

## ARCHIVOS XML

En ciertas situaciones tenemos que convertir objetos en algún formato portátil para, por ejemplo, ser mandado por Internet a un web service. Para esto se suele utilizar el formato **XML (Extendable Markup Language)**.

En .NET, contamos con varias opciones para manipular XML, tanto en operaciones de lectura como de escritura. Debemos considerarlas dependiendo de la situación:

- **XMLSerializer:** Cuando tenemos que escribir datos en XML y volver a leerlos, utilizar serialización nos da solidez y organización a la hora de convertir un objeto en XML. Fácil y rápido de implementar, pero puede impactar el rendimiento del sistema en grandes cantidades.
- **XDocument:** Los métodos estáticos Load() y Save() pueden usarse para cargar o guardar un archivo XML directamente.
- **XMLReader / XMLWriter:** Son 2 clases que manipulan XML a bajo nivel. Si bien hay que escribir la estructura a mano

**Ejemplo:** Partiendo de la clase cliente y de una lista de objetos de tipo cliente generamos un archivo XML mediante la clase XMLSerializer.

```
public class Cliente
{
    public string Nombre { get; set; }
    public string Apellido { get; set; }
    public DateTime Fecha { get; set; }
}
```

```
List<Cliente> listaClientes = new List<Cliente>();
```

## XMLSerializer

En los casos donde no podamos definir el formato XML nosotros, esta es una excelente opción. Es decir, que los archivos XML que vayamos a leer hayan sido generados usando el método **XMLSerializer.Serialize()**. Si vamos a leer archivos XML externos, las otras opciones son mejores. Para serializar la lista de clientes ya creada:

```
XmlSerializer serializer = new XmlSerializer(typeof(List<Cliente>));
TextWriter textWriter = new StreamWriter("c:/clientes_serial.xml");
serializer.Serialize(textWriter, listaClientes);
textWriter.Close();
```

## ARCHIVOS XML

Como podemos ver, el proceso es muy simple. Se obtiene como resultado el siguiente XML:

```
<?xml version="1.0" encoding="utf-8"?>
<ArrayOfCliente xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
www.w3.org/2001/XMLSchema">
  <Cliente>
    <Nombre>Alberto</Nombre>
    <Apellido>Gonzalez</Apellido>
    <FechaRegistro>20-12-2015</FechaRegistro>
  </Cliente>
  <Cliente>
    <Nombre>Romina</Nombre>
    <Apellido>Fernandez</Apellido>
    <FechaRegistro>28-1-2016</FechaRegistro>
  </Cliente>
  <Cliente>
    <Nombre>Paul</Nombre>
    <Apellido>Rodriguez</Apellido>
    <FechaRegistro>2-5-2016</FechaRegistro>
  </Cliente>
</ArrayOfCliente>
```


Ahora, para poder leer el archivo XML y armar una nueva lista de Clientes:

```
XmlSerializer deserializer = new XmlSerializer(typeof(List<Cliente>));
TextReader textReader = new StreamReader("c:/clientes_serial.xml");
List<Cliente> listaSerial;
listaSerial = (List<Cliente>)deserializer.Deserialize(textReader)
textReader.Close();
```

## ARCHIVOS XML

Programar una aplicación que permita registrar clientes y realizar las consultas pertinentes de la agenda.

La agenda será salvaguardada en un fichero XML utilizando la clase [XmlSerializer](#)

 Clientes ×

Buscar

Nuevo

Modificar

Eliminar

Anterior

Siguiente

Guardar

Cancelar

Código:	<input type="text" value="1"/>	Razón social:	<input type="text" value="Iker Sertucha Lista"/>	CIF:	<input type="text" value="G15068091"/>
Compañía:	<input type="text" value="FAXPG"/>	Fecha alta:	<input type="text" value="02/11/2017"/> <div>15</div>		
Domicilio:	<input type="text" value="Félix Estrada Catoira, 3"/>	CP:	<input type="text" value="15007"/>	País:	<div>España</div>
Autonomía:	<div>Galicia</div>	Provincia:	<div>A Coruña</div>	Localidad:	<div>A Coruña</div>
Teléfono:	<input type="text" value="981169336"/>	Móvil:	<input type="text" value="981169336"/>	Email:	<input type="text" value="faxpg@faxpg.es"/>