Projet 2 : EXIA-Saver

Sommaire :

[Présentation du projet : 1](#_Toc469584056)

[Analyse du besoin : 1](#_Toc469584057)

[Composition des groupes : 1](#_Toc469584058)

[Planning prévisionnel et réel 2](#_Toc469584059)

[Les différents types d’écran de veille 3](#_Toc469584060)

[Les fonctions d’affichages 4](#_Toc469584061)

[Programmation sur code block et console 4](#_Toc469584062)

[L’historique 4](#_Toc469584063)

[Conclusion 4](#_Toc469584064)

# Présentation du projet :

Notre projet Exia-saver consistait à créer un écran de veille sur la console Linux avec un tirage aléatoire sur le choix du fond d’écran : statique, dynamique ou interactif.

# Analyse du besoin :

Créer un programme permettant la lecture d’un document PBM et son affichage sur la console avec différentes actions selon l’écran de veille choisi.

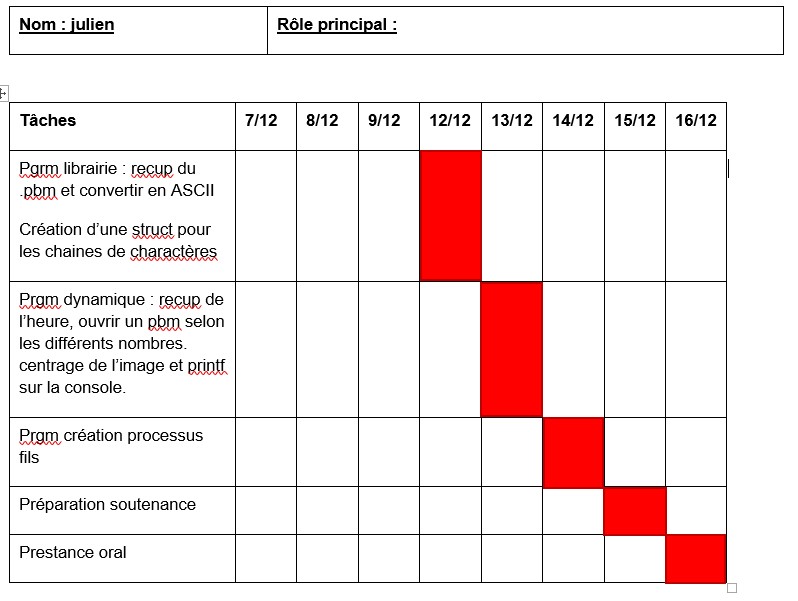
# Composition des groupes :

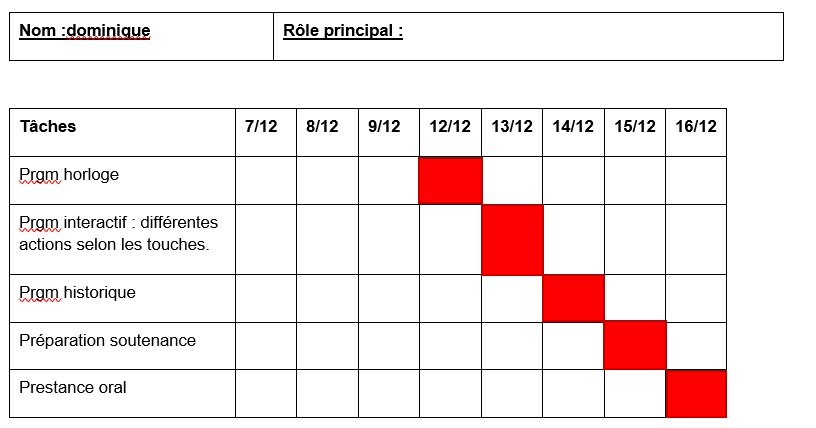
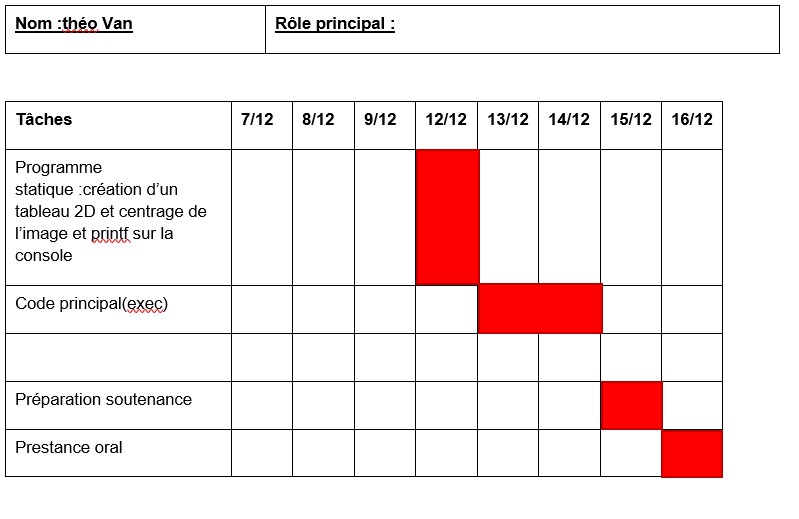
Théo V

Julien J

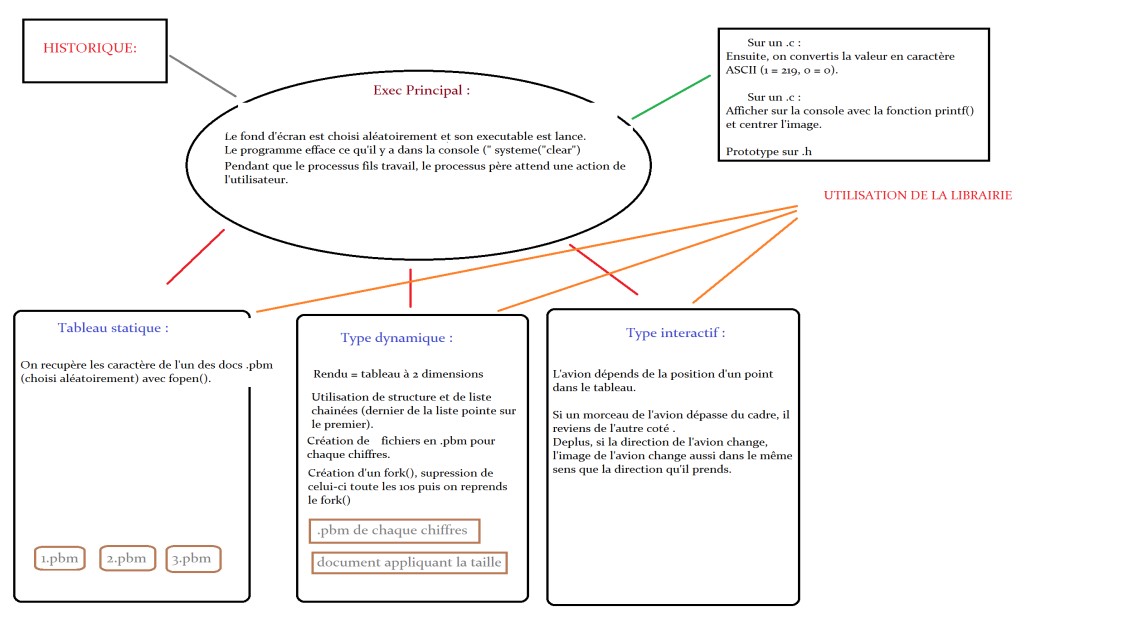
Dominique S

# Planning prévisionnel et réel





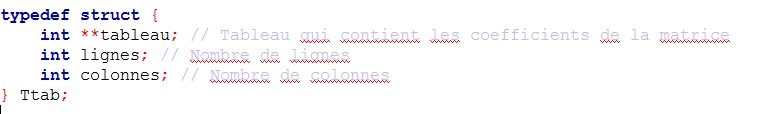
**Choix techniques**



Pour ce projet, nous avons choisi de réaliser ce programme sur code blocks windows et linux.

**Prototypes de nos fonctions**

Pour les prototypes, nous avons utilisé une structure contenant un tableau à 2 dimensions, la longueur et largeur de l’image à afficher

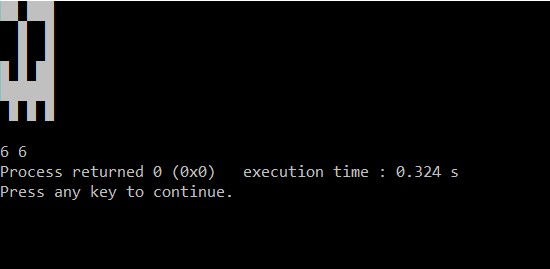


## **Les différents types d’écran de veille**

Notre programme contiendra plusieurs fonctions, dans la fonction principale on choisira aléatoirement un des 3 types d’écran de veille à lancer. On efface au début la console.

Les 3 types sont :

Statique : Pour créer le programme statique, nous avons créé différentes fonctions, une fonction qui permet d’ouvrir un document pbm, le scanner chaine par chaine en filtrant les commentaires et le p1, en retour nous récupérons un tableau 2 dimensions.



Nous avons également séparé les dimensions du tableau pour pouvoir les utiliser pour le centrage et l’affichage .

Dynamique : Pour créer le dynamique, nous utilisons la fonction sfrtime pour récupérer l’heure qui sera stockée dans une chaine de caractère, il faudra donc la filtrer. Après cela, nous mettons un switch qui ouvrira un pbm et le filtrera. Nous prendrons les tableaux de chaque chiffre pour le mettre dans un grand, on le centrera et l’affichera. Il faut aussi rafraichir l’horloge en un temps donné qu’on demandera au début du programme et avec la fonction sleep, on rafraîchit

Interactif : pour ce programme on demande à l’utilisateur d’entrer une certaine touche du clavier. Selon cette touche, nous décalerons le tableau.

## **Les fonctions d’affichages**

Après avoir récupérer le tableau à 2 dimensions contenant les caractères du fichier .pbm, nous cherchons le point en haut à gauche du tableau ou nous souhaitons imprimer l’image.

Ensuite nous parcourrons toutes les cases du tableau représentant l’écran jusqu’à arriver dans l’intervalle des x et des y ou nous souhaitons imprimer l’image, dès que nous dépassons l’extrémité des x de l’intervalle, nous passons à la ligne.

## **Programmation sur code block et console**

Pour exécuter le programme dans le terminal, Nous devons créer des répertoires définis dans nos ressources. Selon le programme lancé, nous pouvons utiliser certains répertoires. Nous utilisons fork et exec pour manipuler

## **L’historique**

Nous stockons dans le fichier historique.txt les informations sur l’écran sélectionné (la date et l’heure d’utilisation et, selon le programme sélectionné, le nom du fichier pbm utilisé pour l’écran statique, la taille d’affichage de l’horloge de l’écran dynamique, les coordonnées de base de l’écran de veille interactif).

## **Conclusion**

Bilan du projet :

Ce projet fut intéressant et nous a permis d’approfondir nos connaissances en C. Pour autant, nous avons était confronté à quelques problèmes comme :

* Lorsque nous avons programmé sur code blocks sous windows, nous n’avons pas été confrontés à tous les problèmes de compilation qu’il y avait sur Linux
* Une mauvaise structure et planification du projet : nous n’avons pas pu penser à toute les fonctions et la structure de notre programme.
* Nous avons manqué également de temps et de quelques compétences en C