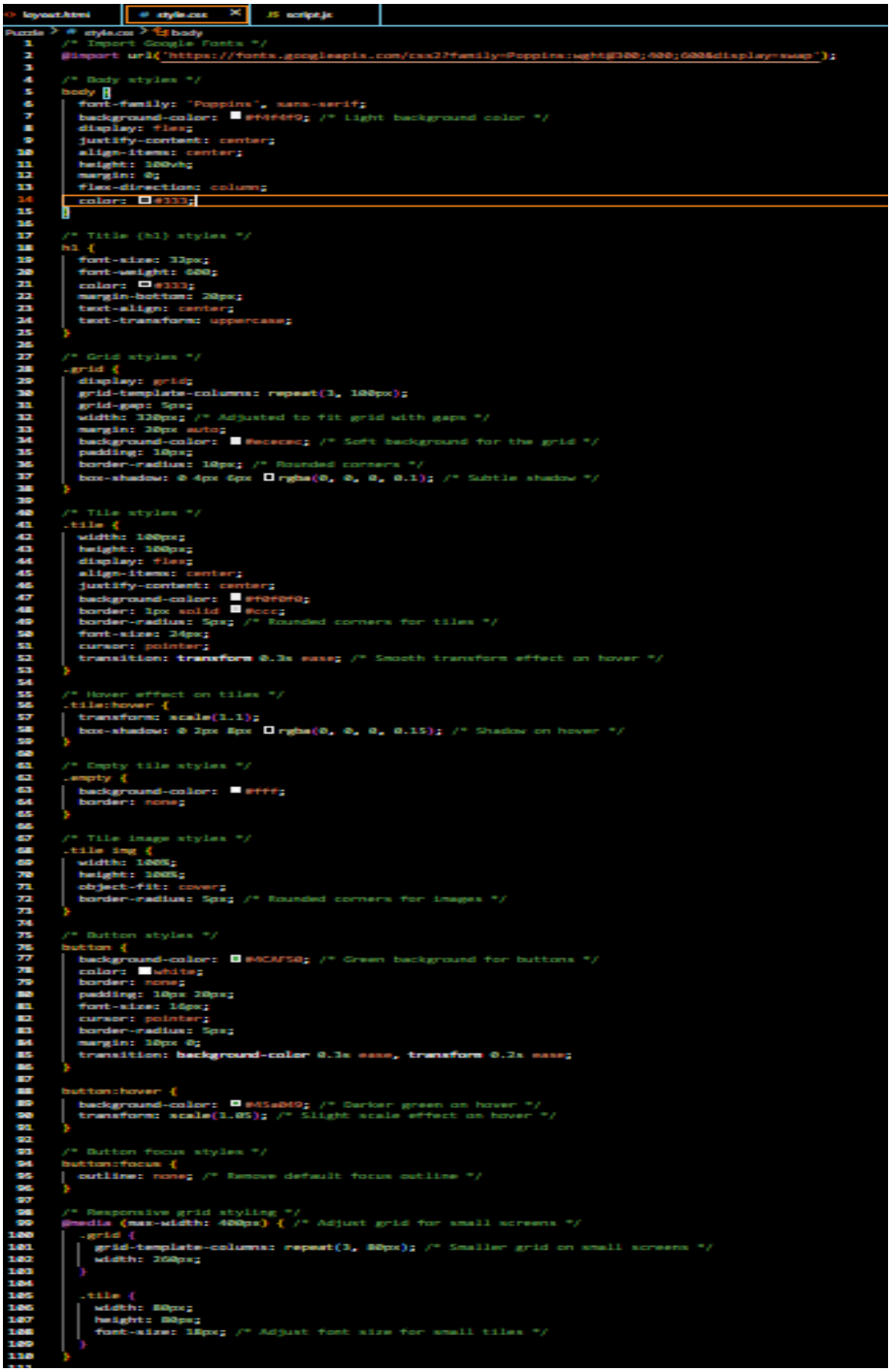
Puzzle_Game

By Anass AJJA


JAVASCRIPT & JQUERY
Exercices
Puzzle
puzzle_image
favicon.ico
layout.html
script.js
style.css
screens
~$2_JS_AnassAJJA.docx
~$3_JS_AnassAJJA.docx
cours1.pdf
cours2.pdf
JavaScript Serie 1.pdf
JavaScript Serie 2 .pdf
JavaScript Serie 3 .pdf
TP1_JS_AnassAJJA.docx
TP1_JS_AnassAJJA.pdf
TP2_JS_AnassAJJA.docx
TP2_JS_AnassAJJA.pdf
TP3_JS_AnassAJJA.docx
Puzzle > layout.html > html > body > button#random
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Image Puzzle Game</title>
7 <link rel="stylesheet" href="style.css">
8 <link rel="icon" href="favicon.ico" type="image/x-icon">
9 </head>
10 <body>
11 <!-- Title -->
12 <h1>Image Puzzle Game</h1>
13
14 <!-- Puzzle Grid -->
15
16
17 <!-- Buttons -->
18 <button id="random">Random</button>
19 <button id="resolve">Resolve</button>
20
21 <script src="script.js"></script>
22 </body>
23 </html>


layout.html
style.css
script.js
Puzzle > style.css > body
/* Import Google Fonts */
@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap');
/* Body styles */
body {
font-family: 'Poppins', sans-serif;
background-color: #f4f4f9; /* Light background color */
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
margin: 0;
flex-direction: column;
color: #333;
}
/* Title (h1) styles */
h1 {
font-size: 32px;
font-weight: 600;
color: #333;
margin-bottom: 20px;
text-align: center;
text-transform: uppercase;
}
/* Grid styles */
.grid {
display: grid;
grid-template-columns: repeat(3, 100px);
grid-gap: 5px;
width: 320px; /* Adjusted to fit grid with gaps */
margin: 20px auto;
background-color: #ececec; /* Soft background for the grid */
padding: 10px;
border-radius: 10px; /* Rounded corners */
box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1); /* Subtle shadow */
}
/* Tile styles */
.tile {
width: 100px;
height: 100px;
display: flex;
align-items: center;
justify-content: center;
background-color: #f0f0f0;
border: 1px solid #ccc;
border-radius: 5px; /* Rounded corners for tiles */
font-size: 24px;
cursor: pointer;
transition: transform 0.3s ease; /* Smooth transform effect on hover */
}
/* Hover effect on tiles */
.tile:hover {
transform: scale(1.1);
box-shadow: 0 2px 8px rgba(0, 0, 0, 0.15); /* Shadow on hover */
}
/* Empty tile styles */
.empty {
background-color: #fff;
border: none;
}
/* Tile image styles */
.tile img {
width: 100%;
height: 100%;
object-fit: cover;
border-radius: 5px; /* Rounded corners for images */
}
/* Button styles */
button {
background-color: #4CAF50; /* Green background for buttons */
color: white;
border: none;
padding: 10px 20px;
font-size: 16px;
cursor: pointer;
border-radius: 5px;
margin: 10px 0;
transition: background-color 0.3s ease, transform 0.2s ease;
}
button:hover {
background-color: #45a049; /* Darker green on hover */
transform: scale(1.05); /* Slight scale effect on hover */
}
/* Button focus styles */
button:focus {
outline: none; /* Remove default focus outline */
}
/* Responsive grid styling */
@media (max-width: 400px) { /* Adjust grid for small screens */
.grid {
grid-template-columns: repeat(3, 80px); /* Smaller grid on small screens */
width: 260px;
}
.tile {
width: 80px;
height: 80px;
font-size: 18px; /* Adjust font size for small tiles */
}
}

layout.html | style.css | script.js

Puzzle > script.js > addEventListener('click') callback

```
const grid = document.getElementById('grid'); // Get the grid element

// Array of tile images, with one slot being null for the empty space
let tiles = [
  'puzzle_image/img1.jpg',
  'puzzle_image/img2.jpg',
  'puzzle_image/img3.jpg',
  'puzzle_image/img4.jpg',
  'puzzle_image/img5.jpg',
  'puzzle_image/img6.jpg',
  'puzzle_image/img7.jpg',
  'puzzle_image/img8.jpg',
  null // Empty space
];

// Function to render the grid
function renderGrid() {
  grid.innerHTML = ''; // Clear existing grid
  tiles.forEach((tile, index) => { // Loop through each tile
    const tileDiv = document.createElement('div'); // Create a div element for each tile
    tileDiv.className = tile === null ? 'tile empty' : 'tile'; // Set the class for empty space

    if (tile) { // If the tile is not empty, create an image element
      const img = document.createElement('img'); // If the tile is not empty, create an image element
      img.src = tile; // Set the image source
      tileDiv.appendChild(img); // Append the image to the tile div
    }

    tileDiv.addEventListener('click', () => moveTile(index)); // Add click event to move the tile
    grid.appendChild(tileDiv); // Append the tile div to the grid
  });
}

// Initial render
renderGrid();
```

layout.html | style.css | script.js

Puzzle > JS script.js > ...

```
36
37  // Function to move the tile into the empty space
38  function moveTile(index) { // Function to move the tile into the empty space
39    const emptyIndex = tiles.indexOf(null); // Get the index of the empty space
40    const validMoves = [index - 1, index + 1, index - 3, index + 3]; // Adjacent indices: left
41
42    // Check if the move is valid and the empty space is adjacent
43    if (validMoves.includes(emptyIndex) && isValidPosition(index, emptyIndex)) { // Check if t
44      [tiles[index], tiles[emptyIndex]] = [tiles[emptyIndex], tiles[index]]; // Swap the tiles
45      renderGrid(); // Re-render the grid with the updated positions
46    }
47  }
48
49  // Check if the move is within bounds (no wrapping around rows)
50  function isValidPosition(index, emptyIndex) { // Check if the move is within bounds (no wrap
51    if (index % 3 === 0 && emptyIndex === index - 1) return false; // Left edge
52    if (index % 3 === 2 && emptyIndex === index + 1) return false; // Right edge
53    return true; // Valid move
54  }
55
56  // Shuffle the tiles
57  document.getElementById('random').addEventListener('click', () => { // Add click event to sh
58    // Shuffle tiles randomly while keeping the empty space intact
59    do { // Keep shuffling until the puzzle is solvable
60      tiles = tiles // Shuffle tiles randomly while keeping the empty space intact
61        .map(value => ({ value, sort: Math.random() })) // Randomize tiles
62        .sort((a, b) => a.sort - b.sort) // Sort the tiles
63        .map(({ value }) => value); // Get the sorted tiles
64    } while (isPuzzleSolvable() === false); // Ensure the puzzle is solvable
65    renderGrid(); // Re-render the grid with shuffled tiles
66  });
67
68  // Check if the puzzle is solvable (optional)
69  function isPuzzleSolvable() { // Count the number of inversions
70    let inversions = 0;   // Count the number of inversions
71    const flatTiles = tiles.filter(t => t !== null); // Flatten the tiles and remove the empty
72    for (let i = 0; i < flatTiles.length; i++) { // Count inversions
73      for (let j = i + 1; j < flatTiles.length; j++) { // Count inversions
74        if (flatTiles[i] > flatTiles[j]) inversions++; // Count inversions
75      }
76    }
77    return inversions % 2 === 0; // Puzzle is solvable if inversions are even
78  }
79
```

```
// Check if the puzzle is solvable (optional)
function isPuzzleSolvable() { // Count the number of inversions
  let inversions = 0;   // Count the number of inversions
  const flatTiles = tiles.filter(t => t !== null); // Flatten the tiles a
  for (let i = 0; i < flatTiles.length; i++) { // Count inversions
    for (let j = i + 1; j < flatTiles.length; j++) { // Count inversions
      if (flatTiles[i] > flatTiles[j]) inversions++; // Count inversions
    }
  }
  return inversions % 2 === 0; // Puzzle is solvable if inversions are ev
}

// Resolve the puzzle (reset to the solved state)
document.getElementById('resolve').addEventListener('click', () => { // A
  tiles = [
    null, // Empty space
    'puzzle_image/img8.jpg',
    'puzzle_image/img7.jpg',
    'puzzle_image/img6.jpg',
    'puzzle_image/img5.jpg',
    'puzzle_image/img4.jpg',
    'puzzle_image/img3.jpg',
    'puzzle_image/img2.jpg',
    'puzzle_image/img1.jpg'
  ];
  renderGrid(); // Re-render the grid to its solved state
});
```

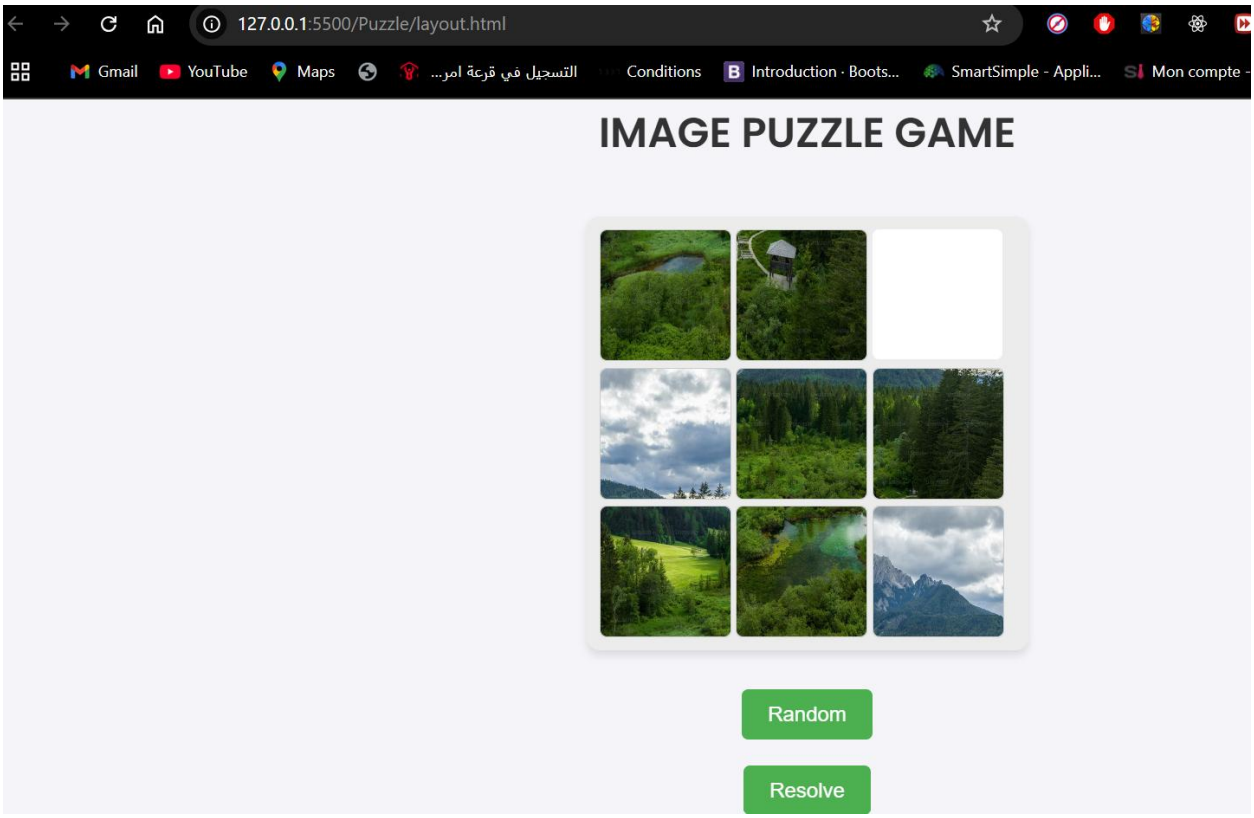
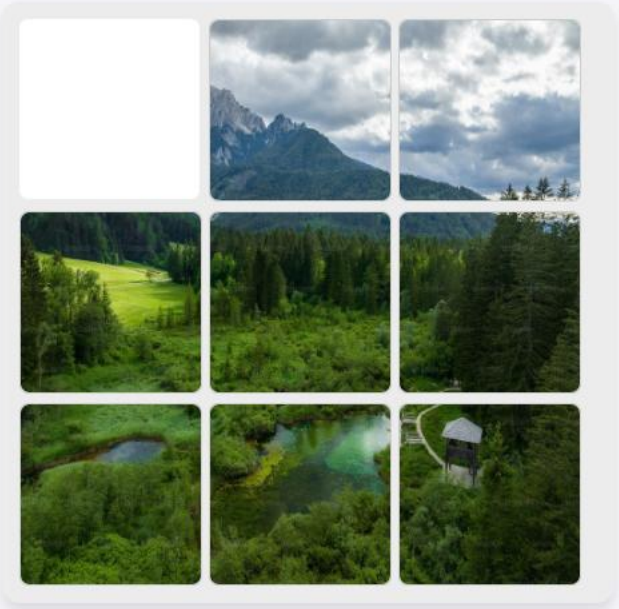

127.0.0.1:5500/Puzzle/layout.html
Gmail
YouTube
Maps
التسجيل في قرعة امر...
Conditions
Introduction · Boots...
SmartSimple - Appli...
Mon compte
IMAGE PUZZLE GAME
Random
Resolve

IMAGE PUZZLE GAME



Random

Resolve