Card
- cost : Map<Resources,Integer>
- age : int
- name : String
- minPlayers : int
+ Card (name : String, minPlayers, age : int, effect : Effect, cost : Multiset<Resources>, cardType: CardType)
+ ...
- displayCardCost() : void
- displayCardDetails() : String
- isChainedTo(Card card): boolean

EffectType
+ COMMERCIAL_BENEFITS
+ EARN_COIN_AND_VICTORY_POINT_FROM_CARDS
+ EARN_COIN_AND_VICTORY_POINT_FROM_STAGES
+ EARN_COINS
+ EARN_COIN_FROM_TYPE_OF_CARD
+ GUILD_BUILDERS
+ GUILD_DECORATORS
+ GUILD_EARN_VICTORY_POINTS
+ GUILD_SCIENTISTS
+ GUILD_SHIPOWNERS
+ MILITARY_MIGHT
+ PRODUCT_RESOURCES
+ SCIENTIFIC_SYMBOLS
+ VICTORY_POINTS

Effect
+ applyEffect(Player player) : void
+ toString() : String

CardCivilianStructures
- chainesCardNames : List<String>
+ CardCivilianStructures (...)
+ getEffect() : EffectVictoryPoints

CardScientificStructures
- chainesCardNames : List<String>
+ CardScientificStructures (...)
+ getEffect() : EffectScientificSymbol

CardGuilds
+ CardGuilds (...)
+ getEffect() : EffectVictoryPoints

CardMilitaryStructures
- chainesCardNames : List<String>
+ CardMilitaryStructures (...)
+ getEffect() : EffectMilitaryMight

CardCommercialStructures
- chainesCardNames : List<String>
+ CardCommercialStructures (...)

CardManufacturedGoods
+ CardManufacturedGoods (...)
+ getEffect() : EffectProductResources

CardRawMaterials
+ CardRawMaterials (...)
+ getEffect() : EffectProductResources

CardType
+ CIVILIAN_STRUCTURES : "Civilian Structures"
+ COMMERCIAL_STRUCTURES :"Commercial Structures"
+ GUILDS : "Guilds"
+ MANUFACTURED_GOODS : "Manufactured Goods"
+ MILITARY_STRUCTURE : "Military Structures"
+ RAW_MATERIALS : "Raw Materials"
+ SCIENTIFIC_STRUCTURES : "Scientific Structures"

EffectCommercialBenefits
- onWhichNeighbour : int
- onWhichRessources : List<Ressources>
+ EffectCommercialBenefits (...)
+ applyEffect(Player) : void
+ toString() : String

EffectEarnCoins
- coins : int
+ EffectEarnCoins (...)
+ applyEffect(Player) : void
+ toString() : String

EffectMilitaryMight
- militaryMightIncrease : int
+ EffectMilitaryMight (...)
+ applyEffect(Player) : void
+ toString() : String

EffectProductResources
- producedResources : Map<Resources, Integer>
+ EffectProductResources (...)
+ applyEffect(Player) : void
+ toString() : String

EffectVictoryPoints
- victoryPointScored : int
+ EffectVictoryPoints (...)
+ applyEffect(Player) : void
+ toString() : String

EffectEarnCoinsFromTypeOfCard
- onWhichTypeOfCard : Class <?>
- amountOfGainsForEachCard : int
+ EffectEarnCoinsFromTypeOfCard(...)
+ applyEffect(Player) : void
+ toString() : String

EffectEarnCoinAndVictoryPointFromStages
- coinsForEach : int
- victoryPointsForEach : int
+ EffectEarnCoinAndVictoryPointFromStages (...)
+ applyEffect(Player) : void
+ toString() : String

EffectEarnCoinAndVictoryPointFromCards
- onWhichTypeOfCard : Class <?>
- coinsForEach : int
- victoryPointsForEach : int
+ EffectEarnCoinAndVictoryPointFromCards (...)
+ applyEffect(Player) : void
+ toString() : String

EffectGuildBuilders
+ EffectGuildBuilders (...)
+ applyEffect(Player) : void
+ toString() : String

EffectGuildDecorators
+ EffectGuildDecorators (...)
+ applyEffect(Player) : void
+ toString() : String

EffectGuildShipowners
+ EffectGuildShipowners (...)
+ applyEffect(Player) : void
+ toString() : String

EffectGuildEarnVictoryPoints
- onWhichTypeOfCard : Class <?>
- amountOfVictoryPointsForEachCard : int
+ EffectGuildEarnVictoryPoints(...)
+ applyEffect(Player) : void
+ toString() : String

EffectScientificSymbol
+ EffectScientificSymbol (...)
+ applyEffect(Player) : void
+ toString() : String
+ getScientificSymbol() : ScientificSymbol

EffectGuildShipowners
+ EffectGuildShipowners(...)
+ applyEffect(Player) : void
+ toString() : String

WonderStage
- costs : Multiset <Resources>
+ WonderStage (...)
+ getCost () : Multiset <Resources>
+ getEffect() : Effect
+ getInstructions : String

GameManager
- numberOfPlayers : int
- currentAge : int
- currentTurn : int
+ GameManager(numberOfPlayers : Integer)
+ setupGame(playersNameList : List <String>) : void
+ startGame() : void
+ endPlayerTurn() : EndTurnEvent
+ endGame() : List<Player>
- distributeCardsToPlayers(currAgeDeck : List <Card>) : void
+ displayRanking(finalRanking : List<Player>) : String
- passCardsToNextPlayer() : void

Player
- name : String
- victoryPoints : int
- treasure : int
- militaryMight : int
- commercialBenefits : Map<Integer, Resources>
+ Player (name : String, wonder : Wonder)
+ canBuild(forCard : boolean) : boolean
+ canBuildAlone(forCard : boolean): boolean
+ calculateBuyNeedFromWhichNeighbour(forCard : boolean) : Optional <BuyNeed>
+ buildCard() : void
+ applyCardEffect() : void
+ isChainedToBuiltCard() : void
+ discardCard() : void
+ buildStage() : void
+ applyStageEffect() : void
+ increaseTreasure(treasureGains : int): void
+ increaseVictoryPoints(victoryPointsGains : int): void
- buyResourceFromNeighbour(resource : Resources, requieredNeighbourResource : int) : void
- processCost(forCard: boolean) : void
- requieredBuyResources(card : Card) : int
+ displayResources() : String
+ addCommercialBenefits(onWhichNeighbour : int, onWhichResources : List<Resources>) : void
+ addVictoryPointsAccordingToScientificCards(): void
+ addVictoryPointsAccordingToEffectAtEndOfGame(): void
- processBattle(age : int) : void
- processBattle(age : int, leftNeighbour : Player) : void
+ displayBattle(neighbour : Player) : void
- displayWinner(age : int) : void
- displayLoser() : void

Wonder
+ name : String
+ currentStage : int
+ Wonder (name : String, producedResources : Resources, stages : List<Stage>, isDay : boolean)
+ isComplete() : boolean
+ getWonderState() : boolean
+ getName() : String
+ getProducedResources() : Resources
+ getStages() : List<Stage>
+ getCurrentStage() : int
+ nextStepInstructions() : String
+ incrementCurrentStage() : void

Stage
+ costs : Map<Resources,Integer>
+ Stage (costs : Map <Resources,Integer>, effect : Effect)
+ getCosts() : Map<Resources,Integer>
+ getEffect() : Effect
+ getInstructions() : String

MainFx
+ main(String[] args): void

ScientificSymbol
+ TABLET
+ COMPASS
+ GEAR_WHEEL
+ of(scientificSymbols : ScientificSymbol []): Multiset<ScientificSymbol>

Resources
+ STONE
+ CLAY
+ WOOD
+ ORE
+ GLASS
+ TEXTILE
+ PAPYRUS
+ Resources (...)
+ of(resources: Resources[]): Multiset<Resources>

BuyNeed
+ coins : int
+ BuyNeed (player : Player, coins : int)

ObjectsInitializer
+ generateCards() : List <Card>
- createFirstAgeDeck() : List <Card>
- createSecondAgeDeck() : List <Card>
- createThirdAgeDeck() : List <Card>
+ generateWonders() : Set <Wonder>

Relationship labels: Manage, Owns, Is, Has in its deck, Choose, Choose as a main card, Has in its built cards, Possess, Neighbours, Overseers, Oversees, Concern, Produces, Has as yield, 1...*, 0...7, 3...7