МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»
(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова»)

БГТУ.СМК-Ф-4.2-К5-01

Факультет О Информационные и управляющие системы
шифр наименование

Кафедра О7 Информационные системы и программная инженерия
шифр наименование

Дисциплина Программирование на языке высокого уровня

КУРСОВАЯ РАБОТА
на тему
Объектно-ориентированная разработка программ с графическим пользовательским интерфейсом «сверху-вниз»

Выполнил студент группы И506Б
Еремишин Н.Н.
Фамилия И.О.

РУКОВОДИТЕЛЬ
Матвеев Т.А. _____
Фамилия И.О. Подпись

Оценка _____

«_____» _____ 2022 г.

САНКТ-ПЕТЕРБУРГ
2021 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель работы – научиться разрабатывать приложения с графическим интерфейсом на языке c# [1] или Windows Forms [2].

В качестве темы курсовой работы была выбрана разработка интернет-браузера с системой закладок и историей посещений web страниц.

Были поставлены следующие задачи: выбрать движок, который отображает web страницу, реализовать отображение web страницы, написать класс, который будет отвечать за сохранение закладок, создать класс, который будет отвечать за историю посещений.

Отчёт содержит следующие разделы: описание предметной области программы – общее описание разработанной программы и её возможности, описание функциональной модели, то есть функций программы, а также взаимодействие пользователя с ней, описание классов – разбор всех полей и методов каждого из них. Отдельный раздел посвящён результатам тестирования программы.

1 Предметная область

Браузер – программа, предназначенная для просмотра сайтов. В браузере присутствуют закладки – сохранённые ссылки для более быстрого доступа к сайту, а также история просмотра страниц.

Для разработки браузера первым делом нужно определиться с движком, который будет отображать web страницу. В c# windows forms по умолчанию встроен движок Internet Explorer [3], поддержку которого прекратили, из-за чего он не может корректно отображать современные сайты.

Далее, в пакетном менеджере была обнаружена библиотека для работы с движком Chromium - Awesomium, разработчик которой тоже прекратил поддержку. Данная библиотека была протестирована и в конечном итоге оказалась неработоспособной.

В конце, была найдена библиотека CefSharp, которая тоже предоставляет возможность работать с движком Chromium. Это библиотека и была выбрана, так как она поддерживается разработчиком и корректно функционирует.

2 Функциональная модель

Исходя из определения браузера, программа должна уметь открывать сайты по ссылке. Для удобства, программа должна уметь работать с вкладками - позволять пользователю открывать, закрывать вкладки, а также переключаться между ними. Также в браузере должно быть реализовано сохранение и удаление закладок, а также их открытие по нажатию. Для истории просмотра должно быть реализовано автоматическое сохранение посещённых страниц, а также удаление.

На рисунке 1 изображена модель вариантов использования программы

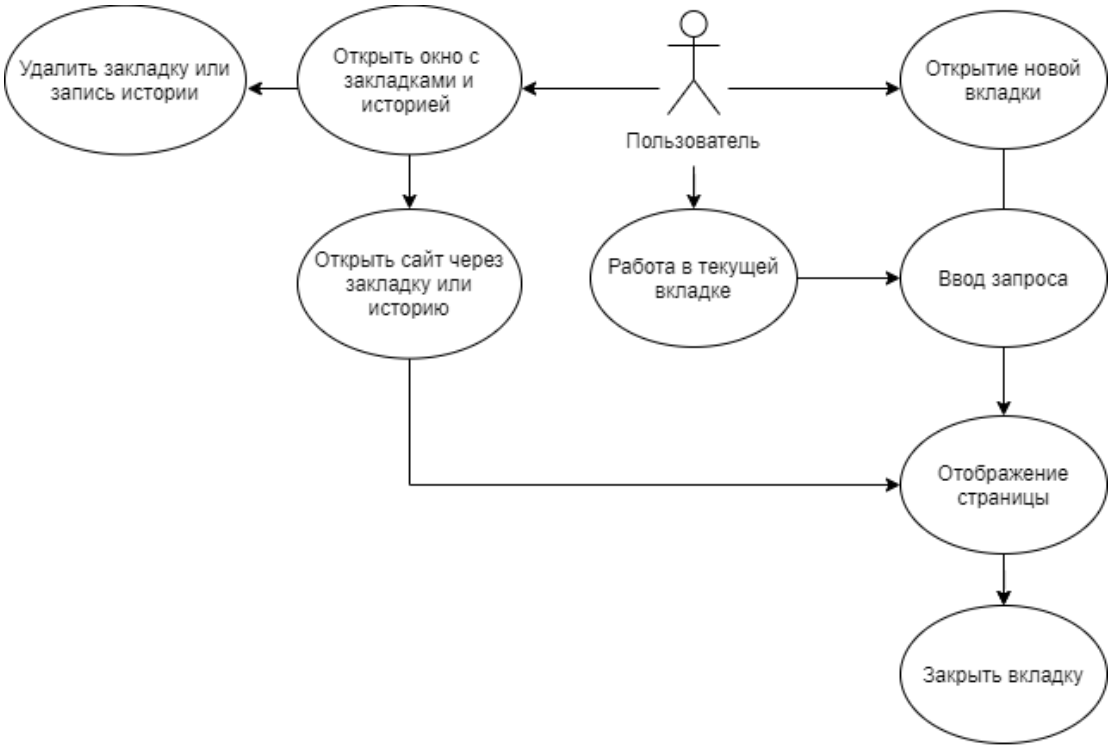

Рисунок 1 – Модель вариантов использования

При запуске, автоматически открывается поисковая система Google. В интерфейсе браузера присутствуют следующие кнопки: добавить закладку, просмотреть историю и закладки, открыть и закрыть вкладку.

Пользователь может видеть “Умную” адресную сроку, которая сама определит, ввёл ли пользователь ссылку на сайт, или же просто запрос. Поисковая строка, во втором случае, сама конвертирует запрос в ссылку и найдёт его в поисковой системе. Строка конвертирует запрос, когда пользователь нажимает кнопку поиска.

Открыв окно с закладками и историей, пользователь видит два столбца: слева – закладки, справа – история. Каждый элемент из каждого столбца можно удалить, нажав на соответствующую кнопку, или же перейти на сайт, нажав на него.

3 Разработанные классы

Для реализации задуманного приложения было необходимо описать несколько классов. Подробное описание этих классов представлено ниже.

Класс BookmarksAndHistory – класс, описывающий одну закладку или одну запись в истории. Его диаграмма представлена на рисунке 2.

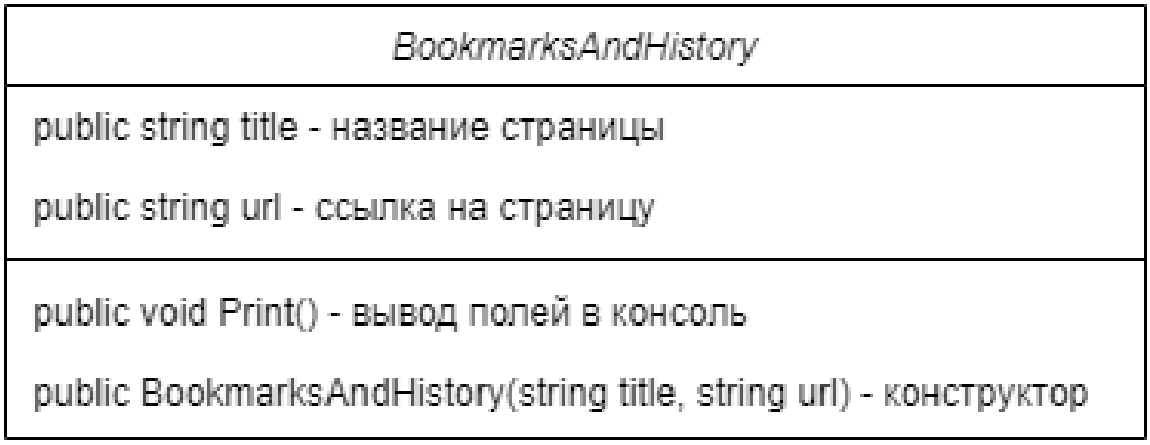

Рисунок 2 – Класс BookmarksAndHistory

Как можно заметить – один класс представляет запись в закладках и запись истории одновременно. Это потому, что две эти записи практически идентичны.

Также можно видеть, что в классе нет методов, которые реализуют работу с файлом и интерфейсом. Эти методы были описаны вне класса:

List<BookmarksAndHistory> GetListFromFile(string fileName) – данный метод возвращает все записи из указанного файла. Всего два файла – с историей и закладками. Принимает параметр – имя файла.

void CreateNewListRecord(string title, string url, List<BookmarksAndHistory> list, string fileName, Action AddToLayout, bool addFirst = false) – добавляет запись в список, а также в файл. Принимает следующие параметры: string title – название страницы, string url – ссылка на страницу, List<BookmarksAndHistory> list – список, в который необходимо добавить запись, string filename – имя в файла, в который необходимо добавить запись, Action AddToLayout – функция, которая выводит список в пользовательский интерфейс, bool addFirst – true, если необходимо добавить

в начало списка, false – в конец.

void RemoveRecordFromTheList(string title, string url, List<BookmarksAndHistory> list, string fileName, Action AddToLayout) – удаляет запись из списка и файла. Принимает следующие параметры: string title – название страницы, string url – ссылка на страницу, List<BookmarksAndHistory> list – список, из которого необходимо удалить запись, string filename – имя в файла, из которого необходимо удалить запись, Action AddToLayout – функция, которая выводит список в пользовательский интерфейс.

bool ListIncludesRecord(BookmarksAndHistory bkmk, List<BookmarksAndHistory> list) – проверяет, есть ли уже такая запись в указанном списке. Принимает следующие параметры: string title – название страницы, string url – ссылка на страницу, List<BookmarksAndHistory> list – проверяемый список.

void AddBookmarksToFlowLayout() – функция, добавляющая все закладки из списка в пользовательский интерфейс, создавая панели с названием сайта и кнопкой удалить закладку.

void AddHistoryToFlowLayout() – функция, добавляющая все элементы истории из списка в пользовательский интерфейс, создавая панели с названием сайта и кнопкой удалить элемент.

Класс UrlUtils – статический класс [4], содержащий в себе реализацию умной поисковой строки. Его диаграмма представлена на рисунке 3.

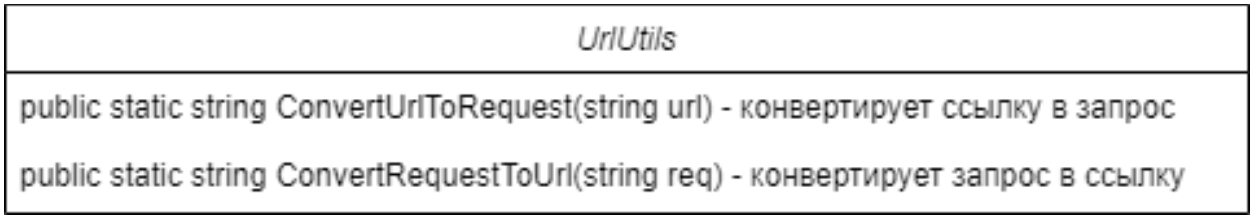

Рисунок 3 – Класс UrlUtils

Класс содержит два метода:

public static string ConvertUrlToRequest(string url) – конвертирует запрос, заданный ссылкой в запрос, заданный словами. Принимает запрос,

заданный ссылкой. Пример работы метода: ссылка https://www.google.ru/search?q=amazon+delivery конвертируется в “amazon delivery”

public static string ConvertRequestToUrl(string req) – конвертирует запрос, заданный словами в запрос, заданный ссылкой. Принимает запрос, заданный словами. Пример работы метода: “amazon delivery” конвертируется в https://www.google.ru/search?q=amazon+delivery

Класс BrowserSettings – класс, содержащий в себе настройки браузера. Его диаграмма представлена на рисунке 4.

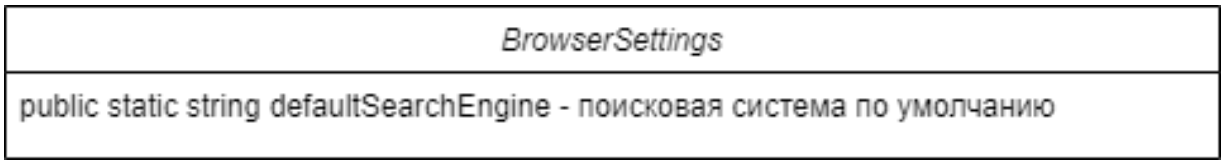

Рисунок 4 – Класс BrowserSettings

В нём только одно поле:

public static string defaultSearchEngine - поисковая система по умолчанию

Класс LifespanHandler – класс, который реализует открытие ссылок в новой вкладке, а не в новом окне. Его диаграмма представлена на рисунке 5.

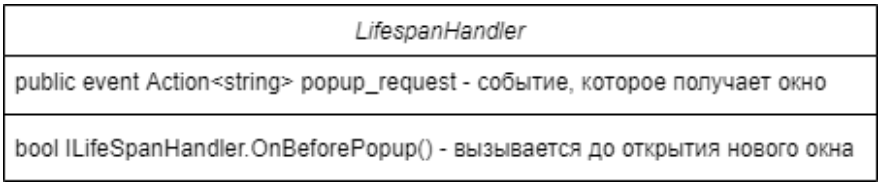

Рисунок 5 – Класс BrowserSettings

Данный класс реализует интерфейс ILifeSpanHandler [5].

Класс содержит в себе поле public event Action<string> popup_request – событие, которое получает всплывающее окно с URL-адресом.

Также класс содержит в себе метод для реализации интерфейса - bool ILifeSpanHandler.OnBeforePopup – он останавливает открытие нового окна и вместо этого открывает ссылку в новой вкладке.

Общая диаграмма всех классов представлена на рисунке 4:

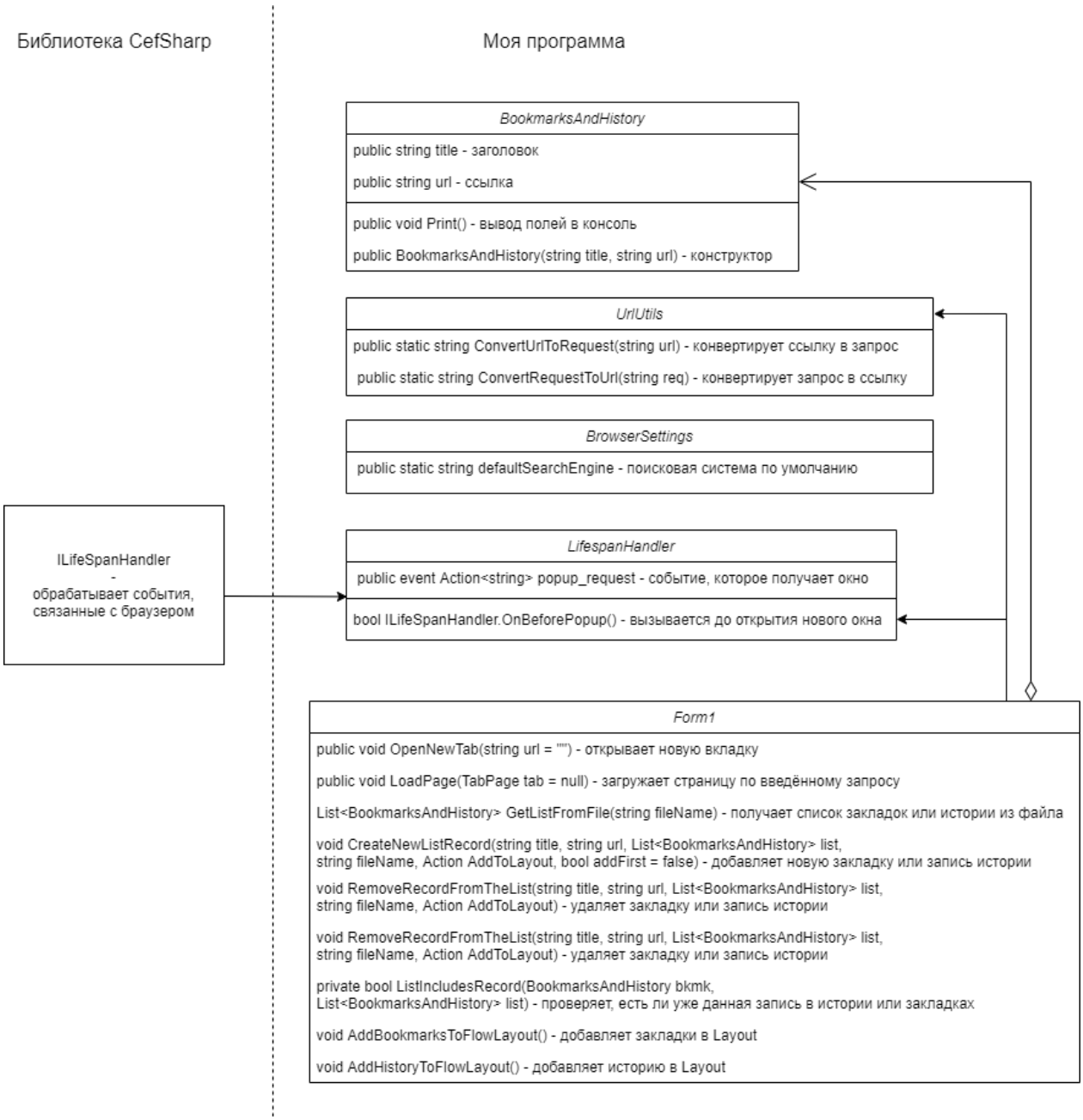

Рисунок 6 – Общая диаграмма классов

4 Структура программы

Программа состоит из следующих файлов:

- BookmarksManager.cs – содержит реализацию сохранения закладок и истории в файл, чтения и удаления из этого файла (общие методы для закладок и истории); содержит метод для вывода закладок на экран;

- HistoryManager.cs – содержит метод для вывода истории на экран;

- BrowserUtils.cs – содержит два класса – BrowserSettings, в котором можно выбрать поисковую систему по умолчанию и UrlUtils – класс, реализующий “умную” адресную строку;

- LifeSpanHandler.cs – данный файл содержит в себе класс, который позволяет открывать ссылки в новой вкладке, а не окне;

- Form1.cs – класс с формой, содержащий в себе все методы взаимодействия с пользовательским интерфейсом, например, открытие новой вкладки;

5 Внешние файлы

Для хранения информации используется два бинарных файла «bookmarks.bin» и «history.bin». В данных файлах хранятся сохранённые пользователем закладки и история просмотра страниц.

6 Тестирование программы

Для того, чтобы удостовериться в корректности работы программы, были проведены следующие тесты: отображение страницы, открытие новой вкладки, отображение истории и закладок, удаление, открытие и создание элементов истории и закладок, работу умной поисковой строки.

6.1 Отображение web страницы.

Перейдя по ссылке, пользователь может видеть соответствующую страницу. Пример приведен на рисунке 7.

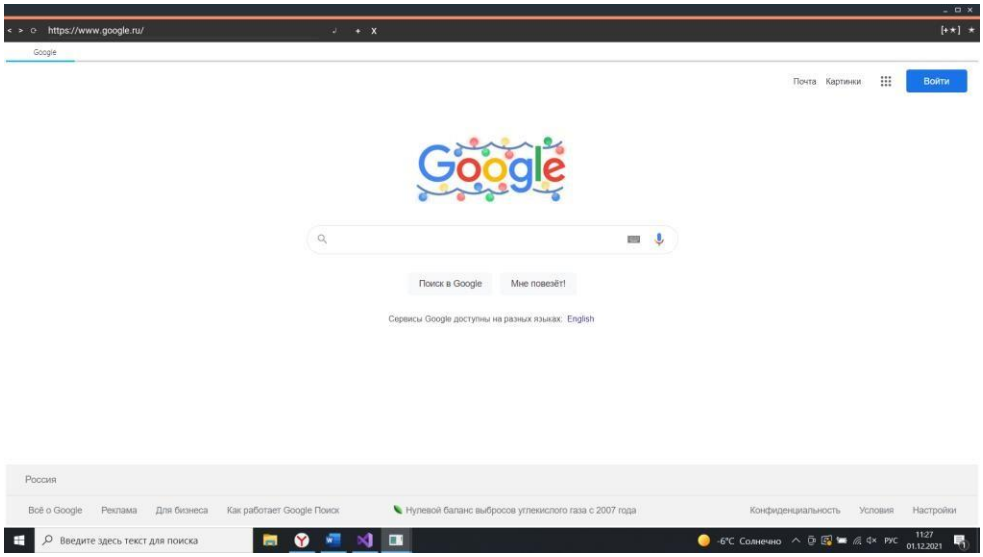

Рисунок 7 – Отображение web страницы

6.2 Открытие новой вкладки.

Для того, чтобы открыть новую вкладку, надо нажать кнопку справа от адресной строки. Пример приведен на рисунке 8.

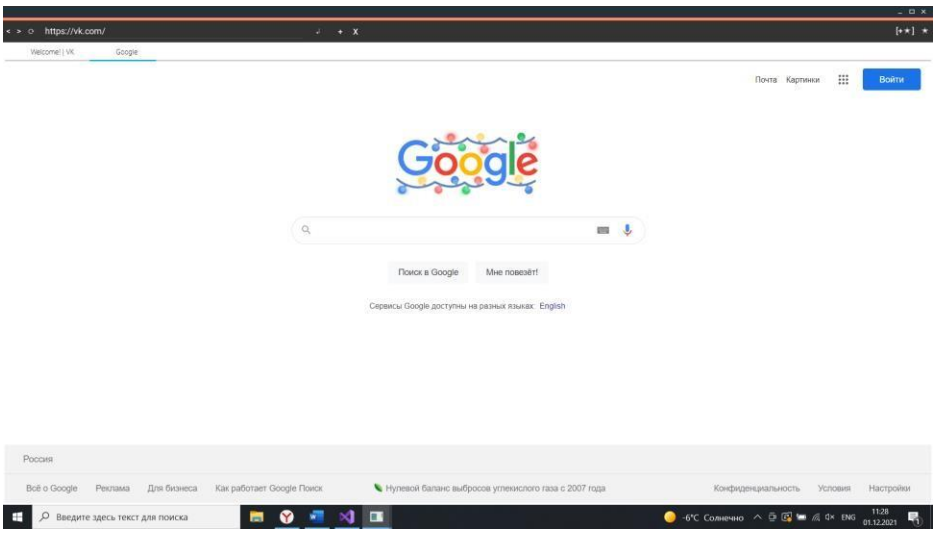

Рисунок 8 – Открытие новой вкладки

6.3 Отображение истории.

История расположена в правом столбце. Корректное изображение истории можно наблюдать на рисунке 9.

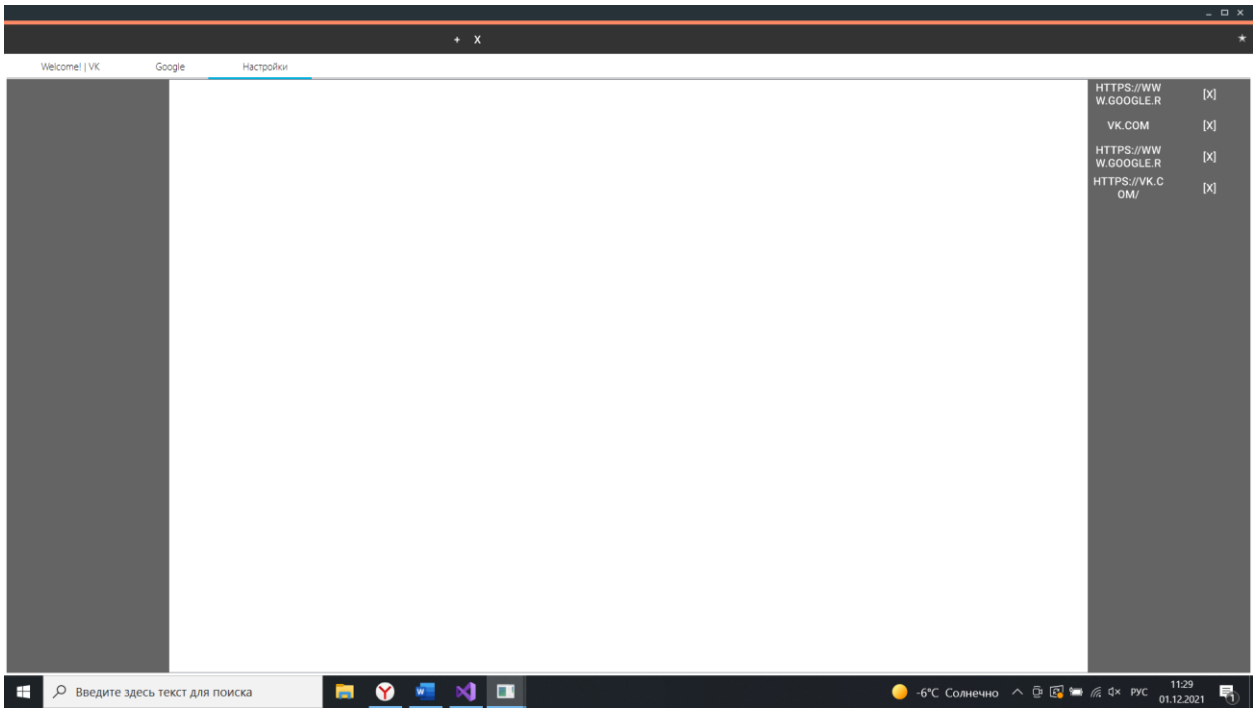

Рисунок 9 – Отображение истории (справа)

6.4 Добавление и удаление закладок и элементов истории.

Для удаления элементов необходимо нажать на крестик. Пример приведен на рисунках 10 и 11.

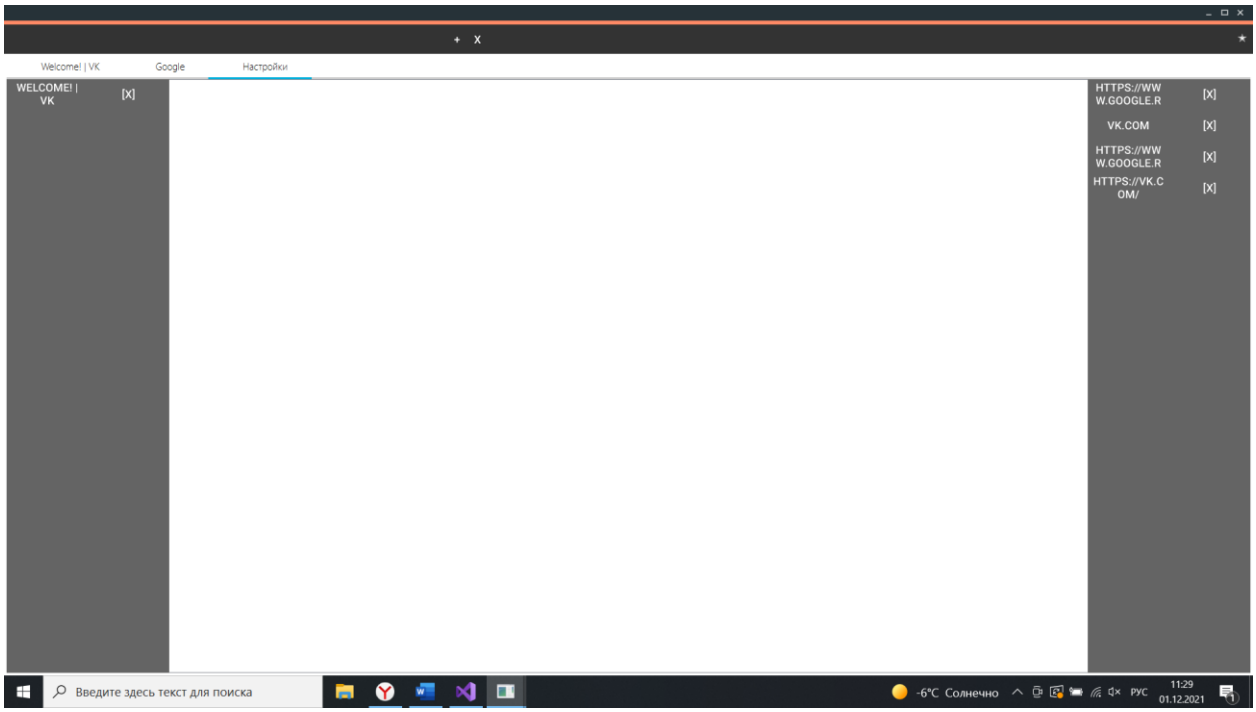

Рисунок 10 – Добавлена новая закладка

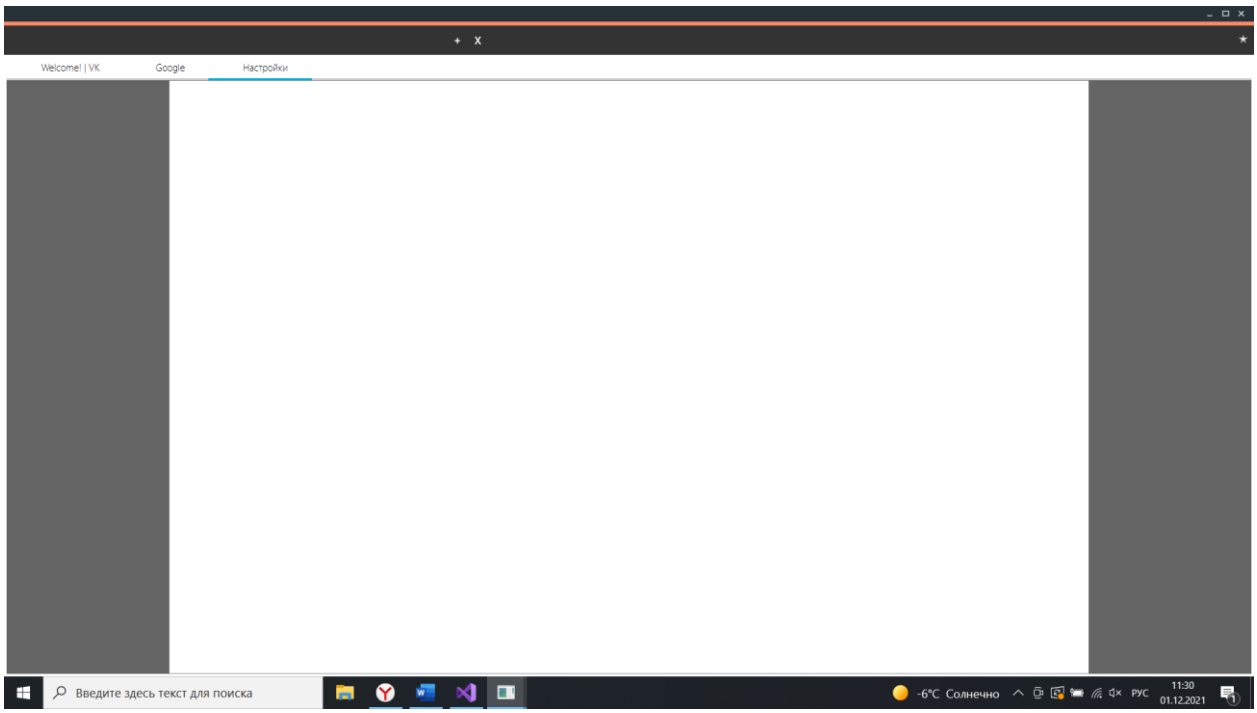

Рисунок 11 – Очищена история и закладки

6.5 Закрытие вкладок.

Вкладка закрывается по нажатию на кнопку, справа от адресной строки. Пример приведен на рисунке 12.

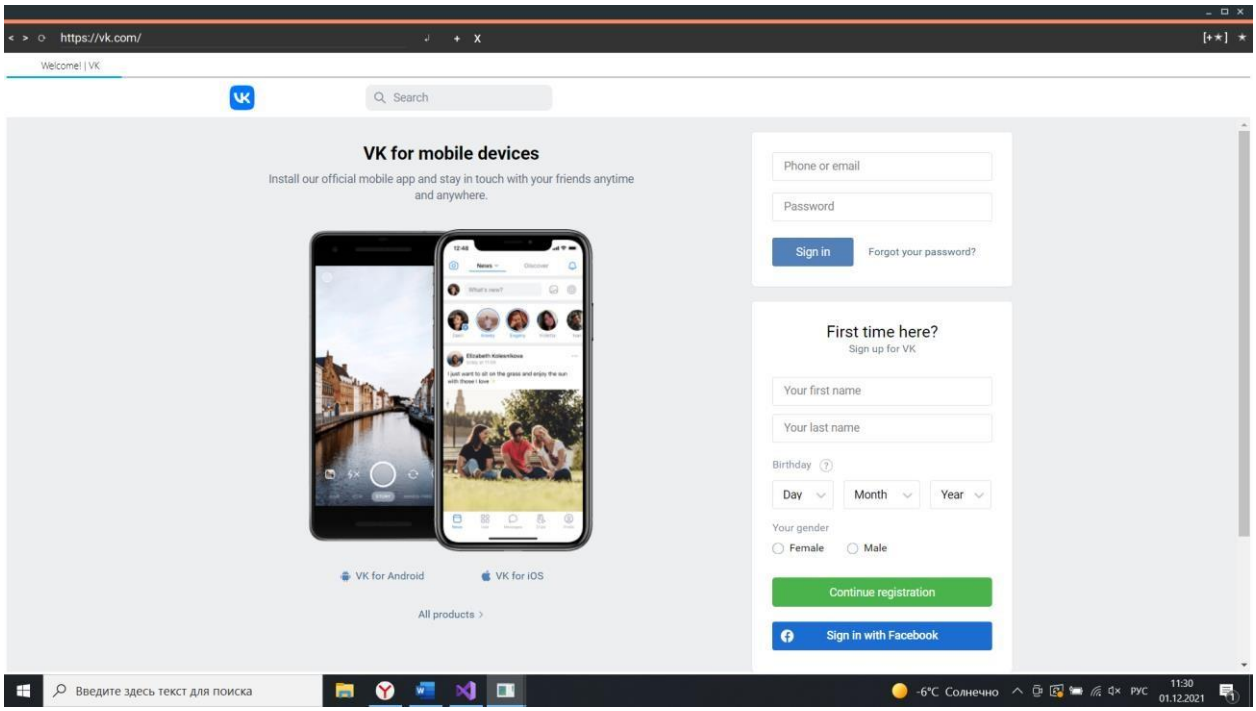

Рисунок 12 – Закрытие вкладок

Также, при попытке закрыть последнюю вкладку, сразу же будет открыта новая, после закрытия. Такую ситуацию можно наблюдать на рисунке 13.

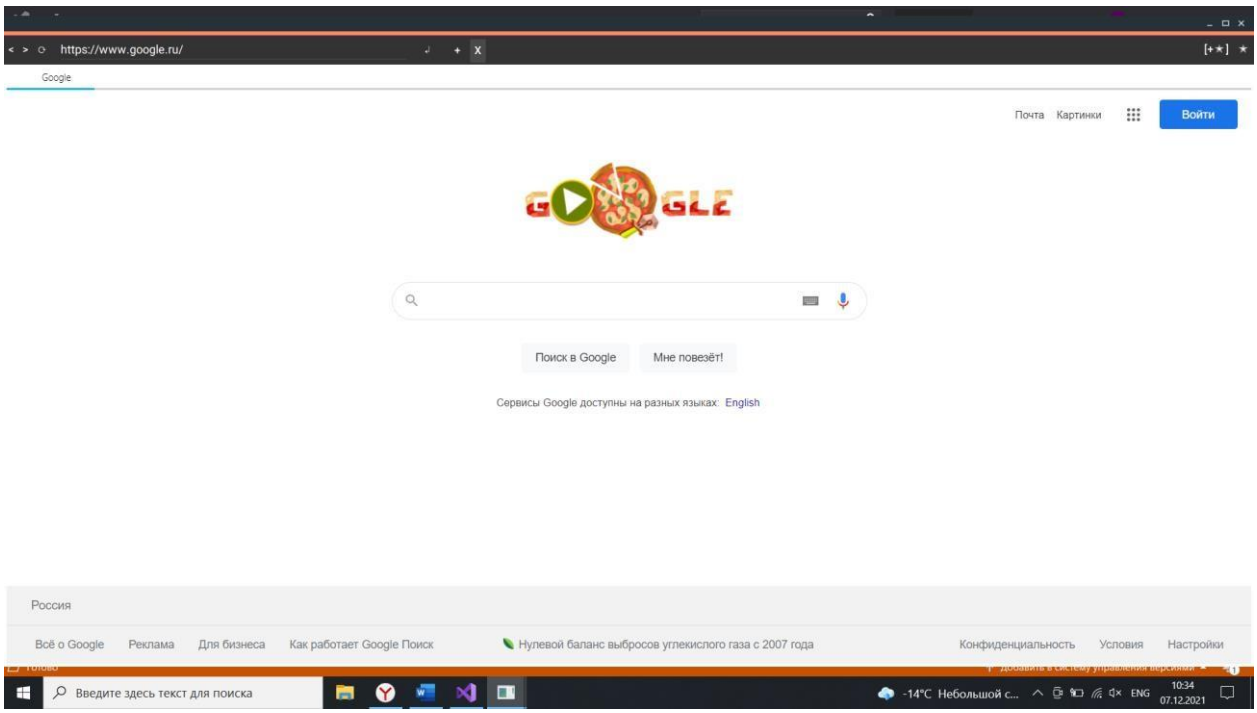

Рисунок 13 – Попытка закрыть последнюю вкладку

6.6 Умная поисковая строка.

Умная строка конвертирует запрос в ссылку. Пример приведен на рисунках 14 и 15.

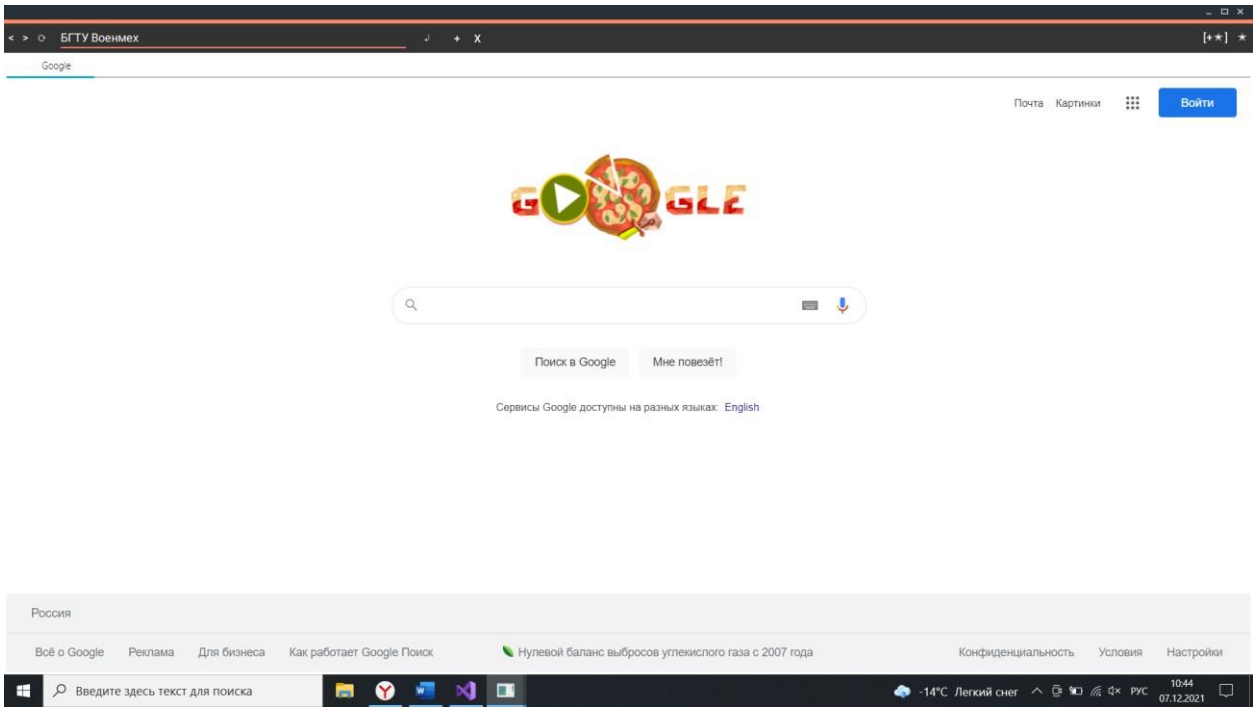

Рисунок 14 – Запрос до конвертации

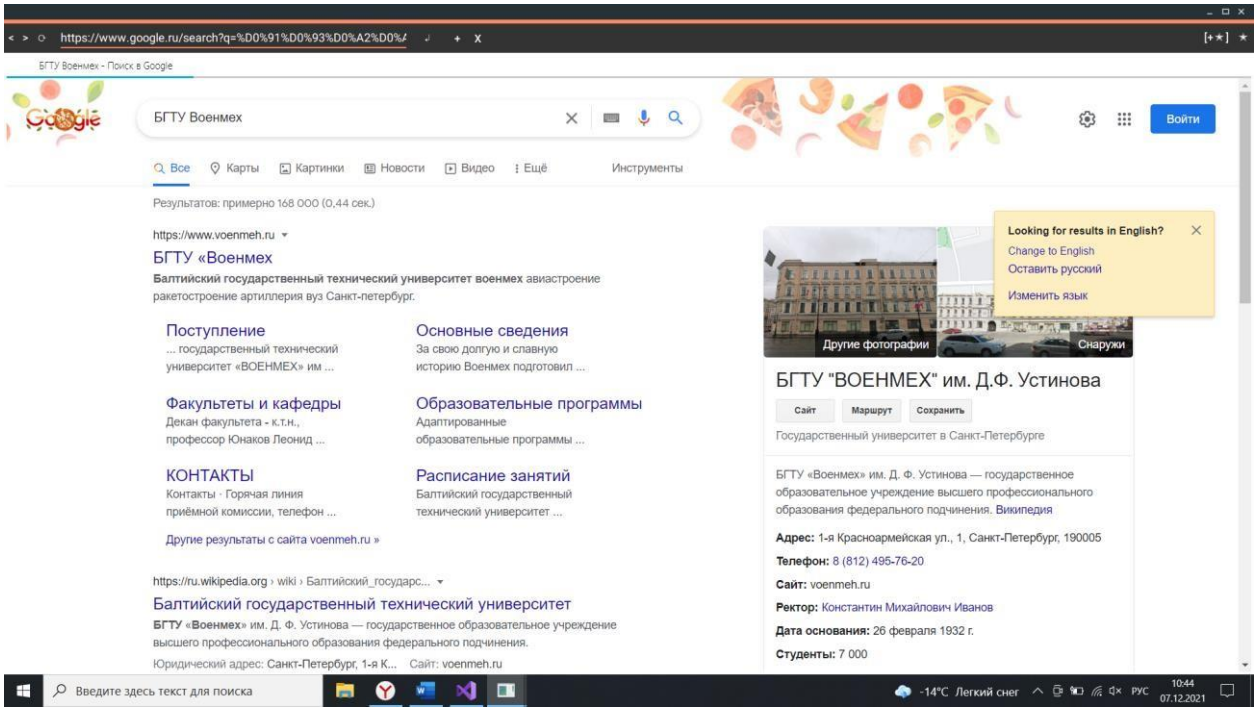

Рисунок 15 – Запрос после конвертации

ЗАКЛЮЧЕНИЕ

В рамках курсовой работы было разработано и протестировано приложение с использованием c# и windows forms и были изучены соответствующие библиотеки. Для разработки браузера был выбран движок Chromium, так как он обеспечивает корректное отображение web страниц. Были выполнены поставленные задачи: сохранение и отображение закладок и истории, открытие web страницы, работа с вкладками. Цель работы достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Эндрю Троелсен, Филипп Джепикс. Язык программирования C# 7 и платформы .NET и .NET Core. — М.: Диалектика, 2019. — 1328 с.

2 Джеффри Рихтер. CLR via C#. — М.: Питер, 2013. — 896 с.

3 Билл Вагнер. Наиболее эффективное программирование на C#. — М.: Вильямс, 2018. — 240 с.

4 Сергей Тепляков. Паттерны проектирования на платформе .NET. — М.: Питер, 2015. — 320 с.

5 Джон Скит. C# для профессионалов. Тонкости программирования. — М.: Вильямс, 2014. — 608 с.

ПРИЛОЖЕНИЕ А

Исходный текст программы

Исходные тексты программ представлены в прилагаемом архиве.