Лабораторна робота № 26

Тема. Використання конструкторів з параметрами для організації взаємодії форм

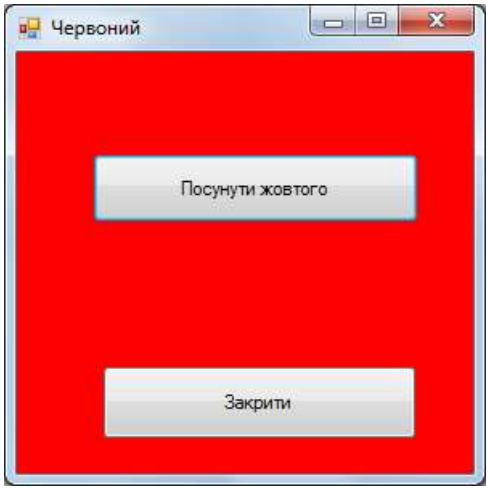
Мета. Формування вмінь і навиків створення та використання форм і їх компонентів. Закріплення вмінь і навиків використання властивостей та методів об'єктів. Застосування вмінь і навиків програмування алгоритмів лінійної, розгалуженої та циклічної структури.

Контрольні запитання

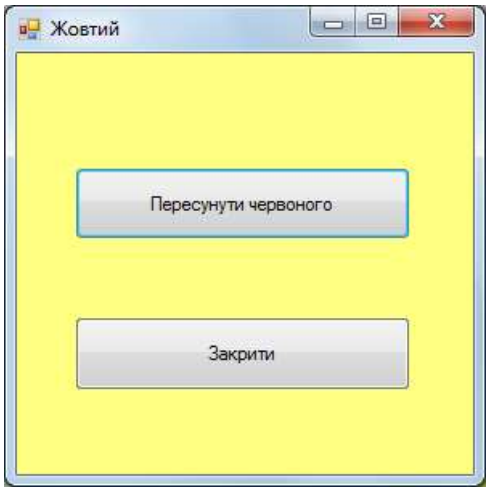
1. Як в проект Windows Form додати ще одну форму? Як змінити ім'я форми?
2. Коли використовуються конструктори з параметрами?
3. Коли для класу створюють декілька конструкторів?
4. Які змінні описуються в методі, які – в класі, а які – в просторі імен?
5. Чому в першій формі перевіряється існування другої перед пересуванням, а в другій формі існування першої форми не перевіряється?

Завдання

1. **Ознайомтеся з двома першими розділами навчального курсу по створенню Windows Forms на сайті https://metanit.com/, починаючи, відповідно, з сторінок https://metanit.com/sharp/windowsforms/1.1.php та https://metanit.com/sharp/windowsforms/2.1.php.**
2. **Для організації взаємодії двох форм створіть спочатку проект Windows Form та дайте йому назву, яка б містила ваше прізвище латинськими літерами та номер лабораторної без пробілів.**
3. **Переробіть першу форму за зразком, наведеним нижче, причому так, щоб колір форми задавався програмно рядком `this.BackColor = Color.Red;` . В процедурі обробки події натиснення другої кнопки для забезпечення завершення роботи програми викличте метод *close();*. Встановіть як назву файлів, так і назву класу форми *FormRed*. Забезпечте початкове відображення цієї форми по центру екрана.**



4. **Додайте до проекту ще одну форму, обираючи послідовно в контекстному меню назви проекту в оглядачі рішень пункти *Добавить – Форма WindowsForm*. Оформіть другу форму за зразком, наведеним нижче, причому так, щоб колір форми задавався в режимі конструктора. В процедурі обробки події натиснення другої кнопки для забезпечення закриття форми викличте метод *close();*. Встановіть як назву файлів, так і назву класу цієї форми *FormYellow*.**



5. **Оскільки друга форма має впливати на першу, то створіть в ній посилання на першу форму, значення в яке занесіть при створенні форми. Використовуючи це посилання, забезпечте пересування першої форми з другої при натисненні її першої кнопки. При цьому код другої форми має бути приблизно таким:**

```
public partial class FormYellow : Form
{
    FormRed FR;
    public FormYellow(FormRed F1)
    {   InitializeComponent();
        FR = F1;
    }

    private void button1_Click(object sender, EventArgs e)
    {   FR.Left += 10;
    }
}
```

6. **Оскільки перша форма має впливати на другу, то створіть в ній посилання на другу форму, значення в яке занесіть при створенні цієї форми в процесі створення першої форми. Використовуючи це посилання, забезпечте пересування другої форми з першої при натисненні її першої кнопки. При цьому код першої форми має бути приблизно таким:**

```
public partial class FormRed : Form
{
    FormYellow FY;

    public FormRed()
    { InitializeComponent();
      this.BackColor = Color.Red;
      FY=new FormYellow(this);
      FY.Show(); }

    private void buttonClose_Click(object sender, EventArgs e)
    { Close(); }

    private void button1_Click(object sender, EventArgs e)
    { if (FY.IsDisposed)
       {FY = new FormYellow(this);
        FY.Show();
       }
      FY.Left+=10; }
}
```

7. **Самостійно забезпечте інші впливи форм одна на одну, наприклад стосовно зміни їх розміру.**