```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi

# Define some colors first:
red='\e[0;31m'
RED='\e[1;31m'
blue='\e[0;34m'
BLUE='\e[1;34m'
cyan='\e[0;36m'
CYAN='\e[1;36m'
GREEN='\e[0;32'
YELLOW='\e[1;33'
NC='\e[0m'              # No Color

# User specific aliases and functions
#-------------------------------------------------------------
# The 'ls' family
#-------------------------------------------------------------
alias ll="ls -l --group-directories-first"
alias la='ls -Al'          # show hidden files
alias lx='ls -lXB'         # sort by extension
alias lk='ls -lSr'         # sort by size, biggest last
alias lc='ls -ltcr'        # sort by and show change time, most recent last
alias lu='ls -ltur'        # sort by and show access time, most recent last
alias lt='ls -ltr'         # sort by date, most recent last
alias lm='ls -al |more'    # pipe through 'more'
alias lr='ls -lR'          # recursive ls
alias tree='tree -Csu'     # nice alternative to 'recursive ls'
alias h='history'
alias j='jobs -l'
alias which='type -a'
alias ..='cd ..'
alias du='du -kh'       # Makes a more readable output.
alias dk='df -kTh'
alias lot='lsof -i tcp'  #List all open files using tcp
alias lou='lsof -i tcp'  #List all open files using udp


#-----------------------------------------------------------
# Functions - Handy functions to have
#-----------------------------------------------------------
function helpme()
{
        echo -e "\n"
        echo -e "\t${BLUE}List of Commands and Aliases$NC"
        echo -e "\n"
        echo -e "\tAliases"
        echo -e "-----------------------------------------------------------------"
        echo -e "${RED}ll$NC        \t\tshow listing with directories first"
        echo -e "${RED}la$NC        \t\tshow hidden files"
```

```
       echo -e "${RED}lx$NC\t\tsort by extension"
       echo -e "${RED}lk$NC\t\tsort by size, biggest last"
       echo -e "${RED}lc$NC\t\tsort by and show change time, most recent last"
       echo -e "${RED}lu$NC\t\tsort by and show access time, most recent last"
       echo -e "${RED}lt$NC \t\tsort by date, most recent last"
       echo -e "${RED}lm$NC         \t\tpipe through 'more'"
       echo -e "${RED}lr$NC \t\trecursive ls"
       echo -e "${RED}tree$NC       \t\tnice alternative to 'recursive ls'"
       echo -e "${RED}h$NC \t\thistory"
       echo -e "${RED}j$NC  \t\tlist jobs"
         echo -e "${RED}..$NC        \t\tcd .."
         echo -e "${RED}du$NC       \t\tmakes format more readable"
         echo -e "${RED}dk$NC       \t\teasier format and more information"
       echo -e "${RED}lot$NC   \t\tList all open files using tcp"
       echo -e "${RED}lou$NC        \t\tList all open files using udp"
         echo -e "\n"
         echo -e "\tCommands"
         echo -e "----------------------------------------------------------------------"
         echo -e "extract <filename>\t\textract any archive"
         echo -e "change2user <username>\t\tsudo to a user and setup X for remote
display"
         echo -e "change2root                   \t\tsudo to root with X display setup for remote
export"
         echo -e "lowercase             \t\tmove filenames to lowercase"
         echo -e "killps                \t\tkill process by name"
         echo -e "repeat        <N> <command>    \t\trepeat command N times"
         echo -e "corename <core>  \t\tGet name of app that created a corefile"
         echo -e "ii                          \t\tGet information about current host"
         echo -e "\n"
}
function extract()      # Handy Extract Program.
{
    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
}

function change2user() #Handy for exporting displays as a different user than the one you
logged in as
```

```
{
    echo "Changing permissions on Xauthority file."
    chmod 644 ~/.Xauthority
    echo -n "Getting user shell...."
    NEWUSERSHELL=`ypcat passwd | grep $1 | awk -F: '{ print $7 }'`
    echo $NEWUSERSHELL
    XAUTHORITY=$HOME/.Xauthority DISPLAY=$DISPLAY sudo su - $1 $NEWUSERSHELL
}

function change2root() #Handy for exporting the display when you su - root
{
    echo "Changing permissions on Xauthority file."
    chmod 644 ~/.Xauthority
    echo -n "Getting user shell...."
    NEWUSERSHELL=/bin/bash
    echo $NEWUSERSHELL
    XAUTHORITY=$HOME/.Xauthority DISPLAY=$DISPLAY sudo su -
}

function lowercase()  # move filenames to lowercase
{
   for file ; do
      filename=${file##*/}
      case "$filename" in
      */*) dirname==${file%/*} ;;
      *) dirname=.;;
      esac
      nf=$(echo $filename | tr A-Z a-z)
      newname="${dirname}/${nf}"
      if [ "$nf" != "$filename" ]; then
         mv "$file" "$newname"
         echo "lowercase: $file --> $newname"
      else
         echo "lowercase: $file not changed."
      fi
   done
}

function killps()                 # Kill by process name.
{
   local pid pname sig="-TERM"   # Default signal.
   if [ "$#" -lt 1 ] || [ "$#" -gt 2 ]; then
      echo "Usage: killps [-SIGNAL] pattern"
      return;
   fi
   if [ $# = 2 ]; then sig=$1 ; fi
   for pid in $(my_ps| awk '!/awk/ && $0~pat { print $1 }' pat=${!#} ) ; do
      pname=$(my_ps | awk '$1~var { print $5 }' var=$pid )
      if ask "Kill process $pid <$pname> with signal $sig?"
         then kill $sig $pid
      fi
   done
}
function reboot()
```

```
{
   if ask "Are you sure you wish to reboot `hostname`?"
      then echo "rebooting"
   fi
}

function my_ip() # Get IP adresses.
{
   MY_IP=$(/sbin/ifconfig eth0 | awk '/inet/ { print $2 } ' | \
sed -e s/addr://)
   MY_ISP=$(/sbin/ifconfig eth0 | awk '/P-t-P/ { print $3 } ' | \
sed -e s/P-t-P://)
}

function ii()   # Get current host related info.
{
   echo -e "\nYou are logged on ${RED}$HOST"
   echo -e "\nAdditionnal information:$NC " ; uname -a
   echo -e "\n${RED}Users logged on:$NC " ; w -h
   echo -e "\n${RED}Current date :$NC " ; date
   echo -e "\n${RED}Machine stats :$NC " ; uptime
   echo -e "\n${RED}Memory stats :$NC " ; free
   my_ip 2>&- ;
   echo -e "\n${RED}Local IP Address :$NC" ; echo ${MY_IP:-"Not connected"}
   echo -e "\n${RED}ISP Address :$NC" ; echo ${MY_ISP:-"Not connected"}
   echo -e "\n${RED}Open connections :$NC "; netstat -pan --inet;
   echo
}

function repeat()       # Repeat n times command.
{
   local i max
   max=$1; shift;
   for ((i=1; i <= max ; i++)); do  # --> C-like syntax
      eval "$@";
   done
}


function ask()          # See 'killps' for example of use.
{
   echo -n "$@" '[y/n] ' ; read ans
   case "$ans" in
      y*|Y*) return 0 ;;
      *) return 1 ;;
   esac
}

function corename()   # Get name of app that created a corefile.
{
   for file ; do
      echo -n $file : ; gdb --core=$file --batch | head -1
   done
}
```