

Projet shell : minifier.sh

Système et programmation système

Eric Merlet
Université de Franche-Comté

Licence 2 Informatique
2016 – 2017

Spécifications

Ce premier projet centré sur la programmation shell est prévu sur **deux séances**. N'hésitez pas à le compléter entre les deux séances.

Les fichiers transférés par un serveur WEB à destination des applications clientes (navigateurs) doivent être *les moins lourds possibles* afin de diminuer leur durée de téléchargement. Une solution consiste à *minifier* les fichiers HTML, CSS, JavaScript, c'est à dire supprimer tous les caractères qui ne servent à rien (et qui sont de toute façon supprimés par le navigateur quand il parse), à savoir les commentaires, les saut de ligne et retour chariot, les tabulations, les espaces consécutifs (remplacés par un seul).

Il faut bien-entendu veiller à ne pas modifier le rendu des pages (résultat de l'interprétation des codes HTML et CSS par le navigateur). De même, il est conseillé de vérifier les fichiers minifiés avec les validateurs W3C correspondants.

Le but du projet est d'écrire un script `minifier.sh` qui prend en argument un chemin vers le répertoire source racine d'un site web et génère dans un répertoire destination (également passé en argument) une arborescence de répertoires et de fichiers identique en "minifiant" les fichiers HTML et/ou CSS.

→ Arguments du script

Le script `minifier.sh` doit disposer d'une option `--help` :

```
$ ./minifier.sh --help
usage : ./minifier.sh [OPTION]... dir_source dir_dest
```

Minifie les fichiers HTML et/ou CSS avec :

`dir_source` chemin vers le répertoire racine du site web à minifier

`dir_dest` chemin vers le répertoire racine du site web minifié

OPTIONS

`--help` affiche l'aide et sort

`-v` affiche la liste des fichiers minifiés; et pour chaque

fichier, ses tailles finale et initiale, et son pourcentage de réduction

`-f` si le fichier `dir_dest` existe, son contenu est supprimé sans demander de confirmation d'effacement

`--css` les fichiers CSS sont minifiés

`--html` les fichiers HTML sont minifiés

si aucune des 2 options précédentes n'est présente, les fichiers HTML et CSS sont minifiés

`-t tags_file` les caractères d'"espace blanc" précédants les tags (ouvrants ou fermants) listés dans le fichier `tags_file` sont supprimés

Présence et validité des arguments :

- si l'option `--help` est utilisée, elle doit être la seule présente ;
- les options `--css`, `--html`, `-f`, `-v` et `-t tags_file` ne peuvent être présentes qu'une seule fois ;
- l'option `-t` doit être immédiatement suivie d'un chemin vers un fichier existant et lisible ;
- toute autre option doit être rejetée ;
- si l'option `--help` est absente, les 2 arguments `dir_source` et `dir_dest` doivent obligatoirement être présents ;
- l'argument `dir_source` doit être un chemin vers un répertoire existant ;
- l'argument `dir_dest` doit être différent de l'argument `dir_source` ;
- en absence de l'option `-f`, si l'argument `dir_dest` est un chemin vers un fichier (ou un répertoire) déjà existant, sa suppression doit être précédée d'une demande de confirmation d'effacement ; en cas de refus, l'opération est annulée ;

→ **Remarques**

- la profondeur de l'arborescence du répertoire source est quelconque ;
- les fichiers HTML et CSS peuvent être situés dans n'importe quel répertoire de l'arborescence ;
- on supposera que tous les noms des fichiers et des répertoires du répertoire source ne contiennent pas d'espaces, mais uniquement des lettres, des chiffres et des caractères underscore (`_`).

Réalisation

Les exercices qui suivent sont relativement indépendants et peuvent être, pour certains, traités dans le désordre.

Exercice 1 : Arguments de la ligne de commande

Dans cette partie, nous allons traiter la ligne de commande à travers les variables spéciales `$0`, `$1` et `$2`, ... ainsi que la variable spéciale `$#` qui contient le nombre d'arguments (sans le nom du script).

Question 1.1 Écrire une fonction `help` qui affiche le message d'aide indiqué ci-dessus.

Question 1.2 Afficher l'aide et quitter si l'option `--help` est la seule présente.

Question 1.3 Quand l'option `--help` est absente, vérifier la validité des arguments transmis. Pour cela, vous pouvez utiliser la commande `shift` et mémoriser les options sélectionnées dans des variables. Dès qu'une erreur est détectée, afficher un message indiquant l'origine de l'erreur, inviter l'utilisateur à utiliser l'option `--help` pour obtenir de plus amples informations, puis sortir.

Exemples :

```
$ ./minifier.sh
Les chemins vers les répertoires dir_source et dir_dest doivent être spécifiés
Saisissez « ./minifier.sh --help » pour plus d'informations.
$ ./minifier.sh -y
L'option -y n'est pas supportée
Saisissez « ./minifier.sh --help » pour plus d'informations.
$
```

Exercice 2 : Minifier les fichiers HTML

Si vous utilisez `sed`, je vous rappelle que, même si `sed` permet de réaliser des traitements multilignes, les commandes de base (`s` et `d`) présentées en cours traite le flux d'entrée ligne par ligne.

D'où, la *proposition* de traitement suivante :

Question 2.1 Remplacer tous les caractères Line Feed (`'\n'`) par un espace en utilisant la commande `tr(1)`.

Question 2.2 Supprimer les commentaires (cette étape peut être omise dans un premier temps).

Question 2.3 Remplacer les répétitions de caractères d'"espace blanc" (espace, carriage return (`'\r'`), tabulation horizontale (`'\t'`), ...) par un seul espace. RTFM `ispace(3)`.

Une fois l'étape précédente réalisée, il reste encore des caractères espace que l'on peut supprimer sans modifier le rendu de la page HTML. Ainsi, on peut supprimer les espaces qui précèdent les balises ouvrantes et fermantes des éléments HTML de type **block**. En revanche, on ne peut pas supprimer les espaces entre les éléments HTML de type **en ligne**; ces derniers constituent des boîtes anonymes "rendues" par le navigateur. D'où l'idée de stocker dans un simple fichier texte (éditable et donc modifiable par l'utilisateur) les éléments HTML de type block. Le chemin vers ce fichier `tags_file` est transmis au script via l'option `-t`.

Question 2.4 Si l'option `-t` est correctement renseignée, lire le fichier `tags_file` et mémoriser son contenu dans une variable.

Question 2.5 Pour chaque élément HTML présent dans le fichier, supprimer l'éventuel espace qui précède la balise ouvrante ou fermante de l'élément. Attention, le nom des balises n'est pas sensible à la casse et peut être écrit en minuscules, en majuscules ou dans un mélange des deux. `<body>` est équivalent à `<BODY>` qui est équivalent à `<Body>` qui est équivalent à `<BoDy>`, ...

Exercice 3 : Minifier les fichiers CSS

A vous de proposer un traitement permettant de minifier les fichiers CSS. Vous pouvez pour cela visualiser le contenu du fichier `presentation.css` minifié présent sur Moodle, et obtenu à partir du fichier correspondant du corrigé du TP1 (Langages du Web).

Exercice 4 : Reproduire l'arborescence du répertoire source dans le répertoire destination

Bien entendu, les noms des répertoires et des fichiers doivent être identiques dans les répertoires source `dir_source` et destination `dir_dest`.

Question 4.1 Si l'argument `dir_dest` est un chemin vers un fichier (ou un répertoire) déjà existant, supprimer le (si l'option `-f` est présente ou sinon, si l'utilisateur répond "oui" à la demande de confirmation d'effacement).

Ensuite, plusieurs stratégies sont possibles :

- recopier récursivement le contenu du répertoire `dir_source` dans le répertoire `dir_dest`, puis rechercher et modifier (c'est à dire minifier) les fichiers HTML et/ou CSS dans le répertoire `dir_dest` ;
- rechercher récursivement les répertoires et les fichiers présents dans le répertoire `dir_source`, et traiter un par un chaque répertoire et chaque fichier :
 - les répertoires (vides) doivent être créés dans `dir_dest`, ou dans le bon sous-répertoire
 - les fichiers non minifiés doivent être simplement copiés dans `dir_dest`, ou dans le bon sous-répertoire
 - les fichiers à minifier doivent être créés en les plaçant directement au bon endroit dans `dir_dest`

Dans tous les cas, si la création du répertoire `dir_dest` échoue, un message signalant le problème doit être affiché et le script doit se terminer. Exemple :

```
$ ./minifier.sh 24sur7 toto/tata
mkdir: impossible de créer le répertoire « toto/tata »: No such file or directory
Création du répertoire toto/tata impossible
$
```

→ quelques pistes

- RTFM `find(1)`
- RTFM `basename(1)`
- l'expansion de variable permet de soustraire un préfixe ou un suffixe à la chaîne de caractères contenue dans la variable (cf. exemples "substring processing" présentés dans le carnet de travaux pratiques)

Exercice 5 : Calcul des pourcentages de réduction de la taille des fichiers minifiés

Dans le cas où l'option `-v` (VERBOSE) est sélectionnée, le script doit afficher la liste des fichiers; et pour chaque fichier, ses tailles finale et initiale, et son pourcentage de réduction. Exemples :

```
$ ./WEB/minifier.sh WEB/24sur7 WEB/24sur7_R -f -v
Fichier CSS : WEB/24sur7_R/styles/presentation.css --> 1971 / 3202 : 38 %
Fichier HTML : WEB/24sur7_R/html/index_01.html --> 4198 / 4670 : 10 %
Fichier HTML : WEB/24sur7_R/html/index_10.html --> 6089 / 6971 : 12 %
Fichier HTML : WEB/24sur7_R/html/index_02.html --> 6019 / 6673 : 9 %
$
$ ./WEB/minifier.sh WEB/24sur7 WEB/24sur7_R -f -v -t WEB/balises.txt
Fichier CSS : WEB/24sur7_R/styles/presentation.css --> 1971 / 3202 : 38 %
Fichier HTML : WEB/24sur7_R/html/index_01.html --> 4171 / 4670 : 10 %
Fichier HTML : WEB/24sur7_R/html/index_10.html --> 6053 / 6971 : 13 %
Fichier HTML : WEB/24sur7_R/html/index_02.html --> 5986 / 6673 : 10 %
$
$ ./WEB/minifier.sh WEB/24sur7 WEB/24sur7_R -f -v -t WEB/balises.txt --html
Fichier HTML : WEB/24sur7_R/html/index_01.html --> 4171 / 4670 : 10 %
Fichier HTML : WEB/24sur7_R/html/index_10.html --> 6053 / 6971 : 13 %
Fichier HTML : WEB/24sur7_R/html/index_02.html --> 5986 / 6673 : 10 %
$
$ ./WEB/minifier.sh WEB/24sur7 WEB/24sur7_R -t WEB/balises.txt --html
Voulez vous supprimer WEB/24sur7_R ? [y/n] y
$
```

Évaluation

La qualité de l'architecture du script (découpage en fonctions, ...) sera prise en compte. Les choix d'implémentation doivent être justifiés à l'aide de commentaires dans le script. Chaque fonction doit être précédée d'un bloc de commentaires indiquant notamment son rôle et les paramètres attendus.

Vous devrez déposer une archive compressée contenant votre script et un fichier texte README dans le dépôt MOODLE prévu à cet effet avant la date indiquée. Cette archive aura pour nom les deux noms du binôme, mis bout à bout et séparés par le caractère underscore (exemple : «Dupont_Durand.tar.gz»).

- le script doit être interprétable par l'interpréteur de commandes **dash**. Dans le cas contraire, votre rendu serait considéré comme hors sujet ;
- le fichier README doit contenir :
 - une conclusion/bilan sur le produit final (ce qui est fait, testé, non fait)
 - un bilan technique sur les outils utilisés
 - une présentation des améliorations possibles mais non réalisées
 - un bilan par rapport au travail en binôme