La serializzazione

Il termine serializzazione indica il processo di conversione di un **oggetto** in un flusso di byte, allo scopo di archiviare tale oggetto o trasmetterlo alla memoria, a un database o a un file. Il fine principale della serializzazione è salvare lo stato di un oggetto per consentirne la ricreazione in caso di necessità. Il processo inverso è denominato deserializzazione.

Quando abbiamo a che fare con una nostra classe in una nostra applicazione, potrebbe essere necessario salvare lo stato dell'oggetto istanziato, per esempio, in un file. Questa attività viene tecnicamente definita

PERSISTENZA DI UN OGGETTO

cioè la capacità di memorizzazione dei dati in esso contenuti per poi poterli ripristinare in seguito.

Facciamo un piccolo esempio.

Creo questa semplice classe:

```
class Studente
    {
        public string Nome = "";
        public string Cognome = "";
        public string Classe = "";
    }
```

Per renderla persistente, ho due strade:

1. salvare i suoi campi in un file con le tecniche viste in precedenza
2. utilizzare la **SERIALIZZAZIONE** che è più immediata

L'utilizzo della serializzazione, prevede la necessità di '**MARCARE**' la classe come '*serializzabile*' cioè contrassegnarla come tale.
Questo lo si fa semplicemente aggiungendo un particolare *attributo* laddove voglio salvarne lo stato.

Modifico la classe precedente aggiungendo l'attributo '*Serializable*', così:

```
[Serializable]
class Studente
    {
        public string Nome = "";
        public string Cognome = "";
        public string Classe = "";
    }
```

l'attributo consente la memorizzazione, in questo esempio, di tutti i suoi membri.

Codice di esempio:

```
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;

namespace Serializzazione1
{
    class Program
    {
        static void Main(string[] args)
        {
            Studente Alunno = new Studente();
            Alunno.Nome = "Michele";
            Alunno.Cognome = "De Benedittis";
            Alunno.Classe = "4C";
            IFormatter formatter = new BinaryFormatter();
            Stream stream = new FileStream("Alunno.bin",
                                           FileMode.Create,
                                           FileAccess.Write,
                                           FileShare.None);
            formatter.Serialize(stream, Alunno);
            stream.Close();
        }
    }
}
```

Spiegazione:

`Studente Alunno = new Studente();`	Creo l'oggetto di tipo Alunno

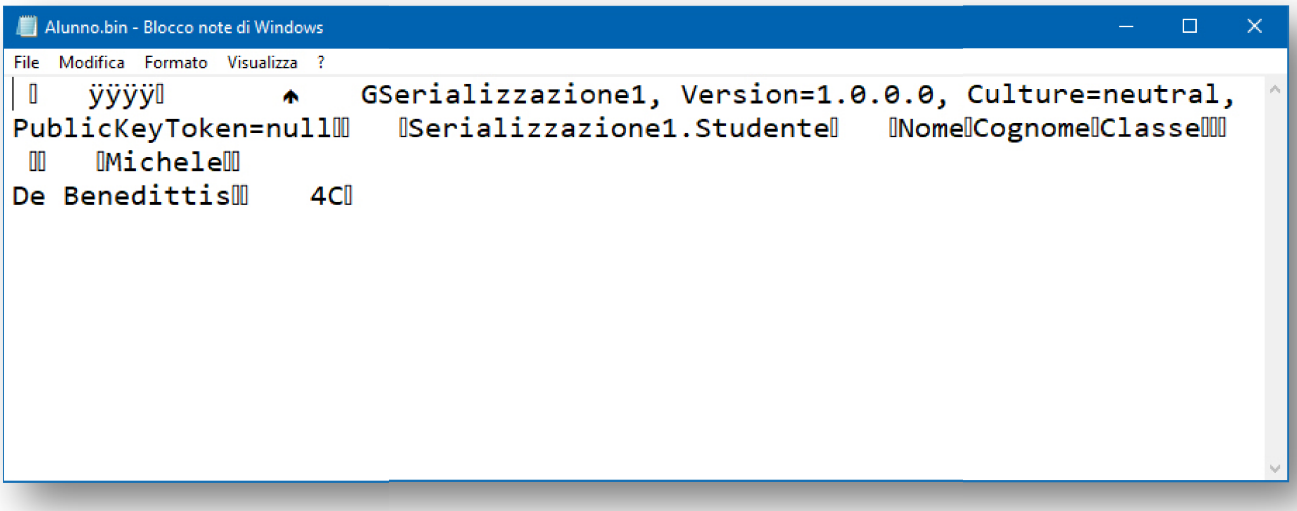
`Alunno.Nome = "Michele";`
`Alunno.Cognome = "De Benedittis";`
`Alunno.Classe = "4C";`
	Imposto i membri

`IFormatter formatter = new BinaryFormatter();`	`IFormatter` è una cosiddetta 'interfaccia' che consente la corretta formattazione degli oggetti da serializzare attraverso l'oggetto formatter creato con l'istanza della classe **BinaryFormatter**()

`Stream stream = new FileStream("Alunno.bin",`
`FileMode.Create,`
`FileAccess.Write,`
`FileShare.None);`
	Creo un oggetto di tipo **Stream** attraverso l'istanza della classe **FileStream**. Il costruttore prevede quattro parametri che sono autodescrittivi

`formatter.Serialize(stream, Alunno);`	Serializzo l'oggetto 'Alunno' richiamando il metodo `Serialize` dall'oggetto 'formatter' nel file `"Alunno.bin"`
`stream.Close();`	Chiudo il flusso

Una volta lanciato il programma, portiamoci nella directory del progetto e apriamo il file "Alunno.bin" con il blocco note e vediamo cosa contiene:



Il contenuto del file è ciò che il BinaryFormatter ha memorizzato dell'oggetto. Si notano le seguenti informazioni:

1. Serializzazione1 - il nome del programma
2. Version - la sua versione
3. Culture=neutral - rappresenta l'impostazioni cultura non associate ad alcun paese
4. PublicKeyToken - Un token a chiave pubblica è qualcosa che rende unico l' assembly. Ogni assembly .NET compilato è firmato da una chiave privata. Questa chiave privata è univoca e utilizzata per identificare l'assembly. Ma questa chiave privata non è pubblicamente visibile. Prima di essere esposto pubblicamente, viene eseguito l'hashing in una stringa lunga di 16 caratteri (64 bit) che viene chiamata token chiave pubblica. Però in questo esempio è null.
5. Serializzazione1.Studente - è la classe che è stata serializzata **non** una sua istanza
6. Nome Cognome Classe - l'elenco dei campi serializzati e, a seguire, i valori impostati

Inoltre compaiono strani caratteri che vengono utilizzati come controllo e informazione della serializzazione.