Unix / Linux - Réductions d'entrée / sortie de shell

Dans ce chapitre, nous discuterons en détail des redirections d'entrée / sortie de Shell. La plupart des commandes système Unix prennent des entrées de votre terminal et renvoient la sortie résultante à votre terminal. Une commande lit normalement son entrée à partir de l'entrée standard, qui se trouve être votre terminal par défaut. De même, une commande écrit normalement sa sortie en sortie standard, qui est à nouveau votre terminal par défaut.

Réorientation de la sortie

La sortie d'une commande normalement destinée à une sortie standard peut être facilement détournée vers un fichier à la place. Cette capacité est connue sous le nom de redirection de sortie.

Si le fichier de notation > est ajouté à une commande qui écrit normalement sa sortie en sortie standard, la sortie de cette commande sera écrite pour déposer à la place de votre terminal.

Vérifier ce qui suit **qui** commande qui redirige la sortie complète de la commande dans le fichier utilisateur.

```
$ who > users
```

Notez qu'aucune sortie n'apparaît au terminal. En effet, la sortie a été redirigée du périphérique de sortie standard par défaut (le terminal) dans le fichier spécifié. Vous pouvez vérifier le fichier utilisateur pour le contenu complet −

```
$ cat users
oko         tty01   Sep 12 07:30
ai          tty15   Sep 12 13:32
ruth        tty21   Sep 12 10:10
pat         tty24   Sep 12 13:07
steve       tty25   Sep 12 13:03
$
```

Si une commande a sa sortie redirigée vers un fichier et que le fichier contient déjà certaines données, ces données seront perdues. Prenons l'exemple suivant −

```
$ echo line 1 > users
$ cat users
line 1
```

```
$
```

Vous pouvez utiliser l'opérateur > > pour ajouter la sortie dans un fichier existant comme suit −

```
$ echo line 2 >> users
$ cat users
line 1
line 2
$
```

Rédirection d'entrée

Tout comme la sortie d'une commande peut être redirigée vers un fichier, l'entrée d'une commande peut être redirigée à partir d'un fichier. Comme le **caractère supérieur à >** est utilisé pour la redirection de sortie, le **caractère inférieur à TAG1>** est utilisé pour rediriger l'entrée d'une commande.

Les commandes qui prennent normalement leur entrée à partir de l'entrée standard peuvent voir leur entrée redirigée à partir d'un fichier de cette manière. Par exemple, pour compter le nombre de lignes dans le fichier *utilisateurs* généré ci-dessus, vous pouvez exécuter la commande comme suit −

```
$ wc -l users
2 users
$
```

Lors de l'exécution, vous recevrez la sortie suivante. Vous pouvez compter le nombre de lignes dans le fichier en redirigeant l'entrée standard du **wc** commande à partir du fichier *utilisateurs* −

```
$ wc -l < users
2
$
```

Notez qu'il y a une différence dans la sortie produite par les deux formes de la commande wc. Dans le premier cas, le nom des utilisateurs de fichiers est répertorié avec le nombre de lignes; dans le second cas, ce n'est pas le cas.

Dans le premier cas, wc sait qu'il lit ses entrées des utilisateurs de fichiers. Dans le deuxième cas, il sait seulement qu'il lit ses entrées à partir de l'entrée standard afin de ne pas afficher le nom du fichier.

Ici le document

A **ici document** est utilisé pour rediriger l'entrée dans un script ou programme shell interactif.

Nous pouvons exécuter un programme interactif dans un script shell sans action de l'utilisateur en fournissant l'entrée requise pour le programme interactif ou le script shell interactif.

Le formulaire général pour un **ici** le document est −

```
command << delimiter
document
delimiter
```

Ici, le shell interprète le **< <** opérateur comme instruction de lire l'entrée jusqu'à ce qu'il trouve une ligne contenant le délimiteur spécifié. Toutes les lignes d'entrée jusqu'à la ligne contenant le délimiteur sont ensuite introduites dans l'entrée standard de la commande.

Le délimiteur dit à la coquille que le **ici** le document est terminé. Sans cela, le shell continue de lire l'entrée pour toujours. Le délimiteur doit être un seul mot qui ne contient ni espaces ni onglets.

Voici l'entrée de la commande **wc -l** pour compter le nombre total de lignes −

```
$wc -l << EOF
   This is a simple lookup program
        for good (and bad) restaurants
        in Cape Town.
EOF
3
$
```

Vous pouvez utiliser le **ici document** pour imprimer plusieurs lignes en utilisant votre script comme suit −

Démo en direct

```
#!/bin/sh

cat << EOF
This is a simple lookup program
for good (and bad) restaurants
in Cape Town.
EOF
```

Lors de l'exécution, vous recevrez le résultat suivant −

```
This is a simple lookup program
for good (and bad) restaurants
in Cape Town.
```

Le script suivant exécute une session avec le **vi** éditeur de texte et enregistre l'entrée dans le fichier **test.txt**.

```
#!/bin/sh

filename=test.txt
vi $filename <<EndOfCommands
i
This file was created automatically from
a shell script
^[
ZZ
EndOfCommands
```

Si vous exécutez ce script avec vim agissant comme vi, vous verrez probablement une sortie comme suit −

```
$ sh test.sh
Vim: Warning: Input is not from a terminal
$
```

Après avoir exécuté le script, vous devriez voir le texte suivant ajouté au fichier **test.txt** −

```
$ cat test.txt
This file was created automatically from
a shell script
$
```

Jeter la sortie

Parfois, vous devrez exécuter une commande, mais vous ne voulez pas que la sortie soit affichée à l'écran. Dans de tels cas, vous pouvez supprimer la sortie en la redirigeant vers le fichier **/dev / null** −

```
$ command > /dev/null
```

Voici la commande est le nom de la commande que vous souhaitez exécuter. Le fichier **/dev / null** est un fichier spécial qui rejette automatiquement toutes ses entrées.

Pour supprimer à la fois la sortie d'une commande et sa sortie d'erreur, utilisez la redirection standard pour rediriger **STDERR** à **DÉBUT** −

```
$ command > /dev/null 2>&1
```

Ici **2** représente **STDERR** et **1** représente **DÉBUT**. Vous pouvez afficher un message sur STDERR en redirigeant STDOUT dans STDERR comme suit −

```
$ echo message 1>&2
```

Commandes de réorientation

Voici une liste complète des commandes que vous pouvez utiliser pour la redirection −

Sr.No.	Commande et description
1	**fichier pgm >** La sortie de pgm est redirigée vers le fichier
2	**fichier pgm <** Programme pgm lit ses entrées à partir du fichier
3	**fichier pgm > >** La sortie de pgm est ajoutée au fichier
4	**fichier n >** Sortie du flux avec descripteur **n** redirigé vers le fichier
5	**fichier n > >** Sortie du flux avec descripteur **n** ajouté au fichier
6	**n > & m** Fusionne la sortie du flux **n** avec flux **m**
7	**n < & m** Fusionne l'entrée du flux **n** avec flux **m**
8	**< < tag** L'entrée standard vient d'ici à la prochaine balise au début de la ligne
9	**|** Prend la sortie d'un programme ou d'un processus et l'envoie à un autre

Notez que le descripteur de fichier **0** est normalement une entrée standard (STDIN), **1** est une sortie standard (STDOUT), et **2** est une sortie d'erreur standard (STDERR).