R5.A.10 Nouveaux paradigmes de base de données

Rapport sur le jeu Punto de Clément MONFORT

mécanique du jeu

Technologie utilisée

Le langage de programmation utilisé pour développer le **Punto** est le **C** avec la librairies **OCML**, une librairie maison basée sur la librairie **OCL** et **CSFML** un binding de la librairie graphique **C++ SFML**, le principale défaut est que je devait presque tout développer de 0 ce qui donne un temps de développement plus long qu'un autre langage, cependant on à un contrôle totale sur ce qui se passe dans l'application et de meilleur performance.

Pour installer les dépendances un script shell est disponible mais il fait pour les distro Debian uniquement, il installe **docker**, **C**, **CSFML**, **make**, **gcc**, **libbson** et les drivers pour **mongodb**, **MySQL** et **MySQL**.

Le langage de programmation utilisé pour le déploiement est un **script shell** utilisant l'outil make (un ***Makefile***) pour déployer les 2 **containers docker MySQL** et **mongodb** ainsi que compiler le programme **Punto.exe**. La base **MySQL** est déjà présente dans l'application mais une commande est fournie dans le **readme** pour la générer à nouveau.

Programmation & algorithme

Le jeu à été divisé en plusieurs sections :

- Une partie contenant les ressources graphiques nécessaires comme les polices d'écritures ou les sons.

- Une partie contenant la scène du jeu c'est à dire les éléments représentant l'interface graphique et le plateau de jeu et son état actuel comme le score maximal et à qui il est attribué.

- Une autre partie représentant le groupe de joueurs, leur decks...

- Et enfin la partie système qui contient la fenêtre de jeu ainsi que les variables importantes comme le nombre de cartes à aligner, le tour de qu'elle joueur…

Les cartes & Plateau de jeu

Les cartes contiennent une variable avec leur couleur ainsi que leur valeur et sont graphiquement des élément appelé **sfShape** d'une forme carré avec un texte qui reflète leur valeur, les bordures de la carte ainsi que le texte porte la couleur qui est attribué à la carte.

Le plateau de jeu est une matrice d'une longueur 6 par une largeur 6 de carte blanche d'une valeur de 0.

Le deck d'un joueur est un tableau de 9 à 18 cartes dépendamment de si ils sont 4 ou 2, elle on une valeur de 1 à 9 et une couleur jaune, rouge, verte ou cyan.

Les Joueurs

Les joueurs sont représentés par un tableau de joueur qui à une taille de 2 ou 4. Les objet joueur sont composé d'un deck, un tableau de carte avec 1 à 9 soit d'une série d'une seule couleur si le groupe est d'une taille de 4 ou de deux série avec des couleur distinct si le groupe est d'une taille de 2, de la taille de ce deck, d'un nom et d'un compteur du nombre de tour jouer par le joueur.

Quand un joueur à fini son tour on passe au prochain joueur du tableau, quand le pointeur atteint la taille du nombre de joueur il retourne à 0 et on incrémente le compteur du nombre de manches.

Gestion des événement

Les événements représentent toutes les entrées réalisées par le joueurs ainsi que les réaction du plateau de jeux relative à ses derniére :

- Toute action à une réaction sonore et graphique.

- Les actions de fin de partie :
    - Appuyer sur echap
    - Alt + f4
    - En cas de combinaison gagnante.

- Indiquer au joueur où est placée la souris comparé au tableau de jeu.
    - Si le souris est est bien dans le plateau de jeu :
        - La carte sera afficher en blanc si la souris est sur la carte.
        - La carte sera affichée en magenta si il est possible d'insérer la carte.
            - Il est possible d'insérer une carte si la carte en dessous est plus à une couleur différente de la carte du joueur et une valeur plus petite que la carte du joueur.
                - et est dans l'une des 4 cases du centre.
                - ou à une carte d'une autre couleur que blanche dans ses 8 cartes adjacentes.
                    - L'événement est enregistré dans la base de données.

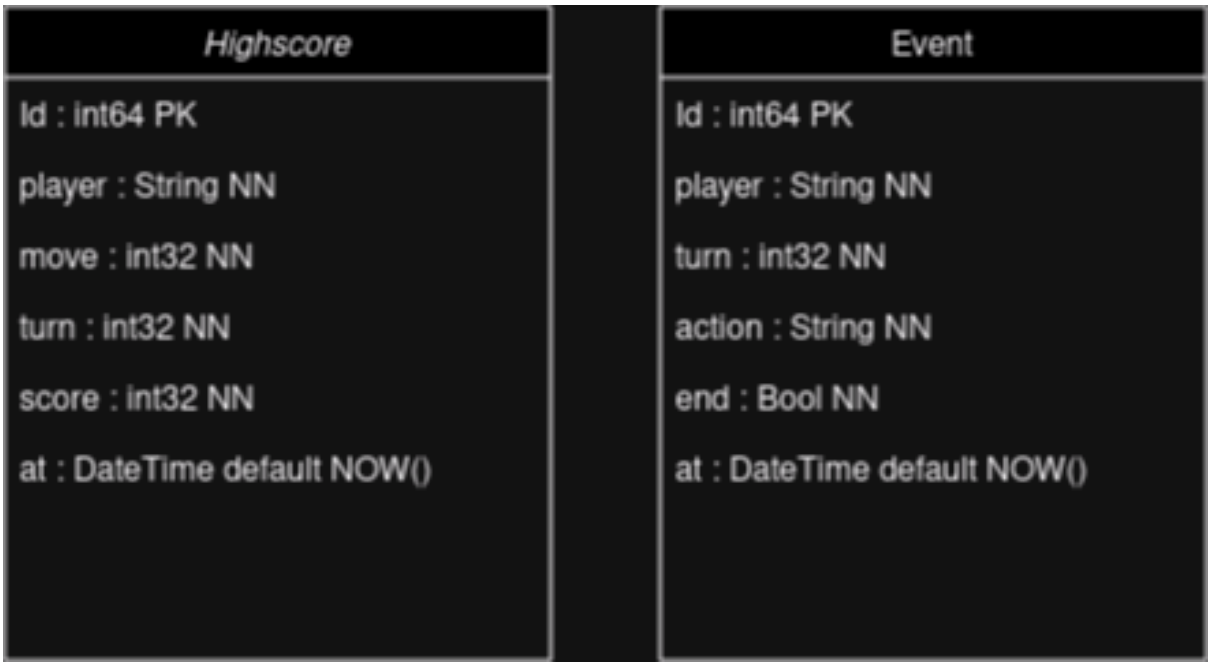
- Trouver le plus grand score
    - Quand un joueur insère une carte on récupère ses coordonnées et on vérifie la ligne x ainsi que la colonne y pour vérifier s'il y a un meilleur score.
        - Si les coordonnées suivent l'un des vecteur des diagonales elles sont aussi vérifiées.
            - Si le plus grand score trouvé a une suite plus longue que le meilleur score actuel ou la même longueur mais un score plus petit il devient le meilleur score.
                - Si le meilleur score à une longueur égale à la longueur maximum on envoie un événement de fin de partie.
                    - Le score est enregistré dans la base de données.

Interface Utilisateur

Les joueurs disposent d'un H.U.D (Head Up display aussi appelé affichage tête haute) qui indique le tour du joueur ainsi que la carte qu'il doit jouer, combien il lui reste de carte dans son deck, le nombre de mouvements, tours et manches effectué.

Bases de données

Les projets possèdent 3 bases de données, les joueurs doivent en choisir une en début de partie et ne peuvent pas en changer avant la fin de la partie en relançant le jeu. Il y tout d'abord une base de données relationnelle **MySQL** dans un conteneur docker à l'adresse *127.0.1.1:21020/tcp* et une base **NoSQL** mongodb dans un conteneur docker à l'adresse *127.0.1.1:21000/tcp*. Les deux sont construits depuis un Dockerfile puis déployés avec un docker compose et enfin une base embarquée MySQL dans le fichier *data/db/punto.db*, les requête à ces dernière sont directement fait par une connection direct depuis l'application Punto.exe par le biais de driver et non par une api.



Elle implémente toutes les trois les deux même table :

- Event qui sert de backlogs de toutes les parties de Punto (les mouvements des joueurs) avec leur noms, le tour à laquelle l'action à été effectuer un texte expliquant l'action et un booléen indiquant si l'action à terminé la partie.

- Highscore qui contient tous les meilleurs scores de chaque partie, avec le nom du joueur, le nombre de tour totales de la partie, le nombre de mouvements qu'a fait le joueur et son score.

- Toutes les tables ont un identifiant propre et comportent la date et le temps de l'insertion de l'élément et Les insertion mongodb passe par un fichier bson avant d'être convertie en json.