Serialización

19.2

Programación II y Laboratorio de Computación II

Edición 2018

Serialización

- ¿Qué es?
  - Es el proceso de convertir un objeto en memoria en una secuencia lineal de bytes.
- ¿Para qué sirve?
  - Para pasarlo a otro proceso.
  - Para pasarlo a otra máquina.
  - Para grabarlo en disco.
  - Para grabarlo en una base de datos.

Formatters

- Controlan el formato de la serialización.
- Serialización a XML
  - Por defecto incluye sólo las propiedades y atributos públicos.
- Serialización Binaria
  - Por defecto incluyen todos los atributos y propiedades, ya sean públicas o privadas.
- ¿Y después?
  - Se reconstruye el objeto mediante Deserialización - proceso inverso.
  - Puede ser en el mismo proceso o no, en la misma máquina o no.

Serialización XML

Serialización XML

- La serialización XML sólo serializa los atributos públicos y los valores de propiedad de un objeto en una secuencia XML.
- La serialización XML no convierte los métodos, indexadores, atributos privados ni propiedades de sólo lectura (salvo colecciones de sólo lectura).
- La clase central de la serialización XML es **XmlSerializer** y sus métodos más importantes son **Serialize** y **Deserialize**.

Serialización XML

- La secuencia XML que genera XmlSerializer cumple con la recomendación 1.0 del W3C (www.w3.org) acerca del lenguaje de definición de esquemas XML (XSD).

- Además, los tipos de datos generados cumplen las especificaciones enumeradas en el documento titulado "XML Schema Part 2: Datatypes".

- Al crear una aplicación que utiliza la clase XmlSerializer, debe tener en cuenta los siguientes elementos y sus implicaciones:

Serialización XML

- La clase XmlSerializer crea archivos C# (.cs) y los compila en archivos .dll en el directorio especificado por la variable de entorno TEMP; la serialización se produce con esos archivos DLL.
- Una clase debe tener un constructor por defecto para que XmlSerializer pueda serializarla.
- Sólo se pueden serializar los atributos y propiedades públicas.
- Los métodos no se pueden serializar.

XMLSerializer

- XmlSerializer (System.Type type)
- Inicializa una nueva instancia de la clase XmlSerializer la cual puede serializar objetos del tipo especificado en el parámetro type.

- Serialize (System.IO.Stream stream, Object o)
- Serializa el objeto especificado y escribe en un documento Xml usando el Stream especificado.

- Deserialize (System.IO.Stream stream)
- Deserializa el documento Xml contenido por el Stream especificado.

XmlTextWriter

- Provee una manera de generar archivos con contenido de datos XML que cumple con la recomendación 1.0 del W3C (www.w3.org) acerca del lenguaje de definición de esquemas XML (XSD).

- Métodos:
  - XmlTextWriter (string filename, System.Text.Encoding encoding)
  - Crea una instancia de XmlTextWriter.
  - El filename indica en que archivo se escribirá.
  - Con encoding se indicará cual será la codificación.

Ejemplo

```
using System.Xml;
using System.Xml.Serialization;
//...
Dato p = new Dato();   //Objeto a serializar.
XmlTextWriter writer;  //Objeto que escribirá en XML.
XmlSerializer ser;     //Objeto que serializará.

//Se indica ubicación del archivo XML y su codificación.
writer = new XmlTextWriter(ArchivoXml, Encoding.UTF8);
//Se indica el tipo de objeto ha serializar.
ser = new XmlSerializer(typeof(Dato));
//Serializa el objeto p en el archivo contenido en writer.
ser.Serialize(writer, p);
//Se cierra la conexión al archivo
writer.Close();
```

XmlTextReader

- Provee una manera de leer archivos con contenido de datos XML.

- Métodos:
  - XmlTextReader (string url)
  - Crea una instancia de XmlTextReader.
  - El url indica en que archivo están los datos XML.

Ejemplo

```
using System.Xml;
using System.Xml.Serialization;
//...
//Objeto que alojará los datos contenidos en el archivo XML.
Dato aux = new Dato();
XmlTextReader reader;   //Objeto que leerá XML.
XmlSerializer ser;      //Objeto que Deserializará.

//Se indica ubicación del archivo XML.
reader = new XmlTextReader(ArchivoXml);
//Se indica el tipo de objeto ha serializar.
ser = new XmlSerializer(typeof(Dato));
//Deserializa el archivo contenido en reader, lo guarda en aux.
aux = (Dato)ser.Deserialize(reader);
//Se cierra el objeto reader.
reader.Close();
```

Resumen

- Se debe colocar un constructor por defecto en las clases a serializar.
- Sólo se guardaran los atributos o propiedades públicas.
- Si hay relación de herencia, se deberá colocar [XmlInclude(typeof(Clase))] en la clase base e indicando cada clase heredada.
- Espacio de nombres: System.Xml.Serialization

Serialización Binaria

Serialización Binaria

- Para poder hacer una serialización binaria se debe agregar el marcador [Serializable]

```
[Serializable]
class MiClase
{
}
```

BinaryFormatter

- Serializa y Deserializa objetos en formato binario.
- Se encuentra en el espacio de nombres System.Runtime.Serialization.Formatters.Binary
- Puede serializar atributos públicos y privados.
- Una clase debe tener un constructor por defecto para que BinaryFormatter pueda serializarla.
- Los métodos más importantes de la clase BinaryFormatter son:
  - Serialize
  - Deserialize

Métodos BinaryFormatter

- BinaryFormatter()
  - Inicializa una nueva instancia de la clase BinaryFormatter.
- Serialize(System.IO.FileStream seralizationStream, Object graph)
  - Serializa el objeto especificado y escribe en un archivo binario usando el serializationStream especificado.
- Deserialize(System.IO.FileStream serializationStream)
  - Deserializa el archivo binario contenido por el serializationStream especificado.

FileStream

- Genera un objeto para leer, escribir, abrir y cerrar archivos.
- Métodos:
  - FileStream (string path, System.IO.FileMode mode)
  - Inicializa una instancia de FileStream, indicando ubicación y el modo en que se creará o abrirá el archivo.
  - Read (byte[] array, int offset, int count)
  - Lee un bloque de bytes y escribe los datos en el buffer dado.
  - Seek (long offset, System.IO.SeekOrigin origin)
  - Establece la posición del stream al valor dado.
  - Write (byte[] array, int offset, int count)
  - Escribe un bloque de bytes en el stream.

Ejemplo Serialización

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

//...
Dato aux = new Dato(); // Objeto a serializar.
Stream fs;             // Objeto que escribirá en binario.
BinaryFormatter ser;  // Objeto que serializará.

//Se indica ubicación del archivo binario y el modo.
fs = new FileStream(ArchivoBinario, FileMode.Create);
//Se crea el objeto serializador.
ser = new BinaryFormatter();
//Serializa el objeto p en el archivo contenido en fs.
ser.Serialize(fs, p);
//Se cierra el objeto fs
fs.Close();
```

Ejemplo Deserialización

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

//...
//Objeto que alojará los datos contenidos en el archivo binario.
Dato aux = new Dato();
Stream fs;              // Objeto que leerá en binario.
BinaryFormatter ser;  // Objeto que Deserializará.

//Se indica ubicación del archivo binario y el modo.
fs = new FileStream(pathBinario, FileMode.Open);
//Se crea el objeto deserializador.
ser = new BinaryFormatter();
//Deserializa el archivo contenido en fs, lo guarda en aux.
aux = (Dato)ser.Deserialize(fs);
//Se cierra el objeto fs.
fs.Close();
```