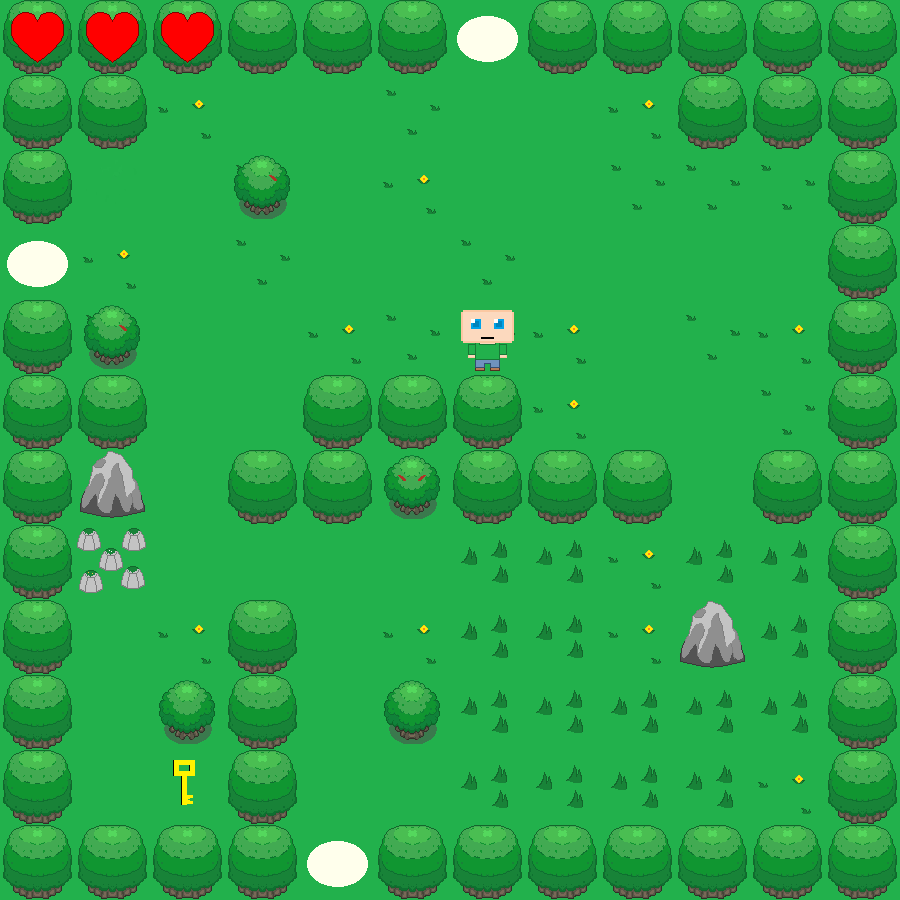
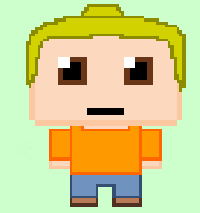
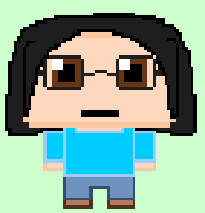
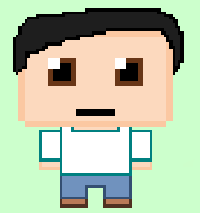
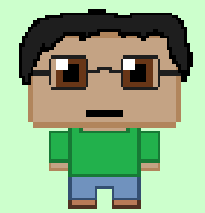
***A Classic Tale***



Un jeu de :

 CLARA  ANNE

 YANN  THOMAS

Professeur d’ISN : M. Gastin

Présentation

Au cours de cette année scolaire, tout comme un TPE, l’ISN propose un projet noté à concevoir en groupe autonome. Ce projet doit mettre en évidence des compétence propre à l’informatique et aux sciences du numérique.

Ainsi après quelque cours nous apprenant les bases avec des logiciels de programmation comme Processing et Java-school l’heure fut venue de choisir et de concevoir notre projet !

Au sein de notre groupe nous étions d’accord sur une chose : nous utiliserons Processing, qui code en java et notre but serait de faire un jeu vidéo. Mais lequel ?

Pour ne pas complexifier les choses, le style du jeu serait retro (comme les premiers mario ou comme pac-man) en effet les graphismes en seraient plus “carré”. après quelques réflexions nous avons décidé de faire un jeu de plateforme comme mario car dans les faits cela semblait assez simple à faire. Les déplacements latéraux de notre carré (les images du perso n'étant pas encore dessinés) furent assez simple car nous nous sommes inspiré d’un jeu créer avec notre professeur, mais le déplacement verticale fut impossible à coder (en raison de la gravité).

soit le perso restait en haut soit il ne décollait pas du sol.

Nous avons donc compris que le déplacement case par case, une caméra fixée sur la carte (comme dans les vieux zelda) étaient à notre portée d’apprenti développeur.

Ce qu’on voulait c'était une aventure mobile ou l’on traverserait des mondes et non un jeu figée comme Pac-man ou tetris, où des ennemis serait la pour entraver notre quête.

C’est ainsi qu’est né “A classic Tale “ mettant en scène notre petit héros chauve parcourant différents biomes esquivant des monstres variés.

**Cahier des charges de l'équipe**

Développer un jeu aux multiples fonctions :

* -un écran d'accueil présentant les différents niveaux de difficulté
* -Diverses cartes en références aux multiples environnements
* -Un héros et ses mouvements
* -Une faction d'ennemis
* -La possibilité de mourir mais aussi de réessayer
* -faire de petites boîtes de dialogue avec un PNJ (personnage non joueur)

Nous cherchions un projet qui nous permettrait d'exploiter les ressources de Processing, le logiciel de programmation dont nous disposions. Le concept de jeu vidéo s'est très vite imposé car il permet d'exploiter les fonctionnalités graphiques et musicales du logiciel, tout en nous imposant des défis de programmation qui nous plairait à tous .

Notre défi était de créer un jeu original consistant à traverser plusieurs niveaux tous différents les uns des autres. Les ennemis étant présents afin de compliquer ce parcours. Chaque environnement original et scénario était créé par l'imagination du groupe.



Nous avions également pour but de mettre en scène des personnages pouvant bouger les uns à côté des autres. Cela était la base du jeu. Tout d'abord nous avons donné naissance à notre héros puis de nouveaux personnages ont été créés. Tous ont des rôles propres et sont attribués à un terrain. Il s'agissait également de créer des ennemis et des éléments comme la lave pouvant provoquer la mort du héros.

Nous avons construit notre projet à partir de notre culture personnelle. Or, notre principale expérience de l'informatique s'est faite par les jeux vidéos. Nous nous sommes d’abord tournés vers les jeux Mario et son style de jeu mais nous avons finalement opté pour un style en “vue du dessus” comme les premiers jeux Zelda.

Nous nous sommes donc inspirés de références du genre pour établir la structure de notre jeu: un personnage incarnant le joueur, qui évolue à travers différents niveaux pour survivre et atteindre un objectif.

Équipés des ordinateurs du lycée mais aussi nos ordinateurs personnels, nous avons utilisés les logiciels proposés. Il s'agissait en l’occurrence de Processing. Ce logiciel était nouveau pour nous qui n'avions jamais codé auparavant.

Au fur et à mesure, nous nous sommes familiarisés avec ce dernier ce qui nous a permis de mettre en place un jeu intéressant.

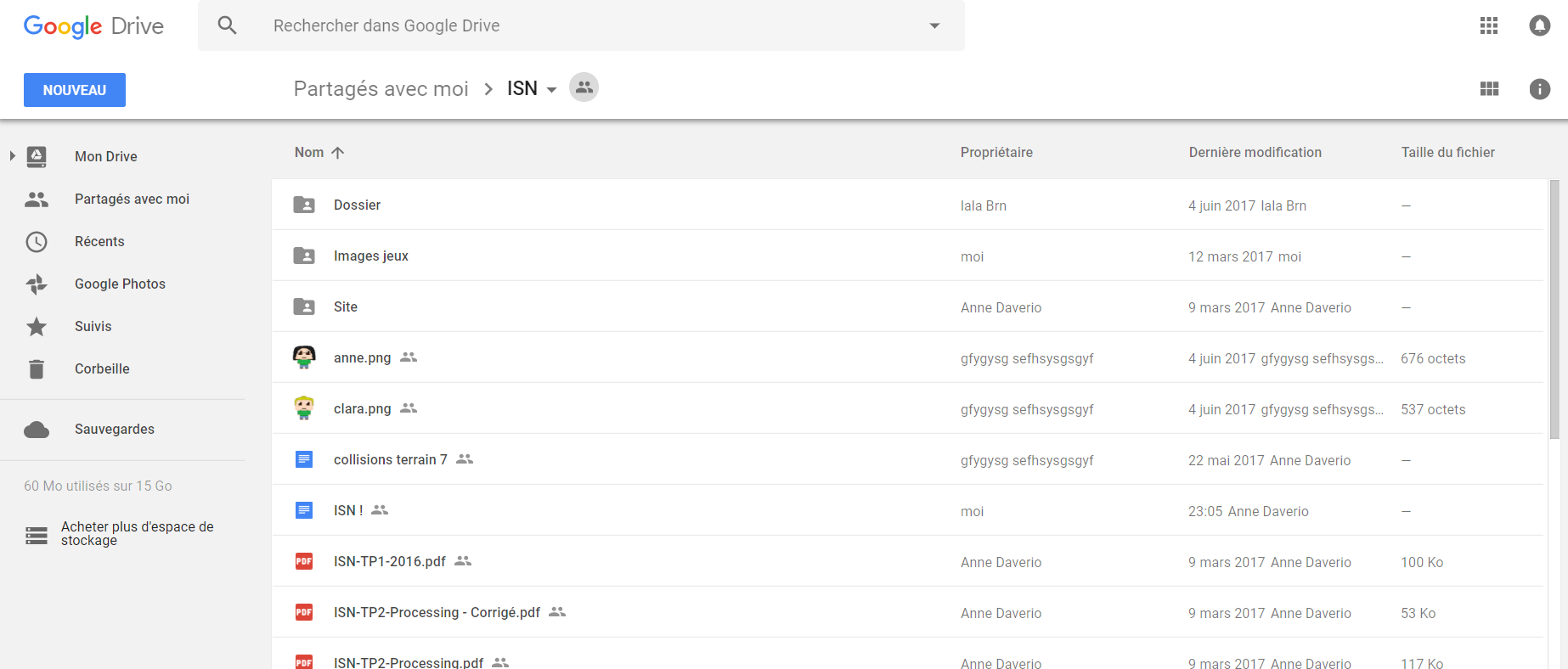
Puis, nous avons également eu recours à PhotoFiltre, un logiciel de traitement d’image gratuit. Ce logiciel a permis la création des divers environnements. Grâce à ce dernier, il a été possible de modifier facilement les terrains et d'y placer divers obstacles grâce à une grille ou la taille des cases pouvait être modifiée . Utilisé auparavant, il a été pour certains l'occasion de découvrir de nouvelles fonctionnalités cachées et d’apprendre à l’utiliser.

Nous avons aussi eu à construire un site web composé de quelques pages liées entre elles. Nous avons donc appris à utiliser les langages HTML et CSS. Il nous fallait faire la mise en page, insérer des images, des liens, créer des listes et des tableaux… Autant de découvertes qui nous ont fait voir d’un oeil neuf le réseau Internet que nous utilisons chaque jour.

Répartition des tâches et collaboration au sein de l’équipe

| Tâches | Planning | Noms |
| --- | --- | --- |
| Faire bouger le personnage | du 05/12 au 12/12 | Clara |
| Afficher le personnage | du 03/01 au 09/01 | Anne |
| Créer les terrains dans la forêt (dimensions) | du 03/01 au 09/01 | Thomas (décors)  (dimension) |
| Intégrer les collisions | tout au long de l’année, pour  chaque terrain ajouté | Clara |
| Faire bouger les ennemis (chemin déjà tracé/bouger en ligne droite après avoir vu le personnage) | du 09/01 au 16/01 | Anne |
| Définir les ennemis (dégât au personnage) | du 09/01 au 23/01 | Thomas |
| Dessiner les personnages (le héros et les ennemis) | du 09/01 au 16/01 | Thomas |
| Afficher les terrains et leur spécificités | du 16/01 au 30/01 | Clara |
| Transition entre les terrains | du 23/01 au 06/02, puis à chaque terrain ajouté | Yann |
| Interactions entre le héros et les élément du jeu (bouger caisse, parler,...) | du 23/01 au 06/02 | Anne |
| Dessiner les objets | du 06/02 au 27/02 | Thomas |
| Mettre plus d’obscurité sur un terrain | du 27/02 au 06/03 | Clara |
| Créer un événement avec un PNJ (personnage non-joueur)(dialogue/action) | du 27/02 au 20/03 | Yann |
| Dessiner le terrain transition (passage forêt/montagne) | du 06/03 au 20/03 | Thomas  Yann |
| Dessiner les terrains dans la montagne | du 20/03 au 03/04 | Thomas  Clara |
| Créer des déplacements spécifique au yéti | du 27/03 au 03/04 | Anne |
| Dessiner le yéti | du 03/04 au 24/04 | Thomas |
| Système de vie (vies et Game Over) | du 20/03 au 03/04 | Thomas |
| Créer différentes difficultés (sélection et effets) | du 03/04 au 24/04 | Yann / Anne |
| Créer une animation pour un ennemi (fumée) | du 02/05 au 15/05 | Thomas(dessin)  Yann (animation) |
| Ralentissement dans la neige ? (à confirmer) |  |  |
| Afficher l’écran de fin | du 15/05 au 22/05 | Yann |
| Musiques du jeu | du 22/05 au 29/05 | Yann |

Nous avons très souvent travaillé en dehors des séances pour continuer le projet. La plupart des tâches listées dans ce tableau se sont effectués lors du travail personnel et le nombre de séance écrit dans la case “planning” servait très souvent à confirmer par les autres membres de l’équipe si cela convenait. Nous avons utilisés Google Drive pour partager et modifier les éléments du projet, en créant un dossier où seul nous avons accès. C’était un moyen efficace pour partager les informations sans avoir comme problème que tel éléments n’était pas disponible pour les autres car il était en possession d’une seule personne. Pour discuter en dehors des séances nous avons créé un groupe de conversation sur le réseau social Facebook.



Par ailleurs, il a fallu concilier la réalisation du programme avec celle de notre site en HTML, ainsi qu’avec l’écriture de nos dossiers pour l’évaluation. Ce sont des tâches que nous avons réalisées à partir du mois d’avril, et qui ont demandé beaucoup d’investissement.

Ma réalisation

Pour notre projet, le travail a été réparti entre les 4 membres du groupe. J’ai donc aidé comme les autres tout au long de l’élaboration du programme, pour créer de nouvelles fonctions ou étoffer celles déjà créées.

Mon rôle a plutôt consisté à trouver des manières de résoudre des problèmes dans notre programme, pour permettre au projet de continuer à avancer.

Plus particulièrement, j’ai travaillé à afficher une image pour notre personnage et à donner un déplacement autonome aux ennemis, en particulier au “yéti”. J’ai aussi rédigé des pages de notre site en HTML, avec le fichier en CSS qui les accompagne.

A propos de notre site, le codage HTML en lui même a été une découverte. Nous avons eu à créer 4 pages liées entre elles pour présenter notre jeu, notre équipe, ainsi que la spécialité ISN en général. Et nous avons pour cela appris les bases de ce langage ainsi que du CSS.

*Les fonctions que j’ai codées*

* *void imageperso()*

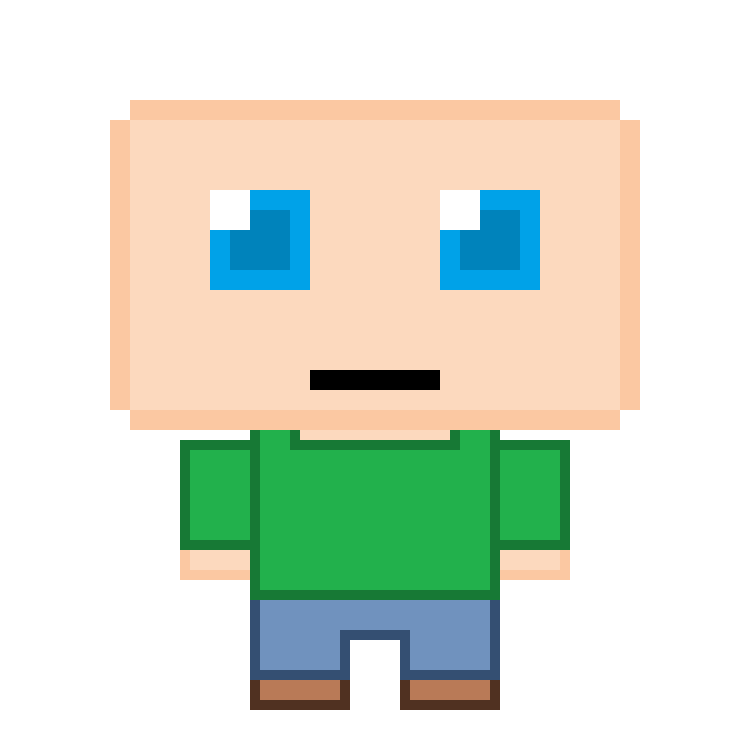
Son nom est assez explicite: c’est une fonction qui affiche l’image du personnage aux coordonnées où il se trouve actuellement.

L’image elle même, ainsi que la position du personnage, changent selon les actions réalisées par le joueur. Cela est géré par une autre fonction, le void bougerperso(). Le rôle du void imageperso() est de lier l’image à ces coordonnées.

Mais cette fonction sert aussi à afficher les objets que possède le personnage, ainsi que le nombre de vies qu’il lui reste.



*Fonctionnement:*



-Afficher l’image : *image(perso,persoX,persoY,75,75);* ;

-Afficher l’image de la clef et de la carotte si le personnage les possède ;

-En fonction du nombre de vies restant au joueur (stocké dans la variable coeur),

afficher le même nombre de coeurs pour signaler cette information au joueur.

* *void ennemi()*

Cette fonction gère la position et les déplacements des ennemis pour chaque terrain.

En effet, nous avons choisi d’utiliser au maximum 5 ennemis sur un terrain. Nous avons donc créé 5 entités, qui tuent le héros lorsqu’elles se trouvent aux mêmes coordonnées que lui. Mais sur les terrains où nous utilisons moins d’ennemis, il faut décider que faire de ceux que nous n’utilisons pas. Cette fonction a pour rôle de placer ces ennemis en trop dans le coin en haut à gauche du terrain, qui est toujours occupé par un élément de décor et donc inaccessible pour le joueur. Il serait problématique de se faire tuer par un ennemi qui n’est pas sensé exister ! Par ailleurs, cette fonction remplace aussi l’image des ennemis non utilisés par une image vide, pour les rendre invisible. 

Le void ennemi() est également de gérer les déplacements propres à chaque ennemi: certains font des allers-retours, d’autres avancent en ligne droite quand ils aperçoivent le personnage, d’autres encore tournent en rond… À chaque terrain, les ennemis sont différents, et ont des déplacements différents. Il est donc nécessaire de les redéfinir à chaque fois.

Pour le dernier terrain, il faut aussi gérer l’animation d’un nouvel ennemi: c’est une ombre maléfique, qui ondule: il faut donc la faire changer d’image régulièrement.

*Fonctionnement:*

-La fonction est divisée en plusieurs parties correspondant aux différents terrains à l’aide de boucles if concernant la variable *terrain*, qui change de valeur à chaque terrain.

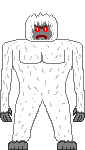
-Tout d’abord, on définit les coordonnées initiales de chaque ennemi. Elles sont initialisées à 0,0 si l’ennemi n’est pas utilisé.

-On établit ensuite le déplacement des ennemis utilisés:

-si il fait des allers-retours, on augmente régulièrement ses coordonnées, et on établit une condition qui le fait repartir dans l’autre sens une fois un certain point atteint.

-si il fonce vers le personnage, on crée une variable qui change de valeur quand le personnage passe à la même abscisse ou ordonnée que l’ennemi, et on augmente régulièrement les coordonnées de l’ennemi si la variable a changé de valeur.

-si il tourne en rond, on définit la direction vers laquelle il va avancer en fonction de sa position.



* *void yeti()*

Cette fonction définit un ennemi nouveau: le yéti, qui se déplace aléatoirement, sauf si il aperçoit le personnage. Dans ce cas, il avance dans sa direction à une vitesse plus grande que sa vitesse normale.

De plus, dans son déplacement, il doit prendre en compte le décor et ne pas passer au travers. Nous avons donc dû établir pour lui un système semblable à celui des collisions pour le personnage.

Son fonctionnement étant très différent des autres ennemis, nous avons préféré créer une fonction uniquement pour lui.

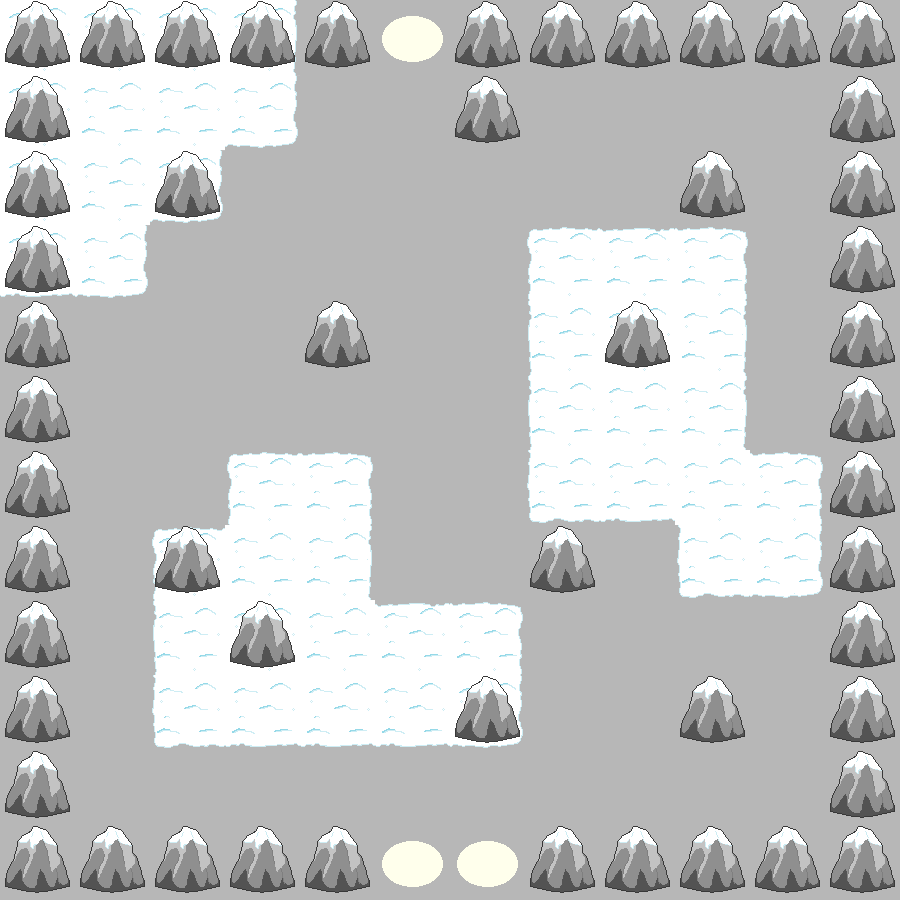
*Fonctionnement:*

-On crée une variable *dir* qui prend aléatoirement un valeur entre 1 et 4: elle définira sa direction vers laquelle il sera tourné

-On définit une variable *vu* qui change de valeur si le personnage se trouve à la même abscisse ou ordonnée que le yéti au moment où celui-ci est tourné vers lui.

-Quand *vu* change de valeur, on fait avancer le yéti dans la direction du personnage.

-Quand *vu* est égal à 0, le yéti avance d’une case dans la direction choisie aléatoirement avec *dir*



*Terrain du yéti*

*Les difficultés que j’ai rencontrées*

Tout ce que nous avons réalisé n’a pas toujours été facile ni évident. En soi, créer de toute pièces un programme informatique n’est jamais un processus simple, surtout lorsque l’on n’a jamais programmé auparavant: il faut apprendre un nouveau langages, avec ses règles et ses limites, puis concevoir chaque élément avant de le transcrire en code informatique. Cette transcription est probablement la partie la plus délicate.

Ainsi, pour le void imageperso(), il nous a fallu comprendre que la commande *PImage* créait un “emplacement” où on devait placer une image avec *loadImage.*

Par la suite, cela nous a permis de définir la position où il fallait afficher l’image, sans se soucier de quelle image on allait placer là, car cela est défini dans un autre void.

Dans le void ennemi(), nous avons mis un certain temps avant de résoudre un certain problème: les ennemis du terrain précédent restaient présents, et causaient la mort du personnage de manière imprévisible. Mais supprimer complètement ces ennemis pour un seul terrain nous posait aussi beaucoup de problèmes, car nous avions du mal à les remettre en place ensuite.

La solution a finalement été assez simple. Comme je l’ai dit plus haut, il suffisait de modifier les coordonnées des ennemis inutiles pour les placer à un endroit où il ne pourraient pas entrer en contact avec le personnage.

Pour le reste des déplacements des ennemis, en revanche, le mécanisme était plutôt instinctif. Il suffisait de bien définir les différentes conditions qui modifieraient le déplacement.

Le void yeti(), lui, a été un nouveau défi. Tout d’abord, pour définir dans quelle direction le yéti va avancer après avoir vu le personnage, nous avons eu des difficultés. Si on déclenche le déplacement à partir du moment où l’abscisse du personnage et du yéti sont égales,il peut arriver que le yéti fonce à gauche alors que le personnage est à sa droite.

C’est pour cela que nous avons ajouté donné 4 valeurs à la variable *vu* , une pour chaque position possible du personnage par rapport au yéti: au dessus, en dessous, à gauche et à droite. Ainsi, pour chaque valeur de *vu* , on déclenche le déplacement approprié. Par ailleurs, si l’abscisse ou l’ordonnée du yéti est égale à celle du personnage mais que le yéti n’est pas tourné dans la bonne direction (avec la variable *dir*), la variable *vu* restera égale à 0.

Il a ensuite fallu éviter que le yéti traverse le décor. Nous avons donc dû créer un système de collision identique à celui du personnage, dans les 4 directions. Heureusement, les collisions étant déjà faites pour le personnage, nous avons seulement eu à les copier, en créant des variables pour remplacer celles gérant la collision du personnage.

En somme, avec un peu de persévérance et d’imagination, nous avons réussi à surmonter presque tous nos problèmes, ou à les contourner dans le pire des cas.

Intégration de mon travail

au sein du projet

Tout d’abord, le void imageperso() permet au joueur de comprendre ce qu’il se passe: l’entité “perso” ne correspond à rien en dehors du programme. Il est nécessaire de la représenter par une image, sans quoi le joueur avancerait à l’aveugle. Et l’affichage de sa vie et de ses objets permet au joueur d’avoir accès en temps réel à son statut, et d’adapter son comportement en conséquence.

Un autre aspect important de notre jeu est le déplacement des ennemis. Il rend le jeu bien plus dynamique et en augmente la difficulté, d’autant plus que la vitesse de déplacement des ennemis dépend de la difficulté sélectionnée.

Les différentes trajectoires des ennemis mettent le joueur au défi à chaque niveau, en le surprenant et en l’obligeant à élaborer des stratégies pour atteindre la fin du jeu. Ainsi, pour éviter les ennemis “arbres maléfiques”, il faut avoir déjà expérimenté leur déplacement et savoir qu’il faut déclencher leur déplacement puis reculer pour les éviter. De même, pour surmonter l’épreuve du yéti, on peut se mettre à l’abri derrière un rocher pour le semer.

Finalement, une grande partie de notre jeu repose sur la capacité du joueur à comprendre et à réagir aux dangers extérieurs. Il est donc important de créer des déplacements à la fois cohérents et inventifs.

Perspectives et bilan

En regardant notre projet à ce jour, il apparaît que la structure et les règles de notre jeu sont bien établies. C’est un jeu que nous avons réussi à rendre cohérent et jouable, et en cela nous avons déjà rempli une part essentielle de notre cahier des charges.

Mais il manque peut-être d’aboutissement sur certains points. Nous avions beaucoup de projets que nous n’avons pas tous réalisés, faute de temps ou de moyens.

Par exemple, il était prévu de créer d’autres terrains dans d’autres environnements (comme un désert), mais nous avons dû nous contenter d’aller à l’essentiel pour faire un terrain final.

Nous avions d’ailleurs initialement prévu un boss de fin, qui combattrait avec des projectiles à éviter. Mais encore une fois, le temps nous a manqué.

Dans la perspective d’une reprise du codage de ce jeu, peut être pourrions nous chercher à réaliser ces idées. Il pourrait aussi être intéressant de s’intéresser plus au parcours du joueur qui découvre le jeu, car les conditions à remplir pour avancer dans le jeu ne sont pas toujours évidentes.

Mais en somme, on peut dire que ce jeu nous a beaucoup appris, tant en matière d’informatique que de gestion d’une équipe et d’un temps limité. Nous avons dû approfondir nos connaissances par nous mêmes pour relever les défis que nous nous étions fixés, et faire preuve de persévérance. Bien que de nos jours, beaucoup utilisent des programmes déjà existants pour faire une grande partie de ce que nous avons fait, je trouve qu’il est important de connaître les fondations du monde informatique qui nous entoure.

*Fin*