Shopping List Exercise

Reuben Simpson

Build a Shopping List Application

- Build an application where user can input shopping items and quantity to create a shopping list
- The application displays the up-to-date shopping list at any time
- The application also needs to allow user to delete items

Layout the elements

- Text box's ID = item
- Text box's ID = qty
- <p> ID = shopping-list-display
- Button calls addItem();


```
<html>
  <body>
    <h1>Shopping List</h1>
    <input type="text" id="item" placeholder="Item. e.g. Tomato" /> X
    <input type="number" id="qty" placeholder="Quantity. e.g. 1" />
    <button onclick="addItem()">Add item to list</button>
    <p id="shopping-list-display"></p>
  </body>
</html>
```

Add Script

- Add <script>
- What functions do we need?
  - addItem
  - displayListItems
  - deleteItem

```
<html>
    <body>
    <h1>Shopping List</h1>
    <input type="text" id="item" placeholder="Item. e.g. Tomato" /> x
    <input type="number" id="qty" placeholder="Quantity. e.g. 1" />
    <button onclick="addItem()">Add item to list</button>
    <p id="shopping-list-display"></p>

        <script>
            function addItem() {

            }

            function displayListItems() {

            }

            function deleteItem() {

            }
        </script>
    </body>
</html>
```

Data Structure

```
Array [
   0: { item: "tomato", quantity: 2},
   1: { item: "bread", quantity: 1},
   2: { item: "eggs", quantity: 12},
   3: { item: "milk", quantity: 1},
   4: { item: "coffee", quantity: 1}
]
```

- What data structure do we use to hold the shopping list?
- Where do we declare the variable? (think about scope)

```
<script>
            //shoppingList is an array [] of object { item, quantity }
            const shoppingList = [];

            function addItem() {

            }

            function displayListItems() {

            }

            function deleteItem() {

            }
 </script>
```

addItem

- What is the objective of addItem()?
- Where do we get the values?
- How do we add the item to the shoppingList?

```
function addItem() {
    //Create an Object and then add the object into shoppingList array
    const itemToAdd = {
      item: document.getElementById("item").value,
      quantity: document.getElementById("qty").value,
    };
    shoppingList.push(itemToAdd);
    displayListItems();
  }
```

displayList

- What is the objective of displayListItems()?
- Where do we display the list?
- How do we get the contents of the list?

```
function displayListItems() {
     //clear the display
     document.getElementById("shopping-list-display").innerHTML = "";
     //for each item in the shoppingList array, display item, quantity, and a delete button
     for (let shoppingListEl of shoppingList) {
     //update the HTML to what is currently rendered, plus a new string
     document.getElementById("shopping-list-display").innerHTML += `${shoppingListEl.item} x ${shoppingListEl.quantity}
     }
 }
```

Display the delete icon

- We need to append to each line item: `<span onclick="deleteItem(1);">  </span><br />`
- As an argument, we need to give it the name of the item to delete
- HTML icon ☒ is ❎
  - http://www-db.deis.unibo.it/courses/TW/DOCS/w3schools/charsets/ref_utf_dingbats.asp.html

```
function displayListItems() {
     //clear the display
     document.getElementById("shopping-list-display").innerHTML = "";
     //for each item in the shoppingList array, display item, quantity, and a delete button
     for (let shoppingListEl of shoppingList) {
     //update the HTML to what is currently rendered, plus a new string
     document.getElementById("shopping-list-display").innerHTML += `${shoppingListEl.item} x ${shoppingListEl.quantity}
     <span style="cursor: pointer" onclick="deleteItem(${shoppingList.indexOf(shoppingListEl)})">&#10062</span><br/>`;
     }
 }
```

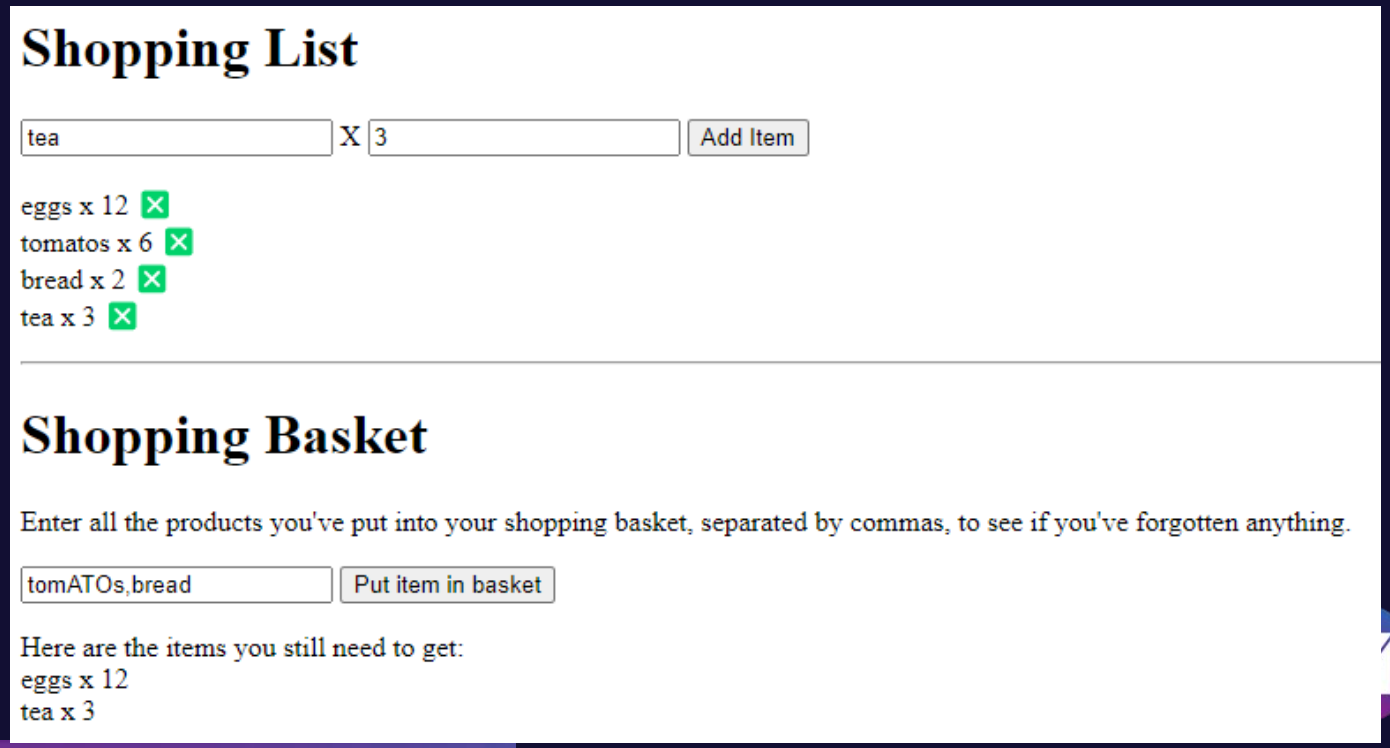
deleteItem

- Passed in itemToDelete as an argument
- How do we find the item in the shoppingList?
- Delete using an array function splice(itemLocation, howManyItems)

```
function deleteItem(itemToDelete) {
    //delete the item use splice method
    shoppingList.splice(itemToDelete, 1);
    displayList();
  }
```

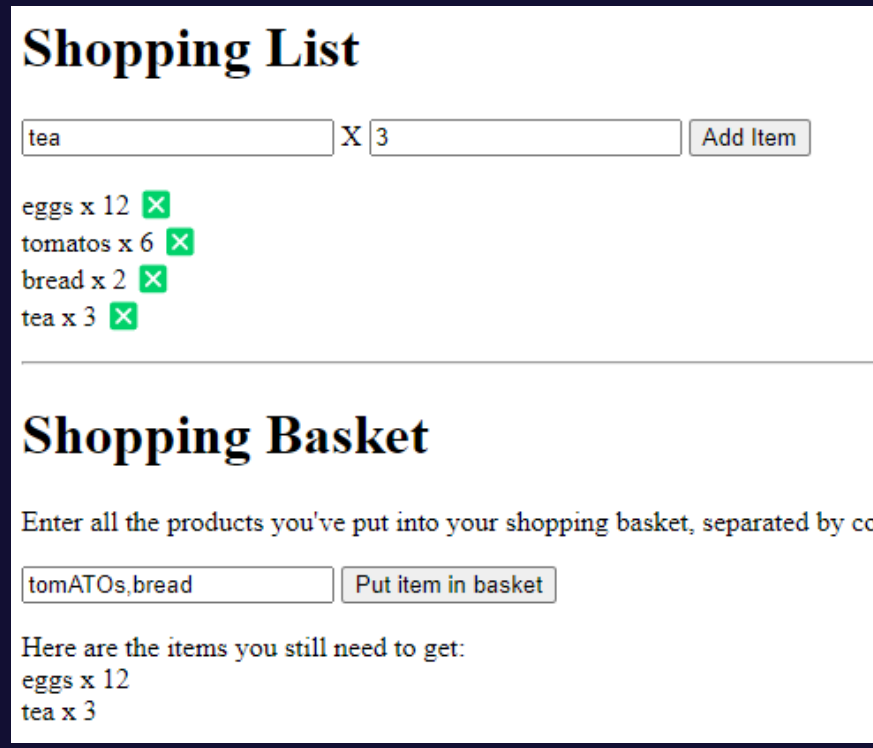
Let’s build a Shopping Basket

- Build underneath your Shopping List application
- Add code that allows users to input items in the current shopping basket, which can include many single-word items separated by a comma (e.g. “tomato,eggs,milk”). You can ignore quantity for this exercise.
- Upon submitting, display the items that the user still has to add into the basket (i.e. items in the shopping list but not in the basket)



Adding extra fields

- Add the HTML elements
- Textbox with an ID=shopping-basket
- Button that onclick calls addToBasket() function
- P with an ID=forgotten-items-display




```
<hr />
 <h1>Shopping Basket</h1>
 <p>
   Enter all the products you've put into your shopping basket, separated by commas,
    to see if you've forgotten anything.
 </p>
 <input type="text" id="shopping-basket" placeholder="e.g. tomato,eggs,milk" />
 <input type="button" value="Add item(s) to basket" onclick="addToBasket();" />
 <p id="forgotten-items-display"></p>
```

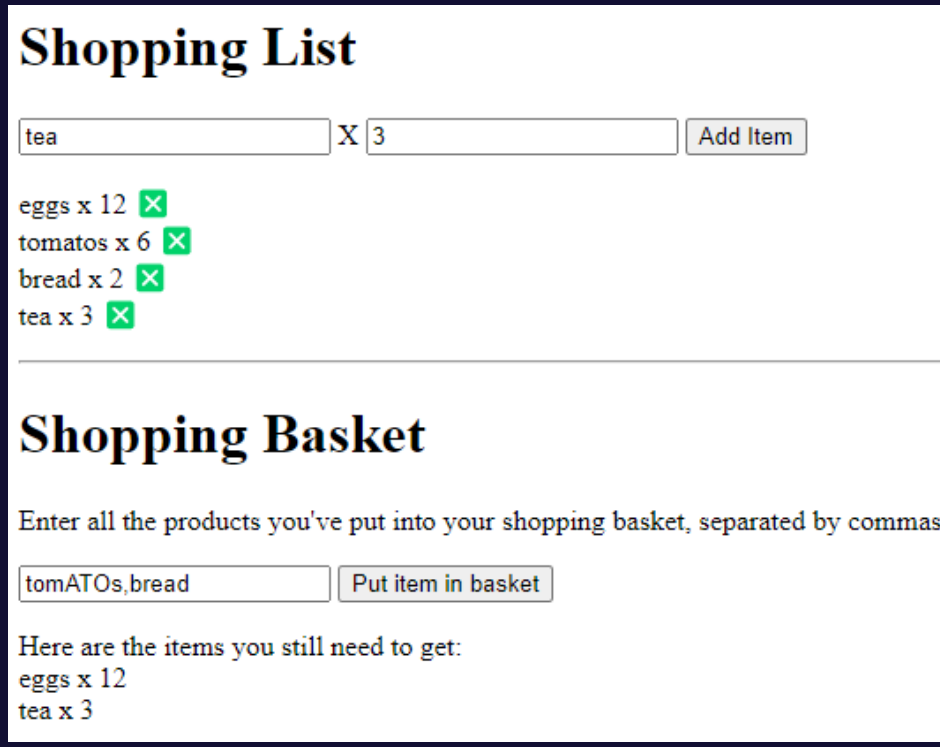
Add Function

- Add function declaration code in appropriate place

```
function addToBasket() {


}
```

- What is the objective of this function?
- What are the steps?



Split user input into an array

- Where do you take the input from user?
- How do you cater for upper case vs lower case?
- What function do you use to split it into an array?

```
//split() splits the inputted form data by commas, and returns a new array
    const basket = document.getElementById("shopping-basket").value.toLowerCase().split(",");
```

Make a list of difference

- Use filter to make a new list that passed a test

```
function addToBasket() {
    //split() splits the inputted form data by commas, and returns a new array
    const basket = document.getElementById("shopping-basket").value.toLowerCase().split(",");

    //filter() returns a new array of elements that contain pass a condition
    const forgottenItems = shoppingList.filter(function (shoppingListEl) {
    // Condition: Check to see if the basket array includes the shopping list item (of the currently looping
         element).
    // If the element does exist in the basket array, do NOT return true and so do NOT add the current element
         to the filtered array (Because of the NOT operator)
    return !basket.includes(shoppingListEl.item.toLowerCase());
  });
```

Displaying the results

```
function addToBasket() {
    //split user input into an array
    const basket = document.getElementById("shoppingBasket").value.toLowerCase().split(",");
       console.log(basket);

       const itemsStillToBuy = shoppingList.filter(function (shoppingListElement) {
         const itemInBasket = shoppingListElement.item.toLowerCase();
         return !basket.includes(itemInBasket);
       });

  //if forgottenItems array is empty, then display a success message
  if (forgottenItems.length === 0) {
      document.getElementById("forgotten-items-display").innerHTML = "------YOU GOT EVERYTHING 😌------";
  } else {
  //if forgottenItems is not empty, then display all items in the forgottenItems array
  document.getElementById("forgotten-items-display").innerHTML = "Here are the items you still need to get:<br/>";
  //display forgottenItems
       for (let forgottenItemEl of forgottenItems) {
       //update the HTML to what is currently rendered, plus a new string
       document.getElementById(
         "forgotten-items-display"
       ).innerHTML += `${forgottenItemEl.item} x ${forgottenItemEl.quantity} <br/>`;
       }
    }
```

MISSION READY

www.missionreadyhq.com

DARE TO DEVELOP

Thank you | Reuben Simpson