GameMode (enum)

f	GUI	
f	TEXT	
m	GameMode()	
m	valueOf(String)	GameMode
m	values()	GameMode[]

ModeSelectMenu (class)

f	mainFrame	JFrame
f	gameModeGUIRadioButton	JRadioButton
f	gameModeTextRadioButton	JRadioButton
m	ModeSelectMenu()	

Model (interface)

m	changeTurn()	void
m	isPitNearBy(Cell)	boolean
m	isWumpusNearBy(Cell)	boolean
m	move(Direction)	String
m	shoot(Direction, int)	String
p	currentPlayer	PlayerInterface
p	currentPlayerCell	Cell
p	exploredCells	List<Cell>
p	gameOver	boolean
p	maze	MazeInterface
p	mazeCols	int
p	mazeRows	int
p	numOfArrows	int
p	player2Cell	Cell
p	playerCell	Cell
p	playerTurn	int
p	twoPlayerMode	boolean
p	win	boolean
p	wumpusCell	Cell

HTWModel (class)

m	HTWModel()	
m	changeTurn()	void
m	isPitNearBy(Cell)	boolean
m	isWumpusNearBy(Cell)	boolean
m	move(Direction)	String
m	shoot(Direction, int)	String
p	currentPlayer	PlayerInterface
p	currentPlayerCell	Cell
p	exploredCells	List<Cell>
p	gameOver	boolean
p	maze	MazeInterface
p	mazeCols	int
p	mazeRows	int
p	numOfArrows	int
p	player2Cell	Cell
p	playerCell	Cell
p	playerTurn	int
p	twoPlayerMode	boolean
p	win	boolean
p	wumpusCell	Cell

View (interface)

m	createMaze(String[])	MazeInterface
m	displayStatusMessage(String)	void
m	restoreMenu()	void
m	showGameArea()	void
m	showMessageDialog(String)	void
m	showPlayerTurn(int)	void
m	startGame()	void
m	startShoot()	void
m	stopShoot()	void
p	controller	Controller
p	randomSeed	int

HTWView (class)

f	mainFrame	JFrame
f	mainPanel	MainPanel
f	gamePanel	GamePanel
f	menuPanel	MenuPanel
f	controlPanel	ControlPanel
f	displayPanel	JPanel
f	gameScrollPane	JScrollPane
m	HTWView()	
m	createMaze(String[])	MazeInterface
m	displayStatusMessage(String)	void
m	restoreMenu()	void
m	showGameArea()	void
m	showMessageDialog(String)	void
m	showPlayerTurn(int)	void
m	startGame()	void
m	startShoot()	void
m	stopShoot()	void
p	controller	Controller
p	randomSeed	int

HTWTextView (class)

f	scanner	Scanner
m	HTWTextView()	
m	createMaze(String[])	MazeInterface
m	displayStatusMessage(String)	void
m	initText(String[])	MazeInterface
m	restoreMenu()	void
m	showGameArea()	void
m	showMessageDialog(String)	void
m	showPlayerTurn(int)	void
m	startGame()	void
m	startShoot()	void
m	stopShoot()	void
p	controller	Controller
p	randomSeed	int

Controller (class)

f	view	View
f	model	Model
m	Controller(Model, View)	
m	checkStatus()	void
m	clickMove(int, int)	void
m	init(GameMode)	void
m	initMaze()	void
m	isPitNearBy(Cell)	boolean
m	isWumpusNearBy(Cell)	boolean
m	main(String[])	void
m	move(Direction)	String
m	shoot(Direction, int)	String
m	startGame()	void
m	startShoot()	void
m	stopShoot()	void
m	updateNumOfArrow()	void
p	currentPlayerArrow	int
p	exploredCells	List<Cell>
p	gameOver	boolean
p	maze	MazeInterface
p	mazeCols	int
p	mazeRows	int
p	player2Cell	Cell
p	playerCell	Cell
p	shooting	boolean
p	twoPlayerMode	boolean
p	win	boolean
p	wumpusCell	Cell

MainPanel

f 🔒	view	View
m 🔒	MainPanel(View)	

Parameters

f 🔒	WIDTH	int
f 🔒	HEIGHT	int
f 🔒	CELL_WIDTH	int
f 🔒	CELL_HEIGHT	int
f 🔒	RANDOM_SEED_RANGE	int
m 🔒	Parameters()	

PlayerMode

f 🔒	SINGLE_PLAYER	
f 🔒	TWO_PLAYER	
m 🔒	PlayerMode()	
m 🔒	*valueOf(String)*	PlayerMode
m 🔒	*values()*	PlayerMode[]

GamePanel

f 🔒	view	View
f 🔒	player2Image	BufferedImage
f 🔒	pitNearByImage	BufferedImage
f 🔒	wumpusNearByImage	BufferedImage
f 🔒	pitImage	BufferedImage
f 🔒	batImage	BufferedImage
f 🔒	wumpusImage	BufferedImage
f 🔒	caveImage1up	BufferedImage
f 🔒	caveImage1down	BufferedImage
f 🔒	caveImage1left	BufferedImage
f 🔒	caveImage1right	BufferedImage
f 🔒	caveImage3NoUp	BufferedImage
f 🔒	caveImage3NoDown	BufferedImage
f 🔒	caveImage3NoLeft	BufferedImage
f 🔒	caveImage3NoRight	BufferedImage
f 🔒	tunnelImageLeftUp	BufferedImage
f 🔒	tunnelImageLeftDown	BufferedImage
f 🔒	tunnelImageRightUp	BufferedImage
f 🔒	tunnelImageRightDown	BufferedImage
f 🔒	tunnelImageStraightHor	BufferedImage
f 🔒	tunnelImageStraightVer	BufferedImage
f 🔒	caveImageMaps	HashMap<Integer, BufferedImage>
f 🔒	playerImage	BufferedImage
f 🔒	caveImage0	BufferedImage
f 🔒	caveImage4	BufferedImage
m 🔒	GamePanel(View)	
m 🔒	drawCell(Graphics, Controller, Cell)	void
m 🔒	paintComponent(Graphics)	void

MenuPanel

f 🔒	view	View
f 🔒	numOfRowTextField	TextField
f 🔒	numOfColTextField	TextField
f 🔒	numOfRemainingWallTextField	TextField
f 🔒	numOfSuperBatCellTextField	TextField
f 🔒	numOfBottomlessPitTextField	TextField
f 🔒	randomSeedTextField	TextField
f 🔒	mazeTypeButtonGroup	ButtonGroup
f 🔒	mazeTypePerfectButton	JRadioButton
f 🔒	mazeTypeRoomButton	JRadioButton
f 🔒	mazeWrappedButtonGroup	ButtonGroup
f 🔒	mazeWrappedButton	JRadioButton
f 🔒	mazeNotWrappedButton	JRadioButton
f 🔒	playerNumOfArrowTextField	TextField
f 🔒	playerNumRadioButtonGroup	ButtonGroup
f 🔒	onePlayerRadioButton	JRadioButton
f 🔒	twoPlayerRadioButton	JRadioButton
m 🔒	MenuPanel(View)	
m 🔒	createMaze()	MazeInterface
m 🔒	initGameSettingPanel()	JPanel
m 🔒	initMazeSettingPanel()	JPanel
m 🔒	initPlayerSettingPanel()	JPanel
p 🔒	randomSeed	int

ControlPanel

f 🔒	view	View
f 🔒	manualLabel	JTextArea
f 🔒	currPlayerTextArea	JTextArea
f 🔒	exitBtn	JButton
f 🔒	playGameBtn	JButton
f 🔒	newGameBtn	JButton
f 🔒	msgLabel	JTextArea
f 🔒	shootDirRadioButtonGroup	ButtonGroup
f 🔒	shootDirUpRadioButton	JRadioButton
f 🔒	shootDirDownRadioButton	JRadioButton
f 🔒	shootDirLeftRadioButton	JRadioButton
f 🔒	shootDirRightRadioButton	JRadioButton
f 🔒	shootDistTextField	TextField
f 🔒	shootButton	JButton
m 🔒	ControlPanel(View)	
m 🔒	disableShootComponents()	void
m 🔒	enableShootComponents()	void
m 🔒	showMessage(String)	void
m 🔒	showPlayerTurn(int)	void

MazeInterface (interface)

- assignBatCells() : void
- assignPitCells() : void
- changeTurn() : void
- creatMaze() : void
- getPossibleMoves(Cell) : List<Direction>
- isPitNearBy(Cell) : boolean
- isWumpusNearBy(Cell) : boolean
- makeAMove(Direction) : String
- printMaze() : String
- setPlayer(PlayerInterface, int, int, int) : void
- setPlayerPos(Cell) : String
- setWumpusCell(int, int) : void
- shootAArrow(Cell, Direction, int) : String
- allCells : List<Cell>
- allEdges : List<Edge>
- batCells : int
- cols : int
- currentPlayer : PlayerInterface
- deletedEdges : List<Edge>
- exploredCells : List<Cell>
- gameOver : boolean
- pitCells : int
- player : PlayerInterface
- player2 : PlayerInterface
- playerTurn : int
- possibleMoves : List<Direction>
- remainingWalls : int
- rows : Cell
- startCell : Cell
- twoPlayerMode : boolean
- wrapped : boolean
- wumpusCell : Cell

AbstractMaze (class)

- chanceInBAT : double
- chanceInBATPIT : double
- seed : int
- playerMode : PlayerMode
- player1 : PlayerInterface
- rand : Random
- cellMap : HashMap<Integer, HashMap<Integer, Cell>>
- cellIndex : HashMap<Cell, Integer>
- remainingEdges : List<Edge>
- maxWalls : int
- allWalls : int
- player1turn : boolean
- parents : HashMap<Cell, Cell>
- numberOfCaves : int
- numberOfTunnels : int
- AbstractMaze(int, int, boolean, int, int, int, PlayerMode)
- AbstractMaze(int, int, int, boolean, int, int, int, PlayerMode)
- assignBatCells() : void
- assignPitCells() : void
- changeTurn() : void
- checkValidness(int, int, int, boolean, int, int) : void
- creatCellsAndNeighbors() : void
- creatMaze() : void
- createAllEdges() : void
- createDeletedEdges() : void
- findParent(Cell) : Cell
- getArrowPosition(Cell, Direction, int) : String
- getCellsKind() : void
- getMinWalls(int, int, boolean) : int
- getOp(Direction) : Direction
- getPossibleMoves(Cell) : List<Direction>
- grabbedByBAT(Cell) : String
- isPitNearBy(Cell) : boolean
- isWumpusNearBy(Cell) : boolean
- makeAMove(Cell, Direction) : Cell
- makeAMove(Direction) : String
- meetPIT(Cell) : String
- moveToCave(Cell, Direction) : Cell
- printMaze() : String
- setPlayer(PlayerInterface, int, int, int) : void
- setPlayerPos(Cell) : String
- setWumpusCell(int, int) : void
- shootAArrow(Cell, Direction, int) : String
- toString() : String
- union(Cell, Cell) : void
- allCells : List<Cell>
- allEdges : List<Edge>
- batCells : int
- cols : int
- currentPlayer : PlayerInterface
- deletedEdges : List<Edge>
- exploredCells : List<Cell>
- gameOver : boolean
- pitCells : int
- player : PlayerInterface
- player2 : PlayerInterface
- playerTurn : int
- possibleMoves : List<Direction>
- remainingWalls : int
- rows : int
- startCell : Cell
- twoPlayerMode : boolean
- wrapped : boolean
- wumpusCell : Cell

PlayerInterface (interface)

- arrows : int
- currPosition : Cell
- exploredCells : List<Cell>
- posIndex : int
- status : PlayerStatus

Player (class)

- Player()
- Player(int)
- toString() : String
- arrows : int
- currPosition : Cell
- exploredCells : List<Cell>
- posIndex : int
- status : PlayerStatus

Edge (class)

- Edge(Cell, Cell)
- toString() : String
- from : Cell
- to : Cell

Cell (class)

- Cell()
- Cell(int, int)
- equals(Object) : boolean
- hashCode() : int
- toString() : String
- cellKind : CellKind
- cellType : CellType
- neighbors : List<Cell>
- x : int
- y : int

PlayerStatus (enum)

- LIVE
- DEAD
- WIN
- PlayerStatus()
- valueOf(String) : PlayerStatus
- values() : PlayerStatus[]

Direction (enum)

- NORTH
- SOUTH
- EAST
- WEST
- Direction()
- valueOf(String) : Direction
- values() : Direction[]

CellKind (enum)

- CAVE
- TUNNEL
- CellKind()
- valueOf(String) : CellKind
- values() : CellKind[]

CellType (enum)

- NORMAL
- PIT
- BAT
- PITANDBAT
- CellType()
- valueOf(String) : CellType
- values() : CellType[]

RoomMaze (class)

- RoomMaze(int, int, int, boolean, int, int, int, PlayerMode)

PerfectMaze (class)

- PerfectMaze(int, int, boolean, int, int, int, PlayerMode)

WrappingRoomMaze (class)

- WrappingRoomMaze(int, int, int, int, int, int, PlayerMode)

Test Plan

Test plan for Model
1. Test Interface MazeInterface, similar test cases for Class RoomMaze, PerfectMaze and WrappingRoomMaze
   a. MazeInterface maze = new RoomMaze(rows, cols, remainingWalls, isWrapped, batCells, pitCells);
      Test getter method: rows, cols, ramainingWalls, isWrapped, getPitCells, getBatCells
      Test the number of pit cells and superbets cells are as expected

   b. Exception
      rows < 0 or cols < 0, the remainingWalls is more than expected.
      The initial number of arrows is less than 1.

2. Test class Cell
   Test if getX(), getY(), getCellTyper(), getCellKind(), getNeighbors() return the expected value.

3. Test class Edge
   Test if getFrom() and getTo() return the expected cell value.

4. Test interface PlayerInterface
   Test if the getArrows(), getStatus(), getCurrPosition(), getPosIndex() return the expected value.