SUDOKU GAME

TEAM MEMBERS :

P. Sai Praneeth (20MCME18)*
M. Likhith Sesha Sai (20MCME29)

CLASSES INVOLVED :

1. PlayGame (Main Class)
2. GenerateSudoku
3. JTextFieldLimit

USAGE :

Keep all the three classes in a single folder. Go to the folder, Compile all the three classes and run the class “PlayGame” to play Sudoku.

OUR APPROACH TO THE PROJECT :

For a user to play the game, we need a Frame containing Grid Layout containing Sudoku grids and the user should be able to enter text in the text fields. Hence we need to create a 2 dimensional array of Text Fields in which some of them are not editable by the user.

To play the game, we first need to generate a valid sudoku grid. Thus we created a class “GenerateSudoku” which creates a new valid sudoku grid everytime we run the code.

To avoid Number Format Exceptions we need to restrict the user to enter only certain elements (1,2,3…9) and also restrict them not to enter more than one digit in a textfield. The class “JTextFieldLimit” helps in doing this.

DESCRIBING EACH CLASS :

1. GenerateSudoku

In this class we start generating the Sudoku grid by first generating the 3 diagonal 3X3 boxes using a random number generator and checking every time whether it is valid to put that number in the specific 3X3 box.

After generating the 3 diagonal boxes (3X3) we start to fill the remaining 3X3 grids by calling a function “RemainingGrids” recursively. This function also uses a random generator and everytime checks whether it is valid to put that number in the row, column and box.

2. JTextFieldLimit :

This class is extended from the Class PlainDocument. It has an integer as an instance variable which is the maximum limit of the text that we can enter in a textfield. In our case the limit is 1.

3. PlayGame :

This is the Main Class in the game. It implements ActionListener. It contains the Majority of the State Information. It also contains instances of the other two classes in its state information, leading to a weak association between them as shown in the Screenshot 1.png. We begin with creating a frame, a grid layout, a two dimensional array of Text Fields, a LabelPrompt, 2 buttons (Validate and Reset) and 4 different panels for different components. After that we generate the Sudoku grid using the available class, and store the values of the grid in the 9x9 Text field array using two for loops. We also added a specification that users cannot enter anything other than the numbers from 1 to 9. If at all the user enters anything then the function e.consume() will consume and it is not entered in the field.

Now we need to make some Text fields empty (In our case at most 3 fields are empty in every 3x3 grid) and also restrict edit access to the other Text fields. For that we call a method named "make3grid", it uses two for loops and a random generator for allocating empty cells and to make non editable cells.

We have two buttons: Validate and Reset. The Reset button will simply clear the content of editable Text fields. The Validate button when clicked prompts the user in three cases :

1. Some of the TextFields are Still Empty. (We ask the user to fill all the grids first).
2. All the grids are filled but the solution is wrong (We inform user that Solution is incorrect and ask him/her to try again)
3. All grids filled and Solution is Correct (Inform the user that the Solution is correct)

Note : The Maximum number of empty cells in a 3X3 box can be changed accordingly by the user if he wants to, by changing the value of the integer variable “Empty_in_grid” in the state information of PlayGame class.