1. Что такое сериализация, десериализация?

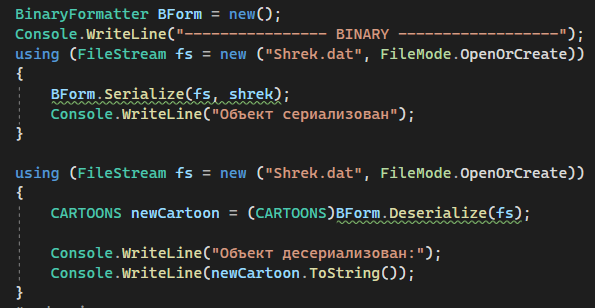
Сериализация — это процесс преобразования объекта в поток байтов для сохранения или передачи в память, базу данных или файл. Эта операция предназначена для того, чтобы сохранить состояния объекта для последующего воссоздания при необходимости.

Десериализация (deserialization) — это процесс создания структуры данных из битовой последовательности путем перевода этой последовательности в объекты и их упорядочивания (структуризации). Для десериализации данных используются десериализаторы. Десериализатор (deserializer) — это объект (или класс), который представляет собой набор функций и методов для десериализации данных.

2. Какие существуют форматы сериализации? Поясните структуру для каждого формата. Какие классы для работы с ними существуют в .NET?

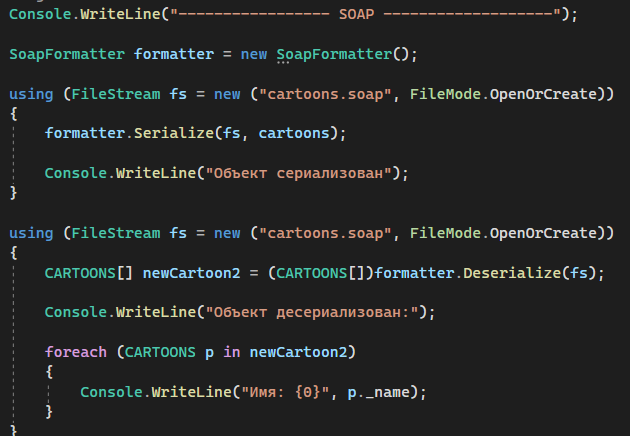
* Бинарный (в двоичное представление)



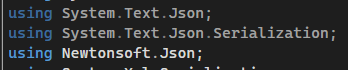


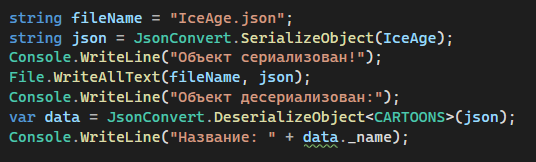
* SOAP - Протокол SOAP (Simple Object Access Protocol) представляет простой протокол для обмена данными между различными платформами. При такой сериализации данные упакуются в конверт SOAP, данные в котором имеют вид xml-подобного документа.

 \*необходимо установить soap.dll



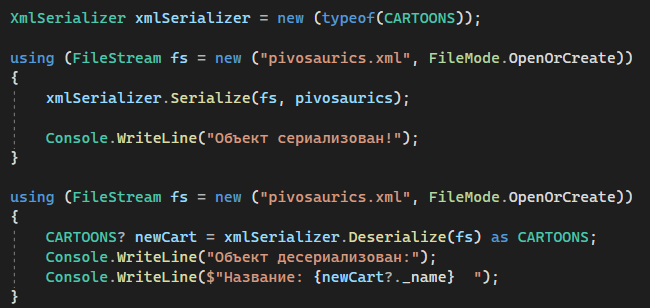
* JSON - **JSON** (JavaScript Object Notation) является одним из наиболее популярных форматов для хранения и передачи данных. JSON — это популярный формат текстовых данных, который используется для обмена данными в современных веб - и мобильных приложениях. Кроме того, JSON используется для хранения неструктурированных данных в файлах журналов или базах данных NoSQL



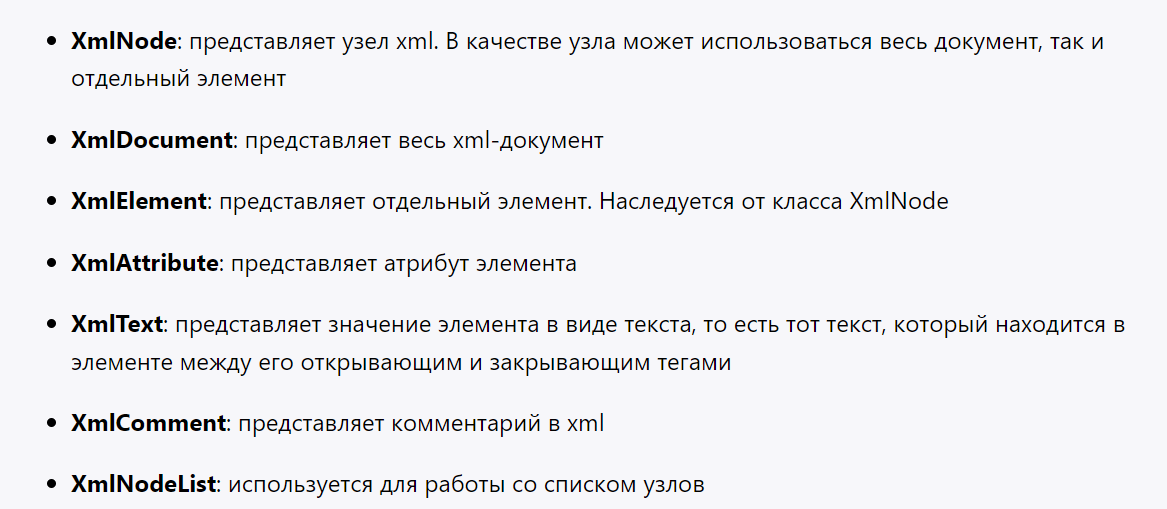


* XML – преобразование объекта в XML-файл

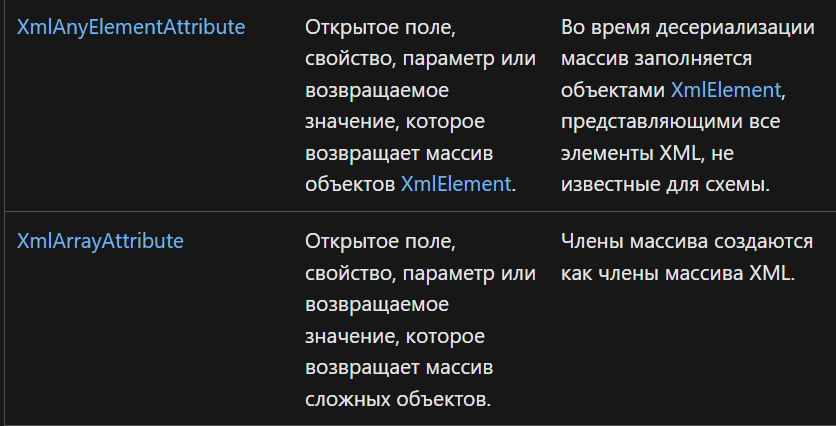


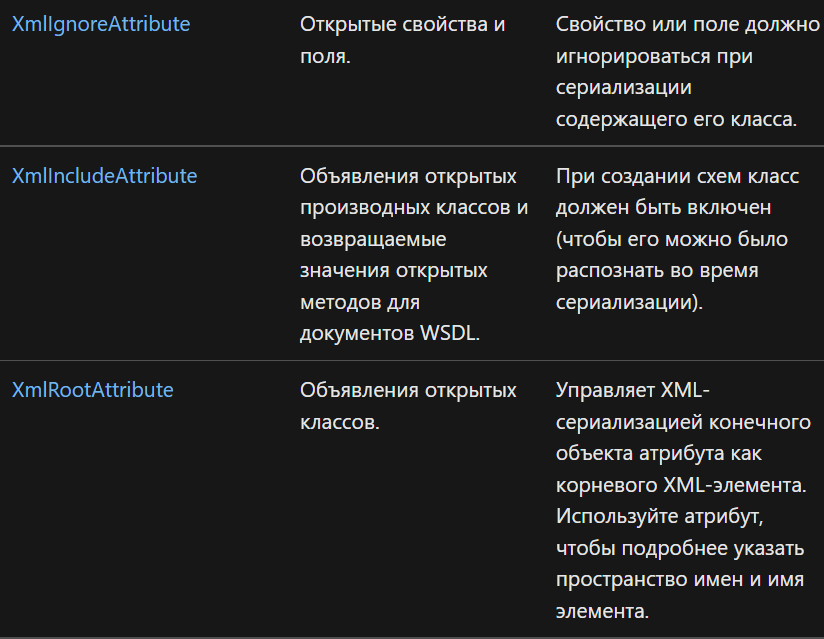


3. Какие классы существуют в пространстве имен System.Xml?

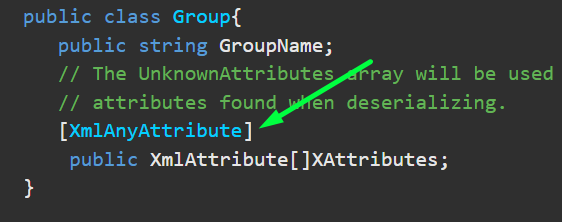


4. Какие атрибуты используются для настройки XML сериализации?





\*Пример использования



5. В чем отличие BinaryFormatter или SoapFormatter?

Сериализация в разные форматы.

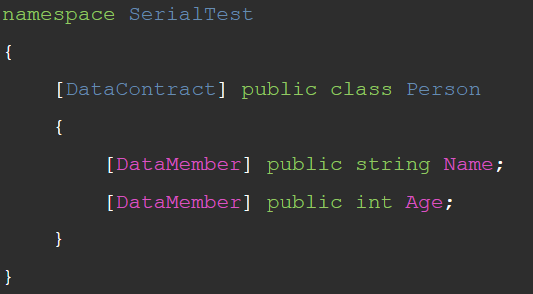
6. Что такое сериализация контрактов данных, контракт данных?

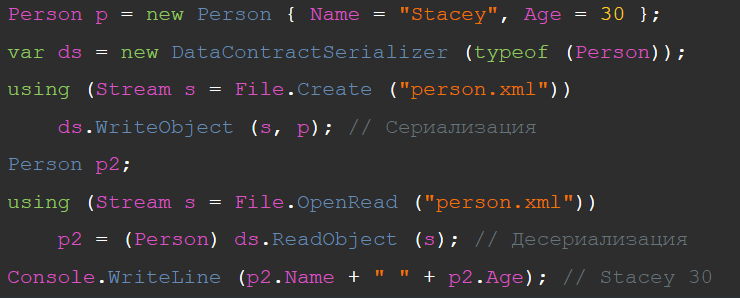
*Контракт данных*– это тип (класс или структура), объект которого описывает информационный фрагмент. Подразумевается, что этот фрагмент может быть сохранён, а затем восстановлен. Работу с контрактами данных можно рассматривать как один из механизмов сериализации.

**Сериализатор контрактов данных** является самым новым и универсальным. Он может быть использован при обмене информацией через протоколы обмена сообщениями. Он также обеспечивает хорошую переносимость версий и дает возможность десериализовать данные полученные из более ранней или более поздней версии типа, за счет того что устраняет связь между низкоуровневыми деталями объектов и структурой сериализованных данных. Сериализатор контрактов данных может быть использован в большинстве задач вместо двоичного и XML сериализатора.

Использование сериализатора контрактов данных предполагает следующие три шага

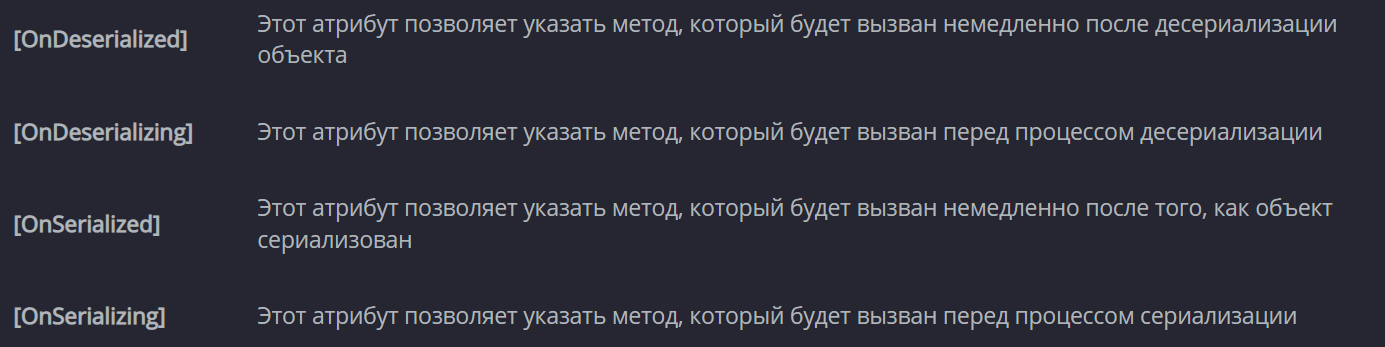
* выбрать класс для использования: DataContractSerializer или NetDataContractSerializer
* добавить сериализуемым типам и членам атрибуты [DataContract]и [DataMember] (соответственно)
* создать экземпляр сериализатора и вызвать его метод WriteObject или ReadObject





**\*** Контракт данных - формальное соглашение между службой и клиентом, абстрактно описывающее данные, обмен которыми происходит. Это значит, что для взаимодействия клиент и служба не обязаны совместно использовать одни и те же типы, достаточно совместно использовать одни и те же контракты данных.

7. Где и для чего используются атрибуты [OnSerializing], [OnSerialized], [OnDeserializing], [OnDeserialized]?



8. Что такое XPath? Приведите пример.

**XPath** представляет язык запросов в XML. Он позволяет выбирать элементы, соответствующие определенному селектору.

Рассмотрим некоторые наиболее распространенные селекторы:

.

выбор текущего узла

..

выбор родительского узла

\*

выбор всех дочерних узлов текущего узла

person

выбор всех узлов с определенным именем, в данном случае с именем "person"

@name

выбор атрибута текущего узла, после знака @ указывается название атрибута (в данном случае "name")

@+

выбор всех атрибутов текущего узла

element[3]

выбор определенного дочернего узла по индексу, в данном случае третьего узла

//person

выбор в документе всех узлов с именем "person"

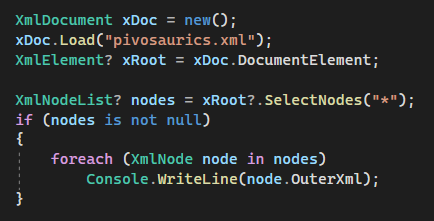
person[@name='Tom']

выбор элементов с определенным значением атрибута. В данном случае выбираются все элементы "person" с атрибутом name='Tom'

person[company='Microsoft']

выбор элементов с определенным значением вложенного элемента. В данном случае выбираются все элементы "person", у которых дочерний элемент "company" имеет значение 'Microsoft'

//person/company



9. Какие возможности дает LINQ to Xml. Приведите примеры

Позволяет создавать XML-документы.

