Руководство пользователя.

Содержание

[**1. ВВЕДЕНИЕ** 2](#_Toc96808255)

[1.1. ОБЛАСТЬ ПРИМЕНЕНИЯ 2](#_Toc96808256)

[1.2. КРАТКОЕ ОПИСАНИЕ ВОЗМОЖНОСТЕЙ 2](#_Toc96808257)

[1.3. УРОВЕНЬ ПОДГОТОВКИ ПОЛЬЗОВАТЕЛЯ 2](#_Toc96808258)

[1.4. ПЕРЕЧЕНЬ ЭКСПЛУАТАЦИОННОЙ ДОКУМЕНТАЦИИ 2](#_Toc96808259)

[**2. ПОДГОТОВКА К РАБОТЕ** 2](#_Toc96808260)

[2.1 УСТАНОВКА Visual Studio. 2](#_Toc96808261)

[2.1 Проверка работоспособности Visual Studio. 3](#_Toc96808262)

[2.2 Настройка компиляции проекта 6](#_Toc96808263)

[**3. ГРАФИЧЕСКИЙ РЕДАКТОР Windows Forms** 8](#_Toc96808264)

# **1. ВВЕДЕНИЕ**

## 1.1. ОБЛАСТЬ ПРИМЕНЕНИЯ

Windows Forms — это технология интеллектуальных клиентов для NET Framework. Она представляет собой набор управляемых библиотек, упрощающих выполнение стандартных задач, таких как чтение из файловой системы и запись в нее

## 1.2. КРАТКОЕ ОПИСАНИЕ ВОЗМОЖНОСТЕЙ

В состав Windows Forms входят многофункциональные элементы пользовательского интерфейса, позволяющие воссоздавать возможности таких сложных приложений, как Microsoft Office.

## 1.3. УРОВЕНЬ ПОДГОТОВКИ ПОЛЬЗОВАТЕЛЯ

Для облегчения работы с Windows Forms рекомендуется:

- ­ Иметь общие сведения о Visual Studio и языке программирования С#.

## 1.4. ПЕРЕЧЕНЬ ЭКСПЛУАТАЦИОННОЙ ДОКУМЕНТАЦИИ

1. Инструкция по установке Visual Studio.

2. Руководство пользователя (настоящий документ).

# **2.** **ПОДГОТОВКА К РАБОТЕ**

## 2.1 УСТАНОВКА Visual Studio.

* Зайти на сайт visualstudio.microsoft.com/ru/downloads/
* Нажать на раздел Скачать бесплатно

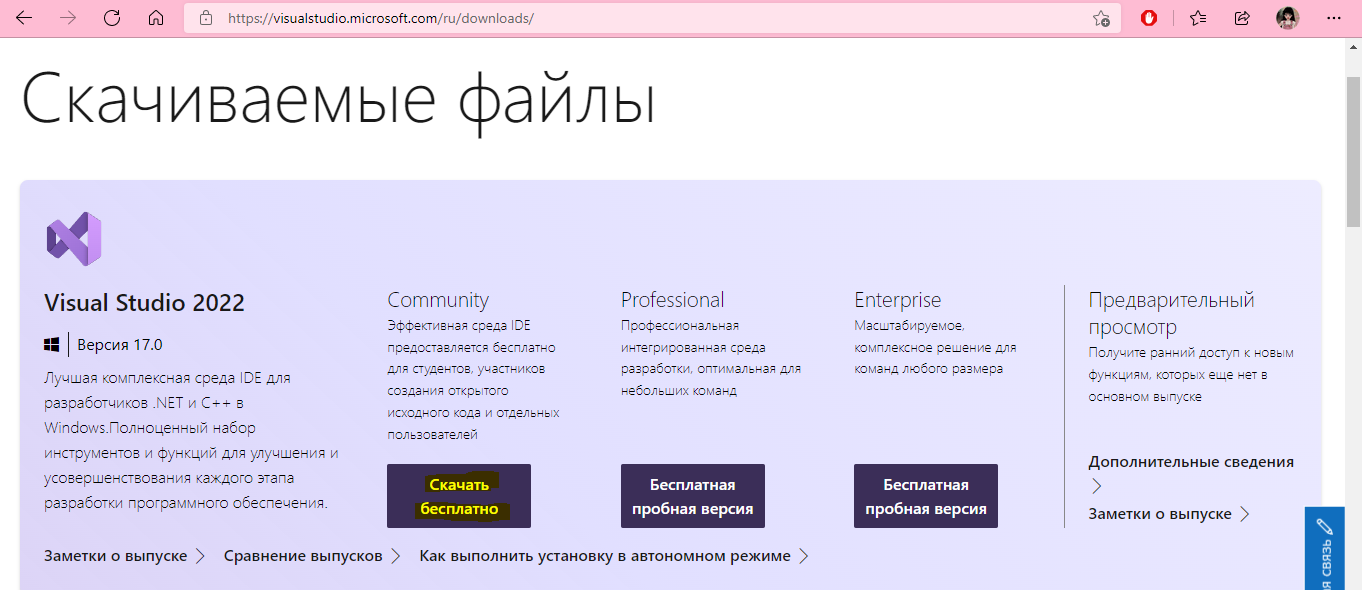


Рисунок 1. Установка с сайта.

* Чтобы добавить в Visual Studio поддержку проектов для Windows Forms и C# и .NET 6, в программе установки среди рабочих нагрузок нужно выбрать только пункт Разработка классических приложений .NET. Можно выбрать и больше опций или вообще все опции, однако стоит учитывать свободный размер на жестком диске - чем больше опций будет выбрано, соответственно тем больше места на диске будет занято. После того как загрузка завершилась требуется нажать на этот файл и запустить его. После запуска оставлять все параметры по умолчанию и нажимать Next до начала установки.

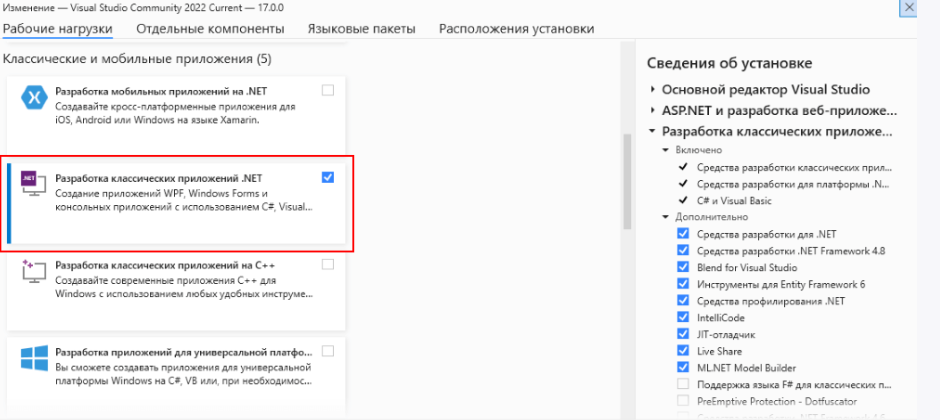


Рисунок 2. Выбор нужных опций.

## 2.1 Проверка работоспособности Visual Studio.

После установки среды и всех ее компонентов, запустим Visual Studio и создадим проект графического приложения. На стартовом экране выберем Create a new project (Создать новый проект)

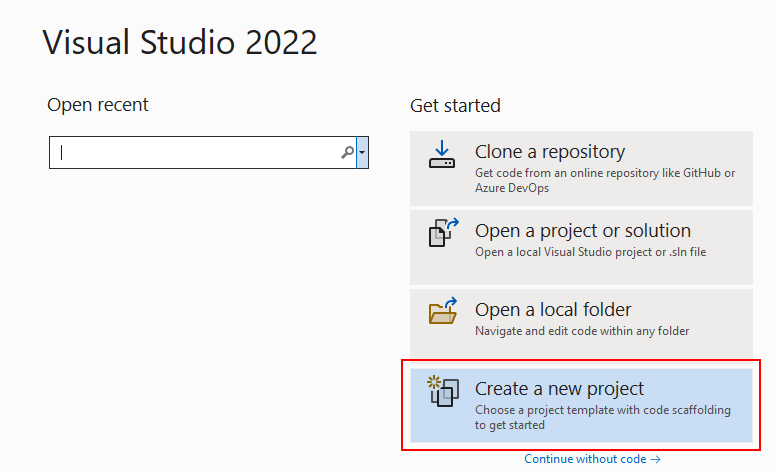


Рисунок 3. Создание проекта.

На следующем окне в качестве типа проекта выберем Windows Forms App:

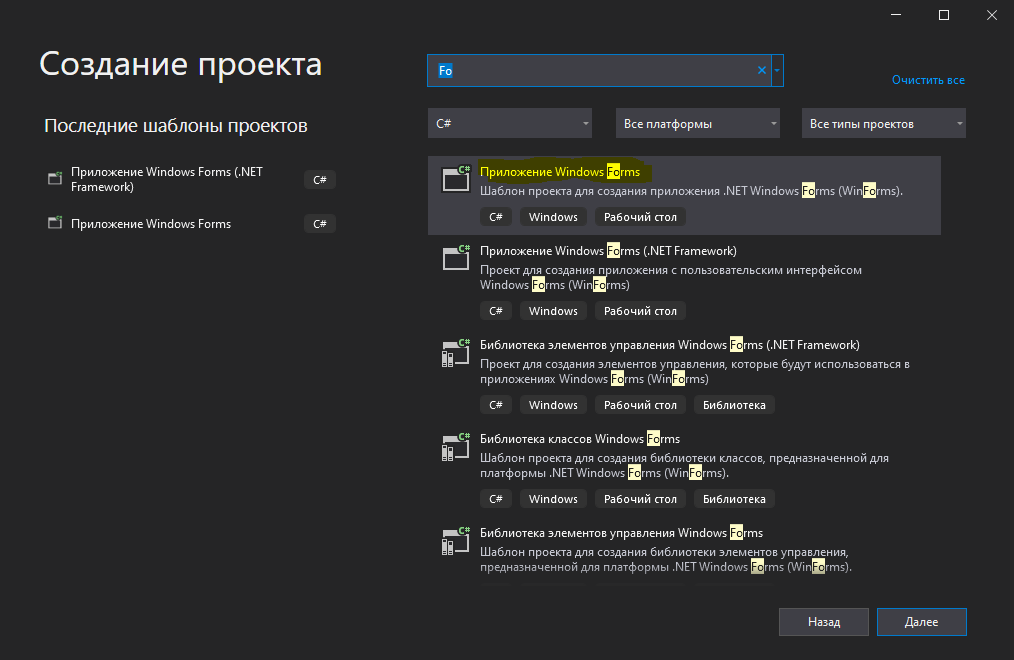


Рисунок 3. Выбор типа проекта.

Стоит отметить, что среди шаблонов можно увидеть еще тип Windows Forms App (.NET Framework) - его НЕ надо выбирать, необходим именно тип Windows Forms App.

Далее на следующем этапе нам будет предложено указать имя проекта и каталог, где будет располагаться проект.

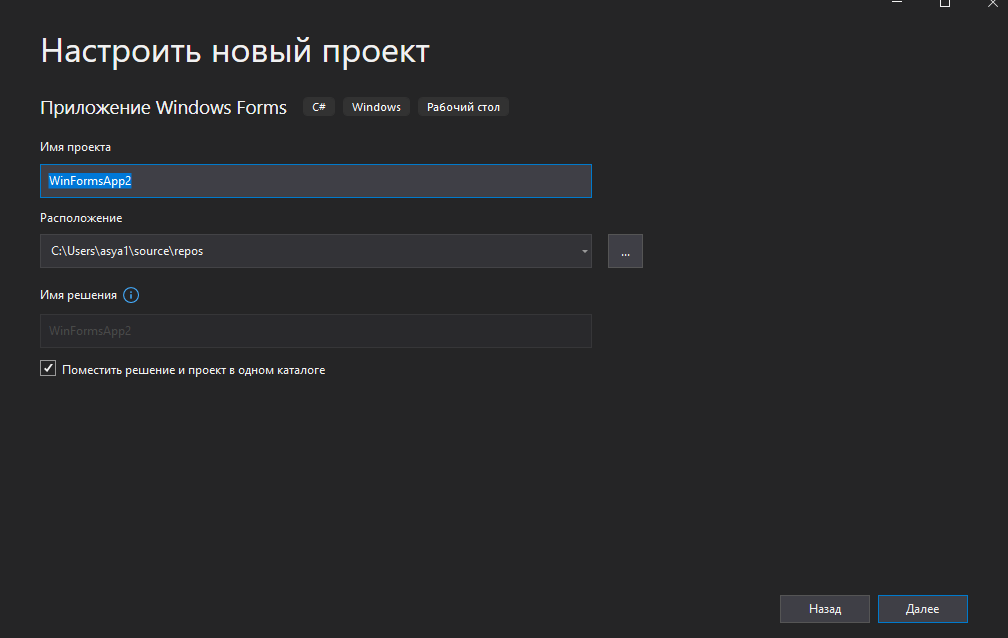


Рисунок 4. Настройка проекта.

На следующем окне Visual Studio предложит нам выбрать версию .NET, которая будет использоваться для проекта. По умолчанию здесь выбрана последняя на данный момент версия - .NET Core 3.1 Оставим и нажмен на кнопку Create (Создать) для создания проекта.

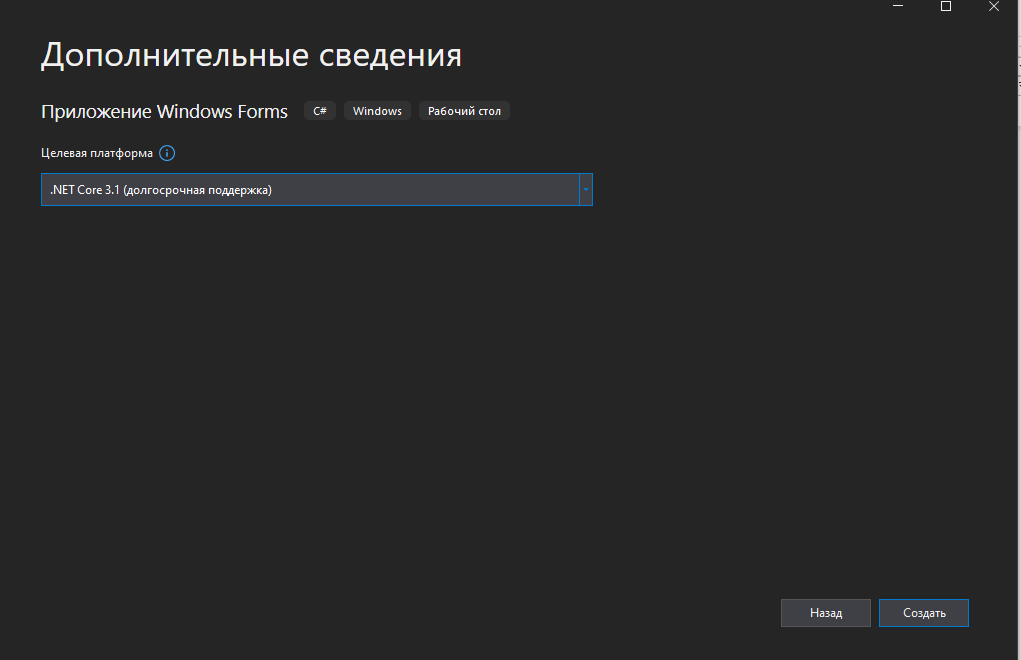


Рисунок 5. Дополнительные сведения о проекте.

Далее можно будет приступать к работе

### 2.2 Настройка компиляции проекта

Кроме того, проект Windows Forms имеет еще один важный файл, как и все проекты на языке C# - файл конфигурации проекта. Для его открытия нажмем двойным кликом левой кнопкой мыши на название проекта или нажмем правой кнопкой мыши на название проекта и появившемся контекстном меню выберем пункт Edit Project File

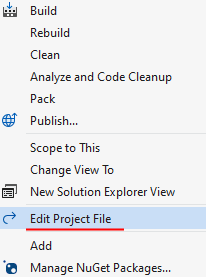


Рисунок 6. Edit Project File

В итоге нам откроется файл с расширением csproj, который называется по имени проекта (например, HelloApp.csproj) со следующим содержимым:

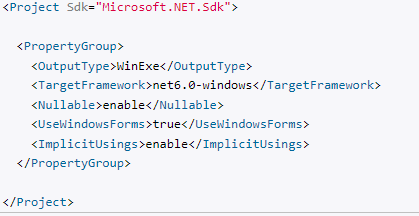


Рисунок 7. Расширение csproj.

Для компиляции приложения Windows Forms указаны следующие настройки:

* OutputType: определяет выходной тип проекта. Должен иметь значение WinExe - то есть выполняемое приложение с расширением exe под Windows
* TargetFramework: определяет применяемую для компиляции версию фреймворка .NET. Поскольку при создании проекта была выбрана версия .NET 6, а сам проект зависит от компонентов Windows, то здесь должно быть значение net6.0-windows
* Nullable: подключает в проект функционалность ссылочных nullable-типов
* UseWindowsForms: указывает, будет ли проект использовать Windows Forms (для этого устанавливается значение true)
* ImplicitUsings: подключает в проект функциональность неявно подключаемых глобальных пространств имен

#### 2.3. Запуск приложения

Чтобы запустить приложение в режиме отладки, нажмем на клавишу F5 или на зеленую стрелочку на панели Visual Studio.

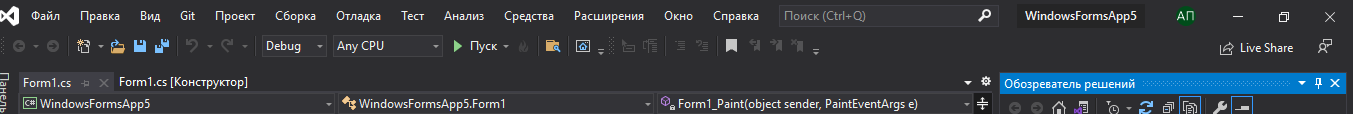


Рисунок 8. Отладка.

После этого запустится пустая форма Form1 по умолчанию.



Рисунок 9. Запуск формы.

После запуска приложения студия компилирует его в файл с расширением exe.

# **3. ГРАФИЧЕСКИЙ РЕДАКТОР Windows Forms**

Для открытия формы в режиме графического дизайнера нажмем на в структуре проекта на файл Form1.cs либо левой кнопкой мыши двойным кликом, либо правой кнопкой мыши и в появившемся контекстном меню выберем View Designer (также можно использовать комбинацию клавиш Shift+F7)

После этого в Visual Studio откроется выбранная форма в графическом виде.

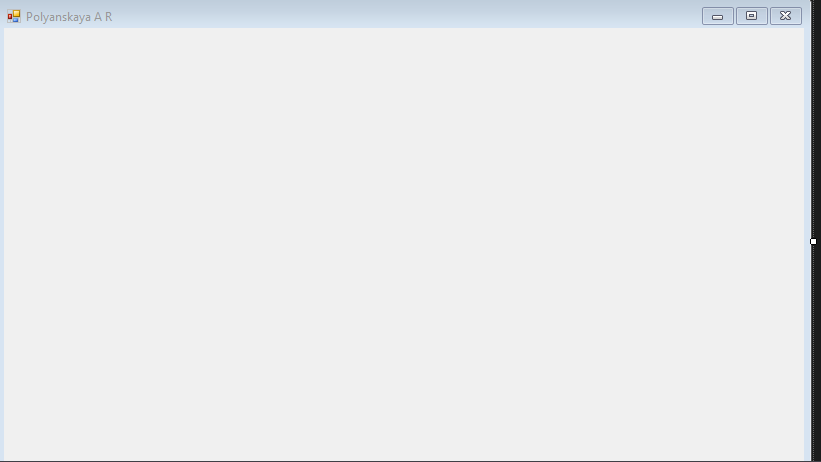


Рисунок 10. Новое название формы.

Но Visual Studio имеет еще одну связанную функциональность. Она обладает панелью графических инструментов. И мы можем, вместо создания элементов управления в коде C#, просто переносить их на форму с панели инструментов с помощь мыши. Так, перенесем на форму какой-нибудь элемент управления, например, кнопку. Для этого найдем в левой части Visual Studio вкладку Toolbox (Панель инструментов). Нажмем на эту вкладку, и у нас откроется панель с элементами, откуда мы можем с помощью мыши перенести на форму любой элемент:

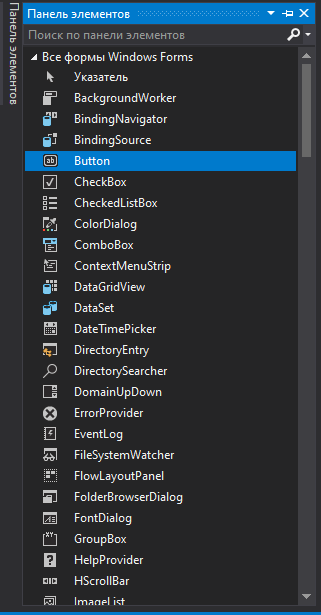


Рисунок 12. Панель элементов. Кнопка.

Найдем среди элементов кнопку и, захватив ее указателем мыши, перенесем на форму:

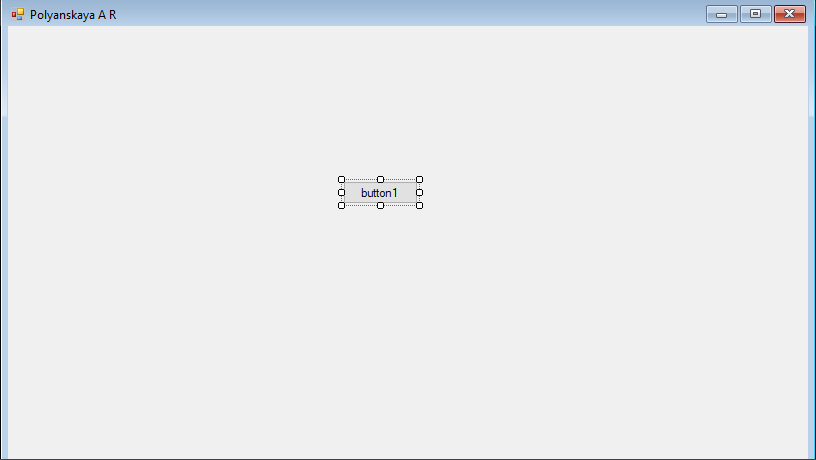


Рисунок 13. Кнопка на форме.

Причем при выборе кнопки она открывается в окне свойств и, как и для всей формы, для кнопки в окне свойств мы можем изменить значения различных свойств.

Кроме того, если после переноса кнопки на форму мы откроем файл Form1.

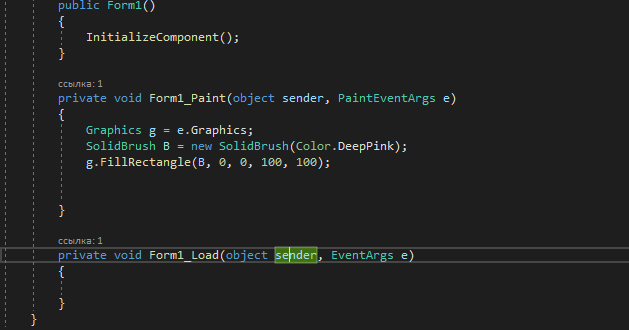


Рисунок 14.

Мы видим, что в класс Form1 была добавлена переменная button1 типа Button и для этой переменной, как и для объекта формы, задан ряд свойств. И если в окне свойств мы поменяем значения этих свойств, то в этом файле также изменяться их значения.

Добавим простейший код на языке C#, который бы выводил сообщение по нажатию кнопки. Для этого перейдем в файл кода Form1.cs, который связан с этой формой.

Кнопка обладает событием Click, которое генерируется при нажатии. В данном случае в конструкторе формы мы подвязываем к кнопке button1 в качестве обработчика события нажатия метод button1\_Click, в котором с помощью метода MessageBox.Show выводит сообщение. Текст сообщения передается в метод в качестве параметра.

Стоит отметить, что графический дизайнер позволяет автоматически сгенерировать обработчик нажатия кнопки. Для этого надо в окне дизайнера нажать на кнопку на форме двойным щелчком мыши.

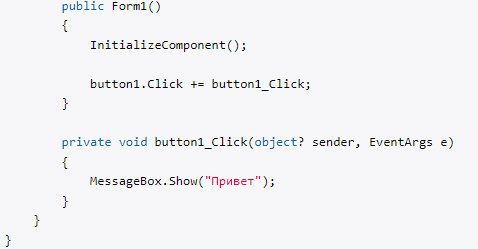


Рисунок 15. Код.

Теперь запустим проект и мы увидим форму с кнопкой, на которую мы можем нажать и получить сообщение:

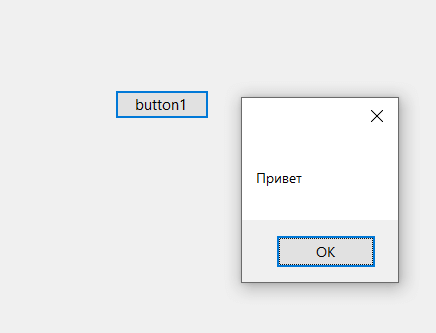


Рисунок 16. Результат.