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 № 2

по дисциплине «РПИ»

на тему: «Работа с файловой системой и потоками в Node.js»

Выполнил: студент гр. ИП-32

Суховенко Э.С.

Принял: преподаватель

Процкая М.А.

Гомель 2022

**Цель работы:** изучить основные возможности по работе с файловой системой и потоками в Node.js

**Задание**

Задание 1. Написать приложение, которое логирует ошибки (вводятся пользователем) в текстовом файле. В файл добавляется дата ошибки и ее описание.

Задание 2. Создать массив объектов на основе класса в соответствии со своим вариантом. Записать этот массив в формате JSON в файл.

|  |  |  |
| --- | --- | --- |
| 12 | Планшет | Диагональ экрана  Процессор  Разрешение экрана  Операционная система  Оперативная память  Внутренняя память |

Задание 3. Произвести считывание данных из созданного файла в новый массив. Вывести массив в консоль.

Задание 4. Создать в папке проекта две папки src и dist. В папку src скопировать исходные файлы приложений (\*.js). В папку dist скопировать созданные файлы.

**Код программы**

**//** index

import { ERRORS\_LOG\_FILEPATH, INITIAL\_TABLETS, RES\_TABLETS\_FILEPATH } from './constants'  
import { *logs* } from './tasks/logs'  
import { *readTablets* } from './tasks/readTablets'  
import { *writeTables* } from './tasks/writeTablets'  
import { *inputString* } from './inputUtils'  
  
const main = async () => {  
 const error = await *inputString*('Input error')  
 const description = await *inputString*('Input description')  
 const date = new Date().toDateString()  
 *logs*(ERRORS\_LOG\_FILEPATH, {  
 error,  
 description,  
 date,  
 })  
 *writeTables*(RES\_TABLETS\_FILEPATH, INITIAL\_TABLETS)  
 const tables = *readTablets*(RES\_TABLETS\_FILEPATH)  
 console.log(tables)  
}  
  
main()

// constants

import { CPU, Diagonal, LocalMemory, OperatingSystem, RandomAccessMemory, Resolution, Tablet } from './types'  
  
export const ERRORS\_LOG\_DIRPATH = './out'  
export const ERRORS\_LOG\_FILEPATH = './out/errors.log'  
export const RES\_TABLETS\_FILEPATH = './res/tablets.json'  
  
export const INITIAL\_TABLETS: Tablet[] = [  
 {  
 diagonal: Diagonal.*\_1600x900*,  
 CPU: CPU.*mediatek1200*,  
 resolution: Resolution.*seven*,  
 operatingSystem: OperatingSystem.*android*,  
 randomAccessMemory: RandomAccessMemory.*large*,  
 localMemory: LocalMemory.*large*,  
 },  
 {  
 diagonal: Diagonal.*\_1280x720*,  
 CPU: CPU.*snapdragon600*,  
 resolution: Resolution.*five*,  
 operatingSystem: OperatingSystem.*ios*,  
 randomAccessMemory: RandomAccessMemory.*small*,  
 localMemory: LocalMemory.*small*,  
 },  
]

// tasks/logs

import fs from 'fs'  
import { ERRORS\_LOG\_DIRPATH } from '../constants'  
  
export const *logs* = (filePath: string, options: object) => {  
 if (!fs.*existsSync*(filePath)) {  
 fs.*mkdirSync*(ERRORS\_LOG\_DIRPATH)  
 }  
 fs.*writeFileSync*(filePath, JSON.stringify(options, null, ' '))  
}

// tasks/readTablets

import fs from 'fs'  
import { Tablet } from '../types'  
  
export const *readTablets* = (filePath: string): Tablet[] => {  
 const fileData: string = fs.*readFileSync*(filePath).toString()  
 return JSON.parse(fileData)  
}

// tasks/writeTablets

import { Tablet } from '../types'  
import fs from 'fs'  
  
export const *writeTables* = (filePath: string, tablets: Tablet[]) => {  
 fs.*writeFileSync*(filePath, JSON.stringify(tablets, null, ' '))  
}

// inputUtils

import { prompts } from 'prompts'  
  
const argsToPrompt: {  
 type: 'text' | 'number'  
 name: string  
} = {  
 type: 'text',  
 name: 'value',  
}  
  
const createFullArgsToPrompt = (message: string) => ({  
 ...argsToPrompt,  
 message,  
})  
  
export const *inputString* = async (message: string) => {  
 return prompts.*text*(createFullArgsToPrompt(message))  
}

// types

export enum Diagonal {  
 *\_2560x1440* = '2560x1440',  
 *\_1920x1080* = '1920x1080',  
 *\_1280x720* = '1280x720',  
 *\_1600x900* = '1600x900',  
}  
  
export enum CPU {  
 *a11* = 'a11',  
 *snapdragon600* = 'snapdragon600',  
 *mediatek1200* = 'mediatek1200',  
}  
  
export enum Resolution {  
 *five* = '5.0',  
 *six* = '6.0',  
 *seven* = '7.0',  
}  
  
export enum OperatingSystem {  
 *android* = 'android',  
 *ios* = 'ios',  
 *windows* = 'windows',  
}  
  
export enum RandomAccessMemory {  
 *small* = '1',  
 *large* = '4',  
 *superLarge* = '8',  
}  
  
export enum LocalMemory {  
 *small* = '16',  
 *large* = '32',  
 *superLarge* = '64',  
}  
  
export type Tablet = {  
 diagonal: Diagonal  
 CPU: CPU  
 resolution: Resolution  
 operatingSystem: OperatingSystem,  
 randomAccessMemory: RandomAccessMemory,  
 localMemory: LocalMemory,  
}

// res/tablets.json

[  
 {  
 "diagonal": "1600x900",  
 "CPU": "mediatek1200",  
 "resolution": "7.0",  
 "operatingSystem": "android",  
 "randomAccessMemory": "4",  
 "localMemory": "32"  
 },  
 {  
 "diagonal": "1280x720",  
 "CPU": "snapdragon600",  
 "resolution": "5.0",  
 "operatingSystem": "ios",  
 "randomAccessMemory": "1",  
 "localMemory": "16"  
 }  
]

**Результат**

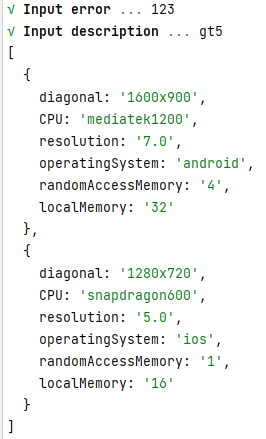


Рисунок 1

**Вывод**: изучил основные возможности по работе с файловой системой и потоками в Node.js