Цикл презентаций «ООП на Delphi» посвящен объектно – ориентированному программированию с использованием одной из самых распространенных систем быстрой разработки приложений – Delphi

Используя данный учебный курс, можно самостоятельно овладеть основами объектно – ориентированного программирования на Delphi. Для расширения Ваших знаний к курсу приложен ряд учебных пособий и справочников по Delphi

Цикл содержит 15 презентаций:

ООП на Delphi – 1: Знакомство с системой программирования Borland Delphi. Объекты (компоненты) и их свойства и методы

ООП на Delphi – 2: Первая программа на Delphi, сохранение и компиляция

ООП на Delphi – 3: Программное изменение свойств объектов

ООП на Delphi – 4: Условия в Delphi. Создание простого теста

ООП на Delphi – 5: Элементы ввода и вывода информации. Обработка исключений

ООП на Delphi – 6: Заставка программы и элемент таймер

ООП на Delphi – 7: Программируем свою игрушку

ООП на Delphi – 8: Меню программы, диалоги

ООП на Delphi – 9: Создаем свой текстовый редактор

ООП на Delphi – 10: Базы данных на Delphi

ООП на Delphi – 11: Калькулятор на Delphi. Обработка исключительных ситуаций

ООП на Delphi – 12: Создаем тестирующую систему

ООП на Delphi – 13: Графика на Delphi

Delphi использует язык программирования Объект Паскаль, поэтому лучше сначала изучить обычный Паскаль и поработать в ТурбоПаскале, а затем и переходить к Delphi – перейти будет очень просто, т.к синтаксис языка остается неизменным.

Изучение ООП на Delphi желательно проводить в старших профильных классах – количество часов, отводимое на информатику там вполне достаточно для освоения основ ООП на Delphi

Объектно – ориентированное программирование на

Borland® DELPHI - 2

DELPHI - 2

На этом уроке:

Мы научимся создавать программу, сохранять и компилировать ее, а также читать исходный код

Вопросы:

1. Создание программы, сохранение и компиляция
2. Разбираемся с исходным кодом

Borland®

1. Создание программы, сохранение и компиляция

На первом уроке мы познакомились с интерфейсом Delphi, компонентами и их свойствами.

Сейчас давайте научимся **создавать, правильно сохранять и компилировать программу**

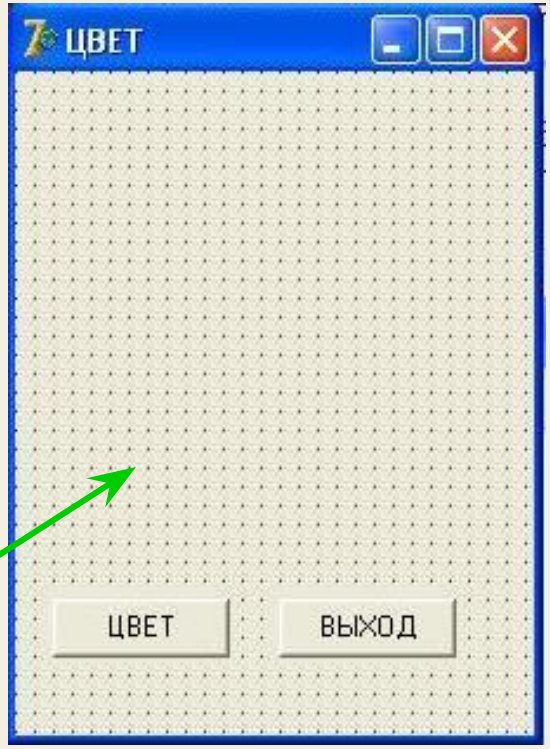
1. Запустим Delphi (Пуск – Все программы – Borland Delphi – Delphi 7).
   Автоматически при запуске Delphi создается новый проект – Project 1

2. Разберем сразу на примере:

Пусть надо создать программу, которая по нажатию кнопки случайным образом изменяла бы цвет нашей формы.

Для этого поместим на форму 2 кнопки: первую назовем «ЦВЕТ» а вторую «ВЫХОД», зададим нужные размеры формы, уцепившись за ее границы и перетащив их в нужное положение, заголовок формы назовем «ЦВЕТ» (свойство Caption)

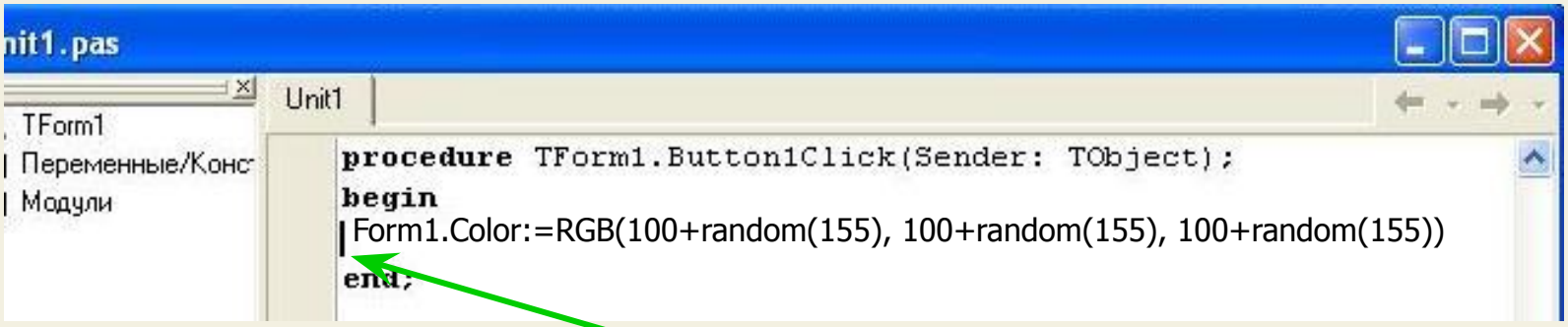
У нас получится примерно так:



3. Сейчас нам нужно, чтобы при нажатии кнопки «ЦВЕТ»(событии) цвет формы менялся произвольным образом. Из Паскаля мы знаем, что для генерации случайных чисел служит функция **random**, а для задания цвета экрана (и формы) применяется модель **RGB**

4. Чтобы перейти к окну редактирования кода и написать процедуру реакции кнопки «ЦВЕТ» сделаем по ней двойной щелчок и мы оказываемся в редакторе кода

Что мы там видим?



5. Мы видим, что Delphi автоматически создала процедуру обработки события нажатия на кнопку (Button1.Click)

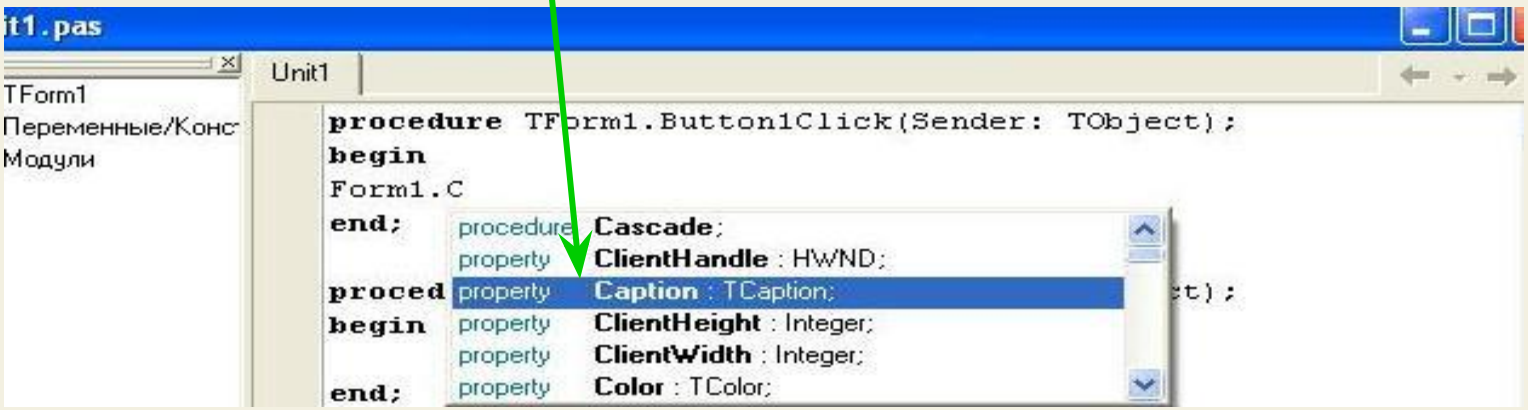
Между **begin** и **end** нам надо вставить код генерации цвета формы:

Form1.Color:=RGB(100+random(155), 100+random(155), 100+random(155))

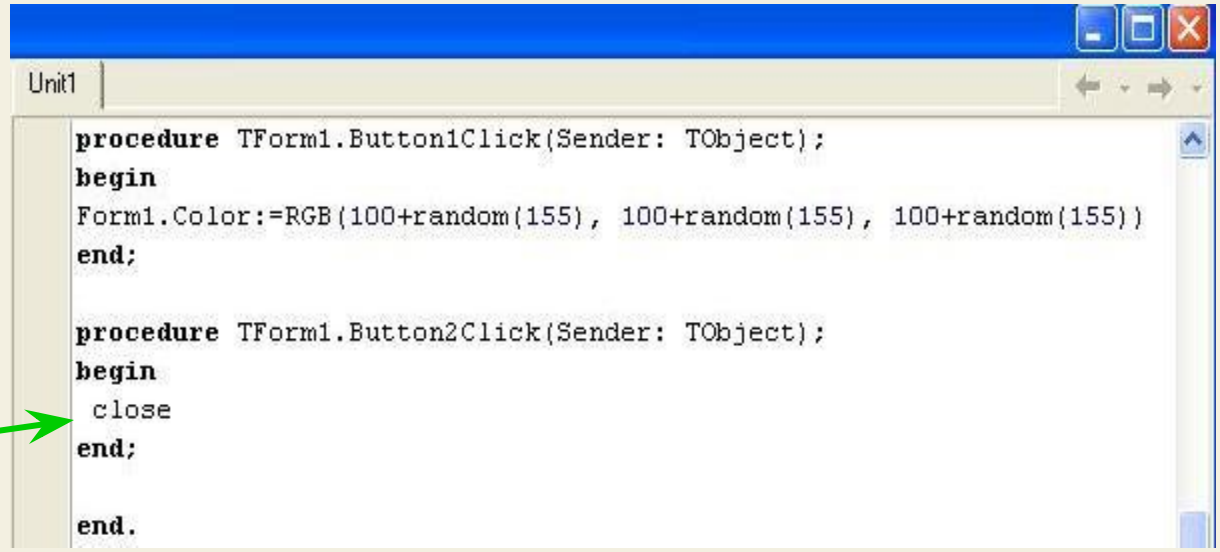
(Свойству формы – Color(цвет) присваивается произвольное значение, причем каждому цвету (красному, зеленому и синему) – это нам известно из Паскаля)

Причем Delphi помогает нам при вводе кода:

Как только мы напечатали Form1. и поставили точку, выходит окно с набором свойств формы, где мы можем выбрать нужное свойство **Caption** и нажать **Enter**



6. Сейчас напишем код для обработки нажатия мышкой по второй кнопке **«ВЫХОД».** Для этого делаем двойной щелчок по кнопке «ВЫХОД» и мы опять в редакторе кода, где вводим команду **close** (закрытие приложения)

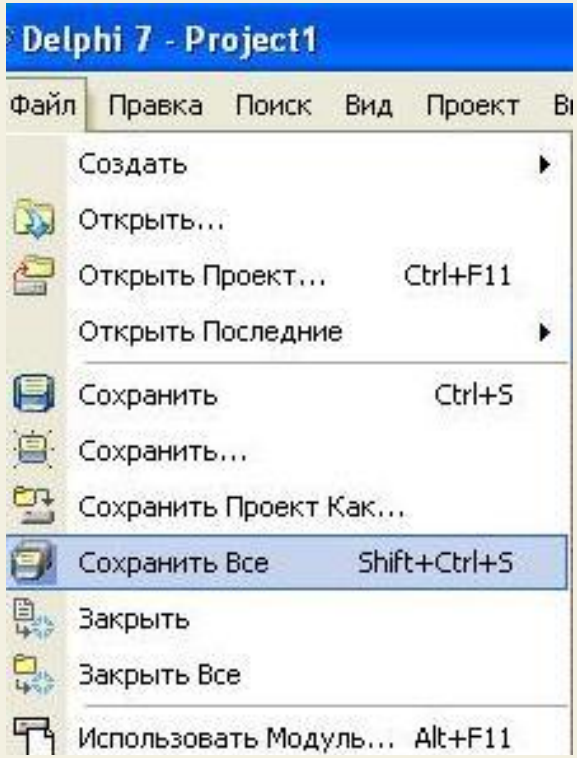


Итак, мы создали процедуры обработки событий нажатия на каждую кнопку, попробуем запустить программу, нажав F9

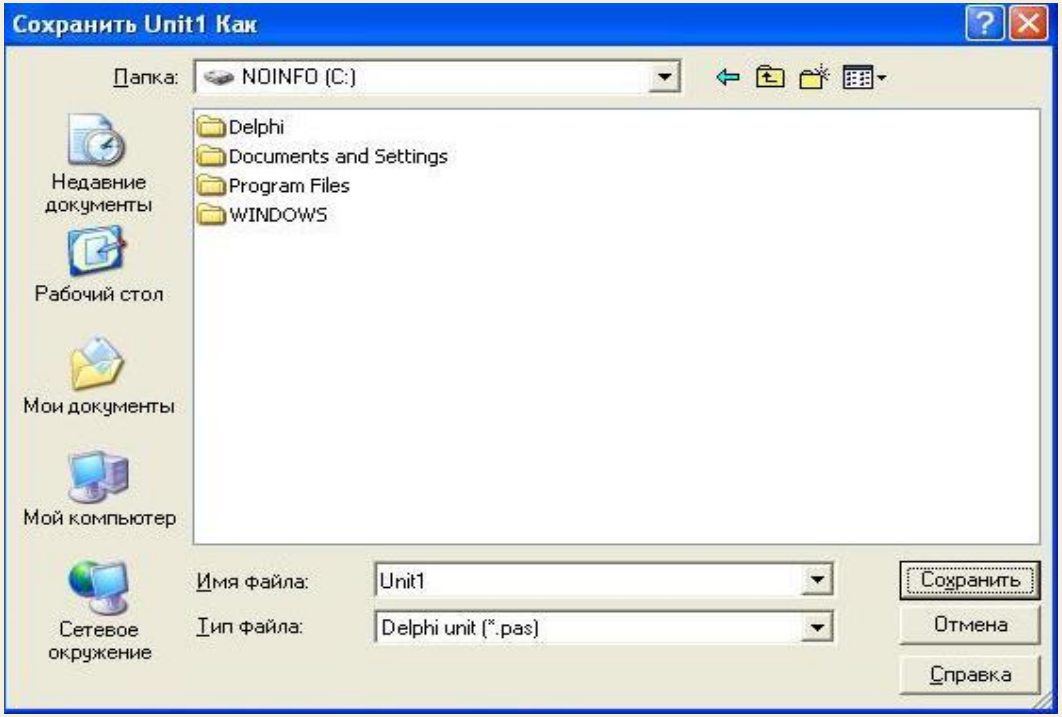
Сейчас при нажатии несколько раз кнопки «ЦВЕТ» форма красится произвольными цветами, а при нажатии кнопки «ВЫХОД» форма закрывается.

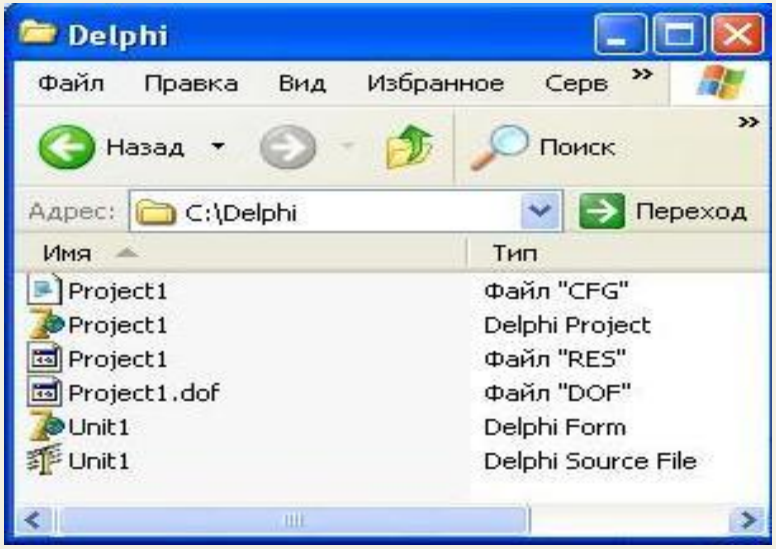
Посмотреть ->

7. После того, как мы проверили работу программы, надо ее сохранить. Для этого заходим в меню Delphi: **Файл – Сохранить Все**



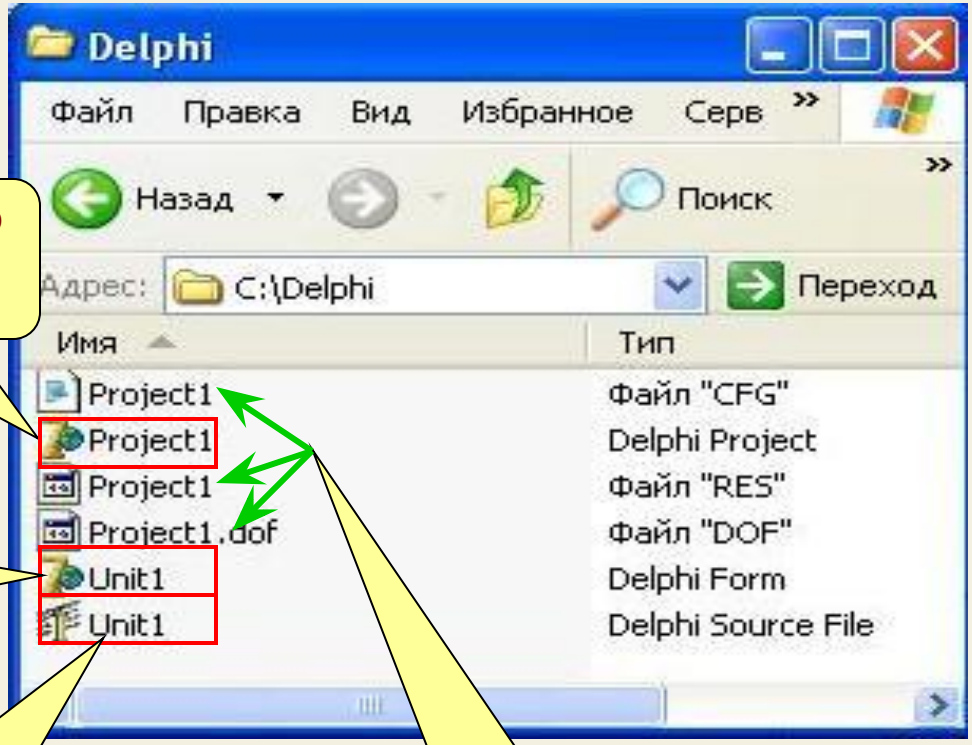
8. Выходит окно сохранения файла, по умолчанию Delphi сохраняет проект в своем каталоге, но для удобства создадим , например, на диске С каталог Delphi, откроем его и сохраним наш проект там





9. При сохранении файлов проекта оказалось, что подтверждать сохранение приходится 2 раза, а сохраненных файлов оказалось 6 штук

Какие файлы у нас сохранились?



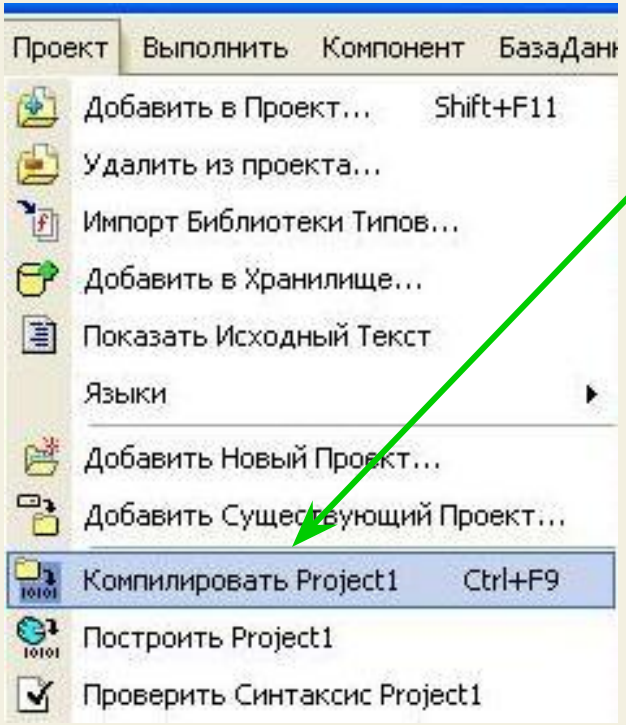
Главный файл нашего проекта

Файл нашей формы с описанием ее свойств

Файл модуля с расширением *.pas – здесь исходный код нашей программы

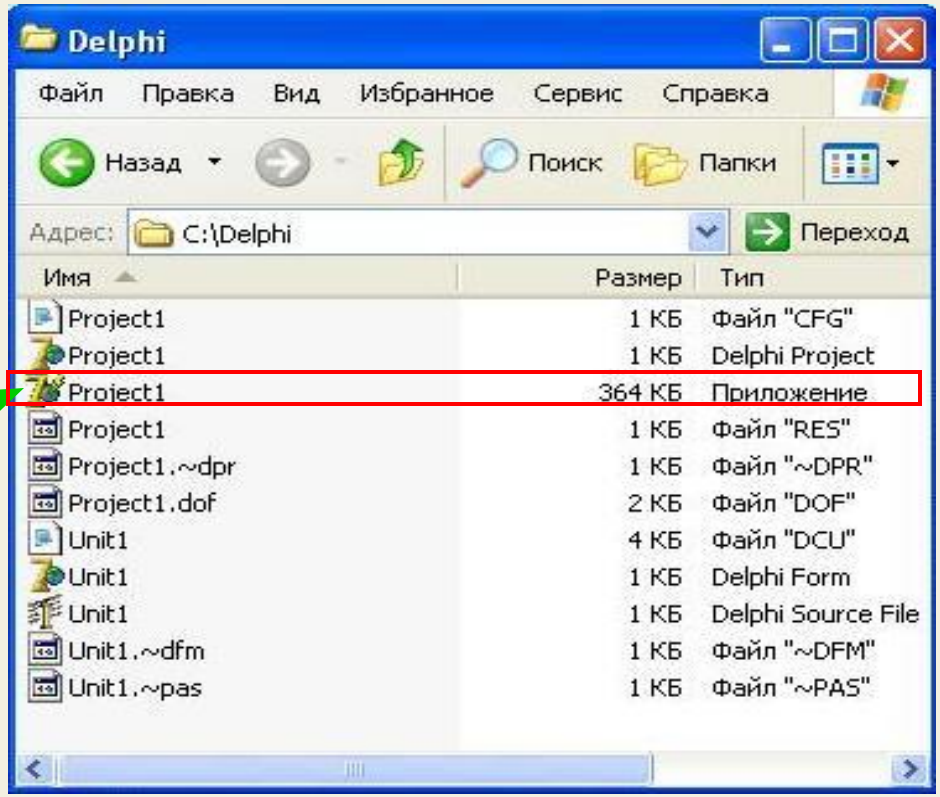
Дополнительные файлы ресурсов, которые Delphi создает автоматически

10. Осталось откомпилировать наш проект, т.е. создать программу, которая будет работать на любом компьютере без Delphi (т.е. EXE – файл или исполнимый файл)



Для этого откроем в Delphi наш проект (Project 1) и в меню Проект выберем «компилировать Project 1

В результате в нашей папке C:\Delphi появляется EXE – файл - это и есть готовая программа, которая выполняется на любом компьютере



Готовое приложение (EXE – файл)

Borland®

2. Разбираемся с исходным кодом

Unit1

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Control
  Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
Form1.Color:=RGB(100+random(155), 100+random(155), 100+random(155))
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
 close
end;
```

А сейчас давайте разберемся с исходным кодом программы, который в большей части создала Delphi – мы добавили совсем немного кода для кнопок

Откроем Delphi, а в ней наш проект: Файл – открыть …

Для перехода в окно редактора кода нажмем F12

В результате мы видим, что исходный код программы достаточно большой, и наш вклад – только две строчки

Unit1

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Control
  Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
Form1.Color:=RGB(100+random(155), 100+random(155), 100+random(155))
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
 close
end;
```

Итак, по порядку:

Это заголовок модуля проекта – **Unit 1**

Интерфейсная часть (объявление всех объектов модуля – типов, переменных …)

Unit1

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Control
  Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
Form1.Color:=RGB(100+random(155), 100+random(155), 100+random(155))
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
 close
end;
```

Итак, по порядку:

Это автоматически подключаемые Delphi модули

Здесь мы видим объявление объектов, которые используются в нашем проекте: Это наша форма (TForm1), Кнопки «ЦВЕТ» и «ВЫХОД» - соответственно Button1 и Button2, а также процедуры обработки событий нажатия на эти кнопки: Button1.Click и Button2.Click

Unit1

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Control
  Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
Form1.Color:=RGB(100+random(155), 100+random(155), 100+random(155))
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
 close
end;
```

Итак, по порядку:

Это **закрытый раздел** класса
Сюда могут помещаться объявления переменных, функций и процедур, включаемых в класс формы, но не доступных для других модулей

Открытый раздел класса
Сюда могут помещаться объявления переменных, функций и процедур, включаемых в класс формы и доступных для других модулей

Сюда могут помещаться объявления типов, констант, переменных, функций и процедур, к которым будет доступ из других модулей, но которые не включаются в класс формы

Unit1

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Control
  Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
Form1.Color:=RGB(100+random(155), 100+random(155), 100+random(155));
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
 close
end;
```

Итак, по порядку:

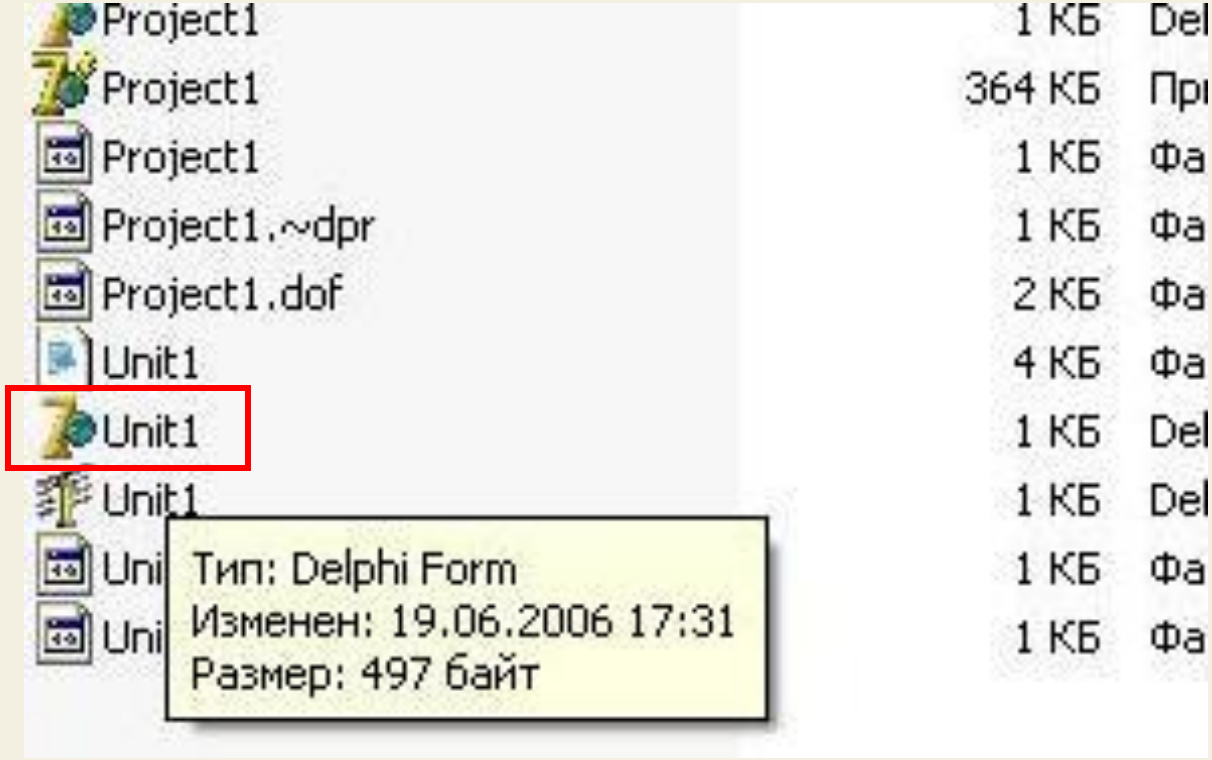
Исполняемая часть модуля, содержащая основной код и отражающая логику и алгоритм работы программы

Процедура обработки первой кнопки («ЦВЕТ»)

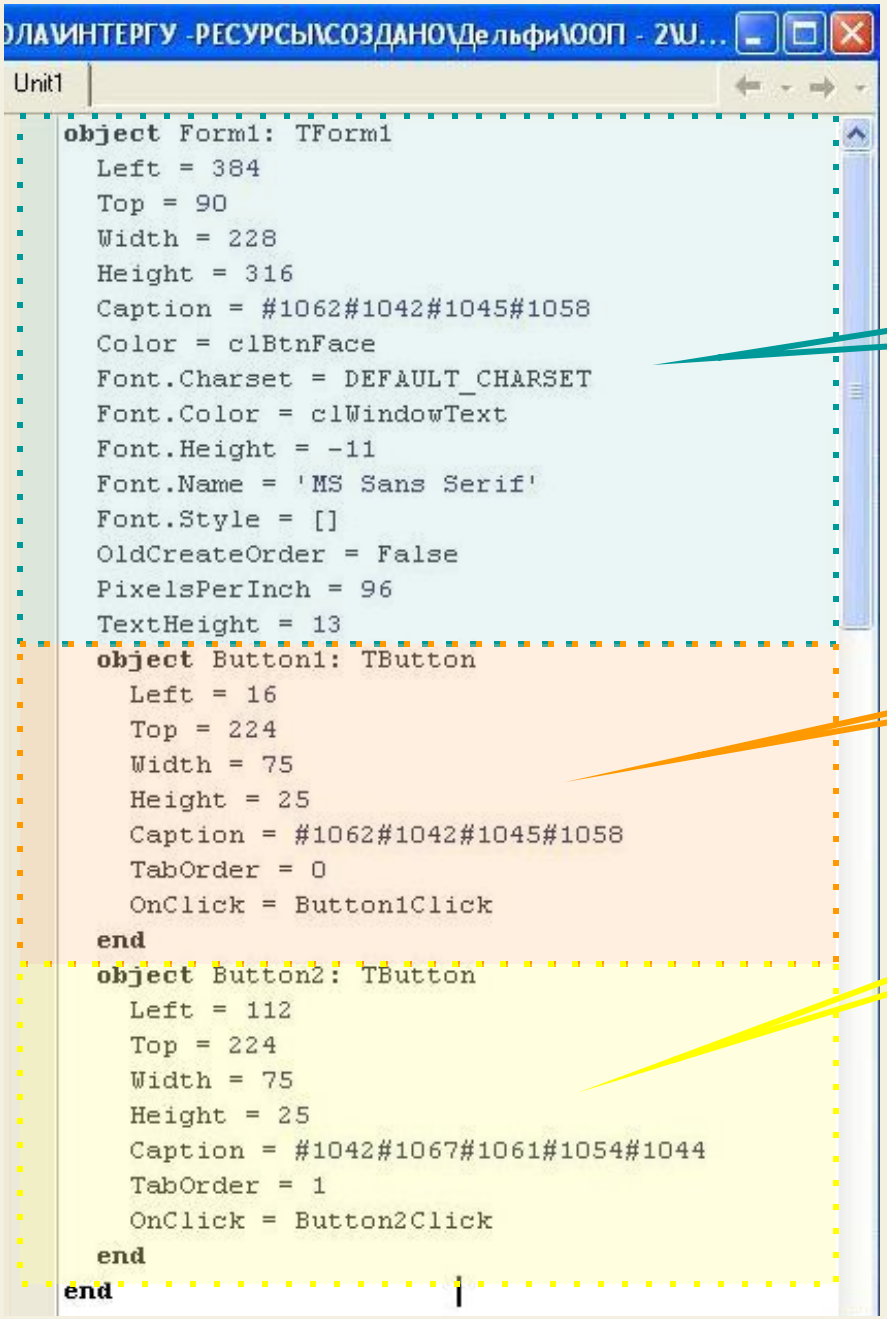
Процедура обработки второй кнопки («ВЫХОД»)

Итак, мы поняли какой код содержит модуль проекта (файл с расширением *.pas)

Сейчас давайте посмотрим еще один файл – файл формы



Откроем его с помощью Delphi и внимательно посмотрим код



В коде мы видим описание формы и объектов, находящихся на ней

Описание формы и ее свойств

Описание свойств первой кнопки

Описание свойств второй кнопки

Самостоятельно разберитесь, что означают строки кода для формы и кнопок

ИТОГИ УРОКА:

На этом уроке мы научились создавать программу, сохранять и компилировать ее, а также познакомились с файлами проекта и научились читать код модуля

НА СЛЕДУЮЩЕМ УРОКЕ:

ООП на Delphi – 3:

Мы научимся программно изменять свойства объектов на практических примерах

Домнин Константин Михайлович
E – mail: kdomnin@list.ru
2006 год.