Rapport de soutenance: Intermédiaire

Synapse

Endless Code Studio

Léo CAPMARTIN, Thomas CORBIERE
Hugo SAINT GERMES, Olivier TALANE

28 Avril 2021

Table des matières

Introduction

Grâce à une certaine avance sur le planning du multijoueur et de la physique du jeu, nous avons pu nous concentrer principalement sur la réalisation des énigmes et de l'intelligence artificielle. De plus nous avons aussi eu le temps d'améliorer la physique du jeu et l'interface utilisateur.

Nous avons maintenu une bonne dynamique de groupe que nous avions instaurée au début de notre projet. Cette dynamique nous a permis d'avancer comme nous l'entendions et ainsi être dans les temps par rapport à notre programme.

Notre rapport est basé sur 4 grands axes.

- ***Chronologie : Les tâches déjà effectuées***
- ***Avancement du site web***
- ***Comparaison avec l'objectif fixé***
- ***Le travail à venir***

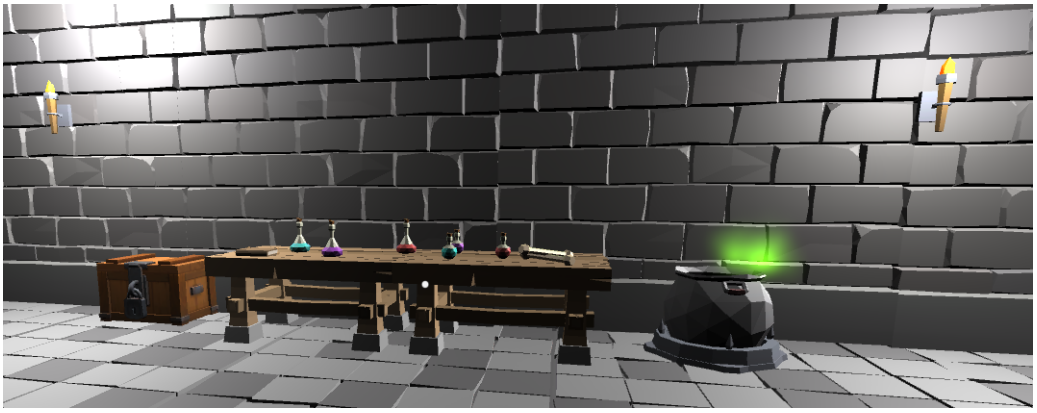
Chronologie

2.1 Les énigmes

- **L'alchimie:**

  Principe: Le but du joueur est de choisir entre différentes potions positionnées sur une table et de les mettre dans le bon ordre dans un chaudron. Si le joueur se trompe dans la réalisation de cette énigme toutes les potions seront éjectées du chaudron et il devra tout recommencer.

  Fonctionnement: Chaque potion ainsi que le chaudron possèdent le layer Interaction qui permet l'interaction entre le joueur et les GameObject (voir physique). Pour vérifier que le joueur insère les potions dans l'ordre, les potions possèdent chacune leur propre tag qui se découpe en deux parties (Exemple de tag: Potion, potion1). Cela nous permet de connaître le type de l'objet et d'avoir accès à son numéro associé. Pour renvoyer les potions du chaudron lorsque le joueur se trompe, nous utilisons simplement un RigidBody.AddForce.



- **L'échiquier:**

  Principe : Lorsque le joueur interagit avec le plateau d'échecs, il va avoir une interface 2D d'une partie d'échecs déjà commencés qui va apparaître à l'écran où il suffit de faire 2 mouvements pour mettre l'autre équipe en échec et mat. Ce plateau de jeu se génère au début de la partie. Si le joueur ne fait pas le bon mouvement, la pièce déplacée revient à son emplacement initial. Une fois le roi adverse mit en échec et mat, l'énigme enclenche un autre évènement.

  Fonctionnement : Au lancement de la partie, un plateau de jeu est sélectionné parmi une base de données de plateau de façon aléatoire. La façon dont sont enregistrés les plateaux est assez simple, c'est un tableau de 9 lignes et 8 colonnes. Les 8 premières lignes représentent l'emplacement des différentes pièces, par exemple "BT1" représente la pièce "tour noire n°1". La dernière ligne permet de savoir quels sont les mouvements à faire par le joueur, puis le troisième élément permet au programme de savoir quoi faire une fois que le joueur a fait le premier mouvement. Ici "WQ,0,6" veut dire que le premier mouvement que le joueur doit effectuer est le déplacement de la dame blanche sur la case (0,6), et "0,4,0,6" représente d'abord les coordonnées de la pièce à bouger (0,4), puis vers où la déplacer (0,6), donc la tour noire n°1 va manger la dame blanche.

Ici la séquence de jeu donne : Dame Blanche à côté du roi noir, puis tour noire mange la dame, puis cavalier met en échec et mat le roi en se plaçant en (1,5).

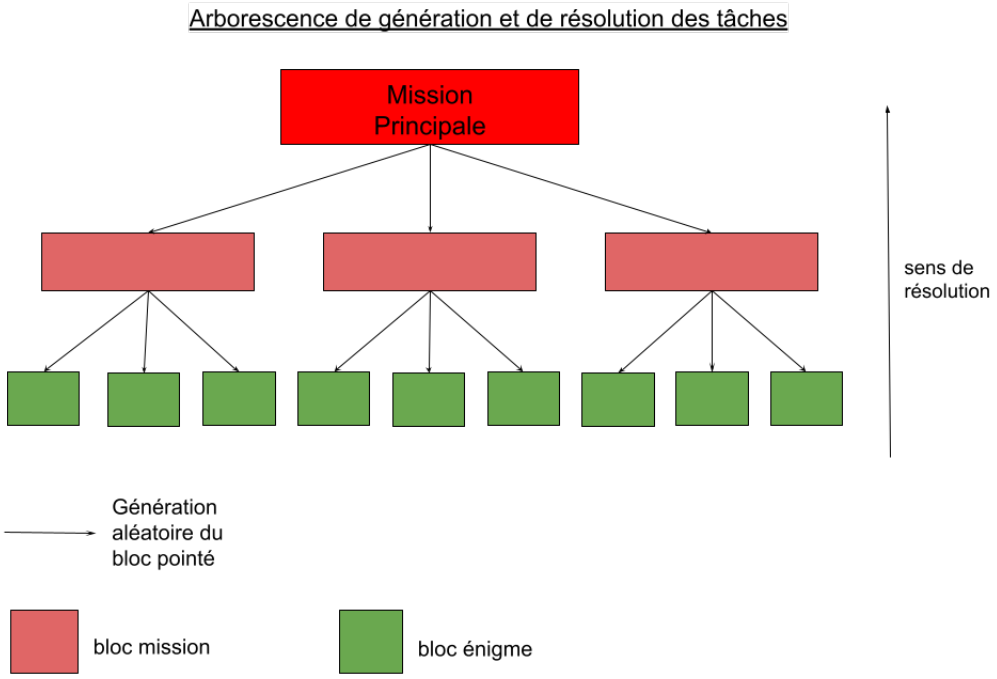
```
private string[,] enigmeBoard1 =
{
    {"","","","BQ","BT1","","","BR" },
    {"BP1","BF1","BP3","BP4","","","BP5","BP6" },
    {"","BP2","","","","","","WC1" },
    {"","","","WQ","","","BF2","" },
    {"","","WP1","","WP2","","","" },
    {"WP3","","","","","","","" },
    {"","WP4","WP5","","","WP6","WP7","WP8" },
    {"WT1","","","","","WT2","WR","" },
    {"WQ,0,6", "WC1,1,5", "0,4,0,6", "", "", "", "", ""},
};

void Start()
{
    board.Create();
    pieceManager.Setup(board, enigmeBoard1);
}
```

2.2 L'intelligence artificielle

2.2.1 Principe

L'intelligence artificielle que nous avons décidé d'implémenter est un algorithme qui générera une salle d'énigmes aléatoires mais ordonnées. En effet, pour que notre jeu ne soit pas qu'une suite d'énigmes à résoudre dans n'importe quel ordre le plus rapidement possible, l'IA fera en sorte que les énigmes et les missions aient un lien entre elles. Ainsi, par exemple, la mission X ne pourra être résolue uniquement si les énigmes Y et Z auront déjà été complétées.
En suivant ce principe, on peut facilement implémenter ce système sous la forme d'arbre général, les noeuds internes étant les missions et les noeuds externes les énigmes. Il devient donc possible de dresser un schéma théorique de l'arborescence des énigmes et missions.



Ici la mission principale est la dernière à être résolue alors que les énigmes seront les premières.

2.2.2 Fonctionnement

Les énigmes sont des composants du jeu pouvant être résolues à tout moment. Elles constituent généralement un casse-tête pouvant prendre différentes formes (Ex : trouver le bon mouvement à faire sur un plateau d'échec/trouver le bon objet à déplacer dans la pièce). Une fois résolue, l'énigme donne accès à un indice permettant de résoudre une mission interne.
Les missions sont des composants qui peuvent être résolus uniquement si tous les indices la concernant ont été récoltés. Une fois résolue, une mission peut agir de deux manières différentes en fonction de son importance.

- **Mission Principale :** Lorsque la mission est résolue, elle met fin à la partie. Le premier joueur à la résoudre gagne.

- **Mission Interne :** Lorsque la mission est résolue, elle donne accès à un indice pour résoudre la mission principale.

2.2.3 Implémentation

Pour implémenter notre IA, Léo a travaillé sur un projet séparé.
Cependant, celle-ci n'est pas encore dans le projet Synapse car nous n'avons pas encore assez d'énigme à fournir pour l'implémenter correctement.

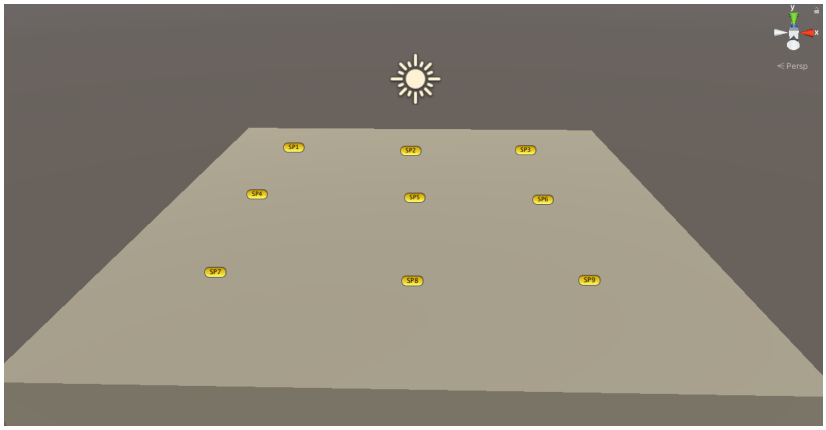
Nous avons dû créer deux nouvelles classe dans notre code permettant d'implémenter l'IA correctement.

- Tout d'abord, la classe Enigma. Cette classe est abstraite et hérite de MonoBehaviour, ce qui lui permet d'hériter des méthodes déjà présentes dans Unity. Toutes les énigmes et missions devront posséder un script qui hérite de Enigma.

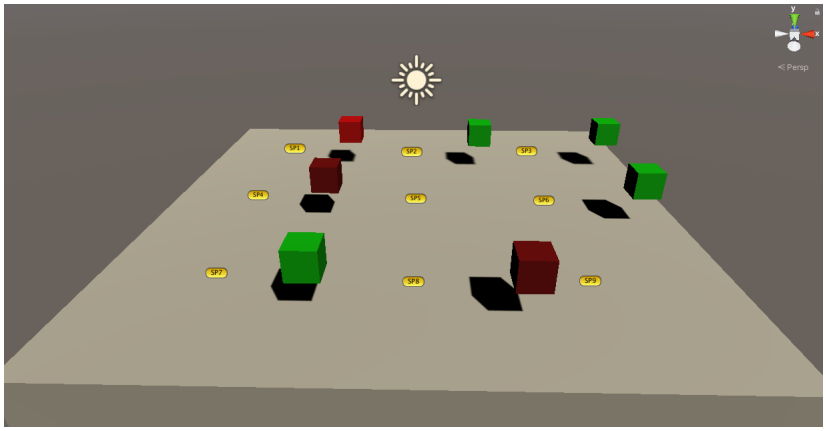
- Nous avons aussi créé la classe GeneralTree qui comme son nom l'indique est un arbre général. Cette classe à deux attributs : root, la racine du type GameObject et children, une liste de GeneralTree, contenant tous les arbres généraux enfant de enfant de ce noeud.

Pour fonctionner correctement, notre IA va d'abords créer récursivement un arbre général en piochant une énigme ou une mission (en fonction d'où elle en est dans la génération) dans les énigmes et missions disponibles.
Une fois l'arbre généré, elle s'occupe de placer dans la scène chacune des énigmes et des missions au bon endroit. Pour ce faire, chaque énigme et mission possèdent une liste de points d'apparition dans laquelle l'IA va piocher pour faire apparaître l'énigme ou la mission.

Ici la scène est avant la génération des énigmes par l'IA.
En jaune les différents points d'apparitions pour les énigmes et les missions.



Ici la scène est après la génération des énigmes par l'IA.
En jaune les différents points d'apparitions pour les énigmes et les missions, en vert les énigmes, à Bordeaux les missions, en rouge la mission principale.



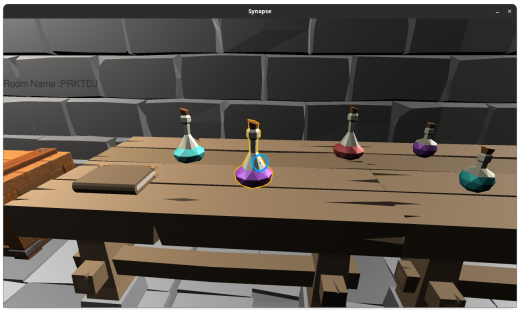
2.3 La physique

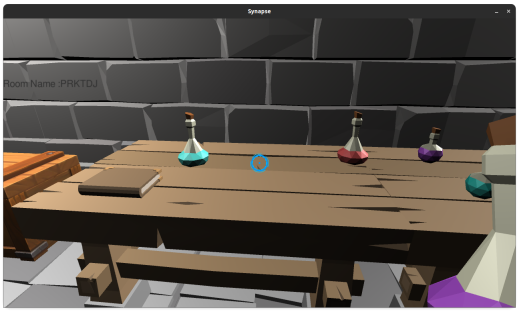
La physique du jeu étant déjà bien implémentée et avancée, nous n'avons plus que l'interaction avec les objets à ajouter.
Pour interagir avec un objet le joueur devra le viser et être à la bonne distance de l'objet puis appuyer sur la touche E. Pour vérifier ces trois conditions nous avons utilisé une méthode d'Unity : le Raycast. Cette méthode envoie un rayon (invisible aux yeux du joueur) à partir d'un point donné, dans une direction donnée et sur une longueur donnée. Une fois le

rayon envoyé, on peut récupérer dans une variable toutes les informations le concernant, notamment si il a rencontré un objet et si oui lequel. Il suffit alors de vérifier si on peut interagir avec cet objet. Si c'est le cas et que le joueur appuie sur la touche E alors l'interaction se lance.

Grâce à ce système, il devient possible d'ajouter n'importe quel type d'interaction. nous avons pour le moment ajouté 3 interactions :

- Il est possible d'attraper un objet et de le poser.





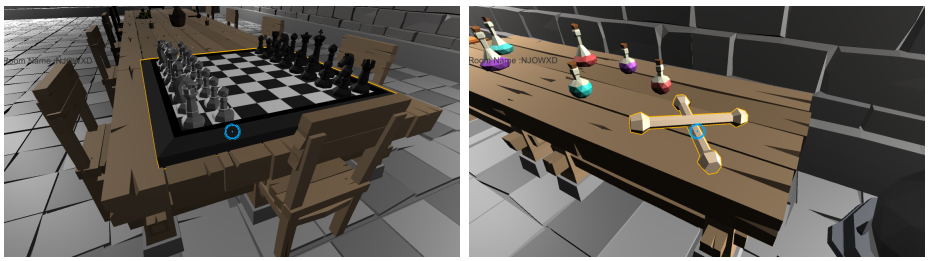
- En visant le chaudron et en tenant une potion, il est possible de la placer à l'intérieur.
- En visant le plateau d'échec, il est possible de lancer l'énigme des échecs.

2.4 Le game design

2.4.1 Interface utilisateur

Afin d'améliorer l'expérience de jeu nous avons décidé de rajouter un curseur animé aux joueurs afin qu'ils puissent plus précisément interagir avec l'environnement. Grâce à l'aide de l'asset QuickOutline, des même raycast utilisés dans la physique du jeu et d'un tag permettant de savoir si le joueur peut interagir avec l'objet, nous avons fait en sorte que lorsqu'un joueur regarde un objet avec lequel il peut interagir celui-ci se surligne. Cependant ajouter cette fonctionnalité nous a posé beaucoup de difficulté notamment Le fait que le raycast passe à travers les murs. Le surlignement des objets se voyait donc à travers les murs. De plus au début nous

utilisions une autre méthode qui consistait à changer le matériel, de l'objet. Cependant avec celle-ci tous les objets étaient surlignés dès le début de la partie et il fallait les pointer au moins une fois chacun pour que cela fonctionne. Tous ces problèmes nous ont amené à utiliser l'asset QuickOutline.
Voici le résultat final:



2.4.2 Implémentation des énigmes

Afin d'implémenter le jeu d'échecs a notre map, nous avions premièrement décidé de charger une nouvelle scène puis de recharger l'ancienne avec la map une fois l'énigme finie. Cependant, lorsqu'un joueur lançait l'énigme, celle-ci s'activait pour les deux joueurs. Pour résoudre ce problème, nous avons à la place créé un canvas pour chaque joueur avec le jeu d'échecs et lorsque le joueur décide d'interagir avec le plateau d'échecs nous faisons apparaître le canvas qui correspond. Ce canvas disparaît une fois que le joueur a réussi l'énigme. Pour éviter que le joueur puisse réaliser l'énigme à l'infini nous avons ajouté au joueur un booléen qui passe à vrai lorsqu'il a réussi l'énigme et ainsi l'empêche de la refaire.

La flèche rouge désigne le premier mouvement que doit faire le joueur et celle en bleu le deuxième.

Site Web

https://synapsesite.github.io/

3.1 Outil utilisé

C'est Thomas qui s'est chargé de la mise en place du site.

HUGO

HUGO est un framework open source qui a pour but de faciliter la création de site web static.

Pourquoi avoir utilisé ce logiciel ?

HUGO nous a semblé rapidement extrêmement intéressant. En effet, ce framework a très peu de liens avec des alternatives telles que WordPress ou bien Wix car il demande de véritables notions de code (HTML5, CSS3 et javascript) pour réaliser le site web.

Une fonctionnalité d'HUGO qui nous a fortement intéressés était la gestion du "responsive" qui est très facilité. De plus, travailler avec HUGO est très agréable car lorsque nous travaillons en local pour construire le site, nous pouvons le mettre dans un des serveurs d'HUGO qui permet de voir en temps réel sans même recharger la page l'avancement de notre projet.

3.2 Mise en page

Notre site est constitué de quatre fenêtres différentes:

- La première fenêtre, qui est aussi la principale, sera une courte description de notre jeu permettant à l'utilisateur de comprendre qui est le groupe et quel est le type de jeu. De plus, on y trouvera un bouton pour télécharger le jeu sous windows ou sous linux.

- La seconde fenêtre expliquera le fonctionnement détaillé de notre jeu, tel que les règles du jeu et le fonctionnement détaillé de la partie.

- La troisième fenêtre sera un onglet actualité, où nous mettrons toutes les dernières fonctionnalités ajoutées et toutes les informations diverses sur l'avancée de notre jeu.

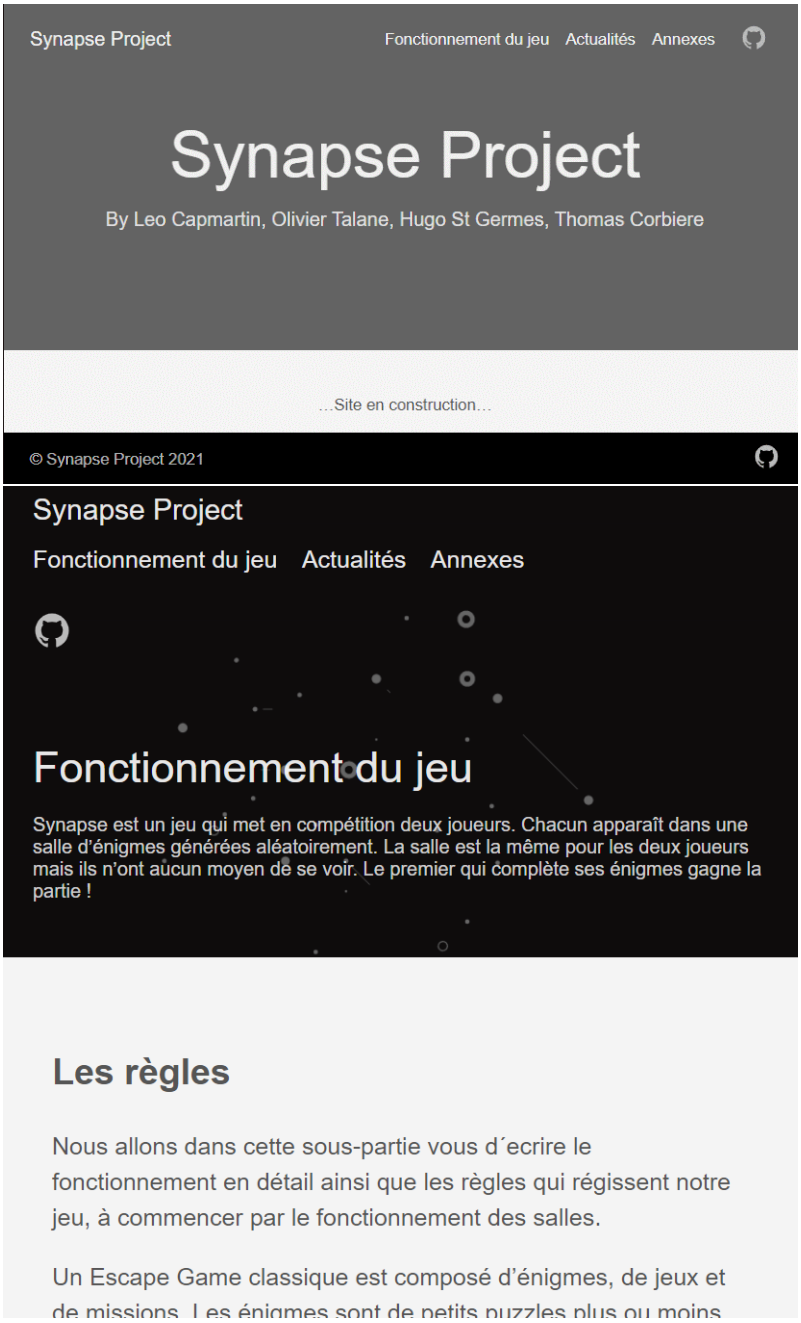
- La quatrième fenêtre regroupera tous les liens qui peuvent être utile au joueur tel qu'un lien vers notre GitHub, et avoir accès à tous nos rapports de soutenance et cahier des charges.

Chaque fenêtre possède les mêmes caractéristiques:

- **l'entête :** qui prend presque toute la place disponible sur l'écran de l'utilisateur. Il est constitué du titre de la fenêtre actuellement affiché, d'une barre de navigation permettant de se déplacer entre différentes fenêtres. Chaque fenêtre possède aussi une courte description pour que l'utilisateur puisse se repérer facilement dans notre site.

- **Corps de la page :** Regroupe toutes les informations importantes de la page telle que du texte, des liens, et des images.

- **Bas de page :** Il s'agit d'un bandeau noir identique en bas de chaque page du site. Il contient un lien vers notre GitHub ainsi que le nom de notre jeu.



3.3 Hébergement

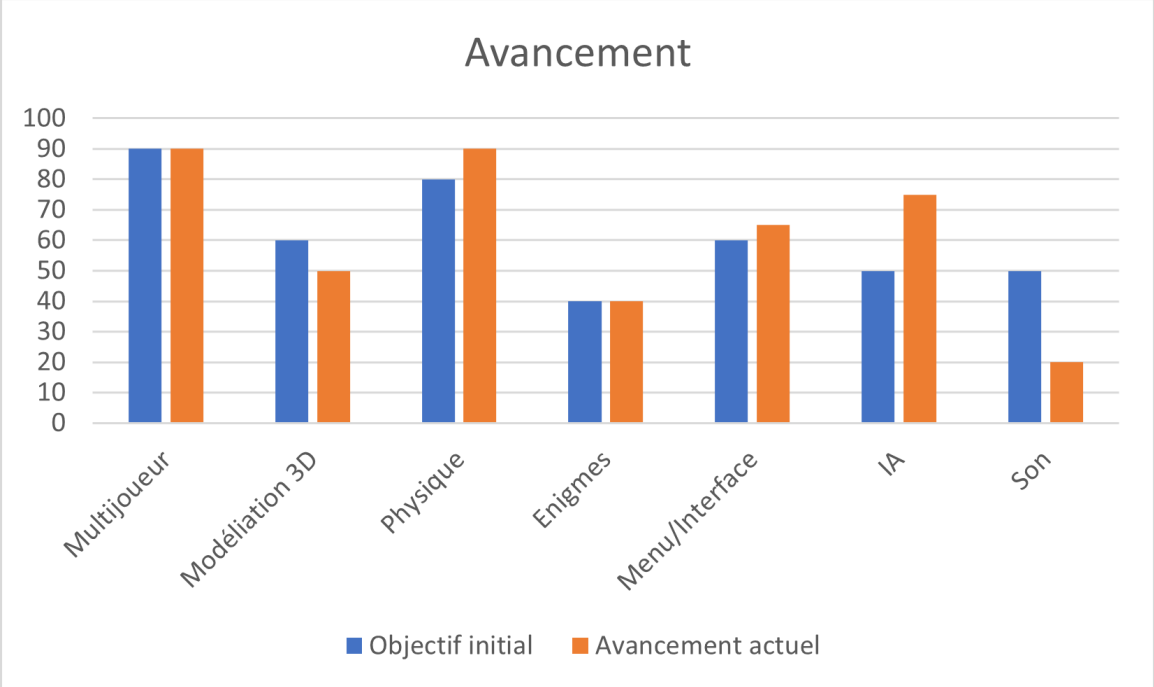
Pour l'hébergement de notre site web, nous avons décidé d'utiliser GitHub Pages, pour sa facilité d'utilisation et son tarif très intéressant puisque

gratuit.

Le fonctionnement est très simple, nous avons dû créer un nouveau compte GitHub qui aurait le nom de notre site web sinon cela aurait créé des soucis au niveau de l'URL. Par la suite, nous avions juste à build notre projet et l'insérer dans notre nouveau répertoire GitHub Pages. Par la suite, nous avons juste dû faire quelques manipulations pour modifier les paramètres de notre site.

Avancement des tâches

Grâce à l'avance importante que nous avions sur le multijoueur et la physique nous avons pu nous concentrer sur la réalisation des premières énigmes et de l'intelligence artificielle qui ont bien avancé depuis la dernière soutenance. De plus nous avons aussi eu le temps d'améliorer notre physique en rajoutant les interactions entre le joueur et les objets. Cependant nous ne nous sommes pas concentré sur la partie sonore comme nous l'imaginions.



Tâches à venir

5.1 Site web:

Pour la prochaine soutenance, le site sera évidemment terminé. Pour cela, nous allons devoir compléter chaque page avec les informations manquantes, nous allons aussi rajouter des images et des logos pour le rendre le site plus vivant et plus agréable à consulter. Pour finir, nous aimerons si nous avons le temps d'ajouter un webGL à notre site pour que le joueur puisse avoir un aperçu de notre jeu depuis son navigateur. De ce qui est hébergement, nous allons rester sur GitHub Pages puisqu'il s'agit d'une solution gratuite qui répond parfaitement à nos attentes.

5.2 Les énigmes et mission:

Malgré l'avancement de cette partie, le travail a réalisé dessus reste très important. Il se peut que tous les membres participent à celle-ci lorsque leurs autres tâches auront été réalisé.
Pour la prochaine soutenance, nous allons implémenter les énigmes et les missions suivantes:

- **Calcul mental :** Trois calculs simples seron posés au joueur. Les réponses pour chacun des calculs seront demandées sous forme de QCM. Le joueur aura 6 secondes par question. S'il se trompe, tout est remis à 0. Les calculs seront déterminés de manière aléatoire.
- **Tableaux à remettre à la bonne place :** Des tableaux seront placés dans la salle. Le joueur devra les remettre à la bonne place grâce à une carte indiquant leur emplacement.

- **Remettre l'heure :** Une horloge sera placée dans la salle. Le joueur devra la remettre à la bonne heure (celle-ci sera donnée par un autre objet comme le PC virtuel).
- **Éléments à ouvrir :** Un coffre, une boîte ou une porte seront à ouvrir avec un code. Les chiffres, les lettres ou les couleurs qui composeront le code seront éparpillés un peu partout dans la salle.
- **Énigmes visuelles :** Des questions de type charades ou rébus seront placés dans la pièce. Le joueur devra trouver la réponse du problème et l'écrire dans un PC virtuel.
- **Déplacement d'éléments :** Des éléments du décor seront déplaçables, ce qui permettra de révéler des objets essentiels dans la résolution des missions.
- **Messages codés :** Un message encodé en binaire, en morse ou en César, devra être utilisé pour résoudre une autre énigme (par ex: un plan de la salle pour les tableaux)
- **Puzzle :** Grâce aux énigmes résolues le joueur pourra récupérer des pièces qui lui permettront de résoudre le puzzle.
- **Interrupteurs :** Le joueur devra trouver des interrupteurs cachés à activer.
- **Boîtier électrique :** Des fils électriques devront être branchés au bon endroit pour déclencher un évènement.

5.3 L'intelligence artificielle

Nous avons déjà une intelligence artificielle fonctionnelle mais sur un projet à part. Pour la suite nous devrons donc l'implémenter dans le projet principal pour corriger au plus vite les problèmes et bugs que cela pourra générer.

5.4 Mode de jeu

Le mode multijoueur est fonctionnel cependant il nous reste le monde contre la montre en solo à implémenter ainsi qu'une base de données qui nous permettra de sauvegarder les scores obtenus.

5.5 Le son

Nous allons aussi sérieusement nous pencher sur la question du son dans le jeu afin d'améliorer l'immersion du joueur dans l'environnement et possiblement rajouter des énigmes nécessitant du son.

Conclusion

La partie physique et multijoueur sont maintenant quasiment terminés. De plus, nous avons aussi fait les énigmes clés qui pourrons-nous servir de squelette pour continuer à implémenter le reste des énigmes, et ainsi augmenter la durée de vie de notre jeu et rendre l'expérience du joueur bien plus intéressante.

Grâce à une bonne efficacité de travail et une bonne organisation au sein du groupe, nous devrons être dans les temps pour rendre le projet que nous avions imaginé.