Data Representation
Lab 06.2: Functionality on the webpage

Lecturer: Andrew Beatty

Use the webpage from in Lab06.1 to create a webpage that has the functionality to:

1. Create a new book **C**
2. View all the books **R**
3. Update a book **U**
4. Delete a book **D**

We'll read first

Show create.

1. Hide the create form. Modify the <div> surrounding the form so that it is hidden. Give this <div> an id so that it can be discovered in JavaScript.

```
<div id='createUpdateForm' style="display: none">
```

2. Give the create button and the table ids so that they can be discovered by JavaScript.

```
<button id="button-showCreate">
.....
<table class="table" id="bookTable">
```

3. Write a function called showCreate() that hides the create button and the table, and shows the createUpdateForm <div>

```
function showCreate(){
    document.getElementById('button-showCreate').style.display="none"
    document.getElementById('bookTable').style.display="none"
    document.getElementById('createUpdateForm').style.display="block"
  }
```

4. Call the function from when the createButton is clicked.

```
<button id="button-showCreate" onclick="showCreate()">
```

5. Add ids to the spans that surround the words create and update and the two buttons in the form <div>

```
<span id="createLabel">Create a</span> <span id="updateLabel" style=”display:none” >update</span> Car
....
<span><button id="button-doCreate" onclick="doCreate()">Create</button></span>
<span><button id="button-doUpdate" onclick="doUpdate()">Update</button></span>
```

6. Add code to showCreate() that shows the create button and word and hides the update button and word. (Test it)

```
        document.getElementById('createLabel').style.display="inline"
        document.getElementById('updateLabel').style.display="none"

        document.getElementById('button-doCreate').style.display="block"
        document.getElementById('button-doUpdate').style.display="none"
```

Show Update 1

7. Create a showUpdate(buttonElement) function, that shows the form. This function should show the update button and word and hide the create button and word.
8. Call this function from each of the update buttons in the table, "this" is the element that is activated

```
        <td><button onclick="showUpdate(this)">Update</button></td>
```

9. We will be coming back to this function later

Do Create

10. Create a doCreate() function and call it from the create button. Test it with a console.log(XXX)

```
        <td><button id="button-doCreate" onclick="doCreate()">Create</button>
```

```
function doCreate(){
        console.log("creating a book")
}
```

11. Create a function called showViewAll that shows the create button and the table and hides the createUpdateForm. (ie the opposite to the showCreate above)

12. Call this function from doCreate()

```
        function doCreate(){
    showViewAll()
  }
```

13. When you test your code you will notice that the data is still in the form when you click create a second time, we should clear this data. Create a function called clearForm. We will use the querySelector to find the inputs, instead of giving them all ids. (The disabled= false is for later).

```
function clearForm(){
    var form = document.getElementById('createUpdateForm')

    form.querySelector('input[name="id"]').disabled = false
    form.querySelector('input[name="id"]').value  =''
    form.querySelector('input[name="author"]').value=''
    form.querySelector('input[name="title"]').value=''
    form.querySelector('input[name="price"]').value=''
}
```

14. Call this from the doCreate.

```
function doCreate(){
    // other code here
    clearForm()
    showViewAll()
}
```

15. We need to read the book from the form, so create a function to read each of the values from the form and returns a book object populated with those values

```
function getBookFromForm(){
            var form = document.getElementById('createUpdateForm')

             var book = {}
            book.id = form.querySelector('input[name="id"]').value
            book.title = form.querySelector(input[name="title"]').value
            book.author = form.querySelector('input[name="author"]').value
            book.price = form.querySelector('input[name="price"]').value
            console.log(JSON.stringify(book))
            return book
}
```

16. Call it from the doCreate and check it works with a console.log

```
function doCreate(){
            book = getBookFromForm()
            console.log(book)
            clearForm()
             showViewAll()

}
```

17. We will now need to add the book to the table, there is a lot to be done with this. Make a function called addBookToTable and call it from the doCreate.

```
function doCreate(){
            // other code here
           book = getBookFromForm()
           addBookToTable(book)

           clearForm()
            showViewAll()
```

   a. In the function find the table and add a row to it at the end. And add a cell to that row and add the book id to that cell (test this)

```
function addBookToTable(book){
      var tableElement = document.getElementById('bookTable')
      var rowElement = tableElement.insertRow(-1)
// set attribure here
      var cell1 = rowElement.insertCell(0);
      cell1.innerHTML = book.id

}
```

   b. Add the rest of the data into the table (test it)

```
function addBookToTable(book){
     var tableElement = document.getElementById('bookTable')
     var rowElement = tableElement.insertRow(-1)
     // set attribure here
     var cell1 = rowElement.insertCell(0);
     cell1.innerHTML = book.id
     var cell2 = rowElement.insertCell(1);
     cell2.innerHTML = book.titie
     var cell3 = rowElement.insertCell(2);
     cell3.innerHTML = book.author
     var cell4 = rowElement.insertCell(3);
     cell4.innerHTML = book.price
     var cell5 = rowElement.insertCell(4);
     cell5.innerHTML = '<button onclick="showUpdate(this)">Update</button>'
     var cell6 = rowElement.insertCell(5);
     cell6.innerHTML = '<button onclick=doDelete(this)>delete</button>'

  }
```

show update2

18. When we are doing the update we would like the form populated with the details of the book we are updating, with that in mind we will need to do two things
    a. Read the book data from the current row,
    b. Populate the form with that book data.
19. Write a function called getBookFromRow, that takes in a Row element and returns a car object.

```
function getBookFromRow(rowElement){
    var book ={}
    book.id  = rowElement.cells[0].firstChild.textContent
    book.title = rowElement.cells[1].firstChild.textContent
    book.author = rowElement.cells[2].firstChild.textContent
    book.price = rowElement.cells[3].firstChild.textContent
    return book
  }
```

20. Write a function called populateFormWithBook, which takes in a book

```
          function populateFormWithBook(book){
                   var form = document.getElementById('createUpdateForm')
                  form.querySelector('input[name="id"]').disabled = true
                   form.querySelector('input[name="id"]').value  = book.id
                   form.querySelector(input[name="title"]').value= book.title
                  form.querySelector('input[name="author"]').value= book.author
                  form.querySelector('input[name="price"]').value= book.price
 }
```

21. Call these functions in showUpdate() so that the form is populated from the table. The rowElement is the grandparent of the buttonElement that was passed in as a parameter

```
function showUpdate(buttonElement){
....// other code here

    var rowElement = buttonElement.parentNode.parentNode
    // these is a way of finding the closest <tr> which would safer, closest()

    var book = getBookFromRow(rowElement)
    populateFormWithBook(book)
}
```

doUpdate

22. When the user clicks the update button on the form, we will need to do two actions.
    a. Read the book from the form (like we did on doCreate)
    b. Modify the row that contains the book, to do this we will need to find the row, one solution is to add an id to each row, say the book.id. You will also need to modify addToTable, later

```
    <tr id="123">
        <td>123</td>
        <td>zen</td>
        <td>budda</td>
        <td>25,000</td>
        <td><button onclick="showUpdate(this)">Update</button></td>
        <td><button onclick="doDelete(this)">Delete</button></td>

    </tr>
    <tr id="12 G 123">....
```

23. use the function called getBookFromForm, that returns a book
24. Create a function called setBookInRow that takes the car and rowElement and populated the row with the car (the opposite to getBookFromRow)

```
function setBookInRow(rowElement, book){
    rowElement.cells[0].firstChild.textContent= book.id
    rowElement.cells[1].firstChild.textContent= book.title
    rowElement.cells[2].firstChild.textContent= book.authorl
    rowElement.cells[3].firstChild.textContent= book.price
  }
```

25. Put these together in doUpdate

```
function doUpdate(){
    var book = getBookFromForm();
    var rowElement = document.getElementById(book.id)
    setBookInRow(rowElement,book)

    clearForm()
    showViewAll()
}
```

Delete

26. Put a doDelete(this) function into each of the delete buttons (like we did in update)
27. We will use the deleteRow function which takes in the index of the row you wish to delete so we need to find the row and get its index. Write the code for doDelete.

```
function doDelete(buttonElement){
    var tableElement = document.getElementById('bookTable')
    var index = buttonElement.parentNode.parentNode.rowIndex;
    tableElement.deleteRow(index);
}
```

Fix bugs

28. Test the code and notice that new books are not being updated, if you inspect the DOM you will notice that they do not have an id in the row like the other rows. In addBookToTable add a like that will set the id attribute of the new row to the reg

```
rowElement.setAttribute('id',book.id)
```