Практическая работа 6
Введение в XAML и WPF

Выполнила:
Студентка группы ПР-31
Савостенко Дарья

ОГЛАВЛЕНИЕ

ЦЕЛЬ РАБОТЫ

Цель работы познакомиться с графической система Windows Presentation Foundation (WPF), изучить язык разметки XAML, его основные элементы. Получить базовые знания о создании проектов в WPF, оформлении их в XAML.

ВЫПОЛНЕНИЕ ЗАДАНИЙ

Мне нужно создать проект в WPF. Для начала я открываю среду разработки Visual Studio. Нажимаю кнопку «Создать проект» (Рисунок 1 - Открытая Visual Studio). Ввожу в строку поиска «WPF» и нахожу нужный мне вид проекта (Рисунок 2 – Поиск проекта WPF). Даю проекту название, определяю ему расположение на компьютере и создаю проект (Рисунок 3 - Создание проекта).

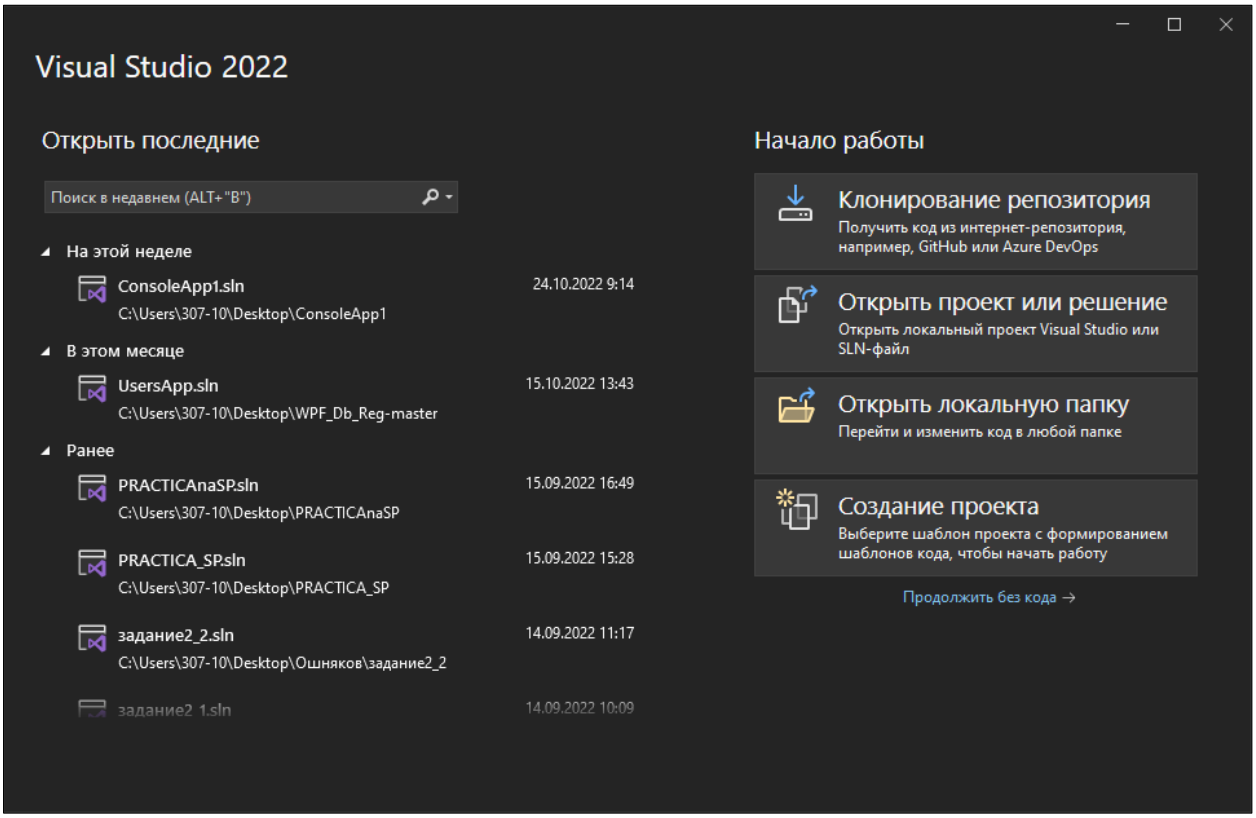

Рисунок 1 - Открытая Visual Studio

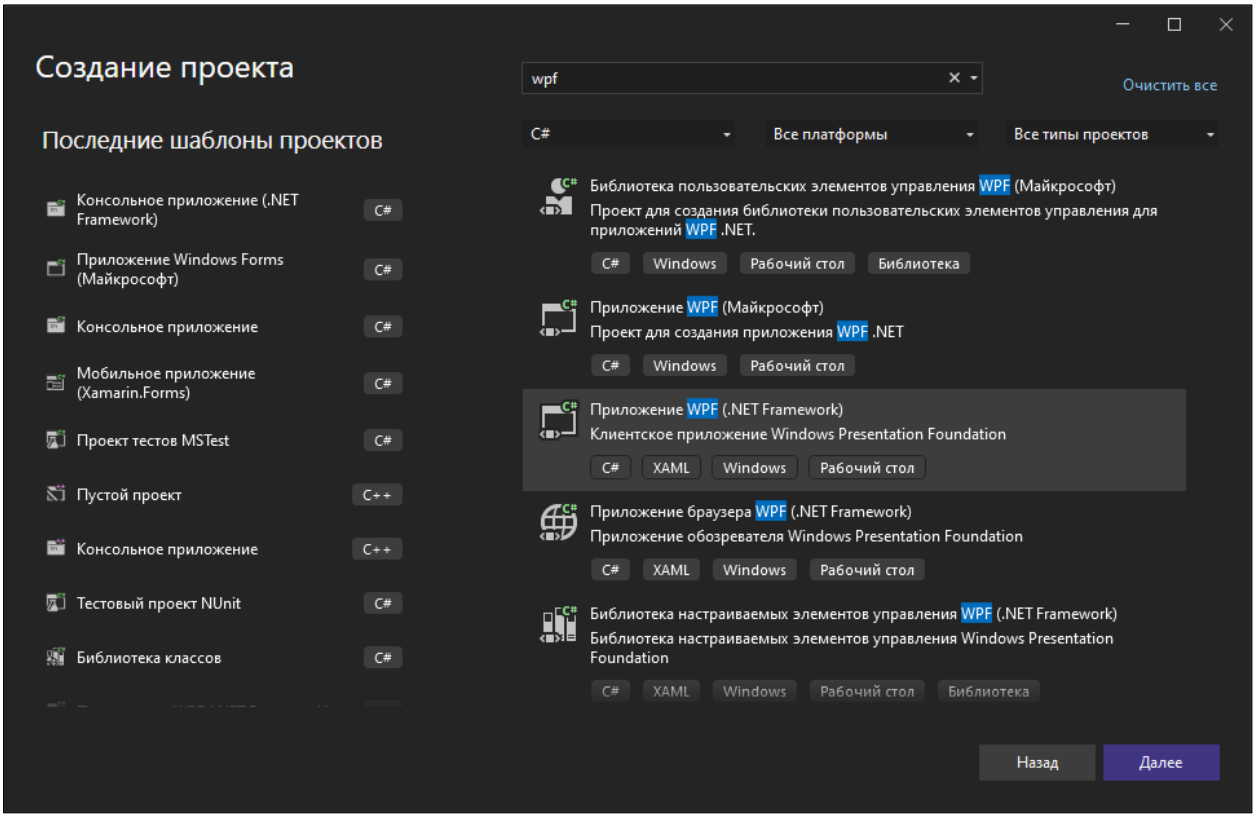

Рисунок 2 – Поиск проекта WPF

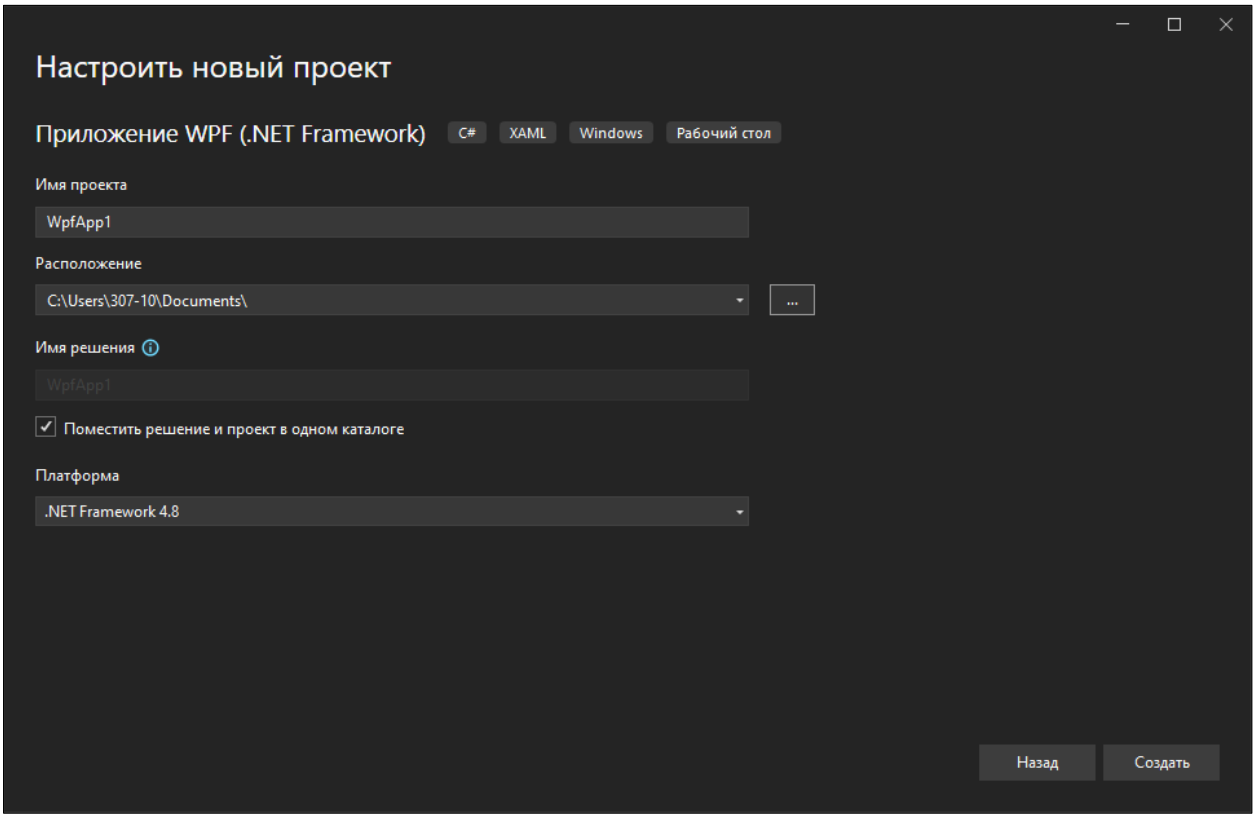

Рисунок 3 - Создание проекта

В итоге создаётся проект, где есть конструктор окна (Рисунок *4* - Конструктор окна) и код XAML (рис).

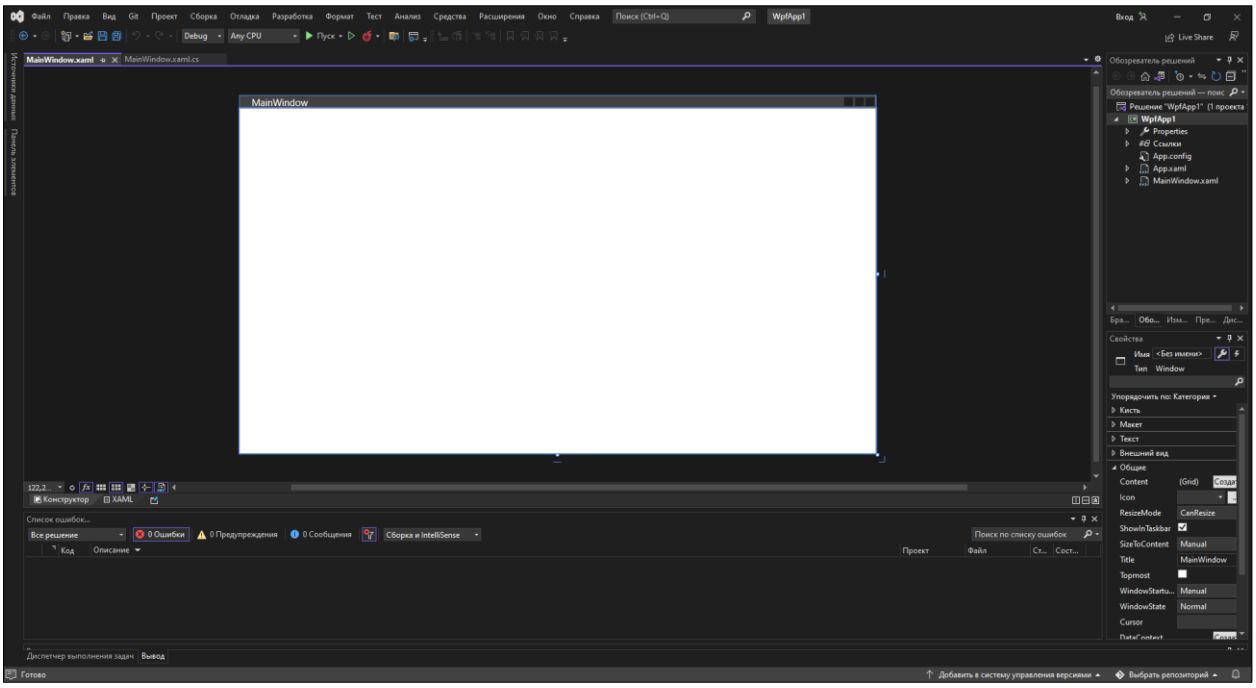

Рисунок 4 - Конструктор окна

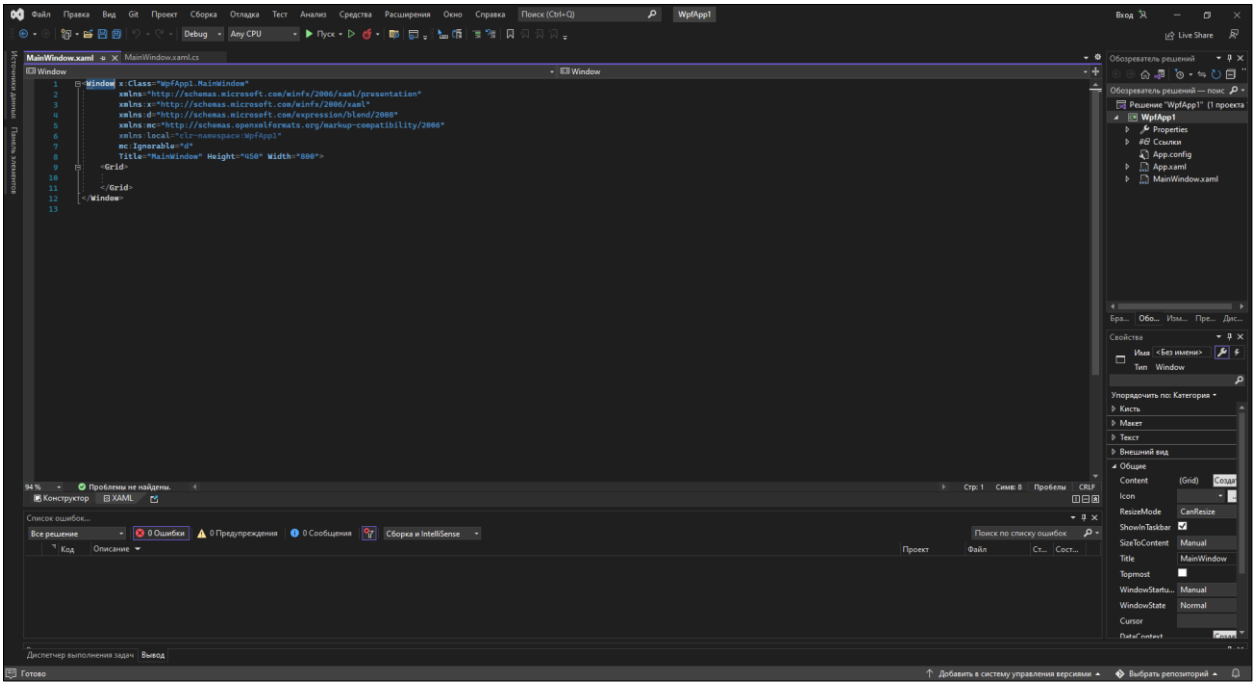

Рисунок 5 - код XAML

Мне нужно сделать кнопку, размещённую по центру. Для этого я создаю элемент <Button> со свойствами HorizontalAlignment="Center" и VerticalAlignment="Center". Также я задаю ширину Width равную 150 и высоту

Height равную 30. Текст в кнопку задам через Content=”Обычная кнопка”, размер шрифта FontSize будет “17”. В итоге код на XAML выглядит так – Рисунок 6 - Код кнопки, а сама кнопка выглядит так – Рисунок 7 - кнопка на конструкторе.

```
<Window x:Class="WpfApp1.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:WpfApp1"
        mc:Ignorable="d"
        Title="MainWindow" Height="450" Width="800">
    <Grid>
        <Button x:Name="Btn1"
        HorizontalAlignment="Center"
        VerticalAlignment="Center"
        Width="150"
        Height="30"
        FontSize="17"
        Content="Обычная кнопка"
        Foreground="#006699"
        BorderBrush="#303030"/>

    </Grid>

</Window>
```

Рисунок 6 - Код кнопки

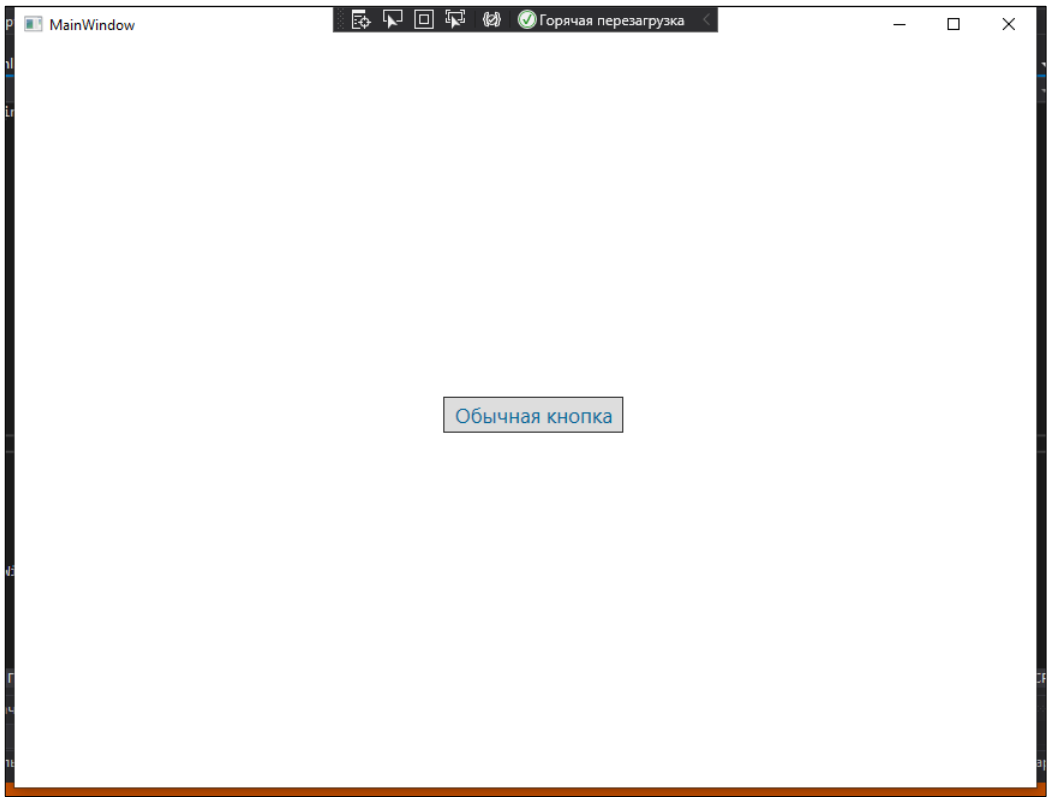

Рисунок 7 - кнопка на конструкторе

Так как свойства кнопки HorizontalAlignment и VerticalAlignment имеют значение "Center", кнопка остаётся в центре окна даже если изменить его размер и форму (Рисунок 8 - кнопка в центре).

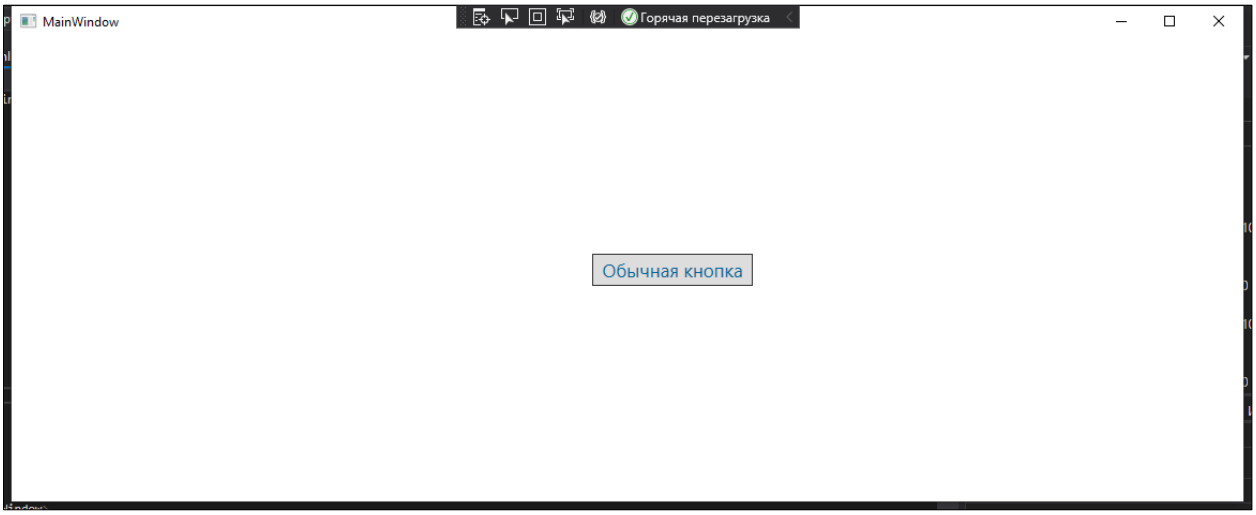

Рисунок 8 - кнопка в центре

Также мне нужно задать градиентный фон кнопке. Для этого я изменяю синтаксис элемента кнопки с <Button/> на <Button></Button>. Теперь в кнопку можно вставлять другие элементы и задавать значения свойствам с помощью деревьев элементов. Так я задаю значение фона через элемент <Button.Background>, в котором прописываю элементы <LinearGradientBrush> и <LinearGradientBrush.GradientStops>, где указываю элементы <GradientStop> со своёствами Color="LightBlue" и "DarkBlue" и Offset="0" и “1”. В итоге кнопка будет переливаться со светло-голубого на тёмно-голубой. Код на XAML будет выглядеть так – Рисунок 9 - код градиента, а сама кнопка в конструкторе вот так – Рисунок 10 - кнопка с градиентом.

```
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="116*"/>
        <RowDefinition Height="303*"/>
    </Grid.RowDefinitions>
    <Button x:Name="Btn1"
    HorizontalAlignment="Center"
    VerticalAlignment="Center"
    Width="150"
    Height="30"
    FontSize="17"
    Content="Обычная кнопка"
    Foreground="#006699"
    BorderBrush="#303030" Margin="321,78,321,195" Grid.Row="1">

        <Button.Background>
            <LinearGradientBrush>
                <LinearGradientBrush.GradientStops>
                    <GradientStop Color="LightBlue" Offset="0" />
                    <GradientStop Color="DarkBlue" Offset="1" />
                </LinearGradientBrush.GradientStops>
            </LinearGradientBrush>
        </Button.Background>

    </Button>

</Grid>
```

Рисунок 9 - код градиента

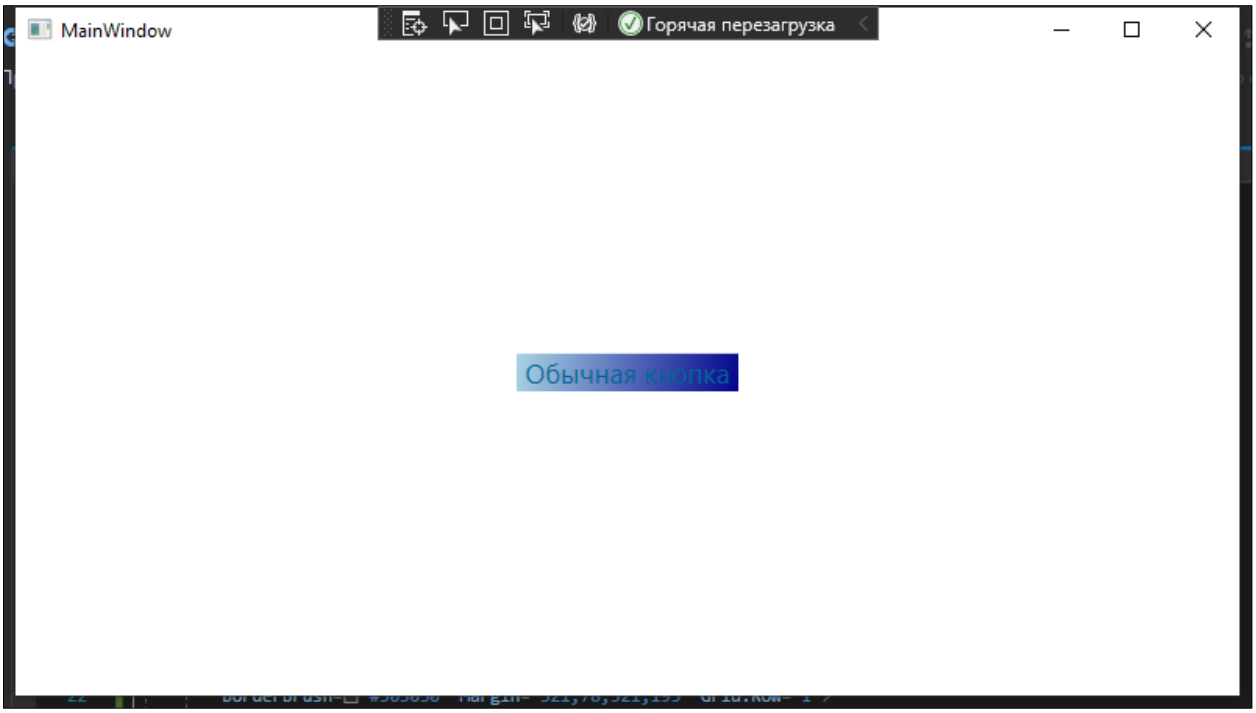

Рисунок 10 - кнопка с градиентом

Теперь мне нужно изменить цвет окна на сине-голубой градиент. Для этого я создаю прямоугольник <Rectangle>, задаю ему элемент

<Rectangle.Fill>, в котором задаю уже элемент <LinearGradientBrush>. В заголовке элемента <LinearGradientBrush> изменяю значения элементов StartPoint и EndPoint с “0, 0” и “1, 1” на “0.5, 0” и “0.5, 1”. Затем я создаю четыре элемента <GradientStop> со свойством <Offset> “0.0”, “0.2”, “0.8” “1” соответственно. Первый и последний будут иметь цвет DarkBlue, а два средних – LightBlue. В итоге получается вот такой код XAML – Рисунок 11 - код градиентного окна, и вот такое градиентное окно – Рисунок 12 - градиентное окно.

```
<Rectangle>

    <Rectangle.Fill>

        <LinearGradientBrush StartPoint="0.5, 0" EndPoint="0.5, 1">

            <GradientStop Color="DarkBlue" Offset="0.0"/>
            <GradientStop Color="LightBlue" Offset="0.2"/>
            <GradientStop Color="LightBlue" Offset="0.8"/>
            <GradientStop Color="DarkBlue" Offset="1"/>

        </LinearGradientBrush>

    </Rectangle.Fill>

</Rectangle>
```

Рисунок 11 - код градиентного окна

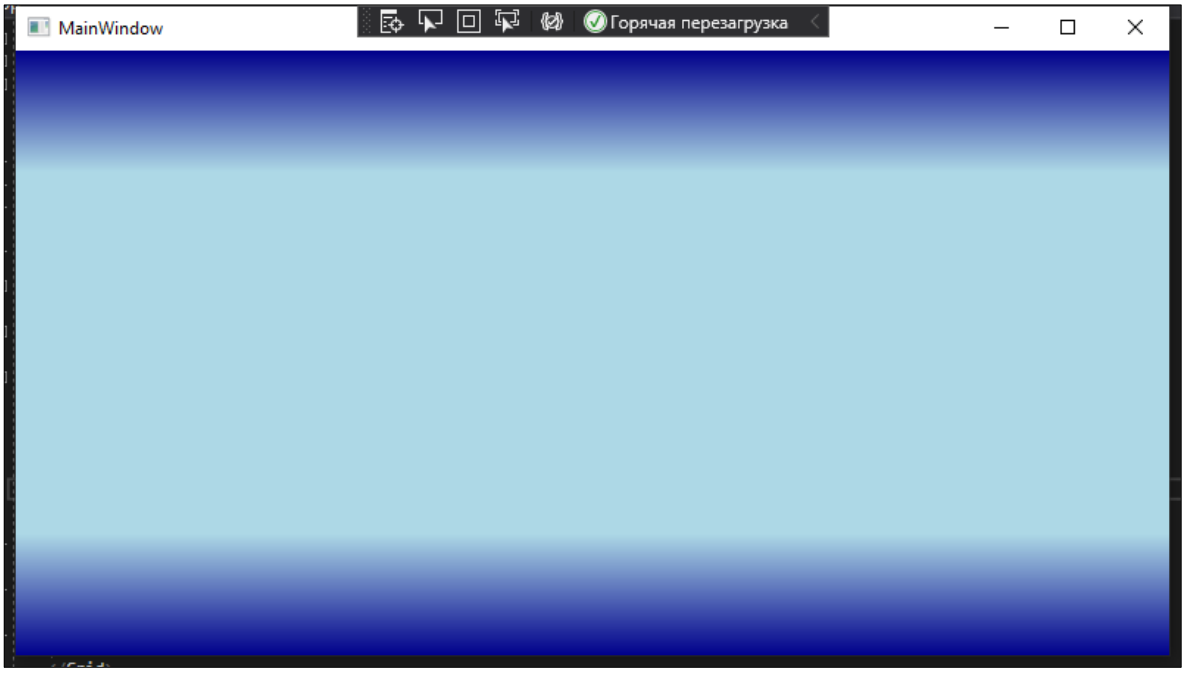

Рисунок 12 - градиентное окно

ВЫВОД

По результату работы я познакомилась с графической система Windows Presentation Foundation (WPF), изучила язык разметки XAML и его основные элементы. Получила базовые знания о создании проектов в WPF и оформлении их в XAML.