

Shopping Cart Exercise

This exercise should be discussed in the class. In this exercise you will practice how to implement a simplified shopping cart by using JavaScript. You will create a shopping cart for an online purchase order system of books. The stock information is provided by an external XML file **books.xml**.

You can find all the required files for this exercise on Moodle.

You can find a solution of this exercise on the course Github repository.

Add Books to the Shopping Cart Title Author Price Add Item Item Title Price Quantity Total Cost: 0.00 Empire Burlesque Bob Dylan 10.90 Add 1 Hide your heart Bonnie Tyler 19.90 Add Greatest Hits Dolly Parton 29.90 Add Still got the blues Gary Moore 50.20 Add Eros Ramazzotti 35.90 4 Eros Add One night only Bee Gees 10.90 Add Sylvias Mother Dr.Hook 18.10 Add Clear Cart Check Out Maggie May **Rod Stewart** 28.50 Add Andrea Bocelli Romanza 12.80

The outcome of this exercise should look similar to the following:

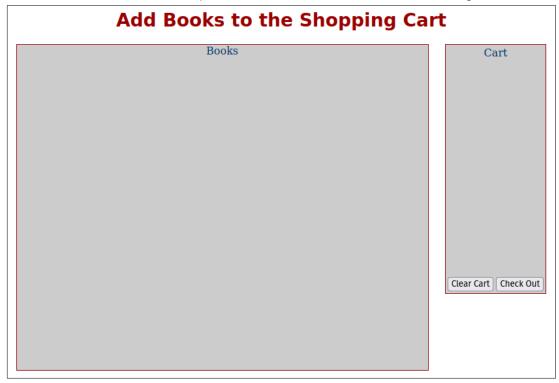
Get ready

- 1. Create or open a website project in your preferred IDE.
- 2. Create a new HTML file **ShoppingCart.html** and attach the given CSS stylesheet **shopping-cart.css** to the page.
- 3. Create page elements on **ShoppingCart.html** by using the pre-defined styles. Here is the source code the comments explains the code:



```
<h1>Add Books to the Shopping Cart</h1>
<div id="txtContainer">
 <div class="display products" id="txtBookInfo">Books</div>
 <div class="display">
   <div id="txtCart">Cart</div>
      <input type="button" value="Clear Cart" onclick="clearCart();" />
      <input type="button" value="Check Out" onclick="checkOut();" />
      </div>
</div>
```

4. The outcome of this part should look similar to the following:





Load stock information from books.xml

- 1. Create a new JavaScript file **shopping-cart.js** in your server folder **Shopping Cart Exercise**, and attach it to **ShoppingCart.html**.
- 2. Inside file **shopping-cart.js**, define function **loadXMLFile(xmlFile)** which takes an XML file as a parameter and converts it into an XML Document object as the return object.

```
function loadXMLFile(xmlFile) {
    xmlhttp = new XMLHttpRequest();

    //Load the XML file and return an XML object
    xmlhttp.open("GET", xmlFile, false);
    xmlhttp.send();
    return (xmlhttp.responseXML);
}
```

3. Inside file **shopping-cart.js**, define two global variables to hold the shopping cart later:

```
let nextItem = 0; //The index of next item to be added
let currentCart = new Array(); //This array contains the current items in the cart
```

4. Inside file shopping-cart.js, define function displayBooks() which loads the XML file books.xml and save the data into an XML Document object xmlDoc. It then constructs an HTML table that displays the key information for all the books within <div id="txtBookInfo">. The comments explain the code.

```
function displayBooks() {
  xmlDoc = loadXMLFile("books.xml");
  txt = "";
  txt += "ItemTitleAuthorPriceAdd";
  books = xmlDoc.documentElement.getElementsByTagName("B00K");
  for (let i = 0; i < books.length; i++) {</pre>
     txt += "";
     txt += "" + i + "";
     bookChildren = books[i].getElementsByTagName("TITLE");
     txt += "" + bookChildren[0].firstChild.
     nodeValue + "";
     bookChildren = books[i].getElementsByTagName("AUTHOR");
     txt += "" + bookChildren[0].firstChild.nodeValue + "";
     bookChildren = books[i].getElementsByTagName("PRICE");
     txt += "" + bookChildren[0].firstChild.
     nodeValue + "";
     txt += "<img src='book-icon.jpg'/>";
     txt += "<button type='button' id='bntAddTocart'
onclick='addToCart(" + i + ")'>Add to Cart</button>" + "";
     txt = txt + "";
  txt += "";
  document.getElementById("txtBookInfo").innerHTML = txt;
```



5. Inside file shopping-cart.js, call function displayBooks() inside a window.onload event function.

```
window.onload = function () {
    displayBooks();
}
```

6. This part of the outcome should be similar to the following, when a book icon is clicked, it will call function **addToCart(i)** with the item number ,i.e. **i**, as a parameter.

Add Books to the Shopping Cart Item Title Price Add Cart Author 0 Empire Burlesque Bob Dylan 10.90 Add to Cart 1 Hide your heart Bonnie Tyler 19.90 Add to Cart 2 **Greatest Hits** Dolly Parton 29.90 Add to Cart 3 Still got the blues Gary Moore 50.20 Add to Cart 35.90 Eros Eros Ramazzotti Add to Cart 4 One night only Bee Gees 10.90 5 Add to Cart Add to Cart 6 Sylvias Mother Dr.Hook 18.10 Clear Cart Check Out 7 Maggie May Rod Stewart 28.50 Add to Cart 8 12.80 Add to Cart Romanza Andrea Bocelli 29.70



Build the shopping cart

Inside file shopping-cart.js, define function displayCart() which
constructs the shopping cart table from the global array currentCart
and displays it within <div id="txtCart">. The array will hold all the
selected items within the shopping cart.

```
function displayCart() {
  txt = "
  align='center' style='margin:0px;border:#666 solid;'>";
  txt += "ItemTitlePriceQuantity";
  totalCost = 0;
  for (i = 0; i < nextItem; i++) {
     txt += "" + currentCart[i][0] + "";
     txt += "" + currentCart[i][2] + "";
     txt += "" + currentCart[i][3] + "";
     totalCost += currentCart[i][2] * currentCart[i][3];
     txt = txt + "";
  //Display the total cost
  txt += " Total Cost: " +
  parseFloat(totalCost).toFixed(2) + "";
  txt += "";
  //Display shopping cart table in <div id="txtCart">
  document.getElementById("txtCart").innerHTML = txt;
```

2. Inside file **shopping-cart.js**, update the **window.onload()** function to also call the **displayCart()** function:

```
window.onload = function () {
    displayBooks();
    displayCart();
}
```

3. The outcome of this part should look similar to the following:



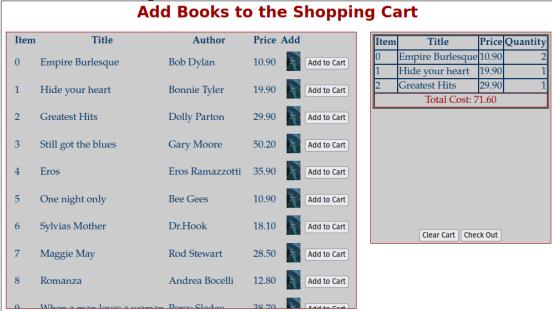


4. Inside file **shopping-cart.js**, define function **addToCart(selectedItem)** which is called when a book icon is clicked. It checks if the selected item has been added once, if YES then its quantity is increased by one, otherwise, it adds a new item to the shopping cart.

```
function addToCart(selectedItem) {
   let addedIndex = -1;
    for (i = 0; i < nextItem; i++) {</pre>
        if (currentCart[i][0] == selectedItem) {
            addedIndex = i;
    if (addedIndex == -1) {
       currentCart[nextItem] = new Array();
       //Put the item details into the shopping cart
       currentCart[nextItem][0] = selectedItem;
       currentCart[nextItem][1] = document.getElementById("title" + selectedItem).
        innerHTML;
       currentCart[nextItem][2] = document.getElementById("price" + selectedItem).
        currentCart[nextItem][3] = 1; //set the quantity
       nextItem += 1;
       currentCart[addedIndex][3] += 1;
   displayCart();
```



5. Try to click on some of the book icons, the outcome should be similar to the following:



Handle the two buttons

 Inside file shopping-cart.js, define function clearCart() which is called when the Clear Cart button is clicked. It clears the current item array and refreshes the shopping cart:

```
function clearCart() {
    //Reinitialise the current item array
    nextItem = 0;
    currentCart = new Array();

    //Refresh the shopping cart
    displayCart();
}
```

 Inside file shopping-cart.js, define function checkOut() which is called when the button Check Out is clicked. It saves the selected items in the shopping cart into an XML document object and clears the shopping cart. It also calls the given PHP file ProcessingOrder.php to save the selected items in an external XML file item.xml (when run in an environment with PHP available).



```
function checkOut() {
   txt = "<ITEMS>";
   for (i = 0; i < nextItem; i++) {
       txt = txt + "<ITEM>";
       txt = txt + "<TITLE>" + currentCart[i][1] + "</TITLE>";
       txt = txt + "<PRICE>" + currentCart[i][2] + "</PRICE>";
       txt = txt + "<QUTANTY>" + currentCart[i][3] + "</QUTANTY>";
       txt = txt + "</ITEM>";
   txt = txt + "</ITEMS>";
   parser = new DOMParser();
   xmlDoc = parser.parseFromString(txt, "text/xml");
   xmlhttp = new XMLHttpRequest();
   //Send the XML string to server
   let url = "ProcesingOrder.php?XMLStr=" + txt;
   xmlhttp.open("GET", url, false);
   xmlhttp.send(null);
   //Clear cart
   clearCart();
```

Compare your completed exercise with the solution on the course Github repository.