Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Операционный системы и системное программирование (ОСиСП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему:

Программное средство «Text Reader»

БГУИР КП 1-40 01 01 005 ПЗ

Студент: гр. 451003

Борикова Е. В.

Руководитель:

Зуев С. И.

Минск 2016

**СОДЕРЖАНИЕ**

Введение 4

1 Анализ предметной области 5

1.1 Анализ аналогов программного средства 5

2 Разработка алгоритма 8

2.1 Use Case диаграмма 8

3 Разработка программного средства 9

3.1 Написание кода 9

3.1.1 Вспомогательные переменные 9

3.1.2 Модуль "Glyph.h" 10

3.1.3 Модуль "Char.h" и "Char.cpp" 10

3.1.4 Модуль "Text\_Reader.h" 11

3.1.5 Модуль "Text\_Reader.cpp" 11

4 Разработка программного средства 9

3 Руководство пользователя 9

Заключение 24

Список использованных источников 25

Приложение А. Исходный код программы 26

**ВВЕДЕНИЕ**

Для работы с текстовыми документами существуют программы, которые называются текстовыми редакторами.

Текстовый редактор – это прикладная программа, позволяющая создавать текстовые документы, редактировать их, просматривать содержимое документа на экране, распечатывать документ. Примером простого текстового редактора является программа Блокнот.

Существуют текстовые редакторы, которые позволяют управлять не только содержанием документа, но и его внешним видом. Такие редакторы называются текстовыми процессорами. Текстовый процессор – это текстовый редактор с более широкими возможностями (форматирование текста, вставка графических объектов, проверка правописания и т.д.) Примером простого текстового процессора является программа Word.

Существует множество текстовых редакторов - от простейших учебных до мощных издательских систем, с помощью которых делают книги и т.п.

Наименьший элемент текста – символ. Слова – это символьные последовательности, отделяемые друг от друга пробелами или знаками препинания. Структурными единицами текста являются строка, абзац, страница, раздел. Абзац — несколько предложений, связанных по смыслу. Клавиша Enter в текстовых процессорах вставляет в текст символ конца абзаца.

Основной режим работы текстового редактора - это ввод и редактирование. При работе с текстовым редактором по экрану монитора перемещается курсор, который указывает текущую позицию для ввода. Символ, набираемый с клавиатуры помещается в позицию курсора. Под редактированием понимается внесение любых изменений в набранный текст.

Windows API ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) application programming interfaces) — общее наименование целого набора базовых функций [интерфейсов программирования приложений](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9) операционных систем семейств [Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows) корпорации «[Майкрософт](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B9%D0%BA%D1%80%D0%BE%D1%81%D0%BE%D1%84%D1%82)». Является самым прямым способом взаимодействия приложений с Windows.

Windows API спроектирован для использования в языке [Си](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)) для написания [прикладных программ](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), предназначенных для работы под управлением операционной системы MS Windows. Работа через Windows API — это наиболее близкий к операционной системе способ взаимодействия с ней из прикладных программ.  Windows API представляет собой множество функций, структур данных и числовых констант, следующих соглашениям языка Си. Все языки программирования, способные вызывать такие функции и оперировать такими типами данных в программах, исполняемых в среде Windows, могут пользоваться этим API.

1. Анализ предметной области
   1. Анализ аналогов программного средства

**«**Notepad++**»** — свободный текстовый редактор с открытым исходным кодом для Windows с подсветкой синтаксиса большого количества языков программирования и разметки. Поддерживает открытие более 100 форматов. Базируется на компоненте Scintilla, написан на C++ с использованием STL, а также Windows API и распространяется под лицензией GNU General Public License. Базовая функциональность программы может быть расширена как за счёт плагинов, так и сторонних модулей, таких как компиляторы и препроцессоры. [1]

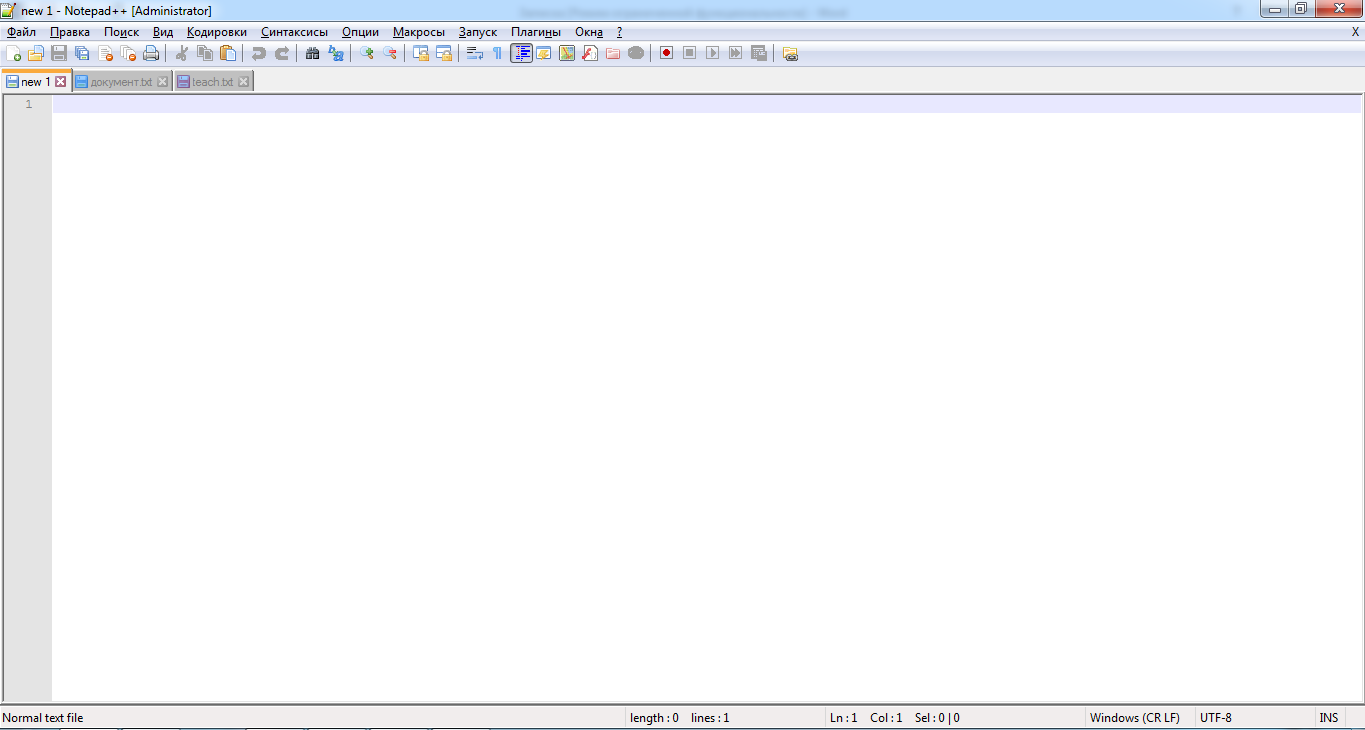


Рисунок 1.1 – Notepad ++

Пользователь имеет ряд возможностей таких как [1]:

* Подсветка синтаксиса
* Сворачивание кода
* Автодополнение и автоматическое закрытие скобок и тэгов (если активировано)
* Закладки
* Регулярные выражения для поиска и замены
* Запись и воспроизведение макросов
* Сравнение файлов
* Менеджер проектов
* Карта документа
* Переопределение любых горячих клавиш
* Резервное копирование сохраняемых файлов (включается в настройках)
* Трансформация текста при помощи подключённого плагина TextFX
* Поддержка и конвертирование кодировок ANSI, UTF-8 и UCS-2
* Блоковое выделение текста, одновременное выделение нескольких разных мест (с Ctrl)
* Мультистрочное редактирование (с использованием Alt)

**«**Sublime Text**»** — кроссплатформенный текстовый редактор. Поддерживает плагины на языке программирования Python. Программа часто используется как редактор исходного кода или HTML.

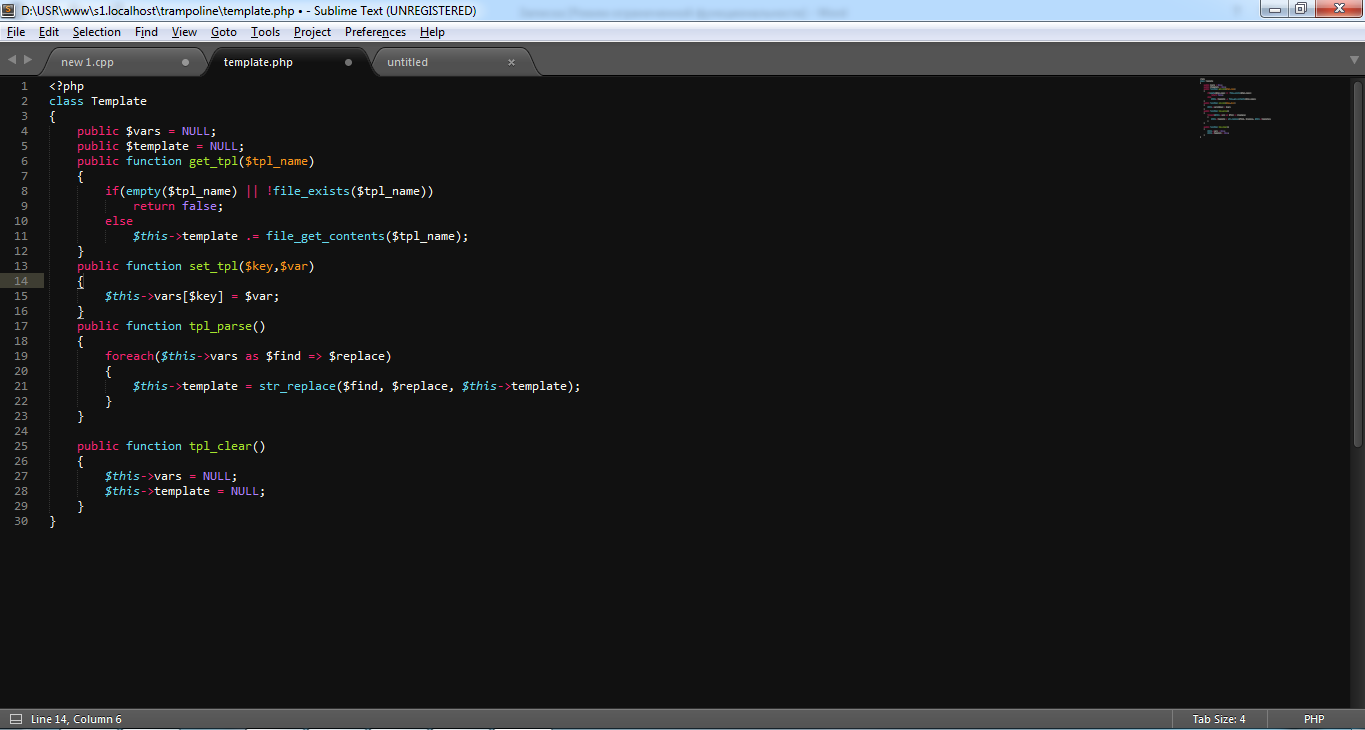


Рисунок 1.2 – Sublime Text

На данный момент уже доступна версия Sublime Text 3. Данный текстовый редактор является очень мощным текстовым редактором, в котором существует огромные дополнительные возможности.

* 1. Постановка задачи

Задачей данной курсовой работы является создание упрощенной версии программного средства текстовый редактор.

Реализовать следующие функции:

* Создание;
* Редактирование;
* Удаление;
* Загрузка из файла;
* Cохранение документа;
* Распечатать текстовый документ;

1. разработка алгоритма
   1. Use Case диаграмма

Представим Use Case диаграмму взаимодействия пользователя с программным средством (рис. 2.1.1). Данная диаграмма отображает весь пользовательский функционал программы.

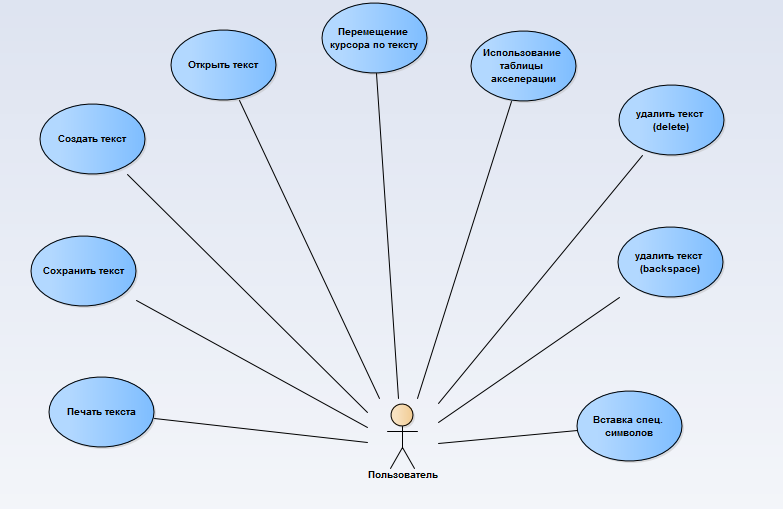


Рис. 2.1.1 – Use Case диаграмма

1. Разработка программного средства

При проектировании программы была выбрана методика нисходящего проектирования: сначала была проработана основная концепция программы (обработка нажатие клавиш, отображение символов).

В рамках данного курсового проекта планируется разработать программное средство.

Для разработки программного средства использовать язык программирования C++ Microsoft Visual Studio 2013. Среда для разработки – Microsoft Visual Studio 2013.

3.1 Написание кода

* + 1. Вспомогательные переменные

Программа использует глобальные переменные:

* Дескриптор главного окна (HWND).
* HINSTANCE hInst - дескриптор экземпляра окна.
* TCHAR szTitle– текст заголовка тулбара.
* TCHAR szWindowClass – имя главного окна.
* HIMAGELIST g\_hImageList – лист картинок, отображаемые на тулбаре.
* OPENFILENAME ofn – указатель на структуру, используемую для открытия файла.
* HFONT font – указатель на шрифт.
* RECT rect - структура содержит координаты верхнего левого и нижнего правого углов прямоугольника.
* list<Glyph\*> glyph\_list – указатель на список glyph.
* list<Glyph\*>::iterator CarriagePosition – указатель, который используется курсором, который перемещается перечислением glyph’ов из списка.
* SCROLLINFO si – указатель на структуру для колеса прокрутки.

Ниже приведены define переменные:

* #define IDC\_MYICON 2
* #define IDD\_TEXT\_READER\_DIALOG 102
* #define IDS\_APP\_TITLE 103
* #define IDD\_ABOUTBOX 103
* #define IDM\_ABOUT 104
* #define IDM\_EXIT 105
* #define IDI\_TEXT\_READER 107
* #define IDI\_SMALL 108
* #define IDC\_TEXT\_READER 109
* #define IDR\_MAINFRAME 128
* #define ID\_FILE\_NEW 32771
* #define ID\_FILE\_OPEN 32772
* #define ID\_FILE\_SAVE 32773
* #define ID\_COLOR 32777
* #define ID\_COLOR\_PEN 32778
* #define ID\_FONT 32779
* #define ID\_FONT\_FONT 32780
* #define ID\_ACCELERATOR32805 32805
* #define ID\_FILE\_PRINT 32806
* #define IDC\_STATIC -1
* #define MAX\_LOADSTRING 100
* #define CORRECT\_CARET\_POSITION 45
* #define WIDTH\_SCROOL 20
* #define SIZE 20
  + 1. Модуль «Glyph.h»

Модуль «Glyph.h» состоит из одного класса Glyph, который содержит открытые свойства и методы: деконструктор, получить высоту и ширину, метод рисовать, с помощью которого мы выводим символ на экран.

Данный класс нам необходим, чтобы создать список вводимых символов с клавиатуры, т.е. нам необходимо хранить где-то символы, которые мы ввели, для этого мы и будем использовать список типа Glyph.

Также для реализации передвижения курсора нам необходимо использовать list<Glyph\*>::iterator. Тип итератора для управляемой последовательности. Описывает тип объекта Glyph\*, который можно использовать как произвольно-доступный итератор для контролируемой последовательности.

* + 1. Модуль «Char.h» и «Char.cpp»

Модуль «Char.h» находится один класс Char, который наследуется от класса Glyph и дополняется скрытыми методами и свойствами: вычислить область экрана, высота, ширина, размер, состояние перевода на новую строку, значение символа, шрифт, цвет.

Реализация всех методов находится в файле «Char.cpp». В данном файле у нас находится конструктор для инициализации объекта Char:

Char::Char(int size, HFONT font, wchar\_t symbol, COLORREF rgb, bool state)

{

this->size = size;

this->font = font;

this->symbol = symbol;

this->rgb = rgb;

this->state = state;

computeRect();

}

Символ имеет тип данных wchar\_t. В Windows API тип wchar\_t именуется как WCHAR и имеет фиксированный размер 16 бит, что не позволяет кодировать весь набор символов Unicode (больше 1 миллиона). Поэтому нарушается стандарт ANSI/ISO C, который требует, чтобы символьный тип wchar\_t поддерживал все представимые в системе символы в одном объекте wchar\_t. По сути, в WinAPI под WCHAR подразумевается 2-байтное слово из кодировки UTF-16LE (как тип WORD), поэтому символы с кодами выше FFFF16кодируются парой WCHAR (так называемые «суррогаты») и всем API-функциям передаётся не количество символов, а размер символьного массива в машинных словах.[4] В общем, символ строки может быть представлен в виде 1-го байта и 2-х байтов.

Обычно однобайтовый символ это символ кодировки ANSI- в этой кодировке представлены все английские символы. А 2-х байтовый символ это кодировка UNICODE, в которой могут быть представлены все остальные языки в мире. В общем, символ строки может быть представлен в виде 1-го байта и 2-х байтов.

Обычно однобайтовый символ это символ кодировки ANSI- в этой кодировке представлены все английские символы. А 2-х байтовый символ это кодировка UNICODE, в которой могут быть представлены все остальные языки в мире. Для этого работе с файлом мы должны учитывать кодировку, т.е. при открытии файла необходимо указать с какой кодировкой будем работать:

\_wfopen\_s(&outFile, (LPWSTR)fullpath, L"r+,ccs=UTF-16LE");

Таким образом данное программное средство может поддерживать все языки мира и корректно отображать, сохранять и записывать любые символы.

* + 1. Модуль «Text\_Reader.h»

Данный модуль содержит объявление функций, которые выполняют основную ревлизацию в программе. В файле Text\_reader.cpp находится реализация этих функций.

* + 1. Модуль « Text\_Reader.cpp»

Данный модуль является основным модулем данной программы, который содержит обработку всех сообщений и реализацию основных функций программы.

Функция, приведенная ниже, является аналогом int main() в консольном приложении.

int APIENTRY \_tWinMain(\_In\_ HINSTANCE hInstance, \_In\_opt\_ HINSTANCE hPrevInstance,

\_In\_ LPTSTR lpCmdLine, \_In\_ int nCmdShow);

WinMain – название функции. Она имеет четыре параметра. Первый из них – HINSTANCE hInstance. Переменная hInstance является дескриптором экземпляра окна (это некий код оконной процедуры, идентификатор, по которой ОС будет отличать её от остальных окон). Через него можно обращаться к окну в процессе работы в других функциях, что-либо менять в параметрах окна.  HINSTANCE является одним из многочисленных типов данных определенных в WinAPI.  Дескриптор объекта можно описать так: HANDLE h.

В функции WinMain() вызывается функция CreateWindow(), в которой мы инициализируем структуру окна. Также в главной функции обрабатываются все сообщения посылаемые окну.

Функция GetMessage() обрабатывает сообщения, посылаемые ОС. Первый параметр – это адрес структуры MSG, в которую помещается очередное сообщение. Второй параметр — дескриптор окна. Третий и четвёртый параметры указывают порядок отбора сообщений. Цикл прекращается, если она получает сообщение WM\_QUIT. В таком случае она возвращает FALSE и мы выходим из программы. Функции TranslateMessage() и  DispatchMessage() в цикле нужны для интерпретации самих сообщений. Обычно это используется при обработке нажатых кнопок на клавиатуре

В данной прграмме обрабатываются сообщения такие как:

* WM\_CREATE;
* WM\_VSCROLL (
  + SB\_TOP;
  + SB\_BOTTOM;
  + SB\_LINEUP;
  + SB\_LINEDOWN;
  + SB\_THUMBPOSITION);
* WM\_COMMAND (ID\_FILE\_NEW, ID\_FILE\_OPEN, ID\_FILE\_SAVE, ID\_FILE\_PRINT, IDM\_ABOUT, IDM\_EXIT);
* WM\_SYSKEYDOWN;
* WM\_SYSCHAR;
* WM\_CHAR;
* WM\_KEYDOWN (
  + VK\_LEFT;
  + VK\_RIGHT;
  + VK\_RETURN;
  + VK\_UP;
  + VK\_DOWN;
  + VK\_LSHIFT;
  + VK\_RSHIFT;
  + VK\_DELETE;
  + VK\_BACK;
  + VK\_TAB);
* WM\_SIZING;
* SW\_MAXIMIZE;
* WM\_PAINT;
* WM\_SETFOCUS;
* WM\_KILLFOCUS;
* WM\_DESTROY;

Функция WndProc() будет получать сообщения, которые Windows будет отсылать окну:

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

В сообщении WM\_PAINT вызывается функция, которая печатает наши символы.

case WM\_PAINT:

print\_glyphs(hWnd);

break;

Функция print\_glyphs(hWnd) принимает параметр - дескриптор экрана, на котором будут выводится наши символы. Данная функция высчитывает ширину экрана, и если ширина экрана больше чем строка символов, то переносит позицию на новую строку. Что очень удобно, когда мы меняем размер окна, текст сам редактируется и нет необходимости в горизонтальном колесе прокрутке.

За перемещение курсора вверх отвечает функция: void key\_up(HWND hWnd). В данной функции присутствуют различного рода проверки. Приведем схему алгоритма данной функции (рис. 3.1.5.1).

Функция key\_down(HWND hWnd) действует по похожему алгоритму.

Функция delete\_glyph(HWND hWnd) вызывается при нажатии кнопки «Delete» и удаляет символ стоящий после курсора. По схожему принципу действует функция backspace(HWND hWnd), только удаляет символ, который стоит после курсора и вызывается при нажатии клавиши «Backspace».

При создании окна вызывается функции, с помощью которой мы устанавливаем курсор в начальную позицию. Код данной функции приведен ниже:

void ToCreateCaret(HWND hWnd)

{

CreateCaret(hWnd, (HBITMAP)0, 2, SIZE + 2);

SetCaretPos(0, CORRECT\_CARET\_POSITION);

ShowCaret(hWnd);

}

При изменении размера экрана, необходимо перерисовать окно и текст в нем. Для этого вызывается функция, в которой мы заново высчитываем высоту и ширину окна. Код функции приведен ниже:

void repaintWindow(HWND hWnd)

{

GetClientRect(hWnd, &rect);

heightWindow = rect.bottom - rect.top;

widthWindow = rect.right - rect.left;

print\_glyphs(hWnd);

}

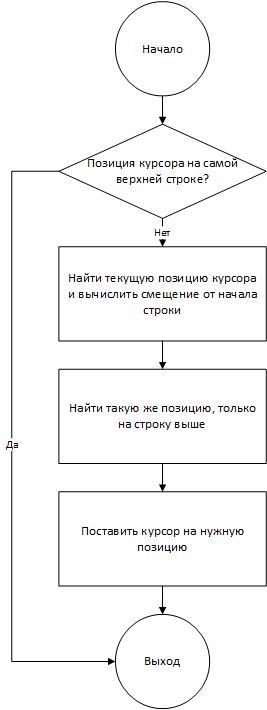


Рис. 3.1.5.1 – схема алгоритма функции key\_up(HWND hWnd)

Данное программное средство обеспечивает работу с файлами следующими функциями:

* Save(HWND hWnd);
* Open(HWND hWnd);

В данных функциях для получения имени файла используется структура OPENFILENAME ofn с помощью которой можно получить путь к файлу. Поля данной структуры приведены на рис. 3.1.5.2.

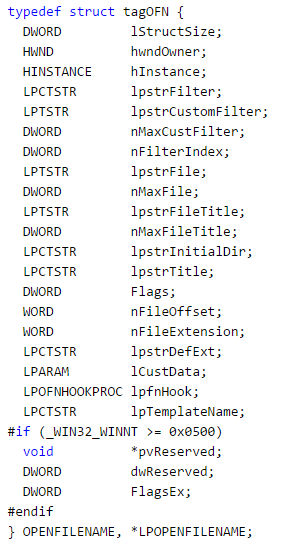


Рис. 3.1.5.2 – структура OPENFILE

После того как получили имя файла, происходит сохранения в файл нашего текста. Приведем схему алгоритма функции Save (рис. 3.1.5.3).



Рис. 3.5.1.3 - схему алгоритма функции Save

Чтобы распечатать наш документ, необходимо вызвать функцию:

Print(HWND h, int x1, int x2, int y1, int y2),

в которую передаем координаты нашей рабочей области.

В этой функции используется структура PRINTDLG для инициализации стандартного окна Печать (Print). После того, как пользователь закрывает блок диалога, система возвращает информацию об определяемом пользователем выборе печати в этой структуре. Структура имеет вид:

typedef struct tagPD

{

DWORD lStructSize;

HWND hwndOwner;

HANDLE hDevMode;

HANDLE hDevNames;

HDC hDC;

DWORD Flags;

WORD nFromPage;

WORD nToPage;

WORD nMinPage;

WORD nMaxPage;

WORD nCopies;

HINSTANCE hInstance;

DWORD lCustData;

LPPRINTHOOKPROC lpfnPrintHook;

LPSETUPHOOKPROC lpfnSetupHook;

LPCTSTR lpPrintTemplateName;

LPCTSTR lpSetupTemplateName;

HANDLE hPrintTemplate;

HANDLE hSetupTemplate;

} PRINTDLG;

Функция int GetDeviceCaps( \_In\_ HDC hdc, \_In\_ int nIndex)

извлекает данные об устройствах для указанного устройства.

1. Разработка программного средства

В качестве основы для проведённого функционального тестирования, были взяты функциональные требования, описывающие необходимые задачи, которые решает ПО. Результаты приведены в таблице 4.1.

Таблица 4.1 – Тестовые сценарии

|  |  |  |  |
| --- | --- | --- | --- |
| **Функциональность** | **Выполняемые действия** | **Ожидаемый результат** | **Результат теста** |
| Отображение набранного текста | Ввод текста с клавиатуры | Вывод текста на экран | Пройден |
| Переход на новую строку | Нажать на клавиатуре «Enter» | Перевод курсора на новую строку | Пройден |
| Вставка вертикальной табуляции | Нажать на клавиатуре «Tab» | Вставка вертикальной табуляции | Пройден |
| Вставка спец. символов | Ввод с клавиатуры символов: ! « № ; % : ? \* ( ) \_ + @ # $ ^ & {} [] ,. < > / \ | “ ` ~ | Отображение в рабочей области: ! « № ; % : ? \* ( ) \_ + @ # $ ^ & {} [] ,. < > / \ | “ ` ~ | Пройден |
| Отображение символов на арабском языке | Ввод символов с клавиатуры на арабском языке: ابينيبمسي ميستبىءزؤ مسيتب | Отображение символов в рабочей области на арабском языке: ابينيبمسي ميستبىءزؤ مسيتب | Пройден |
| Напечатать текстовый документ | Нажатие на панели инструментов кнопки «Print» или через контекстное меню File- > Print | Открытие другой программы, которая обрабатывает текстовый документ, который мы передели | Пройден |
| Создание нового документа | Нажатие комбинации клавиш Ctrl + N, или через контекстное меню File->New, или на панели инструментов нажать кнопку «New» | Отображение листа символов, который предварительно очистился | Пройден |
| Открытие текстового документа | Нажатие комбинации клавиш Ctrl + O, или через контекстное меню File->Open, или на панели инструментов нажать кнопку «Open» | Отображение текстового документа | Пройден |
| Продолжение таблицы 4.1 – Тестовые сценарии | | | |
| Сохранение текстового документа | Нажатие комбинации клавиш Ctrl + S, или через контекстное меню File->Save, или на панели инструментов нажать кнопку «Save» | Сохранение текстового документа | Пройден |
| Сохранение и отображение текстового документа только на английском языке | Нажатие комбинации клавиш Ctrl + S, или через контекстное меню File->Save, или на панели инструментов нажать кнопку «Save» | Сохранение и отображение текстового документа | Пройден |
| Сохранение и отображение текстового документа только на русском языке | Нажатие комбинации клавиш Ctrl + S, или через контекстное меню File->Save, или на панели инструментов нажать кнопку «Save» | Сохранение и отображение текстового документа | Пройден |
| Сохранить текстовый документ в уже существующий файл | Нажатие комбинации клавиш Ctrl + S, или через контекстное меню File->Save, или на панели инструментов нажать кнопку «Save». Далее указать имя файла, который уже существует | Замена и сохранение документа | Пройден |
| Просмотр информации о программе | Нажатие на панели инструментов кнопки «About» или через контекстное меню File- > About | Появление нового окна с информацией об авторе, год, имя программы | Пройден |
| Передвижение курсора влево | Нажатие на клавиатуре стрелочки влево | Перемещение курсора | Пройден |
| Позиция курсора не изменяется влево, если дошли до начала листа символов | Нажатие на клавиатуре стрелочки влево в начале листа символов | Позиция курсора не изменяется | Пройден |
| Передвижение курсора вправо | Нажатие на клавиатуре стрелочки вправо | Перемещение курсора | Пройден |
| Позиция курсора не изменяется вправо, если дошли до конца листа символов | Нажатие на клавиатуре стрелочки вправо в конце листа символов | Позиция курсора не изменяется | Пройден |
| Передвижение курсора вверх | Нажатие на клавиатуре стрелочки вверх | Перемещение курсора | Пройден |
| Продолжение таблицы 4.1 – Тестовые сценарии | | | |
| Позиция курсора не изменяется вверх, если курсор на самом верху | Нажатие на клавиатуре стрелочки вверх, когда находимся на самой первой строке | Позиция курсора не изменяется | Пройден |
| Передвижение курсора вниз | Нажатие на клавиатуре стрелочки вниз | Перемещение курсора | Пройден |
| Позиция курсора не изменяется вниз, если курсора находится на последнее позиции | Нажатие на клавиатуре стрелочки вниз, когда находимся на последней позиции | Позиция курсора не изменяется | Пройден |
| Удаление символов | Нажатие клавиши на клавиатуре «Delete» | Удаление символа, стоящего после курсора, позиция курсора не изменяется | Пройден |
| Удаление символов, которых не существует | Нажатие клавиши на клавиатуре «Delete» в позиции, где после курсора не следует символов | Обработка данной исключительной ситуации. Исключения не прервали работу | Пройден |
| Удаление символов | Нажатие клавиши на клавиатуре «Backspace» | Удаление символа, стоящего перед курсором, позиция курсора смещается влево на одну позицию | Пройден |
| Удаление символов, которых не существует | Нажатие клавиши на клавиатуре «Backspace» в позиции где перед курсором не следует символов | Обработка данной исключительной ситуации. Исключения не прервали работу | Пройден |

1. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Чтобы установить данное программное средство необходимо скопировать себе на компьютер (ноутбук) .exe файл. Кликните двойным щелчком на ярлык с надписью «Text Reader» (рис. 5.1).

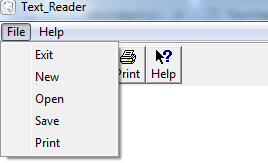
Рисунок 5.1 – Ярлык программы

Программа запустится на вашем компьютере. Запустив приложение, пользователь увидит главное окно (рис 5.2).

Рисунок 5.2 – Главное окно программы

После открытия окна в рабочей области уже можно увидеть курсор. Для создания текста необходимо начать вводить текст с помощью клавиатуры. Текст сразу будет отображаться на экране в рабочей области.

Если нам необходимо сохранить наш текст, то это можно сделать нажатием комбинации клавиш Ctrl + S, нажатием кнопки «Save» на панели инструментов (рис. 5.5) или через контекстное меню: File-> Save. (рис. 5.3 Далее открывается окно для задания местоположения и имени файла. Для подтверждения выбора файла необходимо нажать кнопку «ОК». Выполняется сохранение текста в файл. Если ввести в диалоговое окно «Save file as…» (рис. 5.3) несуществующий файл, то программное средство создаст этот файл, что очень удобно и позволяет сохранить немного времени.



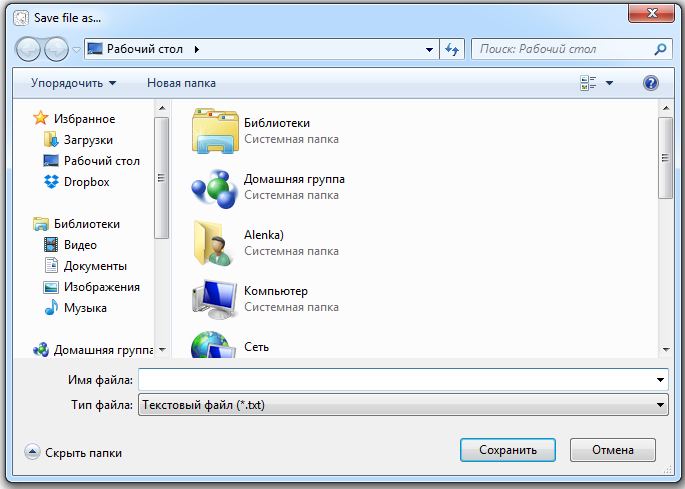
Рис. 5.3 – Контекстное меню

Рис. 5.4 – Окно «Save file as…»

Если нам необходимо создать новый текстовый документ, то это можно сделать нажатием комбинации клавиш Ctrl + N, нажатием кнопки «New» на панели инструментов (рис 5.5) или через контекстное меню: File-> New(рис 5.3). Рабочая область и список символов очищается, курсор перемещается в верхний левый угол рабочей области.



Рис.5.5– Панель инструментов

Для редактирования текста реализованы удобные и привычные нам функции. Например, для удаления символа необходимо нажать клавишу «Backspace» или «Delete». Чтобы переместить курсор в нужную позицию необходимо использовать клавиши стрелка вниз, вверх, вправо и влево.

Если нам необходимо распечатать наш текстовый документ, то это можно сделать с помощью кнопки «Print» на панели инструментов (рис. 5.5) или через контекстное меню File-> Print (рис. 5.3). Далее выбираем программу, через которую хотим распечатать текстовый документ (рис. 5.6), и работаем уже с ней.

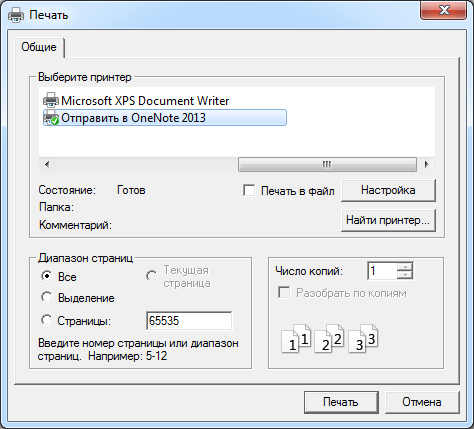


Рис. 5.6 – Вспомогательная программа для печати документа

Также программное средство позволяет нам открывать и загружать файлы. Это можно сделать с помощью кнопки «Open» на панели инструментов(рис. 5.5), через контекстное меню File-> Open (рис. 5.3) или комбинацией клавиш Ctrl + O. Откроется окно «Open» (рис. 5.6) в котором мы должны указать имя файла, который хотим загрузить.

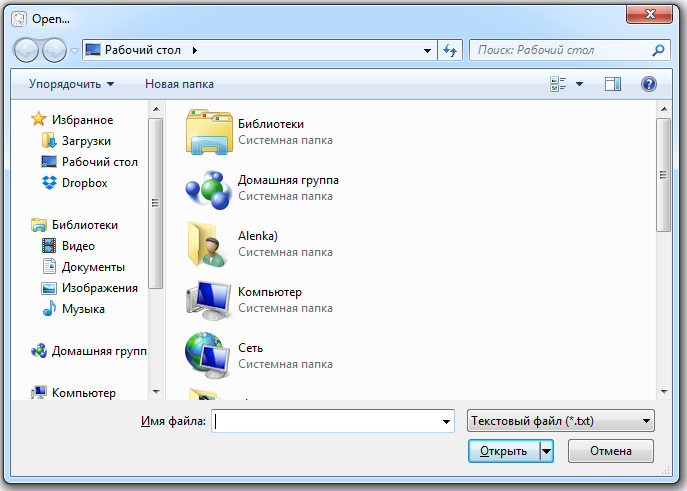


Рис. 5.6 – Окно «Open»

Программное средство предоставляет возможность узнать информацию об авторе, иконку, год издания и версию программы. Это можно сделать с помощью панели инструментов нажатием кнопки «About» или через контекстное меню Help-> About (рис. 5.7). Откроется диалоговое окно (рис. 5.8).

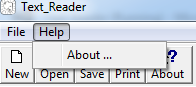


Рис. 5.7 – Вкладка в меню Help

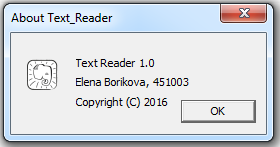


Рис. 5.8 – диалоговое окно «About»

Чтобы выйти из программы «Text Reader» необходимо нажать на крестик в меню программы (рис. 5.8), также можно нажатием комбинации клавиш Ctrl + Q или через контекстное меню программы (рис. 5.3)

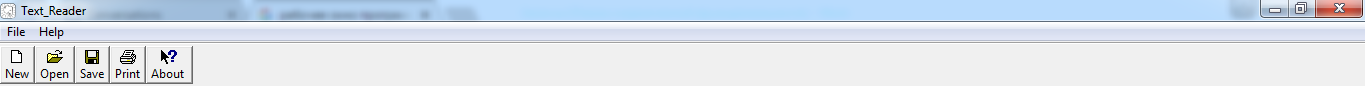


Рис. 5.8 – Меню программы

Данное приложение очень быстро, просто и удобно в использовании. Это программное средство подойдет любому пользователю для различных целей.

**ЗАКЛЮЧЕНИЕ**

В современном мире становится все больше и больше информации, которую необходимо запомнить, но к сожалению, мы не всегда способны все помнить и часто забывает полезную информацию. Часто нам необходимо быстро записать информацию, сохранить ее, отредактировать, открыть. Целью данной курсовой работы – разработать программное средство, которое позволит создавать, редактировать, сохранять и считывать текст.

Результатом данной курсовой работы является программное средство «Text Reader», созданное в интегрированной среде разработки Microsoft Visual Studio 2013 на языке с++. Приложение опирается на платформу WinAPI32.

В ходе разработки было изучено большое количество литературы из различных источников. Были усовершенствованы знания о работе с файлами. Получены новый знания связанные с обработкой клавиш, а также с работой курсора. Полезным было получить навык с работой с различными кодировками.

Возможно дальнейшее усовершенствование проекта:

* Добавление колеса прокрутки;
* Добавление функции «Отмена шага»;
* Обработать сообщения поступающие от мышки;
* Подсветка синтаксиса;

Достоинства программного средства:

* Прост в использовании;
* Существуют команды быстрого доступа;
* Обрабатывает любые символы на любых языках;
* Программа занимает мало места;

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Текстовые редакторы [Электронный ресурс]. – Текстовый редактор. Текстовый процессор. Режим доступа: <http://school497.ru/download/u/02/les11/les.html>. Дата доступа: 12.12.2016
2. Википедия Свободная энциклопедия [Электронный ресурс]. - Notepad++. Режим доступа: https://ru.wikipedia.org/wiki/Notepad%2B%2B. Дата доступа: 12.12.2016.
3. Википедия Свободная энциклопедия [Электронный ресурс]. - Sublime Text. Режим доступа: - <https://ru.wikipedia.org/wiki/Sublime_Text>. Дата доступа: 12.12.2016.
4. Википедия Свободная энциклопедия [Электронный ресурс]. – Широкий символ, wchar\_t. Режим доступа: - <https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%80%D0%BE%D0%BA%D0%B8%D0%B9_%D1%81%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB>. Дата доступа: 12.12.2016.
5. Макс Шлее, Профессиональное программирование на С++ – М.: БХВ-Петербург, 2010. – 883 с.
6. Федоренко Ю.П., Алгоритмы и программы на C++ Builder –М.: ДМК Пресс,2010. –544 с.
7. Мартин.Р., Чистый код. Создание, анализ и рефакторинг: Петербург,2015. –464 с
8. Джеффри Рихтер, Windows для профессионалов: создание эффективных Win32\_приложений с учетом специфики 64 разрядной версии Windows / Пер. с англ. — 4-е изд. — Спб.: Питер; М.: Издательство «Русская Редакция»; 2008. — 720 стр.: ил.

Приложение А. Исходный код программы

Main.cpp

Map.h