Project 2: Puzzle!

Ryan Garland and Sam Cooney

Outline

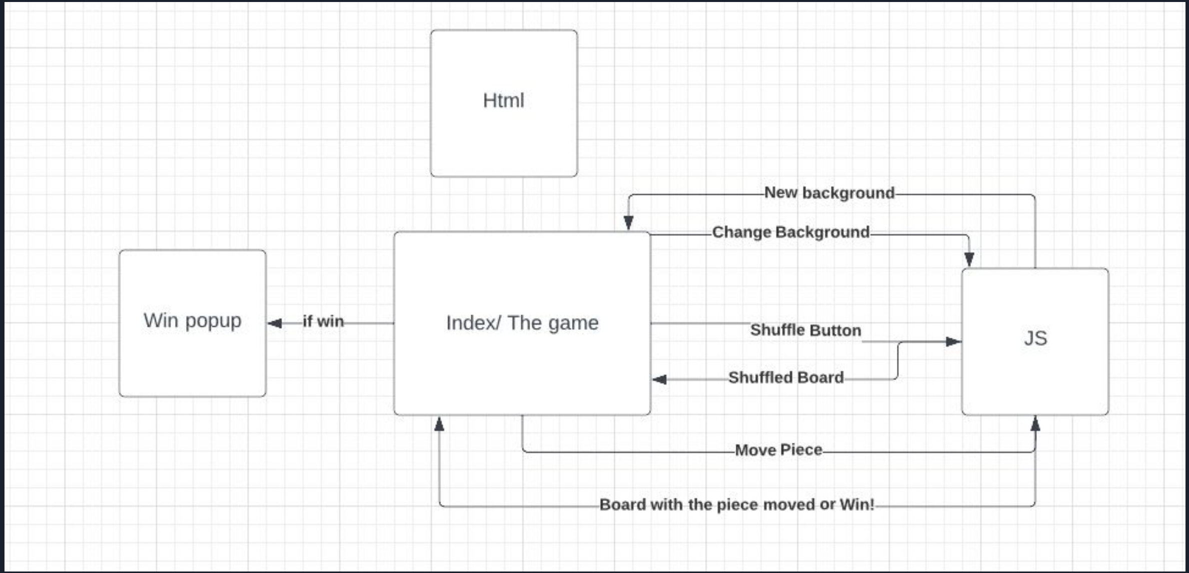
- Introduction
  - What extra features we chose
  - User
  - Design
  - Testing
- Demo
- Show source code

User: How do you make a 15 Slide Puzzle?

- Problem:
  - Create a puzzle game
  - Add the following extra features:
    - End-of-game notification
    - Multiple backgrounds
    - Extra Animation (after winning)
    - Game time with some music file
- Inputs
  - User shuffles the board
  - User clicks movable pieces to solve the game
- Outputs
  - Current board state
  - Time elapsed
  - Number of moves made
  - Win screen

SCRUM and UML

- Benefits
  - Allows to the plan out the roadmap for the project
  - Allowed us to reformat certain elements
  - Allowed to split work evenly

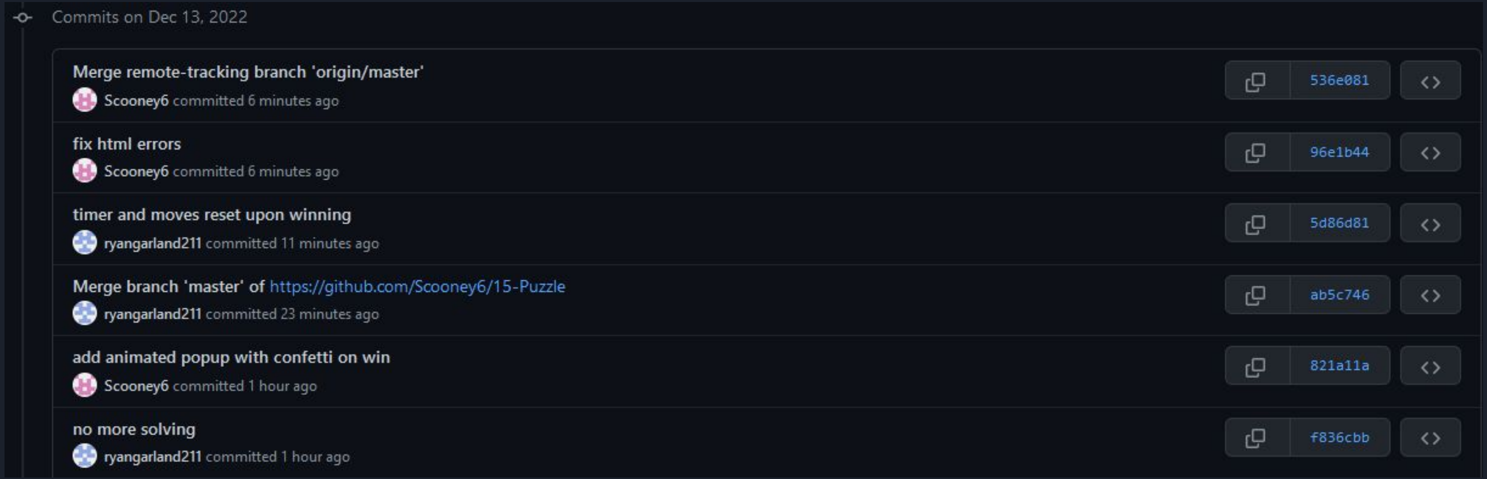


Design: Flow of the game

1. Index/Game Screen
    a. Contains buttons to change background, and shuffle the board
    b. Shows current board state
    c. If the player won, a window will pop up saying a message, move count and time elapsed
2. Javascript
    a. Handles shuffle logic
    b. Handles moving pieces and updating the board
    c. Handles changing the background
    d. Keeps and returns time elapsed
    e. Keeps and returns moves made

Testing

- Tested Features after implementation
- Example tests:
  - Checking if shuffle resets the timer and moves
  - Check to see if win pop up comes after winning
  - Time can increase past 60 seconds
  - Game can be reshuffled after winning



Commits on Dec 13, 2022

Merge remote-tracking branch 'origin/master'
Scooney6 committed 6 minutes ago — 536e081

fix html errors
Scooney6 committed 6 minutes ago — 96e1b44

timer and moves reset upon winning
ryangarland211 committed 11 minutes ago — 5d86d81

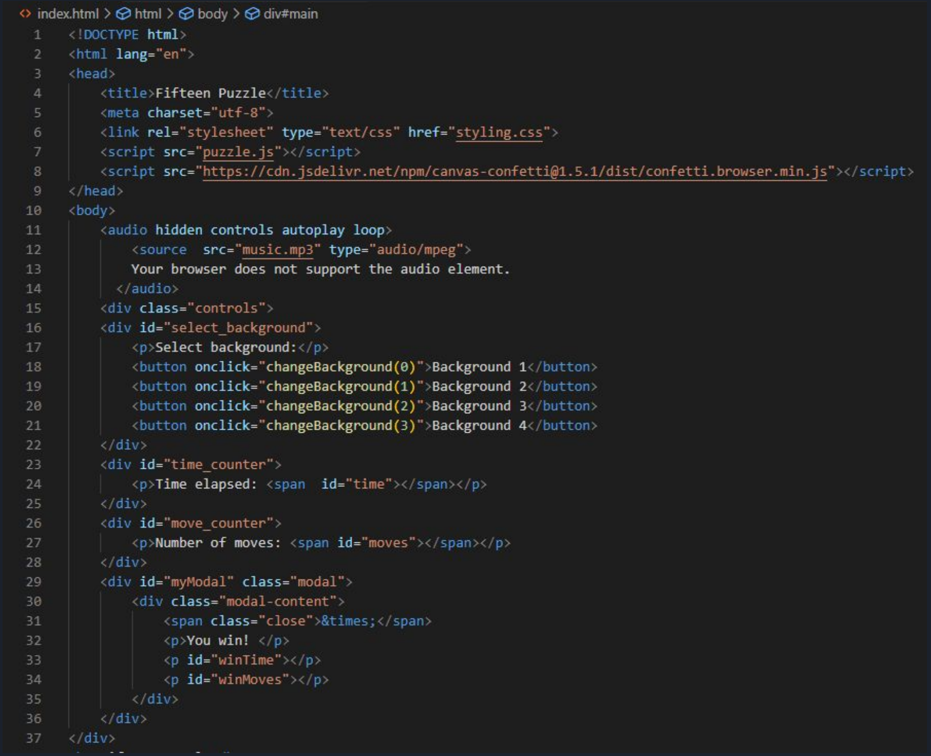
Merge branch 'master' of https://github.com/Scooney6/15-Puzzle
ryangarland211 committed 23 minutes ago — ab5c746

add animated popup with confetti on win
Scooney6 committed 1 hour ago — 821a11a

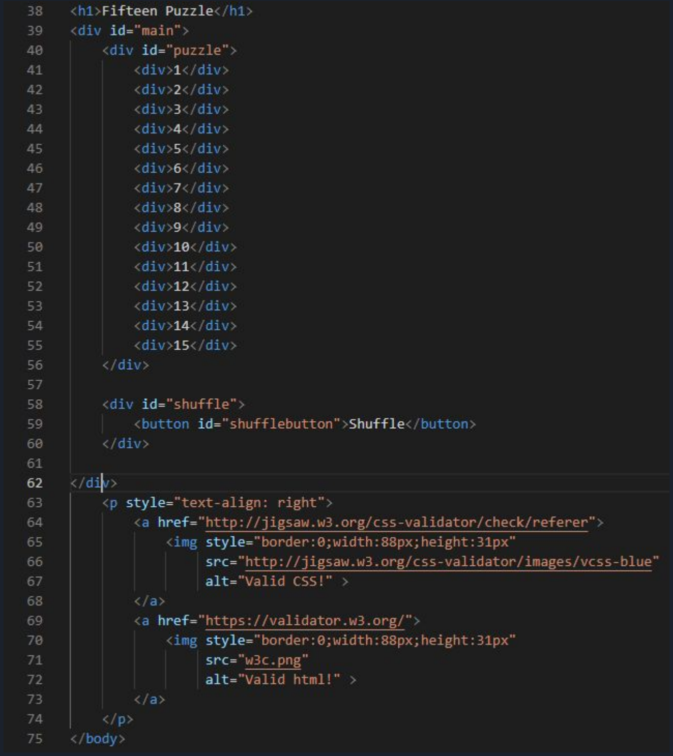
no more solving
ryangarland211 committed 1 hour ago — f836cbb

Demo

The Code: index.html

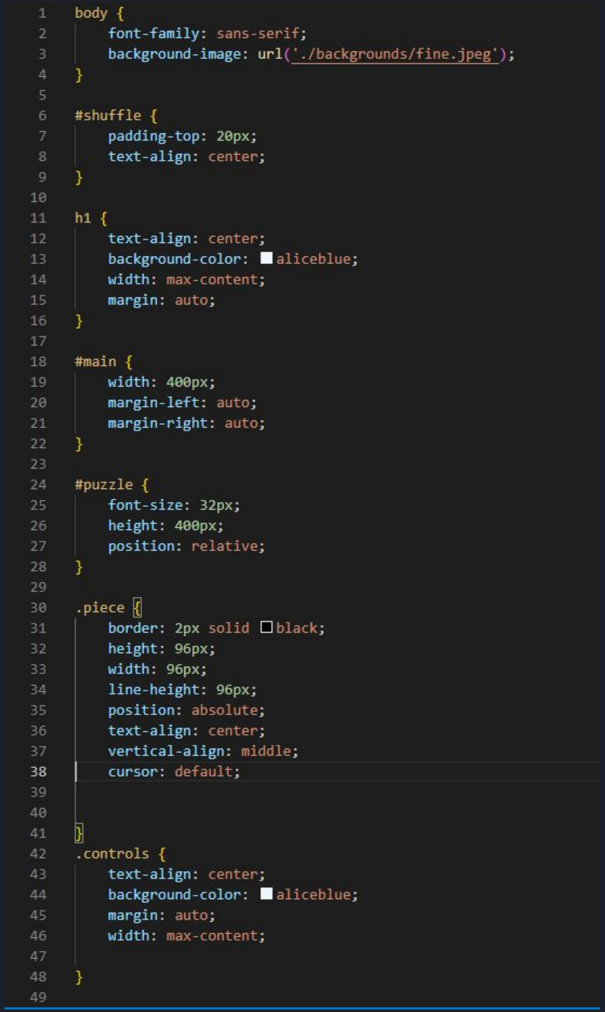


```html
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Fifteen Puzzle</title>
    <meta charset="utf-8">
    <link rel="stylesheet" type="text/css" href="styling.css">
    <script src="puzzle.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/canvas-confetti@1.5.1/dist/confetti.browser.min.js"></script>
</head>
<body>
    <audio hidden controls autoplay loop>
        <source  src="music.mp3" type="audio/mpeg">
        Your browser does not support the audio element.
      </audio>
    <div class="controls">
    <div id="select_background">
        <p>Select background:</p>
        <button onclick="changeBackground(0)">Background 1</button>
        <button onclick="changeBackground(1)">Background 2</button>
        <button onclick="changeBackground(2)">Background 3</button>
        <button onclick="changeBackground(3)">Background 4</button>
    </div>
    <div id="time_counter">
        <p>Time elapsed: <span  id="time"></span></p>
    </div>
    <div id="move_counter">
        <p>Number of moves: <span id="moves"></span></p>
    </div>
    <div id="myModal" class="modal">
        <div class="modal-content">
            <span class="close">&times;</span>
            <p>You win! </p>
            <p id="winTime"></p>
            <p id="winMoves"></p>
        </div>
    </div>
</div>
```



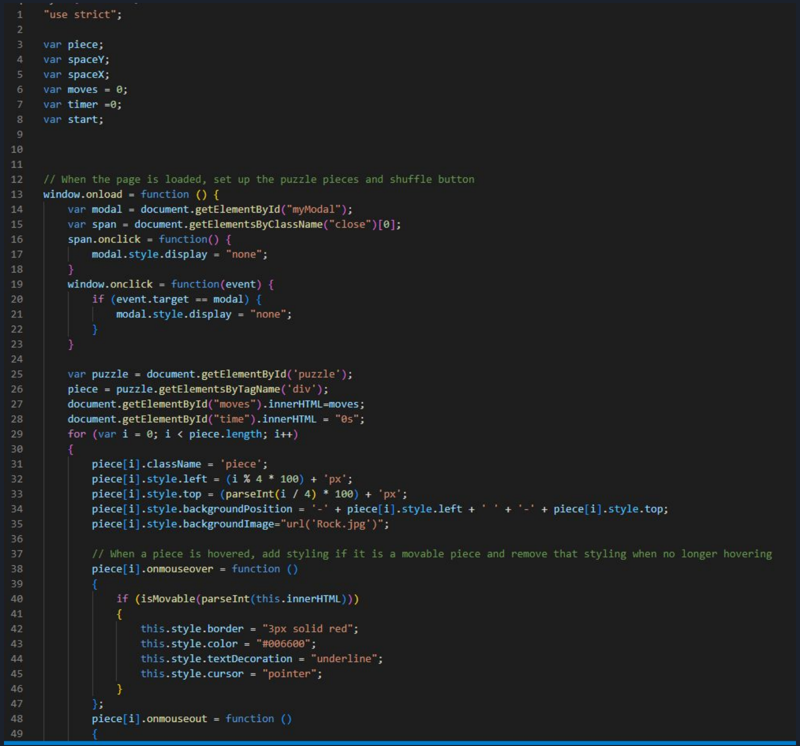
```html
<h1>Fifteen Puzzle</h1>
<div id="main">
    <div id="puzzle">
        <div>1</div>
        <div>2</div>
        <div>3</div>
        <div>4</div>
        <div>5</div>
        <div>6</div>
        <div>7</div>
        <div>8</div>
        <div>9</div>
        <div>10</div>
        <div>11</div>
        <div>12</div>
        <div>13</div>
        <div>14</div>
        <div>15</div>
    </div>

    <div id="shuffle">
        <button id="shufflebutton">Shuffle</button>
    </div>

</div>
    <p style="text-align: right">
        <a href="http://jigsaw.w3.org/css-validator/check/referer">
            <img style="border:0;width:88px;height:31px"
                 src="http://jigsaw.w3.org/css-validator/images/vcss-blue"
                 alt="Valid CSS!" >
        </a>
        <a href="https://validator.w3.org/">
            <img style="border:0;width:88px;height:31px"
                 src="w3c.png"
                 alt="Valid html!" >
        </a>
    </p>
</body>
```

The Code: Styling

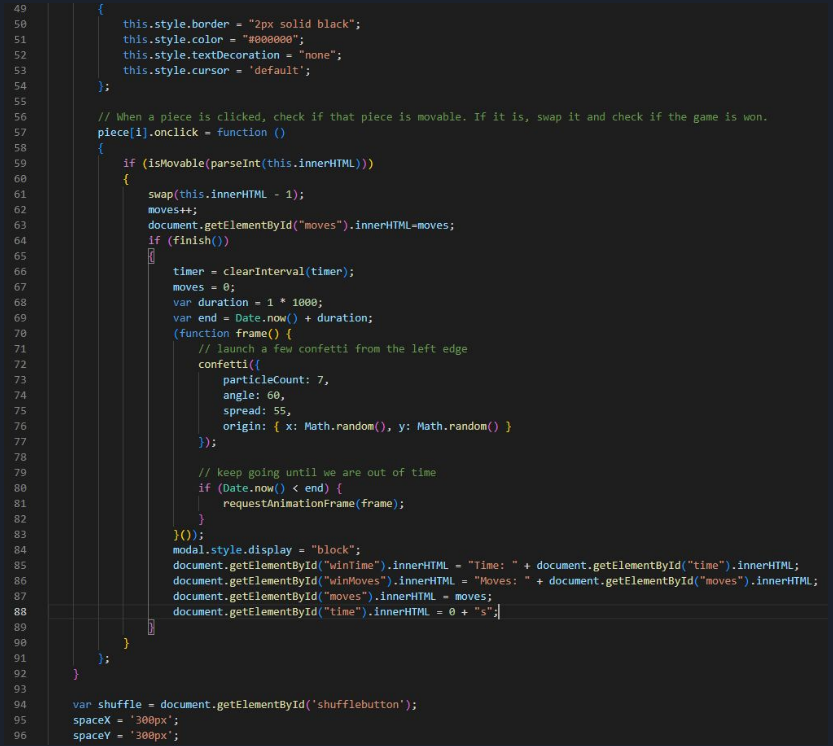


```css
body {
    font-family: sans-serif;
    background-image: url('./backgrounds/fine.jpeg');
}

#shuffle {
    padding-top: 20px;
    text-align: center;
}

h1 {
    text-align: center;
    background-color: aliceblue;
    width: max-content;
    margin: auto;
}

#main {
    width: 400px;
    margin-left: auto;
    margin-right: auto;
}

#puzzle {
    font-size: 32px;
    height: 400px;
    position: relative;
}

.piece {
    border: 2px solid black;
    height: 96px;
    width: 96px;
    line-height: 96px;
    position: absolute;
    text-align: center;
    vertical-align: middle;
    cursor: default;


}
.controls {
    text-align: center;
    background-color: aliceblue;
    margin: auto;
    width: max-content;

}

```

The Code: puzzle.js

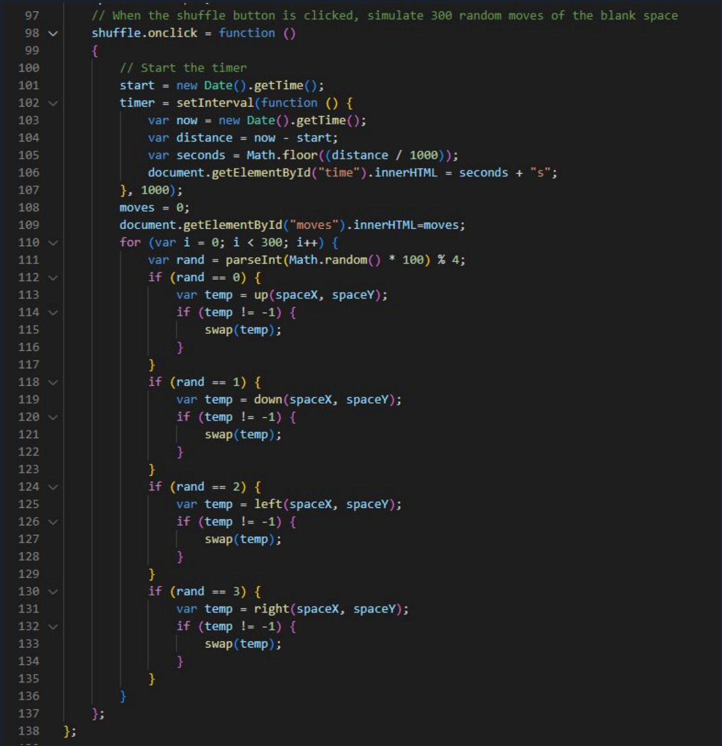


```javascript
"use strict";

var piece;
var spaceY;
var spaceX;
var moves = 0;
var timer =0;
var start;



// When the page is loaded, set up the puzzle pieces and shuffle button
window.onload = function () {
    var modal = document.getElementById("myModal");
    var span = document.getElementsByClassName("close")[0];
    span.onclick = function() {
        modal.style.display = "none";
    }
    window.onclick = function(event) {
        if (event.target == modal) {
            modal.style.display = "none";
        }
    }

    var puzzle = document.getElementById('puzzle');
    piece = puzzle.getElementsByTagName('div');
    document.getElementById("moves").innerHTML=moves;
    document.getElementById("time").innerHTML = "0s";
    for (var i = 0; i < piece.length; i++)
    {
        piece[i].className = 'piece';
        piece[i].style.left = (i % 4 * 100) + 'px';
        piece[i].style.top = (parseInt(i / 4) * 100) + 'px';
        piece[i].style.backgroundPosition = '-' + piece[i].style.left + ' ' + '-' + piece[i].style.top;
        piece[i].style.backgroundImage="url('Rock.jpg')";

        // When a piece is hovered, add styling if it is a movable piece and remove that styling when no longer hovering
        piece[i].onmouseover = function ()
        {
            if (isMovable(parseInt(this.innerHTML)))
            {
                this.style.border = "3px solid red";
                this.style.color = "#006600";
                this.style.textDecoration = "underline";
                this.style.cursor = "pointer";
            }
        };
        piece[i].onmouseout = function ()
        {
```

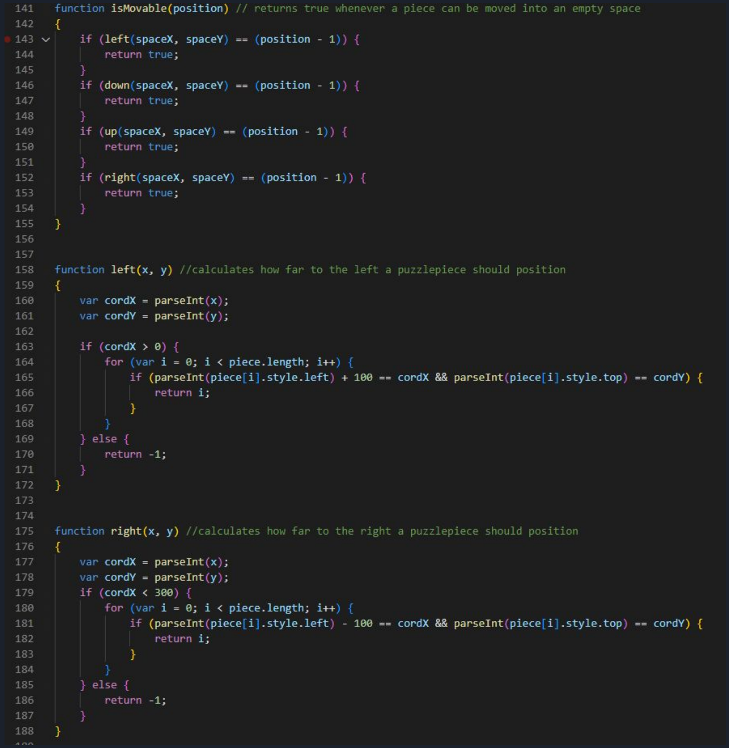


```javascript
        {
            this.style.border = "2px solid black";
            this.style.color = "#000000";
            this.style.textDecoration = "none";
            this.style.cursor = 'default';
        };

        // When a piece is clicked, check if that piece is movable. If it is, swap it and check if the game is won.
        piece[i].onclick = function ()
        {
            if (isMovable(parseInt(this.innerHTML)))
            {
                swap(this.innerHTML - 1);
                moves++;
                document.getElementById("moves").innerHTML=moves;
                if (finish())
                {
                    timer = clearInterval(timer);
                    moves = 0;
                    var duration = 1 * 1000;
                    var end = Date.now() + duration;
                    (function frame() {
                        // launch a few confetti from the left edge
                        confetti({
                            particleCount: 7,
                            angle: 60,
                            spread: 55,
                            origin: { x: Math.random(), y: Math.random() }
                        });

                        // keep going until we are out of time
                        if (Date.now() < end) {
                            requestAnimationFrame(frame);
                        }
                    }());
                    modal.style.display = "block";
                    document.getElementById("winTime").innerHTML = "Time: " + document.getElementById("time").innerHTML;
                    document.getElementById("winMoves").innerHTML = "Moves: " + document.getElementById("moves").innerHTML;
                    document.getElementById("moves").innerHTML = moves;
                    document.getElementById("time").innerHTML = 0 + "s";
                }
            }
        };
    }

    var shuffle = document.getElementById('shufflebutton');
    spaceX = '300px';
    spaceY = '300px';
```

The Code: puzzle.js (cont.)

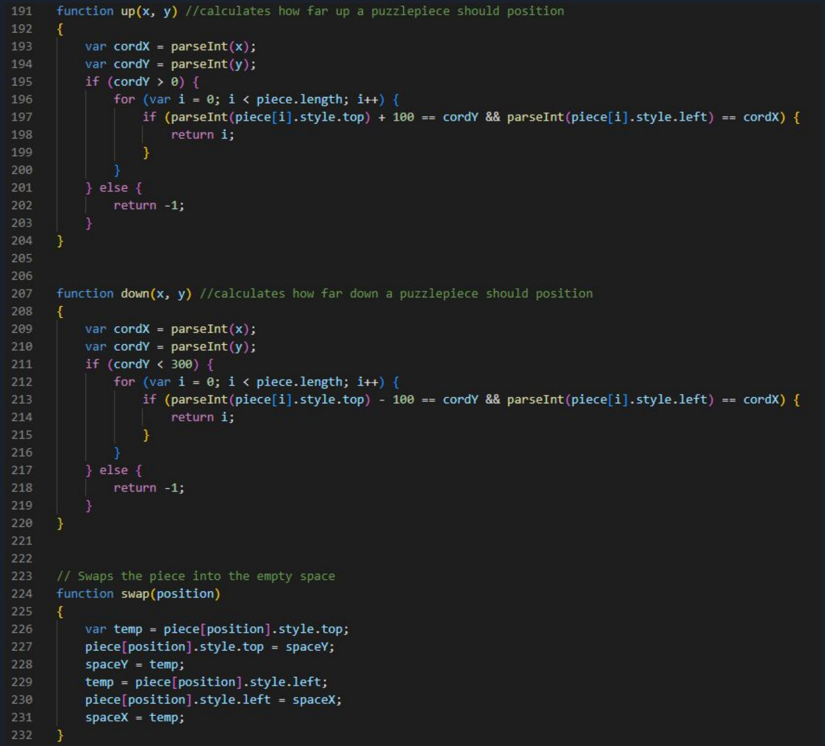


```
    // When the shuffle button is clicked, simulate 300 random moves of the blank space
    shuffle.onclick = function ()
    {
        // Start the timer
        start = new Date().getTime();
        timer = setInterval(function () {
            var now = new Date().getTime();
            var distance = now - start;
            var seconds = Math.floor((distance / 1000));
            document.getElementById("time").innerHTML = seconds + "s";
        }, 1000);
        moves = 0;
        document.getElementById("moves").innerHTML=moves;
        for (var i = 0; i < 300; i++) {
            var rand = parseInt(Math.random() * 100) % 4;
            if (rand == 0) {
                var temp = up(spaceX, spaceY);
                if (temp != -1) {
                    swap(temp);
                }
            }
            if (rand == 1) {
                var temp = down(spaceX, spaceY);
                if (temp != -1) {
                    swap(temp);
                }
            }
            if (rand == 2) {
                var temp = left(spaceX, spaceY);
                if (temp != -1) {
                    swap(temp);
                }
            }
            if (rand == 3) {
                var temp = right(spaceX, spaceY);
                if (temp != -1) {
                    swap(temp);
                }
            }
        }
    };
};
```

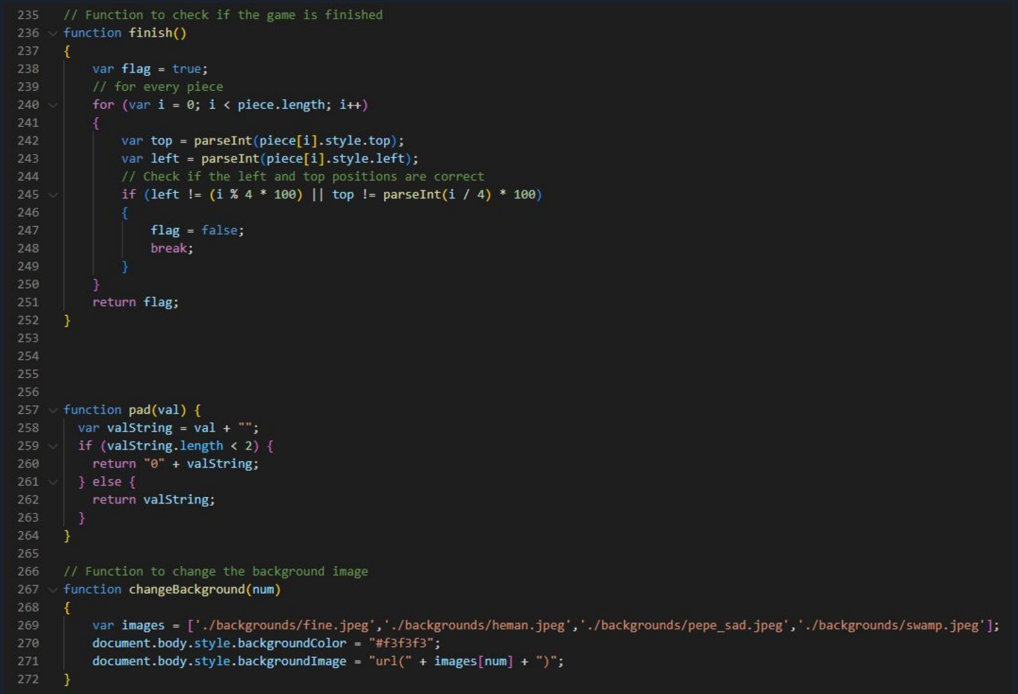


```
function isMovable(position) // returns true whenever a piece can be moved into an empty space
{
    if (left(spaceX, spaceY) == (position - 1)) {
        return true;
    }
    if (down(spaceX, spaceY) == (position - 1)) {
        return true;
    }
    if (up(spaceX, spaceY) == (position - 1)) {
        return true;
    }
    if (right(spaceX, spaceY) == (position - 1)) {
        return true;
    }
}


function left(x, y) //calculates how far to the left a puzzlepiece should position
{
    var cordX = parseInt(x);
    var cordY = parseInt(y);

    if (cordX > 0) {
        for (var i = 0; i < piece.length; i++) {
            if (parseInt(piece[i].style.left) + 100 == cordX && parseInt(piece[i].style.top) == cordY) {
                return i;
            }
        }
    } else {
        return -1;
    }
}


function right(x, y) //calculates how far to the right a puzzlepiece should position
{
    var cordX = parseInt(x);
    var cordY = parseInt(y);
    if (cordX < 300) {
        for (var i = 0; i < piece.length; i++) {
            if (parseInt(piece[i].style.left) - 100 == cordX && parseInt(piece[i].style.top) == cordY) {
                return i;
            }
        }
    } else {
        return -1;
    }
}
```

The Code: puzzle.js (cont.2)



```javascript
function up(x, y) //calculates how far up a puzzlepiece should position
{
    var cordX = parseInt(x);
    var cordY = parseInt(y);
    if (cordY > 0) {
        for (var i = 0; i < piece.length; i++) {
            if (parseInt(piece[i].style.top) + 100 == cordY && parseInt(piece[i].style.left) == cordX) {
                return i;
            }
        }
    } else {
        return -1;
    }
}


function down(x, y) //calculates how far down a puzzlepiece should position
{
    var cordX = parseInt(x);
    var cordY = parseInt(y);
    if (cordY < 300) {
        for (var i = 0; i < piece.length; i++) {
            if (parseInt(piece[i].style.top) - 100 == cordY && parseInt(piece[i].style.left) == cordX) {
                return i;
            }
        }
    } else {
        return -1;
    }
}


// Swaps the piece into the empty space
function swap(position)
{
    var temp = piece[position].style.top;
    piece[position].style.top = spaceY;
    spaceY = temp;
    temp = piece[position].style.left;
    piece[position].style.left = spaceX;
    spaceX = temp;
}
```



```javascript
// Function to check if the game is finished
function finish()
{
    var flag = true;
    // for every piece
    for (var i = 0; i < piece.length; i++)
    {
        var top = parseInt(piece[i].style.top);
        var left = parseInt(piece[i].style.left);
        // Check if the left and top positions are correct
        if (left != (i % 4 * 100) || top != parseInt(i / 4) * 100)
        {
            flag = false;
            break;
        }
    }
    return flag;
}




function pad(val) {
  var valString = val + "";
  if (valString.length < 2) {
    return "0" + valString;
  } else {
    return valString;
  }
}

// Function to change the background image
function changeBackground(num)
{
    var images = ['./backgrounds/fine.jpeg','./backgrounds/heman.jpeg','./backgrounds/pepe_sad.jpeg','./backgrounds/swamp.jpeg'];
    document.body.style.backgroundColor = "#f3f3f3";
    document.body.style.backgroundImage = "url(" + images[num] + ")";
}
```

The End, Thank you!