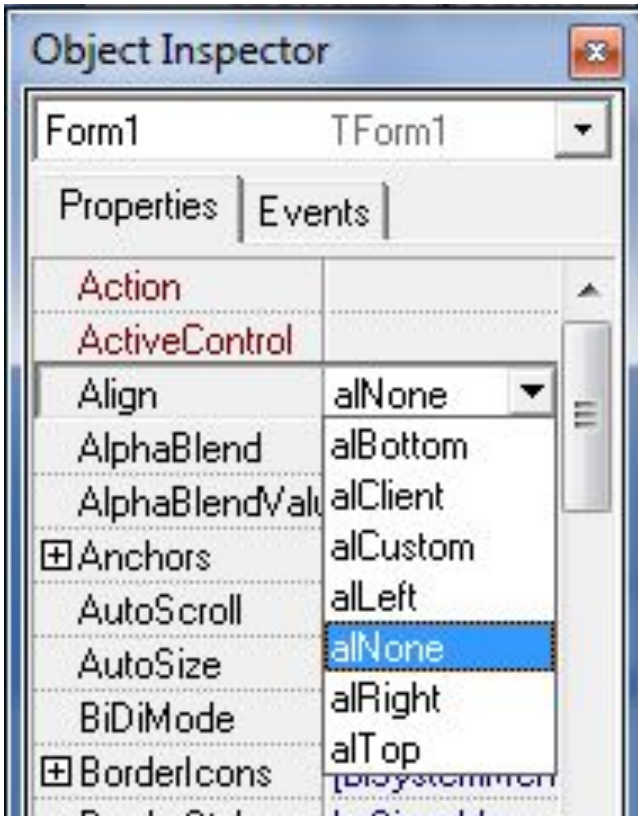
Визуальное программирование. Пустая форма. Компонента Standard

Основные свойства формы

Align - выравнивание компонента

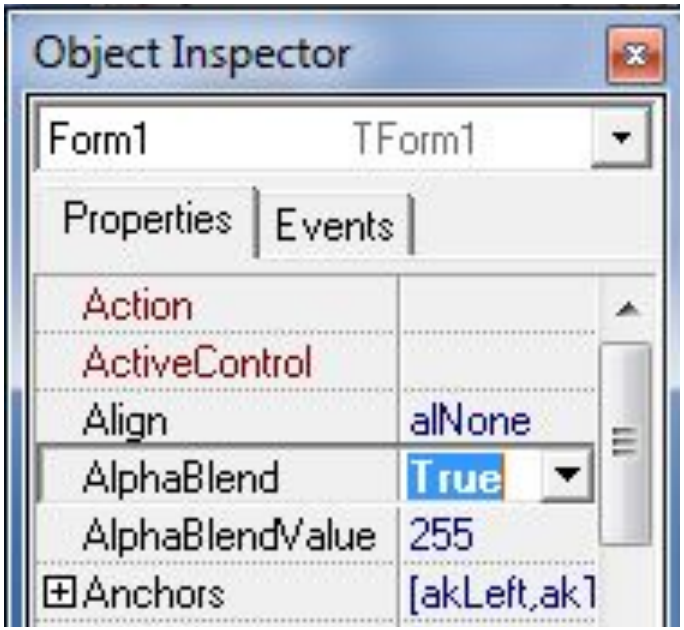


Выравнивание компонента. Любой компонент может быть выровнен по одной из сторон родительского компонента.

Этому свойству можно присвоить следующие значения:

- *alNone* – нет выравнивания
- *alClient* – на весь экран
- *alBottom* – выравнивание по нижнему краю
- *alLeft* - выравнивание по левому краю
- *AlRight* - выравнивание по правому краю
- *AlTop* - выравнивание по верхнему краю

AlphaBlend - прозрачность формы

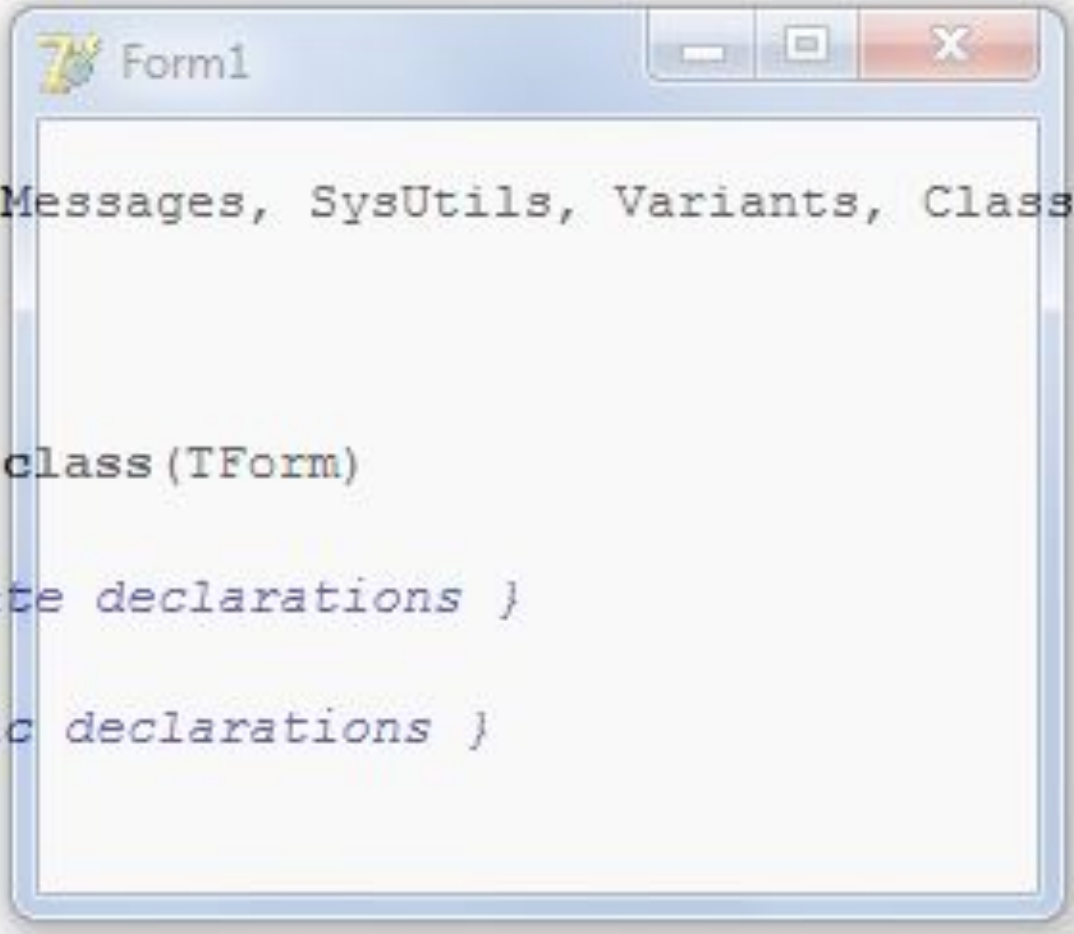


Если это свойство равно true, то окно будет прозрачным.

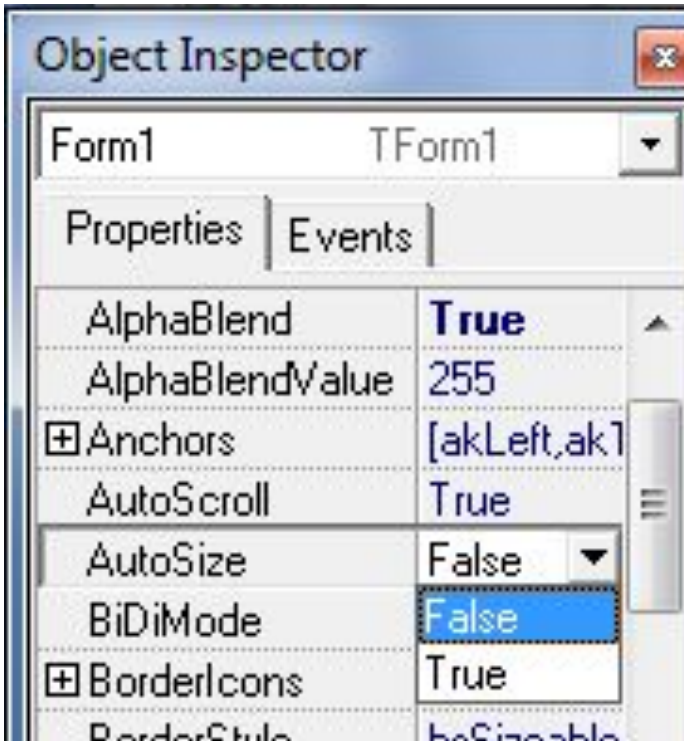
Степень прозрачности задаётся через свойство AlphaBlendValue (от 0 до 255).

!!! Прозрачность работает не на всех системах.

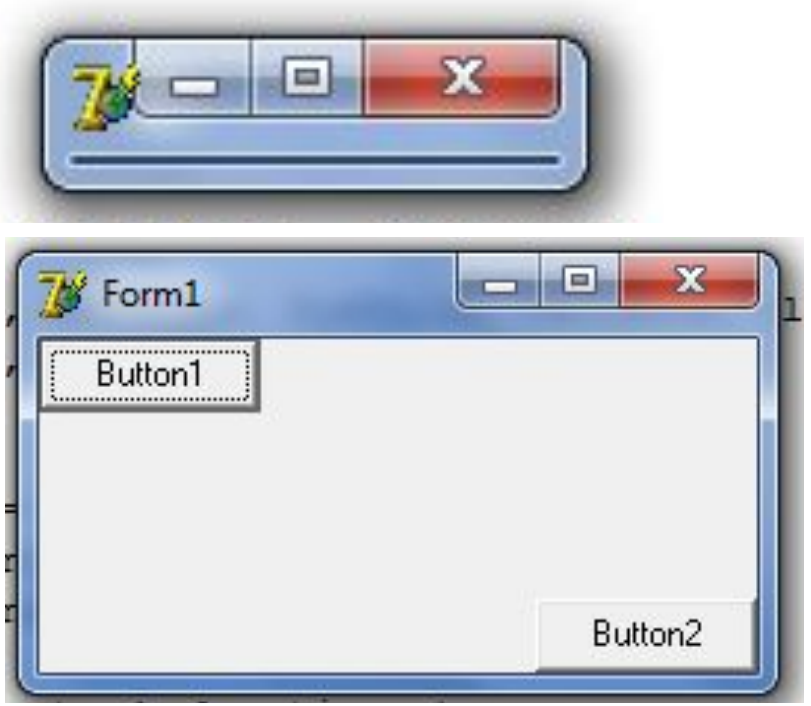
```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graph
  Dialogs;

type
  TForm1 = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
```



AutoSize – размеры формы



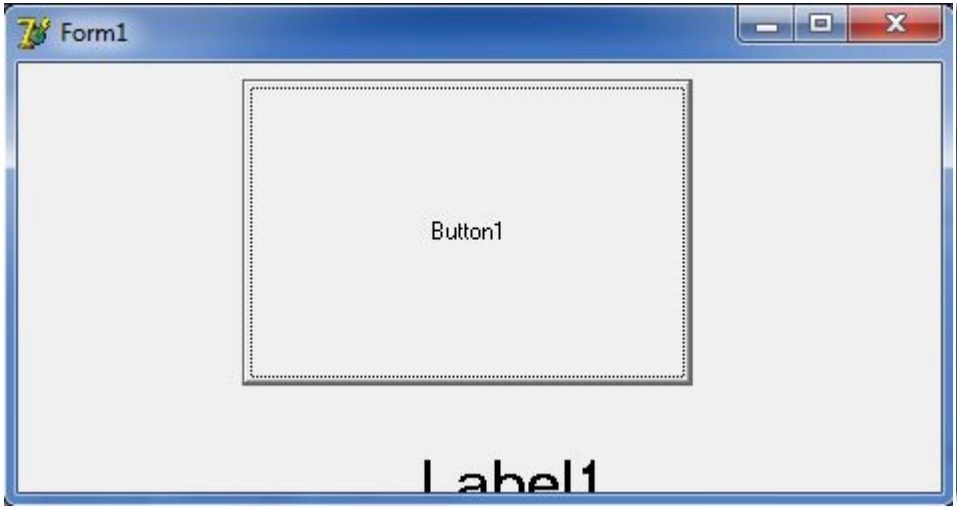
Если это свойство равно true, то окно формы имеет минимальный размер с учетом всех компонент на ней.

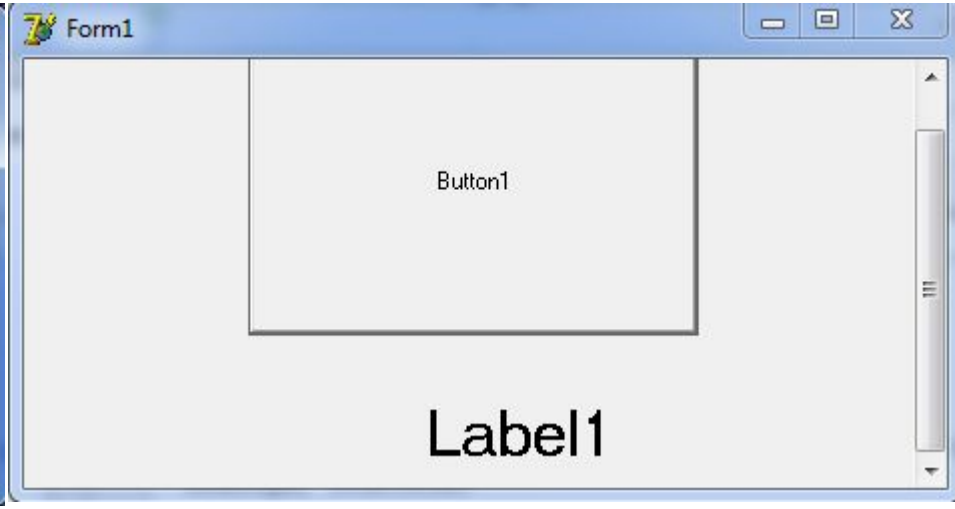


AutoScroll

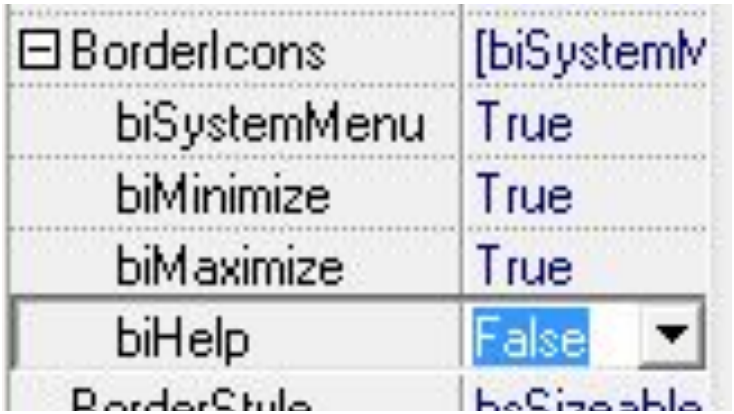
Тип свойства – логический.

Будет ли форма автоматически производить скроллинг, или нет.





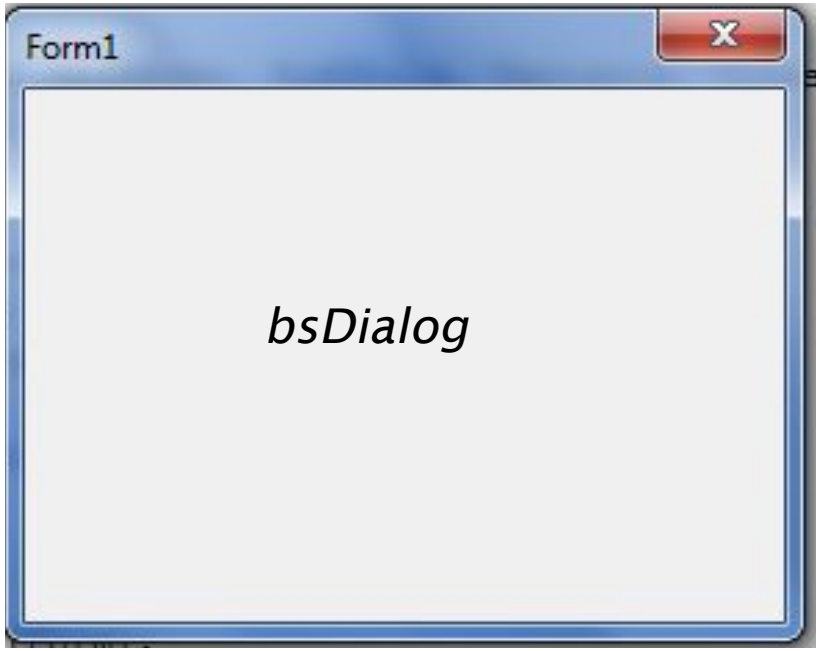
BorderIcons - свойство определяющее, какие кнопки должны присутствовать у окна



- *biSystemMenu* – показать меню (иконка слева в строке заголовка окна).
- *biMinimize* – кнопка минимизации окна.
- *biMaximize* – кнопка максимизации окна.
- *biHelp* – кнопка помощи.

BorderStyle – свойство формы, отвечающее за вид оборки окна

- *bsSizeable* – установлено по умолчанию. Стандартное окно, с нормальной оборкой, которое может изменять свои размеры.
- *bsSizeToolWin* – аналог bsSizeable
- *bsDialog* – окно выглядит в виде диалога без кнопок минимизации и максимизации.
- *bsNone* – окно вообще без оборки.
- *bsSingle* –окно с фиксированным размером и изменять его мышкой нельзя.
- *bsToolWindow* – аналог bsSingle



bsDialog

bsNone

bsSingle

bsSizeable

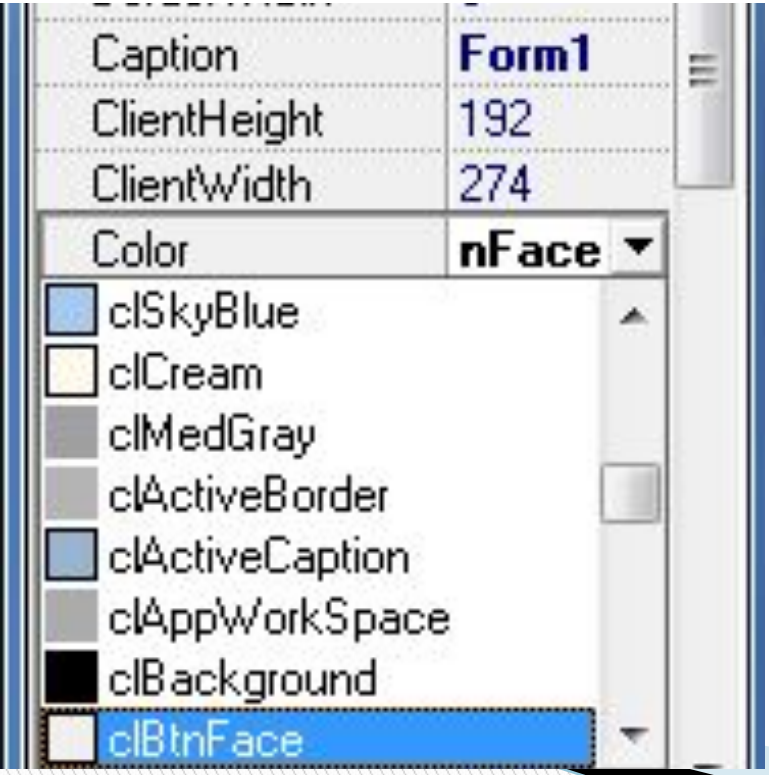
bsSizeToolWin

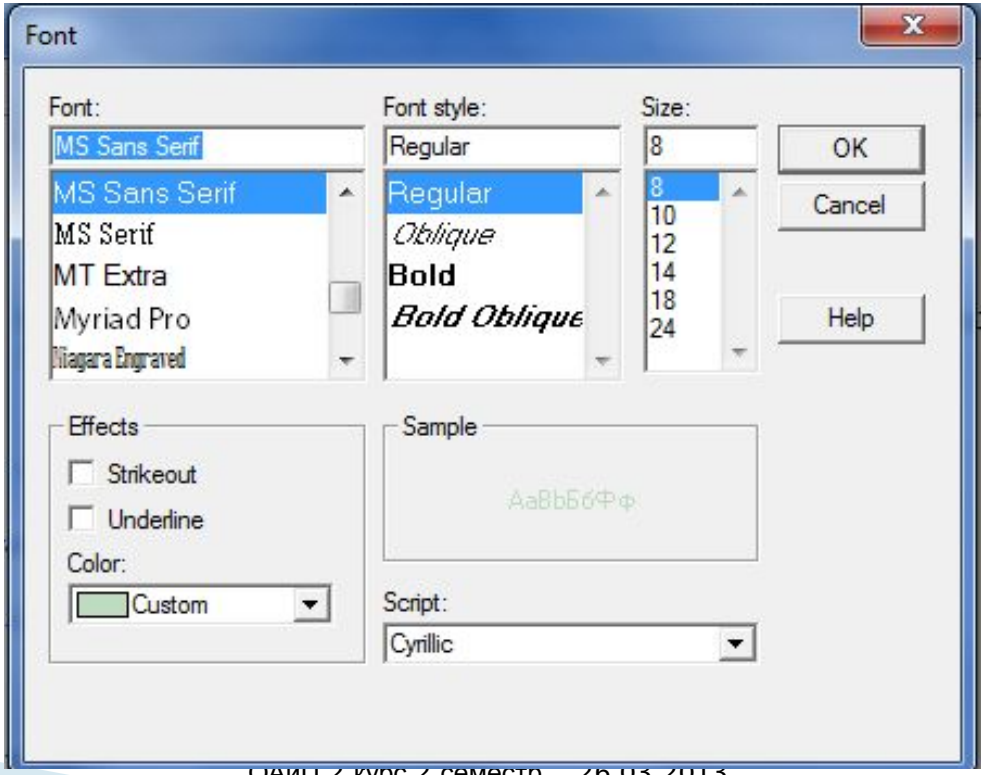
bsToolWindow

Caption – это строковое свойство, которое отвечает за заголовок окна.

Color – цвет клиентской области окна.

Font – шрифт используемый при выводе текста на форме.





Height* и *Width – высота и ширина окна. Тип свойства – целое число.

Left* и *Top – левая и верхняя позиции окна. Тип свойства – целое число.

Constraints – **в этом свойстве содержатся максимальные значения размеров окна.**

MaxHeight – **максимальная высота окна.**
MaxWidth – **максимальная ширина окна.**
MinHeight – **минимальная высота окна.**
MinWidth – **минимальная ширина окна.**

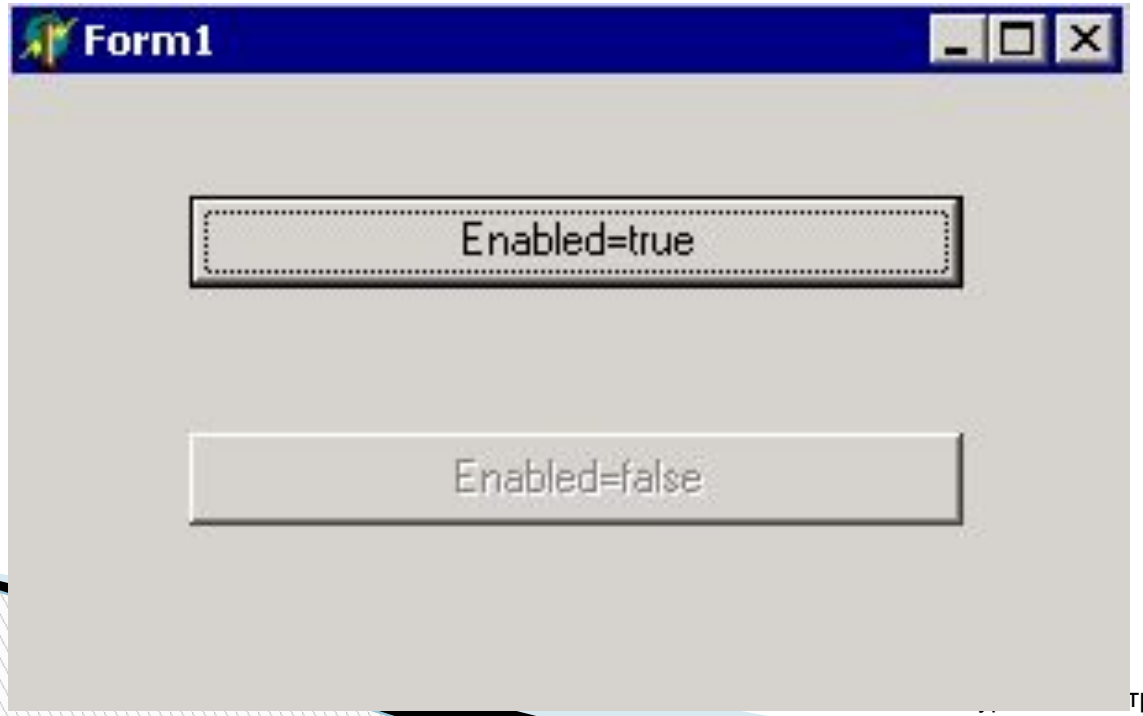
установив эти значения, окно нельзя будет растянуть больше максимального размера и уменьшить меньше минимального.

Cursor – это свойство отвечает за курсор, который будет отображаться при наведении мышкой на форму/компонент.

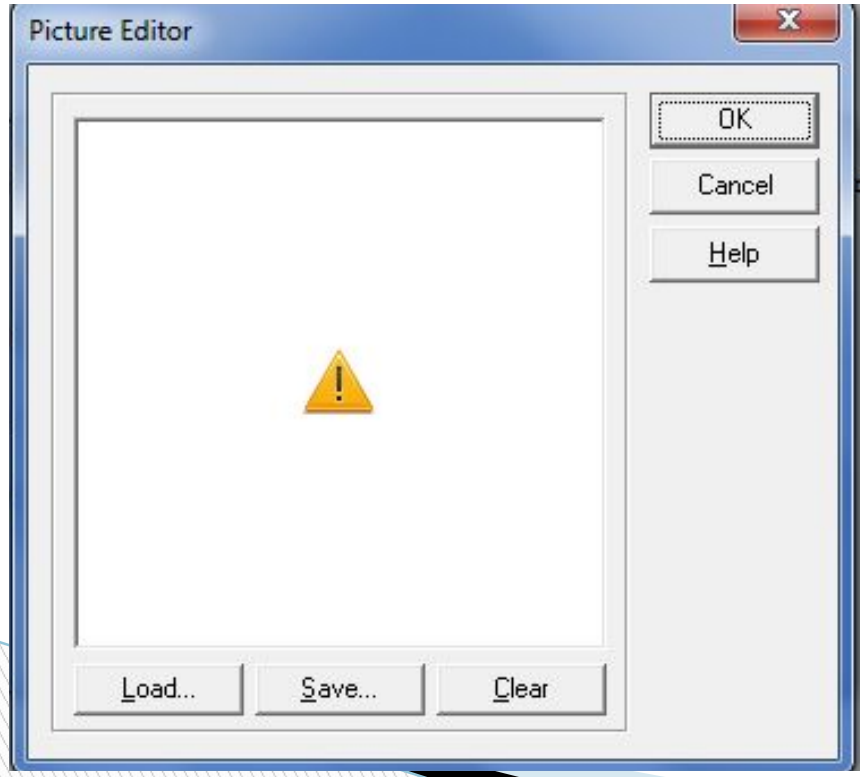
Имя курсора	Вид	Имя курсора	Вид
crNone	Нет	CrArrow	
crCross		crIBeam	
crSizeNESW		crSizeNS	
crSizeNWSE		crSizeWE	
crUpArrow		crHourGlass	
crDrag		crNoDrop	
crHSplit		crVSplit	
crMultiDrag		crSQLWait	
crNo		crAppStart	
crHelp		crHandPoint	
crSize		crSizeAll	

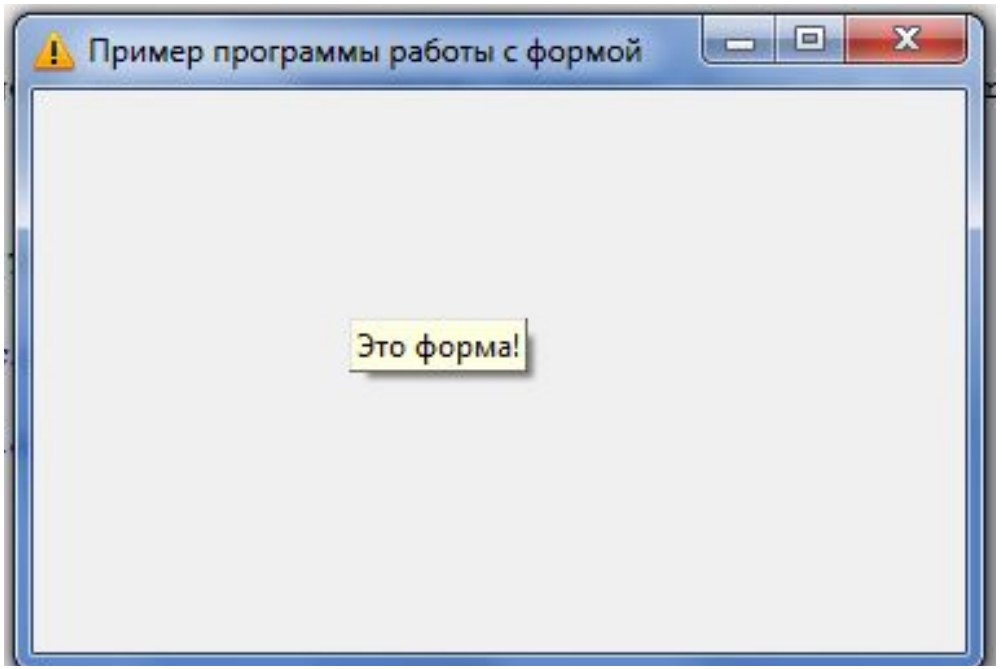
Name – имя формы/компонента.
Этим именем будет называться объект, отвечающий за эту форму/компонент (только в начале добавится буква T).

Enabled – Тип свойства – логический.
Доступность компонента. Если это свойство равно true, то пользователь может работать с этим компонентом. Иначе компонент недоступен и окрашен серым цветом.

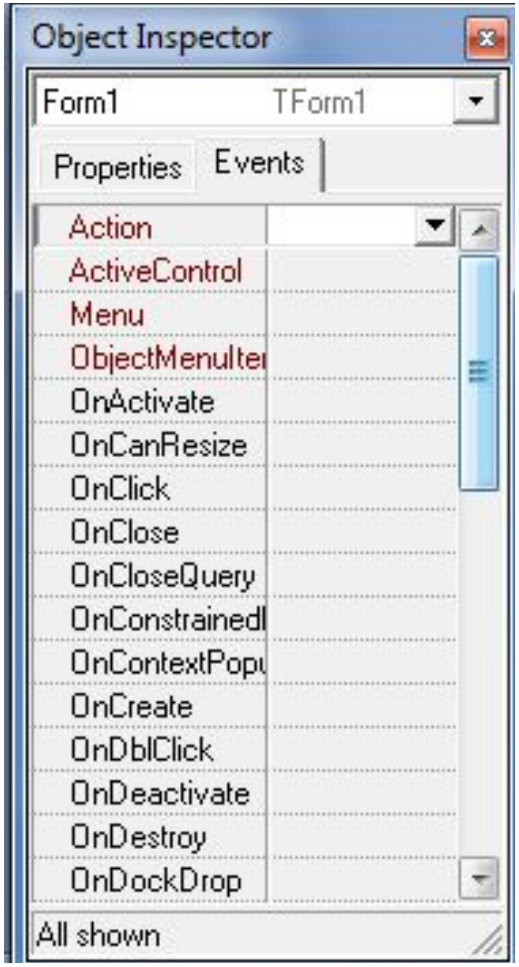


Icon – иконка отображающаяся в заголовке окна
Hint – текст подсказки, который будет появляться в строке состояния при наведении мышкой на форму/компонент.
ShowHint – Тип свойства – логический. Оно указывает - нужно ли показывать подсказки.





Основные события формы



Обработчик события – это процедура, которая вызывается по наступлению какого-то события.

Событие	Описание
OnActivate	Когда приложение стало активным
OnCanResize	Генерируется перед тем, как изменить размер окна.
OnClick	Генерируется, когда пользователь щёлкнул по форме.
OnClose	Генерируется, когда окно закрывается.
OnCloseQuery	Генерируется до закрытия окна (запрос подтверждения на закрытие).
OnCreate	Генерируется, когда окно создаётся.
OnDblClick	Генерируется, когда пользователь дважды щёлкнул по окну.
OnDeactivate	Генерируется, когда окно деактивируется.
OnDestroy	Когда окно уничтожается.
OnHide	Генерируется, когда окно исчезает.
OnKeyDown	Генерируется, когда нажата клавиша на клавиатуре.
OnKeyPress	Генерируется, когда нажата и отпущена клавиша на клавиатуре.
OnKeyUp	Генерируется, когда отпущена клавиша на клавиатуре.

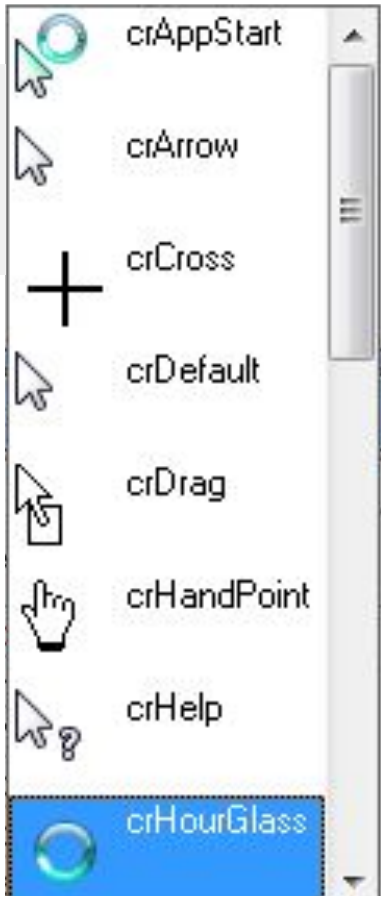
OnMouseDown	Генерируется, когда нажата кнопка мыши.
OnMouseMove	Генерируется, когда двигается мышка.
OnMouseUp	Генерируется, когда отпущена кнопка мыши.
OnMouseWheel	Генерируется колёсиком мыши.
OnMouseWheelDown	Генерируется, когда колёсико мыши прокручено вниз.
OnMouseWheelUp	Генерируется, когда колёсико мыши прокручено вверх.
OnPaint	Генерируется, когда надо перерисовать окно.
OnResize	Генерируется, когда надо изменить размеры окна.
OnShortCut	Когда нажата горячая клавиша.
OnShow	Когда показано окно

Компоненты страницы Standart, их использование и свойства

Кнопка (TButton)

Caption – название кнопки (тип - строка)

Cursor – вид курсора при наведении

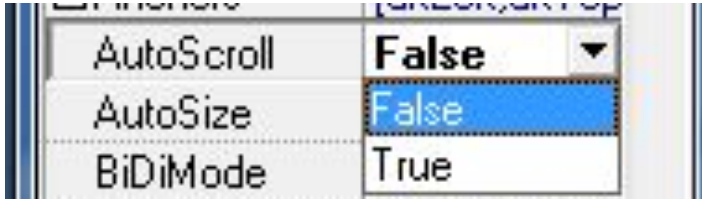


Enabled – доступ к кнопке (тип - логический)

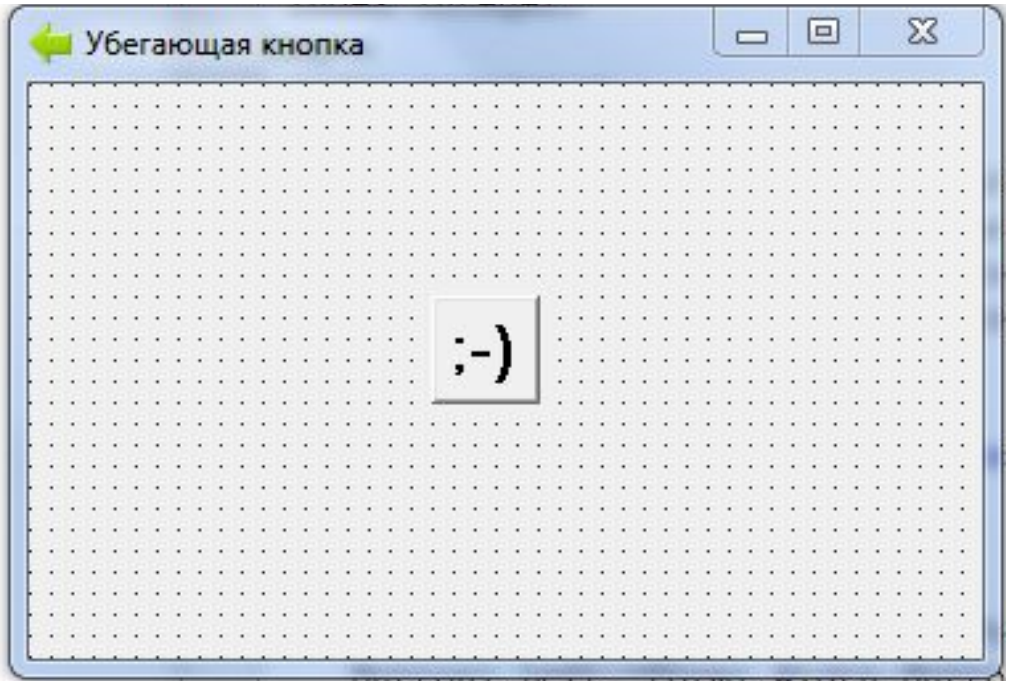
Cancel – нажатие на Esc будет эквивалентно нажатию на кнопку (тип - логический)

Программа «Убегающая кнопка»

Для начала изменим свойство формы *AutoScroll* на ***False***, чтобы на форме не появлялись автоматически полосы прокрутки.



Далее, поместим на форму кнопку и установим настройки



Теперь необходимо создать обработчик события OnMouseMove

```
procedure TForm1.Button1MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
var
 index:integer;
begin
 index:=random(4);
 case index of
  0: Button1.Left:=Button1.Left+Button1.Width; {кнопка сдвинется вправо}
  1: Button1.Left:=Button1.Left-Button1.Width; {кнопка сдвинется влево}
  2: Button1.Top:=Button1.Top+Button1.Height;  {кнопка сдвинется вниз}
  3: Button1.Top:=Button1.Top-Button1.Height;  {кнопка сдвинется вверх}
 end;
```

Параметры обработчика события OnMouseMove

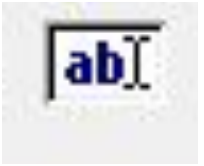
- Параметр **Shift** типа TShiftState определяет, какие вспомогательные клавиши на клавиатуре нажаты в момент передвижения мыши.

- Параметры **X** и **Y** определяют координаты курсора в клиентской области компонента. Благодаря этому можно обеспечить различную реакцию в зависимости оттого, где расположен курсор.

Допишем необходимый код в Button1MouseMove

```
{проверка наличия кнопки в пределах окна}
if Button1.Left<0 then
  Button1.Left:=0;

if (Button1.Left+Button1.Width)>Form1.Width then
   Button1.Left:=Form1.Width-Button1.Width;

if Button1.Top<0 then
    Button1.Top:=0;

if (Button1.Top+Button1.Height)>Form1.Height then
   Button1.Top:=Form1.Height-Button1.Height;

end;
```

Строки ввода (TEdit)



Основное свойство – ***Text (тип String)***

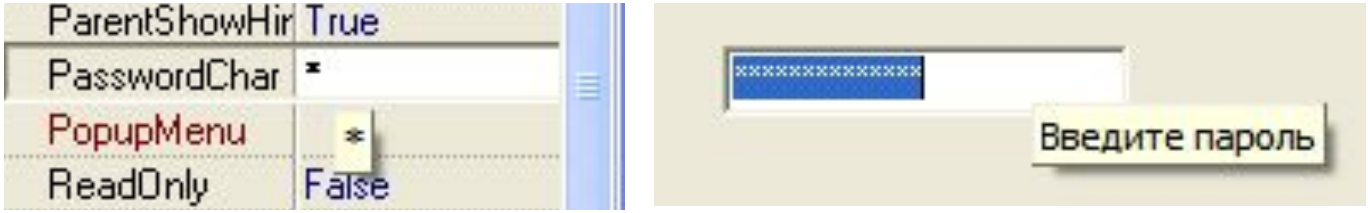
Свойство ***ReadOnly*** определяет поле только для чтения

Функции перевода из строки в число: ***StrToInt, StrToFloat,*** ***наоборот IntToStr, FloatToStr.***

Свойство ***MaxLength*** определяет максимальную длину вводимого текста. Если MaxLength = 0, то длина текста не ограничена

Свойство ***Modified***, доступное только во время выполнения, показывает, проводилось ли редактирование текста в окне

Свойство ***PasswordChar*** позволяет превращать окно редактирования в окно ввода пароля.



Свойство ***BorderStyle*** устанавливает стиль границы поля

Свойство ***CharCase*** устанавливает регистр вводимых символов

Пример, как разрешить только ввод цифр, знака "-" и разделителя.

```
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
Begin
case Key of
#8,'0'..'9' :  ;
'.',',':  begin
        if Key <> DecimalSeparator then  Key := DecimalSeparator;
        { заменим разделитель на допустимый }
        if Pos(DecimalSeparator, Edit1.text) <> 0  then Key := Chr(0);
        {запрет ввода второго разделителя }
        end;
        '-':    { минус можно ввести только первым символом}
             if Length(Edit1.text) <> 0  then Key := Chr(0);
       else  { остальные символы запрещены  }   key := Chr(0);
 end;
end;
```

Многострочное поле ввода (TMemo)

Основные свойства:

- ***Font*** задает формат (шрифт, его атрибуты, выравнивание) одинаковый для всего текста.

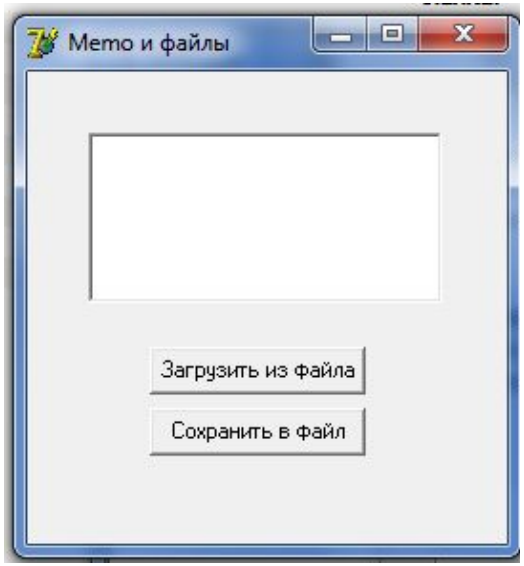
За содержимое текста отвечает свойство ***Lines***.

Это свойство - объект типа TStrings, и имеет свои свойства и методы.

- ***Lines*** содержит текст окна в виде списка строк и имеет тип TStrings.
- ***Count*** - число строк в тексте.
- ***Clear*** - очистка текста в окне.
- ***Add*** - занесения новой строки в конец текста.
- ***Strings[Index: Integer]*** -доступ к отдельной строке текста

Для загрузки текста из файла применяется метод ***LoadFromFile.***

Сохранение текста в файле осуществляется методом ***SaveToFile.***

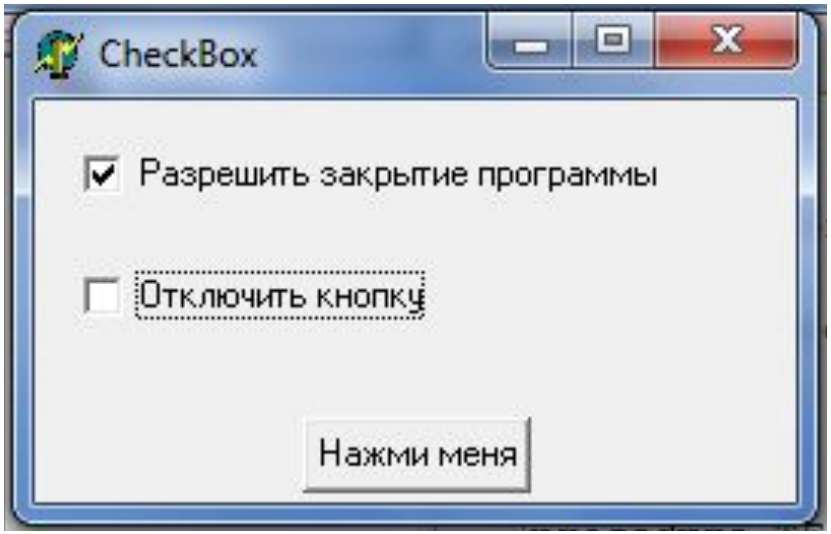



```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Memo1.Lines.LoadFromFile('Input.txt');
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  Memo1.Lines.SaveToFile('Output.txt');
end;
```

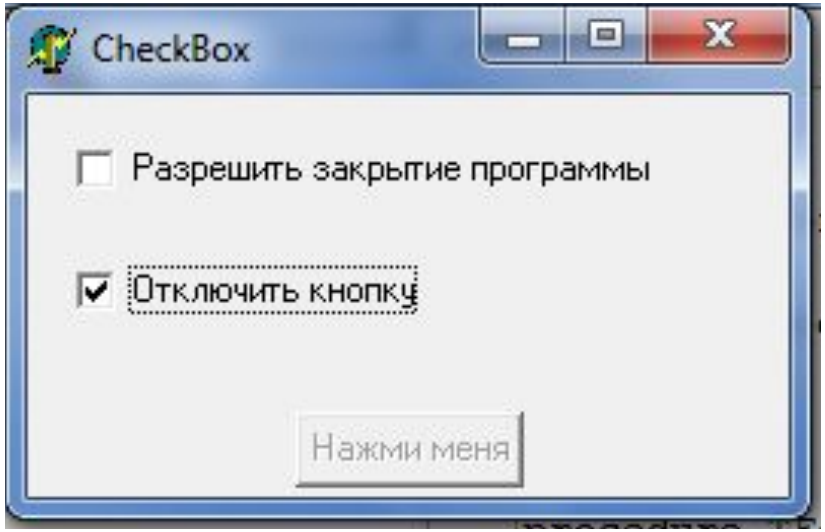
Индикатор (TCheckBox)

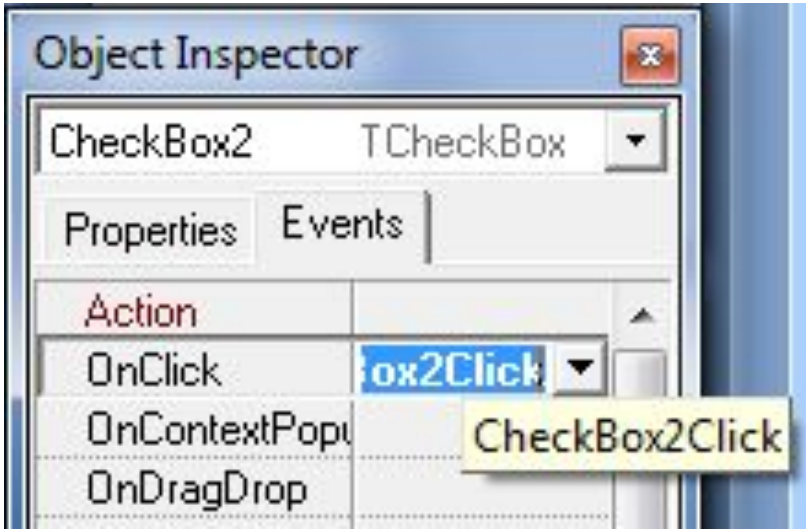
Основное свойство – ***Checked***.

Если компонент выделен, то оно равно ***True***, иначе ***False***.




```
procedure TForm1.Button1Click(Sender: TObject);
begin
 if CheckBox1.Checked then
  Close;
end;
```



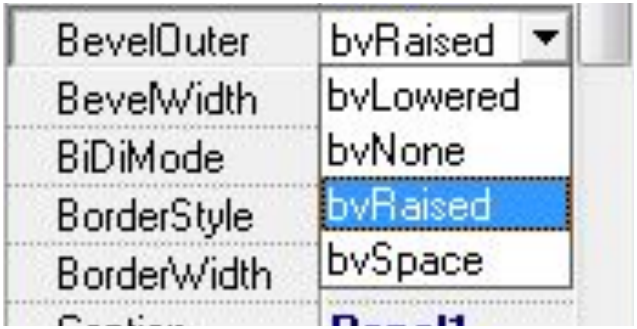


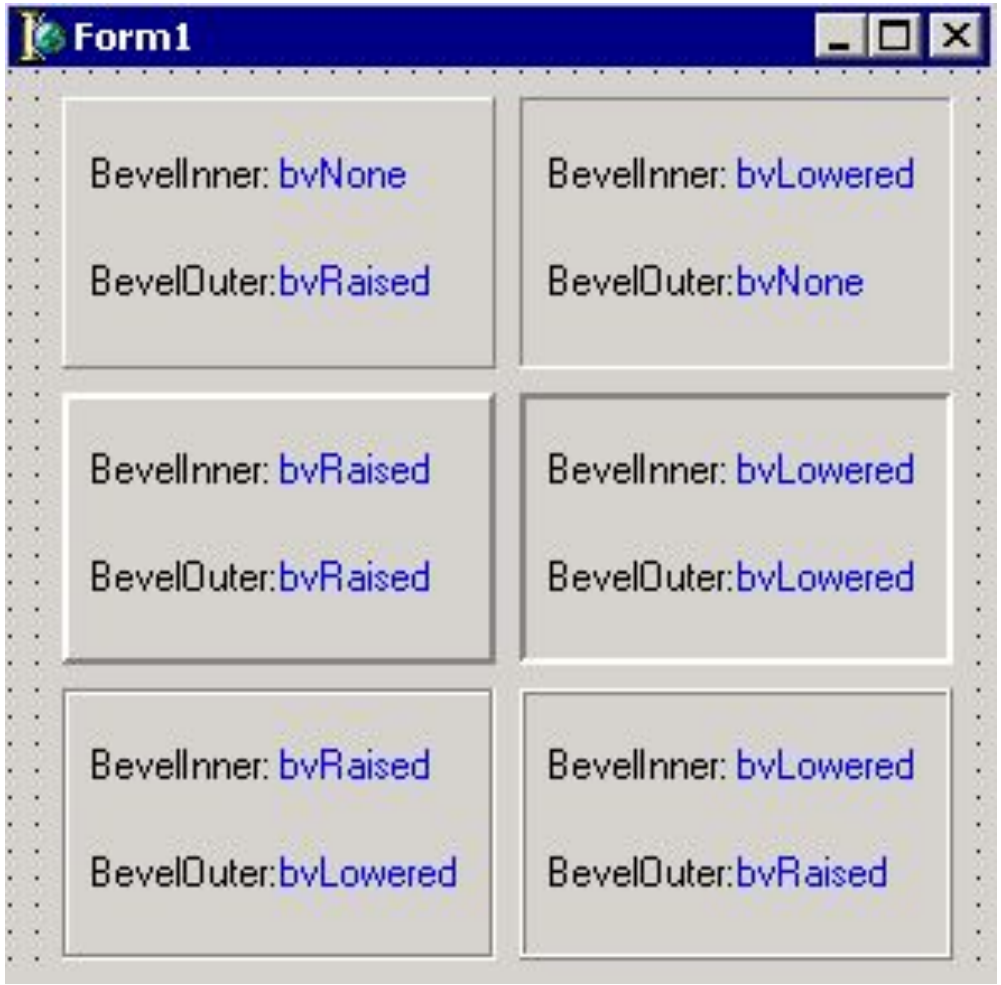

```
procedure TForm1.CheckBox2Click(Sender: TObject);
begin
 Button1.Enabled:=not CheckBox2.Checked;
 {либо
 if CheckBox2.Checked
 then Button1.Enabled:=false;     }
end;
```

Панели (TPanel)

Основное назначение Panel: компоновка компонентов в окне формы. Однако панель можно использовать и для вывода текстов.

Основные свойства:
BevelInner и
BevelOuter,
отвечающие за внешний





Кнопки выбора (TRadioButton)

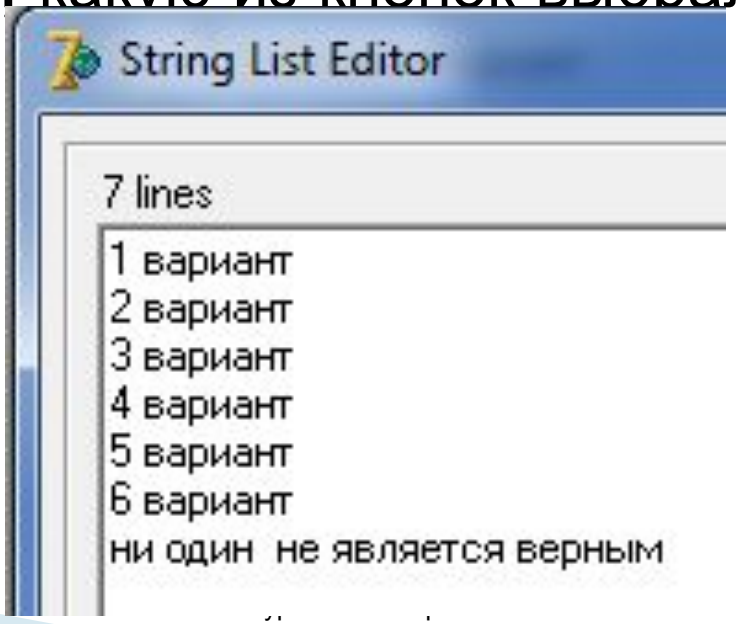
Основное свойство – ***Checked***.
Если компонент выделен, то оно равно ***True***, иначе ***False***.
В отличие от CheckBox выбрать можно только одну из кнопок.

Двойной выбор на форме можно осуществить, если компоненты *RadioButton* убрать на отдельные панели *Panel*

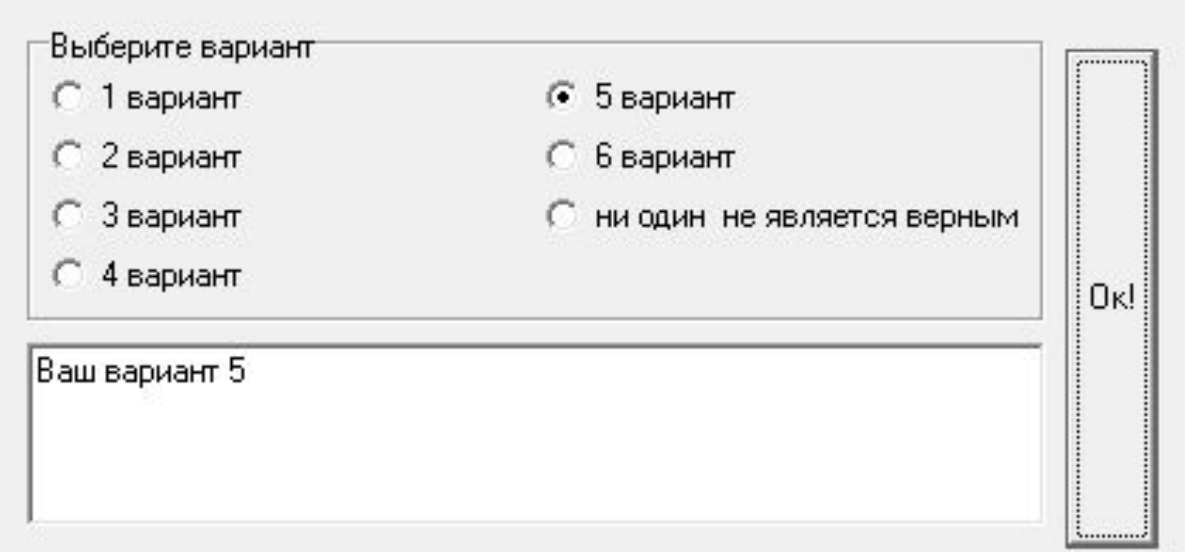
Группы радиокнопок (TRadioGroup)

Основные свойства:

- ***Caption*** - название
- ***Items*** - надписи кнопок и их количество (Объект типа TString)
- ***Columns*** – кнопки можно разместить в несколько столбцов
- ***ItemIndex*** - определяет, какую из кнопок выбрал пользователь



В Memo1 по нажатию Ok! выводится выбранный вариант

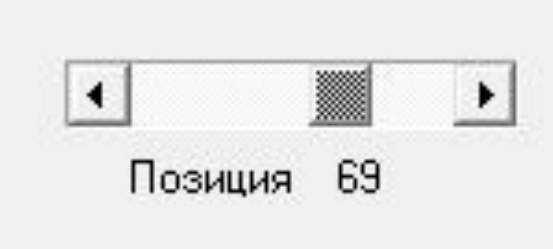



```
procedure TForm1.Button1Click(Sender: TObject);
begin
 if   RadioGroup1.ItemIndex<>6
 then  begin
   Memo1.Clear;
   Memo1.Lines.Add('Ваш вариант '+IntToStr(RadioGroup1.ItemIndex+1));   end
 else   begin
   Memo1.Clear;
   Memo1.Lines.Add('Ни один не является верным');   end;
end;
```

Полосы прокрутки (TScrollBar)

Основные свойства:

- ***Position*** - позиция, в которую пользователь переместил ползунок
- ***Min*** и ***Max*** – пределы изменения свойства Position
- ***Kind*** - горизонтальное или вертикальное расположение полосы



Основное событие:

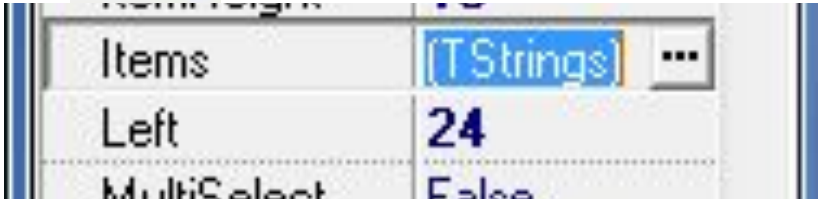
OnChange – перемещение ползунка или нажатие на клавиши со стрелками

```
procedure TForm1.ScrollBar1Change(Sender: TObject);
begin
   Label1.Caption:=IntToStr(ScrollBar1.Position);
end;
```

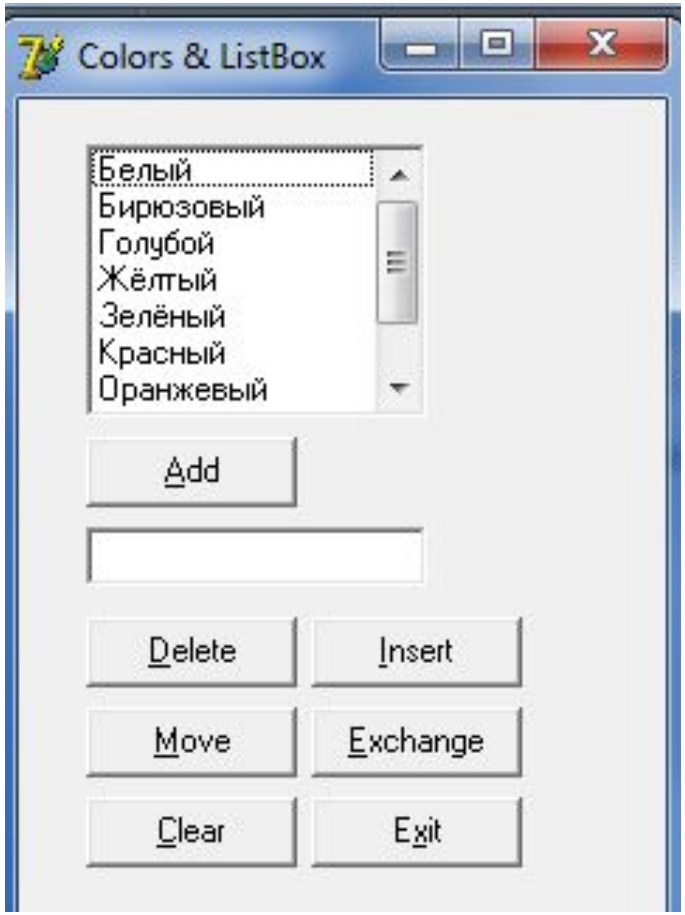
Списки выбора (TListBox)

Основные свойства:

- ***Items*** – элементы списка
- ***MultiSelect*** - разрешает пользователю множественный выбор в списке
- ***ItemIndex*** - индекс выбранной строки
- ***Columns*** - число столбцов, в которых будет отображаться список
- ***Sorted*** - позволяет упорядочить список по алфавиту
- ***AutoComplete*** - позволяет быстро находить строку списка, нажимая только первый символ.



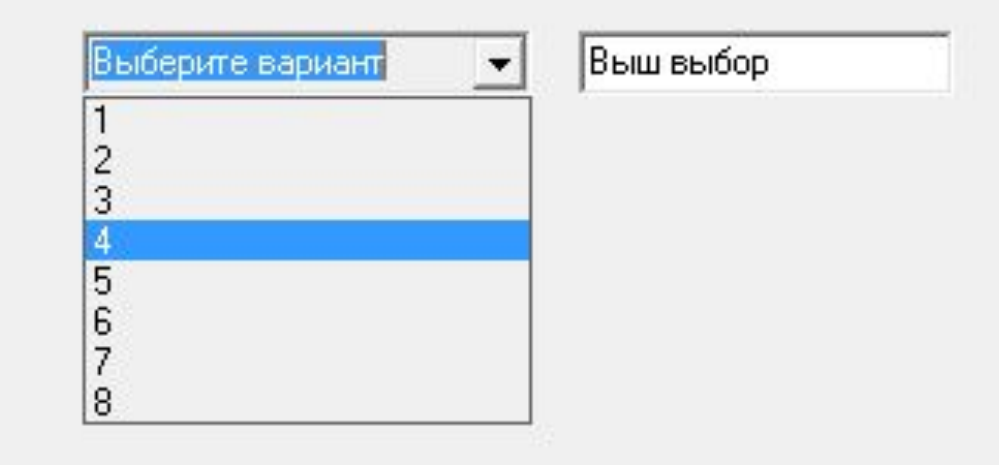
- ***Items.Add*** – добавление строки
- ***Items.Delete*** – удаление строки
- ***Items.Exchange*** – обмен строк
- ***Items.Clear*** – очистка списка
- ***Items.Move*** – перемещение строки
- ***Items.Insert*** – вставка



```
{Добавление строки в конец списка}
procedure TForm1.Button1Click(Sender: TObject);
begin
   ListBox1.Items.Add(Edit1.Text);
end;
{Удаление выделенной строки}
procedure TForm1.Button2Click(Sender: TObject);
begin
  ListBox1.Items.Delete(ListBox1.ItemIndex);
end;
{очистка списка}
procedure TForm1.Button3Click(Sender: TObject);
begin
 ListBox1.Items.Clear;
end;
```

Выпадающие списки (TComboBox)

По своей работе, свойствам и методам похожи на списки выбора.




```
procedure TForm1.ComboBox1Change(Sender: TObject);
begin
  Edit1.Text:=ComboBox1.Items.Strings[ComboBox1.ItemIndex];
end;
```