sed

Éditeur de flux, utilisé pour effectuer des transformations de texte basiques

OPTIONS

-n, –quiet, –silent Par défaut, sed affiche le motif à la fin de chaque cycle. Cette option désactive cet affichage.

-e SCRIPT, –expression=SCRIPT Ajoute les commandes dans le script au jeu de commandes à lancer durant le traitement de l'entrée

-f SCRIPTFILE, –file=SCRIPTFILE AJoute les commandes contenues dans le script au jeu de commande à lancer durant le traitement de l'entrée

-i [SUFFIX], –in-place [=SUFFIX] Spécifie que les fichiers sont à éditer sur place. Sed créé un fichier temporaire et envoie la sortie dans ce fichier au lieu de stdout. Cette option implique '-s'. Quand la fin du fichier est atteinte, il est renommé au fichier d'origine. L'extension, si fournie, est utilisé pour renommer l'ancien fichier.

-l N, –line-length=N Spécifie la longueur de ligne par défaut pour la commande 'l'. 0 signifie de ne jamais couper les lignes. Défaut : 70.

–posix Se conforme à POSIX. identique à définir POSIXLY_CORRECT.

-b, –binary Pour les OS faisant la distinction entre les fichiers binaire et fichiers textes, comme MS-DOS, Windows.

–follow-symlinks Seulement avec -i. Si le fichier spécifié est un lien symbolique, sed suit le lien et édite la cible.

-r, –regexp-extended Utilise les expressions régulières étendues au lieu des expressions régulières de base.

-s, –separate Par défaut, sed considère les fichiers spécifiés comme un seul long flux. Cette option permet à l'utilisateur de les considérer comme flux séparés. Les numéros de ligne sont relatifs au début de chaque fichier, '$' réfère à la dernière ligne de chaque fichier, et les fichiers invoqués depuis des commandes 'R' sont remis au début de chaque fichier

-u, –unbuffered Met en tampon l'entrée et la sortie aussi minimal que pratique. Particulièrement utile si l'entrée via depuis des commandes comme tail -f, et que vous souhaitez voir la sortie transformée le plus vite possible.

-z, –null-data, –zero-terminated Traite l'entrée comme jeu de ligne, chacune terminée par un ASCII NUL au lieu d'une nouvelle ligne.

Fonctionnement

sed maintient 2 tampons : l'espace _pattern_ actif, et l'espace auxiliaire _hold_. Ils sont vide à l'initialisation. sed opères en effectuant le cycle suivant sur chaque ligne d'entrée : d'abord, sed lit une ligne depuis le flux d'entrée, supprime tout newline de fin, et la place dans l'espace pattern. Les commandes sont ensuite exécutées ; chaque commande peut avoir une adresse associée : les adresses osnt un type de code de condition, et une commande est seulement exécutée si la condition est vérifiée avant d'exécuter la commande.

Quand la fin du script est atteint, sauf si l'option -n est utilisé, le contenu de l'espace pattern est affiché dans le flux de sortie, en ajoutant un newline à la fin s'il avait été supprimé. Puis le cycle démarre pour la ligne suivante.

Sauf pour les commandes spéciales (comme 'D'), l'espace pattern est supprimé entre 2 cycles. L'espace hold, conserve ses données entre les cycles (voir les commandes h, H, x, g, G pour se déplacer entre les tampons)

Expressions régulières

Les adresses dans un script sed peut être une des formes suivantes :

NUMBER Spécifie un numéro de ligne qui va matcher seulement cette ligne dans l'entrée.

FIRST~STEP Match chaque ligne par pas en commençant avec le ligne FIRST. Par exemple, pour sélectionner les lignes impaires : 1~2.

$ Matche la dernière ligne du dernier fichier d'entrée, ou la dernière ligne de chaque fichier quand -i ou -s sons spécifiés

/REGEXP/ Sélectionne les lignes qui matchent l'expression régulière. L'expresion vide (//) répète la dernière expression régulière qui a matché.

\%REGEXP% Matche l'expression régulière, mais permet d'utiliser un autre délimiteur que '/'

/REGEXP/I, \%REGEXP%I l'expression régulière matche en étant insensible à la casse.

/REGEXP/M, \%REGEXP%M matche l'expression régulière en mode multi-ligne. '^matche également la chaîne vide aprèse un newline, '$' la chaîne vide avant un newline. '\`' et '\'' matchent toujours le début ou la fin du tampon. ',' ne matche pas un caractère newline en mode multi-ligne

0,/REGEXP/ Un numéro de ligne 0 peut être utilisé dans une spécification d'adresse pour que sed tente de matcher l'expression régulière dans la première ligne d'entrée également. En d'autres termes, 0,/REGEXP/ est similaire à 1,/REGEXP/, excepté que si ADDR2 matche la première ligne de l'entrée, 0,/REGEXP/ la considère à la fin de la plage.

ADDR1,+N Matche ADDR1 et les N lignes suivantes

ADDR1,~N Matche ADDR1 et les lignes suivantes jusqu'à la prochaine ligne dont le numéro de ligne soit un multiple de N

! Ajouter ! à la fin d'une spécification d'adresse inverse le sense du match.

Syntaxe des expressions régulières

CHAR Un simple caractère ordinaire correspondant à lui-même

***** Correspond à une séquence de 0 ou plusieurs instances de correspondances pour l'expression réguière précédante, qui doit être un caractère ordinaire, un caractère spéciale, un regexp groupé, une une expression entre crochet.

\+ comme * mais matche un ou plusieurs

\? comme * mais matche exactement I séquences (I est un entier décimal entre 0 et 255)

\{I,J\} Matche entre I et J, inclusifs, séquences

\{I,\} Matche plus que ou égal à I séquences

\(REGEXP\) Groupe le REGEXP interne dans son ensemble

. Matche n'importe quel caractère, incluant newline

^ Matche la chaîne null au début de l'espace pattern

$ Idem à ^mais réfère à la fin de l'espace pattern

[LIST], [^LIST] Matche tout caractère simple dans LIST. Par exemple, [aeiou] matche toutes les voyelles. ^inverse le sense de la liste

REGEXP1\|REGEXP2 Matche soit REGEXP1 ou REGEXP2. Utilise les parenthèses pour utilise des expressions régulières alternative complexes.

REGEXP1REGEXP2 Matche la concaténation de REGEXP1 et REGEXP2.

\DIGIT Matche la n-ième sous-expression \(...\) dans l'expression régulière

\n Matche le caractère newline

\CHAR Matche CHAR où CHAR est '$', '*', '.', '[', '\', ou '^'.

Exemples

Matche abcdef
abcdef

matche 0 ou plusieurs a, suivi par un caractère b :
a*b
Matche b ou ab
a\?b
matche un ou plusieurs a, suivi par un ou plusiuers b
a\+b\+
Matchent tous les caractères dans une chaîne, incluant une chaîne vide
.*
Matchent tous les caractères dans une chaîne, d'au moins un caractère
.\+
Matche un chaîne commençant avec 'main' suivi par une parenthèse ouvrante et fermante. n, (et) ne doivent pas être adjacents
^main.*(.*)
Matche une chaîne commençant avec #
^#
matche une chaîne se terminant avec un \
\\$
Matche une chaîne consisant d'un simple signe dollar
\$
Matche les lettres ASCII ou les chiffres
[a-zA-Z0-9]
(ici <TAB> est un simple caractère). Matche une chaîne d'un ou plusieurs caractères, aucun d'entre eux n'est un espace ou une tabulation. Généralement, celà signifie un mot
[^<TAB>]\+
Matche une chaîne consistant de 2 sous-chaînes égales, séparées par un newline
^\(.*\)\n\1$
Matche 9 caractères suivis par un A
.\{9\}A$
Matche le début d'une chaîne qui contient 16 caractères, le dernier est un A
^.\{15\}A

commande usuelles

Aucune adresse permise. Commence un commentaire, qui contient jusqu'au prochain newline.

q [EXIT-CODE] Quitte sed dans traiter plus de commandes ou entrée, et retourne le code donné

d Supprime l'espace pattern ; démarrant immédiatement un nouveau cycle

p Affiche l'espace pattern sur stdout. généralement utilisé avec l'option -n

n Si auto-print est actif, affiche l'espace pattern, puis remplace l'espace pattern avec la ligne suivante.

{ COMMANDS } Un groupe de commandes peut être enfermé dans des accolades.

commande s

La syntaxe de la commande s (s pour substitute) est 's/REGEXP/REPLACEMENT/FLAGS' Les caractères '/' peuvent être uniformément remplacés par un autre caractère. La commande s est probablement la plus importante de sed. sed tente de correspondre l'espace pattern avec REGEXP, et si le match réussit, remplace la portion de l'espace pattern qui matche avec REPLACEMENT.

REPLACEMENT peut contenir \N (N étant un nombre de 1 à 9) références aux portion de matche entre \(et \) dans REGEXP. REPLACEMENT peut contenir '&' qui réfère au match comple. Les séquences suivantes peuvent être inculs :

\L Active le remplacement en minuscule jusqu'à un \U ou \E

\l Met le prochain caractère en minuscule

\U Active le remplacement en majuscule jusqu'à un \L ou \E

\u Met le prochain caractère en majuscule

\E Stop la conversion démarrée par \L ou \U

Flags

g Applique le remplacement de tous les matches au REGEXP, pas simplement le premier

NUMBER Applique le remplacement seulement au NUMBER-ième matche

p Si la substitution est effective, affiche le nouvel espace pattern

w FILE-NAME Si la substitution est effective, écris le résultat dans le fichier nommé.

e Permet à un pipe d'entrée depuis une commande shell dans l'espace pattern. Si une substitution est effective, la commande trouvée dans l'espace pattern est exécutée et l'espace pattern est remplacée avec sa sortie.

I, i I match de manière insensible à la casse.

M, m M matche les expressions régulière en mode multi-ligne

Commandes moins usuelles

y/SOURCE-CHARS/DEST-CHARS/ Traduit tous caractères dans l'espace pattern qui matche un des SOURCE-CHARS avec le caractères correspondant DEST-CHARS

a\, TEXT Accèpte 2 adresses. Met en file les lignes de texte qui suivent cette commande (chacune sauf celles se terminant avec un '\', qui sont supprimés de la sortie) sur la sortie à la fin du cycle courant, ou quand la ligne suivante est lue.

i\, TEXT Affiche immédiatement les lignes de texte qui suivent cette commande (chacune sauf celles se terminant avec un \, qui sont supprimées de la sortie).

c\, TEXT Supprime les lignes matchant l'adresse ou la plage d'adresse, et sort les lignes de texte qui suivent cette commande à la place de la dernière ligne (ou à la place de chaque ligne, si aucune adresse n'est spécifiée). Un nouveau cycle est démarré après cette commande vu que l'espace pattern sera supprimé.

= Affiche le numéro de ligne d'entrée avec un newline

l N Affiche l'espace pattern dans une forme non ambigüe : les caractères non-imprimables sont affiché dans une forme échappé style C. N définis la longueur de ligne. 0 ne coupe pas les lignes.

r FILENAME Accèpte 2 adresses. Met en file le contenu de FILENAME à lire et à insérer dans le flux de sortie à la fin du cycle courant, ou quand la prochaine ligne d'entrée est lue. Si FILENAME ne peut pas être lu, il est traité comme si c'était un fichier vide, sans erreur.

w FILENAME écrit l'espace pattern dans FILENAME.

D Si l'espace pattern ne contient pas de newline, démarre un nouveau cycle normal comme si la commande d avait été donnée. Sinon, supprime le texte dans l'espace pattern jusqu'au newline, et redémarre le cycle avec l'espace pattern résultant, sans lire un newline d'entrée

N Ajoute un newline à l'espace pattern, puis ajoute la ligne suivante de l'entrée dans l'espace pattern.

P Affiche la portio de l'espace pattern jusqu'au newline

h Remplace le contenu de l'espace hold avec le contenu de l'espace pattern

H Ajoute un newline au contenu de l'espace hold, puis y ajoute le contenu de l'espace pattern.

g Remplace le contenu de l'espace pattern avec le contenu de l'espace hold

G Ajoute un newline au contenu de l'espace pattern, puis y ajoute le contenu de l'espace hold.

x Échange le contenu des espace pattern et hold.

Commande avancées

Dans la plupart des cas, il est préférrable d'utiliser des commandes comme awk ou perl au lieu de ces commandes

: LABEL Aucune adresse permise. Spécifie l'emplacement d'un LABEL pour les commandes de branchement.

b LABEL Branche inconditionneles à LABEL.

t LABEL Branche à LABEL seulement s'il a été substitué depuis que la dernière ligne d'entrée a été lue ou qu'un branchement conditionnel a été fait.

Commandes spécifiques à GNU sed

e [COMMAND] Permet de pipe une entrée depuis une commande shell dans l'espace pattern. Sans paramètres, cette commande exécute la commande trouvée dans l'espace pattern et remplace l'espace pattern avec la sortie ; et supprime tout newline de fin. Avec un paramètre, cette commande interprète comme commande et envoie sa sortie sur le flux de sortie.

F Affiche le nom du fichier du fichier d'entrée

L N Remplis et joint les lignes dans l'espace pattern pour produire des lignes en sortie de (au plus) N caractères, comme fmt. Si N est omis, le défaut tel que spécifié sur la ligne de commande est utilisé.

Q [EXIT-CODE] n'accepte qu'une seule adresse. fonctionne comme 'q'

R FILENAME Met en file une ligne de FILENAME et l'insert dans le flux de sortie à la fin du cycle courant, ou quand la prochaine ligne est lue. Si FILENAME ne peut pas être lu, ou si la fin est atteind, aucune ligne n'est ajoutée.

T LABEL Brachement sur LABEL uniquement si la substitution n'a pas réussit.

v VERSION Ne fait rien, mais échoue si sed n'a pas les extensions GNU.

W FILENAME Écrit dans le fichier la portion de l'espace pattern jusqu'à un newline.

z Vide le contenu de l'espace pattern. Généralement identique 's/.*//', mais est plus efficate.

Extensions GNU pour les échappements dans les expressions régulières

\a alert

\f form feed

\n nouvelle ligne

\r retour charriot

\t tabulation horizontale

\v Tabulation verticale

\cX Matche CONTROL-X, ou X est un caractère.

\dXXX Caractère ASCII où XXX est la valeur décimale

\oXXX Caractère ASCII où XXX est la valeur octale

\xXXX Caractère ASCII où XXX est la valeur héxadécimale

\b caractère \

\w Matche tout caractère 'word' (lettre, chiffre ou _)

\W Matche tout caractère non word

\b Matche une fin de mot, qui matche si le caractère à gauche est un caractère word, et le caractère à droite est un caractère non-word

\B Matche tout sauf une fin de mot.

\‘ Matche seulement au début de l'espace pattern. Différent de ^en mode multi-ligne

\’ Matche seulement la fin de l'espace pattern. Différent de $ en mode multi-ligne