Thomas Angles

JavaScript Project

Sudoku Board Generator

11/8/2019

**STEPS FOR ALGORITHM**

1. Create multiple 2D arrays for indexing and filling the entire board.\
   1. Array 1: 2D array for holding the final values of the Sudoku Board.
   2. Array2: a 2D array of size 81 each of which has a size of 2. Of which, element 0 is the corresponding Row, and element 1 is the corresponding Column.
2. Call my insert number function.
   1. This function goes through the entire board and initializes each space with a random number for the board positions array to act as the position in the 2nd array from step 1. Then, it makes 2 variables: rowCheck and colCheck.
   2. After each value is checked, the Sudoku board at the rowCheck colCheck is set to the checked Value.
   3. After the valid value is added to the Sudoku Board, the corresponding spot from the Board Positions array is removed so it does not check it again.
   4. It does this for 81 values.
3. Check location which takes in a row and col variable.
   1. Makes an array from 1 to 9. For keeping track of which numbers are not used in each row, column, and 3x3 block.
   2. Row Check
      1. Loop through all values from passed in row.
      2. If it is a valid number (not 0 or ‘-’), get the index and remove it from the array.
   3. Column check
      1. Loop through all values from passed in Column.
      2. If it is a valid number (not 0 or ‘-’), get the index and remove it from the array.
   4. Block check
      1. Initialize two new variables rowBlock, and colBlock. These values are set to the corresponding block from the passed in row and column.
      2. Loop through all values from passed in the Block.
      3. If it is a valid number (not 0 or ‘-’), get the index and remove it from the array.
   5. Return a random value from the remaining array to be used in step 2b.
4. Sudoku Board is the logged in the console as a JavaScript Table.
5. Use toHTML function to conver JavaScript Table to an HTML Table.

***Index.html***

<!DOCTYPE html>

<html lang="en">

<head>

<link rel="stylesheet" href="sudoku\_style.css">

</head>

<body>

<h1>

<span class ="Title">Thomas Angles' Sudoku Board Generator!</span>

</h1>

<h2 class="description"> For This Project, I made a Sudoke Puzzle generator.

This generator works in many steps.

</h2>

<h3>For those interested more in the steps, I have typed out the basic algorithm used.

I will add the instruction set used in this program to the GitHub Repo

</h3>

<script src = "Sudoku\_generator.js"></script>

</body>

</html>

***Sudoku\_style.css***

h1 {

background: linear-gradient(to right, orange , yellow, green, cyan, blue, violet);

-webkit-background-clip: text;

-webkit-text-fill-color: transparent;

font-size: 60px;

line-height: 60px;

}

tr:hover{background-color: limegreen;}

tr { vertical-align: baseline;}

td{

color: Green;

}

table {

width:50%;

height:50%;

float:center;

border:50px Red;

background-color: papayawhip;

}

***Sudoku\_generator.js***

let sudokuBoard = Array(9);

let boardPositions = Array(9);

//Make Board

function createSudokuBoard() {

for (let i = 0; i < 9; i++) {

sudokuBoard[i] = Array(9);

}

for (let i = 0; i < 81; i++) {

boardPositions[i] = Array(2);

}

let positionCount = 0;

while (positionCount < 81) {

for (let i = 0; i < 9; i++) {

let arrayCount = 0;

while (arrayCount < 9) {

boardPositions[positionCount][0] = i;

boardPositions[positionCount][1] = arrayCount;

positionCount++;

arrayCount++;

}

}

}

insertNumber();

}

//INSERT

function insertNumber() {

let i = 0;

while (i < 81) {

let location = boardPositions[(Math.floor(Math.random() \* boardPositions.length))];

let rowCheck = location[0];

let colCheck = location[1];

let value = checkLocation(rowCheck, colCheck);

sudokuBoard[rowCheck][colCheck] = value;

if (!value) {

sudokuBoard[rowCheck][colCheck] = '-';

}

let indexCoordinate = boardPositions.indexOf(location);

boardPositions.splice(indexCoordinate, 1);

i++;

}

}

//LOCATION

function checkLocation(row, column) {

let choice = [];

let indexElement;

let end;

for (let i = 0; i < 9; i++) {

choice.push(i + 1);

}

//THROUGH ROW

for (let i in sudokuBoard[row]) {

if (sudokuBoard[row][i]) {

indexElement = choice.indexOf(sudokuBoard[row][i]);

choice.splice(indexElement, 1);

}

}

//THROUGH COL

for (let j in sudokuBoard) {

if (sudokuBoard[j][column]) {

indexElement = choice.indexOf(sudokuBoard[j][column]);

choice.splice(indexElement, 1);

}

}

let rowBlock = Math.floor(row / 3) \* 3;

let colBlock = Math.floor(column / 3) \* 3;

for (let i = rowBlock; i < rowBlock + 3; i++) {

for (let j = colBlock; j < colBlock + 3; j++) {

if (sudokuBoard[i][j]) {

indexElement = choice.indexOf(sudokuBoard[i][j]);

choice.splice(indexElement, 1);

}

}

}

end = choice[(Math.floor(Math.random() \* choice.length))];

return end;

}

//TO HTML

function toHTML(t) {

let tmp = "<table border=1>";

for (let i = 0; i < t.length; i++) {

tmp += "<tr>";

for (let j = 0; j < t[i].length; j++) {

tmp += "<td>" + t[i][j] + "</td>";

}

tmp += "</tr>";

}

tmp += "</table>";

return tmp;

}

createSudokuBoard();

console.table(sudokuBoard);

document.write(toHTML(sudokuBoard));