# Блок 2. Работа с JavaScript в браузере.

## Занятие 1. Работа с DOM через JavaScript

### 1) Написать функцию, которая выводит список всех сотрудников в список <ul>.

Функция **showEmployees(employees)** должна использовать DOM для создания элементов <li> списка <ul>. В каждом элементе должны выводиться данные об одном сотруднике. Должно выводиться имя и фамилия сотрудника. Результат должен помещаться в PLACEHOLDER, заранее добавленный на страницу:

<**div id="employeesPlaceholder"**></**div**>

Перед повторным заполнением плейсхолдер должен очищаться:

**function** *clearEmployeesPlaceholder*() {

***document***.getElementById(***PLACEHOLDER***).innerHTML = '';

}

Также объявите константу PLACEHOLDER, равную **employeesPlaceholder.**

Функцию нужно поместить в файл ui.js и подключить файл к HTML-файлу.

**function** *showEmployees*(employees) {  
 *clearEmployeesPlaceholder*();  
 **const** ul = ***document***.createElement(**"ul"**);  
  
 **for** (**let** employee **of** employees) {  
 **const** li = ***document***.createElement(**"li"**);  
 ul.appendChild(li);  
  
 li.innerHTML = employee.**name**+**" "**+employee.**surname**;  
  
 }  
 ***document***.getElementById(***PLACEHOLDER***).appendChild(ul);  
}

### 2) Запускать функцию показа списка сотрудников при открытии страницы.

Для этого нужно сделать функцию runUI, в которой будут помещены все действия, выполняемые при загрузке страницы.

Данную функцию можно вызывать после подключения скриптов в HTML:

<**script src="employees-json.js"**></**script**>  
<**script src="service.js"**></**script**>  
<**script src="ui.js"**></**script**>

В конце документа добавьте вызов runUI:

<**script**>  
 *runUI*();  
</**script**>

Сама функция пока что выглядит тривиально:

**function** *runUI*() {  
 *showEmployees*(***DATA***.**employees**);  
}

### 3) Создать форму с возможностью добавить нового сотрудника и полями name и surname.

Сама форма должна располагаться в index.html:

<**div**>  
 Имя:  
 <**input id="name" placeholder="Имя"**>  
  
 Фамилия:  
 <**input id="surname" placeholder="Фамилия"**>  
  
 <**button onclick="***addEmployeeUI*()**"**>Добавить сотрудника</**button**>  
</**div**>

При нажатии на кнопку должна вызываться функция *addEmployeeUI():*

**function** *addEmployeeUI*() {  
 **const** name = ***document***.getElementById(**"name"**).**value**;  
 **const** surname = ***document***.getElementById(**"surname"**).**value**;  
 **const** id = *addEmployee*(name, surname);  
 *showEmployees*(***DATA***.**employees**);  
}

### 4) Нужно предусмотреть валидацию данных формы. Если name или surname не заполнены, должно показываться сообщение об ошибке.

Для этого после формы следует предусмотреть соответствующий плейсхолдер:

<**div id="addEmployeeFormErrorMessage"**></**div**>

А в самой функции нужно его заполнять в случае наличия ошибок:

**function** *addEmployeeUI*() {  
 **let** errorHTML = **""**;

**const** name = ***document***.getElementById(**"name"**).**value**;  
 **if** (name==**""**) {  
 **errorHTML** += **"- Имя сотрудника должно быть задано<br>"**;  
 }  
 **const** surname = ***document***.getElementById(**"surname"**).**value**;  
 **if** (surname==**""**) {  
 **errorHTML** += **"- Фамилия сотрудника должна быть задана<br>"**;  
 }  
 ***document***.getElementById(**"addEmployeeFormErrorMessage"**)  
 .innerHTML = errorHTML;  
 **if** (errorHTML.length != 0) **return**;

*addEmployee*(name, surname);  
 *showEmployees*(***DATA***.**employees**);  
}

### 5) После добавления сотрудника данные в форме следует очищать.

Для этого в функции *addEmployeeUI()* нужно очищать поля формы после отправки:

***document***.getElementById(**"name"**).**value = "";**

***document***.getElementById(**"surname"**).**value = "";**

**Дополнительные задания:**

1) Реализовать возможность задать дату рождения сотрудника

2) Реализовать валидацию поля ввода даты рождения

## Занятие 2. Расширение функциональности. Работа с событиями.

### 1) Реализуем возможность удаления сотрудника из списка

В цикле функции showEmployees() добавьте код для добавления кнопки Удалить для каждого сотрудника в списке:

Также необходимо реализовать метод removeEmployeeUI. Эта функция должна вызывать removeEmployee(id) из service.js. После удаления сотрудника необходимо перерисовать список всех сотрудников вызовом showEmployees().

### 2) Создаем возможность добавлять руководителя.

Для сотрудника можно будет задавать руководителя. Это будет делаться путем выбора из выпадающего списка всех сотрудников.

Для этого в форму добавления сотрудника добавим выпадающий список:

Руководитель:  
<**select id="managerSelect"**></**select**>

Этот список надо заполнить списков всех сотрудников. Для этого реализуйте стандартную функцию **fillSelect(select, values, selectedValue)**.

Данная функция будет принимать ссылку на select в первом параметре, массив объектов с текстом и значением во втором, и выбранное значение - в третьем.

Например, следующий код:

fillSelect(document.getElementById("managerSelect"), [

{ text: " Иван Иванов", value: 133 },

{ text: " Петр Петров", value: 134 }], 133);

Заполнит выпадающий список двумя значениями: Иван Иванов и Петр Петров, при этом Иван Иванов будет выбран изначально.

Теперь нам надо создать массив values для заполнения дроп-дауна списком всех сотрудников. Это можно сделать так:

**function** *getEmployeesOptions*() {  
**}**

Остается заполнить дроп-даун списком сотрудников. Нужно для этого изменить runUI()

Теперь необходимо при добавлении сотрудника сохранять ссылку на менеджера. Ссылка будет называться managerRef, и будет содержать id сотрудника, являющегося руководителем. Для этого в service.js реализуем функцию setEmployeeManager(id, managerId). Также в функции addEmployeeUI() будем получать и сохранять id менеджера

### 3) Добавляем возможность изменить руководителя в списке.

Реализуем возможность показывать руководителя. Для этого в функции showEmployees() в цикле по сотрудникам добавим возможность отображать не просто имя руководителя, а выпадающий список с выбранным руководителем.

Также добавим обработчик события onchange. При изменении руководителя ссылка managerRef должна обновляться.

**Дополнительные задания**

1) Реализовать возможность изменить имя, фамилию и дату рождения сотрудника