Group-Project: ToDo-List

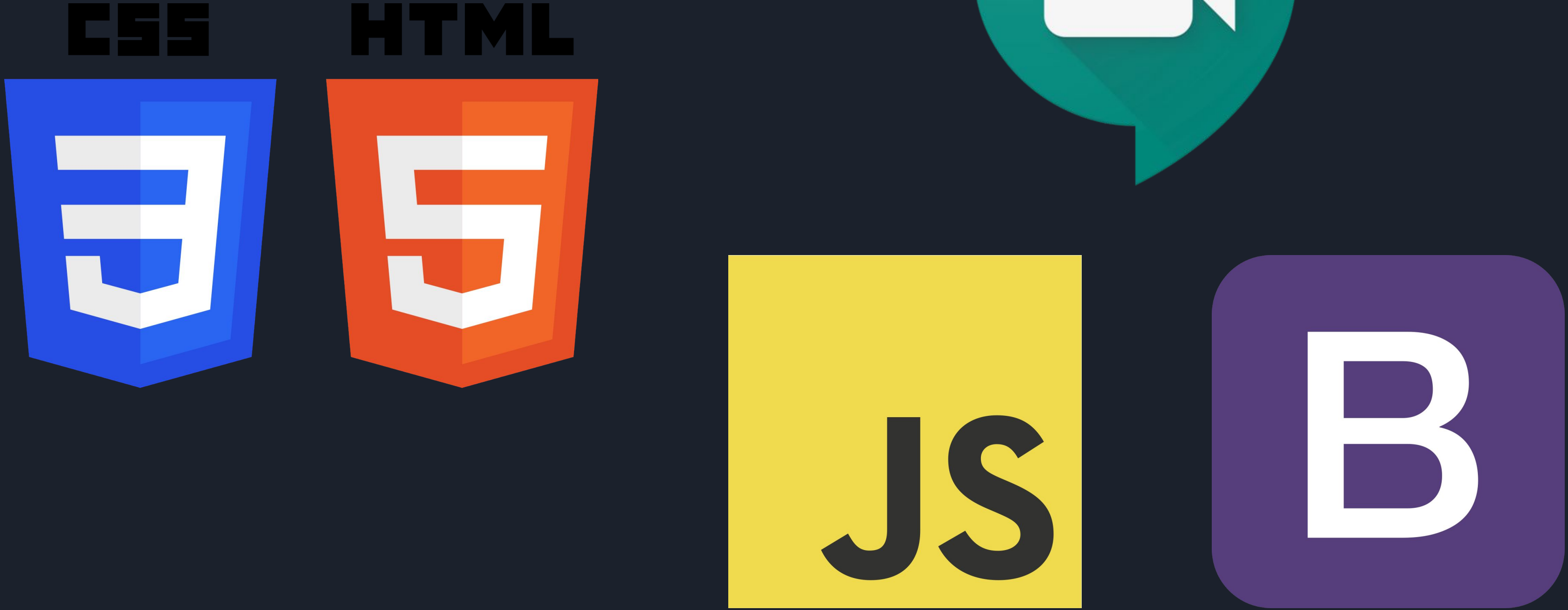
Made possible by
Simone Sorze & Natascha Hof

Context

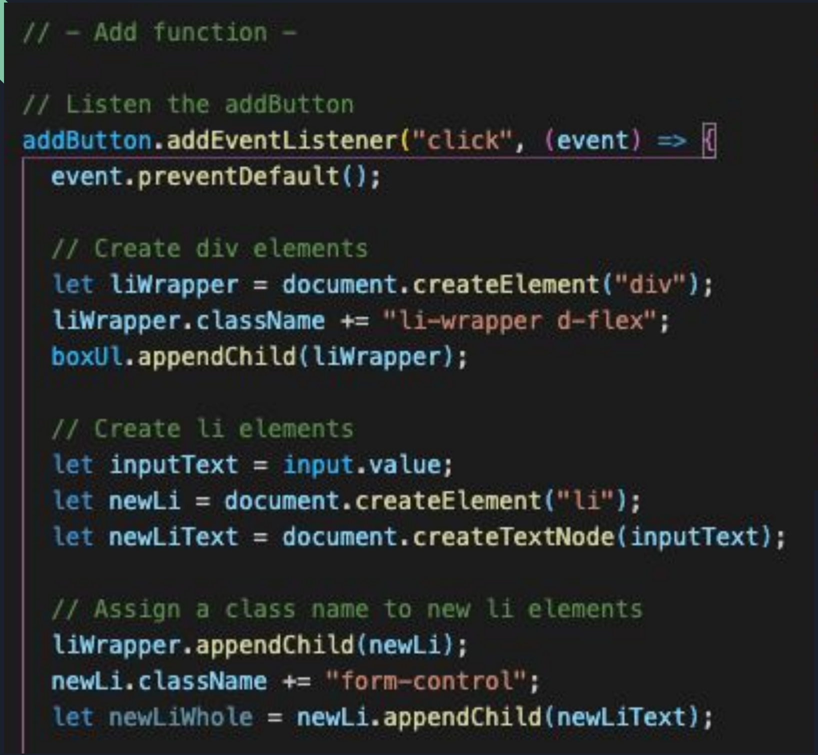
- **Why?**
  - Weekly assignment:
    - Coding a Todo-List
    - Must contain opportunities to add items, delete items and edit items
    - Due date: 14th August 2020

- How?
  - The work has been done in 2 steps:
    - Planning and task assignments
    - Group and individual coding time

The Stack



Code Snippets- Some code (1)



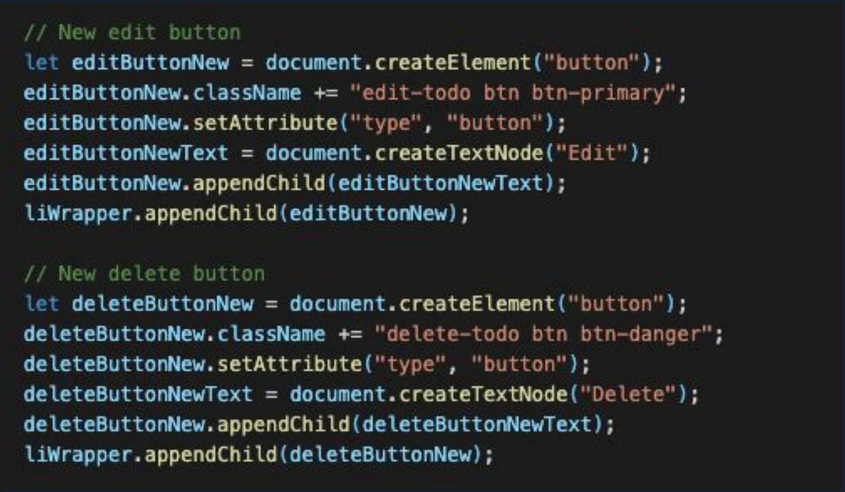
```
// - Add function -

// Listen the addButton
addButton.addEventListener("click", (event) => {
  event.preventDefault();

  // Create div elements
  let liWrapper = document.createElement("div");
  liWrapper.className += "li-wrapper d-flex";
  boxUl.appendChild(liWrapper);

  // Create li elements
  let inputText = input.value;
  let newLi = document.createElement("li");
  let newLiText = document.createTextNode(inputText);

  // Assign a class name to new li elements
  liWrapper.appendChild(newLi);
  newLi.className += "form-control";
  let newLiWhole = newLi.appendChild(newLiText);
```

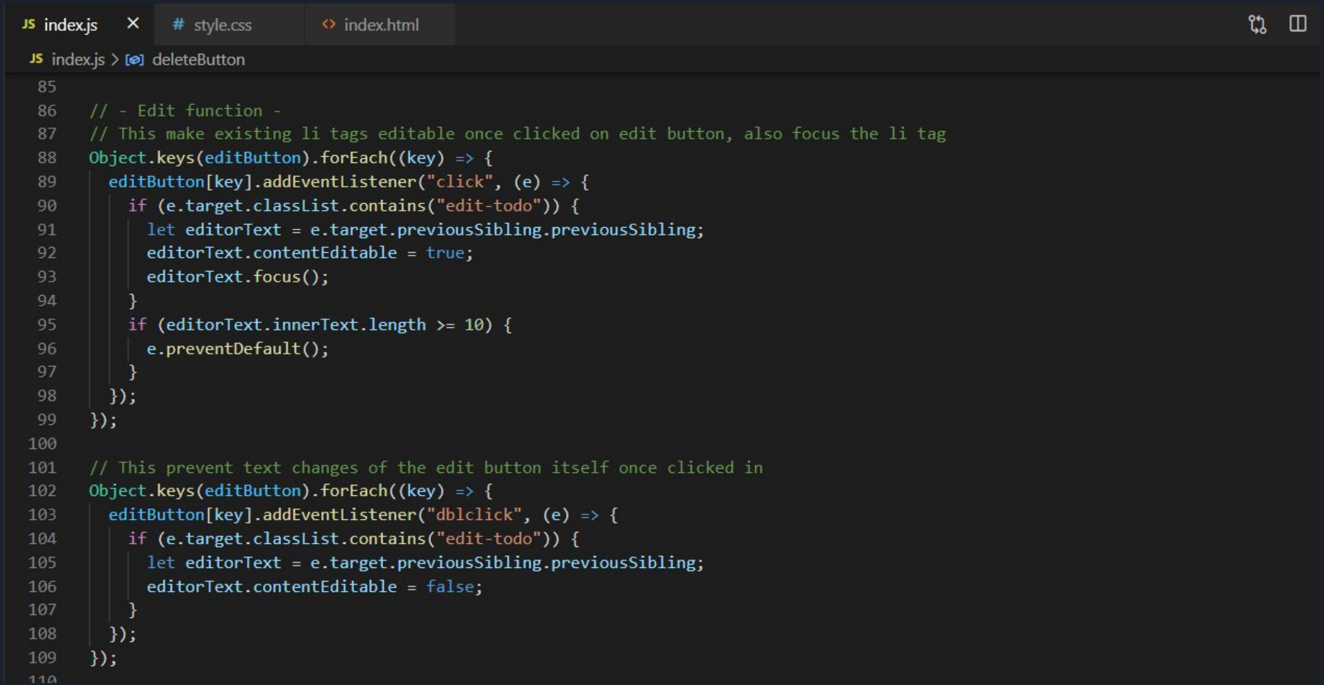


```
// New edit button
let editButtonNew = document.createElement("button");
editButtonNew.className += "edit-todo btn btn-primary";
editButtonNew.setAttribute("type", "button");
editButtonNewText = document.createTextNode("Edit");
editButtonNew.appendChild(editButtonNewText);
liWrapper.appendChild(editButtonNew);

// New delete button
let deleteButtonNew = document.createElement("button");
deleteButtonNew.className += "delete-todo btn btn-danger";
deleteButtonNew.setAttribute("type", "button");
deleteButtonNewText = document.createTextNode("Delete");
deleteButtonNew.appendChild(deleteButtonNewText);
liWrapper.appendChild(deleteButtonNew);
```

+ adding functionalities of delete and edit buttons on new items

Code Snippets - Some code (2)



```javascript

// - Edit function -
// This make existing li tags editable once clicked on edit button, also focus the li tag
Object.keys(editButton).forEach((key) => {
  editButton[key].addEventListener("click", (e) => {
    if (e.target.classList.contains("edit-todo")) {
      let editorText = e.target.previousSibling.previousSibling;
      editorText.contentEditable = true;
      editorText.focus();
    }
    if (editorText.innerText.length >= 10) {
      e.preventDefault();
    }
  });
});

// This prevent text changes of the edit button itself once clicked in
Object.keys(editButton).forEach((key) => {
  editButton[key].addEventListener("dblclick", (e) => {
    if (e.target.classList.contains("edit-todo")) {
      let editorText = e.target.previousSibling.previousSibling;
      editorText.contentEditable = false;
    }
  });
});
```

An example of the Edit function, more details about Edit and Delete functions are explained in our repository file

- https://github.com/Natascha2020/ToDoList/blob/master/index.js -

Live-Demo

https://natascha2020.github.io/ToDoList/

Reflections

Difficulties	Improvements
Making <li> tags editable (usable like inputs)	Complete change from <li> tag to <input> tag (?)
Targeting the right selector (e.g. parentElement.firstChild)	/
Handling Objects in a loop with an event assigned (edit/delete buttons)	/