МІНІСТЕРСТВО ВНУТРІШНІХ СПРАВ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВНУТРІШНІХ СПРАВ

Кафедра інформаційних технологій факультету № 4

ТЕКСТ ЛЕКЦІЇ

з навчальної дисципліни «Алгоритмізація та програмування»
обов'язкових компонент
освітньої програми першого рівня вищої освіти
125 «Кібербезпека»
(«Поліцейська діяльність у кіберсфері», «Безпека інформаційних та комунікаційних систем»)

за темою – «Найпростіше введення/виведення у Windows Forms»

Харків 2019

ЗАТВЕРДЖЕНО
Науково-методичною радою
Харківського національного
університету внутрішніх справ
Протокол від _24.01.19_ № _1____

СХВАЛЕНО
Вченою радою факультету № 4
Протокол від _16.01.19_ № _1____

ПОГОДЖЕНО
Секцією Науково-методичної ради
ХНУВС з технічних дисциплін

Протокол від _17.01.19_ № _1____

Розглянуто на засіданні кафедри інформаційних технологій _(протокол від 15.01.19_ № 1)_

Розробники:
1. _Зав. кафедри,к.т.н., доцент Струков В.М._

Рецензенти:
1. _д.т.н., професор Зацеркляний М.М._

План лекції

1. Просте введення даних.
2. Просте виведення даних.

Література:

Основна:

1. Э. Троелсен. Язык программирования C# 5.0 и платформа .NET 4.5. , 6-е изд. : Пер. с англ. — М. : ООО "И.Д. Вильямс", 2013. — 1312 с. : ил. — Парал. тит. англ.
2. Конспект лекцій.

Додаткова:

3. Программирование на языке C#: учеб. пособие / Л.В. Соловей, Н.Н. Мирошниченко, Н.Г. Пономарёв. – Х. : НТУ «ХПИ», 2016. – 356 с.

Текст лекції

1. Просте введення даних.

Треба сказати, що введення даних з клавіатури і виведення даних на екран дисплея у Windows Forms істотно відрізняється від того, як це робилося в консольних додатках. Це пов'язано з тим, що стандартні вбудовані засоби консольних додатків (оператори *Read, Write*) призначені для роботи в інтерфейсі командного рядка. Застосування ж, що розробляються в середовищі у Windows Forms, призначені для роботи в середовищі *Windows,* тобто в графічному діалоговому віконному призначеному для користувача інтерфейсі. А в цьому випадку оператори *Read* і *Write* вже не використовуються. Тут використовуються саме діалогові засоби віконного типу, на описі яких ми і зупинимося нижче.

Просте введення даних у *Windows Forms* (введення скалярних значень) можна здійснювати, щонайменше, 6 способами:

1. За допомогою візуального компонента **TextBox**;
2. За допомогою візуального компонента **MaskedTextBox**;
3. За допомогою візуального компонента **NumericUpDown**;
4. За допомогою візуального компонента **DomainUpDown**;
5. Програмним шляхом.
6. З файлу.

Введення даних за допомогою компонента TextBox.

Для реалізації цього способу використовується візуальний компонент *TextBox* з групи стандартних елементів управління.

Компонент *TextBox* призначений для введення текстової або числової інформації, яка може поміщатися в один (або декілька) рядок. У рядку введення з текстом можна виконувати всі прості операції редагування тексту: виділення, видалення, копіювання.

Компонент *TextBox* має багато властивостей, які згруповані у групи – 1) Зовнішній вигляд (Внешний вид); 2) Дані; 3) Макет; 4) Поведінка (Поведение); 5) Інші (Прочее); 6) Разработка; 7) Спеціальні можливості; 8) Фокус.

Група властивостей Внешний вид.

Властивості цієї групи визначають, як виглядатиме на формі компонент. Основними параметрами зовнішнього вигляду є наступні: колір фону компонента, колір і стиль тексту, який відображається в полі введення, колір і стиль границі, параметри вирівнювання тексту, стиль курсору. Ці властивості доступні у режимі Конструктора (вікно Свойства), а також можуть бути змінені або встановлені під час виконання додатку програмним шляхом.

Ім’я	Тип	Призначення	
BackColor	Приймає одне із іменованих значень структури Color або числове значення 32-розрядного числа у шістнадцятковому представленні	Повертає або задає колір фону елемента управління.	Простий тип типу *enum*
ForeColor	Приймає одне із іменованих значень структури Color або числове значення 32-розрядного числа у шістнадцятковому представленні	Повертає або задає колір тексту елемента управління.	Простий тип типу *enum*
Font	Складений тип	Визначає стиль шрифту, яким відображається текст в полі введення	Составний тип
BorderStyle	Складений тип	Визначає стиль границі компонента	Составний тип
Text	string	Має значення рядка тексту, що вводиться в поле	string
TextAlign	Приймає одне із іменованих	Визначає тип	Простий

	значень Left, Right, Center	вирівнювання тексту в полі введення	Простий тип типу *enum*
Lines	Складений тип	Масив рядків тексту	Масив рядків типу string

Приклад.
textBox1.ForeColor = Color.Red;
textBox1.BackColor = Color.Yellow;

Група властивостей Макет.

Властивості цієї групи визначають, як буде розташований компонент на формі. Основними параметрами зовнішнього вигляду є наступні: координати розміщення компонента у контейнері; відстань до інших компонентів контейнера; розмір компонента.

Ім’я	Тип	Призначення	
Location	Структура, яка має тип *Point(X, Y)*	Визначає положення верхнього лівого кута компонента у контейнері.	Структура
Margin	Структура типу *Padding*, яка має 8 полів	Визначає відстань до інших елементів управління в контейнері.	Структура
Size	Структура типу *Size*, яка має два основних поля – *Height, Width*	Повертає або задає висоту і ширину елемента управління	Составний тип

Приклад.
textBox1.Location = new Point(15, 15);

Група властивостей Поведінка.

Властивості цієї групи визначають поведінку компонента під час виконання додатку.

Ім’я	Тип	Призначення	
ContextMenuStrip	Ім’я компоненту типу *ContextMenuStrip*	Визначає наявність	Посилання на

		спливаючого меню для данного компоненту.	компонент
Enabled	Логічний тип, що має одне з двох значень – *true* або *false*	Визначає доступність компонента під час виконання.	Скалярний тип
Visible	Логічний тип, що має одне з двох значень – *true* або *false*	Визначає видимість компонента під час виконання.	Скалярний тип

Для введення числових даних за допомогою компонента у відповідному обробщику подій потрібен фрагмент коду наступного типу:

```
private void button1_Click(object sender, EventArgs e)
    {
      int a, b, c;
      a = Convert.ToInt32(textBox1.Text);
      b = Convert.ToInt32(textBox2.Text);
      c = a + b;
      label1.Text = Convert.ToString(c);
    }
```

Введення цілих чисел за допомогою компоненту NumericUpDown.

Компонент **NumericUpDown** призначений для введення цілих чисел, які знаходяться в певному діапазоні. Діапазон задається в процесі розробки застосування програмістом. Під час виконання програми користувач може вводити числове значення в поле введення одним з двох способів:

1) шляхом введення з клавіатури і 2) за допомогою стрілок.

Введене значення автоматично перетвориться до цілого типу.

Основними властивостями цього компоненту є:

- *Increment:integer* – задає крок, на величину якого зменюється (збільшується або зменшується) клацанням лівою кнопкою миші по одній із стрілок („вверх” або „вниз”) значення, що вводиться; за умовчанням крок приросту дорівнює 1;
- *MaxLength:integer* – задає максимальну кількість символів, які можуть бути введені у вікно введення; якщо введена кількість

символів буде більша, ніж *MaxLength,* то вона автоматично буде зменшеною до *MaxLength;* якщо *MaxLength* встановити меншим або рівним 0 (наприклад -1 або -25), то контроль кількості введених символів буде відключений;

- *MaxValue:integer* – задає максимальне значення числа, яке може бути введене в поле введення;

- *MinValue:integer* – задає мінімальне значення числа, яке може бути введене в поле введення;

- *Value:integer* – задає мінімальне значення числа, яке може бути введене в поле введення;

- *ReadOnly:boolean* – дозволяє (*false*) або забороняє (*true*) введення даних в поле введення *SpinEdit.*

ЗАУВАЖЕННЯ 1. *Я*кщо *MaxValue= MinValue*, то діапазон значення, що вводиться, обмежується тільки діапазоном представлення значень типу integer.

ЗАУВАЖЕННЯ 2. *Я*кщо введене значення більше, ніж *MaxValue,* то воно автоматично зменшується до значення *MaxValue.*

ЗАУВАЖЕННЯ 3. *Я*кщо введене значення менше, ніж *MinValue,* то воно автоматично збільшується до значення *MinValue.*

Переваго використання компоненту *SpinEdit* в порівнянні з компонентом *Edit* при введенні цілих чисел є відсутність необхідності перетворення введеного значення до цілого типу.

Приклад можливого використання компоненту *SpinEdit:*

```
Var k:integer;
 . . . . . .
k:= SpinEdit1.Value+5;
```

Просте виведення даних на екран.

Розглянемо наступні способи виведення даних:

1. З використанням стандартної процедури *MessageBox.Show*.
2. З використанням вікна виведення компоненту *Label*.

<u>Виведення повідомлень з використанням стандартних процедур.</u>

Найпростішим засобом виведення текстових повідомлень в *Windows Forms* використання процедури *MessageBox.Show*.

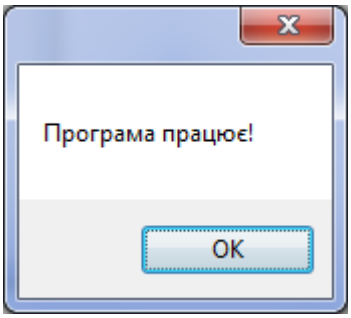
Формат звернення до процедури *MessageBox.Show* має наступний вигляд:

MessageBox.Show(<рядковий вираз>);

Для виведення якого-небудь повідомлення на екран необхідно вставити в обробник події оператор, наприклад, наступного вигляду:

```
MessageBox.Show(“Програма”);
```

Після виконання цього оператора на екрані з'явиться наступне вікно:



Доцільно використовувати цю процедуру для виведення інформаційних або застережливих повідомлень.

<*Тип*> - тип повідомлення (інформаційний, попереджувальний або ін.): *MtError, MtInformation, MtConfirmation, MtWarning;*

<*Кнопки*> - число і вид кнопок. На цьому місці записуються заключені в квадратні дужки іменовані константи, що визначають вигляд і кількість використовуваних кнопок, наприклад: [*MbOk, MbCancel*].

Можливе використання наступних іменованих констант: *MbYes, MbNo, MbHelp, MbRetry.*

<*Контекст*> - визначає номер екрану довідкової системи, який з'являється при натисненні клавіші *F1* (якщо довідка не потрібна, то в цьому місці ставиться 0).

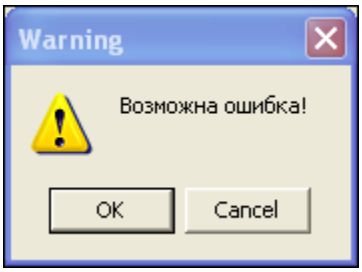
Значення, що повертаються: *Mb* міняється на *Mr*. Для визначення, яка саме кнопка була натиснута, можна скористатися наступним оператором:

if R=MrOk then . . . ;

Приклад.

R:=MessageDlg('Возможна ошибка!',MtWarning,[MbOk, MbCancel],0);

При виконанні цього оператора на екрані з'явиться вікно наступного вигляду:



ЗАУВАЖЕННЯ. Якщо за допомогою описаних процедур потрібно вивести числове значення, то його заздалегідь необхідно перетворити в текстове представлення за допомогою процедур *Str, IntToStr, FloatToStr,* наприклад:

Var k:integer;

. . .

k:=2;

ShowMessage('k='+IntToStr(K));

Виведення повідомлень за допомогою компоненту Label.

Для виведення даних на екран за допомогою компоненту  *Label* використовується властивість *Caption* цього компоненту. Ця властивість доступна як в режимі візуального програмування, так і в процесі виконання програми. Для формування тексту при виконанні програми необхідно в обробнику подій тексту записати рядок наступного вигляду:

Label1.Caption:=<Строковий вираз>;

ЗАУВАЖЕННЯ 1. Для виведення числових значень вони повинні бути заздалегідь перетворені в текстове уявлення за допомогою процедур *Str*, *IntToStr*, *FloatToStr*, наприклад:

Var k:integer;

. . .

k:=2;

Label1.Caption:='k='+IntToStr(K);

ЗАУВАЖЕННЯ 2. Компонент Label можна використовувати для виведення багаторядкових повідомлень. Для цього використовують управляючий символ 'перехід в початок наступного рядка' - #13.

Приклад.

Label1.Caption:='Строка1'+#13+' Строка2';

Після виконання цього оператора в полі компоненту *Label* будуть виведені два рядки:

Строка1

Строка2

Введення значень за допомогою компоненту LabeledEdit.

Введення значень за допомогою компоненту *Edit* незручне тим, що для позначення його призначення необхідно використовувати ще одну компоненту – *Label.* Для усунення цієї маленької незручності в пізніх версіях *Windows Forms* в палітру *Additional* введений комбінований компонент  – помічене поле введення – *LabeledEdit.*

ОКРІМ ХАРАКТЕРНИХ ДЛЯ КОМПОНЕНТІВ *EDIT* І *LABEL* ВЛАСТИВОСТЕЙ, ТАКИХ ЯК *CAPTION* І *TEXT*, ЦЕЙ КОМПОНЕНТ ВОЛОДІЄ ТАКОЮ ВЛАСТИВІСТЮ ЯК

LabelPosition, яке визначає положення мітки (позначення) поля введення – зверху (*lpAbove*), знизу (*lpBelow*), зліва (*lpLeft*) або справа (*lpRight*). Властивість *Caption* є складовою частиною комплексної властивості *EditLabel*.