mcedit

- Objet : mcedit, Utilisation
- Niveau requis :
  débutant, avisé
- Commentaires : l'éditeur de Midnight Commander
- Débutant, à savoir : Utiliser GNU/Linux en ligne de commande, tout commence là !
- Suivi :
  à-tester
  - Création par martin_mtl le 11/12/2012
  - Testé par .... le ....
- Commentaires sur le forum : C'est ici[1)]

Présentation

mcedit est un lien vers mc, Midnight Commander, le forçant à lancer immédiatement son éditeur interne. L'éditeur est une version pour terminaux de l'éditeur X Window autonome cooledit.

Utilisation

Tapez simplement dans un terminal **mcedit** et le chemin de votre fichier :

```
mcedit /mon/fichier
```

Pour une utilisation plus pointu, vous pouvez utilisez diverses options.

Synopsys

```
mcedit [ [+numéro] fichier [-bcCdfhstVx?] ]
```

Options

Tableau des options

Option	Action
+numéro	Aller au numéro de ligne spécifié (ne pas insérer d'espace entre le signe « + » et le numéro).
-b	Force un affichage en noir et blanc.
-c	Force le mode couleur sur les terminaux où mcedit utilise le noir et blanc par défaut.

Option	Action
-C <mot-clé>=<FGcouleur>,<BGcouleur>:<mot-clé>=...	Utilisé pour spécifier un jeu de couleurs différent, où mot-clé peut être normal, selected, marked, markselect, errors, reverse menu, menusel, menuhot, menuhotsel ou gauge. Les couleurs sont optionnelles et peuvent être black, gray, red, brightred, green, brightgreen, brown, yellow, blue, brightblue, magenta, brightmagenta, cyan, brightcyan, lightgray ou white (Voyez la section Couleurs dans mc pour plus d'informations).
-d	Désactive la prise en charge de la souris.
-f	Affiche les chemins de recherche des fichiers de Midnight Commander intégrés à la compilation.
-t	Utilisé uniquement si le code a été compilé avec Slang et terminfo : utiliser la valeur de la variable TERMCAP pour obtenir les informations sur le terminal,au lieu des informations provenant de la base de données des terminaux globale au système.
-V	Affiche le numéro de version du programme.
-x	Forcer le mode xterm. Utilisé quand on utilise des terminaux supportant les fonctionnalités de xterm (deux modes d'écran, et la faculté d'envoyer des séquences d'échappement de souris).

Fonctionnalités

L'éditeur interne de fichiers fournit la plupart des fonctionnalités des éditeurs plein écran habituels. Il a une limite de taille de fichiers extensible de 16 Mo, et édite sans difficulté les fichiers binaires.

Les fonctionnalités qu'il supporte actuellement sont :

- copie
- déplacement,
- effacement et couper/coller de blocs
- annulation touche pour touche (key for key undo)
- menus déroulants
- insertion de fichier
- définition de macros
- recherche et remplacement d'expressions rationnelles (et notre propre recherche et remplacement scanf-printf)
- mise en évidence de texte MSW-MAC avec shift-flèches (uniquement pour la console Linux)
- bascule insertion/écrasement

- césure de mots
- toutes sortes d'options de disposition en colonnes
- mise en évidence de la syntaxe pour divers types de fichiers.
- Enfin, il y a également une option permettant d'envoyer des blocs de texte via un tube à des commandes shell comme indent et ispell.

Touches

L'éditeur est très facile à utiliser et ne requiert aucune assistance.
Pour voir ce que font les touches, consultez simplement le menu déroulant approprié.

Les autres touches sont :

TOUCHES	***ACTION***
Maj + touches fléchées	pour effectuer de la mise en évidence de texte (console Linux uniquement)
Ctrl+Inser	pour copier dans le fichier ~/.mc/cedit/cooledit.clip
Maj+Inser	pour coller à partir de ~/.mc/cedit/cooledit.clip.
Maj+Suppr	coupe dans ~/.cedit/cooledit.clip
Ctrl+Suppr	efface le texte mis en évidence - console Linux uniquement.

La touche de complètement (voyez mc.1) effectue également un retour sec sans indentation automatique.

La mise en évidence avec la souris fonctionne également, et vous pouvez surcharger l'action de la souris de la façon habituelle en gardant la touche Maj appuyée pendant que vous traînez (glissez) la souris pour permettre le fonctionnement normal de la mise en évidence avec la souris dans un terminal.

Pour définir une macro, appuyez sur Ctrl+R et tapez ensuite les combinaisons de touches que vous voulez exécuter. Appuyez à nouveau sur Ctrl+R quand vous avez fini.
Vous pouvez ensuite affecter la macro à n'importe quelle touche en appuyant sur cette touche.
La macro est exécutée quand vous appuyez sur Ctrl+A et ensuite sur la touche affectée.
La macro est également exécutée si vous appuyez sur Meta, Ctrl ou Echap et la touche spécifiée, à condition que la touche ne soit pas utilisée pour une autre fonction.

Une fois définies, les commandes de la macro prennent place dans le fichier ~/.mc/cedit/cooledit.macros. N'éditez PAS ce fichier si vous avez l'intention de réutiliser des macros dans la même session d'édition, car mcedit place en mémoire cache les définitions des touches de macros.
mcedit écrase désormais une macro si une macro utilisant la même touche existe déjà, de sorte que vous n'avez pas besoin d'éditer ce fichier. Vous devrez également relancer tous les autres éditeurs en cours d'exécution pour que ces macros prennent effet.

F19 formatera du code C quand il est mis en évidence. Un fichier exécutable nommé ~/.mc/cedit/edit.indent.rc sera créé pour vous à partir du modèle par défaut. N'hésitez pas à l'éditer si nécessaire.

C-p exécutera ispell sur un bloc de texte d'une façon similaire. Le fichier script sera appelé ~/.mc/cedit/edit.spell.rc.

Redéfinir des touches

Les touches peuvent être redéfinies à partir du menu d'options de Midnight Commander.

Mise en évidence de la syntaxe

Depuis la version 3.6.0, cooledit dispose de la mise en évidence de la syntaxe. Cela signifie que les mots-clés et les contextes (comme les commentaires C, les constantes de type chaîne de caractères, etc.) sont mis en évidence dans des couleurs différentes. La section suivante explique le format du fichier ~/.mc/cedit/Syntax.

Le fichier ~/.mc/cedit/Syntax est réexaminé lors de l'ouverture de n'importe quel fichier par l'éditeur.
Le fichier contient des règles pour la mise en évidence, placées sur des lignes séparées, qui définissent quels mots-clés seront mis en évidence et dans quelle couleur.
Le fichier est également divisé en sections, chacune commençant par une ligne comprenant la commande file, suivie d'une expression rationnelle.
L'expression rationnelle précise le nom de fichier auquel s'applique ce jeu de règles.
Après cela, on trouve une description à afficher à gauche de la fenêtre d'éditeur expliquant le type de fichier à l'utilisateur.
Un troisième argument optionnel est une expression rationnelle à comparer avec la première ligne de texte du fichier.
Si le nom du fichier ou la première ligne de texte correspond, alors ces règles seront chargées.

La fin d'une section coïncide avec le début d'une nouvelle section.
Chaque section est divisée en contextes, et chaque contexte contient des règles.
Un contexte est une portée à l'intérieur du texte à laquelle appartient un ensemble de règles. Par exemple, la région située à l'intérieur d'un commentaire de type C (c.-à-d. entre /* et */) possède sa propre couleur. C'est un contexte, bien qu'il ne renfermera pas d'autres règles car il n'y a probablement rien que l'on veuille voir mis en évidence à l'intérieur d'un commentaire C.

Une section de programmation C triviale pourrait ressembler à ceci :

```
file .\*\\.c C\sProgram\sFile (#include|/\\\*)

wholechars abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_

# couleurs par défaut
context default
  keyword  whole  if       24
  keyword  whole  else     24
  keyword  whole  for      24
  keyword  whole  while    24
  keyword  whole  do       24
  keyword  whole  switch   24
  keyword  whole  case     24
  keyword  whole  static   24
  keyword  whole  extern   24
  keyword         {        14
```

```
  keyword         }        14
  keyword         '*'      6

# commentaires C
context /\* \*/ 22

# directives de préprocesseur C
context linestart # \n 18
  keyword  \\\n  24

# constantes chaîne de caractères C
context " " 6
  keyword  %d    24
  keyword  %s    24
  keyword  %c    24
  keyword  \\"   24
```

Chaque contexte débute par une ligne de la forme :

```
 context [exclusive] [whole|wholeright|wholeleft] [linestart] délim
[linestart] délim [avant-plan] [arrière-plan]
```

Le premier contexte fait exception. Il doit débuter par la commande

```
 context default [avant-plan] [arrière-plan]
```

ou sinon cooledit renverra une erreur.

L'option linestart dicte que le délimiteur délim doit démarrer au début d'une ligne.

L'option whole spécifie que délim (le délimiteur) doit être un mot complet. Un mot complet est un groupe de caractères qui peut être modifié n'importe où dans le fichier avec la commande wholechars. La commande wholechars au-dessus définit simplement l'ensemble exactement à sa valeur par défaut et aurait par conséquent pu être omise. Pour spécifier qu'un mot ne doit être complet que sur la gauche, vous pouvez utiliser l'option wholeleft, et d'une façon similaire sur la droite. Les groupes de caractères de gauche et de droite peuvent être réglés séparément avec

```
 wholechars [left|right] caractères
```

L'option exclusive provoque la mise en évidence du texte situé entre les délimiteurs, mais pas les délimiteurs eux-mêmes.

Chaque règle est une ligne de la forme :

```
 keyword [whole|wholeright|wholeleft] [linestart] chaîne avant-plan
[arrière-plan]
```

Les chaînes de contexte ou de mots-clés sont interprétés en sorte que vous puissiez inclure des tabulations et des espaces avec les séquences \t et \s. Les sauts de ligne et le \ sont spécifiés avec \n et
respectivement. Étant donné que les caractères d'espacement sont utilisés comme séparateurs, ils ne

peut être utilisés tels quels. De plus, * doit être utilisé pour spécifier un *. Le * lui-même est un joker qui correspond à n'importe quelle longueur de caractères. Par exemple,

```
keyword          '*'      6
```

colore tous les constantes caractères C en vert. Vous auriez également pu utiliser

```
keyword          "*"      6
```

pour colorer les constantes chaînes de caractères, sauf que la chaîne reconnue ne peut s'étendre au-delà des sauts de ligne. Le joker peut également être utilisé à l'intérieur de délimiteurs de contexte, mais vous ne pouvez pas utiliser un joker comme dernier ou premier caractère.

Il est important de remarquer la ligne

```
keyword  \\\n  24
```

Elle définit un mot-clé contenant les caractères \ et saut de ligne. Puisque les mots-clés ont une plus haute priorité que les délimiteurs de contexte, ce mot-clé empêche que le contexte se termine à la fin d'une ligne si elle se termine par un \ permettant ainsi à une directive de préprocesseur C de continuer sur plusieurs lignes.

Les couleurs sont elles-mêmes numérotées de 0 à 26 et sont expliquées plus bas dans OPTIONS COMPORTEMENTALES SUPPLÉMENTAIRES. Vous pouvez également utiliser n'importe quelle couleur dont le nom est spécifié dans /usr/lib/X11/rgb.txt (en fait, uniquement celles dont le nom n'est composé que d'un seul mot). Il vaut mieux n'utiliser que les couleurs numériques pour limiter l'utilisation de la palette de couleurs.

Les commentaires peuvent être inclus sur une ligne séparée et commencent par un « # ».

À cause de la simplicité de l'implémentation, il y a quelques cas de figure complexes qui ne seront pas traités correctement mais cela ne provoque qu'un agacement mineur. Dans l'ensemble, un large spectre de situations assez compliquées est pris en compte avec ces règles simples. C'est une bonne idée de jeter un coup d'½il au fichier de syntaxe pour voir quelques uns des trucs que vous pouvez utiliser avec un peu d'imagination. Si vous ne supportez pas les règles que j'ai codées, et que vous pensez disposer d'une règle qui pourrait être utile, envoyez-moi un email avec votre requête. Néanmoins, ne demandez pas de support pour les expressions rationnelles, car cela est tout bonnement impossible.

Un conseil utile est de travailler le plus possible avec les choses que vous pouvez faire plutôt que d'essayer des choses que cette implémentation ne peut traiter. Souvenez-vous également que le but de la mise en évidence de la syntaxe est de rendre la programmation moins sujette aux erreurs, et pas d'embellir le code.

Couleurs

Les couleurs par défaut peuvent être modifiées en les concaténant à la variable d'environnement MC_COLOR_TABLE. Des paires de couleurs d'avant-plan et d'arrière-plan peuvent par exemple être spécifiées avec :

```
MC_COLOR_TABLE="$MC_COLOR_TABLE:\
editnormal=lightgray,black:\
editbold=yellow,black:\
editmarked=black,cyan"
```

Options

La plupart des options peuvent à présent être réglées à partir de la boîte de dialogue d'options de l'éditeur. Voyez le menu Options. Les options suivantes sont définies dans ~/.mc/ini, et ont des équivalents évidents dans la boîte de dialogue.

Vous pouvez les modifier pour changer le comportement de l'éditeur, en éditant le fichier.
À moins que cela ne soit mentionné, un 1 active l'option, et un 0 la désactive, comme d'habitude.

use_internal_edit

```
  Cette option est ignorée lors de l'invocation de mcedit.
```

editor_key_emulation

```
  1 pour les touches Emacs , et 0 pour les touches Cooledit normales.
```

editor_tab_spacing

```
  Interpréter le caractère de tabulation comme ayant cette longueur (8 par
défaut). Vous devriez éviter d'utiliser autre chose que 8 car la plupart des
autres éditeurs et visualisateurs de texte supposent un espacement de
tabulation de 8. Utilisez editor_fake_half_tabs pour simuler un espacement
de  tabulation plus faible.
```

editor_fill_tabs_with_spaces

```
  Ne jamais insérer d'espacement de tabulation. Insérer à la place des
espaces (code ascii 20h) pour remplir l'espace jusqu'à la taille de
tabulation désirée.
```

editor_return_does_auto_indent

```
  L'appui sur Entrée placera des tabulations pour correspondre à
l'indentation de la première ligne supérieure comportant du texte.
```

editor_backspace_through_tabs

```
  Faire en sorte qu'un simple appui sur la touche d'effacement arrière
(backspace) efface tout l'espace jusqu'à la marge de gauche s'il n'y a pas
de texte entre le curseur et la marge de gauche.
```

editor_fake_half_tabs

```
  Ceci émulera une demi-tabulation pour ceux qui veulent programmer avec un
espacement de tabulation de 4, mais qui veulent que la taille de la
tabulation reste 8 (de sorte que le code sera formaté de la même façon quand
il sera affiché par d'autres programmes). Lorsque vous éditez entre du texte
et la marge de gauche, le déplacement et les tabulations seront effectués
comme si un espacement de tabulation valait 4, alors que l'on utilise des
espaces et des tabulations normales pour un remplissage optimal. Lorsque
vous éditez n'importe où ailleurs, une tabulation normale est insérée.
```

editor_option_save_mode

```
  (0, 1 ou 2.) Le mode de sauvegarde (voyez également le menu options) vous
permet de modifier la méthode de sauvegarde d'un fichier. L'« Enregistrement
rapide » (0) enregistre le fichier immédiatement, tronquant le fichier sur
disque jusqu'à une longueur nulle (c.-à-d. l'effaçant), et écrivant le
contenu de l'éditeur dans ce fichier. Cette méthode est rapide, mais
dangereuse, car une erreur système se produisant durant la sauvegarde d'un
fichier laissera celui-ci dans un état d'écriture partielle, ce qui peut
rendre les données irrécupérables. Lors de la sauvegarde, l'« Enregistrement
sécurisé » (1) active la création d'un fichier temporaire dans lequel le
contenu du fichier est écrit en premier lieu. En cas de problème, le fichier
original n'est pas modifié. Quand le fichier temporaire est écrit
correctement, il prend le nom du fichier original, le remplaçant ainsi. La
méthode la plus sûre est de « Faire des sauvegardes » (2), où un fichier de
sauvegarde est créé avant que le moindre changement ait été effectué. Vous
pouvez spécifier votre propre extension de fichier de sauvegarde dans la
boîte de dialogue. Notez que sauvegarder deux fois remplacera aussi bien
votre sauvegarde que votre fichier original.
```

Divers

(La recherche et le remplacement avec scanf ne fonctionnaient pas correctement auparavant. Avec cette version, les problèmes liés à la recherche et au remplacement ont été résolus.)

Vous pouvez utiliser la recherche et remplacement de scanf pour chercher et remplacer une chaîne de format C. Jetez d'abord un coup d'½il aux pages de manuel de sscanf et sprintf pour voir ce qu'est une chaîne de caractères de format, et savoir comment elle fonctionne. Voici un exemple : supposez que vous voulez remplacer toutes les occurrences de, disons, un crochet ouvrant, trois nombres séparés par des virgules, et un crochet fermant, par le mot pommes, le troisième nombre, le mot oranges et ensuite le deuxième nombre, il faudrait alors remplir la boîte de dialogue Remplacement comme suit :

Entrez la chaîne à rechercher

```
(%d,%d,%d)
```

Entrez la chaîne de remplacement

```
pommes %d oranges %d
```

Entrez l'ordre des arguments

```
3,2
```

La dernière ligne spécifie que le troisième et ensuite le deuxième nombre seront utilisés au lieu du premier et du deuxième.

Il est conseillé d'utiliser cette fonctionnalité avec « Confirmation avant remplacement » activé, car on suppose avoir détecté une correspondance à chaque fois que le nombre d'arguments trouvés correspond au nombre donné, ce qui ne représente pas toujours une correspondance réelle. Scanf traite également les caractères d'espacement comme s'ils étaient élastiques. Notez que le format %[de scanf est très utile pour l'analyse de chaînes de caractères et de caractères d'espacement.

L'éditeur affiche également les caractères non-us (160 et +). Quand vous éditez des fichiers binaires, vous devriez régler Affichage des bits (NdT : ou Bits d'affichage) à 7 bits dans le menu d'options pour conserver un espacement correct.

Fichiers

```
/usr/lib/mc.hlp
```

Le fichier d'aide du programme.

```
/usr/lib/mc/mc.ini
```

Les réglages par défaut globaux au système de Midnight Commander, utilisés uniquement si l'utilisateur ne possède pas de fichier ~/.mc/ini.

```
/usr/lib/mc/mc.lib
```

Réglages globaux de Midnight Commander. Les réglages présents dans ce fichier sont globaux à n'importe quel Midnight Commander ; il est utile de définir des réglages de terminaux globaux à un site.

```
$HOME/.mc/ini
```

Configuration propre à l'utilisateur. Si ce fichier est présent, alors la configuration en est extraite et chargée au lieu du fichier de démarrage global au système.

```
$HOME/.mc/cedit/
```

Répertoire temporaire propre à l'utilisateur où les commandes de blocs sont traitées et sauvegardées.

Licence

Ce programme est distribué sous les termes de la GNU General Public License comme publiée par la Free Software Foundation. Voyez l'aide intégrée de Midnight Commander pour obtenir des détails sur

la licence, et sur l'absence de garantie.

Disponibilité

La dernière version de ce programme peut être trouvée sur les miroirs listés sur le site de GNOME http://www.gnome.org/.

Auteurs

Paul Sheer (psheer@obsidian.co.za) est le développeur de l'éditeur interne de Midnight Commander.

Bogues

Les bogues devraient être rapportés à mc-devel@gnome.org.

Source et remerciement :

- http://itx-technologies.com/man-linux/mcedit-editeur-de-texte-pour-terminal-entierement-fonctionnel-pour-systemes-de-type-unix

1)
N'hésitez pas à y faire part de vos remarques, succès, améliorations ou échecs !