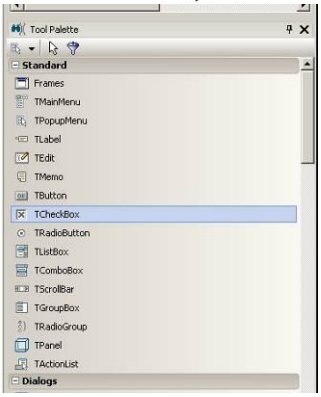
Тема: Форми та елементи управління.

Мета: Познайомитися із налаштуванням компонентів і форм: зміна їх властивостей, програмування обробника події, використання вікна повідомлення у візуальному середовищі програмування Delphi.

1. Основні компоненти Windows-програми.

Середовище розробки Delphi орієнтоване перш за все на створення програм для Windows. При цьому особлива увага приділяється можливості візуальної розробки додатків за допомогою великої кількості готових компонентів Delphi, що дозволяє уникнути ручного кодування. Компоненти Delphi охоплюють практично всі аспекти застосування сучасних інформаційних технологій.

Знайомство із основними компонентами Windows-програми розпочнемо зі сторінки Standard палітри компонентів. На ній розміщені 16 об'єктів, які найбільш часто зустрічаються у програмах: кнопки, списки, вікна введення тощо. Набір і порядок компонентів на кожній сторінці можна конфігурувати: додавати до наявних компонентів нові, змінювати їхню кількість та послідовність тощо.

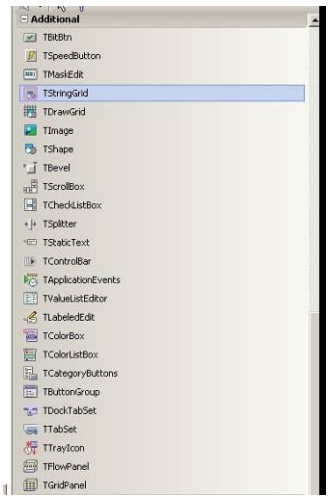


Перерахуємо компоненти Delphi із палітри Standard та наведемо деякі зауваження щодо їх застосування.

Frames	Об'єкт-контейнер для виведення вмісту фреймів на форму. Натиснути його можна тільки, якщо Ви заздалегідь створили через File – New – New Items, Delphi Files – Frame один чи декілька фреймів – тоді Frames запропонує на вибір список фреймів, які він може вивести на форму. Наприклад, ефектно виглядить форма, коли в залежності від ситуації на ній змінюється конфігурація кнопок. Для створення такої форми програміст повинен заготовити декілька фреймів, які при виконанні програми можуть бути видимими чи невидимими.
TMainMenu	Компонент для внесення меню до програми. При розміщенні MainMenu на форму він виглядає, як звичайна іконка. Іконки такого типу називають невидимими компонентами, оскільки під час виконання програми вони стають невидимими. Створення меню включає три кроки: 1. Розміщення MainMenu на форму. 2. Виклик дизайнера меню. 3. Визначення пунктів меню в дизайнері меню.
TPopupMenu	Меню, яке «спливає». Цей тип меню з'являється при клацанні правою кнопкою мишки (контекстне).

TLabel	Компонент для відображення тексту на екрані. Ви можете змінювати шрифт та колір мітки, двічі клацнувши на властивості Font в інспекторі об'єктів. Це легко зробити й під час виконання програми, написавши лише один рядок коду.
TEdit	Стандартний елемент управління Windows для введення текстів. Він може бути використаний для відображення короткого фрагменту тексту і дозволяє користувачу вводити текст під час виконання програми.
TMemo	Багаторядковий текстовий редактор. Використовується для введення користувачем і відображення багаторядкового тексту без функцій форматування.
TButton	Кнопка, яка дозволяє здійснювати деякі дії під час виконання програми. В Delphi все робиться дуже просто. Розмістивши Button на форму, шляхом подвійного клацання Ви можете створити заготовку обробника події – у даному випадку це натискування кнопки.
TCheckBox	Незалежний перемикач. Використовується його властивість Checked (відмічено), яка має значення true або false, що змінюється при клацанні мишкою. Відображається у вигляді рядка тексту із маленьким віконцем зліва. На питання, розміщене у рядку компоненту, можна дати ствердну відповідь, поставивши позначку у віконці.
TRadioButton	Залежний перемикач, який використовується для вибору тільки одного із декількох варіантів. Для цього компонент об'єднується з одним чи декількома такими ж компонентами у групу. Клацання мишкою на компоненті призводить до його виділення та зняття виділення раніш відміченого компоненту.
TListBox	Список вибору, який містить список запропонованих варіантів (опцій) та дає можливість проконтролювати поточний вибір.
TComboBox	Список, що розкривається; багато в чому нагадує ListBox, проте, на відміну від нього, дозволяє вводити інформацію у маленькому полі введення зверху ListBox. Є декілька типів ComboBox. З них найбільш популярним є «випадаючий список» (drop-down combo box). Це – багаторядкове меню, яке в неробочому стані згорнути до одного активного рядка. При активації цей компонент розгортається до повного списку.
TScrollBar	Смуга прокручування, являє собою вертикальну чи горизонтальну смужку, яка керує зміщенням візуального представлення компонентів, які не поміщуються повністю у вікні програми.
TGroupBox	Компонент-контейнер, який використовується для візуальних цілей та для вказівки Windows, який порядок переміщення по компонентах на формі при натискуванні клавіші Tab.
TRadioGroup	Група залежних перемикачів. Містить спеціальні властивості для обслуговування декількох зв'язаних між собою залежних перемикачів.
TPanel	Панель, яка, як і GroupBox, служить для об'єднання декількох компонентів. Містить внутрішню та зовнішню межі, що дозволяє створювати ефекти «натиснення» та «опуклості».
TActionList	Список дій, які служать для програмування реакції програми на дії користувача, пов'язані із вибором одного із групи однотипних керуючих елементів, таких, як опції меню, кнопки тощо.

Навіть для створення нескладних додатків наведений перелік компонентів може виявитись недостатнім, тому розглянемо другу сторінку компонентів Additional.



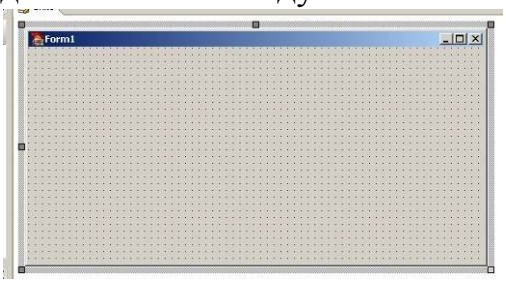
Перерахуємо компоненти сторінки Additional, що найбільш часто вживаються, із деякими коментарями щодо їх застосування.

TBitBtn	Командна кнопка. Відрізняється від стандартної кнопки Button можливістю відображати піктограми.
TSpeedButton	Піктографічна кнопка. Використовується, як правило, для швидкого доступу до опцій головного меню.
TMaskEdit	Аналог Edit, який забезпечує можливість форматованого введення. Формат визначається у властивості EditMask. В редакторі властивостей для EditMask є заготовки для введення – виведення дати, валюти тощо.
TStringGrid	Таблиця рядків. Цей компонент володіє потужними властивостями для представлення текстової інформації в табличному вигляді.
TDrawGrid	Таблиця зображень. Використовується для представлення зображень у табличному вигляді.
TImage	Малюнок. Використовується для відображення зображень, у тому числі піктограм та метафайлів.
TShape	Фігура. За допомогою цього компоненту можна вставити на форму правильну фігуру – прямокутник (квадрат), еліпс (коло).
TBevel	Межа. Служить для виділення окремих частин форми трьохмірними рамками та смужками.
TScrollBox	Панель із смужками прокрутки. На відміну від компоненту Panel автоматично вставляє смужки прокрутки, якщо розміщені на ній компоненти виходять за її межі.
TCheckListBox	Список багатоваріантного вибору. Відрізняється від стандартного компоненту ListBox наявністю поряд із кожною опцією незалежного перемикача типу CheckBox, який дозволяє вибрати відразу декілька опцій.
TSplitter	Границя. Цей компонент створює границю між двома видимими компонентами і дає користувачу можливість переміщувати її, збільшуючи один компонент за рахунок іншого.
TStaticText	Статистичний текст. Відрізняється від стандартного компоненту Label наявністю власного Windows-вікна, що дозволяє обводити текст рамкою або виділяти його у вигляді «втопленої» частини форми.

2. Елементи керування та їх атрибути.

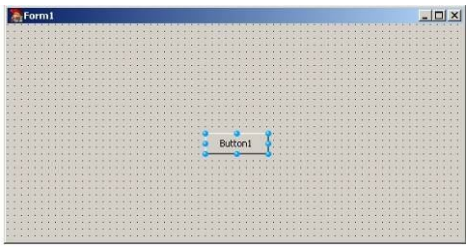
При розробці програм у візуальних середовищах програміст працює не лише із формами. Істотною підтримкою виступають «візуальні компоненти». В інтерфейсі Delphi заздалегідь заготовлено декілька інструментальних закладок, які містять багато дуже зручних візуальних компонентів.

Візуальні компоненти являють собою логічно завершений програмний код, який виконує операції з даними окремих модулів. Практично будь-який програмний код може бути представлений у вигляді компонента. Візуальний компонент – це заздалегідь заготовлений тип даних, багато в чому подібний до integer, string, array, хоча й набагато складніший. Переносячи компонент із інструментальної лінійки на форму, ми створюємо в програмі «екземпляр об'єкту», тобто, змінну даного типу. Наприклад, у нас була порожня форма. Відкривши вікно програми, ми побачимо заготовку для написання коду:




```
type
  TForm1 = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

Перенесемо на форму кнопку (змінну типу TButton), й опис форми автоматично зміниться:




```
type
  TForm1 = class(TForm)
    Button1: TButton;
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

На формі з'явилась змінна Button1 типу TButton (екземпляр об'єкту «кнопка»).

Візуальні компоненти мають п'ять груп основних характеристик:

Властивості (Properties) – параметри, які описують характерні ознаки об'єкта, наприклад, ширина компоненту на формі (Width), висота компоненту (Height), доступність (логічна властивість Enabled) тощо. Набір властивостей для різних компонентів різний. Зокрема, у невидимих об'єктів немає ні «ширини», ні «висоти».

Події (Events) – набір дій на компонент, які викликають виконання ним деяких програмних операцій. Ці дії можуть бути як від зовнішніх пристроїв (наприклад, OnMouseDown – натискування кнопки мишки у момент, коли вказівник мишки знаходиться над компонентом), так і програмними (наприклад, OnCreate – дії, які потрібно виконувати при створенні форми).

Методи (Methods) – набір процедур та функцій, які потрібно виконати компоненту у відповідь на відповідну подію. Програміст за допомогою інспектора об'єктів як правило записує код лише в потрібні процедури, залишаючи незаповненими непотрібні реакції на події. В більш складному варіанті можна програмувати «акціями» (перший рядок на закладці Events у вікні інспектора об'єктів). У цьому випадку всі методи програмуються окремо і просто назначаються даному компоненту.

Ще дві характеристики – Повідомлення (Messages) та Контекст (Context) – визначають правила передачі управлінської інформації між компонентами і режим графічного відображення компонентів.

Перші кроки в програмуванні візуальних компонентів ми зробимо, створивши невелику програму, яка полягає у призначенні деяких властивостей обмеженої кількості компонентів та програмуванню елементарних методів.

Завдання. Необхідно створити програму, в якій на форму винесені дві великі кнопки із картинками (саме для цього використовується компонент SpeedButton). На одній кнопці зображено кішку, на іншій – мишку. Спочатку кішка стоїть перед ніркою, звідки тільки виглядає мишачий хвостик. Коли користувачу набридає чекати, він може натиснути на кнопку «Кішка». Кішка нявкає та лягає спати. Із нірки з'являється мордочка мишки. Якщо тапер натиснути на кнопку «Мишка», то миша вискакує із нірки, але бачить кішку, лякається і ховається у нірку, залишаючи зовні лише хвостик. При цьому кішка просинається і знову стоїть біля нірки.

Можна виділити чотири фази дії цієї програми:

- Кішка стоїть біля нірки, з якої виглядає мишачий хвостик (вихідне положення).
- Кішка спить, а мишка вилізла з нірки (після натискування кнопки «Кішка»).
- Кішка стоїть біля нірки, при цьому нявкає, з нірки виглядає мишачий хвостик (натискування кнопки «Кішка»).
- Кішка спить, а з нірки з'являється мордочка мишки (натискування кнопки «Мишка»).

Якщо відволіктись від образного опису задачі, можна зробити висновок: програміст зіткнувся тут із проблемою трьохкратної заміни картинок на кнопках. Картинка кішки: в активному стані вона «стоїть», у недоступному стані – «спить», в натиснутому стані – «нявкає». Картинка мишки: в активному стані вона «виглядає», в недоступному – «сховалась», в натиснутому – «вибігає».

Можливість подібної заміни картинок уже передбачена самим компонентом SpeedButton. У даному випадку, хоча це може й здатись дивним, написання програми полягає в основному не в запису програмного коду, а в роботі із графічним редактором. З цього й розпочнемо.

Компонент SpeedButton має властивість Glyph, в яку записується ім'я файлу, що містить картинку для виведення на кнопку. Визначення цієї властивості потрібно виконувати тоді, коли картинка буде готова. Особливість компоненту полягає у тому, що в різних режимах він може виводити одну, дві або три картинки в залежності від стану кнопки.

Картинки не масштабуються. Тому, наприклад, якщо наші кнопки, наприклад, мають розміри 75 х 75, в графічному редакторі потрібно малювати картинки висотою 70 пікселів і шириною 210 пікселів (тобто тричі по 70 пікселів). Зручніше це зробити, намалювавши спочатку три однотонні рамки, які будуть потім витерті.

Порядок малювання картинки визначається компонентом SpeedButton: її лівий фрагмент виводиться при не натиснутій активній кнопці, середній – при відключеній (властивість «доступність» Enabled встановлена у стан false), правий – тимчасово виводиться у момент натискування кнопки.

Щоб картинка не зміщувалась при натискуванні кнопки, лівий та правий її фрагменти краще копіювати, змінюючи лише частину деталей (кішка «нявкає»). По завершенні малювання опорні рамки потрібно витерти.

Таким же чином виконуємо три фрагменти малюнку мишки.

Тепер внесемо картинки кішки на кнопки. Виділяємо кнопку, вибираємо властивість Glyph, клацанням мишки на полі введення вносимо потрібні значення і у вікні, що відкривається, вказуємо ім'я файлу. За замовчуванням у властивості NumGlyphs (кількість картинок) встановлено значення 1, тобто, одна картинка (виводиться центральна частина виконаного малюнку). Змінюємо це значення на 3 – у нас заготовлено 3 картинки на кнопку. Аналогічним чином робимо й із картинками мишки.

Після цього розроблену нами форму можна запустити. Не дивлячись на те, що ми не написали жодного рядку коду, створений додаток виглядає цілком працездатним – він запускається, кнопки натискуються, на кнопках змінюються картинки; проте для отримання зовсім працездатної, такої, що відповідає умові задачі програми, все-таки доведеться написати декілька рядків програмного коду.

На кнопці SpeedButton1 («кішка») вибираємо рядок обробника коду OnClick та вписуємо в процедуру два рядки:

```
procedure TForm1.SpeedButton1Click(Sender: TObject);
begin
 SpeedButton1.Enabled:=false;
 SpeedButton2.Enabled:=true;
end;
```

На кнопці SpeedButton2 («мишка») вибираємо рядок обробника коду OnClick та вписуємо в процедуру два рядки:

```
procedure TForm1.SpeedButton2Click(Sender: TObject);
begin
 SpeedButton2.Enabled:=false;
 SpeedButton1.Enabled:=true;
end;
```

Така програма діє у відповідності із поставленим завданням.

На завершення декілька слів про вправила інтерпретації малюнка. При вставці компоненту Image ліва нижня точка малюнку (піксел) визначає «колір прозорості», якщо у властивості Transparent встановлене значення true. На кнопці відображаються не всі точки цього кольору, і на малюнку просвічується сірий фон.

У властивість NumGlyphs можна вводити числа від 1 до 4, при цьому:

- Якщо там знаходиться 1, то одна й та ж картинка виводиться на поверхню кнопки незалежно від її стану.
- Якщо там знаходиться 2, лівий фрагмент малюнку виводиться на кнопку в активному (у тому числі й натиснутому) стані, а правий – у недоступному стані.

– Якщо там знаходиться 4, то значить, що у нас є ще й четверта картинка для фіксації кнопки в натиснутому стані.

Для «кнопок із фіксацією» на формі потрібно розмістити декілька SpeedButton, у яких у властивості GroupIndex міститься одне й те саме число. В такому випадку ці кнопки вважаються групою; при натискуванні однієї кнопки вона фіксується, а всі інші – віджимаються у вихідний активний стан. На зафіксованій кнопці й виводиться четвертий фрагмент малюнку. Такий режим є зручним для програмування деяких автоматизованих тестів із вибором одного із декількох варіантів відповіді.

Пропонуємо Вам ще один приклад жартівливого завдання. Внести зміни до попереднього завдання таким чином, щоб отримати картинку «Мартовские коты» - якщо натиснути на кнопку із котом, він «кричить», потім лягає спати, а другий кіт просинається – і навпаки.

Особливістю цього прикладу є те, що Вам не потрібно малювати ще малюнки, змінювати програмний код тощо. Вистачає лише зміни однієї властивості: замінити для SpeedButton2 назву файлу (тепер це один і той же малюнок як для першої, так і для другої «кішок»). Програмний код співпадає із кодом програми «Кішка-мишка».

3. Використання вікон повідомлень.

Візуальне середовище Delphi дозволяє застосовувати у програмах стандартні вікна повідомлень, імпортовані із WinAPI. Таких вікон декілька, і можливості введення даних із їх допомогою дуже обмежені. Проте ці вікна мають і позитивні властивості. По-перше, їх легко програмувати; по-друге, вони є вікнами модельного діалогу, можуть переривати програму і дозволяють ввести дані або підтвердити певну дію навіть із циклу чи рекурсивної функції.

Вікно повідомлень.

Функція MessageBox є зручним засобом для виведення на екран невеликого повідомлення. Синтаксис:

k:=messagebox (номер вікна, заголовок, рядок, код кнопок);

Параметри цієї функції:

– Номер вікна – це числове значення, атрибут вікна, яке викликало вікно повідомлення. Використовується для багато віконних користувацьких додатків. В одновіконному додатку у номері вікна ставиться 0.
– Заголовок – рядок, який буде виводитись у заголовку вікна повідомлення.
– Рядок – текстова інформація, яка виводиться у вікні повідомлення. Зауважимо, що ширина вікна визначається довжиною цього рядка або сумарною шириною кнопок (якщо рядок виявиться коротшим сумарної ширини кнопок). Висота вікна – це висота рядка плюс висота кнопок. Якщо потрібний рядок, що складається із декількох рядків, в його текст необхідно вписати коди переведення рядка (#13), наприклад:

‘Наша Таня’ + #13 + ‘Громко плачет,’ + #13 + ‘Уронила в речку мячик’

– Код кнопок визначає, який їх набір буде виведений у вікні повідомлення:

- MB_AbortRetryIgnore - Abort, Retry, Ignore
- MB_OK - OK (за замовчування)
- MB_OK_Cancel - OK, Cancel
- MB_RetryCancel - Retry, Cancel
- MB_YesNo - Yes, No
- MB_YesNoCancel - Yes, No, Cancel

Іконка у вікні визначається константами:

- MB_IconExclamation, MB_IconWarning - знак «!»
- MB_IconInformation, MB_IconAsterisk - знак «i»
- MB_IconQuestion - знак «?»
- MB_IconStop, MB_IconError, MB_IconHand - знак «x»

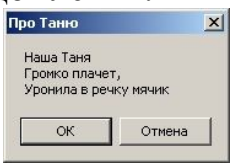
Для значень, які повертає функція в залежності від того, яка кнопка натиснута, є відповідні константи (символьні або числові):

Кнопки вікна повідомлень та їх позначення:

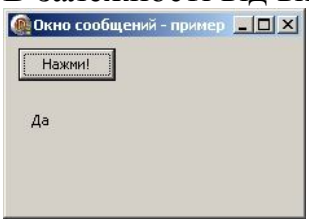
Кнопка	Константа (символьна)	Значення (числове)
OK	IDOK	1
Cancel	IDCancel	2
Abort	IDAbort	3
Retry	IDRetry	4
Ignore	IDIgnore	5
Yes	IDYes	6
No	IDNo	7

Функцію MessageBox можна використовувати як процедуру без повернення значення.

Для прикладу напишемо програму, яка б виводила наступний вірш у вікні повідомлення:



В залежності від вибраної відповіді на формі з'являвся б напис «Да» чи «Нет».



Дана програма реалізується наступним програмним кодом:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs, StdCtrls;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Label1: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
```

```
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
var k: integer;
begin
 k:=messagebox (0,'Наша Таня' + #13 + 'Громко плачет,' +#13 + 'Уронила в речку мячик','Про Таню',mb_OkCancel);
 if k=1 then
  label1.caption:='Да'
 else
  label1.caption:='Нет';
end;
end.
```

Вікно попередження.

На відміну від вікна повідомлення, у вікні попередження можна розмістити довільну комбінацію із восьми допустимих кнопок, окрім того, воно дозволяє повернути два числових параметри. Викликається вікно попередження наступним чином:

k:=messagedlg (рядок, тип вікна, перелік кнопок, число_help);

Параметри цієї функції:

- Рядок – текст, який виводиться у вікні попередження.
- Тип вікна – цей параметр визначається значком, який буде відображатись у вікні. Можливі наступні константи:
  - mtWarning – знак «!» у трикутнику;
  - mtError – знак «x» у кружечку;
  - mtInformation – літера «i» (синя) в білому кружечку;
  - mtConfirmation – знак «?» у кружечку;
  - mtCustom – порожнє вікно без знаку.
- Список кнопок – множина зарезервованих констант. Допустимі наступні значення, кожне із яких визначає відповідну кнопку:
  - mbOK – кнопка OK;
  - mbCancel – кнопка Cancel;
  - mbYes – кнопка Yes;
  - mbNo – кнопка No;
  - mbAbort – кнопка Abort;
  - mbRetry – кнопка Retry;
  - mbIgnore – кнопка Ignore;
  - mbAll – кнопка All;
  - mbNoToAll – кнопка NoToAll;
  - mbYesToAll – кнопка YesToAll.

Кнопки на вікні визначаються множиною у квадратних дужках. Якщо випадково двічі вкажемо одну й ту саму кнопку, то на формі з’явиться лише один її екземпляр. Причому, як і в звичайних множинах, порядок їх запису в операторі не має значення – вони виводяться у порядку зростання кодів повернення. Наприклад, якщо потрібно завершити діалог натискуванням однієї із трьох кнопок All, NoToAll,

YesToAll, то для цього у квадратних дужках потрібно записати: [mbAll, mbNoToAll, mbYesToAll].

Зверніть увагу на те, що натискування кнопки повертає відповідне значення:

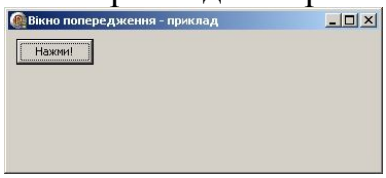
- mrNone – нічого не натиснуто (0);
- mrOk – кнопка OK (1);
- mrCancel – кнопка Cancel (2);
- mrAbort – кнопка Abort (3);
- mrRetry – кнопка Retry (4);
- mrIgnore – кнопка Ignore (5);
- mrYes – кнопка Yes (6);
- mrNo – кнопка No (7);
- mrAll – кнопка All (10);

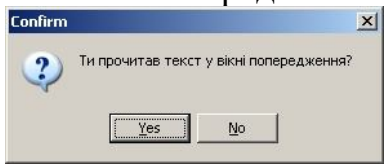
– число_help – числовий код для виклику контекстної допомоги.

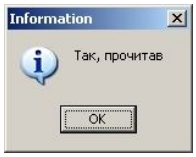
Функцію виклику вікна попередження можна використовувати як процедуру без повернення значення, тоді вона виконує задачу тимчасової зупинки програми із виведенням заданого значення.

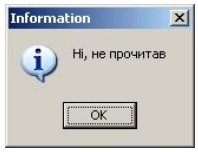
Ширина вікна визначається довжиною рядка або сумарною шириною кнопок. Висота – це рядок плюс кнопочка.

Приклад використання вікна попередження:









Потрібно вивести на екран вікно із двома кнопками Yes і No, а потім – вікно із відповіддю «Так, прочитав» чи «Ні, не прочитав» у залежності від вибору кнопки у першому вікні.

Текст програми наступний:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs, StdCtrls;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Label1: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
  if messagedlg ('Ти прочитав текст у вікні попередження?',mtConfirmation,
[mbYes,mbNo],0)=mrYes then
    messagedlg('Так, прочитав',mtInformation,[mbOk],0)
    else
```

```
    messagedlg('Hi, не прочитав',mtInformation,[mbOk],0);
  end;
  end.
```

Позиціоноване вікно попереджень MessageDlgPos

Вікно цього типу повністю повторює звичайне вікно попереджень, проте воно може бути виведене у будь-якій точці екрану.

k:=messagedlg (рядок, тип вікна, перелік кнопок, число_help, x, y);

Параметри цієї функції мають ті ж самі значення, що й у messagedlg, проте добавились ще два параметри – зміщення по горизонталі та вертикалі. Підрахунок ведеться від верхнього лівого кута екрану, а не від форми програми.

Вікно введення InputBox.

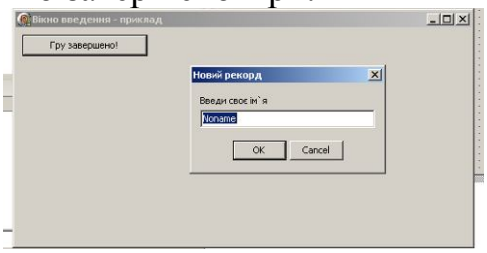
Це вікно призначене для введення рядка. Вікно викликається наступною функцією:

s:=inputbox (заголовок, підказка, рядок за замовчуванням);

Параметри функції:

- заголовок – напис на рядку заголовка вікна;
- підказка – рядок тексту, який виводиться над елементом Edit в цьому вікні;
- рядок за замовчуванням – вхідний текст у вікні до початку введення.

Приклад. Потрібно написати параметри для реєстрації гравця у випадку успішно завершеної гри.



Текст програми наступний:

```
unit Unit1;
interface
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
{$R *.dfm}

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

```
    inputstring:=inputbox ('Новий рекорд','Введи своє ім`я','Noname');
  end;
  end.
```

Рядок, набраний в елементі введення, повертається у програму та може бути присвоєний будь-якій текстовій змінній.

4. Теоретичні питання для самоконтролю.

1. Для чого призначені компоненти MainMenu, PopupMenu, Label, Edit, які знаходяться у палітрі компонентів Standard?
2. Для чого призначені компоненти Memo, Button, CheckBox, RadioButton, які знаходяться у палітрі компонентів Standard?
3. Для чого призначені компоненти ListBox, ComboBox, GroupBox, RadioGroup, які знаходяться у палітрі компонентів Standard?
4. Для чого призначені компоненти BitBtn, SpeedButton, StringGrid, DrawGrid, які знаходяться у палітрі компонентів Additional?
5. Для чого призначені компоненти Image, Shape, CheckListBox, StaticText, які знаходяться у палітрі компонентів Additional?
6. Що являють собою візуальні компоненти?
7. У чому полягає особливість повідомлень та контекстів?
8. Перерахуйте можливості, які надає вікно повідомлень MessageBox.
9. Перерахуйте можливості, які надає вікно попереджень MessageDlg.
10. Для чого призначене вікно введення InputBox?