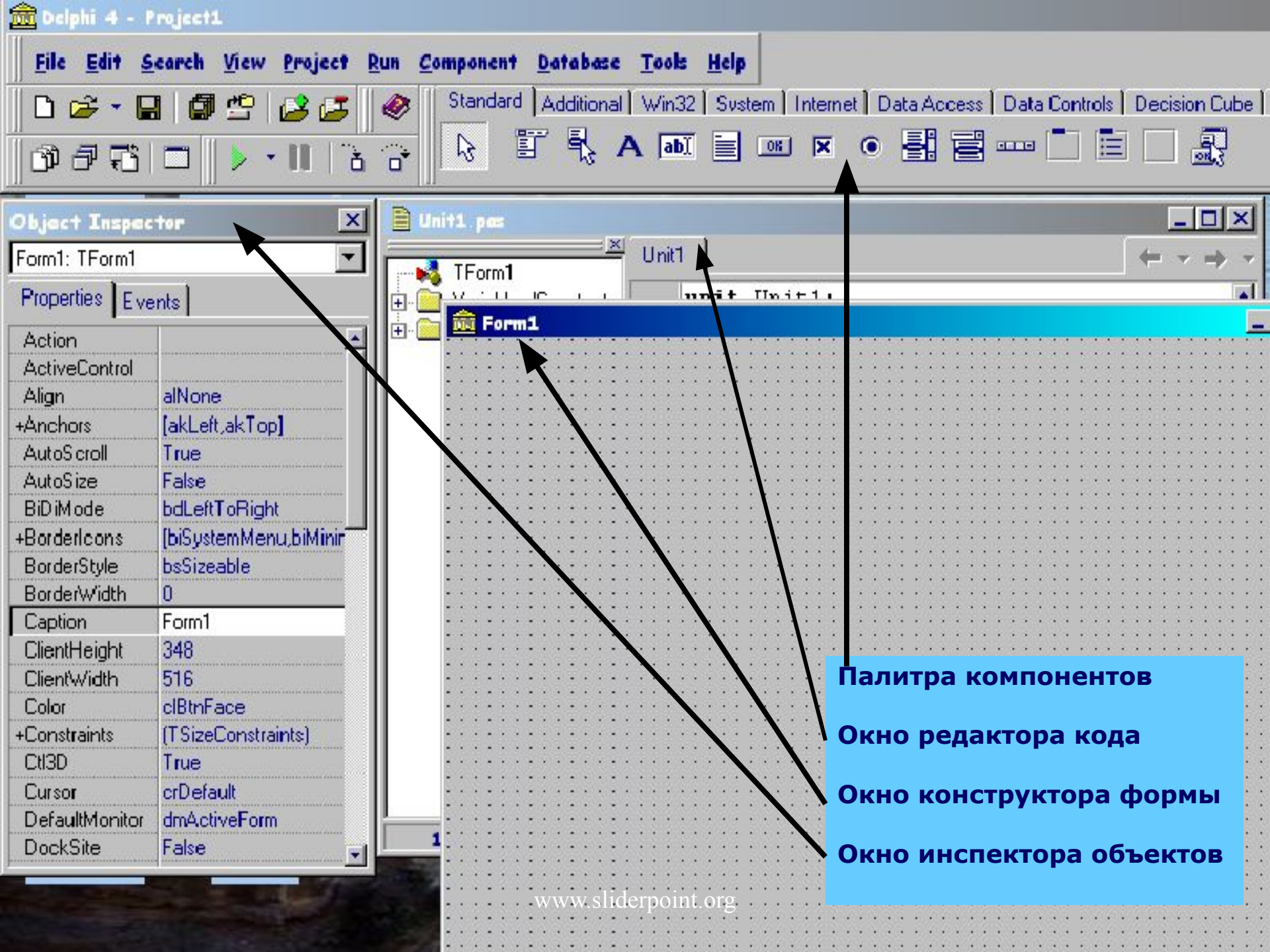


Лекция 2

Delphi



Object Inspector

Form1: TForm1

Properties Events

Action	
ActiveControl	
Align	alNone
+Anchors	[akLeft,akTop]
AutoScroll	True
AutoSize	False
BiDiMode	bdLeftToRight
+BorderIcons	[biSystemMenu,biMinimize]
BorderStyle	bsSizeable
BorderWidth	0
Caption	Form1
ClientHeight	348
ClientWidth	516
Color	clBtnFace
+Constraints	[TSizeConstraints]
Ctl3D	True
Cursor	crDefault
DefaultMonitor	dm ActiveForm
DockSite	False

Unit1.pas

Unit1

TForm1

Form1

Палитра компонентов
Окно редактора кода
Окно конструктора формы
Окно инспектора объектов

Окно редактора кода

Окно редактора кода (**Unit1.pas**) создает исходный текст модуля разрабатываемого приложения. Каждый редактируемый файл находится на отдельной странице, доступ к которой можно получить, щелкнув на ее ярлычке.

Окно конструктора формы

Окно конструктора формы (**Form1**) в нем выполняется проектирование формы. Для чего на форму из Палитры компонентов помещаются необходимые элементы.

Окно инспектора объектов

Окно инспектора объектов отображает свойства и события объектов для текущей формы **Form1**. Его можно отобразить с помощью команды **View / Object Inspector** или нажатия клавиши **<F11>**. Окно инспектора объектов имеет 2 вкладки **Properties** (Свойства) и **Events** (События).

Расширения файлов

- .Dpr** **Основной файл проекта!!**
- .Dfm** **Файл формы !!**
- .Pas** **Файл модуля Паскаля !!**
- .Res** **Файлы ресурсов !(создается
при первом сохранении)**

Обязательные
файлы
для работы
приложения

- .Opt** **Файл параметров проекта**
- .~*** **Резервные копии файлов**
- .Dcr** **Delphi Component Resource !**
- .Dcu** **Скомпилированный файл модуля**
- .Dsm** **Промежуточный код**
- .Dll** **Файлы динамических библиотек**
- .Dsk** **Конфигурация рабочей области**
- .Exe** **Исполняемые файлы**

Пример простого приложения

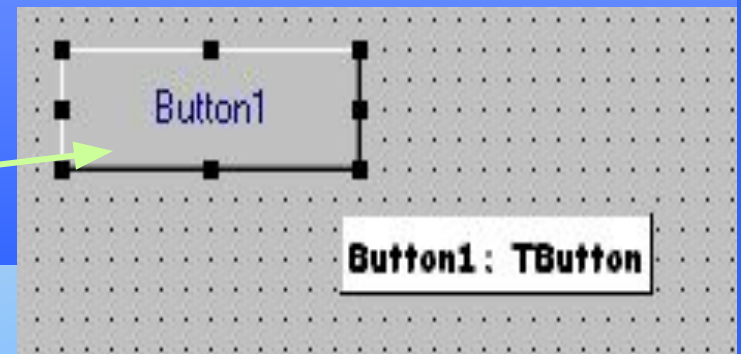
Создание нового приложения начинается с того, что Delphi автоматически предлагает пустое окно - форму **Form1**.

Интерфейс приложения составляют компоненты, которые можно выбирать из **Палитры компонентов**, размещать их на форме и изменять их свойства.

Выбор компонента в Палитре выполняется щелчком мыши на нужном компоненте, например, кнопке **Button** и после этого щелкнуть на форме. На ней появится выбранный компонент. После размещения компонента на форме Delphi автоматически вносит изменения в файл модуля, т.е. добавляется строка **<Название компонента>:<Тип компонента>**

```
TForm1 = class(TForm)
  Button1: TButton;
```

Внешний вид компонента определяют его свойства, которые доступны в окне **Инспектора объектов**, когда компонент на форме выделен.





В окне Инспектора объектов приводятся названия всех свойств и их значения.

Свойства представляют собой атрибуты, определяющие способ отображения и функционирования компонентов при выполнении приложения. Изменять значения свойств можно непосредственно в Инспекторе, при этом сразу же изменяется соответствующий компонент.

Дадим нашей кнопке другой заголовок:

напишем в строке **Caption** - **About Form**

Некоторые свойства Button:

Cursor:TCursor - изображение мыши при наведении на кнопку (например: `crHandPoint` )

Default:Boolean - нажатие кнопки выполняется по клавише Enter, если значение Default=True

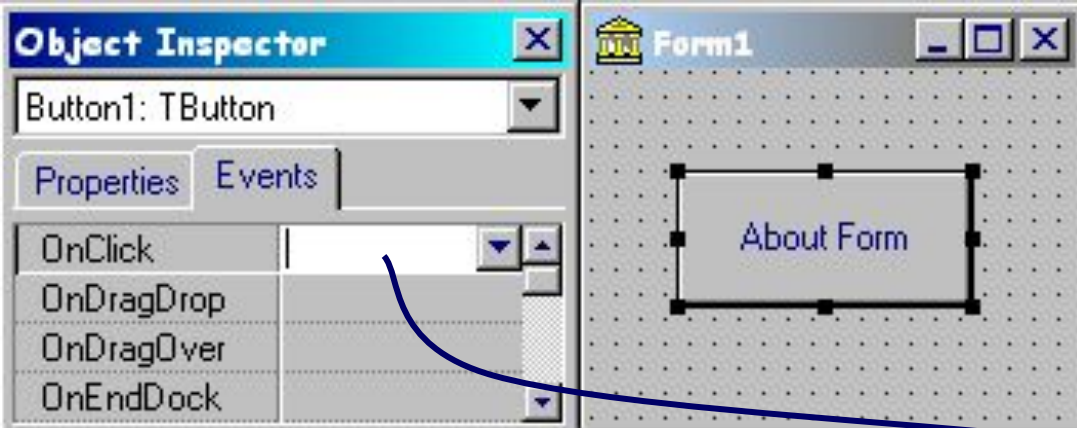
Font:TFont - шрифт

Height:Integer - высота кнопки в пикселях

Width:Integer - ширина кнопки в пикселях

Left:Integer - горизонтальная координата левого верхнего угла.

Top:Integer - вертикальная координата левого верхнего угла.



Чтобы кнопка могла реагировать на какое либо событие, необходимо указать **процедуру обработки события**, которая будет вызываться при возникновении данного события.

Для этого нужно сделать двойной щелчок в области события **OnClick** (или двойной щелчок по кнопке **About Form**), в результате Delphi автоматически создает в модуле формы заготовку **процедуры-обработчика**.

```
implementation
  {$R *.DFM}
  procedure TForm1.Button1Click(Sender: TObject);
  begin
  |
  end;
end.
```

Напишем в процедуру

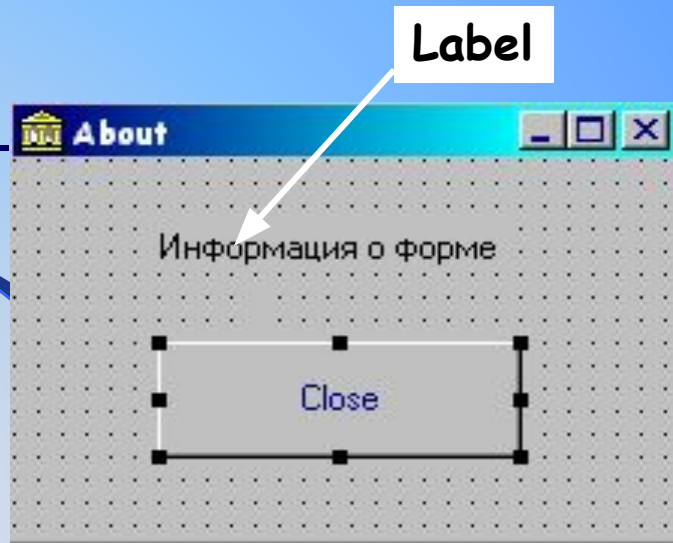
```
var About: TAbout;
begin
  About:=TAbout.Create(self);
  About.Show;
end;
```

Теперь при нажатии на кнопку **About Form** будет появляться еще одна форма по имени **About**, которую мы опишем в модуле Unit2.

Опишем вторую форму в Unit2.pas. В главном меню выберем пункт **File / New / Form**

Зададим этой форме **Caption** и **Name - About**

Поместим на форму кнопку **Button** с заголовком **Close** и компоненту **Label**, который представляет собой простой текст, который не редактируется при выполнении программы. Текст вводится в поле **Caption**.



Некоторые свойства Label

AutoSize: Boolean - автоматическая коррекция размера **Label**, в зависимости от текста надписи

Aligment: TAligment - способ выравнивания текста внутри компонента, может принимать следующие значения:

- **taLeftJustify** - выравнивание по левому краю
- **taCenter** - центрирование текста
- **taRightJustify** - выравнивание по правому краю

WordWrap: Boolean - автоматический перенос слов на другую строку

Transparent: Boolean - прозрачная надпись или закрашенная. Цвет закрашивания устанавливается свойством **Color**.

При нажатии на кнопку **Close** сделаем так, чтобы форма About закрывалась.

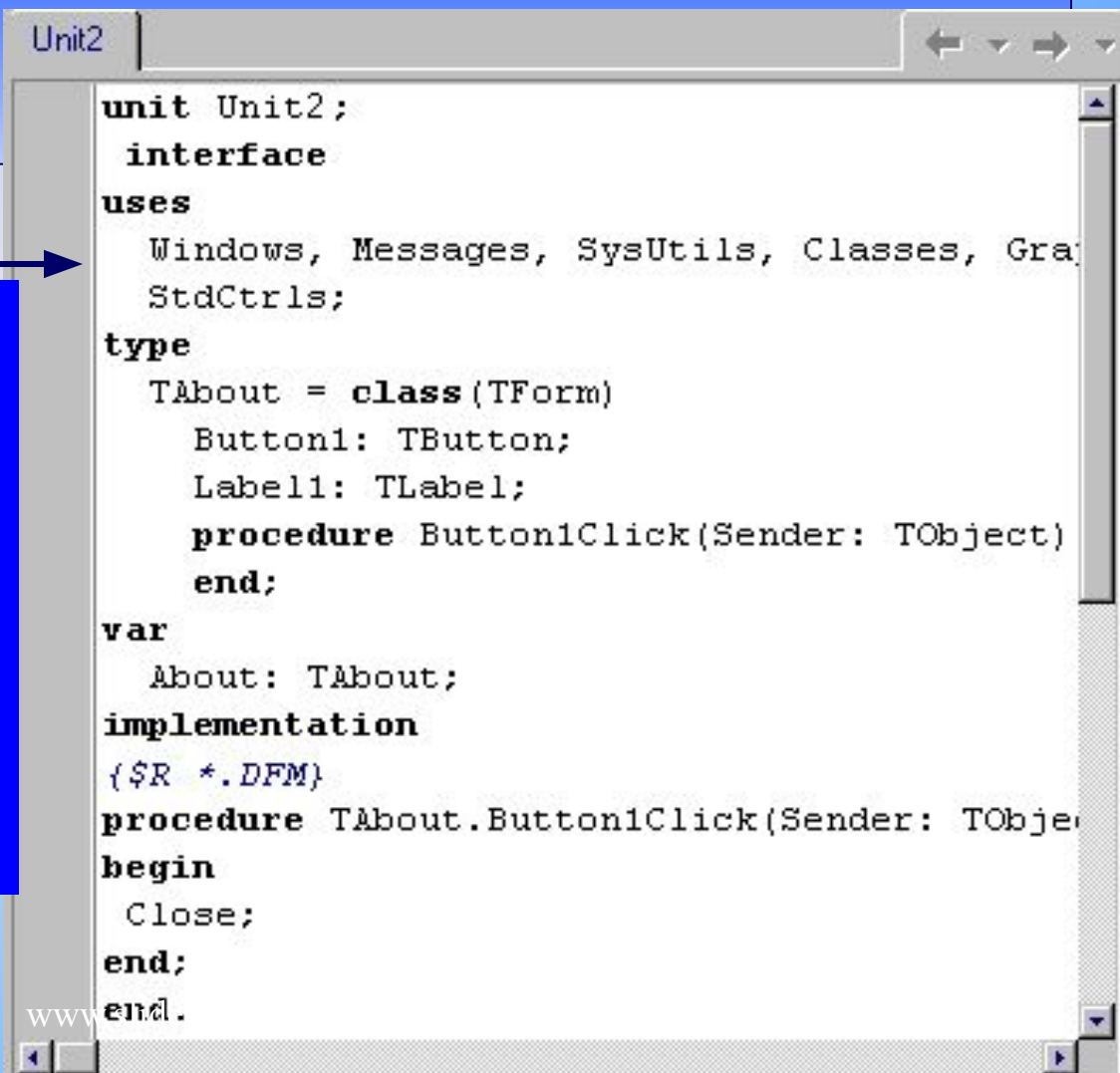
```
procedure TAbout.Button1Click(Sender: TObject);  
begin  
    Close;  
end;
```

Наш модуль Unit2 имеет вид

При нажатии кнопки **About Form** первой формы на экране отображается вторая форма, которая до этого была невидима.

Т.к. из модуля **Unit1** осуществляется операция со второй формой, то в разделе **implementation** модуля **Unit1** поместим код **uses Unit2**.
Или **File / Use Unit...**

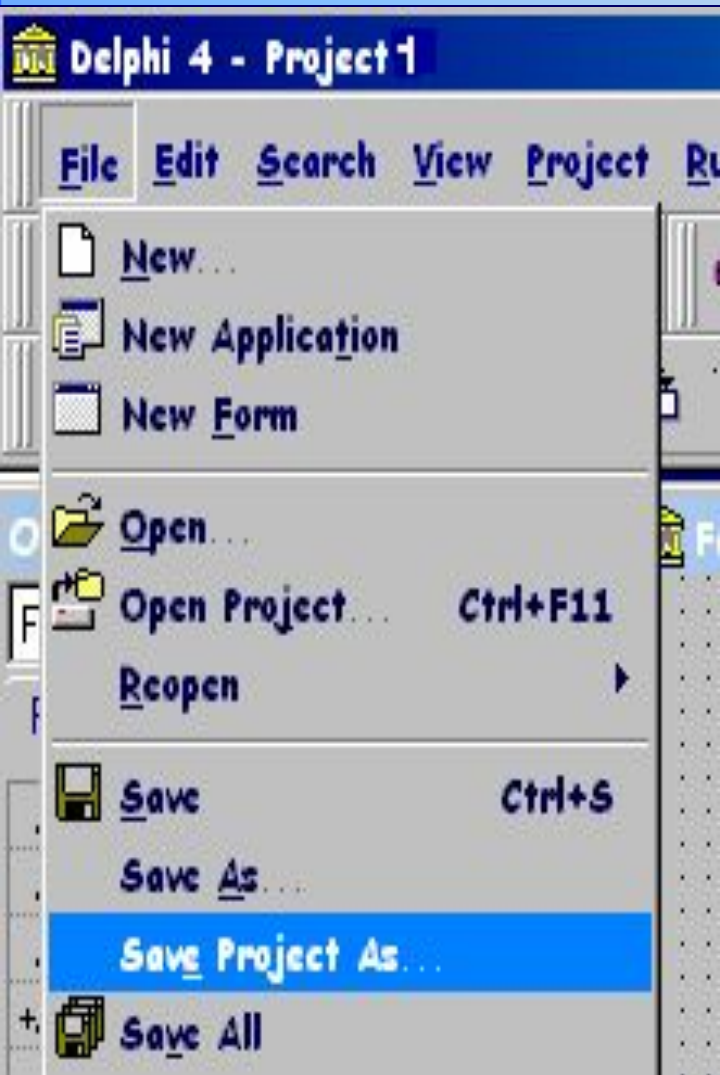
```
implementation  
uses unit2;
```



```
Unit2  
unit Unit2;  
interface  
uses  
    Windows, Messages, SysUtils, Classes, Gra  
    StdCtrls;  
type  
    TAbout = class (TForm)  
        Button1: TButton;  
        Label1: TLabel;  
        procedure Button1Click(Sender: TObject)  
        end;  
var  
    About: TAbout;  
implementation  
    {$R *.DFM}  
    procedure TAbout.Button1Click(Sender: TObject)  
    begin  
        Close;  
    end;  
end.
```

Сохранение проекта

Для того, чтобы сохранить проект, в главном меню выберем пункт **File / Save Project As...**



Дадим нашему проекту имя **Project1**

Компиляция

Запуск процесса компиляции выполняется по команде:

Project / Compile <Project1 >

Компиляция проекта для получения приложения может быть выполнена на любой стадии разработки проекта.

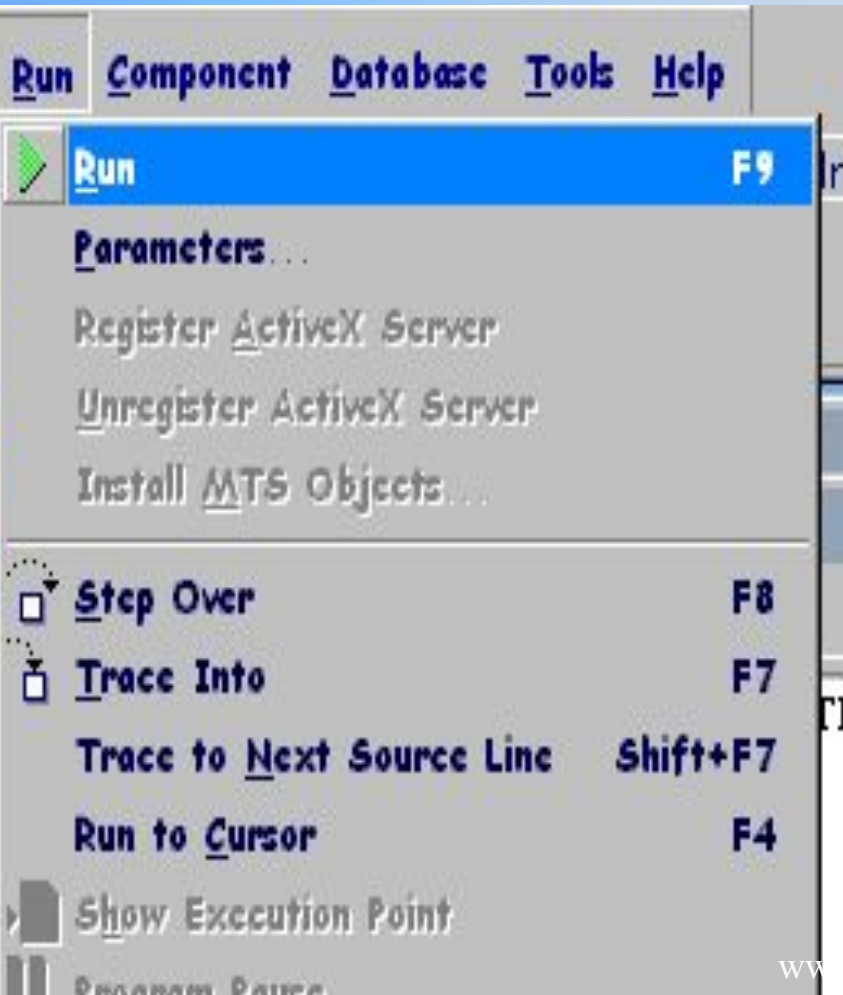
Это удобно для проверки вида и правильности функционирования отдельных компонентов формы, а также для проверки создаваемого кода.

После компиляции проекта создается исполняемый файл приложения с именем файла проекта .



Запуск проекта

Запустить проект на выполнение можно из среды Delphi и из среды Windows. Из среды Delphi осуществляется командой **Run / Run** или нажатием клавиши **<F9>**



Замечание

- Нельзя запустить вторую копию приложения при уже запущенном приложении.
- Продолжить разработку проекта можно только после завершения работы приложения.
- При заикливания (зависании) приложения завершение его работы необходимо выполнять средствами Delphi с помощью команды **Run / Program Reset** или нажатия клавиш **<Ctrl>+<F2>**

Приложение которое мы создали будет иметь вид:

