लिनक्स □□□□□□ कॉन्फ़िगरेशन

यह bashrc फ़ाइल □□□□□ में □□□□ शेल परिवेश को कॉन्फ़िगर करती है। यह प्रॉम्प्ट को कस्टमाइज़ करती है, एलियासेज़ सेट करती है, प्रॉक्सी सेटिंग्स प्रबंधित करती है, और □□□ जैसे टूल्स के साथ एकीकृत होती है। नीचे प्रमुख कॉन्फ़िगरेशन का विवरण दिया गया है:

1. मूल सेटिंग्स:

- HISTCONTROL=ignoreboth: इतिहास में डुप्लीकेट कमांड और स्पेस से शुरू होने वाले कमांड को अनदेखा करता है।
- shopt -s histappend: इतिहास फ़ाइल में नई इतिहास प्रविष्टियों को जोड़ता है।
- HISTSIZE=1000: मेमोरी में रखी जाने वाली इतिहास प्रविष्टियों की संख्या सेट करता है।
- HISTFILESIZE=2000: इतिहास फ़ाइल के अधिकतम आकार को सेट करता है।
- shopt -s checkwinsize: टर्मिनल विंडो के आकार को अपडेट करता है।

2. रंगीन प्रॉम्प्ट:

- यदि टर्मिनल इसका समर्थन करता है तो रंगीन कमांड प्रॉम्प्ट कॉन्फ़िगर करता है।

3. विंडो शीर्षक:

- वर्तमान उपयोगकर्ता, होस्ट और वर्किंग डायरेक्टरी को प्रदर्शित करने के लिए टर्मिनल विंडो का शीर्षक सेट करता है।

4. निर्देशिका रंग:

- यदि dircolors उपलब्ध है तो ls कमांड के लिए रंगीन आउटपुट सक्षम करता है।

5. एलियासेज़:

- alias ll='ls -alF': विस्तृत जानकारी के साथ सभी फ़ाइलों को सूचीबद्ध करता है।
- alias la='ls -A': छिपी हुई फ़ाइलों सहित सभी फ़ाइलों को सूचीबद्ध करता है।
- alias l='ls -CF': कॉलम में फ़ाइलों को सूचीबद्ध करता है।
- alias alert='notify-send ...': किसी कमांड के समाप्त होने के बाद डेस्कटॉप अधिसूचना भेजता है।

6. बैश एलियासेज़ फ़ाइल:

- कस्टम एलियासेज़ के लिए एक अलग फ़ाइल (~/.bash_aliases) शामिल करता है।

7. बैश पूर्णता:

- यदि उपलब्ध हो तो बैश पूर्णता सक्षम करता है।

8. पथ कॉन्फ़िगरेशन:

- export PATH=...: PATH पर्यावरण चर में विभिन्न निर्देशिकाएँ जोड़ता है, जिसमें □□□□, रूबी रत्न, स्थानीय बाइनरी और सिस्टम बाइनरी शामिल हैं।

9. प्रॉक्सी प्रबंधन:

- `export GLOBAL_PROXY='127.0.0.1:7890'`: प्रॉक्सी सर्वर पते के लिए एक चर परिभाषित करता है।
- `function start_proxy { ... }`: निर्दिष्ट प्रॉक्सी का उपयोग करने के लिए `HTTP_PROXY`, `HTTPS_PROXY`, `http_proxy`, `https_proxy`, और `ALL_PROXY` पर्यावरण चर सेट करता है।
- `function start_proxy_without_prefix { ... }`: `start_proxy` के समान, लेकिन `http://` उपसर्ग के बिना प्रॉक्सी चर सेट करता है।
- `function stop_proxy { ... }`: प्रॉक्सी चर अनसेट करता है, जिससे प्रॉक्सी प्रभावी रूप से अक्षम हो जाता है।
- `export NO_PROXY="localhost,127.0.0.1,.example.com,::1"`: उन होस्ट को निर्दिष्ट करता है जिन्हें प्रॉक्सी को बायपास करना चाहिए।

10. □□□ प्रॉक्सी:

- `function start_git_proxy { ... }`: HTTP HTTPS Git
- `function stop_git_proxy { ... }`: Git

11. डिफ़ॉल्ट प्रॉक्सी:

- `start_proxy`:
- `start_git_proxy`: Git

12. पायथन एलियासेज़:

- `alias python=python3`: `python3` का उपयोग करने के लिए `python` सेट करता है।
- `alias pip=pip3`: `pip3` का उपयोग करने के लिए `pip` सेट करता है।

13. □□□ संदेश □□ एलियासेज़:

- `function gpa { ... }`: मिस्ट्रल □□□ के साथ `gitmessageai.py` पायथन स्क्रिप्ट चलाने और पुल और पुश करने की अनुमति देने के लिए एक एलियास `gpa` बनाता है।
- `function gca { ... }`: परिवर्तनों को धकेले बिना समान स्क्रिप्ट चलाने के लिए एक एलियास `gca` बनाता है।
- `function gm { ... }`: समान स्क्रिप्ट चलाने और केवल कमिट संदेश प्रिंट करने के लिए एक एलियास `gm` बनाता है।

14. पुल और रीबेस के साथ □□□ पुश:

- `function gpp { ... }`: परिवर्तनों को पुश करने का प्रयास करता है, और यदि यह विफल हो जाता है, तो यह रीबेस के साथ पुल करने का प्रयास करता है और फिर से पुश करता है।

15. पूर्व-निष्पादन प्रॉक्सी जांच:

- `preexec() { ... }`: यह फ़ंक्शन प्रत्येक कमांड से पहले निष्पादित होता है। यह जांचता है कि क्या कमांड नेटवर्क-निर्भर कमांड की सूची में है। यदि यह है, और यदि कोई प्रॉक्सी चर सेट हैं, तो यह प्रॉक्सी सेटिंग्स प्रदर्शित करता है।

- `local network_commands=( ... )`: यह सरणी उन कमांड को सूचीबद्ध करता है जिन्हें नेटवर्क-निर्भर माना जाता है।
- `display_proxy() { ... }`: यह फ़ंक्शन वर्तमान प्रॉक्सी सेटिंग्स प्रदर्शित करता है।

16. प्रॉक्सी फ़ंक्शन जांचें:

- `function checkproxy { ... }`: वर्तमान □□□□ और □□□□□ प्रॉक्सी सेटिंग्स, साथ ही □□□ प्रॉक्सी सेटिंग्स प्रदर्शित करता है।

```
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
}
```