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

last updated June 2, 2007 in **Debian Linux, Download of the day, Linux, Linux desktop, Linux distribution, Shell scripting, Suse Linux, Tips, Tuning, Ubuntu Linux, UNIX**

So how do you setup, change and pimp out Linux / UNIX shell prompt?

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 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
```

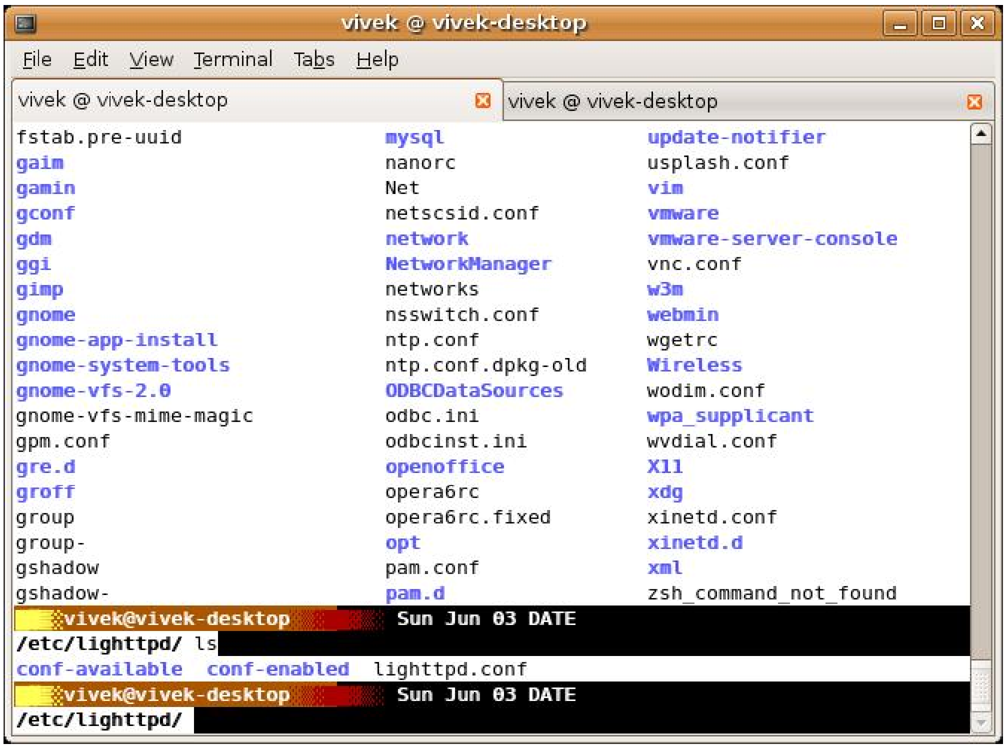
Append the code as follows

```
# 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.

My firepower prompt

Check this out:



```
fstab.pre-uuid          mysql                  update-notifier
gaim                    nanorc                 usplash.conf
gamin                   Net                    vim
gconf                   netscsid.conf          vmware
gdm                     network                vmware-server-console
ggi                     NetworkManager         vnc.conf
gimp                    networks               w3m
gnome                   nsswitch.conf          webmin
gnome-app-install       ntp.conf               wgetrc
gnome-system-tools      ntp.conf.dpkg-old      Wireless
gnome-vfs-2.0           ODBCDataSources        wodim.conf
gnome-vfs-mime-magic    odbc.ini               wpa_supplicant
gpm.conf                odbcinst.ini           wvdial.conf
gre.d                   openoffice             X11
groff                   opera6rc               xdg
group                   opera6rc.fixed         xinetd.conf
group-                  opt                    xinetd.d
gshadow                 pam.conf               xml
gshadow-                pam.d                  zsh_command_not_found
  vivek@vivek-desktop          Sun Jun 03 DATE
/etc/lighttpd/ ls
conf-available  conf-enabled  lighttpd.conf
  vivek@vivek-desktop          Sun Jun 03 DATE
/etc/lighttpd/
```

(click to enlarge)

You can also create complex themes for your bash shell using bashish. Bashish 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
```