

## Введение

Для реализации дипломной работы были использованы следующие программные продукты, языки программирования и функции:



**Microsoft Visual Studio 2019** — линейка продуктов компании **Microsoft**, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows **Forms** и визуально привлекательные возможности взаимодействия с пользователем благодаря системе Windows Presentation Foundation (**WPF**). Приложение «**Мультимедийный материал для обучения по профессии "Оператор ЭВМ"**» написано на языках: C#, C++ и VB.NET.



**C#** — объектно-ориентированный язык программирования. Разработан в 1998 — 2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсбера и Скотта Вильтаума, как язык разработки приложений для платформы Microsoft .NET Framework.



**C++** — компилируемый, статически типизированный язык программирования общего назначения. Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Синтаксис C++ унаследован от языка C.



**Visual Basic .NET** (VB.NET) — объектно-ориентированный язык программирования, который можно рассматривать как очередной виток эволюции Visual Basic (VB), реализованный на платформе Microsoft .NET.

**Visual Studio Community** — стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (**IDE**) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения. Помимо стандартного редактора и отладчика, которые существуют в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства авто

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

ВСК.090204.07.00.000 ПЗ

Лист

4

завершения кода, графические конструкторы и многие другие функции для упрощения процесса разработки.

**Common Language Runtime** (англ. **CLR** — общезыковая исполняющая среда) — исполняющая среда для байт-кода CIL (MSIL), в которой компилируются программы, написанные на .NET-совместимых языках программирования (C#, Managed C++, Visual Basic. CLR является одним из основных компонентов пакета Microsoft.



**.NET Framework 4.7** — программная платформа, выпущенная компанией Microsoft в 2002 году. Основой платформы является общезыковая среда исполнения Common Language Runtime (CLR), которая подходит для разных языков программирования. Функциональные возможности CLR доступны в любых языках программирования, использующих эту среду. Основной идеей при разработке .NET Framework являлось обеспечение свободы разработчика за счёт предоставления ему возможности создавать приложения различных типов, способные выполняться на различных типах устройств и в различных средах.

**IntelliSense** — набор функций, отображающих сведения о коде непосредственно в редакторе и в некоторых случаях автоматически создающих небольшие отрывки кода. По сути, это базовая документация,строенная в редактор, с которой Вам не приходится искать информацию где-то еще. Функции IntelliSense зависят от языка. Дополнительные сведения были взяты мной в руководствах по IntelliSense для C# , C++ и IntelliSense для Visual Basic.



**Adobe Acrobat Pro DC** (версия 2019.012.20040) — это полностью обновленная настольная версия лучшего в мире решения для работы с файлами PDF. В состав продукта входит мобильное приложение, позволяющее заполнять, подписывать и отправлять формы PDF с любых устройств. Облачные сервисы Document Cloud в Acrobat Pro DC позволяют создавать, экспортить, редактировать и отслеживать файлы PDF, открывая их в любом web-браузере.



**Adobe Photoshop 2020** — многофункциональный графический редактор, разрабатываемый и распространяемый компанией Adobe Systems. В основном работает с растровыми изображениями, однако имеет

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

5

некоторые векторные инструменты. Продукт является лидером рынка в области коммерческих средств редактирования растровых изображений и наиболее известной программой разработчика.

 **PicPick v5.1.1** — профессиональный инструмент для захвата экрана для Windows. Функционал **Picpick** является продвинутым для такого маленького и бесплатного приложения: вставка различных стрелок, указателей, фигур, подчеркиваний, надписей, комментариев и т.д. Также есть возможность наложение красивых эффектов: инвертирование, градиент серого, тени, размытие, резкость, яркость и прочее. В общем присутствует все что необходимо при работе с графикой.

 **Notepad v7.8.7** — свободный текстовый редактор с открытым исходным кодом для Windows с подсветкой синтаксиса, разметки, а также языков описания аппаратуры VHDL и Verilog. 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. Поддерживает открытие более 100 форматов.

 **Install Creator Pro v.2** — очень простой и в то же время профессиональный инструмент для создания инсталляционных файлов. Режим “wizard” позволяет выбрать файлы, которые Вы хотите добавить в инсталляцию, задать инсталляционный путь по умолчанию, добавить лицензионное соглашение и т.д. После чего программа создаёт сжатый “.exe” файл, Вы сможете добавить любые иконки, логотипы и текст, настроить дополнительные опции: DLL, OCX, REG, TLB регистрации, установка шрифтов, ярлыков на рабочий стол и многое другое. Также можно добавить опцию полной деинсталляции созданного приложения.

 **Smart Install Maker 5.04** — условно-бесплатная утилита с закрытым исходным кодом, предназначенная для создания инсталляторов в 32-битных операционных системах Microsoft Windows. Утилита предоставляет алгоритм “Cabinet” для сжатия и создания компактных инсталляторов, поддерживает

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

6

создание многоязычного инсталлятора (до 25 языков), а также предоставляет средства для управления внешним видом программы установки.



**Setup Factory v9.5.0.0** — отличная программа для создания инсталляторов, которая не только компилирует и добавляет все файлы, необходимые для установки программы, но и включает в себя дополнительные опции настройки, чтобы пользователи программного обеспечения могли наслаждаться абсолютно профессиональным мастером настройки.

### Понятие пакета программ

Гражданский кодекс — систематизированный законодательный акт, содержащий расположенные по определённой системе нормы гражданского права. При этом может использоваться как институционная система, так и пандектная система. В ст.1261 ГК РФ содержится определение программы для ЭВМ (*далее — программа*), в качестве разновидности которой наряду с операционной системой упоминается и программный комплекс. Однако само это понятие в ГК РФ не раскрывается. В отличие от аппаратно-программного комплекса (включающего, как это следует из самого его наименования, аппаратные средства — электронные и механические части устройств (**hardware**) и программное обеспечение (**software**)), программный комплекс могут составлять только программы (*software*).

Из содержания ст.1261 ГК РФ вытекает, что всякая программа предназначена для выполнения конкретной функции (функций), ориентированной на достижение определенного результата. При этом ГОСТ 19.101-77 «*Виды программ и программных документов*» предусматривает подразделение программ на два вида:

1) **компонент** (*программа, рассматриваемая как единое целое, выполняющая законченную функцию и применяемая самостоятельно или в составе комплекса*);

2) **комплекс** (*программа, состоящая из двух или более компонентов и (или) комплексов, выполняющих взаимосвязанные функции, и применяемая самостоятельно или в составе другого комплекса*).

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

7

Исходя из ст. 1261 ГК РФ и с учетом названного ГОСТа можно заключить, что программы образуют комплекс (*т.е. программный комплекс*) только при условии, что они нацелены на выполнение взаимосвязанных функций, приводящих к достижению искомого результата.

В зависимости от функций, на выполнение которых нацеливаются программы, они обычно подразделяются на:

- **системные** (*выполняющие функции управлению ресурсами компьютера, осуществляющие поддержку работоспособности системы обработки информации или повышения эффективности ее использования, восстановление работы системы после выявления неисправностей в технических средствах*);
- **инструментальные** (*облегчающие процесс создания новых программ*);
- **прикладные** (*обеспечивающие выполнение необходимых пользователю функций, решение конкретных задач*).

Например, под прикладными программами (**application program**) в силу положений ГОСТ 19781-90 «Обеспечение систем обработки информации программное. Термины и определения» понимают программы, предназначенные для решения задачи или класса задач в определенной области применения системы обработки информации. Иными словами, прикладные программы (называемые "приложениями под конкретную область применения" или кратко — **приложениями**) используются пользователями для достижения конкретного результата при решении задачи в соответствующей предметной области.

Таким образом, прикладные программы образуют программный комплекс в смысле ст.1261 ГК только при условии, что они: 1) *выполняют взаимосвязанные функции*; 2) *нацелены на достижение искомого результата в одной предметной области*. В качестве примера можно вспомнить программный комплекс ФНС РФ, используемый в целях осуществления налогового административного администрирования.

*Как отличить программный комплекс от пакета программ?* В соответствии с положениями ГОСТ 15971-90 «Системы обработки информации программное. Термины и определения» термином "пакет прикладных программ" (**application**

Изм.	Лист	№ докум.	Подпись	Дата	Лист	8
					<b>ВСК.090204.07.00.000 ПЗ</b>	

**program package**) обозначается система прикладных программ, предназначенная для решения задач определенного класса. То есть в отличие от программного комплекса, в котором объединены программы, нацеленные на решение задачи (*задач*) в одной предметной области, пакет прикладных программ (*пакет приложений*) объединяет программы ("компоненты" в терминологии ГОСТ 19.101-77), которые решают схожие задачи в разных предметных областях. К особенностям пакетов приложений обычно относят: стандартный пользовательский и программный интерфейс каждого компонента, облегченный перенос данных между компонентами, наличие базы данных для хранения данных и их передачи приложениям, возможность выбора самим пользователем состава пакета и т.д.

Одним из самых известных пакетов приложений является пакет Microsoft Office, в состав которого входят различные программы для работы с различными типами документов: таблицами, текстами, электронными таблицами, базами данных и т.д. (Microsoft: *Word*, *Publisher*, *Excel*, *OneNote*, *Outlook* и т.д.).

Следовательно, можно говорить о принципиальных различиях между программным комплексом и пакетом приложений. Необходимость разграничения данных понятий обусловлена тем, что по смыслу ст.1261 ГК РФ программный комплекс признается единым объектом интеллектуальной собственности (*по сути, он приравнен к единичной программе для ЭВМ*), тогда как пакет программ следует рассматривать как набор нескольких самостоятельных объектов интеллектуальной собственности (*нескольких приложений*).

Моя дипломная работа (2021 года), является пакетом программ. Который можно и дальше дорабатывать, улучшать процессы и увеличивать их число до бесконечности, для получения необходимого результата (*независимо от прикладной области*).

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

9

## Профессия «Оператор ЭВМ»

Информация, с которой работает оператор ЭВМ, может содержаться в самых разных форматах. Он может работать как с текстовыми, так и с аудио-, видео- и графическими редакторами, программами для просмотра и воспроизведения, информационными интернет-ресурсами и пр. Должен уметь вводить информацию в компьютер также разными способами: набрать на компьютере текст, отсканировать (*т.е. оцифровывать*) текст или изображение, перенести уже готовые материалы с другого носителя (*CD/DVD диска, USB-флэшки и т.п.*).

Поле деятельности оператора ЭВМ зависит от места работы. Один из вариантов работы — внесение данных в электронную базу товаров в торговой организации. Это очень ответственная обязанность, хотя и довольно рутинная.

Более творческая работа у сотрудников издательств, редакций печатных и интернет-проектов. Там обязанности могут варьироваться от обычного набора текста, до обработки аудио- и видеофайлов, создания слайд-шоу и т.п. для размещения на сайте.

Профессию оператора ЭВМ можно отнести к числу стартовых. Во-первых, для овладения ею достаточно получить начальное профессиональное образование. Во-вторых, обработчик цифровой информации — это пользователь ПК, владеющий некоторыми программами на профессиональном уровне, основами построения информационных систем. Это хорошая база для дальнейшего образования в этой области.

Оператор ЭВМ может работать в банках, страховых компаниях, торговых и промышленных фирмах, Call-центрах, издательствах, компаниях, занятых разработкой и поддержанием сайтов, архивах и т.д., и т.п.

Данная профессия предполагает такие качества, как ответственность, исполнительность, усидчивость, способность к сосредоточенной работе, коммуникабельность.

Оператор должен уметь работать с пакетами офисных программ (*Microsoft Office* и *OpenOffice.org*), пользоваться Интернетом, электронной почтой. Также

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

ВСК.090204.07.00.000 ПЗ

Лист

10

может быть, востребовано знание графических программ (*CorelDraw*, *Adobe Photoshop*, *GIMP* и др.). Большим достоинством считается умение печатать быстро и грамотно (*желательно вслепую*). Помимо этого, оператор должен уверенно обращаться со сканером, принтером и другими периферийными устройствами. Хорошо понимать принципы построения компьютерных сетей.

Чтобы освоить профессию оператор ЭВМ (*оператор ПК*) нужно окончить колледж и получить квалификацию «*мастер по обработке цифровой информации*». Эта профессия входит в перечень программ начального профессионального образования.

Имея основную базу навыков работы на компьютере, можно всегда оставаться на своем месте. На любом предприятии сотрудники, обладающие знаниями оператора ПК весьма ценные.

### Общие принципы интерфейса

Разработка интерфейса велась в соответствии с поставленной для дипломной работы задачей, т.е. продукт должен предназначаться для пользователей, которые хотят освоить данную профессию. Но, в процессе разработки, я пришел к выводу, что данное приложение можно использовать для обучения самым разнообразным дисциплинам (“для всего!”) — просто добавляя соответствующую учебную информацию и составляя тестирование по предоставленным материалам.

Реальное приложение WPF неизбежно содержит изрядное количество элементов пользовательского интерфейса (*заставка* (Рисунок 1), окно входа (Рисунок 2), *панель регистрации* (Рисунок 3), окно для учебных материалов (Рисунок 4) и *тестирований*, элементов ввода, графического содержимого, систем меню, строк состояния и т.п., которые я постарался хорошо организовать внутри содержащего их окна. Кроме того, виджеты пользовательского интерфейса ведут себя адекватно и в случае изменения конечным пользователем размеров всего окна или его части (как в случае рабочего (“учебного”, с тестами) окна “*Оператор ЭВМ*”). Для гарантирования того, что элементы управления WPF сохранят свое

						Лист
Изм.	Лист	№ докум.	Подпись	Дата	<b>ВСК.090204.07.00.000 ПЗ</b>	11

положение в принимающем окне, я предусмотрел значительное количество панелей и внешний форм (*и часто использовал свойство “Margin”*).

При объявлении элемента управления непосредственно внутри окна, не имеющего панелей, я размещаю его в центре окна. Независимо от того, как пользователь будет изменять размеры окна, виджет пользовательского интерфейса всегда окажется на равном удалении от всех четырех границ клиентской области.

Т.к. окно WPF может содержать только один элемент... Чтобы разместить более одного элемента и создать практичный пользовательский интерфейс, я поместить в окно контейнеры и добавлял элементы уже в эти контейнеры. Это ограничение обусловлено тем фактом, что класс **“Window”** унаследован от **“ContentControl”** (*базовый класс для элементов управления*).

В WPF компоновка определяется используемым контейнером. Хотя есть несколько контейнеров, среди которых можно выбирать, "идеальное" окно WPF следует описанным ниже ключевым принципам:

- Элементы (*такие как элементы управления*) не должны иметь явно установленных размеров. Вместо этого они растут, чтобы уместить свое содержимое. Например, появляются полосы прокрутки при добавлении текста. Можно ограничить элементы управления приемлемыми размерами, устанавливая максимальное и минимальное их значение.
- Элементы не указывают свою позицию в экранных координатах. Вместо этого они упорядочиваются своим контейнером на основе размера, порядка и (*необязательно*) другой информации, специфичной для контейнера компоновки.
- Контейнеры компоновки "разделяют" доступное пространство между своими дочерними элементами. Они пытаются обеспечить для каждого элемента его предпочтительный размер (*на основе его содержимого*), если только позволяет свободное пространство. Они могут также выделять дополнительное пространство одному или более дочерним элементам.
- Контейнеры компоновки могут быть вложенными. Типичный пользовательский интерфейс начинается с **“Grid”** — наиболее развитого

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

12

контейнера, и содержит другие контейнеры компоновки, которые организуют меньшие группы элементов, такие как текстовые поля с метками, элементы списка, значки в панели инструментов, колонка кнопок и т.д.

Для обрамления элементов интерфейса использую холодные цвета (*за основу беру синий цвет*). Для вложенных элементов интерфейса применяю теплые цвета. Это связано с психологией восприятия. Форма, как и в большинстве случаев — это просто прямоугольник (*за исключением окна: “для входа и регистрации”*). Основные ЭИ (*чаще используемые*) оформлены в виде кнопок (представил разными геометрическими формами) или выделены разными цветами, ко многим кнопкам я применил эффекты анимации и добавлял “**Tooltip**” — т.е. сделал многие кнопки интерактивными, при наведении на них появляется подсказка с текстом или изображением. Постарался не делать слишком маленькие элементы, чтобы не возникало трудностей с их использованием.

Элементы упорядочены (*сгруппированы по “вкладкам”*) по программным продуктам. Т.е. на каждой вкладке речь идёт об **одной** программе. Элементы размещают в следующей градации: слева направо, сверху вниз. Слева направо в верху самые значимые элементы (*по наименованиям программных продуктов*), внизу — менее значимые (*кнопки, для вызова дополнительных форм и меню*). Это связано с порядком чтения текста.

Экспериментирую с многоязычным пользовательским интерфейсом... **MUI** (Multilingual User Interface) — это название технологии Microsoft для Microsoft Windows, Microsoft Office и других приложений, которая позволяет устанавливать несколько языков интерфейса в одной системе. В системе с MUI каждый пользователь сможет выбрать свой предпочтительный язык отображения. Данная технология была представлена в Windows 2000 и с тех пор использовалась в каждом выпуске (*вплоть до Windows 10*). Технология MUI защищена международным патентом «**Многоязычный пользовательский интерфейс для операционной системы**». Изобретатели — *Бьорн К. Реттиг, Эдвард С. Миллер, Грегори Уилсон и Шан Сюй*. Язык можно менять во время работы приложения «Оператор ЭВМ», без

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

13

необходимости перезапуска программы (перевожу лишь часть элементов интерфейса).

В самом начале реализую “программный” аудио интерфейс. Благодаря которому озвучены некоторые действия пользователя (в частности смена языка).

Данная работа представляет собой обширный API-интерфейс имеющий насыщенный дизайн и интерактивность. В отличие от устаревшей технологии Windows Forms, WPF позволяет мне включить в проект новую модель построения пользовательских приложений (в основе *WPF* лежит мощная инфраструктура, основанная на **DirectX**). Также учитываю обычные привычки пользователя.

Например, в Windows кнопка закрыть находится в правом верхнем углу, и в программе «Оператор ЭВМ» аналогичную кнопку размещаю там же (*а в окне регистрации — “справа”*). Т.е. в интерфейсе делаю как можно больше аналогий, с известными пользователю вещами, но и отличиям нахожу достойное применение.



Рисунок 1 – Заставка при запуске программы

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

14



Рисунок 2 – Окно входа для зарегистрированных пользователей (*требуется ввести почту и пароль, кнопка закрытия в левом верхнем углу, часы и языковая панель внизу справа, DataGrid с зарегистрированными пользователями почти внизу*)



Рисунок 2 – Панель регистрации в программе

<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>

ВСК.090204.07.00.000 ПЗ

Лист

15

Начальный вид главного окна можно увидеть в **Приложении 1**.

Данная программа, на текущий момент, даёт пользователям хорошие знания по следующим программным продуктам:

- **Windows 10** — операционная система для персональных компьютеров и рабочих станций, разработанная корпорацией Microsoft в рамках семейства Windows NT. Система призвана стать единой для разных устройств, таких как персональные компьютеры, планшеты, смартфоны, консоли Xbox One и пр. Согласно статистическим данным сайта *W3Schools*, Windows 10 занимает первое место в мире среди операционных систем, используемых для доступа к сети Интернет, опередив в апреле 2017 года предыдущего лидера — Windows 7;
- **Windows 7** — пользовательская операционная система семейства Windows NT компании Microsoft. Следует по времени выхода за Windows Vista и предшествует Windows 8. Расширенная поддержка прекращена 14 января 2020 года. Поддержка для ОС на сегодняшний день платная, действует до 10 января 2023 года;
- **Linux Ubuntu** — является 1-м в списке самых популярных дистрибутивов Linux для веб-серверов. Ubuntu ([ʊ'buntu:]; от зулу ubuntu — человечность; «Убунту») — дистрибутив Linux, основанный на Debian GNU/Linux. Основным разработчиком и спонсором является компания Canonical. В настоящее время проект активно развивается и поддерживается свободным сообществом;
- **ArchLinux** — независимый дистрибутив GNU/Linux общего назначения, оптимизированный для архитектуры x86-64, который стремится предоставить последние «стабильные» версии программ, следя модели *rolling release*. По умолчанию пользователю предоставляется минималистичная базовая система, в которую пользователь может добавить то, что ему требуется. Для установки, удаления и обновления пакетов используется пакетный менеджер **Расман**;

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

Лист

16

- **Mac OS LionX** — операционная система для персональных компьютеров и серверов, разработанная **Apple**. Mac OS X Lion сохраняет копии документов каждый раз, когда Вы их открываете и создает их каждый час во время их редактирования. Затем, при желании, вы сможете просмотреть большой список редакций одного и того же документа. Отображается это очень наглядно — в виде каскада окон;
- **Mac OS Sierra** — операционная система для персональных компьютеров и серверов, разработанная Apple. Это тринадцатая по счёту версия macOS. Анонсирована 13 июня 2016 года на конференции WWDC 2016. Публичный релиз ОС состоялся 20 сентября 2016 года;
- **MS DOS** — (англ. Disk Operating System — дисковая операционная система, ДОС) — в широком смысле слова, операционная система для компьютеров, ориентированных на использование дисковых накопителей, таких как жёсткий диск и дискета. Любая ДОС поддерживает одну или несколько файловых систем для организации хранения, чтения и записи с накопителей. Современные графические ОС, такие как Windows или Linux, также попадают под это понятие;
- **Microsoft Office 2016** — версия офисного пакета компании Microsoft, следующая за Microsoft Office 2013. Релиз продукта состоялся 23 сентября 2015 года. В составе пакета можно приобрести такие приложения как Word, Excel, PowerPoint, OneNote, Outlook, Skype для бизнеса, Access и Publisher. Отдельно от пакета распространяются Visio и Project;
- **Microsoft Word** — данное программное обеспечение предназначено для создания, редактирования, просмотра текстовых документов и обмена ими. Новая версия текстового процессора Microsoft Word 2016 стала более структурированной и удобной для восприятия, предоставляя быстрый доступ ко всем командам и панелям инструментов;
- **Microsoft Excel** — данное программное обеспечение предназначено для работы с электронными таблицами в целях ведения как финансовой отчетности предприятия, так и личной бухгалтерии. Продукт Microsoft Office

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

17

Excel предоставляет возможности экономико-статистических расчетов, графические инструменты и язык макропрограммирования **VBA** (Visual Basic для приложений). Microsoft Excel является одним из наиболее популярных аналитических систем и содержит усовершенствованные средства построения диаграмм и совместного доступа к информации;

- **Microsoft Access** — это система управления базами данных (**СУБД**) от Microsoft, которая объединяет реляционный Microsoft Jet Database Engine с графическим интерфейсом пользователя и инструментами разработки программного обеспечения. Имеет широкий спектр функций, включая связанные запросы, связь с внешними таблицами и базами данных;
- **Adobe Acrobat Reader** — это бесплатное автономное приложение, в котором можно открывать, просматривать, подписывать, печатать, комментировать и проверять файлы PDF, выполнять поиск по тексту, а также предоставлять общий доступ к документам PDF другим пользователям;
- **Adobe After Effect** — программное обеспечение компании Adobe Systems для редактирования видео и динамических изображений, разработки композиций, анимации и создания различных эффектов;
- **Adobe Illustrator** — инструмент для векторного рисования и редактирования художественных работ. Он помогает создавать графические проекты во всех стандартных форматах, а также предоставляет функции печати, размещения на веб-сайтах. Платформа разработана для специалистов в области графического дизайна;

**Исходный код** данного приложения можно расширять и улучшать — количество учебных дисциплин, рабочих модулей, различных тестов и приложений ограничено разве что фантазией разработчика.

В боковом меню размещаю ещё несколько кнопок для других программных продуктов (*на случай дальнейшей разработки*).

Для описания каждого программного продукта использую класс **FlowDocument**. Это позволяет мне размещать и форматировать содержимое потока

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

Лист

18

с помощью дополнительных возможностей работы с документами, таких как разбивка на страницы и столбцы. Данный класс обеспечивает надежную модель содержимого для дочернего содержимого. Дочерние элементы верхнего уровня, содержащиеся в, **FlowDocument** должны быть производными от **Block** (где могут содержаться один или несколько объектов).



**Рисунок №4 – Окно для учебных материалов и тестирований (внизу можно видеть кнопки: Поиск, Количество страниц, Масштабирование, Сохранить текст (в формате \*.xaml), Перезагрузить страницу, Печать, а также можно просмотреть Конспект по данной теме, перейти на сайты: КГА ПОУ «ВСК» и «Visual Studio», Открыть: «Проводник», «Реестр» и «Блокнот»)**

Изм.	Лист	№ докум.	Подпись	Дата	Лист	19
					<b>ВСК.090204.07.00.000 ПЗ</b>	

## Пространства имён

Пространство имён (*англ. namespace*) — некоторое множество, под которым подразумевается модель, абстрактное хранилище или окружение, созданное для логической группировки уникальных идентификаторов (*то есть имён*).

Пространство имён — это не более чем группа типов данных, но дающая тот эффект, что имена всех типов данных в пределах пространства имён автоматически снабжаются префиксом — названием пространства имён. Пространства имён можно вкладывать друг в друга.

Основное пространство имён, с которого нужно начинать знакомство с пространствами называется **System**. В нем содержится набор ключевых типов, с которыми любой разработчик .NET будет использовать. Создание функционального приложения на C# невозможно без добавления ссылки хотя бы на пространство имён **System**, поскольку все основные типы данных (**System.Int32**, **System.Boolean**) определены именно в нём.

Для подключения пространства имён в C# используется ключевое слово **«using»**. В частности, файл «**MainWindow.xaml.cs**» содержит следующие пространства имён:

```
using System; // обязательно  
using System.Collections.Generic; // интерфейсы и классы, определяющие  
// универсальные коллекции  
using System.ComponentModel; // классы, используемые для реализации  
// поведения компонентов и элементов управления  
using System.Windows;  
using System.Windows.Controls; // это элемент управления WPF  
using System.Windows.Data; // работа часов/дат  
using System.Windows.Forms; // работа с формами  
using System.Windows.Input; // операции ввода  
// все команды реализуют интерфейс System.Window.Input.ICommand:  
using System.Windows.Media; // всё что связано с мультимедиа
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

20

```

using System.Windows.Media.Imaging; // проигрывать-просматривать
using System.Windows.Media.Animation; // чтобы использовать данные из
параметров

using MessageBox = System.Windows.MessageBox; // определение команды
using System.Collections.ObjectModel; // подключаю пользователей для входа
using OpenQA.Selenium; // подключаю Web Driver's
using OpenQA.Selenium.Chrome; // у меня стоит хром (легко поменять)

// можно подключить любой из установленных на компьютере браузеров

using OpenQA.Selenium.Support.UI;
using System.IO; // для работы с файлами
using Microsoft.Win32; // для тестов (встроенных)

// для работы JavaScript-ов / для использования обязательно [ComVisible (true)]
using System.Runtime.InteropServices; // для поддержки СОМ-взаимодействия
и службы вызова платформы

using System.Windows.Threading; // для поддержки системы работы с
потоками Windows Presentation Foundation (WPF)

using Excel = Microsoft.Office.Interop.Excel; // подключение библиотеки Excel
using Word = Microsoft.Office.Interop.Access; // у меня 2019 офис
using DocumentFormat.OpenXml.Office2016.Drawing.ChartDrawing;
using System.Media; //подключаю пространство имен SoundPlayer
using Application = System.Windows.Forms.Application; // для реестра
using Math = System.Math; // стандартные математические функции
using System.Data; // работа с данными
using System.Text; // работа с текстом
using System.Xml; // работа с *.xml
using System.Xml.Linq;
using System.Xml.Serialization; // для взаимодействия с xml-файлами
using System.Xml.Configuration;
using System.Runtime.Serialization; // для сериализации класса
using System.Runtime.Serialization.Formatters;

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

21

```
using System.Runtime.Serialization.Formatters.Binary;
using System.Runtime.Versioning;
using AllLoader.userMembers; // вымышленные студенты
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Documents; // документы
using System.Windows.Shapes; // фигуры
using System.Collections; // коллекции
using AllLoader; // основное пространство имен
using Microsoft.Office.Interop.Excel; // для работы с офисом
using Microsoft.Office.Interop.Access;
using Microsoft.Office.Interop.Word;
using System.Reflection;
using System.Configuration; // для сохранения настроек программы
using System.Collections.Specialized;
using NeuroSpeech.XamlToPDF; // платно
using AllLoader.Tests; // для тестирований
using iText.Kernel.Pdf; // платное дополнение
using iText.Layout;
using iText.Layout.Element;
using iText.Layout.Properties;
using PdfSharp; // подключаю библиотеки для работы с PDF
using PdfSharp.Pdf;
using PdfSharp.Drawing;
using PdfSharp.Charting;
using PdfSharp.FONTS;
using System.Windows.Navigation; // навигация по окнам и формам
using System.IO.Packaging;
using System.Printing; // печать из программы
using System.Windows.Xps.Packaging;
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

22

```

using System.Windows.Xps; // работа с PDF
using System.Windows.Xps.Serialization;
using DotLiquid; // для конспектов (динамически формируемые)
using dotTemplate = DotLiquid.Template;
using System.Windows.Markup; // предоставляет типы, поддерживающие
XAML
using System.Windows.Forms.ComponentModel; // для запуска уже
скомпилированных проектов
using System.Diagnostics; // взаимодействие с системными процессами,
журналами
using System.Globalization; // языковые настройки

```

**Идентификатор**, определённый в пространстве имён, ассоциируется с этим пространством. Один и тот же идентификатор, может быть, независимо определён в нескольких пространствах. Таким образом, значение, связанное с идентификатором, определённым в одном пространстве имён, может иметь (или не иметь) такое же значение, как и такой же идентификатор, определённый в другом пространстве. Определено правило, которое обязывает указывать, к какому пространству имён принадлежит идентификатор (то есть его определение).

В программе «Оператор ЭВМ» основное пространство имен — «**AllLoader**»:

### namespace AllLoader

```

{ // добавляю диспетчер ресурсов, обеспечивающий удобный доступ к ресурсам,
// связанным с языком и региональными параметрами (Сборка: mscorelib.dll)
    [System.Runtime.InteropServices.ComVisible(true)]
    [System.Serializable]
    public static class GlobalizationExtensions { } // для работы с языками
    [System.Flags]
    public enum DragDropEffects { }
    // для возможности операций перетаскивания
    public enum CultureTypes { } // для работы с языками

```

Изм.	Лист	№ докум.	Подпись	Дата

```
public class ResourceManager { } // для работы с ресурсами
```

## Класс MainWindow

Это очень важный класс, который реализует главное окно, содержащее в себе типовые виджеты, необходимые большинству приложений, такие как меню, секции для панелей инструментов, рабочую область, строки состояния. Кроме этого, окно приложения имеет несколько элементов меню, которые располагаются ниже области заголовка окна.

```
public partial class MainWindow : System.Windows.Window
{
    // переменная для передачи языковых настроек между формами
    // базовое число пользователей
    // первоначальное число ноликов в номере (6 -> 5)
    public string forNumber = "00000";
    // массив для пользователей, которые зарегистрированы
    public string[,] nowUserMember = new string[1000000, 9];
    // для создания таблицы внутри (на экране)
    public bool Users = true;
    // для реализации входа в программу
    public System.Data.DataTable _dataTable;
    public System.Data.DataTable DataTable
    {
        // для обращения к таблице
        get { return _dataTable; }
        set { _dataTable = value; }
    }
    // в значении "true" можно входить по почте и паролю
    public bool forEnterLogin = true;
    // в значении "false" необходимо ввести данные для регистрации
    // использую созданный класс userMember
    // для создания таблицы внутри (в памяти программы)
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

24

```

        public System.Data.DataTable _tableUserMembers;
        public System.Data.DataTable tableUserMembers
        {
            get { return _tableUserMembers; }
            set { _tableUserMembers = value; }
        } // для добавления в список учащихся (1 человек)
        public ObservableCollection<userMember> userMembers = new
ObservableCollection<userMember>();
//internal List<ListUserMembers> NewUser { get => newUser; set => newUser = value; }

        // таймер для отложенного появления
        DispatcherTimer timer = new DispatcherTimer();
        SoundPlayer begin; // объявляю переменную типа SoundPlayer
        public string MG = ""; // для случайного выбора мальчиков-девочек
        public bool menInput = false; // если произойдет выбор мальчиков
        public bool girlInput = false; // если произойдет выбор девочек
        // для выбора 20-ти слушателей
        public short for20 = 0;

        // объявляю массив для работы с файлами xml из ресурсов проекта
        public bool[] enabledXmlFile = new bool[20] { true, true, true, true, true,
true, true, true, true, true, true, true, true, true, true, true, true, true };

        // после работы с каждым файлом значение будет меняться на "false"
        private List<ListUserMembers> newUser = new List<ListUserMembers>();
        // пример лицензионного кода iText.License_licenseKey ("MTEzM
UAzMTM4MmUzMjJlMzBJaWF3ZHF1ZHBMa1lvQmRCUHZIWThOSWhjNINSVkjx
VE92VGVmZ0F5akQ0PQ==");

        // создаю массив для случайных пользователей (мальчиков)
        public string[][] rndMenUser = { new string[] {"Иванов", "Петров", "Сидоров",
"Бутов", "Мукин", "Бычков", "Цаплин", "Завьялов", "Трофимов", "Оглоблин"},

new string[] {"Игорь", "Сергей", "Григорий", "Василий", "Федор", "Емельян",
"Афанасий", "Валерий", "Дмитрий", "Тимофей"}, // имена
    
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

25

```

        new string[] {"Алексеевич", "Александрович", "Борисович", "Валерьевич",
"Макарович",     "Свойских",     "Васильевич",     "Петрович",     "Глебович",
"Простаковский"}, // отчества

        new string[] {"ivanov_ia@gmail.com",      "petrov-SA@mail.ru",
"GBSidorov@yandex.ru",          "butvv@yahoo.com",       "home240694@bk.ru",
"web_QW@list.ru",                "dargzell@icq.ru",       "fro-prim@rambler.ru",
"malodchik@gmail.com", "izopren@yandex.ru"}, // email-ящики

        new string[] {"+79087035468",    "89021206619",    "84232215846",
"+79135501254",    "79234542112",    "89520123002",    "89448010292",    "89083274242",
"89070035566",    "+79665012824"}, // сотовые номера

        new string[] {"2","4","1","2","3","2","1","4","3","3"}, // номер курса и группы

        new string[] {"У-12-241-о", "У-14-252-о", "У-22-353-о", "У-21-401-о", "У-12-
241-о", "У-14-252-о", "У-22-353-о", "У-21-401-о", "У-14-252-о", "У-22-353-о"},

        new string[] {"qwerty0",   "qwerty0",   "qwerty0",   "qwerty0",   "qwerty0",
"qwerty0", "qwerty0", "qwerty0", "qwerty0", "qwerty0"} // пароль на вход (иллюзия);

        // также создаю массив для случайных пользователей (девочек)

public string[][] rndWomanUser = {

        new string[] {"Самойлова", "Иванова", "Зимина", "Ершова", "Беспалова",
"Блинова", "Некрасова", "Лукина", "Носкова", "Виноградова"},

        new string[] {"Алия", "Женевьев", "Сусанна", "Джульетта", "Людмила",
"Лидия", "Грета", "Устинья", "Ульяна", "Мелиана"},

        new string[] {"Артемовна", "Николаевна", "Константиновна", "Тимуровна",
"Натановна", "Федоровна", "Юлиановна", "Семеновна", "Иосифовна", "Якововна"},

        new string[] {"alyasm@bk.ru", "jn-ivanova@yahoo.com", "zimasus@list.ru",
"ershova_dt@gmail.com", "bespalove@outlook.com", "blidiya@rambler.ru", "nek-
gret@mail.ru", "lukus_vl@ya.ru", "uliana-NU@list.ru", "vinomilo@ukr.net"},

        new string[] {"+79330619203",    "89088718980",    "79120682905",
"+79162756901", "89308392539", "+79469334507", "+79512845188", "89458878846",
"+79465561779", "89287911245"},

        new string[] {"2", "1", "4", "1", "1", "2", "3", "1", "2", "2"},


```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

26

<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>

ВСК.090204.07.00.000 ПЗ

Лист

27

```

"rv7bp@gmail.com", "93@outlook.com",    "er@gmail.com",      "o0my@gmail.com",
"715qy08@gmail.com", "vubx0t@mail.ru", "wnhborq@outlook.com", "gq@yandex.ru",
"ic0pu@outlook.com", "o7khr@yandex.ru", "2shlaq@outlook.com" };

public string[] enumEmail2 = { "wrts90puk@yandex.ru", "k8sjebg1y@mail.ru",
"u7yhwf1vb@mail.ru", "f@outlook.com",..... }; // и т.д.

public string[] enumEmail3 = { ..... }; // и т.д.

// создаю списки (листы) для 20 пользователей

public List<string> Phone1 = new List<string> { "89080256466",
"+79047725849", "79657077312", "89020534095", "79246399040", "+79645922441",
"89020730391", "89023982722", "+79639157354", "79022586366", "89086560127",
"79642660052", "89024114747", "79024511313", "79247316365",
"+79245586521", "89029129078", "+79638284422", "+79243875961", "79632091034" };

public List<string> Phone2 = new List<string> { ..... // и т.д.};

public List<string> Phone3 = new List<string> { ..... // и т.д.};

public List<short> Course1 = new List<short> { 2, 3, 1, 1, 2, 4, 3, 2, 2, 3, 3, 1, 4,
1, 1, 1, 2, 2, 2, 3};

public List<short> Course2 = new List<short> { 1, 2, 3, 2, 2, 1, 1, 1, 2, 3, 4, 1, 4,
2, 1, 3, 2, 4, 3, 1 }; // и т.д.

public List<short> Course3 = new List<short> { 3, 2, 2, 3, 1, 2, 1, 3, 3, 1, 2, 2, 1,
3, 3, 2, 1, 3, 3, 2 };

public List<string> Group1 = new List<string> { "У-11-211-о", "У-21-214-о",
"У-12-232-о", "У-23-221-о", "У-13-213-з", "У-11-211-о", "У-21-214-о", "У-12-232-о",
"У-23-221-о", "У-14-128-о", "У-10-214-о", "У-13-124-о", "У-14-202-о", "У-20-201-о",
"У-24-120-о", "У-12-132-о", "У-22-202-о", "У-11-102-о", "У-21-213-з", "У-13-212-о" };

public List<string> Group2 = new List<string> { ..... // и т.д.};

public List<string> Group3 = new List<string> { ..... // и т.д.};

// задаю пароль "по умолчанию" (одинаковый)

public string defaultPASS = "qwerty0";

// выборка данных из: SurName1, FirstName1, MiddleName1, enumEmail1, Phone1,
Course1, Group1, defaultPASS

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

28

<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>

ВСК.090204.07.00.000 ПЗ

## Лист

29

## Заставка и переменные для главного окна приложения

Перед тем как пользователь сможет увидеть главное окно, я ставлю заставку для данного проекта — это два файла: **SplashScreenWindow.xaml (1)** и **SplashScreenWindow.xaml.cs (2)**.

Делаю начальную заставку в виде окна (1):

```
<Window x:Class="AllLoader.SplashScreenWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:AllLoader" Height="1000" Width="720"
    WindowStartupLocation="CenterScreen" WindowStyle="None"
    AllowsTransparency="True" >
    <Grid> <!-- размещаю "ProgressBar" под картинкой -->
        <ProgressBar x:Name="progressBar" />
    <!-- рисунок создаю в Photoshop (2020) -->
    <Image Source="Images>Loading\Screen.png" Margin="0,40,0,100"></Image>
    <!-- текст который “проявляется” при прохождении "progressBar-a" -->
    <TextBlock HorizontalAlignment="Center" FontStyle="Normal" Margin="0,895,0,0"
        Height="66" FontWeight="ExtraBold" Foreground="WhiteSmoke" FontSize="50">
        Loading "Operator IBM"...
    </TextBlock>
</Grid> </Window>
```

И для активации данного окна (2):

namespace AllLoader

```
{    public partial class SplashScreenWindow : Window
    {
        public SplashScreenWindow()
        {
            InitializeComponent();
        }
        public double Progress
```

Изм.	Лист	№ докум.	Подпись	Дата

```

    { // для работы "ProgressBar"
        get { return progressBar.Value; }
        set { progressBar.Value = value; }     }     }
}

```

Чтобы первоначально было запущено именно это окно (*чтобы сперва загружалась именно заставка, а не “MainWindow”*) — его нужно указать в файле **App.xaml.cs**:

```

protected override void OnStartup(StartupEventArgs e)
{
    // как только происходит запуск приложения
    base.OnStartup(e);

    // активирую заставку и становлю ее в качестве главного окна приложения
    var splashScreen = new SplashScreenWindow();
    this.MainWindow = splashScreen; // т.е. главное окно это заставка
    splashScreen.Show(); // и показываю её на экране

    // заставка возможна только в другом потоке (нужно другое окно)
    Task.Factory.StartNew(() =>
    {
        // работу “прогесбара” осуществляю по таймеру (частями), чтобы
        // сообщать о ходе загрузки (100 частей полосы загрузки)

        for (int i = 1; i <= 100; i++) // закрашивается зелёным цветом
        {
            // моделирую часть выполняемой работы (1 часть – время-мс)
            System.Threading.Thread.Sleep(30);

            // в потоке пользовательского интерфейса, использую "Диспетчер",
            // связанный с заставкой, чтобы обновлять индикатор выполнения
            splashScreen.Dispatcher.Invoke(() => splashScreen.Progress = i);
        }

        // после окончания создаю и отображаю основное окно
        this.Dispatcher.Invoke(() =>
        {
            // устанавливаю главное окно приложения и закрываю заставку
            var mainWindow = new MainWindow();
            this.MainWindow = mainWindow;
            mainWindow.Show(); // показываю главное окно
        });
    });
}

```

```
splashScreen.Close(); // окно заставки закрываю
});});}
```

**App.xaml** — это XAML-файл определяющий приложение WPF и все ресурсы приложения. Этот файл также используется для указания пользовательского интерфейса, в данном случае заставка **SplashScreenWindow.xaml**, которая автоматически отображается при запуске приложения.

После создания заставки **создаю ряд переменных**, которые необходимы для дальнейшей работы данной программы:

**public MainWindow()**

```
{ // создаю рабочие каталоги
    string wordPath = @"C:\OperatorIBM";
    string subWordPath = @"OperatorIBM\Word";
    // проверяю если ли он уже есть
    DirectoryInfo dirWInfo = new DirectoryInfo(wordPath);
    if (!dirWInfo.Exists)
    { // тогда создаю его
        dirWInfo.Create();
    }
    dirWInfo.CreateSubdirectory(subWordPath);

    // для установки счетчиков
id0=id1=id2=id3=id4=id5=id6=id7=id8=id9=id10=id11=id12=id13=id14=id15=0;
    string excelPath = @"C:\OperatorIBM";
    string subExcelPath = @"OperatorIBM\Excel";
    // проверяю если ли он уже есть
    DirectoryInfo dirEInfo = new DirectoryInfo(excelPath);
    if (!dirEInfo.Exists)
    { // раз нет - создаю
        dirEInfo.Create();
    }
    dirEInfo.CreateSubdirectory(subExcelPath);
```

Изм.	Лист	№ докум.	Подпись	Дата

```

        // создаю каталог слушателей (студентов)

        string generalPath = @"C:\OperatorIBM";
        string subPath = @"OperatorIBM\baseMembers";
        DirectoryInfo dirInfo = new DirectoryInfo(generalPath);
        if (!dirInfo.Exists)
        {
            // создаю каталог для зарегистрировавшихся
            dirInfo.Create();
        }
        // если каталог есть - добавляю каталог для пользователей
        dirInfo.CreateSubdirectory(subPath);

InitializeComponent(); // включает элементы управления (по умолчанию)

        if (Properties.Settings.Default.setLang != null)
        {
            // передача языковых настроек
Properties.Settings.Default.xLang = Properties.Settings.Default.setLang;
Properties.Settings.Default.xTableLang = Properties.Settings.Default.setLang;
        }
        // сохранение настроек

loadDataMembersScreen.DataContext = ConfigurationManager.AppSettings .
Get("baseFilePath");           // установка таймера

        timer = new DispatcherTimer();

        // завожу лист куда сразу добавляю (сперва лишь) 4-х человек

        newUser.Add(new ListUserMembers() { ID = "1", SurName = "Федоренко",
FirstName = "Леонид", MiddleName = "Александрович", Email =
"flaneed@gmail.com", Phone = "89644444473", Course = "5", Group = "У-14-237-з",
PASS = "qwerty1" });
        newUser.Add(new ListUserMembers() { ID = "2", SurName =
"Толок", FirstName = "Ирина", MiddleName = "Евгеньевна", Email =
"distanc.tolok.20@gmail.com", Phone = "*****", Course = "0", Group = "0-00-
VSK-0", PASS = "qwerty2" });
        newUser.Add(new ListUserMembers() { ID = "3",
SurName = "Лысенко", FirstName = "Иван", MiddleName = "Александрович",
Email = "bragili@ya.ru", Phone = "79502814792", Course = "0", Group = "0-00-VSK-
0", PASS = "qwerty3" });


```

```
    newUser.Add(new ListUserMembers() { ID = "4", SurName = "Овчинникова",  
FirstName = "Лариса", MiddleName = "Александровна", Email =  
"zaochvsk@yandex.ru", Phone = "79242337133", Course = "0", Group = "0-00-VSK-0",  
PASS = "qwerty4" });
```

// впоследствии добавляю всех учителей кого тогда нашел на сайте ВСК

// придумываю корпоративный ящик для каждого сотрудника (он же  
"логин") это "ФИО@сайтВУЗа.ru"

```
    newUser.Add(new ListUserMembers() { ID = "5", SurName = "Андреев",  
FirstName = "Вячеслав", MiddleName = "Владимирович", Email =  
"andreev_vv@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-  
0", PASS = "qwerty5" });
```

```
    newUser.Add(new ListUserMembers() { ID = "6", SurName = "Антохина",  
FirstName = "Светлана", MiddleName = "Павловна", Email =  
"antohina_sp@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-  
0", PASS = "qwerty6" });
```

```
    newUser.Add(new ListUserMembers() { ID = "7", SurName = "Барвинок",  
FirstName = "Людмила", MiddleName = "Сергеевна", Email =  
"barvinok_ls@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-  
0", PASS = "qwerty7" });
```

```
    newUser.Add(new ListUserMembers() { ID = "8", SurName = "Буртасов",  
FirstName = "Александр", MiddleName = "Иванович", Email =  
"burtasov_ai@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-  
0", PASS = "qwerty8" });
```

```
    newUser.Add(new ListUserMembers() { ID = "9", SurName = "Бабенко",  
FirstName = "Евгений", MiddleName = "Николаевич", Email = "babenko50@bk.ru",  
Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty9" });
```

```
    newUser.Add(new ListUserMembers() { ID = "10", SurName = "Байкин",  
FirstName = "Сергей", MiddleName = "Александрович", Email =  
"baikin_sa@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0",  
PASS = "qwerty10" });
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

34

```
    newUser.Add(new ListUserMembers() { ID = "11", SurName = "Герман",
FirstName = "Елена", MiddleName = "Дмитриевна", Email =
"german_ed@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-
0", PASS = "qwerty11" });


```

```
    newUser.Add(new ListUserMembers() { ID = "12", SurName = "Головин",
FirstName = "Дмитрий", MiddleName = "Иванович", Email =
"golovin_di@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-
0", PASS = "qwerty12" });


```

```
    newUser.Add(new ListUserMembers() { ID = "13", SurName = "Гулевич",
FirstName = "Лариса", MiddleName = "Сергеевна", Email =
"gulevich_ls@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-
0", PASS = "qwerty13" });


```

```
    newUser.Add(new ListUserMembers() { ID = "14", SurName = "Даниленко",
FirstName = "Юлия", MiddleName = "Владимировна", Email =
"danilenko_uv@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-
VSK-0", PASS = "qwerty14" });


```

```
    newUser.Add(new ListUserMembers() { ID = "15", SurName = "Данилова",
FirstName = "Людмила", MiddleName = "Николаевна", Email =
"danilova_ln@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-
0", PASS = "qwerty15" });


```

```
    newUser.Add(new ListUserMembers() { ID = "16", SurName = "Кононова",
FirstName = "Ольга", MiddleName = "Владимировна", Email =
"kononova_ov@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-
VSK-0", PASS = "qwerty16" });


```

```
    newUser.Add(new ListUserMembers() { ID = "17", SurName = "Котенко",
FirstName = "Юлия", MiddleName = "Сергеевна", Email = "kotenko_us@vstehn.ru",
Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty17" });


```

```
    newUser.Add(new ListUserMembers() { ID = "18", SurName = "Крысанова",
FirstName = "Надежда", MiddleName = "Александровна", Email = "krysanova_nd@vstehn.ru",
Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty18" });


```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

35

```

"kresanova_na@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty18" });

newUser.Add(new ListUserMembers() { ID = "19", SurName = "Красикова", FirstName = "Елена", MiddleName = "Викторовна", Email = "krasikova_ev@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty19" });

newUser.Add(new ListUserMembers() { ID = "20", SurName = "Крюков", FirstName = "Алексей", MiddleName = "Алексеевич", Email = "kryukov_aa@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty20" });

newUser.Add(new ListUserMembers() { ID = "21", SurName = "Лебедев", FirstName = "Игорь", MiddleName = "Владимирович", Email = "lebedev_iv@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty21" });

newUser.Add(new ListUserMembers() { ID = "22", SurName = "Литошенко", FirstName = "Денис", MiddleName = "Александрович", Email = "litoshenko_da@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty22" });

newUser.Add(new ListUserMembers() { ID = "23", SurName = "Лобова", FirstName = "Татьяна", MiddleName = "Жановна", Email = "lobova_tj@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty23" });

newUser.Add(new ListUserMembers() { ID = "24", SurName = "Лукина", FirstName = "Кира", MiddleName = "Владимировна", Email = "lukina_kv@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty24" });

newUser.Add(new ListUserMembers() { ID = "26", SurName = "Миргеев", FirstName = "Андрей", MiddleName = "Александрович", Email = "mirgeev_aa@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty26" });

```

Изм.	Лист	№ докум.	Подпись	Дата

*ВСК.090204.07.00.000 ПЗ*

*Лист*

*36*

newUser.Add(new ListUserMembers() { ID = "27", SurName = "**Миллер**", FirstName = "**Наталья**", MiddleName = "**Валерьевна**", Email = "[miller\\_nv@vstehn.ru](mailto:miller_nv@vstehn.ru)", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "**qwerty27**" });

newUser.Add(new ListUserMembers() { ID = "28", SurName = "**Матвиенко**", FirstName = "**Владимир**", MiddleName = "**Александрович**", Email = "[matvienko\\_va@vstehn.ru](mailto:matvienko_va@vstehn.ru)", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "**qwerty28**" });

newUser.Add(new ListUserMembers() { ID = "29", SurName = "**Николаев**", FirstName = "**Сергей**", MiddleName = "**Олегович**", Email = "[nikolaev\\_so@vstehn.ru](mailto:nikolaev_so@vstehn.ru)", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "**qwerty29**" });

newUser.Add(new ListUserMembers() { ID = "30", SurName = "**Паскарь**", FirstName = "**Ирина**", MiddleName = "**Михайловна**", Email = "[paskar\\_im@vstehn.ru](mailto:paskar_im@vstehn.ru)", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "**qwerty30**" });

newUser.Add(new ListUserMembers() { ID = "31", SurName = "**Пивоварцева**", FirstName = "**Ирина**", MiddleName = "**Владимировна**", Email = "[pivovarceva\\_iv@vstehn.ru](mailto:pivovarceva_iv@vstehn.ru)", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "**qwerty31**" });

newUser.Add(new ListUserMembers() { ID = "32", SurName = "**Пошивайло**", FirstName = "**Валентина**", MiddleName = "**Степановна**", Email = "[poshivailo\\_vs@vstehn.ru](mailto:poshivailo_vs@vstehn.ru)", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "**qwerty32**" });

newUser.Add(new ListUserMembers() { ID = "33", SurName = "**Рубан**", FirstName = "**Нина**", MiddleName = "**Ивановна**", Email = "[ruban\\_ni@vstehn.ru](mailto:ruban_ni@vstehn.ru)", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "**qwerty33**" });

newUser.Add(new ListUserMembers() { ID = "34", SurName = "**Родионов**", FirstName = "**Руслан**", MiddleName = "**Андреевич**", Email = "

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

37

"radionov\_ra@vstehn.ru", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty34" });

 newUser.Add(new ListUserMembers() { ID = "35", SurName = "Сновидов", FirstName = "Евгений", MiddleName = "Борисович", Email = "snovidov\_eb@vstehn.ru", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty35" });

 newUser.Add(new ListUserMembers() { ID = "36", SurName = "Степанова", FirstName = "Ирина", MiddleName = "Тимофеевна", Email = "stepanova\_it@vstehn.ru", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty36" });

 newUser.Add(new ListUserMembers() { ID = "37", SurName = "Сергеева", FirstName = "Людмила", MiddleName = "Николаевна", Email = "sergeeva\_ln@vstehn.ru", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty37" });

 newUser.Add(new ListUserMembers() { ID = "38", SurName = "Стрембицкая", FirstName = "Галина", MiddleName = "Семеновна", Email = "strembichkaya\_gs@vstehn.ru", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty38" });

 newUser.Add(new ListUserMembers() { ID = "39", SurName = "Сизова", FirstName = "Наталья", MiddleName = "Васильевна", Email = "sizova\_nv@vstehn.ru", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty39" });

 newUser.Add(new ListUserMembers() { ID = "40", SurName = "Титко", FirstName = "Александр", MiddleName = "Евгеньевич", Email = "titko\_ae@vstehn.ru", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty40" });

 newUser.Add(new ListUserMembers() { ID = "41", SurName = "Ускова", FirstName = "Антонина", MiddleName = "Николаевна", Email = "uskova\_an@vstehn.ru", Phone = "\*\*\*\*\*", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty41" });

Изм.	Лист	№ докум.	Подпись	Дата

VSK.090204.07.00.000 ПЗ

Лист

38

```
    newUser.Add(new ListUserMembers() { ID = "42", SurName = "Фалеева",  
FirstName = "Валентина", MiddleName = "Николаевна", Email =  
"faleeva_vn@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-  
0", PASS = "qwerty42" });
```

```
    newUser.Add(new ListUserMembers() { ID = "43", SurName = "Фенюков",  
FirstName = "Валерий", MiddleName = "Васильевич", Email =  
"fenukov_vv@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-  
0", PASS = "qwerty43" });
```

```
    newUser.Add(new ListUserMembers() { ID = "44", SurName = "Шмонин",  
FirstName = "Игорь", MiddleName = "Викторович", Email =  
"shmonin_iv@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-  
0", PASS = "qwerty44" });
```

```
    newUser.Add(new ListUserMembers() { ID = "45", SurName = "Яровая",  
FirstName = "Ольга", MiddleName = "Васильевна", Email =  
"yarovaya_ov@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-  
VSK-0", PASS = "qwerty45" });
```

// установленные значения сразу можно выводить на экран

```
ListUserMembers.ItemsSource = newUser;
```

// чтобы вывести(показать) список на экран

```
CollectionView view = (CollectionView) CollectionViewSource.GetDefaultView  
(ListUserMembers.ItemsSource);
```

// сортировку делаю по номеру

```
view.SortDescriptions.Add (new SortDescription ("ID", ListSortDirection.  
Ascending));
```

```
    userMembers.Add(new userMember() { ID = "1", SurName = "Федоренко",  
FirstName = "Леонид", MiddleName = "Александрович", Email =  
"flaneed@gmail.com", Phone = "89644444473", Course = "5", Group = "У-14-237-3",  
PASS = "qwerty1" });
```

```
    userMembers.Add(new userMember() { ID = "2", SurName = "Толок",  
FirstName = "Ирина", MiddleName = "Евгеньевна", Email =
```

Изм.	Лист	№ докум.	Подпись	Дата

*ВСК.090204.07.00.000 ПЗ*

Лист

39

```

"distanc.tolok.20@gmail.com", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty2" });

    userMembers.Add(new userMember() { ID = "3", SurName = "Лысенко",
FirstName = "Иван", MiddleName = "Александрович", Email = "bragili@ya.ru",
Phone = "79502814792", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty3" });

    userMembers.Add(new userMember() { ID = "4", SurName = "Овчинникова",
FirstName = "Лариса", MiddleName = "Александровна", Email =
"zaochvsk@yandex.ru", Phone = "79242337133", Course = "0", Group = "0-00-VSK-0",
PASS = "qwerty4" });

    userMembers.Add( // и т.д. по образу и подобию (с 1 до 45)

        userMembers.Add(new userMember() { ID = "45", SurName = "Яровая",
FirstName = "Ольга", MiddleName = "Васильевна", Email =
"yarovaya_ov@vstehn.ru", Phone = "*****", Course = "0", Group = "0-00-VSK-0", PASS = "qwerty45" });

        // переменная - экземпляр класса SoundPlayer

        begin = new SoundPlayer(); // добавляю звуки в ресурсы проекта
        // беру из ресурсов звуковой файл (в соответствии с языком)
        begin.Stream = Properties.Resources.rusOperatorIBM;
        begin.Play(); // проигрываю
        // для работы аналоговых часов на форме (стрелка секунды)

Storyboard lineseconds = (Storyboard)linesecond.FindResource("for_seconds");
lineseconds.Begin();
lineseconds.Seek(new TimeSpan(0, 0, 0, DateTime.Now.Second, 0));
        // для работы аналоговых часов на форме (стрелка минуты)

Storyboard lineminutes = (Storyboard)lineminute.FindResource("for_minutes");
lineminutes.Begin();
lineminutes.Seek(new TimeSpan(0, 0, 0, DateTime.Now.Minute,
DateTime.Now.Second, 0));
        // для работы аналоговых часов на форме (стрелка часы)

Storyboard linehours = (Storyboard)linehour.FindResource("for_hours");

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

40

```

linehours.Begin();

linehours.Seek(new TimeSpan(DateTime.Now.Hour, DateTime.Now.Hour,
DateTime.Now.Minute, DateTime.Now.Second, 0));

// организую внутреннюю базу данных (без привязки к SQL/Access и т.д.)

Loaded += MainWindow_Loaded;

// заношу первые строки в базу "userMembers"

Topmost = false;

// разрешаю основному окну быть не поверх остальных!!!

butLeftEnterRound.ToolTip = "Чтобы открыть форму входа:\r\nНажмите красную
кнопку со стрелкой"; // подсказки при наведении

butRightEnterRound.ToolTip = "Чтобы открыть форму регистрации:\r\nНажмите
зелёную кнопку со стрелкой";

butEnterRight.ToolTip = "Сдвинуть влево / есть аккаунта";

butEnterLeft.ToolTip = "Сдвинуть вправо / нет аккаунт";

// создаю привязку команды

CommandBinding commandBinding = new CommandBinding();

// устанавливаю команду

commandBinding.Command = ApplicationCommands.Help;

// устанавливаю метод, который будет выполняться при вызове команды

commandBinding.Executed += CommandBinding_Executed;

// добавляю привязку к коллекции привязок элемента Button

helpButton.CommandBindings.Add(commandBinding);

// в самом начале доступны поля для уже зарегистрированных пользователей
// достаточно ввести почту и пароль, указанные выше

// свободная лицензия на подключение к различным документам

// это для GemBox - ComponentInfo.SetLicense("FREE-LIMITED-KEY");

newSurname.IsEnabled = false; // фамилия

newName.IsEnabled = false; // имя

newMiddleName.IsEnabled = false; // отчество

newTelefon.IsEnabled = false; // телефон

```

Изм.	Лист	№ докум.	Подпись	Дата

```

newCourse.IsEnabled = false; // курс
newGroup.IsEnabled = false; // группа
// доступность кнопок (входа/регистрации)
butEnterRight.IsEnabled = false;
butEnterRight.Content = "X"; // если кнопка не доступна
butEnterLeft.IsEnabled = true;
// так как элемент управления "WebBrowser" по умолчанию
// работает в режиме совместимости с Internet Explorer 7 версии
// пытаюсь переключить на более новые версии - для поддержки скриптов
// завожу и вызываю метод: "forWebBrowserLevel"
// в котором обращаюсь к веткам реестра
// MG = "man"; - чтобы заполнить Grid мальчиками
// MG = "girl"; - чтобы заполнить Grid девочками
// запуск PopUp и скрытие по таймеру (30 сек)
mouseRightButtonDown(); // верхний левый угол }

```

### **Создание стилей оформления и поведения в App.xaml**

Платформа Windows Presentation Foundation (**WPF**) позволяет создавать приложения для настольных систем Windows с привлекательным пользовательским интерфейсом.

В основе WPF лежит независимый от разрешения векторный модуль визуализации, использующий возможности современного графического оборудования. Возможности этого модуля расширяются с помощью комплексного набора функций разработки приложений, которые включают в себя язык **XAML**, элементы управления, привязку к данным, макет, двумерную и трехмерную графику, анимацию, стили, шаблоны, документы, мультимедиа, текст и типографические функции. WPF является частью .NET, поэтому можно создавать приложения, включающие другие элементы **.NET API**.

WPF позволяет разрабатывать приложения, используя как разметку, так и код программной части, что привычно для разработчиков на ASP.NET. Разметка

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

**Лист**

**42**

XAML обычно используется для определения внешнего вида приложения, а управляемые языки программирования (код программной части) — для реализации его поведения. Такое разделение внешнего вида и поведения имеет ряд преимуществ:

1. Затраты на разработку и обслуживание снижаются, так как разметка, определяющая внешний вид, не связана тесно с кодом, обуславливающим поведение.
2. Повышается эффективность разработки, так как дизайнеры, занимающиеся внешним видом приложения, могут работать параллельно с разработчиками, реализующими поведение приложения.
3. Глобализация и локализация приложений WPF упрощена.

Платформа WPF предоставляет широкий, гибкий и масштабируемый набор графических функций, который обладает перечисленными ниже преимуществами.

**Независимость графики от разрешения и устройства.** Основной единицей измерения в графической системе WPF является аппаратно-независимый пиксель, размер которого составляет 1/96 дюйма вне зависимости от разрешения экрана. Это создает основу для независимой от разрешения и аппаратной платформы отрисовки. Каждый аппаратно-независимый пиксель автоматически масштабируется в соответствии с заданным в системе количеством точек на дюйм (DPI).

**Повышение точности.** Система координат WPF основана на числах двойной точности с плавающей запятой, а не числах одинарной точности. Значения преобразования и прозрачности также выражаются числами двойной точности. Платформа WPF также поддерживает широкую цветовую палитру (scRGB) и имеет встроенную поддержку управления входными данными из разных цветовых схем.

**Расширенная поддержка графики и анимации.** Платформа WPF упрощает программирование графики, автоматически управляя анимированными сценами. Вам не нужно беспокоиться об обработке сцен, циклах отрисовки и билинейной интерполяции. Кроме того, WPF обеспечивает поддержку проверки попадания и полную поддержку альфа-версии компоновки.

Изм.	Лист	№ докум.	Подпись	Дата

**Аппаратное ускорение.** Система графики WPF использует возможности графического оборудования, чтобы снизить нагрузку на ЦП.

**Стили** позволяют разработчикам и дизайнерам стандартизировать внешний вид своего продукта. Платформа WPF предоставляет строгую модель стилей, в основе которой лежит элемент **Style**.

**Стилизация** и использование шаблонов Windows Presentation Foundation (WPF) относятся к набору возможностей, которые позволяют разработчикам и дизайнерам создавать визуально привлекательные эффекты и согласованный внешний вид своих продуктов. При настройке внешнего вида приложения необходима строгая модель стилизации и шаблонов, обеспечивающая обслуживание и совместное использование внешнего вида в приложениях и между ними. WPF предоставляет такую модель.

Еще одной возможностью модели стилизации WPF является разделение представления и логики. Дизайнеры могут создавать внешний вид приложения только с помощью **XAML** в то же самое время, когда разработчики работают над логикой программы, используя языки **C#** или **Visual Basic**.

Элемент **Style** можно рассматривать как удобный способ применения набора значений свойств к нескольким элементам. Стиль можно использовать для любого элемента, производного от *FrameworkElement* или *FrameworkContentElement*, например **Window** или **Button**.

Чаще всего стиль объявляется как ресурс в разделе **Resources** файла **XAML**. Так как стили являются ресурсами, для них действуют те же правила определения области, что и для всех других ресурсов. Проще говоря, то, где вы объявляете стиль, влияет на то, где этот стиль может быть применен. Например, если объявить стиль в корневом элементе файла XAML определения приложения, стиль может использоваться в любом месте приложения.

В файле **App.xaml** размещают следующие стили (указываю только часть):

<!-- здесь определяю блок **Application.Resources** для приложения -->

```
<Style x:Key="StandardTextBlockStyle" TargetType="TextBlock">
    <Setter Property="FontSize" Value="22" />
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

44

```

<Setter Property="Foreground" Value="White" />

</Style>      <!-- отдельный (основной) стиль, чтобы кнопки не менялись от
добавления новых стилей --&gt;

&lt;Style x:Key="someStyle" TargetType="Button"&gt;
    &lt;Setter Property="Opacity" Value="1" /&gt;
    &lt;Setter Property="HorizontalAlignment" Value="Center" /&gt;
    &lt;Setter Property="VerticalAlignment" Value="Center" /&gt;
&lt;/Style&gt;

<!-- убираю границу для кнопки "clear" --&gt;

&lt;Style x:Key="clearButton" TargetType="Button" &gt;
    &lt;Setter Property="Opacity" Value="1" /&gt;
    &lt;Setter Property="HorizontalAlignment" Value="Center" /&gt;
    &lt;Setter Property="VerticalAlignment" Value="Center" /&gt;
    &lt;Setter Property="Template"&gt;
        &lt;Setter.Value&gt;
            &lt;ControlTemplate TargetType="Button"&gt;
&lt;Border Background="{TemplateBinding Background}" BorderBrush="Transparent"
BorderThickness="0" CornerRadius="30"&gt;
&lt;ContentPresenter HorizontalAlignment="Center" VerticalAlignment="Center" /&gt;
            &lt;/Border&gt;
        &lt;/ControlTemplate&gt;
        &lt;/Setter.Value&gt;    &lt;/Setter&gt;
    &lt;/Style&gt;

<!-- в данном стиле убираю границу для кнопки использую "BaseOn" --&gt;

&lt;Style x:Key="gameTTT" TargetType="Button" BasedOn="{StaticResource
DefaultButton}"&gt;
    &lt;Setter Property="Opacity" Value="1" /&gt;
    &lt;Setter Property="HorizontalAlignment" Value="Center" /&gt;
    &lt;Setter Property="VerticalAlignment" Value="Center" /&gt;
    &lt;Setter Property="Template"&gt;
</pre>

```

Изм.	Лист	№ докум.	Подпись	Дата

```

<Setter.Value>
    <ControlTemplate TargetType="Button">
        <Border Background="{TemplateBinding Background}" BorderBrush="#F700FC"
            BorderThickness="1" CornerRadius="20">
            <ContentPresenter HorizontalAlignment="Center" VerticalAlignment="Center" />
        </Border>      </ControlTemplate>      </Setter.Value>      </Setter>
    </Style>

<!-- для кнопок внизу -->
<Style x:Key="butDown" TargetType="Button">
    <Setter Property="Opacity" Value="1" />
    <Setter Property="BorderThickness" Value="1" />
    <Setter Property="BorderBrush" Value="Yellow" />
</Style>

<!-- отдельный (основной) стиль, для тех кто вошёл -->
<Style x:Key="afterLoginStyle" TargetType="Button">
    <Setter Property="Opacity" Value="1" />
    <Setter Property="HorizontalAlignment" Value="Center" />
    <Setter Property="VerticalAlignment" Value="Center" />
</Style>

<!-- отдельный стиль, для таблички с пользователями -->
<Style x:Key="gridMembers" TargetType="DataGrid">
    <Setter Property="Foreground" Value="White" />
    <Setter Property="Background" Value="Blue" />
</Style>

<!-- определяю стиль «градиента» для кнопок -->
<LinearGradientBrush x:Key="WhiteBlueGradient" StartPoint="0,0" EndPoint="1,1">
    <GradientStop Offset="0" Color="#0020BB" />
    <GradientStop Offset="0.5" Color="White" />
    <GradientStop Offset="1" Color="#0020BB" />
</LinearGradientBrush>

```

Изм.	Лист	№ докум.	Подпись	Дата

```

<!-- определяю стиль для «круглых» кнопок -->
<Style x:Key="RoundButtons" TargetType="Button">
    <Setter Property="Background" Value="DodgerBlue" />
    <Setter Property="Foreground" Value="White" />
    <Setter Property="Template">
        <Setter.Value>
            <ControlTemplate TargetType="Button">
                <Border Background="{TemplateBinding Background}" BorderBrush="Black"
                    BorderThickness="2" CornerRadius="20">
                    <ContentPresenter HorizontalAlignment="Center" VerticalAlignment="Center" />
                </Border> </ControlTemplate> </Setter.Value> </Setter>
        </Setter>
    </Style>
<!-- округлые углы у прямоугольника -->
<Style x:Key="RoundRectangle" TargetType="Rectangle">
    <Setter Property="Opacity" Value="1" />
</Style>
<!-- определяю стиль для текстовых блоков -->
<Style x:Key="textBlockRoundBorder" TargetType="TextBlock">
    <Setter Property="Background" Value="White" />
    <Setter Property="Foreground" Value="White" />
</Style> <!-- чтобы не писать для каждой кнопки отдельный стиль
задаю универсальный "DefaultButton" -->
<Style BasedOn="{StaticResource DefaultButton}"
    TargetType="{x:Type Button}" />
</ResourceDictionary> </Application.Resources>
</Application>

```

Изм.	Лист	№ докум.	Подпись	Дата

## Стили в MainWindow.xaml

```
<Window.Resources>

<ResourceDictionary>           <!-- скачиваю анимированные курсоры -->
    <!-- показываю анимацию курсора при наведении на кнопки -->
    <Button x:Key="forHelpButton" Cursor="Cursors/HelpMainWindow.ani"/>
    <Button x:Key="forButEnterRight" Cursor="Cursors/rightbutton.ani"/>
    <Button x:Key="forButEnter" Cursor="Cursors/enterappstarting.ani"/>
    <Button x:Key="forLoadDataMembersScreen" Cursor = "Cursors/
datagridcross.ani"/>

    <Button x:Key="forButEnterLeft" Cursor="Cursors/leftbutton.ani"/>
    <Button x:Key="forCloseButton" Cursor="Cursors/exitcancel.ani"/>
    <Style TargetType="{x:Type Button}" x:Key="DefaultButton">
        <Setter Property="Background" Value="Blue" />
        <Setter Property="Foreground" Value="White" />
        <Setter Property="BorderThickness" Value="0" />
    </Style>
    <Style x:Key = "HeaderColumns" TargetType = "{x:Type
DataGridColumnHeader}">
        <Setter Property="HorizontalAlignment" Value="Center" />
    </Style>
<!-- для этого создаю соответствующие поддиректории -->
    <ObjectDataProvider x:Key="language" />
<!-- движение слева в правую сторону -->
    <PathGeometry x:Key="pathWindowRegistr"
        Figures="M0,0 L-16,-16 1340,-16 1340,440 -16,440 z"/>
<!-- движение справа в левую сторону -->
    <PathGeometry x:Key="pathWindowRegistrFromRight"
        Figures="M0,0 L45,45 -16,-16 z"/>
<PathGeometry x:Key="pathEllipse" // три “круга” летают по окну программы
        Figures="M0,0 L25,25 120,-15 30,36 200,140 59,89 25,40 z"/>
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

48

```

<PathGeometry x:Key="pathImage"
Figures="M0,0 L425,5 120,515 20,440 525,40 550,445 590,455 340,487 470,690 z"/>
<PathGeometry x:Key="pathEllipseThree"
Figures="M0,0 L125,25 420,-15 40,540 50,445 470,690 340,487 470,30 25,240 25,25
120,-15 20,140 25,240 340,487 470,690 z"/>
    <!-- реализую аналоговые часы / для движения секундной стрелки -->
    <Storyboard x:Key="for_seconds" RepeatBehavior="Forever">
        <DoubleAnimationUsingKeyFrames Storyboard.TargetName="linesecond"
Storyboard.TargetProperty="(UIElement.RenderTransform).(TransformGroup.Children)[
2].(RotateTransform.Angle)">
            <EasingDoubleKeyFrame KeyTime="0" Value="-90">
                <EasingDoubleKeyFrame.EasingFunction>
                    <BackEase Amplitude="0.4" EasingMode="EaseOut" />
                </EasingDoubleKeyFrame.EasingFunction>
            </EasingDoubleKeyFrame>
            <EasingDoubleKeyFrame KeyTime="0:0:1" Value="-84">
                <EasingDoubleKeyFrame.EasingFunction>
                    <BackEase Amplitude="0.4" EasingMode="EaseOut" />
                </EasingDoubleKeyFrame.EasingFunction>
            </EasingDoubleKeyFrame>
            <EasingDoubleKeyFrame KeyTime="0:0:2" Value="-78">
                <EasingDoubleKeyFrame.EasingFunction>
                    <BackEase Amplitude="0.4" EasingMode="EaseOut" />
                </EasingDoubleKeyFrame.EasingFunction>
            </EasingDoubleKeyFrame>
            <EasingDoubleKeyFrame KeyTime="0:0:3" Value="-72">
                <EasingDoubleKeyFrame.EasingFunction>
                    <BackEase Amplitude="0.4" EasingMode="EaseOut" />
                </EasingDoubleKeyFrame.EasingFunction>
            </EasingDoubleKeyFrame>

```

Изм.	Лист	№ докум.	Подпись	Дата

```

<EasingDoubleKeyFrame KeyTime="0:0:4" Value="-66 // .... и т.д..... до:
<EasingDoubleKeyFrame KeyTime="0:1:00" Value="270">
    <EasingDoubleKeyFrame.EasingFunction>
        <BackEase Amplitude="0.4" EasingMode="EaseOut" />
    </EasingDoubleKeyFrame.EasingFunction>
</EasingDoubleKeyFrame>
</DoubleAnimationUsingKeyFrames>
</Storyboard>

<!-- для движения минутной стрелки -->
<Storyboard x:Key="for_minutes" RepeatBehavior="Forever">
    <DoubleAnimationUsingKeyFrames Storyboard.TargetName="lineminute"
Storyboard.TargetProperty="(UIElement.RenderTransform).(TransformGroup.Children)[
2].(RotateTransform.Angle)">
        <EasingDoubleKeyFrame KeyTime="0" Value="-90" />
        <EasingDoubleKeyFrame KeyTime="0:1:00" Value="-84" />
        <EasingDoubleKeyFrame KeyTime="0:2:00" Value="-78" />
        <EasingDoubleKeyFrame KeyTime="0:3:00" Value="-72" />
<!-- и т.д. для каждой минуты -->
        <EasingDoubleKeyFrame KeyTime="0:59:00" Value="264" />
        <EasingDoubleKeyFrame KeyTime="1:00:00" Value="270" />
    </DoubleAnimationUsingKeyFrames>
</Storyboard>

<!-- для движения стрелки часов -->
<Storyboard x:Key="for_hours">
    <DoubleAnimationUsingKeyFrames Storyboard.TargetName="linehour"
Storyboard.TargetProperty="(UIElement.RenderTransform).(TransformGroup.Children)[
2].(RotateTransform.Angle)"> <!-- это для стрелки часов (по 1 часу) -->
        <EasingDoubleKeyFrame KeyTime="0:00:00" Value="-90" />
        <EasingDoubleKeyFrame KeyTime="1:00:00" Value="-60" />
        <EasingDoubleKeyFrame KeyTime="2:00:00" Value="-30" />

```

Изм.	Лист	№ докум.	Подпись	Дата

```

<EasingDoubleKeyFrame KeyTime="3:00:00" Value="-00" />
<EasingDoubleKeyFrame KeyTime="4:00:00" Value="30" />
<EasingDoubleKeyFrame KeyTime="5:00:00" Value="60" />
<EasingDoubleKeyFrame KeyTime="6:00:00" Value="90" />
<EasingDoubleKeyFrame KeyTime="7:00:00" Value="120" />
<EasingDoubleKeyFrame KeyTime="8:00:00" Value="150" />
<EasingDoubleKeyFrame KeyTime="9:00:00" Value="180" />
<EasingDoubleKeyFrame KeyTime="10:00:00" Value="210" />
<EasingDoubleKeyFrame KeyTime="11:00:00" Value="240" />
<EasingDoubleKeyFrame KeyTime="12:00:00" Value="270" />
</DoubleAnimationUsingKeyFrames>
</Storyboard>
</ResourceDictionary>
</Window.Resources>

```

### **Добавление виртуальных студентов из файлов \*.xml**

В Visual Studio есть возможность создавать файлы «\*.xml» (Рисунок 5). Так как я разместил в памяти, сперва, только учителей, собираю информацию для вымышленных студентов колледжа. Обращаюсь к различных сайтам (<http://freegenerator.ru/fio> , <https://ciox.ru/generator-full-name> , <https://www.calc.ru/generator-sluchaynykh-nomerov-telefona.html> и т.д.), где возможна генерация случайных ФИО и номеров. Самостоятельно формирую группы и все собранные данные обрабатываю в Microsoft Excel 2019. Где создаю таким образом несколько десятков студентов.

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

*Лист*

**51**

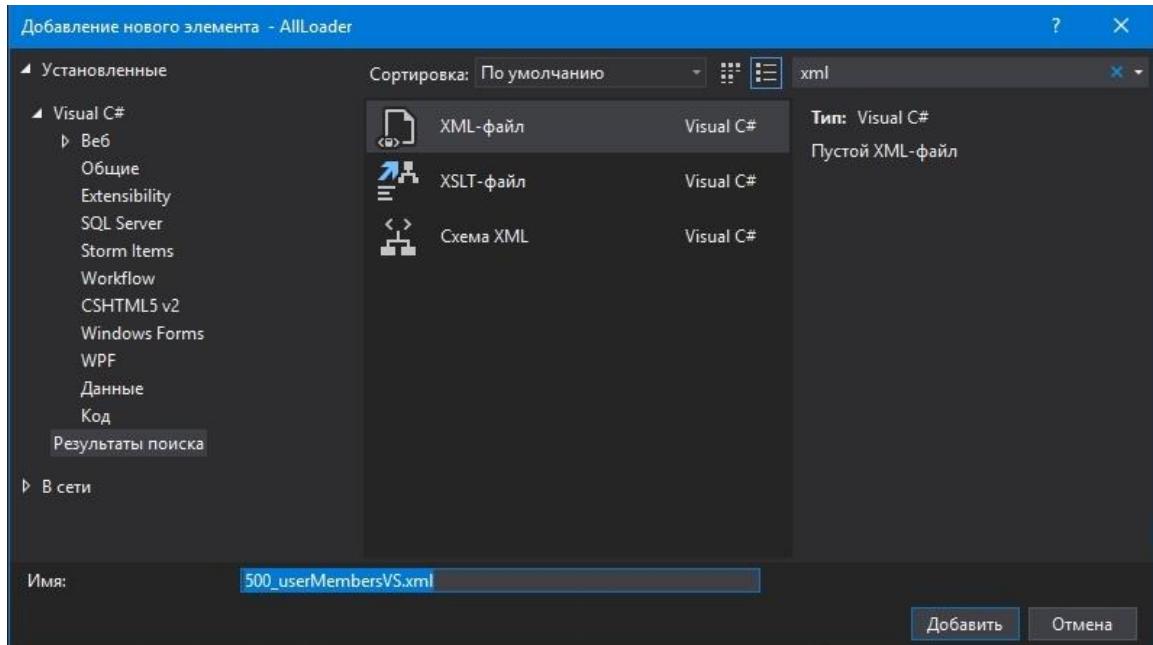


Рисунок 5 – Добавление в проект файлов (\*.userMembersVS.xml)

Но сам процесс такого создания крайне неудобен... В самом MS Excel, гораздо удобнее конвертировать файл (\*.xlsx) в файл данных **XML** (*такая возможность расположена на вкладке «Разработчик»*). Для этого сперва создаю схему, основанную на необходимых мне данных. Эта схема будет определять структуру XML-файлов (Рисунок 6).

	A	B	C	D	E	F	G	H	I
1	<b>id</b>	<b>SurName</b>	<b>FirstName</b>	<b>MiddleName</b>	<b>Email</b>	<b>Phone</b>	<b>Course</b>	<b>Group</b>	<b>PASS</b>
2									
3									
4									

Рисунок 6 – Схема данных для файлов XML

Разбиваю полученные данные о студентах на несколько групп (*по 20, 30, 40, 50, 100 и 500 человек – на разных листах*). Открываю «**Источник XML**», предназначенный для управления картами XML (Рисунок 7). В книге создаю **11** листов (для будущих файлов *xml*), где размещено более 900 студентов (Рисунок 7.1).

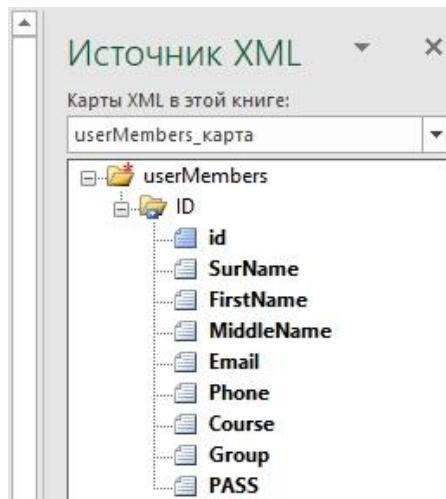


Рисунок 7 – Источник XML

	A	B	C	D	E	F	G	H
1	Глухова	Екатерина	Алексеевна	david.holland80@gmail.com	79161893290	3	У-14-236-3	qwerty0
2	Кобрина	Валерия	Викторовна	cficus@yahoo.com	79647761812	3	У-24-236-о	qwerty0
3	Нуждайкин	Татьяна	Викторовна	rogerball@vanmet.com	84959739403	2	У-12-124-о	qwerty0
4	Овсянники	Ирина	Владимировн	mike@drivetraffic.com	79299360792	2	У-11-105-о	qwerty0
5	Петров	Сергей	Петрович	ljsc07@yahoo.com	79091676547	3	У-13-142-о	qwerty0
6	Подгорная	Наталья	Юрьевна	e_flores72@hotmail.com	79055352244	1	У-21-106-о	qwerty0
7	Тюменцев	Иван	Владимирови	oneyda_med20@yahoo.com	79031529851	2	у-22-121-о	qwerty0
8	Шкарупа	Дмитрий	Васильевич	marvinhn26@yahoo.com	79255295538	4	У-14-114-о	qwerty0
9	Шуршин	Евгений	Михайлович	alvesdelivery@cox.net	79169212366	2	У-13-211-о	qwerty0
10	Горрова	Анна	Олеговна	oscarlolo56@yahoo.com	84232186000	1	У-12-201-о	qwerty0

Рисунок 7.1 – Книга со студентами в Excel, перед экспортом в XML

Данные файлы я размещаю в ресурсах проекта «AllLoader» (Рисунок 9). Для этого создаю папку «userMembers». На главном окне размещаю ComboBox (3), где можно выбрать количество студентов, и кнопку «ADD» (4), для добавления пользователей в DataGridView на экране (Рисунок 8).



Рисунок 8 – ComboBox (3) и кнопка “ADD”

Изм.	Лист	№ докум.	Подпись	Дата	Лист	53
					ВСК.090204.07.00.000 ПЗ	

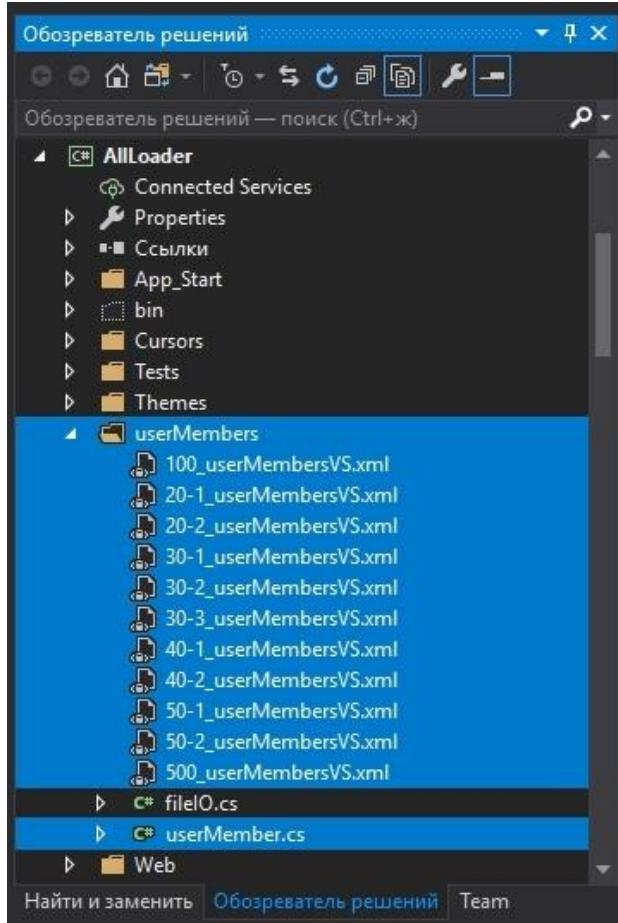


Рисунок 9 – Размещение xml-файлов в ресурсах проекта

### Обращение к xml-файлам в ресурсах проекта

Для обращения к имеющимся в проекте файлам (*списки студентов*), создаю класс «**userMembers.cs**»:

**namespace AllLoader.userMembers**

{*// способ создать сериализуемый класс — отметить его атрибутом [Serializable]*

**[Serializable]**

**public class userMembers : ISerializable** *// интерфейс для обращения*

{*// определяю класс для ввода/добавления пользователей*

**[JsonProperty("id")]**

**public string ID { get; set; }**

**[JsonProperty("surname")]**

**public string SurName { get; set; }**

**[JsonProperty("firstname")]**

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

**Лист**

**54**

```

public string FirstName { get; set; }
[JsonProperty("middlename")]
public string MiddleName { get; set; }
[JsonProperty("email")]
public string Email { get; set; }
[JsonProperty("phone")]
public string Phone { get; set; }
[JsonProperty("course")]
public string Course { get; set; }
[JsonProperty("group")]
public string Group { get; set; }
[JsonProperty("pass")]
public string PASS { get; set; }

// для сериализации класса
public void GetObjectData(SerializationInfo info, StreamingContext context)
{ // интерфейс для работы с классом "userMember" / запись значений
    info.AddValue("ID", ID);
    info.AddValue("SurName", SurName);
    info.AddValue("FirstName", FirstName);
    info.AddValue("MiddleName", MiddleName);
    info.AddValue("Email", Email);
    info.AddValue("Phone", Phone);
    info.AddValue("Course", Course);
    info.AddValue("Group", Group);
    info.AddValue("PASS", PASS);      }
public userMembers() { }

// запись полученных значений в буфер программы
public userMembers(string name = "",
                    string id = "", string surname = "", string firstname = "",
                    string middlename = "", string email = "", string phone = "",

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        string course = "", string group = "", string pass = "")

    {           ID = id;           SurName = surname;           FirstName = firstname;
    MiddleName = middlename;           Email = email;           Phone = phone;
    Course = course;           Group = group;           PASS = pass;       }

// возврат существующих значений, преобразование в формируемую строку

public override string ToString()
{
    // для записи значений в файл (запись только для одного студента !)
    return string.Format("ID: {0}; ФИО: {1} ", " ", "{2} ", " ", "{3}. " +
    "(Группа: {7} Курс: {6}) Почта: {4}; Телефон: {5} " +
    "Пароль: {8}", ID, SurName, FirstName, MiddleName,
    Email, Phone, Course, Group, PASS);
}

// для записи полученных данных из класса в файл

public userMembers(SerializationInfo info, StreamingContext context)
{
    // получаю имеющиеся данные и их формат (string)
    ID = (string)info.GetValue("ID", typeof(string));
    SurName = (string)info.GetValue("SurName", typeof(string));
    FirstName = (string)info.GetValue("FirstName", typeof(string));
    MiddleName = (string)info.GetValue("MiddleName", typeof(string));
    Email = (string)info.GetValue("Email", typeof(string));
    Phone = (string)info.GetValue("Phone", typeof(string));
    Course = (string)info.GetValue("Course", typeof(string));
    Group = (string)info.GetValue("Group", typeof(string));
    PASS = (string)info.GetValue("PASS", typeof(string));      }

public class usersMembers
{
    public List<usersMembers> ListUsersMembers { get; set; } = new
List<usersMembers>();      }

```

После выбора количества студентов в «**ComboBox**» и нажатия кнопки «**ADD**» начинает работать написанная мной функция **"addDataGridMembers\_Click"** (более 800 строк кода, поэтому с сокращениями):

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

Лист

56

```

public void addDataGridMembers_Click(object sender, RoutedEventArgs e)
{
    // добавить новых пользователей по выбору в ComboBox-е
    // строки: NewCountUsers = AllLoader.Properties.Settings.Default.haved_Row;
    // переменная для количества новых пользователей (число)

    string userCount = System.Convert.ToString(addCount.SelectedItem);

    // обрезаю первые 38 символов, а именно:
    userCount = userCount.Remove(0, 38);

    // умножаю на количество полей для одного пользователя
    // CountNewUsers = Convert.ToInt16(userCount) * 9;

    // какое количество было выбрано:
    int plusUser = addCount.SelectedIndex;

    switch (plusUser) {
        case 0: //10 чел. // если выбраны мальчики
            if ((MG == "men") && (menInput == false))
{
                // чтобы нельзя было второй раз добавить тех же "мальчиков" в 'Grid'
                menInput = true;
                MG = "girl"; // переключаю на девочек
                // в цикле добавляю новых пользователей в DataGrid
                for (int q = 0; q < 100; q += 9)
{
                    // конвертирую счетчик в строку
                    string any = Convert.ToString(q);

                    // добавляю нового пользователя к общему количеству
                    // AllLoader.Properties.Settings.Default.baseUsers += 1;
                    // для правильных порядковых номеров пишу функцию –
                        "addNewNumber_UserMember"
                    // но всё это вымышленные пользователи - просто для количества
                    // отправляю в функцию для установки правильного номера
                    if (q == 0) { // для мальчиков
                        this._dataTable.Rows.Add(addNewNumberTen_UserMember(forNumber) +
(Convert.ToString(NewCountUsers++)),
                        Convert.ToString(rndMenUser[0][q]),
                        Convert.ToString(rndMenUser[1][q]),
                        Convert.ToString(rndMenUser[2][q]),

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

57

```

        Convert.ToString(rndMenUser[3][q]),
        Convert.ToString(rndMenUser[4][q]),
        Convert.ToString(rndMenUser[5][q]),
        Convert.ToString(rndMenUser[6][q]),
        Convert.ToString(rndMenUser[7][q]));
    // заполняю ObservableCollection
userMembers.Add(new userMember() {
    ID = (Convert.ToString(NewCountUsers)),
    SurName = (Convert.ToString(rndMenUser[0][q])),
    FirstName = (Convert.ToString(rndMenUser[1][q])),
    MiddleName = Convert.ToString(rndMenUser[2][q]),
    Email = Convert.ToString(rndMenUser[3][q]),
    Phone = Convert.ToString(rndMenUser[4][q]),
    Course = Convert.ToString(rndMenUser[5][q]),
    Group = Convert.ToString(rndMenUser[6][q]),
    PASS = Convert.ToString(rndMenUser[7][q]) );
    // заполняю виртуальный лист
newUser.Add(new ListUserMembers() {
    ID = (Convert.ToString(NewCountUsers)),
    SurName = Convert.ToString(rndMenUser[0][q]),
    FirstName = Convert.ToString(rndMenUser[1][q]),
    MiddleName = Convert.ToString(rndMenUser[2][q]),
    Email = Convert.ToString(rndMenUser[3][q]),
    Phone = Convert.ToString(rndMenUser[4][q]),
    Course = Convert.ToString(rndMenUser[5][q]),
    Group = Convert.ToString(rndMenUser[6][q]),
    PASS = Convert.ToString(rndMenUser[7][q]) );
    // заношу в лист полученные данные
ListUserMembers.ItemsSource = newUser;
    // чтобы вывести(показать) список на экран
CollectionView view = (CollectionView)
CollectionViewSource.GetDefaultView(ListUserMembers.ItemsSource);
    // сортировку делаю по номеру
view.SortDescriptions.Add(new SortDescription("ID", ListSortDirection.Ascending));
    // и добавляю в таблицу: "_tableUserMembers"
_tableUserMembers.Rows.Add(addNewNumberTen_UserMember(forNumber) +
    (Convert.ToString(NewCountUsers)),
    Convert.ToString(rndMenUser[0][q]),
    Convert.ToString(rndMenUser[1][q]),
    Convert.ToString(rndMenUser[2][q]),
    Convert.ToString(rndMenUser[3][q]),
    Convert.ToString(rndMenUser[4][q]),
    Convert.ToString(rndMenUser[5][q]),
    Convert.ToString(rndMenUser[6][q]),
    Convert.ToString(rndMenUser[7][q]));

```

Изм.	Лист	№ докум.	Подпись	Дата

```

AllLoader.Properties.Settings.Default.haved_Row++; }

// || (q == 45) || (q == 54) || (q == 63) || (q == 72) || (q == 81) || (q == 90)
else if (((q != 0) && (q == 9)) || (q == 18) || (q == 27) || (q == 36) || (q == 45) || (q
== 54) || (q == 63) || (q == 72) || (q == 81))
{
    // заношу в DataGridView (на главное форме) на экране
    this._dataTable.Rows.Add(addNewNumberTen_UserMember(forNumber) +
(Convert.ToString(NewCountUsers++)),
Convert.ToString(rndMenUser[0][q / 9]),
Convert.ToString(rndMenUser[1][q / 9]),
Convert.ToString(rndMenUser[2][q / 9]),
Convert.ToString(rndMenUser[3][q / 9]),
Convert.ToString(rndMenUser[4][q / 9]),
Convert.ToString(rndMenUser[5][q / 9]),
Convert.ToString(rndMenUser[6][q / 9]),
Convert.ToString(rndMenUser[7][q / 9]));
    // заполняю ObservableCollection (это было в памяти)
userMembers.Add(new userMember() { ID = (Convert.ToString(NewCountUsers)),
SurName = (Convert.ToString(rndMenUser[0][q / 9])), FirstName =
(Convert.ToString(rndMenUser[1][q / 9])), MiddleName =
Convert.ToString(rndMenUser[2][q / 9]), Email =
Convert.ToString(rndMenUser[3][q / 9]), Phone =
Convert.ToString(rndMenUser[4][q / 9]), Course =
Convert.ToString(rndMenUser[5][q / 9]), Group =
Convert.ToString(rndMenUser[6][q / 9]), PASS =
Convert.ToString(rndMenUser[7][q / 9]) });
    // заполняю виртуальный лист
newUser.Add(new ListUserMembers() {
ID = (Convert.ToString(NewCountUsers)),
SurName = Convert.ToString(rndMenUser[0][q / 9]),
FirstName = Convert.ToString(rndMenUser[1][q / 9]),
MiddleName = Convert.ToString(rndMenUser[2][q / 9]),
Email = Convert.ToString(rndMenUser[3][q / 9]),
Phone = Convert.ToString(rndMenUser[4][q / 9]),
Course = Convert.ToString(rndMenUser[5][q / 9]),
Group = Convert.ToString(rndMenUser[6][q / 9]),
PASS = Convert.ToString(rndMenUser[7][q / 9]) });
    // заношу в лист полученные данные
ListUserMembers.ItemsSource = newUser;
    // чтобы вывести(показать) список на экран
CollectionView view = (CollectionView)
CollectionViewSource.GetDefaultView(ListUserMembers.ItemsSource);
    // сортировку делаю по номеру
view.SortDescriptions.Add(new SortDescription("ID", ListSortDirection.Ascending));
    // и добавляю в таблицу: "_tableUserMembers"
_tableUserMembers.Rows.Add(addNewNumberTen_UserMember(forNumber) +

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

59

```

(Convert.ToString(NewCountUsers)),
Convert.ToString(rndMenUser[0][q / 9]),
Convert.ToString(rndMenUser[1][q / 9]),
Convert.ToString(rndMenUser[2][q / 9]),
Convert.ToString(rndMenUser[3][q / 9]),
Convert.ToString(rndMenUser[4][q / 9]),
Convert.ToString(rndMenUser[5][q / 9]),
Convert.ToString(rndMenUser[6][q / 9]),
Convert.ToString(rndMenUser[7][q / 9]));

// добавляю количество слушателей(строк) в системных параметрах приложения
AllLoader.Properties.Settings.Default.haved_Row++;
} } forMenUser = true; }

// значит выбраны девочки и идёт повтор кода!!! (это было просто из памяти)
// это было из массивов, а далее работа с xml файлами из ресурсов

else if ((for20 == 3) && (fileXml[0] == true) && (enabledFileXml[0] == 0))
{
    // если есть разрешение для работы с файлом проекта

if ((enabledXmlFile[0] == true) && (fileXml[0] == true) && (enabledFileXml[0] == 0)) {
    var xml20_1 = XDocument.Parse(Properties.Resources._20_1_userMembersVS);

        // создаю пустой объект DataSet
        DataSet dataSet = new DataSet();

        // и заполняю его данными из прочитанного в ресурсах xml-файла
        dataSet.ReadXml(xml20_1.CreateReader());

        // перебор всей таблицы (полученной из файла)
        foreach (System.Data.DataTable dataTable in dataSet.Tables)
        {
            if (Properties.Settings.Default.xml201 == false)
            {
                // прохожу по всем колонкам
                foreach (DataColumn column in dataTable.Columns)
                {
                    // прохожу по всем строкам
                    foreach (DataRow row in dataTable.Rows)
                    {
                        // получаю все ячейки данной строки
                        var cells = row.ItemArray;

                        // вывожу полученные данные на экран
                        if ((id0 < 20) && (fileXml[0] == true) && (enabledFileXml[0] == 0))

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        { _dataTable.LoadDataRow(row.ItemArray, true);    }

        id0++; // делаю отметку что данный xml-файл уже был использован
Properties.Settings.Default.xml201 = true; } } } }

                                enabledFileXml[0] = 1;
                                fileXml[0] = false;
                                enabledXmlFile[0] = false;
                                for20 = 4; } } // и т.д. 20-30-...-100

```

**//... и последний файл в ресурсах на 500 студентов**

```

case 6: //500
if ((fileXml[10] == true) && (enabledFileXml[10] == 0))
{ // если есть разрешение для работы с файлом проекта
    if (enabledXmlFile[10] == true)

{ var xml500 = XDocument.Parse(Properties.Resources._500_userMembersVS);

    // создаю пустой объект DataSet
    DataSet dataSet500 = new DataSet();
    // и заполняю его прочитанным в ресурсах xml-файлом
    dataSet500.ReadXml(xml500.CreateReader());
    // перебор всей таблицы
    foreach (System.Data.DataTable dataTable in dataSet500.Tables)
    {
        if (Properties.Settings.Default.xml500 == false)
        { // прохожу по всем колонкам
            foreach (DataColumn column in dataTable.Columns)
            { // прохожу по всем строкам
                foreach (DataRow row in dataTable.Rows)
                { // получаю все ячейки данной строки
                    var cells = row.ItemArray;
                    // вывожу полученные данные на экран
                    if (id10 < 500)
                    { _dataTable.LoadDataRow(row.ItemArray, true); } id10++;
                    // делаю отметку что данный xml-файл уже был использован
Properties.Settings.Default.xml500 = true; } } }

                                enabledFileXml[10] = 1;
                                fileXml[10] = false;
                                enabledXmlFile[10] = false;

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        }
        else if (fileXml[10] == false)
        {
            MessageBox.Show("Полтысячи виртуальных слушателей из массива" +
                " уже были добавлены из файла проекта. " +
                "Дальнейшее добавление указанного количества невозможно."); } } break;
        default: // сообщение об использовании данного файла
            MessageBox.Show("Дальнейшее добавление из ресурсов невозможно.");
            break; } }

```

Чтобы правильно формировался порядковый номер, создаю функцию для подсчета нулей в порядковом номере для студента:

```

public string addNewNumber_UserMember(string fNum)
{
    // для правильного числа ноликов в номерах на экране
    // добавляю сперва 5, т.к. 4 пользователей и пустую строку я внёс в программу
    // потом добавляю учителей ВСК (пароль: "qwerty" + число порядкового номера
    // "после всех нулей")

    if (AllLoader.Properties.Settings.Default.haved_Row < 100) { fNum = "00000"; }
    else if (AllLoader.Properties.Settings.Default.haved_Row < 1000) { fNum = "0000"; }
    else if (AllLoader.Properties.Settings.Default.haved_Row < 10000) { fNum = "000"; }
    else if (AllLoader.Properties.Settings.Default.haved_Row < 100000) { fNum = "00"; }
    else if (AllLoader.Properties.Settings.Default.haved_Row < 1000000) { fNum = "0"; }
    else if (AllLoader.Properties.Settings.Default.haved_Row < 10000000) { fNum = ""; }

    return fNum;
}

```

Если пользователь самостоятельно вводит необходимые данные, их принимает другая функция — **addNewUserMember**:

```

public void addNewUserMember(int x, string Num, string Sur, string Nam, string Mid,
                            string Em, string Ph, string Ku, string Gr, string Pass)
{
    // функция для добавления новых пользователей в программу
    for (short y = 0; y < 9; y++) { }

```

Изм.	Лист	№ докум.	Подпись	Дата

```

if (y == 0) { nowUserMember[x, y] = addNewNumber_UserMember(Num) +
Convert.ToString(AllLoader.Properties.Settings.Default.baseUsers + 1); }

    if (y == 1) { nowUserMember[x, y] = Sur; }
    if (y == 2) { nowUserMember[x, y] = Nam; }
    if (y == 3) { nowUserMember[x, y] = Mid; }
    if (y == 4) { nowUserMember[x, y] = Em; }
    if (y == 5) { nowUserMember[x, y] = Ph; }
    if (y == 6) { nowUserMember[x, y] = Ku; }
    if (y == 7) { nowUserMember[x, y] = Gr; }
    if (y == 8) { nowUserMember[x, y] = Pass; } }

// дополнительно вывод новых пользователей в DataGrid программы на экран

this._dataTable.Rows.Add(addNewNumber_UserMember(Num) +
Convert.ToString(AllLoader.Properties.Settings.Default.baseUsers + 1), Sur, Nam, Mid,
                    // увеличиваю общее количество пользователей Em, Ph, Ku, Gr, Pass);
                    // AllLoader.Properties.Settings.Default.haved_Row++; }

```

## Встроенная библиотека PDF

Скачиваю в Интернете различные книги по языкам программирования C++ и C# (*и на русском и на английском языках*). Размещаю их в ресурсах проекта, в частности в папке «**Resources**» (Рисунок 10). Для наименований книг по C# на русском языке беру последовательность от **0001.pdf** до **0030.pdf**; для книг по C++ от **Cpp01.pdf** до **Cpp16.pdf**; для книг по C# на английском языке — от **en001.pdf** до **en034.pdf**.

Добавляю в проект окно WPF — «**PDFreader.xaml**»:

```

<Window x:Class="AllLoader.PDFreader"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

```

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

Лист

63

```

xmlns:local="clr-namespace:AllLoader" WindowStartupLocation="CenterScreen"
mc:Ignorable="d" Icon="Icons/Books/book.png"
Title="ВСК библиотека PDF" MinHeight="750" MinWidth="1000" >
<Grid> // разбиваю окно на участки
<Grid.ColumnDefinitions>

```

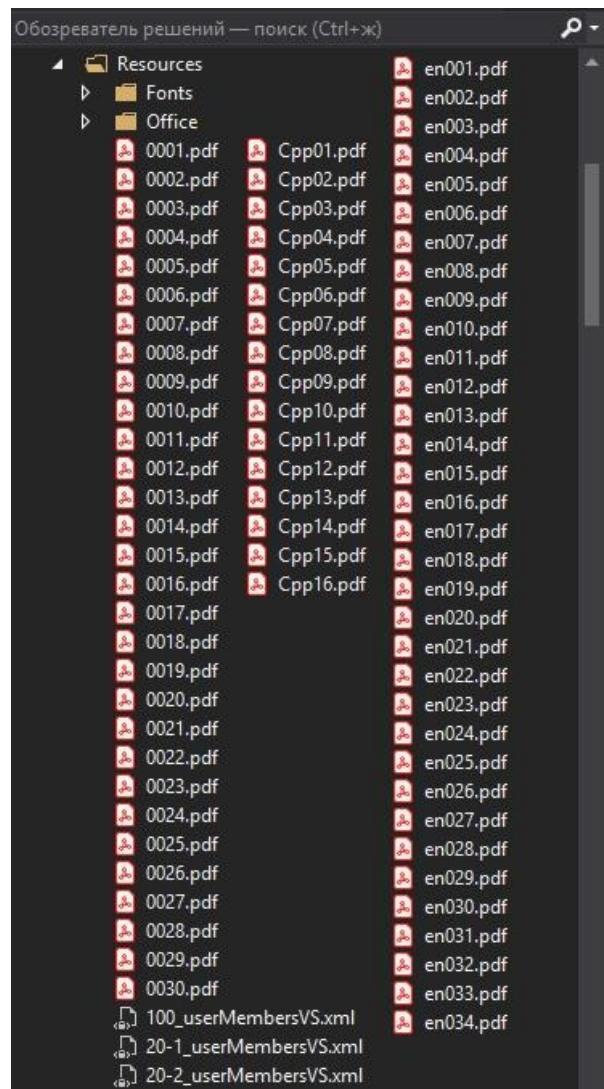


Рисунок 10 – Размещение pdf-файлов в ресурсах проекта

```

<ColumnDefinition Width="500*"/>
<ColumnDefinition Width="5*"/>
</Grid.ColumnDefinitions>
<Grid.RowDefinitions>
    <RowDefinition Height="40"/></RowDefinition>
    <RowDefinition Height="40"/></RowDefinition>

```

Изм.	Лист	№ докум.	Подпись	Дата

```

<RowDefinition Height="40"></RowDefinition>
<RowDefinition Height="Auto"></RowDefinition>
</Grid.RowDefinitions>

// FlowDocumentScrollViewer - отображает документ с полосой прокрутки
<FlowDocumentScrollViewer Grid.Row="3" x:Name="DocViewer"
    Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible" >
<FlowDocument x:Name="forDocViewer"
    ScrollViewer.VerticalScrollBarVisibility="Visible">
<Paragraph x:Name="oneP">
    <Image Source="Images/Tooltip/vsk.png" Width="120" Height="120"
Margin="0,0,20,-10" /> <Run FontSize="140" Foreground="DarkOrange">WPF</Run>
</Paragraph>
<Paragraph FontSize="22"> WPF, что означает Windows Presentation Foundation, является последним подходом Microsoft к среде графического интерфейса пользователя, используемым с платформой .NET framework.

Некоторые из преимуществ: <Paragraph>
<List>    <ListItem>
    <Paragraph FontSize="18" Margin="5,1,0,15" FontStyle="Italic">
Он новее и, следовательно, больше соответствует текущим стандартам;
    </Paragraph>        <ListItemText>
    <ListItemText>
        <Paragraph FontSize="18" Margin="5,1,0,15" FontStyle="Italic">
Microsoft использует его для множества новых приложений, например Visual Studio;
    </Paragraph>        <ListItemText>        <ListItemText>
    <Paragraph FontSize="18" Margin="5,1,0,15" FontStyle="Italic">
Он более гибкий, так что Вы можете делать больше вещей без необходимости писать или покупать новые элементы управления.
    </Paragraph>        <ListItemText>        </List>

```

Изм.	Лист	№ докум.	Подпись	Дата

<Paragraph TextAlignment="Center" Foreground="#060EC6"  
 FontStyle="Italic" FontSize="18">  
 Таблица статистических данных: WinForms / WPF</Paragraph>  
 <Table CellSpacing="0">  
 <TableRowGroup>  
 <TableRow Background="#08ECEC" FontWeight="Bold">  
 <TableCell></TableCell>  
 <TableCell>  
 <Paragraph TextAlignment="Right">WinForms</Paragraph>  
 </TableCell>  
 <TableCell> <Paragraph TextAlignment="Right">WPF</Paragraph>  
 </TableCell>  
 </TableRow> </TableRowGroup>  
 <TableRowGroup>  
 <TableRow>  
 <TableCell Background="#DDF105" FontWeight="Bold">  
 <Paragraph Margin="10,5,0,5">Строки кода:</Paragraph>  
 </TableCell>  
 <TableCell> <Paragraph Margin="0,5,0,5"  
 TextAlignment="Right">1.718.000</Paragraph>  
 </TableCell> <TableCell>  
 <Paragraph Margin="0,5,0,5" TextAlignment="Right">1.542.000</Paragraph>  
 </TableCell> </TableRow>  
 </TableRowGroup>  
 <TableRowGroup> <TableRow>  
 <TableCell Background="#DD7BF7" FontWeight="Bold">  
 <Paragraph Margin="10,5,0,5">Разработчики:</Paragraph>  
 </TableCell> <TableCell>  
 <Paragraph Margin="0,5,0,5" TextAlignment="Right">633.000</Paragraph>  
 </TableCell> <TableCell>

Изм.	Лист	№ докум.	Подпись	Дата

BCK.090204.07.00.000 ПЗ

Лист

66

```

<Paragraph Margin="0,5,0,5"
TextAlignment="Right">981.000</Paragraph>
    </TableCell>      </TableRow>
</TableRowGroup>      </Table>
</FlowDocument>
</FlowDocumentScrollView>
<StackPanel Grid.Row="0" Orientation="Horizontal" Grid.ColumnSpan="2"
x:Name="onePanelPDF" Height="40">
    <TextBlock Text="    Файл:" FontSize="24" VerticalAlignment="Center"
x:Name="txtFile1PDF"          FontStyle="Normal"           Background="Aqua"
HorizontalAlignment="Left"     Width="80"   Margin="10,-5,5,-5"   ToolTip="Нужно
выбрать pdf-файл"/>
    <!-- перечень книги-pdf на русском языке про C# -->
<ComboBox x:Name="comboPDF" Width="550" Height="33" SelectedIndex="2"
Background="Coral" FontSize="18" FontStyle="Italic">
    <ComboBoxItem Content="Ресурсы и стили элементов управления в
приложениях WPF. А.Б.Шамшев. 2013. (160 стр.)" x:Name="pdf0001" />
    <ComboBoxItem Content="Пример разработки приложений WPF.
Р.К.Ахмадулин. 2016. (26 стр.)" x:Name="pdf0002" />
    <ComboBoxItem Content="Документация по интегрированной среде
разработки Visual Studio. Microsoft. 2019. (2237 стр.)" x:Name="pdf0003" />
    <ComboBoxItem Content="Электронное учебное пособие «РАЗРАБОТКА
WPF ПРИЛОЖЕНИЙ В С#». А.О.Балакин. 2018. (58 стр.)" x:Name="pdf0004" />
    <ComboBoxItem Content="Создание панели инструментов. А.В.Абрамян.
2017/2018. (20 стр.)" x:Name="pdf0005" />
    <ComboBoxItem Content="Курсоры и иконки. А.В.Абрамян. 2017/2018. (27
стр.)" x:Name="pdf0006" />
    <ComboBoxItem Content="Подробное руководство WPF4. Адам Натан.
2011. (889 стр.)" x:Name="pdf0007" />

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

67

<ComboBoxItem Content="Библия С# 2-е издание. М.Фленов. 2011. (542 стр.)" x:Name="pdf0008"/>

<ComboBoxItem Content="Полное руководство C#4.0. Герберт Шилдт. 2011. (949 стр.)" x:Name="pdf0009"/>

<ComboBoxItem Content="Карманный справочник C#7.0. Джозеф Албахари. 2017. (226 стр.)" x:Name="pdf0010"/>

<ComboBoxItem Content="Справочник C#7.0 (полное описание языка). Джозеф Албахари. 2018. (1026 стр.)" x:Name="pdf0011"/>

<ComboBoxItem Content="C# для начинающих. Б.Пахомов. 2014. (432 стр.)" x:Name="pdf0012"/>

<ComboBoxItem Content="Учебное пособие C# для школьников. М.Дрейер. 2010. (128 стр.)" x:Name="pdf0013"/>

<ComboBoxItem Content="C# на примерах. П.В.Евдокимов. 2016. (304 стр.)" x:Name="pdf0014"/>

<ComboBoxItem Content="Библия C# 3-е издание. М.Фленов. 2016. (546 стр.)" x:Name="pdf0015"/>

<ComboBoxItem Content="Изучаем C# 3-е издание. Грин.Дж., Э.Стиллмен. 2014. (816 стр.)" x:Name="pdf0016"/>

<ComboBoxItem Content="Увлекательное программирование в C#. И.П.Дудина. 2013. (44 стр.)" x:Name="pdf0017"/>

<ComboBoxItem Content="Принципы, паттерны и методики гибкой разработки на языке C#. Роберт Мартин. 2011. (757 стр.)" x:Name="pdf0018"/>

<ComboBoxItem Content="Программирование на C# для начинающих. Основные сведения. А.Васильев. 2018. (586 стр.)" x:Name="pdf0019"/>

<ComboBoxItem Content="Программирование на языке высокого уровня C#. Т.А.Павловская. 2016. (246 стр.)" x:Name="pdf0020"/>

<ComboBoxItem Content="Разработка алгоритмов с использованием принципов ООП на языке C#. А.А.Калинин. 2014. (106 стр.)" x:Name="pdf0021"/>

<ComboBoxItem Content="Разработка обслуживаемых программ на языке C#. Джуст Виссер. 2017. (194 стр.)" x:Name="pdf0022"/>

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

68

<ComboBoxItem Content="Язык C#. Базовый курс. В.В.Подбельских. 2013.  
 (427 стр.)" x:Name="pdf0023"/>

<ComboBoxItem Content="Язык C#: Краткое описание и введение в  
 технологии программирования. О.М.Котов. 2013. (210 стр.)" x:Name="pdf0024"/>

<ComboBoxItem Content="Unity в действии. Мультиплатформенная  
 разработка на C#. Джозеф Хокинг. 2016. (336 стр.)" x:Name="pdf0025"/>

<ComboBoxItem Content="Работа с базами данных на языке C#. Технология ADO .NET. О.Н.Евсеева. 2016. (171 стр.)" x:Name="pdf0026"/>

<ComboBoxItem Content="LINQ. Карманный справочник C#3.0. Джозеф Албахари. 2009. (241 стр.)" x:Name="pdf0027"/>

<ComboBoxItem Content="Microsoft Visual C#. Подробное руководство. Джон Шарп. 2017. (848 стр.)" x:Name="pdf0028"/>

<ComboBoxItem Content="CLR via C#. Программирование на платформе .NET Framework 4.5 на языке C#. Рихтер Дж. 2013. (896 стр.)" x:Name="pdf0029"/>

<ComboBoxItem Content="Технология XAML. 2017. (20 стр.)" x:Name="pdf0030"/>

</ComboBox> // кнопка для открытия данного источника

<Button Click="btnOpenPDF\_Click" Background="Aqua" FontWeight="Bold" Style="{StaticResource someStyle}" Margin="5,2,5,2" Width="80" Height="33" FontSize="16" Foreground="Black" BorderThickness="2" BorderBrush="Blue" ToolTip="Открыть во внешней  
 программе">Открыть</Button> </StackPanel>

<StackPanel Grid.Row="1" Orientation="Horizontal" Grid.ColumnSpan="2" x:Name="twoPanelPDF" Height="40">
 <TextBlock Text="Fail:" FontSize="24" VerticalAlignment="Center" x:Name="txtFile2PDF" FontStyle="Normal" Background="#00FF6D" HorizontalAlignment="Left" Width="80" Margin="10,-5,5,-5" ToolTip="Нужно выбрать pdf-файл" Grid.Row="1" />

Изм.	Лист	№ докум.	Подпись	Дата

```

<ComboBox      x:Name="comboEnPDF"      Width="550"      Height="33"
SelectedIndex="3" Grid.Row="1"

    Background="Coral" FontSize="18" FontStyle="Italic">

<!-- далее перечень книг-pdf на английском языке про C# -->

<ComboBoxItem Content="C Sharp Programming. Wikibooks.org. 2013. (175
pages)" x:Name="en001"/>

<ComboBoxItem Content="C# 6.0 the .NET 4.6 Framework. Andrew Troelsen.
2015 (1837 pages)" x:Name="en002"/>

<ComboBoxItem Content="C# 6.0 in a Nutshell. Joseph Albahari. 2016. (1133
pages)" x:Name="en003" />

<ComboBoxItem Content="C# 7 Quick Syntax Reference: A Pocket Guide to
the Language, APIs, and Library. Mikael Olsson. 2018. (178 pages)" x:Name="en004"/>

<ComboBoxItem Content="C# 7.1 and .NET Core 2.0 - Modern Cross-
Platform Development. Mark J. Price. 2017. (1131 pages)" x:Name="en005"/>

<ComboBoxItem Content="C# Code Contracts Succinctly by Dirk Strauss.
2016. (90 pages)" x:Name="en006"/>

<ComboBoxItem Content="C# Programming Yellow Book. Rob Miles. 2015.
(214 pages)" x:Name="en007"/>

<ComboBoxItem Content="Data Capture and Extraction with C# Succintly by
Ed Freitas. (85 pages)" x:Name="en008"/>

<ComboBoxItem Content="Design Patterns in C#. Vaskaran Sarcar. 2018. (465
pages)" x:Name="en009"/>

<ComboBoxItem Content="Programming Dictionary in C#. Mahesh Chand.
2012. (6 pages)" x:Name="en010"/>

<ComboBoxItem Content="Working with Directories in C#. Mahesh Chand.
2012. (10 pages)" x:Name="en011"/>

<ComboBoxItem Content="Dissecting a C# Application. Inside SharpDevelop.
Christian Holm. 2004. (538 pages)" x:Name="en012"/>

<ComboBoxItem Content="FileInfo in C#. Mahesh Chand. 2012. (16 pages)"
x:Name="en013"/>

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

70

<ComboBoxItem Content="Fundamentals of Computer Programming with C#. Svetlin Nakov. 2013. (1122 pages)" x:Name="en014"/>

<ComboBoxItem Content="\_GRAY\_Hat\_C#. Brandon Perry. 2017. (285 pages)" x:Name="en015"/>

<ComboBoxItem Content="Illustrated C# 7. Daniel Solis. 2018. (817 pages)" x:Name="en016"/>

<ComboBoxItem Content="Object-Oriented Programming Using C#. Naveed Zaman. 2013. (80 pages)" x:Name="en017"/>

<ComboBoxItem Content="Introduction to C# 8. Bassam Alugili. 2020. (59 pages)" x:Name="en018"/>

<ComboBoxItem Content="Learning C# by Developing Games with Unity 5.x. Greg Lukosek. 2016. (230 pages)" x:Name="en019"/>

<ComboBoxItem Content="Mastering C# Concurrency. Eugene Agafonov. 2015. (285 pages)" x:Name="en020"/>

<ComboBoxItem Content="Mastering Unity Scripting. Alan Thorn. 2015. (280 pages)" x:Name="en021"/>

<ComboBoxItem Content="Neural Networks in Unity. C# Programming for Windows 10. Abhishek Nandy. 2018. (166 pages)" x:Name="en022"/>

<ComboBoxItem Content="Object-Oriented Programming using C#. Simon Kendal. 2018. (403 pages)" x:Name="en023"/>

<ComboBoxItem Content="Object-Oriented Programming in C# Succinctly. Sander Rossel. 2016. (95 pages)" x:Name="en024"/>

<ComboBoxItem Content="Printing in C# Made Easy. C# Corner Authors Team. 2007. (82 pages)" x:Name="en025"/>

<ComboBoxItem Content="Pro C# 7. With .NET and .NET Core. Andrew Troelsen. 2017. (1410 pages)" x:Name="en026"/>

<ComboBoxItem Content="Professional C# 7 and .NET Core 2.0. Christian Nagel. 2018. (1439 pages)" x:Name="en027"/>

<ComboBoxItem Content="Programming XAML. Beginners Guide. Mahesh Chand. 2014. (61 pages)" x:Name="en028"/>

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

71

```

    <ComboBoxItem Content="The Art of Unit Testing Second Edition with
examples in C#. Roy Osherove. 2014. (294 pages)" x:Name="en029"/>
    <ComboBoxItem Content="The C# Player's Guide. RB Whitaker. 2012-2015.
(369 pages)" x:Name="en030"/>
    <ComboBoxItem Content="Writing High-Performance .NET Code. Ben
Watson. 2014. (282 pages)" x:Name="en031"/>
    <ComboBoxItem Content="Pro WPF in CSharp 2010 Windows Presentation
Foundation in .NET 4. Matthew MacDonaid. 2010. (1201 pages)" x:Name="en032"/>
    <ComboBoxItem Content="Pro.WPF 4.5 in C# Windows Presentation
Foundation in .NET 4.5. Matthew MacDonaid. 2012. (1095 pages)" x:Name="en033"/>
    <ComboBoxItem Content="Pro WPF 4.5 in C# Windows Presentation
Foundation in .NET 4.5 (EXPERT) Matthew MacDonaid. 2012. (1476 pages)"
x:Name="en034"/>
</ComboBox> // кнопка для открытия данного источника
<Button Click="btnEnOpenPDF_Click" Background="#00FF6D"
FontWeight="Bold" Style="{StaticResource someStyle}" Margin="5,2,5,2" Width="80"
Height="33" FontSize="16" Foreground="Black" BorderThickness="2"
BorderBrush="Blue" ToolTip="Открыть во внешней программе">Open</Button>
</StackPanel>
<StackPanel Grid.Row="2" Orientation="Horizontal" Grid.ColumnSpan="2"
x:Name="threePanelPDF" Height="40">
    <TextBlock Text="Файл:" FontSize="24" VerticalAlignment="Center"
FontStyle="Normal" Background="#FF9200"
        HorizontalAlignment="Left" Width="80" Margin="10,-5,5,-5"
x:Name="txtFile3PDF" ToolTip="Нужно выбрать pdf-файл"/>
        <!-- далее перечень книг-pdf по С++ -->
    <ComboBox x:Name="combo3RuPDF" Width="550" Height="33"
SelectedIndex="4" Grid.Row="2" Background="Coral" FontSize="18"
FontStyle="Italic">

```

Изм.	Лист	№ докум.	Подпись	Дата

<ComboBoxItem Content="C++ для "чайников". Стефан Р. Дэвис. 2003.  
(337 стр.)" x:Name="pdfCpp01"/>

<ComboBoxItem Content="C++ для начинающих. Стенли Липпман. 1985.  
(1194 стр.)" x:Name="pdfCpp02"/>

<ComboBoxItem Content="C++. Мастер-класс в задачах и примерах.  
М.Кузнецов, И.Симдянов. 2007. (479 стр.)" x:Name="pdfCpp03"/>

<ComboBoxItem Content="C++. Руководство для начинающих. Герберт  
Шилдт. 2005. (668 стр.)" x:Name="pdfCpp04"/>

<ComboBoxItem Content="C++. Священные знания. Стивен С. Дьюхерст.  
2012. (232 стр.)" x:Name="pdfCpp05"/>

<ComboBoxItem Content="Qt 5.10. Профессиональное программирование  
на C++. Макс Шлее. 2018. (1074 стр.)" x:Name="pdfCpp06"/>

<ComboBoxItem Content="Алгоритмы и программы. Язык C++.  
Е.А.Конова, Г.А.Поллак. 2017. (386 стр.)" x:Name="pdfCpp07"/>

<ComboBoxItem Content="Введение в язык Си++. А.В.Столяров. 2018.  
(139 стр.)" x:Name="pdfCpp08"/>

<ComboBoxItem Content="Изучаем C++ создавая игры в UE4. Уильям  
Шериф. 2016. (297 стр.)" x:Name="pdfCpp09"/>

<ComboBoxItem Content="Изучаем Си. А.Купник. 2001. (233 стр.)"  
x:Name="pdfCpp10"/>

<ComboBoxItem Content="Искусство программирования игр на C++.  
Михаил Фленов. 2006. (258 стр.)" x:Name="pdfCpp11"/>

<ComboBoxItem Content="Как программировать на С. Харви Дейтел, Пол  
Дейтел. 2009. (1002 стр.)" x:Name="pdfCpp12"/>

<ComboBoxItem Content="Метaprogramмирование шаблонов C++ в  
задачах математической физики. М.М.Краснов. 2017. (85 стр.)"  
x:Name="pdfCpp13"/>

<ComboBoxItem Content="Моя первая программа на С/C++. А.Нейбауэр.  
2002. (267 стр.)" x:Name="pdfCpp14"/>

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

73

```

<ComboBoxItem Content="Новые сложные задачи на C++. 40 новых
головоломных примеров с решениями. Герб Саттер. 2005. (271 стр.)"
x:Name="pdfCpp15"/>

<ComboBoxItem Content="Объектно-ориентированное программирование
в C++. Р.Лафоре. 2004. (924 стр.)" x:Name="pdfCpp16"/>

</ComboBox> // ниже кнопка для открытия данного источника

<Button Click="btn3RuOpenPDF_Click" Background="#FF9200"
FontWeight="Bold" Style="{StaticResource someStyle}"
Margin="5,2,5,2" Width="80" Height="33" FontSize="16"
Foreground="Black" BorderThickness="2"
BorderBrush="Blue" ToolTip="Открыть во внешней программе">
    Открыть</Button> </StackPanel> </Grid> </Window>

```

Таким образом создаю три **ComboBox**-а с перечнем книг, размещенных в ресурсах проекта, и привязываю порядковый номер к настоящему наименованию книги. А в файле «**PDFreader.xaml.cs**» создаю обработчики для трёх кнопок «**Открыть**»:

```

private void btnOpenPDF_Click(object sender, EventArgs e)
{
    // для открытия во внешней PDF программе
    if (comboPDF.SelectedItem == pdf0001)
    {
        // указываю в какой папке по отношению к проекту
        String openPDFFile = @"0001.pdf"; // лежит файл
        System.IO.File.WriteAllBytes(openPDFFile, global::AllLoader.Properties.
Resources._0001); // на основании расширения запускаю данную программу
        // процесс) для PDF
        System.Diagnostics.Process.Start(openPDFFile);
    }
    else if (comboPDF.SelectedItem == pdf0002)
    {
        String openPDFFile = @"0002.pdf";
        System.IO.File.WriteAllBytes(openPDFFile, global::AllLoader.Properties.
Resources._0002);
    }
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        System.Diagnostics.Process.Start(openPDFFile);      }

else if ( // и т.д. до файла @"0030.pdf"
private void btnEnOpenPDF_Click(object sender, RoutedEventArgs e) { 2

{ // для открытия во внешней PDF программе

if (comboEnPDF.SelectedItem == en001)

{ // указываю в какой папке по отношению к проекту

String openPDFFileEn = @"en001.pdf"; // лежит файл

System.IO.File.WriteAllBytes(openPDFFileEn, global::AllLoader.Properties.

Resources.en001); // по расширению запускаю программу (процесс) для PDF

System.Diagnostics.Process.Start(openPDFFileEn);      }

else if (comboEnPDF.SelectedItem == en002)

{ // указываю в какой папке по отношению к проекту

String openPDFFileEn = @"en002.pdf"; // лежит файл

System.IO.File.WriteAllBytes(openPDFFileEn, global::AllLoader.Properties.

Resources.en002); // на основании расширения запускаю данную программу

(процесс) для PDF

System.Diagnostics.Process.Start(openPDFFileEn);      }

else if ( // и т.д. до файла @"en034.pdf"
private void btn3RuOpenPDF_Click(object sender, RoutedEventArgs e) { 3

{ if (combo3RuPDF.SelectedItem == pdfCpp01)

{ // указываю в какой папке по отношению к проекту

String openPDFFileCPP = @"Cpp01.pdf"; // лежит файл

System.IO.File.WriteAllBytes(openPDFFileCPP, global::AllLoader.Properties.

Resources.Cpp01); // на основании расширения запускаю данную программу

(процесс) для PDF

System.Diagnostics.Process.Start(openPDFFileCPP);  }

else if (combo3RuPDF.SelectedItem == pdfCpp02)

{ // указываю в какой папке по отношению к проекту

String openPDFFileCPP = @"Cpp02.pdf"; // лежит файл

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        System.IO.File.WriteAllBytes(openPDFFileCPP, global::AllLoader.Properties.
Resources.Cpp02); // запускаю программу (процесс) для PDF
        System.Diagnostics.Process.Start(openPDFFileCPP);    }

else if ( // и т.д. до файла @"Cpp16.pdf"

```

Данные книги можно открывать, редактировать и сохранять в произвольном месте. Каждая книга «скрыта» в памяти программы, при нажатии кнопки «Открыть» программа записывает «*файл-книги.pdf*» в корневой каталог (с *наименованием, которое было сохранено в ресурсах*). Но если оставить папку и наименование без изменений, и отредактировать файл, то при следующей попытке открытия данной книги она будет браться не из памяти, а с места на жестком диске, где впервые была записана (*создана*) из памяти.

## Класс **FlowDocument**

Мне хотелось создать насыщенное приложения с большими объемами текста, со сложным форматированием и встроенными изображениями. Для этого в WPF можно использовать две группы документов:

- ① **Фиксированные документы (fixed documents).** Формат и расположение содержимого таких документов фиксировано и не может быть изменено. На различных устройствах с различным разрешением экрана содержимое будет выглядеть одинаково и не будет оптимизировано. Такие документы преимущественно предназначены для печати. Для фиксированных документов WPF использует стандарт XPS (XML Paper Specification);
- ② **Потоковые документы (flow documents).** Эти документы предназначены для просмотра на экране монитора. А WPF выполняет оптимизацию документа под конкретные параметры среды.

Я остановил свой выбор на *потоковых документах*. В WPF они представлены классом **FlowDocument**, который может включать в себя различные потоковые элементы (*flow elements*). Эти элементы не являются стандартными элементами управления (*они не представлены в Панели элементов Visual Studio*),

Изм.	Лист	№ докум.	Подпись	Дата	Лист
					ВСК.090204.07.00.000 ПЗ 76

как, например, ‘Button’ или ‘TextBlock’, а наследуются от базового класса **FrameworkContentElement** и поэтому поддерживают такие механизмы, как привязка, анимация и другие, но не используют компоновку. Всю иерархию потоковых элементов можно представить следующим образом (Рисунок 11):

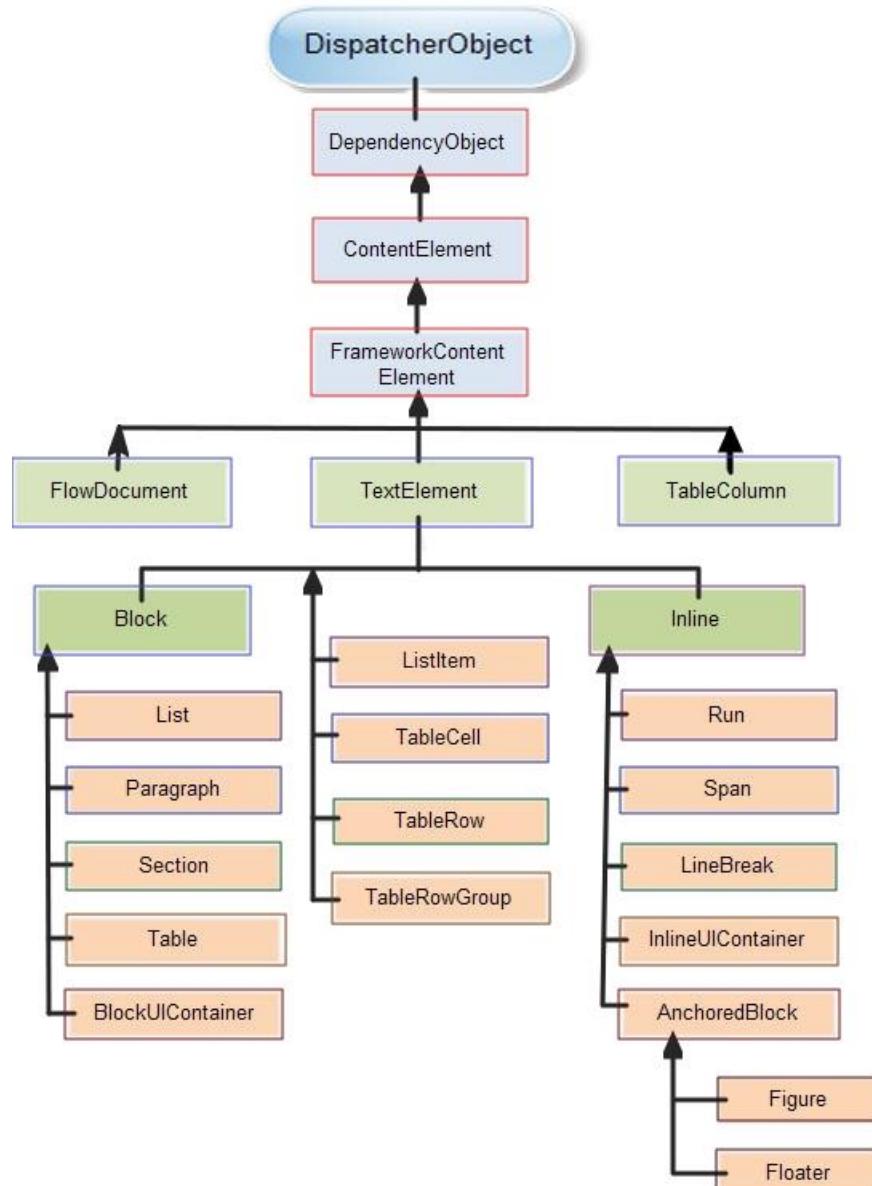


Рисунок 11 – Иерархия потоковых элементов в WPF

Для создания потоковых документов и использования объекта “**FlowDocument**“ я должен поместить его в один из контейнеров — **FlowDocumentReader**, **FlowDocumentPageViewer** или **FlowDocumentScrollView**. Например, для описания Microsoft Word 2016:

```

<FlowDocumentReader x:Name="docViewerWord" Background="#4FE416"
    Grid.Column="0" Margin="10,3,10,3">
    <FlowDocument ColumnWidth="auto" ColumnGap="15" MinPageWidth="600"
    FontSize="18" x:Name="wordReadDocView">
        <Paragraph TextAlignment="Center" FontSize="18" FontStyle="Normal"
        FontWeight="Bold" Foreground="#D4056E">
            Описание Microsoft Word 2016
            </Paragraph> <!-- здесь идёт описание: текст, таблицы, рисунки и т.д. -->
        </FlowDocument>
    </FlowDocumentReader>

```

В качестве содержимого FlowDocument принимает один или несколько потоковых элементов. Все эти элементы являются наследниками класса “TextElement” и могут быть блочными (**block**) и строчными (**inline**). Я использую блочные элементы, к которым относятся следующие: [Paragraph](#), [List](#), [Table](#), [BlockUIContainer](#) и [Section](#):

```

<BlockUIContainer>
    <Image x:Name="word" Source="pack://application:,,,/Pages/OF/2016/WORD/
    Word.jpg" MaxWidth="300"/>
</BlockUIContainer>

```

<Paragraph TextAlignment="Left" FontSize="16" Foreground="#020E6D">

<Bold>Microsoft Word 2016</Bold> – одна из наиболее актуальных редакций офисного текстового редактора, предназначенного для создания, оформления и печати любых материалов, включающих в себя текстовые и графические элементы. Также пользователь может импортировать в содержимое Word 2016 таблицы Excel, диаграммы, графики, клипарты, изображения из сети, списки SmartArt и прочие компоненты. В программе доступно совместное рецензирование контента с опциональным добавлением комментариев и правок, их принятием или отклонением другими пользователями, обладающими соответствующими привилегиями. Среди прочих инструментов форматирования

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

78

присутствует интеллектуальный конструктор тем и шаблонов, применяемый при составлении макета документа, выбор подложки, цвета страницы и компоновки границ рабочей области. Редактор также может конфигурировать отступы, межстрочные интервалы, поля и ориентацию страниц, добавлять колонки и последовательные номера строк. Более того, приложение вмещает продвинутый инструментарий для составления оглавления или содержания документов с указанием референсных ссылок на выбранные разделы с помощью одного клика мыши.

</Paragraph>

<Paragraph TextAlign="Center">  
<Hyperlink NavigateUri="https://www.microsoft.com/ru-ru/download/details.aspx?id=55008" Foreground="#B30089">

Обновление для Microsoft Word 2016 (KB3178720) 32-разрядный выпуск – официальная страница.

</Hyperlink>  
</Paragraph>  
<BlockUIContainer>  
<Image x:Name="wd00" Source="pack://application:,,,/Pages/OF/2016/WORD/wd00.jpg" MaxWidth="400"/>  
</BlockUIContainer>  
<Paragraph TextAlign="Center" FontSize="18" FontStyle="Normal" FontWeight="Bold" Foreground="#D4056E">

Как использовать Microsoft Word

</Paragraph>

<Paragraph TextAlign="Left" FontSize="16" Foreground="#020E6D">

Как создать простой документ.

</Paragraph>

<Paragraph TextAlign="Left" FontSize="16" Foreground="#020E6D">

1. Откройте Microsoft Word. Для этого дважды нажмите на иконку Microsoft Word.

</Paragraph>

Изм.	Лист	№ докум.	Подпись	Дата

```

<BlockUIContainer>
    <Image x:Name="wd01" Source="pack://application:,,,/Pages/OF/2016/
WORD/wd01.jpg" MaxWidth="400"/>
</BlockUIContainer>
<Paragraph TextAlignment="Left" FontSize="16" Foreground="#020E6D">

```

2. Просмотрите доступные шаблоны. В правой части окна можно увидеть несколько важных шаблонов:

```

</Paragraph>
<BlockUIContainer>
    <Image x:Name="wd02" Source="pack://application:,,,/Pages/OF/2016/
WORD/wd02.jpg" MaxWidth="400"/>
</BlockUIContainer>

```

Элемент Table организует вывод содержимого в виде таблицы. Он имеет вложенный элемент TableRowGroup. Этот элемент позволяет задать однообразный вид таблицы и содержит коллекцию строк - элементов TableRow (строку таблицы). А каждый элемент TableRow содержит несколько элементов TableCell (ячейка таблицы). В элементе TableCell затем уже размещают блочные элементы с содержимым, например, элементы Paragraph:

```

<Table>
    <Table.Columns>
        < TableColumn Width="2*" />
        < TableColumn Width="20*" />
    </Table.Columns>
    < TableRowGroup FontSize="18" Foreground="#0C922C" >
        < TableRow >
            < TableCell >
                < Paragraph Margin="3,2,10,2" FontSize="24" Foreground=
"#FF2100" >-</Paragraph>
            < TableCell >
                < Paragraph >

```

**<Bold>**Новый документ**</Bold>** — пустой документ со стандартным форматированием.

```
</Paragraph>

</TableCell>

</TableRow>

<TableRow> <TableCell>

<Paragraph Margin="3,2,3,2" FontSize="24" Foreground="#FF2100">-</Paragraph>
```

```
    </TableCell>

    <TableCell>

        <Paragraph>

            <Bold>Креативное резюме/сопроводительное письмо</Bold>
```

— пустое резюме (и сопроводительное письмо) в заданном формате.

```
</Paragraph>

</TableCell>

</TableRow> <TableRow> <TableCell>

<Paragraph Margin="3,2,3,2" FontSize="24" Foreground="#FF2100">-</Paragraph>
```

```
    </TableCell> <TableCell>

        <Paragraph>

            <Bold>Реферат с фото обложкой</Bold> — тип документа,
```

ориентированный на академическую направленность.

```
</Paragraph>

</TableCell> </TableRow> <TableRow>

<TableCell>

<Paragraph Margin="3,2,3,2" FontSize="24" Foreground="#FF2100">-</Paragraph>
```

```
    </TableCell> <TableCell>

        <Paragraph>
```

**<Bold>**Титульная страница faxa**</Bold>** — вводная часть faxa.

Изм.	Лист	№ докум.	Подпись	Дата

BCK.090204.07.00.000 ПЗ

Лист

81

```

</Paragraph>
</TableCell> </TableRow> <TableRow>
<TableCell>
<Paragraph Margin="3,2,3,2" FontSize="24" Foreground="#FF2100">-</Paragraph>

```

— Конкретные шаблоны можно найти прямо в программе, воспользовавшись строкой поиска вверху окна.

```

</Paragraph>
</TableCell> </TableRow>
<TableRowGroup> </Table>
<BlockUIContainer>
<Image x:Name="wd03"
Source="pack://application:,,,/Pages/OF/2016/WORD/wd03.jpg" MaxWidth="400"/>

```

Для определения столбцов в элементе применяется коллекция “**Table.Columns**”. Каждый столбец представляет элемент “**TableColumn**”, у которого я задаю ширину ( **TableColumn Width="20\***”).

Элемент “**BlockUIContainer**” позволяет добавить в документ различные элементы управления, которые не являются блочными или строчными элементами. Так, я добавляю к документу созданные картинки (Рисунок 12):

```

<BlockUIContainer>
<Image x:Name="wd42" Source="pack://application:,,,/Pages/OF/2016/
WORD/wd42.jpg" MaxWidth="400"/>

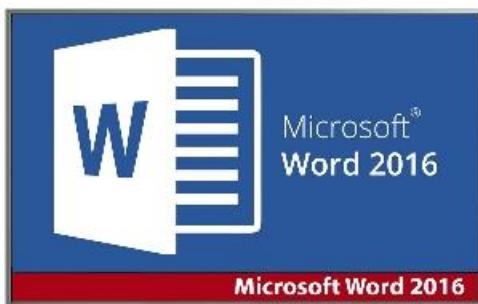
```

Рисунки редактирую в программе PicPick и размещаю в ресурсах проекта (чтобы обратится к ним: **pack://application:,,,/Pages...**).

Изм.	Лист	№ докум.	Подпись	Дата

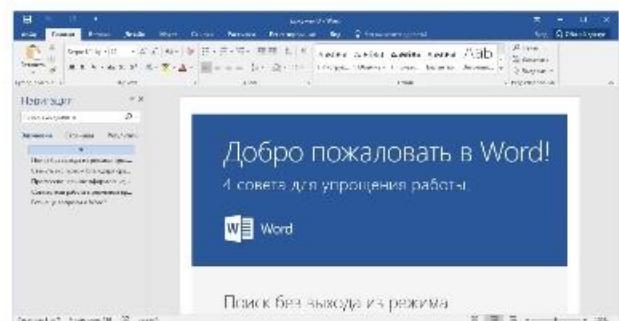


## Описание Microsoft Word 2016



**Microsoft Word 2016** - одна из наиболее актуальных редакций офисного текстового редактора, предназначенного для создания, оформления и печати любых материалов, включающих в себя текстовые и графические элементы. Также пользователь может импортировать в содержимое Word 2016 таблицы Excel, диаграммы, графики, клипарты, изображения из сети, списки SmartArt и прочие компоненты. В программе доступно совместное рецензирование контента с optionalным добавлением комментариев и правок, их принятием или отклонением другими пользователями, обладающими соответствующими привилегиями. Среди прочих инструментов форматирования присутствует интеллектуальный конструктор тем и шаблонов, применяемый при составлении макета документа, выбор подложки, цвета страницы и компоновки границ рабочей области. Редактор также может конфигурировать отступы, межстрочные интервалы, поля и ориентацию

## Обновление для Microsoft Word 2016 (KB3178720) 32-разрядный выпуск – официальная страница.



## Как использовать Microsoft Word

Как создать простой документ.

1. Откройте Microsoft Word. Для этого дважды нажмите на иконку Microsoft Word.



Рисунок 12 – Страница с картинками и ссылками

Добавляю функциональность, размещая элемент “[Hyperlink](#)” (ссылку для перехода) внутри “[Paragraph](#)” (web-страница откроется прямо в текущем окне программы):

```
<Paragraph TextAlignment="Center">
    <Hyperlink NavigateUri="https://www.microsoft.com/ru-ru/download/details.aspx?id=55008" Foreground="#B30089">
```

Обновление для Microsoft Word 2016 (KB3178720) 32-разрядный выпуск – официальная страница.

```
    </Hyperlink>
```

```
</Paragraph>
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

83

## Функционал кнопок нижнего меню «Оператор ЭВМ»



Рисунок №13 – Кнопки нижнего меню в «Оператор ЭВМ»

1. Кнопка поиска в представленном учебном материале. Даёт возможность найти в открытом документе интересующее слово или фразу. После нажатия появляется поле с надписью «*Введите текст для поиска...*», где необходимо указать, что Вас интересует, а слева будут стрелки для перехода к соответствующим местам в тексте, где были найдены совпадения.
2. При наведении курсора на данную клавишу вылетает подсказка с картинкой. А после нажатия запускается браузер, и пользователь попадает на главную страницу КГА ПОУ «ВСК». Всё это я реализовал следующим программным кодом:

```

<Button Panel.ZIndex="200" x:Name="lisenceButVSK" Width="90"
    Click="lisenceButVSK_Click" Content="КГА ПОУ "ВСК"""
    Foreground="WhiteSmoke" Style="{StaticResource butDown}"
    Tag="https://vstehn.ru/" TextBlock.TextAlignment="Center">
    <Button.ToolTip>
        <ToolTip Background="#90004455">
            <StackPanel>
                <TextBlock Margin="3" HorizontalAlignment="Center"
                    Foreground="#FAFF00"> КГА ПОУ "ВСК"
                </TextBlock>
                <Image Width="50" Margin="3"
                    Source="Images/Tooltip/vsk.png" />
                <TextBlock Margin="3" HorizontalAlignment="Center"
                    Foreground="#FAFF00">

```

Изм.	Лист	№ докум.	Подпись	Дата	Лист	84
					<b>ВСК.090204.07.00.000 ПЗ</b>	

```

Open https://vstehn.ru </TextBlock>
</StackPanel>
</ToolTip>
</Button.ToolTip>
<Button.Background>
<LinearGradientBrush StartPoint="0.5,0" EndPoint="0.5,1">
<GradientStop Offset="0" Color="Black" />
<GradientStop Offset="1" Color="#FFCD00CD" />
<GradientStop Color="#FF0C00FF" />
</LinearGradientBrush>
</Button.Background>
<Button.BorderBrush>Yellow</Button.BorderBrush>
<Button.BorderThickness>1</Button.BorderThickness>
</Button>

```

3. Такой же код создан и для клавиши «Visual Studio 2019», но идет загрузка страницы «<https://visualstudio.microsoft.com/ru/downloads/>», на которой можно загрузить Visual Studio 2019.
4. После нажатия данной клавиши происходит запись представленных в данный момент учебных материалов на жесткий диск (*m.e. «страница.xaml»*). Для каждого обзора в ресурсах проекта мной создана отдельная страница. Ниже демонстрирую программный код для страницы с информацией по операционной системе Windows 10:

namespace AllLoader.Pages

```

{ // Логика взаимодействия для Windows10.xaml
    public partial class Windows10 : Page
    {
        // переменная сохранён ли файл
        public bool sW10 = false;
        // "имя" для хранения файла
        string path = "windows10.xaml";
    }
}

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист  
85

```

public Windows10()
{
    InitializeComponent();
}

// с помощью класса XamlWriter произвожу запись в файл *.xaml
public void saveW10_Click(object sender, RoutedEventArgs e)
{
    // класс FileStream предоставляет реализацию абстрактного члена Stream,
    // для потоковой работы с различными файлами
    using (FileStream fs = File.Open(path, FileMode.Create))
    {
        // xaml-файл можно открыть как обычный текстовый файл
        if (docViewer.Document != null)
        {
            // класс XamlWriter - запись в файл
            XamlWriter.Save(docViewer.Document, fs);
            MessageBox.Show("Файл \"windows10.xaml\" успешно сохранен в
директории программы (txt). Его можно открыть любым текстовым редактором.");
            sW10 = true; // значит файл записан
        }
    }
}

```

После работы данного кода, на жестком диске появляется файл «windows10.xaml». Его можно открыть обычным текстовым редактором.

**5.** Данная кнопка позволяет распечатать весь учебный материал по выбранной теме. После нажатия открывается диалоговое окно для печати, где можно выбрать принтер. У каждого материала в программе есть свой «**x:Name**» (своё имя). В частности, для материала по операционной системе Windows 10 было задано в описании «**FlowDocument**», следующим образом:

...<FlowDocument ColumnWidth="auto" ColumnGap="15" MinPageWidth="600"
FontSize="18" **x:Name="w10docView"**>...

И для печати именно этого материала использую следующий код за кнопкой:

```

public void printW10_Click(object sender, RoutedEventArgs e)
{
    // вызываю "PrintDialog"
    PrintDialog printDlg = new PrintDialog();
    // переменная для передачи данных из FlowDocument x:Name="w10docView"
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

IDocumentPaginatorSource idpSource = w10docView;
// вызываю метод 'PrintDocument' для отправки документа на принтер
printDlg.PrintDocument(idpSource.DocumentPaginator, "");
}

```

6. Для создания конспектов я создаю дополнительное окно и страницы по различным темам. Окно называю: **Solution.xaml** (и **.xaml.cs**). Использую следующие пространства имён:

```

using System.Collections.Generic;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.IO;
using System.IO.Packaging;
using System.Windows.Xps.Packaging;
using System.Windows.Xps.Serialization;
using DotLiquid;
using dotTemplate = DotLiquid.Template;
using System.Windows.Markup;

```

Для этого окна мне необходимо создать конспекты по существующим темам, чтобы они сразу загружались в данное окно, в зависимости от того с какой страницы поступил такой “вызов”. Динамический конспект для Windows 10 «**reportw10.lqd**» создаю в Visual Studio следующим образом:

```

<FlowDocument xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
              xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
    <FlowDocument.Resources>
        <Style TargetType="TextBlock">
            <Setter Property="FontSize" Value="14"/>
            <Setter Property="Margin" Value="5"/>
            <Setter Property="TextWrapping" Value="Wrap"/>
        </Style>
    </FlowDocument.Resources>

```

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

*Лист*

**87**

```

</Style>

</FlowDocument.Resources>
<Paragraph FontSize="24">
    <Bold>{{ Title }}</Bold>
</Paragraph>
<Paragraph FontSize="16">
    {{ Subtitle }}
</Paragraph>
<Paragraph FontSize="16">
    <Bold>Информация в формате PDF:</Bold>
</Paragraph>
    {% for step in Steps -%}
<BlockUIContainer>
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="Auto"/>
            <ColumnDefinition />
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition />
        </Grid.RowDefinitions>
        <TextBlock Text="{{ step.Title }}" Foreground="#003481" FontWeight="Bold"
Grid.Column="1" Grid.Row="1"/>
        <TextBlock Text="{{ step.Description }}" Grid.Column="1" Grid.Row="2"/>
    </Grid>
</BlockUIContainer>
    {% endfor -%}
<Paragraph>
<Hyperlink NavigateUri="https://www.vstehn.ru/">КТА ПОУ "ВСК"</Hyperlink>

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

88

</Paragraph> </FlowDocument>

Но данные для каждого конспекта, для каждой тематики, нужно оформлять в программном коде самостоятельно. Т.е. нужно указать, что из себя будут представлять следующие переменные: {{ Title }}, {{ Subtitle }}, {% for step in Steps -%}, {% endfor -%}.

// вначале объявляю переменные для формирования контекста

```
public string PO, FS;
```

Так как в программе представлено несколько учебных тем, делаю выборку из всех имеющихся (*при нажатии клавиши - 6* (Рисунок 11)):

// выбор шаблона для загрузки страницы

```
if (PO == "W10") { FS = "Temp\\reportw10.lqd"; } // если выбран Windows 10
else if (PO == "W7") { FS = "Temp\\reportw7.lqd"; }
else if (PO == "LU") { FS = "Temp\\reportlu.lqd"; } // если выбран Ubuntu
else if (PO == "AR") { FS = "Temp\\reportar.lqd"; }
else if (PO == "LX") { FS = "Temp\\reportlxx.lqd"; } // и т.д. и т.п.....
else if (PO == "GD") { FS = "Temp\\reportggdrv.lqd"; }
else if (PO == "SK") { FS = "Temp\\reportskype.lqd"; }
else if (PO == "ON") { FS = "Temp\\reportone.lqd"; }
else if (PO == "OUT") { FS = "Temp\\reportout.lqd"; }
else if (PO == "PH") { FS = "Temp\\reportph.lqd"; }
else if (PO == "PP") { FS = "Temp\\reportpp.lqd"; }
else if (PO == "SF") { FS = "Temp\\reportsf.lqd"; }
else if (PO == "SI") { FS = "Temp\\reportsi.lqd"; }
else if (PO == "WR") { FS = "Temp\\reportwr.lqd"; }
else if (PO == "ACC") { FS = "Temp\\reportacc.lqd"; }
```

Если выбран шаблон конспекта для Windows 10:

// происходит чтение шаблона **.lqd** для соответствующей страницы

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

89

```

if (PO == "W10") // для Windows 10
{
    using (var stream = new FileStream("Temp\\reportw10.lqd", FileMode.Open))
    {
        using (var reader = new StreamReader(stream))
        { // динамически загружаются строки конспекта
            var templateString = reader.ReadToEnd(); // чтение по строкам
            var template = dotTemplate.Parse(templateString); // временные строки
            var docContext = CreateDocumentContextW10(); // формирую контекст
            var docString = template.Render(docContext); // записываю контекст в
переменную – чтобы показать

            DocViewer.Document = (FlowDocument)XamlReader.Parse(docString);
        } // на основании загруженных строк формируется FlowDocument
    } } else if (PO == "W7") // и т.д.
}

```

«Конспект» создаю самостоятельно (“заранее”) в тексте программы (*просто сокращаю представленную на странице ("windows10.xaml") информацию – и так для всех страниц*):

```

public DotLiquid.Hash CreateDocumentContextW10()
{
    // вывод конспекта по теме "Windows 10"
    var context = new
    {
        Title = "MICROSOFT WINDOWS 10",
        Subtitle = "Операционная система Windows 10 имеет следующие
системные требования:",
        Steps = new List<dynamic>{
            new { Title = "1. ", Description = "Процессор с частотой не менее 1 ГГц; "},
            new { Title = "2. ", Description = "ОЗУ от 1 Гб (для 32x систем) и 2 Гб (для
64x систем); "},
            new { Title = "3. ", Description = "От 16 до 20 Гб свободного места на
жестком диске; "},
            new { Title = "4. ", Description = "Наличие DirectX 9 и выше; "}
    }
}

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист  
90

```

new { Title = "", Description = "Для мобильных устройств системные
требования несколько ниже."},
new { Title = "ОТЛИЧИТЕЛЬНЫЕ ОСОБЕННОСТИ WINDOWS 10",
Description = ""},
new { Title = "", Description = "УНИВЕРСАЛЬНОСТЬ, УЛУЧШЕННЫЙ
ПУСК, УНИВЕСАЛЬНЫЙ ПОИСК, ВИРТУАЛЬНЫЙ РАБОЧИЙ СТОЛ, " +
"ПРИКРЕПЛЕНИЕ ОКОН, ДОРАБОТКИ ПРОВОДНИКА; "},
new { Title = "ДРУГИЕ УЛУЧШЕНИЯ", Description = ""},
new { Title = "", Description = "    а) Обновленный приветственный экран и
экран блокирования устройства; " +
"б) Возможность входа в систему за счет службы биометрических данных
Windows Hello; " +
"в) В расширение предыдущего пункта можно сказать об использовании
системы отпечатков в качестве паролей на мобильных устройствах или планшетах;
" +
"г) «Панель управления» заменена «Параметрами» с более
ориентированным на пользователя интерфейсом; " +
"д) Часть значков перерисована; " +
"е) Обновление часов и календаря; " +
"ж) Магазин приложений Windows стал более удобным; " +
"з) Вместо Internet Explorer используется Microsoft Edge; " +
"к) Имеется новое приложение «Начало работы»; "},
new { Title = "РЕДАКЦИИ WINDOWS 10", Description = ""}, // и т.д. "},
new { Title = "", Description = "    Microsoft проделала замечательную работу
по фиксации ошибок. "},
};

// в конце - возврат конспекта (вывод на экран)
return DotLiquid.Hash.FromAnonymousObject(context);      }

```

При нажатии данной клавиши появится следующее окно (Рисунок 14):

Изм.	Лист	№ докум.	Подпись	Дата	Лист
					ВСК.090204.07.00.000 ПЗ 91

**MICROSOFT WINDOWS 10**

Операционная система Windows 10 имеет следующие системные требования:

**Информация в формате PDF:**

1. Процессор с частотой не менее 1 ГГц;
2. ОЗУ от 1 Гб (для 32х систем) и 2 Гб (для 64х систем);
3. От 16 до 20 Гб свободного места на жестком диске;
4. Наличие DirectX 9 и выше;

Для мобильных устройств системные требования несколько ниже.

**ОТЛИЧИТЕЛЬНЫЕ ОСОБЕННОСТИ WINDOWS 10**

УНИВЕРСАЛЬНОСТЬ, УЛУЧШЕННЫЙ ПУСК, УНИВЕСАЛЬНЫЙ ПОИСК, ВИРТУАЛЬНЫЙ РАБОЧИЙ СТОЛ, ПРИКРЕПЛЕНИЕ ОКОН, МЕНЕДЖЕР ФАЙЛОВ, УЛУЧШЕННАЯ КОМАНДНАЯ СТРОКА, ПРОСМОТР ЗАДАЧ, ДОРаботки ПРОВОДНИКА;

**ДРУГИЕ УЛУЧШЕНИЯ**

а) Обновленный приветственный экран и экран блокирования устройства; б) Возможность входа в систему за счет службы биометрических данных Windows Hello; в) В расширение предыдущего пункта можно сказать об использовании системы отпечатков в качестве паролей на мобильных устройствах или планшетах; г) «Панель управления» заменена «Параметрами» с более ориентированным на пользователя интерфейсом; д) Часть значков перерисована; е) Обновление часов и календаря; ж) Магазин приложений Windows стал более удобным; з) Вместо Internet Explorer используется Microsoft Edge; к) Имеется новое приложение «Начало работы»;

**РЕДАКЦИИ WINDOWS 10**

- Windows 10 Домашняя. - Windows 10 Домашняя для одного языка.
- Windows 10 Домашняя с Bing.
- Windows 10 Профессиональная.
- Windows 10 Мобильная.
- Windows 10 Корпоративная.
- Windows 10 для образовательных учреждений.
- Windows 10 Мобильная корпоративная.
- Windows 10 IoT Домашняя.

**ПУСК**

1. Пуск + плитки (по умолчанию): наиболее часто используемые приложения, список программ, ярлыки и живые плитки. 2. Пуск: наиболее часто используемые приложения, список программ, ярлыки, отсутствие живых плиток. 3. Пуск + плитки (полный экран): наиболее часто используемые приложения, список программ, ярлыки, живые плитки. И всё в полноэкранном режиме.

**РЕЖИМ ПЛАНШЕТА**

Одной из новых функций Windows 10 является адаптация интерфейса для ПК и планшетов. С Windows 10 можно пользоваться новой операционной системой в двух режимах – в настольном, а также планшетном. При включении планшетного режима размеры плиток сильно увеличиваются.

**ЦЕНТР ЦВЕДОМЛЕНИЙ**

Несколько основных направлений уведомлений: первые от приложений (электронная почта, Твиттер, ВКонтакте, Новости, Погода и другое), вторые для включения/отключения функций ОС и третьи для аксессуаров и сторонних устройств (наушники, смартфон, USB-накопитель, DVD-диск).

**БЫСТРЫЕ ПЕРЕКЛЮЧАТЕЛИ**



Рисунок №14 – Окно с динамическим конспектом по Windows 10

В этом окне тоже реализую две клавиши «Печать» и «Обновить». При печати выводится лишь информация, представленная на экране, т.к. это «динамический» объект (“использовать” можно лишь то, что видно на экране).

7. Клавиша запускает «Проводник».
8. Клавиша запускает «Реестр».
9. Клавиша вновь обращается к конспекту, который только что был представлен на экране (*так как все конспекты очень «маленькие», эту работу трудно увидеть*).
10. Клавиша позволяет изменить количество страниц на экране.
11. Клавиша запускает «Блокнот».
12. Клавиша масштабирует загруженную на экран информацию.

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

92

## Создание элементов верхнего меню в главном окне «Оператор ЭВМ»

Для создания меню использую «**Grid**». Это наиболее мощный и часто используемый контейнер, напоминающий обычную таблицу. Он содержит столбцы и строки, количество которых задает разработчик. Для определения строк используется свойство **RowDefinitions**, а для определения столбцов — свойство **ColumnDefinitions**:

```
<!-- разбиваю меню на строки и колонки -->
```

```
<Grid Background="#000BFF">
    <Grid.RowDefinitions><!-- 1-я строка -->
        <RowDefinition Height="55" /><!-- общее меню -->
        <RowDefinition Height="55" /><!-- меню программы "Оператор ЭВМ" -->
        <RowDefinition Height="*" /><!-- высота для всего остального -->
        <RowDefinition Height="30" /><!-- внизу информационный блок -->
    </Grid.RowDefinitions>
    <!-- первая ("0") строка / для верхнего меню -->
    <Grid Grid.Row="1" Background="#193df2"><!-- меняю цвет верх.меню -->
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="350" />
            <!-- (1) 0-колонка, слева, для кнопки меню -->
            <ColumnDefinition Width="*" />
            <!-- (2) 1-колонка, под поиск -->
            <ColumnDefinition Width="50" />
            <!-- (3) 2-колонка - текстовый блок, при навигации по меню -->
            <ColumnDefinition Width="*" />
            <!-- (4) 3 - размер окна можно изменять -->
            <ColumnDefinition Width="*" />
            <!-- (5) 4-колонка, под поиск -->
            <ColumnDefinition Width="52" />
        </Grid.ColumnDefinitions>
    </Grid>
</Grid>
```

Изм.	Лист	№ докум.	Подпись	Дата

```

<!-- (6) 5-колонка, под кнопку для теста -->
<ColumnDefinition Width="52" />
<!-- (7) 6-колонка, под кнопку для справки -->
<ColumnDefinition Width="52" />
<!-- (8) 7-колонка, под кнопку для выхода -->
</Grid.ColumnDefinitions>

```

Подбор цветов произвожу в программе «**Just Color Picker**».

```

<!-- делаю кнопку меню (слева вверху) / для «Background» и т.д. указываю цвет-->
<Button x:Name="LeftMenu" Grid.Row="1" Grid.Column="0" Width="50"
Height="50" Margin="0,0,0,0" HorizontalAlignment="Left" VerticalAlignment="Stretch"
Background="#0F038D" BorderBrush="Red" BorderThickness="3,3,3,3"
Click="leftMenu_Click" Foreground="White" Panel.ZIndex="9" Style="{StaticResource
afterLoginStyle}" ToolTipService.ToolTip="Главное меню">
    <StackPanel>
        <Image x:Name="ButtonMenu" Height="40" Margin="-2,-2,-2,-2"
Panel.ZIndex="10" Opacity="1" RenderTransformOrigin="0.5,0.5"
Source=".\\Images/Button\\menu.png" ToolTip="Главное меню">
            <Image.RenderTransform>
                <TransformGroup><ScaleTransform /><SkewTransform />
                    <RotateTransform Angle="-33.85" /><!-- наклон рисунка -->
                <TranslateTransform /></TransformGroup></Image.RenderTransform>
            </Image></StackPanel></Button>
<!-- кнопка для поиска - справа от текста -->
<Button x:Name="butFinder" Grid.Row="1" Width="50" Height="50"
Margin="3,0,3,0" HorizontalAlignment="Right" Click="butFinder_Click"
Background="#0F038D" BorderBrush="#FD9BFF" BorderThickness="3,3,3,3"
FontFamily="Segoe UI Emoji" FontSize="22" Foreground="White"
Style="{StaticResource afterLoginStyle}" ToolTipService.ToolTip="Начало">
    <StackPanel>

```

Изм.	Лист	№ докум.	Подпись	Дата

```

<Image x:Name="butImgFinder" Margin="3" Opacity="3"
Source=".\\Images\\Button\\search.png" ToolTip="Начало" />
</StackPanel>
</Button>

<!-- кнопка для открытия списка тестов -->
<Button x:Name="butRightTest" Grid.Row="1" Grid.Column="5" Width="50"
Height="50" Click="butRightTest_Click" Margin="-4,0,4,0"
HorizontalAlignment="Right" VerticalAlignment="Stretch" Background="#0F038D"
BorderBrush="#00FF0B" BorderThickness="3,3,3,3" Foreground="White" Opacity="5"
Style="{StaticResource afterLoginStyle}" ToolTipService.ToolTip="Открыть список
тестов">
<StackPanel>
<Image x:Name="butImgForTest" Margin="-6,-6,-6,-6" Opacity="6"
Source=".\\Images\\Button\\test.png" ToolTip="Открыть список тестов" />
</StackPanel></Button>

<Button x:Name="butRightExit" Grid.Row="1" Grid.Column="7" Width="50"
Height="50" Margin="1,0,1,0" HorizontalAlignment="Left" VerticalAlignment="Stretch"
Background="#0F038D" BorderBrush="Red" BorderThickness="3,3,3,3"
Click="butRight_Exit_Click" FontFamily="Segoe MDL2 Assets" FontSize="24"
FontWeight="Bold" Foreground="White" Opacity="3" Style="{StaticResource
afterLoginStyle}" ToolTipService.ToolTip="Выход / Смена пользователя">
<StackPanel>
<Image x:Name="butImgExit" Margin="3"
KeyDown="butRight_Exit_Click" MouseDown="butRight_Exit_Click" Opacity="3"
Source="Images\\Button\\TurnOff.png" ToolTip="Выход / Смена пользователя" />
</StackPanel></Button></Grid>

<Menu Grid.Row="0" Grid.RowSpan="2" Width="2000" Height="55" Margin="-
10,2,0,0" HorizontalAlignment="Left" VerticalAlignment="Top" Background="#193df2"
Opacity="3">
<DockPanel Width="2000" Background="#193df2">

```

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

```

<Menu DockPanel.Dock="Top">
    <!-- кнопка раскрывает/сворачивает боковое меню -->
    <MenuItem x:Name="butFail" Width="100" Height="55" Margin="0,-2,0,0"
        VerticalAlignment="Bottom" Background="#193df2" BorderBrush="White"
        BorderThickness="2" FontSize="14" ToolTip="Работа с файлами"
        FontWeight="Bold" Foreground="White" Header="Файл">
        <MenuItem.Icon>
            <Image Width="39" ToolTip="Работа с файлами"
                x:Name="imgButFail" Height="39" Margin="-2,-17,-22,-18"
                Source=".\\Images\\menu.ico" /></MenuItem.Icon>
    <!-- РАБОТА С ФАЙЛАМИ – скомпилированных мной программ --> (Рисунок 15)
    <MenuItem x:Name="menuOpenFolder" Height="30" Foreground="#0822FF"
        Background="LightCoral" Click="newOpenFolder_Click" Header="Open Explorer / Пример проводника">
        <MenuItem.Icon> <Image Margin="1,-10,-5,-10"> <Image.Source>
            <BitmapImage UriSource="/Images/Files/drive.png" /> </Image.Source> </Image>
        </MenuItem.Icon></MenuItem> <!-- разделяю пункты меню => «Separator» -->
        <Separator Height="3" Background="DarkMagenta" />
    <!-- работа с текстовыми файлами -->
    <MenuItem x:Name="menuNew" Height="30" Foreground="#0822FF"
        Background="#9DF74F" Click="newTxtFail_Click" Header="_New / Новый *.txt">
        <MenuItem.Icon> <Image Margin="1,-10,-5,-10"> <Image.Source>
            <BitmapImage UriSource="/Images/Menu/newtxt.png" />
        </Image.Source> </Image></MenuItem.Icon></MenuItem>
        <Separator Height="3" Background="#03008B" />
    <!-- запуск созданного просмотра файлов *.docx -->
    <MenuItem x:Name="menuOpen" Height="30" Foreground="#0822FF"
        Background="#FCF661" Click="openDocxFail_Click" Header="_Open Word / Открыть *.docx">
        <MenuItem.Icon> <Image Margin="1,-10,-5,-10"> <Image.Source>

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        <BitmapImage UriSource="/Images/Menu/Word.png" />
    </Image.Source> </Image> </MenuItem.Icon> </MenuItem>
    <Separator Height="3" Background="#158003" />

<!-- запуск созданного pdf редактора -->

    <MenuItem x:Name="butOPPDF" Height="30" Foreground="#0822FF"
Background="#ED99FC" Click="butOPPDF_Click" Header="_PDF Editor / PDF
редактор"> <MenuItem.Icon> <Image Margin="1,-10,-5,-10"> <Image.Source>
        <BitmapImage UriSource="/Images/Menu/oppdfedit.png" />
    </Image.Source> </Image> </MenuItem.Icon> </MenuItem>
    <Separator Height="3" Background="#C3041D" />

<!-- запуск созданного txt/rtf редактора -->

    <MenuItem x:Name="butOPRTB" Height="30" Foreground="#0822FF"
Background="#7AFFA2" Click="butOPRTB_Click" Header="_VSK text editor /
Текстовый редактор"> <MenuItem.Icon> <Image Margin="1,-10,-5,-10">
    <Image.Source> <BitmapImage UriSource="/Images/Menu/optextedit.png" />
</Image.Source> </Image> </MenuItem.Icon> </MenuItem>
    <Separator Height="3" Background="#906500" />

<!-- запуск созданного калькулятора -->

    <MenuItem x:Name="butOPSimpleCalc" Height="30" Foreground="#0822FF"
Background="#FFB364" Click="butOPSimpleCalc_Click" Header="_Simple calculator /
Простой калькулятор"> <MenuItem.Icon> <Image Margin="1,-10,-5,-10">
    <Image.Source> <BitmapImage UriSource="/Images/Menu/opcacalc.png" />
</Image.Source> </Image> </MenuItem.Icon> </MenuItem>
    <Separator Height="3" Background="#008790" />

<!-- запуск больших часов (закрытие <Alt+F4>) -->

    <MenuItem x:Name="butOPBigClock" Height="30" Foreground="#0822FF"
Background="#E864FF" Click="butOPBigClock_Click" Header="_Big clock / Большие
часы"> <MenuItem.Icon> <Image Margin="1,-10,-5,-10"> <Image.Source>
        <BitmapImage UriSource="/Images/Menu/opbigclock.png" />
    </Image.Source> </Image> </MenuItem.Icon> </MenuItem>

```

```

<Separator Height="3" Background="#02720E" />

<MenuItem x:Name="butOPHtmlBinding" Height="30" Foreground="#0822FF"
Background="#FCF661" Click="butOPHtmlBinding_Click" Header="_HTML Binding /
HTML практика"><MenuItem.Icon><Image Margin="1,-10,-5,-10"><Image.Source>
<BitmapImage UriSource="/Images/Menu/htmlbinding.png" />
</Image.Source></Image></MenuItem.Icon></MenuItem>
<Separator Height="3" Background="#CB0FB7" /></MenuItem>
<!-- открываю интернет браузер --&gt;
&lt;MenuItem x:Name="butInet" Width="125" Height="55" Margin="-2,-2,0,0"
VerticalAlignment="Bottom" Background="#193df2" BorderBrush="White"
BorderThickness="2" FontSize="14" FontWeight="Bold" Foreground="White"
ToolTip="Открыть Vsk-Browser" Header=" Интернет " Click="butInet_Click"&gt;
&lt;MenuItem.Icon&gt;
&lt;Image Width="39" x:Name="imgButInet" ToolTip="Открыть Vsk-
Browser" Height="39" Margin="-2,-16,-21,-17" Source=".\\Images\\Button\\internet.png"
/&gt;&lt;/MenuItem.Icon&gt;&lt;/MenuItem&gt;
<!-- запуск калькулятора --&gt;
&lt;MenuItem x:Name="butCalc" Width="150" Height="55" Margin="-2,-2,0,0"
VerticalAlignment="Bottom" Background="#193df2" BorderBrush="White"
BorderThickness="2" FontSize="14" Click="butCalc_Click" FontWeight="Bold"
Foreground="White" Header=" Калькулятор "&gt;&lt;MenuItem.Icon&gt;&lt;Image
Width="40" Height="40" Margin="0,-16,-26,-17"
Source=".\\Images\\Button\\calculator.png" /&gt;&lt;/MenuItem.Icon&gt;&lt;/MenuItem&gt;
<!-- показывает текущее время --&gt;
&lt;MenuItem x:Name="butTime" Width="165" Height="55" Margin="0,-2,0,0"
HorizontalAlignment="Stretch" VerticalAlignment="Bottom" Background="#193df2"
BorderBrush="White" BorderThickness="0" FontSize="20" FontStyle="Italic"
Foreground="White" Header="" IsEnabled="False"&gt;&lt;MenuItem.Icon&gt;&lt;Image
</pre>

```

Изм.	Лист	№ докум.	Подпись	Дата

```

Width="37" Height="37" Margin="1,-14,-23,-17" Source=".\\Images\\timeclock.ico" />
</MenuItem.Icon></MenuItem>

<!-- кнопка загружает библиотеку с PDF источниками --&gt;

    &lt;MenuItem x:Name="butHelp" Width="120" Height="55" Margin="-2,-2,0,0"
HorizontalAlignment="Stretch" VerticalAlignment="Bottom" Background="#193df2"
BorderBrush="White" BorderThickness="2" FontSize="14" Click="butHelp_Click"
ToolTip="Справка" FontWeight="Bold" Foreground="White" Header=" Справка "&gt;
        &lt;MenuItem.Icon&gt; &lt;Image Width="40" ToolTip="Справка"
x:Name="imgButHelp" Height="40" Margin="-2,-19,-20,-19"
Source=".\\Images\\help.ico" /&gt;&lt;/MenuItem.Icon&gt;&lt;/MenuItem&gt;

<!-- кнопка для информации о текущем пользователе (не реализовано) --&gt;

    &lt;MenuItem x:Name="butAccount" Width="120" Height="55" Margin="-2,-2,0,0"
HorizontalAlignment="Right" VerticalAlignment="Bottom"
Background="#193df2" BorderBrush="White" BorderThickness="2" ToolTip="Текущий
пользователь" FontSize="14" Click="infoUserMember_Click" FontWeight="Bold"
Foreground="White" Header=" Аккаунт "&gt;
        &lt;MenuItem.Icon&gt; &lt;Image Width="38" x:Name="imgButAccount"
ToolTip="Текущий пользователь" Height="40" Margin="0,-18,-22,-18"
Source=".\\Images\\Button\\users.png" /&gt;&lt;/MenuItem.Icon&gt;&lt;/MenuItem&gt;

    &lt;MenuItem x:Name="PASS1" Width="200" Height="55" Margin="-2,-2,0,0"
HorizontalAlignment="Right" VerticalAlignment="Bottom" Background="#193df2"
BorderBrush="White" BorderThickness="2" IsEnabled="False" /&gt;

<!-- кнопка переключения языка интерфейса --&gt; (Рисунок 16)

    &lt;MenuItem x:Name="butLang" Width="120" Height="55" Margin="-2,-2,0,0"
HorizontalAlignment="Right" VerticalAlignment="Bottom" Background="#193df2"
BorderBrush="White" BorderThickness="2" FontSize="14" FontWeight="Bold"
Foreground="White" Header=" Язык "&gt;
        &lt;MenuItem.Icon&gt;&lt;Image Width="38" Height="40" Margin="0,-18,-22,-18"
Source=".\\Images\\Lang\\language.png" /&gt;&lt;/MenuItem.Icon&gt;
</pre>

```

```

<MenuItem x:Name="langRU" Height="30" Background="#0822FF"
Foreground="White" Click="langRU_Click" Header="Русский язык">
<MenuItem.Icon><Image Margin="1,-10,-5,-10"><Image.Source>
    <BitmapImage UriSource="/Images/Lang/flagRussia.png" />
</Image.Source></Image></MenuItem.Icon></MenuItem>
<MenuItem x:Name="langEN" Height="30" Background="#0822FF"
Click="langEN_Click" Foreground="White" Header="Английский язык">
<MenuItem.Icon><Image Margin="1,-10,-5,-10"><Image.Source>
    <BitmapImage UriSource="/Images/Lang/flagUnitedStates.png" />
</Image.Source></Image></MenuItem.Icon></MenuItem>
<MenuItem x:Name="langCN" Height="30" Background="#0822FF"
Click="langCN_Click" Foreground="White" Header="Китайский язык">
<MenuItem.Icon><Image Margin="1,-10,-5,-10"><Image.Source>
    <BitmapImage UriSource="/Images/Lang/flagChina.png" />
</Image.Source></Image></MenuItem.Icon></MenuItem>
<MenuItem x:Name="langJP" Height="30" Background="#0822FF"
Click="langJP_Click" Foreground="White" Header="Японский язык">
<MenuItem.Icon><Image Margin="1,-10,-5,-10"><Image.Source>
    <BitmapImage UriSource="/Images/Lang/flagJapan.png" />
</Image.Source></Image></MenuItem.Icon></MenuItem>
<MenuItem x:Name="langKO" Height="30" Background="#0822FF"
Click="langKO_Click" Foreground="White" Header="Корейский язык">
<MenuItem.Icon><Image Margin="1,-10,-5,-10"><Image.Source>
    <BitmapImage UriSource="/Images/Lang/flagKorea.png" />
</Image.Source></Image></MenuItem.Icon></MenuItem>
<MenuItem x:Name="PASS2" Width="800" Height="55" Margin="-2,-2,0,0"
HorizontalAlignment="Right" VerticalAlignment="Bottom" Background="#193df2"
BorderBrush="White" BorderThickness="2" IsEnabled="False" />
</Menu>      </DockPanel>      </Menu>

```

Изм.	Лист	№ докум.	Подпись	Дата



Рисунок №15 – Меню «Файл» (для запуска скомпилированных программ)



Рисунок №16 – Меню для выбора языка в «Оператор ЭВМ»

## Код программной части для обработчиков событий меню «Файл»

При нажатии пункта меню «Файл» раскрывается ниспадающий список из восьми пунктов:

1. Пункт **«Open Explorer / Пример проводника»** позволяет запустить пробную версию проводника (Рисунок 17):

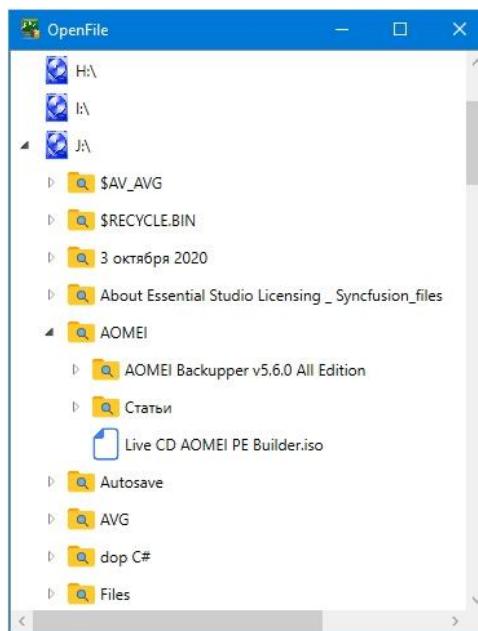


Рисунок №17 – Пример проводника

Добавляю к проекту окно «**OpenFile.xaml**»:

```
<Window x:Class="AllLoader.OpenFile" ..... // код генерируется
автоматически, ниже добавляю (заголовок и размеры окна):
Title="OpenFile" Height="400" Width="350">
<Grid> <!-- область просмотра проводника -->
    <TreeView x:Name="FolderView" Panel.ZIndex="3000" > <!-- просмотр дерева
каталога -->
        <TreeView.Resources> <!-- ресурсы для каталога -->
            <Style TargetType="{x:Type TreeViewItem}"> <!-- задаю стиль оформления -->
                <Setter Property="HeaderTemplate" >
                    <Setter.Value> <!-- для заголовков добавляю значок и название в
'Images/Files' -->
```

Изм.	Лист	№ докум.	Подпись	Дата

```

<DataTemplate><StackPanel Orientation="Horizontal">
    <!-- первоначальное значение Source="/Images/Files\drive.png" -->
    <!-- через 'Binding' привязываю картинки к типу объекта: файл, папка, диск -->
        <Image Source="{Binding RelativeSource={RelativeSource
Mode=FindAncestor, AncestorType={x:Type TreeViewItem}}}, Path=Tag, Converter=
{x:Static local:HeaderToImageConverter.Instance}" Width="25" Margin="2" />
            <!-- букву диска беру через 'Binding' -->
    <TextBlock Text="{Binding}" VerticalAlignment="Center" /></StackPanel>
</DataTemplate> </Setter.Value> </Setter> </Style>
</TreeView.Resources> </TreeView>
</Grid> </Window>

```

Программный код самого элемента меню «[OpenFile.xaml.cs](#)» выглядит так:

```

public partial class OpenFile : Window
{
    // #pragma region позволяет указать блок кода, который можно развернуть или
    // свернуть, делаю для себя, чтобы облегчить понимание кода
    #region Constructor
    public OpenFile()
    {
        InitializeComponent();
    }
    #endregion

    // при загрузке окна / при первом открытии
    #region On Loaded
    private void Window_Loaded(object sender, RoutedEventArgs e)
    {
        // обращаюсь к логическим дискам (и получаю список дисков в ПК)
        foreach (var drive in Directory.GetLogicalDrives())
        {
            // завожу переменную для каталога
            var item = new TreeViewItem()
            {
                // указываю название диска (получаю)
                Header = drive,    // указываю "полный" путь (получаю)
                Tag = drive };
    }
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        item.Items.Add(null); // добавляю элементы (фiktивные)

// просматриваю расширение получаемого элемента-папки

// создаю для этого метод 'Folder_Expanded'

        item.Expanded += Folder_Expanded;

// добавляю в созданное дерево каталогов для просмотра

        FolderView.Items.Add(item); } }

#endregion

#region Folder Expanded

// для папок

private void Folder_Expanded(object sender, RoutedEventArgs e)
{
    #region Initial Checks

    var item = (TreeViewItem)sender;

// если элемент содержит только фiktивные данные

    if (item.Items.Count != 1 || item.Items[0] != null) return;

    // отчищаю путь

    item.Items.Clear();

// получаю полный путь

    var fullPath = (string)item.Tag;

#endregion

#region Get Folders

// завожу "переменную-список" для списка директорий

    var directories = new List<string>();

// просматриваю весь каталог 'от начала - до конца'

    try { // попытка получить все подкаталоги

        var dirs = Directory.GetDirectories(fullPath); // если в папке есть ещё папки

        if (dirs.Length > 0) directories.AddRange(dirs);

    } // получение

    catch { }

// для каждого каталога

    directories.ForEach(directoryPath =>

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

104

```

{ // завожу переменную для "поддиректорий"
    var subItem = new TreeViewItem()
    {
        // полный путь в названии / заголовок как имя папки
        Header = GetFileName(directoryPath), // и как полный путь
        Tag = directoryPath };
    subItem.Items.Add(null);
    subItem.Expanded += Folder_Expanded; // еще директория
    item.Items.Add(subItem); // добавляю как родительскую });
}

#endregion
#region Get Files
// завожу "переменную-список" для списка файлов
var files = new List<string>();
// опять просматриваю весь каталог от начала - до конца
try
{
    // попытка получить все файлы
    var fs = Directory.GetFiles(fullPath);
    if (fs.Length > 0) files.AddRange(fs);
}
// получение
catch { }

// для каждого файла
files.ForEach(filePath =>
{
    // завожу переменную для файлов
    var subItem = new TreeViewItem()
    {
        Header = GetFileName(filePath), // имя файла
        Tag = filePath // полный путь };
    item.Items.Add(subItem); #endregion #endregion
}

// для файлов / поиск файла или папки в полном пути
public static string GetFileName(string path)
{
    // если путь отсутствует, возвращаю "пустоту"
    if (string.IsNullOrEmpty(path)) return string.Empty;
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

// переменная пути, делаю для "пути" все слэши обратными
    var normalizedPath = path.Replace('/', '\\');

// переменная пути, когда уже нет подкаталогов
    var lastIndex = normalizedPath.LastIndexOf('\\');

// если в пути уже не встречаются слэши
    if (lastIndex <= 0)      return path; // возвращаю путь

// возвращаю имя после последнего "бэк-слэша" (файла)
    return path.Substring(lastIndex + 1);      }  }  }

```

Запускаю форму проводника следующим образом:

```

private void newOpenFolder_Click(object sender, RoutedEventArgs e)
{
    // открытие собственного диалогового окна для просмотра каталога папок
    OpenFileDialog fOpenFile = new OpenFileDialog();
    // устанавливаю свои значки для файлов и папок
    fOpenFile.Show();  }

```

2. Пункт **«New / Новый \*.txt»** позволяет открыть страницу для работы с текстовыми файлами (Рисунок 18). Реализую следующие команды: *Новый, Открыть, Сохранить, Закрывать, Вырезать, Копировать, Вставить, Шрифт* (в ресурсах проекта размещаю шрифты):

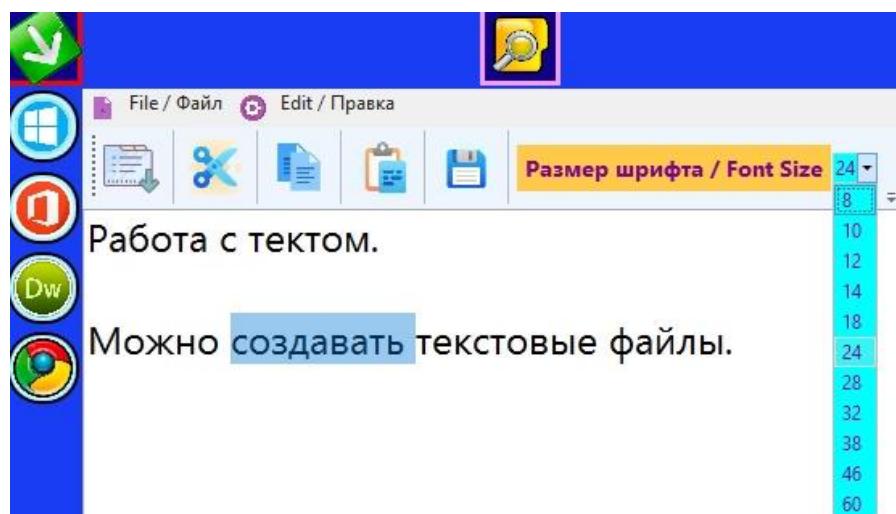


Рисунок №18 – Окно для работы с файлами \*.txt

После того как пользователь набрал текст, он может его сохранить, указав наименование своего файла (Рисунок 19):

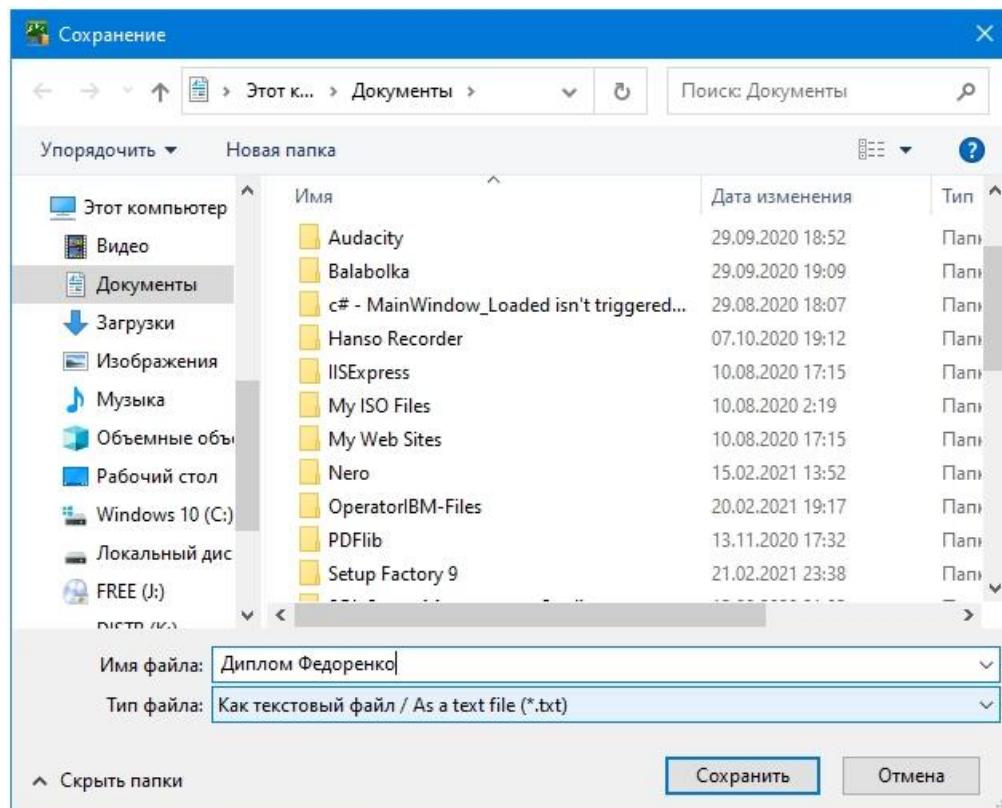


Рисунок №19 – Окно для сохранения текстового файла

Страницу для работы с текстом открываю следующим программным кодом:

```
private void newTxtFail_Click(object sender, RoutedEventArgs e)
{
    // страница для работы с txt-файлами
    frameAll.Navigate(new txtPage());
}

<MenuItem x:Name="menuNew" Height="30" Foreground="#0822FF"
Background="#9DF74F" Click="newTxtFail_Click" Header="_New /
Новый *.txt">

<MenuItem.Icon> <Image Margin="1,-10,-5,-10"> <Image.Source>
    <BitmapImage UriSource="/Images/Menu/newtxt.png" />
</Image.Source>
</Image> </MenuItem.Icon>
</MenuItem>
```

Изм.	Лист	№ докум.	Подпись	Дата

3. Пункт «Open Word / Открыть \*.docx» позволяет открывать для просмотра **простые** файлы **\*.docx** (Рисунок 20). В момент первого запуска из ресурсов загружается созданный мной в Word-e файл (**Rendering Test.docx**), в котором привожу вариант возможного форматирования документа:

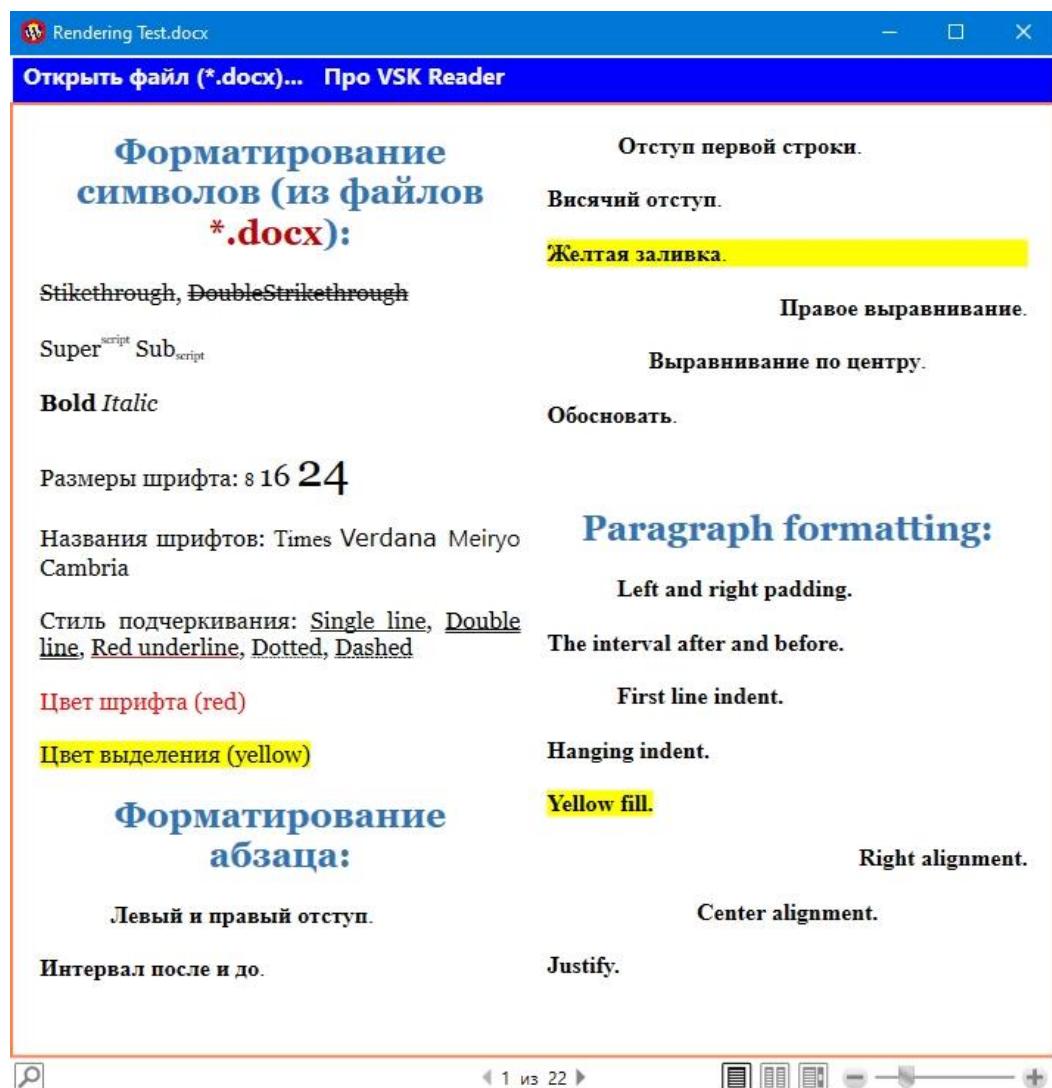


Рисунок №20 – Окно для просмотра простых файлов \*.docx

Элемент меню реализую следующим образом:

```
<MenuItem x:Name="menuOpen" Height="30" Foreground="#0822FF"
Background="#FCF661" Click="openDocxFail_Click" Header="_Open Word / Открыть
*.docx"> <MenuItem.Icon> <Image Margin="1,-10,-5,-10"> <Image.Source>
<BitmapImage UriSource="/Images/Menu/Word.png" />
</Image.Source> </Image> </MenuItem.Icon> </MenuItem>
```

Изм.	Лист	№ докум.	Подпись	Дата

После выбора данного пункта меню, запускается форма для просмотра:

```
public void openDocxFail_Click(object sender, RoutedEventArgs e)
{
    // запускаю форму для просмотра документов *.docx
    VSKReader fvskReader = new VSKReader();
    fvskReader.Show(); // "Rendering Test.docx" – создаю сам в MS Word
} // этот файл сразу открывается из ресурсов проекта (для примера)
```

Сам файл «**VSKReader.xaml.cs**» содержит следующий программный код:

```
public partial class VSKReader : Window
{
    // указываю на созданный файл *.docx
    public string path = "Rendering Test.docx";
    public VSKReader()
    {
        InitializeComponent(); // открываю файл
        this.ReadDocx(path); } // путь к файлу
    public void ReadDocx(string path) // переменная для открытия файла
    {
        using (var stream = File.Open(path, FileMode.Open, FileAccess.Read,
FileShare.ReadWrite))
        { // переменная для загрузки информации из файла (во «FlowDocument»)
            var flowDocumentConverter = new DocxToFlowDocumentConverter(stream);
            // прочтение полученной информации
            flowDocumentConverter.Read();
            this.flowDocumentReader.Document = flowDocumentConverter.Document;
            this.Title = System.IO.Path.GetFileName(path); } }
    public void onOpenFileDialogged(object sender, RoutedEventArgs e)
    {
        // вызываю диалоговое окно
        var openFileDialog = new OpenFileDialog() // для открытия файлов
        { DefaultExt = ".docx", // фильтр по расширению
Filter = "Документы Word (.docx)|*.docx" };
        // если выбран файл с верным расширением – открываю его
        if (openFileDialog.ShowDialog() == true)
```

Изм.	Лист	№ докум.	Подпись	Дата

```

        this.ReadDocx(openFileDialog.FileName);      }

    public void onAboutClicked(object sender, RoutedEventArgs e)
    { // “рисую” краткую справку, и показываю эту справку по продукту
        new AboutWindow().ShowDialog();      } (Рисунок 21)

```



Рисунок №21 – Справка для просмотра простых файлов \*.docx

4. Пункт «**PDF Editor / PDF редактор**» позволяет открывать созданный мной редактор «Работа с файлами PDF». Данное приложение работает с любыми файлами \*.pdf (в том числе и взятыми из встроенной библиотеки).

Для работы данного редактора подключаю библиотеки **Syncfusion**:

```
public partial class WindowBookPDF : Window
```

```
{ // стоимость лицензии Syncfusion на 1ПК - 2495 у.е., на 5 ПК – 6000 у.е. и т.д.
```

Изм.	Лист	№ докум.	Подпись	Дата

**VSK.090204.07.00.000 ПЗ**

Лист

110

```

# region Constructor // доступ к сайту Syncfusion закрыть для жителей РФ
public WindowBookPDF()
{
    // на англоязычных форумах нахожу код активации для "Syncfusion - PDF"
Syncfusion.Licensing.SyncfusionLicenseProvider.RegisterLicense("xxxxx...xxxxx");

    // Syncfusion - компания из США, на которую распространяется
законодательство США. А, следовательно, из-за моего нахождения в России мне
запрещён доступ к каким-либо материалам сайта "syncfusion.com". Но через "Tor
Browser" мне удаётся получить данный продукт. В частности "Syncfusion Essential
Studio for WPF" и оказался доступным другой инфо-ресурс: https://help.syncfusion.
com/ - на английском языке (устанавливаю Syncfusion 2020 года - 6.1 гигабайт)
}

InitializeComponent();

ImageSourceConverter converter = new ImageSourceConverter();
}

# region Helper Methods

// получает полный путь к шаблону PDF или изображению
// <param name="fileName"> Имя файла </param>
// <param name="image"> Истинно, если изображение </param>
// <returns> Путь к файлу </returns>
private string GetFullPath(string fileName, bool image)
{
}

// здесь можно открывать и редактировать PDF книги из программы "Оператор
ЭВМ", а также любые другие имеющиеся у Вас файлы PDF

#if NETCORE
    string fullPath = AppDomain.CurrentDomain.BaseDirectory +
        @"..\..\..\..\..\..\Common\";
#else
    string fullPath = AppDomain.CurrentDomain.BaseDirectory + @"..\..\..\..\..\Common\";
#endif

    string folder = image ? "Images" : "Data";
    // возвращаю полный путь до PDF файла
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        return string.Format(@"{0}\{1}\PDF\{2}", fullPath, folder, fileName);
    }

# endregion

} }

```

При нажатии открывается следующее окно (Рисунок 22), где я открываю pdf-книгу, взятую из ресурсов проекта **AllLoader**:

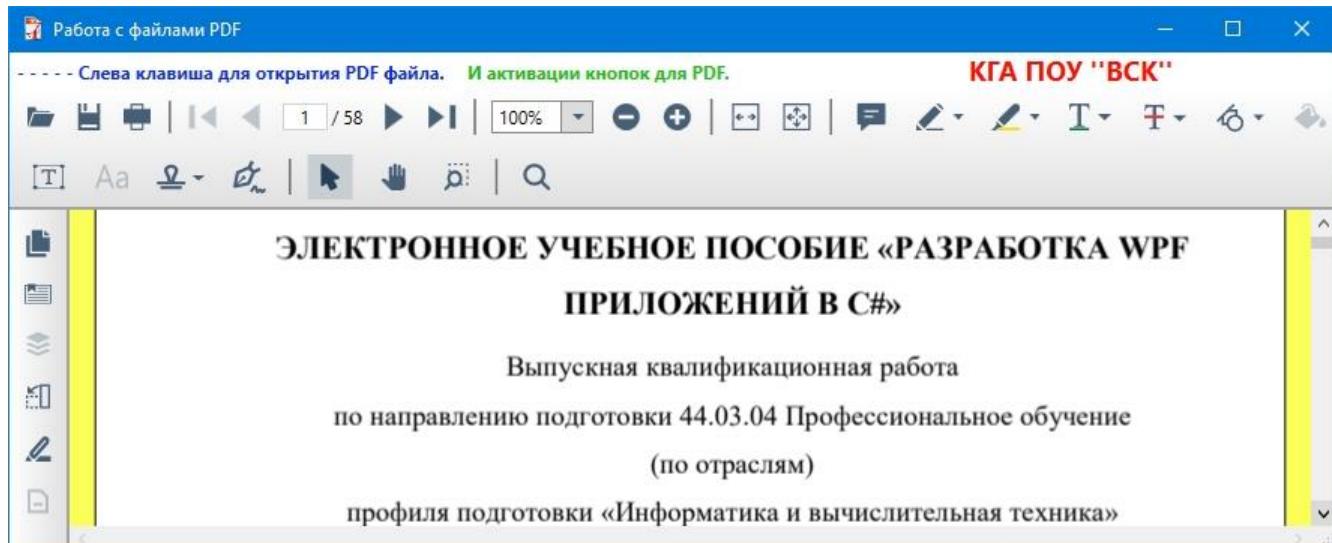


Рисунок №22 – Редактор для PDF файлов

Здесь можно открывать, печатать и редактировать любые **pdf**-источники. Программный код содержит следующее:

```

<Window x:Class="VSKBookingPDF.WindowBookPDF" .....>
    xmlns:cc="clr-namespace:Syncfusion.Windows.PdfViewer;
    assembly=Syncfusion.PdfViewer.WPF"
    xmlns:syncfusion="http://schemas.syncfusion.com/wpf"
    xmlns:local="clr-namespace:VSKBookingPDF"

```

А внизу окна обязательно нужно подключить элемент управления **Syncfusion**, который включает в себя множество инструментов для работы с PDF:

```

<cc:PdfViewerControl BorderBrush="Black"
    x:Name="pdfViewerBooks" AllowDrop="True"

```

Изм.	Лист	№ докум.	Подпись	Дата

*VSK.090204.07.00.000 ПЗ*

Лист

112

Background="#FBFF57" Grid.RowSpan="3" Margin="0,23,0,0">>

</cc:PdfViewerControl>

5. Пункт «**VSK text editor / Текстовый редактор**». В этом редакторе реализую дополнительные возможности по работе с текстом: списки, отступы, выравнивание и т.д. Текст можно сохранить в различных форматах: \*.txt, \*.html, \*.rtf (*количество расширений можно увеличить*). Вот так выглядит сам редактор (Рисунок 23):

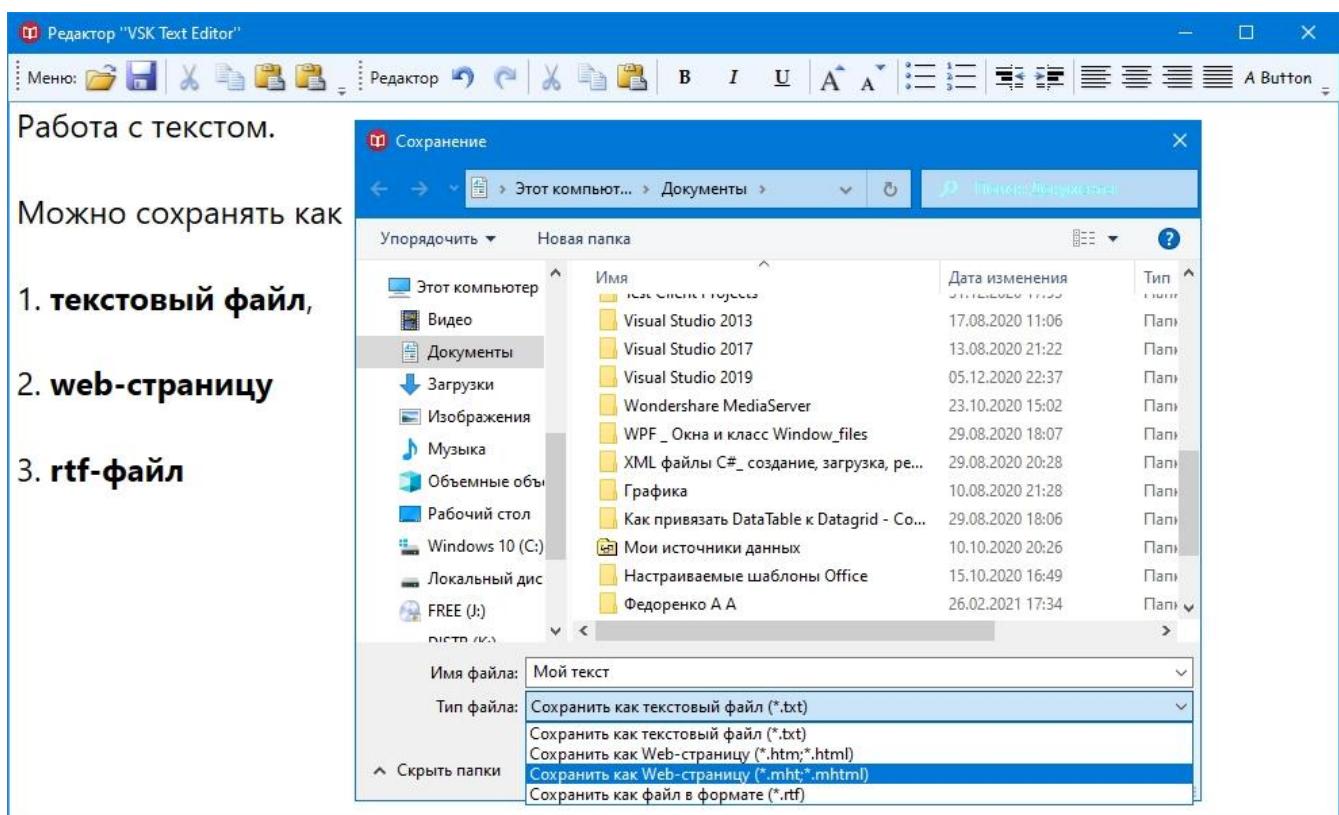


Рисунок №23 – Редактор для файлов: txt, html, rtf

Пункт меню для запуска данного редактора содержит следующий код:

```
<MenuItem <!-- запуск созданного текстового (txt/html/rtf) редактора -->
    x:Name="butOPRTB" Height="30" Foreground="#0822FF"
    Background="#7AFFA2" Click="butOPRTB_Click" Header="_VSK text editor /
    Текстовый редактор ">
    <MenuItem.Icon> <Image Margin="1,-10,-5,-10"> <Image.Source>
        <BitmapImage UriSource="/Images/Menu/optextedit.png" />
```

Изм.	Лист	№ докум.	Подпись	Дата

```
</Image.Source> </Image> </MenuItem.Icon>
</MenuItem>
```

А код, который выполняется при нажатии данного пункта меню:

```
private void butOPRTB_Click(object sender, RoutedEventArgs e)
{
    // создаю и запускаю текстовый редактор (txt, rtf, html)
    VskRichTextEditor fRTBOPLoad = new VskRichTextEditor();
    fRTBOPLoad.Show();      }
```

Для создания данного редактора добавляю в проект **AllLoader** следующее окно **VskRichTextEditor.xaml**:

```
<Window x:Class="AllLoader.VskRichTextEditor"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:AllLoader"
    mc:Ignorable="d" Icon="RichTextEditor.png"
    Title="Редактор "VSK Text Editor"" Height="650" Width="1000">
    <Window.CommandBindings> <!-- системные команды для работы с файлами -->
        <CommandBinding Command="ApplicationCommands.Open"
            Executed="Open_Executed" /> <!-- открыть -->
        <CommandBinding Command="ApplicationCommands.Save"
            Executed="Save_Executed" /> <!-- сохранить -->
    </Window.CommandBindings> <!-- меню и поле редактора -->
    <DockPanel> <!-- оформляю меню для работы с файлами и документами -->
        <ToolBarTray DockPanel.Dock="Top">
            <ToolBarTray.Resources>
                <Style TargetType="Image">
                    <Style.Triggers> <!-- пока нет данных кнопки меню не доступны -->
```

Изм.	Лист	№ докум.	Подпись	Дата

```

<Trigger Property="IsEnabled" Value="False"><!-- уменьшаю видимость -->
    <Setter Property="Opacity" Value="0.5"/>
</Trigger>  </Style.Triggers>
</Style><!-- стиль - общие характеристики для пунктов меню -->
<Style TargetType="TextBlock"> <Setter Property="Width" Value="30"/>
<Setter Property="TextAlignment" Value="Center"/> <Setter Property="FontFamily" Value="Palatino Linotype"/> <Setter Property="FontSize" Value="14"/> <Setter Property="FontWeight" Value="Bold"/>
</Style></ToolBarTray.Resources>
<ToolBar Header="Меню:" ToolTip="Основные команды для работы с файлами"
         x:Name="menuEditorFileRtf">
    <ToolBar.CommandBindings><!-- прописываю команды -->
        <CommandBinding Command="Open" Executed="Open"/>
        <CommandBinding Command="Save" Executed="Save" CanExecute="CanSave" />
        <CommandBinding Command="SaveAs" Executed="Save" CanExecute="CanSave" />
        <CommandBinding Command="Cut" Executed="Cut" CanExecute="CanCut" />
        <CommandBinding Command="Copy" Executed="Copy" CanExecute="CanCopy" />
        <CommandBinding Command="Paste" Executed="Paste" CanExecute="CanPaste"/>
    </ToolBar.CommandBindings>
    <!-- оформляю кнопки - замена (Command="Open", Command="Save") -->
    <Button Click="Open_Executed" ToolTip="Открыть файл"
           x:Name="butRtfFileOpen" >
        <Image Source="Icons/Open.png"/><!-- добавляю картинки -->
    </Button>
    <Button Click="Save_Executed" ToolTip="Сохранить файл"
           x:Name="butRtfFileSave"><Image Source="Icons/Save.png"/></Button>
    <Separator/>
    <Button Command="Cut" ToolTip="Вырезать" x:Name="butRtfTxtCut">
        <Image Source="Icons/Cut.png"/> </Button> <Button Command="Copy"
        ToolTip="Копировать" x:Name="butRtfTxtCopy"><Image Source="Icons/Copy.png"/>

```

```

</Button> <Button Command="Paste" CommandParameter="prepend"
ToolTip="Вставить" x:Name="butRtfTxtPaste"><Image Source="Icons/Paste.png"/>
</Button><Button Command="Paste" CommandParameter="append"
ToolTip="Вставить добавление" x:Name="butRtfTxtPasteAppend">
<Image Source="Icons/Paste.png"/> </Button> </ToolBar> <!-- добавляю картинки -->
<ToolBar Header="Редактор" ToolTip="Команды для работы с текстом"
x:Name="menuEditorTxtRtf"> <Button Command="Undo" ToolTip="Отменить"
x:Name="butRtfUndo"> <Image Source="Icons/Undo.png"/> </Button>
<Button Command="Redo" ToolTip="Повторить" x:Name="butRtfRedo">
<Image Source="Icons/Redo.png"/> </Button> <Separator/>
<Button Command="Cut" ToolTip="Вырезать" x:Name="butRtfCut">
<Image Source="Icons/Cut.png"/> </Button>
<Button Command="Copy" ToolTip="Копировать" x:Name="butRtfCopy">
<Image Source="Icons/Copy.png"/> </Button>
<Button Command="Paste" ToolTip="Вставить" x:Name="butRtfPaste">
<Image Source="Icons/Paste.png"/> </Button> <Separator/>
<Button Command="ToggleBold" ToolTip="Жирный" x:Name="butRtfBold">
<TextBlock Text="B" FontWeight="Bold"/> </Button>
<Button Command="ToggleItalic" ToolTip="Курсив" x:Name="butRtfItalic">
<TextBlock Text="I" FontStyle="Italic"/> </Button>
<Button Command="ToggleUnderline" ToolTip="Подчеркивание"
x:Name="butRtfUnderline"> <TextBlock Text="U" TextDecorations="Underline"/>
</Button> <Separator/> <!-- добавляю картинки – из ресурсов -->
<Button Command="IncreaseFontSize" ToolTip="Увеличить размер шрифта"
x:Name="butRtfIncrease"> <Image Source="Icons/IncreaseFontSize.png"/> </Button>
<Button Command="DecreaseFontSize" ToolTip="Уменьшить размер шрифта"
x:Name="butRtfDecrease"> <Image Source="Icons/DecreaseFontSize.png"/> </Button>
<Separator/>
<Button Command="ToggleBullets" ToolTip="Список" x:Name="butRtfBullets">
<Image Source="Icons/ToggleBullets.png"/> </Button>

```

Изм.	Лист	№ докум.	Подпись	Дата

```

<Button Command="ToggleNumbering" ToolTip="Список с номерами"
x:Name="butRtfNumbers"><Image Source="Icons/ToggleNumbering.png"/></Button>
<Separator/> <Button Command="DecreaseIndentation"
ToolTip="Уменьшить отступ" x:Name="butRtfDecInd">
<Image Source="Icons/DecreaseIndentation.png"/> </Button>
<Button Command="IncreaseIndentation" ToolTip="Увеличить отступ"
x:Name="butRtfIncInd"><Image Source="Icons/IncreaseIndentation.png"/></Button>
<Separator/> <!-- добавляю картинки из ресурсов проекта -->
<Button Command="AlignLeft" ToolTip="Выровнять по левому краю"
x:Name="butRtfAliLeft"><Image Source="Icons/AlignLeft.png"/></Button>
<Button Command="AlignCenter" ToolTip="Выровнять по центру"
x:Name="butRtfAliCenter"><Image Source="Icons/AlignCenter.png"/></Button>
<Button Command="AlignRight" ToolTip="Выровнять по правому краю"
x:Name="butRtfAliRight"><Image Source="Icons/AlignRight.png"/></Button>
<Button Command="AlignJustify" ToolTip="Выровнять по ширине"
x:Name="butRtfAliJustify"><Image Source="Icons/AlignJustify.png"/></Button>
<!-- оставляю для дальнейшего дополнения/разработки -->
<Button FontFamily="Technical Italic, Comic Sans MS, Arial">A Button</Button>
</ToolBar> </ToolBarTray>
<!-- область для ввода и редактирования текста (информации) -->
<RichTextBox x:Name="richTextBox" AcceptsTab="True" FontSize="24"
VerticalScrollBarVisibility="Auto"/> </DockPanel> </Window>

```

Логика взаимодействия в **VskRichTextEditor.xaml.cs** (работа команд при нажатии на определённый клавиши; в код вставляю комментарии):

```

public partial class VskRichTextEditor : Window
{
    // форма для редактирования текстовых документов
    public VskRichTextEditor()
    {
        // в элементе управления RTB
        InitializeComponent();    } // планировал... осуществить работу с Word

```

Изм.	Лист	№ докум.	Подпись	Дата

```

private void Open(object sender, ExecutedRoutedEventArgs e)
{
    // диалоговое окно для выбора файлов
    var dialog = new OpenFileDialog()
    { // устанавливаю фильтр открываемых документов
        AddExtension = true, Filter =
        // возможные фильтры (*.docx;*.docm;*.doc;*.dotx;*.dotm;*.dot;*.htm;*.html;*.rtf;
        *.txt) | *.docx;*.docm; *.dotx; *.dotm;*.doc;*.dot;*.htm;*.html;*.rtf;*.txt|" + ...
        "Web-страницы (*.htm;*.html)|*.htm;*.html|" +
        "Файлы в формате (*.rtf)|*.rtf|" +
        "Обычные файлы (*.txt)|*.txt" };
    if (dialog.ShowDialog() == true)
        using (var stream = new MemoryStream())
        { // данные из файла преобразую в поток "rtf"
            DocumentModel.Load(dialog.FileName).Save(stream, SaveOptions.RtfDefault);
            stream.Position = 0;
            // и полученные данные передаю в RichTextBox на форму
            var textRange = new TextRange(this.richTextBox.Document.ContentStart,
                this.richTextBox.Document.ContentEnd);
            // текст из переменной передаю в поток ".Rtf"
            textRange.Load(stream, DataFormats.Rtf);      }      }
    // чтобы сохранить файл
}

private void Save(object sender, ExecutedRoutedEventArgs e)
{
    // вызываю диалоговое окно для сохранения файла
    var dialog = new SaveFileDialog()
    { // добавляю ещё одно расширение "mhtml"
        AddExtension = true, Filter =
        "Сохранить как Web-страницу (*.htm;*.html)|*.htm;*.html|" +
        "Сохранить как Web-страницу (*.mht;*.mhtml)|*.mht;*.mhtml|" +
        "Сохранить как файл в формате (*.rtf)|*.rtf|" +
        "Сохранить как текстовый файл (*.txt)|*.txt" };
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

if (dialog.ShowDialog(this) == true)
{
    using (var stream = new MemoryStream())
    {
        // завожу переменную чтобы сохранить данные в файл
        var textRange = new TextRange(this.richTextBox.Document.ContentStart,
this.richTextBox.Document.ContentEnd);

        // сохраняю, в потоке RTF, содержимое x:Name="richTextBox"
        textRange.Save(stream, DataFormats.Rtf); stream.Position = 0;
        // преобразование текста в выбранный пользователем формат

        DocumentModel.Load(stream, LoadOptions.RtfDefault).Save(dialog.FileName);
        // пользователь может назначать имя для файла
        Process.Start(dialog.FileName);    }    }

// клавиша для открытия файла
private void Open_Executed(object sender, RoutedEventArgs e)
{
    // диалоговое окно для открытия файлов
    OpenFileDialog dlg = new OpenFileDialog(); // фильтр по расширениям
    dlg.Filter = "Обычные текстовые файлы (*.txt)|*.txt|" +
    "Web-страницы (*.htm;*.html)|*.htm;*.html|" +
    "Файлы в формате (*.rtf)|*.rtf|" +
    "Все файлы (*.*)|*.*";
    // делаю варианты (расширения) для открытия
    if (dlg.ShowDialog() == true)
    {
        // использую класс FileStream - считывание из файла и запись в файл
        FileStream fileStream = new FileStream(dlg.FileName, FileMode.Open);
        // через переменную передаю инфо для размещения в "richTextBox"
        TextRange range = new TextRange.richTextBox.Document.ContentStart,
richTextBox.Document.ContentEnd);

        range.Load(fileStream, DataFormats.Rtf);    }    }

// клавиша для сохранение файла
private void Save_Executed(object sender, RoutedEventArgs e)
{
    // диалоговое окно для сохранения файла
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

SaveFileDialog dlg = new SaveFileDialog() // фильтр
dlg.Filter = "Сохранить как текстовый файл (*.txt)|*.txt|" +
    "Сохранить как Web-страницу (*.htm;*.html)|*.htm;*.html|" +
    "Сохранить как Web-страницу (*.mht;*.mhtml)|*.mht;*.mhtml|" +
    "Сохранить как файл в формате (*.rtf)|*.rtf";
// делаю четыре варианта (расширения файла)
if (dlg.ShowDialog() == true)
{
    // класс FileStream - позволяет работать как с txt-файлами, так и с бинарными
    FileStream fileStream = new FileStream(dlg.FileName, FileMode.Create);
    // через переменную передаю инфо для сохранения в файл
    TextRange range = new TextRange(richTextBox.Document.ContentStart,
                                    richTextBox.Document.ContentEnd);
    range.Save(fileStream, DataFormats.Rtf);      }      }

// функция для "вырезания" текста
private void Cut(object sender, ExecutedRoutedEventArgs e)
{
    // сперва копирую в буфер
    this.Copy(sender, e);
    // а после отчищаю выделенную область от текста
    this.richTextBox.Selection.Text = string.Empty;      }

// функция для копирования текста
private void Copy(object sender, ExecutedRoutedEventArgs e)
{
    // завожу переменную чтобы работать с буфером обмена
    using (var stream = new MemoryStream())
    {
        // выделение фрагмента текста
        this.richTextBox.Selection.Save(stream, DataFormats.Rtf);
        stream.Position = 0;
        // сохраняю выделенный текст в буфер обмена (память)
        DocumentModel.Load(stream, LoadOptions.RtfDefault).Content.SaveToClipboard();    }
}

// функция для вставки текста из буфера обмена
private void Paste(object sender, ExecutedRoutedEventArgs e)

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

120

```

    { // использую ту же переменную
        using (var stream = new MemoryStream())
    {

// завожу переменную для передачи текста и сохранения её в потоке RTF
        var textRange = new TextRange(this.richTextBox.Document.ContentStart,
            this.richTextBox.Document.ContentEnd);
        textRange.Save(stream, DataFormats.Rtf);
        stream.Position = 0;

// беру документ из потока RTF и добавляю к нему содержимое буфера обмена
        var document = DocumentModel.Load(stream, LoadOptions.RtfDefault);
        var position = (string)e.Parameter == "prepend" ? document.Content.Start :
document.Content.End;
        position.LoadFromClipboard();           stream.Position = 0;

// сохраняю документ в потоке RTF
        document.Save(stream, SaveOptions.RtfDefault);           stream.Position = 0;
// и загружаю его в RichTextBox на форме
        textRange.Load(stream, DataFormats.Rtf);      }      }

private void CanSave(object sender, CanExecuteRoutedEventArgs e)
{
// проверка на возможность сохранения
    if (this.richTextBox != null)
    {

// если текст есть - завожу переменную для всего текста
        var document = this.richTextBox.Document;
        // переменную для начала контекста
        var startPosition =
            document.ContentStart.GetNextInsertionPosition(LogicalDirection.Forward);
        // и переменную для конца контекста
        var endPosition =
            document.ContentEnd.GetNextInsertionPosition(LogicalDirection.Backward);
}

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

121

```

// если такие переменные возможны и имеют разное значение
e.CanExecute = startPosition != null && endPosition != null &&
startPosition.CompareTo(endPosition) < 0;

} // вызываю диалог сохранения файла
else e.CanExecute = false; // если всё пусто - ничего не делается      }

private void CanCut(object sender, CanExecuteRoutedEventArgs e)
{
// проверка на возможность вырезать текст
e.CanExecute = this.richTextBox != null && !this.richTextBox.Selection.IsEmpty;
}

// выделенная область не пуста
private void CanCopy(object sender, CanExecuteRoutedEventArgs e)
{
// проверка на возможность скопировать текст
e.CanExecute = this.richTextBox != null && !this.richTextBox.Selection.IsEmpty;
}

// выделенная область не пуста
private void CanPaste(object sender, CanExecuteRoutedEventArgs e)
{
// проверка на возможность вставить текст из буфера
e.CanExecute = this.richTextBox != null && this.richTextBox.IsKeyboardFocused;
}

// если не пустой и фокус на "текст-боксе"    } }

```

Получился более продвинутый редактор для текстовых файлов, если сравнивать с “первой версией” (чем в – ”Пункт «**New / Новый \*txt**”, страница 106).

6. Пункт «Simple calculator / Простой калькулятор». В конструкторе размещаю необходимые кнопки для калькулятора (Рисунок 24):

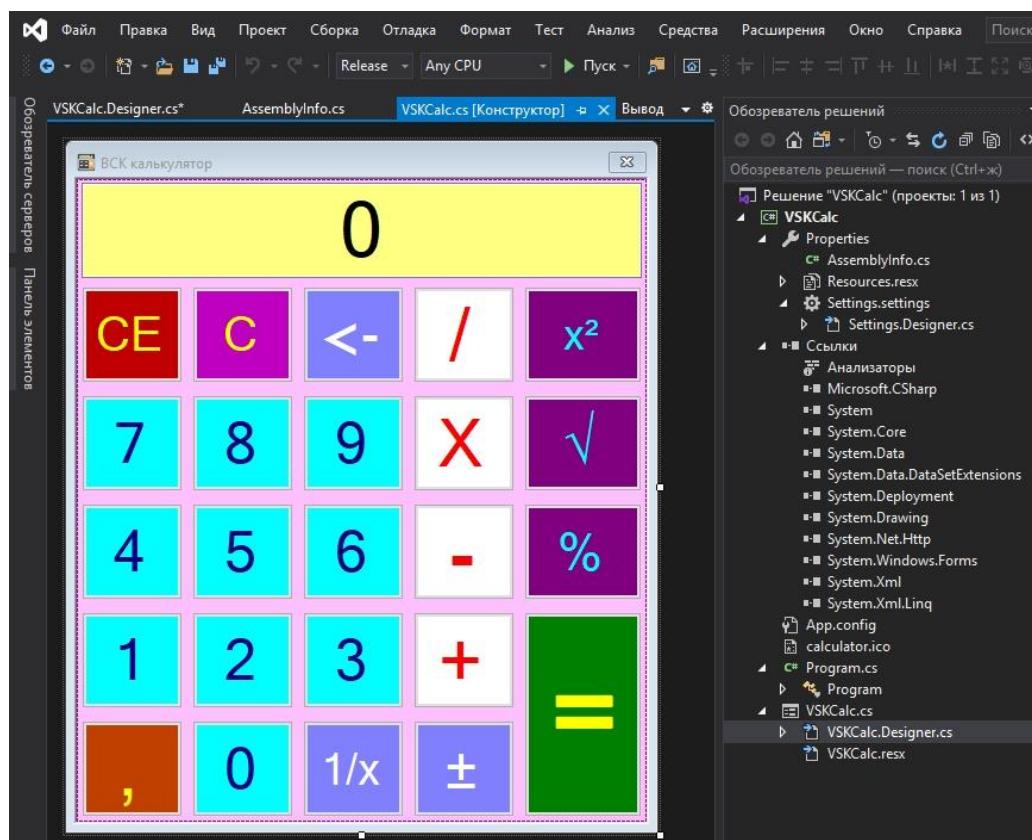


Рисунок №24 – Конструктор простого калькулятора

Пишу программный код для работы калькулятора:

namespace VSKCalc

```
{
    public partial class VSKCalc : Form
    {
        // задаю переменные для мат.операций (D – знак (+/-), 1 и 2 - переменные)
        public string D;      public string N1;      public bool n2;
        public VSKCalc()
        {
            n2 = false; // при запуске ничего не введено (ноль)
            InitializeComponent();
        }
        private void button23_Click(object sender, EventArgs e) // просто меняю знак -+
        {
            double dn, res; // конвертирую введенное значение - Convert.ToDouble
            dn = Convert.ToDouble(textBox1.Text);
            res = -dn; // промежуточная переменная для нового результата
            textBox1.Text = res.ToString(); // обратно в строку
        }
}
```

Изм.	Лист	№ докум.	Подпись	Дата

```

private void button22_Click(object sender, EventArgs e) // клавиша «0»
{
    // обработка чтобы нельзя было ввести много «0» подряд
    if (n2)      {      n2 = false;      textBox1.Text = "0";      }
}

Button B = (Button)sender; // тип параметра sender, является Object
// здесь, для доступа к свойству Text, я тип Object привожу к типу Button.
if(textBox1.Text=="0") textBox1.Text = B.Text; // если есть только ноль, то
ничего не добавляю – остаётся ноль. А если есть другие цифры(,) добавляю «0»
else textBox1.Text = textBox1.Text + B.Text;      }

private void button6_Click(object sender, EventArgs e) // сбрасываю значение на «0»
{
    textBox1.Text = "0";      }

private void button20_Click(object sender, EventArgs e) // для сложения - беру число
{
    Button B = (Button)sender;      D = B.Text;      N1 = textBox1.Text;
    n2 = true;      } // которое уже было введено

private void button24_Click(object sender, EventArgs e) // кнопка равно
{
    double dn1, dn2, res;      res = 0;      dn1 = Convert.ToDouble(N1);
    dn2 = Convert.ToDouble(textBox1.Text);      if (D == "+")
    { // два числа привожу к Double
        res = dn1 + dn2;      } // складываю
        if (D == "-")      {      res = dn1 - dn2; // отнимаю      }
        if (D == "X")      {      res = dn1 * dn2; // умножаю      }
        if (D == "/")      {      res = dn1 / dn2; // делю      }
        if (D == "%")      {      res = dn1 * dn2 / 100; // проценты      }
    D = "="; n2 = true; textBox1.Text = res.ToString(); // вывожу значение в поле      }

private void button2_Click(object sender, EventArgs e) // извлекаю корень
{
    double dn,res;      dn = Convert.ToDouble(textBox1.Text);
    res = Math.Sqrt(dn);      textBox1.Text = res.ToString();      }

private void button3_Click(object sender, EventArgs e) // возвожу в квадрат
{
    double dn, res;      dn = Convert.ToDouble(textBox1.Text);
    res = Math.Pow(dn,2);      textBox1.Text = res.ToString();      }

private void button4_Click(object sender, EventArgs e) // делаю единицу на число

```

Изм.	Лист	№ докум.	Подпись	Дата

```

{      double dn, res;      dn = Convert.ToDouble(textBox1.Text);
          res = 1/dn;      textBox1.Text = res.ToString();      }

private void button21_Click(object sender, EventArgs e) // проверяю – ставлю запятую
{
    if(!textBox1.Text.Contains(","))      textBox1.Text = textBox1.Text + ",";
}

private void button7_Click(object sender, EventArgs e) // убираю цифры справа
{
    textBox1.Text=textBox1.Text.Substring(0, textBox1.Text.Length - 1);
    if (textBox1.Text == "")      textBox1.Text = "0";  }  }

```

Данный код компилирую отдельно (*Widows.Forms*). Для запуска данной программы в **AllLoader** использую следующий программный код:

```

private void butOPSimpleCalc_Click(object sender, RoutedEventArgs e)
{
    // запускаю созданный калькулятор
    ProcessStartInfo simpleOPcalculator = new ProcessStartInfo();
    // указываю на созданный exe файл (путь - имя файла)
    // чтобы не использовать двойных слэшей '\\' ставлю "@"
    simpleOPcalculator.FileName = @"C:\OperatorIBM\VSKCalc\VSKCalc.exe";
    // после запускаю оболочку процесса без создания самого окна
    simpleOPcalculator.CreateNoWindow = true;
    // полученный процесс запускаю на исполнение
    simpleOPcalculator.UseShellExecute = true;
    // создаю объект класса 'Process' для запуска "VSKCalc.exe"
    Process simpleOPCalcProcess = Process.Start(simpleOPcalculator);
    // в конце инициализирую запуск данного файла      }

```

7. Пункт **«Big clock / Большие часы»**. Переделываю созданные для главного окна маленькие часы. Использую для стрелок не скаченные рисунки, а сам рисую стрелки через “координаты”. Увеличиваю и немного изменяю рисунок циферблата в Photoshop 2020 (Рисунок 25).

<Grid>

<Border BorderThickness="10" BorderBrush="Black" CornerRadius="300"

Изм.	Лист	№ докум.	Подпись	Дата

*VSK.090204.07.00.000 ПЗ*

Лист

125

```

Width="520" Height="520"> <!-- размер всего окна -->
<Grid Height="500" Width="500" HorizontalAlignment="Center"
      <!-- положение окна --> VerticalAlignment="Center">

<Border CornerRadius="350">
    <Border.Background>
        <!-- циферблат в ресурсах -->
        <ImageBrush ImageSource="Assets/clock.png"/>
    </Border.Background> </Border>

<Border CornerRadius="350" Background="#778889BB"/>
// стрелка для часов

<Border x:Name="hour" CornerRadius="0 15 15 0" Height="10" Width="130"
        BorderThickness="3" BorderBrush="#FF000BFC" Margin="0,245,
119.958,245" HorizontalAlignment="Right" RenderTransformOrigin="0,0.5">
    <Border.Effect> <!-- и от стрелки тень -->
    <DropShadowEffect BlurRadius="9" ShadowDepth="1"/>
    </Border.Effect>
    <Border.RenderTransform>
        <TransformGroup>
            <ScaleTransform/> <SkewTransform/>
            <RotateTransform Angle="-90"/>
            <TranslateTransform/>
        </TransformGroup>
    </Border.RenderTransform>
</Border>

<Border x:Name="minute" CornerRadius="0 15 15 0" Height="5"
        Width="160" // стрелка минутная
        BorderThickness="2" BorderBrush="#FF1AD109" Margin="0,247,90,
247.514" HorizontalAlignment="Right" RenderTransformOrigin="0, 0.5"
        Background="{x:Null}">
    <Border.Effect> // тень

```

```

        <DropShadowEffect BlurRadius="9" ShadowDepth="1"/>
    </Border.Effect>
    <Border.RenderTransform>
        <TransformGroup> <ScaleTransform/> <SkewTransform/>
        <RotateTransform Angle="-90"/>
        <TranslateTransform/>
    </TransformGroup>
    </Border.RenderTransform> </Border>
<Border x:Name="second" CornerRadius="0 15 15 0" Height="3"
Width="220" Background="Red" Margin="249.979,248.5,30.02,248.5"
RenderTransformOrigin="0, 0.5"> // стрелка секундная
    <Border.Effect> // тень
        <DropShadowEffect BlurRadius="9" ShadowDepth="1"/>
    </Border.Effect>
    <Border.RenderTransform>
        <TransformGroup> <ScaleTransform/> <SkewTransform/>
        <RotateTransform Angle="-90"/>
        <TranslateTransform/> </TransformGroup>
    </Border.RenderTransform>
</Border>
<Ellipse Fill="#0025D9" Width="20" Height="20" // по центру
    HorizontalAlignment="Center" VerticalAlignment="Center">
    <Ellipse.Effect> // тень
        <DropShadowEffect BlurRadius="5" ShadowDepth="1"/>
    </Ellipse.Effect>
</Ellipse> </Grid>
</Border> </Grid>
</Window>

```

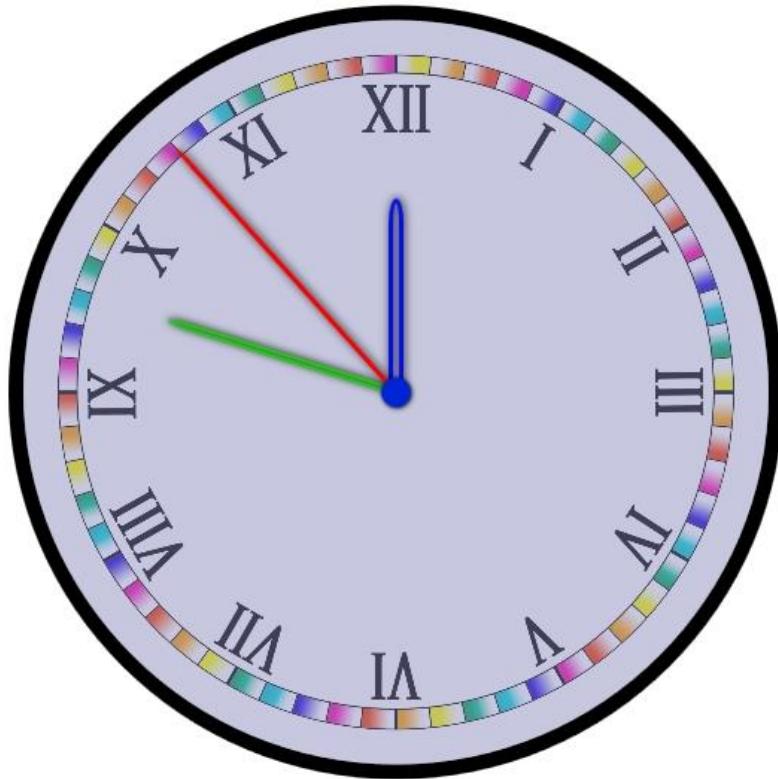


Рисунок №25 – Большие часы по центру экрана

Чтобы убрать часы с экрана нужно нажать комбинацию клавиш «**Alt+F4**».

## 8. Пункт **«HTML Binding / HTML практика»**.

Нахожу на сайте <https://github.com/> библиотеки для работы в html-кодом, с комментариями на английском языке: [HtmlCssParser.cs](#), [HtmlFromXamlConverter.cs](#), [HtmlLexicalAnalyzer.cs](#), [HtmlParser.cs](#), [HtmlSchema.cs](#), [HtmlTokenType.cs](#), [HtmlToXamlConverter.cs](#). Добавляю в проект **VSKBinding** папку «**HtmlConverter**», а в ней размещаю полученные библиотеки (Рисунок 26). В окне **Window** размещаю элемент управления **TabControl**, где устанавливаю 7 вкладок. На первых шести вкладках, через ‘**Bindind**’ связываю три элемента управления: 1. **TextBox** 2. **RichTextBox** 3. **WebBrowser**. Здесь можно вносить исправления в html-код и сразу видеть какие наступят изменения в WebBrowser-е ниже. Для этого привязываю строки, содержащие HTML в TextBox-е, к содержимому соответствующего WebBrowser-а на вкладке:

```

    HtmlRichTextBoxBehavior.cs      MainWindow.xaml*  X
    <TextBlock x:Name="Instruction
        Foreground="#16E902" Fon
<!-- в TextBox нужно вводить html-коды -->
    <TextBox x:Name="TextBox0" Grid.Row="1" Fo
        Margin="10,0" Height="200" Background
        HorizontalAlignment="Stretch" Vertica
        Text="&lt;p&ampgtEnter your HTML code h
    1. &lt;a href="http://www.micros
    2. &lt;a href="https://www.vsteh
    3. &lt;a href="https://www.vl.ru
    4. &lt;a href="https://www.farpo
    5. &lt;a href="https://www.drom.
    6. &lt;a href="https://www.googl
    7. &lt;a href="https://livebook.
    8. &lt;a href="https://ok.ru/&qu
    9. &lt;a href="https://www.faceb
    10. &lt;a href="https://www.aliex
        &lt;/p&ampgt&lt;p&ampgt "
        TextWrapping="Wrap" />
    <RichTextBox Grid.Row="2" Height="200" Bac
        local:HtmlRichTextBoxBehavior.Tex
        IsDocumentEnabled="True" Vertical
        IsReadOnly="True" />
<!-- редактировать RTF запрещаю -->
<!-- через биндинг привязка к тексту из TextBox на
    <Border BorderBrush="Red" BorderThickness="4" Grid.Row="3" Grid.RowSpan="4"
        local:WebBrowserBehavior.B
    </Border>
    </Grid>
    </StackPanel>
</TabItem.Content>
</TabItem>

```

Рисунок №26 – Библиотеки для работы с html

На седьмой вкладке размещаю **WebBrowser** (**x:Name="WebforHTML6"**), который благодаря свойству «**Margin**» (**Margin="0, -400"**), закрывает собой и ‘*TextBox*’, и ‘*RichTextBox*’ (находящиеся на 1 и 2 строке вкладки). При нажатии на гиперссылку происходит открытия соответствующего сайта.

```

<Border BorderBrush="Red" BorderThickness="4" Grid.Row="3" Grid.RowSpan="4"
    Margin="0, -400" ><WebBrowser Grid.Row="3" Grid.RowSpan="4"
x:Name="WebforHTML6" Height="auto" local:WebBrowserBehavior.Body="{ Binding
    ElementName=TextBox6, Path=Text}" /> </Border>

```

На первых шести вкладках пишу лёгкие задания (*no Html*). Выполнение которых позволяет увидеть, как изменяется вид страницы, при внесении исправлений в html-код. Ниже привожу программный код для первой вкладки:

```
<TabItem x:Name="oneTabHtml"> // первая вкладка
    <TabItem.Header> <StackPanel Orientation="Horizontal">
        <Ellipse Height="10" Width="10" Fill="#2BC900" />
        <TextBlock x:Name="brText" Margin="3" Background="#10FAFA"
            Foreground="Red" FontSize="20"></TextBlock>
    </StackPanel> </TabItem.Header>
    <TabItem.Content> <StackPanel> <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="25"/> // заголовок – тема – задание
            <RowDefinition Height="200"/> // html код – можно писать
            <RowDefinition Height="200" /> // richtextbox – нельзя изменять
            <RowDefinition MinHeight="500"/> // webBrowser – итог
        </Grid.RowDefinitions>
        <TextBlock x:Name="InstructionInput0"
            Foreground="#16E902" FontSize="16" HorizontalAlignment="Center"
            FontWeight="Bold" /> <!-- в TextBox нужно вводить html-коды -->
        <TextBox x:Name="TextBox0" Grid.Row="1" FontSize="20"
            Foreground="Blue" Margin="10,0" Height="200" Background="LightYellow"
            HorizontalAlignment="Stretch" VerticalAlignment="Top" Text="&lt;p&gt;
Enter your HTML code here. Through the Binding, the html code entered will be
displayed below. First in the element: `` RichTextBox ''. And at the very bottom of the
window, in the `` WebBrowser '' element (these are the simplest controls in C #).
        1. &lt;a href="http://www.microsoft.com" &gt; _Open site Microsoft >>>_&lt;/a&gt;
        2. &lt;a href="https://www.vstehn.ru/" &gt;&lt;b&gt; _Open site "VSK" &lt;/b&gt; >>>_&lt;/a&gt; // ссылка на сайт «ВСК»
```

3. &lt;a href="https://www.vl.ru/">&lt;i&gt; \_Open site Vladivostok</i> >>>\_&lt;/a&gt;

4. &lt;a href="https://www.farpost.ru/vladivostok/"> \_Open site "Farpost" >>>\_&lt;/a&gt;

5. &lt;a href="https://www.drom.ru/"> \_Car sales in Russia (DROM) >>>\_&lt;/a&gt;

6. &lt;a href="https://www.google.com/"> \_Open site Google >>>\_&lt;/a&gt;

7. &lt;a href="https://livebook.manning.com/book/wpf-in-action-with-visual-studio-2008/table-of-contents/">&lt;b&gt; \_"WPF in Action with Visual Studio 2008" &lt;/b&gt; >>>\_&lt;/a&gt;

8. &lt;a href="https://ok.ru/">&lt;i&gt; \_Open site "Odnoklassniki"</i> >>>\_&lt;/a&gt;

9. &lt;a href="https://www.facebook.com/"> \_Open site "Facebook.com" >>>\_&lt;/a&gt;

10. &lt;a href="https://www.aliexpress.com/"> \_Aliexpress.com >>>\_&lt;/a&gt;

```

    &lt;/p&gt;&lt;p&gt; " TextWrapping="Wrap" />
    <RichTextBox Grid.Row="2" Height="200" Background="#E4FA6A"
    Foreground="DarkMagenta" local:HtmlRichTextBoxBehavior.Text="{ Binding
    ElementName = TextBox0, Path=Text}" IsDocumentEnabled="True" VerticalAlignment
    = "Top" FontSize="16" Margin="10,0" IsReadOnly="True" />
    <!-- редактировать RTF запрещаю -->
    <!-- через 'биндинг' привязка к тексту из 'TextBox' наверху -->
    <Border BorderBrush="Red" BorderThickness="4" Grid.Row="3"
    Grid.RowSpan="4"> <WebBrowser Grid.Row="3" Grid.RowSpan="4"
    x:Name="WebforHTML" Height="auto" local:WebBrowserBehavior.Body= "{Binding
    ElementName=TextBox0, Path=Text}" /> </Border> </Grid> </StackPanel>
    </TabItem.Content> </TabItem>
  
```

Работу данного кода можно видеть ниже (Рисунок 27).

Изм.	Лист	№ докум.	Подпись	Дата	Лист
					ВСК.090204.07.00.000 ПЗ 131

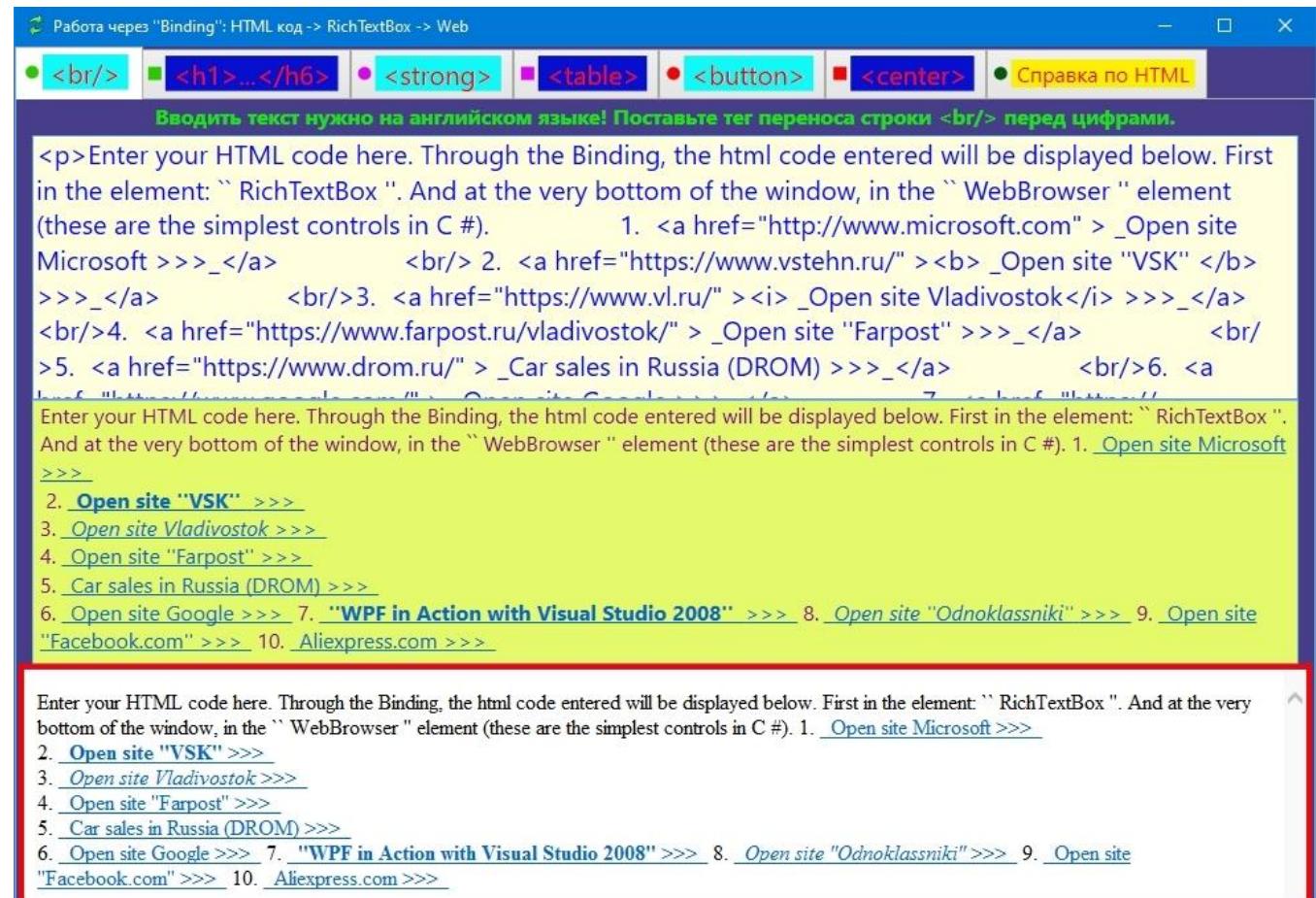


Рисунок № 27 – Первая вкладка для HTML. Начало работы через “Binding”

### 3D плеер для аудио и видео файлов

На сайте <https://metanit.com/> знакомлюсь с основами работы с трехмерной графикой. И создаю 3D плеер для проигрывания следующих аудио и видео файлов:

```
private void ChooseFile_Click(object sender, RoutedEventArgs e)
{
    // диалог для выбора типа медиа файлов – кнопка «Выбрать»
    OpenFileDialog open = new OpenFileDialog();
    open.Multiselect = false; // точно доступными делаю следующие файлы
    open.Filter = "Аудио файлы (*.wma;*.mp3;*.wav;*.ogg)|*.wma;*.mp3;*.wav; *.ogg" +
        "|Видео файлы (*.wmv;*.mpg;*.avi)|*.wmv;*.mpg;*.avi" +
        "|Все файлы (*.*)|*.*";
    if (open.ShowDialog() == true) // если файл успешно выбран
    {
        // выбранный файл становится источником для плеера (контентом)
        mediaPlayer.Source = new Uri(open.FileName, UriKind.Absolute);
```

Изм.	Лист	№ докум.	Подпись	Дата	Лист
					132

ВСК.090204.07.00.000 ПЗ

```

System.IO.FileInfo f = new System.IO.FileInfo(open.FileName);
nowPlaying.Content=f.Name.Substring(0,(f.Name.Length-f.Extension.Length)); } }

```

Работать с графикой в принципе очень интересно, но работать с трехмерной — особенно интересно. И WPF предлагает удобный инструментарий для этого. Конечно, создавать суперсложные трехмерные сцены или игры на WPF — очень непростой процесс, и лучше для этого выбрать **DirectX**, **OpenGL**, либо специально заточенные под это фреймворки и движки на подобие **Unity**, **Monogame** и т.д. Однако для относительно несложных приложений трехмерные возможности WPF вполне подходят.

Для создания трехмерной сцены в приложении WPF требуется несколько компонентов:

- Окно просмотра (**Viewport3D**), которое и содержит трехмерную сцену.
- Сам объект или геометрия.
- Камера, которая устанавливает, как сцена или объект будет отображаться.
- Освещение, которое и содержит трехмерную сцену.
- Материал, который вместе с освещением определяет внешний вид трехмерного объекта.

Итак, контейнером верхнего уровня для трехмерной сцены является объект **Viewport3D**. Формально это такой же объект со всеми свойствами, как кнопка или текстовое поле, который я могу позиционировать на форме, как угодно.

Для использования данного 3D-плеера мне хочется создать и установить права доступа для следующих пользователей. Создаю класс **User.cs**:

```

namespace forVsk3DPlayer // для плеера
{
    // класс для тех у кого есть права на использование плеера
    class User
    {
        // первый порядковый номер - 1000
        private static int id=1000;
        // для прибавления к номеру через 'id'
        private int uid = 0;           private string name;

```

Изм.	Лист	№ докум.	Подпись	Дата

```

private string user;           private string pass;           public int Id
{
    // получаю и устанавливаю вводимую информацию
    get { return uid; }          set { uid = value; }
    public string Username // - Логин
    {
        // логин - это Фамилия (по-русски с большой буквы)
        get { return user; }          set { user = value; }
        public string Pass // - Пароль
        {
            // пароль - это Имя
            get { return pass; }          set { pass = value; }
        }
    public User()
    {
        // сперва все поля для входа пустые
        user = name = pass = "";      id++;      uid = id;
    }
public User(string n,string u,string p)
{
    // три переменных на каждого пользователя
    user = u; name = n; pass = p; // и порядковый номер      id++; uid = id;
}
public string Name
{
    // это ФИО для списка пользователей
    get { return name; }          set { name = value; }
}

```

Вход и использование проигрывателя без логина и пароля невозможен. Для доступа создаю список зарегистрированных пользователей:

```

public partial class Vsk3DPlay : Window
{
    // создаю список зарегистрированных пользователей
    private ArrayList userList = new ArrayList();
    System.Windows.Threading.DispatcherTimer
    clock = new System.Windows.Threading.DispatcherTimer();
    System.Windows.Threading.DispatcherTimer
    mediaClock = new System.Windows.Threading.DispatcherTimer();
    // без прав - вход невозможен
    private bool login = false;

```

```

public Vsk3DPlay()
{
    // заношу в программу Пользователей: ФИО / Логин / пароль
    userList.Add(new User("Федоренко Л.А.", "Федоренко", "Леонид"));
    // Логин — это Фамилия, Пароль — это Имя - на руском языке
    // а ФИО, исключительно для списка, у кого есть права
    userList.Add(new User("Толок И.Е.", "Толок", "Ирина"));
    // это доступ вообще для всех учителей
    userList.Add(new User("Лысенко И.А.", "Лысенко", "Иван"));
    userList.Add(new User("Овчинникова Л.А.", "Овчинникова", "Лариса"));
    userList.Add(new User("Андреев В.В.", "Андреев", "Вячеслав"));
    userList.Add(new User("Антохина С.П.", "Антохина", "Светлана"));
    userList.Add(new User("Барвинок Л.С.", "Барвинок", "Людмила"));
    userList.Add(new User("Буртасов А.И.", "Буртасов", "Александр"));
    userList.Add(new User("Бабенко Е.Н.", "Бабенко", "Евгений"));
    userList.Add(new User("Байкин С.А.", "Байкин", "Сергей"));
    userList.Add(new User("Герман Е.Д.", "Герман", "Елена"));
    userList.Add(new User("Головин Д.И.", "Головин", "Дмитрий"));
    userList.Add(new User("Гулевич Л.С.", "Гулевич", "Лариса"));
    userList.Add(new User("Даниленко Ю.В.", "Даниленко", "Юлия"));
    userList.Add(new User("Данилова Л.Н.", "Данилова", "Людмила"));
    userList.Add(new User("Кононова О.В.", "Кононова", "Ольга"));
    userList.Add(new User("Котенко Ю.С.", "Котенко", "Юлия"));
    userList.Add(new User("Крысанова Н.А.", "Крысанова", "Надежда"));
    userList.Add(new User("Красикова Е.В.", "Красикова", "Елена"));
    userList.Add(new User("Крюков А.А.", "Крюков", "Алексей"));
    userList.Add(new User("Лебедев И.В.", "Лебедев", "Игорь"));
    userList.Add(new User("Литошенко Д.А.", "Литошенко", "Денис"));
    userList.Add(new User("Лобова Т.Ж.", "Лобова", "Татьяна"));
    userList.Add(new User("Лукина К.В.", "Лукина", "Кира"));
    userList.Add(new User("Лысенко И.А.", "Лысенко", "Иван"));
    userList.Add(new User("Миргееев А.А.", "Миргееев", "Андрей"));
    userList.Add(new User("Миллер Н.В.", "Миллер", "Наталья"));
    userList.Add(new User("Матвиенко В.А.", "Матвиенко", "Владимир"));
    userList.Add(new User("Николаев С.О.", "Николаев", "Сергей"));
    userList.Add(new User("Паскарь И.М.", "Паскарь", "Ирина"));
    userList.Add(new User("Пивоварцева И.В.", "Пивоварцева", "Ирина"));
    userList.Add(new User("Пошивайло В.С.", "Пошивайло", "Валентина"));
    userList.Add(new User("Рубан Н.И.", "Рубан", "Нина"));
    userList.Add(new User("Родионов Р.А.", "Родионов", "Руслан"));
    userList.Add(new User("Сновидов Е.Б.", "Сновидов", "Евгений"));
    userList.Add(new User("Степанова И.Т.", "Степанова", "Ирина"));
    userList.Add(new User("Сergeева Л.Н.", "Сergeева", "Людмила"));
    userList.Add(new User("Стрембицкая Г.С.", "Стрембицкая", "Галина"));
    userList.Add(new User("Сизова Н.В.", "Сизова", "Наталья"));
    userList.Add(new User("Титко А.Е.", "Титко", "Александр"));
}

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

135

```

userList.Add(new User("Ускова А.Н.", "Ускова", "Антонина"));
userList.Add(new User("Фалеева В.Н.", "Фалеева", "Валентина"));
userList.Add(new User("Фенюков В.В.", "Фенюков", "Валерий"));
userList.Add(new User("Шмонин И.В.", "Шмонин", "Игорь"));
userList.Add(new User("Яровая О.В.", "Яровая", "Ольга"));

// доступ для всех студентов – просто логин - "student" и пароль - "student"
userList.Add(new User("Student", "student", "student"));
InitializeComponent();

// через биндинг показываю пользователей на одной стороне куба
Binding bind = new Binding();
bind.Source = userList; // обращаюсь к созданному списку
listview.SetBinding(ListView.ItemsSourceProperty, bind); // вывожу список

// запуск основной стороны (куба) проигрывателя
this.Loaded += new RoutedEventHandler(Vsk3DPlay_Loaded);
// RoutedEventArgs() - инициализирует новый экземпляр класса RoutedEventArgs
this.MouseWheel += new MouseWheelEventHandler(Global_MouseWheel);
// MouseWheelEventHandler() делегат используется событиями в WPF
}

void Vsk3DPlay_Loaded(object sender, RoutedEventArgs e)
{
    // при вращении куба
    clock.Tick += new EventHandler(clock_Tick);
    clock.Interval = new TimeSpan(0,0,0,0,10);

    // время - для воспроизведения медиа
    mediaClock.Tick += new EventHandler(mediaClock_Tick);
    mediaClock.Interval = new TimeSpan(0,0,0,0,100);

    // для громкости и поиска
    VolSlider.ToolTip = "Громкость";
    PositionSlider.ToolTip = "Искать";
}

void mediaClock_Tick(object sender, EventArgs e)
{
    // время проигрыша в миллисекундах
    PositionSlider.Value+=100;
    mediaPlayer.ToolTip = TimeSpan.FromMilliseconds(PositionSlider.Value);
}

void Global_MouseWheel(object sender, MouseWheelEventArgs e)
{
    if (login)
    {
        // если был вход - колёсиком мыши можно увеличить экран
        double zoom = (e.Delta / 120) * 5;
        if (camera.FieldOfView + zoom >= 29 && camera.FieldOfView + zoom <= 59)
            camera.FieldOfView += zoom;
    }
}

```

Основные кнопки проигрывателя: «[Играть](#)», «[Стоп](#)», «[Пауза](#)» («[Выбрать](#)»):

```

private void play_Click(object sender, RoutedEventArgs e)
{
    // чтобы воспроизвести медиа или остановить
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

if (mediaPlayer.Source!=null) mediaClock.Start();
if(mediaPlayer.Source!=null) mediaPlayer.Play();      }

private void stop_Click(object sender, RoutedEventArgs e)
{
    // чтобы остановить проигрывание
    mediaPlayer.Stop();      mediaClock.Stop();      PositionSlider.Value = 0; }

private void pause_Click(object sender, RoutedEventArgs e)
{
    // чтобы поставить на паузу
    mediaPlayer.Pause();      mediaClock.Stop(); }
} (Рисунок 28)

```

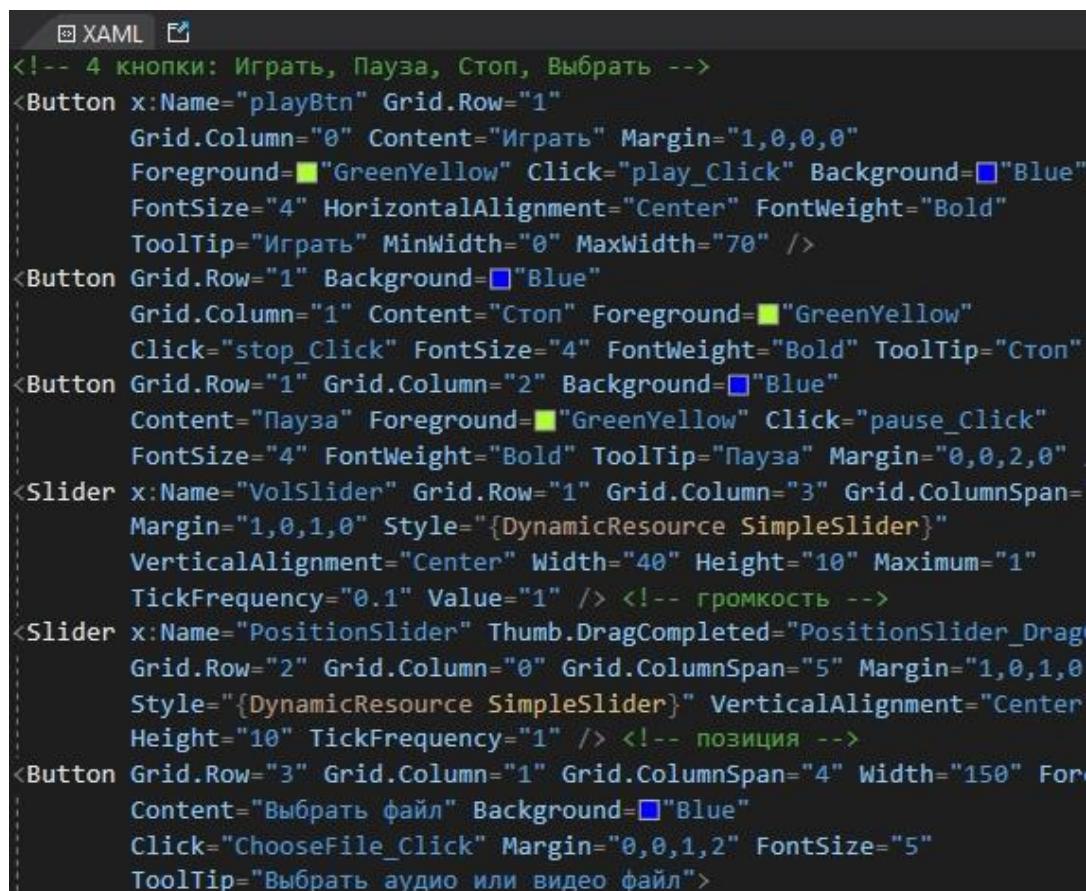


Рисунок № 28 – Кнопки проигрывателя

В файле **forVsk3DPlayerStyles.xaml** определяю набор стилей элементов управления, которые являются упрощенными отправными точками для создания Пользователем собственных элементов управления. Указываю "Кисти": используемые для определения цвета фона, переднего плана, выделения, включения всех элементов управления. Если нужно изменить цвет элемента управления, можно просто поменять кисть.

## Административный доступ к ПО

С сайта <http://www.dotnetmastermind.de/> скачиваю и добавляю ссылку на библиотеку «**WPFSmartLibraryLight35.dll**» в свой проект (Рисунок 29).

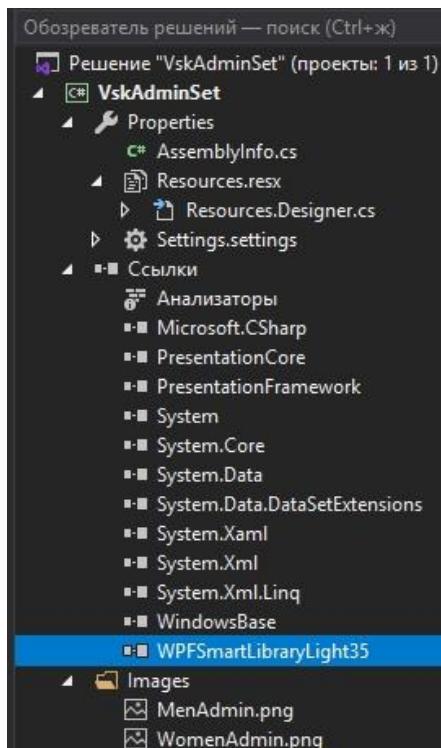


Рисунок № 29 – Добавляю ссылку на «WPFSmartLibraryLight35.dll»

Использую словари ресурсов для загрузки окна из данного файла:

<Window.Resources>

    <ResourceDictionary> <!-- объявляю словари ресурсов -->

        <ResourceDictionary.MergedDictionaries>

            <ResourceDictionary Source="/WPFSmartLibraryLight35;  
            component/ResourceDictionaries/CommonRD.xaml" />

            <ResourceDictionary Source="../ResourceDictionaries/  
            LoginDialogRD.xaml" />   </ResourceDictionary.MergedDictionaries>

    </ResourceDictionary>

</Window.Resources>

<!-- макет будущего окна / разбиваю окно на 15 зон -->

    <Grid x:Name="LayoutRoot">

        <Grid.ColumnDefinitions>

Изм.	Лист	№ докум.	Подпись	Дата

```

        <ColumnDefinition Width="*" />    <ColumnDefinition Width="*" />
        <ColumnDefinition Width="*" />    <ColumnDefinition Width="*" />
        <ColumnDefinition Width="*" />

    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions><RowDefinition Height="*" /><RowDefinition Height="*" /><RowDefinition Height="*" /></Grid.RowDefinitions>
<!-- оформляю табличное административное меню / использую разные цвета --&gt;
&lt;Label Grid.Row="0" Grid.Column="0" Content="Учителя" Background=
"{StaticResource GreenTileBrush}" Margin="10,10,10,65"&gt;&lt;/Label&gt;
&lt;Button      x:Name="forTeachers"      Grid.Row="0"      Grid.Column="0"
Margin="10,135,10,0"  Content=" Учителю "  FontSize="20"  FontWeight="Bold"
Foreground="WhiteSmoke" Click="forTeachers_Click" HorizontalAlignment="Stretch"
Height="50" Width="auto" Background="Green"/&gt; // Первая кнопка
&lt;Label Grid.Row="0" Grid.Column="1" Content="Студенты" Height="auto"
Background="{StaticResource TomatoTileBrush}" Margin="10,10,10,65"&gt;&lt;/Label&gt;
&lt;Button      x:Name="forTeachStudents"     Grid.Row="0"     Grid.Column="1"
Margin="10,135,10,0"  Content=" Журнал "  FontSize="20"  FontWeight="Bold"
Foreground="WhiteSmoke" Click="forTeachStudents_Click" // Вторая кнопка
HorizontalAlignment="Stretch" Height="50" Width="auto" Background="Tomato"/&gt;
&lt;Label Grid.Row="0" Grid.Column="2" Content="Реестр" Margin="10,10,10,65"
Background="{StaticResource PurpleTileBrush}" &gt;&lt;/Label&gt;
&lt;Button      x:Name="forReestrPC"      Grid.Row="0"      Grid.Column="2"
Margin="10,135,10,0"  Content=" Реестр "  FontSize="20"  FontWeight="Bold"
Foreground="WhiteSmoke" Click="forReestrPC_Click"   HorizontalAlignment=
"Stretch" Height="50" Width="auto" Background="Purple"/&gt; // Третья кнопка
&lt;Label Grid.Row="0" Grid.Column="3" Content="Просмотр *.sql"
Margin="10,10,10,65" Background="{StaticResource TomatoTileBrush}" &gt;&lt;/Label&gt;
&lt;Button      x:Name="forSQLviewer"      Grid.Row="0"      Grid.Column="3"
Margin="10,135,10,0"  Content=" SQL файлы "  FontSize="20"  FontWeight="Bold"
Foreground="WhiteSmoke" Click="forSQLviewer_Click" Height="50" Width="auto"
</pre>

```

HorizontalAlignment="Stretch" Background="Tomato"/> // Четвертая кнопка  
 <Label Grid.Row="0" Grid.Column="4" Content="База \*.mdb" Margin="10,10,10,65"  
 Background="{StaticResource GreenTileBrush}" ></Label>  
 <Button x:Name="forDBvskMSAcc" Grid.Row="0" Grid.Column="4"  
 Margin="10,135,10,0" Content="База данных" FontSize="20" FontWeight="Bold"  
 Foreground="DarkRed" Click="forDBvskMSAcc\_Click" Height="50" Width="auto"  
 HorizontalAlignment="Stretch" Background="Cyan"/> // Пятая кнопка  
 <Label Grid.Row="1" Grid.Column="0" Content="Медиа Плеер"  
 Margin="10,10,10,65" Background="{StaticResource BlueTileBrush}"/>  
 <Button x:Name="forMediaPlayer" Grid.Row="1" Grid.Column="0"  
 Margin="10,135,10,0" Content="Video/Sound" FontSize="20" FontWeight="Bold"  
 Foreground="WhiteSmoke" Click="forMediaPlayer\_Click" HorizontalAlignment=  
 "Stretch" Height="50" Width="auto" Background="Orange"/> // Шестая кнопка  
 <Label Grid.Row="1" Grid.Column="1" Content="Общий журнал"  
 Margin="10,10,10,65" Background="{StaticResource PlumTileBrush}"/>  
 <Button x:Name="forAllMark" Grid.Row="1" Grid.Column="1"  
 Margin="10,135,10,0" Content="Успеваемость" FontSize="20" FontWeight="Bold"  
 Foreground="WhiteSmoke" Click="forAllMark\_Click" HorizontalAlignment="Stretch"  
 Height="50" Width="auto" Background="Purple"/> // Седьмая кнопка  
 <Label Grid.Row="1" Grid.Column="2" Content="Библиотека" Margin="10,10,10,65"  
 Background="{StaticResource DarkRedTileBrush}"/>  
 <Button x:Name="forLibBooks" Grid.Row="1" Grid.Column="2"  
 Margin="10,135,10,0" Content="Библиотека" FontSize="20" FontWeight="Bold"  
 Foreground="WhiteSmoke" Click="forLibBooks\_Click" HorizontalAlignment=  
 "Stretch" Height="50" Width="auto" Background="SeaGreen"/> // Восьмая кнопка  
 <Label Grid.Row="1" Grid.Column="3" Content="Пусто" Margin="10,10,10,65"  
 Background="{StaticResource PlumTileBrush}"/>  
 <Label Grid.Row="1" Grid.Column="4" Content="База данных" Margin="10,10,10,65"  
 Background="{StaticResource BlueTileBrush}"/>

```

<Button x:Name="forDBvskAccess" Grid.Row="1" Grid.Column="4"
Margin="10,135,10,0" Content=" База *.accdb " FontSize="20" FontWeight="Bold"
Foreground="DarkRed" Click="forDBvskAccess_Click" Height="50" Width="auto"
HorizontalAlignment="Stretch" Background="Cyan"/> // Девятая кнопка

<!-- возврат к блокировке окна -->

<Button x:Name="btnLock" Grid.Row="2" Grid.Column="2"
FontStretch="Normal" FontWeight="Bold" Foreground="Red"
Background="WhiteSmoke" FontSize="22" Content="Блокировка"
Click="btnLock_Click" Margin="20,70,20,70" Width="Auto" />

<!-- английские надписи в полях берутся из файла "WPFSmartLibraryLight35"
поэтому их не могу изменить -->

<wpfsl:SmartLoginOverlay x:Name="VskAdminControl" FullSpan="On"
IsUserOptionAvailable="True" UserName="{Binding UserName}" Password="{Binding
Password}" UserImageSource="{Binding UserImageSource}"
AdditionalSystemInfo="Введите Ваш "Логин" и "Пароль"" Command="{Binding
SubmitCommand}" CommandParameter="{Binding RelativeSource={RelativeSource
Self} }" /> <!-- здесь через binding веду проверка введённых логина и пароля,
которые указаны в файле "ViewModels/LoginViewModel.cs" -->

<!-- далее (ниж окна) можно установить другие свойства для фона -->

<!-- у меня (синий цвет основной), внизу - что требуется от пользователя -->

<Label Grid.RowSpan="3" Grid.ColumnSpan="5" FontSize="11" HorizontalAlignment=
"Center" VerticalAlignment="Bottom" Foreground="Yellow" Margin="0,50"
Content="Для получения административных прав необходимо ввести:" />
<Label Grid.RowSpan="3" Grid.ColumnSpan="5" FontSize="10" HorizontalAlignment=
"Center" VerticalAlignment="Bottom" Foreground="LightPink" Margin="0,30"
FontWeight="Bold" Content=""Логин -> UserName": * * * * * и "/>
<Label Grid.RowSpan="3" Grid.ColumnSpan="5" FontSize="10" HorizontalAlignment=
"Center" VerticalAlignment="Bottom" Foreground="LightPink" Margin="0,10"
FontWeight="Bold" Content=""Пароль -> Password": * * * * * . /> </Grid>
</Window>

```

Изм.	Лист	№ докум.	Подпись	Дата

В файле проекта **LoginViewModel.cs** устанавливаю следующие правила для входа (*это два пользователя с административными правами*):

```
private void getAllUser()
{
    // здесь можно реализовать код для большего числа пользователей
    this.userList = new List<User>()
    {
        new User() { UserName="Admin", Password="VskVI2020",
                    ImageSourcePath = Path.Combine( userImagePath, "WomenAdmin.png" ) },
        new User() { UserName="AdminDefault", Password="Release",
                    ImageSourcePath = Path.Combine( userImagePath, "MenAdmin.png" ) },{};{}}
    }
```

При вводе пользователем '**UserName**' в соответствующее поле (Рисунок 30), будет меняться и картинка (путь к ней – «**ImageSourcePath**» в ресурсах проекта):

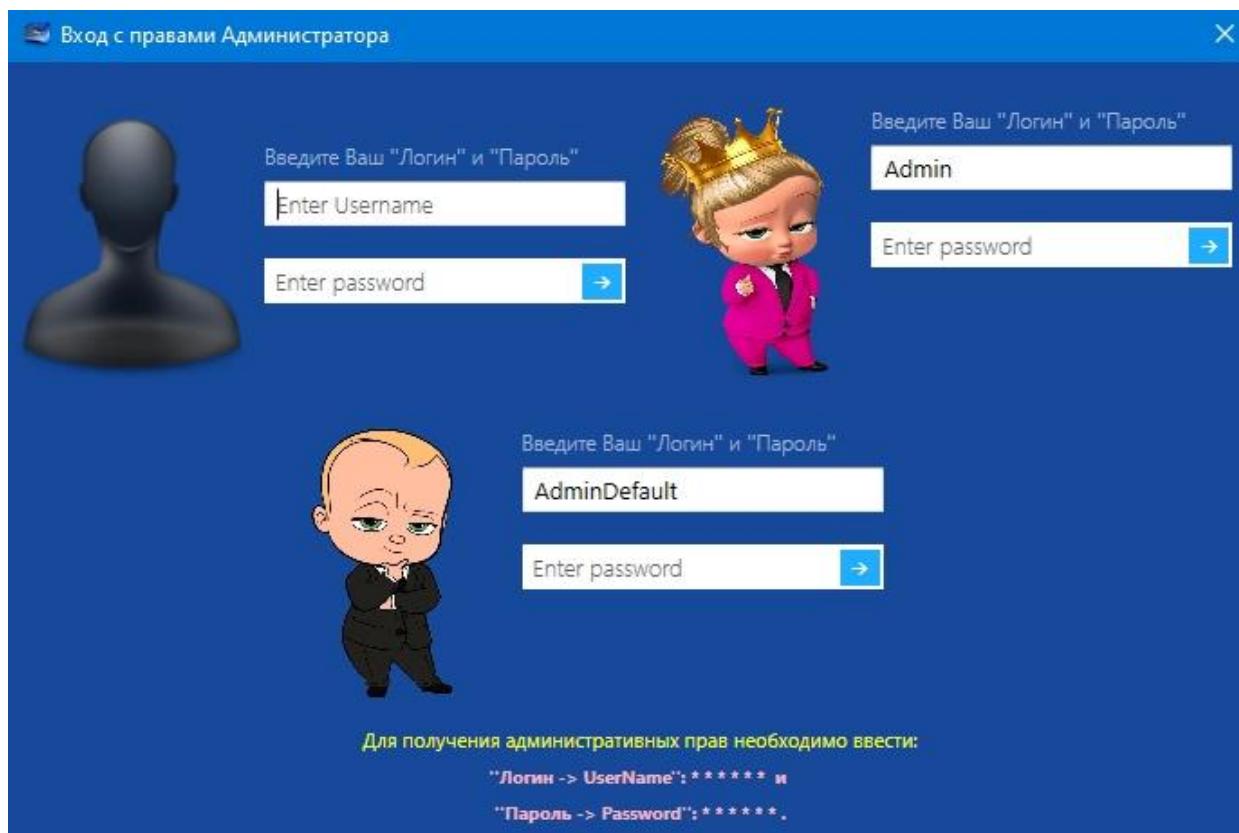


Рисунок № 30 – Смена картинки при входе с права администратора

После осуществления входа в данном окне отобразятся кнопки для запуска других, скомпилированных мной, приложений (Рисунок 31).

Изм.	Лист	№ докум.	Подпись	Дата



Рисунок № 31 – Меню программ для администратора

### Создание базы данных \*.mdb для учёта сдачи тестов

Использую Microsoft Access профессиональный плюс 2019 — MSO (16.0.13628.20318), 64-разрядная версия. Создаю небольшую базу данных, в формате ‘Access 2002-2003’, для учёта студентов, которые сдали или не сдали общее тестирование (Рисунок 32).

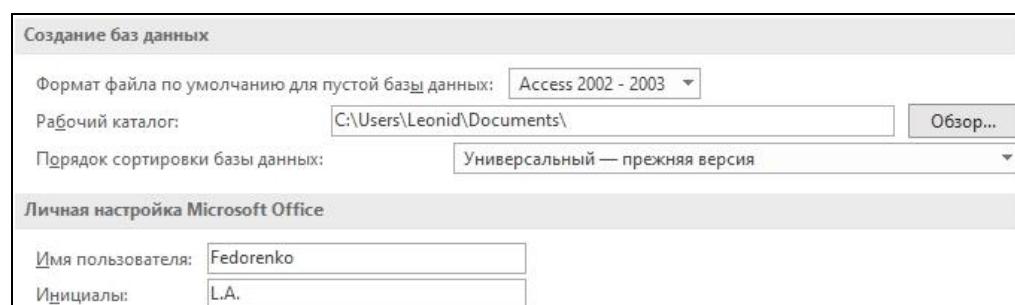


Рисунок № 32 – Формат файла базы данных Access 2002-2003

Создаю следующие объекты Access:

- 1) таблицы "tblStudents" (Рисунок 34) и "tblCourse";
- 2) общий запрос на выборку данных из существующих таблиц, в зависимости от выбранных на форме условий: "SearchRequest";
- 3) форму "Поиск среди студентов (с результатами тестирования)".

Изм.	Лист	№ докум.	Подпись	Дата	Лист	143
					ВСК.090204.07.00.000 ПЗ	

Чтобы вручную не заполнять таблицу студентов, перехожу на вкладку «**Внешние данные**» и использую уже созданный мною источник данных из файла **Excel** (страница 51 / Рисунок 33).

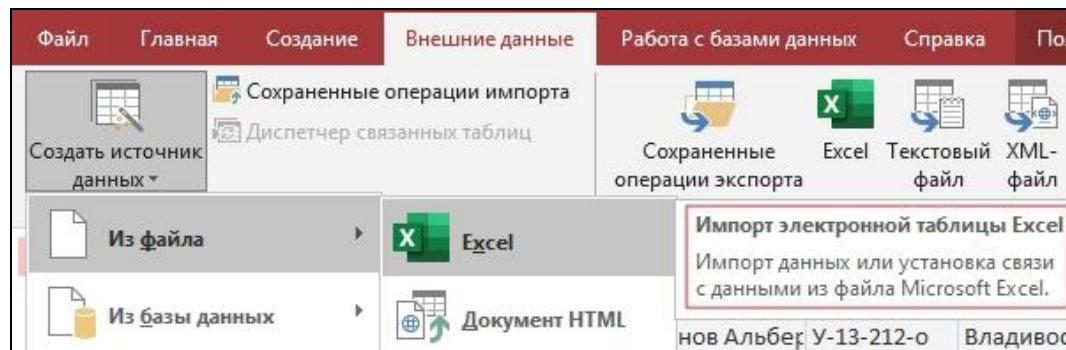


Рисунок № 33 – Импорт таблицы из Excel

St	Exar	StudentFIO	Group	City
1	<input checked="" type="checkbox"/>	Федоренко Леонид Александрович	У-14-216-з	Владивосток/У-14-216-з
2	<input type="checkbox"/>	Хохлова Наталья Валерьевна	У-21-108-о	Уссурийск/У-21-108-о
3	<input type="checkbox"/>	Мазур Евгений Владимирович	У-13-210-о	Партизанск/У-13-210-о
4	<input checked="" type="checkbox"/>	Цаплина Ирина Витальевна	У-21-220-о	Находка/У-21-220-о
5	<input type="checkbox"/>	Абрамов Геннадий Иванович	У-09-113-о	Артем/У-09-113-о
6	<input checked="" type="checkbox"/>	Пантелеимонов Альберт Геннадьевич	У-13-212-о	Владивосток/У-13-212-о
7	<input checked="" type="checkbox"/>	Мукосеева Стела Витальевна	У-21-128-о	Артём/У-21-128-о
8	<input checked="" type="checkbox"/>	Яговенко Андрон Миронович	У-14-219-з	Артём/У-14-219-з
9	<input checked="" type="checkbox"/>	Котяш Изяслав Яковович	У-13-128-о	Находка/У-13-128-о
10	<input type="checkbox"/>	Панов Серафим Феоктистович	У-12-138-о	Владивосток/У-12-138-о
11	<input checked="" type="checkbox"/>	Янушкене Виссарион Адольфович	У-13-138-о	Владивосток/У-13-138-о
12	<input checked="" type="checkbox"/>	Басманова Ираида Геннадьевна	У-14-229-о	Владивосток/У-14-229-о
13	<input checked="" type="checkbox"/>	Швардигула Бронислав Геннадьевич	У-21-228-о	Уссурийск/У-21-228-о
14	<input type="checkbox"/>	Блинова Оксана Станиславовна	У-12-228-о	Ливадия/У-12-228-о
15	<input checked="" type="checkbox"/>	Терехова Ефросиния Борисовна	У-13-128-з	Владивосток/У-13-128-з
16	<input checked="" type="checkbox"/>	Шидловский Макар Самсонович	У-14-219-о	Владивосток/У-14-219-о
17	<input checked="" type="checkbox"/>	Кружков Прокл Пахомович	У-21-128-з	Владивосток/У-21-128-з
18	<input checked="" type="checkbox"/>	Михальченко Федот Семёнович	У-12-128-о	Владивосток/У-12-128-о

Рисунок № 34 – Таблица «tblStudents»

В общем запросе размещаю следующие условия (Рисунок 35):

1. «Like "\*" & [Forms]![frmTestStudentSearch]![txtFilterMainName] & "\*"» — выборка по ФИО студента;
2. «[Forms]![frmTestStudentSearch]![txtFilterCity]» — выборка по городу проживания;
3. «[Forms]![frmTestStudentSearch]![txtFilterLevel]» — выборка по номеру курса;

4. «<DateAdd("d";1;[Forms]![frmTestStudentSearch]![txtFilterEndDate])» и

«>=[Forms]![frmTestStudentSearch]![txtStartDate]» — выборки по дате.

[Forms]	StudentFIO tblStudents	City tblStudents	Kурс tblStudents	Дата поступления tblStudents
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
IsNull				
IsNull	Like "*" & [Forms]![frm			
IsNull		[Forms]![frmTestStude		
IsNull	Like "*" & [Forms]![frm	[Forms]![frmTestStude		
IsNull			[Forms]![frmTestStude	
IsNull	Like "*" & [Forms]![frm			[Forms]![frmTestStude
IsNull				[Forms]![frmTestStude
IsNull	Like "*" & [Forms]![frm	[Forms]![frmTestStude	[Forms]![frmTestStude	
IsNull				[Forms]![frmTestStude
IsNull	Like "*" & [Forms]![frm	[Forms]![frmTestStude	[Forms]![frmTestStude	
IsNull				[Forms]![frmTestStude
IsNull	Like "*" & [Forms]![frm	[Forms]![frmTestStude	[Forms]![frmTestStude	>=[Forms]![frmTestStude
IsNull				>=[Forms]![frmTestStude

Рисунок № 35 – Часть общего запроса

Проверяю работу базы данных в MS Access (Рисунок 35).

Поиск по результатам тестов

### Форма поиска по результатам тестирования.

Критерий отбора:

Тип	Строка поиска	Город/Группа	Курс	Диапазон дат
Все	ен	Владивосток	2	От <input type="text"/> До <input type="text"/> <span>ФИЛЬР</span> Убрать

Результат отбора:

№	Зачет (Да/Нет)	ФИО студента	Город/Группа	Курс	Дата тестирования
18	<input checked="" type="checkbox"/>	Михальченко Федот Сократович	Владивосток/У-12-128-о	2	03.04.2018 14:19:41
35	<input checked="" type="checkbox"/>	Бурякова Ефросинья Семеновна	Владивосток/У-13-142-о	2	03.04.2018 14:19:41
54	<input type="checkbox"/>	Басманова Ираида Геннадьевна	Владивосток/У-14-229-о	2	17.08.2016 17:43:01
164	<input checked="" type="checkbox"/>	Ярмоненко Архип Викентьевич	Владивосток/У-14-219-о	2	17.08.2016 17:43:01
217	<input checked="" type="checkbox"/>	Гречченко Мартын Викентьевич	Владивосток/У-24-236-о	2	17.08.2016 17:43:01
274	<input checked="" type="checkbox"/>	Сивкова Валентина Ивановна	Владивосток/У-12-138-о	2	17.08.2016 17:43:01
297	<input checked="" type="checkbox"/>	Лукьяннова Валерия Евгеньевна	Владивосток/У-13-120-з	2	17.08.2016 17:43:01
329	<input checked="" type="checkbox"/>	Мукаконен Ирина Олеговна	Владивосток/У-14-114-о	2	17.08.2016 17:43:01
362	<input checked="" type="checkbox"/>	Кристман Екатерина Евгеньевна	Владивосток/У-22-121-о	2	17.08.2016 17:43:01
373	<input checked="" type="checkbox"/>	Александрова Екатерина Евгеньевна	Владивосток/У-13-128-з	2	17.08.2016 17:43:01
416	<input checked="" type="checkbox"/>	Ожегова Елена Юрьевна	Владивосток/У-14-229-з	2	17.08.2016 17:43:01
417	<input checked="" type="checkbox"/>	Пестrikova Ольга Евгеньевна	Владивосток/У-21-228-о	2	17.08.2016 17:43:01
437	<input checked="" type="checkbox"/>	Редковолосова Евгения Александровна	Владивосток/У-13-120-з	2	17.08.2016 17:43:01
450	<input checked="" type="checkbox"/>	Феофанова Елена Игоревна	Владивосток/У-12-138-о	2	17.08.2016 17:43:01
460	<input checked="" type="checkbox"/>	Леоненко Наталья Васильевна	Владивосток/У-14-229-о	2	17.08.2016 17:43:01
468	<input checked="" type="checkbox"/>	Плахтий Анжелика Геннадьевна	Владивосток/У-22-121-о	2	17.08.2016 17:43:01
504	<input checked="" type="checkbox"/>	Коваленко Ольга Степановна	Владивосток/У-13-211-о	2	17.08.2016 17:43:01
512	<input checked="" type="checkbox"/>	Крицкая Екатерина Евгеньевна	Владивосток/У-12-228-о	2	17.08.2016 17:43:01
543	<input checked="" type="checkbox"/>	Васильева Елена Борисовна	Владивосток/У-13-138-о	2	17.08.2016 17:43:01
547	<input checked="" type="checkbox"/>	Денисюк Евгения Николаевна	Владивосток/У-13-128-з	2	17.08.2016 17:43:01
561	<input checked="" type="checkbox"/>	Петухова Елена Владимировна	Владивосток/У-21-128-о	2	17.08.2016 17:43:01
644	<input checked="" type="checkbox"/>	Зеленова Людмила Ефимовна	Владивосток/У-13-211-о	2	17.08.2016 17:43:01
660	<input checked="" type="checkbox"/>	Волобуева Ольга Евгеньевна	Владивосток/У-12-138-о	2	17.08.2016 17:43:01
746	<input checked="" type="checkbox"/>	Архипченко Тимур Александрович	Владивосток/У-13-142-о	2	17.08.2016 17:43:01
758	<input checked="" type="checkbox"/>	Семенов Дмитрий Владимирович	Владивосток/У-14-219-о	2	17.08.2016 17:43:01
807	<input checked="" type="checkbox"/>	Журавлева Ирина Геннадьевна	Владивосток/У-21-128-о	2	17.08.2016 17:43:01
846	<input checked="" type="checkbox"/>	Худякова Елена Геннадьевна	Владивосток/У-14-236-з	2	17.08.2016 17:43:01
890	<input checked="" type="checkbox"/>	Шуршин Евгений Михайлович	Владивосток/У-13-211-о	2	17.08.2016 17:43:01
924	<input checked="" type="checkbox"/>	Преображенская Анна Викторовна	Владивосток/У-13-211-о	2	17.08.2016 17:43:01
*	<input type="checkbox"/>				04.03.2021 14:22:43

Запрос SQL по установленным условиям:  
 [City] Like "\*Владивосток\*" AND ([StudentFIO] Like "\*ен\*") AND ([Курс] = 2)

Документация: <https://vstehn.ru/home/osn-svedenia>

Запись: 1 из 29 С фильтром Поиск

Рисунок № 36 – Проверка базы данных через критерии отбора

Изм.	Лист	№ докум.	Подпись	Дата	Лист
					ВСК.090204.07.00.000 ПЗ

## Просмотр базы данных \*.mdb из проекта C#

Создаю проект **VskSearchResultTests** (целевая рабочая среда: .NET Framework 4.7) в “Обозреватель решений” добавляю созданный файл базы данных «**VskSearchResultTests.mdb**» (Рисунок 37).

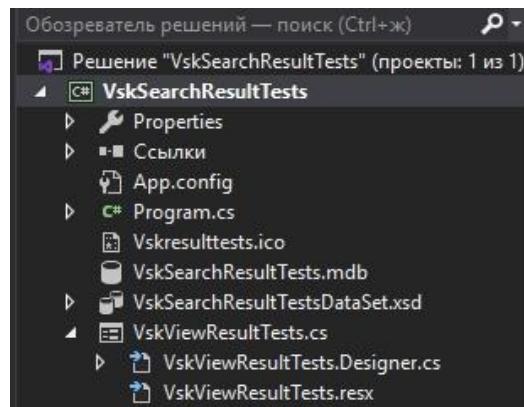


Рисунок № 37 – Добавление файла базы данных

Размещаю на форме «**dataGridView1**», который будет показывать все данные из таблицы "tblStudents". Чтобы осуществить подключение к созданной базе данных перехожу в «Обозреватель серверов» (Рисунок 38).

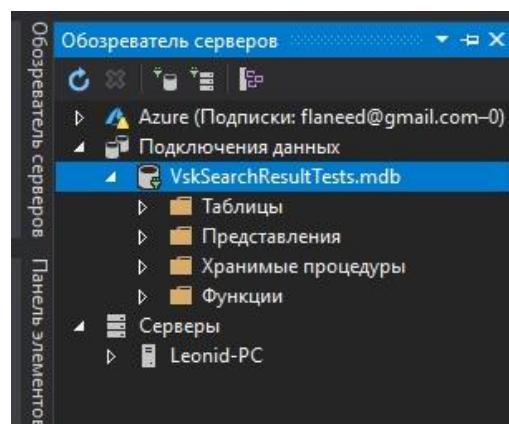


Рисунок № 38 – Обозреватель серверов

В меню «Средства» выбираю «Подключитесь к базе данных...», где выбираю необходимую для экспорта таблицу и устанавливаю «**VskSearchResultTestsDataSet.xsd**». Автоматически создаётся пользовательский инструмент «**MSDataSetGenerator**», преобразующий файл базы данных в процессе разработки (Рисунок 39).

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

146

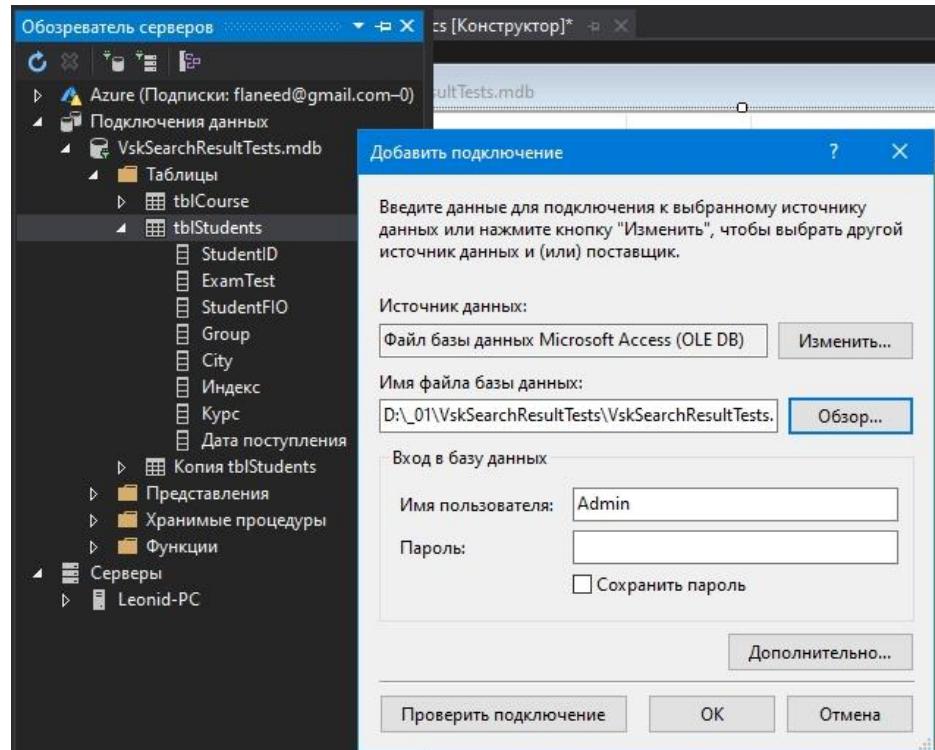


Рисунок № 39 – Подключение файла данных Microsoft Access

Перехожу в свойства «**dataGridView1**» (Рисунок 40) и устанавливаю названия заголовков, их размер и «**ToolTipText**» (при наведении курсора).

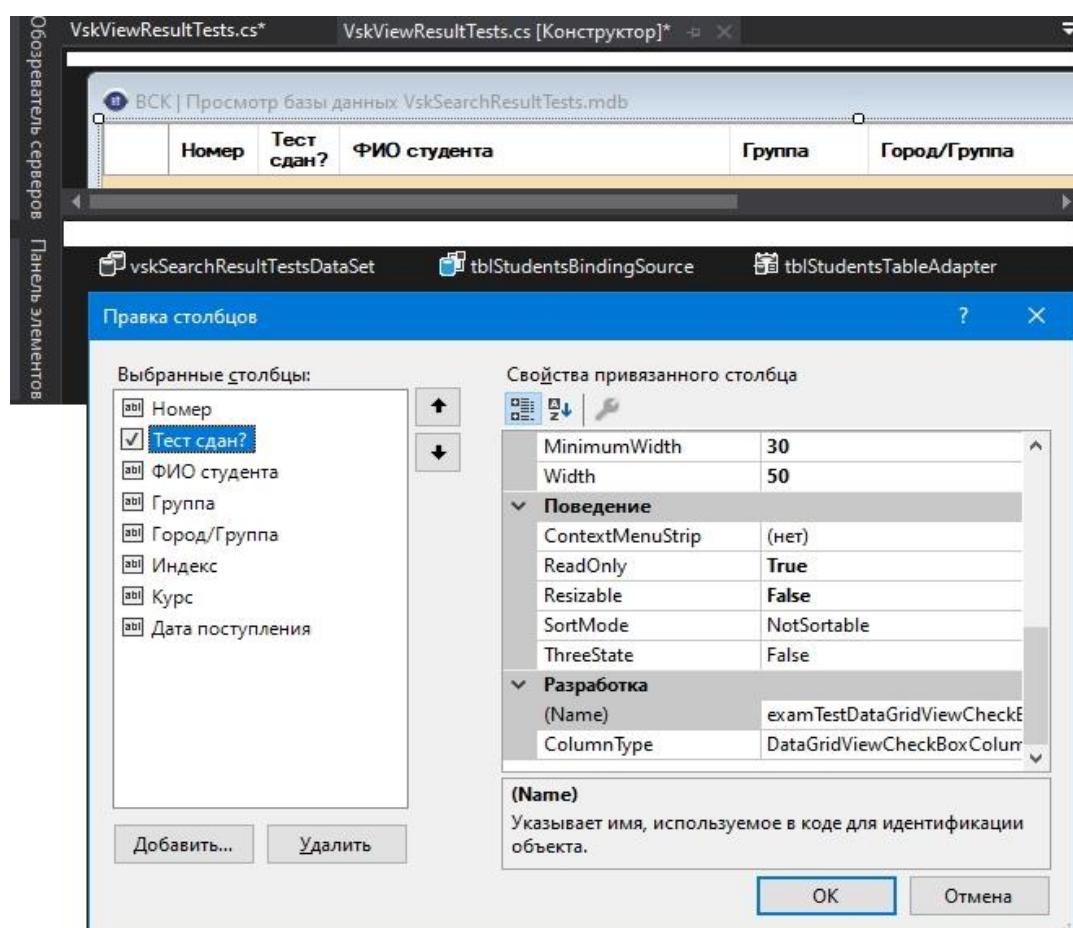


Рисунок № 40 – Правка столбцов в проекте «VskViewResultTest.cs»

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

В разделе свойства «Стиль окна» размещаю файл значка для формы «**Vskresulttests.ico**». После чего компилирую данный проект.

Обязательным условием работы скомпилированного файла является наличие файла «**VskSearchResultTests.mdb**» в той же директории, где находится исполняемый файл.

### Создание и просмотр базы данных \*.accdb для «ВСК»

В Microsoft Access 2019 Создаю простую базу данных «**vskbase.accdb**». Таблицы и запрос представлена на рисунке № 41.

The screenshot shows the Microsoft Access 2019 interface. The title bar reads 'vskbase : база данных - D:\\_01\VskEditBase\vskbase.accdb (Формат файла...)'.

The ribbon tabs are: Файл, Главная, Создание, Внешние данные, Работа с базами данных, Справка.

The left pane shows the 'All objects' list with sections for Таблицы (Tables) and Запросы (Queries). The 'Студенты' table is selected and highlighted in red.

The right pane displays the 'Студенты' table data in a grid format:

Ко	Фамилия	Имя	Отчество	Г
1	Федоренко	Леонид	Александрович	г.Е У-
2	Шушания	Майя	Семёновна	г.Е У-
3	Шкарупа	Дмитрий	Васильевич	г.Р У-
4	Шаляпина	Надежда	Александровна	г.Р У-
5	Хитров	Олег	Яковлевич	с.Г У-
6	Трофимец	Михаил	Иванович	г.Г У-
7	Топузов	Эльдар	Эскендеровна	г.Е У-
8	Суховерхова	Оксана	Васильевна	г.Е У-
9	Мордвин	Владимир	Владимировна	г.Е У-
10	Овсянникова	Ирина	Владимировна	г.Е У-
11	Крючков	Сергей	Борисович	г.Е У-
12	Вольная	Соня	Владимировна	г.Е У-
13	Фесенко	Татьяна	Михайловна	г.Е У-

Рисунок № 41 – База данных «vskbase.accdb»

При создании таблиц «Студенты» и «Учителя» в ‘Конструкторе’ добавляю столбец «Фотография», где «Тип данных» выбираю — «Поле объекта OLE». Это позволяет мне размещать в ячейках данного столбца **bitmap**-изображения.

В Интернетескачу ряд изображений, которые обрабатываю в Adobe Photoshop 2020. Привожу их к одинаковому размеру и сохраняю на жестком диске. После, каждую из этих фотографий, открываю и копирую выделенную область в редакторе и вставляю скопированные данные в ячейки столбца «Фотография», для каждого студента и учителя (Рисунок 42).

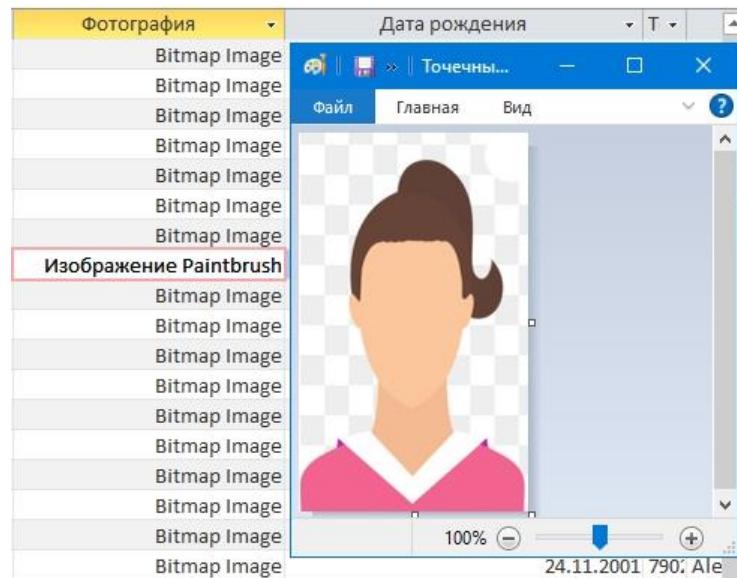


Рисунок № 42 – Столбец «Фотография»

Далее создаю связи между таблицами (Рисунок 43).

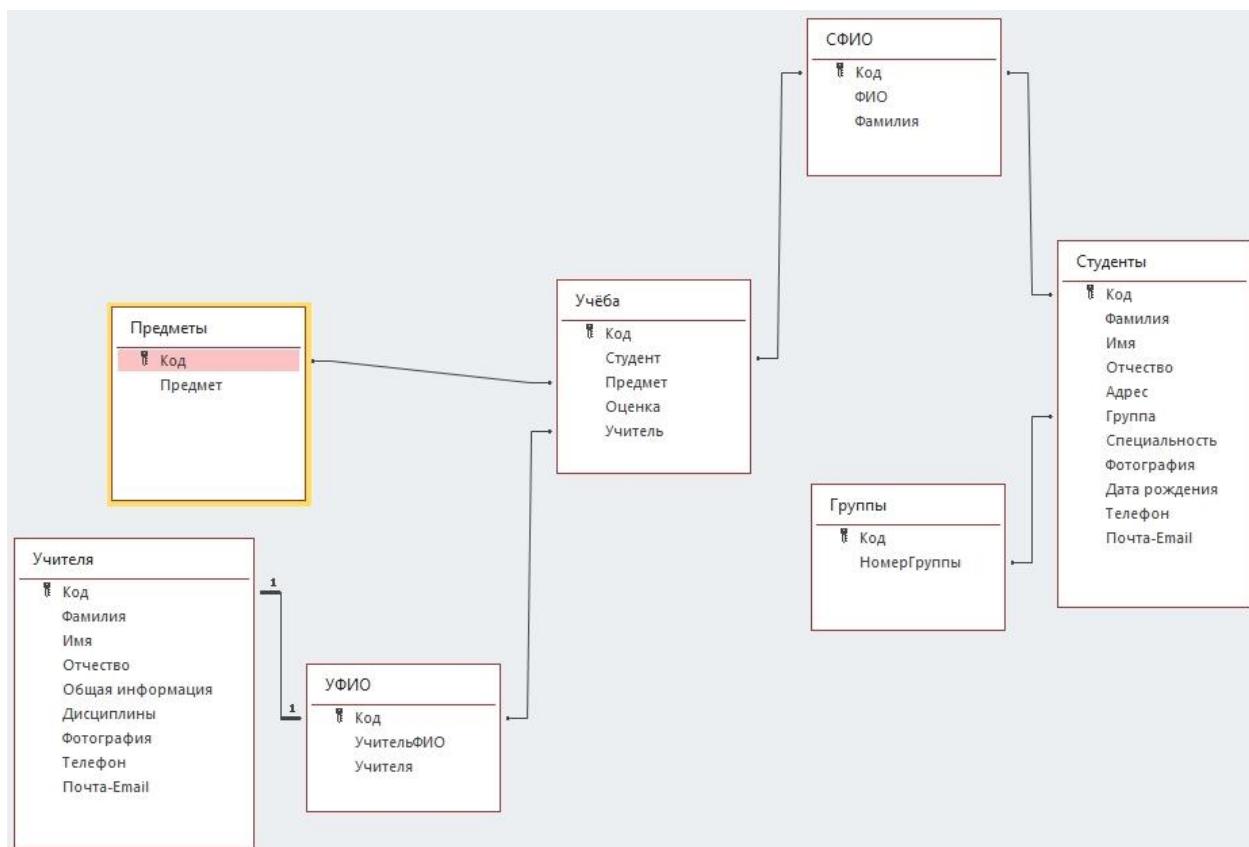


Рисунок № 43 – Схема данных в vskbase.accdb

На форме размещаю «**TabControlVskBase**» с тремя вкладками: **Студенты**, **Учителя**, **Учебный процесс**. Для каждой вкладки отдельно реализую подключение к файлу базы данных: **vskbaseDataSet.xsd**, **vskbaseDataSet1.xsd**, **vskbaseDataSet2.xsd** (Рисунок 44).

Изм.	Лист	№ докум.	Подпись	Дата	Лист	149
					<b>ВСК.090204.07.00.000 ПЗ</b>	

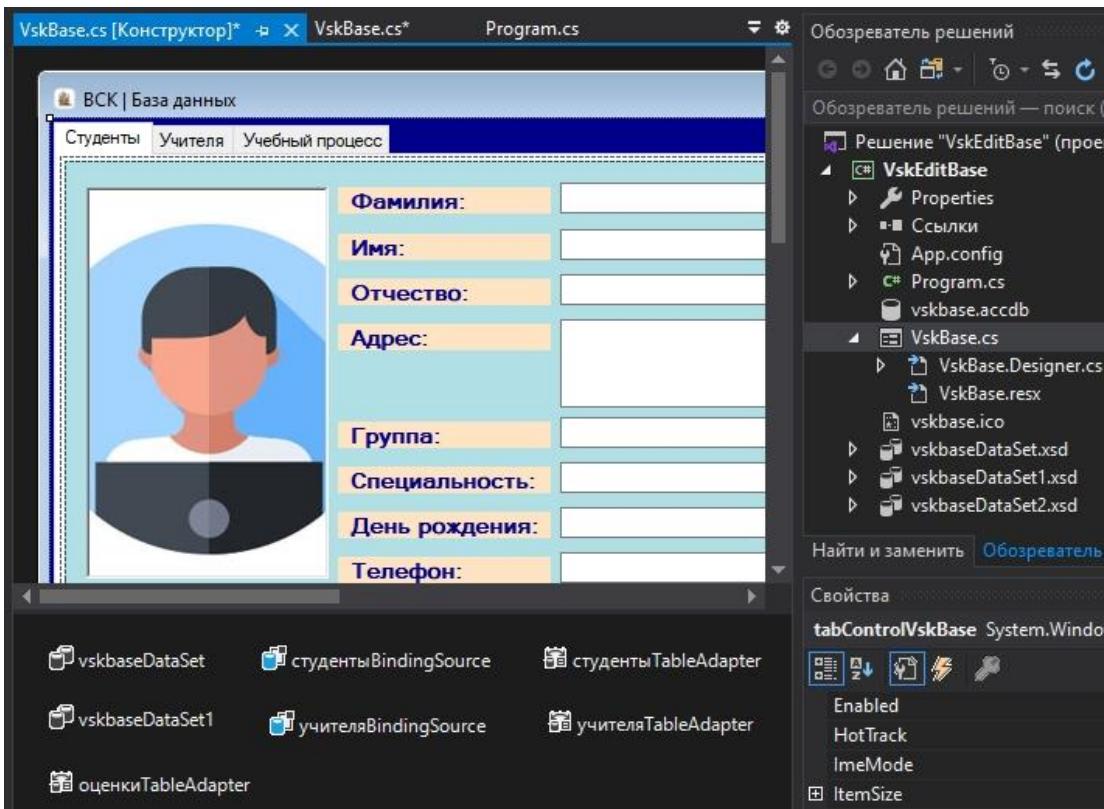


Рисунок № 44 – Три вкладки – три подключения

В проекте реализую следующий программный код (*с комментариями*):

```

public VskBase()
{
    InitializeComponent(); // запрещаю изменение
    panelStudent.Enabled = false;    pnlTeachers.Enabled = false;    }

private void VskBase_Load(object sender, EventArgs e)
{
    // TODO: данная строка кода позволяет загрузить данные в таблицу
    "vskbaseDataSet2.Оценки". Может быть перемещена или удалена.

    this.оценкиTableAdapter.Fill(this.vskbaseDataSet2.Оценки);

    // TODO: данная строка кода позволяет загрузить данные в таблицу
    "vskbaseDataSet1.Учителя". Может быть перемещена или удалена.

    this.учителяTableAdapter.Fill(this.vskbaseDataSet1.Учителя);

    // TODO: данная строка кода позволяет загрузить данные в таблицу
    "vskbaseDataSet.Студенты". Может быть перемещена или удалена.

    this.студентыTableAdapter.Fill(this.vskbaseDataSet.Студенты);    }

private void btnBrowse_Click(object sender, EventArgs e)
{
    // кнопка для фотокарточки
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

try // запускаю файловый диалог для выбора фото
{
    // ставлю фильтр для выбора BMP-изображений
    // множественный выбор запрещаю
    // !!! в базу сохраняются только фото с расширением *.BMP
    using (OpenFileDialog ofdFoto = new OpenFileDialog() { Filter =
        "Фотография или рисунок в формате BMP|*.bmp", ValidateNames = true, Multiselect =
        false }) { // если файл с фото выбран
        if (ofdFoto.ShowDialog() == DialogResult.OK)
            // загружаю фото в "fotoBoxJpeg"
            fotoBoxJpeg.Image = Image.FromFile(ofdFoto.FileName); } }
    catch (Exception ex) { // вывожу сообщение об ошибке
        MessageBox.Show(ex.Message, "Ошибка при загрузке фотографии (файл
        только в формате *.bmp)", MessageBoxButtons.OK, MessageBoxIcon.Error); } }

private void btnTeachPic_Click(object sender, EventArgs e)
{
    try // кнопка для фотокарточки
    {
        // запускаю файловый диалог для выбора фото и т.д. и т.п. (см.выше)
        using (OpenFileDialog ofdFoto = new OpenFileDialog() { Filter =
            "Фотография или рисунок в формате BMP|*.bmp", ValidateNames = true, Multiselect =
            false }) { // если файл с фото выбран
            if (ofdFoto.ShowDialog() == DialogResult.OK)
                // загружаю фото в "fotoBoxJpeg"
                pictureBoxTeacher.Image = Image.FromFile(ofdFoto.FileName); } }
        catch (Exception ex)
        { // вывожу сообщение об ошибке
            MessageBox.Show(ex.Message, "Ошибка при загрузке фотографии (файл
            только в формате *.bmp)", MessageBoxButtons.OK, MessageBoxIcon.Error); } }

private void btnNewStudent_Click(object sender, EventArgs e)
{
    // использую try-catch чтобы отлавливать ошибки
    try { // разрешаю ввод данных
        panelStudent.Enabled = true;
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

// фокус на поле для фамилии
txtSurname.Focus();

// добавление новой строки в таблицу "Студенты"
this.vskbaseDataSet.Студенты.AddСтудентыRow(this.vskbaseDataSet.Студенты.New
СтудентыRow());

// движение в конец таблицы
студентыBindingSource.MoveLast();    }

catch (Exception ex)
{
    // вывожу сообщение об ошибке
    MessageBox.Show(ex.Message, "Ошибка при создании нового студента",
MessageBoxButtons.OK, MessageBoxIcon.Error);

    // данные в базе не меняю
    студентыBindingSource.ResetBindings(false); } }

private void btnNewTeacher_Click(object sender, EventArgs e)
{
    // использую try-catch чтобы отлавливать ошибки
    try { // разрешаю ввод данных
        pnlTeachers.Enabled = true; // фокус на поле для фамилии
        txtSecondNameTeacher.Focus();

        // добавление новой строки в таблицу "Студенты"
        this.vskbaseDataSet1.Учителя.AddУчителяRow(this.vskbaseDataSet1.Учителя.
NewУчителяRow()); // движение в конец таблицы
        учителяBindingSource.MoveLast(); }

    catch (Exception ex) { // вывожу сообщение об ошибке
        MessageBox.Show(ex.Message, "Ошибка при создании нового учителя",
MessageBoxButtons.OK, MessageBoxIcon.Error); // данные в базе не меняю
        учителяBindingSource.ResetBindings(false); } }

private void btnEditStudent_Click(object sender, EventArgs e)
{
    // разрешаю изменение
    panelStudent.Enabled = true;
    // фокус на поле для фамилии
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

txtSurname.Focus(); }

private void btnEditTeacher_Click(object sender, EventArgs e)
{
    // разрешаю изменение
    pnlTeachers.Enabled = true;
    // фокус на поле для фамилии
    txtSecondNameTeacher.Focus();
}

private void btnCancelStudent_Click(object sender, EventArgs e)
{
    // завершаю операции (ничего не меняю в базе)
    panelStudent.Enabled = false;
    студентыBindingSource.ResetBindings(false);
}

private void btnCancelEditTeacher_Click(object sender, EventArgs e)
{
    // завершаю операции (ничего не меняю в базе)
    pnlTeachers.Enabled = false;
    учителяBindingSource.ResetBindings(false);
}

private void btnSaveStudent_Click(object sender, EventArgs e)
{
    // использую try-catch чтобы отлавливать ошибки
    try { // завершаю редактирование
        студентыBindingSource.EndEdit();
        // и обновляю данные в таблице
        студентыTableAdapter.Update(this.vskbaseDataSet.Студенты);
        // завершаю редактирование (запрещаю)
        panelStudent.Enabled = false; }

    catch (Exception ex) { // вывожу сообщение об ошибке
        MessageBox.Show(ex.Message, "Ошибка при сохранении данных",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
        // данные в базе не меняю
        студентыBindingSource.ResetBindings(false); } }

private void btnSaveTeacher_Click(object sender, EventArgs e)
{
    // использую try-catch чтобы отлавливать ошибки
    try { // завершаю редактирование
}

```

```

учителяBindingSource.EndEdit();
// и обновляю данные в таблице
учителяTableAdapter.Update(this.vskbaseDataSet1.Учителя);
// завершаю редактирование (запрещаю)
pnlTeachers.Enabled = false; }

catch (Exception ex) {
// вывожу сообщение об ошибке
MessageBox.Show(ex.Message, "Ошибка при сохранении данных",
MessageBoxButtons.OK, MessageBoxIcon.Error);

// данные в базе не меняю
учителяBindingSource.ResetBindings(false); } }

private void txtBoxSearch_TextChanged(object sender, KeyPressEventArgs e)
{
// для поиска
if (e.KeyChar == (char)13)
{ // проверка на пустоту
if (string.IsNullOrEmpty(textBoxSearch.Text))
    dataGridViewStudents.DataSource = студентыBindingSource;
else { // ищу в таблице для всех полей совпадение с текстом
var query = from o in this.vskbaseDataSet.Студенты
            where o.Фамилия.Contains(textBoxSearch.Text) ||
o.Имя.Contains(textBoxSearch.Text) || o.Отчество.Contains(textBoxSearch.Text) ||
o.Группа.Contains(txtGroup.Text) || o.Адрес.Contains(textBoxSearch.Text) ||
o._Почта_Email.Contains(txtEmail.Text)
select o;

// выбираю найденное и обновляю DataGridView
dataGridViewStudents.DataSource = query.ToList(); } } }

private void txtBoxSearch_TeacherChanged(object sender, KeyPressEventArgs e)
{
// для поиска
if (e.KeyChar == (char)13) { // проверка на пустоту
if (string.IsNullOrEmpty(txtTeachBoxSearch.Text))
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        dataGridViewTechers.DataSource = учителяBindingSource;
    else { // ищу в таблице для всех полей совпадение с текстом
        var query = from o in this.vskbaseDataSet1.Учителя
                    where o.Фамилия.Contains(txtTeachBoxSearch.Text) ||
o.Имя.Contains(txtTeachBoxSearch.Text) || o.Отчество.Contains(txtTeachBoxSearch.
Text) || o.Общая_информация.Contains(txtTeachBoxSearch.Text) ||
o.Дисциплины.Contains(txtTeachBoxSearch.Text) ||
                    o._Почта_Email.Contains(txtTeachBoxSearch.Text)
        select o; // выбираю найденное и обновляю DataGrid
        dataGridViewTechers.DataSource = query.ToList(); } } }

```

Ниже привожу пример работы данного проекта (Рисунок 45).

The screenshot shows a Windows application window titled 'BCK | База данных'. The main interface consists of several input fields for a teacher's personal information:

- Фамилия:** Федоренко
- Имя:** Леонид
- Отчество:** Александрович
- Адрес:** г. Владивосток, Первореченский р-н, ул. Марченко 15, кв. 100, инд. 690087
- Группа:** У-14-226-з
- Специальность:** 09.03.02 "Информационные системы"
- День рождения:** 31.01.1981
- Телефон:** +79644444473
- Почта (Email):** flaneed@gmail.com

Below these fields is a yellow button labeled 'Изменить \*.bmp'.

At the bottom of the form, there are five buttons: 'ПОИСК:' (Search), 'Новый' (New), 'Правка' (Edit), 'Завершить' (Finish), and 'Сохранить' (Save).

Below the form is a table listing several teachers:

Код	Фамилия	Имя	Отчество	Адрес
1	Федоренко	Леонид	Александрович	г. Владивосток, Первореченский р-н, ул. Марченко 15, кв. 100, инд. 690087
2	Шушания	Майя	Семёновна	г. Владивосток, Первореченский р-н, ул. Балляева 34, кв. 18, инд. 690087
3	Шкарупа	Дмитрий	Васильевич	г. Артем, Лазовский р-н, ул. Карла-Маркса 12, кв. 304, инд. 690203
4	Шаляпина	Надежда	Александровна	г. Владивосток, ул. Новожилова 21, кв. 28 инд. 690011
5	Хитров	Олег	Яковлевич	с. Покровка, ул. Красноармейская, д. 14, инд. 692561

Рисунок № 45 (часть) – Работа проекта «BCK | База данных»

## Экспорт списка учителей «ВСК» в MS Excel

Создаю проект «[VskDataGridViewToExcel](#)». Разбиваю окно на две строки:  
`<Grid.RowDefinitions> <RowDefinition Height="50"/> <RowDefinition Height="*"/>`  
`</Grid.RowDefinitions>.`

На первой строке устанавливаю одну кнопку, для передачи данных в Excel:  
`<Button Grid.Row="0" HorizontalAlignment="Right" Width="330" Height="35"`  
`ToolTip="Export to Excel" FontSize="18" Margin="0,0,20,0" Content="Экспорт`  
`списка в Microsoft Excel" Foreground="BlueViolet" FontWeight="ExtraBold"`  
`Click="btnEmployee_Click" Name="btnEmployee"/>.`

Всю вторую строку занимает **DataGrid**, где представлены учителя, у которых есть доступ на вход в программу «Курс обучения «Оператор ЭВМ». Данный элемент заполняю через программный код, уже имеющимися у меня данными:

1. Файл **WindowExcel.XAML**, оформление DataGrid:

```
... <toolkit:DataGrid Grid.Row="1" Name="dgEmployee" SelectionMode="Single"
Background="Aqua" AutoGenerateColumns="False" IsReadOnly="False" Margin="5,5,
5,5"> <toolkit:DataGrid.Columns> // привязка к колонкам через 'binding'
<toolkit:DataGridTextColumn Binding="{Binding Path=ID}" Header="№ п/п"
Width="70" />
<toolkit:DataGridTextColumn Binding="{Binding Path=SurName}" Header="Фамилия"
Width="120" />
<toolkit:DataGridTextColumn Binding="{Binding Path=Name}" Header="Имя"
Width="80" />
<toolkit:DataGridTextColumn Binding="{Binding Path=MidName}"
Header="Отчество" Width="120" />
<toolkit:DataGridTextColumn Binding="{Binding Path=Email}" Header="Почта"
Width="*" />
<toolkit:DataGridTextColumn Binding="{Binding Path=Phone}" Header="Телефон"
Width="110" />
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

156

```

<toolkit:DataGridTextColumn Binding="{Binding Path=Course Header='Курс'
Width='50' />
<toolkit:DataGridTextColumn Binding="{Binding Path=Group}" Header="Группа"
Width="90"/>
<toolkit:DataGridTextColumn Binding="{Binding Path=Pass}" Header="Пароль"
Width="70"/>
</toolkit:DataGrid.Columns> </toolkit:DataGrid> ...

```

2. Файл **WindowExcel.XAML.cs**, в котором **программным** кодом осуществляю заполнение **DataGrid**-а (Рисунок 46):

```

public WindowExcel() { InitializeComponent();
    Employees Teachers = new Employees();
    Employee teacher_1 = new Employee();
    teacher_1.ID = "0000001"; teacher_1.SurName = "Федоренко"; teacher_1.Name =
    "Леонид"; teacher_1.MidName = "Александрович"; teacher_1.Email =
    "flaneed@gmail.com"; teacher_1.Phone = "+79644444473"; teacher_1.Course = "5";
    teacher_1.Group = "У-14-237-з"; teacher_1.Pass = "qwerty1"; Teachers.Add(teacher_1);
    Employee teacher_2 = new Employee();
    teacher_2.ID = "0000002"; teacher_2.SurName = "Толок"; teacher_2.Name = "Ирина";
    teacher_2.MidName = "Евгеньевна"; teacher_2.Email = "distanc.tolok.20@gmail.com";
    teacher_2.Phone = "*****"; teacher_2.Course = "0"; teacher_2.Group = "0-00-
    VSK-0"; teacher_2.Pass = "qwerty2"; Teachers.Add(teacher_2);
    Employee teacher_3 = new Employee();
    teacher_3.ID = "0000003"; teacher_3.SurName = "Лысенко"; teacher_3.Name =
    "Иван"; teacher_3.MidName = "Александрович"; teacher_3.Email = "bragili@ya.ru";
    teacher_3.Phone = "+79502814792"; teacher_3.Course = "0"; teacher_3.Group = "0-00-
    VSK-0"; teacher_3.Pass = "qwerty3"; Teachers.Add(teacher_3);
    Employee teacher_4 = new Employee();
    teacher_4.ID = "0000004"; teacher_4.SurName = "Овчинникова"; teacher_4.Name =
    "Лариса"; teacher_4.MidName = "Александровна"; teacher_4.Email = "zaochvsk@"

```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

157

```

yandex.ru"; teacher_4.Phone = "+79242337133"; teacher_4.Course = "0"; teacher_4.
Group = "0-00-VSK-0"; teacher_4.Pass = "qwerty4"; Teachers.Add(teacher_4);

Employee teacher_5 = new Employee();

teacher_5.ID = "0000005"; teacher_5.SurName = "Андреев"; teacher_5.Name =
"Вячеслав"; teacher_5.MidName = "Владимирович"; teacher_5.Email = "andreev_vv@vstehn.ru";
teacher_5.Phone = "*****"; teacher_5.Course = "0"; teacher_5.Group =
"0-00-VSK-0"; teacher_5.Pass = "qwerty5"; Teachers.Add(teacher_5); // и т.д.....
Employee teacher_44 = new Employee();

teacher_44.ID = "0000044"; teacher_44.SurName = "Шмонин"; teacher_44.Name =
"Игорь"; teacher_44.MidName = "Викторович"; teacher_44.Email = "shmonin_iv@vstehn.ru";
teacher_44.Phone = "*****"; teacher_44.Course = "0"; teacher_44.
Group = "0-00-VSK-0"; teacher_44.Pass = "qwerty44" ; Teachers.Add(teacher_44);

dgEmployee.ItemsSource = Teachers; } // заполняю этими данными

```

Экспорт таблицы в Excel

Экспорт списка в Microsoft Excel

№ п/п	Фамилия	Имя	Отчество	Почта	Телефон	Курс	Группа	Пароль
0000001	Федоренко	Леонид	Александрович	flaneed@gmail.com	+79644444473	5	У-14-237-з	qwerty1
0000002	Толок	Ирина	Евгеньевна	distanc.tolok.20@gmail.c	*****	0	0-00-VSK-0	qwerty2
0000003	Лысенко	Иван	Александрович	bragili@ya.ru	+79502814792	0	0-00-VSK-0	qwerty3
0000004	Овчинникова	Лариса	Александровна	zaochvsk@yandex.ru	+79242337133	0	0-00-VSK-0	qwerty4
0000005	Андреев	Вячеслав	Владимирович	andreev_vv@vstehn.ru	*****	0	0-00-VSK-0	qwerty5
0000006	Антохина	Светлана	Павловна	antohina_sp@vstehn.ru	*****	0	0-00-VSK-0	qwerty6
0000007	Барвинок	Людмила	Сергеевна	barvinok_ls@vstehn.ru	*****	0	0-00-VSK-0	qwerty7
0000008	Буртасов	Александр	Иванович	burtasov_ai@vstehn.ru	*****	0	0-00-VSK-0	qwerty8
0000009	Бабенко	Евгений	Николаевич	babenko_en@vstehn.ru	*****	0	0-00-VSK-0	qwerty9
0000010	Байкин	Сергей	Александрович	baikin_sa@vstehn.ru	*****	0	0-00-VSK-0	qwerty10
0000011	Герман	Елена	Дмитриевна	german_ed@vstehn.ru	*****	0	0-00-VSK-0	qwerty11
0000012	Головин	Дмитрий	Иванович	golovin_di@vstehn.ru	*****	0	0-00-VSK-0	qwerty12
0000013	Гулевич	Лариса	Сергеевна	gulevich_ls@vstehn.ru	*****	0	0-00-VSK-0	qwerty13
0000014	Даниленко	Юлия	Владимировна	danielenko_uv@vstehn.ru	*****	0	0-00-VSK-0	qwerty14
0000015	Данилова	Людмила	Николаевна	danilova_ln@vstehn.ru	*****	0	0-00-VSK-0	qwerty15
0000016	Кононова	Ольга	Владимировна	kononova_ov@vstehn.ru	*****	0	0-00-VSK-0	qwerty16
0000017	Котенко	Юлия	Сергеевна	kotenko_us@vstehn.ru	*****	0	0-00-VSK-0	qwerty17
0000018	Крысанова	Надежда	Александровна	kresanova_na@vstehn.ru	*****	0	0-00-VSK-0	qwerty18
0000019	Красикова	Елена	Викторовна	krasikova_ev@vstehn.ru	*****	0	0-00-VSK-0	qwerty19
0000020	Крюков	Алексей	Алексеевич	kryukov_aa@vstehn.ru	*****	0	0-00-VSK-0	qwerty20
0000021	Лебедев	Игорь	Владимирович	lebedev_iv@vstehn.ru	*****	0	0-00-VSK-0	qwerty21
0000022	Литошенко	Денис	Александрович	litoshenko_da@vstehn.ru	*****	0	0-00-VSK-0	qwerty22
0000023	Лобова	Татьяна	Жановна	lobova_tj@vstehn.ru	*****	0	0-00-VSK-0	qwerty23
0000024	Лукина	Кира	Владимировна	lukina_kv@vstehn.ru	*****	0	0-00-VSK-0	qwerty24
0000025	Лысенко	Иван	Александрович	liesenko_ia@vstehn.ru	*****	0	0-00-VSK-0	qwerty25
0000026	Миргеев	Андрей	Александрович	mirgeev_aa@vstehn.ru	*****	0	0-00-VSK-0	qwerty26
0000027	Миллер	Наталья	Валерьевна	miller_nv@vstehn.ru	*****	0	0-00-VSK-0	qwerty27

Рисунок № 46 – Экспорт DataGrid в MS Excel

Изм.	Лист	№ докум.	Подпись	Дата	Лист	ВСК.090204.07.00.000 ПЗ	158
------	------	----------	---------	------	------	-------------------------	-----

3. Для обращения к созданным таким образом данным необходим **класс**:

```
public class Employee
{ private string Id;
    public string ID { get { return Id; } set { Id = value; } }
    private string surName;
    public string SurName { get { return surName; } set { surName = value; } }
    private string name;
    public string Name { get { return name; } set { name = value; } }
    private string midName;
    public string MidName { get { return midName; } set { midName = value; } }
    private string email;
    public string Email { get { return email; } set { email = value; } }
    private string phone;
    public string Phone { get { return phone; } set { phone = value; } }
    private string course;
    public string Course { get { return course; } set { course = value; } }
    private string group;
    public string Group { get { return group; } set { group = value; } }
    private string pass; // здесь можно увидеть какой у Вас пароль на вход!!!
    public string Pass { get { return pass; } set { pass = value; } }}
```

4. Также необходим класс и для ‘генерации’ файла **Excel** (Рисунок 47):

```
public class VskDataGridViewToExcel<T, U>
{
    where T : class
    where U : List<T>
    {
        public List<T> dataToPrint; // ввожу необходимые ссылки на объекты Excel
        private Excel.Application _excelApp = null; private Excel.Workbooks _books = null;
        private Excel._Workbook _book = null; private Excel.Sheets _sheets = null; private
        Excel._Worksheet _sheet = null; private Excel.Range _range = null; private Excel.Font
        _font = null; // далее необязательная переменная аргумента
```

Изм.	Лист	№ докум.	Подпись	Дата

ВСК.090204.07.00.000 ПЗ

Лист

159

```

private object _optionalValue = Missing.Value;
// для создания отчета и подфункций
public void GenerateReport()
{ try { if (dataToPrint != null) { if (dataToPrint.Count != 0) {
Mouse.SetCursor(Cursors.Wait); CreateExcelRef(); FillSheet(); OpenReport();
Mouse.SetCursor(Cursors.Arrow); } } }
catch (Exception e) {
    MessageBox.Show("Ошибка при создании отчета Excel"); }
finally { ReleaseObject(_sheet); ReleaseObject(_sheets);
ReleaseObject(_book); ReleaseObject(_books); ReleaseObject(_excelApp); } }

// делаю приложение MS Excel видимым
private void OpenReport() { _excelApp.Visible = true; }

// заполняю лист Excel
private void FillSheet() { object[] header = CreateHeader();
WriteData(header); }

// записываю данные в таблицу Excel
private void WriteData(object[] header)
{ object[,] objData = new object[dataToPrint.Count, header.Length];
for (int j = 0; j < dataToPrint.Count; j++) { var item = dataToPrint[j];
for (int i = 0; i < header.Length; i++) {
var y = typeof(T).InvokeMember(header[i].ToString(), BindingFlags.GetProperty,
null, item, null);
objData[j, i] = (y == null) ? "" : y.ToString(); } } }
AddExcelRows("A2", dataToPrint.Count, header.Length, objData);
AutoFitColumns("A1", dataToPrint.Count + 1, header.Length); }

// метод автоматической подгонки столбцов в соответствии с данными
private void AutoFitColumns(string startRange, int rowCount, int colCount)
{ _range = _sheet.get_Range(startRange, _optionalValue);
_range = _range.get_Resize(rowCount, colCount);
_range.Columns.AutoFit(); }

```

Изм.	Лист	№ докум.	Подпись	Дата

```

// создаю заголовок из полученных свойств
private object[] CreateHeader()
{
    PropertyInfo[] headerInfo = typeof(T).GetProperties();

// создаю массив для заголовков и добавляю его на лист, начиная с ячейки A1
    List<object> objHeaders = new List<object>();
    for (int n = 0; n < headerInfo.Length; n++)
    {
        objHeaders.Add(headerInfo[n].Name); }
    var headerToAdd = objHeaders.ToArray();
    AddExcelRows("A1", 1, headerToAdd.Length, headerToAdd);
    SetHeaderStyle(); return headerToAdd; }

```

	A	B	C	D	E	F	G	H	I
1	ID	SurName	Name	MidName	Email	Phone	Co	Group	Pass
2	1	Федоренкс	Леонид	Александрови	flaneed@gmail.com	8E+10	5	У-14-237-з	qwerty1
3	2	Толок	Ирина	Евгеньевна	distanc.tolok.20@gmail	*****	0	0-00-VSK-0	qwerty2
4	3	Лысенко	Иван	Александрови	bragili@ya.ru	8E+10	0	0-00-VSK-0	qwerty3
5	4	Овчиннико	Лариса	Александрови	zaochvsk@yandex.ru	8E+10	0	0-00-VSK-0	qwerty4
6	5	Андреев	Вячесла	Владимирович	andreev_vv@vstehn.ru	*****	0	0-00-VSK-0	qwerty5
7	6	Антохина	Светла	Павловна	antohina_sp@vstehn.ru	*****	0	0-00-VSK-0	qwerty6
8	7	Барвинок	Людмил	Сергеевна	barvinok_ls@vstehn.ru	*****	0	0-00-VSK-0	qwerty7
9	8	Буртасов	Алексан	Иванович	burtasov_ai@vstehn.ru	*****	0	0-00-VSK-0	qwerty8
10	9	Бабенко	Евгений	Николаевич	babenko_en@vstehn.ru	*****	0	0-00-VSK-0	qwerty9
11	10	Байкин	Сергей	Александрови	baikin_sa@vstehn.ru	*****	0	0-00-VSK-0	qwerty10
12	11	Герман	Елена	Дмитриевна	german_ed@vstehn.ru	*****	0	0-00-VSK-0	qwerty11
13	12	Головин	Дмитрий	Иванович	golovin_di@vstehn.ru	*****	0	0-00-VSK-0	qwerty12
14	13	Гулевич	Лариса	Сергеевна	gulevich_ls@vstehn.ru	*****	0	0-00-VSK-0	qwerty13
15	14	Даниленкс	Юлия	Владимирови	danienko_uv@vstehn.i	*****	0	0-00-VSK-0	qwerty14
16	15	Данилова	Людмил	Николаевна	danilova_ln@vstehn.ru	*****	0	0-00-VSK-0	qwerty15
17	16	Кононова	Ольга	Владимирови	kononova_ov@vstehn.r	*****	0	0-00-VSK-0	qwerty16
18	17	Котенко	Юлия	Сергеевна	kotenko_us@vstehn.ru	*****	0	0-00-VSK-0	qwerty17
19	18	Крысанова	Надежда	Александрови	kresanova_na@vstehn.	*****	0	0-00-VSK-0	qwerty18
20	19	Красикова	Елена	Викторовна	krasikova_ev@vstehn.r	*****	0	0-00-VSK-0	qwerty19
21	20	Крюков	Алексей	Алексеевич	kryukov_aa@vstehn.ru	*****	0	0-00-VSK-0	qwerty20
22	21	Лебедев	Игорь	Владимирови	lebedev_iv@vstehn.ru	*****	0	0-00-VSK-0	qwerty21
23	22	Литошенкс	Денис	Александрови	litoshenko_da@vstehn.	*****	0	0-00-VSK-0	qwerty22

Рисунок № 47 – Файл MS Excel с паролями на вход (колонка I)

Изм.	Лист	№ докум.	Подпись	Дата	Лист	ВСК.090204.07.00.000 ПЗ	161
------	------	----------	---------	------	------	-------------------------	-----

```

// устанавливаю стиль заголовка полужирным шрифтом

private void SetHeaderStyle()
{
    _font = _range.Font;      _font.Bold = true; }

// метод добавления строк в Excel

private void AddExcelRows(string startRange, int rowCount, int colCount,
object values) {
    _range = _sheet.get_Range(startRange, _optionalValue);
    _range = _range.get_Resize(rowCount, colCount);
    _range.set_Value(_optionalValue, values); }

// для создания экземпляров параметров приложения Excel

private void CreateExcelRef()
{
    _excelApp = new Excel.Application();
    _books = (Excel.Workbooks)_excelApp.Workbooks;
    _book = (Excel._Workbook)(_books.Add(_optionalValue));
    _sheets = (Excel.Sheets)_book.Worksheets;
    _sheet = (Excel._Worksheet)(_sheets.get_Item(1)); } // только 1 лист

```

5. В файле **MainWindow.xaml.cs** размещаю код для запуска данного проекта, где можно увидеть, какой установлен пароль для каждого преподавателя:

```

private void vskTeachPasswords_Click(object sender, RoutedEventArgs e)
{
    // экспорт списка DataGrid в Excel (+ видно пароль)

    ProcessStartInfo forTeachPasswords = new ProcessStartInfo();
    forTeachPasswords.FileName = @"C:\OperatorIBM\VskDataGridToExcel
\VsksDataGridToExcel.exe";

    forTeachPasswords.CreateNoWindow = true;
    forTeachPasswords.UseShellExecute = true;
    Process forTeachPasswordsProcess = Process.Start(forTeachPasswords); }

```

Кнопку для запуска выше представленного кода и для просмотра указанной таблицы размещаю в самом первом окне проекта (Рисунок 48):

Изм.	Лист	№ докум.	Подпись	Дата	Лист
					<b>ВСК.090204.07.00.000 ПЗ</b>

```

<!-- кнопка для вывода DataGrid в файл Excel со столбцом для пароля -->
<Button x:Name="vskTeachPasswords" ToolTip="Экспорт DataGrid в Excel
(with passwords)" BorderThickness="0" Width="50" Height="50"
Background="Transparent" FontSize="20" Opacity="1" Margin="-685,748,0,0"
Style="{StaticResource gameChess}" FontStyle="Normal"
Cursor="Cursors/exportData.ani" Click="vskTeachPasswords_Click">
    <StackPanel Orientation="Horizontal">
        <Image Source="Images/Button/excel.png" Height="50" Width="50" />
    </StackPanel> </Button>

```



Рисунок № 48 – Кнопка для экспорта в Excel

## Тестирование

В самом начале (11 стр.) я заявлял, что данная программа, может быть, пригодна для обучения и тестирования по самым разнообразным дисциплинам. Поэтому я создал несколько тестов, а именно (*привожу ToolTip's, которые появляются при прохождении курсора над клавишей* (Рисунок 49)):

1. Пройти общий лёгкий тест (*встроен в AllLoader*);
2. Пройти тест **по базам данных** ([VskTest01](#));
3. Пройти тест **по операционной системе Windows** ([VskTest00](#));
4. Пройти тест "Алгоритмы и теория вычислений" ([VskTest02](#));
5. Тест **по основам разработки HTML** ([VskTest03](#));
6. Тест по "**Visual Basic for Application**" ([VskTest05](#));
7. Тест **по компьютерной графике** ([VskTest06](#));
8. Тест **по технологии программирования** ([VskTest09](#));
9. Тест **по MS Word** ([VskTest10](#));
10. Тест **по MS Excel** ([VskTest11](#));

Изм.	Лист	№ докум.	Подпись	Дата	Лист	163
					<b>BCK.090204.07.00.000 ПЗ</b>	

11. Тест "Назначение и архитектура ПК" ([VskTest13](#));
12. Для теста по C++ ([VskTest15](#));
13. Второй тест по C++ ([VskTest16](#));
14. Третий тест по C++ ([VskTest17](#));
15. Тест по метрологии и стандартизации ([VskTest12](#));
16. Тест по теме "Бухгалтерский учет" ([VskTest08](#));
17. Для теста по теме "Восточные славяне в VI-IX веках" ([VskTest07](#));
18. Тест по "Философии" ([VskTest14](#));
19. Тест по "Английскому языку" ([VskTest04](#));
20. Пройти тест по языку программирования C# (*встроен в AllLoader*).

Рисунок № 49 – 20 тестирований, запускаемые из AllLoader



Ниже я привожу пример программного кода для теста «[VskTest00](#)». Так как все тесты (18 шт.), созданные мной отдельно от главного проекта, своим программным кодом никак друг от друга не отличаются — меняется лишь общее число вопросов и количество вопросов, случайно выбранных, для тестирования, на которые нужно ответить (от 10 до 20 шт., но это число может быть совершенно любым).

#### namespace [VskTest00](#)

```
{ public partial class VskFrmTest : Form      {
    string file_name = "vsktest00.dat"; // файл в котором я сохранил
    вопросы и ответы, установив следующие условия: 1) первая строка это сам вопрос;
    2) вторая строка это правильный ответ на вопрос 3)-4)-5) неправильные ответы
    string[,] massiv; // массив для текстовых строк
```

Изм.	Лист	№ докум.	Подпись	Дата

```

int total, kolvo_voprosov, praviln_otvetov, nepraviln_otvetov;

// переменные: всего строк-вопросов, общее число вопросов именно в тесте, ответы:
// 'правильные' и 'неверные'

static Random rand = new Random(); // генератор случайных чисел

public VskFrmTest()
{
    // загружаю форму InitializeComponent();
    // загружаю тест init_test(); }

private void init_test()
{
    // инициализирую новый тест / надпись на кнопке
    button_next.Text = "СЛЕДУЮЩИЙ ВОПРОС ТЕСТА";
    // всего вопросов в тесте
    kolvo_voprosov = 10; // в этом тесте даётся всего 10 вопросов из XXX
    // вначале правильных и неправильных по нолям
    praviln_otvetov = 0; // правильно      nepraviln_otvetov = 0; // неправильно
    // загрузка файла с вопросами      load_file();
    // выбор radio_checked(); // убрать radio_turn_on_off();
    // фон для результата
    label_result.BackColor = System.Drawing.Color.DarkBlue;
    // метка для результата - красным цветом
    label_result.ForeColor = System.Drawing.Color.Fuchsia;
    // но вначале теста ничего нет
    label_result.Text = "";      kolvo_voprosov--; // уменьшаю кол-во вопросов
    show_question(); // показываю вопрос (следующий)      }
}

private void load_file()
{
    // загружаю файл с тестом (с вопросами-ответами)
    try { //читываю по строкам
        string[] lines = File.ReadAllLines(file_name, Encoding.UTF8);
        // делю по пять строк
        total = lines.Length / 5;      // первая строка вопрос (в файле)
        // потом четыре строки с ответами
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

massiv = new string[kolvo_voprosov, 5];

// и потом повторяются - заношу всё в массив

int[] temp = new int[kolvo_voprosov];

int j;           int k = 0;

do

{ // цикл, случайно выбираю вопрос из всего количества

    j = rand.Next(0, total) * 5;

    if (!temp.Contains(j)) // для каждого вопроса

    { // и переставляю ответы

        massiv[k, 0] = lines[j];

        for (int i = 1; i < 5; i++)

            massiv[k, i] = lines[j + i];

        temp[k] = j;      k++;

    } // продолжаю это, пока ещё есть вопросы

} while (!(k == kolvo_voprosov)); }

catch (Exception ex) { MessageBox.Show(ex.Message); }

private void show_question()

{ // показываю вопрос(ы)

    radio_checked(); // радиобаттон

    label_question.Text = massiv[kolvo_voprosov, 0]; // вопрос – строка из массива

    radio_tags(rand.Next(1,7)); // случайная расстановка ответов

    radio_answer1.Text = massiv[kolvo_voprosov, Convert.ToInt16(radio_answer1.Tag)];

    radio_answer2.Text = massiv[kolvo_voprosov, Convert.ToInt16(radio_answer2.Tag)];

    radio_answer3.Text = massiv[kolvo_voprosov, Convert.ToInt16(radio_answer3.Tag)];

    radio_answer4.Text = massiv[kolvo_voprosov, Convert.ToInt16(radio_answer4.Tag)];

}

private void button_next_Click(object sender, EventArgs e) // следующий вопрос

{ // если вопросов больше нет - заново!

    if (kolvo_voprosov < 0) { init_test(); return; }

```

```

if (!(radio_answer1.Checked || radio_answer2.Checked || radio_answer3.Checked ||
      radio_answer4.Checked))

{ // если вообще ничего не выбрано
    MessageBox.Show("Выберите вариант ответа!"); return;
}

} // если выбран один из вариантов – проверяю правильно ли?

if ((radio_answer1.Checked & Convert.ToInt16(radio_answer1.Tag) == 1) ||
    (radio_answer2.Checked & Convert.ToInt16(radio_answer2.Tag) == 1) ||
    (radio_answer3.Checked & Convert.ToInt16(radio_answer3.Tag) == 1) ||
    (radio_answer4.Checked & Convert.ToInt16(radio_answer4.Tag) == 1))
{
    // кол-во вопросов уменьшаю, а кол-во данных ответов увеличиваю
    praviln_otvetov++;
    kolvo_voprosov--;
}

else
{
    // показываю правильный ответ, если пользователь ошибся
    MessageBox.Show("Правильный вариант ответа: "+massiv[kolvo_voprosov, 1]);
    nepraviln_otvetov++;
    kolvo_voprosov--;
}

if (kolvo_voprosov < 0)
{
    // показываю результат теста
    show_result();
    label_question.Text = "Ваш результат теста внизу формы.";
    // предлагаю пройти повторно
    button_next.Text = "Пройти данный тест заново!";
    radio_turn_on_off(); return;
}

show_question();
}

private void radio_turn_on_off()
{
    if (kolvo_voprosov < 0)
    {
        // если вопросов уже нет – скрываю пункты
        radio_answer1.Visible = false;
        radio_answer2.Visible = false;
        radio_answer3.Visible = false;
        radio_answer4.Visible = false;
    }
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

    } else

    { // если вопросы ещё есть - показываю
        radio_answer1.Visible = true;
        radio_answer2.Visible = true;
        radio_answer3.Visible = true;
        radio_answer4.Visible = true;      }      }

private void radio_checked()

{ // при загрузке вопроса никаких отметок об ответе нет!

    radio_answer1.Checked = false;
    radio_answer2.Checked = false;
    radio_answer3.Checked = false;
    radio_answer4.Checked = false;      }

private void radio_tags(int i)

{
    switch (i)

{ // перебор вариантов вопроса-ответа

    case 1: radio_answer1.Tag = 1; radio_answer2.Tag = 4; radio_answer3.Tag = 3;
              radio_answer4.Tag = 2; break;

    case 2: radio_answer1.Tag = 1; radio_answer2.Tag = 4; radio_answer3.Tag = 2;
              radio_answer4.Tag = 3; break;

    case 3: radio_answer1.Tag = 4; radio_answer2.Tag = 1; radio_answer3.Tag = 3;
              radio_answer4.Tag = 2; break;

    case 4: radio_answer1.Tag = 2; radio_answer2.Tag = 3; radio_answer3.Tag = 1;
              radio_answer4.Tag = 4; break;

    case 5: radio_answer1.Tag = 4; radio_answer2.Tag = 1; radio_answer3.Tag = 2;
              radio_answer4.Tag = 3; break;

    case 6: radio_answer1.Tag = 3; radio_answer2.Tag = 2; radio_answer3.Tag = 4;
              radio_answer4.Tag = 1; break;      }

private void show_result()

{ // показываю результаты тестирования (внизу формы)

```

Изм.	Лист	№ докум.	Подпись	Дата

```
label_result.Text = "Правильных ответов: "+praviln_otvetov.ToString()+"\n"+  
"Неправильных ответов: "+nepraviln_otvetov.ToString(); }
```

Скомпилированные проекты для тестов размещаю в каталоге — «C:\OperatorIBM\AllLoader\Pages\Temp».

Кнопка меню со списком программ, по которым происходит обучение, и кнопка, которая выводит список тестов, взаимоисключающие. Т.е. если одно меню открыто, и пользователь выбирает другое меню, то меню, которое уже развернуто, убираю с экрана:

```
private void leftMenu_Click(object sender, RoutedEventArgs e)  
{ // "выпадение" и "скрытие" бокового меню (слева)  
    if (panelOperator1.Width == 50)  
        { panelOperator1.Width = 350;  
            panelOperator2Tests.Width = 0; fon2Panel.Visibility = Visibility.Hidden;  
            fon2Panel.Width = 0; }  
    else { panelOperator1.Width = 50; }  
 } // минимальная ширина 50 бокового меню (слева)  
  
// butRightTest, butResult (не создано) и leftMenu взаимоисключающие  
private void butRightTest_Click(object sender, RoutedEventArgs e)  
{ // "выпадение" и "скрытие" бокового меню (справа)  
    if (panelOperator2Tests.Width == 0)  
        { panelOperator2Tests.Width = 350; fon2Panel.Visibility = Visibility.Visible;  
            panelOperator1.Width = 50; fon2Panel.Width = 293; }  
    else { panelOperator2Tests.Width = 0; fon2Panel.Width = 0; }  
 } // минимальная ширина 0 - бокового меню для тестов (справа)
```

После завершения теста результат высвечивается в нижней части формы.

Изм.	Лист	№ докум.	Подпись	Дата

## **Вывод**

Изучать C# относительно легко. Он относительно молодой (выпущен в 2001 году), но показывает положительную динамику, с растущим сообществом и поддержкой. C# — объектно-ориентированный язык программирования, разработанный Microsoft. Этот язык работает на платформе .NET, которая используется для написания настольных приложений и игр в Windows. Хотя C# также подходит для разработки мобильных приложений. Он построен на языках С и С++. И, по моему мнению, его гораздо проще и удобнее использовать.

Тенденции разработки программного обеспечения, тренды инструментов программирования имеют большую ценность. Ряд новых языков программирования или фреймворков только запущены, а некоторые исчезают почти каждый год. Наряду с динамическими технологиями программного обеспечения инструменты разработки также неустойчивы в своем присутствии на рынке.

C# — простой язык для изучения. У C# огромнейшее онлайн-сообщество, а в интернете полно материалов и курсов для изучения. Данная программа для обучения студентов информационной грамотности — может явится важным источником информации для преподавательского состава и студентов, содействовать их усилиям по достижению целей, сформулированных в учебных программах. Она опирается на современные тенденции, Интернет, учебники, книги, электронные архивы и библиотеки, а также представляет единый подход к обучению различным дисциплинам и может быть интегрирована в официальную систему подготовки. Тем самым, я как бы запускаю катализический процесс, который должен охватить многих людей и способствовать общему развитию их способностей.

Исходный код снабжен комментариями и легко поддаётся расширению и модернизации. Т.е. можно с легкостью не только заменить, но и добавить любые другие обучающие материалы, на любую другую предметную область или расширить уже представленный материал.

Изм.	Лист	№ докум.	Подпись	Дата

**ВСК.090204.07.00.000 ПЗ**

**Лист**

**170**

# Приложение 1

## Начало разработки пакета программ

