

## Лабораторна робота №7. Windows Forms. Контейнер Panel. Створення додатку Labyrynt

**Тема:** Контейнер Panel. Створення додатку Labyrynt.

**Мета заняття:** навчитись працювати з контейнерами GroupBox, Panel і FlowLayoutPanel, навчитись задавати розміри елементів та їх позиціонування в контейнері, закріпити навички роботи з елементом Label та обробниками подій MouseEnter, MouseLeave у середовищі Microsoft Visual Studio.

### 1. Теоретичні відомості

#### 1.1 Елементи GroupBox, Panel і FlowLayoutPanel

GroupBox є спеціальним контейнером, який обмежений від решти форми кордоном. Він має заголовок, який встановлюється через властивість Text. Щоб зробити GroupBox без заголовка значення властивості Text має бути порожньою. Нерідко цей елемент використовується для групування перемикачів - елементів RadioButton, так як дозволяє розмежувати їх групи (рис. 1.1).

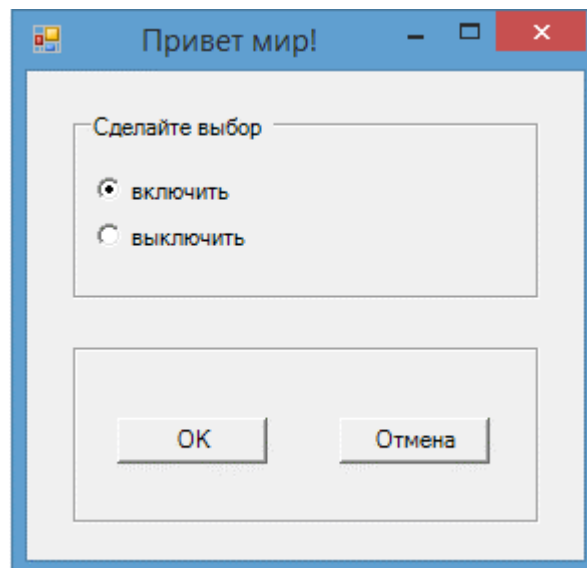


Рисунок 1.1 – Приклад застосування контейнера GroupBox

Елемент Panel являє панель і також, як і GroupBox, об'єднує елементи в групи. Вона може візуально зливатися з іншою формою, якщо вона має те ж значення кольору фону у властивості BackColor, що і форма. Щоб її виділити можна задати властивість BorderStyle, яка за замовчуванням має значення None (відсутність межі).

Також якщо панель має багато елементів, які виходять за її межі, можемо зробити щоб панель прокручувалась, встановивши її властивість **AutoScroll** в **True** (рис. 1.2).

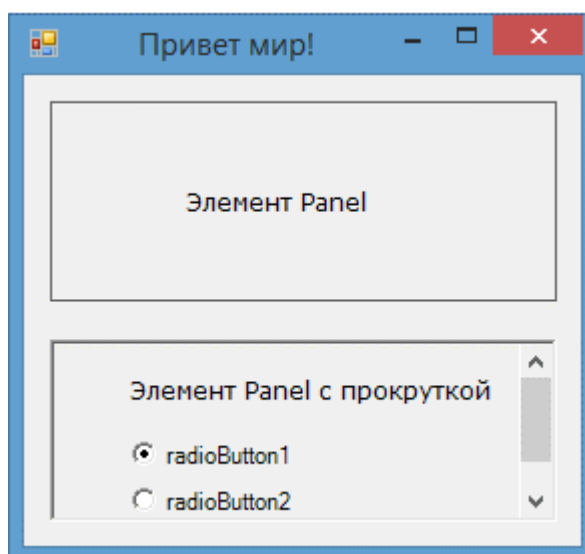


Рисунок 1.2 – Приклад застосування контейнера Panel

Також, як і форма, **GroupBox** та **Panel** мають колекції елементів, які можна динамічно додавати в ці контейнери. Наприклад, на формі є елемент **GroupBox**, який має ім'я **groupBox1**:

```
1 private void Form1_Load(object sender, EventArgs e)
2 {
3     Button helloButton = new Button();
4     helloButton.BackColor = Color.LightGray;
5     helloButton.ForeColor = Color.Red;
6     helloButton.Location = new Point(30, 30);
7     helloButton.Text = "Привет";
8     groupBox1.Controls.Add(helloButton);
9 }
```

Для вказівки розташування елемента в контейнері використовується структуру **Point**: **new Point(30, 30)**, який в конструкторі передаємо розміщення по осях **X** та **Y**. Ці координати встановлюються щодо лівого верхнього кута контейнера - в даному випадку елемента **GroupBox**.

При цьому треба враховувати, що контейнером верхнього рівня є форма, а елемент **groupBox1** сам знаходиться в колекції елементів форми. І при бажанні ми могли б видалити його:

```
1 this.Controls.Remove(groupBox1);
```

Елемент **FlowLayoutPanel** є успадкований від класу **Panel**, і тому наслідує всі його властивості. Однак при цьому має додаткову функціональність. Так, цей елемент дозволяє змінювати позиціонування та компоновку дочірніх елементів при зміні розмірів форми під час виконання програми.

Властивість елемента **FlowDirection** дозволяє задати напрямок, в якому спрямовані дочірні елементи. За замовчуванням має значення **LeftToRight**-тобто елементи будуть розташовуватися починаючи від лівого верхнього краю. Наступні елементи будуть йти вправо. Це властивість також може набувати таких значень: **RightToLeft** - елементи розташовуються від правого верхнього кута в лівий бік, **TopDown** - елементи розташовуються від лівого верхнього кута і йдуть вниз, **BottomUp** - елементи розташовуються від лівого нижнього кута і йдуть вгору (рис. 1.3).

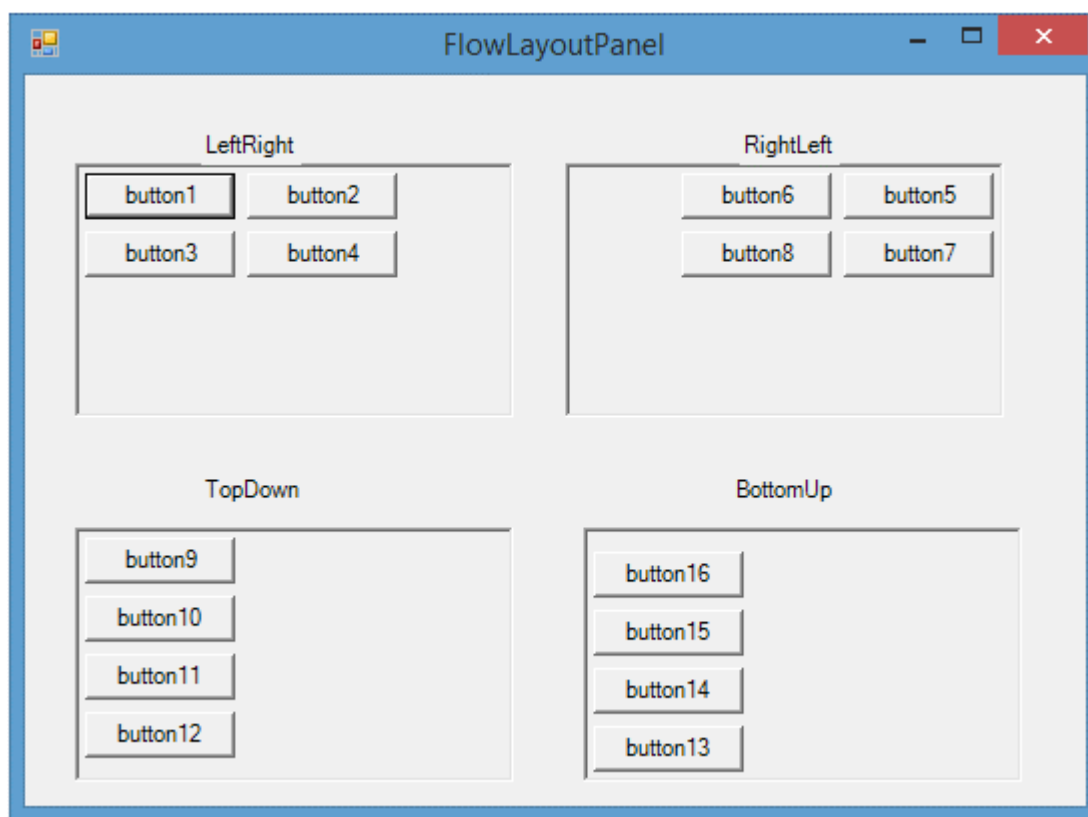


Рисунок 1.3 – Властивість **FlowDirection** контейнера **FlowLayoutPanel**

При розташуванні елементів важливу роль відіграє властивість **WrapContents**. За замовчуванням воно має значення **True**. Це дозволяє переносити елементи, які не поміщаються в **FlowLayoutPanel**, на новий рядок або в новий стовпець. Якщо вона має значення **False**, то елементи не переносяться, а до контейнера додаються смуги прокрутки, якщо властивість **AutoScroll** встановлена в **true**.

Фрагмент коду для динамічного створення елемента **Panel**, а в ньому **textbox**:

```
Panel p = new Panel();  
p.Controls.Add(new TextBox());
```

## 1.2 Розміри елементів і їх позиціонування в контейнері позиціонування

Для кожного елемента керування можемо визначити властивість **Location**, яке задає координати верхнього лівого кута елемента щодо контейнера. При перенесенні елемента з панелі інструментів на форму ця властивість встановлюється автоматично. Однак потім у вікні Властивостей можемо вручну поправити координати положення елемента (рис. 1.4):

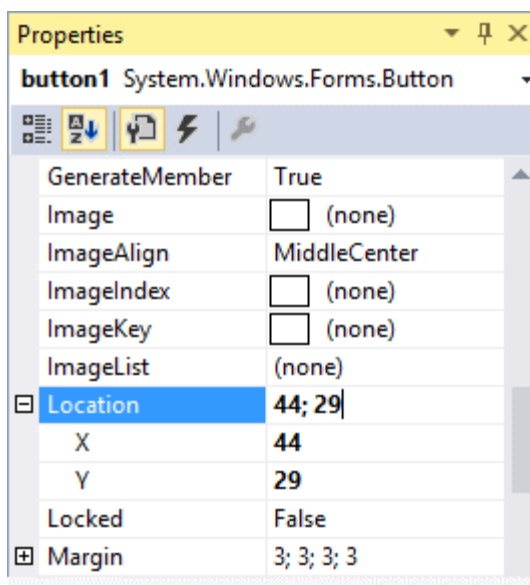


Рисунок 1.4 – Властивість **Location** елемента керування

Також ми можемо встановити позицію елемента в коді:

```
1 private void Form1_Load(object sender, EventArgs e)
2 {
3     button1.Location = new Point(50, 50);
4 }
```

За допомогою властивості **Size** можна задати розміри елемента (рис. 1.5):

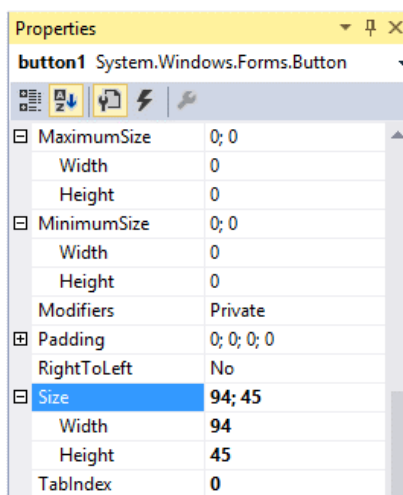


Рисунок 1.5 – Властивість **Size** елемента керування

Додаткові властивості **MaximumSize** та **MinimumSize** дозволяють обмежити мінімальний та максимальний розміри. Установка властивостей в кодї:

```
1 button1.Size = new Size { Width = 50, Height = 25 };
2 // установка свойств по отдельности
3 button1.Width = 100;
4 button1.Height = 35;
```

Додаткові можливості по позиціонуванні елемента дозволяє визначити властивість **Anchor**. Це властивість визначає відстань між однією зі сторін елемента і стороною контейнера. Якщо при роботі з контейнером будемо його розтягувати, то разом з ним буде розтягуватися і вкладений елемент. Автоматично у кожного елемента додається атрибут властивості **Top**, **Left** (рис. 1.6):

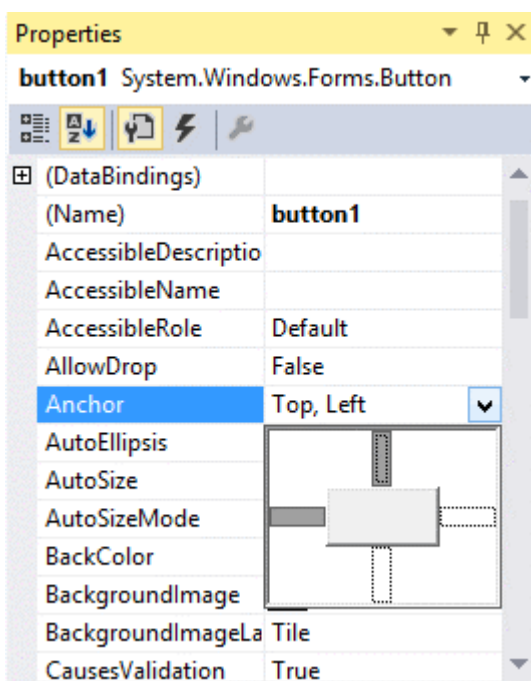


Рисунок 1.6 – Позиціонування елемента керування властивістю **Anchor**

Це означає, що якщо будемо розтягувати форму вліво або вгору, то елемент збереже відстань від лівої і верхньої межі елемента до меж контейнера, в якості якого виступає форма. Можливо задати чотири можливих значення для цієї властивості або їх комбінацію: **Top**, **Bottom**, **Left**, **Right**. Наприклад, якщо змінимо значення цієї властивості на протилежне - **Bottom**, **Right**, буде незмінною відстань між правою і нижньою стороною елемента та формою. При цьому треба відзначити, що дана властивість враховує відстань до меж контейнера, а не форми. Тобто якщо на формі є елемент **Panel**, а на **Panel** розташована кнопка, то на кнопку впливатиме зміна меж **Panel**, а не форми. Розтягування форми буде в цьому випадку впливати тільки, якщо вона впливає

на контейнер **Panel**. Щоб задати цю властивість в коді, треба використовувати перерахування **AnchorStyles** :

```
1 button1.Anchor = AnchorStyles.Left;
2 // задаємо комбінацію значень
3 button1.Anchor = AnchorStyles.Left | AnchorStyles.Top;
```

Властивість **Dock** дозволяє прикріпити елемент до певної сторони контейнера. За замовчуванням вона має значення **None** (рис. 1.7), але також можна задати ще п'ять значень: **Top** – елемент притискається до верхньої межі контейнера, **Bottom** – елемент притискається до нижньої межі контейнера, **Left** – елемент притискається до лівого боку контейнера, **Right** – елемент прикріплюється до правого боку контейнера, **Fill** – елемент заповнює весь простір контейнера.

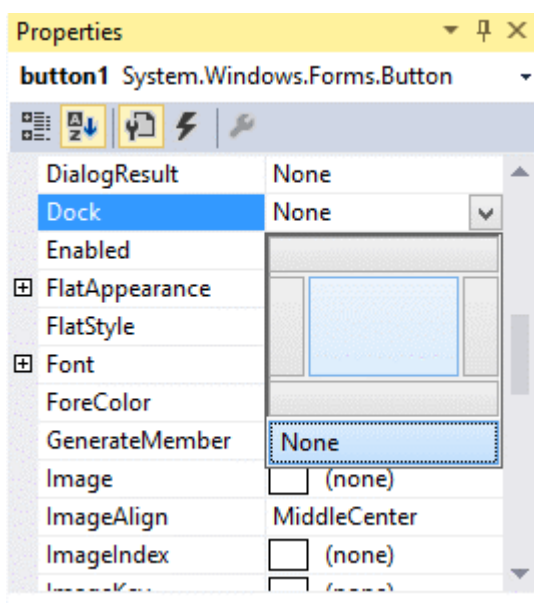


Рисунок 1.7 – Властивість **Dock** елемента керування

## 2. Практичне завдання



У процесі виконання завдань лабораторної роботи необхідно формувати набори тестових даних для перевірки правильності виконання програмного коду. Створений код і результати перевірки його роботи потрібно помістити у звіт. Тестувати роботу програми рекомендується після додання чи зміни кожного оператора виведення.

### Завдання 1. Створити проект для розв'язання задачі.

*При старті програми, курсор повинний бути на старті, якщо курсор доторкається стінки лабіринту (внутрішньої або зовнішньої), то його перекидає на початок.*

1. Створюємо проект **winForm** з назвою **Labyrinth**. Розмір форми довільний.

2. Властивість **Text** форми змінюємо на «Лабіринт».
3. Додаємо на форму елемент «**Panel**», розтягуємо по краям так, щоб був невеликий зазор.
4. Властивість **FormBorderStyle** виставляємо значення **Fixed3D** (забороняє зміну розмірів форми користувачем).
5. Відключаємо кнопку «Розгорнути» заголовку вікна. Для цього встановлюємо властивість форми **MaximizeBox** в значення **False**.
6. Виділяємо панель та встановлюємо властивість **BorderStyle** значення **Fixed3D**. Це необхідно, щоб добре була видима межа поля гри.
7. Додаємо на форму елемент **Label**. Властивості **AutoSize** встановлюємо значення **False**. Властивості **BackColor** задаємо колір **SkyBlue**. Видаляємо значення властивості **Text**.
8. Виділяємо елемент **Label** та робимо множину копій (Ctrl+C — Ctrl+V). З цих елементів будуюмо лабіринт (рис. 2.1).

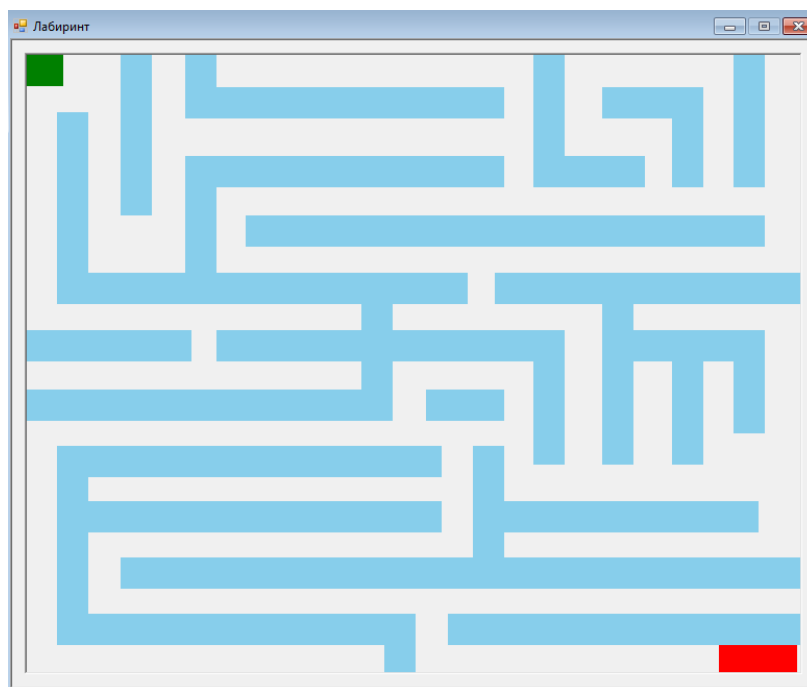


Рисунок 2.1 – Зовнішній вигляд додатку, що проектується

9. Додаємо **Label** фініш: властивість **Name** – **labelFinish**, **Text** – «Фініш», колір червоний. Додаємо **Label** старт: властивість **Name** – **labelStart**, **Text** – «Старт», колір – зелений.
10. Виділяємо усі сині **label** та створюємо обробник події **MouseEnter** подвійним натисненням (рис. 2.2).

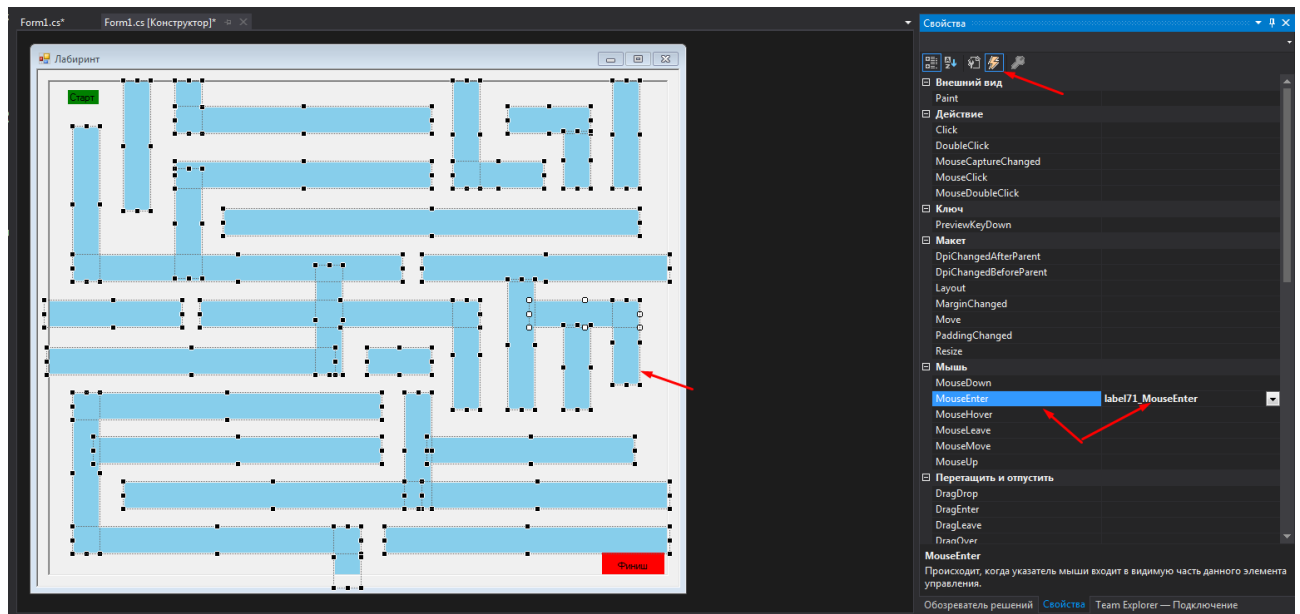


Рисунок 2.2 – Створення обробника події MouseEnter для елементів label

11. На формі виділяємо тільки елемент **Panel** та створюємо обробник події **MouseLeave** і вказуємо подію **Label71\_MouseEnter** (рис. 2.3).

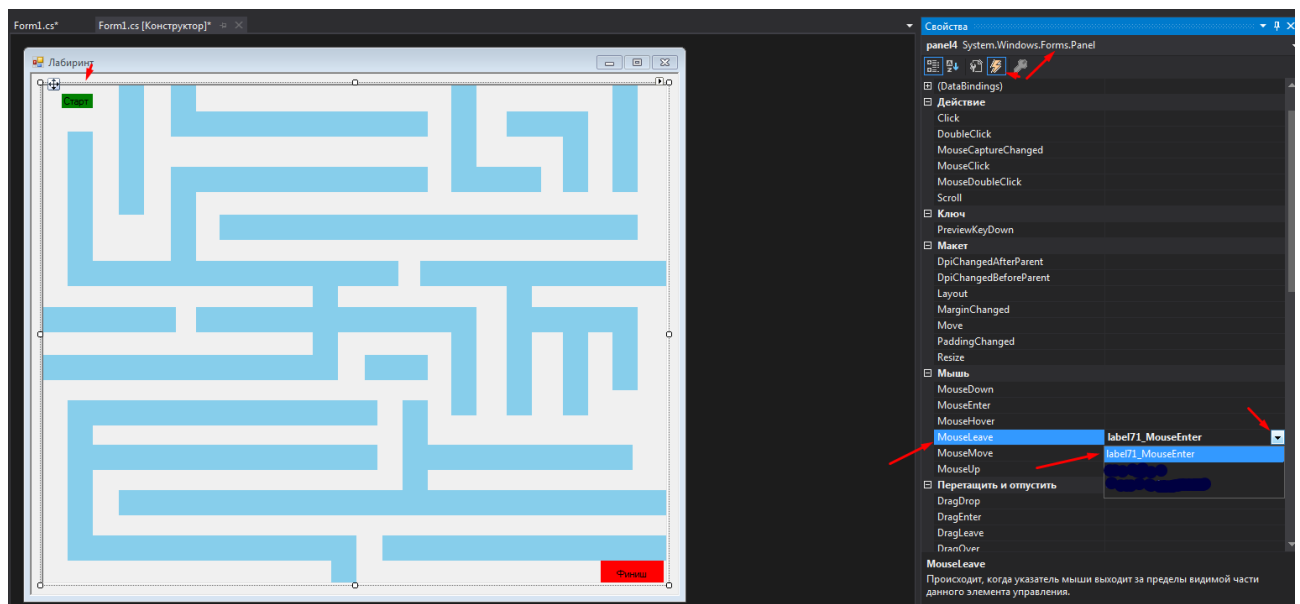


Рисунок 2.3 – Створення обробника події MouseLeave для елемента Panel

12. Створюємо обробник для форми **Load**, який необхідний щоб в момент запуску форми задати позицію курсору (рис. 2.4).



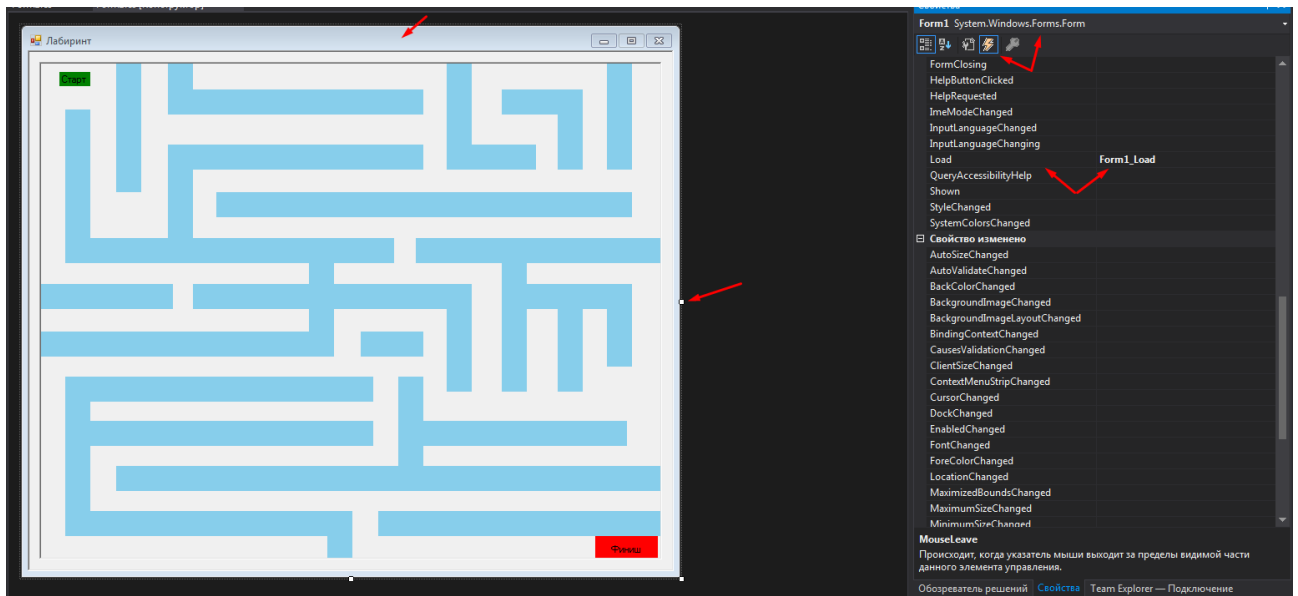


Рисунок 2.4 – Створення обробника події Load для елемента Form

13 Виділяємо елемент **labelFinish** та створюємо обробник події **MouseEnter** для нього (рис. 2.5).

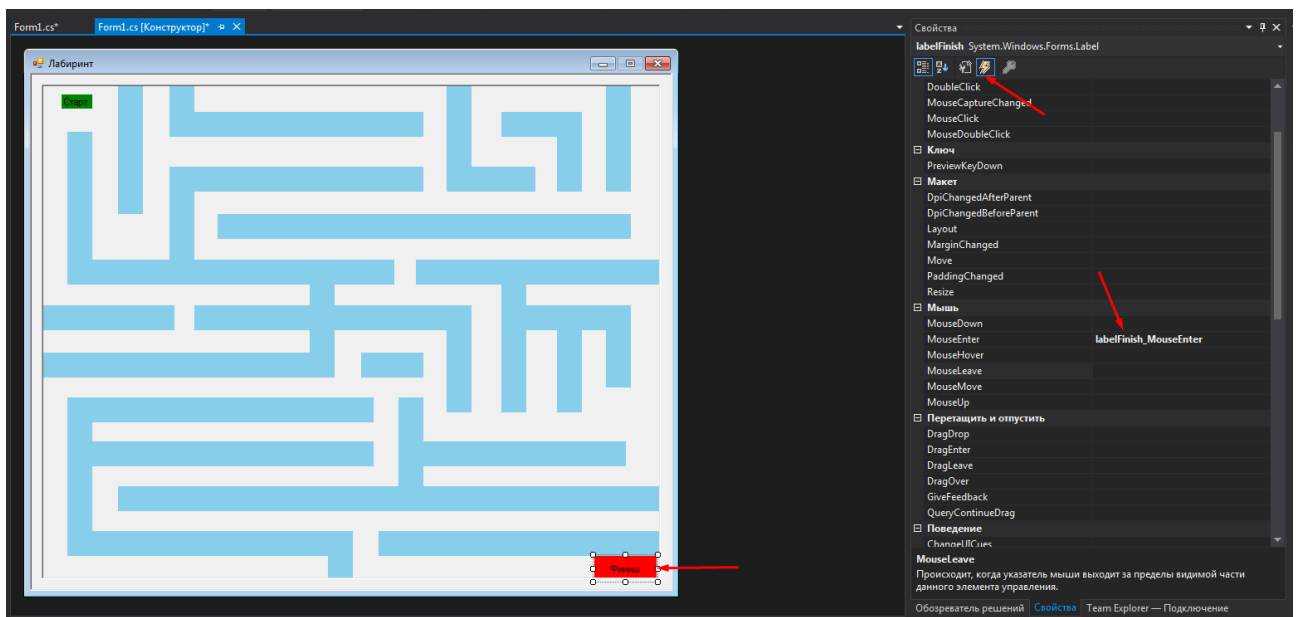


Рисунок 2.5 – Обробник події MouseEnter элементу labelFinish

### Лістинг програмного коду проекту

```
using System;
using System.Drawing;
using System.Windows.Forms;

namespace Labyrinth
{
    public partial class Form1 : Form
    {
```

```

public Form1()
{
    InitializeComponent();
}

//При доторканні до стіни лабіринту, позиція буде оновлюватись до стартової.

private void label71_MouseEnter(object sender, EventArgs e)
{
    Cursor.Position = new Point(this.Location.X + 55, this.Location.Y + 60);
}

//Виставляємо стартову позицію курсору.
// +55 и +60 це позиція labelStart відносно позиції форми на робочому столі.

private void Form1_Load(object sender, EventArgs e)
{
    Cursor.Position = new Point(this.Location.X + 55, this.Location.Y + 60);
}

//При наведенні на Finish появиться повідомлення
private void labelFinish_MouseEnter(object sender, EventArgs e)
{
    MessageBox.Show("Перемора!!!", "Ура!!!", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
}
}

```

### 3. Контрольні запитання

1. Призначення контейнерів GroupBox, Panel, FlowLayoutPanel. У чому різниця їх застосування
2. Як задати розміри елементів та їх позиціонування в контейнері?
3. Яку призначення властивостей: Location, Size, MaximumSize, MinimumSize, Anchor, Dock
4. Для чого використовується властивість FormBorderStyle елемента Panel?
5. Для чого використовується властивість BorderStyle елемента Panel?
6. Для чого використовується властивість MaximizeBox елемента Form?
7. Для чого використовується атрибут Fixed3D властивостей FormBorderStyle, BorderStyle елемента Panel?
8. Як вибрати декілька елементів Label?
9. Для чого використовується обробник події MouseEnter? Як додати таку

- подію в проект?
10. Для чого використовується обробник події MouseLeave? Як додати таку подію в проект?
  11. Які існують події при роботі з PictureBox? Як їх налаштувати?
  12. Для чого використовується обробник події форми Load?

### **Література**

1. Евдокимов П. В. С# на примерах. СПб.: Наука и Техника, 2019. 320 с.
2. Маки А. Введение в .NET 4.0 и Visual Studio 2010 для профессионалов; пер. с англ. М. : ООО ИД "Вильямс", 2010. 416 с
3. С# 7.0. Справочник. Полное описание языка.: Пер. с англ. СПб.: ООО "Альфа-книга", 2018. 1024 с.
4. Троелсен, Эндрю, Джепикс, Филипп. Язык программирования С# 7 и платформы .NET и .NET Core. СПб. : ООО "Диалектика", 2018. 1328 с.
5. Офіційний сайт компанії Microsoft щодо технологій WPF та Win-dows Forms [Електронний ресурс]. – Режим доступу : <http://window-sclient.net>.
6. С#. Теорія та практика. URL: [https://www.bestprog.net/uk/sitemap\\_ua/c-3](https://www.bestprog.net/uk/sitemap_ua/c-3)
7. Сажин А. Справочник по языку программирования С#. URL: <https://brainoteka.com/blogs/c-spravochnik>.
8. С# Tutorial URL <https://www.theengineeringprojects.com>.
9. Уроки С#. URL: <https://itproger.com/course/csharp>.
10. Полное руководство по С# 8 и .NET Core. URL: <https://metanit.com/sharp/>