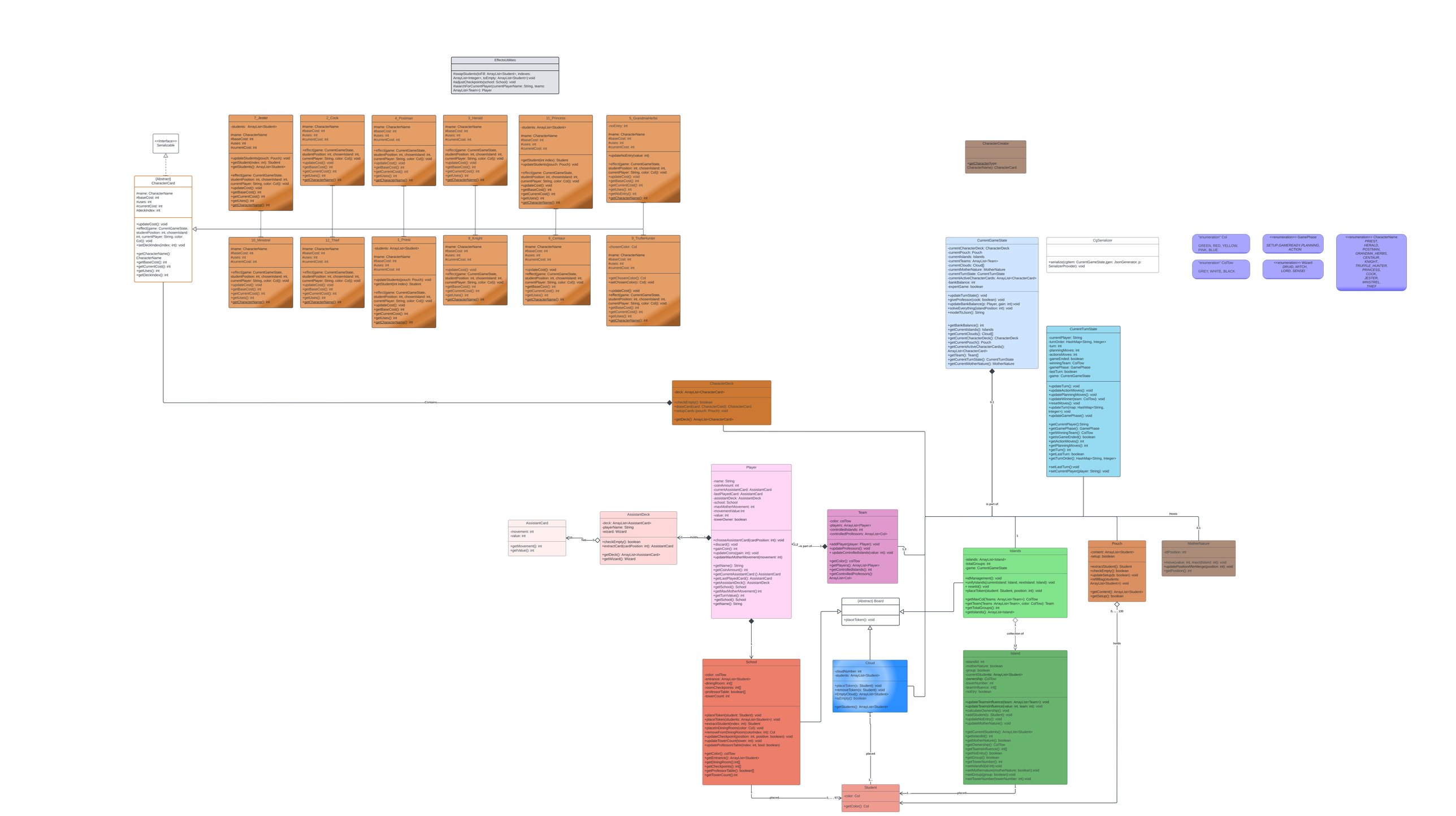
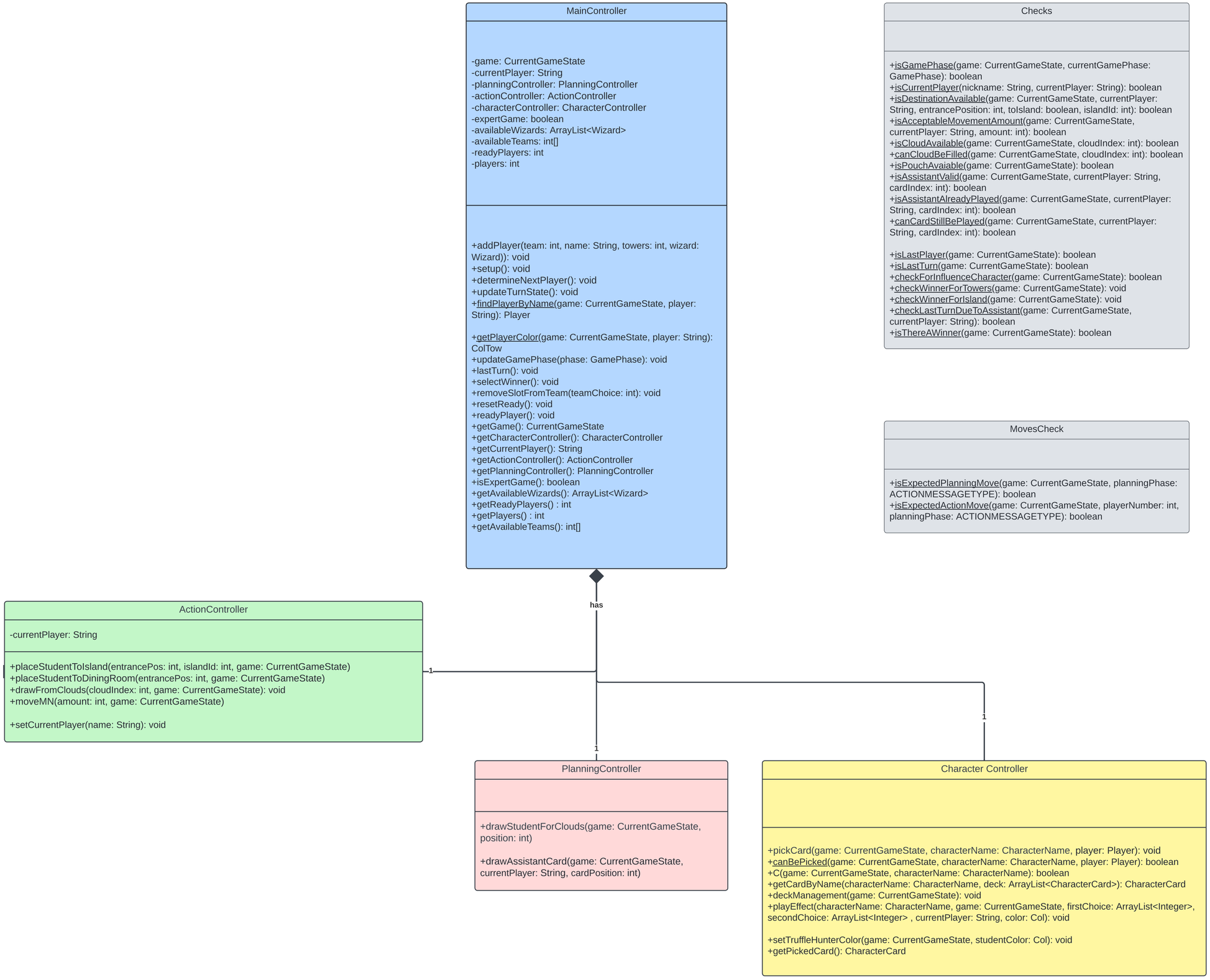
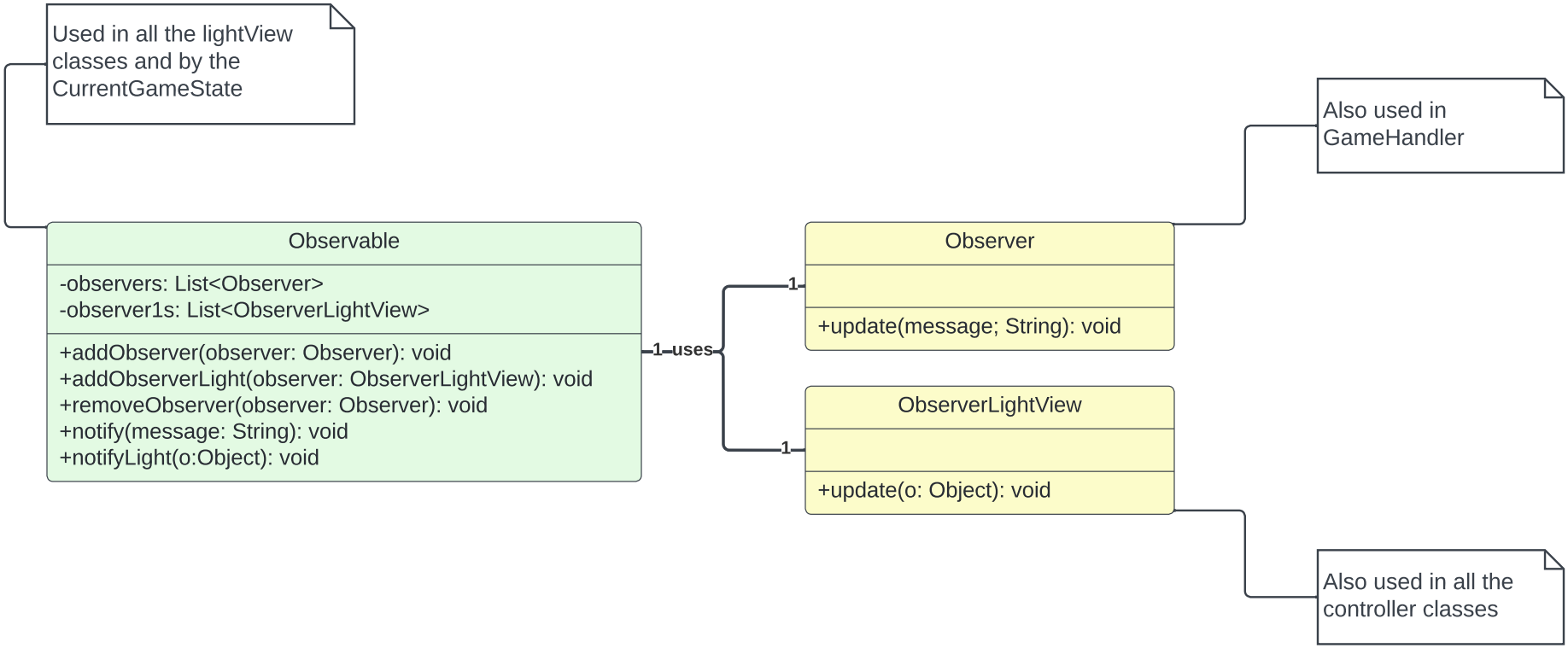
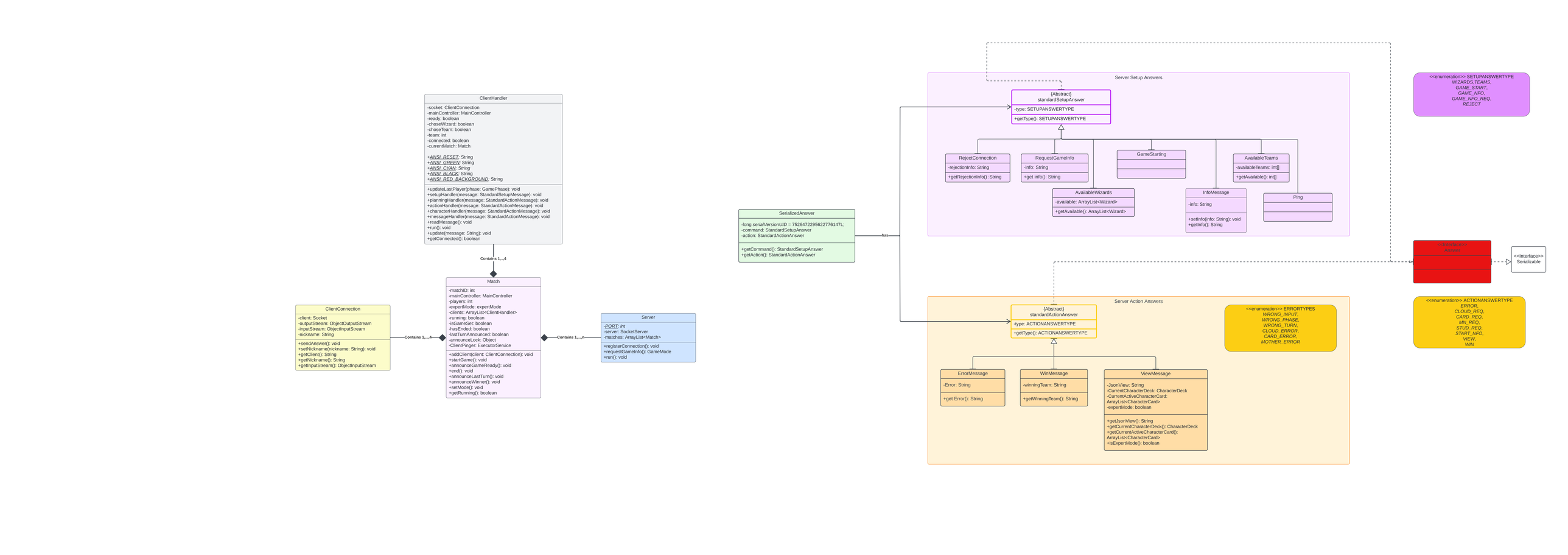
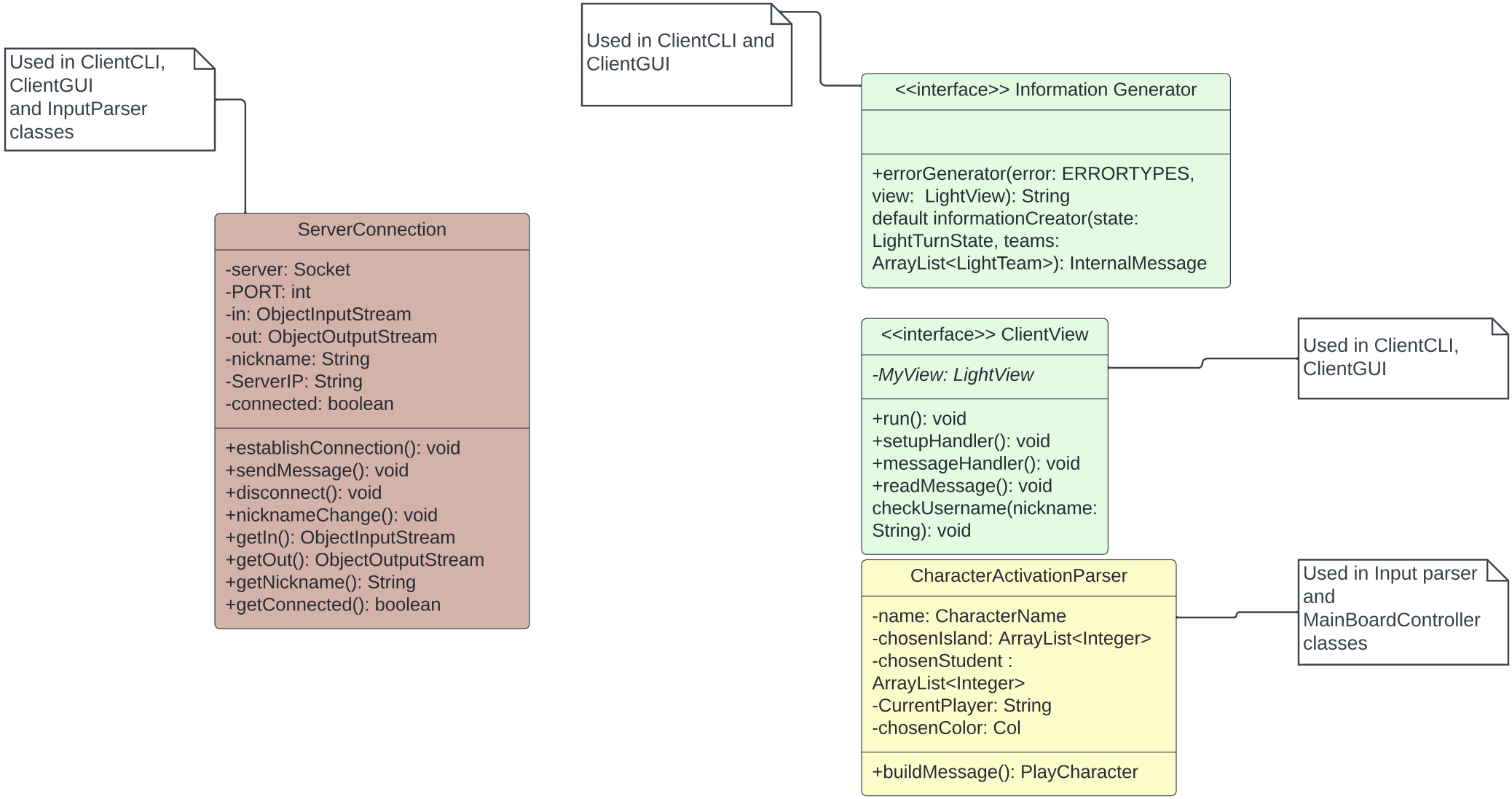
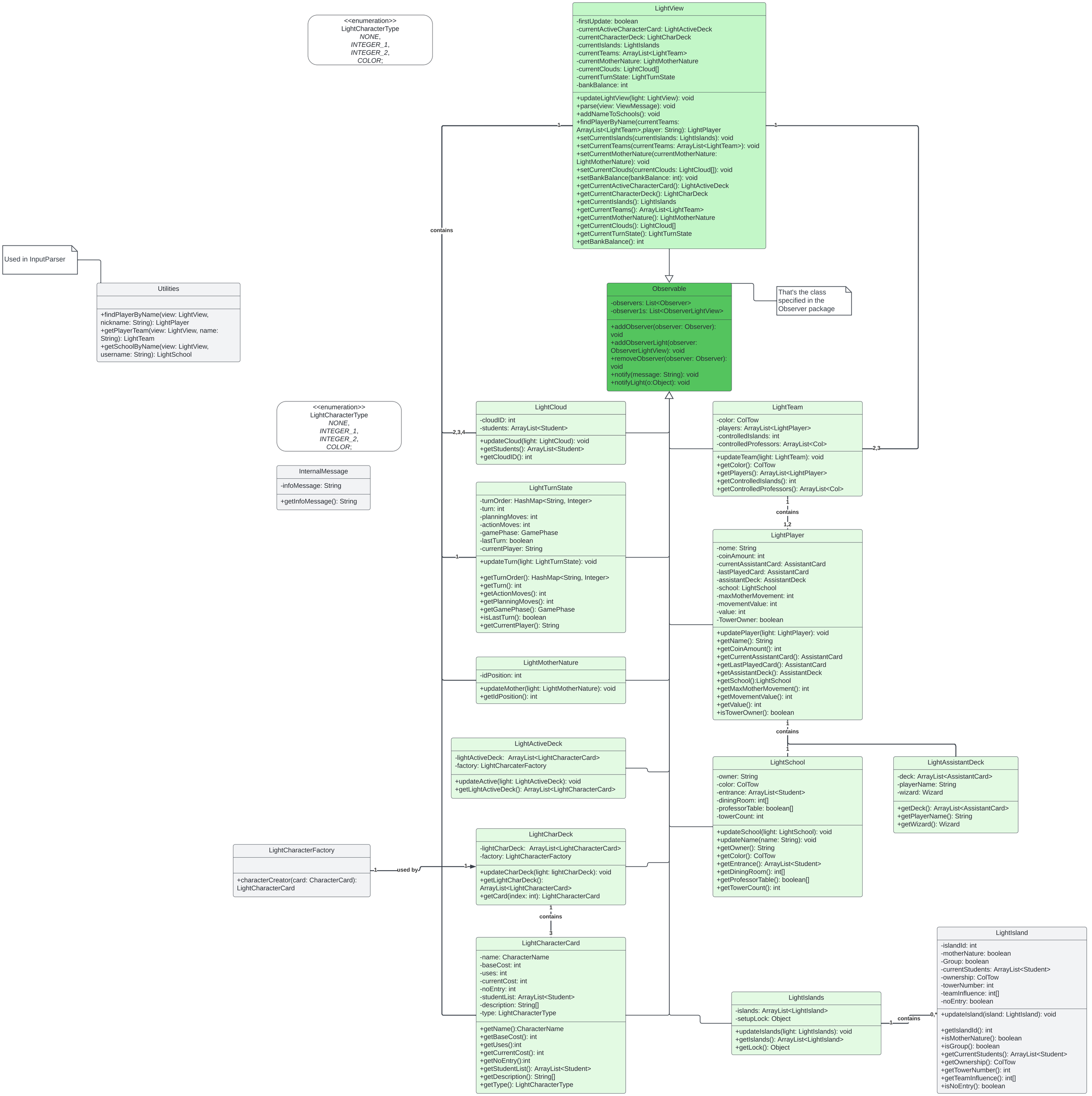

EffectsUtilities
#swapStudents(toFill: ArrayList<Student>, indexes: ArrayList<Integer>, toEmpty: ArrayList<Student>): void
#adjustCheckpoints(school: School): void
#searchForCurrentPlayer(currentPlayerName: String, teams: ArrayList<Team>): Player
<<Interface>>
Serializable
{Abstract}
CharacterCard
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
#deckIndex: int
+updateCost(): void
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+setDeckIndex(index: int): void
+getCharacterName(): CharacterName
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getDeckIndex(): int
7_Jester
-students: ArrayList<Student>
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+updateStudents(pouch: Pouch): void
+getStudent(index: int): Student
+getStudents(): ArrayList<Student>
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
2_Cook
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
4_Postman
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
3_Herald
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
11_Princess
-students: ArrayList<Student>
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+getStudent(int index): Student
+updateStudents(pouch: Pouch): void
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
5_GrandmaHerbs
-noEntry: int
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+updateNoEntry(value: int)
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getNoEntry(): int
+getCharacterName(): int
10_Minstrel
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
12_Thief
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+setCharacterName(): int
1_Priest
-students: ArrayList<Student>
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+updateStudents(pouch: Pouch): void
+getStudent(int index): Student
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+updateCost(): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
8_Knight
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+updateCost(): void
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
6_Centaur
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+updateCost(): void
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
9_TruffleHunter
-chosenColor: Col
#name: CharacterName
#baseCost: int
#uses: int
#currentCost: int
+getChosenColor(): Col
+setChosenColor(c: Col): void
+updateCost(): void
+effect(game: CurrentGameState, studentPosition: int, chosenIsland: int, currentPlayer: String, color: Col)): void
+getBaseCost(): int
+getCurrentCost(): int
+getUses(): int
+getCharacterName(): int
CharacterCreator
+getCharacter(type: CharacterName): CharacterCard
Contains
CharacterDeck
-deck: ArrayList<CharacterCard>
+checkEmpty(): boolean
+drawCard(card: CharacterCard): CharacterCard
+setupCards (pouch: Pouch): void
+getDeck(): ArrayList<CharacterCard>
CurrentGameState
-currentCharacterDeck: CharacterDeck
-currentPouch: Pouch
-currentIslands: Islands
-currentTeams: ArrayList<Team>
-currentClouds: Cloud[]
-currentMotherNature: MotherNature
-currentTurnState: CurrentTurnState
-currentActiveCharacterCards: ArrayList<CharacterCard>
-bankBalance: int
-expertGame: boolean
+updateTurnState(): void
+giveProfessor(cook: boolean): void
+updateBankBalance(p: Player, gain: int):void
+solveEveryIslandPosition: int): void
+modelToJson(): String
+getBankBalance(): int
+getCurrentIslands(): Islands
+getCurrentClouds(): Cloud[]
+getCurrentCharacterDeck(): CharacterDeck
+getCurrentPouch(): Pouch
+getCurrentActiveCharacterCards(): ArrayList<CharacterCard>
+getTeam(): Team[]
+getCurrentTurnState(): CurrentTurnState
+getCurrentMotherNature(): MotherNature
CgSerializer
+serialize(cgItem: CurrentGameState,jgen: JsonGenerator, p: SerializerProvider): void
"enumeration" Col
GREEN, RED, YELLOW, PINK, BLUE
"enumeration" ColTow
GREY, WHITE, BLACK
<<enumeration>> GamePhase
SETUP,GAMEREADY,PLANNING, ACTION
<<enumeration>> Wizard
DRUID, WITCH, LORD, SENSEI
<<enumeration>> CharacterName
PRIEST, HERALD, POSTMAN, GRANDMA_HERBS, CENTAUR, KNIGHT, TRUFFLE_HUNTER, PRINCESS, COOK, JESTER, MINSTREL, THIEF
CurrentTurnState
-currentPlayer: String
-turnOrder: HashMap<String, Integer>
-turn: int
-planningMoves: int
-actionsMoves: int
-gameEnded: boolean
-winningTeam: ColTow
-gamePhase: GamePhase
-lastTurn: boolean
-game: CurrentGameState
+updateTurn(): void
+updateActionMoves(): void
+updatePlanningMoves(): void
+updateWinner(team: ColTow): void
+resetMoves(): void
+updateTurn(map: HashMap<String, Integer>): void
+updateGamePhase() :void
+getCurrentPlayer():String
+getGamePhase(): GamePhase
+getWinningTeam(): ColTow
+getIsGameEnded(): boolean
+getActionMoves(): int
+getPlanningMoves(): int
+getTurn(): int
+getLastTurn: boolean
+getTurnOrder(): HashMap<String, Integer>
+setLastTurn():void
+setCurrentPlayer(player: String): void
0,1
is part of
Hosts
0,1
AssistantCard
-movement: int
-value: int
+getMovement(): int
+getValue(): int
1
0
has
AssistantDeck
-deck: ArrayList<AssistantCard>
-playerName: String
-wizard: Wizard
+checkEmpty(): boolean
+extractCard(cardPosition: int): AssistantCard
+getDeck(): ArrayList<AssistantCard>
+getWizard(): Wizard
1
1
holds
Player
-name: String
-coinAmount: int
-currentAssistantCard: AssistantCard
-lastPlayedCard: AssistantCard
-assistantDeck: AssistantDeck
-school: School
-maxMotherMovement: int
-movementValue:int
-value: int
-towerOwner: boolean
+chooseAssistantCard(cardPosition: int): void
+discard(): void
+gainCoin(): int
+updateCoins(gain: int): void
+updateMaxMotherMovement(movement: int)
+getName(): String
+getCoinAmount(): int
+getCurrentAssistantCard () AssistantCard
+getLastPlayedCard(): AssistantCard
+getAssistantDeck(): AssistantDeck
+getSchool(): School
+getMaxMotherMovement():int
+getTurnValue(): int
+getSchool(): School
+getName(): String
1,2
is part of
1
Team
-color: colTow
-players: ArrayList<Player>
-controlledIslands: int
-controlledProfessors: ArrayList<Col>
+addPlayer(player: Player): void
+updateProfessors(): void
+updateControlledIslands(value): void
+getColor(): colTow
+getPlayers(): ArrayList<Player>
+getControlledIslands(): int
+getControlledProfessors(): ArrayList<Col>
1,3
{Abstract} Board
+placeToken(): void
School
-color: colTow
-entrance: ArrayList<Student>
-diningRoom: int[]
-roomCheckpoints: int[]
-professorTable: boolean[]
-towerCount: int
+placeToken(student: Student): void
+placeToken(students: ArrayList<Student>): void
+extractStudent(index: int): Student
+placeInDiningRoom(color: Col): void
+removeFromDiningRoom(colorIndex: int): Col
+updateCheckpoint(position: int, positive: boolean): void
+updateTowerCount(tower: int): void
+updateProfessorsTable(index: int, bool: boolean)
+getColor(): colTow
+getEntrance(): ArrayList<Student>
+getDiningRoom():int[]
+getCheckpoints(): int[]
+getProfessorTable(): boolean[]
+getTowerCount():int
Cloud
-cloudNumber: int
-students: ArrayList<Student>
+placeToken(s: Student): void
+removeToken(s: Student): void
+EmptyCloud(): ArrayList<Student>
+isEmpty(): boolean
+getStudents(): ArrayList<Student>
1
placed
1
1
placed
1, ... , 97-3
Student
-color: Col
+getColor(): Col
Islands
-islands: ArrayList<Island>
-totalGroups: int
-game: CurrentGameState
+idManagement(): void
+unifyIslands(currentIsland: Island, nextIsland: Island): void
+ resetId(): void
+placeToken(student: Student, position: int): void
+getMaxCol(Teams: ArrayList<Team>): ColTow
+getTeamTeams :ArrayList<Team>, color: ColTow): Team
+getTotalGroups(): int
+getIslands(): ArrayList<Island>
1
collection of
12
Island
-islandId: int
-motherNature: boolean
-group: boolean
-currentStudents: ArrayList<Student>
-ownership: ColTow
-towerNumber: int
-teamInfluence: int[]
-noEtry: boolean
+updateTeamsInfluence(team: ArrayList<Team>): void
+updateTeamsInfluence(value: int, team: int): void
+calculateOwnership(): void
+addStudent(s: Student): void
+updateNoEntry(): void
+updateMotherNature(): void
+getCurrentStudents(): ArrayList<Student>
+getIslandId(): int
+getMotherNature(): boolean
+getOwnership(): ColTow
+getTeamsInfluence(): int[]
+getNoEntry(): boolean
+getGroup(): boolean
+getTowerNumber(): int
+setIslandId(id:int):void
+setMothernature(motherNature: boolean):void
+setGroup(group: boolean):void
+setTowerNumber(towerNumber: int):void
1
placed
Pouch
-content: ArrayList<Student>
-setup: boolean
+extractStudent(i): Student
+checkEmpty(): boolean
+updateSetup(b: boolean): void
+refillBag(students: ArrayList<Student>): void
+getContent(): ArrayList<Student>
+getSetup(): boolean
0, ... ,130
holds
MotherNature
-idPosition: int
+move(value: int, maxIsland: int): void
+updatePositionAfterMerge(position: int): void
+getPosition(): int

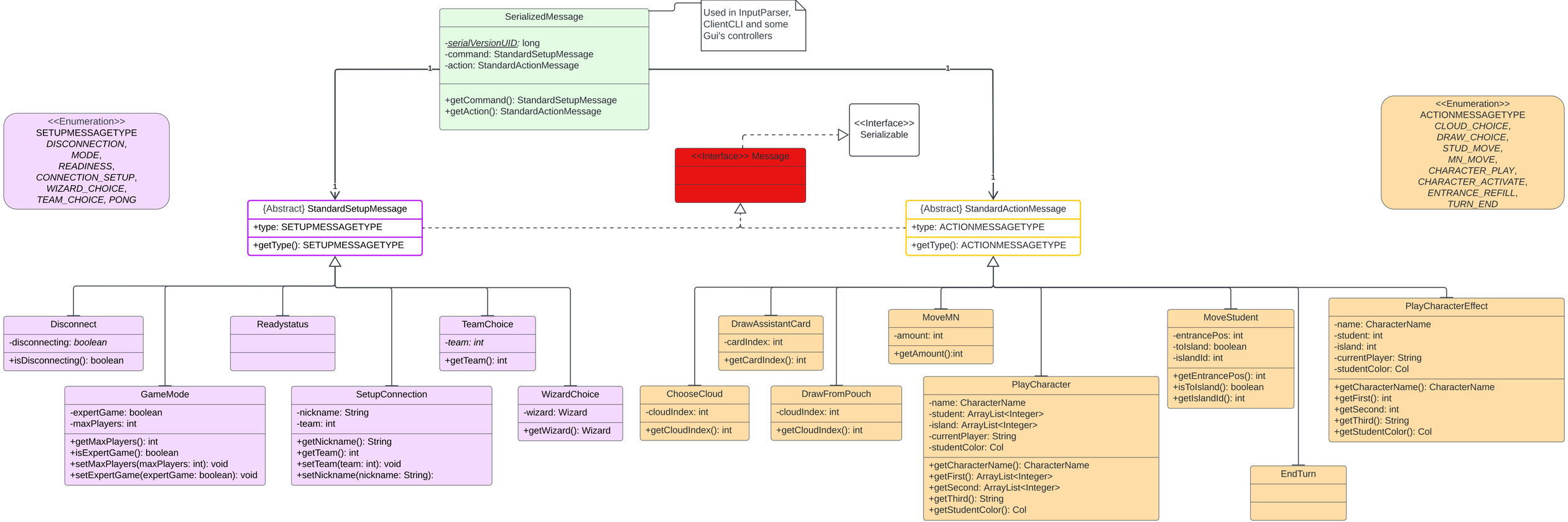

MainController
-game: CurrentGameState
-currentPlayer: String
-planningController: PlanningController
-actionController: ActionController
-characterController: CharacterController
-expertGame: boolean
-availableWizards: ArrayList<Wizard>
-availableTeams: int[]
-readyPlayers: int
-players: int
+addPlayer(team: int, name: String, towers: int, wizard: Wizard)): void
+setup(): void
+determineNextPlayer(): void
+updateTurnState(): void
+findPlayerByName(game: CurrentGameState, player: String): Player
+getPlayerColor(game: CurrentGameState, player: String): ColTow
+updateGamePhase(phase: GamePhase): void
+lastTurn(): void
+selectWinner(): void
+removeSlotFromTeam(teamChoice: int): void
+resetReady(): void
+readyPlayer(): void
+getGame(): CurrentGameState
+getCharacterController(): CharacterController
+getCurrentPlayer(): String
+getActionController(): ActionController
+getPlanningController(): PlanningController
+isExpertGame(): boolean
+getAvailableWizards(): ArrayList<Wizard>
+getReadyPlayers() : int
+getPlayers() : int
+getAvailableTeams(): int[]
Checks
+isGamePhase(game: CurrentGameState, currentGamePhase: GamePhase): boolean
+isCurrentPlayer(nickname: String, currentPlayer: String): boolean
+isDestinationAvailable(game: CurrentGameState, currentPlayer: String, entrancePosition: int, toIsland: boolean, islandId: int): boolean
+isAcceptableMovementAmount(game: CurrentGameState, currentPlayer: String, amount: int): boolean
+isCloudAvailable(game: CurrentGameState, cloudIndex: int): boolean
+canCloudBeFilled(game: CurrentGameState, cloudIndex: int): boolean
+isPouchAvaiable(game: CurrentGameState): boolean
+isAssistantValid(game: CurrentGameState, currentPlayer: String, cardIndex: int): boolean
+isAssistantAlreadyPlayed(game: CurrentGameState, currentPlayer: String, cardIndex: int): boolean
+canCardStillBePlayed(game: CurrentGameState, currentPlayer: String, cardIndex: int): boolean
+isLastPlayer(game: CurrentGameState): boolean
+isLastTurn(game: CurrentGameState): boolean
+checkForInfluenceCharacter(game: CurrentGameState): boolean
+checkWinnerForTowers(game: CurrentGameState): void
+checkWinnerForIsland(game: CurrentGameState): void
+checkLastTurnDueToAssistant(game: CurrentGameState, currentPlayer: String): boolean
+isThereAWinner(game: CurrentGameState): boolean
MovesCheck
+isExpectedPlanningMove(game: CurrentGameState, planningPhase: ACTIONMESSAGETYPE): boolean
+isExpectedActionMove(game: CurrentGameState, playerNumber: int, planningPhase: ACTIONMESSAGETYPE): boolean
has
1
1
1
ActionController
-currentPlayer: String
+placeStudentToIsland(entrancePos: int, islandId: int, game: CurrentGameState)
+placeStudentToDiningRoom(entrancePos: int, game: CurrentGameState)
+drawFromClouds(cloudIndex: int, game: CurrentGameState): void
+moveMN(amount: int, game: CurrentGameState)
+setCurrentPlayer(name: String): void
PlanningController
+drawStudentForClouds(game: CurrentGameState, position: int)
+drawAssistantCard(game: CurrentGameState, currentPlayer: String, cardPosition: int)
Character Controller
+pickCard(game: CurrentGameState, characterName: CharacterName, player: Player): void
+canBePicked(game: CurrentGameState, characterName: CharacterName, player: Player): boolean
+C(game: CurrentGameState, characterName: CharacterName): boolean
+getCardByName(characterName: CharacterName, deck: ArrayList<CharacterCard>): CharacterCard
+deckManagement(game: CurrentGameState): void
+playEffect(characterName: CharacterName, game: CurrentGameState, firstChoice: ArrayList<Integer>, secondChoice: ArrayList<Integer> , currentPlayer: String, color: Col): void
+setTruffleHunterColor(game: CurrentGameState, studentColor: Col): void
+getPickedCard(): CharacterCard

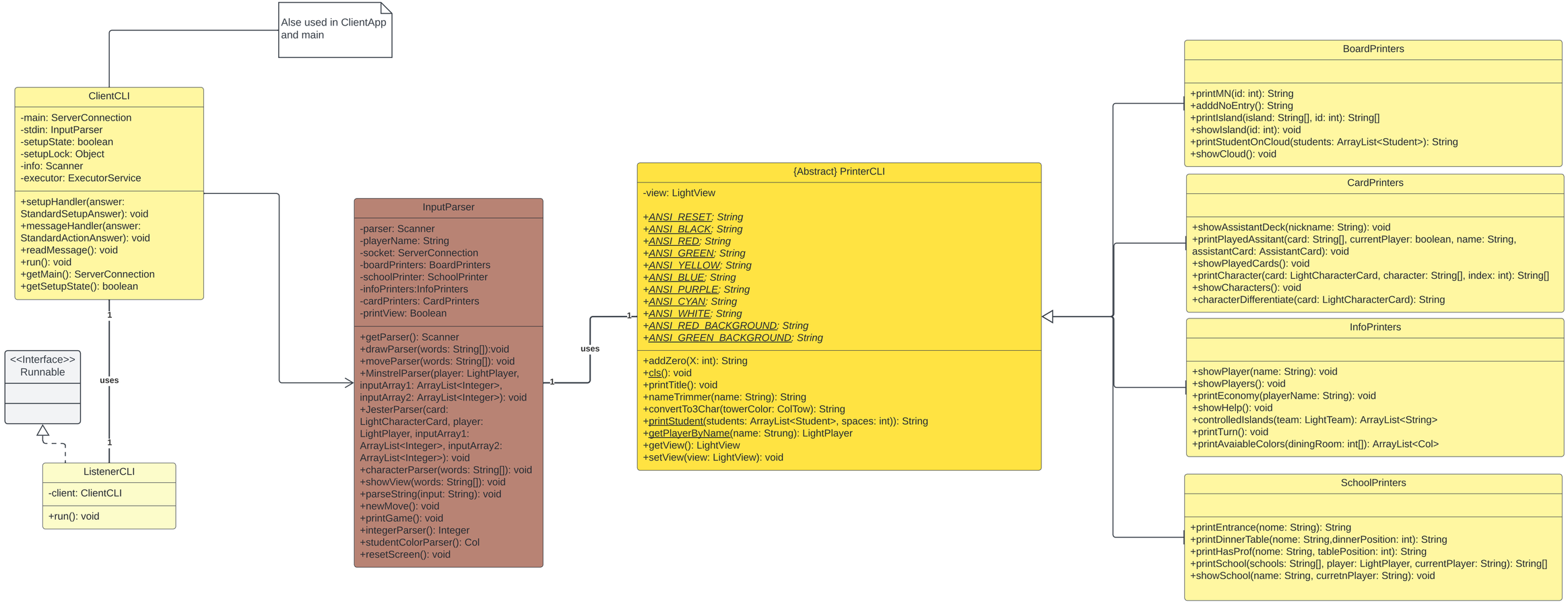

Used in all the lightView classes and by the CurrentGameState
Observable
-observers: List<Observer>
-observer1s: List<ObserverLightView>
+addObserver(observer: Observer): void
+addObserverLight(observer: ObserverLightView): void
+removeObserver(observer: Observer): void
+notify(message: String): void
+notifyLight(o:Object): void
1
uses
1
1
Observer
+update(message; String): void
ObserverLightView
+update(o: Object): void
Also used in GameHandler
Also used in all the controller classes

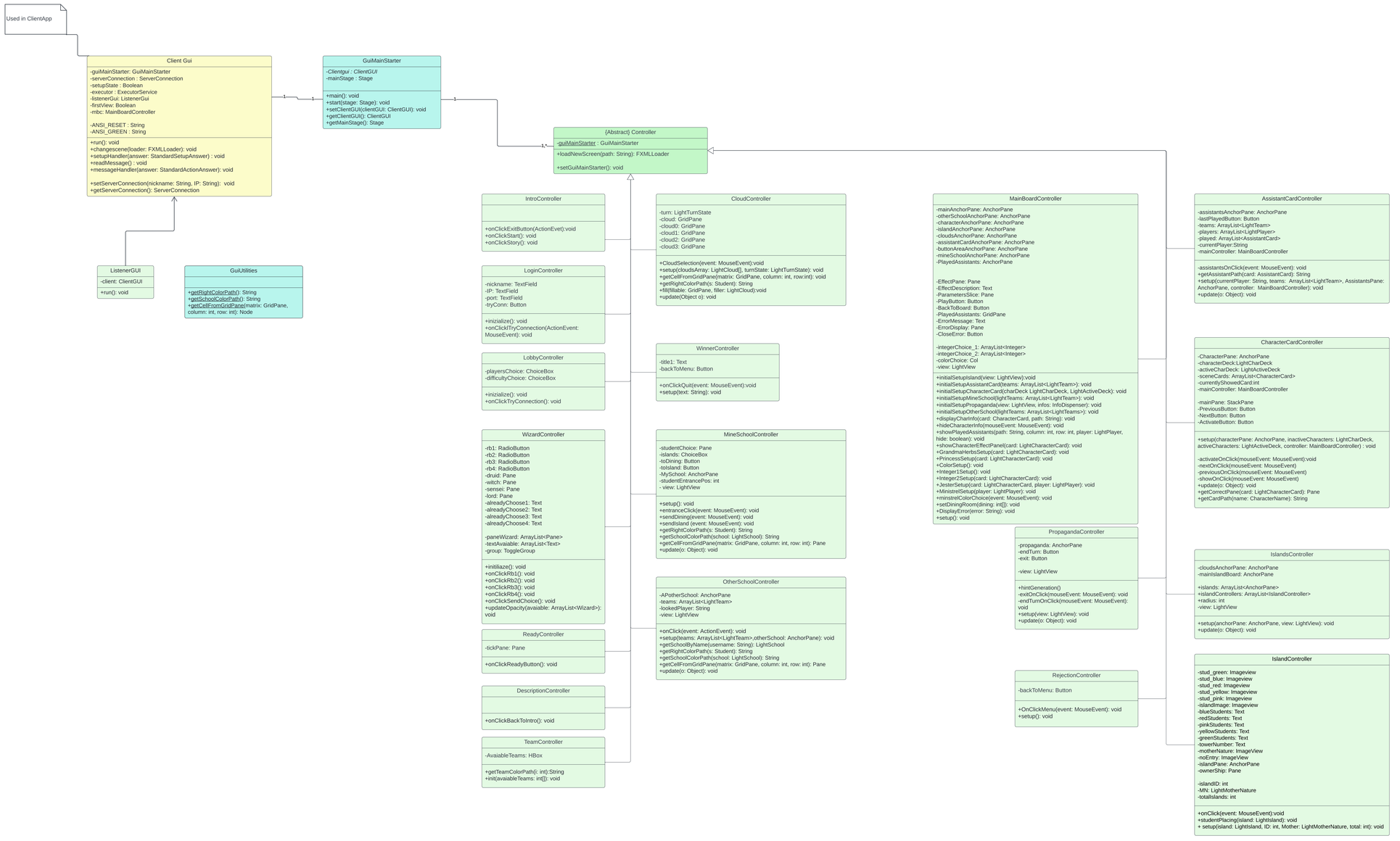

ClientHandler
-socket: ClientConnection
-mainController: MainController
-ready: boolean
-choseWizard: boolean
-choseTeam: boolean
-team: int
-connected: boolean
-currentMatch: Match
+ANSI_RESET: String
+ANSI_GREEN: String
+ANSI_CYAN: String
+ANSI_BLACK: String
+ANSI_RED_BACKGROUND: String
+updateLastPlayer(phase: GamePhase): void
+setupHandler(message: StandardSetupMessage): void
+planningHandler(message: StandardActionMessage): void
+actionHandler(message: StandardActionMessage): void
+characterHandler(message: StandardActionMessage): void
+messageHandler(message: StandardActionMessage): void
+readMessage(): void
+run(): void
+update(message: String): void
+getConnected(): boolean
Contains 1...4
Match
-matchID: int
-mainController: MainController
-players: int
-expertMode: expertMode
-clients: ArrayList<ClientHandler>
-running: boolean
-isGameSet: boolean
-hasEnded: boolean
-lastTurnAnnounced: boolean
-announceLock: Object
-ClientPinger: ExecutorService
+addClient(client: ClientConnection): void
+startGame(): void
+announceGameReady(): void
+end(): void
+announceLastTurn(): void
+announceWinner(): void
+setMode(): void
+getRunning(): boolean
ClientConnection
-client: Socket
-outputStream: ObjectOutputStream
-inputStream: ObjectInputStream
-nickname: String
+sendAnswer(): void
+setNickname(nickname: String): void
+getClient(): String
+getNickname(): String
+getInputStream(): ObjectInputStream
Contains 1...4
Contains 1...n
Server
-PORT: int
-server: SocketServer
-matches: ArrayList<Match>
+registerConnection(): void
+requestGameInfo(): GameMode
+run(): void
SerializedAnswer
-long serialVersionUID = 7526472295622776147L;
-command: StandardSetupAnswer
-action: StandardActionAnswer
+getCommand(): StandardSetupAnswer
+getAction(): StandardActionAnswer
has
Server Setup Answers
[Abstract]
standardSetupAnswer
-type: SETUPANSWERTYPE
+getType(): SETUPANSWERTYPE
RejectConnection
-rejectionInfo: String
+getRejectionInfo() :String
RequestGameInfo
-info: String
+get info(): String
GameStarting
AvailableTeams
-availableTeams: int[]
+getAvailable(): int[]
AvailableWizards
-available: ArrayList<Wizard>
+getAvailable(): ArrayList<Wizard>
InfoMessage
-info: String
+setInfo(info: String): void
+getInfo(): String
Ping
<<enumeration>> SETUPANSWERTYPE
WIZARDS,TEAMS,
GAME_START,
GAME_NFO,
GAME_NFO_REQ,
REJECT
<<Interface>>
Answer
<<Interface>>
Serializable
Server Action Answers
[Abstract]
standardActionAnswer
-type: ACTIONANSWERTYPE
+getType(): ACTIONANSWERTYPE
<<enumeration>> ERRORTYPES
WRONG_INPUT,
WRONG_PHASE,
WRONG_TURN,
CLOUD_ERROR,
CARD_ERROR,
MOTHER_ERROR
<<enumeration>> ACTIONANSWERTYPE
ERROR,
CLOUD_REQ,
CARD_REQ,
MN_REQ,
STUD_REQ,
START_NFO,
VIEW,
WIN
ErrorMessage
-Error: String
+get Error(): String
WinMessage
-winningTeam: String
+getWinningTeam(): String
ViewMessage
-JsonView: String
-CurrentCharacterDeck: CharacterDeck
-CurrentActiveCharacterCard:
ArrayList<CharacterCard>
-expertMode: boolean
+getJsonView(): String
+getCurrentCharacterDeck(): CharacterDeck
+getCurrentActiveCharacterCard():
ArrayList<CharacterCard>
+isExpertMode(): boolean

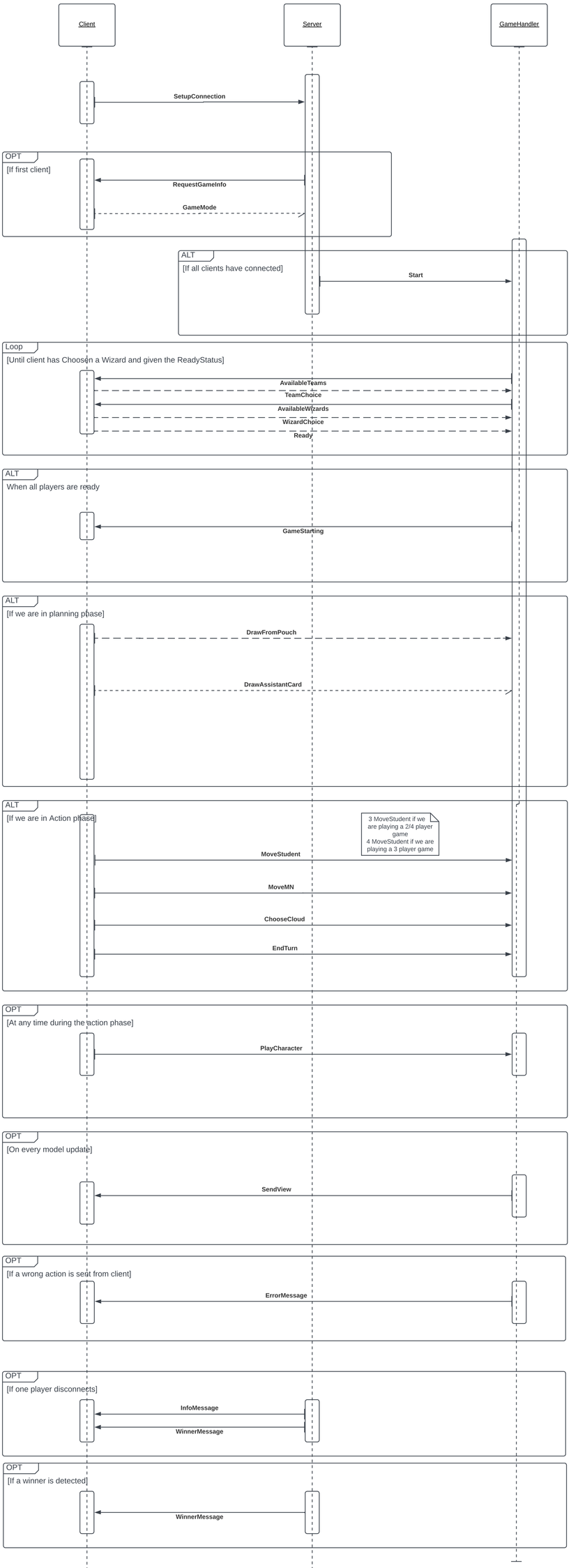

Used in ClientCLI, ClientGUI and InputParser classes
ServerConnection
-server: Socket
-PORT: int
-in: ObjectInputStream
-out: ObjectOutputStream
-nickname: String
-ServerIP: String
-connected: boolean
+establishConnection(): void
+sendMessage(): void
+disconnect(): void
+nicknameChange(): void
+getIn(): ObjectInputStream
+getOut(): ObjectOutputStream
+getNickname(): String
+getConnected(): boolean
Used in ClientCLI and ClientGUI
<<interface>> Information Generator
+errorGenerator(error: ERRORTYPES, view: LightView): String
default informationCreator(state: LightTurnState, teams: ArrayList<LightTeam>): InternalMessage
<<interface>> ClientView
-MyView: LightView
+run(): void
+setupHandler(): void
+messageHandler(): void
+readMessage(): void
checkUsername(nickname: String): void
Used in ClientCLI, ClientGUI
CharacterActivationParser
-name: CharacterName
-chosenIsland: ArrayList<Integer>
-chosenStudent : ArrayList<Integer>
-CurrentPlayer: String
-chosenColor: Col
+buildMessage(): PlayCharacter
Used in Input parser and MainBoardController classes


<<enumeration>>
LightCharacterType
NONE,
INTEGER_1,
INTEGER_2,
COLOR;
LightView
-firstUpdate: boolean
-currentActiveCharacterCard: LightActiveDeck
-currentCharacterDeck: LightCharDeck
-currentIslands: LightIslands
-currentTeams: ArrayList<LightTeam>
-currentMotherNature: LightMotherNature
-currentClouds: LightCloud[]
-currentTurnState: LightTurnState
-bankBalance: int
+updateLightView(light: LightView): void
+parse(view: ViewMessage): void
+addNameToSchools(): void
+findPlayerByName(currentTeams: ArrayList<LightTeam>,player: String): LightPlayer
+setCurrentIslands(currentIslands: LightIslands): void
+setCurrentTeams(currentTeams: ArrayList<LightTeam>): void
+setCurrentMotherNature(currentMotherNature: LightMotherNature): void
+setCurrentClouds(currentClouds: LightCloud[]): void
+setBankBalance(bankBalance: int): void
+getCurrentActiveCharacterCard(): LightActiveDeck
+getCurrentCharacterDeck(): LightCharDeck
+getCurrentIslands(): LightIslands
+getCurrentTeams(): ArrayList<LightTeam>
+getCurrentMotherNature(): LightMotherNature
+getCurrentClouds(): LightCloud[]
+getCurrentTurnState(): LightTurnState
+getBankBalance(): int
1
contains
1
Used in InputParser
Utilities
+findPlayerByName(view: LightView, nickname: String): LightPlayer
+getPlayerTeam(view: LightView, name: String): LightTeam
+getSchoolByName(view: LightView, username: String): LightSchool
Observable
-observers: List<Observer>
-observer1s: List<ObserverLightView>
+addObserver(observer: Observer): void
+addObserverLight(observer: ObserverLightView): void
+removeObserver(observer: Observer): void
+notify(message: String): void
+notifyLight(o:Object): void
That's the class specified in the Observer package
<<enumeration>>
LightCharacterType
NONE,
INTEGER_1,
INTEGER_2,
COLOR;
InternalMessage
-infoMessage: String
+getInfoMessage(): String
LightCloud
-cloudID: int
-students: ArrayList<Student>
+updateCloud(light: LightCloud): void
+getStudents(): ArrayList<Student>
+getCloudID(): int
2,3,4
LightTeam
-color: ColTow
-players: ArrayList<LightPlayer>
-controlledIslands: int
-controlledProfessors: ArrayList<Col>
+updateTeam(light: LightTeam): void
+getColor(): ColTow
+getPlayers(): ArrayList<LightPlayer>
+getControlledIslands(): int
+getControlledProfessors(): ArrayList<Col>
2,3
1
contains
1,2
LightTurnState
-turnOrder: HashMap<String, Integer>
-turn: int
-planningMoves: int
-actionMoves: int
-gamePhase: GamePhase
-lastTurn: boolean
-currentPlayer: String
+updateTurn(light: LightTurnState): void
+getTurnOrder(): HashMap<String, Integer>
+getTurn(): int
+getActionMoves(): int
+getPlanningMoves(): int
+getGamePhase(): GamePhase
+isLastTurn(): boolean
+getCurrentPlayer(): String
1
LightPlayer
-nome: String
-coinAmount: int
-currentAssistantCard: AssistantCard
-lastPlayedCard: AssistantCard
-assistantDeck: AssistantDeck
-school: LightSchool
-maxMotherMovement: int
-movementValue: int
-value: int
-TowerOwner: boolean
+updatePlayer(light: LightPlayer): void
+getName(): String
+getCoinAmount(): int
+getCurrentAssistantCard(): AssistantCard
+getLastPlayedCard(): AssistantCard
+getAssistantDeck(): AssistantDeck
+getSchool():LightSchool
+getMaxMotherMovement(): int
+getMovementValue(): int
+getValue(): int
+isTowerOwner(): boolean
1
contains
1
LightMotherNature
-idPosition: int
+updateMother(light: LightMotherNature): void
+getIdPosition(): int
LightActiveDeck
-lightActiveDeck: ArrayList<LightCharacterCard>
-factory: LightCharcaterFactory
+updateActive(light: LightActiveDeck): void
+getLightActiveDeck(): ArrayList<LightCharacterCard>
LightSchool
-owner: String
-color: ColTow
-entrance: ArrayList<Student>
-diningRoom: int[]
-professorTable: boolean[]
-towerCount: int
+updateSchool(light: LightSchool): void
+updateName(name: String): void
+getOwner(): String
+getColor(): ColTow
+getEntrance(): ArrayList<Student>
+getDiningRoom(): int[]
+getProfessorTable(): boolean[]
+getTowerCount(): int
LightAssistantDeck
-deck: ArrayList<AssistantCard>
-playerName: String
-wizard: Wizard
+getDeck(): ArrayList<AssistantCard>
+getPlayerName(): String
+getWizard(): Wizard
LightCharacterFactory
+characterCreator(card: CharacterCard): LightCharacterCard
1
used by
1
LightCharDeck
-lightCharDeck: ArrayList<LightCharacterCard>
-factory: LightCharacterFactory
+updateCharDeck(light: lightCharDeck): void
+getLightCharDeck(): ArrayList<LightCharacterCard>
+getCard(index: int): LightCharacterCard
1
contains
3
LightCharacterCard
-name: CharacterName
-baseCost: int
-uses: int
-currentCost: int
-noEntry: int
-studentList: ArrayList<Student>
-description: String[]
-type: LightCharacterType
+getName():CharacterName
+getBaseCost(): int
+getUses():int
+getCurrentCost(): int
+getNoEntry():int
+getStudentList(): ArrayList<Student>
+getDescription(): String[]
+getType(): LightCharacterType
LightIslands
-islands: ArrayList<LightIsland>
-setupLock: Object
+updateIslands(light: LightIslands): void
+getIslands(): ArrayList<LightIsland>
+getLock(): Object
1
contains
0,*
LightIsland
-islandId: int
-motherNature: boolean
-Group: boolean
-currentStudents: ArrayList<Student>
-ownership: ColTow
-towerNumber: int
-teamInfluence: int[]
-noEntry: boolean
+updateIsland(island: LightIsland): void
+getIslandId(): int
+isMotherNature(): boolean
+isGroup(): boolean
+getCurrentStudents(): ArrayList<Student>
+getOwnership(): ColTow
+getTowerNumber(): int
+getTeamInfluence(): int[]
+isNoEntry(): boolean


SerializedMessage
-serialVersionUID: long
-command: StandardSetupMessage
-action: StandardActionMessage
+getCommand(): StandardSetupMessage
+getAction(): StandardActionMessage
Used in InputParser, ClientCLI and some Gui's controllers
1
1
1
1
<<Enumeration>>
SETUPMESSAGETYPE
DISCONNECTION,
MODE,
READINESS,
CONNECTION_SETUP,
WIZARD_CHOICE,
TEAM_CHOICE, PONG
<<Interface>> Message
<<Interface>>
Serializable
<<Enumeration>>
ACTIONMESSAGETYPE
CLOUD_CHOICE,
DRAW_CHOICE,
STUD_MOVE,
MN_MOVE,
CHARACTER_PLAY,
CHARACTER_ACTIVATE,
ENTRANCE_REFILL,
TURN_END
{Abstract} StandardSetupMessage
+type: SETUPMESSAGETYPE
+getType(): SETUPMESSAGETYPE
{Abstract} StandardActionMessage
+type: ACTIONMESSAGETYPE
+getType(): ACTIONMESSAGETYPE
Disconnect
-disconnecting: boolean
+isDisconnecting(): boolean
Readystatus
TeamChoice
-team: int
+getTeam(): int
GameMode
-expertGame: boolean
-maxPlayers: int
+getMaxPlayers(): int
+isExpertGame(): boolean
+setMaxPlayers(maxPlayers: int): void
+setExpertGame(expertGame: boolean): void
SetupConnection
-nickname: String
-team: int
+getNickname(): String
+getTeam(): int
+setTeam(team: int): void
+setNickname(nickname: String):
WizardChoice
-wizard: Wizard
+getWizard(): Wizard
ChooseCloud
-cloudIndex: int
+getCloudIndex(): int
DrawAssistantCard
-cardIndex: int
+getCardIndex(): int
DrawFromPouch
-cloudIndex: int
+getCloudIndex(): int
MoveMN
-amount: int
+getAmount():int
PlayCharacter
-name: CharacterName
-student: ArrayList<Integer>
-island: ArrayList<Integer>
-currentPlayer: String
-studentColor: Col
+getCharacterName(): CharacterName
+getFirst(): ArrayList<Integer>
+getSecond: ArrayList<Integer>
+getThird(): String
+getStudentColor(): Col
MoveStudent
-entrancePos: int
-toIsland: boolean
-islandId: int
+getEntrancePos(): int
+isToIsland(): boolean
+getIslandId(): int
EndTurn
PlayCharacterEffect
-name: CharacterName
-student: int
-island: int
-currentPlayer: String
-studentColor: Col
+getCharacterName(): CharacterName
+getFirst(): int
+getSecond: int
+getThird(): String
+getStudentColor(): Col


Alse used in ClientApp and main
ClientCLI
-main: ServerConnection
-stdin: InputParser
-setupState: boolean
-setupLock: Object
-info: Scanner
-executor: ExecutorService
+setupHandler(answer: StandardSetupAnswer): void
+messageHandler(answer: StandardActionAnswer): void
+readMessage(): void
+run(): void
+getMain(): ServerConnection
+getSetupState(): boolean
1
uses
1
<<Interface>>
Runnable
ListenerCLI
-client: ClientCLI
+run(): void
InputParser
-parser: Scanner
-playerName: String
-socket: ServerConnection
-boardPrinters: BoardPrinters
-schoolPrinter: SchoolPrinter
-infoPrinters:InfoPrinters
-cardPrinters: CardPrinters
-printView: Boolean
+getParser(): Scanner
+drawParser(words: String[]):void
+moveParser(words: String[]): void
+MinstrelParser(player: LightPlayer, inputArray1: ArrayList<Integer>, inputArray2: ArrayList<Integer>): void
+JesterParser(card: LightCharacterCard, player: LightPlayer, inputArray1: ArrayList<Integer>, inputArray2: ArrayList<Integer>): void
+characterParser(words: String[]): void
+showView(words: String[]): void
+parseString(input: String): void
+newMove(): void
+printGame(): void
+integerParser(): Integer
+studentColorParser(): Col
+resetScreen(): void
1
uses
1
{Abstract} PrinterCLI
-view: LightView
+ANSI_RESET: String
+ANSI_BLACK: String
+ANSI_RED: String
+ANSI_GREEN: String
+ANSI_YELLOW: String
+ANSI_BLUE: String
+ANSI_PURPLE: String
+ANSI_CYAN: String
+ANSI_WHITE: String
+ANSI_RED_BACKGROUND: String
+ANSI_GREEN_BACKGROUND: String
+addZero(X: int): String
+cls(): void
+printTitle(): void
+nameTrimmer(name: String): String
+convertTo3Char(towerColor: ColTow): String
+printStudent(students: ArrayList<Student>, spaces: int)): String
+getPlayerByName(name: Strung): LightPlayer
+getView(): LightView
+setView(view: LightView): void
BoardPrinters
+printMN(id: int): String
+adddNoEntry(): String
+printIsland(island: String[], id: int): String[]
+showIsland(id: int): void
+printStudentOnCloud(students: ArrayList<Student>): String
+showCloud(): void
CardPrinters
+showAssistantDeck(nickname: String): void
+printPlayedAssitant(card: String[], currentPlayer: boolean, name: String, assistantCard: AssistantCard): void
+showPlayedCards(): void
+printCharacter(card: LightCharacterCard, character: String[], index: int): String[]
+showCharacters(): void
+characterDifferentiate(card: LightCharacterCard): String
InfoPrinters
+showPlayer(name: String): void
+showPlayers(): void
+printEconomy(playerName: String): void
+showHelp(): void
+controlledIslands(team: LightTeam): ArrayList<String>
+printTurn(): void
+printAvaiableColors(diningRoom: int[]): ArrayList<Col>
SchoolPrinters
+printEntrance(nome: String): String
+printDinnerTable(nome: String,dinnerPosition: int): String
+printHasProf(nome: String, tablePosition: int): String
+printSchool(schools: String[], player: LightPlayer, currentPlayer: String): String[]
+showSchool(name: String, curretnPlayer: String): void


Used in ClientApp
Client Gui
-guiMainStarter: GuiMainStarter
-serverConnection : ServerConnection
-setupState : Boolean
-executor : ExecutorService
-listenerGui: ListenerGui
-firstView: Boolean
-mbc: MainBoardController
-ANSI_RESET : String
-ANSI_GREEN : String
+run(): void
+changescene(loader: FXMLLoader): void
+setupHandler(answer: StandardSetupAnswer) : void
+readMessage() : void
+messageHandler(answer: StandardActionAnswer): void
+setServerConnection(nickname: String, IP: String): void
+getServerConnection(): ServerConnection
GuiMainStarter
-ClientGui : ClientGUI
-mainStage : Stage
+main(): void
+start(stage: Stage): void
+setClientGUI(clientGUI: ClientGUI): void
+getClientGUI(): ClientGUI
+getMainStage(): Stage
{Abstract} Controller
-guiMainStarter : GuiMainStarter
+loadNewScreen(path: String): FXMLLoader
+setGuiMainStarter(): void
ListenerGUI
-client: ClientGUI
+run(): void
GuiUtilities
+getRightColorPath(): String
+getSchoolColorPath(): String
+getCellFromGridPane(matrix: GridPane, column: int, row: int): Node
IntroController
+onClickExitButton(ActionEvet):void
+onClickStart(): void
+onClickStory(): void
CloudController
-turn: LightTurnState
-cloud: GridPane
-cloud0: GridPane
-cloud1: GridPane
-cloud2: GridPane
-cloud3: GridPane
+CloudSelection(event: MouseEvent):void
+setup(cloudsArray: LightCloud[], turnState: LightTurnState): void
+getCellFromGridPane(matrix: GridPane, column: int, row:int): void
+getRightColorPath(s: Student): String
+fill(fillable: GridPane, filler: LightCloud):void
+update(Object o): void
LoginController
-nickname: TextField
-IP: TextField
-port: TextField
-tryConn: Button
+inizialize(): void
+onClickITryConnection(ActionEvent: MouseEvent): void
LobbyController
-playersChoice: ChoiceBox
-difficultyChoice: ChoiceBox
+inizialize(): void
+onClickTryConnection(): void
WinnerController
-title1: Text
-backToMenu: Button
+onClickQuit(event: MouseEvent):void
+setup(text: String): void
WizardController
-rb1: RadioButton
-rb2: RadioButton
-rb3: RadioButton
-rb4: RadioButton
-druid: Pane
-witch: Pane
-sensei: Pane
-lord: Pane
-alreadyChoose1: Text
-alreadyChoose2: Text
-alreadyChoose3: Text
-alreadyChoose4: Text
-paneWizard: ArrayList<Pane>
-textAvaiable: ArrayList<Text>
-group: ToggleGroup
+initiliaze(): void
+onClickRb1(): void
+onClickRb2(): void
+onClickRb3(): void
+onClickRb4(): void
+onClickSendChoice(): void
+updateOpacity(avaiable: ArrayList<Wizard>): void
MineSchoolController
-studentChoice: Pane
-islands: ChoiceBox
-toDining: Button
-toIsland: Button
-MySchool: AnchorPane
-studentEntrancePos: int
- view: LightView
+setup(): void
+entranceClick(event: MouseEvent): void
+sendDining(event: MouseEvent): void
+sendIsland (event: MouseEvent): void
+getRightColorPath(s: Student): String
+getSchoolColorPath(school: LightSchool): String
+getCellFromGridPane(matrix: GridPane, column: int, row: int): Pane
+update(o: Object): void
ReadyController
-tickPane: Pane
+onClickReadyButton(): void
OtherSchoolController
-APotherSchool: AnchorPane
-teams: ArrayList<LightTeam>
-lookedPlayer: String
-view: LightView
+onClick(event: ActionEvent): void
+setup(teams: ArrayList<LightTeam>,otherSchool: AnchorPane): void
+getSchoolByName(username: String): LightSchool
+getRightColorPath(s: Student): String
+getSchoolColorPath(school: LightSchool): String
+getCellFromGridPane(matrix: GridPane, column: int, row: int): Pane
+update(o: Object): void
DescriptionController
+onClickBackToIntro(): void
TeamController
-AvaiableTeams: HBox
+getTeamColorPath(: int):String
+init(avaiableTeams: int[]): void
MainBoardController
-mainAnchorPane: AnchorPane
-otherSchoolAnchorPane: AnchorPane
-characterAnchorPane: AnchorPane
-islandAnchorPane: AnchorPane
-cloudsAnchorPane: AnchorPane
-assistantCardAnchorPane: AnchorPane
-buttonAreaAnchorPane: AnchorPane
-mineSchoolAnchorPane: AnchorPane
-PlayedAssistants: AnchorPane
-EffectPane: Pane
-EffectDescription: Text
-ParametersSlice: Pane
-PlayButton: Button
-BackToBoard: Button
-PlayedAssistants: GridPane
-ErrorMessage: Text
-ErrorDisplay: Pane
-CloseError: Button
-integerChoice_1: ArrayList<Integer>
-integerChoice_2: ArrayList<Integer>
-colorChoice: Col
-view: LightView
+initialSetupIsland(view: LightView):void
+initialSetupAssistantCard(teams: ArrayList<LightTeam>): void
+initialSetupCharacterCard(charDeck LightCharDeck, LightActiveDeck): void
+initialSetupMineSchool(lightTeams: ArrayList<LightTeam>): void
+initialSetupPropaganda(view: LightView, infos: InfoDispenser): void
+initialSetupOtherSchool(lightTeams: ArrayList<LightTeams>): void
+displayCharInfo(card: CharacterCard, path: String): void
+hideCharacterInfo(mouseEvent: MouseEvent): void
+showPlayedAssistants(path: String, column: int, row: int, player: LightPlayer, hide: boolean): void
+showCharacterEffectPanel(card: LightCharacterCard): void
+GrandmaHerbsSetup(card: LightCharacterCard): void
+PrincessSetup(card: LightCharacterCard): void
+ColorSetup(): void
+Integer1Setup(): void
+Integer2Setup(card: LightCharacterCard): void
+JesterSetup(card: LightCharacterCard, player: LightPlayer): void
+MinistrelSetup(player: LightPlayer): void
+minstrelColorChoice(event: MouseEvent): void
+setDiningRoom(dining: int[]): void
+DisplayError(error: String): void
+setup(): void
PropagandaController
-propaganda: AnchorPane
-endTurn: Button
-exit: Button
-view: LightView
+hintGeneration()
-exitOnClick(mouseEvent: MouseEvent): void
-endTurnOnClick(mouseEvent: MouseEvent): void
+setup(view: LightView): void
+update(o: Object): void
RejectionController
-backToMenu: Button
+OnClickMenu(event: MouseEvent): void
+setup(): void
AssistantCardController
-assistantsAnchorPane: AnchorPane
-lastPlayedButton: Button
-teams: ArrayList<LightTeam>
-players: ArrayList<LightPlayer>
-played: ArrayList<AssistantCard>
-currentPlayer:String
-mainController: MainBoardController
-assistantsOnClick(event: MouseEvent): void
+getAssistantPath(card: AssistantCard): String
+setup(currentPlayer: String, teams: ArrayList<LightTeam>, AssistantsPane: AnchorPane, controller: MainBoardController): void
+update(o: Object): void
CharacterCardController
-CharacterPane: AnchorPane
-characterDeck:LightCharDeck
-activeCharDeck: LightActiveDeck
-sceneCards: ArrayList<CharacterCard>
-currentlyShowedCard:int
-mainController: MainBoardController
-mainPane: StackPane
-PreviousButton: Button
-NextButton: Button
-ActivateButton: Button
+setup(characterPane: AnchorPane, inactiveCharacters: LightCharDeck, activeCharacters: LightActiveDeck, controller: MainBoardController) : void
-activateOnClick(mouseEvent: MouseEvent):void
-nextOnClick(mouseEvent: MouseEvent)
-previousOnClick(mouseEvent: MouseEvent)
-showOnClick(mouseEvent: MouseEvent)
+update(o: Object): void
+getCorrectPane(card: LightCharacterCard): Pane
+getCardPath(name: CharacterName): String
IslandsController
-cloudsAnchorPane: AnchorPane
-mainIslandBoard: AnchorPane
+islands: ArrayList<AnchorPane>
+islandControllers: ArrayList<IslandController>
+radius: int
-view: LightView
+setup(anchorPane: AnchorPane, view: LightView): void
+update(o: Object): void
IslandController
-stud_green: Imageview
-stud_blue: Imageview
-stud_red: Imageview
-stud_yellow: Imageview
-stud_pink: Imageview
-islandImage: Imageview
-blueStudents: Text
-redStudents: Text
-pinkStudents: Text
-yellowStudents: Text
-greenStudents: Text
-towerNumber: Text
-motherNature: ImageView
-noEntry: ImageView
-islandPane: AnchorPane
-ownerShip: Pane
-islandID: int
-MN: LightMotherNature
-totalIslands: int
+onClick(event: MouseEvent):void
+studentPlacing(island: LightIsland): void
+ setup(island: LightIsland, ID: int, Mother: LightMotherNature, total: int): void


Client
Server
GameHandler
SetupConnection
OPT
[If first client]
RequestGameInfo
GameMode
ALT
[If all clients have connected]
Start
Loop
[Until client has Choosen a Wizard and given the ReadyStatus]
AvailableTeams
TeamChoice
AvailableWizards
WizardChoice
Ready
ALT
When all players are ready
GameStarting
ALT
[If we are in planning phase]
DrawFromPouch
DrawAssistantCard
ALT
[If we are in Action phase]
3 MoveStudent if we are playing a 2/4 player game
4 MoveStudent if we are playing a 3 player game
MoveStudent
MoveMN
ChooseCloud
EndTurn
OPT
[At any time during the action phase]
PlayCharacter
OPT
[On every model update]
SendView
OPT
[If a wrong action is sent from client]
ErrorMessage
OPT
[If one player disconnects]
InfoMessage
WinnerMessage
OPT
[If a winner is detected]
WinnerMessage