

Паспорт программы

Компетенция	Программные решения для бизнеса
Уровень	Начальный
Формат проведения	Онлайн
Время проведения	45 минут
Максимальное количество участников	6 человек
Возрастная категория участников	7-11 класс
Доступность для участников с инвалидностью и ОВЗ	Не адаптировано
Автор программы	Переоридорога Лада Андреевна
Должность	Разработчик ПО

Введение

Компетенция «Программные решения для бизнеса» связана с разработкой информационных систем, которая может включать общение с заказчиками, проектирование, работу с базами данных, разработку настольных/мобильных приложений и многое другое.

Специалисты по программным решениям могут работать в крупных государственных компаниях, среднем и малом бизнесе, а также быть фрилансерами.

Чтобы стать таким специалистом, нужно хорошо знать математику, развивать логику. Для обучения подойдут такие специальности: Программная инженерия, Информационные системы и программирование и др.

Этап	Содержание	Время
Постановка задания	Общая формулировка задания в рамках пробы: Задание: Разработка приложения «Заметки» с использованием текстовых файлов Демонстрация финального результата: Запуск приложения с демонстрацией работоспособности заданных функций	8
Выполнение задания	Обзор языка программирования C# и его возможностей Обзор среды разработки (VS) Создание форм и работа со свойствами: Name, Grid, LinearGradientBrush, GradientStop, Color, Offset, Header, minHeight, BorderBrush, Margin, TextWrapping, VerticalAlignment и тд Создание элементов формы: Label, TextBox, ListBox, Button, TabControl и тд. Написание кода программы Тестирование и отладка программы, проверка результатов	25
Контроль и оценка	Критерии успешного выполнения задания: внимательность, последовательность выполнения шагов, аккуратное оформление графического интерфейса пользователя, творчество студента в выборе цветовых решений и размеров элементов, работоспособность программы Рекомендации для наставника по контролю результата, процедуре оценки: при проверке учитывать работоспособность программы и визуальное оформление графического интерфейса пользователя (цвета, размеры, Рекомендации для наставника по контролю результата, процедуре оценки: при проверке учитывать работоспособность программы и визуальное оформление графического интерфейса пользователя (цвета, размеры, Рекомендации для наставника по контролю результата, процедуре оценки: при проверке учитывать работоспособность программы и визуальное оформление графического интерфейса пользователя (цвета, размеры	7
Рефлексия полученного опыта	Разрабатывается отдельно группой методологов проекта	5

Инфраструктурный лист

Тип	Наименование	Технические характеристики с необходимыми примечаниями	Ед .	Расчет (на группу/на 1 чел.)	Степень необходимости (необходимо/опционально)
Оборудование, ПО	Компьютер в сборе с клавиатурой и мышью	Processor - Intel Core i7; Ethernet - 100/1000 mbps; RAM - 16GB или больше; HDD 500 Gb или больше;	шт	1	необходимо
Оборудование, ПО	Монитор	Мониторы LCD 21" или больше	шт	1	необходимо
Оборудование, ПО	ПО ОС Microsoft Windows	Программное обеспечение ОС Microsoft Windows 10 Pro (Edu) https://www.microsoft.com/ru-ru/windows/get-windows-10	шт	1	необходимо
Оборудование, ПО	ПО Microsoft Office	Программное обеспечение Microsoft Office 2016 или 365 (Word, Excel, Power Point) https://products.office.com/en-us/get-office-oem-download-page	шт	1	необходимо
Оборудование, ПО	ПО Adobe Reader	Программное обеспечение Adobe Reader DC https://get.adobe.com/ru/reader/	шт	1	необходимо
Оборудование, ПО	ПО для архивации	Программное обеспечение WinRAR или 7-Zip http://www.rarlab.com/download.htm http://www.7-zip.org/download.html	шт	1	необходимо
Оборудование, ПО	ПО .NET Framework	Программная платформа .NET Framework 4.6.2 https://www.microsoft.com/net/download/framework	шт	1	необходимо
Оборудование, ПО	ПО Microsoft Visual Studio	Программное обеспечение Microsoft Visual Studio 2017 Community https://www.visualstudio.com/ru/thank-you-downloading-visual-studio/?sku=Community	шт	1	необходимо
Оборудование, ПО	Discord	Программное обеспечение для организации ВКС	шт	1	необходимо

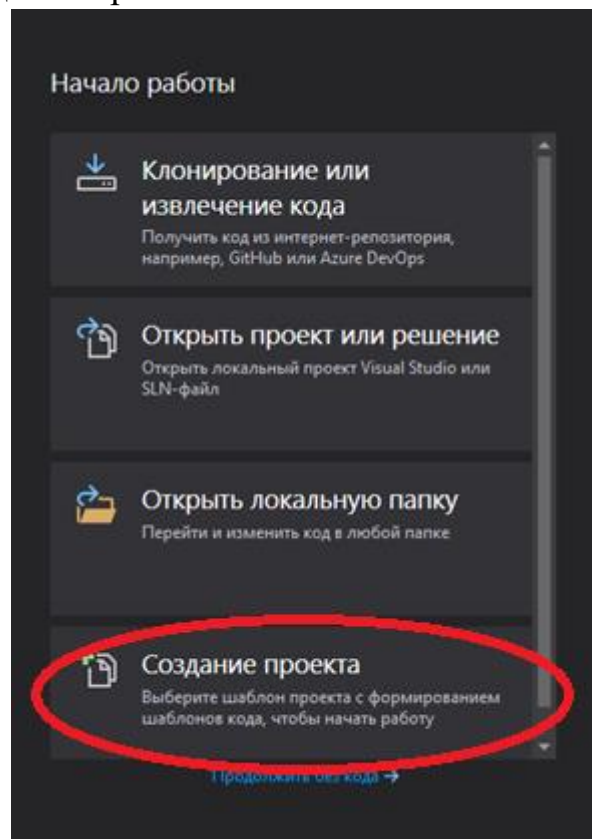
Приложение №1 – Пример выполнения

В данном приложении пример формирования методических рекомендаций, которые необходимо выдать участнику профессиональной пробы! Прошу обратить внимание – не смотря на очный формат работы и возможную демонстрацию работы в VisualStudio на проекторе, предоставление методических рекомендаций по выполнению задания обязательно! Желательно материалы выдавать в печатном виде, чтобы участник мог забрать описание с собой. Данный материал вы можете доработать: добавить руководство по стилю, рассмотреть правила именования идентификаторов и т.д.

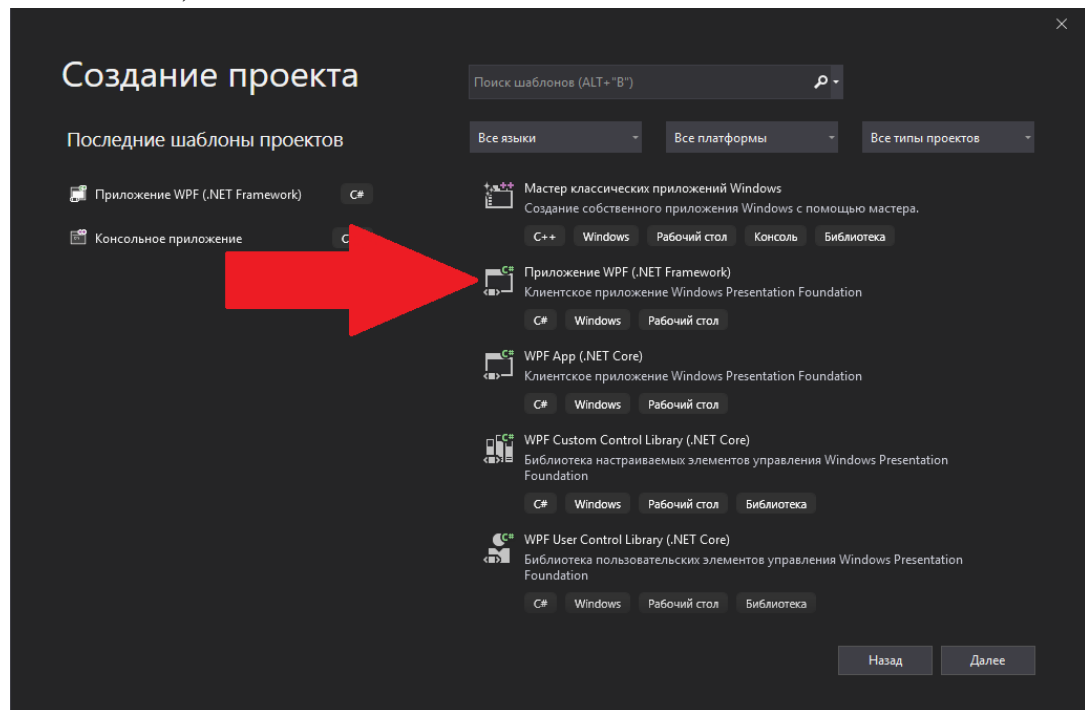
1. Создание проекта в Visual Studio (далее VS)

1.1. Запускаем VS

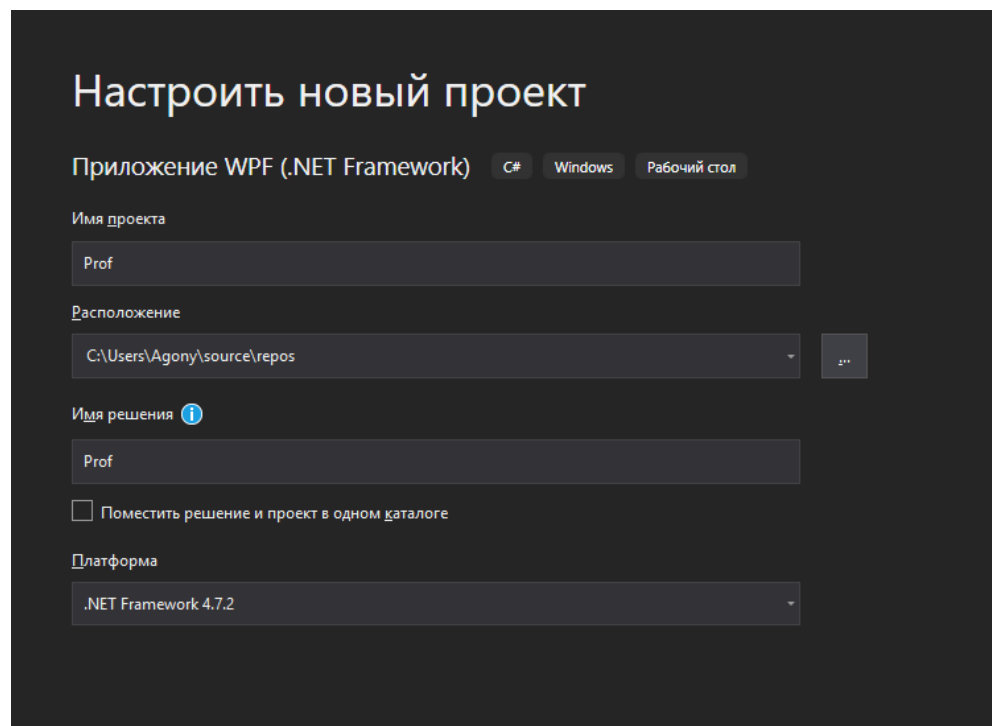
1.2. Выбираем «Создание проекта»



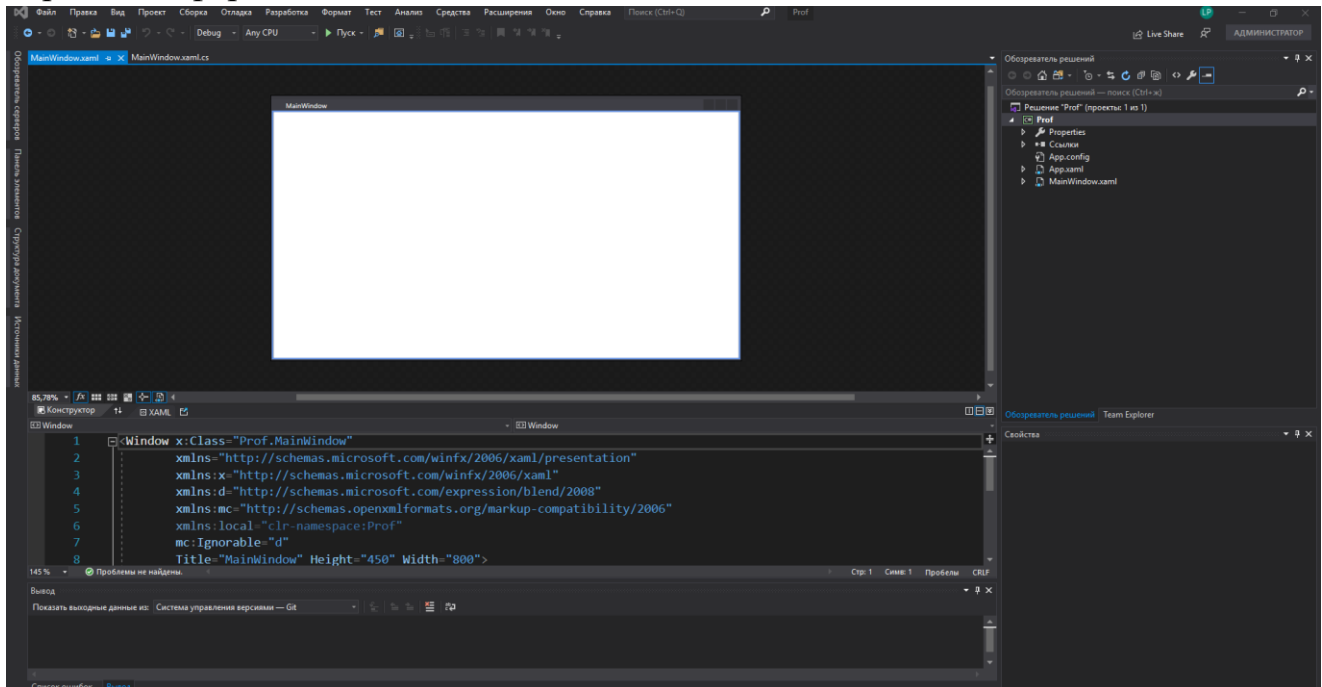
1.3. Выбираем следующую технологию «Приложение WPF(.NET Framework)»



1.4. Указываем расположение файла и его название

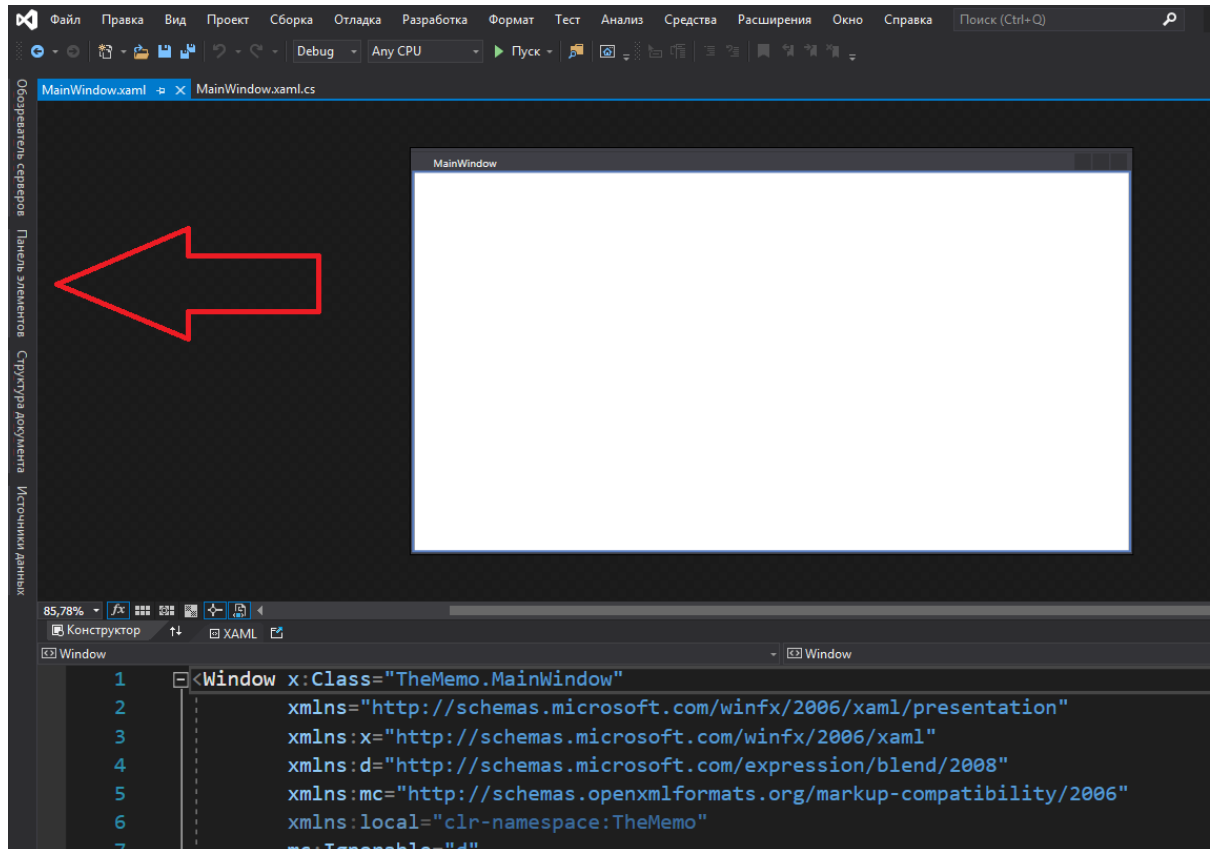


- 1.5. В результате должно образоваться следующее рабочее поле с пустой первичной формой.

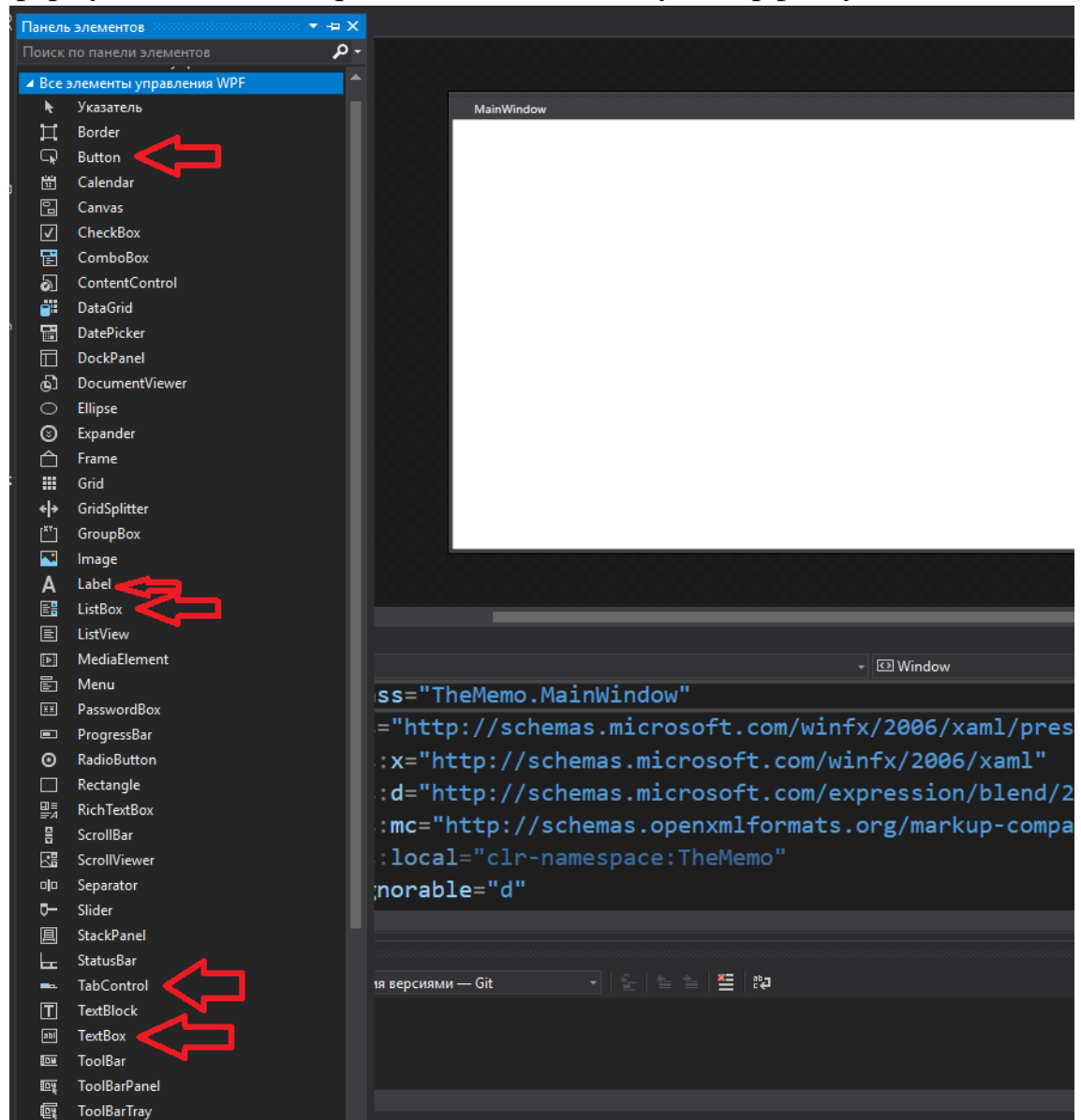


2. Создание главной формы

- 2.1. В Вашем рабочем поле, должна быть вкладка «Панель элементов» (если вдруг ее не наблюдается, то нажмите сочетание клавиш Ctrl+Alt+X).



2.2. Из всех элементов нам понадобятся следующие. Перетаскиваем их на форму. И с помощью разметки задаем всему интерфейсу внешний вид.



```

<Window x:Class="Prof.MainWindow"
    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:Prof"
    mc:Ignorable="d"
    Title="Памятка" Height="450" Width="800">
    <Grid>
        <Grid.Background>
            <LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
                <GradientStop Color="#F5DEB3" Offset="0"/>
                <GradientStop Color="#CCC71CD8" Offset="1"/>
            </LinearGradientBrush>
        </Grid.Background>
        <TabControl Height="auto" Margin="10,40,10,10" Width="auto">
            <TabItem Header="Заметки" Background="#AFEEEE" Width="100">
                <Grid Background="#FFF0F5">

```

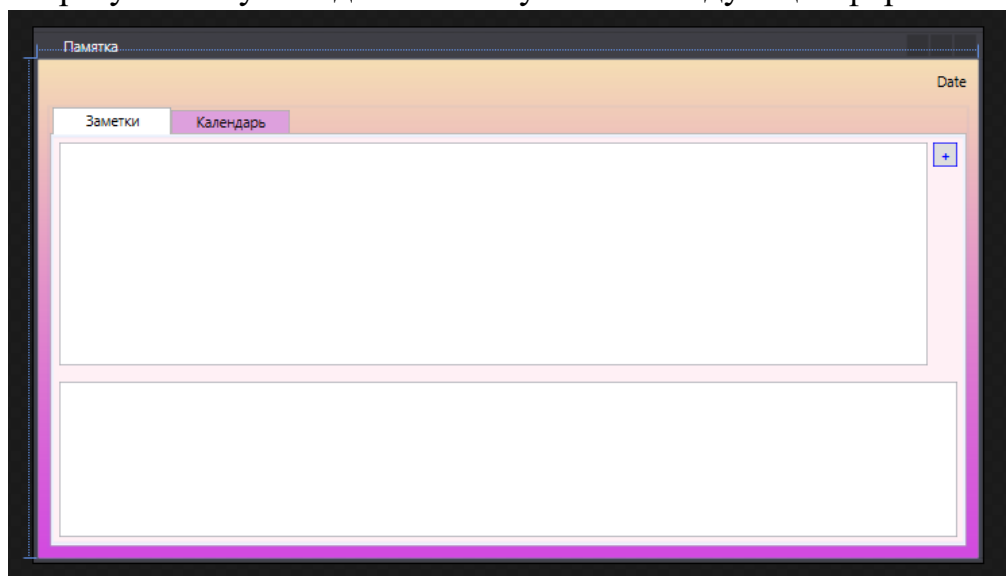
```

        <Button x:Name="button1" Content="+" Foreground="Blue" BorderBrush="Blue"
HorizontalAlignment="Right" Margin="0,5,5,5" VerticalAlignment="Top" Width="20" Height="20"
FontWeight="Bold" MouseDown="button1_MouseDown" Click="button1_Click"/>
        <TextBox x:Name="textbox1" MinHeight="130" Height="130" Margin="5,0,5,5"
TextWrapping="Wrap" Text="" Width="auto" IsReadOnly="True" VerticalAlignment="Bottom"/>
        <ListBox x:Name="listBox1" Height="187" Margin="5,5,30,0" Width="auto"
VerticalAlignment="Top" MouseDown="listBox1_MouseDown"
SelectionChanged="listBox1_SelectionChanged" />
    </Grid>
</TabItem>
<TabItem Header="Календарь" Background="#DDA0DD" Width="100">
    <Grid Background="#FFF0F5"/>
</TabItem>
</TabControl>
<Label x:Name="label1" Content="Date" HorizontalAlignment="Right" Margin="0,5,5,5"
VerticalAlignment="Top"/>

</Grid>
</Window>

```

2.3. В результате у Вас должна получиться следующая форма:



3. Программирование элементов формы и событий

3.1. Нажимаем «F7» и переходим в код формы.

3.2. Подключаем следующие директивы:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using System.IO;
```

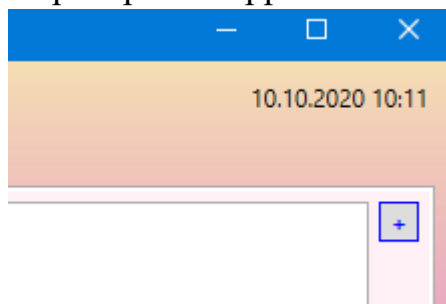
3.3. Объявим глобальные переменные, чтоб иметь доступ к ним из любого фрагмента кода

```
23 public partial class MainWindow : Window
24 {
25     string[] allfiles;
26     int countFiles = 0;
27     Ссылка: 0 | 0 изменений | 0 авторов, 0 изменений
28     public MainWindow()
29     {
30         InitializeComponent();
31     }
32 }
```




3.4. Сделаем отображение даты на label. Для этого пишем следующий код:

```
public MainWindow()
{
    InitializeComponent();
    label1.Content = DateTime.Now;
}
```

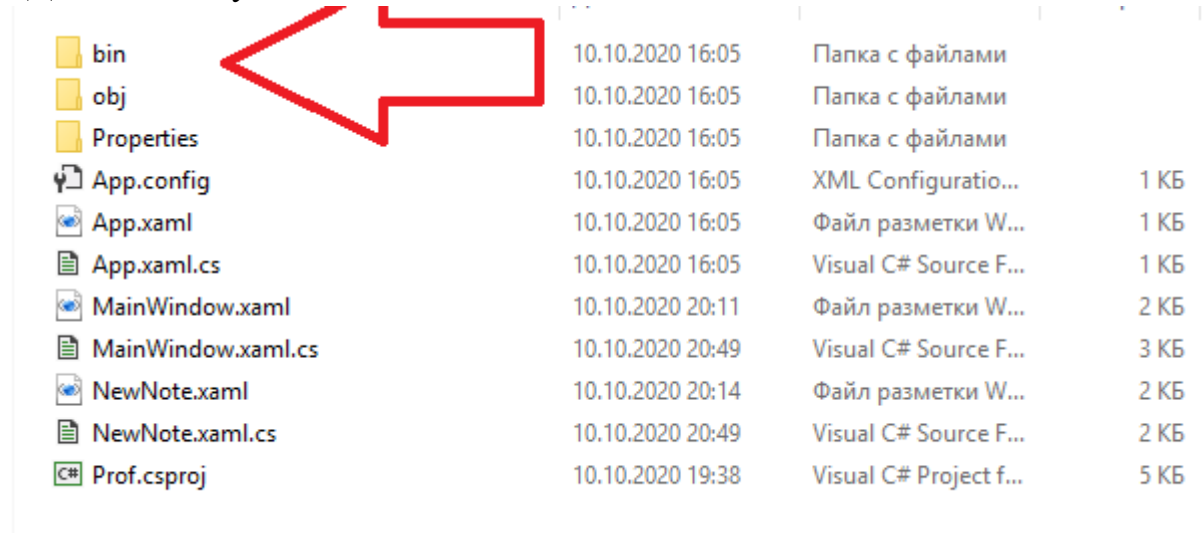
3.5. Проверяем корректное отображение. Должно появиться следующее:



3.6. Зайдем заранее в папку с проектом. Открываем папку соответствующую названию проекта

	.vs	10.10.2020 16:05	Папка с файлами
	Prof	10.10.2020 20:49	Папка с файлами
	Prof.sln	10.10.2020 16:05	Visual Studio Solu... 2 КБ

3.7. Далее в папку в bin

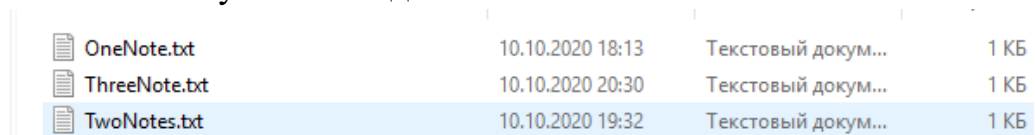


bin	10.10.2020 16:05	Папка с файлами	
obj	10.10.2020 16:05	Папка с файлами	
Properties	10.10.2020 16:05	Папка с файлами	
App.config	10.10.2020 16:05	XML Configuratio...	1 КБ
App.xaml	10.10.2020 16:05	Файл разметки W...	1 КБ
App.xaml.cs	10.10.2020 16:05	Visual C# Source F...	1 КБ
MainWindow.xaml	10.10.2020 20:11	Файл разметки W...	2 КБ
MainWindow.xaml.cs	10.10.2020 20:49	Visual C# Source F...	3 КБ
NewNote.xaml	10.10.2020 20:14	Файл разметки W...	2 КБ
NewNote.xaml.cs	10.10.2020 20:49	Visual C# Source F...	2 КБ
Prof.csproj	10.10.2020 19:38	Visual C# Project f...	5 КБ

3.8. В единственную там папку «Debug».

3.9. И в ней создадим папку «notes».

3.10. Создадим в ней несколько файлов формата «.txt», заполним их какими-то случайными данными

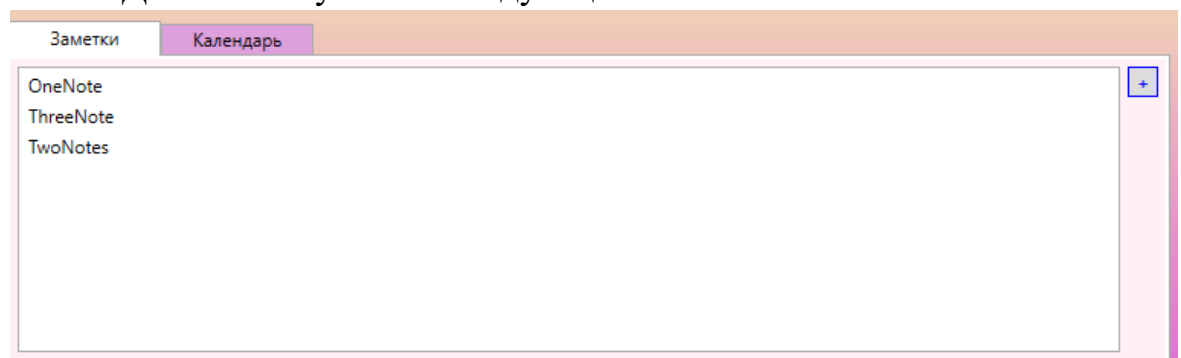


OneNote.txt	10.10.2020 18:13	Текстовый докум...	1 КБ
ThreeNote.txt	10.10.2020 20:30	Текстовый докум...	1 КБ
TwoNotes.txt	10.10.2020 19:32	Текстовый докум...	1 КБ

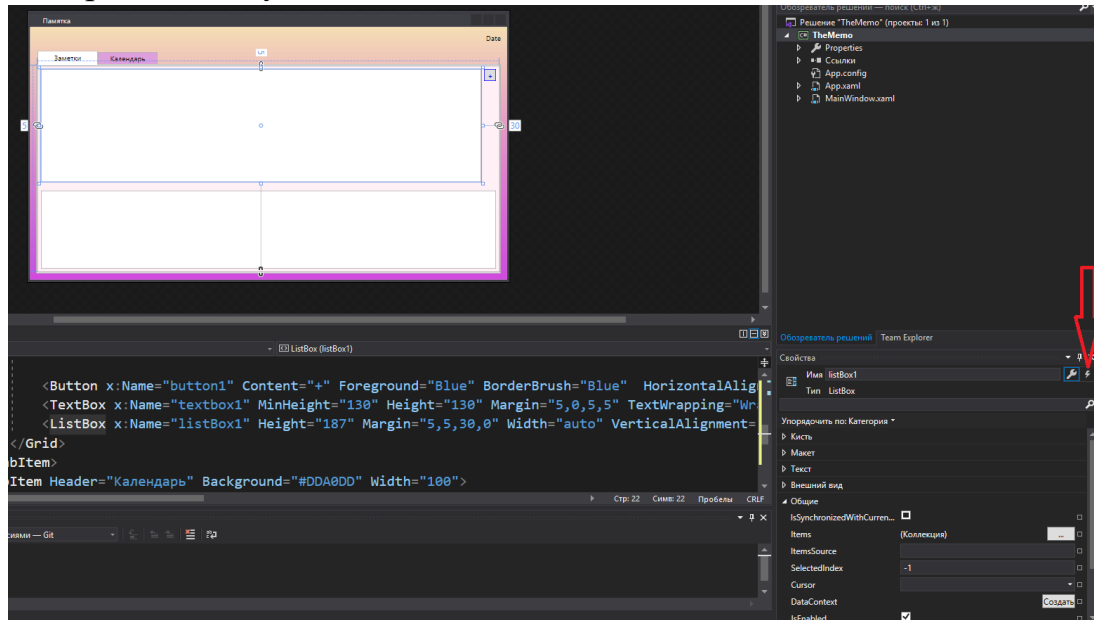
3.11. Напишем функцию для того, чтоб в ListBox происходило отображение всех имеющихся заметок

```
public void showAllNotes()
{
    allfiles = Directory.GetFiles(Environment.CurrentDirectory + "\\notes", "*.txt");
    foreach (string onlyName in allfiles)
    {
        allfiles[countFiles] = System.IO.Path.GetFileNameWithoutExtension(onlyName);
        listBox1.Items.Insert(countFiles, allfiles[countFiles]);
        countFiles++;
    }
    countFiles = 0;
}
```

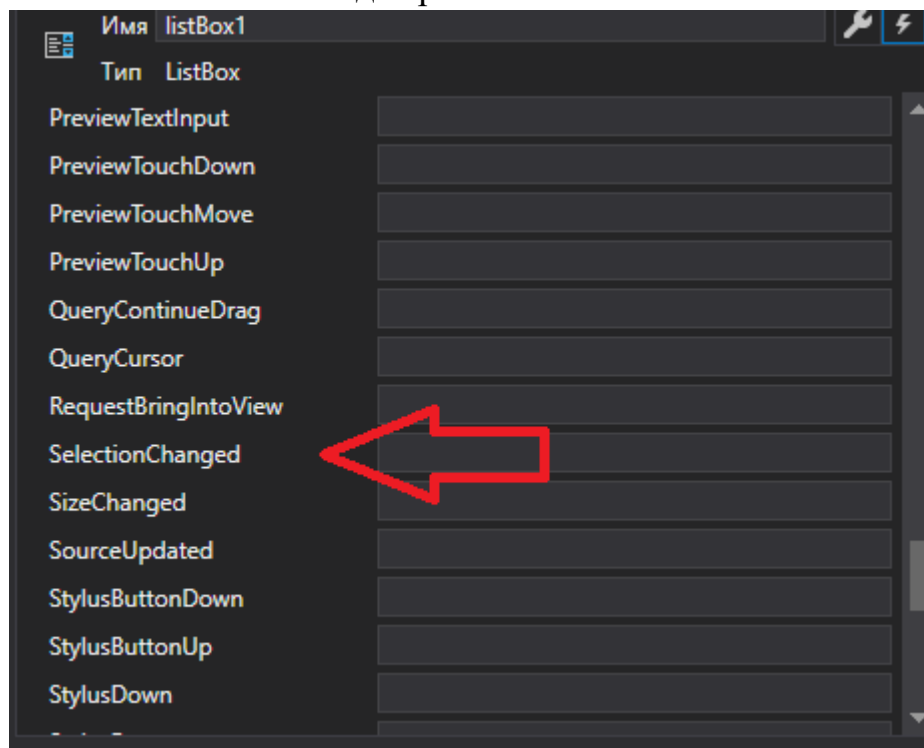
3.12. Должно получиться следующее:



- 3.13. Запрограммируем, чтоб в TextBox отображалось содержимое заметки, выбранного в ListBox. Для этого, сначала нажмем на ListBox, выберем вкладку события в лево



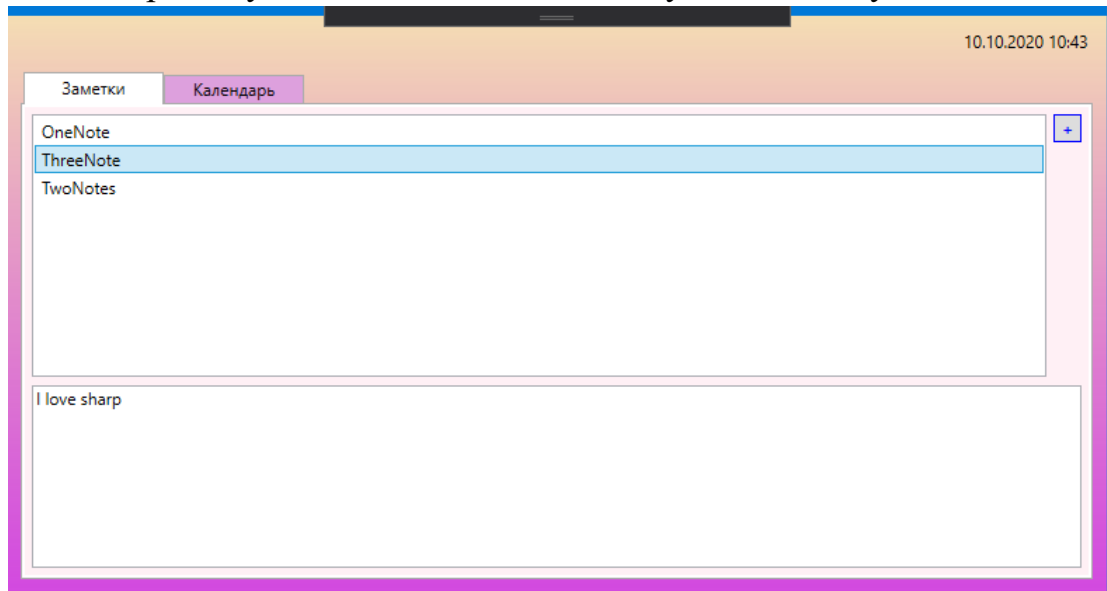
- 3.14. Находим событие «SelectionChanged» и в поле около него в жмём два раза ЛКМ.



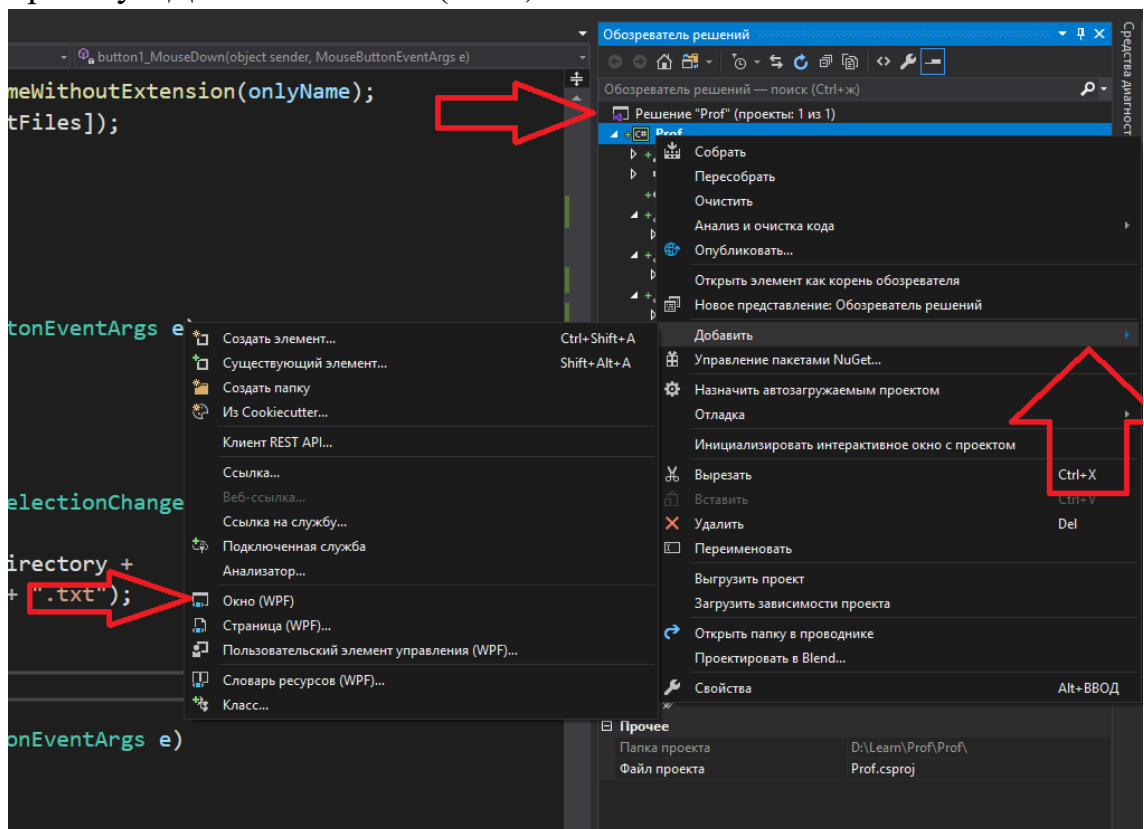
- 3.15. Фокус переходит в код, где создается функция события «private void listBox1_SelectionChanged». Пишем в ней следующий программный код

```
private void listBox1_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
    string data = File.ReadAllText(Environment.CurrentDirectory +
        "\\notes\\" + listBox1.SelectedItem.ToString() + ".txt");
    textBox1.Text = data;
}
```

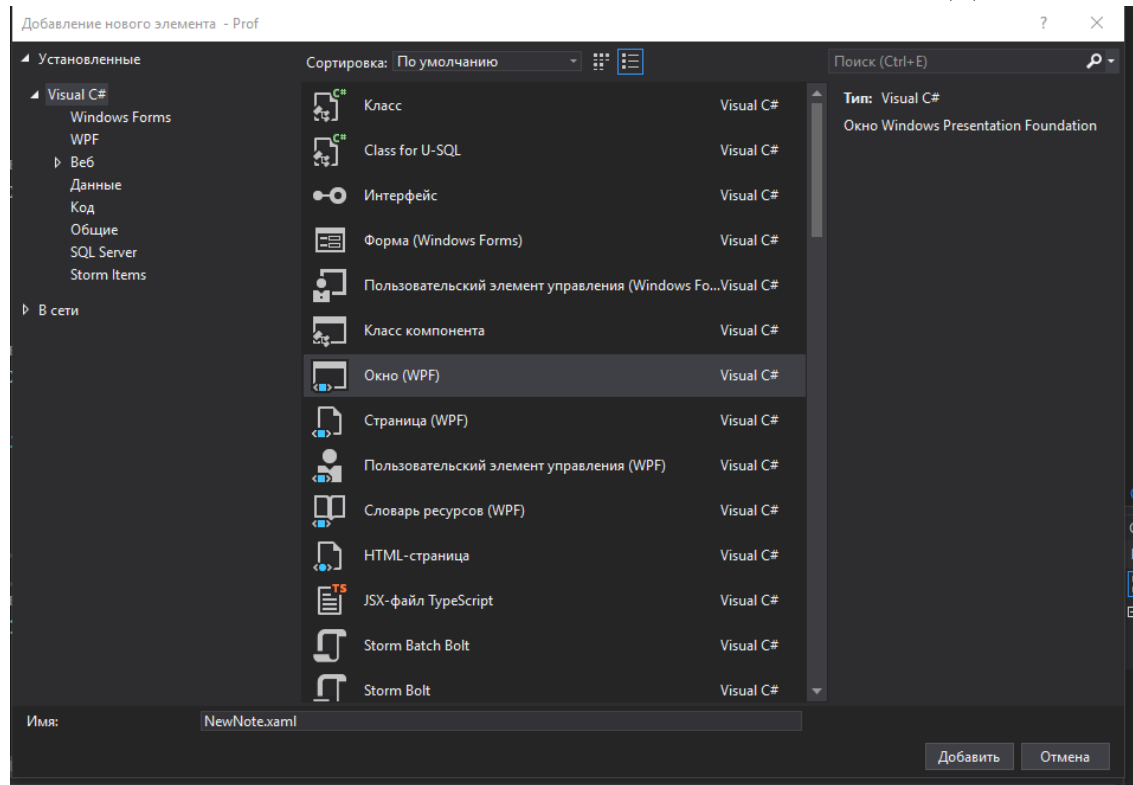
3.16. При запуске и выводе должно получиться следующее:



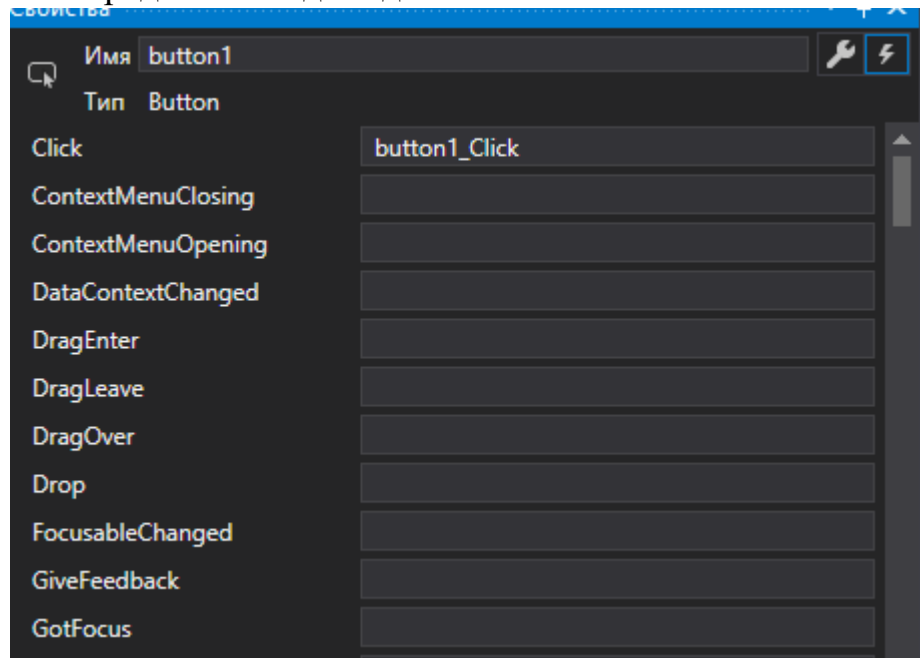
3.17. Програмируем кнопку для создания новой записи. Заранее создадим новую форму для создания новой заметки. Нажимаем ПКМ по проекту->Добавить->Окно(WPF)



3.18. В появившемся окне пишем название и нажимаем «Добавить».



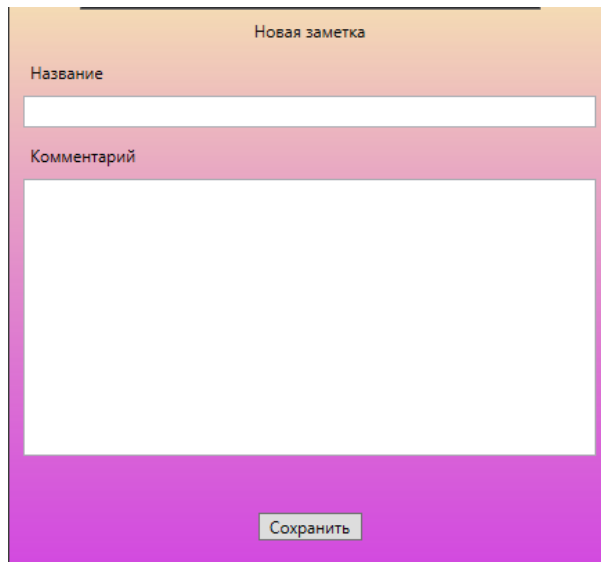
3.19. Выделяем кнопку, находим в событиях «Click», нажимаем по полю рядом ЛКМ дважды.



3.20. Пишем для новой функции следующий код:

```
private void button1_Click(object sender, RoutedEventArgs e)
{
    NewNote newNote = new NewNote(this);
    newNote.Show();
}
```

Теперь при нажатии на кнопку открывается дополнительное окно.



3.21. Запрограммируем этому окну следующий пользовательский интерфейс:

```
<Window x:Class="Prof.NewNote"
  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:Prof"
  mc:Ignorable="d"
  Title="NewNote" Height="450" Width="450">
  <Grid>
    <Grid.Background>
      <LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
        <GradientStop Color="#F5DEB3" Offset="0"/>
        <GradientStop Color="#CCC71CD8" Offset="1"/>
      </LinearGradientBrush>
    </Grid.Background>
    <Label x:Name="label2" Content="Название" HorizontalAlignment="Left"
      Margin="10,40,0,0" VerticalAlignment="Top"/>
    <Label x:Name="label3" Content="Комментарий" HorizontalAlignment="Left"
      Margin="10,100,0,0" VerticalAlignment="Top"/>
    <Label x:Name="label1" Content="Новая заметка" HorizontalAlignment="Center"
      Margin="0,10,0,0" VerticalAlignment="Top"/>
    <TextBox x:Name="textBox1" Height="23" Margin="10,70,10,0"
      TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="auto"/>
    <TextBox x:Name="TextBox2" MinHeight="200" Height="auto"
      Margin="10,130,10,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top"
      Width="auto"/>
    <Button Content="Сохранить" HorizontalAlignment="Center" Margin="0,0,0,20"
      VerticalAlignment="Bottom" Width="75" Click="Button_Click"/>
  </Grid>
</Window>
```

3.22. Запрограммируем на кнопку событие «Click». Код формы должен иметь следующий вид

```
public partial class NewNote : Window
{
    MainWindow mainWindow;
    ссылка: 1 | 0 изменений | 0 авторов, 0 изменений
    public NewNote(MainWindow lastWindow)
    {
        mainWindow = lastWindow;
        InitializeComponent();
    }

    ссылка: 1 | 0 изменений | 0 авторов, 0 изменений
    private void Button_Click(object sender, RoutedEventArgs e)
    {
        if (textBox1.Text.Length > 0 && TextBox2.Text.Length > 0)
        {
            if (File.Exists(Environment.CurrentDirectory + "\\notes\\" + textBox1.Text + ".txt"))
                File.Create(Environment.CurrentDirectory + "\\notes\\" + textBox1.Text + ".txt").Close();
            File.WriteAllText(Environment.CurrentDirectory + "\\notes\\" + textBox1.Text + ".txt", TextBox2.Text);
            mainWindow.listBox1.Items.Clear();
            mainWindow.showAllNotes();
            this.Close();
        }
        else
        {
            MessageBox.Show("Вы не указали одно или несколько из полей!\n Заполните все поля");
        }
    }
}
```

3.23. Проверяем работоспособность приложения и отсутствие ошибок