Need for Spear

Syntax Terminators

Emir Fatih Ayyıldız

Tunahan Öztürk

Table of Contents

Vision

Introduction:

Need for spear is a block breaker game. The objective is to clear all blocks as quickly as possible to get the highest score. Player needs to watch out for the remaining lives while Ymir makes it harder for the player. Levels can be harder by adding more blocks into the game with the players desire.

Positioning:

Business Opportunity:

As a popular video game genre we find the existing block breaker games lacking new implementations. We will create a game where the customers enjoy the addition of Ymir into the game. This game will be fun to play with friends and family with the competitive system.

Problem Statement:

Existing block breaking games lack style and fun. We will solve this problem by implementing a challenging enemy named Ymir and a competitive mode where players will race to get the highest score.

Stakeholder Descriptions:

Market Demographics:

Our main target group is male gamers between 13-30 because we have seen the block breaker genre. However we do not wish to limit our demographics to this as the game will be for everyone.

User-Level Goals:

Player: Play a single player game with the addition of Ymir. Build their own levels.

Product Overview:

Product Perspective:

The game will be playable on Windows and Apple computers and run on Java programming language. Internet connection is not required as an offline game

Summary of Benefits:

Story: The player will be immersed into the game with an engaging story

Fun: The game will have features which we believe to be lacking in other games such as an enemy.

Art: The unique artstyle will engage players as our research shows.

Creativity: Players can build their own levels and can come up with interesting combinations.

Assumptions and Dependencies:

We assume that the game is run on a Apple operating system with the latest version of Java.

Cost and Pricing:

The game will cost $5 which is less than its alternatives to build a player base. There are no additional costs on our side other than wages and the cost to keep the game running.

Licensing and Installation:

The customer can download one copy of the game after paying the price.

Teamwork Organization

Emir Fatih Ayyıldız

1. Paddle
2. Obstacles
    2.1. Firm Obstacle
    2.2. Gift Obstacle
3. Building mode

4. Running mode
5. Varius game features
6. Implementing Ymir and Abilities
    6.1. Infinite Void
    6.2. Double Accel

Tunahan Öztürk

1. Obstacles
    1.1. Simple Obstacle
    1.2. Explosive Obstacle
2. Building Mode
3. Running mode
4. Database
    4.1. Save
    4.2. Load
    4.3. Sign-Up
    4.4. Sign-In
5. Implementing Ymir and abilities
    5.1. Hollow Purple

As this work is made with 2 people, many work has been planned and done together. In almost all classes, both of our work can be seen. There has been a good coordination between us and in times where we faced with obstacles, we helped each other out.

Use Cases

Use Case: Authenticating User

Scope: Need for Spear game

Level: user goal

Primary Actor: Player

Stakeholders and Interests:

– Player: Wants to login to their account.

Preconditions: The player just started the game.

Success Guarantee (or Postconditions): Player logs in successfully.

Main Success Scenario:

1. The player enters their username and password.
2. “Login successful” displayed.
3. Player is redirected to menus.
4. After they are done player can choose the logout option or simply exit the game.

Extensions:

1a. Username does not exist.

1. “Wrong Username” displayed.
2. Player can enter another username.

1b. Player creates a new username.

1. Player clicks new user button.
2. Player enters username
3. Player enters password
4. System saves username and password
5. “New user created” displayed.
6. Player redirected to login screen

1c. Player enters wrong password

1. “Wrong Password” displayed.
2. Player can create new password after doing a security check
3. New password is saved to the system
4. Player redirected to login screen

3a. Player tries to start game saved by another player.

1. “Unauthorized load operation” displayed
2. Player redirected to menu

Special Requirements:

- Visible UI on a medium sized screen and a keyboard.

Technology and Data Variations List:

1. Keyboard input is required to enter username.

3. Keyboard input required to navigate the menus

Frequency of Occurrence: Happens once when program is run or player logs out.

Open Issues:

- How will the password security be given?

Use Case: Loading the Game

Scope: Need for Spear game

Level: user goal

Primary Actor: Player

Stakeholders and Interests:

– Player: Wants to load a save file.

Preconditions: The player is in the menu screen.

Success Guarantee (or Postconditions): Player loads a save file and starts the game.

Main Success Scenario:

1. The player chooses “Load game” from the menu screen.
2. The player chooses the file from their computer system.

3. The player waits until message on screen.
4. “Game Loaded” is displayed
5. Player starts the game.

Extensions:

3a. The save file isn’t cannot be reconstructed.

1. “File Corrupted” displays on screen.
2. Player returns to menu screen.

3b. A problem causes to loading operation to take too long.

1. There is a timeout limit. If it passes loading operation is aborted.
2. “File cannot be loaded” displays on screen.
3. Player returns to menu screen.

Special Requirements:

- Visible UI on a medium sized screen and a keyboard and mouse.

Technology and Data Variations List:

1. Keyboard input is required to navigate the menu.
2. File explorer is system dependent.
3. The save file will be a text file that the system can reconstruct a game from.

Frequency of Occurrence: Happens once before game start.

Open Issues:

- How will the save file be read?

- What should the file contain?

- Can we make it so that a save file and a map file are essentially the same?

Use Case: Starting a Game

Scope: Need for Spear game

Level: user goal

Primary Actor: Player

Stakeholders and Interests:

– Player: Wants to start a game.

Preconditions: The player has logged in and in the menus.

Success Guarantee (or Postconditions): Player starts a game successfully.

Main Success Scenario:

1. The player chooses “play game” menu item.
2. Player either loads a map file or a save file.
3. Player waits until the game displays.
4. The game starts and the paddle is in initial position.
5. The player uses keyboard input to launch the ball at a desired position.

Extensions:

2a. The file is corrupted.

1. "File Corrupted" displayed.
2. Player returns to menu.

2b. File takes too long to load.

1. After the timeout period is over "could not load file" is displayed.
2. Player returns to Menus.

2c. Player tries to load a save game by another player

1. "Permission denied" displayed.
2. Player returns to menus.

Special Requirements:

- Visible UI on a medium sized screen and a keyboard.

Technology and Data Variations List:

1. Keyboard input is required choose menu item.
2. File explorer is system dependent.

5. Keyboard input required to launch ball

Frequency of Occurrence: Happens once before game start.

Open Issues:

- Can the save game file and map file have the same format?

Use Case: Deflecting the Ball

Scope: Need for Spear game

Level: user goal

Primary Actor: Player

Stakeholders and Interests:

– Player: Wants to move the stake respectively to hit the ball

Preconditions: The game is running and live is more than 0.

Success Guarantee (or Postconditions): ball changes direction respectively.

Main Success Scenario:

1. stick moves respectively.
2. Ball changes direction depending on the slope of the stick.
3. Ball moves upwards.

Extensions:

1a. the ball doesn't match with the position of the stick

Special Requirements:

- Visible UI on a medium sized screen and a keyboard.

Technology and Data Variations List:

1. Keyboard input is required to move the paddle.
2. Slope of stake changes with keyboard buttons

Frequency of Occurrence: every time ball comes to bottom of the map

Open Issues:

How to change the slope of the stake and reflect the ball respectively

Use Case: Ending the Game

Scope: Need for Spear game

Level: game goal

Primary Actor: Player

Stakeholders and Interests:

– Player: Wants to break all barriers

Preconditions: Player pauses game, no lives left, no barriers left to break

Success Guarantee (or Postconditions): game ends

Main Success Scenario:

1. One of preconditions is provided
2. Game stops
3. Score is saved (in case there is a high score table)
4. Shows “game over” text
5. Shows exit button

Extensions:

Special Requirements:

- Visible UI on a medium sized screen and a keyboard.

Technology and Data Variations List:

1. Mouse or keyboard input required to stop the game or select "exit" button

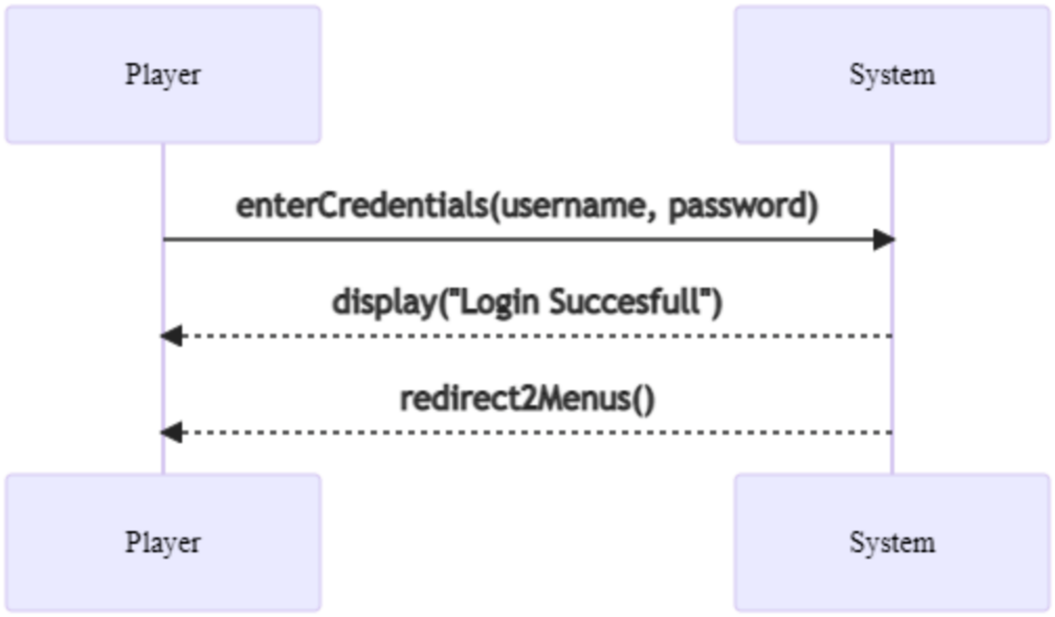
Frequency of Occurrence: every time when one of preconditions is succeeded

Open Issues:

how the player ends the game without saving (if previously saved game ends, will the game be deleted or can be started from the saved files)?

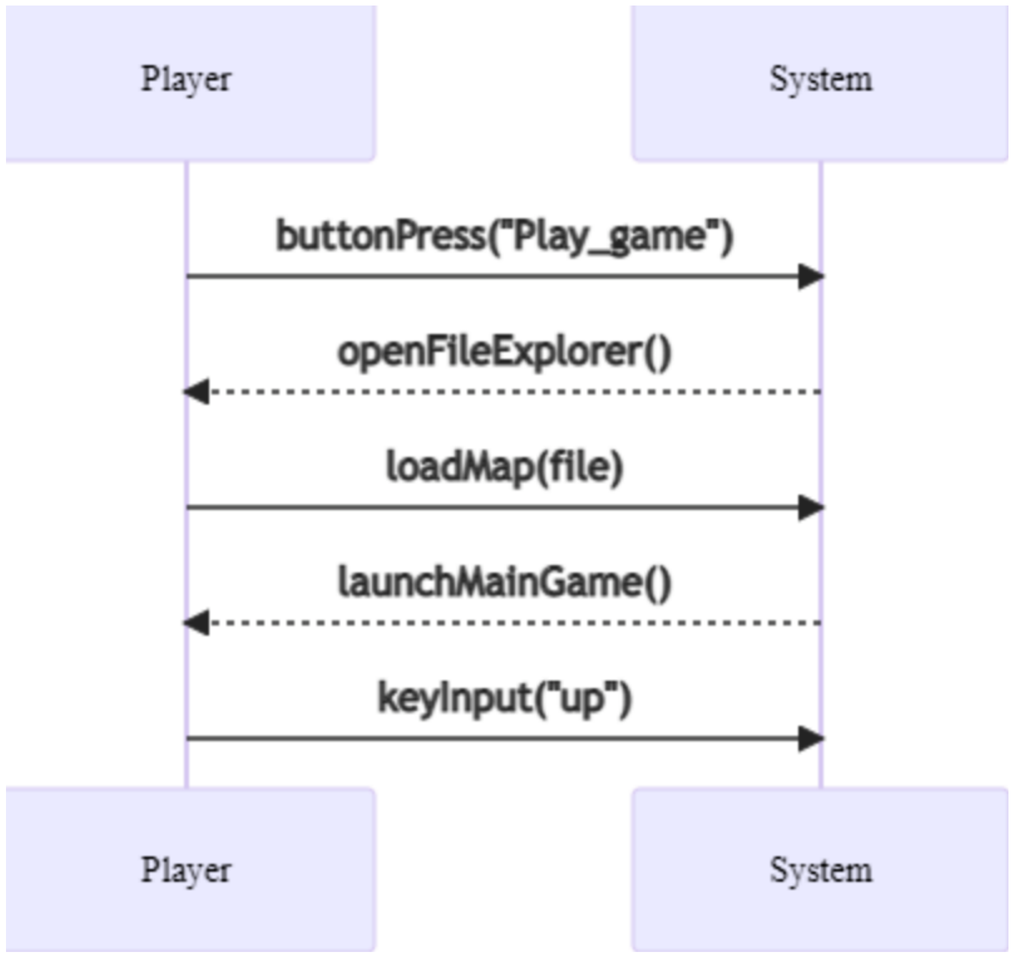
Sequence/Communication Diagrams

SSD for authentication


Player
System
enterCredentials(username, password)
display("Login Succesfull")
redirect2Menus()
Player
System

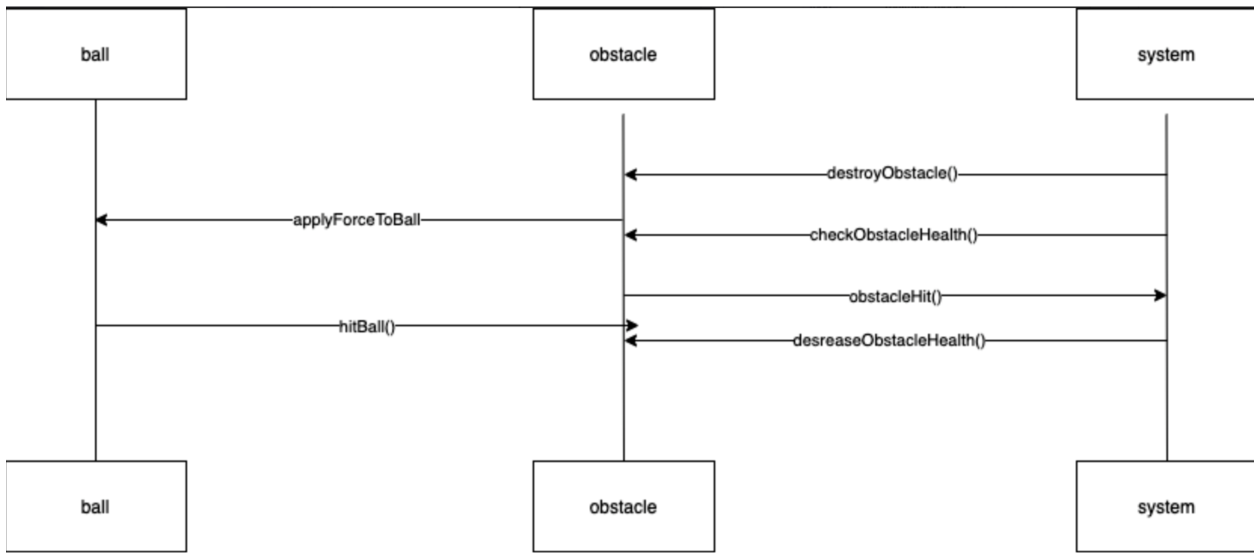
Player enters username and password to start the game. If the player is registered, player is directed to the menu. Else, player can choose to create an account to continue. It is important to use accounts because you can only choose files that you have saved, and nobody else.

SSD for starting the game



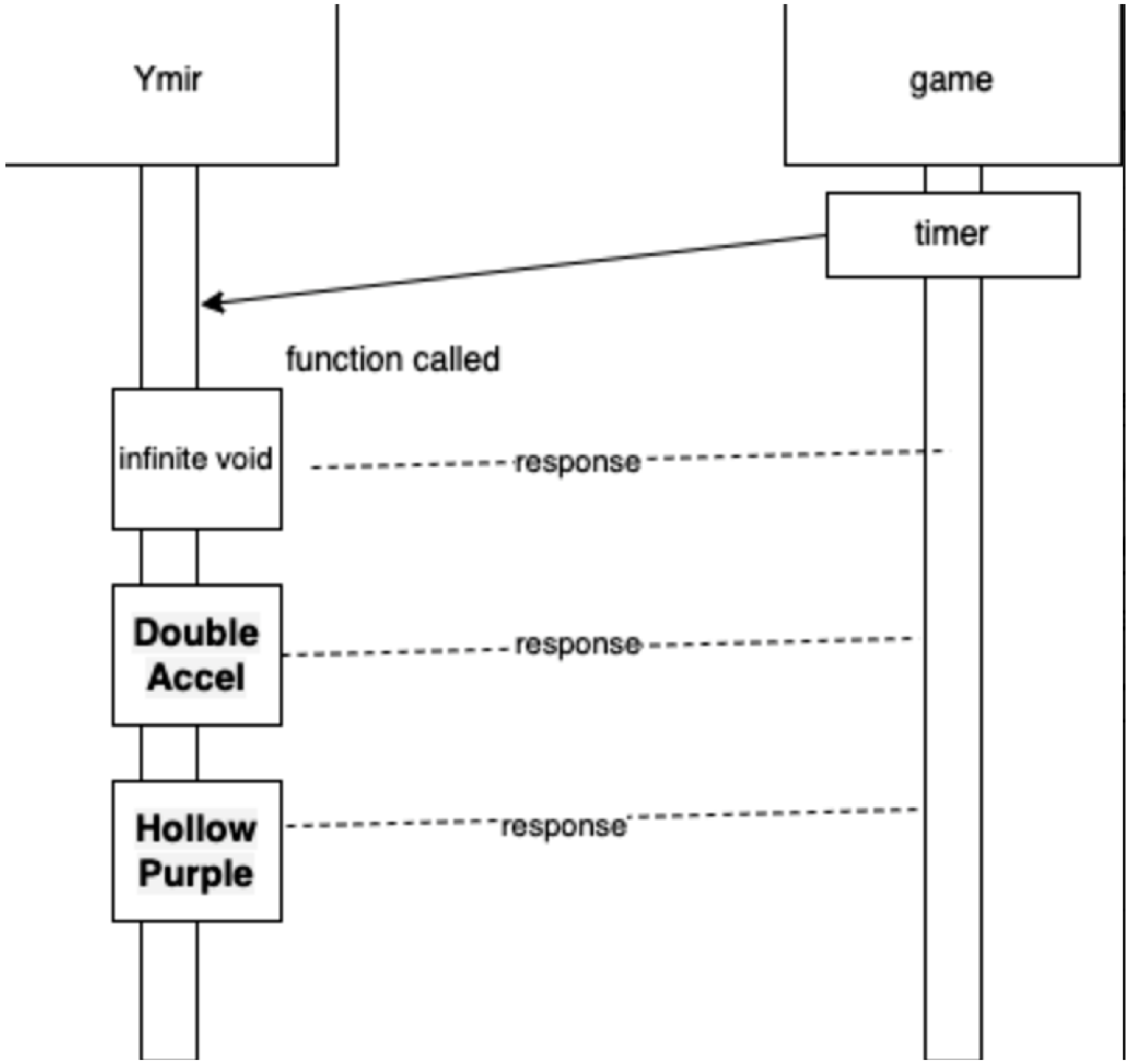
After registering, you can create a new game or load a saved file. the map will load and you can start playing the game by entering a key or clicking the start button

SSD for breaking blocks



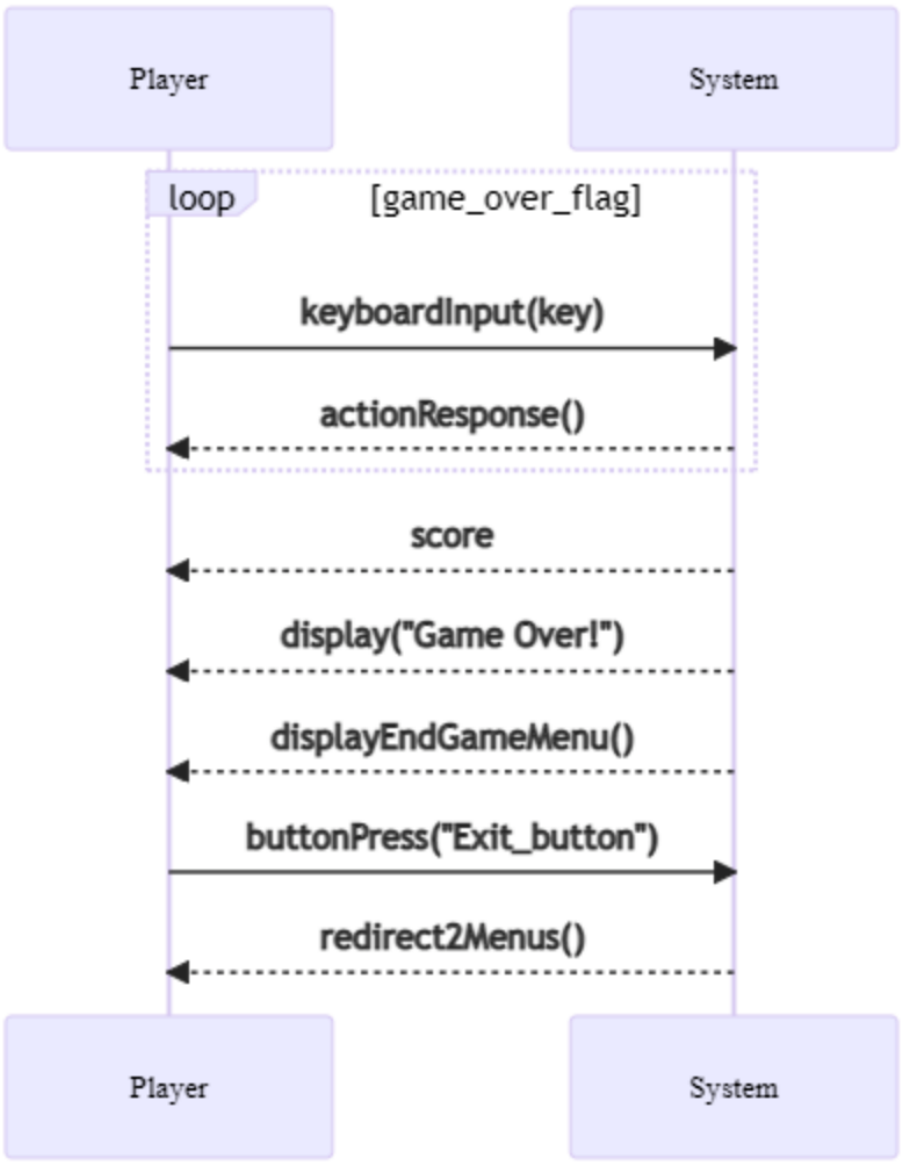
When the ball intersects with obstacles, the game checks the conditions to redirect the ball. Obstacle health is checked and it is determined if the obstacle is destroyed, reduced health or drops a drop.

SSD for Ymir



The game has a timer where in every 30 seconds, one of the abilities is used with ¼ probability. When the ability is chosen, the rule is applied for 15 seconds. Actions of Ymir can be stopped by destroying the Ymir obstacle.

SSD for game over



When no lives left or all the obstacles are destroyed, the score and game over texts show up on the screen.

Operation Contracts

Operation:	getChildren.obstacles.get(i).setFrozen(true)
Cross Reference:	use case Infinite Void
Precondition:	timer hits 30 seconds and Ymir chose function Infinite Void
Postconditions:	some obstacles are frozen for 15 seconds so they don’t lose health

Operation:	enhancedSphere.getShape.setSpeedx/y(half)
Cross Reference:	use case double accent
Precondition:	timer hits 30 seconds and Ymir chose function double accent
Postconditions:	enhanced sphere movement is halved for 15 seconds

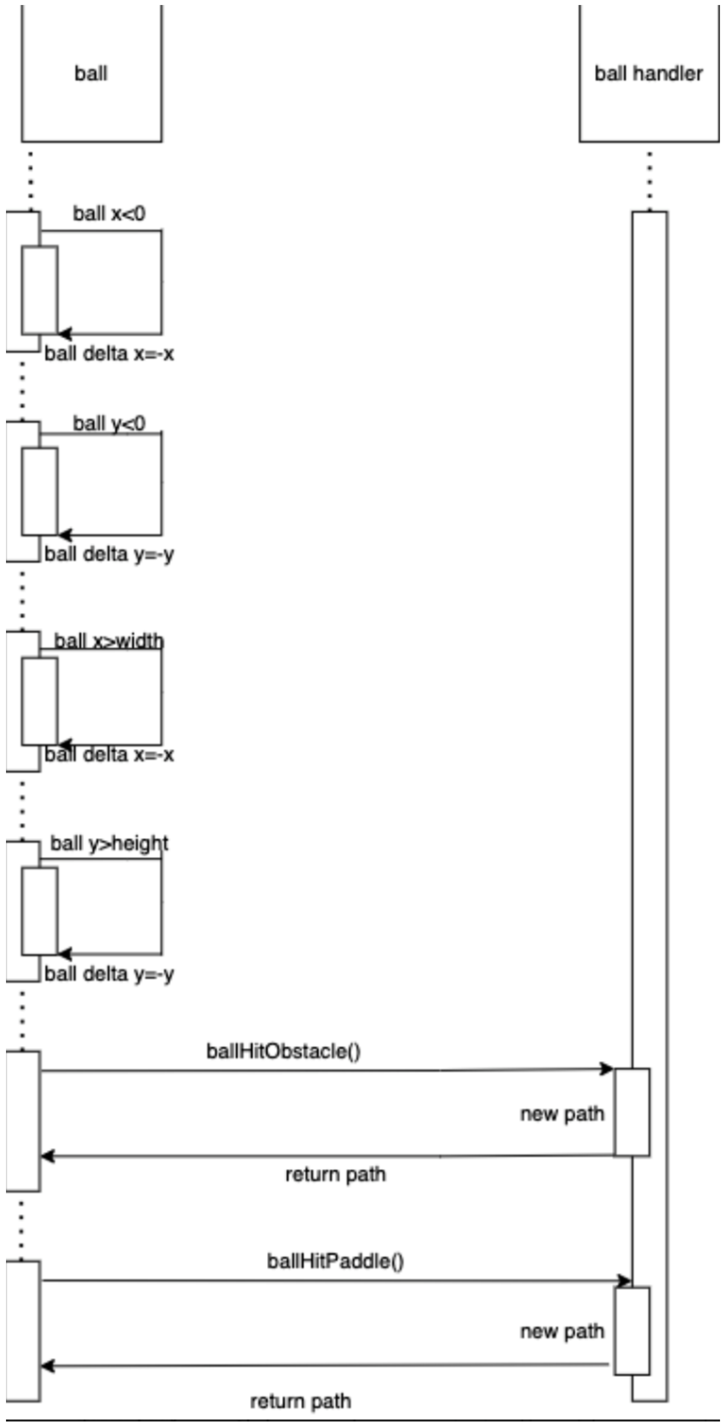
Operation:	getChildren.obstacles.add(simpleObstacle)
Cross Reference:	use case Hollow Purple
Precondition:	timer hits 30 seconds and Ymir chose function Hollow Purple
Postconditions:	new purple obstacles that don’t give points when broken

Operation:	destroy()
Cross Reference:	Use case destroy obstacle
Precondition:	Ball hits the obstacle and obstacle health decreased to 0
Postconditions:	Obstacle is destroyed and points awarded to the player

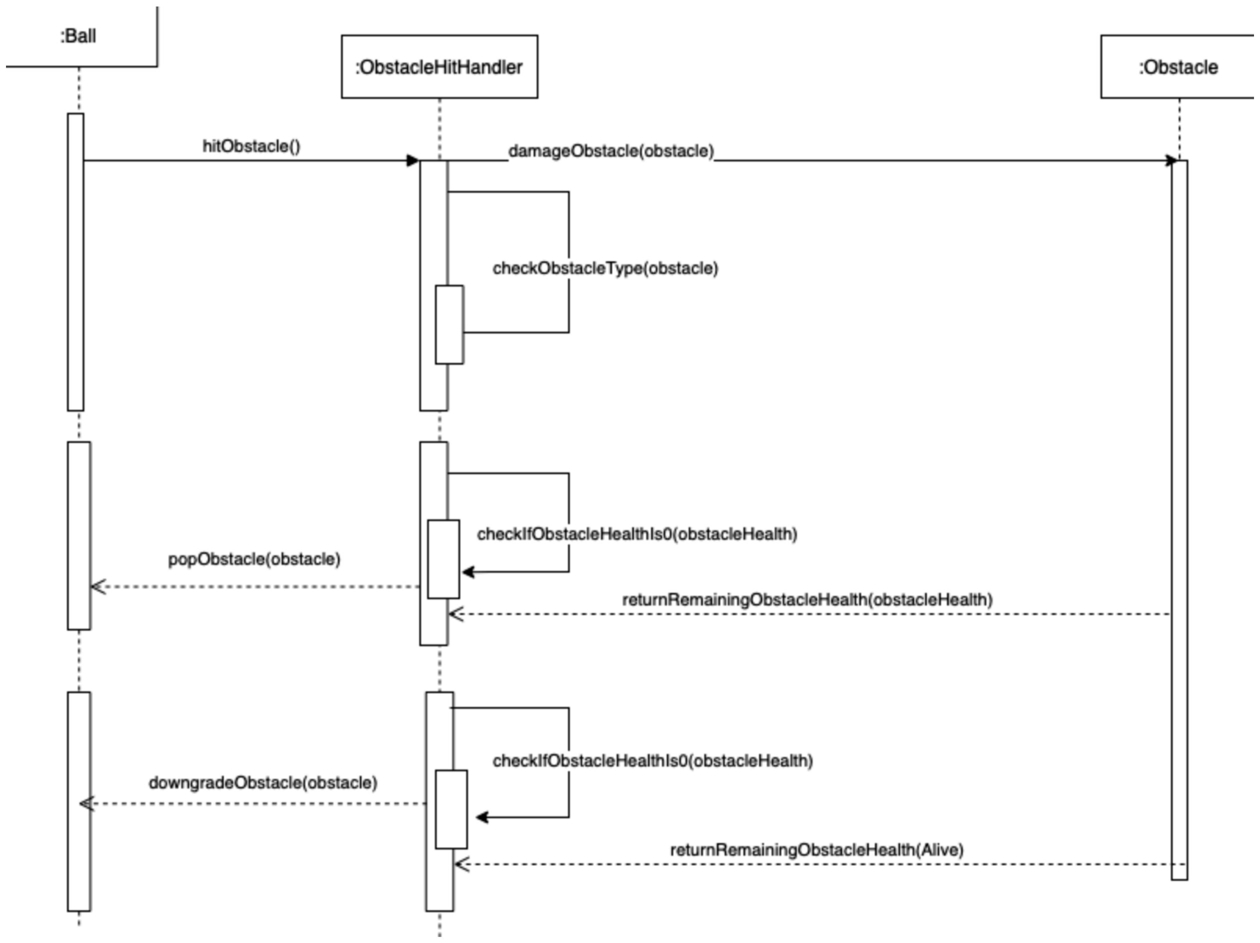
Operation:	melt()
Cross Reference:	Use case Infinite void
Precondition:	15 seconds have passed since Infinite void ability is used
Postconditions:	Obstacles are vulnerable

Interaction Diagrams

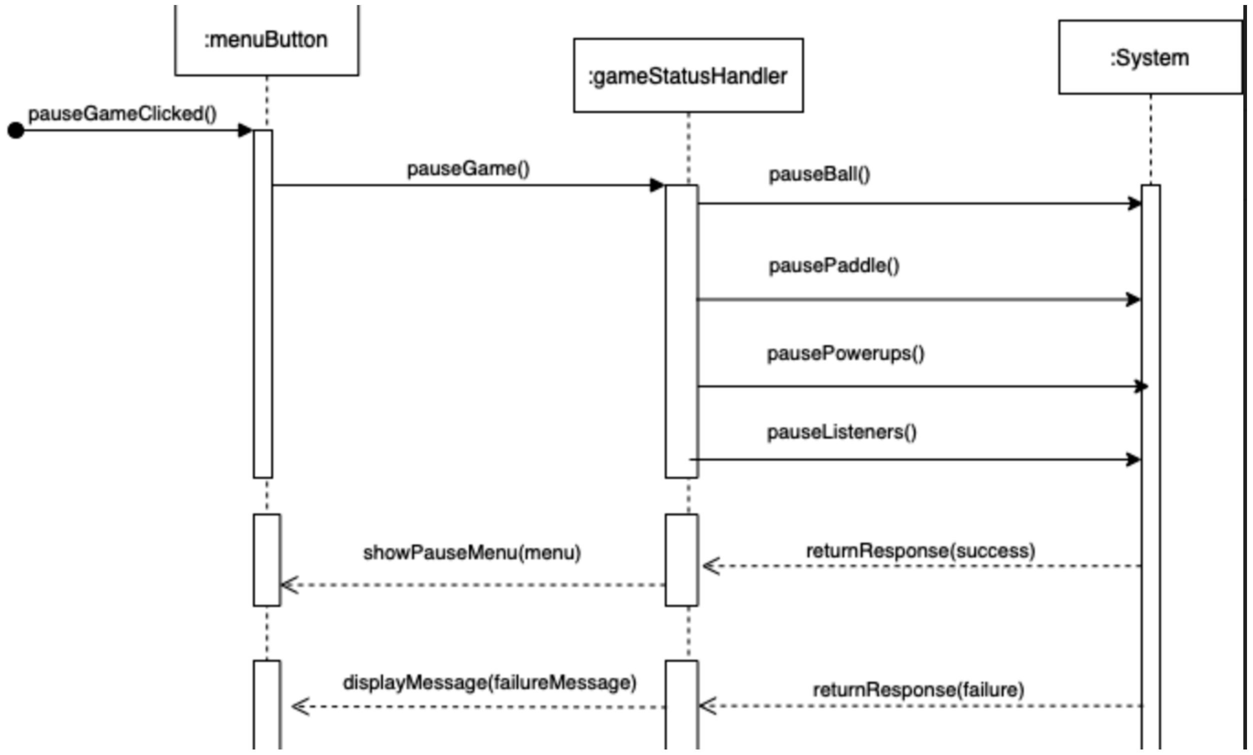
Ball interaction diagram shows how the ball changes directions with the given conditions.



Obstacle interaction diagram shows how the obstacle is handled.

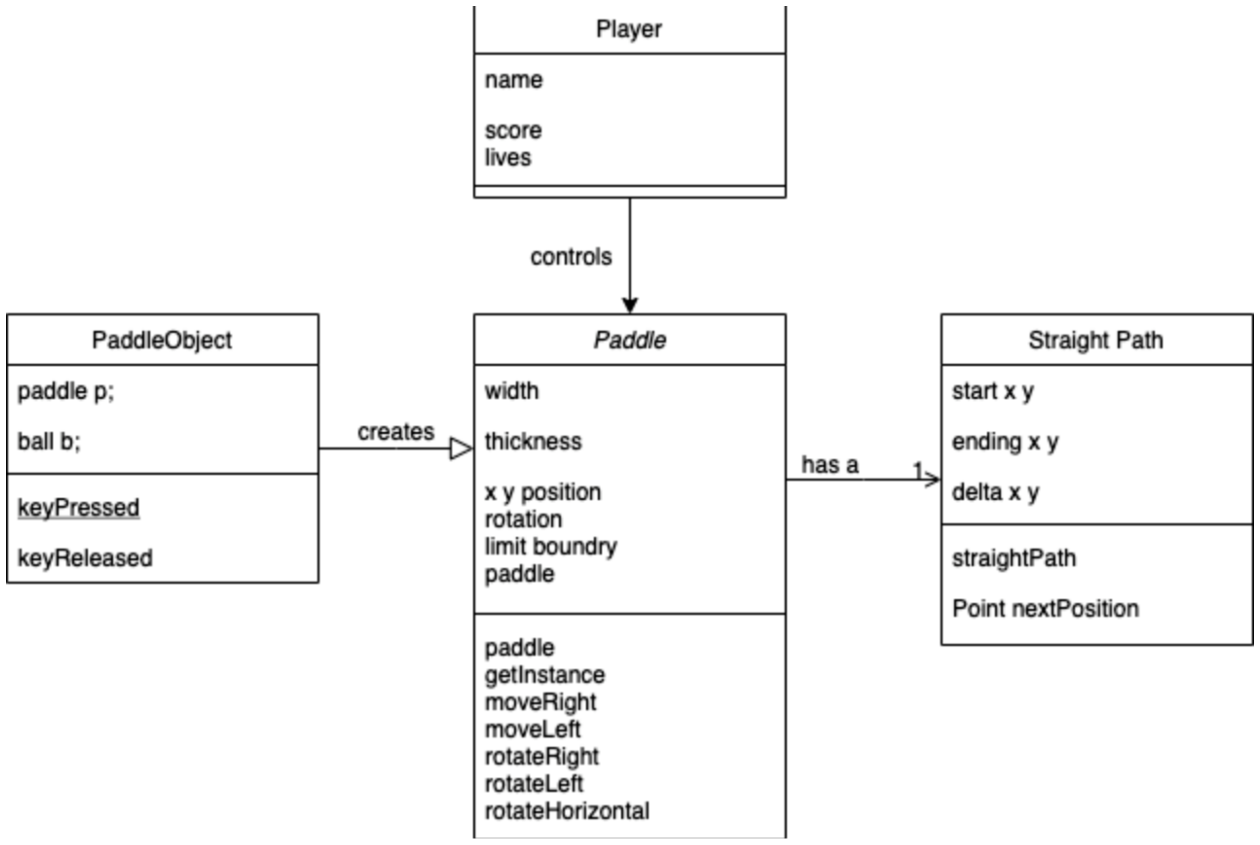

:Ball
:ObstacleHitHandler
:Obstacle
hitObstacle()
damageObstacle(obstacle)
checkObstacleType(obstacle)
checkIfObstacleHealthIs0(obstacleHealth)
popObstacle(obstacle)
returnRemainingObstacleHealth(obstacleHealth)
checkIfObstacleHealthIs0(obstacleHealth)
downgradeObstacle(obstacle)
returnRemainingObstacleHealth(Alive)

Pause interaction diagram shows how the game is paused

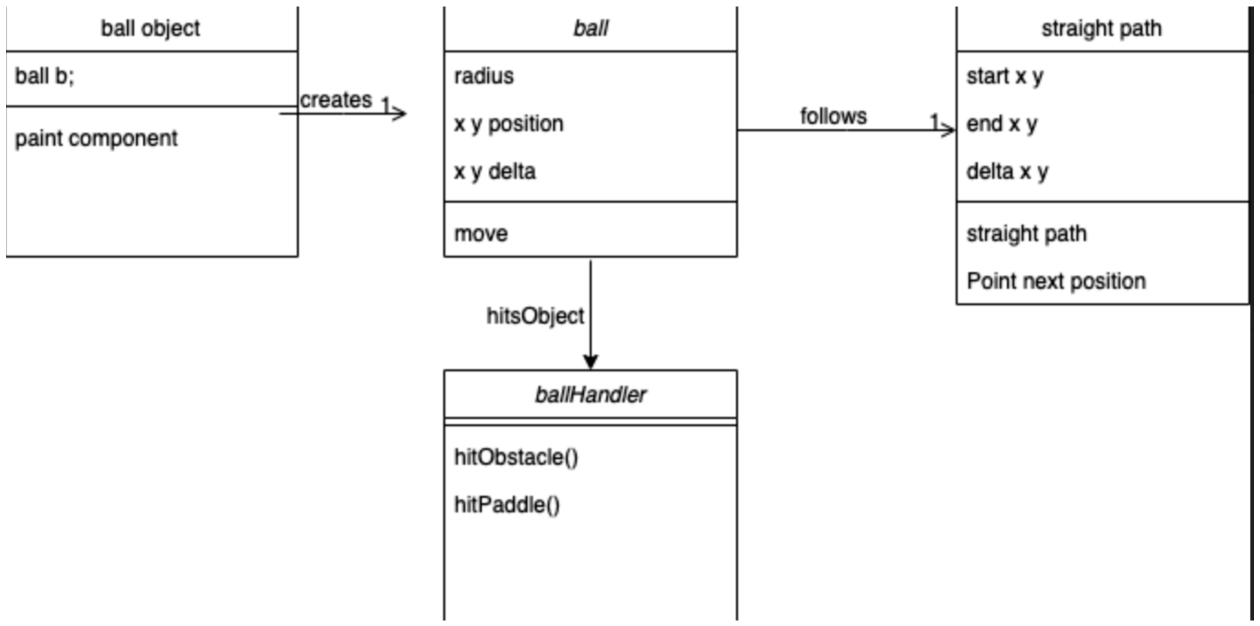


Class Diagrams

Paddle class diagram shows how the paddle is handled. The Paddle object is created using the Paddle class in the Game class with the given specifications. The paddle has some methods to control it and the paddle is moved by using the Straight Path class.



The ball is created by using the ball class in the Game class. The path is pointed by straight path class. Depending on the conditions like hitting the paddle, obstacles or barriers, a new path is created.



Game window (main class) creates a game window to start the game. It has pre-made object specifications. User and the game window is handled in this class.

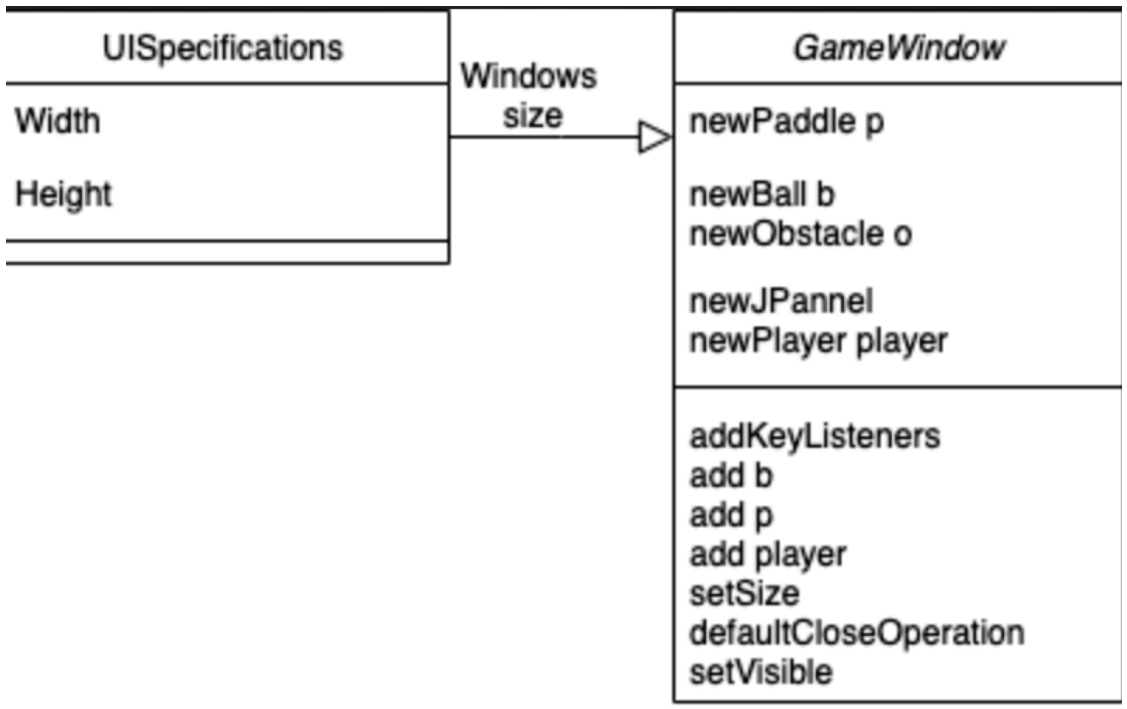
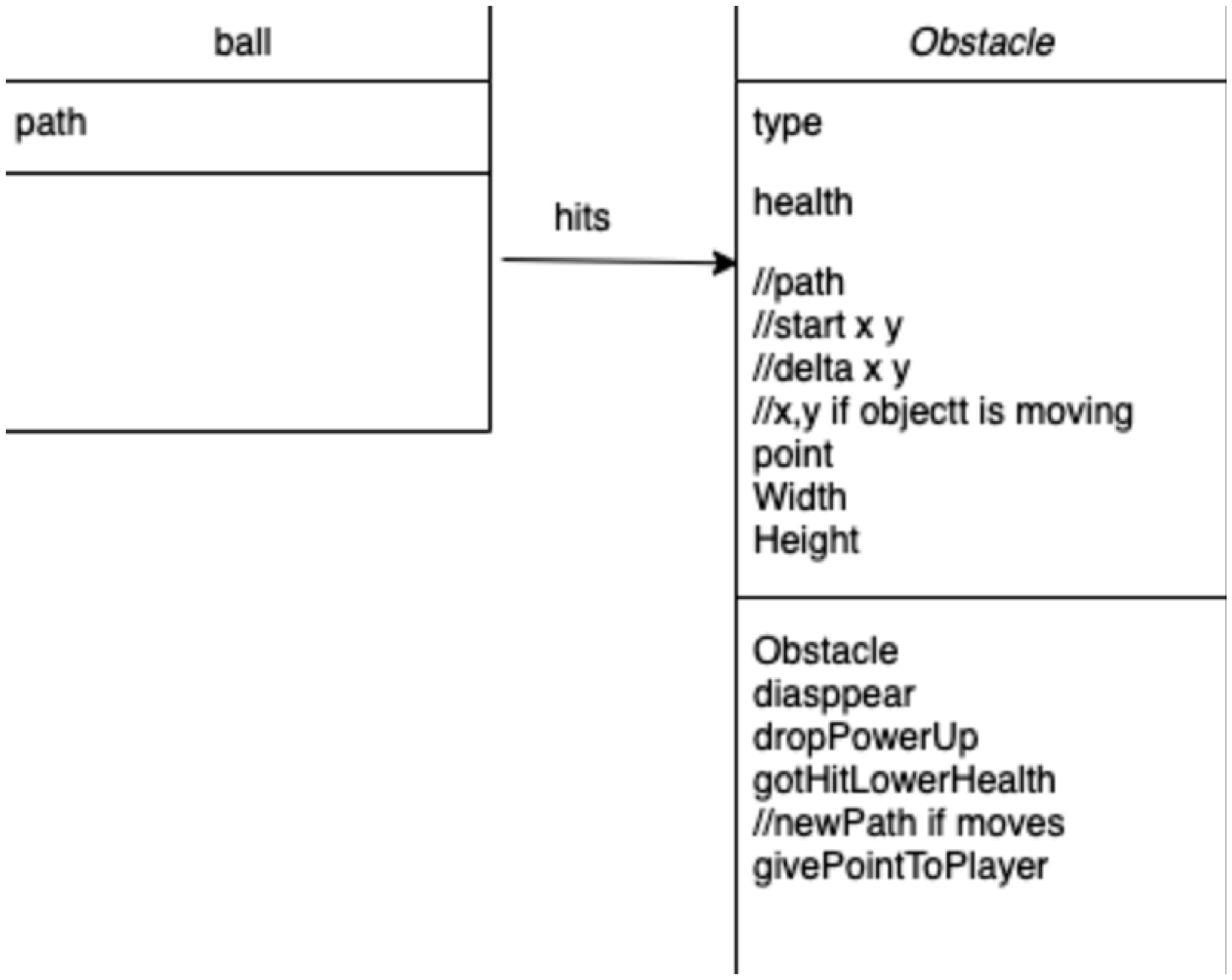
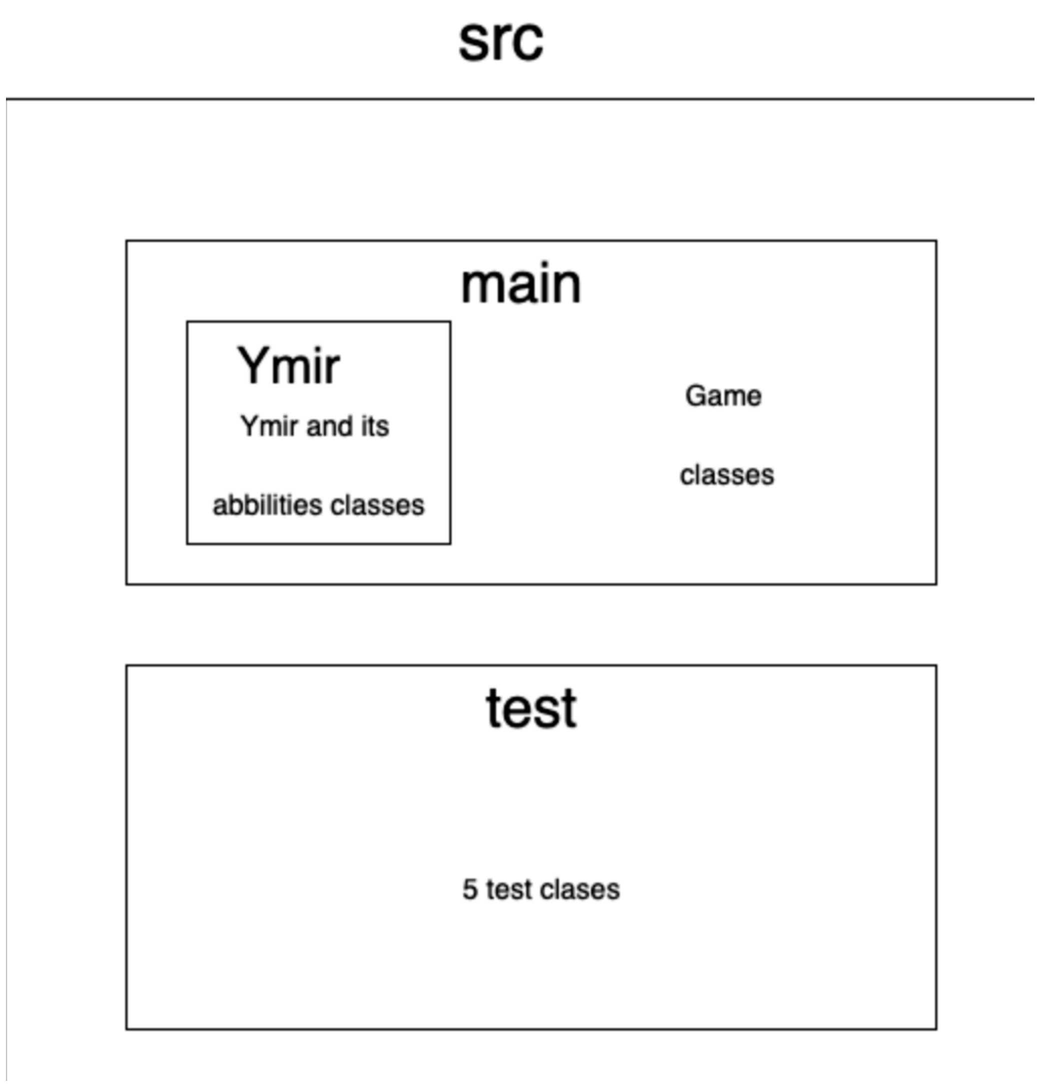


Diagram for the ball hitting the obstacle is shown. Obstacle, ball and straight bath classes are used. In such case, a new path for the ball is created and some changes are made for the obstacle. If the Obstacle has lives left, lives are decreased. Else, obstacle is destroyed and points awarded to the player.



Package Diagram



Design alternatives, design patterns and principles

Singleton- used to create paddle and the ball and get instances of them to redirect the attributes of these objects.

Strategy- used to make Ymir’s abilities

Prototype- used to recreate simple obstacles with purple color, creating obstacles for the Ymir’s hollow purple ability.

Adapter- used to create an interface to implement Ymr’s Strategy pattern

Polymorphism- used to create different types of obstacles

Serializable- used for the user class

Supplementary Specifications

- Login system is used. Players data is stored in a file by using Serializable. The saved game files can only be accessed by the player that created it.
- Tip box can be seen in the menu. It could be adjusted to be more helpful,but the feature is there.
- Least amount of obstacles in a map is 75 simple, 10 firm, 10 gift, 5 explosive and 1 Ymir obstacles. It can be incremented before starting the game with the exception of Ymir's obstacle.
- When the game is paused, you can change the locations of obstacles as you desire by using the mouse.
- While playing the game, Ymir will make it harder for the player. Destroying Ymir's obstacle will stop it.
- Hitting the corner of the obstacle will deflect the ball 180 degrees.
- You can save the game to play it later. Your points and lives will be saved.
- Paddle and the ball are called in class "game" by using singleton.
- Strategy pattern is used for Ymir.
- Polymorphism is used for creating different obstacle types.
- Prototype is used for Ymir's hollow purple ability by recreating simple obstacles.
- Adapter is used to create an interface for Ymir.
- Test classes worked in previous version however after new implementations, they may not work due to changes in the code.

- Frontend is the class called “main” where buttons, stage and text fields are implemented and backend is the class called “game”. All calculations are done in the class “game”.

Glossary

Noble Phantasm

Internally called paddle. Used by player to deflect the ball. Can be angled 45 or -45 degrees by the player (A,D keys) . Can be moved horizontally by the player at its length by second (left, right keys). Length is %10 of screen width. 20 pixels thick.

L: Length of paddle.

Chance

Internally called life. Player has 3 at the start of game. Usually 1 life is lost if the ball reaches bottom of the screen.

Obstacle

The aim of the game is to break them all.

Rectangular Obstacle

This kind of obstacle has size L/5 * 20 pixel

Circular Obstacle

This kind of obstacle are circles with radius of 15 pixels

Simple Obstacle

Takes one hit to break. Rectangular.

Firm Obstacle

Takes multiple hits to be destroyed. Amount displayed on obstacle. Rectangular.

Explosive Obstacle

Falls down when broken. Must be avoided by the paddle or the player loses 1 life. Circular.

Gift Obstacle

Drops a powerup when broken. Rectangular.

Score

Measure of player skill. Starts at 0. Each broken obstacle gives 300/Gametime score.

Chance Giving Ability

Increases life count by 2.

Enchanted Sphere

16x16 size. Moves at 15 pixel per second. Internally called ball. Breaks obstacles when it hits them. Moves in a straight line and reflected at geometrically correct angles when it hits an object or boundary of the screen. If it reaches the bottom it despawns 1 life is lost and player gets a new ball to launch from the paddle. If it hits a corner it is reflected perpendicularly. If it hits a circular object it does not deflect.

Building Mode

Game mode where the player can place objects by mouse to create a game level. There has to be at least 75 simple obstacles, 10 firm obstacles, 5 explosive obstacles, 10 gift obstacles. Can be added more and obstacles can be moved while the game is not moving

Save File

File that contains information about block positions, score and lives. Each player can access their own save file. Can be created any time during gameplay.