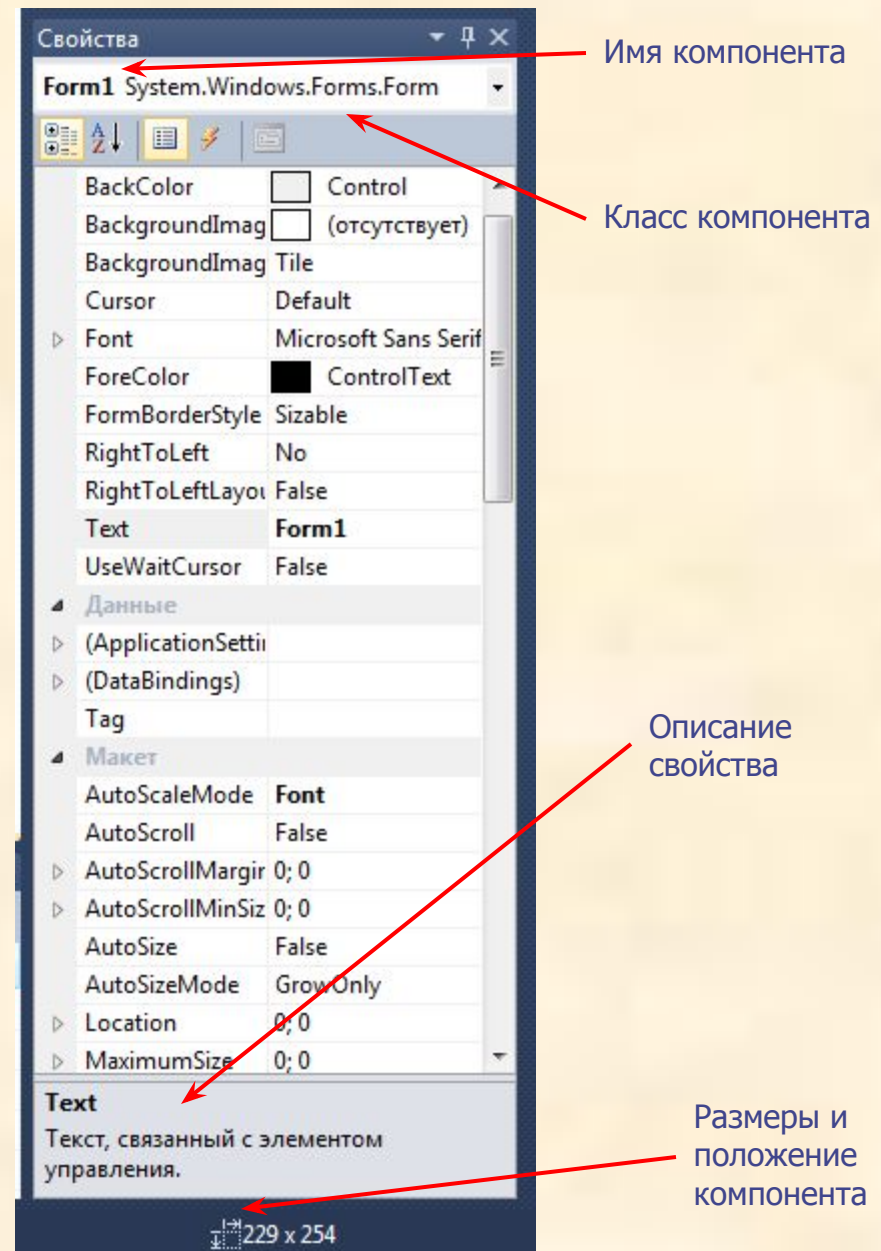
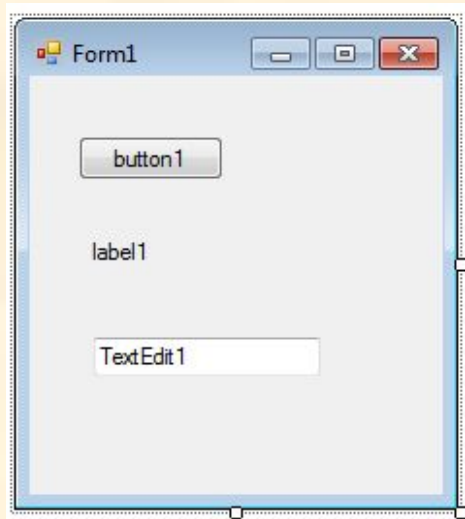


3. Свойства визуальных компонентов.

Будем разбирать на примере
формы **Form1**
кнопки **Button1**
надписи **Label1**
текстового окна **TextBox1**



Свойства

Form1 System.Windows.Forms.Form

Имя компонента

Класс компонента

BackColor	Control
BackgroundImage	(отсутствует)
BackgroundImageTiled	
Cursor	Default
Font	Microsoft Sans Serif
ForeColor	ControlText
FormBorderStyle	Sizable
RightToLeft	No
RightToLeftLayout	False
Text	Form1
UseWaitCursor	False

Данные

Макет

AutoScaleMode	Font
AutoScroll	False
AutoScrollMargin	0; 0
AutoScrollMinSize	0; 0
AutoSize	False
AutoSizeMode	GrowOnly
Location	0; 0
MaximumSize	0; 0

Text

Текст, связанный с элементом управления.

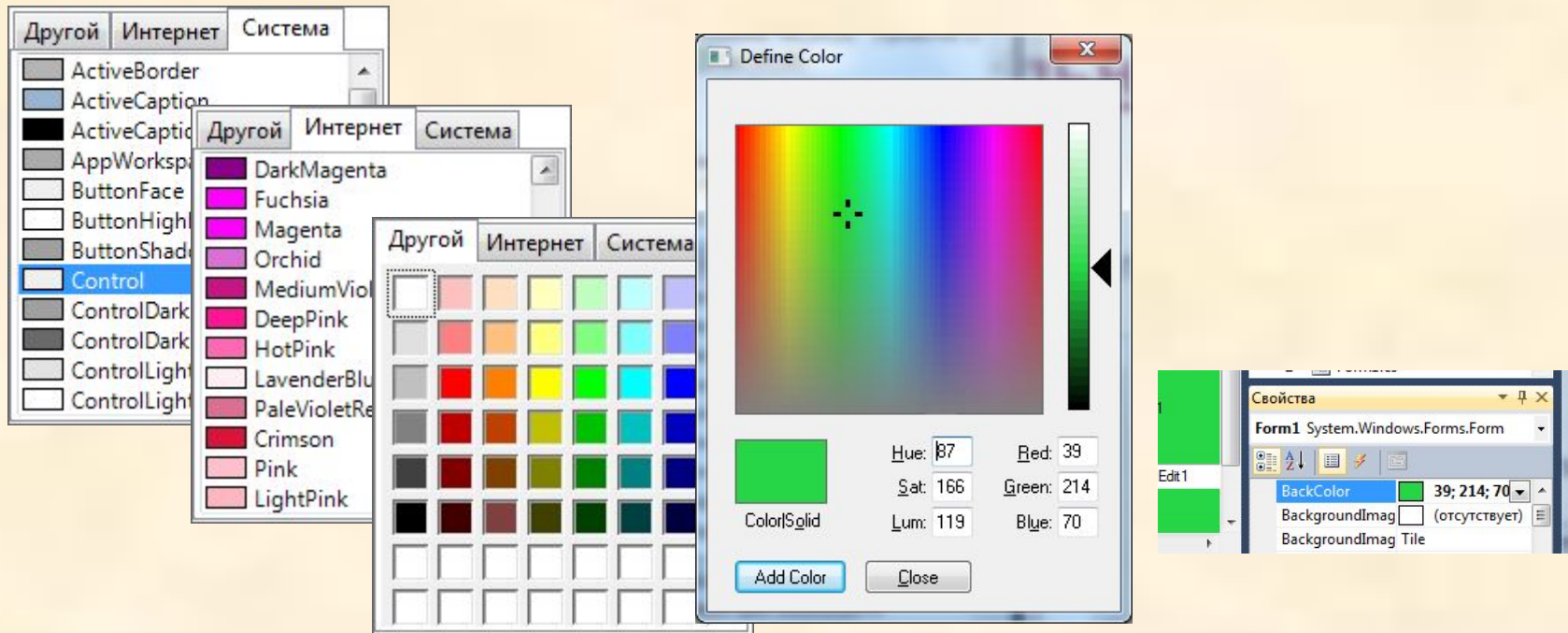
Описание свойства

Размеры и положение компонента

229 x 254

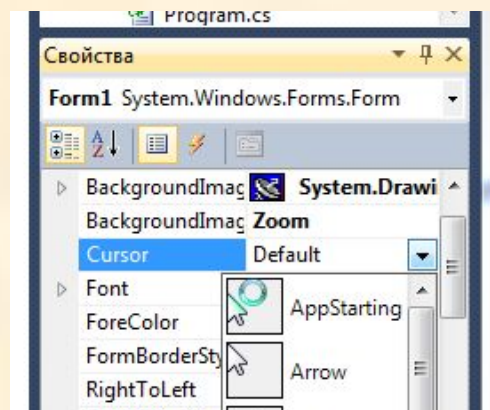
1. Внешний вид компонента

BackColor – цвет фона компонента



```
this.BackColor = Color.FromArgb(100, 50, 200);  
BackColor = Color.Fuchsia;
```

Cursor – Рисунок курсора.



```
19 private void Form1_Load(object sender, EventArgs e)
20 {
21     this.Cursor=Curs
22 }
23 }
24 }
25 }
```

class System.Windows.Forms.Cursors
Предоставляет коллекцию объектов System.Windows.Forms.Cursor для использования приложением Windows Forms.

- Cursor
- CursorChanged
- CursorConverter
- Cursors
- DefaultCursor
- OnCursorChanged
- OnParentCursorChanged
- UseWaitCursor

```
this.Cursor = Cursors.Hand;
```

Text – Название формы (текст на поверхности компонента).

```
this.Text = "Моя формочка";
```

Text – Название формы (текст на поверхности компонента).

```
this.Text = "Моя формочка«;
```

ForeColor – цвет текста на поверхности компонента.

```
textBox1.ForeColor = Color.Blue; // цвет текста
```

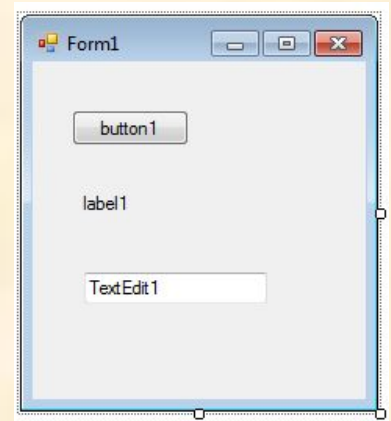
FormBorderStyle – внешний стиль формы.

Sizeable – изменяемые размеры

FixedSingle – неизменяемые размеры

FixedDialog – неизменяемые размеры и нет системной кнопки

None – форма без границ



```
this.FormBorderStyle = FormBorderStyle.None;
```

Font – шрифт текста на поверхности компонента.

Подсвойство	Тип	Описание
Style	FontStyle	Стиль шрифта. Содержит значения: Regular, <u>Bold</u> , <u>Italic</u> , <u>Underline</u> , <u>Strikeout</u> , которые можно накапливать.
Name	string	Гарнитура шрифта.
Size	float	Размер шрифта в пунктах.

Изменить шрифт программно можно только, **создав новый шрифт** (например на основе текущего).

```
label1.Font = new Font("Consolas", 20.0F,  
    FontStyle.Bold | FontStyle.Italic);
```

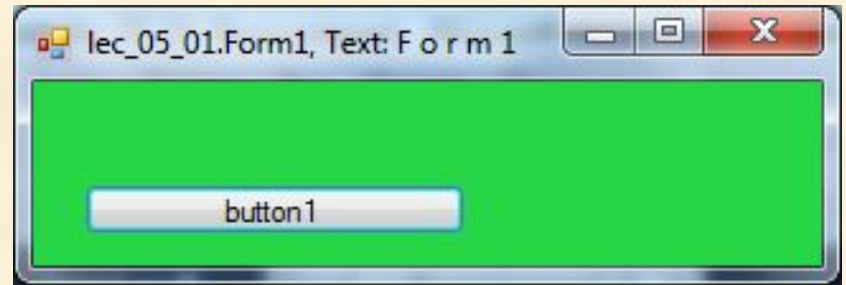
```
label1.Font = new Font(label1.Font, FontStyle.Underline);
```

2. Положение и размеры компонента

Свойство	Тип	Описание
Location	Point (X,Y)	Представляет координаты левого верхнего угла компонента относительно левого верхнего угла контейнера
Size	Size (Width, Height)	Ширина и высота компонента
Left	int	Расстояние между левой границей компонента и левой границей клиентской области его контейнера
Right	int	Расстояние между правой границей компонента и левой границей клиентской области его контейнера
Width	int	Ширина компонента
Top	int	Расстояние между верхней границей компонента и верхней границей клиентской области контейнера
Bottom	int	Расстояние между нижней границей компонента и верхней границей клиентской области контейнера
Height	int	Высота компонента
ClientSize	Size	Ширина и высота клиентской области компонента

Примеры

```
button1.Location = new Point(10, 40);  
button1.Size = new Size(100, 20);  
button1.Left = 20;  
button1.Width += 50;
```



MinimumSize (тип Size) – минимальные размеры компонента, по умолчанию (0; 0).

MaximumSize (тип Size) – максимальные размеры компонента, по умолчанию (0; 0).

```
this.MaximumSize = new Size(300, 100);
```

StartPosition – начальная позиция формы.

CenterScreen – в центре экрана

Manual – в соответствии с **Location**

WindowState – состояние формы.

Normal – обычная форма

Minimized – свернутая форма

Maximized – распахнутая форма

3. Другие свойства компонентов

Свойство	Тип	Описание
Enabled	bool	Признак работоспособности компонента
Visible	bool	Признак видимости компонента
Focused	bool	Признак нахождения компонента «в фокусе»
Name	string	Имя компонента в программе
Tag	int	Свободная величина

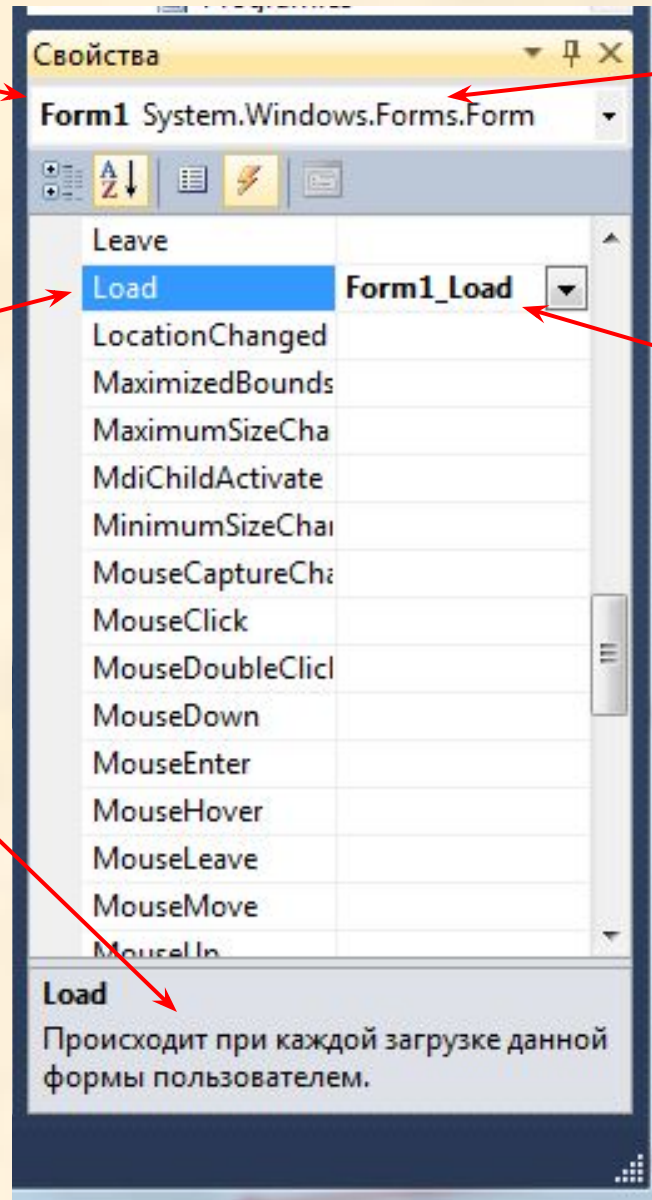
4. Особые свойства формы

Свойство	Тип	Описание
Icon	Icon	Значок для формы
KeyPreview	bool	True - форма просматривает события клавиатуры подчинённых компонент
Opacity	double	Уровень <u>непрозрачности</u> для формы (в %). Значение по умолчанию — 1,00 (100%)
ShowInTaskbar	bool	True , если форма должна отображаться на панели задач Windows во время выполнения (по умолчанию является true).
ShowIcon	bool	Отображается ли иконка
MinimizeBox	bool	Отображается ли кнопка «свернуть»
MaximizeBox	bool	Отображается ли кнопка «распахнуть»

4. События визуальных компонентов.

Имя компонента

Класс компонента



Имя события

Имя функции
– обработчика
события

Описание события

1. События мыши

При **простом** щелчке мышкой компонент генерирует события:

Событие **MouseDown**.

Событие **Click**.

Событие **MouseClick**.

Событие **MouseUp**.

При **двойном** щелчке мышкой компонент генерирует события:

Событие **MouseDown**.

Событие **Click**.

Событие **MouseClick**.

Событие **MouseUp**.

Событие **MouseDown**.

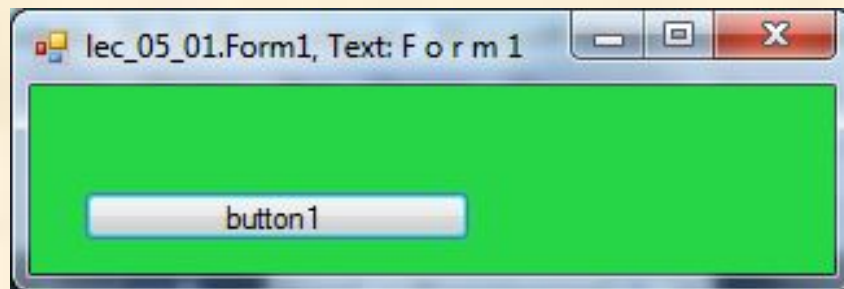
Событие **DoubleClick**.

Событие **MouseDoubleClick**.

Событие **MouseUp**.

1. Click – щелчок по компоненту (мышкой или клавишей ENTER).

```
private void Form1_Click(object sender, EventArgs e)
{
    Text = sender.ToString();
}
```



sender – во всех событиях представляет объект - источник события.

Методы: ToString() – информация об объекте (string)
Equals(any) – равен ли объекту any (bool)
GetType() – тип объекта (type).

e – описание события (тип EventArgs – самый простой).

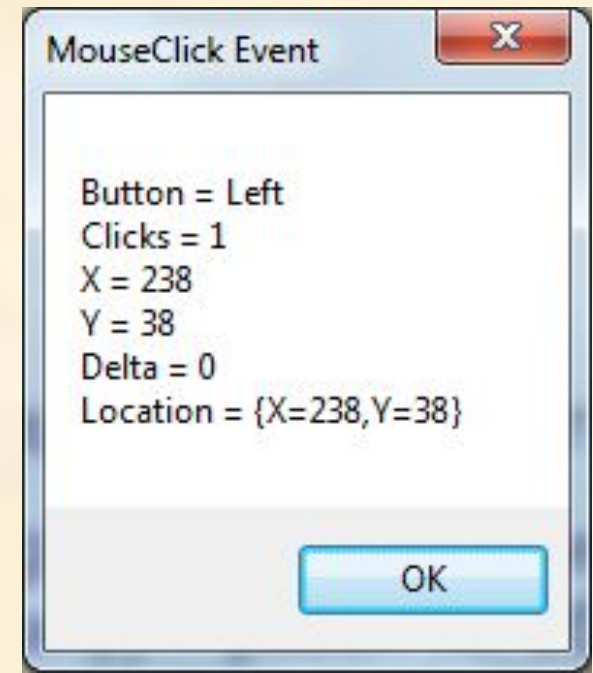
Методы: ToString() – информация о событии (string)
GetType() – тип события (type).

2. **MouseClicked** – щелчок мышкой по компоненту.

```
private void Form1_MouseClick(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButton.Left) Text = "111";
}
```

Свойство параметра e	Тип	Описание
Button	MouseButton	Какая кнопка мыши была нажата. Варианты: None, Left, Right, Middle.
Clicks	int	Число нажатий и отпусканий кнопки мыши.
Delta	int	Число со знаком, указывающее количество делений, на которое повернулось колесико мыши.
Location	point	Расположение указателя мыши на момент создания события.
X	int	Координата x указателя мыши на момент события.
Y	int	Координата y указателя мыши на момент события.

```
private void Form1_MouseClick(object sender, MouseEventArgs e)
{
    StringBuilder MBSt = new StringBuilder();
    MBSt.AppendFormat("{0} = {1}\n", "Button", e.Button);
    MBSt.AppendFormat("{0} = {1}\n", "Clicks", e.Clicks);
    MBSt.AppendFormat("{0} = {1}\n", "X", e.X);
    MBSt.AppendFormat("{0} = {1}\n", "Y", e.Y);
    MBSt.AppendFormat("{0} = {1}\n", "Delta", e.Delta);
    MBSt.AppendFormat("{0} = {1}  ", "Location", e.Location);
    MBSt.AppendLine();
    MessageBox.Show(MBSt.ToString(), "MouseClick Event");
}
```



3. **MouseDown** – происходит при нажатии кнопки мыши.
4. **MouseUp** – происходит при отпускании кнопки мыши.
5. **MouseDoubleClick** – происходит двойном щелчке мышкой.
6. **MouseMove** – происходит при движении мышки над компонентом.

В этих событиях параметр **e** имеет тип **MouseEventArgs**.

7. **MouseEnter** – происходит, когда указатель мыши оказывается на компоненте.
8. **MouseLeave** – происходит, когда указатель мыши покидает компонент.

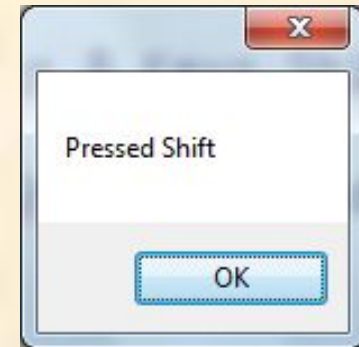
В этих событиях параметр **e** имеет тип **EventArgs**.

Внимание!

Если при работе с мышкой надо отследить состояние управляющих клавиш клавиатуры, можно воспользоваться возможностями объектов **Control** и **Keys**, например, так:

```
private void Form1_MouseDown(object sender, MouseEventArgs e)
{
    if ((Control.ModifierKeys & Keys.Shift) == Keys.Shift)
        MessageBox.Show("Pressed " + Keys.Shift);
}
```

Свойство Keys	Описание
LButton	Левая кнопка мыши
Shift, ShiftKey	Клавиша SHIFT
Control	Клавиша Ctrl
Alt	Клавиша Alt
F1	Клавиша F1
LWin	Левая клавиша с эмблемой Windows



2. События клавиатуры

События нажатия клавиши происходят в следующем порядке:

KeyDown

KeyPress

KeyUp

Событие **KeyPress** не вызывается нажатием управляющих (не символьных) клавиш. Однако нажатие таких клавиш вызывает события **KeyDown** и **KeyUp**.

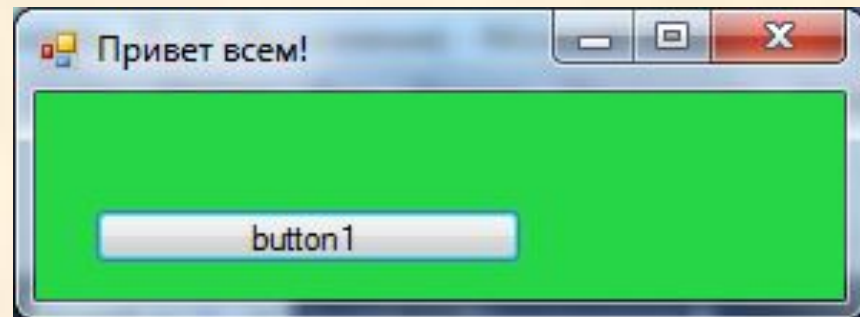
Если на форме имеются другие компоненты, то форма получает данное событие, если её свойство **KeyPreview** установлено в **True**.

Если в обработчике любого клавиатурного события установить **e.Handled** в значение **True**, то обработка происходит без предоставления другим элементам управления возможности получать события клавиатуры.

1. **KeyPress** – ввод символа. Тип параметра – **KeyPressEventArgs**.

Свойство параметра e	Тип	Описание
Handled	bool	Используется для запрещения дальнейшей обработки (e.Handled = true)
KeyChar	char	Символ нажатой клавиши.

```
private void Form1_KeyPress(object sender, KeyPressEventArgs e)
{
    if (e.KeyChar == 8) Text = "";
    else Text = Text + e.KeyChar;
}
```



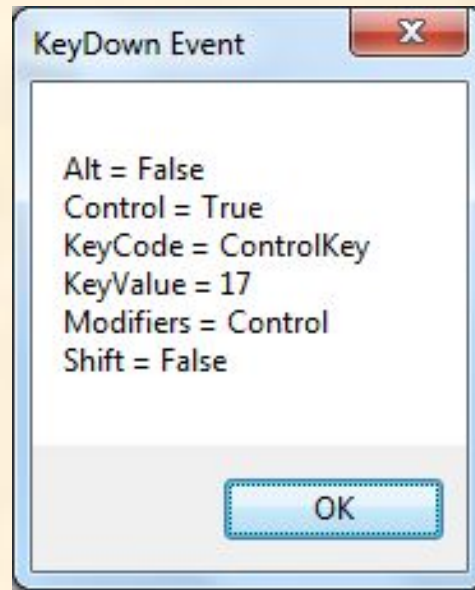
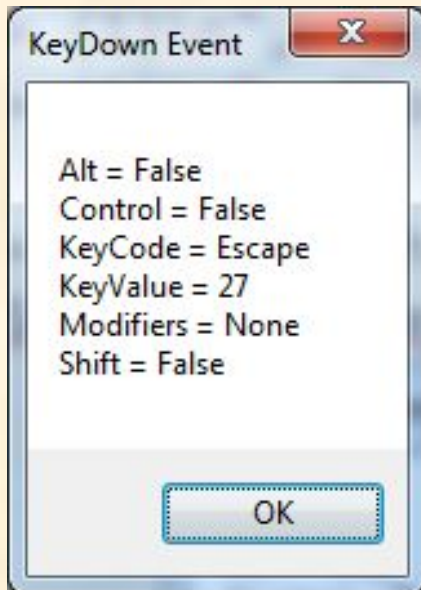
2. KeyDown – нажатие клавиши клавиатуры.

3. KeyUp – отпускание клавиши клавиатуры.

Тип параметра – **KeyEventArgs**.

Свойство параметра e	Тип	Описание
Alt	bool	Была ли нажата клавиша ALT.
Control	bool	Была ли нажата клавиша CTRL.
Shift	bool	Была ли нажата клавиша SHIFT.
Modifiers	Keys	Комбинация управляющих клавиш.
KeyCode	Keys	Код нажатой клавиши.
KeyValue	int	Числовое значение кода нажатой клавиши.

```
private void Form1_KeyDown(object sender, EventArgs e)
{
    StringBuilder MBSt = new StringBuilder();
    MBSt.AppendFormat("{0} = {1}\n", "Alt", e.Alt);
    MBSt.AppendFormat("{0} = {1}\n", "Control", e.Control);
    MBSt.AppendFormat("{0} = {1}\n", "KeyCode", e.KeyCode);
    MBSt.AppendFormat("{0} = {1}\n", "KeyValue", e.KeyValue);
    MBSt.AppendFormat("{0} = {1}\n", "Modifiers",
        e.Modifiers);
    MBSt.AppendFormat("{0} = {1}\n", "Shift", e.Shift);
    MessageBox.Show(MBSt.ToString(), "KeyDown Event");
}
```



Пример.

Нажатие комбинации клавиш Ctrl+X или F10 закрывают форму.

```
private void Form1_KeyDown(object sender,  
                             KeyEventArgs e)  
{  
    if ((e.Control & (e.KeyCode == Keys.X))  
        | (e.KeyCode == Keys.F10)) this.Close();  
}
```

3. Системные события

Возникают, когда форма испытывает воздействие со стороны операционной системы.

1. Load – происходит до первоначального отображения формы. Используется для настройки формы и для создание на ней дополнительных объектов. Тип параметра – **EventArgs**.

2. FormClosing – происходит перед закрытием формы.

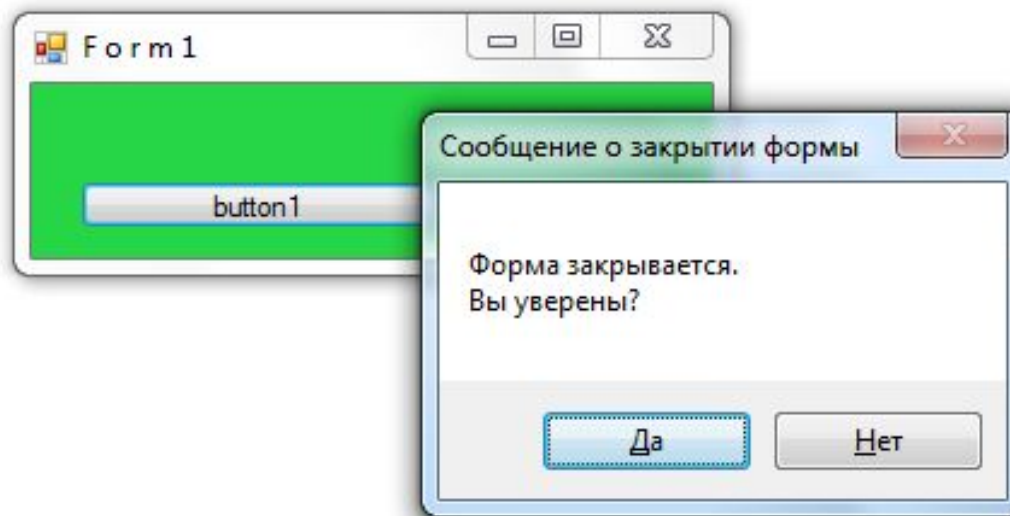
3. FormClosed – происходит в момент закрытия формы.

Используются для освобождения дополнительных ресурсов.
Для обоих событий:

Свойство параметра e	Тип	Описание
Cancel	bool	Если True, то форма не закроется
CloseReason	enum	Получает значение, указывающее, почему форма закрывается (UserClosing, WindowsShutDown, TaskManagerClosing, FormOwnerClosing).

Пример: организация запроса на закрытие формы.

```
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    string message = "Форма закрывается. \nВы уверены?";
    string caption = "Сообщение о закрытии формы";
    MessageBoxButtons buttons = MessageBoxButtons.YesNo;
    DialogResult result;
    result = MessageBox.Show(this, message, caption, buttons);
    if (result == DialogResult.No) e.Cancel = true;
}
```



События формы	Описание
Activated	Происходит при активации формы в коде или с помощью пользователя.
Deactivated	Происходит при потере фокуса неактивной формой.
Shown	Происходит при первом отображении формы.
Move	Происходит при перемещении элемента управления.
Paint	Происходит при перерисовке элемента управления.
Resize	Происходит при изменении размеров элемента управления.

Группа событий связана с изменением свойств:

BackColorChanged, BackgroundImageChanged, FontChanged, LocationChanged, SizeChanged, TextChanged, VisibleChanged и т.п.

5. Методы формы.

Методы формы	Описание
Activate	Активирует форму и переводит на нее фокус.
Deactivate	Происходит при потере фокуса неактивной формой.
BringToFront	Помещает элемент управления «наверх»
SendToBack	Отправляет элемент управления «в глубину»
Close	Закрывает форму.
Focus	Задаёт фокус ввода элементу управления.
Hide	Скрывает элемент управления от пользователя.
Show	Отображает форму.
On...	Вызывает данное событие (например, OnClick)