

# Пользовательский интерфейс

## Windows Forms

Юрий Литвинов  
y.litvinov@spbu.ru

04.04.2025

# Windows Forms

- ▶ Самая старая из GUI-библиотек для .NET
- ▶ Событийно-ориентированная
- ▶ Простая
- ▶ Давно не развивается, но поддерживается и используется до сих пор

# Простой пример

```
public class MyForm : Form {
    private Button button1;

    public MyForm()
    {
        button1 = new Button();
        button1.Size = new Size(40, 40);
        button1.Location = new Point(30, 30);
        button1.Text = "Click me";
        this.Controls.Add(button1);
        button1.Click += new EventHandler(Button1Click);
    }

    private void Button1Click(object sender, EventArgs e)
        => MessageBox.Show("Hello World");

    [STAThread]
    static void Main()
    {
        Application.EnableVisualStyles();
        Application.Run(new MyForm());
    }
}
```

# Control

- ▶ Корень иерархии элементов управления
- ▶ Отвечает за пользовательский ввод, события, позицию на экране
  - ▶ В том числе, информацию для лейаутов
- ▶ Ambient property — свойство, наследуемое от родителя
  - ▶ Cursor, Font, BackColor, ForeColor и RightToLeft
- ▶ Controls — коллекция дочерних контролов

# Обработка пользовательского ввода

## ► События с клавиатуры

- KeyDown, KeyPress, KeyUp
- PreProcessMessage, WndProc

```
private void textBox1_KeyDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.F1 && (e.Alt || e.Control || e.Shift))
    {
        Help.ShowPopup(textBox1, "Enter your name.",
            new Point(textBox1.Bottom, textBox1.Right));
    }
}
```

## ► События мыши

- MouseDown, Click, MouseClick, MouseUp
- Или MouseDown, Click, MouseClick, MouseUp, MouseDown, DoubleClick, MouseDoubleClick, MouseUp

Демонстрация

# Демонстрация

# Best practices

- ▶ Отделяйте логику работы программы от представления
  - ▶ Писать логику прямо в обработчиках совсем плохо
  - ▶ Уровневая архитектура
  - ▶ Model-View-Presenter
    - ▶ Может быть overkill-ом для небольших проектов
- ▶ Форма должна адекватно вести себя при ресайзе
- ▶ Пользовательский интерфейс не должен ломать привычек пользователя
- ▶ Самые часто используемые элементы интерфейса должны находиться ближе всего к рабочей области
- ▶ Не следует думать, что пользователи будут читать документацию

# Валидация ввода

- ▶ Простой способ — `MaskedTextBox`
- ▶ “Правильный” способ — событие *Validating*
  - ▶ `CancelEventArgs`, свойство `Cancel`
  - ▶ Последовательность событий при потере фокуса: `Leave`, `Validating`, `Validated`, `LostFocus`
  - ▶ Свойство `AutoValidate`
  - ▶ Можно инициировать вручную, методом `Validate` или `ValidateChildren`



# Data binding

```
public partial class Form1 : Form
{
    public string MyText { get; set; } = "Click me";

    public Form1()
    {
        InitializeComponent();
        button1.DataBindings.Add("Text", this, "MyText");
    }
}
```