POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Prova Finale di Ingegneria del Software

Anno Accademico 2023-2024

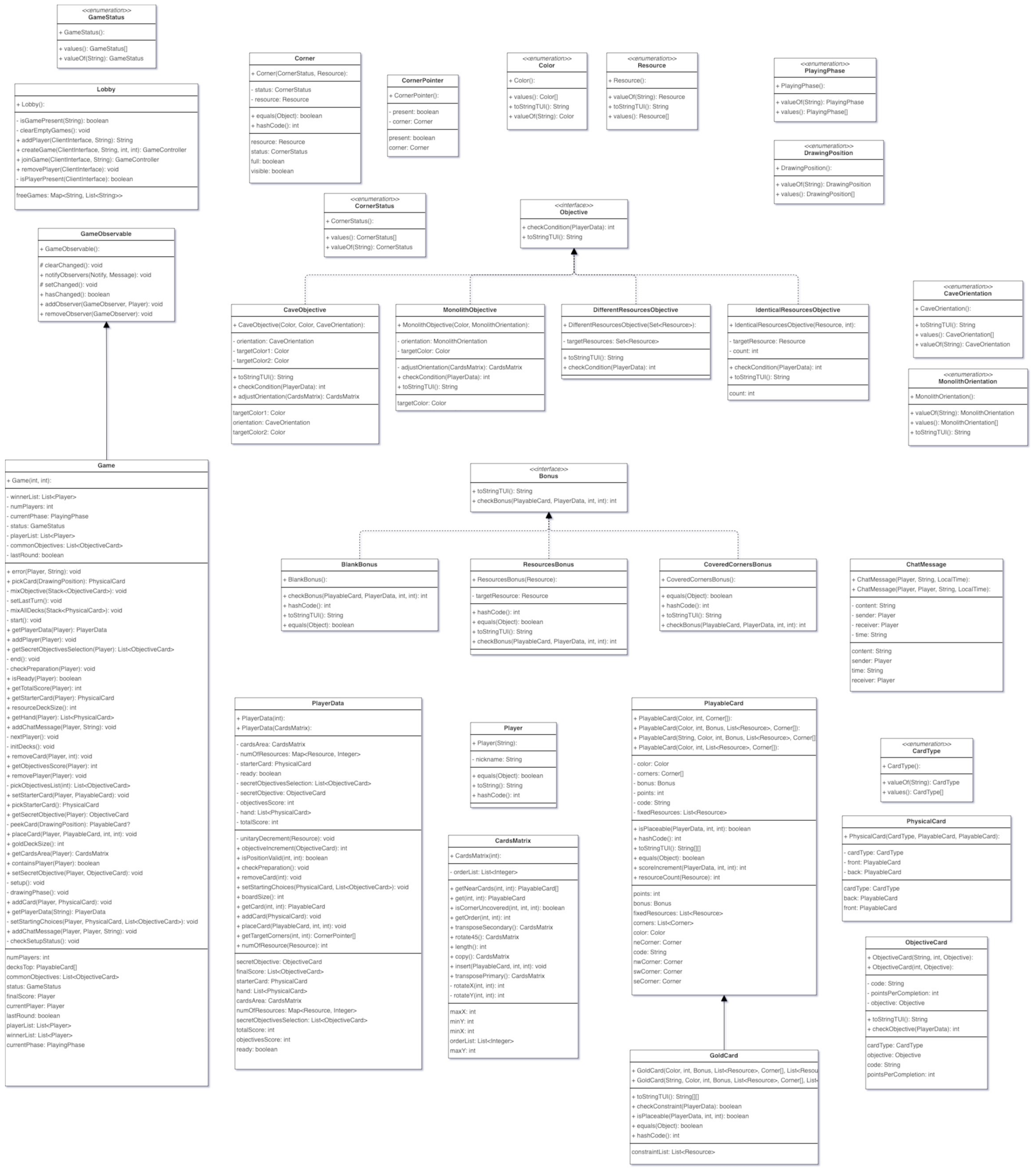
Codex Naturalis

Annalisa Pellicari – Pietro Ravasi – Francesco Tagliabue – Giovanni Versiglioni

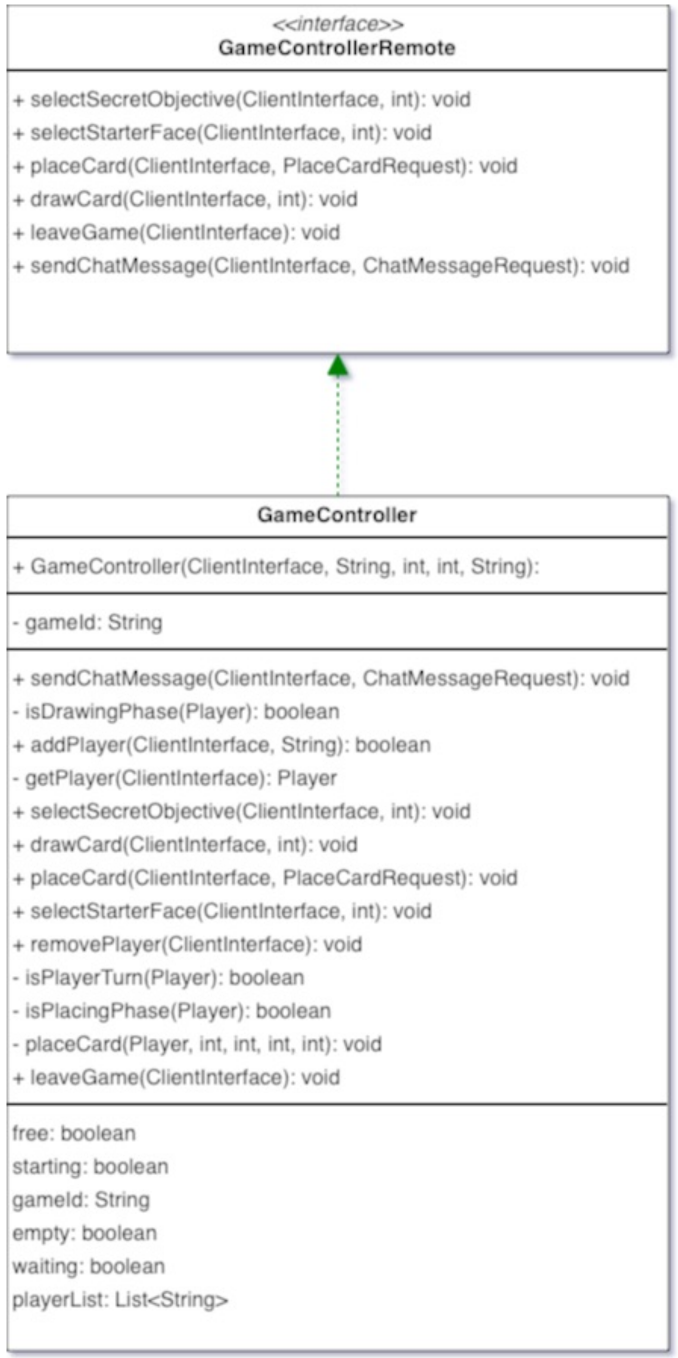
Funzionalità Avanzate Implementate

- Partite Multiple: Realizzare il server in modo che possa gestire più partite contemporaneamente. Ai fini dell'implementazione di questa funzionalità aggiuntiva, le regole precedentemente specificate in merito alla creazione delle partite possono essere modificate in base alle esigenze implementative o di interfaccia utente.
- Resilienza alle Disconnessioni: I giocatori disconnessi a seguito della caduta della rete o del crash del client, possono ricollegarsi e continuare la partita. Mentre un giocatore non è collegato, il gioco continua saltando i turni di quel giocatore. Se rimane attivo un solo giocatore, il gioco viene sospeso a meno che non si ricolleghi almeno un altro giocatore, oppure scade un timer, che decreta la vittoria dell'unico giocatore rimasto connesso.
- Chat: Client e server devono offrire la possibilità ai giocatori coinvolti in una partita di chattare tra di loro, inviando messaggi (testuali) indirizzati a tutti i giocatori della partita o a un singolo giocatore.
- Persistenza: Fare in modo che il server salvi periodicamente lo stato della partita su disco, in modo che l'esecuzione possa riprendere da dove si è interrotta, anche a seguito del crash del server stesso. Per riprendere una partita, i giocatori si dovranno ricollegare al server utilizzando gli stessi nickname, una volta che questo sia tornato attivo. Si assume che il disco costituisca una memoria totalmente affidabile.

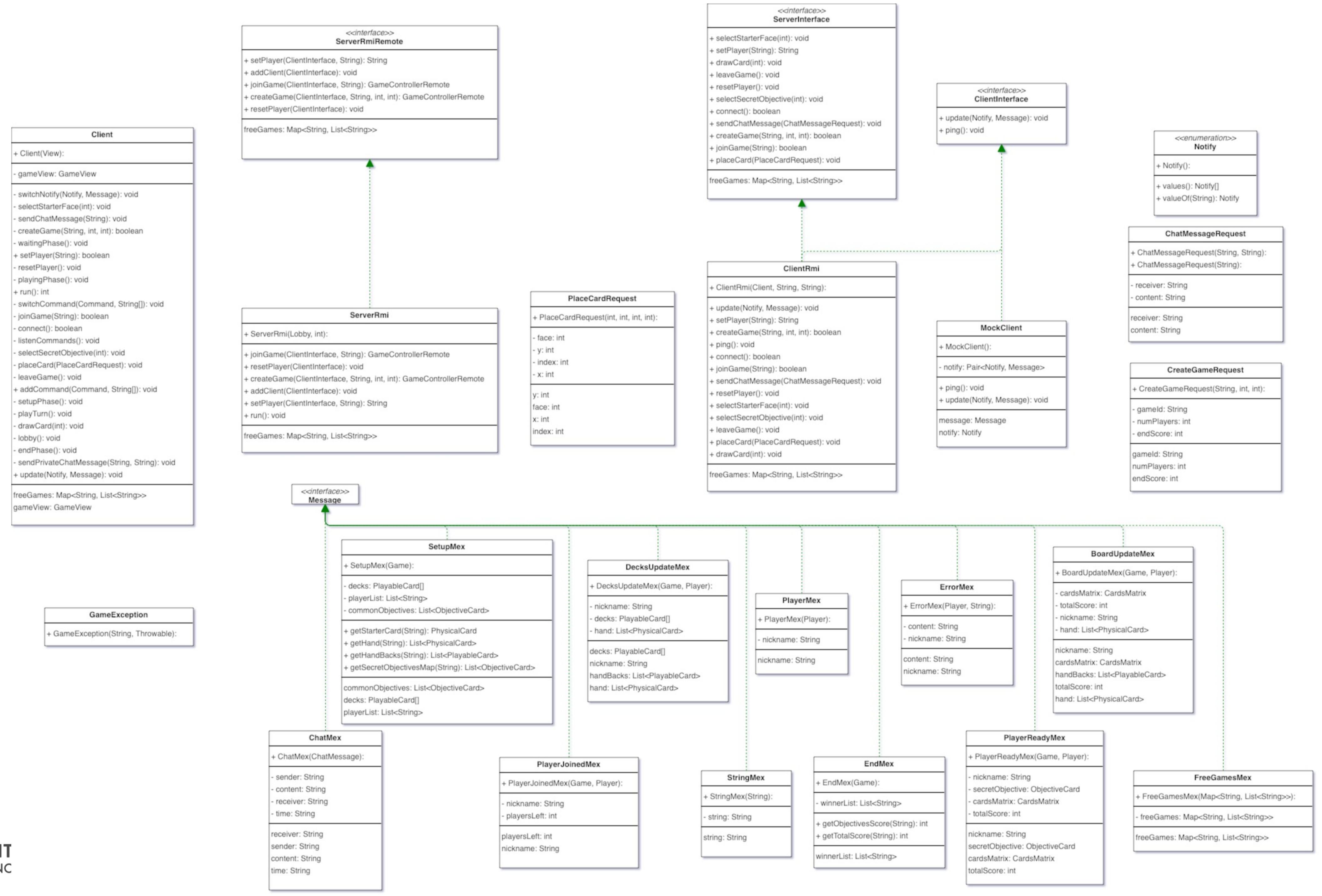
UML: Model


<<enumeration>> GameStatus
+ GameStatus():
+ values(): GameStatus[]
+ valueOf(String): GameStatus
Lobby
+ Lobby():
- isGamePresent(String): boolean
- clearEmptyGames(): void
+ addPlayer(ClientInterface, String): String
+ createGame(ClientInterface, String, int, int): GameController
+ joinGame(ClientInterface, String): GameController
+ removePlayer(ClientInterface): void
- isPlayerPresent(ClientInterface): boolean
freeGames: Map<String, List<String>>
GameObservable
+ GameObservable():
# clearChanged(): void
+ notifyObservers(Notify, Message): void
# setChanged(): void
+ hasChanged(): boolean
+ addObserver(GameObserver, Player): void
+ removeObserver(GameObserver): void
Game
+ Game(int, int):
- winnerList: List<Player>
- numPlayers: int
- currentPhase: PlayingPhase
- status: GameStatus
- playerList: List<Player>
- commonObjectives: List<ObjectiveCard>
- lastRound: boolean
+ error(Player, String): void
+ pickCard(DrawingPosition): PhysicalCard
- mixObjective(Stack<ObjectiveCard>): void
- setLastTurn(): void
- mixAllDecks(Stack<PhysicalCard>): void
- start(): void
+ getPlayerData(Player): PlayerData
+ addPlayer(Player): void
+ getSecretObjectivesSelection(Player): List<ObjectiveCard>
- end(): void
- checkPreparation(Player): void
+ isReady(Player): boolean
+ getTotalScore(Player): int
+ getStarterCard(Player): PhysicalCard
+ resourceDeckSize(): int
+ getHand(Player): List<PhysicalCard>
+ addChatMessage(Player, String): void
- nextPlayer(): void
- initDecks(): void
+ removeCard(Player, int): void
+ getObjectivesScore(Player): int
+ removePlayer(Player): void
- pickObjectivesList(int): List<ObjectiveCard>
+ setStarterCard(Player, PlayableCard): void
+ pickStarterCard(): PhysicalCard
+ getSecretObjective(Player): ObjectiveCard
- peekCard(DrawingPosition): PlayableCard?
+ placeCard(Player, PlayableCard, int, int): void
+ goldDeckSize(): int
+ getCardsArea(Player): CardsMatrix
+ containsPlayer(Player): boolean
+ setSecretObjective(Player, ObjectiveCard): void
- setup(): void
- drawingPhase(): void
+ addCard(Player, PhysicalCard): void
+ getPlayerData(String): PlayerData
- setStartingChoices(Player, PhysicalCard, List<ObjectiveCard>): void
+ addChatMessage(Player, Player, String): void
- checkSetupStatus(): void
numPlayers: int
decksTop: PlayableCard[]
commonObjectives: List<ObjectiveCard>
status: GameStatus
finalScore: Player
currentPlayer: Player
lastRound: boolean
playerList: List<Player>
winnerList: List<Player>
currentPhase: PlayingPhase
Corner
+ Corner(CornerStatus, Resource):
- status: CornerStatus
- resource: Resource
+ equals(Object): boolean
+ hashCode(): int
resource: Resource
status: CornerStatus
full: boolean
visible: boolean
CornerPointer
+ CornerPointer():
- present: boolean
- corner: Corner
present: boolean
corner: Corner
<<enumeration>> CornerStatus
+ CornerStatus():
+ values(): CornerStatus[]
+ valueOf(String): CornerStatus
<<enumeration>> Color
+ Color():
+ values(): Color[]
+ toStringTUI(): String
+ valueOf(String): Color
<<enumeration>> Resource
+ Resource():
+ valueOf(String): Resource
+ toStringTUI(): String
+ values(): Resource[]
<<enumeration>> PlayingPhase
+ PlayingPhase():
+ valueOf(String): PlayingPhase
+ values(): PlayingPhase[]
<<enumeration>> DrawingPosition
+ DrawingPosition():
+ valueOf(String): DrawingPosition
+ values(): DrawingPosition[]
<<interface>> Objective
+ checkCondition(PlayerData): int
+ toStringTUI(): String
CaveObjective
+ CaveObjective(Color, Color, CaveOrientation):
- orientation: CaveOrientation
- targetColor1: Color
- targetColor2: Color
+ toStringTUI(): String
+ checkCondition(PlayerData): int
+ adjustOrientation(CardsMatrix): CardsMatrix
targetColor1: Color
orientation: CaveOrientation
targetColor2: Color
MonolithObjective
+ MonolithObjective(Color, MonolithOrientation):
- orientation: MonolithOrientation
- targetColor: Color
- adjustOrientation(CardsMatrix): CardsMatrix
+ checkCondition(PlayerData): int
+ toStringTUI(): String
targetColor: Color
DifferentResourcesObjective
+ DifferentResourcesObjective(Set<Resource>):
- targetResources: Set<Resource>
+ toStringTUI(): String
+ checkCondition(PlayerData): int
IdenticalResourcesObjective
+ IdenticalResourcesObjective(Resource, int):
- targetResource: Resource
- count: int
+ checkCondition(PlayerData): int
+ toStringTUI(): String
count: int
<<enumeration>> CaveOrientation
+ CaveOrientation():
+ toStringTUI(): String
+ values(): CaveOrientation[]
+ valueOf(String): CaveOrientation
<<enumeration>> MonolithOrientation
+ MonolithOrientation():
+ valueOf(String): MonolithOrientation
+ values(): MonolithOrientation[]
+ toStringTUI(): String
<<interface>> Bonus
+ toStringTUI(): String
+ checkBonus(PlayableCard, PlayerData, int, int): int
BlankBonus
+ BlankBonus():
+ checkBonus(PlayableCard, PlayerData, int, int): int
+ hashCode(): int
+ toStringTUI(): String
+ equals(Object): boolean
ResourcesBonus
+ ResourcesBonus(Resource):
- targetResource: Resource
+ hashCode(): int
+ equals(Object): boolean
+ toStringTUI(): String
+ checkBonus(PlayableCard, PlayerData, int, int): int
CoveredCornersBonus
+ CoveredCornersBonus():
+ equals(Object): boolean
+ hashCode(): int
+ toStringTUI(): String
+ checkBonus(PlayableCard, PlayerData, int, int): int
ChatMessage
+ ChatMessage(Player, String, LocalTime):
+ ChatMessage(Player, Player, String, LocalTime):
- content: String
- sender: Player
- receiver: Player
- time: String
content: String
sender: Player
time: String
receiver: Player
PlayerData
+ PlayerData(int):
+ PlayerData(CardsMatrix):
- cardsArea: CardsMatrix
- numOfResources: Map<Resource, Integer>
- starterCard: PhysicalCard
- ready: boolean
- secretObjectivesSelection: List<ObjectiveCard>
- secretObjective: ObjectiveCard
- objectivesScore: int
- hand: List<PhysicalCard>
- totalScore: int
- unitaryDecrement(Resource): void
+ objectiveIncrement(ObjectiveCard): int
+ isPositionValid(int, int): boolean
+ checkPreparation(): void
+ removeCard(int): void
+ setStartingChoices(PhysicalCard, List<ObjectiveCard>): void
+ boardSize(): int
+ getCard(int, int): PlayableCard
+ addCard(PhysicalCard): void
+ placeCard(PlayableCard, int, int): void
+ getTargetCorners(int, int): CornerPointer[]
+ numOfResource(Resource): int
secretObjective: ObjectiveCard
finalScore: List<ObjectiveCard>
starterCard: PhysicalCard
hand: List<PhysicalCard>
cardsArea: CardsMatrix
numOfResources: Map<Resource, Integer>
secretObjectivesSelection: List<ObjectiveCard>
totalScore: int
objectivesScore: int
ready: boolean
Player
+ Player(String):
- nickname: String
+ equals(Object): boolean
+ toString(): String
+ hashCode(): int
CardsMatrix
+ CardsMatrix(int):
- orderList: List<Integer>
+ getNearCards(int, int): PlayableCard[]
+ get(int, int): PlayableCard
+ isCornerUncovered(int, int, int): boolean
+ getOrder(int, int): int
+ transposeSecondary(): CardsMatrix
+ rotate45(): CardsMatrix
+ length(): int
+ copy(): CardsMatrix
+ insert(PlayableCard, int, int): void
+ transposePrimary(): CardsMatrix
- rotateX(int, int): int
- rotateY(int, int): int
maxX: int
minY: int
minX: int
orderList: List<Integer>
maxY: int
PlayableCard
+ PlayableCard(Color, int, Corner[]):
+ PlayableCard(Color, int, Bonus, List<Resource>, Corner[]):
+ PlayableCard(String, Color, int, Bonus, List<Resource>, Corner[]
+ PlayableCard(Color, int, List<Resource>, Corner[]):
- color: Color
- corners: Corner[]
- bonus: Bonus
- points: int
- code: String
- fixedResources: List<Resource>
+ isPlaceable(PlayerData, int, int): boolean
+ hashCode(): int
+ toStringTUI(): String[][]
+ equals(Object): boolean
+ scoreIncrement(PlayerData, int, int): int
+ resourceCount(Resource): int
points: int
bonus: Bonus
fixedResources: List<Resource>
corners: List<Corner>
color: Color
neCorner: Corner
code: String
nwCorner: Corner
swCorner: Corner
seCorner: Corner
GoldCard
+ GoldCard(Color, int, Bonus, List<Resource>, Corner[], List<Resou
+ GoldCard(String, Color, int, Bonus, List<Resource>, Corner[], List<
+ toStringTUI(): String[][]
+ checkConstraint(PlayerData): boolean
+ isPlaceable(PlayerData, int, int): boolean
+ equals(Object): boolean
+ hashCode(): int
constraintList: List<Resource>
<<enumeration>> CardType
+ CardType():
+ valueOf(String): CardType
+ values(): CardType[]
PhysicalCard
+ PhysicalCard(CardType, PlayableCard, PlayableCard):
- cardType: CardType
- front: PlayableCard
- back: PlayableCard
cardType: CardType
back: PlayableCard
front: PlayableCard
ObjectiveCard
+ ObjectiveCard(String, int, Objective):
+ ObjectiveCard(int, Objective):
- code: String
- pointsPerCompletion: int
- objective: Objective
+ toStringTUI(): String
+ checkObjective(PlayerData): int
cardType: CardType
objective: Objective
code: String
pointsPerCompletion: int

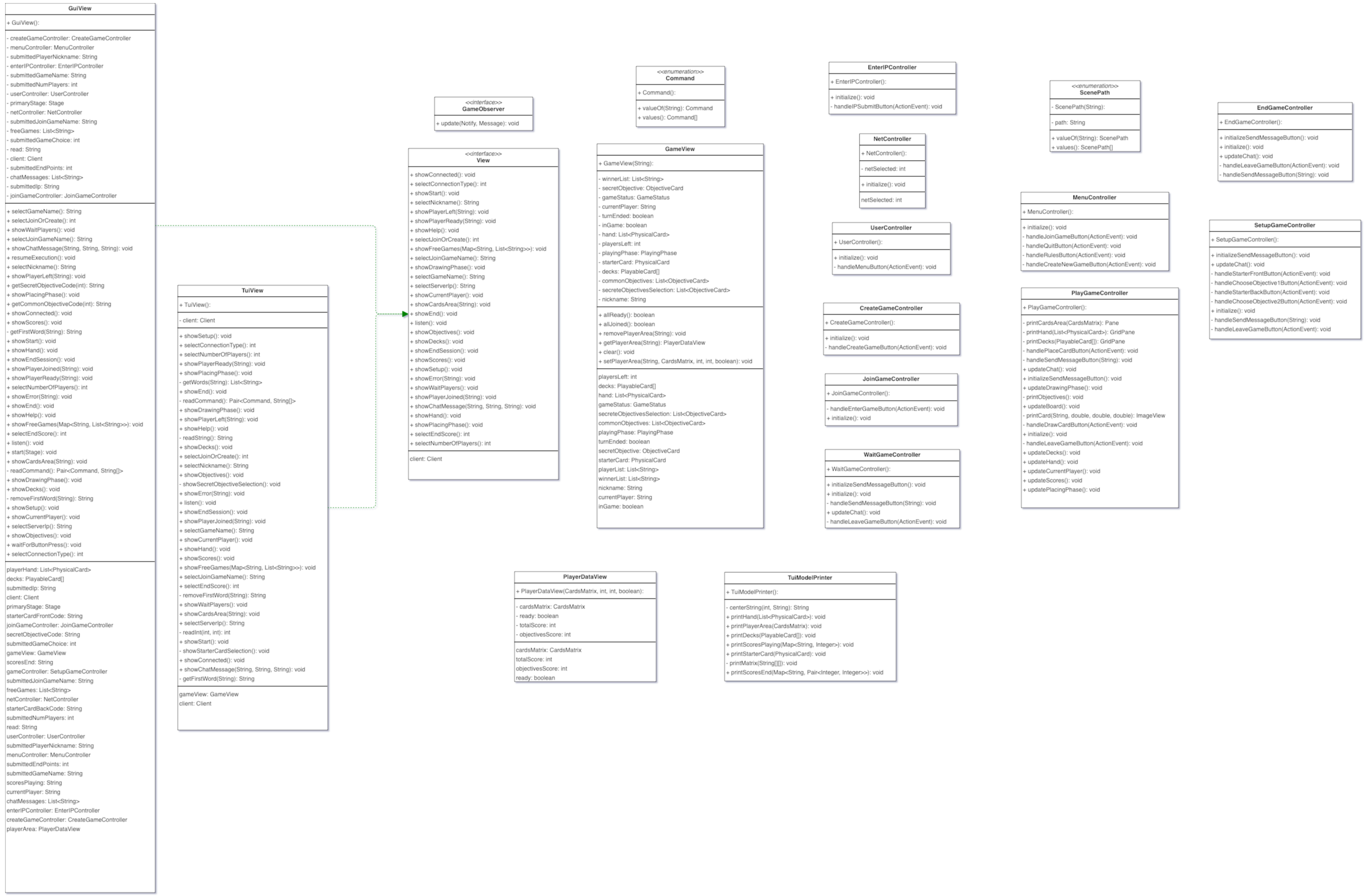
UML: Controller


<<interface>>
GameControllerRemote
+ selectSecretObjective(ClientInterface, int): void
+ selectStarterFace(ClientInterface, int): void
+ placeCard(ClientInterface, PlaceCardRequest): void
+ drawCard(ClientInterface, int): void
+ leaveGame(ClientInterface): void
+ sendChatMessage(ClientInterface, ChatMessageRequest): void
GameController
+ GameController(ClientInterface, String, int, int, String):
- gameId: String
+ sendChatMessage(ClientInterface, ChatMessageRequest): void
- isDrawingPhase(Player): boolean
+ addPlayer(ClientInterface, String): boolean
- getPlayer(ClientInterface): Player
+ selectSecretObjective(ClientInterface, int): void
+ drawCard(ClientInterface, int): void
+ placeCard(ClientInterface, PlaceCardRequest): void
+ selectStarterFace(ClientInterface, int): void
+ removePlayer(ClientInterface): void
- isPlayerTurn(Player): boolean
- isPlacingPhase(Player): boolean
- placeCard(Player, int, int, int, int): void
+ leaveGame(ClientInterface): void
free: boolean
starting: boolean
gameId: String
empty: boolean
waiting: boolean
playerList: List<String>

UML: Rete


<<interface>>
ServerRmiRemote
+ setPlayer(ClientInterface, String): String
+ addClient(ClientInterface): void
+ joinGame(ClientInterface, String): GameControllerRemote
+ createGame(ClientInterface, String, int, int): GameControllerRemote
+ resetPlayer(ClientInterface): void
freeGames: Map<String, List<String>>
<<interface>>
ServerInterface
+ selectStarterFace(int): void
+ setPlayer(String): String
+ drawCard(int): void
+ leaveGame(): void
+ resetPlayer(): void
+ selectSecretObjective(int): void
+ connect(): boolean
+ sendChatMessage(ChatMessageRequest): void
+ createGame(String, int, int): boolean
+ joinGame(String): boolean
+ placeCard(PlaceCardRequest): void
freeGames: Map<String, List<String>>
<<interface>>
ClientInterface
+ update(Notify, Message): void
+ ping(): void
<<enumeration>>
Notify
+ Notify():
+ values(): Notify[]
+ valueOf(String): Notify
Client
+ Client(View):
- gameView: GameView
- switchNotify(Notify, Message): void
- selectStarterFace(int): void
- sendChatMessage(String): void
- createGame(String, int, int): boolean
- waitingPhase(): void
+ setPlayer(String): boolean
- resetPlayer(): void
- playingPhase(): void
+ run(): int
- switchCommand(Command, String[]): void
- joinGame(String): boolean
- connect(): boolean
- listenCommands(): void
- selectSecretObjective(int): void
- placeCard(PlaceCardRequest): void
- leaveGame(): void
+ addCommand(Command, String[]): void
- setupPhase(): void
- playTurn(): void
- drawCard(int): void
- lobby(): void
- endPhase(): void
- sendPrivateChatMessage(String, String): void
+ update(Notify, Message): void
freeGames: Map<String, List<String>>
gameView: GameView
ChatMessageRequest
+ ChatMessageRequest(String, String):
+ ChatMessageRequest(String):
- receiver: String
- content: String
receiver: String
content: String
ServerRmi
+ ServerRmi(Lobby, int):
+ joinGame(ClientInterface, String): GameControllerRemote
+ resetPlayer(ClientInterface): void
+ createGame(ClientInterface, String, int, int): GameControllerRemote
+ addClient(ClientInterface): void
+ setPlayer(ClientInterface, String): String
+ run(): void
freeGames: Map<String, List<String>>
PlaceCardRequest
+ PlaceCardRequest(int, int, int, int):
- face: int
- y: int
- index: int
- x: int
y: int
face: int
x: int
index: int
ClientRmi
+ ClientRmi(Client, String, String):
+ update(Notify, Message): void
+ setPlayer(String): String
+ createGame(String, int, int): boolean
+ ping(): void
+ connect(): boolean
+ joinGame(String): boolean
+ sendChatMessage(ChatMessageRequest): void
+ resetPlayer(): void
+ selectStarterFace(int): void
+ selectSecretObjective(int): void
+ leaveGame(): void
+ placeCard(PlaceCardRequest): void
+ drawCard(int): void
freeGames: Map<String, List<String>>
MockClient
+ MockClient():
- notify: Pair<Notify, Message>
+ ping(): void
+ update(Notify, Message): void
message: Message
notify: Notify
CreateGameRequest
+ CreateGameRequest(String, int, int):
- gameId: String
- numPlayers: int
- endScore: int
gameId: String
numPlayers: int
endScore: int
<<interface>>
Message
GameException
+ GameException(String, Throwable):
SetupMex
+ SetupMex(Game):
- decks: PlayableCard[]
- playerList: List<String>
- commonObjectives: List<ObjectiveCard>
+ getStarterCard(String): PhysicalCard
+ getHand(String): List<PhysicalCard>
+ getHandBacks(String): List<PlayableCard>
+ getSecretObjectivesMap(String): List<ObjectiveCard>
commonObjectives: List<ObjectiveCard>
decks: PlayableCard[]
playerList: List<String>
DecksUpdateMex
+ DecksUpdateMex(Game, Player):
- nickname: String
- decks: PlayableCard[]
- hand: List<PhysicalCard>
decks: PlayableCard[]
nickname: String
handBacks: List<PlayableCard>
hand: List<PhysicalCard>
PlayerMex
+ PlayerMex(Player):
- nickname: String
nickname: String
ErrorMex
+ ErrorMex(Player, String):
- content: String
- nickname: String
content: String
nickname: String
BoardUpdateMex
+ BoardUpdateMex(Game, Player):
- cardsMatrix: CardsMatrix
- totalScore: int
- nickname: String
- hand: List<PhysicalCard>
nickname: String
cardsMatrix: CardsMatrix
handBacks: List<PlayableCard>
totalScore: int
hand: List<PhysicalCard>
ChatMex
+ ChatMex(ChatMessage):
- sender: String
- content: String
- receiver: String
- time: String
receiver: String
sender: String
content: String
time: String
PlayerJoinedMex
+ PlayerJoinedMex(Game, Player):
- nickname: String
- playersLeft: int
playersLeft: int
nickname: String
StringMex
+ StringMex(String):
- string: String
string: String
EndMex
+ EndMex(Game):
- winnerList: List<String>
+ getObjectivesScore(String): int
+ getTotalScore(String): int
winnerList: List<String>
PlayerReadyMex
+ PlayerReadyMex(Game, Player):
- nickname: String
- secretObjective: ObjectiveCard
- cardsMatrix: CardsMatrix
- totalScore: int
nickname: String
secretObjective: ObjectiveCard
cardsMatrix: CardsMatrix
totalScore: int
FreeGamesMex
+ FreeGamesMex(Map<String, List<String>>):
- freeGames: Map<String, List<String>>
freeGames: Map<String, List<String>>

UML: View



```
GuiView
+ GuiView():
- createGameController: CreateGameController
- menuController: MenuController
- submittedPlayerNickname: String
- enterIPController: EnterIPController
- submittedGameName: String
- submittedNumPlayers: int
- userController: UserController
- primaryStage: Stage
- netController: NetController
- submittedJoinGameName: String
- freeGames: List<String>
- submittedGameChoice: int
- read: String
- client: Client
- submittedEndPoints: int
- chatMessages: List<String>
- submittedIp: String
- joinGameController: JoinGameController
+ selectGameName(): String
+ selectJoinOrCreate(): int
+ showWaitPlayers(): void
+ selectJoinGameName(): String
+ showChatMessage(String, String, String): void
+ resumeExecution(): void
+ selectNickname(): String
+ showPlayerLeft(String): void
+ getSecretObjectiveCode(int): String
+ showPlacingPhase(): void
+ getCommonObjectiveCode(int): String
+ showConnected(): void
+ showScores(): void
- getFirstWord(String): String
+ showStart(): void
+ showHand(): void
+ showEndSession(): void
+ showPlayerJoined(String): void
+ showPlayerReady(String): void
+ selectNumberOfPlayers(): int
+ showError(String): void
+ showEnd(): void
+ showHelp(): void
+ showFreeGames(Map<String, List<String>>): void
+ selectEndScore(): int
+ listen(): void
+ start(Stage): void
+ showCardsArea(String): void
- readCommand(): Pair<Command, String[]>
+ showDrawingPhase(): void
+ showDecks(): void
- removeFirstWord(String): String
+ showSetup(): void
+ showCurrentPlayer(): void
+ selectServerIp(): String
+ showObjectives(): void
+ waitForButtonPress(): void
+ selectConnectionType(): int
playerHand: List<PhysicalCard>
decks: PlayableCard[]
submittedIp: String
client: Client
primaryStage: Stage
starterCardFrontCode: String
joinGameController: JoinGameController
secretObjectiveCode: String
submittedGameChoice: int
gameView: GameView
scoresEnd: String
gameController: SetupGameController
submittedJoinGameName: String
freeGames: List<String>
netController: NetController
starterCardBackCode: String
submittedNumPlayers: int
read: String
userController: UserController
submittedPlayerNickname: String
menuController: MenuController
submittedEndPoints: int
submittedGameName: String
scoresPlaying: String
currentPlayer: String
chatMessages: List<String>
enterIPController: EnterIPController
createGameController: CreateGameController
playerArea: PlayerDataView
```

```
TuiView
+ TuiView():
- client: Client
+ showSetup(): void
+ selectConnectionType(): int
+ selectNumberOfPlayers(): int
+ showPlayerReady(String): void
+ showPlacingPhase(): void
- getWords(String): List<String>
+ showEnd(): void
- readCommand(): Pair<Command, String[]>
+ showDrawingPhase(): void
+ showPlayerLeft(String): void
+ showHelp(): void
- readString(): String
+ showDecks(): void
+ selectJoinOrCreate(): int
+ selectNickname(): String
+ showObjectives(): void
- showSecretObjectiveSelection(): void
+ showError(String): void
+ listen(): void
+ showEndSession(): void
+ showPlayerJoined(String): void
+ selectGameName(): String
+ showCurrentPlayer(): void
+ showHand(): void
+ showScores(): void
+ showFreeGames(Map<String, List<String>>): void
+ selectJoinGameName(): String
+ selectEndScore(): int
- removeFirstWord(String): String
+ showWaitPlayers(): void
+ showCardsArea(String): void
+ selectServerIp(): String
- readInt(int, int): int
+ showStart(): void
- showStarterCardSelection(): void
+ showConnected(): void
+ showChatMessage(String, String, String): void
- getFirstWord(String): String
gameView: GameView
client: Client
```

```
<<interface>>
GameObserver
+ update(Notify, Message): void
```

```
<<interface>>
View
+ showConnected(): void
+ selectConnectionType(): int
+ showStart(): void
+ selectNickname(): String
+ showPlayerLeft(String): void
+ showPlayerReady(String): void
+ showHelp(): void
+ selectJoinOrCreate(): int
+ showFreeGames(Map<String, List<String>>): void
+ selectJoinGameName(): String
+ showDrawingPhase(): void
+ selectGameName(): String
+ selectServerIp(): String
+ showCurrentPlayer(): void
+ showCardsArea(String): void
+ showEnd(): void
+ listen(): void
+ showObjectives(): void
+ showDecks(): void
+ showEndSession(): void
+ showScores(): void
+ showSetup(): void
+ showError(String): void
+ showWaitPlayers(): void
+ showPlayerJoined(String): void
+ showChatMessage(String, String, String): void
+ showHand(): void
+ showPlacingPhase(): void
+ selectEndScore(): int
+ selectNumberOfPlayers(): int
client: Client
```

```
<<enumeration>>
Command
+ Command():
+ valueOf(String): Command
+ values(): Command[]
```

```
GameView
+ GameView(String):
- winnerList: List<String>
- secretObjective: ObjectiveCard
- gameStatus: GameStatus
- currentPlayer: String
- turnEnded: boolean
- inGame: boolean
- hand: List<PhysicalCard>
- playersLeft: int
- playingPhase: PlayingPhase
- starterCard: PhysicalCard
- decks: PlayableCard[]
- commonObjectives: List<ObjectiveCard>
- secreteObjectivesSelection: List<ObjectiveCard>
- nickname: String
+ allReady(): boolean
+ allJoined(): boolean
+ removePlayerArea(String): void
+ getPlayerArea(String): PlayerDataView
+ clear(): void
+ setPlayerArea(String, CardsMatrix, int, int, boolean): void
playersLeft: int
decks: PlayableCard[]
hand: List<PhysicalCard>
gameStatus: GameStatus
secreteObjectivesSelection: List<ObjectiveCard>
commonObjectives: List<ObjectiveCard>
playingPhase: PlayingPhase
turnEnded: boolean
secretObjective: ObjectiveCard
starterCard: PhysicalCard
playerList: List<String>
winnerList: List<String>
nickname: String
currentPlayer: String
inGame: boolean
```

```
EnterIPController
+ EnterIPController():
+ initialize(): void
- handleIPSubmitButton(ActionEvent): void
```

```
NetController
+ NetController():
- netSelected: int
+ initialize(): void
netSelected: int
```

```
UserController
+ UserController():
+ initialize(): void
- handleMenuButton(ActionEvent): void
```

```
CreateGameController
+ CreateGameController():
+ initialize(): void
- handleCreateGameButton(ActionEvent): void
```

```
JoinGameController
+ JoinGameController():
- handleEnterGameButton(ActionEvent): void
+ initialize(): void
```

```
WaitGameController
+ WaitGameController():
+ initializeSendMessageButton(): void
+ initialize(): void
- handleSendMessageButton(String): void
+ updateChat(): void
- handleLeaveGameButton(ActionEvent): void
```

```
<<enumeration>>
ScenePath
- ScenePath(String):
- path: String
+ valueOf(String): ScenePath
+ values(): ScenePath[]
```

```
MenuController
+ MenuController():
+ initialize(): void
- handleJoinGameButton(ActionEvent): void
- handleQuitButton(ActionEvent): void
- handleRulesButton(ActionEvent): void
- handleCreateNewGameButton(ActionEvent): void
```

```
PlayGameController
+ PlayGameController():
- printCardsArea(CardsMatrix): Pane
- printHand(List<PhysicalCard>): GridPane
- printDecks(PlayableCard[]): GridPane
- handlePlaceCardButton(ActionEvent): void
- handleSendMessageButton(String): void
+ updateChat(): void
+ initializeSendMessageButton(): void
+ updateDrawingPhase(): void
- printObjectives(): void
+ updateBoard(): void
- printCard(String, double, double, double): ImageView
- handleDrawCardButton(ActionEvent): void
+ initialize(): void
- handleLeaveGameButton(ActionEvent): void
+ updateDecks(): void
+ updateHand(): void
+ updateCurrentPlayer(): void
+ updateScores(): void
+ updatePlacingPhase(): void
```

```
EndGameController
+ EndGameController():
+ initializeSendMessageButton(): void
+ initialize(): void
+ updateChat(): void
- handleLeaveGameButton(ActionEvent): void
- handleSendMessageButton(String): void
```

```
SetupGameController
+ SetupGameController():
+ initializeSendMessageButton(): void
+ updateChat(): void
- handleStarterFrontButton(ActionEvent): void
- handleChooseObjective1Button(ActionEvent): void
- handleStarterBackButton(ActionEvent): void
- handleChooseObjective2Button(ActionEvent): void
+ initialize(): void
- handleSendMessageButton(String): void
- handleLeaveGameButton(ActionEvent): void
```

```
PlayerDataView
+ PlayerDataView(CardsMatrix, int, int, boolean):
- cardsMatrix: CardsMatrix
- ready: boolean
- totalScore: int
- objectivesScore: int
cardsMatrix: CardsMatrix
totalScore: int
objectivesScore: int
ready: boolean
```

```
TuiModelPrinter
+ TuiModelPrinter():
- centerString(int, String): String
+ printHand(List<PhysicalCard>): void
+ printPlayerArea(CardsMatrix): void
+ printDecks(PlayableCard[]): void
+ printScoresPlaying(Map<String, Integer>): void
+ printStarterCard(PhysicalCard): void
- printMatrix(String[][]): void
+ printScoresEnd(Map<String, Pair<Integer, Integer>>): void
```

POLITECNICO
MILANO 1863