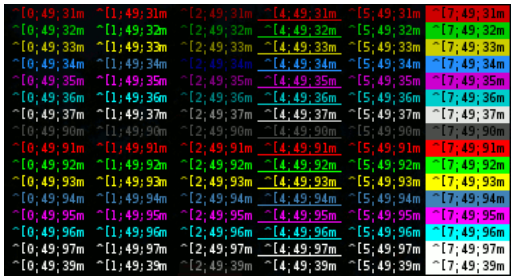
Notations used in the table:

- "ok": Supported by the terminal.
- "~": Supported in a special way by the terminal.
- "-": Not supported at all by the terminal.

Demonstration programs

Colors and formatting (16 colors)

The following shell script displays a lot of possible combination of the attributes (but not all, because it uses only one formatting attribute at a time).

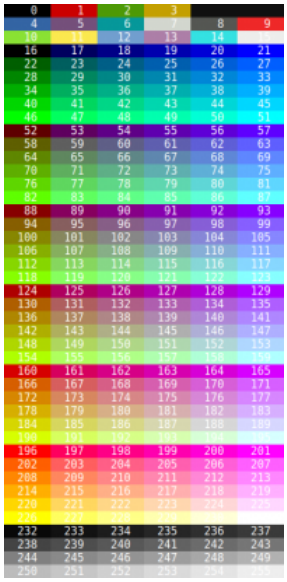


colors_and_formatting.sh

```
#!/bin/bash

# This program is free software. It comes without any warranty, to
# the extent permitted by applicable law. You can redistribute it
# and/or modify it under the terms of the Do What The Fuck You Want
# To Public License, Version 2, as published by Sam Hocevar. See
# http://sam.zoy.org/wtfpl/COPYING for more details.

#Background
for clbg in {40..47} {100..107} 49 ; do
        #Foreground
        for clfg in {30..37} {90..97} 39 ; do
                #Formatting
                for attr in 0 1 2 4 5 7 ; do
                        #Print the result
                        echo -en "\e[${attr};${clbg};${clfg}m ^[${attr};${
                done
                echo #Newline
        done
done

exit 0
```

256 colors

The following script display the 256 colors available on some terminals and terminals emulators like **XTerm** and **GNOME Terminal**.



256-colors.sh

```
#!/bin/bash

# This program is free software. It comes without any warranty, to
# the extent permitted by applicable law. You can redistribute it
# and/or modify it under the terms of the Do What The Fuck You Want
# To Public License, Version 2, as published by Sam Hocevar. See
# http://sam.zoy.org/wtfpl/COPYING for more details.

for fgbg in 38 48 ; do # Foreground / Background
    for color in {0..255} ; do # Colors
        # Display the color
        printf "\e[${fgbg};5;%sm  %3s  \e[0m" $color $color
        # Display 6 colors per lines
        if [ $((($color + 1) % 6)) == 4 ] ; then
            echo # New line
        fi
    done
    echo # New line
done

exit 0
```

Links

- Linux console codes manual ("man console_codes")
- XTerm Control Sequences
- Compilation of all escape sequences
- ANSI escape code (Wikipedia)

1) Does not work with most of the terminal emulators, works in the tty and XTerm.
2) Some terminals supports only the first 8 colors (30..37 and 40..47), and some others does not support any color at all.
3) GTK Widget used in GNOME Terminal, Nautilus Terminal, XFCE4 Terminal...

Discussion

William C Grisaitis, 2011/11/13 01:00

Thanks! This was invaluable in customizing my PS1's:

if [[${EUID} == 0]] ; then
PS1='\e[1;31;48;5;234m\u \e[38;5;240mon \e[1;38;5;28;48;5;234m\h \e[38;5;54m\d \@\e[0m\n\e[0;31;48;5;234m[\w] \e[1m\$\e[0m '
else
PS1='\e[1;38;5;56;48;5;234m\u \e[38;5;240mon \e[1;38;5;28;48;5;234m\h \e[38;5;54m\d \@\e[0m\n\e[0;38;5;56;48;5;234m[\w] \e[1m\$\e[0m '
fi

@caravaggisto

Reply

Barry Scott, 2012/06/14 19:41

Great work on terminal compatibility.I have been trying to get blinking text on a Linux tty(at the console). Do you have any idea if it's possible?

Reply

Anatoly, 2017/09/21 09:54

Not all terminal support blinking, and dim too. Before i think there are only 16 colors support. But now I see 256 are. It's very good. But for 16 only you may design
pseudo graphic interface, draw good windows and all graphical controls in text mode. Only you need(for russians) use DOS866 encodding set. It containe full set of pseudo graphic symbols, others no. There is set of libraries of pseudo graphic controls. And you may easily make TUI(text user interface) API like GUI. But library is Turbo Vision for DOS 16 only. But this libs in source code available. If you want you may rewrite them for Linux platform. And if use 256 colors you get more better design nearest to GUI. Many years I try to find redy solution but failed. So if you want to do that you need to all work by yourself. But result will best. This API take tens times less resources and quicker then gui. They don't require GUI regime at all... It will be best
But API of all. There is only 1 restriction. You don't must draw pictures, graphics, videos and so on where you need pixel draw indeed. But there are little API like that. Most API don't need pixel draw at all.

Reply

Fabien LOISON, 2012/06/14 19:54

I Think it is possible, but I haven't found how to do that

Reply

Barry Scott, 2012/06/14 20:11

I have looked at infocmp for linux the terminal(TERM=linux) I use and I see blink referenced in it but I'm having a hard time understanding the file format. The cursor blinks why disable blinking text.

Reply

Warron, 2013/04/04 17:09

Great page on bash coloring and attributes.

I was actually looking to find out if there is a way to combine attributes {BOLD, Blink, etc} around the same subset of text in doing a bash echo command with the -e option.

Can you help with this matter?

\\War

Reply

Fabien LOISON, 2013/04/04 19:54

Hello,

You can combine attributes with a semicolon:

```
echo -e "\e[1;5m Bold+Blink \e[0m"
echo -e "\e[1;4;31m Bold+Underline+Red \e[0m"
```

Note that the blink attribute is supported only by few terminals (XTerm, tty).

Regards,

Reply

Warron, 2013/04/11 19:49

Thank you Fabian.

That worked splendidly! You are the man!

Reply

Konrad, 2015/07/25 19:51

Thank you!

konrad@vps1 ~/web/abc▌▌▌ mkdir xyz

```
PS1='\[\e[0m\]\[\e[48;5;236m\]\[\e[38;5;105m\]\u\[\e[38;5;105m\]@\[\e[38;5;105m\]\h\[\e[38;5;105m\]     \[\e[38;5;221m\]\w\
[\e[38;5;221m\]\[\e[38;5;105m\]\[\e[0m\]\[\e[38;5;236m\]\342\226\214\342\226\214\342\226\214\[\e[0m\]'
```

root:

```
PS1='\[\e[0m\]\[\e[48;5;236m\]\[\e[38;5;197m\]\u\[\e[38;5;197m\]@\[\e[38;5;197m\]\h\[\e[38;5;105m\]     \[\e[38;5;221m\]\w\
[\e[38;5;221m\]\[\e[38;5;105m\]\[\e[0m\]\[\e[38;5;236m\]\342\226\214\342\226\214\342\226\214\[\e[0m\]'
```

Reply

Per Bothner, 2015/11/28 19:03

Note that the 256-colors.sh script uses a tab character, which has different behavior on different emulators.
On xterm and Konsole, TAB moves the cursor, without touching the skipped-over positions (so the background color is unchanged), while Gnome Terminal appears to effectively write spaces (so the background color is changed). Your images show the latter, but note that is incompatible with xterm.

Reply

egmontkob, 2017/10/10 10:01

Note that Gnome Terminal (actually VTE version 0.44.2) has also changed its behavior to be like xterm, making the patch from the next comment necessary.

Reply

Per Bothner, 2015/11/28 19:14

A fix for 256-colors.sh that uses printf instead of tabs:

```
#Display the color
echo -en "\e[${fgbg};5;${color}m"
printf "%4d " ${color}
echo -en "\e[0m"
```

Also, the upper cbound should be 255, not 256:

```
for color in {0..255} ; do #Colors
```

Reply

Nga Nguyen Duy, 2015/12/06 20:55

I don't know what is the difference between the <ESC> characters:

\e

\033

\x1B

Can somebody explain for me?
Thank in advance.

Reply

Fabien LOISON, 2015/12/07 08:29, 2015/12/07 08:31

Hello,

This is only three ways to represent the same character. There will be no differences between using one representation or an other.

* \e is a convenient way provided by Bash to insert the Escape character.

* 33 is the position of the Escape character in the ASCII table expressed in octal (base 8, in decimal this is equal to 27)
* 1B is the position of the Escape character in the ASCII table expressed in hexadecimal (base 16)

→ So \0nn and \xNN are just a way to insert a character by providing its position in the ASCII table, in octal or hexadecimal format.

You can find an the ASCII table here → https://duckduckgo.com/?q=ascii+table&t=canonical

Reply

g alexander, 2016/02/12 21:13

you are a bash scripting color, ascii man among boys

Reply

g alexander, 2016/02/12 21:20

good work.

with that gradient,i was trying to work how to put text inside of it to get a gradient of of text but the text just repeats with the loop. how can i put this into a function something like gradient "some text" blue white or gradient "more text" blue white yellow, function gradient(){}?

Reply

Mohsen Pahlevanzadeh, 2016/03/15 01:33

blink code doesn't work.

for example:

echo -e "Normal \033[5mHello"

Normal Hello

It's normal print Normal Hello, Not blink.

Can you write truly blink text?

Reply

Fabien LOISON, 2016/03/15 08:13

Hello,

blink do not work on vte based terminals (most linux terminal, like gnome-terminal, tilda, guake, terminator, xfce4-terminal,...)

You can try with xterm, it should work on it.

See the compatibility table for more info: http://misc.flogisoft.com/bash/tip_colors_and_formatting?&#terminals_compatibility

Reply

egmontkob, 2017/12/23 22:39

Blinking is going to work in gnome-terminal and friends beginning with VTE 0.52 (to be released in March 2018).

Reply

Fabien LOISON, 2017/12/25 14:05

Thank you for the information :)

Reply

Gerry, 2016/04/12 18:26

Here's a little more on resetting:

\e[0m resets all colors and attributes.
\e[20m resets only attributes (underline, etc.), leaving colors unchanged.
\e[39m resets only foreground color, leaving attributes unchanged.
\e[49m resets only background color, leaving attributes unchanged.

Reply

Ron, 2016/05/13 13:17

(Taken from http://makandracards.com/makandra/1090-customize-your-bash-prompt :)
\u: current username
\h: hostname up to the first ., \H: full hostname
\w: current working directory, \W: same, but only the basename
$(__git_ps1 "%s"): your current git branch if you're in a git directory, otherwise nothing

\$: if the effective UID is 0: #, otherwise $
\d: the date in "Weekday Month Date" format (e.g., "Tue May 26")
\t: the current time in 24-hour HH:MM:SS format, \T: same, but 12-hour format, \@: same, but in 12-hour am/pm format
\n: newline
\r: carriage return
\\: backslash

Reply

Fabien LOISON, 2016/05/13 13:21

@Ron: \u, \h &co are available only in prompts:

http://misc.flogisoft.com/bash/tip_customize_the_shell_prompt

Reply

Toby, 2016/06/13 13:11

Please, please, please DON'T encourage people to put the raw terminal codes into their message strings! That way lies madness, because not all the world is a VT100/VT220/etc. Instead, use the 'tput' program to generate the correct code (if one exists) for the user's terminal. That is much more portable, and doesn't clutter the poor user's screen with lots of escape character clutter when they run your program from a non-terminal environment.

Reply

Fabien LOISON, 2016/06/13 13:21

Of course it is better to use libs or programs that abstract all the things and make it works with almost any terminals. But it still usefull to know how it works behind :)

Reply

fujisan, 2016/06/14 09:05

On a mate terminal with a white background, the bold (echo -e "Default \e[1mDefault") is actually white so impossible to see the characters.

Reply

Fabien LOISON, 2016/06/14 12:05

In GNOME Terminal there is an option to set the color of the bold text (right click → Profiles → Profile Settings → Colors → Bold colors), there should be the same on mate-terminal.

PS: I translated the menu label from my french gnome-terminal. In yours, it can be slightly deferent.

Reply

Aakash Martand, 2016/09/23 08:30

Nice work.

would you please explain the control sequence of 8/16 Colors and 88/256 Colors

Reply

Fabien LOISON, 2016/09/26 10:47

what do you want I explain ?

Reply

Aakash Martand, 2016/09/26 13:14

Like in your example, \e[30;48;5;82m World you've used 4 parameters. Is there any specific sequence for that?

As I understand,

30 is for black text.

48 is for what?

5 is for blink which is not happening, not even in Xterm.

82 is background color.

please help.

Reply

Fabien LOISON, 2016/09/26 13:24, 2016/09/26 13:25

Ah ok,

In 8/16 color mode:

"3x" is for foreground color
"4x" is for background color

In 88/256 color mode:

"38;5" means "the next number is a foreground color in 88/256 color mode"
"48;5" means "the next number is a background color in 88/256 color mode"

so "38;5;XXX" and "48;5;XXX" allow you to select colors in 88/256 color mode.

In your example ("\e[30;48;5;82m"),
"30" is for back foreground (text in black)
"48;5;82" is for green background (in 88/256 color mode)

Reply

Aakash Martand, 2016/09/26 14:01

Now I clearly understand.

Thanx buddy.

keep rocking.

Reply

Joe, 2016/10/11 17:13

This is an awesome document! It is well written! Thanks for making it clear.

Cheers,

+ Joe

Reply

Mark, 2016/10/20 09:02

Perfect tips! One more question - how make colored backgroud to whole line?

Reply

Fabien LOISON, 2016/10/20 11:48

I do not know other solution than filling the line with spaces...

Reply

egmontkob, 2017/10/10 10:40

In terminals that support "bce" (background color erase), the "el" (clear to end of line) sequence fills up the line with the current background color. This bce is supported by most graphical terminal emulators, while it's not supported by screen and tmux. An advantage of using this feature is that you don't end up with tons of space characters on copy-paste.

Example usage might so something like this:

```
if tput el; then
tput bce
else
# fill up manually with spaces
fi
```

Reply

Eddie, 2016/11/15 12:48

Hi all,

In my shell script formating text (bold/colors) all works and the results look correct

if the output is sent to standard output. (Just calling the script ./myscript.sh

But, if i redirect the output into a file i only see original text including

statements such as ESC[90G ESC[1;32 and so on.

Any ideas?

1. Content of myscript.sh:

echo -e "OKAY TO BE PRINTED IN COLUMN 50 OF THIS LINE \e[20G OKAY"

2. ./myscript.sh &> output.txt 2>&1

3. Use Notepad++ to open output.txt: I see

OKAY TO BE PRINTED ON COLUMN 50 OF THIS LINE ESC[20G OKAY

If i use cat to show the content i see the correct results.

However, i want to see the same result in the text file as it is shown on default output.

Reply

Eddy, 2016/11/15 12:57

Hi all,
do you know how can i make this formating to be kept in the file if i redirect the output of my shell script?
1. Content of my shell script "myscript.sh"
echo -e "PRINT RED HELLO AT COLUMN POSITION 80 \e[80G \e[91m HELLO"
2. ./myscript.sh &> output.txt
3. Content of output.txt:
PRINT RED HELLO AT COLUMN POSITION 80 ESC[80G ESC[91m HELLO

Many thanks for your support in advance.

Regards, Eddy

Reply

Fabien LOISON, 2016/11/17 19:23

Hello,

You cannot see the formating in your text editor, because it is your terminal emulator (XTerm, GNOME Terminal, Konsole,...) that generates colors when there is some special byte sequence in the output. Your text editor will just display the content of the file, it will not interpret it.

Regards,

Reply

Emeric, 2016/11/24 17:59

Hey guys, here is another script to display 256 colors in a terminal.

To be honnest it's basically the same but the output is a bit more... readable.

```
for fgbg in 38 48 ; do

i=0

for color in {0..15} ; do

if [ $i -lt 10 ] ; then

echo "\x1B[${fgbg};5;${color}m"' '${color}' '"\x1B[0m" | tr -d '\n'

else

echo "\x1B[${fgbg};5;${color}m"' '${color}' '"\x1B[0m" | tr -d '\n'

fi

i=$(($i+1))

if [ $((i % 8)) == 0 ] ; then

echo

fi

done

i=0

for color in {16..255} ; do

if [ $i -lt 84 ] ; then

echo "\x1B[${fgbg};5;${color}m"' '${color}' '"\x1B[0m" | tr -d '\n'

else
```

```
echo "\x1B[${fgbg};5;${color}m"' '${color}' '"\x1B[0m" | tr -d '\n'

fi

i=$(($i+1))

if [ $((i % 6)) == 0 ] ; then

echo

fi

done

echo

echo

done

exit 0
```

(sorry for the horrible indentation, no way to fix this unfortunately)

Reply

ET, 2017/05/22 09:09

Just wanted to say thanks..
It is really informative and helpful, and a it's shame there is no formal document about this..
Also, just adding 22 as normal attribute code.

Reply

NeoBeum, 2017/05/23 11:35

Hi, thanks for that intro to unix terminal. This is for other people trying to memorise the colour sequences...
Last year I was bored in class learning the Windows terminal & Visual studio; I worked in hardware before I started studying, I made a chart for my classmates that translated what the code effectively was telling the 'pixels' what to do. So I made a wrapper that just turned 'bitswitches' on and off for each of the primary light colours and told them - 'Rather than trying to remember or have to look up what colour combinations output what, remember it as R.G.B. and a Power Intensity... if you want Bright Red, that's Full power, with Red Only... if you want a purple, it's Red and Blue for Magenta, and half power...If you know what the other two are, yellow and cyan, you won't need to remember 255 colours any more.'
The K in the chart represents 'Key' and the others on the HSB are dependent on how the manufacturer programmed in the logic circuit for high+ or low- voltage and the main circuit flag#.
http://i.imgur.com/YRNlKoZ.png
Soon after, the rest of the class were printing out rainbows for Hello World.

Text version of the chart: View it in monospace font, no tabs, just spaces.

```
## HARDWARE REFERENCE
DECIMAL 128 64 32 16 8 4 2 1
COLOR + BBF F- # K B G R
BINARY 0 0 1 1 0 1 0 1
HEX 3 5
FOREGROUND F F
BACKGROUND BB

## HARDWARE REFERENCE
DECIMAL 128 64 32 16 8 4 2 1
COLOR + BBF F- # K B G R
BINARY 1 0 0 1 0 0 0 1
HEX 9 1
FOREGROUND F F
BACKGROUND BB
```

Reply

sumit , 2017/06/10 10:18

Hello,

My concern is , I have make 1 shell script which output come in colourful. So my requirement is I have save this output in .csv and i want when i fetch this .csv in local desktop output also come in colourful. Please help

Reply

Garry, 2017/09/15 22:01

So the 256 colors - is there anywhere where I can look up what RGB values these match to?

For example, let's say (background) color 121, it's a light green. It is pretty close to "Pale Green" i.e. Red=152 Green=251 Blue=152 (or if you prefer hex, 98FB98). Is there somewhere I can look up the RGB values for 121, etc.?

So I'm trying to setup something that will use my prompt to change the colors, like this (works in bash, but not ksh):

```
PS1="\033[48;5;121m\033[34m\033[7m${LOGNAME}@${HOSTNAME}#\033[27m "
```

In my .profile, it will look up some information and set PS1 accordingly. For example, production servers would get one color of background, development servers another color. Linux servers get one color of foreground, Solaris another, etc. So, if I'm logged into a development Linux box, and I login from there into a production Solaris box, my colors will change - giving me a visual cue that I'm on a production server now, etc.

I have some other things that I want to use matching colors for, and I can define the colors using RGB. If I use color 121 for development, I'd need to know what RGB value that equates to so that I can use that same color to represent development on other things where I would define the color with RGB.

So is there are chart that shows these 256 colors and their RGB equivalents?

Reply

Fabien LOISON, 2017/09/17 18:24

Hello, I do not know where you can find the list of the default palette color. But there is a way to use RGB values in some terminals, I will update this article when I will have some time.

For the background color, the sequance is "\033[48;2;R;G;Bm" (e.g. "\033[48;2;255;64;0m Hello \033[0m")

For the foreground color, the sequance is "\033[38;2;R;G;Bm" (e.g. "\033[38;2;255;64;0m Hello \033[0m")

Reply

Jan Dolinár, 2017/09/27 12:58

Minor correction:

In xterm \e[21m does NOT perform reset of bold. According to docs (http://invisible-island.net/xterm/ctlseqs/ctlseqs.html#h2-Functions-using-CSI-_-ordered-by-the-final-character_s_) \e[21m is "doubly underlined". To correctly reset either bold or dim to normal on xterm, one must actually use \e[22m. Which makes it pretty un-intuitive and pretty much the only way to find out is the hard way :-(Other terminals (at least VTE based ones), work just as described on this page.

Reply

hello, 2017/09/27 14:31

i just called to say i love you

no but seriously this was an insanely useful guide

Reply

egmontkob, 2017/10/10 10:10

Several terminal emulators now support 16 million colors, a.k.a. truecolors. See https://gist.github.com/XVilka/8346728 for details.

Reply

Matthias Delamare, 2017/11/05 12:16

For a better presentation, change this line
if [$(($color + 1) % 10)) == 0] ; then

... to the following one :

if [$(($color + 1) % 6)) == 4] ; then

You'll have a better comprehension, and choosing the color will be easier for you.

Reply

Fabien LOISON, 2017/11/06 08:18, 2017/11/08 08:30

You are right, it is more readable like this, I will update later ;)

Edit: updated! :)

Reply

Steve, 2018/04/10 14:28

Awesome page, saved me a sh** ton of working finding this info, thanks again! :)

Reply

Jordan, 2018/04/12 12:58

256 colors with a blacklist of the hardest colors to see if your background is white or black

https://gist.github.com/hypergig/ea6a60469ab4075b2310b56fa27bae55

Reply

zpo, 2018/07/06 07:16

Thanks a lot for sharing, you have my best wishes. But now I am working on logs stuff. I tried to output shell scripts'printing to log files but it didn't work when I check the log. Colors and bold styles seem only work in Terms, does it?

Reply

Fabien LOISON, 2018/07/07 14:57

Hello,

yes, this works only when you print text in a terminal

Reply

me, 2018/07/26 19:07

Here's a cool one:
PS1="\[\e[91m\]\u\[\e[38;5;208m\]@\[\e[92m\]\h:\[\e[96m\]\$PWD\[\e[35m\]//$(date +"%D-%H:%M" | sed 's/\//-/g')\n\[\e[38;5;21m\][$]~> \[\e[0m\] "

Reply

Boo, 2018/07/31 07:08

So… when are you going to finish this page:

https://misc.flogisoft.com/_detail/bash/ico/tip_cursor_movements.png?id=bash%3Ahome ?

I need similar bash tips on cursor movement.

Thank you. :)

Reply

Fabien LOISON, 2018/07/31 07:16, 2018/07/31 07:17

Hello, I haven't touched to this wiki for a looong time, I will probably never do it...

anyway, the main codes for moving the cursor are:

* echo -e "\e[1;1H" → moves the cursor to (1,1), that's the top left corner of the terminal (you can change the numbers to move somewhere else)
* echo -e "\e[2J" → Clear the terminal

Reply

(hide), 2018/08/09 09:04

Blink work in xfce4-terminal 0.8.7.4 (Xfce 4.12). Testined 'echo -e "\e[1;5m Bold+Blink \e[0m"'.

Reply

Fabien LOISON, 2018/08/09 09:09

You are right, it seems it is now implemented in VTE, so it works in all VTE-based terminals (GNOME Terminal, Tilda, Terminator,...).

Thanks for the update :) Reply

Real name:

E-Mail:

Enter your comment:

Save Subscribe to comments Preview

bash/tip_colors_and_formatting.txt · Last modified: 2018/08/09 09:14 by flozz