

Serialización XML

Serialización XML. Introducción

- XML (eXtensible Markup Language) es un estándar diseñado para el almacenamiento y transporte de información.
- El proceso de serialización de un objeto consiste en almacenar la totalidad de la información de un objeto en memoria secundaria para su posterior recuperación.
- Aunque en .NET se puede serializar en binario, la serialización en XML es especialmente interesante dada la fácil lectura de los ficheros generados.

Serialización XML en .NET

- En .NET el mecanismo de serialización XML más cómodo es mediante el uso de reflexión, mediante el cual prácticamente obtenemos esta serialización de forma automática.
- **using System.Xml;**
- **using System.Xml.Serialization;**
- Básicamente debemos procurar propiedades públicas de todo elemento a serializar de la clase (siempre mejor que atributos/campos públicos). El constructor por defecto también debe estar disponible.
- *La clase tiene que ser pública.*
- Mediante atributos específicos, mediante [] antes de cada campo o clase, podemos controlar de forma más exacta esta serialización

Ejemplo

La clase

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

namespace Serializacion
{
    public class Persona
    {
        public String Nombre { get; set; }
        public String Apellido1 { get; set; }
        public String Apellido2 { get; set; }
        public int Edad { get; set; }

        public Persona(string nombre, string ape1, string ape2, int edad)
        {
            Nombre = nombre;
            Apellido1 = ape1;
            Apellido2 = ape2;
            Edad = edad;
        }

        public Persona() { } //Es preciso para la serialización
    }
}
```

Ejemplo

El código

```
Persona persona = new Persona("Pepe", "Devesa", "LLinares", 25);  
  
StreamWriter myWriter = new StreamWriter("persona.xml");  
  
XmlSerializer mySerializer = new XmlSerializer(typeof(Persona));  
mySerializer.Serialize(myWriter, persona);  
myWriter.Close();
```

Creamos el fichero

Definimos el tipo

Serializamos el objeto

Otro código

```
List<Persona> personas = new List<Persona>();  
personas.Add(new Persona("Pepe", "Devesa", "LLinares", 25));  
personas.Add(new Persona("Pepe", "Devesa", "LLinares", 25));  
personas.Add(new Persona("Pepe", "Devesa", "LLinares", 25));  
  
StreamWriter myWriter = new StreamWriter("persona.xml");  
  
XmlSerializer mySerializer = new XmlSerializer(typeof(List<Persona>));  
mySerializer.Serialize(myWriter, personas);  
myWriter.Close();
```

Deserialización...

Declaramos la variable de instancia

```
Persona p = null;  
StreamReader myReader = new StreamReader("persona.xml");  
XmlSerializer mySerializer = new XmlSerializer(typeof(Persona));  
p = (Persona)mySerializer.Deserialize(myReader);  
  
listBox1.Items.Add(p.Nombre + " " + p.Apellido1 + " " + p.Apellido2 + " " + p.Edad);
```

Cargamos los valores en la clase..

...y los mostramos

Algunos atributos útiles

- Mediante atributos específicos, mediante [] antes de cada campo o clase, podemos controlar de forma más exacta esta serialización.
- Algunos atributos útiles:
 - XmlRoot(), para dar un nombre distinto al objeto serializado.
 - XmlAttribute(), para definir un campo como atributo y no elemento XML
 - XmlIgnore(), para que se ignore un elemento en la serialización.
 - etc.

Ejemplo en la clase

General

No hay controles utilizables en este grupo. Arrastre un elemento a este texto y agréguelo al cuadro de herramientas.

Form1.cs Form1.cs [Diseño] Persona.cs Form1.cs Form1.cs [Diseño]

Serializacion.Persona

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Xml.Serialization;

namespace Serializacion
{
    public class Persona
    {
        [XmlAttribute()]
        public String DNI { get; set; }

        public String Nombre { get; set; }
        public String Apellido1 { get; set; }
        public String Apellido2 { get; set; }
        public int Edad { get; set; }

        public Persona(string dni, string nombre, string ape1, string ape2, int edad)
        {
            DNI = dni;
            Nombre = nombre;
            Apellido1 = ape1;
            Apellido2 = ape2;
            Edad = edad;
        }

        public Persona() { } //Es preciso para la serialización
    }
}
```

En la **clase** añadimos el espacio de nombres y el atributo específico..

Explorador de soluciones

Solución 'Taller' (3 proyectos)

- ejemplo
- Fitxers
- Serializacion
 - Properties
 - References
 - Form1.cs
 - Persona.cs
 - Program.cs

Explorador de sol... Team Explorer Vista de clases

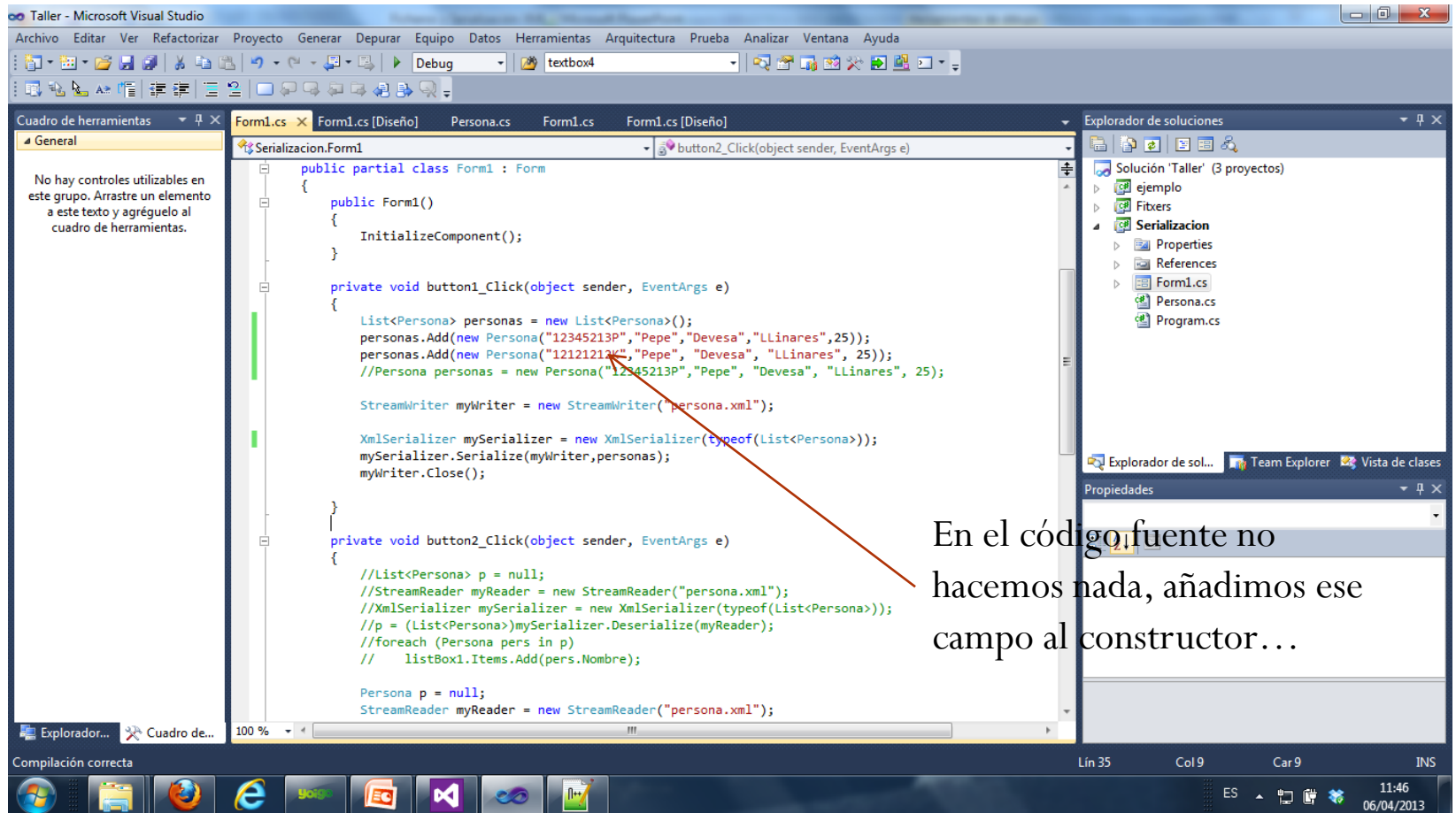
Propiedades

Compilación correcta

Lín 21 Col 23 Car 23 INS

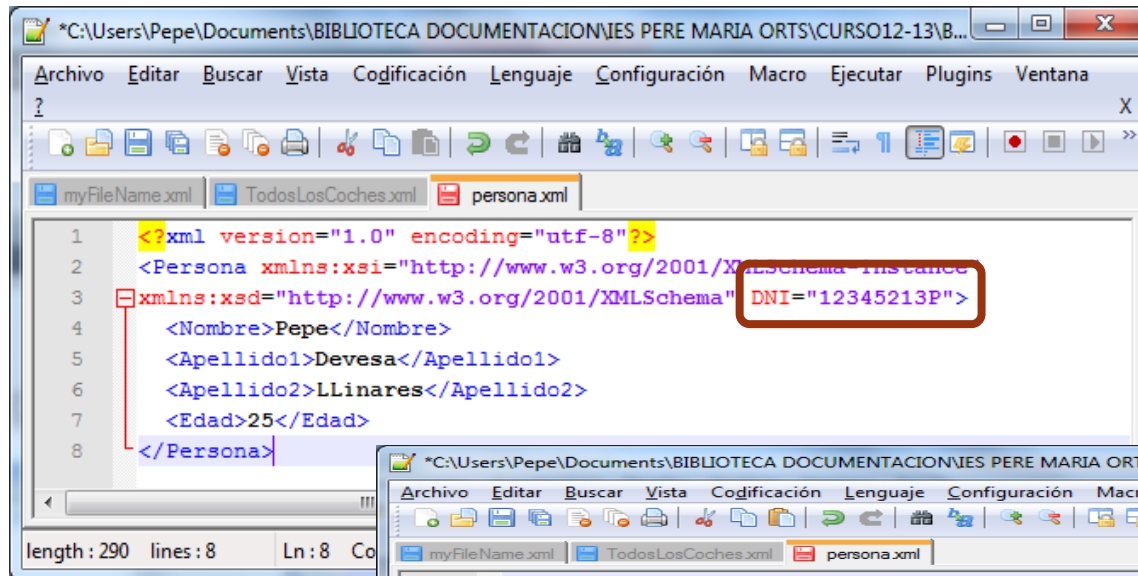
ES 11:42 06/04/2013

Ejemplo en el programa



...y el resultado es:

Un registro

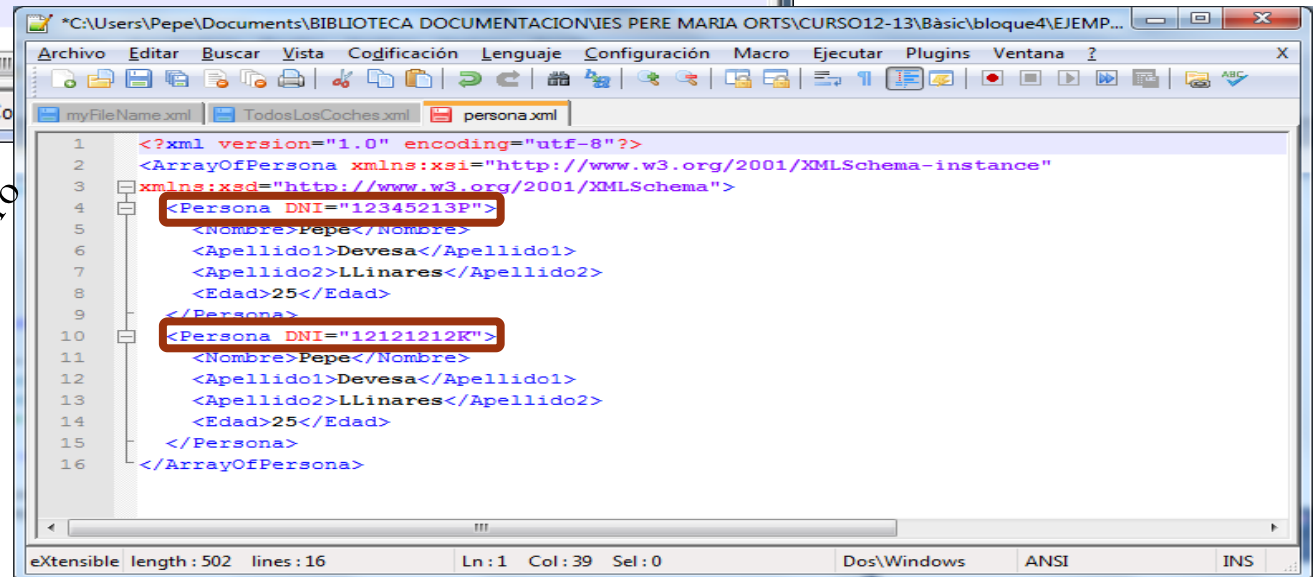


A screenshot of an XML editor window titled '*C:\Users\Pepe\Documents\BIBLIOTECA DOCUMENTACION\IES PERE MARIA ORTS\CURSO12-13\B...'. The window shows a single XML record in 'persona.xml'. The XML code is as follows:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <Persona xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3 xmlns:xsd="http://www.w3.org/2001/XMLSchema" DNI="12345213P">
4   <Nombre>Pepe</Nombre>
5   <Apellido1>Devesa</Apellido1>
6   <Apellido2>LLinares</Apellido2>
7   <Edad>25</Edad>
8 </Persona>
```

The status bar at the bottom indicates 'length: 290 lines: 8 Ln: 8 Co: 1'.

Una lista de registro



A screenshot of an XML editor window titled '*C:\Users\Pepe\Documents\BIBLIOTECA DOCUMENTACION\IES PERE MARIA ORTS\CURSO12-13\Básic\bloque4\EJEMP...'. The window shows a list of two XML records in 'ArrayOfPersona.xml'. The XML code is as follows:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <ArrayOfPersona xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
4   <Persona DNI="12345213P">
5     <Nombre>Pepe</Nombre>
6     <Apellido1>Devesa</Apellido1>
7     <Apellido2>LLinares</Apellido2>
8     <Edad>25</Edad>
9   </Persona>
10  <Persona DNI="12121212K">
11    <Nombre>Pepe</Nombre>
12    <Apellido1>Devesa</Apellido1>
13    <Apellido2>LLinares</Apellido2>
14    <Edad>25</Edad>
15  </Persona>
16 </ArrayOfPersona>
```

The status bar at the bottom indicates 'eXtensible length: 502 lines: 16 Ln: 1 Col: 39 Sel: 0 Dos\Windows ANSI INS'.