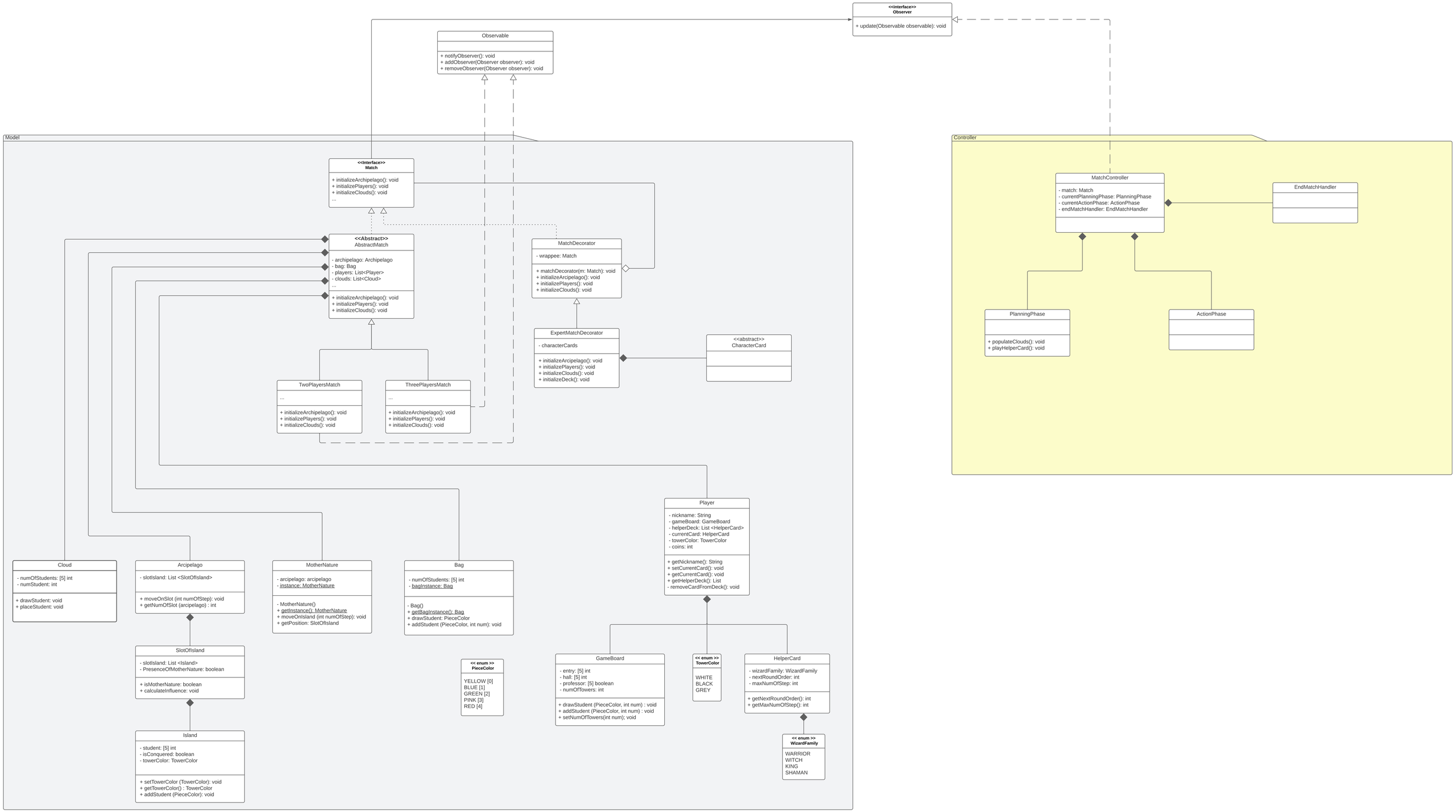
<<interface>>
Observer
+ update(Observable observable): void
Observable
+ notifyObserver(): void
+ addObserver(Observer observer): void
+ removeObserver(Observer observer): void
Model
<<Interface>>
Match
+ initializeArchipelago(): void
+ initializePlayers(): void
+ initializeClouds(): void
...
<<Abstract>>
AbstractMatch
- archipelago: Archipelago
- bag: Bag
- players: List<Player>
- clouds: List<Cloud>
...
+ initializeArchipelago(): void
+ initializePlayers(): void
+ initializeClouds(): void
MatchDecorator
- wrappee: Match
+ matchDecorator(m: Match): void
+ initializeArcipelago(): void
+ initializePlayers(): void
+ initializeClouds(): void
ExpertMatchDecorator
- characterCards
+ initializeArcipelago(): void
+ initializePlayers(): void
+ initializeClouds(): void
+ initializeDeck(): void
<<abstract>>
CharacterCard
TwoPlayersMatch
...
+ initializeArchipelago(): void
+ initializePlayers(): void
+ initializeClouds(): void
ThreePlayersMatch
...
+ initializeArchipelago(): void
+ initializePlayers(): void
+ initializeClouds(): void
Cloud
- numOfStudents: [5] int
- numStudent: int
+ drawStudent: void
+ placeStudent: void
Arcipelago
- slotIsland: List <SlotOfIsland>
+ moveOnSlot (int numOfStep): void
+ getNumOfSlot (arcipelago) : int
MotherNature
- arcipelago: arcipelago
- instance: MotherNature
- MotherNature()
+ getInstance(): MotherNature
+ moveOnIsland (int numOfStep): void
+ getPosition: SlotOfIsland
Bag
- numOfStudents: [5] int
- bagInstance: Bag
- Bag()
+ getBagInstance(): Bag
+ drawStudent: PieceColor
+ addStudent (PieceColor, int num): void
Player
- nickname: String
- gameBoard: GameBoard
- helperDeck: List <HelperCard>
- currentCard: HelperCard
- towerColor: TowerColor
- coins: int
+ getNickname(): String
+ setCurrentCard(): void
+ getCurrentCard(): void
+ getHelperDeck(): List
- removeCardFromDeck(): void
SlotOfIsland
- slotIsland: List <Island>
- PresenceOfMotherNature: boolean
+ isMotherNature: boolean
+ calculateInfluence: void
Island
- student: [5] int
- isConquered: boolean
- towerColor: TowerColor
+ setTowerColor (TowerColor): void
+ getTowerColor() : TowerColor
+ addStudent (PieceColor): void
<< enum >>
PieceColor
YELLOW [0]
BLUE [1]
GREEN [2]
PINK [3]
RED [4]
GameBoard
- entry: [5] int
- hall: [5] int
- professor: [5] boolean
- numOfTowers: int
+ drawStudent (PieceColor, int num) : void
+ addStudent (PieceColor, int num) : void
+ setNumOfTowers(int num); void
<< enum >>
TowerColor
WHITE
BLACK
GREY
HelperCard
- wizardFamily: WizardFamily
- nextRoundOrder: int
- maxNumOfStep: int
+ getNextRoundOrder(): int
+ getMaxNumOfStep(): int
<< enum >>
WizardFamily
WARRIOR
WITCH
KING
SHAMAN
Controller
MatchController
- match: Match
- currentPlanningPhase: PlanningPhase
- currentActionPhase: ActionPhase
- endMatchHandler: EndMatchHandler
EndMatchHandler
PlanningPhase
+ populateClouds(): void
+ playHelperCard(): void
ActionPhase