Уроки № 1, 2.
Тема:
Идеи объектно-ориентированного программирования. Работа в среде программирования Delphi 7. Запуск программ на выполнение. Создание простого проекта.

Цель уроков:

- Познакомить учеников с новым направлением в программировании – с визуальным программированием, со средой программирования DELPHI 7, некоторыми визуальными компонентами и их свойствами, порядком действий при создании и сохранении проекта.
- Кроме этого, ученики смогут получить первые практические навыки работы в среде программирования DELPHI 7 – загружать с диска и запускать на выполнение готовый проект и создать новый проект по предложенному образцу.

План урока:

1. Актуализация опорных знаний – На протяжении урока по мере возникновения вопросов.
2. Идеи объектно-ориентированного подхода.
3. Среда программирования DELPHI 7.
   - Загрузка
   - Вид экрана
4. Форма приложения, ее основные свойства.
5. Компоненты Label, Edit, Button и их свойства.
6. Событие и процедура обработки события.
7. Некоторые события в DELPHI 7.
8. Программирование командных кнопок.
9. Особенности ввода-вывода в Delphi:
10. Сохранение проекта.
11. Практическая работа № 1 «Знакомство со средой программирования DELPHI 7, запуск программ на выполнение. Создание простого проекта»
12. Выставление оценок за пр. р.
13. Домашнее задание: выучить раздел 2., 1 – 3. стр. 160-170.
14. Итоги урока.

Ход урока:

2 Идеи объектно-ориентированного подхода.

В последние десятилетия в программировании возник и получил широкое развитие объектно-ориентированный подход. Это метод программирования, имитирующий реальную картину мира: информация, используемая для решения задачи, представляется в виде множества взаимодействующих объектов. Каждый из объектов имеет свои свойства и способы поведения. Взаимодействие объектов осуществляется при помощи передачи сообщений: каждый объект может получать сообщения от других объектов, запоминать информацию и обрабатывать ее определенным образом и, в свою очередь, посылать сообщения.

Объектно-ориентированная идеология используется практически во всех современных программных продуктах, включая ОС.

Первый объектно-ориентированный язык программирования Simula-67 был создан как средство моделирования работы различных приборов и механизмов.

В настоящее время широко используются системы визуального программирования: DELPHI, VISUAL BASIC, VISUAL C++.

Популярность этих ЯП объясняется тем, что они делают сложную технологию работы с визуальными компонентами WINDOWS доступной широкому кругу пользователей путем использования графического интерфейса.

3 Среда программирования DELPHI 7.

Delphi – среда разработки программ, относящаяся к так называемым **RAD**-системам **(Rapid Application Development** – среда быстрой разработки приложений), суть которых состоит в том, что среда разработки берет на себя большую часть рутинной работы по созданию и настройке стандартизированных элементов интерфейса разрабатываемого проекта – кнопок, полей ввода данных, списков, переключателей и др.

Слово **Delphi** – это название города в древней Греции, в котором пророчествовали оракулы.

Версия **Delphi 1** была выпущена в 1995 году.

Программирование в среде Delphi7 существенно отличается от программирования в процедурных ЯП.

Основные принципы, которые отличают этот ЯП от процедурных:

- Отделение объектов программы, которые связаны с интерфейсом пользователя, от их программной части.
- Скорость и простота создания и настройки интерфейса пользователя, в котором используются готовые блоки – кнопки ,поля ввода, переключатели и т.д.

Программист может сосредоточится на разработке экранной формы (окна приложения) и программировании процедур и функций обработки событий – щелчка мыши или нажатия клавиши.

В качестве основы ЯП Delphi использует язык Object Pascal.

Запуск:

Пуск ⇨ Программы ⇨ Borland Delphi7 ⇨ Delphi7

Вид экрана:

5 окон

1. Главное окно Delphi7
   a. Строка заголовка,
   b. Строка меню,
   c. Панели инструментов,
   d. Палитра компонентов.
2. **Окно стартовой формы** – заготовка главного окна разрабатываемого приложения.
3. Окно редактора свойств объектов – **Object Inspector** – предназначено для редактирования свойств объектов.
4. Окно просмотра списка объектов **– Object Tree View.**
5. Окно редактора кода программы – в нем набирают текст программы.

4 Форма приложения, ее основные свойства.

Работа над новым проектом начинается с создания стартовой формы. Она создается путем изменения значений свойств формы **Form1** и добавления к форме необходимых компонентов – полей ввода и вывода данных, кнопок и пр.

Свойства формы определяют ее внешний вид: размер, положение на экране, текст заголовка, вид рамки.

Для просмотра и изменения значений свойств формы и ее компонентов используется окно **Инспектора объектов** Object Inspector. В верхней части окна указано имя выделенного объекта, в левой колонке вкладки **Properties** (свойства) – перечислены свойства этого объекта, а в правой – значения этих свойств.

Основные свойства формы.

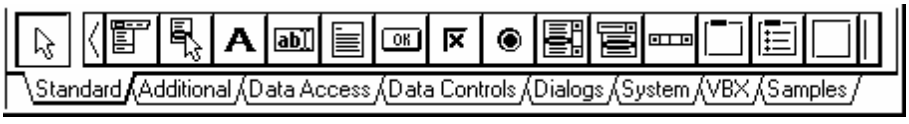
Свойство	Описание
Name	Имя формы. Используется в программе для управления формой и доступа к ее компонентам. Стандарное имя Form1, Form2…
Caption	Текст заголовка окна.
Width	Ширина формы в пикселях.
Height	Высота формы.
Top	Расстояние от верхней границы формы до верхней границы экрана.
Left	Расстояние от левой границы формы до левой границы экрана.
BorderStyle	Вид границы BsSizeable – обычная. Во время работы программы можно изменять размер окна. BsSingle – тонкая. Нельзя изменить размер окна. BsNone – нет граныцы. Окно выводится без заголовка и его положение и размер изменить нельзя.
Color	Цвет фона.
Font	Шрифт

Настройка свойств объектов может проводится по-разному: для некоторых свойств нужно ввести значения с клавиатуры **(Caption)**, свойства **Width, Height, Top, Left** можно вводить с клавиатуры, а можно установить, изменяя размеры и местоположение самой формы, значения свойств **BorderStyle** и **Color** выбираются из списка, если в строке значения свойств есть кнопка с тремя точками, то нажатие на ней открывает соответствующее диалоговое окно **(Font).**

5 Компоненты Label, Edit, Button и их свойства.

В палитре компонентов можно выбрать практически любой компонент, который мы можем встрелить в диалоговых окнах Windows и других программ. Число таких элементов в настоящее время довольно велико и продолжает расти, причем внутренняя структура многих из этих элементов довольно сложная. Однако базовые элементы интерфейса уже вряд ли будут принципиально меняться. Их состав, свойства, принципы использования являются практически промышленным стандартом и одинаковы в любой среде разработки современных программ. Рассмотрим создание стандартных элементов интерфейса на примере работы с компонентами библиотеки VCL (Visual Component Library) в среде Delphi.

Базовые стандартные элементы расположены на странице Standart палитры компонент Delphi. Все эти объекты такая же часть Windows, как мышь или окно.



Сегодня мы рассмотрим следующий набор компонент:.

Button – стандартная кнопка, обычно кнопка используется для запуска действия, которое должно произойти после нажатия на эту кнопку. Свойства **Color** для оформления надписи (**Caption**) у кнопки нет.

Label – метка, используется как надпись или как область вывода информации для чтения. Свойство **AutoSize=True** определит минимизацию размера метки под текст надписи, **Aligment** – центровку этого текста, **WordWrap** – возможность расположения текста в несколько строк, **Transparent** – прозрачность при наложении на другие элементы.

Edit – строка ввода. Заголовка (**Caption**) у этого компонента **нет**, но есть свойство **Text** как содержимое строки. Это свойство можно как считывать, так и записывать.

6 Событие и процедура обработки события.

Завершив работу по зозданию формы приложения, можно приступать к написанию текста программы. Но перед этим обсудим очень важные при программировании в WINDOWS понятия.

- **Событие**
- **Процедура обработки события**

Щелчок на изображении командной кнопки – это пример того, что в WINDOWS называется событием.

Событие **(Event)** – это то, что происходит во время работы программы. В Delphi каждому событию присвоено имя.

7 Некоторые события Delphi.

Событие	Когда происходит
OnClick	При щелчке кнопкой мыши.
OnDblClick	При двойном щелчке кнопкой мыши.
OnKeyPress	При нажатии клавиши клавиатуры
OnCreate	При создании объекта (формы).
OnMauseDown	При нажатии кнопки мыши.
OnMauseUp	При отпускании кнопки мыши.

Реакцией на событие должно быть какое-нибудь действие.

В Delphi реакция на событие реализуется как **процедура обработки события**. Таким образом, для того чтобы программа выполняла некоторую работу в ответ на действия пользователя, программист должен написать процедуру обработки соответствующего события.

8 Программирование командных кнопок.

Чтобы приступить к написанию процедуры, нужно на форме выбрать объект для которого мы собираемся писать процедуру. Затем в окне Инспектора объектов выбрать вкладку **Events**. В левой колонке перечислены события, предусмотренные для этого объекта. Если для этого объекта уже написаны процедуры обработки событий, то в правой части рядом с именем события выводится имя этой процедуры.

Для создания процедуры нужно дважды щелкнуть в поле имени процедуры. Откроется окно редактора кода, в котором уже добавлен шаблон процедуры обработки этого события:

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

end;
```

Имя процедуры состоит из: TForm1 – имя формы, Button1 – имя объекта, Click –имя события.

В этот шаблон программист должен вписать код программы по правилам ЯП Object Pascal. Для ускорения ввода программного кода можно воспользоваться шаблонами. Они вызываются нажатием <Ctrl+ J>.

9 Особенности ввода-вывода в Delphi:

Рассмотрим текст программы из практической работы. Данные в программу поступают из полей редактирования **Edit1** и **Edit2** путем обращения к их свойствам **Text**, в которые во время работы программы введет пользователь. Для правильной работы программы строка должна содержать только цифры. Вывод информации осуществляется в компонент **Label4** в его свойство **Caption**. Свойства **Text** и **Caption** могут содержать информацию только символьного типа, поэтому необходимо использовать стандартные функции преобразования.

Для преобразования строки в числа в программе используются стандартные функции **StrToInt** и **StrToFloat**

```
// Получить исходные. данные из полей ввода
dist:= StrToInt(Edit1.Text);
t:= StrToFloat(Edit2.Text);
```

Для преобразования числа в строку используются стандартные функции
IntToStr и **FloatToStr**

```
// Вывод результата
Label4.Caption:= 'Дистанция: ' + Edit1.Text + ' м' + #13 +
  'Время: ' + IntToStr(min) + ' мин ' + IntToStr(sek) + ' сек ' + #13 +
  'Скорость: ' + FloatToStrF(v, ffFixed,4,2) + 'км/час';
end;
```

Строка вывода формируется путем присоединения строк – конкатенации. Символ **#13** обозначает перевод на новую строку.

Процедуры обработки событий мы пишем в файле модуля формы.

10 Сохранение проекта.

File ⇨ Save. Если проект еще ни разу не был сохранен, Delphi предложит сохранить сначала файл главного модуля, а затем файл модуля формы.

Правила сохранения проектов:

1. Для каждого проекта создавать отдельную папку (т.к. каждый проект содержит несколько файлов).
2. Имя файла проекта (.DPR) должно отличаться от имени файла модуля (.PAS). Имена всех остальных файлов Delphi присваивает автоматически.

Создание .EXE файла – компиляция.

Project ⇨Compile

Project ⇨ Build All Projects.

Состав модуля.

```
unit Unit1;
interface
```

раздел интерфейса – содержит описание стандартных модулей и перечень объектов, размещенных на форме и процедур обработки событий

```
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls;

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

var
  Form1: TForm1;

implementation
```

реализации – содержит раздел описания локальных переменных, процедур и функций.

```
{$R *.dfm}
```

инициализации – в нем располагаются тексты процедур

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

end;

end.
```

Структура проекта

Проект – это набор файлов, используя которые компилятор создает исполняемый файл программы(.EXE - файл).

Обычно проект состоит из:

1. Главный файл проекта (.DPR);
2. Файла описания проекта (.DOF);
3. Файла ресурсов (.RES);
4. Файла описания формы (.DFM);
5. Файла конфигурации (.CFG);
6. Файла модуля формы (.PAS).