Aegina : rapport de soutenance n° 1

Florian Amsallem (amsall_f) , Théo Issarni (issarn_t),
Julien Mounier (mounie_a), Romain Mounier (mounie_r)

AIM^2
Aegina

Table des matières

Introduction

Ce rapport contient l'état du développement du jeu Aegina au 16 mars 2016. Nous détaillons d'abord les différents éléments composant le jeu. Nous expliquons ensuite, chronologiquement, les développements réalisés par les différent membres de l'équipe. Les développements prévus d'ici la seconde soutenance sont ensuite présentés.

1 Univers du jeu

Aegina est un jeu de survie dans lequel le joueur incarne Ille un orphelin aux capacités physiques hors normes. Le joueur sera amené à découvrir les mystères de l'univers dans lequel il évolue. Dans cette partie, nous vous révélons ces dits secrets.

1.1 Aegina

Le monde d'Aegina est constitué d'une quasi infinité d'îles volantes se trouvant au-dessus d'un vide béant appelé les *abysses*. Ce monde, contrairement au nôtre, est instable. Il ne partage pas nos lois de l'équilibre et de nombreux phénomènes étranges s'y produisent sans que l'on puisse les expliquer. La présence des *abysses* n'étant qu'un de ces phénomènes.

L'instabilité de ce monde est principalement due à la présence de cristaux sur certaines îles. Ceux-ci sont capables de déchirer les trames de l'espace et du temps permettant de réaliser des miracles comme le retour dans le temps ou la téléportation. La puissance des cristaux est amplifiée lorsqu'ils sont activés. Malgré toutes ces instabilités un écosystème s'est tout de même développé. Des animaux tels que des lapins, papillons et moutons peuplent les îles d'Aegina. Mais d'autres espèces plus agressives et plus exotiques sont apparues suite à l'instabilité du monde. Parmi elles nous trouvons les *slimes*. Un instinct naturel pousse toutes ces créatures agressives à s'attaquer aux cristaux lorsqu'ils sont activés pour les empêcher de rendre le monde plus instable qu'il ne l'est déjà.

L'activation des cristaux ne peut être réalisée que par un être doué de raison. On pourrait donc penser qu'en l'absence d'un tel être en Aegina et avec une telle armée de créatures prêtes à défendre ce monde celui-ci aurait trouvé une certaine forme d'équilibre. Mais c'était sans compter la présence d'envahisseurs, d'êtres extérieurs à ce monde qui utilisent celui-ci à des fins barbares, les êtres humains. Bien que ne vivant pas dans ce monde et n'ayant aucune information sur celui-ci, les êtres humains on trouvé un moyen de « téléporter » leurs déchets dans un néant qui n'est autre qu'Aegina. Lorsque les déchets arrivent directement dans les *abysses*, ils n'ont aucune influence sur la stabilité d'Aegina. S'ils tombent sur une

île, ils peuvent représenter un apport en ressource pour Aegina. L'aspect néfaste pour Aegina provient d'une sorte très particulière presque barbare de déchets.

Certaines autorités humaines utilisent la « téléportation » pour se débarrasser d'autres humains. Malheureusement une fois dans le monde d'Aegina ceux-ci tentent désespérément de survivre. Ils finissent en général par arriver à activer les cristaux dont les pouvoirs quasi divins leur permettent d'échapper à la faim et à la mort. Certains ont alors voyagé d'île en île, créant quelques infrastructures telles que des ponts, laissant leur trace dans ce monde, mais l'amenant petit à petit à sa fin sans le savoir. Heureusement pour ce monde, jusqu'à présent tous ont fini par périr sous l'assaut des créatures locales ou se sont suicidés en se jetant dans les *abysses* ne pouvant pas rentrer chez eux. Mais il arrivera bien un jour où l'un d'eux emportera avec lui ce monde dans sa chute.

1.2 Ille

Ille est le personnage que le joueur incarne dans Aegina. De père et de mère inconnus, Ille survit depuis son plus jeune âge seul dans une forêt. Bien que n'ayant pas d'amis, sauf si l'on peut considérer un écureuil comme un ami, Ille est très chaleureux et amical. Cependant, Ille n'est pas un enfant sauvage comme son histoire le laisse penser, il est légèrement civilisé et a lui-même choisi de vivre en autarcie à cause de sa particularité. En effet, depuis sa plus jeune enfance, Ille possède une force surhumaine (et un certain talent pour la construction). Il a lui-même construit sa maison à l'âge de neuf ans et est capable d'abattre des arbres comme un bucheron chevronné depuis qu'il sait manier une hache. Cependant, cette force effraie facilement les enfants de son âge et même les adultes, mais malgré tout Ille vit de façon paisible. Les repas de Ille ne sont pas très extravagants, un jour cela peut être un sanglier et le lendemain un simple champignon. Une seule chose concernant sa nourriture lui importe, cette dernière doit être préparée car Ille tient toujours au fond de lui à se rapprocher des siens et l'action de préparer la nourriture lui donne l'impression d'être vraiment humain.

Son seul défaut, si l'on peut considérer ceci comme un défaut, est d'être simplet. C'est d'une certaine façon une chance car c'est ce qui à permis à l'aventure d'Aegina de commencer.

Lors d'un hiver très rude, Ille n'eut pas d'autre choix que de voler de la nourriture dans un village avoisinant. Malheureusement, la discrétion n'étant pas son fort, il fut surpris en plein méfait et un garde tenta de l'arrêter. Ille tenta de passer le garde, mais il ne contrôla pas sa force et, en le bousculant, il le tua. Illie n'avait pas réalisé que le garde était extrêmement affaibli par le froid et la faim. Ille fut alors poursuivi pour meurtre et rapidement arrêté car malgré sa force Ille restait humain. Il fut condamné à mort et envoyé dans le néant qui n'était autre qu'Aegina. C'est à son réveil que le jeu commence.

Cependant, Ille n'est pas forcement seul en Aegina. Il pourra rencontrer d'autres personnes car la mort par envoi dans le néant est une pratique assez courante. Retrouver des amis dans ce monde dangereux peut être une bonne chose car dans Aegina le nombre de vos amis fait en partie votre force.

1.3 Le cristal

Comme déjà dit précédemment, les cristaux dans Aegina sont un élément très important car ils possèdent d'énormes pouvoirs et sont responsables de l'instabilité du monde. Ils sont en apparence indestructibles, mais il est toutefois possible de les rendre inactifs. Pour s'activer ils ont besoins d'absorber divers minerais ce qui arrive très rarement naturellement, mais peut être facilement réalisé par des êtres conscients.

Le cristal semble réagir avec tous les minerais spécifiques du monde d'Aegina ainsi qu'avec de nombreuses créatures qui, en présence d'un cristal actif, perdent toute raison et tentent de le désactiver.

Peu d'informations supplémentaires sont disponibles sur les cristaux. Leur composition est totalement inconnue et personne ne sait d'où ils viennent. Ils ne semblent pas exister depuis la création du monde car leur forme géométriques et leurs pouvoirs en font la création d'êtres conscients possédant une technologie très avancée. Mais il ne s'agit que de suppositions qui n'ont que peu d'importance pour notre histoire.

1.4 Les minerais

De le monde d'Aegina on peut trouver de nombreux minerais qui permettront d'aider le joueur dans son aventure. Certains de ces minerais existent dans notre monde, mais Aegina possède aussi des minerais uniques. Dans cette partie nous allons vous présenter les minerais exclusifs au monde d'Aegina.

1.4.1 Mithril

Le mithril, dans le monde d'Aegina, est un minerai vert clair un peu plus solide que le fer. Assez rare à la surface des îles, on en trouve toujours en petite quantité en sous-sol puisqu'il est l'un des deux minerais nécessaires à la « survie » d'une île. Le mithril a l'étrange particularité de modifier les phénomènes étranges d'Aegina parfois en les amplifiant, d'autres fois en les stabilisant. Il est impossible de savoir comment celui-ci va réagir avant de l'utiliser.

A l'état naturel, il catalyse les instabilités et les relâche sous forme de vagues d'énergie rendant le minerai chaud au toucher. Une fois épuré, le mithril continue de catalyser les

instabilités, mais peut alors relâcher son énergie sous de nombreuses autres formes que la chaleur, comme le ferait un cristal, mais à bien plus faible échelle.

D'ailleurs l'énergie catalysée et stockée par le mithril semble être la même que celle se trouvant dans les cristaux car les deux matériaux peuvent échanger cette énergie. Ainsi quelques chercheurs hérétiques qui se sont trouvés sur Aegina ont supposé que les cristaux étaient formés d'un alliage composé en majorité de mithril. Ces chercheurs ont trouvé la mort quelques jours plus tard, tués par des sangliers après avoir transféré toute l'énergie de leur cristal dans du mithril.

1.4.2 Floatium

Le floatium est un minerai quasi transparent que l'on peut trouver dans Aegina. Les humains qui arrivent dans Aegina le confondent souvent avec du quartz lorsqu'ils en voient dans la roche. Mais celui-ci est bien plus transparent et s'apparenterait plus à du verre ou de la glace. La vrai particularité du floatium se remarque une fois le minerai nettoyé de toute impureté. La première constatation est que celui-ci est bien plus solide que ce à quoi on peut penser, sa résistance égale celle de l'acier, mais surtout il est impossible de le faire tomber par terre. En effet, une fois celui-ci lâché, il flotte.

Ce minerai est encore un mystère d'Aegina car personne ne sait pourquoi il flotte. Il n'est pas plus léger que l'air puisqu'il reste à une distance fixe du sol, mais il ne semble pas s'agir non plus de force magnétique. De toute façon peu d'expériences ont eu lieu sur le floatium car, en Aegina, en plus d'être un minerai très rare, les personnes capables de faire des expérimentations scientifiques le sont encore plus.

La seule information sûre est que le floatium est, avec le mithril, l'un des deux composants permettant aux îles de voler. Cependant, les cristaux et le floatium peuvent réagir entre eux, le cristal semble pouvoir absorber les capacités du floatium tandis que le floatium est responsable de la rotation et de la lévitation du cristal, mais tout cela reste de la théorie.

Néanmoins, il est possible de conjecturer que si un cristal a une trop grande influence sur une île, il pourrait provoquer sa chute. Mais les seules personnes connaissant la vérité sont sûrement mortes avec leurs îles.

1.4.3 Sunkium

Le sunkium est un minerai d'une couleur mauve très sombre. C'est un minerai légendaire qui possède de nombreuses qualités car il est le minerai le plus rare, le plus lourd, le plus sombre, le plus solide, le plus cool, le plus classe, le plus *dark* et le plus unique d'Aegina. Pourtant, malgré sa solidité, il n'est pas si difficile à extraire des roches car il semble repousser

les autres minerais lors de sa formation, ce qui permet de le trouver dans des alvéoles formées par la roche. Cependant une fois extrait son poids reste tout de même un problème pour le transporter.

Mais ce qui en fait le minerai le plus cool, c'est qu'il peut être utilisé très simplement pour la création d'épées ou d'armures. En effet, il a la même température de fusion que le fer et une fois mélangé à du mithril il chauffe et refroidi à une vitesse exceptionnelle. De plus la couleur sombre des armures formées avec le sunkium apporte un côté cool et *dark* à toute personne les portant. Mais malheureusement cette impression ne dure que quelques secondes car rares sont ceux capables de se mouvoir dans une telle armure. La plupart se retrouvent coincés dans celle-ci, incapables de bouger ou de l'enlever, amenant à la mort la plus stupide d'Aegina.

Le sunkium, comme le mithril et le floatium, réagit avec les cristaux mais sa réaction est peut être la plus impressionnante des trois, après tout c'est le minerai le plus cool. Le sunkium change de volume et de masse sous l'influence du cristal mais pas comme tout le monde s'y attendrait car plus son volume diminue, plus il devient lourd. Il se met alors à absorber de la lumière, assombrissant les alentours. Le cristal quant à lui peut absorber le sunkium mais personne ne sait réellement ce que ça lui apporte.

Certaines théories stipulent que les pouvoirs du cristal viendraient de minuscules particules de sunkium qui déformeraient l'espace temps. Mais cette théorie s'appuyant sur d'autres théories qui n'ont pas été prouvées, sa véracité est plus que discutable. Sans la présence des cristaux, le sunkium serait le minerai le plus mystérieux d'Aegina.

2 Réalisation

Maintenant que vous connaissez mieux l'univers d'Aegina nous pouvons vous parler de l'état actuel du project.

2.1 Pré-janvier

2.1.1 Cycle jour/nuit (Julien)

Bien relire pour vérifier que je n'ai pas introduit d'erreur

Cette partie traitera principalement des *skybox* et de l'utilisation que nous en faisons. La *skybox* est la boîte dans laquelle se déroule le jeu. Ses cotés représentent les limites visibles du monde et il est impossible d'en sortir. La *skybox* permet donc de représenter le ciel ou n'importe quel autre élément de dernier plan.

Pour réaliser le cycle jour nuit, nous avons dû déterminer la durée d'une journée que l'on a divisée en 30 phases. Chacune d'elle correspondant à une image de *skybox*. Seulement cette méthode à présenté quelques inconvénients. Entre autre, le changement de *skybox* était trop brusque et les différentes *skybox* n'étaient pas homogènes. C'est pourquoi nous avons dû trouver une autre méthode : la *directional light*.

La *directional light* est un gameobject propose par unity. Son intérêt majeur est que l'orientation de cette lumière influence la *skybox* du monde. De plus nous pouvons choisir la couleur et l'intensité de cette lumière. Pour ce faire nous utilisons une fonction qui convertit la longueur d'onde des rayons lumineux en couleur RGB [1] et nous représentons le spectre solaire à l'aide de la fonction mathématique suivante où λ est la longueur et t le temps :

$$spectre(\lambda, t) = 255 * ((3.25t - 4)\lambda^2 + (-4t + 4)\lambda + (0,08(t - 1)))$$

La figure 1 montre les évolutions du spectre lumineux. L'arc des abscisses représente la longueur d'onde et celui des ordonnées l'intensité. Pour chaque valeur de t (moment de la journée), nous pouvons donc calculer un spectre lumineux. La courbe bleue ($t = 0$) représente le spectre quand le soleil est au zénith et la courbe rouge ($t = 1$) correspond au spectre au moment du coucher du soleil. Les courbes vertes ($t = 0,3$ et $t = 0,6$) montrent quant à elles l'évolution de la courbe. A chaque instant, nous échantillonnons la courbe actuelle en prenant 40 valeurs équiréparties, ce qui nous donne un tableau des couleurs (en RGB) que notre astre est censé émettre. Ce tableau contient environ 40 valeurs or une *directional light* ne possède qu'une seule couleur. Il a donc fallu trouver un moyen pour réunir ces 40 couleurs

1. http ://stackoverflow.com/questions/1472514/convert-light-frequency-to-rgb

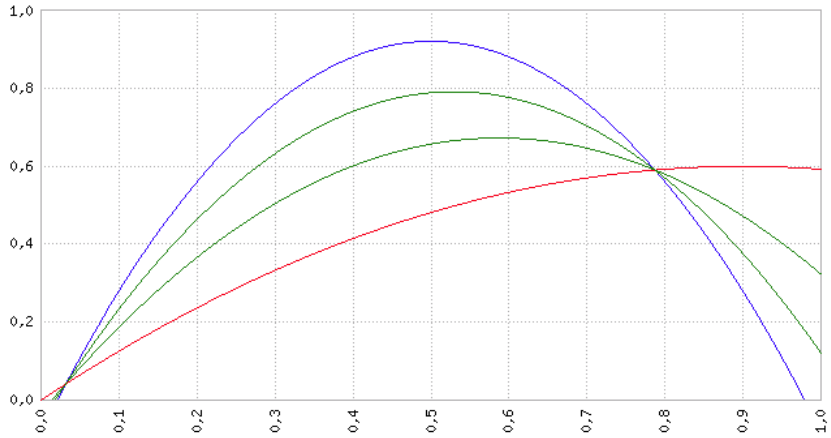

FIGURE 1 – évolution du spectre lumineux en fonction du temps

en une seule. Pour ce faire, nous parcourons toutes les valeurs du tableau et nous formons une nouvelle couleur en choisissant les plus grandes valeurs de rouge, de vert et de bleu rencontrées.

Nous utilisons cette même fonction en fixant $\lambda = 0$ pour définir l'intensité lumineuse.

Nous avons ensuite implémenté la rotation et le déplacement de la *directional light*. Ce fut assez simple car nous étudions les mouvements circulaires en cours de physique. Il nous a donc suffit de copier les équations paramétriques horaires en base cartésienne. C'est-à-dire :

$$x(t) = Rcos(wt)$$

$$z(t) = Rsin(wt)$$

Où R est le rayon de l'orbite et w la vitesse angulaire.

Ainsi, la *directional light* suit une orbite circulaire autour du point de coordonnées (0,0,0). L'orientation de la *directional light* est définie grâce à la fonction *lookat* qui oriente un *gameobject* vers un autre *gameobject*, le point (0,0,0) dans notre cas.

Un procédé similaire est utilisé pour la première scène du jeu. En effet, dans cette scène, la caméra orbite autour d'une île et nous nous servons de la fonction *lookat* pour orienter la camera vers l'île. Mais, contrairement au cas de la *directional light*, le déplacement utilise les propriétés de la base polaire. En effet, comme l'objet est toujours orienté vers le centre de son orbite, il suffit de lui appliquer une vitesse constante pour qu'il décrive une trajectoire circulaire.

2.1.2 Class *Item* (Romain)

Pour pouvoir gérer les ressources que possède un joueur, nous avons dû créer une classe que nous avons tout simplement choisi d'appeler *Item*.

La classe *Item* permet de créer un patron d'objet ne contenant que des constantes immuables qui sont le nom (dans chacune des langues disponible, pour le moment il s'agit du français et de l'anglais), la description (elle aussi dans chacune des langues proposées), un identifiant unique, une quantité maximale par personnage, une texture 2D et un élément correspondant à l'objet 3D. Cependant, le joueur peut hypothétiquement posséder des objets en nombreux exemplaires. Les objets que celui-ci possède sont donc une nouvelle classe appelée *ItemStack* contenant une quantité (inférieure à la quantité max de l'objet) et un pointeur vers l'objet qu'il représente. Ainsi, les informations communes à plusieurs exemplaires d'un même objet ne sont pas dupliquées, seules la quantité de l'objet peut changer. Il y a alors un gain de mémoire.

2.1.3 Animation 3D (Théo)

Ille est un personnage que nous avons téléchargé depuis l'asset store de unity qui possédait initialement deux animations, « marcher » et « rester immobile ». Cependant ces deux seules animations ne nous suffisaient pas. Nous avons alors décidé de faire les animations de nos personnages nous-même. Le premier problème que nous avons rencontré est que notre personnage n'était pas accessible depuis notre version de *blender*, trop récente pour le modèle 3D que nous avons. Il nous a donc fallu trouver une ancienne version de *blender* compatible avec notre model 3D. Depuis nous essayons d'améliorer nos animations dans le but d'être le plus réaliste possible mais sans expérience dans ce domaine il est difficile de faire des animations convenables et celles-ci sont systématiquement remises en questions par les membres de l'équipe et donc modifiées.

2.1.4 Déplacement du personnage et de sa camera (Théo et Julien)

Le joueur détermine les déplacements de son personnage. Celui-ci peut avancer (avec la touche « w » ou « up »), reculer (avec la touche « s » ou « down »), s'orienter vers la droite (avec la touche « q » ou « right ») et s'orienter vers la gauche (avec la touche « d » ou « left »). Notre personnage peut courir si ce dernier se déplace et que l'utilisateur appuie sur la touche « shift », il peut aussi sauter si l'utilisateur appuie sur la barre d'espace.

Le saut à d'ailleurs été un élément problématique à implanter car notre personnage ne devait pouvoir sauter qu'à partir du sol pour éviter qu'il ne s'envole en enchaînant des sauts

dans les airs. Nous avons donc créé des restrictions permettant au personnage de ne sauter que depuis certains éléments du décors (qui regroupent presque tous les éléments).

A partir de ce moment notre personnage était devenu opérationnel mais à quoi cela servait-il si nous ne pouvions pas le voir ?

Nous avons alors décidé de créer une camera qui suivrait notre personnage où qu'il aille. Cette caméra n'est pas une simple caméra fixe, elle possède des caractéristiques variées grâce auxquelles une grande liberté est accordée au joueur. Elle s'oriente grâce à la souris et même si on dit qu'elle suit le joueur, l'orientation des mouvements du joueur ne dépendent que de l'orientation de la caméra. Par exemple, faire reculer le joueur le fait en réalité se diriger vers le caméra.

La vue à la troisième personne permet au joueur de voir son personnage de coté, de devant ou encore de derrière. Tout cela en étant proche ou éloigné de son personnage, ce qui permet aussi d'avoir une vision plus générale de l'environnement. L'éloignement de la caméra et du personnage se contrôle avec la molette de la souris, il est donc malheureusement indispensable de posséder une souris à molette pour profiter à 100% de la caméra (le trackpad d'une ordinateur portable peut lui aussi permettre de profiter de la liberté de la caméra).

La vue à la première personne permet au joueur de voir exactement ce que son personnage voit. La passage d'une vue à l'autre se fait automatiquement en approchant ou éloignant la caméra du personnage.

2.1.5 Models 3D (Julien)

Nous avons décidé de doter notre jeu d'un style graphique original. Nous avons cherché l'inspiration en parcourant l'asset store. Nous y avons trouvé le modèle de notre personnage principal et nous nous sommes accordés pour un univers cartoon.

Nous avons donc commencé la réalisation de notre première île mais ce fut un échec (figure 2). Nous n'avions pas assez bien défini la forme que nous voulions lui donner et il a donc fallu recommencer le modèle car celui-ci n'était pas vraiment cartoon et n'était pas adapté à la génération aléatoire du monde.

Nous avons alors décidé de réaliser tous nos futurs modèles grâce au logiciel Blender et d'importer le moins possible depuis l'asset store. Nous nous sommes donc recentrés sur l'aspect cartoon et en avons profité pour réaliser nos premiers éléments à savoir les arbres et un rocher (figure 3).

La réalisation des modèles suivants s'est faite et continuera à se faire petit à petit en fonction des demandes engendrées par l'avancement du projet.

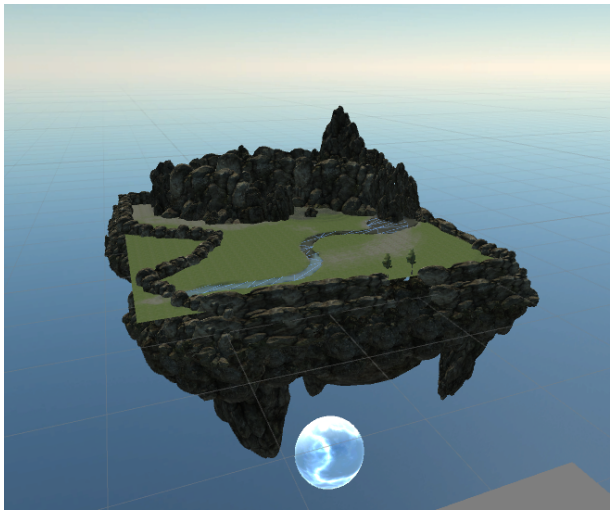

FIGURE 2 – Première île

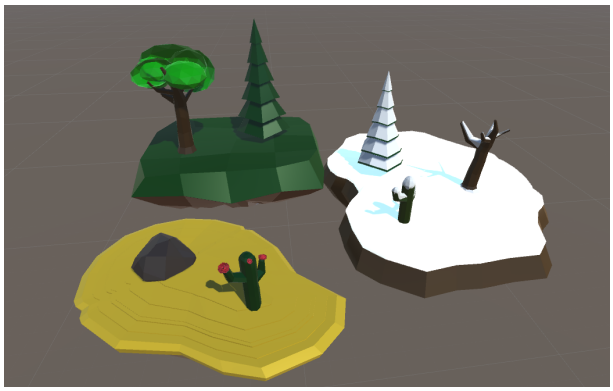

FIGURE 3 – Premiers éléments

2.2 Janvier

2.2.1 Inventaire (Romain et Florian)

Pour que notre personnage puisse posséder un grand nombre d'objets de toutes sortes sans avoir à les porter sur lui et donc se déplacer assez facilement nous avons créé un inventaire (figure 4). Celui-ci est constitué de 24 cases sous forme de 4 lignes et 6 colonnes. Chaque case peut contenir un élément, qui peut être déplacé grâce à un *drag and drop*. De plus le joueur peut avoir une description de chacun des éléments en passant sa souris sur celui-ci. Cet inventaire s'ouvre lorsque l'on appui sur la touche *inventaire* (touche *i* par défaut) et peut ensuite être fermé avec la même touche *inventaire* ou la touche *escape*.

Lorsque l'inventaire est ouvert le joueur et la camera ne peuvent plus bouger via les

FIGURE 4 – Format de l'inventaire

touches de déplacement ou la souris (mais le joueur continuera de tomber si celui-ci était en train de chuter). L'inventaire se sauvegarde automatiquement en local lorsque le jeu est quitté normalement ou lorsque qu'une modification de son contenu s'effectue (quand un objet est jeté ou récupéré). Ce même inventaire est alors récupéré d'une partie sur l'autre.

2.2.2 Barre de selection (Florian)

Lorsque nous sommes en train de bouger il est impossible de voir ce que contient notre inventaire et il faut nécessairement l'ouvrir, et donc s'arrêter, pour interagir avec lui. Actuellement ce n'est pas un problème mais quand il sera nécessaire d'utiliser des ressources pour survivre et interagir avec l'environnement, nous voulons que ces dernières soient facilement accessibles sans devoir s'arrêter (pour pouvoir boire une potion de santé alors qu'un slime nous poursuit par exemple).

Nous avons donc ajouté une barre de sélection (figure 5) synchronisée avec la 4^e ligne de notre inventaire. Le joueur peut sélectionner une case de cette barre en utilisant les touches alphanumériques correspondantes et utiliser l'objet s'y trouvant. Cette barre apparaît même quand le joueur n'a pas affiché son inventaire puisque son but est justement d'utiliser des objets hors de l'inventaire.

2.2.3 Menus (Romain et Florian)

Un élément essentiel à tout jeu est indéniablement de pouvoir le quitter et le commencer. Cela paraît évident mais il a tout de même fallu créer des interfaces permettant ces actions au début du jeu (pour commencer une partie, figure 6) et au milieu du jeu (pour arrêter de

FIGURE 5 – barre d'outil

jouer, figure 7).

Pour cela nous avons créé des menus permettant non seulement de démarrer une partie ou de quitter le jeu mais aussi de modifier de nombreuses options.

Actuellement le menu permet de modifier la langue et le volume sonore du jeu. En cours de jeu, ces menus s'ouvrent avec la touche *escape* quand aucune autre interface n'est ouverte et se ferment avec la même touche ou avec le bouton *continuer*. Lorsque le menu est ouvert, le joueur et la camera du joueur ne peuvent pas bouger comme lorsque l'inventaire est ouvert. Lors du lancement du jeu on arrive directement à un menu mais celui-ci ne se ferme pas avec la touche *escape*, par contre il est possible d'utiliser le raccourci pour accéder aux options.

L'apparence graphique des menus à longtemps était discutée. Un rectangle simple coloré comme l'inventaire ne plaisait pas donc nous avons essayé de mettre comme fond un parchemin. Cependant le placement des écritures sur ce parchemin ne paraissait pas centré et celui-ci pouvait prendre des formes bizarres lorsque la dimension de la fenêtre était changée. Nous avons alors décidé de ne pas mettre de fond au menu, ce qui est finalement très acceptable esthétiquement.

FIGURE 6 – Menu d'accueil

FIGURE 7 – Menu dans le jeu

2.2.4 Multi-joueurs (Florian)

Le mode multi-joueurs est une part importante du projet car celle-ci permettra la coopération des joueurs pour progresser dans le jeu. Nous avons dans un premier temps utilisé les fonctionnalités de *Unet* pour pouvoir créer un serveur (ou un host dans notre cas), mais aussi pouvoir rejoindre un serveur grâce a son adresse IP.

Nous avons dans un second temps synchronisé la position, la rotation et les actions des joueurs. Pour cela on a d'abord utilisé le composant d'unity *NetworkTransform*, mais celui-ci ne permettait pas la synchronisation fluide des personnages. Du coup nous avons codé nous-même un script permettant la synchronisation des personnages. Son principe est d'envoyer au serveur la position et la rotation du personnage dès que le joueur les modifie, puis celui-ci envoie ces mêmes informations à tous les autres clients. De plus pour rendre les déplacements le plus fluide possible nous utilisons le principe d'interpolation. Cela consiste à partir de l'ancienne et de la nouvelle position d'un joueur d'en déduire des positions intermédiaires.

Pour finir nous avons synchronisé les éléments, les sons et les caractéristiques du monde, que ce soient les îles, les biomes, les bruits de pas des joueurs, l'état de la journée (jour/nuit) ou bien les arbres.

2.2.5 Son (Théo et Romain)

Afin de ne pas avoir un jeu fade, nous avons introduit quelques sons. Ces sons nous permettent d'avoir une base que l'on pourra enrichir au fur et à mesure de l'avancée du développement de notre jeu. Nous avons tout d'abord ajouté un son pour l'ouverture et la fermeture de l'inventaire et menu. Par la suite nous avons ajouté des bruits de pas afin

d'entendre notre personnage marcher ou courir. Par souci de réalisme nous avons synchronisé ces sons afin qu'ils soient entendus par tous les joueurs.

2.3 Février

2.3.1 Génération aléatoire du monde (Florian et Julien)

Comme nous l'avions spécifié dans notre cahier des charges, le monde du jeu est généré de manière aléatoire. Pour cette première soutenance, nous avons implémenté la génération aléatoire d'un *chunk*, et la liaison des *chunks* entre eux, pour pouvoir générer un monde différent à chaque début de partie.

Pour le moment, le monde contient quatre *chunks* reliés par des ponts, chacun de ces *chunks* est choisi aléatoirement parmi des *chunks* préfabriqué composé d'îles, de ponts et des ancres. Par la suite un biome sélectionné aléatoirement est appliqué sur le *chunk* ce qui influencera le type d'éléments qui remplacerons les ancres. Pour finir chaque ancre est donc remplacé par un élément cohérent avec le biome. Si le biome est *désert* par exemple alors une ancre peut être remplacé aléatoirement par un cactus, un rocher, du sable... ou rien (c'est à dire un élément vide). Cette méthode de génération implique des milliers de possibilité pour un mondes composé seulement de quatre *chunk*.

Nous avons donc tous les éléments pour créer des *chunks*, il ne restera plus qu'a faire apparaître le monde de manière procédurale avec un agencement des *chunks* correct. Puis il sera aussi nécessaire de sauvegarder le monde avec les modifications que les joueurs y ont apportées.

2.3.2 Drop d'Item (Florain et Julien)

Le drop d'un item correspond à son passage de l'inventaire au monde 3D. Pour cela, le joueur à la possibilité depuis son inventaire de glisser un tas d'objets hors de l'inventaire. Il peut aussi grâce à une touche (la touche « Q »par défaut), lancer l'objet sélectionné dans sa barre d'outil. Ces actions vont faire apparaître l'objet dans le monde et le lancer à quelques mètres du personnage. Dans cet état, n'importe quel joueur peut alors ramasser l'objet dès qu'il est assez proche de lui. La récupération de l'objet ce fait par des échanges d'information entre serveur et client pour garantir que seul un joueur ramasse l'objet, et qu'il ait suffisamment de place dans son inventaire pour le stocker. De plus si deux objets identiques sont assez proches alors ils vont se rassembler. Cela permet d'afficher moins d'objets et donc d'améliorer les performances du jeu. D'autre part les objets « hors inventaire » depuis plus d'une minute disparaissent, cela permet encore une fois d'améliorer

les performances et de ne pas avoir une île envahie de ressources non utilisées.

2.3.3 Bibliothèque (Florian, Romain et Théo)

Attention, bibliothèque est un terme qui a une signification très particulière en programmation, ça ne correspond par à ce que vous faites, il faut changer le titre de la section

Lors de la création du jeu, il nous est arrivé plusieurs fois de vouloir créer une grande liste d'éléments contenant l'ensemble des *Item* existants, l'ensemble des textes avec leur traduction en anglais ou l'ensemble des éléments pouvant être générés dans le monde. Nous avons donc créé des énumérations contenant tous ces éléments et permettant d'accéder à chacun d'eux facilement. Le choix des textes affichés dans le jeu peut alors être dynamique et se faire en fonction de la langue. Ainsi tous les textes présents dans le jeu changent lors du changement de langue à l'exception des messages informatifs du chat.

2.3.4 Histoire (Romain et Théo)

En plus du fait que le contexte de notre monde soit fourni, le jeu lui-même va tourner autour d'une histoire. Cependant, l'histoire n'est pas encore définie et elle n'a donc pas encore pu être implémentée dans le jeu. Il existe néanmoins un script (au sens théâtrale du terme) racontant le début du jeu et sa fin. Ce script contient entre autre un didacticiel devant permettre au joueur de comprendre les bases du jeu.

Il est important de noter que les éléments se trouvant dans la partie Univers du jeu de ce rapport ne seront pas tous directement accessibles aux joueurs lors de l'aventure. Certaines parties devront être découvertes durant l'aventure, notamment tout ce qui concerne le fonctionnement du monde d'Aegina et l'influence négative des cristaux.

2.4 Mars

2.4.1 Interaction avec l'environnement (Florian et Julien)

Il a été décidé que le joueur ne pourrait interagir avec son environnement que d'une manière très particulière. En effet, la plupart du temps, le joueur utilise un curseur pour sélectionner l'objet avec lequel il veut interagir. Mais ce mécanisme implique un majeur inconvénient : la souris doit être réservée au déplacement du curseur et ce n'est pas envisageable dans notre cas car la souris est utilisée pour le déplacement de la caméra.

Il a donc fallu trouver une autre méthode pour sélectionner les objets et bien faire comprendre au joueur quel est l'objet actuellement sélectionné. Après concertation, nous avons remarqué que l'objet avec lequel le joueur veut interagir est souvent l'objet le plus

proche de lui (*nearElement*). Nous avons donc décidé qu'un joueur ne pourrait interagir qu'avec le *nearElement* et ce en utilisant le clic droit. L'avantage de n'interagir qu'avec le *nearElement* est que celui-ci est unique. Il ne peut y avoir qu'un seul objet le plus proche et si plusieurs object sont a égale distance, alors le premier rencontré sera considéré comme *nearElement*.

Maintenant que la méthode théorique est en place, il faut l'appliquer. Pour ce faire, il suffit de comparer les distances séparant le personnage des objets environnants. La sélection du *nearElement* se résume donc à une recherche de minimum. Mais il faut définir l'ensemble de recherche et, pour cela, être capable de différencier les objets de l'île par exemple. La solution que nous utilisons est la suivante : nous ajoutons un *tag* particulier à tous les objects avec lesquels le joueur peut interagir. Pour ne pas avoir à considérer l'ensemble des objets de monde, nous définissons une sphère d'interaction, le joueur ne peut interagir qu'avec les objets se trouvant à l'intérieur de cette sphère, les autres sont trop éloignés. L'objet le plus proche du joueur se trouve donc obligatoirement à l'intérieur de cette sphère.

Le *nearElement* est maintenant identifié mais il est impossible pour le joueur de faire la différence entre le *nearElement* et les autres éléments. Pour mettre en valeur le *nearElement*, nous utilisons un *shader* particulier. Les *shaders* indiquent au système de rendu comment un objet doit être affiché à l'écran. Le *shader* que nous utilisons ajoute un contour rouge à la silhouette de l'objet et assombrit les couleurs de l'objet. Il devient donc très facile de distinguer le *nearElement* des autres objets.

2.4.2 Cristal (Romain et Julien)

Une particularité du monde d'Aegina est la présence, sur certaines de ses îles, d'un cristal. Ce cristal n'est pas juste comme les autres éléments du décors tels que les arbres ou les pierres. Ces derniers sont là pour être récoltés et transformés par les joueurs, les cristaux eux sont indestructibles. Dans la nature les cristaux ont une couleur grise indiquant qu'il n'ont pas été activés par un joueur, ils n'ont alors par d'influence visible sur l'environnement. Lorsqu'un joueur tente d'interagir avec un cristal inactif, une fenêtre (figure 8) apparait proposant au joueur d'activer le cristal en échange d'un coût en ressources. Les informations contenues dans cette fenêtre ne sont pas encore toutes définies, mais elle a été dimensionnée pour pouvoir contenir les prochains ajouts prévu.

Une fois activé, le cristal change de couleur pour prendre celle de l'équipe du joueur (bleu lorsque tout les joueurs sont dans la même équipe). A l'activation, un cristal se spécialise dans l'une des trois catégories suivantes de manière aléatoire :

— **Attaque (60%) :** catégorie des cristaux d'attaque (ou de défense ce qui revient au

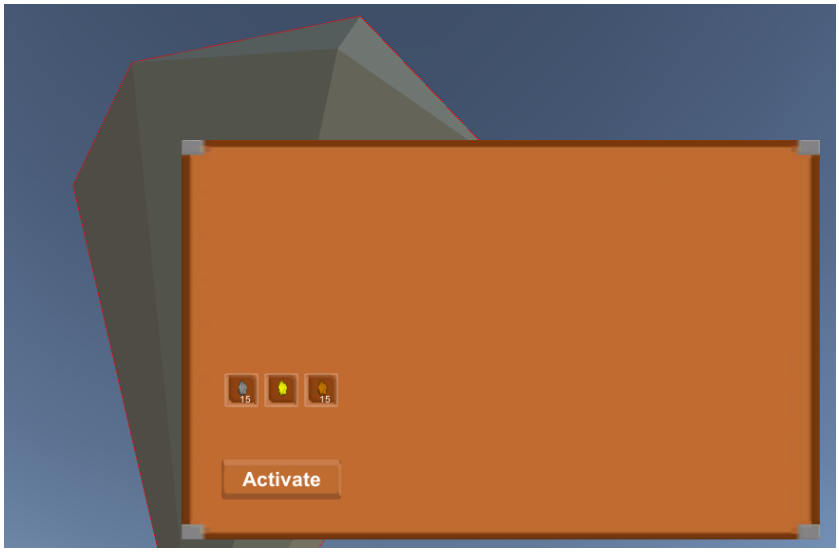

FIGURE 8 – Interface du cristal

même).

— **Récolte (30%) :** catégorie des cristaux de production
— **Portail (10%) :** catégorie des cristaux de portail.

Actuellement, il est impossible de connaître la spécialisation d'un cristal dans le jeu, mais cela n'a que peu d'importance car pour l'instant les cristaux activés ne sont différents des cristaux inactif que par leur couleur et n'ont aucune influence sur le jeu.

2.4.3 Chat (Florian)

Comme déjà précisé, la coopération est un élément important du jeu. Pour cela la communication est primordiale. Nous avons donc ajouté un chat qui permet aux joueurs de s'envoyer des messages sans passer par un logiciel tiers. Le chat ne permet pas seulement de communiquer, il permet aussi de rentrer certaines commandes (communément appelé *cheatcode*), en voici l'énumération :

/help : énumère les commandes avec leurs arguments ;

/time <value (int)> : détermine l'heure de la journée ;

/give <id (int)> <quantity (int)> : ajoute à l'inventaire du joueur la quantité indiquée de l'objet dont l'identifiant correspond au paramètre ;

/nick <name (string)> : change le nom du personnage ;

/msg <player (string)> <message (string)> : envoie un message privé au joueur dont le nom du personnage est passé en paramètre ;

Certaines de ces commandes sont surchargées, c'est a dire que nous pouvons mettre plusieurs types d'informations en argument, par exemple la commande */time day* est acceptée et va mettre le soleil au zénith. D'autres commandes ont des alias par exemple la commande */msg* peut aussi être écrit */m*.

2.4.4 Musique (Théo)

Afin de compléter l'univers que nous essayons de créer à travers notre jeu, nous avons ajouté des musiques de fond. Nous avons d'abord ajouté une musique qui est associée au menu principal uniquement. Trois autres musiques ont par la suite étaient ajoutées, chacune d'elles est associée à un thème différent, la forêt, le désert ou la neige, qui correspond chacun à un des biomes possibles pour les îles. Pour le moment le choix de la musique écoutée se fait aléatoirement, il devra ensuite se faire en fonction de l'île sur laquelle se trouve le personnage. En multi-joueurs chaque personne peut se retrouver avec des musiques différentes au même moment.

2.4.5 Survie (Théo)

La survie étant le thème de notre jeu, il nous fallait construire de bonnes bases pour le développement de cette dernière. Aujourd'hui les éléments clés de notre survie sont présents. Parmi ces éléments, trois sont visibles directement en jeu grâce à des barres (figure 9). Deux de ces barres, l'une symbolisant la soif et l'autre la faim du personnage, se situent en haut à droite de l'écran tandis que la troisième barre, symbolisant la santé (les points de vie), se trouve au dessus de la barre de sélection.

FIGURE 9 – Disposition des différentes barres

Lorsque le barre de faim ou de soif est vide, les point de vie de votre personnage commencent à descendre. Cependant ces deux attributs (soif et faim) ne sont pas les seuls à pouvoir s'attaquer à vos points de vie, si vous décidez de sauter dans le vide, votre santé en souffrira également car malgré sa force et sa volonté, Ille ne sait pas voler.

Quand vous vous retrouvez sans point de vie (barre de santé vide) vous sentirez vos forces disparaître, c'est ce que nous appelons communément la mort. Cependant, dans notre grande bonté, nous avons décidé que la mort ne serait pas tout de suite définitive et donc chaque mort sera accompagnée d'un nouveau début. Pour le moment, le joueur *respawn*, il est téléporté sur la première île qu'il a traversée et ses attributs (santé, soif et faim) sont remis au maximum.

2.4.6 Alpha (Florian)

La dernière étape avant la soutenance était de partager notre première version, *v1.0-alpha*. Ce partage nous a permis d'avoir des retours de bugs et quelques avis. Les bugs importants rapportés étaient :

- Le respawn du joueur qui parfois ne fonctionnait pas ;
- Un bug graphique au niveau de l'élément proche lorsque la camera était à l'intérieur ;
- La musique qui en multi-joueurs pouvait se superposer avec la musique des autres joueurs alentour.

Nous avons corrigé la totalité des bugs rapportés pour produire la version *v1.1-alpha* puis grâce aux avis, nous avons ajouté et modifié quelques détails artistiques. Ces modifications ont abouti sur une version *v.1.2-alpha* qui est la version présentée lors de la première soutenance.

3 Prévisions

Dans cette partie, nous allons vous présenter les différents concepts que nous implémenterons pour la soutenance n° 2 prévue entre le 2 et le 6 mai 2016. Ces ajouts seront concentrés sur le *gameplay* car jusqu'a présent il y a eu de nombreux éléments ajoutés au jeu mais aucun ne permet réellement de jouer et de s'amuser.

3.1 Les quatre grands axes

Les quatre grands axes sont les sont quatre ajouts qui permettrons d'ajouter de nombreux éléments de *gameplay* à Aegina.

3.1.1 Création d'objet

L'ajout de la création d'objet permettra au joueur d'accéder à une interface de création dans son inventaire lui offrant la possibilité de transformer des objets qu'il possède actuellement en d'autres objets. La liste des objets qu'il pourra créer dépendra de l'avancement du jeu mais aussi de la présence de certains ateliers dans les environs (comme une forge par exemple). Cette liste dépendra aussi d'un *livre des recettes*. Une fois que le joueur aura créé un nouvel objet près d'un atelier spécifique, la recette de fabrication sera ajoutée à son livre et il pourra ensuite la réaliser n'importe où dans le monde (ce système est inspiré de *Terraria* et *Don't starve*).

3.1.2 Ajout des créatures

L'ajout des créatures sera une part importante de la prochaine version. Elles devront pouvoir se déplacer de manière intelligente. Certaines seront passives mais d'autres seront agressives et devront attaquer les joueurs. Ces derniers pourront alors se battre contre ces créatures et récupérer des objets pour progresser dans leur aventure. Les créatures ne s'attaquent pas entre elles. L'ensemble des créatures possèderont une IA gérant leurs déplacements et leur comportement. Lorsqu'aucun joueur ne se trouve dans les environs, cette IA devra permettre de petits déplacements sur l'île sans rentrer dans les éléments du décor ou tomber dans le vide. Cette IA sera commune à l'ensemble des créatures qu'elles soient passives ou agressives. La gestion du comportement des créatures sera différente lorsqu'un joueur se trouve dans les environs, elle dépendra de l'agressivité ou nou de la créature (les créatures agressives attaquerons le joueur alors que les passives fuirons par exemple).

3.1.3 Ajout des succès

Les succès permettront aux joueurs de suivre l'avancement de l'histoire ainsi que de se remémorer tous les faits héroïques qu'ils ont réalisés. Un joueur pourra consulter ses succès en appuyant sur une touche (à définir) ce qui ouvrira une interface.

Dans cette fenêtre, les succès serrons hiérarchisés sous la forme d'un arbre. C'est à dire qu'il faudra absolument que le joueur ait réussi un dit succès pour espérer remporter les succès suivants. Cette structure nous permettra de définir certains événements de l'histoire comme étant des succès et ainsi la bloquer tant que le joueur n'aura pas effectué une opération clé de l'histoire. Le déclenchement des dialogues de l'histoire seront donc déclenchés par la réussite des succès.

3.1.4 PVP et Conquête

Cet ajout comporte l'amélioration des cristaux pour que ceux-ci produisent un véritable effet sur l'environnement. Nous souhaitons aussi ajouter un second mode multi-joueurs où ceux-ci ne joueront pas en coopération mais les uns contre les autres dans le but de prendre le contrôle des cristaux adverses. Ce second mode se passera dans un monde aux dimensions limitées avec une plus grande présence de cristaux et dans lequel il n'y aura pas d'histoire aussi développée.

L'amélioration des cristaux dans le mode de jeu en coopération permettra de placer les cristaux au centre du *gameplay* car ceux-ci deviendront essentiels à la survie du joueur. En effet les fonctionnalités de *respawn* dépendront des cristaux.

3.2 Les améliorations

Cette partie contiendra des améliorations des éléments déjà existant ne créant pas forcement du *gameplay*.

3.2.1 Survie

La première amélioration sera de donner la possibilité au joueur de récolter des ressources grâce au principe d'interaction présenté précédemment. Parmi ces ressources, certaines seront consommables. Ces consommables permettront au joueur de boire et de manger (plus besoin d'attendre la mort pour retrouver une barre de soif pleine).

Enfin, de nouveaux paramètres tels que la résistance et les dégâts seront à prendre en compte suite a l'implémentation des créatures hostiles et du PVP.

3.2.2 Génération aléatoire et sauvegarde

Les améliorations de la génération aléatoire consisteront à un ajout de chunks et d'éléments, mais aussi la création d'un système de *seed* qui permettra entre autre une sauvegarde du monde simple et performante. Un *seed* est un simple nombre qui est lié au monde, cela implique que deux mondes ayant le même *seed* seront identiques.

3.2.3 Histoire

L'amélioration de l'historie se fera en deux partie : l'écriture des partie non encore définies de l'histoire et l'implémentation d'une partie de l'histoire dans le jeu. Une patrie de l'histoire sera réalisée sous forme de didacticiel, ce qui permettra à tout nouveau joueur de découvrir le monde et les différentes interfaces mises à sa disposition.

3.2.4 Environnement sonore

L'ajout de sons/musiques nous permettra de dynamiser notre jeu. L'ajout des créatures amènera à l'ajout des sont qui leur sont associés. Ces bruits rendront les créature plus impressionnantes ou au contraire attirantes, attention à ne pas se fier aux apparences qui peuvent être trompeuses...

L'ajout d'une musique qui sera consacrée à la nuit permettra de marquer une différence, autre que visuelle, entre le jour et la nuit dans le monde de Aegina.

Conclusion

Annexe

Script de l'histoire

Légende

- structure
- **monologue du narrateur**
- *indications scéniques*

Histoire

L'histoire sera racontée par un narrateur. Nom actuel du personnage : Ille (i majuscule deux l minuscule et un e minuscule)
La narration sera découpée en deux parties :

- Les cinématiques, les points importants de l'histoire, qui permettront au jeuxd'avancer (cinématique sous forme de dessin qui se suivent ou de jeux de caméra dans le monde 3D)
- Les « monologues » du narrateur qui auront lieu pendant que le personnage bouge et n'interrompront pas les actions du personnage. Ces monologues auront comme but de préciser le but du jeu ou de faire des remarque anodines sur l'environnement.

Début :
but concret : premier pas (tutoriel)
Personnage bloqué
En ouvrant les yeux Ille sentit l'air et les doux rayons du soleil caresser son visage. Ille avait l'impression d'avoir dormi très longtemps à tel point que Ille doutait de savoir encore se mouvoir. Ille décida alors de se lever et de faire un peu d'exercice pour faire passer cette impression.
Personnage débloqué
Sous but :
Ille commence par se déplacer un petit peu pour se réhabituer à ses jambes (touche droite gauche haut et bas)

- se déplacer

S'étant habitué à ses jambes, iI décide de se mettre à courir (touche shift enfoncée)

- courir

Ille est tellement en forme qu'il pense même pouvoir se permettre de sauter sans problème (touche espace)

— sauter

Juste après avoir sauté, il entend un sinistre grognement.

petite pause dramatique

Ille a faim. Cependant dans cette contrée sauvage, Il devine qu'il doit chasser sa nourriture. Mais pour chasser Il a besoin d'une arme. Justement Il remarque un bâton sur le chemin, L'arme parfaite pour lui (ramasser les objets en avançant dessus)

— récupérer un item au sol (un bâton par exemple)

Après avoir mis le bâton dans sa poche, il se dit qu'il devrait peut être le sortir de celle-ci pour que son arme soit utilisable (ouvrez l'inventaire avec I et faites glisser le bâton dans la 4^{eme} ligne correspondant aux équipements rapides)

— ouvrir l'inventaire

— mettre le bâton dans la barre de raccourcis

Il se met à la recherche de nourriture et il en se retournant tombe nez à nez avec un lapin. Ni une ni deux, il attaque son petit déjeuner (attaquer en ...)

— utiliser le bâton pour tuer un lapin

Cependant, il avait un vague souvenir que la viande crue n'était pas très digeste au petit déjeuner. Faute de feu Il décida quand même de manger sa viande préparée en brochette. ouvrez l'inventaire puis créez une brochette)

— créer de la nourriture à partir du bâton et du lapin tué (brochette)

Finalement le repas de Ille est près. Il ne lui suffit plus que de consommer sa brochette et le grondement de son ventre cessera (utiliser un consommable avec clic droit)

— utiliser un consommable (ici la nourriture)

but concret : activer le premier cristal

Personnage bloqué

Maintenant que Ille a trouvé de quoi survivre, il peutt enfin se reposer. Jusqu'à présent Ille n'a pensé qu'à manger mais une fois le ventre rempli il se mit à penser et à se questionner « Mais où suis-je ? Quel est cet endroit ? » En effet cet archipel d'îles volantes ne lui est pas familier « Attend ! ! ! Des îles volantes ? ? Mais ce n'est pas possible, ça ne peut pas exister » Et pourtant ce qui s'étend à perte de vue n'est pas une illusion. Ille ne se trouve plus chez lui, peut être même plus dans son monde. *petite pause*

Quand Ille a assimilé ce qui lui arrive, il prend une décision « Il faut absolument que je comprenne où je suis et pour ça il faut que j'explore ces drôles d'îles » Et c'est ainsi que Ille commence son aventure.

Personnage débloqué

sous but :

Pour commencer son exploration ,la première chose que Ille fait est de traverser un pont séparant deux îles.

— emprunter un pont

Il ne manque pas de se demander ce qu'un pont peut bien faire dans cet archipel ayant pourtant l'air abandonné et espère que celui-ci ne va pas s'écrouler sous son poids...heureusement cela n'arrive pas. en traversant ce pont ,Ille voit à l'horizon une sorte de reflet et intrigué décide de chercher sa source.

— trouver un cristal

Le reflet vient d'un gigantesque cristal taillé comme un prisme, mais le plus impressionnant est que celui-ci lévite ...Malgré sa stupeur, Ille se ressaisit assez vite « hum pas si étonnant que ça vu que je me trouve actuellement sur une île volant dans le ciel ». Après s'être calmé et posé, il voit que le cristal réagit à sa présence et plus précisément au métal de sa boucle de ceinture qui semble attirée vers le cristal. Ille décide donc d'aller chercher du métal pour tester ces étranges réactions et s'éloigne rapidement ne voulant pas perdre sa précieuse ceinture. En s'éloignant Il se dit que pour récupérer du métal cela va être plus dur que de ramasser du bois ou de tuer des lapins. Il doit donc fabriquer quelque chose pour récupérer et raffiner le métal.

— fabriquer une pioche

— créer une forge

Maintenant qu'il possède une minuscule forge Il peut enfin produire du métal pour le cristal.

— récolter les ressources pour activer le cristal

— activer le cristal

Cinématique !!

première/mauvaise fin :

Ille a finalement réussi. Après tout ses efforts il a recréé un portail lui permettant de revenir chez lui. C'est en hâte qu'il efface ses derniers doutes et plonge dans le portail. Il ne sait toujours pas qui il est, mais cela n'a plus d'importance. Il pourra facilement répondre à ces questions un fois chez lui. Cependant, il

a oublié un dernier détail. S'il se trouve ici ce n'est sûrement pas par hasard. Quelque chose ou quelqu'un a dû l'y envoyer et peut-être que rien n'empêchera cette chose de le renvoyer ici...ou pire.

Cri d'agonie au loin

seconde/bonne fin : à écrire