Deadwood UML Class Diagram: Group 1

UI

+ start(): void
+ moveCheck(): String
+ roleCheck(): String
+ roleChoose(ArrayList<ExtraRole>, ArrayList<StarringRole>, int): Role
+ moveTo(ArrayList<String>): String
+ workCheck(): String
+ finalCheck(): String
+ upgradeCheck(): String
+ upgradeWithDollarsOrCredits(): String
+ printPlayerStatus(Actor): void
+ printMovieStatus(SceneRoom): void
+ printDiceRoll(int): void

BoardController

- cardCollection: List<SceneCard>
- boardSections: List<BoardSection>
- randomGenerator: Random
- playerQ: LinkedList<Actor>
- days: int

+ playerTurn(): void
+ closeRoom(SceneRoom, Actor): Actor
+ starBonus(ArrayList<Actor>, int, Actor): int
+ resetRoles(SceneRoom): void
+ validBonus(SceneCard): boolean
+ getCurrRoom(String): Room
+ getAdjRooms(String): ArrayList<String>
+ endPlayerTurn(Actor): void
+ askRole(String, Actor): ArrayList[]
+ refreshBoard(): void
+ endGame(): void
+ setBoardSections(ArrayList<BoardSection>): void
+ setCardCollection(ArrayList<SceneCard>): void
+ setDays(): void
+ getBoardSections(): Collection<BoardSection>
+ getCardCollection(): Collection<SceneCard>
+ getDays(): int
- parseSceneCards(String): ArrayList<SceneCard>
- parseRooms(String, String): ArrayList<BoardSection>
+ pullCard(): SceneCard
+ setActorCredits(Actor, int): void
+ setActorDollars(Actor, int): void
+ setPlayerQ(LinkedList<Actor>): void
+ getMoviesLeft(): int
+ setMoviesLeft(int): void
+ setActorsCredits(Actor, int): void
+ setActorDollars(Actor, int)): void
+ getPlayerQ(): Queue<Actor>
+ setPlayerQ(LinkedList<Actor>): void

Actor

- wallet: Wallet
- rank: int
- playerID: String
- boardPosition: Room
- roleStatus: boolean
- shotBonus: int

+ act(): int
+ rehearse(): int
+ movePlayer(Room): void
+ setBoardPosition(Room): void
+ getBoardPosition(): Room
+ setRank(int): void
+ getRank(): int
+ setRoleStatus(boolean): void
+ getRoleStatus(): boolean
+ getShotBonus(): int
+ setShotBonus(int): void

Wallet

- dollars: int
- credits: int

+ getDollars(): int
+ getCredits(): int
+ incDollars(int): void
+ incCredits(int): void
+ decDollars(int): void
+ decCredits(int): void

Relation 0..n — 1

Dice

- rand: Random
- value: int

+ compareTo(Dice): int
+ getvalue(): int

Role (Abstract)

- rank: int
- title: String
- catch_phrase: String
- rehearsal_tokens: int
- occupied: boolean
- occupant: String

+ toString(): String
+ abstract payout(): int[]
+ abstract bonusPayout(): int
+ getRank(): int
+ getTitle(): String
+ isOccupied(): boolean
+ setOccupied(boolean): void
+ getOccupant(): String
+ setOccupant(String): void

BoardSection

- RoomObjects: ArrayList<Room>
- roomID: String

+ getRoomObjects(): ArrayList<Room>
+ setRoomObjects(ArrayList<Room>): void
+ getRoomID(): String
+ setRoomID(String): void

Room (Abstract)

- adjacentRooms: ArrayList<String>
- title: String

+ getAdjacentRooms(): ArrayList<String>
+ setAdjacentRooms(ArrayList<String>): void
+ getTitle(): String
+ setTitle(String): void

SceneCard

- budget: int
- starringRoles: ArrayList<StarringRole>
- id: String

+ getStarringRole(): ArrayList<StarringRole>
+ getBudget(): int

StarringRole

+ StarringRole(int, String, String)
+ payout(): int[]
+ compareTo(StarringRole): int

ExtraRole

+ ExtraRole(int, String, String)
+ payout(): int[]
+ bonusPayout(): int

Trailer

+ Trailer(String, ArrayList<String>)

SceneRoom

- shotCounter: int
- extraRole: ArrayList<ExtraRole>
- movieStatus: boolean
- sceneCard: SceneCard
- savedShotCounters: int

+ getExtraRoles(): ArrayList<ExtraRole>
+ setShotCounter(int): void
+ getSavedShotCounters(): int
+ getMovieStatus(): boolean
+ getShotCounter(): int
+ decShotCounter(): void
+ getSceneCard(): SceneCard
+ setSceneCard(SceneCard): void
+ toString(): String

CastingOffice

+ CastingOffice(String, ArrayList<String>)
+ getUpgradeHash(): HashMap<Integer, Integer[]>

Extends