```
/*  "15 puzzle", expanded to be SIZE-squared puzzle.
     Written by Claude Anderson for CSSE 280.  April 11, 2013
     Revised October 2, 2015
          renamed BOARD_SIZE to BOARD_WIDTH
          Changed names of parameters to swapDomElements
          Enhanced comments throughout the file.
*/
(function () {
    "use strict";

    // CONSTANTS
    var SIZE;  // 4 for 16 puzzle, 5 for 25 puzzle, etc.
    var NUM_SQUARES;
    var BOARD_WIDTH = 500;  // 500 pixels
    var BORDER_SIZE = 3;    // 3 pixels
    var SQUARE_SIZE;

    // OTHER VARS
    var tileList = [];
    var spacePos;
    var emptySpace = null;
    var minSwaps;
    var maxSwaps;
    var gameStarted = false;


    function getSize() {
        var radios = document.getElementsByName("size");
        for (var i = 0; i < radios.length; i++) {
            if (radios[i].checked) {
                SIZE = parseInt(radios[i].value);
            }
        }
        NUM_SQUARES = SIZE * SIZE;
        SQUARE_SIZE = parseInt(BOARD_WIDTH - (SIZE) * (BORDER_SIZE * 2)) / SIZE;
        minSwaps = NUM_SQUARES;
        maxSwaps = NUM_SQUARES * 20;
        setupBoard();
    }

    // called when the page first loads to create tiles and empty space
    function setup() {
        document.getElementById("sizeChoice").onclick = getSize;
    }

    function setupBoard() {
        gameStarted = false;
        tileList = [];
        var boardDiv = document.getElementById("board");
        boardDiv.innerHTML = " ";
        boardDiv.style.width = BOARD_WIDTH + "px";

        for (var i = 1; i <= NUM_SQUARES; i++) {
            var nodeName = i;
            if (i == NUM_SQUARES) {
                nodeName = "space";
            }
            var newSpan = document.createElement("span");
            newSpan.className = "tile";
            newSpan.id = nodeName;
            newSpan.textContent = nodeName;
            newSpan.style.width = SQUARE_SIZE + "px";
            newSpan.style.height = SQUARE_SIZE + "px";
            newSpan.style.lineHeight = SQUARE_SIZE + "px";
            newSpan.style.border = BORDER_SIZE + "px solid red";
            if (i === NUM_SQUARES) {
                newSpan.classList.add("space");  // class is now "tile space"
                newSpan.innerHTML = "&nbsp;";
                spacePos = NUM_SQUARES - 1;
                emptySpace = newSpan;
            }
            tileList.push(newSpan);        // Add tile at end of list
            boardDiv.appendChild(newSpan); // Add to the DOM
            newSpan.onclick = moveTile;    // Make it respond to clicks.
        }
        shuffleTiles();
        gameStarted = true;
    }

    function shuffleTiles() {
        // Shuffle the tiles
        var numSwaps = parseInt(minSwaps + Math.random() * (maxSwaps - minSwaps));
        var swapCount = 0;
        while (swapCount < numSwaps) {
            var index = parseInt(Math.random() * NUM_SQUARES);
            swapCount += moveTile.call(tileList[index]);  // tileList[index] will be "this" inside moveTile.
        }
    }
```

```javascript

    function swapArrayElements(a, p1, p2) {
        var temp = a[p1];
        a[p1] = a[p2];
        a[p2] = temp;
    }

    // If clicked tile is next to the empty space, swap them.
    function moveTile() {
        var pos = tileList.indexOf(this);
        var diff = Math.abs(pos - spacePos);
        if (diff == 1 && sameRow(pos, spacePos, SIZE) || diff == SIZE) {
            swapDomElements(this, emptySpace);
            swapArrayElements(tileList, pos, spacePos);
            spacePos = pos;
            if (isSolved()) {
                setTimeout(function () { alert("You have won"); }, 100);
            }
            return 1;
        } else {
            return 0;
        }
    }

    function sameRow(pos1, pos2, rowSize) {
        return rowNumber(pos1, rowSize) == rowNumber(pos2, rowSize);
    }

    function rowNumber(pos, rowSize) {
        return parseInt(pos / rowSize);
    }

    function isSolved() {
        if (!gameStarted) {
            return false;
        }
        var value = 0;
        for (var i = 0; i < tileList.length - 1; i++) {
            var tmp = parseInt(tileList[i].textContent);
            if (Number.isNaN(tmp) || tmp <= value) {
                return false;
            } else {
                value = tmp;
            }
        }
        return true;
    }

    // Exchange the locations of two elements in the DOM.
    // Assumes that neither element is the parent of the other.
    // from
http://stackoverflow.com/questions/10716986/swap-2-html-elements-and-preserve-
event-listeners-on-them
    function swapDomElements(element1, element2) {
        // create marker element and insert it where element1 is
        var temp = document.createElement("div");

        // insert temp before element1
        element1.parentNode.insertBefore(temp, element1);

        // move element1 to immediately before element2
        element2.parentNode.insertBefore(element1, element2);

        // move element2 to immediately before where element1 used to be
        temp.parentNode.insertBefore(element2, temp);

        // remove temporary marker node
        temp.parentNode.removeChild(temp);
    }
    window.onload = setup;
})();
```