МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И

МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

СВЯЗИ И ИНФОРМАТИКИ»

Кафедра «Сети связи и системы коммутации»

Лабораторная работа №7

по дисциплине «Web-программирование»

Выполнил студент

группы БФИ1901

Кочеринский Н.В.

Проверил:

Москва, 2021

**Оглавление**

[**1 Задание на лабораторную работу** 3](#_Toc89792531)

[**2 Ход лабораторной работы.** 4](#_Toc89792532)

[2.1 Изучение основ JavaScript, создание простейших функций и использование базовых операторов. 4](#_Toc89792533)

[2.2 Работа с элементами DOM с помощью JavaScript 8](#_Toc89792534)

[**Список используемых источников** 14](#_Toc89792535)

# **1 Задание на лабораторную работу**

1. Напишите функцию, которая фильтрует студентов по группе. Наименование группы, по которой будет проводиться фильтрация, вводится пользователем с клавиатуры.;
2. Вам необходимо написать функцию фильтрации студентов по средней оценке, так, чтобы на экран выводился список студентов, средний балл которых выше заданного. Средний балл, по которому будет проводиться фильтрация, вводится пользователем с клавиатуры.
3. Сделайте другую реализацию функции-обработчика, которая была бы более компактна. Предложенный вариант довольно громоздок, однако он хорошо иллюстрирует некоторые возможности манипулирования DOM с помощью JavaScript. Весь описанный выше функционал можно реализовать с помощью css-классов и их изменения через функцию-обработчик. Сделайте так, чтобы класс folded добавлялся не кнопке, по которой кликнули, а родителю всего поста (это тот самый элемент, у которого в html-разметке установлен класс one-post). В таком случае свойство display можно изменять через css-стили:

.one-post.folded .article-author{

/\* данный стиль применится только для элементов класса

.article-author, у которых родитель с классом one-post имеет также класс folded \*/

display: none;

}

1. По аналогии с разработанной вами функцией, установите через CSS-стили «исчезновение» остальных элементов поста.
2. Загрузите ваш проект на любой гит-репозиторий (GitHub, GitLab, Google Code, Bitbucket и т.п.).

# **2 Ход лабораторной работы.**

## 2.1 Изучение основ JavaScript, создание простейших функций и использование базовых операторов.

Создадим новую папку js в директории с проектом, затем поместим в неё файл helloworld.js. Затем подключим данный скрипт к файлу главной страницы со статьями таким образом:

<script src="{{ STATIC\_URL }}/static/js/helloworld.js"> </script>

Запустим сервер и проверим, подключен ли скрипт, это представлено на рисунке 1.

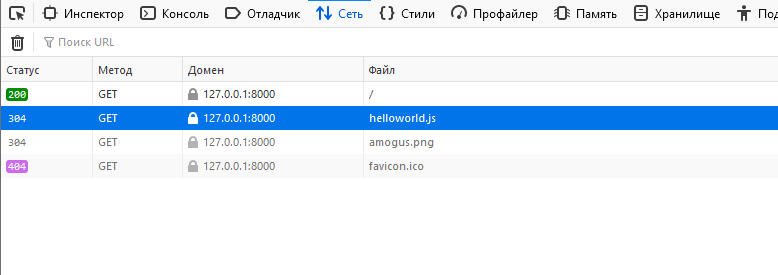


Рисунок 1 – Проверка на подключение скрипта.

Затем, поместим в файл helloworld.js программный код, который представлен на листинге 1.

Листинг 1 - файл helloworld.js

var ***groupmates*** = [  
 {  
 "name": "Максим",  
 "surname": "Агарков",  
 "group": "БФИ1901",  
 "marks": [4, 3, 5]  
 },  
   
 {  
  
 "name": "Смирнов",  
 "surname": "Феликс",  
 "group": "БФИ1901",  
 "marks": [4, 4, 4]  
 },  
  
 {  
 "name": "Шацкий",  
 "surname": "Егор",  
 "group": "БФИ1901",  
 "marks": [5, 5, 5]  
 }  
];

С помощью команды console.log() выведем в браузер массив из объектов. Это представлено на рисунке 2.

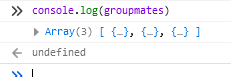


Рисунок 2 - Вывод в консоли.

Далее создадим функцию, которая будет выводить данные о студентах. Листинг файла helloworld.js с добавлением новой функции представлен на листинге 2. Результат работы функции представлен на рисунке 3.

Листинге 2 - файл helloworld.js

var ***groupmates*** = [  
 {  
 "name": "Максим",  
 "surname": "Агарков",  
 "group": "БФИ1901",  
 "marks": [4, 3, 5]  
 },  
  
 {  
  
 "name": "Смирнов",  
 "surname": "Феликс",  
 "group": "БФИ1901",  
 "marks": [4, 4, 4]  
 },  
  
 {  
 "name": "Шацкий",  
 "surname": "Егор",  
 "group": "БФИ1901",  
 "marks": [5, 5, 5]  
 }  
];  
  
 var rpad = function (str, length) {  
 str = str.toString(); // преобразование в строку  
 while (str.length < length) str = str + " ";  
 return str; // добавление пробела в конец строки return str; // когда все пробелы добавлены, возвратить строку  
 };  
  
 var printStudents = function (students) {  
 ***console***.log(  
 rpad("Имя", 15),  
 rpad("Фамилия", 15),  
 rpad("Группа", 8),  
 "Оценки"  
 );  
  
 for (var i = 0; i <= students.length - 1; i++) {  
 ***console***.log(  
 rpad(students[i]["name"], 15),  
 rpad(students[i]["surname"], 15),  
 rpad(students[i]["group"], 8),  
 students[i]["marks"].join(",")  
 );  
 }  
 ***console***.log("\n"); // добавляется пустая строка в конце вывода  
 };

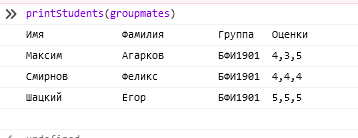


Рисунок 3 - Результат работы.

Далее напишем две функции, одна из которых будет фильтровать студентов по группе, вторая будет выводить студентов, средний балл которых выше заданного. Это представлено на рисунке

Листинг 3 - файл helloworld.js

var ***groupmates*** = [  
 {  
 "name": "Максим",  
 "surname": "Агарков",  
 "group": "БФИ1901",  
 "marks": [4, 3, 5]  
 },  
  
 {  
  
 "name": "Смирнов",  
 "surname": "Феликс",  
 "group": "БФИ1901",  
 "marks": [4, 4, 4]  
 },  
  
 {  
 "name": "Шацкий",  
 "surname": "Егор",  
 "group": "БФИ1901",  
 "marks": [5, 5, 5]  
 }  
];  
  
 var rpad = function (str, length) {  
 str = str.toString();  
 while (str.length < length) str = str + " ";  
 return str;  
 };  
  
 var printStudents = function (students) {  
 ***console***.log(  
 rpad("Имя", 15),  
 rpad("Фамилия", 15),  
 rpad("Группа", 8),  
 "Оценки"  
 );  
  
 for (var i = 0; i <= students.length - 1; i++) {  
 ***console***.log(  
 rpad(students[i]["name"], 15),  
 rpad(students[i]["surname"], 15),  
 rpad(students[i]["group"], 8),  
 students[i]["marks"].join(",")  
 );  
 }  
 ***console***.log("\n");  
 };  
 let groupFilter = () => {  
 let group = prompt("Введите группу");  
 let newGroups = ***groupmates***.filter((item) => item["group"] == group);  
 if (newGroups.length == 0) {  
 ***console***.log("Студентов из такой группы нет");  
 } else {  
 printStudents(newGroups);  
 }  
 };  
  
 let marksFilter = () => {  
 let marks = +prompt("Введите среднюю оценку");  
 let newMarks = ***groupmates***.filter(  
 (item) =>  
 item["marks"].reduce((prev, next) => prev + next) /  
 item["marks"].length >=  
 marks  
 );  
  
 if (newMarks.length == 0) {  
 ***console***.log("Студентов с таким средним баллом нет");  
 } else {  
 printStudents(newMarks);  
 }  
 };

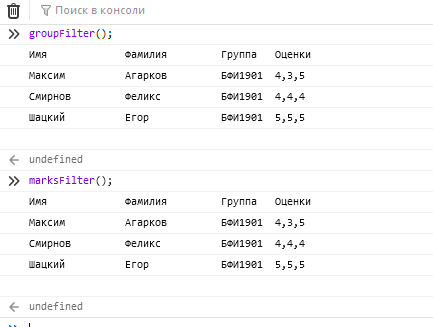


Рисунок 4 – Результат фильтрации.

## 2.2 Работа с элементами DOM с помощью JavaScript

Создадим файл fold-post.js, программный код которого представлен на листинге 4, а затем подключим данный скрипт к html-документу. Результат работы представлен на рисунке 5.

Листинг 4 – файл fold-post.js

var ***foldBtns*** = ***document***.getElementsByClassName("fold-button");  
for (var ***i*** = 0; ***i***<***foldBtns***.length; ***i***++) {  
 ***foldBtns***[***i***].addEventListener("click", function (event) {  
 ***console***.log("you clicked ", event.target);  
 });  
}

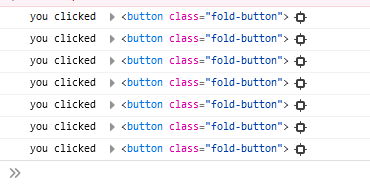


Рисунок 5 – Результат.

Далее необходимо реализовать функции скрывания информации статей. Программный код представлен на листинге 5. Результат представлен на рисунке 6.

Листинг 5 - файл fold-post.js

***window***.addEventListener("DOMContentLoaded", () => {  
 const foldBtns = ***document***.querySelectorAll(".fold-button");  
 for (let i = 0; i < foldBtns.length; i++){  
 foldBtns[i].addEventListener("click", (e) => {  
 e.target.parentElement.getElementsByClassName("article-author")[0].style.display = "none";  
 e.target.parentElement.getElementsByClassName("article-created-date")[0].style.display = "none";  
 e.target.parentElement.getElementsByClassName("article-text")[0].style.display = "none";  
 e.target.innerHTML = "развернуть";  
 });  
 }  
});



Рисунок 6 – Результат

Далее реализуем проверку на наличие у кнопки класса folded, по нажатии на которую будет показываться или скрываться информация о статье. Это представлено на листинге 6.

Листинг 5 - файл fold-post.js

***window***.addEventListener("DOMContentLoaded", () => {  
 const foldBtns = ***document***.querySelectorAll(".fold-button");  
 for (let i = 0; i < foldBtns.length; i++) {  
 foldBtns[i].addEventListener("click", (e) => {  
 if (e.target.className == "fold-button folded") {  
 e.target.innerHTML = "свернуть";  
 e.target.className = "fold-button";  
 var displayState = "block";  
 } else {  
 e.target.innerHTML = "развернуть";  
 e.target.className = "fold-button folded";  
 var displayState = "none";  
 }  
 e.target.parentElement.getElementsByClassName(  
 "article-author"  
 )[0].style.display = displayState;  
 e.target.parentElement.getElementsByClassName(  
 "article-created-date"  
 )[0].style.display = displayState;  
 e.target.parentElement.getElementsByClassName(  
 "article-text"  
 )[0].style.display = displayState;  
 });  
 }  
});

Далее по плану лабораторной работы, необходимо сделать нашу функцию компактнее, для неё необходимо изменить файл со стилями в соответствии с требованиями лабораторной работы. Файлы представлены на листингах 6-8, результат работы представлен на рисунке 7.

Листинг 6 – fold-post.js

***window***.addEventListener("DOMContentLoaded", () => {  
 const wrappers = ***document***.querySelector(".archive");  
  
 wrappers.addEventListener("click", (e) => {  
 target = e.target;  
 if (target && target.matches(".fold-button")) {  
 if (target.parentElement.classList.contains("folded")) {  
 target.innerHTML = "Свернуть";  
 target.parentElement.classList.remove("folded");  
 } else {  
 ***console***.log(target.parentElement);  
 target.innerHTML = "Развернуть";  
 target.parentElement.classList.add("folded");  
 }  
 }  
 });  
});

Листинг 7 – archive.html

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Архив статей</title>  
 <link rel="stylesheet" href="{{ STATIC\_URL }}/static/article.css">  
</head>  
<body>  
<script src="{{ STATIC\_URL }}/static/js/helloworld.js"></script>  
<script src="{{ STATIC\_URL }}/static/js/fold-post.js"></script>  
 <div class="header">  
 <img src="{{ STATIC\_URL }}/static/img/amogus.png"/>  
 <ul class="buttons">  
 <li class="linkButton"><a class="link" href="http://127.0.0.1:8000/article/new/">Создать статью</a></li>  
 <li class="linkButton"><a class="link" href="/registration"> Регистрация</a></li>  
 <li class="linkButton"><a class="link" href="/logIn">Вход</a></li>  
 <li class="linkButton"><a class="link" href="/logOut">Выйти</a></li>  
 </ul>  
 </div>  
 <div class="archive">  
 {% for post in posts %}  
 <div class="one-post">  
 <h2 class="post-title"><a href="http://127.0.0.1:8000/article/{{post.id}}">{{post.title}}</a></h2>  
 <button class="fold-button">Cвернуть</button>  
 <div class="article-info">  
 <div class="article-author">{{ post.author.username }}</div>  
 <div class="article-created-date">{{ post.created\_date }}</div>  
 <p class="article-text">{{ post.get\_excerpt }}</p>  
 </div>  
 </div>  
 {% endfor %}  
 </div>  
<script src="{{ STATIC\_URL }}/static/js/fold-post.js"></script>  
</body>  
</html>

Листинг 8 – article.css

body{  
 background: #1abc9c;  
 font-family: Tahoma, Arial, sans-serif;  
 color: #ffffff  
}  
  
img {  
 display: block;  
 width: 318px;  
 margin-left: auto;  
 margin-right: auto;  
}  
  
.archive {  
 width: 960px;  
 margin-left: auto;  
 margin-right: auto;  
}  
  
.post-title a {  
 color: #ffffff;  
}  
  
.article-author {  
 width: 50%;  
 float: left;  
}  
  
.article-created-date {  
 text-align: right;  
}  
  
.article-image{  
 display: block;  
 width: 318px;  
 margin-left: 0;  
}  
  
  
.article-border p{  
 text-align: right;  
}  
  
.article-text{  
 width: 960px;  
 text-align: justify;  
}  
.article-created-data{  
 text-align: right;  
}  
  
.content{  
 text-align: center;  
 padding-top: 70px;  
}  
  
input[name="title"]{  
 padding: 5px;  
 margin-bottom: 10px;  
 border: 1px solid #888;  
 outline: none;  
 -moz-appearance: none;  
 width: 200px;  
 text-align: center;  
 border-radius: 40px;  
}  
  
textarea[name="text"]{  
 padding: 25px;  
 margin-bottom: 10px;  
 border: 1px solid #888;  
 outline: none;  
 -moz-appearance: none;  
 width: 650px;  
 height: 350px;  
 resize: none;  
 border-radius: 40px;  
 scrollbar-width: thin;  
}  
  
.create{  
 display: flex;  
 flex-direction: column;  
 align-items: center;  
}  
  
.save\_button{  
 padding: 10px;  
 width: 150px;  
 background-color: white;  
 border: none;  
 border-radius: 40px;  
 color: #1abc9c;  
 font-weight: bold;  
 letter-spacing: 0.06em;  
 margin-top: 10px;  
}  
  
.save\_button:hover{  
 color: white;  
 background-color: #1abc9c;  
 box-shadow: 1px 1px 10px 10px;  
 transition-duration: 0.3s;  
}  
  
.linkButton{  
padding-bottom: 10px;  
}  
  
  
.link {  
 color: white;  
 font-weight: bold;  
}  
  
.buttons{  
 list-style-type: none;  
 margin-left: 422px;  
}  
  
.fold-button{  
 border: 1px;  
 color: white;  
 font-size: 18px;  
 background: transparent;  
 font-weight: bold;  
 text-decoration: underline;  
 float: right;  
 cursor: pointer;  
}  
  
.one-post.folded .article-info {  
 display: none;  
}  
  
.one-post.folded .article-text {  
 display: none;  
}

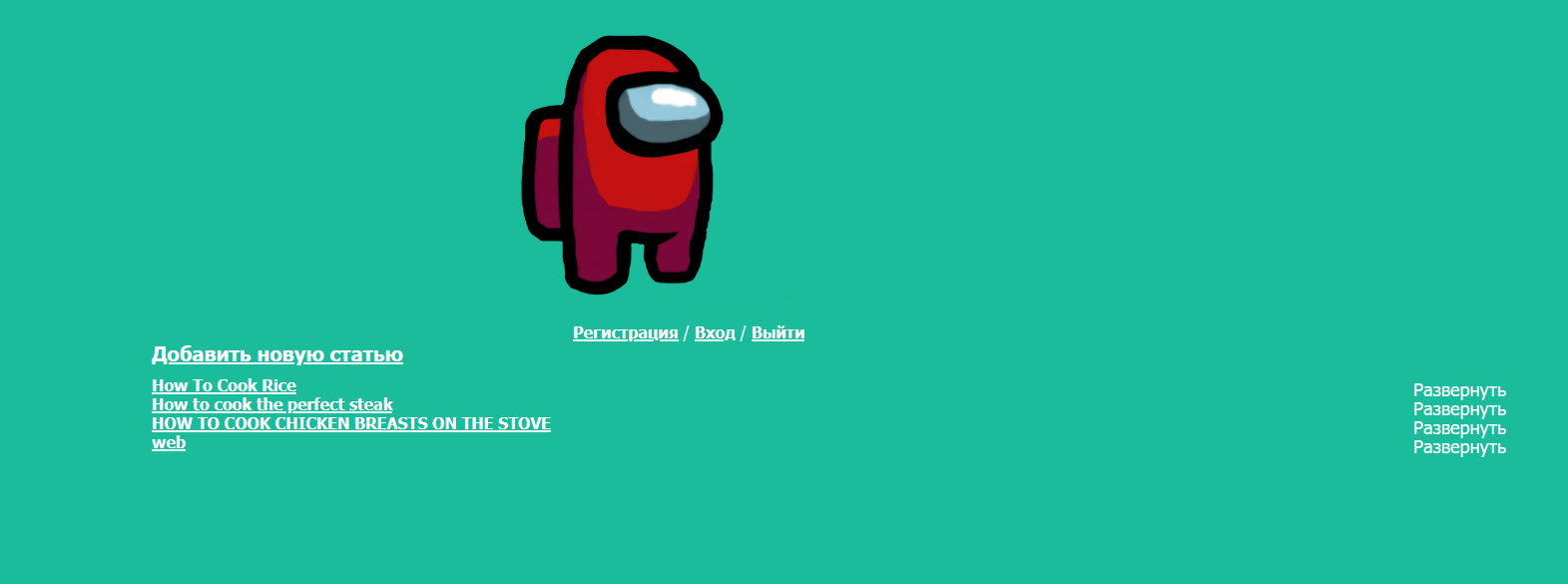


Рисунок 7 - Результат

**Список используемых источников**

1 ГОСТ 7.32-2017 СИБИД. Отчет о научно-исследовательской работе. Структура и правила оформления.

2 ГОСТ 7.1-2003 Библиографическая запись. Библиографическое описание.