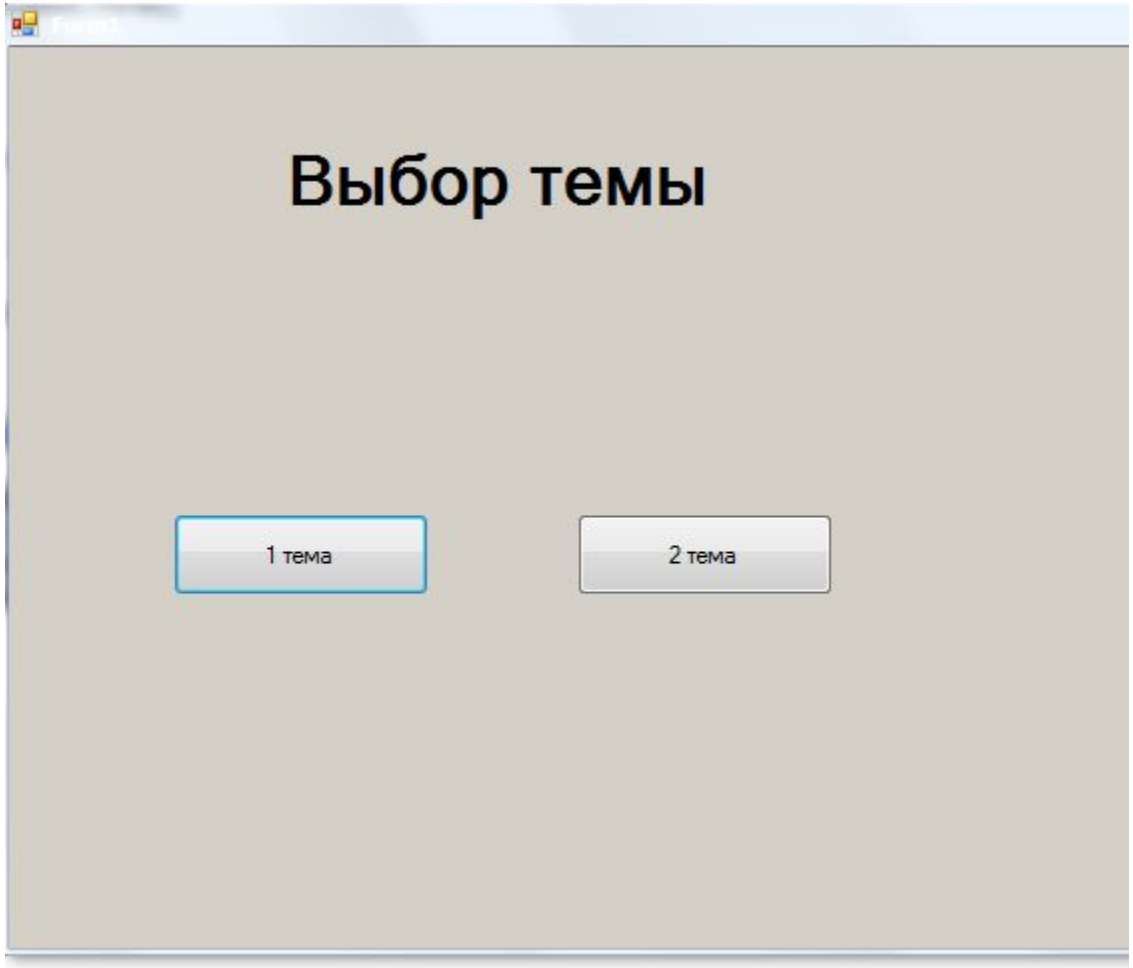
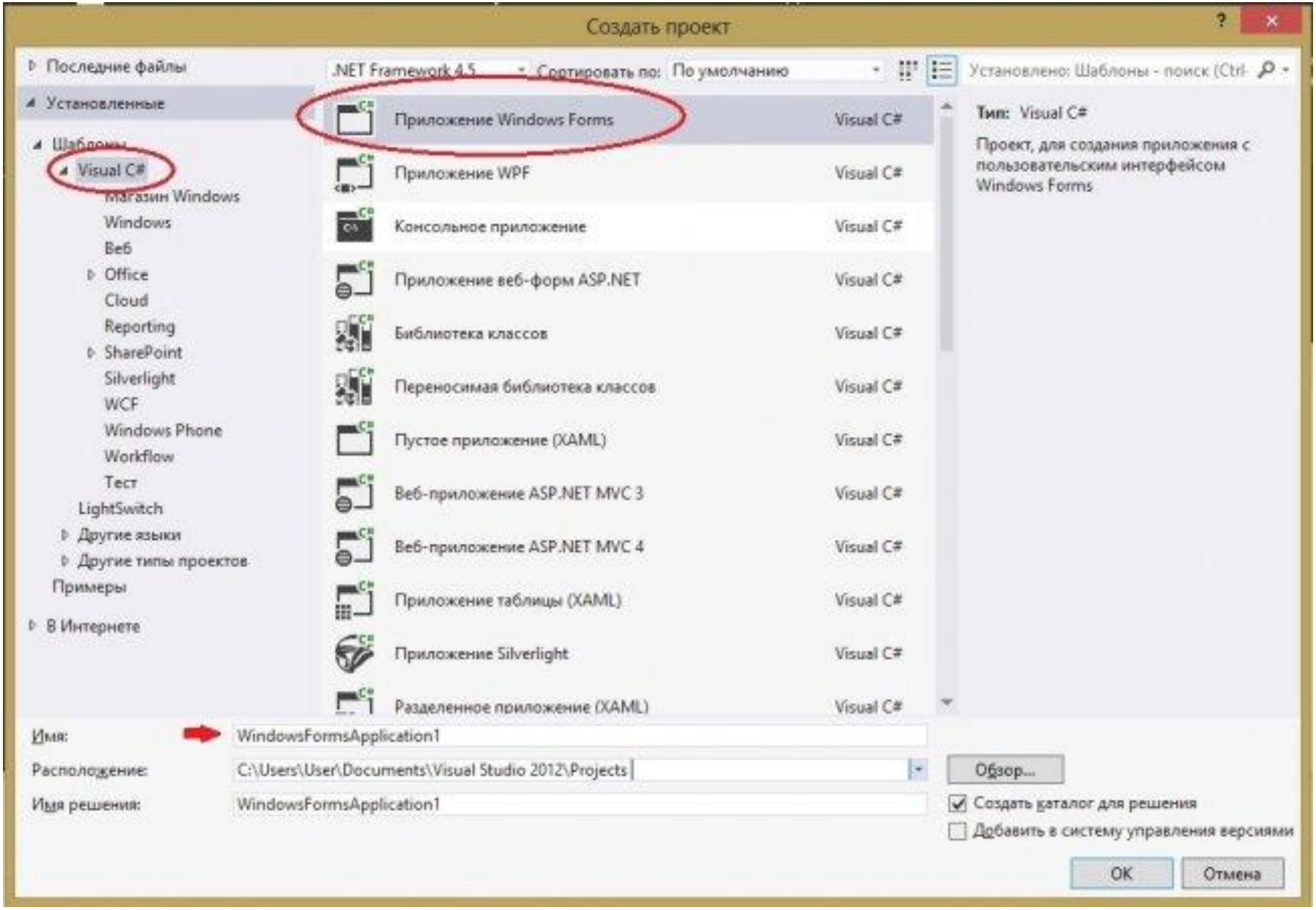
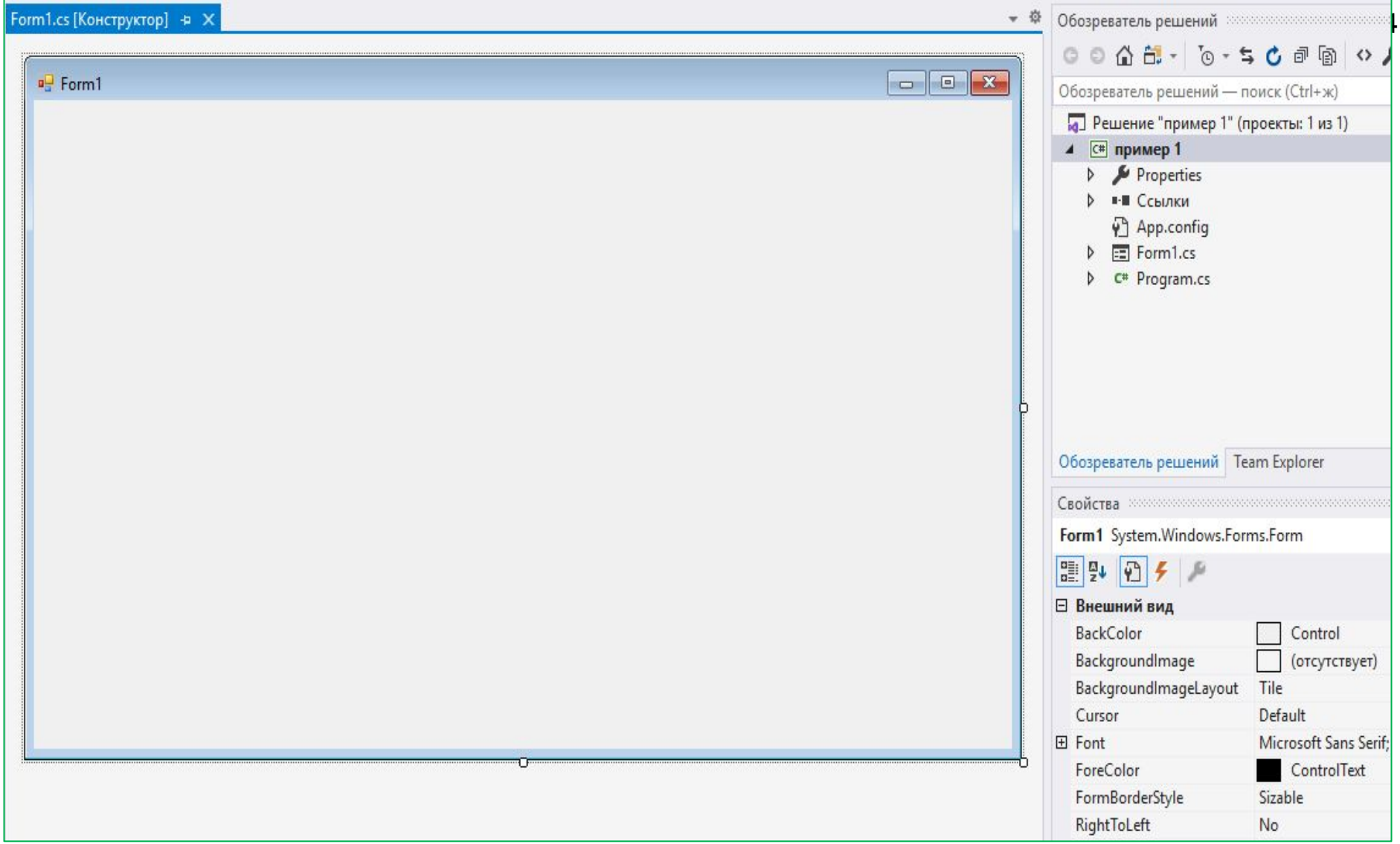
Программирование на языке Си#

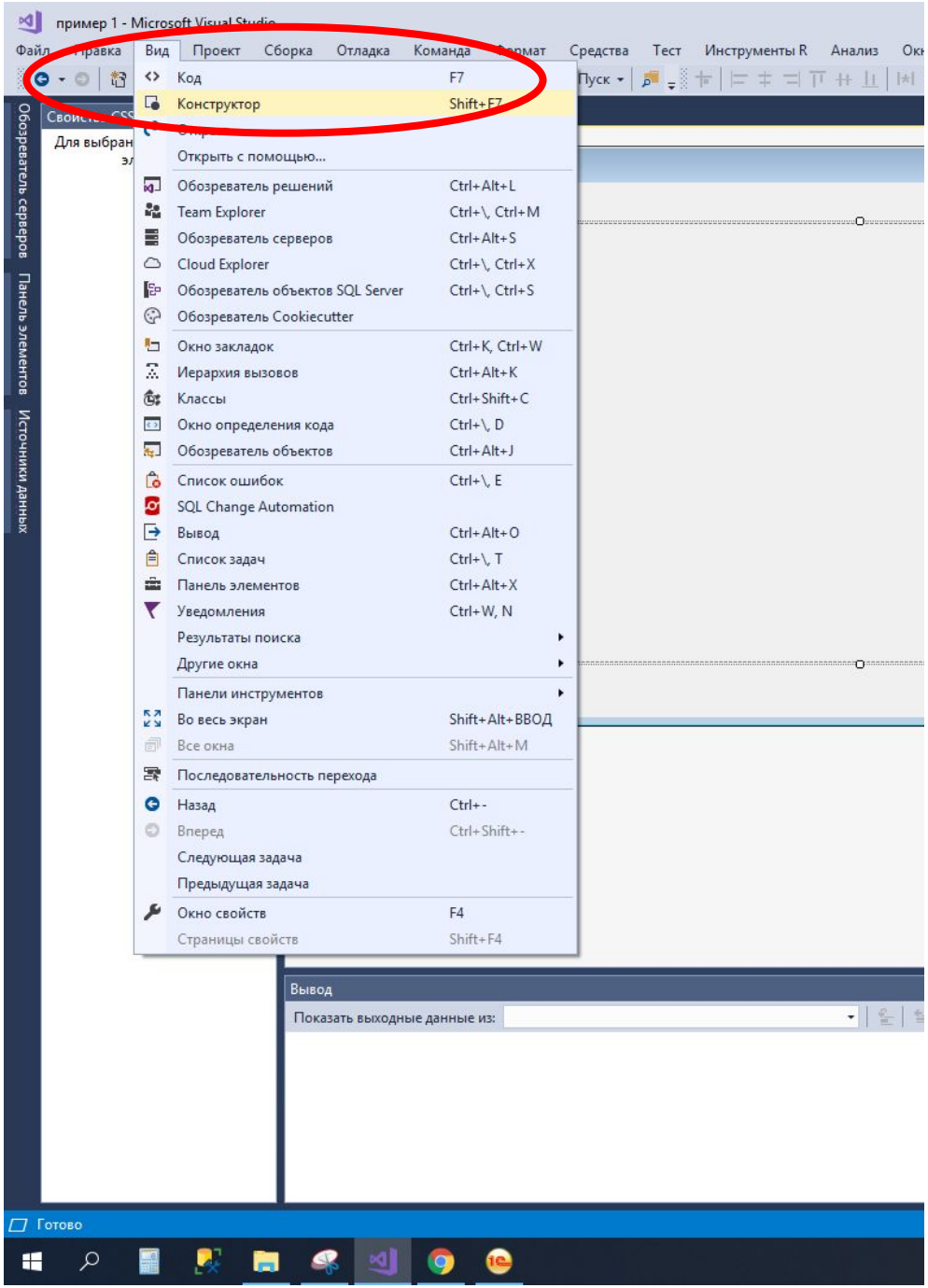
Тема 9. ФОРМА

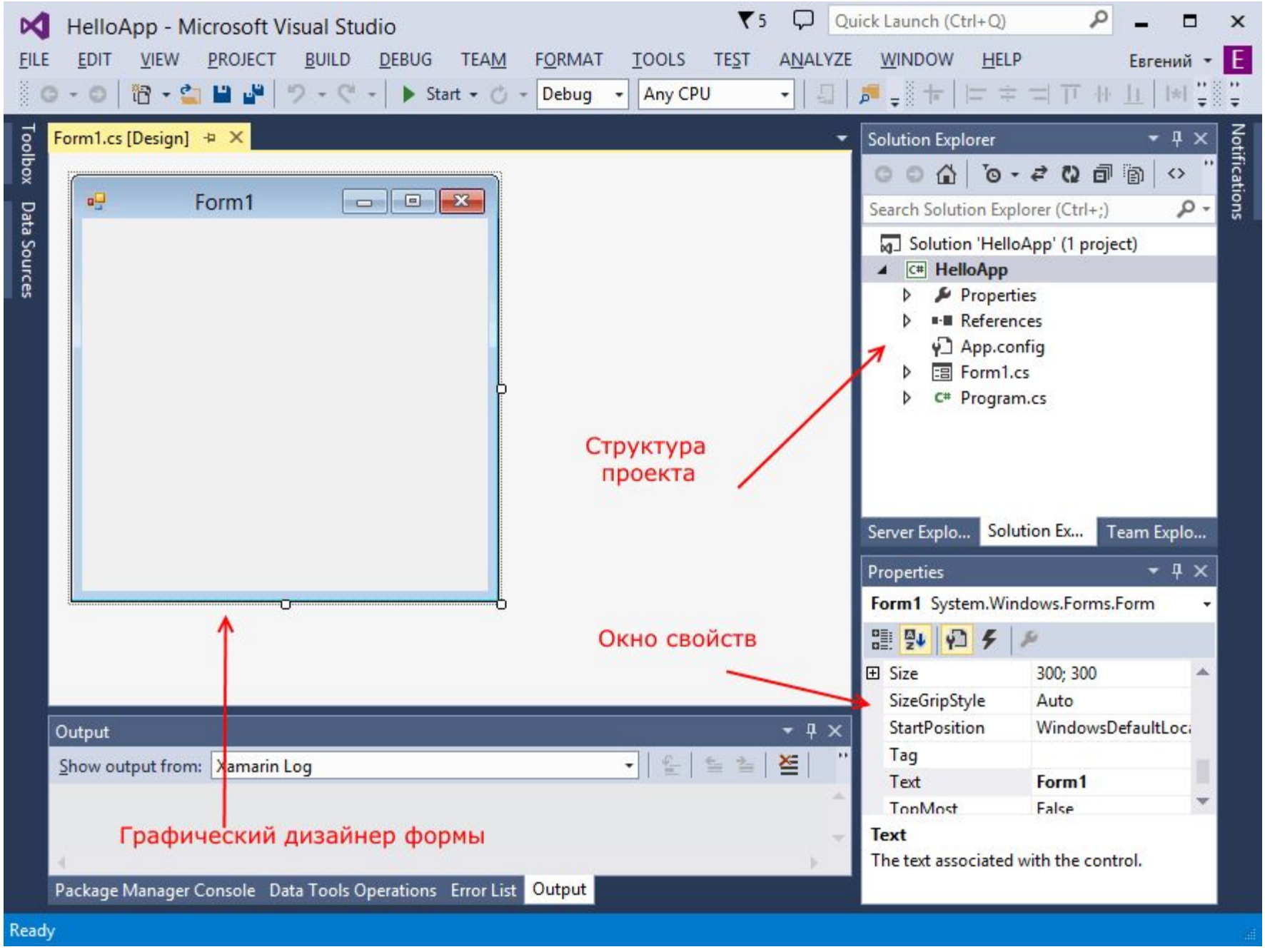
Работа с конструктором Формы

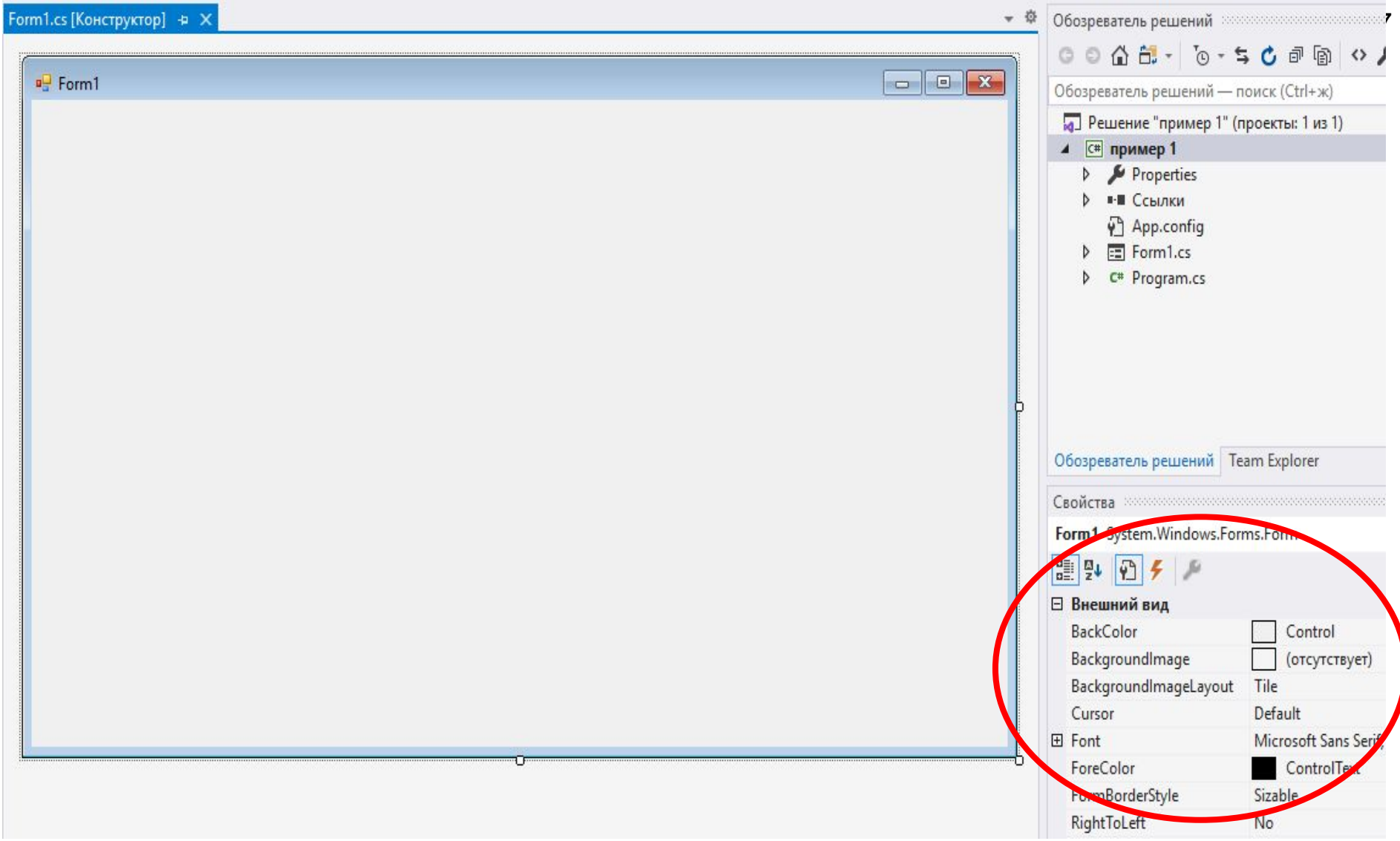


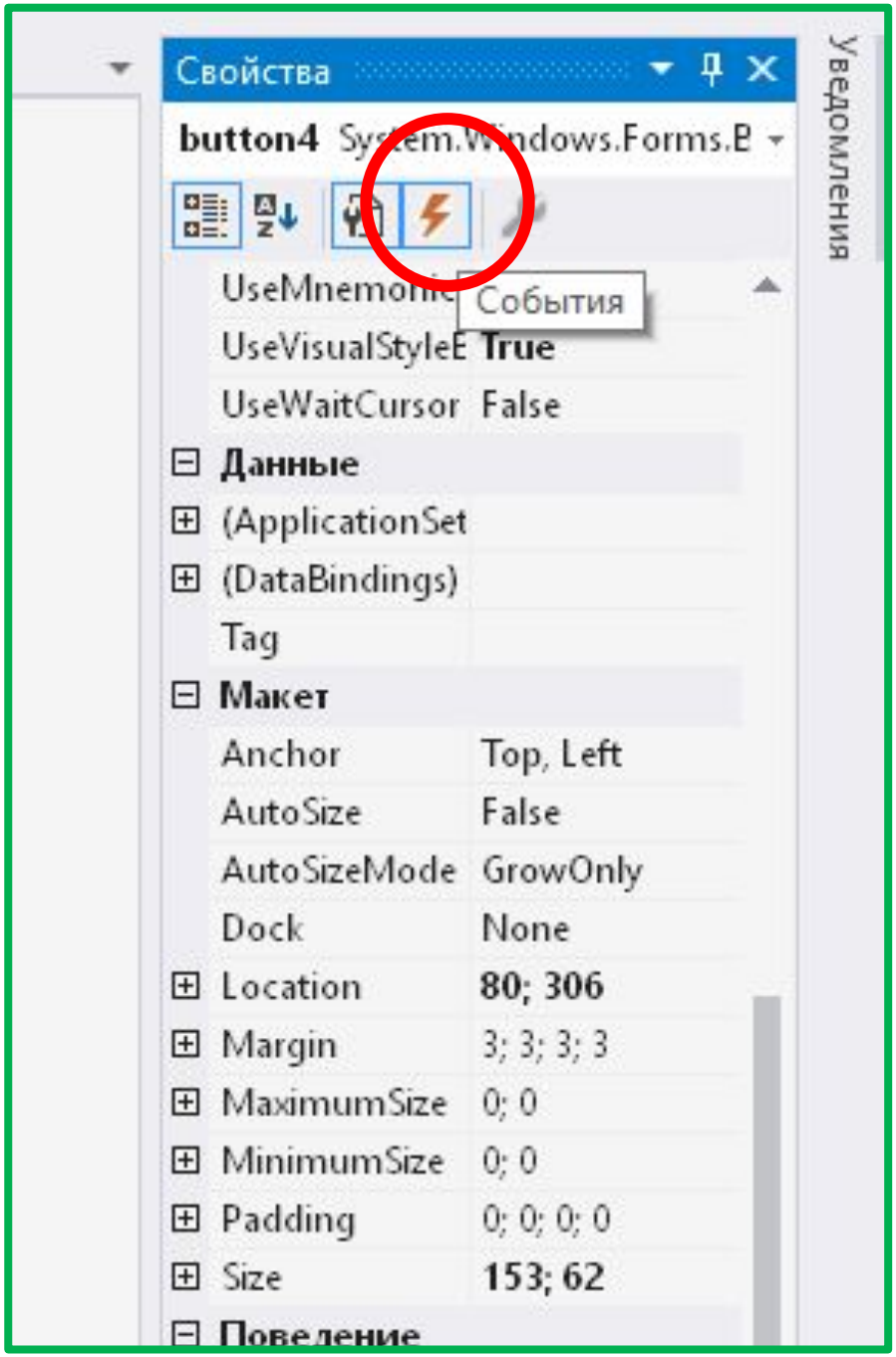

Создать проект
Последние файлы
.NET Framework 4.5
Сортировать по:
По умолчанию
Установлено: Шаблоны - поиск (Ctrl-
Установленные
Шаблоны
Visual C#
Магазин Windows
Windows
Веб
Office
Cloud
Reporting
SharePoint
Silverlight
WCF
Windows Phone
Workflow
Тест
LightSwitch
Другие языки
Другие типы проектов
Примеры
В Интернете
Приложение Windows Forms
Visual C#
Приложение WPF
Visual C#
Консольное приложение
Visual C#
Приложение веб-форм ASP.NET
Visual C#
Библиотека классов
Visual C#
Переносимая библиотека классов
Visual C#
Пустое приложение (XAML)
Visual C#
Веб-приложение ASP.NET MVC 3
Visual C#
Веб-приложение ASP.NET MVC 4
Visual C#
Приложение таблицы (XAML)
Visual C#
Приложение Silverlight
Visual C#
Разделенное приложение (XAML)
Visual C#
Тип: Visual C#
Проект, для создания приложения с пользовательским интерфейсом Windows Forms
Имя:
WindowsFormsApplication1
Расположение:
C:\Users\User\Documents\Visual Studio 2012\Projects
Обзор...
Имя решения:
WindowsFormsApplication1
Создать каталог для решения
Добавить в систему управления версиями
OK
Отмена

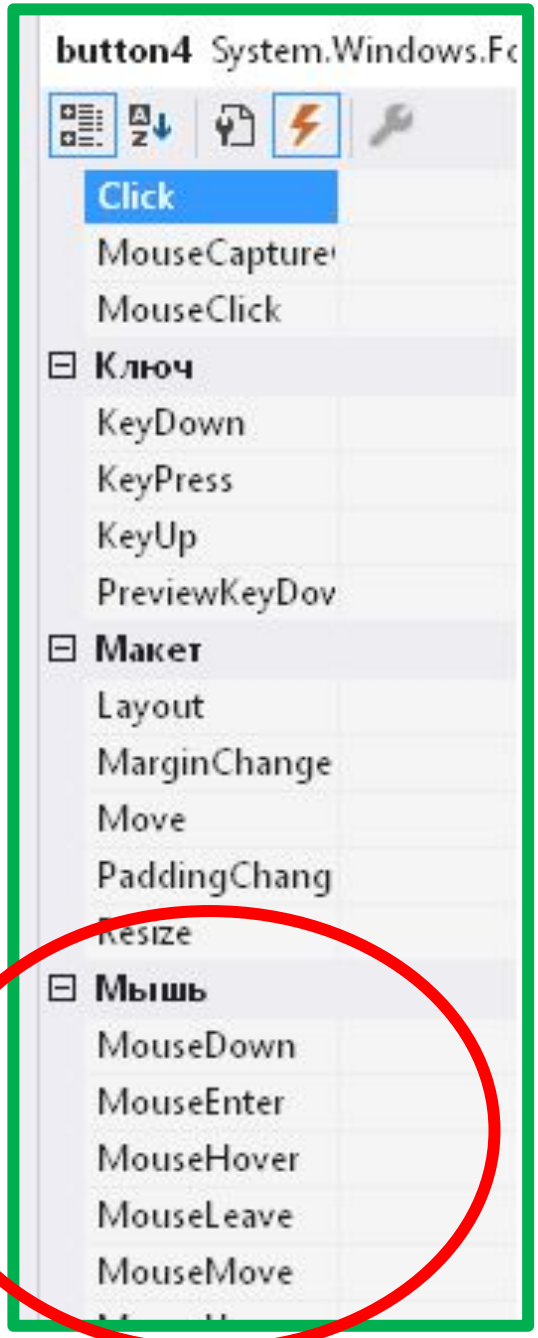

Form1.cs [Конструктор]
Form1
Обозреватель решений
Обозреватель решений — поиск (Ctrl+ж)
Решение "пример 1" (проекты: 1 из 1)
пример 1
Properties
Ссылки
App.config
Form1.cs
Program.cs
Обозреватель решений
Team Explorer
Свойства
Form1 System.Windows.Forms.Form
Внешний вид
BackColor
Control
BackgroundImage
(отсутствует)
BackgroundImageLayout
Tile
Cursor
Default
Font
Microsoft Sans Serif;
ForeColor
ControlText
FormBorderStyle
Sizable
RightToLeft
No

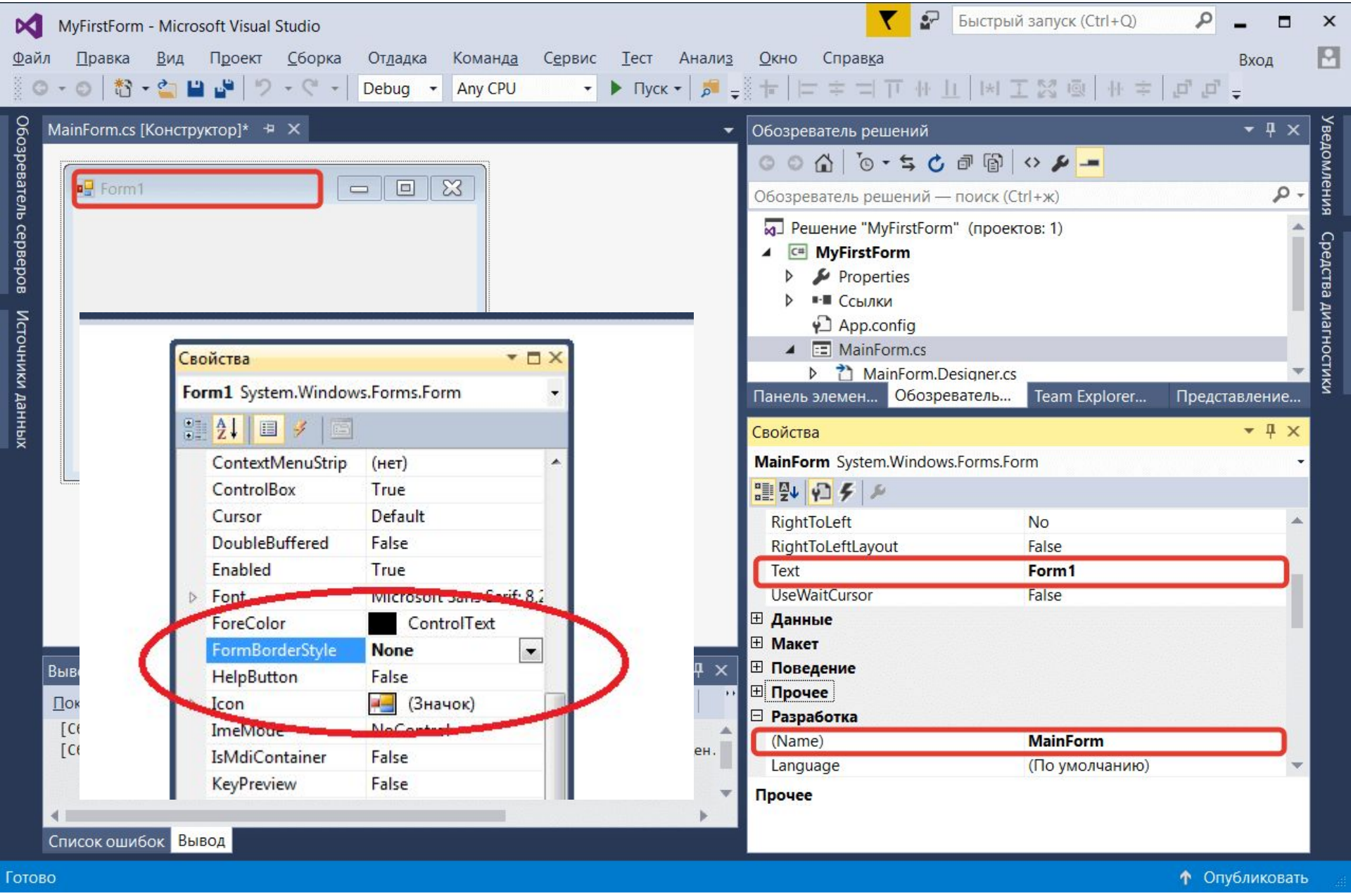

пример 1 - Microsoft Visual Studio
Файл
Правка
Вид
Проект
Сборка
Отладка
Команда
Формат
Средства
Тест
Инструменты R
Анализ
Пуск
Код F7
Конструктор Shift+F7
Открыть с помощью...
Обозреватель решений Ctrl+Alt+L
Team Explorer Ctrl+\, Ctrl+M
Обозреватель серверов Ctrl+Alt+S
Cloud Explorer Ctrl+\, Ctrl+X
Обозреватель объектов SQL Server Ctrl+\, Ctrl+S
Обозреватель Cookiecutter
Окно закладок Ctrl+K, Ctrl+W
Иерархия вызовов Ctrl+Alt+K
Классы Ctrl+Shift+C
Окно определения кода Ctrl+\, D
Обозреватель объектов Ctrl+Alt+J
Список ошибок Ctrl+\, E
SQL Change Automation
Вывод Ctrl+Alt+O
Список задач Ctrl+\, T
Панель элементов Ctrl+Alt+X
Уведомления Ctrl+W, N
Результаты поиска
Другие окна
Панели инструментов
Во весь экран Shift+Alt+ВВОД
Все окна Shift+Alt+M
Последовательность перехода
Назад Ctrl+-
Вперед Ctrl+Shift+-
Следующая задача
Предыдущая задача
Окно свойств F4
Страницы свойств Shift+F4
Обозреватель серверов
Панель элементов
Источники данных
Вывод
Показать выходные данные из:
Готово

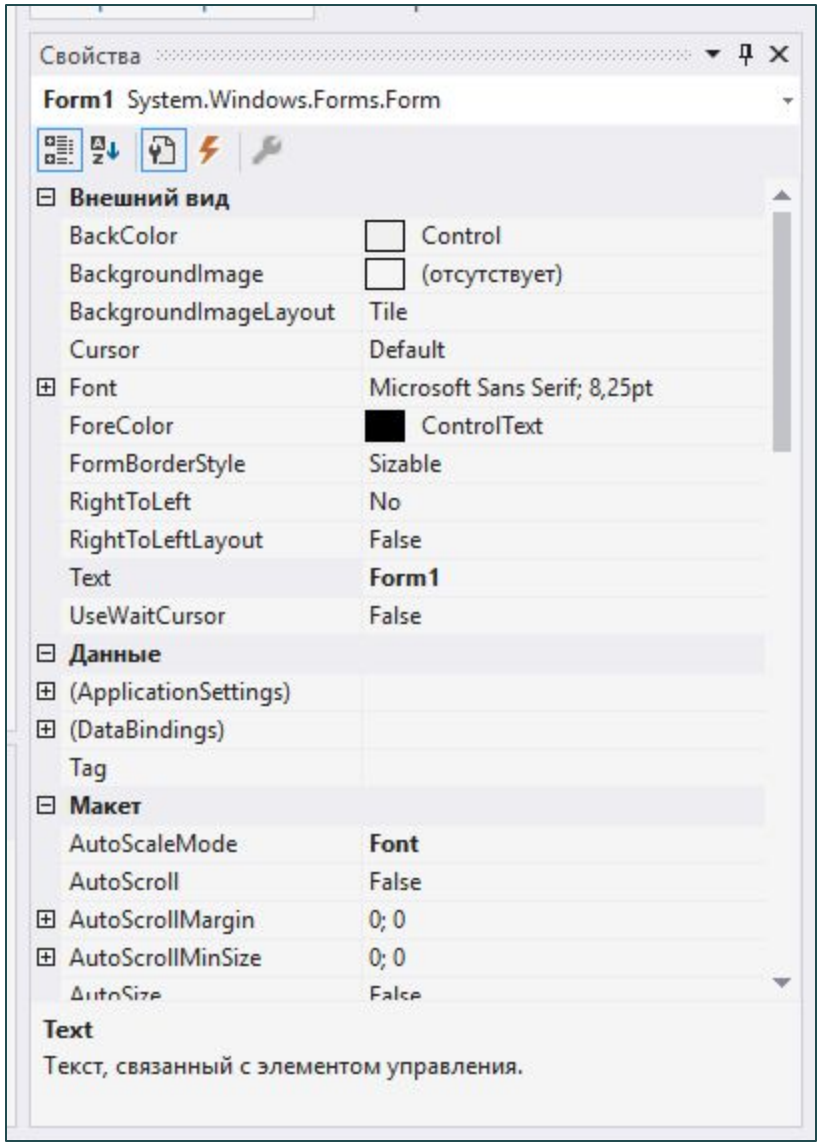

HelloApp - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM FORMAT TOOLS TEST ANALYZE WINDOW HELP
Евгений
Start
Debug
Any CPU
Toolbox
Data Sources
Form1.cs [Design]
Form1
Структура проекта
Окно свойств
Графический дизайнер формы
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'HelloApp' (1 project)
HelloApp
Properties
References
App.config
Form1.cs
Program.cs
Server Explo...
Solution Ex...
Team Explo...
Notifications
Properties
Form1 System.Windows.Forms.Form
Size 300; 300
SizeGripStyle Auto
StartPosition WindowsDefaultLoc
Tag
Text Form1
TopMost False
Text
The text associated with the control.
Output
Show output from: Xamarin Log
Package Manager Console
Data Tools Operations
Error List
Output
Ready

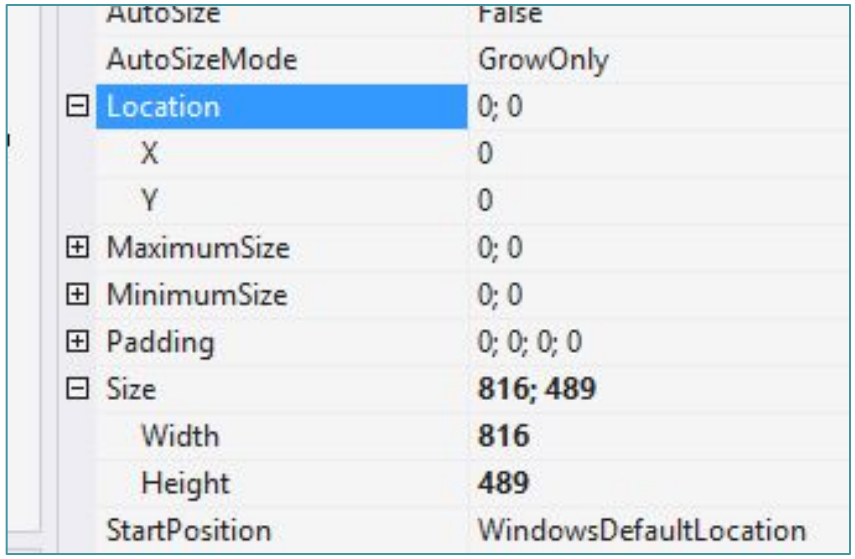

Form1.cs [Конструктор]
Form1
Обозреватель решений
Обозреватель решений — поиск (Ctrl+ж)
Решение "пример 1" (проекты: 1 из 1)
пример 1
Properties
Ссылки
App.config
Form1.cs
Program.cs
Обозреватель решений
Team Explorer
Свойства
Form1 System.Windows.Forms.Form
Внешний вид
BackColor
Control
BackgroundImage
(отсутствует)
BackgroundImageLayout
Tile
Cursor
Default
Font
Microsoft Sans Serif
ForeColor
ControlText
FormBorderStyle
Sizable
RightToLeft
No

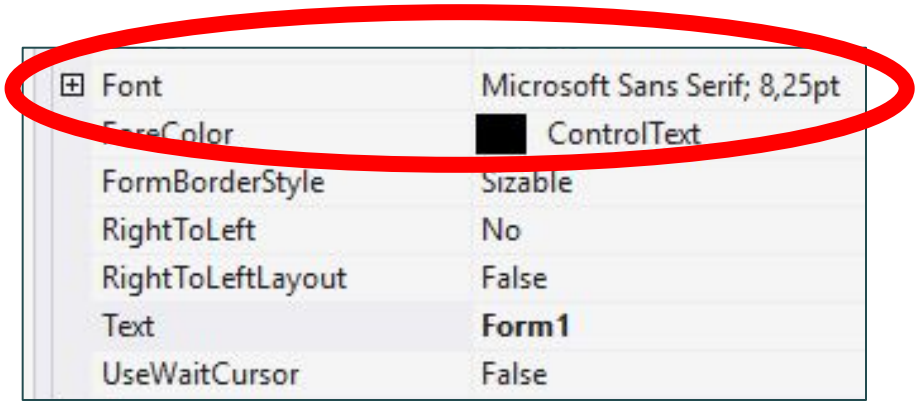

Свойства
button4 System.Windows.Forms.B
Уведомления
События
UseMnemonic
UseVisualStyleE True
UseWaitCursor False
Данные
(ApplicationSet
(DataBindings)
Tag
Макет
Anchor Top, Left
AutoSize False
AutoSizeMode GrowOnly
Dock None
Location 80; 306
Margin 3; 3; 3; 3
MaximumSize 0; 0
MinimumSize 0; 0
Padding 0; 0; 0; 0
Size 153; 62
Поведение

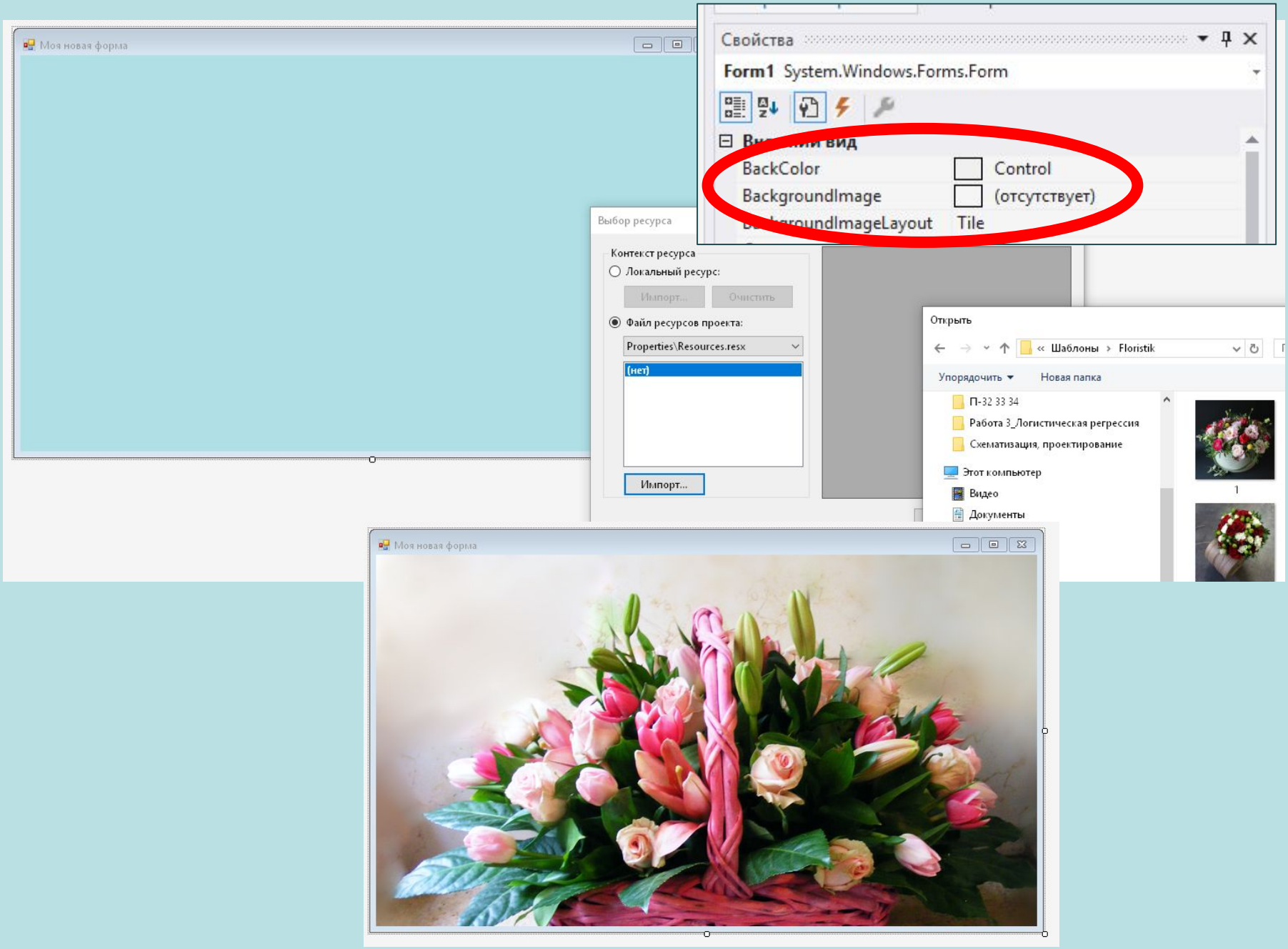

button4 System.Windows.Fc
Click
MouseCapture
MouseClick
Ключ
KeyDown
KeyPress
KeyUp
PreviewKeyDov
Макет
Layout
MarginChange
Move
PaddingChang
Resize
Мышь
MouseDown
MouseEnter
MouseHover
MouseLeave
MouseMove

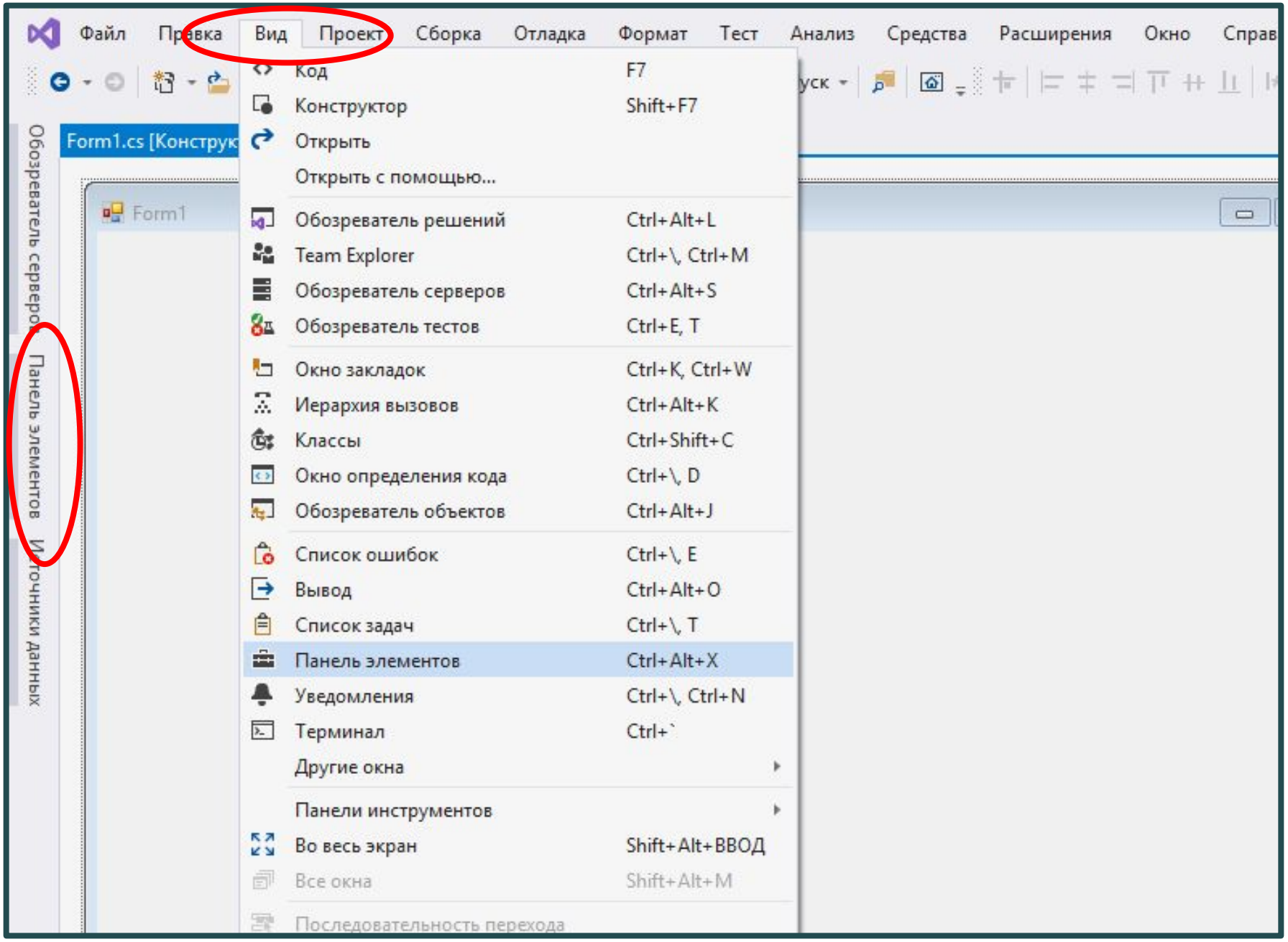

MyFirstForm - Microsoft Visual Studio
Быстрый запуск (Ctrl+Q)
Файл Правка Вид Проект Сборка Отладка Команда Сервис Тест Анализ Окно Справка
Вход
Debug
Any CPU
Пуск
MainForm.cs [Конструктор]*
Form1
Свойства
Form1 System.Windows.Forms.Form
ContextMenuStrip (нет)
ControlBox True
Cursor Default
DoubleBuffered False
Enabled True
Font
ForeColor ControlText
FormBorderStyle None
HelpButton False
Icon (Значок)
IsMdiContainer False
KeyPreview False
Обозреватель решений
Обозреватель решений — поиск (Ctrl+ж)
Решение "MyFirstForm" (проектов: 1)
MyFirstForm
Properties
Ссылки
App.config
MainForm.cs
MainForm.Designer.cs
Панель элемен...
Обозреватель...
Team Explorer...
Представление...
Свойства
MainForm System.Windows.Forms.Form
RightToLeft No
RightToLeftLayout False
Text Form1
UseWaitCursor False
Данные
Макет
Поведение
Прочее
Разработка
(Name) MainForm
Language (По умолчанию)
Прочее
Обозреватель серверов
Источники данных
Уведомления
Средства диагностики
Список ошибок
Вывод
Готово
Опубликовать


Свойства
Form1 System.Windows.Forms.Form
Внешний вид
BackColor Control
BackgroundImage (отсутствует)
BackgroundImageLayout Tile
Cursor Default
Font Microsoft Sans Serif; 8,25pt
ForeColor ControlText
FormBorderStyle Sizable
RightToLeft No
RightToLeftLayout False
Text Form1
UseWaitCursor False
Данные
(ApplicationSettings)
(DataBindings)
Tag
Макет
AutoScaleMode Font
AutoScroll False
AutoScrollMargin 0; 0
AutoScrollMinSize 0; 0
AutoSize False
Text
Текст, связанный с элементом управления.


AutoSize False
AutoSizeMode GrowOnly
Location 0; 0
X 0
Y 0
MaximumSize 0; 0
MinimumSize 0; 0
Padding 0; 0; 0; 0
Size 816; 489
Width 816
Height 489
StartPosition WindowsDefaultLocation


Font Microsoft Sans Serif; 8,25pt
ForeColor ControlText
FormBorderStyle Sizable
RightToLeft No
RightToLeftLayout False
Text Form1
UseWaitCursor False


Моя новая форма
Свойства
Form1 System.Windows.Forms.Form
BackColor
Control
BackgroundImage
(отсутствует)
Tile
Выбор ресурса
Контекст ресурса
Локальный ресурс:
Импорт...
Очистить
Файл ресурсов проекта:
Properties\Resources.resx
(нет)
Импорт...
Открыть
Шаблоны
Floristik
Упорядочить
Новая папка
П-32 33 34
Работа 3_Логистическая регрессия
Схематизация, проектирование
Этот компьютер
Видео
Документы
1
Моя новая форма


Файл
Правка
Вид
Проект
Сборка
Отладка
Формат
Тест
Анализ
Средства
Расширения
Окно
Справ
Код F7
Конструктор Shift+F7
Открыть
Открыть с помощью...
Обозреватель решений Ctrl+Alt+L
Team Explorer Ctrl+\, Ctrl+M
Обозреватель серверов Ctrl+Alt+S
Обозреватель тестов Ctrl+E, T
Окно закладок Ctrl+K, Ctrl+W
Иерархия вызовов Ctrl+Alt+K
Классы Ctrl+Shift+C
Окно определения кода Ctrl+\, D
Обозреватель объектов Ctrl+Alt+J
Список ошибок Ctrl+\, E
Вывод Ctrl+Alt+O
Список задач Ctrl+\, T
Панель элементов Ctrl+Alt+X
Уведомления Ctrl+\, Ctrl+N
Терминал Ctrl+`
Другие окна
Панели инструментов
Во весь экран Shift+Alt+ВВОД
Все окна Shift+Alt+M
Последовательность перехода
Form1.cs [Конструк
Form1
Обозреватель серверов
Панель элементов
Источники данных

Основные группы элементов управления

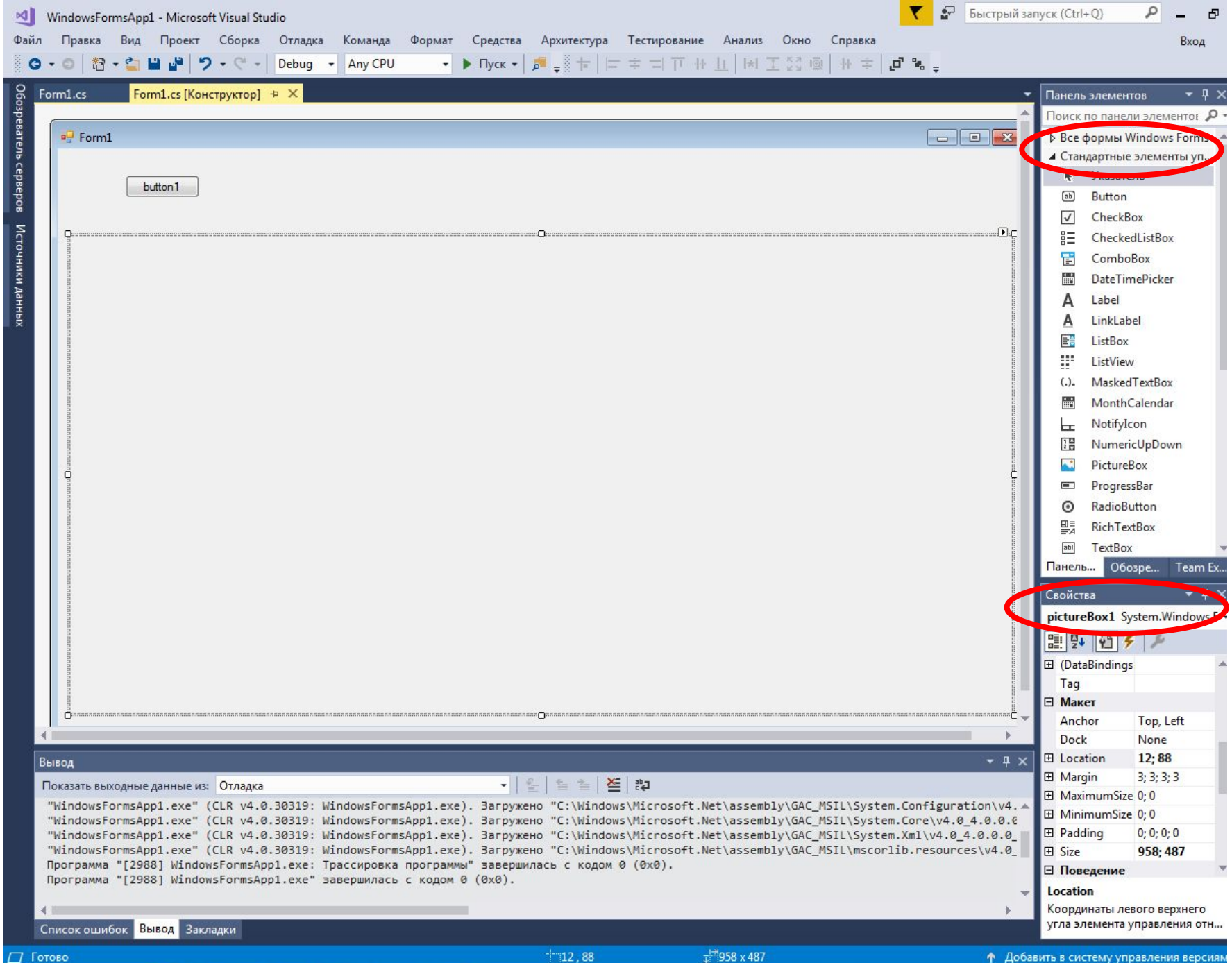
Категория	Интерфейсные элементы
Редактирование текста	**TextBox, RichTextBox**
Отображение текста	**Label, LinkLabel, StatusBar**
Выбор из списка	**CheckedListBox, ComboBox, DomainUpDown, ListBox, ListView, NumericUpDown, TreeView**
Отображение графики	**PictureBox**
Хранение графики	**ImageList**
Ввод значений	**CheckBox, CheckedListBox, RadioButton, TrackBar**
Ввод даты	**DateTimePicker, MonthCalendar**
Диалоговые панели	**ColorDialog, FontDialog, OpenFileDialog, PrintDialog, PrintPreviewDialog, SaveFileDialog**
Создание меню	**MenuStrip, ContextMenuStrip**
Команды	**Button, LinkLabel, NotifyIcon, ToolBar**
Объединение компонентов	**Panel, GroupBox, TabControl**

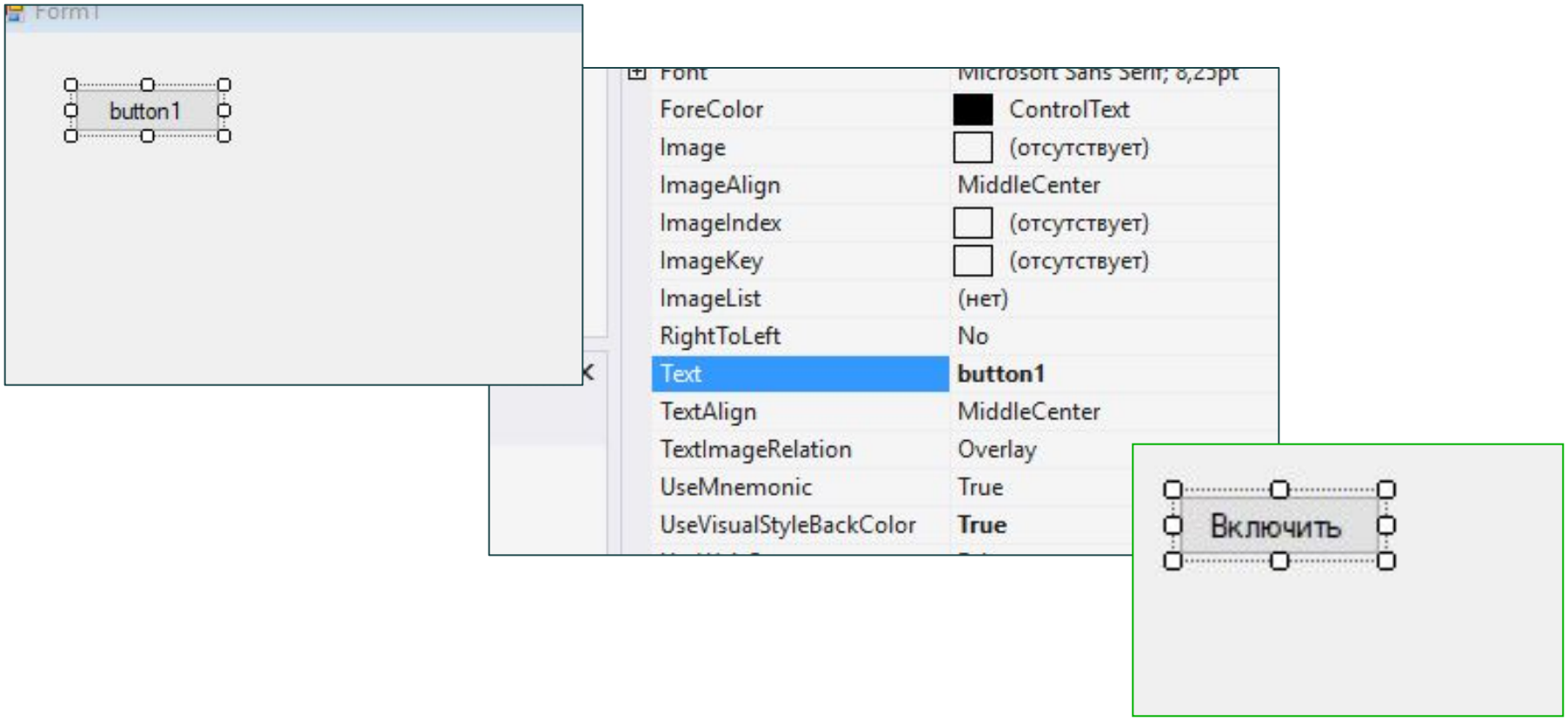
Элементы управления представляют собой визуальные классы, которые получают введенные пользователем данные и могут инициировать различные события. Все элементы управления наследуются от класса Control и поэтому имеют ряд общих свойств:

1) Anchor: Определяет, как элемент будет растягиваться
2) BackColor: Определяет фоновый цвет элемента
3) BackgroundImage: Определяет фоновое изображение элемента
4) ContextMenu: Контекстное меню, которое открывается при нажатии на элемент правой кнопкой мыши. Задается с помощью элемента ContextMenu
5) Cursor: Представляет, как будет отображаться курсор мыши при наведении на элемент
6) Dock: Задает расположение элемента на форме
7) Enabled: Определяет, будет ли доступен элемент для использования. Если это свойство имеет значение False, то элемент блокируется.
8) Font: Устанавливает шрифт текста для элемента
9) ForeColor: Определяет цвет шрифта
10) Location: Определяет координаты верхнего левого угла элемента управления
11) Name: Имя элемента управления
12) Size: Определяет размер элемента
13) Width: ширина элемента
14) Height: высота элемента
15) TabIndex: Определяет порядок обхода элемента по нажатию на клавишу Tab
16) Tag: Позволяет сохранять значение, ассоциированное с этим элементом управления

Задание 1

Поместить на форму кнопку «Включить« и окно pictureBox . Загрузить картинку в окно pictureBox.


WindowsFormsApp1 - Microsoft Visual Studio
Быстрый запуск (Ctrl+Q)
Файл Правка Вид Проект Сборка Отладка Команда Формат Средства Архитектура Тестирование Анализ Окно Справка
Вход
Debug
Any CPU
Пуск
Обозреватель серверов
Источники данных
Form1.cs
Form1.cs [Конструктор]
Form1
button1
Панель элементов
Поиск по панели элементов
Все формы Windows Forms
Стандартные элементы уп...
Button
CheckBox
CheckedListBox
ComboBox
DateTimePicker
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MonthCalendar
NotifyIcon
NumericUpDown
PictureBox
ProgressBar
RadioButton
RichTextBox
TextBox
Панель...
Обозре...
Team Ex...
Свойства
pictureBox1 System.Windows
(DataBindings
Tag
Макет
Anchor Top, Left
Dock None
Location 12; 88
Margin 3; 3; 3; 3
MaximumSize 0; 0
MinimumSize 0; 0
Padding 0; 0; 0; 0
Size 958; 487
Поведение
Location
Координаты левого верхнего угла элемента управления отн...
Вывод
Показать выходные данные из: Отладка
"WindowsFormsApp1.exe" (CLR v4.0.30319: WindowsFormsApp1.exe). Загружено "C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Configuration\v4.
"WindowsFormsApp1.exe" (CLR v4.0.30319: WindowsFormsApp1.exe). Загружено "C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Core\v4.0_4.0.0.0
"WindowsFormsApp1.exe" (CLR v4.0.30319: WindowsFormsApp1.exe). Загружено "C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0_
"WindowsFormsApp1.exe" (CLR v4.0.30319: WindowsFormsApp1.exe). Загружено "C:\Windows\Microsoft.Net\assembly\GAC_MSIL\mscorlib.resources\v4.0_
Программа "[2988] WindowsFormsApp1.exe: Трассировка программы" завершилась с кодом 0 (0x0).
Программа "[2988] WindowsFormsApp1.exe" завершилась с кодом 0 (0x0).
Список ошибок
Вывод
Закладки
Готово
12 , 88
958 x 487
Добавить в систему управления версиям


Form1
button1
Font
Microsoft Sans Serif; 8,25pt
ForeColor
ControlText
Image
(отсутствует)
ImageAlign
MiddleCenter
ImageIndex
(отсутствует)
ImageKey
(отсутствует)
ImageList
(нет)
RightToLeft
No
Text
button1
TextAlign
MiddleCenter
TextImageRelation
Overlay
UseMnemonic
True
UseVisualStyleBackColor
True
Включить

PictureBox предназначен для показа изображений. Он позволяет отобразить файлы в формате bmp, jpg, gif, а также метафайлы изображений и иконки.

Для установки изображения в PictureBox можно использовать ряд свойств:

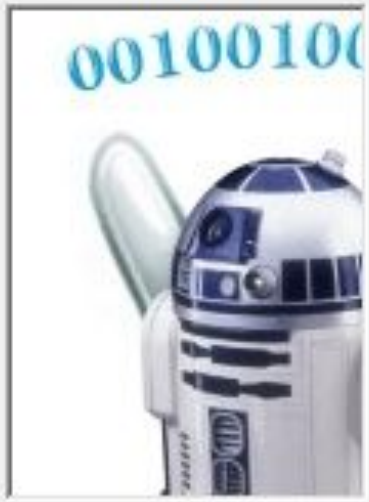
- **Image**: устанавливает объект типа Image
- **ImageLocation**: устанавливает путь к изображению на диске или в интернете
- **InitialImage**: некоторое начальное изображение, которое будет отображаться во время загрузки главного изображения, которое хранится в свойстве Image
- **ErrorImage**: изображение, которое отображается, если основное изображение не удалось загрузить в PictureBox

Размер изображения

Для установки изображения в PictureBox используется свойство **SizeMode**, которое принимает следующие значения:

- Normal: изображение позиционируется в левом верхнем углу PictureBox, и размер изображения не изменяется. Если PictureBox больше размеров изображения, то по справа и снизу появляются пустоты, если меньше - то изображение обрезается
- StretchImage: изображение растягивается или сжимается таким образом, чтобы вместиться по всей ширине и высоте элемента PictureBox
- AutoSize: элемент PictureBox автоматически растягивается, подстраиваясь под размеры изображения
- CenterImage: если PictureBox меньше изображения, то изображение обрезается по краям и выводится только его центральная часть. Если же PictureBox больше изображения, то оно позиционируется по центру.
- Zoom: изображение подстраивается под размеры PictureBox, сохраняя при этом пропорции

Normal



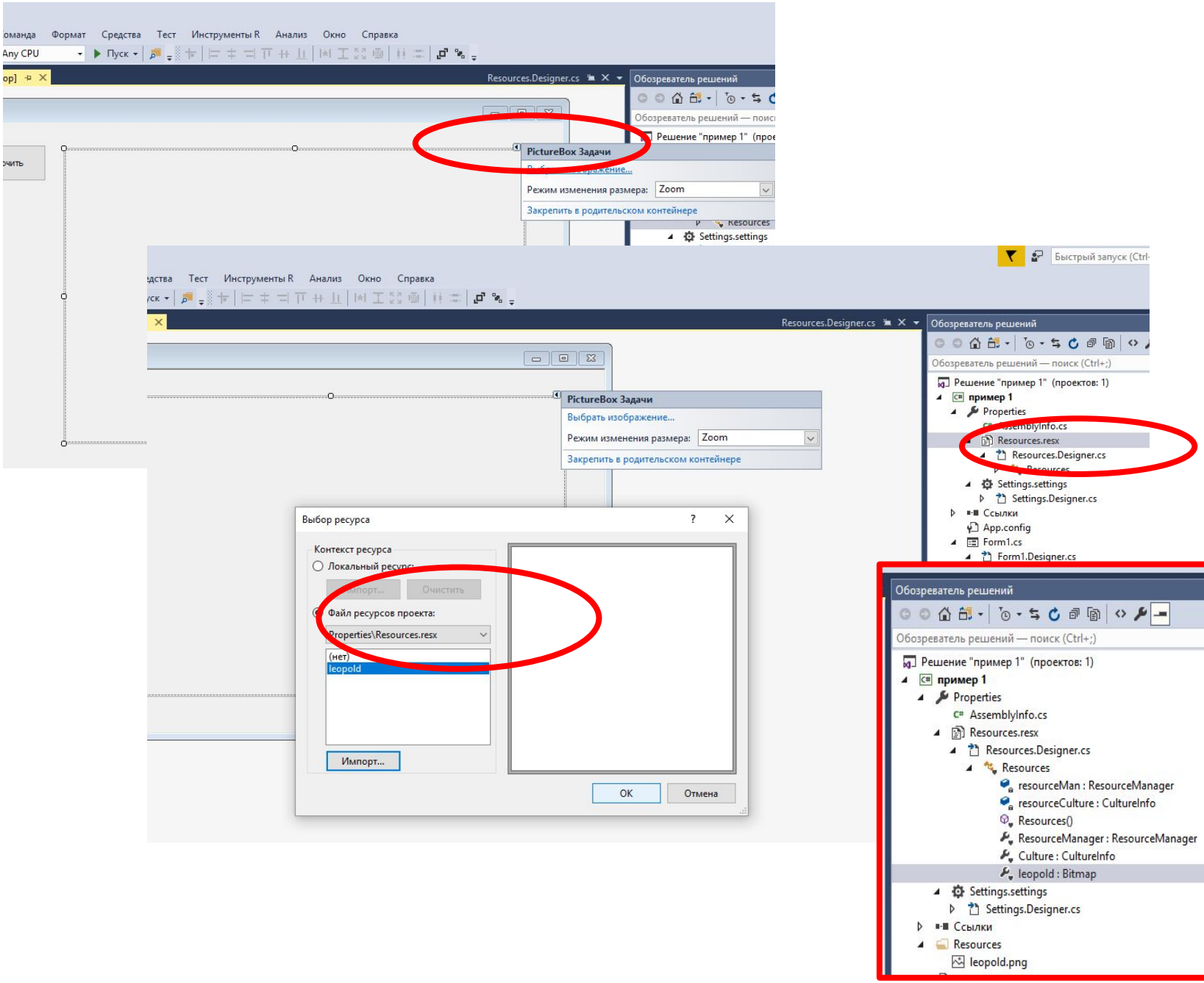
AutoSize

StretchImage

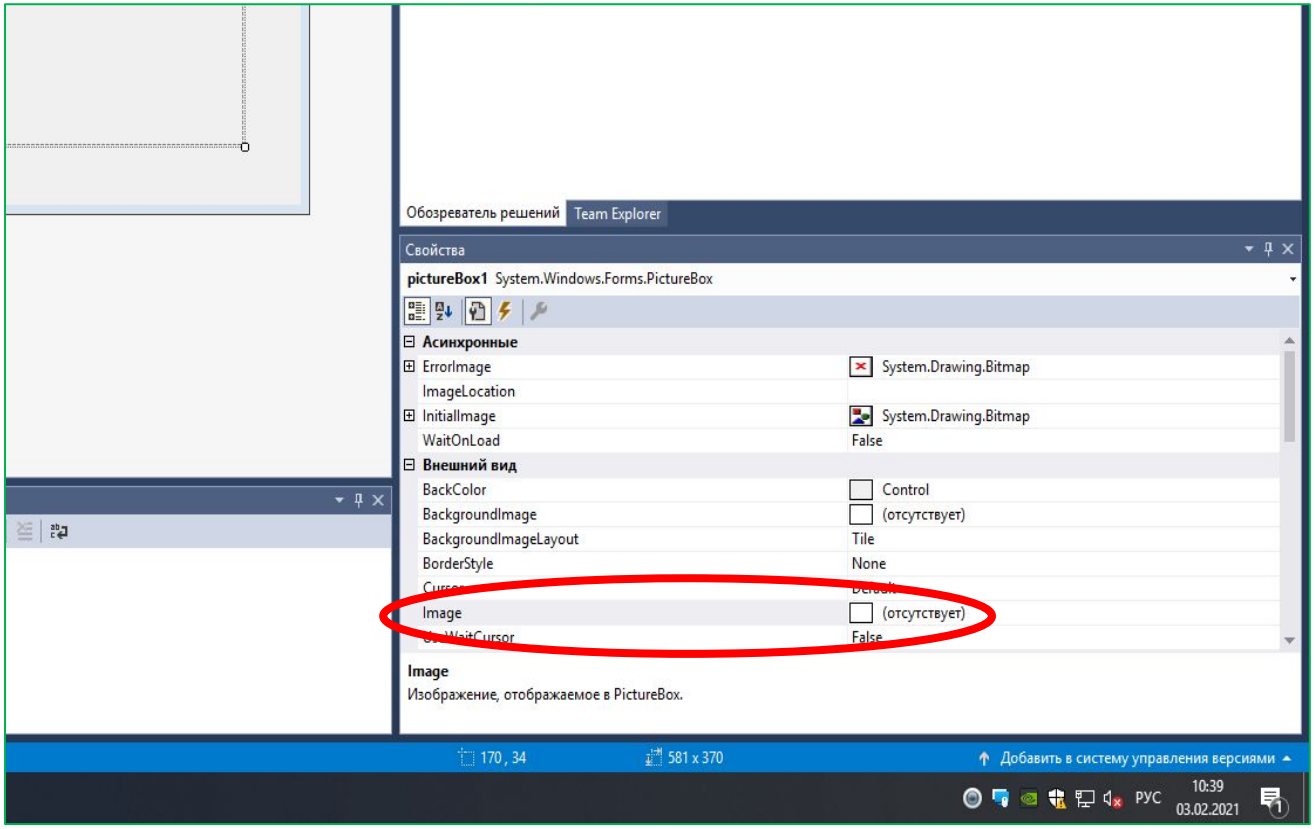
CenterImage

Zoom

1 способ

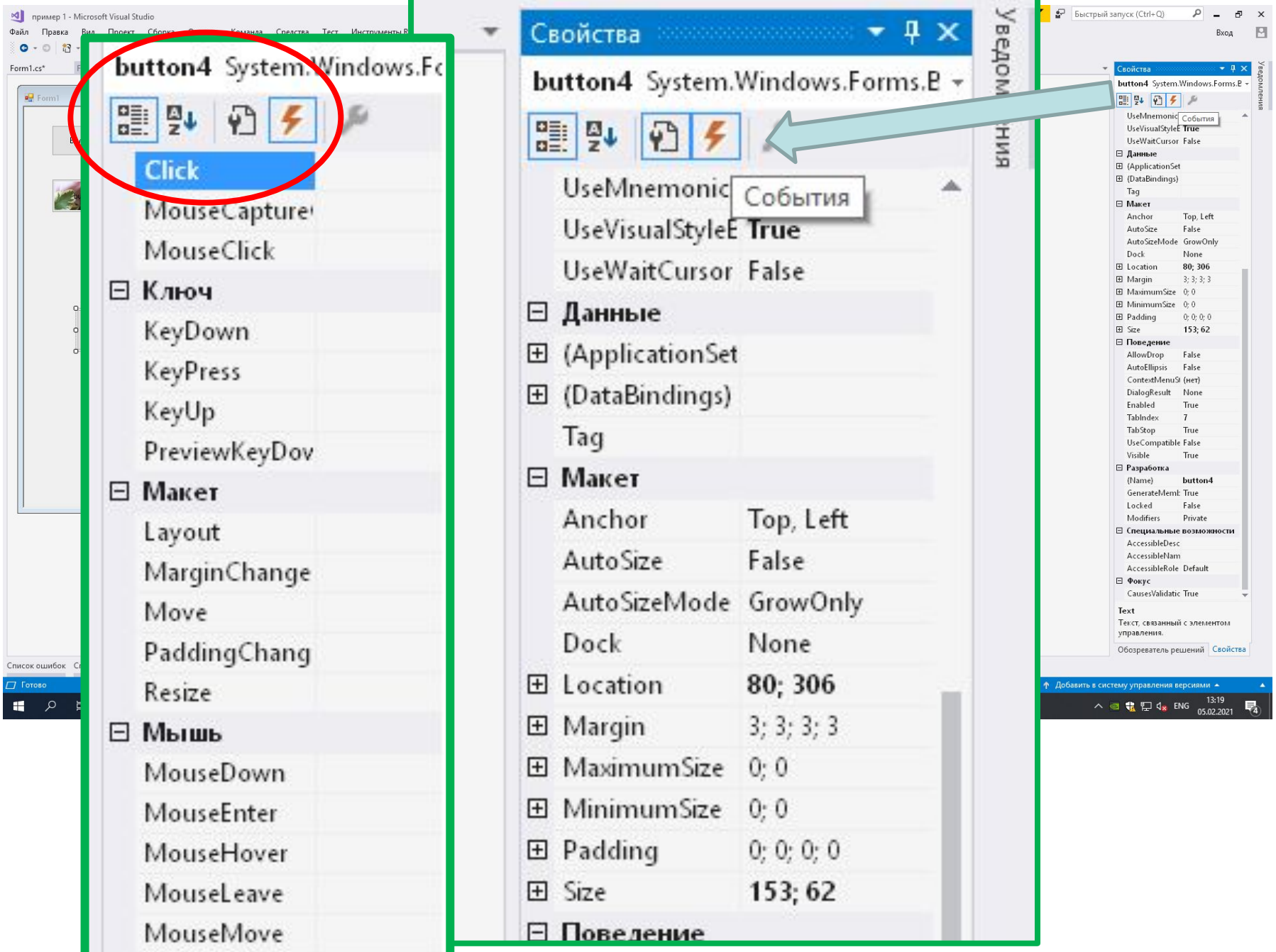

Any CPU
Пуск
Resources.Designer.cs
Обозреватель решений
Обозреватель решений — поиск (Ctrl+;)
Решение "пример 1" (проектов: 1)
пример 1
Properties
AssemblyInfo.cs
Resources.resx
Resources.Designer.cs
Resources
resourceMan : ResourceManager
resourceCulture : CultureInfo
Resources()
ResourceManager : ResourceManager
Culture : CultureInfo
leopold : Bitmap
Settings.settings
Settings.Designer.cs
Ссылки
App.config
Form1.cs
Form1.Designer.cs
Resources
leopold.png
PictureBox Задачи
Выбрать изображение...
Режим изменения размера:
Zoom
Закрепить в родительском контейнере
Выбор ресурса
Контекст ресурса
Локальный ресурс:
Импорт...
Очистить
Файл ресурсов проекта:
Properties\Resources.resx
(нет)
leopold
Импорт...
OK
Отмена

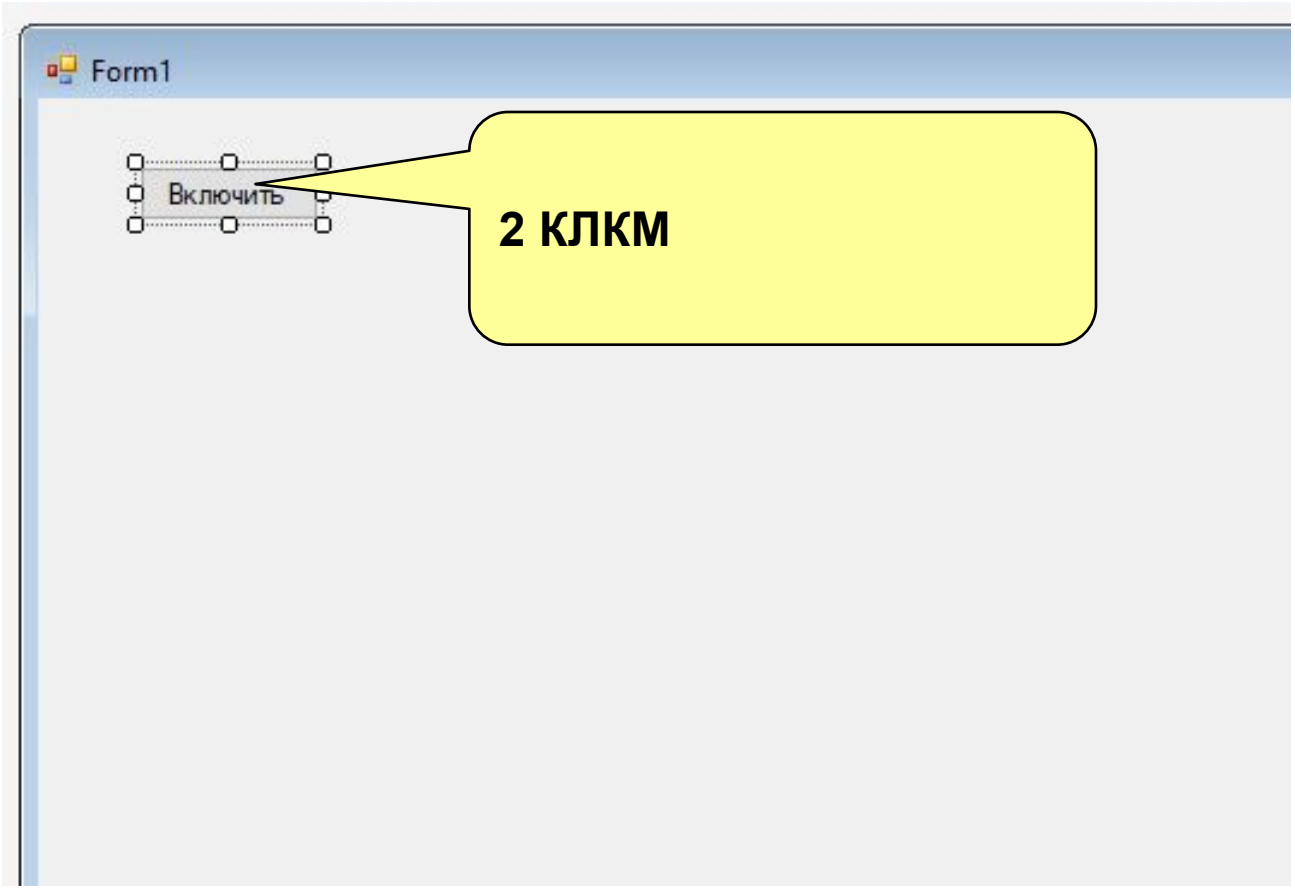
2 способ


Обозреватель решений
Team Explorer
Свойства
pictureBox1 System.Windows.Forms.PictureBox
Асинхронные
ErrorImage
System.Drawing.Bitmap
ImageLocation
InitialImage
System.Drawing.Bitmap
WaitOnLoad
False
Внешний вид
BackColor
Control
BackgroundImage
(отсутствует)
BackgroundImageLayout
Tile
BorderStyle
None
Image
(отсутствует)
False
Image
Изображение, отображаемое в PictureBox.
170 , 34
581 x 370
Добавить в систему управления версиями
РУС
10:39
03.02.2021

Задание 2

Поместить на форму кнопку «Включить« и окно pictureBox . При нажатии на кнопку появляется картинка из коллекции.


Свойства
button4 System.Windows.Forms.B
UseMnemonic
UseVisualStyleB True
UseWaitCursor False
Данные
(ApplicationSet
(DataBindings)
Tag
Макет
Anchor Top, Left
AutoSize False
AutoSizeMode GrowOnly
Dock None
Location 80; 306
Margin 3; 3; 3; 3
MaximumSize 0; 0
MinimumSize 0; 0
Padding 0; 0; 0; 0
Size 153; 62
Поведение
button4 System.Windows.Fc
Click
MouseCapture
MouseClick
Ключ
KeyDown
KeyPress
KeyUp
PreviewKeyDov
Макет
Layout
MarginChange
Move
PaddingChang
Resize
Мышь
MouseDown
MouseEnter
MouseHover
MouseLeave
MouseMove
События


Form1
Включить
2 КЛКМ

Form1.cs* | Form1.cs [Конструктор]*

пример 1 | пример_1.Form1

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

namespace пример_1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {

        }
    }
}
```

Form1.cs* | Form1.cs [Конструктор]*

пример 1 | пример_1.Form1

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

namespace пример_1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            pictureBox1.Image = Image.FromFile("Z:\\Прилепина Е.В\\Основы программирования\\Картинки\\foto.jpg");
        }
    }
}
```

Вместо одного \ пишем два \\ !!!!!!

Кнопка

Наиболее часто используемым элементом управления является кнопка. Обрабатывая событие нажатия кнопки, может производить те или иные действия.

Оформление кнопки

Чтобы управлять внешним отображением кнопки, можно использовать свойство FlatStyle. Оно может принимать следующие значения:

1. Flat - Кнопка имеет плоский вид
2. Popup - Кнопка приобретает объемный вид при наведении на нее указателя, в иных случаях она имеет плоский вид
3. Standard - Кнопка имеет объемный вид (используется по умолчанию)
4. System - Вид кнопки зависит от операционной системы

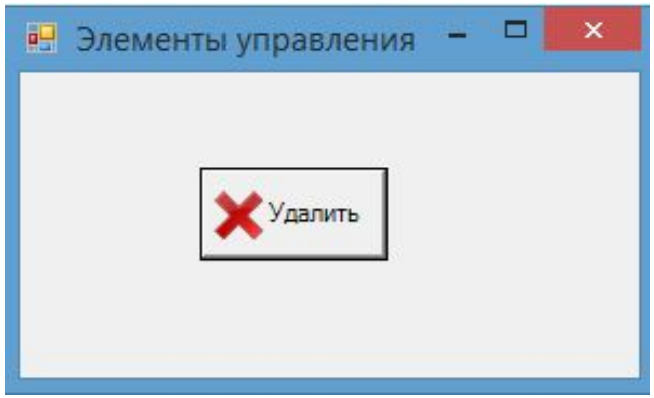
Изображение на кнопке

Как и для многих элементов управления, для кнопки можно задавать изображение с помощью свойства BackgroundImage. Однако мы можем также управлять размещением текста и изображения на кнопки. Для этого надо использовать свойство TextImageRelation. Оно приобретает следующие значения:

1. Overlay: текст накладывается на изображение
2. ImageAboveText: изображение располагается над текстом
3. TextAboveImage: текст располагается над изображением
4. ImageBeforeText: изображение располагается перед текстом
5. TextBeforeImage: текст располагается перед изображением

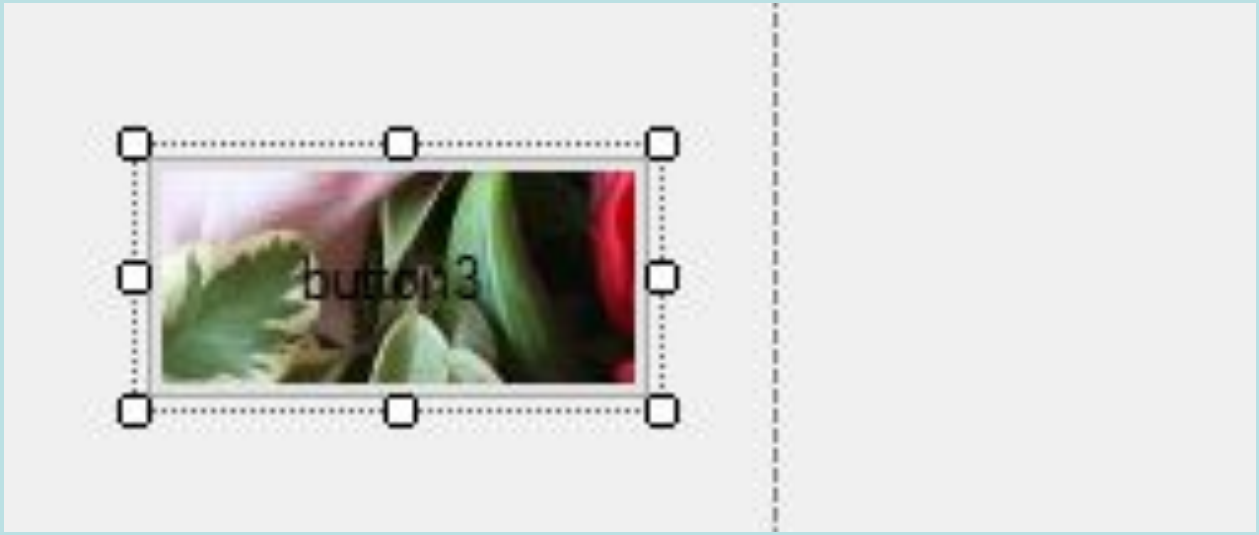
Компоненту Button (командная кнопка) можно ставить в соответствие функцию, которая будет выполнена при нажатии на кнопку.

Свойство TextImageRelation
установить для него значение ImageBeforeText.
В итоге мы получим кнопку, где сразу после изображения идет надпись на кнопке:

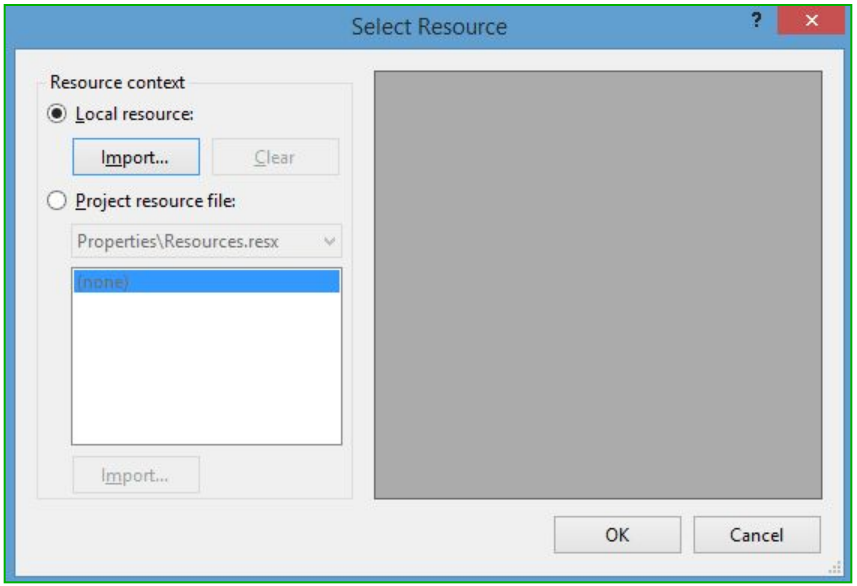


Задание 3

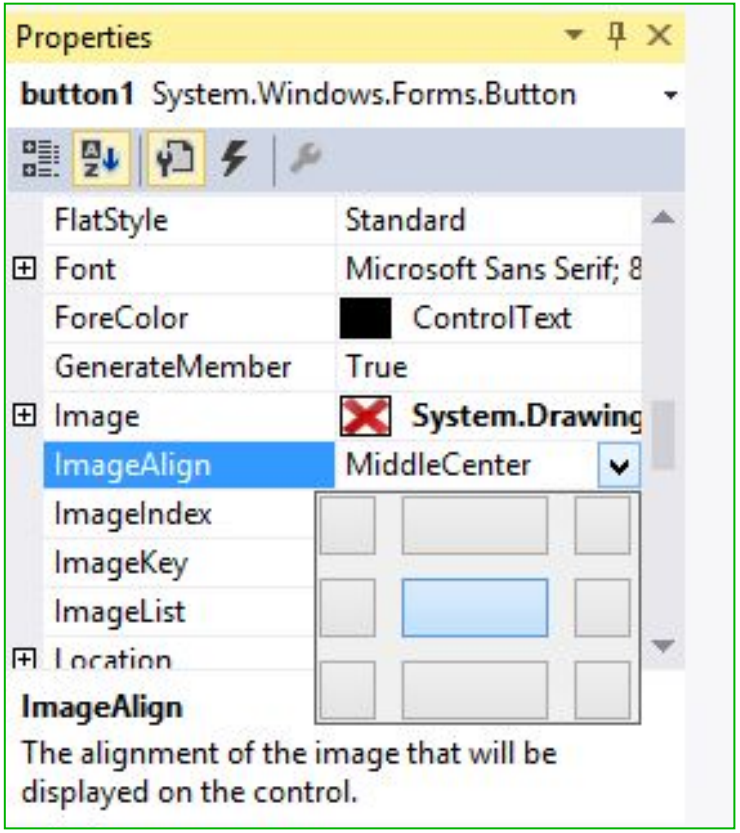
Поместить на форму кнопку . Настроить интерфейс кнопки, добавив картинку и надпись.



Установить для кнопки изображение. Для этого выберем кнопку и в окне Свойств нажмем на поле Image (не путать с BackgroundImage). Откроется диалоговое окно установки изображения:



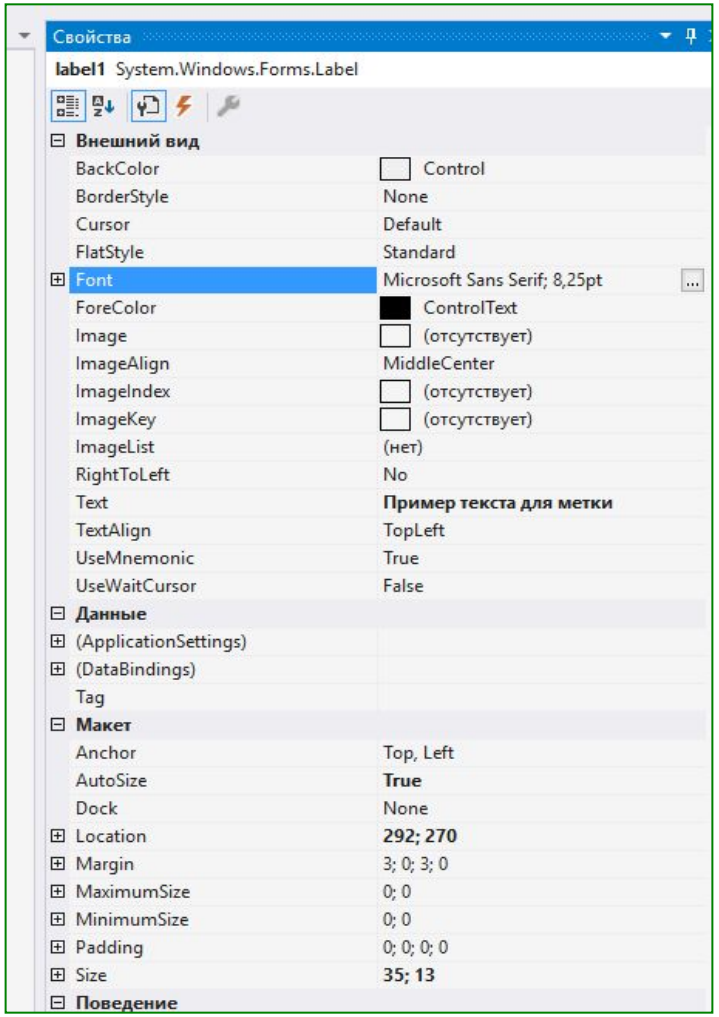
После выбора изображения установить свойство ImageAlign, которое управляет позиционированием изображения на кнопке:



Метки и ссылки

Label

Для отображения простого текста на форме, доступного только для чтения, служит элемент Label. Чтобы задать отображаемый текст метки, надо установить свойство Text элемента.



LinkLabel

Особый тип меток представляют элементы LinkLabel, которые предназначены для вывода ссылок, которые аналогичны ссылкам, размещенным на стандартных веб-станиц.

Во время работы приложения невозможно редактировать содержимое меток, поэтому их рекомендуют использовать для вывода пояснительного текста и результатов.

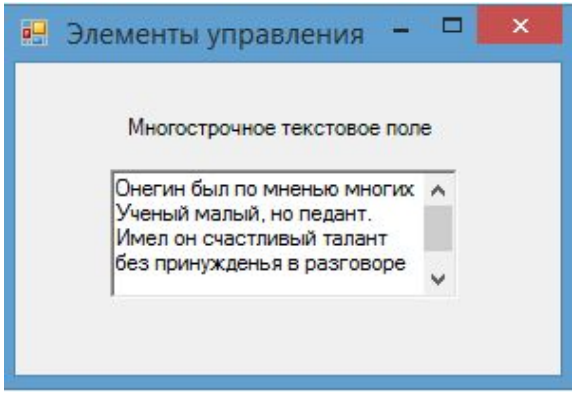
Текстовое поле TextBox

Для ввода и редактирования текста предназначены текстовые поля - элемент TextBox. Текст элемента TextBox можно установить или получить с помощью свойства Text.

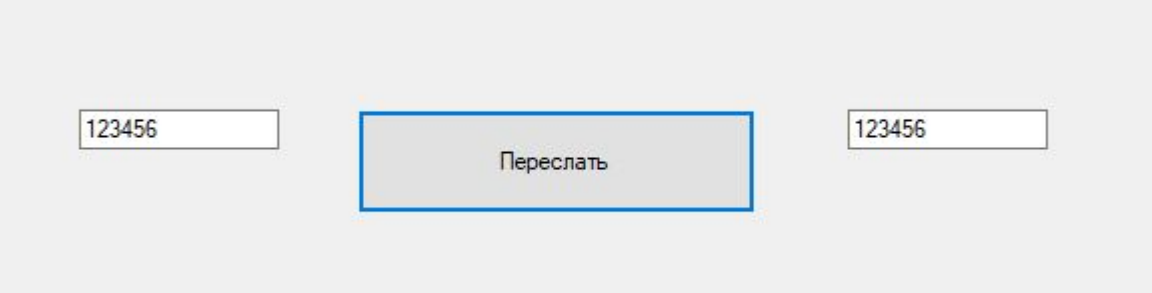
По умолчанию создается однострочное текстовое поле. Для отображения больших объемов информации в текстовом поле нужно использовать его свойства Multiline и ScrollBars. При установке для свойства Multiline значения true, все избыточные символы, которые выходят за границы поля, будут переноситься на новую строку.
Кроме того, можно сделать прокрутку текстового поля, установив для его свойства ScrollBars одно из значений:

- **None**: без прокруток (по умолчанию)
- **Horizontal**: создает горизонтальную прокрутку при длине строки, превышающей ширину текстового поля
- **Vertical:** создает вертикальную прокрутку, если строки не помещаются в текстовом поле
- **Both**: создает вертикальную и горизонтальную прокрутку

TextBox (строка редактирования) предназначена для ввода/вывода, тип данных в нем – всегда String и все преобразования должны выполняться программистом.



Задание 4.
Поместить на форму два поля ввода и кнопку "Переслать".
При нажатии на кнопку текст из первого поля ввода копируется во второе.

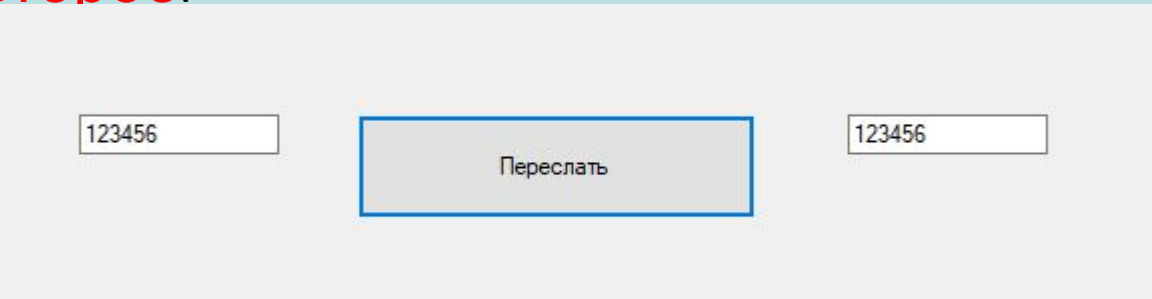



```
ссылка: 1
private void button2_Click(object sender, EventArgs e)
{
    textBox1.Text = textBox2.Text;
}
```

Задание 4.

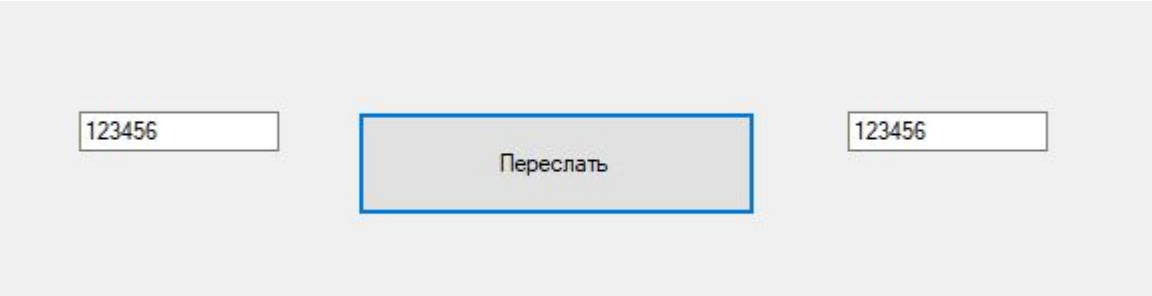
Поместить на форму два поля ввода и кнопку "Переслать".

При нажатии на кнопку текст из первого поля ввода копируется во второе.




```
ссылка: 1
private void button2_Click(object sender, EventArgs e)
                2                   1
    textBox1.Text = textBox2.Text;
}
```

Задание 4.
Поместить на форму два поля ввода и кнопку "Переслать".
При нажатии на кнопку текст из первого поля ввода копируется во второе.

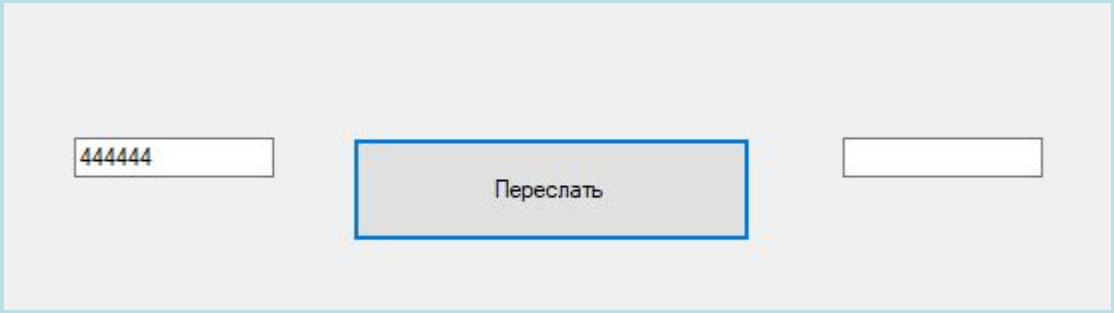



```
ссылка: 1
private void button2_Click(object sender, EventArgs e)
{
    textBox1.Text = textBox2.Text;
}
```

А если написать:
textBox2.Text = textBox2.Text + textBox1.Text
Что будет?

Задание 5.

При нажатии на кнопку текст из первого поля ввода переписывается во второе, то есть исчезает в первом поле и появляется во втором.




```
ссылка: 1
private void button2_Click(object sender, EventArgs e)
{
    textBox1.Text = textBox2.Text;
    textBox2.Text = "";
}
```

или

```
textBox1.Clear();
```

Задание 6.

Создадим шуточную программу, представляющую собой диалоговое окно с двумя кнопками.

Назовем его SocOpros. Из окна Toolbox перетаскиваем на форму две кнопки Button и надпись Label и устанавливаем следующие свойства элементов управления и формы:

Объект, свойства	**Значение**
Text	Социологический опрос
Label1, свойство	**Значение**
Bold	True
Text	Вы довольны своей зарплатой?
Button1, свойство	**Значение**
Name	Btnyes
Text	Да
Button2, свойство	**Значение**
Name	Btnno
Text	Нет

Щелкаем дважды по кнопке "Да".

В обработчике этой кнопки вставляем следующий код:

```
void btnyes_Click(object sender, EventArgs e)
{
MessageBox.Show("Мы и не сомневались, что Вы так думаете!");
 }
```

Выделяем кнопку "Нет".
Открываем окно Properties (События).
Переключаемся в окно событий и дважды щелкаем в поле MouseMove.
В обработчике связываем движение мыши с координатами кнопки и устанавливаем координаты, куда она будет возвращаться, если во время своего движения выйдет за указанную область:

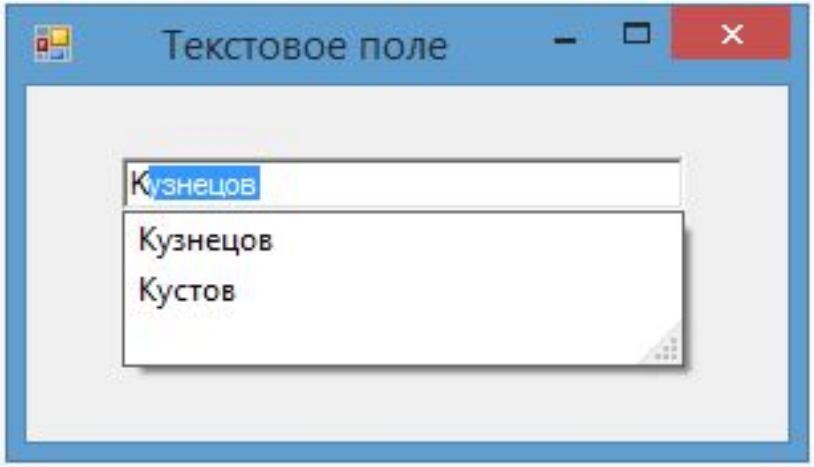
```
private void Btnno_MouseMove(object sender, MouseEventArgs e)
 {
Btnno.Top -= e.Y;
 Btnno.Left += e.X;
if (Btnno.Top < -10 || Btnno.Top > 100)
Btnno.Top = 60;
 if (Btnno.Left < -80 || Btnno.Left > 250)
Btnno.Left = 120;
}
```

Автозаполнение текстового поля

Элемент TextBox обладает достаточными возможностями для создания автозаполняемого поля. Для этого нам надо привязать свойство AutoCompleteCustomSource элемента TextBox к некоторой коллекции, из которой берутся данные для заполнения поля.

```
AutoCompleteStringCollection source = new AutoCompleteStringCollection()
{
    "Кузнецов",
    "Иванов",
    "Петров",
    "Кустов"
};
textBox1.AutoCompleteCustomSource = source;
textBox1.AutoCompleteMode = AutoCompleteMode.SuggestAppend;
textBox1.AutoCompleteSource = AutoCompleteSource.CustomSource;
```

Элементы в список могут добавляться как во время разработки, так и программным способом!



Перенос по словам

Чтобы текст в элементе TextBox переносился по словам, надо установить свойство WordWrap равным true. То есть если одно слово не умещается на строке, то но переносится на следующую. Данное свойство будет работать только для многострочных текстовых полей.

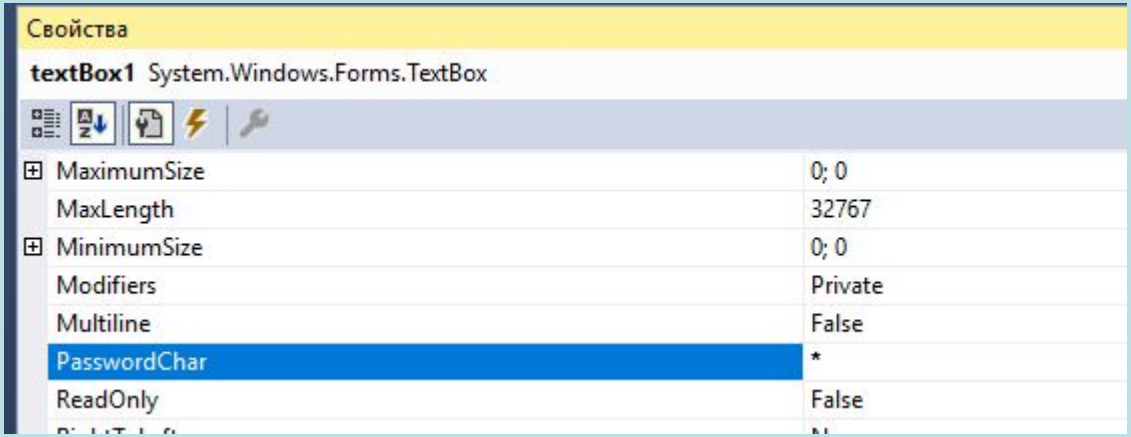
Ввод пароля

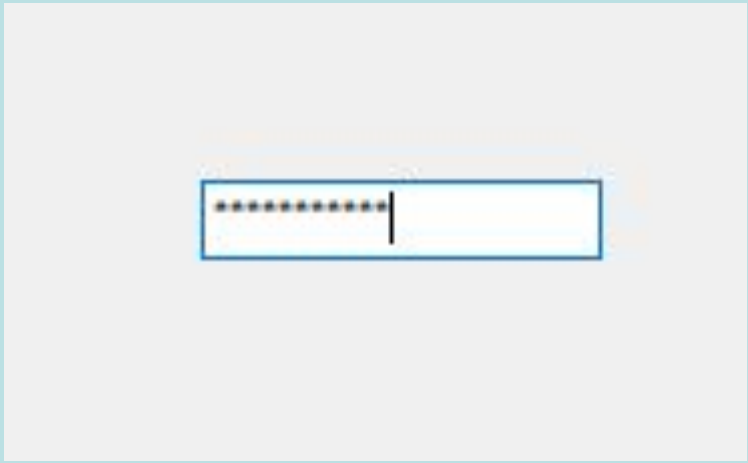
Также данный элемент имеет свойства, которые позволяют сделать из него поле для ввода пароля. Так, для этого надо использовать PasswordChar и UseSystemPasswordChar.

Свойство PasswordChar по умолчанию не имеет значение, если мы установим в качестве него какой-нибудь символ, то этот символ будут отображаться при вводе любых символов в текстовое поле.
Свойство UseSystemPasswordChar имеет похожее действие. Если мы установим его значение в true, то вместо введенных символов в текстовом поле будет отображаться знак пароля, принятый в системе, например, точка.

Задание 7.
Свойство PasswordChar

У элемента TextBox установить в Свойство PasswordChar какой-нибудь символ, то этот символ будут отображаться при вводе любых символов в текстовое поле.

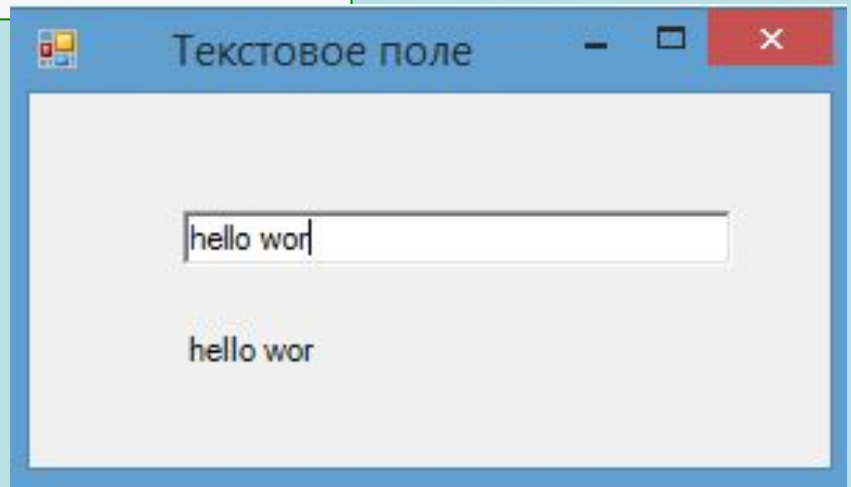




Задание 8.

Поместить на форму кроме текстового поля метку и сделать так, чтобы при изменении текста в текстовом поле также менялся текст на метке:

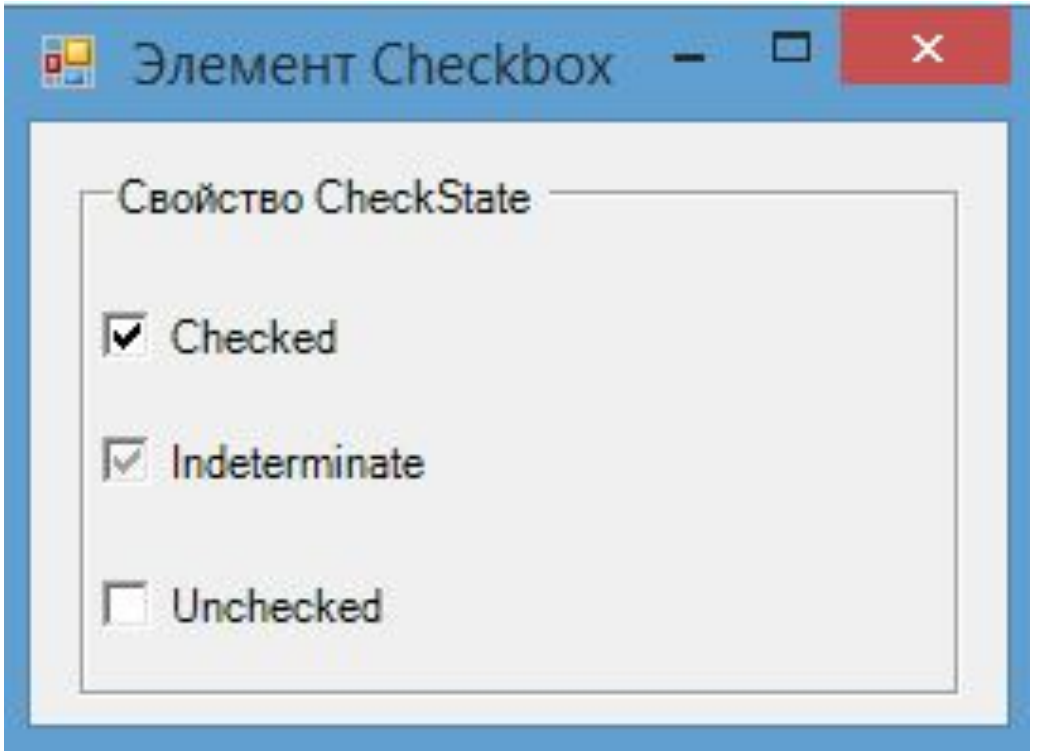
```
public Form1()
{
    InitializeComponent();

    textBox1.TextChanged += textBox1_TextChanged;
}

private void textBox1_TextChanged(object sender, EventArgs e)
{
    label1.Text = textBox1.Text;
}
```



CheckBox

Элемент CheckBox или флажок предназначен для установки одного из двух значений: отмечен или не отмечен. Чтобы отметить флажок, надо установить у его свойства **Checked** значение true.

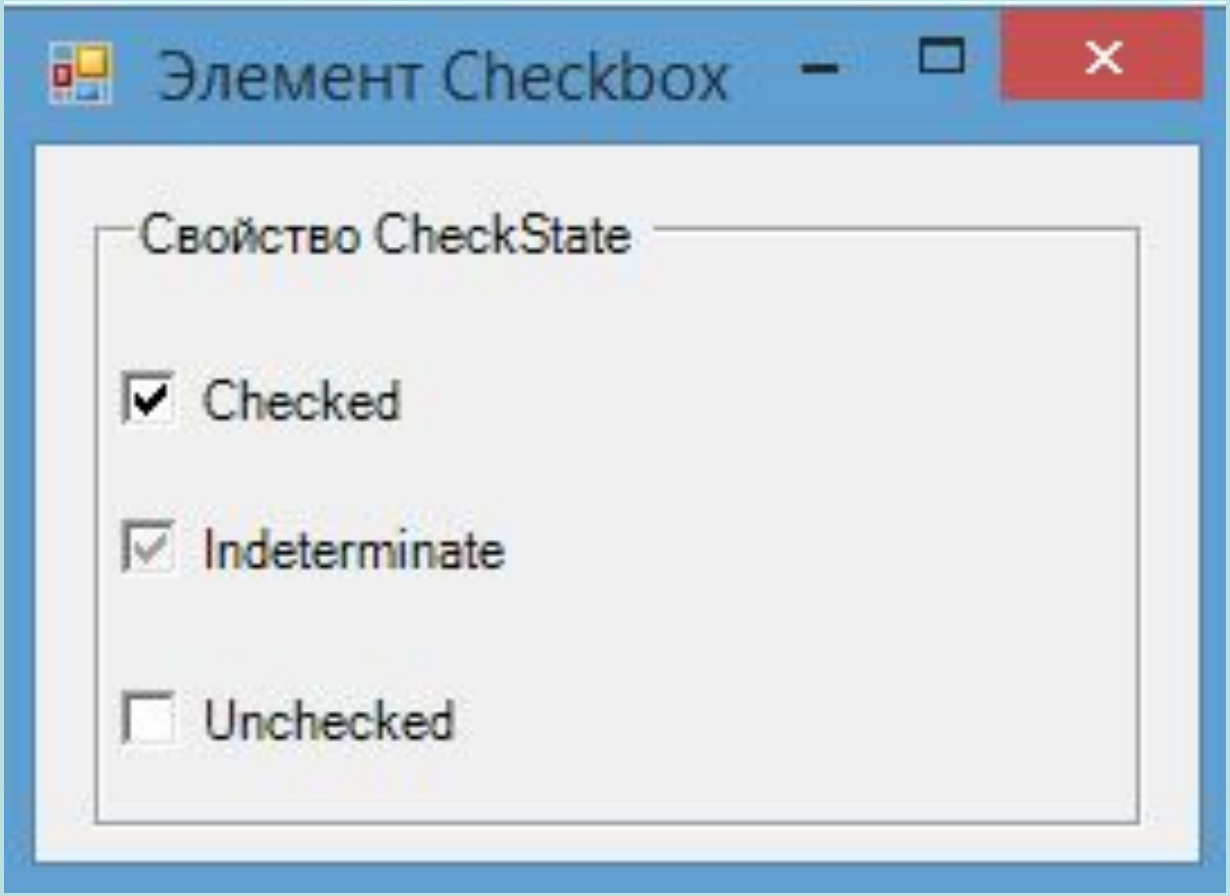
Кроме свойства Checked у элемента CheckBox имеется свойство **CheckState**, которое позволяет задать для флажка одно из трех состояний - Checked (отмечен), Indeterminate (флажок не определен - отмечен, но находится в неактивном состоянии) и Unchecked (не отмечен)



Задание 9.

Поместить на форму 3 флажка.

Проверить работу со свойствами - **Checked , CheckState**

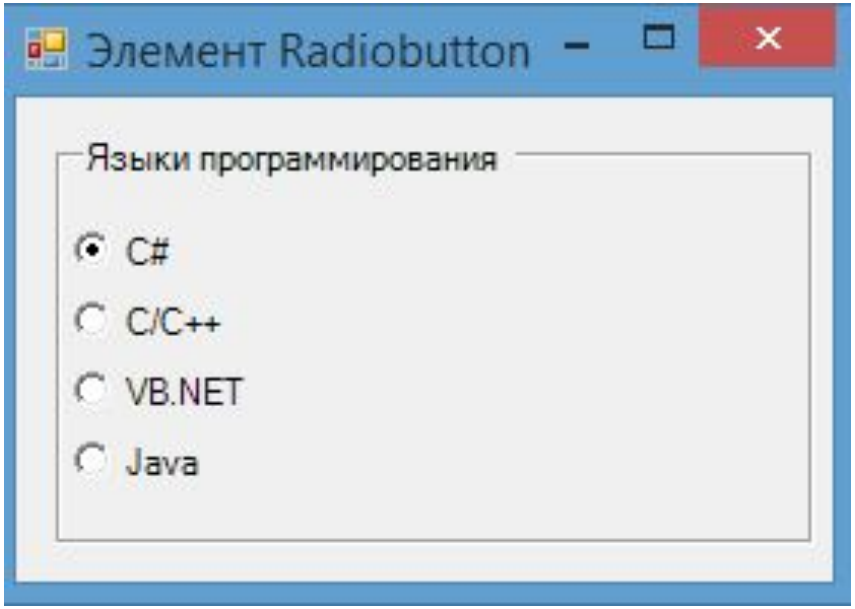


Radiobutton

На элемент CheckBox похож элемент RadioButton или переключатель. Переключатели располагаются группами, и включение одного переключателя означает отключение всех остальных.

Чтобы установить у переключателя включенное состояние, надо присвоить его свойству Checked значение true.

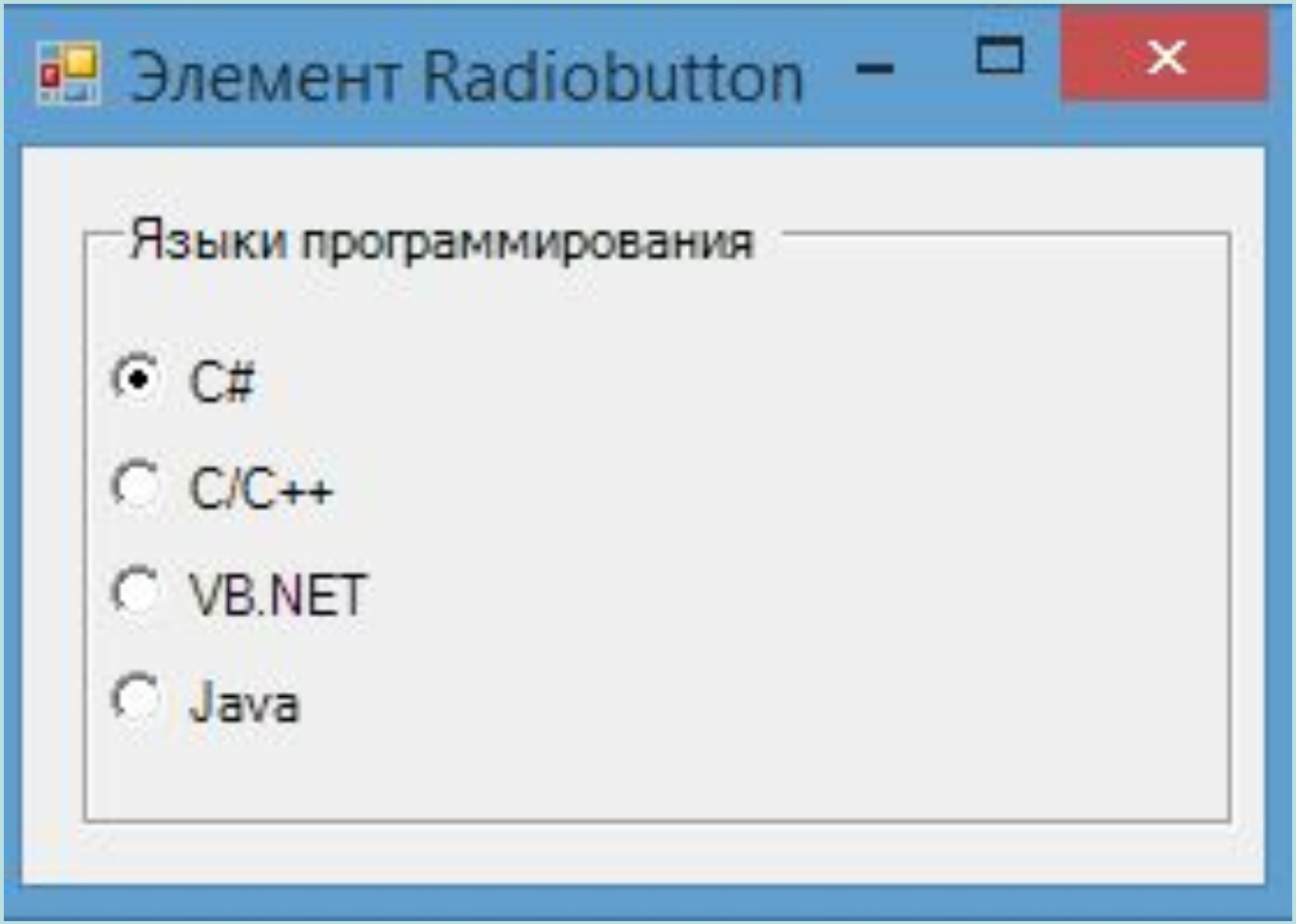
Для создания группы переключателей, из которых можно бы было выбирать, надо поместить несколько переключателей в какой-нибудь контейнер, например, в элементы GroupBox или Panel. Переключатели. находящиеся в разных контейнерах, будут относиться к разным группам:



Задание 10.

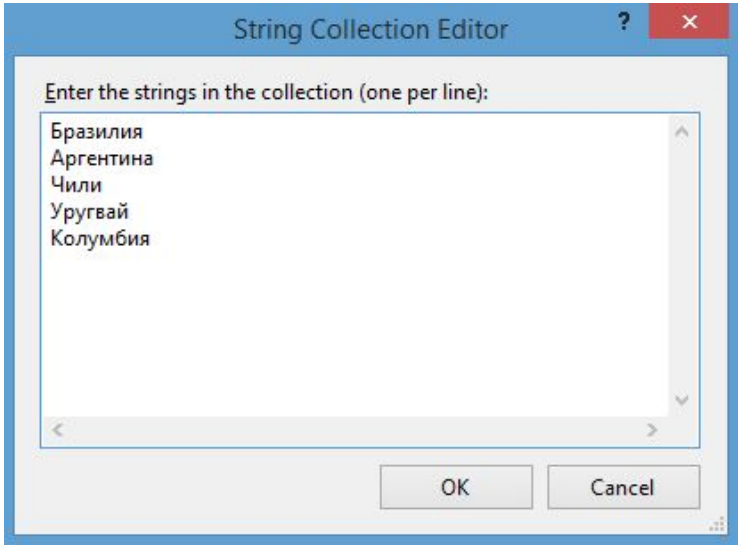
Поместить на форму элементы **GroupBox и Panel.** Поместить по 3 радиокнопки в каждый.

Проверить работу со свойством - **Checked.**

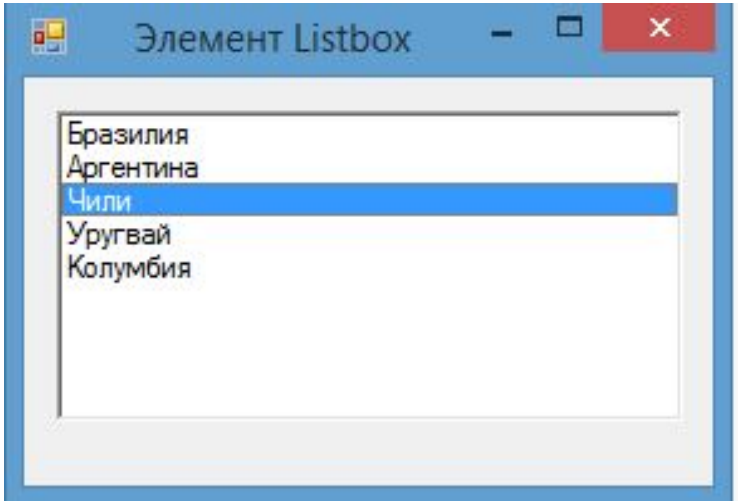


ListBox Элемент ListBox представляет собой простой список. Ключевым свойством этого элемента является свойство Items, которое как раз и хранит набор всех элементов списка.

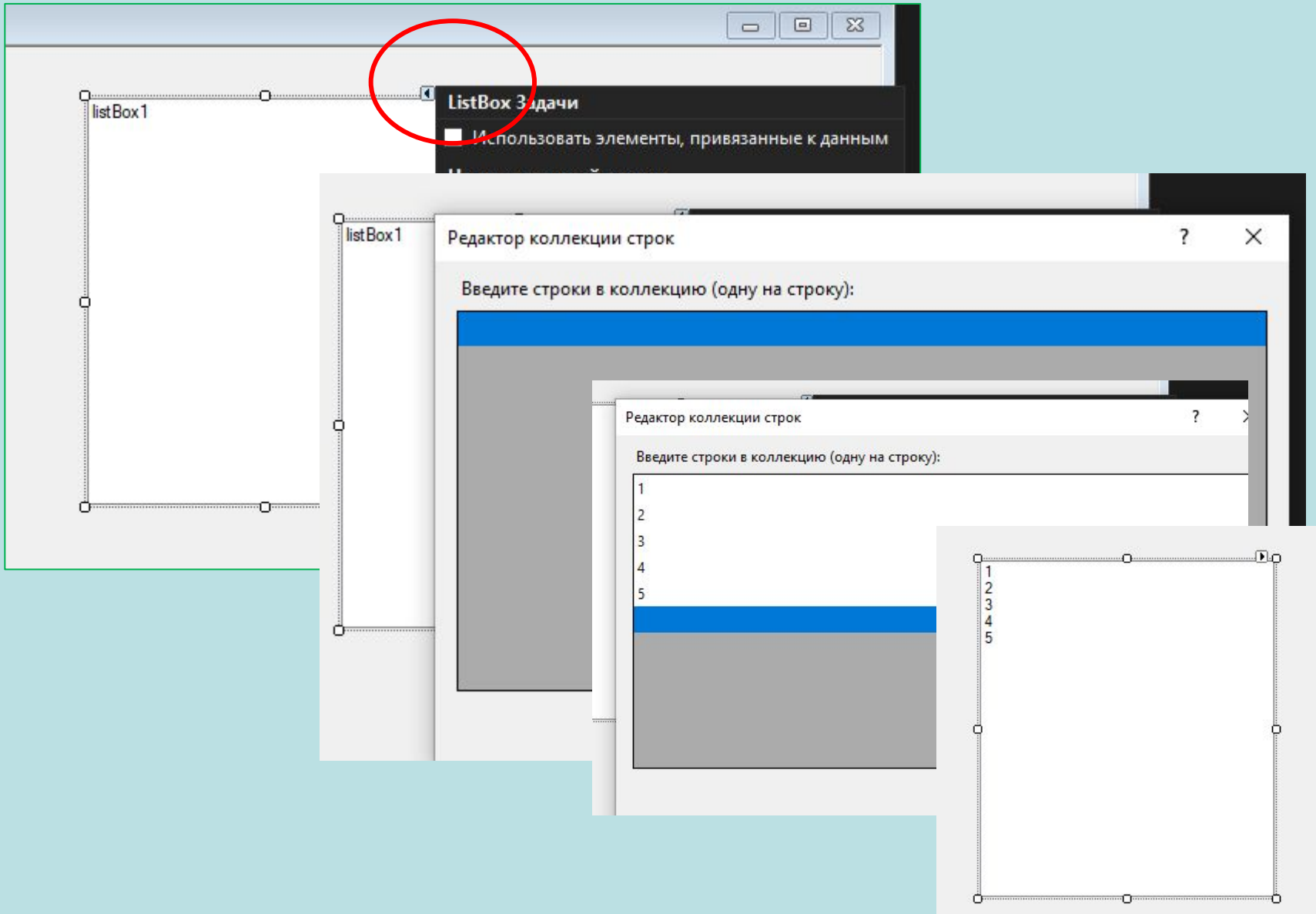
Элементы в список могут добавляться как во время разработки, так и программным способом. В Visual Studio в окне Properties (Свойства) для элемента ListBox мы можем найти свойство Items. После двойного щелчка на свойство нам отобразится окно для добавления элементов в список:



В пустое поле мы вводим по одному элементу списка - по одному на каждой строке. После этого все добавленные нами элементы окажутся в списке, и мы сможем ими управлять:

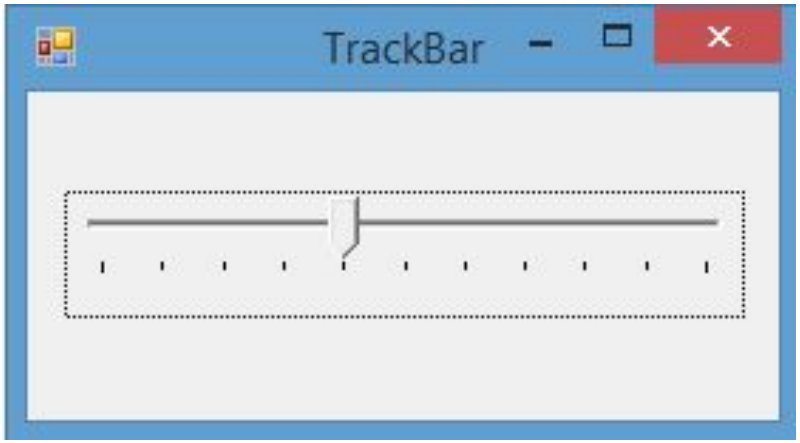


Задание 11.
Поместить на форму элемент **ListBox.** Добавьте элементы в список.



TrackBar

TrackBar представляет собой элемент, который с помощью перемещения ползунка позволяет вводить числовые значения.

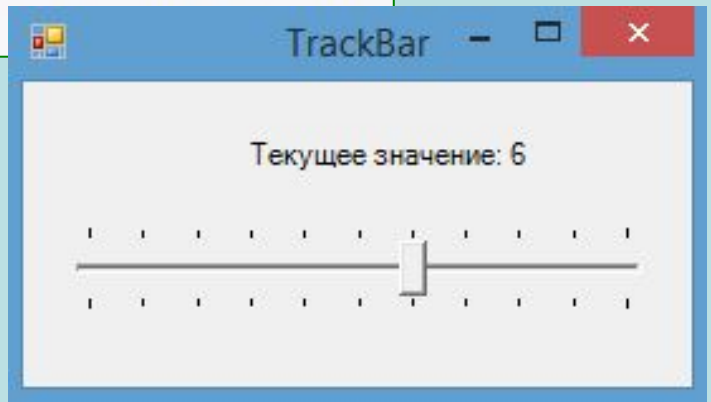


Некоторые важные свойства TrackBar:

- Orientation: задает ориентацию ползунка - расположение по горизонтали или по вертикали
- TickStyle: задает расположение делений на ползунке
- TickFrequency: задает частоту делений на ползунке
- Minimum: минимальное возможное значение на ползунке (по умолчанию 0)
- Maximum: максимальное возможное значение на ползунке (по умолчанию 10)
- Value: текущее значение ползунка. Должно находиться между Minimum и Maximum

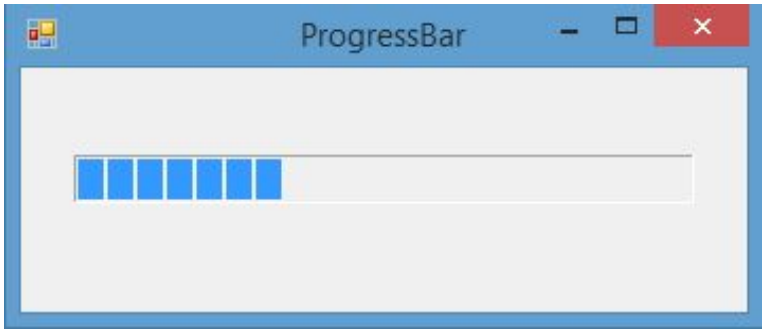
Задание 12.
Поместите на форму TrackBar. Настройте его событие.
К наиболее важным событиям элемента следует отнести событие Scroll, которое позволяет обработать перемещение ползунка от одного деления к другому. Что может быть полезно, если нам надо, например, устанавливать соответствующую громкость звука в зависимости от значения ползунка, либо какие-нибудь другие настройки.

```
public Form1()
{
    InitializeComponent();
    // установка обработчика события Scroll
    trackBar1.Scroll+=trackBar1_Scroll;
}

private void trackBar1_Scroll(object sender, EventArgs e)
{
    label1.Text = String.Format("Текущее значение: {0}", trackBar1.Value);
}
```



ProgressBar

Элемент ProgressBar служит для того, чтобы дать пользователю информацию о ходе выполнения какой-либо задачи.

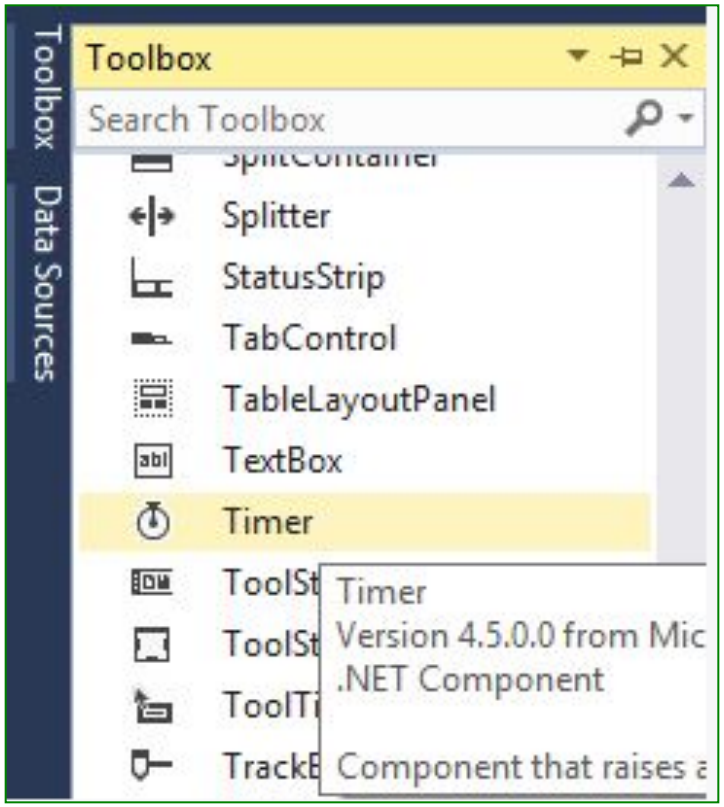


Наиболее важные свойства ProgressBar:

- Minimum : минимальное возможное значение
- Maximum : максимальное возможное значение
- Value : текущее значение элемента
- Step : шаг, на который изменится значение Value при вызове метода PerformStep

Timer

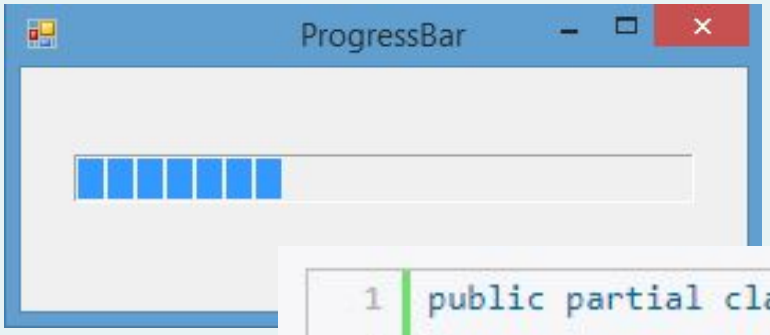
Timer является компонентом для запуска действий, повторяющихся через определенный промежуток времени. Хотя он не является визуальным элементом, но его а также можно перетащить с Панели Инструментов на форму:



Наиболее важные свойства и методы таймера:

- Свойство Enabled: при значении true указывает, что таймер будет запускаться вместе с запуском формы
- Свойство Interval: указывает интервал в миллисекундах, через который будет срабатывать обработчик события Tick, которое есть у таймера
- Метод Start(): запускает таймер
- Метод Stop(): останавливает таймер

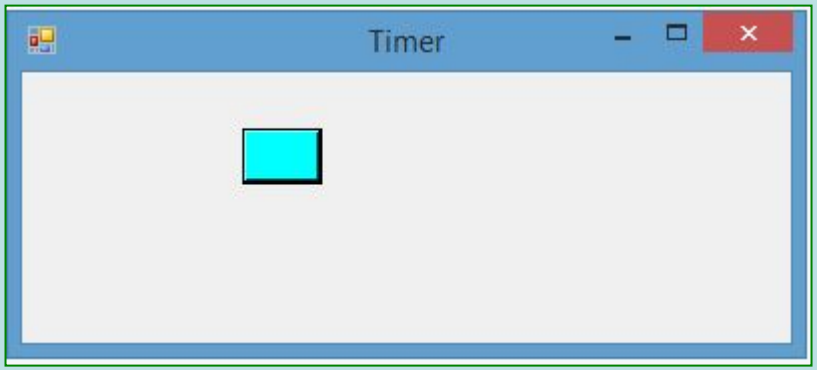
Задание 13.
Для имитации работы прогрессбара поместите на форму таймер и в коде формы определите следующий код:




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

        timer1.Interval = 500; // 500 миллисекунд
        timer1.Enabled = true;
        timer1.Tick += timer1_Tick;
    }
    // обработчик события Tick таймера
    void timer1_Tick(object sender, EventArgs e)
    {
        progressBar1.PerformStep();
    }
}
```

Задание 14 .

Создайте форму, на которую добавьте кнопку и таймер.



Условие:

1. В конструкторе формы устанавливаются начальные значения для таймера, кнопки и формы.
2. Через каждый интервал таймера будет срабатывать обработчик timer1_Tick, в котором изменяется положение кнопки по горизонтали с помощью свойства button1.Left.
3. А с помощью дополнительной переменной koef можно управлять направлением движения.
4. Кроме того, с помощью обработчика нажатия кнопки button1_Click можно либо остановить таймер (и вместе с ним движение кнопки), либо опять его запустить.

Пример кода:

```
public partial class Form1 : Form
{
    int koef = 1;
    public Form1()
    {
        InitializeComponent();

        this.Width = 400;
        button1.Width = 40;
        button1.Left = 40;
        button1.Text = "";
        button1.BackColor = Color.Aqua;

        timer1.Interval = 500; // 500 миллисекунд
        timer1.Enabled = true;
        button1.Click += button1_Click;
        timer1.Tick += timer1_Tick;
    }
```

```
// обработчик события Tick таймера
void timer1_Tick(object sender, EventArgs e)
{
    if (button1.Left == (this.Width-button1.Width-10))
    {
        koef=-1;
    }
    else if (button1.Left == 0)
    {
        koef = 1;
    }
    button1.Left += 10 *koef;
}
```

```
// обработчик нажатия на кнопку
void button1_Click(object sender, EventArgs e)
{
    if(timer1.Enabled==true)
    {
        timer1.Stop();
    }
    else
    {
        timer1.Start();
    }
}
```