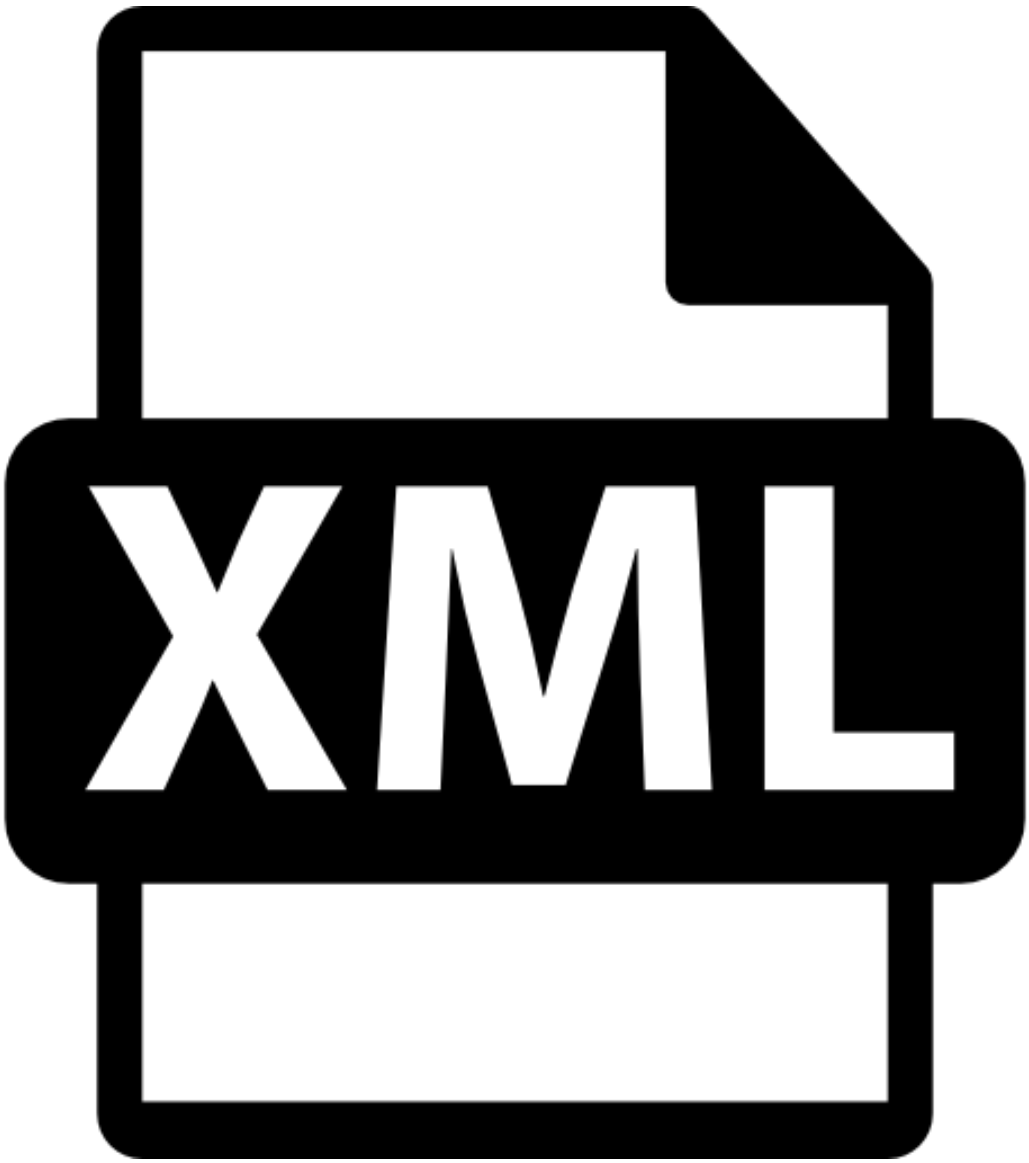


Sérialisation et Désérialisation

Julien CONDOMINES | StJoSup | 18 mars 2022

Table des matières

I-Introduction

Il est aujourd'hui question d'étudier la notion de sérialisation et de désérialisation. La sérialisation est le processus de conversion des informations d'état d'un objet sous une forme qui peut être stockée ou transmise. En termes simples, c'est le processus de conversion d'un objet en une séquence d'octets. La désérialisation est le processus inverse de sérialisation, le processus de désérialisation d'un tableau d'octets dans un objet et de le restaurer dans un objet. Alors, quels problèmes la sérialisation et la désérialisation résolvent-elles?

L'objet est sérialisé dans un tableau d'octets, l'objet peut être utilisé pour la transmission entre les processus à travers le réseau et l'objet peut être stocké en permanence dans un dispositif de stockage. C'est le principal problème que la sérialisation résout: les problèmes de stockage et les problèmes de transmission.

Nous allons voir ici les trois possibilités offertes par DotNet.

II-La sérialisation SOAP

La sérialisation SOAP crée un fichier XML en utilisant "l'enveloppe" SOAP. SOAP (Simple Object Access Protocol) est un protocole au-dessus de HTTP qui propose un format de messagerie pour la communication de machine à machine. Ce protocole est utilisé dans les services Web de communication et d'activation d'objets distants. L'avantage de ce protocole est qu'il est standard, ce qui rompt avec les protocoles ou technologies "propriétaires", Corba ou COM.

Pour le matérialiser en cSharp, il est nécessaire d'ajouter la référence « using System.Runtime.Serialization.Formatters.Soap » ainsi que de la déclarer la classe comme sérialisable avec la notion « [System.Serializable] »

```
using System.Runtime.Serialization.Formatters.Soap;
[System.Serializable]
```

Pour la désérialisation, c'est l'opération inverse, on va lire le fichier et « reconstruire » le fichier que l'on a sérialisé précédemment.

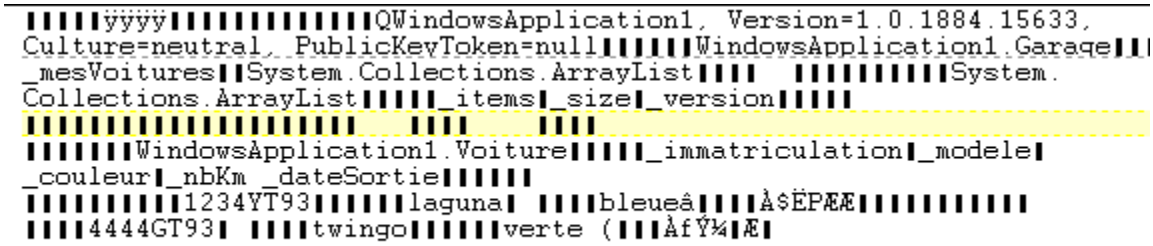
III-Sérialisation binaire

La meilleure raison d'utiliser la sérialisation binaire est de rendre persistant l'état d'un objet afin de pouvoir le recréer à l'identique ultérieurement. Cet objet peut alors être stocké dans un fichier ou envoyé en tant que flux à travers le réseau. Un autre avantage est que la sérialisation binaire conserve l'état des membres publics et privés de l'objet.

A contrario, il n'est pas aisé d'utiliser les flux générés dans ou depuis une application tierce.

Il faut ici remplacer la référence précédente SOAP par la référence Binary « using System.Runtime.Serialization.Formatters.Binary »

Un exemple de fichiers sérialiser en binaire :



IV-La sérialisation XML

La sérialisation XML sérialise uniquement les champs publics et les valeurs de propriété d'un objet dans un flux de données XML. La sérialisation XML n'inclut pas d'informations de type. Par exemple, si un objet **Book** se trouve dans l'espace de noms **Library**, il n'y a aucune garantie qu'il soit désérialisé dans un objet du même type.

Il faut ici remplacer la référence par « using System.Runtime.Serialization; » et « using System.Xml.Serialization; »

```
using System.Xml.Serialization;
```

VI-Conclusion

Nous avons donc vu ici la méthode qui permet de compresser un objet en un formulaire afin de le rendre plus facilement transportable pour par exemple le déplacer par internet via http entre un client et un serveur.

VII-Annexes

SERIALISATION SOAP

Classe Form

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Runtime.Serialization.Formatters.Soap;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Runtime.Serialization.Formatters.Binary;
using System.Xml.Serialization;

namespace Sérialisation_SOAP
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
```

```
        }

        private void btn_SerialiseSOAP_Click(object sender, EventArgs e)
        {
            txt.Text = "";
            Voiture v = new Voiture("1234YT93", "laguna", "bleue", 1250,
"12/1/2005");
            Voiture v1 = new Voiture("4444GT93", "twingo", "verte", 10400,
"1/11/2004");
            Garage g = new Garage();
            g.AddVoiture(v);
            g.AddVoiture(v1);
            FileStream fichier = new FileStream("garage.sr",
FileMode.Create);
            SoapFormatter sf = new SoapFormatter();
            sf.Serialize(fichier, g);
            fichier.Close();
        }

        private void btn_DeserialiseSoap_Click(object sender, EventArgs e)
        {
            FileStream fichier = new FileStream("garage.sr",
FileMode.Open);
            SoapFormatter sf = new SoapFormatter();
            Garage g = (Garage)sf.Deserialize(fichier);
            txt.Text = g.GetVoiture(1).immatriculation;
            fichier.Close();

        }

        private void button1_Click(object sender, EventArgs e)
        {
            FileStream fichier = new FileStream("garageBinaire.txt",
FileMode.Open);
            BinaryFormatter bf = new BinaryFormatter();
            Garage g = (Garage)bf.Deserialize(fichier);
            txt.Text = g.GetVoiture(1).immatriculation;
            fichier.Close();
```

```
        }

        private void button2_Click(object sender, EventArgs e)
        {
            txt.Text = "";
            Voiture v = new Voiture("1234YT93", "laguna", "bleue", 1250,
"12/1/2005");
            Voiture v1 = new Voiture("4444GT93", "twingo", "verte", 10400,
"1/11/2004");
            Garage g = new Garage();
            g.AddVoiture(v);
            g.AddVoiture(v1);
            FileStream fichier = new FileStream("garageBinaire.txt",
FileMode.Create);
            BinaryFormatter bf = new BinaryFormatter();
            bf.Serialize(fichier, g);
            fichier.Close();

        }

        private void btn_SerXML_Click(object sender, EventArgs e)
        {
            txt.Text = "";
            Voiture v = new Voiture("121GFT93", "Twingo", "verte", 1235,
"12/12/2004");
            XmlSerializer serializer = new XmlSerializer(typeof(Voiture));
            FileStream fichier = new FileStream("voitureXML.xml",
FileMode.Open);
            serializer.Serialize(fichier, v);
            fichier.Close();

        }

        private void btn_DesXML_Click(object sender, EventArgs e)
        {
            FileStream fichier = new FileStream("voitureXML.xml",
FileMode.Open);
            XmlSerializer serializer = new XmlSerializer(typeof(Voiture));
            Voiture v = (Voiture)serializer.Deserialize(fichier);
            txt.Text = v.immatriculation;
```

```
            fichier.Close();

        }
    }
}
```

Classe Garage

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Xml.Serialization;
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Soap;


namespace Sérialisation_SOAP
{
    [System.Serializable]
    class Garage
    {
        public Garage()
        {
            _mesVoitures = new ArrayList();
        }
        public void AddVoiture(Voiture v)
        {
            _mesVoitures.Add(v);
        }
        public Voiture GetVoiture(int n)
        {
            return (Voiture)_mesVoitures[n];
        }
        private ArrayList _mesVoitures;
```

Classe Voiture

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Sérialisation_SOAP
{
    [System.Serializable]
    public class Voiture
    {
        public Voiture() { }
        public Voiture(string imma, string modele, string couleur, int
nbKms, string dateSortie)
        {
            this._immatriculation = imma;
            this._modele = modele;
            this._couleur = couleur;
            this._nbKm = nbKms;
            this._dateSortie = Convert.ToDateTime(dateSortie);
        }

        public string immatriculation
        {
            get
            {
                return this._immatriculation;
            }
            set
            {
                this._immatriculation = value;
            }
        }
        public string modele
        {
            get
            {
                return this._modele;
            }
```

```
            set
            {
                this._modele = value;
            }
        }
        public string couleur
        {
            get
            {
                return this._couleur;
            }
            set
            {
                this._couleur = value;
            }
        }
        public int nbKm
        {
            get
            {
                return this._nbKm;
            }
            set
            {
                this._nbKm = value;
            }
        }
        public string dateSortie
        {
            get
            {
                return Convert.ToString(this._dateSortie);
            }
            set
            {
                this._dateSortie = Convert.ToDateTime(value);
            }
        }
        public void Roule(int kms)
        {
```

```
            this._nbKm += kms;
        }
        private string _immatriculation;
        private string _modele;
        private string _couleur;
        private int _nbKm;
        private DateTime _dateSortie;
    }


}
```