Vim - les macros

- Objet : commande macro avec vim
- Niveau requis :
  débutant, avisé
- Commentaires : *Créer et enregistrer une commande d'édition vim réutilisable à volonté.*
- Débutant, à savoir : Utiliser GNU/Linux en ligne de commande, tout commence là !
- La commande d'édition VIM - Détail
- Suivi :
  à-tester
  - Création par martin_mtl le 11/12/2012
  - Testé par .... le ....
- Commentaires sur le forum : C'est ici[1)]

Introduction

Avec La commande d'édition VIM, nous pouvons créer une macro[2)] d'une commande vim complète que nous souhaitons répéter ultérieurement.

Exemple

Dans un fichier d'une page web, nous nous trouvons à devoir réactiver des liens préparés mais commentés (et donc inactifs) par // ainsi :

```
// ligne web ancrée
// ligne web ancrée
// ligne web ancrée
```

```
// ligne web ancrée
// ligne web ancrée
// ligne web ancrée
```

```
// ligne web ancrée
// ligne web ancrée
// ligne web ancrée
```

Nous désirons supprimer : ''// “ devant chacune des troisième ligne pour obtenir ceci :

```
// ligne web ancrée
// ligne web ancrée
ligne web ancrée
```

```
// ligne web ancrée
// ligne web ancrée
ligne web ancrée
```

```
// ligne web ancrée
// ligne web ancrée
ligne web ancrée
```

Nous ouvrons ce fichier avec **vim**, puis, pour déclencher l'enregistrement de la macro, nous tapons :

```
qa
```

- q = enregistrement de la macro
- a = nom de la macro

Un avis : **Enregistrement** survient et tout ce que nous tapons à présent est l'enregistrement de cette macro nommée “a”.

Nous pouvons voir en direct sur le texte ce que notre commande vim exécute sur la ligne visée.
Terminer la macro par Q

En cas d'erreur, il suffit de quitter la macro par Q

puis, d'utiliser la lettre U

pour restaurer la ligne et recommencer la macro différemment, en la nommant de manière identique.

Pour relancer cette macro nommée “a”, tapons @+A

et voilà.

TP d'une macro vim

Concrètement, pour en revenir à notre exemple, en mode commande (tapez echap) après s'être placé sur la troisième ligne du premier groupe de ligne, tapons :

```
qa:s/\/\/ //
```

(validation)

```
4jq
```

(validation)

```
2@a
```

(validation)

Dans le détail

```
qa:s/\/\/ //
```

Notez les caractères *antislach* d'échappement afin que le caractère / soit lu comme un caractère

normal et non comme un signe spécial. Voir : regexp.

Validez et aussitôt, la ligne concernée devient :

```
// ligne web ancrée
// ligne web ancrée
ligne web ancrée
```

pour se placer 4 lignes plus loin, continuons la macro en tapant :

```
4jq
```

ce qui correspond à :

- 4j = demander d'aller 4 lignes plus bas
- q = arrêter l'enregistrement de la macro.

Et maintenant ?

Et maintenant, nous pouvons réenclencher cette macro nommée “a” sur les lignes suivantes.
Nous sommes déjà positionnés sur la prochaine ligne à modifier, tapons :

```
2@a
```

- 2 = exécuter 2 fois. À noter que le nombre 2 de la répétition s'adapte aux nombres de lignes que nous voulons modifier. Il peut être supérieur à l'exemple présenté ici.
- @a = la macro nommée “a”

Et sur *validation*, les lignes visées seront modifiées. 🙂

De :

```
// ligne web ancrée
// ligne web ancrée
ligne web ancrée
```

```
// ligne web ancrée
// ligne web ancrée
// ligne web ancrée
```

```
// ligne web ancrée
// ligne web ancrée
// ligne web ancrée
```

Nous obtenons bien :

```
// ligne web ancrée
// ligne web ancrée
ligne web ancrée
```

```
// ligne web ancrée
```

```
// ligne web ancrée
ligne web ancrée
```

```
// ligne web ancrée
// ligne web ancrée
ligne web ancrée
```

Super !

Commentaire et remerciements

Bien sûr, au début, cela demande un peu de reflexion, mais je pense qu'il serait dommage de ne pas utiliser cet outil *macro de vim* plus opportunément encore !

Merci à **adrien** *sur le chan* #slackware-fr *et à toute la communauté pour sa joyeuse entraide (malgrès les charriages à propos de nos distros libres respectives ! Scrongnongnon...)*

1)

N'hésitez pas à y faire part de vos remarques, succès, améliorations ou échecs !

2)

macro : enregistrement d'une ligne de commandes enchaînées les unes après les autres dans leur continuité