



Time Quest 42

Soutenance Finale

Un projet réalisé par l'équipe :

Time's Masters

(Avec AISSA Sohane, DANYCAN Danae, DI-SANTO Alexandre et ROYER Julia)



Sommaire



I- Avant propos

II- Origine de nos conceptions

III- Segmentation du projet

A/ Site web

B/ Les graphismes

C/ Choix des ressources

D/ Les obstacles

E/ Implémentation des menus

- Menu principal
- Menu pause (avec la sauvegarde)
- Menu inventaire
- Menu d'achat

F/ Choix des coûts de chaque évolutions

IV-Conclusion

A/ Bilan du rendu par rapport au cahier des charges

B/ Point collectif sur les apports de ce projet

C/ Annexes

I- Avant propos



Nous voilà arrivés au terme du délai laissé pour la création de notre jeu : Time Quest 42.

C'est avec une certaine fierté que nous vous livrons notre projet, ce projet qui nous occupe depuis plusieurs mois.

Nous avons dû faire face à plusieurs contraintes/obstacles qui sur le moment nous ont paru insurmontables.

Au début de ce projet, nous étions quatre élèves découvrant le monde du supérieur, aujourd'hui, nous sommes un groupe qui a su trouver un terrain d'entente, occupant chacun une place importante, se complétant.

Ce dernier rapport retrace plus de 20 semaines de travail.

Avant d'expliciter nos choix et de retracer notre chemin jusqu'à cette livraison, nous vous proposons une remise en contexte :

Qu'est-ce que Time Quest 42 ?

Time Quest 42 est un jeu de plateau, le but étant de renverser la château adverse situé à son opposé sur le plateau.

Mais attention !

Ce n'est pas chose facile, il va falloir braver les obstacles au risque de perdre la vie, user de stratégies, et surtout appeler des alliés pour arriver à ce but.

Mais pourquoi ce type de jeu ?

Ce choix a été influencé par deux jeux : "The Battle Of Polytopia" et "Civilization", mais c'est surtout un choix stratégique :

Nous ne voulions pas faire un jeu de plateforme, ou une "imitation" d'un jeu connu, car beaucoup de jeux pré faits du style sont disponibles



sur internet, et ce projet était pour nous une opportunité de se dépasser, de produire un jeu à notre image, avec nos propres règles et sans être tenté de collecter des travaux en se contentant de modifier les designs.

Nous avons donc monté pièce par pièce Time Quest 42, le moindre détail a été minutieusement réfléchi, nous avons fait des essais, plus ou moins concluants, mais au final nous avons un jeu à notre image.

Alors c'est vrai que notre projet a eu au départ un peu de mal à avancer, c'est vrai que la date butoire se rapprochait dangereusement, mais nous avons fait preuve de communication et d'entraide lors de cette réalisation.

Chacun d'entre nous a apporté quelque chose d'indispensable au groupe, nous avons fait preuve de complémentarité.

Malgré plusieurs phases de flou, qui nous ont pas mal occupées, nous avons adoré travailler sur Time Quest 42, voir le jeu prendre forme, le premier personnage bouger, le premier "Build" sur UNITY réussi, les premières actions possibles.

Au-delà de l'aspect codage, la création d'un jeu demande tellement de ressources supplémentaires, il faut penser à tout, bien choisir la musique, les personnages, la bonne couleur de case...



II- Origine de nos conceptions

Nous n'avons jamais explicité chacun de nos choix, que ce soit la décision du titre, des couleurs utilisées sur notre site, le choix du nom du groupe ou encore des personnages.

Alors voici l'origine de nos décisions les plus importantes :

Times Quest 42

Le times Quest ("la Quête du temps"), fait référence dans un premier temps à la notion de temporalité dans le jeu, en effet, sur le plateau nous retrouvons plusieurs indicateurs de différentes époques :

- Le médiéval, le passé donc représenté par les personnages (soldats, cavaliers, rois) ainsi que par les châteaux.
- Le présent lui est représenté par les bouteilles de coca.
- Le futur est caractérisé par la magie du roi.

Le 42 fait référence au nombre d'heures au royaume des Times Masters, le but est donc de faire tomber le château adverse en une journée. 42 a été choisi par les maîtres du temps car il est considéré comme un nombre magique.



Mise en page du site web

L'image de fond est une image libre de droits que nous avons légèrement retouchée.

Cette image nous inspire un mode médiéval, un endroit en forêt, au calme, certainement une forêt du plateau.

Nous avons foncé le vert de l'image afin de procurer une sensation d'apaisement, de fraîcheur.

Cette image colle totalement à Time Quest 42.

Times Masters

A part la date de rendu du projet, nous nous sommes placés en maître du temps, nous avons inventé une temporalité en changeant le nombre d'heures par jour.

Choix des personnages

N'ayant aucune compétence en pixel art, nous nous sommes tournés vers des banques de données pour trouver les personnages qui prendront place sur notre plateau.

Ce fut une longue et rude course à travers les différentes banques de données.

Pour différencier les deux équipes nous avons utilisé GIMP pour colorer nos personnages (une équipe sera bleue, l'autre rouge).



Donc voici notre soldat, premier personnage d'une équipe.
Il est brave, vaillant, jeune, mais pas très expérimenté...
Il ne sait pas encore manier l'épée, il débute.



Le cavalier lui est un soldat arrivé à l'âge adulte.
Il possède un cheval, peut donc aller plus vite et être plus précis.



Le roi est le personnage le plus fort, il est capable d'user de magie pour transformer des troupes ennemis en alliées !

III- Segmentation du projet

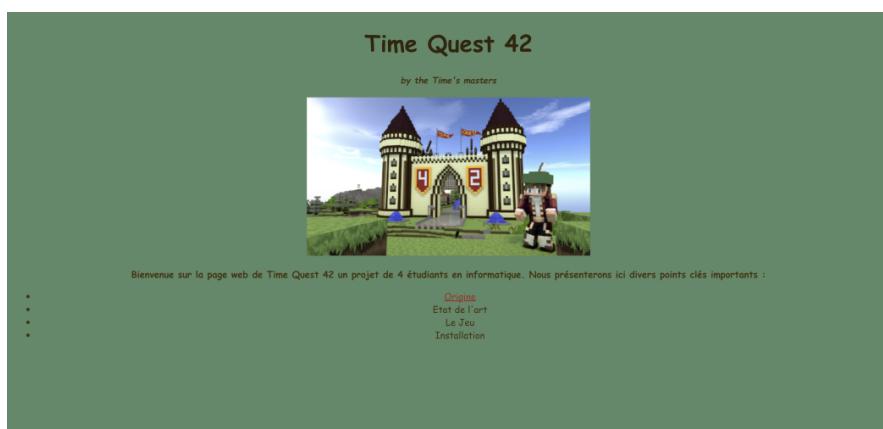


A) Le Site Web

1. La découverte de l'html et du css

Ce n'est que pour ce projet que nous avons découvert l'utilisation de l'html et du css pour la création d'une page web. Nous avons commencé par faire cela car en effet c'était plutôt gratifiant d'arriver à créer un site tous ensemble en ayant suivi un cours sur OpenClassroom un site qui propose des formations dans différents domaines en ligne dont certaines sont gratuites.

Ce cours nous a en effet donné envie d'aller plus loin en découvrant comment passer d'un site web de présentation comme ce que nous avons fait pour ce projet à une page web dynamique avec du php ou du javascript permettant l'interaction et l'adaptation de la page avec son utilisateur.



Bienvenue sur la page web de Time Quest 42 un projet de 4 étudiants en informatique. Nous présenterons ici divers points clés importants :

[Origine](#)
[Etat de l'art](#)
[Le Jeu](#)
[Installation](#)

(Site web avant la première soutenance)



L'html classe les données selon leur type, leur place sur la page, si c'est une entête, si c'est un pied de page, une bordure ou bien si cela fait partie du corps de la page. L'html est capable d'accorder des niveaux d'importances, utile pour les algorithmes qui conseillent des pages en fonction de certains mots clés, les mots importants sont mis en avant lorsque l'algorithme cherche à savoir la pertinence d'un contenu par rapport à ce que recherche l'internaute.

L'html permet aussi d'accorder des types de contenus à ses textes : listes, titres, paragraphes, liens, questionnaires ou autres.

Le css nous a lui plus permis de gérer l'aspect esthétique de la page web, on a utilisé des systèmes de blocs pour réaliser l'en-tête, le css nous permet de choisir si l'on veut que les divisions d'un bloc préconstruit s'empilent les uns sur les autres ou bien les uns à côté des autres.

Il nous a permis de choisir, la police, sa couleur, sa taille et ses effets. Le css permet aussi de dessiner les contours des blocs et de choisir l'aspect d'un fond.

Il y a plein de possibilités très amusantes et très faciles à découvrir pour créer des pages web grâce au html et au css.



2. La construction des différentes sections



(Site web à ce jour)

En ayant commencé par la création du site web cela nous a permis de pouvoir l'améliorer, le compléter au fur et à mesure de la réalisation du projet.

Nous avons commencé par la création de la page d'accueil sur laquelle nous testions les fonctionnalités présentées sur OpenClassroom, puis avons créé les pages annexes en rapport avec notre cahier des charges comme la partie : Origine, Le jeu, ou bien encore l'état de l'art.

Cela nous a permis de comprendre comment organiser ses dossiers pour appeler les différentes pages web commandées par d'autres fichiers html et css et sans trop travailler sur le contenu car il était déjà présent sur le cahier des charges.



Tout ce que nous venons de citer a été fait pour la première soutenance. Pour la seconde nous avons priorisé l'avancement du projet en lui-même qui est quand même plus important, mais nous avons tout de même rajouté les sections demandées en consignes que nous n'avions pas encore ajoutées.

Comme la section "Tutoriel" ou l'on peut voir comment l'on joue au jeu grâce à une vidéo commentée du jeu.

La section "L'avancement" décrit ce que nous avons fait jusqu'à présent, mais aussi les problèmes rencontrés.

Il y a également la section "Installation" où nous avons commencé à réfléchir à la manière d'installer un fichier ou un dossier depuis une page web.

Nous avons également modifié les sections qui nécessitaient du changement notamment "Le Jeu" car ce dernier avait évolué dû aux changements de quelques règles.

Le site a été mis en ligne pour la seconde soutenance grâce à github, un hébergeur gratuit.

Pour cette troisième soutenance nous avons complété ce qu'il restait à faire pour permettre une installation fiable du jeu et des documents allant avec : cahier des charges, manuel d'installation et manuel d'utilisation, rapports de soutenances.



Nous avons pour finir, remplacé la première vidéo tuto par une vidéo de déroulement d'une partie qui montre comment jouer à Time Quest 42.

De plus, pour la dernière soutenance, nous avons fait en sorte que le logo du jeu apparaisse dans les onglets du navigateur de recherche.

B) Les Graphismes

Dans un jeu les graphismes sont très importants. En effet, ils font partie de l'immersion du joueur et d'appréciation du jeu. Ce n'est pas forcément la technicité des graphismes le plus important mais que l'ensemble forme un tout, une ambiance bien particulière.

Le moindre objet n'allant pas avec ce tout peut ruiner l'ambiance, l'univers d'un jeu.

Le jeu aurait été moins immersif/plaisant si les différents éléments graphiques n'étaient pas cohérent ensemble.

C'est pourquoi, nous avons pris soin à ce que TimeQuest 42 soit agréable à regarder. Cela n'a pas forcément été facile à faire car dans l'équipe nous sommes tous très différents et n'avons pas le même avis sur ce qui va bien ensemble ou non.

Donc le choix de l'aspect physique de notre projet ne fut pas si facile.

1) La Carte

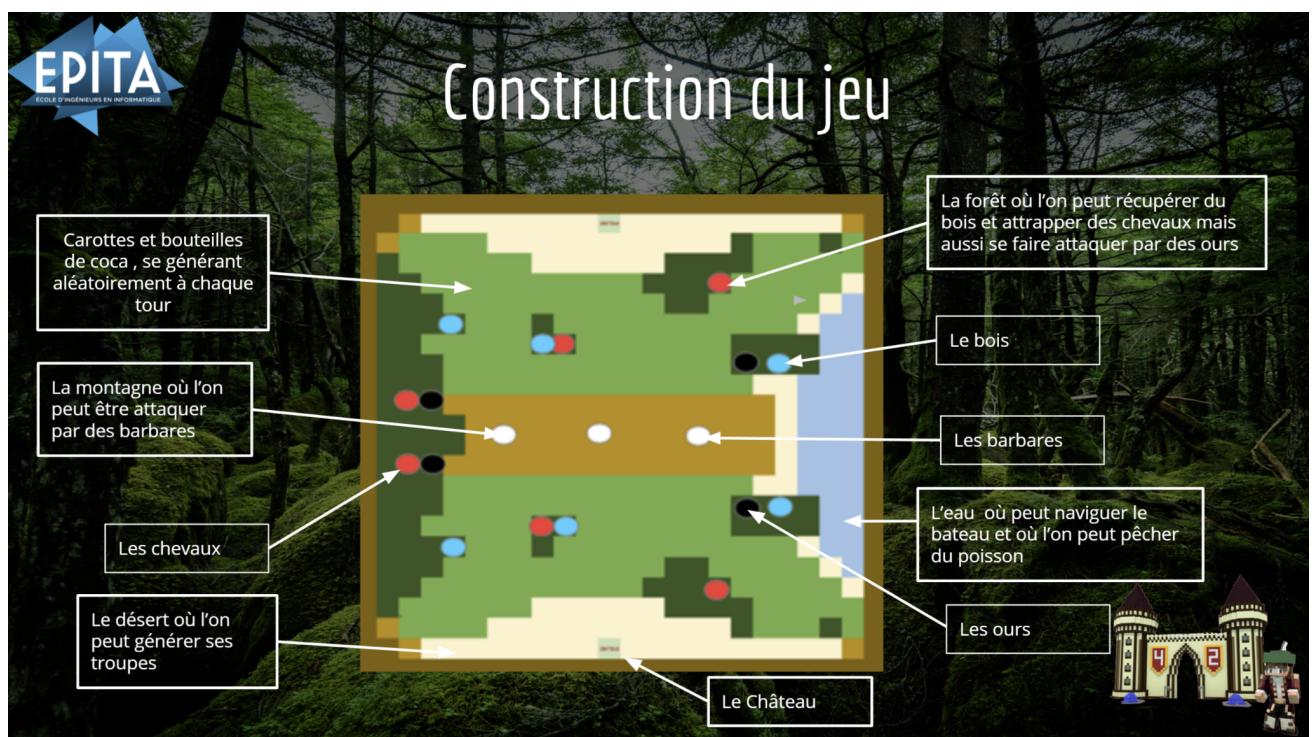
Nous avons commencé par construire le plateau de jeu en choisissant sa dimension, les zones qui le composent ainsi que la répartition sur sa surface .



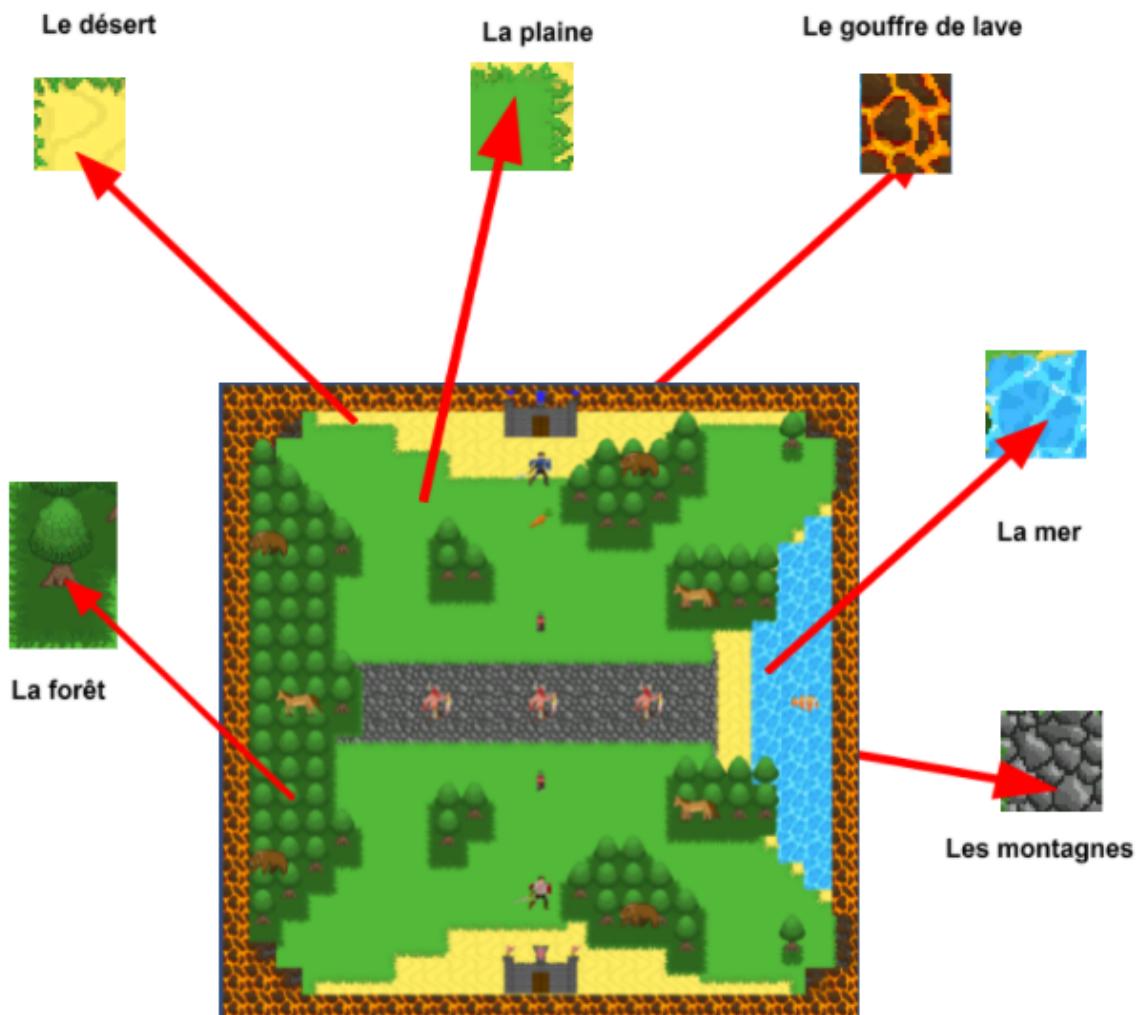
Nous voulions au début faire un plateau de $42 * 42$ tuiles, mais nous nous sommes rendus compte que les cases allaient être vraiment trop petites sur un écran.

Nous avions en effet fait une simulation sur excel avant de commencer à modéliser la carte.

Voici la dernière maquette qui a été faite avant de passer au concret :



Une fois la carte représentée, nous avons commencé à choisir des représentations pour chaque zone et avons modélisé la carte :



2) Les items

Le choix des items devait être représentatif du message que l'on voulait faire passer avec. Nous avons donc cherché plusieurs designs pour trouver celui qui correspondait le mieux à l'ambiance de TimeQuest 42.



Les designs suivants ont été trouvés sur google et notamment sur la banque de ressources de unity où certains créateurs proposent des packages gratuits.

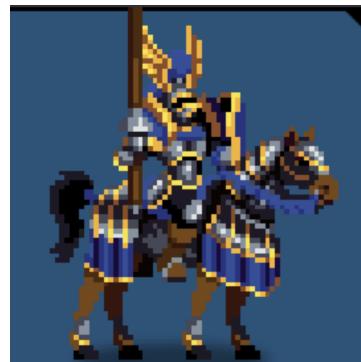
Voici les idées que nous avons eu mais que nous n'avons pas retenues pour diverses raisons : style, taille ou autres :

a) Les personnages



Nous voulions des personnages pixélisés.

Ce cavalier était très beau mais trop grand, en le ramenant à la dimension pour qu'il rentre sur le plateau il était inintelligible.



Même si notre roi a des pouvoirs magiques, nous trouvions que le roi que nous avons choisi faisait plus roi.



b) Les Intelligences artificielles



Nous voulions un ours, même si
celui-là était vraiment sympa.

Celui-ci faisait trop sage, nous voulions
un être qui fasse plus sauvage pour
le barbare.



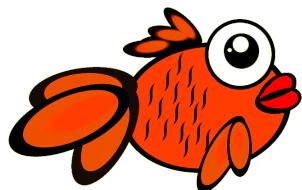
Cet ours faisait trop humain.

c) Les ressources



Ce bois était trop réaliste, nous voulions quelque chose de plus "cartoon".

Cet or faisait trop pirate,
Il tranchait avec notre univers.



Ce poisson n'était tout simplement pas au goût de toute l'équipe.

3)Les animations

Pour la conception graphique, nous avons pensé qu'il serait plus intéressant de placer quelques animations sur la carte et les personnages.

En effet le côté dynamique apporté par les animations rend moins ennuyant le jeu, donc avec l'aide des animations Time Quest 42 est beaucoup plus attractif.



Nous avons commencé par créer les animations de la carte car celle-ci était finie et les actions des personnages étaient encore en construction.

C'est l'animation de l'eau que nous avons commencé par réaliser, puis celle des crabes.



Ensuite, une fois que les fonctions des personnages étaient finies, nous avons pu nous atteler aux animations de ceux-ci, les package trouvés sur la banque de ressources Unity nous permettait de trouver des images des personnages dans plusieurs situations et de pouvoir faire nos animations.



Pour cela, quand le personnage est à l'arrêt, on fait tourner en boucles les images des "idle" qui correspondent aux personnages inactifs. Ensuite quand un personnage attaque, on appelle les images d'attaque

Ainsi nous avons des personnages qui réagissent physiquement à ce que demande le joueur. Ce dernier comprend que ce qu'il a demandé à bien était effectué et ne gaspille pas des coups de déplacement en pensant que ce qu'il a demandé n'a pas été fait .

Nous avons finalement pu associer les bandes sons aux bons moments pour donner une dimension sonore à Time Quest 42, cette dimension sonore fait partie de l'univers qui donne une ambiance au jeu et immerge les joueurs dans la partie.



C) Le Choix Des Ressources

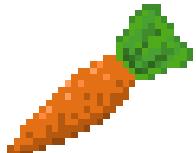
1. Les différentes ressources

Nous voulions faire un jeu de plateau où l'on démarre de quasiment rien et où pour gagner il faut utiliser les ressources du plateau pour évoluer.

Le choix des ressources était très important pour faire un jeu intéressant et permettant une grande diversité d'actions et de parties. Nous avons donc pris grand soin de nos choix pour les ressources. Il en fallait des plus précieuses que d'autres et qu'elles puissent avoir des utilisations et interactions différentes.

C'est ainsi que nous avons d'abord créé des ressources alimentaires simples permettant simplement d'être vendues pour de l'argent ou être utilisées pour nourrir nos personnages.

Ces ressources alimentaires sont :



a) Les carottes :

C'est la ressource alimentaire la plus basique de jeu, elle ne rapporte qu'un point de vie en la mangeant et 2 vies en la vendant.,

b) Le coca



C'est la deuxième ressource alimentaire la plus importante du jeu.



Elle donne 2 vies en la mangeant et 3 or en la vendant. Elle est aussi rare que la carotte : 2 unités par tour et elle apparaît aussi aléatoirement à chaque tour partout sur la carte sauf sur l'eau.



c) Le poisson

C'est la ressource la plus rare du jeu, il n'y en a qu'un seul et il n'apparaît que sur l'eau aléatoirement à chaque tour. Il n'est donc capturable qu'en bateau. Le poisson donne 3 vies en le mangeant et 4 or en le vendant.

Nous avons aussi voulu faire des ressources permettant de démultiplier les outils pour battre son adversaire. Grâce à ces ressources d'évolution le joueur peut créer de nouvelles troupes ou construire un bateau avec divers avantages.

Ces ressources d'évolutions sont :



a) Le bois

C'est la ressource la moins précieuse et la plus facile d'accès du jeu, dès que l'on est dans forêt on récolte une unité de bois.

Elle ne vaut pas grand chose : 1 or à la revente, mais son utilité permet surtout la construction du bateau.

b) Le bateau



Le bateau est un des outils les plus utiles du jeu. On peut aller sur l'eau grâce à lui, où il n'y a pas d'ennemis, il avance vite et permet une traversée plus rapide du plateau. Et on peut pêcher du poisson grâce à lui.

Il faut cependant avoir 12 bois pour pouvoir l'acheter !

b) Les chevaux



Il y a 3 zones dédiées aux chevaux sur la carte. Il y a toujours un cheval à disposition dessus. Nous avons décidé d'utiliser les chevaux pour permettre la création d'unités différentes et plus fortes que le soldat.

Les chevaux servent à la construction de cavaliers, ils peuvent aussi être vendus 9 pièces d'or. Mais pour en capturer un il faut être prêt à dépenser 2 vies.



c) L' or



L'or est la ressource de base du jeu, c'est la première ressource que possède le joueur, dès le début du jeu.

C'est aussi une ressource impossible à obtenir directement. En effet pour l'avoir il faut soit vendre des objets, soit tuer des adversaires et récupérer un peu de l'or ennemi. Mais l'or permet de créer de nouvelles troupes, un soldat pour 3 pièces, un cavalier pour 5 pièces, un roi pour 20 pièces.

Ainsi, les différentes ressources que nous avons mises en place permettent de donner de la diversité à Time Quest 42 et surtout des mécaniques permettant de rendre le jeu plus intéressant grâce à plein de stratégies et options différentes.

2. La collecte des ressources



La façon dont nous avons voulu collecter ces ressources si importantes a bien évolué tout au long de la conception du jeu.

En effet, depuis le début où nous n'avions encore jamais fait de projets dans le genre, nous pouvions seulement imaginer avec les quelques tp déjà réalisés comment nous allions procéder à la récolte des différents bonus.

Ensuite, en avançant et en reliant les différentes parties de notre projet, réalisées par différentes personnes ayant des idées différentes pour réaliser la même chose, cette idée de comment nous voulions collecter les objets a évoluée.

Lors de la première soutenance, nous voulions récolter les ressources grâce à un bouton qui apparaîtrait quand cela serait possible. Nous pensions que cela rendrait le jeu plus clair et intuitif.

Finalement, nous nous sommes rendus compte que la gestion des boutons n'était pas forcément facile et agréable, avec un plateau de plutôt grandes dimensions, où les cases étaient au final assez petites à l'écran pour que le plateau soit visible dans son intégralité, avec des boutons qui apparaîtrait près de l'objet dès que collectable cela aurait été peu pratique et ces boutons auraient donné un aspect surchargé au jeu.

C'est pourquoi dans un second temps, nous avons voulu faire apparaître un menu des actions effectuables par un même personnage lorsque celui-ci était sélectionné.



Malheureusement, même si certains de ces menus n'ont pas pu être mis de côté, nous avons préféré en limiter les boutons pour plus de clarté mais aussi plus de facilité de code de notre côté. En effet, trop de boutons pourraient rendre le jeu incompréhensible.

Finalement, nous nous sommes rendus compte que pour récupérer un objet il suffisait peut-être juste de passer dessus ! En effet, si l'on a pas envie de récolter un objet nous n'irons pas vers lui donc on ne le récoltera pas.

Mais si on le veut on fera l'inverse : nous nous dirigerons vers l'objet souhaité. Nous avons trouvé cela plutôt intuitif, plus clair et surtout plus facile à coder pour nous, grâce à des systèmes de booléens et de fonctions permettant de savoir si le joueur positionne un de ses personnages sur un objet de la map .

La récolte d'un objet coûte un déplacement, quel qu'il soit, à part le bois. Voici alors un extrait des fonctions faites pour collecter les objets :

```
//cueillette de carotte
bool saveca = Cancollectcarotte();
if (saveca)
{
    //collectcarotte = true;
    Objscript car = quelcarotte();
    car.X = -60;
    car.Y = -60;
    car.transform.position = new Vector3( x: -60, y: -60, z: 0);
    jeu.objlist.Remove(car);
    if (Team == 0)
        jeu.objequipeRouge.Add(car);
    else
        jeu.objequipeBleu.Add(car);
    saveca = false;
    Deplace --;
}
```



1) La collecte des carottes

C'est cette partie de la fonction MOOVE dans le script des personnages qui nous permet de "collecter" la carotte. Cette fonction MOOVE est appelée dans la fonction Update du jeu lorsqu'un personnage est sélectionné.

Dans cette partie de MOOVE qui est faite pour collecter la carotte, on commence par voir si on peut collecter une carotte.

Si c'est le cas, on regarde quelle est cette carotte que l'on veut collecter.

Comme on ne veut pas que la même carotte soit collectable lors du même tour alors qu'elle a déjà été ramassée.

On commence par enlever la carotte de la liste des objets appartenant au plateau. Puis, nous ajoutons l'objet à la liste des items appartenant à l'équipe du personnage qui a ramassé l'objet.
Pour finir, si un personnage ramasse un objet, ça lui coûte un déplacement donc on décrémente son déplacement de un.

2) Savoir si l'on peut collecter la carotte

C'est bien pratique de collecter la carotte mais il est important de ne le faire que quand c'est possible. C'est pourquoi, nous avons fait des fonctions permettant de savoir, pour chaque ressource, si l'on pouvait la recueillir et de quelle ressource il s'agissait.



Voici ces fonctions pour la carotte :

```
public bool Cancollectcarotte()
{
    bool res = false;
    foreach (var VARIABLE :Objscript in jeu.objlist)
    {
        if (X == VARIABLE.X )
        {
            if (Y == VARIABLE.Y &&VARIABLE.data.name=="carotte")
            {
                res = true;
            }
        }
    }
    return res;
}
```

Cette fonction sert à savoir si l'on peut collecter une carotte, elle est présente dans le script des personnages, et regarde si sa position est commune avec la position d'une des carottes des objets des plateaux.

Et elle renvoie true si c'est le cas, la fonction pour savoir de quelle carotte il s'agit est assez similaire. A la place de retourner un booléen on retourne l'objet en question si il existe, null sinon :

```
public Objscript quelcarotte()
{
    Objscript obj = null;
    foreach (var VARIABLE :Objscript in jeu.objlist)
    {
        if (VARIABLE.X-X<=1&&VARIABLE.X-X>=-1)
        {
            if ( VARIABLE.Y - 1 == Y || VARIABLE.Y == Y|| VARIABLE.Y+1==Y)
            {
                if (VARIABLE.data.name == "carotte")
                {
                    obj = VARIABLE;
                }
            }
        }
    }
}
```



3. L'utilisation des ressources

Nous avons vu plus haut les différentes façons d'utiliser les ressources, ici nous voulons expliquer comment nous avons fait en sorte que cela soit possible dans Time Quest 42.

Nous avons réalisé cela grâce aux menus dont nous vous avons parlé plus haut que nous ne pouvions pas remplacer. Pour utiliser les ressources il y a alors trois menus.

L'un servant à acheter des troupes : "Buy", et le dernier à utiliser les ressources simples : les vendre ou bien encore les acheter : "inventory".

En cliquant sur le personnage ses données générales apparaissent notamment le nombre de pièces d'or de toute son équipe mais aussi le nombre de bois de toute son équipe.

Voici un exemple d'écran lorsque un soldat est sélectionné au début :





Nous avons finalement réussi à faire une récolte d'objets optimal claire et peu coûteuse, malgré plusieurs changements de direction. Pour au final faire un Time Quest 42 qui avec sa gestion de ressources nous aura bien fait réfléchir sur différentes façons de faire et réfléchir à la méthode la plus adaptée à un objectif.



C) Les obstacles

Qu'est ce qu'un jeu de stratégie sans obstacles pour freiner notre quête de la victoire ?

Il était important pour nous de mettre en place des mécaniques contraignantes pour le joueur afin de pimenter TimeQuest 42.

La plus grande contrainte de TimeQuest 42, reste pour nous le temps en effet même si 42 tours peuvent paraître beaucoup avec les freins plus concrets présent sur le plateau le temps passe soudainement très vite !!

Avec ce temps, il est impossible de gagner en se servant uniquement de son petit soldat de départ. Les joueurs sont donc encouragés à parcourir la carte en quête de ressources pour former une équipe plus puissante !

Et donc ils vont forcément se frotter à des ennemis qui peuvent piquer !

Il y a dans TimeQuest 42 deux grands types d'ennemis concrets :

1) Les Intelligences Artificielles

_____ Lors de la lecture des consignes du sujet, nous avons compris qu'il était obligatoire de présenter une intelligence artificielle dans le projet présenté. C'est pourquoi nous avons tout d'abord réfléchi à comment intégrer des intelligences artificielles à TimeQuest 42.



Nous voulions que cette intelligence artificielle donne une touche suffisamment intéressante au jeu pour être pertinente.

Mais nous ne voulions pas qu'elle soit trop complexe à coder, en effet nous ne connaissions pas du tout la façon dont on pouvait faire fonctionner une intelligence artificielle. Et surtout ce n'est absolument pas la chose la plus importante du jeu, nous ne voulions pas être trop ambitieux sur l'utilité de cette IA.

Nous aurions pu travailler sur une IA permettant de jouer seul à TimeQuest 42. Mais cela était beaucoup trop complexe par rapport à notre temps et notre niveau actuel. En effet, dans un jeu de stratégie comme celui-ci, les IA choisissant les coups comme un joueur sont très complexes avec toutes les possibilités qu'offre l'environnement du jeu .

On aurait aussi pu travailler sur une IA construisant le plateau de jeu avec des zones et le placement des ressources aléatoirement mais de manière équitable mais cela aussi nous paraissait trop ambitieux .

Puis nous est venu l'idée de rendre le plateau de jeu plus croustillant avec des êtres sauvages pouvant attaquer les personnages des deux joueurs. Et pourquoi pas des êtres sauvages intelligents ? Des bêtes sentant lorsqu'un personnage se rapproche d'eux et pouvant aller à l'attaque lorsqu'ils le voient ? Une tâche plutôt simple mais qui donne plus de piment pour l'expérience de jeu des deux joueurs.



Il ne faut pas seulement faire tomber le château ennemi dans le délai donné mais aussi faire attention à ces bêtes prêtes à nous égorer. Nous avons alors créé deux IA remplissant cette condition du projet :



a) Les barbares :

Les barbares sont les intelligences artificielles les plus faibles du jeu. Ils ne font que 3 dégâts lors d'une attaque. Ils ne peuvent se déplacer que d'une case à chaque tour. Leur milieu de vie sont les montagnes, ils ne peuvent pas sortir de cette zone. Il y a 3 barbares en tout sur la carte.



b) Les ours

L'ours est la seconde unité libre à laquelle il faut faire attention ! Cette unité fait 4 dégâts quand elle attaque, heureusement on ne peut la trouver qu'en forêt ! Il y a 4 ours répartis sur la carte, ils peuvent se déplacer de 3 tuiles à chaque tour.

Après les IA il y a un autre obstacle sur le plateau auquel il faut faire attention !



2) Les troupes du joueur ennemi

En effet, il ne faut pas oublier que le but de TimeQuest 42 est de battre son adversaire !

Il faut l'affronter, pour renverser son château . Pour cela nous avons mis en place des mécaniques permettant d'attaquer les troupes ennemis.

Comme nous l'avons vu plus haut, un joueur peut construire son équipe à partir de 3 unités différentes plus ou moins fortes possédant chacune leurs facultés.

Du soldat au roi en passant par le cavalier, nous pouvons attaquer et tuer les troupes ennemis ! Mais c'est aussi valable pour notre adversaire, en effet il faut faire attention à ses troupes car elles peuvent elles aussi nous attaquer et nous tuer !

Et attention à ça ! En plus de perdre un outil dans la conquête du château ennemi, en perdant un soldat nous perdons aussi une partie de notre or.

La quantité dépendant de l'importance de la troupe ennemie tuée :

B) Le Cavalier

A) Le Soldat



Point de vie : 10

Déplacement : 3

Attaque : 3

Perte : 1 or

Coût : 3 or

Point de vie : 12

Déplacement : 5



Perte : 2 or

Coût : 3 or et 1 cheval

Attaque : 5

C) Le Roi

Point de vie : 14

Déplacement : 6



Perte : 10 or

Coût : 20 or

Attaque : 8 et capacité à transformer 2 troupes ennemis en alliées



Ainsi, grâce à la création des intelligences artificielles mais aussi des différentes mécaniques construites pour les interactions personnage à personnage nous avons donné plus de challenge à TimeQuest 42. Car bien sûr un challenge n'en est plus un sans contraintes.

Et la réussite est d'autant plus exquise lorsque l'on a réussi à surmonter les épreuves les plus dures. Ces obstacles rendent donc plus attractif et moins ennuyant le jeu car les jeux de stratégies sont souvent longs, il a fallu penser à varier les interactions pour éviter que le joueur s'ennuie.

Grâce à la création de ces obstacles nous avons appris à doser la gestion de la difficulté car avec trop d'obstacles le jeu serait impossible et dans le cas inverse trop facile.

Dans les deux cas, cela donne un jeu ennuyeux !



E/ Implémentation des menus

- Menu principal

Pour ce menu, comme pour les autres, nous nous sommes appuyés sur les vidéos proposées par « Tuto Unity FR ».

La principale difficulté a été d'arranger le code proposé par le vidéaste pour l'implémenter à notre projet, plus particulièrement pour le menu d'action et l'inventaire. En effet, les bibliothèques ne sont pas les mêmes en Csharp qu'en Unity et demandent des connaissances supplémentaires, par exemple pour les «GameObjects» ou les «Transforms» qui n'existent pas dans le code habituel.

Après le visionnage de plusieurs vidéos (Voir annexes), nous avons pu recréer un menu "basique": avec les boutons "play" et "quit". Nous envisagions déjà à ce moment-là de travailler sur la sauvegarde, nous avons donc rajouté un bouton "save".

Puis le bouton "setting", qui nous a pris le plus de temps, il fallait en effet faire le lien entre les différentes scènes, gérer les réglages du son présent sur UNITY et le code. Pour les textures, nous avons choisi un thème médiéval, nous nous sommes donc tournés vers une texture papier-bois, texture qui rappelle d'après nous la nature, et qui nous fait penser à Times Quest 42.

Dans un premier temps, nous nous sommes mis d'accord sur le fait que nous commencerons à implémenter dans la fenêtre setting le son et le plein écran.

Or, lorsque nous avons commencé à lancer notre jeu sur plusieurs pc, nous nous sommes rendus compte qu'un réglage de la résolution était indispensable, car les différents ordinateurs n'affichaient pas les mêmes choses.



- Menu pause

Ce menu a été le plus facile à implémenter puisqu'il ne comporte que trois boutons : "RESUME", "SAVE" et "QUIT".

La partie qui nous a posé le plus de problèmes a été l'implémentation de la sauvegarde.

Nous nous sommes donc lancés dans la mise en place d'une fonction qui écrit toutes les données de la partie sur un fichier texte (les différents objets et personnages, sans oublier tous les attributs qui vont avec leur position, leurs apparences ou bien encore les scripts qui vont avec).

Il ne faut pas oublier le contenu des possessions de chaque joueur et du jeu contenu dans les listes.

Il ne suffisait plus qu'à coder une fonction qui lit ces fichiers et qui donne au jeu les attributs de la partie déjà enregistrée. Ce que nous avons fait pour le menu principal.

Times Quest 42 est en effet un jeu de stratégie, il peut donc durer assez longtemps, d'où cette idée de sauvegarde.



- Menu inventaire

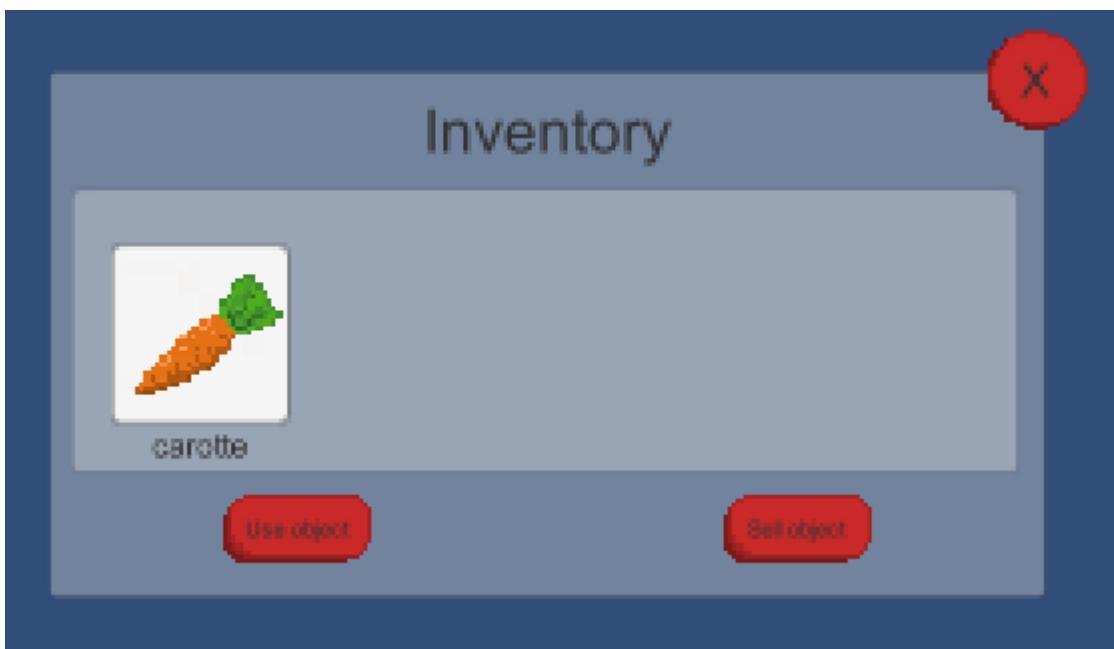


Nous avons implémenté ce menu pour pouvoir stocker les ressources récupérées sur la carte.

Ce stockage servira ensuite à vendre ou à consommer ces dernières.

Pour résumer, dans cet inventaire, il y aura les carottes, le coca et les cheveux que possèdent chaque équipe.

Nous avons donc fait en sorte de créer un menu qui réunit toutes ces caractéristique, le voici :



Ici le personnage a ramassé une carotte qui s'est ajoutée à son inventaire.

Le bouton "Use object", va permettre au joueur de consommer sa carotte et donc d'augmenter ses vies.

Il peut également la vendre (avec le bouton "sell object"), et donc augmenter son or.



Voici l'explication de la fonction qui met à jour le contenu de l'inventaire :

```
void UpdateInventoryItems(List<Objscript> items)
{
    for (int i = 0; i < itemButtonsParent.childCount; i++)
    {
        Destroy(itemButtonsParent.GetChild(i).gameObject);
    }
    for (int i = 0; i < items.Count; i++)
    {
        GameObject button = Instantiate(itemButtonPrefab, itemButtonsParent);
        ItemButtonInventory buttonScript = button.GetComponent<ItemButtonInventory>();
        buttonScript.itemName.text = items[i].name;
        buttonScript.itemImage.sprite = items[i].image;
        buttonScript.manager = actionMenuManager;
        obj.Add(items[i]);
    }
}
```

La fonction commence par détruire tous les objets déjà présents dans l'inventaire pour être sûr de repartir de zéro et ne pas afficher des objets en double, c'est peut-être plus complexe qu'une suppression uniquement si nécessaire mais beaucoup plus sûr.

Ensuite dans le deuxième for, Update Inventory items va créer un nouvel icône pour chaque objet de la liste ainsi lorsqu'on clique sur un personnage il suffit de l'appeler avec la liste d'objets (type de l'objet en paramètre de la fonction) de son équipe pour qu'elle affiche tous les objets de l'équipe voulue.



Voici ici l'explication de la fonction pour utiliser les objets depuis l'inventaire. Par utilisation, nous entendons le fait de ne pas vendre les objets mais bien de manger de la nourriture, de pouvoir naviguer avec un bateau ou autres.

```

public void UseObject()
{
    string num = actionManager.selectbutton;
    Objscript res = null;
    if (actionManager.jeu.PersoSelect.Team == 0)
    {
        foreach (Objscript aaa in actionManager.jeu.objequipeRouge)
        {
            if (aaa.name == num)
                res = aaa;
        }
    }
    else
    {
        foreach (Objscript aaa in actionManager.jeu.objequipeBleu)
        {
            if (aaa.name == num)
                res = aaa;
        }
    }
    if (res != null && res.data.name != "boat")
    {
        actionManager.jeu.PersoSelect.Vie += res.GainVie;
        actionManager.jeu.objequipeBleu.Remove(res);
        actionManager.jeu.objequipeRouge.Remove(res);
    }
    else if (res != null)
    {
        actionManager.jeu.PersoSelect.monter();
    }
    actionManager.selectbutton = "";
    List<Objscript> OTeam;
    if (actionManager.jeu.PersoSelect.Team == 0)
        OTeam = actionManager.jeu.objequipeRouge;
    else
        OTeam = actionManager.jeu.objequipeBleu;
    UpdateInventoryItems(OTeam);
}

```



Pour commencer, nous récupérons nom de l'objet que l'on veut utiliser une fois qu'il a été cliqué et nous préparons la variable contenant l'objet à utiliser

```
string num = actionMenuManager.selectbutton;
Objscript res = null;
```

Ensuite, nous séparons la recherche car l'équipe bleu et rouge n'ont pas forcément les mêmes objets dans leur inventaire. C'est pourquoi nous avons fait une liste avec les objets de l'équipe rouge et une avec les objets de l'équipe bleu.

Ainsi nous commençons par vérifier à quelle équipe appartient le joueur pour qui apparaît l'inventaire. Puis, lorsque nous trouvons un objet qui porte le même nom que le bouton qui a été cliqué, nous sauvegardons cet objet dans le paramètre réponse "res".

```
if (actionMenuManager.jeu.PersoSelect.Team == 0)
{
    foreach (Objscript aaa in actionMenuManager.jeu.objequipeRouge)
    {
        if (aaa.name == num)
            res = aaa;
    }
}
else
{
    foreach (Objscript aaa in actionMenuManager.jeu.objequipeBleu)
    {
        if (aaa.name == num)
            res = aaa;
    }
}
```



Maintenant, nous avons récupéré l'objet à utiliser, il faut donc l'utiliser. Mais le type de l'objet influe sur ses actions donc nous regardons quel objet c'est.

Si ce n'est pas un bateau, c'est forcément une ressource alimentaire qu'on veut donc manger. Alors on attribue ses points de vie au personnage avec lequel nous avons ouvert l'inventaire, puis nous le supprimons de la liste des objets de son équipe.

```
if (res != null && res.data.name != "boat")
{
    actionMenuManager.jeu.PersoSelect.Vie += res.GainVie;
    actionMenuManager.jeu.objequipeBleu.Remove(res);
    actionMenuManager.jeu.objequipeRouge.Remove(res);
}
```

En revanche si c'est un bateau que nous voulons utiliser, nous allons chercher à monter sur ce bateau si c'est possible avec le personnage sélectionné.

```
else if (res != null)
{
    actionMenuManager.jeu.PersoSelect.monter();
}
```



Pour finir, il ne reste plus qu'à mettre à jour les données de l'inventaire. Pour cela, nous commençons par attribuer la nouvelle liste de ressources de l'inventaire "Team".

Et, nous finissons par appeler la fonction update pour mettre à jour l'inventaire.

La fonction utilisée pour vendre les objets est très similaire hormis le passage ou on les vend.

```
if (res != null)
{
    actionMenuManager.jeu.totalmoney[actionMenuManager.jeu.PersoSelect.Team] += res.GainOr;
    actionMenuManager.jeu.objequipeBleu.Remove(res);
    actionMenuManager.jeu.objequipeRouge.Remove(res);
    actionMenuManager.selectbuton = "";
}
```

Ici, au lieu d'ajouter les points de vie au personnage sélectionné, nous ajoutons simplement l'argent qu'il vaut à la vente à la trésorerie de l'équipe.

La fonction MONTER qui permet à un personnage de monter sur un bateau est utile pour comprendre la façon d'utiliser un bateau.

MONTER est appelée si nous utilisons un bateau depuis l'inventaire.



```

public void monter()
{
    if (monterbateau())
    {
        bateau = true;
        Deplace = 6;
        Objscript bateau1 = quelbateau();
        if (Team == 0)
            jeu.objequipeRouge.Remove(bateau1);
        else
            jeu.objequipeBleu.Remove(bateau1);
        jeu.objlist.Add(bateau1);
        bateau1.istaken = this;
    }
}

```

Cette fonction n'est pas compliquée, nous regardons si le soldat concerné peut monter sur le bateau. C'est-à-dire qu'il s'engage dans l'eau.

Si c'est le cas nous allons faire en sorte que le bateau puisse avancer avec le personnage en attribuant au navire ce personnage comme possesseur. Et en disant qu'il est pris.

On a pas oublié d'enlever ce bateau de la liste des objets de l'équipe du joueur pour que ce dernier n'apparaisse plus dans l'inventaire et ne soit pas utilisable. Mais pour ne pas le perdre on l'ajoute à la liste des objets du plateau .



- Menu d'achat

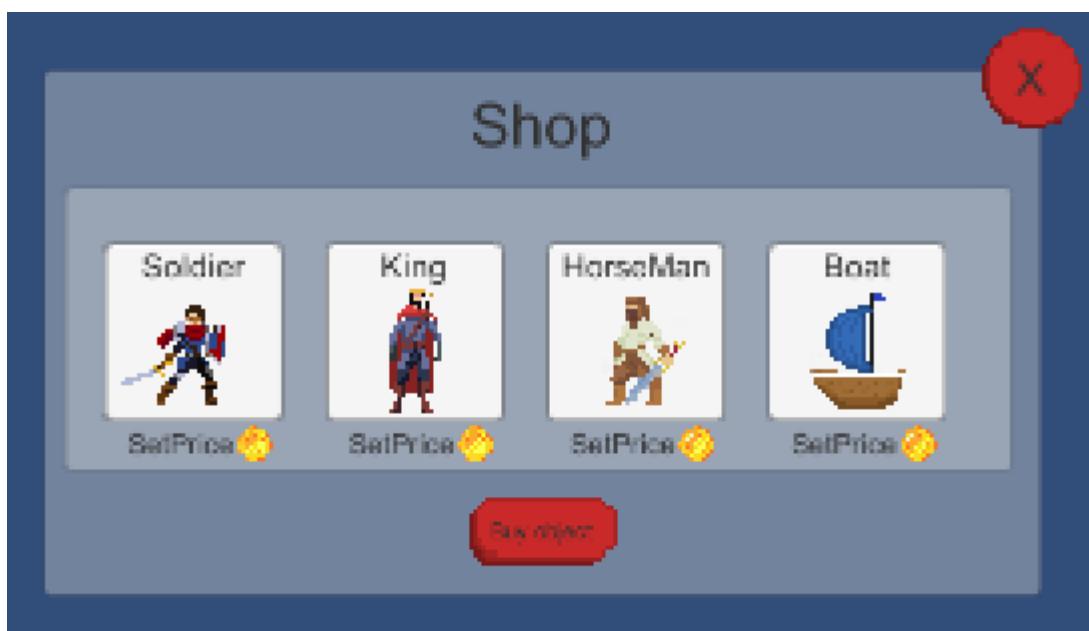
Le menu d'achat va nous permettre d'acheter des alliés.

En effet, si un joueur possède les ressources nécessaires à cet achat il va pouvoir grâce à ce menu, acheter des personnages.

Pour rappel, pour un soldat il faut 3 or, pour un cavalier il faudra un cheval et 3 or et pour un roi le coût s'élève à 20 or.

Il fallait donc implémenter un menu qui présente chaque personnage avec leur coûts, afin que chaque joueur puisse facilement accéder à cette "boutique".

Le voici :



Donc, il est possible d'acheter un soldat, un roi, un cavalier ou un bateau.

Les prix s'afficheront en dessous de ces derniers, il suffit de sélectionner le personnage/objet et de cliquer sur "Buy object", afin de pouvoir l'acheter.



F) Choix des coûts de chaque évolution

L'une des choses qui a été le plus dur à faire et qui est encore en évolution au moment de la rédaction de ce rapport de projet c'est le réglage des coûts de chaque évolution.

Les coûts d'évolution sont les ressources nécessaires pour réaliser des actions dans le jeu, comme les déplacements.

Etant donné que le soldat a été le premier personnage imaginé, c'est sur lui que nous nous sommes appuyés pour la création des autres personnages/objets.

Donc pour rappel notre soldat possède 10 vies, et son coût est 3 or. Partant de là, il a fallu créer des évolutions donc que les personnages suivants soient plus avantageux que le soldat.

Mais attention, il ne faut pas que ce soit chose facile d'avoir un cavalier ou un roi.

Pour le cavalier, un soldat expérimenté ayant plus de vies (nous les avons fixées à 12), il faut capturer un cheval au risque de se faire agresser par une troupe ennemie, et donc de perdre la vie.

En contrepartie, ce cavalier se déplacera aussi plus vite (5 cases contre 3 pour le soldat).

Il nous reste maintenant le dernier personnage, le roi.
Il représente le personnage le plus fort du royaume, celui qui a tous les droits.

Mais comme pour le cavalier, cette fois-ci le coût est de 20 or. En échange de cela, il aura la possibilité de se déplacer encore plus vite (jusqu'à 7 déplacements) et il aura le pouvoir de transformer un joueur



adverse en allié, un avantage plus qu'intéressant pour arriver au terme de la quête.

Jusqu'à là, nos personnages étaient prêts.

Mais en s'arrêtant ici, le jeu n'aurait pas été assez intéressant.

Le plateau est assez grand pour que les deux équipes n'aient pas à se croiser.

Or pour que Time Quest 42 soit un jeu de stratégies, il a fallu provoquer ces rencontres, proposer plusieurs façons d'arriver au château adverse.

Pour ce faire, nous avons implémenté sur la carte plusieurs événements :

Nous avons déjà explicité plus haut les modalités/valeurs de chaque événement, mais comment avons-nous fixé ces prix/coûts ?

Pour la nourriture, tout a été un choix de groupe, en effet les friandises apparaissent de manière aléatoire, et servent à apporter un peu de réconfort aux joueurs sans pour autant être un avantage considérable.

Le plus gros réconfort est apporté par le poisson, en effet il n'y en a qu'un, et pour l'attraper, il faut utiliser un bateau qui est assez cher.

Pour la construction du bateau, il faut 12 bois :

Ce chiffre nous paraît idéal.

C'est un multiple de 3, donc un soldat pourra s'en procurer un après 4 tours dans la forêt.

Ce qui rend le bateau difficilement accessible, il procure certes un bel avantage (il permet d'avancer plus vite en un tour), mais pour arriver à l'utiliser, il faut ramasser beaucoup de bois au risque que l'équipe adverse ne traverse la carte.



Le cheval est indispensable à la création d'un cavalier, mais il peut aussi être vendu si le joueur est en manque d'or, c'est la ressource la plus chère à la vente, car elle permet d'avoir un avantage considérable si l'on s'en sert pour avoir un cavalier.

Nous avons également implémenté un score par équipe afin de déterminer un gagnant à la fin des 42 tours si aucun des châteaux n'est détruit.

Ce score représente la somme des vies de tous les personnages d'un joueur et de l'or d'un joueur.

S'ajoutent ensuite les points de vie du château et le nombre de ressources dans l'inventaire.

Si les deux scores sont égaux, alors c'est une égalité parfaite.

IV- Conclusion



A/ Reprise du cahier des charges

Le projet étant livré, il semble important de reprendre le cahier des charges afin de constater les changements et les difficultés que nous avons rencontré.

Comme vu au dessus, certaines caractéristiques du jeu ont dû être revues.

En effet, après plusieurs essais, nous avons dû remodeler le nombre de vies, les attaques, la position des IA de chaque personnage/objet.

A part cela nous sommes heureux de pouvoir amener un jeu qui correspond parfaitement à celui annoncé sur le cahier des charges.

Nous avons donc apporté de légères modifications sans s'éloigner de notre idée principale. Tout ce qui a été annoncé a été livré.

Pour ce qui est de la répartition des tâches, nous l'avons complètement remodelée. En effet, il ne s'agissait plus de se partager le travail à effectuer par secteur (graphismes, code, site web), mais de s'entraider, nous nous sommes rendus compte que pour avancer, il est important de se serrer les coudes, donc pour l'organisation, nous avons fait preuve de complémentarité.

Cela a été possible grâce à une organisation minutieuse et des profils avec chacun leurs atouts.

Le serveur Discord du groupe, nous permettait d'échanger en cas de problèmes, en plus des deux heures d'appel par semaine afin de faire un point sur notre avancée, de partager nos écrans afin de se débloquer, de se fixer des objectifs.



Alors oui il y a eu des moments de doute, oui nous avons quelquefois pris du retard, mais nous avons su le rattraper, et trouver des solutions, des alternatives pour répondre à nos objectifs.

Par moments nous nous pensions trop ambitieux mais au final, nous sommes fiers d'avoir pu combler nos ambitions.

B/ Point collectif sur les apports du projet

La réalisation de ce projet n'a pas été facile et cela sur plusieurs aspects. En effet, nous n'avions aucune expérience pour ce qui est de la réalisation d'un jeu vidéo. Nous voulions quand même faire quelque chose de différent.

Jusqu'à la première soutenance nous avons plus travaillé sur la conception du jeu et la compréhension de nos outils. Une fois que nous avions compris comment utiliser Unity, chacun de nous était d'accord sur une chose : la réalisation technique du jeu n'était pas l'obstacle le plus gros de la réalisation de ce projet.

La partie la plus complexe de la réalisation de Time Quest 42 c'était le travail de groupe, en effet, le chemin vers l'entente et la complémentarité a été long.

Pour reprendre notre parcours chronologiquement, nous avons d'abord fait face au doute et aux conflits. En effet nos quatres personnalités, nos façons de travailler diffèrent beaucoup. Certains d'entre nous aiment travailler la nuit, d'autres le matin. Certains plus angoissés préfèrent prévenir un éventuel retard en se mettant rapidement au travail, d'autres préfèrent s'organiser une semaine de travail intense et fournir moins de travail durant les semaines de cours.

Ce fut donc d'abord compliqué de trouver un terrain d'entente, nous avions des goûts différents, des envies différentes, et étant donné que



notre jeu n'est pas un jeu "connu", "déjà existant", il a fallu se mettre d'accord sur le moindre détail.

Nous avons donc organisé une "Réunion d'Urgence", où il a fallu mettre les choses à plat, briser la glace. Cette réunion à certes été longue, mais à l'issue de cette dernière nous avons pu mettre en place un planning de développement du projet, où chacun peut travailler à son rythme, et surtout ne pas se sentir seul.

Il a fallu faire des compromis, mais c'est de cette façon qu'est né le groupe soudé que nous sommes aujourd'hui. Alors ces derniers mois, nous avons mûri, nous avons appris à nous ouvrir aux autres et à communiquer.

Nous nous sommes rendu compte que pour travailler en groupe, maintenir une certaine hygiène de travail est indispensable.

Nous avons découvert le sentiment de travailler pour un but commun, la fierté de faire avancer le groupe, de nous porter soutiens, aussi émotionnellement que dans le travail.

Et tout cela a été possible grâce à une chose très importante :

Notre envie commune de livrer notre premier jeu dans les délais et notre soif d'apprendre.

Finalement, nous sommes tous fiers d'avoir fini ce projet, cela nous a appris à nous adapter à des méthodes de travail différentes des nôtres. Grâce à ce projet nous avons aussi réussi à mettre en avant les qualités de chacun pour obtenir le travail le plus pertinent et efficace possible.

Nous sommes tous plus satisfaits les uns que les autres d'avoir réussi notre premier jeux vidéo, en ayant trouver des alternatives à des problèmes que nous n'avions pas envisagé. Time Quest 42 nous a fait grandir et ce sur plus d'un plan .



C/ Annexes

Lien du site web :

<https://timequest42.github.io/TimeQuest42/>

Choix musique :

<https://www.musicscreen.be/musique-libre-de-droit/Catalogue/L-erreur.html>

Choix personnages :

<https://sventhole.itch.io/>

Création du menu :

https://www.youtube.com/watch?v=4LkiX_XioXg

<https://www.youtube.com/watch?v=ABWt1ipTAMg&t=948s>

<https://www.youtube.com/watch?v=GURPmGoAOoM>

Modélisation de la Carte :

https://www.youtube.com/watch?v=MOehmikBox8&list=PLHpC3gVKYuvfc_xrThC4arYTjOyU2mKWH8

Implémentation de la souris :

<https://www.youtube.com/watch?v=VN1gryMOpWo&t=591s>

Sauvegarde : TP5-Registre de Dumbledore (cours de programmation)

https://drive.google.com/file/d/1l-_4uM_3njsAOsxw1Ad34xFuDxNRs0iG/view?usp=sharing