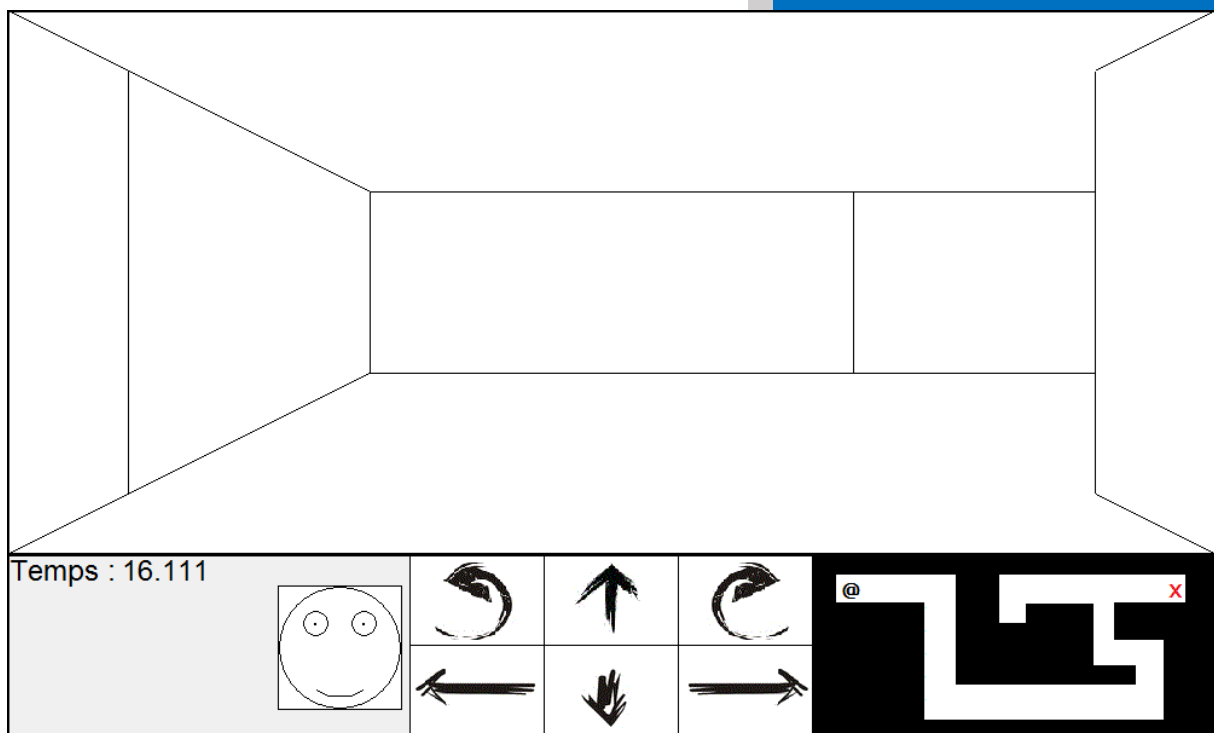


Cahier des charges fonctionnel

Sujet 12 : Le labyrinthe





Sommaire

Présentation du projet

- Consignes
- Fonctionnalités obligatoires
- Fonctionnalités facultatives
- Difficultés rencontrées
- Partage des tâches
- Diagramme de Gantt

Fonctionnement du programme

- Choix du module graphique
- 'fonctions_du_la_byrinthe.py'
- 'graphismes_du_labyrinthe.py'
- 'lancement_du_jeu.py'

Sources et images du jeu

- Liens utilisés
- Images utilisées
- Images du jeu



Présentation du projet

Consignes :

Le but de ce projet est de représenter une vue subjective des couloirs composants un labyrinthe ainsi que de permettre le déplacement dans ces couloirs à l'aide d'une zone de 6 boutons.

La raison pour laquelle nous avons décidé de choisir ce sujet est parce que, par rapport aux autres sujets, il est celui qui se rapproche le plus de la conception d'un jeu moderne. Sa réalisation nous permettra alors d'acquérir un minimum d'expérience dans la fabrication d'un jeu en Python3.

La conception de ce projet nous a permis de mettre en avant les connaissances en programmation acquise durant ce premier semestre durant les cours d'algorithmique et programmation et de laisser place à notre imagination sur la réalisation du projet.

Fonctionnalités obligatoires :

Voici une liste des fonctionnalités qu'ils nous étaient nécessaires d'implémenter dans notre programme :

- Vue subjective des couloirs du Labyrinthe
- Déplacement a l'aide d'une zone de 6 boutons
- Validité des mouvements (ne pas traverser les murs)
- Détection de la fin de la partie
- Importation du labyrinthe depuis un fichier texte
- Existence de murs à plus d'une case de distance
- Définition de fonctions pour les tâches répétitives

- Utilisation de docstrings pour chaque fonction définie

Fonctionnalités facultatives :

Voici une liste des fonctionnalités et de suggestions pouvant être implémentées dans notre programme afin de l'améliorer une fois que toutes les fonctionnalités obligatoires seront présentes :

- Traduction du programme en anglais
- Affichage d'une carte
- Possibilité de sauvegarde
- Création d'étages
- Génération aléatoire du labyrinthe (ou du moins de plusieurs blocs créés antérieurement changeant de position aléatoirement)
- Génération de PNJ et de leur rencontre
- Génération d'ennemi(s) ayant un déplacement
- Création de couleurs et de textures pour graphismes
- Affichage de messages ou de panneaux d'affichage
- Récupération d'objets ayant une fonctionnalité (une boussole, une carte etc.)
- Affichage du temps et d'un ou plusieurs records
- Affichage du visage du perso dans l'HUD (à la manière de Doom)
- Affichage de statistiques de vie, de dommage et de l'arme actuellement portée
- Création d'un mode « Aventure » séparé en plusieurs niveaux (avec une difficulté croissante). Dans ce cas les labyrinthes seront préconçus.
- Ajout d'un mode « Arcade » où les labyrinthes sont générés aléatoirement en plus du mode « Aventure »
- Ajout d'un menu de sélection permettant de choisir le mode de jeu ou de quitter le jeu

- Possibilité de se déplacer grâce aux touches du clavier et non en cliquant sur des boutons

Difficultés rencontrées :

La partie qui nous posait problème en premier lieu, fut l'imagination du fonctionnement du programme. En effet, il était difficile de commencer à écrire un programme sans savoir où nous allions tout en gardant à l'esprit que ce que nous écrivions ne servait peut être à rien.

La création des graphismes restera cependant LE problème majeur durant toute la réalisation du projet. C'était tant un problème que nous nous en sommes occupé en tout dernier.

Enfin, les fonctions de déplacement du personnage furent longues et compliquées à écrire. Elles nous donnèrent beaucoup de fil à retordre et nous firent changer le type de variable utilisé pour contenir les coordonnées ainsi que les éléments du labyrinthe à plusieurs reprises.

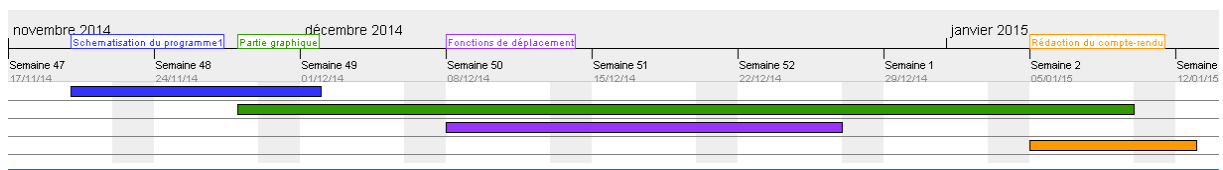
Partage des tâches :

Au départ, nous avons séparé la conception du programme en deux grandes parties : une partie graphique, et une sur le déroulement du programme. Au bout d'un certain temps nous nous rendîmes compte que la seconde partie posait des problèmes de compatibilité avec la partie graphique. De ce fait, le travail partagé devint un travail en commun simultané. Une fois la partie contenant les fonctionnalités opérationnelles, nous concentrèrent nos efforts sur la partie graphique, celle-ci fut difficile à comprendre et redondante à programmer.

Nicolas s'occupa de relier les fonctions de graphismes sans arguments à la fonction 'affichage_graphismes' et Mohamed se chargea de

l'écriture de ces fonctions ainsi que de la conception des différents menus.

Diagramme de Gantt :





Fonctionnement du programme

Choix du module graphique :


Pour réaliser l’affichage du jeu ainsi que tout ce qui concerne les graphismes, nous devons utiliser un module.

Notre choix s’est porté sur celui proposé dans l’énoncé, ‘upemtk’. En effet, bien que moins complet, son utilisation était moins risquée que celle de ‘tkinter’ car en cas de problèmes, il nous suffisait de demander de l’aide aux professeurs de la fac.

Fonctions du labyrinthe.py :

Ce fichier contient toutes les fonctions venant à être appelées dans le fichier d’exécution principal ‘lancement_du_jeu.py’. En voici une liste ainsi qu’une brève description de chacune :

- **Chargement_du_labyrinthe** : Insère le labyrinthe dans une grosse liste
- **Attribution_des_actions** : Assigne le clic sur un bouton à une action
- **Déplacement_personnage** : Effectue le déplacement du personnage en fonction de l’action demandée. Elle empêche le personnage de traverser les murs

- 
- **Affichage_menu_principal** : Affiche le menu principal
 - **Affichage_choix_du_niveau** : Affiche le menu de choix du niveau
 - **Affichage_HUD** : Affiche l'HUD du jeu contenant une carte, le visage du personnage, les boutons de déplacement et le temps
 - **Affichage_graphismes** : Affiche les graphismes du jeu contenant les murs et les virages
 - **Calcul_profondeur_vision** : Calcule la distance entre le personnage et le prochain mur en face de lui
 - **Boucle_du_jeu** : Utilise les fonctions servant à faire fonctionner le jeu
 - **Calcul_position_virages** : Calcule les positions des virages se trouvant à gauche et à droite du personnage
 - **verification_nouveau_record** : Vérifie si le temps mit par le joueur est un des meilleurs temps du niveau ou non
 - **affichage_meilleurs_scores** : Affiche le menu de victoire avec les meilleurs scores du niveau



Graphismes du labyrinthe.py :

Ce fichier contient toutes les fonctions sans arguments qui seront appelées lors de l'utilisation de la fonction 'affichage_graphismes'. Elles servent à dessiner les murs et les virages du labyrinthe dans toutes les situations impliquant une profondeur ne dépassant pas 3 cases. Si la profondeur est plus grande que 3 cases, le mur d'en face est simplement retiré car il est visuellement trop loin pour être aperçu.

Sources et images du jeu

Liens utilisés :

Voici une liste des liens nous ayant aidé dans la conception de ce programme :

- www.ilay.org/yann/articles/maze/
- <http://sametmax.com/les-docstrings/>
- <https://docs.python.org/3.2/>
- <https://www.python.org/dev/peps/pep-0257/>
- <http://weblog.jamisbuck.org/2011/1/10/maze-generation-prim-s-algorithm>
- http://python.developpez.com/cours/apprendre-python3/?page=page_11
- <http://openclassrooms.com/courses/apprenez-a-programmer-en-python/les-listes-et-tuples-1-2>

Images du jeu :

Voici les différents menus pouvant être atteints depuis le fichier d'exécution, ainsi que quelques images in-game :

