

Лабораторная работа № 15. Сериализация, десериализация объектов.

Тема: Сериализация, десериализация объектов

Цель работы: Формирование умений и навыков использования сериализованных объектов.

Общие теоретические сведения:

Сериализация - процесс перевода какой-либо структуры данных в последовательность битов. Обратной к операции сериализации является операция десериализации (структуризации) — восстановление начального состояния структуры данных из битовой последовательности.

Зачем нужна сериализация?

Для хранения и удобного получения данных из файла была придумана сериализация. Все объекты записываются в определенном формате. Таким образом, в одном приложении мы сериализуем данные, в другом десериализуем и получаем готовые объекты.

Рассмотрим несколько способов сериализовать объект.

Binary — представляет из себя преобразование объектов в поток байтов, который затем записывается в поток данных. После десериализации объекта создается точная копия исходного объекта.

SOAP — протокол обмена структурированными сообщениями в распределённой вычислительной среде. Первоначально SOAP предназначался в основном для реализации удалённого вызова процедур. Сейчас протокол используется для обмена произвольными сообщениями в формате XML, а не только для вызова процедур.

REST — архитектурный стиль взаимодействия компонентов распределённого приложения в сети. REST представляет собой согласованный набор ограничений, учитываемых при проектировании распределённой гипермедиа-системы. В определённых случаях это приводит к повышению производительности и упрощению архитектуры. Представлена в виде JSON.

Рассмотрим пример. Даны два класса. Student с полями name строкового типа, age — тип int, group — тип Group. Класс Group имеет поля name (название группы) — тип string, номер группы — тип int. Задание создать 10 групп, создать 300 студентов, и записать их в случайную группу, а далее объекты сериализовать массив студентов и массив групп, различными способами: binary, soap, xml, json.

[Serializable]

Ссылка: 9

public class Student

```
{  
    [NonSerialized]  
    int age;  
    Ссылка: 2  
    public string Name { get; set; }  
  
    Ссылка: 2  
    public int Age  
    {  
        get { return age; }  
        set  
        {  
            if (value <= 13)  
                age = value + 5;  
            else if (value <= 17)  
                age = value + 3;  
            else  
                age = value;  
        }  
    }  
}
```

Ссылка: 2

public Group Group { get; set; }

Ссылка: 1

```
public Student(string name, int age)  
{  
    Name = name;  
    Age = age;  
}
```

Ссылка: 0

public Student(){}

Ссылка: 7

public override string ToString()

```
{  
    return $"Студент {Name} возраст {Age} учится в группе {Group}";  
}
```

[DataContract]

Ссылка: 9

public class Group

```
{  
    [DataMember]  
    Ссылка: 2  
    public int Number { get; set; }  
  
    Ссылка: 2  
    public string Name { get; set; }  
  
    Ссылка: 0  
    public Group(){}  
  
    Ссылка: 1  
    public Group(int number)  
    {  
        Name = "BEST";  
        Number = number;  
    }  
  
    Ссылка: 7  
    public override string ToString()  
    {  
        return $"{{Name}}--{{Number}}";  
    }  
}
```

```

class Runner
{
    static Group[] groups = new Group[10];
    static Student[] students = new Student[300];
    static Random random = new Random();

    Ссылка: 0
    static void Main(string[] args)
    {
        GreateGroups();
        CreateStudents();
        PrintStudents();

        var binarySerialiazable = new BinaryFormatter();
        using (var file = new FileStream("students.bin", FileMode.OpenOrCreate))
        {
            binarySerialiazable.Serialize(file, students);
        }

        Console.WriteLine("-----");
        using (var file = new FileStream("students.bin", FileMode.OpenOrCreate))
        {
            var desirializableBinary = binarySerialiazable.Deserialize(file) as Student[];
            if (desirializableBinary != null)
            {
                foreach (var item in desirializableBinary)
                {
                    Console.WriteLine(item);
                }
            }
        }
    }
}

```

```

var soap = new SoapFormatter();
using (var file = new FileStream("GROUPS.soap", FileMode.OpenOrCreate))
{
    soap.Serialize(file, groups);
}

Console.WriteLine("-----");
using (var file = new FileStream("GROUPS.soap", FileMode.OpenOrCreate))
{
    var desirializableBinary = soap.Deserialize(file) as Group[];
    if (desirializableBinary != null)
    {
        foreach (var item in desirializableBinary)
        {
            Console.WriteLine(item);
        }
    }
}

```

```

XmlSerializer xmlformatter = new XmlSerializer(typeof(Student[]));
using (var file = new FileStream("students.xml", FileMode.OpenOrCreate))
{
    xmlformatter.Serialize(file, students);
    Console.WriteLine("XML created");
}

Console.WriteLine("-----");

using (var file = new FileStream("students.xml", FileMode.OpenOrCreate))
{
    var desXmlFormat = xmlformatter.Deserialize(file) as Student[];
    if (desXmlFormat != null)
    {
        Console.WriteLine(desXmlFormat[15]);
    }
}

```

```

DataContractJsonSerializer jsonformatter = new DataContractJsonSerializer(typeof(Group[]));
using (var file = new FileStream("grougs.json", FileMode.OpenOrCreate))
{
    jsonformatter.WriteObject(file, groups);
    Console.WriteLine("json created");
}

Console.WriteLine("-----");

using (var file = new FileStream("grougs.json", FileMode.OpenOrCreate))
{
    var desjson = jsonformatter.ReadObject(file) as Group[];
    if (desjson != null)
    {
        Console.WriteLine(desjson[2]);
    }
}

```

Практическая часть

Общее задание: Написать два любых класса (которые как-нибудь, между собой связаны, хотя бы как в примере), написать у каждого класса поля и свойства (если необходимы). Найти себе напарника, и договорится, кто будет сериализовать объекты, а кто будет десериализовать объект. (**напарник работает на отдельном компьютере в отдельном проекте**). Создать массивы объектов, ну или просто объекты. Реализовать сериализацию и десериализацию объекта согласно варианту (вариант у преподавателя).