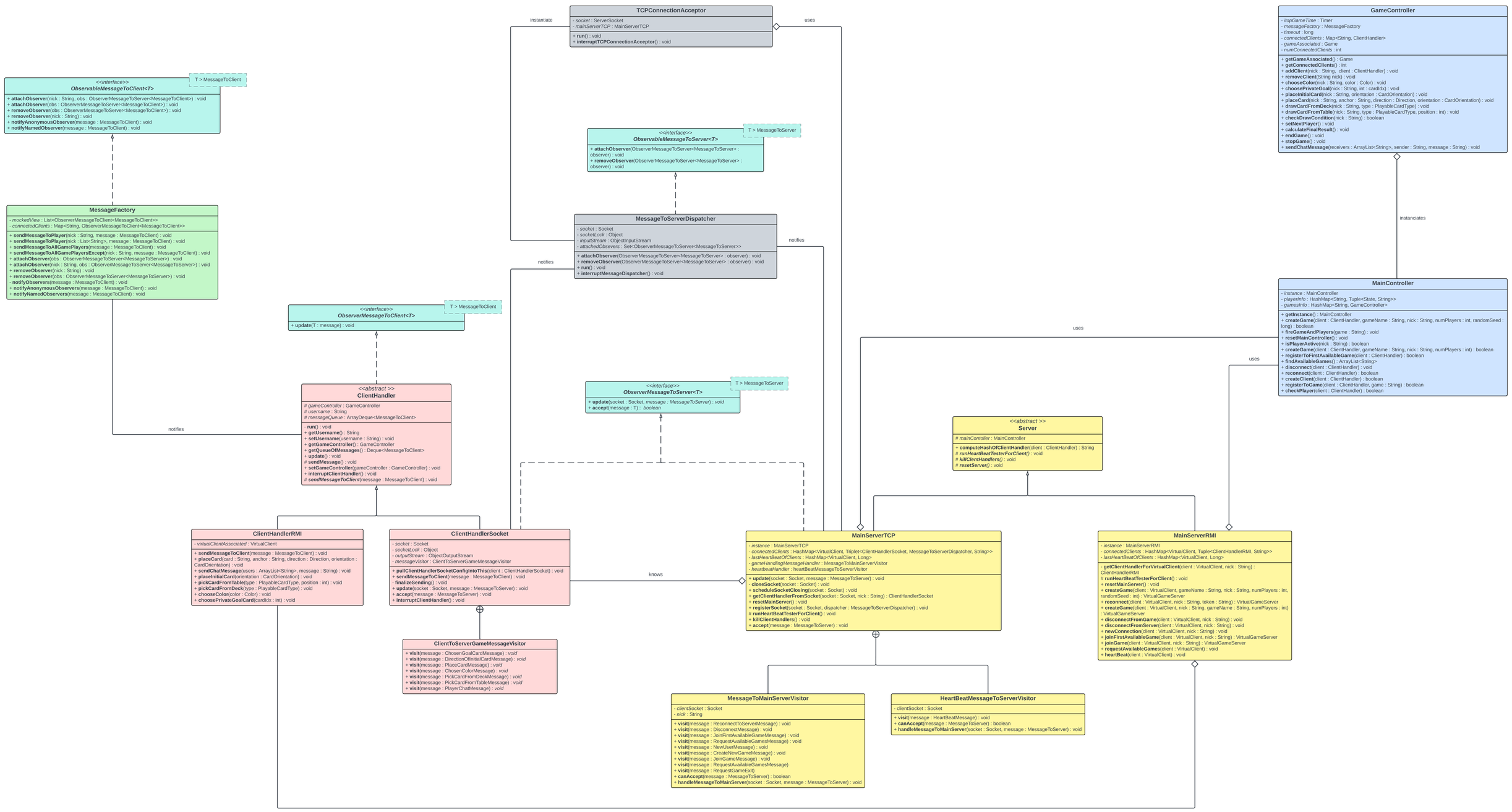
TCPConnectionAcceptor
- socket : ServerSocket
- mainServerTCP : MainServerTCP
+ run() : void
+ interruptTCPConnectionAcceptor() : void
instantiate
uses
GameController
- itopGameTime : Timer
- messageFactory : MessageFactory
- timeout : long
- connectedClients : Map<String, ClientHandler>
- gameAssociated : Game
- numConnectedClients : int
+ getGameAssociated() : Game
+ getConnectedClients() : int
+ addClient(nick : String, client : ClientHandler) : void
+ removeClient(String nick) : void
+ chooseColor(nick : String, color : Color) : void
+ choosePrivateGoal(nick : String, int : cardIdx) : void
+ placeInitialCard(nick : String, orientation : CardOrientation) : void
+ placeCard(nick : String, anchor : String, direction : Direction, orientation : CardOrientation) : void
+ drawCardFromDeck(nick : String, type : PlayableCardType) : void
+ drawCardFromTable(nick : String, type : PlayableCardType, position : int) : void
+ checkDrawCondition(nick : String) : boolean
+ setNextPlayer() : void
+ calculateFinalResult() : void
+ endGame() : void
+ stopGame() : void
+ sendChatMessage(receivers : ArrayList<String>, sender : String, message : String) : void
<<interface>>
ObservableMessageToClient<T>
T > MessageToClient
+ attachObserver(nick : String, obs : ObserverMessageToServer<MessageToClient>) : void
+ attachObserver(obs : ObserverMessageToServer<MessageToClient>) : void
+ removeObserver(obs : ObserverMessageToServer<MessageToClient>) : void
+ removeObserver(nick : String) : void
+ notifyAnonymousObserver(message : MessageToClient) : void
+ notifyNamedObserver(message : MessageToClient) : void
<<interface>>
ObservableMessageToServer<T>
T > MessageToServer
+ attachObserver(ObserverMessageToServer<MessageToServer> : observer) : void
+ removeObserver(ObserverMessageToServer<MessageToServer> : observer) : void
MessageFactory
- mockedView : List<ObserverMessageToClient<MessageToClient>>
- connectedClients : Map<String, ObserverMessageToClient<MessageToClient>>
+ sendMessageToPlayer(nick : String, message : MessageToClient) : void
+ sendMessageToPlayer(nick : List<String>, message : MessageToClient) : void
+ sendMessageToAllGamePlayers(message : MessageToClient) : void
+ sendMessageToAllGamePlayersExcept(nick : String, message : MessageToClient) : void
+ attachObserver(obs : ObserverMessageToServer<MessageToServer>) : void
+ attachObserver(nick : String, obs : ObserverMessageToServer<MessageToServer>) : void
+ removeObserver(nick : String) : void
+ removeObserver(obs : ObserverMessageToServer<MessageToServer>) : void
+ notifyObservers(message : MessageToClient) : void
+ notifyAnonymousObservers(message : MessageToClient) : void
+ notifyNamedObservers(message : MessageToClient) : void
MessageToServerDispatcher
- socket : Socket
- socketLock : Object
- inputStream : ObjectInputStream
- attachedObsevers : Set<ObserverMessageToServer<MessageToServer>>
+ attachObserver(ObserverMessageToServer<MessageToServer> : observer) : void
+ removeObserver(ObserverMessageToServer<MessageToServer> : observer) : void
+ run() : void
+ interruptMessageDispatcher() : void
notifies
notifies
instanciates
MainController
- instance : MainController
- playerInfo : HashMap<String, Tuple<State, String>>
- gamesInfo : HashMap<String, GameController>
+ getInstance() : MainController
+ createGame(client : ClientHandler, gameName : String, nick : String, numPlayers : int, randomSeed : long) : boolean
+ fireGameAndPlayers(game : String) : void
+ resetMainController() : void
+ isPlayerActive(nick : String) : boolean
+ createGame(client : ClientHandler, gameName : String, nick : String, numPlayers : int) : boolean
+ registerToFirstAvailableGame(client : ClientHandler) : boolean
+ findAvailableGames() : ArrayList<String>
+ disconnect(client : ClientHandler) : void
+ reconnect(client : ClientHandler) : boolean
+ createClient(client : ClientHandler) : boolean
+ registerToGame(client : ClientHandler, game : String) : boolean
+ checkPlayer(client : ClientHandler) : boolean
uses
uses
<<interface>>
ObserverMessageToClient<T>
T > MessageToClient
+ update(T : message) : void
<<abstract >>
ClientHandler
gameController : GameController
username : String
messageQueue : ArrayDeque<MessageToClient>
- run() : void
+ getUsername() : String
+ setUsername(username : String) : void
+ getGameController() : GameController
+ getQueueOfMessages() : Deque<MessageToClient>
+ update() : void
# sendMessage() : void
+ setGameController(gameController : GameController) : void
+ interruptClientHandler() : void
# sendMessageToClient(message : MessageToClient) : void
notifies
<<interface>>
ObserverMessageToServer<T>
T > MessageToServer
+ update(socket : Socket, message : MessageToServer) : void
+ accept(message : T) : boolean
<<abstract >>
Server
mainContoller : MainController
+ computeHashOfClientHandler(client : ClientHandler) : String
# runHeartBeatTesterForClient() : void
# killClientHandlers() : void
# resetServer() : void
ClientHandlerRMI
- virtualClientAssociated : VirtualClient
+ sendMessageToClient(message : MessageToClient) : void
+ placeCard(card : String, anchor : String, direction : Direction, orientation : CardOrientation) : void
+ sendChatMessage(users : ArrayList<String>, message : String) : void
+ placeInitialCard(orientation : CardOrientation) : void
+ pickCardFromTable(type : PlayableCardType, position : int) : void
+ pickCardFromDeck(type : PlayableCardType) : void
+ chooseColor(color : Color) : void
+ choosePrivateGoalCard(cardIdx : int) : void
ClientHandlerSocket
- socket : Socket
- socketLock : Object
- outputStream : ObjectOutputStream
- messageVisitor : ClientToServerGameMessageVisitor
+ pullClientHandlerSocketConfigIntoThis(client : ClientHandlerSocket) : void
+ sendMessageToClient(message : MessageToClient) : void
- finalizeSending() : void
+ update(socket : Socket, message : MessageToServer) : void
+ accept(message : MessageToServer) : void
+ interruptClientHandler() : void
knows
MainServerTCP
- instance : MainServerTCP
- connectedClients : HashMap<VirtualClient, Triplet<ClientHandlerSocket, MessageToServerDispatcher, String>>
- lastHeartBeatOfClients : HashMap<VirtualClient, Long>
- gameHandlingMessageHandler : MessageToMainServerVisitor
- heartbeatHandler : heartBeatMessageToServerVisitor
+ update(socket : Socket, message : MessageToServer) : void
- closeSocket(socket : Socket) : void
+ scheduleSocketClosing(socket : Socket) : void
+ getClientHandlerFromSocket(socket : Socket, nick : String) : ClientHandlerSocket
+ resetMainServer() : void
+ registerSocket(socket : Socket, dispatcher : MessageToServerDispatcher) : void
# runHeartBeatTesterForClient() : void
+ killClientHandlers() : void
+ accept(message : MessageToServer) : void
MainServerRMI
- instance : MainServerRMI
- connectedClients : HashMap<VirtualClient, Tuple<ClientHandlerRMI, String>>
- lastHeartBeatOfClients : HashMap<VirtualClient, Long>
- getClientHandlerForVirtualClient(client : VirtualClient, nick : String) : ClientHandlerRMI
runHeartBeatTesterForClient() : void
+ resetMainServer() : void
+ createGame(client : VirtualClient, gameName : String, nick : String, numPlayers : int, randomSeed : int) : VirtualGameServer
+ reconnect(client : VirtualClient, nick : String, token : String) : VirtualGameServer
+ createGame(client : VirtualClient, nick : String, gameName : String, numPlayers : int) : VirtualGameServer
+ disconnectFromGame(client : VirtualClient, nick : String) : void
+ disconnectFromServer(client : VirtualClient, nick : String) : void
+ newConnection(client : VirtualClient, nick : String) : void
+ joinFirstAvailableGame(client : VirtualClient, nick : String) : VirtualGameServer
+ joinGame(client : VirtualClient, nick : String) : VirtualGameServer
+ requestAvailableGames(client : VirtualClient) : void
+ heartBeat(client : VirtualClient) : void
ClientToServerGameMessageVisitor
+ visit(message : ChosenGoalCardMessage) : void
+ visit(message : DirectionOfInitialCardMessage) : void
+ visit(message : PlaceCardMessage) : void
+ visit(message : ChosenColorMessage) : void
+ visit(message : PickCardFromDeckMessage) : void
+ visit(message : PickCardFromTableMessage) : void
+ visit(message : PlayerChatMessage) : void
MessageToMainServerVisitor
- clientSocket : Socket
- nick : String
+ visit(message : ReconnectToServerMessage) : void
+ visit(message : DisconnectMessage) : void
+ visit(message : JoinFirstAvailableGameMessage) : void
+ visit(message : RequestAvailableGamesMessage) : void
+ visit(message : NewUserMessage) : void
+ visit(message : CreateNewGameMessage) : void
+ visit(message : JoinGameMessage) : void
+ visit(message : RequestAvailableGamesMessage)
+ visit(message : RequestGameExit)
+ canAccept(message : MessageToServer) : boolean
+ handleMessageToMainServer(socket : Socket, message : MessageToServer) : void
HeartBeatMessageToServerVisitor
- clientSocket : Socket
+ visit(message : HeartBeatMessage) : void
+ canAccept(message : MessageToServer) : boolean
+ handleMessageToMainServer(socket : Socket, message : MessageToServer) : void
knows