

RE: Setting category / event ID when tracing in VS2008

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vb/2008-02/msg00090.html>

- *From:* wawang@xxxxxxxxxxxxxxxxxxxxxxxx ("Walter Wang [MSFT]")
 - *Date:* Sat, 02 Feb 2008 10:13:12 GMT
-

Hi Tommy,

I'm glad to hear that the issue is now resolved.

Sorry about my last two replies, they were all truncated and I just found out it's caused by my tool used to post the reply, apparently it uses a single line with only "." as EOF.

I will post my reply again for completeness.

To learn how to specify category for the event log entry, we need to first learn some background on how event log works. I will leave the details to MSDN ([http://msdn2.microsoft.com/en-us/library/aa363632\(VS.85\).aspx](http://msdn2.microsoft.com/en-us/library/aa363632(VS.85).aspx)). Some key quoted content:

Message Files (Windows)
(<http://msdn2.microsoft.com/en-us/library/aa363669.aspx>)

<blockquote>
Each event source should register message files that contain description strings for each event identifier, event category, and parameter. Register these files in the EventMessageFile, CategoryMessageFile, and ParameterMessageFile registry values for the event source.
</blockquote>

Here our focus is on the CategoryMessageFile, which is a message text file ([http://msdn2.microsoft.com/en-us/library/aa385646\(VS.85\).aspx](http://msdn2.microsoft.com/en-us/library/aa385646(VS.85).aspx)) to define our category. For example, save following content (including the 4th line with only a dot) in a file (for example: cats.mc):

```
MessageId=0x1
Language=English
MyCategory1
(should be a "." without quote)
```

RE: Setting category / event ID when tracing in VS2008

Start Visual Studio Command Prompt (I assume you have installed Visual C++), type following command:

```
mc cats.mc
rc cats.rc
link /dll /noentry /machine:X86 /out:cats.dll cats.res
```

This will create "cats.dll" in current directory which will be the CategoryMessageFile and contains only one category "MyCategory1".

Now we can create an installer (you can choose to add the installer to your exist assembly, later an administrator can use "installutil.exe" from .NET framework to install the event log source to registry):

VB.NET:

```
Imports System
Imports System.Configuration.Install
Imports System.Diagnostics
Imports System.ComponentModel

<RunInstaller(True)> _
Public Class MyEventLogInstaller
Inherits Installer
Public Sub New()
Me.myEventLogInstaller.Source = "MyLogSource"
Me.myEventLogInstaller.Log = "MyLogName"
Me.myEventLogInstaller.CategoryCount = 1
Me.myEventLogInstaller.CategoryResourceFile = "c:\temp\cats.dll"
MyBase.Installers.Add(Me.myEventLogInstaller)
End Sub

Public Shared Sub Main()
End Sub

Private myEventLogInstaller As EventLogInstaller = New EventLogInstaller
End Class
```

C#:

```
using System;
using System.Configuration.Install;
using System.Diagnostics;
using System.ComponentModel;

[RunInstaller(true)]
public class MyEventLogInstaller: Installer {
private EventLogInstaller myEventLogInstaller;
```

RE: Setting category / event ID when tracing in VS2008

```
public MyEventLogInstaller() {
myEventLogInstaller = new EventLogInstaller();
myEventLogInstaller.Source = "MyLogSource";
myEventLogInstaller.Log = "MyLogName";
myEventLogInstaller.CategoryCount = 1;
myEventLogInstaller.CategoryResourceFile = @"c:\temp\cats.dll";

Installers.Add(myEventLogInstaller);
}
public static void Main() { }
```

Compile this file:

VB.NET:

```
vbc MyInstaller.vb
```

C#:

```
csc MyInstaller.cs
```

Install it:

```
installutil /i MyInstaller.exe
```

This will create the event log source in registry, along with our customized category resource file.

(Note we have registered the event log source under customized log name "MyLogName", you can use "Application", "System" or "Security" if you need to save your event log under these default event logs.)

Now we can use the EventLog class in .NET Framework to write an event log using our customized event log source:

VB.NET:

```
Dim ev as new EventLog("MyLogName", System.Environment.MachineName,
"MyLogSource")
ev.WriteEntry("Hello", EventLogEntryType.Information, 123, 1)
```

C#:

```
EventLog ev = new EventLog("MyLogName", System.Environment.MachineName,
"MyLogSource");
ev.WriteEntry("Hello", EventLogEntryType.Information, 123, 1);
```

RE: Setting category / event ID when tracing in VS2008

This will create event log ID 123 with category 1 (which shows MyCategory1 in event log viewer).

Now the only task left is how to let the EventLogTraceListener to use our customized event log source when logging trace output. Unfortunately the EventLogTraceListener class is sealed, and internally it's always using category id 0 when using the EventLog class to write event log entry.

Fortunately it's not difficult to implement a custom TraceListener to workaround this. If you are not writing a general purpose EventLogTraceListener, we only need to override two methods:

VB.NET:

```
Public Class MyEventLogTraceListener
    Inherits TraceListener

    Public Overrides Sub Write(ByVal o As String)
        Me.ev.WriteEntry(o, EventLogEntryType.Information, 123, 1)
    End Sub

    Public Overrides Sub WriteLine(ByVal message As String)
        Me.Write(message)
    End Sub

    Private ev As EventLog = New EventLog("MyLogName",
        Environment.MachineName, "MyLogSource")
End Class
```

C#:

```
public class MyEventLogTraceListener : TraceListener
{
    private EventLog ev = new EventLog("MyLogName",
        Environment.MachineName, "MyLogSource");

    public override void Write(string o)
    {
        ev.WriteEntry(o, EventLogEntryType.Information, 123, 1);
    }

    public override void WriteLine(string message)
    {
        this.Write(message);
    }
}
```

I've simplified the code by hardcoding the event log entry type; you may refer to the default EventLogTraceListener's implementation using Reflector (<http://www.aisto.com/roeder/dotnet/>).

RE: Setting category / event ID when tracing in VS2008

Now you can add this customized TraceListener to output your trace output to event log with correct category:

VB.NET:

```
Dim tlEventLog as new MyEventLogTraceListener()  
Trace.Listeners.Clear()  
Trace.Listeners.Add(tlEventLog)
```

C#:

```
MyEventLogTraceLisenter tlEventLog = new MyEventLogTraceListener();  
Trace.Listeners.Clear();  
Trace.Listenres.Add(tlEventLog);
```

Regards,

Walter Wang (wawang@xxxxxxxxxxxxxxxxxxxxxx, remove 'online.')

Microsoft Online Community Support

=====
When responding to posts, please "Reply to Group" via your newsreader so that others may learn and benefit from your issue.
=====

This posting is provided "AS IS" with no warranties, and confers no rights.

.