Introduction à Bash Utils

Dimitri OBEID

2021

Table des matières

1 Introduction à Bash Utils

1.1 Présentation

Bash Utils est une librairie, c'est à dire un ensemble de fonctions utilitaires, regroupées et mises à disposition afin de pouvoir être utilisées sans avoir à les réécrire.

Comme son nom le suggère, il s'agit d'une librairie orientée vers le langage Bash, un langage de script permettant une interaction simple avec le shell (interpréteur de commandes) du système d'exploitation, ainsi qu'avec ses différents programmes.

Bash Utils est divisé en deux parties : la librairie, et les modules, expliqués plus en détails dans le fichier **modules/Fonctionnement.pdf**.

Le shell utilisé est le Bash, qui est de loin le shell le plus utilisé dans le monde d'UNIX. Ce choix s'explique non seulement par sa vaste présence, mais aussi par sa très vaste documentation disponible et ses fonctionnalités (notamment les tableaux).

1.2 Utilisation

Pour utiliser cette librairie, il vous suffira de sourcer un seul fichier, qui est installé par défaut dans votre dossier personnel :

- **$HOME/.Bash-utils-init.sh**

Ce script initialise tous les modules de la librairie passés en argument lors de l'exécution de ce dernier.

1.3 Architecture

La partie librairie de Bash Utils est un ensemble de fichiers source shell à sourcer dans un script, d'outils de développement ou encore de scripts déjà prêts à l'exécution.

Liste des dossiers :

1.3.1 bin

Ce lien symbolique (systèmes UNIX seulement) pointe vers le dossier **res/dev-tools/dev-bin**.

Ce dossier contient les fichiers exécutables nécessaires au bon fonctionnement de la librairie.

1.3.2 docs

Ce dossier contient toute la documentation utile aux développeurs et aux utilisateurs, disponible en plusieurs langues.

1.3.3 install

Ce dossier contient les fichiers et dossiers nécessaires à l'installation automatique de la librairie.

1.3.4 lib

Ce dossier est le plus important de tous, car il contient tous les fichiers source de la librairie (fonctions et variables).

Il contient les sous-dossiers suivants :

- **functions :** Ce sous-dossier contient tous les fichiers de fonctions dans différents sous-dossiers.
- **lang :** Ce sous-dossier contient tous les fichiers nécessaires à la traduction des scripts (fonctionnalité future).

Description des sous-dossiers du dossier functions :

1.3.4.1 compiled

Ce dossier contient les sous-dossiers suivants :

- **stable** : ce dosssier contient
- **unstable** : ce dosssier contient

Les différences entre les fichiers des deux sous-dossiers mentionnés ci-dessus sont expliquées dans la section **Bash-utils-(stable)-lang.sh**

1.3.4.2 functions

Chaque sous-dossier est lié à un module et contient les fichiers source de ce dernier.

Chaque fichier source contient des fonctions qui sont propres au nom du fichier. Par exemple, le fichier **Files.lib** du module **main** ne contient que des fonctions de traitement de fichiers (création, suppression, archivage, obtention des permissions, etc...).

1.3.4.3 lang

Rappel : fonctionnalité future, certaines idées peuvent encore changer.

Le fichier **lang.csv** contient les différentes traductions de la librairie dans un fichier au format CSV. Il sera parsé (analysé) par le script d'initialisation.

1.3.4.4 variables

Chaque fichier définit des variables propres aux catégories décrites dans le nom de chaque fichier. Par exemple, les variables définies dans le fichier **colors.var** sont des variables définissant le code de la couleur d'un texte à afficher.

1.3.5 res

Ce dossier contient plusieurs sous-dossiers :

- **dev-tools :** Ce dossier contient des scripts aidant au développement de la librairie, avec leurs dépendances (fichiers binaires, fichiers raccourcis **desktop**, icônes et fichiers sources dans le cas où un fichier serait trop long).
- **graphs :** Ce dossier contient des graphiques (faits avec le logiciel Draw.io) servant à visualiser l'architecture du fonctionnement d'un composant du framework.
- **ideas :** Ce dossier contient des fichiers où j'écris mes idées de fonctionnalités à implémenter dans la librairie.
- **src :** Ce dossier contient des scripts préparés facilitant l'utilisation d'un système UNIX, en utilisant les fonctionnalités implémentées par la librairie.
- **tests :** Ce dossier contient de nombreux scripts servant ou ayant servi de fichiers de tests pour de nombreuses fonctionnalités, implémentées ou en cours d'implémentation.

1.3.6 tmp

Ce dossier, absent lors du téléchargement de la librairie, est créé par le fichier d'initialisation du module **main**, et sert à enregistrer les fichiers temporaires générés par un projet, tels que des fichiers de logs, de traduction, etc...

1.3.7 win

À l'instar du symlink (lien symbolique) **bin**, ce raccourci Windows (pour les utilisateurs de Windows Subsystem for Linux, sachant que les symlinks UNIX ne sont pas nativement reconnus par l'explorateur de fichier de Windows) pointe vers le dossier **res/dev-tools/dev-bin**.

2 Fonctionnement

Au vu du nombre de fichiers, il serait très fastidieux de mettre à jour chaque script pour sourcer de nouveaux fichiers, voire même de sourcer manuellement chaque fichier, puis de déclarer des variables, quelques fonctions, de configurer et vérifier chaque étape d'initialisation du script principal.

Il serait même encore plus fastidieux de réécrire tout ce code, ainsi que de le modifier dans chaque fichier en cas d'ajout de fonctionnalités ou en cas de bug, et encore plus de répéter toutes ces étapes pour chaque module à ajouter, voire à supprimer au cas où.

Et tout ceci, sans même parler de l'ajout d'autres modules, où dans certains cas, l'utilisateur voudra que certaines fonctionnalités réagissent différemment dans des situations précises.

Fort heureusement, trois scripts shell au choix s'occupent de cette partie : **Bash-utils-init.sh**, situé dans le dossier personnel, **Bash-utils-lang.sh**, situé dans le dossier **.Bash-utils/compiled/unstable** ou **Bash-utils-stable-lang.sh**, situé dans le dossier **.Bash-utils/compiled/stable**.

La seule procédure que vous devez réaliser dans votre script pour utiliser les fonctionnalités proposées par Bash Utils est de sourcer l'un des deux fichiers cités ci-dessus. Chacun d'entre eux s'occupera de réaliser toutes les étapes ci-dessous :

2.1 Script $HOME/Bash-utils.sh

Si vous choisissez d'opter pour l'inclusion de ce fichier,

- Inclure le fichier "Aliases.conf", listant les alias de chaque fonction de librairie du module, si un tel fichier est présent dans le dossier des fichiers de configuration du module.
- Inclure le fichier de configuration "Module.conf", qui liste les chemins des autres fichiers de configuration, puis le fichier d'initialisation dudit module.
- Inclure le script d'initialisation, qui inclut les fichiers de librairie associés au module, ainsi que les fichiers de configuration listés dans le fichier "Module.conf" mentionné ci-dessus.

Les fichiers mentionnés sont décrits plus en détils dans ce fichier :

- **Bash-utils/docs/fr/modules/Fonctionnement.pdf**.

Les fonctions utilisées sont documentées dans ce fichier :

-

2.2 Bash-utils-(stable)-lang.sh

Note : Rien ne différencie techniquement les fichiers **.Bash-utils/compiled/unstable/Bash-utils-lang.sh** et **.Bash-utils/compiled/stable/Bash-utils-stable-lang.sh**, si ce n'est qu'il est absolument recommandé de ne pas toucher au dernier fichier mentionné, car :

- il est inclus dans les fichiers exécutables des devtools
- il a été débogué avec la commande **Shellcheck**, dont le code de retour commande est le code 0, ce qui indique que cet outil n'a détecté aucune erreur de programmation, de style, de frappe, etc...
- son code a été testé en profondeur pour garantir une certaine stabilité et une exécution sans bugs majeurs, avant que l'ensemble des fichiers ne soient compilé en un seul fichier

Fonctionnement : Étant donné que le code du module principal entier et du script d'initialisation (plus leurs fichiers de configuration) est contenu dans un seul fichier, aucun fichier supplémentaire n'est inclus.

Cependant, chaque étape d'initialisation du framework est exécutée de la même manière que lors de l'inclusion du script **$HOME/Bash-utils-init.sh**.

2.3 Étapes d'initialisation

2.3.1 Variables de modules

En premier lieu, le script d'initialisation initialise les variables globales contenant les chemins des multiples dossiers du gestionnaire de modules.

2.3.2 Version de Bash

En second lieu, le script vérifie que la version minimale de Bash utilisée soit la version 4.0.0.

2.3.3 Boucle d'initialisation

En troisième lieu, il exécute une boucle pour initialiser chaque module passé en argument lors de l'exécution de ce script d'initialisation.

D'abord, il vérifie que le module existe dans le dossier **~/.Bash-utils**, puis il source le fichier d'initialisation de variables associé, situé dans le dossier **~/.Bash-utils/config/modules/$nom_du_module** et nommé **module.conf**, qui doit initialiser d'autres fichiers de configurations à placer de préférence dans ce même dossier.

Ensuite, le script en fait de même pour le fichier d'initialisation des fonctions de librairie associé, situé dans le dossier **~/.Bash-utils/modules/$nom_du_module**, et nommé **Initializer.sh**, avant de reboucler tant que tous les modules n'ont pas été initialisés.

2.3.4 Variables de statut

En quatrième et dernier lieu, le script modifie des variables de statut, dont la valeur initialement assignée dans le fichier **~/.Bash-utils/config/modules/main/Status.conf** servait à initialiser la librairie.

2.4 Étapes post-inclusion

---

3 Documentations des éléments de Bash Utils

3.1 Fonctions

Tous les fichiers de documentation en français concernant la description des fonctions se situent dans le dossier **docs/fr/Bash/functions**.

3.2 Variables

Tous les fichiers de documentation en français concernant la description des variables se situent dans le dossier **docs/fr/Bash/variables**.

Par ailleurs, dans les fichiers de documentation des fonctions, les chemins absolus des dossiers du gestionnaire de module et de la librairie sont, respectivement, représenté ainsi :

- **$__BU_MAIN_ROOT_DIR_PATH :** Cette variable globale enregistre le chemin du dossier racine de la librairie Bash Utils, celui qui contient toutes les documentation, les fichiers sources et des fichiers de ressources.

- **$__BU_MODULE_UTILS_ROOT :** Cette variable globale enregistre le chemin du dossier racine des modules la librairie Bash Utils, celui qui est installé dans le dossier personnel de chaque utilisateur, et qui contient tous les fichiers de configuration et d'initialisation de chaque module.

La première variable est définie dans le fichier **$HOME/Bash-utils-init.sh**, tandis que la seconde variable est définie dans le fichier **$HOME/.Bash-utils/config/modules/main/module.conf**.

---

4 Installation de Bash Utils

4.1 Configurations

Pour utiliser les fonctionnalités de la librairie depuis n'importe quel chemin sans avoir à réécrire le chemin du fichier d'initialisation dans une myriade de fichiers éparpillés sur le disque dur, il est fortement recommandé d'installer le gestionnaire de modules de la façon décrite dans le fichier **README INSTALL.md**.

4.2 Installation

4.3 Mise à jour