

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ПОЛТАВСЬКА ПОЛІТЕХНІКА
ІМЕНІ ЮРІЯ КОНДРАТЮКА»**

**Навчально-науковий інститут інформаційних технологій і робототехніки
 Кафедра комп'ютерних та інформаційних технологій і систем**

i

Лабораторна робота № 1
з навчальної дисципліни
«Технології на платформі NET»

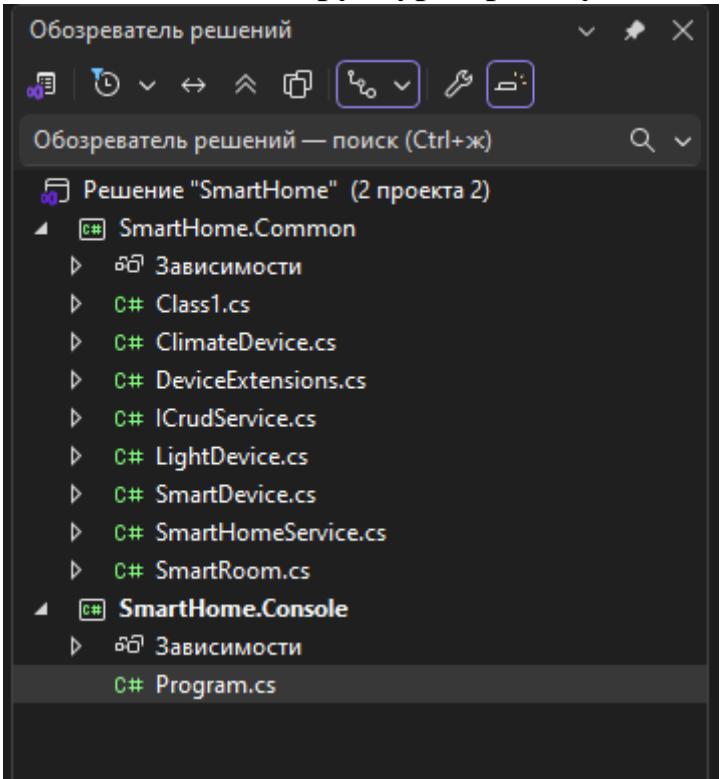
Виконав:
*Студент групи 403-TH
Солонецький Роман Миколайович*

Перевірив:
Тютюнник Петро Богданович

Полтава 2025

Тема: Основи програмування у мові C#. ООП у мові C#. Базові конструкції та Типи

Файлова структура проекту



Обозреватель решений

Обозреватель решений — поиск (Ctrl+ж)

Решение "SmartHome" (2 проекта)

- SmartHome.Common
 - Зависимости
 - Class1.cs
 - ClimateDevice.cs
 - DeviceExtensions.cs
 - ICrudService.cs
 - LightDevice.cs
 - SmartDevice.cs
 - SmartHomeService.cs
 - SmartRoom.cs
- SmartHome.Console
 - Зависимости
 - Program.cs

Сортировать

Имя	Дата изменения	Тип	Размер
.git	28.12.2025 23:42	Папка с файлами	
SmartHome.Common	28.12.2025 23:39	Папка с файлами	
SmartHome.Console	28.12.2025 23:39	Папка с файлами	
.gitignore	27.12.2025 23:33	txtfile	8 КБ
SmartHome.slnx	28.12.2025 23:18	VisualStudio.Laun...	1 КБ

Реалізація базових конструкцій C#

The screenshot shows a code editor with two tabs open: `LightDevice.cs` and `SmartRoom.cs`. Both files are part of the `SmartHome.Common` namespace.

LightDevice.cs:

```
namespace SmartHome.Common;
public class LightDevice : SmartDevice
{
    public int Brightness { get; set; }
    public string Color { get; set; }
    public string BulbType { get; set; }

    public LightDevice() : base() { }

    public LightDevice(string name, int brightness = 50) : base(name)
    {
        Brightness = brightness;
        Color = "White";
        BulbType = "LED";
    }
}
```

SmartRoom.cs:

```
namespace SmartHome.Common;
public class SmartRoom
{
    public Guid Id { get; set; } = Guid.NewGuid();
    public string RoomName { get; set; }
    public int Floor { get; set; }
    public double Area { get; set; }

    public SmartRoom(string name, int floor, double area)
    {
        RoomName = name;
        Floor = floor;
        Area = area;
    }
}
```

Реалізація Generic CRUD сервісу

The screenshot shows a code editor with several tabs at the top: ICrudService.cs*, SmartRoom.cs, LightDevice.cs, and SmartRoomService.cs. The ICrudService.cs* tab is active, displaying the following interface definition:

```
namespace SmartHome.Common;
public interface ICrudService<T>
{
    void Create(T element);
    T Read(Guid id);
    IEnumerable<T> ReadAll();
    void Update(T element);
    void Remove(T element);
}
```

Below this, the SmartRoomService.cs tab is shown, containing the concrete implementation of the service:

```
public class SmartHomeService<T> : ICrudService<T> where T : class
{
    private List<T> _items = new List<T>();

    public void Create(T element) => _items.Add(element);

    public IEnumerable<T> ReadAll() => _items;

    public T Read(Guid id)
    {
        return _items.FirstOrDefault(x => (Guid)x.GetType().GetProperty("Id")?.GetValue(x) == id);
    }

    public void Update(T element) { }

    public void Remove(T element) => _items.Remove(element);
}
```

Реалізовано сервіс, що працює з вбудованою колекцією `List<T>` та забезпечує повний цикл роботи з даними

Результати виконання додаткового завдання

```
// Додаткове завдання: Збереження у файл
public void Save(string filePath)
{
    string json = JsonSerializer.Serialize(_items);
    File.WriteAllText(filePath, json);
}

// Додаткове завдання: Завантаження з файлу
public void Load(string filePath)
{
    if (File.Exists(filePath))
    {
        string json = File.ReadAllText(filePath);
        // Додаємо обробку випадку, коли файл пустий
        _items = JsonSerializer.Deserialize<List<T>>(json) ?? new List<T>();
        Console.WriteLine("Дані успішно завантажені.");
    }
}
```

Реалізовано збереження даних у форматі JSON та їх завантаження з файлу для забезпечення постійного зберігання стану системи.

Робота з Git та GitHub

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).
Required fields are marked with an asterisk (*).

1 General

Owner * **1issue** / Repository name * **NUPP_NET_2025_403_TN_Solonetskyi_Lab**
NUPP_NET_2025_403_TN_Solonetskyi_Lab is available.

Great repository names are short and memorable. How about [shiny-eureka?](#)

Description
0 / 350 characters

2 Configuration

Choose visibility * Choose who can see and commit to this repository **Public**

Add README READMEs can be used as longer descriptions. [About READMEs](#) Off

Add .gitignore .gitignore tells git which files not to track. [About ignoring files](#) VisualStudio

Add license Licenses explain how others can use your code. [About licenses](#) No license

Create repository

1issue / NUPP_NET_2025_403_TN_Solonetskyi_Lab

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

NUPP_NET_2025_403_TN_Solonetsk... Public Pin Watch 0 Fork 0 Star 0

lab1 Go to file + <> Code

This branch is 1 commit behind main .

1issue Перенесено проект SmartHome у клонований репозиторій... b213e48 · 29 minutes ago

SmartHome.Common Перенесено проект SmartHome у ... 29 minutes ago

SmartHome.Console Перенесено проект SmartHome у ... 29 minutes ago

.gitignore Initial commit yesterday

SmartHome.slnx Перенесено проект SmartHome у ... 29 minutes ago

About
No description, website, or topics provided.

Activity 0 stars 0 watching 0 forks

Releases
No releases published [Create a new release](#)

Packages
No packages published [Publish your first package](#)

Languages
C# 100.0%

Suggested workflows
Based on your tech stack

Контрольні питання

1. Що таке класи та структури? Яка різниця між класом та структурою? Яка різниця між value та reference типами

Класи — це посилальні типи (reference types), що зберігаються в купі; структури — типи значень (value types), що зберігаються в стеку.

2. Що таке абстрактний клас? Яка різниця між звичайним та абстрактним класом?

Це клас-шаблон, на основі якого не можна створити об'єкт; він може містити методи без реалізації для обов'язкового перевизначення нащадками.

3. Що таке інтерфейс? Яка різниця між абстрактним класом та інтерфейсом?

Інтерфейс — це «контракт» поведінки без стану, а клас може реалізувати багато інтерфейсів. Наслідування можливе лише від одного абстрактного класу.

4. Що таке наслідування? Які є модифікатори доступу у мові C#?

Наслідування - це механізм ООП, що дозволяє створювати новий клас на основі вже існуючого.

public, private, protected, internal та їх комбінації, що регулюють видимість елементів коду.

5. Що таке статичні поля та методи? Яка різниця між звичайними та статичними полями та методами?

Це елементи, що належать самому класу, а не його екземплярам; вони існують в єдиному екземплярі для всіх об'єктів класу.

6. Що таке делегати? Які види делегатів існують?

Це типи, що описують посилання на методи з певною сигнатурою; існують одиночні, групові та вбудовані (Action, Func, Predicate) делегати.