Algonquin College Logo

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A11

Game Interface

Team:

[Boyu Li] - Id: [041003345]

Game Proposal - Battleship

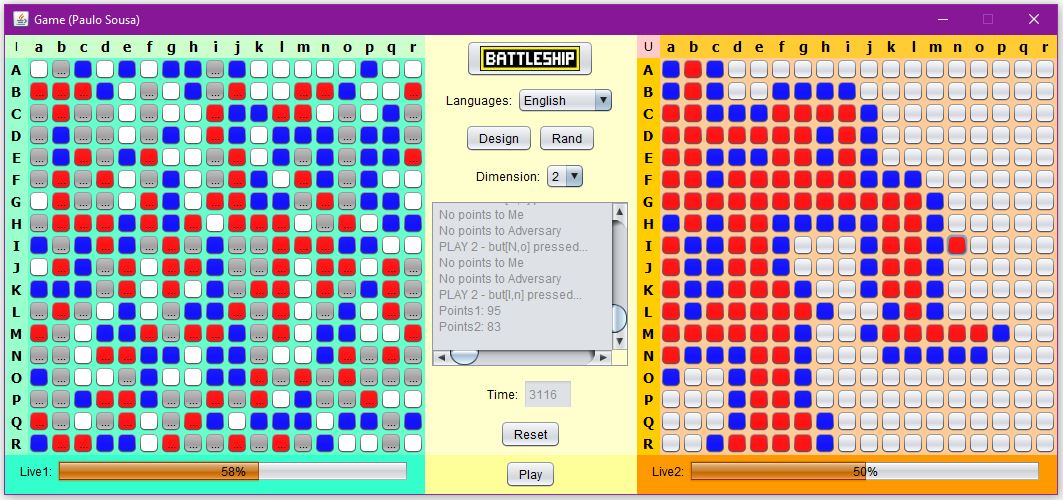
***This template is suggested (not mandatory) to answer A11 Specification.***

|  |  |
| --- | --- |
| **Part**  **1** | **GUI Definition** |

**EXPLANATION**

*The purpose of this assignment is to define the elements of the GUI application to be used in your game implementation.*

* ***Example (Prof. suggestion)****:*



* ***Note****: The professor interface is also a proposal. It means that your own implementation can be different. What does matter is that the game functionality will be respected.*
  1. **Defining the Functionalities**

**Main Behavior**

*Include the list of functionalities that you imagine to have in the game (they can be from Swing or JavaFX).*

***Answer:***

***The class will be included in the bracket()***

JFrame: The main window of the game (baseWindow)

JPanel: Displays the User Panel at the left side of the game (userPanel)

JPanel: Displays the coordinates of the column at the top of the User Panel (topUserPanel)

JLabel: Displays the coordinate of each column (columnUserLabel)

JPanel: Displays the coordinates of the row at the left of the User Panel (leftUserPanel)

JLabel: Displays the coordinate of each row (rowUserLabel)

JPanel: Displays the game of the life at the bottom of the User Panel (bottomUserPanel)

JProgressBar: Displays the game life (lifeUserBar)

JPanel: Displays User Playing Area (the matrix) at the center of the User Panel (boardPanel)

JButton: Displays the matrix of the buttons (matrixUserButton)

JPanel: Displays the Adversary Panel at the right side of the game (adversaryPannel)

JPanel: Displays the coordinates of the column at the top of the Adversary Panel (topAdversaryPanel)

JLabel: Displays the coordinate of each column (columnAdversaryLabel)

JPanel: Displays the coordinates of the row at the left of the Adversary Panel (leftAdversaryPanel)

JLabel: Displays the coordinate of each row (rowAdversaryLabel)

JPanel: Displays the game of the life at the bottom of the Adversary Panel (bottomAdversaryPanel)

JProgressBar: Displays the game life (lifeAdversaryBar)

JPanel: Displays User Playing Area (the matrix) at the center of the Adversary Panel (boardPanel)

JButton: Displays the matrix of the buttons (matrixAdversaryButton)

JPanel: Displays the Control Panel at the center of the game (controlPanel)

JLabel: Displays the logo of the game (logoLabel)

JCommonBox: The selector of the game language (languageBox)

JButton: Setting the game mode to design mode (designButton)

JButton: Setting the game mode to random mode (randButton)

JCommonBox: The selector of the game dimension (dimensionBox)

JTextArea: Displaying the game log/history (historyArea)

JTextField: Counting and displaying the game time in second (timeField)

JButton: Resetting the game (resetButton)

JButton: Start to play the game (playtButton)

**Functionalities and Behaviors**

*Check, for example, on-line examples (*[*https://www.battleshiponline.org/*](https://www.battleshiponline.org/)*):*



*What are the behaviors and functionalities that you will provide? How these elements are related with functionalities.*

***Example****: Answer these questions:*

* *Who are the actors (who can design / play the game)?*
* *What are the preconditions (requirements) for some functionalities?*
* *And the post-conditions / results?*

***Answer:***

1. There will two actors to design/play the game. The first actors are users, they can design/play the game based on their own thoughts. The second actor is the game itself; it can help user to randomly design a game for them.
2. ***The preconditions (requirements) for some functionalities:***

The logo displaying of the game will be implemented by JLabel

The Language switching will be implemented by JComboBox

The Design mode switching will be implemented by Jbutton

The Random mode switching will be implemented by Rand

The Dimension switching will be implemented by JComboBox

The Game time counting and displaying will be implemented by JTextField

The Game resetting will be implemented by JButton

The Game playing will be implemented by JButton

The Game log displaying will be implemented by JTextArea

The User Panel and Adversary Panel will be implemented by the matrix of the JButton

The Coordinate of the User Panel and Adversary Panel will be implemented by the JLabel

The Displaying of game life will be implemented by the JProgressBar

1. ***The post-conditions / results:***

The game can show its logo on the GUI

The users can switch the languages by choosing any language option on the “Languages” combo box

The users can play the game in design mode by hitting the “Design” button

The users can play the game in random mode by hitting the “Rand” button

The users can switch the Dimension by choosing any digit option on the “Dimension” combo box

The users game time will be displayed in second at the “Time” text field

The users can reset the game by hitting the “Reset” button

The users can start the game by hitting the “Play” button

The game can display its log/history on the GUI

The game can show its User Panel and Adversary Panel like a square grid

The game can show the coordinate of the User Panel and Adversary Panel

The game can show its life at the bottom of the User Panel and Adversary Panel

**Languages**

*Define (at least two) languages to be used – remembering that English is mandatory for one option.*

***Example****: The second language (French) will be chosen, since this is my natural (birthplace) language.*

*Answer*: The second language I would choose is Chinese, as it is my native language. Additionally, there is a large population of active gamers in China, which would make it an effective marketing strategy if this were a real use case for the game.

**Details**

*Drawn the UC (Use-Case) diagram (ex: in an image from Paint / Visio / Powerpoint slide, or any sketch tool), describing:*

* *Manual / automatic features (ex: user selections / time features);*
* *Relationships between actors / functionalities.*
  1. **Template Solution**

**UC Diagram** (example):



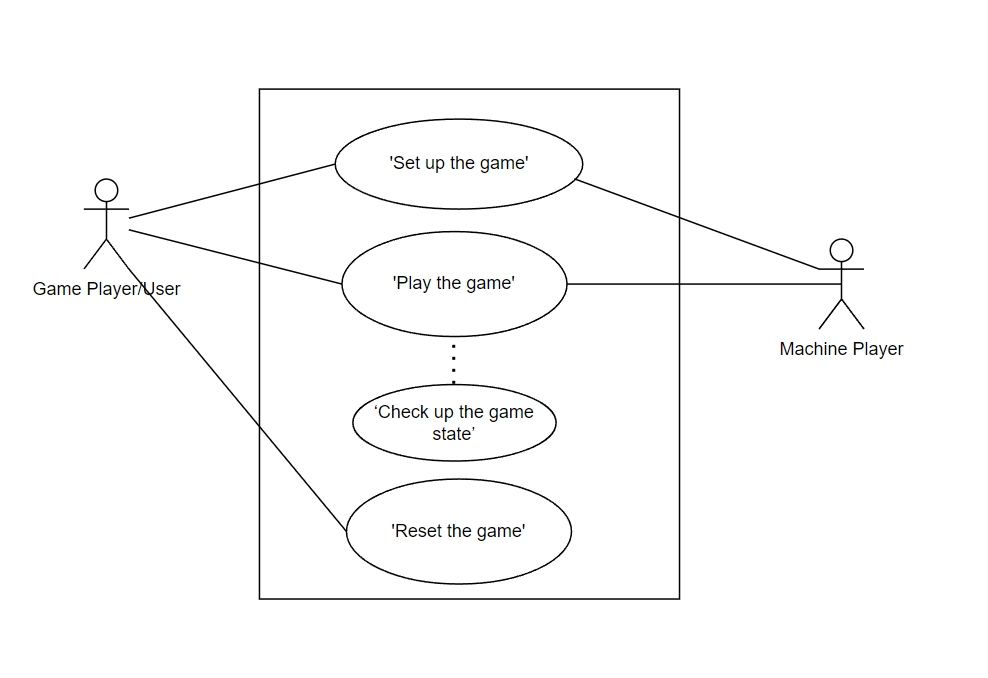
**Actors table** (example):

|  |  |
| --- | --- |
| **Actors** |  |
| Bank Customer | This actor represents a person with a valid Bank Card. The Bank Card is theirs and they know the PIN Code. |
| Bank | This actor represents the financial institution that provides services to the ATM. Responsible for verifying Bank Customers, authorizing transactions and recording completed transactions. |
| Maintenance Person | This actor represents the person responsible for maintaining the Automated Teller Machine, refilling paper, and replenishing cash. |

**UC table** (example):

|  |  |
| --- | --- |
| **Use Cases** |  |
| Withdraw Cash | This use case describes how the Bank Customer uses the ATM to withdraw money his/her bank account. |
| Transfer Funds | This use case describes how the Bank Customer uses the ATM to transfer money between different bank accounts. |
| Deposit Funds | This use case describes how the Bank Customer deposits money to an account. |
| Refill Machine | This use case describes how the Maintenance Person refills money, receipt paper and envelopes. |

***Answer:***

******

**Actors table**

|  |  |
| --- | --- |
| **Actors** |  |
| Game Player/User | This actor represents a person is trying to use/play the game |
| Machine Player | This actor represents the Game Application which reacts the user with corresponding functionalities when user doing any operation for the game. |

**UC table**

|  |  |
| --- | --- |
| **Use Cases** |  |
| Setup the game | This use case describes how the user to set up the game with their own preference on language, dimension, mode on the game application. |
| Play the game | This use case describes how the user play the game on the panel of the game application. |
| Check up the game state | This use case describes how the user check the state of the game on the GUI such as life, time on the game application. |
| Reset the game | This use case describes how the user can reset and get a new turn of the game on the game application. |

**Basic cycle**

*Create a brief description about how your game can be used.*

***Example****: If you have to design the solution to be saved and played later, how are the stems. Most importantly, how someone can play the* ***Battleship****.*

* *Note: your process does not need to be followed exactly when you are going to the implementation. For while, it is only a script about how to play.;*

***Answer:***

The rule of the game should follow up the rule of the Battleship.

Before the game start, the users should fulfill the basic game information. Firstly, the user should select the language from the language combo box. Then, they should choose the game mood either “Design” or “Random” mode by hitting the button. After that, they should select the dimension from the dimension combo box. Once everything setting up, the user can simply start the game by hit the “play” button.

The user should play the battleship at the User Panel (left one). User can choose the attacking area by simply hit the one piece of button on the panel. After that, the game would tell you whether successfully cause a damage by color of the button. The life of the user will be displayed at the bottom of the panel.

User can reset their game anytime by hit the reset button.

**FINAL SUGGESTIONS**

*Here some ideas to think about your language....*

* *Try to create a game whose execution can be very intuitive (easy to be played).*
* *Remember that this game will be in fact implemented only in the next assignment.*

**References**

*[Include eventual references used here]*

* ***NOTE****: Even if you use one specific tool (ex: ChatGPT), report it here.*

Algonquin College

Summer, 2023