Game

player: Player
introduction
currentStage: Stage
stages: Stage[]
items: Item[]

---

constructor(playerName: string)

initiateAllStages()

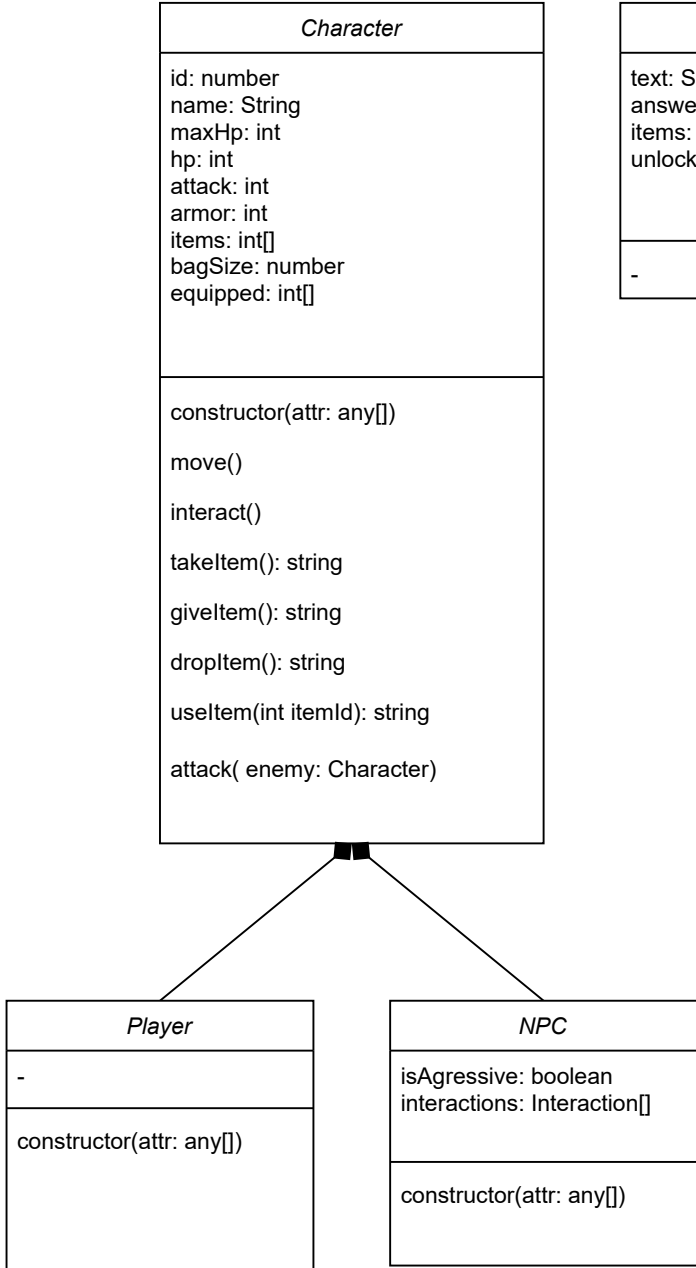changeCurrentStage(stage: Stage)

movePlayer(destination: number)

InOutManager

-

---

readline(question: string): string

readLine(message: string)

Instruction

information: string
command: string
options: string[]

---

toString(): string

do(option: number, game: Game)

GameManager

game: Game
inout: InOutManager
win: boolean
instructionList: Instruction[]

---

initializeGame: Game

runGame()

step()

doAction()

createInfoInstruction()

Stage

id: int
name: string
information: String
nextStageIds: number[]
nextStages: Stage[]
characters: Character[]
items: Item[]
locked: boolean

---

constructor(attr: any[])

getStageInformation(): string

getNextStageOptions(): string[]

getItemOptions(): string[]

removeCharacter(c: Character): Character

addCharacter(c: Character)

initiateNextStages(g: Game)

giveItem(index: int)

Item

id: int
name: String
type: String
increaseHp: int
bagSize: int
hp: int
attack: int
armor: int
opensStage: int

---

constructor(attr: any[])

toString(): string

Character

id: number
name: String
maxHp: int
hp: int
attack: int
armor: int
items: int[]
bagSize: number
equipped: int[]

---

constructor(attr: any[])

move()

interact()

takeItem(): string

giveItem(): string

dropItem(): string

useItem(int itemId): string

attack(enemy: Character)

Interaction

text: String
answer: String
items: int[]
unlocks: int[]

-

Player

-

---

constructor(attr: any[])

NPC

isAgressive: boolean
interactions: Interaction[]

---

constructor(attr: any[])