Configuration du fichier Linux Bashrc

Ce fichier `bashrc` configure l'environnement du shell Bash sous Linux. Il personnalise l'invite de commande, définit des alias, gère les paramètres proxy et s'intègre à des outils comme Git. Voici une description des configurations clés :

1. **Paramètres de base :**

   - `HISTCONTROL=ignoreboth` : Ignore les commandes dupliquées et les commandes commençant par un espace dans l'historique.
   - `shopt -s histappend` : Ajoute les nouvelles entrées d'historique au fichier d'historique.
   - `HISTSIZE=1000` : Définit le nombre d'entrées d'historique à conserver en mémoire.
   - `HISTFILESIZE=2000` : Définit la taille maximale du fichier d'historique.
   - `shopt -s checkwinsize` : Met à jour la taille de la fenêtre du terminal.

2. **Invite de commande colorée :**

   - Configure une invite de commande colorée si le terminal la prend en charge.

3. **Titre de la fenêtre :**

   - Définit le titre de la fenêtre du terminal pour afficher l'utilisateur actuel, l'hôte et le répertoire de travail.

4. **Couleurs des répertoires :**

   - Active la sortie colorée pour la commande `ls` si `dircolors` est disponible.

5. **Alias :**

   - `alias ll='ls -alF'` : Liste tous les fichiers avec des informations détaillées.
   - `alias la='ls -A'` : Liste tous les fichiers, y compris ceux cachés.
   - `alias l='ls -CF'` : Liste les fichiers en colonnes.
   - `alias alert='notify-send ...'` : Envoie une notification sur le bureau après la fin d'une commande.

6. **Fichier d'alias Bash :**

   - Inclut un fichier séparé pour les alias personnalisés (`~/.bash_aliases`).

7. **Complétion Bash :**

   - Active la complétion Bash si disponible.

8. **Configuration du chemin :**

- `export PATH=...` : Ajoute divers répertoires à la variable d'environnement `PATH`, y compris ceux pour CUDA, les gemmes Ruby, les binaires locaux et les binaires système.

9. Gestion du proxy :

- `export GLOBAL_PROXY='127.0.0.1:7890'` : Définit une variable pour l'adresse du serveur proxy.
- `function start_proxy { ... }` : Définit les variables d'environnement `HTTP_PROXY`, `HTTPS_PROXY`, `http_proxy`, `https_proxy` et `ALL_PROXY` pour utiliser le proxy spécifié.
- `function start_proxy_without_prefix { ... }` : Semblable à `start_proxy`, mais définit les variables proxy sans le préfixe `http://`.
- `function stop_proxy { ... }` : Désaffecte les variables proxy, désactivant ainsi le proxy.
- `export NO_PROXY="localhost,127.0.0.1,.example.com,::1"` : Spécifie les hôtes qui doivent contourner le proxy.

10. Proxy Git :

- `function start_git_proxy { ... }` : Configure Git pour utiliser le proxy global pour les connexions HTTP et
- `function stop_git_proxy { ... }` : Désaffecte les paramètres du proxy Git.

11. Proxy par défaut :

- `start_proxy` : Démarre le proxy par défaut.
- `start_git_proxy` : Démarre le proxy Git par défaut.

12. Alias Python :

- `alias python=python3` : Définit `python` pour utiliser `python3`.
- `alias pip=pip3` : Définit `pip` pour utiliser `pip3`.

13. Alias IA de message Git :

- `function gpa { ... }` : Crée un alias `gpa` pour exécuter un script Python `gitmessageai.py` avec l'API Mistral et autoriser le pull et le push.
- `function gca { ... }` : Crée un alias `gca` pour exécuter le même script sans pousser les modifications.
- `function gm { ... }` : Crée un alias `gm` pour exécuter le même script et n'afficher que le message de validation.

14. Push Git avec pull et rebase :

- `function gpp { ... }` : Tente de pousser les modifications, et si cela échoue, tente de tirer avec rebase puis de pousser à nouveau.

15. Vérification du proxy avant exécution :

- `preexec() { ... }` : Cette fonction est exécutée avant chaque commande. Elle vérifie si la commande figure dans une liste de commandes dépendantes du réseau. Si c'est le cas, et si des variables proxy sont définies, elle affiche les paramètres du proxy.
- `local network_commands=( ... )` : Ce tableau répertorie les commandes considérées comme dépendantes du réseau.
- `display_proxy() { ... }` : Cette fonction affiche les paramètres proxy actuels.

16. Fonction de vérification du proxy :

- `function checkproxy { ... }` : Affiche les paramètres proxy HTTP et HTTPS actuels, ainsi que les paramètres proxy Git.

```
case $- in
    *i*) ;;
      *) return;;
esac

HISTCONTROL=ignoreboth
shopt -s histappend
HISTSIZE=1000
HISTFILESIZE=2000
shopt -s checkwinsize

[ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"

if [ -z "${debian_chroot:-}" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi

case "$TERM" in
    xterm-color|*-256color) color_prompt=yes;;
esac

if [ -n "$force_color_prompt" ]; then
    if [ -x /usr/bin/tput ] && tput setaf 1 >&/dev/null; then
    color_prompt=yes
    else
    color_prompt=
    fi
fi
```

```
if [ "$color_prompt" = yes ]; then
    PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
else
    PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi
unset color_prompt force_color_prompt

case "$TERM" in
xterm*|rxvt*)
    PS1="\[\e]0;${debian_chroot:+($debian_chroot)}\u@\h: \w\a\]$PS1"
    ;;
*)
    ;;
esac

if [ -x /usr/bin/dircolors ]; then
    test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors -b)"
    alias ls='ls --color=auto'

    alias grep='grep --color=auto'
    alias fgrep='fgrep --color=auto'
    alias egrep='egrep --color=auto'
fi

alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

alias alert='notify-send --urgency=low -i "$([ $? = 0 ] && echo terminal || echo error)" "$(history|tail -n1|s

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
```

```
fi



export PATH="/usr/local/cuda-12.2/bin:/home/lzw/.local/share/gem/ruby/3.0.0/bin:/home/lzw/.local/bin:/usr/loca

export GLOBAL_PROXY='127.0.0.1:7890'

function start_proxy {
    export HTTP_PROXY="http://$GLOBAL_PROXY"
    export HTTPS_PROXY="http://$GLOBAL_PROXY"
    export http_proxy="http://$GLOBAL_PROXY"
    export https_proxy="http://$GLOBAL_PROXY"
    export HTTP_PROXY_REQUEST_FULLURI=false
    export HTTPS_PROXY_REQUEST_FULLURI=false
    export ALL_PROXY=$http_proxy
}

function start_proxy_without_prefix {
    export http_proxy=$GLOBAL_PROXY
        export HTTP_PROXY=$GLOBAL_PROXY
        export https_proxy=$GLOBAL_PROXY
    export HTTPS_PROXY=$GLOBAL_PROXY
    export HTTP_PROXY_REQUEST_FULLURI=false
    export HTTPS_PROXY_REQUEST_FULLURI=false
        export ALL_PROXY=$http_proxy
}

function stop_proxy {
    export http_proxy=
        export HTTP_PROXY=
        export https_proxy=
    export HTTPS_PROXY=
    export HTTP_PROXY_REQUEST_FULLURI=true
    export HTTPS_PROXY_REQUEST_FULLURI=true
        export ALL_PROXY=
}

export NO_PROXY="localhost,127.0.0.1,.example.com,::1"
```

```
function start_git_proxy {
  git config --global http.proxy $GLOBAL_PROXY
  git config --global https.proxy $GLOBAL_PROXY
}

function stop_git_proxy {
  git config --global --unset http.proxy
  git config --global --unset https.proxy
}

start_proxy
start_git_proxy

alias python=python3
alias pip=pip3

function gpa {
  python ~/bin/gitmessageai.py --api mistral --allow-pull-push
}

function gca {
  python ~/bin/gitmessageai.py --no-push
}

function gm {
  python ~/bin/gitmessageai.py --only-message
}

function gpp {
  git push || {
    echo "Push failed, attempting pull and merge"
    git pull --rebase || {
      echo "Pull failed, please resolve conflicts manually"
      return 1
    }
    git push || {
      echo "Push failed after pull, please resolve conflicts manually"
      return 1
    }
```

```
  }
}

preexec() {
    local network_commands=(
        "gpa"
        "git"
        "ssh"
        "scp"
        "sftp"
        "rsync"
        "curl"
        "wget"
        "apt"
        "yum"
        "dnf"
        "npm"
        "yarn"
        "pip"
        "pip3"
        "gem"
        "cargo"
        "docker"
        "kubectl"
        "ping"
        "traceroute"
        "netstat"
        "ss"
        "ip"
        "ifconfig"
        "dig"
        "nslookup"
        "nmap"
        "telnet"
        "ftp"
        "nc"
        "tcpdump"
        "adb"
        "bundle"
        "brew"
```

```
        "cpanm"
        "bundle exec jekyll"
        "make"
        "python"
        "glcoud"
    )

    local cmd
    cmd=$(echo "$1" | awk '{print $1}')

    display_proxy() {
        echo -e "  **Proxy Settings Detected:**"

        [ -n "$HTTP_PROXY" ] && echo "    - HTTP_PROXY: $HTTP_PROXY"
        [ -n "$HTTPS_PROXY" ] && echo "    - HTTPS_PROXY: $HTTPS_PROXY"

        echo ""
    }

    for network_cmd in "${network_commands[@]}"; do
        if [[ "$1" == "$network_cmd"* ]]; then
            if [ -n "$HTTP_PROXY" ] || [ -n "$http_proxy" ] || \
               [ -n "$HTTPS_PROXY" ] || [ -n "$https_proxy" ] || \
               [ -n "$ALL_PROXY" ] || [ -n "$all_proxy" ]; then

                display_proxy
            fi
            break
        fi
    done
}

function checkproxy {
  echo "HTTP_PROXY: $HTTP_PROXY"
  echo "HTTPS_PROXY: $HTTPS_PROXY"
  echo "Git HTTP Proxy:"
  git config --get http.proxy
  echo "Git HTTPS Proxy:"
  git config --get https.proxy
}
```