Sam FREDDI

CFPT Ecole Informatique

Maître d’apprentissage : Pascal Henauer

Résolution d’un labyrinthe par la méthode d’un arbre de recherche

Documentation technique

Table des matières

[Introduction 3](#_Toc164146903)

[Rappel du cahier des charges 3](#_Toc164146904)

[Objectif et description de l’application 3](#_Toc164146905)

[Organisation du suivi 3](#_Toc164146906)

[Livrables 3](#_Toc164146907)

[Matériel et outils nécessaires 3](#_Toc164146908)

[Méthodologie de travail 4](#_Toc164146909)

[La méthode en 6 étapes 4](#_Toc164146910)

[1- S’informer 4](#_Toc164146911)

[2- Planifier 4](#_Toc164146912)

[3- Décider 4](#_Toc164146913)

[4- Réaliser 4](#_Toc164146914)

[5- Contrôler 4](#_Toc164146915)

[6- Évaluer 5](#_Toc164146916)

[Méthodologie de sauvegarde 5](#_Toc164146917)

[Méthode du 3-2-1 5](#_Toc164146918)

[Planification 5](#_Toc164146919)

[Planification prévisionnelle 5](#_Toc164146920)

[Planification effective 5](#_Toc164146921)

[Diagramme général de l’application 5](#_Toc164146922)

[Analyse des fonctionnalités 6](#_Toc164146923)

[Analyse organique 6](#_Toc164146924)

[MCD 6](#_Toc164146925)

[MLD 6](#_Toc164146926)

[Architecture 6](#_Toc164146927)

[Structure MVC 7](#_Toc164146928)

[Librairies et outils utilisés 7](#_Toc164146929)

[PHP 8.1 7](#_Toc164146930)

[Bootstrap Studio (Bootstrap 5.3) 7](#_Toc164146931)

[DBeaver 24.0.1 7](#_Toc164146932)

[GitHub 2.34.1 (WSL) 8](#_Toc164146933)

[Apache 2.4.52 (WSL) 8](#_Toc164146934)

[MariaDB 15.1 (WSL) 8](#_Toc164146935)

[Chemin des données 8](#_Toc164146936)

[Algorigramme d’une méthode complexe 8](#_Toc164146937)

[Les tests 8](#_Toc164146938)

[Périmètre 8](#_Toc164146939)

[Environnement 8](#_Toc164146940)

[Scénarios 8](#_Toc164146941)

[Suivi évolutif des tests 8](#_Toc164146942)

[Conclusion 9](#_Toc164146943)

[Difficultés rencontrés 9](#_Toc164146944)

[Amélioration possibles 9](#_Toc164146945)

[Bilan personnel 9](#_Toc164146946)

[Remerciements 9](#_Toc164146947)

[Bibliographie 9](#_Toc164146948)

[Glossaire 9](#_Toc164146949)

[Annexes 9](#_Toc164146950)

# Introduction

Ce document est destiné aux experts en charge du suivi de l’application Résolution d’un labyrinthe par la méthode d’un arbre de recherche dans le cadre du TPI (travail pratique individuel).

Ce document retrace les points clés de la réalisation de ce travail.

# Rappel du cahier des charges

Cette partie correspond à la première étape de la méthode en 6 étapes.

## Contexte

Afin de valider notre formation de CFC en Informatique au CFPT-I, il nous est demandé d’effectuer un travail pratique individuel en fin de 4e année. Pour ce travail, il m’est demander de créer une application génération et solutions de labyrinthe en 3D.

## But du projet

## Description de l’application

Dans le cahier des charges, on me demande de réaliser…

Le projet du TPI décrit vise à créer un logiciel capable générer un labyrinthe de manière aléatoire avec des blocs prédéfinit et de trouver le chemin le plus court entre une entrée et une sortie dans un labyrinthe.

Lorsqu’une génération de carte est terminée. L’utilisateur a la possibilité de sauvegarder la carte et de l’importer depuis le menu.

L'algorithme utilisé repose sur la recherche dans un arbre, avec un bot doté de capteurs et d'une trace du parcours. À chaque nœud, le bot est dupliqué dans toutes les directions possibles. Les bots qui rencontrent un mur meurent, laissant derrière eux une trace rouge affichée pendant une demi seconde. A la fin de l’algorithme seule ceux qui trouvent la ou les sorties seront afficher avec une trace verte et si aucune sortie n’est trouvé une trace jaune sera assigné a tracé le plus proche de la sortie. Les bots en cours de recherche auront eu une trace bleue.

L’algorithme de recherche est divisé en 3 modes distincts. Le mode manuel, semi-automatique et automatique.

Le mode manuel a pour but de trouver le chemin le plus court entre une entrée et une sortie donné par l’utilisateur. Si aucun chemin ne relie l'entrée et la sortie, le programme le fait savoir à l'utilisateur et lui propose 4 options. Changer de sortie. Changer de mode, relancer une génération ou importer une carte sauvegardée.

Le mode semi-automatique a pour but de trouver toutes les sorties possibles avec une entrée donnée par l’utilisateur. L’algorithme va tester tous les chemins possibles avec l’entré donné et nous retourner toutes les sorties possibles.

L'objectif du mode automatique est d'évaluer la performance dans le labyrinthe. L'algorithme explore chaque entrée du labyrinthe pour déterminer le nombre de sorties, le temps nécessaire pour achever le labyrinthe et la distance parcourue par les bots depuis cette entrée. Après avoir effectué le test sur chaque entré le programme nous retourne le score du labyrinthe. Un tableau avec le numéro de chaque entré est visible. Quand l’utilisateur clique sur une entré du tableau le programme nous affiche le parcours effectué par les bots depuis cette entré et nous donne tous les valeurs liées à cette entré.

Si le labyrinthe tester à un score qui se trouve dans le top 10. Il est automatiquement sauvegardé.

Une interface utilisateur est envisagée pour le projet, avec la possibilité de sélectionner une entrée et une sortie, de visualiser le processus de recherche.

De plus, les prérequis incluent une partie de la construction d'un labyrinthe automatique, ainsi que le développement d'un bot capable de se déplacer tout droit et de savoir dans qu’elle type de nœud il se trouve.

Ces compétences sont décrites dans le plan de formation, acquises lors des ateliers New Technologie et TPI.

## Organisation du suivi

|  |  |  |
| --- | --- | --- |
| Élève | Maître d’apprentissage | Experts |
| FREDDI Sam | Pascal HENAUER | Thomas TETART |
| / | / | Brian NYDEGGER |

## Livrables

* Rapport de projet
* Manuel utilisateur
* Résumé du rapport du TPI
* Journal de travail

## Matériel et outils nécessaires

|  |  |
| --- | --- |
| Matériel | Version / Model |
| Poste de travail (PC) avec un système d’exploitation Windows 10 Pro. | 10.0.19045 |
| Clé USB | 16 Go |
| SSD Samsung (Disque de travail) | 870 EVO - 250 Go |

|  |  |
| --- | --- |
| Outil | Version |
| Visual Studio Code | 1.82.0 |
| GitHub (Terminal) | 2.34.1 |
| Unity3D | 3.7.0 |
| Editor version | 2022.3.17f1 |
| Google Drive | *En ligne* |
| Word 2016 | 16.0 |
| Excel 2016 | 16.0 |

# Méthodologie de travail

Afin de réaliser ce travail de la manière la plus efficace possible j’ai choisi d’utiliser la méthode dite « en 6 étapes » qui se prête particulièrement bien aux travaux personnels comme le TPI.

Vous pourrez remarquer au long de la lecture que, dans certaines parties du document, il sera noté l’étape de la méthode correspondante.

Ci-après une explication simple de la méthode ainsi que chaque étape afin de comprendre la manière dont le travail a été réalisé.

## La méthode en 6 étapes

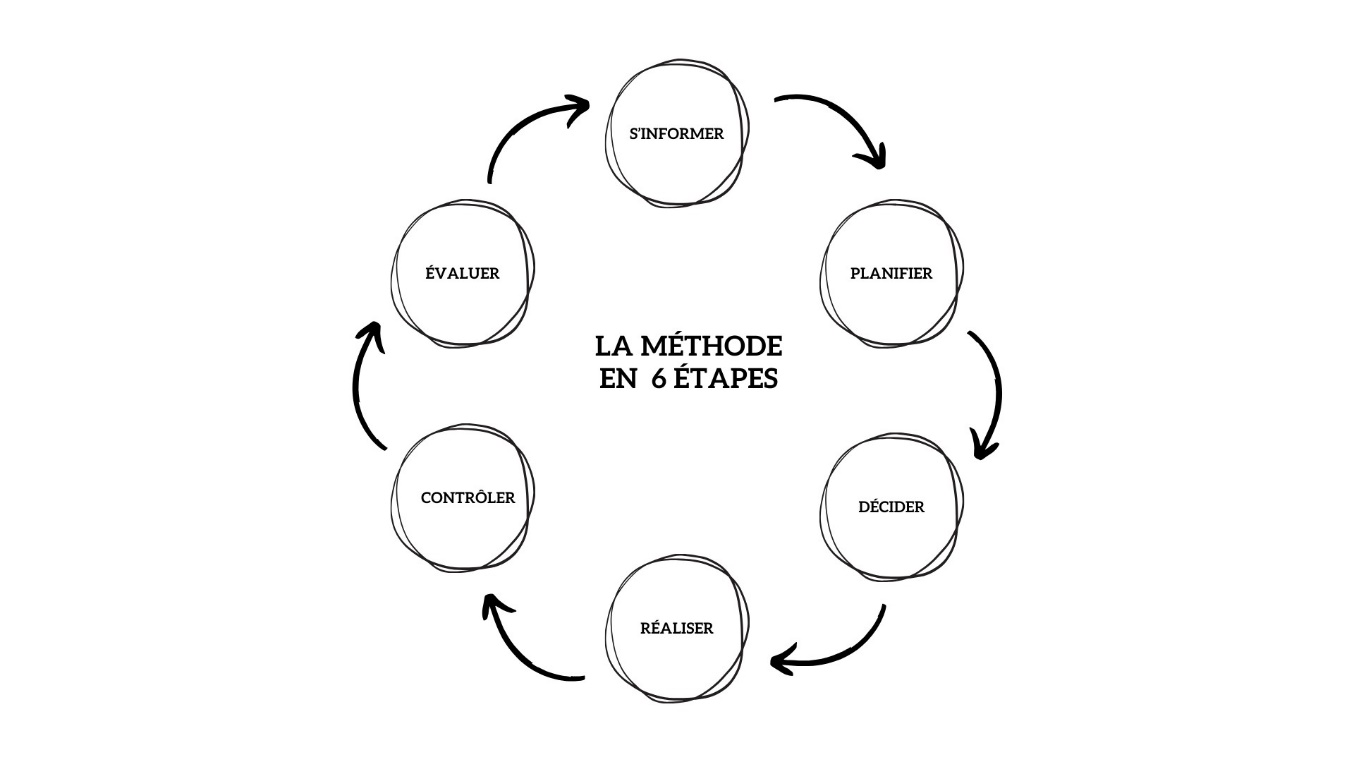
Cette méthode consiste à passer par plusieurs étapes pour réaliser un cahier des charges défini afin d’être le plus efficace possible.

Figure 1 : Schéma de la méthode en 6 étapes

Les étapes sont les suivantes :

### S’informer

Je dois comprendre et cerner exactement ce qu'on me demande avec toutes les contraintes qui me sont imposés. Dans le cas où il me manquerait une ou plusieurs informations pour la pleine compréhension de mon mandat, je vais chercher les informations et je vais m'informer afin que je sois en mesure de comprendre pleinement le travail demandé.

### Planifier

Il faut que je détermine les tâches nécessaires à la réalisation du mandat, que je les mette en ordre correctement en fonction de l'importance d'une tâche par rapport à une autre. Il faut également fixer la durée estimée de chaque tâche.

### Décider

Comment réaliser ce qui a été planifié. S'il y a plusieurs possibilités pour arriver au but du mandat, laquelle est la plus optimale ? C'est à cette étape que je dois prendre toutes les décisions de directions que le travail doit prendre pour arriver au but du mandat, le tout en respectant les contraintes imposées.

### Réaliser

Je dois me mettre à la tâche, réaliser ce qui est prévu dans ma planification en respectant les choix que j'ai fait lors de l'étape 3. Dans le cas où je dois faire autrement que ce qui a été prévu avant, avant de faire quoi que ce soit je dois me poser la question suivante : « Quelles sont les conséquences d’éventuelles dérogations à la planification ? »

### Contrôler

Contrôle du travail effectué, correspond aux tests à réaliser durant le travail. Le but est de contrôler que tout le travail effectué à l'étape 4 a correctement été fait et que le tout fonctionne parfaitement.

### Évaluer

On va chercher à savoir ce qui a bien fonctionné, ce qui pourrait être amélioré et dans le cas où il y a des choses qui ne fonctionnes pas, savoir pourquoi elles ne fonctionnent pas et ce qu'il faudrait faire et améliorer pour le faire fonctionner.

# Méthodologie de sauvegarde

Lors de la réalisation de ce travail il est de mon ressort de garantir les sauvegardes de l’avancement du projet, pour cela je vais utiliser la méthode du 3-2-1 qui garantit la sauvegarde du projet sur plusieurs supports.

## Méthode du 3-2-1

Cette méthode de sauvegarde est très simple même si son titre n’est pas très explicatif.

Pour faire simple, afin de respecter cette méthode il faut sélectionner 3 supports de sauvegarde différents.

Pour ma part les 3 supports seront les suivants : clé USB, cloud (Google Drive) et en local directement sur mon SSD de travail. J’ai sélectionné précisément ces 3 supports pour 1 seule raison, il peut arriver n’importe quoi lors de l’avancée du travail, j’aurais dans tous les cas un support contenant le projet et je minimise grandement les risques de perte d’avancement ou bien même de perte de projet.

# Planification

Dans le cadre du TPI, 2 planifications sont à effectuer. La planification prévisionnelle, qui est le reflet de la prévision des tâches, du temps passé sur chacune d’elles et les jalons. Puis, la planification effective qui est le reflet de comment s’est réellement déroulé le travail.

Ces 2 planifications sont importantes pour avoir une vue globale tout le long de l’avancement du projet pour savoir quand je suis en retard ou en avance sur le planning et pour pouvoir continuer le travail en conséquences. Mais également pour avoir une vue d’ensemble à la fin du projet pour en apprendre plus sur moi-même et pouvoir affiner mes planifications prévisionnelles futures.

Ci-dessous vous pouvez retrouver mes 2 planifications.

Cette partie correspond à l’étape 2 et 3 de la méthode en 6 étapes.

## Planification prévisionnelle

## Planification effective

[Ajouter l’image de ma planification effective]

# Analyse des fonctionnalités

**Générer une carte aléatoire** : Cette fonctionnalité permet à l'utilisateur de créer des labyrinthes avec un algorithme de génération aléatoire.

**Choisir de placer une entrée et une sortie** : Permet à l'utilisateur de personnaliser le labyrinthe en décidant où se trouvent l'entrée et la sortie.

**Voir le chemin le plus court entre l'entrée et la sortie** : Un algorithme de recherche entre une entrer et une sortie. Le programme trouve la sortie s’il en existe une où nous donne le chemin le plus proche.

**Tester une entrée et regarder toutes les sorties possibles** : Permet aux utilisateurs de tester une entrer différente et de percevoir toute les sorties possibles.

**Tester tout le labyrinthe et regarder le score obtenu** : Fournit une évaluation de la performance globale du labyrinthe, en prenant en compte des facteurs tels que le temps et la distance parcourue et le nombre de sortie cumulé par sortie.

**Afficher la liste des entrées et donner les informations suivant l’entrée :** Cette fonctionnalité fournit des détails sur chaque entrée du labyrinthe. Nombre de sortie possible, distance parcouru, score de l’entrée.

**Sauvegarder une carte avec un nom choisi** : Permet aux utilisateurs de sauvegarder leurs labyrinthes.

**Afficher la liste des cartes sauvegardées et du top 10** : Fournit un répertoire des cartes sauvegardées ainsi qu'un classement des meilleurs labyrinthes.

**Pouvoir charger une carte sauvegardée** : Permet aux utilisateurs de reprendre un labyrinthe intéressant.

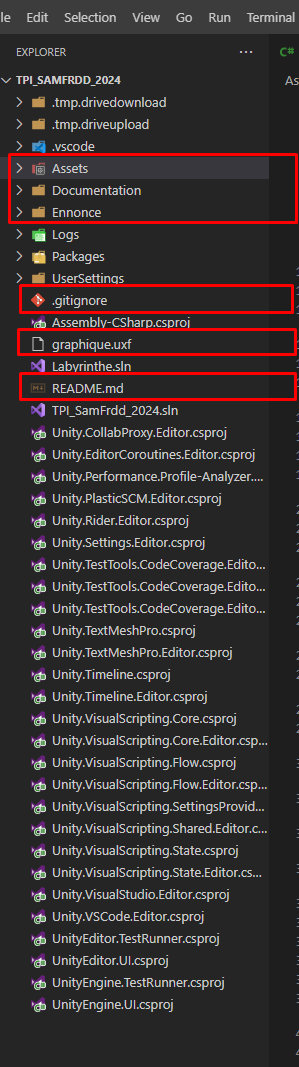
# Analyse organique

L’analyse organique parle du fonctionnement interne. C’est donc dans cette partie que nous allons aborder la partie technique de ce travail en parlant de la structure de ma base de données, de l’architecture et de l’arborescence de mon application et le fonctionnement de mon application en passant par les librairies utilisées et les échanges internes de données.

## 

## **Structure des dossiers**

Voici la structure du projet. Les dossiers et fichiers encadré sont ceux que je sauvegarde. Le reste sont des fichiers temporaires ou utilise à l’éditeur de Unity.



1. Dossier asset

Le dossier contient tous le contenu que j’utilise sur unity. Tous mes scripts y sont placés ainsi que les scènes, les UI, les préfab et les matériaux.

1. Documentation

Ce dossier contient tous ma documentation concernant le projet.

1. Enonce

Ce dossier contient l’énoncé du TPI ainsi que la grille d’évaluation.

1. Fichier. gitignore

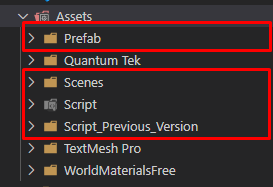
Le fichier « .gitignore » est un fichier listant les fichiers et répertoires qui sont ignorés lors du push et du commit.

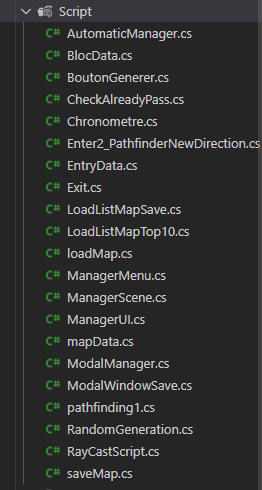
1. Fichier graphique.uxf

Le fichier « graphique.uxf » a été utilisé pour générer les diagrammes.

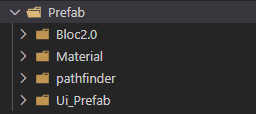
1. Fichier Readme.md

Ce fichier donne certaine indication sur la structure des fichiers.

Le dossier 'Assets' est organisé comme suit : il comprend un dossier 'Prefab', un dossier 'Quantum Tek' contenant des actifs récupérés depuis l'Unity Asset Store, un dossier scène englobant toutes mes scènes, un dossier 'Script' comprenant tous les scripts que j'ai créés pour ce projet, un dossier 'Script\_Previous\_Version' renfermant les scripts auxquels je n'ai pas touché depuis les prérequis, ainsi que 'TextMesh Pro', une extension d'Unity regroupant des actifs UI, et 'WorldMaterialsFree', également issu de l'Unity Asset Store.

  
  
Voici le dossier Script qui content toute mes classes que j’ai utilisées pour ce projet.

## 

Voici le dossier Prefab qui contient 4 dossiers

* Bloc2.0

Le dossier Bloc 2.0 contient les blocs du labyrinthe pour la génération aléatoire

* Meterial

Le dossier Material contient tous les matériaux du projet.

* Pathfinder

Le dossier Pathfinder contre la prefab du bot

* Ui\_Prefab

Le dossier Ui\_Prefab contient toute les prefab des Ui. Prefab des boutons etc…

## Architecture

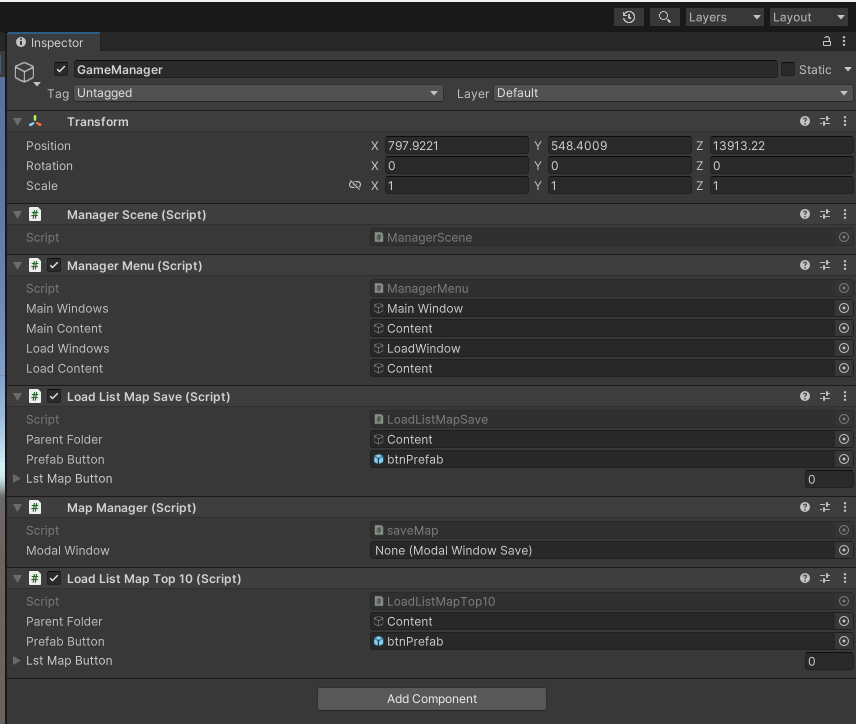
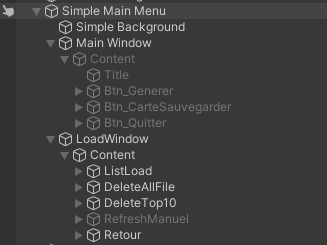
Pour réaliser ce projet Unity3D, j’ai choisi de suivre les méthodes d’Object que Unity nous donne à disposition. Pour l’écriture du code j’ai décidé de suivre la méthode DEMETER que nous avons appris lors de notre formation.

## Chemin des données

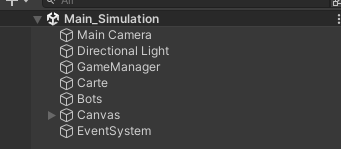
Structure des donnés dans les scènes Unity

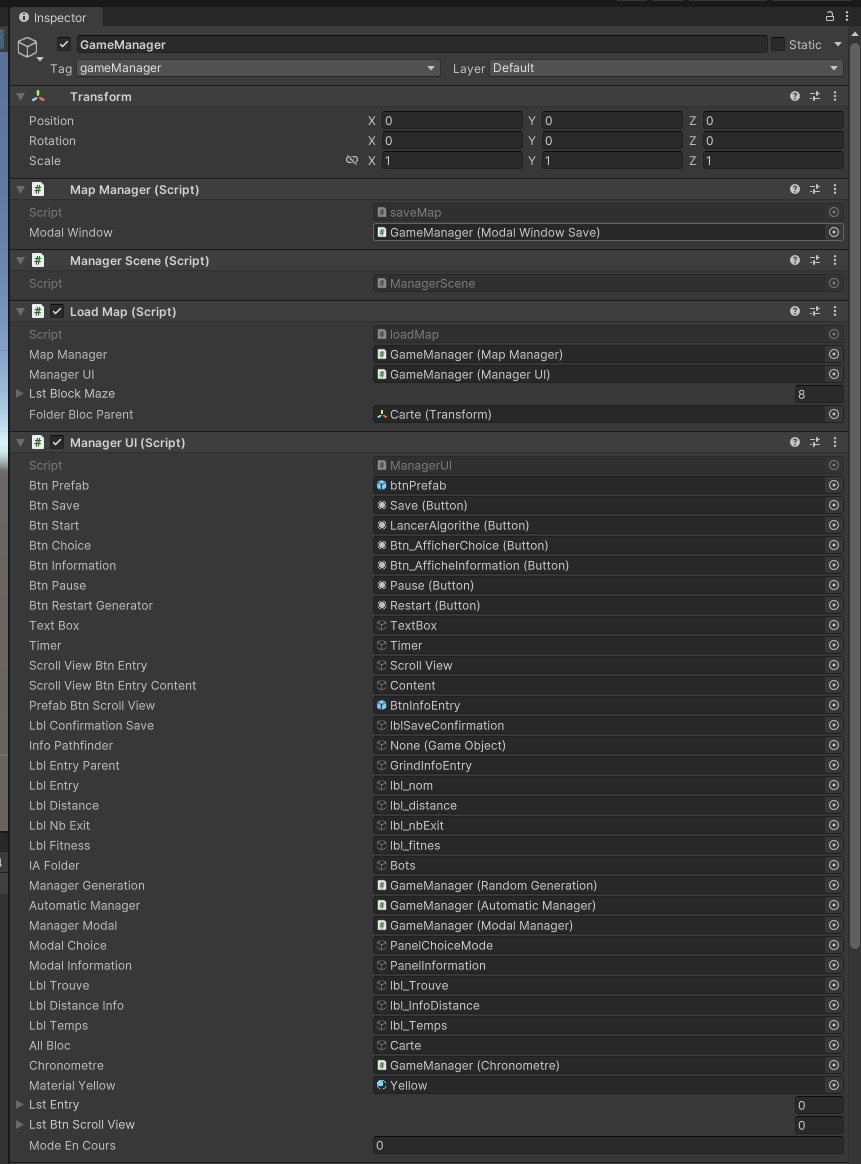
[A REMPLIR]

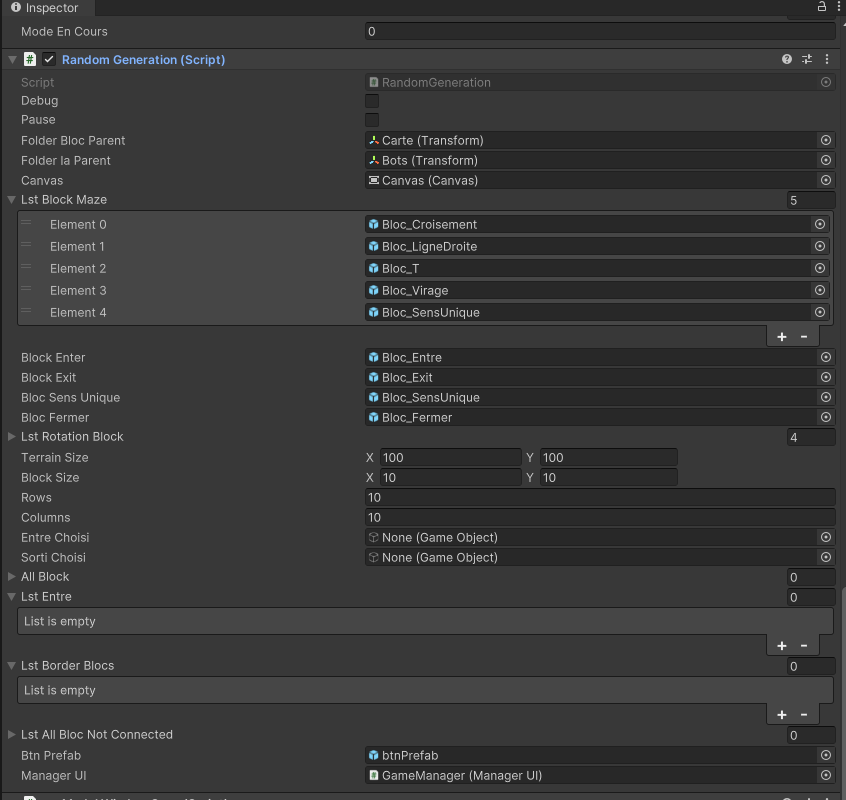


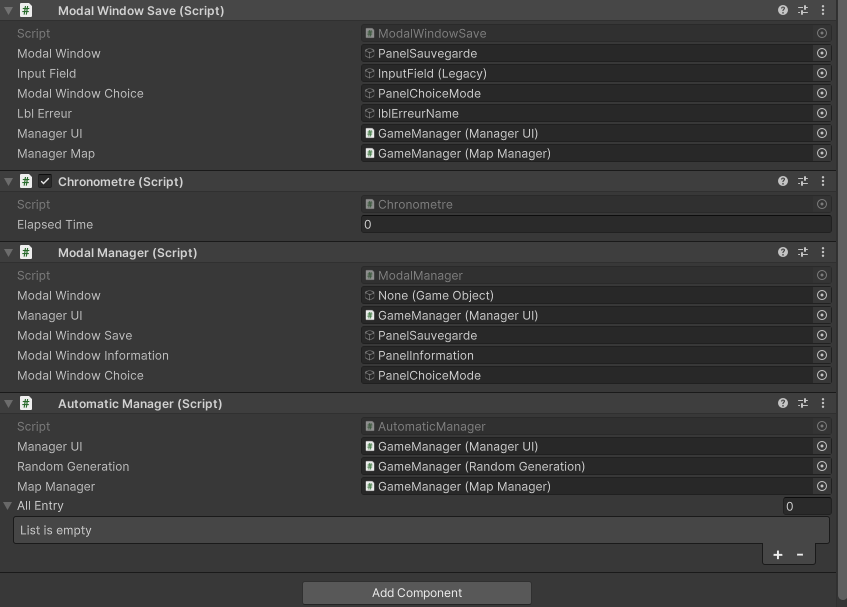
 

Main simulation









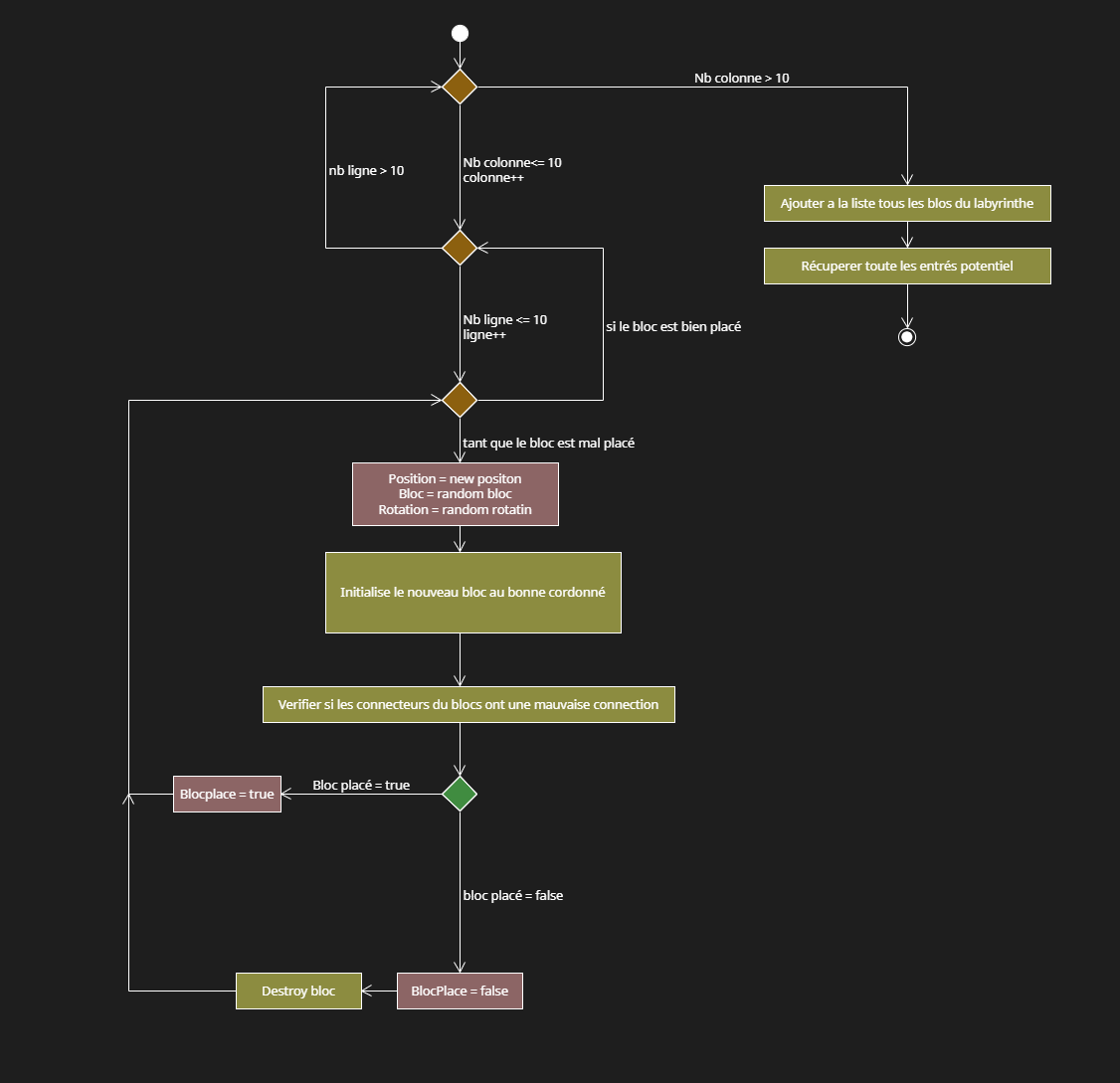
## Diagramme d’une méthode complexe

Cette partie est un bout de l’étape 4 de la méthode en 6 étapes.

### Diagramme sur la génération aléatoire

Blocs

Système de connecteur

Voici le diagramme simplifié de la génération du labyrinthe.

### Diagramme sur le bot

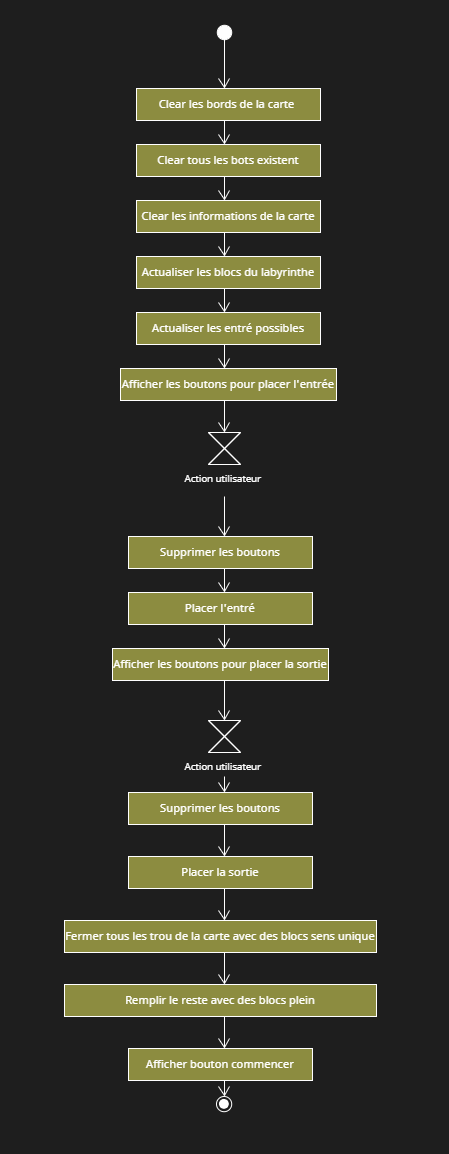
Système de rayCast

Machine d’état

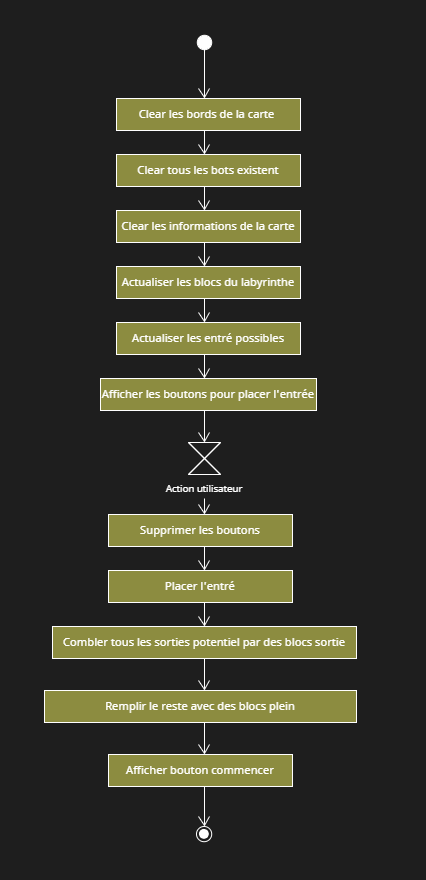
### Diagramme sur le mode 1

Voici le diagramme simplifié du mode manuel. 0

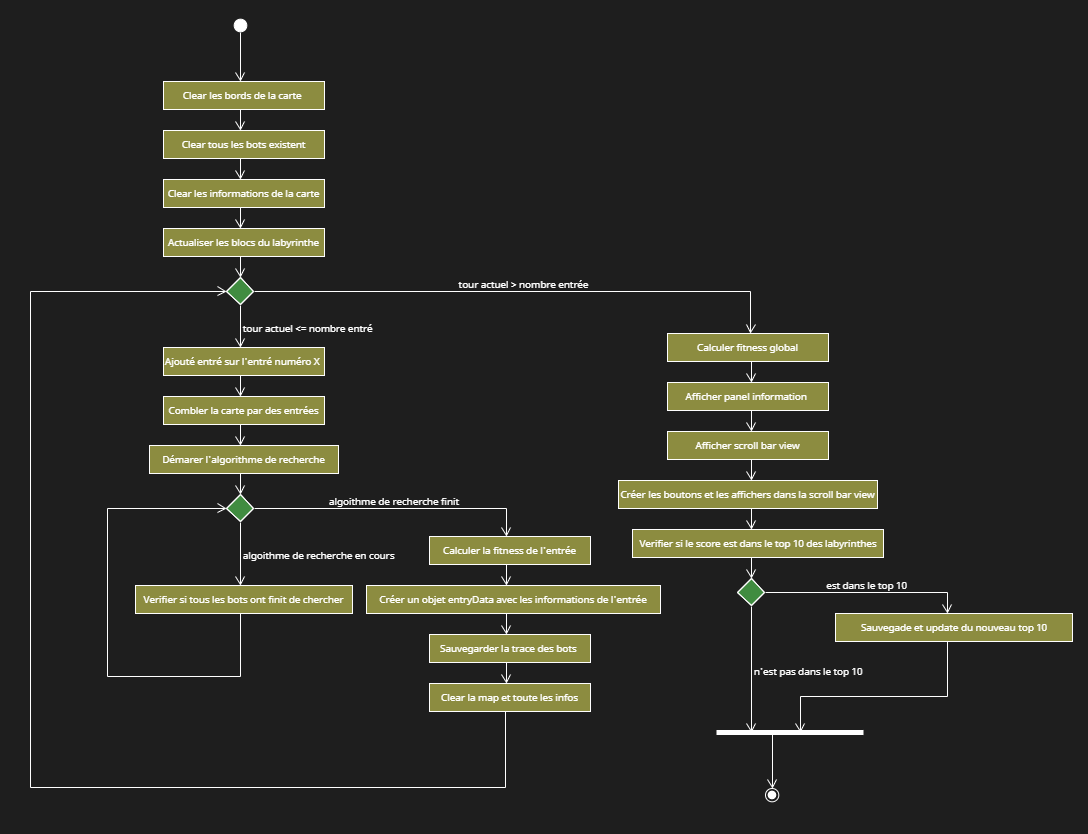
Ce programme s’exécute après l’exécution du mode manuel par l’utilisateur.



### Diagramme sur le mode 2



### Diagramme sur le mode 3



## Assets

# Les tests

Cette partie corresponds à l’étape 5 de la méthode en 6 étapes.

## Périmètre

Les objectifs du plan de test sont les suivants :

* Valider les fonctionnalités du cahier des charges
* Réaliser cette validation à la fin du cahier des charges

## Environnement

Les outils constituants mon environnement pour réaliser les différents tests sont les suivants :

* Windows 10 Pro.
* UnityEditor 2022.3.17f1
* Carte graphique X
* Processeur i7-8700
* 32 GO de ram

## Scénarios

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No.** | **User Story** | **Scénario** | **Données** | **Résultat attendu** |
| T001 | En tant qu’utilisateur je veux pouvoir lancer une génération aléatoire depuis le menu | 1 Depuis le menu l’utilisateur clic sur le bouton générer | Clic sur le bouton | L’application lance le début d’une génération de carte |
| T002 | En tant qu’utilisateur je veux pouvoir voir l’évolution de la génération de la carte | 1 L’utilisateur a cliqué lancé la génération du labyrinthe |  | L’application généré une carte aléatoirement en posant les blocs les un après les autres sans problèmes de connexion entre les routes |
| T003 | En tant qu’utilisateur je veux pouvoir placer une entré sur mon labyrinthe | 1 Quand l’utilisateur choisit le mode 1 il doit pouvoir placer une entré  2 L’utilisateur clique sur l’entre numéro 2 | Clic sur le bouton d’entré N.2 | Un bloc entré est correctement placé à l’endroit où l’utilisateur a cliquée |
| T004 | En tant qu’utilisateur je veux pouvoir placer une sortie sur mon labyrinthe | 1 Quand l’utilisateur choisit le mode 1 il doit pouvoir placer une entré  2 L’utilisateur clique sur la sortie numéro 4 | Clic sur le bouton d’entré N.4 | Un bloc sortie est correctement placé à l’endroit où l’utilisateur a cliquée et toute les autres sortie potentiel se retrouve bloqué avec un bloc sens unique |
| T005 | En tant qu’utilisateur je veux pouvoir voir le programme qui ferme totalement toute les sortie de mon labyrinthe avec les blocs adéquats au mode choisit | 1 Après que l’utilisateur ai placé une entré dans le mode 2  2 le programme complète automatiquement le labyrinthe |  | Toute les trou sur la carte sont bien comblé par des blocs a sens unique dans la bonne orientation |
| **Test No.** | **User Story** | **Scénario** | **Données** | **Résultat attendu** |
| T006 | En tant qu’utilisateur je veux pouvoir ouvrir un panel pour ouvrir le choix des mode a disposition | L’utilisateur ouvre un panel avec le bouton « afficher mode » après la génération | Clic sur le bouton affiche mode | Nous ouvre un panel avec a l’intérieur le choix des 3 modes |
| T007 | En tant qu’utilisateur je veux pouvoir lancer le mode numéro 3 et que le programme lance directement les entrés et sortie nécessaire au bon fonctionnement de ce mode | 1 L’utilisateur clique sur le bouton mode 3 après la génération | Clic sur le bouton mode 3 | L’application place l’entré au bloc numéro 1 et toute les autres possibilité seront des sorties |
| T008 | En tant qu’utilisateur je veux pouvoir voir le bot se dupliquer lors d’un croisement | L’utilisateur débute l’algorithme et nous voulons voir le bot qui se duplique en X fois lors de ce premier croisement | Clique sur le bouton commencer l’algorithme | Le bot « originel » reste au milieu et 2 « enfant / close » vont dans les X possibilité |
| T009 | En tant qu’utilisateur je veux pouvoir voir une trace bleu dernière le bot lorsqu’il effectue la recherche de la sortie | 1 L’utilisateur lance le mode 1 | Clique sur le mode 1 | Il y’a bien un trainé bleu lorsque le bot a toujours la possibilité de trouvé une sortie |
| T010 | En tant qu’utilisateur je veux pouvoir voir une trace verte derrière le bot qui a trouvé une sortie | 1 L’utilisateur lance le mode 1 | Clique sur le mode  Une carte avec une entrée et sortie possible1 | Il y’a bien la trainé en vert pour le bot qui a trouvé la sortie. La trainé est commencé du début jusqu’à la fin du parcours |
| T011 | En tant qu’utilisateur je veux pouvoir voir une trace rouge qui disparait au bout de 0,5 seconde lorsque mon bot rentre dans une section sens unique | 1 L’utilisateur lance le mode 1 | Clique sur le mode | Il y’a bien une trace rouge qui disparait au bout de 0,5 seconde lorsque un bot se rentre en collision avec un bloc a sens unique |
| T012 | En tant qu’utilisateur je veux pouvoir voir une trace jaune derrière le bot qui est le plus proche de la sortie | 1 L’utilisateur lance le mode 1 | Clique sur le mode  Une carte avec une entré et une sortie pas possible | On s’attend a voir une trace jaune derrière le bot qui est le plus proche de la sortie donné. Le plus proche en terme de distance |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No.** | **User Story** | **Scénario** | **Données** | **Résultat attendu** |
| T013 | En tant qu’utilisateur je veux pouvoir voir si le bot ne trouve pas de sortie | 1 L’utilisateur clique sur le bouton mode 1 après la génération | Clic sur le bouton mode 1 | L’application m’informe sous forme de texte que aucune sortie n’a été trouvé avec l’entré la sortie choisi |
| T014 | En tant qu’utilisateur je veux pouvoir voir dans une liste toute les carte que j’ai pu sauvegarder | 1 L’utilisateur clique depuis le menu sur le bouton « charger une carte sauvegarder » | Clique sur le bouton | Nous retourne une liste de bouton cliquable qui représente chaque carte que nous avons sauvegarder |
| T015 | En tant qu’utilisateur je veux pouvoir avoir la possibilité de sauvegarder un labyrinthe que j’apprécie avec un nom que je choisi | L’utilisateur lance une génération aléatoire et clique sur le bouton sauvegarder | Clique sur le bouton sauvegarder et rentre la valeur « Test » comme nom de carte | Nous ferme la fenêtre modal de sauvegarde et nous affiche un message de succès. Nous sauvegarde dans fichier Test.xml dans l’arborescence de fichier |
| T016 | En tant qu’utilisateur je veux pouvoir pouvoir ne pas choisir de nom a la carte que je sauvegarde | L’utilisateur lance une génération aléatoire et clique sur le bouton sauvegarder | Confirme la sauvegarde sans mettre de nom | Nous ferme la fenêtre modal de sauvegarde et nous affiche un message de succès. Nous sauvegarde dans fichier NomUnique.xml dans l’arborescence de fichier |
| T017 | En tant qu’utilisateur je veux pouvoir recommencer une génération d’un nouveau labyrinthe juste après avoir terminé une génération précédente | L’utilisateur clique sur le bouton, Relancer après la génération automatique de la carte | Clique sur le bouton relancer | Nous voulons le début d’une nouvelle génération. L’affichage est remis a 0 et l’ancienne carte est supprimé de la scène |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No.** | **User Story** | **Scénario** | **Données** | **Résultat attendu** |
| T018 | En tant qu’utilisateur je veux pouvoir lancer l’algorithme du mode 1 et voir mon chemin jusqu’à la sortie | L’utilisateur clique sur le mode 1 après une génération aléatoire et choisi une entré et une sortie. | Bouton mode 1  Choisi entré n.2  Choisi entré n.5 | Le bot commence et effectue correctement l’algorithme de recherche. Le programme nous retourne toute les donner relative à l’algorithme lancé. |
| T019 | En tant qu’utilisateur je veux pouvoir lancer le mode 2 et regarder toute les sorties possible avec l’entrer que je choisi | L’utilisateur clique sur le mode 2 après une génération aléatoire et choisi une entré | Bouton mode 1  Choisi entré n.2 | Le bot commence et effectue correctement l’algorithme de recherche. Le programme nous retourne toute les donner relative à l’algorithme lancé. |
| T020 | En tant qu’utilisateur je veux pouvoir voir la fitness de mon labyrinthe en cliquant sur le mode 3 | L’utilisateur clique sur le mode 3. |  | Chaque entré est tester 1 par 1. Et a la fin le programme nous retourne un tableau avec chaque entré et son fitness classé par ordre décroissant |
| T021 | En tant qu’utilisateur je veux pouvoir cliqué sur une des entré dans le tableau et voir son parcours jusqu’à chaque sortie | L’utilisateur attend que l’algorithme du mode 3 se finisse pour ensuite cliquer sur la première ligne du tableau retourné | Clique sur une des entré présenté dans a la fin du mode 3 | Affiche le trace exact depuis l’entré choisi jusqu’à chaque sorti possible |
| T022 | En tant qu’utilisateur je veux pouvoir voir les 10 meilleurs labyrinthe stocké sur mon application en fonctions de leur fitness | L’utilisateur clique sur le bouton « carte sauvegarder » depuis le menu |  | Nous affiche les 10 meilleurs carte sauvegarder automatiquement en fonction de leur fitness |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No.** | **User Story** | **Scénario** | **Données** | **Résultat attendu** |
| T023 | En tant qu’utilisateur je veux pouvoir quitter l’application depuis le bouton quitter du menu | L’utilisateur clique sur le bouton quitter depuis le menu | Clique sur le bouton menu | L’application se ferme directement |
| T024 | En tant qu’utilisateur je veux pouvoir supprimer toute mes carte sauvegarder | L’utilisateur clique sur le bouton « Supprimer » en dessous de la liste de carte que nous avons sauvegarder | Clique sur le bouton supprimer correspondant | Nous supprime toute les cartes que nous avons sauvegarder et l’affichage est update automatiquement |

## Suivi évolutif des tests

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test No.** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** |
| T001 | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T002 | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T003 | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T004 | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T005 | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T006 | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T007 | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T008 | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T009 | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T010 | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T011 | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T012 | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T013 | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T014 | ❌ | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ |  |  |
| T015 | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ |  |  |
| T016 | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ |  |  |
| T017 | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ |  |  |
| T018 | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T019 | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T020 | ❌ | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ |  |  |
| T021 | ❌ | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ |  |  |
| T022 | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ |  |  |
| T023 | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ | ✅ | ✅ |  |  |
| T024 | ❌ | ❌ | ❌ | ❌ | ❌ | ✅ | ✅ | ✅ | ✅ |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |

## Rapport de test

[Les tests qui n’ont jamais fonctionnés]

[Explication des faits importants liés au TPI]

# Conclusion

Cette partie corresponds à l’étape 6 de la méthode en 6 étapes.

## Difficultés rencontrés

[A REMPLIR]

## Amélioration possibles

[A REMPLIR]

## Bilan personnel

[A REMPLIR]

## Remerciements

[A REMPLIR]

# Bibliographie

Source d’information méthode en 6 étapes : <https://www.afci-ju.ch/fichiers/Planification-par-la-mthode-des-6-tapes-15.pdf>

Logo de PHP : <https://fr.m.wikipedia.org/wiki/Fichier:PHP-logo.svg>

Logo de Bootstrap studio : <https://en.wikipedia.org/wiki/File:Bootstrap_Studio_Logo.png>

Logo de DBeaver : <https://www.crunchbase.com/organization/dbeaver/technology>

Logo de GitLab : <https://logowik.com/gitlab-vector-logo-4667.html>

Logo d’apache : <https://commons.wikimedia.org/wiki/File:Apache_HTTP_server_logo_%282019-present%29.svg>

Logo de MariaDB : <https://linuxnatives.net/2015/10-reasons-to-migrate-to-mariadb-if-still-using-mysql>

# Glossaire

|  |  |
| --- | --- |
| CRUD | Acronyme anglais pour « Create, Read, Update and Delete ». Il s’agit des actions les plus fréquentes réalisées sur une base de données. |
| User Story | Il s’agit d’un élément issu de du développement agile. Le but d’une « User Story » est de décrire une fonctionnalité sous la forme d’une petite phrase vue par l’utilisateur/client. |
| MVC | Il s’agit d’un modèle qui permet de séparer le plus possibles les différentes parties du code, qui notamment utilisé dans les applications web. « MVC » est l’acronyme de « Model, Vue, Controller ». |

# Annexes

L’ensemble des livrables définis au début de ce document se trouvent dans le même répertoire.