



# JavaScript Slot Machine

*Dynamic Interactive JavaScript DOM project*

## JavaScript Slot Machine coding project Dynamic Interactive JavaScript DOM project

Explore how you can create elements have user interactions and trigger visual events making your web pages come to life.

### 1 Project Setup create HTML and JavaScript files

Setup html file prep to add JavaScript coding. Create HTML game container element, link to JavaScript source files. Select main output element using JavaScript.

## 2 JavaScript SlotMachine create interactive Button. Setup of HTML Web Page and JavaScript Code

Select main container element, add button for interaction. Allow user to toggle button content and select and invoke function on button click action. Setup of core Global Game properties to make the application dynamically adjust with new game object values. Append elements to the page with JavaScript. Create elements with JavaScript. Add event listeners to elements with JavaScript.

### Lesson Challenge

1. Create elements to contain game content, information element, button for interaction and main content element
2. Add event listener to the button for the user to press
3. Toggle button content invoke function on button click

```
<!DOCTYPE html>
<html>
  <head><title>JavaScript DOM Project</title>
  <style>

  </style>
</head>
<body>
  <div class="myDiv">Hello World</div>
  <script src="apps3.js"></script>
</body>
</html>
```

```
const output = document.querySelector('.myDiv');
const message = document.createElement('div');
const gameArea = document.createElement('div');
const btn = document.createElement('button');
message.innerHTML = output.textContent;
output.innerHTML = '';
output.append(message);
output.append(btn);
output.append(gameArea);

const game = {total:4,inPlay:false,coins:100,speed:5,totItems:4,main:[]};
btn.textContent = 'SPIN';
```

```

btn.addEventListener('click', (e)=>{
  if(btn.textContent == 'SPIN' && !game.inPlay){
    btn.textContent = 'STOP';
    startSpin();
  }else{
    game.inPlay = false;
    btn.textContent = 'SPIN';
  }
})

function startSpin(){
  console.log('Spinning ' + game.inPlay);
}

```

### 3 Create Game Elements JavaScript Apply CSS

Add and update the toggle of the clickable button element. Invoke a JavaScript function on click. Use of DOMContentLoaded event to build game board once the DOM is loaded and ready to use. Get document body properties to use values within the JavaScript code document.body.clientWidth. Create element maker function to generate elements within the JavaScript code, add and append new element to the parent, add a class, element tag, and html content within the element.

#### Lesson Challenge

1. Create and launch game board makers when DOM is loaded
2. Create function to handle making elements for the page
3. Apply styling to elements to center and position on the page

```

const output = document.querySelector('.myDiv');
output.innerHTML = document.body.clientWidth;
const message = makerElement(output, 'div', 'message', 'message');

```

```

const gameArea = makerElement(output, 'div', '', 'gameArea');
const btn = makerElement(output, 'button', 'SPIN', 'btn');
const game = {total:3, inPlay:false, coins:100, speed:5, totItems:5, main:[]};

window.addEventListener('DOMContentLoaded', init);
btn.addEventListener('click', (e)=>{
    if(btn.textContent == 'SPIN' && !game.inPlay){
        btn.textContent = 'STOP';
        startSpin();
    }else{
        game.inPlay = false;
        btn.textContent = 'SPIN';
    }
})

function init(){
    console.log('ready');
    gameArea.style.width = game.total * 100 + 'px';
    let leftPos = (document.body.clientWidth - (game.total * 100)) / 2;
    console.log(leftPos);
    gameArea.style.left = leftPos + 'px';

    for(let i=0; i<game.total; i++){
        game.main[i] = makerElement(gameArea, 'div', '', 'wheel')
        for(let x=0; x<game.totItems; x++){
            const el = makerElement(game.main[i], 'div', x+1, 'box');
        }
        game.main[i].style.left = i * 100 + 'px';
    }
}

function makerElement(parent, ele, html, myClass){
    const el = document.createElement(ele);
    el.classList.add(myClass);
    el.innerHTML = html;
    parent.append(el);
    return el;
}

function startSpin(){
    console.log('Spinning ' + game.inPlay);
}

```

## 4 Update CSS styling to set dynamically created elements on page

Adding CSS to position elements, set widths and heights to set the content on the page by applying classes with JavaScript to the newly created page elements.

### Lesson Challenge

1. Update CSS to apply positioning of page elements
2. Update game object values to ensure they work with updated and applied styling, adjust JavaScript if needed.
3. Apply styling and test the code.

```
<!DOCTYPE html>
<html>
  <head><title>JavaScript DOM Project</title>
  <style>
    * {
      box-sizing: border-box;
    }
    .box{
      position: relative;
      width:100px;
      height:100px;
      border: 1px solid black;
      top:0;
      left:0;
      font-size: 3em;
      line-height: 100px;
      text-align: center;
    }
    .wheel{
      position: absolute;
      left:0px;
      top:0px;
    }

    .gameArea {
      position: absolute;
```

```

    left:0px;
    top:100px;
    width:300px;
    height: 103px;
    border: 1px solid red;
  }
</style>
</head>
<body>
  <div class="myDiv">Hello World</div>
  <script src="apps3.js"></script>
</body>
</html>

const output = document.querySelector('.myDiv');
output.innerHTML = document.body.clientWidth;
const message = makerElement(output, 'div', 'message', 'message');
const gameArea = makerElement(output, 'div', '', 'gameArea');
const btn = makerElement(output, 'button', 'SPIN', 'btn');
const game = {total:3, inPlay:false, coins:100, speed:5, totItems:8, main:[]};

window.addEventListener('DOMContentLoaded', init);
btn.addEventListener('click', (e) => {
  if (btn.textContent == 'SPIN' && !game.inPlay) {
    btn.textContent = 'STOP';
    startSpin();
  } else {
    game.inPlay = false;
    btn.textContent = 'SPIN';
  }
})

function init() {
  console.log('ready');
  gameArea.style.width = game.total * 100 + 'px';
  let leftPos = (document.body.clientWidth - (game.total * 100)) / 2;
  console.log(leftPos);
  gameArea.style.left = leftPos + 'px';

  for (let i=0; i<game.total; i++) {
    game.main[i] = makerElement(gameArea, 'div', '', 'wheel')
    for (let x=0; x<game.totItems; x++) {
      const el = makerElement(game.main[i], 'div', x+1, 'box');

```

```

    }
    game.main[i].style.left = i * 100 + 'px';
  }
}

function makerElement(parent,ele,html,myClass){
  const el = document.createElement(ele);
  el.classList.add(myClass);
  el.innerHTML = html;
  parent.append(el);
  return el;
}

function startSpin(){
  console.log('Spinning ' + game.inPlay);
}

```

## 5 JavaScript adding animation frame to create smooth movement of elements.

The `window.requestAnimationFrame()` method tells the browser that you wish to perform an animation and requests that the browser calls a specified function to update an animation before the next repaint. The method takes a callback as an argument to be invoked before the repaint. Add animation frames that can be added and removed with a global object.

### Lesson Challenge

1. Add animation frame
2. Interaction with button to stop and start animation frames with JavaScript

```

const output = document.querySelector('.myDiv');
output.innerHTML = document.body.clientWidth;

```

```

const message = makerElement(output, 'div', 'message', 'message');
const gameArea = makerElement(output, 'div', '', 'gameArea');
const btn = makerElement(output, 'button', 'SPIN', 'btn');
const game = {total:3, inPlay:false, coins:100, speed:5, totItems:8, main:[]};

window.addEventListener('DOMContentLoaded', init);
btn.addEventListener('click', (e) => {
  if (btn.textContent === 'SPIN' && !game.inPlay) {
    btn.textContent = 'STOP';
    startSpin();
  } else {
    game.inPlay = false;
    cancelAnimationFrame(game.ani);
    btn.textContent = 'SPIN';
  }
})

function init() {
  console.log('ready');
  gameArea.style.width = game.total * 100 + 'px';
  let leftPos = (document.body.clientWidth - (game.total * 100)) / 2;
  console.log(leftPos);
  gameArea.style.left = leftPos + 'px';

  for (let i=0; i<game.total; i++) {
    game.main[i] = makerElement(gameArea, 'div', '', 'wheel')
    for (let x=0; x<game.totItems; x++) {
      const el = makerElement(game.main[i], 'div', x+1, 'box');
    }
    game.main[i].style.left = i * 100 + 'px';
  }
}

function makerElement(parent, ele, html, myClass) {
  const el = document.createElement(ele);
  el.classList.add(myClass);
  el.innerHTML = html;
  parent.append(el);
  return el;
}

function startSpin() {

```

```

    game.inPlay = true;
    console.log('Spinning ' + game.inPlay);
    game.ani = requestAnimationFrame(spin);
}

function spin(){
    console.log('running');
    game.ani = requestAnimationFrame(spin);
}

```

## 6 Movement of Slot Wheels with JavaScript Page element style updates.

Update the position of the element on the page, move the elements restack the order of elements within a parent element. Getting element property values to use within the code to update position. offsetTop with JavaScript. Conditions and calculations to manipulate element style positions top and left to create animation of elements with JavaScript code.

### Lesson Challenge

1. movement of page elements
2. adding values to element objects
3. updating element values updating style position values
4. Setting and Getting position property values

```

const output = document.querySelector('.myDiv');
output.innerHTML = document.body.clientWidth;
const message = makerElement(output, 'div', 'message', 'message');
const gameArea = makerElement(output, 'div', '', 'gameArea');
const btn = makerElement(output, 'button', 'SPIN', 'btn');
const game = {total:3, inPlay:false, coins:100, speed:5, totItems:8, main:[]};
let spinner = 500;

```

```

window.addEventListener('DOMContentLoaded',init);
btn.addEventListener('click', (e)=>{
  if(btn.textContent == 'SPIN' && !game.inPlay){
    btn.textContent = 'STOP';
    spinner=500;
    startSpin();
  }else{
    game.inPlay = false;
    cancelAnimationFrame(game.ani);
    btn.textContent = 'SPIN';
  }
})

function init(){
  console.log('ready');
  gameArea.style.width = game.total * 100 + 'px';
  let leftPos = (document.body.clientWidth - (game.total *100)) /2;
  console.log(leftPos);
  gameArea.style.left = leftPos + 'px';

  for(let i=0;i<game.total;i++){
    game.main[i] = makerElement(gameArea,'div','', 'wheel')
    for(let x=0;x<game.totItems;x++){
      const el = makerElement(game.main[i], 'div', x+1, 'box');
    }
    game.main[i].style.left = i * 100 + 'px';
  }
}

function makerElement(parent,ele,html,myClass){
  const el = document.createElement(ele);
  el.classList.add(myClass);
  el.innerHTML = html;
  parent.append(el);
  return el;
}

function startSpin(){
  game.inPlay = true;
  spinner=500;
  console.log('Spinning ' + game.inPlay);
}

```

```

    for(let i =0;i<game.total;i++){
        game.main[i].mover = Math.floor(Math.random()*50)+10;
    }
    game.ani = requestAnimationFrame(spin);
}

function spin(){
    spinner--;
    if(spinner<=0){
        game.inPlay = false;
        cancelAnimationFrame(game.ani);
        btn.textContent = 'SPIN';
    }
    if(game.inPlay){
        console.log('running');
        let holder = [];
        for(let i=0;i<game.total;i++){
            let el = game.main[i];
            let elY = el.offsetTop;
            console.log(el.offsetTop);
            if(el.mover > 0){
                el.mover--;
                console.log(el.mover);
                elY += game.speed;
                if(elY > -150){
                    elY -= 100;
                    const last = el.lastElementChild;
                    el.prepend(last);
                }
                el.style.top = elY + 'px';
            }
        }
        game.ani = requestAnimationFrame(spin);
    }
}

```

## 7 JavaScript Game Movement debugging and Fixes

JavaScript Game movement and debugging. How to troubleshoot your JavaScript game application and how to create and optimize gameplay. Update the game area styling with CSS. Adding and removing classes from JavaScript objects. Game playthrough and updates in element positions, style properties done with JavaScript.

### Lesson Challenge

1. Add user message information
2. test movement update to have a better final position.
- 3 Add styling as needed.

```
const output = document.querySelector('.myDiv');
output.innerHTML = '';
const message = makerElement(output, 'div', 'message', 'message');
const gameArea = makerElement(output, 'div', '', 'gameArea');
const btn = makerElement(output, 'button', 'SPIN', 'btn');
const game = {total:4, inPlay:false, coins:100, speed:5, totItems:8, main:[]};
let spinner = 500;

window.addEventListener('DOMContentLoaded', init);
btn.addEventListener('click', (e) => {
  if (btn.textContent == 'SPIN' && !game.inPlay) {
    btn.textContent = 'STOP';
    btn.style.backgroundColor = 'red';
    spinner = 500;
    startSpin();
  } else {
    game.inPlay = false;
    cancelAnimationFrame(game.ani);
    btn.style.backgroundColor = 'green';
    btn.textContent = 'SPIN';
  }
})

function init() {
  //console.log('ready');
  btn.style.backgroundColor = 'green';
}
```

```

gameArea.style.width = game.total * 100 + 'px';
let leftPos = (document.body.clientWidth - (game.total * 100)) / 2;
//console.log(leftPos);
gameArea.style.left = leftPos + 'px';

for(let i=0;i<game.total;i++){
    game.main[i] = makerElement(gameArea,'div','', 'wheel')
    for(let x=0;x<game.totItems;x++){
        const el = makerElement(game.main[i], 'div', x+1, 'box');
    }
    game.main[i].style.left = i * 100 + 'px';
}
}

function makerElement(parent,ele,html,myClass){
    const el = document.createElement(ele);
    el.classList.add(myClass);
    el.innerHTML = html;
    parent.append(el);
    return el;
}

function updateMessage(html){
    message.innerHTML = html;
}

function startSpin(){
    game.coins --;
    updateMessage(`You have ${game.coins} left.`);
    game.inPlay = true;
    spinner=500;
    //console.log('Spinning ' + game.inPlay);
    for(let i =0;i<game.total;i++){
        game.main[i].mover = Math.floor(Math.random()*50)+10;
    }
    game.ani = requestAnimationFrame(spin);
}

function spin(){
    spinner--;
    if(spinner<=0){

```

```

    game.inPlay = false;
    cancelAnimationFrame(game.ani);
    btn.textContent = 'SPIN';
}

//console.log('running');
let holder = [];
for(let i=0;i<game.total;i++){
    let el = game.main[i];
    let elY = el.offsetTop;
    //console.log(el.offsetTop);
    if(el.mover > 0){
        el.mover--;
        //console.log(el.mover);
        elY += game.speed;
        if(elY > -150){
            elY -= 100;
            const last = el.lastElementChild;
            el.prepend(last);
        }
        if(el.mover == 0 && elY % 50 != 0)
            el.mover++;
    }
    el.style.top = elY + 'px';
}
if(game.inPlay){
    game.ani = requestAnimationFrame(spin);
}
}

```

## 8 Movement and Game Results Setting Conditions for win JavaScript

Final spin results and retrieving the values of the output results. Comparison functions and setting up win conditions for JavaScript Game Object. JavaScript

Game movement and debugging. How to troubleshoot your JavaScript game application and how to create and optimize gameplay.

## Lesson Challenge

1. Collect the output values and position of the final spin results into an array
2. update and compare final spin result to prepare for comparisons on results.

```
const output = document.querySelector('.myDiv');
output.innerHTML = '';
const message = makerElement(output, 'div', 'message', 'message');
const gameArea = makerElement(output, 'div', '', 'gameArea');
const btn = makerElement(output, 'button', 'SPIN', 'btn');
const game = {total:4, inPlay:false, coins:100, speed:5, totItems:8, main:[]};
let spinner = 500;

window.addEventListener('DOMContentLoaded', init);
btn.addEventListener('click', (e) => {
  if (btn.textContent == 'SPIN' && !game.inPlay) {
    btn.textContent = 'STOP';
    btn.style.backgroundColor = 'red';
    spinner = 500;
    startSpin();
  } else {
    stopGamePlay();
  }
})

function init() {
  //console.log('ready');
  btn.style.backgroundColor = 'green';
  gameArea.style.width = game.total * 100 + 'px';
  let leftPos = (document.body.clientWidth - (game.total * 100)) / 2;
  //console.log(leftPos);
  gameArea.style.left = leftPos + 'px';

  for (let i = 0; i < game.total; i++) {
```

```

        game.main[i] = makerElement(gameArea, 'div', '', 'wheel')
        for(let x=0;x<game.totItems;x++){
            const el = makerElement(game.main[i], 'div', x+1, 'box');
            el.faceValue = x+1;
        }
        game.main[i].style.left = i * 100 + 'px';
    }
}

function makerElement(parent, ele, html, myClass){
    const el = document.createElement(ele);
    el.classList.add(myClass);
    el.innerHTML = html;
    parent.append(el);
    return el;
}

function updateMessage(html){
    message.innerHTML = html;
}

function startSpin(){
    game.coins --;
    updateMessage(`You have ${game.coins} left.`);
    game.inPlay = true;
    spinner=500;
    ///console.log('Spinning ' + game.inPlay);
    for(let i =0;i<game.total;i++){
        game.main[i].mover = Math.floor(Math.random()*50)+10;
    }
    game.ani = requestAnimationFrame(spin);
}

function spin(){
    spinner--;
    if(spinner<=0){
        stopGamePlay();
    }

    let holder = [];
    for(let i=0;i<game.total;i++){

```

```

    let el = game.main[i];
    let elY = el.offsetTop;
    if(el.mover > 0){
        el.mover--;
        elY += game.speed;
        if(elY > -150){
            elY -= 100;
            const last = el.lastElementChild;
            el.prepend(last);
        }
        if(el.mover == 0 && elY % 50 != 0){
            el.mover++;
        }

        el.style.top = elY + 'px';
    }else{
        let viewEl = el.children[2];
        let outputVal = elY == -200 ? viewEl.faceValue : '-';
        let tempObj = {
            'txt' : viewEl.faceValue,
            'elY' : elY,
            'outputV' : outputVal,
            'output' : viewEl.textContent
        }
        holder.push(tempObj);
    }
}

if(holder.length >= game.total){
    stopGamePlay();
    holder.sort();
    console.log(holder);
}

if(game.inPlay){
    game.ani = requestAnimationFrame(spin);
}
}

function stopGamePlay(){
    game.inPlay = false;
    cancelAnimationFrame(game.ani);
    btn.textContent = 'SPIN';
}

```

```
btn.style.backgroundColor = 'green';
}
```

## 9 JavaScript Slot Machine Win Conditions and payout for matches

How to create a final tally object that can be used to calculate the final results for the player. Track matches and number of occurrences to be able to apply calculations on win.

### Lesson Challenge

1. Calculate the payout - using the number of matches and the value of the match
2. Create a function that can track the values and has conditions to adjust the payout value accordingly

```
const output = document.querySelector('.myDiv');
output.innerHTML = '';
const message = makerElement(output, 'div', 'message', 'message');
const gameArea = makerElement(output, 'div', '', 'gameArea');
const btn = makerElement(output, 'button', 'SPIN', 'btn');
const game = {total:5, inPlay:false, coins:100, speed:5, totItems:4, main:[]};
let spinner = 500;

window.addEventListener('DOMContentLoaded', init);
btn.addEventListener('click', (e) => {
  if (btn.textContent === 'SPIN' && !game.inPlay) {
    btn.textContent = 'STOP';
    btn.style.backgroundColor = 'red';
    spinner = 500;
    startSpin();
  } else {
    stopGamePlay();
  }
});
```

```

    }
  })

function init(){
  ///console.log('ready');
  btn.style.backgroundColor = 'green';
  gameArea.style.width = game.total * 100 + 'px';
  let leftPos = (document.body.clientWidth - (game.total *100)) /2;
  ///console.log(leftPos);
  gameArea.style.left = leftPos + 'px';

  for(let i=0;i<game.total;i++){
    game.main[i] = makerElement(gameArea,'div','', 'wheel')
    for(let x=0;x<game.totItems;x++){
      const el = makerElement(game.main[i], 'div', x+1, 'box');
      el.faceValue = x+1;
    }
    game.main[i].style.left = i * 100 + 'px';
  }
}

function makerElement(parent,ele,html,myClass){
  const el = document.createElement(ele);
  el.classList.add(myClass);
  el.innerHTML = html;
  parent.append(el);
  return el;
}

function updateMessage(html){
  message.innerHTML = html;
}

function startSpin(){
  game.coins --;
  updateMessage(`You have ${game.coins} left.`);
  game.inPlay = true;
  spinner=500;
  ///console.log('Spinning ' + game.inPlay);
  for(let i =0;i<game.total;i++){
    game.main[i].mover = Math.floor(Math.random()*50)+10;
  }
}

```

```

    }

    game.ani = requestAnimationFrame(spin);
}

function spin(){
    spinner--;
    if(spinner<=0){
        stopGamePlay();

    }

    let holder = [];
    for(let i=0;i<game.total;i++){
        let el = game.main[i];
        let elY = el.offsetTop;
        if(el.mover > 0){
            el.mover--;
            elY += game.speed;
            if(elY > -150){
                elY -= 100;
                const last = el.lastElementChild;
                el.prepend(last);
            }
            if(el.mover == 0 && elY % 50 != 0){
                el.mover++;
            }

            el.style.top = elY + 'px';
        }else{
            let viewEl = el.children[2];
            let outputVal = elY == -200 ? viewEl.faceValue : '-';
            let tempObj = {
                'txt' : viewEl.faceValue,
                'elY' : elY,
                'outputV' : outputVal,
                'output' : viewEl.textContent
            }
            holder.push(tempObj);
        }
    }

    if(holder.length >= game.total){
        stopGamePlay();
        holder.sort();
    }
}

```

```

    console.log(holder);
    const myObj = {};
    holder.forEach((val)=>{
        if(val.outputV !== '-'){
            if(myObj[val.outputV]){
                myObj[val.outputV]++;
            }else{
                myObj[val.outputV] = 1;
            }
        }
    })
    payout(myObj);

}

if(game.inPlay){
    game.ani = requestAnimationFrame(spin);
}
}

function payout(score){
    for(const prop in score){
        let val = Number(score[prop]); //how many occurrences
        console.log(prop + ' x ' + val);
        let basePay = game.total / 2;
        if(val >= 2){
            let totalPaid = Math.floor(val * basePay);
            if(prop === '2s'){
                console.log('You Got more than 2 - 2s');
                totalPaid *= 5;
            }
            game.coins += totalPaid;
            let html = `Matched item ${prop} X ${val} Payout ${totalPaid} Coins
${game.coins}`;
            updateMessage(html);
        }
    }
}
}

```

```
function stopGamePlay(){  
  game.inPlay = false;  
  cancelAnimationFrame(game.ani);  
  btn.textContent = 'SPIN';  
  btn.style.backgroundColor = 'green';  
}
```

## 10 JavaScript Slot Machine Final Code Tweaks and Updates

Improvement of visuals, testing of game play to ensure proper functionality.  
Update to the global game object values to test dynamic content and game play.  
Adding of icons for more appealing game visuals, use fo colors to add more appeal to game play.

Icons to select to easily add images to the projects

<https://fontawesome.com/icons?d=gallery>

CDN to bring JavaScript Code for FontAwesome into project

<https://cdnjs.com/libraries/font-awesome>

Google Fonts select from over 1000 fonts

<https://fonts.google.com/>

### Lesson Challenge

1. Improve visuals for project adding icons and fonts
2. Make CSS adjustments to position elements and information
3. Test and ensure the application is working

```

<!DOCTYPE html>
<html>
  <head><title>JavaScript DOM Project</title>
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.2/css/all.min.css"
integrity="sha512-HK5fgLBL+xu6dm/Ii3z4xh1SÜyZgTT9tuc/hSrtw6uzJOvgRr2a9jyxxTlely+B+xFam
JKVSTbpM/CuL7qxO8w==" crossorigin="anonymous" />
    <style>
      @import
url('https://fonts.googleapis.com/css2?family=Hanalei&family=Roboto+Slab&display=swap'
);
      @import
url('https://fonts.googleapis.com/css2?family=Shadows+Into+Light&display=swap');
      * {
        box-sizing: border-box;
      }
      body{
        font-family: 'Roboto Slab', serif;
      }
      .box{
        font-family: 'Hanalei', cursive;
        position: relative;
        width:100px;
        height:100px;
        border: 1px solid black;
        top:0;
        left:0;
        font-size: 3em;
        line-height: 100px;
        text-align: center;
      }
      .wheel{
        position: absolute;
        left:0px;
        top:0px;
      }

      .gameArea {
        position: absolute;
        left:0px;
        top:80px;
        width:300px;

```

```
height: 103px;
border: 1px solid #333;
overflow: hidden;

}

.topMessage{
  position: absolute;
  left:50%;
  top:10px;
  background-color: black;
  width:80%;
  height:60px;
  color:white;
  padding:5px;
  font-size: 2em;
  font-family: 'Shadows Into Light', cursive;
  transform: translate(-50%,0);
  text-align: center;
  line-height: 40px;
}

.message{
  text-align: center;
  font-size:1em;
  color:white;
}

.btn {
  margin: auto;
  width:80%;
  display: block;
  padding:10px;

  color:white;
  font-size: 1.5em;
  border-radius: 25px;
}

.myDiv{
  padding:20px;
  height:200px;
  background-color:dimgrey;
  width:50%;
  margin:190px auto;
}
```

```

</style>
</head>
<body>
  <div class="myDiv">Hello World</div>
  <script src="apps3.js"></script>
</body>
</html>

const output = document.querySelector('.myDiv');

const iconImages = ['<i class="fas fa-heart"></i>', '<i class="fas fa-dice-d20"></i>', '<i class="far fa-life-ring"></i>', '<i class="fas fa-apple-alt"></i>', '<i class="fas fa-lemon"></i>', '<i class="fas fa-pepper-hot"></i>'];

output.innerHTML = '';
const messageTop = makerElement(output, 'div', 'JavaScript Slot Machine', 'topMessage');
const gameArea = makerElement(output, 'div', '', 'gameArea');
const btn = makerElement(output, 'button', 'SPIN', 'btn');
const message = makerElement(output, 'div', 'message', 'message');

const game =
{total:5, inPlay:false, coins:100, speed:15, totItems:iconImages.length, main:[]};
let spinner = 500;

window.addEventListener('DOMContentLoaded', init);
btn.addEventListener('click', (e) => {
  if (btn.textContent === 'SPIN' && !game.inPlay) {
    btn.textContent = 'STOP';
    btn.style.backgroundColor = 'red';
    spinner = 500;
    startSpin();
  } else {
    stopGamePlay();
  }
})

function init() {
  //console.log('ready');

```

```

    btn.style.backgroundColor = 'green';
    gameArea.style.width = game.total * 100 + 'px';
    let leftPos = (document.body.clientWidth - (game.total * 100)) / 2;
    ///console.log(leftPos);
    gameArea.style.left = leftPos + 'px';

    for(let i=0;i<game.total;i++){
        game.main[i] = makerElement(gameArea,'div','', 'wheel')
        for(let x=0;x<game.totItems;x++){
            const el = makerElement(game.main[i], 'div', iconImages[x], 'box');
            let myColor = x > 2 ? 'red' : 'blue';
            if(x==0){myColor = 'purple';}
            if(x>4){myColor = 'green';}
            if(x==4){myColor = 'orange';}
            el.style.color = myColor;
            el.faceValue = x+1;
        }
        game.main[i].style.left = i * 100 + 'px';
    }
}

function makerElement(parent,ele,html,myClass){
    const el = document.createElement(ele);
    el.classList.add(myClass);
    el.innerHTML = html;
    parent.append(el);
    return el;
}

function updateMessage(html){
    message.innerHTML = html;
}

function startSpin(){
    game.coins --;
    updateMessage(`You have ${game.coins} left.`);
    game.inPlay = true;
    spinner=500;
    ///console.log('Spinning ' + game.inPlay);
    for(let i =0;i<game.total;i++){
        game.main[i].mover = Math.floor(Math.random()*150)+10;
    }
}

```

```

    }

    game.ani = requestAnimationFrame(spin);
}

function spin(){
    spinner--;
    if(spinner<=0){
        stopGamePlay();

    }

    let holder = [];
    for(let i=0;i<game.total;i++){
        let el = game.main[i];
        let elY = el.offsetTop;
        if(el.mover > 0){
            el.mover--;
            elY += game.speed;
            if(elY > -150){
                elY -= 100;
                const last = el.lastElementChild;
                el.prepend(last);
            }
            if(el.mover == 0 && elY % 50 != 0){
                el.mover++;
            }

            el.style.top = elY + 'px';
        }else{
            let viewEl = el.children[2];
            let outputVal = elY == -200 ? viewEl.faceValue : '-';
            let tempObj = {
                'txt' : viewEl.faceValue,
                'elY' : elY,
                'outputV' : outputVal,
                'output' : viewEl.textContent
            }
            holder.push(tempObj);
        }
    }

    if(holder.length >= game.total){
        stopGamePlay();
        holder.sort();
    }
}

```

```

        console.log(holder);
        const myObj = {};
        holder.forEach((val)=>{
            if(val.outputV !== '-'){
                if(myObj[val.outputV]){
                    myObj[val.outputV]++;
                }else{
                    myObj[val.outputV] = 1;
                }
            }
        })
        payout(myObj);

    }

    if(game.inPlay){
        game.ani = requestAnimationFrame(spin);
    }
}

function payout(score){
    for(const prop in score){
        let val = Number(score[prop]); //how many occurrences
        console.log(prop + ' x ' + val);
        let basePay = game.total / 2;
        if(val >= 2){
            let totalPaid = Math.floor(val * basePay);
            if(prop === '2'){
                console.log('You Got more than 2 - 2s');
                totalPaid *= 5;
            }
            game.coins += totalPaid;
            let html = `Matched item ${prop} X ${val} Payout ${totalPaid} Coins
${game.coins}`;
            updateMessage(html);
        }
    }
}

function stopGamePlay(){
    game.inPlay = false;
    cancelAnimationFrame(game.ani);
    btn.textContent = 'SPIN';
}

```

```
btn.style.backgroundColor = 'green';  
}
```