

В.В. Подбельский

Использованы иллюстрации пособия Daniel Solis, Illustrated C#

Иллюстрации к курсу лекций по дисциплине «Программирование на C#»

12. Часть 1

Знакомство с WPF

Виды WPF-Приложений

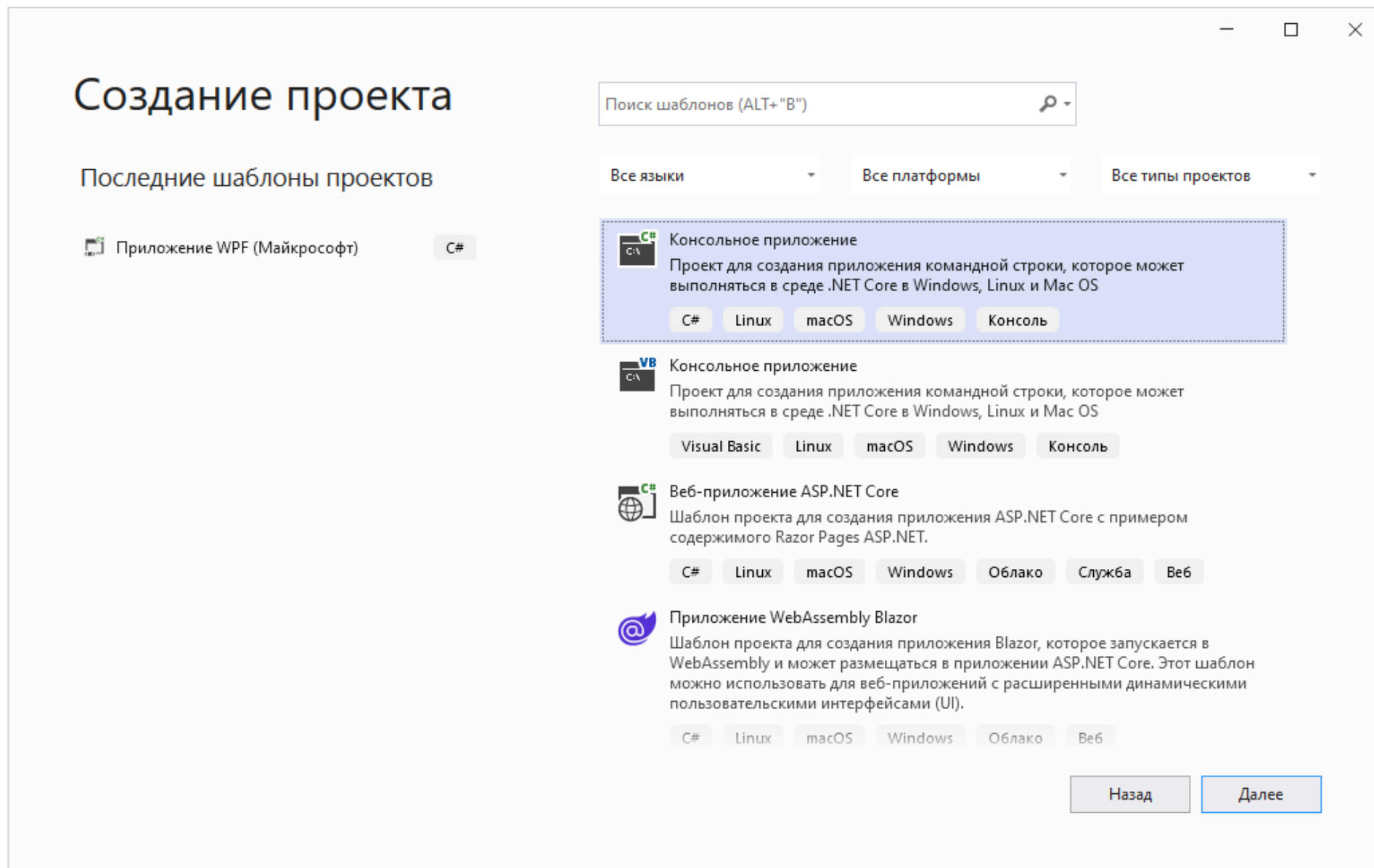
Актуально:

- Windows Desktop Apps;
- Xamarin Apps

Устарело:

- Windows Store Apps;
- XBAP – XAML Browser Applications (Interactive web applications).
- Silverlight.

Консольный Проект с WPF в Visual Studio-1



Консольный Проект с WPF в Visual Studio-2

Настроить новый проект

Консольное приложение C# Linux macOS Windows Консоль

Имя проекта

WPFConsoleDemoApp

Расположение

C:\Users\Danii\source\repos

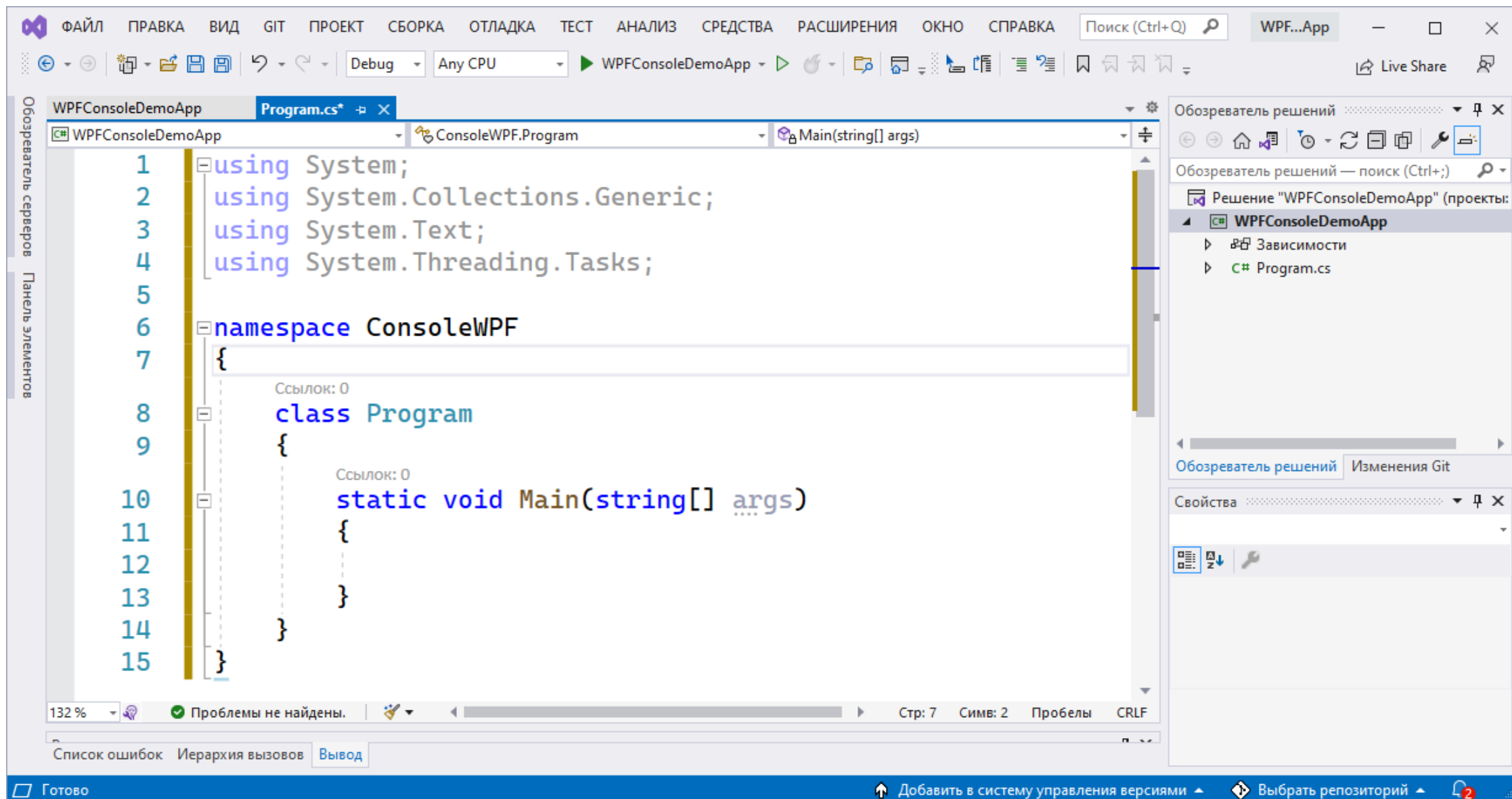
Имя решения ⓘ

WPFConsoleDemoApp

☐ Поместить решение и проект в одном каталоге

Назад Далее

Консольный Проект с WPF в Visual Studio-3



Переключение на WPF-Проект

Вручную отредактируйте файл *.csproj:

- измените OutputType на WinExe
- добавьте элемент UseWPF со значением true;
- измените целевую библиотеку (TargetFramework) на net5.0-windows.

```
<Project Sdk="Microsoft.NET.Sdk">  
  <PropertyGroup>
```

```
    <OutputType>WinExe</OutputType>
```

```
    <TargetFramework>net5.0-windows</TargetFramework>
```

```
    <ApplicationIcon />
```

```
    <StartupObject>WPF_00_Console.Program</StartupObject>
```

```
    <UseWPF>true</UseWPF>
```

```
  </PropertyGroup>  
</Project>
```

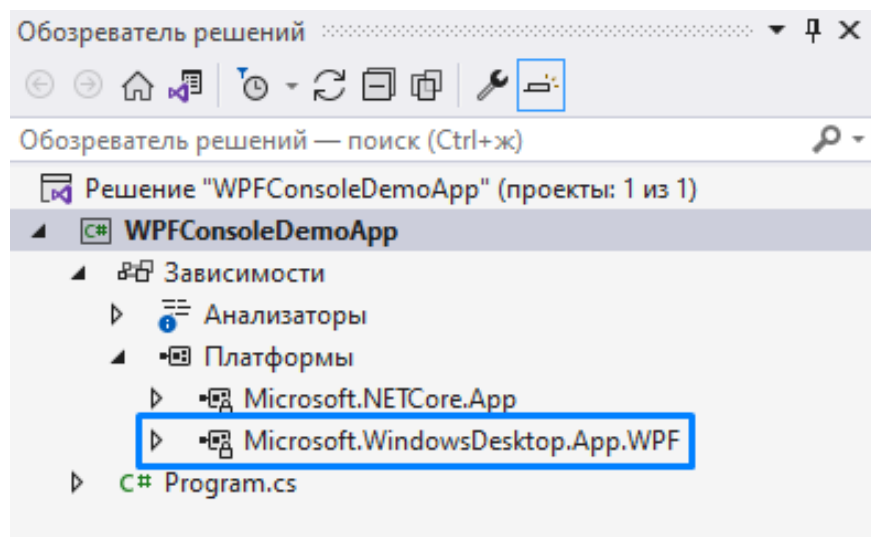
Основные Пространства Имян и Сборки WPF

Для использования WPF Вам достаточно **2 пространств имён**:

- `using System;`
- `using System.Windows.`

Сборки явным образом добавлять не нужно – они варьируются в зависимости от выбранной целевой ОС (см. предыдущий слайд)

Пример: Windows

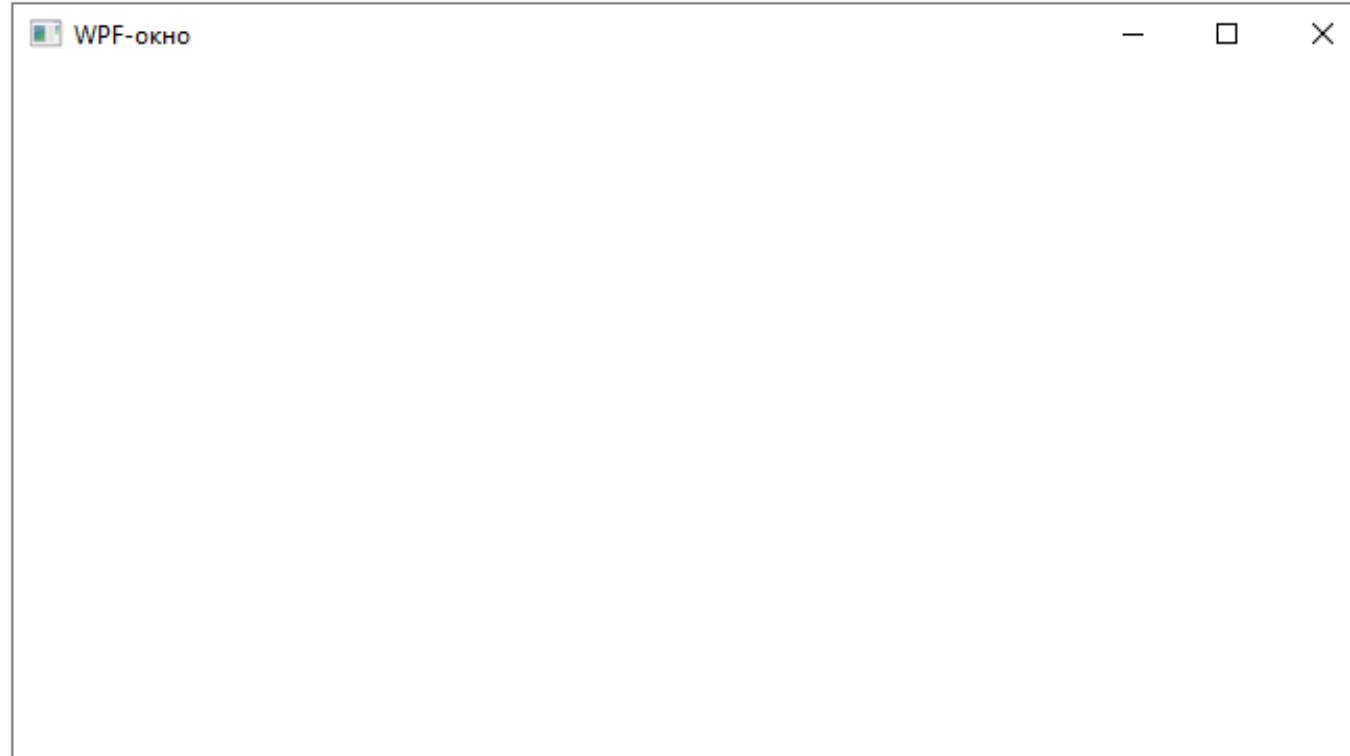


Обновлённый Код Program.cs

```
using System;
using System.Windows;
using System.Windows.Media;

namespace ConsoleWPF {
    class Program {
        [STAThread]
        static void Main() {
            Window myWindow = new Window();    // Объект окна.
            myWindow.Title = "WPF-окно";
            Application app = new Application(); // Объект приложения.
            app.Run(myWindow);
        }
    }
}
```


Результат Выполнения (Windows)



На заметку: аналогично, для использования `Windows.Forms` можно добавить пространство имён `System.Windows.Forms`

Пример Наполнения Окна WPF

```
Window myWindow = new Window();  
myWindow.Title = "WPF-окно";  
myWindow.FontFamily = new FontFamily("Verdana");  
myWindow.FontSize = 20;  
myWindow.SizeToContent = SizeToContent.WidthAndHeight;  
myWindow.BorderThickness = new Thickness(10, 10, 10, 10);  
myWindow.BorderBrush = Brushes.Beige;  
myWindow.Content = @"
```

Меня учили многие-
И добрые, и строгие,
Я строгих не любил.


...

Меня учили многие
Но выучили строгие.
И вышло, что в итоге я
Всех добрых позабыл.

...

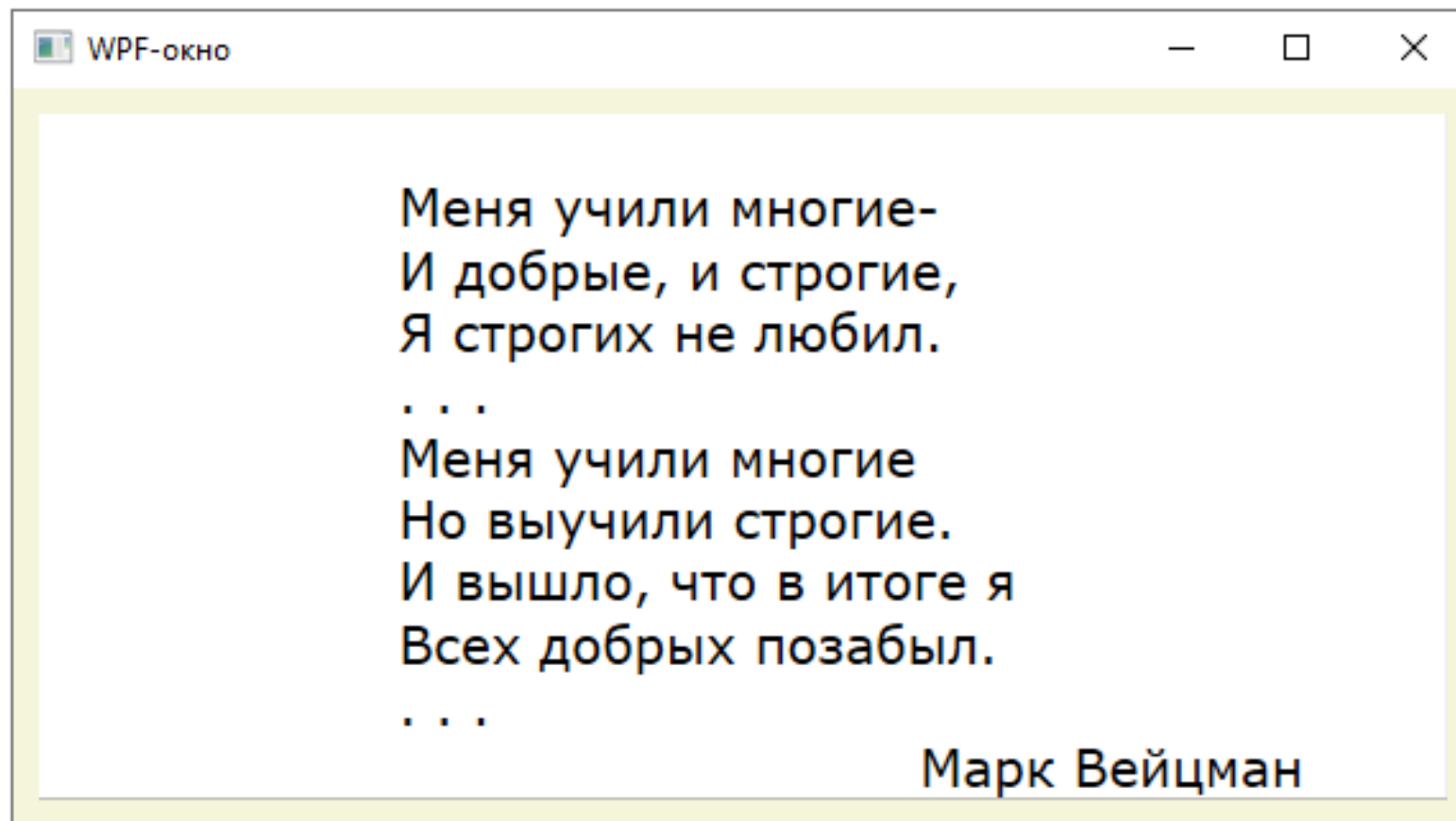
Марк Вейцман";

```
Application app = new Application();  
app.Run(myWindow);
```

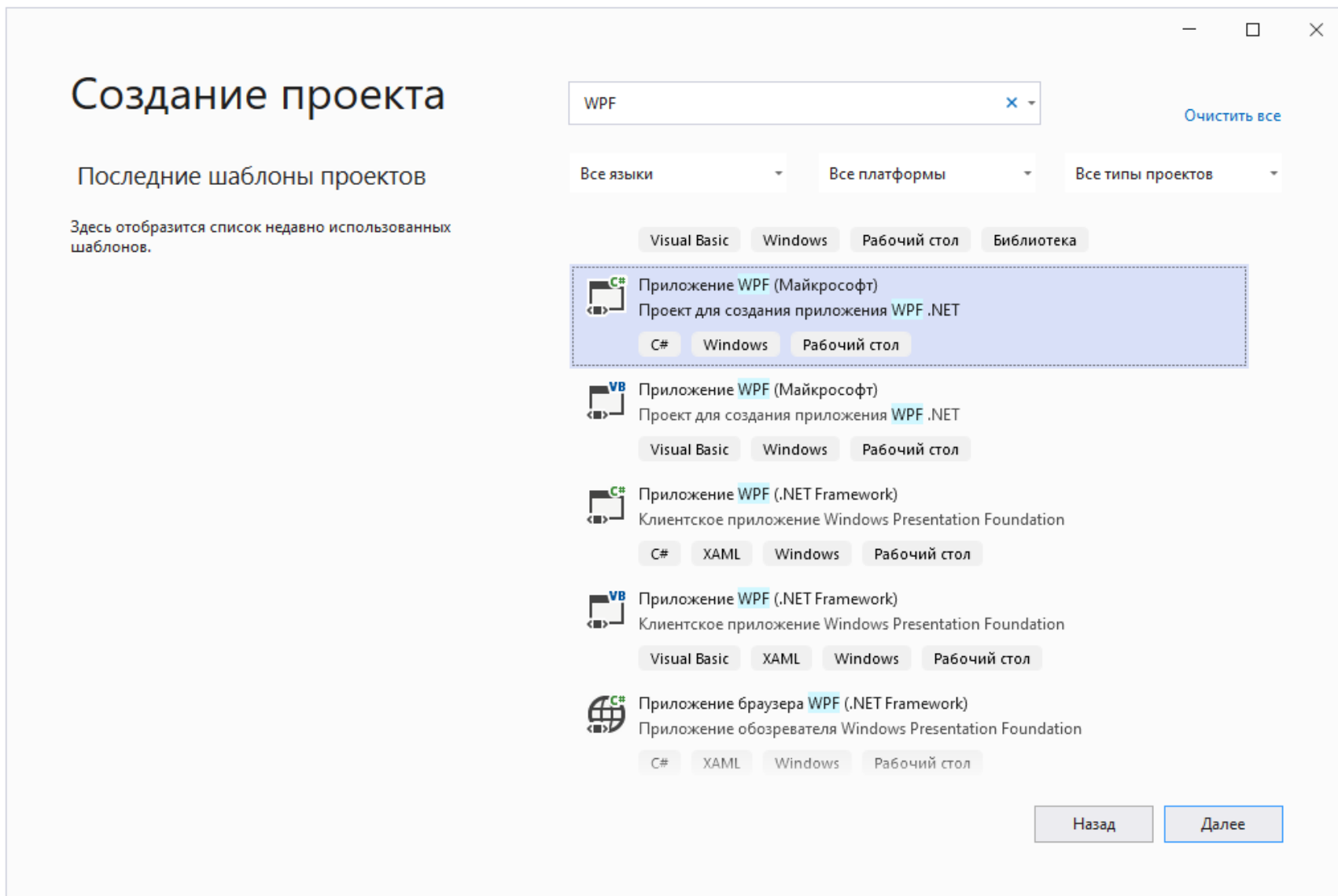


Попробуйте добавить данный
код внутрь Main.

Результат Выполнения



Создание WPF-Проекта в Visual Studio-1



Создание WPF-Проекта в Visual Studio-2

Настроить новый проект

Приложение WPF (Майкрософт) C# Windows Рабочий стол

Имя проекта

WPFDemoApp

Расположение

C:\Users\Danii\source\repos

Имя решения ⓘ

WPFDemoApp

☐ Поместить решение и проект в одном каталоге

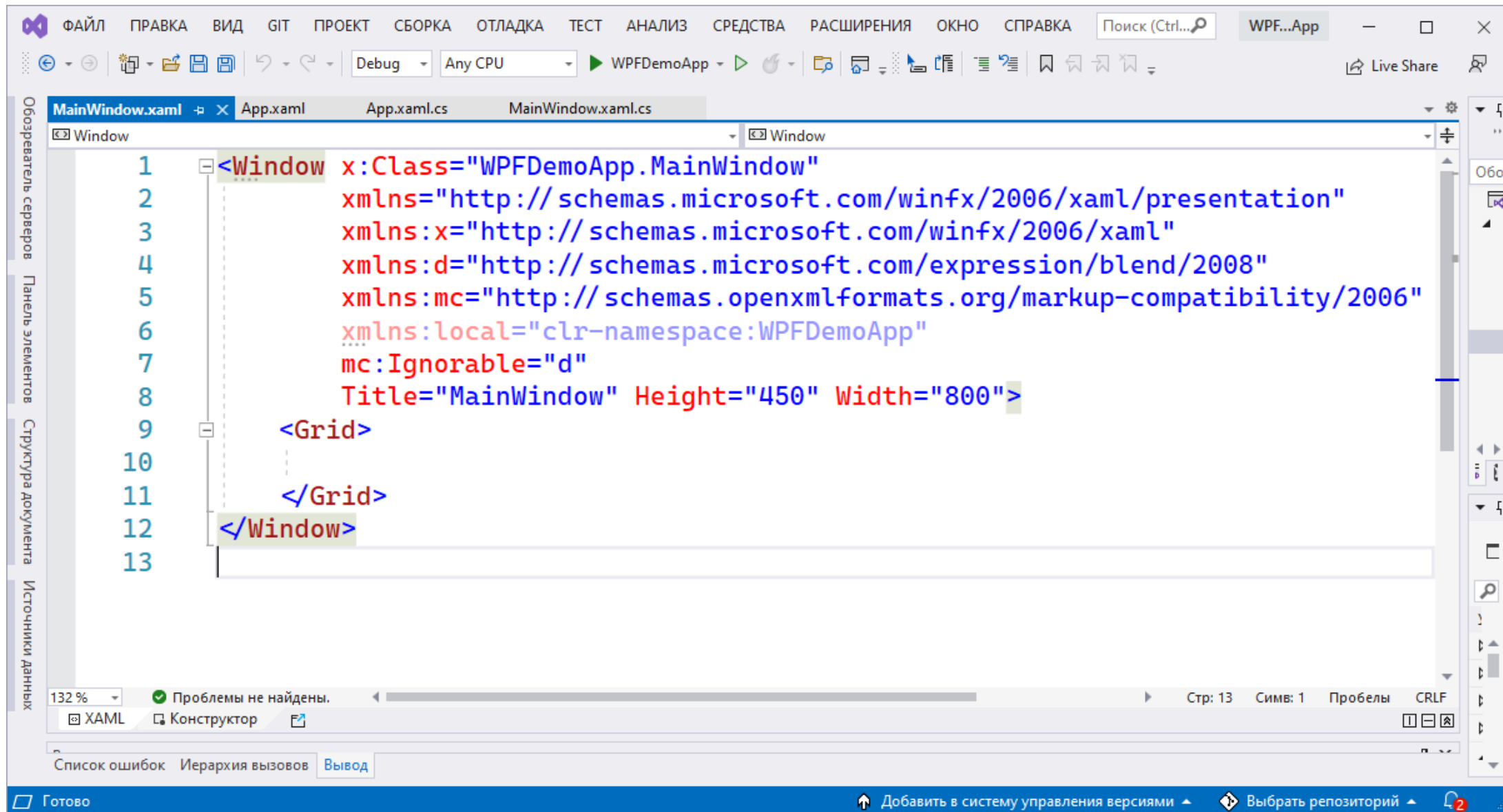
Назад Далее

Файлы WPF-Проекта в Visual Studio

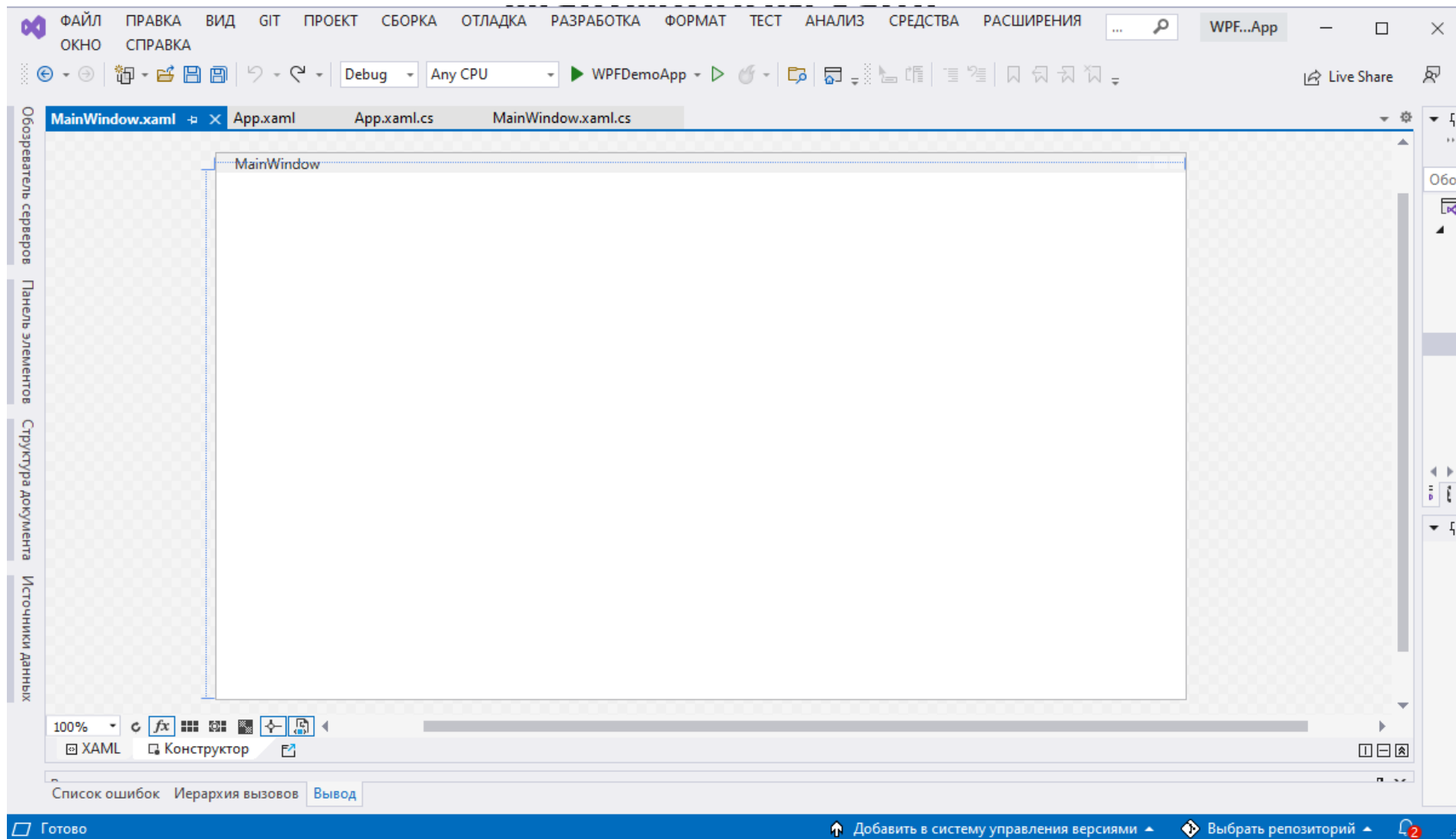
По умолчанию WPF-приложение состоит из нескольких основных сущностей:

- 1) MainWindow.xaml
- 2) MainWindow.xaml.cs
- 3) App.xaml
- 4) App.xaml.cs
- 5) AssemblyInfo.cs

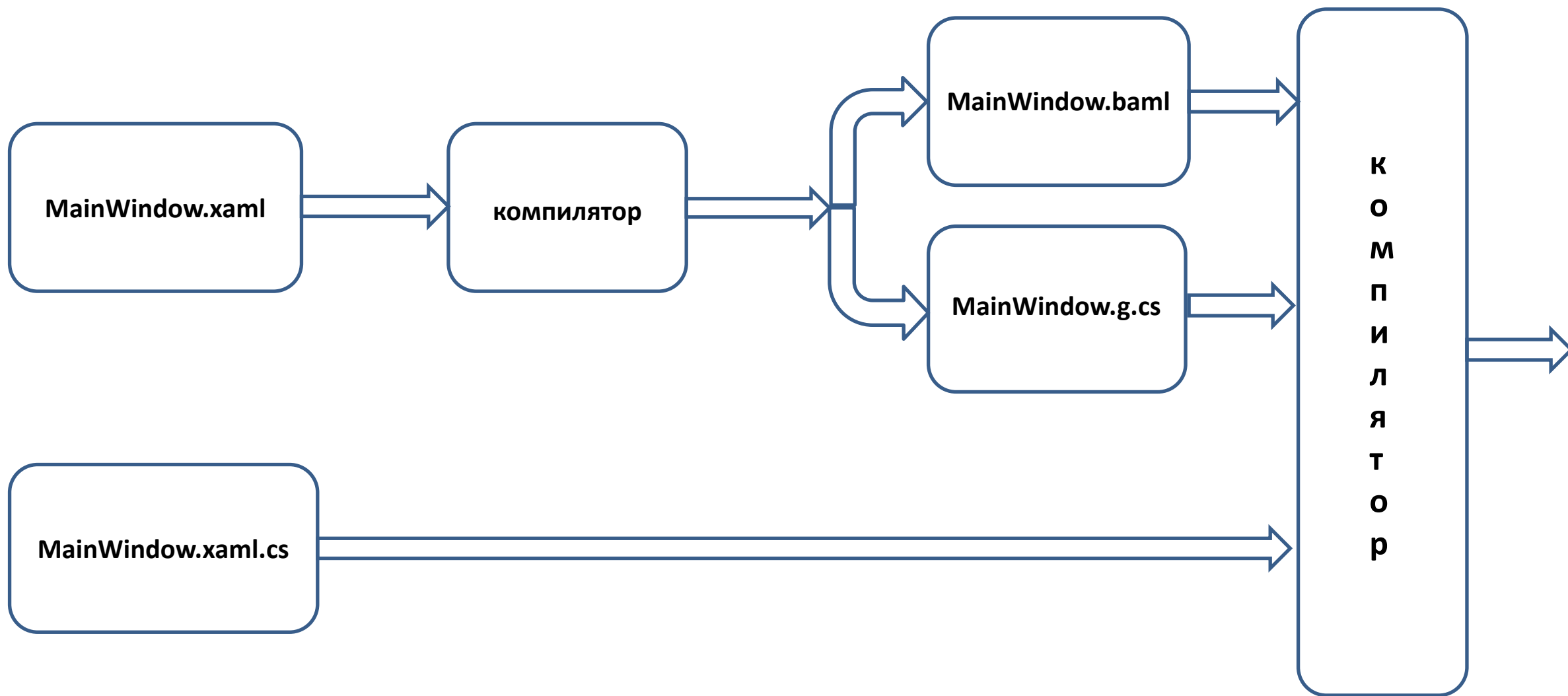
Окно Редактора XAML



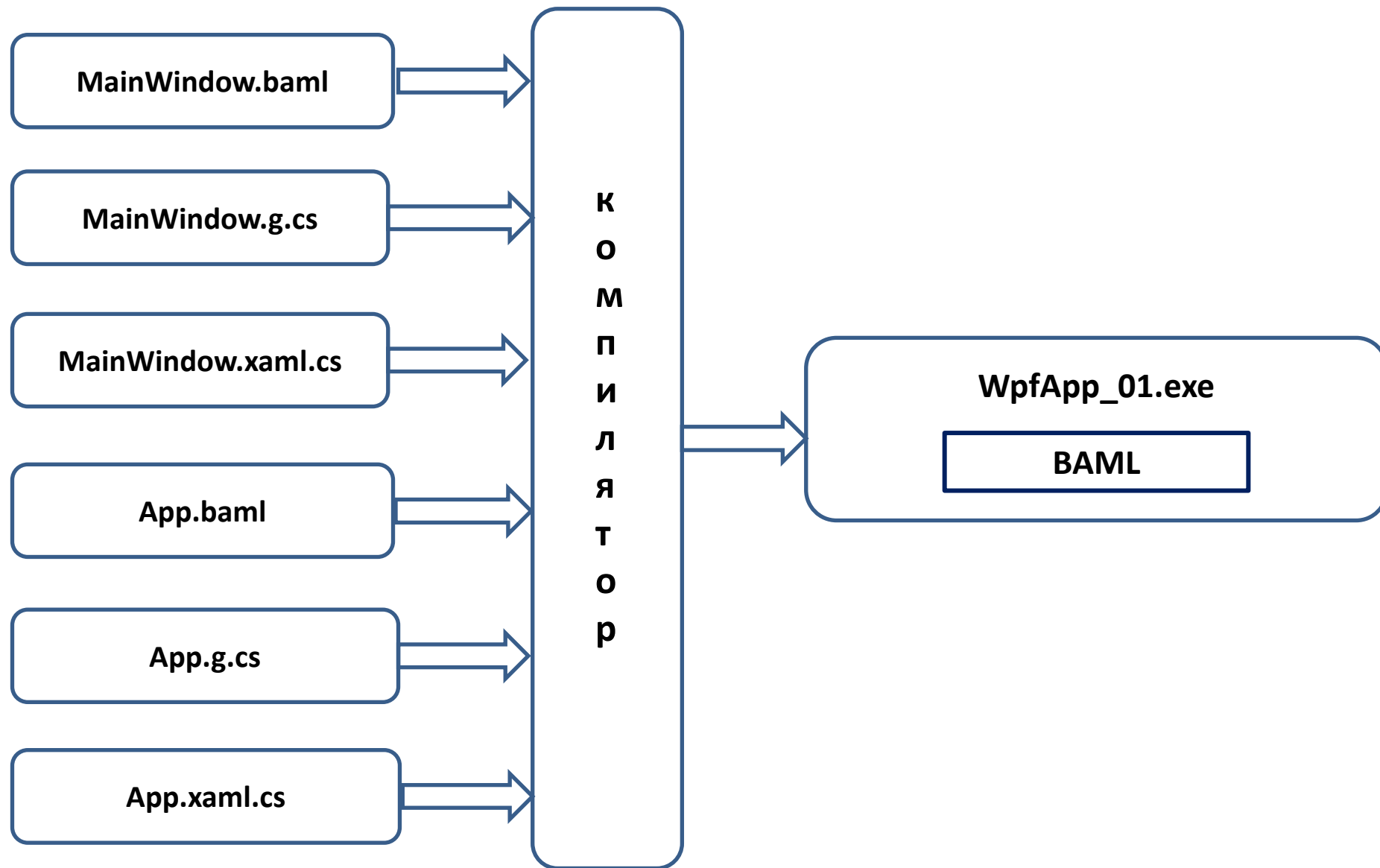
Визуальное Представление MainWindow.xaml



Компиляция WPF-Приложения: Этап 1



Компиляция WPF-Приложения: Этап 2



Пользовательский Интерфейс WPFDemoApp



Пространства имен в MainWindow.xaml.cs

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows; // WPF  
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;
```

Код из Файла MainWindow.xaml.cs

```
namespace WPFDemoApp
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
    }
}
```

Разметка Файла App.xaml

```
<Application x:Class="WPFDemoApp.App"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="clr-namespace:WPFDemoApp"
    StartupUri="MainWindow.xaml">
    <Application.Resources>

    </Application.Resources>
</Application>
```

Код из Файла App.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Threading.Tasks;
using System.Windows;

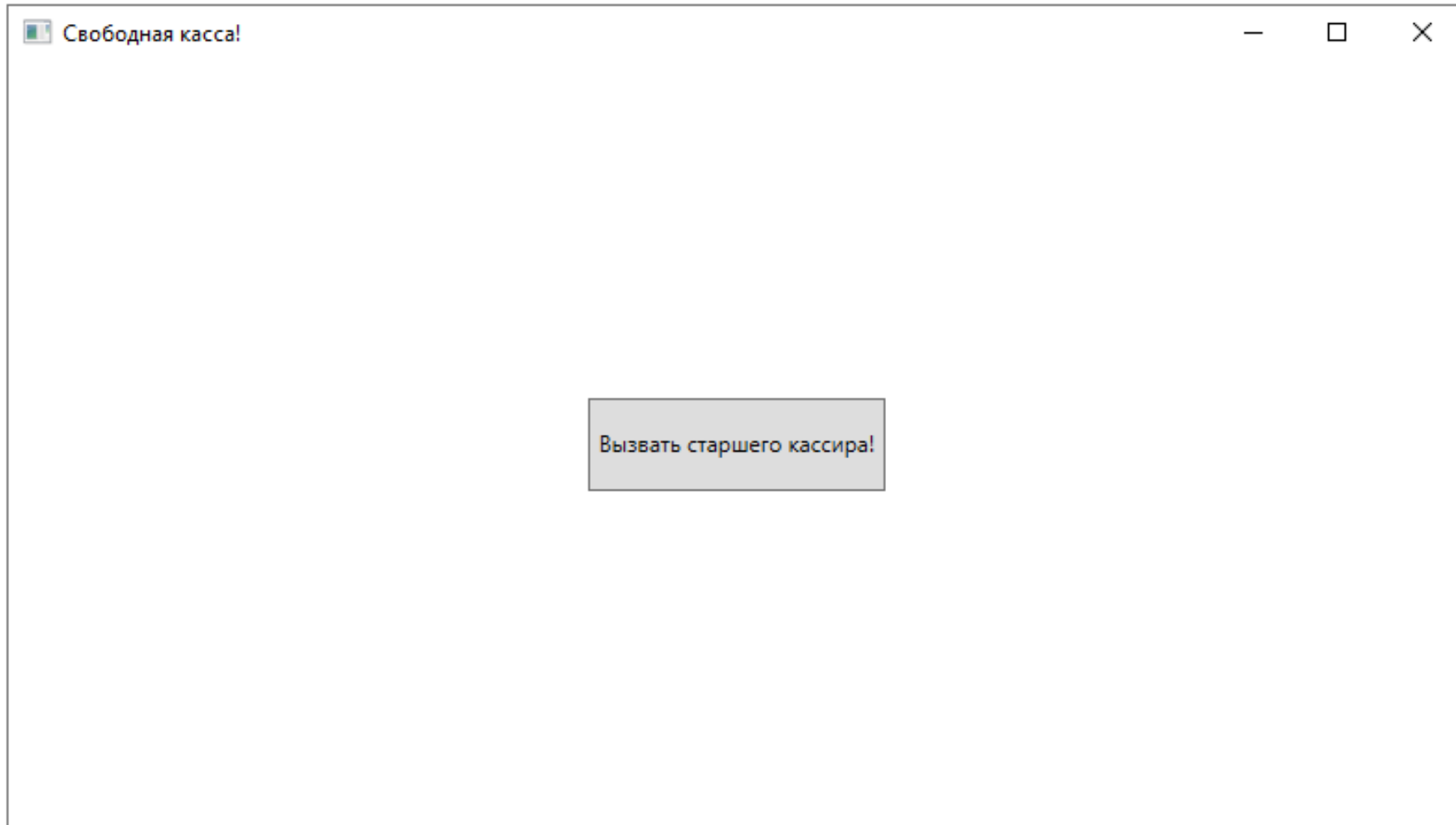
namespace WPFDemoApp
{
    /// <summary>
    /// Interaction logic for App.xaml
    /// </summary>
    public partial class App : Application
    {
    }
}
```

Добавление Кнопки в MainWindow.xaml.cs

```
using System.Windows;
using System.Windows.Controls;

namespace WPFDemoApp {
    public partial class MainWindow : Window {
        private Button button;    // Ссылка на объект класса Button.
        public MainWindow() {
            InitializeComponent();
            Title = "Свободная касса!";
            button = new Button();
            button.Content = "Вызвать старшего кассира!";
            button.Width = 160;
            button.Height = 50;
            button.Margin = new Thickness(10);
            Content = button;    // "Размещаем" кнопку в окне.
        }
    }
}
```


Полученное Окно с Кнопкой-Пустышкой



Позиционирование WPF-Элементов

Перечисления:

- `enum HorizontalAlignment {Center, Left, Right, Stretch}`
- `enum VerticalAlignment {Bottom, Center, Stretch, Top}`

Свойства:

- `HorizontalAlignment HorizontalAlignment { get; set; }`
- `VerticalAlignment VerticalAlignment { get; set; }`

Добавление Поведения Кнопке

```
private int state = 0; // Номер позиции кнопки.  
void ChangeButtonPos()  
{ // Метод изменения позиции кнопки между углами экрана по нажатию.  
    switch (state) {  
        case 0:  
            button.HorizontalAlignment = HorizontalAlignment.Right;  
            button.VerticalAlignment = VerticalAlignment.Top;  
            break;  
        case 1:  
            button.VerticalAlignment = VerticalAlignment.Bottom;  
            break;  
        case 2:  
            button.HorizontalAlignment = HorizontalAlignment.Left;  
            break;  
        case 3:  
            button.VerticalAlignment = VerticalAlignment.Top;  
            break;  
    }  
    state = (state + 1) % 4;  
}
```

Добавление Обработчика по Нажатию Кнопки

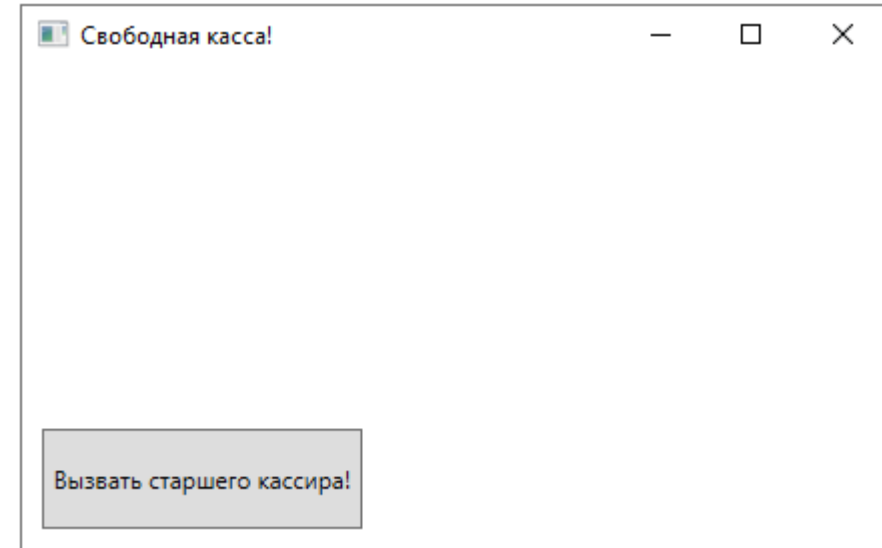
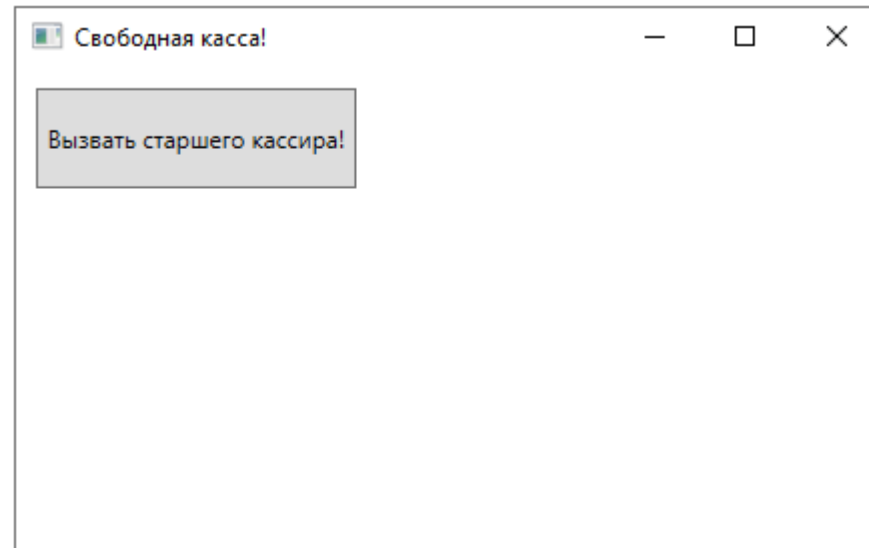
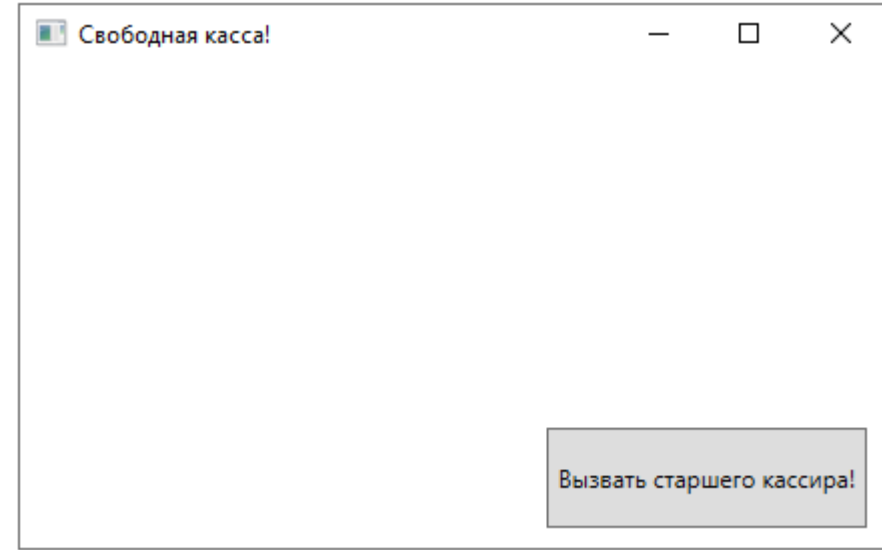
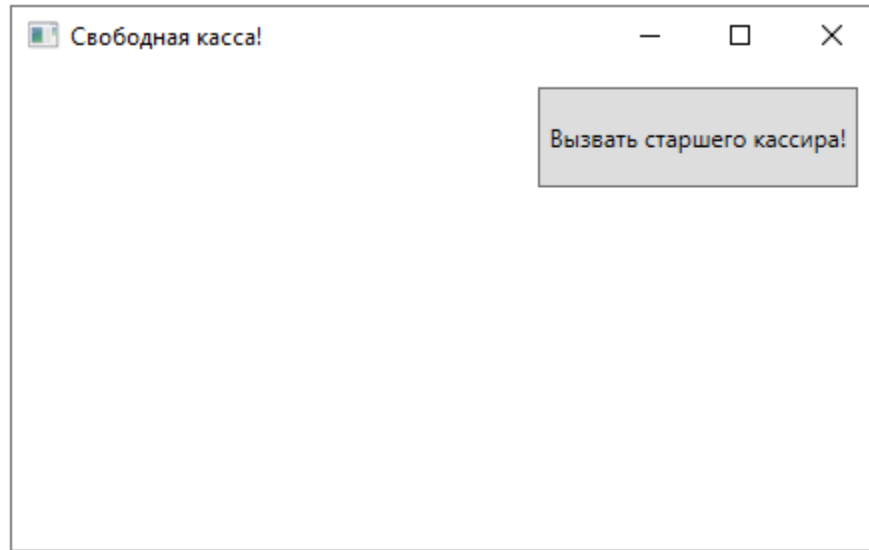
В качестве обработчика нажатия на кнопку используется событие делегат-типа `EventHandler<TEventArgs>` из BCL:

```
public delegate void EventHandler<TEventArgs>  
    (object source, TEventArgs e)  
where TEventArgs : EventArgs;
```

Для обработки события по нажатию кнопки необходимо подписать обработчик нажатия кнопки с вызовом метода `ChangeButtonPos()` после создания объекта-кнопки:

```
button.Click += (s, e) => ChangeButtonPos();
```

Изменение Состояний Кнопки по Нажатиям



Добавление Изображения

```
using System;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Media.Imaging;

namespace WPFDemoApp {
    public partial class MainWindow : Window {
        private Image imageJPG; // Ссылка на объект класса Image
        public MainWindow() {
            string curDir = Environment.CurrentDirectory;
            char sep = System.IO.Path.DirectorySeparatorChar;
            InitializeComponent();
            Title = "JPG-изображение в окне";
            imageJPG = new Image();
            imageJPG.Source = new BitmapImage(
                new Uri($"{curDir}{sep}parrot.jpg", UriKind.Absolute));
            Content = imageJPG;
        }
    }
}
```

Изображение лежит рядом
с исполняемым файлом.

Выведенная Картинка

