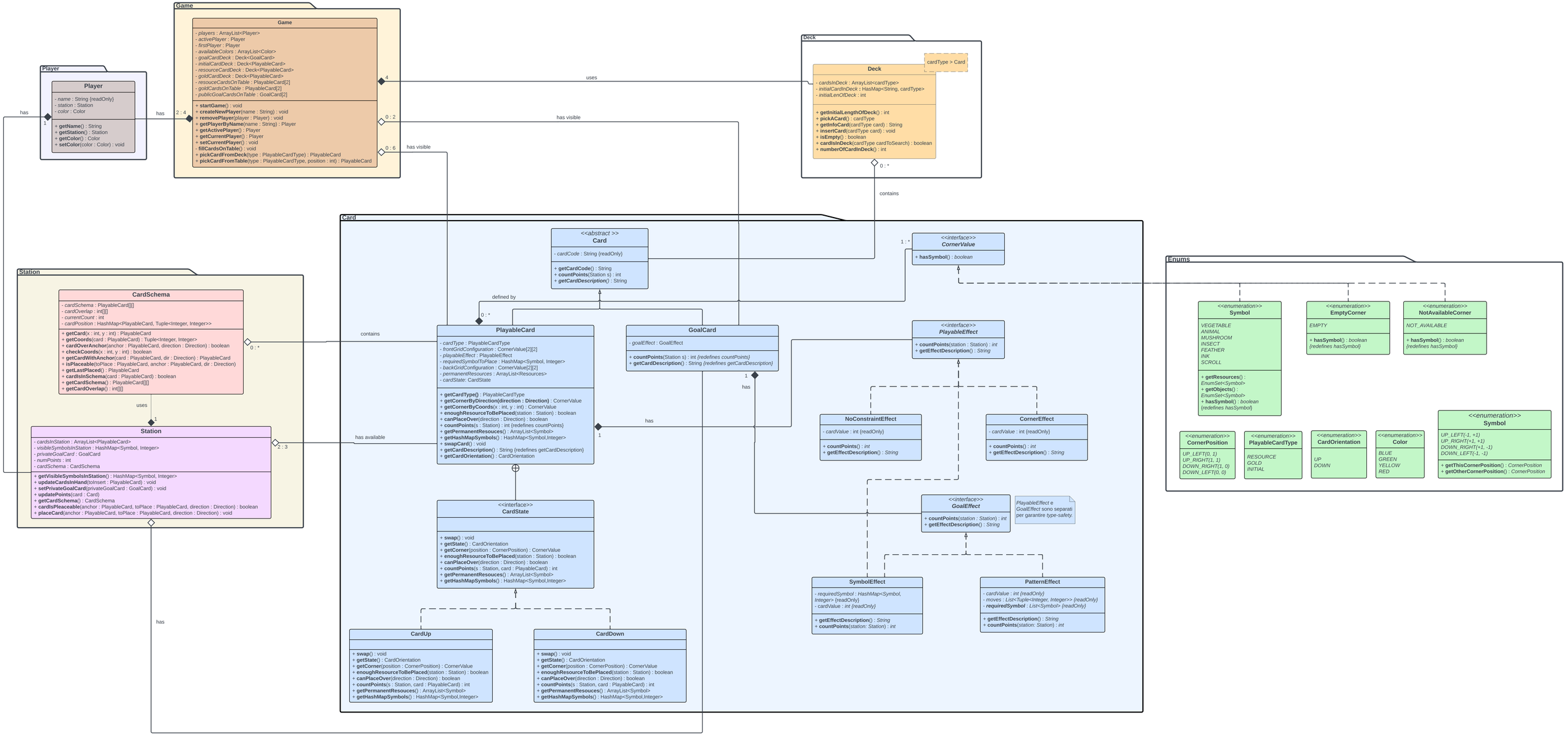
Game
Game
- players : ArrayList<Player>
- activePlayer : Player
- firstPlayer : Player
- availableColors : ArrayList<Color>
- goalCardDeck : Deck<GoalCard>
- initialCardDeck : Deck<PlayableCard>
- resourceCardDeck : Deck<PlayableCard>
- goldCardDeck : Deck<PlayableCard>
- resoucecCardsOnTable : PlayableCard[2]
- goldCardsOnTable : PlayableCard[2]
- publicGoalCardsOnTable : GoalCard[2]
+ startGame() : void
+ createNewPlayer(name : String) : void
+ removePlayer(player : Player) : void
+ getPlayerByName(name : String) : Player
+ getActivePlayer() : Player
+ getCurrentPlayer() : Player
+ setCurrentPlayer() : void
- fillCardsOnTable() : void
+ pickCardFromDeck(type : PlayableCardType) : PlayableCard
+ pickCardFromTable(type : PlayableCardType, position : int) : PlayableCard
Player
Player
- name : String {readOnly}
- station : Station
- color : Color
+ getName() : String
+ getStation() : Station
+ getColor() : Color
+ setColor(color : Color) : void
has
1
has
2 : 4
4
uses
Deck
cardType > Card
Deck
- cardsInDeck : ArrayList<cardType>
- initialCardInDeck : HasMap<String, cardType>
- initialLenOfDeck : int
+ getInitialLengthOfDeck() : int
+ pickACard() : cardType
+ getInfoCard(cardType card) : String
+ insertCard(cardType card) : void
+ isEmpty() : boolean
+ cardIsInDeck(cardType cardToSearch) : boolean
+ numberOfCardInDeck() : int
0 : 2
has visible
0 : 6
has visible
0 : *
contains
Card
<<abstract >>
Card
- cardCode : String {readOnly}
+ getCardCode() : String
+ countPoints(Station s) : int
+ getCardDescription() : String
1 : *
<<interface>>
CornerValue
+ hasSymbol() : boolean
defined by
0 : *
contains
0 : *
PlayableCard
- cardType : PlayableCardType
- frontGridConfiguration : CornerValue[2][2]
- playableEffect : PlayableEffect
- requiredSymbolToPlace : HashMap<Symbol, Integer>
- backGridConfiguration : CornerValue[2][2]
- permanentResources : ArrayList<Resources>
- cardState: CardState
+ getCardType() : PlayableCardType
+ getCornerByDirection(direction : Direction) : CornerValue
+ getCornerByCoords(x : int, y : int) : CornerValue
+ enoughResourceToBePlaced(station : Station) : boolean
+ canPlaceOver(direction : Direction) : boolean
+ countPoints(s : Station) : int {redefines countPoints}
+ getPermanentResouces() : ArrayList<Symbol>
+ getHashMapSymbols() : HashMap<Symbol,Integer>
+ swapCard() : void
+ getCardDescription() : String {redefines getCardDescription}
+ getCardOrientation() : CardOrientation
GoalCard
- goalEffect : GoalEffect
+ countPoints(Station s) : int {redefines countPoints}
+ getCardDescription() : String {redefines getCardDescription}
1
has
<<interface>>
PlayableEffect
+ countPoints(station : Station) : int
+ getEffectDescription() : String
NoConstraintEffect
- cardValue : int {readOnly}
+ countPoints() : int
+ getEffectDescription() : String
CornerEffect
- cardValue : int {readOnly}
+ countPoints() : int
+ getEffectDescription() : String
has
1
<<interface>>
GoalEffect
+ countPoints(station : Station) : int
+ getEffectDescription() : String
PlayableEffect e GoalEffect sono separati per garantire type-safety.
SymbolEffect
- requiredSymbol : HashMap<Symbol, Integer> {readOnly}
- cardValue : int {readOnly}
+ getEffectDescription() : String
+ countPoints(station: Station) : int
PatternEffect
- cardValue : int {readOnly}
- moves : List<Tuple<Integer, Integer>> {readOnly}
- requiredSymbol : List<Symbol> {readOnly}
+ getEffectDescription() : String
+ countPoints(station: Station) : int
<<interface>>
CardState
+ swap() : void
+ getState() : CardOrientation
+ getCorner(position : CornerPosition) : CornerValue
+ enoughResourceToBePlaced(station : Station) : boolean
+ canPlaceOver(direction : Direction) : boolean
+ countPoints(s : Station, card : PlayableCard) : int
+ getPermanentResouces() : ArrayList<Symbol>
+ getHashMapSymbols() : HashMap<Symbol,Integer>
CardUp
+ swap() : void
+ getState() : CardOrientation
+ getCorner(position : CornerPosition) : CornerValue
+ enoughResourceToBePlaced(station : Station) : boolean
+ canPlaceOver(direction : Direction) : boolean
+ countPoints(s : Station, card : PlayableCard) : int
+ getPermanentResouces() : ArrayList<Symbol>
+ getHashMapSymbols() : HashMap<Symbol,Integer>
CardDown
+ swap() : void
+ getState() : CardOrientation
+ getCorner(position : CornerPosition) : CornerValue
+ enoughResourceToBePlaced(station : Station) : boolean
+ canPlaceOver(direction : Direction) : boolean
+ countPoints(s : Station, card : PlayableCard) : int
+ getPermanentResouces() : ArrayList<Symbol>
+ getHashMapSymbols() : HashMap<Symbol,Integer>
Station
CardSchema
- cardSchema : PlayableCard[][]
- cardOverlap : int[][]
- currentCount : int
- cardPosition : HashMap<PlayableCard, Tuple<Integer, Integer>>
+ getCard(x : int, y : int) : PlayableCard
+ getCoords(card : PlayableCard) : Tuple<Integer, Integer>
+ cardOverAnchor(anchor : PlayableCard, direction : Direction) : boolean
+ checkCoords(x : int, y : int) : boolean
+ getCardWithAnchor(card : PlayableCard, dir : Direction) : PlayableCard
+ isPlaceable(toPlace : PlayableCard, anchor : PlayableCard, dir : Direction)
+ getLastPlaced() : PlayableCard
+ cardIsInSchema(card : PlayableCard) : boolean
+ getCardSchema() : PlayableCard[][]
+ getCardOverlap() : int[][]
uses
1
Station
- cardsInStation : ArrayList<PlayableCard>
- visibleSymbolsInStation : HashMap<Symbol, Integer>
- privateGoalCard : GoalCard
- numPoints : int
- cardSchema : CardSchema
+ getVisibleSymbolsInStation() : HashMap<Symbol, Integer>
+ updateCardsInHand(toInsert : PlayableCard) : void
+ setPrivateGoalCard(privateGoalCard : GoalCard) : void
+ updatePoints(card : Card)
+ getCardSchema() : CardSchema
+ cardIsPleaceable(anchor : PlayableCard, toPlace : PlayableCard, direction : Direction) : boolean
+ placeCard(anchor : PlayableCard, toPlace : PlayableCard, direction : Direction) : void
2 : 3
has available
has
Enums
<<enumeration>>
Symbol
VEGETABLE
ANIMAL
MUSHROOM
INSECT
FEATHER
INK
SCROLL
+ getResources() : EnumSet<Symbol>
+ getObjects() : EnumSet<Symbol>
+ hasSymbol() : boolean {redefines hasSymbol}
<<enumeration>>
EmptyCorner
EMPTY
+ hasSymbol() : boolean {redefines hasSymbol}
<<enumeration>>
NotAvailableCorner
NOT_AVAILABLE
+ hasSymbol() : boolean {redefines hasSymbol}
<<enumeration>>
CornerPosition
UP_LEFT(0, 1)
UP_RIGHT(1, 1)
DOWN_RIGHT(1, 0)
DOWN_LEFT(0, 0)
<<enumeration>>
PlayableCardType
RESOURCE
GOLD
INITIAL
<<enumeration>>
CardOrientation
UP
DOWN
<<enumeration>>
Color
BLUE
GREEN
YELLOW
RED
<<enumeration>>
Symbol
UP_LEFT(-1, +1)
UP_RIGHT(+1, +1)
DOWN_RIGHT(+1, -1)
DOWN_LEFT(-1, -1)
+ getThisCornerPosition() : CornerPosition
+ getOtherCornerPosition() : CornerPosition