User Manual

MidnightRaven12

April 2025

> Given n lines of code, you review it $O(\sqrt{n})$ times
>
> - Anonymous

Contents

§1 Introduction

Welcome to my project!

§1.1 What is Whack-A-Mole for?

Aim training is one of the most important skills that needs to developed in this country. More than the economic crises, and all of the political turmoil. Therefore, we have created Whack-A-Mole in order to address these concerns.

Bad aim can lead to a lot of psychological impacts, and can hinder a child's social life. Ever since the 1980s when video games were invented, there was a higher demand for good aim. Nowadays, the modern lifestyle is to go back home and then to game.

And for many children, not having good aim means that they get left out. An estimated 1 in 3 children will be left out. Additionally an estimated 65% of statistics will be made up by 2040, (and more as we speak), as the Clicker Team, we will provide a solid product on training aim.

§1.2 Mechanics

It's just a basic game where you can, you know, whack some moles. You click on flashing red and yellow lights in order to gain points. Conversely, if you press on Grey Circles, you lose points and lives. There's also powerups, and stuff like that. So, really, just a basic minimalist game that people will play and just aim train on. It's minimalistic design can help ensure your child's concentration.

§1.3 Basic Usage

First, either use git clone, or just copy and paste the entire repository. If you want to run the game, please use the 'scripts/run.ps1' script in order to run the game. (If you want to compile the game, please use the script that is named 'compile', and use 'compile.sh' for Linux Systems.) (cd scripts, and then ./compile.ps1 or ./compile.sh)

This compile script will compile all of your scripts into the (OUTPUT_DIR Directory.) To add more, simply follow the pattern spelled out in the (javacCmd).

It will output in the final_project folder!

§2 Technical Stuff

Here are the more technical things in the program.

§2.1 Classes

Those two important classes are the WhackAMoleGame.class and the Hole.class with the corresponding UML diagram here.

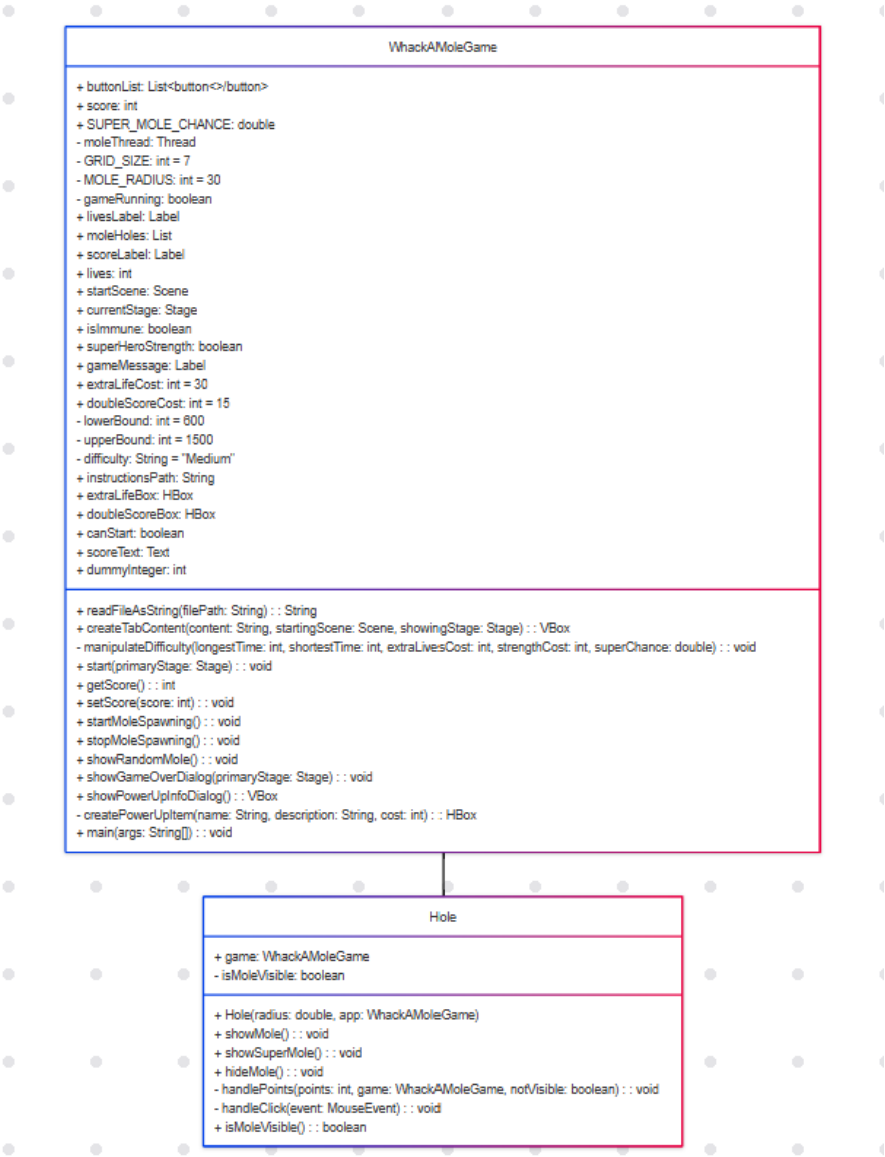


Figure 1: Class Diagram

In particular, the two most important methods are: start, (which starts the game), and startMoleSpawning. Those code snippets are:

```
@Override
public void start(Stage primaryStage) {
    primaryStage.setTitle("Whack-A-Mole");
    currentStage = primaryStage;

    // Start screen      You, 21 hours ago • Here is the final project!

    VBox startLayout = new VBox(20);
    startLayout.setStyle("-fx-alignment: center; -fx-padding: 20px;");
    Label title = new Label(text:"Whack-A-Mole");
    title.setFont(Font.font(32));
    Button instructionsButton = new Button(text:"Instructions");
    instructionsButton.setStyle("-fx-alignment: center; -fx-font-size: 16px;");
    Button difficultyStageButton = new Button(text:"Start Game");
    difficultyStageButton.setStyle("-fx-font-size: 16px;");
    startLayout.getChildren().addAll(title, instructionsButton, difficultyStageButton);
    startScene = new Scene(startLayout, 600, 400);
```

Figure 2: Start Method

and:

```
private void startMoleSpawning() {
    gameRunning = true;

    // Threads to make sure that JavaFX does not collide.
    moleThread = new Thread(() -> {
        while (gameRunning) {
            try {
                Random rand = new Random();
                int sleepTime = rand.nextInt(upperBound - lowerBound) + lowerBound;
                Thread.sleep(sleepTime); // Wait for the specified time
                // Update the UI with mole appearance
                Platform.runLater(this::showRandomMole);

            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    });
    moleThread.start();
}       You, 21 hours ago • Here is the final project! …
```

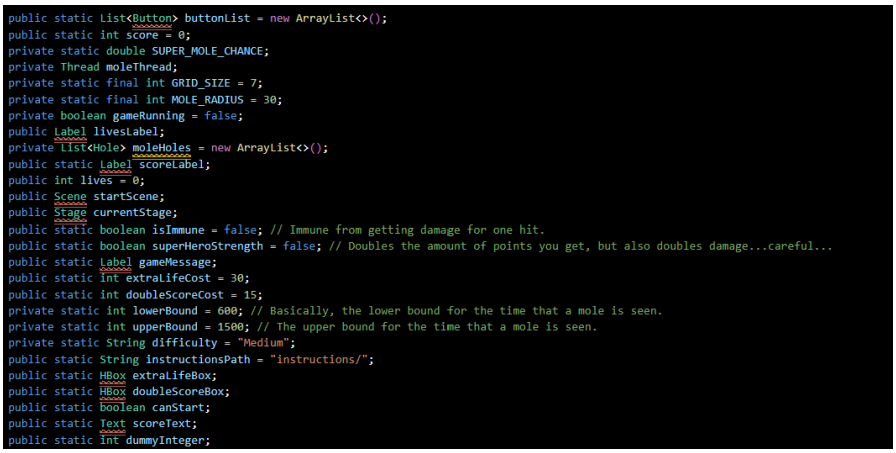
Figure 3: Start Mole Method

The idea of startMoleSpawning is basically just, to start moles, as well as the start scene. Lastly, we also have:

These are variables to change how you play the game.

§2.2 Game Mechanics

There are a few game mechanics, including powerups such as: extraLives, and doubleScore which are referenced above.



```
public static List<Button> buttonList = new ArrayList<>();
public static int score = 0;
private static double SUPER_MOLE_CHANCE;
private Thread moleThread;
private static final int GRID_SIZE = 7;
private static final int MOLE_RADIUS = 30;
private boolean gameRunning = false;
public Label livesLabel;
private List<Hole> moleHoles = new ArrayList<>();
public static Label scoreLabel;
public int lives = 0;
public Scene startScene;
public Stage currentStage;
public static boolean isImmune = false; // Immune from getting damage for one hit.
public static boolean superHeroStrength = false; // Doubles the amount of points you get, but also doubles damage...careful...
public static Label gameMessage;
public static int extraLifeCost = 30;
public static int doubleScoreCost = 15;
private static int lowerBound = 600; // Basically, the lower bound for the time that a mole is seen.
private static int upperBound = 1500; // The upper bound for the time that a mole is seen.
private static String difficulty = "Medium";
public static String instructionsPath = "instructions/";
public static HBox extraLifeBox;
public static HBox doubleScoreBox;
public static boolean canStart;
public static Text scoreText;
public static int dummyInteger;
```

Figure 4: Config Values