Utiliser xargs [How To Section Shell, ligne de commande]

1. xargs

La commande **xargs** sous UNIX est un utilitaire de ligne de commande permettant de créer un pipeline d'exécution à partir d'une entrée standard. Bien que des outils tels que grep puissent accepter l'entrée standard en tant que paramètre, de nombreux autres outils ne le peuvent pas. L'utilisation de **xargs** permet à des outils tels que **echo**, **rm** et **mkdir** d'accepter les entrées standard en tant qu'arguments.

2. Utilisation de base

Exemple : création de 3 répertoires avec xargs

```
echo 'one two three' | xargs mkdir
ls
one two three
```

Vous me direz que ça va plus vite en tapant simplement

```
mdkir one two three
```

Pas faux, sur ce coup.... Mais vous vous doutez bien qu'on va un peu pousser les choses...

3. Associer xargs et find : des exemples concrets

3.1 En gros

On lance **find** (sur le répertoire courant (.) ou dans un répertoire spécifique (/Nom/Du/Dir) avec une option de recherche (**-name** ,basée sur le nom ; **-time**, basé sur la date, etc...) et tout ce qu'il trouve, on lui applique ce qui est lancé après le pipeline c'est à dire après xargs. Tout simple donc. A savoir si on joue avec des fichiers ou des répertoires avec des espaces, on passera l'option **-print0 à find** et **-0 à xargs**.

Bon maintenant que vous savez, je pense qu'il vaut mieux mettre quelques exemples concrets

3.2 Exemples concrets

Trouver les fichiers temporaires vieux de plus de 14 jours et les supprimer

```
find /tmp -mtime +14 | xargs rm
```

Avancer toutes les dates du répertoire “date” de 2 jours

```
find date/  -print0 | xargs -0 -I '{}' touch -r '{}' -d '+2 day' '{}'
```

Ouvrir tous les fichiers trouvés avec gedit

```
find . -name "foo*" -print0 | xargs -0 gedit
```

Trouver tous les fichiers contenant l'occurence abc

```
find -name "*.txt" | xargs grep "abc"
```

Trouver tous les fichiers png d'une archive tar.gz

```
find Images/Dossier/ -name "*.png" -type f -print0 | xargs -0 tar -cvzf
Arcgive.tar.gz
```

Connaitre le nombre de lignes, mots, caractères des fichiers d'une liste

```
ls *.txt | xargs wc
```

Effacer des fichiers du répertoire courant (ici les fichiers avec extension .c)

```
find . -name "*.c" | xargs rm -rf
```

Le même si les fichiers contiennent des espaces

```
find . -name "*.c" -print0 | xargs -0 rm -rf
```

Le même sur un dossier spécifique

```
find /home/MonUser/MonDossier/ -name "*.c" -print0 | xargs -0  rm -rf
```

Générer une liste compacte des users d'un système

```
cut -d: -f1 < /etc/passwd | sort | xargs echo
```

4. L'option -I de xargs

Il est possible d'exécuter plusieurs commandes avec **xargs** en utilisant l'indicateur **-I**. Ceci remplace les occurrences de l'argument par l'argument transmis à **xargs**. Les impressions suivantes font écho à une chaîne et créent un dossier.

```
cat test.txt
tata
toto
```

```
tutu
```

```
cat test.txt | xargs -I % sh -c 'echo %; mkdir %'
tata
toto
tutu
```

```
ls
tata toto tutu
```

5. L'option -t de xargs

Cette options (**-t**) permet l'affichage des commandes qui sont effectuées.

```
echo 'one two three' | xargs -t rm
rm one two three
```

6. exec VS xargs

On retrouve les mêmes fonctionnalités “**find -exec” que “find | xargs**” avec des subtilités bien sûr. Mais les tests effectués sur de grosses quantités de fichiers montrent que l'association “**find | xargs**” est beaucoup plus rapide.

Exemple sur plus de 1 500 images

```
time find . -name "*.jpg" -exec ls {} ;
real    0m6.618s
user    0m1.465s
sys     0m4.396s
```

```
time find . -name "*.jpg" -print0 | xargs -0 ls
real    0m1.120s
user    0m0.594s
sys     0m0.527s
```

Donc on va dire que pour une utilisation simple, xargs va plus vite ;) (attention dans certains cas, il vaut mieux utiliser exec)

7. Conclusions

Bon voilà une commande très puissante Plus de détail

```
man xargs
man find
```

```
man exec
```