pkg it.polimi.ingsw.model

<<enum>> ActionMarker
- <<enum constant>> DISCARD_GREEN : char
- <<enum constant>> DISCARD_YELLOW : char
- <<enum constant>> DISCARD_BLUE : char
- <<enum constant>> DISCARD_PURPLE : char
- <<enum constant>> MOVE_TWICE : char
- <<enum constant>> MOVE_AND_SHUFFLE : char

<<enum>> Resource
- <<enum constant>> SHIELD : char
- <<enum constant>> SERVANT : char
- <<enum constant>> STONE : char
- <<enum constant>> GOLD : char

<<enum>> BannerEnum
- <<enum constant>> GREEN : char
- <<enum constant>> YELLOW : char
- <<enum constant>> BLUE : char
- <<enum constant>> PURPLE : char

Game
- gameId : String
- currentPlayer : Player
+ init() : void
+ checkEnd() : boolean
+ checkWinner() : Player
+ checkDiscardedResources() : boolean

GameSettings
- DEVELOPMENT_CARD_NUM : int
- FAITH_TRACK_LENGTH : int
- developmentCards : DevelopmentCard[]
- leaderCardNum : int
- dashboardProductionPower : ProductionPower
- vaticanReportsNum : int
- vaticanReports : List<VaticanReport>
- faithTrackVictoryPoints : int[]
+ saveSettings(path : String) : void
+ loadSettings(path : String) : void

<<interface>> Marble
+ getResource(p : Player) : Resource

WhiteMarble

YellowMarble

BlueMarble

PurpleMarble

Player
- nickname : String
- id : String
+ init() : void

GameBoard
+ init() : void
+ discardCard(c : Card, d : Dashboard) : void
+ getLeaderCard() : LeaderCard
+ getResourcesFromMarket(move : int, p : Player) : List<Resource>

Market
- GRID_ROW_LENGTH : int
- GRID_COL_LENGTH : int
- marblesGrid : Marble[]
+ reset() : void
+ getResources(move : int, p : Player) : List<Resource>

Warehouse
- deposit : int
- locker : int
+ reset() : void
+ clearDeposit() : void
+ clearLocker() : void
+ storeInDeposit(resToAdd : List<Resource>) : void
+ storeInLocker(resToAdd : List<Resource>) : void
+ removeFromWarehouse(cost : int) : void
+ checkAdd(resToAdd : List<Resource>) : boolean
+ checkRemoveLegality(cost : int) : void

Dashboard
+ moveFaithMarker(pos : int) : void
+ computePlayerPoints() : int
+ insertLeaderCard(c : LeaderCard, position : int) : void
+ insertDevelopmentCard(c : DevelopmentCard, position : int) : void
+ storeResourceInLocker(r : Resource, quantity : int) : void

Hand
+ discardCard(c : int) : void
+ playLeaderCard(c : int) : void

Deck
+ init(cards : List<Card>) : void
+ shuffle() : void
+ add(c : Card) : void
+ getCard() : Card

DevelopmentCardGrid
- GRID_ROW_LENGTH : int
- GRID_COL_LENGTH : int
- DEVELOPMENT_CARD_NUM : int
+ init() : void
+ buy(row : int, column : int) : DevelopmentCard
+ isBuyable(row : int, column : int, playerResources : Map<Resource,Integer>) : boolean

FaithTrack
- position : int
- vaticanReports : Map<Integer,VaticanReport>
- faithTrackVictoryPoints : int[]
+ reset() : void
+ moveFaithMarker(pos : int) : boolean
+ checkVaticanVictoryPoints(vaticanReportEnd : int) : void

VaticanReport
- start : int
- end : int
- victoryPoints : int
- missed : boolean
- achieved : boolean
+ reset() : void
+ missed() : void
+ achieved() : void
+ isMissed() : boolean
+ isAchieved() : boolean

<<interface>> Card
+ play(dashboard : Dashboard, position : int) : void
+ activate(p : Player) : void

LeaderCard
- id : int
- victoryPoints : int
- resourceCost : Map<Resource,Integer>
- bannerCost : Map<Banner,Integer>
+ discard(d : Dashboard) : void
+ isPlayable(playerResources : Map<Resource,Integer>, playerBanners : Map<Banner,Integer>) : boolean

<<interface>> SpecialAbility
+ activate(p : Player) : void

DevelopmentCardDiscount

WhiteMarbleResource

WarehouseExtraSpace

ProductionPower
- resourceRequired : Map<Resource,Integer>
- resourceProduced : Map<Resource,Integer>
- numberFaithPoints : int
- selectableResource : boolean
+ giveOneSelectableResource(p : Player, r : Resource) : void
+ isSelectableResource() : boolean

DevelopmentCard
- id : int
- victoryPoints : int
- resourceCost : Map<Resource,Integer>
+ isActivatable(playerResources : Map<Resource,Integer>) : boolean

Banner
- color : BannerEnum
- level : int
+ equalsColor(b : Banner) : boolean
+ equalsLevel(b : Banner) : boolean
+ equals(b : Banner) : boolean