

A decorative graphic on the left side of the slide consists of two overlapping parallelograms. The front one is blue and the back one is a light green. They are positioned diagonally, with the blue one partially covering the green one.

# Project 2: Puzzle!

Ryan Garland and Sam Cooney



# Outline

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

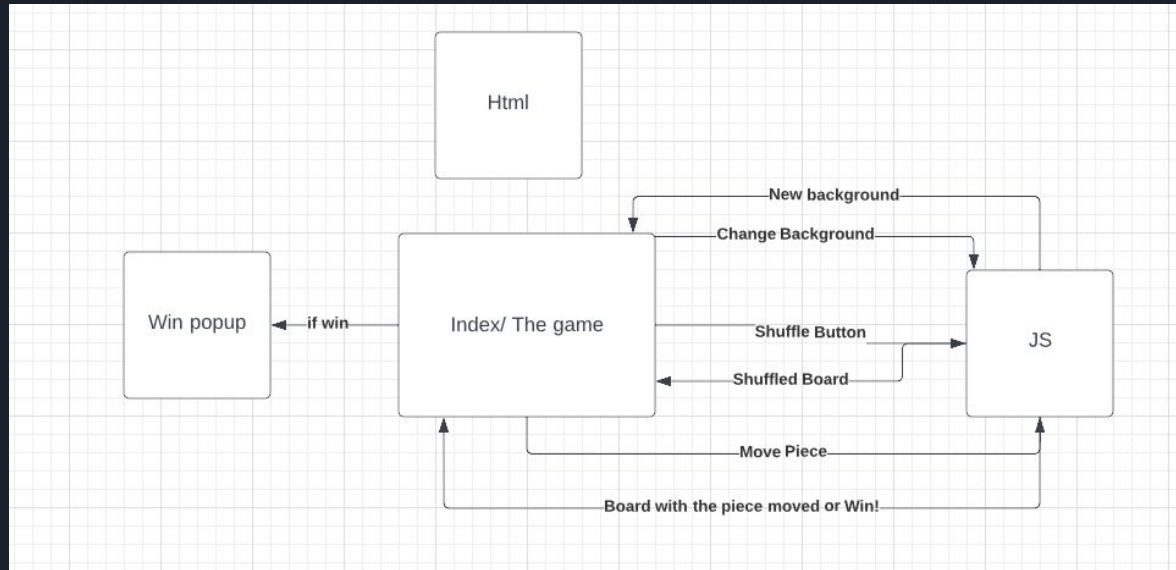


# User: How do you make a 15 Slide Puzzle?

- 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 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'	 536e081	
 Scooney6 committed 6 minutes ago		
fix html errors	 96e1b44	
 Scooney6 committed 6 minutes ago		
timer and moves reset upon winning	 5d86d81	
 ryangarland211 committed 11 minutes ago		
Merge branch 'master' of <a href="https://github.com/Scooney6/15-Puzzle">https://github.com/Scooney6/15-Puzzle</a>	 ab5c746	
 ryangarland211 committed 23 minutes ago		
add animated popup with confetti on win	 821a11a	
 Scooney6 committed 1 hour ago		
no more solving	 f836cbb	
 ryangarland211 committed 1 hour ago		

Demo



# The Code: index.html

```
<? index.html > <? html > <? body > <? div#main
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title>Fifteen Puzzle</title>
5    <meta charset="utf-8">
6    <link rel="stylesheet" type="text/css" href="styling.css">
7    <script src="puzzle.js"></script>
8    <script src="https://cdn.jsdelivr.net/npm/canvas-confetti@1.5.1/dist/confetti.browser.min.js"></script>
9  </head>
10 <body>
11   <audio hidden controls autoplay loop>
12     <source src="music.mp3" type="audio/mpeg">
13     Your browser does not support the audio element.
14   </audio>
15   <div class="controls">
16     <div id="select_background">
17       <p>Select background:</p>
18       <button onclick="changeBackground(0)">Background 1</button>
19       <button onclick="changeBackground(1)">Background 2</button>
20       <button onclick="changeBackground(2)">Background 3</button>
21       <button onclick="changeBackground(3)">Background 4</button>
22     </div>
23     <div id="time_counter">
24       <p>Time elapsed: <span id="time"></span></p>
25     </div>
26     <div id="move_counter">
27       <p>Number of moves: <span id="moves"></span></p>
28     </div>
29     <div id="myModal" class="modal">
30       <div class="modal-content">
31         <span class="close">&times;</span>
32         <p>You win! </p>
33         <p id="winTime"></p>
34         <p id="winMoves"></p>
35       </div>
36     </div>
37 </div>
```

```
38 <h1>Fifteen Puzzle</h1>
39 <div id="main">
40   <div id="puzzle">
41     <div>1</div>
42     <div>2</div>
43     <div>3</div>
44     <div>4</div>
45     <div>5</div>
46     <div>6</div>
47     <div>7</div>
48     <div>8</div>
49     <div>9</div>
50     <div>10</div>
51     <div>11</div>
52     <div>12</div>
53     <div>13</div>
54     <div>14</div>
55     <div>15</div>
56   </div>
57   <div id="shuffle">
58     <button id="shufflebutton">Shuffle</button>
59   </div>
60 </div>
61
62 </div>
63 <p style="text-align: right">
64   <a href="http://jigsaw.w3.org/css-validator/check/referer">
65     
68   </a>
69   <a href="https://validator.w3.org/">
70     
73   </a>
74 </p>
75 </body>
```





# The Code: Styling

```
1 body {
2   font-family: sans-serif;
3   background-image: url('../backgrounds/fine.jpeg');
4 }
5
6 #shuffle {
7   padding-top: 20px;
8   text-align: center;
9 }
10
11 h1 {
12   text-align: center;
13   background-color: ■aliceblue;
14   width: max-content;
15   margin: auto;
16 }
17
18 #main {
19   width: 400px;
20   margin-left: auto;
21   margin-right: auto;
22 }
23
24 #puzzle {
25   font-size: 32px;
26   height: 400px;
27   position: relative;
28 }
29
30 .piece {
31   border: 2px solid □black;
32   height: 96px;
33   width: 96px;
34   line-height: 96px;
35   position: absolute;
36   text-align: center;
37   vertical-align: middle;
38   cursor: default;
39 }
40
41
42 .controls {
43   text-align: center;
44   background-color: ■aliceblue;
45   margin: auto;
46   width: max-content;
47 }
48
49
```

# The Code: puzzle.js

```
1 "use strict";
2
3 var piece;
4 var spaceY;
5 var spaceX;
6 var moves = 0;
7 var timer = 0;
8 var start;
9
10
11
12 // When the page is loaded, set up the puzzle pieces and shuffle button
13 window.onload = function () {
14     var modal = document.getElementById("myModal");
15     var span = document.getElementsByClassName("close")[0];
16     span.onclick = function () {
17         modal.style.display = "none";
18     }
19     window.onclick = function (event) {
20         if (event.target == modal) {
21             modal.style.display = "none";
22         }
23     }
24
25     var puzzle = document.getElementById("puzzle");
26     piece = puzzle.getElementsByTagName("div");
27     document.getElementById("moves").innerHTML = moves;
28     document.getElementById("time").innerHTML = "0s";
29     for (var i = 0; i < piece.length; i++)
30     {
31         piece[i].className = "piece";
32         piece[i].style.left = (i % 4 * 100) + "px";
33         piece[i].style.top = (parseInt(i / 4) * 100) + "px";
34         piece[i].style.backgroundColor = "- " + piece[i].style.left + " " + "-" + piece[i].style.top;
35         piece[i].style.backgroundImage = "url('Rock.jpg')";
36     }
37
38     // When a piece is hovered, add styling if it is a movable piece and remove that styling when no longer hovering
39     piece[i].onmouseover = function ()
40     {
41         if (isMovable(parseInt(this.innerHTML)))
42         {
43             this.style.border = "3px solid red";
44             this.style.color = "#000000";
45             this.style.textDecoration = "underline";
46             this.style.cursor = "pointer";
47         }
48     };
49     piece[i].onmouseout = function ()
50     {
51 
```

```
52         this.style.border = "2px solid black";
53         this.style.color = "#000000";
54         this.style.textDecoration = "none";
55         this.style.cursor = "default";
56     };
57
58     // When a piece is clicked, check if that piece is movable. If it is, swap it and check if the game is won.
59     piece[i].onclick = function ()
60     {
61         if (isMovable(parseInt(this.innerHTML)))
62         {
63             swap(this.innerHTML - 1);
64             moves++;
65             document.getElementById("moves").innerHTML = moves;
66             if (finish())
67             {
68                 timer = clearInterval(timer);
69                 moves = 0;
70                 var duration = 1 * 1000;
71                 var end = Date.now() + duration;
72                 (function frame() {
73                     // Launch a few confetti from the left edge
74                     confetti({
75                         particleCount: 7,
76                         angle: 60,
77                         spread: 55,
78                         origin: { x: Math.random(), y: Math.random() }
79                     });
80
81                     // Keep going until we are out of time
82                     if (Date.now() < end) {
83                         requestAnimationFrame(frame);
84                     }
85                 })();
86                 modal.style.display = "block";
87                 document.getElementById("winTime").innerHTML = "Time: " + document.getElementById("time").innerHTML;
88                 document.getElementById("winMoves").innerHTML = "Moves: " + document.getElementById("moves").innerHTML;
89                 document.getElementById("moves").innerHTML = moves;
90                 document.getElementById("time").innerHTML = 0 + "s";
91             }
92         }
93     };
94
95     var shuffle = document.getElementById("shufflebutton");
96     spaceX = "300px";
97     spaceY = "300px";

```

# The Code: puzzle.js (cont.)

```
97 // When the shuffle button is clicked, simulate 300 random moves of the blank space
98 shuffle.onclick = function ()
99 {
100     // Start the timer
101     start = new Date().getTime();
102     timer = setInterval(function () {
103         var now = new Date().getTime();
104         var distance = now - start;
105         var seconds = Math.floor((distance / 1000));
106         document.getElementById("time").innerHTML = seconds + "s";
107     }, 1000);
108     moves = 0;
109     document.getElementById("moves").innerHTML=moves;
110     for (var i = 0; i < 300; i++) {
111         var rand = parseInt(Math.random() * 100) % 4;
112         if (rand == 0) {
113             var temp = up(spaceX, spaceY);
114             if (temp != -1) {
115                 swap(temp);
116             }
117         }
118         if (rand == 1) {
119             var temp = down(spaceX, spaceY);
120             if (temp != -1) {
121                 swap(temp);
122             }
123         }
124         if (rand == 2) {
125             var temp = left(spaceX, spaceY);
126             if (temp != -1) {
127                 swap(temp);
128             }
129         }
130         if (rand == 3) {
131             var temp = right(spaceX, spaceY);
132             if (temp != -1) {
133                 swap(temp);
134             }
135         }
136     }
137 }
138 };
```

```
141 function isMovable(position) // returns true whenever a piece can be moved into an empty space
142 {
143     if (left(spaceX, spaceY) == (position - 1)) {
144         return true;
145     }
146     if (down(spaceX, spaceY) == (position - 1)) {
147         return true;
148     }
149     if (up(spaceX, spaceY) == (position - 1)) {
150         return true;
151     }
152     if (right(spaceX, spaceY) == (position - 1)) {
153         return true;
154     }
155 }
156
157 function left(x, y) //calculates how far to the left a puzzlepiece should position
158 {
159     var cordX = parseInt(x);
160     var cordY = parseInt(y);
161
162     if (cordX > 0) {
163         for (var i = 0; i < piece.length; i++) {
164             if (parseInt(piece[i].style.left) + 100 == cordX && parseInt(piece[i].style.top) == cordY) {
165                 return i;
166             }
167         }
168     } else {
169         return -1;
170     }
171 }
172
173 function right(x, y) //calculates how far to the right a puzzlepiece should position
174 {
175     var cordX = parseInt(x);
176     var cordY = parseInt(y);
177     if (cordX < 300) {
178         for (var i = 0; i < piece.length; i++) {
179             if (parseInt(piece[i].style.left) - 100 == cordX && parseInt(piece[i].style.top) == cordY) {
180                 return i;
181             }
182         }
183     } else {
184         return -1;
185     }
186 }
187
188 }
```

# The Code: puzzle.js (cont.2)

```
191 function up(x, y) //calculates how far up a puzzlepiece should position
192 {
193     var cordX = parseInt(x);
194     var cordY = parseInt(y);
195     if (cordY > 0) {
196         for (var i = 0; i < piece.length; i++) {
197             if (parseInt(piece[i].style.top) + 100 == cordY && parseInt(piece[i].style.left) == cordX) {
198                 return i;
199             }
200         }
201     } else {
202         return -1;
203     }
204 }
205
206 function down(x, y) //calculates how far down a puzzlepiece should position
207 {
208     var cordX = parseInt(x);
209     var cordY = parseInt(y);
210     if (cordY < 300) {
211         for (var i = 0; i < piece.length; i++) {
212             if (parseInt(piece[i].style.top) - 100 == cordY && parseInt(piece[i].style.left) == cordX) {
213                 return i;
214             }
215         }
216     } else {
217         return -1;
218     }
219 }
220
221 // Swaps the piece into the empty space
222 function swap(position)
223 {
224     var temp = piece[position].style.top;
225     piece[position].style.top = spaceY;
226     spaceY = temp;
227     temp = piece[position].style.left;
228     piece[position].style.left = spaceX;
229     spaceX = temp;
230 }
231
232 }
```

```
235 // Function to check if the game is finished
236 function finish()
237 {
238     var flag = true;
239     // for every piece
240     for (var i = 0; i < piece.length; i++)
241     {
242         var top = parseInt(piece[i].style.top);
243         var left = parseInt(piece[i].style.left);
244         // Check if the left and top positions are correct
245         if (left != (i % 4 * 100) || top != parseInt(i / 4) * 100)
246         {
247             flag = false;
248             break;
249         }
250     }
251     return flag;
252 }
253
254 function pad(val) {
255     var valString = val + "";
256     if (valString.length < 2) {
257         return "0" + valString;
258     } else {
259         return valString;
260     }
261 }
262
263 // Function to change the background image
264 function changeBackground(num)
265 {
266     var images = ["/backgrounds/fine.jpeg", "/backgrounds/heman.jpeg", "/backgrounds/pepe_sad.jpeg", "/backgrounds/swamp.jpeg"];
267     document.body.style.backgroundColor = "#f3f3f3";
268     document.body.style.backgroundImage = "url(" + images[num] + ")";
269 }
270
271 }
```

The End, Thank you!

