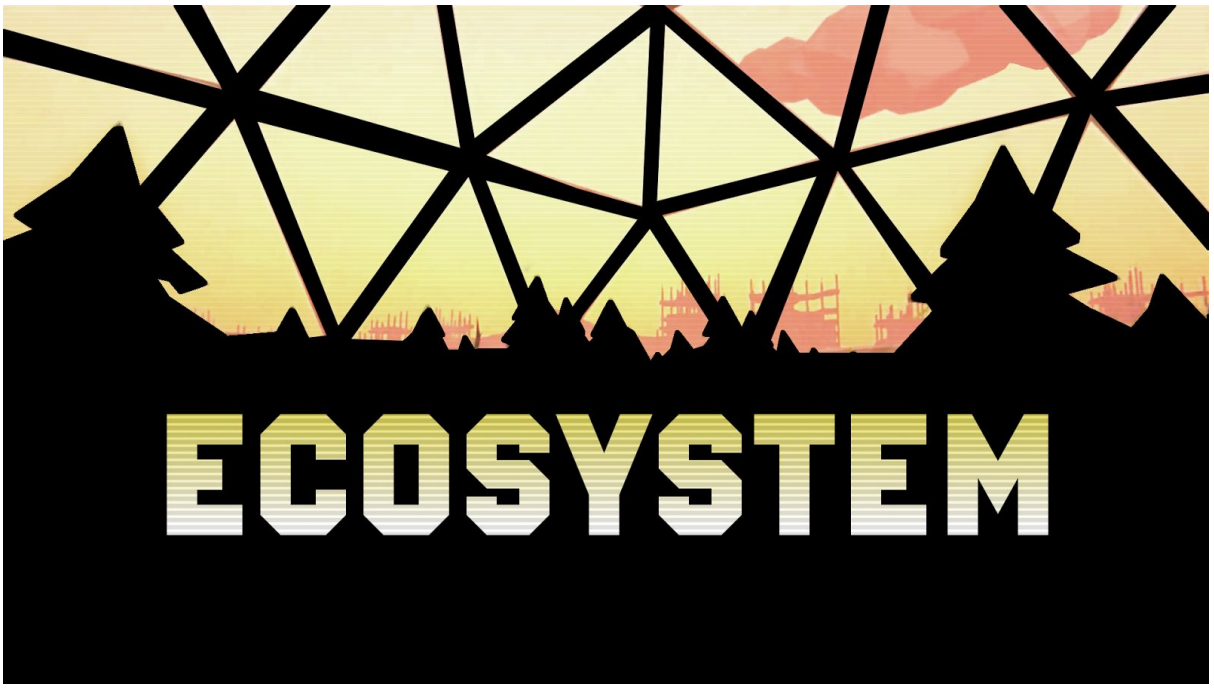


Équipe

Emmanuel SCHMÜCK (Master 2 Prog)

Maeva LAUZIER (Master 2 Prog)

Nicolas KABIRY (Master 2 Prog)

Matthias GAILLARD (Master 2 Prog)

Alfred BRISAC (DU Level Design)

Nathan GIBAUD (DU Level Design)

Rémi MOREAUX (DU Level Design)

Emmanuel BOURREAU (DU Info3d)

Sommaire

Fiche Signalétique

Nom provisoire: EcoSystem

Support: PC

Genre: Survie/Gestion

Vue: Première personne

Thème: Écologie

Nombre de joueurs: 1

Âge: 12+

Contrôles: Clavier/Souris ou Manette

Durée d'une session: 20 à 30 minutes

Pitch

Sur une terre dévastée, le joueur incarne un robot explorant de gigantesques dômes qui reproduisent des écosystèmes. Il doit y construire des habitations et ramener la vie humaine dans cette nature épargnée, sans détruire son équilibre fragile.

Scénario

Le jeu prend place dans un lointain futur. La terre est devenue un désert inhabitable. La nature ne survit que sous quelques étranges dômes isolés.
Le joueur incarne un robot dont le but est de développer des villages humains sous ces dômes.
Pour ceci, il doit construire différents bâtiments en veillant à ce que la pollution ne perturbe pas l'équilibre de chaque écosystème.
Il est guidé par la voix d'une mystérieuse intelligence artificielle. Cette dernière l'aide ponctuellement et lui parle d'une ancienne civilisation, à l'origine du fléau qui a détruit la nature.
En début de partie, l'intelligence artificielle se présente comme un allié précieux. Elle explique au joueur ses objectifs et le conseille sur la marche à suivre.
Au fur et à mesure de la partie, le joueur va apprendre à se méfier d'elle. En effet, elle aura tendance à lui mettre des bâtons dans les roues. Fait-elle ça dans un but précis ou est-elle tout simplement défaillante? Quoi qu'il en soit, le joueur devra gérer les caprices de l'IA tout en menant à bien sa mission.
En recoupant les informations trouvées dans le dôme et celles divulguées par l'IA dans ses dialogues, le joueur pourra comprendre les intentions réelles de cette dernière.

En réalité, c'est l'IA qui est à l'origine de la destruction du monde. Elle estimait que l'espèce humaine courrait à la destruction de la planète. Elle a donc décidé de tout recommencer à zéro en prenant le contrôle de l'évolution de l'humanité.
Une question restera, cependant, en suspend: est-ce que l'IA est défaillante et repose sur une logique erronée ou a-t-elle choisit la seule option permettant de sauver la planète de l'humanité?

Unique Selling Points

- **Un gameplay riche dans un écosystème simulé qui réagit aux actions du joueur.**

- **Une sensibilisation à l'écologie et au développement durable par le gameplay**

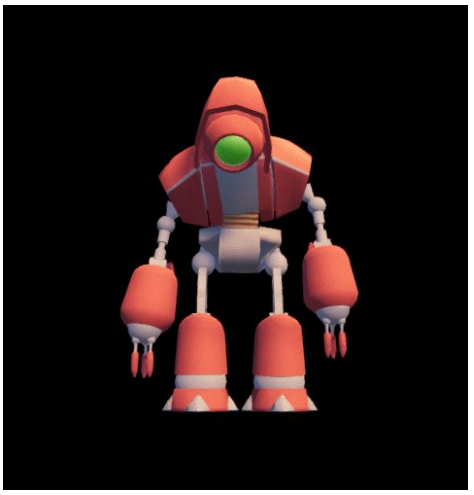
Intentions de réalisation

Le but du projet est de proposer un gameplay varié dans un monde dynamique et cohérent. Le joueur pourra interagir de différentes façons avec un environnement qui évolue devant lui en fonction de ses actions.
Il intègre également un message écologique et permet de sensibiliser le joueur aux problèmes du développement durable. Le joueur sera confronté à des dilemmes actuellement vécus par l'humanité.

Résumé de Gameplay

Le joueur incarne un robot qui doit construire un village habitable afin d'atteindre son objectif de population dans un dôme à l'écosystème riche mais fragile. Il doit faire preuve de discernement afin de ne pas être à l'origine de l'extinction d'une espèce, sous peine de perdre la partie.
Il doit explorer son environnement pour trouver des ressources qu'il va utiliser pour construire différents bâtiments. Il pourra découvrir d'autres robots cachés afin de les réactiver et de les contrôler. Au fur et à mesure de la partie, le joueur pourra découvrir différents éléments de scénario et en apprendre plus sur l'intelligence artificielle qui guide ses pas. Il aura accès à de nouvelles zones et à de nouveaux bâtiments. Ces bâtiments auront toujours des désavantages et des avantages (coût / nécessitant des matériaux rares/ plus ou moins polluants). Les actions du joueur, et en particulier la production d'énergie, ont un impact sur la température du dôme. Ainsi, si le dôme est trop chaud ou trop froid, le joueur sera incapable d'y faire vivre les villageois ou certaines espèces.
Si le joueur se fait toucher, (en se faisant attaquer par des ennemis, par exemple) il perd de l'énergie. Un robot qui n'a plus d'énergie doit être réactivé. Le joueur fait game over et doit recharger sa sauvegarde si tous ses robots sont en pannes.
Une fois sa mission accomplie, le joueur a accès à d'autres dômes aux caractéristiques différentes. Il aura pour but d'y implanter l'espèce humaine en conservant l'équilibre de leurs écosystèmes.

Mécaniques de base



Actions

- Se déplacer
- Explorer
- Détruire
- Récolter
- Planter
- Recycler
- Activer des robots
- Construire
- Interagir avec le dome
- Découvrir des indices sur l'ancienne civilisation (bribes de scénario)

Modes du robot

Le joueur peut basculer du mode exploration au mode construction à tout moment.

Actions disponibles dans tous les modes

Se Déplacer (touches ZQSD)

Le personnage peut avancer et tourner dans toutes les directions.

Courir (touche SHIFT enfoncée)

Le personnage se déplace plus vite

Sauter (Barre Espace)

Le personnage peut sauter et atteindre les hauteurs.

Viser (Souris)

Le personnage peut scruter les alentours (caméra) ou viser spécifiquement un objet ou un ennemi.

Mode exploration

En mode exploration, le joueur peut interagir avec l'environnement pour récolter des ressources. Il peut choisir de tuer ou de neutraliser les animaux qu'il croise.

Recycler/Tuer (clic gauche)

Le personnage peut tuer un être vivant ou recycler un élément de l'environnement en maintenant un rayon sur sa cible. Une fois la cible détruite, elle est remplacée par les ressources qui lui correspondent.

Déplacer/Geler (clic droit)

Le personnage gèle instantanément la cible qu'il a visé. En maintenant son rayon, il peut déplacer sa cible à sa guise. Une fois la cible relâchée, elle reste gelée pendant quelques secondes.

Mode construction

Construire un bâtiment (clic gauche)

Le bâtiment sélectionné est construit à l'endroit ciblé si le joueur possède les ressources nécessaires.

Recycler un bâtiment (clic droit)

Le bâtiment ciblé est détruit. Il est remplacé par une partie des ressources qui ont été utilisées pour le construire. Avec un robot non spécialisé, le joueur peut récupérer 50% des matériaux. Mais ce pourcentage peut être amélioré en utilisant un robot recycleur.

Changer de bâtiment (molette)

Faire tourner la molette permet de sélectionner le bâtiment suivant.

Survie

Lorsque le joueur explore l'environnement, il doit prendre en considération les dangers qu'il va rencontrer. En effet, ils peuvent endommager le robot qu'il incarne.
Un robot trop endommagé est désactivé. Il devra être réactivé pour être à nouveau utilisable. Si le joueur n'a aucun autre robot actif, il perd la partie. Dans le cas contraire, il prend automatiquement le contrôle d'un autre robot actif.

Carnivores

Lorsqu'un robot est incarné par le joueur, il se fait attaquer par les carnivores qui le prennent pour une proie potentielle. Chaque attaque de carnivore endommage le robot.

Températures extrêmes

Lorsqu'un robot incarné par le joueur se trouve dans une zone avec une température extrême (Désert chaud ou Désert glacial), il perd de l'énergie petit à petit et risque de se désactiver.

Améliorations

Robots supplémentaires

En explorant le monde, le joueur peut trouver des robots cachés et désactivés, afin de les redémarrer. A tout moment, dans le menu PAUSE, le joueur peut prendre le contrôle d'un des robots qu'il a activé.

Spécialisation des robots

Chaque robot a une spécialisation. Les spécialisations sont indiquées sur la carte des robots activés dans le menu PAUSE.

Recycleur

Un robot recycleur permet de recycler plus de ressources sur un élément.

Neutraliseur

Un robot neutraliseur peut geler les animaux pendant une plus longue période.

Explorateur

Un robot explorateur est plus résistant aux températures extrêmes et aux prédateurs.

Annihilateur

Un robot annihilateur peut détruire les formes de vie de façon instantannée. Mais il ne récupère aucune ressource lorsqu'il les détruit.

Gestion/construction

Construction des bâtiments

Le joueur peut construire différents bâtiments à l'aide des ressources qu'il récolte. Les bâtiments d'habitation permettent d'augmenter la population maximale du dôme. Pour alimenter les habitants, le joueur devra construire des bâtiments de production de nourriture ainsi que des générateurs d'électricité.

Gestion de la température

Certains bâtiments posés par le joueur créent de la pollution. Cette dernière a pour effet d'augmenter progressivement la température du dôme. Pour contrebalancer cet effet, le joueur a la possibilité de planter des arbres qui absorbent une partie de la pollution.
Il a également la possibilité d'enclencher un dispositif d'urgence qui recouvre le dôme. Ceci a pour effet de diminuer la température du dôme mais empêche les arbres d'absorber la pollution. Ce n'est donc pas une solution viable sur le long terme.

Gestion des radiations

Certains bâtiments posés par le joueur créent des déchets radioactifs. Ces derniers ont pour effet de tuer tout ce qui les entoure. Ainsi, le joueur doit choisir une zone qui ne comportera aucune vie. Il a la possibilité de déplacer les déchets mais ne peut pas les détruire.

Ecosystème

L'écosystème du dôme fonctionne selon des règles et peut être perturbé par les actions du joueur. En effet, plus le joueur pollue le dôme, plus la température augmente.

La chaîne alimentaire

Chaque espèce (animale ou végétale) ne peut vivre que dans une certaine plage de températures. Ainsi, les cerfs se trouveront plutôt dans les zones tempérées alors que les zèbres se trouveront, en général, dans les zones chaudes.

Végétaux (arbres, herbes, céréales...)

Ils poussent sur le terrain qui leur correspond. Leur nombre varie en fonction du nombre d'herbivores.
Outre la mort naturelle, les végétaux meurent lorsque le terrain sur lequel ils se trouvent ne leur correspond plus. Ils peuvent également être mangés par les herbivores.

Herbivores (cerfs, zèbres…)

Chaque espèce peut vivre dans une certaine plage de températures. Ils recherchent des végétaux pour se nourrir. Leur nombre varie en fonction du nombre de végétaux et du nombre de carnivores.
Outre la mort naturelle, les herbivores meurent lorsqu'ils se trouvent dans une zone avec une température trop basse ou trop élevée. Ils peuvent être mangés par les carnivores.

Carnivores (loups, ours...)

Chaque espèce peut vivre dans une certaine plage de températures. Ils investissent les ruines et ont besoin de se nourrir d'herbivores. Leur nombre varie en fonction du nombre d'herbivores.
Outre la mort naturelle, les carnivores meurent lorsqu'ils se trouvent dans une zone avec une température trop basse ou trop élevée.

La mort des animaux est synonyme d'un dérèglement de la chaîne alimentaire. Un manque de prédateur fait augmenter le nombre de proies qui deviennent une menace pour la végétation. A l'inverse un manque de végétation affament les populations d'herbivores et menace les prédateurs.

Les biomes

Les biomes sont des zones qui s'activent via un index défini en fonction de l'altitude, de la température et de la proximité avec l'eau (humidité).

Tempéré

Température minimale: 10°C
Température maximale: entre 30 et 45°C en fonction de l'humidité
Terrain vert
Espèces présentes: Loups - Cerfs - Chênes - Céréales - Herbes vertes

Savane

Température minimale: entre 30 et 45°C en fonction de l'humidité
Température maximale: entre 45 et 60°C en fonction de l'humidité
Terrain jaune
Espèces présentes: Lions - Zèbres - Cactus - Herbes jaunes

Désert chaud

Température minimale: entre 45 et 60°C en fonction de l'humidité
Terrain jaune
Espèces présentes: Aucune

Taïga

Température minimale: -5°C
Température maximale: 10°C
Terrain blanc
Espèces présentes: Ours polaires - Lièvres - Sapins - Herbes blanches

Désert glacial

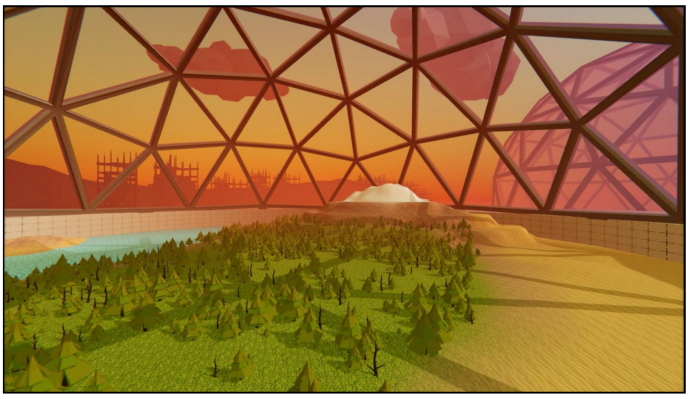
Température maximale: -5°C
Terrain blanc
Espèces présentes: Ours polaire, pingouin

La température

Les actions du joueurs peuvent engendrer des modifications sur la température. La pollution, par exemple, réchauffe progressivement le dôme. La fermeture du dôme peut également engendrer une baisse rapide de la température.
Les variations de températures modifient les biomes, et les déplacent. Animaux et végétaux ne peuvent survivre que dans une certaines plage de température. De plus, le niveau de l'eau est affecté par la température.

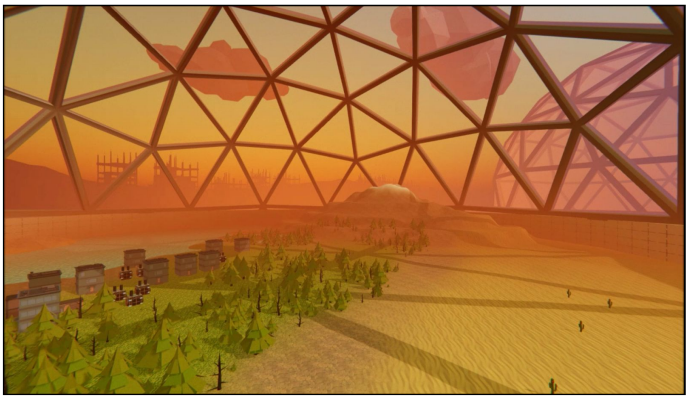
Température de départ

Présence d'une zone tempérée, d'une zone désertique et d'une zone de montagne enneigée.



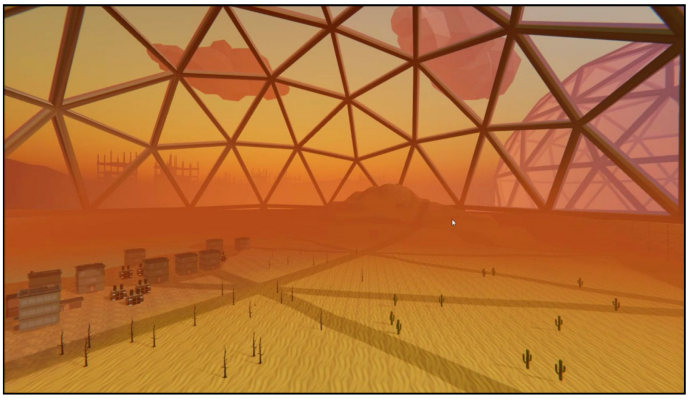
En milieu de partie

Les actions du joueur ont réchauffé le dôme, le désert a gagné en surface au profit de la zone tempérée et la neige a commencé à fondre.



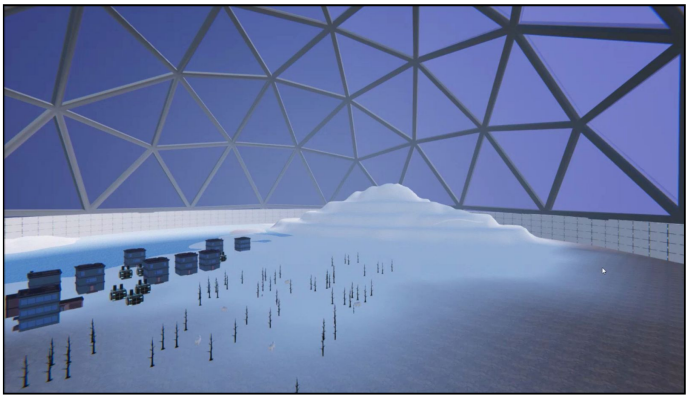
Dôme trop chaud

Si les actions du joueur augmentent trop la température, le dôme entier peut devenir désertique. Les espèces animales disparaissent et le joueur se rapproche d'une situation de gameover.



Dôme trop froid

Le joueur peut opacifier le dôme pour empêcher le soleil d'y pénétrer et ainsi baisser la température. Cependant cette action peut vite déclenchée une ère glaciaire dans le dôme, ce n'est donc pas une solution viable à long terme.



L'eau (humidité)

Animaux et végétaux ont besoin d'un minimum d'humidité pour survivre. L'humidité décroît en fonction de la distance avec le point d'eau le plus proche. Animaux et végétaux ont tendance à se rapprocher des points d'eau.

Les points d'eau peuvent voir leur volume varier en fonction des conditions climatiques (glace / évaporation), ou des actions du joueur (construction d'un barrage).

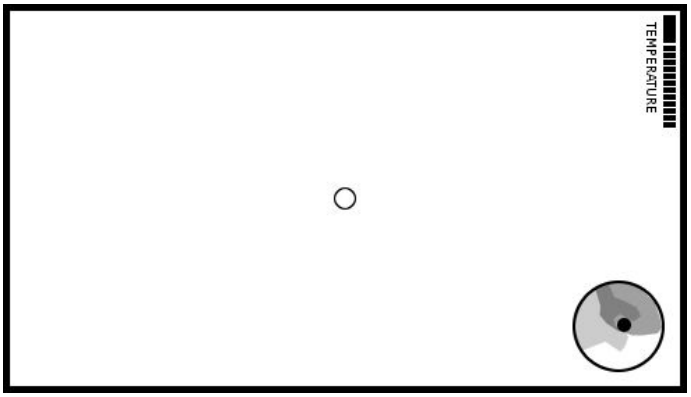
Ce point est à développer autant au niveau game design que programmation; c'est une source de nouveauté pour la suite.

In-Game overview

Interface Utilisateur

Mode exploration

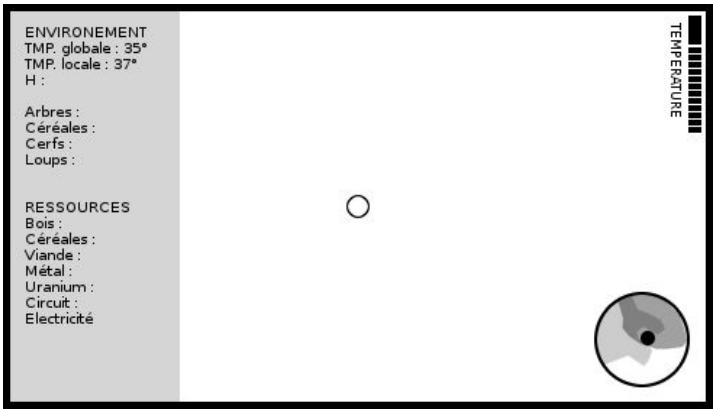
- Réticule
- Jauge de température
- MiniMap du dôme



Inventaire et Stats (Appuyer sur TAB pour afficher, effacer)

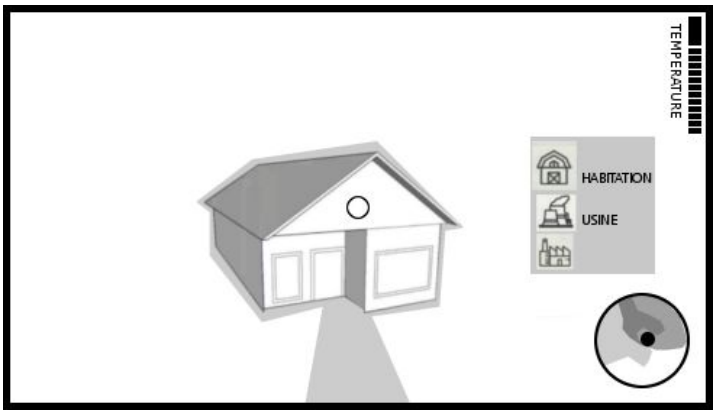
Affichage en temps réel (le joueur peut continuer à jouer en même temps) de :

- Températures globale et locale (détail)
- Statistiques du Dôme (nombre d'animaux, de végétaux, humidité)
- Statistiques du robot (Spécialité, énergie)
- Ressources possédées (Viande, Céréales, Métal, Uranium…)



Construction (Appuyer sur A pour changer de mode)

- la fenêtre affiche les détails des bâtiments constructibles
- les bâtiments que le joueur ne peut pas construire actuellement sont grisés (manque de ressource)



Enchaînement des menus

Ecran Titre

- Nouvelle partie
- Charger une partie
- Paramètres
- Quitter le jeu

Nouvelle partie :
Le joueur a le choix de faire le tutoriel ou pas. Il lui sera conseillé de le faire si c'est sa première partie.
Voulez-vous passer le tutoriel ?

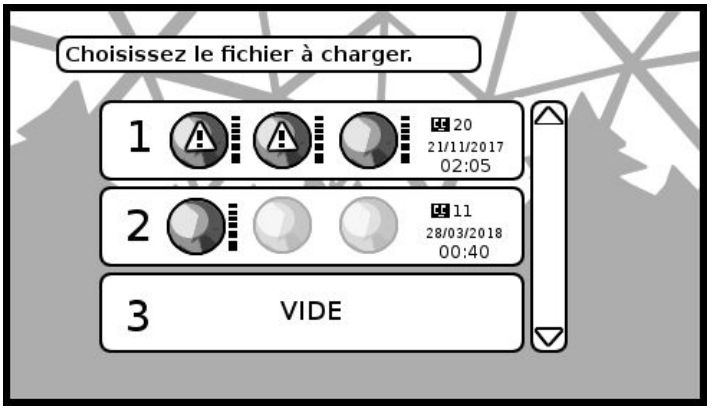
- Oui
- Non

Si le joueur choisit oui, il sera téléporté dans le premier dôme (petit, tempéré, tutoriel). Si il choisit non, il pourra alors choisir entre plusieurs grands dômes.

Charger une partie
Le joueur accédera aux slots de sauvegarde.
Slot de Sauvegarde vide : Empty
Slot de Sauvegarde utilisé :

- Numéro du Slot
- Temps de Jeu
- Date et heure dernière sauvegarde
- Icônes des dômes débloqués (trois à quatre)
- Température des dômes débloqués
- Robots activés

En cliquant sur le slot de sauvegarde correspondant, le joueur pourra reprendre sa partie.



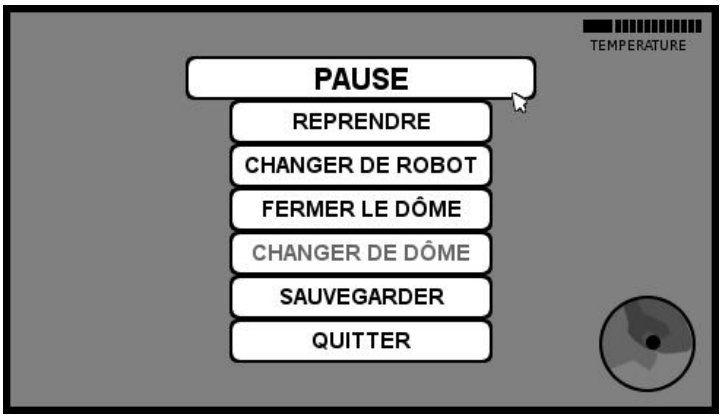
Quitter le jeu :
Êtes-vous sûr de quitter le jeu ?

- Oui
- Non

Menu PAUSE

Le Menu Pause se lance quand on appuie sur ECHAP.
Contrairement à l'inventaire, le jeu n'est plus en temps réel et se fige. Le joueur peut diriger le curseur de la souris pour choisir une option :

- Reprendre
- Changer de Robot
- Fermer / Ouvrir le dôme
- Changer de Dôme
- Sauvegarder
- Quitter



Reprendre (ou ECHAP) :
Reprendre sa partie et quitter le menu Pause.

Changer de Robot :
Une carte avec la position de tous les robots actifs s'affiche et le joueur peut cliquer sur l'un d'eux pour changer de point de vue et le contrôler. La spécialisation de chaque robot est également indiquée.

Fermer / Ouvrir le Dôme :
Les volets du dôme s'ouvrent ou se ferment, ce qui influe sur la température. (la température baisse dans un dôme fermé)

Changer de Dôme :
Le joueur peut téléporter son robot actuel dans un autre dôme. (seulement si au moins un autre dôme est débloqué)

Sauvegarder :
Sauver sa progression à tout moment.

Quitter :
Quitter le jeu / revenir à l'écran-titre.

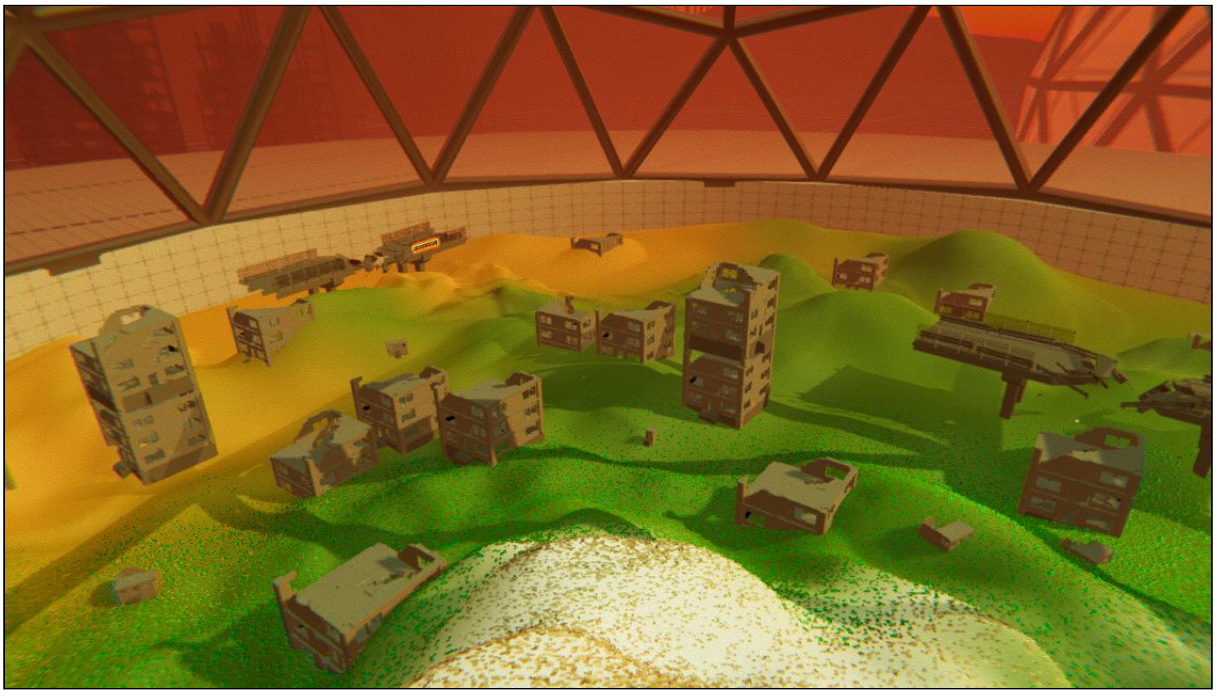
Level Design

Levels concepts

Taille : 500m à 1000m de diamètre
Quantité : 4 à 10 niveaux
Temps de jeu : de 20 min à >1h par niveau

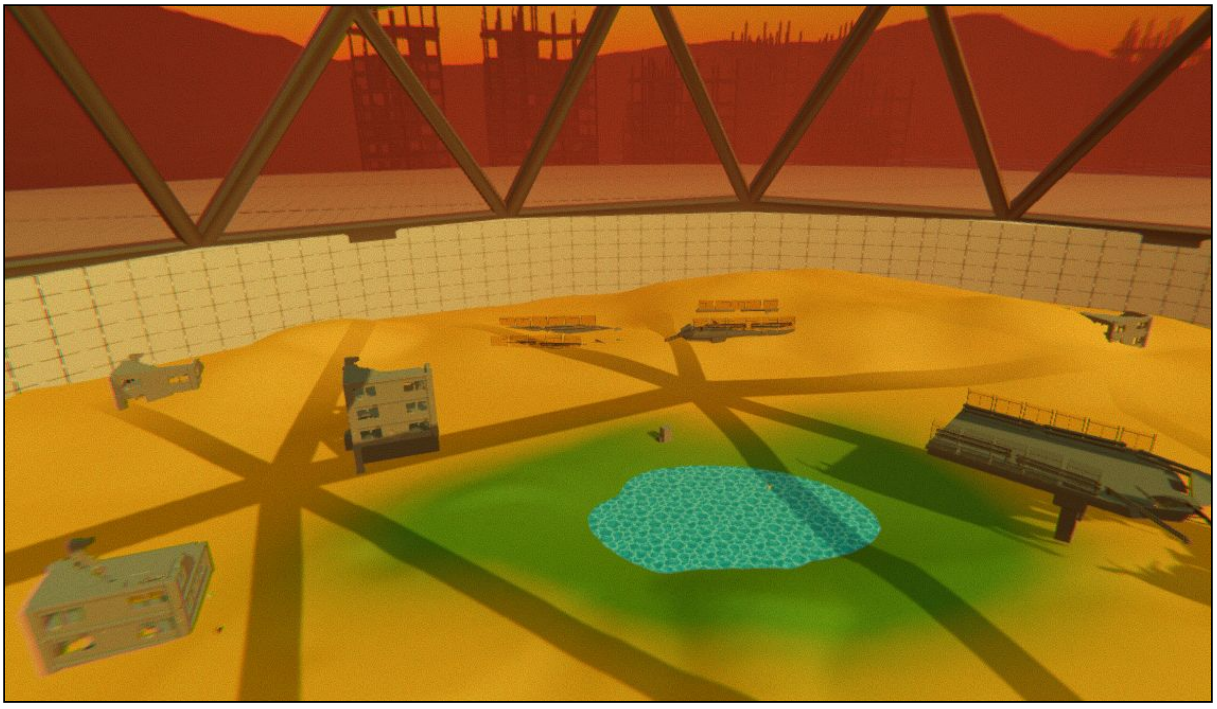
Les niveaux peuvent varier par leur température, leur relief, leur quantité d'eau, leur quantité de ruines, et leur espèces d'animaux.
Certains niveaux peuvent être inhospitaliers au début de la partie, et doivent être rendus viables avant de pouvoir se développer.
Différents éléments de scénarios pourront y être amenés sous la forme d'événements scriptés.

Ville en ruine



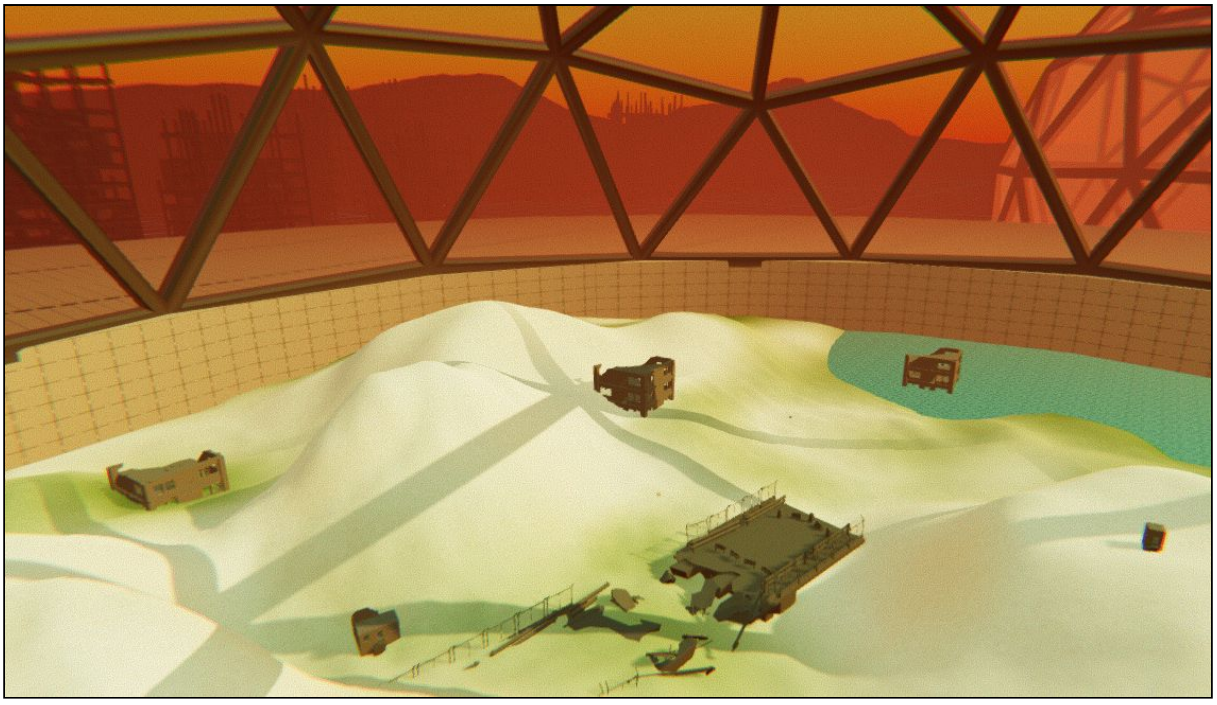
- Dôme tempéré à relief varié
- Grande quantité de ressources
- Importance de l'exportation
- Animaux: cerf, loup, ours,...

Oasis de la savane



- Dôme avec très peu de relief
- Très peu d'eau
- Animaux: zèbres, girafes, éléphants, tigres,...

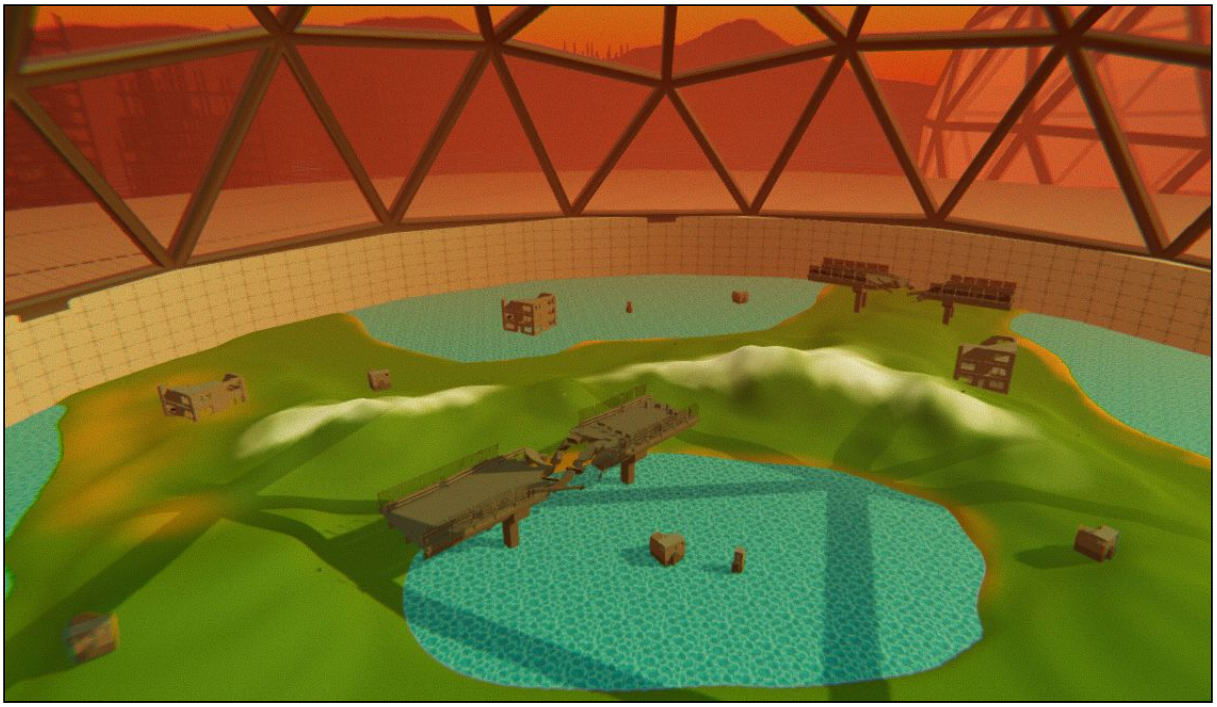
Montagnes enneigées



- Dôme à fort relief
- Basse température de départ

- Animaux: Ours, loups, élans,...

Grands lacs



- Dôme à relief varié
- Beaucoup d’eau
- Animaux: cerfs, loups, ours,...

Environnement

Robots

Des robots désactivés, semblables à celui que contrôle le joueur, sont présents dans l'environnement. Une fois que le joueur les a trouvé, il peut les réactiver afin de pouvoir incarner plusieurs robots à tour de rôle.

Ruines

Plusieurs ruines sont présentes dans chaque dôme. Elles peuvent être explorées afin d'y

trouver des composants rares tels que des circuits imprimés ou de l'uranium. Elles sont, cependant dangereuses car elles sont souvent investies par les carnivores qui les utilisent comme des tanières.

Les panneaux de contrôle

Plusieurs panneaux de contrôle sont présents dans le dôme. Outre les quelques éléments de scénario qu'ils fournissent, ils permettent au joueur de faire appel à l'IA pour recouvrir le dôme.
Les panneaux de contrôle servent également d'interface au joueur pour la spécialisation de ses robots.

Découpage du jeu

Le jeu se découpe en plusieurs missions. Chacune d'entre elle demande d'atteindre l'objectif de population sans détruire l'écosystème du dôme pour être réussie.
Chaque mission se déroulera dans un dôme différent, proposant des caractéristiques uniques.

Déroulement d'une mission

Départ

Au début de chaque mission, le joueur apparaît dans un dôme qui contient un village plus ou moins développé. Son but est de développer ce dernier jusqu'à atteindre l'objectif de population fixé. Il doit alors commencer à explorer son environnement et à récolter des ressources.

Milieu de mission

Lors de sa mission, le joueur doit faire attention à ne pas trop dérégler l'écosystème. En effet, certaines variables peuvent avoir un très grand impact sur tout le dôme (Une température trop extrême peut créer des zones désertiques et aboutir à l'extinction d'une espèce, par exemple).
Plus la mission avance, plus le joueur a d'outils à sa disposition. Il débloque de nouveaux robots, de nouveaux bâtiments, etc...
Le joueur doit alors se servir de tous les outils à sa disposition pour trouver un équilibre entre les besoin de son village et les besoins du dôme.

Conditions de victoire

La mission est une réussite si l'objectif de population humaine est atteint.

Conditions de défaite

La mission est un échec si l'une de ces conditions est remplie:

- tous les robots sont désactivés
- une espèce disparaît du dôme (sa population atteint 0 individu)
- le village ne comporte plus aucun humain

Durée d’une mission

La durée moyenne d’une des premières missions est de 20 minutes. Les dernières missions seront plus longue, leur durée pourra dépasser une heure.

Débloquer de nouvelles missions

Au début de la partie, le joueur n’a accès qu’à la mission tutoriel et la première mission. C’est en réussissant les missions qui lui sont proposées que le joueur peut en débloquer d’autres.
Les missions débloquées en dernier sont les plus difficiles et les plus longues.

Recommencer une mission

A tout moment, le joueur peut décider de recommencer une mission, qu’il l’ait réussi ou non. Ceci lui permet de tester différentes stratégies de développement. Il pourra ainsi voir quels sont les risques inhérents à chacunes et dans quelles situations elles sont plus appropriées.

Direction artistique

Style graphique

Le jeu aura un style graphique “Low Poly” ce qui implique d’avoir des assets graphiques très bas en nombre de polygons.

Les modèles graphiques seront en “flat shading” afin qu’il n’y est pas de lissage de lumière sur la surface des modèles, permettant ainsi d’obtenir un rendu plat sur chaque face des modèles.

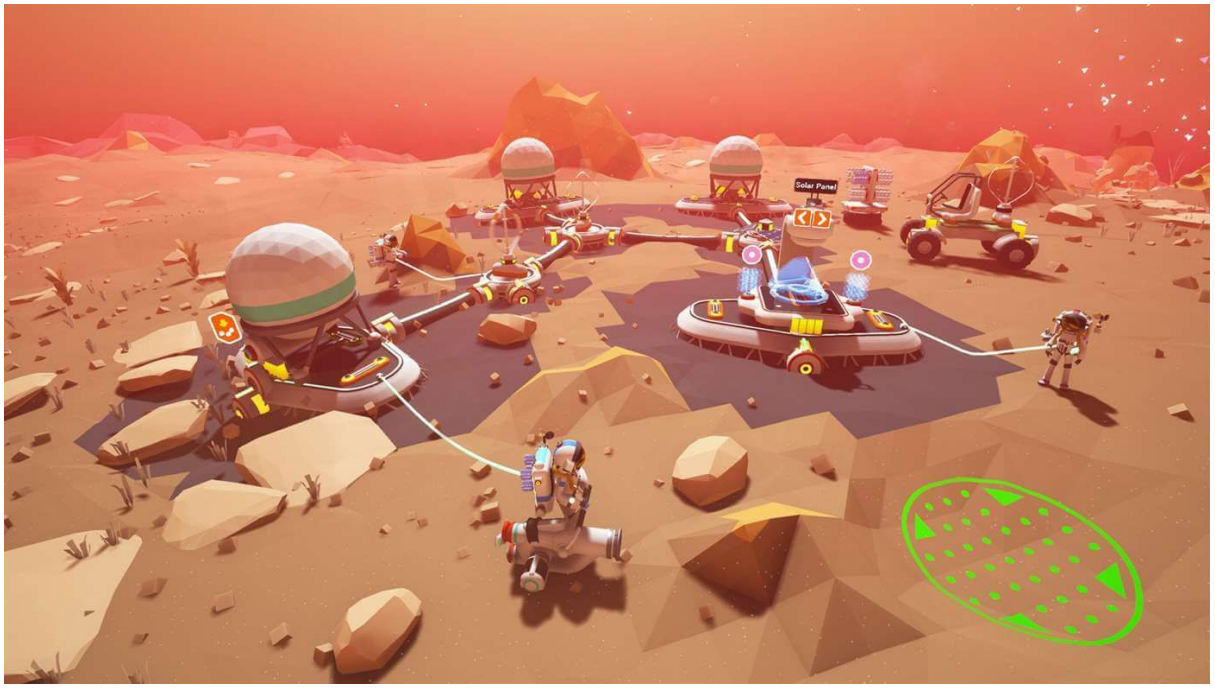
En ce qui concerne le “texturing/coloring”, chaque face des modèles aura une couleur unie. Le procédé se rapprochera donc du “vertex color” mais pour des raisons d’utilisation et de maintenabilité, chaque couleur sera un pixel sur un atlas contenant toutes les couleurs du jeu.

A Forest Path, Milan Vasek

Références Graphiques

L’univers se déroule dans un futur où l’humanité est sur le point de disparaître et où la nature à repris ses droits.
La végétation y est donc abondante, et la technologie rétro-futuriste.

Horizon Zero Dawn, Guerilla Games, 2017



Astroneer, System Era Softworks, 2016

Assets graphiques (low-poly)

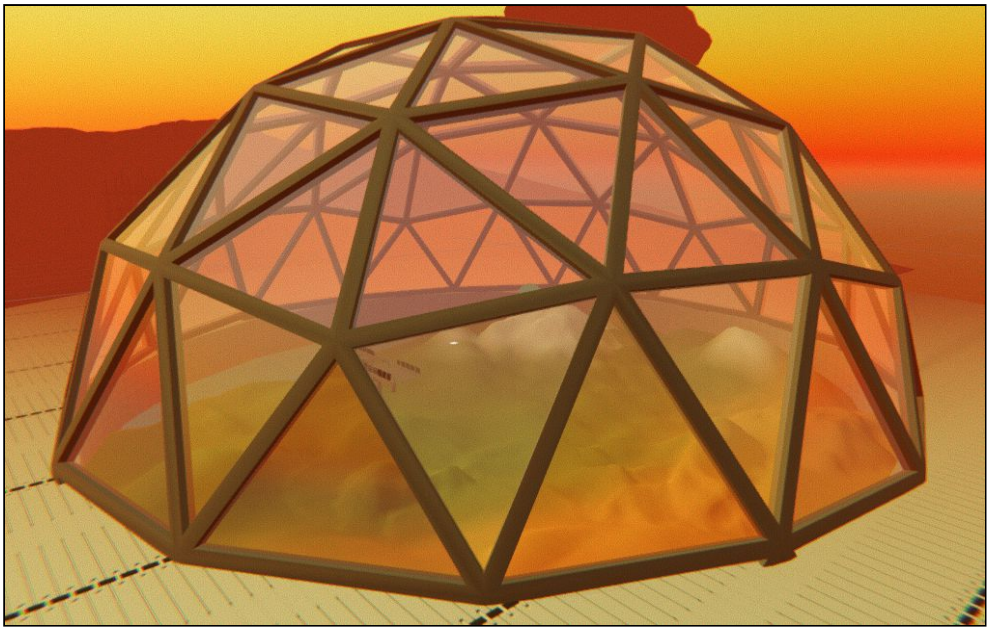
Animaux (modèles 3D et animations)

Nous avons déjà un pack d'assets contenant 10 animaux low-poly animés

Nature (modèles 3D) : végétation diversifiée, roches, skybox, rivières / lac

Ruines (modèles 3D) : bloc modulaire de bâtiments en ruines, escaliers, poste de communication avec le dôme, objects d'ambiances (caisses, pipeline, fer rouillé / ruine)

Bâtiments (modèles 3D) : bloc modulaires de bâtiments (type futuriste), champs, bâtiment d'élevage, habitations, usine type high-tech, éolienne, hydrolienne / moulin à eau, panneau solaire.



Objets (modèles 3D) : ressources, composants électroniques, épaves

Faune, Flore, Joueur :

	Basique	Avancé	Finalisé
Robot	x		
Cerf			x
Loup			x
Ours			x
Elephant			x
Giraffe			x
Manchot			x
Zebre		x	
Tigre			(asset store: 15$)
Sapin		x	
Arbre 2	x		

Arbre 3	x		
Cactus		x	
Herbe	x		
Cereales	x		

Bâtiments :

	Basique	Avancé	Finalisé
Ruines (modulaires)		x	
Dôme	x		
Murs du dôme	x		
Panneaux			

Props :

	Basique	Avancé	Finalisé
Epave voiture		x	
Ordinateurs		x	
Buche		x	
Circuits	x		
Uranium			
Déchets		x	
Viande	x		

Sound design

FX

Sons émis par les bâtiments en actifs
Son de rayon tracteur
Son rayon recycleur
Son signalant une action impossible (cible trop éloignée)
Son de déplacement du robot en mode marche et course (Herbe, Sable et Neige)

Son de saut du robot
Son d'endommagement du robot
Son des menus
Son d'entrée/sortie de l'inventaire
Son de construction des bâtiments
Son à l'activation/la désactivation des bâtiments
Son animaux
Son habitants humains (conversations incompréhensibles)

Voix

Intelligence artificielle (synthétiseur vocal)

Ambiance

Ambiance désert: Vent uniquement
Ambiance chaud: criquets, bruit de vent
Ambiance tempéré: Gazouillis d'oiseaux, bruit de vents, hululement d'animaux, bruit de branches...
Ambiance froid: Bruits de cent, sons étouffés, craquement des branches
Ambiance glaciale: Vent uniquement

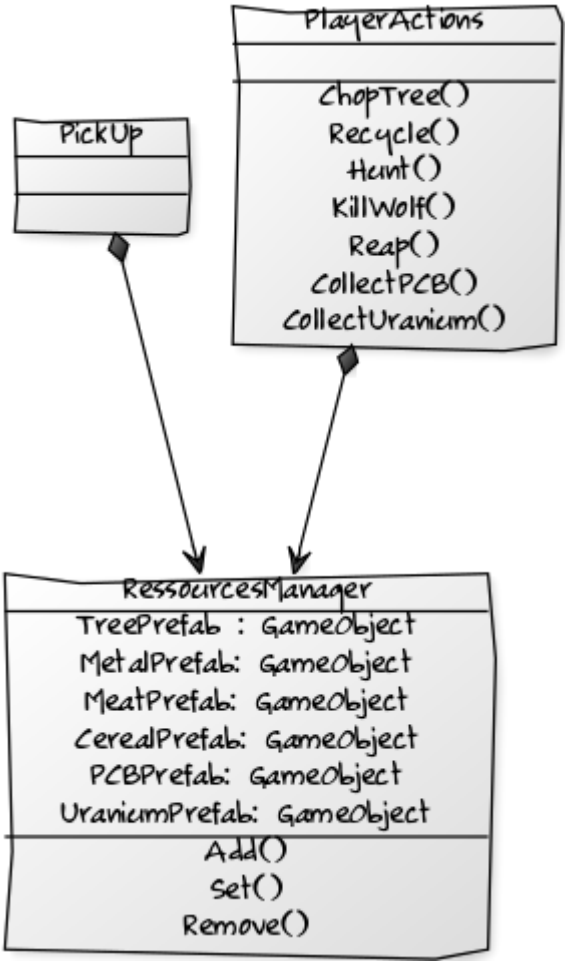
Musique

Musique d'ambiance in-game
Musique menu

Programmation

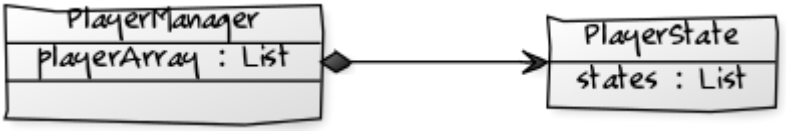
Les fonctionnalités sont séparés en domaines compartimentés afin que les programmeurs puissent progresser facilement en parallèle.

Actions du joueur



Les actions possibles du joueurs se retrouvent dans la classe *PlayerActions* attachée au Player. Lorsqu'une action menant à la récolte d'une ressource est effectuée (comme couper un arbre, chasser une biche …) , le "préfab" instancié est récupéré à partir de la classe *RessourceManager*. Cette classe comme son nom l'indique gère tout ce qui a attrait aux ressources des "préfabs" aux ajouts/suppressions. Une fois que l'objet récoltable est instancié, c'est la classe *PickUp* qui prends le relais des opérations. Cette classe se trouve sur les ressources ramassables, et elle gère leur récolte par le joueur. En effet, lorsque celui-ci récupère une ressource, elle va faire appel au *RessourceManage*r afin de mettre à jour les valeurs nécessaires.

Le changement de joueur



Lorsque le joueur découvre un autre robot et le "répare" , il peut ensuite changer de "main" robot et contrôler l'un ou l'autre à tour de rôle. Le *PlayerManager* est une classe qui gère l'ajout d'un nouveau robot comme étant jouable dans la liste PlayerArray et détecte la

volonté du joueur de changer de robot. Lorsque celui-ci fait l'action de changer, le *PlayerManager* consulte la classe PlayerState, c'est elle qui va actualiser l'état du joueur afin que celui-ci puisse changer de robot.

Construction de bâtiment

Afin de construire un bâtiment le joueur doit se mettre en mode "construction", puis il lui suffit de maintenir le clic gauche de sa souris enfoncé pour faire apparaître une prévisualisation du bâtiment sélectionné ainsi qu'un rayon lumineux allant de sa position vers la prévisualisation.

Pour faire apparaître le rayon, le script *BuildingCreator* utilise simplement un *LineRenderer* couplé avec le script *BeamEffect* (gérant un système de particule).

En ce qui concerne la prévisualisation du bâtiment et sa boite englobante, *BuildingCreator* charge grâce à Resources.LoadAll() tous les prefabs de bâtiments au démarrage du script. Puis dès que le joueur bascule en mode construction et sélectionne un bâtiment le script va automatiquement créer une copie du prefab dont il retirera tout les scripts pour ne garder que le mesh (auquel est appliqué le *Material* correspondant). Pour créer la boîte englobante un objet Bounds est utilisé avec la méthode Bounds.encapsulate, permettant de redimensionner correctement un cube créé au préalable.

Une fois la prévisualisation et sa boite englobante créées, il n'y a plus qu'à faire un Raycast pour déterminer la position de ceux-ci. Une fois la position déterminée il faut adapter les *Materials* afin de faire comprendre au joueur si la construction est possible ou non à cet endroit. En effet la construction est possible uniquement sur le Terrain (sauf eau) à un endroit dégagé et relativement plat (la valeur en y de la normale du point de collision raycast->Terrain doit être inférieure à une certaine valeur). Il faut aussi rajouter un écart de sécurité à tous les bâtiments afin d'éviter qu'un joueur ne puisse s'emprisonner en construisant des bâtiments tout autour de lui.

Lorsque le joueur relève son clic gauche en pointant un endroit apte à la construction, il ne reste donc plus qu'à instancier à nouveau le prefab avec tout ses scripts à l'endroits pointé

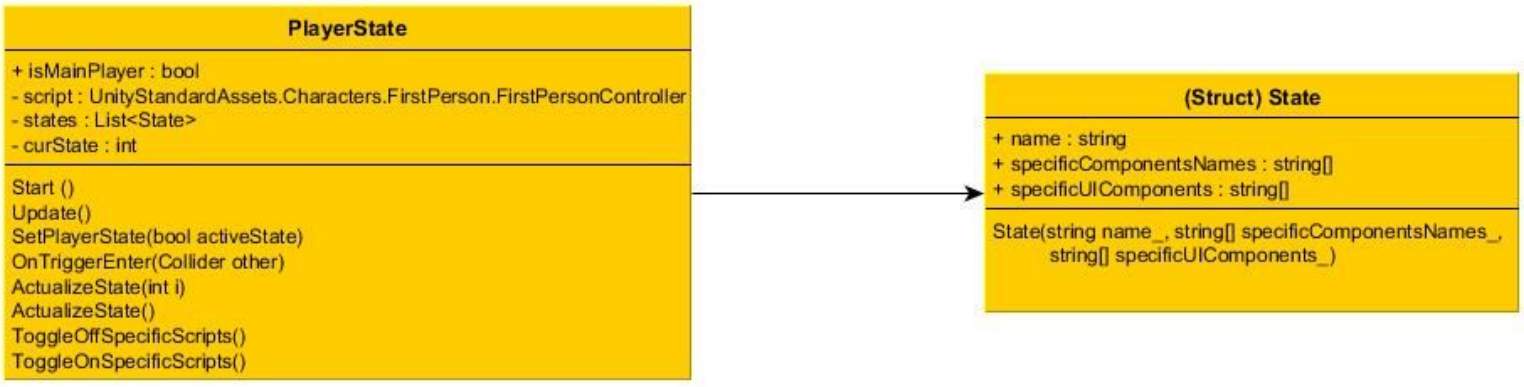
en rajoutant en dessous de celui-ci un cube faisant guise de fondations (de même dimensions que la boite englobante). Cela permet de masquer d'éventuels écarts entre le bâtiment et le sol. Le script en profite également pour appeler les scripts du préfab permettant de déduire des ressources le coût du bâtiment et met à jour la liste de *BuildingManager* stockant le nombre de chaque type de building présent sur le terrain.

Une fois instancié, chaque script du bâtiment démarre, consommant les ressources, en produisant d'autres, instanciant des déchets nucléaires, générant des nuages, causant de la pollution, etc

Rayon Tracteur

De la même manière que pour la construction d'un bâtiment le rayon tracteur du joueur est un *LineRenderer* combiné avec le script *BeamEffect*. Il suffit au joueur d'appuyer sur le clic gauche pour déclencher un *Raycast* (avec *LayerMask*) *qui* est utilisé pour déterminer la position de fin du rayon. Lorsque le rayon rentre en collision avec une cible déplaçable (animaux ou humains), le script applique un Material et appelle la méthode *Freeze()* présente sur le composant *AnimalController* de la cible afin de l'empêcher de bouger et de stopper son évolution (faim, vieillesse, etc). Dès que le joueur relâche le bouton de la souris le Material d'origine est appliqué une nouvelle fois et la méthode *Unfreeze* est appelée redonnant à la cible son comportement ordinaire.

Basculement de mode



En appuyant sur *A* le joueur peut basculer son robot d'un mode à un autre. Cette fonctionnalité est gérée dans le script *PlayerState*. Celui-ci va activer/désactiver les scripts et composants de l'UI spécifiques à chaque mode lors du basculement. Ceux-ci sont stockées

dans des structures *State*, elles mêmes stockées dans une liste. Cette structure a été choisie pour sa modularité. Il est en effet très facile grâce à cela de rajouter un mode de jeu, il suffit juste de rajouter à la liste une nouvelle structure en spécifiant ses scripts et éléments d'UI spécifiques et le reste des scripts sont déjà prêts.

Évolution des animaux & végétaux

(WildlifeManager)

Les animaux et végétaux sont gérés par un manager qui les génère en début de partie et s'occupe de leur évolution.
La génération se fait à partir de "seeds" : quelques animaux / végétaux placés sur le terrain. L'algorithme place d'autres entités sur le terrain, à une certaine distance des entités "seeds", jusqu'à atteindre la quantité spécifié.
L'évolution a lieu à chaque frame, mais seulement sur une quantité fixe d'entités pour ne pas perdre trop de performances (le coût reste ainsi constant). L'évolution consiste en un appel à la fonction UpdateState() du script LifeController présent sur chaque entité vivante.
À chaque évolution, l'entité grandit et passe un test de continuité : elle continue à vivre si le test passe, sinon elle meurt. Le test se base sur l'âge de l'entité et sur les conditions de température et d'humidité. L'entité va aussi tenter de se reproduire si les conditions le permettent (contrainte de population maximale).
Les effectifs de populations sont déterminés par un rapport entre prédateur et proie, par exemple pour 1000 arbres il ne peut y avoir que 100 cerfs, et pour 100 cerfs il ne peut y avoir que 10 loups.

Comportement des animaux

Les animaux se déplacent continuellement vers des destinations semi-aléatoires, autour d'un point d'intérêt (tanière / maison). Les prédateurs qui détectent une proie à proximité se déplacent vers elle à la place. En arrivant au contact d'une proie, le prédateur déclenche sa mort. Régulièrement les animaux font des pauses (idle) de quelques secondes.
Pour optimiser les performances, les animaux sont inactifs s'ils sont trop loin du joueur (animation et mouvement désactivés).

État du dome

Le singleton DomeState gère l'état global du dôme : température, humidité en chaque point, ainsi que l'état ouvert ou fermé. Il contient des méthodes publiques permettant à tout objet d'avoir un impact sur la température et/ou d'avoir accès aux valeurs de température/humidité en chaque point.

Évolution du terrain

L'élement terrain du niveau est peint de façon dynamique pour pouvoir observer les modifications de celui-ci en direct en fonction de la température. Les textures sont renseignées dans le terrain manager et obéissent à des règles qui prennent en compte l'altitude en un point, la température locale, et l'humidité qui est calculée comme l'inverse de la distance à un point d'eau. Ces règles ont besoin d'une courbe qui nous permet de mieux contrôler les différents seuils à partir desquelles la neige ou le désert apparaît.

Comme c'est une opération coûteuse en performances nous avons dû optimiser la façon de faire. Au lieu de tout peindre d'un coup nous l'avons quadrillé et séparé en case qui s'actualisent à la suite à chaque frame. Chaque case est un carré de côté delta (par défaut : 4x4). De plus nous écrivons dans le terrainData que si c'est nécessaire de modifier la valeur déjà présente à un epsilone près, car chaque écriture invoque un "redraw" coûteux.
La "Contrôle Texture Résolution" à un impact significatif sur les performances il est nécessaire qu'elle soit le plus bas possible.
Le même procédé est utilisé pour la cartes des détailles afin que les deux actualisent la même case en même temps. Cette actualisation étant moins coûteuse que la précédente il est possible de jouir d'une plus grande résolution pour la "Details Map Résolution".
Cet élément était au début du projet vu comme un point technique décisif pour la suite du projet. Celui-ci n'est maintenant plus une menace pour les performances du jeu, bien que nous aurons surement d'autres optimisations à faire par la suite car celà reste une grosse partie de nos performances.

Améliorations :

- Améliorer le terrain manager afin que toutes les options du terrain soient directement renseignées dedans sans que l'on ai à manipuler le terrainData directement.
- Equilibrer les règles du terrain, le cycle de vie de la végétation, des animaux, afin que les dynamiques entre ces éléments soient interdépendantes et jouables.
- Affiner le comportement des animaux : combat, déplacement en meute, chasse, fuitent, migration.
- Apporter plus de visibilité aux actions que fait le joueur en ajoutant des feedbacks (visuel, snap, son).
- Gérer l'eau dans le gameplay : montée des eaux, rivières, courant, inondation, moulin à eaux.
- Niveler le terrain sous les bâtiments de façon intelligente

Gestion de projet

Level Designers :

Le jeu se déroule dans plusieurs dômes indépendants. Les levels designers pourront ainsi travailler indépendamment sur chaque dôme sans se gêner.

- sculpture des terrains dans le dôme (relief, eau, textures...) + rochers
- ruines (bâtiments à explorer)
- équilibrage / tests
- éléments de narration
- scripting d'éléments simples de gameplay
- renfort graphisme

M2 Prog :

Les fonctionnalités techniques sont séparées en domaines compartimentés afin que les programmeurs puissent progresser facilement en parallèle. Les programmeurs sont en communication constante avec des level designers chargés de tester les nouvelles fonctionnalités.
Chaque fonctionnalité est développé sur une branche du projet Git, afin de garder une version master fonctionnelle en permanence.
En début de production, l'accent sera mis sur l'élaboration d'une architecture propre d'une part, et sur le prototypage des nouvelles fonctionnalités d'autre part afin de pouvoir les tester au plus vite.

Graphistes :

Les graphistes diviseront leur temps de production entre tous les projets. Néanmoins nous avons un graphiste référent qui supervisera la production et la cohérence visuelle.

Par ailleurs, un graphiste de l'année dernière pourra allouer une partie de son temps à notre projet en télétravail (Charles Clauzel, qui a déjà élaboré les ruines et le dôme).

Outils utilisés

- GIT (Synchronisation du projet Unity)
- Google Drive (Partage de documents hors du projet)
- Trello (Organisation)
- Discord (Communication)

Communication

Plateformes

- Page facebook du projet (à créer)
- Page facebook collective : https://www.facebook.com/gamagoragames2017/
- Page itch.io http://gamagora.itch.io/

Stratégie de communication

- Devlog et screenshots à partir de début mars sur une base hebdomadaire.
- Teaser début avril.

Business model

Le but de ce business model est de rentabiliser les investissements financiers personnels (hors coût de la formation GAMAGORA).
Le jeu pourra être proposé sur la plateforme de distribution STEAM. Il serait alors proposé entre 5 et 10€, selon le nombre de features que nous auront pu intégrer au projet.

Dépenses

Achat d'assets

Pack d'animaux low poly animés : 20€

Licences logicielles

Unity: 0€ tant que nous ne dépassons pas 100 000 $ de revenu

Communication

15 Tasses personnalisées: 150€
15 Sweat Shirts personnalisés: 350€

Distribution

Steam direct: 85€ (100$)

Total

605€

Revenus

Prix de vente à 5€

TVA (20%): -1€
Taxe Steam (30%): -1.5€
⇒ Revenu: 2.5€ par vente
Unity reste gratuit en dessous de 340 000 unités vendues
243 unités à vendre pour atteindre la rentabilité

Prix de vente à 10€

TVA(20%): -2€
Taxe Steam (30%): -3€
⇒ Revenu: 5€ par vente
Unity reste gratuit en-dessous de 17 000 unités vendues
122 unités à vendre pour atteindre la rentabilité

Annexes

Ressources

Le bois

Il est récolté en coupant les arbres grâce à l'outil de récolte.

Graines

Elles sont obtenues en coupant les arbres grâce à l'outil de récolte.

La viande

Elle est obtenue en chassant des animaux à l'aide de l'outil de récolte ou grâce aux bâtiments d'élevage.

Les céréales

Elles sont récoltées dans l'environnement à l'aide de l'outil de récolte ou obtenues grâce aux plantations.

Le métal

Il est obtenu en recyclant les épaves trouvées dans l'environnement à l'aide de l'outil de récolte. Sa quantité est limitée.

L'uranium

Il est obtenu en recyclant les épaves trouvées dans les ruines. Sa quantité est limité mais son usage est presque illimité.

Les circuits imprimés

Ils sont obtenus en explorant les ruines présentes dans l'environnement. Ils sont rares et leur nombre est limité.

L'énergie

Elle est obtenue grâce aux centrales d'énergie. Elle est utilisée de façon continue par certains bâtiments

Bâtiments

Habitations

Habitation de base (Coût à la création: 10 bois et 1 métal)

Taille: 6m de haut, 5m de largeur [Immeuble à 2 étages (4 appartements)]
Produit 4 places pour les habitants si elle reçoit assez d'électricité de façon continue (12 unités d'électricité).
Chaque habitant présent a besoin de 3 céréales et 1 viandes toutes les 120 secondes. Sinon, il part du village.

Habitation améliorée (Coût à la création: 5 bois et 5 métal)

Taille: 10m de haut, 10m de largeur [immeuble à 3 étages (12 appartements)]
Produit 12 places pour les habitants si elle reçoit assez d'électricité de façon continue (24 unités d'électricité).
Chaque habitant présent a besoin de 3 céréales et 1 viandes toutes les 120 secondes. Sinon, il part du village.

Production d'énergie

Générateur à bois (Coût à la création: 10 bois et 1 métal)

Taille: Cube de 5m de côté
Consomme 3 bois toutes les 30 secondes pour rester actif.
Produit 30 unités d'électricité tant qu'il est actif.
Produit 10 unités de pollution (CO_2) toutes les 30 secondes tant qu'il est actif

Générateur nucléaire (Coût à la création: 5 bois, 5 métal et 1 uranium)

Taille: Demi sphère de 6m de diamètre
Rest actif indéfiniment.
Produit 100 unités d'électricité tant qu'il est actif.
Produit 1 déchet radioactif toutes les 120 secondes tant qu'il est actif.
Les déchets radioactifs détruisent toute forme de vie dans un rayon de quelques mètres.

Générateur solaire (Coût à la création: 3 bois, 3 métal et 1 circuit imprimé)

Taille: "cube" de 8m de côté
Ne consomme rien mais prend beaucoup de place.
Son activité dépend de la quantité de soleil.
Produit entre 0 et 30 unités d'électricité.

Générateur hydroélectrique (Coût à la création: 3 bois, 10 métal)

Taille: barrage en arc de cercle de 3m de haut et 5m de long (minimum)
Réduit de 90% la taille du cours d'eau en aval.

Produit entre 0 et 200 unités d'électricité en fonction de la taille du cours d'eau.

Fermes

Parcelle (Coût à la création: 9 céréales et 3 bois)

Taille: carré de 3m de côté
Crée une zone de pousse pour 9 céréales. Seules les céréales peuvent pousser dans cette zone.
Les céréales se multiplient à l'intérieur de cette zone au lieu de pousser au hasard du dôme.
Les céréales peuvent être récoltées à la main ou grâce à un récolteur.

Ferme (Coût à la création: 10 bois)

Taille: carré de 10m de côté
Crée une zone de reproduction pour les herbivores qui sont placés dedans.
Ils ont le même cycle de vie que dans la nature mais donnent naissance à plus de petits.

Barrière (Coût à la création: 5 bois)

Se place sur une parcelle pour la protéger des animaux ou sur une ferme pour la protéger des prédateurs.

Récolteur (Coût à la création: 5 bois et 2 métal)

Taille: Carré de 6m de côté
Se place à côté d'une parcelle ou d'une ferme afin d'y récolter automatiquement la production de la façon la plus rentable possible.

Plantations

Arbre (Coût à la création: 1 bois)

Place une jeune pousse de l'arbre correspondant.

Céréale (Coût à la création: 1 céréal)

Place une jeune pousse de céréale.

Entrepôt

Taille: rectangle de 10m sur 30m
Il permet de stocker les ressources.
Il permet les échanges de ressources entre les différents bâtiments.

Etat de l’avancement du Projet

Partie Technique	Avancement
Gestion du joueur : (déplacement, inventaire, système de combat, actions contextuelles, interaction avec les ressources).	Prototypé à ≈ 75%, reste : interaction dome, sauvegarde
Gestion du Jeu : condition de victoire et défaite, action du dôme, Interaction avec l’IA (dialogues), interface/menus.	≈ 0%, Non fait.
Mécaniques liées au village : arrivée/mort des villageois, implémentation des actions des bâtiments, consommation des ressources des villageois, pollution.	Prototypé à ≈ 50%, reste : arrivé/ mort des villageois. comportement des villageois.
Simulation de l’environnement : gestion de la pollution, modifications des terrains en fonction de la température et de la zone géographique.	Prototypé à ≈ 50%, reste : optimisation, règles plus complètes
Simulation du comportement des animaux : déplacement, attaque (micro) / migration (macro), gestion des populations / reproduction, , interaction animaux/environnement.	Prototypé à ≈ 50% reste : Complexifier le comportement, améliorer navigation, animation
Implémentation des effets : bruitage, musique/ambiance, FX, animations.	Prototypé à ≈ 25%