

# 1. Лабораторная работа №5

## Сериализация

**Цель работы:** знакомство со способами сериализации объектов в XML и JSON.

**Задача работы:** научиться сохранять объекты в файл в форматах XML и JSON.

**Время выполнения работы:** 4 часа (2 занятия)

**Результат выполнения работы:** программа, обеспечивающая функционал согласно заданию.

### 1.1.Задание (общая часть)

- a) Создайте проект типа «Консольное приложение». Название проекта: GGGGGG\_NNN\_Lab9, где GGGGGG - номер группы, NNN – фамилия.
- b) Добавьте в решение новый проект – библиотеку классов. Название проекта: XXX.Domain, где XXX – название вашего решения
- c) В проекте XXX.Domain Опишите класс-контейнер и класс-контенеризуемый согласно предметной области индивидуального задания
- d) В проекте XXX.Domain Опишите интерфейс

```
interface ISerializer
{
    IEnumerable<CCC> DeSerializeByLINQ(string fileName);
    IEnumerable<CCC> DeSerializeXML(string fileName);
    IEnumerable<CCC> DeSerializeJSON(string fileName);
    void SerializeByLINQ(IEnumerable<CCC> xxx, string fileName);
    void SerializeXML(IEnumerable<CCC> xxx, string fileName);
    void SerializeJSON(IEnumerable<CCC> xxx, string fileName);
}
```

CCC – класс-контейнер, описанный в п.п. b)

- e) Добавьте в решение новый проект – библиотеку классов. Название проекта: SerializerLib
- f) В проекте SerializerLib, опишите класс Serializer, реализующий интерфейс ISerializer.  
– Методы SerializeXXX записывают коллекцию объектов в файл с именем fileName.

SerializeByLINQ – записывает объекты в формате XML. Объект XML создать с помощью LINQ-to-XML

`SerializeXML` - записывает объекты в формате XML. Сериализацию выполнить с помощью класса `XmlSerializer`

`SerializeJSON` - записывает объекты в формате JSON. Сериализацию выполнить с помощью `NewtonSoft.Json` (в среде .NET Framework) или `System.Text.Json` (в среде .Net Core).

– Методы `DeSerializeXXX` считывает данные из файла с именем `fileName` и десериализуют их в тип `IEnumerable<XXX>`. Способы десериализации такие же, что и в методах `SerializeXXX`.

g) В классе `Main`

– Создать коллекцию объектов класса-контейнера (отношение один-к-одному, например, в компьютере один винчестер). Заполнить коллекцию 5-6 объектами

– С помощью методов класса `Serializer` записать созданную коллекцию в три разных файла. Посмотреть содержимое файлов.

– С помощью методов класса `Serializer` прочитать файлы. В коде убедиться, что прочитанные данные совпадают с исходной коллекцией (**использовать интерфейс `IEquatable<T>` или `IEqualityComparer<T>`**).

## **1.2.Индивидуальные задания**

- a) Предметная область: Здание – Отопительная система;*
- b) Предметная область: Компьютер – Винчестер;*
- c) Предметная область: Больница – Приемное отделение.;*
- d) Предметная область: Завод – Склад деталей;*
- e) Предметная область: Аэропорт – Взлетная полоса;*
- f) Предметная область: Вокзал – Багажное отделение;*
- g) Предметная область: Фирма – Отдел кадров;*
- h) Предметная область: Ресторан – Кухня;*
- i) Предметная область: Персонаж фильма – Актер;*
- j) Предметная область: Библиотека – Книгохранилище;*
- k) Предметная область: Автомобиль – Двигатель;*
- l) Предметная область: Спутник – Планета;*
- m) Предметная область: Билет – Пассажир;*
- n) Предметная область: Водитель – Автобус;*
- o) Предметная область: Артист – Гонорар;*
- p) Предметная область: Пациент – Диагноз;*
- q) Предметная область: Собака – Порода;*
- r) Предметная область: Спортсмен – Вид спорта;*
- s) Предметная область: Строительный объект – Прораб;*
- t) Предметная область: Пирамида - Фараон*

**Инструменты и средства программирования**

## 2. Лабораторная работа №6

### Сборки и метаданные

**Цель работы:** знакомство с метаданными сборок .Net

**Задача работы:** научиться динамически загружать сборки и вызывать методы загруженных сборок.

**Время выполнения работы:** 2 часа

**Результат выполнения работы:** программа, обеспечивающая функционал согласно заданию.

### 2.1. Задание

- a) Создать новый проект
- b) Описать класс *Employee* (сотрудник), содержащий любые свойства типа *int* и *bool* и свойство *Name* (имя) типа *string*

- c) Описать обобщенный интерфейс

```
interface IFileService<T> where T:class
{
    IEnumerable<T> ReadFile(string fileName);
    void SaveData(IEnumerable<T> data, string fileName);
}
```

Метод ReadFile – считывает данные из файла с именем filename (данные в файле хранятся в формате json)

Метод SaveData сохраняет коллекцию data в файл с именем filename (данные сохраняются в формате json)

- d) Добавить в решение новый проект – **библиотеку классов**

#### Рекомендация.

Для удобства поиска библиотеки откройте окно свойств созданной библиотеки и в закладке «Build – Output Path» укажите путь к папке основного проекта.

- e) В **библиотеке классов** описать обобщенный класс *FileService<T>*, реализующий интерфейс IFileService<T> (см. выше). Для записи/чтения объектов используйте сериализацию/десериализацию Json.

**Примечание:** Не подключайте созданную библиотеку к основному проекту (как references или dependencies)

- f) В классе Program:

- Создать коллекцию объектов класса Employee. Заполнить коллекцию 5-6 объектами
- **Динамически** загрузить созданную библиотеку классов
- С помощью класса FileService (из библиотеки) записать в созданный файл коллекцию объектов класса Employee.
- С помощью класса FileService (из библиотеки) прочитать данные из записанного файла
- Вывести в консоль содержимое данных, прочитанных из файла

### 3. Лабораторная работа №7

#### Многопоточное программирование

**Цель работы:** знакомство с пространствами имен **System.Threading**.

**Задача работы:** научиться запускать методы в отдельных потоках, использовать механизмы синхронизации для управления работой потоков.

**Время выполнения работы:** 4 часа (2 занятия)

**Результат выполнения работы:** программа, обеспечивающая функционал согласно заданию.

**Классы создавать в отдельном проекте – библиотека классов .NET.** ||

#### 3.1. Задание .

1) Создать класс, содержащий метод вычисления интеграла функции  $y=\sin(x)$  на участке от 0 до 1 (использовать метод прямоугольников). Для итерации использовать шаг 0,00000001. Для увеличения времени выполнения вычисления на каждой итерации введите задержку в виде цикла из 100000 формальных вычислений (например, умножения двух чисел)

2) Предусмотреть в методе получение времени, затраченного на выполнение метода (использовать методы класса **StopWatch**). Результат выполнения передать с помощью события.

3) Предусмотреть возможность вывода информации о прогрессе выполнения метода (**использовать механизм событий, без применения интерфейса IProgress**).

4) В классе **Program**:

- Запустить вычисление интеграла в отдельном потоке
- Выводить в консоль прогресс выполнения метода вычисления в

виде:



- После завершения вычисления вывести:

Поток xxxxxxxxxxxx: Завершен с результатом: XXXX

– Запустить два экземпляра метода в разных потоках, предварительно установив приоритеты для одного потока Highest, а для второго Lowest. По завершении вычислений вывести результат и время выполнения для каждого потока.

5) Доработать код функции вычисления интеграла так, чтобы при запуске функции в нескольких потоках, выполнялся только один поток. Проверьте результат, запустив сразу 5 потоков.

6) Доработать код функции вычисления интеграла так, чтобы при запуске функции в нескольких потоках, выполнялся только заданное количество потоков (1, 2, 3 ...), а остальные переходили в режим ожидания. Проверьте результат, запустив сразу 5 потоков.

## 4. Лабораторная работа №7

### Многопоточное программирование

**Цель работы:** знакомство с пространством имен `system.Threading.Tasks`

**Задача работы:** научиться запускать методы в отдельных потоках, использовать механизмы синхронизации для управления работой потоков.

**Время выполнения работы:** 4 часа

**Результат выполнения работы:** программа, обеспечивающая функционал согласно заданию.

**Классы создавать в отдельном проекте – библиотека классов .NET.** ||

#### 4.1. Задание

Описать **один** класс предметной области согласно варианту. Класс должен содержать свойства:

- `Id` (уникальный номер),
- название объекта,
- свойство, необходимое для подсчета статистической информации

Создать класс `StreamService<T>`, в котором описать три метода:

1. `public async Task WriteToStreamAsync(Stream stream, IEnumerable<T> data, IProgress<string> progress)` – записывает коллекцию *data* в поток *stream*; **ОБЯЗАТЕЛЬНО:** установите задержку, чтобы запись выполнялась медленно, в течение 3-5 секунд.
2. метод `public async Task CopyFromStreamAsync(Stream stream, string filename, IProgress<string> progress)` - копирует информацию из потока *stream* в файл с именем *fileName*;
3. метод `public async Task<int> GetStatisticsAsync(string fileName, Func<T, bool> filter)` считывает объекты типа *T* из файла с именем *filename* и возвращает количество объектов, удовлетворяющих условию *filter*.

Использовать **асинхронные** методы чтения/записи в поток/файл

Предусмотреть **синхронизацию с помощью объектов синхронизации** методов 1 и 2 (копирование должно выполняться только после окончания записи)

В методах 1 и 2 предусмотреть оповещение о начале и конце записи/чтения в/из потока *stream* с указанием номера потока выполнения (`Thread.CurrentThread.ManagedThreadId`). Соответствующее сообщение должно быть выведено в консоль в классе Program (использовать `IProgress`)

В классе Program:



1. Сделать метод Main асинхронным (`static async Task Main(string[] args)`)
2. Создать коллекцию из 1000 объектов согласно индивидуальному заданию.
3. Вывести в консоль номер потока выполнения и сообщение о начале работы;
4. Синхронно (без использования *await*) запустить методы 1 и 2 класса StreamService (для гарантирования последовательности запуска потоков установите задержку 100-200 мс между запусками методов). В качестве данных передавайте коллекцию из п.2. В качестве параметра stream использовать MemoryStream. Методы 1 и 2 должны использовать **один и тот же** экземпляр *stream*.
5. Вывести в консоль номер потока выполнения и сообщение о том, что потоки 1 и 2 запущены;
6. **Ожидать** завершения выполнения методов 1 и 2;
7. **Асинхронно** получить статистические данные (метод GetStatisticsAsync);
8. Полученные статистические данные вывести в консоль
9. Модифицировать методы 1 и 2 для генерирования события о процессе выполнения задания (использовать IProgress). В событии передавать Id потока и выполняемое действие. В классе Program подписаться на событие и вывести в консоль информацию в виде:

Поток xxxxxxxxxxxx: уууу

Где: xxx – id потока, ууу – процент выполнения задачи

#### Рекомендации к заданию 2:

1000 объектов можно создать в цикле, присваивая формальные значения свойств класса в виде \$«[Имя свойства] {i}». Можно также использовать генератор случайных чисел.

Для генерирования случайных строковых данных можно использовать NuGet пакет LoremNET (<https://www.nuget.org/packages/LoremNET/2.0.0> , <https://github.com/trichards57/Lorem.Universal.NET/blob/master/readme.md> )

#### 4.2. Варианты заданий

1. **Предметная область – недвижимость.** Статистическая информация – количество домов, в которых количество жильцов больше 100;
2. **Предметная область – компьютеры.** Статистическая информация – количество компьютеров определенной марки

3. **Предметная область – пассажиры.** Статистическая информация – количество пассажиров, у которых есть багаж
4. **Предметная область – сотрудники предприятия.** Статистическая информация – количество сотрудников старше 35 лет
5. **Предметная область – автопарк.** Статистическая информация – количество автомобилей, у которых техосмотр должен проводиться в текущем году
6. **Предметная область – агентство по трудоустройству.** Статистическая информация – количество кандидатов на работу определенного профиля
7. **Предметная область – автосалон.** Статистическая информация – количество автомобилей с объемом двигателя более 2 литров
8. **Предметная область – художественная галерея.** Статистическая информация – количество работ определенного мастера
9. **Предметная область – пассажиры.** Статистическая информация – количество пассажиров, у которых есть багаж
10. **Предметная область – продукты питания.** Статистическая информация – количество продуктов, у которых закончился срок годности
11. **Предметная область – музыка.** Статистическая информация – количество песен определенного исполнителя
12. **Предметная область – студенты.** Статистическая информация – количество студентов, у которых средний балл больше 9
13. **Предметная область – биология.** Статистическая информация – количество существ, которые умеют летать
14. **Предметная область – обучающие курсы.** Статистическая информация – количество курсов, на которых количество слушателей больше 10
15. **Предметная область – игрушки.** Статистическая информация – количество игрушек для детей от 6 лет
16. **Предметная область – клиенты банка.** Статистическая информация – количество клиентов, открывших счет в текущем году
17. **Предметная область – багаж.** Статистическая информация – количество багажа, у которого вес превышает 20 кг.
18. **Предметная область – пациенты больницы.** Статистическая информация – количество пациентов с определенным диагнозом
19. **Предметная область – соревнования по стрельбе.** Статистическая информация – количество спортсменов, у которых сумма баллов больше 80

20. **Предметная область – персонажи игры.** Статистическая информация – количество персонажей, которые владеют определенным оружием.