Guelate Seyo 18904103

Rapport Projet : Double Dragon

I / Animation , sprites , décors

Mon but dans ce projet était de faire un scénario dans lequel on retrouve une suite de combats entre le sprite principal et ses différents ennemis , dans l'objectif d'atteindre son trésor à la fin .
L'animation ce compose de sons pour donner vie aux combats , d'une map en trois niveaux différents et différentes interactions avec le décor de la map et les objets .

On retrouve un total de quatres sprites : celui qui représente le personnage principal du jeu et les trois autres qui sont ses ennemis . les mouvements des sprites leur permettent soit de se battre ou d'avancer dans le jeu , mais aussi de grimper pour le personnage principal . Ont distingue deux types de combats celui du corps à corps et du combat à distance dans la deuxième map .

II / Collision , Parallax , génération procédural , explication code

Pour représenter la collision entre les sprites j'ai utilisé différentes manières . Dans le premier cas j'ai implémenté une fonction qui prends en paramètre les rectangles invisible dans lesquels sont introduit les sprites .

Chaque sprite a son rectangle . Lorsque le point x du premier rectangle qui est celui du premier sprite est inférieur à la largeur du second rectangle qui est celui du second sprite . Et lorsque la largeur du premier rectangle est supérieur au point x du second rectangle alors on détecte une collision entre les deux sprites et c'est à partir de la que le combat peut commencer .
Néanmoins j'ai énormément bloqué pour reproduire ce cas avec les autres sprites ennemis du coups j'ai implémenté une autre façon de faire .

Lorsque la position de l'ennemi est égale à la position du sprite principal + une valeur (elle change en fonction des ennemis et de la position des sprites) , on détecte alors une collision .

Concernant l'observation de la map j'ai fait une fonction qui fait coulisser horizontalement la div dans laquelle ce trouve le canvas .

La génération procédurale est présente dans différents cas . Lorsqu'un combat ce finit un ennemi appairait automatiquement ils ne sont pas placés de manières prédéfini .

C’est le même cas pour le rechargement de la vie du personnage principal principal lorsqu’il prends le cœur qui incrémente la valeur de sa santé . De même pour l’apparition des blocs qui viennent compléter le pont dans le dernier niveau .

III / Ressources , améliorations

Pour réaliser ce travail j’ai utiliser en grande partie vos ressources données en cours . Notamment le tilesets pour faire la map . Je me suis basé sur la logique de votre code pour couper les images afin de faire les différents mouvements des sprites .

Cependant à cause d’un temps de blocage important je n’ai pas pu avancer comme il le fallait du coup je me suis obligé de fournir ce rendu afin de ne pas être pénalisé . Les points que je comptais améliorer pour un rendu meilleur sont l’implémentation de certaines fonction qui se répète plusieurs fois dans le code comme celle à chaque chargement d’image . Je voulais reproduire une fonction qui sera appeler chaque image , au lieu de charger chaque image comme je l’ai fait .

De plus à cause du temps je n’ai pas pu changer le nom de certaine variable qui paraissent parfois perturbante dans la lecture du code . et enfin nettoyer au maximum le code par une syntaxe simplifier et claire caractérisé par plusieurs module .