

FaktNet AS

Log4Net Dashboard

Setup and Configuration

support@l4ndash.com

© FaktNet AS, all rights reserved



Version 2.4

TABLE OF CONTENTS

Introduction.....	3
Log4Net vs. the Log4Net Dashboard.....	3
Required knowledge.....	3
Concepts.....	4
Installing Log4Net Dashboard	7
System requirements	7
Installing Log4Net Dashboard	7
Configuration of L4NDash	8
Customizing the SummaryTable in the Dashboard page	9
Defining data sources.....	11
The SqlServer provider	14
Database access (security).....	15
Increasing performance.....	15
Log4NET appender definition.....	16
The Oracle provider.....	18
Log4NET appender definition.....	19
The Xml TextFile provider.....	21
Log file access / security	22
Performance / Memory consumption	22
Log4NET appender definition.....	23
The Windows Events Log provider	24
Log4NET appender definition.....	24
Using predefined search filters.....	25
Adding custom columns	26
Custom columns in xml log files	27
Expanding the width of the web page.....	28
Visualizers.....	29
DateTime Visualizer.....	31
String Visualizer	32
XML Visualizer	33
Other parameters in the <appSettings> section	34

INTRODUCTION

The purpose of this document is to:

- Describe the steps needed to install Log4Net Dashboard
- Log4Net dashboard can be configured in several ways. An important step in installing Log4Net Dashboard is therefore to do the necessary configuration. Most of this document deals with the options and possibilities to configure and tailor Log4Net Dashboard.

Before you read this document and to familiarize yourself with the application, you may want to check out the online demonstration site, available on <http://demo.l4ndash.com>

LOG4NET VS. THE LOG4NET DASHBOARD

Log4Net is an open source apache project (originally ported from log4j by Neoworks ltd.), see <http://logging.apache.org/log4net/history.html>

Log4net Dashboard is a commercial application (however with a free Developer version) created by FaktNet AS, a Norwegian based Consulting Company, please see <http://www.L4NDash.com> or <http://www.FaktNet.com>

REQUIRED KNOWLEDGE

- The typical Log4Net Dashboard users already have an application using log4Net. To use Log4Net Dashboard you need to have a basic understanding of Log4Net.
- You need to be able to configure one of the three Log4Net appenders supported by Log4Net Dashboard.
 - Fileappener (or Rolling file appender) that uses the XmlLayout.
 - AdoNetAppender that writes log events to a database
 - EventLogAppender that writes log events to the Windows Event Log
- You need to know how to configure a virtual directory in IIS (Internet Information Server). To do this you may need to have administrator privileges.

CONCEPTS

Throughout this document we refer to different areas of the application. Below is a picture illustrating the different areas (panels) of the dashboard page.

Dashboard
New log events
Clean up log
Settings
Help
About
Homepage

Summary
Summary Table

	TOTAL	FATAL	ERROR	WARN	INFO	DEBUG
Last Hour	10000		88	55	4589	5268
Previous Hour	0					
22 Hours before	19998	19	222	26	4821	14910
Yesterday	9999	4	125	16	2700	7154
Week before	19999	4	1139	16	8090	10750
Month before	9998		86	20	1031	8861
Older	289016	213	56771	840	231192	

Log Rows
Log Rows Panel

ID	DATE	LEVEL	LOGGER	MESSAGE
411257	05/28/07 06:53:40	FATAL	Rss2Vxml.Rss2Vxml	An unknown error / exception has occurred
411227	05/28/07 06:53:40	FATAL	Rss2Vxml.Rss2Vxml	An unknown error / exception has occurred
411221	05/28/07 06:53:40	FATAL	Rss2Vxml.Rss2Vxml	Missing translation file - file not found
411055	05/28/07 06:53:40	FATAL	Rss2Vxml.Rss2Vxml	Missing translation file - file not found

Log Row Detail
Log Row Detail Panel

ID:	411221	Date:	05/28/2007 06:53:40	Level:	FATAL
Thread:	3168	Logger:	Rss2Vxml.Rss2Vxml		
Message:	Missing translation file - file not found				
Exception:					

Filter
Filter Panel

Predefined filters: ---

Logger: Starts with

Thread: Starts with

Message: Starts with

Exception: Starts with

Levels to include: FATAL ERROR WARN INFO DEBUG

Save filter as:

The Logger Summary page (new in version 2.4) has a *Logger Summary Panel* as displayed below:

Dashboard Logger summary New log events Clean up log Settings Help About Homepage						
Summary						
	TOTAL	FATAL	ERROR	WARN	INFO	DEBUG
FNFrww.FNUtil	416					416
L4NDash.CleanUp	328					328
Database	1495		12			1483
L4NDashUtil	57		57			
PageDashboard.Dashboard	525					525
PageHelp.Help	94					94
PageSettings.Settings	98					98
SessionData	153					153
UcHelp	125		6			119
L4NWebDashboard.Database	86	16	70			
L4NDashUtil	38		38			
Rss2Vxml.Rss2Vxml	955	14	21	20	474	426
RssPage	194		2			192
TagService	1	1				
Xlat	29		6			23
WebDataFangst.Database	764	20				744
HttpPage	676		54	40		582
Tidevann	378					378
WebDataFangst	1836		4		1798	34

Below is a table defining the commonly used concepts in this document.

Concept	Explanation
<i>L4NDash</i>	The nickname for the Log4Net dashboard application.
<i>Summary Table</i>	The first panel in the dashboard page showing the total number of log events. The Summary Table is categorized by log level and time period.
<i>Log Row Panel</i>	The second panel showing a list of log rows. The log rows displayed are based on a selection made in the summary table. The user can click anywhere in the summary table and display the log rows based on the selection.
<i>Log Row Detail Panel</i>	Details about one log event. The Detail Panel is displayed when the user is selecting one log row in the Log Rows Panel.
<i>Filter Panel</i>	The panel where the user can specify a filter. The filter will restrict the log events that are displayed. A filter will affect all panels displaying log rows (i.e. Summary table, Log Rows panel and Log Row Detail panel).

<i>Logger Summary Panel</i>	The first panel in the logger summary page, showing the total number of log events for each logger. The table is categorized by log level.
<i>L4NDashWebRoot</i>	The folder / directory where Log4net dashboard is installed, for example c:\inetpub\wwwroot\L4NDash
<i>L4NDashUserAccount</i>	The user account that runs the web site. Which account this is depends on how the web site is configured in Internet Information Server.
<i>Data Source</i>	A Data Source is the storage from where L4NDash reads log events displayed in the application.
<i>Data Source Provider</i>	A type (class) that handles the process of retrieving log events from a specific <i>Data Source</i> .
<i>Standard columns</i>	<p>Log4Net defines a set of standard columns (or items) that are logged at all times. The standard columns are:</p> <ul style="list-style-type: none"> • Logger – Usually a class name in your application. • Level – The log level. The standard levels are Fatal, Error, Warning, Info and debug. • Thread – Numeric thread id from .Net framework. • Message – The message your application is logging. • Exception – The exception for the log event. Not all log events have an exception logged.
<i>Custom columns</i>	Log4Net offers rich possibilities to log information in addition to the standard columns. These pieces of information are logged into what we have chosen to name Custom columns.

Throughout this document, the concepts mentioned in the table above are written in *italic*.

INSTALLING LOG4NET DASHBOARD

SYSTEM REQUIREMENTS

System requirements:

- Internet Information Server
- Microsoft SQL Server (only if you are using SQL Server as logging storage).
- Oracle DBMS (only if you are using Oracle as logging storage).
- 3MB free disk space
- .Net Framework version 2.0

INSTALLING LOG4NET DASHBOARD

To install log4net Dashboard:

1. Download the zip file that contain Log4Net Dashboard from www.l4ndash.com
2. Unzip the file to a suitable folder, for example c:\inetpub\wwwroot\l4ndash
3. Create a virtual directory in Internet information server pointing the folder chosen in the previous step.

If you are upgrading from a previous version, please install a new directory instead of upgrading an old version. After the installation you need to move your settings from the old web.config to the new web.config file.

After you have installed Log4Net Dashboard you need to configure your data sources. The configuration of data sources is explained later in this document.

A common issue with Log4Net Dashboard is the access privileges given to the *L4NDashUserAccount*, we emphasize the importance of granting the necessary access privileges to this user account.

CONFIGURATION OF L4NDASH

The configuration information for *L4NDash* is kept in the `web.config` file located in the *L4NDashWebRoot*. This configuration information controls the behavior of the application, and it gives you a lot of possibilities to change and customize *L4NDash*.

- Define new data sources
- Change the periods used in the summary table
- Change the appearance of column values that are retrieved from the log
- Define the custom columns you want to display

CUSTOMIZING THE SUMMARYTABLE IN THE DASHBOARD PAGE

The *Summary Table* is defined under the <l4ndash> section. By altering the *Summary Table* configuration you can define the number of lines and the time span for each row in the *Summary Table*.

Each row in the *Summary Table* is defined through a <summarytableperiod Name="periodname"> section, as shown in the example below:

```
<summarytableperiod name="Last Hour">
  <comment value="Shows events for the past hour"/>
  <timediff value="1" valuetype="hour" start="now"/>
</summarytableperiod>
```

Parameter	Explanation
Name	The name of the period. This is the text displayed as a row header in the <i>Summary Table</i> .
Comment	A longer text that will be displayed when the user are hovering over the row header in the <i>Summary Table</i> (tool tip).
Timediff	The timediff section defines how the time period for this row is calculated. The timediff section must contain the three parameters described below.
Timediff valuetype	Can be hour, day, month and year.
Timediff value	A number defining the timespan for the summary table row
Timediff start	When the period will start. Valid values are now, perviousperiodstart and previousperiodend.

Below is an example on a *Summary Table* with three periods:

```
<summarytableperiod name="Today">
  <comment value="Shows events for today"/>
  <timediff value="24" valuetype="hour" start="now"/>
</summarytableperiod>

<summarytableperiod name="Yesterday">
  <comment value="Shows events for the previous hour"/>
  <timediff value="24" valuetype="hour" start="previousperiodend"/>
</summarytableperiod>

<summarytableperiod name="Older">
  <comment value="Older"/>
  <timediff value="99" valuetype="year" start="previousperiodend"/>
</summarytableperiod>
```

Using the above configuration the *Summary Table* will be displayed as:

Summary						
	TOTAL	FATAL	ERROR	WARN	INFO	DEBUG
Today	0					
Yesterday	0					
Older	369066	240	58579	978	257796	51473

If the current date and time is 2 May 2007 16:00, the table below exemplifies the period start and the period end.

Period Name	Period start	Period end
Today	2 May 2007 16:00:00	2 May 2007 15:00:00
Yesterday	2 May 2007 15:00:00	1 May 2007 15:00:00
Older	1 May 2007 15:00:00	1 May 1908: 15:00:00

DEFINING DATA SOURCES

For most users it is sufficient to define one *Data Source*, however *L4NDash* gives you the possibility to define several *Data Sources*. The first *Data Source* will be the default. In the “settings” page the user can choose to use one of the other defined *Data Sources*.

The *Data Sources* are defined in the `<l4ndash>` section. For each data source you must define a `<datasource>` section. Currently, *L4NDash* have *Data Source Providers* to read log events from:

- Microsoft SQL Server
- Oracle SQL Server
- Windows Event Log
- Text files (containing log events formatted with the `log4net.Layout.XmlLayout`).

You can define several *Data Sources* for each *Data Source Provider*, just make sure you give them unique names.

The different *Data Source Providers* are implemented through a public provider interface. This gives you or others the possibility to program new providers for any kind of logs you need to view through *L4NDash*. The *L4NDash* SDK will provide you with information and examples on how to write your own *Data Source Provider*.

One definition of a *Data Source* can contain several sub sections:

- Parameters for defining the *Data Source Provider*
- Parameters to configure the *Data Source Provider*
- Custom columns
- Visualizers

This is what a general data source section can look like:

```
<datasource name="DataSoucUniqueName" Description="Data source description">
  <provider type="typename" assembly="assembly ref"/>
  <predefinedsearchfilter value="relative path to the
                           file storing predefined filters"/>
  <providersettings>
    <!-- provider dependent settings parameters -->
  </providersettings>
  <customcolumns>
    <customcolumn name="Columnname">
      <label value="label value"/>
      <rowstabledisplay value="yes|no"/>
      <rowsdetaildisplay value="yes|no"/>
      <filter value="yes|no"/>
    </customcolumn>
  </customcolumns>
</datasource>
```

```

        </customcolumn>

        <!-- more custom columns -->

    </customcolumns>

    <columnvisualizers>
        <visualizer Columnname="Date">
            <provider type="typename" assembly="assembly refernec"/>
            <providersettings>
                <!-- provider dependent settings parameters -->
            </providersettings>

        </visualizer>
        <!-- more visualizers -->
    </columnvisualizers>

</datasource>

```

If you need to define several *Data Sources*, you create several <datasource> sections.

A configuration section for one data source must include

```

<datasource name="Unique name" Description="Description of the datasource">
    <provider type="providerclass (type)" assembly="assembly reference"/>
    <providersettings>
        <!-- provider dependent parameteres -->
    </providersettings>

</datasource>

```

Parameter	Explanation
Name	A unique name identifying the <i>Data Source</i> .
Description	A text describing the <i>Data Source</i> . This text will be displayed in the settings page where the user can choose among the defined <i>Data Source</i> .
Type (in provider)	The class name of the class implementing the provider. This must include the full namespace.
Assembly	The assembly where the "Type" can be found. The format of the assembly reference string is: AssemblyName [,Version=9.9.9.9]

Setup and Configuration

support@l4ndash.com



	<p>[,Culture=Culture][,PublicKeyToken=Token]</p> <p>A typical example of a assembly reference is:</p> <p>L4NDashXmlTextFileProvider, Version=2.4, Culture=neutral</p>
ProviderSettings	<p>This section is dependent on the <i>Data Source Provider</i> and will be explained later in the sections covering the different <i>Data Sources Providers</i>.</p>

THE SQLSERVER PROVIDER

The SqlServer provider reads log rows from a SqlServer database. An example of the default definition of the SqlServer provider is listed below

```
<datasource name="LocalSql" Description="Log4Net log rows stored in SQLServer">
  <provider type="L4NDashSqlServerProvider.SqlServerProvider"
    assembly="L4NDashSqlServerProvider, Version=2.4,Culture=neutral" />
  <providersettings>
    <connectionstring value=
      "Data Source=(local); initial Catalog=Log4Net;Integrated Security=True;" />

    <SqlCommandTimeout value="60" />
    <MaxNoOfRowsToRetrieve value="1000" />

  </providersettings>

  <NameMapping TableName="Log4Net "
    IdColumn="Id "
    DateColumn="Date "
    LevelColumn="Level "
    ThreadColumn="Thread "
    LoggerColumn="Logger "
    MessageColumn="Message "
    ExceptionColumn="Exception" />

</datasource>
```

The Log table (the table named in the table name parameter) must contain the *Standard Columns* (i.e. Id, Date, Logger, Level, Thread, Message and Exception). In addition you can of course define *Custom Columns*.

Config Settings	Explanation
ConnectionString	The Sql server connection string.
TableName	Name of Log4Net log table in the database. Obsolute, use the NameMapping element.

SqlCommandTimeout	Number of seconds to wait for a sql command to finish. If a sql command executes for more than the specified time, the command will be terminated.
MaxNoOfRowsToRetrieve	The maximum number of rows to retrieve into the <i>Log Row Panel</i> . This number will not affect the number of rows used to produce the <i>Summary Table</i> .
NameMapping	Optional defines mapping for the names in the database, ie. the table name and the column names. The SqlProvider use this default names: TableName: log4Net Columns: Id, Date, Level, Logger, Message, Thread, Exception

DATABASE ACCESS (SECURITY)

If you use a Sql Server connection string that uses integrated security, you must make sure that the *L4NDashUserAccount* have the necessary access rights in the database. This also depends on whenever you are using impersonating or not, and the version of Internet Information server you are running. Please check your IIS documentation and grant the necessary rights to the log table using the statement below.

If the user account is ASPNET and the log table is named Log4Net, the statement would be:

```
-- Add login for user ASPNET
CREATE LOGIN [machinename\ASPNET] FROM WINDOWS WITH DEFAULT_DATABASE=[master],
DEFAULT_LANGUAGE=[us_english]

-- Add user to Log4Net database
CREATE USER [ASPNET] FOR LOGIN [machinename\ASPNET]

-- Grant access to the log table
Grant Select on Log4Net to ASPNET

-- The user will also issue delete staements from the Clean up log page
Grant Delete on Log4Net to ASPNET
```

You need to adjust these sql statements to reflect your specific environment.

INCREASING PERFORMANCE

The performance of the Sql Server provider can be increased by creating indexes on the log table. We recommend that you create two indexes as defined below.

```
CREATE UNIQUE INDEX idxDate ON Log4Net([Date] DESC , [Id] DESC )  
CREATE INDEX idxLevel ON Log4Net([Date] DESC , [Level])
```

In addition you can limit the maximum number of rows *L4NDash* retrieved into the Log rows panel. This can also improve performance. It can be done by changing the value of the MaxNoOfRowsToRetrieve parameter. (Most users do not want to page through a large amount of rows).

LOG4NET APPENDER DEFINITION

Below is an example of an AdoNetAppender compatible with *L4NDash*.

```
<appender name="AdoNetAppender" type="log4net.Appender.AdoNetAppender">  
  <bufferSize value="1"/>  
  <connectionType value="System.Data.SqlClient.SqlConnection, System.Data,  
Version=1.0.5000.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"/>  
  <connectionString value="Data Source=(local); initial Catalog=Log4Net;Integrated  
Security=True;"/>  
  <commandText value="INSERT INTO Log4Net  
([Date],[Thread],[Level],[Logger],[Message],[Exception]) VALUES (@log_date, @thread,  
@log_level, @logger, @message, @exception)"/>  
  <parameter>  
    <parameterName value="@log_date"/>  
    <dbType value="DateTime"/>  
    <layout type="log4net.Layout.RawTimeStampLayout"/>  
  </parameter>  
  <parameter>  
    <parameterName value="@thread"/>  
    <dbType value="String"/>  
    <size value="255"/>  
    <layout type="log4net.Layout.PatternLayout">  
      <conversionPattern value="%thread"/>  
    </layout>  
  </parameter>  
  <parameter>  
    <parameterName value="@log_level"/>  
    <dbType value="String"/>  
    <size value="50"/>  
    <layout type="log4net.Layout.PatternLayout">  
      <conversionPattern value="%level"/>  
    </layout>  
  </parameter>  
  <parameter>  
    <parameterName value="@logger"/>  
    <dbType value="String"/>  
    <size value="255"/>  
    <layout type="log4net.Layout.PatternLayout">  
      <conversionPattern value="%logger"/>  
    </layout>  
  </parameter>  
  <parameter>  
    <parameterName value="@message"/>
```

```
<dbType value="String"/>
<size value="4000"/>
<layout type="log4net.Layout.PatternLayout">
  <conversionPattern value="%message"/>
</layout>
</parameter>
<parameter>
  <parameterName value="@exception"/>
  <dbType value="String"/>
  <size value="4000"/>
  <layout type="log4net.Layout.ExceptionLayout"/>
</parameter>
</appender>
```

THE ORACLE PROVIDER

The Oracle provider reads log rows from an Oracle database. An example of the default definition of the Oracle provider is listed below

```
<datasource name="LocalOracle" Description="Log4Net log rows stored in Oracle ">
  <provider type="L4NDashOracleProvider.SqlOracleProvider"
    assembly="L4NDashOracleProvider, Version=2.4, Culture=neutral"/>

  <providersettings>
    <connectionString value="Data Source=XE;User Id=system;Password=manager"/>
    <tablename value="Log4Net"/>
    <SqlCommandTimeout value="60"/>
    <MaxNoOfRowsToRetrieve value="1000"/>
  </providersettings>

  <NameMapping TableName="Log4Net "
    IdColumn="Id "
    DateColumn="DateTime "
    LevelColumn="Log_Level "
    ThreadColumn="Thread "
    LoggerColumn="Logger "
    MessageColumn="Message "
    ExceptionColumn="Exception" />
</datasource>
```

The Log table (the table named in the table name parameter) must contain the *Standard Columns* (i.e. Id, DateTime, Log_Level, Logger, Thread, Message and Exception). Please note that the column name used in Oracle is different from SqlServer (due to different reserved words in the two database system).

In addition to the *Standard Columns* you can of course define *Custom Columns*.

Settings	Explanation
ConnectionString	The Oracle connection string.
SqlCommandTimeout	Number of seconds to wait for a sql command to finish. If a sql command executes for more than the specified time, the command will be terminated.

MaxNoOfRowsToRetrieve	The maximum number of rows to retrieve into the <i>Log Row Panel</i> . This number will not affect the number of rows used to produce the <i>Summary Table</i> .
NameMapping	<p>Optional defines mapping for the names in the database, ie. the table name and the column names.</p> <p>The OracleProvider use this default names:</p> <p>TableName: log4Net</p> <p>Columns: Id, DateTime, Log_Level, Logger, Message, Thread, Exception</p>

LOG4NET APPENDER DEFINITION

Below is an example of an AdoNetAppender compatible with *L4NDash*.

```
<appender name="AdoNetAppender_Oracle" type="log4net.Appender.AdoNetAppender">
  <connectionType value="Oracle.DataAccess.Client.OracleConnection,
Oracle.DataAccess, Version=2.102.2.20, Culture=neutral, PublicKeyToken=89b483f429c47342"
  />
  <connectionString value="Data Source=XE;User Id=system;Password=manager"/>
  <commandText value="INSERT INTO Log4Net (Datetime, Thread, Log_Level, Logger,
Message, Exception) VALUES (:log_date, :thread, :log_level, :logger, :message,
:exception)" />
  <bufferSize value="1" />
  <parameter>
    <parameterName value=":log_date" />
    <dbType value="DateTime" />
    <layout type="log4net.Layout.RawTimeStampLayout" />
  </parameter>
  <parameter>
    <parameterName value=":thread" />
    <dbType value="String" />
    <size value="255" />
    <layout type="log4net.Layout.PatternLayout">
      <conversionPattern value="%thread" />
    </layout>
  </parameter>
  <parameter>
    <parameterName value=":log_level" />
    <dbType value="String" />
    <size value="50" />
    <layout type="log4net.Layout.PatternLayout">
      <conversionPattern value="%level" />
    </layout>
  </parameter>
  <parameter>
    <parameterName value=":logger" />
    <dbType value="String" />
    <size value="255" />
    <layout type="log4net.Layout.PatternLayout">
      <conversionPattern value="%logger" />
    </layout>
  </parameter>
</appender>
```

```
</layout>
</parameter>
<parameter>
  <parameterName value=":message" />
  <dbType value="String" />
  <size value="4000" />
  <layout type="log4net.Layout.PatternLayout">
    <conversionPattern value="%message" />
  </layout>
</parameter>
<parameter>
  <parameterName value=":exception"/>
  <dbType value="String"/>
  <size value="4000"/>
  <layout type="log4net.Layout.ExceptionLayout"/>
</parameter>
</appender>
```

To create a table in oracle that is compatible with L4Ndash you can use the statement below:

Oracle have no concept of Identity columns (as SqlServer), to achieve the same functionality we create a trigger on the log table. The trigger uses a Oracle sequence to get the next value for the Id column.

```
CREATE TABLE Log4Net
(
  Id          NUMBER(18) NOT NULL
  ,
  DateTime   TIMESTAMP(3) NOT NULL,
  Thread     VARCHAR2(255)
  ,
  Log_Level  VARCHAR2(255)
  ,
  Logger     VARCHAR2(255)
  ,
  Message    VARCHAR2(4000)
  ,
  Exception  VARCHAR2(4000)
);

ALTER TABLE Log4Net ADD
(
  CONSTRAINT Log4Net_pk PRIMARY KEY (Id)
)
;

CREATE Sequence Log4Net_seq;
CREATE OR REPLACE TRIGGER Log4Net_BIns Before
  INSERT ON Log4Net FOR EACH ROW WHEN (new.Id IS NULL) BEGIN
  SELECT Log4Net_seq.NEXTVAL INTO :new.id FROM dual;
END;
```

THE XML TEXTFILE PROVIDER

The Xml TextFile provider reads log rows from one or several text file.

The text file(s) must be created/generated from Log4Net using the "log4net.Layour.XmlLayout". An example of a suitable appender definition is found later in this section.

L4NDash uses FileSystemWatcher objects to detect when log files is changes. Based on notifications from the FileSystemWatcher the provider will read new log events from the log files asynchronously and update the internal Cache.

The data source definition for the text file provider is:

```
<datasource name="LocalXMLFile" Description="Local XML file">
  <provider type="L4NDashXmlTextFileProvider.XmlTextFileProvider"
    assembly="L4NDashXmlTextFileProvider, Version=2.4, Culture=neutral"/>

  <providersettings>
    <filename value="c:\Logs\Log.xml" />
    <MaxRows value="1000" />
    <SleepOnRename value="100" />

  </providersettings>
</datasource>
```

Parameter	Explanation
Filename	<p>The name of the file(s) that the provider will retrieve log events from.</p> <p>You can specify several log files. If you do, you must separate them by comma. For example:</p> <p>filename value="c:\Logs\Log.xml, c:\logs\Log.Xml.1, c:\Logs\Log.Xml.2"</p> <p>The filename can also include wild characters, for example:</p> <p>filename value="c:\Logs\Log*.xml, c:\Logs2\Log*.Xml"</p>
MaxRows	<p>The maximum number of rows the provider will return to the Log Rows Panel. This number will not affect the number of rows that are used calculating the Summary Table.</p>

SleepOnRename	<p>The number of milliseconds the provider will sleep when it is notified from a FileSystemWatcher that one of the log files is renamed.</p> <p>The reason for having this parameter is to allow Log4net's RollingFileAppender to finalize the renamed operation for all involved files before the provider resets itself and retrieves all the involved files.</p>
---------------	---

LOG FILE ACCESS / SECURITY

Log files are often stored outside *L4NDashWedbRoot*. This usually means that you need to assign read access to the log file(s) to *L4NDashUserAccount*.

PERFORMANCE / MEMORY CONSUMPTION

The internals of the provider:

- The first time the provider is used it will read all the log files and create an internal cache containing the log events from all the files. This cache will be shared among all users.
- When a log file is changed (new log events is added) the provider will detect this by use of SystemFileWatcher objects. It will only retrieve the newly added log events (it keep track of the size of the files).
- If you are using the RollingFileAppender and one of the log files are renamed (as a result of a "rolling" operation) it will be detected, the internal cache will be cleared and all the log files will be retrieved to build a new cache (see also the SleepOnRename parameter).

Since the Xml Text file provider keeps all the log events in memory, it is quick. The exception to this is the initial load both on start up and on file renames operations.

However, this comes with a cost; it consumes memory. As a rule, you can take the size of the log files and multiply it by 10 to calculate the amount of memory needed.

For logging scenarios where the size of the logs is significant and the use of the XmlTextFileProvider is not feasible, you may want to consider logging to a database instead using the AdoNetAppender.

LOG4NET APPENDER DEFINITION

Below is an example of a log4net appender definition that will produce a log file compatible with the *L4NDash* text file provider. Please note that the prefix value of the XmlLayout is empty i.e. "". From version 2.3 of *L4NDash* the default prefix in log4net will also be handled (a prefix value of "log4net"), that is the prefix value used by the XmlLayout when the prefix parameter is left out.

```
<appender name="FileAppender" type="log4net.Appender.FileAppender">
  <file value="c:\\Logs\\Log.xml"/>
  <appendToFile value="true"/>
  <locationinfo value="false"/>
  <layout type="log4net.Layout.XmlLayout">
    <param name="Prefix" value=""/>
  </layout>
</appender>
```

THE WINDOWS EVENTS LOG PROVIDER

The Windows Event Log provider reads log rows from the windows event log. This provider is not limited to deal with log rows generated from Log4Net but can show all log events from the Windows event log.

```
<datasource name="LocalEventLog" Description="Windows event log">
  <provider type="L4NDashEventLogProvider.EventLogProvider"
    assembly="L4NDashEventLogProvider, Version=2.2, Culture=neutral"/>

  <providersettings>
    <machinename value="."/>
    <logname value="system;application;security"/>
    <cachetime value="60">
  </providersettings>
</datasource>
```

Parameter	Explanation
Machinename	Points to the machine to fetch the event log from use (period) to fetch the log from the local machine.
Logname	Names of the logs that should be retrieved. Separate the different log names by semicolon.
Cachetime	Describes how often the provider will retrieve data from the log. The cachetime is specified in seconds.

LOG4NET APPENDER DEFINITION

Below is a simple example on a log4Net appender definition for an EventLogAppender.

```
<appender name="EventLogAppender" type="log4net.Appender.EventLogAppender" >
  <layout type="log4net.Layout.PatternLayout">
    <conversionPattern value="[%thread] %-5level %logger - %message%newline" />
  </layout>
</appender>
```

USING PREDEFINED SEARCH FILTERS

For all *Data Sources* it is possible to define a file containing predefined search filters. The file can be updated from the filter panel in the Log4Net Dashboard application.

```
<datasource name="xxxxx" Description="xxxxxxx">  
  <provider type="xxxxxx" assembly="xxxxxxx"/>  
  <predefinedsearchfilter value="File name path"/>
```

Parameter	Explanation
Predefinedsearchfilter	The name of the file where predefined search filters are defined / stored. The file name path is relative to the application directory.

It is possible to store predefined search filters from several datasources into the same file, i.e. several *Data Sources* may point to the same file.

If the end, users shall have the possibility to update and store new filters, the *L4NDashUserAccount* need to have write access to the folder where the predefined search filters are stored.

ADDING CUSTOM COLUMNS

It is common to log *Custom Columns* in addition to the *Standard Columns* (Date, Level, Logger, Thread, Message and Exception). This paragraph explains how you setup *L4NDash* to display these columns.

Custom Columns can be defined different for each data source. The definition of *Custom Columns* is therefore added to the data source definition in the web.config file. The generalized web.config syntax for this is shown below.

```
<datasource .....=" ">
    <customcolumns>
        <customcolumn name="MyDatabaseColumnName">
            <label value="MyLabelr" />
            <rowstabledisplay value="yes|no" />
            <rowsdetaildisplay value="yes|no" />
            <filter value="yes|no" />
        </customcolumn>

        <!-- more custom columns definitions -->
    </customcolumns>
</datasource>
```

The section `<customcolumn>` needs to be repeated for each column. For one column you can define the parameters:

Parameter	Explanation
Name	The name of the column in the data source.
Label	The label you want to be displayed in the dashboard.
RowsTableDisplay	Yes or No, decides if the custom column is displayed in the log rows panel.
RowsDetailDispaly	Yes or No, decides if the custom column is displayed in the log row detail panel.
Filter	Yes or no, defines if the custom column should be included in the filter panel, thus giving the user the ability to use the column for filtering.

In version 2.1 it was possible to define a "substring display" using the parameter format `substringg(99)`. This functionality has been replaced by the Visualizer, and the parameters defining the Visualizers.

If you for example, have three *Custom Columns*: *HostName*, *UserName* and *UserAccountData*, and you would like *Hostname* and *Username* to be displayed in the *Log Row Panel* and in the *Log Row Detail Panel*.

In addition you want the *UserAccountData* to be displayed in the *Log Row Detail Panel* and Make the *HostName* and *UserName* available for filtering.

Then your custom columns section would be:

```
<customcolumns>
  <customcolumn name="HostName">
    <label value="Host Name" />
    <rowstabledisplay value="yes" />
    <rowsdetaildisplay value="yes" />
    <filter value="yes" />
  </customcolumn>

  <customcolumn name="UserName">
    <label value="User" />
    <rowstabledisplay value="yes" />
    <rowsdetaildisplay value="yes" />
    <filter value="yes" />
  </customcolumn>

  <customcolumn name="UserAccountData">
    <label value="Account information" />
    <rowstabledisplay value="no" />
    <rowsdetaildisplay value="yes" />
    <filter value="no" />
  </customcolumn>
</customcolumns>
```

When you restrict the display of a custom column the full column will be available if the user click on the "Show message on separate page" link on the bottom of the detail pane.

CUSTOM COLUMNS IN XML LOG FILES

When you use XML log files as the *Data Source*, the log file can contain the xml elements:

- Properties
- Global-properties
- LocationInfo

These xml elements contain several attributes / subnodes. You can choose if you want to display one or several of the attributes /subnoes as separate *Custom Columns* or the whole element as one *Custom Column*.

If you choose to display the whole element as one *Custom Column* you need to use the element name as the name of the *custom column* (properties, global-properties or locationinfo).

If you choose to display single attributes / subnodes you name the *Custom Column* with the name of the attribute / subnode, for example "class" that is an attribute within the locationinfo element.

The following attributes are by default available:

- From locationinfo (if your appender is configured to log locationinfo):
 - class
 - method
 - file
 - line
- From properties:
 - HostName
- In addition to the *Standard Columns* and xml log file will contain:
 - Domain
 - Username

EXPANDING THE WIDTH OF THE WEB PAGE

If you add several columns you may want to expand the width of the web page. To expand the width of the web page you simply need to modify the style sheet defined in the MainStyle.css file (The MainStyle.css file is located on the *L4NDashWebRoot*).

The width of an *L4NDash* page is defined in the .Container “width” parameter. If you increase this you also need to reposition the help panel (which is displayed on the right hand side of a page). This is done by increasing the .HelpPanel left parameter according to the same size as you did on the .Container width parameter.

VISUALIZERS

L4NDash comes with one standard visualizer assembly. This assembly contains three different visualizers:

- String visualizer
- Datetime visualizer
- XMI Visualizer

A visualizer can render a column value in 3 different ways.

- Short display; this format is used in the Log Rows panel.
- Medium display; this format is used in the Log Rows Detail panel.
- Long display; this format is used when a log event is displayed on a separate page.

A visualizer is defined by inserting a “Visualizer” block into the datasource definition in the web.config. The options for configuration a visualizer is dependent of the kind of visualizer you are using, however the general configuration section is:

```
<Visualizer ColumnName="The name of the column">
  <provider type="Class and namespace of visualizer"
    assembly="Assembly reference" />

  <providersettings>
    <!-- provider dependent parameters -->
  </providersettings>
</Visualizer>
```

Parameter	Explanation
ColumnName	Name of the column, can be one of the <i>standard columns</i> , or one of the <i>custom columns</i> . When defining visualizers for <i>custom columns</i> , the column name must refer to the name (and not the label) used in the <i>custom column</i> definition.
Type (in provider)	The class name of the class implementing the provider. This must include the full namespace.

Assembly	<p>The assembly where the “Type” can be found. The format of the assembly reference string is:</p> <pre>AssemblyName [,Version=9.9.9.9] [,Culture=Culture][,PublicKeyToken=Token]</pre> <p>A typical example of a assembly reference is:</p> <pre>L4NDashDefaultVisualizer.DefaultStringVisualizer, Version=2.4,Culture=neutral</pre>
----------	---

DATETIME VISUALIZER

To define a DateTime visualizer you use the Visualizer config statement:

```
<Visualizer ColumnName="The name of the column">
  <provider type="L4NDashDefaultVisualizer.DefaultDateTimeVisualizer"
    assembly="L4NDashDefaultVisualizer, Version=2.4,Culture=neutral"/>

  <providersettings>
    <ShortDisplayFormat value="MM/dd/yy HH:mm:ss"/>
    <MediumDisplayFormat value="MM/dd/yyyy HH:mm:ss"/>
    <LongDisplayFormat value="MM/dd/yyyy HH:mm:ss"/>
  </providersettings>
</Visualizer>
```

Parameter	Explanation
ShortDisplayFormat	The date time format used when the column is displayed in the <i>Log Row Panel</i> .
MediumDisplayFormat	The date time format used when the column is displayed in the <i>Log Row Detail Panel</i> .
LongDisplayFormat	The date time format used when the column is displayed in a separate page.

The date time formats can use the same formatting options as in .NET, or to be more precise the formatting accepted by `DateTime.ToString` method.

STRING VISUALIZER

To define a String visualizer you use the config statement:

```
<Visualizer ColumnName="The name of the column">
  <provider type="L4NDashDefaultVisualizer.DefaultStringVisualizer"
    assembly="L4NDashDefaultVisualizer, Version=2.4, Culture=neutral"/>

  <providersettings>
    <ShortDisplayMaxLength value="500"/>
    <ShortDisplayTruncate value="right"/>
    <MediumDisplayMaxLength value="500"/>
    <MediumDisplayTruncate value="right"/>
  </providersettings>
</Visualizer>
```

Parameter	Explanation
ShortDisplayMaxLength	The output will be truncated to this length when the column is displayed in the <i>Log Row Panel</i>
ShortDisplayTruncate	Left or Right. Defines if the truncation should be done in the start (left) or at the end (right) of the string.
MediumDisplayMaxLength	The output will be truncated to this length when the column is displayed in the <i>Log Row Detail Panel</i>
MediumDisplayTruncate	Left or Right. Defines if the truncation should be done in the start (left) or at the end (right) of the string.

XML VISUALIZER

Logging blocks of XML is a common way to expand log4Net logging. The Xml Visualizer provides you with formatting and syntax coloring of xml blocks.

To define a XML visualizer you use the config statment:

```
<Visualizer ColumnName="The name of the column">
  <provider type="L4NDashDefaultVisualizer.DefaultXmlVisualizer"
    assembly="L4NDashDefaultVisualizer, Version=2.4, Culture=neutral"/>

  <providersettings>
    <ShortDisplayMaxLength value="60"/>
    <MediumDisplayMaxLength value="500"/>
  </providersettings>
</Visualizer>
```

Parameter	Explanation
ShortDisplayMaxLength	The output will be truncated to this length when the column is displayed in the <i>Log Row Panel</i>
MediumDisplayMaxLength	The output will be truncated to this length when the column is displayed in the <i>Log Row Detail Panel</i>

OTHER PARAMETERS IN THE <APPSETTINGS> SECTION

The web.config file contains additional parameters you can change:

Parameter	Explanation
DateTimeFormat	How date time values are displayed. You can use the formats defined in .NET framework, for example "MM/dd/yyyy HH:mm:ss"
DateFormat	How date values are displayed. You can use the formatters defined in .NET framework, for example "MM/dd/yyyy"
UserInputDateFormat	Date format for user input.
SummaryTableCacheTime	Number of seconds the <i>Summary Table</i> will be kept in memory. During this time the <i>Summary Table</i> will not be updated to reflect changes in the data source. Altering this parameter can increase / decrease performance.
RowTabelShowMessage RowTableShowLogger	Valid values are 0 (false) and 1 (true). Use these parameters to turn off the display of the Message and/or the logger column in the <i>Log Row Panel</i> .
HelpModus	Valid values are 0 (false) and 1 (true). Use this to turn on/off display of the help text that exists for each web page.
SkipVersionCheck	1 or 0. <i>L4NDash</i> will from time to time check if there is a new version of <i>L4NDash</i> available from www.l4ndash.com . If a new version is available the user will be notified. If you set this parameter to 1, the version check will not be performed.