

Тема 2.2.

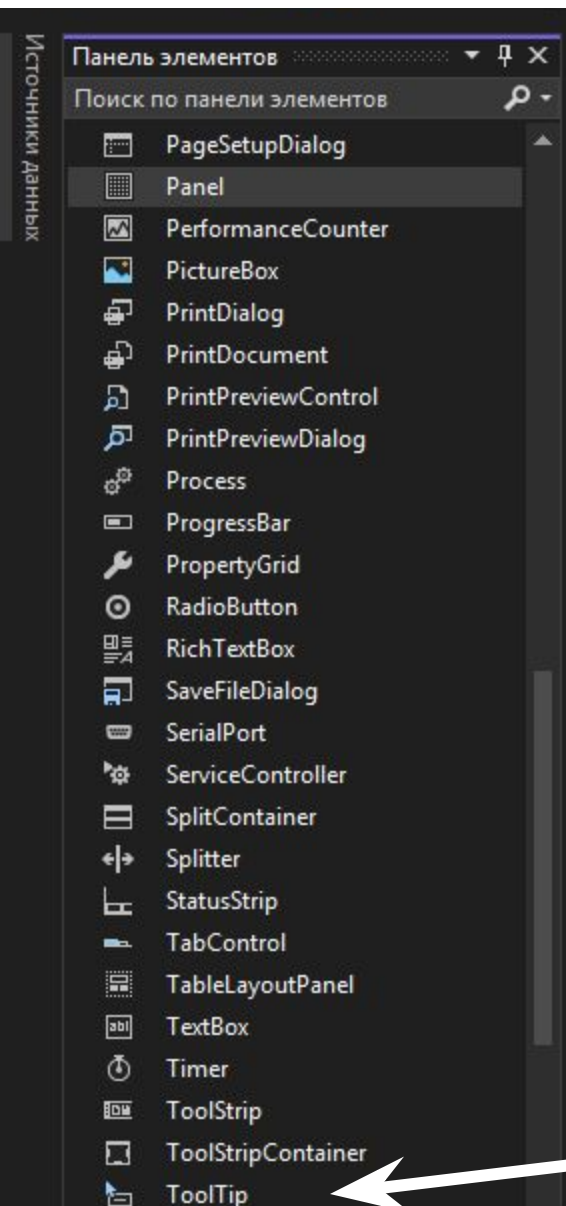
Разветвляющаяся конструкция языка C# - if

тема продолжается, только рассмотрим 2 элемента +

Вопросы:

1. Оператор if
2. Простые и сложные условия
3. Решение задач

Подсказка ToolTip в C#



Представляет небольшое прямоугольное всплывающее окно, в котором отображается краткое описание назначения элемента управления, когда пользователь наводит указатель мыши на элемент управления.

События

Disposed Возникает при удалении компонента путем вызова метода `Dispose()`.

(Унаследовано от `Component`)

Draw Происходит при отображении всплывающей подсказки, если для свойства `OwnerDraw` установлено значение `true`, а для свойства `IsBalloon` — значение `false`.

Popup Происходит перед первоначальным отображением всплывающей подсказки. Это событие по умолчанию для класса `ToolTip`.

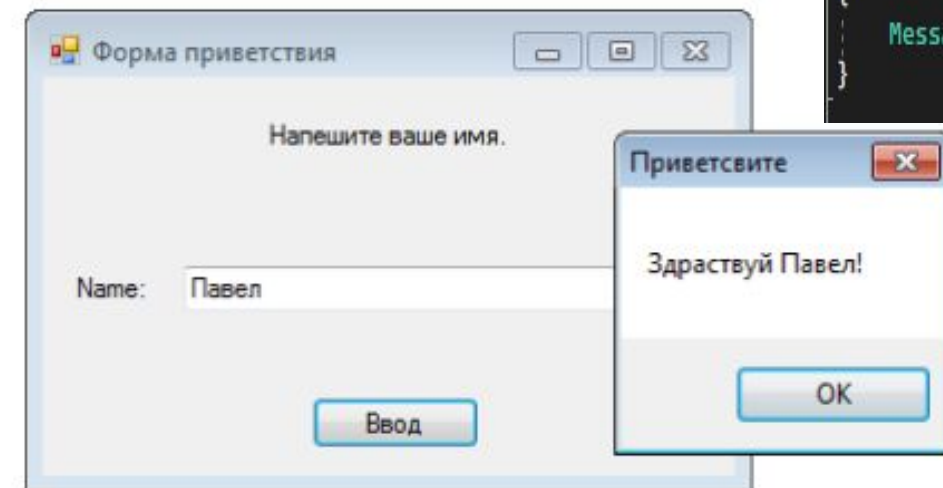
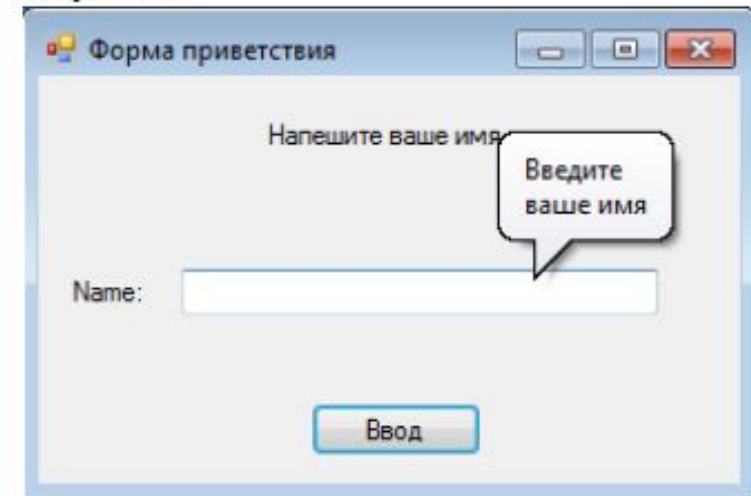
Текст подсказки не отображается для элементов управления, которые отключены. Если **ShowAlways** свойство не задано `true`, всплывающие подсказки не отображаются, если контейнер неактивен.

Класс ToolTip предоставляет следующие свойства и методы для изменения поведения и внешнего вида подсказки по умолчанию.

Категория	Связанные члены
Отображение вручную	Active, Show, Hide, ShowAlways, Popup, StopTimer
Время подсказки	AutoPopDelay, InitialDelay, ReshowDelay, AutomaticDelay, StopTimer
Content	SetToolTip, GetToolTip, StripAmpersands, ToolTipIcon, ToolTipTitle, RemoveAll
Внешний вид	BackColor, ForeColor, IsBalloon, OwnerDraw, UseAnimation, UseFading

Задание 1: Повторить пример (первый вариант)

Результат:



```
Ссылка 1
private void Form1_Layout(object sender, LayoutEventArgs e)
{
    this.Text = "Форма приветствия";
    label1.Text = "Name: ";
    label2.Text = "Напишите ваше имя.";
    button1.Text = "Ввод";
    //----- реализация Tooltip
    tooltip1.SetToolTip(textBox1, "Введите\ваше имя");
    tooltip1.IsBalloon = true;
}

Ссылка 1
private void button1_Click_1(object sender, EventArgs e)
{
    MessageBox.Show("Здравствуй " + textBox1.Text + "!", "Приветствие");
}
```

Задание 2: Повторить пример (второй вариант)

Расположить элемент `toolTip1`, выбрать элемент управления, чтобы задать подсказку, в свойстве `ToolTip` на `toolTip1` задаем текст подсказки

The screenshot shows a Visual Studio IDE with a Windows Forms application named "Form1". The form contains a text box and a button labeled "Результат". A blue arrow labeled "2" points to the text box. In the bottom-left corner, a blue arrow labeled "1" points to the `toolTip1` element in the component tray. On the right, the Properties window is open, showing the properties for `textBox1`. A blue arrow labeled "3" points to the `ToolTip на toolTip1` property, which is set to "Введите ФИО".

Обозреватель решений — поиск (Ctrl+;)

Решение "WindowsFormsApp6" (1 проекта 1)

- WindowsFormsApp6
 - Properties
 - Ссылки
 - App.config
 - Form1.cs
 - Form1.Designer.cs
 - Form1.resx
 - C# Program.cs

Обозреватель решений | Изменения Git

Свойства

textBox1 System.Windows.Forms.TextBox

TabStop	True
Tag	
Text	
TextAlign	Left
ToolTip на toolTip1	Введите ФИО
UseSystemPasswordChar	False
UseWaitCursor	False
Visible	True
WordWrap	True

ToolTip на toolTip1
Задает подсказку, отображаемую при задержке указателя мыши над элементом управления.

Элемент MaskedTextBox

Данный 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

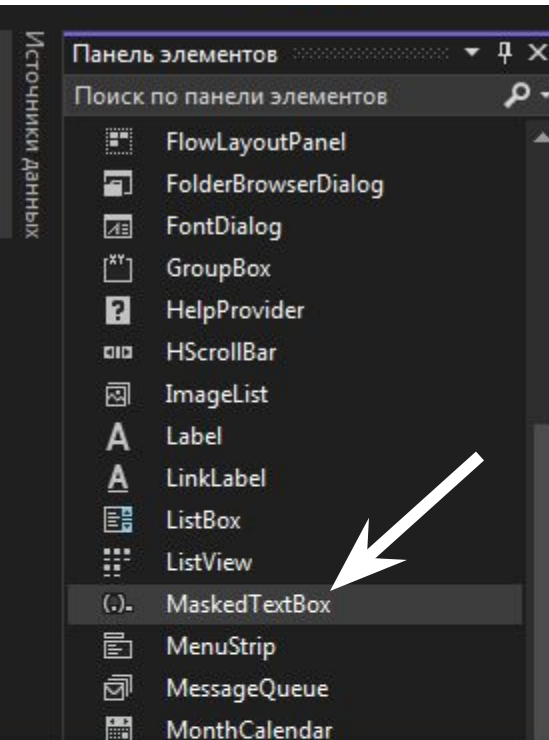
.: Задаёт позицию разделителя целой и дробной части

,: 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(Properties), нажмем на него и нам отобразится окно для задания одного из стандартных шаблонов маски.

В частности мы можем выбрать **Phone number** (Телефонный номер), который подразумевает ввод в текстовое поле только телефонного номера:

Select a predefined mask description from the list below or select Custom to define a custom mask.

Mask Description	Data Format	Validating Type
Numeric (5-digits)	12345	Int32
Phone number	(574) 555-0123	(none)
Phone number no area co...	555-0123	(none)
Short date	12/11/2003	DateTime
Short date and time (US)	12/11/2003 11:20	DateTime
Social security number	000-00-1234	(none)
Time (European/Military)	23:20	DateTime
Time (US)	11:20	DateTime
Zip Code	98052-6399	(none)
<Custom>		(none)

Mask: (999) 000-0000 Use ValidatingType

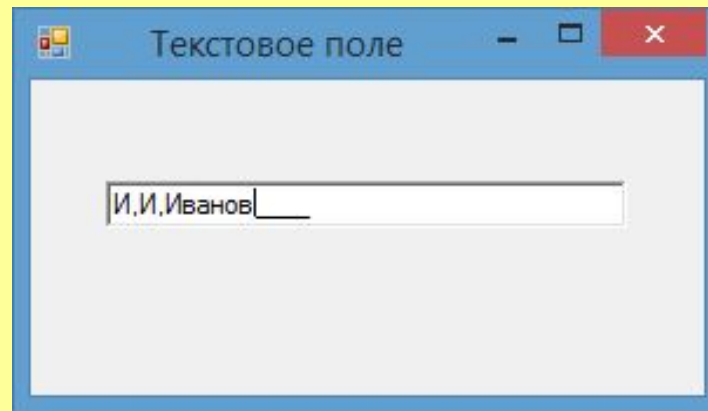
Preview: () - -

OK Cancel

Теперь при запуске мы сможем ввести в текстовое поле только цифры, получив в итоге телефонный номер.

Задание 2: Повторить маску ввода

Теперь сделаем свою маску. Например, создадим маску для ввода инициалов имени и отчества и фамилий ограниченной длины в текстовое поле. Для этого присвоим свойству `Mask` значение `L.L.L????????`. Тогда ввод в текстовое поле будет выглядеть следующим образом:



Данный элемент также представляет нам ряд свойств, которые можно использовать для управления вводом. Так, свойство **BeepOnError** при установке значения `true` подает звуковой сигнал при введении некорректного символа.

Свойство **HidePromptOnLeave** при установке в `true` при потере текстовым полем фокуса скрывает, указанные в `PromptChar`

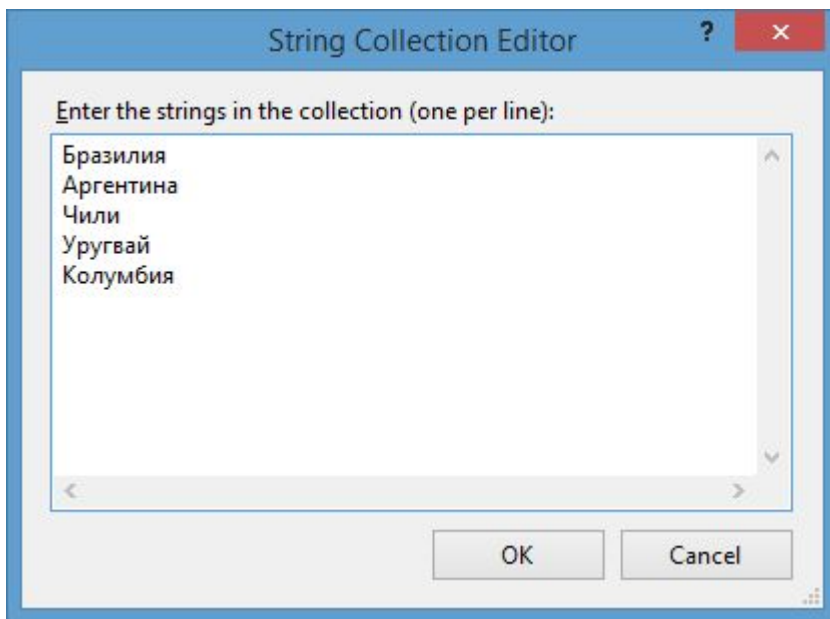
Свойство **PromptChar** указывает на символ, который отображается в поле на месте ввода символов. По умолчанию стоит знак подчеркивания.

Свойство **AsciiOnly** при значении `true` позволяет вводить только `asci`-символы, то есть символы из диапазона `A-Z` и `a-z`.

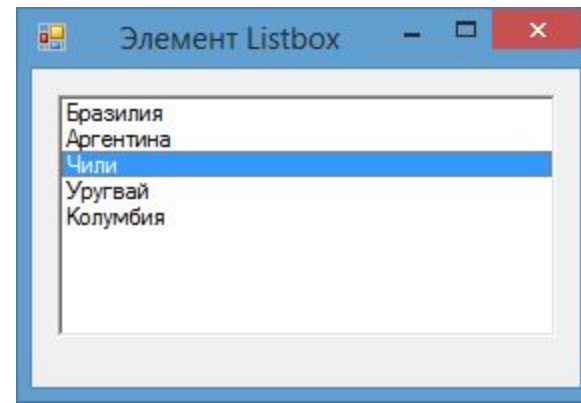
Элемент ListBox

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:



Выделение элементов списка

При выделении элементов списка мы можем ими управлять как через индекс, так и через сам выделенный элемент. Получить выделенные элементы можно с помощью следующих свойств элемента `ListBox`:

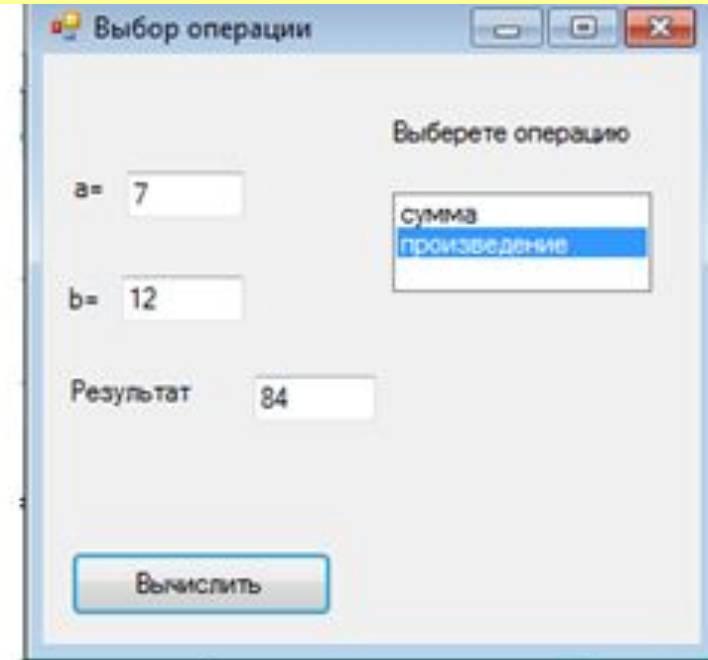
- **SelectedIndex**: : возвращает или устанавливает номер выделенного элемента списка. Если выделенные элементы отсутствуют, тогда свойство имеет значение -
- **SelectedIndices**:: возвращает или устанавливает коллекцию выделенных элементов в виде набора их индексов
- **SelectedItem**: возвращает или устанавливает текст выделенного элемента
- **SelectedItems**: : возвращает или устанавливает выделенные элементы в виде коллекции о умолчанию список поддерживает выделение одного элемента. Чтобы добавить возможность выделения нескольких элементов, надо установить у его свойства `SelectionMode` значение `MultiSimple`.

Чтобы выделить элемент программно, надо применить метод `SetSelected(int index, bool value)`, где `index` - номер выделенного элемента. Если второй параметр - `value` имеет значение `true`, то элемент по указанному индексу выделяется, если `false`, то выделение наоборот скрывается

Чтобы снять выделение со всех выделенных элементов, используется метод `ClearSelected`.

Задание №3: Произвести вычисления, после выбора операции, повторить

```
private void button1_Click(object sender, EventArgs e)
{
double a = Convert.ToDouble(textBox1.Text);
double b = Convert.ToDouble(textBox2.Text);
double c = 0;
string z = listBox1.SelectedItem.ToString();
if (z == "сумма")
c = a + b;
if (z == "произведение")
c = a * b;
textBox3.Text = c.ToString();
}
```



Здесь в четвертой строке тела метода использовано свойство SelectedItem, которое соответствует выбранному члену списка. Этот член списка преобразуется в строку, и полученное значение присваивается переменной z. Затем в зависимости от значения z выполняется сложение или умножение.