**Lab 9: JavaScript and the DOM**

The aim of this lab is to use JavaScript to manipulate the DOM within a web page.

We will follow the approach of defining our JavaScript in a separate file from our HTML page (and any CSS style files).

In fact, we will build on the JavaScript we have already been writing and add functions to work with the Book class and the array of Books you have already created.

The page will have a basic structure defined in HTML. However, we will use JavaScript to populate the title of the page and then when the user requests it we will use JavaScript to generate a list of books to be displayed within the page.

Note you will need to make sure all the html and JavaScript files are now in the html directory you created for the Linux box and shared between that VM and your host computer using Vagrant.

# Step 1:

Create (or edit) the index.html page to resemble this:

A screenshot of a cell phone

Description automatically generated

Note it uses Bootstrap and the CSS style file from earlier labs.

An example of what this might look like in the html file is given below:

<html>

<head>

<title>Hello World</title>

<meta charset=UTF-8" />

<!-- Bootstrap CSS -->

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" crossorigin="anonymous">

<!-- Custom CSS -->

<link rel="stylesheet" href="bookshop-styles.css">

<script src="bookshop.js">

</script>

</head>

<body>

<h1>Welcome Placeholder</h1>

<p>

This independent bookshop aims to give you the best books from publishers whether they are small or large.

</p>

<p>

Browse our virtual shelves and enjoy!

</p>

<form>

<input type="button" value="click to show books" />

</form>

<ul>

<li>Click button to show books ...</li>

</ul>

</body>

</html>

Style file:

form {

border: 1px solid blue;

padding: 5px;

width: 50%;

}

h1 {

color: green;

font-family: verdana;

font-size: 200%;

text-align: center;

}

# Step 2:

Now modify the body element such that it is defined as follows:

<body onload="setup()">

This indicates that the setup() function should be run when the body is loaded.

This function should replace the ‘Welcome Placeholder’ text with a Welcome message generated by the function.

So that the page now looks like this:

A screenshot of a cell phone

Description automatically generated

We will break this process down into two functions based on the Single Responsibility Principle. That is one function will provide the string to display (this means it could be loaded from a database or property file) and one function will handle updating the DOM within the web page.

We will therefore have:

function getWelcomeMessage() {

return "Welcome to John's Bookshop";

}

function setup() {

let heading = document.getElementsByTagName("h1")[0];

heading.innerHTML = getWelcomeMessage();

}

# Step 3:

Make the button work

Our final step is to make it so that when the user clicks on the button., the books within the bookstore are listed, for example:

A screenshot of a cell phone

Description automatically generated

To do this we will need specify that a function should be called when the button is click, for example:

<form>

<input type="button" value="click to show books" onclick="showBooks()" />

</form>

In this case the function is called showBooks() and will be defined in the bookshop.js file along with all the other bookshop elements. The function needs to:

1. Get hold of the existing unordered list element (ul)

let listItem = document.querySelector("ul");

1. Remove the current content

let range = document.createRange();

range.selectNodeContents(listItem);

range.deleteContents();

1. Add each book in the array of books as a new list element (li element), for example

for (let book of books) {

listValue = document.createElement("li");

listValue.textContent = book;

listItem.appendChild(listValue);

}

An example of what this function might look like is given below:

**function** showBooks() {

**let** listItem = document.querySelector("ul");

**let** range = document.createRange();

range.selectNodeContents(listItem);

range.deleteContents();

**for** (**let** book **of** books) {

listValue = document.createElement("li");

listValue.textContent = book;

listItem.appendChild(listValue);

}

}