

Log4Net使用详解（续）

标签: [log4net](#) [layout](#) [thread](#) [exception](#) [string](#) [newline](#)

2010-11-23 10:35

28903人阅读

[评论\(29\)](#)

[收藏](#)

[举报](#)

分类: [C#基础（110）](#) [asp.net（103）](#)

版权

声明：本文为博主原创文章，未经博主允许不得转载。

说明自从上次在2008年在博客上发表过有关log4net的用法介绍文章之后（网

址: <http://blog.csdn.net/zhoufoxcn/archive/2008/03/26/2220533.aspx>），有不少朋友在博文下留言询问一些细节，现在就一些比较普遍的问题做一些稍微深入的解答，希望大家满意。

首先说明一点的是，log4net解决的问题是提供一个记录日志的框架，它提供了向多种目标写入的实现，比如利用log4net可以方便地将日志信息记录到文件、控制台、Windows事件日志和数据库（包括MS SQL Server, Access, Oracle9i, Oracle8i, DB2, SQLite）中，一般来说我们只需要提供一个描述性的字符串，然后log4net就会自动提供有关运行时的一些信息。

Log4Net的版本仍是1.2.10（2008年我写博文的时候也是这个版本），有.NET1.0和.NET1.1和.NET2.0版本，如果有正在使用高于.NET2.0开发的也不用担心，可以直接引用这个类库，像在.NET2.0中开发一样，它的网址是: <http://logging.apache.org/log4net/>

关于在Web中支持的问题

在我们开发项目时都会使用到config文件，可以在config文件中配置log4net。这一点Web项目和WinForm项目都是一样的。需要注意的是，因为在Web项目中一般以较低权限的角色来运行Web项目的，所以在使用文件型日志时要注意不要放在本项目根文件夹之外。

在config文件中的配置

要使用log4net，首先要在config文件的<configSections>节点中增加配置（如果没有这个节点请手动增加），如下：

```
[xhtml]
01. <configSections>
02.     <section name="log4net" type="System.Configuration.IgnoreSectionHandler"/>
03. </configSections>
```

除此之外，还要在顶级节点<configuration>下增加<log4net>子节点。在<log4net>节点下就可以增加<appender>子节点，每个<appender>子节点代表一种记录日志的方式（仅在这里配置了不代表启用了）。

具体说来有如下Appender:

AdoNetAppender: 利用ADO.NET记录到数据库的日志。

AnsiColorTerminalAppender: 在ANSI 窗口终端写下高亮度的日志事件。 [\[载\]](#)

AspNetTraceAppender: 能用asp.net中Trace的方式查看记录的日志。

BufferingForwardingAppender: 在输出到子Appenders之前先缓存日志事件。

ConsoleAppender: 将日志输出到控制台。

EventLogAppender: 将日志写到Windows Event Log.

FileAppender: 将日志写到文件中。

LocalSyslogAppender: 将日志写到local syslog service (仅用于UNIX环境下).

MemoryAppender: 将日志存到内存缓冲区。

NetSendAppender: 将日志输出到Windows Messenger service.这些日志信息将在用户终端的对话框中显示。

RemoteSyslogAppender: 通过UDP网络协议将日志写到Remote syslog service。

RemotingAppender: 通过.NET Remoting将日志写到远程接收端。

RollingFileAppender: 将日志以回滚文件的形式写到文件中。

SmtpAppender: 将日志写到邮件中。

TraceAppender: 将日志写到.NET trace 系统。

UdpAppender: 将日志connectionless UDP datagrams的形式送到远程宿主或以UdpClient的形式广播。

关于使用**log4net**中可能会使用到的一些参数

%m(message):输出的日志消息，如**ILog.Debug(...)**输出的一条消息

%n(new line):换行

%d(datetime):输出当前语句运行的时刻

%r(run time):输出程序从运行到执行到当前语句时消耗的毫秒数

%t(thread id):当前语句所在的线程ID

%p(priority):日志的当前优先级别，即**DEBUG**、**INFO**、**WARN**...等

%c(class):当前日志对象的名称，例如：

%f(file):输出语句所在的文件名。

%l(line): 输出语句所在的行号。

%数字: 表示该项的最小长度，如果不够，则用空格填充，如“**%-5level**”表示**level**的最小宽度是5个字符，如果实际长度不够5个字符则以空格填充。

下面以一个实际的例子来说明问题，比如在配置中有“**%date [%thread] (%file:%line) %-5level %logger**

[%property{NDC}] - %message%newline”，那么实际的日志中会是如下格式：

“记录时间：2010-11-17 16:16:36,561 线程ID:[9] 日志级别：文件：所在行**ERROR** 出错类：

Log4NetDemo.Program property:[(null)] - 错误描述：error

System.Exception: 在这里发生了一个异常,Error Number:2036084948”

关于对数据库的支持

前面已经说过，**log4net**是支持包括**MS SQL Server**, **Access**, **Oracle9i**,**Oracle8i**,**DB2**,**SQLite**在内的数据库的，如果是文件型数据库（如**Access**或**SQLite**）的话就需要指定数据库文件的位置（在**Web**中最好指定在有读写权限的文件夹下，并且实现创建好表），如果是网络数据库就需要指定正确的数据库连接字符串。

比如要记录到**Oracle**数据库中，在配置文件中可以增加一个< appender>节点，配置如下：

```

01. <appender name="AdoNetAppender_Oracle" type="log4net.Appender.AdoNetAppender">
02.     <connectionType value="System.Data.OracleClient.OracleConnection, System.Data.OracleClient" />
03.     <connectionString value="data source=[mydatabase];User ID=[user];Password=[password]" />
04.     <commandText value="INSERT INTO Log (Datetime,Thread,Log_Level,Logger,Message) VALUES (:log_date, :thread, :log_level, :logger, :message)" />
05.     <bufferSize value="128" />
06.     <parameter>
07.         <parameterName value=":log_date" />
08.         <dbType value="DateTime" />
09.         <layout type="log4net.Layout.RawTimeStampLayout" />
10.     </parameter>
11.     <parameter>
12.         <parameterName value=":thread" />
13.         <dbType value="String" />
14.         <size value="255" />
15.         <layout type="log4net.Layout.PatternLayout">
16.             <conversionPattern value="%thread" />
17.         </layout>
18.     </parameter>
19.     <parameter>
20.         <parameterName value=":log_level" />
21.         <dbType value="String" />
22.         <size value="50" />
23.         <layout type="log4net.Layout.PatternLayout">
24.             <conversionPattern value="%level" />
25.         </layout>
26.     </parameter>
27.     <parameter>
28.         <parameterName value=":logger" />
29.         <dbType value="String" />
30.         <size value="255" />
31.         <layout type="log4net.Layout.PatternLayout">
32.             <conversionPattern value="%logger" />
33.         </layout>
34.     </parameter>
35.     <parameter>
36.         <parameterName value=":message" />
37.         <dbType value="String" />
38.         <size value="4000" />
39.         <layout type="log4net.Layout.PatternLayout">
40.             <conversionPattern value="%message" />
41.         </layout>
42.     </parameter>
43. </appender>

```

当然从上面的配置中的SQL语句中可以看得出这个表的参数，日志表的创建语句如下：

[c-sharp]

```
01. create table log (
```

```

02.      Datetime timestamp(3),
03.      Thread varchar2(255),
04.      Log_Level varchar2(255),
05.      Logger varchar2(255),
06.      Message varchar2(4000)
07.  );

```

载:

在本例中周公采用了将日志记录到SQLite这个单机数据库的方式，并且还将记录日志时的文件名和行号，创建SQLite的SQL语句如下：

[c-sharp]

```

01.  CREATE TABLE Log (
02.      LogId          INTEGER PRIMARY KEY,
03.      Date           DATETIME NOT NULL,
04.      Level          VARCHAR(50) NOT NULL,
05.      Logger         VARCHAR(255) NOT NULL,
06.      Source         VARCHAR(255) NOT NULL,
07.      Message        TEXT DEFAULT NULL
08.  );

```

载:

增加的< appender>节点配置如下：

[c-sharp]

```

01.  <appender name="AdoNetAppender_SQLite" type="log4net.Appender.AdoNetAppender">
02.      <bufferSize value="100" />
03.      <connectionType value="System.Data.SQLite.SQLiteConnection, System.Data.SQLite, Version=
04.      <!--SQLite连接字符串-->
05.      <connectionString value="Data Source=c://log4net.db;Version=3.7.2" />
06.      <commandText value="INSERT INTO Log (Date, Level, Logger,Source, Message) VALUES (@Date,
07.      <parameter>
08.          <parameterName value="@Date" />
09.          <dbType value="DateTime" />
10.          <layout type="log4net.Layout.RawTimeStampLayout" />
11.      </parameter>
12.      <parameter>
13.          <parameterName value="@Level" />
14.          <dbType value="String" />
15.          <layout type="log4net.Layout.PatternLayout">
16.              <conversionPattern value="%level" />
17.          </layout>
18.      </parameter>
19.      <parameter>
20.          <parameterName value="@Logger" />
21.          <dbType value="String" />
22.          <layout type="log4net.Layout.PatternLayout">
23.              <conversionPattern value="%logger" />
24.          </layout>
25.      </parameter>

```

```

26.     <parameter>
27.         <parameterName value="@Source" />
28.         <dbType value="String" />
29.         <layout type="log4net.Layout.PatternLayout">
30.             <conversionPattern value="%file:%line" />
31.         </layout>
32.     </parameter>
33.     <parameter>
34.         <parameterName value="@Message" />
35.         <dbType value="String" />
36.         <layout type="log4net.Layout.PatternLayout">
37.             <conversionPattern value="%message" />
38.         </layout>
39.     </parameter>
40. </appender>

```

从上面的配置中可以看出插入数据的SQL语句及参数信息，下面的<parameter>节点就是指定插入的数据，比如我们指定SQL语句中的"@Source"参数来源于log4net中的"%file:%line"参数，也就是这两个参数用":"用连接起来。

控制日志文件大小的问题

对于一个长时间使用并且有大量业务日志的系统来说，如果使用FileAppender将日志一直记录到一个文件中会引起性能低下的问题，我曾见过有个系统的日志文件达到了800多M，最后系统无法及时响应了，在这种情况下可考虑使用RollingFileAppender循环记录日志，一种是指定文件的最大长度，如果超过了就重新生成一个文件，如下面的配置：

[xhtml]

```

01. <appender name="RollingFileAppender" type="log4net.Appender.RollingFileAppender">
02.     <file value="RollingFileAppender_log.txt" />
03.     <appendToFile value="true" />
04.     <rollingStyle value="Size" />
05.     <maxSizeRollBackups value="10" />
06.     <maximumFileSize value="100KB" />
07.     <staticLogFileName value="true" />
08.     <layout type="log4net.Layout.PatternLayout">
09.         <conversionPattern value="%date [%thread] (%file:%line) %-5level %logger [%property{ND
10.     </layout>
11. </appender>

```

在上面的配置中，每个日志文件最大100KB，最大日志文件个数是10生成的日志文件名会是RollingFileAppender_log.txt.1, RollingFileAppender_log.txt.2 ... RollingFileAppender_log.txt.10，如果记录的日志超过10个，会从RollingFileAppender_log.txt.1开始覆盖。

还有一种方式就是按照日期记录日志，它的配置如下：

载：

[xhtml]

```

01. <appender name="RollingLogFileAppender_DateFormat" type="log4net.Appender.RollingFileAppender"
02. <file value="RollingLogFileAppender_DateFormat_log.txt" />
03. <appendToFile value="true" />
04. <rollingStyle value="Date" />
05. <!--<datePattern value="yyyyMMdd-HH:mm" />-->
06. <datePattern value="yyyyMMdd" />
07. <layout type="log4net.Layout.PatternLayout">
08. <conversionPattern value="%date [%thread]
(%file:%line) %-5level %logger [%property{NDC}] - %message%newline" />
09. </layout>
10. </appender>

```

这样一来，每天的日志都写入到一个文件中，当天的日志文件名

为“RollingLogFileAppender_DateFormat_log.txt”，非当天的日志都会带上当天的日期，

如“RollingLogFileAppender_DateFormat_log.txt20101117”表示2010年11月17日的日志，这样就可以很方便地
区分每天的日志了，将来查找起来也相当方便。

| 载:

在配置中启用和关闭日志

在config文件中可以很方便地关闭和启用日志，就是在<root>进行配置，如下就是一个例子：

[xhtml]

```

01. <root>
02. <!--文件形式记录日志-->
03. <appender-ref ref="LogFileAppender" />
04. <!--控制台控制显示日志-->
05. <appender-ref ref="ConsoleAppender" />
06. <!--Windows事件日志-->
07. <!--<appender-ref ref="EventLogAppender" />-->
08. <!--SQLite事件日志-->
09. <appender-ref ref="AdoNetAppender_SQLite" />
10. <!--RollingFileAppender事件日志-->
11. <appender-ref ref="RollingFileAppender" />
12. <!--RollingFileAppender事件日志，每天一个日志-->
13. <appender-ref ref="RollingLogFileAppender_DateFormat" />
14. <!-- 如果不启用相应的日志记录，可以通过这种方式注释掉
15. <appender-ref ref="AdoNetAppender_Access" />
16. -->
17. </root>

```

在上面的例子中可以看出，如果想增加日志输出目的地，增加<appender-ref>节点就是了，注意后面的ref是在
config中配置的appender name，如果想要取消，删除或者注释掉这行就可以了。

Log4Net的使用

| 载:

首先要添加config文件，在类库项目、命令行程序及WinForm中添加的是app.config，在WebForm中添加的是
web.config。

下面是一个在WinForm项目中的使用Log4Net的例子：

[c-sharp]

```
01. using System.Collections.Generic;
02. using System.Text;
03. using log4net;
04. using System.Reflection;
05.
06. //注意下面的语句一定要加上，指定log4net使用.config文件来读取配置信息
07. //如果是WinForm（假定程序为MyDemo.exe，则需要一个MyDemo.exe.config文件）
08. //如果是WebForm，则从web.config中读取相关信息
09. [assembly: log4net.Config.XmlConfigurator(Watch = true)]
10. namespace Log4NetDemo
11. {
12.     class Program
13.     {
14.         static void Main(string[] args)
15.         {
16.             Random random = new Random();
17.             for (int i = 0; i < 1; i++)
18.             {
19.                 //创建日志记录组件实例
20.                 //ILog log = log4net.LogManager.GetLogger(MethodBase.GetCurrentMethod().De
21.
22.                 ILog log=log4net.LogManager.GetLogger(typeof(Program));
23.                 //记录错误日志
24.                 //log.Error("error", new Exception("在这里发生了一个异
25.                 常,Error Number:"+random.Next()));
26.                 //记录严重错误
27.                 //log.Fatal("fatal", new Exception("在发生了一个致命错误,
28.                 Exception Id: "+random.Next()));
29.                 //记录一般信息
30.                 //log.Info("提示：系统正在运行");
31.                 //记录调试信息
32.                 //log.Debug("调试信息： debug");
33.                 //记录警告信息
34.                 //log.Warn("警告： warn");
35.             }
36.             Console.WriteLine("日志记录完毕。");
37.             Console.Read();
38.         }
39.     }
}
```

在WebForm中也可以使用Log4net，下面是一个在ASP.NET中使用Log4Net的例子：

[c-sharp]


```
01. using System;
02. using System.Collections.Generic;
03. using System.Web;
04. using System.Web.UI;
05. using System.Web.UI.WebControls;
06. using log4net;
07. using System.Reflection;
08.
09. //如果是web项目，需要在global.asax的Application_Start方法中配置web.config的引
    用,log4net.Config.XmlConfigurator.Configure();
10. [assembly: log4net.Config.XmlConfigurator(Watch = true)]
11. public partial class _Default : System.Web.UI.Page
12. {
13.     protected void Page_Load(object sender, EventArgs e)
14.     {
15.         if (!Page.IsPostBack)
16.         {
17.             Random random = new Random();
18.             for (int i = 0; i < 1; i++)
19.             {
20.                 //创建日志记录组件实例
21.                 ILog log = log4net.LogManager.GetLogger(MethodBase.GetCurrentMethod().Decl
22.
23.                 //ILog log = log4net.LogManager.GetLogger(typeof(Program));
24.                 //记录错误日志
25.                 //log.Error("error", new Exception("在这里发生了一个异
    常,Error Number:"+random.Next()));
26.                 //记录严重错误
27.                 //log.Fatal("fatal", new Exception("在发生了一个致命错误,
    Exception Id: "+random.Next()));
28.                 //记录一般信息
29.                 //log.Info("提示: 系统正在运行");
30.                 //记录调试信息
31.                 //log.Debug("调试信息: debug");
32.                 //记录警告信息
33.                 log.Warn("警告: warn");
34.                 Response.Write("日志记录完毕。");
35.             }
36.         }
37.     }
38. }
```

可以看出它们的代码基本没有区别。

下面是一个在WinForm下的config文件的完整配置，该配置文件所使用的代码就是上面所使用到的代码，使用LogFileAppender、ConsoleAppender、EventLogAppender、AdoNetAppender_SQLite、RollingFileAppender、RollingLogFileAppender_DateFormat方式记录日志都在本地通过测试。完整的config文件代码如下：

[xhtml]

```
01. <?xml version="1.0" encoding="utf-8" ?>
02. <configuration>
03.   <configSections>
04.     <section name="log4net" type="System.Configuration.IgnoreSectionHandler"/>
05.   </configSections>
06.   <appSettings>
07.     <!-- To enable internal log4net logging specify the
08.           following appSettings key -->
09.     <add key="log4net.Internal.Debug" value="true"/>
10.   </appSettings>
11.   <log4net>
12.     <!--定义输出到文件中-->
13.     <appender name="LogFileAppender" type="log4net.Appender.FileAppender">
14.       <!--每条日志末尾的文字说明-->
15.       <footer value="by 周公" />
16.       <!--定义文件存放位置-->
17.       <file value="LogFileAppender_log.txt" />
18.       <appendToFile value="true" />
19.       <datePattern value="yyyyMMdd-HH:mm:ss" />
20.       <layout type="log4net.Layout.PatternLayout">
21.         <!--每条日志末尾的文字说明-->
22.         <footer value="by 周公" />
23.         <!--输出格式-->
24.         <!--样例: 2010-11-
17 15:50:23,443 [9] (D:/CSProjects/Log4NetDemo/Log4NetDemo/Program.cs:27) FATAL Log4NetDem
25. System.Exception: 在发生了一个致命错误, Exception Id: 548828745-->
26.       <conversionPattern value="记录时间: %date 线程ID:[%thread] 日志级别: 文件: 所在
行%-5level 出错类: %logger property:[%property{NDC}] - 错误描述: %message%newline" />
27.     </layout>
28.   </appender>
29.   <!--定义输出到控制台命令行中-->
30.   <appender name="ConsoleAppender" type="log4net.Appender.ConsoleAppender">
31.     <layout type="log4net.Layout.PatternLayout">
32.       <conversionPattern value="%date [%thread] (%file:%line) %-5level %logger [%property{NDC}]" />
33.     </layout>
34.   </appender>
35.   <!--使用Rolling方式记录日志
36.   每个日志文件最大100KB, 生成的日志文件名会是log.txt.1,log.txt.2 ...log.txt.10
37.   如果记录的日志超过10个, 会从log.txt.1开始覆盖
38.   -->
39.   <appender name="RollingFileAppender" type="log4net.Appender.RollingFileAppender">
40.     <file value="RollingFileAppender_log.txt" />
41.     <appendToFile value="true" />
42.     <rollingStyle value="Size" />
43.     <maxSizeRollBackups value="10" />
44.     <maximumFileSize value="100KB" />
45.     <staticLogFileName value="true" />
46.     <layout type="log4net.Layout.PatternLayout">
47.       <conversionPattern value="%date [%thread] (%file:%line) %-5level %logger [%property{NDC}]" />
48.     </layout>
```

```
49. </appender>
50.     <!--使用Rolling方式记录日志
51.     按照日来记录日志
52.     -->
53. <appender name="RollingLogFileAppender_DateFormat" type="log4net.Appender.RollingFileAppen
54.     <file value="RollingLogFileAppender_DateFormat_log.txt" />
55.     <appendToFile value="true" />
56.     <rollingStyle value="Date" />
57.     <!--<datePattern value="yyyyMMdd-HH:mm" />-->
58.     <datePattern value="yyyyMMdd" />
59.     <layout type="log4net.Layout.PatternLayout">
60.         <conversionPattern value="%date [%thread]
(%file:%line) %-5level %logger [%property{NDC}] - %message%newline" />
61.     </layout>
62. </appender>
63.
64.     <!--记录到SQLite这样的单机数据库中去
65.     创SQLite表的SQL语句:
66.     CREATE TABLE Log (
67.         LogId          INTEGER PRIMARY KEY,
68.         Date            DATETIME NOT NULL,
69.         Level           VARCHAR(50) NOT NULL,
70.         Logger          VARCHAR(255) NOT NULL,
71.         Source          VARCHAR(255) NOT NULL,
72.         Message         TEXT DEFAULT NULL
73.     );
74.     -->
75. <appender name="AdoNetAppender_SQLite" type="log4net.Appender.AdoNetAppender">
76.     <bufferSize value="100" />
77.     <connectionType value="System.Data.SQLite.SQLiteConnection, System.Data.SQLite, Version=
78.     <!--SQLite连接字符串-->
79.     <connectionString value="Data Source=c://log4net.db;Version=3;" />
80.     <commandText value="INSERT INTO Log (Date, Level, Logger,Source, Message) VALUES (@Date,
81.     <parameter>
82.         <parameterName value="@Date" />
83.         <dbType value="DateTime" />
84.         <layout type="log4net.Layout.RawTimeStampLayout" />
85.     </parameter>
86.     <parameter>
87.         <parameterName value="@Level" />
88.         <dbType value="String" />
89.         <layout type="log4net.Layout.PatternLayout">
90.             <conversionPattern value="%level" />
91.         </layout>
92.     </parameter>
93.     <parameter>
94.         <parameterName value="@Logger" />
95.         <dbType value="String" />
96.         <layout type="log4net.Layout.PatternLayout">
97.             <conversionPattern value="%logger" />
98.         </layout>
```

```
99.     </parameter>
100.    <parameter>
101.        <parameterName value="@Source" />
102.        <dbType value="String" />
103.        <layout type="log4net.Layout.PatternLayout">
104.            <conversionPattern value="%file:%line" />
105.        </layout>
106.    </parameter>
107.    <parameter>
108.        <parameterName value="@Message" />
109.        <dbType value="String" />
110.        <layout type="log4net.Layout.PatternLayout">
111.            <conversionPattern value="%message" />
112.        </layout>
113.    </parameter>
114. </appender>
115. <!-- 定义输出到SQL Server数据库中-->
116. <!--
117. 在SQL Server中创建表的SQL语句
118. CREATE TABLE [dbo].[Log] (
119. [Id] [int] IDENTITY (1, 1) NOT NULL,
120. [Date] [datetime] NOT NULL,
121. [Thread] [varchar] (255) NOT NULL,
122. [Level] [varchar] (50) NOT NULL,
123. [Logger] [varchar] (255) NOT NULL,
124. [Message] [varchar] (4000) NOT NULL,
125. [Exception] [varchar] (2000) NULL
126. );
127. -->
128. <appender name="AdoNetAppender_SQLServer" type="log4net.Appender.AdoNetAppender">
129.     <bufferSize value="100" />
130.     <connectionType value="System.Data.SqlClient.SqlConnection, System.Data" />
131.     <connectionString value="data source=[database server];initial catalog=
[database name];integrated security=false;persist security info=True;User ID=
[user];Password=[password]" />
132.     <commandText value="INSERT INTO Log ([Date],[Thread],[Level],[Logger],[Message],
[Exception]) VALUES (@log_date, @thread, @log_level, @logger, @message, @exception)" />
133.     <parameter>
134.         <parameterName value="@log_date" />
135.         <dbType value="DateTime" />
136.         <layout type="log4net.Layout.RawTimeStampLayout" />
137.     </parameter>
138.     <parameter>
139.         <parameterName value="@thread" />
140.         <dbType value="String" />
141.         <size value="255" />
142.         <layout type="log4net.Layout.PatternLayout">
143.             <conversionPattern value="%thread" />
144.         </layout>
145.     </parameter>
146.     <parameter>
```

```

147.     <parameterName value="@log_level" />
148.     <dbType value="String" />
149.     <size value="50" />
150.     <layout type="log4net.Layout.PatternLayout">
151.         <conversionPattern value="%level" />
152.     </layout>
153. </parameter>
154. <parameter>
155.     <parameterName value="@logger" />
156.     <dbType value="String" />
157.     <size value="255" />
158.     <layout type="log4net.Layout.PatternLayout">
159.         <conversionPattern value="%logger" />
160.     </layout>
161. </parameter>
162. <parameter>
163.     <parameterName value="@message" />
164.     <dbType value="String" />
165.     <size value="4000" />
166.     <layout type="log4net.Layout.PatternLayout">
167.         <conversionPattern value="%message" />
168.     </layout>
169. </parameter>
170. <parameter>
171.     <parameterName value="@exception" />
172.     <dbType value="String" />
173.     <size value="2000" />
174.     <layout type="log4net.Layout.ExceptionLayout" />
175. </parameter>
176. </appender>
177. <!--定义输出到Oracle9i中-->
178. <!--
179. 在Oracle9i中创建表的SQL语句
180. create table log (
181. Datetime timestamp(3),
182. Thread varchar2(255),
183. Log_Level varchar2(255),
184. Logger varchar2(255),
185. Message varchar2(4000)
186. );
187. -->
188. <appender name="AdoNetAppender_Oracle" type="log4net.Appender.AdoNetAppender">
189.     <connectionType value="System.Data.OracleClient.OracleConnection, System.Data.OracleClient" />
190.     <connectionString value="data source=[mydatabase];User ID=[user];Password=[password]" />
191.     <commandText value="INSERT INTO Log (Datetime,Thread,Log_Level,Logger,Message) VALUES (:log_date, :thread, :log_level, :logger, :message)" />
192.     <bufferSize value="128" />
193.     <parameter>
194.         <parameterName value=":log_date" />
195.         <dbType value="DateTime" />
196.         <layout type="log4net.Layout.RawTimeStampLayout" />

```

```
197.     </parameter>
198.     <parameter>
199.         <parameterName value=":thread" />
200.         <dbType value="String" />
201.         <size value="255" />
202.         <layout type="log4net.Layout.PatternLayout">
203.             <conversionPattern value="%thread" />
204.         </layout>
205.     </parameter>
206.     <parameter>
207.         <parameterName value=":log_level" />
208.         <dbType value="String" />
209.         <size value="50" />
210.         <layout type="log4net.Layout.PatternLayout">
211.             <conversionPattern value="%level" />
212.         </layout>
213.     </parameter>
214.     <parameter>
215.         <parameterName value=":logger" />
216.         <dbType value="String" />
217.         <size value="255" />
218.         <layout type="log4net.Layout.PatternLayout">
219.             <conversionPattern value="%logger" />
220.         </layout>
221.     </parameter>
222.     <parameter>
223.         <parameterName value=":message" />
224.         <dbType value="String" />
225.         <size value="4000" />
226.         <layout type="log4net.Layout.PatternLayout">
227.             <conversionPattern value="%message" />
228.         </layout>
229.     </parameter>
230. </appender>
231. <!--定义输出到IBM DB2中-->
232. <!--
233. 在DB2中创建表的SQL语句
234. CREATE TABLE "myschema.LOG" (
235.     "ID" INTEGER NOT NULL GENERATED ALWAYS AS IDENTITY (
236.         START WITH +1
237.         INCREMENT BY +1
238.         MINVALUE +1
239.         MAXVALUE +2147483647
240.         NO CYCLE
241.         NO CACHE
242.         NO ORDER
243.     ),
244.     "DATE" TIMESTAMP NOT NULL,
245.     "THREAD" VARCHAR(255) NOT NULL,
246.     "LEVEL" VARCHAR(500) NOT NULL,
247.     "LOGGER" VARCHAR(255) NOT NULL,
```

```
248. "MESSAGE" VARCHAR(4000) NOT NULL,
249. "EXCEPTION" VARCHAR(2000)
250. )
251. IN "LRGTABLES";
252. -->
253. <appender name="AdoNetAppender_DB2" type="log4net.Appender.AdoNetAppender">
254.   <bufferSize value="100" />
255.   <connectionType value="IBM.Data.DB2.DB2Connection,IBM.Data.DB2" />
256.   <connectionString value="server=192.168.0.0;database=dbuser;user Id=username;password=" />
257.   <commandText value="INSERT INTO myschema.Log (Date,Thread,Level,Logger,Message,Exception)" />
258.   <parameter>
259.     <parameterName value="@log_date" />
260.     <dbType value="DateTime" />
261.     <layout type="log4net.Layout.RawTimeStampLayout" />
262.   </parameter>
263.   <parameter>
264.     <parameterName value="@thread" />
265.     <dbType value="String" />
266.     <size value="255" />
267.     <layout type="log4net.Layout.PatternLayout">
268.       <conversionPattern value="%thread" />
269.     </layout>
270.   </parameter>
271.   <parameter>
272.     <parameterName value="@log_level" />
273.     <dbType value="String" />
274.     <size value="500" />
275.     <layout type="log4net.Layout.PatternLayout">
276.       <conversionPattern value="%level" />
277.     </layout>
278.   </parameter>
279.   <parameter>
280.     <parameterName value="@logger" />
281.     <dbType value="String" />
282.     <size value="255" />
283.     <layout type="log4net.Layout.PatternLayout">
284.       <conversionPattern value="%logger" />
285.     </layout>
286.   </parameter>
287.   <parameter>
288.     <parameterName value="@message" />
289.     <dbType value="String" />
290.     <size value="4000" />
291.     <layout type="log4net.Layout.PatternLayout">
292.       <conversionPattern value="%message" />
293.     </layout>
294.   </parameter>
295.   <parameter>
296.     <parameterName value="@exception" />
297.     <dbType value="String" />
298.     <size value="2000" />
```

```
299.     <layout type="log4net.Layout.ExceptionLayout" />
300.   </parameter>
301. </appender>
302. <!--定义输出到windows事件中-->
303. <appender name="EventLogAppender" type="log4net.Appender.EventLogAppender">
304.   <layout type="log4net.Layout.PatternLayout">
305.     <conversionPattern value="%date [%thread] %-5level %logger [%property{NDC}] - %message" />
306.   </layout>
307. </appender>
308. <!--定义输出到数据库中，这里举例输出到Access数据库中，数据库为C盘的log4net.mdb
309. 创建Access表的SQL语句：
310. -->
311. <appender name="AdoNetAppender_Access" type="log4net.Appender.AdoNetAppender">
312.   <connectionString value="Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:/log4net.mdb" />
313.   <commandText value="INSERT INTO LogDetails ([LogDate],[Thread],[Level],[Logger],[Message]) VALUES (@logDate, @thread, @logLevel, @logger,@message)" />
314.   <!--定义各个参数-->
315.   <parameter>
316.     <parameterName value="@logDate" />
317.     <dbType value="String" />
318.     <size value="240" />
319.     <layout type="log4net.Layout.PatternLayout">
320.       <conversionPattern value="%date" />
321.     </layout>
322.   </parameter>
323.   <parameter>
324.     <parameterName value="@thread" />
325.     <dbType value="String" />
326.     <size value="240" />
327.     <layout type="log4net.Layout.PatternLayout">
328.       <conversionPattern value="%thread" />
329.     </layout>
330.   </parameter>
331.   <parameter>
332.     <parameterName value="@logLevel" />
333.     <dbType value="String" />
334.     <size value="240" />
335.     <layout type="log4net.Layout.PatternLayout">
336.       <conversionPattern value="%level" />
337.     </layout>
338.   </parameter>
339.   <parameter>
340.     <parameterName value="@logger" />
341.     <dbType value="String" />
342.     <size value="240" />
343.     <layout type="log4net.Layout.PatternLayout">
344.       <conversionPattern value="%logger" />
345.     </layout>
346.   </parameter>
347.   <parameter>
348.     <parameterName value="@message" />
```



```
349.         <dbType value="String" />
350.         <size value="240" />
351.         <layout type="log4net.Layout.PatternLayout">
352.             <conversionPattern value="%message" />
353.         </layout>
354.     </parameter>
355. </appender>
356. <!-- 定义日志的输出媒介，下面定义日志以四种方式输出。也可以下面的按照一种类型或其他类型输出。 -
->
357. <root>
358.     <!-- 文件形式记录日志-->
359.     <appender-ref ref="LogFileAppender" />
360.     <!-- 控制台控制显示日志-->
361.     <appender-ref ref="ConsoleAppender" />
362.     <!-- Windows事件日志-->
363.     <!-- <appender-ref ref="EventLogAppender" />-->
364.     <!-- SQLite日志-->
365.     <!-- <appender-ref ref="AdoNetAppender_SQLite" />-->
366.     <!-- RollingFileAppender事件日志-->
367.     <appender-ref ref="RollingFileAppender" />
368.     <!-- RollingFileAppender事件日志，每天一个日志-->
369.     <appender-ref ref="RollingLogFileAppender_DateFormat" />
370.     <!-- 如果不启用相应的日志记录，可以通过这种方式注释掉
371.     <appender-ref ref="AdoNetAppender_Access" />
372.     -->
373. </root>
374.
375. </log4net>
376. <!-- <system.diagnostics>
377.     <trace autoflush="true">
378.         <listeners>
379.             <add
380.                 name="textWriterTraceListener"
381.                 type="System.Diagnostics.TextWriterTraceListener"
382.                 initializeData="D:/CSProjects/Log4NetDemo/Log4NetDemo/bin/log4net.txt" />
383.             </listeners>
384.         </trace>
385.     </system.diagnostics>-->
386.
387. </configuration>
```

总结：

本篇主要是补充在上一篇关于Log4Net中未尽之处，并集中详尽回答了一些朋友在该篇博文下的提问，如有未尽之处，请在本篇下留言。如有初学者碰巧路过不知config文件为何物，请看《asp.net夜话之十一：web.config详解》，网址是：<http://blog.csdn.net/zhoufoxcn/archive/2008/11/10/3265141.aspx>。