



# Scraft - Rapport de projet

Vincent AUPEST, Mathis BERTRAND

Felix MUHLKE, Louis VIDELIER

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	Origine . . . . .	4
1.2	Nature du projet . . . . .	5
1.3	Contexte de Scraft . . . . .	5
1.4	Objectifs . . . . .	7
1.5	Fiche du projet . . . . .	7
1.6	État de l'art . . . . .	7
<b>2</b>	<b>Distribution des tâches</b>	<b>9</b>
2.1	Ré-attribution des tâches . . . . .	9
<b>3</b>	<b>Rappel du projet</b>	<b>10</b>
3.1	Cahier des charges/Produit initial . . . . .	10
<b>4</b>	<b>Découpage du projet</b>	<b>10</b>
<b>5</b>	<b>Rappel des avancées</b>	<b>11</b>
5.1	Soutenance 1 . . . . .	11
5.2	Soutenance 2 . . . . .	12
<b>6</b>	<b>Avancement</b>	<b>13</b>
6.1	Planification . . . . .	13
6.2	Explications . . . . .	14
6.3	Player . . . . .	14
6.4	Menu principal . . . . .	15
6.5	Menu Multijoueur . . . . .	16
6.6	Menu en jeu . . . . .	17
6.7	Interfaces . . . . .	17
6.8	Déplacement des entités & Navmesh . . . . .	19
6.9	Création d'entités . . . . .	20
6.10	Détails de la map . . . . .	22
6.11	Construction . . . . .	22
6.12	Craft . . . . .	24
6.13	Props . . . . .	25
6.14	Attaques/Popup/Recul . . . . .	26

6.15 Cycle jour/nuit et Heure . . . . .	28
6.16 Lumière . . . . .	29
6.17 Energie / Sprint . . . . .	30
6.18 Spawner d'entités/Vague . . . . .	30
6.19 Affichage tête haute . . . . .	31
6.20 Personnalisation du Skin . . . . .	32
6.21 Sauvegarde des données . . . . .	34
6.22 Son . . . . .	34
6.23 Réglage du son . . . . .	36
6.24 Items . . . . .	37
6.25 Inventaire . . . . .	38
6.26 Assignations des touches . . . . .	39
6.27 Projectiles . . . . .	40
6.28 Fin de partie . . . . .	41
6.29 Machines - Générateurs Tourelles automatiques . . . . .	41
6.30 Réseau . . . . .	47
6.31 Équilibrage . . . . .	47
<b>7 Structure du projet</b>	<b>48</b>
7.2 Fonctionnel . . . . .	48
7.3 Operationnel . . . . .	50
<b>8 Site</b>	<b>50</b>
8.1 Introduction . . . . .	50
8.2 Responsive . . . . .	51
8.3 Contenu du site Web . . . . .	52
8.4 Évolution du site de Megawaves Software . . . . .	53
<b>9 Logo</b>	<b>53</b>
<b>10 Conclusion</b>	<b>54</b>
10.1 Produit final . . . . .	55
<b>11 Sources</b>	<b>56</b>
11.1 Forums . . . . .	56
11.2 Vidéos . . . . .	56
11.3 Chaînes . . . . .	56

# 1 Introduction

## 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.

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, dans lequel le joueur aurait pour but de défendre sa base.

Notre idée s'est ensuite précisée au fur-et-à-mesure du temps, grâce aux réunions et à l'avancement du projet.

L'idée finale de Scraft est née après des heures de communications entre les membres de l'équipe, mais aussi grâce à une forte inspiration à d'autres titres, comme Rimworld, ou encore Factorio.

Nous ne pouvons pas non plus négliger notre inspiration à Minecraft.

- **Vincent AUPEST** : Anciennement élève de terminale, avec la spécialité NSI, j'avais déjà effectué quelques projets sous python. Dans le cadre de ma scolarité j'ai même effectué d'autres projets de jeux vidéo. Aujourd'hui, je me sens capable d'assurer le projet Scraft, 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

Le titre Scraft s'inscrit dans la prestigieuse lignée des jeux de survie, un genre qui a donné naissance à certains des titres les plus emblématiques de l'histoire du jeu vidéo.

Dans le cas de Scraft, ce mode de jeu est combiné avec un mode "Endless"<sup>1</sup>, dans lequel les joueurs devront savoir s'adapter, de jour en jour, pour être capable de rester en vie.

Nous avons choisi de un style 2d "pixélisé", pour pouvoir plus nous focaliser sur ce qui nous tenait à cœur, le contenu du jeu. Nous avons fait le choix d'avoir un jeu qui en est apparence assez simple, mais qui cache de la technique.

## 1.3 Contexte de Scraft

C'est en octobre 2035 que le destin de l'humanité a soudainement changé. Une pluie de météorites d'une puissance inimaginable s'est écrasée sur Terre, ne laissant derrière elle pratiquement aucune trace de l'humanité. Vous et vos rares co-équipiers avez miraculeusement survécu à cette catastrophe, mais au prix d'une amnésie totale. Vous vous réveillez sur cette île désolée sans aucun

---

1. Endless : Sans fin

souvenir de qui vous êtes ni de ce qui s'est passé.

Dès les premiers instants, vous ressentez une présence surnaturelle qui plane sur l'île. La nuit tombée, des créatures abominables émergent des ténèbres, avides de détruire la météorite qui s'est écrasée récemment. Vous ne comprenez pas pourquoi cette météorite suscite autant d'attention et de convoitise, mais vous êtes convaincu qu'elle détient une force mystérieuse capable de changer le cours des événements.

Guidé par votre instinct de survie, vous devrez utiliser vos connaissances de base en matière de survie pour construire un équipement de fortune. Vous devrez vous défendre contre des vagues incessantes de monstres toujours plus violentes qui cherchent à détruire la météorite coûte que coûte. La création d'abris solides et de fortifications stratégiques deviennent essentielles pour résister à ces attaques incessantes. Chaque nuit qui passe révèle de nouveaux défis avec des ennemis de plus en plus puissants. Votre quête pour survivre vous conduira à explorer les environs dans l'objectif de trouver ressources rares, nécessaires pour développer un équipement plus avancé et des structures de défense solides.

Le mystère qui entoure la météorite et la présence étrange qui hante l'île vous pousse à chercher des réponses.

Dans ce contexte post-apocalyptique impitoyable, votre survie dépendra de votre ingéniosité, de votre capacité à travailler en équipe.

## 1.4 Objectifs

Dans "Scraft", votre objectif ultime est de protéger la météorite qui s'est écrasée sur l'île, chaque soir une vague des zombies toujours plus forte tentera de la détruire. Votre mission est de la défendre des attaques, tout en explorant l'île en quête de ressources pour vous développer.



FIGURE 1 – Météorite

## 1.5 Fiche du projet

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

## 1.6 État de l'art

Lors de la conception de Scraft, nous nous sommes inspirés de différents jeux. Nous pouvons noter une inspiration pour "The Escapist" (a) pour leurs déplacements et gestion du personnage. Nous nous sommes également inspirés de "Factorio" (b) et "Rimworld" (c) pour la gestion des ressources et des machineries. Le système de construction est également assez similaire à celui de "Rimworld". Pour finir cette liste exhaustive de nos inspirations, nous avons "Minecraft" (d). En effet, nous ne pouvons pas négliger notre inspiration pour "Minecraft". Que ce soit pour le système d'inventaire ou dans le

---

1. Player Versus Environnement : Joueur contre environnement



(a) The Escapists



(b) Factorio



(c) Rimworld



(d) Minecraft

FIGURE 2 – Inspirations

style graphique, nous nous sommes parfois inspirés de "Minecraft". Pour créditer cette inspiration, nous avons défini le skin par défaut aux couleurs de celui de Minecraft. Toutes fois, Scraft se distingue des autres jeux de survies grâce aux possibilités qu'il offre. Vous avez la possibilité de découvrir une carte unique, de fabriquer une base qui vous ressemble, ou même de vous épanouir dans la conception de machineries avancées.

## 2 Distribution des tâches

<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

### 2.1 Ré-attribution des tâches

Certaines tâches du tableau ont été changé durant cette période d’intersoutenance. En effet, toujours à cause d’inter-dépendance, nous avons dû ré-attribuer certains rôles.

Pour commencer, nous avons Félix qui s'est vu assigner à la conception des chunks de la carte, puisque le système de construction et de déplacement des

entités (features que Félix s'occupait) avait besoin d'un système de chunk<sup>1</sup> pour fonctionner sans coûter trop de ressources. En contre-partie, Félix s'est vu retirer le système de sauvegarde, pour lui laisser le temps de faire fonctionner et d'optimiser le système de chunks, vital au jeu.

De plus, une feature n'a pas été aboutie, par manque de temps et par soucis de complexité dans l'incorporation en multijoueur. Nous sommes au regret de vous annoncer qu'il n'y aura pas d'arbre de recherche.

Nous avons également tenu à créditer Vincent et Mathis pour s'être occupés de la rédaction du manuel d'utilisation.

## 3 Rappel du projet

### 3.1 Cahier des charges/Produit initial

Le premier jet de Scraft proposait l'idée d'un jeu dans lequel les joueurs devaient défendre le cœur de leur base contre des vagues ennemis. L'idée d'un cycle régulier pour les vagues y était mentionné, ainsi que l'idée d'un arbre de recherche permettant l'exploitation plus aboutie des ressources. Dans notre première vision de Scraft, l'industrie prenait une part importante du jeu. Dans notre première vision du jeu, nous voulions également que le multijoueur-coopératif prenne une place importante dans le gameplay. Le cahier des charges ne fixait pas vraiment d'autre limite au jeu, à l'exception du style graphique en "pixel" 32x32, qui lui avait déjà été mentionné.

## 4 Découpage du projet

Le projet de création de Scraft a été divisé en plusieurs étapes. Tout d'abord, nous avons travaillé sur la mise en place d'un joueur, clé pour commencer la suite du jeu. Petit à petit, nous avons ajouté des features en parallèle du joueur, jusqu'à aboutir à la version finale du jeu. Nous avons opté pour une implémentation en continue de la mise en réseau. Cette technique a été préférée pour éviter d'avoir à perdre du temps à la fin du projet pour tout rendre compatible avec Photon.

Encore une fois, à cause de l'inter-dépendance de certaines features, nous

---

1. Division de la carte en plusieurs parties, voir la section développée sur ce système pour plus d'informations.

avons dû travailler en collaboration afin d'obtenir des résultats cohérents. Pour créer Scraft, nous avons décomposé le projet en plusieurs grands groupes de fonctionnalité, cette décomposition élémentaire nous a permis de travailler en gardant en tête les objectifs. Cette technique de travail nous a permis d'avoir le résultat du projet en temps voulu, mais ne nous a malheureusement pas permis de présenter une version jouable du jeu à chaque soutenance. Nous n'étions capable que de présenter des features du jeu, car nous savions pertinemment que celles-ci étaient vitales à l'articulation du jeu.

## 5 Rappel des avancées

Ce rapport vous présente notre travail autour du projet Scraft. C'est pourquoi nous tenons à vous rappeler le travail qui a été effectué pendant les périodes précédentes.

### 5.1 Soutenance 1

Pour commencer la première soutenance, nous avons précisée certaines de nos idées, et ainsi fixé ce qu'allait être Scraft.

Lors de la première soutenance, nous avions déjà implémenté, une partie des scripts du joueur, une première version des items posables dit "Props" et des items en inventaire et pour finir, le comportement de certaines IA.

Les scripts du joueur concernaient ses déplacements, son inventaire, ses animations et sa mise en réseau. Quant aux IA, nous avions une version des entités agressifs, et de leur capacité à naviguer dans un environnement. En parallèle de ces features, nous avons dessiné une partie des sprites nécessaires à la suite du projet.

### Compte-rendu

Le travail de la soutenance 1 nous a permis d'avoir une version stable du joueur, avec une première mise en réseau, même si malheureusement via Mirror. Nous avions implémenté une partie des fonctions utiles aux joueurs, que ce soit le déplacement, le réseau, ou les animations. Une version de l'inventaire du joueur avait également été implémenté. Ensuite, nous avions implémenté

des versions pour les props et les items. Malheureusement le travail sur les props et les items n'a pas été retenu, car nous l'avons retravaillé peu après la soutenance 1. Cette soutenance nous avait permis de nous projeter dans notre projet, et même si elle ne nous aura pas permis d'implémenter le tier du projet, elle nous aura permis de nous fixer une idée de comment la suite allait se dérouler.

## 5.2 Soutenance 2

Lors de la deuxième soutenance, en plus de proposer de nouvelles features, nous avons retravaillé le fonctionnement d'une grande partie des scripts précédents.

En effet, nos implémentations des props, items et par la suite des IA sont devenues impertinentes.

Cette soutenance aura aussi permis de faire un compte-rendu sur les nouvelles features, qui sont : l'implémentation de nouveaux comportements pour les entités, la construction, les cultures, la carte, les menus et le multijoueur. Nous avions également proposé un premier jet du site web.

Encore une fois, cette seconde soutenance aura permis une clarification et une optimisation du jeu.

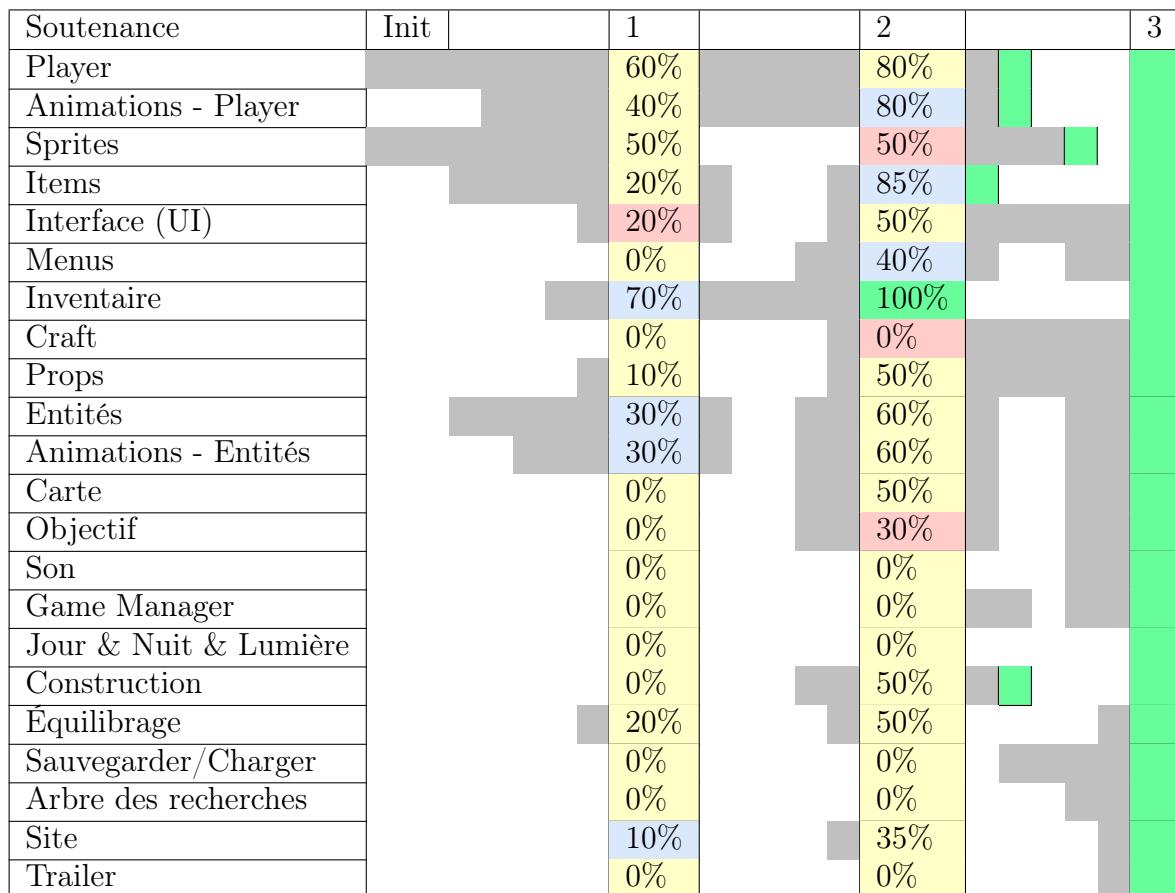
### Compte-rendu

A la suite de cette seconde soutenance, Scraft s'est vu inferer de nouvelles fonctionnalités. Nous avions touché de près ou de loin à pratiquement l'intégralité des features, ce qui nous a permis d'avoir une vision globale du projet. Le plus gros changement lié à la soutenance 2, c'est que nous avons décidé de migrer sur Photon. Cette migration nous a obligé à revoir certaines fonctionnalités. Cette soutenance nous aura également permis de présenter notre avancement sur l'inventaire, le craft, les props et les items dans leur version finale, et pour finir les cultures.

## 6 Avancement

Pour cette troisième et dernière soutenance, nous avons terminé d'implémenter les features qu'il nous manquait, et assemblé la totalité des features, qui étaient pour certaines encore séparées à la deuxième soutenance. Nous n'avons presque pas eu à revenir sur nos anciens scripts, à l'exception de la carte, qui a subi un retravail dans sa conception.

### 6.1 Planification



En cours A l'heure Retard Avance Fini

## 6.2 Explications

Durant cette période d'inter-soutenance, nous avons eu le temps d'implémenter toutes les fonctionnalités utiles au bon fonctionnement du jeu, à l'exception de l'arbre de recherche, que nous avons trouvé substituable.

Nous avons également pris le temps d'optimiser notre jeu, en nettoyant et simplifiant le code, mais aussi et surtout en implémentant un système de chunk qui permet d'éviter de charger et d'actualiser un endroit de la carte pour rien. De plus, la rédaction du rapport et du manuel d'utilisation étant assez "chronophage", nous tenons à créditer Mathis et Vincent pour leur rédaction.

## 6.3 Player

### 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.

### Nouvelles fonctionnalités

- Ajout d'une Popup verte indiquant les points de vie régénéré.
- Ajout de l'énergie.
- Ajout de la touche "courir"<sup>1</sup> pour courrir.
- Refonte du design de l'intégralité de l'ATH.
- Liaison du menu pause.
- Liaison du système de construction
- Liaison du système pour les machines. tem Ajout de la touche "Jeter"<sup>2</sup>

---

1. "maj gauche" par défaut

2. "A" par défaut

pour jeter des Items au sol.

## Auteurs

**BERTRAND Mathis**

### 6.4 Menu principal

#### Description

Précédemment, nous avions fait une version assez simpliste pour accéder au multijoueur. Cette précédente version ne servait qu'à tester le code en jeu, pour détecter d'éventuel problème.

Dorénavant, nous avons une version propre et épurée du menu principal. Celle-ci donne accès au joueur aux menus des Options (voir la section sur les Options pour plus d'informations), au multijoueur, ainsi qu'aux crédits.

#### Nouvelles fonctionnalités

- Ajout des menus options.
- Ajout des crédits.
- Ajout d'un lien vers le multijoueur.
- Ajout d'une interface à l'image de Scraft.

#### Implémentation

L'implémentation du menu principal a été fait en parallèle des autres menus, pour respecter une charte graphique globale.

Le travail sur le menu se résume essentiellement à de l'UI<sup>1</sup>, et permet la navigation entre les autres menus grâce à des boutons.

#### Auteur

**AUPEST Vincent** - Responsable des menus & UI

---

1. Interface Utilisateur

## 6.5 Menu Multijoueur

### Description

Les menus multijoueurs permettent aux joueurs de rejoindre directement une partie ou d'en créer une. Ils permettent également de lancer une partie en solo.

### Nouvelles fonctionnalités

- Retravail de l'accès au multijoueur.
- Ajout d'un mode solo.

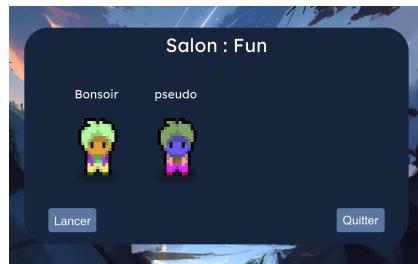


FIGURE 3 – Salon multijoueur

### Implémentation

L'implémentation du menu multijoueur a été assez similaire à celle du menu principal, à l'exception que celui-ci est connectée en réseau, c'est-à-dire que envoyez et recevez des informations, qui doivent d'être traitées pour faire fonctionner correctement votre jeu. L'implémentation en réseau de ces menus nous a permis de prendre en main l'interface Photon.

Cette partie a su montrer quelques défis, que ce soit le lien avec Photon, ou même les animations. Par exemple lorsque vous rejoignez un lobby multijoueur, vous verrez votre personnage et celui de vos amis présents dans le lobby s'animer avec leur propre skin.

Un autre défi aura été de bien lier les scènes entre elles, que ce soit avec le réseau, ou avec la persistance de données.

### Auteur

**AUPEST Vincent** - Responsable des menus & UI

## 6.6 Menu en jeu

### Description

Lorsque vous êtes dans une partie, vous pouvez appuyer sur "Echap" pour ouvrir le menu des options, qui vous permet de quitter le jeu, retourner au menu ou reprendre votre partie.

### Nouvelles fonctionnalités

- Ajout du menu en jeu.

### Implémentation

L'implémentation du menu en jeu a été assez simple, puisqu'elle est en local. La seule contrainte sur l'implémentation du menu en jeu, c'est que dès lorsque l'hôte qui, tous les props et entités disparaissent, car ils étaient stockés par l'hôte. Pour contrer ce problème, nous avons décidé que lorsque l'hôte quitte la partie, tous les joueurs sont renvoyés dans le menu principal.

### Auteur

**AUPEST Vincent** - Responsable des menus & UI

## 6.7 Interfaces

Au sein de Scraft, nous nous sommes rendus compte que bon nombre d'objets partageait des particularités similaires, sans pour autant avoir de liens direct entre eux. Par exemple, les objets doivent pouvoir être ramassables en interagissant avec eux, et les machines doivent pouvoir être activées / désactivées de manière identique. Nous avons donc fait le choix d'utiliser des interfaces C au sein de notre code afin de promouvoir la ré-utilisabilité de notre code. Les Interface sont des composants essentiels des langages orientés objets permettant à une classe quelconque de garantir l'implémentation de certaines méthodes, fournissant ainsi des garanties pour les autres objets qui les appellent dans le code. Nous pouvons ainsi fournir un code plus robuste, à même d'évoluer, le tout sans devoir systématiquement modifier le code du

joueur pour supporter tous nos futurs ajouts. Nous avons utilisés 2 interfaces, les IAttackable et les IIInteractable<sup>1</sup>.

### IAttackable

L'interface IAttackable décrit les objets capable de prendre des dégâts, il s'agit par exemple du joueur, des monstres, des murs, des machines, etc. Celle-ci comporte 2 méthodes : CanBeHit() et OnHit(), qui permettent au joueur et aux autres entités (tels que les projectiles) d'infliger des dégâts à des objets, sans même connaître leur classe concrète.

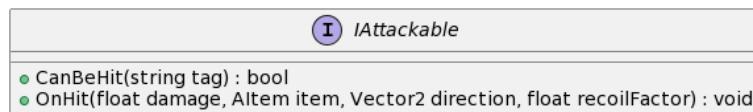


FIGURE 4 – UML de IAttackable

### IIInteractable

L'interface IIInteractable décrit les objets avec lesquels le joueur est à même d'intéragir. Elle comporte 2 méthodes : CanInteract() et Interact(), permettant de mettre en surbrillance l'icône de main active dans l'ATH. Il peut s'agir, par exemple, des items au sol à récupérer, des machines à activer / désactiver ou encore des portes.

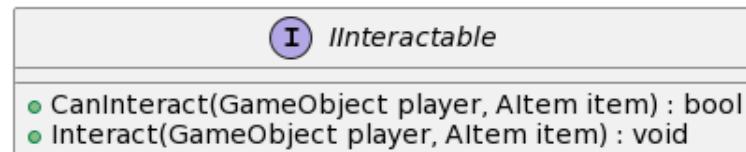


FIGURE 5 – UML de IIInteractable

---

1. Ici, le préfixe "I-" indique qu'il s'agisse d'interface. Il s'agit d'une bonne pratique dans le langage C#.

## Auteurs

**VIDELIER Louis** - Architecture logiciel

**BERTRAND Mathis** - Application des interfaces

## 6.8 Déplacement des entités & Navmesh

### Description

Pour le déplacement des entités, nous avons décidé d'utiliser une technologie inhérente à Unity : le NavMesh. Un Navmesh est une zone sur laquelle un NavMeshAgent peut se déplacer et trouver la trajectoire la plus optimale pour atteindre sa cible. Ce NavMesh a besoin de sources qui lui transmettent les objets sur lesquels l'agent peut marcher et ceux sur lesquels il ne peut pas marcher. Ici nos entités sont donc des NavMeshAgents qui se déplacent en direction du joueur (entités ennemis et neutres), de la météorite (entités ennemis) ou qui fuient le joueur (entités passives). Le NavMesh étant avant tout une technologie utilisé en 3D, nous avons dû utiliser un package NavMeshPlus, qui nous permet d'utiliser un NavMesh en 2D.

### Nouvelles fonctionnalités

- Implémentation de chunks afin d'améliorer les performances.
- Adaptation du déplacement des entités aux chunks.

### Problèmes rencontrés

Durant le développement du mode de construction, nous nous sommes rendus compte que nous étions obligés de mettre le NavMesh à jour à chaque fois que le joueur pose un objet sur la carte. Or cette mise à jour du NavMesh est assez coûteuse en jeu, en particulier sur une carte très grande, ce qui cause notamment des ralentissements importants en jeu. C'est pourquoi nous avons cherché à subdiviser notre carte et donc nos NavMeshs en chunks qui forment ainsi notre carte. Le problème était que le package que nous utilisions ne permettait pas en 2D de réaliser plusieurs NavMeshs reliés les uns aux autres. Tout cela car le NavMesh fonctionne à l'origine sur des surfaces comme des plans en 3D, ce qui explique qu'en 2D puisqu'il n'y a qu'un seul plan il n'est pas possible de réaliser qu'un seul NavMesh.

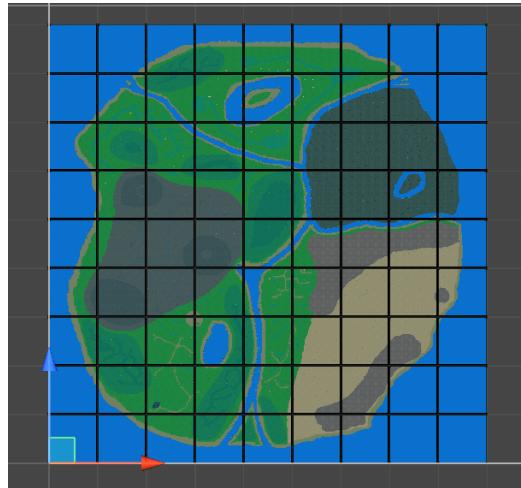


FIGURE 6 – Ensemble des entités

### Implémentation

Afin de résoudre ce problème nous avons modifié les scripts du package NavMeshPlus pour que l'on puisse créer des NavMeshs séparés et reliés par des NavMeshLinks. Nous avons donc divisé la carte en 81 chunks qui ont chacun leur propre NavMesh. Ainsi lorsque l'on a besoin de mettre le NavMesh à jour après avoir placé un objet sur la carte il suffit de vérifier dans quel chunk l'objet se situe et ainsi de mettre à jour le NavMesh correspondant.

### Auteur

**MUHLKE Félix** - Responsable de la construction et des entités.

## 6.9 Crédit d'entités

### Description

Les entités de Scraft font partie des fondations du jeu, en particulier les entités ennemis, puisqu'elles servent au principe même du jeu. Pour ajouter de la diversité au jeu, nous avons ajouté également des entités neutres et passives.

## Nouvelles fonctionnalités

- Ajout des entités ennemis, des zombies (Banshee, Base, Muscular...).
- Ajout des entités neutres (Rats mutants, Elan, Chèvre).
- Ajout des entités passives (Poule, Renard, Canard, Mouton).

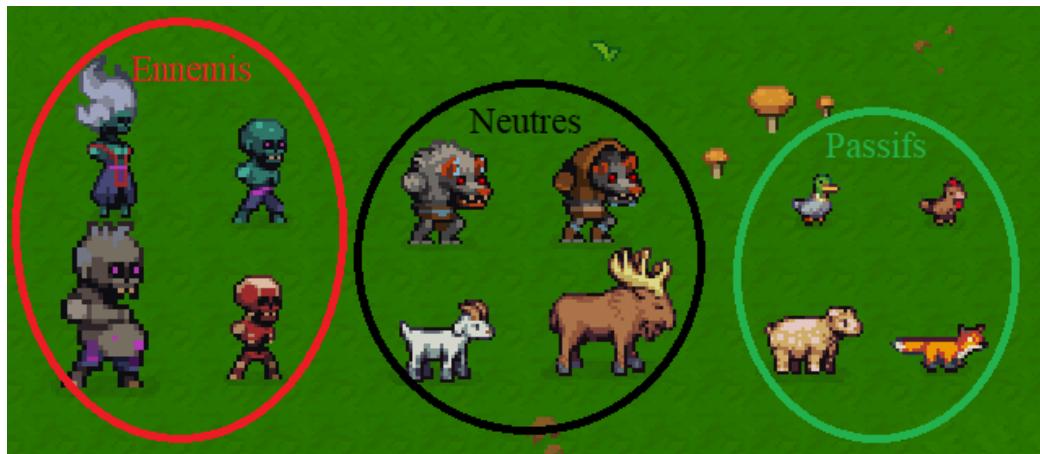


FIGURE 7 – Ensemble des entités

## Implémentation

Les entités sont divisées en trois catégories et donc trois comportements différents : ennemi, neutre et passif.

L'entité ennemie se dirige vers la météorite en évitant les obstacles sur son chemin grâce au NavMesh. Si elle rencontre un joueur dans un certain rayon sur son chemin et qu'il est plus près d'elle que la météorite, elle le suit et l'attaque dans la mesure du possible. Elle attaque également tout ce qui se trouve sur son passage, notamment les constructions.

L'entité neutre se déplace de manière aléatoire dans un rayon autour de son point d'origine. Si un joueur entre dans ce rayon, l'entité neutre suit le joueur et tente de l'attaquer. Si elle le perd de vue, elle retourne à son point d'origine et reprend son mouvement aléatoire. L'entité passive se déplace également de manière aléatoire dans un rayon autour de son point d'origine. Mais si le joueur entre dans ce rayon, l'entité passive fuit dans la direction opposée du joueur. Puis elle revient vers son point d'origine si elle a perdu le joueur de vue.

Nous avons également travaillé les sprites et les animations de toutes les entités.

**Auteur**

**MUHLKE Félix** - Responsable des entités.

## 6.10 Détails de la map

**Description**

Afin de donner au joueur une impression de liberté, nous avons fait de notre mieux pour avoir une carte assez grande. L'implémentation d'un système de chunks a permis l'élargissement de la carte. Une fois la carte assez grande, et déjà designé, nous avons dû "l'habiller", en plaçant des prefabs d'arbre, de rocher et de plantes. Cette étape est cruciale, car sans elle il sera impossible de jouer, et la qualité de son travail joue sur l'immersion du joueur.

**Auteur**

**AUPEST Vincent** - Responsable UI

## 6.11 Construction

**Description**

Le système de construction est très important dans l'évolution du joueur puisqu'il lui permet de se protéger et surtout de protéger la météorite, en construisant des murs, des portes et des machines (tourelles, générateurs...). Il permet de plus au joueur de poser des arbres ainsi que diverses plantes afin de se nourrir. Le joueur peut ainsi placer des objets grâce à ce système en appuyant sur la touche "Tab" par défaut et en sélectionnant l'objet qu'il veut placer. Lors de la sélection, le joueur peut également voir la durabilité et les ressources nécessaires à la construction de l'objet sélectionné. Un curseur permet également d'aider le joueur à placer son objet sur la grille.

**Nouvelles fonctionnalités**

- Ajout de l'interface UI de la construction.

- Implémentation réseau de la construction.
- Adaptation du système de construction au NavMesh.



FIGURE 8 – Interface de construction

## Implémentation

Le système de construction est basé sur une grille implementée avec une matrice. La matrice est accessible par tous les joueurs qui possèdent chacun leur propre GridBuildingSystem qui leur permet de placer des objets dans cette matrice. Cette matrice est composée de GridObjects qui contiennent leur position dans la matrice ainsi qu'un PlacedObject qui a une durabilité, une référence à un GridBuildingSystem (celui de l'hôte) ainsi qu'un ScriptableObject qui contient la hauteur, la largeur, le nom et le prefab de la construction.

A chaque fois que le joueur veut placer un objet sur la grille, le GridBuildingSystem vérifie d'abord s'il n'y a pas un objet avec un Collider2D à l'endroit où il veut le poser, s'il y en a un, un message "Impossible de construire ici" apparaît. Puis, il vérifie si le joueur possède les ressources nécessaires dans son inventaire afin de construire l'objet pour ensuite les consommer en posant l'objet, si ce n'est pas le cas, un message "Matériaux insuffisants" apparaît.

L'interface de construction a été réalisée à partir d'un Canvas et de 2 panneaux en fils, celui de gauche pour la sélection et celui de droite pour la des-

cription. Chaque objet visible à gauche hérite de IPointerDownHandler et est abonné à plusieurs UnityEvents qui permettent d'appeler des méthodes de la BuildingPage qui a une référence de la description et peut ainsi la mettre à jour à chaque fois que le joueur clique sur un des objets présents sur le panneau de droite.

### Auteur

**MUHLKE Félix** - Responsable de la construction

## 6.12 Craft

### 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 à accès à une interface graphique depuis son inventaire 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.

### Implementation

Les craft sont implémentés avec des ScriptableObject qui stock les items nécessaires pour la fabrication d'un item ainsi que le nombre d'items donné après fabrication. Une interface graphique sous forme de livre a été créée pour que l'utilisateur puisse voir les crafts disponibles et ainsi des items. Une fois que l'utilisateur décide de craft un item, l'inventaire est parcouru pour vérifier que le joueur possède bien les items nécessaires, si c'est le cas ils sont alors retiré et l'item résultant est ajouter à l'inventaire du joueur.

### Auteur

**BERTRAND Mathis** - Responsable du Craft



FIGURE 9 – Interface du craft

## 6.13 Props

### Description

Les Props représentent les différents éléments naturels du monde du joueur. Il peut s'agir, par exemple, des arbres, des rochers, des minérais ou de plantes. Au sein de cette nouvelle mise à jour des Props

### Nouvelles fonctionnalités

- Ajout de tous les Props du monde. Permet d'avoir un environnement riche pour le joueur.
- Liaison des CropsProps au cycle Jour/Nuit pour que leur maturité soit relié aux jours passés.
- La croissance des CropProps peut être accélérée par le joueur grâce à de la poudre d'os.
- Les joueurs peuvent maintenant récolter les CropProp.
- Ajout des sons pour tous les Props.
- Les CropsProps sont récoltables par le joueur grâce à l'item "Houe".

- Les Props réapparaissent après avoir été détruits après un certain temps.

## Implémentation

La base des Props était déjà implémenté en utilisant la même logique que les Items. En utilisant un couple AProp / APropData et une hiérarchie de classes, tous les Props sont complètement décrits et .

Afin de lier les CropProp (les différentes plantes que le joueur peut planter et faire pousser), nous avons utilisé le principe des Delegates. Les Delegates sont un concept propre à C# où il est possible de déclarer, à la manière d'une classe, le type d'une fonction de manière intuitive et de référencer ce type comme type pour un argument ou un attribut au sein d'une classe.

Ici, nous avons utilisé les Delagates et des Evénements pour lier les plantes au système Jour/Nuit afin de les faire pousser d'une unité chaque jour, tout en suivant les données spécifiées dans les CropPropData.

## Auteurs

**VIDELIER Louis** - Création des Props.

**BERTRAND Mathis** - Correction Props, Ajout des sons aux Props.

## 6.14 Attaques/Popup/Recul

### Description

L'attaque est la fonctionnalité la plus importante du joueur, il lui permet de récolter des ressources, de chasser et de se défendre face aux monstres. Le joueur peut attaquer à 360° en déplaçant sa souris dans la direction souhaitée. Une attaque est déclenché en appuyant sur la touche "Attaquer" <sup>1</sup>. Attaquer consomme de l'énergie au joueur, il faut donc faire attention et ne pas manquer ses coups. L'utilisateur peut attaquer les animaux, les monstres et les Props (arbres, minerais, feuillage etc...) présents sur la map. Sur les entités une attaque à un faible pourcentage d'être un coup critique, celui ci augmente les dégâts et le recul subi.

---

1. "clic gauche" par défaut

## Implementation

Le joueur attaque avec l'item qu'il tient en main, le choix de cet item a un impact important sur l'attaque. Pour rendre les attaques différentes selon les items, nous avons rajoutés des propriétés sur les ScriptableObject des items (ItemData). Selon les items nous avons une portée, une vitesse de l'attaque, les dégâts et le recul subi par l'attaque différents . Par exemple une lance à une meilleure portée mais attaque moins vite qu'un couteau. L'animation de l'attaque avec un item a été retravaillé pour que sa vitesse d'exécution soit calculé en fonction de l'item utilisé.

Le joueur doit pouvoir savoir combien de dégâts il inflige et subi. Nous avons donc implémenté des Popup. Les popups permettent de faire apparaître un message quelques secondes avant de disparaître. Dès que des dégâts sont infligés à un joueur, une entité ou un props, une popup apparaît.

Pour rendre les combats plus dynamiques nous avons mis un système de recul sur les entités, lorsqu'il subit une attaque, l'entité est projeté plus ou moins dans la direction du coups. Selon le poids de l'entité et l'arme utilisé pour attaquer le recul est plus ou moins important. Le joueur reçoit aussi des reculs par les entités.

Les entités, les joueur, les props, les constructions utilise l'interface IAttackable (pour plus de détails voir section "Interfaces").

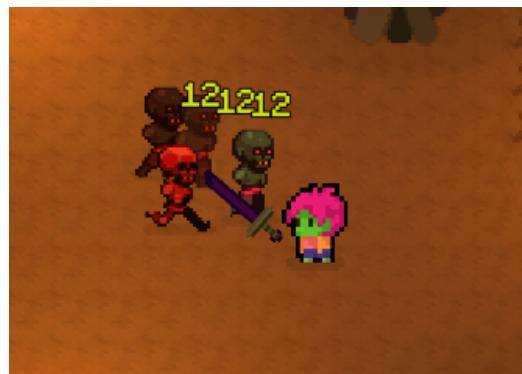


FIGURE 10 – Joueur qui se défend

## Auteur

**BERTRAND Mathis** Responsable du Player

## 6.15 Cycle jour/nuit et Heure

### Description

Le cycle jour nuit est un élément central du jeu, il fait apparaître les vagues de monstres la nuit, fait pousser les plantes et gère la luminosité du jeu. Un cycle complet dure 12 minutes, 6 minutes de journée, 1 min 30 de couchée de soleil, 4 min 30 de nuit et 1 min 30 de levé de soleil. Le nombre de jours passés est indicatif du score du joueur, on rappelle que l'objectif est de survivre le plus de jours possibles.

### Implémentation

Une journée est divisé en 6 états qui ont chacun une durée et une couleur : 2h de levée du soleil 3 heures pur le matin, 7 heures de journée, 2 heures d'après midi, 3h de couchée du soleil et 8 heures de nuit. Au fil des minutes la couleur du jeu transite lentement vers la couleur du prochain état pour ne pas avoir une changement de couleur brut.



(a) Visuel à 12h30



(b) Visuel à 22h15

FIGURE 11 – Exemples de différentes luminosités

### Auteur

**BERTRAND Mathis** - Responsable du cycle Jour/Nuit

## 6.16 Lumière

### Description

La nuit est sombre et le champs de vision du joueur est réduit. Afin de pouvoir défendre sa base de la meilleure des façons, les lumières sont des outils indispensables.

### Implémentation

Il existe deux façons de s'éclairer. La première est la plus simple moins importante, en tenant un briquet en main. Le joueur est entouré d'un faible rayon lumineux, suffisant pour voir autour de soi et se déplacer, en revanche dès qu'il change d'arme en main, la lumière s'éteint. La lumière est gérée par les Light2D de Unity.

La deuxième façon de s'éclairer est de poser des torches via le menu de constructions, par défaut la torche est éteinte, en interagissant avec un briquet, la torche s'allume et éclaire sur un rayon beaucoup plus grand que le briquet. Une coroutine est démarré pour que l'éclairage dure 5 minutes, avant de s'éteindre.



FIGURE 12 – Différentes lumières de nuit

**Auteur****BERTRAND Mathis** - Responsable de la Lumière

## 6.17 Energie / Sprint

**Description**

Par défaut le joueur qui marche n'est très rapide. Le joueur se fait distancer rapidement par les entités passives et ne peut pas échapper aux monstres. En consommant de l'énergie le joueur peut se mettre à courir, ce qui momentanément sa vitesse.

**Implémentation**

Le joueur peut posséder 100 points d'energies maximum, l'énergie peut être consommer pour courir en appuyant sur la touche "Courir"<sup>1</sup>, ou pour attaquer. Quand le joueur consomme a finit de consommer son energie, il doit attendre quelques secondes avant qu'elle ne se régénère. Des particules de poussières ont été rajoutés quand le joueur court.

**Auteur****BERTRAND Mathis** - Responsable du Player

## 6.18 Spawner d'entités/Vague

**Description**

Les zombies ne sont pas présents

**Implémentation**

Pour équilibrer le jeu, nous avons décidé d'une part, de faire 4 spawns différents qui se déplacent dans chacun des angles, dans le but d'éviter que les monstres vous attaquent toujours du même côté. Et puis par la suite dans un second temps, nous avons ajouté un système de "coût" pour l'apparition des monstres. Les monstres n'étant pas tous aussi puissant, nous avons fait

---

1. "maj gauche" par défaut



FIGURE 13 – Joueur qui court

que chaque nuit, une total de  $x$  monstres apparaissent, en prenant compte que ces  $x$  points sont distribués entre les 4 spawners, et que certains monstres en valent plusieurs. Cette implémentation permet un début de partie équilibré, et une progression exponentielle des vagues, tout ça en évitant que les monstres trop puissants n'apparaissent trop tôt dans la partie. De plus, le nombre de joueur présent dans le lobby influe directement le nombre d'entités par vague.

## Auteurs

**BERTRAND Mathis** - Responsable objectif. **MUHLKE Félix** - Responsable des entités non-joueur. **AUPEST Vincent** - Concepteur.

## 6.19 Affichage tête haute

### Description

Lorsque l'utilisateur joue, l'affichage tête haute, aussi appelé ATH est présente à l'écran pendant toute la durée du jeu. L'ATH est notre principal moyen de communication avec l'univers virtuel qui entoure notre personnage.

Elle nous fournit des informations cruciales, des outils et des indicateurs visuels, nous permettant ainsi de prendre des décisions éclairées et de guider nos actions dans le jeu. L'ATH doit être simple, claire et pas trop encombrant.

### **Avancement**

L'ATH se découpe en 6 parties :

Dans le coin supérieur droit, vous avez une boussole qui pointe vers la météorite, ainsi si vous êtes éloigné vous pourrez la retrouver sans problème.

Dans le coin supérieur gauche, vous avez la barre de vie (barre rouge) et la barre d'énergie (barre jaune) de votre personnage. Ainsi que l'heure de la journée (dans le jeu). Voir les sections associées pour plus d'informations.

Dans le coin inférieur droit, vous avez une main qui est le symbole d'interaction, par défaut quand vous ne pouvez intéragir qu'avec rien, la main est grisée. Au contraire, celle ci devient blanche quand vous pouvez intéragir avec quelque chose autour de vous.

Au centre bas, vous avez la barre d'inventaire qui permet au joueur d'accéder en jeu à 10 items. L'item sélectionné est surélevé d'une flèche. Pour changer d'item sélectionné vous pouvez faire rouler la molette de votre souris ou utiliser les raccourcis de barre d'inventaires.

Au centre haut, vous avez la barre qui affiche la vie restante de votre météorite, voir "Défaite" pour plus d'informations.

### **Auteur**

**BERTRAND Mathis** - Responsable Joueur.

## **6.20 Personnalisation du Skin**

### **Description**

Afin d'améliorer l'expérience utilisateur, nous avons décidé d'ajouter une personnalisation des couleurs de skin.



FIGURE 14 – Affichage tête haute en jeu

### Nouvelles fonctionnalités

- Ajout des options de personnalisation.



FIGURE 15 – Menu de personnalisation

### Implémentation

Nous avons implémenté le système de personnalisation en nous appuyant sur celui que nous avions fait pour la première soutenance, que jusqu'à ce jour, nous attribuait des couleurs aléatoire. La version que nous avons implémenté permet de modifier les cheveux, la peau, la veste, le tshirt, le pantalon, ainsi que les chaussures. La modification se fait en modifiant les composants RGB.

**Auteur****AUPEST Vincent** - Responsable UI.

## 6.21 Sauvegarde des données

**Description**

Pour éviter que l'utilisateur ait à renseigner ses informations et préférences à chaque fois qu'il se connecte, nous avons décidé d'ajouter un système de sauvegarde des données.

**Nouvelles fonctionnalités**

- Sauvegarde du pseudonyme.
- Sauvegarde du skin.

**Implémentation**

Nous avons utilisé le format .json pour sauvegarder nos données. Une fois le module Json importé dans nos scripts, nous n'avions plus qu'à créer une class qui stockait les attributs utils, comme le pseudonyme ou le skin.

**Auteur****AUPEST Vincent** - Responsable sauvegarde.

## 6.22 Son

**Description**

Les sons sont des éléments cruciaux quant à l'immersion dans un jeu vidéo. Un jeu vide de son ou avec des effets sonores peu convainquant ne permet pas au joueur de se plonger dans le jeu et de s'immerger dans l'univers qui lui est présenté. Les sons se doivent donc d'être convainquant tant par le choix des effets sonores que dans la manière dont ils sont communiqués dans le jeu, par exemple, la réduction du bruit en fonction de la distance entre la source et des joueurs.

## Nouvelles fonctionnalités

- Ajout effets sonores en réseau.
- Ajout effets sonores dans le périmètre du joueur.
- Ajout d'outils de visualisation.
- Ajout mixeurs audio, contrôle du volume des effets et de la musique.
- Ajout d'objets d'adressage automatique des effets audio pour optimiser la communication réseau entre les joueurs.

## Implémentation

L'implémentation des sons dans Scraft a été faite en suivant perpétuellement la même idée fondamentale : la simplicité d'utilisation pour les autres développeurs.

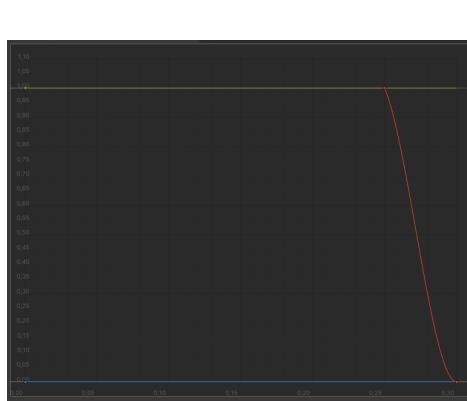
La gestion du son au sein de Scraft est composé de cinq couches. Trois d'entre elles sont des éléments de Unity qui permettent la manipulation du son ; les AudioMixer pour mixer les différents "catégorie" de son (musique ou effets sonore), les AudioListener qui permettent d'avoir un point de référence qui capte les sons joués dans le monde, et les AudioSource qui permettent de jouer des sons. Les éléments restants sont les scripts que nous avons implémenté. Le premier ; l'AudioEmitter, est un wrapper<sup>1</sup> autour d'une AudioSource. Le support du son devant être ajouté dans la majorité des objets, il a été décidé de faire une interface épurée et avec un contrôle très superficiel côté développeur, laissant ainsi l'AudioEmitter contrôler beaucoup de paramètres automatiquement, tel que le calcul des courbes volume/distanc, ou la possibilité de jouer des sons en réseau, autour de l'objet ou pour tous les joueurs. D'autres méthodes auxiliaires permettent de lancer des boucles en arrière plan, et de ne jouer un son que lorsqu'une condition est respectée. Ce système a été utilisé pour les bruits de pas du joueur, les bruits de pas n'étant joués que lorsque le joueur est en mouvement.

Le second, plus simple ; l'AudioManager, permet de contrôler les Mixers de manière intuitive (conversion du volume avec une échelle absolue [0,1] en logarithmique pour les Mixers) pour la gestion du volume du joueur dans les options du jeu.

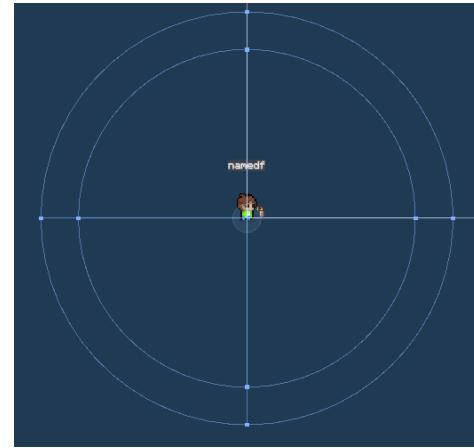
Un autre outils auxiliaire a également été ajouté afin de simplifier l'utilisation des sons au sein de Scraft. Un objet d'indexage automatique, permet de

---

1. Élément logiciel "enveloppant" un autre, permettant un interface plus simple avec celui-ci.



(a) Courbe volume/distancé générée par l'AudioEmitter. L' AudioSource est complètement géré par l' AudioEmitter. Sa configuration est créée automatiquement dans l'éditeur, via les champs de l' AudioEmitter.



(b) Périmètre d'entente des sons joués. Le cercle intérieur indique la zone où le volume est maximal. Hors du cercle extérieur, il n'est plus possible d'entendre le son joué.

FIGURE 16 – Gestion du son au sein des AudioEmitter.

simplifier et d'optimiser la communication des sons d'un client à l'autre. Au lieu de transmettre un son directement sur le réseau ; un indice est communiqué, et chaque client utilise ainsi son indexeur local pour connaître le son à jouer.

## Auteur

**VIDELIER Louis** - Responsable audio et mise en réseau de l'audio.

### 6.23 Réglage du son

#### Description

Le réglage du son est la possibilité pour le joueur de modifier le niveau sonore des sons à une échelle globale dans le jeu.

#### Nouvelles fonctionnalités

- Ajout interface graphique permettant la gestion du son.

## Implémentation

La gestion du son est constituée de deux parties. La première partie est l'interface graphique permettant à l'utilisateur de changer le volume général du jeu de manière intuitive. L'autre partie est l'AudioManager, qui est un singleton que tous les joueurs possède et qui permet de modifier les AudioMixers sous-jacent à l'échelle de tout le jeu.

## Auteurs

**AUPEST Vincent** - Responsable UI

**VIDELIER Louis** - Responsable son, interface AudioManager et Mixers

## 6.24 Items

### Description

Dans cette nouvelle version de Scraft, de nombreux items ont été ajoutés afin de rendre l'expérience du joueur plus riche. Un nouvel item unique ; la clé à molette, a également été ajoutée. Celle-ci permet de gérer les machines.

### Nouvelles fonctionnalités

- Ajout d'une dizaine de nouveaux items.
- Ajout des armes à distance.
- Ajout d'une animation pour les items au sol.
- Les items au sol disparaissent désormais après seulement 2 minutes (précédemment 5).

## Implémentation

Design des sprites via Aseprite, et incorporation des ScriptableObject au jeu. De nouvelles armes ont été ajoutées, les Armes à distance (RangeWeapon). Celles-ci sont basées sur les nouveaux objets Projectiles<sup>1</sup>.

---

1. Voir "Projectiles".

## Auteurs

**BERTRAND Mathis** - Équilibrage général et nouveaux items. **VIDELIER Louis** - Ajout de la Clé à molette. **AUPEST Vincent** - Responsable design.

## 6.25 Inventaire

### 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.

### Nouvelles fonctionnalités

Afin de rendre l'inventaire plus rapide d'utilisation, nous avons rajouté plusieurs raccourcis permettant à l'utilisateur d'être rapide dans sa manipulation :

- Maintenir "Maj gauche" + "clic gauche" sur un item déplace instantanément l'item entre la partie stockage et la barre d'inventaire. Ce raccourci permet également d'équiper rapidement une armure sur le joueur.
- Appuyer sur un des raccourcis de barres d'inventaire au dessus d'un item déplace l'item sur l'emplacement de barre d'action correspondant à la touche.

- Appuyer sur la touche "Jeter" au dessus d'un item jette un item du stack au sol, si "maj gauche" est maintenu en même temps, tout le stack est jeté au sol.
- Ajout des raccourcis pour la barre d'inventaire.
- Ajout du déplacement rapide dans l'inventaire.

### **Implémentation**

Afin d'implémenter les nouvelles fonctionnalités concernant l'inventaire ; nous avons réutilisé le système d'input précédemment utilisé.

### **Auteur**

**BERTRAND Mathis** - Responsable Joueur et Inventaire.

## **6.26 Assignations des touches**

### **Description**

Pour améliorer la prise en main, nous avons ajouté un système d'assignations des touches. Celui-ci permet au joueur d'être plus à l'aise en utilisant le jeu, en laissant au joueur la possibilité de personnaliser ses touches.

### **Nouvelles fonctionnalités**

- Possibilité de changer les touches d'action.

### **Implémentation**

Pour pouvoir ajouter un système d'assignation de touche, nous avons opté pour le module "New Input System" présent dans Unity. Celui-ci nous permet d'assigner facilement les touches aux joueurs. En plus de ça, nous avons bloqué la possibilité au joueur d'assigner la même touche deux fois.

### **Auteur**

**AUPEST Vincent** - Responsable UI.

## 6.27 Projectiles

### Description

Les projectiles sont des petits éléments indépendants capables de faire des dégâts et d'avancer par eux même. Ils sont simples, mais essentiels pour les interactions à distance entre les joueurs et le reste du monde.

### Nouvelles fonctionnalités

- Ajout de projectiles capables auto-propulsés.
- Ajout d'objets divers capables d'instancier des projectiles (Armes à distance, Tourelles).

### Implémentation

Les projectiles, comme les Props, sont implémentés au moyen d'un ScriptableObject (ProjectileData) et d'une classe MonoBehaviour associée (Projectile). Des projectiles prêts à l'emploi peuvent donc être créés sous forme de prefab.

La classe Projectile permet de donner une direction au projectile et contient un champ ProjectileData indiquant les données du projectile (sa vitesse de déplacement, ses dégats à l'impact, etc.). Lorsqu'un nouveau projectile est instancié, il suit perpétuellement la direction qui lui est donnée jusqu'à rencontrer un objet avec une disposant d'une boîte de collision, ou sortir des limites de la carte.

En utilisant l'interface IAttackable (désignant tous les objets qui peuvent recevoir des dégâts<sup>1</sup>), lorsque le projectile rencontre une boîte de collision, elle consulte l'objet rencontré et tente d'appeler les méthodes infligeant des dégâts à l'obstacle, puis, il est automatiquement détruit.

### Auteurs

**VIDELIER Louis** - Responsable projectiles.

**BERTRAND Mathis** - Mise en réseau des projectiles.

---

1. Voir "Interfaces".

## 6.28 Fin de partie

### Description

Lorsque la météorite n'a plus de vie, la partie est terminée pour tous les joueurs. Une fois la météorite détruite, un panneau apparaît sur l'écran des joueurs leur indiquant la fin de la partie et le nombre de jours auxquels ont survécu. Après un 15 secondes, les joueurs sont renvoyés au menu principal.

### Implémentation

À chaque instant, on vérifie les points de vies restants de la météorites. Lorsque ceux-ci tombent à zero, la partie est automatiquement stoppée pour tous les joueurs.

### Auteur

**BERTRAND Mathis** - Responsable écran fin de partie.

## 6.29 Machines - Générateurs Tourelles automatiques

### Description

Les machines forment un point de gameplay important dans Scraft. Elles permettent aux joueurs de défendre leur base de manière automatique à l'aide de tourelles et de générateurs. Les machines sont essentielles lors des longues parties de jeu et sont des alliées de taille pour vous aider à vaincre les vagues de monstre massive.

### Nouvelles fonctionnalités

- Ajout de générateurs qui produisent de l'énergie à partir d'Items fixés. (Générateur à charbon et Générateur à uranium).
- Ajout de tourelles automatiques, capables de tirer sur les montres dans un certain périmètre. (Tourelle en bois, Tourelle en caoutchouc, Tourelle en or et Tourelle en uranium).
- Ajout d'un nouvel outil : la Clé à molette. Permet d'éditer les liens entre les machiens de la scène. Elle est essentielle.

## Implémentation

Les machines fonctionnent en symbiose avec une hiérarchie de classes leur permettant d'avoir des fonctionnalités partagés et cohérentes en utilisant de la réutilisation de code.

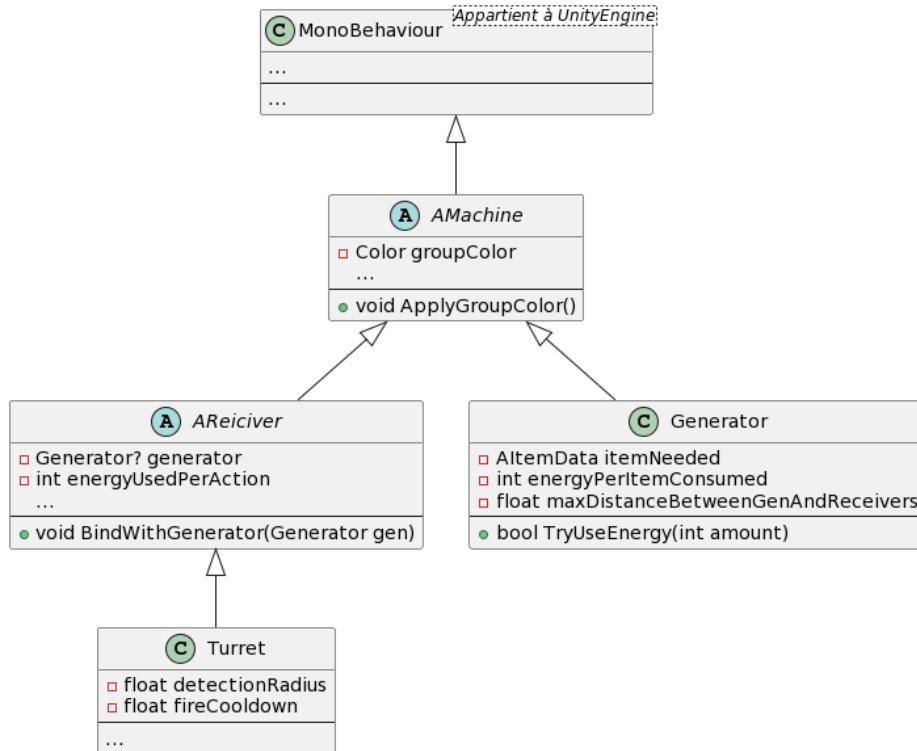


FIGURE 17 – Diagramme UML de la hiérarchie de classes nécessaires pour les machines

**Pour toutes les machines** Les machines sont des constructions, celles-ci apparaissent dans le menu de construction comme une catégorie représentée par un engrenage.

Toutes les machines reprennent la même base à partir de la classe AMachine. Ainsi, toutes les machines ont une couleur afin d'afficher leur groupe en Mode Édition. Les tourelles qui ne sont pas encore liées n'ont pas de groupes par défaut, mais dès lors qu'elles le sont, leurs couleurs de groupe changent avec la couleur du générateur. Cela est réalisé en utilisant des SpriteRenderer et leur

attribut de couleur avec une image blanche. Des lignes entre les différentes machines peuvent être tissées à partir de LineRenderer.



FIGURE 18 – Tourelle liée à un générateur de groupe Rouge

Afin d'afficher l'overlay<sup>1</sup> au dessus des machines en Mode Édition, toutes les machines possèdent une référence vers un script du joueur. Lorsque le joueur porte une Clé à molette en main, les machines actives les différents Sprite-Renderer qu'elles possède pour afficher toutes leurs données et inversement.

---

1. Couche graphique positionnée au-dessus d'une autre et permettant d'afficher des informations supplémentaires.



FIGURE 19 – Affichage de l’overlay lorsque le joueur tient un Clé à molette dans la main.

**Générateurs** Les générateurs sont des objets avec lesquels le joueur peut interagir lorsqu'il tient l'objet demandé par le générateur. Les Générateurs utilisent donc l'interface IInteractable pour afficher l'icône d'interaction en surbrillance lorsque l'interaction est possible.

Les générateurs possèdent un attribut `energyStored` qui leur permet de conserver une quantité d'énergie précise. Lorsqu'une tourelle à besoin de tirer sur un monstre, elle tente d'utiliser une partie de l'énergie du générateur, si celui-ci renvoie Faux (pas assez d'énergie), alors la tourelle ne tire pas et le niveau d'énergie ne change pas.

**Tourelles** Les tourelles sont des constructions au principe fondamentalement simple : lorsqu'elles détectent un ennemi dans un rayon donné et qu'il n'existe aucun mur entre elles et eux, alors elles tentent de tirer.

Les tourelles ont aussi des états qui sont affichés au-dessus d'elle en Mode Édition.

Le premier état concerne le statut activée / désactivée et s'afficher par une petite icône qui peut être soit verte, soit rouge.



(a) Tourelle activée

(b) Tourelle désactivée

FIGURE 20 – Statut de fonctionnement des tourelles

Le second état concerne le statut d'alimentation des tourelles. Si la tourelle n'est pas liée à un générateur ou si le générateur ne contient pas l'énergie nécessaire pour faire un tir, une icône d'éclair clignotante est affichée au dessus des tourelles.



(a) Tourelle sous-alimentée

(b) Tourelle correctement alimentée

FIGURE 21 – Statut d'alimentation des tourelles

## Auteurs

**VIDELIER Louis** - Responsable machine, création de la hiérarchie de classes et création des concepts de jeu.

**BERTRAND Mathis** - Mise en réseau des machines.

**MUHLKE Felix** - Ajout support construction des machines dans la grille de jeu.

**AUPEST Vincent** - Design des éléments graphiques des machines (clé à molette, tourelles, générateurs, icônes de statuts des machines).

## 6.30 Réseau

### Description

Après la soutenance 2, nous avons décidé de ne plus utiliser Mirror pour le réseau mais Photon. Nous avons donc dû modifier ce qui avait été mis en réseau sous Mirror.

### Implémentation

L'intégralité des fonctionnalités de Scraft est assurée en réseau et synchronisée entre les joueurs. Chaque joueur envoie aux autres clients, les actions de son personnage et de ses items. Le cycle jour nuit, les constructions ainsi que les entités sont gérés par l'hôte et envoyés aux clients.

### Auteurs

**BERTRAND Mathis** - Responsable du fonctionnement général du réseau.

**MUHLKE Felix** - Assistant mise en réseau de la construction et des entités.

**VIDELIER Louis** - Mise en réseau du son.

**AUPEST Vincent** - Responsable connexion entre les joueurs.

## 6.31 Équilibrage

### Description

Afin de rendre le jeu plus équilibré et d'augmenter la jouabilité, nous avons choisi de faire une session d'équilibrage avant de sortir le jeu. Ainsi chaque recette de fabrication, de construction, les items données par les Props et les entités ont été réfléchi pour avoir un gameplay harmonieux et équilibré.

### Auteurs

**AUPEST Vincent** - Responsable équilibrage.

**BERTRAND Mathis** - Equilibrage Props.

**MUHLKE Felix** - Equilibrage des entités.

**VIDELIER Louis** - Équilibrage tourelles et machines.

## 7 Structure du projet

### 7.2 Fonctionnel

#### Méthodologie et Technologies



##### InkScape, logiciel dessin vectoriel

InkScape est un logiciel de dessin vectoriel que nous avons utilisé pour faire le logo de Megawaves Software ainsi que celui de Scraft. C'est un logiciel puissant et accessible et est également libre et Open-Source.



##### 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.

#### Gestion du projet

Pour gérer le projet en lui même, nous avons utilisés 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.

## 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. (6982 messages au total)



### Overleaf - Rédaction L<sup>A</sup>T<sub>E</sub>X en équipe

Overleaf est une plateforme que nous utiliserons pour rédiger nos rapports et autres documents au format L<sup>A</sup>T<sub>E</sub>X.

### 7.3 Operationnel

Nous avons réussi à éviter une partie des dépenses en utilisant Photon, qui propose un système d'hébergement gratuit pour nos parties, et github pour l'hébergement de notre site web.

#### Coût du projet

Au final, la réalisation de Scraft nous aura coûté 20€ car nous avons acheté des sprites pour embellir notre jeu.

## 8 Site

### 8.1 Introduction

Le site web a été fait en HTML, CSS et JavaScript pour le module Swiper.JS. Tout le contenu externe au HTML / CSS est délivré en utilisant les cdn<sup>1</sup> des technologies associés (Google Font et Swiper.JS).

Le site Web peut être consulté à l'URL suivante : <https://scraftgame.github.io/>

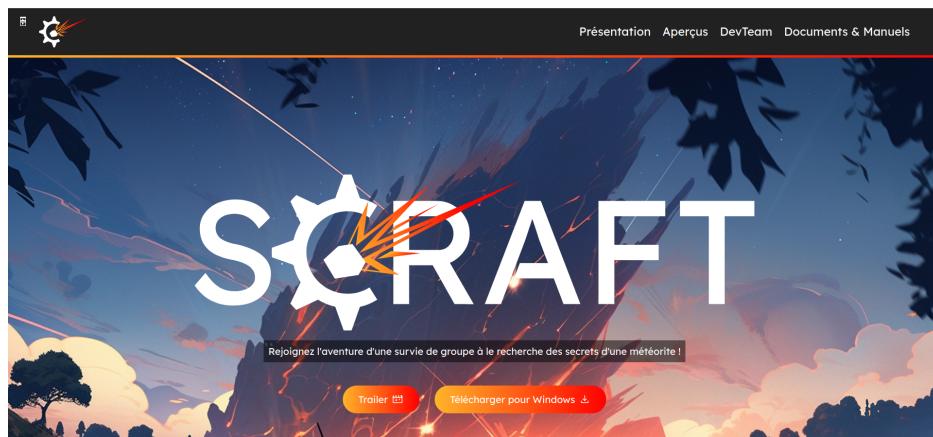
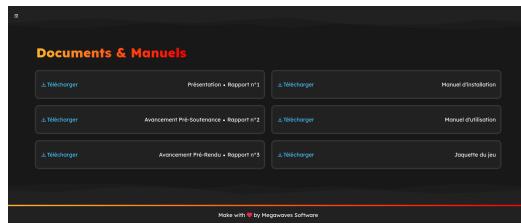


FIGURE 22 – Sommet du site Web de Scraft.

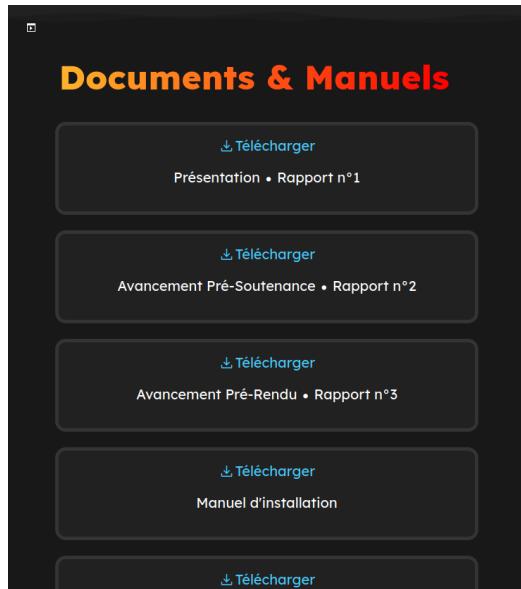
1. Content Delivery Network - Un type de serveur dédié au partage de ressources sur Internet. Ici, les CDN nous permettent de vous donner accès aux polices, aux icônes et à Swiper.JS, pour le carrousel au milieu du site.

## 8.2 Responsive

Le site Web de Scraft est dit "responsive". Cela signifie que les éléments du site s'arrange en fonction de la taille de l'écran du client qui l'affiche, proposant une expérience visuelle plus agréable sur tout type d'écran. Pour cela, la technologie des media-querry est utilisée, pour modifier dynamiquement le style des éléments de la page en fonction de la taille des écrans. Le site supporte donc un affichage sur PC, tablette et téléphone-portable.



(a) Section "Documents & Manuels", vue PC.



(b) Section "Documents & Manuels", vue Mobile.

FIGURE 23 – Affichage du site de manière responsive

## 8.3 Contenu du site Web

Le site web est divisé en plusieurs sections qui seront détaillées ci-dessous.

### Header - Entête

Le header est la partie supérieure du site Web. Elle contient une image du logo de Scraft et une navbar<sup>1</sup> vers les différentes sous-sections du site. Elle est séparé du reste du site avec un dégradé aux couleurs de Scraft.

### Héro

Le héro est la partie destinée à accueillir les usagers sur la page, et afficher les informations marquante pour retenir l'attention. Le héro de la page de Scraft contient une image d'illustration du jeu, le logo de Scraft, une phrase d'acceche et des boutons vers des services externes : Trailer et Téléchargement pour Windows.

### Présentation

La partie Présentation est une partie destinée à présenter les différents points fort du jeu Scraft. Cette partie est responsive en changeant l'alignement des éléments dans le tableau, afin de les faire se superposer en vue mobile.

### Aperçus

La partie Aperçus contient un carrousel<sup>2</sup>. En passant le curseur au dessus de chaque image, une petite bulle de texte apparaît et décrit ce qui y est affichée. Le carrousel a été conçu avec la bibliothèque Swiper.JS.

### DevTeam

La partie DevTeam contient les photos des différents membres du groupe. Cette partie est responsive en changeant l'alignement des éléments dans le tableau, afin de les faire se superposer en vue mobile.

---

1. Barre de navigation. Élément des pages web vers d'autres éléments de la page ou vers des services externes

2. Un carrousel est un élément de design récurrent dans les pages Web, où plusieurs images se suivent dans une direction donnée et où l'utilisateur peut les faire défiler

## Documents & Manuels

La partie Documents Manuels contient des liens vers chaque document à rendre au fil de la production de ce projet. Cette partie est responsive en changeant l'alignement des éléments dans le tableau, afin de les faire se superposer en vue mobile.

## 8.4 Évolution du site de Megawaves Software

Originellement, nous avions prévu de faire le site Web de l'éditeur Megawaves Software, en plus du site de Scraft, mais finallement vous avons préféré nous concentrer uniquement sur le site Web de Scraft, afin de produire un contenu plus en lien avec les attendus initiaux, et plus qualitatif.

## 9 Logo

Le logo de Scraft a été le fruit de plusieurs jours de réflexion. Il mèle un style minimaliste et brutal qui, s'intégrant dans le lettrage, met en évidence la météorite qui s'abat sur l'île et l'une partie d'un écrou qui forme la lettre C du logo. La lettre A a également été retravaillée afin de lui apporter un caractère plus futuriste.

Le choix d'un dégradé pour suivre l'esthétique de Megawaves Software a également été pris. Le logo existe en plusieurs versions ; noir, blanc, "C seul" et "C et flamme coupée" pour les petits formats (tels que les favicons<sup>1</sup>). C'est un logo polyvalent qui apporte une identité propre au jeu et à son esthétique générale. Le logo a été produit sur le logiciel InkScape, qui est un logiciel de dessin vectoriel libre et open-source.

---

1. Icônes des onglets sur les sites Web



(a) Logo principal



(b) C coupé avec flamme



(c) C coupé

FIGURE 24 – Logos de Scraft en fonction des formats

## 10 Conclusion

En conclusion, le projet "Scraft" est une réalisation dont nous sommes fiers, malgré les défis et contraintes auxquels nous avons été confrontés. Notre détermination à rester fidèle à la vision initiale du jeu nous a permis de proposer un produit à la hauteur de nos attentes.

Au cours du développement de ce projet, nous avons dû faire des choix et nous organiser afin de ne pas ralentir notre travail. La suppression de la fonctionnalité "Arbre de recherche" nous a par exemple permis de pouvoir nous concentrer plus sur les détails et l'harmonie de notre jeu.

Nous avons du faire preuve de travail acharné pour terminé le projet à temps et nous sommes heureux du produit rendu. Chacun a contribué avec passion et engagement, et cela se reflète dans la qualité du jeu que nous avons créé.

En résumé, le projet "Scraft" est le fruit d'un travail d'équipe passionné et

dévoué, qui a su surmonter les obstacles et respecter les objectifs fixés.

## 10.1 Produit final

A la suite de ces soutenances, nous sommes heureux de vous proposer le produit "Scraft". Notre jeu s'inscrit donc dans les jeux de survie en deux dimensions. Notre jeu propose une expérience immersive et exigeante, dans un monde post-apocalyptique.

Les joueurs de Scraft se retrouvent isolés sur une île où une météorite est tombée, transformant les habitants en monstres. Les joueurs devront lutter pour leur survie contre des vagues incessantes en protégeant la météorite, dans l'espoir qu'elle soit la clé pour restaurer le monde "d'avant".

Les joueurs auront la possibilité d'évoluer dans un open-world<sup>1</sup> immersif, leur permettant d'explorer des biomes différents, de récolter des ressources, et de développer des défenses. De la survie, des combats, de la construction et de l'exploration, tout le monde y trouvera son compte.

---

1. Monde ouvert

## 11 Sources

### 11.1 Forums

<https://stackoverflow.com/>  
<https://forum.unity.com/>

### 11.2 Vidéos

Input System : [https://www.youtube.com/watch?v=csqVa2Vimao&list=LL&index=1&ab\\_channel=samyam](https://www.youtube.com/watch?v=csqVa2Vimao&list=LL&index=1&ab_channel=samyam)  
Colliders : <https://www.youtube.com/watch?v=Cry7FOHZGN4>  
Aseprite : <https://www.youtube.com/watch?v=59Y60TzNrhk>  
Base : <https://www.youtube.com/watch?v=GFHQaRdnasE>  
Tilemap : [https://www.youtube.com/watch?v=ryISV\\_nH8qw&list=LL&index=5&ab\\_channel=Brackeys](https://www.youtube.com/watch?v=ryISV_nH8qw&list=LL&index=5&ab_channel=Brackeys)  
C# : [https://www.youtube.com/watch?v=RAZjcibFE1A&list=PLUWxWDlz8PYLKlr6F-fwCs02DH1g2hrgS&ab\\_channel=TUTOUNITYFR](https://www.youtube.com/watch?v=RAZjcibFE1A&list=PLUWxWDlz8PYLKlr6F-fwCs02DH1g2hrgS&ab_channel=TUTOUNITYFR)  
ScriptableObject : [https://www.youtube.com/watch?v=r5ZV9hHiI7o&list=LL&index=5&t=484s&ab\\_channel=TUTOUNITYFR](https://www.youtube.com/watch?v=r5ZV9hHiI7o&list=LL&index=5&t=484s&ab_channel=TUTOUNITYFR)  
Base : <https://www.youtube.com/watch?v=7iYWpzL9GkM>  
Sauvegarde : <https://www.youtube.com/watch?v=os9dn9zcwgE&t=1s>

### 11.3 Chaînes

<https://www.youtube.com/@Brackeys>  
<https://www.youtube.com/@CodeMonkeyUnity>