МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ассистент |  |  |  | Ю.В. Ветрова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ПРАКТИЧЕСКОМ ЗАДАНИИ №2 |
|  |
| по курсу: JavaScript, его библиотеки и фреймворки в Frontend-разработке |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4136 |  |  |  | Бобрович Н.С. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

1. **Цель работы:**

Формирование практических навыков работы с асинхронным кодом в языке JavaScript, используя callback-функции и объект Promise.

1. **Формулировка заданий:**

1. Напишите функцию `filterArray`, которая принимает массив чисел и callback функцию. Функция `filterArray` должна вызывать callback функцию для каждого элемента массива и возвращать новый массив, содержащий только те элементы, для которых callback функция вернула `true`. Напишите 2 примера применения этой функции. К примеру, для фильтрации чётных и нечётных значений массива.

2. Напишите асинхронную функцию `fetchData`, которая принимает URL в качестве параметра и возвращает Promise. Функция должна использовать `fetch` для получения данных с указанного URL. Если запрос прошёл успешно (статус ответа 200), Promise должен быть разрешён с полученными данными в виде строки. Если запрос не удался (любой другой статус), Promise должен быть отклонён с сообщением об ошибке.

1. **Скриншоты, иллюстрирующие выполнение отдельных пунктов задания:**

Рис. 1: фильтрация массива (нечётные значения) и получение данных.

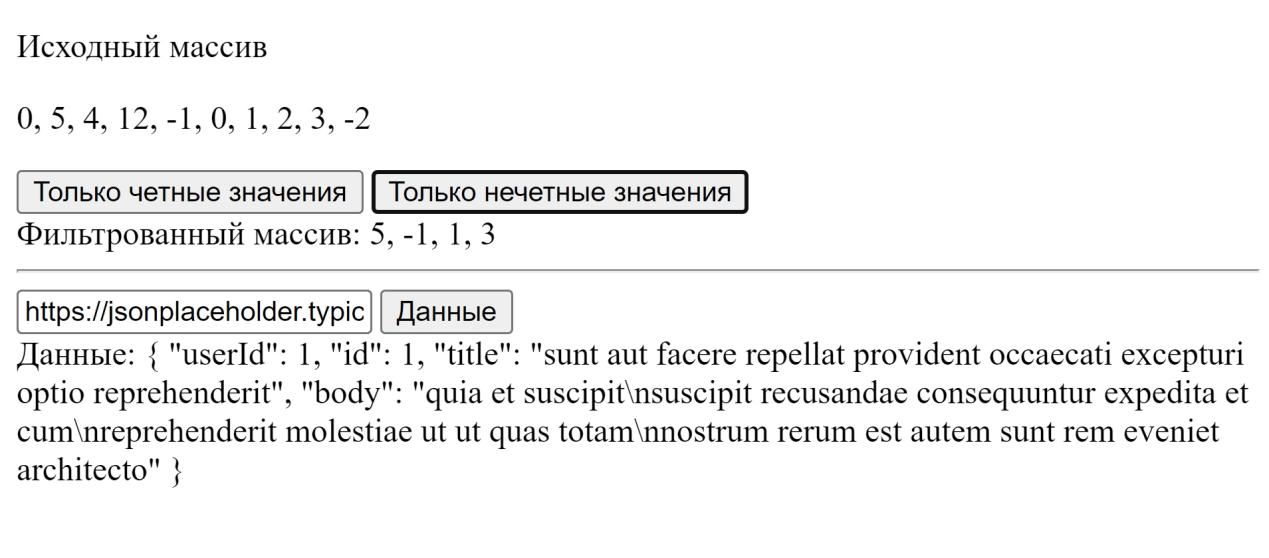
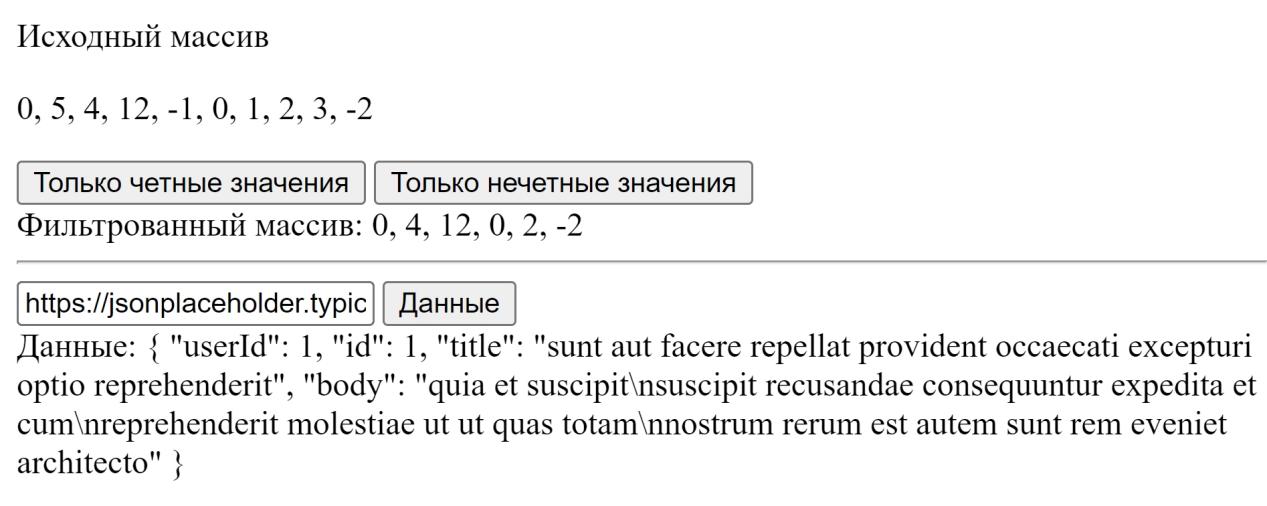


Рис. 2: фильтрация массива (чётные значения) и получение данных.



1. **Листинг:**

Lab2.html:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Lab\_2</title>

</head>

<body>

<div>

<p>Исходный массив</p>

<p id="array"></p>

<button id="evenNumbers">Только четные значения</button>

<button id="notEvenNumbers">Только нечетные значения</button>

<div id="outputArray"></div>

<hr>

</div>

<div>

<input type="text" id="urlExample" placeholder="Введите URL" value="https://jsonplaceholder.typicode.com/posts/1">

<button id="fetchDataButton">Данные</button>

<div id="outputURL"></div>

</div>

<script src="lab2.js"></script>

<script src="app.js"></script>

</body>

</html>

Lab2.js:

const numbers = [0, 5, 4, 12, -1, 0, 1, 2, 3, -2];

document.getElementById('array').textContent = numbers.join(', ');

function filterArray(arr, callback) {

let result = [];

for(let i = 0; i < arr.length; i++) {

if(callback(arr[i])) {

result.push(arr[i]);

}

}

return result;

}

async function fetchData(url) {

const response = await fetch(url);

if (response.ok) {

const data = await response.text();

return data;

} else {

throw new Error(`Ошибка: ${response.status}`);

}

}

document.getElementById('evenNumbers').addEventListener('click', () => {

const isEven = num => num % 2 === 0;

const result = filterArray(numbers, isEven);

document.getElementById('outputArray').textContent = 'Фильтрованный массив: ' + result.join(', ');

});

document.getElementById('notEvenNumbers').addEventListener('click', () => {

const isNotEven = num => num % 2 !== 0;

const result = filterArray(numbers, isNotEven);

document.getElementById('outputArray').textContent = 'Фильтрованный массив: ' + result.join(', ');

});

document.getElementById('fetchDataButton').addEventListener('click', async () => {

const url = document.getElementById('urlExample').value; fetchData(url)

.then(data => {

console.log('Данные:', data);

document.getElementById('outputURL').textContent = 'Данные: ' + data;

})

.catch(error => {

console.error('Ошибка:', error);

document.getElementById('outputURL').textContent = 'Ошибка: ' + error.message;

});

});

1. **Вывод:**

Были сформированы практические навыки работы с асинхронным кодом в языке JavaScript, используя callback-функции и объект Promise.