tCHu - bonus

1. Ajout titre du jeu dans la fenêtre

2. Ajout de page de démarrage, permettant de modifier différents paramètres :
    a. Client : choix du pseudo, adresse et port du serveur
    b. Server : choix du pseudo

    Cette page utilise des boutons qui sont associés à des TextInputDialog ou l'on peut renseigner nos choix concernant les différents paramètres, ou laisser les choix par défaut. Si le texte renseigné est vide (length =< 0), ou si le bouton annuler a été sélectionné, l'ancien texte est conservé.
    Nous avons été contraints de définir Platform.setImplicitExit(false); pour permettre à la page de démarrage de se fermer une fois la partie démarrée sans arrêter tout le fil d'exécution javafX. De plus nous faisons appel à la méthode setOnCloseRequest() exécutant System.exit(0) afin de stopper le programme lorsque l'on ferme la fenêtre.

3. Laisser au client le choix de son pseudo (au lieu que le serveur gère tous les pseudos)

    Attention, la communication en réseau a été modifiée et il se peut que notre jeu ne soit plus compatible avec les applications des autres groupes.

    Nous renseignons le choix du pseudo (choisi grâce à la page de démarrage) en paramètre de RemotePlayerClient. Ce pseudo est renvoyé lorsque le client reçoit la nouvelle commande CHOOSE_USERNAME, définie dans MessageId. Le RemotePlayerProxy qui a envoyé cette commande attend attend la réception de la réponse. Cette réponse modifie la Map de pseudo qui est renseigné à la méthode play() de Game définie dans ServerMain.
    Nous avons choisi de ne pas modifier l'interface Player, car le choix du pseudo est inutile dans les autres implémentations de cette interface.

4. Ajout animation lors du survol avec la souris du deck de cartes quand c'est son tour.

    Il a fallu rajouter du code dans decks.css

5. Ajout d'un indicateur concernant les stations de départ et d'arrivée associées aux tickets sélectionnés par le joueur

    Il a fallu récupérer les coordonnées de chaque station, elles ont été renseignées dans map.css, en s'inspirant de ce qui été utilisé pour les tronçons, associées avec leurs id (le nom n'est pas unique et en plus les accents auraient posé problèmes). Ensuite il a fallu rajouter des méthodes à la classe Ticket permettant de récupérer les stations. Dans MapViewCreator, nous avons rajouté un ChangeListener pour agir dès que la main du joueur a de nouveaux tickets, un cercle est rajouté à la carte, avec des propriétés suffisamment similaires à celles des tronçons.

6. Ajout possibilité de jouer à 3 : pour activer, il faut décommenter PLAYER_3 dans PlayerId.

   Il a fallu rajouter des éléments dans les fichiers css (dans maps et color) pour PLAYER_3 (similaire aux autres joueurs). Rajouter un joueur dans PlayerId. Dans ObservableGameState, pour correspondre à une règle pour les parties à plusieurs joueurs, permettre à un joueur différent de s'emparer du deuxième tronçon d'une route double. Et finalement il a fallu modifier dans ServerMain afin d'autoriser tous les clients.

7. Ajout de spectateurs, ils doivent rejoindre la partie une fois qu'elle a commencé, ils voient l'état du joueur courant (change).

   Pour l'initialiser, il faut faire appel à GraphicalSpectatorAdapter qui prend comme argument un boolean indiquant si l'on souhaite afficher les éléments "privés", après il faut initialiser le spectateur avec initPlayer puis le rajouter dans la liste de spectateurs qui a été passée a Game.play
   Il a fallu créer un GraphicalSpectator et un GraphicalSpectatorAdapter, également SpecatatorMain en tant que client. Et il a fallu modifier Game afin de mettre à jour (receiveInfo et updateState) les spectateurs. ServerMain autorise la connexion de spectateurs sur le port 5109.

8. Ajout de la possibilité de voir combien de points un ticket nous rapporte actuellement, en cliquant dessus.

   Dans PlayerState, rajout d'une méthode ticketPoints prenant comme argument le ticket pour lequel on souhaite connaître les points qu'il rapporte. Ensuite il a fallu rajouter un PlayerState observable dans ObservableGameState. Et finalement dans DecksViewCreator, un listener pour écouter si le joueur clique sur un ticket et afficher le score (avec le texte défini dans StringsFr), mais également mettre à jour le score lorsque le jour fait une action (par exemple prendre une nouvelle route).