



Scraft - Deuxième rapport de soutenance

Vincent AUPEST, Mathis BERTRAND

Felix MUHLKE, Louis VIDELIER

Table des matières

Introduction	4
1 Descriptif du projet	5
1.1 Origine	5
1.2 Nature du projet	6
1.3 Fiche du projet	6
1.4 État de l'art	7
2 Distribution des tâches	8
2.1 Subdivisions réalisées et nouvelles catégories	8
3 Découpage du projet	10
4 Planification et avancement	12
4.0.1 Explications :	12
4.1 Entités Objectif	13
4.1.1 Description	13
4.1.2 Nouvelles fonctionnalités	13
4.1.3 Implémentation	13
4.1.4 Auteur	14
4.2 Construction	14
4.2.1 Description	14
4.2.2 Implémentation	14
4.2.3 Auteur	14
4.3 Items	15
4.3.1 Description du problème	15
4.3.2 Implémentation	15
4.3.3 Problématiques liés à l'implémentation	15
4.3.4 Classes	16
4.3.5 Auteur	16
4.4 Props	16
4.4.1 Description	16
4.4.2 Implémentation	16
4.4.3 Démonstration des CropProps	17
4.4.4 Auteur	17
4.5 Carte	18

4.5.1	Description	18
4.5.2	Implémentation	18
4.5.3	Auteur	18
4.6	Menus/Accès Multijoueur	19
4.6.1	Description	19
4.6.2	Implémentation	19
4.6.3	Auteur	19
4.7	Site Web	20
4.7.1	Description	20
4.7.2	Implémentation	20
4.7.3	Aperçu	20
4.7.4	Auteur	20
4.8	Player	21
4.8.1	Description	21
4.8.2	Nouvelles fonctionnalités	21
4.8.3	Implémentation	22
4.8.4	Auteur	22
4.9	Inventaire	22
4.9.1	Description	22
4.9.2	Nouvelles fonctionnalités	23
4.9.3	Implémentation	25
4.9.4	Auteur	25
4.10	Craft	26
4.10.1	Description	26
4.10.2	Avancement	26
4.10.3	Auteur	26
5	Structure du projet	27
5.1	Fonctionnel	27
5.2	Méthodologique et Technologique	28
5.2.1	Executable	28
5.2.2	Gestion du projet	29
5.2.3	Communication	29
5.3	Opérationnel	30
Conclusion		30

Introduction

L'objectif de ce rapport est de présenter l'état d'avancement de notre projet Scraft, ainsi que de détailler étape par étape le processus de sa conception.

Tout d'abord, il est important de mentionner que certaines sous-étapes de notre planning ont dû être réajustées afin de permettre une progression plus cohérente du projet.

Au cours de cette période, nous avons mis l'accent sur la consolidation de notre code en revenant aux fondamentaux de notre projet. En effet, nous avons dû revoir certains aspects de notre code, car nous avons rencontré des problèmes.

Par ailleurs, au cours de ces deux derniers mois, nous avons réussi à avancer comme prévu sur les tâches que nous avions planifiées. Malgré les contraintes auxquelles nous avons dû faire face, nous avons su maintenir notre projet à jour en augmentant notre charge de travail.

Dans le cadre de ce rapport, nous vous présentons une description complète du projet Scraft, en mettant en avant ses caractéristiques fonctionnelles, opérationnelles et méthodologiques. Nous commencerons par vous faire une description des fonctions clef du jeu, puis nous vous décrirons les aspects plus techniques et organisationnels. Enfin, nous expliquerons comment le jeu a été développé en mettant en évidence les différentes étapes de la planification. Nous espérons que ce rapport de soutenance permettra une meilleure compréhension du projet Scraft et de son développement.

1 Descriptif du projet

1.1 Origine

La création du groupe s'est faite rapidement, puisque dès l'annonce du début du projet S2 le groupe comptait déjà trois membres actifs. Mathis étant le quatrième membre essentiel au bon fonctionnement du projet, le groupe Megawaves est ainsi né.

Ensuite, l'idée de faire un jeu vidéo a été une évidence dès les premières mises en commun du groupe. Nous nous sommes très rapidement mis d'accord sur la création d'un jeu de survie.

L'idée de Scraft est née après des heures de communications entre les membres de l'équipe, mais aussi à l'inspiration de jeux déjà existant comme Rimworld, ou encore Factorio.

- **Vincent AUPEST** : Ancien élève de terminale, avec la spécialité NSI, j'ai déjà effectué plusieurs "projets" de jeux vidéos avec le langage de programmation Python dans le cadre de ma scolarité. Aujourd'hui, je me sens prêt à assurer ce projet ainsi que sa direction.
- **Mathis BERTRAND** : Ayant toujours été proche de l'univers des jeux vidéos, mon intérêt pour l'informatique s'est développé naturellement, mais sans jamais vraiment sans approcher. La spécialité NSI que j'ai suivi au lycée a directement été une révélation pour moi. Dès mes premiers cours de NSI j'étais convaincu sur mon avenir. N'ayant jamais conçu de petits jeux ou de projets d'une telle envergure. Ce projet sera l'occasion de donner une vraie application directe à mon travail. Mes espérances pour notre jeu sont d'en apprendre énormément sur le développement globale d'un projet.
- **Felix MUHLKE** : N'ayant pas fait NSI (ou option Informatique dans mon lycée), je n'ai pas la même assurance que certains de mes camarades dans la création d'un jeu vidéo. Et pourtant, depuis que j'ai découvert l'informatique par mes propres moyens j'ai toujours eu envie d'en faire un. Le faire seul et sans assistance m'avait un peu refroidi, je suis donc impatient à l'idée de commencer ce projet en groupe. J'ai d'ailleurs déjà pu constater le plaisir de faire un projet en groupe lors de l'élaboration du concept de notre jeu.

- **Louis VIDELIER** : Venant d'un petit lycée de province, j'ai toujours aimé ce qui était en lien avec la programmation et l'informatique au cours de ma scolarité (Scratch au collège, Python en seconde et au cours de mes deux ans de NSI). Grand amateur de programmation orientée objet, je me sens prêt à relever les défis de ce projet (Réseau, API avec Unity, etc.) et prendre mes premières marques dans le monde du *gamedev*.

1.2 Nature du projet

De par notre manque d'expérience à l'origine, nous avons préféré nous lancer sur un concept simple mais qui peut se complexifier assez rapidement. C'est pourquoi l'idée du jeu de survie en TPS¹ nous est venu assez rapidement. Le jeu Scraft est donc un jeu de survie coopératif dans lequel les joueurs doivent survivre sur une île, percutée par une météorite, en recueillant des ressources, en cultivant, en construisant et en se défendant contre des vagues d'ennemis. Les joueurs devront coopérer pour survivre et explorer la carte pour trouver de nouvelles ressources et technologies.

Le but du jeu est de survivre le plus longtemps possible, tout en se développant. Le joueur devra gérer sa faim, collecter des ressources et construire une base afin de survivre grâce à ses défenses.

Les joueurs seront plongés dans une carte pré-générée, mais avec une petite part d'aléatoire pour offrir une expérience différente à chaque partie. Scraft propose une expérience de jeu unique et passionnante pour les joueurs qui aiment la survie coopérative dans un environnement hostile.

1.3 Fiche du projet

- Genre : Multijoueur / TPS / PVE²
- Style : Action / Aventure / Survie
- Plateforme : PC Windows
- Moteur de jeu : Unity
- Nombre de joueurs : 1-4

1. Third Person Shooter : jeu à la troisième personne
2. Player Versus Environnement : Joueur contre environnement

1.4 État de l'art



(a) The Escapists



(b) Factorio



(c) Rimworld

Scraft s'inscrit dans la lignée des jeux de survie et de gestion de ressources, un genre qui a vu émerger plusieurs titres populaires au fil des années. Parmi les jeux qui nous ont inspiré, nous pouvons citer Rimworld(c) pour son gameplay, Factorio(b) pour son aspect simulation et sa gestion industrielle et The Escapists(a) pour son rendu visuel. Toutefois, Scraft se distingue de ces jeux grâce à ses particularités uniques, telles que son mode Endless¹, son style graphique distinctif et son gameplay conçu pour éviter les aspects négatifs des autres jeux de survie. Dans Scraft, les joueurs doivent survivre en utilisant des ressources limitées, en explorant leur environnement hostile et en développant leur technologie. Le nom "Scraft" fait référence à la fusion de "scrap" pour "ferraille" et "craft" pour "fabrication", reflétant l'importance de la collecte et de la transformation des ressources dans le jeu.

1. mode de jeu sans fin, l'objectif étant de survivre le plus longtemps

2 Distribution des tâches

2.1 Subdivisions réalisées et nouvelles catégories

Subdivisions

Animations → (Animations - Player) + (Animations - Entités)

Mise en avant → Site + Trailer

Suppression et ajout de catégories

∅ → Game Manager

Intelligence Artificielle → Entités

Génération de la Carte → Carte

La création du "Game Manager" permettra de superviser les appels des fonctions tels que "Jour & Nuit" ou "Objectif", il permettra un gameplay plus complet.

Nous avons également modifié la catégorie "Intelligence Artificielle" en "Entités" pour plus de clarté. "Entités" désignant l'ensemble des objets "vivants" non-joueurs.

<i>Tâches</i>	Louis	Felix	Mathis	Vincent
Player	A	I	R	I
Animations - Player		C	R	
Sprites				R
Items	R			C
Interface (UI)	I	I	A	R
Menus				R
Inventaire			R	
Craft		C	R	I
Props	R			C
Entités	C	R	C	C
Animations - Entités		R	C	
Carte	A			R
Objectif	C	R	A	
Son	R	I	I	I
Game Manager	R	C		
Jour & Nuit & Lumière			R	
Construction	C	R		I
Équilibrage			C	R
Sauvegarder/Charger	I	R		
Arbre des recherches	R	A		C
Site	R			C
Trailer				R

TABLE 1 – R : Responsable, A : Approbateur, C : Consulté, I : Informé

3 Découpage du projet

Le projet de création du jeu Scraft sera divisé en plusieurs étapes, chacune étant consacrée à une partie spécifique du jeu. Tout d'abord, nous avons travaillé sur la mise en place du joueur en implémentant ses mouvements, ses interactions et son lien avec l'inventaire. Pour rendre le jeu plus dynamique, nous avons ajoutés des animations. Tout ça lié au réseau grâce à Mirror.

Nous avons ajouté un système d'items se basant sur des instances C# contenant les informations du stack manipulé, et d'un ScriptableObject qui contient toutes les informations constantes de l'item. Nous utilisons un GameObject DroppedItem qui représente un item au sol.

De plus, nous avons ajouté une interface utilisateur pour l'inventaire du joueur, alliant les items au player et donc permettant par la suite de faire le système de création d'items (craft).

Nous avons également travaillé sur le design du jeu en créant des sprites et des textures pour les différents éléments du jeu.

Ensuite pour le bon fonctionnement du multijoueur, nous avons ajoutés un système de menus. Ces menus seront utilisés pour rejoindre ou créer une partie, mais aussi par la suite pour personnaliser notre joueur.

Pour ajouter de l'interactivité à la carte, nous avons implémenté des éléments tels que les cultures (crops) ou encore les arbres. Nous avons également implémenté le système des entités non-joueur, tel que les entités aggressives/neutres/ passives, de façon à ce qu'elles interagissent avec les joueurs et la carte en réseau.

Pour rendre l'immersion plus importante pour le joueur, nous avons conçu une carte pré-générée et complète, permettant l'exploration mais aussi la redécouverte, avec un système de hotspot¹, pour donner une part de pseudo-aléatoire et plus de variété à la carte.

Quand au GameManager, nous allons créer un script permettant la coor-

1. zone pouvant générer un lieu à forte attraction

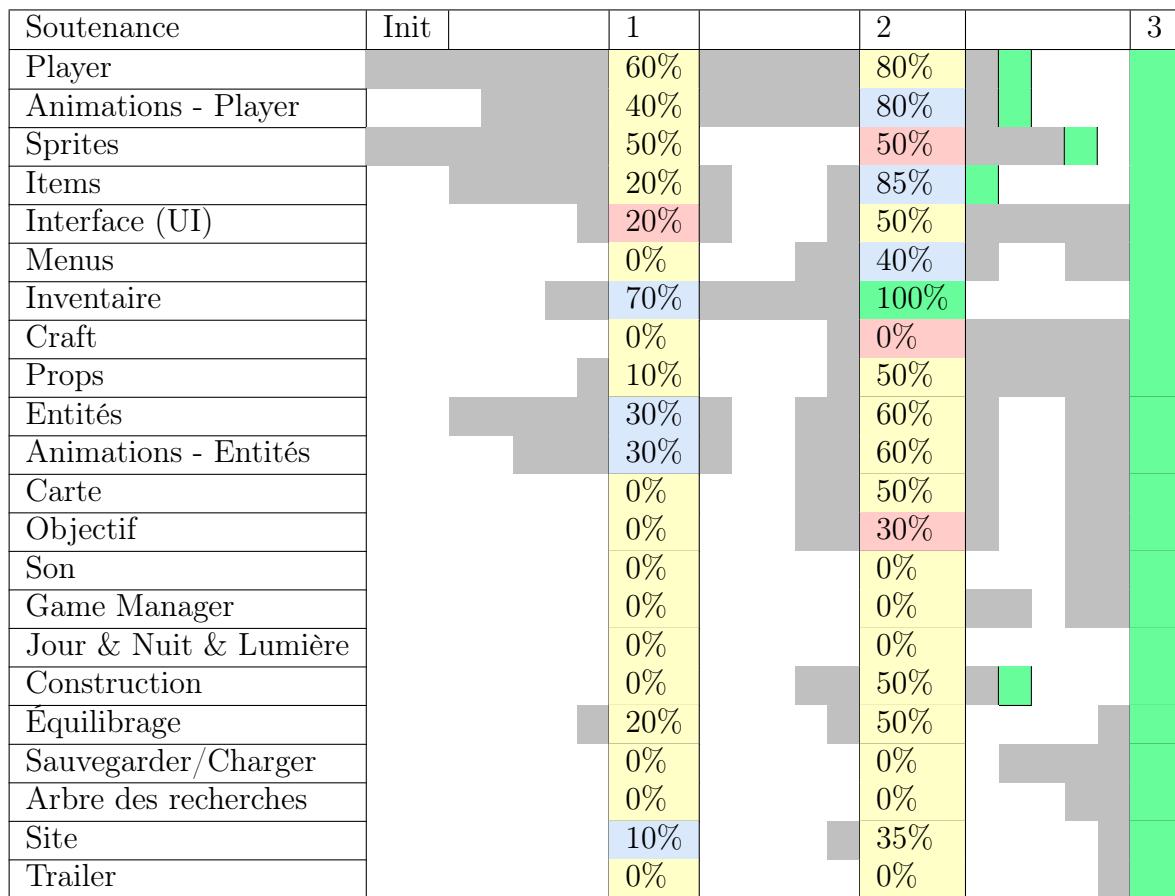
dination des scripts récurrents du projet.

L'ajout de ce GameManager permettra par exemple l'implémentation de l'objectif, avec l'apparition des ennemis et du score, ou encore l'ajout du cycle jour/nuit qui permettra lui a son tour l'ajout des jeux de lumières pour ajouter plus de réalisme.

Nous avons et allons encore implémenter le système de construction, qui demandera un alignement avec la grille du jeu, un système d'interaction avec l'inventaire pour le coût mais aussi un lien avec l'arbre de recherche que nous allons implémenté de manière a avoir un système de point de recherche permettant de débloquer de nouveau craft. Biensur ces deux fonctionnalités nécessiteront l'usage d'UI avancées.

Et pour finir, nous allons faciliter la sauvegarde et le chargement des données en mettant en place un système de persistance des données.

4 Planification et avancement



En cours A l'heure Retard Avance Fini

4.0.1 Explications :

Scraft a pris un léger retard sur plusieurs de ses avancements (Sprites et Objectif). Ce retard a été causé, d'une part, d'un non-besoin, nous n'avions pas besoin de designer de nouveaux sprites et d'autre part, d'un changement de base de notre code, qui aura été nécessaire pour résoudre des conflits.

Bien que cela ait entraîné des retards, cela nous permettra d'améliorer considérablement la qualité du jeu et donc de fournir une expérience de jeu plus stable et complète aux joueurs.

4.1 Entités Objectif

4.1.1 Description

Après avoir implémenté le déplacement des entités ennemis du joueur, nous nous sommes rendu compte que l'utilisation d'un système à partir de vecteurs ne permettait pas assez de modularité puisqu'il se concentrerait uniquement sur les joueurs. En effet, nous avons besoin, pour le bon fonctionnement de notre jeu, que les ennemis se dirigent par défaut vers le cœur de la base du joueur ainsi que vers le joueur si jamais celui-ci est plus proche. Or, ayant un système focalisé uniquement sur les joueurs, l'ajout du cœur de la base comme cible principale était bien plus complexe avec ce système de vecteurs. C'est pourquoi nous avons dû revoir encore une fois notre système de déplacement des entités. Ainsi, nous avons pu implémenter les entités ennemis et neutres avec une nouvelle méthode, tout en leur donnant la possibilité d'attaquer et de mourir. Les entités peuvent ainsi se déplacer en groupe afin d'attaquer la base du joueur, ce qui constitue la base de notre jeu.

4.1.2 Nouvelles fonctionnalités

- Ajout de l'entité neutre
- Ajout de l'attaque des entités ennemis sur les joueurs et constructions
- Ajout de la mort des entités

4.1.3 Implémentation

Pour cela, nous avons utilisé le package NavMesh. En effet, en définissant deux GameObjects, l'un pour les surfaces sur lesquelles nos entités peuvent se déplacer et l'autre pour les obstacles, il nous permet de définir une zone dans laquelle nos entités peuvent se déplacer. Ainsi avec le composant NavMeshAgent, nous pouvons donner une destination à notre entité qui pourra s'y déplacer le plus efficacement possible grâce à notre NavMesh. Les entités ennemis ont donc à tout moment accès à la position du cœur de la base, vers laquelle ils se dirigent s'il n'y a pas de joueurs autour d'eux. Les entités neutres, quant à elles, ont accès à leur position de départ et lorsqu'il n'y a pas

de joueurs autour d'eux, se déplacent de manière aléatoire dans un certain rayon autour de leur position de départ.

4.1.4 Auteur

MUHLKE Félix - Responsable de toute l'implémentation du déplacement des entités

4.2 Construction

4.2.1 Description

L'objectif de la construction est de pouvoir poser des murs et des portes afin de se protéger des ennemis. Les ennemis pourront bien sûr détruire ces constructions afin d'accéder au joueur ou au cœur de la base. Ces constructions sont également dotées d'une durabilité plus ou moins élevée en fonction du matériau dans lequel elles sont construites afin de résister aux attaques.

4.2.2 Implémentation

Pour construire, il suffit donc de placer des props de murs et de portes. Ainsi, l'algorithme vérifie s'il peut placer le prop en regardant s'il sera en collision avec un autre objet statique, si oui, il ne le place pas, sinon il place le prop sur la map. La sélection de l'emplacement futur du prop est faite grâce à la grille de Unity ainsi que, graphiquement par le joueur, grâce à un curseur visuel. De plus, ces murs et portes sont dotés de fonctions permettant de réduire leur durabilité s'ils sont frappés par les joueurs ou les ennemis.

4.2.3 Auteur

MUHLKE Félix - Responsable de la construction

4.3 Items

4.3.1 Description du problème

Au cours de cette période de développement, nous nous sommes rendu compte que l'implémentation des items telle que présentée lors de notre première soutenance était problématique.

En effet, les items n'étaient pas indépendants les uns des autres, même au sein du même type. Par exemple, avec notre ancienne implémentation, chaque **Pioche en fer** ne pouvait pas avoir une durabilité indépendante des autres en jeu. Tous les items au sein du même type devaient avoir les mêmes valeurs pour chaque attribut au même moment. De plus, les items étant des MonoBehaviour, il n'était pas possible pour un inventaire de posséder des items sans les faire disparaître de la scène artificiellement, ce qui n'était pas souhaitable.

C'est pourquoi nous avons dû retravailler les items de la façon suivante.

4.3.2 Implémentation

Afin de résoudre ces problèmes, il a été décidé de déplacer les attributs différents d'une instance à l'autre dans la classe AItem, et de ne plus avoir d'héritage entre ces classes. Les items au sol sont donc désormais représentés sous un autre préfab : DroppedItem.

4.3.3 Problématiques liés à l'implémentation

Cette modification n'a pas été sans difficulté, car les classes abstraites ne sont pas, par défaut, *Serializable*¹ par Unity. Afin de régler ce problème, il nous a fallu ajouter un script C# supplémentaire (modifiant le comportement de l'éditeur Unity) afin de récupérer, la liste des sous-classes non-abstraites de la classe utilisée dans le type.

Ce nouveau script est présent, dans le dossier *Scripts*, sous le nom *Subclass-Picker.cs*.

1. à comprendre, visible dans l'éditeur

4.3.4 Classes

Les diagrammes UML représentants les relations entre les différentes classes nécessaires pour l'implémentation du système d'items sont disponible au format *.png* ci-joints.

[Items]
[ItemData]
[Namespace Complet]

4.3.5 Auteur

VIDELIER Louis - Responsable des reconsidérations de la hiérarchie des classes des items. Extension de la hiérarchie.

4.4 Props

4.4.1 Description

Les Props représentent les éléments matériels présents dans l'univers des joueurs. Il peut s'agir des murs d'une base, des plantes, des rochers, etc. Afin de pouvoir collecter des ressources dans son environnement, se développer et survivre ; les joueurs doivent pouvoir être en mesure de se récupérer des ressources à partir de ces Props afin de se pouvoir développer leur base et ses défenses face aux vagues d'ennemis.

Il existe une catégorie de Props pour chaque interaction possible entre le joueur et le Prop. Actuellement, il existe un Prop pour les plantes avec lesquelles le joueur peut interagir et récolter (telles que le maïs, le blé, les tomates, etc.). Il existe aussi un Prop pour les éléments de l'environnement que le joueur peut détruire pour récupérer des ressources (tels que les minerais, les arbres, etc.).

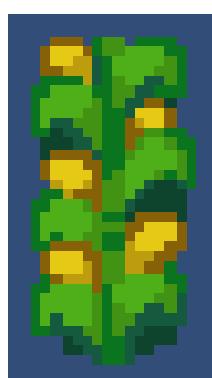
4.4.2 Implémentation

Les Props n'ayant pas les mêmes contraintes que les items (les Props n'ont pas vocation à pouvoir être récupérés directement par les joueurs lorsque ceux-ci sont détruits), ceux-ci ont donc été implémentés de manière équivalente à l'ancienne implémentation des items, au travers d'un MonoBehaviour

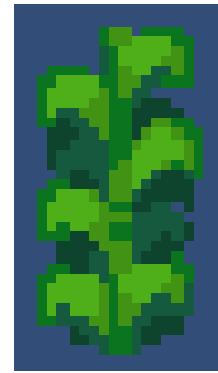
qui contiendra les attributs de l'instance et d'un ScriptableObject pour les données du type de Prop.

4.4.3 Démonstration des CropProps

Les CropProps sont une catégorie de Prop correspondant aux plantes que le joueur peut cultiver. Le joueur peut récupérer les pousses des plantes sous forme d'item avec une houe.



(a) Un plan de maïs, mature.



(b) Un plan de maïs, récupéré par un joueur.

4.4.4 Auteur

VIDELIER Louis - Responsable de l'implémentation et de la création la hiérarchie des classes nécessaire.

4.5 Carte

4.5.1 Description

Nous avons conçu une carte de dimensions 500x500 tiles, où les joueurs peuvent évoluer dans des biomes¹ immersifs. Pour susciter une impression de redécouverte, nous avons inclus un système de hotspots² permettant de générer des lieux d'une grande attraction.

En ce qui concerne la conception de la carte, nous avons eu recours à des algorithmes pour générer et affiner la carte, notamment pour créer des transitions fluides entre l'océan et la côte ou entre deux biomes différents. Nous avons abandonné l'idée de générer entièrement la carte, car cela s'avère trop coûteux en ressources. Lorsqu'il s'agit de cartes de dimensions 500x500 avec plusieurs couches, cela peut affecter négativement l'expérience de jeu du joueur en raison des limitations de son ordinateur.

4.5.2 Implémentation

Pour générer la carte, nous avons suivi une méthode en deux étapes. Tout d'abord, nous avons créé une version de base de la carte en dessinant un croquis dépourvu de détails afin de dessiner les formes et de découper les biomes. Ensuite, nous avons utilisé un package³ de Unity permettant de rajouter des règles entre les tiles⁴ afin d'avoir des transitions entre les biomes plus propres. Cette approche nous a permis d'économiser un temps considérable.

En ce qui concerne le système de hotspots, nous avons prédefini plusieurs zones sur la carte qui sont capables de faire apparaître des GameObjects représentant des zones à fort intérêt pour le joueur.

4.5.3 Auteur

AUPEST Vincent - Designer/Implémentation

-
1. zone géographique possédant des caractéristiques spéciales
 2. zone pouvant générer un lieu à forte attraction
 3. Rule Tile
 4. case de la carte

4.6 Menus/Accès Multijoueur

4.6.1 Description

Nous avons créé des menus permettant aux joueurs de se nommer avant d'accéder aux options de host¹ ou de join². Le système de multijoueur repose sur Mirror, ce qui permet à plusieurs joueurs de se connecter en utilisant l'adresse IP de l'host.

Les menus sont essentiels pour permettre aux joueurs de transmettre toutes les informations nécessaires lors de la connexion, notamment les identifiants des autres joueurs pour permettre les interactions entre eux plus tard, c'est pourquoi nous avons décidé de faire une salle d'attente avant le lancement de la partie.

Cependant, nous n'avons qu'à ce jour implémenté la partie fonctionnelle et interface utilisateur, au gré de l'affichage. Les menus fonctionnent, mais ne sont pas encore habillés.

4.6.2 Implémentation

Pour l'implémentation, j'ai du Override une partie des fonctions préfournies par le NetworkManager présent dans Mirror. Les fonctions présentent de base sur Mirror ne permettaient pas l'accès à un lobby et l'envoie des informations que nous voulions, c'est pourquoi nous avons dû modifier directement les fichiers natifs au module. Ensuite, il ne nous restait plus qu'à faire la partie UI, et bien sûr l'HUD³, Pour finir l'implémentation, nous avons dû passer sur Unity et s'occuper de la partie UI⁴.

4.6.3 Auteur

AUPEST Vincent - Responsable Menus/Interface

-
1. accueillir
 2. rejoindre
 3. Head up display : affichage
 4. interface utilisateur

4.7 Site Web

4.7.1 Description

Le Site web est une partie importante de la communication autour du jeu Scraft. Souhaitant conserver notre idée autour de laquelle "Megawaves Software" est un studio factice, la première version de notre site Web représente le site du studio Megawave Software.

4.7.2 Implémentation

Le site web a été implémenté au travers des langages HTML-5 et CSS-3. Le langage JavaScript pourrait être utilisé dans le futur pour les éléments dynamiques de la page.

4.7.3 Aperçu

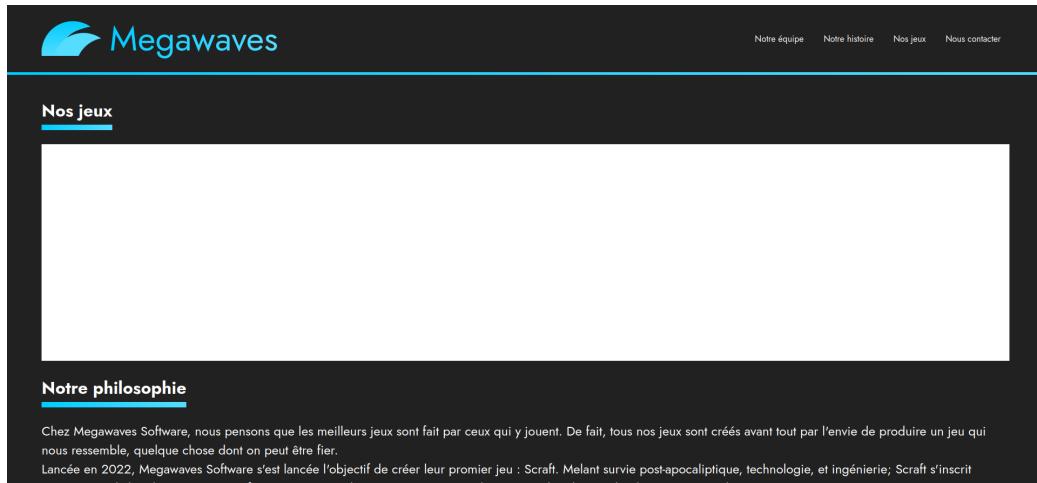


FIGURE 3 – Capture d'écran de la page d'accueil.

4.7.4 Auteur

VIDELIER Louis - Responsable site Web.

4.8 Player

4.8.1 Description

Chaque utilisateur contrôle son propre personnage. Il est ici question du joueur tel que le personnage, capable de se déplacer et d'utiliser les objets qu'il a à sa disposition. Le joueur à une animation pour le rendre vivant et une animation pour déclencher une attaque. Il ne peut tenir et utiliser qu'un item à la fois. L'utilisateur déplace son joueur avec son clavier et vise avec sa souris pour interagir avec son environnement. Le joueur regarde toujours vers le pointeur de la souris et ainsi peut attaquer à 360° autour de lui. Le joueur tient en main l'item sélectionné dans sa barre d'action, par défaut si l'emplacement sélectionné est vide le joueur porte un gant de boxe. De même, l'item tenu est toujours en direction du pointeur de la souris pour bien identifier la direction de l'attaque.

4.8.2 Nouvelles fonctionnalités

- Ajout de la mort du joueur. Lorsque le joueur n'a plus de vie une animation se déclenche et le joueur se couche au sol, il ne peut plus bouger, attaquer et porter d'items. Il pourra par la suite se faire réanimer par les autres utilisateurs. Si le joueur n'est pas réanimé au bout d'un certain temps, le joueur réapparaît sur la carte et tout son équipement est jeté au sol sur le lieu de sa mort.
- Ajout de l'inventaire au player. L'inventaire est maintenant relié au player, ainsi chaque player a son inventaire avec lequel il peut interagir en utilisant toutes les fonctionnalités de l'inventaire.
- Liaison de la barre d'action aux inputs du player. Le player a accès sans interruption à la barre d'action d'inventaire. Grâce à la molette de sa souris, il peut faire défiler l'item sélectionné dans la barre d'action ce qui actualisera l'item pris en main.
- Ajout de l'action "Attaque". En appuyant sur le click gauche de sa souris, le joueur attaque vers le pointeur de sa souris, la portée, les dégâts et le temps d'attente entre chaque attaque sont définis par l'item pris en main par le player.
- Ajout de l'action "Utilisation". En appuyant sur le clic droit de sa souris le joueur va pouvoir utiliser l'item qu'il tient en main. Seul les items consommables et les armes à distance sont utilisables.
- Ajout de l'action "Interagir". En appuyant sur F, le joueur peut inter-

agir avec l'objet le plus proche de lui. Ce comportement est différent selon l'objet.

4.8.3 Implémentation

Pour la liaison entre l'inventaire et le player, l'intermédiaire se fait par un "InventoryManager" qui est un script accessible par le player contenant toutes les informations et principales méthodes de l'inventaire. Ainsi, le joueur peut ajouter des items, connaître l'item qu'il tient en main et savoir le nombre précis de chaque item qu'il contient (ce qui sera utile pour l'implémentation des crafts). Pour l'attaque le player détecte les entités devant lui sur un cercle de rayon défini par l'item qu'il porte, ainsi, il leur inflige des dégâts. L'avantage de cette implémentation plutôt que de faire des raycasts est que nous pouvons frapper plusieurs ennemis en même temps ce qui rendra les combats plus dynamiques. Pour ce qui est de l'interaction, le principe est très similaire à l'attaque sauf que le cercle de détection est au centre du player, le player peut interagir avec des items aux sols en les ramassant, et avec des "Props" comme des cultures pour les récupérer. Enfin pour l'utilisation des items, le player peut consommer des aliments pour régénérer sa vie et tirer des projectiles avec des armes à distance.

4.8.4 Auteur

BERTRAND Mathis - Responsable de la fonctionnalité Player.

4.9 Inventaire

4.9.1 Description

Chaque joueur possède son propre inventaire qui lui permet d'équiper, stocker, jeter, détruire et obtenir des informations sur les items qu'il porte sur lui. L'inventaire est une interface graphique interactive qui s'ouvre et se ferme. L'inventaire est décomposé en 5 parties.

- 1 espace destiné à détruire les items non voulus par le joueur.
- 1 espace destiné à afficher les informations de l'item à inspecter.
- 4 emplacements réservés pour l'armure du joueur.
- 30 emplacements destinés au stockage interne des items portés par le joueur.

- 10 emplacements supplémentaires destinés au stockage, mais accessibles directement au joueur, même l'inventaire fermé. Cette section de l'inventaire est appelée "barre d'action", ou *hot-bar*.

Certains items peuvent être regroupés ensemble en groupes prédéfinis (chaque item définit la quantité maximum d'élément avec lequel il forme un groupe), en général 100. L'inventaire permet aussi de subdiviser des groupes d'items en sous-groupes de taille inférieures.

4.9.2 Nouvelles fonctionnalités

De nouvelles fonctionnalités ont été ajoutées à l'inventaire, celui-ci étant très primitif à l'époque, l'intégralité des fonctionnalités principales sont désormais ajoutés.

- Ajout de la partie de l'interface graphique réservé à l'armure équipée par le joueur. L'inventaire dispose maintenant de quatre emplacements dédiés aux différentes parties de l'armure : casque, plastron, pantalon et bottes.
- Ajout de la partie de l'interface graphique décrivant les informations des items.
Lorsque le joueur clique sur un item dans notre inventaire, cette interface est mise à jour et affiche le nom de l'item, sa description ainsi que ses statistiques (dégâts infligés, durabilité, réduction des dégâts subis...).
- Ajout d'un emplacement dans l'interface graphique permettant de détruire des items. Lorsque l'on glisse un item dessus, il est simplement détruit.
- Ajout du "Drop" des items. Lorsque le joueur glisse un item en dehors de l'inventaire, cet item est supprimé de l'inventaire et jeté au sol.
- Ajout de la division des items regroupables. Le player peut maintenant diviser un "stack"¹ d'items regroupables en plusieurs sous-groupes de taille inférieure. Lorsqu'un joueur glisse un stack au-dessus d'un emplacement libre ou occupé par un item de même type, les items sont déplacés un par un du stack "flottant" vers le stack cible. Cela permet au joueur de pouvoir diviser ses items pour mieux les manipuler, comme, par exemple pouvoir jeter au sol un nombre précis d'items.

1. Une pile (aussi appelé groupe) d'items regroupant un lot de mêmes items.



FIGURE 4 – Interface de l'inventaire du joueur

4.9.3 Implémentation

L'affichage de l'inventaire est réalisé avec l'interface graphique de Unity (UI). L'inventaire est un tableau composé de cases. Ces cases ont pour rôle de stocker l'information de l'item qu'il contient et de l'afficher grâce à une image et au Sprite de l'item.

L'inventaire a besoin de certains inputs de l'utilisateur comme la position de la souris, l'entrée du clic gauche et celle du clic droit. Le clic gauche permet de sélectionner et de déplacer un item tandis que le clic droit permet de diviser les groupes items en sous-groupes (comme explicité précédemment).

L'inventaire, une fois affiché, utilise un système de "Drag and Drop"¹ pour déplacer à l'aide de la position de la souris les différents items dans la grille de l'inventaire. Le joueur "attrape" un item sur une case, et il glisse sa souris vers une nouvelle case de l'inventaire, et le joueur le "relâche". Ces fonctionnalités sont implémentées en utilisant des interfaces Unity qui nous permettent d'obtenir les informations du curseur et de l'item sélectionné.

4.9.4 Auteur

BERTRAND Mathis - Responsable de la fonctionnalité Inventaire.

1. Glisser-Déposer

4.10 Craft

4.10.1 Description

Chaque joueur peut fabriquer des items pour mieux se protéger et évoluer. Le joueur peut fabriquer des armes, munitions, armures, outils et consommables à l'aide des items, plus précisément, les "Products"(items récoltables sur la map permettant uniquement de fabriquer d'autres items). Pour fabriquer le joueur pourra afficher une interface graphique décrivant les items qu'il est en mesure de crafter. Via cette interface graphique les recettes des crafts seront affichées et le joueur pourra directement voir ce qu'il lui faut et ce qui lui manque. Lorsque le joueur voudra fabriquer un item il n'aura qu'à sélectionner l'item voulu et appuyer sur un bouton de validation. S'il contient les ressources nécessaires, celles-ci seront retirées de l'inventaire et l'item fabriqué sera ajouté.

4.10.2 Avancement

À cause de la révision des items qui a été faites, l'inventaire a été retardé et la fonctionnalité craft n'a pas commencé. En revanche, plusieurs méthodes permettant d'obtenir des informations sur l'inventaire ont déjà été implémentées pour faciliter l'implémentation du craft.

4.10.3 Auteur

BERTRAND Mathis - Responsable de la fonctionnalité Craft.

5 Structure du projet

5.1 Fonctionnel

Cette partie évoque l'intégralité des fonctionnalités qui seront présentes dans Scraft.

- Situation initiale : Les différents membres apparaissent à proximité les uns des autres.
- Création d'un arbre des connaissances communs entre les joueurs.
- Système multijoueur en *Peer-to-Peer*¹ (Sur Internet ; ou en LAN²) pour pouvoir jouer directement, sans l'intermédiaire d'un serveur. Chaque joueur peut héberger le jeu en local et ainsi, jouer seul.
- Système d'items qui peuvent être utilisés par les joueurs pour interagir avec leur environnement. Ainsi qu'un système de besoins de nourriture.
- Écosystème de machines, d'électricité entre les machines qui en produisent, les machines qui en consomment, etc.
- Système de génération d'une carte (par région, biome, puis hotspot)
- Système d'inventaire, de remise à zéro lorsque le joueur meurt.
- Des vagues d'ennemis régulières. Ainsi que des entités déjà présentent sur la carte.
- Une configuration des paramètres personnalisés.

1. L'architecture *Peer-to-Peer*, est une architecture en réseau où les différents clients échangent entre eux, sans avoir besoin d'un serveur central. Dans le monde du jeu vidéo, un jeu en *Peer-to-Peer* (abrégé en P2P) est opposé au modèle CGS (*Central Game-Server*), où les joueurs doivent communiquer au travers d'un serveur hébergé par l'éditeur.

2. *Local-Access-Network*, soit le réseau local.

5.2 Méthodologique et Technologique

Au cours du développement de Scraft, nous utiliserons différents logiciels pour concevoir le jeu, gérer sa production, générer de la documentation, ou encore créer l'identité graphique du jeu.

5.2.1 Executable

Pour concevoir le programme du jeu (le fichier exécutable, qui permettra aux joueurs de jouer au jeu), nous utiliserons les technologies suivantes ;

Unity - Moteur de jeu



Le moteur Unity nous permettra de pouvoir directement travailler sur l'environnement de notre jeu, plutôt que sur l'affichage, la physique, etc. Il s'agit d'un moteur de jeu.

C# - Langage de programmation



Le langage de programmation C# est un langage de programmation Orienté Objet de haut niveau utilisé par Unity pour le comportement des objets du jeu.

Rider - IDE



L'IDE Rider est une environnement de développement créé par JetBrains et adapté pour le langage de programmation C#, utilisé par Unity.

5.2.2 Gestion du projet

Pour gérer le projet en lui même, nous utiliserons les technologies suivantes ;

Git - Outil de versionnage



Git est un outil développé par Linus Torvald (plus connu comme étant le créateur du noyau Linux), il permet de gérer le développement d'un logiciel ou d'un projet de manière incrémentale sous la forme de "repos" pour stocker les différentes versions, avec un modèle client-serveur.

GitHub - Host Git et Organisation



GitHub est une plateforme rachetée par Microsoft qui permet la gestion de projet avec Git, accompagné une surcouche pour gérer les membres d'un projet au sein d'une organisation.

5.2.3 Communication

Afin de communiquer entre nous ou pour concevoir des rapports concernant l'avancée du projet, nous utiliserons les technologies suivantes ;

Discord - Tchat en ligne



Discord est une plateforme de communication en ligne ouverte à tous où les utilisateurs se retrouvent au sein de groupes ou de serveurs pour échanger autour de sujet communs. Nous l'utiliserons pour échanger lors des phases de développement du jeu. (3230 messages au total)

Overleaf - Rédaction L^AT_EX en équipe



Overleaf est une plateforme que nous utiliserons pour rédiger nos rapports et autres documents au format L^AT_EX.

5.3 Opérationnel

Comme nous comptons développer notre multijoueur en Peer-To-Peer, nous n'aurons pas besoin de louer un serveur. Nous aurons cependant besoin d'un hébergeur de sites web ce qui représente environ 3€/mois.

Conclusion

Pour conclure, nous sommes fiers de pouvoir présenter notre avancement sur notre jeu Scraft pour cette deuxième soutenance. Nous sommes heureux de constater que malgré certaines difficultés handicapantes, nous avons réussi à maintenir une progression constante et su rebondir sur celles-ci et finir par améliorer notre jeu dans son ensemble afin de respecter au mieux nos engagements actuels et futurs.

Globalement, nous sommes fiers d'avoir pu collaborer plus linéairement afin de produire un écosystème plus riche au sein de notre jeu, plus robuste et en mesure de s'adapter à nos contraintes futures. Plus particulièrement, les échanges au travers des outils de versionnages ont été plus simples en raison d'une maîtrise accrue de nos logiciels, et ce, impliquant une production logicielle accrue et de meilleure qualité.

Nous sommes conscients que nous avons encore une charge de travail importante afin d'atteindre les objectifs que nous nous sommes fixés pour la dernière soutenance. Nous allons donc continuer à travailler dur et à nous concentrer sur les tâches prioritaires afin d'assurer la livraison d'un produit final de qualité. Nous sommes également convaincus que les défis que nous avons rencontrés jusqu'à présent ne sont que des étapes normales dans le processus de développement de notre projet, et que nous sommes en mesure de les surmonter grâce à notre collaboration et à notre détermination. Nous sommes confiants quant à l'avenir de notre projet, et sommes impatients de voir les résultats de notre travail au fur et à mesure de notre progression.