МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гродненский государственный университет имени Янки Купалы»

Факультет математики и информатики

**Кафедра современных технологий программирования**

БЛОЦКИЙ ДМИТРИЙ СЕРГЕЕВИЧ

**Текстовый редактор «Блокнот»**

Курсовая работа

студента 2 курса специальности

1-40 01 01 «Программное обеспечение информационных технологий»

дневной формы получения образования

|  |  |
| --- | --- |
|  | Научный руководитель  Домбровский Павел Эдуардович,  Преподаватель кафедры современных технологий программирования. |
|  |  |

Гродно, 2019

# РЕЗЮМЕ

Тема курсовой работы

**Текстовый редактор «Блокнот»**

Работа содержит: 28 страницы, 2 приложения, 9 использованных источников литературы.

Цель курсовой работы: разработка текстового редактора «Блокнот»

Объект исследования: приложение «Блокнот»

Предмет исследования: реализация приложения «Блокнот» на ЭВМ с помощью языка программирования C#, с использованием графического интерфейса Windows Forms.

Авторская характеристика работы: приложение «Блокнот» написано на языке программирования C#, с использованием графического интерфейса Windows Forms, и представляет собой простую реализацию, разработанную программой Microsoft Visual Studio.

# SUMMARY

The theme of the course work

**The text editor «Notepad»**

The work contains:

The purpose of the course work: the development of a text editor " Notepad»

Object of research: application **«**Notepad»

Subject of research: the implementation of the application "Notepad" on a computer using the C# programming language, using the graphical interface Windows Forms.

The author's characteristic of the work: the application **«**Notepad» is written in the C# programming language, using the Windows Forms graphical interface, and is a simple implementation developed by Microsoft Visual Studio.

содержание

[РЕЗЮМЕ 2](#_Toc27527455)

[SUMMARY 3](#_Toc27527456)

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ 5](#_Toc27527457)

[ВВЕДЕНИЕ 6](#_Toc27527458)

[1. АНАЛИЗ текстового редактора 9](#_Toc27527459)

[1.1. Основные аспекты 9](#_Toc27527460)

[1.2. Обзор существующих решений 9](#_Toc27527461)

[1.3. Выводы по главе 1 12](#_Toc27527462)

[2. ПРОЕКТИРОВАНИЕ приложения «Блокнот» 13](#_Toc27527463)

[2.1. Этапы разработки приложения. 13](#_Toc27527464)

[2.2. Функциональное моделирование. 13](#_Toc27527465)

[2.3. Проектирование графического интерфейса программы. 15](#_Toc27527466)

[2.4. Выводы по главе 2. 16](#_Toc27527467)

[3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ приложения «блокнот» 17](#_Toc27527468)

[3.1. Основные элементы 17](#_Toc27527469)

[3.2. Основная часть 17](#_Toc27527470)

[3.3. Выводы по главе 3 18](#_Toc27527471)

[заключение 19](#_Toc27527472)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 21](#_Toc27527473)

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ

|  |  |  |
| --- | --- | --- |
| MVP  ООП | Model View Presenter  Объектно-ориентированное программирование | Шаблон проектирования.  Методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования. |

# 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.

Основное назначение текстовых редакторов - создавать текстовые файлы, редактировать тексты, просматривать их на экране, изменять формат текстового документа, распечатывать его на принтере.

Набираемый на клавиатуре компьютера текст воспроизводится на экране дисплея в рабочем поле редактора. Специальный значок - курсор указывает то место на экране, на которое пользователь в данный момент может оказывать воздействие (создавать, изменять символы и т. д.) с помощью редактора. Работая с текстовым редактором, можно получить на экране информацию о текущем состоянии курсора, т.е. его координатах на экране (номер строки и позиции в строке), а также о номере страницы текста, его формате, используемом шрифте и т.д.

Интерфейс практически каждого текстового редактора позволяет иметь на экране меню команд управления редактором - изменение режимов работы, обращение за помощью, форматирование текста, печати и т. д. Как правило, меню имеет не только текстовую форму, но и форму пиктограмм, указывающих на выполняемую команду.

**Актуальность курсовой работы** заключается в разработке приложение текстовый редактор предназначенного для редактирования текста.

**Главная задача** курсовой работы заключается в правильности составленных алгоритмов. Потребуется создать полноценно-функционирующий текстовый редактор в среде C# и самостоятельно изучить некоторые функции и аспекты.

**Объект исследования:** приложение текстовый редактор.

**Предмет исследования:** реализация графического приложения «Блокнот» на ЭВМ с помощью языка программирования C#, с использованием графического интерфейса Windows Forms.

**Цель исследования:** разработка графического приложения «Блокнот», его непосредственная реализация и тестирование. В соответствии с объектом, предметом и целью исследования были поставлены следующие задачи:

* Изучить графический интерфейс Windows Forms.
* Разработать программу для создания графического приложения при помощи языка программирования C#; разработать пользовательский интерфейс позволяющий пользователю: безошибочно выполнять действия на ПК, использовать набор иерархических меню, систему подсказок, вернуться к предыдущему состоянию системы.
* Применение теоретических знаний по дисциплине «Языки программирования» и по темам, вынесенным на самостоятельное изучение, на практике для создания программного проекта, реализующего приложение научного характера.
* Изучение шаблона проектирования MVP.
* Получение необходимых практических навыков программирования.
* Приобретение навыков работы с графическим интерфейсом Windows Forms и использование её в научных проектах.

**Методы исследования:** объектно-ориентированное программирование, графический интерфейс Windows Forms, MVP.

**Практическая значимость** курсовой работы определяется огромной популярностью текстовых редакторов в современном мире.

Курсовая работа состоит из двух частей: основной и специальной. Основная часть состоит из компонентов, используемых при написании программы, также методов реализации проекта. Специальная часть состоит из описания программы и рабочего процесса. Завершает курсовую работу заключение и список используемой литературы, которая была использована при составлении курсовой работы и программы.

**Общий объём курсовой работы** составляет 28 страницы. Список использованной литературы состоит из 9 источников.

# **АНАЛИЗ текстового редактора**

## Основные аспекты

Текстовый редактор — самостоятельная компьютерная программа или компонент программного комплекса (например, редактор исходного кода интегрированной среды разработки или окно ввода в браузере), предназначенная для создания и изменения текстовых данных в общем и текстовых файлов в частности.

Текстовые редакторы предназначены для работы с текстовыми файлами в интерактивном режиме. Они позволяют просматривать содержимое текстовых файлов и производить над ними различные действия — вставку, удаление и копирование текста, контекстный поиск и замену, сортировку строк, просмотр кодов символов и конвертацию кодировок, печать и т. п.

Часто интерактивные текстовые редакторы содержат дополнительную функциональность, призванную автоматизировать действия по редактированию (от записываемых последовательностей нажатий клавиш до полноценных встроенных языков программирования), или отображают текстовые данные специальным образом (например, с подсветкой синтаксиса).

Многие текстовые редакторы являются редакторами исходного кода, то есть они ориентированы на работу с текстами программ.

## Обзор существующих решений

Программы-аналоги созданные ранее:

* Notepade2 — Название программы Notepad2 (Блокнот 2) говорит само за себя. Это приложение является усовершенствованным вариантом стандартного Блокнота Windows. Оно было написано Флорианом Балмером в 2004 году при помощи компонента Scintilla, массово использующегося также для разработки других аналогичных программ. Notepad2 имел значительно более развитый функционал, чем Блокнот. Но, в то же время, разработчики стремились, чтобы приложение оставалось небольшим и шустрым, как и его предшественник, и не страдало от переизбытка ненужного функционала. Программа поддерживает несколько кодировок текста, нумерацию строк, авто отступы, работу с регулярными выражениями, подсветку синтаксиса различных языков программирования и разметки, включая HTML, Java, Assembler, C++, XML, PHP и многих других. Вместе с тем, перечень поддерживаемых языков все-таки несколько уступает Notepad++. Кроме того, в отличие от своего более функционально продвинутого конкурента, Notepad2 не может работать в нескольких вкладках, и сохранять файлы, которые созданы в нем, в формате отличном от TXT. Программа не поддерживает работу с плагинами.
* AkelPad — Немного раньше, а именно в 2003 году, примерно в одно время с программой Notepad++, появился текстовый редактор российский разработчиков, получивший название AkelPad. Данная программа, хотя тоже сохраняет созданные ею документы исключительно в формате TXT, но в отличие от Notepad2 поддерживает не в пример большое количество кодировок. Кроме того, приложение может работать в многооконном режиме. Правда, подсветка синтаксиса и нумерация строк у AkelPad отсутствует, но главным преимуществом этой программы над Notepad2 является поддержка плагинов. Устанавливаемые плагины, позволяют существенно расширить функционал AkelPad. Так, один лишь плагин Coder добавляет в программу подсветку синтаксиса, сворачиваемость блоков, автодополнение и некоторые другие функции.
* Sublime Text — В отличие от разработчиков предыдущих программ, создатели приложения Sublime Text изначально ориентировались на то, что оно будет, в первую очередь, использоваться программистами. Sublime Text имеет встроенную функции подсветки синтаксиса, нумерации строк и автодополнения. Кроме того, у программы имеется возможность выделения столбцов и применения множественной правки без выполнения таких сложных действий, как использование регулярных выражений. Приложение помогает отыскать неисправные участки кода.

## Выводы по главе 1

Рассмотрены компьютерные программы типа текстовый редактор, в которых основной задачей является работа с текстовыми файлами в интерактивном режиме. Они позволяют просматривать содержимое текстовых файлов и производить над ними различные действия. Приведены аналоги таких приложений с указанием их достоинств и недостатков. Так, в приложении Notepade поддерживаются только базовые функции. Многие функции были доступны только из меню, а максимальный размер файла составлял 64 килобайта.

Таким образом, для создания современного текстового редактора требуется усвоить огромное количество информации из самых разных областей знаний.

# ПРОЕКТИРОВАНИЕ приложения «Блокнот»

## 2.1. Этапы разработки приложения.

Для достижения поставленной цели необходимо:

изучить теоретические сведения, требуемые для решения поставленной цели;

изучить требования, которым должно удовлетворять приложение;

изучены аналогичные приложения;

C#, Windows Forms интерфейс для разработки приложения;

разработать удобный пользовательский интерфейс;

создать приложение, реализующее применение теоретических знаний на практике для создания программного проекта;

протестировать созданное приложение, при необходимости внести соответствующие изменения.

проектирование функций для работы с картой, меню и другими элементами.

## 2.2. Функциональное моделирование.

Реализация приложения происходит по шаблону MVP. Изображено на рисунке 2.1.

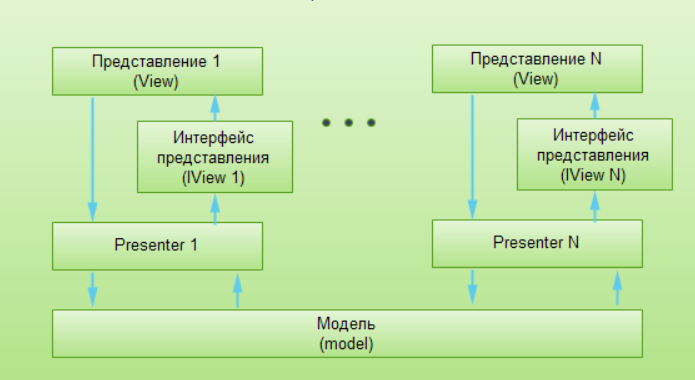


Рис. 2.1.

Реализация калькулятора выполняется по принципу MVP. IView передаёт в Presenter объекты и Presenter принимает эти объекты для дальнейшей реализации изображённой на рисунке 2.2.

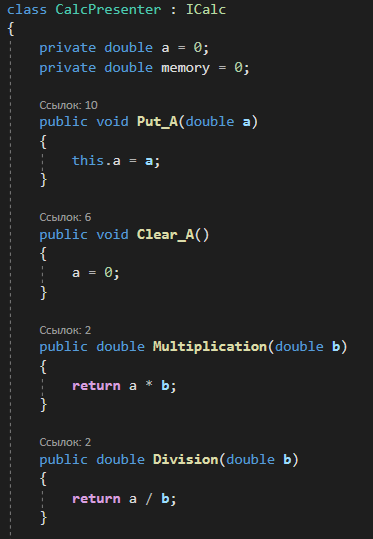


Рис. 2.2.

Мультиязычность реализована при помощи класса Settings и файлов Resource и Settings. В классе Settings происходит выбор языка и выставление языка по умолчанию, при последующем запуске программы значение выбранного языка сохраняется (при помощи файла Settings). Изображено на рисунке 2.3. В файл Settings записана переменная которая запоминает выбранный язык, а в файле Resource хранится перевод на выбранный язык. Изображено на рисунке 2.4.



Рис. 2.3.

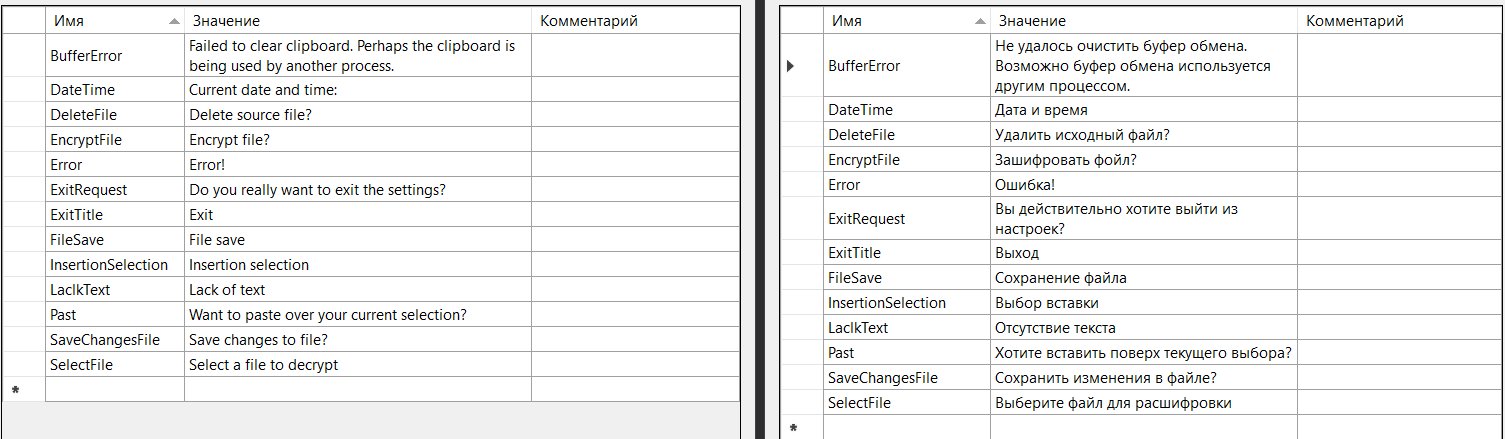


Рис. 2.4.

В созданной .dll библиотеке реализована криптография, при помощи класса Aes. С его помощью можно зашифровывать и расшифровывать текстовые классы по ключу. Изображено на рисунке 2.5.

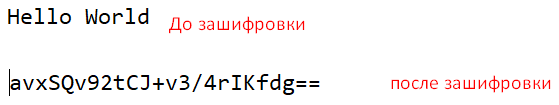


Рис. 2.5.

## 2.3. Проектирование графического интерфейса программы.

Интерфейс приложения изображён на рисунке 2.6.

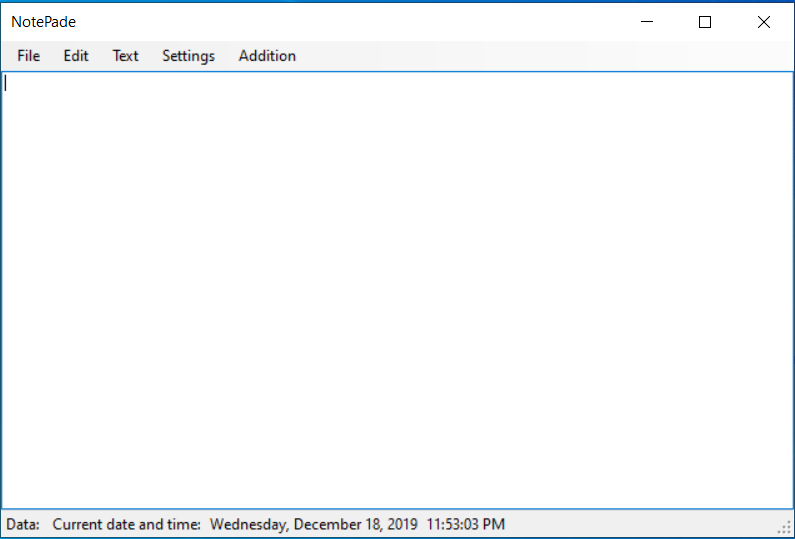


Рис. 2.6.

Интерфейс калькулятора изображён на рисунке 2.7.

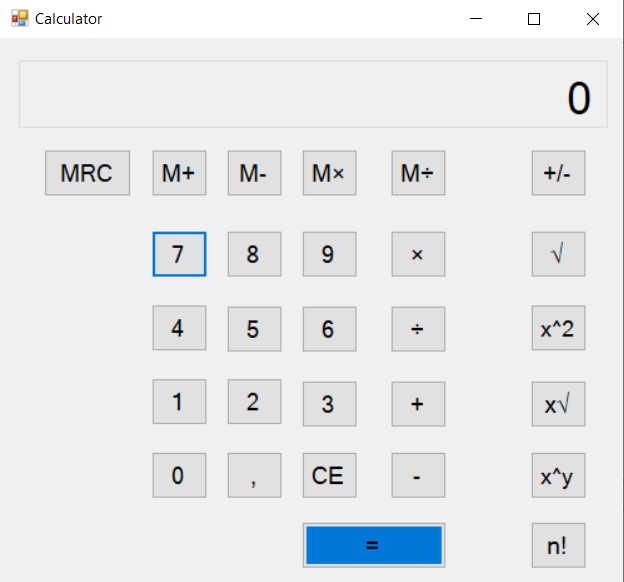


Рис.2.7.

Окно настроек изображено на рисунке 2.8.

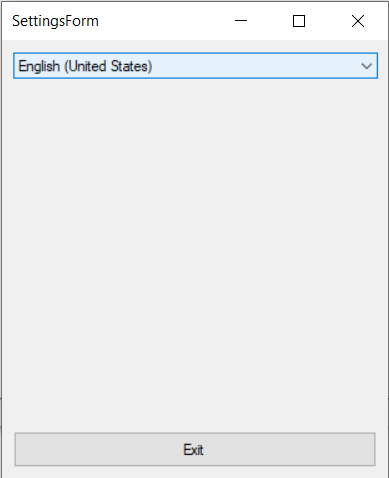


Рис. 2.8.

## 2.4. Выводы по главе 2.

Исходя из всего вышеперечисленного и рассмотренного, можно понять, что создание текстовых редакторов приложение является очень сложным, с одной стороны, но и очень полезным с другой.

Приложение является достаточно сложным, в своём роде, в плане поиска решений, но и, как оказалось, достаточно простым в реализации и довольно занимательным.

Не смотря на всё это, каждая программа имеет свои достоинства и недостатки, которые обуславливаются целевой аудиторией. Каждый пользователь в процессе использования замечает то, что, по его мнению, можно было добавить как необходимую часть для приложения, и то, что совсем не обязательно, и можно было исключить из кода. Кому-то нужен более простой интерфейс для удобства, а кто-то хочет более сложный для множества возможностей и функций.

# ПРОГРАММНАЯ РЕАЛИЗАЦИЯ приложения «блокнот»

## 3.1. Основные элементы

Основным и, пожалуй, самым первым и важным элементом приложения является поле для ввода текста, а если быть точнее, то действия которые можно реализовать над текстом в этом поле. Пользуясь широким функционалом и возможностями Windows Forms, создаётся несколько кнопок для работы с текстом: создание нового файла, открытие файла, сохранение и так далее. Изображено на рисунке 3.1.

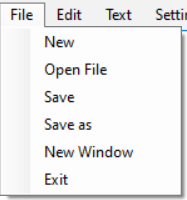


Рис. 3.1.

Когда пользователь хочет создать новый файл и нажимает нужную кнопку, программа, для того чтобы не потерять текст введённый пользователем предлагает сохранить его. После чего происходит обработка события которое вызывается при нажатии кнопки. Тоже самое происходит и при закрытии приложения. Немного иначе работает клавиша вызова нового окна. При её нажатии текущий текст не предлагается сохранить, так как в этом нет необходимости.

При быстром сохранении текста, новый документ сохраняется под названием «New document» в папке с приложением. Изображено на рисунке 3.2.

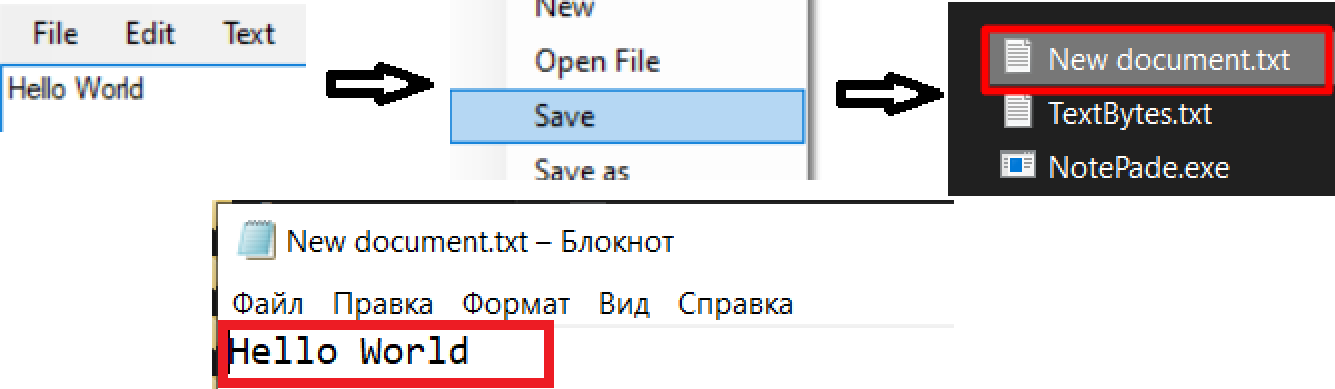


Рис 3.2.

Для пользователя есть возможность сохранить свой файл так, как он пожелает для этого необходимо нажать клавишу «Save as», но реализация несколько сложнее чем с быстрым сохранением.

Так же есть стандартные клавиши для копирования, вставки и вырезки текста. Изображено на рисунке 3.3.

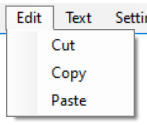


Рис. 3.3.

В случае если необходимо поменять шрифт, размер текста или его цвет, пользователю надо нажать клавишу «Font». Изображено на рисунке 3.4.

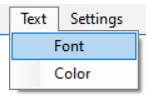


Рис. 3.4.

Для смены языка интерфейса пользователю необходимо перейти во вкладку «Settings». В появившемся окне выбрать необходимый язык. Язык интерфейса поменяется после закрытия окна с настройками.

## 3.2. Основная часть

## 3.2.1. Работа с формой

Основным и пожалуй самым главным элементом программы является класс Form1.cs, которое представляет окно, отображаемого в приложении. Класс Form1.cs используется для создания стандартных, модальных, диалоговых окон.

Используя свойства, определяется внешний вид, размер, цвет и функции управления окнами создаваемого окна или диалогового окна.

Все остальные формы открываются после их вызова из Form1.cs. Изображено на рисунке 3.5.

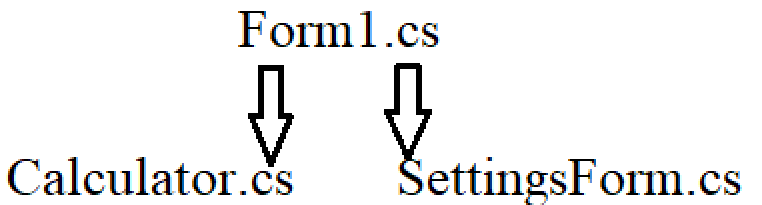


Рис. 3.5.

## 3.2.2. Использование шаблона MVP

Важный момент при создании нового файла. Здесь, в классе PageModel.cs хранится переменная обладающая автоматическими свойствами, которая хранит значение текстбокса. В моём случае в ней хранится пустая строка, но по желанию её можно изменить на любую другую строку и тогда при открытии приложения будет выводится текст который будет записан в переменную. Для этого и были созданы автоматические свойства её чтения и записи. На рисунке 3.6. изображен класс PageModel.cs.

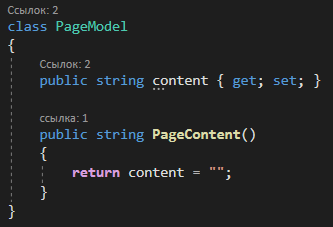


Рис. 3.6.

Класс PagePresenter.cs объединяет интерфейс IPage.cs, так как в нём описана сигнатура метода, и класс PageModel.cs. На рисунке 3.7. изображен класс PagePresenter.cs.

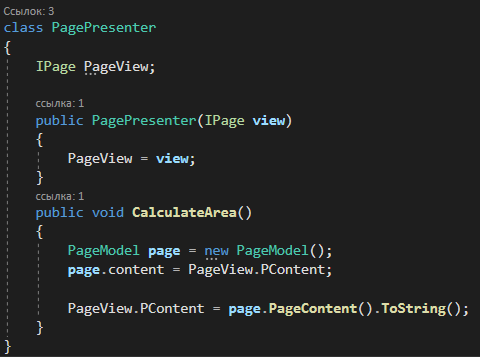


Рис. 3.7.

При реализации калькулятора CalcPresenter.cs наследует интерфейс ICalc.cs. В CalcPresenter.cs происходит выполнение необходимых операций. Изображено на рисунке 2.2.

## 3.2.2. Криптография

Для реализации криптографии использовался Aes класс. Этот класс является двунаправленным, а значит есть возможность как зашифровки так и расшифровки текста при условии что передаётся один и тот же ключ. Но класс создаёт ключ при запуске каждый раз новый. Следовательно если я попробую зашифровать текст, после чего я закрою программу и попробую запустить программу и расшифровать текст, то я увижу совершенно разный результат. Для решения этой проблемы в Aes класс передаётся заранее созданный ключ, который используется как для расшифровки так и для зашифровки. Класс создан так что необходимо передать в него только текст, после чего результат переходит в метод AesEnCryptography если хотим зашифровать и в метод AesDeCryptography если хотим расшифровать. После чего значение переходит в соответствующий метод Encrypt или Decrypt где и происходит зашифровка и расшифровка текста. Изображено на рисунке 3.8.

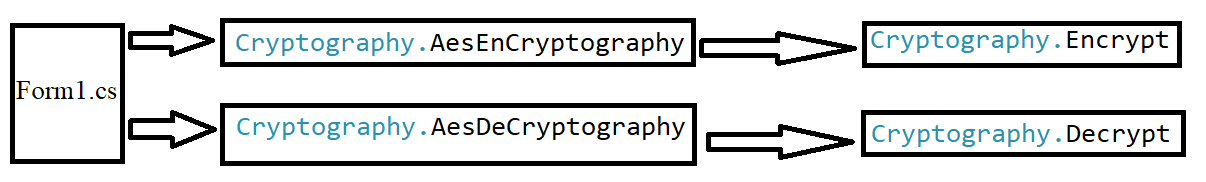


Рис. 3.8.

## 3.2.3. Мультиязычность

Для начала создаётся два файла ресурсов в которых и хранится русская и английская версии приложения. Там прописаны все тексты которые можно встретить в программе.

Основным и пожалуй самым важным элементом является класс SettingsForm.cs. В конструкторе формы прописывается код переключения культуры. Выбирается текущий поток, в котором выполняется приложение и устанавливается язык (культура). Значение культуры хранится в Setings файле и по этому, когда пользователь выберет язык и закроет окно, значение запишется в файл. Для того чтобы не надо было перезапускать всё приложение прописано несколько строк кода которые перезапускают все окна и устанавливают необходимы язык. Изображено на рисунке 3.9.

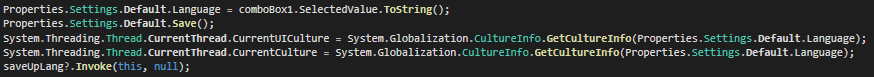


Рис. 3.9.

## 3.3. Выводы по главе 3

Разработанное приложение имеет множество функций, которые дополняют друг друга. Отличительной чертой является использование шаблона MVP. «Блокнот» имеет простой интерфейс и прост в своём использовании. В этой главе были пояснены все действия, которые происходят, непосредственно, в самом коде проекта. Так же проект можно неоднократно совершенствовать и вносить дополнительные функции. Можно вносить дополнительные возможности в приложение. В конечном итоге, блокнот можно адаптировать под популярный текстовый редактор «Microsoft Word».

# заключение

Ни для кого не секрет, что текстовые редакторы уже давно укрепили свою позицию в современном обществе. Не существует, пожалуй, ни одного пользователя, не использующего какой-либо текстовый редактор. Кто-то использует текстовые редакторы для работы, кто-то для учёбы. Но для некоторых текстовые редакторы стали целой жизнью, как для некоторых работников, так и для программистов или дизайнеров, занимающихся разработкой текстовых редакторов.

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. Или существенно расширяют её возможности.

В современных условиях все больше в нашу жизнь входят электронные средства документооборота. Программы, для работы с такими документами тоже быстро развиваются. Появляются все новые мощные и удобные программы, которые могут использоваться как дома, так и на работе.

Программа Notepad включает в себя богатые возможности по редактированию текста, такие как изменение шрифта (применяется только к текущей запущенной версии Notepad'а), копирование, вставка, вырезание, печать, клик, дабл клик. Также в Notepadе присутствует развитая система помощи, которая вызывается по нажатию на клавишу F1.

Отличительными особенностями редактора Notepad являются простота и низкие требования к ресурсам компьютера - вы сможете запустить Notepad даже на компьютере с 640 Кб ОЗУ и процессором Pentium II. При всем при всём этом Notepad имеет полностью графический интерфейс.

Текстовый редактор WordPad предоставляет возможности создания несложных форматированных текстов и является упрощенным вариантом профессионального редактора Word.

Программа WordPad позволяет задавать параметры страниц, форматировать текст, выравнивать абзацы, копировать и вставлять текст, осуществлять его поиск и др.

Пакет MS Office давно уже занимает прочные позиции на рынке офисных программ. MS Office - наиболее популярный пакет офисных приложений. Программы, входящие в этот пакет являются практически стандартом для любого компьютера, домашнего или офисного.

MS Office 2007 - новейшая версия пакета MS Office, которая вобрала в себя все достоинства предыдущих версий MS Office, работа в нем превращает обработку различных документов в простое, приятное и доступное занятие.

Конкурентом Microsoft Word 2003/2007 является на сегодня текстовый редактор Writer из пакета Openoffice.org - инструмент с богатыми возможностями для создания писем, книг, отчетов, информационных бюллетеней, брошюр и других документов. Можно вставлять графику и объекты от других компонентов в документы Writer. Writer может экспортировать файлы в HTML, XHTML, XML, Adobe's Portable Document Format (PDF), и некоторые версии файлов Microsoft Word. Он также соединяется с вашим почтовым клиентом.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Каймин В. А. 1.4. Редактирование текстов на ЭВМ // Информатика: учебник. — 2-е изд., перераб. и доп. — М.: ИНФРА-М, 2001. — 272 с. — («Высшее образование»).

2. Кристиан Нейгел и др. C# 5.0 и платформа .NET 4.5 для профессионалов = Professional C# 5.0 and .NET 4.5. — М.: «Диалектика», 2013. — 1440 с.

Эндрю Троелсен. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е издание = Pro C# 5.0 and the .NET 4.5 Framework, 6th edition. — М.: «Вильямс», 2013. — 1312 с.

3. Джеф Просиз. Программирование для Microsoft .NET = Programming Microsoft .NET. — М.: Русская редакция, 2003. — С. 704.

4. Рихтер Д. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд., 2018. — С. 896.

5. Дейтел П., Дейтел Х. Как программировать на Visual C# 2012. 5-е изд., 2014. — С. 864.

6. Официальный Интернет-портал Microsoft [Электронный ресурс] / Режим доступа: <https://dotnet.microsoft.com/>

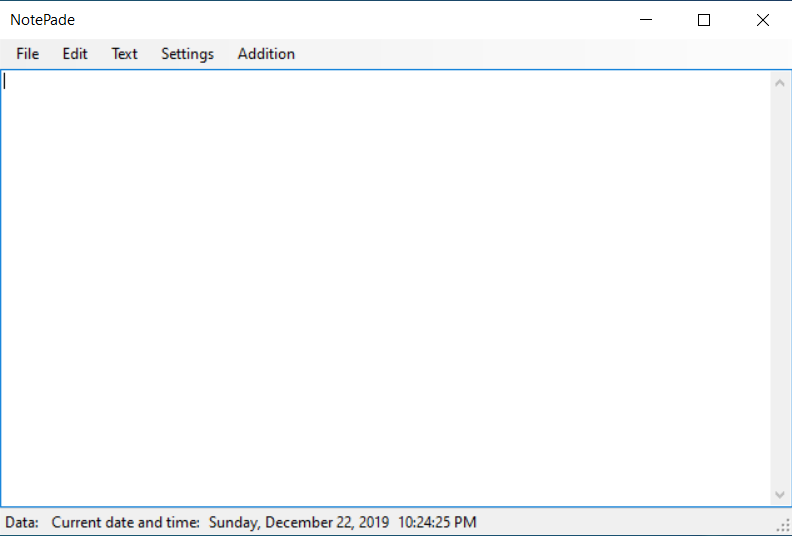
7. Интернет-портал [Электронный ресурс] / Режим доступа: <https://professorweb.ru/my/WPF/documents_WPF/level36/36_4.php>

8. Интернет-портал [Электронный ресурс] / Режим доступа: <https://metanit.com/sharp/windowsforms/1.1.php>

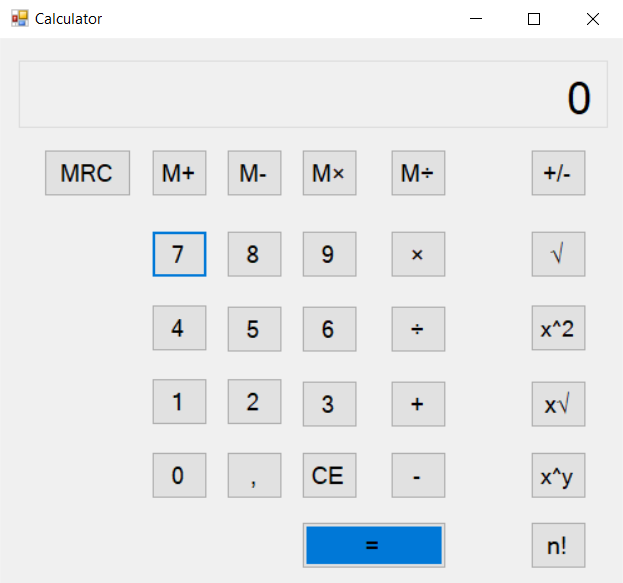
9. Официальный Интернет-портал Microsoft [Электронный ресурс] / Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.form?view=netframework-4.8>

# ПРИЛОЖЕНИЕ 1

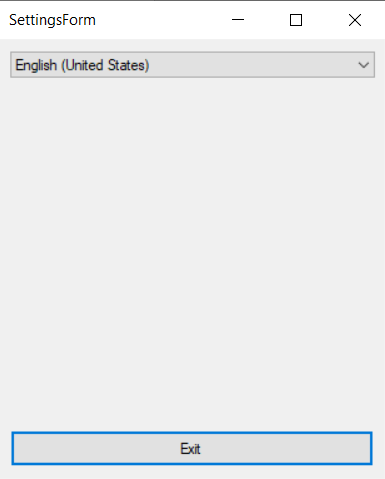
Интерфейс блокнота



Интерфейс калькулятора



Интерфейс окна настроек



# ПРИЛОЖЕНИЕ 2

**Основные функции и логика игры.**

Класс Cryptography

class Cryptography

{

public static void AesEnCryptography(string fileText, string pass1, string pass2, string metod, int pass\_interation, string initial\_vector, int keySize)

{

string bitPath = "TextBytes.txt";

string FinalValue = Encrypt(fileText, pass1, pass2, metod, pass\_interation, initial\_vector, keySize);

using (StreamWriter sw = new StreamWriter(bitPath))

{

sw.Write(FinalValue);

}

}

public static string AesDeCryptography(string fileText, string pass1, string pass2, string metod, int pass\_interation, string initial\_vector, int keySize)

{

string FinalValue = Decrypt(fileText, pass1, pass2, metod, pass\_interation, initial\_vector, keySize);

return FinalValue;

}

public static string Encrypt(string plainText, string password, string salt, string hashAlgorithm, int passwordIterations, string initialVector, int keySize)

{

if (string.IsNullOrEmpty(plainText))

return "";

byte[] initialVectorBytes = Encoding.ASCII.GetBytes(initialVector);

byte[] saltValueBytes = Encoding.ASCII.GetBytes(salt);

byte[] plainTextBytes = Encoding.UTF8.GetBytes(plainText);

PasswordDeriveBytes derivedPassword = new PasswordDeriveBytes(password, saltValueBytes, hashAlgorithm, passwordIterations);

byte[] keyBytes = derivedPassword.GetBytes(keySize / 8);

RijndaelManaged symmetricKey = new RijndaelManaged();

symmetricKey.Mode = CipherMode.CBC;

byte[] cipherTextBytes = null;

using (ICryptoTransform encryptor = symmetricKey.CreateEncryptor(keyBytes, initialVectorBytes))

{

using (MemoryStream memStream = new MemoryStream())

{

using (CryptoStream cryptoStream = new CryptoStream(memStream, encryptor, CryptoStreamMode.Write))

{

cryptoStream.Write(plainTextBytes, 0, plainTextBytes.Length);

cryptoStream.FlushFinalBlock();

cipherTextBytes = memStream.ToArray();

memStream.Close();

cryptoStream.Close();

}

}

}

symmetricKey.Clear();

return Convert.ToBase64String(cipherTextBytes);

}

public static string Decrypt(string cipherText, string password, string salt, string hashAlgorithm, int passwordIterations, string initialVector, int keySize)

{

if (string.IsNullOrEmpty(cipherText))

return "";

byte[] initialVectorBytes = Encoding.ASCII.GetBytes(initialVector);

byte[] saltValueBytes = Encoding.ASCII.GetBytes(salt);

byte[] cipherTextBytes = Convert.FromBase64String(cipherText);

PasswordDeriveBytes derivedPassword = new PasswordDeriveBytes(password, saltValueBytes, hashAlgorithm, passwordIterations);

byte[] keyBytes = derivedPassword.GetBytes(keySize / 8);

RijndaelManaged symmetricKey = new RijndaelManaged();

symmetricKey.Mode = CipherMode.CBC;

byte[] plainTextBytes = new byte[cipherTextBytes.Length];

int byteCount = 0;

using (ICryptoTransform decryptor = symmetricKey.CreateDecryptor(keyBytes, initialVectorBytes))

{

using (MemoryStream memStream = new MemoryStream(cipherTextBytes))

{

using (CryptoStream cryptoStream = new CryptoStream(memStream, decryptor, CryptoStreamMode.Read))

{

byteCount = cryptoStream.Read(plainTextBytes, 0, plainTextBytes.Length);

memStream.Close();

cryptoStream.Close();

}

}

}

symmetricKey.Clear();

return Encoding.UTF8.GetString(plainTextBytes, 0, byteCount);

}

}

Класс SettingsForm

public partial class SettingsForm : Form

{

public event EventHandler saveUpLang;

public SettingsForm()

{

if (!String.IsNullOrEmpty(Properties.Settings.Default.Language))

{

System.Threading.Thread.CurrentThread.CurrentUICulture = System.Globalization.CultureInfo.GetCultureInfo(Properties.Settings.Default.Language);

System.Threading.Thread.CurrentThread.CurrentCulture = System.Globalization.CultureInfo.GetCultureInfo(Properties.Settings.Default.Language);

}

InitializeComponent();

}

private void Exit\_btn\_Click(object sender, EventArgs e)

{

if (MessageBox.Show(Resource.ExitRequest, Resource.ExitTitle, MessageBoxButtons.YesNo) == DialogResult.Yes)

this.Close();

}

private void SettingsForm\_Load(object sender, EventArgs e)

{

comboBox1.DataSource = new System.Globalization.CultureInfo[] {

System.Globalization.CultureInfo.GetCultureInfo("en-US"),

System.Globalization.CultureInfo.GetCultureInfo("ru-RU")

};

comboBox1.DisplayMember = "NativeName";

comboBox1.ValueMember = "Name";

if(!String.IsNullOrEmpty(Properties.Settings.Default.Language))

{

comboBox1.SelectedValue = Properties.Settings.Default.Language;

}

}

private void SettingsForm\_FormClosing(object sender, FormClosingEventArgs e)

{

Properties.Settings.Default.Language = comboBox1.SelectedValue.ToString();

Properties.Settings.Default.Save();

System.Threading.Thread.CurrentThread.CurrentUICulture = System.Globalization.CultureInfo.GetCultureInfo(Properties.Settings.Default.Language);

System.Threading.Thread.CurrentThread.CurrentCulture = System.Globalization.CultureInfo.GetCultureInfo(Properties.Settings.Default.Language);

saveUpLang?.Invoke(this, null);

}

}