

RE: Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

Source:

<http://www.tech-archive.net/Archive/BizTalk/microsoft.public.biztalk.server/2006-10/msg00033.html>

- *From:* zachbonham <zachbonham@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 20 Oct 2006 09:55:01 -0700
-

Michael,

Your right, it was that we were trying to access a property that didn't exist for a message! Somewhat indirectly it seems.. :)

While we didn't find anything specific in the orchestrations accessing a context property from a message part, we did find is that our InvalidPropertyTypeException was still due to a configuration problem.

Some time ago we found that if we enabled tracking for all elements in our custom property schemas, tracking occurred and we didn't have to flip the switch for tracking on all of our messages (via HAT | Configuration | Messages).

While this quickly allowed us to 'track' everything, what we just found is that tracking was attempting to occur for all properties in the property schema for ALL documents. This was not what we intended; everytime the runtime tried to track data for a document that the property wasn't valid for, we'd get an InvalidPropertyTypeException. This could very easily occur many times for a document.

Simply turning off the tracking at the property schema level ,and enabling at the message level, cleared up the issue.

Thanks again!
-Z

A bit more wordy post can be found here:

http://zachbonham.blogspot.com/2006_10_15_archive.html#2013042149553117765

"Michael Elizarov [MSFT]" wrote:

RE: Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

Hi,

This exception is thrown in the following cases:

- (1) Trying to access a data property on a message for which it has not been defined (should be caught by compiler)
- (2) Trying to set the value of the immutable `Size` part context property
- (3) Trying to retrieve the value of a non-part context property from a message part (probably this one, they are doing something like `msg.part(BTS.AckId)` where they should be doing `msg(BTS.AckId)`).

Yes the exception is of significant concern; it is a major user error. Any exception thrown could delay the cleanup of resources, especially unmanaged objects/handles that need to be disposed.

However, out-of-memory errors are usually due to other causes, like the protocol was not designed properly to scale or user code is causing the OOM. For example, users will sometimes do custom .net xslt like:

For example, the following code leaks memory:

```
for(int i=0;i<1000;i++)
{
  xslt.Load(stylesheet);
  //Do other stuff
  xslt.Transform(doc, null, writer);
}
```

Change the code as follows to load XSLT only once and reuse it in a loop:

```
xslt.Load(stylesheet);
for(int i=0;i<1000;i++)
{
  //Do other stuff
  xslt.Transform(doc, null, writer);
}
```

See <http://support.microsoft.com/?id=316775>

HTH.

-- Michael

We receive
Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

during

RE: Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

RE: Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

what appear to be tracking related activities (see call stack below).

However, they still occur when we turn tracking off.

Is this 'exception' of a significant concern? The process doesn't crash, the messages still get delivered, and the tracking data gets tracked

(when

on). However, I'm wondering if there are resources not being cleaned up properly which would tie it to our out of memory issue?

Any suggestions, comments, or info appreciated!

Summary info below...

Thanks!
Zach

Background:
This was identified while attempting to discover why we have a BizTalk

2004

host instance, that contains only orchestration related resources, is experiencing an Out of Memory error in our production environment. There

has

been additional load placed on the components involved over the last few months and it has become an almost daily occurrence.

Using Identify AppSight to record against the process I saw an alarming number of exceptions occurring, however, I wasn't always getting the

exception

type information. I still have an outstanding number of exceptions, primarily unknown or System.FormatExceptions that I'm

RE: Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

running down.

However, dropping out to using windows debugging tools/adplus running in crash mode, dumping on first chance exceptions turned up an awful number

of

instances where the exception turns out to be this InvalidPropertyTypeException.

The !clrstack always looks like this:

```
0:011> !clrstack
Thread 11
ESP EIP
0b79f608 77e55dea [FRAME: HelperMethodFrame]
0b79f634 1613c135 [DEFAULT] [hasThis] Class
Microsoft.XLANGs.RuntimeTypes.MessagePropertyDefinition
Microsoft.XLANGs.Core.XMessage._getMessagePropertyDefinition(Class
System.Type)
0b79fe40 79babd2b [FRAME: ContextTransitionFrame]
```

and the !dumpstack is pretty consistent to this:

```
0:011> !dumpstack
Current frame: kernel32!RaiseException+0x3c
ChildEBP RetAddr Caller, Callee
0b79f520 77e55dea kernel32!RaiseException+0x3c, calling
*** ERROR: Symbol
file could not be found. Defaulted to export symbols for
ntdll.dll -
ntdll!RtlRaiseException
0b79f528 791c6e1f *** ERROR: Symbol file could not be
found. Defaulted
```

to

```
export symbols for mscorsvr.dll -
mscorlib!Ordinal76+0x16e1f, calling
mscorlib!Ordinal76+0x16da0
0b79f53c 791c6e90 mscorsvr!Ordinal76+0x16e90, calling
mscorlib!Ordinal76+0x16de2
0b79f554 791c6ea5 mscorsvr!Ordinal76+0x16ea5, calling
mscorlib!Ordinal76+0x16e7e
0b79f568 79216aed
mscorlib!GetAssemblyMDImport+0x2e1d4, calling
kernel32!RaiseException
0b79f588 799a3eb4 (MethodDesc 0x79b96470 +0xf4
System.Text.StringBuilder.Append), calling
```

RE: Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

RE: Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

mscorsvr!Ordinal76+0x1d925
0b79f5a8 791b637b mscorsvr!Ordinal76+0x637b, calling

mscorsvr!Ordinal76+0x6335

0b79f5c0 7924c8d0
mscorsvr!GetMetaDataPublicInterfaceFromInternal+0x5d70,
calling mscorsvr!GetAssemblyMDIImport+0x2e16f
0b79f5f0 79996f37 (MethodDesc 0x79b96400 +0x67
System.Text.StringBuilder.ToString), calling (MethodDesc
0x79b952d0
System.String.ClearPostNullChar)
0b79f600 7924c89a
mscorsvr!GetMetaDataPublicInterfaceFromInternal+0x5d3a,
calling mscorsvr!Ordinal76+0x63f8
0b79f62c 1613c135 (MethodDesc 0x14473af8 +0x3d
Microsoft.XLANGs.Core.XMessage._getMessagePropertyDefinition),
calling
mscorsvr!GetMetaDataPublicInterfaceFromInternal+0x5d24
0b79f638 1613c09f (MethodDesc 0x14473b18 +0x17
Microsoft.XLANGs.Core.XMessage.GetContentProperty),
calling (MethodDesc
0x14473af8
Microsoft.XLANGs.Core.XMessage._getMessagePropertyDefinition)
0b79f644 16133d6f (MethodDesc 0x14473a48 +0x37
Microsoft.XLANGs.Core.XMessage.GetPropertyValue),
calling (MethodDesc
0x14473b18
Microsoft.XLANGs.Core.XMessage.GetContentProperty)
0b79f654 16133cf4 (MethodDesc 0x14473e38 +0xc
Microsoft.BizTalk.XLANGs.BTXEngine.BTXMessage.GetPropertyValue),
calling
(MethodDesc 0x14473a48
Microsoft.XLANGs.Core.XMessage.GetPropertyValue)
0b79f660 16133cb4 (MethodDesc 0x15964f98 +0xdc
Microsoft.XLANGs.Core.MessageWrapperForTracking.GetPropertyValue)
0b79f674 1613bc2a (MethodDesc 0x1447c720 +0x392
Microsoft.XLANGs.Mozart.NativeInterceptor.GetPromotedProperties)
0b79f6dc 161308b0 (MethodDesc 0x1447c6b0 +0x348
Microsoft.XLANGs.Mozart.NativeInterceptor.BasicTracking),
calling

(MethodDesc

0x1447c720

Microsoft.XLANGs.Mozart.NativeInterceptor.GetPromotedProperties)

0b79f784 161304a9 (MethodDesc 0x1447c550 +0x41
Microsoft.XLANGs.Mozart.NativeInterceptor.TrackEventHandler),
calling

RE: Microsoft.XLANGs.RuntimeTypes.InvalidPropertyTypeException

(MethodDesc 0x1447c6b0
Microsoft.XLANGs.Mozart.NativeInterceptor.BasicTracking)
0b79f79c 16130125 (MethodDesc 0x14477178 +0xcd
Microsoft.XLANGs.Core.Events.FireEvent)
0b79f7fc 1448d484 (MethodDesc 0x14471d30 +0x45c
SupplyChain.BizTalk.Orchestrations.SendJDEShipConfirm.segment1),
calling
(MethodDesc 0x14477178
Microsoft.XLANGs.Core.Events.FireEvent)
0b79f818 77e6bbf5 kernel32!LocalAlloc+0x63, calling
kernel32!SwitchToFiber+0x1a9
0b79f82c 77e6bbf5 kernel32!LocalAlloc+0x63, calling
kernel32!SwitchToFiber+0x1a9
0b79f830 791ca73b mscorsvr!Ordinal76+0x1a73b, calling
kernel32!LocalAlloc
0b79f85c 7c820e5d
ntdll!RtlQueryInformationActivationContext+0xef,

calling

ntdll!RtlQueryInformationActivationContext+0x123
0b79f878 7c820e78
ntdll!RtlQueryInformationActivationContext+0x10a,

calling

ntdll!RtlGetLastWin32Error+0x1a1
0b79f8c4 7c820e78
ntdll!RtlQueryInformationActivationContext+0x10a,

calling

ntdll!RtlGetLastWin32Error+0x1a1
0b79f8c8 77e42b72 kernel32!QueueUserAPC+0x27, calling
ntdll!RtlQueryInformationActivationContext
0b79f8d0 7c821b54 ntdll!ZwQueueApcThread+0xc
[...shortened for brevity...]