Complex XML Serialization

Source: http://www.tech−archive.net/Archive/DotNet/microsoft.public.dotnet.xml/2006−12/msg00091.html

- *From*: Eivind Gussiås Løkseth <eloekset@xxxxxxx>
- *Date*: Sun, 10 Dec 2006 23:04:52 +0100

I've been reading Keith Pijanowski's article − Enrich Your XML Serialization With Schema Providers In The .NET Framework − on how to do custom serialization of objects. Link to the article: http://msdn.microsoft.com/msdnmag/issues/06/06/ClassToContract/default.aspx

Now I'm trying to do the same ting in a more complex scenario, with two nested classes. The nested class is used in a collection property of the main class, and I want that collection to be automatically serialized. Here is a copy of the source without properties and contructors:

```
namespace MeritAdmin.Customer
{
[XmlRoot(ElementName = "application", IsNullable = false, Namespace =
"http://meritconsulting.no/MovexApplicationProperties.xsd)]
[XmlSchemaProvider("GetSchemaFile")]
public class MovexApplicationProperties : IXmlSerializable
{
public const string ApplicationType = "movex";
public const int Version = 2;
private string _administratorUser;
private string _administratorPassword;
private string _servicePack;
private string _movexVersion;
private List<MeritAdmin.Customer.MovexApplicationProperties.Environment> _environments = new
List<MeritAdmin.Customer.MovexApplicationProperties.Environment>();

#region XmlSchema

public static XmlSchemaComplexType GetSchemaFile(XmlSchemaSet schemaSet)
{
string xsdFile = Directory.GetCurrentDirectory() + "\\Customer\\MovexApplicationProperties.xsd";
XmlSerializer schemaSerializer = new XmlSerializer(typeof(XmlSchema));
XmlSchema schema = (XmlSchema)schemaSerializer.Deserialize(XmlReader.Create(xsdFile));
schemaSet.Add(schema);

// Target namespace
string tns = "http://meritconsulting.no/MovzexApplicationProperties.xsd;
XmlQualifiedName application = new XmlQualifiedName("application", tns);
XmlSchemaComplexType applicationType = (XmlSchemaComplexType)schema.SchemaTypes[application];
return applicationType;
}
```

```
#endregion

#region IXmlSerializable Members

public System.Xml.Schema.XmlSchema GetSchema()
{
throw new Exception("The method or operation is not implemented.");
}

public void ReadXml(System.Xml.XmlReader reader)
{
throw new Exception("The method or operation is not implemented.");
}

public void WriteXml(System.Xml.XmlWriter writer)
{
string ns = "http://meritconsulting.no/MovexApplicationProperties.xsd;
writer.WriteAttributeString("type", MovexApplicationProperties.ApplicationType);
writer.WriteElementString("version", MovexApplicationProperties.Version.ToString());
writer.WriteElementString("administratorUser", this.AdministratorUser);
writer.WriteElementString("administratorPassword", this.AdministratorPassword);
writer.WriteElementString("servicePack", this.ServicePack);
writer.WriteElementString("movexVersion", this.MovexVersion);
writer.WriteElementString("environments", ns, string.Empty);

// This is not the right way to do it
//foreach (Environment environment in this.Environments)
//{
// environment.WriteXml(writer);
//}
}

#endregion


[XmlRoot(ElementName = "environment", IsNullable = true, Namespace =
"http://meritconsulting.no/MovexApplicationProperties.xsd)]
[XmlSchemaProvider("GetSchemaFile")]
public class Environment : IXmlSerializable
{
private string _name;
private string _portNumber;

#region XmlSchema

public static XmlSchemaComplexType GetSchemaFile(XmlSchemaSet schemaSet)
{
string xsdFile = Directory.GetCurrentDirectory() + "\\Customer\\MovexApplicationProperties.xsd";
XmlSerializer schemaSerializer = new XmlSerializer(typeof(XmlSchema));
XmlSchema schema = (XmlSchema)schemaSerializer.Deserialize(XmlReader.Create(xsdFile));
```

```
schemaSet.Add(schema);

// Target namespace
string tns = "http://meritconsulting.no/MovexApplicationProperties.xsd;
XmlQualifiedName environment = new XmlQualifiedName("environment", tns);
XmlSchemaComplexType environmentType =
(XmlSchemaComplexType)schema.SchemaTypes[environment];
return environmentType;
}

#endregion

#region IXmlSerializable Members

public XmlSchema GetSchema()
{
throw new Exception("The method or operation is not implemented.");
}

public void ReadXml(XmlReader reader)
{
throw new Exception("The method or operation is not implemented.");
}

public void WriteXml(XmlWriter writer)
{
string ns = "http://meritconsulting.no/MovexApplicationProperties.xsd;
writer.WriteElementString("name", ns, this.Name);
writer.WriteElementString("portNumber", ns, this.PortNumber);
}

#endregion
}
}
}
```

And this is how I'm trying to serialize some test objects:

```
private void Form1_Load(object sender, EventArgs e)
{
MovexApplicationProperties movex = new MovexApplicationProperties("test", "test", "12", "v12Java");
movex.Environments.Add(new MovexApplicationProperties.Environment("TST", "16800"));
List<Type> extraTypes = new List<Type>();
extraTypes.Add(new MovexApplicationProperties.Environment().GetType());
XmlSerializer serializer = new XmlSerializer(movex.GetType(), extraTypes.ToArray());
StringWriter stringWriter = new StringWriter();
serializer.Serialize(stringWriter, movex);
this.textBox1.Text = stringWriter.ToString();
}
```

Here is the result:

```
<?xml version="1.0" encoding="utf−16"?>
```

<application type="movex" xmlns="http://meritconsulting.no/MovexApplicationProperties.xsd>
<version>2</version>
<administratorUser>test</administratorUser>
<administratorPassword>test</administratorPassword>
<servicePack>12</servicePack>
<movexVersion>v12Java</movexVersion>
<environments />
</application>

As you can see, the environments tag isn't aware of the contents of the collection, and the environment doesn't get serialized even though I've passed the Environment type as extra types to the XmlSerializer. What is the right way to do this?

Eivind

.