**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по домашнему заданию

«Реферат на тему “Асинхронное программирование в JavaScript”»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-31Б |  | преподаватель каф. ИУ5 |
| Баринов А. А. |  | Гапанюк Ю. Е. |
| Подпись и дата: |  | Подпись и дата: |
|  |  |  |

Москва, 2023 г.

Реферат на тему “Асинхронное программирование в JavaScript”

Асинхронное программирование становится все более важным аспектом в современном веб-разработке. Однако, прежде чем погружаться в детали асинхронности в JavaScript, давайте рассмотрим основные понятия и принципы этой парадигмы.

Основные понятия асинхронного программирования в JavaScript:

1. Callback функции:

Callback функции – это функции, передаваемые в другие функции в качестве аргументов, и вызываемые после завершения какой-то операции. Они играют ключевую роль в асинхронном коде.

Пример:

function fetchData(callback) {

setTimeout(() => {

console.log('Data fetched successfully!');

callback();

}, 2000);

}

function processData() {

console.log('Data processed.');

}

fetchData(processData);

2. Promises:

Promises предоставляют более удобный способ работы с асинхронным кодом. Они представляют собой объекты, представляющие успешное завершение или ошибку асинхронной операции.

Пример:

function fetchData() {

return new Promise((resolve, reject) => {

setTimeout(() => {

console.log('Data fetched successfully!');

resolve();

}, 2000);

});

}

function processData() {

console.log('Data processed.');

}

fetchData().then(processData);

3. Async/Await:

Async/Await – это синтаксический сахар над промисами, который упрощает написание асинхронного кода, делая его более похожим на синхронный.

Пример:

async function fetchData() {

return new Promise((resolve) => {

setTimeout(() => {

console.log('Data fetched successfully!');

resolve();

}, 2000);

});

}

async function processData() {

console.log('Data processed.');

}

async function main() {

await fetchData();

processData();

}

main();

Вывод:

Асинхронное программирование в JavaScript становится все более важным с ростом сложности веб-приложений. Callback функции, Promises и Async/Await предоставляют разработчикам эффективные инструменты для работы с асинхронным кодом, делая его более читаемым и управляемым. Ознакомление с этими концепциями позволяет создавать более отзывчивые и эффективные веб-приложения.