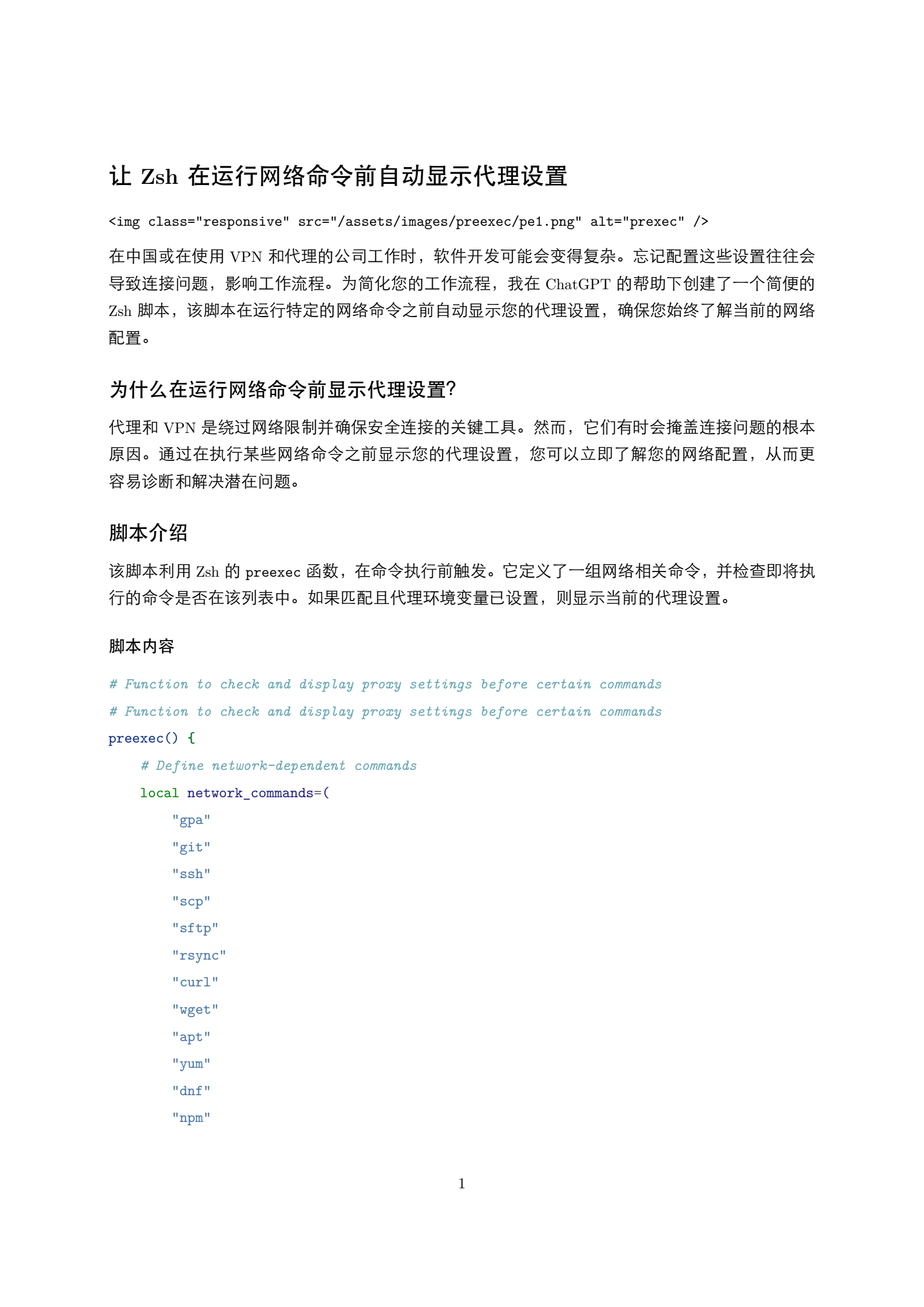
让 Zsh 在运行网络命令前自动显示代理设置



在中国或在使用 VPN 和代理的公司工作时，软件开发可能会变得复杂。忘记配置这些设置往往会导致连接问题，影响工作流程。为简化您的工作流程，我在 ChatGPT 的帮助下创建了一个简便的 Zsh 脚本，该脚本在运行特定的网络命令之前自动显示您的代理设置，确保您始终了解当前的网络配置。

为什么在运行网络命令前显示代理设置?

代理和 VPN 是绕过网络限制并确保安全连接的关键工具。然而，它们有时会掩盖连接问题的根本原因。通过在执行某些网络命令之前显示您的代理设置，您可以立即了解您的网络配置，从而更容易诊断和解决潜在问题。

脚本介绍

该脚本利用 Zsh 的 `preexec` 函数，在命令执行前触发。它定义了一组网络相关命令，并检查即将执行的命令是否在该列表中。如果匹配且代理环境变量已设置，则显示当前的代理设置。

脚本内容

```
# Function to check and display proxy settings before certain commands
# Function to check and display proxy settings before certain commands
preexec() {
    # Define network-dependent commands
    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"
    # Add more commands as needed
)

# Extract the first word (command) from the command line
local cmd
cmd=$(echo "$1" | awk '{print $1}')

# Function to display proxy variables
display_proxy() {
    echo -e "\n  **Proxy Settings Detected:**"
```

```
        [ -n "$HTTP_PROXY" ] && echo "    - HTTP_PROXY: $HTTP_PROXY"
        [ -n "$http_proxy" ] && echo "    - http_proxy: $http_proxy"
        [ -n "$HTTPS_PROXY" ] && echo "    - HTTPS_PROXY: $HTTPS_PROXY"
        [ -n "$https_proxy" ] && echo "    - https_proxy: $https_proxy"
        [ -n "$ALL_PROXY" ] && echo "    - ALL_PROXY: $ALL_PROXY"
        [ -n "$all_proxy" ] && echo "    - all_proxy: $all_proxy"

        echo ""
    }

    # Check if the command is network-dependent
    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
}
```

在 Zsh 中设置脚本

1. 打开您的 .zshrc 文件

使用您喜欢的文本编辑器打开 .zshrc 配置文件。例如：

```
nano ~/.zshrc
```

2. 添加 preexec 函数

将上述脚本粘贴到 .zshrc 文件的末尾，确保它位于文件的最后部分，以避免与其他配置冲突。

3. 保存并关闭文件

如果您使用的是 nano，请按 CTRL + O 保存更改，然后按 CTRL + X 退出编辑器。

4. 应用更改

重新加载您的 .zshrc 文件以立即应用新配置：

```
source ~/.zshrc
```

测试设置

1. 启用代理后测试

临时设置一个代理变量并运行一个网络相关命令：

```
export HTTP_PROXY="http://127.0.0.1:7890"
pip install selenium beautifulsoup4 urllib3
```

预期输出：

```
 **检测到代理设置:**
   - HTTP_PROXY: http://127.0.0.1:7890
   - http_proxy: 127.0.0.1:7890
   - HTTPS_PROXY: 127.0.0.1:7890
   - https_proxy: 127.0.0.1:7890
   - ALL_PROXY: 127.0.0.1:7890
   - all_proxy: 127.0.0.1:7890

Collecting selenium
  Downloading selenium-4.x.x-py2.py3-none-any.whl (xxx kB)
...
```

2. 禁用代理后测试

取消代理变量并运行相同的命令：

```
unset HTTP_PROXY
pip install selenium beautifulsoup4 urllib3
```

预期输出：

```
Collecting selenium
  Downloading selenium-4.x.x-py2.py3-none-any.whl (xxx kB)
...
```

(不会出现代理通知。)

3. 非网络相关命令测试

运行一个本地命令，如 `ls`：

```
ls
```

预期输出：

[文件和目录列表]

(不会出现代理通知。)

自定义脚本

1. 扩展 network_commands 列表

根据您的工作流程，您可能需要添加更多网络相关命令到 network_commands 数组中。

2. 处理别名

确保所有网络相关命令的别名都包含在 network_commands 列表中。您可以通过以下命令查看现有别名：

```
alias
```

如果某个别名未包含在列表中，请将其添加，以确保在使用该别名时触发代理通知。

3. 使用颜色增强可见性

为了在杂乱的终端中更好地显示代理通知，您可以为通知添加颜色：

```
# 在 .zshrc 文件顶部添加颜色代码
GREEN='\033[0;32m'
NC='\033[0m' # 无颜色

display_proxy() {
    echo -e "\n${GREEN} ** 检测到代理设置:**${NC}"

    [ -n "$HTTP_PROXY" ] && echo "   - HTTP_PROXY: $HTTP_PROXY"
    [ -n "$http_proxy" ] && echo "   - http_proxy: $http_proxy"
    [ -n "$HTTPS_PROXY" ] && echo "   - HTTPS_PROXY: $HTTPS_PROXY"
    [ -n "$https_proxy" ] && echo "   - https_proxy: $https_proxy"
    [ -n "$ALL_PROXY" ] && echo "   - ALL_PROXY: $ALL_PROXY"
    [ -n "$all_proxy" ] && echo "   - all_proxy: $all_proxy"
```

```
    echo ""
}
```

结论

在受限网络环境中管理代理设置对于确保软件开发的顺利进行至关重要。通过在 Zsh 配置中实现这个条件性的 `preexec` 函数，您可以确保在执行特定的网络命令之前，您的终端会自动显示代理设置。这种方法不仅保持了终端的整洁，还能让您在关键操作时及时了解网络配置，提升工作效率和问题排查能力。

祝编码愉快!