**EPIC BATTLE NOOTERS**



**Sommaire**

1. Source d’inspiration  
   a – Noot-Noot  
   b – Combat tour par tour
2. Objectif du projet
3. Organisation du projet  
   a – Git

b – Répartition des taches

1. Grandes parties du projet

a – Création des personnages

1 – types

2 – attaques

b – interface combat

1 – affichage et choix des attaques

2 – personnage actif et inactif

1. Démonstration

**I)Source d’inspiration**

*a)Noot-Noot*

Le jeu que nous avons voulu créer est inspiré de 2 références.   
Tout d’abord, l’univers que nous exploitons est l’univers de « Noot-Noot », qui est une vidéo avec un petit pingouin (vidéo accessible [ici](https://www.youtube.com/watch?v=bnmAi53H520)). Cette vidéo à beaucoup été reprise pour plusieurs blague et nous l’apprécions.

*b)Combat au tour par tour*  
La deuxième référence est le jeu vidéo Pokémon. Ce jeu consiste à capturer des créatures nommé Pokémons et de les faires combattre au cours de notre aventure.   
  
Ne pouvant pas recréer un jeu aussi complet que Pokémon, on s’est dit que nous allons uniquement faire un jeu basé sur le système de combat du jeu Pokémon.  
  
Le système de combat en question est un combat au tour par tour, c’est-à-dire que les joueurs ne jouent pas simultanément les attaques mais chacun leur tour. Ce système de combat est bien sûr utilisé dans plein d’autres jeux, mais Pokémon est le plus connue et la référence que plus de gens peuvent avoir.

**II)Objectif du projet**

Tout d’abord, on pensait faire des combats de 5 Nooters (nom de nos créatures) contre 5 Nooters. Bien sur les 5 ne seront pas sur le terrain en simultané, cela inclus donc un système de remplacement à la mort de chaque Nooter. Chaque Nooter est une référence culturelle, comme le termiNootor inspiré de Terminator ou encore le Nootkia qui est inspiré du Nokia3310.   
  
  
Une image contenant croquis, Téléphone mobile, dessin, illustration

Description générée automatiquement Une image contenant noir, obscurité

Description générée automatiquement

Nous avons aussi pensé au tout début faire un menu de choix des personnages, mais par manque de temps nous nous contentons de les prédéfinir dans le code. Donc chaque joueur aura ses personnages prédéfinis en brut dans le code. Nous avons donc fait une manière simple à adapter dans le code. Pour que les combats aient un certain sens, chaque Nooter est d’un certain type, et son type sera donc résistant aux attaques de certains types mais sera faible à d’autres attaques.

**III)Organisation du projet**

*a)Git*

*b)Répartition des taches*

Avant de pouvoir commencer correctement à coder, on a fait un diagramme de classe afin de voir toutes les classes que nous devons créer. On a ensuite créé tous ensemble les classes pour avoir la base du projet.   
Une fois toute les classes créés et réunis, nous somme partie avec la répartition suivante :  
Louis s’occupe de l’interface globale de combat, Arthur fait la classe des types qui est complexe, et Théo s’occupe de la création des Nooters.   
Dans cette organisation, on est partie du principe que Arthur et Théo auront fini leurs taches rapidement, afin de faire les fonctionnalités dont Louis a besoin (exemple : fonction récupération du ratio d’attaques selon les types).

**IV)Grandes parties du projet**

*a)Création des personnages*

Pour la création des personnages, cela se fait dans le constructeur de la classe Nooter.



On crée donc un Nooter en lui passant dans le constructeur un QString avec son nom. Avec ce nom, on voulait faire un switch case dans le constructeur pour initialiser le Nooter, mais switch case n’est pas compatible avec les QString. On est donc resté sur un if … else if, dans lequel les personnages sont définis

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

1 – types

Chaque Nooter possède un attribut type. Cet attribut va définir dans quel « famille » se situe le Nooter, et cela va donc définir à quels types d’attaques il va être résistant ou faible.

2 – attaques

Les attaques sont très importantes pour les combats, car il faut bien choisir quelle attaque utiliser selon son type pour faire encore plus de dégâts à l’adversaire.   
Les attaques sont donc gérées dans la classe attack, et est caractérisé par :

- un nom

- une description (pour décrire l’attaque sur l’interface)

- quantité de dégâts infligés par l’attaque

-cout de l’attaque

-temps de rechargement de l’attaque (pour éviter de trop utiliser la même attaque)

-le type de l’attaque

*b)interface combat*

1 – affichage et choix des attaques

Une image contenant texte, neige, arbre, capture d’écran

Description générée automatiquement

Voici donc l’interface que nous avons. Dessus on peut donc voir les 2 Nooters en combats (ici 2 Nooters identiques) avec leurs barres de point de vie en vert et les point d’énergie en bleu.   
Nous avons les 4 attaques du personnage en bas à droite qui peuvent être sélectionné avec les flèches du clavier. Pour valider l’attaque que nous lançons, on appuie sur la touche espace.   
A gauche des attaques, nous avons un espace ou est écrit la description de l’attaque ou se situe le curseur (après que l’on ait commencé à sélectionner une attaque)

2 – personnage actif et inactif

Nous jouons donc à tour de rôle, en sachant que la personne qui joue aura son personnage à gauche (après chaque attaques les personnages inverse leurs positions). Ce changement permet de comprendre qui est en train de jouer son tour.

**V)Démonstration**

Pour que vous puissiez avoir un aperçu du projet, voici donc un lien vers une vidéo que nous avons fait. Cette vidéo sert donc de démonstration :