ПОПО

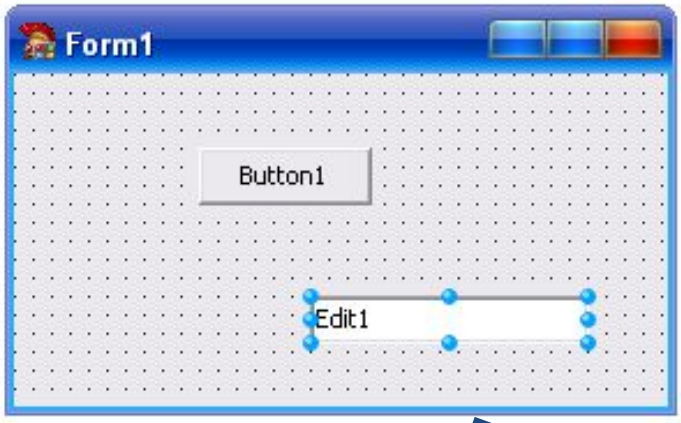
Компонентная модель. Работа с «готовыми» компонентами в режиме Run – time

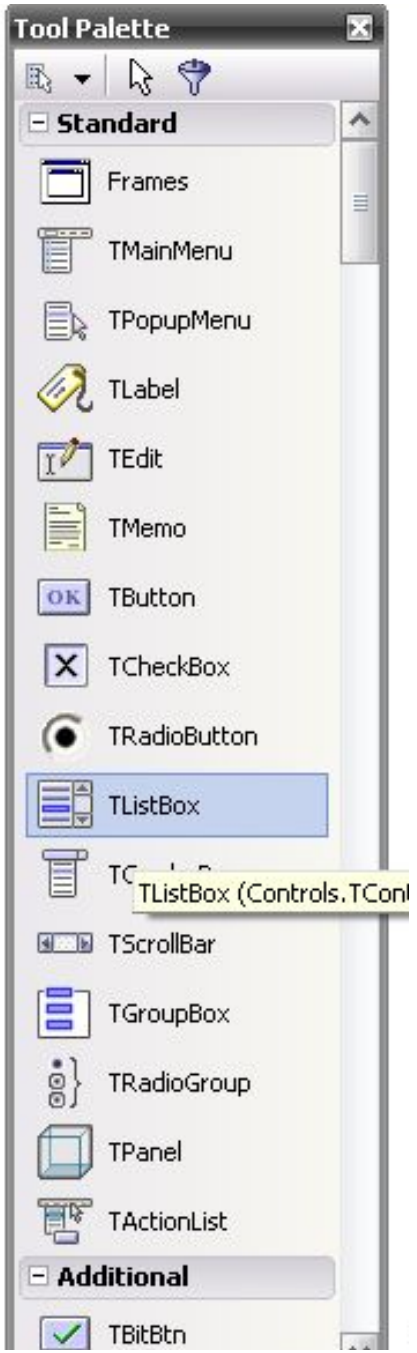
- Создание компонентов
- Приемы создания
- Алгоритм работы

Компоненты – это тоже объекты, только более совершенные. Они обладают такими методами и свойствами, благодаря которым становится возможной работа в визуальном режиме. Существует несколько **библиотек** компонентов.

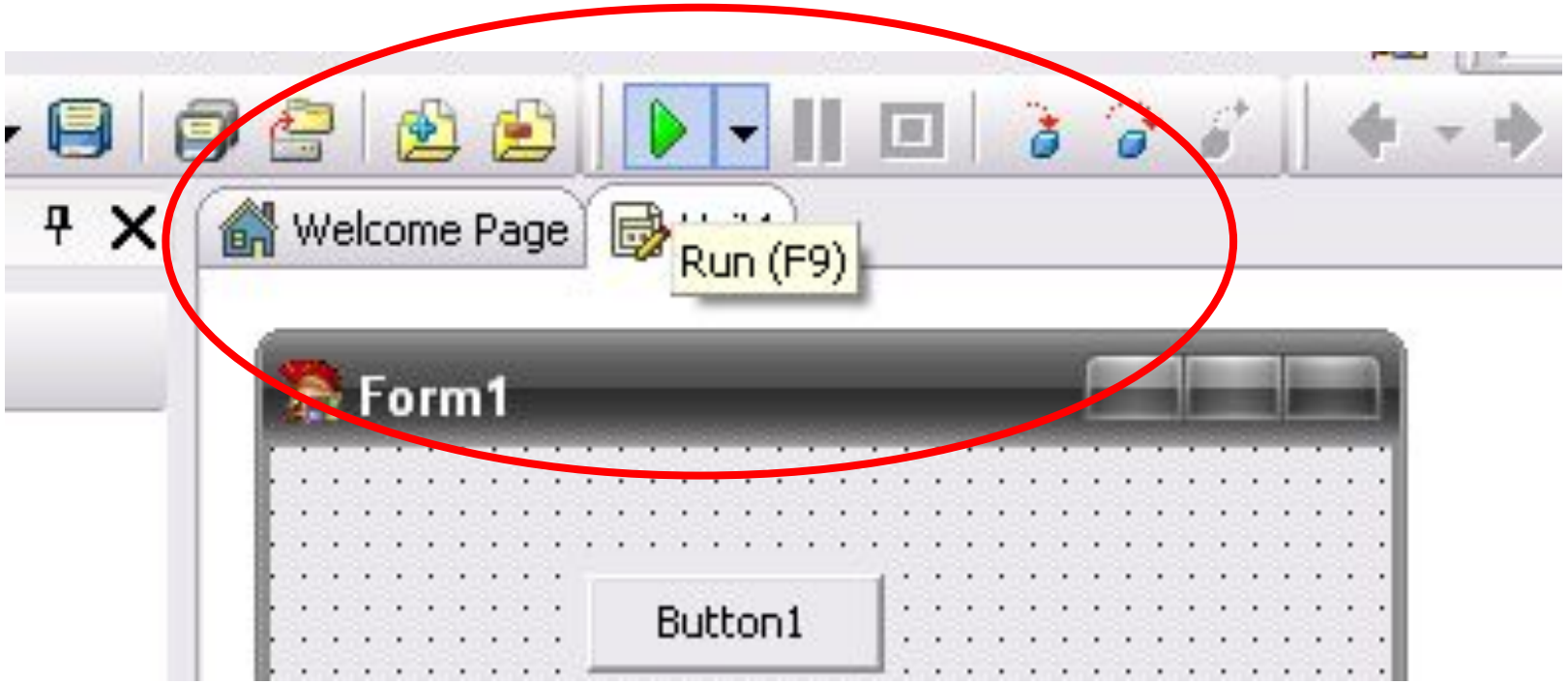
Два режима создания компонентов :

- **Design-time** – в режиме проектирования формы;
- **Run-time** – в режиме выполнения приложения.





Режим Design-time



Режим **Run-time**

Когда удобнее использовать режим Run-time:

- когда нужно создать большое количество однотипных компонентов;
- когда нужно обработать группы компонентов, меняя их свойства «одним махом»;
- когда хотелось бы протестировать новый компонент.

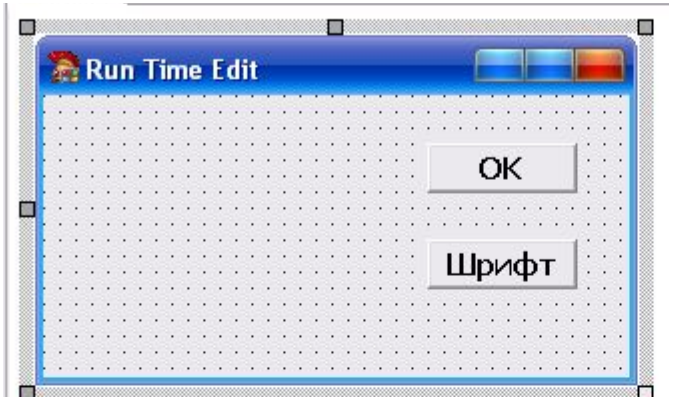
Модули в разделе **Uses**

StdCtrls, Buttons, DateUtils, ExtCtrls;

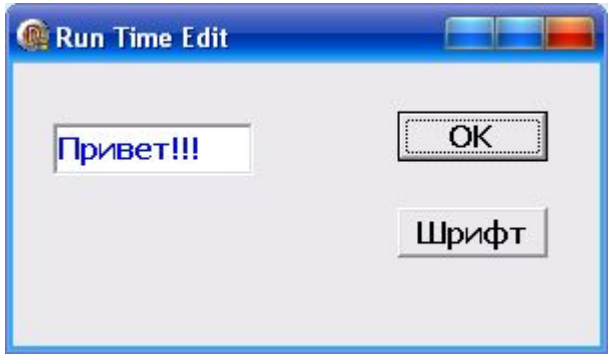
Контролеры за объектами

1. Пример: создадим текстовое поле с произвольным текстом

Режим проектирования формы



Режим запуска приложения



```
procedure TForm1.Button1Click(Sender: TObject);
var    ed:  TEdit;
begin     //Создали визуальный компонент
 ed:=TEdit.Create(Form1);  //Определяем «хозяина»
                                                  //- Form1
 ed.Parent:=self;     //Определяем визуального
                                  // родителя
 ed.Top:=30;     //свойства компонента
 ed.Left:=20;
 ed.Width:=100;
 ed.Font.Color:=Rgb(Random(255),Random(255),Random(255));
 ed.Text:='Привет!!!';
 Randomize;
 end;
```

У создаваемого компонента обязательно должен быть «хозяин» - объект, который отвечает за его корректное удаление и освобождение памяти. А у создаваемого визуального компонента должен быть еще и визуальный родитель – объект, который как раз и отвечает за его отображение.

Все компоненты делятся на визуальные и невизуальные:

Визуальные – видимые компоненты на форме (метка, кнопка, …)

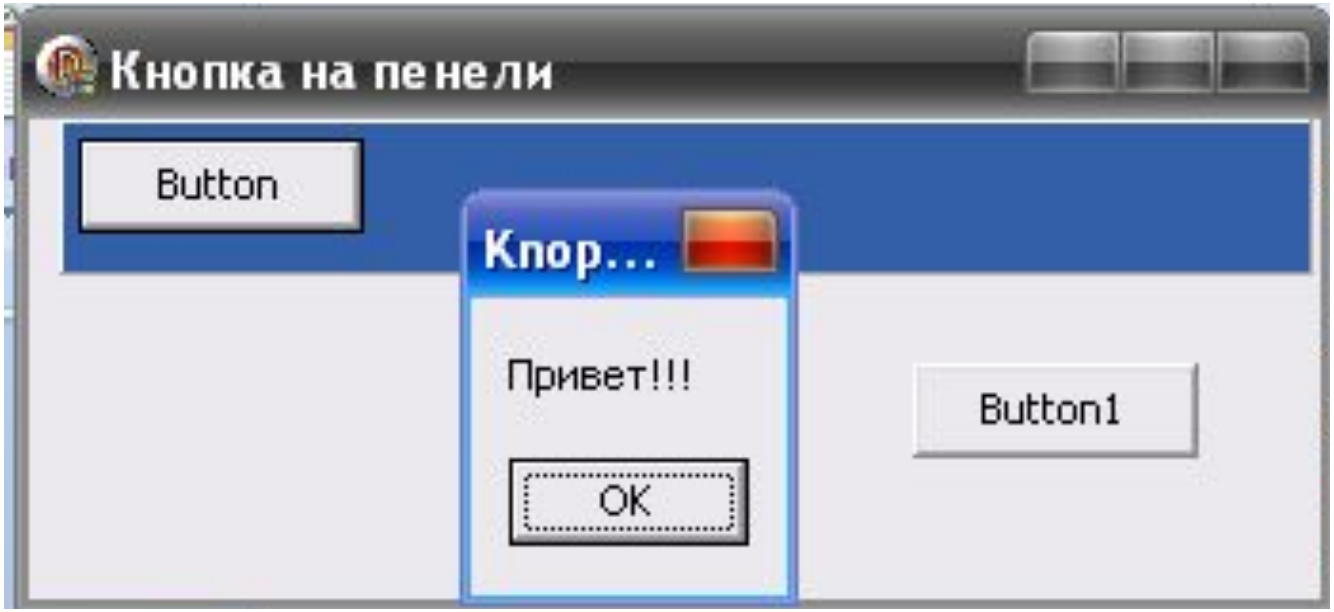
Невизуальные – невидимые компоненты (часы, диалоговые окна, …)

2. Пример: создадим компоненты диалоговых окон

```
procedure TForm1.Button2Click(Sender: TObject);
Var
      Y:  TFontDialog; // переменная типа TFontDialog
      Cvet: TColorDialog;    //тип TColorDialog
begin
  y:=TFontdialog.Create(self);
  y.Execute;

  Cvet:=TColorDialog.Create(Form1);
  Cvet.Execute;
  Form1.Color:=Cvet.Color;
End;
end.
```

3. Пример: создадим кнопку на панели, на которую создадим процедуру



```
procedure TForm1.Button1Click(Sender:
TObject);
 Var New : TButton;
begin
    New:= TButton.Create(Panel1);
    New.Top:=5;
    New.Left:=5;
    New.Name:='Button';
    New.OnClick:=onClickButton;
   //процедура
    New.Parent:=Panel1;
end;
```

В разделе класса формы декларации собственных процедур пропишите процедуру:

private

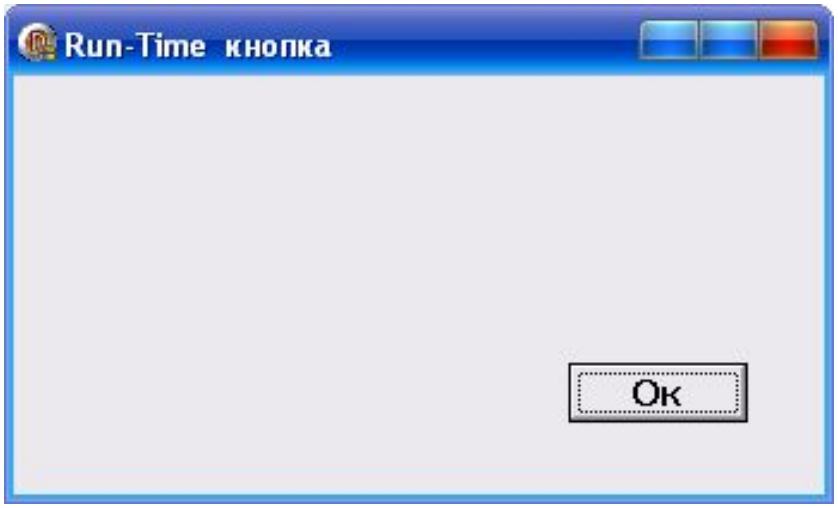
procedure onClickButton(Sender: TObject);

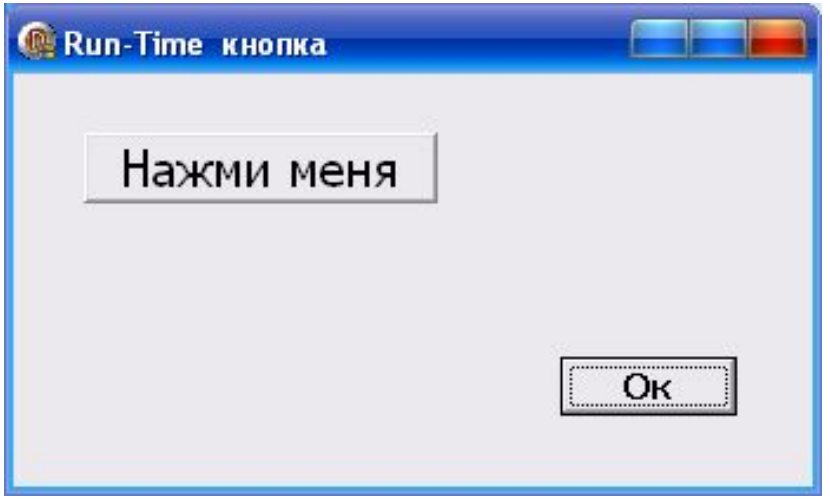
//ввести код процедуры

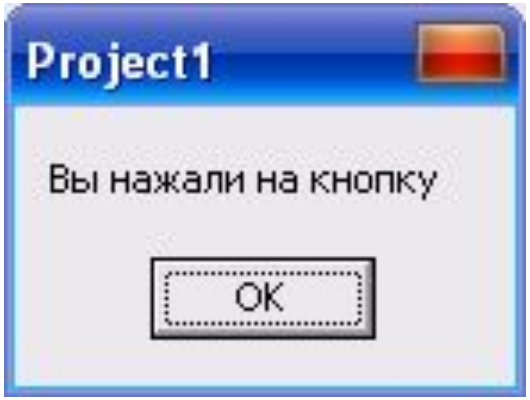
```
procedure TForm1.onClickButton(Sender: TObject);
  begin
    ShowMessage('Привет!!!');
  end;
```

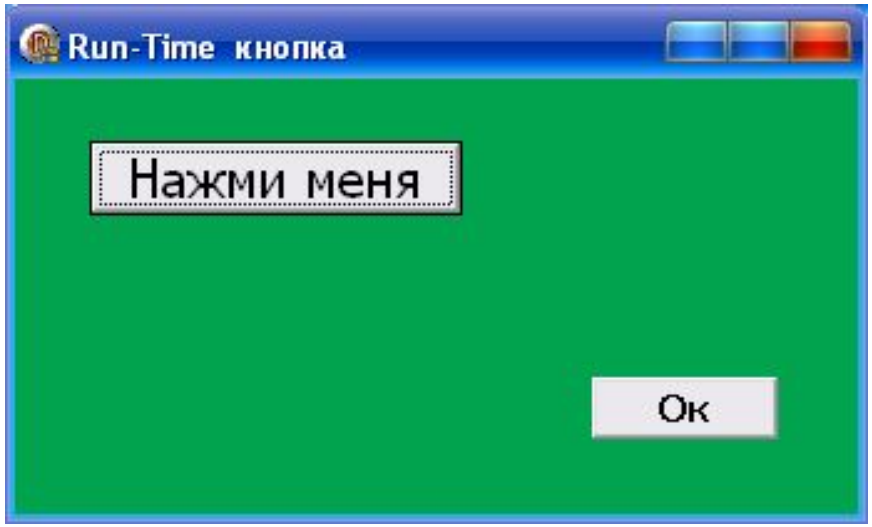
Ctrl + Shift + C

4. Пример


Run-Time кнопка
Ок

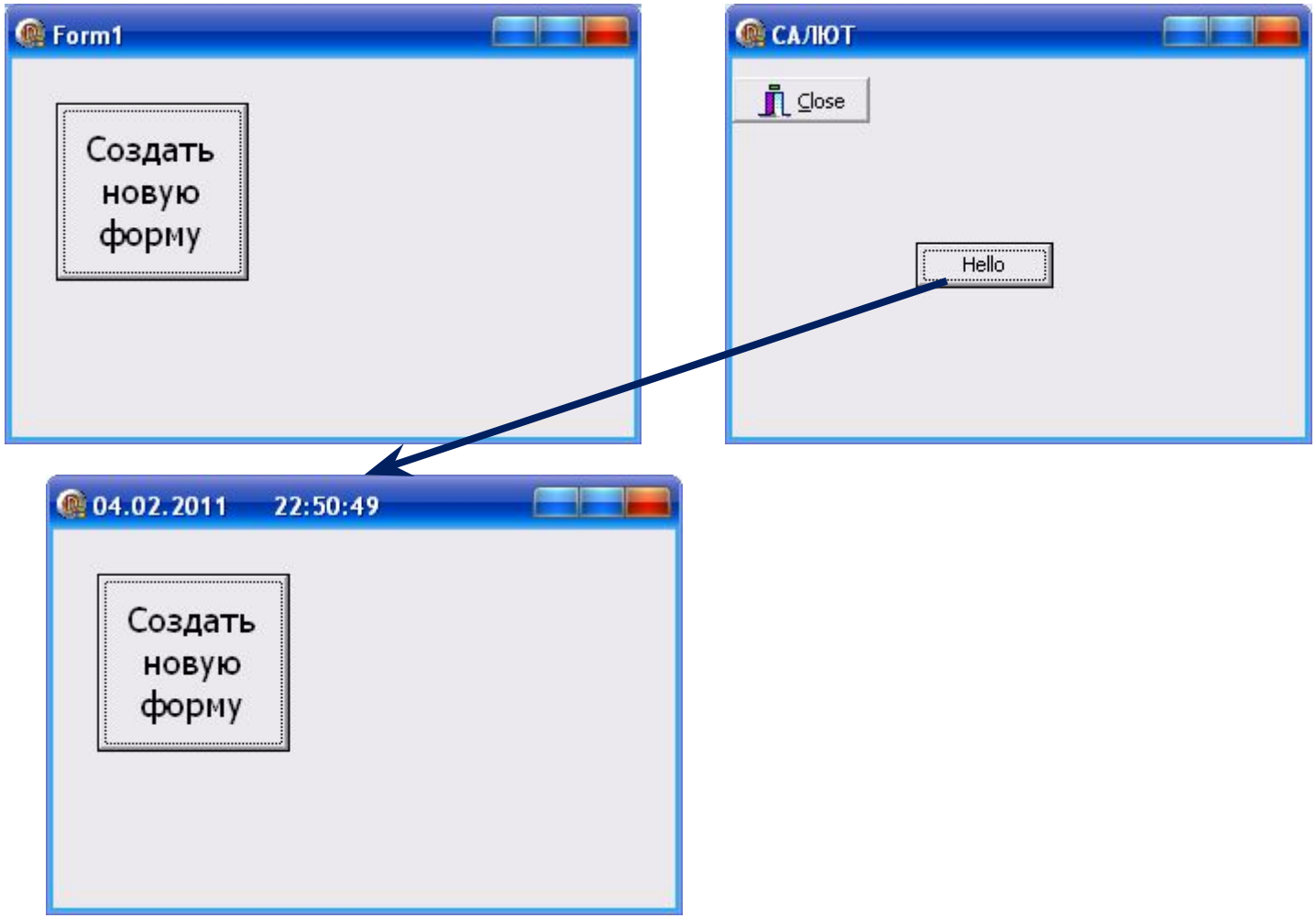

Run-Time кнопка
Нажми меня
Ок


Project1
Вы нажали на кнопку
OK


Run-Time кнопка
Нажми меня
Ок

```
 private
  procedure click(Sender: TObject);
...
procedure TForm1.Click(Sender: TObject);
 begin
    Randomize;
     ShowMessage('Вы нажали на кнопку');
     Form1.Color:=Random(9955565);
 end;
procedure TForm1.Button1Click(Sender: TObject);
    Var  New: TButton;
   begin
   New:=TButton.Create(Form1);
   New.parent:=Form1;
   New.Top:=25;    New.Left:=30;      New.Caption:='Нажми меня';
   New.Height:=30;    New.Width:=150;     New.Font.Size:=16;
   TButton(New).OnClick:=Click;    // ССЫЛКА НА СОБЫТИЕ
   end;
End.
```

5. Пример Можно создать и новую форму, а на ней в свою очередь - кнопку



```
 private
       procedure Click(Sender: TObject);
var
  Form1: TForm1;
     Form_2: Tform;  // КОМПОНЕНТ ТИПА
ФОРМЫ
     but:Tbutton;   //КОМПОНЕНТ ТИПА КНОПКИ
procedure TForm1.Click(Sender: TObject);
begin
  Caption:= DateToStr(Date)+'     ' +TimeToStr(Time)
End;
```

```
procedure TForm1.Button1Click(Sender: TObject);
Var       bb:TBitBtn;  //КОМПОНЕНТ ТИПА  TBitBtn
begin
//создаем форму, а на ней кнопку
  form_2:=Tform.Create(self);
  but:=Tbutton.create(form_2);
  but.Parent:=form_2;
  but.Top:=100;       but.Left:=100;       but.Caption:='Hello';
  form_2.Show;
  Form_2.Caption:='САЛЮТ ';      Form_2.Top:=300;
Form_2.Left:=300;
   but.OnClick:=click;

    //создаем на второй форме  TBitBtn
  bb:=TBitBtn.create(Form_2);
  bb.Parent:=Form_2;    bb.Top:=10;     bb.Kind:=bkClose;
 end;
end.
```

6. Пример: Пусть необходимо отсортировать данные, хранящиеся в Memo. Сам Memo не обладает свойством, которое бы позволяло выполнять сортировку строк. Но при этом Memo работает с информацией строкового типа, а в Delphi имеется абстрактный объект **TstringList**, не просто работающий с такой информацией, но и содержащий необходимый нам метод сортировки.

```
procedure TForm1.Button1Click(Sender: TObject);
var  y:  Tstringlist;
begin
  y:=Tstringlist.Create;     //нет родителя
  y.addstrings(memo1.Lines);  //в объект Y присваиваем построчно текст из Memo
  y.Sort;          //сортируем
  memo2.Clear;
  memo2.Lines.addstrings(y);
end;
end.
```

Или вот еще ***пример 7:*** пусть необходимо залить форму некоторым узором. Здесь, прежде всего нужно решить: какого типа картинка будет использована для повторения – **bmp** или **jpg**.

Если вы хотите использовать картинку типа **bmp**, тогда нужно начать с объявления переменной типом **Tbitmap**, а если вы будете использовать картинку типа **jpg**, то в этом случае нужно будет объявить переменную типом **TJPEGImage**. Ну, а дальше, что-то вроде этого:

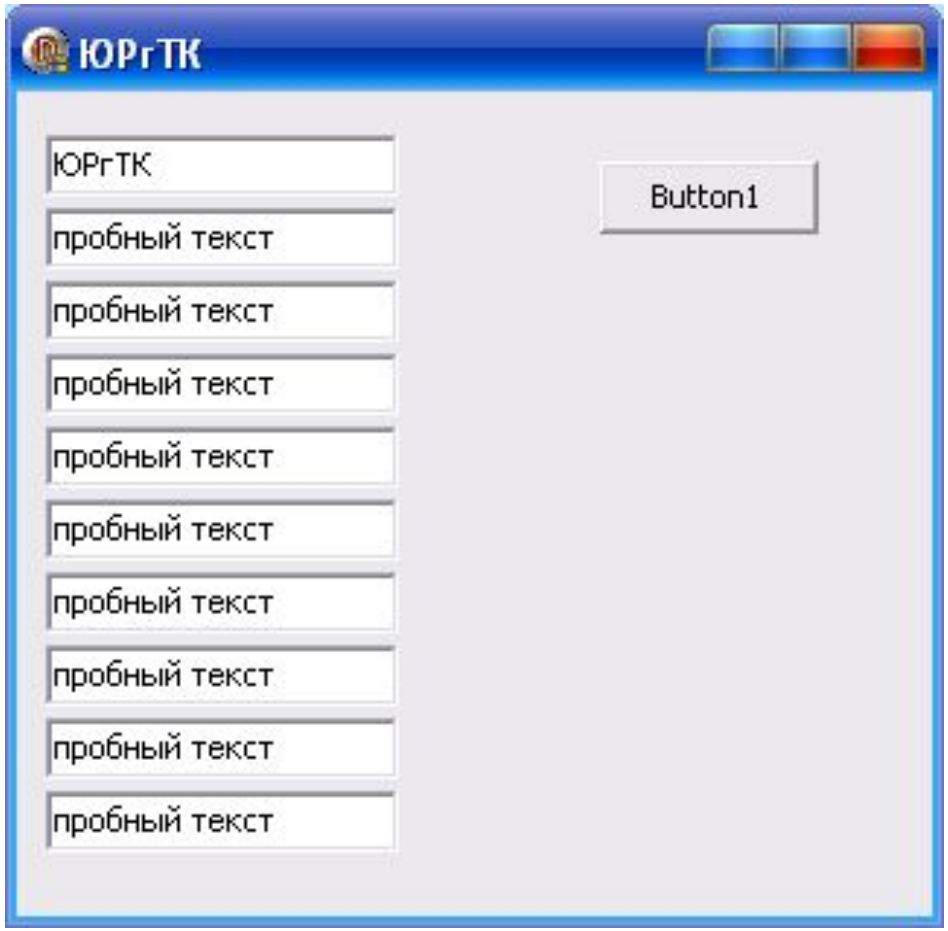

Фон на форме

```
Uses        …,  JPEG;
procedure TForm1.FormDblClick(Sender: TObject);
var uz: TJPEGImage; x,y1:integer;    y:Topendialog;
begin
  y:=Topendialog.Create(self);      y.Execute;
  uz:=TJpegImage.Create;
  uz.LoadFromFile(y.filename);
  x:=0; y1:=0;    //при изменении ее размеров.
   repeat
         repeat
       form1.canvas.draw(x,y1,uz);
            x:=x+uz.Width;
        until (x>form1.ClientWidth);
        x:=0;
        y1:=y1+uz.Height;
   until (y1>form1.ClientHeight);
end;   end.
```

Пример 8 :

```
var    Image:TImage;
 procedure TForm1.FormCreate(Sender: TObject);
begin
        image:=TImage.Create(Form1);
        image.parent:=Form1;
        Image.Height:=Form1.ClientHeight;
        Image.Width:=Form1.ClientWidth;
        Image.Stretch:=True;
        image.picture.loadfromfile('001.jpg');
end;
procedure TForm1.FormPaint(Sender: TObject);
begin
          Image.Height:=Form1.Height;
          Image.Width:=Form1.Width;
end;
end.
```

Пример 9 : Создадим массив из 10 меток, по вводу текста в которые текст переходит в заголовок формы

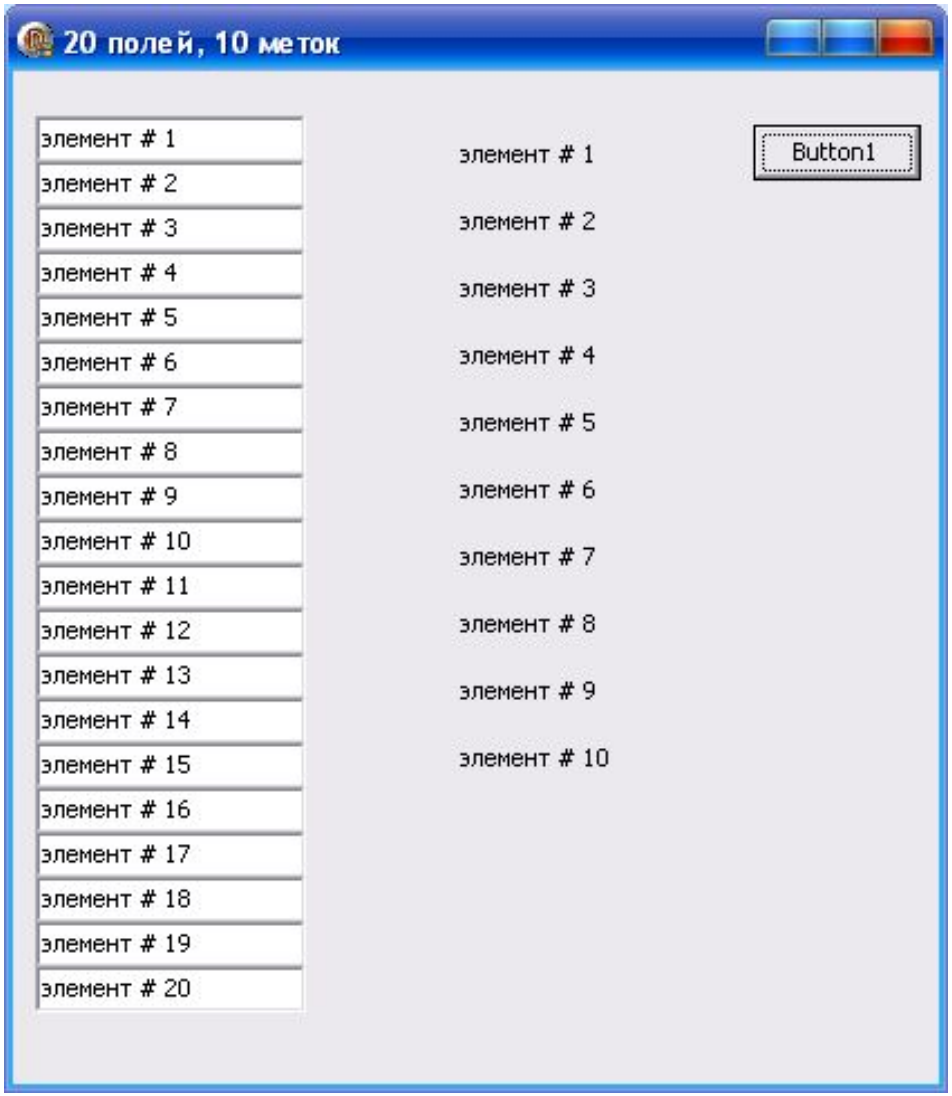


```
Var      Form1: TForm1;
   mas:array[1..10] of TEdit;   //массив
procedure TForm1.FormCreate(Sender: TObject);
begin
   Form1.caption:='';
end;

procedure TForm1.OnEditChange(Sender:
TObject);
Begin
   Form1.caption:=TEdit(sender).text;
end;
```

```
procedure TForm1.Button1Click(Sender: TObject);
Var     i:integer;
begin
   for i:= 1 to 10 do begin
       Mas[i]:=TEdit.Create(Button1);
       Mas[i].parent:=form1;
       Mas[i].Top:=-10+i*25;  //расположение со сдвигом вниз
       Mas[i].Left:=10;
       Mas[i].Text:='пробный текст';
      TEdit(Mas[i]).OnChange:=OnEditChange; //присваиваем
     //процедуру для обработчика события    ИЛИ
                    //Mas[i].OnChange:=OnEditChange;
end;
end;
End.
```

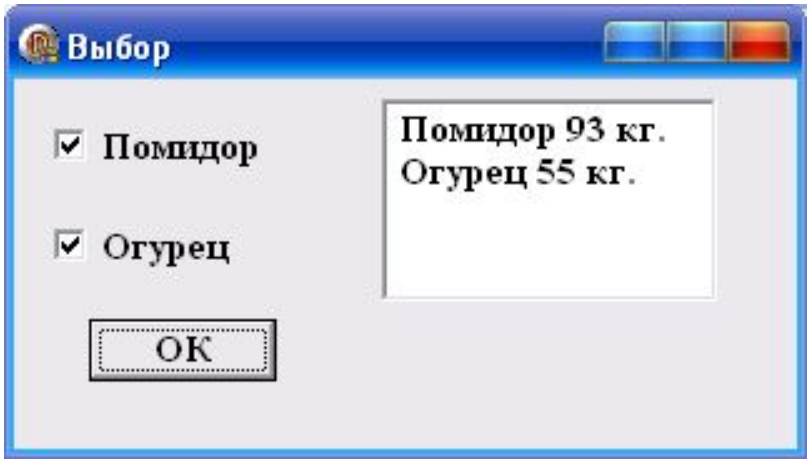

массив меток
OK

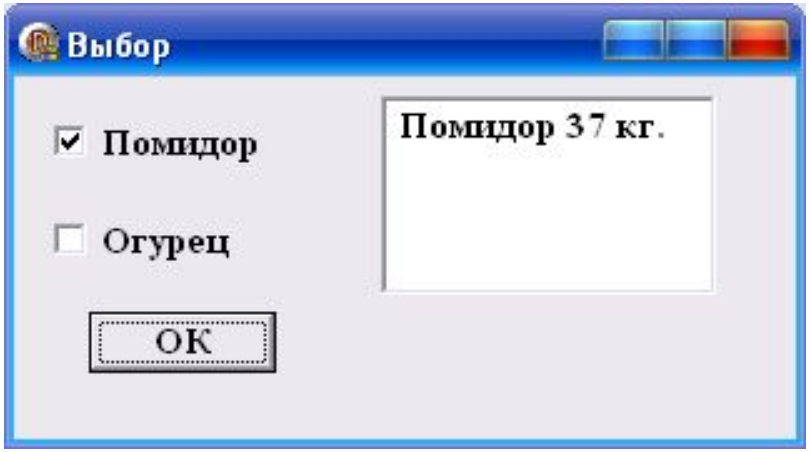

20 полей, 10 меток
элемент # 1
элемент # 2
элемент # 3
элемент # 4
элемент # 5
элемент # 6
элемент # 7
элемент # 8
элемент # 9
элемент # 10
элемент # 11
элемент # 12
элемент # 13
элемент # 14
элемент # 15
элемент # 16
элемент # 17
элемент # 18
элемент # 19
элемент # 20
элемент # 1
элемент # 2
элемент # 3
элемент # 4
элемент # 5
элемент # 6
элемент # 7
элемент # 8
элемент # 9
элемент # 10
Button1

Все компоненты формы хранятся в ее свойстве **Components** в виде массива (**Components[i]** по умолчанию массив организуется самостоятельно).

Количество этих компонентов хранится в свойстве **ComponentCount** (нумеруются с 0).

Пример 10



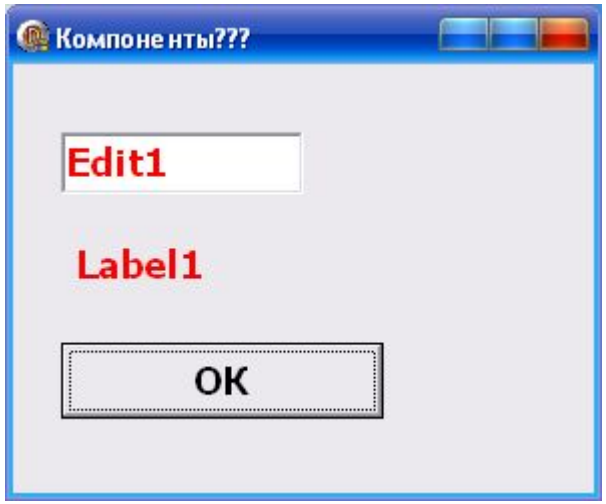


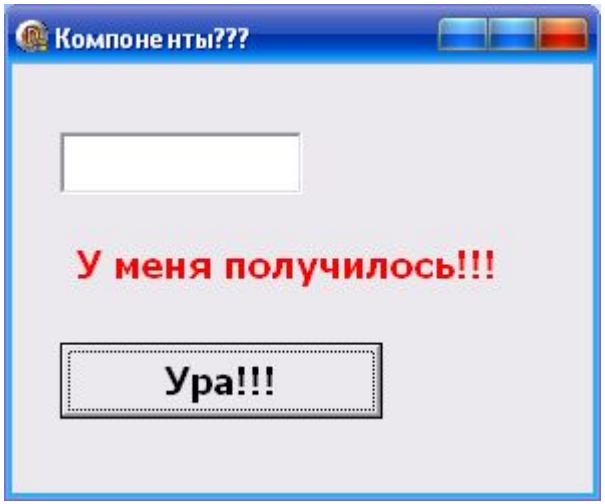
Компоненты созданы в режиме проектирования формы.

В коде программы проверяется, что если компонент относится к типу флажок и если флажок выбран, то в объект MEMO добавляется заголовок флажка и случайная денежная единица.

```
procedure TForm1.Button1Click(Sender: TObject);
var i:integer;
begin
Randomize;        memo1.Clear;
 for i:=0 to componentcount-1 do      begin
    IF (components[i] is TCheckBox) and
       (TCheckBox(components[i]).Checked)
 then
memo1.Lines.Add(TCheckBox(components[i]).Caption+
       ' '+IntToStr(Random(100))+' кг.');
   end;
end;
end.
```

Пример 11





Если компонент текстовое поле, то его очистить,
если компонент метка, то в нем написать «У меня получилось»,
если компонент кнопка, то в заголовке кнопки написать «Ура!!».
(в этом случае все компоненты сделаны в режиме **Design-time**)

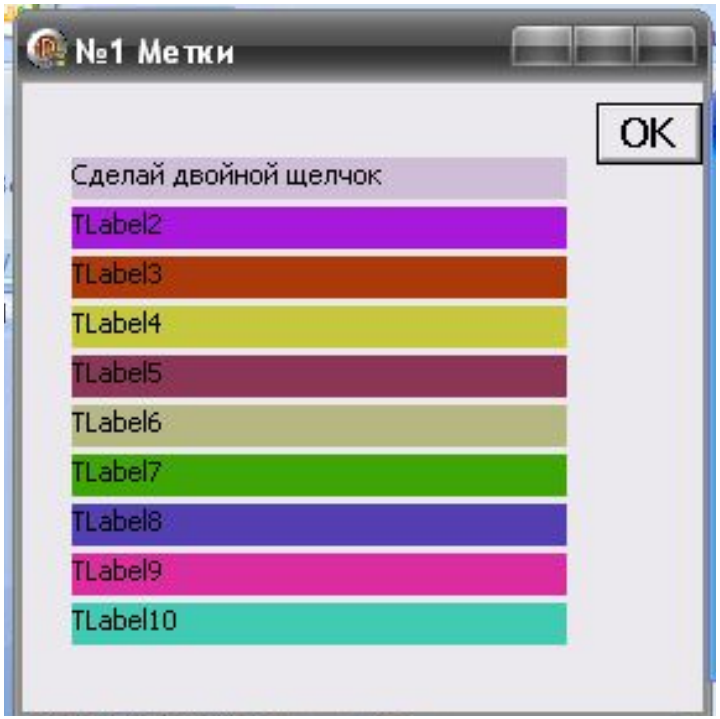
```
procedure TForm1.Button1Click(Sender: TObject);
var i:integer;
begin
  for i:=0 to componentcount - 1   do   begin
     if (components[i] is TEdit)   then
                    Tedit(components[i]).Text:='';
     if (components[i] is TButton)  then
          TButton(components[i]).Caption:='Ура!!!';
     if (components[i] is TLabel)   then
          TLabel(components[i]).Caption:='У меня …!';
   end;
end;
end.
```

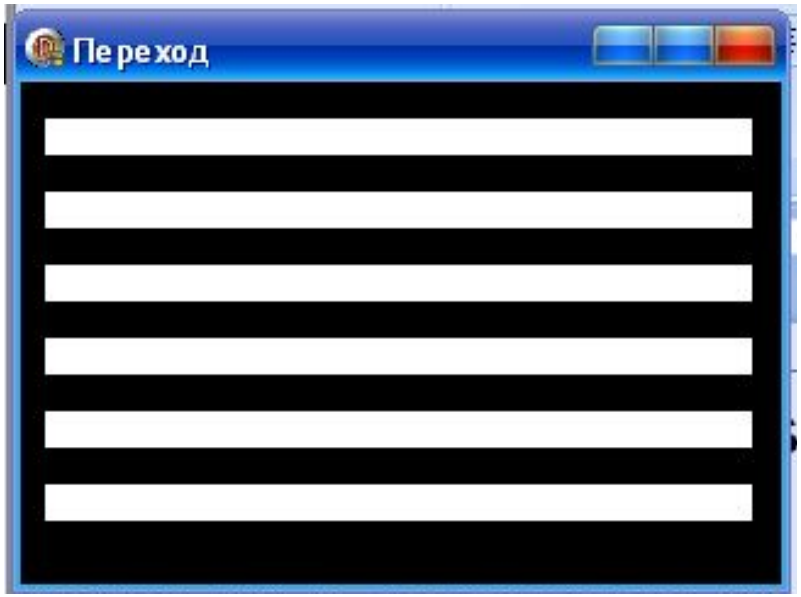
Проверка идет в цикле по всем компонентам на форме

New.Name:=New.ClassName+'1';

Объекту с именем New присвоить классическое имя объекта с цифрой 1

(если объект New метка то имя его будет Label1)


№1 Метки
OK
Сделай двойной щелчок
TLabel2
TLabel3
TLabel4
TLabel5
TLabel6
TLabel7
TLabel8
TLabel9
TLabel10


Переход


Объекты
Label2
Edit1
CheckBox
RadioButto
OK
Label1


Объекты
УРА!!!
Edit1
CheckBox
RadioButto
OK
УРА!!!


Рисунки
1
2
3
4
5
6
7
8


Рисунки
1
2
3
4
5
6
7
8


Рисунки
1
2
3
4
5
6
7
8


Таблица умножения на 5
1 * 5 = 5
2 * 5 = 10
3 * 5 = 15
4 * 5 = 20
5 * 5 = 25
6 * 5 = 30
7 * 5 = 35
8 * 5 = 40
9 * 5 = 45
10 * 5 = 50
OK
Сброс
Новая таблица
Нечет

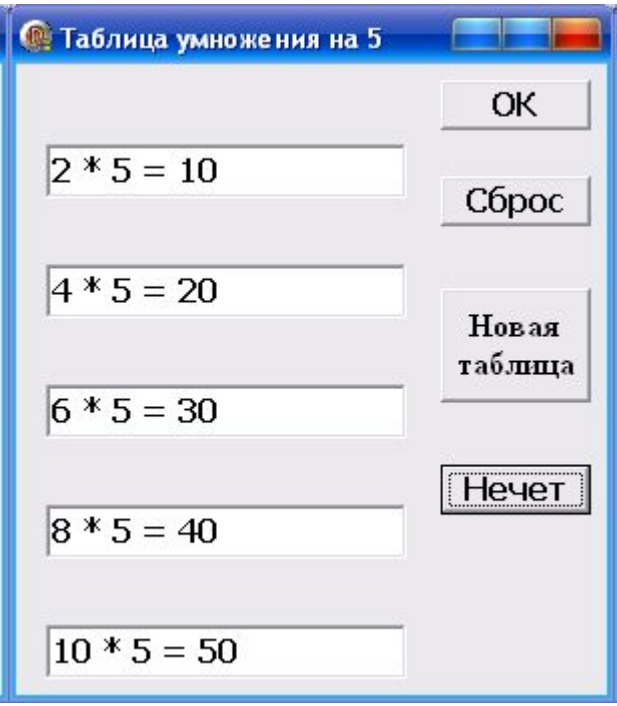


Таблица умножения на 5
2 * 5 = 10
4 * 5 = 20
6 * 5 = 30
8 * 5 = 40
10 * 5 = 50
OK
Сброс
Новая таблица
Нечет

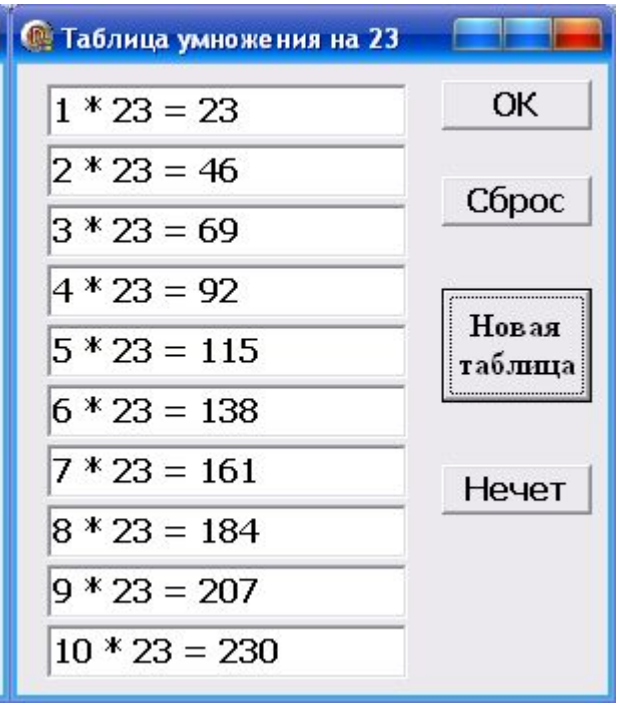


Таблица умножения на 23
1 * 23 = 23
2 * 23 = 46
3 * 23 = 69
4 * 23 = 92
5 * 23 = 115
6 * 23 = 138
7 * 23 = 161
8 * 23 = 184
9 * 23 = 207
10 * 23 = 230
OK
Сброс
Новая таблица
Нечет


флажки
OK
19
17
20
16
17
17
20
19
18
20
Выбрали - 3
Количество четных 0
Количество нечетных 3


Заказ
МЕНЮ НА 05.02.2011
Пирожок 12 2
Пирожное 15
Мороженое 9 2
салат 21 1
Суп 34 1
Каша 27 1
Пельмени 79
Кофе 11 1
Чебурек 20
чай 7
135,00р.
Лист заказа
Сумма


Project1
Пирожок 12 * 2 = 24,00р.
Мороженое 9 * 2 = 18,00р.
салат 21 * 1 = 21,00р.
Суп 34 * 1 = 34,00р.
Каша 27 * 1 = 27,00р.
Кофе 11 * 1 = 11,00р.
ИТОГ: 135,00р.
OK