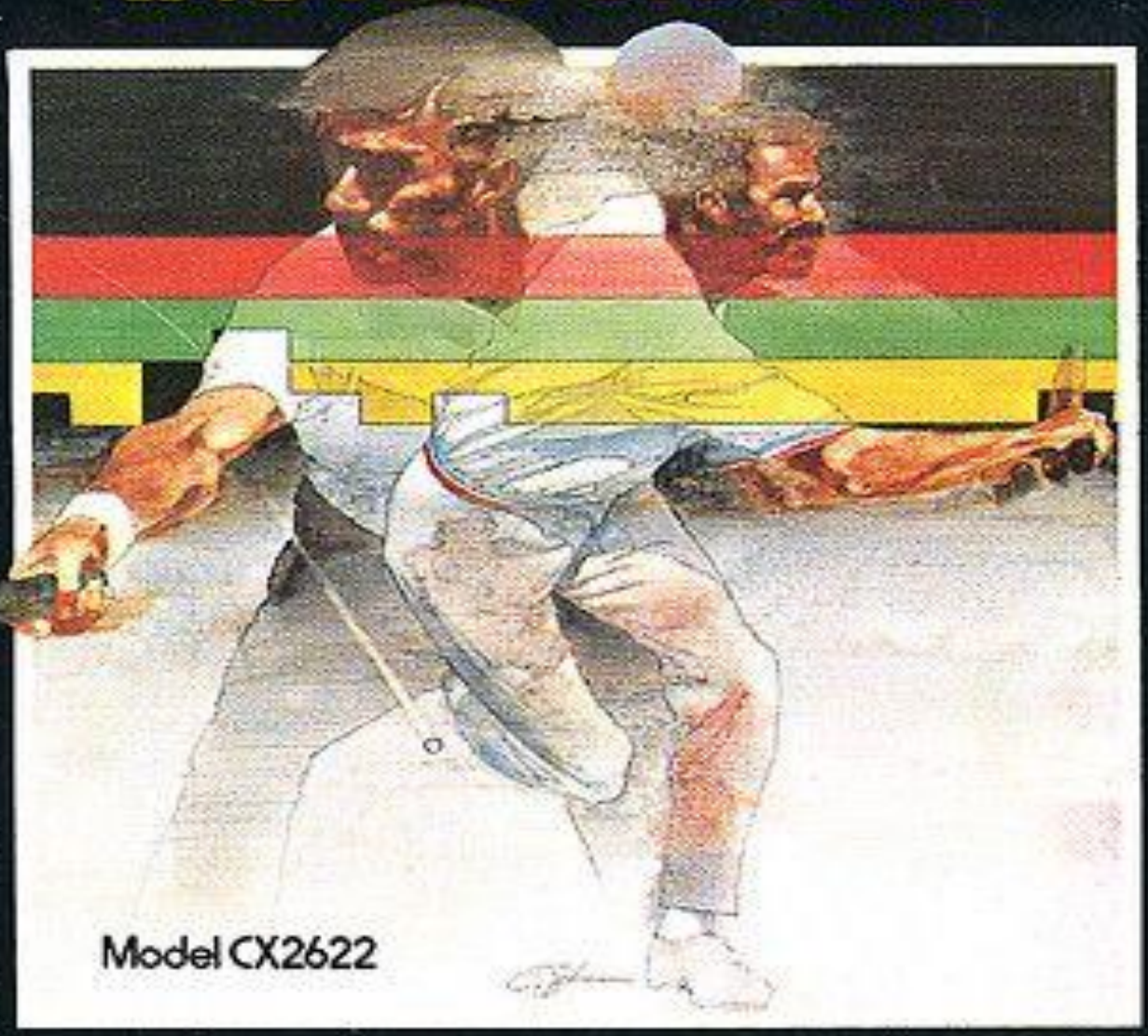
BREAKOUT™

GAME PROGRAM™ INSTRUCTIONS



ATARI®

A Warner Communications Company

ATARI, INC., Consumer Division
1195 Borregas Ave., Sunnyvale, CA 94086

- جامعة الزقازيق
- كلية الهندسة
- قسم الهندسة الكهربية
- الفرقة الثانية

Javafx Project Report :

‘Breakout game’

مقدم من : **Group no. 23**

1 - إبراهيم محمد سعيد حسنين
2 - احمد فوزى فتحي السيد
3 - عبد الرحمن علاء حمدي محمود
4 - محمود عبد الرحمن السيد عبد المقصود
5 - محمود نجدي السيد سلوم
6 - محمود هشام محمود محمد عبد العليم
7 - مصطفى احمد عبد الموجود إبراهيم
8 - مصطفى طاهر محمد أحمد أبو النصر " Team Leader "
9 - مصطفى محمد احمد رأفت احمد
10 - مصطفى محمود يحيى محمود

مقدم إلي : م / احمد عبد الباسط ـ تحت اشراف : د / عمرو الزامل

Abstract :

The breakout game project with JavaFX is a classic arcade-style game that challenges players to break through a wall of bricks using a paddle and a ball. The game is built using the JavaFX platform, which provides an intuitive and user-friendly interface for players to interact with the game.

The project uses object-oriented programming principles to create a modular and extensible design. The game features multiple levels of increasing difficulty, with each level having its own unique set of challenges and obstacles.

JavaFX's built-in animation capabilities are used to create smooth and responsive gameplay, with realistic physics simulations for the ball and paddle. The game's graphics are implemented using JavaFX's scene graph, allowing for easy manipulation and customization of the game's visual elements.

Overall, the breakout game project with JavaFX is a fun and engaging way to learn about game development, object-oriented programming, and the JavaFX platform. With its modular design and customizable elements, it provides a great starting point for developers looking to build their own arcade-style games.

Pseudocode :

Class 1 : Level one :

Creating properties for Pane & Nodes

Start ( ) :

Create scene , backgrounds , buttons and Stage.

Start game ( ) :

Clear Pane , Create map , Start Timeline

Create Blocks ( ) :

2 loop to build rows and columns with different colors.

Create Paddle ( ) :

Mouse & Keyboard movements

Create Ball ( ) .

Create Scoreboard ( ) :

Lives & Score

Move Ball ( ) :

Start move & control impact with borders .

Check Collisions ( ) :

Forever loop to check ball impact with blocks & paddle , Check score to produce the Win , enhance ball speed depends on ball Position Relative To Paddle.

Lose Life ( ) :

Update lives with fall ball and produce the end game. & reset ball call

Reset ball ( ) :

Bring ball back above middle of paddle.

End Game Lose ( ) :

Game Over Screen .

End Game Win ( ) :

Clear Pane , Change background , exit or next level buttons.

Class 2 : Level two :

Create object to use class one methods.

Start ( ) :

Same as level one

Start Game ( ) :

Calling level one methods

Create Blocks ( ) :

2 loop to build rows and columns with same color but different layout.

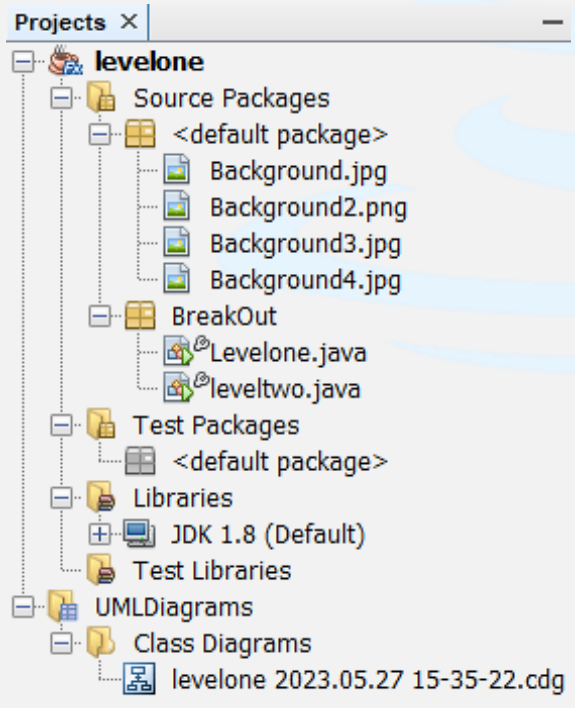
Check Collisions ( ) :

Forever loop to check ball impact with blocks & paddle , Check score to produce the Win , enhance ball speed depends on ball Position Relative To Paddle , change color at one hit and remove at the second.

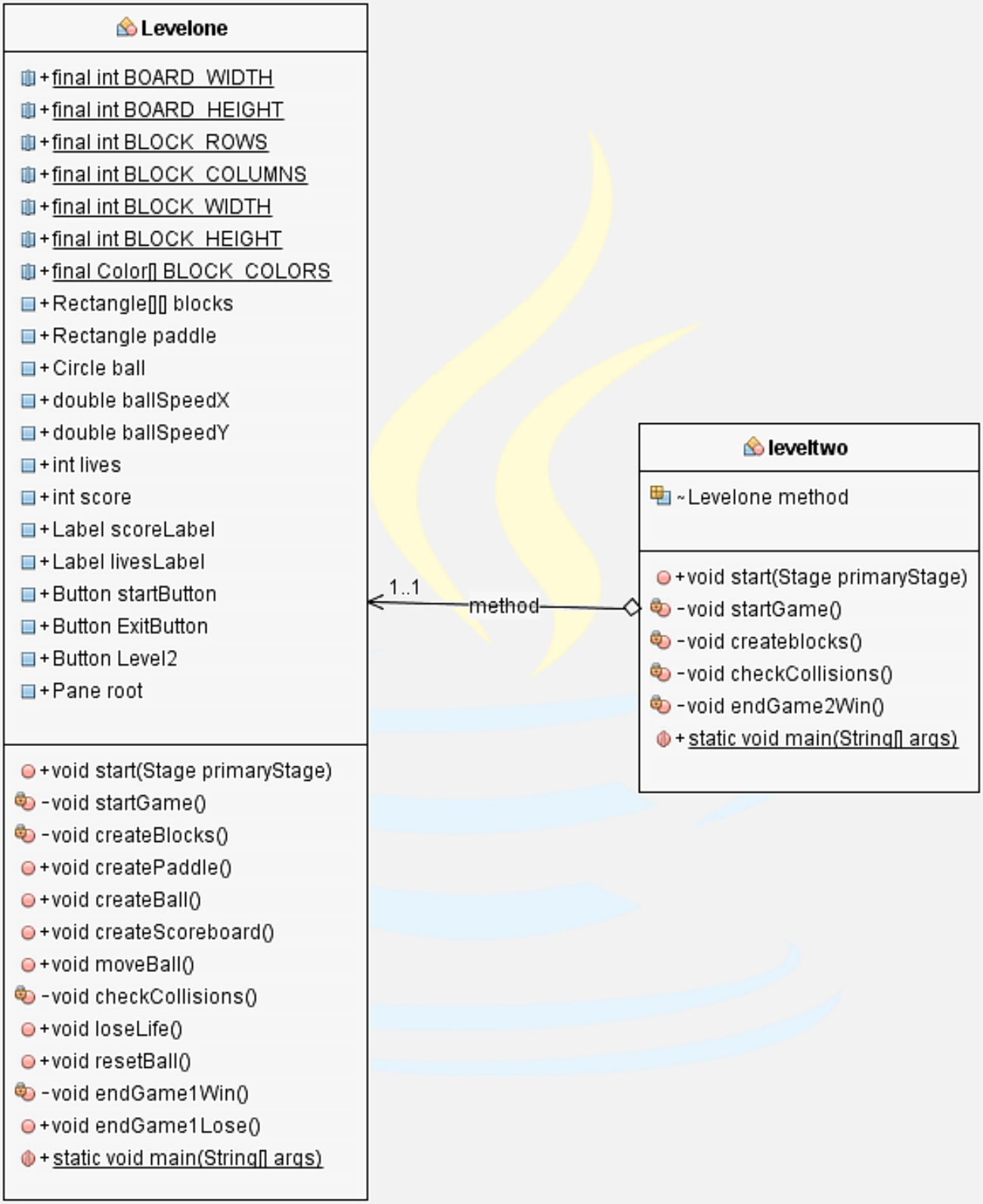
End Game 2 Win ( ) :

Clear Pane , Change background , exit and Congratulations.

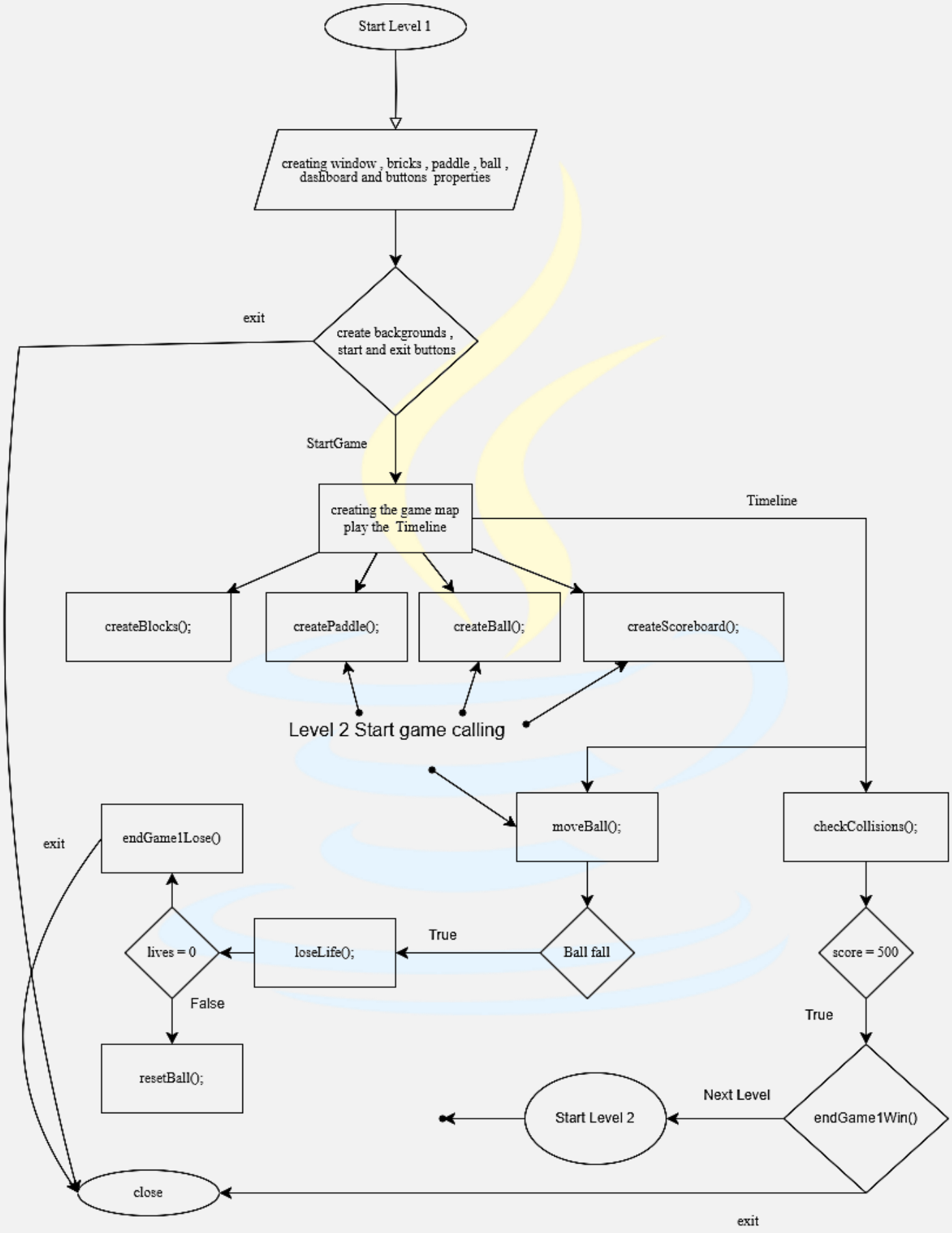
Packages :



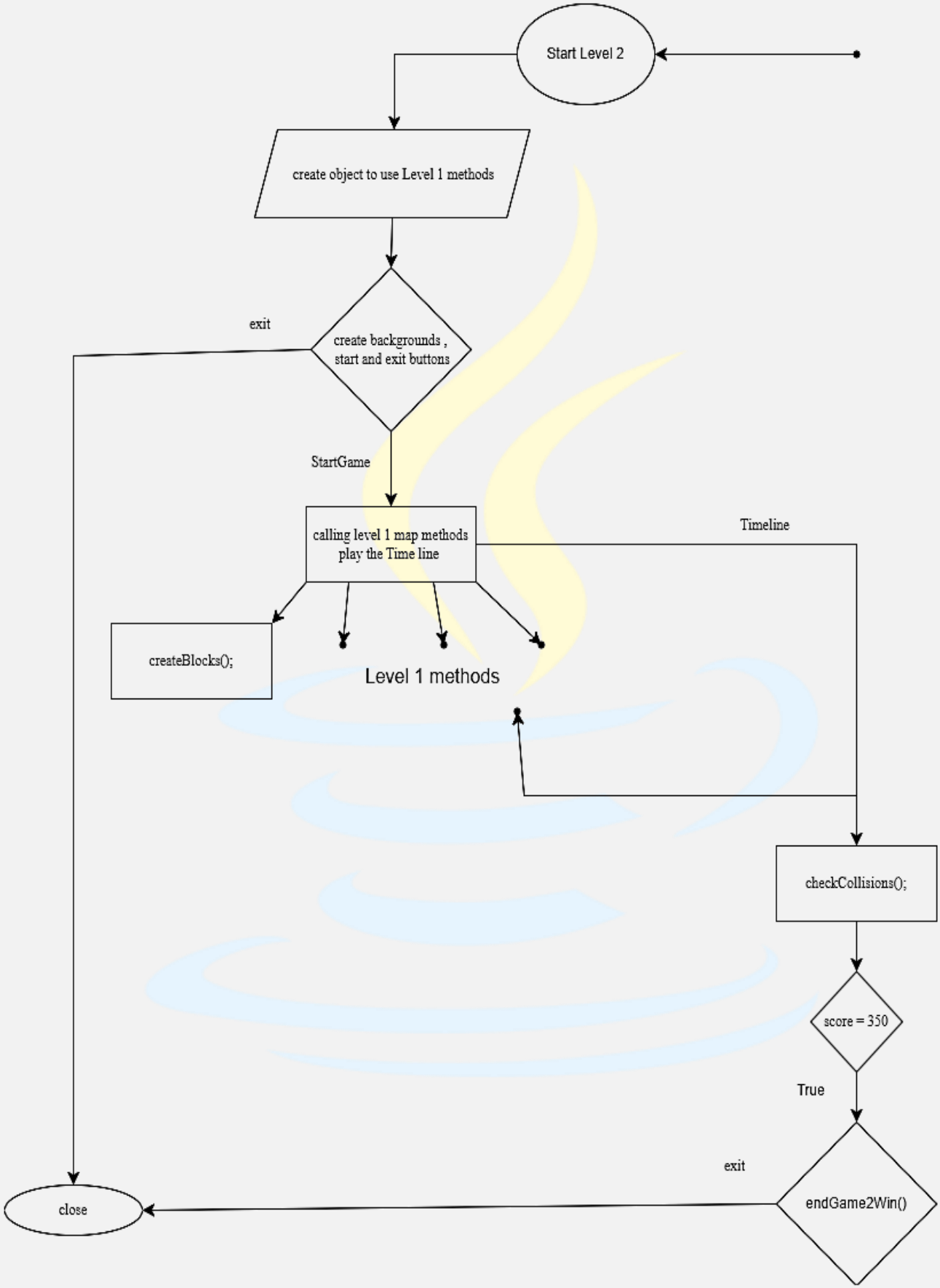
UML :


Levelone
+final int BOARD_WIDTH
+final int BOARD_HEIGHT
+final int BLOCK_ROWS
+final int BLOCK_COLUMNS
+final int BLOCK_WIDTH
+final int BLOCK_HEIGHT
+final Color[] BLOCK_COLORS
+Rectangle[][] blocks
+Rectangle paddle
+Circle ball
+double ballSpeedX
+double ballSpeedY
+int lives
+int score
+Label scoreLabel
+Label livesLabel
+Button startButton
+Button ExitButton
+Button Level2
+Pane root
+void start(Stage primaryStage)
-void startGame()
-void createBlocks()
+void createPaddle()
+void createBall()
+void createScoreboard()
+void moveBall()
-void checkCollisions()
+void loseLife()
+void resetBall()
-void endGame1Win()
+void endGame1Lose()
+static void main(String[] args)
1..1
method
leveltwo
~Levelone method
+void start(Stage primaryStage)
-void startGame()
-void createblocks()
-void checkCollisions()
-void endGame2Win()
+static void main(String[] args)

Flowchart Level one :

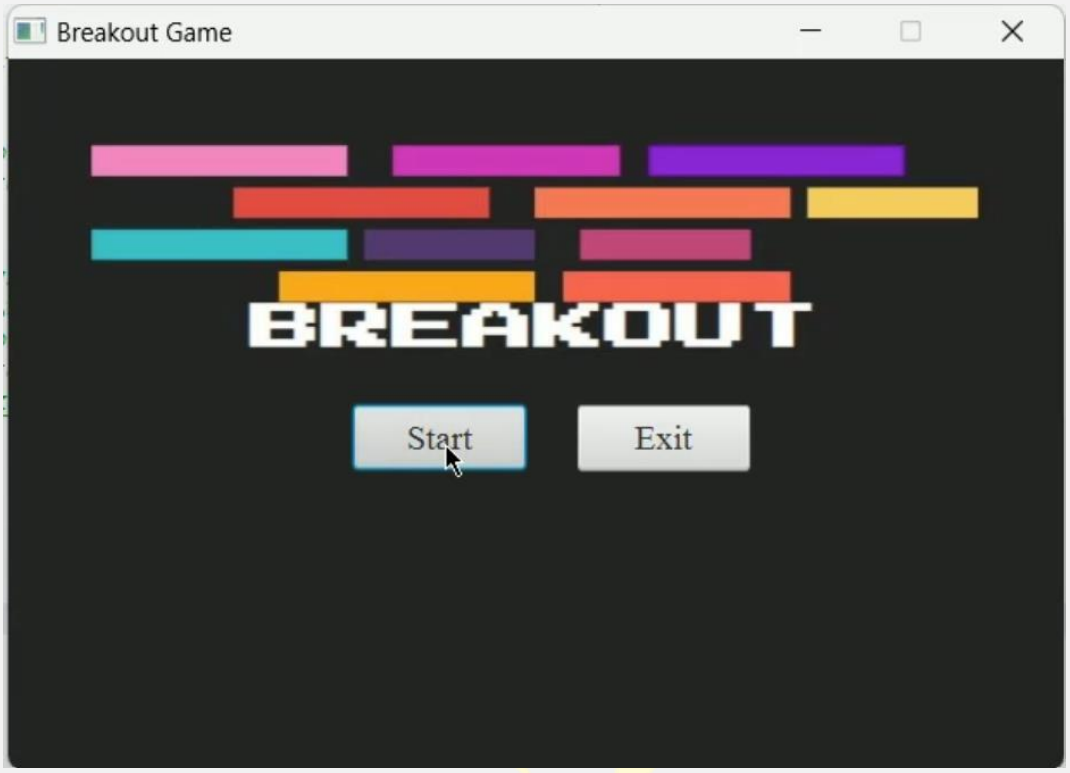


Flowchart Level Two :

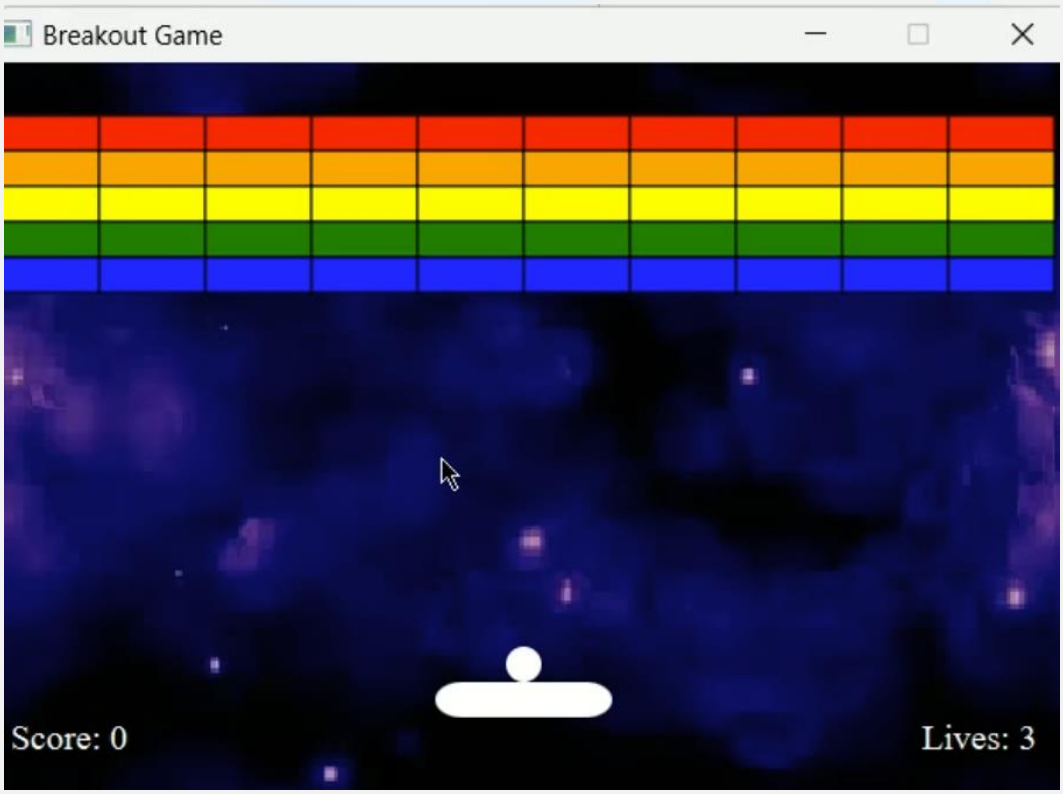


Gameplay :

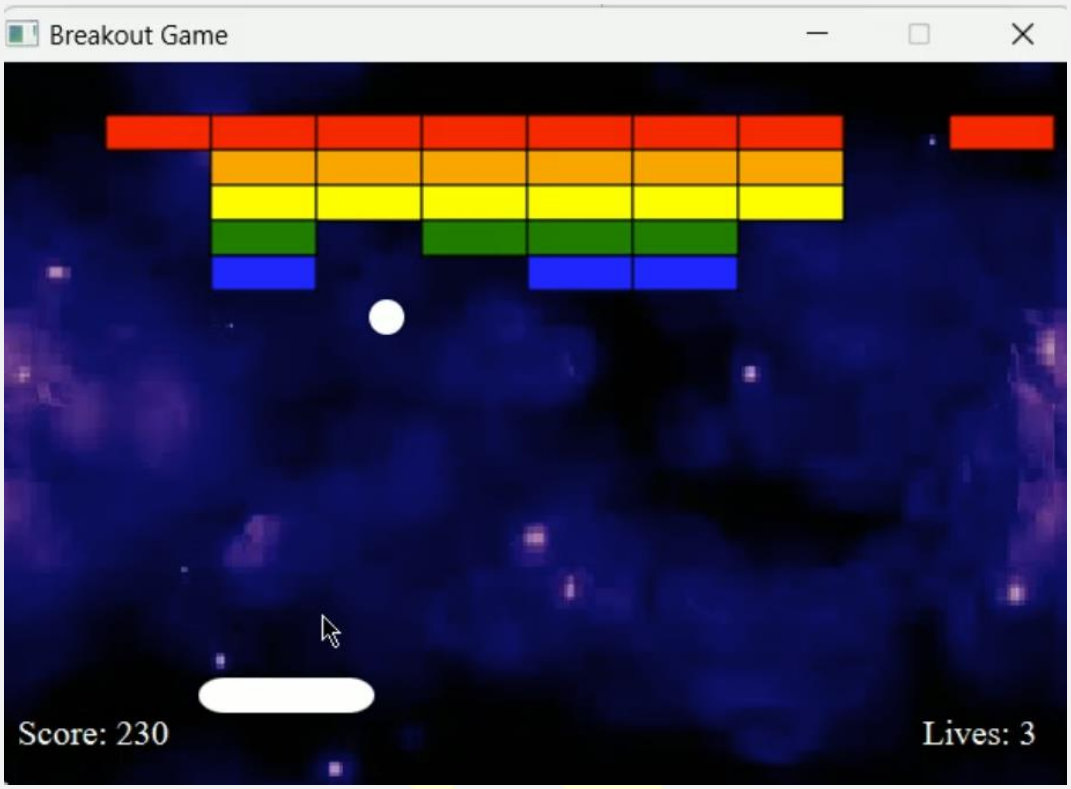
Start game Screen.



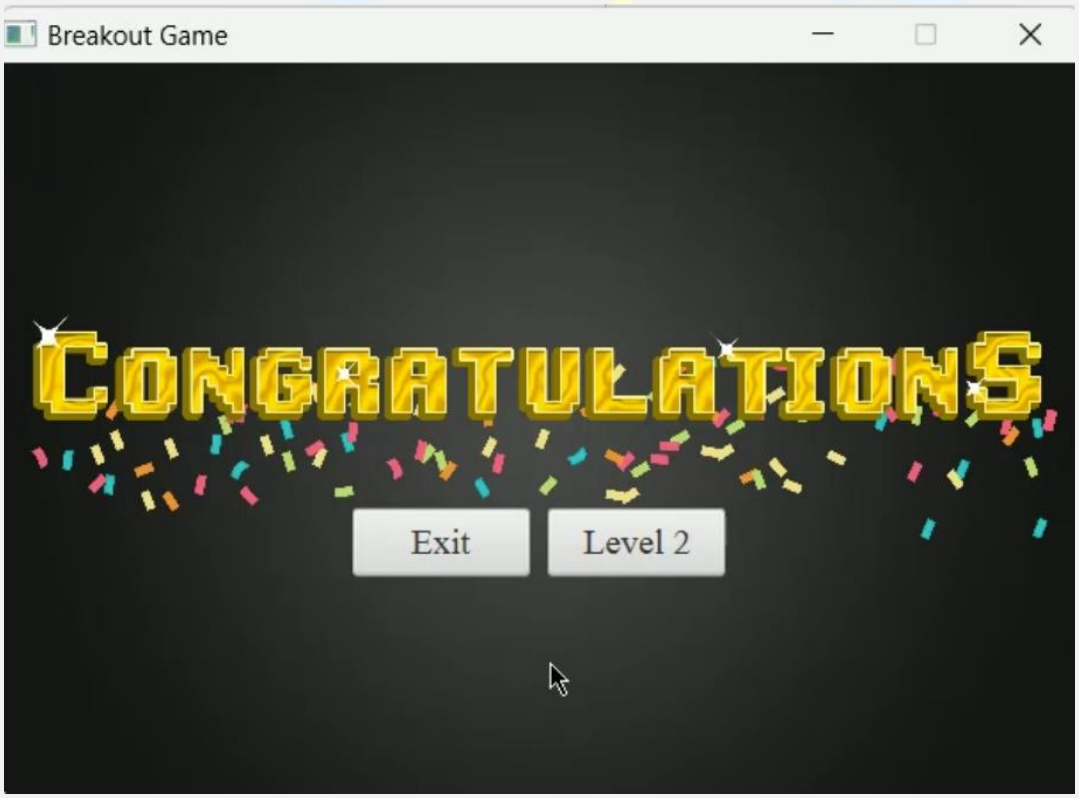
Level one Start Screen.



Level one playing time Screen.

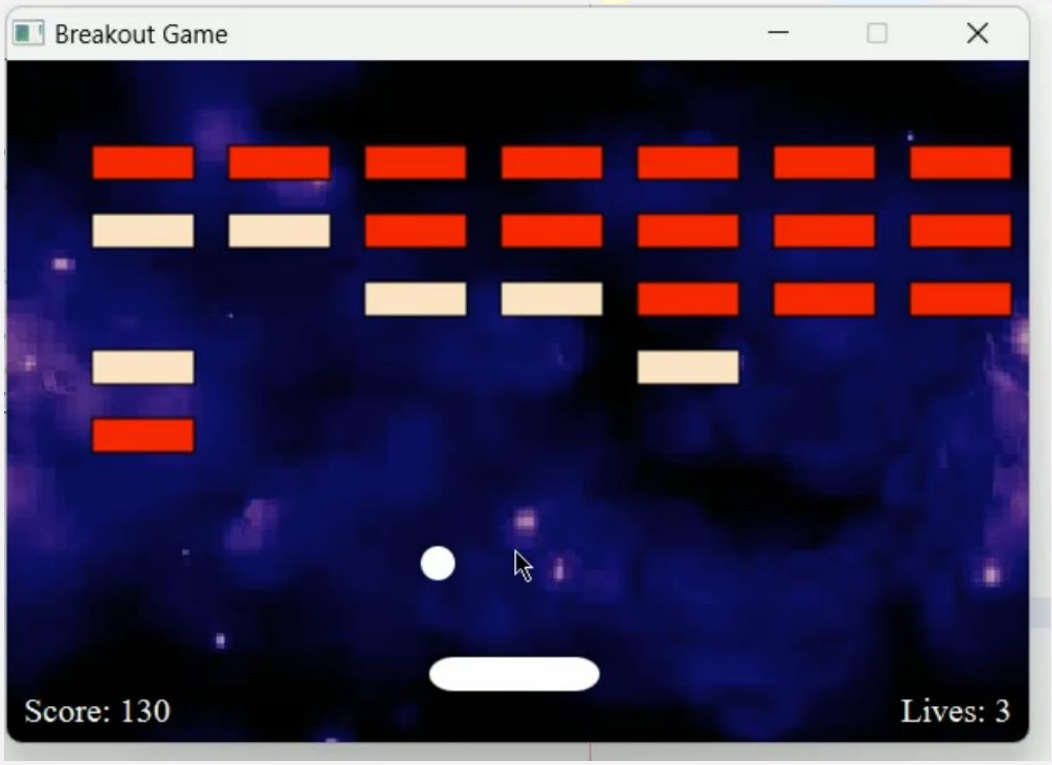


Level one End Screen Win.

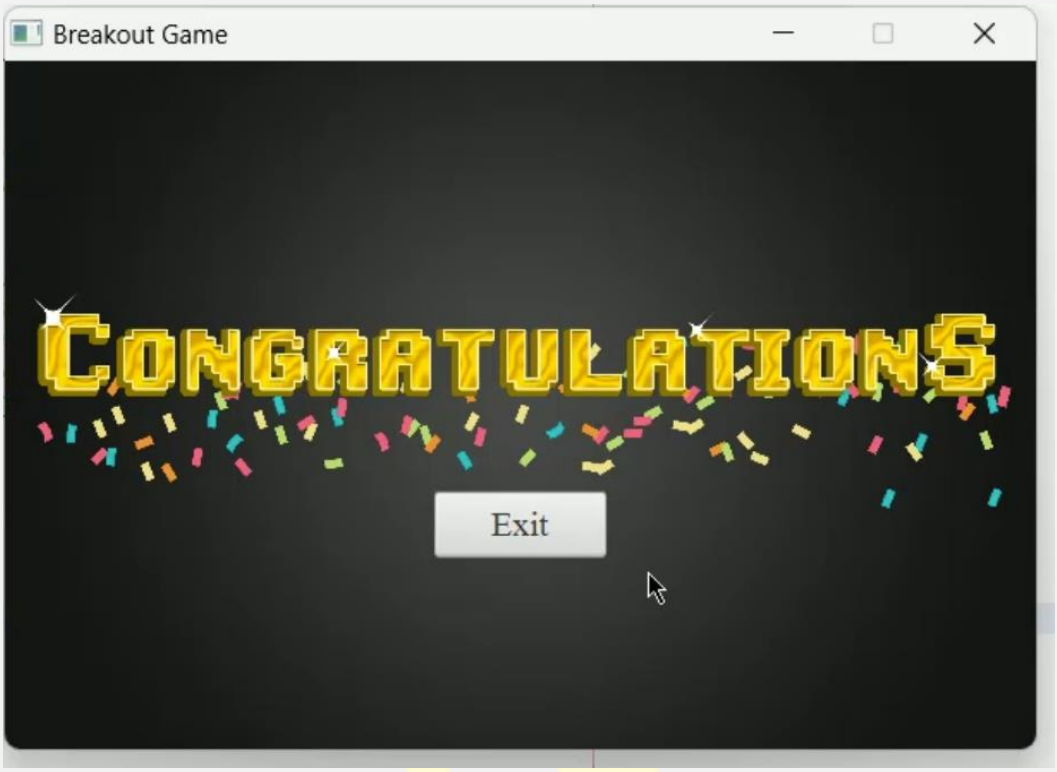


Level Two Start Screen.

Level two playing time Screen.



Level one End Screen Win.



End Screen in case Lose.

Problems Solved :

1 : how to build rows and columns of blocks

sol : use an array of rectangles .

2 : prevent paddle to get out of the Scene using arrow keys

sol : making condition (if paddle X > 0 for left move)

(if paddle X < 500 for right move)

3 : deal with ball fall.

sol : create lives variable and lose life method to deal with it and produce the end game .

4 : condition for win the game .

sol : make final score which you reach when complete all the blocks

5 : using methods from class 1 without rewriting them .

sol : make an object from class one and make the methods which we used public methods and same for the variables .

6 : make the brick harder

sol : make a variable with value (2) for hits reducing by touching the ball .

Recommendations :

1 : power-ups may appear that can help the player's progress such as

Increasing the paddle width – make more balls – fire ball .

2 : power-down can hinder the player's progress such as

Decreasing the paddle width – one live down.

3 : more than levels

4 : sound effects such as : hits sound – win or lose sound .

Resources :

1 : JavaFX Tutorial (tutorialspoint.com)

2 : java - Too Many Collisions for Breakout Game with JavaFX - Stack Overflow

3 : Claude-instant - Poe

4 : Breakout-Game/Paddle.java at master · benjamin-m-hodgson/Breakout-Game · GitHub

5 : JavaFX - YouTube