Manuel d'utilisation

Introduction :

L'eXiaSaver est un écran de veille pour le mode console. Il propose deux écrans choisis aléatoirement, soit un écran statique affichant une image en noir et blanc parmi un jeu de cinq images, soit un écran dynamique qui affiche l'heure, rafraichie toutes les dix secondes.

Il propose aussi une option statistique qui permet d'afficher par exemple l'historiques des lancements de l'eXiaSaver ou le nombre de fois où chaque écran a été lancé.

Commandes d'utilisation :

Ici est décrit chaque commande à entrer pour lancer chaque option de l'eXiaSaver.

./Exia_Saver : cette commande lance l'eXiaSaver.

./Exia_Saver -stats : Cette commande lance les statistiques de l'eXiaSaver.

Appuyer sur une touche pendant l'écran de veille statique : permet de quitter l'écran de veille statique.

Ctrl^C : permet de quitter l'écran de veille dynamique.

Partie 1 : Le Lanceur

Notre lanceur eXiaSaver se compose de plusieurs fonctions, la première est la fonction random, elle permet de retourner un int d'une valeur comprise entre 1 et 3.

Ensuite la fonction choice utilise le return de la fonction random comme paramètre d'un switch.

Dans le cas 1, c'est le saver screen 1 qui sera lancé, là encore on utilise un random pour récupérer une valeur comprise entre 1 et 5, celle- ci correspondra à une des 5 images dans notre bibliothèque. Enfin le programme statique prend la main (sa fonction sera expliquée dans une autre partie).

Dans le cas 2, le lanceur demande à l'utilisateur deux int à l'aide de scanf, ces int servirons de permettre à la fonction qui redimensionnera les images du timer. L'envoi de ces paramètres coïncide avec le lancement du programme Dynamique.

Dans le cas 3, il ne se passe rien car nous n'avons pas eu le temps de l'implémenter.

Nous l'avons tout de même intégré afin de mieux exposer l'aspect aléatoire de l'eXiaSaver.

Partie 2 : Le Statique

Notre programme se compose de 4 fonctions principales :

La première fonction lit le nombre de caractère en largeur et longueur afin de construire un tableau qui contiendra les valeurs binaires composant les fichiers PBM.

La secondes permet de remplir le tableau, un pointeur se déplace dans le fichier et insert les données une part une dans le tableau.

Et la troisième fonction est une fonction d'interprétation, elle reprend chaque valeur du tableau et les remplace par des caractères définis dans le code.

Enfin ; une dernière fonction assure l'affichage au centre de la console, elle utilise le nombre de caractères en longueur et largeur récupéré par la première fonction.

Notre jeu d'images se compose de 5 fichiers au format PBM :

-Le fantôme de PAC-MAN,

-Le logo d'Overwatch,

-Le casque de Genji (personnage du jeu Overwatch),

-Isaac tiré du jeu « The binding of Isaac »,

-un casque de stormtrooper (Star Wars),

-le château donné dans le guide du projet pour combler le default de notre switch.

Partie 3 : Le Dynamique

L'écran de dynamique est un écran de veille qui affiche l'heure actuelle (Paris) centrée dans la console. Elle est rafraichie toutes les dix secondes.

En bas de la console un message apparait disant : « Cet écran sera actualisé dans quelques secondes » suivis d'un point («.») qui s'ajoute en fin de ligne toute les secondes. Une fois arrivé à 10 points et donc 10 seconde, l'heure et actualisé et le schéma se répète.

L'affichage des chiffres de l'heure sont un ensemble d'images en «. pbm » qui sont centrées dans la console.

Pour quitter cet écran de veille, l'utilisateur doit « tuer le processus » en effectuant la commande Ctrl^C.