Проектирование интерфейса программ.

Лекция №6 по курсу
«Информатика»

1. Система программирования Delphi. Основы визуального программирования

Delphi –это система программирования, относящаяся к классу инструментальных средств ускоренной разработки программ. (Rapid Application Development – RAD).

Это ускорение достигается за счет таких свойств Delphi как

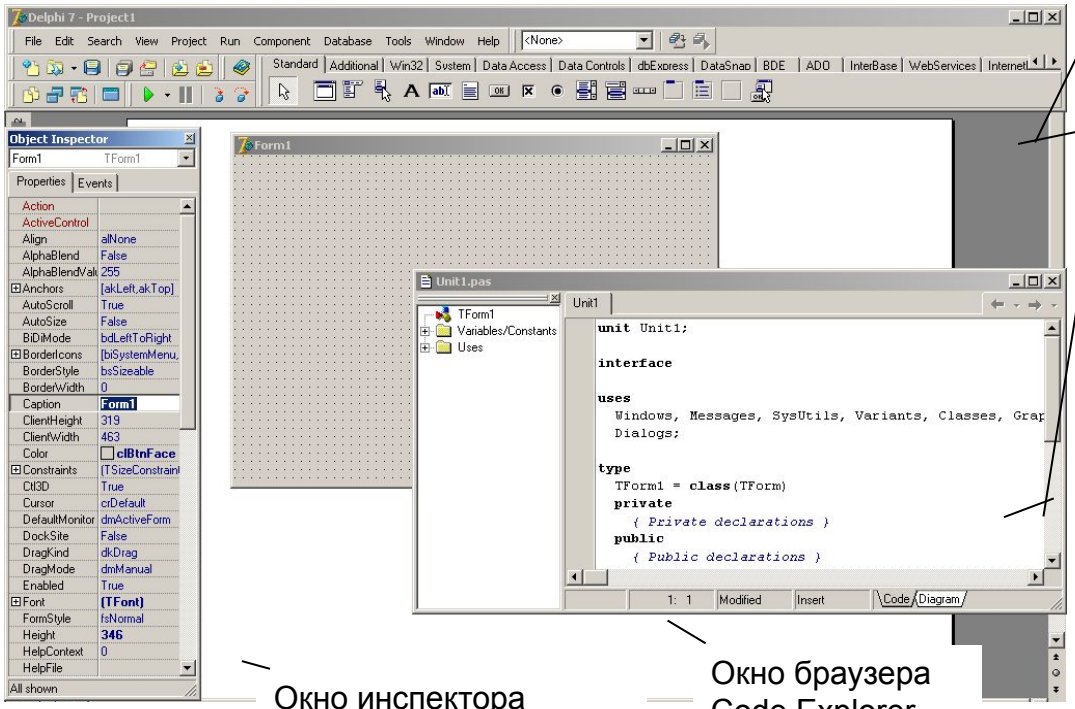
- Визуальное конструирование форм
- Использование библиотеки визуальных компонентов.

Создание программ производится в интегрированной среде разработки (**ИСР**), которая предоставляет программисту готовые формы, на которые с помощью мыши помещаются значки компонентов, имеющихся в библиотеке Delphi. С помощью простых манипуляций можно изменять свойства этих компонентов. При этом сразу виден результат, причем редактор кода Delphi сразу автоматически генерирует код программы.

2. Интегрированная среда разработки (**ИСР**)

ИСР – это среда, в которой есть все необходимое для проектирования, запуска и тестирования создаваемых приложений.

После запуска Delphi на экране компьютера появляются 5 основных окон ИСР. (Вид ИСР зависит от настроек и может меняться)



Главное окно

Палитра компонентов

Окно формы

Окно кода программы

Окно браузера Code Explorer

Окно инспектора объектов Object Inspector

Если окна нет на экране, то его можно вывести на экран с помощью соответствующих команд меню *View*.

Главное окно содержит заголовок, строку меню, панель инструментов и палитру компонентов.

Все визуальные компоненты разделяются на группы, каждая из которых размещается на отдельной странице в палитре компонентов. Каждая страница имеет свое имя.

Для размещения ВК на форме нужно выполнить щелчок по значку компонента в палитре, после чего щелкнуть в нужном месте формы.

ИСР Delphi позволяет одновременно работать только с одним проектом.

3. Общая организация программы в Delphi.

Любая программа содержит следующие файлы, связанные с ней.

- Главный файл проекта, изначально называется PROJECT1.**DPR**.
- Первый модуль программы (**unit)**, который автоматически появляется в начале работы. Файл называется UNIT1.**PAS** по умолчанию, но его можно назвать любым другим именем.

- Файл главной формы, который по умолчанию называется UNIT1.DFM, используется для сохранения информации о внешнем виде главной формы.

- Файл ресурсов PROJECT1.**RES**, создается автоматически и нужен для компиляции.

- Файл, который называется PROJECT1.**CFG** по умолчанию, используется для сохранения установок (опций), связанных с данным проектом. Например, здесь сохраняются директивы компилятора.

- Файл PROJECT1.**EXE** –исполняемый файл, создается после компиляции.

Если сохранить проект под другим именем, то изменят название и файлы с расширением RES, CFG, EXE.
Структура программы создается автоматически и изменять ее не рекомендуется.

Для приложения включающего в свой состав две формы типичный файл проекта имеет следующий вид:

```
program Project1;
uses  {подключение модулей}
  Forms,
  Unit1 in 'UNIT1.PAS' {Form1},
  Unit2 in 'UNIT2.PAS' {Form2};
{$R *.RES}
Begin
  Application.Initialize;
  Application.CreateForm(TForm1,
Form1);
  Application.CreateForm(TForm2,
Form2);
  Application.Run;
end.
```

Файл модуля формы создается автоматически при добавлении новой формы.

По умолчанию к проекту добавляется новая форма, не содержащая компонентов. Текст модуля формы, не содержащей компонентов, приведен ниже.

```
unit Unit2;
interface
uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs;
type
  TForm2 = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
var
  Form2: TForm2;
implementation
{$R *.DFM}
end.
```

4. Технология разработки программ.

Процесс разработки приложения состоит из двух этапов:

I. Создание интерфейса приложения (конструирование формы).

II. Разработка и реализация алгоритма решения задачи

Графический интерфейс – интерфейс, основанный на средствах машинной графики.

Он предполагает взаимодействие человека с компьютером в форме диалога с использованием ввода-вывода на экран графической информации, управления программами с помощью кнопок, меню, окон, экранных панелей и др. элементов управления и т.п.

На **I** этапе нужно разработать вид всех окон приложения, определить их иерархию, а затем в ИСР создать нужное количество форм, разместить на них все необходимые компоненты и установить их свойства с помощью Инспектора объектов.

На **II** этапе нужно разработать алгоритмы всех процедур, написать соответствующие тексты процедур на языке программирования и установить, в результате наступления каких событий нужно будет выполнить созданные процедуры. Затем в ИСР создать обработчики событий – процедуры, которые будут выполнены при наступлении того или иного события.

События наступают в результате действий пользователя – перемещения курсора мыши, нажатия кнопок мыши и т.д., а также в результате работы объектов, например, событие появления окна.

5. Визуальные компоненты, используемые при создании простых оконных приложений.

Визуальные компоненты – это элементы, из которых конструируется видимое изображение, создаваемое программой.

Каждый ВК принадлежит определенному классу объектов и имеет имя. Имена классов в Delphi начинаются с буквы **T**.

Имена ВК состоят из названия класса без буквы T и порядкового номера компонента данного класса.

Например, TForm – класс всех форм, а Form1 – конкретная форма, экземпляр класса.

- **TForm**

Является контейнером для размещения всех остальных компонентов.

Любая программа имеет хотя бы одну связанную с ней форму, которая наз. Главной. Эта форма появляется на экране в момент старта программы.

Программа может иметь много форм, каждая из которых может появляться на экране по мере надобности.

Свойства:

Caption Определяет заголовок формы

Position Определяет положение и размер формы в момент ее появления на экране

Color Определяет цвет формы

TLabel 

(Страница Standard палитры компонентов)

Компоненты этого класса предназначены для размещения на форме текстовых надписей.

Свойства:

Caption центральное свойство, к-е содержит текст надписи.

Font определяет параметры шрифта

TEdit 

(Страница Standard палитры компонентов)

Компонент этого класса представляет собой однострочное редактируемое текстовое поле. Используется для ввода или вывода текстовых строк.

Свойства:

Text содержит отображаемую компонентом строку

ReadOnly Определяет, можно ли редактировать текст (true –нельзя, false – можно).

Font определяет параметры шрифта

TButton

(Страница Standard палитры компонентов)

Кнопка. Предназначена для управления программой.

Обычно используется для создания процедур, реализующих алгоритм решения задачи (обработчиков событий).

Свойства:

Caption Определяет текст, отображаемый на кнопке

Font определяет параметры шрифта

Hint определяет всплывающую подсказку

TBitBtn

(Страница Additional)

Кнопка, на которой может располагаться рисунок.

Свойства:

Glyph Определяет изображение на поверхности кнопки

Kind Определяет одну из 11 станд. разновидностей кнопки

TMemo

(Страница Standard)

Многострочное редактируемое текстовое поле. Используется для ввода, вывода и редактирования достаточно длинного текста.

Свойства:

Lines содержит текст, представляющий пронумерованный набор строк (нумерация начинается с нуля).

ReadOnly Определяет, можно ли редактировать текст (true –нельзя, false – можно).

Свойства ВК можно устанавливать на этапе конструирования формы с помощью Инспектора объектов на вкладке Properties (свойства),

а также программировать изменение свойств при выполнении программы.

При этом в программе для обращения к свойству используется составное имя:

<имя формы>.<имя компонента>.<имя свойства>

Например,

```
Form1.Label3.Caption:=' Привет, Вася!'
```

Если обращение к свойству ВК происходит в модуле, соответствующем форме, на которой он размещен, то имя формы можно не указывать.

Например,

```
    Label3.Caption:=' Привет, Вася!'
```

6. Создание обработчика события

Каждый компонент помимо свойств характеризуется набором событий, на которые он может реагировать (щелчок мыши по компоненту, перемещение, изменение и др.).

Чтобы создать процедуру для обработки события, нужно выполнить двойной щелчок по ВК или дважды щелкнуть по названию события на вкладке **Events** Инспектора объектов. При этом нужный ВК должен быть выделен.

Например,

при двойном щелчке мышью по кнопке **TButton,** в модуль будет вставлена процедура

```
procedure TForm1.Button1Click(Sender:
TObject);
begin
end;
```

7. Ввод значений переменных с помощью ВК.

Для ввода данных можно использовать ВК **TEdit.**
Для этого нужно

- разместить на форме компонент TEdit;

- в процедуре, реализующей алгоритм решения задачи, переменной, значение которой требуется ввести, присвоить значение свойства Text соответствующего компонента.

Если переменная вещественного или целого типов, то значение свойства Text должно быть преобразовано с помощью соответствующей функции.

`StrToInt(St)` преобразует символы строки St в целое число.

`StrToFloat(St)` преобразует символы строки St в вещественное число.

Например,пусть имеется след. описание переменных:

```
Var x:real; k:integer;
```

Тогда для ввода значений переменных x и k в программе пишут след. операторы присваивания:

```
x:=StrToFloat(Edit1.Text);

k:=StrToInt(Edit2.Text);
```

8. Вывод значений переменных с помощью ВК.

- С помощью TEdit:

  - разместить на форме компонент TEdit;
  - в процедуре, реализующей алгоритм решения задачи, свойству Text соответствующего компонента присвоить значение переменной, которую требуется вывести.

Если переменная вещественного или целого типов, то ее значение должно быть преобразовано с помощью соответствующей функции или процедуры.

Функции

`IntToStr(x)` преобразует целое число **x** в строку символов.

`FloatToStr(x)` преобразует вещественное число **x** в строку символов.

Процедура

`Str(x:n1:n2,St)` преобразует вещественное или целое число **x** в строку символов St с форматом.

n1 – общая ширина поля, выделенного под значение **x,**

n2 – количество символов в дробной части (только для вещественных чисел).

Например,пусть имеется след. описание переменных:

```
Var x:real; k:integer;
```

Тогда для вывода значений переменных x и k в программе пишут след. операторы присваивания:

```
Edit1.Text:= FloatToStr(x);
Edit2.Text:= IntToStr(k);
```

Или

```
Str(x:5:2,St); Edit1.Text:=St;
Str(k,St);     Edit2.Text:=St;
```

Переменная St должна быть описана как строковая:

```
St:string;
```

- С помощью **TLabel**

- разместить на форме компонент **TLabel**;

- в процедуре, реализующей алгоритм решения задачи, свойству Caption соответствующего компонента присвоить строку символов, содержащую значения выводимых переменных, преобразованных к строковому типу, и все необходимые комментарии.

Например,

```
Label1.Caption:='x='+  FloatToStr(x)+'  k='+
IntToStr(k);
```

Или

```
Label1.Caption:='x='+ FloatToStr(x);
Label2.Caption:=' k='+ IntToStr(k);
```

Или

```
St:= 'x='+ FloatToStr(x)+' k='+ IntToStr(k);
Label1.Caption:=St;
```

9. Пример простого приложения с графическим интерфейсом

Требуется разработать программу вычисления функции .

$$f = 2\frac{\cos^3 x + 2y}{2{,}5}$$

Порядок решения задачи:

1) разработать проект формы:

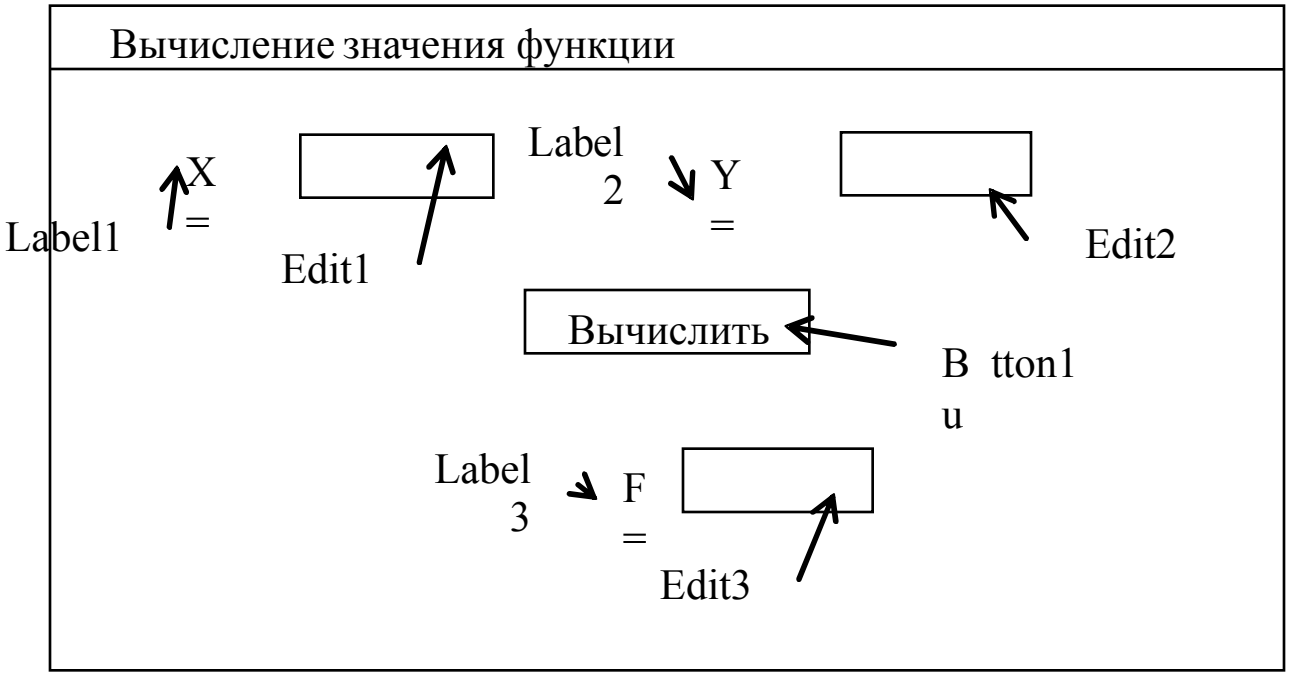

Вычисление значения функции
Label1
X =
Edit1
Label 2
Y =
Edit2
Вычислить
B tton1
u
Label 3
F =
Edit3

Таблица свойств компонентов:

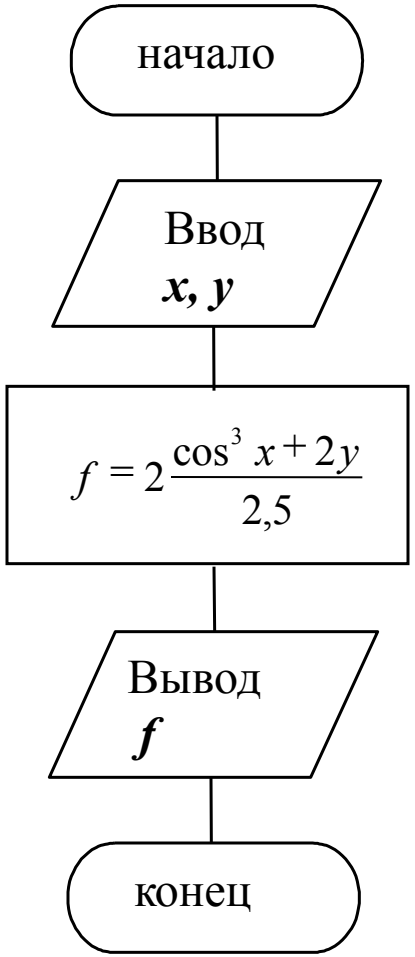
Элемент интерфейса	Компонент	Свойства	Значение свойства
Главное окно программы	Form 1	Caption (заголовок окна) Position (расположение на экране)	Вычисление значения функции poScreenCenter
Комментарий «X=»	Label 1	Caption	X=

Комментарий «Y=»	Label 2	Caption	Y=
Комментарий «F=»	Label 3	Caption	F=

Поле ввода значения X	Edit1	Text	
Поле ввода значения Y	Edit2	Text	
Поле вывода значения F	Edit3	Text ReadOnly	 True
Кнопка «Вычислить»	Button1	Caption Hint ShowHint	Вычислить Вычисление значения функции True

2) разместить все ВК на форме и установить их свойства с помощью Инспектора объектов.

3) разработать алгоритм решения задачи:


начало
Ввод
x, y
$f = 2\frac{\cos^3 x + 2y}{2,5}$
Вывод
f
конец

4) создать процедуру, которая будет выполняться при нажатии кнопки. Для этого нужно выполнить двойной щелчок мышью по кнопке.

5) В окне кода набрать текст процедуры, реализующий составленный алгоритм:

```
procedure TForm1.Button1Click(Sender: TObject);

 Var
    x,y,f:real;


 begin

   x:=StrToFloat(Edit1.Text);

   y:=StrToFloat(Edit2.Text);

   f:=2*(sqr(cos(x))*cos(x)+2*y)/2.5;

   Edit3.Text:= FloatToStr(f);

 end;
```

6) Сохранить полученную программу командой Save All **в отдельной папке**

7) Запустить программу на выполнение командой Run или нажатием клавиши F9 и проверить работу программы с помощью заранее подготовленных тестов.