Linux Bashrc Configuration

This `bashrc` file configures the Bash shell environment in Linux. It customizes the prompt, sets up aliases, manages proxy settings, and integrates with tools like Git. Below is a breakdown of the key configurations:

1. **Basic Settings:**

   - `HISTCONTROL=ignoreboth`: Ignores duplicate commands and commands starting with a space in the history.
   - `shopt -s histappend`: Appends new history entries to the history file.
   - `HISTSIZE=1000`: Sets the number of history entries to keep in memory.
   - `HISTFILESIZE=2000`: Sets the maximum size of the history file.
   - `shopt -s checkwinsize`: Updates the terminal window size.

2. **Color Prompt:**

   - Configures a colored command prompt if the terminal supports it.

3. **Window Title:**

   - Sets the terminal window title to display the current user, host, and working directory.

4. **Directory Colors:**

   - Enables colored output for `ls` command if `dircolors` is available.

5. **Aliases:**

   - `alias ll='ls -alF'`: Lists all files with detailed information.
   - `alias la='ls -A'`: Lists all files, including hidden ones.
   - `alias l='ls -CF'`: Lists files in columns.
   - `alias alert='notify-send ...'`: Sends a desktop notification after a command finishes.

6. **Bash Aliases File:**

   - Includes a separate file for custom aliases (`~/.bash_aliases`).

7. **Bash Completion:**

   - Enables bash completion if available.

8. **Path Configuration:**

- `export PATH=...`: Adds various directories to the `PATH` environment variable, including those for CUDA, Ruby gems, local binaries, and system binaries.

9. Proxy Management:

- `export GLOBAL_PROXY='127.0.0.1:7890'`: Defines a variable for the proxy server address.
- `function start_proxy { ... }`: Sets the `HTTP_PROXY`, `HTTPS_PROXY`, `http_proxy`, `https_proxy`, and `ALL_PROXY` environment variables to use the specified proxy.
- `function start_proxy_without_prefix { ... }`: Similar to `start_proxy`, but sets the proxy variables without the `http://` prefix.
- `function stop_proxy { ... }`: Unsets the proxy variables, effectively disabling the proxy.
- `export NO_PROXY="localhost,127.0.0.1,.example.com,::1"`: Specifies hosts that should bypass the proxy.

10. Git Proxy:

```
- `function start_git_proxy { ... }`: Configures git to use the global proxy for HTTP and HTTPS connections.
- `function stop_git_proxy { ... }`: Unsets the git proxy settings.
```

11. Default Proxy:

```
- `start_proxy`: Starts proxy by default.
- `start_git_proxy`: Starts git proxy by default.
```

12. Python Aliases:

- `alias python=python3`: Sets `python` to use `python3`.
- `alias pip=pip3`: Sets `pip` to use `pip3`.

13. Git Message AI Aliases:

- `function gpa { ... }`: Creates an alias `gpa` to run a python script `gitmessageai.py` with the mistral API and allow pull and push.
- `function gca { ... }`: Creates an alias `gca` to run the same script without pushing changes.
- `function gm { ... }`: Creates an alias `gm` to run the same script and only print the commit message.

14. Git Push with Pull and Rebase:

- `function gpp { ... }`: Attempts to push changes, and if it fails, it tries to pull with rebase and then push again.

15. Pre-Execution Proxy Check:

- `preexec() { ... }`: This function is executed before every command. It checks if the command is in a list of network-dependent commands. If it is, and if any proxy variables are set, it displays the proxy settings.
- `local network_commands=( ... )`: This array lists commands that are considered network-dependent.
- `display_proxy() { ... }`: This function displays the current proxy settings.

16. Check Proxy Function:

- `function checkproxy { ... }`: Displays the current HTTP and HTTPS proxy settings, as well as Git proxy settings.

```
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
}
```