

Compte-rendu du projet de spécialité

Mirage's Journey

Sommaire

Sommaire	2
Introduction	3
1. Principe du jeu	3
2. Motivation	3
3. Contexte du projet	4
Bilan des réalisations	5
1. Le site web, plateforme de présentation et d'hébergement	5
2. Le jeu : ce qui a été fait	6
3. Réalisations en matière de game design	10
Bilan de la conduite de projet	11
1. Comparatif des différences de gestion	11
2. Les points positifs du projet : un jeu fonctionnel	11
3. Les points négatifs : un jeu qui reste incomplet	12
Conclusion et perspective	14
1. Rétrospective par rapport aux attentes en début de projet	14
2. Perspectives pour un développement futur	15
3. Conclusion	15
Annexes	17
1. Membres du projet	17

Introduction

Dans le cadre du projet de spécialité de quatrième année, nous avons décidé de concevoir un jeu vidéo : **Mirage's Journey**. Ce rapport expliquera les grandes lignes du projet : le bilan de sa réalisation, de son organisation et une rétrospective par rapport aux attentes énoncées lors de la partie Gestion de projet.

1. Principe du jeu

Le jeu consiste à diriger deux personnages évoluant sur une carte en 2D à explorer. Il se déroulera en temps réel. Le scénario se déroulera de manière linéaire et alternera entre plusieurs planètes où le joueur pourra vivre des aventures et explorer des mondes inconnus.

Le thème du jeu est centré sur l'exploration spatiale : il raconte l'histoire de deux explorateurs, un scientifique et un chien parlant, qui tentent d'aller là où aucun humain n'a jamais mis les pieds : la planète d'origine des chiens. Au fil de leurs pérégrinations, ils atterrissent sur des régions inexplorées, sauvages, parfois sans vie. D'étranges robots, qui ont dissuadé les anciens explorateurs humains de pousser plus loin leur voyage, essaient de les repousser.



2. Motivation

Nous avons fait ce choix pour plusieurs raisons. Tout d'abord, la création d'un jeu vidéo regroupe de nombreux domaines de l'informatique : outre le développement du jeu en lui-même, la gestion d'une base de données est nécessaire pour stocker les objets et caractéristiques, le développement web nous permettra de mettre notre jeu en ligne.

La gestion d'un tel projet demande également beaucoup d'organisation, car les tâches sont très variées et touchent de nombreux domaines. De plus, avec une équipe de 6 personnes, nous devrons nous assurer de la cohérence du travail de chaque membre à chaque étape. *Mirage's Journey* nous permettra donc d'en apprendre plus sur la gestion de projet, ses outils, ses difficultés et ses avantages.

Enfin, le jeu vidéo est un domaine qui nous passionne tous et que nous connaissons bien. Que ce soit d'un côté technique, artistique ou de game design, nous avons acquis de nombreuses compétences nécessaires pour la création d'un jeu vidéo.

3. Contexte du projet

Contexte général : une première approche du monde du jeu vidéo

Ce projet de spécialité se situe dans le contexte de notre 2e année du cycle ingénieur. Il vise à montrer et développer notre savoir-faire en terme d'organisation de projet et de développement en équipe.

Nous attendions un projet motivant qui pourrait nous en apprendre plus sur le domaine du jeu vidéo. A travers les tutoriels que nous avons étudiés, nous avons eu un aperçu du type développement qualifié *d'indépendant* : les jeux vidéos indépendants sont réalisés par des petits groupes de développeurs, contrairement aux jeux dits AAA créés par des équipes de centaines de personnes.

Une méthodologie pour le premier grand projet de notre cursus

Avant le début de la réalisation technique du projet, nous nous sommes intéressés à la méthodologie grâce à la partie Gestion de projet. Nous avons ainsi défini une expression de besoin pour notre jeu, ainsi qu'un planning à l'aide d'un diagramme de Gantt. Nous avons également essayé de prévoir les risques possibles tout au long du projet afin de pouvoir les éviter ou trouver des solutions en cas d'imprévu.

Les responsables des différentes parties du projet ont été nommés d'après leurs expériences précédentes et leurs connaissances techniques, afin de donner à chacun d'entre nous une première expérience du management dans le cadre d'un projet.

Nous avons été confrontés à plusieurs challenges, dont un sixième membre qui s'est greffé au projet après la partie Gestion de projet. Nous avons su exploiter cette situation au mieux : au final, en passant plus de temps sur la gestion, nous avons vu notre charge de travail individuelle diminuer et la qualité du projet augmenter.

Un point très important de ce projet, est qu'il n'a pas été purement technique, nous avons eu dû réaliser des graphismes pour le jeu (grâce au talentueux M. Nguyen), des musiques d'ambiance (grâce à M. Manscour) et des sons pour les éléments du jeu (grâce au frère de M. Mayaki). Nous avons également eu à réfléchir sur des interfaces, afin de les rendre les plus intuitives possible, par exemple pour le site internet. Enfin, outre les aspects esthétiques, l'aspect organisationnel a été un pilier du projet, même après le début de la partie technique.

En terme de ré-exploitation, il est possible pour des étudiants de reprendre le projet et de le développer, afin de proposer une version complète et jouable au grand public. Il n'est pas prévu de commercialiser le jeu, car son contenu est trop faible pour présenter un quelconque intérêt commercial : payer les licences Steam, AppStore... générerait plus de coûts que les revenus du jeu.

Bilan des réalisations

1. Le site web, plateforme de présentation et d'hébergement

Le jeu est hébergé sur un site web, où le joueur pourra s'inscrire et ainsi garder les sauvegardes en ligne. Une base de donnée permettra de gérer les sauvegardes des différents comptes. Une version anglaise du site est proposée. On peut le retrouver à cette adresse : <http://hebergement.u-psud.fr/mirage> et son code est disponible en ligne à l'adresse suivante : <https://github.com/Drakyll/Mirage-s-Journey>.

Le site a été développé à l'aide des technologies HTML, CSS et Javascript en front end et PHP en back end. Aucun framework n'a été utilisé afin d'essayer de familiariser les non-initiés avec les bases de ces langages.

Le site comporte plusieurs fonctionnalités :

- l'inscription sur le site, nécessaire pour jouer au jeu
- la connexion qui permet d'accéder aux pages de téléchargement du jeu, d'information sur l'équipe et à la page d'information du compte actuellement connecté.
- télécharger le jeu, ce qui permet de jouer chez soi.

Dans d'anciennes versions du site, on pouvait jouer au jeu directement sur le navigateur, pour des raisons de fluidité du jeu, nous préférons maintenant laisser l'utilisateur télécharger le jeu (qui reste modeste en taille en tant qu'exécutable).

On peut identifier plusieurs grandes étapes touchant à la création du site :

- Création de la structure interne du site selon le modèle MVC
- Implémentation d'un système de comptes utilisateurs et de connexion sécurisée
- Création de la parallaxe pour la page d'accueil (l'aspect esthétique étant très important pour la partie marketing du jeu)
- Implémentation du système de traduction en Javascript (l'ajout de langue est maintenant facilité et ne nécessite pas de re-chargement de la page)
- Finalisation du site au niveau graphique

Au niveau du projet, le site a été un aspect relativement peu coûteux en investissement humain, en effet seul deux personnes ont eu besoin de travailler dessus dont une seule à temps plein pendant environ 1 mois et demi. Nous avons donc pu rapidement récupérer des ressources humaines pour travailler sur le jeu.

Cependant, le site reste un point essentiel du projet, puisque si nous avions voulu commercialiser le jeu, un site de bonne qualité et attrayant est la meilleure communication qu'on puisse demander. Nous sommes assez fier du rendu de notre site qui, selon nous, est professionnel autant par l'aspect que par la qualité du code (sécurisation des connexions par token/hash, restriction d'accès aux pages selon le statut ...).



Site web : capture d'écran de la page d'accueil du site.

2. Le jeu : ce qui a été fait

Unity et développement du jeu

Dans le monde du jeu vidéo, deux moteurs de jeu sortent du lot : Unreal Engine et Unity. Cela s'explique par la qualité de leur rendu, mais surtout par leur accessibilité : beaucoup de choses sont facilitées comme la gestion de collision ou encore les animations. Unreal Engine étant un moteur de jeu plus spécialisé dans le développement de jeux 3D, il nous a paru plus pertinent de travailler sur Unity qui est plus adapté pour un jeu en 2 dimensions. De plus, Unity est totalement gratuit d'utilisation lorsque l'on génère pas de profit et propose un moyen efficace de travailler en équipe avec un outil de versionnage intégré.

Un autre point important était la possibilité d'exporter le même code sous plusieurs plateformes. Ainsi, on a la possibilité de laisser le jeu (transpilé en WebGL) sur le site ou disponible au téléchargement (compilé en C#) pour ceux qui n'ont pas une très bonne connexion. Enfin, Unity travaille avec des *GameObjects* qui sont indépendants (sans main ou classe principale). On crée des objets qui évoluent indépendamment dans une scène. Cela facilite énormément le développement à plusieurs : on peut plus facilement se séparer les tâches et ne pas avoir de problème de conflits de code.

En revanche, même si Unity facilite énormément de choses, d'autres tâches sont beaucoup plus compliquées à implémenter. C'est une façon très unique de programmer qui peut parfois être déstabilisante. En entraînant nos capacités d'adaptation, l'expérience procurée par Unity sera sans doute très utile pour notre carrière professionnelle.

Lancement du jeu

Lorsque le joueur lance le jeu, On accède à une première page de connexion. Le joueur peut créer un compte ou se connecter avec un compte déjà existant. Une fois connecté, il a accès à un autre menu dans lequel il peut lancer une nouvelle partie, charger une sauvegarde, quitter le jeu ou encore voir les crédits du jeu. (La page d'option est vide pour l'instant, mais dans une version future, le joueur pourra customiser les touches pour lancer les compétences, pour accéder au menu et pour augmenter ou diminuer le son du jeu.)



Mirage's Journey : menu principal.

Il peut ensuite visionner une cinématique (scénaristiquement incomplète) pour en savoir plus sur l'histoire du jeu. La partie commence alors. Pour l'heure, il n'y a pas de tutoriel mais l'ajout de quelques boîtes de dialogue pour guider le joueur serait judicieux.

L'exploration se base sur une carte animée entièrement peinte à la main. Le joueur peut progressivement découvrir la carte et voir ce qui l'entoure grâce à l'implémentation d'un brouillard de guerre.



Mirage's Journey : écran de jeu au commencement.

Système de combat et de compétences

Le système de combat est basé sur la dualité entre l'homme et le chien. L'être humain possède un arsenal plutôt offensif tandis que le chien dispose d'un bouclier énergétique qui bloque les tirs ennemis. Le chien aura potentiellement des capacités offensives limitées (seulement au corps à corps) afin de pouvoir effectuer des manœuvres comme l'attaque à revers.



Mirage's Journey : joueur en train de combattre des robots.

Les boucliers ont une certaine charge qui diminue à chaque coup bloqué. Le positionnement du chien permettra donc de protéger l'humain qui attaque automatiquement les ennemis. Il faudra toutefois veiller à la santé du chien qui diminue rapidement lorsque le bouclier a été brisé. Ce système permettra de récompenser les joueurs alliant efficacement esquives et blocage de tirs ennemis.

Le joueur aura également accès à des capacités se rechargeant au fil du temps (cooldown), lui permettant de débloquer certaines situations épineuses. Ces capacités sont débloquées via l'arbre de compétence : les feuilles de l'arbre sont débloquées à chaque niveau.



Mirage's Journey : menu des compétences.

Système de quêtes

Afin de gagner un niveau, le joueur devra compléter des quêtes ou tuer des monstres. Pour progresser dans l'histoire, il peut accepter des quêtes en allant parler à un personnage non-joueur (PNJ). Il existe trois types de quêtes : tuer un ennemi désigné (un boss par exemple), prendre un objet ou atteindre une zone prédéfinie. Ce système devrait suffire à coder la plupart des interactions entre les PNJ et le joueur.



Mirage's Journey : le système de quête, une quête est représentée par un point d'exclamation. Pour commencer une quête, le joueur doit entrer en contact avec le PNJ.

Système de sauvegarde

Il est également possible de sauvegarder ou de charger sa progression. Voici un exemple d'un fichier de sauvegarde :

A screenshot of a file editor titled "Edit file" showing a binary file named "/public_html/alex/Binaire/16/Mathias". The file contains a large amount of binary data represented as a sequence of numbers. On the left, there is a list of files in a file manager interface, with "Mathias" selected. The file manager has a red header bar with icons for upload, download, search, etc. The main area shows the file content with a scroll bar. At the bottom right, there are "SAVE & CLOSE" and "SAVE" buttons.

Base de données : un fichier binaire généré pour sauvegarder une partie.

Nous avons fait le choix de sauvegarder toutes les données dans un même fichier. Le principe est donc de créer un fichier par sauvegarde.

Toutes les données à sauvegarder sont rassemblées dans une seule classe conteneur. Cette classe est sérialisée puis transformée en FileStream, lui même transformé en chaîne de caractères, qui est ensuite envoyée dans une requête web grâce à la classe WWW d'Unity et est ensuite traitée par un script PHP. Il est possible de donner le nom du fichier txt généré et le nom du joueur en paramètre, pour ainsi avoir plusieurs sauvegardes par joueur.

Pour pouvoir récupérer la sauvegarde, on effectue l'action inverse. Un fichier PHP lit le fichier txt demandé et est ensuite traduit en octets. On récupère à la fin la classe conteneur avec toutes les données du jeu, permettant ainsi de restaurer le jeu à l'état sauvegardé.

3. Réalisations en matière de game design

Le design du jeu a connu beaucoup d'évolutions au fil du temps : en effet, notre méthode de travail permettait de nous adapter tout en découvrant les possibilités qu'offrent Unity. Au cours de notre avancée, nous avons dû adapter nos attentes en matière de game design pour mieux coller à la technologie utilisée, mais aussi afin de rendre le jeu plus intéressant pour l'utilisateur.

Par exemple, nous avons décidé de passer d'une interface purement 2D en vue du dessus à une interface en 2,5D avec vue inclinée (le personnage peut passer derrière des objets tout en étant vu du dessus). Cela a profondément changé la nature du jeu, ce qui a modifié toutes les tâches liées à celui-ci.

Bilan de la conduite de projet

1. Comparatif des différences de gestion

Voici un tableau regroupant les principaux éléments de management et de gestions ayant différé avec ce qui a été prévu au S7 :

Élément prévu	Travail effectif	Raison du changement
Respect du Gantt chart et des jalons du projet	Nombreux retards par rapport au planning, beaucoup de tâches enlevées et ajoutées	Pas de créneaux stable, méconnaissance des techniques et délais de développement sur Unity
Tests rigoureux	Tests rudimentaires	Manque de temps, nous avons dû donner la priorité à d'autres tâches
Réunions régulières	Communication et développement à distance	Organisation plus efficace au sein de notre groupe, bonne utilisation des moyens de communication

2. Les points positifs du projet : un jeu fonctionnel

Points positifs concernant le côté technique

- Code propre et fonctionnel. Bonne utilisation des mécaniques/préceptes d'Unity (prefabs, gameobject, animators etc. au lieu tout coder nous même à chaque fois)
- Un site internet sécurisé, permettant à l'utilisateur de s'inscrire en toute confiance.

Il existe seulement une version Windows pour l'instant, mais Unity permet d'exporter sur de nombreuses plateformes, l'adaptation multi-plateforme n'est donc pas un problème.

Points positifs concernant l'organisation mise en place

Notre projet est entièrement centré autour de la spécialité informatique. Ainsi, il ne possède pas une organisation aussi complexe que d'autres. En effet, le développement d'un logiciel (basé sur nos propres besoins et non ceux d'un client) peut se faire de manière très flexible. En fonction de la progression du projet et de notre emploi du temps du S8, nous avons pu nous adapter de manière rapide à tout changement, à travers une communication continue et régulière.

Tous les fichiers utilisés pour le projet seront éventuellement mis en ligne sur une plateforme libre d'accès. Cela permettra à tous d'accéder au code du jeu et du site internet afin qu'ils puissent être réutilisés (dans l'esprit du partage open source).

Outils de gestion de projet

Lors du premier semestre, nous avons appris à utiliser divers outils de gestions de projet, tels que le Gantt chart, la matrice RACI, la matrice des risques, les fiches de réunion...

Nous avons utilisé des plateformes et logiciels afin de faciliter le développement :

- un serveur Discord actif et complet permettant de partager les éléments du projet. Discord est un outil très puissant, étant donné qu'il permet aux membres du projet de lancer des visio-conférences gratuites, avec partage d'écran intégré. De plus, il est facile d'utilisation et la majorité des membres de l'équipe l'utilise dans la vie quotidienne et est familier avec son utilisation.
- un dossier Google Drive pour partager tous les documents, images, ressources... en temps réel et perpétuellement à jour pour tous les membres. L'utilisation de Google Drive a évité d'avoir à faire nous-même le versionnage de chaque document, étant donné que tout élément présent sur Drive possède un versionnage complet et daté.

Sans tomber dans l'exercice scolaire (utilisation de Slack par exemple, qui ne nous aurait pas été utile puisque très similaire à Discord), nous avons réussi à maîtriser nos outils et à tirer le meilleur d'eux.

3. Les points négatifs : un jeu qui reste incomplet

Difficultés techniques

N'étant pas familier avec l'environnement de travail Unity, nous avions des attentes un peu trop optimistes quant à la quantité d'éléments à introduire dans le jeu. Nous ne réalisions pas que certaines tâches seraient complexes à mettre en place et difficilement automatisable (créer une compétence ne permet pas forcément de créer facilement les futures compétences).

Parmi ces aspects, on retrouve :

- le pathfinding, basé sur l'algorithme de plus court chemin A* a été très long à mettre en place de manière propre. En effet, Unity dispose de beaucoup d'exemples de pathfinding pour les jeux en 3 dimensions, ce qui a rendu la recherche d'informations pour implémenter un algorithme similaire en 2D plus difficile.
- les compétences, trop ambitieuses dans le game design, n'ont pas pu toutes être implémentées dans le jeu de par leurs grandes différences. Après avoir implémenté un modèle permettant de créer plusieurs compétences d'un coup, certaines autres compétences devenaient extrêmement complexes à intégrer au système.

- l'intelligence artificielle des ennemis, qui est restée très simple dans le jeu faute de temps.

Test et quality control

Malheureusement, nous n'avons pas eu assez de temps pour effectuer la partie test du projet.

Nous avions prévu un système de logs en cas de bugs, permettant de détecter des bugs éventuels. Ces logs auraient pu être stockés sous un format texte, par le même mécanisme que le système de sauvegarde que nous aurions ré-exploité pour les écrire. Les utilisateurs pourraient également soumettre un formulaire, nous permettant en plus d'avoir un feedback supplémentaire. Ainsi, des statistiques auraient pu être effectuées sur la fréquence des bugs, afin de mettre en place une stratégie de patchs correctifs cohérente avec la gravité de ces bugs.

Sur un projet à plus long terme, il serait judicieux d'avoir quelqu'un concentré sur les tests et la partie "post-production" du projet. Cela permettra de faire tester le jeu à des personnes extérieures dites "naïves", qui pourront donner un avis plus objectif sur le projet.

Difficultés de gestion

La principale difficulté dans la gestion de ce projet vient de notre méconnaissance préalable des technologies et du monde du jeu vidéo. Puisque nous effectuions ce projet afin de découvrir ce milieu, nos estimations et attentes n'étaient pas现实的, et ne pouvaient pas l'être. Nous avons donc dû recourir à une pseudo méthode agile sur le tard afin de pouvoir réagir à chaque nouvelle découverte.

L'expression de besoin a constamment évolué au cours du projet et la découverte de choses rendues possibles ou impossibles par Unity a rythmé l'avancée de celui-ci. Ainsi, le diagramme de Gantt, comme prévu, n'a absolument pas été respecté (90% des tâches réalisées sur le projet n'étaient pas dans le Gantt et le même pourcentage de tâches prévues ont été supprimées).

Un autre problème est celui de la disponibilité de l'équipe, les créneaux alloués au projet étant largement insuffisant, il a fallu s'organiser pour télé-travailler en permanence. De plus, ce projet tombe dans ce qu'un de nos professeurs a appelé "l'année la plus chargée de votre vie", nous avons donc dû combattre le fait que nous avions constamment des deadlines plus urgentes que ce projet.

Dans l'ensemble, nous avons réussi à surmonter ces difficultés en opérant de manière agile et en communiquant de manière efficace entre nous, le temps nous a cependant manqué sur la fin du projet, ce qui rend le produit fini inférieur à nos attentes.

Conclusion et perspective

1. Rétrospective par rapport aux attentes en début de projet

Nous avons pris le temps de réfléchir à ce qui a été fait dans le projet par rapport à ce qui avait été prévu de base par l'expression des besoins.

Tableau des différences concernant le jeu

Élément prévu	Travail effectif	Raison du changement
5 niveaux, 5 planètes	1 grand niveau (démo)	Manque de temps
Jeu en 2D vue de dessus	Jeu en 2.5D vu penchée	Style artistique plus adapté à la 2.5D
5 compétences	3 compétences	Difficulté de coder certaines compétences
Des cinématiques, une histoire développée, des voix pour les personnages	Une histoire simplifiée, beaucoup d'éléments de l'histoire passés à la trappe	Manque de temps
Une interface développée avec plusieurs boutons d'option en bas de l'écran	Une interface simplifiée, plus claire et plus facile d'utilisation, s'adaptant à tout type d'écran	Changement de philosophie du design

Nous pouvons noter un écart évident entre ce que nous voulions faire et ce que nous avons fait :

- un seul niveau au lieu de 5 : cela est dû à une mauvaise estimation du temps que prendraient le design des planètes et la réalisation des éléments visuels (décors, ennemis, personnages...).
- moins de diversité dans les décors et les personnages : par exemple, l'astronaute a mis plusieurs jours à être dessiné car l'animation de marche devait se faire sous plusieurs angles.
- aucun équilibrage : le jeu est par moment trop facile, et par moment trop difficile. Il faudrait un véritable travail sur le game design pour le rendre réellement jouable et abordable par tous.

2. Perspectives pour un développement futur

Nous avons tenté de suivre au maximum les préceptes phares du développement sur Unity, à savoir la découpe stricte des codes de chacun en scripts et préfabs distincts. Ainsi, toutes les parties du code sont les plus indépendantes possible et la reprise du projet (ou la correction de bugs éventuels) en est facilitée.

Le gameplay a été beaucoup simplifié afin de pouvoir rendre une version de démonstration fonctionnelle, malgré tout, de nouvelles compétences pourraient relativement facilement être implémentées, car un code générique existe déjà.

Parce que nous n'avions pas prévu au départ de remettre le projet entre d'autres mains, aucune documentation complète n'a été faite. Pour ceux qui souhaiteraient se pencher sur le code, ce rapport est un bon début pour prendre connaissance de toutes les fonctionnalités du programme. Le code est disponible également sur GitHub mais n'est pas forcément à jour, car nous utilisions l'outil de versionnage interne à Unity, plus efficace pour travailler en groupe.

En se basant sur notre code actuel, une équipe motivée pourrait très facilement reprendre le jeu et le finir relativement rapidement. Une structure intelligente ayant été mise en place, la reprise de ce projet serait plutôt centrée sur l'évolution du game design afin de finir de donner au jeu sa touche d'unicité. Des étudiants en game design, animation ou scénaristes pourraient reprendre les concepts de base et développer le jeu à partir de là.

Il reste cependant de nombreuses lacunes du projet à combler : un système de test, de logs et de formulaires de bugs pourrait être implanté par-dessus le jeu actuel. En effet, plus le jeu se complexifie, plus il a besoin d'être débuggé et testé. C'est pour cela qu'un tel système sera très utile dans le futur : sans doute devra-t-il être la première chose à implémenter pour une équipe qui reprendrait le projet.

3. Conclusion

Ce module nous a permis de mieux comprendre le déroulement d'un semestre de développement et l'organisation d'un projet à long terme en informatique.

Il a été très intéressant de réussir à coordonner une équipe de 6 personnes sur un projet de cette envergure, la plupart d'entre nous n'ayant jamais réalisé de management d'équipe, nous avons tous appris au contact les uns des autres et ressortons grandis de cette expérience. Nous avons également appris un nouveau langage, le C#, ainsi qu'une nouvelle manière de créer des logiciels puisque Unity dispose d'une approche particulière pour la création de jeux vidéo et d'interfaces graphiques.

Il nous a également permis d'effectuer une réflexion sur les perspectives d'un projet : en effet, on ne pense pas tout le temps à rendre notre code exploitable par une équipe future.

Or, c'est une pratique courante en entreprise comme en stage et que nous devrons apprendre à faire pour tout ce que nous entreprenons dans notre vie professionnelle.

Remerciements

- Mme Valérie Guimard, notre responsable GP, pour son aide lors de la phase préliminaire du projet
- M. Aurélien Max, notre responsable technique, pour ses conseils tout au long de l'année
- M. Corentin Manscour, notre conseiller game design et compositeur des musiques du jeu
- M. Jabdoul Mayaki qui nous a aidé à implémenter les bruitages du jeu
- Ainsi que les membres du projet W.I.L.D. pour nous avoir soutenus et accompagnés dans cette aventure.

Annexes

1. Membres du projet

Voici les membres de l'équipe Objectif Lune, leur rôle et leur contribution. Mis à part leur mission principale, tous les membres ont contribué au code Unity.

Nathan BONNARD : Responsable Unity

Réalisation des compétences, attaques, du système de quête et de sauvegarde.



Alban DESCOTTES : Responsable IA

Réalisation du pathfinding et de l'IA des ennemis.



Morgan FEURTE : Responsable web

Réalisation du site web, du système de compte et de login et de l'intégration du jeu au site.



Alexandre KIRILLOV : Responsable base de données.

Réalisation du système de sauvegarde et de la mise en ligne de la base de donnée.



Oummar MAYAKI : Assistant Chef de projet, responsable Gestion de Projet et Game design. Réalisation des documents de gestion de projet.



Hien Minh NGUYEN : Chef de projet, responsable graphismes et animations. Réalisation des éléments visuels de l'interface, du jeu et du site web.



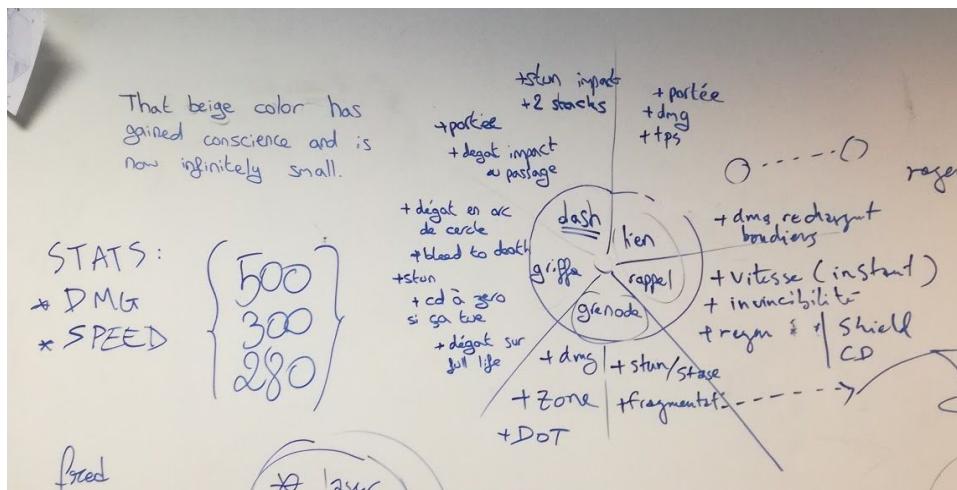


Fig. n° 1 : brouillon préliminaire des compétences. La structure était radicalement différente de ce qui existe actuellement.

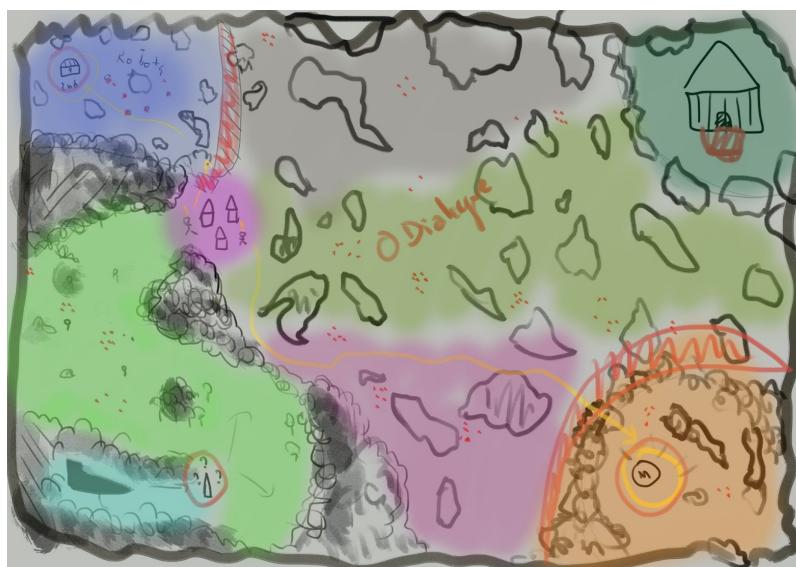


Fig. n° 2 : brouillon préliminaire du premier niveau. Nous pouvons voir la division des zones du jeu, des éléments importants et des zones accessibles seulement après l'acceptation de la quête (grillagé rouge).

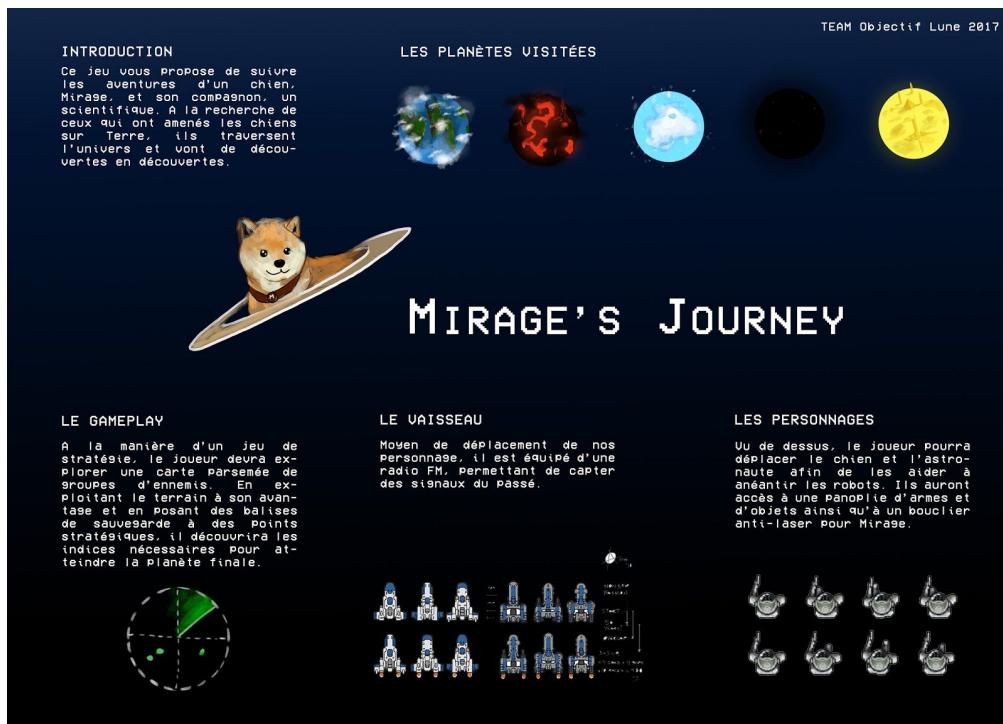


Fig. n° 3 : Poster prévisionnel du projet, mettant en avant les principales caractéristiques du jeu. On peut y voir des éléments qui ne sont pas restés dans la version finale : le système de sauvegarde par balise, les cinq planètes ou encore le thème de la radio.