# Fonctions du fichier de librairie Headers.lib

# 1 Présentation

## 1.1 Présentation

Ce fichier source inclut des fonctions servant à la création de headers, séparant ce qu'un développeur de scripts Bash peut imaginer comme étant deux grandes étapes différentes d'un script.

## 1.2 Définitions des éléments mentionnés

### 1.2.1 Chemins de fichiers

**Nom du fichier : Checkings.lib**

**Dossier parent : $__BU_MAIN_ROOT_DIR_PATH/lib/functions/main**

---

**Nom du fichier : Colors.conf**

**Dossier parent : $__BU_MODULE_UTILS_ROOT/config/modules/main**

---

**Nom du fichier : Status.conf**

**Dossier parent : $__BU_MODULE_UTILS_ROOT/config/modules/main**

---

**Nom du fichier : Text.conf**

**Dossier parent : $__BU_MODULE_UTILS_ROOT/config/modules/main**

---

### 1.2.2 Fonctions externes appelées

**Fonction : IsChar**

**Description :** Cette fonction verifie que la valeur enregistrée dans une variable soit un simple caractère alphanumérique.

**Fichier de définition : Checkings.lib**

---

### 1.2.3 Variables globales externes et / ou variables d'environnement appelées

**Variable : $__BU_MAIN_COLOR...**

**Description :** Chacune des variables globales commençant par ce nom retourne un code couleu encodé lors de l'appel d'une de ces variables, grâce à l'appel simultané d'une fonction.

**Fichier de définition : Colors.conf**

---

**Variable :** $__BU_MAIN_STAT_ECHO

**Description :** Cette variable globale sert à mettre le script dans un état dit “stable”, dans le cas où un rappel de la même fonction dans l’état précédent peut causer une boucle infinie.
(( vérifiant si un texte peut être redirigé vers un fichier de logs sans provoquer de boucles infinies en rappelant certaines fonctions ))

**Fichier de définition :** Status.conf

---

**Variable :** $__BU_MAIN_TXT_CHAR_HEADER_LINE

**Description :** Cette variable globale enregistre le caractère par défaut à afficher sur chaque colonne d’une ligne d’un header.

**Fichier de définition :** Text.conf

---

# 2 Fonctions

## 2.1 Création de base d’un header

**Fonction :** DrawLine

**Description :**
Cette fonction dessine une ligne en remplissant chaque colonne du terminal avec un caractère choisi et passé en second argument.

**p_lineColor** –> Ce paramètre attend un code couleur *.
**p_lineChar** –> Ce paramètre attend un seul caractère, qui sera affiché sur chaque colonne.

* Chaque code couleur est défini dans le fichier **Colors.conf**.

**Fonctionnement :**
La fonction DrawLine vérifie d’abord que la valeur du second paramètre **$p_lineChar** soit bien un simple caractère (alphabétique ou numérique) via la fonction IsChar, définie dans le fichier **Checkings.lib**. Si ce n’est pas le cas, ce paramètre est redéfini, et seul le premier caractère de la chaîne est récupéré pour l’affichage de la ligne.

Cette fonction vérifie ensuite qu’une valeur soit passée en premier argument (paramètre **p_lineColor**). Si c’est le cas, la commande **echo** est appelée avec ses options **-ne** pour encoder la couleur de la ligne de caractère à dessiner.

Une fois ceci fait, une boucle **for** est exécutée. Pour un nombre **i** allant de 1 au nombre de colonnes présentes dans la fenêtre du terminal obtenu grâce à la commande **eval echo**, on affiche **i** fois, sans retour à la ligne, le caractère passé en deuxième argument (**$p_lineChar**) dans le fichier de configurations **Text.conf**.

Enfin, une fois la boucle exécutée, la fonction ré-encode la couleur d’affichage selon la couleur par défaut du terminal, si le paramètre **$p_lineColor** contient une valeur.

**Utilisation :**
Appelez cette fonction pour dessiner une ligne devant remplir la totalité des colonnes d’une ligne du terminal.

**Fonction : HeaderBase**

**Description :**
Cette fonction affiche un header complet.

**Paramètres :**

| | |
|---|---|
| **p_lineColor** | Ce paramètre attend un code couleur |
| **p_lineChar** | Ce paramètre attend un seul caractère |
| **p_stringColor** | Ce paramètre attend un code couleur |
| **p_stringTxt** | Ce paramètre attend une chaîne de caractères |

**Fonctionnement :**
Tout d'abord, la fonction **HeaderBase** vérifie l'état de la variable de statut **$__BU_MAIN_STAT_ECHO**, puis en adaptant la commande d'affichage de texte à appeler, cette fonction appelle la fonction **DrawLine** en y passant en arguments ses deux premiers paramètres.

Une fois ceci fait, un saut de ligne est effectué, et un message est affiché, avec la couleur choisie encodée en premier (troisième paramètre) et le texte passé en quatrième argument.

La commande de ré-initialisation de décoration du terminal (définie dans les fichiers **Colors.conf** et **Text.conf**) est ensuite appelée pour réinitialiser la couleur d'affichage de texte selon la configuration du terminal.

**Utilisation :**
Appelez cette fonction pour afficher un texte entre deux lignes, et pour davantage personnaliser cet affichage si vous ne trouvez pas votre compte parmi les fonctions proposées dans les sections suivantes : **Headers unicolores** et **Headers multicolores**.

---

## 2.2 Headers unicolores

**Description :**
Chacune de ces fonctions appelle la fonction **HeaderBase**, en lui passant ses trois premiers arguments prédéfinis.

**Paramètre :**
**$1** –> Ce paramètre attend la chaîne de caractère à afficher.

**Fonctionnement :**
Chacune des fonctions de cette catégorie appelle la fonction **HeaderBase**, en lui passant en premier et troisième arguments un même code couleur **$__BU_MAIN_COLOR...**, un caractère défini par la variable **$__BU_MAIN_TXT_CHAR_HEADER_LINE** en deuxième argument, puis la chaîne de caractères à afficher en quatrième argument (seul paramètre que prennent ces fonctions).

**Utilisation :**
Appelez une de ces fonctions pour afficher un header unicolore sans avoir à passer ses trois premiers arguments.

---

## 2.3 Headers bicolores

Pour davantage faciliter la tâche des développeurs, des fonctions de création de headers bicolores existent.

Elles fonctionnent chacune de la même manière que les fonctions de la catégorie précédente, à deux exceptions près :

- **le nom :** chaque fonction porte le nom de la couleur à afficher sur chaque ligne, puis le nom de la couleur à appliquer au texte.

- **le nom (bis) :** Il n'y a pas de fonction portant deux fois le nom de la même couleur, car le rendu serait le même que celui des fonctions de la catégorie précédente.

---

## 2.4 Headers tricolores

Étant donné la charge de travail nécessaire pour créer ces headers, il serait beaucoup trop fastidieux de créer tous ces headers prédéfinis.

Pour vous donner une idée de la quantité de travail, prenez la formule mathématique suivante :

$$nbcol * (nbcol - 1) * nbcol$$

Où nbcol = nombre de codes couleurs disponibles.

Au moment où j'ai écrit cette partie de la documentation (dimanche 29 août 2021), il existait 12 codes couleurs disponibles.

En effectuant le calcul précédent, nous obtenons donc **12 * 11 * 12 = 1 584 combinaisons possibles**, sans parler de l'ajout d'éventuels nouveaux codes couleurs.

Il faudra donc se contenter d'appeler la fonction **HeaderBase**, suivie de ses 4 arguments, dont chaque valeur sera différente des autres.