**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

по дисциплине «Разработка приложений для Интернет»

на тему: «EventLoop»

Выполнил: студент гр. ИП-31

Коваленко А.И.

Принял: преподаватель

Самовендюк Н. В.

Гомель 2023

**Цель:** изучить основные понятия EventLoop, возможность по работе с событиями в NodeJs

|  |  |  |
| --- | --- | --- |
| 15 | Фитнес-браслет | Объем оперативной памяти  Поддерживаемые платформы  Датчики |

**Задание 1** Написать скрипт, который наглядно будет демонстрировать работу каждой из фаз EventLoop:

* Создать три таймера SetTimeOut с разным временем срабатывания. При срабатывании таймера выводить время, прошедшее с момента запуска таймера.
* Применить операции ввода-вывода для записи и чтения файла. После окончания операции ввода-вывода выводить время, прошедшее с момента начала операции.
* Добавить вызов SetImmediate. Вывести время, прошедшее с момента вызова.

**Задание 2** Зарегистрировать в системе пользовательское событие. Сгенерировать и обработать пользовательское событие с выводом информации о событии.

**Задание 3** Промоделировать ситуацию, в которой Вы являетесь продавцом товара (описание товара в лабораторной №2). На обновление списка товара подписать нескольких клиентов, которым будет приходить сообщение при добавлении нового товара или модификации старого. Клиенты должны логгировать информацию от продавца в своих журналах в виде файла с занесением времени.

**Листинг программы:**

**task1.js**

const fs = require('fs');

function executeTime( startTime){

const stop = (new Date()).getTime();

console.log(` время ${stop-startTime} мс`)

}

function sayImmediate() {

}

function Immediate() {

let startTime = performance.now()

setImmediate(sayImmediate)

let endTime = performance.now()

console.log("Время setImmidiate = " + (endTime - startTime));

}

let temptime = 1000

const startFirst = (new Date()).getTime();

setTimeout(executeTime, temptime, startFirst);

const startSecond = (new Date()).getTime();

setTimeout(executeTime, temptime \* 3, startSecond);

const startThird = (new Date()).getTime();

setTimeout(executeTime, temptime \* 5, startThird);

let startTimer = performance.now()

fs.readFile("task1out.txt", "utf-8", function (err, data) {

if (err)

throw err;

else

{

let endTime = performance.now()

console.log("Время чтения = " + (endTime - startTimer));

console.log(data)

}

});

let startTimew = performance.now()

fs.writeFile("task1in.txt", "TEXT TEST TEXT", (err) => {

if (err)

console.log(err);

else

{

let endTime = performance.now()

console.log("Время записи = " + (endTime - startTimew));

}

});

Immediate()

**task2.js**

const Emitter = require("events");

let emitter = new Emitter();

let eventName = "say";

emitter.on(eventName, function(){

console.log("Привет!");

});

emitter.emit(eventName);

**task3.js**

import { EventEmitter } from 'node:events';

import fs from 'fs';

class FitnessTracker{

constructor(ram, support, sensor){

this.ram = ram;

this.support = support;

this.sensor = sensor;

}

toString()

{

return `ram: ${this.ram}, support: ${this.support}, sensor ${this.sensor}`

}

}

class Client{

constructor(name, journal){

this.name = name

this.journal = journal;

this.log = this.log.bind(this);

}

log(text) {

fs.appendFile(this.journal , text,(err)=>{});

}

}

class Store{

constructor(){

this.goods = []

this.emitter = new EventEmitter()

this.clients = []

this.emitter.on("add", (data)=>{

this.clients.forEach(element => {

element.log(data)

});

})

}

add(goods)

{

this.goods.push(goods)

this.emitter.emit("add",`\n Добавлен товар: ${this.goods[this.goods.length - 1]} \n в: ${new Date()}`)

}

change(goods)

{

this.goods=goods

this.emitter.emit("add",`\n Товары изменены \n в: ${new Date()}`)

}

addListener(client)

{

this.clients.push(client)

}

}

let c1 = new Client("первый",'first.txt')

let c2 = new Client("второй",'second.txt')

let c3 = new Client("третий",'third.txt')

let st = new Store();

st.addListener(c1)

st.addListener(c2)

st.addListener(c3)

st.add(new FitnessTracker(1024, "andriod", "accelerometer"))

st.add(new FitnessTracker(512, "ios", "ios"))

st.add(new FitnessTracker(256, "ios", "magnetometer"))

let goods = [

new FitnessTracker(200, "mi", "magnetometer"),

new FitnessTracker(400, "mi", "gyroscope")

]

st.change(goods)

**Результаты выполнения программы:**

Результат выполнения задания 1 представлен на рисунке 1.

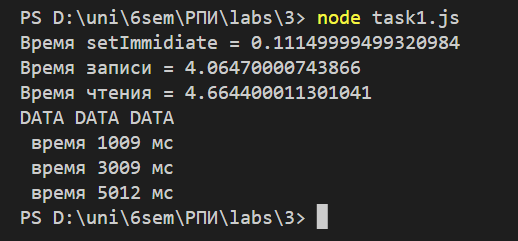
****

Рисунок 1 – Результат выполнения задания 1

Результат выполнения задания 2 представлен на рисунке 2.

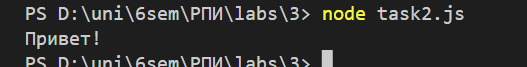
****

Рисунок 2 – Результат выполнения задания 2

Результат выполнения задания 3 представлен на рисунке 3.

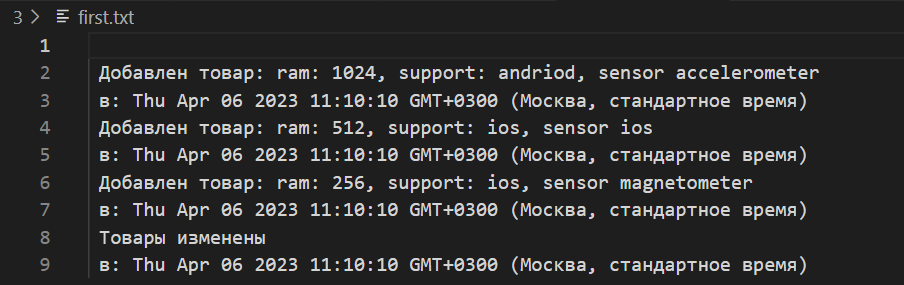


Рисунок 3 – Результат выполнения задания 3

Результат разделения уведомлений для каждого пользователя представлен на рисунке 4.

****

****

****

Рисунок 4 – Результат выполнения задания 3

**Вывод:** в ходе выполнения лабораторной работы были изучены основные понятия EventLoop, возможность по работе с событиями в NodeJs.