Homework 2

UML Diagrams

Emmanuel Jacquez, Hector Dozal

**State Machine**

1

7

* 

2

* 

3

* 

4

* 

5



* 

6

**Transitions**

1. SudokuDialogue class asks user for a Sudoku board size, Board class constructs board of selected size. (4x4, 9x9) and BoardPanel outputs the grid.
2. SudokuDialogue asks user for a button triggered event that retrieves an input of 3 integers, SudokuDialogue asks user for those 3 integers and passes them to Board class, Board class then checks the integers and replaces them.
3. Board class returns to SudokuDiologue or the idle state waiting for user input.
4. SudokuDialogue class asks BoardPanel class to output the board that’s stored is stored in a 2d array that is in Board class.
5. After the output of the integers on the grid the program then returns to the idle state of SudokuDialogue.
6. If the integers entered result in the completion of the game, the Board class determines the Sudoku completion ant outputs the integers the BoardPanel class generated grid, exits from the Board class, congratulates the player and then enters an idle state that will wait for the user to enter a new game.
7. If the player chooses to quit the game the game closes.

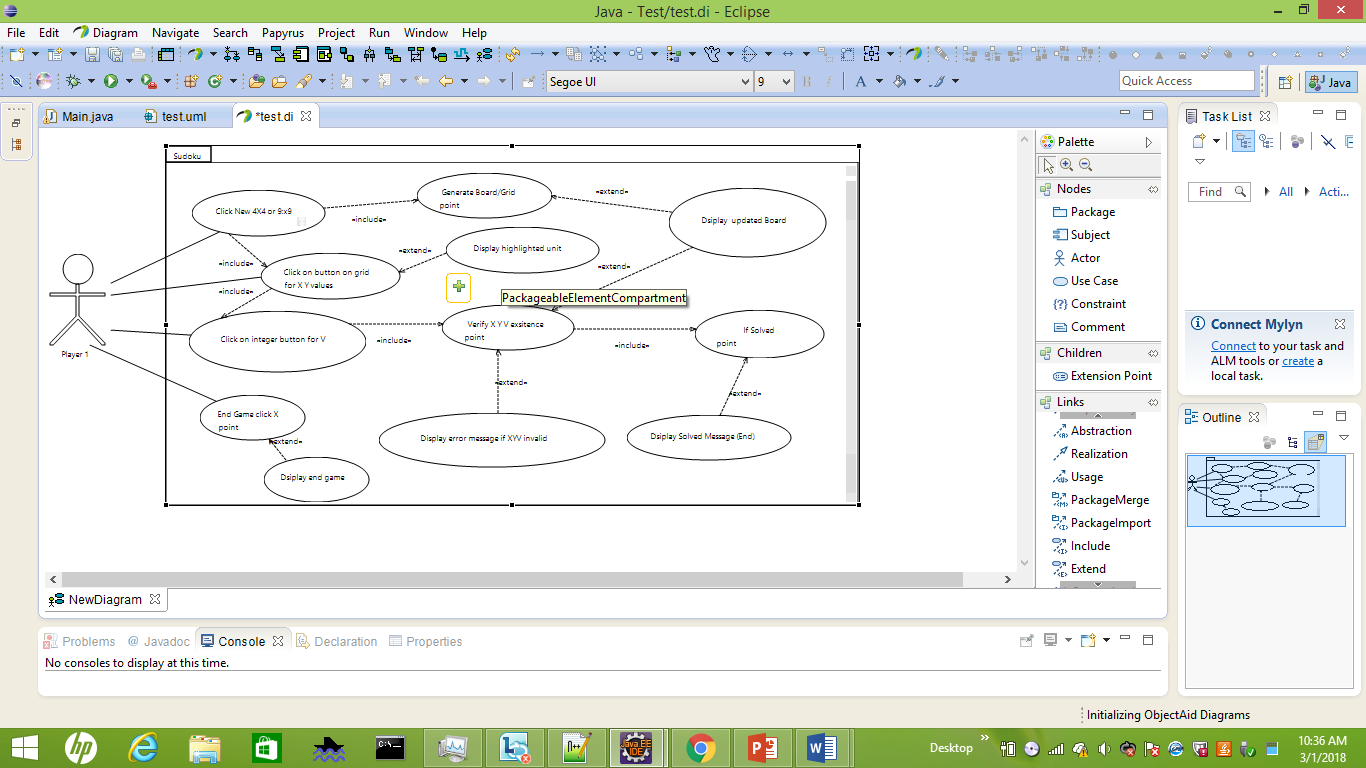
**States**

1. Start State the game enters this state after starting the game, this state’s only purpose is to initiate game and wait for user dependent event from “New x \* x” size button click.
2. Idle state after the player chooses a size the game enters this state in which the player is given a number of options to choose ranging from 1-9.
3. Insert State if the player chooses to enter a number the game moves to this state. In this state the player is

required to click on a number and at a position on the grid that has not been filled on the Sudoku yet.

1. Output state If number clicked on is validated and follows Sudoku format, it will be printed on the grid at the chosen position.
2. End State 1 if the player’s inputs are correct the game enters this state where it congratulates the player using a new window and lets the player start a new game or quit the game.

6. End State 2 if the player chooses to quit the game, the game closes

Use Case Diagram

Class Diagram

Use Case: Play Sudoku game

Description: Create Sudoku and optionally solve Sudoku

Actors: Player 1

Pre-Condition: Game has started

Post-Condition: Sudoku board has been solved; message shown

Alternative-User clicks on X button and system quits and stops game “Goodbye message shown”. User clicks on “New xXx” and a new game will be initiated at any point during the game or at the conclusion.

Exception- System controls possible exception input by user by limiting the input options to non-exception system inputs.

1. System initiates an empty frame with buttons that specify and prompt player for the size of the Sudoku.
2. Once a size button is clicked the Sudoku button a “new game” message is displayed, and a board is constructed with a (size \* size) grid dimension along with clickable buttons ranging from 1>size.
3. After click event on the grid by a user, the board will then retrieve the X and Y location with the use of the click location on the Sudoku board grid.
4. User clicks on integer valued button that will assign the value to the V value that will be inserted to the location of the grid if it follows the game parameters.

4a. Checks X Y V for their current existence in the same Row, Column, and size\*size box

4b. If 4a is false, Sudoku is not updated an error sound/message is displayed and step 3-4 are to be repeated.

4c. if 4a is true, Sudoku is updated and for the user to see and continue updating until board is solved.

1. Once board is filled with integer values <0 || size<= values the Sudoku is solved and a congratulatory message of success is outputted.

System

1. Player initiates game
2. Player clicks button that provides size for Sudoku (4x4 or 9x9) or ends game with X (Exit) button click.
3. Player clicks on new game button to start game of size X.
4. Player clicks on grid location where they would like to insert a value on.
5. Player selects and clicks the button of the number that contains an integer value that would be inserted into clicked grid location.
6. Step 5 is repeated until Sudoku grid is filled with integer values that follow the game parameters.
7. Sudoku Solved. End of game

PLayer 1