[2022/2023]

DEV SAE 1.1
Jeu de paires

tom.moguljak@etu.u-pec.fr ; lyanis.souidi@etu.u-pec.fr

Table des matières

tom.moguljak@etu.u-pec.fr ; lyanis.souidi@etu.u-pec.fr

I. Introduction

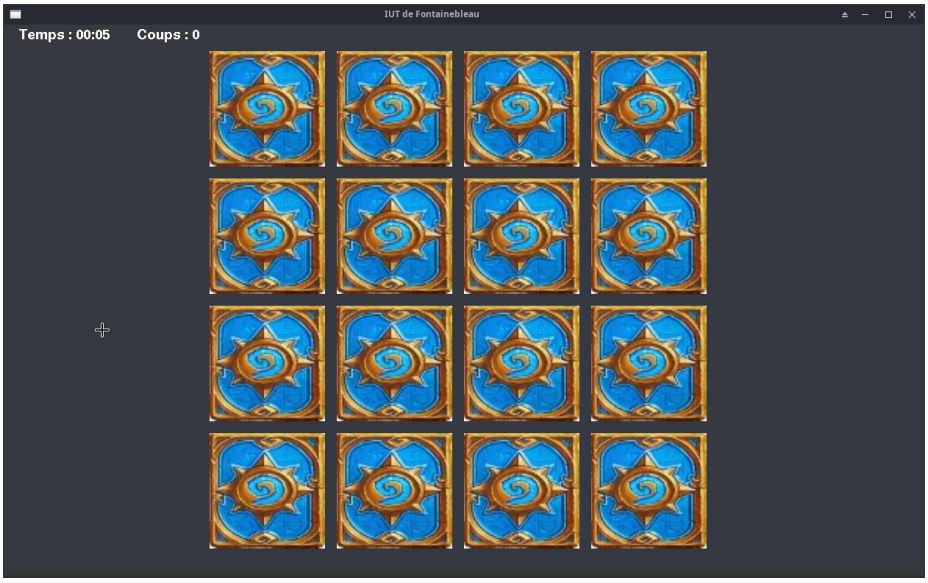
Dans cette SAE, nous allons devoir programmer en C89 un jeu de paires. Le jeu de paires est un jeu de mémorisation avec des paires de cartes portant des illustrations identiques. Les joueurs vont alors retourner chacun leur tour deux cartes, si ces dernières sont identiques, on les laisse retournées et c'est encore à lui de jouer, mais si ces dernières ne sont pas identiques, dès lors on les remet faces cachées et c'est au joueur suivant de jouer. Le jeu se termine une fois que toutes les cartes sont découvertes.

II. Fonctionnalités

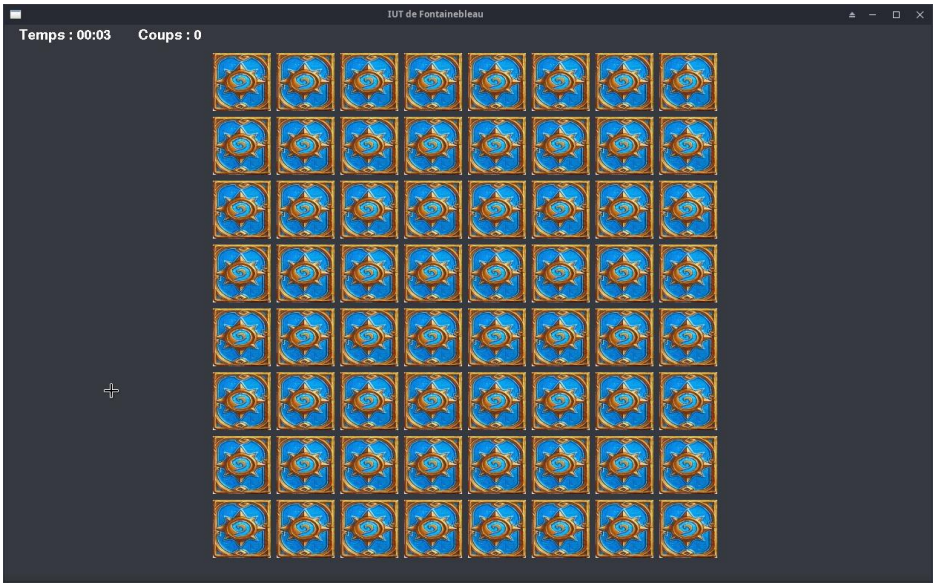
Afin de parler des fonctionnalités de notre programme, intéressons-nous tout d'abord au menu principal. Ce dernier possédant plusieurs fonctionnalités.

Tout d'abord, nous pouvons voir qu'il y a plusieurs boutons sur le menu principal, un bouton "Facile", "Moyen", "Difficile" et un bouton "Quitter". Quand nous appuyons sur les boutons "Facile", "Moyen" ou "Difficile", ils vont appeler une fonction "game", qui va créer un tableau, avec un nombre de colonne et de ligne, qui change en fonction de la difficulté choisie, mais aussi en choisissant aléatoirement un certain nombre d'image parmi toutes celles que nous avons, tout en les dédoublant pour avoir des paires et en les mélangeant. Pour le mode "Facile" nous avons un tableau 2x2 avec 2 images uniques, pour le mode "Moyen" un tableau 4x4 avec 8 images uniques et pour le mode "Difficile" un tableau 8x8 et donc avec 16 images uniques.
Le bouton "Quitter" quant à lui, comme son nom l'indique, va nous permettre de fermer le programme.

Le tableau de jeu du niveau facile

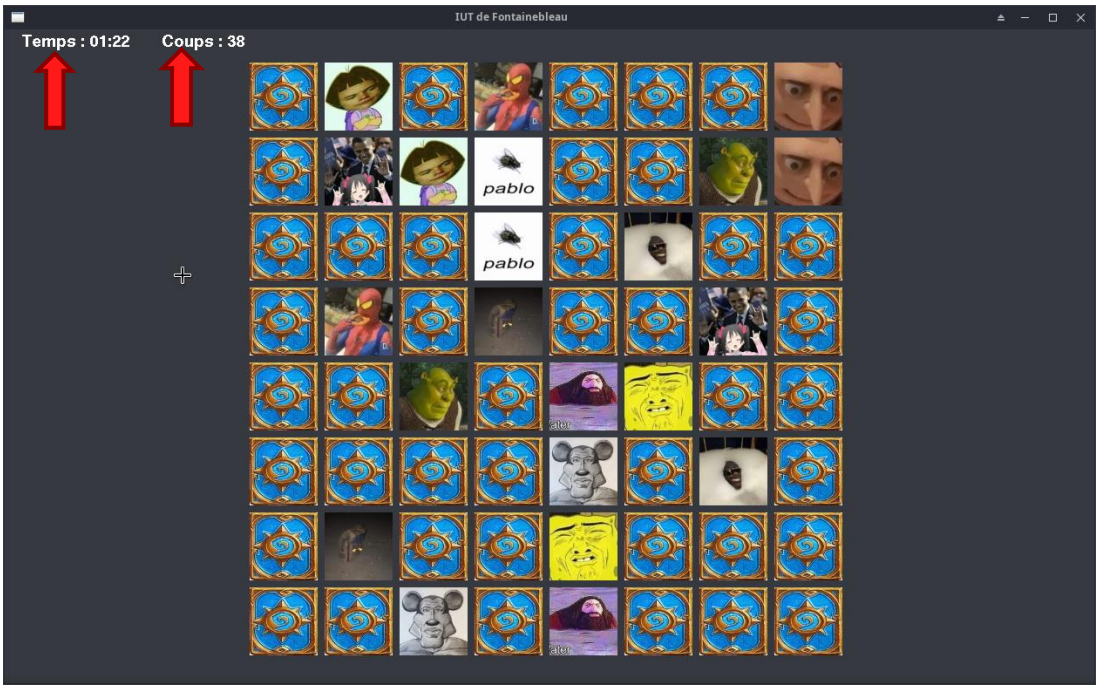


Le tableau de jeu du niveau moyen

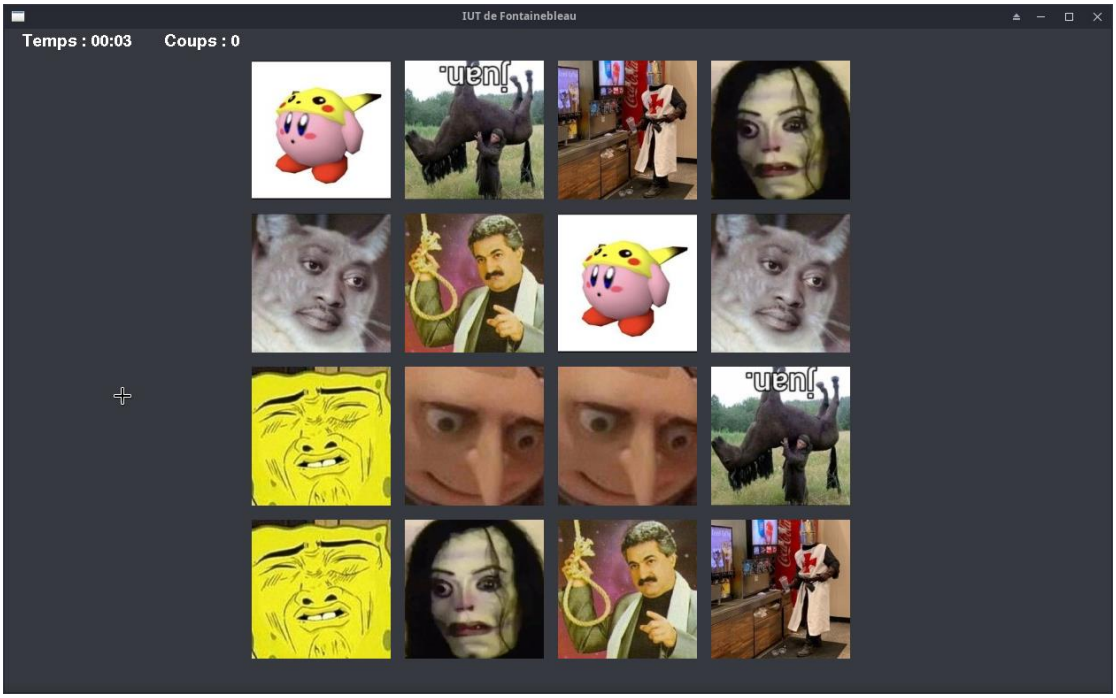


Le tableau de jeu du niveau difficile

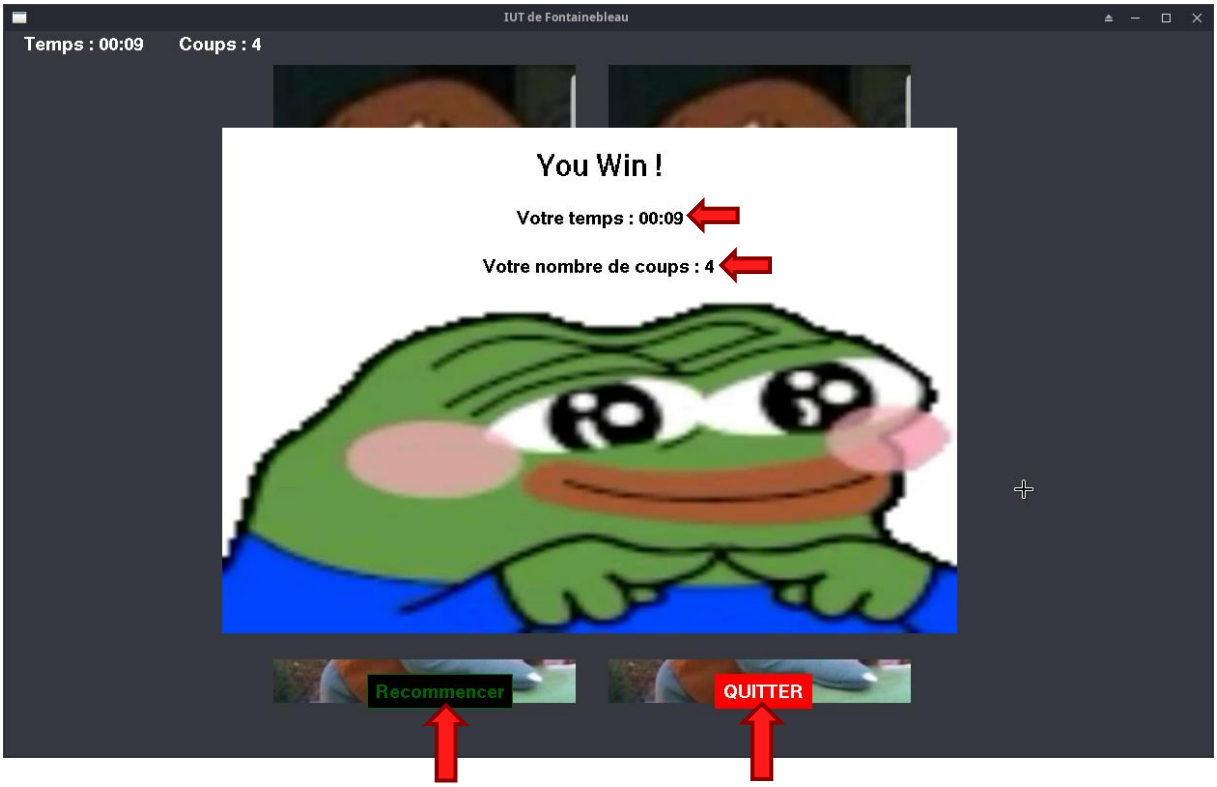
Après cela nous arrivons sur l'écran de jeu, on retrouve sur cet écran, un chronomètre, qui va permettre au joueur de voir son temps pendant qu'il joue, et à la fin, d'avoir son temps final. Nous avons aussi implémenté un compteur de coups, qui augmente à chaque clic de souris sur les cartes.



Nous pouvons aussi retrouver un mode tricheur, qui s'active en appuyant sur la touche "t" va activer le mode tricheur, ce mode a pour but de retourner toutes les cartes, mais aussi de stopper le chronomètre et le compteur de coups, et quand nous appuyons de nouveau sur "t" le mode tricheur se désactive et ainsi le jeu peut reprendre comme avant.



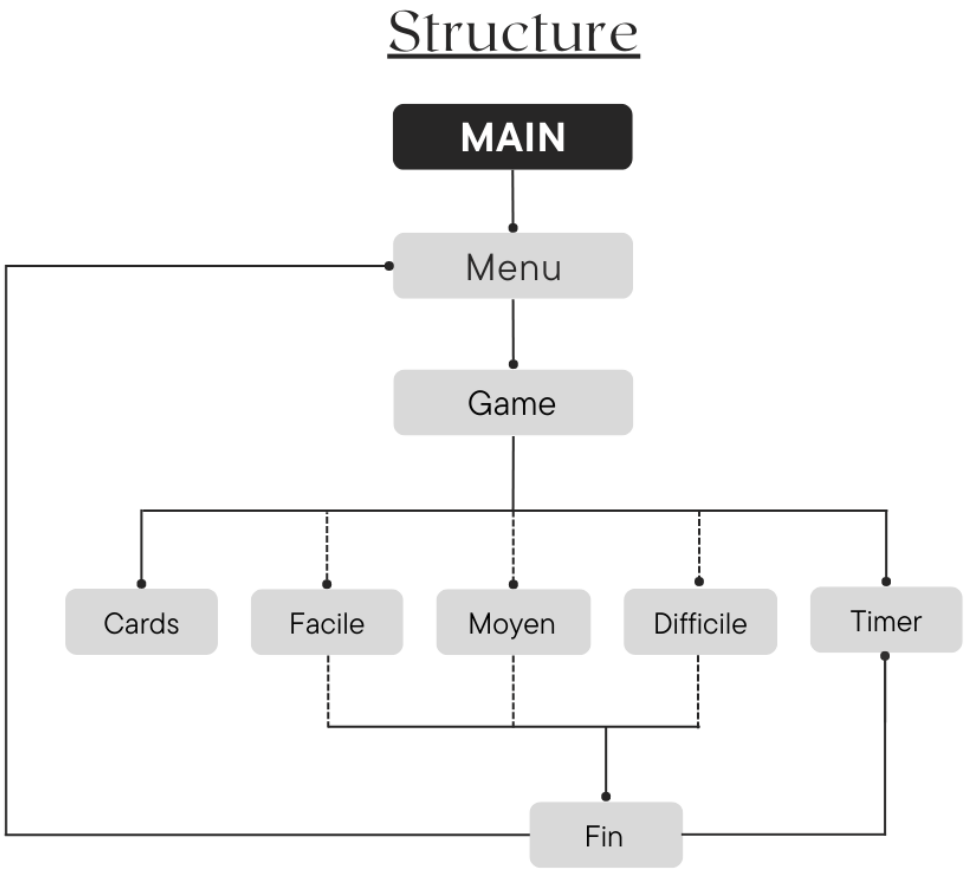
Après avoir retourné toutes les cartes, une fonction "fin" va être appelé afin de faire apparaître l'écran de fin. Dans cet écran, nous retrouvons plusieurs choses, d'une part le temps final du joueur donné grâce au chronomètre mais aussi le nombre de coups réalisé pendant la partie, ensuite un bouton "Recommencer", qui a pour but de nous faire revenir au menu principal et un bouton "Quitter" qui possède la même fonction que celui du menu principal, c'est-à-dire, arrêter le programme.



tom.moguljak@etu.u-pec.fr ; lyanis.souidi@etu.u-pec.fr

III. Structure

Nous avons décidé de structurer notre programme de comme ceci, car il nous semblait qu'un découpage de cette façon était le plus logique à faire.



Tout en haut de cette structure, nous retrouvons bien sûr le fichier "Main", qui va avoir pour but de définir toutes les fonctions, les variables qui vont être utiles au programme. Par exemple c'est lui qui va initialiser la bibliothèque graphique, et créer la fenêtre.

Il va aussi, par le biais d'une fonction, appeler le fichier "menu", qui à son tour va afficher l'interface du menu principal (le fond, les boutons etc…). En plus de faire ça, le menu va nous permettre de quitter le programme avec le bouton "quitter", et avec les boutons "Facile", "Moyen" et "Difficile", le menu va appeler la fonction "game", du fichier du même nom. Cependant il y a une subtilité, chaque bouton appelle la fonction "game" mais avec des arguments différents, car cela va influencer sur la construction du jeu. Cette action sur les boutons est possible grâce à la fonction "check_zone", définie dans "main", qui permet de savoir si la position de la souris quand nous cliquons avec, est dans une zone que nous avons définie au préalable.

Comme dit plus haut, la construction du tableau du fichier "game" est influencé par le bouton choisi, car ce dernier va nous donner des valeurs "colonnes" et "lignes" différentes, qui vont ainsi construire un tableau qui peut changer en fonction des parties. Par exemple en appuyant sur le bouton "Facile", nous allons avoir un tableau de 2x2, alors qu'avec le bouton "Difficile", nous allons avoir un tableau 8x8. En plus de cela, et avec l'aide de la fonction "cards", appelé par "game", le programme va pouvoir choisir un certain nombre d'images de façon aléatoire dans notre bibliothèque, pour ensuite les dupliquer et ensuite les ranger de façon aléatoire pour pouvoir les afficher en même temps que le tableau.

Pendant le jeu, en haut à gauche de l'écran, se trouve un chronomètre, qui indique au joueur son temps de jeu, c'est en appelant les fonctions du fichier "timer". Par exemple avec la fonction "uptade_timer", qui permet le bon fonctionnement du chronomètre en l'actualisant chaque seconde. Les fonctions du chronomètre va être accompagné d'un compteur de coups, qui s'actualise à chaque clique qu'on fait pour retourner une carte, cela grâce à la fonction "update_coups" définie dans "game". En plus de ces deux fonctions, nous avons implémenté dans "game", une partie de programme qui va activer le mode tricheur en appuyant sur la touche "t". En effet, le mode tricheur permet de voir toutes les cartes mais aussi stopper le chronomètre et le compteur de coups en appuyant sur "t", et pour revenir à la normal, nous appuyons de nouveau sur "t".

Une fois le jeu terminé, c'est-à-dire quand toutes les cartes sont retournées, "game" va appeler le fichier "fin". Dans un premier temps, le fichier fin, va créer l'interface de l'écran de fin, cette interface possède plusieurs éléments, notre chronomètre de fin, rendu possible par les fonctions du chronomètre et le nombre de coups que nous avons fait pendant la partie, grâce à la fonction "update_coups". Dans un premier temps, le fichier fin, va créer l'interface de l'écran de fin, cette interface possède plusieurs éléments, notre chronomètre de fin, rendu possible par les fonctions du chronomètre et le nombre de coups que nous avons fait pendant la partie, grâce à la fonction "update_coups". Nous allons aussi retrouver 2 boutons, le boutons recommencer, qui va avoir pour but de nous ramener au fichier "menu", et un bouton quitter, qui lui de son côté va nous servir à arrêter le programme.

IV. Données

Nous avons choisi de créer des structures pour les données les plus utilisées, les zones (*zone*) et les cartes (*card*).

Les zones contiennent 4 informations, la position horizontale (*x*) et verticale (*y*) du coin supérieur gauche dans la fenêtre, la longueur (*L*) et la hauteur (*H*) de la zone. Ces zones sont utilisées pour les identifier les zones cliquables tel que les boutons ainsi que les cartes.

Les cartes contiennent quant à elles 6 informations, une zone, 3 variables indiquant l'état de la carte, un identifiant (*id*) partagé avec sa paire et un chemin d'accès vers l'image de la carte (*file*). Les 3 variables d'état indiquent si la carte est affichée à l'écran (*displayed*), si elle est trouvée (*found*) et si elle est chargée (*loaded*), cette dernière information permet de charger le fichier d'image une seule fois lords de la partie, suite à cela la zone contenant l'image est copié depuis un écran virtuel vers l'écran principal.

Un tableau multidimensionnelle nommé *cards* contient toutes les cartes de la partie, ce tableau contient évidement des structures *card* ce qui nous permet de faire des boucles sur ce tableau pour parcourir toutes les cartes.

V. Algorithme

Afin de remplir le tableau de cartes, notre algorithme commence par faire une sélection aléatoire d'images pour les cartes, afin de faciliter ce processus, nos cartes ont toutes la même extension de

fichier et leur nom est un chiffre allant de 1 à 60 (le chiffre 0 correspond à la carte retrounée), une boucle appelle ensuite une fonction *create_card()* permet de créer les structures des cartes et les renvoi ensuite pour les stocker dans le tableau multidimensionnelle *cards*. Lors de la boucle, des calculs sont effectuées afin de définir l'emplacement et la taille des cartes en fonction de la fenêtre de tel sorte qu'en cas de changement de la taille de la fenêtre (via les constantes *WINDOW_WIDTH*/*HEIGHT*), les calculs se modifient automatiquement pour correspondre à la nouvelle taille. Un formatage de chaîne de caractère est également effectué afin de définir le chemin d'accès vers la carte, ce chemin d'accès est différent en fonction de la taille du tableau et évidement du nom du fichier de la carte.

VI. Conclusion

Tom : En conclusion, je pense que cette SAE m'a apporté beaucoup d'expérience et de connaissance, en retravaillant sur toutes les notions qu'on a pu voir durant le début d'année, mais aussi en apprenant de toutes nouvelles notions. Cela a pu aussi nous apprendre à gérer notre temps et la gestion d'un projet, tout en nous initiant au travail d'équipe, et donc à la répartition du travail. Malgré le stress que ce projet a pu engendrer, je suis personnellement content du travail que l'on a fourni, et du résultat obtenu.

Lyanis : Cette SAÉ nous a permis de rendre concret nos cours de développement mais aussi de mieux nous familiariser avec l'outil git, j'ai grandement apprécié participer à ce projet qui possède encore de nombreuses pistes d'amélioration inexplorées.

tom.moguljak@etu.u-pec.fr ; lyanis.souidi@etu.u-pec.fr