

**Объектно-ориентированное  
программирование.**

**Обзор среды разработки**

**Visual Studio .NET**

ОГУ

Кафедра ВТиЗИ

Галимов Р.Р.

# Исходные данные по курсу

Лекций -17

Лабораторные работы -16

Курсовой проект

Итоговый контроль –экзамен.

Среда программирования: Visual Studio C#  
12

Язык программирования: C#

# Контроль знаний

- Лабораторные работы -4 балла (10 лабораторных работ)=40
- Тесты – 5 баллов(2 раза)=10
- КП – от 0 до 20 баллов = максимум 20
- Посещение занятий -0.5 =16 баллов
- Экзамен -14 баллов
- Допуск к экзамену 70 баллов

# Содержание

1. Язык C# и каркас .NET
2. Обзор среды разработки Visual Studio .NET
3. Создание первого приложения для Windows
4. Задание на лабораторную работу №1.
5. Контрольные вопросы

# Язык C# и каркас .NET

- В 2000 году компания Microsoft представила новый язык программирования C#.
- Своими корнями C# уходит к языкам программирования C, C++ и Java.
- C# обладает примерно такими же возможностями, как и Java; он подходит для самых ответственных задач разработки, включая построение современных крупномасштабных корпоративных приложений, мобильных и «облачных» приложений. ◆

# Объектно-ориентированное программирование

- C# относится к категории объектно-ориентированных языков.
- Программы, написанные на C#, могут использовать **.NET Framework Class Library** - гигантскую библиотеку готовых классов, ускоряющую разработку приложений

# Объектно-ориентированное программирование

## Ключевые возможности .NET Framework Class Library

Базы данных

Отладка

Построение веб-приложений

Многопоточное программирование

Графика

Операции с файлами

Ввод-вывод

Безопасность

Сетевые взаимодействия

Веб-взаимодействия

Управление разрешениями

Графический интерфейс

Мобильные приложения

Структуры данных

Обработка строк

# Событийное программирование

- Программирование на C# управляется событиями.
- Разработчик пишет программу, которая реагирует на события, инициированные пользователем: щелчки кнопкой мыши, нажатия клавиш, срабатывания таймера и (новая возможность Visual C# 2012) прикосновения к сенсорному экрану и жесты, широко используемые на смартфонах и планшетах.



# Визуальное

## программирование

- Microsoft Visual C# является *визуальным языком программирования* – помимо написания команд для построения частей приложения, разработчик может использовать графический интерфейс (GUI, Graphic User Interface) Visual Studio для удобного перетаскивания заранее определенных объектов (таких, как кнопки и текстовые поля) *в нужное место экрана, назначения их текста и изменения размеров.*
- Visual Studio сгенерирует большую часть GUI-кода за вас.

# СРЕДА ВЫПОЛНЕНИЯ Common Language Runtime

- Библиотека классов — это статическая составляющая каркаса.
- В .NET Framework есть и динамическая составляющая — система, определяющая среду выполнения, — CLR (Common Language Runtime).
- Роль этой среды весьма велика — в ее функции входит управление памятью, потоками, безопасностью, компиляция из промежуточного байт-кода в машинный код и многое другое.
- Важный элемент CLR — это мощный механизм «сборки мусора» (garbage collector), управляющий работой с памятью.

# СРЕДА ВЫПОЛНЕНИЯ Common Language Runtime

- Среда CLR (**Common Language Runtime**), еще один ключевой компонент .NET Framework, исполняет программы .NET и предоставляет многие полезные средства для упрощения разработки и отладки.
- CLR представляет собой *виртуальную машину (VM, Virtual Machine)* - программу, которая управляет выполнением других программ и изолирует их от базовой операционной системы и оборудования.
- Исходный код программ, выполняемых под управлением CLR, называется *управляемым кодом*.
- CLR предоставляет управляемому коду различные виды сервиса - интеграцию программных компонентов, написанных на разных языках .NET, обработку ошибок между такими компонентами, повышенную безопасность, автоматическое управление памятью и т. д.

# Платформенная

## независимость

- Если версия .NET Framework существует и установлена на платформе, эта платформа может выполнить любую программу .NET.
- Способность программы выполняться без изменений на разных платформах называется *платформенной независимостью*.
- Единожды написанный код может использоваться на другом типе компьютеров без изменений, а это экономит время и деньги. Вдобавок платформенная независимость расширяет круг пользователей.
- Прежде компаниям приходилось решать, оправдаются ли затраты по адаптации их программ для других платформ (этот процесс называется *портированием*).
- С появлением .NET проблемы портирования отпали (по крайней мере после того, как версии .NET стали доступными на разных платформах).

# Строгая типизация данных

- Все переменные имеют четко определенный конкретный тип данных.
- Обычно не допускаются никакие действия, в результате которых получаются неопределенные типы данных.
- Общая система типов (Common Type System -CTS) описывает predetermined типы данных.

# Сборка мусора

- Сборщик мусора- это инструмент для восстановления памяти, которую запрашивало приложение.
- При недостатки памяти для приложения исполняющая среда вызывает сборщик мусора для освобождения объектов, на которых нет ссылки в коде.
- Сборщик мусора вызывается исполняющей средой по своему усмотрению

# Visual Studio 13 C#

Начальная страница - Microsoft Visual Studio

ФАЙЛ ПРАВКА ВИД QT5 ОТЛАДКА КОМАНДА СЕРВИС ТЕСТ АНАЛИЗ ОКНО СПРАВКА

Начальная страница

## Professional 2013

### Узнайте, что нового в Professional 2013

Информацию о новых функциях и улучшениях в Professional 2013 мож в следующих разделах.

[Сведения о новых функциях см. в разделе Professional 2013](#)  
[Новые возможности .NET Framework 4.5.1](#)  
[Новые возможности службы Team Foundation Service](#)

[Подключитесь к Azure](#)  
[Подробнее о Azure](#)  
[Подключить](#)

[Переместить информацию о новых возможностях](#)

[Новые возможности на платформах Microsoft](#)

Обозреватель серверов  
Панель элементов  
Источники данных

Пуск

[Создать проект...](#)

[Открыть проект...](#)

[Открыть из системы управления версиями...](#)

Последние

[recog\\_server](#)

[recog\\_client](#)

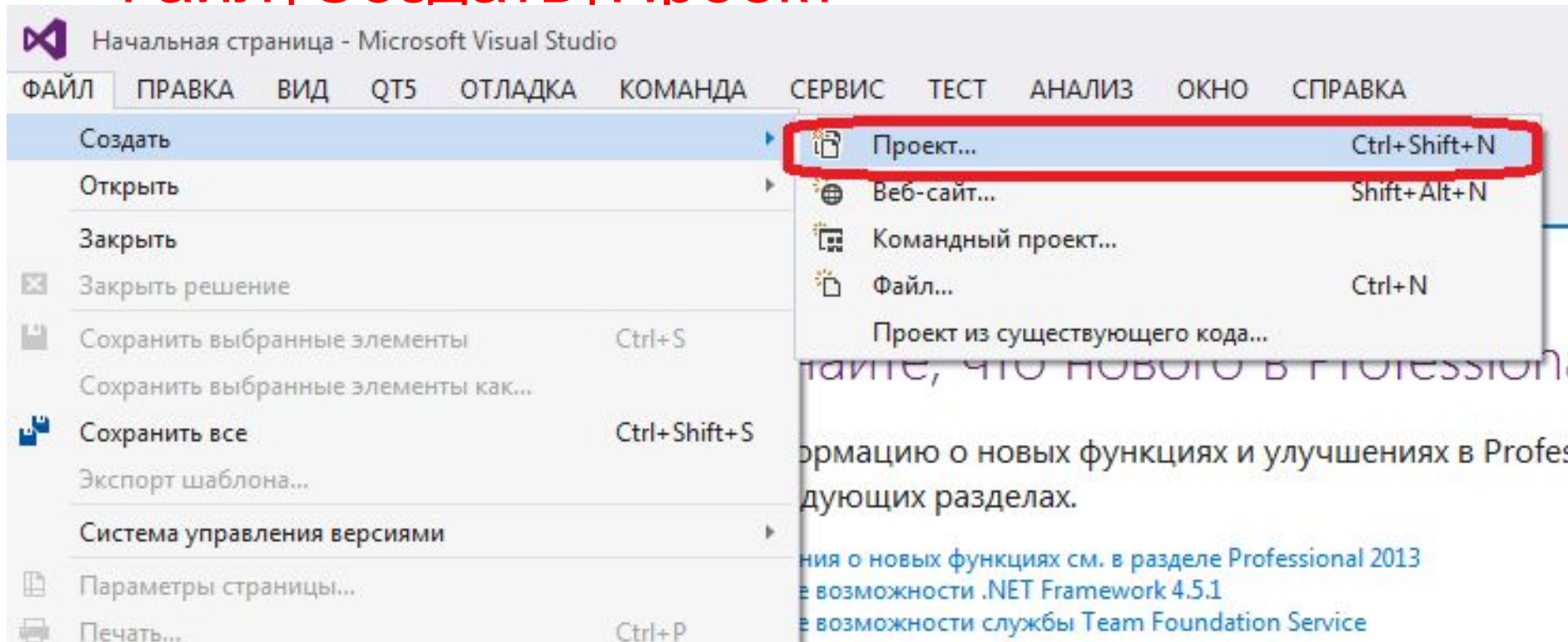
[Test1](#)

[IntegrationSite](#)

# Visual Studio 13 C#

Создать новый проект:

Файл | Создать | Проект





# Visual Studio 13 C#

Создать проект

Последние файлы .NET Framework 4.5 Сортировать по: По умолчанию

Установлено: Шаблоны - поиск (Ctrl+P)

Установленные

Шаблоны

- Visual Basic
- Qt5 Projects
- Visual C#
- Рабочий стол Windows
- Веб
- Office/SharePoint
- Cloud
- LightSwitch
- Reporting
- Silverlight
- WCF
- Workflow
- Тест
- Visual C++

В сети

	Приложение Windows Forms	Visual C#
	Приложение WPF	Visual C#
	Консольное приложение	Visual C#
	Веб-приложение ASP.NET	Visual C#
	Библиотека классов	Visual C#
	Библиотека классов (переносимая)	Visual C#
	Приложение Silverlight	Visual C#
	Библиотека классов Silverlight	Visual C#
	Приложение службы WCF	Visual C#

[Щелкните здесь для поиска шаблонов в Интернете.](#)

Тип: Visual C#  
Проект, для создания приложения с пользовательским интерфейсом Windows Forms

Имя: FirstProg

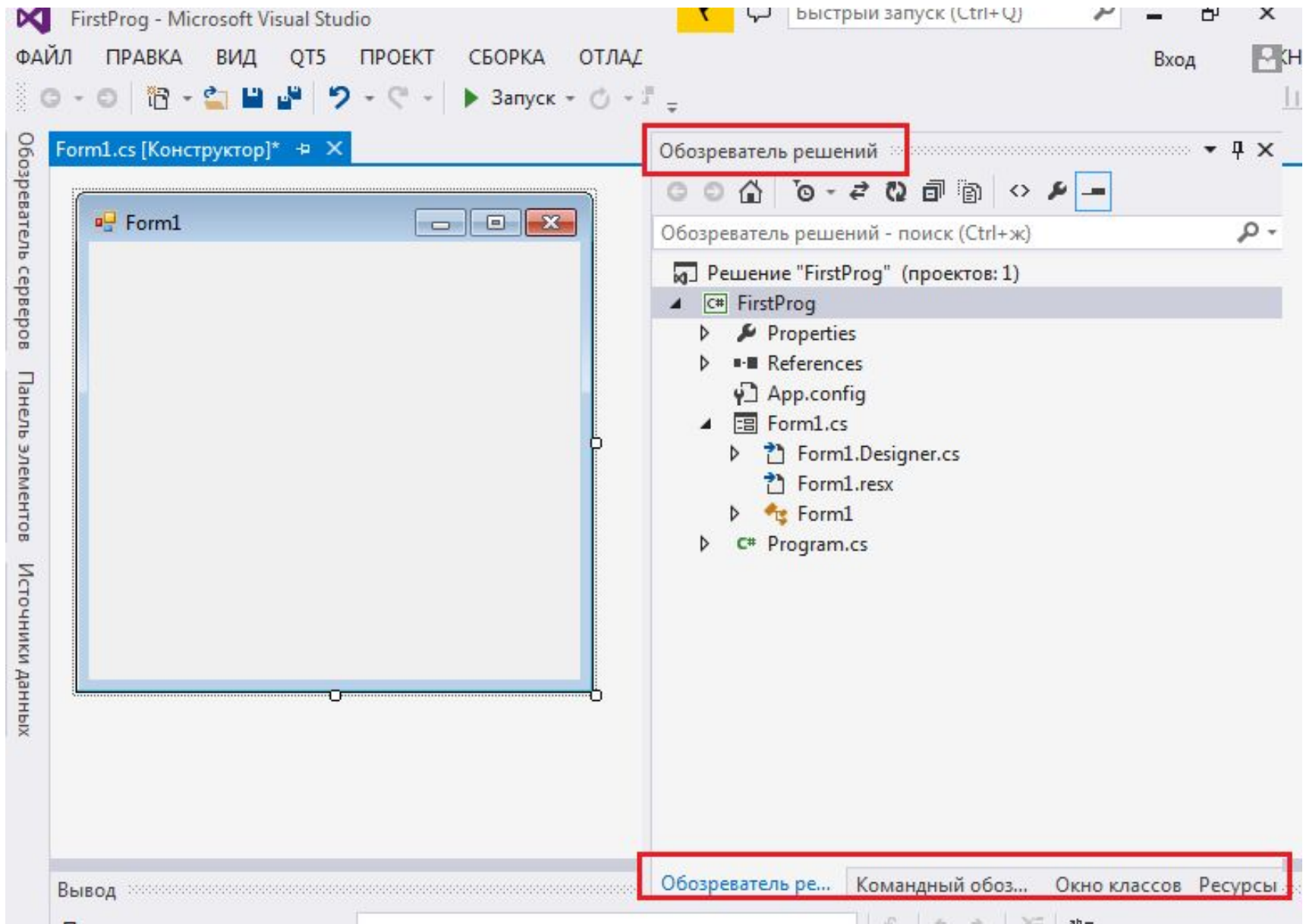
Расположение: D:\Work\ASP\ООП\C#\Projects\ Обзор...

Имя решения: FirstProg

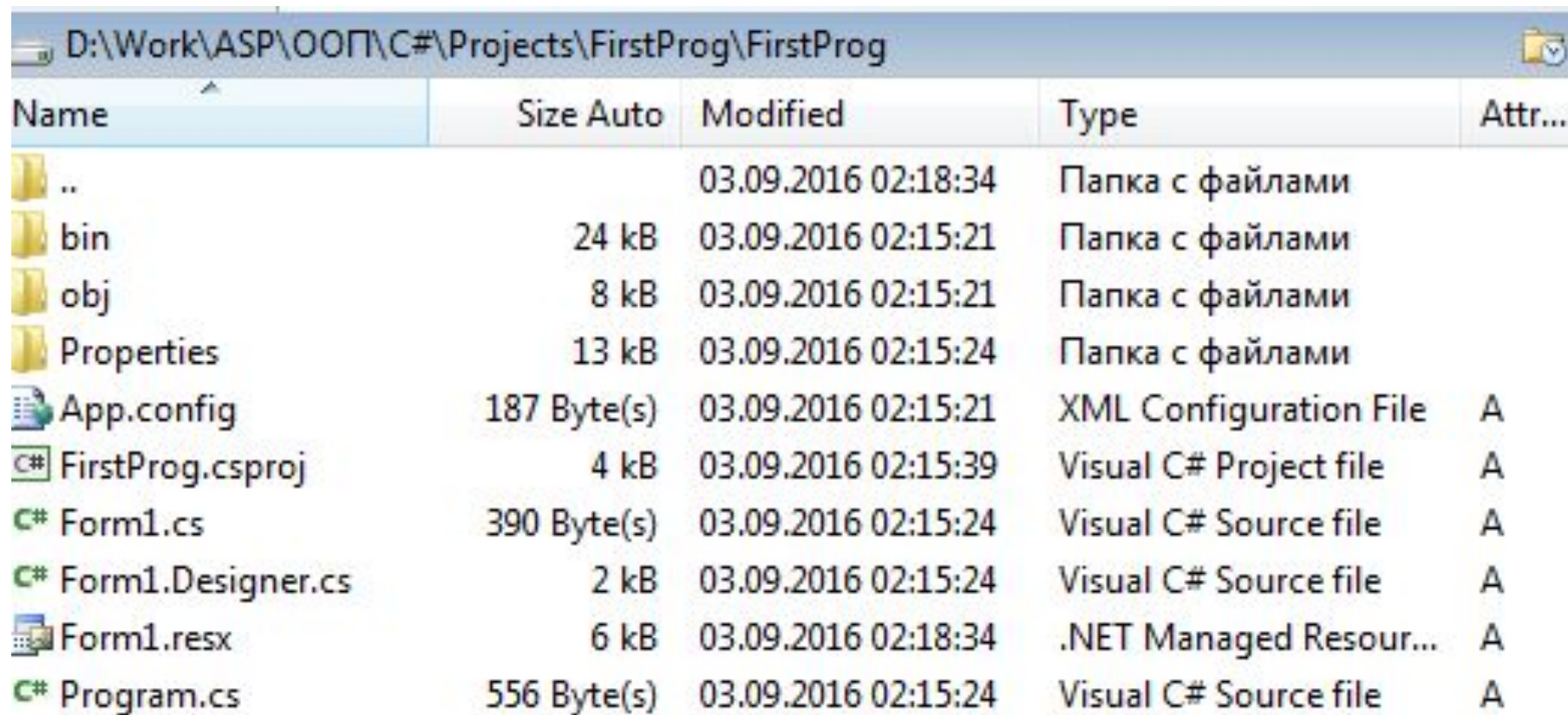
Создать каталог для решения

Добавить в систему управления версиями

# Visual Studio 13 C#

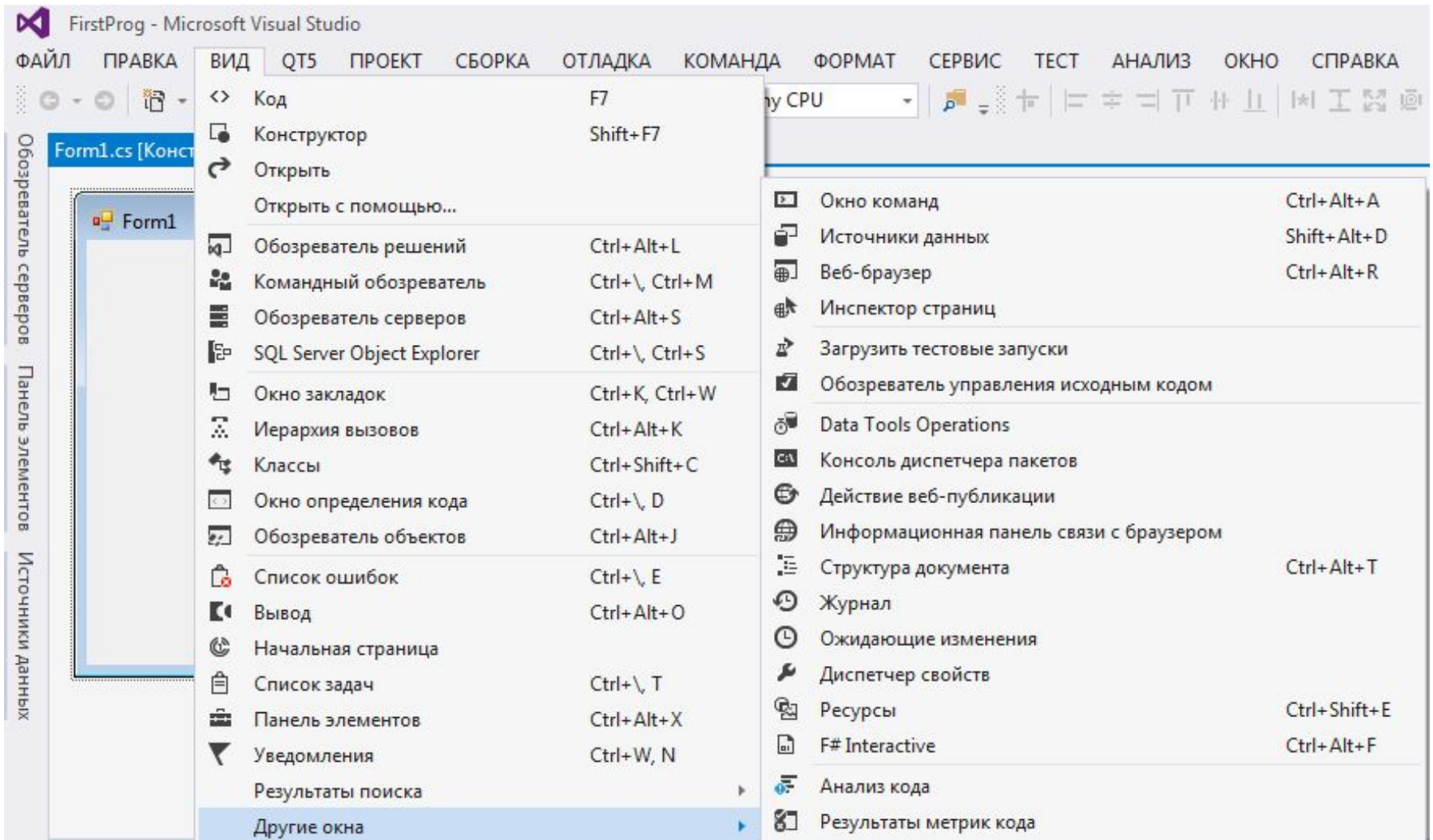


# Visual Studio 13 C#



Name	Size Auto	Modified	Type	Attr...
..		03.09.2016 02:18:34	Папка с файлами	
bin	24 kB	03.09.2016 02:15:21	Папка с файлами	
obj	8 kB	03.09.2016 02:15:21	Папка с файлами	
Properties	13 kB	03.09.2016 02:15:24	Папка с файлами	
App.config	187 Byte(s)	03.09.2016 02:15:21	XML Configuration File	A
FirstProg.csproj	4 kB	03.09.2016 02:15:39	Visual C# Project file	A
Form1.cs	390 Byte(s)	03.09.2016 02:15:24	Visual C# Source file	A
Form1.Designer.cs	2 kB	03.09.2016 02:15:24	Visual C# Source file	A
Form1.resx	6 kB	03.09.2016 02:18:34	.NET Managed Resour...	A
Program.cs	556 Byte(s)	03.09.2016 02:15:24	Visual C# Source file	A

# Visual Studio 13 C#



# Visual Studio 13 C#

The image shows a screenshot of the Visual Studio 13 C# IDE. The main window displays the code for `Form1.cs` in the `FirstProg` namespace. The code includes several `using` statements and a `public partial class Form1` definition with a constructor that calls `InitializeComponent()`.

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace FirstProg
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

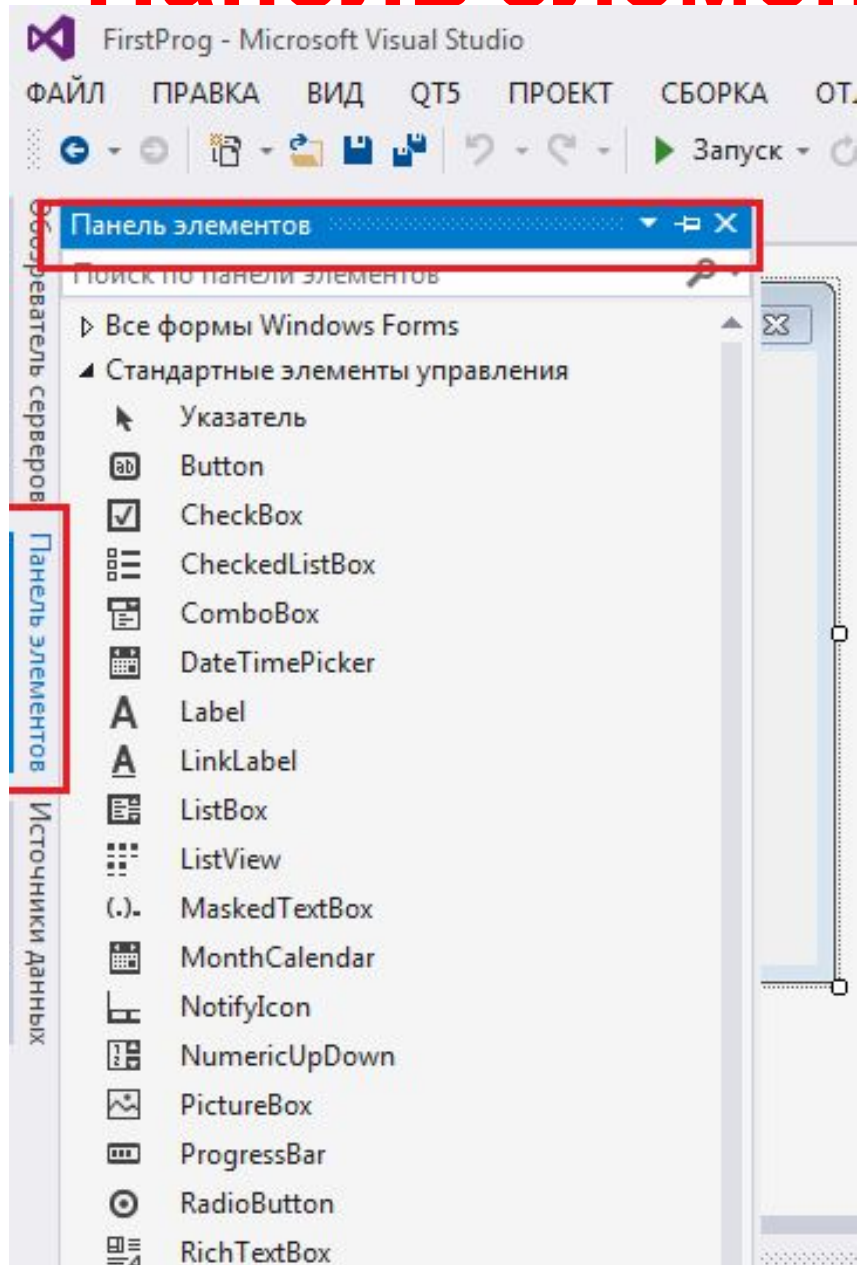
The right-hand side of the IDE shows the **Окно классов** (Class Explorer) pane. It displays the project structure, including the `Form1` class. Below the class explorer, the **Свойства** (Properties) pane is visible, showing the `Dispose(bool)`, `Form1()`, `InitializeComponent()`, and `components` properties.

The top of the IDE shows the menu bar with options like **ФАЙЛ**, **ПРАВКА**, **ВИД**, **QT5**, **ПРОЕКТ**, and **СБОРКА**. The toolbar includes icons for opening files, saving, and running. The status bar at the bottom indicates the current file is `Form1.cs` and the project is `FirstProg`.

# Формы и элементы управления

- Прямоугольник с заголовком Form (*форма*) представляет главное окно создаваемого приложения Windows Forms.
- Приложения Visual C# могут содержать несколько форм ( окон);
- Visual Studio содержит много готовых элементов управления и других компонентов, которые могут использоваться для создания и настройки поведения приложений.

# Панель элементов



# Панель элементов

- Чтобы вызвать окно панели элементов (Toolbox), выполните команду VIEW | Toolbox.
- В нем содержатся элементы, используемые для модификации форм.
- В методологии визуального программирования разработчик «перетаскивает» элементы управления на форму, а IDE генерирует код создания этих элементов.
- Так код создается проще и быстрее, чем при самостоятельном написании.
- Чтобы вести машину, не обязательно знать, как устроен ее двигатель; точно так же не обязательно уметь строить элементы, чтобы использовать их.
- Повторное использование готовых элементов экономит время и деньги при разработке.



# Окно свойств

The image shows a screenshot of Microsoft Visual Studio. The main window displays a Windows Form titled "Первая программа" (First Program). The Properties window on the right is open, showing the properties for the selected form. The form's text property is set to "Первая программа".

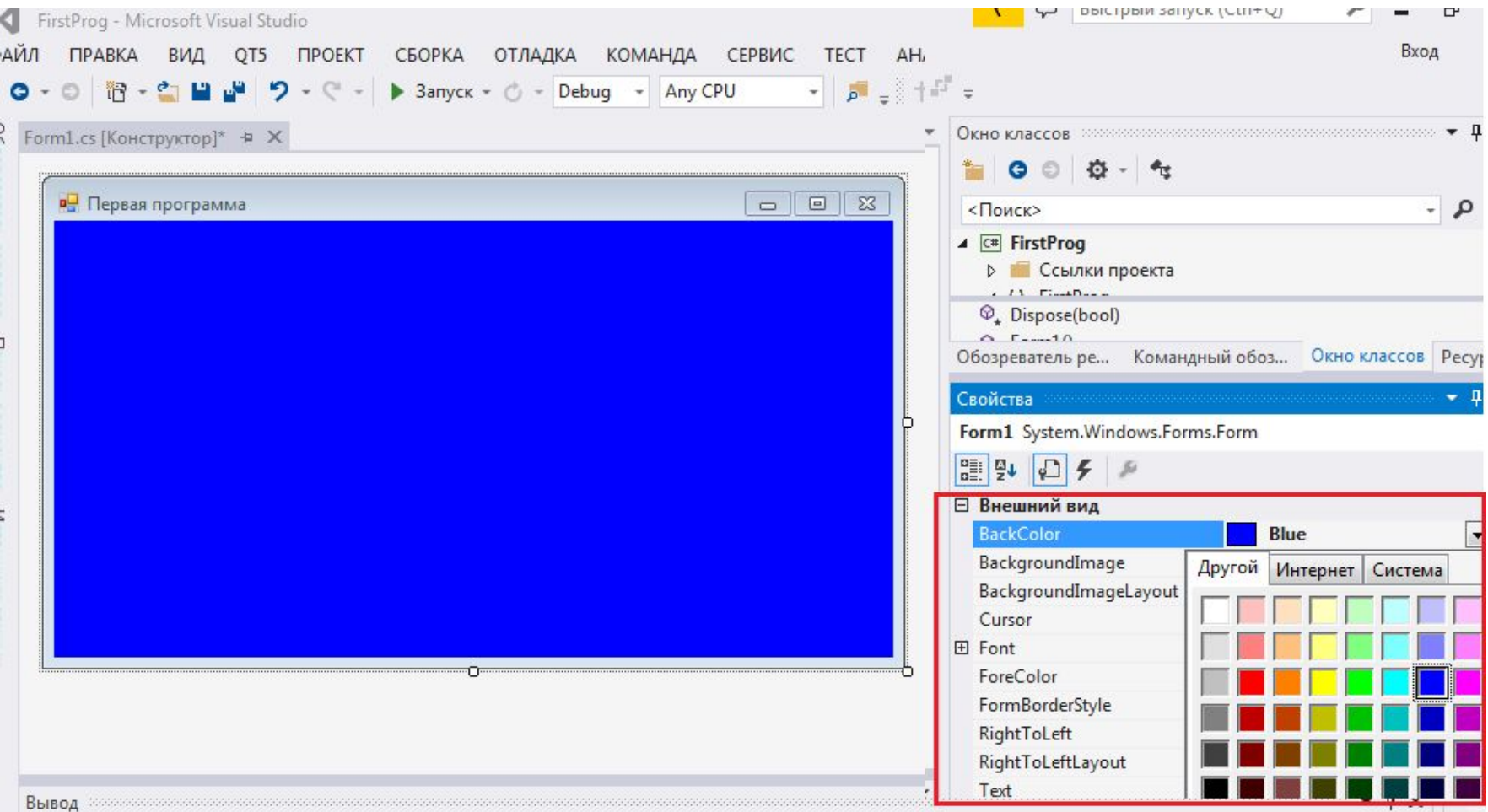
The Properties window shows the following properties for **Form1** (System.Windows.Forms.Form):

Cursor	Default
Font	Microsoft Sans Serif; 8,25pt
ForeColor	ControlText
FormBorderStyle	Sizable
RightToLeft	No
RightToLeftLayout	False
<b>Text</b>	<b>Первая программа</b>
UseWaitCursor	False

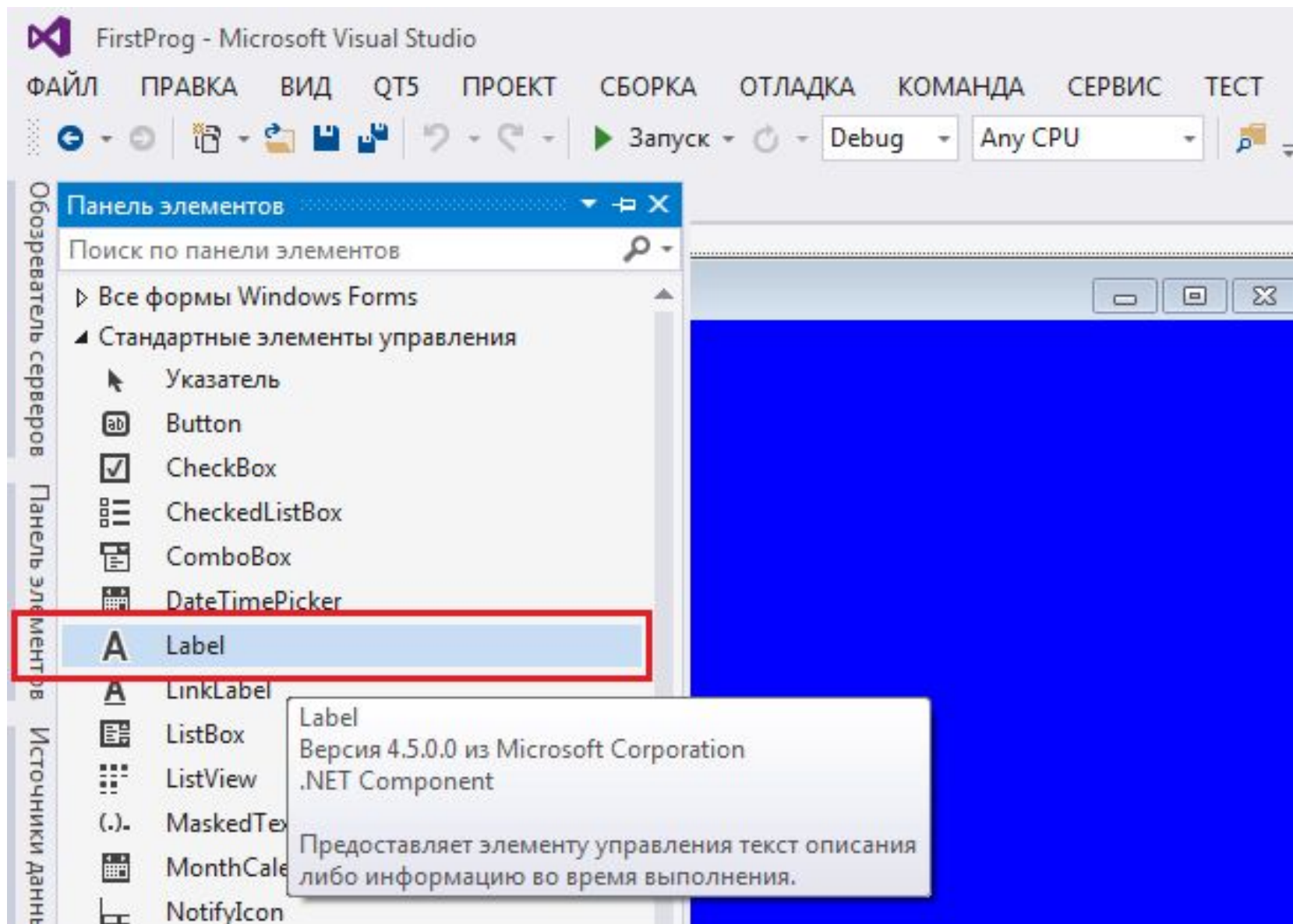
# Окно свойств

- Если окно свойств (Properties) не отображается под окном Solution Explorer, выполните команду VIEW | Properties Window.
- В окне свойств выводятся свойства текущей выделенной формы, элемента управления или файла.
- Свойства описывают характеристики формы или элемента управления: размер, цвет, позицию и т. д.
- Разные формы и элементы управления обладают разными наборами свойств – описание свойства выводится в нижней части окна свойств при выделении этого свойства.

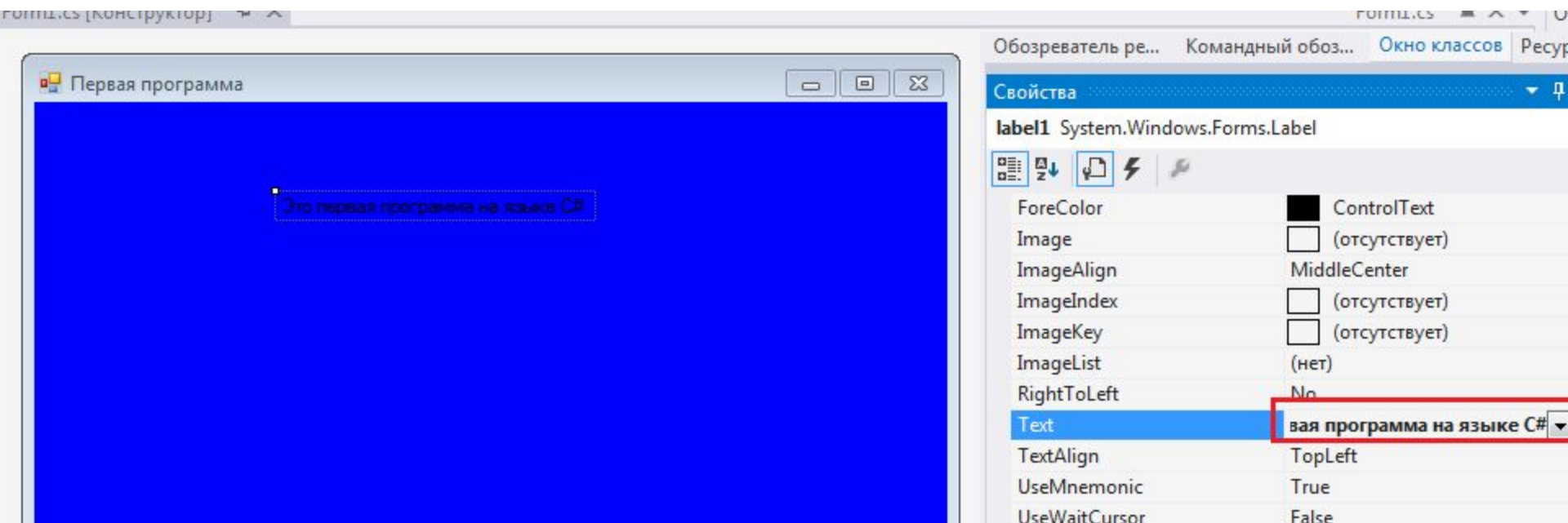
# Первая программа



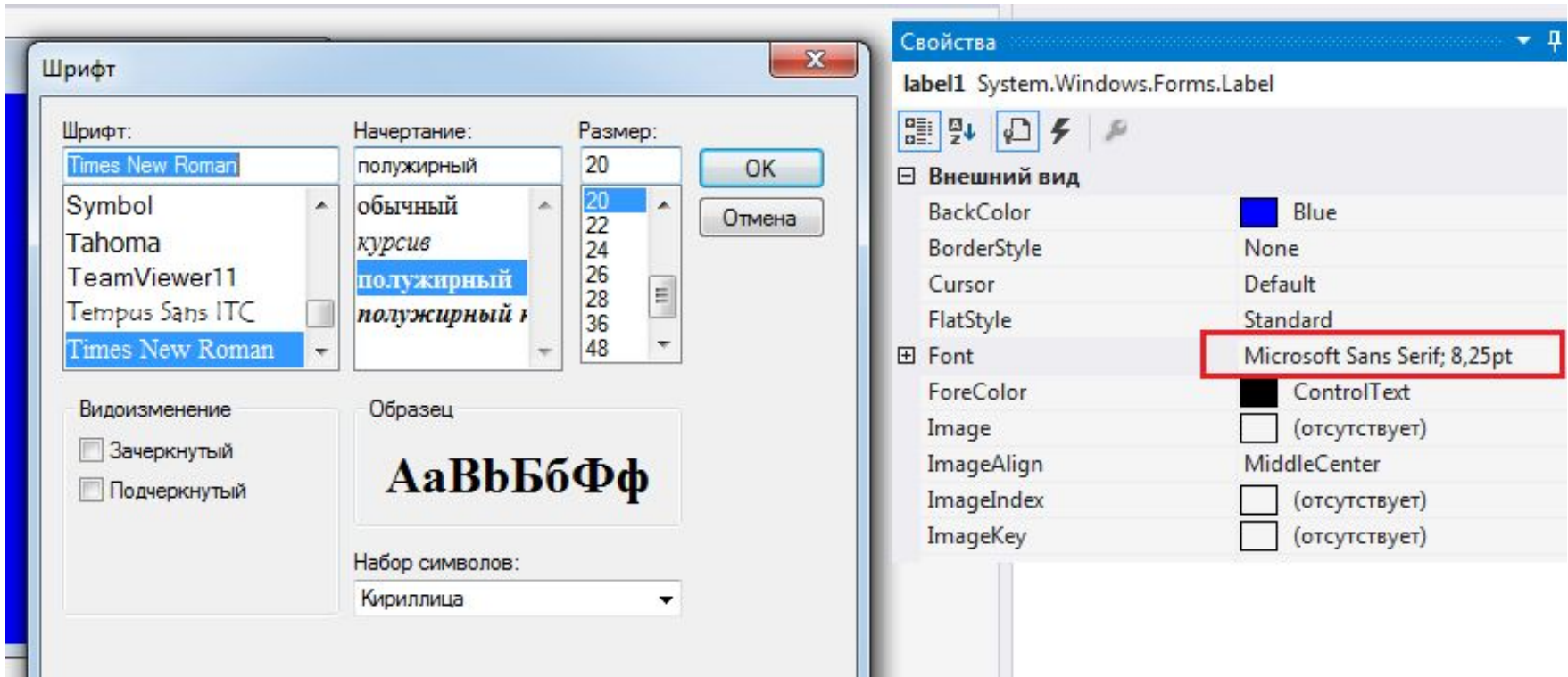
# Первая программа



# Первая программа



# Первая программа



# Первая программа

The image shows a screenshot of a Windows Forms application. The main window, titled "Первая программа", has a blue background. In the center, there is a text box containing the text "Это первая программа на языке C#" in a black, serif font. The text box is surrounded by a dashed border with small squares at the corners, indicating it is selected. To the right of the window is the "Свойства" (Properties) window for the selected control, "label1 System.Windows.Forms.Label". The "TextAlign" property is set to "MiddleCenter" and is highlighted with a red box. The "AutoSize" property is set to "False" and is also highlighted with a red box. Other visible properties include "UseMnemonic", "UseWaitCursor", "Данные", "(ApplicationSettings)", "(DataBindings)", "Tag", "Макет", "Anchor" (set to "Top, Left"), and "Dock" (set to "None").

Первая программа

Это первая программа на языке C#

Свойства

label1 System.Windows.Forms.Label

TextAlign MiddleCenter

UseMnemonic

UseWaitCursor

Данные

(ApplicationSettings)

(DataBindings)

Tag

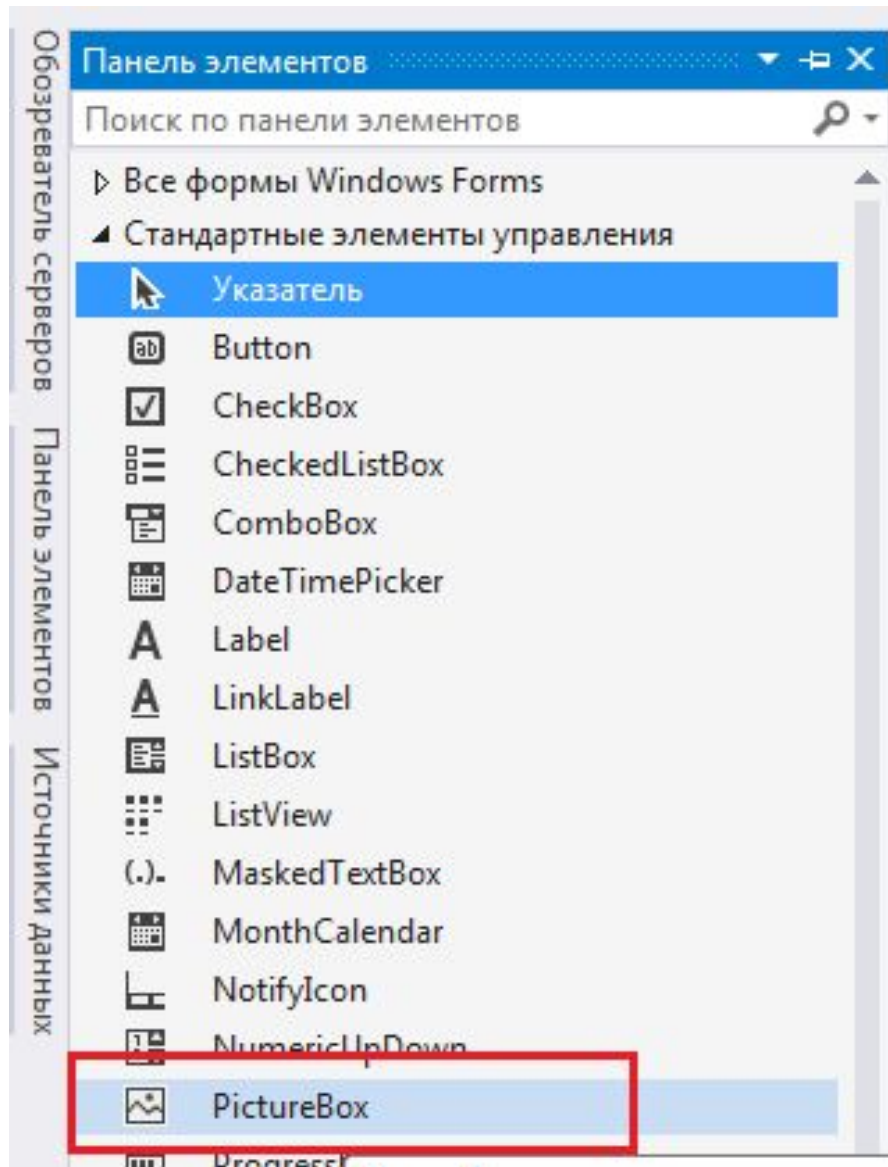
Макет

Anchor Top, Left

AutoSize False

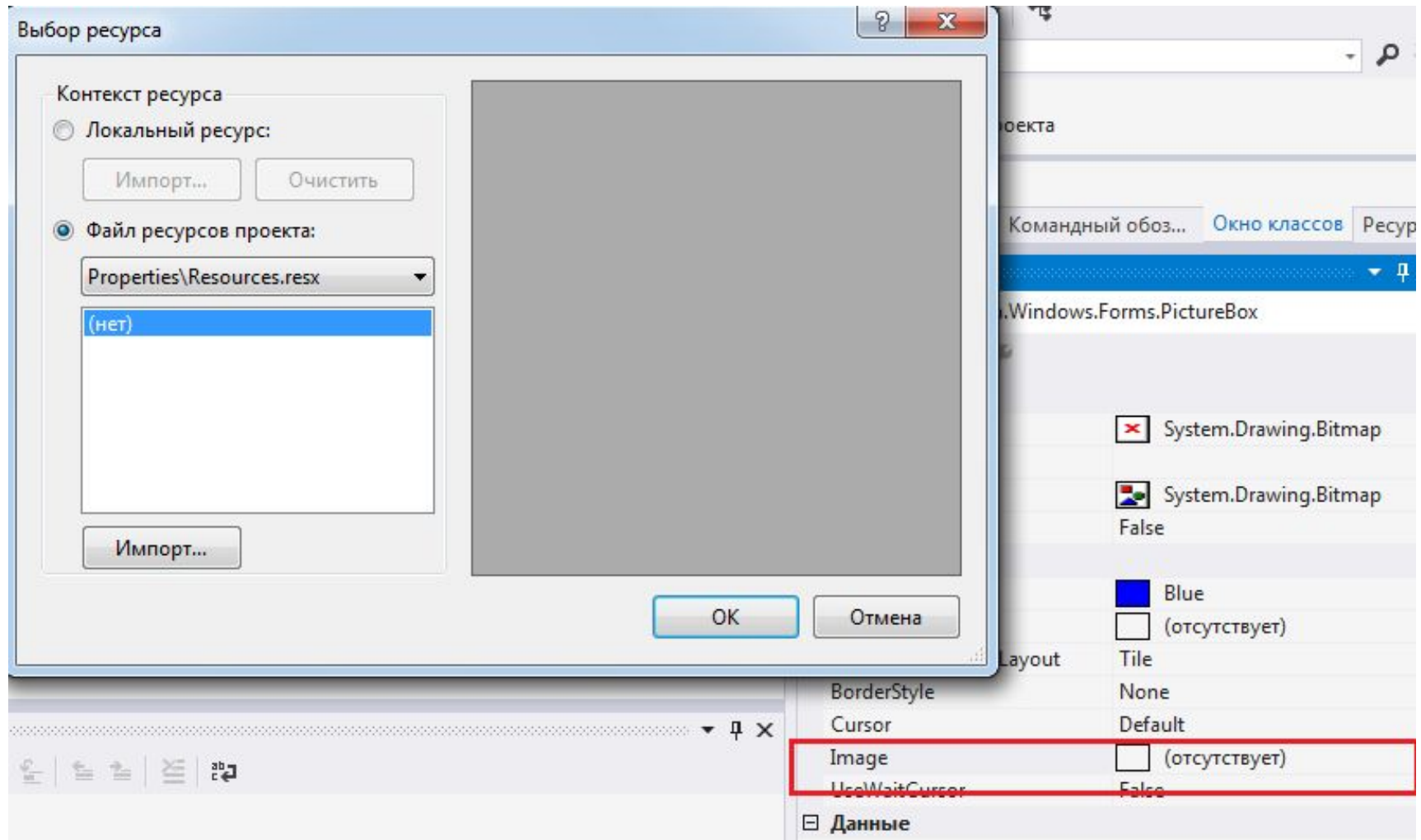
Dock None

# Первая программа





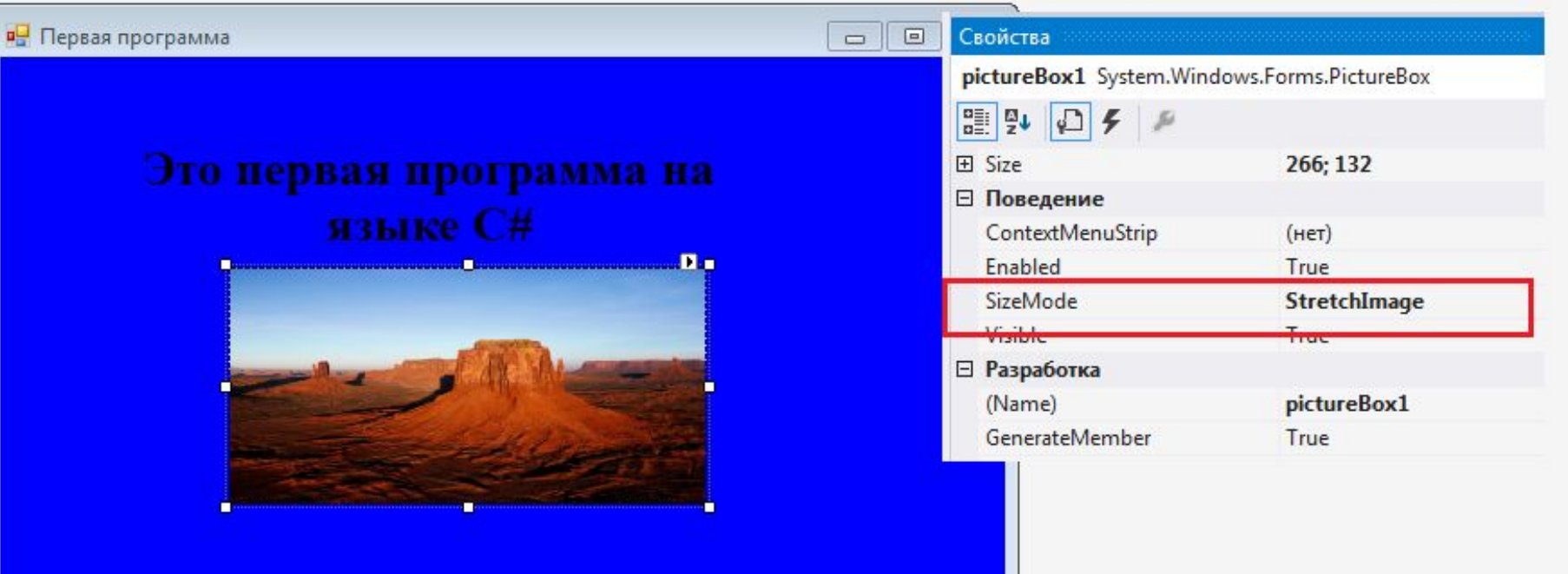
# Первая программа



# Первая программа

Первая программа

Это первая программа на языке C#

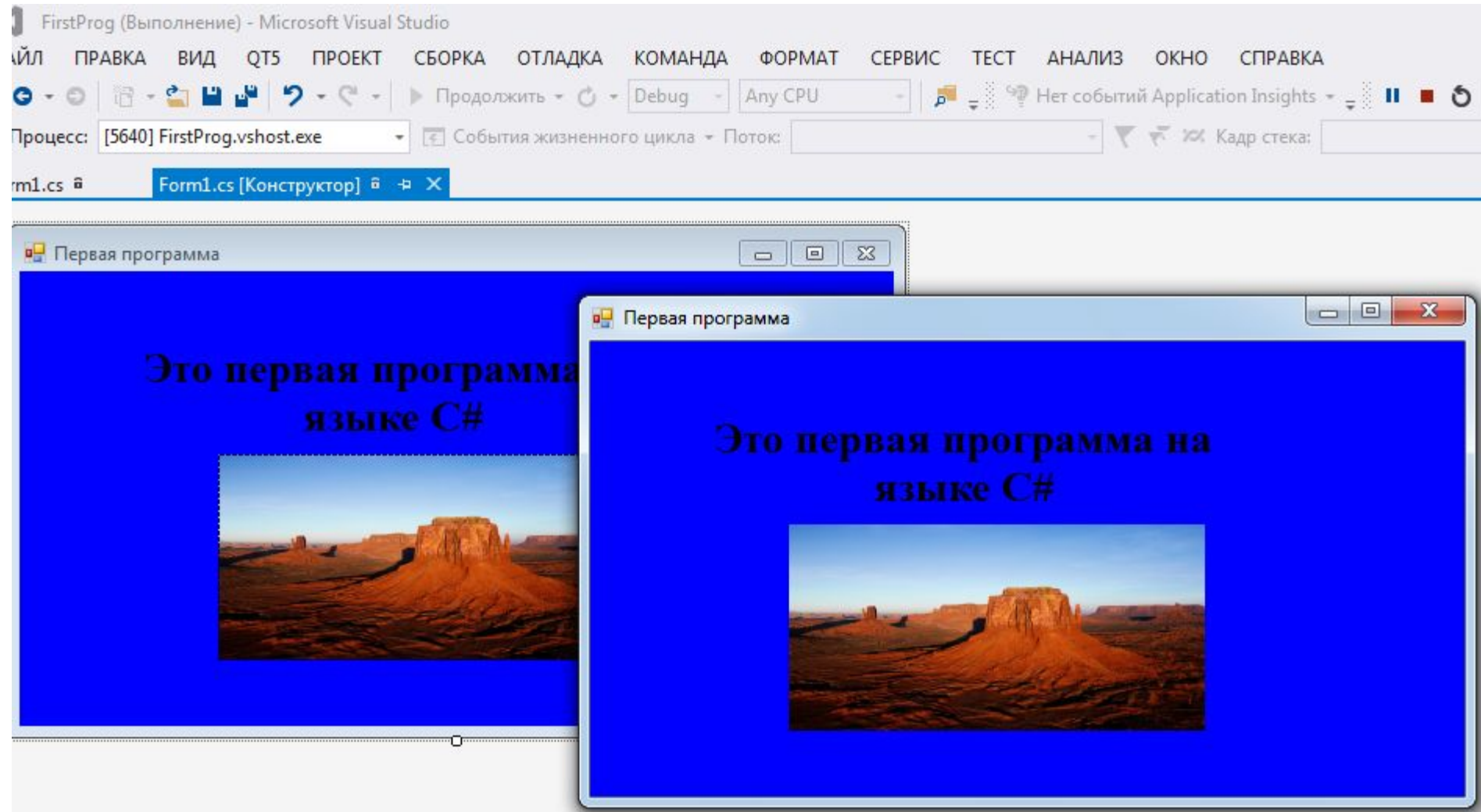


Свойства	
<b>pictureBox1</b> System.Windows.Forms.PictureBox	
Size 266; 132	
<b>Поведение</b>	
ContextMenuStrip	(нет)
Enabled	True
<b>SizeMode</b>	<b>StretchImage</b>
Visible	True
<b>Разработка</b>	
(Name)	pictureBox1
GenerateMember	True

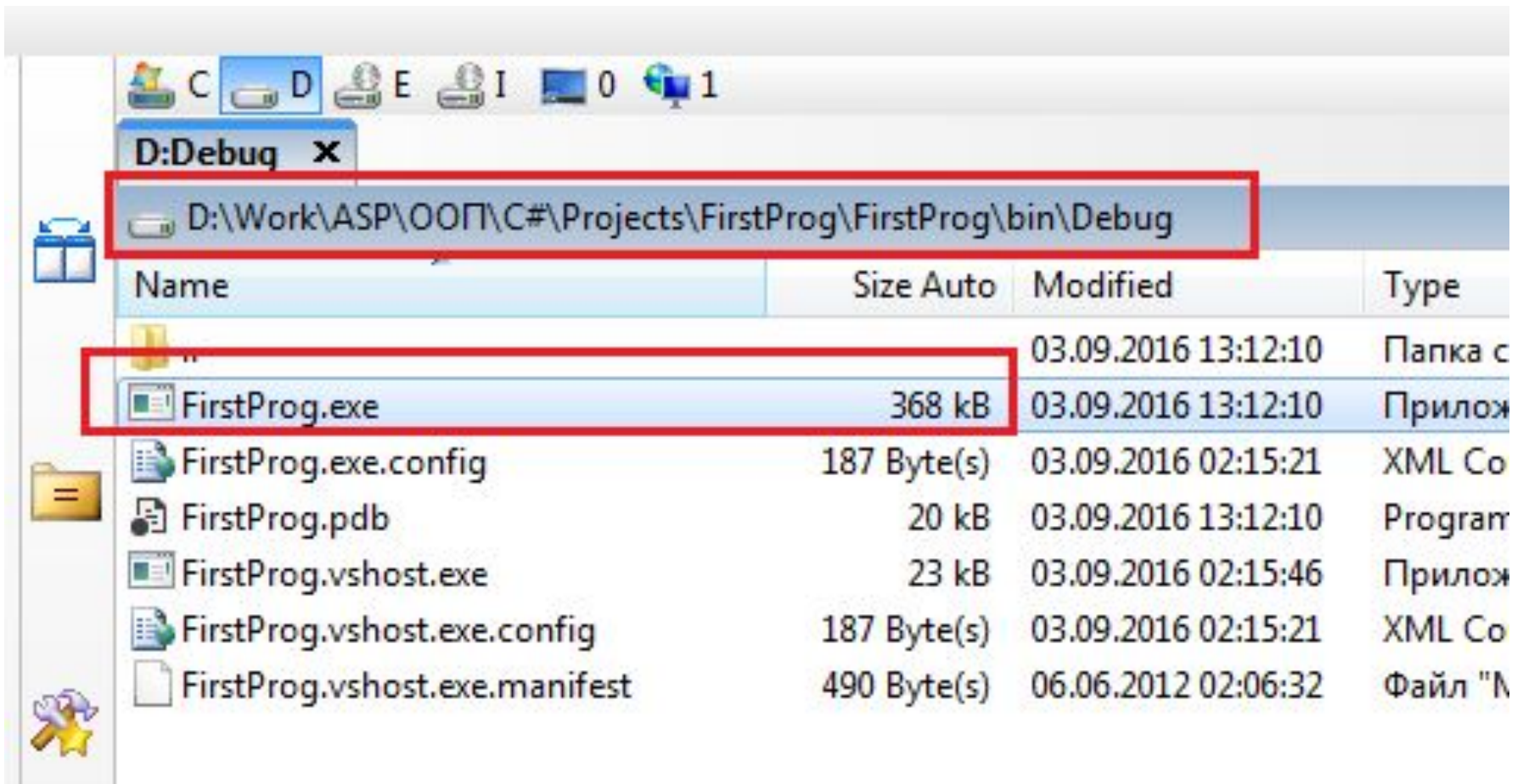
# Сохраните проект

- Выполните команду FILE | Save All, чтобы сохранить всё решение.
- В файле решения (расширение .sln) хранится имя и местонахождение проекта, а файл проекта (расширение .csproject) содержит имена и местонахождения всех файлов проекта.
- Если вы захотите снова открыть свой проект в *будущем*, просто откройте его файл .sln.

# Запуск программы



# Запуск программы



# Форма

- Форма (Form) является контейнером для элементов управления и компонентов.
- При перетаскивании значков с панели элементов на форму Visual Studio генерирует код создания объекта и инициализации его основных свойств.
- Этот код обновляется при изменении свойств элемента управления или компонента в IDE.
- Удаление элемента управления или компонента с формы приводит к удалению соответствующего сгенерированного кода.
- IDE хранит сгенерированный код в отдельном файле с использованием *частичных классов - классов, разбитых на несколько файлов* и собираемых в один класс компилятором. Вы можете написать этот код самостоятельно, но гораздо проще поручить <<черную работу>> Visual Studio

# Обработка событий

- Обычно пользователь взаимодействует с графическим интерфейсом приложения для выполнения операций, которые должны выполняться приложением.
- Например, когда вы пишете сообщение в почтовом клиенте, нажатие кнопки Send приказывает приложению отправить сообщение по заданным адресам электронной почты.
- Графический интерфейс *управляется событиями*. Когда пользователь взаимодействует с компонентом, событие заставляет программу выполнить операцию.
- К распространенным событиям, по которым приложение выполняет операцию, относятся нажатия кнопок, ввод текста в текстовых полях, выбор варианта из меню, закрытие окна и перемещение мыши.
- Со всеми элементами управления графического интерфейса связываются события; объекты других типов также могут связываться
- с событиями. Метод, выполняющий операцию в ответ на событие, называется *обработчиком события (event handler)*, а *общий процесс реагирования на события* называется *обработкой событий*.

# Обработка событий

The image shows a screenshot of the Visual Studio IDE. On the left, a window titled "Form1" is visible, which is currently empty. On the right, the "Обозреватель решений" (Solution Explorer) displays the project structure for "SimpleEvents". The file "SimpleEvents.cs" is selected, and its context menu is open. The menu items include "Действие при сборке" (Build Action), "Копировать в выходной каталог" (Copy to Output Directory), "Пространство имен специализации" (Specialized Name Space), "Специальный инструмент" (Tool), "Процесс" (Process), "Имя файла" (File Name), and "Полный путь" (Full Path). The "Имя файла" item is highlighted in blue, and the "Процесс" item is highlighted in red. Below the Solution Explorer, the "Свойства" (Properties) window is open, showing the "Свойства файла" (File Properties) for "SimpleEvents.cs". The "Действие при сборке" (Build Action) property is set to "Компилировать" (Compile). The "Копировать в выходной каталог" (Copy to Output Directory) property is set to "Не копировать" (Do not copy). The "Пространство имен специализации" (Specialized Name Space) property is set to "Пространство имен специализации" (Specialized Name Space). The "Специальный инструмент" (Tool) property is set to "Специальный инструмент" (Specialized Name Space). The "Процесс" (Process) property is set to "Процесс" (Specialized Name Space). The "Имя файла" (File Name) property is set to "SimpleEvents.cs". The "Полный путь" (Full Path) property is set to "D:\Work\ASP\OOP\C#\Projects\SimpleEvents\SimpleEvents.cs".

Обозреватель решений - поиск (Ctrl+ж)

- Решение "SimpleEvents" (проектов: 1)
  - SimpleEvents
    - Properties
    - References
    - App.config
    - Program.cs
    - SimpleEvents.cs
      - SimpleEvents.Designer.cs
      - SimpleEvents.resx
      - Form1

Анализ кода    Обозреват...    Командны...    Окно класс...    Ресур

Свойства

SimpleEvents.cs    Свойства файла

Действие при сборке    Компилировать

Копировать в выходной каталог    Не копировать

Пространство имен специализации

Специальный инструмент

Процесс

Имя файла    SimpleEvents.cs

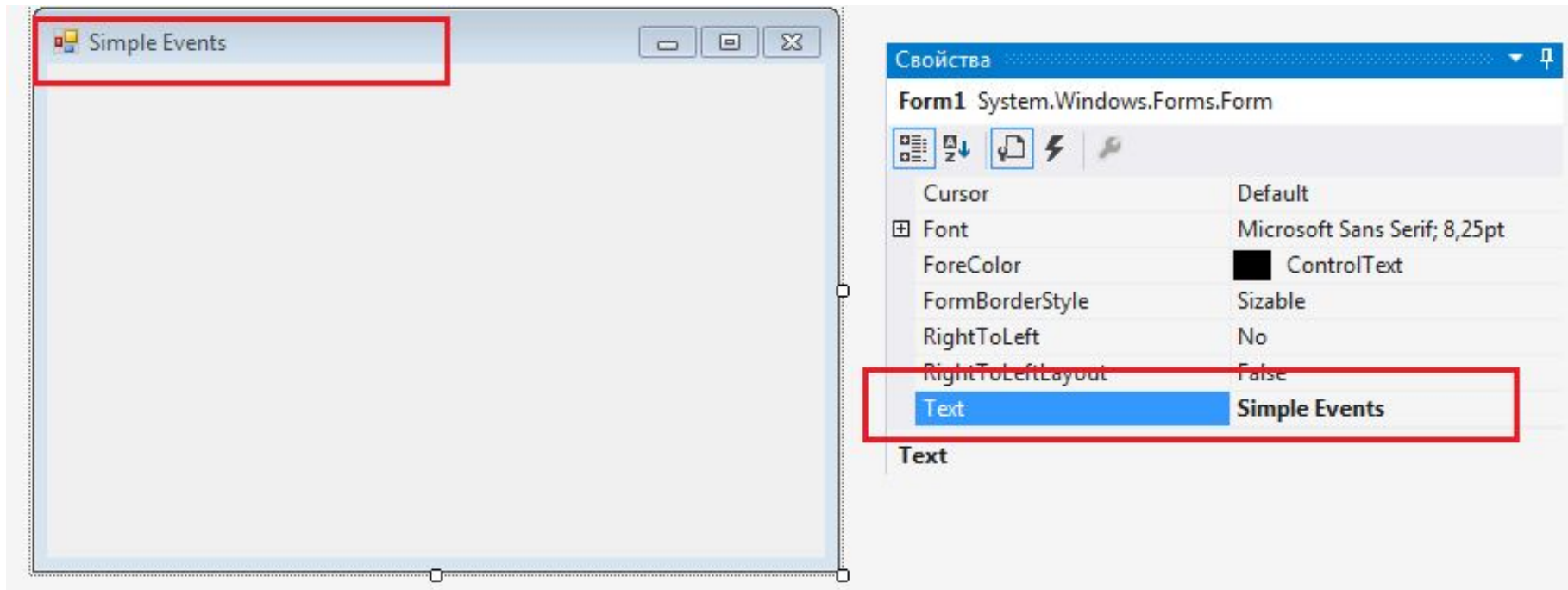
Полный путь    D:\Work\ASP\OOP\C#\Projects\SimpleEvents\SimpleEvents.cs

ывод

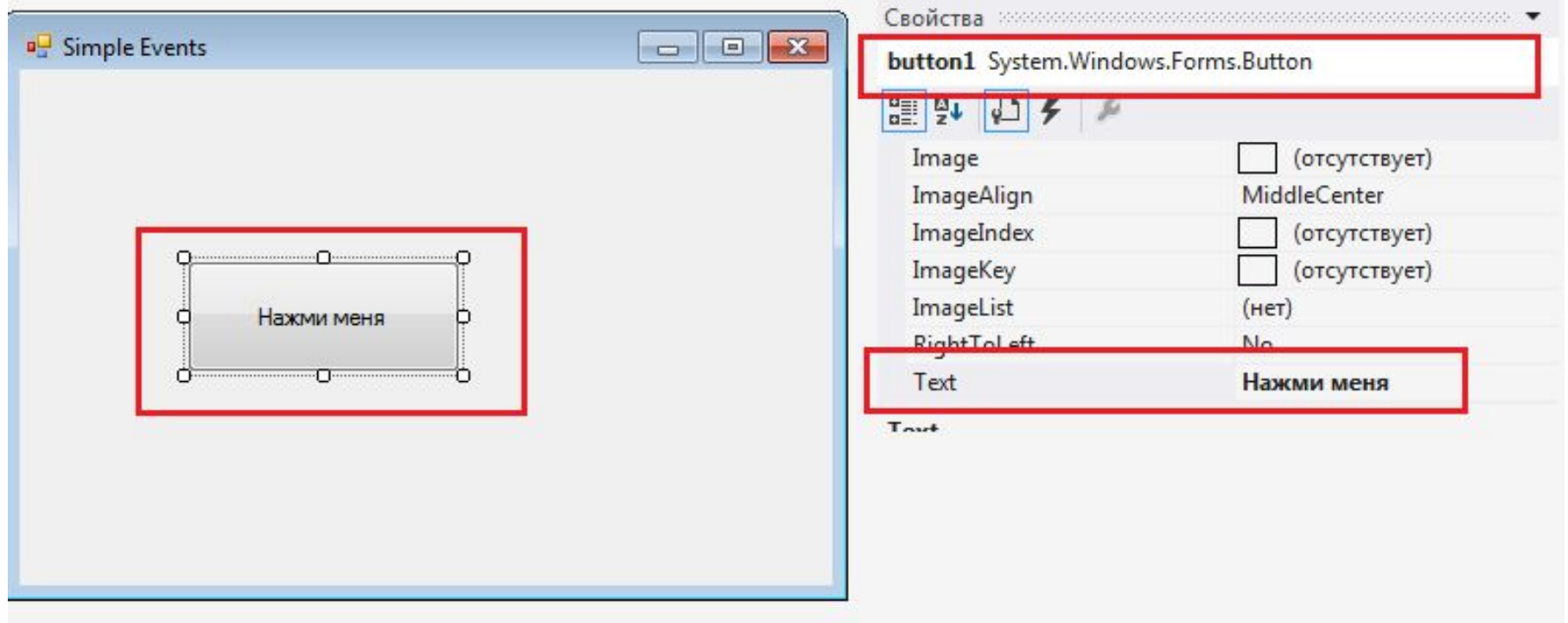
Показать выходные данные от:    Сборка



# Обработка событий



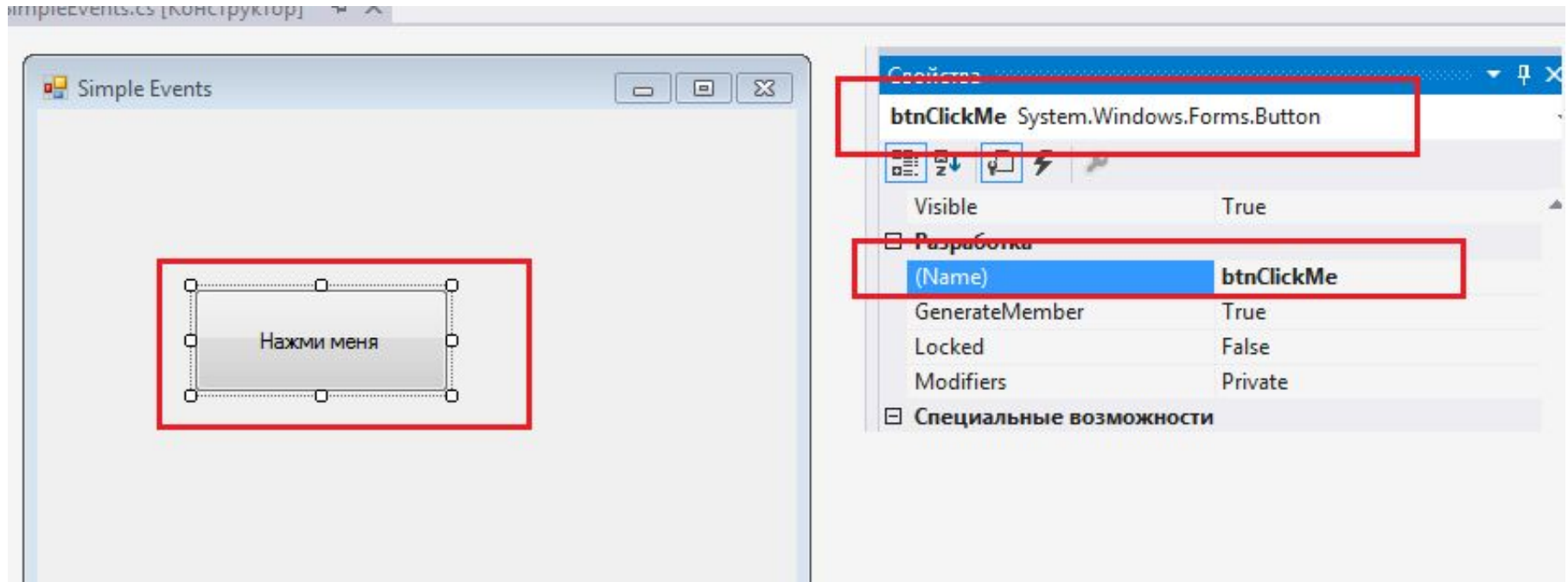
# Добавление кнопки на форму



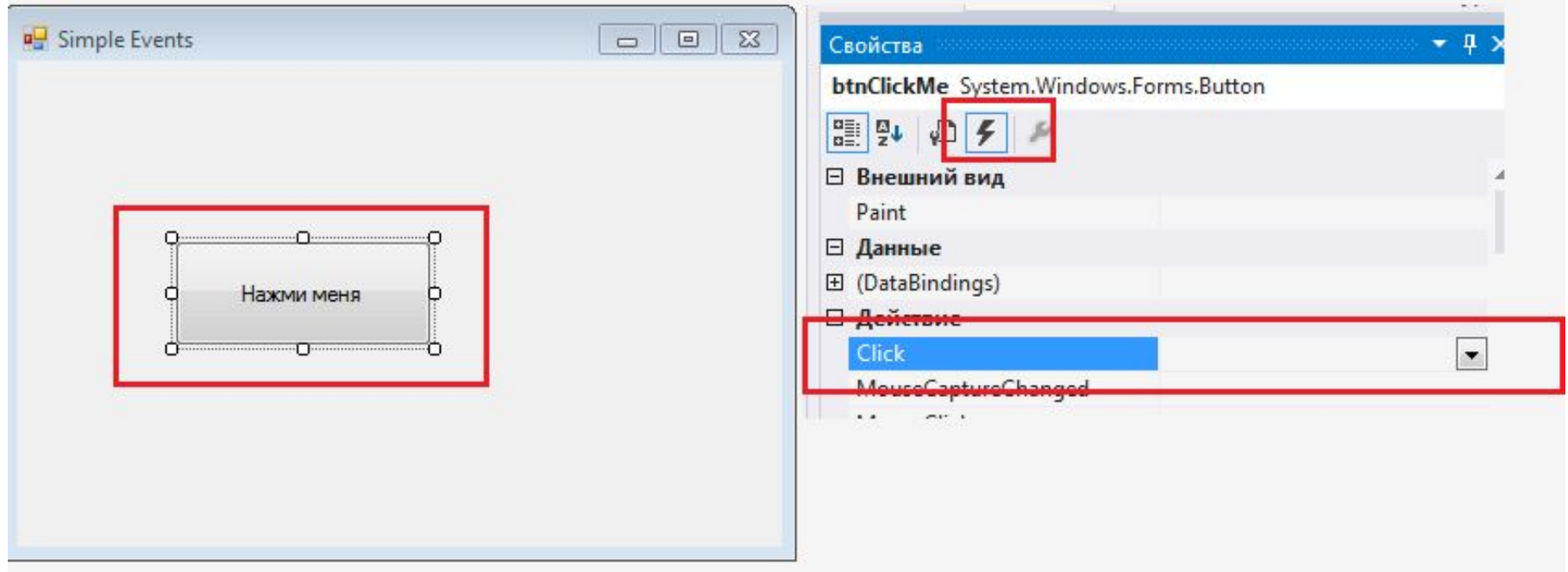
The image shows a screenshot of the Visual Studio IDE. On the left, a window titled "Simple Events" displays a form with a button. The button is a light gray rectangle with a dashed border and eight corner handles, containing the text "Нажми меня". A red rectangular box highlights the button. On the right, the "Свойства" (Properties) window is open, showing the properties for the selected button. The top of the Properties window shows "button1 System.Windows.Forms.Button". Below this, several properties are listed. The "Text" property is highlighted with a red box and is set to "Нажми меня".

Property	Value
Image	<input type="checkbox"/> (отсутствует)
ImageAlign	MiddleCenter
ImageIndex	<input type="checkbox"/> (отсутствует)
ImageKey	<input type="checkbox"/> (отсутствует)
ImageList	(нет)
RightToLeft	No
<b>Text</b>	<b>Нажми меня</b>

# Добавление кнопки на форму



# Добавление обработчика события Click для кнопки



# Добавление обработчика события Click для кнопки

```
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows.Forms;
```

```
namespace SimpleEvents
```

```
{
```

```
    public partial class Form1 : Form
```

```
    {
```

```
        public Form1()
```

```
        {
```

```
            InitializeComponent();
```

```
        }
```

```
        private void btnClickMe_Click(object sender, EventArgs e)
```

```
        {
```

```
            MessageBox.Show("Обработано событие нажатия кнопки");
```

```
        }
```

```
    }
```

```
}
```

## Добавление обработчика события Click для кнопки

- Когда пользователь щелкает на кнопке, наше приложение должно вывести окно сообщения (MessageBox).
- Для этого следует назначить обработчик события Click объекта Button.
- Если сделать двойной щелчок на кнопке, в *программный код* включается следующий пустой обработчик события:

```
private void clickButton_Click( object sender, EventArgs e)
{
}
```
- IDE строит имена методов обработчиков событий по схеме имяОбъекта\_имяСобытия (например, clickButton\_Click). Обработчик clickButton\_Click обрабатывает событие щелчка (click) на элементе управления clickButton.

## Параметры обработчиков событий

- Каждый обработчик события при вызове получает два параметра.
- Первый – обычно с именем **sender** - содержит ссылку на объект, сгенерировавший событие.
- Вторым параметром является ссылка на объект **EventArgs** (или объект класса, производного от EventArgs ), обычно с именем **e**.
- В этом объекте содержится дополнительная информация о возникшем событии. EventArgs - базовый класс для всех классов, представляющих

## Автоматически генерируемый код графического интерфейса

- Visual Studio помещает автоматически сгенерированный код графического интерфейса в файл *Designer.cs* формы (*SimpleEventExampleForm.Designer.cs* в данном примере).
- Так как код создается и поддерживается Visual Studio, обычно вам не приходится его просматривать. Более того, для построения приложений GUI понимать большую часть приведенного кода вообще не обязательно.



# Автоматически генерируемый код графического интерфейса

The image shows a screenshot of the Visual Studio IDE. On the left, the code editor displays the source code for `SimpleEvents.cs`. The code is as follows:

```
namespace SimpleEvents
{
    partial class Form1
    {
        /// <summary>
        /// Требуется переменная конструктора.
        /// </summary>
        private System.ComponentModel.IContainer components;

        /// <summary>
        /// Освободить все используемые ресурсы.
        /// </summary>
        /// <param name="disposing">истинно, если управляет элементами.
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                // ...
            }
        }
    }
}
```

On the right, the Solution Explorer shows the project structure for "SimpleEvents" (1 project). The files listed are:

- SimpleEvents
- Properties
- References
- App.config
- Program.cs
- SimpleEvents.cs
- SimpleEvents.Designer.cs
- SimpleEvents.resx
- Form1

The `SimpleEvents.Designer.cs` file is highlighted with a red rectangle, indicating it is the focus of the automatic code generation process.

# Автоматически генерируемый код графического интерфейса

#region Код, автоматически созданный конструктором форм Windows

```
private void InitializeComponent()
{
    this.btnClickMe = new System.Windows.Forms.Button();
    this.SuspendLayout();
    //
    // btnClickMe
    //
    this.btnClickMe.Location = new System.Drawing.Point(85, 97);
    this.btnClickMe.Name = "btnClickMe";
    this.btnClickMe.Size = new System.Drawing.Size(139, 57);
    this.btnClickMe.TabIndex = 0;
    this.btnClickMe.Text = "Нажми меня";
    this.btnClickMe.UseVisualStyleBackColor = true;
    this.btnClickMe.Click += new System.EventHandler(this.btnClickMe_Click);
    //
    // Form1
    //
}
```

## Свойства и макет элемента управления

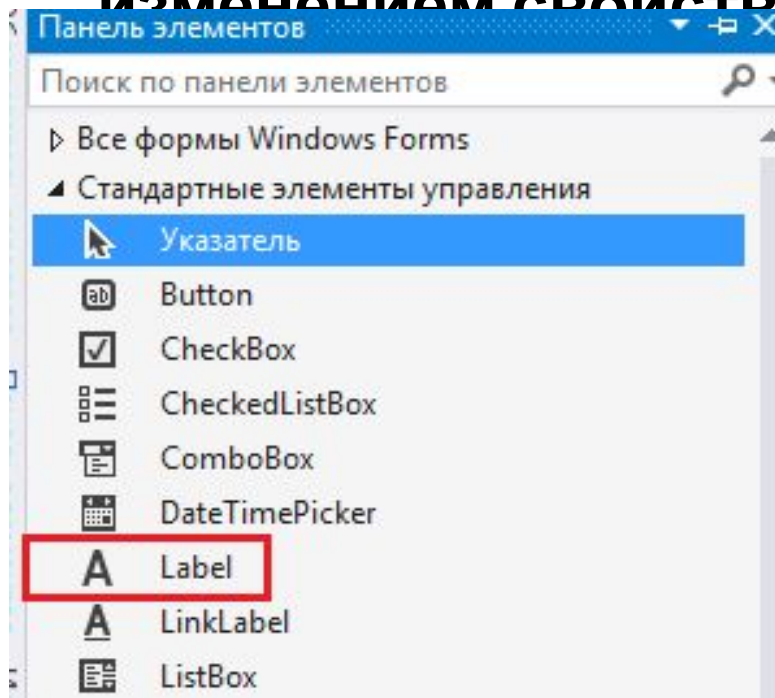
- Элементы управления наследуют от класса Control (пространство имен System. Windows. Forms). На рисунке перечислены некоторые свойства и методы класса Control, которые могут задаваться для многих элементов управления.
- Например, свойство **Text** определяет текст, выводимый на элементе управления. Местонахождение текста зависит от конкретного элемента: для формы текст выводится в заголовке, а для кнопки - непосредственно на поверхности.

# Свойства и макет элемента управления

Свойства и методы класса Control	Описание
Часто используемые свойства	
BackColor	Цвет фона
BackgroundImage	Фоновое изображение
Enabled	Признак доступности элемента управления (то есть возможности взаимодействия с ним со стороны пользователя). Недоступный (заблокированный) элемент обычно выделяется серым цветом
Focused	Признак наличия у элемента фокуса ввода (свойство доступно только во время выполнения)
Font	Шрифт, используемый для вывода текста элемента
ForeColor	Основной цвет элемента управления; обычно определяет цвет текста, определяемого свойством Text
TabIndex	Индекс элемента управления в последовательности перебора. При нажатии клавиши Tab фокус ввода передается между элементами в последовательности, которая определяется разработчиком
TabStop	Если свойство равно true, элемент управления может получить фокус при нажатии клавиши Tab
Text	Текст, связанный с элементом. Местонахождение и оформление текста зависит от типа элемента
Visible	Признак видимости элемента

## Надписи, текстовые поля и кнопки

- *Надписи используются для вывода текстовой информации ( и возможно, графики) и определяются классом **Label**, производным от Control.*
- **Текст, выводимый в надписи, не может напрямую изменяться пользователем (но может быть изменен на программном уровне изменением свойства **Text** объекта **Label**).**



# Надписи, текстовые поля и кнопки

Часто используемые свойства Label	Описание
Font	Шрифт текста надписи
Text	Текст надписи
TextAlign	Режим выравнивания текста в элементе управления — горизонтальный (по левому краю, по центру, по правому краю) и вертикальный (по верхнему краю, по центру, по нижнему краю). По умолчанию используется выравнивание по верхнему и левому краю

# Надписи, текстовые поля и кнопки

Часто используемые свойства Label	Описание
Font	Шрифт текста надписи
Text	Текст надписи
TextAlign	Режим выравнивания текста в элементе управления — горизонтальный (по левому краю, по центру, по правому краю) и вертикальный (по верхнему краю, по центру, по нижнему краю). По умолчанию используется выравнивание по верхнему и левому краю