

RECompare

Comparer une chaîne de caractères avec un modèle pour regarder s'ils correspondent. C'est la plus simple des commandes qui détermine si un texte correspond à une expression régulière (en cas de succès, elle retourne 1, sinon 0). Son usage le plus fréquent est dans la reconnaissance de noms de fichier (RECompare leNom pattern ".*\..sit") ou dans le contrôle des entrées de texte d'utilisateur (RECompare userInput pattern "\\d").

Syntaxe:

RECompare *uneListe* pattern *unTexte*

Paramètres:

uneListe

Le texte soumis à l'opération.

Classe: liste ("list") de chaînes de caractères ("string")

unTexte

Le modèle d'expression régulière.

Classe: chaîne de caractères ("string")

Résultat: Classe: valeur booléenne ("boolean") -- *est-ce que le texte correspond au modèle ?*

REMatch

Retourne une liste de sous-chaînes d'une chaîne de caractères donnant les occurrences trouvées dans un texte qui correspondent au modèle. Cette commande considère que votre attente concerne la ligne entière du texte et non seulement le texte trouvé, aussi retourne-t-elle la ligne entière. Si vous utilisez REMatch, plutôt que "REMatchLines", les seules parties du texte trouvées sont retournées, le reste étant laissé de côté. C'est utilisé lorsque vous voulez extraire une petite partie d'un texte long. Si vous voulez appliquer plusieurs options de formatage du résultat, vous pouvez utiliser le paramètre "using".

Syntaxe:

REMatch *uneListe* pattern *unTexte*
[ignoreCase *uneValeurBooleenne*]
[separator *unTexte2*]
[using *unTexte3*]

Paramètres:

uneListe

Le texte soumis à l'opération.

Classe: liste ("list") de chaînes de caractères ("string")

unTexte

Le modèle d'expression régulière.

Classe: chaîne de caractères ("string")

uneValeurBooleenne

Est-ce que la recherche de correspondance doit ignorer la casse ?

Classe: valeur booléenne ("boolean")

unTexte2

Le caractère séparateur de ligne.

Classe: chaîne de caractères ("string")

Valeur par défaut: retour chariot/CR

unTexte3

pattern to use in generating the extracted values.

Classe: chaîne de caractères ("string")

Résultat: Classe: liste ("list") de chaînes de caractères ("string") -- *la liste des fragments de texte reconnus.*

Si vous désirez le second nombre de chaque ligne:

```
property LeTexte:"12 Albert  
13 Gertrude"
```

```
REMatch someText pattern "^\d+[^0-9]+(\\d+)" with "\\1"
```

REMatchLines

Retourne une liste de lignes d'une chaîne de caractères donnant les occurrences trouvées correspondant au modèle.

Syntaxe:

REMatchLines *uneListe* pattern *unTexte*

[ignoreCase *uneValeurBooleenne*]

[separator *unTexte2*]

[inverse *uneValeurBooleenne2*]

Paramètres:

uneListe

Le texte soumis à l'opération.

Classe: liste ("list") de chaînes de caractères ("string")

unTexte

Le modèle d'expression régulière.

Classe: chaîne de caractères ("string")

uneValeurBooleenne

Est-ce que la recherche de correspondance doit ignorer la casse ?

Classe: valeur booléenne ("boolean")

unTexte2

Le caractère séparateur de ligne.

Classe: chaîne de caractères ("string")

Valeur par défaut: retour chariot/CR

uneValeurBooleenne2

Faut-il retourner les lignes ou la comparaison est mise en échec ?

Classe: valeur booléenne ("boolean")

Résultat: Classe: liste ("list") de chaînes de caractères ("string") -- *la*

REReplace

Remplace tout fragment de texte correspondant au modèle avec le texte de remplacement.

Syntaxe:

```
REReplace uneListe with unTexte pattern unTexte2  
[ ignoreCase uneValeurBooleenne ]  
[ separator unTexte3 ]
```

Paramètres:

uneListe

Le texte soumis à l'opération.

Classe: liste ("list") de chaînes de caractères ("string")

unTexte

Le texte de remplacement.

Classe: chaîne de caractères ("string")

unTexte2

Le modèle d'expression régulière.

Classe: chaîne de caractères ("string")

uneValeurBooleenne

Est-ce que la recherche de correspondance doit ignorer la casse ?

Classe: valeur booléenne ("boolean")

unTexte3

Le caractère séparateur de ligne.

Classe: chaîne de caractères ("string")

Valeur par défaut: retour chariot/CR

Résultat: Classe: chaîne de caractères ("string") -- *la nouvelle version du texte.*

Notez que dans l'exemple qui suit `\r` représente le retour chariot.

```
property LeTexte : "12 Albert  
13 Gertrude"
```

```
REReplace LeTexte pattern "Albert" with "\rLili\r"
```

```
résultat: "12  
Lili
```

```
13 Gertrude"
```

Remplacer toute occurrence d'un ou plusieurs espaces par un tabulation.

Notez que dans l'exemple qui suit `\t` représente la tabulation:

```
property LeTexte : "12      Albert  "
```

REReplace LeTexte pattern " +" with "\\t"

résultat: "12 Albert "

Numérotation de lignes

Numéroter des lignes (avec un espace après le nombre) correspondants à une réussite de la recherche d'une expression rationnelle:

property LeTexte : "A Albert
B Gertrude
C Gérard"

REReplace LeTexte pattern "^" with "# "

résultat: "1 A Albert
2 B Gertrude
3 C Gérard"

De la même manière, vous pouvez insérer le numéro de recherche positive (c'est-à-dire le nombre de fois que le modèle a été trouvé) dans votre chaîne de remplacement. Utiliser le caractère "%" pour désigner ce nombre.

Enfin vous pouvez inclure le caractère d'occurrence exact du texte trouvé dans le texte original en utilisant le caractère "@" .

Références arrières

Encore plus utile est la possibilité le texte que vous venez de trouvé comme partie d'un remplacement. Le caractère "&" est remplacé, lorsqu'il est mis dans votre texte de remplacement, par le texte entier que votre expression régulière vient de reconnaître. Par exemple:

```
REMatch LeTexte pattern "(\\w|.)+@(\\w|.)+" with "mailto:&"
```

vous donnera une liste (dont le délimiteur est le retour chariot) de tous les adresses de courriel d'un document, chacun étant précédé du texte "mailto:". Le modèle fait une reconnaissance d'un ou plusieurs caractères alphanumériques ou de point suivi par un "@", suivi lui-même par un ou plusieurs caractères pouvant être alphanumériques ou un point.

Vous pouvez aussi utiliser cette possibilité pour placer du texte au début de chaque ligne. Par exemple, le script qui suit placera le signe > et un espace avant toute ligne qui n'est pas vide.

property LeTexte : "A Albert
B Gertrude
C Gérard"

REReplace LeTexte pattern "^\.*\S" with "> &"

résultat: "> A Albert
> B Gertrude

> C Gérard"

Vous pouvez même utiliser une partie du texte reconnu au sein de votre chaîne de remplacement. Pour ce faire, mettez la partie de votre modèle de recherche entre parenthèses. Le texte qui a été reconnu par le groupe se trouvant entre parenthèses peut ensuite être utilisé au sein de la chaîne de remplacement. Pour se référer à un groupe particulier, utilisez une séquence "\\ " suivie par le numéro du groupe. Le premier groupe de votre modèle de reconnaissance est numéroté "1", le second "2", et ainsi de suite jusqu'à "9". En d'autres termes, ceci fonctionne comme une référence arrière au sein de votre modèle.

Si vous avez plusieurs parenthèses qui se suivent, la parenthèse ouvrante déterminera le numéro de la référence arrière.

Par exemple, si vous avez un document avec deux colonnes séparées par une tabulation, et que vous voulez intervertir les colonnes, vous pouvez utiliser la commande suivante:

```
REReplace LeTexte pattern "(.*)\\t(.*)" with "\\2\\t\\1"
```

Le texte précédent la tabulation sera reconnu par le premier "(.*)" et peut être référencé dans la chaîne de remplacement comme "\\1". De la même façon, le texte se trouvant après la tabulation peut être référencé comme étant "\\2".

Accessoirement, vous pouvez référencer le texte entier que votre modèle a reconnu avec "\\10". Ceci fonctionne exactement comme le caractère &. Aucun de ces caractères, cependant, ne peut être utilisé au sein de votre modèle.

Modifier la casse

Si vous incluez "\\u" dans votre chaîne de remplacement, le reste du texte de remplacement sera converti en majuscule, "\\l" convertit le reste du texte en minuscule. Ainsi pour cet exemple:

```
property LeTexte : "A Albert"
```

```
REReplace LeTexte pattern ".*" with "\\l&"
```

résultat: "a albert"

Si vous voulez convertir une seule partie du texte de remplacement, utilisez la séquence "\\-" . "\\-" supprime l'effet de tout "\\u" ou "\\l" antérieur. Par exemple, le script suivant convertit la première lettre de tous les mots en majuscule:

```
property LeTexte : "a albert  
gertrude"
```

```
REReplace LeTexte pattern "(\\b.)(\\w*)" with "\\u\\1\\-\\2"
```

résultat: "A albert
Gertrude"

comme cet exemple ne semble pas encore fonctionner, voici l'exemple actuellement fonctionnel:

property LeTexte : "a albert
gertrude"

REReplace LeTexte pattern "\\b." with "\\u&"

résultat: "A Albert
Gertrude"

notez que l'usage de "\\u" et "\\|" dans le texte de remplacement est différent de leur usage dans le modèle de recherche, où ils reconnaissent chacun une simple lettre majuscule ou minuscule.

Les séparateurs

Généralement, un caractère retour chariot est le séparateur idéal, aussi n'avez-vous aucune raison de le changer. Il y a, cependant, des situations qui obligent de le modifier.

Si vous avez besoin de reconnaître les caractères retour chariot, vous devez modifier le séparateur en quelque chose d'autre. Ainsi:

property LeTexte : "a albert
gertrude"

REReplace LeTexte pattern "\\r" with "\\n" separator "]"

résultat: "a albert gertrude"

Ce script remplace tout retour chariot en passage à la ligne, qui est le séparateur usuel de ligne sous UNIX. Le crochet est ici une lettre arbitrairement choisie comme séparateur.

Une autre méthode est de nommer le séparateur avec une lettre qui ne se trouve nulle part dans le texte de recherche. Ainsi, tout le texte sera traité comme une seule ligne. C'est utile lorsque vous avez besoin de chercher et remplacer en utilisant un modèle qui peut couvrir plusieurs lignes. Une note d'avertissement, néanmoins: cette méthode nécessite beaucoup de mémoire si vous travaillez avec un texte long. De même, elle peut ralentir une recherche avec un modèle très complexe.

N'oubliez pas non plus que les caractères spéciaux ^ et \$ reconnaissent le début et fin de ligne. En modifiant le séparateur, ce qui sera considéré comme une ligne sera en conséquence modifié et les résultats des reconnaissances par ^ et \$ changés eux aussi.

Pour utiliser une lettre #,&, @, ou \ au sein de votre chaîne de remplacement, précédez-les de "\\".