

Лабораторна робота №1 (семестр 1)

ТЕМА: Знайомство з середовищем RAD (Rapid Application Development – середовище швидкої розробки додатків).

МЕТА: Вивчити основи середовища RAD для розробки програм-додатків.

ЗНАТИ: Мову програмування C або Pascal відповідно до обраної RAD.

ВМІТИ: Інсталювати RAD (наприклад Builder або Delphi).

ТЕОРЕТИЧНІ ВІДОМОСТІ.

Загальні визначення. Сучасні RAD в більшості випадків базуються на концепції об'єктно-орієнтованого програмування (ООП), що дозволяє значно збільшити складність програм і скоротити час їх розробки за рахунок більш ефективного використання повторного коду.

ООП базується на поняттях класу і екземпляру. Клас являє собою об'єднаний набір даних і підпрограм, призначених для обробки цих даних. Наприклад, клас даних «студент» міг би включати набір змінних, що містять інформацію про студента (ім'я та прізвище, група, в якому він навчається, бали з предметів і т.д.) - їх називають властивості класу даних, і набір операцій (їх називають методами), призначених для управління даними класу «студент» (переведення студента в іншу групу, виставлення оцінки тощо).

Загальні принципи ООП:

Інкапсуляція - об'єднання даних і методів їх обробки, поділ доступу до даних між класами.

Спадкування - класи об'єктів можуть успадковувати властивості один одного; приклад - класи «студент» та «викладач» успадковують деякі властивості класу «людина».

Поліморфізм - методи об'єктів можуть перевизначатися. Приклад - класи об'єктів «прямокутник» і «коло» можуть мати різні методи малювання з одним і тим же ім'ям Draw ()

У ООП програма являє собою не просто послідовно виконуваний набір інструкцій, а сукупність підпрограм, що реагують на зовнішні події, такі як натискання клавіші, кнопки миші і т.д.

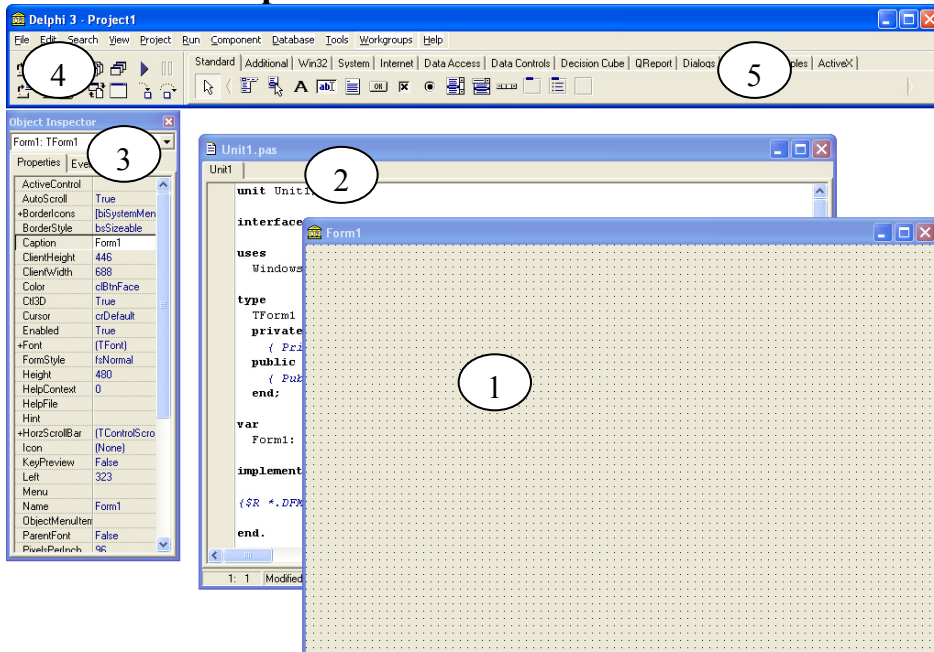
Далі розглянемо основу RAD Delphi 2-8 (dcc32.exe). Опис актуально і в останній версії Embarcadero RAD Studio 2010 (з деякою зміною інтерфейсу і структури).

Типи файлів Delphi:

Текстові: *. DPR - файл проекту, *. PAS - вихідні тексти модулів; *. DOF - опції компілятора

Двійкові: *. DCU - відкомпільований файл модуля; *. DFM - файл форми; *. RES - файл ресурсів програми, *. EXE - файл програми. Таким чином, Delphi створює досить багато файлів і при розробці слід завжди дотримуватися принципу "один проект - одна папка".

Вікно Delphi:



Вікно **форми** - головне вікно нашої майбутньої програми;

Вікно **редактора** (вікно текстів програм) - по одній закладці на кожен програмний модуль;

Вікно **Інспектора Об'єктів** - в цьому вікні ми керуємо властивостями (вкладка "Властивості") і подіями (Events) об'єкта, обраного у вікні форми;

Панель інструментів (SpeedBar);

Палітра компонентів (Component Palette) - готові елементи інтерфейсу для програми (кнопки, перемикачі, поля введення, діалоги тощо).

Меню View - його пункти дозволяють включити і вимкнути окремі вікна Delphi:

Project Manager - включити вікно «склад проекту» (список файлів)

Units - вивести вікно текстів програм

Forms - вивести список форм

Project Source - додати у вікно форми файл проекту *. DPR

Object Inspector - включити інспектор об'єктів

SpeedBar - включити кнопки панелі інструментів

Component Palette - включити палітру компонентів

Файл проекту за умовчанням (Project1.dpr):

```
program Project1;  
uses  
  Forms,  
  Unit1 in 'Unit1.pas' {Form1};  
{$R *.RES}  
begin  
  Application.Initialize;  
  Application.CreateForm(TForm1, Form1);  
  Application.Run;  
end.
```

У простих проектах втручання у файл проекту та його редагування, як правило, не потрібні.

Текст модуля за замовчуванням (Unit1.pas):

```

unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs;
type
  TForm1 = class(TForm)
  { елементи інтерфейсу - записи додаються сюди автоматично }
    Edit1: TEdit;
    Button1: TButton;
  { заголовки підпрограм – додаються сюди автоматично }
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
  { Тут можна описати свої глобальні константи }
var
  Form1: TForm1;
  { Тут можна описати свої глобальні змінні }
implementation
{$R *.DFM}
{ Тут починаються тексти підпрограм модуля }
procedure TForm1.Button1Click(Sender: TObject);
begin
  { Підпрограма-обробник події OnClick створюється автоматично при подвійному
  клацанні по елементу управління у вікні форми - у даному випадку клацання було
  проведено по кнопці Button1 }
end;
end.

```

Таким чином, Delphi автоматично створює процедури обробки подій від елементів управління, і від програміста потрібно лише написати відповідний код. Ім'я **Button1Click** утворено системою від імені інтерфейсного елементу (кнопки) Button1 і назви події OnClick.

Основні властивості форми (вікна програми):

Name - ім'я, заголовок - заголовок вікна, Width, Height - ширина і висота, Font.Name - ім'я основного шрифту, Font.Size - розмір основного шрифту, BorderIcons - вкл. або викл. стандартні кнопки управління вікном, BorderStyle - стиль рамки вікна (значення bsSizeable - розмір можна змінювати, bsDialog - ні), **Position** - позиція вікна при старті програми (poScreenCenter - по екрану центру).

Стандартні компоненти інтерфейсу:





- виділений компонент, властивості якого показані в інспекторі об'єктів.


Загальні властивості: Name - ім'я компоненту, Left, Top - координати лівого верхнього кута компонента щодо клієнтської частини вікна форми, Width, Height - ширина і висота компонента, Font.Name, Font.Size - ім'я і розмір шрифту, Visible - видимість компонента, Enabled - доступність компонента, TabOrder - номер


компонента в послідовності обходу клавішею Tab, Align - вирівнювання компонента (значення alNone, alLeft, alTop, alRight, alBottom, alClient), Color - колір області компонента.


Поки ми будемо використовувати наступні **компоненти**:


 **Label** **Текстова мітка**. Використовується для створення написів на формі. AutoSize - автоматичний розмір, Caption - текст мітки, WordWrap - перенесення за словами, TransParent - прозорість фону мітки;

 **Edit** **Текстове поле редагування**. Використовується для введення рядка даних. Text - введений текст, MaxLength - максимальна довжина тексту (0 - немає обмежень), ReadOnly - тільки для читання, AutoSelect - виділяти текст при переході в полі;

 **Memo** **Багаторядкове текстове поле**. Використовується для редагування великого обсягу даних. Maxlength; ReadOnly; ScrollBars - показувати чи смуги прокрутки; Lines - контейнер для рядків, що містяться в полі. Його основні методи: Lines.Clear - очистити поле; Lines.Add ('рядок') - додати рядок у кінець; Lines.Insert (номер, рядок) - вставити рядок після рядка з зазначеним номером (0 - до початку); Lines.Delete (номер) - видалити рядок після рядка з зазначеним номером (0 - першу); Lines.LoadFromFile ('шлях до файлу') - завантажити рядки з файлу; Lines.SaveToFile ('шлях до файлу') - зберегти рядки в файл;

 **Button** **Кнопка**. Як правило, натискання кнопки запускає яку-небудь процедуру обробки даних. Caption - напис на кнопці;

 **CheckBox** **Перемикач**. Має 2 стани (включено і виключено), як правило, використовується для включення будь-які настройки в програмі. Checked - стан перемикача; Caption - назва перемикача.

 **ListBox** **Список**. Використовується для вибору одного чи більше значень із задалегідь заданого набору варіантів. MultiSelect - чи можна вибирати кілька елементів (якщо так - при натиснутій Ctrl); Sorted - сортувати список; ItemIndex - номер обраного елемента (з 0); Items - контейнер для елементів списку; Items.Count - кількість елементів; Items.Clear - метод для очищення списку; Items.Add (рядок) - метод для додавання елемента; Items.Delete (номер) - видалити елемент з вказаним номером (з 0); Items.Insert (номер; елемент) - вставити елемент перед елементом з вказаним номером (з 0).

Типи і перетворення даних

У Delphi підтримуються всі базові типи даних і операції Паскаля. Для рядків допустимо складання:

```
Var s, s2, name: string; Name: = 'Іван'; S2: = 'Привіт,' # 13 Name;
```

Для перетворення типів використовують наступні функції:

IntToStr (Вираз): String;

FloatToStr (Вираз): String;

FloatToStrF (Вираз, Формат, Точність, Кількість Знаків): String;

Формат - іменована константа (ffGeneral - загальний числовий, ffExponent - науковий, ffFixed - завжди з десятковою крапкою, ffCurrency - грошовий формат). Точність показує кількість знаків в дробовій частині, що використовуються у

розрахунках (використовуються значення 7,15,18). Кількість знаків показує відображуване кількість знаків в дробовій частині.

StrToInt (Рядок): integer; - перетворення рядка в ціле число;

StrToFloat (Рядок): Extended; - перетворення рядка в дійсне число.

Приклад введення даних:

1) з діалогового вікна повідомлення:

Змінна: = InputBox (Заголовок Вікна, Підказка, Значення За Замовчуванням)

s: = InputBox ('Сантиметри і дюйми', 'Введіть довжину в сантиметрах :','");

sm: = StrToFloat (s);

Кнопка ОК: s = введеному значенню

Кнопка Отмена: s = Значення За Замовчуванням

2) з поля вікна діалогу: введення здійснюється зверненням до властивості Text цього поля

a: = StrToFloat (Edit1.Text);

3) з текстового файлу, відкритого для читання

var f: TextFile;

AssignFile (f, 'шлях до файлу');

Reset (f);

ReadLn (f, s);

Edit1.Text: = s;

CloseFile (f);

Виведення даних:

1) У вікно повідомлення: ShowMessage (Повідомлення);

MessageDlg (Повідомлення, Тип, Кнопки, КонтекстСправки): integer;

Тип - це вид повідомлення. Задається іменованою константою: mtWarning (увага), mtError (помилка), mtInformation (повідомлення), mtConfirmation (підтвердження), mtCustom (без піктограми).

Кнопки - список іменованих констант в квадратних дужках. mbYes, mbNo, mbOK, mbCancel, mbHelp, mbAbort, mbRetry, mbIgnore, mbAll

Значення, що повертається - також іменування константа: mrAbort, mrYes, mrOK, mrRetry, mrNo, mrCancel, mrIgnore, mrAll

2) у полі діалогового вікна - Звернення до властивості Caption текстової мітки Label1.Caption: = 'Рівняння не має коренів';

Або до властивості Text поля введення

Edit1.Text: = 'ОК';

Або до властивості Lines багаторядкового поля:

Memo1.Lines.Clear;

Memo1.Lines.Add ('ОК');

ЗАВДАННЯ.

1. Запустити обране програмне середовище RAD, наприклад, BC++ Builder (можливе використання будь-якої іншої RAD за погодженням з лектором).

2. Ознайомитися з командами головного меню, призначенням інспектора об'єктів, редактором форм і редактором кодів, а також палітрою компонент за допомогою підказок.

3. Встановити закладки Standard і послідовно на редактор форм перетягувати візуальні компоненти, навчитися змінювати властивості об'єктів, використовуючи можливості інспектора об'єктів.

4. Запустити програму.

У звіт ЛР помістити відповіді на контрольні запитання. ЛР оформляти по загальноприйнятій формі аналогічно до інших предметів. Для підвищення балів ЛР, необхідно на колоквіумі відповісти на додаткові питання, що стосуються ЛР.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке RAD?
2. Що таке інспектор об'єктів?
3. Що таке редактор форм?
4. Що таке редактор кодів?
5. Що таке палітра компонентів?
6. Яка структура модуля обраної RAD?
7. Яким чином забезпечується запуск програми в середовищі RAD?
8. Якій парадигмі програмування відповідає розглянута RAD (Лекція № 1)?