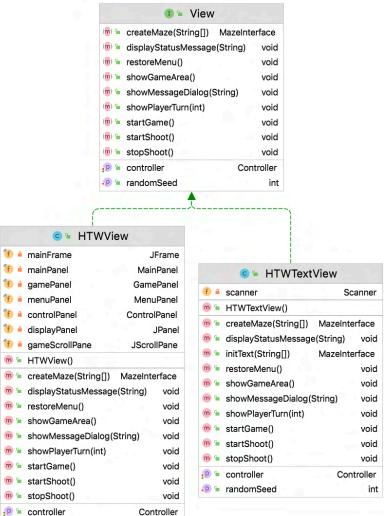


	■ Mode	el
( <mark>m</mark> ) 🔓	changeTurn()	void
( <del>m</del> ) 強	isPitNearBy(Cell)	boolean
(m) 強	isWumpusNearBy(C	Cell) boolean
( <del>m</del> ) 強	move(Direction)	String
(m) 🔓	shoot(Direction, int)	String
p 🔓	currentPlayer P	layerInterface
p 🔓	currentPlayerCell	Cell
P =	exploredCells	List <cell></cell>
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

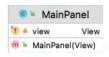
	© ► HTWMode	
m 🔓	HTWModel()	
m 🔓	changeTurn()	void
m 🤷	isPitNearBy(Cell)	boolean
m 🦆	isWumpusNearBy(Cell)	boolean
m 🤷	move(Direction)	String
m 🚡	shoot(Direction, int)	String
p 🔓	currentPlayer Playe	rInterface
p 🔓	currentPlayerCell	Cell
p 🔓	exploredCells	List <cell></cell>
P 🔓	gameOver	boolean
p 🚡	maze Maze	eInterface
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

♠ randomSeed



int

	Controller	
<b>f</b>	view	View
<b>€</b>	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 堶	move(Direction)	String
m 🚡	shoot(Direction, int)	String
m a	startGame()	void
m 🔓	startShoot()	void
m 🔓	stopShoot()	void
m a	updateNumOfArrow()	void
P %	currentPlayerArrow	int
P =	exploredCells	List <cell></cell>
p 🚡	gameOver	boolean
p 🚡	maze Maze	eInterface
P 🔓	mazeCols	int
P =	mazeRows	int
P =	player2Cell	Cell
p 🚡	playerCell	Cell
p 🔓	shooting	boolean
P 1	twoPlayerMode	boolean
P =	win	boolean
p 🔓	wumpusCell	Cell



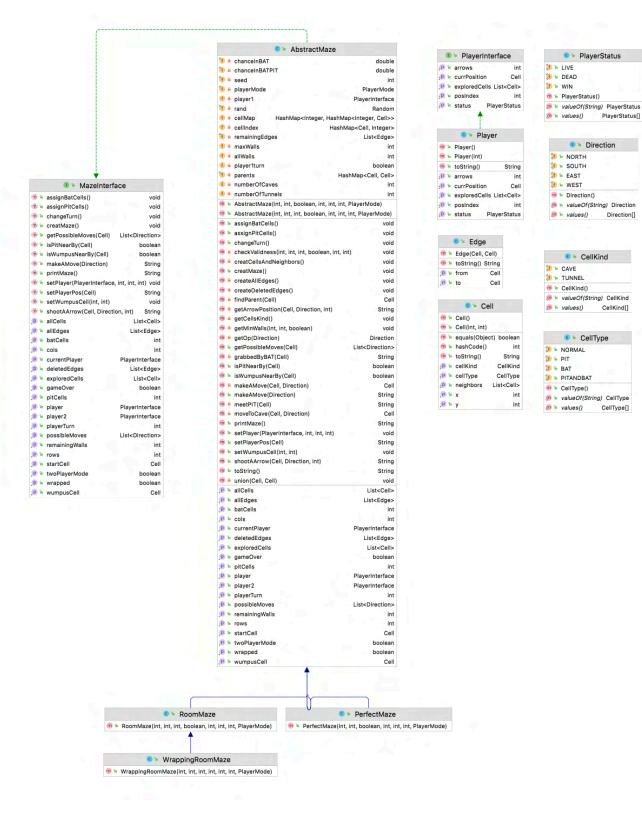
Parameters	
	int
HEIGHT	int
③ □ CELL_WIDTH	int
③ ← CELL_HEIGHT	int
₱ ■ RANDOM_SEED_RANGE	int
m = Parameters()	

	■ PlayerMode
(f) n	SINGLE_PLAYER
<b>∄</b> •	TWO_PLAYER
m ·	PlayerMode()
gn 🚡	valueOf(String) PlayerMode
gn 🦫	values() PlayerMode[]

f a vie	GamePanel	Viev
	yer2lmage	BufferedImage
	NearBylmage	BufferedImage
	impusNearBylmage	BufferedImage
f a pit		BufferedImage
f a ba		BufferedImage
	image	BufferedImage
	velmage1up	BufferedImage
	veimage iup veimage idown	BufferedImag
	velmage1left	BufferedImag
	velmage1right	BufferedImag
	velmage3NoUp	BufferedImag
	velmage3NoDown	BufferedImag
	velmage3NoLeft	BufferedImag
	velmage3NoRight	BufferedImag
	nnellmageLeftUp	BufferedImag
	nnellmageLeftDown	BufferedImag
	nnellmageRightUp	BufferedImag
	nnellmageRightDown	BufferedImag
	nnellmageStraightHor	BufferedImag
	nnellmageStraightVer	BufferedImag
	velmageMaps HashMap <integer,< td=""><td></td></integer,<>	
_	yerlmage	BufferedImag
f a ca	velmage0	BufferedImag
f a ca	velmage4	BufferedImag
n 🖢 Ga	mePanel(View)	
n 🚊 dra	awCell(Graphics, Controller, Cell)	voi
m º pa	intComponent(Graphics)	voi

	MenuPanel	
<b>1</b>	view	View
<b>⊕</b> #	numOfRowTextField	TextField
<b>●</b>	numOfColTextField	TextField
<b>⊕</b> ≜	numOfRemainingWallTextFie	ld TextField
<b>f</b> ≅	numOfSuperBatCellTextField	TextField
<b>●</b> ■	numOfBottomlessPitTextField	d TextField
<b>⊕</b> ≜	randomSeedTextField	TextField
<b>€</b> #	mazeTypeButtonGroup	ButtonGroup
<b>€</b> ≜	mazeTypePerfectButton	JRadioButtor
<b>●</b>	mazeTypeRoomButton	JRadioButtor
<b>●</b>	mazeWrappedButtonGroup	ButtonGroup
€ #	mazeWrappedButton	JRadioButtor
<b>●</b>	mazeNotWrappedButton	JRadioButtor
<b>⊕</b> ≜	playerNumOfArrowTextField	TextField
<b>f</b> a	playerNumRadioButtonGroup	ButtonGroup
♠ · · · · · · · · · · · · · · · · · · ·	onePlayerRadioButton	JRadioButtor
<b>⊕</b> ≜	twoPlayerRadioButton	JRadioButtor
m 🏲	MenuPanel(View)	
m ъ	createMaze()	MazeInterface
m 🖷	initGameSettingPanel()	JPane
m #	initMazeSettingPanel()	JPane
m A	initPlayerSettingPanel()	JPane
P 😘	randomSeed	in

ControlPanel		
<b>(1)</b> A	view	View
1	manualLabel	JTextArea
<b>(1)</b> A	currPlayerTextArea	JTextArea
€	exitBtn	JButton
<b>f</b> a	playGameBtn	JButton
<b>⊕</b> #	newGameBtn	JButton
(f) #	msgLabel	JTextArea
<b>(1)</b> A	shootDirRadioButtonGroup	ButtonGroup
(f) #	shootDirUpRadioButton	JRadioButton
(f) A	shoot Dir Down Radio Button	JRadioButton
(f) A	shootDirLeftRadioButton	JRadioButton
(F) #	shootDirRightRadioButton	JRadioButton
10 4	shootDistTextField	TextField
(f) A	shootButton	JButton
m 🦫	ControlPanel(View)	
m 🖫	disableShootComponents()	void
m 🦫	enableShootComponents()	void
m 🖫	showMessage(String)	void
m 🦫	showPlayerTurn(int)	void



## 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.