# Rapport de projet

## Description

Le principe du projet est de coder le jeu de carte Skyjo en utilisant java avec une interfaces graphique à l’aide de java fxml en respectant le pattern MVC et en implémentent un observateur observé.

Skyjo est un jeu dont l’objectif est d’obtenir le moins de points possibles en inversant et échangeant des cartes. Chaque joueur possède 12 cartes dont la valeur varie de -2 à 12. La description des règles du jeu est disponible via <https://www.regledujeu.fr/regle-du-skyjo>.

## Bibliothèques utiliser

Nous avons utilisé la bibliothèque openjfx et 2 librairies personnalisées Message\_Protocol\_SKYJO et model\_skyjo que l’on a écrites.

## Règle du jeu

### Début d’une partie

Lorsqu’une manche démarre, deux cartes ont été retournées pour chaque joueur. Le joueur avec le plus de points commence. Si les deux joueurs ont le même nombre de points, choisissez au hasard.

### Déroulement d’une manche

Le premier joueur doit choisir entre les deux actions suivantes :

* Piocher la défausse
* Retourner une carte de la pioche.

Si le joueur choisit de piocher une carte de la défausse, il doit échanger la carte défaussée avec une de ses cartes. Il peut l’échanger contre n’importe laquelle de ses cartes, qu’elle soit face visible ou face cachée. Lorsqu’il procède à cet échange, il rend la carte échangée face visible sur la pile de défausse.

Si le joueur choisit de retourner une carte de la pioche, il peut reposer la carte tirée sur la pile de défausse (face visible) et retourner une de ses cartes dont la valeur était cachée OU il peut échanger la carte piochée avec n’importe laquelle de ses cartes. Lorsqu’il procède à cet échange, le joueur dépose son ancienne carte face visible sur la défausse. Après cette action, le nombre de points du joueur est mis à jour et le joueur suivant peut jouer.

### Fin d’une manche

Une manche se décompose en plusieurs tours. Chaque tour permettant aux deux joueurs d’effectuer leurs actions. Une manche se termine lorsqu’un des deux joueurs a retourné toutes ses cartes. Si le premier joueur du tour a retourné toutes ses cartes, le second joueur doit encore effectuer son action. C’est seulement à la fin de ce tour que la manche est terminée.

### Fin du jeu

Si le joueur qui a terminé en premier a un score plus grand ou égal au score de l’autre joueur et si son score est positif, une pénalité lui est attribuée : ses points sont doublés.

## Description des classes

Le projet a été séparé en 3 parties pour une meilleur visualisation : la partie MODEL, la partie SERVEUR et la partie CLIENT.

* Pour la partie MODEL
  + Ce package reprend l’ensemble des classes, et donc des méthodes nécessaire au bon déroulement du jeu. Le jeu sera stocké sur le serveur pour une meilleur fonctionnalité du jeu.
* Pour la partie SERVEUR
  + Ce package reprend les classes lié a la fonctionnalité du serveur.   
    C’est ce package qui va servir d’écrous centrale au projet, car c’est là qu’on se connectera depuis le client, mais là aussi où l’on récupérera les informations souhaités.  
    Pour que cela se déroule de la meilleur des manières, une libraire à été ajouté par Juste.
* Pour la partie CLIENT
  + Ce package reprend les classes lié à la fonctionnalité du client. C’est la que les utilisateur vont pouvoir se connecter et ainsi faire la liaison avec le serveur.

## Répartition des taches

Chaque personnes présentes dans le projet a émis s’est idées, et participé à l’élaboration de l’ensemble des packages. Certaines personnes ont notamment participé davantage sur certaines partie.

Wattier Alexandre : Model / View / Rapport Projet

El Fahsi Abdessalam : View / Model / Doc

Rolecki Maciej : Client / Serveur / Model

Viho Juste Midedji : Client / Serveur / View / Model / Librairie

Gardeur Nicolas : Model / Client / Doc