**How to: Change / Setup bash custom prompt (PS1)**

Most of us work with a shell prompt. By default most Linux distro displays hostname and current working directory. You can easily customize your prompt to display information important to you. You change look and feel by adding colors. In this small howto I will explain howto setup:  
a] Howto customizing a bash shell to get a good looking prompt  
b] Configure the appearance of the terminal.  
c] Apply themes using bashish  
d] Howto pimp out your shell prompt

Prompt is control via a special shell variable. You need to set PS1, PS2, PS3 and PS4 variable. If set, the value is executed as a command prior to issuing each primary prompt.

* **PS1** – The value of this parameter is expanded (see PROMPTING below) and used as the primary prompt string. The default value is **\s-\v\$** .
* **PS2**– The value of this parameter is expanded as with PS1 and used as the secondary prompt string. The default is **>**
* **PS3** – The value of this parameter is used as the prompt for the select command
* **PS4** – The value of this parameter is expanded as with PS1 and the value is printed before each command bash displays during an execution trace. The first character of PS4 is replicated multiple times, as necessary, to indicate multiple levels of indirection. The default is **+**

**How do I display current prompt setting?**

Simply use echo command, enter:  
$ echo $PS1  
Output:

\\u@\h \\W]\\$

**How do I modify or change the prompt?**

Modifying the prompt is easy task. Just assign a new value to PS1 and hit enter key:  
My old prompt –> [vivek@105r2 ~]$  
PS1="touch me : "  
Output: My new prompt

touch me :

So when executing interactively, bash displays the primary prompt PS1 when it is ready to read a command, and the secondary prompt PS2 when it needs more input to complete a command. Bash allows these prompt strings to be customized by inserting a number of backslash-escaped special characters that are decoded as follows:

* **\a** : an ASCII bell character (07)
* **\d** : the date in “Weekday Month Date” format (e.g., “Tue May 26”)
* **\D{format}** : the format is passed to strftime(3) and the result is inserted into the prompt string; an empty format results in a locale-specific time representation. The braces are required
* **\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 shellâ€™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
* **\A** : the current time in 24-hour HH:MM format
* **\u** : the username of the current user
* **\v** : the version of bash (e.g., 2.00)
* **\V** : the release of bash, version + patch level (e.g., 2.00.0)
* **\w** : the current working directory, with $HOME abbreviated with a tilde
* **\W** : the basename of the current working directory, with $HOME abbreviated with a tilde
* **\!** : 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
* **\[** : begin a sequence of non-printing characters, which could be used to embed a terminal control sequence into the prompt
* **\]** : end a sequence of non-printing characters

Let us try to set the prompt so that it can display todayâ€™d date and hostname:  
**PS1="\d \h $**"  
Output:

Sat Jun 02 server $

Now setup prompt to display date/time, hostname and current directory:  
$ **PS1="[\d \t \u@\h:\w ] $** "  
Output:

[Sat Jun 02 14:24:12 vivek@server:~ ] $

**How do I add colors to my prompt?**

You can change the [color of your shell prompt](https://www.cyberciti.biz/faq/bash-shell-change-the-color-of-my-shell-prompt-under-linux-or-unix/) to impress your friend or to make your own life quite easy while working at command prompt.

**Putting it all together**

Let us say when you login as root/superuser, you want to get visual confirmation using red color prompt. To distinguish between superuser and normal user you use last character in the prompt, if it changes from $ to #, you have superuser privileges. So let us set your prompt color to RED when you login as root, otherwise display normal prompt.

Open /etc/bashrc (Redhat and friends) / or /etc/bash.bashrc (Debian/Ubuntu) or /etc/bash.bashrc.local (Suse and others) file and append following code:  
# vi /etc/bashrc  
or  
$ sudo gedit /etc/bashrc  
Append the code as follows

# If id command returns zero, youâ€™ve root access.

if [ $(id -u) -eq 0 ];

then # you are root, set red colour prompt

PS1="\\[$(tput setaf 1)\\]\\u@\\h:\\w #\\[$(tput sgr0)\\]"

else # normal

PS1="[\\u@\\h:\\w] $"

fi

Close and save the file.

You can also create complex themes for your bash shell using bashish. [Bashish](http://bashish.sourceforge.net/) is a theme enviroment for text terminals. It can change colors, font, transparency and background image on a per-application basis. Additionally Bashish supports prompt changing on common shells such as bash, zsh and tcsh. Install bashish using rpm or apt-get command:  
# rpm -ivh bashish\*  
OR  
# dpkg -i bashish\*  
Now start bashish for installing user configuration files:  
$ bashish  
Next you must restart your shell by typing the following command:  
$ exec bash  
To configure the Bashish theme engine, run  
$ bashishtheme

Finally, you can always use [aterm](http://www.afterstep.org/aterm.php) or other terminal program such as rxvt. It supports nice visual effect , like transparency, tinting and much more by visiting profile menu. Select your terminal > click on Edit menu bar > Profiles > Select Profile > Click on Edit button > Select Effects tab > Select transparent background > Close

**My Bash Prompt**

In bash there are several shortcuts you can use. My prompt looks like this:   
  
PS1="[\u@\h \w]\$ "  
  
which produces   
[username@server workingdir]$   
  
You can add color to them as well. There are a number of variables you may use in bash.  
  
\a Alarm (Bell)  
\d Date in fomat "Wed Jun 21"  
\h Hostname  
\j Jobs (Number of Background jobs)  
\l Current Terminal Name  
\n Newline inserted into prompt. (Usefull for 2-line prompts)  
\r Carriage return in prompt  
\s Current Shell  
\t Time in 24 hour format, 16:25:00  
\T Time in 12 hour format, 04:25:00  
\u User's account name  
\v Version and release of bash  
\w Current working directory (full path /home/bill/.mozilla)  
\W Last element of Working directory (.mozilla)  
\V Version, release and patch level of bash  
\\ single backslash inserted into prompt  
\! Number of current command in command history  
\# Number of current command where numbers start at 1 when shell starts  
\A Time in 24 hour format without seconds 16:25  
\@ Time in 12 hour format without seconds, 04:25 p.m.  
\$ indicates root. display $ when not root and # when root.   
\[ starts a sequence of non printing characters ending with \] (explanation to follow)  
\nnn the character in ascii set coresponding to the octal number nnn  
  
The last two commands can add color to the prompt. For example,   
  
\[\033[1;36m\] changes the color to a light blue. \[\033[0m\] changes it to back to off-white. On a standard black background, white letters I use  
PS1="[\[\033[1;36m\]\u\[\033[0m\]@\[\033[1;35m\]\h \[\033[1;32m\]\w\[\033[0m\]]\$ "   
which gives me   
[bill@server /usr/local/bin]$  
  
The color changes are confusing, I know, but I like it.   
  
;