Prog. Tech 3rd Assignment - Documentation

ALI AFZAL   **3. assignment/3th task**   8th December 2022

BLVHI9
blvhi9@inf.elte.hu
Group 1

TASK – LABYRINTH GAME

Create the Labyrinth game, where objective of the player is to escape from this labyrinth. The player starts at the bottom left corner of the labyrinth. He has to get to the top right corner of the labyrinth as fast he can, avoiding a meeting with the evil dragon. The player can move only in four directions: left, right, up or down.

There are several escape paths in all labyrinths. The dragon starts off from a randomly chosen position, and moves randomly in the labyrinth so that it choose a direction and goes in that direction until it reaches a wall. Then it chooses randomly a different direction. If the dragon gets to a neighboring field of the player, then the player dies. Because it is dark in the labyrinth, the player can see only the neighboring fields at a distance of 3 units. Record the number of how many labyrinths did the player solve, and if he loses his life, then save this number together with his name into the database. Create a menu item, which displays a highscore table of the players for the 10 best scores. Also, create a menu item which restarts the game.

Take care that the player and the dragon cannot start off on walls.

Plan:

LabyrinthGUI:- for rendering window

GameManager:- the backbone of the game

Dark:- for rendering the darkness of the rest of the board

Database:- the class to configure the database to store LeaderBoard

Dragon:- storing the properties and states of the enemy dragon

Player:- storing the properties and states of the Player
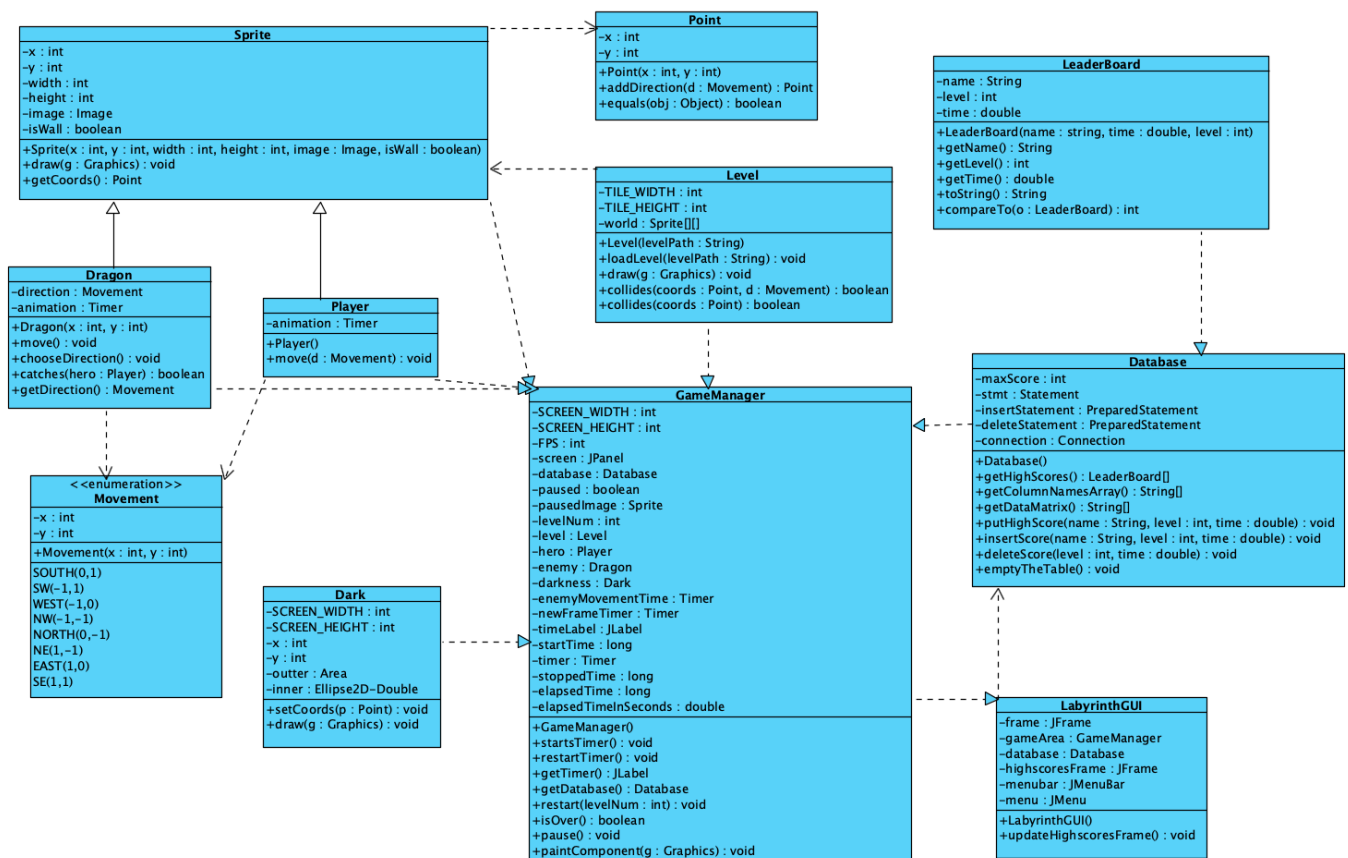
LeaderBoard:- storing the scores

Sprite:- storing the state and property of the individual cells

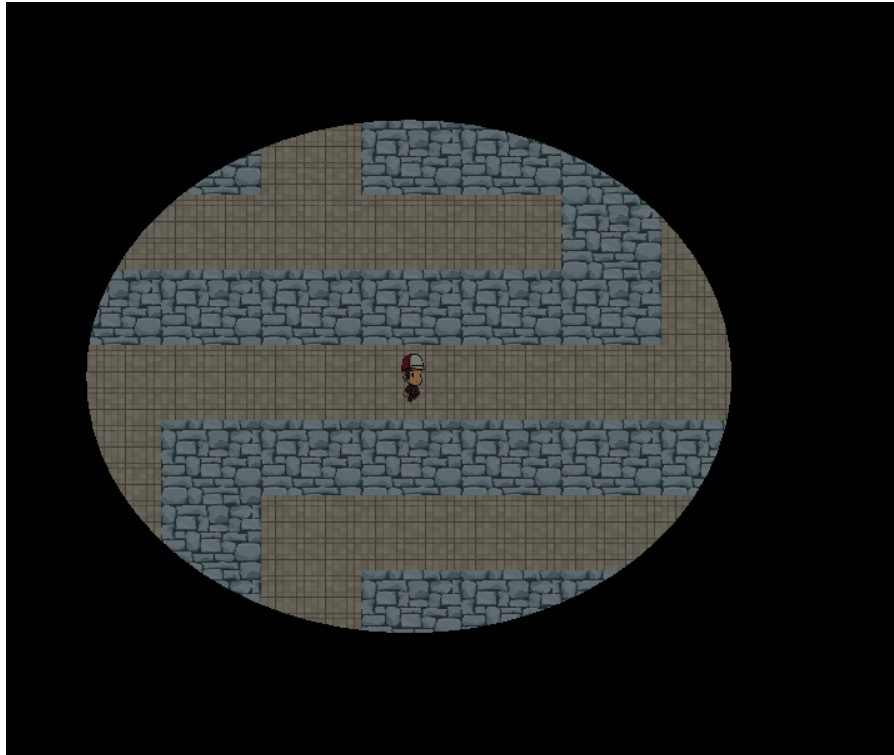**Level:-** class to implement Levels

Movement:- for directed movement

Point :- class to implement simple 2D points

UML Class Diagram:

Sprite
- -x : int
- -y : int
- -width : int
- -height : int
- -image : Image
- -isWall : boolean
- +Sprite(x : int, y : int, width : int, height : int, image : Image, isWall : boolean)
- +draw(g : Graphics) : void
- +getCoords() : Point

Point
- -x : int
- -y : int
- +Point(x : int, y : int)
- +addDirection(d : Movement) : Point
- +equals(obj : Object) : boolean

LeaderBoard
- -name : String
- -level : int
- -time : double
- +LeaderBoard(name : string, time : double, level : int)
- +getName() : String
- +getLevel() : int
- +getTime() : double
- +toString() : String
- +compareTo(o : LeaderBoard) : int

Level
- -TILE_WIDTH : int
- -TILE_HEIGHT : int
- -world : Sprite[][]
- +Level(levelPath : String)
- +loadLevel(levelPath : String) : void
- +draw(g : Graphics) : void
- +collides(coords : Point, d : Movement) : boolean
- +collides(coords : Point) : boolean

Dragon
- -direction : Movement
- -animation : Timer
- +Dragon(x : int, y : int)
- +move() : void
- +chooseDirection() : void
- +catches(hero : Player) : boolean
- +getDirection() : Movement

Player
- -animation : Timer
- +Player()
- +move(d : Movement) : void

<<enumeration>> Movement
- -x : int
- -y : int
- +Movement(x : int, y : int)
- SOUTH(0,1)
- SW(-1,1)
- WEST(-1,0)
- NW(-1,-1)
- NORTH(0,-1)
- NE(1,-1)
- EAST(1,0)
- SE(1,1)

Dark
- -SCREEN_WIDTH : int
- -SCREEN_HEIGHT : int
- -x : int
- -y : int
- -outter : Area
- -inner : Ellipse2D-Double
- +setCoords(p : Point) : void
- +draw(g : Graphics) : void

GameManager
- -SCREEN_WIDTH : int
- -SCREEN_HEIGHT : int
- -FPS : int
- -screen : JPanel
- -database : Database
- -paused : boolean
- -pausedImage : Sprite
- -levelNum : int
- -level : Level
- -hero : Player
- -enemy : Dragon
- -darkness : Dark
- -enemyMovementTime : Timer
- -newFrameTimer : Timer
- -timeLabel : JLabel
- -startTime : long
- -timer : Timer
- -stoppedTime : long
- -elapsedTime : long
- -elapsedTimeInSeconds : double
- +GameManager()
- +startsTimer() : void
- +restartTimer() : void
- +getTimer() : JLabel
- +getDatabase() : Database
- +restart(levelNum : int) : void
- +isOver() : boolean
- +pause() : void
- +paintComponent(g : Graphics) : void

Database
- -maxScore : int
- -stmt : Statement
- -insertStatement : PreparedStatement
- -deleteStatement : PreparedStatement
- -connection : Connection
- +Database()
- +getHighScores() : LeaderBoard[]
- +getColumnNamesArray() : String[]
- +getDataMatrix() : String[]
- +putHighScore(name : String, level : int, time : double) : void
- +insertScore(name : String, level : int, time : double) : void
- +deleteScore(level : int, time : double) : void
- +emptyTheTable() : void

LabyrinthGUI
- -frame : JFrame
- -gameArea : GameManager
- -database : Database
- -highscoresFrame : JFrame
- -menubar : JMenuBar
- -menu : JMenu
- +LabyrinthGUI()
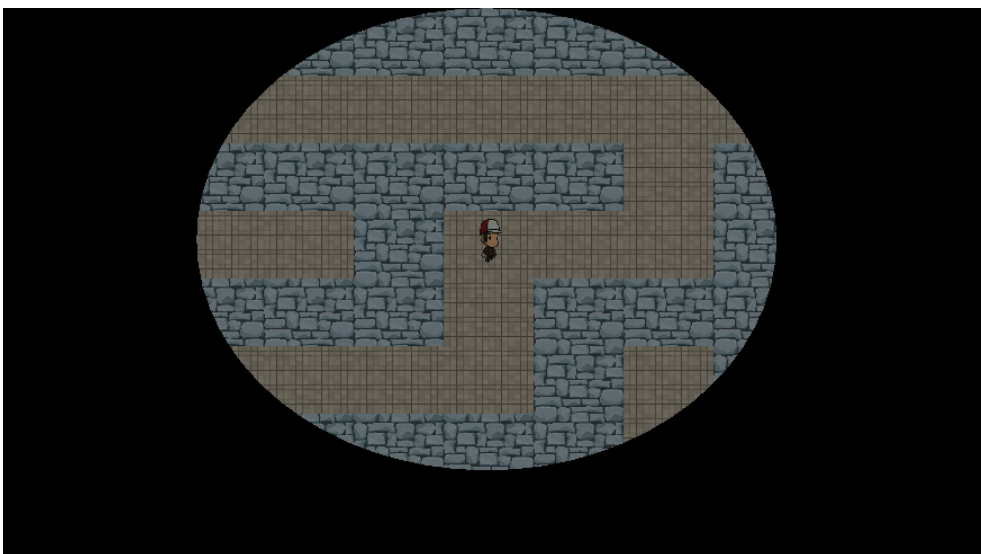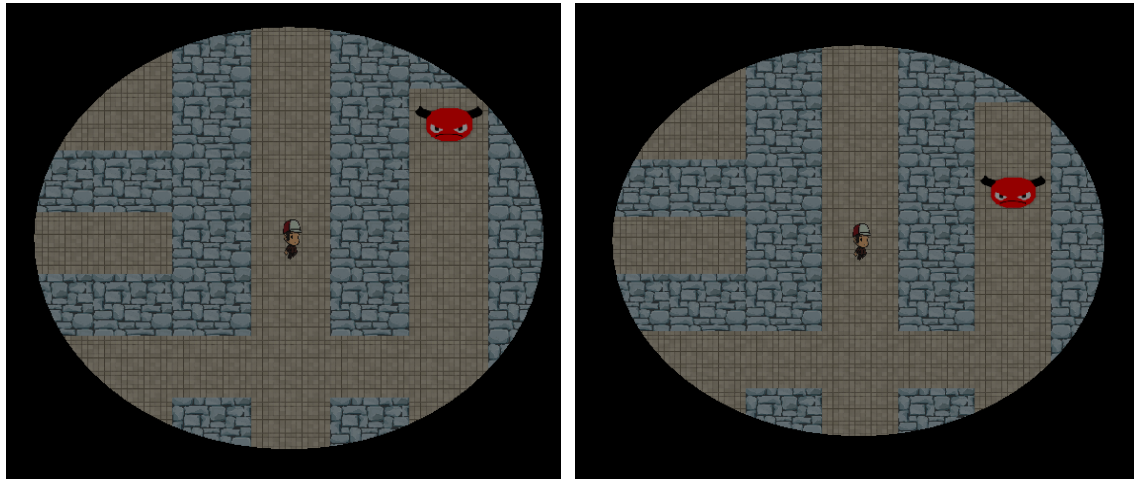- +updateHighscoresFrame() : void

Testing:

Test 1 : Check whether we can see only 3 adjacent units.(Darkness check)

Test 2 : Check whether the player will stop when touching the wall.

Test 3 : Test whether the dragon will automatically turn when it touches the wall.

Test 4 : Test whether LeaderBoard displays the results correctly.

Labyrinth

Menu

Highscores			
#	Name	Levels completed	Time
1	SArunnnnn	7	119.49 s
2	www	5	79.765 s
3	Ali	4	78.268 s
4	ss	3	78.052 s
5	d	2	69.896 s
6	Sorry	1	17.799 s
7	D	1	25.311 s

3.318 s