ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ ПРАКТИЧЕСКОМУ ЗАДАНИЮ №2 |
| по дисциплине: ИТ-модуль "Основы Frontend-разработки" |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4134к |  |  |  | Костяков Н.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2024

##### **Практическое задание № 2 - Асинхронность**

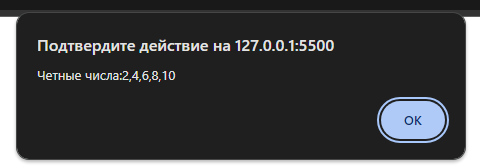
Цель работы: формирование практических навыков работы с асинхронным кодом в языке JavaScript, используя callback-функции и объект Promise.

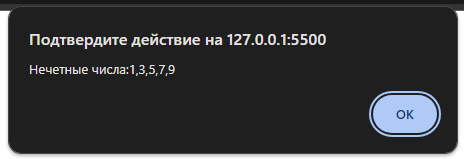
##### **Задание**

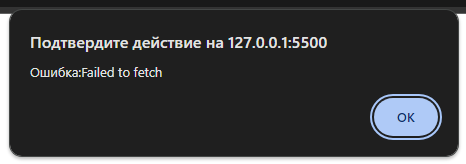
1.  Напишите функцию `filterArray`, которая принимает массив чисел и callback функцию. Функция `filterArray` должна вызывать callback функцию для каждого элемента массива и возвращать новый массив, содержащий только те элементы, для которых callback функция вернула `true`. Напишите 2 примера применения этой функции. К примеру, для фильтрации четных и нечетных 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 прошел успешно (статус ответа 200), Promise должен быть разрешен с полученными данными в виде строки. Если запрос не удался (любой другой статус), Promise должен быть отклонен с сообщением об ошибке.

# Скриншоты







# Листинг

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

<script>

        function filterArray(array, callback) {

    var filteredArray = [];

    for (var i = 0; i < array.length; i++) {

        if (callback(array[i])) {

            filteredArray.push(array[i]);

        }

    }

    return filteredArray;

}

    // Пример фильтрации четных чисел

    var numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

    function isEven(number) {

        return number % 2 === 0;

    }

    var evenNumbers = filterArray(numbers, isEven);

    alert("Четные числа:"+ evenNumbers);

    // Пример фильтрации нечетных чисел

    function isOdd(number) {

        return number % 2 !== 0;

    }

    var oddNumbers = filterArray(numbers, isOdd);

    alert("Нечетные числа:"+ oddNumbers);

    </script>

</body>

</html>

Fetch.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

            async function fetchData(url) {

        try {

            const response = await fetch(url);

            if (!response.ok) {

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

            }

            const data = await response.text();

            return data;

        } catch (error) {

            return Promise.reject(error.message);

        }

    }

    fetchData('https://lms.guap.ru/new/mod/assign/view.php?id=139187')

    .then(data => {

        alert('Данные:' + data);

    })

    .catch(error => {

        alert('Ошибка:' + error);

    });

    </script>

</body>

</html>

# Вывод

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