

# SED

Sed est un éditeur ligne non interactif. Il reçoit du texte en entrée, que ce soit à partir de stdin ou d'un fichier, réalise certaines opérations sur les lignes spécifiées de l'entrée, une ligne à la fois, puis sort le résultat vers stdout ou vers un fichier. A l'intérieur d'un script shell, sed est habituellement un des différents outils composant un tube.

Sed détermine sur quelles lignes de son entrée il va opérer à partir de *l'ensemble d'adresses* qui lui est passé. Cette plage d'adresses est définie soit par des numéros de ligne soit par un motif à rechercher. Par exemple, `3d` indique à sed qu'il doit supprimer la ligne 3 de l'entrée, et `/windows/d` dit à sed que vous voulez que toutes les lignes de l'entrée contenant << windows >> soient supprimées.

De toutes les opérations de la boîte à outils sed, nous nous occuperons principalement des trois les plus communément utilisées. Il s'agit de **printing** (NdT: affichage vers stdout), **deletion** (NdT: suppression) et **substitution** (NdT: euh... substitution :).

**Tableau C-1. Opérateurs sed basiques**

Opérateur	Nom	Effet
<code>[plage-d-adresses]/p</code>	print	Affiche [la plage d'adresse spécifiée]
<code>[plage-d-adresses]/d</code>	delete	Supprime [la plage d'adresse spécifiée]
<code>s/motif1/motif2/</code>	substitute	Substitue motif2 à la première instance de motif1 sur une ligne
<code>[plage-d-adresses]/s/motif1/motif2/</code>	substitute	Substitue motif2 par la première instance de motif1 sur une ligne, en restant dans la <i>plage-d-adresses</i>
<code>[plage-d-adresses]/y/motif1/motif2/</code>	transform	remplace tout caractère de motif1 avec le caractère correspondant dans motif2, en restant dans la <i>plage-d-adresses</i> (équivalent de <b>tr</b> )
<code>g</code>	global	Opère sur <i>chaque</i> correspondance du motif à l'intérieur de chaque ligne d'entrée de la plage d'adresse concernée

ASTUCE

La substitution opère seulement sur la première instance de la correspondance d'un motif à l'intérieur de chaque ligne, sauf si l'opérateur *g* (*global*) est ajouté à la commande *substitute*.

A partir de la ligne de commande et dans un script shell, une opération sed peut nécessiter de mettre entre guillemets et d'utiliser certaines options.

```
sed -e '/^$/d' $nomfichier
# L'option -e fait que la chaîne de caractère suivante est interprétée
comme
#+ une instruction d'édition.
# (Si vous passez une seule instruction à "sed", le "-e" est optionnel.)
# Les guillemets "forts" (') empêchent les caractères de l'ER compris dans
#+ l'instruction d'être interprétés comme des caractères spéciaux par le
corps
#+ du script.
# (Ceci réserve l'expansion de l'ER de l'instruction à sed.)
```

```
#
# Opère sur le texte contenu dans le fichier $nomfichier.
```

Dans certain cas, une commande d'édition **sed** ne fonctionnera pas avec des guillemets simples.

```
nomfichier=fichier1.txt
modele=BEGIN

sed "/^$motif/d" "$nomfichier" # fonctionne comme indiqué
# sed '/^$motif/d' "$nomfichier" a des résultats inattendus.
# Dans cette instance, avec des guillemets forts (' ... '),
#+ "$modele" ne sera pas étendu en "BEGIN".
```

**ASTUCE** Sed utilise l'option **-e** pour spécifier que la chaîne suivante est une instruction ou un ensemble d'instructions. Si la chaîne ne contient qu'une seule instruction, alors cette option peut être omise.

```
sed -n '/xzy/p' $nomfichier
# L'option -n indique à sed d'afficher seulement les lignes correspondant
au
#+ motif.
# Sinon toutes les lignes en entrée s'afficheront.
# L'option -e est inutile ici car il y a une seule instruction d'édition.
```

## Tableau C-2. Exemples d'opérateurs sed

Notation	Effet
8d	Supprime la 8 <sup>e</sup> ligne de l'entrée.
/^\$/d	Supprime toutes les lignes vides.
1,/^\$/d	Supprime les lignes du début à la première ligne vide (inclue).
/Jones/p	Affiche seulement les lignes contenant << Jones >> (avec l'option -n).
s/Windows/Linux/	Substitue << Linux >> à chaque première instance de << Windows >> trouvée dans chaque ligne d'entrée.
s/BSOD/stability/g	Substitue << stability >> à chaque instance de << BSOD >> trouvée dans chaque ligne d'entrée.
s/ *\$//	Supprime tous les espaces à la fin de toutes les lignes.
s/00*/0/g	Comprime toutes les séquences consécutives de zéros en un seul zéro.
/GUI/d	Supprime toutes les lignes contenant << GUI >>.
s/GUI//g	Supprime toutes les instances de << GUI >>, en laissant le reste de la ligne intact.

Substituer une chaîne vide (taille zéro) à une autre est équivalent à supprimer cette chaîne dans une ligne de l'entrée. Le reste de la ligne reste intact. Appliquer `s/GUI//` à la ligne

```
The most important parts of any application are its GUI and sound effects
```

donne

```
The most important parts of any application are its and sound effects
```

L'anti-slash force la continuité sur la ligne suivante pour la commande sed. Ceci a pour effet d'utiliser la *nouvelle ligne* comme fin de ligne de la *chaîne de remplacement*.

```
s/^ */\  
/g
```

Cette substitution remplace les espaces débutant les lignes par un retour chariot. Le résultat final est le remplacement des indentations de paragraphes par une ligne vide entre les paragraphes.

Une plage d'adresse suivie par une ou plusieurs opérations peut nécessiter des accolades ouvrantes et fermantes, avec les retours chariot appropriés.

```
/[0-9A-Za-z] /, /^$/ {  
/^$/d  
}
```

Ceci supprime seulement la première de chaque ensemble de lignes vides consécutives. Ceci peut être utile pour espacer de manière égale un fichier texte, mais en conservant les lignes vides entre paragraphes.

**ASTUCE** Une façon rapide de doubler les espaces dans un fichier texte est `sed G nomfichier`.