

Tutoriels d'aide aux ajouts basiques

Dimitri OBEID

2021

Table des matières

1	Liste des tutoriels	3
1.1	Ajouter une nouvelle langue à la librairie	3
1.1.1	Fichiers de configuration à éditer	3
1.1.2	Fichiers de librairie à éditer	3
1.1.3	Consignes	3
1.2	Ajouter une nouvelle langue à la documentation	3
1.2.1	Fichiers de configuration à éditer	3
1.2.2	Fichiers de librairie à éditer	4
1.2.3	Consignes	4
1.3	Ajouter une nouvelle couleur	4
1.3.1	Fichiers de configuration à éditer	4
1.3.2	Fichiers de librairie à éditer	4
1.3.3	Consignes	4
1.4	Ajouter un nouveau module	5
1.4.1	Fichiers à exécuter	5
1.4.2	Consignes	5
1.5	\$_TITRE	6
1.5.1	Fichiers à éditer	6
1.5.2	Consignes	6
1.6	\$_TITRE	6
1.6.1	Fichiers à éditer	6
1.6.2	Consignes	6
1.7	\$_TITRE	6
1.7.1	Fichiers à éditer	6
1.7.2	Consignes	6

1 Liste des tutoriels

Note : Pour des raisons de lisibilité, veuillez créer les nouvelles fonctions et variables dans l'ordre alphabétique de chacune de leurs liste.

1.1 Ajouter une nouvelle langue à la librairie

1.1.1 Fichiers de configuration à éditer

- `$_BU_MODULE_UTILS_ROOT/config/modules/main/LangCSVCode.conf`
- `$_BU_MODULE_UTILS_ROOT/config/modules/main/LangISOCode.conf`

1.1.2 Fichiers de librairie à éditer

- `$_BU_MAIN_ROOT_DIR_PATH/lib/lang/SetLibLang.sh`

1.1.3 Consignes

Ouvrez le fichier de configuration `LangCSVCode.conf`, puis dans la liste des variables globales, créez une variable globale nommée `__BU_MAIN_LANG_CSV_CODE_`, suivie immédiatement du code ISO 639-1 de la langue que vous souhaitez ajouter.

Ensuite, ouvrez le fichier de librairie `SetLibLang.sh`, puis rendez-vous dans la fonction `SetLibLang`. Une fois dedans, ajoutez une nouvelle condition dans la structure de contrôle `case`, en écrivant le code ISO 639-1 de la langue cible, suivi d'un tiret du 8 ' _ ', utilisé par le système pour relier ce code avec le code du pays ISO 3166, avant d'ajouter une étoile '*' pour ne pas à se soucier de la gestion des différences de langue par pays.

Une fois la condition créée, définissez la en créant une variable nommée `__BU_MAIN_LANG_CSV_CODE`, en lui assignant la variable que vous venez de définir dans le fichier de configuration `LangCSVCode.conf`.

Appelez ensuite la fonction `ParseCSVLbLang`, en lui passant en argument :

- Le chemin du fichier de traduction de la librairie (vous pouvez passer en argument la valeur `$_BU_MAIN_MODULE_LIB_LANG_DIR_PATH/lang.csv`)
- La variable globale `$_BU_MAIN_LANG`, qui enregistre la langue choisie, définie plus bas dans le fichier, dans la fonction `GetLibLang`.
- Le message de succès de recherche du chemin du fichier CSV, dans la langue que vous souhaitez définir.
- Le message d'erreur du header de la fonction `HandleErrors`
- Le message de conseil de la fonction `HandleErrors`

Pour plus de praticité, vous pouvez créer un fichier nommé `LangISOCode.conf` dans le dossier `$_BU_MODULE_UTILS_ROOT/config/modules/main`, et écrire le code ISO 639-1 de votre langue, suivie du code ISO 3166 d'un des pays où la langue à ajouter est parlée.

Pour encore plus de praticité, si la langue que vous souhaitez ajouter est définie dans les paramètres de votre système d'exploitation, vous pouvez taper la commande suivante, en remplaçant la variable `$_BU_MODULE_UTILS_ROOT` par le chemin de la racine du dossier de configurations de Bash Utils (normalement localisé dans votre dossier personnel) :

- `echo $LANG > "$_BU_MODULE_UTILS_ROOT/config/modules/main/LangISOCode.conf"`

1.2 Ajouter une nouvelle langue à la documentation

1.2.1 Fichiers de configuration à éditer

- `$_BU_MODULE_UTILS_ROOT`
-

1.2.2 Fichiers de librairie à éditer

- `$_BU_MODULE_UTILS_ROOT`
-

1.2.3 Consignes

1.3 Ajouter une nouvelle couleur

1.3.1 Fichiers de configuration à éditer

- `$_BU_MODULE_UTILS_ROOT/config/module/main/colors.conf`
-

1.3.2 Fichiers de librairie à éditer

- `$_BU_MAIN_ROOT_DIR_PATH/lib/functions/main/Decho.lib`
 - `$_BU_MAIN_ROOT_DIR_PATH/lib/functions/main/Headers.lib`
-

1.3.3 Consignes

Dans le fichier de configuration `colors.conf`, repérez la section **COLOR ENCODING**, puis la sous-section **COLOR CODES FOR ENCODING**, puis créez une nouvelle variable globale de code couleur, de préférence nommée :

- `__BU_MAIN_COLOR_CODE_<$nom_de_la_couleur_en_anglais>`

Suivie de son code ANSI de la table des couleurs XTERM.

Pour plus d'informations concernant les valeurs de cette table, veuillez cliquer [ici](#), ou exécuter le script `256-color-table.sh`, localisé dans le dossier `$_BU_MAIN_ROOT_DIR_PATH/res/dev-tools/dev-bin`.

Une fois le code couleur ajouté, repérez, dans le même fichier, la sous-section **ENCODING WITH THE "tput" COMMAND AND PRINTED AND REDIRECTED WITH THE "CheckTextColor" FUNCTIONS**, puis créez une nouvelle variable globale de nom de couleur nommée comme la variable créée précédemment, sans le `_CODE`, puis appelez la fonction `CheckTextColor`, suivie du nom de la variable créée dans la sous-section **COLOR CODES FOR ENCODING**.

Regardez la manière dont la fonction `CheckTextColor` est appelée, pour vous aider.

Maintenant, rendez vous dans le fichier de librairie

- `$_BU_MAIN_ROOT_DIR_PATH/lib/functions/main/Decho.lib`

Puis repérez la sous-catégorie suivante :

- **WRITING DIFFERENTLY COLORED/FORMATTED TEXT BETWEEN TEXT - LIST**

Repérez les premières fonctions, toutes nommées `Decho_<$nom_de_couleur_en_anglais>`, puis ajoutez une fonction pour chaque couleur en copiant simplement l'une des fonctions en la renommant `Decho_<$nom_de_la_nouvelle_couleur>` et en remplaçant le second argument de la fonction `Decho` par le nom de la variable globale de nom de couleur associée.

Ensuite, rendez-vous dans le fichier de librairie `$_BU_MAIN_ROOT_DIR_PATH/lib/functions/main/Headers.lib`, puis dans la sous-section **UNICOLOR HEADERS**. Repérez la liste de fonctions nommées :

- `Header<$nom_de_la_couleur_en_anglais>`

Copiez simplement l'une des fonctions, puis renommez la `Header<$nom_de_la_couleur_en_anglais>`, et modifiez les noms des premier et troisième arguments de la fonction `HeaderBase`, en les remplaçant par le nom de la variable globale de noms de couleurs souhaitée.

Rendez vous ensuite dans la sous-section suivante, nommée **BICOLOR HEADERS**, puis pour chaque liste de fonction par couleur, copiez-collez une fonction, puis rennommez la en ne changeant que le nom de la deuxième couleur, puis le nom du troisième argument de la fonction `HeaderBase`.

Finalement, pour chaque couleur que vous ajoutez, créez une nouvelle liste de fonctions bicolores, toujours en respectant l'ordre alphabétique du nom des couleurs en anglais (voir les commentaires avant chaque liste).

Créez une fonction pour chacune des couleurs à supporter, nommée `Header`, suivie du **nom de la nouvelle couleur**, suivie enfin du **nom d'une autre couleur**.

Modifiez la valeur du premier argument en appelant la variable globale de nom de couleur associée à la nouvelle couleur, puis la valeur du troisième argument en appelant la variable globale de nom de couleur dédiée au nom de l'autre couleur à afficher.

Encore une fois, n'hésitez pas à jeter un œil à l'organisation des autres listes de fonctions d'affichage bicolore pour vous aider.

1.4 Ajouter un nouveau module

Note : Le fonctionnement des modules est détaillé dans la section **Introduction** du fichier `Bash-utils/docs/fr/modules`.

1.4.1 Fichiers à exécuter

- `$_BU_MAIN_ROOT_DIR_PATH/res/dev-tools/dev-bin/mkmodules.sh`

1.4.2 Consignes

De manière automatique :

- Rendez-vous EXACTEMENT dans le dossier du fichier shell `mkmodules.sh`, puis exécutez juste ce script en lui passant en argument les noms des modules à créer.
- **Astuce** : Vous pouvez vous y rendre directement via le lien symbolique `bin` dans le dossier racine de la librairie. **Attention**, pour ceux qui utilisent WSL, ce lien ne marche pas dans le moindre explorateur de fichiers sur Windows, mais uniquement en ligne de commande avec votre distribution Linux.

De manière manuelle :

- Rendez vous dans le dossier de configurations des modules `.Bash-utils`, puis rendez vous dans le dossier `config/modules`. Créez un nouveau dossier pour chaque module à créer, puis dans chacun de ces dossiers, créez un fichier nommé `module.conf`.
- Revenez de nouveau dans le dossier racine `.Bash-utils`, puis allez dans le dossier `modules`. Créez un nouveau dossier pour chaque module, puis dans chacun de ces dossiers, créez un fichier nommé `Initializer.sh`.

1.5 \$ _TITRE

1.5.1 Fichiers à éditer

—

1.5.2 Consignes

1.6 \$ _TITRE

1.6.1 Fichiers à éditer

—

1.6.2 Consignes

1.7 \$ _TITRE

1.7.1 Fichiers à éditer

—

1.7.2 Consignes
