KenKen puzzle solver

Tuesday, August 3, 2021 11:45 AM

Application specification:
A KenKen puzzle solver takes in a board of various sizes, a list of values for specific cells and a list of formulas for specific sets of cells. It returns the board with all correct values filled in.
Constraints:

- Size of boards are 3x3 up to 9x9.
- Formula symbols are +,-,x,/.
- If the board can be solved in multiple ways, only one of the ways will be reported.
- KenKen rules state that on a single row, values cannot repeat and are between 1 and size
- KenKen rules state that on a single column, values cannot repeat and are between 1 and size
- KenKen rules state that values in cells need to satisfy the formulas they belong to

Inputs specification:

- Board size: size is 3 to 9
- A list of values for specific cells:
    - x,y,value;x,y,value…
    - x = horizontal location on the board, 0 is left-most, 0<=x<size
    - y = vertical location on the board, 0 is top-most, 0<=y<size
    - value = value of the cell, 1<=value<=size
- A list of formulas for specific sets of cells:
    - Result, symbol,x,y,x,y…;result,symbol,x,y,x…
    - Result: the result of a formula
    - Symbol: +,-,*,/
    - x: 0<=x<size
    - y: 0<=y<size

Input file specification:

- Board size;result,symbol,x1,y1,x2,y2,x3…;

Output specification:

- If the input has no solution, it will report no solution.
- Values on row 1; values on row 2;…

Solver display overview:

- Draw a size * size table
- Each formula is outlined by a thick lines
- Top left corner of each formula shows result followed by symbol (TBD)
- Each cell displays its value

Solver logic overview:

- The value of a cell must be 1<=value<=size of board
- Cells on the same row/column must have unique values
- Each formula needs to be evaluated
    - Addition, two values+: x1+x2… = z;
    - Subtraction, 2 values: x1-x2 = z or x2-x1=z
    - Multiplication, 2 values+: x1*x2… = z
    - Division, 2 value: x1/x2=z or x2/x1=z
    - Empty symbol: x=z
- When available values for cells drop down to 1, x = available value

- When available values for cells drop down to 0, error
- When an available value changes for a cell, the formula containing the cell needs to be reevaluated
- When a cell is solved, all cells with the same x or y need to be reevaluated
- Advanced:
    - When two cells share the same two available values, it triggers all cells that share the same x or y that the above two cells share to get reevaluated (remove the two available values from those two cells…)

Rapid prototyping:

- Reduce KenKen puzzle to 3x3 board with filled cells and formula symbol of +.
- Work flow:
    - Load input from text file
    - Apply solved cells onto board, each solved cell should trigger reevaluation of certain cells (add to priority queue)
    - Apply formulas onto board, each formula should trigger reevaluation of certain cells (add to priority queue)
    - Process priority queue until it is empty
    - Return the results
        - unsolved board can show available values
- Re-evaluation types:
    - Solved a cell: Re-evaluate all cells where x = solvedCell.x || y = solvedCell.y, add job to remove solvedCell.value from all thoses cells onto the priority queue
    - A formula: see above, each case should add one or more items to the priority queue
- Priority queue:
    - Happens in a priority queue,
    - (TBD) if it becomes empty before the board is solved, then guessing is needed; incomplete rules
    - Each queue item needs information on: cell location (x,y), remove available values a,b,c… from cell (can only remove since all cells are assumed to have all values available)
    - Each cell needs to have location (x,y), value, count of available values (availArr.length), available values (availArr)
    - Error if availArr.length === 0;
    - Solve cell if availArr.length === 1; call solvedCellRe-evaluation
- When priority queue is empty
    - if Solved board === cells.solveCount, board is solved
    - Otherwise, board is not solved

What's the relationship between formulas, cell info and work items?

- Formula's format: result, symbol, cell coordinates
- CellInfo's format: value, formula, cell coordinates
- Work item: a simple todo; cell coordinate, value to remove from possible values; trigger reevaluation of the cell
- Some default rules need to be applied somehow:
    - Cell values across a row/column have to be unique <= need to trigger whenever a possible value is removed
    - Cell values have to be between 1 and size <= built in the possibleValues

What causes work items to be created?

- Whenever a cell's possible value changes

What changes a cell's possible value?

- Processing formulas (board.processFormulas())
    - when board is initially loaded

    - when a cell's possibleValue changes (cellInfo.formula)
- When a cell's value is set (board.setCellValue(x, y, value))
    - When first processing formula where symbol === ' '
    - When available values for a cell goes down to 1 <= when will board checks this? TBD

When does a cell's value gets set?

- Initial formula processing where symbol is ' '
- possibleValues.length drops down to 1

Work items and bugs:

- ☑ When loading board initially, show initial set values, process formulas later
    - Instead of calling processing of formulas or removing possible values across cells right away, add them as work items in **board.workitems** (a priority queue).
    - Each work item should have a function to call and a list of arguments
    - Processing of the work items is triggered by solve(), which continuously process each work item until the queue is empty.
- ☑ Add console logging
- ☑ Add support for -
- ☑ Add support for /
- ☑ Add support for + and -
    - Given: **formula.coordinates** cells, a symbol **formula.symbol === '+'**, a result **formula.result**
    - Each cell contains an array of possible values **cells[i].possibleValues**
    - Find: remove possible values that cannot satisfy the formula
    - Thoughts: result = c1 + c2 + c3…cN; to check a possible value **c1.one**; all c2 to cN's values need to be checked; if this formula c1.one = result -c2-c3…cn is satisfied. The value is right.
    - Brute force: take a possible value, calculate possible = result - c1.one, since we don't know what n is until run time, use recursive until the last loop. So a base case for the recursion is index of **coordinates === coordinates.length - 1,** then **return cN.possibleValues.find(x => x === possible);** the reduction loop should be check each possible value for the current coordinate **coordinates[index]**, calculate the possible values and call the next loop **validateAgainstPossibleValues(table, coordinates, index, result, getPossibles);**
- ☑ Add support for when same set of possible values exist for x number of cells on the same row (like 1,5 at [0,1],[1,1]), then the rest of the cells of the same row (1) cannot have those values (remove 1 and 5 from x sets of cells
    - In a simple two cell one, the possible values of the two cells are the same <- easy, just check to see if the possible values are equal
    - In a more complicated 3 cell one, it can look like this 1,4; 2,4; 1,2; there are 3 cells and the 3 cells hold 3 possible values 1,2,4
    - Can I do a count of possible values? For x = 6,

V = 1	2	3	4	5	6
Y = 1	1	1	1	1	
2	2		2	2	2
3	3		3	3	3
	4	4	4		4
5		5			
6		6			

    - What's the brute force way to do this?
        - Maybe first to just to compare the cells for ones that are the same? If they are and their possible values.length === 2, then remove those possible values from the rest of the cells:
    - Can I sort the cells by their possible values?

- For x = 6, sort possible values' lengths from shortest to longest, y=5,6,4,1,2,3; start with 5's, cells = 1, possible values = 2, add 6's, cells = 2, possible values = 2, cells count === possible values count; remove possible values 1 and 3 from the rest of the cells; return
- For x = 4, max possible values = 6 - 1 = 5 5; start from the shortest y = 6, cells = 2, possible values = 2 (<5), add 1's, cells = 4, possible values = 4 (<5), add 3's, cells = 3, possible values = 5 (<5? False) return
- For y = 6, first x = 1, cells = 1, p = 2; add 4's, cells = 2, p = 3, (1,2,4), add 6's, cells = 3, p = 5 (1,2,3,4)… return; the correct way is to do 2,3,4 to make (2,4,5)

6	**{1,5}**	**{1,2,3,4,5}**	**{1,2,3,4}**	**{1,2,3,4}**	**{1,2,3,4,5}**
{1,3,5}	**{1,5}**	**{1,2,3,4,5,6}**	**{1,2,3,4,6}**	**{1,2,4,6}**	**{1,2,4,5,6}**
{1,2,4,5}	**3**	**{1,2,4,5,6}**	**{1,2,4,6}**	**{1,2,4,5,6}**	**{1,2,4,5,6}**
{1,2,3,4,5}	**{1,2,4,5,6}**	**{1,2,3,4,5,6}**	**{1,2,3,4,6}**	**{1,2,3,6}**	**{2,3,4,6}**
{1,4}	**{1,2,6}**	**{2,3,6}**	**5**	**{1,2,3,6}**	**{1,3}**
{1,4}	**{2,4,5}**	**{2,4,5}**	**{2,4}**	**{1,2,3,6}**	**{1,3}**

- Make row and column scrubbing into formula
  - Scrub row y => formula result = y, symbol = 'r', coordinates = ''
  - Scrub column x => formula result = x, symbol = 'c', coordinates = ''
  - Make a cell to be able to point to several formulas

Add sudoku puzzle solving

Tuesday, November 2, 2021 1:36 PM

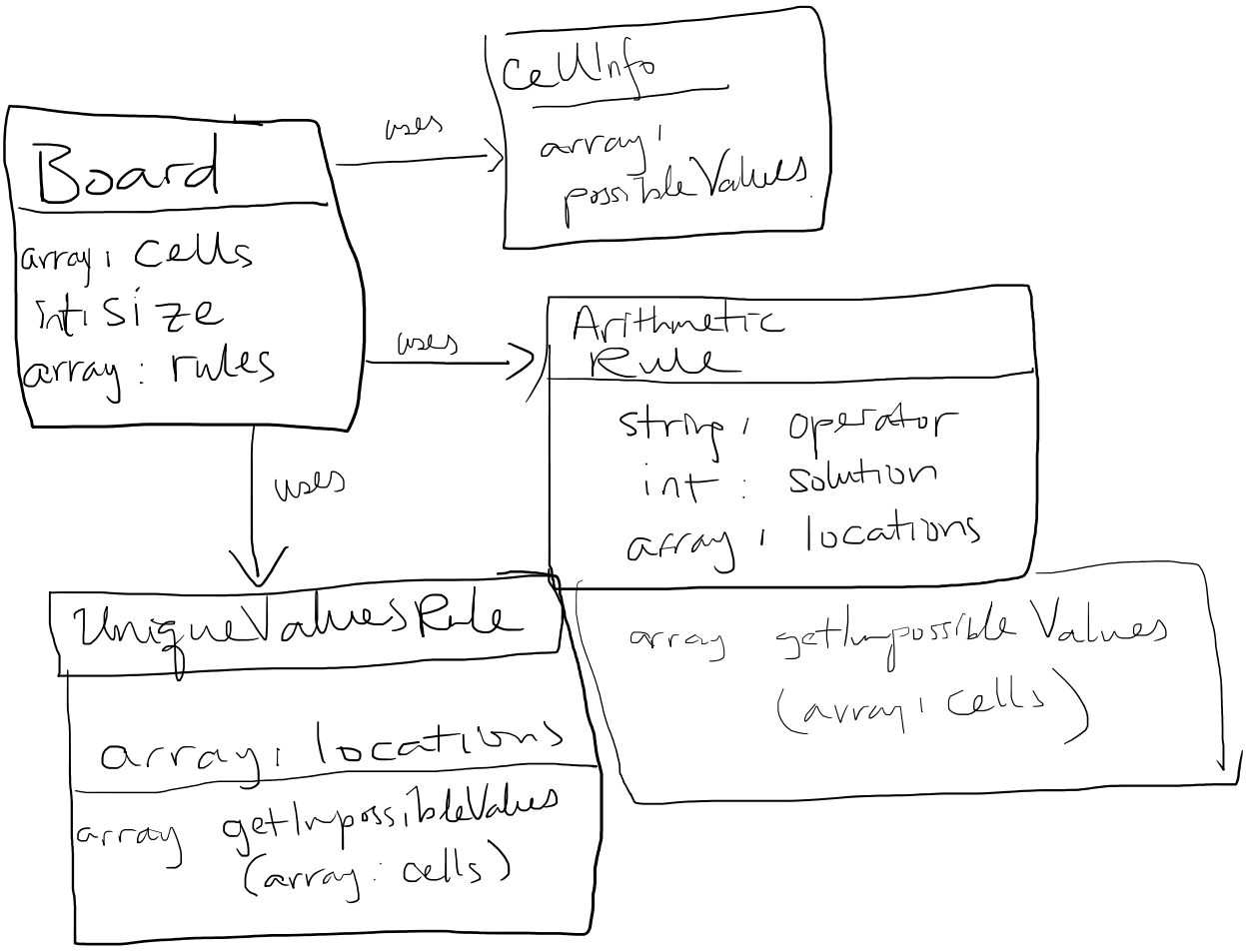
Sudoku is similar to KenDoku. It has a size of 9x9. Each row, column and 3x3 squares need to have unique integer values of 1-9.

Architecture changes to allow to add sudoku puzzle solving:

- [x] Separate arithmetic rule and unique value rules into their own classes since they're not the same:
  - Four function formula: example 5 + a,b means the sum of a,b is 5; cells
    - Only 1,2,3,4 can be the values of a or b.
    - Currently a and b's possible values are 1-9
    - So cells' possible values will change; when a cell's possible values change, this should trigger business logic to inspect the change to decide what to do next: if # of possible values = 1, then trigger set value; if change is related to a formula, then need to evaluate formula
- [x] Board should set value and possible values, not rules. Rule can evaluate cells against itself, then return a list of possible values to remove for its coordinates. (Inversion of control, higher class calls lower class).
  - [x] Arithmetic rules
  - [x] Unique values rules
- [x] Make Board's cells private
- [x] Board needs to create unique value rules for each cluster of cells (every row and column)
- [x] A cell should not track its rules
- [x] Correct board setup cells[row][col] instead of cells[x = col][y=row]
- [x] Remove value property from cell info
- [x] Add rule: when a cell's value is set, then its cell cluster should not have its value listed in their possible values
  - [x] Current rule.coordinates only gives cell location but hard to access and find cell info. Make it into a hash table with key = JSON.stringify(coordinate) and value = cellInfo to make it easier to search?
- [x] Board binds cells to their rules via a hash where key = JSON.stringify(cell's coordinate) and value = cell's value
- [x] Move removePossibleValues from Table to Cell

Model Architecture

Thursday, November 4, 2021 9:41 AM



Add doku puzzle editor and loader

Tuesday, November 2, 2021 1:37 PM

- [x] Architecture considerations to allow for multiple views
    - Model is Board, View is new class View and Controller is new class Controls
    - Investigate reactJS
- [ ] Add a view for sudoku board
    - [ ] Investigate REACT for frontend support
- [ ] A cell should not track its view elements
- [ ] Fix formula view of sudoku board
- [ ]

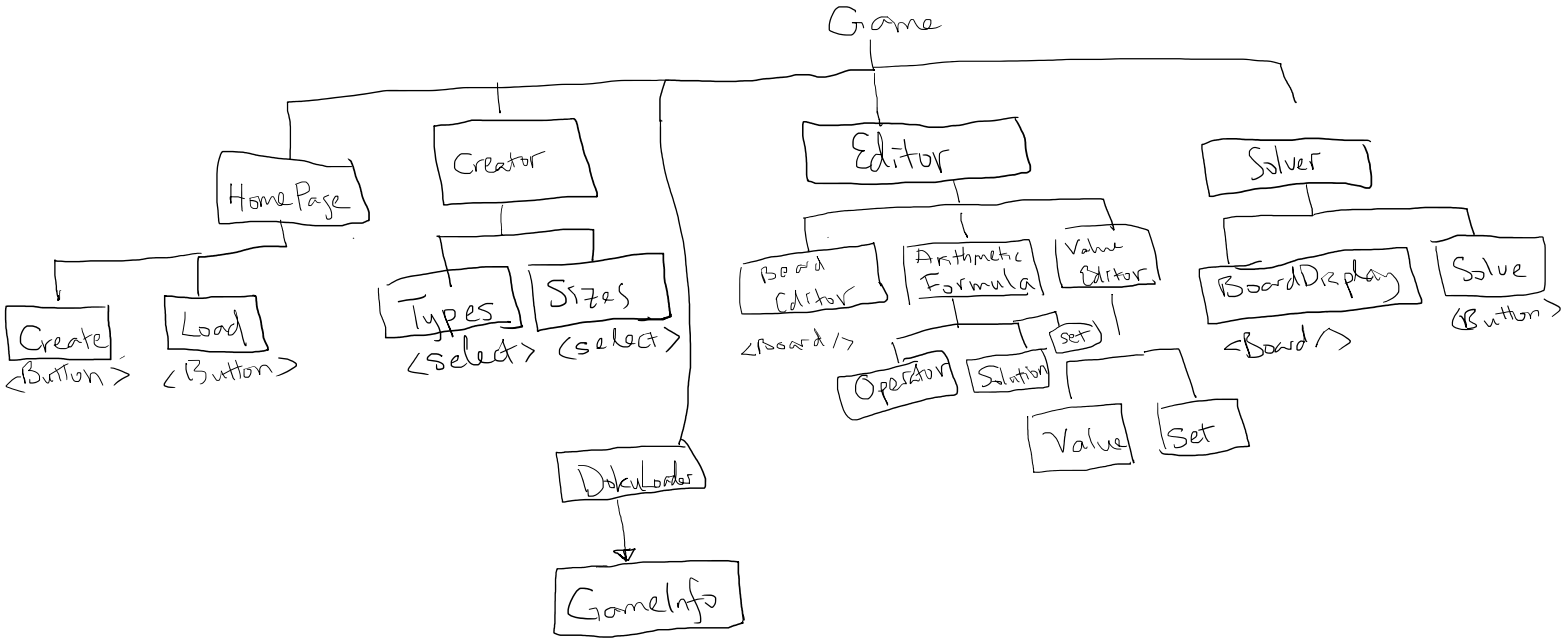
Add ability to load and save game

Wednesday, November 3, 2021 3:56 PM

UI via React

Thursday, November 4, 2021 9:11 AM

COMPONENTS



- Home page has options to create or load a doku game
- Loader lets user select from the doku repository
- Creator lets user select the type of game and the sizes (if KenDoku)
- Editor lets user create a new doku game and save to the repository
- Solver solves the doku game

Home Page

Thursday, November 4, 2021 9:46 AM

Welcome to DokuSolver

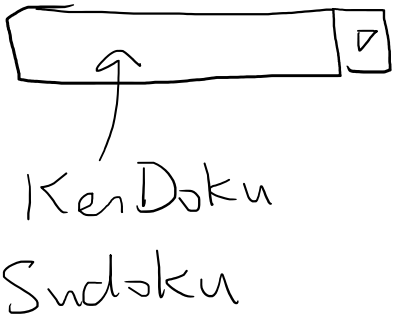
Load existing game

Create new game

Game Editor Initial

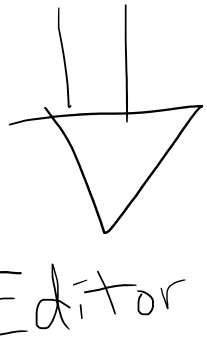
Wednesday, November 3, 2021 3:44 PM

Choose type of game to create:



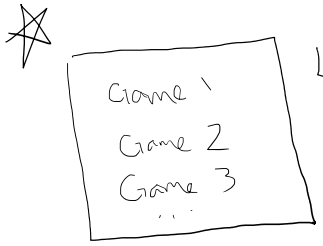
If KenDoku is chosen, then show

Choose size:



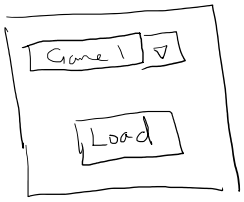
Game loader

Thursday, November 4, 2021 9:18 AM



Links?

OR

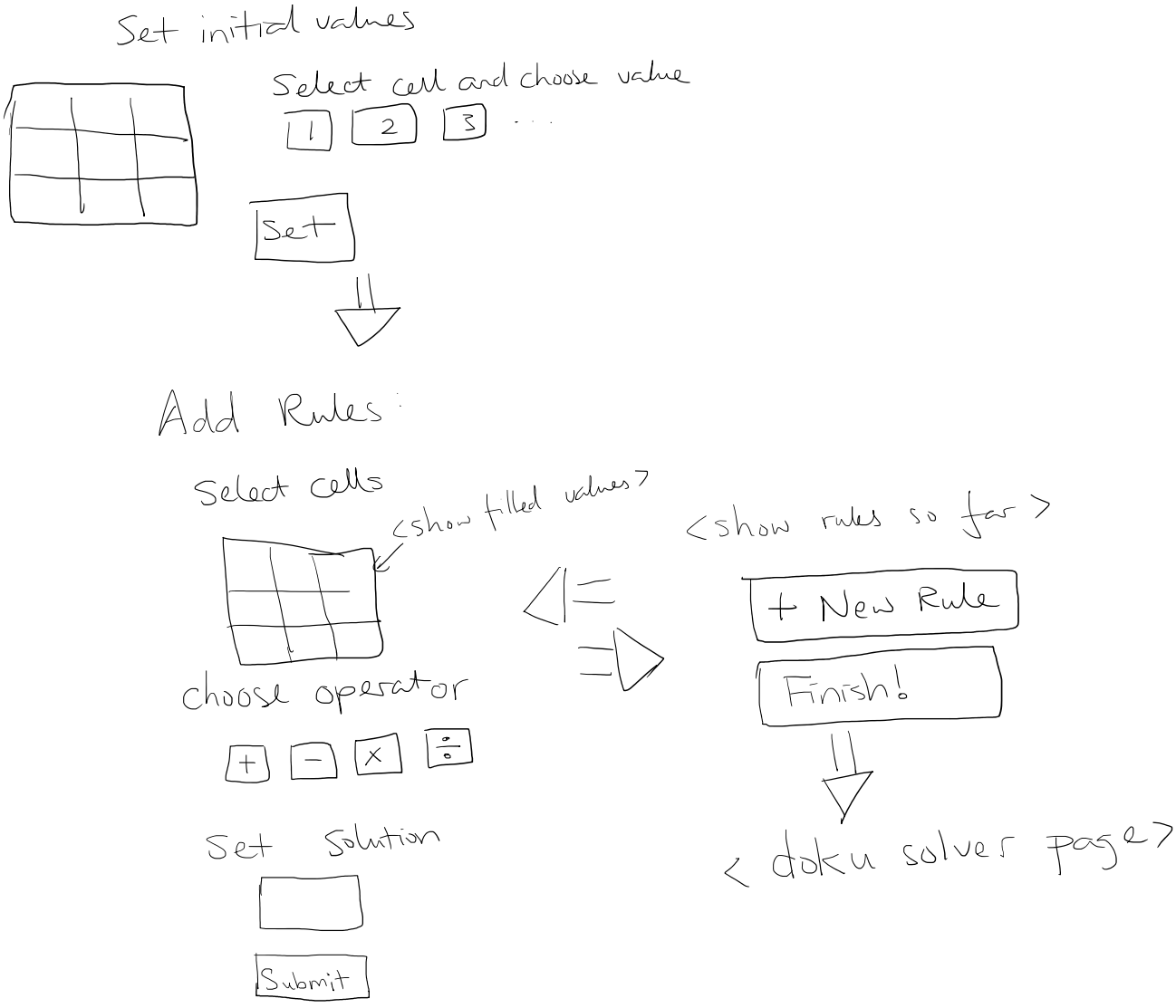


Select & submit

* Easier to read when list gets long.
* Can add description
* Can expand to
  1. Multiple pages
  2. filter according to criteria

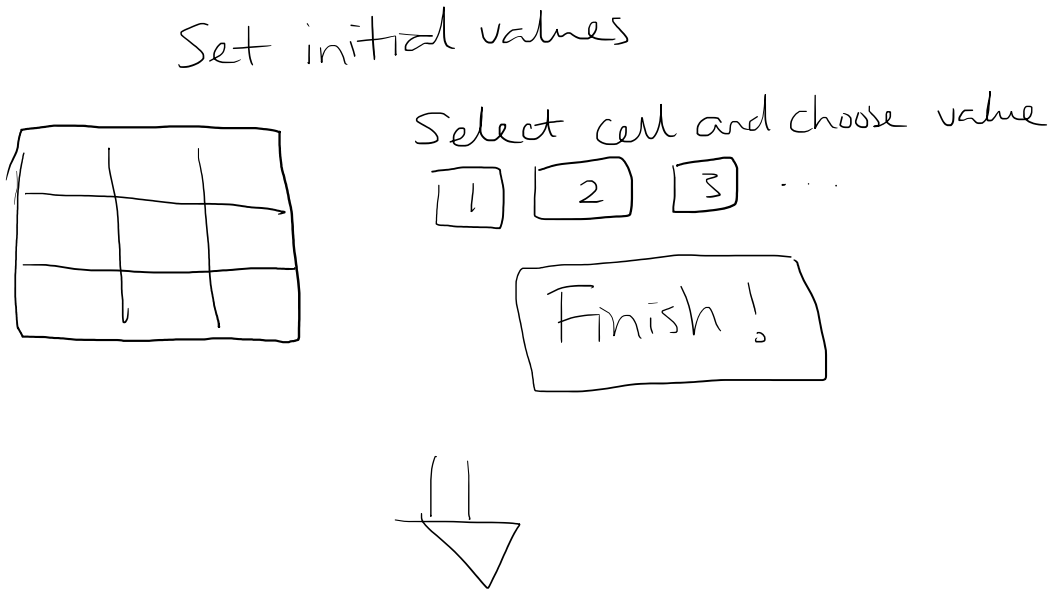
KenDoku editor

Wednesday, November 3, 2021 3:45 PM



Sudoku editor

Wednesday, November 3, 2021 3:52 PM



Puzzle Solver

Wednesday, November 3, 2021 3:55 PM