bashrc shell control

le fichier ~/.bashrc est lu à chaque fois que vous ouvrez votre terminal. c'est le fichier de configuration de votre console si vous êtes utilisateur de **bash**.

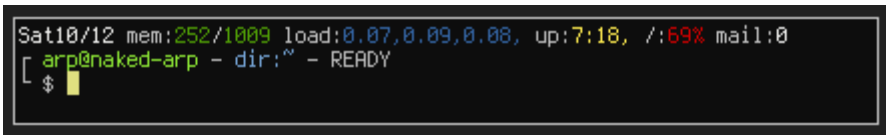
ce fichier permet tellement de choses que je vous montrerais quelques exemples mais je vous laisse consulter ces liens pour une documentation complète:

- bash reference(gb).
- prompt color(gb) sur le wiki archlinux
- manpages-fr de la commande bash

Sommaire

bashrc shell control....1
  prompt color....1
    les variables du prompt....2
    les couleurs du prompt....3
  bash functions()....4
    find function....4
    extract function....4
    space function....4
    infos box function....5
conclusion....6

prompt color



prompt_color

lorsque vous ouvrez votre terminal sur GNU/Linux, l'invite de commande se présente sous la forme "**user@host:dir $**", je vais vous montrer comment personaliser votre invite de commande ou **prompt** et obtenir ceci:

la première ligne ne s'affiche qu'à l'ouverture du terminal, la seconde constitue le prompt. pour parvenir à ce résultat, voici le code à ajouter dans votre ~/.bashrc:

```
## prompt --------------------------------------------------------------
# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "$debian_chroot" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi
# set a fancy prompt (non-color, unless we know we "want" color)
case "$TERM" in
    xterm-color) color_prompt=yes;;
esac
```

```
# uncomment for a colored prompt, if the terminal has the capability; turned
# off by default to not distract the user: the focus in a terminal window
# should be on the output of commands, not on the prompt
force_color_prompt=yes

if [ -n "$force_color_prompt" ]; then
    if [ -x /usr/bin/tput ] && tput setaf 1 >&/dev/null; then
     # We have color support; assume it's compliant with Ecma-48
     # (ISO/IEC-6429). (Lack of such support is extremely rare, and such
     # a case would tend to support setf rather than setaf.)
     color_prompt=yes
    else
     color_prompt=
    fi
fi
## chars
## ├ └ ┌ │ ─ ┬
LOAD=`uptime | awk '{print $8 $9 $10}'`
UPT=`uptime | awk '{print $3}'`
DATE=`date +"%a%d/%m"`
TIME=`date +"%H:%M:%S"`
ROOT=`df -h|grep sda3|awk '{print $5}'`
MEMU=`free -m | sed -n 's|^-.*:[ \t]*\([0-9]*\) .*|\1|gp'`
MEMT=`free -m | sed -n 's|^M.*:[ \t]*\([0-9]*\) .*|\1|gp'`
# mail function depends on claws-mail
if [ "$(pidof claws-mail)" ];then
    MAIL=`claws-mail --status | awk '{print $2}'`
else
    MAIL=offline
fi
if [ "$color_prompt" = yes ]; then
    echo -en "\e[1;37m$DATE\e[m mem:\e[0;32m$MEMU\e[m/\e[0;32m$MEMT\e[m
load:\e[0;34m$LOAD,\e[m up:\e[1;33m$UPT\e[m /:\e[0;31m$ROOT\e[m
mail:\e[1;37m$MAIL\e[m\n"
    PS1='${debian_chroot:+($debian_chroot)}┌ \e[01;32m\u@\h\e[m -
\e[01;34mdir:\w\e[m - READY\n└ $ '
else
    PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi
unset color_prompt force_color_prompt

# If this is an xterm set the title to user@host:dir
case "$TERM" in
xterm*|rxvt*)
    PS1="\[\e]0;${debian_chroot:+($debian_chroot)}\u@\h: \w\a\]$PS1"
    ;;
*)
    ;;
esac
## ----------------------------------------- end of prompt section ##
```

les variables du prompt

bash autorise la personalisation du prompt et accepte les options suivantes (tiré du man fr bash):

séquence	définition
\a	le caractère d'alarme ASCII 07
\d	la date au format « Jour_de_la_semaine Mois Quantième » (p.ex. : “Tue May 26”)
\D{format}	le format est passé à strftime(3) et le résultat est inséré dans la chaîne du prompt: un format vide implique une représentation du temps spécifique aux paramètres régionaux. les accolades sont nécessaires.
\e	le caractère d'échappement ASCII 033
\h	le nom d'hôte de la machine, juqu'au premier point “.”
\H	le nom d'hôte complet de la machine
\n	un saut de ligne
\r	un retour-chariot
\s	le nom du shell, c'est-à-dire le nom de base de $0 (la portion suivant le dernier slash)
\t	l'heure actuelle au format HH:MM:SS sur 24 heures
\T	l'heure actuelle au format HH:MM:SS sur 12 heures
\@	l'heure actuelle sur 12 heures au format HH:MM am/pm
\A	l'heure actuelle au format HH:MM sur 24 heures
\u	le nom de l'utilisateur
\v	la version de bash (par exemple 2.04)
\V	le numéro de version complet (avec niveau de correctifs) de bash, par exemple (2.04.0)
\w	le répertoire de travail en cours, avec $HOME abbrégé en tilde (“~”)
\W	le nom de base du répertoire de travail en cours, avec $HOME abbrégé en tilde (“~”)
\!	le numéro d'historique de la commande
\#	le numéro de la commande
\$	# si l'UID effectif est 0, $ sinon
\nnn	le caractère de code octal nnn
\\	le caractère antislash
\[	débute une série de caractères non-imprimables, qui permettent d'inclure des séquences de contrôle de terminal dans une chaîne d'accueil
\]	fin d'une série de caractères non-imprimables

les couleurs du prompt

les couleurs sont passées au prompt à la suite de la chaîne *\033* ou *\e*. voici une liste des couleurs et leurs codes.

couleur/style	normal	underline	bold	background	intense	bold intense	background intense
black	\e[0;30m	\e[4;30m	\e[1;30m	\e[40m	\e[0;90m	\e[1;90m	\e[1;100m
red	\e[0;31m	\e[4;31m	\e[1;31m	\e[41m	\e[0;91m	\e[1;91m	\e[1;101m
green	\e[0;32m	\e[4;32m	\e[1;32m	\e[42m	\e[0;92m	\e[1;92m	\e[1;102m
yellow	\e[0;33m	\e[4;33m	\e[1;33m	\e[43m	\e[0;93m	\e[1;93m	\e[1;103m
blue	\e[0;34m	\e[4;34m	\e[1;34m	\e[44m	\e[0;94m	\e[1;94m	\e[1;104m
purple	\e[0;35m	\e[4;35m	\e[1;35m	\e[45m	\e[0;95m	\e[1;95m	\e[1;105m
cyan	\e[0;36m	\e[4;36m	\e[1;36m	\e[46m	\e[0;96m	\e[1;96m	\e[1;106m
white	\e[0;37m	\e[4;37m	\e[1;37m	\e[47m	\e[0;97m	\e[1;97m	\e[1;107m
reset color	\e[0m						

bash functions()

les fonctions de bash sont des scripts intégrés à votre ~/.bashrc et donc immédiatemet accessibles depuis votre terminal grâce à des alias. je vous présente ici quelques exemples explicatifs issus de ce fabuleux bashrc.

find function

cette fonction permet de trouver des fichiers de façon récursive dans le répertoire courant selon une expression passée en argument. alias **ff** :

le code pour ~/.bashrc

```
# find ----------------------------------------------------------------
function ff() { find . -type f -iname '*'$*'*' -ls ; }
```

extract function

cette fonction permet d'extraire une archive selon son extension dans le répertoire de travail. alias **extract**.

le code pour ~/.bashrc

```
# extract archives ----------------------------------------------------
function extract()
{
     if [ -f $1 ] ; then
         case $1 in
             *.tar.bz2)   tar xvjf $1     ;;
             *.tar.gz)    tar xvzf $1     ;;
             *.bz2)       bunzip2 $1      ;;
             *.rar)       unrar x $1      ;;
             *.gz)        gunzip $1       ;;
             *.tar)       tar xvf $1      ;;
             *.tbz2)      tar xvjf $1     ;;
             *.tgz)       tar xvzf $1     ;;
             *.zip)       unzip $1        ;;
             *.Z)         uncompress $1   ;;
             *.7z)        7z x $1         ;;
             *)           echo "'$1' cannot be extracted via >extract<" ;;
         esac
     else
         echo "'$1' is not a valid file"
     fi
}
```

il existe plusieurs variantes de cette fonction que vous pouvez trouver facilement sur le web.

space function

cette fonction génère un rapport d'utilisation d'espace disque et l'envoi dans un fichier texte. alias **space**

```
# generate space report ----------------------------------------------
function space() { du -skh * | sort -nr > $HOME/space_report.txt ; }
```

infos box function

il existe plusieurs moyens d'afficher ses infos système: conky, htop, widgets, applications type lxtasks... voici une fonction qui permet de regrouper ces informations pour les rendre accessibles de manière ordonnée depuis votre terminal. alias **ii**.

le code à insérer dans votre ~/.bashrc

```
# infos box ------------------------------------------------------------
# text colors
red='\e[0;31m'
blue='\e[0;34m'
cyan='\e[0;36m'
green='\e[0;32m'
yellow='\e[0;33m'
NC='\e[0m'              # No Color
# background colors
RED='\e[41m'
BLUE='\e[44m'
CYAN='\e[46m'
GREEN='\e[42m'
YELLOW='\e[43m'
```

```
function my_ps() { ps $@ -u $USER -o pid,%cpu,%mem,bsdtime,command ; }

function pp() { my_ps f | awk '!/awk/ && $0~var' var=${1:-".*"} ; }

function my_ip()
{ MY_IP=$(/sbin/ifconfig eth0 | awk '/inet/ { print $2 } ' | sed -e s/addr://) ;
}
EXTIP=`wget -q -O - checkip.dyndns.org | sed -e 's/[^[:digit:]\|.]//g'`
function ii()
{
    echo -e "${RED} You are logged on:$NC " ; hostname
    echo -e "${RED} Additionnal information:$NC " ; uname -a
    echo -e "${RED} Users logged on:$NC " ; w -h
    echo -e "${RED} Current date :$NC " ; date
    echo -e "${CYAN} Machine stats :$NC " ; uptime
    echo -e "${GREEN} Memory stats :$NC " ; free
    my_ip 2>&- ;
    echo -e "${BLUE} Local IP Address :$NC" ; echo ${MY_IP:-"Not connected"}
    echo -e "${BLUE} External IP Address :$NC" ; echo ${EXTIP:-"Not connected"}
    echo -e "${BLUE} Mails :$NC" ; echo ${MAIL:-"Not connected"}
    echo -e "${BLUE} Open connections :$NC "; netstat -pan --inet;
    echo
}
```

vous aurez compris qu'il est assez simple de configurer soi-même des fonctions bash (f)utiles. :)

@suivre...

conclusion

bash est un outil extrêment performant dont je ne ferais jamais le tour, mais je tenterais d'alimenter ce wiki en exemples pratiques et/ou originaux pour vous faire apprécier cette merveille qu'est la CLI. :D

@+

---

contributeur: arpinux