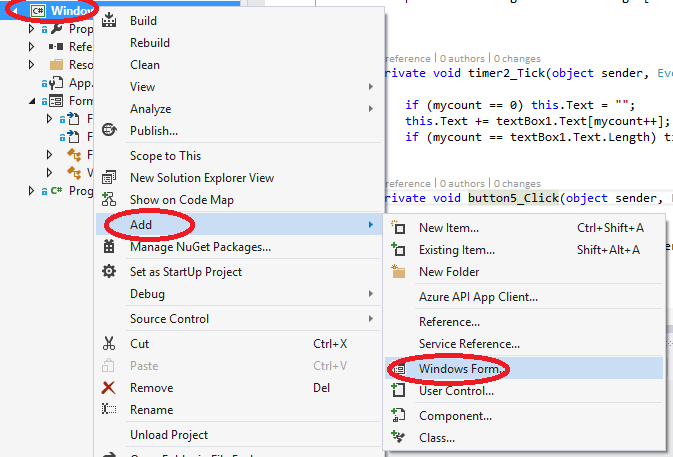
**Необхідність використання додаткових форм**

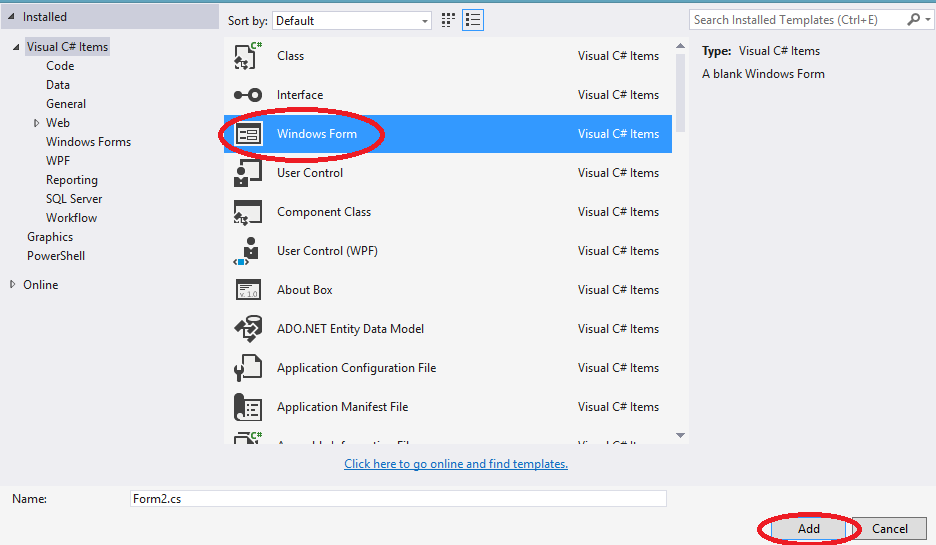
Чим об’ємніша програма, ти важче розмістити всю функціональність на одній формі. В таких випадках створюють додаткові форми. Додаткові форми дуже зручні для відображення різноманітних графіків і діаграм, для редагування і добавлення інформації, для заповнення анкет, для видачі повідомлень користувачу.

Форми бувають **модальні** і **немодальні**.

Модальна форма повністю блокує весь робочий процес до тих пір, поки вона не буде закрита. Немодальна форма не блокує робочий процес, дозволяючи користувачеві переключатися між формами додатку.

**Створення додаткових форм**





**Модальні та немодальні форми**

**Модальне вікно** не дає переключитися на інші вікна цього додатку до тих пір, поки воно не буде закрите.

**Немодальне вікно** дозволяє це зробити.

Для відображення модального вікна використовується метод **ShowDialog()**



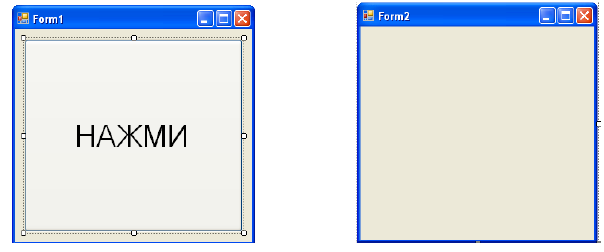
Для відображення немодального вікна використовується метод **Show()**



Form2 form2 = new Form2();

form2.ShowDialog(); //модальне вікно

form2.Show(); //немодальне вікно



private void button1\_Click(object sender, EventArgs e)

{

Form2 f = new Form2();

f.Show();

//f.ShowDialog();

MessageBox.Show("hello");

}

**Обмін даними між формами**

Як правило немає сенсу створювати форми, які між собою не будуть зв’язані. У зв’язку з цим виникає необхідність реалізувати обмін даними між двома формами.

Форма, яка реалізовує виклик додаткової форми, називається **батьківською**, форма, яку викликали – називається **дочірньою**.

Обмін даними між дочірньою і батьківською формами може відбуватися в двох напрямках: передача даних від батьківської до дочірньої і передача даних від дочірньої до батьківської.

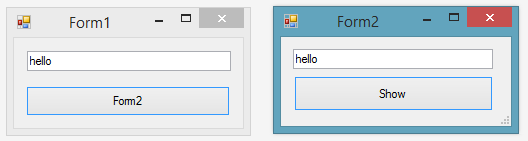
**Передача даних від батьківської форми до дочірньої**

Передачу даних від батьківської форми до дочірньої форми реалізувати простіше, оскільки батьківська форма містить ссилку на дочірню форму.

Передати дані дочірній формі можна наступними способами:

* Через конструктор дочірньої форми.
* Через додаткову функцію або властивість дочірньої форми
* Через перевантаження функції ShowDialog

Передача даних через конструктор дочірньої форми:



public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Form2 f = new Form2(textBox1.Text);

f.ShowDialog();

}

----------------------------------------------------------------------------

public partial class Form2 : Form

{

string str;

public Form2()

{

InitializeComponent();

}

public Form2(string str)

{

InitializeComponent();

this.str = str;

}

private void button1\_Click(object sender, EventArgs e)

{

textBox1.Text = str;

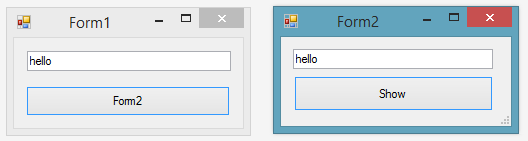
}

}

---------------------------------------------------------------------------

Недолік даного методу в тому, що передати дані дочірній формі можна тільки один раз в момент створення форми.

Передача даних через додаткову функцію або властивість дочірньої форми:



public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Form2 f = new Form2();

f.Str = textBox1.Text;

f.ShowDialog();

}

}

----------------------------------------------------------------------------

public partial class Form2 : Form

{

string str;

public Form2()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

textBox1.Text = str;

}

public string Str

{

set

{

str = value;

}

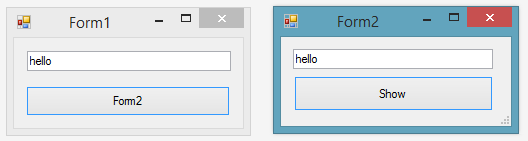
}

}

----------------------------------------------------------------------------

Недолік даного методу в тому, що можна створити і викликати форму не присвоївши властивості ніякого значення, що може спричинити помилки на під час виконання.

Передача даних через перевантаження метода ShowDialog():



public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Form2 f = new Form2();

f.ShowDialog(textBox1.Text);

}

}

----------------------------------------------------------------------------

public partial class Form2 : Form

{

string str;

public Form2()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

textBox1.Text = str;

}

public DialogResult ShowDialog(string str)

{

this.str = str;

return ShowDialog();

}

}

----------------------------------------------------------------------------

Недолік даного методу в тому, що можна викликати перевантаження метода ShowDialog без параметрів, що може спричинити помилки на етапі виконання.

**Передача даних від дочірньої до батьківської форми**

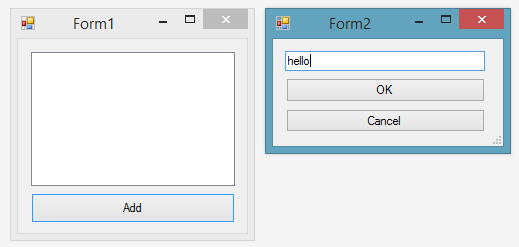
Передача даних від дочірньої до батьківської форми трохи складніша, оскільки дочірня форма не містить ссилки на батьківську форму.

Передати дані батьківській формі можна наступними способами:

* Створити додаткову функцію або властивість на дочірній формі
* Передати ссилку на форму або інший об’єкт в конструктор дочірньої форми або в перевантажений метод ShowDialog.

Також батьківській формі часто потрібно знати чи підтвердив користувач свій вибір, чи відмовився від нього. Наприклад, в додатковій формі відбувається добавлення нового товару. Користувач відкрив форму, але передумав добавляти новий товар. Якраз для таких цілей функція ShowDialog повертає перечислення **DialogResult**.

Передача даних за допомогою додаткової властивості дочірньої форми:



public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Form2 f = new Form2();

DialogResult dr = f.ShowDialog();

if (dr == DialogResult.OK)

{

listBox1.Items.Add(f.Str);

}

}

}

----------------------------------------------------------------------------

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

public string Str

{

get { return textBox1.Text; }

}

private void button1\_Click(object sender, EventArgs e)

{

this.DialogResult = DialogResult.OK;

}

private void button2\_Click(object sender, EventArgs e)

{

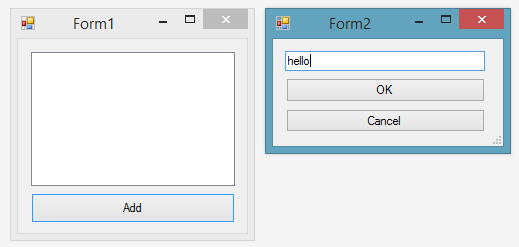
this.DialogResult = DialogResult.Cancel;

}

}

----------------------------------------------------------------------------

Передача даних за допомогою ссилки на форму або інший об’єкт в конструктор дочірньої форми



private void button1\_Click(object sender, EventArgs e)

{

Form2 f = new Form2(listBox1);

f.ShowDialog();

}

----------------------------------------------------------------------------

public partial class Form2 : Form

{

ListBox lb;

public Form2()

{

InitializeComponent();

}

public Form2(ListBox lb)

{

InitializeComponent();

this.lb = lb;

}

private void button1\_Click(object sender, EventArgs e)

{

this.DialogResult = DialogResult.OK;

lb.Items.Add(textBox1.Text);

}

private void button2\_Click(object sender, EventArgs e)

{

this.DialogResult = DialogResult.Cancel;

}

}

----------------------------------------------------------------------------

**Створення немодальної форми**

Для виклику немодальної форми потрібно створити екземпляр другої форми і викликати для неї метод Show.



Потрібно розуміти, що функція, всередині якої буде розташований даний текст, не буде чекати закриття другої форми і продовжить свою роботу. Це може привести до того, що ссилка на другу форму буде знищена. Тому, при виклику немодальної форми не рекомендується створювати локальні ссилки.

