**Содержание**

|  |
| --- |
| Задание…………………………………….…………………..……….... 4 |
| Введение…………………………………………..………………...…… 5 |
| 1 Проектирование приложения.…………......………...…….….……… 6 |
| * 1. Проектирование файлового менеджера……………….………... 6 |
| 2 Разработка приложения…………….………….….…………………. 10 |
| 2.1 Разработка графического интерфейса………………………… 10 |
| 2.2 Разработка программного обеспечения……………………...... 11 |
| 2.3 Обеспечение минимального функционала……………………. 13 |
| 2.4 Добавление основного функционала…………………............... 15 |
| 2.5 Выбор стратегии тестирования и разработки тестов………….. 19 |
| 3 Руководство пользователя………………..…………………………… 22 |
| Заключение……..……………………………………………..……….…. 31 |
| Список использованных источников………………..……..….………... 32  Приложение А……………………………………………………………. 35 |

# 

# **Задание**

1. Необходимо разработать программное обеспечение для работы с файлами и устройствами ввода/вывода, а также для мониторинга ресурсов. Задание курсовой работы состоит из следующих основных пунктов:

Проектирование программного обеспечения.

Разработка графического интерфейса.

Разработка программного обеспечения.

Обеспечение минимального функционала.

Добавление основного функционала.

Выбор стратегии тестирования и разработка тестов.

Формирование отчета по курсовой работе.

1. Способ межпроцессного взаимодействия «Файлового менеджера» и всплывающего окна: Отображение файлов – для последней цифры [1 и 2] зачетной книжки, Сокеты — [ 3 и 4], Каналы — [5 и 6], Семафор — [7 и 8] и Разделяемая память — [9 и 0].
2. Отслеживание всех процессов ОС для 1-5 номеров зачетных книжек и процессы внутри «Файлового менеджера» 6-0 номера.
3. Основной функционал для «Файлового менеджера»:
   1. Имя процесса.
   2. общая загрузка процессора в %.
   3. Маска привязки процесса к процессорам.

# 

# **Введение**

**Тема курсового проекта** Разработка программного комплекса “Файловый менеджер”**.**

**Актуальность темы:** связи c формированием ИТ-технологий, а непосредственно технологий построения Файлового менеджера возникла значимость в разработке программного обеспечения «Файловый менеджер».

**Цель проектирования:** разработка Файлового менеджера.

**Предметом исследования** считается формирование и проектирование Файлового менеджера также, кроме того, создание вспомогательного перечня возможностей, содержащего: Заключение имени пк, кроме того с целью прикладного процесса, подобранного во перечне выполняющихся действий, представить перечень его потоков со указанием свойств каждого потока.

**Объект проектирования:** Файловая система Windows.

**Гипотеза исследования:** если в ходе формирования и проектирования Файлового менеджера использовать новейшие способы исследования рассмотрения, то возможно существенно сократить время, с целью разработки Файлового менеджера.

**Теоретическая значимость:** теоретическая часть помогает познакомиться с Файловой системой и углубиться в смысл темы.

**Практическая значимость** заключается во этом, для того чтобы с помощью расчетов также грамотного проектирования алгоритмов работы системы, создать программное обеспечение «Файловый менеджер».

**Методы исследования:** подбор, анализ, расчет, сопоставление и построение Файлового менеджера.

**Структура проекта** соответствует требованиям и включает в себя следующие этапы: введение, проектирования, построения, вывод, заключение, список источников и литературы.

**1. Проектирование приложения**

**1.1 Проектирование файлового менеджера**

В этом месте разработки проекта нам следует сделать свой выбор с подбором средств разработки также среды разработки, сделать сравнительную таблицу согласно по ним, но кроме того ввести в отчет также изложить подключаемые внешние библиотеки. В курсовой работе были рассмотрены три наиболее популярных сред разработки:

Microsoft Visual Studio — классификация программных продуктов корпорации Microsoft, включающая интегрированную среду разработки программного обеспечения (IDE) и ряд других инструментов.

Project Rider — интегрированная кроссплатформенная среда разработки программных продуктов для платформы .NET, разработанная компанией JetBrains. Поддерживаются языки программирования C#, VB.NET и F#.

Geany — среда разработки программного обеспечения, написанная с использованием библиотеки GTK+. (табл. 1)

Таблица 1

Достоинства и недостатки сред разработки

|  | Достоинства | Недостатки |
| --- | --- | --- |
| Project Rider | Кроссплатформенность. Project Rider работает с Windows, Linux и MacOS. | Стоимость обслуживания, одна версия стоит 139 $. |
| Multiple runtime. Поддержка нескольких запущенных программ. | Программа относительно новая, часть функций находится в разработке. |

|  |  |  |
| --- | --- | --- |
| Microsoft Visual Studio | Функциональность ПО. Visual Studio имеет множество качественных плагинов. С их помощью вы сможете расширить функционал приложения и подключить другие языки. | Баги при переходах с триал-версии. |
| Бесплатное ПО. Версии VS «Community» для обычного пользователя будет достаточно. Можно подключать плагины (в отличие от старой версии Express). | Сложности при освоении данной средой разработки. |
| Язык программирования С#, и среда разработки VS созданы в корпорации Microsoft. | Интуитивно-непонятный интерфейс |
| Имеется облачное хранилище. | — |
| Инструмент Intellsense, технология автодополнения Microsoft. Помогает дописывать функции или подсказывать разработчику в чем скрывается ошибка. | — |
| Geany | Open-Source решение, значит, оно бесплатно. | Open-Source решение, нет полной уверенности в том, что все баги будут починены в ближайшее время. |

Согласно сведениям приведенной выше таблицы, наилучшим решением нашего выбора в применении среды разработки будет являться продукт фирмы Microsoft - Visual Studio, так как данная система включает довольное число различных функций среды разработки.

Уже после подбора среды разработки опишем применяемые нами подключаемые внешние библиотеки.

**Подключаемые библиотеки**

System.IO — Содержит типы для чтения и записи файлов и потоков данных, а также типы для базовой поддержки файлов и папок.

System.Net.Sockets — Предоставляет управляемую реализацию интерфейса сокетов интерфейса сокетов Windows для разработчиков, которым нужно обеспечивать повышенный контроль доступа к сети.

Microsoft.VisualBasic — Содержит типы, поддерживающие среду выполнения Visual Basic.

System.Text — Содержит классы, которые представляют кодировки ASCII и Unicode; абстрактные базовые классы для преобразования блоков знаков в блоки байтов и обратно; вспомогательный класс, который обрабатывает и форматирует объекты типа String, не создавая промежуточные экземпляры String.

System — Содержит базовый класс и фундаментальный класс, которые определяют часто используемые значения и ссылочные типы данных, события и обработчики событий, интерфейсы, атрибуты обработки и исключения.

System.Diagnostics — Предоставляет классы для взаимодействия с системными процессами, журналами событий и счетчиками производительности.

System.Threading — Предоставляет собой классы и интерфейсы для многопоточного программирования в среде разработки.

System.Security.AccessControl — Предоставляет программные элементы, которые контролируют доступ к защищаемым объектам и операции аудита, связанные с безопасностью этих объектов.

System.Security.Principal — Определяет объект субъекта, представляющий контекст защищенности, в котором работает код.

System.Management — Предоставляет доступ к обширному набору управляющей информации и событий, связанных с системой, устройствами и приложениями, которые поддерживают инфраструктуру инструментария управления Windows.

System.Windows.Forms — Содержит классы для создания приложений Windows, которые используют преимущества расширенного пользовательского интерфейса операционной системы Microsoft Windows.

System.Data — Предоставляет доступ к классам, представляющим архитектуру ADO.NET. ADO.NET позволяет создавать компоненты, которые эффективно управляют данными из нескольких источников данных.

System.Linq — Предоставляет классы и интерфейсы, поддерживающие запросы с внедрением LINQ.

# **2 Разработка приложения**

## **2.1 Разработка графического интерфейса**

На рис.1 показан интерфейс файлового менеджера, который обладает следующими функциональными возможностями:

Кнопка «←»- возвращается в предыдущую директорию.

Кнопка «Задание»-выводит информацию о имени ПК и потоках выбранного процесса в отдельный процесс.

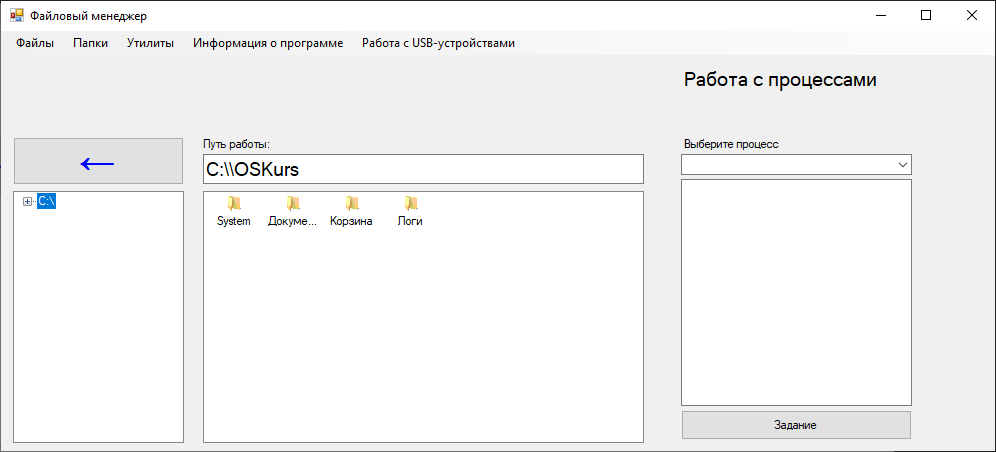


Рис. 1 – Интерфейс файлового менеджера

В основном меню Рис 1.1 находятся основные функции работы с файловым менеджером.

Раздел «Файлы» содержит функции для работы с файлами (создание, переименование и т. д.).

Раздел «Папки» содержит функции для работы с паками (создание, переименование и т.д.).

Раздел «Утилиты» позволяет пользователю запускать программные утилиты, такие как: «Монитор ресурсов», «О дисках», «О системе».

Раздел «Работа с USB-устройствами» содержит функции для работы с подключенными USB-устройствами (Отключение устройства, Переключение на USB).

Раздел «Информация о программе»- позволяет нам узнать информацию о программе: ее создателях, языке программирования и т.д.



Рис. 1.1 – Интерфейс основного функционального меню файлового менеджера

**2.2 Разработка программного обеспечения**

В данном пункте нам необходимо показать, как будет производиться разработка нашего программного обеспечения.

Для начала мы инициализируем конструктор для того, чтобы выводилась только определенная директория Рис. 2.

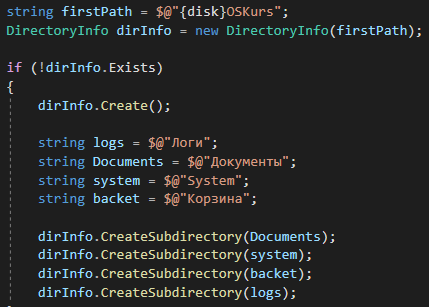


Рис. 2- Создание директорий

Затем, необходимо добавить функцию, позволяющую открывать каталоги в разрабатываемом файловом менеджере Рис. 3.

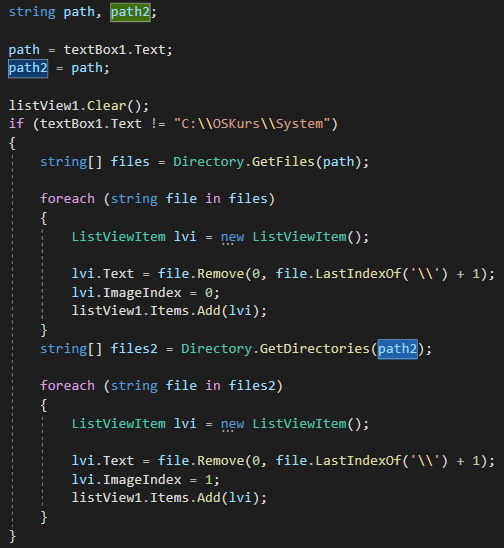


Рис. 3- Функция открытия директории

По заданию необходимо иметь возможность удаления элементов в корзину Рис. 4.

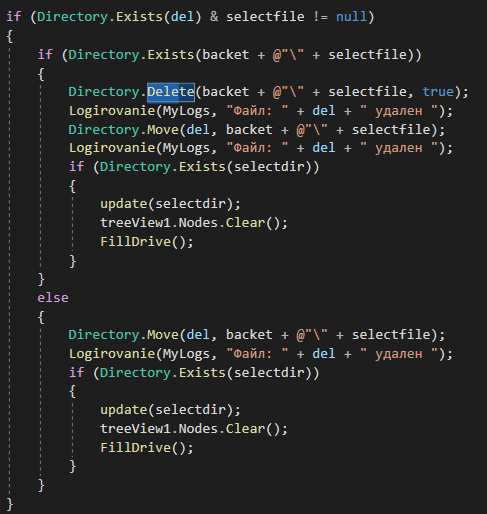


Рис. 4 – Удаление файлов в корзину

Помимо удаления в корзину необходимо реализовать возможность полного удаления элементов Рис. 5.

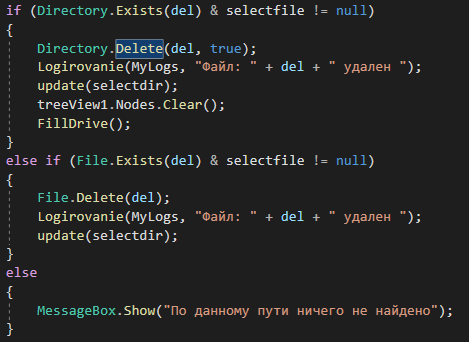


Рис. 5 – Полное удаление элементов

## **2.3 Обеспечение минимального функционала**

В данном этапе работы необходимо обеспечить взаимодействие со съёмными носителями, подключаемые к ОС Рис. 6.

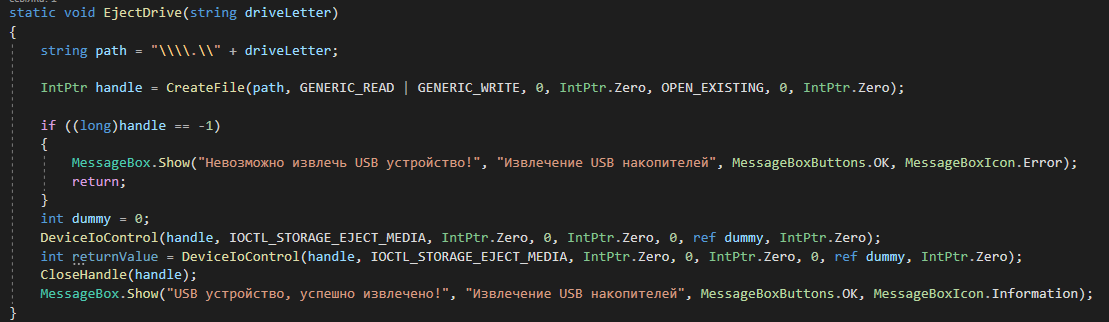


Рис. 6 – Обнаружение и отключение USB-устройства

Также в данном задании необходимо обеспечить открытие в разрабатываемом приложении запуск встроенных системных утилит ОС.

Утилиты: Монитор ресурсов Рис. 7, Информация о дисках Рис. 8, Информация о системе Рис. 9.

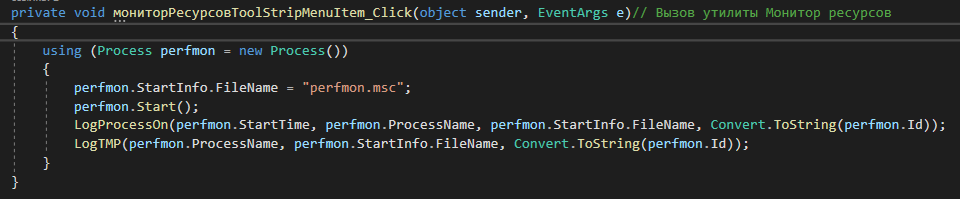


Рис. 7 - Запуск программы «монитор ресурсов»

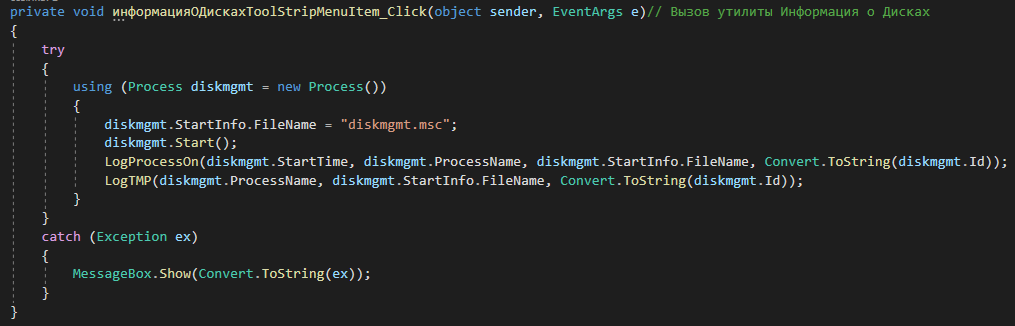


Рис. 8 – Запуск программы «о дисках»

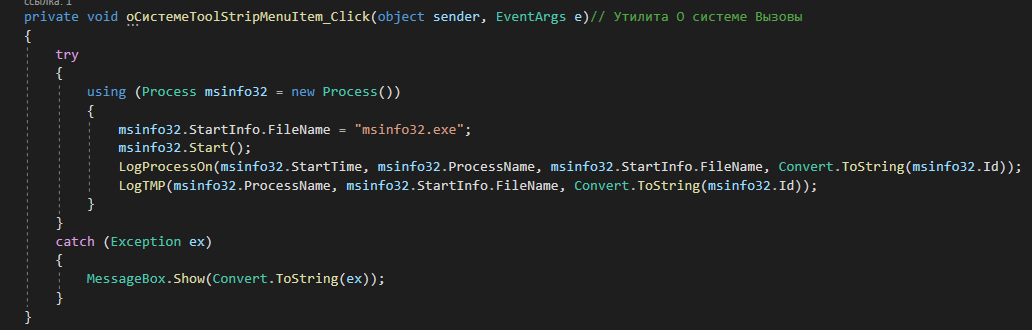


Рис.9 – Запуск программы «информация о системе»

**2.4 Добавление основного функционала**

На данной стадии разработки проекта следует изложить каким образом в программе станет функционировать межпроцессорное связь. Но перед началом описания, следует понять терминологию определения межпроцессорной связи.

Межпроцессорное взаимодействие — это набор методов для обмена данными между потоками процессов. Процессы могут запускаться на одном компьютере или на разных, подключенных к сети. В нашем случае мы будем работать с сокетным межпроцессным взаимодействием.

Разберем терминологию определения Сокета. Сокет – это интерфейс, который представляет собой совокупность адреса в сети и используемого порта. Один процесс является «сервером» и передает данные второму процессу – «клиенту», по его «запросу».

По заданию нам необходимо сделать вывод имени компьютера, также для прикладного процесса, выбранного в списке выполняющихся процессов, вывести список его потоков с указанием свойств каждого потока Рис. 10.



Рис.10 – Вывод свойств потоков и имени компьютера в textbox

А также реализовать получение имени компьютера Рис. 12.



Рис.11 – Реализация получения имени ПК

Работа межпроцессного взаимодействия путем использования сокетов Рис. 12.

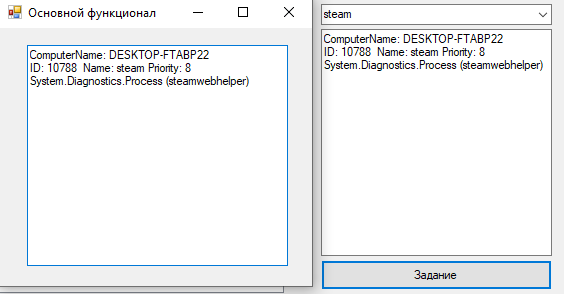


Рис. 12 – Вывод потоков и имени компьютера

Далее нам необходимо создать систему логирования действий пользователя с последующей записью в файл. Имя файла задается пользователем вручную Рис. 13.

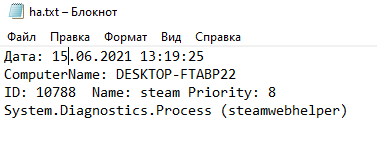


Рис.13- Созданный лог-файл

Далее нам необходимо создать UML-диаграмму Рис. 12 и блок-схему разрабатываемого файлового менеджера Рис. 15. UML-диаграмма это язык графического описания для объектного моделирования в области разработки ПО.

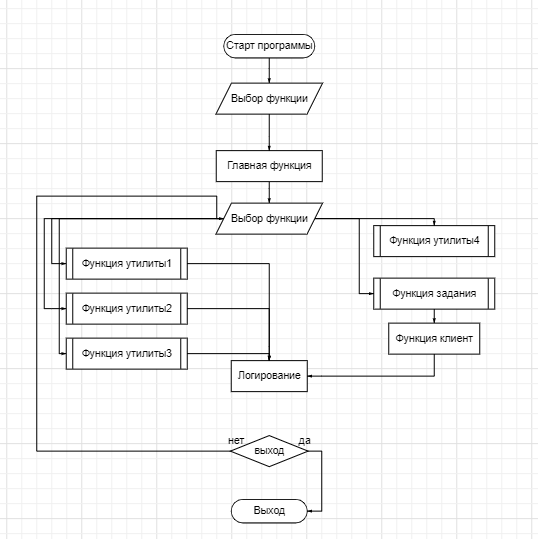


Рис. 15-Блок-схема работы файлового менеджера

**2.5 Выбор стратегии тестирования и разработка тестов**

Перед тем, как перейти к осуществлению новой стадии разработки, следует понять терминологию Стратегии тестирования чтобы можно было осуществить объективное тестирование, но кроме того разрабатывать тесты и методы их выполнения.

Стратегия тестирования – метод, применяемый для отбора тестов, который должны быть интегрирован в тестовый комплекс.

Подразделяют два типа стратегии тестирования:

1. Тестирование “Черного ящика”.

2. Тестирование “Белого ящика”.

Тестирование “Черного ящика” – метод проверки функционального поведения программы с точки зрения внешнего мира, в котором не используется значение относительно внутренней структуры тестируемого объекта (рабочей станции/ЭВМ).

Тестирование “Белого ящика” – метод тестирования, управляемый логикой программы, позволяет исследовать внутреннюю структуру программного продукта.

На основании приведенного выше, проведем тесты работы функций программы, такие как: открытие директорий, открытие файлов, копирование файлов, удаление файлов в корзину, полное удаление файлов, отключение USB-устройства и др.

В приведенной ниже таблице будут использованы совмещенные методы оценки “Черного ящика” и “Белого ящика”.

Таблица 2

Таблица тестов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название Теста | Исходные  данные | Желаемый  результат | Ответное действие программы | Результат |
| Проверить корректное открытие директорий | Выбрать директорию, которую надо открыть | Выбранная директория будет открыта | Директория открывается | Программа работает корректно |
| Проверить корректное открытие файлов | Выбрать файл, который надо открыть | Выбранный файл будет открыт | Файл открывается | Программа работает корректно |
| Проверить корректное удаление файлов | Выбрать файл, который надо удалить | Выбранный файл будет удален | Файл удаляется | Программа работает корректно |
| Проверить корректное создание директорий | Выбрать место, где надо создать новую директорию | Будет создана новая директория | Создается новая директория | Программа работает корректно |
| Проверить корректное удаление файлов в корзину | Выбрать файл, который надо удалить в корзину | Выбранный файл удаляется в корзину | Файл удаляется в корзину | Программа работает корректно |
| Проверить корректное копирование файлов | Выбрать файл, который надо скопировать | Выбранный файл копируется | Файл копируется | Программа работает корректно |
| Проверить корректное отключение USB-устройств | Обнаружить и отключить подключенное USB - устройство | Будет обнаружено и отключено USB -устройство | Программа обнаруживает и отключает USB -устройство | Программа работает корректно |
| Проверить корректное открытие утилит ОС | Открыть встроенную утилиту ОС | Будет открыта встроенная утилита ОС | Программа открывает утилиту | Программа работает корректно |
| Проверить корректный вывод потоков и их свойств | Вывести имена потоков и их свойств | В текстовое поле будут выведены имена потоков и их свойства | Программа выводит имена потоков и их свойства | Программа работает корректно |
| Проверить корректный вывод имени ПК | Вывод имени ПК в текстовое поле | Имя ПК будет выведено в текстовое поле менеджера | Программа выводит имя ПК в текстовое поле | Программа работает корректно |

Как видно из приведенной выше таблицы, результаты тестирования разработанного нами программного продукта проведены с учетом плана тестирования, и он показал полную работоспособность.

# **Руководство пользователя**

В данном этапе разработки программного продукта необходимо написать руководство пользователя для верного использования Файловым менеджером.

Нижеприведенное руководство пользователя было разработано в соответствии с ГОСТ 19.504-79 «Единая система программной документации. Руководство программиста» и ГОСТ 34.698-90 «Автоматизированные системы. Требования к содержанию документов».

Данное руководство должно в обязательном порядке иметь следующие обозначения:

1. Назначение и условия применения программы (табл. 3).

2. Характеристики программы (табл. 4).

3. Входные и выходные данные (табл. 5).

4. Обращение к программе (табл. 6).

5. Аварийные ситуации (табл. 7).

6. Рекомендации по освоению (табл. 8).

Таблица 3

Назначение и условия применения программы

|  |  |
| --- | --- |
| Назначение программы | Программный продукт предназначен для выполнения файловых операции в операционной системе Windows, включая:  1. Копирование файлов  2. Логирование операции  3. Открытие файлов |
| Условия применения программы | Программный продукт имеет следующие системные требования:  1. Поддерживаемые ОС: Windows 7/8/10  2. Процессор: Intel® Core или AMD Ryzen  3. Оперативная память: 512 мб и более  4. Видеокарта: NVIDIA® GeForce® GTX или AMD Radeon  5. Звуковая карта: Windows-совместимая звуковая карта  6. DirectX: DirectX 9.0b  7. Жесткий диск: 500 МБ свободного места |

Таблица 4

Характеристики программы

|  |  |
| --- | --- |
| Характеристики программы | 1. Режим работы – в зависимости от пользователя  2. Контроль правильности выполнения программного продукта – с помощью внутренних системных операторов |

Таблица 5

Входные и выходные данные

|  |  |
| --- | --- |
| Входные и выходные данные | 1. Источниками информации являются:  1.1. Данные, вводимые конечным пользователям  1.2 Программные данные, производимые с помощью работы ПО |

Таблица 6

Обращение к программе

|  |  |
| --- | --- |
| Обращение к программе | 1. Пользователь нажимает на исполняемый файл, необходимый для запуска.  2. После запуска программного продукта, пользователь может произвести действия  копирования, удаления и т.д. путем перемещения по директориям с помощью окон TreeView и ListView.  3. Пользователь может активировать дополнительные функции программного продукта, с помощью вкладок основного меню “Работа с USB-устройствами”, “Файлы”, “Папки” и кнопки “Задание”. |

Таблица 7

Аварийные ситуации

|  |  |
| --- | --- |
| Аварийные ситуации | В данном программном продукте возможны следующие аварийные ситуации:  1. При попытки удалить файлы, находящиеся в корневой папке «System»  появится предупреждения о невозможности данной операции.  2.При попытки переименовать системные файлы и папки появится предупреждение о невозможности данной операции.  3.При нажатии кнопки «Задание» программа зависнет на несколько секунд для обработки и вывода информации |

Таблица 8

Рекомендации по освоению

|  |  |
| --- | --- |
| Рекомендации по освоению | Для работы с программным продуктом “Файловый менеджер” пользователю необходимо:  1. Иметь навыки работы с рабочей станцией.  2. Ознакомиться с действующим руководством пользователя.  3. Если данный программный продукт планируется использовать в предприятиях или организациях – провести инструктаж в соответствии с настоящим руководством пользователя.  4. Первый запуск программного продукта должен проводить системный администратор или лицо, уполномоченное за техническое оснащение квартиры/предприятия в целях отладки. |

# 

# **3 Пример работы разработанного приложения**

* 1. **Стандартная работа приложения**

В данном пункте нужно описать работу разработанного приложения, для того понимать, как работает программный продукт “Файловый менеджер”.

Пользователь может открыть исполняемый файл с устройства пользователя.

При запуске программы пользователь увидит два окна, в левом окне отображаются каталоги, а в правом будет выводиться файлы хранящихся в этих папках Рис. 16.

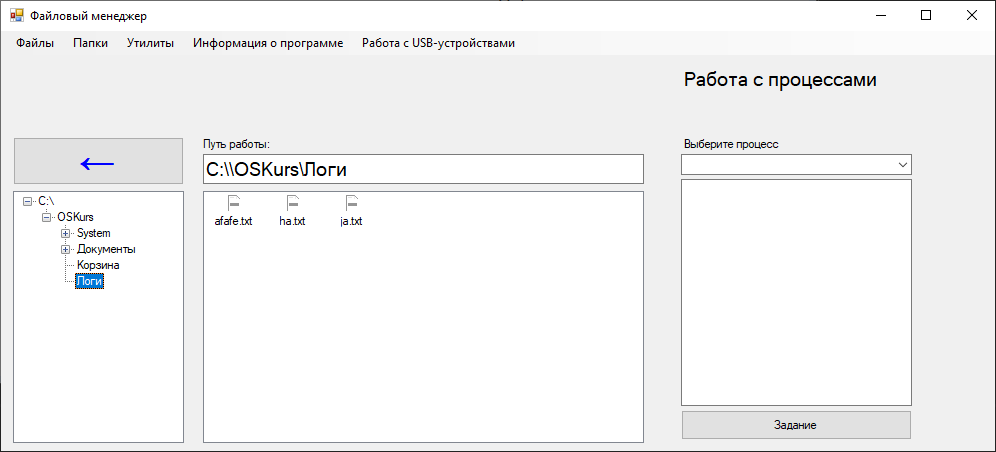


Рис. 16- Показ содержимого папок

Открытие файла осуществляется с помощью двойного щелчка по нему Рис. 17.

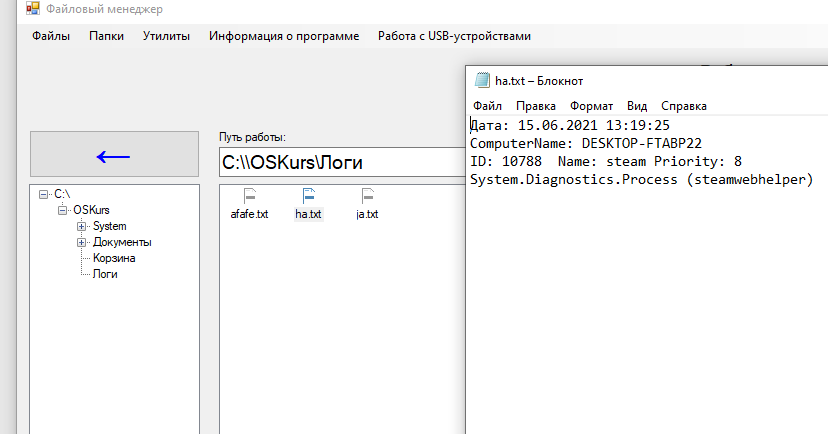


Рис. 17 - Открытие файла

Для удаления в корзину необходимо выбрать файл, после чего нажать ПКМ и выбрать «Удалить» и файл будет перемещен в корзину Рис. 18

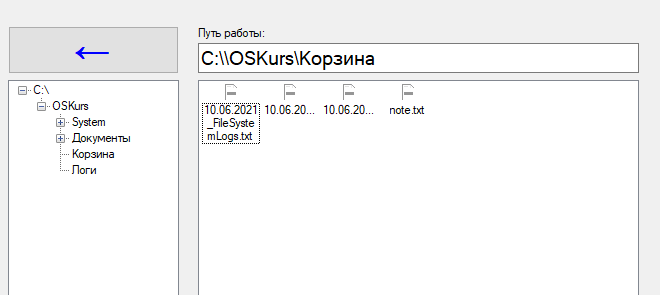


Рис.18 - Удаление файла в корзину

Для получения пользователем информации о имени ПК и вывода информации о потоках выбранного процесса, необходимо в меню выбора процесса кликнуть на интересующий процесс, после чего нажать кнопку «Задание», после выполнения работы, в самом правом окне будет выведена информация (рис.19).

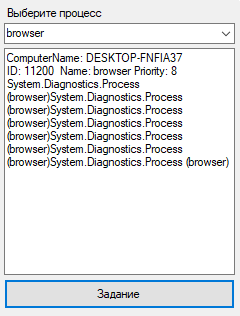


Рис.19- Вывод потоков и имени компьютера

## **Дополнительные функции программы.**

В данном этапе разработки программного продукта нужно описать дополнительные функции программы.

В них входят следующие функции программы:

1. Отслеживание процессов внутри “Файлового менеджера”.

При запуске процесса информация о нем будет записываться в текстовый файл.

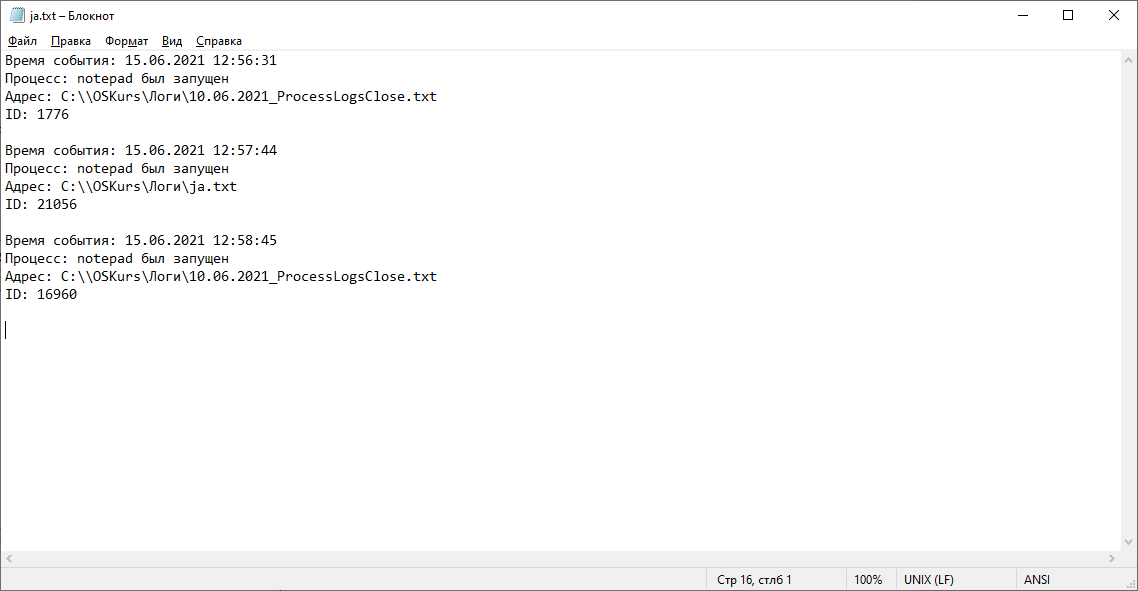


Рис. 20 - Информация о процессах запущенных в файловом менеджере

1. Запуск встроенных в ОС утилит с помощью файлового менеджера.

Если пользователю нужно будет открыть встроенную утилиту для мониторинга производительности ему необходимо будет выбрать вкладку «Утилиты» в основном меню и нажать на «Монитор» Рис. 21 или «О дисках» Рис. 22 или «О системе» Рис. 23.

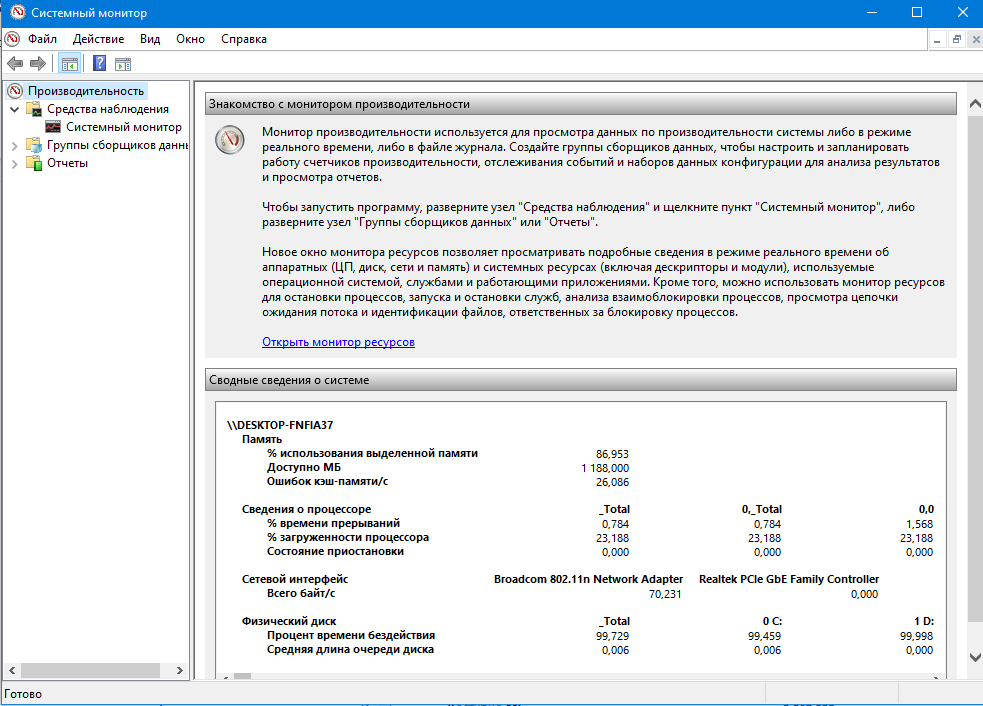
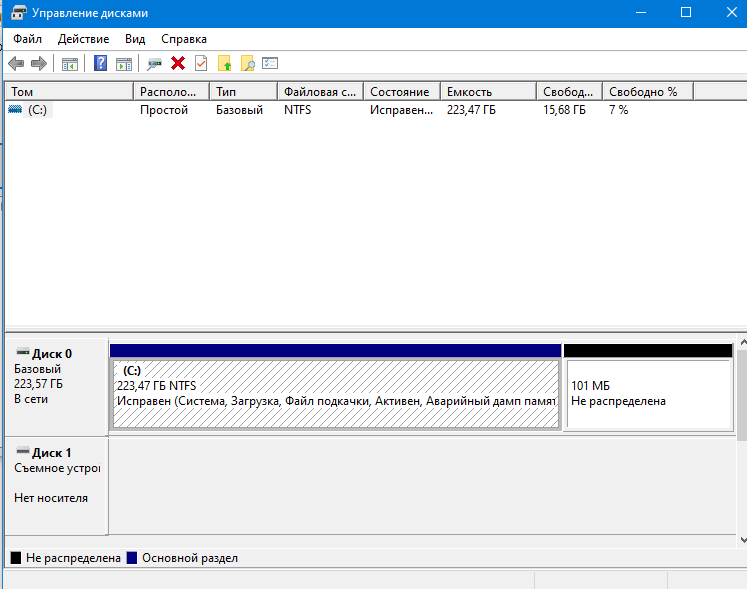


Рис. 21- Монитор ресурсов

Рис. 22 – Управление дисками

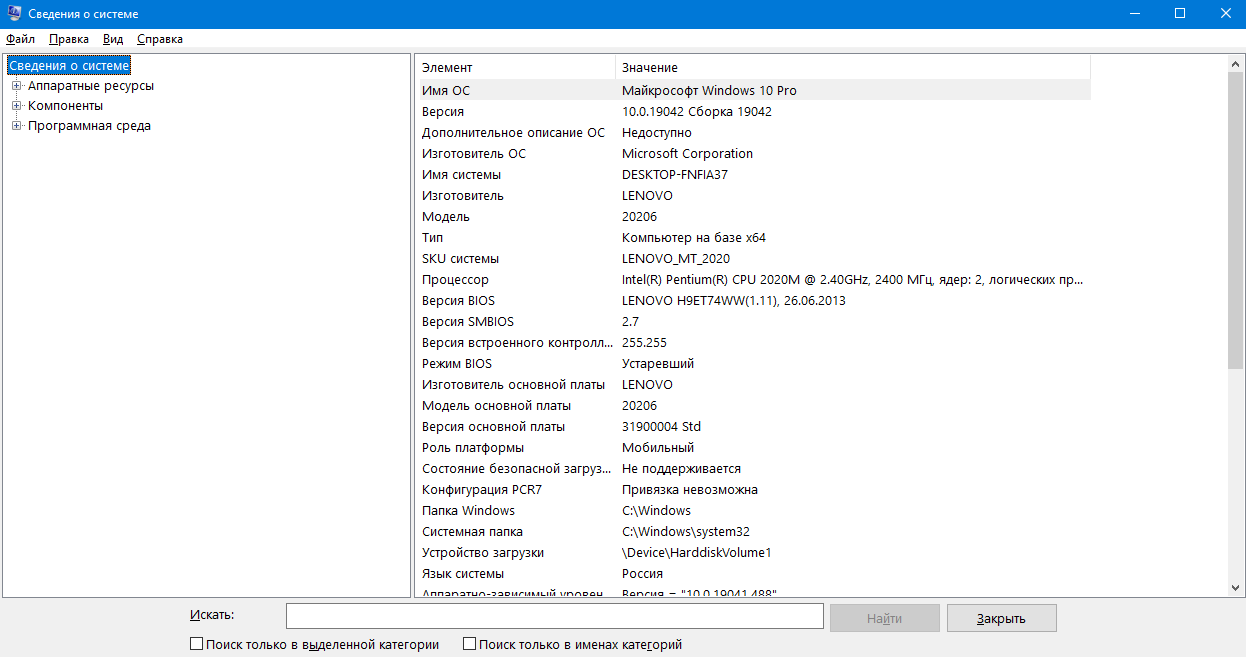


Рис. 23 – Сведения о системе

3. Логирование операции.

В текстовый файл будут записываться все действия пользователя (рис.24).

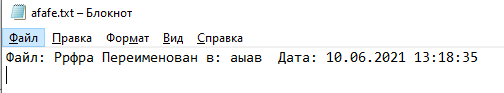


Рис. 24 – Лог-файл

# **Заключение**

Подводя результаты работы согласно разработке ПО, важно отметить, что с целью исполнения курсовой работы был проведен анализ темы в области создания приложения под платформу Windows, было подготовлено руководство пользователя по иcпользованию программы, спроектирован программный продукт. За период выполнения работы были исследованы методы формирования директорий (создание, удаление, запись), запись операций пользователя в отдельный лог-файл, был рассмотрен метод получения данных о потоках выбранного пользователем процесса (сохранения протокола процессов), вывод имени Компьютера, вывод загруженности процессора (в процентах), открытие USB-девайсов и их выключение, кроме того, был проработан вид межпроцессного (клиент-сервер) взаимодействия с применением Сокетов.

**Приложение А**

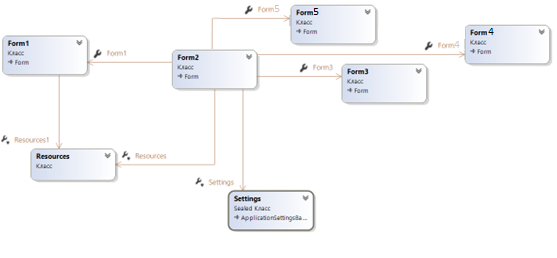


Рис. 14 - UML-диаграмма