

Re: XmlSerializer: deserialize against xsd generated class

Source: <http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.xml/2008-08/msg00003.html>

- *From:* j.a. harriman <jeffrey_no_spam_alias@xxxxxxxxxxxxxxx>
 - *Date:* Fri, 1 Aug 2008 09:05:01 -0700
-

I may have tried to oversimplify my example.

The "real" XML has a number of "xmlns:" entries following the "parentnode".
When I remove them & test, this statement,
"/parentnode/secondchildnode//*[not(node())]", works as I was expecting.

Is there something I need to do with adding namespaces (e.g
XmlNamespaceManager) or something else that I am unaware of?

If there are 5 namespaces in the XML do I need to add a namespace for each
one?

How are the namespaces used in the XPath statement above?

Thanks.

Jeff

"j.a. harriman" wrote:

Building off your example, here's a representation of my actual XML:

```
<parentnode>
<firstchildnode>
<a/>
</firstchildnode>
<secondchildnode>
<b/>
<c>
<d/>
<e>12345</e>
</c>
</secondchildnode>
</parentnode>
```

The XPath expression, "//*[not(node())]", removes all "empty" elements. I
need it to *only* remove "b" & "d" above. "a" would be left alone.

Re: XmlSerializer: deserialize against xsd generated class

Thanks.

Jeff

"Sprotty" wrote:

You could remove the empty elements in the DOM before loading it as you suggest.

Im not seeing a problem with that

```
XmlDocument xmlDoc = new XmlDocument();  
xmlDoc.LoadXml("<parentnode><secondchildnode><b/></secondchildnode></parentnode>");  
XmlNodeList xnl = xmlDoc.SelectNodes("//*[@not(node())]");  
Debug.Assert(xnl.Count == 1);
```

or you could remove it from the object model after its been loaded (deserialized).

If you were using Liquid XML then you may be able to add code in that would remove the empty elements as part of the de-serialization.

Cheers Simon

On 31 Jul, 21:22, j.a. harriman <jeffrey_no_spam_al...@xxxxxxxxxxxxxxxx> wrote:

Simon,

Thanks for the link, I will take a look at it as it may be something we could use.

In the meantime, I did find a code snippet that removes the "empty" nodes but it removes ALL the empty nodes in the XML and I didn't state (earlier) that I want to do this beginning with a particular child node only.

When I use the following XPath statement in XMLSpy's XPath Evaluator, it locates the "empty" node mentioned in the previous posts: /parentnode/secondchildnode//*[not(node())]

When I use the same statement in my C# code, it will not locate any empty nodes "under" the "secondchildnode" (count = 0):

Re: XmlSerializer: deserialize against xsd generated class

```
XmlNodeList element =  
doc.SelectNodes("/parentnode/secondchildnode//*[not(node())]");
```

The plan is to loop through "element" and remove the nodes using the "ParentNode.RemoveChild" of an XmlNode – setting it to each array element, etc.

Any ideas as to why this is not working in .NET VS2005 2.0 framework?

Thanks.

Jeff

"Sprotty" wrote:

Hi Jeff

The code produced by xsd.exe, copes well with simple schema's, but if your working against anything a bit complicated then it can cause issues.

If you continue to have problems then I suggest you take a look at Liquid XML Data Binder

http://www.liquid-technologies.com/Product_XmlDataBinding.aspx

This supports much more of the xsd standard, and the objects are more strongly typed (it also comes with a free XSD editor).

Hope this helps

Re: XmlSerializer: deserialize against xsd generated class

Regards Simon

On 23 Jul, 08:00, "Joe Fawcett"
<joefawc...@xxxxxxxxxxxxxxxxxxx> wrote:

Jeff

IXmlSerializable is
reasonably straightforward.
The newer version needs
three methods, ReadXml,
WriteXml and one to find
the
Schema which is pointed to
by an attribute, I normally
just include the
schema as an embedded
resource.

In ReadXml you get an
XmlReader containing the
XML and use it to populate
the object's fields, either
reading it directly or loading
it into a

DomDocumnt/XPathDocument
if that's easier. WriteXml
takes the fields and
creates an XML document.

There is an example
here:<http://www.devx.com/dotnet/Article/29720>

--

Joe Fawcett (MVP –
XML)<http://joe.fawcett.name>

"j.a. harriman"
<jeffrey_no_spam_al...@xxxxxxxxxxxxxxxxxxx>
wrote in message

Re: XmlSerializer: deserialize against xsd generated class

news:046D4993-D7DE-405A-814C-F5B86AC29005@xxxxxxxxxxxxxxxxxxxx

Thanks for
the answer
Joe.

I also found
some
examples of
using a
style sheet
and doing a
transformation
(Using
XslCompiledTransform)
of the
original
XmlDocument.
I've run a
test
and it seems
to work in
that the
result is as
if the
original
message
hadn't
had them in
there. Are
there any
reasons why
not to
implement
this?

Also, I tried
to locate
good
"beginner"
examples of
using
IXmlSerializable
that might
be similar

Re: XmlSerializer: deserialize against xsd generated class

to what I
need to do,
but came up
short. I
would
like
to to look at
this in
further
detail.

Do you
have any
links to
examples or
are you
aware if any
of the
Microsoft
example
downloads
(such as
SDK) have
any?

Thanks. Jeff

"Joe
Fawcett"
wrote:

Unless
you
want
to
implement
IXmlSerializable
then
I
don't
think
you
can

Re: XmlSerializer: deserialize against xsd generated class

do
what
you
want.
You
can
specify
that
a
null
element
is
shown
as:
<myElement
xsi:nil="true"/>
rather
than
being
omitted
entirely
by
adding
the
XmlElement(IsNullable
=
true)
attribute
to
the
field/property.

Joe
Fawcett
(MVP
-
XML)

<http://joe.fawcett.name>

Re: XmlSerializer: deserialize against xsd generated class

"j.a.
harriman"
<jeffrey_no_spam_al...@xxxxxxxxxxxxxx>
wrote
in
message
news:697F7C61-82AC-4387-8F04-57D200C7C703@xxxx

Hi,

I
have
a
schema
that
has
an
optional
element,
fieldTag4000Field.
If
the
element
is
omitted
from
the
XML
request,
when
it
is
deserialized,
it
will
be
null
when
I
check
it
–
which
is
fine.

Re: XmlSerializer: deserialize against xsd generated class

What happens when the element is supplied as `<fieldTag4000Field/>` (empty), it does not equate to null. I want to be able to handle this at the deserialization level rather than in my edits later.

Is there a way to alter the behavior so when I deserialize it, it is

Re: XmlSerializer: deserialize against xsd generated class

null?

I
also
want
to
add
that
I
am
calling
the
"CanDeserialize"
method
of
the
XmlSerializer
object,
passing
in
the
request
as
a
XmlNodeReader.

Thanks.

The
following
C#
class
snippet
was
generated
by
xsd.exe
V2:

```
[System.CodeDom.Compiler.GeneratedCodeAttribute("2.0.50727.42")]  
[System.SerializableAttribute()]  
[System.Diagnostics.DebuggerStepThroughAttribute]
```

Re: XmlSerializer: deserialize against xsd generated class

```
[System.ComponentModel.DesignerCategoryAttribute("debug")]  
[System.Xml.Serialization.XmlTypeAttribute(AnonymousType = true,  
Namespace = "MyNamespace")]  
public  
partial  
class  
Request_TypeMessageType  
{
```

```
private  
Request_TypeMessageTypeFieldTag4000  
fieldTag4000Field;
```

```
public  
Request_TypeMessageTypeFieldTag4000  
FieldTag4000  
{  
get  
{  
return  
this.fieldTag4000Field;  
}  
set  
{  
this.fieldTag4000Field  
=  
value;  
}  
}  
}
```

```
[System.CodeDom.Compiler.GeneratedCodeAttribute("Microsoft.VisualStudio.  
"2.0.50727.42")]  
[System.SerializableAttribute()]  
[System.Diagnostics.DebuggerStepThroughAttribute()]  
[System.ComponentModel.DesignerCategoryAttribute("debug")]  
[System.Xml.Serialization.XmlTypeAttribute(AnonymousType = true,  
Namespace = "MyNamespace")]  
public  
partial  
class  
Request_TypeMessageTypeFieldTag4000  
{
```

Re: XmlSerializer: deserialize against xsd generated class

```
private
IDType_Type
intermediaryIDCodeField;
```

```
private
bool
intermediaryIDCodeFieldSpecified;
```

```
private
string
intermediaryIdentifierField;
```

```
public
IDType_Type
IntermediaryIDCode
{
get
{
return
this.intermediaryIDCodeField;
}
set
{
this.intermediaryIDCodeField
=
value;
}
}
```

```
[System.Xml.Serialization.XmlIgnoreAttribute()]
public
bool
IntermediaryIDCodeSpecified
{
get
{
return
this.intermediaryIDCodeFieldSpecified;
}
set
{
this.intermediaryIDCodeFieldSpecified
=
```

Re: XmlSerializer: deserialize against xsd generated class

```
value;  
}  
}
```

```
public  
string  
IntermediaryIdentifier  
{  
get  
{  
return  
this.intermediaryIdentifierField;  
}  
set  
{  
this.intermediaryIdentifierField  
=  
value;  
}  
}  
}
```

```
[System.CodeDom.Compiler.GeneratedCodeAttribute  
"2.0.50727.42")  
[System.SerializableAttribute()]  
[System.Xml.Serialization.XmlTypeAttribute(Names  
public  
enum  
IDType_Type  
{
```

B,

C,

D,

F,

Re: XmlSerializer: deserialize against xsd generated class

U,
}-
Hide
quoted
text
-

- Show quoted text -