كلية العلوم السملالية - مراكش
FACULTÉ DES SCIENCES SEMLALIA - MARRAKECH

Université Cadi Ayyad
Faculté des Sciences Semlalia
Départements d'Informatique

**projet de fin d'études**

pour l'obtention de la

Licence Fondamentale en Science Mathématiques et Informatique (SMI)

# Conception et devlopement d'un jeu 2D avec pygame

Projet effectue du 8 Mai 2023 au 26 juillet 2023

Réalisé par :

**BOUGAYOU ZAKARIA && OUAHIB SALMA**

Encadre par :

**ERRAJI LATIFA** Professeur, Département informatique FSSM Marrakech

Soutenu le 26 juin 2023 devane la commission d' examen :

**ERRAJI LATIFA** Professeur, Département informatique FSSM Marrakech

.....

Année Universitaire: 2022 / 2023

# Dédicace :

*Au Dieu tout-puissant notre créateur*

*À nos chères familles*
*Pour leurs sacrifices sans limites*

*À nos chers enseignants*
*Pour leurs soutiens et encouragements*

*À nos chères encadrantes*
*Pour leur soutien et consignes*

*À nos amis,*
*Pour leurs aides et leurs points de vue*

# Remerciements

Nous laissons notre plume écrire pour remercier le Dieu qui nous a laissé vivre jusqu'à ce moment, et nous a aidé durant notre année d'étude, pour dire un grand merci pour nos chers parents, qui lèvent les mains à chaque prière pour nous souhaiter la réussite et le bonheur, qui ont apporté la torche du sacrifice pour éclaircir notre chemin, qui ont fortifié notre volonté.

Aussi, nous adressons à nos chers professeurs que nous rencontrons dès notre première pas à la faculté des sciences Semlalia, ceux qui sacrifient le temps pour nous aider, et nous donner des conseils, dans notre vie professionnelle, et nous aider à développer nos compétences. Un grand respect de notre part à vous.

Ainsi, nous remercions très chaleureusement notre encadrante Mme **ERRAJI LATIFA** , un grand Merci, et un grand respect, pour nous est encadré durant 2 mois avec patience, et pour nous est aidé de réaliser ce projet.

Enfin, sincèrement, nous dédions notre profond respect envers tous nos amis, a tous les étudiants de notre filière Smi, nous avons passé des bons moments, nous souhaitons une bonne continuation a tous et a toutes .

# Table de matières

# Table des figures :

# Résumé

Dans le cadre de notre formation de licence fondamentale, nous sommes amenés à réaliser un projet qui sera le projet de fin d'étude.

Ce projet est la conception et le développement d'un jeu vidéo 2D.

La conception d'un jeu vidéo est une phase cruciale où une idée initiale est transformée en un concept de jeu concret. Cela implique la création d'une histoire captivante, de personnages mémorables, d'univers visuellement attrayants et de mécanismes de jeu engageants. Cette phase de préproduction est essentielle pour définir la vision globale du jeu et poser les bases solides de son développement ultérieur.

Une fois que la conception est établie, le développement entre en jeu.
Cette étape comprend la programmation du code, la création d'éléments graphiques, l'élaboration de bandes sonores et la réalisation de tests rigoureux pour s'assurer du bon fonctionnement du jeu et de son attractivité pour les joueurs.

# INTRODUCTION GÉNÉRALE :

# INTRODUCTION GÉNÉRALE :

Les jeux vidéo ont connu une évolution spectaculaire depuis leur création. Depuis les premiers jeux simples et pixelisés des années 1970 jusqu'aux graphismes réalistes et aux mondes ouverts d'aujourd'hui, l'industrie du jeu vidéo est devenue l'une des formes de divertissement les plus populaires et les plus influentes au monde.

Les jeux vidéo sont des programmes informatiques interactifs qui permettent aux joueurs de vivre des expériences virtuelles dans des mondes fantastiques ou réalistes. Ils offrent une vaste gamme de genres, allant des jeux d'action et d'aventure aux jeux de rôle, en passant par les jeux de sport, les jeux de tir et bien d'autres encore.

L'histoire des jeux vidéo est marquée par des avancées technologiques majeures, telles que l'introduction des consoles de jeux, des ordinateurs personnels et des appareils mobiles. Les consoles de jeux, comme la PlayStation, la Xbox et la Nintendo Switch, offrent des expériences de jeu hautement spécialisées et dédiées, tandis que les ordinateurs personnels permettent une plus grande personnalisation et une plus grande variété de jeux. Les appareils mobiles, tels que les smartphones et les tablettes, ont également révolutionné l'industrie en offrant des jeux accessibles à tout moment et en tout lieu.

Les jeux vidéo ont également évolué du point de vue du gameplay et de la narration. Les jeux d'aujourd'hui offrent souvent des mondes ouverts vastes et détaillés, des mécaniques de jeu complexes et des histoires riches et captivantes. Les joueurs peuvent interagir avec des personnages non jouables, prendre des décisions qui influencent l'intrigue et même participer à des expériences multijoueurs en ligne, où ils peuvent collaborer avec des joueurs du monde entier.

Les jeux vidéo ont également un impact culturel et social important. Ils sont devenus un moyen de divertissement populaire, rassemblant des millions de joueurs à travers le monde. Les jeux vidéo sont également utilisés dans des domaines tels que l'éducation, la formation professionnelle et la recherche scientifique.

Malgré leur popularité croissante, les jeux vidéo ont également suscité des débats sur leur impact sur la santé mentale, la violence et l'addiction. Il est important de trouver un équilibre entre le jeu vidéo et d'autres aspects de la vie quotidienne pour en profiter de manière saine et équilibrée.

En conclusion, les jeux vidéo sont devenus une forme de divertissement omniprésente et influente dans notre société moderne. Ils offrent des expériences interactives et immersives, repoussent les limites technologiques et culturelles, et continuent d'évoluer pour offrir de nouvelles possibilités aux joueurs du monde entier.

Dans notre rapport, nous allons présenter les différentes étapes de notre réalisation de projet en commençant par notre cahier de charge et l'étude de ce dernier comme notre **1er chapitre**.

Puis, on attaque le **2e chapitre**, qui concerne la conception et l'analyse, c'est un chapitre qui contient l'ensemble des digrammes qui nous aide à connaitre le scenario de jeu, et les différents composants de notre jeu.

Enfin, le **3e chapitre** qui se déroule sur les étapes pour utiliser l'outil tiled, l'ensemble des images construite par cet outil, ainsi que la première photo de notre jeu.

# I. CHAPITRE 1: IDENTIFICATION DES BESOINS

# I.CHAPITRE 1:

## I.1.Introduction:

Ce chapitre présent l'identification des besoins d'utilisateur, en premier lieu nous allons définir le cahier de charge. Ensuite nous passons à l'identification des acteurs qui interagiront avec le système et l'identification des différents cas d'utilisation

## I.2.Le cahier de charge :

### I.2.1. Le but de projet:

L'objectif de ce projet est la conception et la réalisation d'un jeu Mario 2D.

Le but du jeu Mario est de contrôler le personnage de Mario à travers une série de niveaux, en évitant les ennemis, en surmontant les obstacles et en collectant des power-ups. L'objectif principal est d'atteindre la fin de chaque niveau en toute sécurité et d'accumuler le plus de points possible. terminer les niveaux dans un délai imparti ou explorer des mondes ouverts.

Le plaisir réside dans l'exploration, la résolution de puzzles et l'amélioration de vos compétences de jeu pour atteindre des objectifs de plus en plus difficiles.

## I.3. Les besoins fonctionnels :

Voici quelques-uns des besoins fonctionnels:

1. Contrôles du personnage : Les joueurs doivent être en mesure de contrôler le personnage principal, Mario, en utilisant des commandes telles que sauter, courir, se déplacer vers la gauche ou la droite, etc. Les contrôles doivent être intuitifs et réactifs pour permettre aux joueurs de naviguer dans le monde du jeu avec précision.

2. Niveaux de jeu : Le jeu Mario est généralement composé de plusieurs niveaux, chacun avec des défis et des obstacles uniques. Les besoins fonctionnels incluent la création de ces niveaux, le placement des ennemis, des plateformes, des bonus, et la conception de la

progression de difficulté au fur et à mesure que les joueurs avancent.

3. Interaction avec les éléments du jeu : Les joueurs doivent pouvoir interagir avec les éléments du jeu, tels que les blocs à frapper pour obtenir des pièces ou des power-ups, les tuyaux pour accéder à des zones secrètes, les ennemis à sauter dessus pour les vaincre, etc.

4. Collecte d'objets : Les jeux Mario comportent souvent des objets à collecter, tels que les pièces de monnaie, les champignons pour obtenir des power-ups, , etc. Les besoins fonctionnels incluent la gestion du ramassage de ces objets, leur affichage sur l'interface utilisateur et leurs effets sur le personnage du joueur.

5. Mécanismes de progression : Les jeux Mario ont souvent des mécanismes de progression pour permettre aux joueurs d'avancer dans le jeu. Cela peut inclure des portes, des échelles, des escaliers, des ascenseurs, etc. Les joueurs doivent être en mesure de comprendre et d'utiliser ces mécanismes pour progresser dans le jeu.

6. Système de points et de classement : Les jeux Mario incluent généralement un système de points qui récompense les joueurs pour leurs performances, tels que le nombre de pièces collectées, les ennemis vaincus, le temps écoulé, etc. Certains jeux Mario peuvent également inclure des classements pour permettre aux joueurs de se comparer à d'autres joueurs.

7. Gestion des vies du joueur : Les jeux Mario utilisent souvent un système de vies où les joueurs commencent avec un certain nombre de vies et en perdent une lorsqu'ils sont touchés par un ennemi ou qu'ils tombent dans un précipice. Les besoins fonctionnels incluent la gestion de ces vies, la détection des collisions avec les ennemis et les obstacles, et la gestion des éventuelles vies supplémentaires.

## I.4. Les besoins techniques :

Voici quelques-uns des besoins techniques  dans le jeux Mario :

1. Graphismes : Les jeux Mario nécessitent des éléments graphiques pour représenter les personnages, les niveaux, les objets et les effets visuels. Les besoins techniques comprennent la modélisation et l'animation des personnages, la création de décors et de paysages, l'utilisation de textures, d'effets spéciaux, etc.

2. Moteur de jeu : Les jeux Mario nécessitent un moteur de jeu pour gérer les interactions, les mouvements, les collisions, les effets physiques, la caméra, etc. Le moteur de jeu doit être capable de gérer les contrôles du joueur, de détecter les collisions avec les ennemis et les objets, de gérer les mouvements du personnage principal et de rendre le monde du jeu de manière fluide.

3. Son : Les jeux Mario utilisent des effets sonores et de la musique pour créer une expérience audio immersive. Les besoins techniques incluent l'intégration de différents sons pour les mouvements, les sauts, les collisions, les dialogues, les musiques de fond, etc.

4. Multijoueur (le cas échéant) : Certains jeux Mario peuvent offrir des fonctionnalités multijoueurs, où plusieurs joueurs peuvent jouer ensemble en coopération ou en compétition. Les besoins techniques incluent la mise en place d'une connectivité réseau, la synchronisation des actions des joueurs, la gestion des lobbies, etc.

6. Optimisation des performances : Les jeux Mario doivent être optimisés pour garantir des performances fluides, en particulier en ce qui concerne les graphismes et la fréquence d'images. Cela implique l'utilisation efficace des ressources matérielles disponibles, la gestion de la mémoire, la réduction des temps de chargement, l'optimisation des algorithmes, etc.

7. Support de plateforme : Les jeux Mario peuvent être développés pour différentes plateformes, telles que les consoles de jeux, les ordinateurs personnels, les appareils mobiles, etc. Les besoins techniques incluent la prise en charge des fonctionnalités spécifiques à chaque plateforme, l'adaptation des contrôles, l'optimisation des

performances en fonction des spécifications matérielles, etc.

# I.5. Le diagramme de cas d'utilisation :

## I.5.1. Le diagramme de cas d'utilisation simple :

Un cas d'utilisation (Use case) « représente un ensemble de séquences d'actions réalisées par le système et produisant un résultat observable intéressant pour un acteur particulier »

### I.5.1.1. Identification d'acteurs :

- **Joueur**

L'acteur "joueur" dans le contexte du jeu Mario représente la personne qui interagit avec le jeu en tant que joueur. L'acteur joueur est celui qui contrôle le personnage principal, Mario, et participe activement à l'expérience de jeu

- **Ennemie**

L'acteur "ennemi" dans le jeu Mario représente les personnages ou les entités hostiles avec lesquelles le joueur, contrôlant Mario, doit interagir. Les ennemis sont des obstacles pour le joueur et peuvent prendre différentes formes, comportements et niveaux de difficulté tout au long du jeu.

L'objectif principal des ennemis est de rendre le jeu plus difficile pour le joueur en essayant de causer des dégâts à Mario. Si Mario entre en contact avec un ennemi, il peut perdre une vie, être blessé ou subir d'autres effets négatifs, selon les règles du jeu.

### I.5.1.2. Le diagramme de cas d'utilisation :

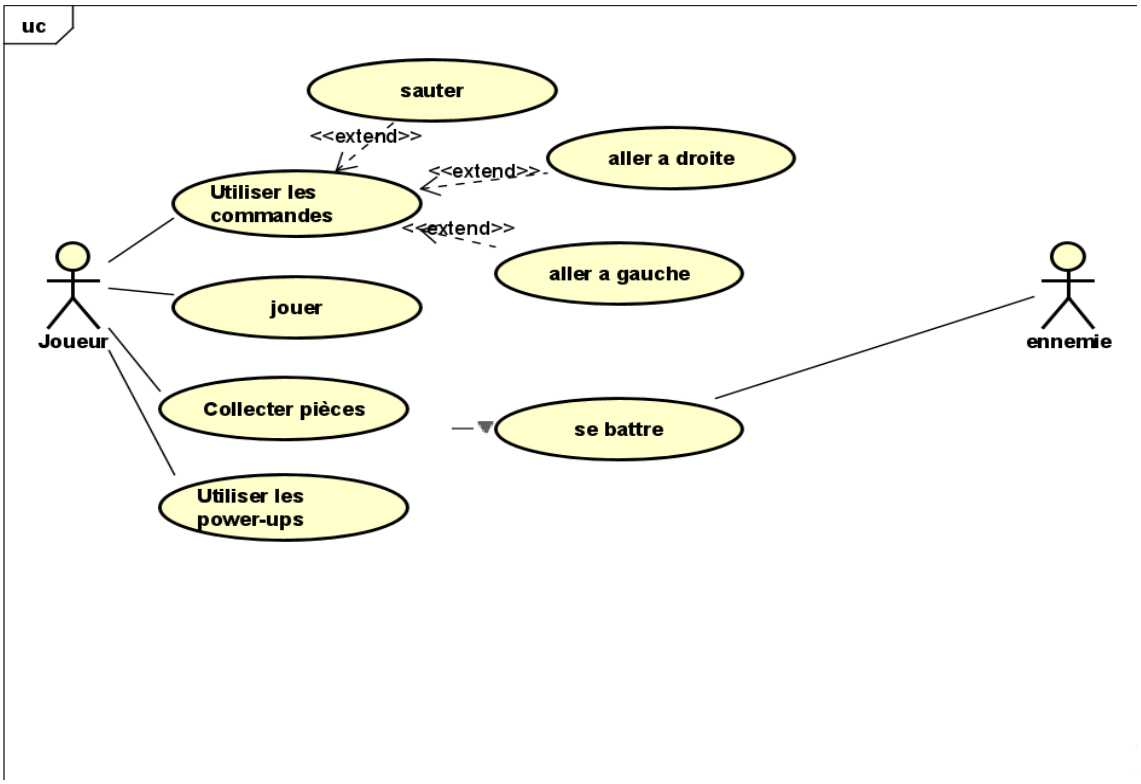



*Figure 1 Diagramme de cas d'utilisation :*

### I.5.1.3. Description textuelle des cas d'utilisation :

1. Utiliser les commandes : Le joueur peut utiliser les commandes du jeu, telles que les touches du clavier ou les boutons d'une manette de jeu, pour contrôler les actions de Mario, telles que sauter, courir, se déplacer à gauche ou à droite, etc.
1. Jouer : Le joueur peut jouer au jeu en contrôlant Mario à travers les niveaux, en surmontant les obstacles, en collectant des pièces et en atteignant l'objectif du niveau.
2. Collecter pièces : Le joueur peut collecter des pièces de monnaie réparties dans les niveaux. La collecte de pièces peut être un objectif en soi ou peut contribuer à obtenir des récompenses supplémentaires.
3. Attaquer les ennemis : Le joueur peut permettre à Mario d'attaquer les ennemis en sautant dessus ou en utilisant des power-ups. L'attaque des ennemis permet de les vaincre et de progresser dans le jeu.
4. Utiliser les power-ups : Le joueur peut utiliser les power-ups, tels que le champignon

qui agrandit Mario ou l'étoile d'invincibilité, pour obtenir des capacités spéciales temporaires et faciliter la progression dans le jeu.

## I.5.2. Le diagramme de cas d'utilisation Business :

Le diagramme de cas d'utilisation Business est un type spécifique de diagramme de cas d'utilisation qui se concentre sur les interactions entre les acteurs et le système d'un point de vue plus orienté vers les processus métier et les activités commerciales. Dans le contexte du jeu Mario, un diagramme de cas d'utilisation Business pourrait illustrer les interactions entre les acteurs et le système liées aux aspects commerciaux du jeu.

### I.5.2.1. Le diagramme de cas d'utilisation Business :

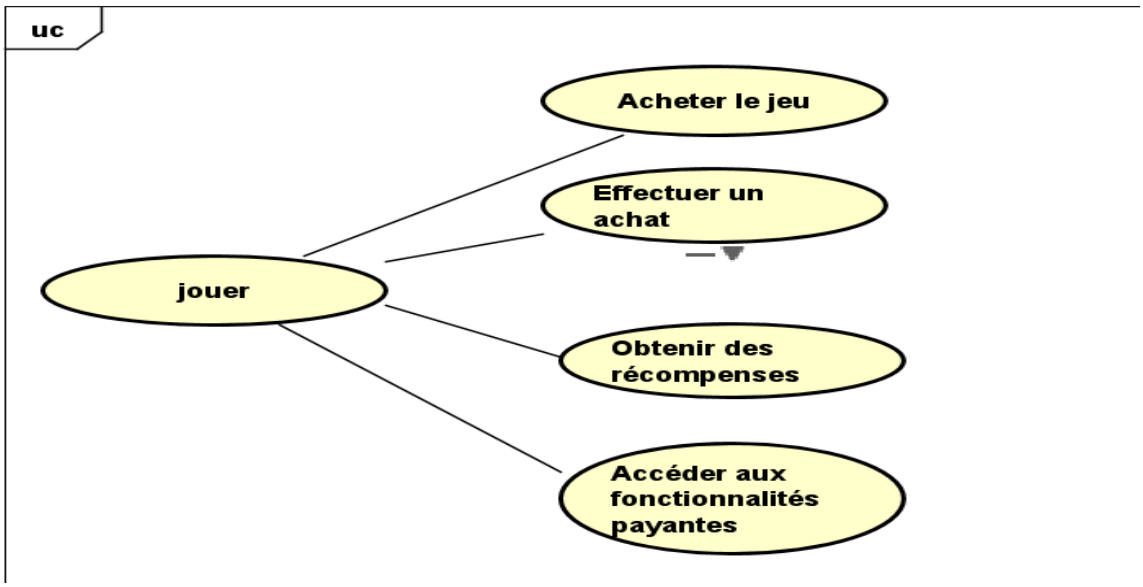



*Figure 2 : Diagramme de cas d'utilisation Business*

### I.5.2.2. Description textuelle des cas d'utilisation :

Dans ce diagramme de cas d'utilisation Business, l'acteur principal est le "Joueur", représentant la personne qui joue au jeu Mario. Voici une description des cas d'utilisation associés :

1. Acheter le jeu : Le joueur peut acheter le jeu Mario, que ce soit en ligne, dans un magasin ou par d'autres moyens de distribution. Cela implique généralement un processus de transaction commerciale.

2. Effectuer un achat : Une fois que le joueur a le jeu, il peut effectuer des achats supplémentaires, tels que des DLC (contenu téléchargeable) ou des extensions, qui peuvent offrir de nouveaux niveaux, personnages ou fonctionnalités supplémentaires.

3. Obtenir des récompenses : Le joueur peut obtenir des récompenses ou des avantages en effectuant des actions spécifiques dans le jeu, en atteignant certains objectifs ou en accomplissant des succès. Ces récompenses peuvent inclure des personnages supplémentaires, des costumes, des pouvoirs spéciaux, etc.

4. Accéder aux fonctionnalités payantes : Certains aspects du jeu peuvent nécessiter des fonctionnalités payantes, telles que l'accès à des fonctionnalités en ligne, des classements mondiaux, des tournois, etc. Le joueur peut utiliser ces fonctionnalités supplémentaires en payant des frais ou en souscrivant à un abonnement.

Ce diagramme de cas d'utilisation Business met l'accent sur les interactions commerciales entre le joueur et le système de jeu Mario. Il illustre les différentes activités liées aux achats, aux récompenses et aux fonctionnalités payantes qui peuvent enrichir l'expérience de jeu du joueur.

## I.6.Conclusion :

Dans ce chapitre, nous avons documenté les besoins du jeu Mario dans le cahier des charges. Puis, nous avons élaboré le diagramme de cas d'utilisation et Le diagramme de cas d'utilisation Business, pour chaque cas d'utilisation, nous avons fourni une description textuelle.

# II. chapitre 2:

## Conception et modélisation

# II. chapitre 2:

## II.1.Introduction :

la phase de Conception et modélisation dans l'UML est une étape cruciale pour traduire les besoins du système en une conception cohérente et compréhensible. Elle permet aux concepteurs et aux développeurs de visualiser et de communiquer efficacement la structure, le comportement et les interactions du système, facilitant ainsi le développement et la mise en œuvre ultérieure du jeu Mario.

## II.2. Diagramme d'états transition:

Le diagramme d'états-transition (State Transition Diagram en anglais) est l'un des diagrammes UML les plus couramment utilisés pour modéliser le comportement d'un système. Il représente les différents états d'un objet ou d'un système, ainsi que les transitions entre ces états en réponse à des événements

Le diagramme d'états-transition est utilisé pour modéliser le comportement réactif d'un système, en mettant l'accent sur les changements d'états et les transitions qui se produisent en réponse à des événements. Il permet de visualiser et de comprendre le flux de contrôle et le cycle de vie d'un objet ou d'un système.

### II.2.1 Diagramme d'états transition pour le Hero :

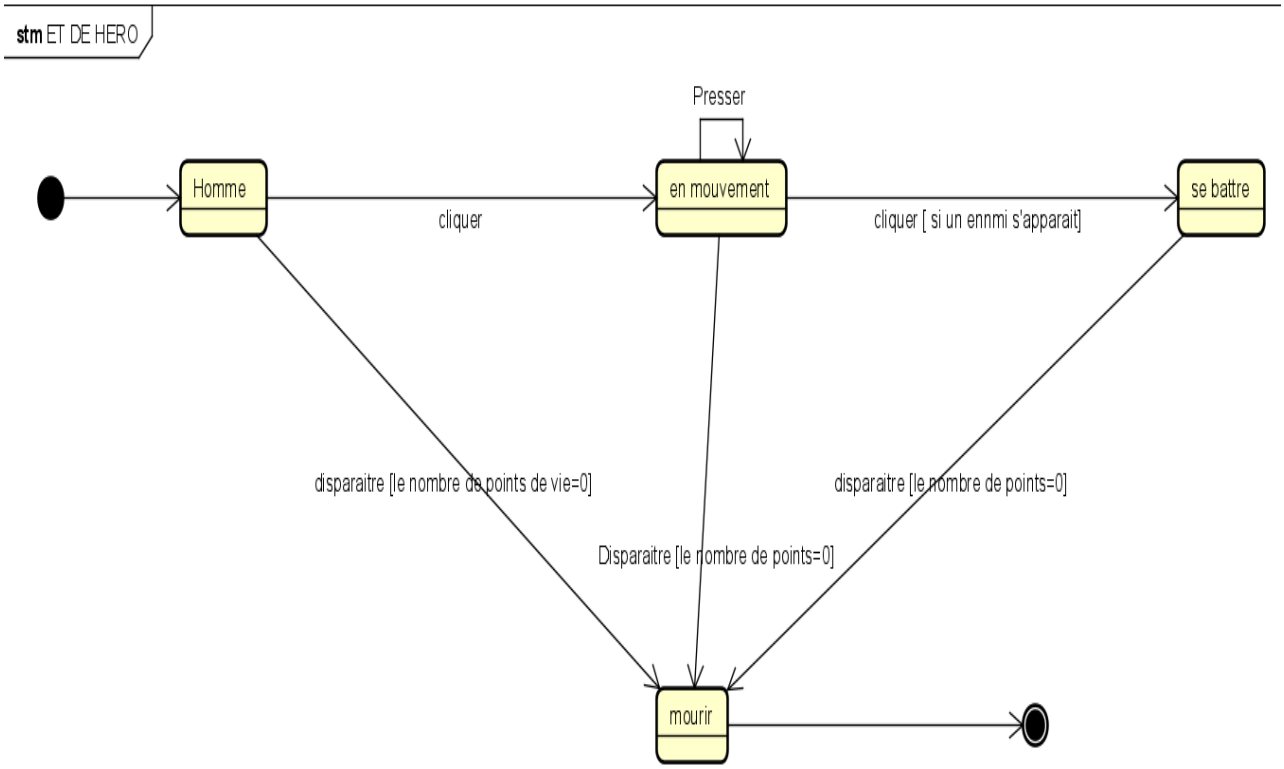



*Figure 3 : Diagramme d'état pour le joueur Hero*

#### II.2.1.1. Description textuelle:

Héro est un objet de jeu, même s'il est un personnage, il change d'états selon les activités de joueur.

Généralement, il est en mouvement si le joueur touche l'un des clés du clavier. Il se bat contre les ennemis, et bien sûr, si le nombre de points de vie est nulle, Héro mort (la fin de la partie).

### II.2.2 Diagramme d'états transition pour l'ennemi:

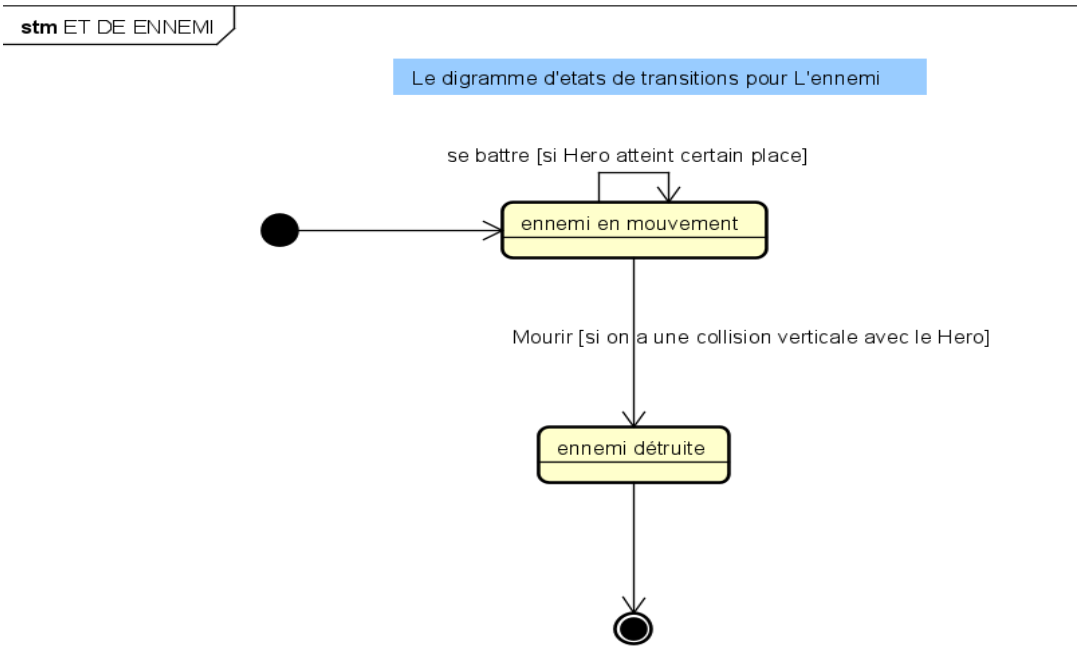



*Figure 4 :Diagramme d'état pour l'ennemi*

### II.2.2.1. Description textuelle:

D'après ce diagramme, on peut distinguer que l'objet Ennemi de notre jeu change d'états, l'ennemi est en mouvement et il est détruit après l'événement « mourir » si on a une collision verticale avec le Héro.

## II.2.3 Diagramme d'états transition pour le Hero :

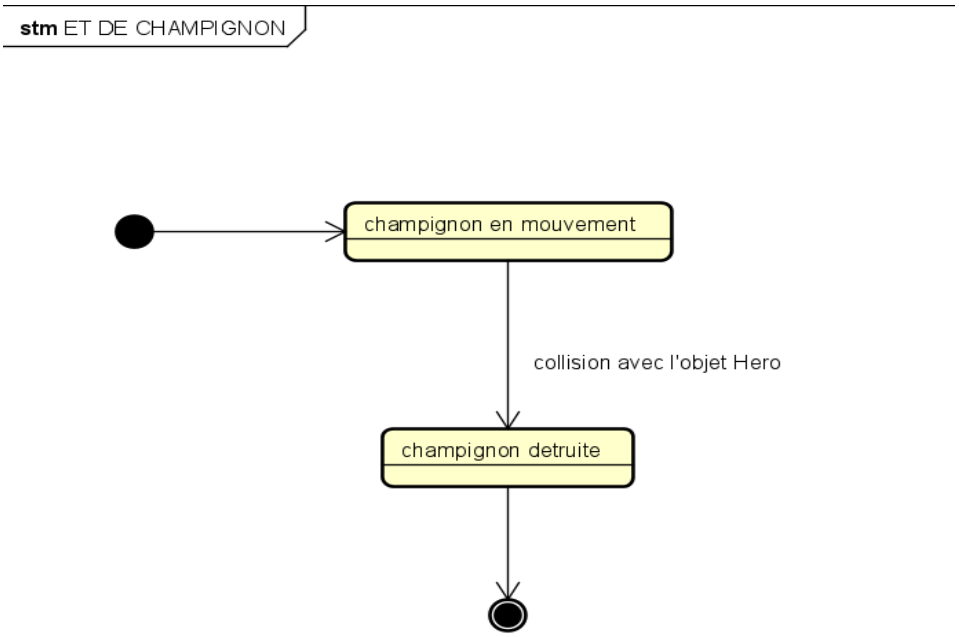



*Figure 5 :Diagramme d'état pour le champignon*

### II.2.3.1. Description textuelle:

D'après ce diagramme, on peut distinguer que l'objet champignon de notre jeu change d'états, de champignon existant en champignon détruit après la collision avec le Héro.

## II.2.4 Diagramme d'états transition pour la barre de vie :

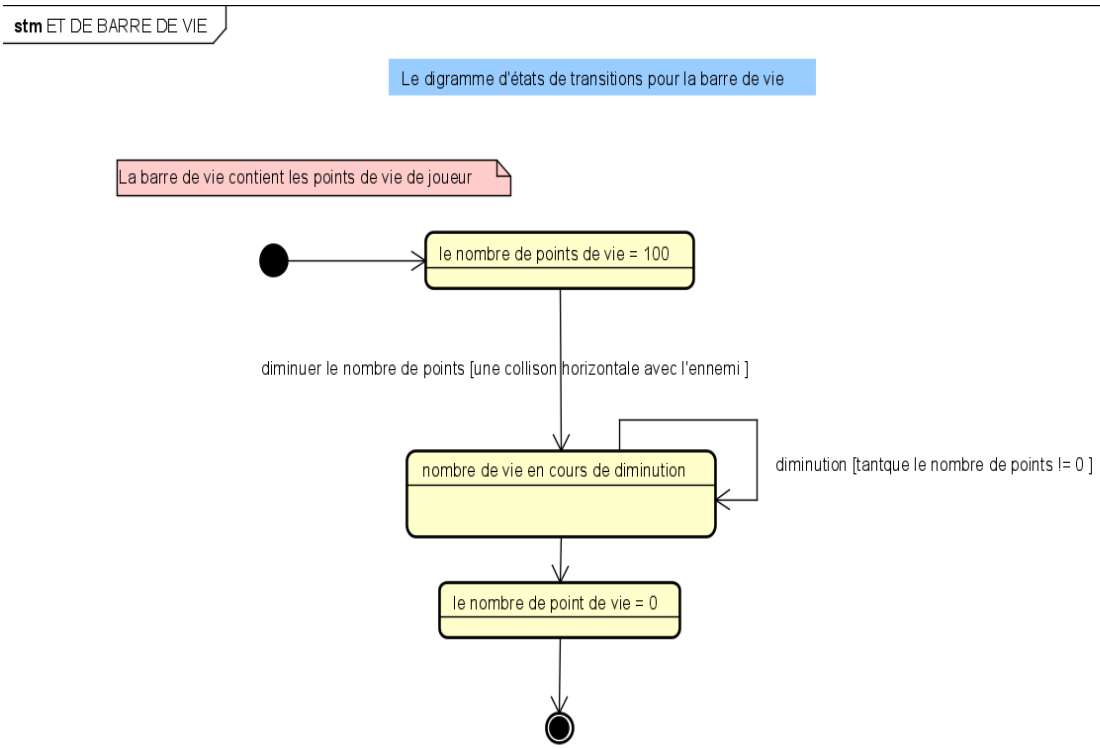



*Figure 6 :Diagramme d'état pour la barre de vie*

### II.2.4.1. Description textuelle:

D'après ce diagramme, l'objet barre de vie change d'états durant le jeu.

Si le nombre de vie > 0, le nombre de vie se diminue à cause de la collision horizontale de Hero et l'ennemi.

Si le nombre de vie=0, c'est la fin de la partie.

## II.3. diagramme de classe:

Le diagramme de classe est l'un des diagrammes UML les plus couramment utilisés pour modéliser la structure statique d'un système logiciel. Il permet de représenter les classes, les attributs, les méthodes et les relations entre les classes.

Le diagramme de classe facilite la compréhension de la structure du système logiciel, en montrant les relations entre les classes, les attributs et les méthodes. Il est utilisé comme base pour la mise en œuvre du système, en fournissant une représentation visuelle des entités et de leurs interactions.

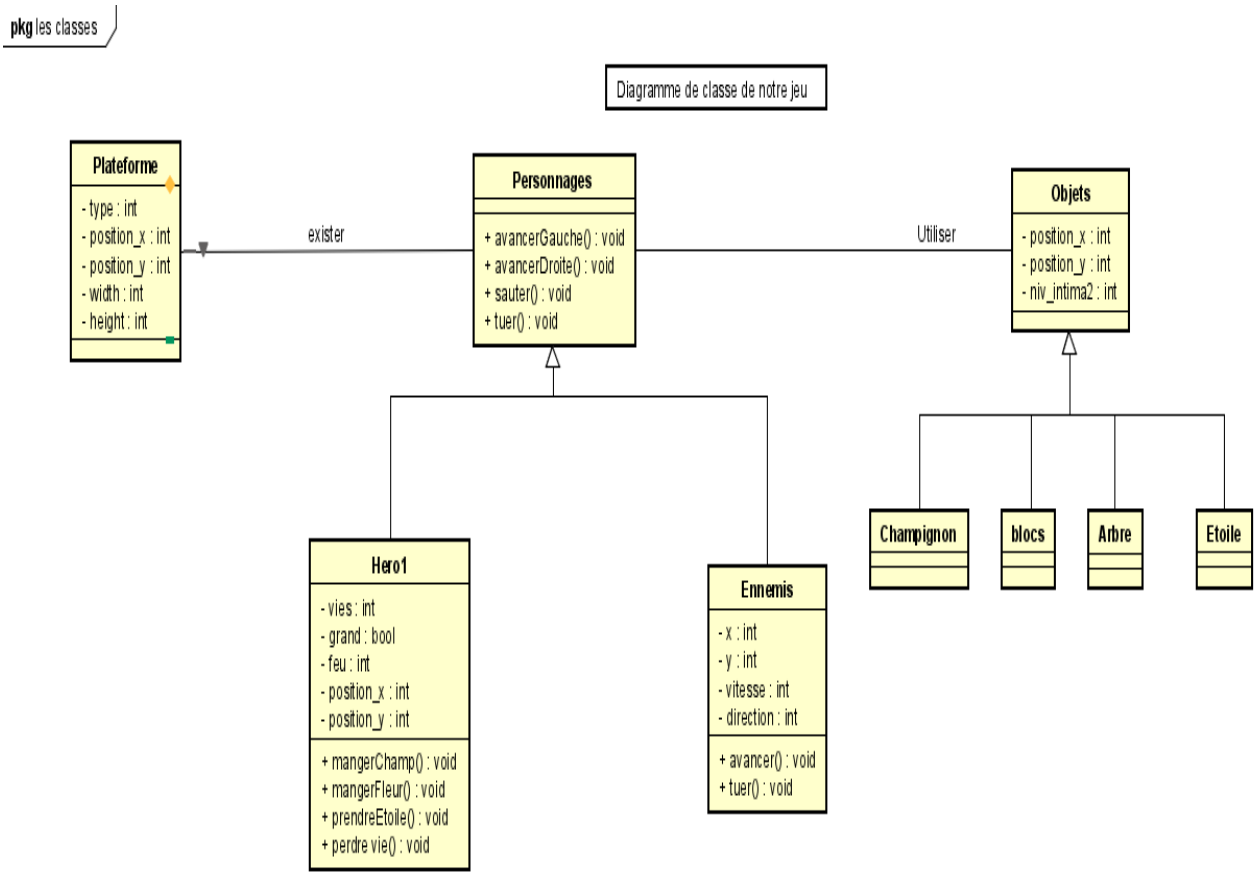



*Figure 7 :Diagramme de classe*

### II.3.1. Description textuelle:

Notre diagramme de classe regroupe l'ensemble des entités qu'on a dans notre sujet, avec leurs attributs, et leurs méthodes, ainsi que l'ensemble des relations et des associations qui existe entre eux, en se basant sur les règles de gestion suivantes :

Règle 1 : Les personnages de jeu existe dans une plateforme.

Règle 2 : Les personnages de jeu utilise des objets.

Règle 3 : Hero et ennemi font partie de personnages de jeu.

## II.4. diagramme d’activité:

Le diagramme d'activité est un diagramme UML qui permet de modéliser le déroulement des activités et des actions dans un processus, une procédure ou un workflow. Il est principalement utilisé pour représenter le comportement dynamique d'un système.

Le diagramme d'activité permet de représenter visuellement le déroulement des activités, les conditions de décision, les boucles et les synchronisations dans un processus. Il est utile pour comprendre et communiquer le flux des actions et des décisions dans un système ou un processus logiciel.

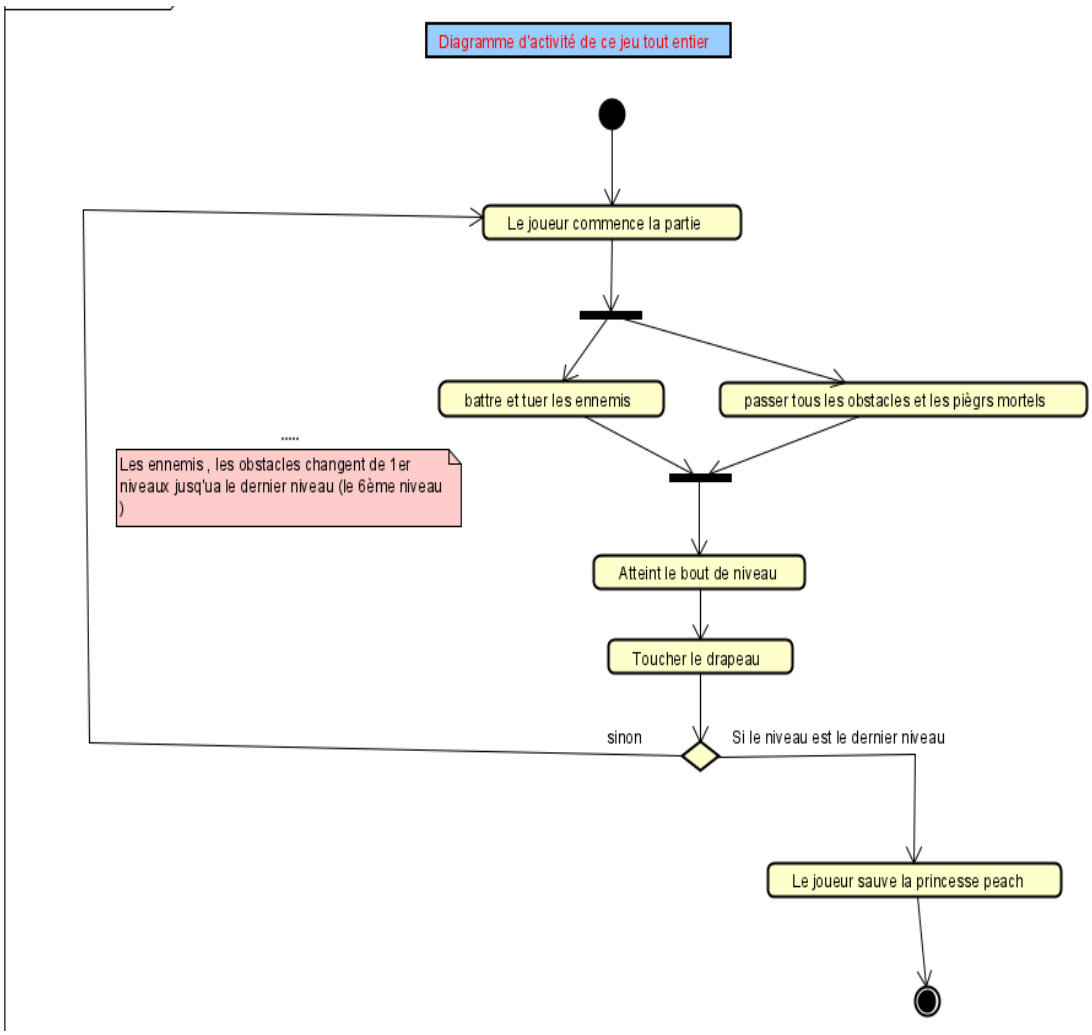



*Figure 8: Diagramme d’activité pour le jeu*

### II.4.1. Description textuelle:

Ce diagramme montre le scenario en général de notre jeu, le joueur commence la partie, ensuite il batte et tue les ennemis aussi qu'il passe les obstacles et les pièges mortels, après il atteint le bout de niveau, toucher le drapeau.

Ici, on a deux scenarios possibles :

Si le joueur joue le niveau 6 de jeu, alors le jeu sera terminé.

Sinon, c'est-à-dire le joueur joue un niveau de 1 à 5, alors, il commence la partie de jeu.

### II.4.2. version plus détaillée du diagramme d'activité:

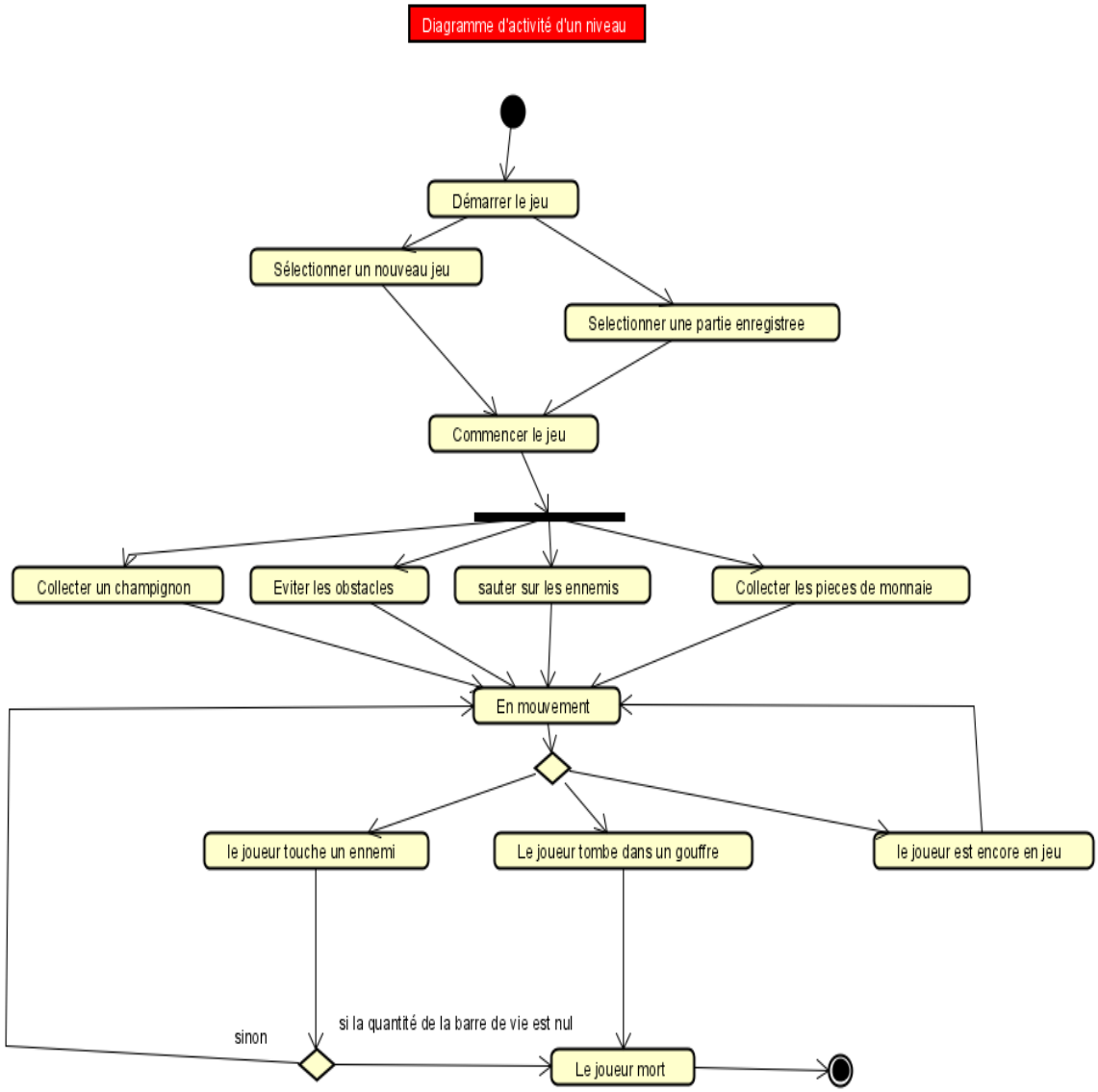



*Figure 9: Diagramme d'activité pour le niveau*

### II.4.2.1. Description textuelle:

Le joueur démarre le jeu, on a deux scenarios possibles, soit le joueur veut sélectionner le nouveau jeu (sélectionner l'un des niveaux qui sont déjà passés) ou sélectionner le niveau de la dernière fois, le jeu commence, on a plusieurs activités que le joueur peut faire et contrôle : « collecter un champignon », « Eviter les obstacles », « sauter sur les ennemis », « collecter les pièces de monnaie », tous cela ça veut dire que le joueur est en mouvement.

A partir de ce moment, plusieurs scenarios possibles, soit « le joueur est encore en jeu » : le joueur est en mouvement, essaie de traverser le niveau.

Soit « le joueur touche l'ennemi », et ici la barre de vie se diminue. Si la barre de vie est nulle alors le joueur mort.

Soit « Le joueur tombe dans un gouffre » et donc le joueur mort (la fin du partie).

## II.5. diagramme de déploiement:

Le diagramme de déploiement représente l'architecture physique déploiement permet de supportant l'exploitation du système.

Il est similaire à un réseau constitué de nœuds et de connexions entre ces nœuds qui modélise cette architecture.

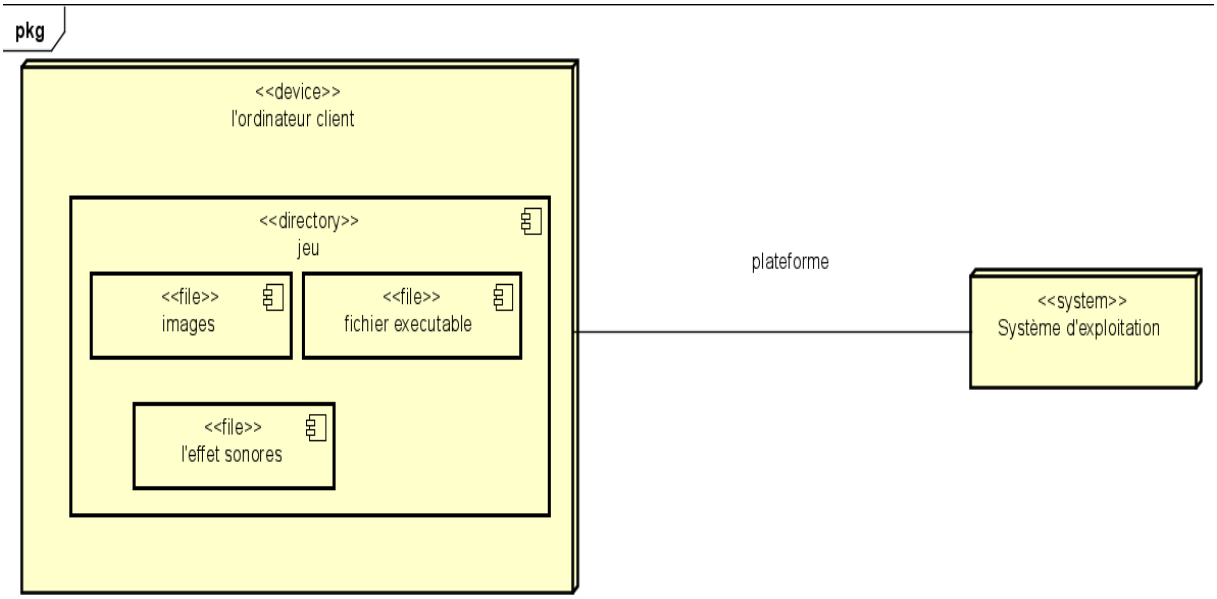



*Figure 10: Diagramme de déploiements*

### II.5.1. Description textuelle:

Le fichier exécutable est un fichier crée à partir de scripts Python, ce qui facilite la distribution de vos applications Python à des utilisateurs qui n'ont pas Python installé sur leur système.

Le fichier exécutable contient les bibliothèques requises et votre script.

Ce fichier doit être dans un répertoire qui contient les sons et les images utilisables lors de développement.

Ainsi, un ordinateur client besoin d'un système d'exploitation qui est un nœud tout entier.

## II.6. diagramme de composants:

Un diagramme de composants est un type de diagramme UML (Unified Modeling Language) qui représente les composants logiciels d'un système et leurs relations. Il permet de visualiser la structure interne d'un système et les dépendances entre les différents composants.

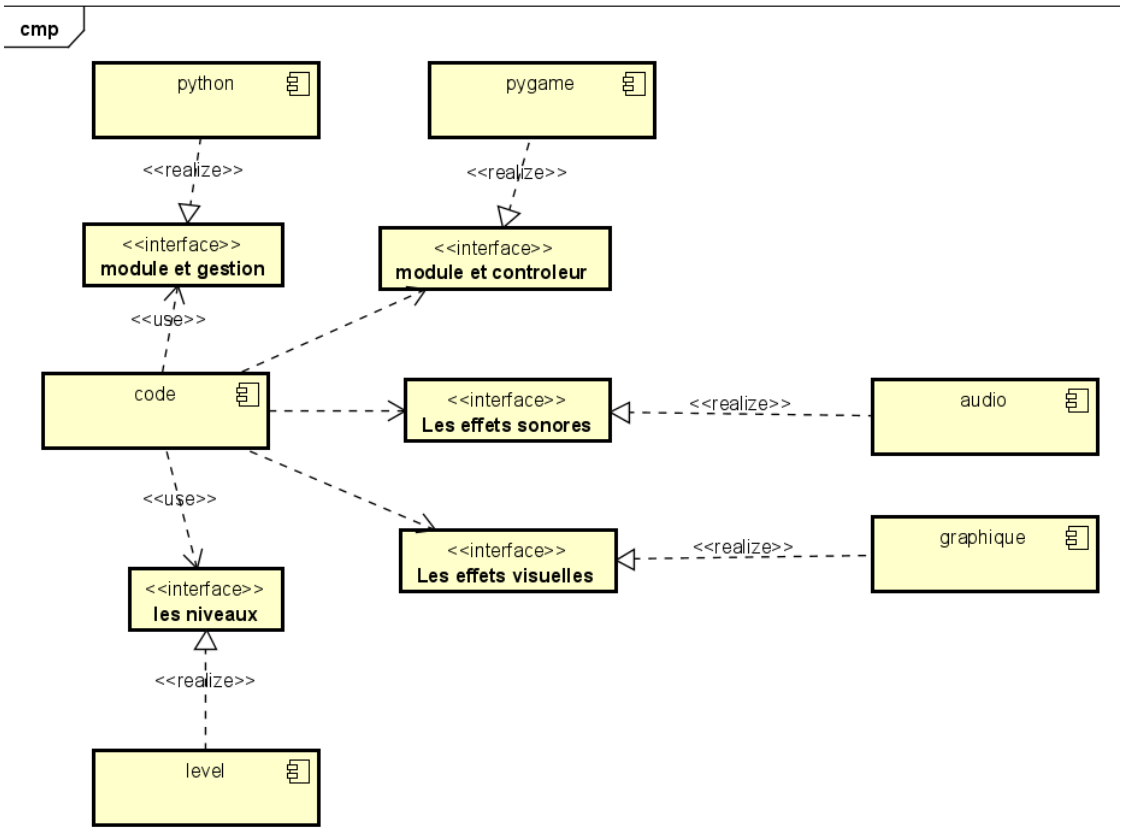



*Figure 11 : Diagramme de composants*

## II.6.1. Description textuelle:

Lors de développement de notre jeu, nous avons structuré l'ensemble de notre code en quatre composants principaux, ce sont des répertoires.

Le répertoire ***code*** contient un ensemble de code source (.py).

Le répertoire ***audio*** contient la musique utilisable dans le jeu.

Le répertoire ***niveau*** contient l'ensemble des fichiers dont on est besoin pour avoir les

niveaux.

Le répertoire ***graphique*** contient l'ensemble des photos pour créer la première interface, les ennemis, les pièces de monnaie ...

# III. chapitre 3:

## LES TECHNOLOGIES ET IMAGES DU TRAVAIL

# III. chapitre 3:

## III.1. LES TECHNOLOGIES ET LES LOGICIELS:

### III.1.1. Python:


Python est un langage de programmation interprété, de haut niveau, polyvalent et facile à lire et à écrire. Il a été créé par Guido van Rossum et sa première version a été publiée en 1991. Depuis lors, Python est devenu très populaire grâce à sa syntaxe concise et expressive, ainsi qu'à sa vaste bibliothèque standard et à sa communauté active.

Voici quelques caractéristiques et points forts de Python :

1. Syntaxe claire et lisible : Python favorise la lisibilité du code avec une syntaxe épurée et une indentation significative, ce qui facilite la compréhension du code par les programmeurs.

2. Facilité d'apprentissage : Python est souvent recommandé comme langage d'initiation à la programmation en raison de sa simplicité syntaxique. Il est facile à apprendre pour les débutants tout en restant puissant pour les développeurs expérimentés.

3. Large communauté et bibliothèque standard : Python bénéficie d'une communauté active de développeurs qui contribuent à son développement et à la création de nombreuses bibliothèques tierces. La bibliothèque standard de Python offre également de nombreux modules et outils prêts à l'emploi pour diverses tâches.

4. Portabilité : Python est un langage multiplateforme, ce qui signifie qu'il peut être exécuté sur différents systèmes d'exploitation tels que Windows, macOS et Linux, offrant ainsi une grande flexibilité.

5. Utilisations variées : Python est utilisé dans de nombreux domaines tels que le développement web, l'analyse de données, l'intelligence artificielle, l'automatisation de tâches, les sciences et la recherche, la création d'interfaces graphiques, etc.

Python dispose également de nombreuses bibliothèques spécialisées, telles que NumPy pour le calcul scientifique, Pandas pour l'analyse de données, Django pour le développement web, TensorFlow pour l'apprentissage automatique, et bien d'autres.

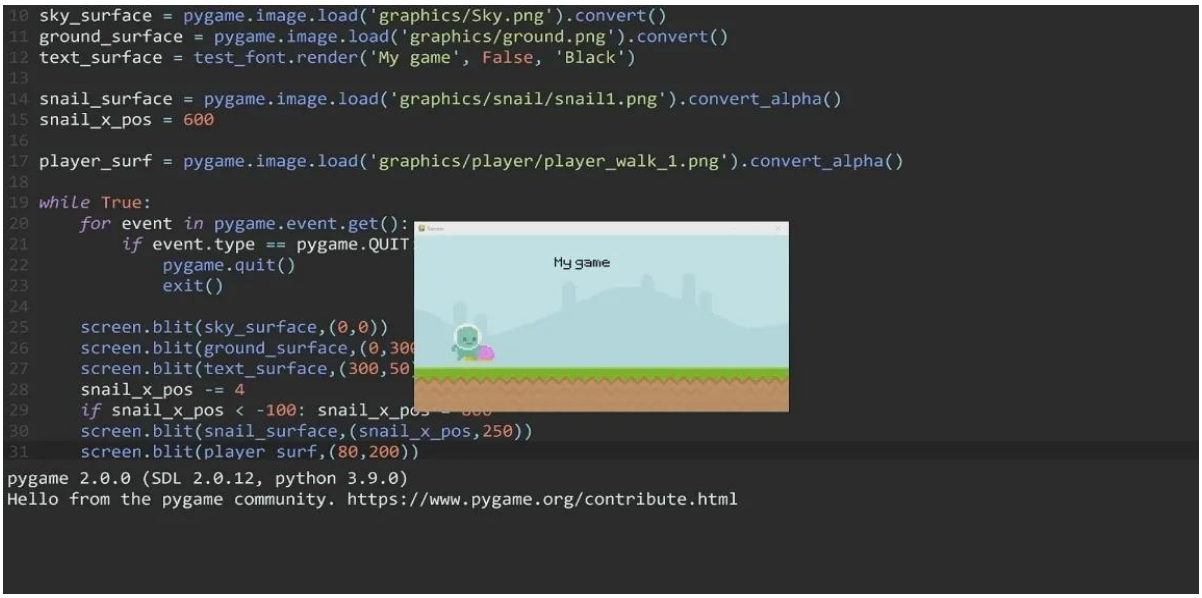
En résumé, Python est un langage de programmation populaire et polyvalent, apprécié pour sa simplicité, sa lisibilité et sa vaste communauté de développeurs.

### III.1.2. Pygame

### III.1.2.1. Qu'est-ce que PyGame ?

PyGame est **un jeu de modules Python multiplateformes, libre et open-source**, conçu pour la **création de jeux vidéo et autres contenus multimédia**. Il regroupe des bibliothèques de graphismes et de sons pensées pour être utilisées avec le langage de programmation.

Cet outil fut initialement créé par Pete Shinners, dans le but de **remplacer PySDL** après l'abandon du projet. Depuis l'an 2000, il s'agit d'un projet communautaire relaxé sous la licence de **logiciel libre GNU Lesser General Public Licence**. Il peut donc être distribué sous forme de logiciel open source ou commercial.



### III.1.2.2.Comment fonctionne PyGame ?

PyGame repose sur la **bibliothèque Simple DirectMedia Layer (SDL)**, permettant l'interface et la gestion d'appareils d'entrée tels qu'un clavier, une souris ou un joystick et d'appareils de sortie comme un écran ou un haut-parleur.

Le but est de permettre le **développement de jeu vidéo en temps réel sans les mécaniques de bas niveau du langage C et de ses dérivés**. Les fonctions de jeu les plus basiques sont abstraites, simplifiant fortement le développement.

En d'autres termes, plutôt que d'avoir à écrire un moteur de jeu en Python depuis le début, PyGame **permet d'exploiter des fonctions déjà codées**. Ceci permet de se concentrer sur le jeu en lui-même.

Par ailleurs, SDL propose aussi des fonctionnalités de mathématiques vectorielles, détection de collision, gestion de sprite 2D, manipulation de matrice de pixels, transformations, filtrage et dessin. Elle prend aussi en charge le MIDI et les polices freetype.

### III.1.2.3. SDL:

**SDL (Simple DirectMedia Layer)** est une bibliothèque multiplateforme qui fournit un accès bas niveau aux fonctionnalités audio, vidéo, d'entrée et d'affichage graphique des systèmes d'exploitation. Voici un schéma simplifié du fonctionnement de SDL :

| **Application SDL** |
|---|
| **Événements** |
| **Fenêtre SDL** |
| **Surface SDL** |
| **Texture/Buffer SDL** |
| **Affichage** |

| **Audio** |
|---|
| **Entrée** |

L'application SDL est le point d'entrée de votre programme. Elle gère l'initialisation de SDL, la gestion des événements, les boucles de jeu, etc.
Les événements représentent les entrées de l'utilisateur (clavier, souris, joystick, etc.) et les événements système (fermeture de fenêtre, redimensionnement, etc.). L'application SDL reçoit ces événements et y réagit en conséquence.
La fenêtre SDL représente la zone visible à l'écran. Elle peut être créée, redimensionnée, déplacée, etc. Elle est composée de surfaces SDL.
Les surfaces SDL sont des zones mémoires où sont stockées les données graphiques. Elles peuvent représenter des images, des animations, des textures, etc. Les surfaces peuvent être manipulées (dessiner, copier, coller, etc.) pour créer des effets visuels.
Les textures/buffers SDL sont des améliorations des surfaces SDL, utilisées pour accélérer le rendu graphique. Elles peuvent être chargées à partir de surfaces ou créées à partir de zéro. Les textures sont ensuite affichées sur l'écran.
L'affichage gère l'affichage des textures à l'écran. Il peut y avoir plusieurs affichages, mais généralement un seul est utilisé. L'affichage est responsable du double buffering, de la synchronisation verticale et du rendu graphique final à l'écran.
L'audio gère la lecture et l'enregistrement du son. Il permet de charger des fichiers audios, de les jouer, de régler le volume, etc.
L'entrée gère les périphériques d'entrée tels que le clavier, la souris, les manettes de jeu, etc. Elle permet de détecter les entrées de l'utilisateur et d'y réagir.
**Pygame,** quant à lui, utilise SDL comme base et fournit une interface de haut niveau pour simplifier le développement de jeux en Python. Voici un aperçu simplifié du fonctionnement de Pygame :

| **Application Pygame** |
| --- |
| **Événements** |
| **Fenêtre** |
| **Sprites** |
| **Sons** |
| **Contrôles** |

L'application Pygame est le point d'entrée de votre programme. Elle initialise Pygame, gère les événements, les boucles de jeu

### III.1.2. visual studio code :


Visual Studio Code (VS Code) est un environnement de développement intégré (IDE) léger et gratuit, développé par Microsoft. Il est largement utilisé par les développeurs pour écrire du code dans différents langages de programmation, y compris Python, JavaScript, C++, Java, et bien d'autres

### III.1.3. Astah UML :

Astah UML (anciennement connu sous le nom de JUDE) est un outil de modélisation UML (Unified Modeling Language) populaire et puissant. Il est utilisé pour la création de diagrammes UML afin de concevoir, modéliser et documenter des systèmes logiciels.

### III.1.4. tiled :

**le système tileset** :

Un système de tileset est une méthode de représentation graphique utilisée dans la création de jeux vidéo 2D, en particulier les jeux de style rétro ou les jeux basés sur des tuiles (tile-based games).

Dans un jeu utilisant un système de tileset, l'écran de jeu est divisé en une grille régulière de petites images appelées tuiles (tiles). Chaque tuile représente un élément graphique spécifique, comme une partie du décor, un personnage, un objet interactif, etc. Les tuiles sont généralement de taille fixe et peuvent être carrées ou rectangulaires.
L'ensemble des tuiles utilisées dans un jeu est appelé le tileset. Il s'agit d'une collection d'images individuelles regroupées dans un seul fichier ou une seule ressource. Chaque tuile est identifiée par un numéro ou une coordonnée dans le tileset, ce qui permet de les organiser et de les utiliser de manière cohérente dans le jeu.
Le système de tileset offre plusieurs avantages. Tout d'abord, il permet de créer des niveaux ou des cartes de jeu de manière modulaire et efficace. Au lieu de dessiner chaque pixel du décor individuellement, vous pouvez assembler les tuiles pour former des structures plus grandes, ce qui simplifie le processus de conception et permet des modifications rapides.
De plus, le système de tileset permet d'économiser de la mémoire et des ressources graphiques. En utilisant un ensemble de tuiles réutilisables, il n'est pas nécessaire de stocker des images uniques pour chaque élément du jeu. Cela réduit la taille des fichiers et facilite la gestion des ressources graphiques.

Enfin, le système de tileset facilite l'animation des objets et des personnages. En combinant différentes tuiles dans des séquences d'images, il est possible de créer des effets d'animation fluides pour les mouvements, les attaques, les transitions, etc.

En résumé, un système de tileset est une méthode de représentation graphique basée sur l'utilisation de tuiles (images) assemblées pour former des niveaux, des cartes ou des décors de jeu. Cela permet une conception modulaire, une économie de ressources graphiques et facilite l'animation des éléments du jeu.

**Tiled:**

Est un éditeur de cartes open-source qui permet de créer des niveaux et des environnements pour les jeux en 2D. Il offre une interface conviviale pour concevoir des cartes basées sur des tuiles (tiles) et les organiser de manière à former des niveaux de jeu.

Dans le contexte de Pygame, Tiled est souvent utilisé en conjonction avec cette

bibliothèque pour la création de jeux en 2D.

**Tiled et Pygame :**

1. Création des niveaux : Tiled permet aux développeurs de concevoir les niveaux du jeu en plaçant des tuiles sur une grille. Les tuiles représentent les différents éléments du niveau tels que les sols, les murs, les obstacles, les objets interactifs, etc. On peut également définir des propriétés spécifiques aux tuiles, comme les collisions ou les informations supplémentaires.

2. Exportation des niveaux : Une fois que le niveau est conçu dans Tiled, il peut être exporté dans différents formats de fichiers compatibles avec Pygame, tels que le format TMX (Tile Map XML). Ce format contient les informations sur la disposition des tuiles, les propriétés des tuiles et d'autres éléments du niveau.

3. Chargement des niveaux dans Pygame : Dans le code Pygame, le fichier exporté depuis Tiled peut être chargé et analysé pour obtenir les informations sur les tuiles et les propriétés du niveau. Pygame offre des fonctionnalités pour charger des fichiers TMX et extraire les données nécessaires pour construire les niveaux dans le jeu.

4. Affichage des niveaux : Une fois les données du niveau extraites, Pygame peut utiliser ces informations pour afficher les tuiles à l'écran, en respectant la disposition et les propriétés définies dans Tiled. Les tuiles peuvent être affichées de manière statique ou animée, en fonction des besoins du jeu.

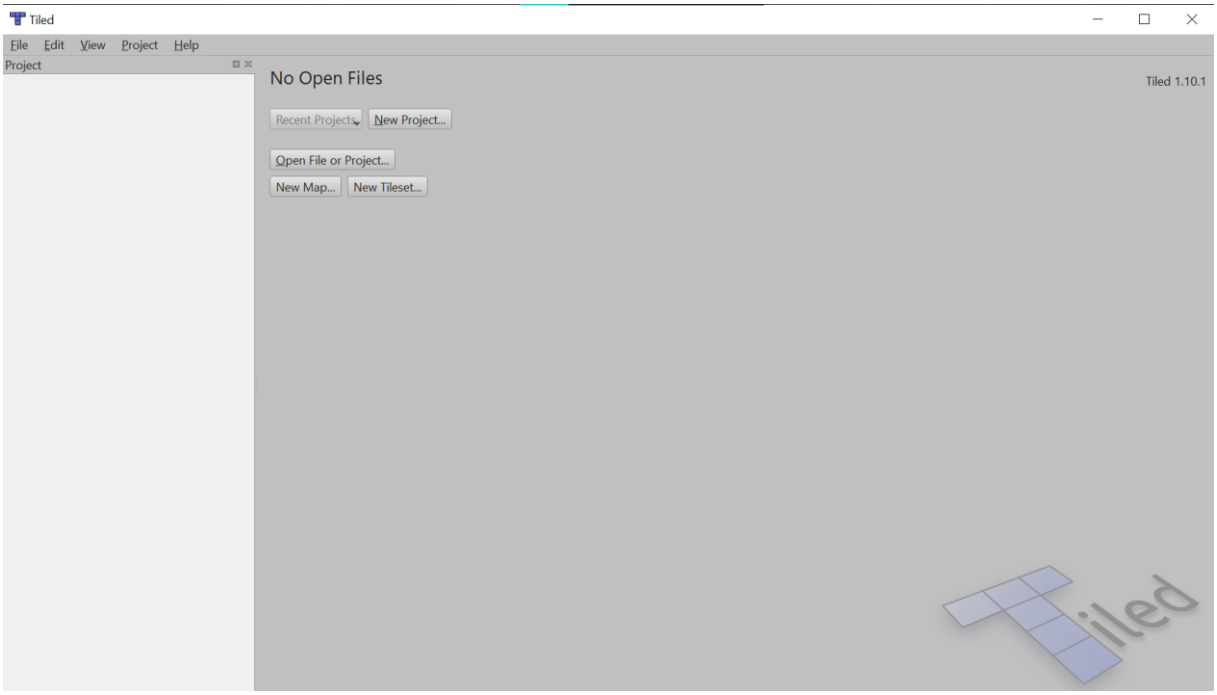
En utilisant Tiled avec Pygame, les développeurs peuvent bénéficier d'un éditeur de cartes spécialisé pour la conception de niveaux, tout en utilisant les fonctionnalités graphiques et d'affichage de Pygame pour créer des jeux 2D immersifs et interactifs.

En résumé, Tiled est un éditeur de cartes permettant de créer des niveaux de jeu en 2D basés sur des tuiles. Il est utilisé avec Pygame pour concevoir, exporter et charger des niveaux dans les jeux, offrant ainsi une solution pratique pour la création de mondes de jeu structurés et visuellement attrayants.

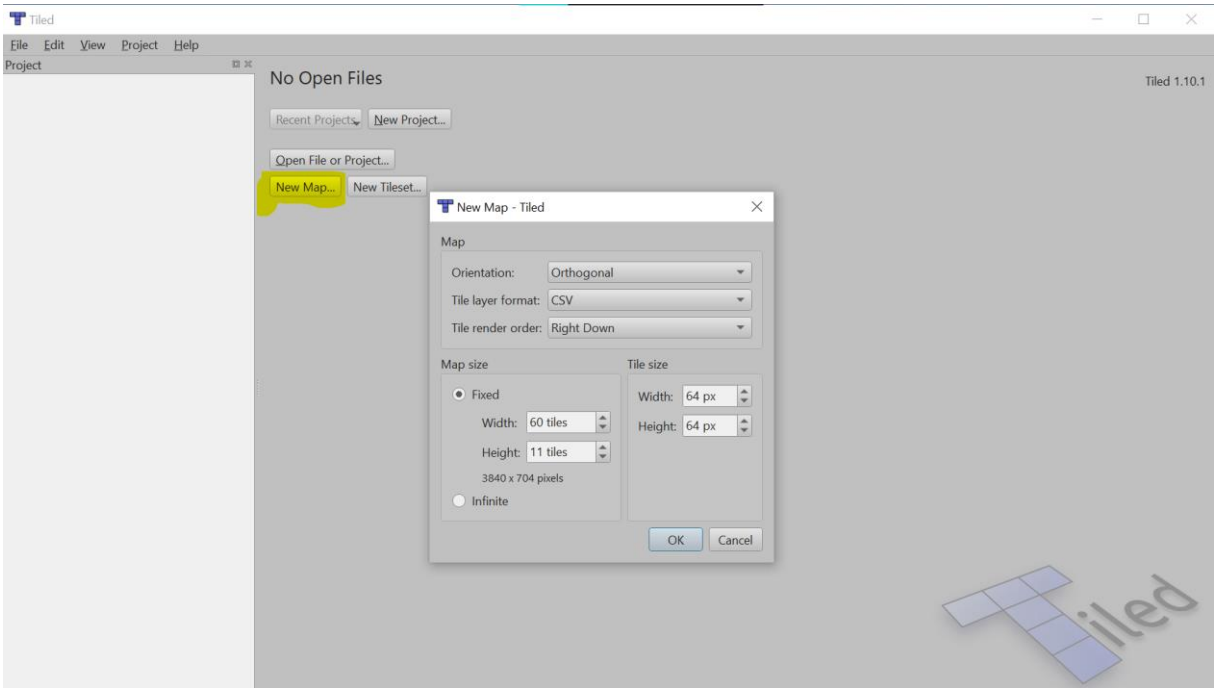
Nous avons utilisé l'outil pour créer les différents plateformes (surfaces solides sur lesquelles le Hero marche et saute), Voici les étapes de l'utilisation :

Etape 1 : Installation du logiciel via le lien : https://www.mapeditor.org/

Etape 2 : Ouvrir le logiciel :




Etape 3 : Choisir New map



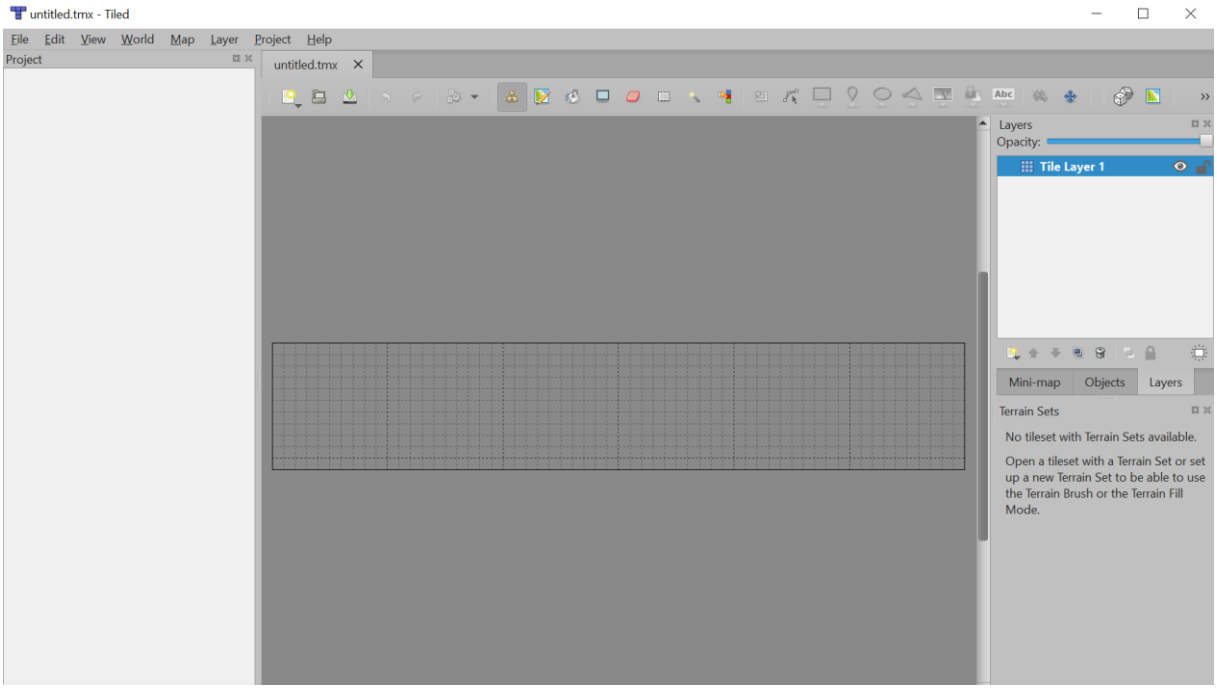

Etape 4 : Choisir width et height de Map size , width et height de tile size :

La longueur et la largeur de map est le nombre de tuiles en largeur et longueur.

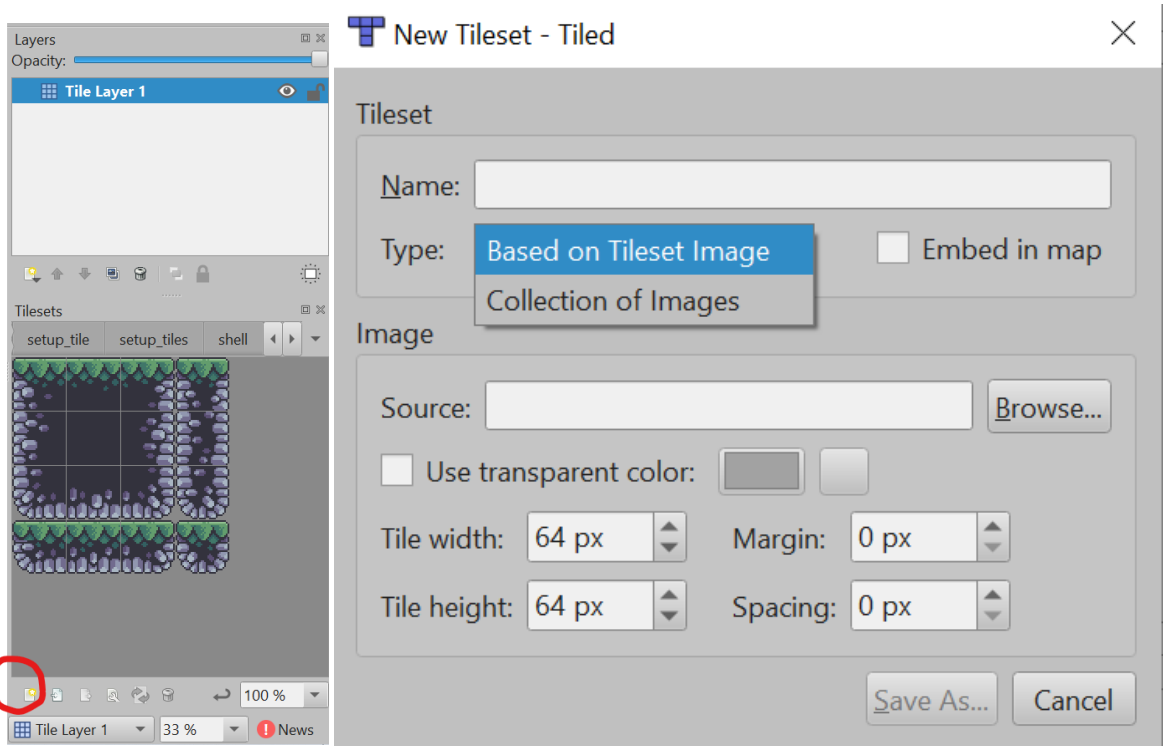
La longueur et la largeur de tile size est en pixels.

Le niveau 1 , 2, 3 et 4 : ont un largeur de map de 80 , alors que 5 et 6 ont la largeur de map 120 .

Etape 5 : Cliquer OK :

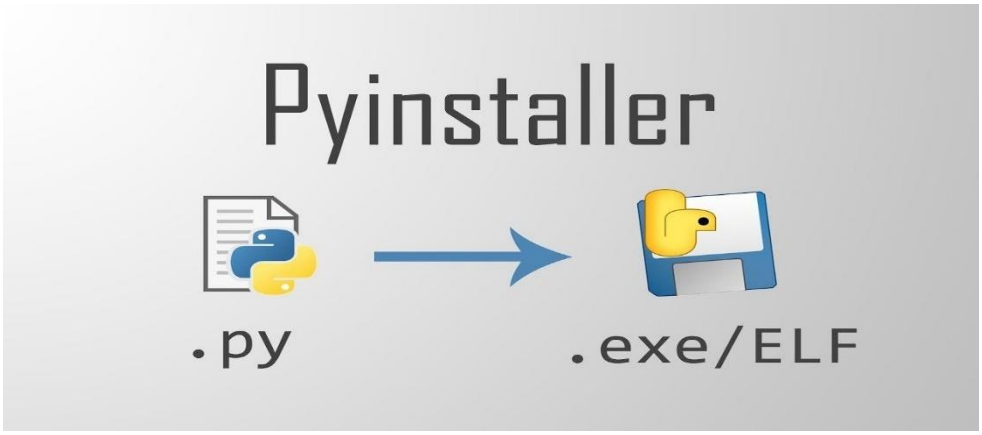



Etape 6 : Choisir les photos :




Soit c'est une image, soit c'est une image basée sur des tuiles.

### III.1.5. Pyinstaller:




Après le développement de jeu, il faut le déployer pour avoir le fichier exécutable pour qu'il puisse être téléchargeable par les joueurs (utilisateurs), d'où on est besoin d'un outil de déploiement, il en a plusieurs ex : py2exe , PyOxidizer, cx_freeze….

Nous avons choisi d'utiliser l'outil pyinstaller car il est facile, rapide, offre différentes fonctionnalités.

Voici une petite définition et explication :

PyInstaller est un outil permettant de créer des exécutables autonomes à partir de scripts Python. Il permet de convertir vos scripts Python en fichiers exécutables, de sorte que vous pouvez les distribuer facilement à des utilisateurs qui n'ont pas Python installé sur leur système.

Lorsque vous développez une application Python, vous avez généralement besoin d'un environnement Python pour exécuter le script. Cependant, cela peut poser des problèmes si vous souhaitez distribuer votre application à des utilisateurs qui n'ont pas Python ou les dépendances requises installées. C'est là que PyInstaller entre en jeu.

PyInstaller analyse votre script Python et toutes ses dépendances pour les empaqueter dans un seul exécutable. Il crée un fichier binaire qui inclut l'interpréteur Python, les bibliothèques requises et votre script, de sorte que l'application peut être exécutée sans avoir besoin d'installer Python séparément.

Une fois que vous avez généré l'exécutable avec PyInstaller, vous pouvez le distribuer à des utilisateurs sur différentes plateformes (Windows, macOS, Linux) sans se soucier des

dépendances ou des problèmes d'installation. Cela facilite la diffusion de vos applications Python à un public plus large.

PyInstaller prend également en charge plusieurs options de personnalisation, vous permettant de spécifier des icônes personnalisées pour l'exécutable, d'inclure des fichiers supplémentaires nécessaires à votre application et de configurer d'autres paramètres avancés.

En résumé, PyInstaller est un outil pratique pour créer des exécutables autonomes à partir de scripts Python, ce qui facilite la distribution de vos applications Python à des utilisateurs qui n'ont pas Python installé sur leur système.

## III.2. LES IMAGES ( FRAMES ) :

Dans Pygame, une image est une représentation statique d'un élément graphique, comme un personnage, un objet ou un décor du jeu. Les images peuvent être créées à l'aide d'un logiciel de création graphique et sont généralement au format PNG, JPEG ou BMP. Pygame fournit des fonctions pour charger et afficher des images dans une fenêtre de jeu. Vous pouvez charger une image avec la fonction **`pygame.image.load()`** et l'afficher à l'écran à l'aide de la fonction **`pygame.Surface.blit()`**.

Le terme "frame" (ou trame) fait référence à une image individuelle dans une séquence d'animation. Lorsqu'une animation est jouée, elle consiste en une série de frames affichées successivement à un rythme rapide, ce qui crée l'illusion du mouvement. Dans Pygame, vous pouvez représenter une animation en regroupant plusieurs images dans une liste et en les affichant séquentiellement dans la boucle principale du jeu.

### III.2.1. les ennemis :

les ennemis sont des éléments interactifs qui tentent d'entraver la progression du joueur et lui causer des dommages

### III.2.2. les coins :

il y a deux types de pièces et de "coins" (monnaie) que le joueur peut collecter

1. Pièces d'argent: Ce sont les pièces les plus courantes dans le jeux Mario. Le joueur peut les collecter pour gagner une points

2. Pièces jaunes : Ces pièces ont une valeur plus élevée que les pièces d'argent normales. Leur collecte donne 2 point

### III.2.3. Les images de niveaux :

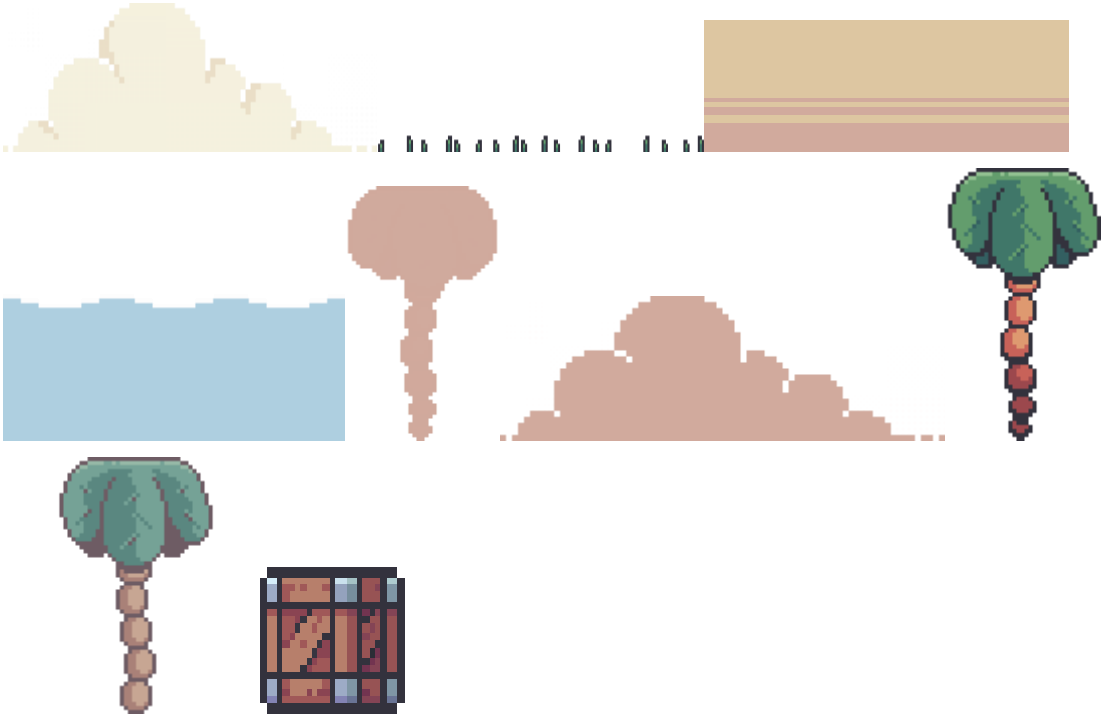
Les images de niveaux du jeu Mario sont des illustrations représentant visuellement les différents environnements, décors et éléments de gameplay présents dans les niveaux des jeux Mario. Ces images permettent de donner un aperçu visuel des niveaux et de montrer les obstacles, les ennemis, les plateformes et les mécanismes spécifiques qui caractérisent chaque niveau.



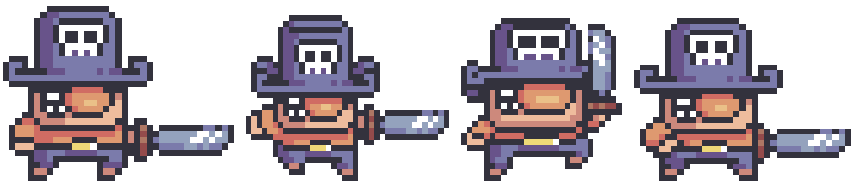
### III.2.4. Les images de decoration :

Les images de décoration du jeu Mario représentent les différents éléments visuels qui composent les décors et les environnements des niveaux dans les jeux Mario. Ces éléments de décoration sont conçus pour créer l'atmosphère, l'esthétique et l'ambiance

des différents mondes et niveaux du jeu. Ils comprennent des objets, des paysages, des structures architecturales, des végétaux, des textures et des motifs spécifiques.



### III.2.5. Les images du player :



### III.2.6. Champignon :

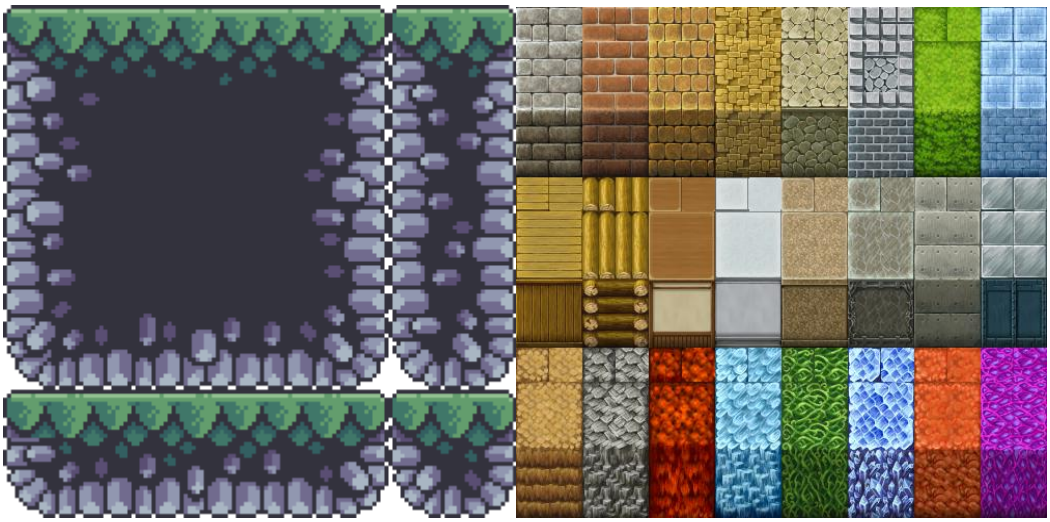
Lorsque Mario le collecte, il gagne une vie supplémentaire. C'est un objet précieux pour accumuler des vies supplémentaires dans le jeu.

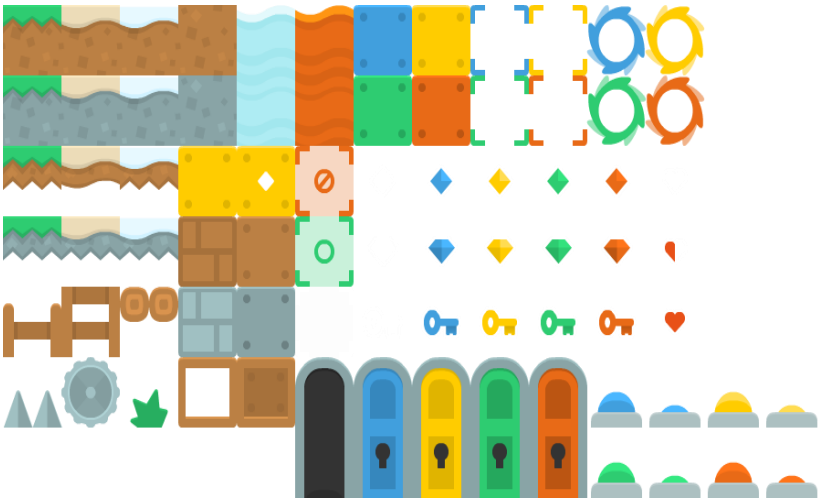
### III.2.7. La barre de vie :

La barre de vie, également connue sous le nom de barre de santé , est une représentation visuelle de la quantité de vie ou de santé du personnage le jeu .

Elle est souvent affichée à l'écran pour permettre aux joueurs de suivre facilement l'état de santé. La barre de vie est présentée sous la forme d'une ligne horizontale ou qui se remplit ou se vide en fonction des dommages subis par le personnage.

### III.2.8. Les images de terrain :





# IV. chapitre 4:

## PHASE D'IMPLÉMENTATION

# IV. chapitre 4:

## IV.1.Introduction:

Après avoir présenté une étude conceptuelle dans le chapitre précédent, Nous allons, dans ce chapitre présenter les résultats de notre travail (quelque interface graphique), et finir par une conclusion.

## IV.2. PAGE DE SÉLECTION DE NIVEAUX:

la page où les joueurs peuvent choisir parmi une liste de niveaux disponibles, et afficher la progression du joueur, montrant les niveaux déjà terminés et ceux qui restent à jouer

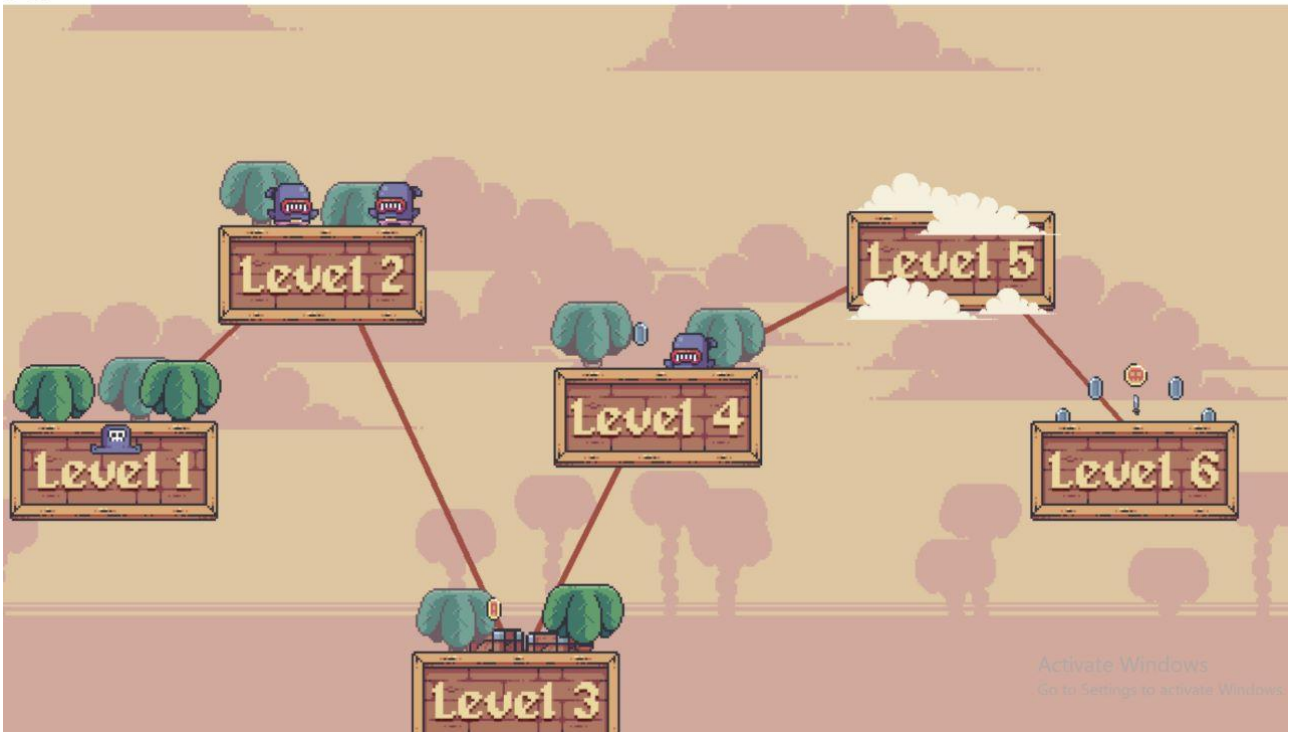



Figure 21 : le jeu

## IV.3. LES STAGES DU JEU :

Chaque niveau représente un environnement ou un défi spécifique que les joueurs doivent parcourir et surmonter.

- Le premier niveau où il y a un étage d'un type extrait de l'image num:1 et se caractérise par un niveau de difficulté facile

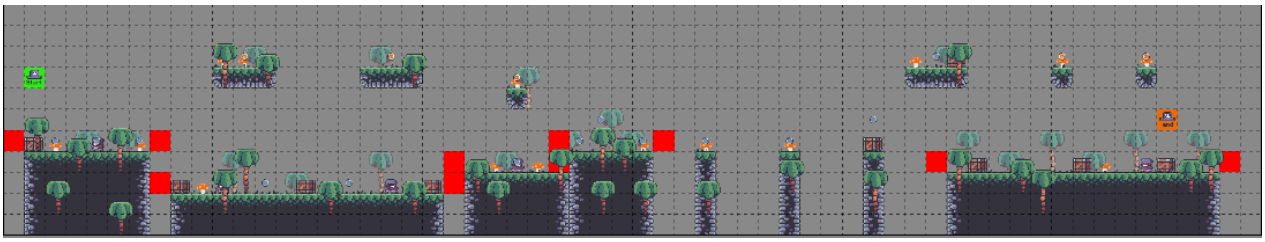


**Figure 15 : La plateforme de niveau 1**

- Le premier niveau où il y a un étage d'un type extrait de l'image num:1 et se caractérise par un niveau de difficulté moyen

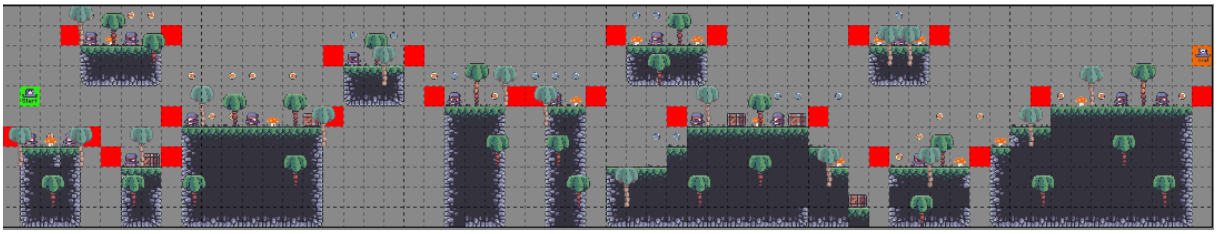


**Figure 16 : La plateforme de niveau 2**

- Le premier niveau où il y a un étage d'un type extrait de l'image num:2 et se caractérise par un niveau de difficulté moyen

**Figure 17 : La plateforme de niveau 3**

- Le premier niveau où il y a un étage d'un type extrait de l'image num:2 et se caractérise par un niveau de difficulté difficile

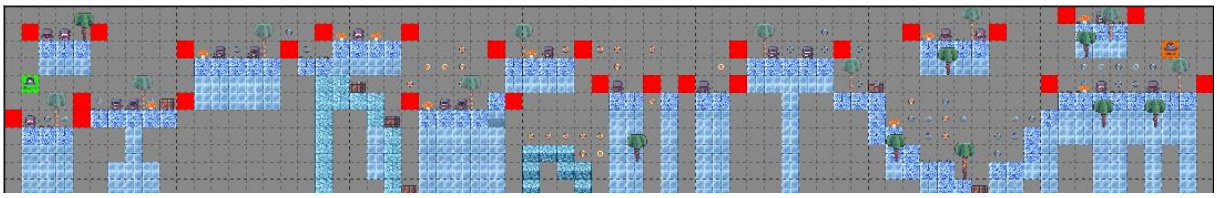


**Figure 18 : La plateforme de niveau 4**

- Le premier niveau où il y a un étage d'un type extrait de l'image num:3 et se caractérise par un niveau de difficulté difficile

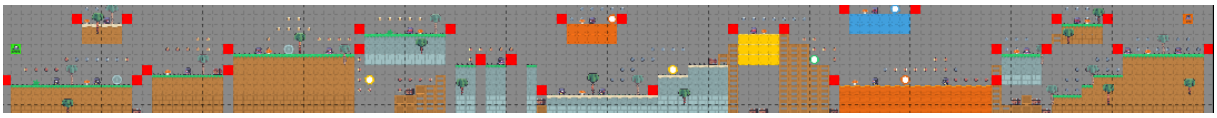


**Figure 19 : La plateforme de niveau 5**

- Le premier niveau où il y a un étage d'un type extrait de l'image num:3 et se caractérise par un niveau de difficulté difficile

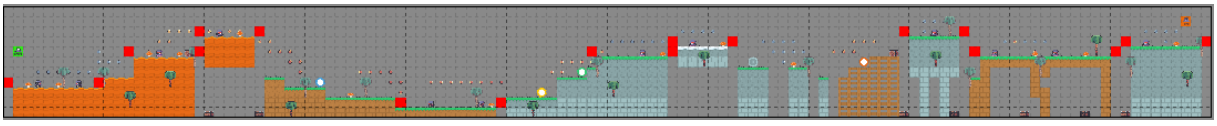


**Figure 20 : La plateforme de niveau 6**

# Conclusion général

En conclusion, le développement du jeu Mario a été une expérience enrichissante et stimulante. À travers les différentes étapes, de l'identification des besoins à l'implémentation, nous avons pu créer un jeu captivant et divertissant. En étudiant les attentes des joueurs, nous avons pu concevoir des niveaux stimulants, des mécaniques de jeu engageantes et une interface utilisateur conviviale.

L'utilisation des outils et technologies appropriés a été essentielle pour concrétiser notre vision du jeu.  nous avons pu créer des environnements interactifs, des animations fluides et des effets visuels attrayants. Les langages de programmation et les logiciels de conception graphique ont été des ressources précieuses dans le processus de développement.

Le développement de ce jeu nous a également permis de développer nos compétences en programmation, en résolution de problèmes et en gestion de projet. Nous avons appris à optimiser le code, à déboguer les erreurs et à collaborer efficacement en équipe pour atteindre nos objectifs.

En fin de compte, le jeu Mario que nous avons créé est un hommage à l'univers emblématique de Mario. Il offre aux joueurs une expérience immersive, mettant en avant les défis, l'exploration et le sauvetage de la princesse Peach. Notre jeu est le résultat d'un travail acharné, de la créativité et de la passion que nous avons investies dans ce projet.

En conclusion, le développement de ce jeu Mario a été une aventure passionnante, nous laissant avec un sentiment d'accomplissement et une passion renouvelée pour la création de jeux. Nous sommes fiers du résultat final et espérons que les joueurs apprécieront l'expérience que nous avons créée.

# Référence


- https://www.mapeditor.org/
- https://www.python.org/
- https://www.pygame.org/
- https://www.libsdl.org/
- https://astah.net/products/astah-uml/
- https://code.visualstudio.com/