**Program Instruction:**

**Name Field**

**Exit Button**

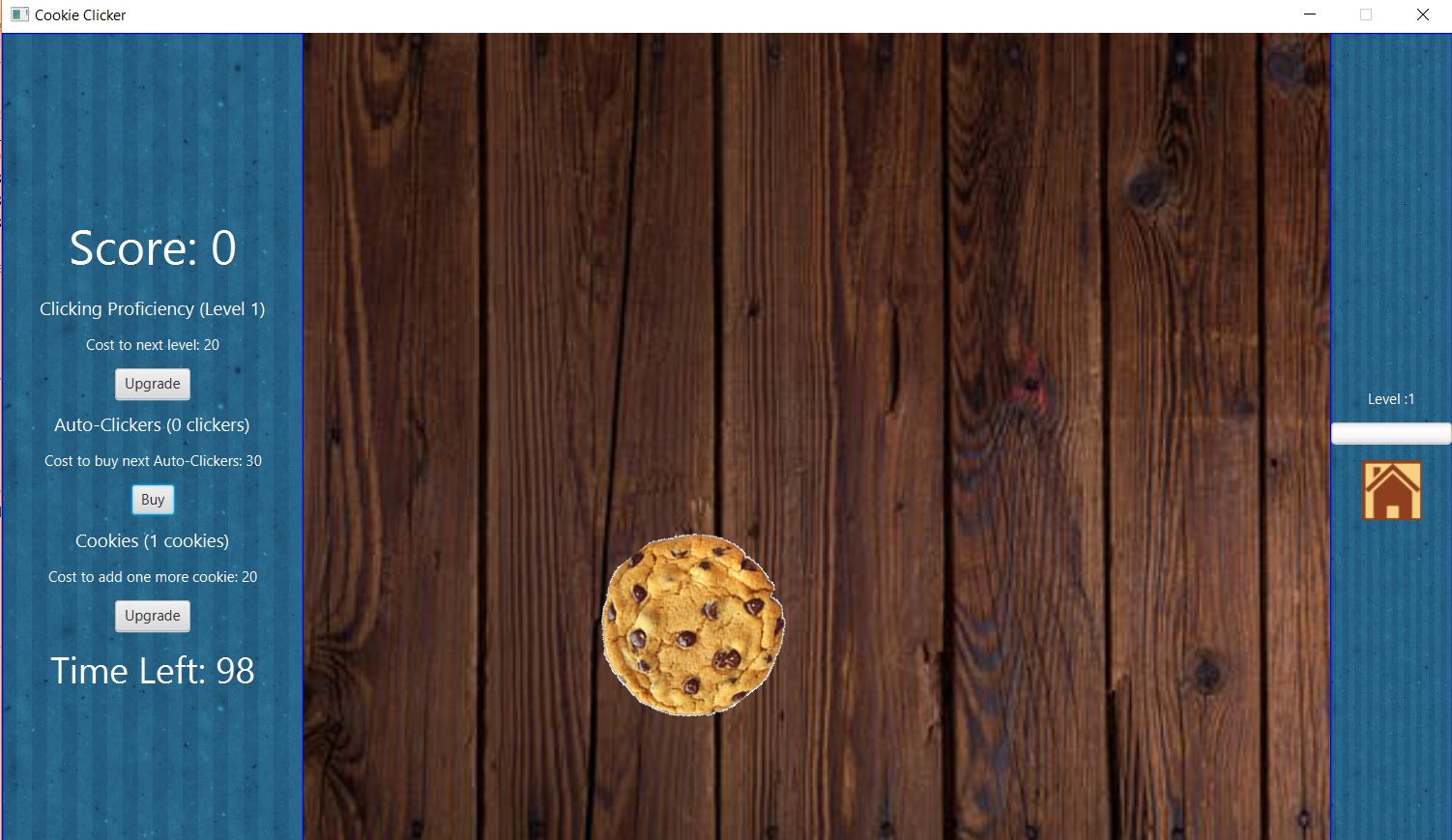
**Play Button**

**High Score Button**

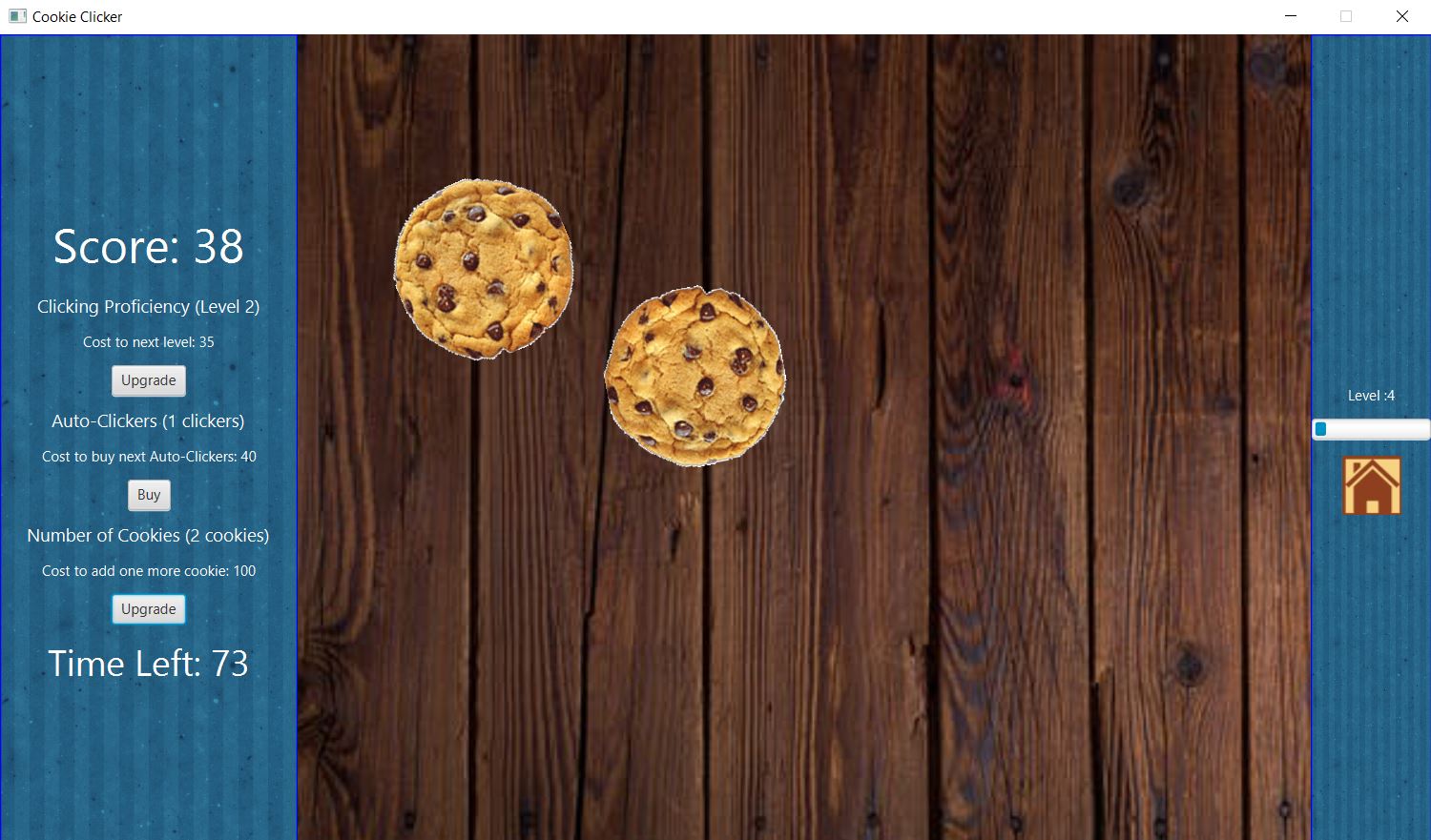


This is the first page when you start the program. It consist of three buttons. The left button is a high score button. If you click it, you will be directed to a high score page. The button in the center is for start playing. Click it and you will be directed to main playing page. The last button on the right is for exit the program. The text filed on the bottom is for entering player’s name.

If this filed is blank, player name will be “Unnamed Player”.



On the main playing page, you have to click the cookie as many time as you can in order to get a score (1 click = 1 point) until the time run out. The cookie will move randomly on the page. Every click you clicked will increase your level which is displayed on the right side. The higher your level is, the faster the cookie move. There is also a home button if you want to return to the home page (your ongoing game will not be saved).



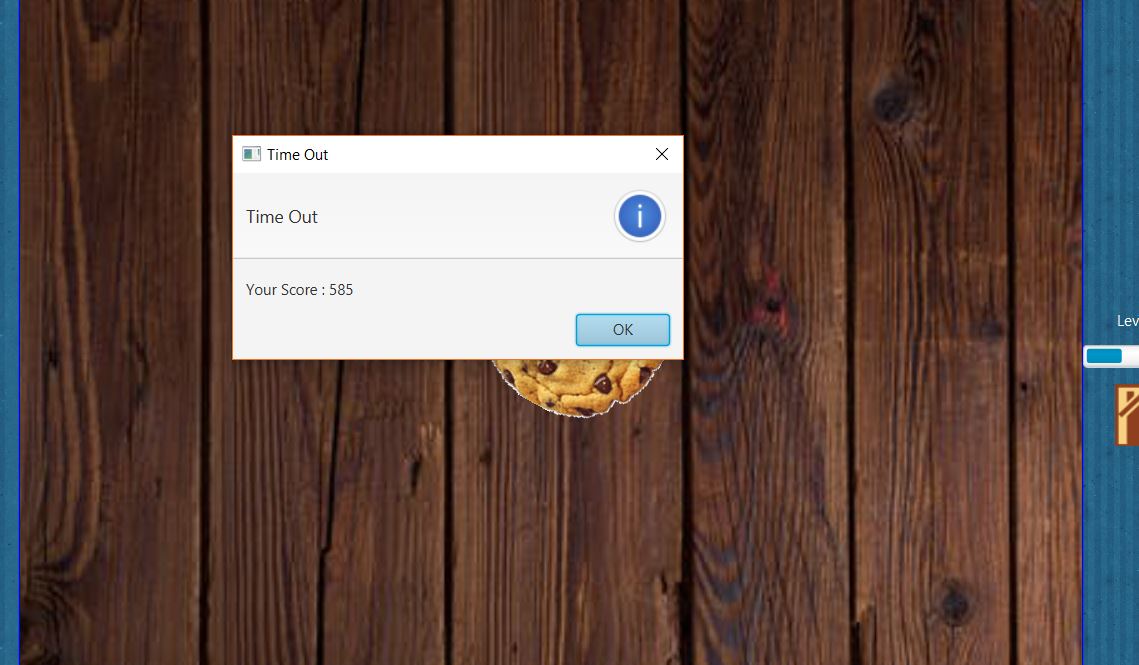
There are three upgrade which is displayed on the left side beneath the score to help you get the score easier. First is “Clicking Proficiency” upgrade. This upgrade will increase the score per click. The more you upgrade this, the more score earned per click.

Next is “Auto Clicker” upgrade. This upgrade will auto increase score every second. The more you upgrade this, the more score earned per second.

Last is “Add Cookie” upgrade. This upgrade will add cookie to the game so you will have more chance clicking the cookies.

In order to upgrade any upgrade, you have to pay using your score and the more you upgrade, the higher the cost is.

When the time run out, there will be a pop up showing your score like this.



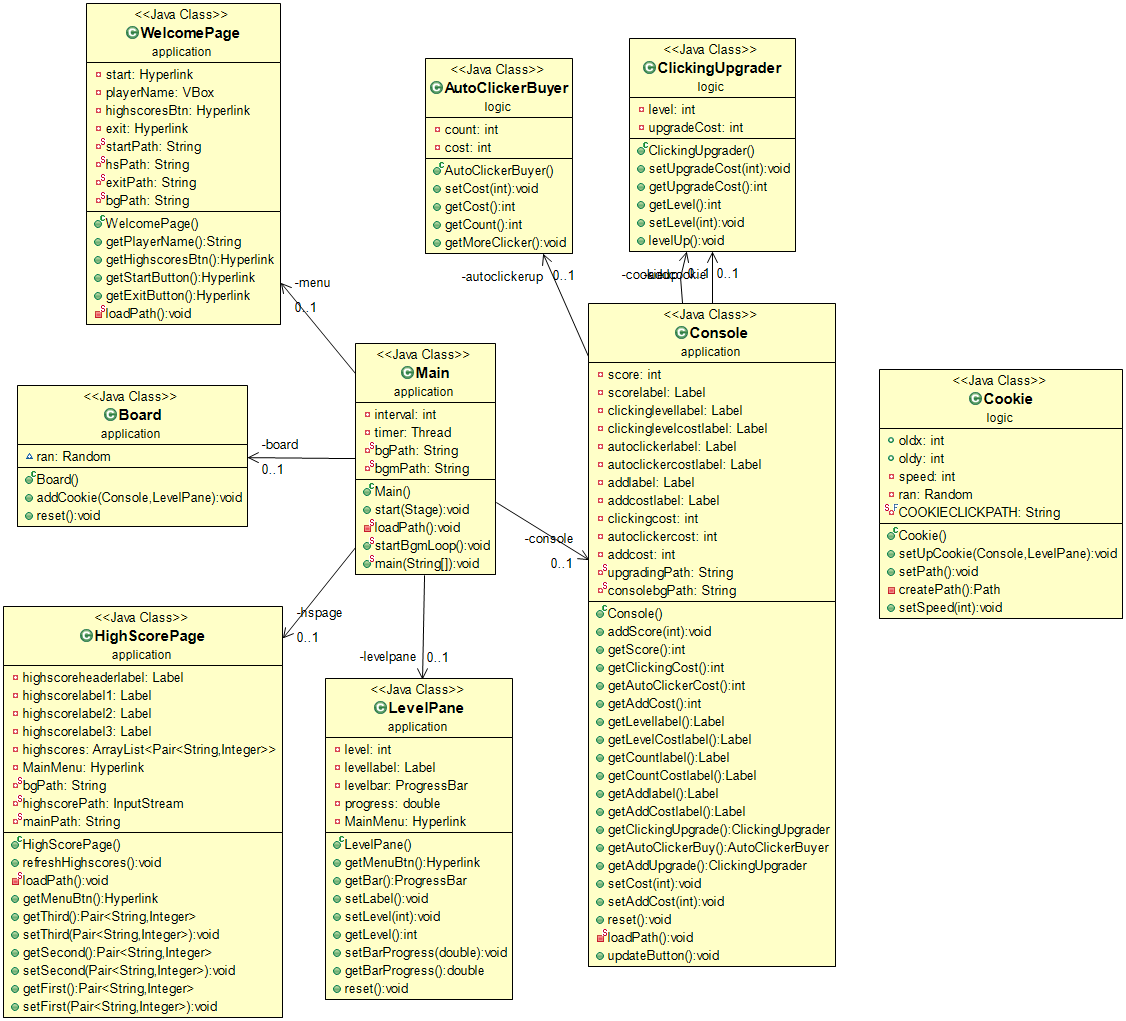
And then you will be directed to a high score page. If your score is high enough your name will automatically appear on the ranking.



There are home button at the bottom that will bring you back to the home page.

**Implementation Details:**

The diagram of the game is illustrated in the picture below. There are 6 classes in application package: **Main**, **Board**, **Console**, **LevelPane**, **HighScorePage**, and **WelcomePage**, and 3 classes in logic package: **Cookie**, **ClickingUpgrader**, and **AutoClickerBuyer**.



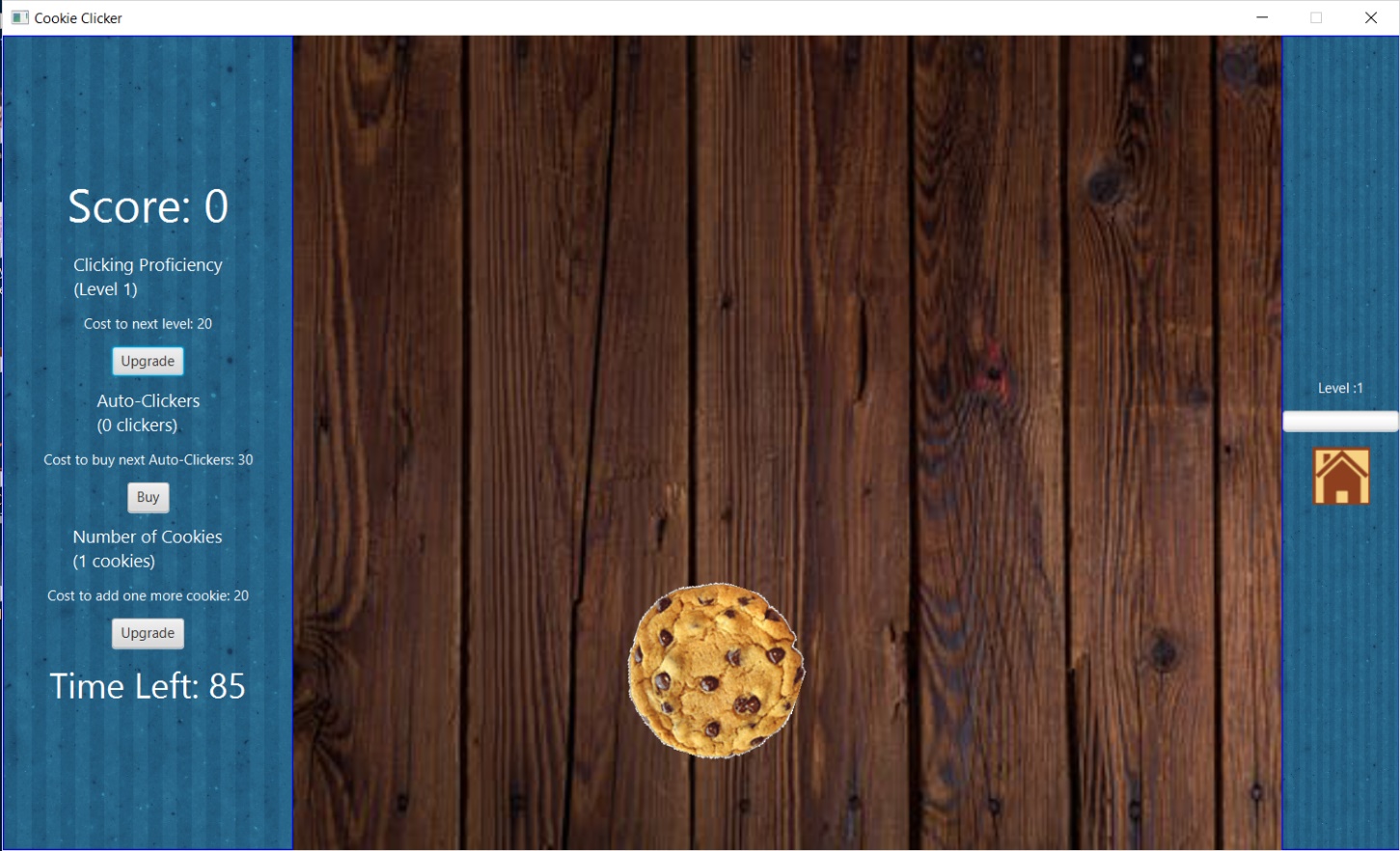
*The UML diagram of the program.*

The structure of GUI can be broken down as in the the pictures below.



**WelcomePage**

*The outline showing the core structure of the game’s main menu.*



**Main (root container)**

**LevelPane**

**Console**

**Board**

*The outline showing the core structure of the game’s in-game.*



**HighscorePage**

*The outline showing the core of the game’s highscore ranking.*

\* Note that Access Modifier Notations can be listed bellowed

+ (**public**)

- (**private**)

# (**protected**)

**static** will be underlined.

**abstract** will be italic.

**1. Class Logic.ClickingUpgrader extends Button**

1.1 Field

|  |  |
| --- | --- |
| - int level | This upgrade’s current level. (Max level is 10) |
| -int upgradeCost | This upgrade’s cost for upgrading to next level. |

1.2 Constructor

|  |  |
| --- | --- |
| + ClickingUpgrader() | Initializes a button with text of “Upgrade”. Sets level to 1. Calling setUpgradeCost(int level) to set upgradeCost. |

1.3 Method

|  |  |
| --- | --- |
| + void setUpgradeCost(int level) | Set upgradeCost depending on the upgrade’s current level.  - Level 1 upgrade’s upgradeCost is 20.  - Level 2 or more upgrade’s upgradeCost is:  (Previous level’s upgradeCost) + ((Current level + 1) \* 5) |
| + void levelUp() | Set current upgrade’s level to next level and current upgrade’s upgradeCost according to its level that has been set. (If current upgrade’s level is 10, return nothing) |
| Getters & Setters | level has its getter and setter. But upgradeCost has only getter. |

**2. Class Logic.AutoClickerBuyer extends Button**

2.1 Field

|  |  |
| --- | --- |
| - int count | The current number of auto-clickers. (The max number is 10) |
| - int cost | The current cost to buy next auto-clicker. |

2.2 Constructor

|  |  |
| --- | --- |
| + AutoClickerBuyer() | Initializes a button with text of “Buy”. Sets count to 0. Calling setCost(int count) to set cost. |

2.3 Method

|  |  |
| --- | --- |
| + void setCost(int count) | Set cost of next auto-clicker depending on the current number of auto-clickers.  - First auto-clicker’s cost is 30.  - Another auto-clicker’s cost is higher than the previous auto-clicker’s cost by 10. |
| + void getMoreClicker() | Get one more auto-clicker. If there are 10 auto-clickers already, return nothing. |
| Getter methods for every fields |  |

**3. Class Logic.Cookie extends ImageView**

3.1 Field

|  |  |
| --- | --- |
| + int oldx | This cookie’s current x coordinate. |
| + int oldy | This cookie’s current y coordinate. |
| - int speed | This cookie’s speed. (Speed is originally set to 2000) |
| - Random ran | An instance that is used to random an integer. |
| - String COOKIECLICKPATH | A final file directory of “cookie\_click.mp3” |

3.2 Constructor

|  |  |
| --- | --- |
| Cookie() | Shows “Cookie.gif”. Sets oldx to 75 and oldy to 10. |

3.3 Method

|  |  |
| --- | --- |
| + void setUpCookie(Console console, LevelPane lp) | Setups current cookie. This method should:  - Sets current cookie’s speed to:  2000 - (The player’s level \* 100).  - When current cookie is clicked:   * Plays “cookie\_click.mp3” * Adds score on console, depending on Clicking Proficiency’s level. * If player’s level is higher than 10, increases progress in lp by 2%. If not, increases progress in lp by 5%. * If progress in lp is full, player’s level is increased by 1 and set progress to zero. |
| + void setPath() | Instantiates PathTransition for moving current cookie. |
| - Path createPath() | Creates and returns a Path for current cookie which is set a destination randomly in board. |
| Setter method for speed |  |

**4. Class Application.Console extends VBox**

4.1 Field

|  |  |
| --- | --- |
| - int score | The player’s current score. |
| - Label scorelabel | Label that shows the player’s current score. |
| - Label clickinglevellabel | Label that shows the player’s clicking proficiency level. |
| - Label clickinglevelcostlabel | Label that shows cost for upgrading the player’s clicking proficiency. |
| - Label autoclickerlabel | Label that shows the number of auto-clickers that the player has. |
| - Label autoclickercostlabel | Label that shows cost for buying next auto-clicker. |
| - Label addlabel | Label that shows the number of cookies on the board. |
| - Label addcostlabel | Label that shows cost for buying next cookie. |
| - int clickingcost | Cost for upgrading the player’s clicking proficiency level. |
| - int autoclickercost | Cost for buying next auto-clicker. |
| - int addcost | Cost for buying next cookie. |
| - ClickingUpgrader cookieup | Button that upgrades the player’s clicking proficiency level. |
| - AutoClickerBuyer autoclickerup | Button that buys next auto-clicker. |
| - ClickingUpgrader addcookie | Button that buys next cookie. |
| - String upgradingPath | A file directory of “upgrading.mp3” |
| - String consolebgPath | A file directory of “consolebg.jpg” |

4.2 Constructor

|  |  |
| --- | --- |
| + Console() throws IOException | Initializes the Console. This method should:  - Sets the alignment of the box to Pos.CENTER, the box has a width of 250.  - Sets the insets padding of 15.  - Sets the spacing of the Console to 10.  - Calls loadPath() to instantiate upgradingPath and consolebgPath.  - Border setting is:   * Its color is blue. * Its stroke style is solid. * Its radii of each of the four corner is empty. * Its width is default.   - Background setting is:   * Its size is 500\*1000. * Its image is “consolebg.jpg”   - Instantiates each field (All labels’ color is white):   * scoreLabel is a label with the text “Score: ” + this.score (Its font size is 40) * clickinglevellabel is a label with the text “Clicking Proficiency\n(Level ” + cookieup.getLevel() + “)” (Its font size is 15) * clickinglevelcostlabel is a label with the text “"Cost to next level: ” + clickingcost * Instantiates cookieup using ClickingUpgrader. * autoclickerlabel is a label with the text “Auto-Clickers\n(” + autoclickerup.getCount() + “ clickers)” (Its font size is 15) * autoclickercostlabel is a label with the text “Cost to buy next Auto-Clickers:” + autoclickercost * Instantiatess autoclickerup using AutoClickerBuyer. * addlabel is a label with the text “Number of Cookies\n(” + addcookie.getLevel() + “ cookies)” (Its font size is 15) * addcostlabel is a label with the text “Cost to add one more cookie: ” + addcost * Instantiates addup using ClickingUpgrader.   - When cookieup button is clicked:   * Plays “upgrading.mp3” * Decreases the player’s score by its cost. * Increases its level by 1. * Updates clickingcost, clickinglevellabel, and clickinglevelcostlabel.   - When autoclickerup button is clicked:   * Plays “upgrading.mp3” * Decreases the player’s score by its cost. * Increases its level by 1. * Updates autoclickercost, autoclickerlabel, and autoclickercostlabel.   - Instantiates Timer that increases the player’s score every one second, depending on how many auto-clickers do the player has.  - Sets its minimum size to 200\*200.  - Adds each field (except upgradingPath and consolebgPath) to the box as a children in the correct order. |

4.3 Method

|  |  |
| --- | --- |
| + void addScore(int n) | Increases the player’s score by n and updates scorelabel. |
| + void reset() | Resets all fields. |
| - void loadPath() | Instantiates upgradingpath and consolebgpath. |
| + void updateButton() | Update the status of each upgrade button.  If the score isn’t enough to upgrade, the button will be disabled. |
| Getters & Setters | Every field (except scorelabel, upgradingpath, and consolebgpath) has its getter. Only clickingcost and addcost has its setter. |

**5. Class Application.Board extends Pane**

5.1 Field

|  |  |
| --- | --- |
| + Random ran | An instance that is used to random an integer. |

5.2 Constructor

|  |  |
| --- | --- |
| + Board() | Initialized the Board. This method should:  - Sets the inset padding of 15.  - Sets the pane’s width of 850. |

5.3 Method

|  |  |
| --- | --- |
| + void addCookie(Console console, LevelPane lp) | Initializes a new cookie that has been setup already and adds it to Board. |
|  |  |
|  |  |
| + void reset() | Removes all nodes from Board. |

**6. Class Application.LevelPane extends VBox**

6.1 Field

|  |  |
| --- | --- |
| - int level | The player’s current level. |
| - Label levellabel | Label that shows the player’s current level. |
| - ProgressBar levelbar | ProgressBar that shows how much the number of clicks need to go to next level. |
| - double progress | The current progress in levelbar. |
| - HyperLink MainMenu | HyperLink that sends the player to WelcomePage. |

6.2 Constructor

|  |  |
| --- | --- |
| + LevelPane() | Initializes the LevelPane. This method should:  - Sets the spacing of the LevelPane to 10.  - Sets the alignment of the box to Pos.CENTER.  - Border setting is:   * Its color is blue. * Its stroke style is solid. * Its radii of each of the four corner is empty. * Its width is default.   - Background setting is:   * Its size is 500\*1000. * Its image is “consolebg.jpg”   - Instantiates each field:   * Sets level to 1. * Sets progress to 0. * levellabel is a label with the text “Level: ” + level. (Its color is white) * levelbar is a progress bar that its progress is progress. * MainMenu is a hyperlink with the image “main.png” and its border setting is Border.EMPTY.   - Adds each field (except progress) to the box as a children in the correct order. |

6.3 Method

|  |  |
| --- | --- |
| + void setLabel() | Updates levellabel follow level. |
| + void setBarProgress(double p) | Increases progress by p and updates levelbar. |
| + void reset() | Sets level to 1 along with levellabel and |
| Getters & Setters | Every fields (except levellabel) has its getter. Only level has its setter. |

**7. Class Application.WelcomePage extends VBox**

7.1 Field

|  |  |
| --- | --- |
| - Hyperlink start | Hyperlink that sends the player to in-game. |
| - VBox playerName | VBox that contains a label and a textfield for entering the player’s name. |
| - Hyperlink highscoresBtn | Hyperlink that send the player to HighscorePage. |
| - Hyperlink exit | Hyperlink that exits the game. |
| - String startPath | A file directory of “start.png” |
| - String hsPath | A file directory of “hs.png” |
| - String exitPath | A file directory of “exit.png” |
| - String bgPath | A file directory of “bg.jpeg” |

7.2 Constructor

|  |  |
| --- | --- |
| + WelcomePage() | Initializes the WelcomePage. This method should:  - Sets the spacing of WelcomePage to 10.  - Sets the alignment of the box to Pos.CENTER.  - Instantiates each field:   * title is a label with the text “Cookie Clicker” (Its font style is Ravie. Its font size is 100. It is white and underlined.” * Calls loadPath() to initialize startPath, hsPath, exitPath, and bgPath. * start is a hyperlink with the image “start.png” and its border setting is Border.EMPTY. * playerName is a VBox that contains a label with text “Player’s Name: ” (Its font style is Ravie. Its font size is 15. It is white.) and a textfield that has a width of 150 and prompt text “Enter your name her.” * highscoresBtn is a hyperlink with the image “hs.png”, which has been resized to 150\*150, and its border setting is Border.EMPTY. * exit is a hyperlink with the image “exit.png”, which has been resized to 150\*150, and its border setting is Border.EMPTY.   - Background setting is:   * Its size is 500\*1000. * Its image is “bg.jpeg”   - Adds each field (except startPath, hsPath, exitPath, bgPath, highscoresBtn, and exit) to the box as a children in the correct order. |

7.3 Methods

|  |  |
| --- | --- |
| + String getPlayerName() | Returns the text in the textfield of playerName. |
| - void loadPath() | Instantiates startPath, hsPath, exitPath, and bgPath |
| Getter methods for start, highscoresBtn, and exit |  |

**8. Class Application.HighScorePage extends VBox**

8.1 Field

|  |  |
| --- | --- |
| - Label highscoreheaderrlabel | Label that is the header of HighScorePage. |
| - Label highscorelabel1 | Label that shows name and score of the player who got the first place. |
| - Label highscorelabel2 | Label that shows name and score of the player who got the second place. |
| - Label highscorelabel3 | Label that shows name and score of the player who got the third place. |
| - ArrayList<Pair<String,Integer>> highscores | ArrayList that is used for collecting name and score of three players who got the highest score. |
| - Hyperlink MainMenu | Hyperlink that sends the player to WelcomePage. |
| - String bgPath | A file directory of “bg.jpeg” |
| - String highscorePath | A file directory of “highscore.txt” |
| - String mainPath | A file directory of “main.png” |

8.2 Constructor

|  |  |
| --- | --- |
| + HighScorePage() | Initializes the HighScorePage. This method should:  - Sets the spacing of HighScorePage to 15.  - Sets the alignment of the box to Pos.CENTER.  - Background setting is:   * Its size is 500\*1000. * Its image is “bg.jpeg”   - Instantiates each field:   * Calls loadPath() to instantiate bgPath, highscorePath, and mainPath. * highscores is ArrayList that its size is 3. Instantiates BufferReader to read “highscore.txt”, creates three pairs and adds them to highscores * highscoreheaderlabel is a label with the text “Highscore Ranking” (Its font style is Ravie. Its font size is 70. It is white.) * highscorelabel1 is a label with the text “#1: ” + highscores.get(0).getKey() + “ ” + highscores.get(0).getValue() (Its font style is Ravie. Its font size is 50. It is white.) * highscorelabel2 is a label with the text “#2: ” + highscores.get(1).getKey() + “ ” + highscores.get(1).getValue() (Its font style is Ravie. Its font size is 50. It is white.) * highscorelabel3 is a label with the text “#3: ” + highscores.get(2).getKey() + “ ” + highscores.get(2).getValue() (Its font style is Ravie. Its font size is 50. It is white.) * MainMenu is a hyperlink with the image “main.png” , which has been resized to 150\*150, and its border setting is Border.EMPTY.   - Adds each field (except bgPath, highscorePath, and mainPath) to the box as a children in the correct order. |

8.3 Methods

|  |  |
| --- | --- |
| + void refreshHighscores() | Writes a new highscores on “higscores.txt” and updates highscorelabel1, highscorelabel2, and highscorelabel3. |
| - void loadPath() | Instantiates bgPath, highscorePath, and mainPath. |
| Getters & Setters | MainMenu and each pair in highscores has its getter. But only pairs in highscores has its setter. |

**9. Class Application.Main extends Application**

9.1 Field

|  |  |
| --- | --- |
| - int interval | Current time for playing this round. |
| - Console console | An instance of Console. |
| - Board board | An instance of Board. |
| - LevelPane levelpane | An instance of LevelPane. |
| - WelcomePage menu | An instance of WelcomePage. |
| - HighScorePage hspage | An instance of HighScorePage. |
| - Thread timer | Thread for counting time left. |
| - String bgPath | A file directory of “bg.jpeg” |
| - String bgmPath | A file directory of “Fluffing a Duck.mp3” |

9.2 Method

|  |  |
| --- | --- |
| + void start(Stage primaryStage) throws Exception | The main entry point of JavaFX application. This method should:  - Creates a root container using HBox with “bg.jpeg” as background. (Its size is 500\*1000)  - Initializes Console, Board, and LevelPane and adds them to the root container.  - When addcookie button in the Console is clicked:   * Decreases the player’s score by its cost. * Add one more cookie to the Board. * Updates addcost, addlabel, and addcostlabel.   - When start button in the WelcomePage is clicked:   * Sets the root as the scene. * Initializes interval to 100 and a label called timeElapsed with the text “Time Left: ” + interval. (Its font size is 30 and it is white.) * Initializes an information-type alert with a title “Time Out” and a header text “Time Out” * Initializes timer that decreases the interval every 1 second by 1 while updates timeElapsed. If the interval is lower or equals 10, timeElapsed’s color will become red. * When the interval is zero, sets its color to white and show the information-type alert. * Checks that the player’s final score. If it is higher than the three high scores in “highscores.txt”, write it with new high score. * Sets HighScorePage as the scene. * Resets Console, Board, and LevelPane with their own reset().   - Sets the highscoresBtn in WelcomePage, the MainMenu in HighScorePage, the exit in LevelPane, and exit in WelcomePage to set the appropriate scene.  - Sets the WelcomePage as the first scene and title “Cookie Clicker”, not resizable and then show the stage |
| - void loadPath() | Instantiates bgmPath. |
| + void startBgmLoop() | - Calls loadPath() to instantiate bgmPath.  Plays “Fluffing a Duck.mp3” on loop. |
| + void main(String [] args) | An entry point of the application. |