Model

AssistantCard
- playerVer : int
- numberStep : int
+ getPlayerValue() : int
+ getNumberStep() : int
+ setNumberSteps() : void

Assistant Deck
- cards : ArrayList<AssistantCard>
- wizardName : AssistantName
+ chosenCard(name : String) : Assistant Card
+ getCards() : ArrayList<Assistant Card>
+ remove(card : Assistant Card) : boolean
+ getWizardName() : AssistantName

<<enumeration>> AssistantName
- KING : enum
- WITCH : enum
- VETERAN : enum
- MAGICIAN : enum

Player
- nickname : String
- color : PlayerColor
- boardPlayer : Board
- portfolio : int
- cards : AssistantDeck
- lastCardUsed : AssistantCard
- influence : int
+ getColor() : PlayerColor
+ getNickname() : String
+ addCoin() : void
+ getBoardPlayer() : Board
+ setCards(cards : AssistantDeck) : void
+ getCards() : AssistantDeck
+ getLastCardUsed() : AssistantCard
+ setLastCardUsed(assistantCardChosen : AssistantCard) : void
+ getPortfolio() : int
+ setPortfolio(portfolio : int) : void
+ removeCard(cardChosen : AssistantCard) : void
+ calculateInfluence(island : Island) : void
+ addCoin() : void
+ getInfluence() : int
+ setInfluence(influence : int) : void

Game
- players : ArrayList<Player>
- bag : HashMap<Color,Integer>
- coins : int
- islands : ArrayList<Island>
- motherNaturePos : int
- characterCards : ArrayList<CharacterCard>
- clouds : ArrayList<HashMap<Color,Integer>>
- numOfMerge : int
- professors : ArrayList<Color>
- effectInfluence : boolean
- sizeEntrance : int
- currentPlayerIndex : int
+ setNextPlayer() : void
+ getActivePlayer() : Player
+ setActivePlayer() : void
+ shiftMotherNature(numOfSteps : int) : void
+ removeCharacterCard(card : CharacterCard) : boolean
+ drawFromBag(numOfStudents : int) : HashMap<Color,Integer>
+ moveTowerToIsland(p : Player, islandID : int) : void
+ moveStudentToCloud(id : int, numOfStudentsToMove : int) : void
+ getSizeEntrance() : int
+ setSizeEntrance(sizeEntrance : int) : void
+ setCharacterCards(characterCards : ArrayList<CharacterCard>) : void
+ getProfessors() : ArrayList<Color>
+ getCharacterCards() : ArrayList<CharacterCard>
+ getIslands() : ArrayList<Island>
+ setMotherNaturePos(motherNaturePos : int) : void
+ getMotherNaturePos() : int
+ setEffectInfluence(effectInfluence : boolean) : void
+ setPlayers(players : ArrayList<Player>) : void
+ createNewPlayer(nickname : String, color : PlayerColor) : boolean
+ getClouds() : ArrayList<HashMap<Color,Integer>>
+ setNumberClouds(num : int) : void
+ getCoins() : int
+ setCoins(coins : int) : void
+ getCurrentPlayerIndex() : int
+ getPlayers() : ArrayList<Player>
+ calculateInfluence() : void
+ decideInfluence() : void
+ getNumOfMerge() : int
+ getActiveIsland() : Island
+ merge(island1ID : int, island2ID : int) : void
+ emptyBag() : boolean
+ getBag() : HashMap<Color,Integer>
+ setBag(bag : HashMap<Color,Integer>) : void
+ operation25() : void
+ studentsFromCloudToEntrance(id : int) : void
+ operation27() : void
+ moveStudentToEntrance(studentColor : Color) : boolean
+ moveStudentToHall(studentColor : Color) : boolean
+ moveStudentToIsland(studentColor : Color, id : int) : boolean

<<enumeration>> PlayerColor
- BLACK : enum
- WHITE : enum
- GREY : enum

Board
- halls : HashMap<Color,Integer>
- entrance : HashMap<Color,Integer>
- professors : ArrayList<Color>
- towers : int
+ checkTowerSpaceIsEmpty() : boolean
+ getTowers() : int
+ getHall(color : Color) : int
+ insertStudentEntrance(studentColor : Color) : void
+ getNumberOfStudents() : int
+ getColor() : Color
+ getProfessors() : ArrayList<Color>
+ setProfessors(professors : ArrayList<Color>) : void
+ insertProf(professor : Color) : void
+ removeProf(professor : Color) : boolean
+ getSizeProfessors() : int
+ checkProfessorTableIsEmpty() : boolean
+ getEntrance() : HashMap<Color,Integer>
+ removeStudentEntrance(studentColor : Color) : boolean
+ operation9() : void
+ setHall(colorHall : Color, studentToAdd : int) : void
+ setTowers(towers : int) : void
+ addTowers(numOfTowers : int) : void
+ removeTowers(numOfTowers : int) : boolean

Character Deck
- deck : ArrayList<CharacterCard>
+ mixDeck() : void
+ draw3Card() : void
+ getCards() : ArrayList<CharacterCard>

<> CharacterCard
- price : int
- name : CharacterName
- numOfStudents : int
- numOfBanTile : int
- source : ArrayList<Color>
- destination : ArrayList<Color>
- studentColorForEffects : Color
- islandChosen : int
- students : HashMap<Color,Integer>
+ getName() : CharacterName
+ doAction() : void
+ getPrice() : int
+ setPrice(price : int) : void
+ setName(name : CharacterName) : void
+ setNumOfStudents(numOfStudents : int) : void
+ getNumOfStudents() : int
+ getStudents() : HashMap<Color,Integer>
+ setStudents(students : HashMap<Color,Integer>) : void
+ getNumOfBanTile() : int
+ setNumOfBanTile(numOfBanTile : int) : void
+ getStudentColorForEffects() : Color
+ setStudentColorForEffects(studentColorForEffects : Color) : void
+ getSource() : ArrayList<Color>
+ setSource(source : ArrayList<Color>) : void
+ getDestination() : ArrayList<Color>
+ setDestination(destination : ArrayList<Color>) : void
+ getIslandChosen() : int
+ setIslandChosen(islandChosen : int) : void

<<enumeration>> CharcaterName
- CENTAUR : enum
- FLORIST : enum
- FRIAR : enum
- INNKEEPER : enum
- JOLLY : enum
- KNIGHT : enum
- MESSENGER : enum
- MUSHROOMMAN : enum
- PIRATE : enum
- PRINCESS : enum
- THIEF : enum
- VIOLINIST : enum

Island
- students : HashMap<Color,Integer>
- id : int
- numberOfLink : int
- towers : int
- colorOwner : PlayerColor
- banTile : boolean
+ getNumberOfStudent() : int
+ getStudents() : map<Color,int>
+ getId() : int
+ getOwnerIsland() : Player
+ insertStudent(studentColor : Color) : void
+ insertTower(towerColor : PlayerColor) : void
+ removeTower() : void
+ getTowers() : int
+ getStudents() : HashMap<Color,Integer>
+ setNumberOfLink(numberOfLink : int) : void
+ operation33() : void
+ setTowers() : void
+ getOwner() : PlayerColor
+ setOwner(colorOwner : PlayerColor) : void
+ resetTowers() : void
+ getBanTile() : boolean
+ setBanTile(banTile : boolean) : void

Cloud
- students : map<Color,int>
+ getCloud() : int
+ getStudents() : map<Color,int>
+ insertStudent(studentColor : Color) : void
+ removeStudents() : boolean

<<enumeration>> Color
- YELLOW : enum
- PURPLE : enum
- RED : enum
- GREEN : enum
- BLUE : enum

Knight
+ doAction() : void

Princess
+ doAction() : void

Jolly
+ doAction() : void

Thief
+ doAction() : void

MushroomsMan
+ doAction() : void

Violinist
+ doAction() : void

Florist
+ doAction() : void

Centaur
+ doAction() : void

Messenger
+ doAction() : void

Pirate
+ doAction() : void

Innkeeper
+ doAction() : void

Friar
+ doAction() : void