Even Semester (2023)

BINUS UNIVERSITY

BINUS INTERNATIONAL

Final Project Cover Letter

(Individual Work)

Student Information:

Surname: Huang **Given Name:** Jessica **Student ID:** 2602213031

Course Code : COMP6699001 **Course Name :** Object Oriented Programming

Class : L2BC **Lecturer :** Jude Joseph Lamug Martinez, MCS

Type of Assignment : Final Project Report

Jessica Angela Huang - 2602213031

Submission Pattern

Due Date **:** 16 June 2023 **Submission Date :** 16 June 2023

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer's instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating, and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity, and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

Declaration of Originality

By signing this assignment, I understand, accept, and consent to BiNus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student:

Jess

Jessica Angela Huang

Table of Contents

Jessica Angela Huang - 2602213031

A. Description

I. Introduction

BrickBreaker is a classic arcade game that has captivated players for decades. In this report, we will explore the application in developing a BrickBreaker game. The purpose of this report is to analyze the OOP principles and design patterns utilized in the development of BrickBreaker, highlighting their role in creating a modular and extensible game structure. I chose this program because I want to have a game that can kill a bit of time when I get bored as this game doesn't take too long to play and I can leave it be anytime I want. I came to think of this game when I was browsing something I could do for my OOP project and then Brick breaker caught my eye. I remember I used to play this game as a kid to kill time, and back then it was my favorite game to kill time.

Here are the links needed for the program.

Github Link: https://github.com/JessicaAngelah/OOP-FP.git

Youtube Demo: https://www.youtube.com/watch?v=2B81xjFUTNs

II. The function of the program

In this report on BrickBreaker, the focus is on analyzing the different functions and responsibilities of the classes and objects involved in the game. The function of Brick Breaker is to provide an engaging and addictive gaming experience where players aim to break all the bricks on the screen using a paddle and a bouncing ball. Its simple yet challenging gameplay has made it a popular and enduring classic in the world of video games.

B. Design

I. Requirements of the program

- Swing: It is a GUI (Graphical User Interface) framework in Java that provides a set of components and classes. It is required here for creating interactive and visually appealing applications.

II. Function of each part of the program

In this program I used a window with the size of 700x600. The program starts out with the starting screen with bricks, ball and the paddle. The controls of the programs is:

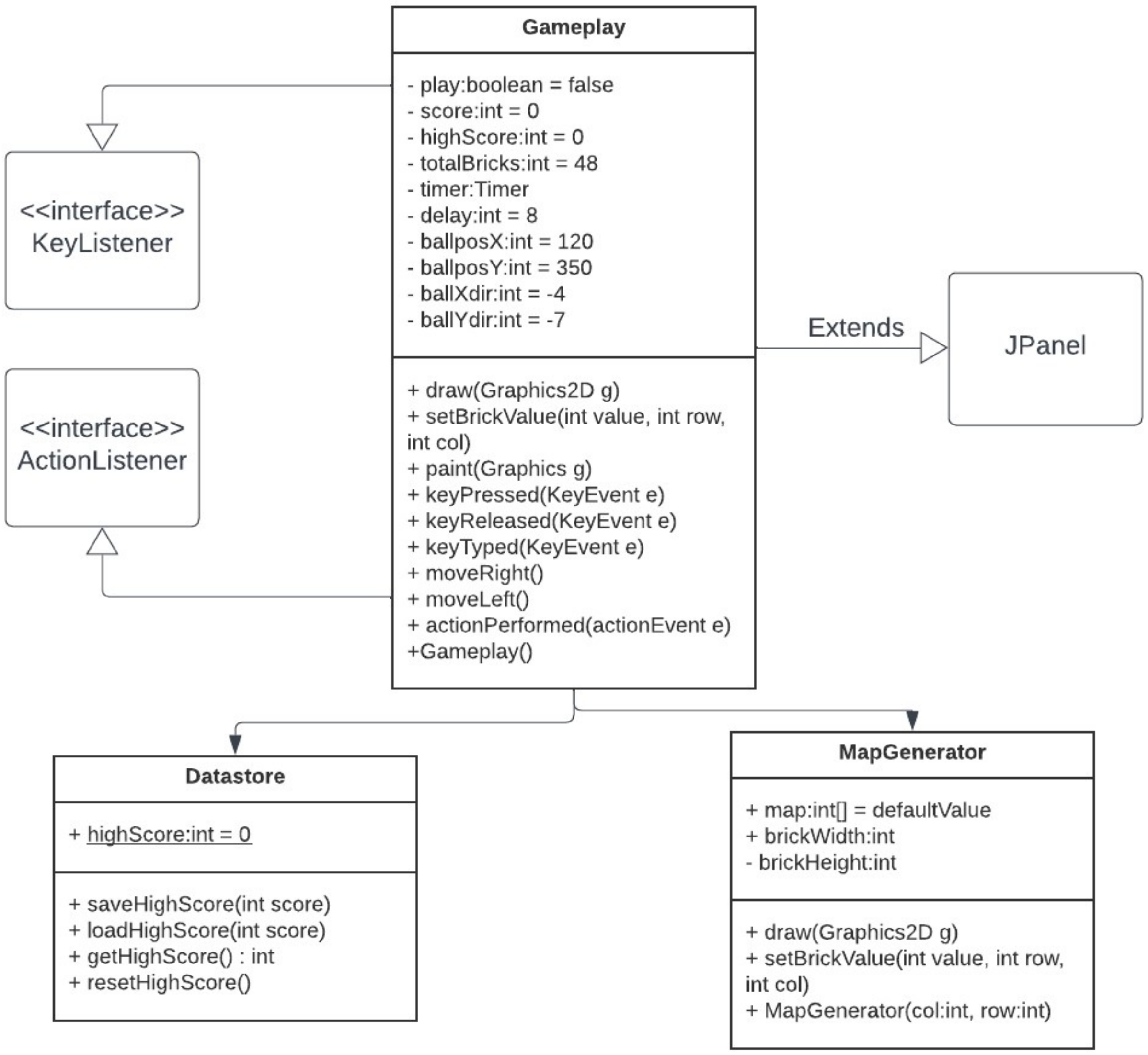
- Left arrow: moves the paddle to the left
- Right arrow: moves the paddle to the right
- Enter button: restarts the game
- ‘ E ‘ button: resets the high score

More explanation of the function code is in the source code comments in my Github repository.

Brickbreaker is a popular and timeless game, with variations available on different platforms, including arcade machines, home consoles, and mobile devices. Its simplicity and addictive nature have made it a beloved classic in the world of video games. The game starts with a window (a map) that consists of several rows of rectangular shapes (bricks) at the top. At the bottom of the screen, there is a paddle that the player controls. Above the paddle, there is a ball ready to be launched. The game begins by launching the ball from the paddle towards the bricks. The ball moves in a straight line and bounces off the walls and bricks it encounters. The player's goal is to break all the bricks on the screen. To break a brick, the player needs to hit it with the ball. When the ball hits a brick, the brick disappears, and the player earns points. The player controls the paddle horizontally, using the left and right arrow keys. The paddle's movement helps the player position it under the ball to bounce it back up towards the bricks. When the ball hits the paddle, it bounces back up towards the bricks. The angle at which the ball bounces depends on the position where it hits the paddle. By controlling the paddle's position, the player can influence the ball's trajectory and aim it towards desired targets. As the game progresses, the ball continues to move and bounce around the game screen. The player must keep the ball from reaching the bottom edge of the screen. If the ball touches the bottom edge, the player loses a life. The player earns points for breaking bricks. The score can be based on the number of bricks destroyed. The game continues until the player choose to start over or quit.

C. Implementation

I. UML Diagram



II. File Directories

- ‘Datastore.java’ : This class obtains, resets and stores the high score that the player gets when playing the game.
- ‘Gameplay.java’ : This class represents the gameplay logic of a game. It sets up the game environment, including the paddle, ball, bricks, and score display. It handles user input, such as moving the paddle, restarting the game, and updates the game state based on the ball's movement and collision with bricks and the paddle. It also handles game over and victory conditions, updating the score and high score accordingly.
- ‘Main.java’ : The main class that lets you run your application and initializes the components of the game.
- ‘MapGenerator.java’ : This class sets up a map consisting of bricks for a game and provides functionality to draw and modify the bricks on the map.
- ‘data.txt’ : This file stores the high score that the player get at the end of the round.

III. Dependencies

- import java.io.*; : An import statement that allows you to use various input and output-related classes
- import java.awt.event.*; : An import statement that allows you to access classes and interfaces related to event handling
- import javax.swing.*; : An import statement that imports all the classes and interfaces from the javax.swing package. It allows you to use Swing components and features directly in your code, making it more convenient and concise. The javax.swing package is part of the Java Swing framework, which provides a set of GUI components and tools for building desktop applications. It includes classes for creating windows, buttons, panels, menus, dialog boxes, and other graphical user interface elements. Imports more advanced Swing GUI components,
- import java.awt.*; : An import statement imports all the classes, interfaces, and enums from the java.awt package. Imports the basic AWT GUI components.
- import javax.swing.Timer; : It imports the Timer class from the javax.swing package in Java. It is used here to perform tasks repeatedly and after a specific delay.
- import javax.swing.JFrame; It imports the JFrame class from the javax.swing package in Java. The JFrame class is a fundamental class in the Swing framework and represents a top-level window or frame in a GUI

application. It provides the basic structure for creating and managing windows with various components.

- import java.awt.BasicStroke; : It imports the BasicStroke class from the java.awt package in Java. The BasicStroke class is a part of the Abstract Window Toolkit (AWT) and represents a basic stroke used for drawing lines and shapes.
- import java.awt.Color; : It imports the Color class from the java.awt package in Java. The Color class is a part of the Abstract Window Toolkit (AWT) and represents a color in the RGB (Red, Green, Blue) color model. It provides a set of predefined color constants and methods for creating custom colors. Example: Color.magenta, Color.pink
- import java.awt.Graphics2D; :It allows access to all the classes, interfaces, and enums from the java.awt package without specifying each one individually. It allows you to utilize the GUI components and perform graphics operations directly in your code.

IV. Extensibility

1. UML (Unified Modeling Language) diagrams are used to visually represent and communicate the structure, behavior, and relationships of systems in a clear and concise manner.
2. Comment lines are used to give further explanations for the functions and methods to help other programmers to understand.

D. Evaluation

I. Does the program work properly

The program does work properly as I followed a video tutorial on youtube, which really helped on doing this project and made me learn a lot of java functions that I didn't understand before. In addition, I changed the speed of the ball and other variables, as well as adding more functions like data store, so that I have a file system to fulfill the requirements. Then I also add the high score as well as the function to reset the high score. I also fixed a few errors on the codes which I find very painful, as you have to check and find line after line what went wrong. All in all, this program works well as it reaches the purpose intended in the first place.

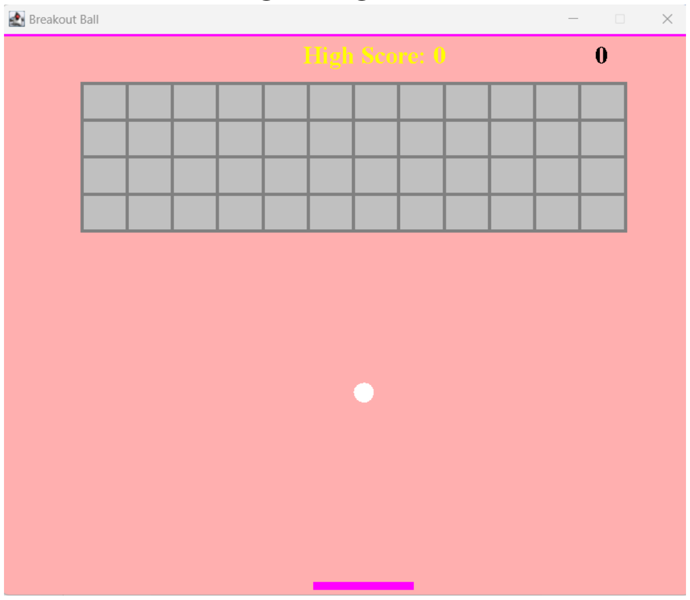
II. Future improvements that can be done

The improvements I would like to add to this game would be adding more levels and the ability to pause the game. I would also like to add power ups like how a brick breaker game would be having, as well as background musics and sound effects. The other improvements aren't necessarily that hard, but I will just need to sit down and do it.

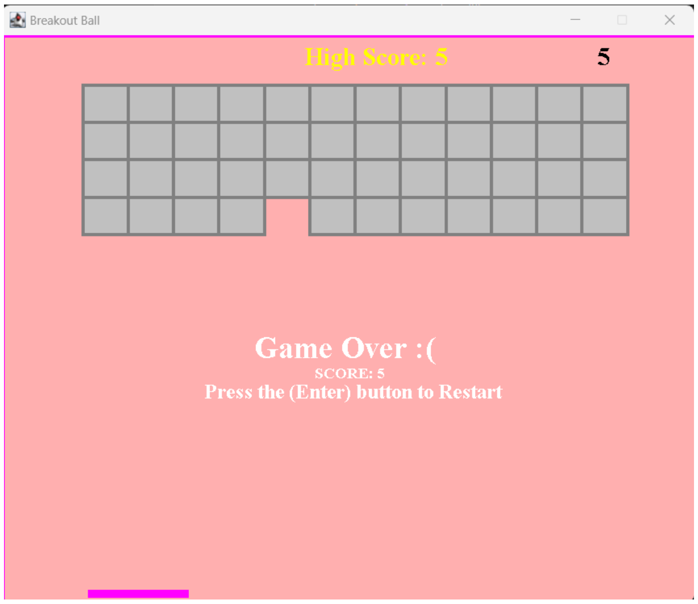
Jessica Angela Huang - 2602213031

E. Evidence of a working program

Beginning Screen



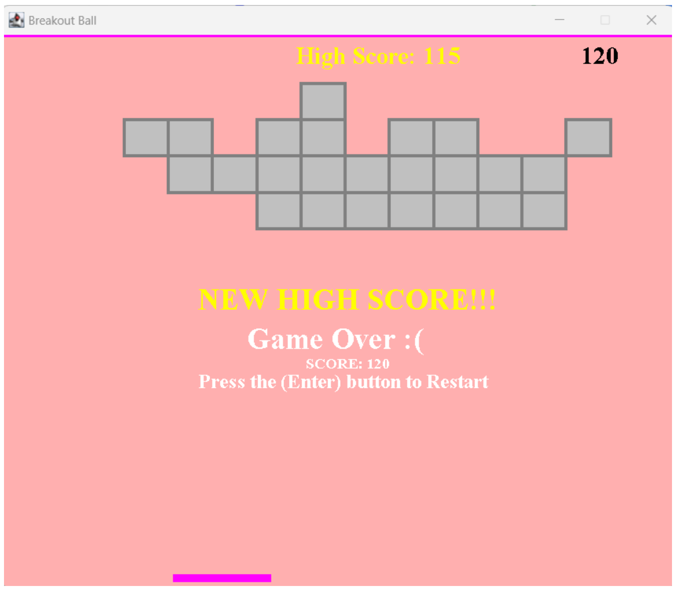
Game Over Screen



Game Over :(
SCORE: 5
Press the (Enter) button to Restart

Jessica Angela Huang - 2602213031

New High Score



High Score: 115

120

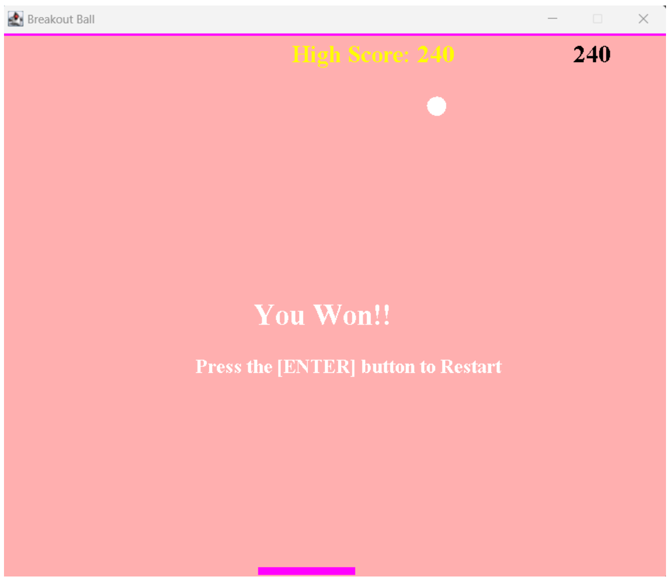
NEW HIGH SCORE!!!

Game Over :(

SCORE: 120

Press the (Enter) button to Restart

You Won



High Score: 240

240

You Won!!

Press the [ENTER] button to Restart