Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №3

з дисципліни

«Основи розробки програмного забезпечення на платформі

Microsoft.NET»

«JSON»

Виконала ІП-21 Голованьов Г.О.

Київ 2024

**Комп‘ютерний практикум № 3**

**JSON**

**Мета:** ознайомитися з обробкою даних у форматі JSON в C#.

# Опис архітектури проекту

Програма має такі класи: Commandant, Hostel, Settlement і Student.

Commandant – клас, що являє собою сутність «комендант». Містить таку інформацію: індефікаційний номер, ім’я, вік, досвід та гендер. Об’єкти даного класу можна вивести за допомогою перевантаженого методу ToString().

Student – клас, що являє собою сутність «студент». Містить таку інформацію: індефікаційний номер, ім’я, факультет, кафедра, курс,рік народження та гендер. Об’єкти даного класу можна вивести за допомогою перевантаженого методу ToString().

Hostel – клас, що являє собою сутність «гуртожиток». Містить таку інформацію: індефікаційний номер, адреса, кількість кімнат та індефікаційний номер коменданта. Об’єкти даного класу можна вивести за допомогою перевантаженого методу ToString().

Settlement – проміжний клас, що являє собою сутність «заселення». Слугує для вирішення зв’язку many-to-many між класами Student та Hostel. Містить таку інформацію: індефікаційний номер, індефікаційний номер студента, індефікаційний номер гуртожитку, час заселення та час виселення. Об’єкти даного класу можна вивести за допомогою перевантаженого методу ToString().

Зв’язки між класами:

Student та Hostel – many-to-many, вирішений за допомогою класу Settlement.

Багато студентів можуть жити в багатьох гуртожитках

Hostel та Commandant – one-to-many. Вахтер може працювати в будь-якому з гуртожитків, але в одному.

**ЕR-модель**

**A diagram of a student

Description automatically generated**

# Текст програми

**Commandant.cs**

public class Commandant

{

public int Id;

public string Name;

public int Age;

public int Experience;

public string Gender;

public override string ToString()

{

return $"Id: {Id}, Name: {Name}, Age: {Age}, Experience: {Experience} years";

}

}

**Hostel.cs**

public class Hostel

{

public int Id;

public string Address;

public int AmountOfRooms;

public int CommandantId;

public override string ToString()

{

return $"Id: {Id}, Address: {Address}, Amount of Rooms: {AmountOfRooms}, Commandant Id: {CommandantId}";

}

}

**Student.cs**

public class Student

{

public int Id;

public string Name;

public string Faculty;

public string Cathedra;

public int Course;

public string Gender;

public int YearOfBirth;

public override string ToString()

{

return $"Id: {Id}, Name: {Name}, Faculty: {Faculty}, Cathedra: {Cathedra}, Course: {Course}";

}

}

**Settlement.cs**

public class Settlement

{

public int Id;

public int StudentId;

public int HostelId;

public DateTime CheсkInTime;

public DateTime CheсkOutTime;

public override string ToString()

{

return $"Id: {Id}, StudentId: {StudentId}, HostelId: {HostelId}, CheckInTime: {CheсkInTime}, CheckOutTime: {CheсkOutTime}";

}

}

**Program.cs**

namespace net\_lab3

{

internal class Program

{

static void Main(string[] args)

{

//creating data

List<Student> students = new List<Student>

{

new Student { Id = 1, Name = "Joe", Faculty = "FICT", Cathedra = "IPI", Course = 1, Gender = "Male", YearOfBirth = 2006 },

new Student { Id = 2, Name = "Nadiya", Faculty = "FICT", Cathedra = "IPI", Course = 1, Gender = "Female", YearOfBirth = 2006 },

new Student { Id = 3, Name = "Alexandr", Faculty = "FICT", Cathedra = "IPI", Course = 2, Gender = "Male", YearOfBirth = 2005 },

new Student { Id = 4, Name = "Myhailo", Faculty = "FICT", Cathedra = "IPI", Course = 2, Gender = "Male", YearOfBirth = 2004 },

new Student { Id = 5, Name = "Oksana", Faculty = "FICT", Cathedra = "OT", Course = 3, Gender = "Female", YearOfBirth = 2002 }

};

List<Commandant> commandants = new List<Commandant>

{

new Commandant { Id = 1, Name = "Adriy", Age = 40, Experience = 5, Gender = "Male" },

new Commandant { Id = 2, Name = "Olena", Age = 45, Experience = 1, Gender = "Female" },

new Commandant { Id = 3, Name = "Sophia", Age = 60, Experience = 23, Gender = "Female" },

new Commandant { Id = 4, Name = "Oksana", Age = 70, Experience = 4, Gender = "Female" },

new Commandant { Id = 5, Name = "Maxik", Age = 43, Experience = 14, Gender = "Male" }

};

List<Hostel> hostels = new List<Hostel>

{

new Hostel { Id = 1, Address = "Boholubova", AmountOfRooms = 200, CommandantId = 5 },

new Hostel { Id = 2, Address = "Retardova", AmountOfRooms = 250, CommandantId = 4 },

new Hostel { Id = 3, Address = "Politehnichna", AmountOfRooms = 140, CommandantId = 3 },

new Hostel { Id = 4, Address = "Knushna", AmountOfRooms = 600, CommandantId = 2 },

new Hostel { Id = 5, Address = "Simonova", AmountOfRooms = 800, CommandantId = 1 }

};

List<Settlement> settlements = new List<Settlement>

{

new Settlement { Id = 1, StudentId = 2, HostelId = 3, CheсkInTime = new DateTime(2015, 12, 13), CheсkOutTime = new DateTime(2016, 1, 13) },

new Settlement { Id = 2, StudentId = 3, HostelId = 1, CheсkInTime = new DateTime(2015, 11, 12), CheсkOutTime = new DateTime(2015, 11, 12) },

new Settlement { Id = 3, StudentId = 1, HostelId = 2, CheсkInTime = new DateTime(2015, 1, 1), CheсkOutTime = new DateTime(2015, 2, 1) },

new Settlement { Id = 4, StudentId = 4, HostelId = 3, CheсkInTime = new DateTime(2015, 3, 13), CheсkOutTime = new DateTime(2016, 4, 13) },

new Settlement { Id = 5, StudentId = 5, HostelId = 1, CheсkInTime = new DateTime(2015, 5, 13), CheсkOutTime = new DateTime(2016, 7, 13) }

};

//serialize data

SerializeData(students, commandants, hostels, settlements);

//deserialize data

List<Student> deserializedStudents = DeserializeData<Student>("students.json");

List<Hostel> deserializedHostels = DeserializeData<Hostel>("hostels.json");

List<Commandant> deserializedCommandants = DeserializeData<Commandant>("commandants.json");

List<Settlement> deserializedSettlements = DeserializeData<Settlement>("settlements.json");

//display

DisplayData(deserializedStudents,deserializedCommandants,deserializedHostels,deserializedSettlements);

Console.ReadKey();

DispalayDataJsonDocument();

Console.ReadKey();

DisplayDataJsonNode();

Console.ReadKey();

}

static void DispalayDataJsonDocument()

{

// JsonDocument

Console.WriteLine("\nJsonDocument:\n");

using (JsonDocument document = JsonDocument.Parse(File.OpenRead("students.json")))

{

Console.WriteLine("Students:");

foreach (JsonElement element in document.RootElement.EnumerateArray())

{

int id = element.GetProperty("Id").GetInt32();

string name = element.GetProperty("Name").GetString();

string faculty = element.GetProperty("Faculty").GetString();

string cathedra = element.GetProperty("Cathedra").GetString();

int course = element.GetProperty("Course").GetInt32();

string gender = element.GetProperty("Gender").GetString();

int yearOfBirth = element.GetProperty("YearOfBirth").GetInt32();

Console.WriteLine($"Id: {id}, Name: {name}, Faculty: {faculty}, Cathedra: {cathedra}," +

$" Course: {course}, Gender: {gender}, Year of Birth: {yearOfBirth}");

}

}

using (JsonDocument document = JsonDocument.Parse(File.OpenRead("commandants.json")))

{

Console.WriteLine("\nCommandants:");

foreach (JsonElement element in document.RootElement.EnumerateArray())

{

int id = element.GetProperty("Id").GetInt32();

string name = element.GetProperty("Name").GetString();

int age = element.GetProperty("Age").GetInt32();

int experience = element.GetProperty("Experience").GetInt32();

string gender = element.GetProperty("Gender").GetString();

Console.WriteLine($"Id: {id}, Name: {name}, Age: {age}, Experience: {experience}, Gender: {gender}");

}

}

using (JsonDocument document = JsonDocument.Parse(File.OpenRead("hostels.json")))

{

Console.WriteLine("\nHostels:");

foreach (JsonElement element in document.RootElement.EnumerateArray())

{

int id = element.GetProperty("Id").GetInt32();

string address = element.GetProperty("Address").GetString();

int amountOfRooms = element.GetProperty("AmountOfRooms").GetInt32();

int commandantId = element.GetProperty("CommandantId").GetInt32();

Console.WriteLine($"Id: {id}, Address: {address}, AmountOfRooms: {amountOfRooms}, CommandantId: {commandantId}");

}

}

using (JsonDocument document = JsonDocument.Parse(File.OpenRead("settlements.json")))

{

Console.WriteLine("\nSettlements:");

foreach (JsonElement element in document.RootElement.EnumerateArray())

{

int id = element.GetProperty("Id").GetInt32();

int studentId = element.GetProperty("StudentId").GetInt32();

int hostelId = element.GetProperty("HostelId").GetInt32();

DateTime checkInTime = element.GetProperty("CheсkInTime").GetDateTime();

DateTime checkOutTime = element.GetProperty("CheсkOutTime").GetDateTime();

Console.WriteLine($"Id: {id}, StudentId: {studentId}, HostelId: {hostelId}, CheckInTime: {checkInTime}, CheckOutTime: {checkOutTime}");

}

}

}

static void DisplayDataJsonNode()

{

Console.WriteLine("\nJsonNode:\n");

// Читання та обробка файлу students.json

string studentsJson = File.ReadAllText("students.json");

JsonNode studentsNode = JsonNode.Parse(studentsJson);

Console.WriteLine("Students:");

foreach (JsonNode studentNode in studentsNode.AsArray())

{

Console.WriteLine($"\nId: {studentNode["Id"].GetValue<int>()}");

Console.WriteLine($"Name: {studentNode["Name"].GetValue<string>()}");

Console.WriteLine($"Faculty: {studentNode["Faculty"].GetValue<string>()}");

Console.WriteLine($"Cathedra: {studentNode["Cathedra"].GetValue<string>()}");

Console.WriteLine($"Course: {studentNode["Course"].GetValue<int>()}");

Console.WriteLine($"Gender: {studentNode["Gender"].GetValue<string>()}");

Console.WriteLine($"Year of Birth: {studentNode["YearOfBirth"].GetValue<int>()}");

}

// Читання та обробка файлу hostels.json

string hostelsJson = File.ReadAllText("hostels.json");

JsonNode hostelsNode = JsonNode.Parse(hostelsJson);

Console.WriteLine("\nHostels:");

foreach (JsonNode hostelNode in hostelsNode.AsArray())

{

Console.WriteLine($"\nId: {hostelNode["Id"].GetValue<int>()}");

Console.WriteLine($"Address: {hostelNode["Address"].GetValue<string>()}");

Console.WriteLine($"Amount of Rooms: {hostelNode["AmountOfRooms"].GetValue<int>()}");

Console.WriteLine($"Commandant Id: {hostelNode["CommandantId"].GetValue<int>()}");

}

// Читання та обробка файлу settlements.json

string settlementsJson = File.ReadAllText("settlements.json");

JsonNode settlementsNode = JsonNode.Parse(settlementsJson);

Console.WriteLine("\nSettlements:");

foreach (JsonNode settlementNode in settlementsNode.AsArray())

{

Console.WriteLine($"\nId: {settlementNode["Id"].GetValue<int>()}");

Console.WriteLine($"Student Id: {settlementNode["StudentId"].GetValue<int>()}");

Console.WriteLine($"Hostel Id: {settlementNode["HostelId"].GetValue<int>()}");

Console.WriteLine($"Check In Time: {settlementNode["CheсkInTime"].GetValue<DateTime>()}");

Console.WriteLine($"Check Out Time: {settlementNode["CheсkOutTime"].GetValue<DateTime>()}");

}

// Читання та обробка файлу commandants.json

string commandantsJson = File.ReadAllText("commandants.json");

JsonNode commandantsNode = JsonNode.Parse(commandantsJson);

Console.WriteLine("\nCommandants:");

foreach (JsonNode commandantNode in commandantsNode.AsArray())

{

Console.WriteLine($"\nId: {commandantNode["Id"].GetValue<int>()}");

Console.WriteLine($"Name: {commandantNode["Name"].GetValue<string>()}");

Console.WriteLine($"Age: {commandantNode["Age"].GetValue<int>()}");

Console.WriteLine($"Experience: {commandantNode["Experience"].GetValue<int>()}");

Console.WriteLine($"Gender: {commandantNode["Gender"].GetValue<string>()}");

}

}

static void SerializeData(List<Student> students, List<Commandant> commandants,

List<Hostel> hostels, List<Settlement> settlements)

{

string studentsJson = JsonConvert.SerializeObject(students, Newtonsoft.Json.Formatting.Indented);

File.WriteAllText("students.json", studentsJson);

string commandantsJson = JsonConvert.SerializeObject(commandants, Newtonsoft.Json.Formatting.Indented);

File.WriteAllText("commandants.json", commandantsJson);

string hostelsJson = JsonConvert.SerializeObject(hostels, Newtonsoft.Json.Formatting.Indented);

File.WriteAllText("hostels.json", hostelsJson);

string settlementsJson = JsonConvert.SerializeObject(settlements, Newtonsoft.Json.Formatting.Indented);

File.WriteAllText("settlements.json", settlementsJson);

}

static List<T> DeserializeData<T>(string path)

{

List<T> list = new List<T>();

string textJson = File.ReadAllText(path);

if (!string.IsNullOrWhiteSpace(textJson))

{

list = JsonConvert.DeserializeObject<List<T>>(textJson);

}

return list;

}

static void DisplayData(List<Student> students, List<Commandant> commandants,

List<Hostel> hostels, List<Settlement> settlements)

{

Console.WriteLine("Students: ");

foreach (Student student in students)

{

Console.WriteLine(student);

}

Console.WriteLine("Commandants: ");

foreach (Commandant commandant in commandants)

{

Console.WriteLine(commandant);

}

Console.WriteLine("Hostels: ");

foreach (Hostel hostel in hostels)

{

Console.WriteLine(hostel);

}

Console.WriteLine("Settlements: ");

foreach (Settlement settlement in settlements)

{

Console.WriteLine(settlement);

}

}

}

}

# Скріншоти виконання

A screenshot of a computer screen

Description automatically generated

Рис. 3.1 – Використання десеріалізації

A screenshot of a computer screen

Description automatically generated

Рис. 3.2 – Використання JsonDocument

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Рис. 3.3 – 3.6 – Використання JsonNode