

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
учреждение образования  
«Гродненский государственный университет имени Янки Купалы»  
Факультет математики и информатики  
Кафедра современных технологий программирования

Лысяков Кирилл Евгеньевич

**Разработка текстового редактора “RTFVim”**

Курсовая работа  
по дисциплине «Языки программирования»  
студента 2 курса специальности  
1-40 01 01 «Программное обеспечение информационных технологий»  
дневной формы получения образования

Научный руководитель  
Деева Наталия Владимировна,  
Старший преподаватель  
кафедры современных  
технологий  
программирования

Гродно, 2020

## РЕЗЮМЕ

Лысяков Кирилл Евгеньевич

Тема курсовой работы: «Разработка текстового редактора формата RTF».

Работа содержит: 23 страниц, 4 использованных источников литературы.

Ключевые слова: C#, Microsoft Visual Studio, WPF, MVVM, RTF.

Цель курсовой работы: разработка «Текстового редактора в формате RTF».

Объект исследования: приложение Текстовый редактор Word и его аналоги, текстовые процессоры.

Предмет исследования: реализация приложения «Текстовый редактор формата RTF» на ЭВМ с помощью WPF и языка программирования C#, с использованием Microsoft Visual Studio и паттерна MVVM.

В работе были использованы следующие методы: сравнительный анализ, проведение аналогии.

Авторская характеристика работы: приложение «Текстовый редактор формата RTF» написано на языке программирования C#, и представляет собой реализацию алгоритмов обработки массивов и классов, реализованные в среде разработки Microsoft Visual Studio.

## Summary

K. E. Lysyakov

Topic of the course work: "development of a text editor in rtf format".

The work contains: - pages, - used literature sources.

Keywords: c#, microsoft visual studio, wpf, mvvm, rtf.

The purpose of the course work: development of a "text editor in rtf format".

Object of research: The Word text editor application and its analogues, word processors.

Subject of research: implementation of the application "Text editor of RTF format" on a computer using wpf and the C# programming language, using Microsoft Visual Studio and the MVVM pattern.

The following methods were used in the work: comparative analysis, drawing an analogy.

Author's description of the work: the application "Text Editor of RTF format" is written in the C# programming language, and is an implementation of algorithms for processing arrays and classes implemented in the Microsoft Visual Studio development environment.

## Содержание

ВВЕДЕНИЕ.....	5
<b>ГЛАВА 1. Анализ предметной области.....</b>	<b>6</b>
1.1 Описание предметной области .....	6
1.2 Существующие решения.....	6
1.3 Выводы по главе 1 .....	9
<b>ГЛАВА 2. ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ.....</b>	<b>11</b>
2.1 Проектирование функциональной части приложения.....	11
2.2 Архитектура приложения.....	12
2.3 Проектирование графического интерфейса приложения.....	12
2.4 Выводы по главе 2.....	15
<b>ГЛАВА 3. Программная реализация приложения.....</b>	<b>16</b>
3.1 Обзор и анализ средств реализации.....	16
3.2 Описание модели приложения, сущности.....	17
3.3 Описание структуры проекта .....	21
3.4. Выводы по главе 3.....	22
<b>ЗАКЛЮЧЕНИЕ.....</b>	<b>23</b>
<b>СПИСОК ЛИТЕРАТУРЫ .....</b>	<b>24</b>

## ВВЕДЕНИЕ

Практически каждый пользователь компьютера встречается с необходимостью подготовки тех или иных документов — писем, статей, служебных записок, отчетов, рекламных материалов и т.д. Разумеется, эти документы можно подготавливать и без компьютера, например на пишущей машинке. Однако с появлением персональных компьютеров стало значительно проще и удобнее, а следовательно, и выгоднее подготавливать документы с помощью компьютеров.

При использовании персональных компьютеров для подготовки документов текст редактируемого документа выводится на экран, и пользователь может в диалоговом режиме вносить в него свои изменения. Все внесенные изменения сразу же отображаются на экране компьютера, и потом при распечатке выводится красиво и правильно оформленный текст, в котором учтены все сделанные пользователем исправления. Пользователь может переносить куски текста из одного места документа в другое, использовать несколько видов шрифтов для выделения отдельных участков текста, печатать подготовленный документ на принтере в нужном числе экземпляров.

Удобство и эффективность применения компьютеров для подготовки текстов привели к созданию множества программ для обработки документов. Такие программы называются *редакторами текстов* (Word Processors). Возможности этих программ различны — от программ, предназначенных для подготовки небольших документов простой структуры, до программ для набора, оформления и полной подготовки к типографскому изданию книг и журналов (издательские системы).

Основные функции этого класса прикладных программ заключаются в вводе и редактировании текстов. Дополнительные функции состоят в автоматизации процессов ввода и редактирования. Для операций ввода, вывода и сохранения данных текстовые редакторы вызывают и используют системное программное обеспечение. Впрочем, это характерно и для всех прочих видов прикладных программ, и в дальнейшем мы не будем специально указывать на этот факт.

С этого класса прикладных программ обычно начинают знакомство с программным обеспечением и на нем отрабатываются первичные навыки взаимодействия с компьютерной системой.

# ГЛАВА 1. Анализ предметной области

## 1.1 Описание предметной области

Приложение «Текстовый редактор» реализовано при помощи Microsoft Visual Studio. Microsoft Visual Studio – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии WPF, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone. NET Compact Framework и Silverlight

Графический интерфейс был разработан с помощью технологии WPF (Windows Presentation Foundation), которая является частью экосистемы платформы .NET и представляет собой подсистему для построения графических интерфейсов.

Главные причины выбора именно этой технологии:

- Декларативное определение графического интерфейса с помощью специального языка разметки XAML
- Независимость от разрешения экрана
- Взаимодействие с WinForms
- Аппаратное ускорение графики
- Создание приложений под множество ОС семейства Windows - от Windows XP до Windows 10.

## 1.2 Существующие решения

Текстовый редактор Word с автоматической проверкой орфографии.



Текстовый редактор Microsoft Word – мощное приложение с большим количеством функций. В этой программе можно набирать и редактировать тексты, использовать разные шрифты и стили, менять цвет текста, форматировать документы, вставлять изображения, ссылки, таблицы,

диаграммы, формулы, выноски. Редактор самостоятельно проверяет орфографию, предлагает варианты написания слов, следит за расстановкой знаков препинания. Microsoft Word содержит большое количество символов и автофигур, позволяет использовать художественный текст, делать несложные презентации и Web-страницы. Для удобства работы в приложении есть много вспомогательных функций, позволяющих быстро находить нужные фразы и отрывки текста, заменять одно слово другим во всем документе и многое другое. Также нужно упомянуть удобный интерфейс, позволяющий быстро освоить приложение. Казалось бы, зачем искать что-то еще, но... есть одно «но». Microsoft Word – не бесплатное приложение. Конечно, тем, для кого работа на дому в интернете, к примеру, по набору текста стала источником стабильного и достаточно высокого дохода, имеет смысл купить этот редактор. Но, если человек использует подобное ПО достаточно редко, можно выбрать что-то похожее, только бесплатно.

Текстовый редактор LibreOffice Writer.



LibreOffice Writer – на данный момент это самый мощный среди бесплатных текстовых редакторов. Он позволяет работать с документами Microsoft Word, RTF, создавать HTML документы. В нем также можно вставлять в тексты таблицы, картинки, мультимедийные объекты и другие элементы. В LibreOffice Writer имеется редактируемый словарь и функция проверки орфографии. Интерфейс программы напоминает ранние версии Word, поэтому освоить его несложно. Тем более что есть русская версия приложения. Одним словом, этот редактор можно смело назвать бесплатным аналогом или упрощенной версией Microsoft Word. Есть и другие бесплатные приложения (AbiWord, OpenOffice), но, судя по отзывам пользователей, им далеко до LibreOffice Writer.

Текстовый редактор Блокнот.



Блокнот – это самый простой текстовый редактор, который входит в стандартный пакет установки системы Windows. Он работает с расширением TXT, но может открывать файлы INF, INI, LOG.

Редактор Блокнот имеет совсем небольшой набор функций. Он позволяет набирать тексты, выбирать шрифты, осуществлять поиск, автоматически заменять слова, сохранять документы и отправлять их на печать. Такие функции, как форматирование, проверка орфографии, вставка изображений, использование разных цветов в этом приложении недоступны.

Тем не менее, Блокнот полезен не только начинающим, но и опытным пользователям, как простой и удобный вспомогательный инструмент. Вот лишь некоторые возможности этой программы:

Блокнот работает с текстами в кодировках Unicode, UTF-8 и ANSI и позволяет выполнять преобразование одной кодировки в другую. Для этого можно просто выбрать нужную кодировку при сохранении файла.

Если набрать текст в Microsoft Word или другом аналогичном редакторе, а потом опубликовать на сайте или в блоге, там появится много лишнего. Конечно, если движок нормальный, этого не произойдет. Но лучше сначала вставить текст в блокнот, а уже оттуда скопировать его для публикации. Для этого можно использовать горячие клавиши Ctrl+C (скопировать) и Ctrl+V (вставить).

В Блокноте можно делать или редактировать несложные Web-страницы. Для сохранения файла в формате HTML достаточно в поле «имя файла» ввести «название.html», а в поле «тип файла» выбрать «все файлы».

Блокнот позволяет убрать и тот «мусор», который может появиться при копировании текста из каких-нибудь редакторов. Например, есть такая программа для распознавания текста, которая расшифровывает сканированные документы. Очень удобно, но все распознать она не может, и в тексте остается много непонятных значков. Чтобы их убрать, достаточно скопировать результат в Блокнот.

Редактор текста Google, позволяющий печатать текст онлайн бесплатно.



В Google есть замечательный редактор, позволяющий набирать текст прямо в браузере, в режиме онлайн, без установки приложения на компьютер. Правда, чтобы получить к нему доступ, надо *создать аккаунт в Google* или воспользоваться уже существующим. После этого пользователю будут доступны удобные сервисы для работы с документами, в том числе и редактор текста онлайн, о котором хочу сказать пару слов.

Этот онлайн сервис находится во вкладке «Диск», которая появляется в верхней части браузера при выборе системы поиска Google. Чтобы начать печатать текст, надо нажать на яркую красную кнопку «Создать», потом «Документ», после этого в новой вкладке откроется редактор.

По своим функциональным возможностям редактор текста Google – это что-то среднее между Microsoft Word и Блокнотом. Он поддерживает несколько текстовых форматов (DOCX, RTF, TXT), а также HTML, PDF. В нем можно форматировать тексты, использовать разные шрифты и стили, менять цвет текста, вставлять таблицы, рисунки, формулы, ссылки, специальные символы, номера страниц, сноски и комментарии, осуществлять поиск и проверку орфографии



(редактор подчеркивает слова с ошибками и предлагает варианты их написания). Еще одна уникальная функция – это *перевод текста* на разные языки. Переведенный текст открывается в новом окне, что позволяет сравнить его с оригиналом.

Все документы автоматически сохраняются в разделе «Мой диск», где их можно оставить, если тексты еще нужны, или скачать на компьютер. Кстати, все это можно делать с мобильного телефона.

В общем, в онлайн редакторе Google есть все, что нужно для работы с текстами. Если бы он еще позволял в автоматическом режиме выполнять проверку документа на антиплагиат, это был бы лучший на данный момент бесплатный редактор. К сожалению, такого редактора текста онлайн в Яндексе пока нет.

### 1.3 Вывды по главе 1

На данном этапе разработки была выбрана идея, разрабатываемой программы. Составлялся список задач, решение которых будет реализовано в данной программе.

Основная задача — это разработать приложение, которое будет включать в себя различные функции для обработки текста.

Предполагаемые функции текстового редактора:

- Редактирование текста
- Работа с различными участками текста
- Удаление, выделение, копирование, вставка текста
- Открытие, закрытие, сохранение, создание нового текстового файла
- Изменение шрифтов (размеров, семейств)
- Изменение способов выделения текста (подчёркивание, курсив, цвет, выделение жирным)
- Поиск и замена в тексте
- Сортировка слов в тексте по алфавиту, длине
- Разделение текста на предложения
- Получение конкорданса (справка о вхождении слов в текст)
- Получение уникальных слов в тексте
- Работа с несколькими документами

Алгоритм написания программы:

- 1) Продумать основные функции.
- 2) Продумать архитектуру программы.
- 3) Написать программу.
- 4) Тестирование.

В последнее время компьютерные технологии продвигаются очень интенсивно, и это способствует бурному развитию программного обеспечения. Каждые полгода выходят продукты с множеством нововведений. Так и текстовые редакторы не стоят на месте. С каждым разом все больше и больше функций заключают в себе данные программы. Но их развитие поставлено таким образом, что с каждой новой версией программа сохраняет предыдущий набор возможностей и пользователь может использовать как старые, так и новые функции, последние введены лишь для облегчения работы с программой.

Широкие возможности текстовых редакторов позволили компьютеру практически вытеснить пишущие машинки из делопроизводства, а использование компьютерных издательских систем во многом изменило организацию подготовки рукописи к изданию, автоматизировало труд людей нескольких типографских профессий - верстальщика, наборщика, корректора и др.

## ГЛАВА 2. ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

### 2.1 Проектирование функциональной части приложения

В первой главе были разобраны реализации решений по данной тематике. Сейчас необходимо разработать набор функций программы. Так как целью данной работы является создание текстового редактора, то пользователю должен быть организован удобный и быстрый доступ к функциям редактирования текста. Было принято решение разместить всё в одном окне с возможность открывать и сворачивать часть интерфейса по желанию пользователя, оставляя максимуму места для написания текста. Сами функции были разделены на две группы:

- для взаимодействия с самой программой и ОС
- для взаимодействия с текстом

Для обеспечения удобства работы с окном маленьких размеров при открытии элемента одной группы все другие из той же группы скрываются.

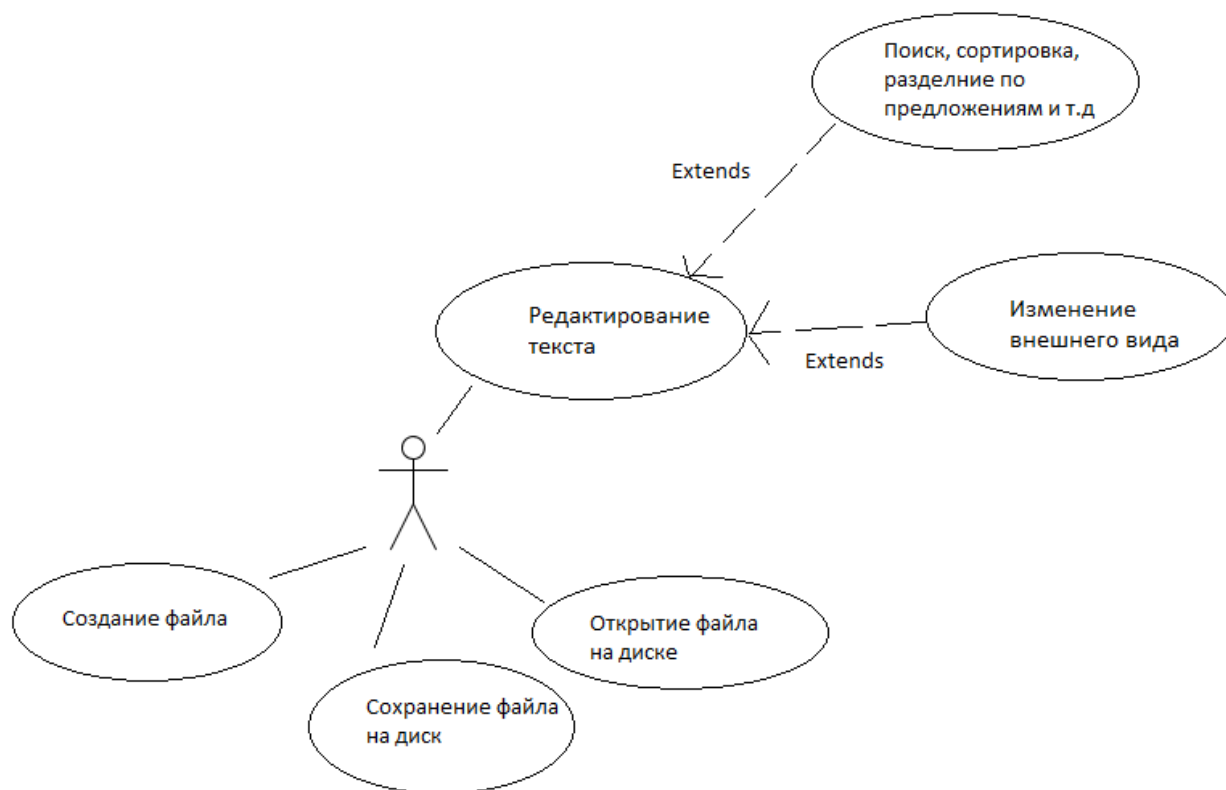


Рис.2.1.1 UML Диаграмма

## 2.2 Архитектура приложения

Программа имеет одно главное окно, в котором находится место для редактирования текста, путь к файлу, кнопки для переключения между разными текстовыми документами, кнопки для открытия панелей расширяющих возможности редактирования, кнопка для взаимодействия с ОС, сами панели.

## 2.3 Проектирование графического интерфейса приложения

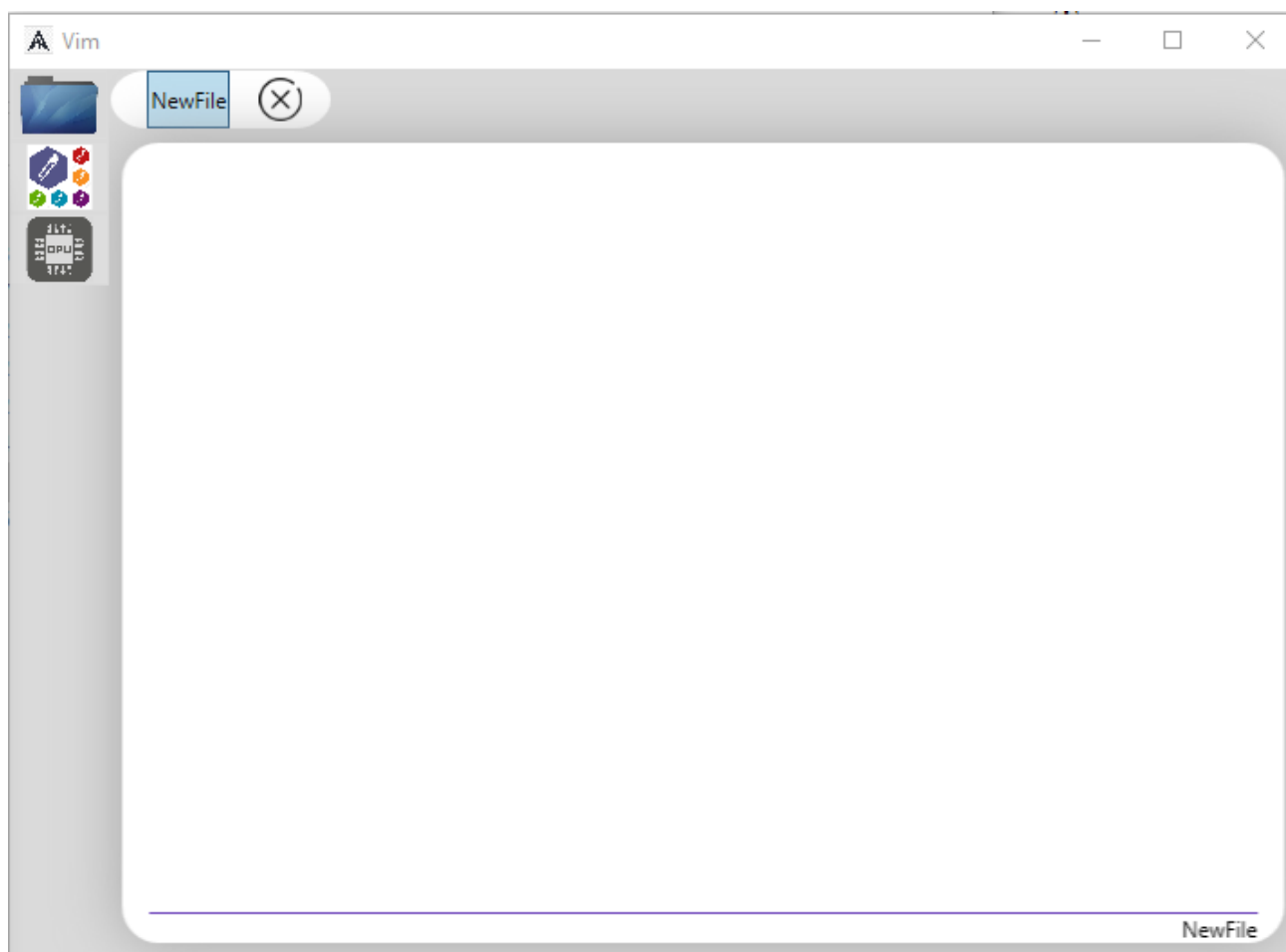


Рис.2.3.1 Главное окно

Нажав на кнопку с иконкой папки откроется панель для открытия, сохранения и создания нового файла (Рис.2.3.2).

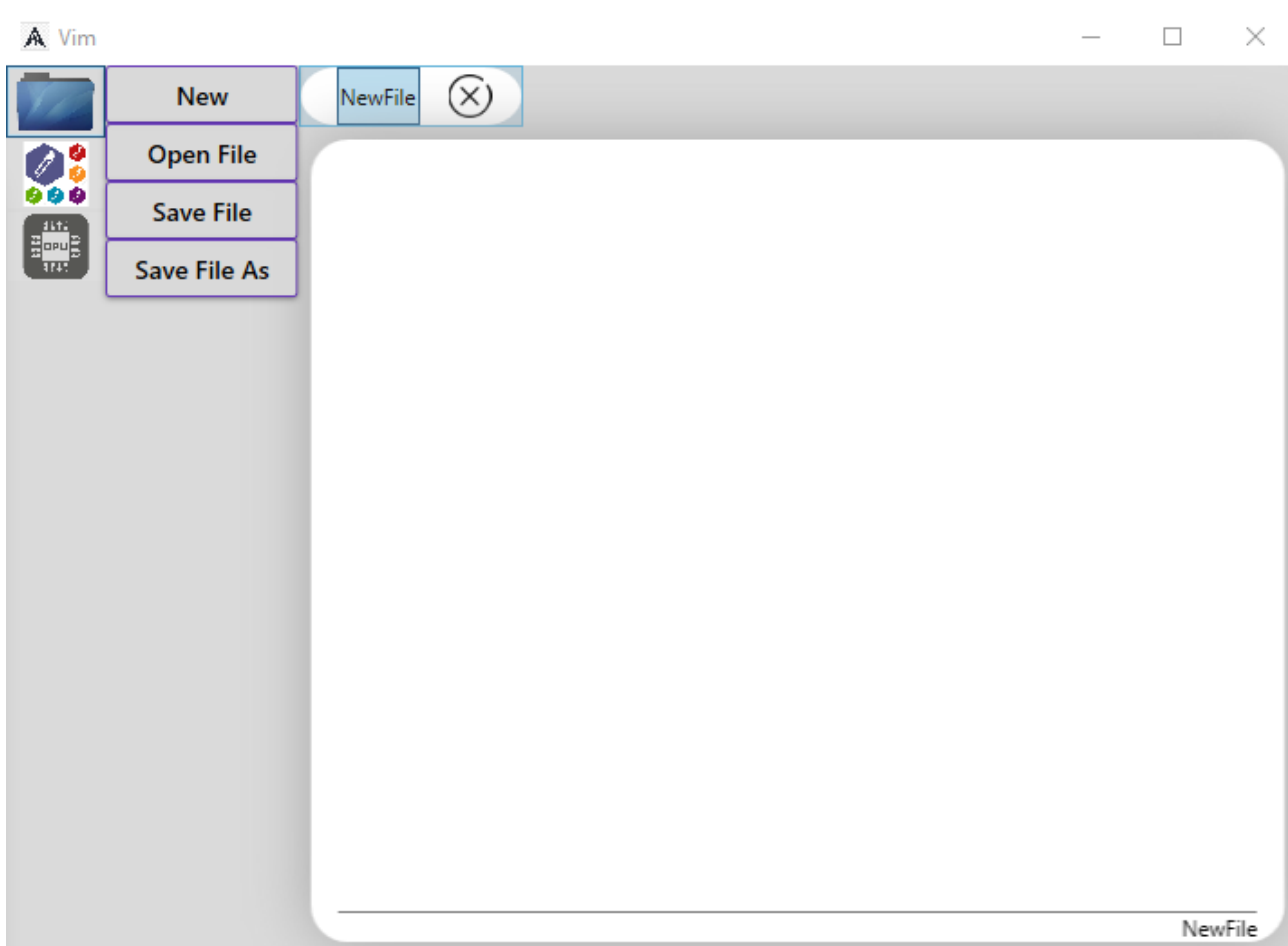


Рис.2.3.2 Панель для работы с файлами

Вторая кнопка слева (сверху-вниз) отвечает за открытие панели изменения вида текста (Рис.2.3.3):

- Шрифты
- Размер
- Выделение жирным
- Подчёркивание
- Курсив
- Цвет фона текста
- Цвет текста
- Вставка изображения

Стоит отметить, что выбранные семья шрифта, размер, курсив, подчёркивание и выделение жирным автоматически меняется при изменении расположения курсора.

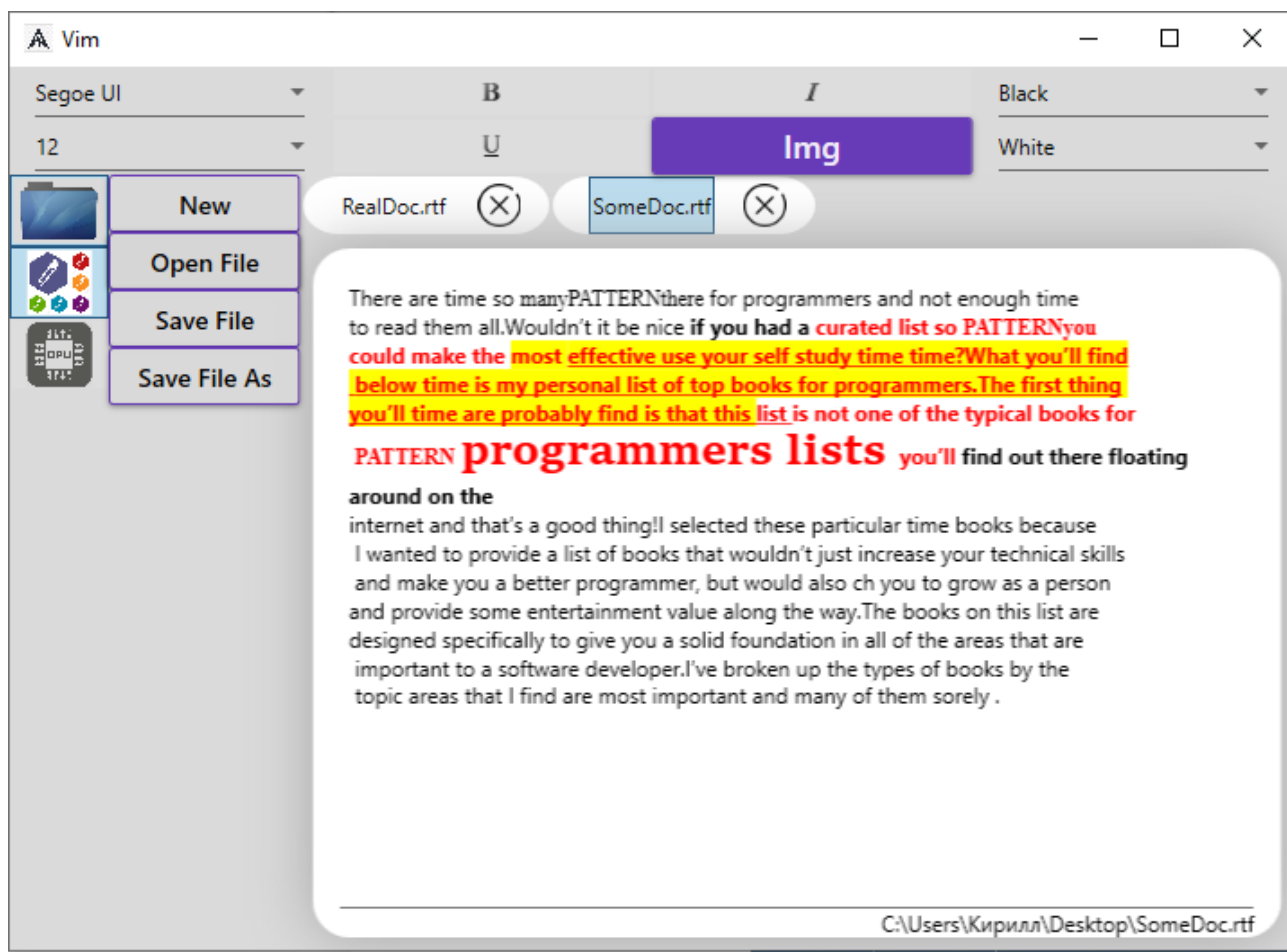


Рис.2.3.3 Панель форматирования текста

На рис.2.3.3. можно увидеть две кнопки сразу над текстовым листом, которые позволяют переключаться между разными файлами. Какой файл выбран в данный момент выделяется на кнопке.

Кнопка с иконкой CPU отвечает за функции поиска, сортировки, замены, поиска уникальных слов, разделения по предложениям и получении справки о вхождении слов в текст (Рис.2.3.5).

Сортировка происходит по словам и может быть разных типов (Рис.2.3.4):

- По алфавиту A-Z
- По алфавиту Z-A
- По длине возраста
- По длине убыва

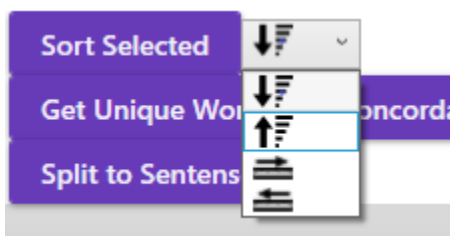


Рис.2.3.4 Сортировка

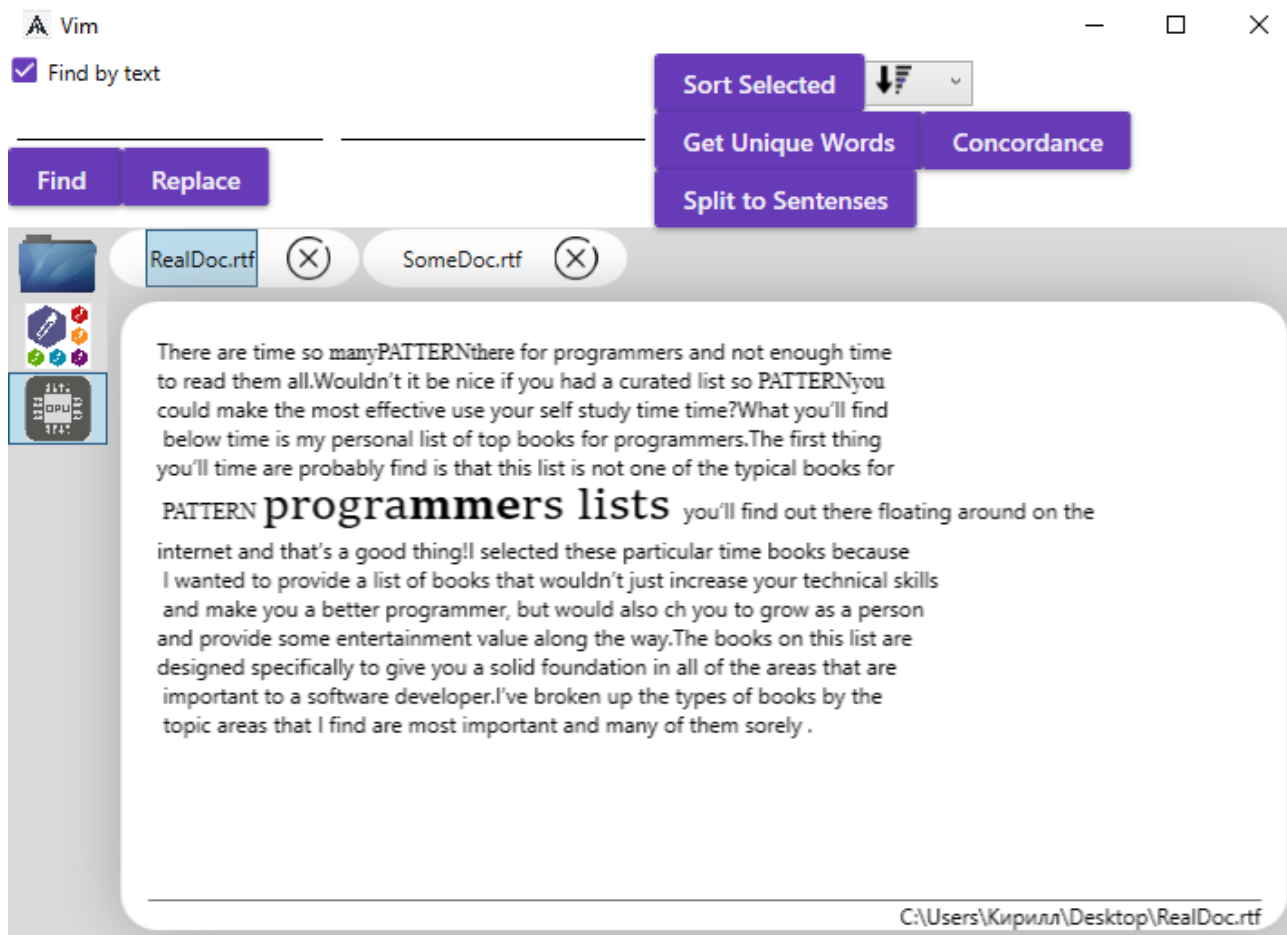


Рис.2.3.5 Панель текстового процессора

## 2.4 Выводы по главе 2

На этом этапе были поставлены конкретные задачи и был определен дизайн проекта.

Графический интерфейс было решено реализовать с использованием платформы WPF. Были проанализированы существующие решения для того, чтобы избежать возможных ошибок в проектировании.

Для более детального ознакомления требования были сопровождаемы картинками

## ГЛАВА 3. Программная реализация приложения

### 3.1 Обзор и анализ средств реализации

C# -объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft.NETFramework. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC23270. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Переняв многое от своих предшественников —языков C++, Pascal, Модула, Smalltalkи, в особенности, Java—C#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличие от C++ не поддерживает множественное наследование классов (между тем допускается множественное наследование интерфейсов

Технология WPF(WindowsPresentationFoundation) является частью платформы .NETи представляет собой подсистему для построения графических интерфейсов. Одной из важных особенностей является использование языка декларативной разметки интерфейса XAML, основанного на XML: можно создавать насыщенный графический интерфейс, используя или декларативное объявление интерфейса, или код на языке C#.

Использование паттерна MVVM позволяет тестировать все детали интерфейса, не используя сложных инструментальных средств. Паттерн MVVM (Model-View-ViewModel) позволяет отделить логику приложения от визуальной части (представления). Данный паттерн является архитектурным, то есть он задает общую архитектуру приложения.

Модель описывает используемые в приложении данные. Модели могут содержать логику, непосредственно связанную этими данными, например, логику валидации свойств модели. В то же время модель не должна содержать никакой логики, связанной с отображением данных и взаимодействием с визуальными элементами управления.



View или представление определяет визуальный интерфейс, через который пользователь взаимодействует с приложением. Применительно к WPF представление - это код в xaml, который определяет интерфейс в виде кнопок, текстовых полей и прочих визуальных элементов.

ViewModel или модель представления связывает модель и представление через механизм привязки данных. Если в модели изменяются значения свойств, при реализации моделью интерфейса INotifyPropertyChanged автоматически идет изменение отображаемых данных в представлении, хотя напрямую модель и представление не связаны.

## 3.2 Описание модели приложения, сущности

Согласно паттерну MVVM классы я поделил по 3 категориям.

Начнём с Model (Рис.3.2.1):

TextProcessor – написанная мной библиотека, позволяющая представить текст в виде иерархии классов для более удобной работы.

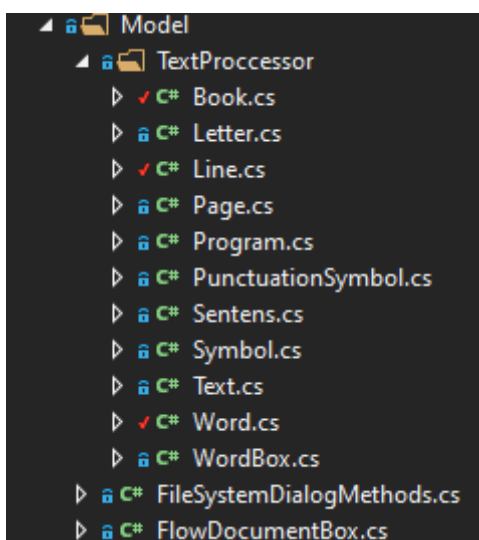
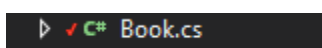
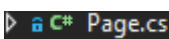



Рис.3.2.1







Содержит:

- набор  Page.cs
- метод считывания файла
- метод деления линий на страницы
- максимальное кол-во линий в странице
- конструктор делящий на страницы при считывании из файла
- конструктор делящий на страницы из строки





▸  C# Page.cs



Содержит

- набор   Line.cs
- методы получения уникальных слов со страницы
- номер страницы



▸   Line.cs

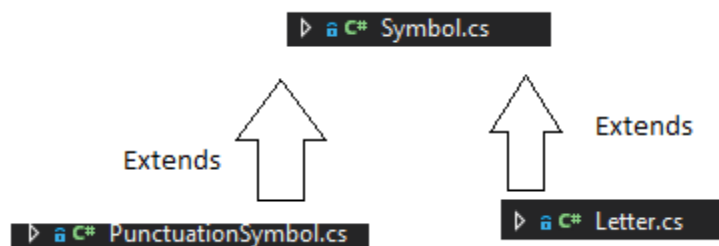
Содержит



- наборы   PunctuationSymbol.cs и   Word.cs
- номер линии
- методы получения уникальных слов со строки
- методы сортировки строки
- конструктор, делящий на слова и знаки

▸   Word.cs

Содержит

- набор   Symbol.cs
- длину слова
- конструктор



▸   Text.cs

Содержит:

- набор   Sentens.cs

- конструктор

▸ C# Sentens.cs

Содержит

- наборы ▸ C# PunctuationSymbol.cs и ▸ ✓ C# Word.cs
- конструкторы заполнения через строку , наборы знаков препинания и слов
- свойство Value, которое возвращает значение в виде строки

▸ C# WordBox.cs

Содержит:

- Слово
- Номера строк, в которых оно встречалось
- Количество повторений слова
- Ссылку на место в редактируемом тексте

Programm.cs- некоторые методов для работы с текстом

Letter.cs – вспомогательные методы проверки символов

FileDialogMethods.cs- методы открытия и сохранения файлов

FlowDocumentBox.cs- удобная обёртка документа

Содержит :

- Путь в системе
- Сам документ
- Ссылку на кнопку для этого документа

ViewModel:

▲ C# ViewModel  
 ▸ C# FlowDocumentsListViewModel.cs  
 ▸ ✓ C# TextProcessorViewModel.cs

FlowDocumetnsList ViewModel.cs:

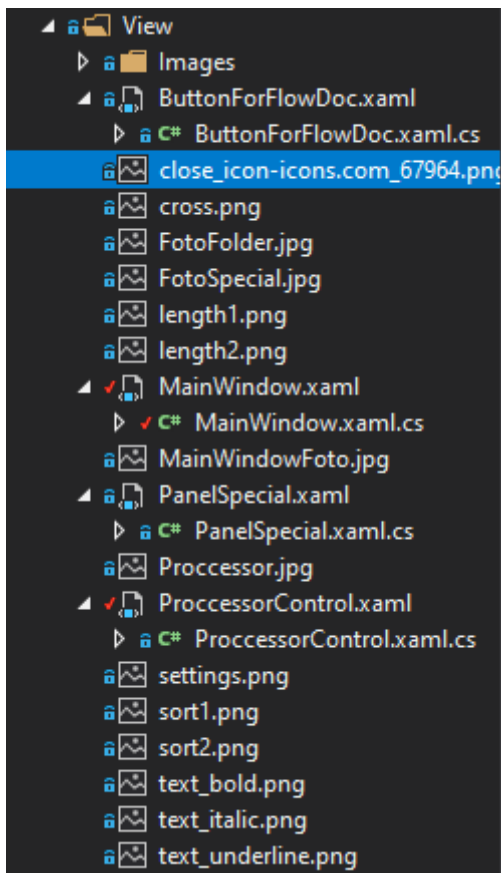
- Коллекция документов
- Выбранный документ

- Методы открытия, создания , сохранения файлов
- Событие изменения текущего файла

TextProcessorViewModel.cs:

- Выбранный документ
- Методы описанные в панели с иконкой процессора

View (Все файлы \*.xaml – разметка окон )



MainWindow.xaml.cs – главное окно

Содержит

- Обработчики событий нажатий
- Все классы ViewModel
- Все панели

ButtonForFlowDocument.xaml.cs- просто кнопка для переключения между документами

Содержит

- Обработчики событий
- Ссылку на файл

Содержит

- Обработчики событий

PanelSpecial.xaml.cs – верхняя панель для украшения текста

Содержит

- Обработчики событий

ProccessorControl.xaml.cs – верхняя панель для поиска, разделения и т.д. текста

Содержит

- Обработчики событий

### 3.3 Описание структуры проекта

Общий алгоритм запуска программы:

1. Запуск класса Application
2. Инициализация главного окна
3. Создание usercontrols элементов, панелей, кнопок
4. Создание FlowDocumetnViewModel и ProccessorViewModel
5. Создание во FlowDocumentViewModel коллекции документов
6. Привязка MainWindow к событиям изменения коллекции документов

Допустим пользователь совершил какое-то действие которое касается только отображения самого MainWindow. Тогда обработчики событий MainWindow срабатывают и изменяют отображение. Например: пользователь нажал кнопку отображения панели. Тогда обработчик скроет все другие панели этой группы и отобразит выбранное.

Допустим пользователь захотел открыть новый файл. Тогда MainWindow получает событие и вызывает метод у FlowDocumentViewModel для открытия файла. Метод FlowDocumentViewModel вызывает метод из

FileSystemDialogMethods, который, используя диалоговое окно, получает от юзера необходимый файл и возвращает его как документ. FlowDocumentViewModel, получив файл, добавляет его в коллекцию и делает его текущим в окне. При изменении коллекции, она отправляет событие, которое получает MainWindow и создает новую кнопку для файла. Когда меняется выбранный файл, MainWindow так же об этом узнает и записывает в строки добавленный файл.

ProcessorViewModel спроектирован точно так же, как и FlowDocumentViewModel. При вызове методов ProcessorViewModel создаются объекты Book или Text, которые обрабатывают текст. Результатом методов является текст, который они возвращают, MainWindow выводит в новом окне.

Коротко архитектуру можно описать след. порядком:

1. Пользователь совершил действие
2. View обработал это действие
3. Если необходимо вызвал методы подходящего класса ViewModel
4. ViewModel соответствующе изменил себя
5. После View подписанный на события ViewModel меняет отображение в соответствии с текущим состоянием ViewModel.

### **3.4. Выводы по главе.**

Текстовый редактор написан на C# с применением паттерна MVVM, имеет очень удобную архитектуру для дальнейшего расширения функционала. В этой главе были пояснены все основные элементы кода программы.

## **ЗАКЛЮЧЕНИЕ**

Целью работы являлось создание текстового редактора с возможностью форматирования текста.

Для разработки программы использовалась платформа WPF, язык программирования C# и паттерн MVVM. Данная связка имеет достаточную вычислительную производительность и предоставляет полный доступ к отладке. При разработке программы я получил практические навыки в работе с классами и закрепил теоретические знания по объектно-ориентированному программированию.

Можно сделать вывод, что данная цель была достигнута.

## СПИСОК ЛИТЕРАТУРЫ

Ведущая в мире платформа разработки программного обеспечения,  
<https://github.com>

Библиотека программиста. Обучающие материалы для разработчиков,  
<https://proglib.io>

Популярная система вопросов и ответов о программировании,  
<https://ru.stackoverflow.com>

Обучающий ресурс [metanit.com](https://metanit.com)