Sudoku - User Manual

Tsugunobu Miyake , Doruk Ozar , Reva Sharma , Kit Jackson
Prof. Brian King
Software Engineering and Design

Introduction:

Based on an early mathematical analysis concept designed in 1782, Sudoku is a logic-based numbers game. The objective of a Sudoku puzzle is to replace the blanks (or zeroes) in a 9x9 grid such that each row, column, and 3x3 box contains each of the digits between 1 to 9. A well-made Sudoku puzzle has a single solution and can be solved logically, however it might be necessary to use "guess and test" techniques to rule out potential solutions. The complexity of the search determines the difficulty of the puzzle. Most players use a pencil with a paper version like a crossword puzzle. Below is an example of a puzzle in its initial (left) and completed (right) stages.

8			4		6			7
						4		
	1					6	5	
5		9		3		7	8	
				7				
	4	8		2		1		3
	5	2					9	
		1						
3			9		2			5

8	3	5	4	1	6	9	2	7
2	9	6	8	5	7	4	3	1
4	1	7	2	9	3	6	5	8
5	6	9	1	3	4	7	8	2
1	2	3	6	7	8	5	4	9
7	4	8	5	2	9	1	6	3
6	5	2	7	8	1	3	9	4
9	8	1	3	4	5	2	7	6
3	7	4	9	6	2	8	1	5

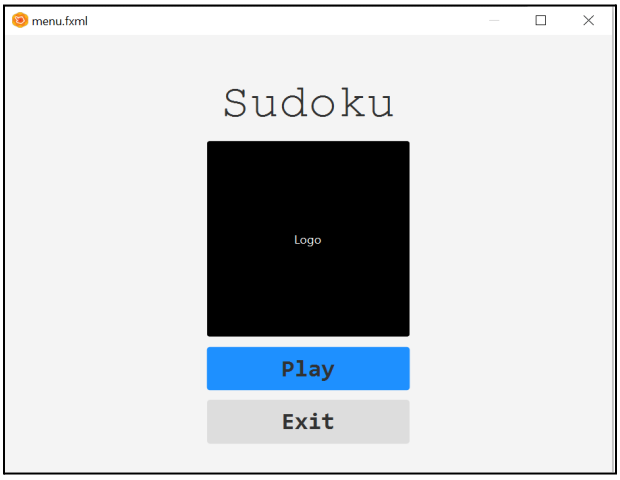
Our project is a GUI based version of this popular game Sudoku. The main objective of this project was the design, implementation and evaluation of this application that allows users to solve Sudoku games and potentially request for hints. Sudoku puzzles can be solved with the resultant app. Users of the app can solve a puzzle by inputting numbers and highlighting cells.

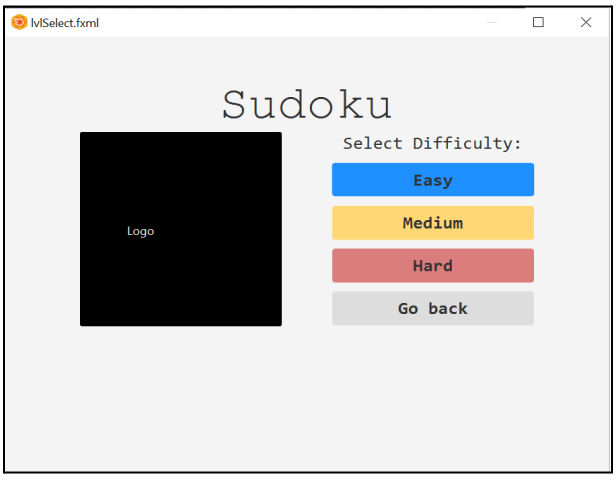
Additionally, undo and delete functionalities are also a part of the game, and cells that were filled improperly are highlighted.

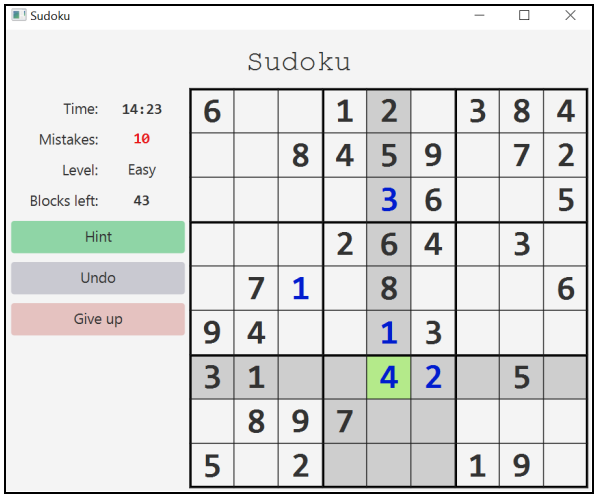
Background and Motivation:

While we always knew we wanted to build a game, we poked around with a few ideas before deciding on the idea of Sudoku. During the research part of our project, we looked at a number of different games online but then collectively decided on Sudoku as its rules and implementation are difficult, but at the same time not too complex that we couldn't present a completed product in due time, i.e. Sudoku is an easy to play game but it is still complicated enough that it could be implemented within a month's time.

Through research it was easy to find that there are multiple sites online that offer play of Sudoku. This game has been around for a long time and its fascination has carried through ages with several programmers attempting to replicate this product. Despite this, we could not come across many solutions that involved using JavaFX and Scene Builder to implement this game, and decided to build it using these tools. Therefore, enough demand and appeal existed so that we could go forward with building Sudoku. Some of our initial UI mockup screens are shown below:

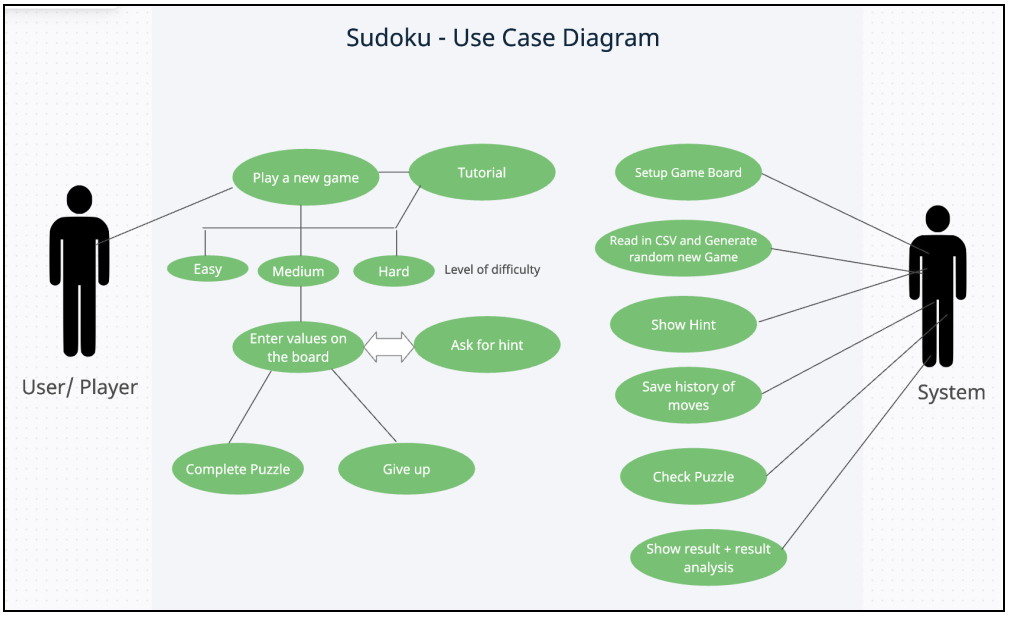






Our motivation for this project was to put all the tools and concepts we have learnt over the course of this semester to use, while building upon certain skills that we did not find enough time to delve into during class. Implementing a game that is challenging enough to still leave us asking thought provoking questions proved to be a logical line of action because we all appreciate games and tasks that push us just a little bit outside of our comfort zones. Additionally, upon evaluation we figured that this particular game would make use of our respective strengths.

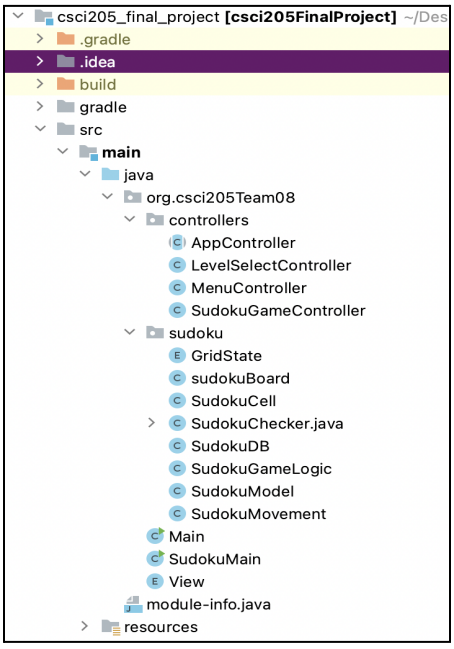
The strategy is to focus on a specific blank cell and use the constraints to eliminate invalid choices. It is easier to start in a sub-grid that has relatively more values than in a mostly empty part of the grid, since there will be less constraint information available to eliminate some choices. Based on the compilation of our user stories, the below use case diagram comprehensively shows how a user would generally go about playing the game. Similarly, the general overview of the tasks managed by our system is also shown:



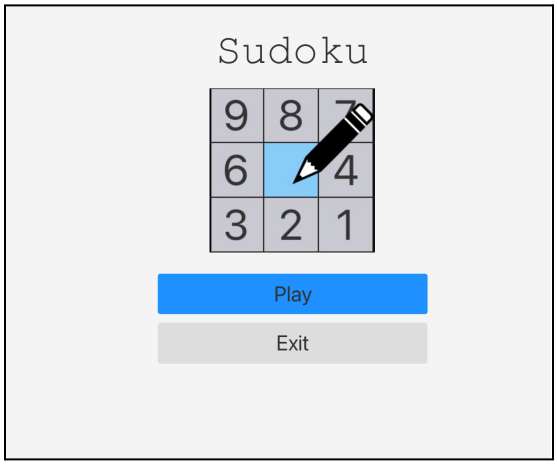
Instructions:

The user only interacts with the JavaFX scene which gets generated when the application is being run. This game is made for users who have played Sudoku before and also for new players who can get familiar with the game through the inbuilt tutorial option. The following instructions should be followed:

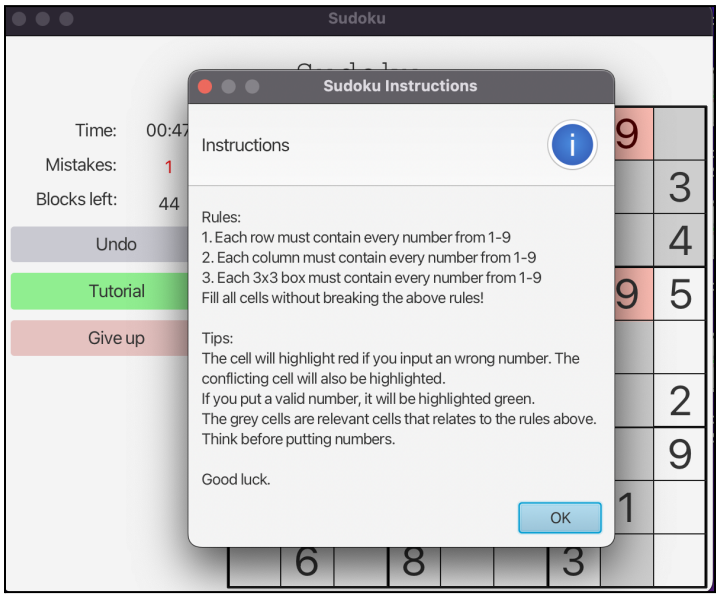
1. In order to play the game, select the SudokuMain class (as shown in the figure below) and run the program.



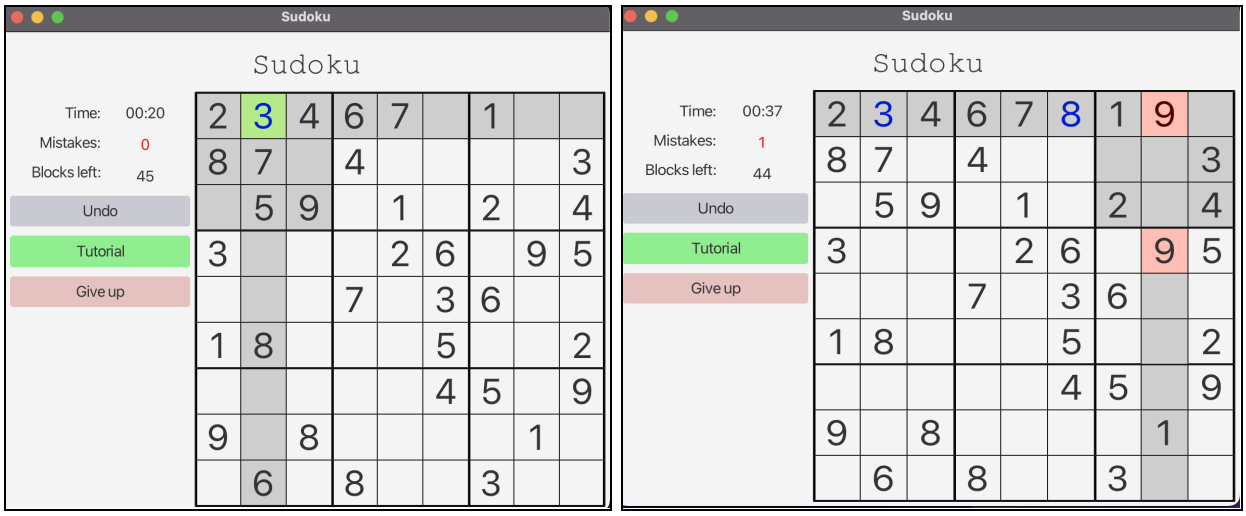
2. The program should run and the player would be presented with the following screen once the user selects play.



3. A random Sudoku board appears on the screen. You can choose to look at the tutorial by clicking on the relevant option.

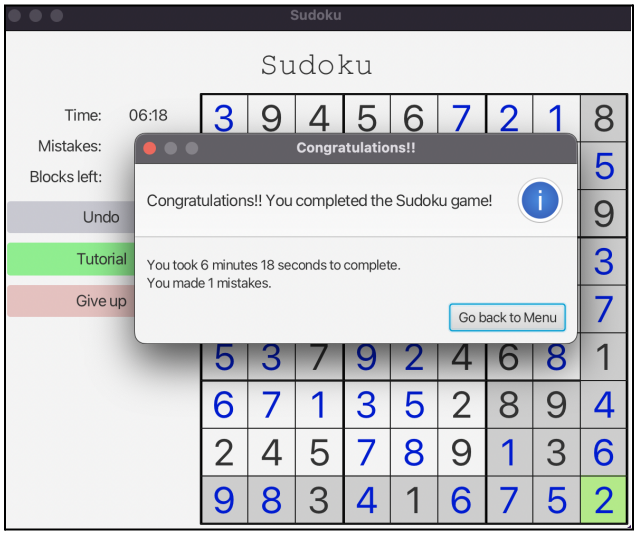


4. The preset numbers that are a part of the quiz are marked in black and the numbers put in by the user are shown in blue.



Valid input is highlighted in green and invalid input is highlighted in red. Subsequently, the conflicting cells are also highlighted red. This input is not saved, hence if the user clicks other cells, the cells will become blank again. The analysis of all moves is shown on the left side of the screen.

5. Once the user inputs all the correct numbers, the puzzle is complete and we are prompted with the following completion popup.



6. Either close the application or play another game.