**How to: Override Encoded SOAP XML Serialization**

**.NET Framework 4**

The process for overriding XML serialization of objects as SOAP messages is similar to the process for overriding standard XML serialization. For information about overriding standard XML serialization, see [How to: Specify an Alternate Element Name for an XML Stream](http://msdn.microsoft.com/en-us/library/athddy89.aspx).

**To override serialization of objects as SOAP messages**

1. Create an instance of the [SoapAttributeOverrides](http://msdn.microsoft.com/en-us/library/system.xml.serialization.soapattributeoverrides.aspx) class.
2. Create a **SoapAttributes** for each class member that is being serialized.
3. Create an instance of one or more of the attributes that affect XML serialization, as appropriate, to the member being serialized. For more information, see "Attributes That Control Encoded SOAP Serialization".
4. Set the appropriate property of **SoapAttributes** to the attribute created in step 3.
5. Add **SoapAttributes** to **SoapAttributeOverrides**.
6. Create an **XmlTypeMapping** using the **SoapAttributeOverrides**. Use the **SoapReflectionImporter.ImportTypeMapping** method.
7. Create an **XmlSerializer** using **XmlTypeMapping**.
8. Serialize or deserialize the object.

**Example**

The following code example serializes a file in two ways: first, without overriding the XmlSerializer class's behavior, and second, by overriding the behavior. The example contains a class named Group with several members. Various attributes, such as the SoapElementAttribute, have been applied to class members. When the class is serialized with the SerializeOriginal method, the attributes control the SOAP message content. When the SerializeOverride method is called, the behavior of the XmlSerializer is overridden by creating various attributes and setting the properties of a SoapAttributes to those attributes (as appropriate).

**C#**

[Copy](javascript:CodeSnippet_CopyCode('CodeSnippetContainerCode_f9611dfd-7571-4b30-a76c-d6094aded8f5');" \o "Copy to clipboard.)

using System;

using System.IO;

using System.Xml;

using System.Xml.Serialization;

using System.Xml.Schema;

public class Group

{

[SoapAttribute (Namespace = "http://www.cpandl.com")]

public string GroupName;

[SoapAttribute(DataType = "base64Binary")]

public Byte [] GroupNumber;

[SoapAttribute(DataType = "date", AttributeName = "CreationDate")]

public DateTime Today;

[SoapElement(DataType = "nonNegativeInteger", ElementName = "PosInt")]

public string PostitiveInt;

// This is ignored when serialized unless it is overridden.

[SoapIgnore]

public bool IgnoreThis;

public GroupType Grouptype;

[SoapInclude(typeof(Car))]

public Vehicle myCar(string licNumber)

{

Vehicle v;

if(licNumber == "")

{

v = new Car();

v.licenseNumber = "!!!!!!";

}

else

{

v = new Car();

v.licenseNumber = licNumber;

}

return v;

}

}

public abstract class Vehicle

{

public string licenseNumber;

public DateTime makeDate;

}

public class Car: Vehicle

{

}

public enum GroupType

{

// These enums can be overridden.

small,

large

}

public class Run

{

public static void Main()

{

Run test = new Run();

test.SerializeOriginal("SoapOriginal.xml");

test.SerializeOverride("SoapOverrides.xml");

test.DeserializeOriginal("SoapOriginal.xml");

test.DeserializeOverride("SoapOverrides.xml");

}

public void SerializeOriginal(string filename)

{

// Creates an instance of the XmlSerializer class.

XmlTypeMapping myMapping =

(new SoapReflectionImporter().ImportTypeMapping(

typeof(Group)));

XmlSerializer mySerializer =

new XmlSerializer(myMapping);

// Writing the file requires a TextWriter.

TextWriter writer = new StreamWriter(filename);

// Creates an instance of the class that will be serialized.

Group myGroup = new Group();

// Sets the object properties.

myGroup.GroupName = ".NET";

Byte [] hexByte = new Byte[2]{Convert.ToByte(100),

Convert.ToByte(50)};

myGroup.GroupNumber = hexByte;

DateTime myDate = new DateTime(2002,5,2);

myGroup.Today = myDate;

myGroup.PostitiveInt= "10000";

myGroup.IgnoreThis=true;

myGroup.Grouptype= GroupType.small;

Car thisCar =(Car) myGroup.myCar("1234566");

// Prints the license number just to prove the car was created.

Console.WriteLine("License#: " + thisCar.licenseNumber + "\n");

// Serializes the class and closes the TextWriter.

mySerializer.Serialize(writer, myGroup);

writer.Close();

}

public void SerializeOverride(string filename)

{

// Creates an instance of the XmlSerializer class

// that overrides the serialization.

XmlSerializer overRideSerializer = CreateOverrideSerializer();

// Writing the file requires a TextWriter.

TextWriter writer = new StreamWriter(filename);

// Creates an instance of the class that will be serialized.

Group myGroup = new Group();

// Sets the object properties.

myGroup.GroupName = ".NET";

Byte [] hexByte = new Byte[2]{Convert.ToByte(100),

Convert.ToByte(50)};

myGroup.GroupNumber = hexByte;

DateTime myDate = new DateTime(2002,5,2);

myGroup.Today = myDate;

myGroup.PostitiveInt= "10000";

myGroup.IgnoreThis=true;

myGroup.Grouptype= GroupType.small;

Car thisCar =(Car) myGroup.myCar("1234566");

// Serializes the class and closes the TextWriter.

overRideSerializer.Serialize(writer, myGroup);

writer.Close();

}

public void DeserializeOriginal(string filename)

{

// Creates an instance of the XmlSerializer class.

XmlTypeMapping myMapping =

(new SoapReflectionImporter().ImportTypeMapping(

typeof(Group)));

XmlSerializer mySerializer =

new XmlSerializer(myMapping);

TextReader reader = new StreamReader(filename);

// Deserializes and casts the object.

Group myGroup;

myGroup = (Group) mySerializer.Deserialize(reader);

Console.WriteLine(myGroup.GroupName);

Console.WriteLine(myGroup.GroupNumber[0]);

Console.WriteLine(myGroup.GroupNumber[1]);

Console.WriteLine(myGroup.Today);

Console.WriteLine(myGroup.PostitiveInt);

Console.WriteLine(myGroup.IgnoreThis);

Console.WriteLine();

}

public void DeserializeOverride(string filename)

{

// Creates an instance of the XmlSerializer class.

XmlSerializer overRideSerializer = CreateOverrideSerializer();

// Reading the file requires a TextReader.

TextReader reader = new StreamReader(filename);

// Deserializes and casts the object.

Group myGroup;

myGroup = (Group) overRideSerializer.Deserialize(reader);

Console.WriteLine(myGroup.GroupName);

Console.WriteLine(myGroup.GroupNumber[0]);

Console.WriteLine(myGroup.GroupNumber[1]);

Console.WriteLine(myGroup.Today);

Console.WriteLine(myGroup.PostitiveInt);

Console.WriteLine(myGroup.IgnoreThis);

}

private XmlSerializer CreateOverrideSerializer()

{

SoapAttributeOverrides mySoapAttributeOverrides =

new SoapAttributeOverrides();

SoapAttributes soapAtts = new SoapAttributes();

SoapElementAttribute mySoapElement = new SoapElementAttribute();

mySoapElement.ElementName = "xxxx";

soapAtts.SoapElement = mySoapElement;

mySoapAttributeOverrides.Add(typeof(Group), "PostitiveInt",

soapAtts);

// Overrides the IgnoreThis property.

SoapIgnoreAttribute myIgnore = new SoapIgnoreAttribute();

soapAtts = new SoapAttributes();

soapAtts.SoapIgnore = false;

mySoapAttributeOverrides.Add(typeof(Group), "IgnoreThis",

soapAtts);

// Overrides the GroupType enumeration.

soapAtts = new SoapAttributes();

SoapEnumAttribute xSoapEnum = new SoapEnumAttribute();

xSoapEnum.Name = "Over1000";

soapAtts.SoapEnum = xSoapEnum;

// Adds the SoapAttributes to the

// mySoapAttributeOverridesrides.

mySoapAttributeOverrides.Add(typeof(GroupType), "large",

soapAtts);

// Creates a second enumeration and adds it.

soapAtts = new SoapAttributes();

xSoapEnum = new SoapEnumAttribute();

xSoapEnum.Name = "ZeroTo1000";

soapAtts.SoapEnum = xSoapEnum;

mySoapAttributeOverrides.Add(typeof(GroupType), "small",

soapAtts);

// Overrides the Group type.

soapAtts = new SoapAttributes();

SoapTypeAttribute soapType = new SoapTypeAttribute();

soapType.TypeName = "Team";

soapAtts.SoapType = soapType;

mySoapAttributeOverrides.Add(typeof(Group),soapAtts);

// Creates an XmlTypeMapping that is used to create an instance

// of the XmlSerializer class. Then returns the XmlSerializer.

XmlTypeMapping myMapping = (new SoapReflectionImporter(

mySoapAttributeOverrides)).ImportTypeMapping(typeof(Group));

XmlSerializer ser = new XmlSerializer(myMapping);

return ser;

}

}