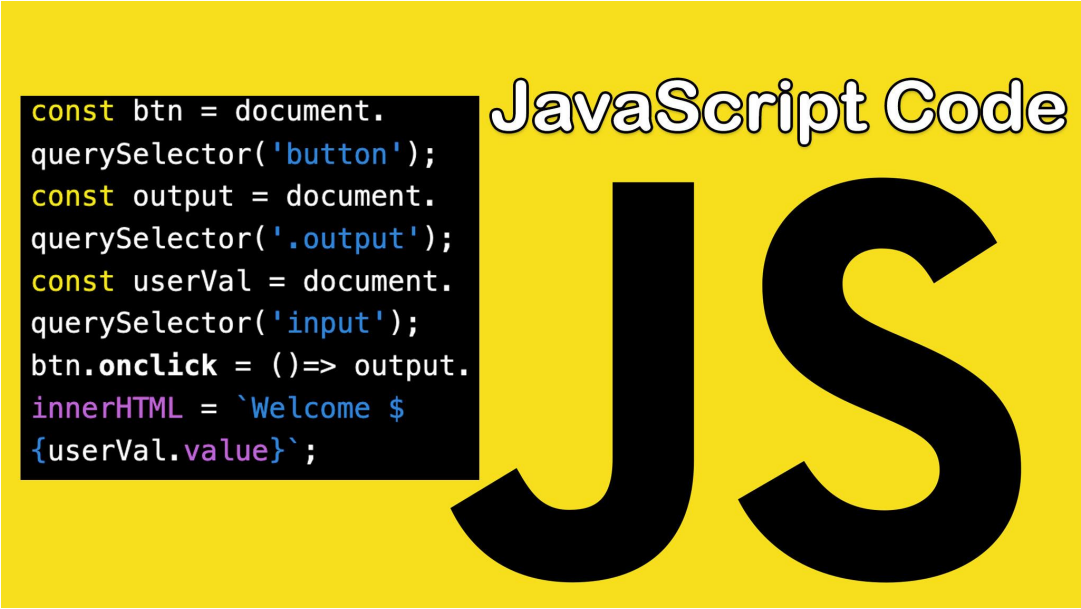
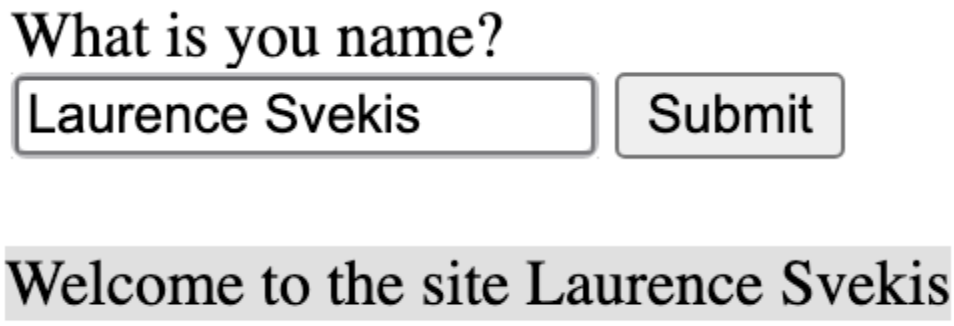
JavaScript DOM coding exercises



Do you want to learn more about JavaScript and how elements work. They are objects which can hold values just like any other objects in JavaScript. The three exercises below will demonstrate creating page elements with JavaScript, how to add values to the element objects, and how to update page elements. There is also an exercise on how they work with the page element objects, and the difference between function expressions and function declarations.

Laurence Svekis https://basescripts.com/

Web page dynamic welcome message



Create a page welcome message :

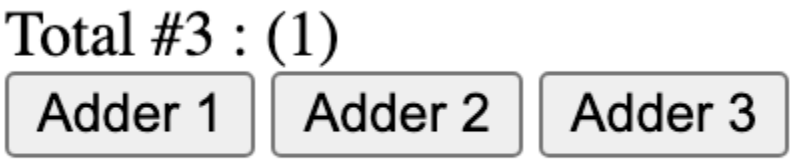
1. Get the user name in the input field
2. When the button is pressed add an event that get the user name and creates a welcome message on the page
3. Add a check to ensure the length of the input is larger than 3 characters long

```
<!DOCTYPE html>
<html>
<head>
   <title>JavaScript Course</title>
</head>
<body>
<div class="output">What is you name?</div>
<input type="text" name="userName">
<button>Submit</button>
<script src="app.js"></script>
</body>
```

```
</html>

const output = document.querySelector('.output');
const userVal =
document.querySelector('input[name="userName"]');
const btn = document.querySelector('button');
userVal.style.borderColor = '#333';
btn.onclick = ()=>{
   if(userVal.value.length > 3){
       const message = `Welcome to the site ${userVal.value}`;
       output.textContent =  message;
       userVal.style.display = 'none';
       btn.style.display = 'none';
   }
   else{
       output.textContent = 'Name length must be 3+ characters';
       userVal.style.borderColor = 'red';
   }
}
```

Function Expression vs Function Declaration

This Counter



Create 3 page counters using different functions, expression, declaration and arrow format. Counters will all work independently as the value of the total is contained within the instance of the function object using this.

Page counters with functions :

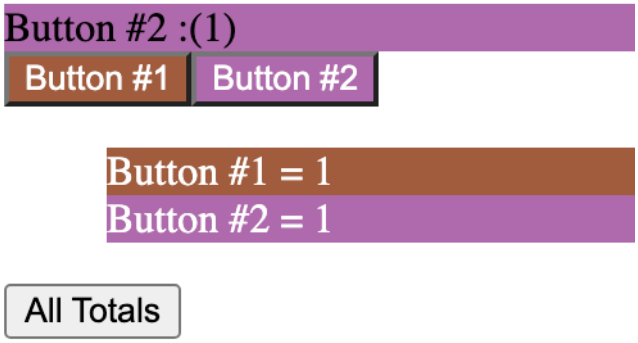
1. Select all the page buttons
2. Create a function expression that will increment the counter on button press
3. Create a function expression with the arrow format that will track and increment the count on the page
4. Create a function declaration that will track and increment the count on the page.

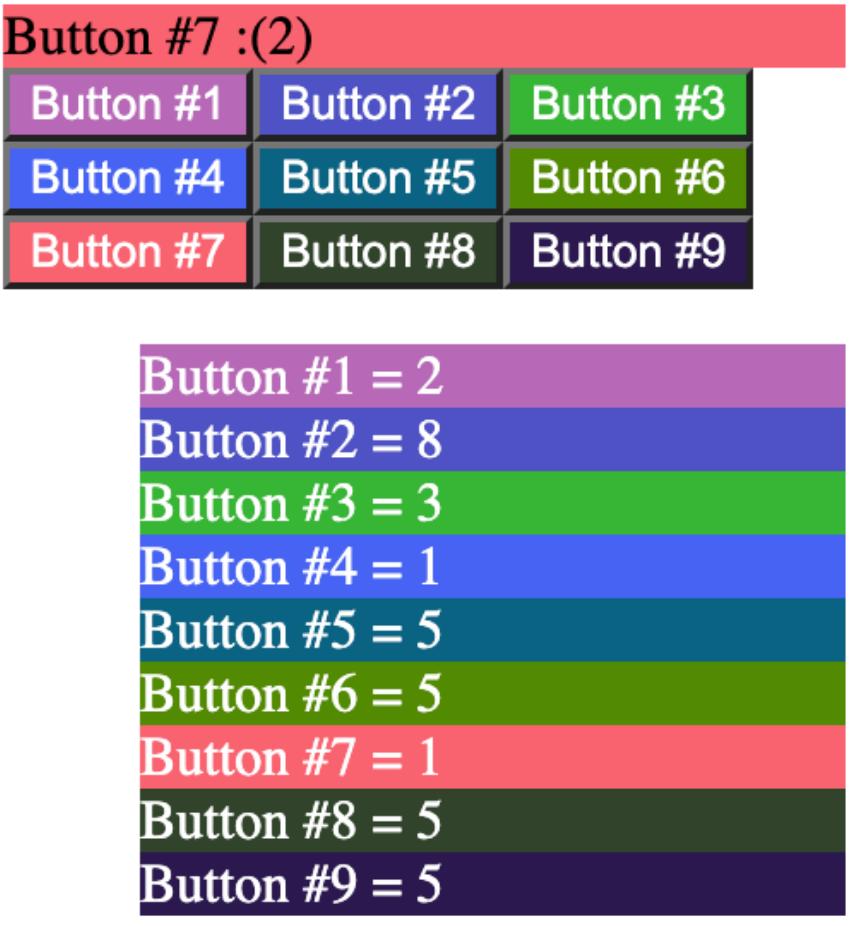
```
<!DOCTYPE html>
<html>
<head>
   <title>JavaScript Course</title>
</head>
<body>
<div class="output"></div>
<button class="btn1">Adder 1</button>
<button class="btn2">Adder 2</button>
```

```
<button class="btn3">Adder 3</button>
<script src="app1.js"></script>
</body>
</html>



const btn1 = document.querySelector('.btn1');
const btn2 = document.querySelector('.btn2');
const btn3 = document.querySelector('.btn3');
const output = document.querySelector('.output');
//Function Expression
const adder2 =  function(){
   if(!this.total) {
       this.total = 1;
   }else{
       this.total++;
   }
   output.textContent = `Total #2 : (${this.total})`;
}
const adder3 = ()=>{
   if(!this.total) {
       this.total = 1;
   }else{
       this.total++;
   }
   output.textContent = `Total #3 : (${this.total})`;
}
btn1.onclick = adder1;
```

```
btn2.onclick = adder2;
btn3.onclick = adder3;

///Function Declaration

function adder1(){
   if(!this.total) {
       this.total = 1;
   }else{
       this.total++;
   }
   output.textContent = `Total #1 : (${this.total})`;
}
```

Dynamic JavaScript DOM page counters Element Objects examples





Dynamically create page buttons that can be used to count totals separately. Create a button to output all the result totals. Only using JavaScript NO HTML elements

Dynamic Page counters with JavaScript :

1. Create a global object to set the number of buttons to be created
2. Create a main page element that will contain all the new elements
3. Create a function to create page elements, adding the element type to a parent object. Include a parameter in the function for the inner content of the element.
4. Loop through and create all the buttons, set a default total of 0 for each one. On click create a function that will update and output the current value of this element total. You can add styling to the buttons
5. Add a class of “btn” to each new button so that they can be distinguished from the main total tally button. Create a button that will output all the current button total results.
6. When the tally button is clicked create a function that will select all the elements with a class of “btn” and loop through them.
7. For each element with the class of “btn” create a new list item element, output that into an unordered list on the main page. The list item contents should show the element total and the element textContent. You can also select the style to match the button style property values.

```
<!DOCTYPE html>
<html>
<head>
   <title>JavaScript Course</title>
</head>
<body>
<script src="app2.js"></script>
</body>
</html>
```

```
const val = {
   num : 2
};
const main =  elemaker('div',document.body,'');
const output =  elemaker('div',main,'');
for(let i=0;i<val.num;i++){
   const btn =  elemaker('button',main,`Button #${i+1}`);
   btn.total = 0;
   btn.style.backgroundColor =
'#'+Math.random().toString(16).slice(2,8);
   btn.style.color = 'white';
   btn.classList.add('btn');
   btn.onclick = adder;
}
const output1 =  elemaker('div',main,'');
const btnMain = elemaker('button',main,`All Totals`);
btnMain.onclick = ()=>{
   const btns = document.querySelectorAll('.btn');
   output1.innerHTML = '';
   const ul = elemaker('ul',output1,'');
   btns.forEach((ele,index)=>{
       const li = elemaker('li',ul,`${ele.textContent} =
${ele.total}`);
       li.style.backgroundColor = ele.style.backgroundColor;
       li.style.color  = ele.style.color;
   })
}
```

```
function elemaker(eleT,parent,html){
   const ele =  document.createElement(eleT);
   ele.innerHTML = html;
   return parent.appendChild(ele);
}

function adder(e){
   this.total++;
   output.style.backgroundColor= this.style.backgroundColor;
   output.innerHTML = `${this.textContent} :(${this.total})`;
}
```

DOM create Page elements adding style

Create interactive elements that can store the current color value input an array, for later use. Also creates buttons to update the body background color to the value of the button text. Color style and events with Dynamic Elements DOM.

Page elements Events and Style Exercise :

1. Using JavaScript create two buttons on the page.
2. Add a click event to the first button, that when pressed should list out all the stored color values from the holding array. When pressed create page elements that have the color value from the holding array, and are clickable elements. Once clicked the element should be removed from the page.
3. The second button should store the color value from the input field into a holding array.
4. The second button should also create a page element button, that has the text content of the input field color value.
5. On change of the input value update the background color to match the input field value.

```
<!DOCTYPE html>
<html>
<head>
   <title>JavaScript Course</title>
</head>
<body>
   <input type="color">
   <script src="app4.js"></script>
</body>
</html>

const myInput = document.querySelector('input[type="color"]');
console.log(myInput);
const holder = [];
const main = document.createElement('div');

const btn = document.createElement('button');
const btn2 = document.createElement('button');
btn.textContent = "Save Color";
btn2.textContent = "List Saved Colors";
document.body.prepend(btn);
document.body.prepend(btn2);
document.body.prepend(main);
console.log(main);

btn2.onclick = ()=>{
   holder.forEach((ele)=>{
       const span = document.createElement('span');
```

```
        main.append(span);
        span.style.width = '50px';
        span.style.height = '50px';
        span.style.display = 'inline-block';
        span.style.backgroundColor = ele;
        span.onclick = ()=>{
            span.remove();}
    })
}


btn.onclick = ()=>{
    holder.push(myInput.value);
    //main.textContent = holder.toString();
    const btn1 = document.createElement('button');
    main.append(btn1);
    btn1.textContent = myInput.value;
    btn1.style.backgroundColor = myInput.value;
    btn1.style.color = 'white';
    btn1.onclick = ()=>{
        document.body.style.backgroundColor = btn1.textContent;
    }
}


myInput.onchange = (e)=>{
    console.log(e.target.value);
    console.log(myInput.value);
```

```
    document.body.style.backgroundColor = myInput.value;
    ///console.log(this.value);
}
```

Slider from Dynamic Elements using JSON data

Using JSON data, load the json file when the DOM content is loaded. Create the page slides dynamically with JavaScript code. Add interaction to navigate between slides listening for clicks on the buttons.

JSON slides navigate slides with JavaScript Exercise :

1. Select the main slider element as a variable named slider in JavaScript.
2. Create a function that gets invoked once the DOM content loads. Give the function a name of adder()
3. Within adder() create the slides, using the data from the JSON file apply the text title, html content, and styling to the slide elements. Add them to the page.

4. If it's the first item in the data then add the class of active to the first one only
5. Add an event listener to the buttons, when clicked check if it contains the next class, create a function named mover() to handle the result
6. Get the current element with the active class. Using traversing, get the next sibling to the current element, if no next element exists then select the first child of the parent slider element.
7. Get the previous element, if it's null then select the last child of the parent slider.
8. Remove the active class for the current element, add the active class to the new element either previous or next.

HTML and CSS

```
<!DOCTYPE html>
<html>

<head>
    <title>JavaScript Course</title>
    <style>
        * {
            box-sizing: border-box;
        }

        .main {
            width: 90vw;
            margin: auto;
            background-color: black;
            position: relative;
```

```
    height: 300px;
    overflow: hidden;
    color:white;
}

.slide {
    text-align:center;
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: blue;
    opacity: 0;
}

.slide.active {
    opacity: 1;
}

.btn {
    position: absolute;
    bottom: 2rem;
    padding: 10px;
    background-color: rgba(0, 0, 0, 0.5);
    border-radius: 10px;
    font-size: 0.9em;
    color: white;
```

```
            cursor:pointer;
        }
        .btn:hover{
            background-color: rgb(0, 0, 0);
        }

        .next {
            right: 3rem;
        }

        .prev {
            left: 3rem;
        }
    </style>
</head>

<body>
    <div class="main">
        <div>
            <div class="slider">
            </div>
            <div class="prev btn">< Prev</div>
            <div class="next btn">Next ></div>
        </div>
    </div>
    <script src="app5.js"></script>
</body>
```

```
</html>
```

JAVASCRIPT CODE

```
const slider = document.querySelector('.slider');
const btns = document.querySelectorAll('.btn');
window.addEventListener('DOMContentLoaded',init);
btns.forEach(btn =>{
   btn.onclick = ()=>{
       console.log('clicked');
       mover(btn.classList.contains('next'));
   }
})

function mover(dir){
   const cur = document.querySelector('.active');
   cur.classList.remove('active');
   const nex = cur.nextElementSibling ||
slider.firstElementChild;
   const pre = cur.previousElementSibling ||
slider.lastElementChild;
   const newActive = (dir) ? nex : pre;
   newActive.classList.add('active');
}

function init(){
   //console.log('ready');
   fetch('app5.json')
```

```
        .then(response => response.json())
        .then(data =>{
            adder(data);
        })
}

function adder(data){
    //slider
    data.forEach((item,index)=>{
        const slide = document.createElement('div');
        slide.classList.add('slide');
        const title = document.createElement('h2');
        title.textContent = item.title;
        title.style.textAlign = 'center';
        if(index==0) slide.classList.add('active');
        const div  = document.createElement('div');
        div.innerHTML = item.html;
        slide.append(title);
        slide.append(div);
        slide.style.backgroundColor = item.back;
        slide.style.color = item.color;
        slider.append(slide);
    })
}
```

JSON file

```
[
    {
```

```
        "title": "slide 1",
        "html": "<h1>Laurence</h1>",
        "back": "purple",
        "color": "white"
    },
    {
        "title": "slide 2",
        "html": "<h1>Svekis</h1>",
        "back": "blue",
        "color": "white"
    },    {
        "title": "slide 4",
        "html": "<h1>Svekis</h1>",
        "back": "blue",
        "color": "white"
    },
    {
        "title": "slide 3",
        "html": "<h1>JavaScript</h1>",
        "back": "yellow",
        "color": "black"
    }
]
```