ECOQUEST

RAPPORT FINAL

GOUDAL VICTOR, DIGUAT MARIUS, ZOUNGRANA SOULEYMANE, MAURER GILLES

TABLE DES MATIERES

INTRODUCTION

DESCRIPTION DU PROJET

Ce projet est en collaboration avec une association de surveillance de la qualité de l'air (ATMO Grand EST) et intervient dans le cadre du projet européen Life V-Air. Le but est de développer un jeu sérieux de type « God based RPG » pour sensibiliser à la qualité de l'air.

L'écologie est un sujet majeur de notre époque et beaucoup ne sont pas assez sensibiliser à ces questions qui auront tant d'impact sur le monde de demain. Notre objectif est donc de viser en particulier les décideurs politiques, comme les maires ou les conseillers municipaux. Ceux qui ont le pouvoir de changer les choses mais à qui il peut manquer cette sensibilité sur le sujet. Le jeu leur permettra donc de prendre du recul durant un petit moment pour observer une ville d'un œil extérieur. Cela leur permettra de se rendre compte des différents aspects que prends le combat pour l'écologie, voir l'impact que peuvent avoir certaines décisions et d'être sensibiliser à l'importance de la bonne qualité de l'air.

Le public cible est donc composé de décideurs politiques. Ces gens n'ont pas beaucoup d'expérience avec la réalité virtuelle. L'objectif est donc de concevoir un jeu qui sera marquant pour eux mais surtout qui soit simple à prendre en main. Le joueur devra se sentir à l'aise et ne pas perdre ses repères afin que l'expérience le marque positivement.

Le jeu prendra la forme d'un monde composée de cinq secteurs : les résidences, l'agriculture, les industries et la production d'énergie. Cette ville sera autonome et le joueur pourra l'observer évoluer petit à petit. Régulièrement, le joueur aura le choix entre différentes cartes qu'il pourra jouer dans un second temps. Ces cartes, permettront d'agir sur les différents éléments de la ville de plusieurs manières. Ces actions auront ensuite un impact sur certains indicateurs comme la pollution de l'air, le bonheur de la population ou l'énergie disponible dans la ville.

OUTILS

Pour ce projet, nous avons utilisés plusieurs logiciels pour différents aspects. Voici donc un résumé des logiciels principaux que nous avons utilisé et des raisons qui justifies ces choix.

- Modélisation : Pour la modélisation, nous avons intégralement utilisé le logiciel *Blender*. Ce logiciel est en effet gratuit et ses multiples fonctionnalités nous ont permis de modéliser librement tout ce que nous avions en tête. De plus, Souleymane avait déjà beaucoup d'expérience sur *Blender*.
- Développement du jeu : Le développement du jeu a été fait sur *Unity*. Nous avons choisi ce logiciel plutôt que *Unreal Engine* car il nous semblait plus simple à prendre en main. De plus, il semble que *Unreal Engine* demande plus de ressources ce qui nous empêchait de lancer le projet depuis nos ordinateurs portables.
- Autres logiciels : Nous avons également utilisé *Canva* pour réaliser les slides du tutoriel et *Inkscape* pour tout ce qui concerne la réalisation des logos et autres visuel 2D.

BUDGET

Afin de correspondre vraiment à un projet professionnel, nous avons réalisé établis le budget du projet avant de nous plonger dans son implémentation. Pour étudier le budget, nous avons considérés que nous sommes une entreprise indépendante, ce qui nous permet d'estimer les dépenses que nous aurions dû faire notamment sans le prêt de matériel de l'UTBM. Mais aussi de réfléchir au budget qui aurait pu être débloqué pour nous si cela avait été nécessaire.

Nous étudierions aussi les estimations de gains si le projet n'était pas pour un organisme à but non lucratif.

DEPENSES

Pour assurer le bon déroulement de notre projet, nous avons établi une liste détaillée des dépenses prévues, comprenant le matériel nécessaire, les salaires théoriques des quatre membres du groupe et d'autres potentiels dépenses.

Au niveau du matériel, on compte le casque Meta Quest 3 (128 Go). A cela s'ajoute le prix des ordinateurs nécessaire pour développer le projet. Pour prendre en compte le cout sur la durée du projet, on considérera que le projet a duré 6 mois et que le prix d'un ordinateur est amorti sur 3 ans (durée légale pour les équipements informatiques). Les prix d'achats des ordinateurs sont 1500€ pour le portable de Souleymane, 800€ pour le portable de Victor, 1800€ pour le fixe et la carte graphique de Marius et 200€ pour le fixe prêté à Gilles pour le projet. Cela donne un total de 4300€ soit pour 6 mois environ 750€

En plus de ces différentes dépenses, pour être totalement réaliste par rapport à un projet de ce type en entreprise, il faudrait également compter le prix de notre lieu de travail, de l'électricité et du chauffage. Pour cela, on comptera le prix d'accès à une salle de coworking durant 6 mois. Selon les tarifs trouvés pour un tel espace à Belfort, cela reviendrait à 230€ par mois soit un total de 1380€.

Le prix des outils numériques utilisés dans le projet devrait aussi être pris en compte. Cependant, tous les logiciels que nous avons utilisés sont gratuits. Unity ne met des frais de téléchargement que si le jeu a été fait avec une version 6 du logiciel, notre jeu étant développé en version 3.7, ces frais ne s'appliquent pas au projet.

Enfin, la dernière source de dépense est la rémunération des intervenants au projet. Pour estimer ce cout, on considérera que nous avons tous les quatre travaillé 150h sur le projet car c'est le temps annoncé dans les informations de l'UV MV50 même si dans la réalité, le temps passé sur le projet est sans doute supérieur. Selon les statistiques des salaires à la sortie de l'UTBM, la rémunération annuelle moyenne est de 39 500€, ce qui si on se base sur le nombre d'heures de travail légal dans une année (1607) fait un salaire à l'heure de 24.58€. Ainsi, la dépense totale en salaire pour ce projet est de 14 748€.

Si on totalise ces dépenses, le cout total du projet est donc estimé à 1080€ de matériel plus 14 748€ de salaires ce qui fait un total de 15 828€.

GAINS

Ce projet ayant pour but de sensibiliser des politiques, il ne sera pas vendu et ne génèrera donc pas de revenus directs. Cependant, on peut s'intéresser au budget global du projet Life V-Air financé à 60% par l'Union Européenne et qui s'élève à 1.3 millions d'euros. Ce projet est décomposé en deux parties : un serious game et un escape game. EcoQuest n'étant pas prévu à l'origine par le projet d'ATMO Grand Est, le budget initial n'en prends pas compte, mais on pourrait estimer qu'au vue du potentiel de ce jeu, l'argent nécessaire à sa réalisation soient débloqué dans le budget global.

On peut aussi estimer les gains du projet en considérant qu'il a été fait par une startup qui compte le commercialiser. Dans un tel cadre, les profits pourraient venir de la commercialisation du jeu en lui-même mais également de la mise en place de formations pour les entreprises, les mairies ou d'autres organisations qui souhaiteraient sensibiliser leur personnel.

PROPRIETE INTELLECTUELLE

Si ce projet avait été réalisé par une startup qui visait la commercialisation, la propriété intellectuelle du projet serait importante. Elle serait décomposée en trois parties :

- La marque EcoQuest et son logo.
- Les modèles du jeu et ses différents éléments.
- Le code source du jeu

La propriété intellectuelle permettrait de s'assurer que le projet continue de nous appartenir même lorsque nous l'aurons commercialisé à des organisations ou à des particuliers.

RESUME DU BUDGET

Voici donc un tableau qui récapitule les dépenses liées au projet ainsi que les potentielles sources de gains. Il est cependant très difficile d'estimer numériquement les gains qui pourraient être générés par le projet.

	Dépenses	Gains
Casque Meta Quest 3	550 €	
Ordinateurs (amortis sur 3 ans)	750 €	
Locaux, électricité, chauffage	1380 €	
Salaires	14 748 €	
Unity	0 €	
Ventes, publicités		??
Subventions		??
Total	17 428 €	??

POUR ALLER PLUS LOIN

Dans le cadre de l'UV MG01 (Management de l'innovation et entrepreneuriat), Marius et son groupe ont pu réaliser un travail complémentaire sur l'analyse d'un potentiel marché et du business plan si ce projet venait à être utilisé par une startup. L'intégralité de cette analyse peut être retrouvée dans les annexes.

ORGANISATION

REPARTITION DES TACHES

Afin de faciliter notre organisation et la répartition des différentes tâches, nous avons décidé d'affecter à chaque membre du groupe un groupe de tâche en lien avec ses capacités et son domaine de prédilection. Cette répartition n'empêche pas chacun de réaliser les autres tâches qui ne font pas partie de son domaine, mais permet d'avoir un responsable pour les aspects les plus importants du projet.

Souleymane étant celui avec le plus d'expérience sur blender et sur la modélisation 3D de manière générale, c'est lui qui a été chargé de toute la partie design et modélisation.

Victor s'est occupé de toute la partie des indicateurs, des données de la ville et de l'équilibrage du jeu. Il est également intervenu sur de nombreux autres aspects du développement.

Marius ayant déjà une bonne expérience sur Unity a été chargé de la gestion du développement sur Unity. Il a veillé à ce que les différentes choses développées sur Unity soient cohérentes entre elles, que le projet soit clair et bien ordonné. Il a notamment développé la génération de la carte et le déplacement des IA.

Enfin, Gilles a été chargé de gérer la communication avec les différentes personnes externes. Il s'est aussi occupé du développement des interfaces et des interactions.

OUTILS D'ORGANISATION

Pour collaborer efficacement tous les quatre, nous avons mis en place plusieurs outils. Pour discuter, nous avons créé un serveur discord, sur lequel nous avons fait un salon pour chaque domaine qu'allait aborder notre projet. On retrouve ainsi un salon pour le design, un pour l'IHM ou encore un pour le management et l'organisation.

En plus de ce serveur discord, nous avons créé une équipe sur Notion afin que chacun puisse enregistrer de nouvelles tâches, regarder les tâches qu'il lui reste à faire ou signaler la fin d'une tâche. Ainsi, assez régulièrement, nous discutons ensemble des prochaines tâches et nous les répartissons entre nous.

Finalement, pour tout ce qui concerne la rédaction de rapport ou le partage de documents, nous avons mis en place un drive partagé. Chacun peut donc avoir accès et modifier les rapports. Ce drive est aussi utilisé pour la partie du projet reliée à MV53 où nous avons réfléchis ensemble sur d'autres aspects du projet.

DESIGN

INSPIRATION

Pour le design de ce projet, nous avons choisis de nous orienter vers le style Low-poly. Plus simple à réaliser, plus léger au niveau des vertex, ce style est très agréable à regarder et permet d'offrir une atmosphère agréable.

EXEMPLES MODELES

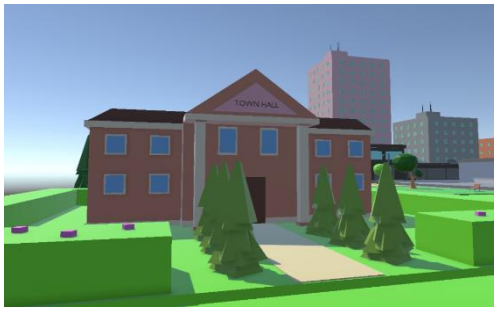
Dans le jeu, un très grand nombre de modèles peuvent être retrouvés. En voici un petit aperçu pour illustrer le style que nous avons choisi pour EcoQuest.

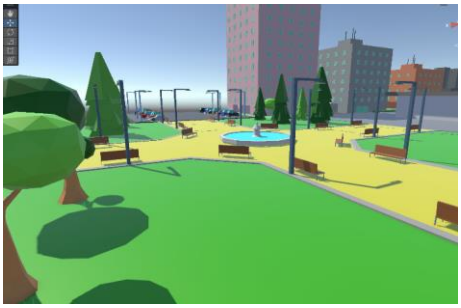
Nous avons réalisé une grande variété de modèle 3D correspondant aux différentes zones ainsi qu'aux éléments de gameplay.

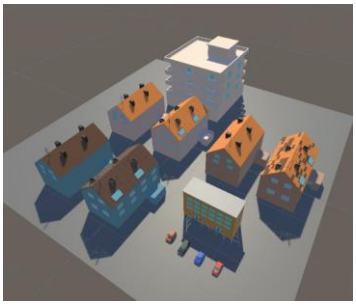
LA ZONE D'HABITATIONS

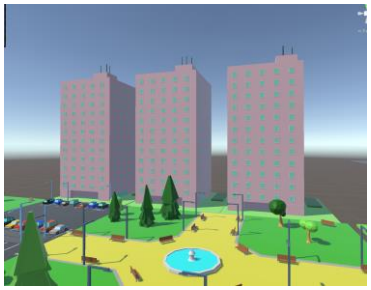
Dans la zone ville, nous avons réalisé :

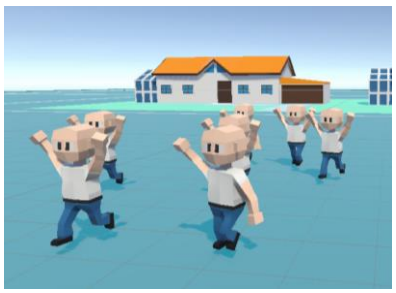
- Différentes variations de maisons dépendant du type d'énergie utilisé,
- De nombreux véhicules (Taxi, voitures personnelles, bus)
- Une mairie
- Une zone de manifestation
- Les routes pour la circulation des véhicules ainsi que des stations de bus
- Ainsi que des personnages et d'autres éléments rendant la scène plus intéressante







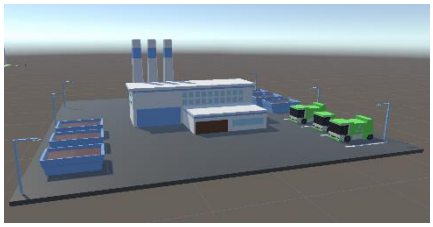


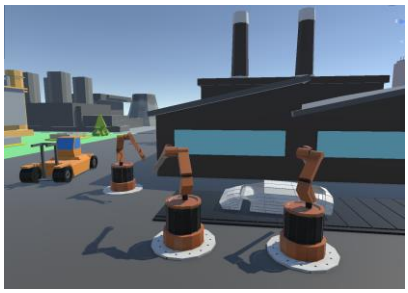


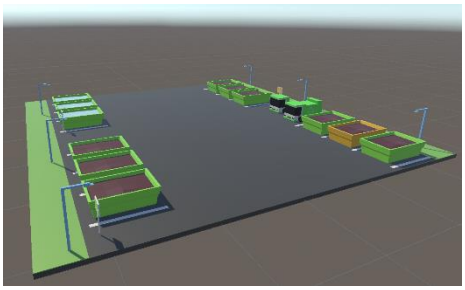
LA ZONE INDUSTRIELLE

Dans cette zone, nous avons réalisé les modèles suivants :

- Une usine de fabrication de véhicule
- Une usine super polluante
- Use usine parfaite
- Une usine normale
- Et un réseau de recyclage (collecte, tri, incinération)



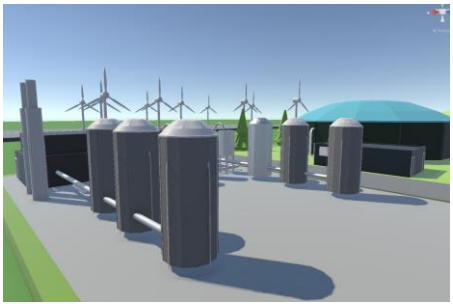




LA ZONE ENERGIE

Nous pouvons y retrouver les modèles suivants

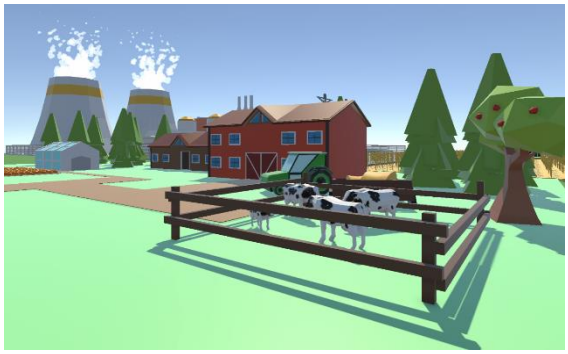
- Une centrale nucléaire
- Des éoliennes
- Une centrale biogaz
- Ainsi que des plaques solaires

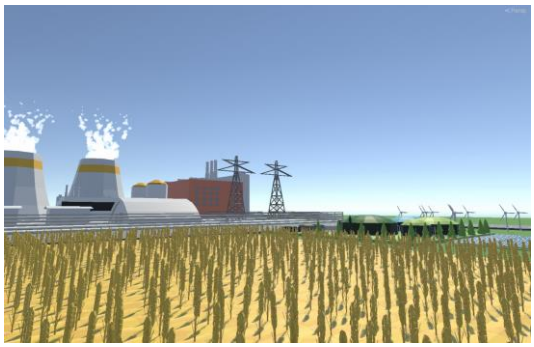


LA ZONE AGRICULTURE

Dans cette zone, nous avons réalisé les modèles suivants :

- Une ferme avec des animaux tels que des vaches
- Des tracteurs
- Des arbres fruitiers
- Différents types de champs (blé, tomates, concombres, salades, ...)





ANIMATIONS

Enfin, une partie du travail sur les modèles 3D a aussi été de les animer pour leur donner vie. Dans le jeu, on retrouve donc plusieurs modèles animés comme les vélos, les personnages où même les véhicules.

Nous avons d'abord modélisé un personnage dénommé Thibaut à partir duquel nous avons créé plusieurs variations de personnage. Pour le reste du travail, nous sommes restés sur Thibaut pour simplifier l'avancement du projet.

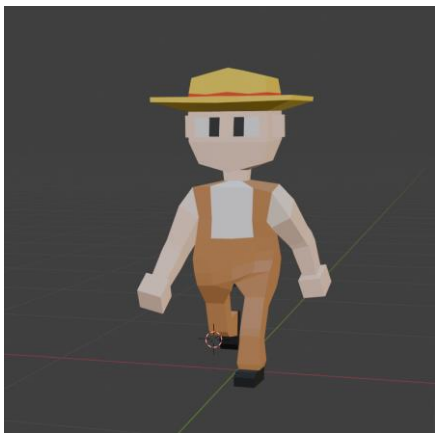
Nous avons également modélisé un assistant dénommé Finn, qui est nuage vous aidant dans les tutoriels dans la suite du jeu.

Nous avons pu réaliser plusieurs animations pour les personnages telles que :

- L'idle (une animation pour les personnages statique)
- Le bike (une animation d'un personnage sur un vélo)
- Le walk (une animation de personnage qui marche)
- Revolt (une animation exprimant le mécontentement de Thibaut)

Enfin nous avons réalisé une animation où l'on voit Finn flotter subtilement, une animation des éoliennes avec les hélices tournantes ainsi qu'une animation de voitures

Thibaut et ses variations



Une armée de Thibaut très mécontents

Finn très content d'être où il est

LA CARTE DU JEU

Lors d'une partie, une grande partie du temps de jeu consiste à observer la carte pour voir l'effet de nos actions. Il était donc essentiel que la carte soit complexe et bien détaillé afin d'offrir une belle expérience à l'utilisateur. Voici donc les différents éléments techniques autour de la carte principale d'EcoQuest.

GENERATION PROCEDURALE

Dès le début du projet, nous nous sommes lancé le défi de permettre une génération automatique et aléatoire de la carte. Cette fonctionnalité, en plus d'offrir au joueur une expérience unique à chaque parti, nous permet aussi de paramétrer facilement chaque aspect de la carte et d'ainsi pouvoir y faire des modifications à n'importe quel moment. Nous avons donc réussi à faire un algorithme capable de générer toute la carte et pouvant être paramétrer entièrement.

La génération du terrain peut être décomposer en 2 grand algorithmes :

- MapGenerator : Création du relief du terrain
- FillMapManager : Remplissage du terrain

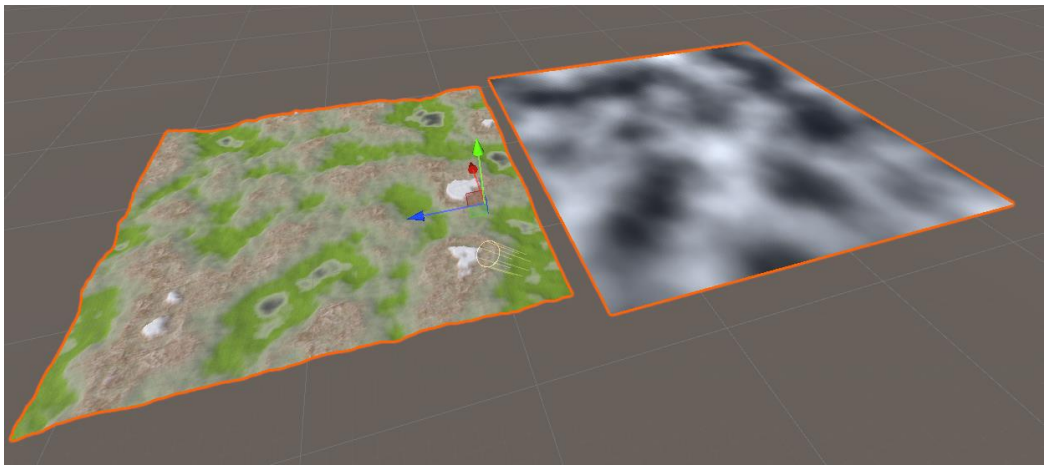
CREATION DU RELIEF

La première étape consistait à développer un algorithme permettant de générer automatiquement le relief du terrain. Pour cela, nous avons utilisé la technique courante du **Perlin Noise**. Marius, ayant déjà travaillé sur ce sujet lors d'un projet précédent, a pu démarrer avec une base solide.

Nous nous sommes appuyés sur la playlist de Sebastian Lague sur le sujet, une référence connue dans le monde du jeu vidéo, pour réaliser ce premier algorithme : Landmass Generation.

Cet algorithme nous offre un contrôle précis sur l'apparence du terrain. En effet, tout étant paramétré, nous pouvons ajuster le relief en modifiant la lacunarité, l'amplitude, la persistance ou le nombre d'octaves, qui sont les paramètres clés du Perlin Noise.

De plus, nous pouvons également ajuster d'autres options, tels que le **falloffMap** pour générer un monde en forme d'île, ou encore le **flatShading** pour obtenir un ombrage plat, adapté au style low-poly.



Nous pouvons voir sur cette image, le terrain généré à gauche et le perlin noise à l'origine du relief.

DEVELOPPEMENT DU SHADER

Nous avons également consacré du temps au développement du **shader** du terrain. Ce shader nous permet de contrôler les couleurs et les effets lumineux du terrain. Il comporte deux parties principales :

1. **Affectation de Couleur par Hauteur** : La première partie du shader attribue une couleur à chaque vertex en fonction de sa hauteur. Cela permet de simuler différentes altitudes et zones climatiques.

2. **Vérification de la Position par Rapport aux Zones** : La deuxième partie vérifie la position des vertices par rapport aux zones définies, ce qui permet de créer des effets spécifiques, comme rendre invisible la partie du terrain en dehors du périmètre de vue, facilitant ainsi la navigation. Nous reviendrons en détails sur la navigation dans la partie sur les actions de l'utilisateur.

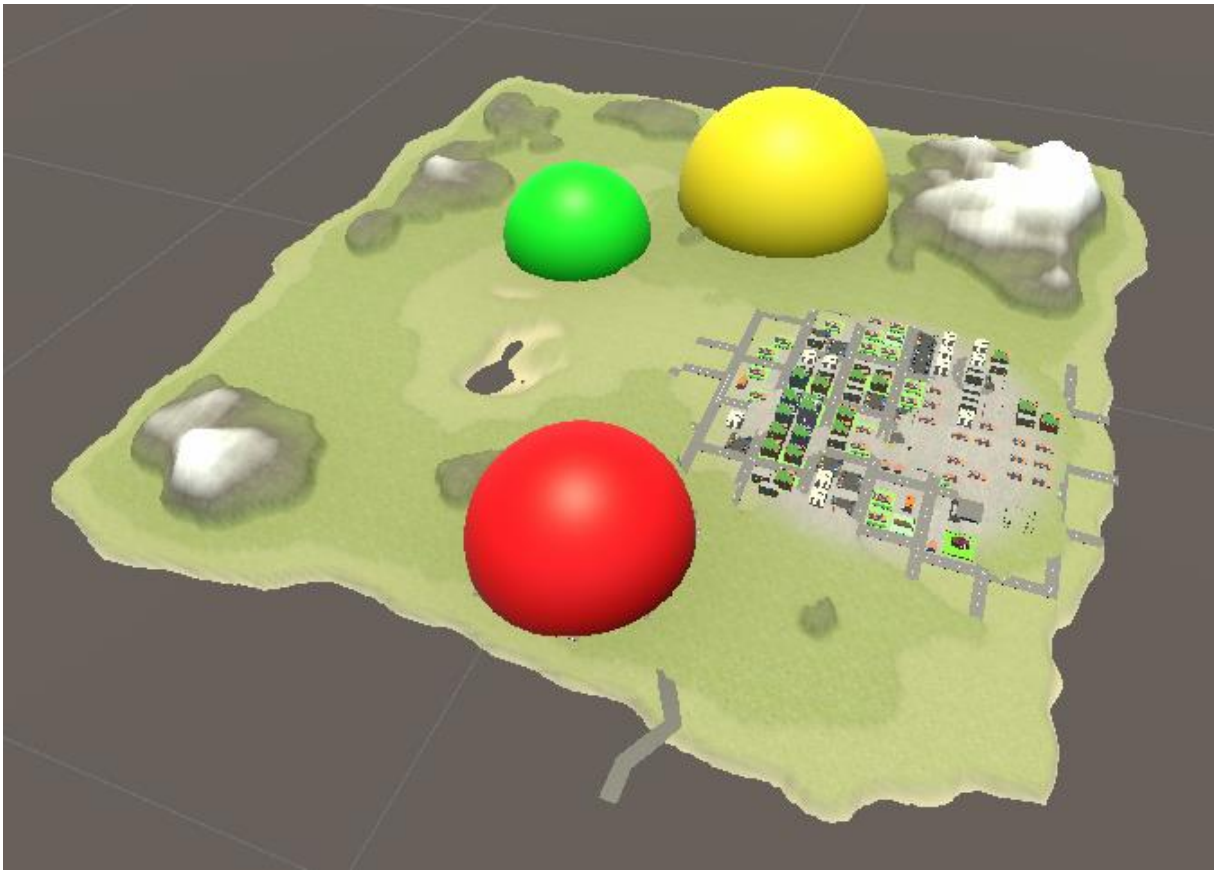
GENERATION DU CONTENU DU TERRAIN

Après avoir généré le terrain, le défi suivant consistait à y ajouter du contenu. Ce processus a été décomposé en plusieurs étapes clés :

1. Placer les zones
2. Remplir les zones
3. Générer les routes reliant les zones
4. Générer la nature

PLACER LES ZONES

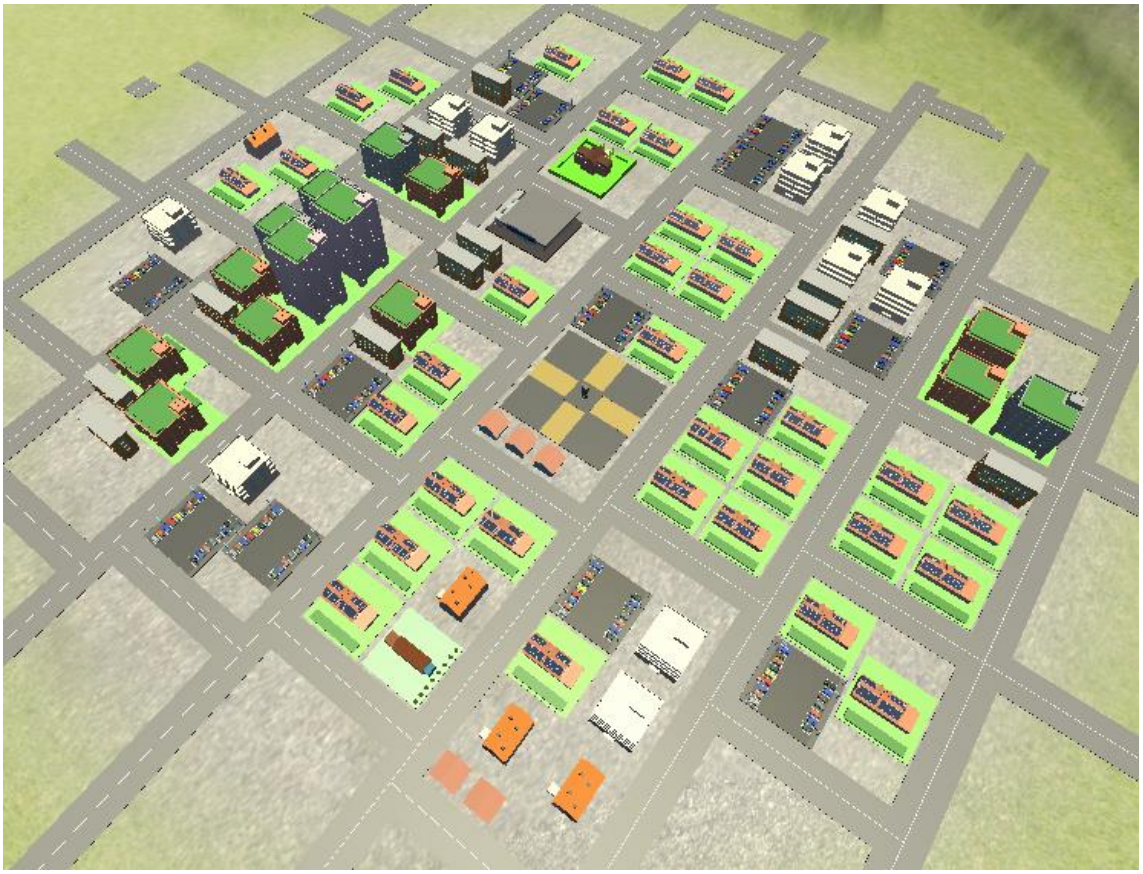
Le principe de l'algorithme de placement des zones est le suivant : nous positionnons chaque zone aléatoirement, puis nous vérifions que celle-ci est suffisamment plate et qu'elle ne chevauche pas une autre zone. En fonction de la génération du terrain et de la taille souhaitée pour chaque zone, ce positionnement peut échouer. Une fois les zones placées, nous les aplatissons afin d'éviter des problèmes ultérieurs lors du remplissage. Cependant, vérifier au préalable que la platitude est sous un certain seuil permet de préserver l'intégrité du terrain déjà généré.



REMPLIR LES ZONES

Chaque zone est représentée dans le code par un tableau à deux dimensions. Pour remplir chaque zone, nous commençons par générer les routes. Les routes horizontales sont générées avec un écart fixe, puis reliées par des petites routes verticales placées aléatoirement suivant une distribution gaussienne.

Ensuite, nous plaçons des bâtiments en utilisant le **Perlin Noise** pour garantir une cohérence dans la disposition des bâtiments, représentant ainsi des quartiers homogènes.

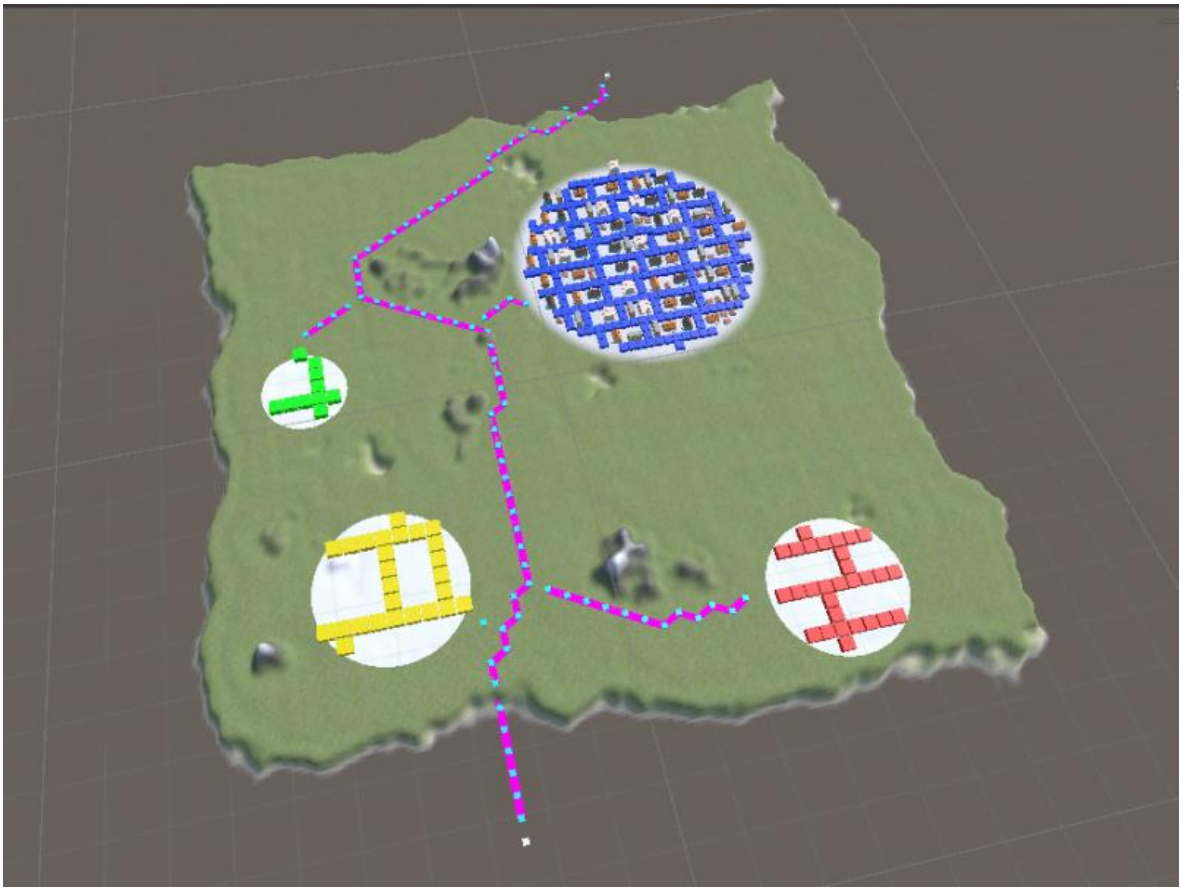


GENERATION DES GRANDES ROUTES

Pour relier les "grandes routes" traversant le terrain et reliant les différentes zones, nous utilisons l'algorithme **A***. Nous commençons par tracer une grande route reliant deux points placés aux extrémités opposées du terrain. L'algorithme A* permet de trouver le chemin le plus rapide tout en évitant les zones à hauteur variable et les zones déjà placées. Nous répétons ensuite le processus pour relier les zones à cette route principale, en connectant à chaque fois le point de la route et la case de la zone les plus proches.

GENERATION VISUELLE DES ROUTES

L'algorithme nous fournit les points par lesquels les routes passent, mais il reste à générer l'aspect visuel. Pour les routes dans les zones, nous utilisions un système de tuiles carrées placées sur chaque case de route. Cependant, ce système ne fonctionne pas pour les grandes routes avec des virages. Nous avons donc décidé de générer nous-mêmes les **meshs** suivant chaque liste de points. Nous avons aplati les vertices du terrain autour des lignes reliant chaque point de nos routes, puis nous avons appliqué la méthode suivante : pour chaque point de la route, nous prenons deux points espacés d'une certaine distance à gauche et à droite en fonction du vecteur de direction de la route (entrant et sortant). Nous créons ensuite deux triangles formant un rectangle représentant la route entre ces points. Bien que le rendu des angles reste brut, le résultat a été jugé satisfaisant.



GENERATION DE LA NATURE

Pour la génération de la nature, nous avons testé plusieurs méthodes. La plus convaincante fut l'utilisation de plusieurs **Perlin Noise** combinés pour instancier des arbres. Cependant, après avoir compilé le projet, nous avons constaté que le casque VR avait du mal à faire tourner le jeu à cause du trop grand nombre d'arbres instanciés, entraînant une surcharge de vertices à calculer.

CONCLUSION

Le développement de ces algorithmes a demandé beaucoup de temps. Cependant, ils apportent une grande valeur ajoutée en permettant de générer des mondes complètement différents sans changer une seule ligne de code. Si nous souhaitons continuer à développer ce projet, ces algorithmes constitueront une base solide pour des améliorations futures, permettant notamment de changer facilement le paysage, les reliefs et donc l'ambiance de la map.

ZONES

Sur la carte, on peut retrouver 4 zones majeurs qui ont toutes leurs spécificités. Le but est de diversifier au maximum les différentes actions possible et d'ainsi, sensibiliser le joueur à de nombreux aspects de l'écologie.

On retrouve ainsi, la zone d'habitation, la zone d'agriculture, la zone industrielle et la zone de production d'énergie. Chaque zone ses défis qui lui sont propre et a un impact sur les indicateurs globaux de la ville.

IA

Les IAs que nous avons eu le temps de développer dans EcoQuest correspondent surtout aux déplacements de différents agents :

- La conduite dans une zone
- La conduite dans les grandes routes
- Le déplacement de personnage dans une place

Ces éléments étaient importants pour le jeu car il permette de ramener un peu de dynamisme dans notre monde.

Ces IAs n'ont pas d'impact sur les indicateurs du jeu mais sont générés en fonction de nos indicateurs et statistiques. En fonction de l'état de la population la répartition des véhicules dans le jeu est différente.

Pour la conduite, nous prenons en compte la position actuelle et la position suivante qui seront des points pour les grandes routes ou des cases pour les zones. Une fois la position suivante atteinte la position suivante deviens la position actuelle et nous calculons une nouvelle position suivante.
Le code reste très simple, nos voitures n'ont pas d'objectif particulier dans le monde et permette surtout de représenter les transports de notre monde.

Une partie importante sont également les scripts qui permette d'instancier ces agents. Nous avons séparé 2 cas : spécifique et généraux. La génération des véhicules dans le monde est un événement général qui arrivera forcément dans le monde. Cependant la génération des bus est un événement spécifique qui dépend si le centre de bus a été instancié ou non. Pour ce cas-là c'est un script spécifique BusGestion qui gérera l'instanciation des bus lors de sa création.

INDICATEURS

LES CHOIX ET L'EVOLUTION DES INDICATEURS DES OBJETS

A l'origine, nous étions partis sur des indicateurs logique mais qui n'avais pas trop de sens. Chaque valeur était un entier entre 0 et l'infini.
Voici nos 10 premiers indicateurs.

- Prix
- Energie
- Pollution de l'air
- Pollution des sols
- Biodiversité
- Ecologie
- Taille de population
- Santé
- Bonheur
- Sensibilité

Cependant, nous avons fait évoluer ces indicateurs pour les rendre plus réalistes, cad que nous pouvons nous baser sur des valeurs concrètes. Nous avons choisi des valeurs qui parlent à tout le monde et dont il est plus facile de se le représenter.

AJOUT DE NOTIONS DE COURT ET LONG TERME

Pour tous les paramètres qui concernent directement l'objet et non son influence sur la population, nous avons ajouté des effets immédiats et sur le long terme.

Prenons l'exemple de l'argent lié à un objet, nous remplaçons la variable prix (sous-entendu prix d'achat) par :

- Le coût de construction de l'objet (constructionCost)
- Le coût de destruction de l'objet (destructionCost)
- Le profit par mois (profitsPerMonth)
- Le coût par mois (lossesPerMonth)

Ce qui rend plus précis et plus réaliste la représentation d'un objet dans le jeu avec cette notion d'effet immédiat et d'effet sur le long terme. Cette notion de long terme débloque une nouvelle dimension pour le joueur qui ne doit pas oublier l'impact de tous les bâtiments.

REMPLACEMENT DES CRITERES D'ECOLOGIES / DE POLLUTIONS

Par défaut, nous avions choisis :

- Pollution de l'air
- Pollution des sols
- Biodiversité

Ces derniers semblent être assez représentatif du niveau d'écologie et de la pollution d'une ville. Mais avec du recul, il est difficile de comprendre précisément à quoi ils correspondent, surtout au niveau du développement du jeu et de l'équilibrage.

Pour simplifier la chose, nous avons remplacé ces indices par des taux concrets.

- **Pollution de l'air → Quantité de CO2 dans l'air (kg)** : L'indicateur pollution de l'air comprend de très nombreux facteurs entre les gaz à effet de serre, les gaz toxiques, les fines particules, etc... Pour une histoire de simplification du gameplay, nous avons résumer la pollution de l'air au taux de CO2, ce qui est aussi plus visuel et plus simple à ce représenter pour le joueur.
- **Pollution des sols → Quantité de déchets produits (tonnes)** : L'indicateur pollution des sols est très important et est très vaste. On peut parler de plein de facteurs : le rejet de déchets plastiques dans l'environnement, la pollution des sols par l'agriculture (engrais / pesticide), pollution des nappes phréatiques. On a alors pris la décision de résumer cet indicateur par la quantité de déchets produits et rejetés dans l'environnement. Finalement, que ce soient des déchets plastiques, des engrais, des pesticides, ce sont tous des déchets solides produits par l'homme qui pollues l'environnement.
- **Biodiversité → Superficie d'espace vert (m^2)** : La biodiversité est en quelque sorte la quantité d'espèce vivante par m2. Ce qui est assez clair mais en même temps difficile à se représenter. De plus, il est compliqué de juger l'influence dont certains objets sur la biodiversité. Ainsi, pour simplifier, nous nous sommes dit que plus il y avait de m2 verts, plus il y avait de biodiversité, donc nous avons directement remplacer l'indicateur biodiversité par la superficie d'espace vert.

EVOLUTION DES STATISTIQUES DE LA POPULATION

Concernant la population, à l'origine, nous avions pour chacun des indicateurs des entiers entre 0 et l'infini mais finalement ça ne voulait un peu rien dire sachant que ça pouvait aller jusqu'à l'infini.

Ainsi, sachant que la population est une liste de citoyens, il était préférable de remplacer cet entier par un taux / un pourcentage. Ainsi dire que 70% de la population est en bonne santé est plus visuel que dire que la santé est à 7346.

Vous trouverez alors des indicateurs variants entre 0 et 1 :

- Santé
- Bonheur
- Sensibilité
- Acceptabilité (non implémenter car dépend des autres facteurs et encore plus)

INDICATEURS CALCULES

Dans une logique d'avoir des indicateurs globaux représentant un peu l'état global du niveau d'écologie de la ville et de la population, nous nous sommes penchés plus en détails sur 2 statistiques : **niveau écologique, niveau sociétal**.

Ces indicateurs sont très importants car ils permettent dans un premier temps de donner un avis global au joueur sur son avancement en terme écologique, il permet aussi de définir si le joueur a réussi ou si le joueur a échoué.

Taux d'écologie (overallEcologyRate) : C'est un indicateur en pourcentage du niveau d'écologie de la ville avec 0% une ville très polluante sans la moindre verdure et avec 100% une ville très verte qui ne rejette pas de CO2 et qui recycle tous ces déchets. Ce taux est une moyenne de 6 scores :

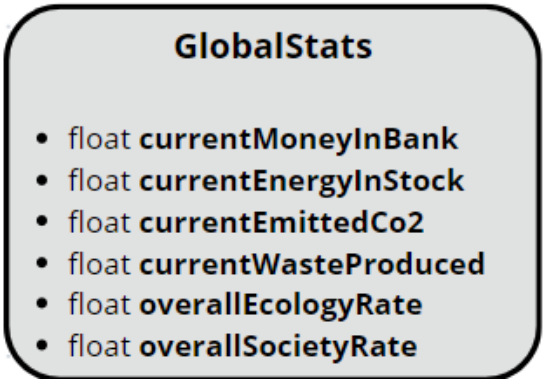
- Sensibilité de la population : 0 ... 1
- Taux d'espaces verts : 0 ... 1 Calcul = taux actuel / max d'espaces verts possible
- Taux de CO2 émis depuis le début du jeu : 0 ... 1 Calcul = 1 - Co2 rejeté / max possible
- Taux de déchets rejetés depuis le début du jeu : 0 ... 1 Calcul = déchets rejetés / max possible
- Quantité de CO2 émise par mois : 0 ... 1
- Quantité de déchets émis par mois : 0 ... 1

Taux sociétal (overallSocietyRate) : Il définit le niveau moyen de la population avec en 0%, une population malheureuse, en mauvaise santé, mal éduqué et en 100%, toute la population est en parfaite santé, très contente et très sensibilisé.

RESUME DES INDICATEURS CHOISIS

Pour plus de clarté dans le code, nous avons défini 3 types d'indicateurs / statistiques.

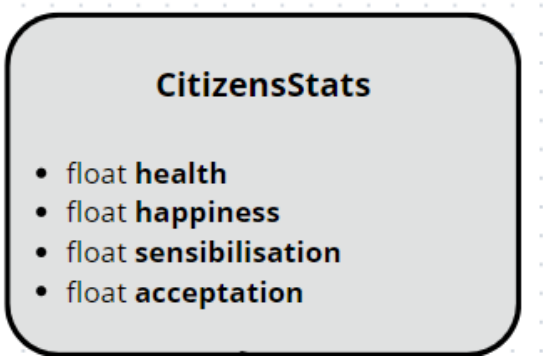
STATISTIQUES GLOBALES (CLASS GLOBALSTATS)



Les statistiques globales dépendent du temps ou sont calculés. En effet, les indicateurs : currentMoneyInBank, currentEnergyInStock, currentEmittedCo2, currentWasteProduced définissent l'état du jeu à un instant précis et ne peuvent pas être recalculé car chaque mois, ils évoluent en fonctions des bâtiments qui modifient les statistiques par mois (c'est à dire quasiment tous).

Dans les statistiques globales sont ajoutés les 2 indicateurs les plus importants : overallEcologyRate et overallSocietyRate. Ces 2 là permettent de définir l'avancement du jeu. En effet, les conditions de défaite et de victoire se basent sur ces 2 indicateurs.

LES STATISTIQUES MOYENNES DES CITOYENS (CLASS CITIZENSSTATS)



Ces informations sont très simplement une moyenne des statistiques de chaque membre de la population.

LES STATISTIQUES LIES AUX OBJETS DE LA MAP (CLASS STATS)

Même si ces statistiques peuvent être globale, nous les avons différenciés de la classe GlobalStats car elles ne dépendent pas du temps et peuvent être recalculé à n'importe quel moment.

Cette classe est présente sur chaque objet de la map. Ainsi, pour calculer ces statistiques globales, nous faisons la somme des statistiques pour chacun des objets instanciés.

SOURCES DE STATISTIQUES

Les statistiques peuvent être réparties en 2 grandes catégories. En effet, les statistiques peuvent être influencé par 2 choses : soit les bâtiments qui sont placés sur la carte, soit les citoyens qui affectent les statistiques lorsqu'ils font des actions.

LES BATIMENTS PLACES SUR LA CARTE

Les bâtiments placés sur la carte possèdent tous des statistiques définies à l'avance par le csv « objects.csv », ils possèdent aussi des niveaux d'améliorations qui augmentent ou réduisent certains des statistiques d'objets.

LA POPULATION

La population ajoute de l'aléatoire dans le jeu. En effet, elle est générée aléatoirement en prenant au hasard des valeurs sur une gaussiennes. Les valeurs de générations aléatoires sont stockées dans CitizensGestion et sont initialisé par le csv « initialPopStats ».

meanHeal	0.5
stddevHea	1
meanHapp	0.5
stddevHap	1
meanSens	0.5
stddevSen	1
meanDista	10
stddevDist	10
meanSalar	45000
stddevSala	17000

Chaque individu est généré de manière aléatoire mais suivant une tendance et influe directement sur la société et l'écologie, notamment par le choix personnel du mode de transport.

LES EVENEMENTS D'INFLUENCE DES STATISTIQUES

La classe StatManager est « le dirigeant » de toutes les classes liées aux statistiques. C'est donc lui qui possède toutes les fonctions évènements appelés par le boss ultime « GameManager ». Sans compter les fonctions d'initialisations, StatManager possède 5 fonctions évènementiels qui régissent toutes les interactions avec les statistiques.

CHOIX DE CARTES PAR LE JOUEURS

L'application d'une carte par le joueur modifie de manière instantanée certaines statistiques du jeu. Ainsi, nous avons dû définir une fonction appelée « UpdateStatsFromCardEvent » qui est appelée à chaque fois que le joueur joue une carte.

Ces actions sur les statistiques :

- **Mettre à jour les statistiques globales** avec « UpdateGlobalStatsFromCard ». D'où son nom, elle permet de manière immédiate, de modifier les informations de GlobalStats, cad l'argent total, l'énergie totale, etc. En effet, on soustrait le coût de la carte et l'énergie demandé pour la réalisation de la carte. Et on ajoute la pollution de CO2 et des déchets ajoutée que l'action engendre.
- **Mettre à jour les statistiques objets** avec « UpdateObjectsStatsFromObjects ». Sachant que dans la majorité des cas une carte construit un bâtiment, le détruit ou l'améliore, ainsi il faut modifier la somme des statistiques liés aux objets.
- **Faire évoluer la population** avec « UpdatePopulationStatsFromCard ». Cette action n'influence pas directement les citoyens. En effet, elle modifie les statistiques globales de génération. Et de cela, on remplace un certain pourcentage de la population (10% par exemple, ce qui va modifier les statistiques globales de la population.
- **Faire évoluer la taille de la population**. Si la carte rajoute ou supprime des habitations, alors on modifie le nombre de citoyen en conséquent.

CHOIX DES TRANSPORTS QUOTIDIENS PAR LES CITOYENS

Nous avons défini 5 secteurs : ville, agriculture, industrie, énergie et transports. Alors que les statistiques liées aux secteurs ville, agriculture, industrie, énergie sont facile à calculer car il faut juste additionner les statistiques de chaque bâtiment construit, il est plus compliqué de calculer les statistiques des transports. Tout simplement car le secteur ne possède pas de zone dans le jeu où l'on pose des bâtiments. Ainsi le calcul est plus complexe et se base sur la pollution de chaque citoyen en fonction de ce qu'il prend.

Ces actions sur les statistiques :

- **Mettre à jour les statistiques objets** par la détermination de mode de transport de chaque citoyen avec UpdateDailyTransportStats. Pour chaque citoyen, en fonction de leur condition de vie (salaire, distance aux travail, santé, etc.) et de pondération (la pondération permet de définir la proportion d'influence de chaque statistique du citoyen pour chaque mode de transport (par exemple, un mec qui n'est pas en bonne santé a beaucoup plus de chance de prendre la voiture)), nous déterminons le mode de transport le plus probable. Dès que le mode de transport est choisi, on modifie les statistiques du citoyen. En effet par exemple, s'il prend le vélo ou s'il va à pied au travail, il fait du sport, il sera donc en meilleur santé. Puis on ajoute les stats de pollution du mode de transport aux stats globales. BTW, si un citoyen prend le bus, il ne pollue pas car qu'un citoyen prenne le bus ou non, le bus passera quand même et polluera quand même par la même occasion.

- **Faire évoluer les statistiques globales de population** par UpdateCitizensStats. En effet, comme les transports influes sur la santé des citoyens, il faut de nouveau additionner les statistiques de tous les citoyens.

id	name	health	happiness	sensibilisa	meanDista	stddevDist	meanSalar	stddevSalary
0	walk	1	0.5	1	2	2	30000	10000
1	car	0.2	0.2	0.1	20	5	50000	30000
2	taxi	0.2	0.2	0.05	10	1	80000	15000
3	bike	0.7	0.5	1	5	2	30000	30000
4	electricSc	0.7	0.8	1	3	1.5	40000	30000
5	bus	0.5	0.5	0.6	15	5	35000	10000
6	subway	0.6	0.5	0.7	10	3	40000	20000
7	tramway	0.6	0.5	0.7	10	3	40000	20000
8	train	0.4	0.4	0.8	50	10	60000	10000
9	plane	0.1	0.2	0.1	1000	100	120000	30000

MISE A JOUR MENSUELLES DES STATISTIQUES GLOBALES PAR LES STATISTIQUES MENSUELLES

De manière mensuelle, pour les indicateurs globaux d'argent, d'énergie, de Co2 et de déchets, on ajoute ce qu'il faut ajouter par mois et on soustrait ce qu'il faut soustraire par mois. Cela se fait par la fonction UpdateGlobalStatsFromObjects.

MIS A JOUR DU TEMPS

Chaque seconde, il faut mettre à jour le temps en l'incrémentant. En effet, notre jeu n'est pas basé sur le temps réel, tout simplement car nous voulons le manipuler comme bon nous semble. Nous voulons l'arrêter, le remettre, l'accélérer et le ralentir.

MIS A JOUR DU DASHBOARD

Dès qu'une modification est faite dans les statistiques, nous devons mettre à jour les informations affichés sur l'écran du dashboard. Le temps se fait par l'intermédiaire d'une autre fonction car il ne modifie pas les statistiques à proprement parlé. Mais tous les autres évènements ci-dessus mettent à jour les statistiques.

Ainsi il faut d'abord recalculer les statistiques calculables comme overallEcologyRate et overallSocietyRate.

Puis nous pouvons afficher tous les statistiques.

EQUILIBRAGE

INITIALISATION DU JEU

Nous devons générer au début un jeu stable, cad qui n'est pas en déficit budgétaire, qui produit assez d'énergie pour subvenir à ces besoins

BATIMENTS PARTICULIERS

Les bâtiments particuliers ont des actions uniques sur les statistiques en plus des actions basiques de consommations d'énergie, de production de CO2 et de déchets :

- **Bâtiments énergies** : permettent de subvenir à la consommation de la ville et des industries.
- **Industries** : permettent de gagner de l'argent au détriment de la pollution. Leurs productions d'argents, de pollution varient en fonction de la taille de la population.
- **Industries des déchets** : permet de détruire les déchets produits par la ville et les industries principalement.
- **Les zones vertes et agricoles** : permettent de réduire la quantité de Co2 en l'absorbant. L'absorption par m2 n'est pas suffisante, ainsi il faut rajouter une variable qui représente l'absorptions des forêts et océans.

Ces bâtiments permettent l'équilibrage des statistiques.

INTERFACES ET INTERACTIONS

Durant le développement d'EcoQuest, nous avons bien veillé à toujours songer au futur utilisateur du jeu. En effet, le public cible n'étant pas forcément habitué à ce genre de technologies, il était primordial de proposer une application facile à prendre en main. Pour concevoir nos interfaces, nous nous sommes donc basés sur les enseignements de réflexion centrée utilisateur que nous avons eu. Voici un aperçu de nos différentes interfaces ainsi que des interactions possibles pour l'utilisateur.

INTERFACES

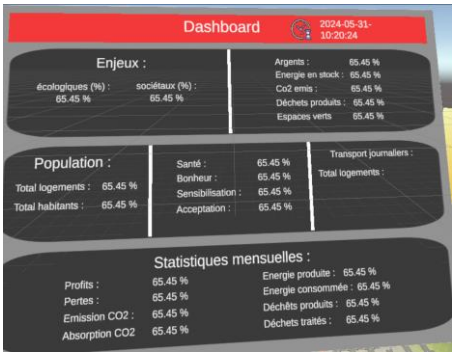
STYLE DES INTERFACES

Pour concevoir nos interfaces, nous voulions quelque chose proche du style low-poly pour que nos menus, et nos différentes zones d'affichage soit accordés avec le reste du jeu. Nous avons donc sélectionné des exemples d'interfaces déjà existante pour nous en inspirer.

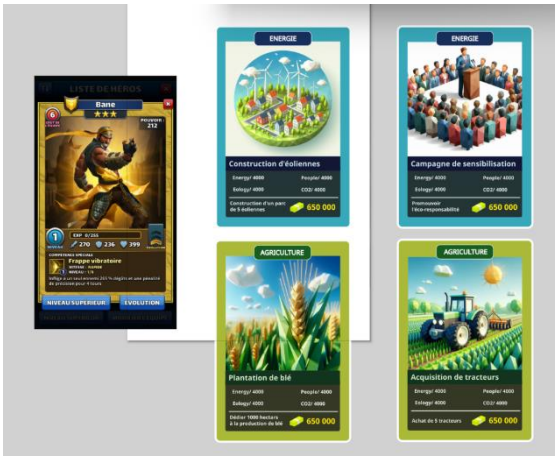
Finalement, nous avons choisis un fond commun pour tous nos menus 2D. De plus, les différents boutons ont tous été fait de la même couleur afin que le joueur les repères facilement. La couleur des boutons a d'ailleurs été choisis car elle va bien avec la couleur de fond des interfaces tout en contrastant suffisamment avec.

Nous avons aussi veillé à respecter l'alignement des différents éléments présents dans les interfaces notamment dans le tutoriel pour respecter au maximum les règles d'ergonomie.

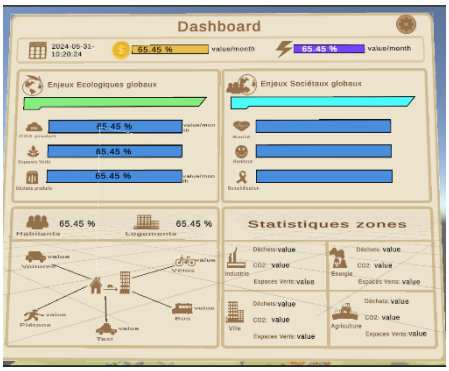
ANCIENNES INTERFACES GRAPHIQUES



NOUVEAUX PROTOTYPES CARTES DE JEUX



RESULTAT FINAL



DETAILS DES INTERFACES

Dans le jeu, on retrouve donc de nombreuses interfaces avec lesquelles le joueur peut interagir :

Le menu principal : C'est le première interface que verra l'utilisateur, elle lui permet de lancer une partie, de faire le tutoriel ou de quitter l'application.

Le tutoriel : Il est composé de plusieurs slides qui expliquent courtement comment jouer.

Les paramètres : Cette interface permet de mettre en pause le jeu. Le joueur a aussi la possibilité de modifier le volume du jeu. Il peut aussi quitter la partie s'il le désire.

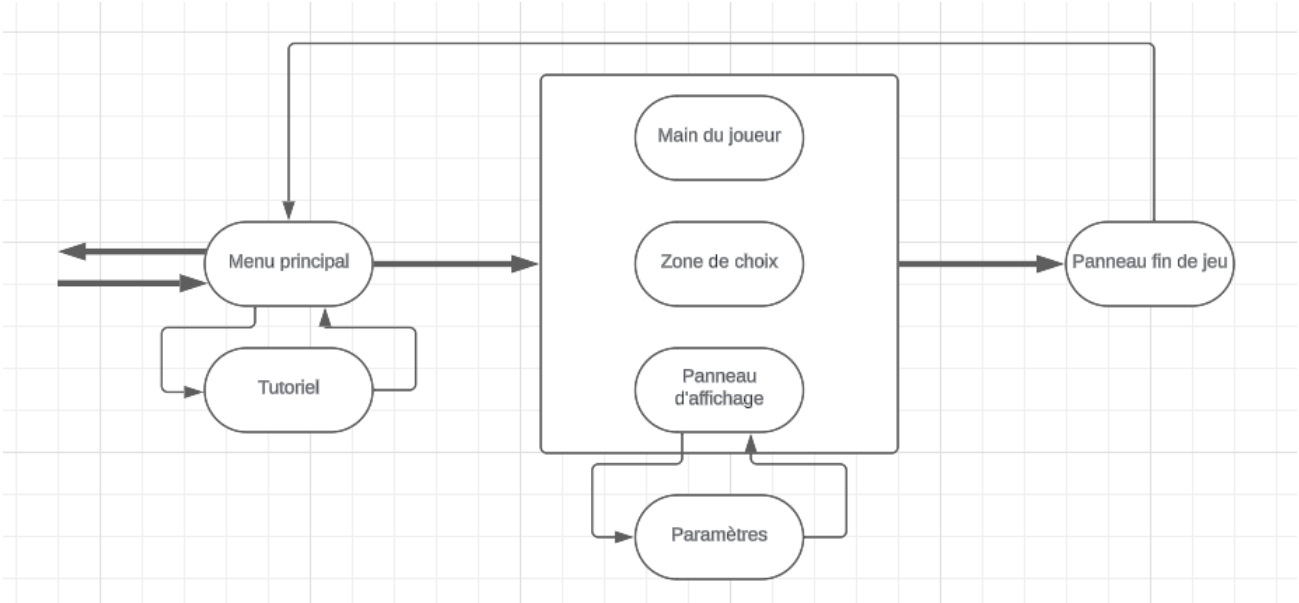
Le panneau d'affichage : C'est la plus grosse zone d'affichage du jeu. Cette interface contient un aperçu des différents indicateurs du jeu. Un bouton permet aussi d'ouvrir les paramètres depuis ce panneau.

La zone de choix des cartes : Cette zone permet de faire choisir ses prochaines cartes au joueur, le joueur a la possibilité de cliquer sur une carte pour la sélectionner puis quand son choix est fait de valider pour que les cartes viennent dans sa main. On y retrouve aussi un petit compteur qui s'actualise pour indiquer combien de cartes ont été sélectionnées.

La main du joueur : Cette interface contient les cartes que le joueur a déjà choisi. Elle apparait lorsque le joueur tourne son avant-bras. Le joueur peut ensuite attraper les cartes qui y sont présentes pour les jouer.

Le panneau de fin de jeu : Lorsque le jeu est fini, une dernière interface est affichée, elle donne la possibilité au joueur de retourner au menu ou de continuer à jouer.

Ce schéma résume les liens entre les différentes interfaces durant la partie.



POLICE D'ECRITURE

En plus de choisir les mêmes éléments graphiques pour les fonds des interfaces et pour les boutons, nous avons également choisi deux polices d'écriture pour tout le projet. Ces deux polices ont été choisis à la suite de la réflexion mené en MV53 sur le style graphique. Ce sont toutes les deux des polices gratuites. La police *Cheque* est utilisée pour les titres et *Exo2* pour le reste des textes. Ces deux polices permettent également de créer une vraie identité graphique pour le projet.

SONS DU JEU

Pour les sons du jeu, nous avons mis un fond sonore différent pour chaque zone. Ensuite, un calcul est fait pour estimer lorsque le joueur est assez proche d'une zone. Ainsi, en se déplaçant sur la carte zoomer, le joueur peut entre les différents fond sonores qui s'enchainent en fonction de la zone afficher. Lorsque la map est dézoomé, il n'y a aucun fond sonore.

Au son des zones s'ajoute les enregistrements du nuage. Comme nous le verrons dans la partie sur le contenu du jeu, le nuage est le personnage principal du jeu et son rôle est important pour guider l'utilisateur et donner des indications. Le nuage a donc toute une liste de phrase qu'il peut dire au fil du jeu. A chaque fois qu'il parle, sa phrase est affichée dans une bulle mais également dites à haute voix. Pour générer les audios, nous avons utilisé une IA en ligne qui permet de donner un résultat plus ou moins réaliste. Cette IA permettait aussi de choisir entre de nombreuses voix, nous en avons essayé de choisir celle qui allait le plus avec le personnage.

Pour finir, nous avons essayé de mettre une musique de fond générale mais nous n'en avons pas trouvé qui nous satisfaisait. De plus cela faisait beaucoup de choses en même temps pour l'utilisateur et nous ne voulions pas que ce dernier ce sente perdu parce que tous ses sens sont trop sollicités.

INTERACTIONS

METAPHORE D'INTERACTION

Nous avons aussi réfléchi à l'expérience utilisateur au moment de choisir les interactions que le joueur devra faire. Toujours dans l'idée de s'assurer que l'utilisateur ne perde pas tous ses repères, nous avons choisis de représenter les actions sous forme de cartes. Cela permet donc de rappeler au joueur le type de jeu qu'il connait. On retrouve donc plusieurs références à un jeu de cartes dans les interactions. Par exemple, les cartes que le joueur a choisies sont retrouvables dans « sa main », pour représenter cela, nous avons mis la zone de deck au niveau du poignet gauche et pour activer cette zone l'utilisateur doit tourner son bras comme lorsqu'il souhaite regarder ses cartes dans une vraie partie. Pour jouer la carte, le joueur doit la saisir, comme il pourrait le faire sur un vrai jeu de carte et la déplacer jusqu'au nuage qui s'occupera de faire l'action.

Pour que l'action à faire soit clair lorsque le joueur prend une carte, nous avons fait en sorte que le nuage se déplace vers le joueur à chaque fois qu'il saisit une carte ce qui attire l'attention sur lui et permet de lui donner la carte facilement.

Ainsi, les différentes interfaces et interactions du jeu ont été faites en essayant au maximum de respecter les éléments que nous avons pu voir en cours de MV53. Ne pas perdre l'utilisateur par une utilisation trop complexe était un vrai défi lors du développement d'EcoQuest et nous avons mis en place tous ces éléments pour lui permettre de garder ses repères et d'apprécier son expérience en AR.

CONTROLES DE LA CARTE

Enfin, pour améliorer l'expérience de l'utilisateur, il était nécessaire de lui permettre de zoomer sur la carte et de s'y déplacer. Nous avons donc mis en place des contrôles de la carte avec les joysticks. Le joueur a donc la possibilité de zoomer, faire tourner la carte et de déplacer la carte.

Pour faire cela, nous avons mis en place un cercle qui délimite ce qui est affiché et ce qui ne l'est pas. Ce cercle est notamment fait grâce à un shader qui s'applique la map et qui permet de cacher les zones en dehors.

Pour le zoom, nous agrandissons la map sans bouger le cercle, cela permet d'agrandir la zone actuelle pour plus de précisions. Lors de cette action, nous faisons également des calculs de position pour que le zoome se fasse par rapport au centre du cercle et pas au centre de la carte.

Pour le déplacement, nous changeons la position de la map sans bouger celle du cercle. Cela permet de donner un effet de glissement et de parcourir la carte quand on est zoomé. Pour cette action nous avons eu un peu de mal à ce que la direction du glissement dépende de la position du joueur. Cela pourrait d'ailleurs être encore amélioré.

Enfin le joueur a la possibilité de faire tourner la carte. Encore une fois, ici seule la map tourne et le cercle reste fixe. Le plus dur avec cette action est qui fallait prendre en compte quel endroit de la carte nous étions en train de regarder. En effet, le centre de la rotation doit être le centre du cercle et pas le centre de la map.

En plus de ces contrôles aux joysticks, le joueur a aussi la possibilité de déplacer la map dans son environnement. Pour cela, nous utilisons les scripts d'OVR qui permettent de saisir un objet et de le déplacer. Cependant, nous avons apporté des modifications à ces scripts pour que quand le joueur déplace la carte, le cercle se déplace avec. Cela permet donc au joueur de prendre la carte et de la déplacer dans l'environnement. L'utilisateur a aussi la possibilité d'agrandir la carte et le cercle en la prenant à deux mains. Ainsi, le joueur a totalement la main pour adapter la carte virtuelle à l'environnement réel qui l'entoure.

ORGANISATION DU CODE

Étant donné la complexité et la diversité des sujets abordés dans notre projet, nous avons mis l'accent sur la propreté du code et la structure de celui-ci. Notre approche se base sur l'implémentation de plusieurs classes "Manager" pour organiser et gérer les différents aspects du jeu de manière efficace.

Classes Manager

Nous avons mis en place plusieurs classes "Manager" pour gérer divers domaines du jeu. Voici une liste des principaux managers utilisés :

- **CardManager** : Gestion des cartes.
- **ObjectManager** : Gestion des objets du jeu.
- **AgentManager** : Gestion des agents.
- **StatManager** : Gestion des statistiques.
- **AnimationManager** : Gestion des animations.

Ces scripts sont des **MonoBehaviour** placés sur des objets vides dans la scène.

GameManager

Tous ces managers étant interdépendants, nous avons créé un **GameManager**. Ce dernier a pour rôle de :

- Gérer les interactions entre les différents managers.
- Initialiser le jeu.
- Gérer le temps.

Structure des Classes

Chaque manager s'appuie sur plusieurs classes pour répartir le travail de manière cohérente. Nous avons principalement utilisé deux types de classes :

- **Classes Simples** : Utilisées lorsqu'il est nécessaire d'avoir une instance de la classe dans un manager.
- **Classes Statics** : Utilisées pour les utilitaires (Utils), contenant des fonctions réutilisables sans avoir besoin d'instance.

Scripts MonoBehaviour

En plus des managers, nous avons de nombreux scripts **MonoBehaviour** attachés à différents objets, tels que :

- **MapController** et **UpdateTerrainRenderer** sur le mesh du terrain.

- **CloudController** sur les nuages.
- **ModelGestion** sur les objets instanciés dans les zones.

Ces scripts interagissent avec les managers pour récupérer les informations nécessaires sur le jeu.

Bénéfices de la Structure

Nous sommes très satisfaits de cette structure pour plusieurs raisons :

1. **Éviter les Effets de Pelote de Laine** : En structurant notre code de cette manière, nous avons évité les interdépendances chaotiques souvent rencontrées dans Unity.
2. **Facilité de Maintenance** : Cette organisation permet de reprendre et de comprendre le code plus facilement, ce qui est essentiel pour la maintenance et les futures évolutions du projet.

CONTENU DU JEU

Pour que le jeu puisse être utilisé et servir à sensibiliser ses utilisateurs, il est très important que son contenu soit bon. Pour cela, nous avons notamment mis en place une histoire avec un personnage principal. Dans le contenu on retrouve aussi toutes les cartes et tous les objets qui ont été créés.

DEROULE DU JEU

Le but du jeu est donc d'améliorer les indicateurs écologiques pour sauver la ville. En plus de la gauge visible sur le panneau d'affichage, nous avons choisis de représenter cet objectif avec un personnage principal : Finn le nuage. Finn est donc un nuage qui change de couleur en fonction de l'état écologique de la ville. Plus la ville est polluée plus Finn est gris foncé. Le but pour le joueur est donc d'aider Finn à retrouver sa couleur blanche en améliorant l'état de la ville.

Le nuage est présent tout au long du jeu, c'est à lui qu'il faut donner les cartes que l'on veut jouer et il a également toute une liste de phrase qu'il dit au fil du jeu pour donner des indications au joueur.

Au fur et à mesure que le jeu avance, le joueur peut donc voir les indicateurs évolués et cela le mène à différents cas de fin de jeu :

- Victoire : Si le joueur arrive à dépolluer totalement la ville, il gagne automatiquement la partie.
- Défaite : Si le joueur n'a plus d'argent ou pas assez d'énergie pour alimenter la ville ou que la population n'est plus assez heureuse, il aura perdu la partie.

En cas de défaite, le joueur a le droit à une petite animation qui lui montre la place du village avec la population entrain de manifester.

CARTES

Le plus important dans le jeu, ce sont les cartes qui vont permettre d'agir sur la ville. Nous avons implémenté 5 types de cartes différents :

- Construction : permet de construire un ou plusieurs nouveaux objets sur la map. Lorsqu'une carte construction est jouée une petite animation zoome sur l'emplacement et nous montre l'objet apparaitre avec un son de construction.
- Destruction : même principe que pour la construction.
- Remplacement : permet de remplacer certains objets par d'autres. L'animation nous montre d'abord les objets disparaitre puis nous montre les nouveaux se créé, toujours avec le son correspondant.
- Amélioration : Permet d'améliorer certains objets en leur rajoutant des sous objets. Par exemple, une carte amélioration ajoute des panneaux solaires à certaines maison. Là aussi, une petite animation nous montre les changements.
- Evènements : ces types de cartes permettent d'influer directement sur les statistiques de la population. Le code est normalement prêt pour ce type de carte mais nous n'en avons créé aucune.

DONNEES

Les données de tous les objets sont aussi très importantes, leur prix, leur consommation et toutes ces informations sont cruciales à l'avancer du jeu. Pour l'instant nous avons mis des données très proche de la réalité, cependant l'équilibrage n'est clairement pas parfait. Il faudrait en théorie essayer pendant des heures et des heures de faire varier les différentes données pour arriver à un jeu équilibré qui offre une belle expérience de jeu à l'utilisateur.

Pour l'instant, l'écart entre les prix des cartes est très grand tout comme celui entre la consommation des objets ou la production d'énergie des différents modes de production.

POUR ALLER PLUS LOIN

Même si ce projet est déjà bien complet, il y a encore de nombreuses choses qui pourraient être ajoutées. Soit que nous avons déjà commencé à implémenter soit que nous avons imaginés mais pas eu le temps de développer.

Voici donc une liste non exhaustive des éléments que nous aurions aimés développer en plus :

- Faire plusieurs versions du terrain en jouant sur les paramètres
- Permettre au joueur de choisir lui-même les paramètres de la génération de la map
- Faire une version améliorée du tutoriel qui s'intègre pleinement dans le jeu
- Ajouter plus de choses vivantes sur la map, comme des oiseaux ou des nuages
- Donner plus de place à Finn le nuage et développer l'histoire autour du jeu
- Ajouter des évènements à différents moment du jeu
- Améliorer les designs des interfaces graphiques
- Rajouter des effets spéciaux lorsqu'une carte est jouée
- Mode multijoueur

CONCLUSION

Ce projet était très ambitieux et nous sommes incroyablement surpris de ce que nous avons réussi à faire et du résultat que nous obtenons. Bien sûr, c'est dommage de voir tout ce qui pourrait être fait et ce que nous n'avons pas eu le temps de finir. Mais nous sommes tous les quatre fiers du travail accompli.

Nous avons fait énormément de choses sur ce projet et nous en oublions sans doute une partie dans ce rapport. Il reste néanmoins encore beaucoup de choses qui pourrait être faites. Le plus gros manque selon nous reste l'équilibrage des données liées aux objets et la diversité des cartes à jouer. C'est en effet ce qui empêche ce jeu d'être parfaitement jouable et de sensibiliser correctement le joueur. Le code est donc totalement prêt pour le jeu idéal mais il faudrait encore passer beaucoup de temps sur ces données et sur les cartes.

Si Life V-Air voulait vraiment exploiter ce jeu, il faudrait donc améliorer certains détails, revoir précisément les données du jeu et rajouter de nouvelles cartes. Les différents éléments abordés précédemment pourraient aussi être ajoutés.

En ce qui concerne, il se pourrait que Marius reprenne notre travail pour en faire un projet entrepreneurial, cependant rien n'est encore acté.

Pour conclure, EcoQuest était un projet ambitieux qui nous as poussé dans nos limites et nous as permis d'en apprendre énormément sur le logiciel Unity et sur la création de jeu de manière générale. Il nous semble clair que le projet a un très gros potentiel mais cela nécessite encore des ajustements. Bien sûr, aucun jeu n'a été parfait lors de sa première version et on espère donc que le EcoQuest 4.0 permettra finalement bel et bien de sensibiliser un maximum de personnes à ces enjeux écologiques qui sont bien réels et si important pour notre futur.