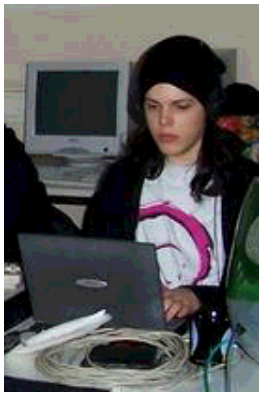
LinuxFocus article number 335
http://linuxfocus.org

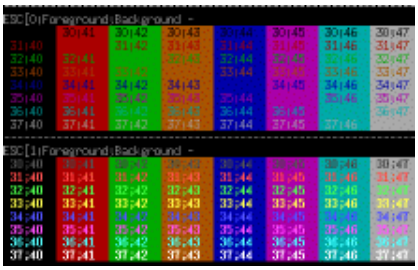


by Nico Golde
<nico/at/ngolde.de>

About the author:

At present Nico is still a student. Since a few years he keeps very busy with Linux, he also takes part in a number of Open Source Projects.

Colorful Shells -- Using ANSI Color Codes



Abstract:

In an ANSI compatible terminal (like xterm, rxvt, konsole ...) text may be shown in colors different from black/white. This article will demonstrate text in bold or in color.

Translated to English by:
Jürgen Pohl
<sept.sapins(at)verizon.net>

General

In real life every Linux user gets to face the Bash. At first glance that looks very boring, but there are many possibilities to give one's shell the personal touch. Colored enhancement of the shell prompt make the shell unique as well as better legible.

In my description I will refer to the Bash shell. The escape sequences may differ between terminals, for this text I am using an ANSI terminal.

Configuration of the Shell

Setting of shell colors happens in the personal configuration file of the bash *~/.bashrc* or in the global configuration file */etc/bashrc*. The appearance of the prompt is being set with the PS1 variable in

bashrc.

Generally, the entry should look like this:

```
~/.bashrc: PS1="\s-\v\$ "
```

\s stands for the name of the shell and -\v for its version. At the end of the prompt we are placing a $. Since this gets a bit boring, the following entry - which is default in most Linux distributions - may be used:

```
~/.bashrc: PS1="\u@\h \w \$ "
```

This stands for user@ current_directory $, which is the normal shell prompt most Linux users are familiar with.

Escape Sequences

To add a personal touch by coloring the prompt we are using escape sequences. An escape sequence is a control instruction which orders the shell to execute a specific step. An escape sequence usually begins with ESC (thus the name). In the shell chown as ^[. This way of writing needs a bit getting used to, \033 accomplishes the same (ESC is ascii 27 decimal = 33 octal).

To enter an escape sequence directly into the shell we have to precede it with crtl-v: *CTRL-v ESC*.

Using the Colors of the Shell

I am going to explain the colors of the shell with an example prompt.

```
~/.bashrc: PS1="\[\033[0;32;40m\u@\h:\w\$ \]"
```

This displays the complete prompt in green. Like this:

nico@ebrain:~$

\033 starts the escape sequence, with [we are beginning the color definition. The following 0 specifies default font width. Other possibilities for this I am going to introduce later. The string will be enclosed in \[and \] to prevent the text of the escape sequence from showing up in the display of the shell and taking too much space.

Next we are choosing the color of the foreground (in this case 32, which is green). The background color 40 stands for black. To prevent the text after the prompt from being green, we are closing the escape sequence with \033[0m, which is the default color of the shell. For the foreground as well as the background 8 colors are available.

Choices: red, green, yellow, blue, magenta, cyan and white. The color codes for this are 30 (black), 31 (red), 32 (green), 33 (yellow), 34 (blue), 35 (magenta), 36 (cyan), 37 (white).

Setting the background colors follows the same scheme, but instead of the first digit ’3’ we are using ’4’, like 40, 41, 42, 43, 44, 45, 46, 47.

Example:

```
~/.bashrc: PS1="\[\033[0;37;44m\u@\033[0;32;43m\h:\033[0;33;41m\w$\033[0m\]"
```

This gives us a very colorful prompt:

nico@ebrain:~$

To test these settings we are using *export PS1="string"*, later we may transfer the setting into *.bashrc*. My current prompt looks like this:

```
PS1="\[\033[1;34;40m[\033[1;31;40m\u@\h:\w\033[1;34;40m]\033[1;37;40m $\033[0;37;0m\
```

[nico@ebrain:~]

Text Properties

As previously mentioned, the ’0’ after the first escape sequence is the default color setting for the text of the shell prompt. For the text properties the following values make sense: 0, 1, 22, 4, 24, 5, 25, 7, 27 with the following meaning: default, bold, not bold, underlined, not underlined, blinking and not blinking, invers, not invers.

With the help of the following short script we can have a look at the color combinations.

```
#!/bin/sh
############################################################
# Nico Golde <nico(at)ngolde.de> Homepage: http://www.ngolde.de
# Last change: Mon Feb 16 16:24:41 CET 2004
############################################################

for attr in 0 1 4 5 7 ; do
    echo "-------------------------------------------------------------------"
    printf "ESC[%s;Foreground;Background - \n" $attr
    for fore in 30 31 32 33 34 35 36 37; do
        for back in 40 41 42 43 44 45 46 47; do
            printf ’\033[%s;%s;%sm %02s;%02s  ’ $attr $fore $back $fore $back
        done
    printf ’\n’
    done
    printf ’\033[0m’
done
```

The script can be downloaded as a *tar.gz* from: showansicol.tar.gz

Another Application

The ability to set colors in the shell is not only useful to create a more beautiful shell prompt but may also be beneficial for the programming of a program for the console.

For every use of colors the use of libraries such as *slang* or *ncurses* would be necessary, this would greatly inflate the size of the binary file. *Ncurses* has the advantage of being more or less independant from the type of terminal.

Examples in C

A ’Hello World’ in green text:

```
#include <stdio.h>
int main(void){
  const char *const green = "\033[0;40;32m";
  const char *const normal = "\033[0m";
  printf("%sHello World%s\n", green, normal);
  return 0;
}
```

Another useful escape sequence is *printf("\033[2J")*, it has the same effect like *system(clear)* but the header file *unistd.h* can be left off.

With *printf("\033[1K"*) we can delete a line.

Examples for init-Scripts

If we would like to get a colorful, good legible confirmation of the successful start of the *init* scripts of */etc/init.d* instead of simply ’.’ we may accomplish that again with an escape sequence.

This is an excerpt from a *cron init script*:

```
#!/bin/sh
# Start/stop the cron daemon.
test -f /usr/sbin/cron || exit 0

 case "$1" in
 start)  echo -n "Starting periodic command scheduler: cron"
     start-stop-daemon --start --quiet --exec /usr/sbin/cron

                 echo "."
;;
```

The successful start of *cron* will be shown with a period. The color feature of this could be handled by [Ok] through a change of the *echo* string, e.g.:

```
#!/bin/sh
# Start/stop the cron daemon.
test -f /usr/sbin/cron || exit 0
case "$1" in
start)  echo -n "Starting periodic command scheduler: cron"
        start-stop-daemon --start --quiet --exec /usr/sbin/cron
echo "\[ \033[1;34;40m[ \033[1;32;40mOk \033[1;34;40m]\033[0m\]"
```

```
;;
```

Applying these settings to all *ini*t scripts is very time consuming, except we are using the escape sequence \033 - since *Ctrl-v* which is not interpreted as a character.

Feedback

Feedback, critique, bugs, etc. please mail to "nico at ngolde.de". Have fun...

2005-01-14, generated by lfparser_pdf version 2.51