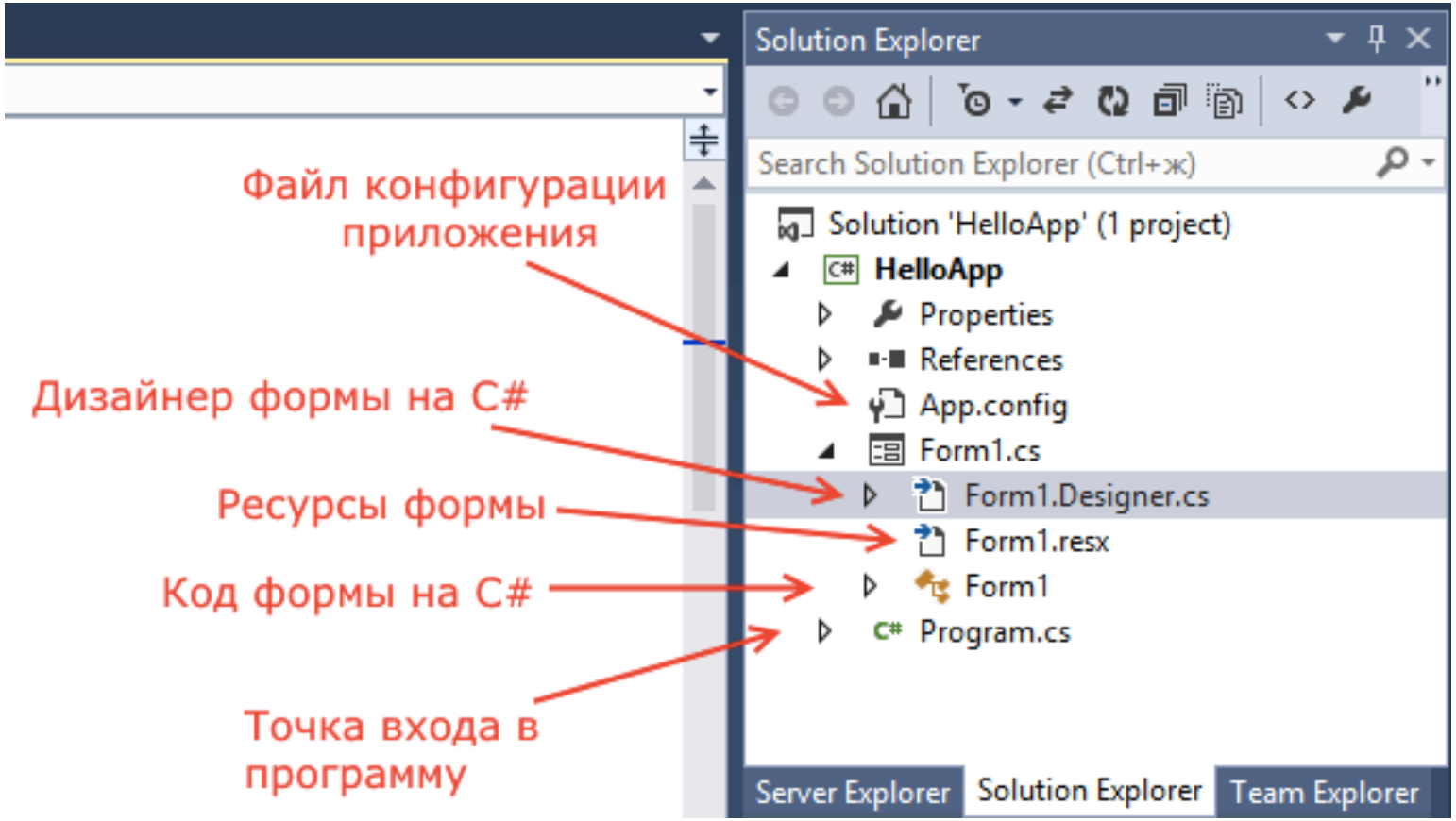
Работа с формами

Внешний вид приложения - формы. Формы являются основными строительными блоками. Они предоставляют контейнер для различных элементов управления.



Program.cs:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5  using System.Windows.Forms;
6
7  namespace HelloApp
8  {
9      static class Program
10     {
11         [STAThread]
12         static void Main()
13         {
14             Application.EnableVisualStyles();
15             Application.SetCompatibleTextRenderingDefault(false);
16             Application.Run(new Form1());
17         }
18     }
19 }
```

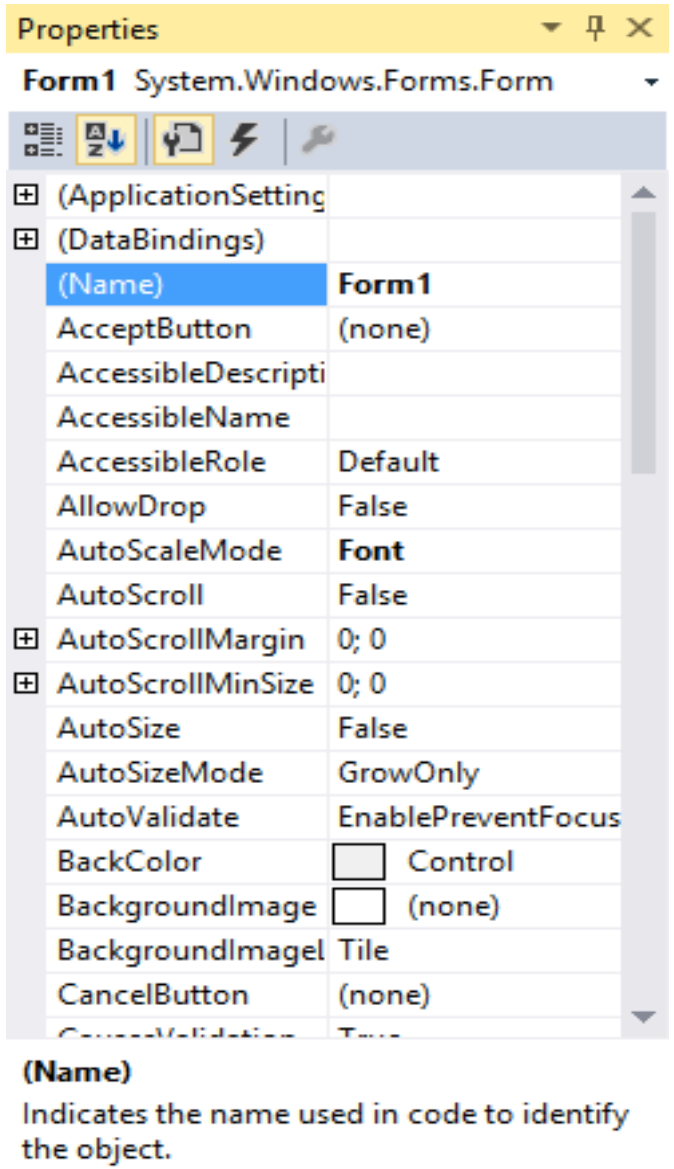
Form1.Designer.cs

```
namespace HelloApp
{
    partial class Form1
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.SuspendLayout();
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(284, 261);
            this.Name = "Form1";
            this.Text = "Привет мир!";
            this.ResumeLayout(false);

        }

        #endregion

    }
}
```

Form1.cs

```
using System;
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 HelloApp
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

Основные свойства форм



•**Name**: устанавливает имя формы - точнее имя класса, который наследуется от класса Form

•**BackColor**: указывает на фоновый цвет формы. Щелкнув на это свойство, мы сможем выбрать тот цвет, который нам подходит из списка предложенных цветов или цветовой палитры

•**BackgroundImage**: указывает на фоновое изображение формы

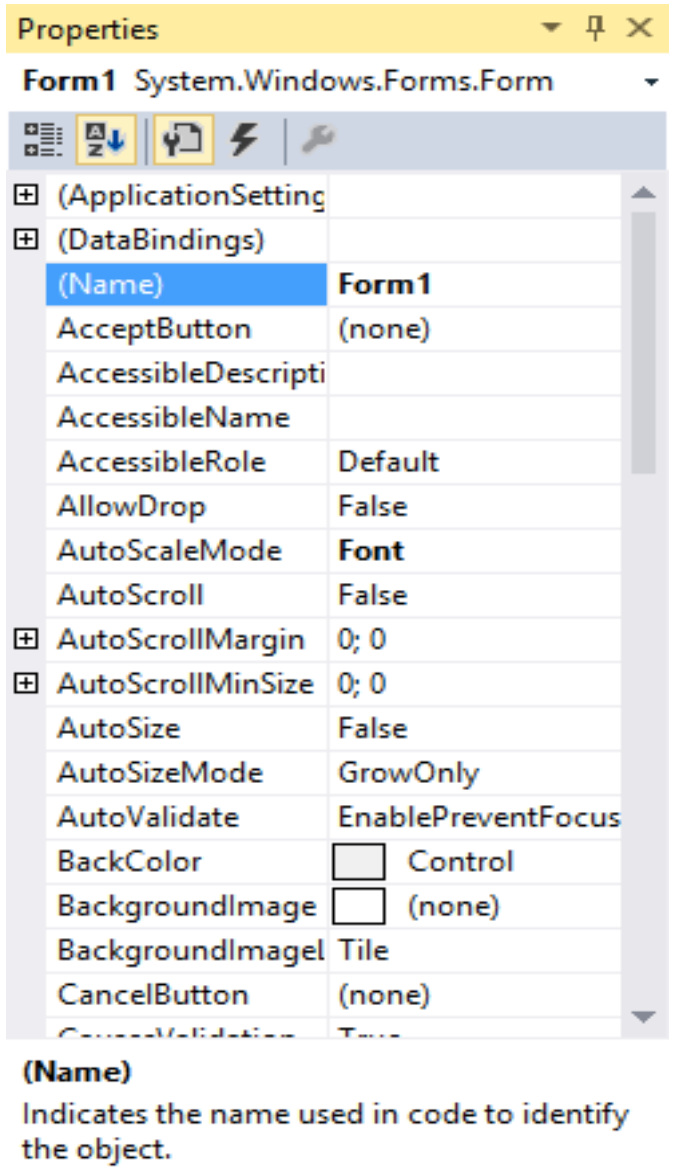
•**BackgroundImageLayout**: определяет, как изображение, заданное в свойстве BackgroundImage, будет располагаться на форме.

•**ControlBox**: указывает, отображается ли меню формы. В данном случае под меню понимается меню самого верхнего уровня, где находятся иконка приложения, заголовок формы, а также кнопки минимизации формы и крестик. Если данное свойство имеет значение false, то мы не увидим ни иконку, ни крестика, с помощью которого обычно закрывается форма

•**Cursor**: определяет тип курсора, который используется на форме

•**Enabled**: если данное свойство имеет значение false, то она не сможет получать ввод от пользователя, то есть мы не сможем нажать на кнопки, ввести текст в текстовые поля и т.д.

Основные свойства форм



Font: задает шрифт для всей формы и всех помещенных на нее элементов управления. Однако, задав у элементов формы свой шрифт, мы можем тем самым переопределить его

ForeColor: цвет шрифта на форме

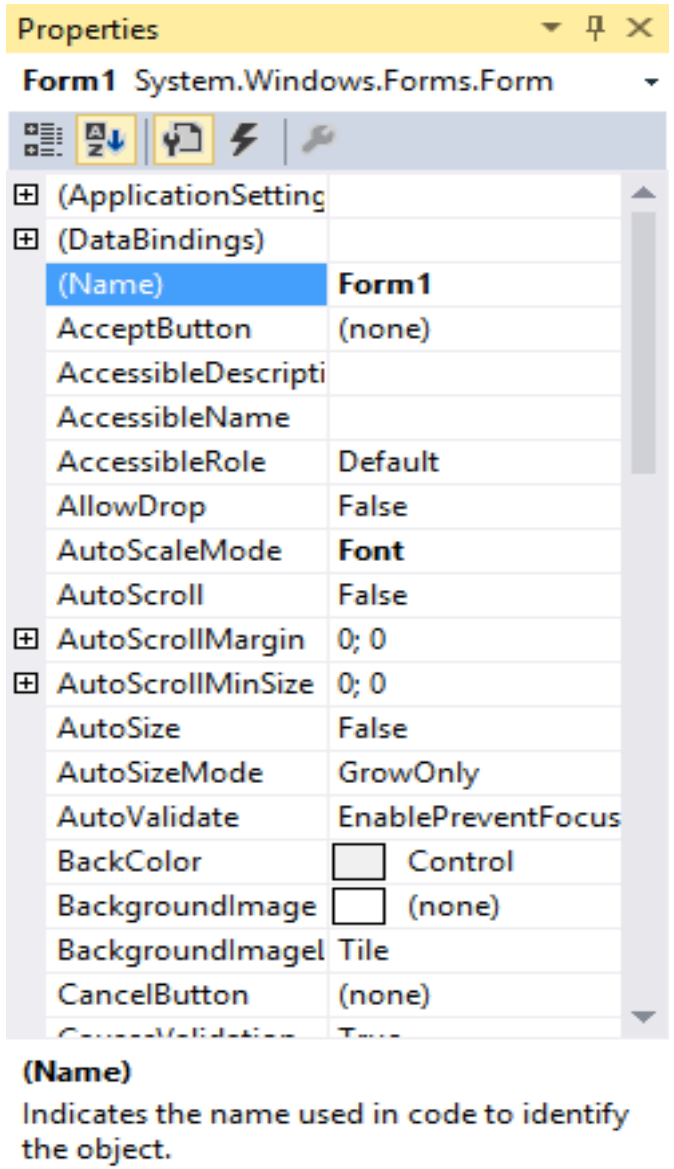
FormBorderStyle: указывает, как будет отображаться граница формы и строка заголовка. Устанавливая данное свойство в None можно создавать внешний вид приложения произвольной формы

HelpButton: указывает, отображается ли кнопка справки формы

Icon: задает иконку формы

Location: определяет положение по отношению к верхнему левому углу экрана, если для свойства StartPosition установлено значение Manual

Основные свойства форм



MaximizeBox: указывает, будет ли доступна кнопка максимизации окна в заголовке формы

MinimizeBox: указывает, будет ли доступна кнопка минимизации окна

MaximumSize: задает максимальный размер формы

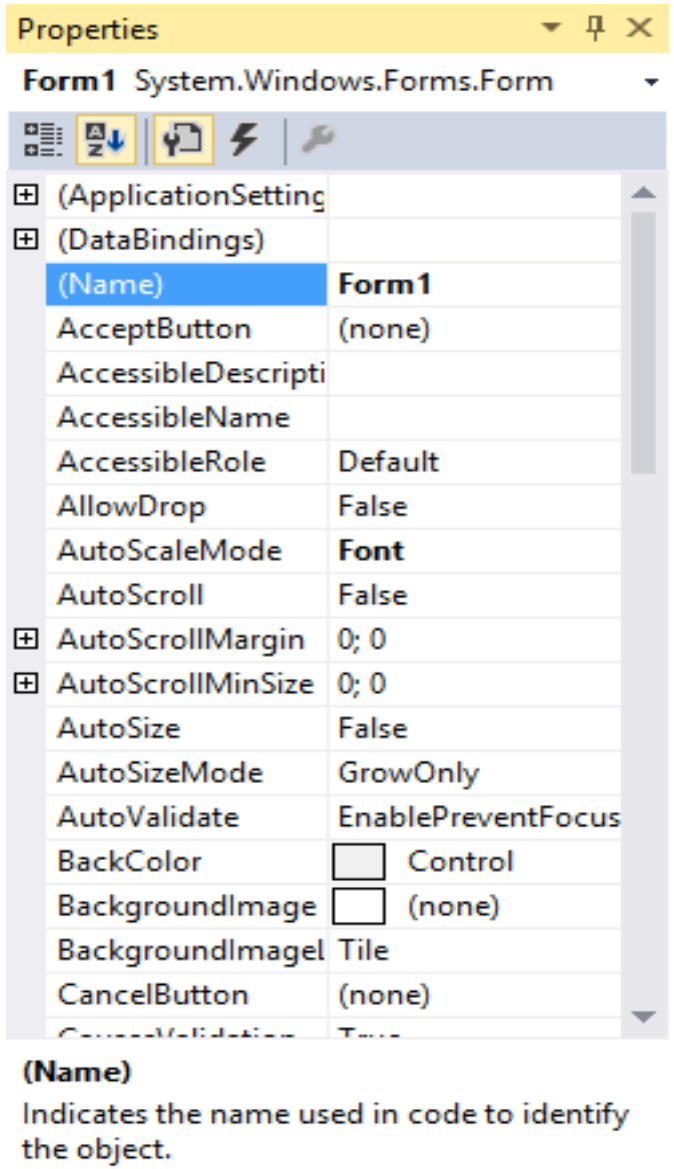
MinimumSize: задает минимальный размер формы

Opacity: задает прозрачность формы

Size: определяет начальный размер формы

StartPosition: указывает на начальную позицию, с которой форма появляется на экране

Основные свойства форм



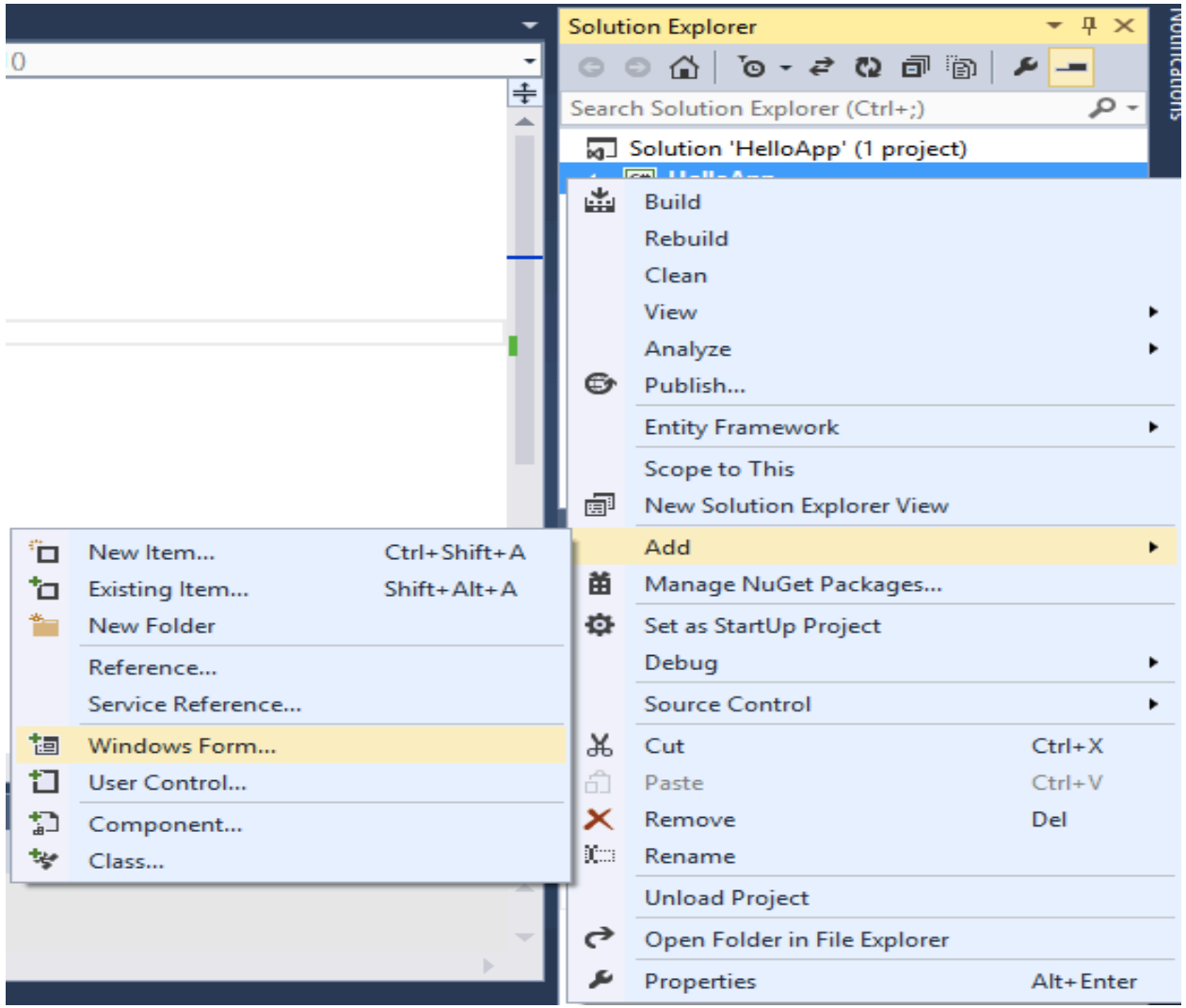
Text: определяет заголовок формы

TopMost: если данное свойство имеет значение true, то форма всегда будет находиться поверх других окон

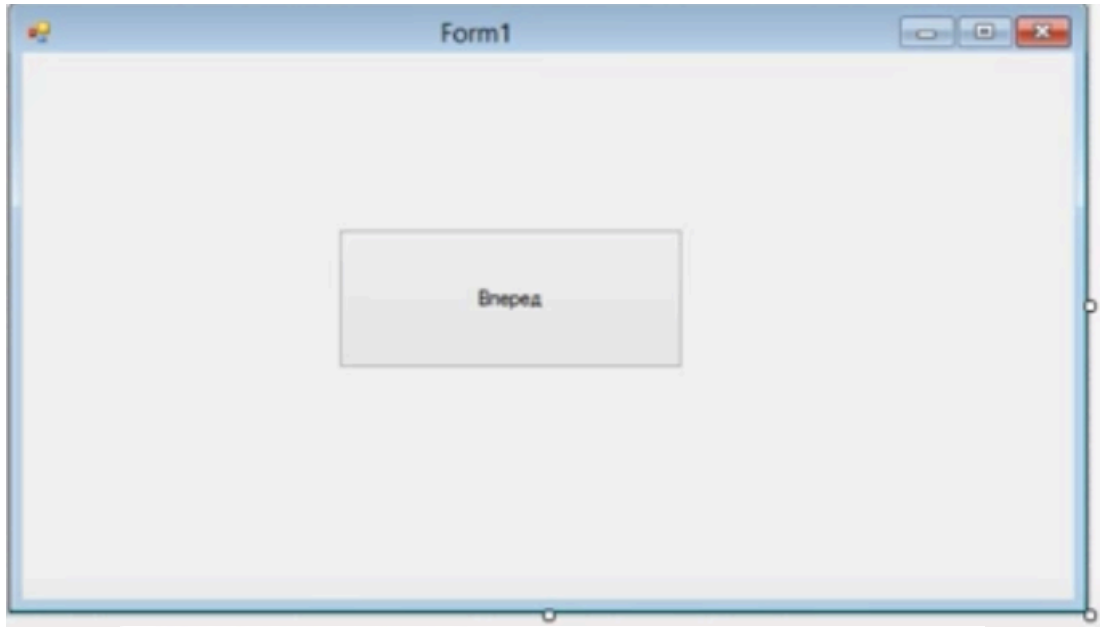
Visible: видима ли форма, если мы хотим скрыть форму от пользователя, то можем задать данному свойству значение false

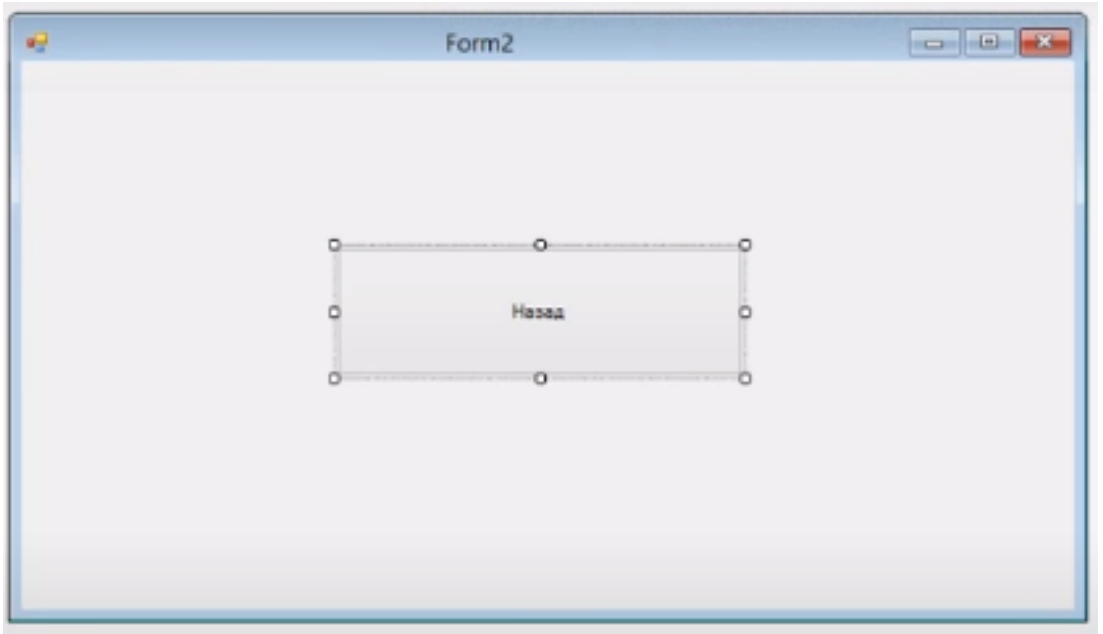
WindowState: указывает, в каком состоянии форма будет находиться при запуске: в нормальном, максимизированном или минимизированном

Добавление форм. Взаимодействие между формами



Добавление форм. Взаимодействие между формами

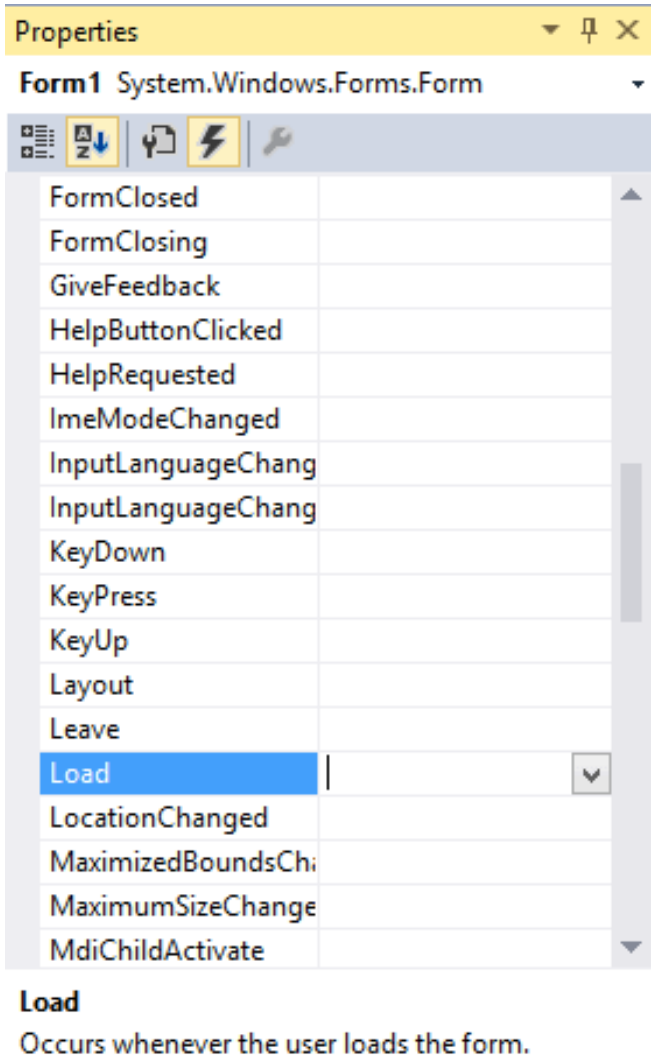


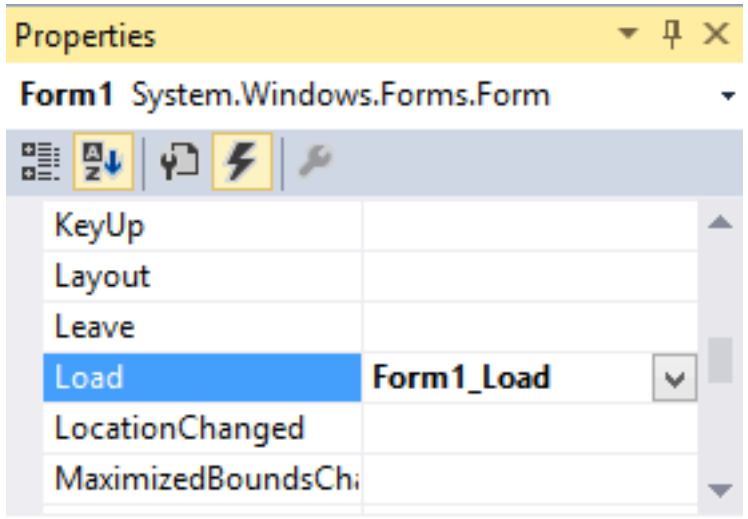



```
namespace Переход
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Form2 fr2 = new Form2();
            fr2.Show();
            Hide();
        }
    }
}
```

```
public partial class Form2 : Form
{
    public Form2()
    {
        InitializeComponent();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        Form1 fr1 = new Form1();
        fr1.Show();
        Hide();
    }
}
```

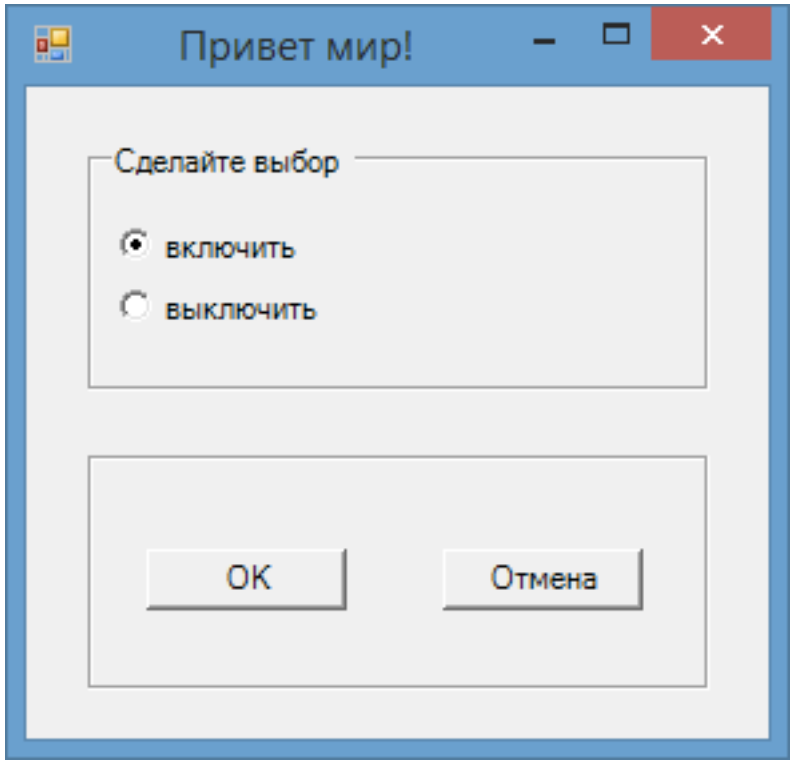
События в Windows Forms. События формы






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

    private void Form1_Load(object sender,
EventArgs e)
    {

    }
}
```

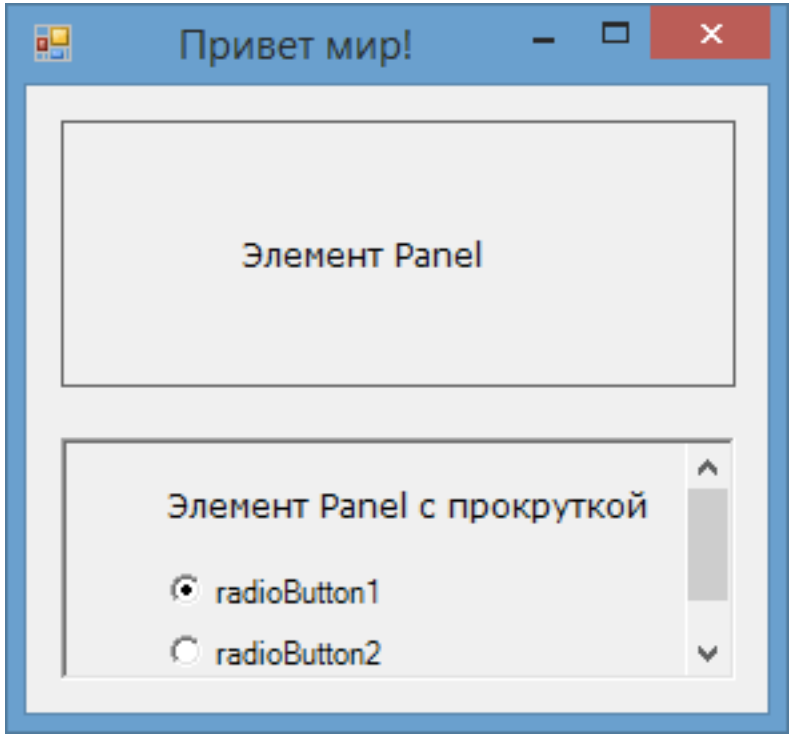
Контейнеры в Windows Forms: GroupBox



GroupBox представляет собой специальный контейнер, который ограничен от остальной формы границей. Он имеет заголовок, который устанавливается через свойство Text. Чтобы сделать GroupBox без заголовка, в качестве значения свойства Text просто устанавливается пустая строка.

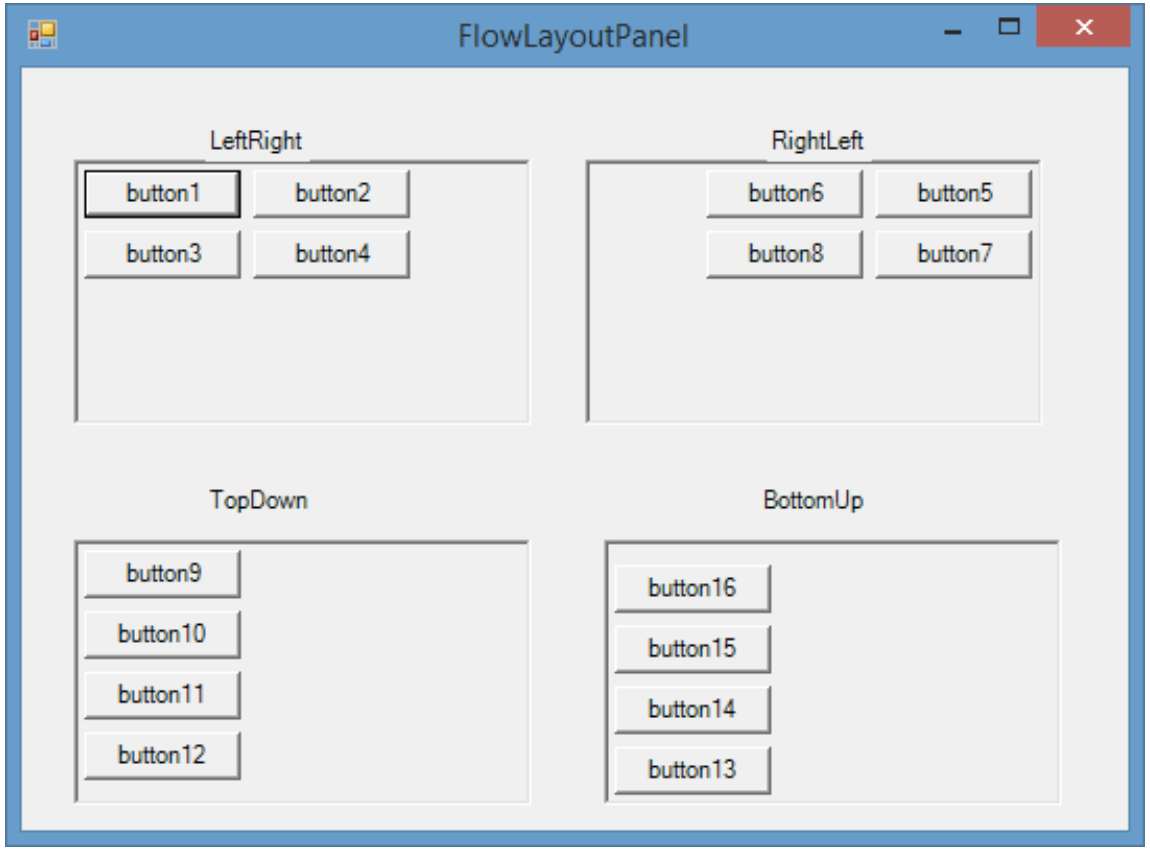
Нередко этот элемент используется для группирования переключателей - элементов RadioButton, так как позволяет разграничить их группы.

Контейнеры в Windows Forms: Panel



Элемент Panel представляет панель и также, как и GroupBox, объединяет элементы в группы. Она может визуально сливаться с остальной формой, если она имеет то же значение цвета фона в свойстве BackColor, что и форма. Чтобы ее выделить можно кроме цвета указать для элемента границы с помощью свойства BorderStyle, которое по умолчанию имеет значение None, то есть отсутствие границ.
Также если панель имеет много элементов, которые выходят за ее границы, мы можем сделать прокручиваемую панель, установив ее свойство AutoScroll в true

Контейнеры в Windows Forms: FlowLayoutPanel



Элемент FlowLayoutPanel является унаследован от класса Panel, и поэтому наследует все его свойства. Свойство элемента **FlowDirection** позволяет задать направление, в котором направлены дочерние элементы.

LeftToRight (по умолчанию) - элементы будут располагаться начиная от левого верхнего края

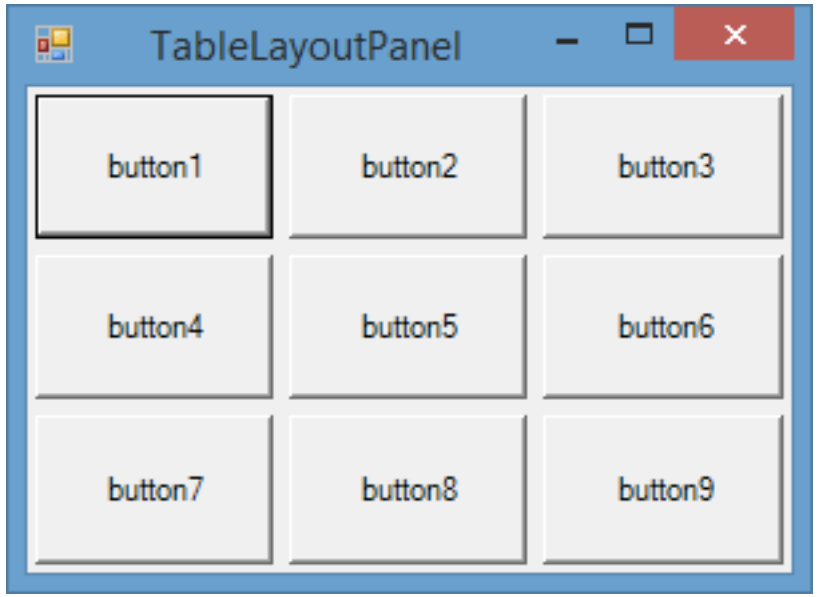
RightToLeft - элементы располагаются от правого верхнего угла в левую сторону

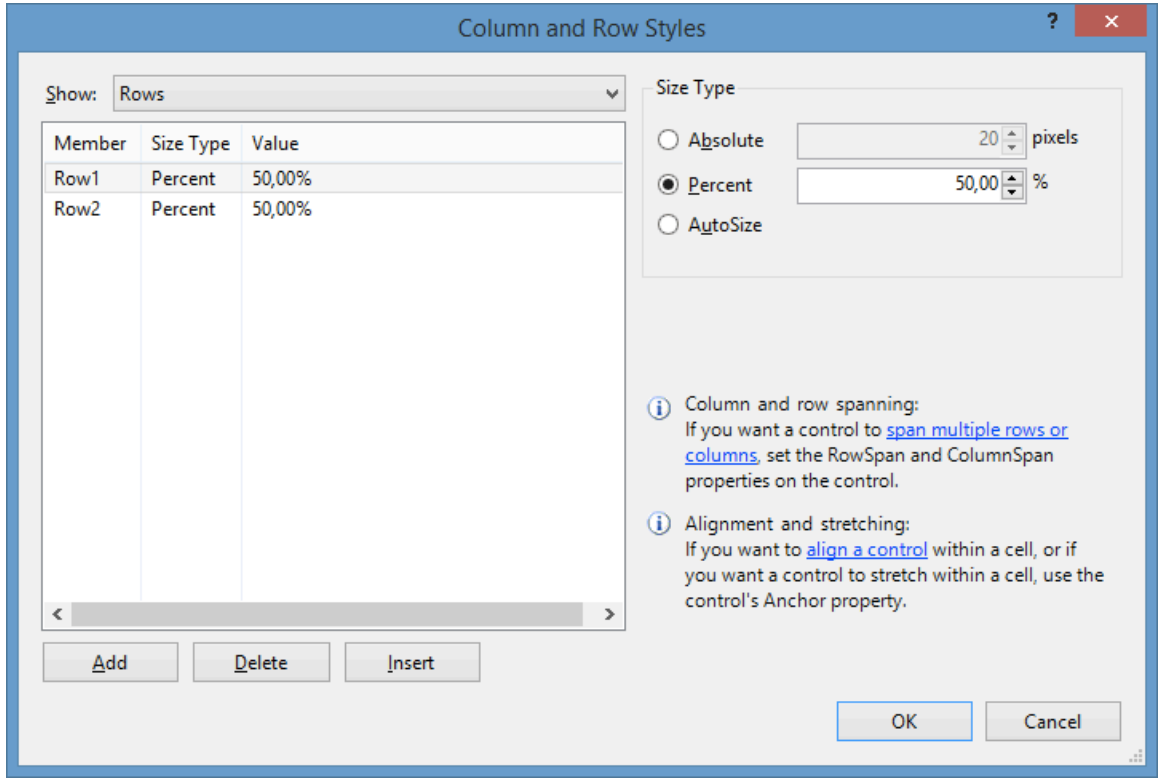
TopDown - элементы располагаются от левого верхнего угла и идут вниз

BottomUp - элементы располагаются от левого нижнего угла и идут вверх

При расположении элементов важную роль играет свойство WrapContents. По умолчанию оно имеет значение True. Это позволяет переносить элементы, которые не умещаются в FlowLayoutPanel, на новую строку или в новый столбец. Если оно имеет значение False, то элементы не переносятся, а к контейнеру просто добавляются полосы прокрутки, если свойство AutoScroll равно true.

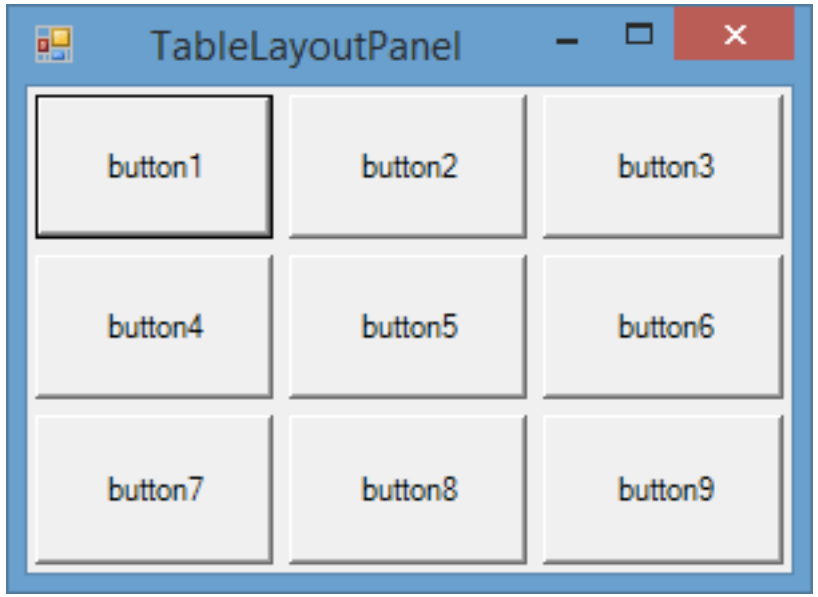
Контейнеры в Windows Forms: TableLayoutPanel

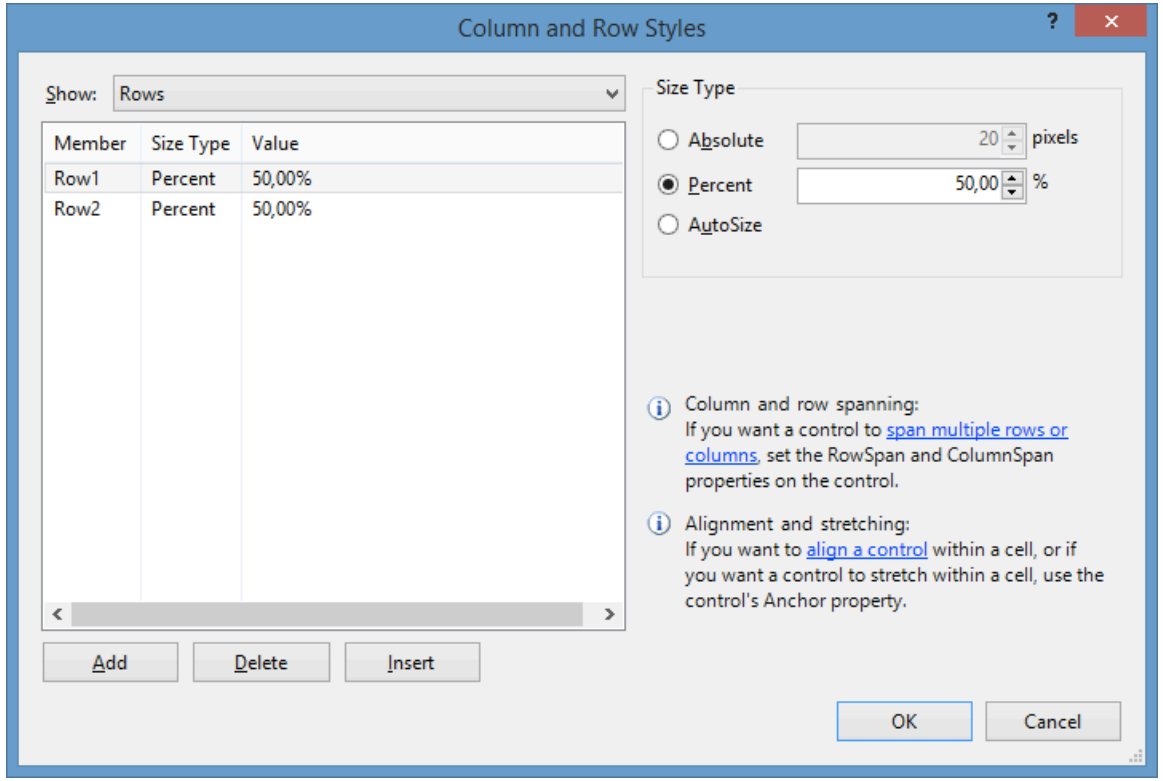




Элемент TableLayoutPanel также переопределяет панель и располагает дочерние элементы управления в виде таблицы, где для каждого элемента имеется своя ячейка. Если нам хочется поместить в ячейку более одного элемента, то в эту ячейку добавляется другой компонент TableLayoutPanel, в который затем вкладываются другие элементы.

Контейнеры в Windows Forms: TableLayoutPanel



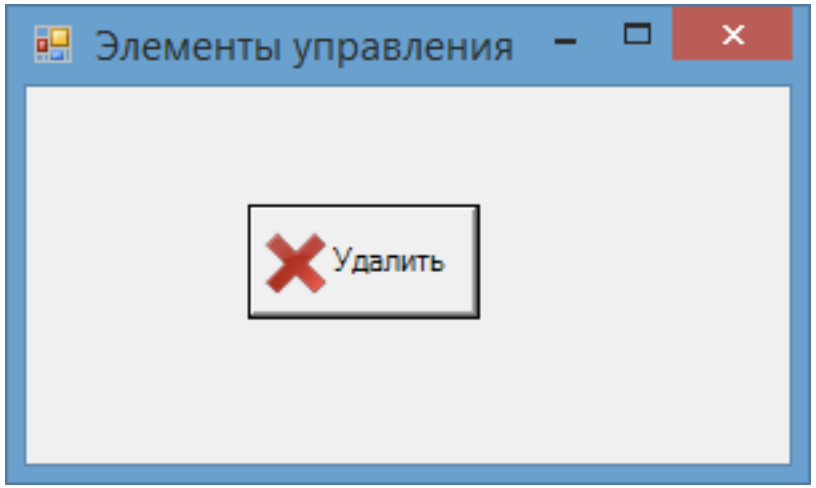


Элемент TableLayoutPanel также переопределяет панель и располагает дочерние элементы управления в виде таблицы, где для каждого элемента имеется своя ячейка. Если нам хочется поместить в ячейку более одного элемента, то в эту ячейку добавляется другой компонент TableLayoutPanel, в который затем вкладываются другие элементы.

Элементы управления в Windows Forms

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

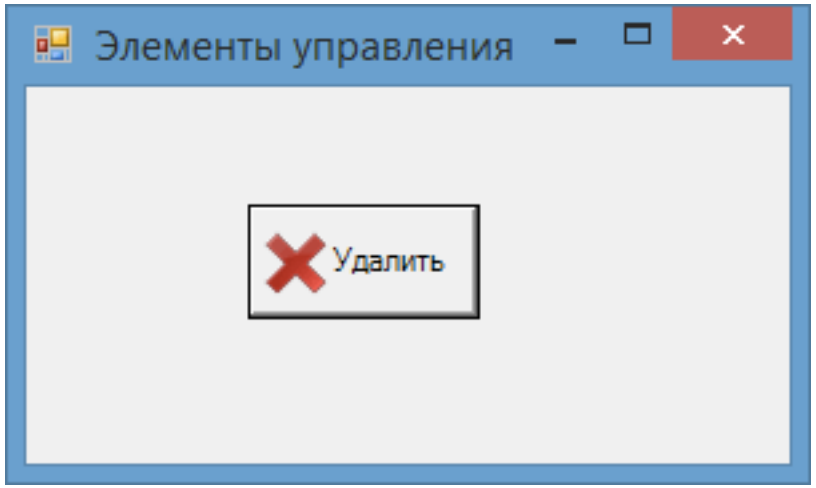
Элементы управления в Windows Forms: Кнопка



Оформление кнопки
Чтобы управлять внешним отображением кнопки, можно использовать свойство **FlatStyle**.

Оно может принимать следующие значения:
•**Flat** - Кнопка имеет плоский вид
•**Popup** - Кнопка приобретает объемный вид при наведении на нее указателя, в иных случаях она имеет плоский вид
•**Standard** - Кнопка имеет объемный вид (используется по умолчанию)
•**System** - Вид кнопки зависит от операционной системы

Элементы управления в Windows Forms: Кнопка

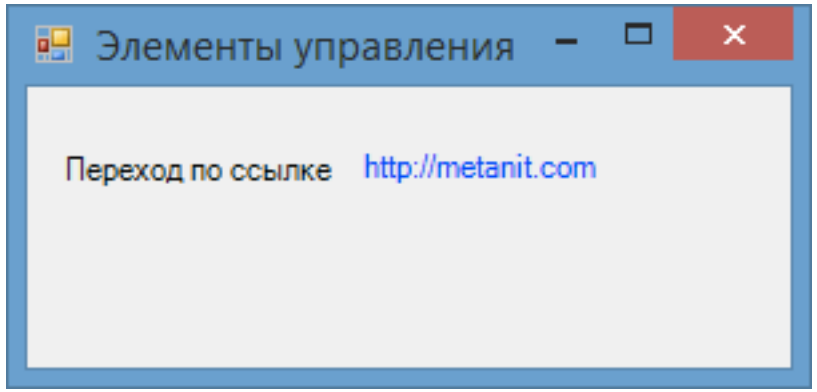


Изображение на кнопке

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:

- **Overlay**: текст накладывается на изображение
- **ImageAboveText**: изображение располагается над текстом
- **TextAboveImage**: текст располагается над изображением
- **ImageBeforeText**: изображение располагается перед текстом
- **TextBeforeImage**: текст располагается перед изображением

Элементы управления в Windows Forms: Метки и ссылки



Label

LinkLabel

Свойство **ActiveLinkColor** задает цвет ссылки при нажатии

Свойство **LinkColor** задает цвет ссылки до нажатия, по которой еще не было переходов

Свойство **VisitedLinkColor** задает цвет ссылки, по которой уже были переходы

Кроме цвета ссылки для данного элемента мы можем задать свойство **LinkBehavior**, которое управляет поведением ссылки. Это свойство принимает четыре возможных значения:

SystemDefault: для ссылки устанавливаются системные настройки

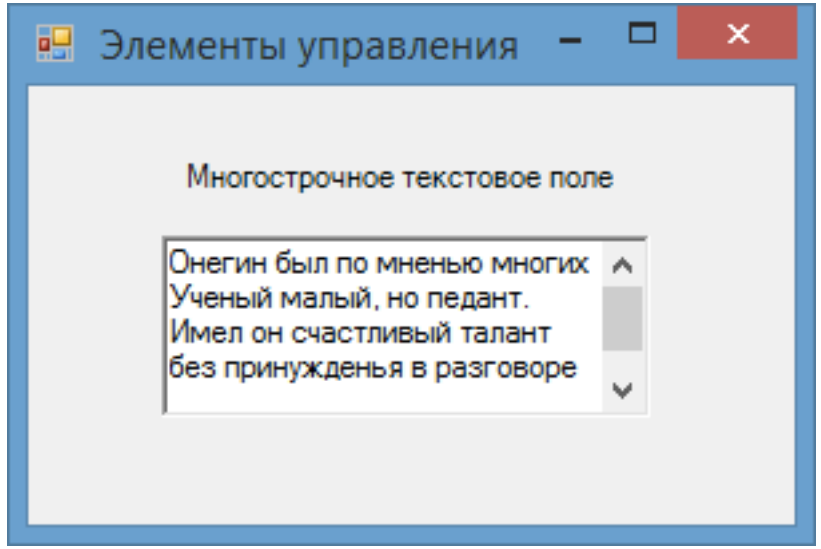
AlwaysUnderline: ссылка всегда подчеркивается

HoverUnderline: ссылка подчеркивается только при наведении на нее курсора мыши

NeverUnderline: ссылка никогда не подчеркивается

По умолчанию весь текст на данном элементе считается ссылкой. Однако с помощью свойства **LinkArea** мы можем изменить область ссылки.

Элементы управления в Windows Forms: Текстовое поле TextBox

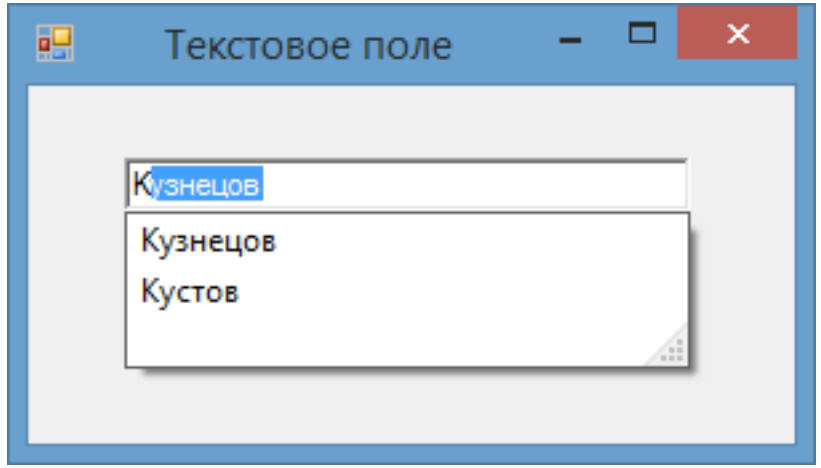


По умолчанию при переносе элемента с панели инструментов создается однострочное текстовое поле.

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

Элементы управления в Windows Forms: Текстовое поле TextBox



Режим автодополнения, представленный свойством **AutoCompleteMode**, имеет несколько возможных значений:

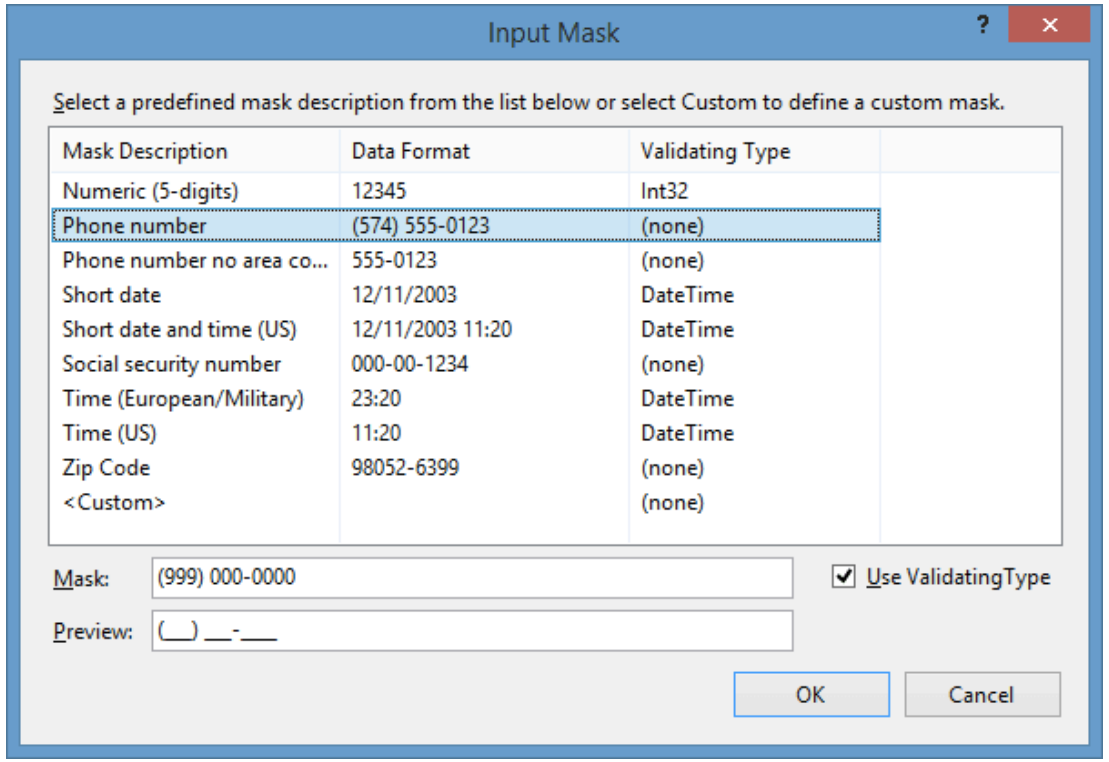
None: отсутствие автодополнения

Suggest: предлагает варианты для ввода, но не дополняет

Append: дополняет введенное значение до строки из списка, но не предлагает варианты для выбора

SuggestAppend: одновременно и предлагает варианты для автодополнения, и дополняет введенное пользователем значение

Элементы управления в Windows Forms: Маска - TextBox



Элемент MaskedTextBox по сути представляет обычное текстовое поле. Однако данные элемент позволяет контролировать ввод пользователя и проверять его автоматически на наличие ошибок.

Чтобы контролировать вводимые в поле символы, надо задать маску. Для задания маски можно применять следующие символы:

0: Позволяет вводить только цифры

9: Позволяет вводить цифры и пробелы

#: Позволяет вводить цифры, пробелы и знаки '+' и '-'

L: Позволяет вводить только буквенные символы

?: Позволяет вводить дополнительные необязательные буквенные символы

A: Позволяет вводить буквенные и цифровые символы

.: Задает позицию разделителя целой и дробной части

,: Используется для разделения разрядов в целой части числа

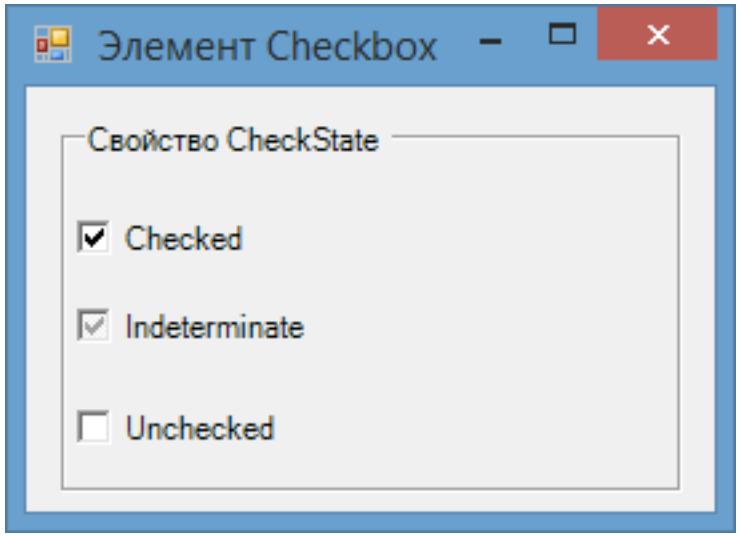
:: Используется в временных промежутках - разделяет часы, минуты и секунды

/: Используется для разделения дат

$: Используется в качестве символа валюты

Чтобы задать маску, надо установить свойство Mask элемента. Найдя это свойство в окне свойств(Porperties), нажмем на него и нам отобразится окно для задания одного из стандартных шаблонов маски.

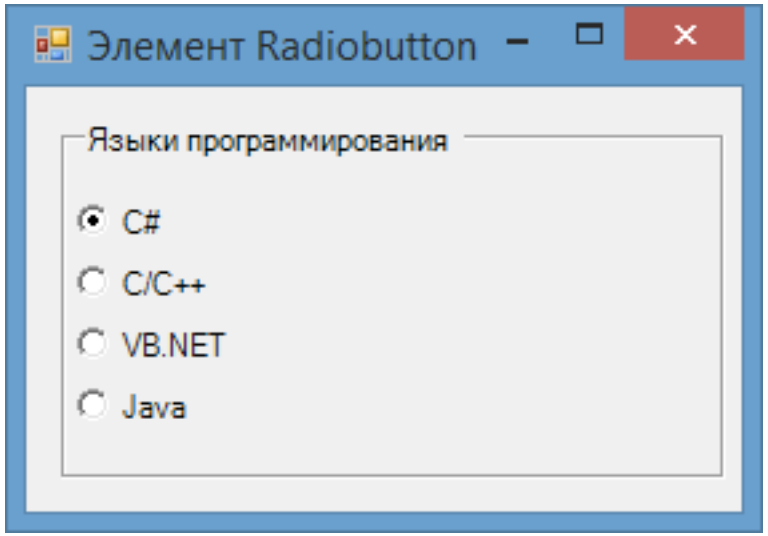
Элементы управления в Windows Forms: Radiobutton и CheckBox



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)

Элементы управления в Windows Forms: Radiobutton и CheckBox



Radiobutton

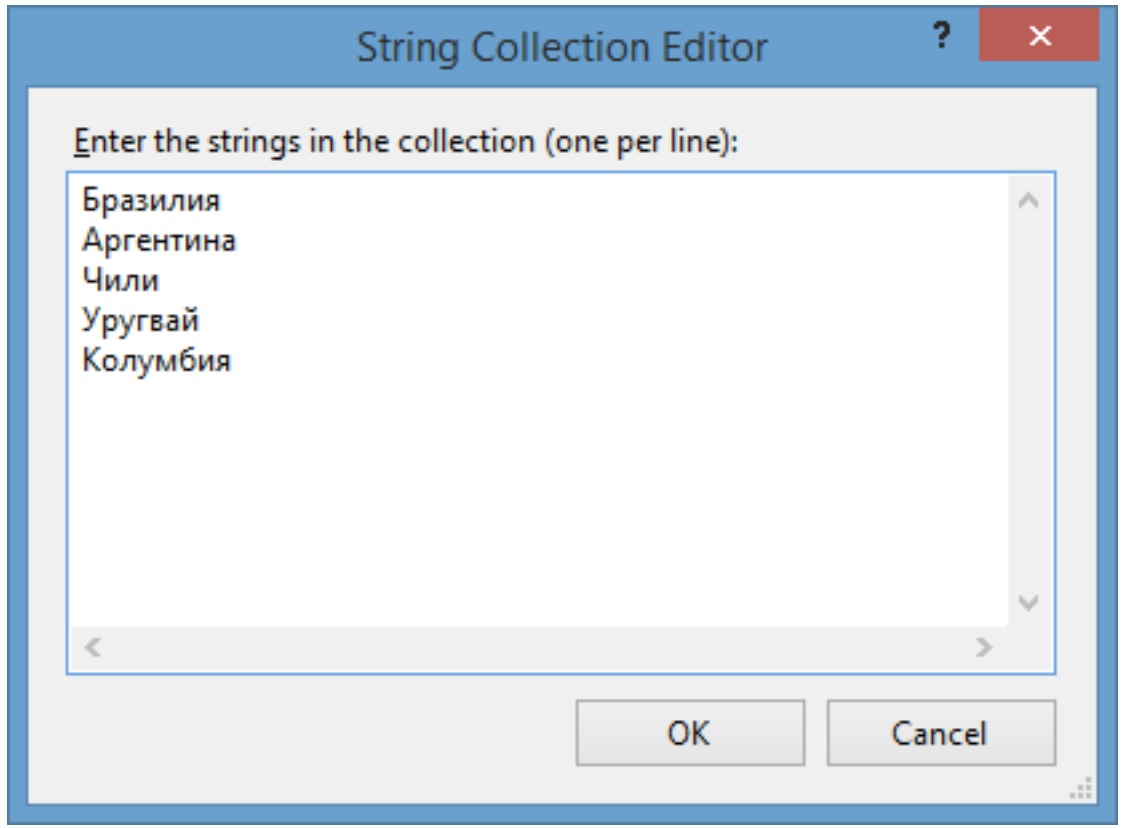
На элемент CheckBox похож элемент RadioButton или переключатель. Переключатели располагаются группами, и включение одного переключателя означает отключение всех остальных.

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

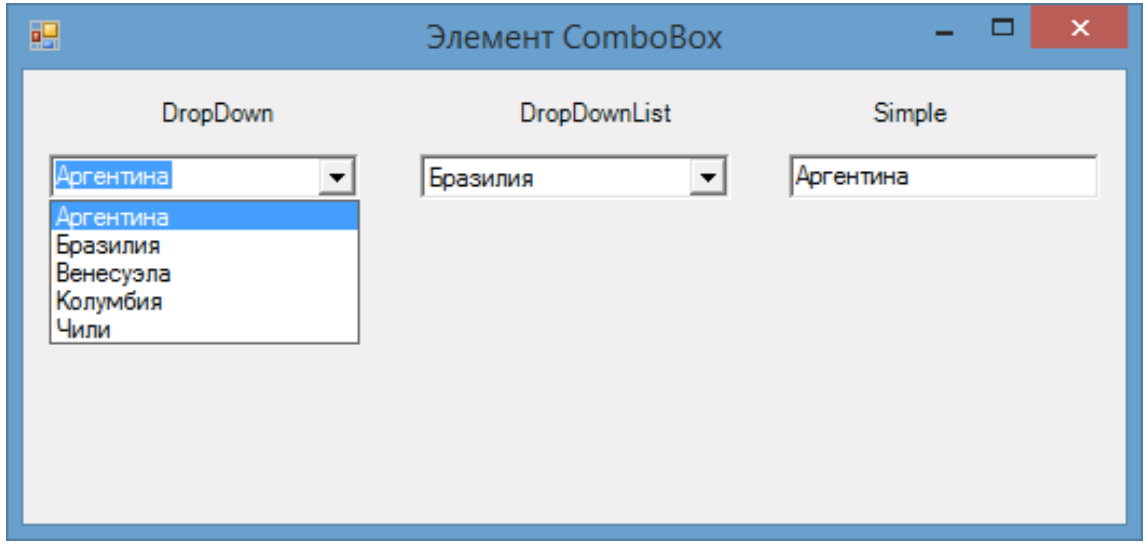
Элементы управления в Windows Forms: ListBox



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

Элементы управления в Windows Forms: ComboBox



Элемент ComboBox образует выпадающий список и совмещает функциональность компонентов ListBox и TextBox. Для хранения элементов списка в ComboBox также предназначено свойство **Items**

Свойство **MaxDropDownItems** позволяет задать число видимых элементов списка - от 1 до 100. По умолчанию это число равно 8.

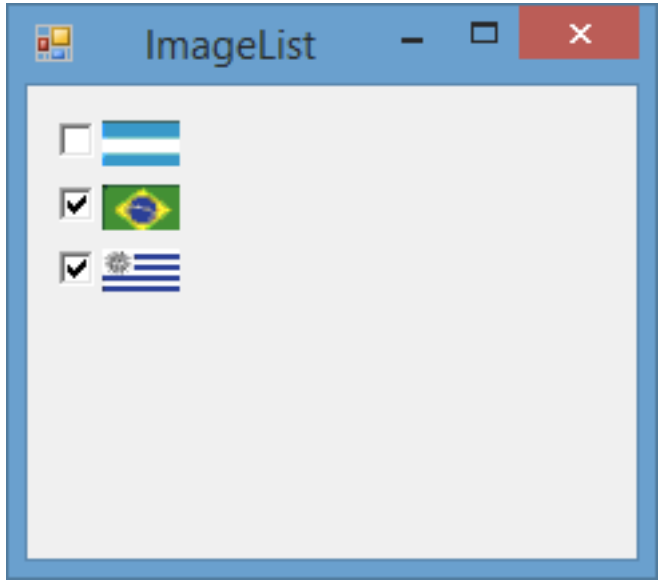
Другое свойство **DropDownStyle** задает стиль ComboBox. Оно может принимать три возможных значения:

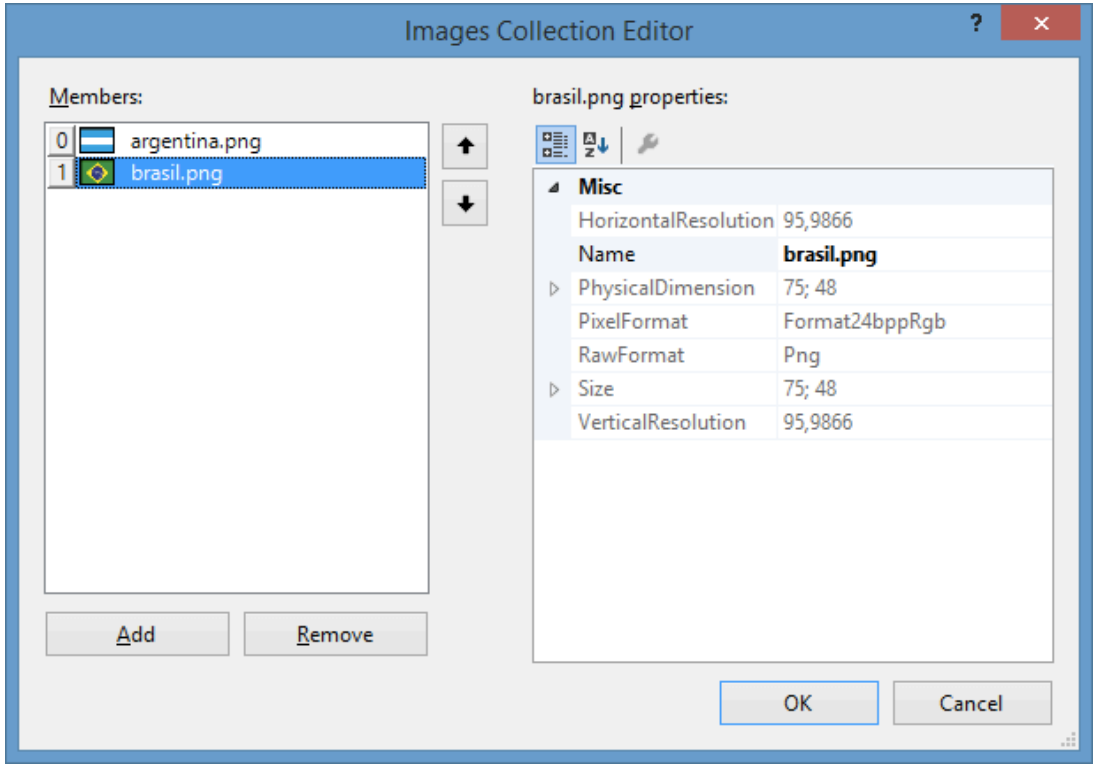
Dropdown: используется по умолчанию. Мы можем открыть выпадающий список вариантов при вводе значения в текстовое поле или нажав на кнопку со стрелкой в правой части элемента, и нам отобразится собственно выпадающий список, в котором можно выбрать возможный вариант

DropdownList: чтобы открыть выпадающий список, надо нажать на кнопку со стрелкой в правой стороне элемента

Simple: ComboBox представляет простое текстовое поле, в котором для перехода между элементами мы можем использовать клавиши клавиатуры вверх/вниз

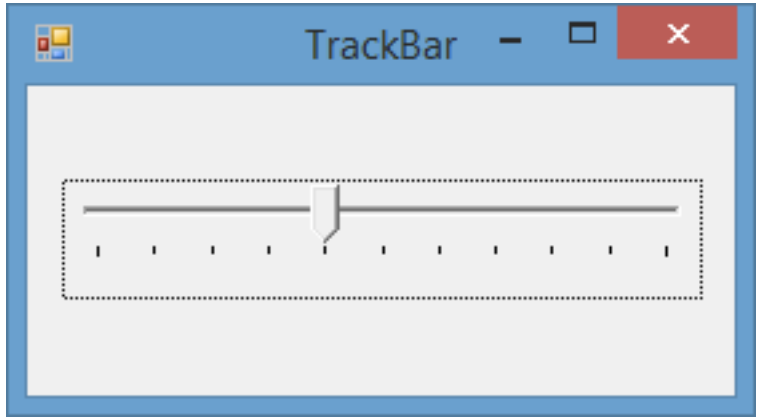
Элементы управления в Windows Forms: ImageList

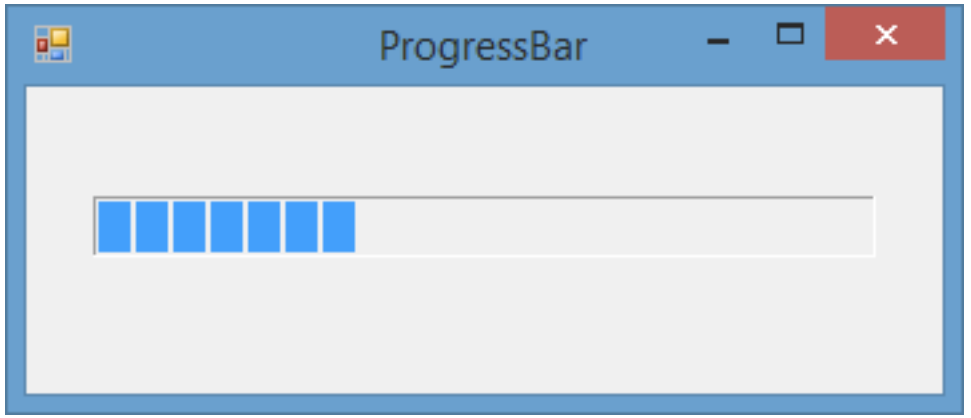


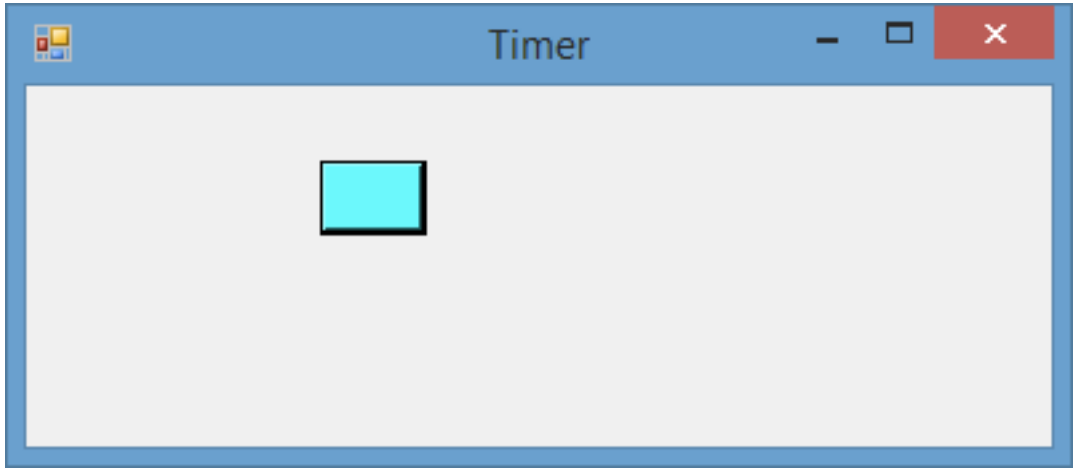


ImageList не является визуальным элементом управления, однако он представляет компонент, который используется элементами управления. Он определяет набор изображений, который могут использовать такие элементы, как ListView или TreeView.

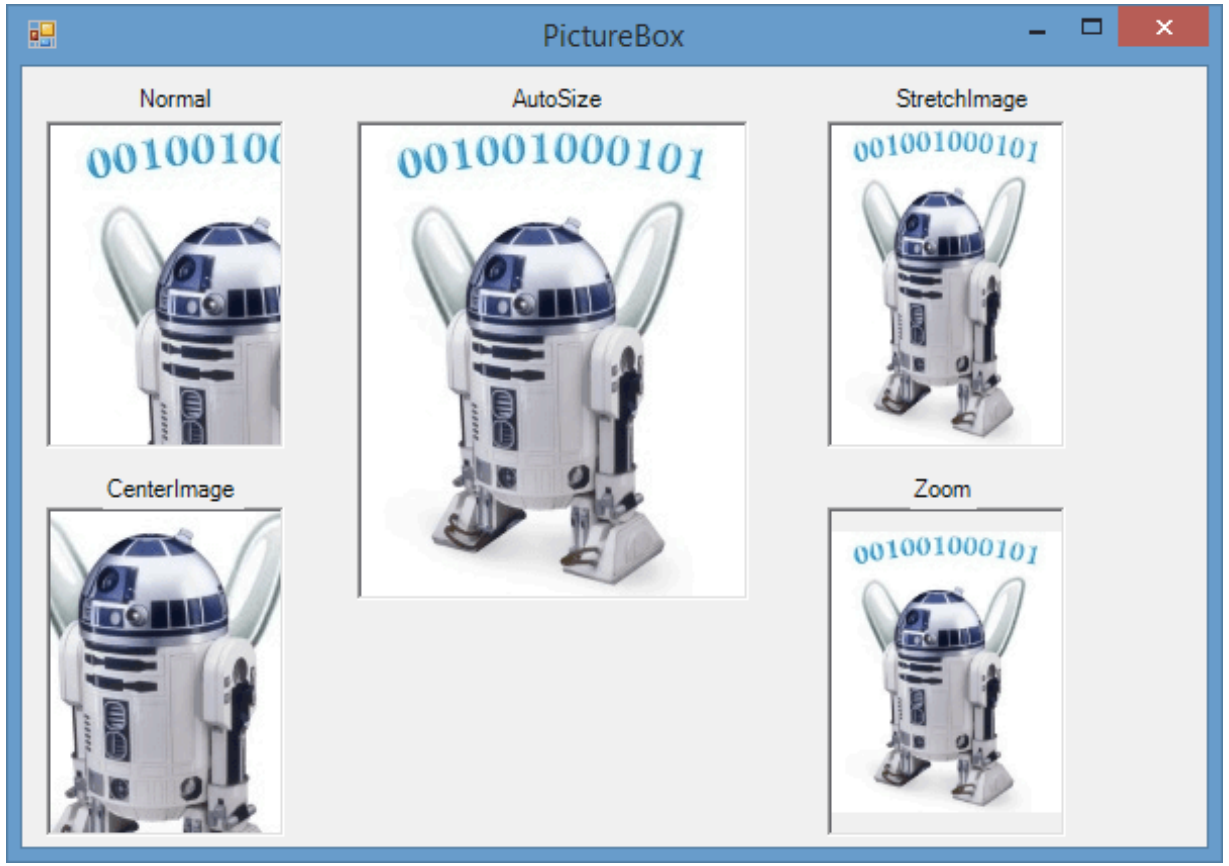
Элементы управления в Windows Forms: TrackBar, Timer и ProgressBar







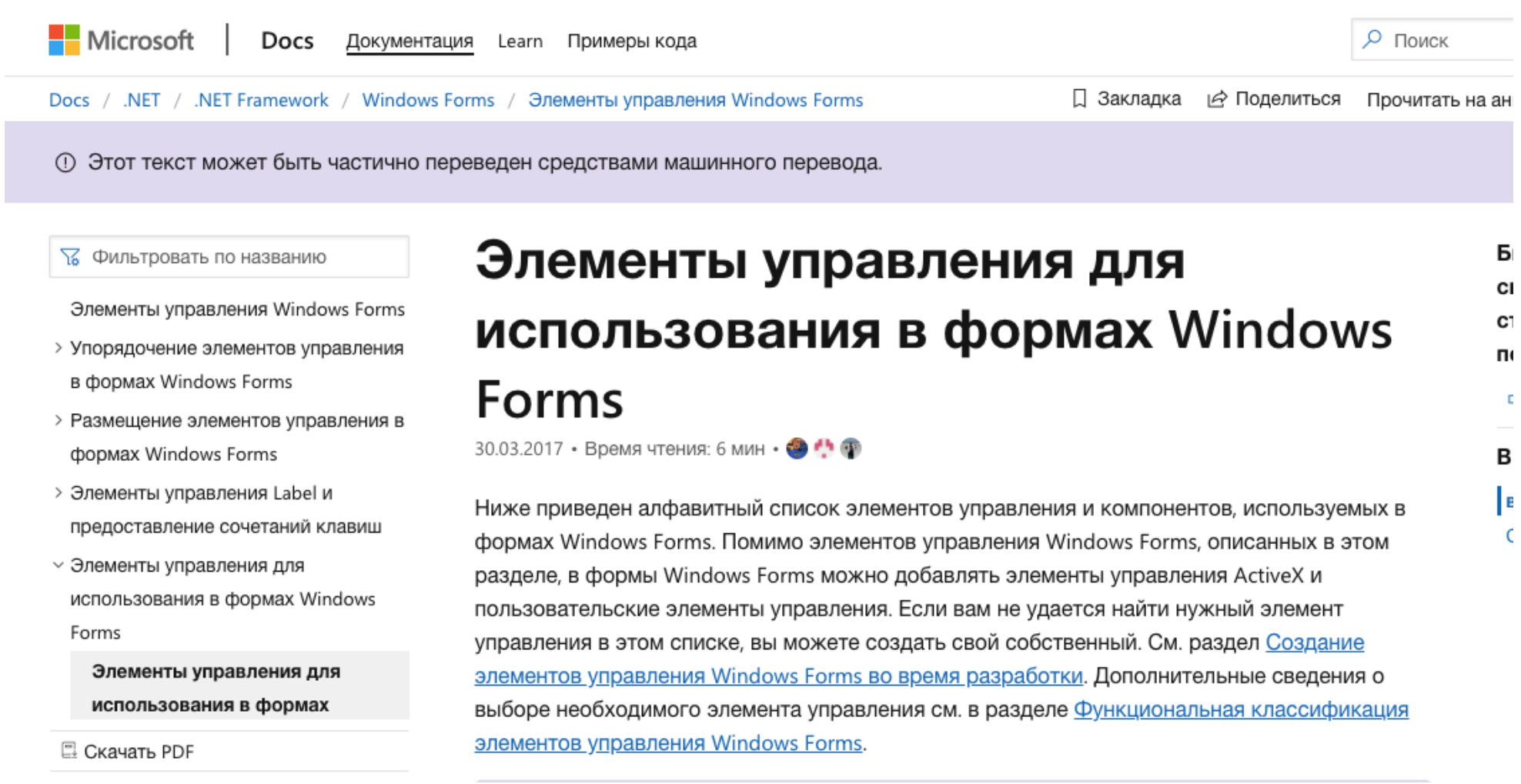
Элементы управления в Windows Forms: PictureBox



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 обраом, чобы 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

Элементы управления в Windows Forms



https://docs.microsoft.com/ru-ru/dotnet/framework/winforms/controls/controls-to-use-on-windows-forms