UML Class Diagram

Package: Card

Package: Factory

EnemyFactory
+CreateCard(typename: string, receipe: JSON, nb: number): Array<Card>

DwarfFactory
+CreateCard(typename: string, receipe: JSON, nb: number): Array<Card>

CardFactory
+CreateCard(typename: string, receipe: JSON, nb: number): Array<Card>

BonusFactory
+CreateCard(typename: string, receipe: JSON, nb: number): Array<Card>

TrophyFactory
+CreateCard(typename: string, receipe: JSON, nb: number): Array<Card>

ReceipeFunctions
+PATH_JSON: string
+receipes: Map<string Map<string JSON>>

+loadReceipes(map:Map<string,Map<string,JSON>>,folder:string)
+json2map(json: JSON): Map<Any Any>
+getDeck(deckName: string): JSON

InitializeDeck
+receipes: Map<string, Map<string, JSON>>
+deck: JSON

-createCard(factoryType: string, name: string, receipe: JSON, number: number): Array<Card>
-getReceipe(stackName, typename, name): JSON
+generateDeck(): Map<string Array<Card>>

Bonus
+fight_value: number
+pickaxe_value: number

+printCard()

Dwarf
+first_value: number
+second_value: number
+up_symbol: boolean
+stop_symbol: boolean

+printCard()

Card
+name: string

+printCard()

Treasure
+gold_value: number
+trophy: boolean

+printCard()

Enemy
+fight_value: number
+gold_value: number
+end_mine: boolean

+printCard()

Stack
+name: string
+type: StackType
+collection: Array<Card>

+addCard(card: Card)
+removeCard(card: Card)
+addCollection(collection: Array<Card>)
+getCardByName(name: string)
+moveCardToStack(card: Card, stack: Stack)

«enumeration» StackType
Mine
PlayerHand
PlayerMine
RecruitCenter
unUsedCards

Launcher
+generateDeck()
+playGame()

Module

Question
+async promtp(): number

Action

Blaster
+blasterAction(blaster: Card, noMine: number, gameboard: GameBoard, player: number)

Scout
+scoutAction(scout: Card, noMine: number, gameboard: GameBoard, player: number)

Picker
+pickerAction(card: Card, noMine: number, gameboard: GameBoard, player: number)
+cardMinedAction(card: Card, noMine: number, gameboard: GameBoard, player: number)

Game

Player
+playerHand: Stack
+mines: Array<Stack>
+trophy: Array<Card>
+treasure: Stack

+moveCardToMine(noCard: number, noMine: number)

Game
+gameboard: GameBoard
+selectedPlayer: number

+playGame()
+doRound()
+playCard()
+cardAction(card: Card, noMine: number)

GameBoard
+id: number
+nbPlayers: number
+player: Array<Player>
+mines: Array<Stack>
+discards: Array<Stack>
+recruitCenter: Stack
+trophy: Array<Card>
+unUsedCards: Stack

+fisherYatesShuffle(cards: Array<Card>): Array<Card>
+divideBy(nb: number, cards: Array<Card>, endMine: Array<Card>): Array<Stack>
+getPlayersStack(card: Array<Card>): Array<Stack>

+players
+stacks