Classes et structures de donnés

Player

{

String name ;

Int score ;

Int status ; // 0 = afk, 1 = ready for game

List<Card> cards ;

}

MessageInfo

{

Int code ; // code du message

String description // description du message

}

Team

{

Int id ;

String nom ;

Player player1 ;

Player player2 ;

Int score\_actuel ;

}

Bet

{

Enum Couleur  couleur; // couleur de l’atout parrié

Int point ; // nombre de point parié

Player player ;

Team team;

}

Protocol JCoinch :

Commun

cmd : envoi de message

PacketMessage

Arg1 : int code

Arg2 : Object message

Client

1 Connexion

Cmd1 : athentification

PacketAuth

Arg1 : String player\_name

Cmd2 : wait for a game

// le serveur répond en attente de x joueurs

// ou ready

PacketWaitForAGame

Cmd3 : bet on a color

PacketBetColor

Arg1 : Couleur couleur;

Arg2 : int points  // les points parié sur l’atout, si 0 = passer

Cmd4 : LookCards

Cmd4 : Play a card

PacketPlayCard

Arg1 : Card card // envoi la carte à posé

Server

Cmd1 : envoi les carte que possède le joueur

PacketCardInfo

Arg1 : List<Card> cards

Cmd2 : envoi le dernier pari valide

PacketBetInfo

Arg1 : Bet bet ; // dernier pari

Description du protocol

Connexion :

1. Client> se connecte au serveur sur le port 42442
2. Serveur> envoi un message (class Message )de confirmation de connexion
3. Client> envoi son nom
4. Serveur> envoi confirmation nom valide
5. Client> envoi qu’il est prêt pour une nouvelle partie
6. Serveur> confirme son inscription à la stack d’attente de partie et lui précise des informations (ex : nombre de joueur déjà présent, son équipe)
7. Serveur> envoi la message la partie commence

Jeu :

# Le jeu commence

# préparation de la manche

1. Serveur> envoi liste des cartes du joueur
2. Serveur> envoi message de demande de pari et lui donne des informations (les couleurs déjà pariées par tel joueurs)
3. Client> pari sur une couleur avec infos (score de pari)
4. Serveur> confirme et envoi à tous les joueurs les informations du pari
5. Serveur> valide le paris

# la manche commence

1. Serveur> envoi liste de carte et demande de jouer une carte
2. Client> joue une carte
3. Serveur> valide carte ou rejette et demande de rejouer
4. Serveur> notifie la carte jouée
5. Serveur> notifie équipe gagne la stack et donne les points

# Quand il n’y a plus de carte le jeu fini

1. Serveur> notifie fin de la manche et donne les infos

# seconde manche

# fin du jeu 1000 points

1. Serveur> notifie la fin du jeu et annonce l’équipe gagnante