**Customizing bash prompt**

On a Linux command line terminal, the prompt is the text to the left of the commands you enter. The default prompt varies for every system, but it usually gives you an indication of your username, your machine’s host name and your current working directory etc. Also, it ends with a dollar sign $ if you’re working as a normal user. If you’re working with root privileges, it ends with # instead.

The prompt looks can be customized to include relevant information. This article will show how to customize your bash prompt. Bash is the default shell on most Linux distributions.

Bash primarily uses two variables that define how your prompt looks like:

**PS1** specifies the format of your regular prompt that appears to the left of every new command you type

**PS2** defines the continuation prompt: It appears when you enter a line ending with a backslash to continue input on the next line.

Customization of regular prompt:

As discussed earlier regular prompt can be customize using variable PS1. You can assign different values to this variable in your shell session. Default value of PS1 is ‘[\u@\h \W]\$’ for a non-root user on most the Linux distribution.

Any character followed by a back slash is special character for bash and they are decoded as follows:

\a an ASCII bell character (07)

\d the date in "Weekday Month Date" format

(e.g., "Tue May 26")

\e an ASCII escape character (033)

\h the hostname up to the first `.'

\H the hostname

\j the number of jobs currently managed by the

shell

\l the basename of the shell's terminal device

name

\n newline

\r carriage return

\s the name of the shell, the basename of $0

(the portion following the final slash)

\t the current time in 24-hour HH:MM:SS format

\T the current time in 12-hour HH:MM:SS format

\@ the current time in 12-hour am/pm format

\u the username of the current user

\v the version of bash (e.g., 2.00)

\V the release of bash, version + patchlevel

(e.g., 2.00.0)

\w the current working directory

\W the basename of the current working direc­

tory

\! the history number of this command

\# the command number of this command

\$ if the effective UID is 0, a #, otherwise a

$

\nnn the character corresponding to the octal

number nnn

\\ a backslash

\[ begin a sequence of non-printing characters,

which could be used to embed a terminal con­

trol sequence into the prompt

\] end a sequence of non-printing characters

Non-printing escape sequences have to be enclosed in \[\e[ and \].

So for user ‘guest’ on host ‘mypc’ in user’s home directory, ‘[\u@\h \W]\$’ will be decoded as

[guest@mypc ~]$

Colored prompt:

You can change color of prompt using escape character ‘\e’. For example following PS1 value will change color of your prompt to light green:

PS1='\[\e[1;32m\][\u@\h \W]\$\[\e[0m\]'

The string above contains color-set escape sequences (start coloring: \[\e[color\], end coloring: \[\e[m\])

As mentioned before, non-printing escape sequences have to be enclosed in \[\e[ and \]. For color escape sequences, color code should also be followed by a lowercase m.

Other color codes supported by bash is give below:

Black 0;30 Dark Gray 1;30

Blue 0;34 Light Blue 1;34

Green 0;32 Light Green 1;32

Cyan 0;36 Light Cyan 1;36

Red 0;31 Light Red 1;31

Purple 0;35 Light Purple 1;35

Brown 0;33 Yellow 1;33

Light Gray 0;37 White 1;37

You can also set background color of prompt by using 41 for a Red background, 44 for Blue background, and 45 for a pink background, etc. There are no bold background colors. Combinations can be used, like Light Red text on a Blue background: \[\e[44;1;31m\], although setting the colors separately seems to work better (ie. \[\e[44m\]\[\e[1;31m\]).

Other codes:

Other codes are available to use with escape character. You can include following codes with escape character.

4: Underscore,

5: Blink,

7: Inverse, and

8: Concealed

Customization of continuation prompt:

As discussed earlier continuation prompt can be customize using variable PS2. Here we’ll see, How does PS2 work with continuation prompt? For example if we set value of PS2=’continue>’ and run ‘pwd’ command in continuation the prompt will look like:

[guest@mypc ~]$pw\

continue>d

/home/guest

[guest@mypc ~]$

All the above rules explained for PS1 are also applicable to PS2.

Running user script with every prompt:

If you want to run your own command depending on some conditions, you need to put that part of your prompt into the variable PROMPT\_COMMAND. It is expected to contain a script that is executed just before PS1 is printed. For example following PROMPT\_COMMAND prints non zero exit code returned by a program executed on the prompt:

PROMPT\_COMMAND='RET=$?; if [ $RET != 0 ] ; then echo "rc: $RET"; fi'

Colorful directory listing:

Although colorful directory listing is not in control of bash, I am discussing this in this article as people often think it is under control of shell. Colorful listing of directory is done by the ‘ls’ command itself. For colorful listing ‘ls --color=auto’ command is used. In default ‘ls’ command, color is disabled. To make directory listing by default colorful create an alias of ‘ls --color=auto’ to ‘ls’ and save it in your .bashrc file. The alias command is given below:

alias ls='ls --color=auto'

Persisting changes:

Your prompt is not kept across all shell sessions when you simply set PS1, PS2 and/or PROMPT\_COMMAND in your shell. To keep your changes across sessions and restarts of your machine, add the relevant assignments of PS1, PS2 and PROMPT\_COMMAND to your .bash\_profile or .bashrc file in your home directory. Save the file, open a new bash session, and enjoy your custom prompt!