

Crochets Zsh

Après avoir exploré “Faire en sorte que Zsh affiche les paramètres de proxy avant d’exécuter des commandes réseau”, j’ai approfondi les hooks de Zsh avec ChatGPT. Voici un aperçu concis pour référence future.

Dans Zsh, les hooks vous permettent d’exécuter des fonctions personnalisées à des moments spécifiques lors des opérations du shell. Au-delà de `preexec`, Zsh propose plusieurs hooks pour améliorer votre environnement :

1. `precmd`

- Quand : Avant que l’invite ne soit affichée.
- Utilisation : Mettre à jour l’invite ou effectuer un nettoyage.
- Exemple :

```
precmd() {  
    echo "Prêt pour la prochaine commande !"  
}
```

2. `chpwd`

- Quand : Lorsque le répertoire courant change.
- Utilisation : Mettre à jour les variables d’environnement ou déclencher des actions en fonction du répertoire.
- Exemple :

```
chpwd() {  
    echo "Changement vers : $PWD"  
}
```

3. preexec_functions & precmd_functions

- Quand : Similaire à preexec et precmd, mais prend en charge plusieurs fonctions.
- Utilisation : Ajouter plusieurs actions sans écraser les hooks existants.
- Exemple :

```
precmd_functions+=(additional_precmd)

additional_precmd() {
    echo "Tâche precmd supplémentaire."
}
```

4. TRAPDEBUG

- Quand : Après chaque commande, avant l'affichage des résultats.
- Utilisation : Débogage, journalisation des commandes.
- Exemple :

```
TRAPDEBUG() {
    echo "Exécuté : $1"
}
```

5. TRAPEXIT

- Quand : Lorsque le shell se termine.
- Utilisation : Tâches de nettoyage ou affichage de messages de sortie.
- Exemple :

```
TRAPEXIT() {
    echo "Au revoir !"
}
```

6. Hooks zle

- Quand : Pendant l'édition de ligne.
- Utilisation : Personnaliser le comportement de la ligne de commande.
- Exemple :

```
zle-line-init() {  
    echo "Édition d'une nouvelle commande."  
}  
zle -N zle-line-init
```

7. Hooks d'Histoire (zshaddhistory, zshremovehistory)

- Quand : Lors de l'ajout ou de la suppression d'entrées dans l'historique.
- Utilisation : Filtrer ou gérer l'historique.
- Exemple :

```
zshaddhistory() {  
    [[ $1 == *"secret"* ]] && return 1  
    return 0  
}
```

8. periodic

- Quand : À des intervalles définis par period.
- Utilisation : Vérifications ou mises à jour régulières.
- Exemple :

```
periodic() {  
    echo "Exécution de la tâche périodique..."  
}
```

9. add-zsh-hook

- Objectif : Ajouter des fonctions de manière sécurisée à des hooks.
- Utilisation : Ajouter plusieurs fonctions sans écraser les précédentes.
- Exemple :

```
add-zsh-hook precmd another_precmd
```

```
another_precmd() {  
    echo "Une autre fonction precmd."  
}
```

Résumé

Le système de hooks de Zsh est polyvalent, permettant l'automatisation et la personnalisation :

- `preexec` : Avant l'exécution de la commande.
- `precmd` : Avant l'affichage de l'invite de commande.
- `chpwd` : Lors du changement de répertoire.
- `TRAPDEBUG` : Post-commande pour le débogage.
- `TRAPEXIT` : À la sortie du shell.
- Hooks `zle` : Pendant l'édition de ligne.
- Hooks d'historique : Gestion de l'historique des commandes.
- `periodic` : À des intervalles définis.
- `add-zsh-hook` : Ajout de plusieurs fonctions de hook.

L'utilisation de ces hooks peut grandement améliorer votre expérience avec Zsh, rendant votre shell plus efficace et adapté à votre flux de travail.