Our Lost Friend

UML Diagrams

Start
f : JFrame background : ImageIcon backgroundLabel : JLabel login : JButton register : JButton userName : JTextArea password : JPasswordField fileName : String in : BufferedReader out : BufferedWriter accounts : String[][] users : String[] passwords : String[] scores : String[] numOfUsers : int curUser : String curPass : String
+ Start() + saveUsers() : void + getUsers() : int + getUser() : String + getAccounts() : String[][] + getPass() : String + addUser() : void + makeButton(b : JButton) : void + accessGranted() : void + register() : void + login() : void + actionPerformed(e : ActionEvent) : void + userCheck() : boolean

TitleScreen
f : JFrame userName : String backgroundLabel : JLabel titleLabel : JLabel continueButton : JButton Title : ImageIcon buttonFont : Font biggerFont: Font background : ImageIcon clip : Clip

+ TitleScreen()
+ actionPerformed(e : ActionEvent) : void
+ makeContinueButton(continueButton : JButton)

MainMenu
f : JFrame userName : String exitGame : JButton leaderboard : JButton gameStart : JButton tutorial : JButton changePassword : JButton backgroundLabel : JLabel titleLabel : JLabel authors : JLabel signedInAs : JLabel buttonFont : Font biggerFont : Font textFont : Font changedPass : boolean clip : Clip background : ImageIcon Title : ImageIcon
+ MainMenu() + makeButton(b: JButton) : void + makeLabel(l : JLabel) : void + actionPerformed(e : ActionEvent) : void + reprompt() : void

TutorialPage
f: JFrame back: JButton background: ImageIcon Title: ImageIcon backgroundLabel: JLabel titleLabel: JLabel text: JLabel text2: JLabel text3: JLabel text4: JLabel text5: JLabel buttonFont : Font biggerFont : Font textFont : Font
+ TutorialPage() + makeButton(b: JButton) : void + makeLabel(l : JLabel) : void + actionPerformed(e : ActionEvent) : void

changePassword
f: JFrame accounts: String[][] username: JTextArea password: JPasswordField newPassword: JPasswordField backgroundLabel: JLabel changePassword: JButton done: JButton fileName: String out: BufferedWriter numOfUsers: int curUser: String curPass: String enteredUser: String enteredPass: String buttonFont: Font
+ changePassword() + makeButton(b : JButton) : void + actionPerformed(e : ActionEvent) : void + changePass(): void + saveUsers(): void

Leaderboard
f: JFrame fileName: String userName: String numUsers: int accounts: String[][] buttonFont : Font biggerFont : Font textFont : Font backgroundLabel: JLabel titleLabel: JLabel signedInAs: JLabel scores: JLabel background: ImageIcon Title: ImageIcon back: JButton
+ Leaderboard() + displayLabel(): void + makeTextLabel(l : JLabel): void + makeBackButton(b : JButton): void + actionPerformed(e : ActionEvent) : void + updateScore(playerName : String, updatedScore : int): void

Game
numUsers: int players: PlayerType[] accounts: String[][] username: String userNum: int fileName: String out: BufferedWritter Frame: JFrame
+ Game() + getUserScore(username : String) + PlayerType[] getPlayers() + saveScore(updatedScore : int): void + saveData(): void + getUserNum(user : String) + close(): void + end(): void

Background	
	x : double y : double dx : double dy : double
+	Background(s : String)
+	setPosition(x : double, y : double) : void
+	setVector(dx : double, dy : double) : void
+	update() : void
+	draw(g : Graphics2D) : void

PlayerType	
	name: String score: int
+	PlayerType(name : String, score : int)
+	PlayerType(name : String)
+	PlayerType(score : int)
+	PlayerType()
+	getName() : String
+	getScore() : int
+	setScore(score : int): void
+	setName(name : String): void
+	sortPlayers(p : PlayerType[]): void

MenuState	
	buttons : String[] currentSelection : int buttonFont : Font biggerFont : Font background : Image
+	MenuState(gm : GameManager)
+	init() : void
+	tick() : void
+	draw(g : Graphics) : void
+	keyPressed(k : int) : void
+	keyReleased(k : int) : void

GameState	
	gm : GameManager xOffset : double yOffset : double startX : double startY : double farthest : double
+	GameState(gm : GameManager)
+	init() : void
+	tick() : void
+	draw(g : Graphics) : void
+	keyPressed(k : int) : void
+	keyReleased(k : int) : void
+	menu() : void

Level1State	
	player : Player map : Map
+	Level1State(gm : GameManager)
+	init() : void
+	tick() : void
+	draw(g : Graphics) : void
+	keyPressed(k : int) : void
+	keyReleased(k : int) : void

GamePanel

WIDTH : int
HEIGHT : int
backgrounds : Background[]
bg : Background
thread : Thread
isRunning : boolean
FPS : int
targetTime : long
gm : GameManager
wait : long
start : long
elapsed : long

+ GamePanel()
+ start() : void
+ run() : void
+ tick() : void
+ paintComponent(g : Graphics) : void
+ keyTyped(e : KeyEvent) : void
+ keyPressed(e : KeyEvent) : void
+ keyReleased(e : KeyEvent) : void

GameManager

states : Stack<GameState>

+ tick() : void
+ draw(g : Graphics) : void
+ keyPressed(k : int) : void
+ keyReleased(k : int) : void

Collision

+ playerBlock(p : Point, b : Block)
+ playerMovingBlock(p : Point, b : MovingBlock)

Level2State

player : Player
map : Map

+ Level2State(gm : GameManager)
+ init() : void
+ tick() : void
+ draw(g : Graphics) : void
+ keyPressed(k : int) : void
+ keyReleased(k : int) : void

Level3State

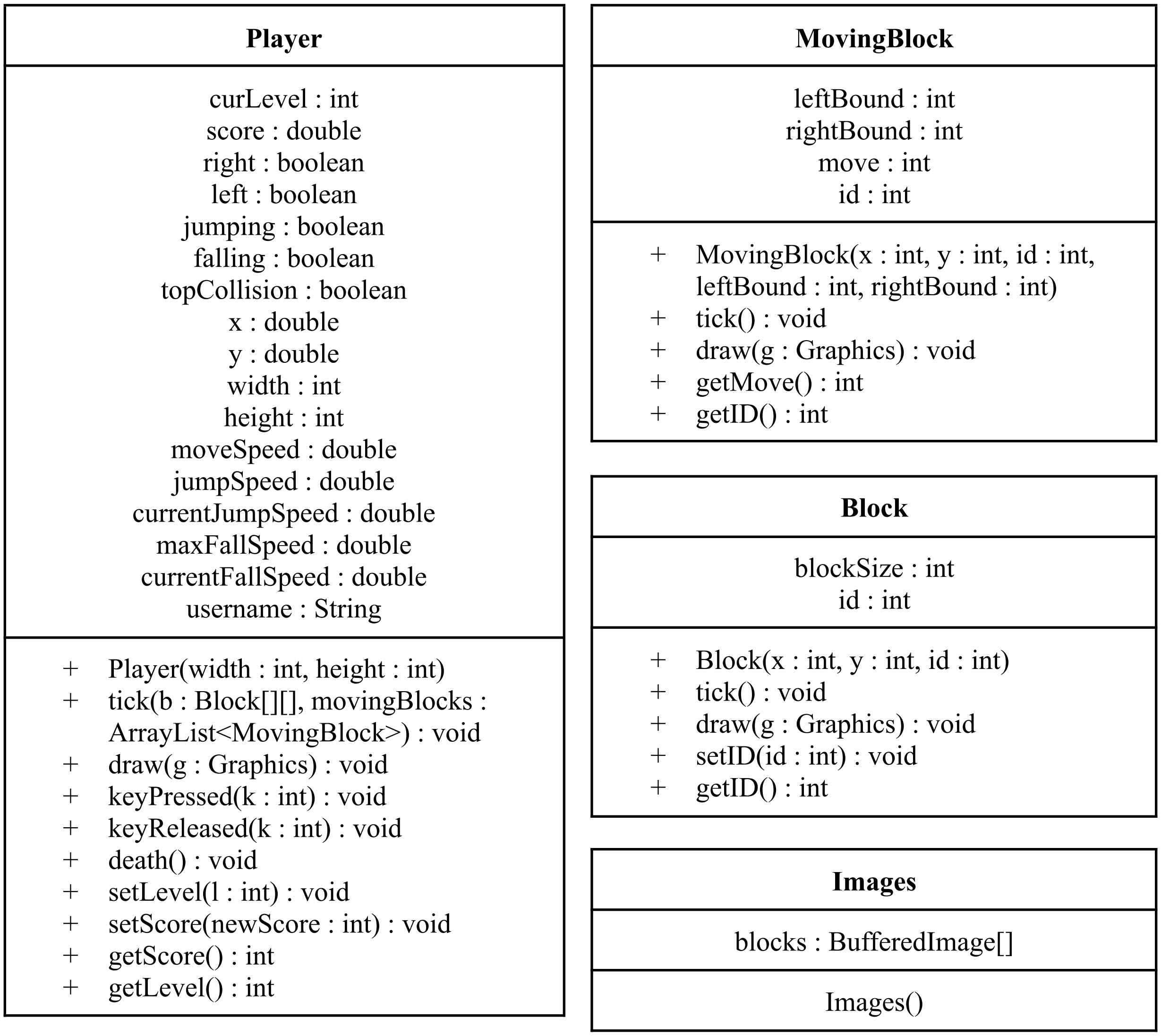
player : Player
map : Map

+ Level3State(gm : GameManager)
+ init() : void
+ tick() : void
+ draw(g : Graphics) : void
+ keyPressed(k : int) : void
+ keyReleased(k : int) : void

Map

path : String
line : String
width : int
height : int
movingBlocks : ArrayList<MovingBlock>
blocks: Blocks[][]

+ Map(mapPath : String)
+ draw (g : Graphics) : void
+ loadMap() : void
+ tick() : void
+ getBlocks() : Block[][]
+ getMovingBlocks() : ArrayList<MovingBlock>



Player

curLevel : int
score : double
right : boolean
left : boolean
jumping : boolean
falling : boolean
topCollision : boolean
x : double
y : double
width : int
height : int
moveSpeed : double
jumpSpeed : double
currentJumpSpeed : double
maxFallSpeed : double
currentFallSpeed : double
username : String

+ Player(width : int, height : int)
+ tick(b : Block[][], movingBlocks : ArrayList<MovingBlock>) : void
+ draw(g : Graphics) : void
+ keyPressed(k : int) : void
+ keyReleased(k : int) : void
+ death() : void
+ setLevel(l : int) : void
+ setScore(newScore : int) : void
+ getScore() : int
+ getLevel() : int

MovingBlock

leftBound : int
rightBound : int
move : int
id : int

+ MovingBlock(x : int, y : int, id : int, leftBound : int, rightBound : int)
+ tick() : void
+ draw(g : Graphics) : void
+ getMove() : int
+ getID() : int

Block

blockSize : int
id : int

+ Block(x : int, y : int, id : int)
+ tick() : void
+ draw(g : Graphics) : void
+ setID(id : int) : void
+ getID() : int

Images

blocks : BufferedImage[]

Images()

Project Functionalities

Start/Login Page:

- The user is presented with two text areas, one for their username and one for their password
- If they do not have an account then they are able to register for a new account
- If they do they are sent to the main menu
- Account information will be stored in a text file named “accounts.txt”
- User will input this information for the first time when signing up
- User will re-enter this information when logging in to the game

Main menu:

- User is shown five buttons on a main menu where they will choose to either
    - View the leaderboard
    - Exit the game
    - See the instructions on how to play the game
    - Start the game
    - Change their account’s password
- In the top right corner, there will be a notification stating the account that the current instance is logged into.

Change Password Page:

- In this section of the game, the player will be able to change their current account’s password
- The window prompts the user to enter their username, password, and new password
- If the credentials are correct, the password will change to the new password entered by the user and stored inside of “accounts.txt”

Leaderboard

- High scores of each account will be taken from accounts.txt
- High scores will be displayed on screen when the player goes to view the leaderboard in game
- The leaderboard stores up to a total of 10 top scores from different players

Game Exit:

- Exits the game

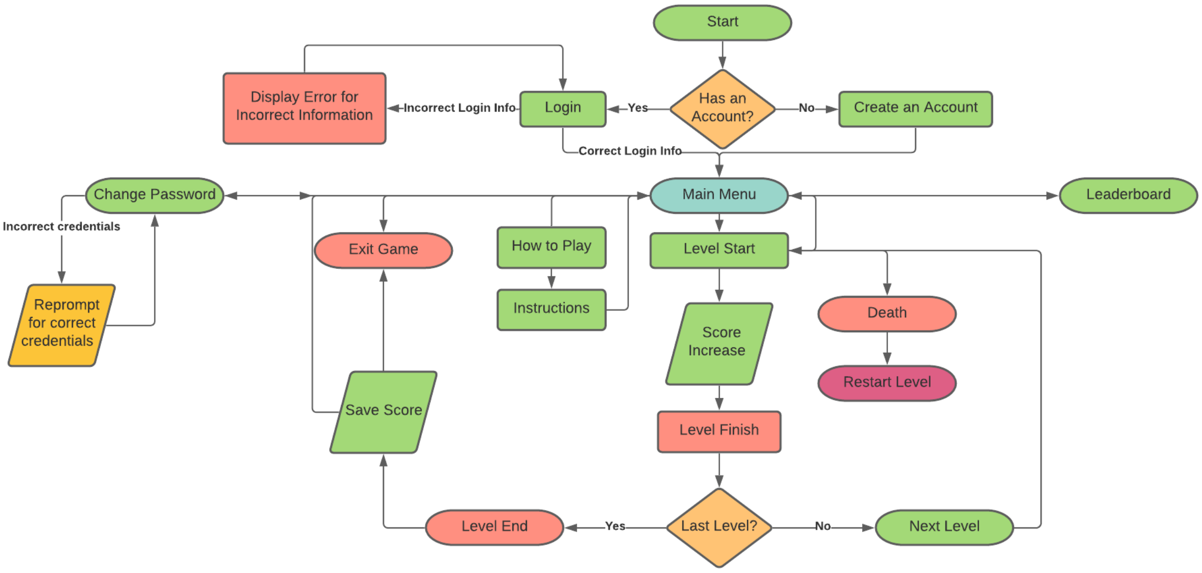
Tutorial Game:

- Tells the user the objectives that they must achieve to complete a level
- To survive the user is to progress through each level while avoiding obstacles
- The user will gain points just by moving forward(1 point per second alive)

Game Play:

- After the player presses play the user will be immediately taken into the menu for the game
- In the levels of this game, the player must progress by moving to the right and avoiding bumping into obstacles and falling
- The player will complete the level when he or she reaches the end of the current level they are on
- After the first level is completed the player will be taken to the next level
- The player's score will saved as they are progressing and will be saved on the leaderboard under the player's account name
- If the player dies in the game they will have to restart the level
- The player can press escape when they are in the game to return to the game's menu
- The player will move to the left and right and jump by pressing A, D, and space bar respectively

Flow Chart


Start
Has an Account?
Yes
No
Login
Create an Account
Incorrect Login Info
Display Error for Incorrect Information
Correct Login Info
Main Menu
Change Password
Incorrect credentials
Reprompt for correct credentials
Leaderboard
Exit Game
How to Play
Instructions
Level Start
Score Increase
Death
Restart Level
Level Finish
Save Score
Last Level?
Yes
Level End
No
Next Level