Milestone 4: TheView
Andy Cui | April 14, 2022

Model

Use ReadOnly Interface to allow View class access Model method

<<interface>> Room
+ getName(): String
+ setName(name: String): void
+ getRect(): Rectangle
+ setRect(rect: Rectangle): void
+ getIndex(): int
+ setIndex(index: int): void
+ getItems(): ArrayList<Item>
+ getPlayers(): ArrayList<Player>
+ isHasPet(): boolean
+ displayCurt(): void
+ display(): void

<<interface>> WorldModel
+ getName(): String
+ setName(name: String): void
+ getRows(): int
+ setRows(rows: int): void
+ getColumns(): int
+ setColumns(columns: int): void
+ getTc(): Target
+ getRooms(): ArrayList<Room>
+ getNeighbors(pos: int): ArrayList<Room>

<<interface>> ReadOnlyWorldModel

<<interface>> Target
+ getName(): String
+ setName(name: String): void
+ getHealth(): int
+ setHealth(health: int): void
+ getPosition(): int
+ setPosition(position: int): void
+ getTc(): Target
+ getPet(): Pet
+ setPet(pet: Pet): void
+ move(): void

<<interface>> Item
+ getName(): String
+ setName(name: String): void
+ getAmountOfDamage(): int
+ setAmountOfDamage(amountOfDamage: int): void

<<interface>> Player
+ getName(): String
+ setName(name: String): void
+ getPosition(): int
+ setPosition(position: int): void
+ getItems(): ArrayList<Item>
+ isRobot(): boolean
+ setRobot(isRobot: boolean): void
+ display(): void
+ attack(): void
+ turn(rand: int): void
+ turnRobot(): void

<<interface>> Pet
+ getName(): String
+ setName(name: String): void
+ getPosition(): int
+ setPosition(position: int): void
+ move(): void

RoomImpl
- name: String
- index: int
- rect: Rectangle
- items: ArrayList<Item>
- players: ArrayList<Player>

+ RoomImpl(name: String, index: int, rect: Rectangle)
+ getName(): String
+ setName(name: String): void
+ getRect(): Rectangle
+ setRect(rect: Rectangle): void
+ getIndex(): int
+ setIndex(index: int): void
+ getItems(): ArrayList<Item>
+ getPlayers(): ArrayList<Player>
+ isHasPet(): boolean
+ displayCurt(): void
+ display(): void

WorldModelImpl
- name: String
- rows: int
- columns: int
- tc: Target
- rooms: ArrayList<Room>

+ WorldModelImpl()
+ WorldModelImpl(name: String, rows: int, cols: int)
+ isNeighbor(room1: Room, room2: Room)
+ getName(): String
+ setName(name: String): void
+ getRows(): int
+ setRows(rows: int): void
+ getColumns(): int
+ setColumns(columns: int): void
+ getTc(): Target
+ getRooms(): ArrayList<Room>
+ getNeighbors(pos: int): ArrayList<Room>

TargetImpl
- name: String
- health: int
- position: int
- pet: Pet

+ TargetImpl()
+ TargetImpl(name: String, health: int, position: int, getName: String)
+ getName(): String
+ setName(name: String): void
+ getHealth(): int
+ setHealth(health: int): void
+ getPosition(): int
+ setPosition(position: int): void
+ getPet(): Pet
+ setPet(pet: Pet): void
+ move(): void

ItemImpl
- name: String
- amountOfdamage: int

+ ItemImpl(name: String, amountOfDamage: int)
+ getName(): String
+ setName(name: String): void
+ getAmountOfDamage(): int
+ setAmountOfDamage(amountOfDamage: int): void

PlayerImpl
- name: String
- position: int
- items: ArrayList<Item>
- limitCanCarry: int
- isRobot: boolean

+ PlayerImpl(name: String, position: int, limitCanCarry: int, isRobot: boolean)
- pickupItem(item: Item, pos: int)
- move(): void
- lookAround(): void
- movePet(): void
- selectMostDanger(): Item
- canBeSeen(): boolean
+ getName(): String
+ setName(name: String): void
+ getPosition(): int
+ setPosition(position: int): void
+ getItems(): ArrayList<Item>
+ isRobot(): boolean
+ setRobot(isRobot: boolean): void
+ display(): void
+ attack(): void
+ turn(rand: int): void
+ turnRobot(): void

PetImpl
- name: String
- position: int

+ PetImpl()
+ PetImpl(name: String, position: int)
+ getName(): String
+ setName(name: String): void
+ getPosition(): int
+ setPosition(position: int): void
+ move(): void

Controller

<<interface>> WorldModel
+ getName(): String
+ setName(name: String): void
+ getRows(): int
+ setRows(rows: int): void
+ getColumns(): int
+ setColumns(columns: int): void
+ getTc(): Target
+ getRooms(): ArrayList<Room>
+ getNeighbors(pos: int): ArrayList<Room>

Use Word interface as fasad to access the game model

ControllerImpl
- scan: Scanner
- out: Appendable
- players: ArrayList<Player>
- maxTurns: int

+ ControllerImpl(in: Scanner, out: Appendable)
- save(world: WorldModel): void
- parseFile(path: String, world: WorldModel): void
- addPlayer(player: Player): void
- loadData(world: WorldModel): void
- initialize(world: WorldModel): void
+ startGame(world: WorldModel)

<<interface>> Controller
+ processCommandString(gameCommand: String): String
+ startGame(world)

WorldGuiController
- model: WorldModel
- view: WorldView

+ WorldGuiController(model: WorldModel, view: WorldView)
+ play()
+ handlePlayer(player: Player): void
+ handleRoom(room: Room): void
+handleItem(item: Item): void
+ handleTarget(target: Target): void
+ handlePet(pet: Pet): void

<<interface>> Features
+ handlePlayer(player: Player): void
+ handleRoom(room: Room): void
+ handleItem(item: Item): void
+ handleTarget(target: Target): void
+ handlePet(pet: Pet): void

<<interface>> WorldView

<<interface>> WorldModel

<<interface>> GameCommand
+ startGame(world: World)

SetMaxRound
- maxRound: int
- out: Appendable

+ SetMaxRound(maxRound: int, out: Appendable): void
+ startGame(world: World)

SetMaxPlayers
- maxPlayers: int
- out: Appendable

+ SetMaxPlayers(maxPlayers: int, out: Appendable): void
+ startGame(world: World)

AddPlayer
- name: int

DisplayPlayerStatus
- playerIndex: int
- out: Appendable

+ DisplayRoom(roomIndex: int, out: Appendable): void
+ startGame(world: World)

Move
- playerName: String
- roomIndex: int
- out: Appendable

+ AddPlayer(playerName: String, roomIndex: int, out: Appendable): void
+ startGame(world: World)

PickUp
- characterName: String
- roomIndex: int
- out: Appendable

+ AddCharacter(playerName: String, roomIndex: int, out: Appendable): void
+ startGame(world: World)

LookAround

MovePetToOtherRoom
- roomIndex: int
- out: Appendable

+ MovePen(roomIndex: int, out: Appendable): void
+ startGame(world: World)

Attack
- attack: int
- out: Appendable

+ Attack(attack: int, out: Appendable): void
+ startGame(world: World)

View

<<interface>> WorldView
+ makeVisible()
+ addClickListener(listener: Controller)
+ sendMessage(message: String)
+ refresh()

<<interface>> Features
+ handlePlayer(player: Player): void
+ handleRoom(room: Room): void
+ handleItem(item: Item): void
+ handleTarget(target: Target): void
+ handlePet(pet: Pet): void

WorldViewImpl
m: WorldModel
v: WorldView

+ WorldViewImpl(title: String, model: ReadOnlyWorldModel)
+ makeVisible()
+ addClickListener(listener: Controller)
+ sendMessage(message: String)
+ refresh()

WorldGuis
+ CELL_DIM: int = 800
+ OFFSET: int = 800
+ FONT_SIZE: int = 400
- serialVersionUID: int

- model: ReadOnlyWorldModel

+ WorldGuis(ReadOnlyWorldModel)
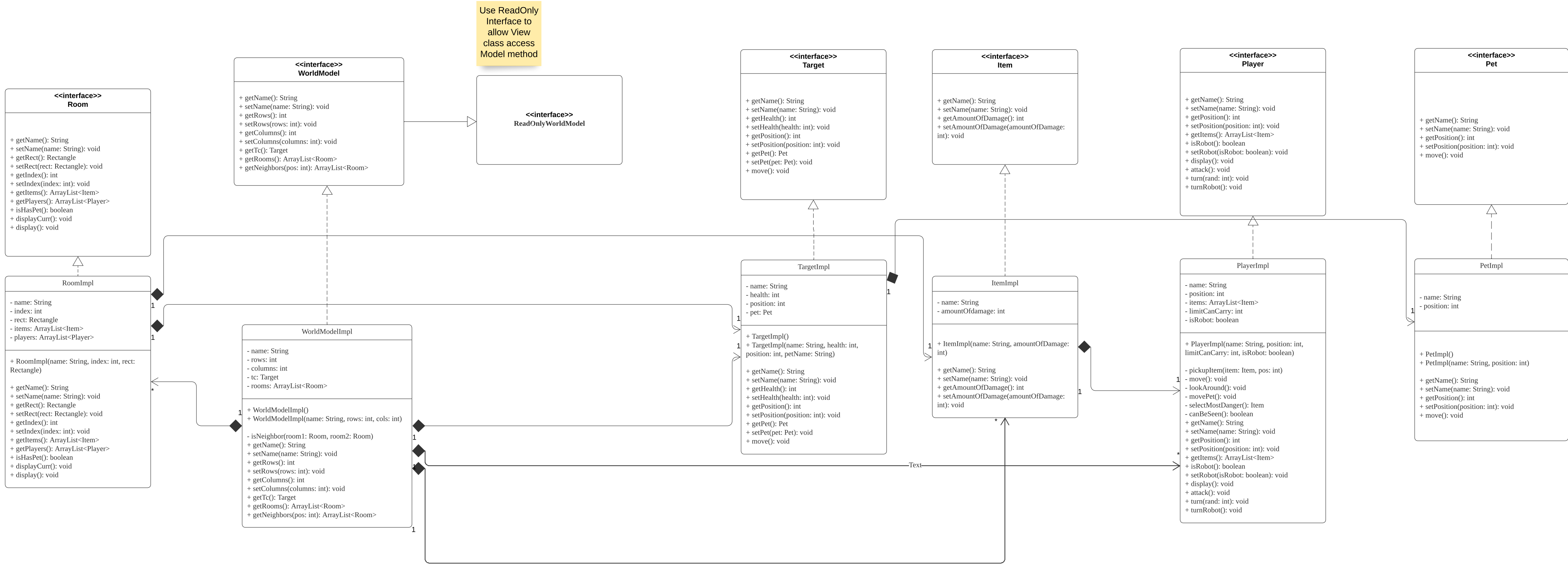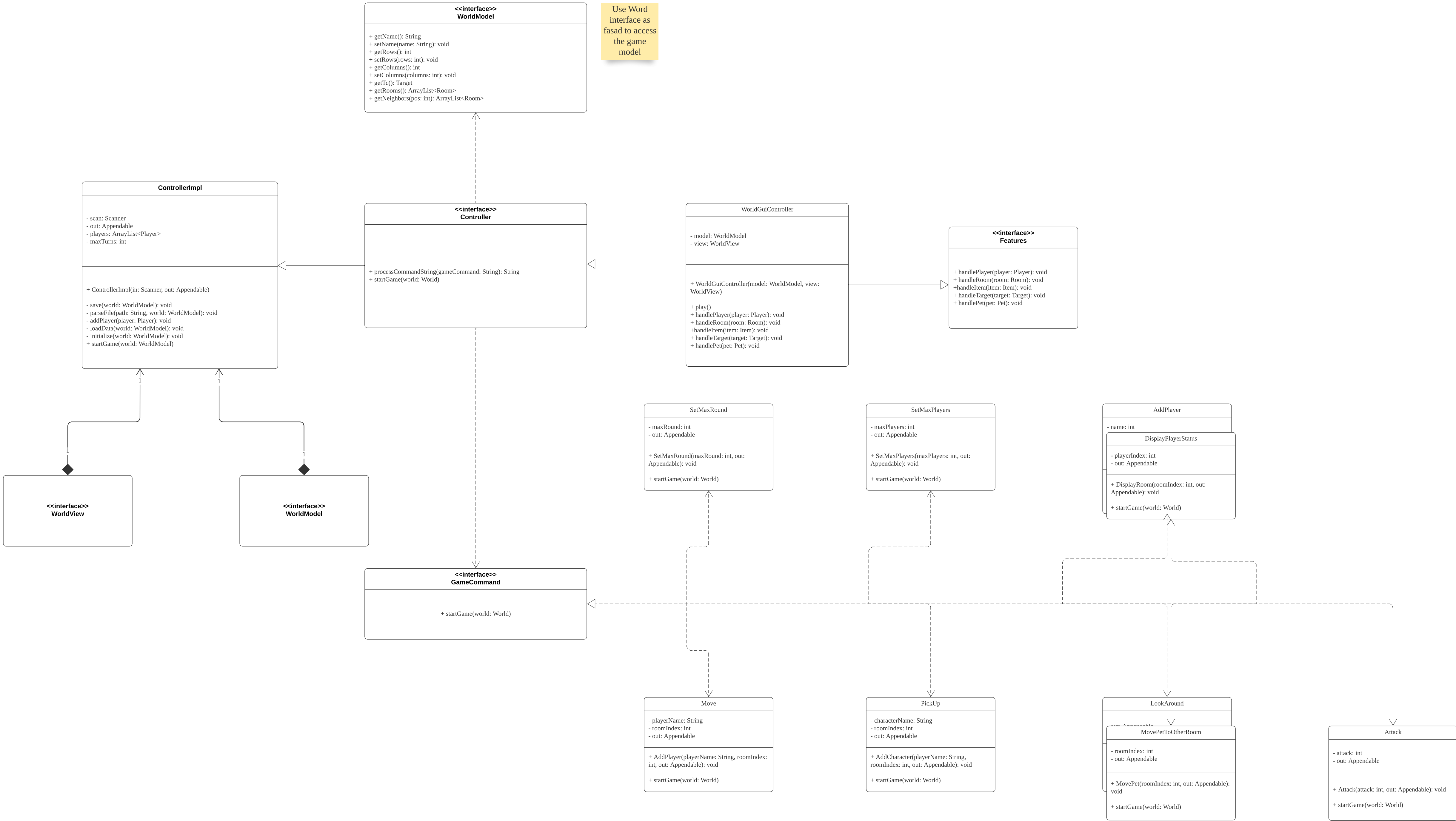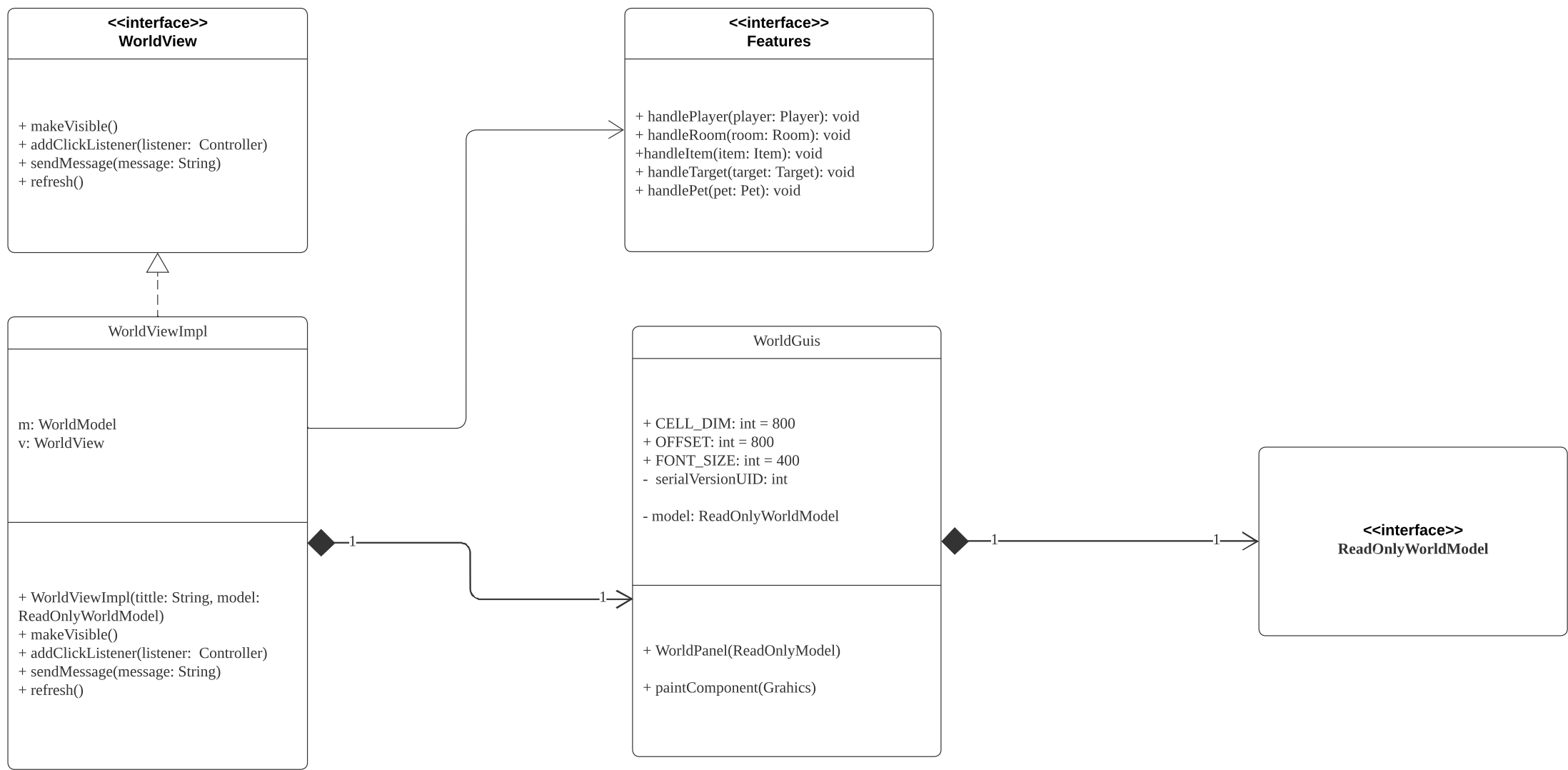+ paintComponent(Graphics)

<<interface>> ReadOnlyWorldModel

Testing design for Milestone4: The View

Created by Andy(Xiang-yu) Cui

NUID: 001514414

Reference documents(UML): Milestone4_ theview

Model Test:

`worldModel.java`

Testing `instance`	Input	Expected Value
General test	read("Doctor Lucky's Mansion", 36, 30)	program works
test wrong path	read("abc")	IllegalArgumentException

Testing `getName()` & `setName()`	Input	Expected Value
Set name	read("Doctor Lucky's Mansion")	Successful
Get name	read("Doctor Lucky's Mansion")	Doctor Lucky's Mansion

Testing `getRows()` & `setRows()`	Input	Expect Value
Set total rows	read(36)	Successful
Get total rows	read(36)	36

Testing `getCols()` & `setCols()`	Input	Expect Value
Set total cols	read(30)	Successful
Get total cols	read(30)	30

Get Target	Input	Expect Value
Testing `getTc()`	Target("Doctor Lucky", 50, 0, "Fortune the Cat")	Successful

Testing `getNeighbours()`	Input	Expect Value
General input	read(0)	Billiard Room, 1; Dining Hall, 3; Drawing Room, 4

Testing `getRooms`	Input		Expect Value
Get Rooms	rooms("Armory", 0, rect, world)		22 19 23 26 Armory

`Room.java`

Testing `getRooms()`	Input	Expect Value
General total rooms	rmtotal(1)	1
Total rooms with negative integer	rmtotal(-1)	IllegalArgumentException
Total rooms with invalid string	rmtotal("abc")	IllegalArgumentException
Total rooms with zero room	rmtotal(0)	IllegalArgumentException
Total rooms with large integer	rmtotal(9999)	9999

Testing `getName()`	Input	Expect Value
General room's name	rooms("Armory", 22, 19, 23, 26, 0, "Revolver")	Armory
Room's name with integer	rooms("Armory1", 22, 19, 23, 26, 0, "Revolver")	Armory1
Room's name with invalid string	rooms("@#", 22, 19, 23, 26, 0, "Revolver")	IllegalArgumentException
Room's name with pure integer	rooms("123", 22, 19, 23, 26, 0, "Revolver")	123

Testing `getIndex()` & `setIndex`	Input	Expect Value
Get room's index	rooms("0", Armory, rect)	0
Set room's index	rooms("0", Armory, rect)	Successful

Testing `getItems()`	Input	Expect Value
General input	rooms("0", Armory, rect)	Revolver

Testing `getPlayers`	Input	Expect Value
General set player1(Tom) in Room 0	rooms("0", Armory, rect)	player1: Tom

`Item.java`

Testing `getItemPos()`	Input	Expect Value
General item position	itt("Crepe Pan", 8, 3)	16 3 21 10
items with non-exist name	itt("abc", 8, 3)	IllegalArgumentException

Testing `getItemAttack()`	Input	Expect Value
General item damage	itt("Crepe Pan", 8, 3)	3
Total items with negative integer	iitt("Crepe Pan", 8, -3)	IllegalArgumentException
Total items with invalid string	itt("Crepe Pan", 8, "abc")	IllegalArgumentException
Total items with negative room index	itt("Crepe Pan", -1, 3)	IllegalArgumentException

Testing `getItemName()`	Input	Expect Value
General item name	itt("Crepe Pan", 8, 3)	Crepe Pan
Item's name with integer	itt("Crepe Pan1", 8, 3)	Crepe Pan1
Item's name with invalid string	itt("Crepe Pan", 8, 3)	IllegalArgumentException
Item's name with pure integer	itt("123", 8, 3)	123

`TargetCharacter.java`

Testing instance and `getCharacterName()`	Input	Expected Value
All general input	target("Doctor Lucky", 50, 22, 19, 23, 26)	Doctor Lucky
Invalid type in our case we test	target("", 50, 22, 19, 23,	IllegalArgumentException

Testing `getCharacterPos()`	Input	Expected Value
All general input	target("Doctor Lucky", 50, 22, 19, 23, 26)	22, 19, 23, 26
Invalid type	target("Doctor Lucky", -50, -22, -19, -23, -26)	IllegalArgumentException

Testing `getCharacterHp()`	Input	Expected Value
All general input	target("Doctor Lucky", 50, 22, 19, 23, 26)	50
Invalid type input	target("Doctor Lucky", -50, -22, -19, -23, -26)	IllegalArgumentException

Testing `getTargetAttack()`	Input	Expected Value
All general input	player1("Tom", "Crepe Pan", 22, 19, 23, 26)	22
Invalid type negative int	player1("Tom", "Crepe Pan", -22, 19, 23, 26)	IllegalArgumentException

`Pet.java`

Testing `getPetPos`	Input	Expected Value
All general input	petSimple(1)	1
Invalid type negative int	petSimple(-1)	IllegalArgumentException

Testing `getMaster`	Input	Expected Value
All general input	petSimple(1)	Doctor Lucky

Testing `petLookAround`	Input	Expected Value
All general input	rooms("Armory", 22, 19, 23, 26, 0, "Revolver") Tom.inspect;	upperLeftCorner: Dinning Hall; lowerLeftCorner: Billiard Room, Upper: Drawing Room
if No wall	rooms("AC Room", 22, 19, 23, 26, 0, "Revolver")	No wall

Testing	Tom.inspect;	

Testing `currentRoomInfo()`	Input	Expect Value
Current Room	room[0].getCurrentRoom	Armory, 0

`Player.java`

Testing instance and `getPlayerName()`	Input	Expected Value
All general input	player("Tom", "Crepe Pan", 22, 19, 23, 26)	Tom
Invalid type empty	player("", "Crepe Pan", 22, 19, 23, 26)	IllegalArgumentException

Testing `getNumberOfPlayer()`	Input	Expected Value
All general input	playerCount(2)	2
Invalid type negative	platerCount(-2)	IllegalArgumentException

Testing instance and `getPlayerPos()`	Input	Expected Value
All general input	player("Tom", "Crepe Pan", 22, 19, 23, 26)	22, 19, 23, 26
Invalid type empty lowerCorner	player("Tom", "Crepe Pan", 22, 19)	IllegalArgumentException

Testing instance and `getHoldItem()`	Input	Expected Value
All general input	player("Tom", "Crepe Pan", 22, 19, 23, 26)	Crepe Pan
Invalid type empty	player("Tom", "", 22, 19, 23, 26)	IllegalArgumentException

Testing `pickUp()`	Input	Expected Value
All general input	rooms("Armory", 22, 19, 23, 26, 0, "Revolver") Tom.pickUp;	Revolver
Invalid type empty	rooms("Armory", 22, 19, 23, 26, 0, "") Tom.pickUp;	null

Testing `inspect()`	Input	Expected Value
All general input	rooms("Armory", 22, 19, 23, 26, 0, "Revolver") Tom.inspect;	upperLeftCorner: Dinning Hall; lowerLeftCorner: Billiard Room, Upper: Drawing Room
if No wall	rooms("AC Room", 22, 19, 23, 26, 0, "Revolver") Tom.inspect;	No wall

Testing `attack()`	Input	Expected Value
All general input	rooms("Armory", 22, 19, 23, 26, 0, "Revolver") Tom.attack; show.getCharacterHp;	47

Controller Test:

Testing `DisplayRoom()`	Input	Expected Value
All general input	StringReader(display)	roomDisplay;
txt fil is not exist	StringReader(display)	IllegalArgumentException

Testing `AddPlayer()`	Input	Expected Value
All general input	StringReader("Tom", 1)	player Tom already add to game
roomIndex is not exist	StringReader("Jarry", -1)	IllegalArgumentException

Testing `AddCharacter()`	Input	Expected Value
All general input	StringReader(1)	npc already add to game
roomIndex is not exist	StringReader(-1)	IllegalArgumentException

Testing `MoverPlayer()`	Input	Expected Value
All general input	StringReader(1)	current room index is 1
roomIndex is not exist	StringReader(-1)	IllegalArgumentException

Testing `PickItem()`	Input	Expected Value
All general input	StringReader(1)	player: XXX pick the item "Crepe Pan"
item is not exist	StringReader(1)	current room is nothing

Testing `DisplayPlayerStatus()`	Input	Expected Value
All general input	StringReader(1)	player: XXX
player is not exist	StringReader(99)	player is not exist

Testing `MovePet()`	Input	Expected Value
All general input	StringReader(1)	pet moved to index 1
pet move is invalid	StringReader(99)	pet can not move there

Testing `Attempt()`	Input	Expected Value
All general input	StringReader(1)	Attempt attack is 2
Attempt is empty	StringReader(22)	Attempt is nothing

View Test:

So, the GUI test by function and check the GUI is create successful or not.

Testing `WorldViewImpl()`	Input	Expected Value
All general input	startGui()	checked successful
Gui created fail	startGui()	IllegalException

Testing `makeVisible()`	Input	Expected Value
All general input	startGui()	checked successful
Gui created fail	startGui()	IllegalException

Testing `addClickListener()`	Input	Expected Value
All general input	startGui()	checked successful
Gui created fail	startGui()	IllegalException

Testing `sendMessage()`	Input	Expected Value
All general input	testMessage("Kill Target Game")	checked successful
Gui created fail	testMessage("")	""