Using Apache's log4net in ASP.NET

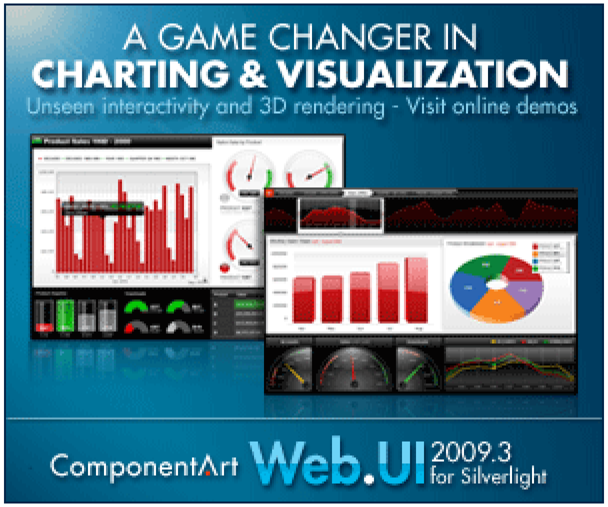
Published: 07 Dec 2004
Unedited - Community Contributed

Abstract
In this simple, seven-step guide, Anubhav (Ash) Srivastava shows how to use Apache's log4net component to add logging capabilities to your ASP.NET applications.

by Anubhav(Ash) Srivastava
Feedback

Average Rating:
Views (Total / Last 10 Days): 76611/ 384

Article Contents:



Introduction

[Back To Top]

[Download Code]

Log4net is a .NET component from Apache Foundation that provides a simple mechanism for logging messages. Log4net provides seven levels of logging, which are listed below from lowest to highest priority:

1. ALL
2. DEBUG
3. INFO

4. WARN
5. ERROR
6. FATAL
7. OFF

These levels help control the output to log files. If in the configuration file the level for a class is set to ERROR, then all messages from that class at a level below ERROR (i.e., DEBUG, INFO or WARN) will not be logged. This will come in very handy when the application goes into production. If you find yourself a little confused, do not worry. Let's proceed with a sample application and you will soon be a log4net pro.

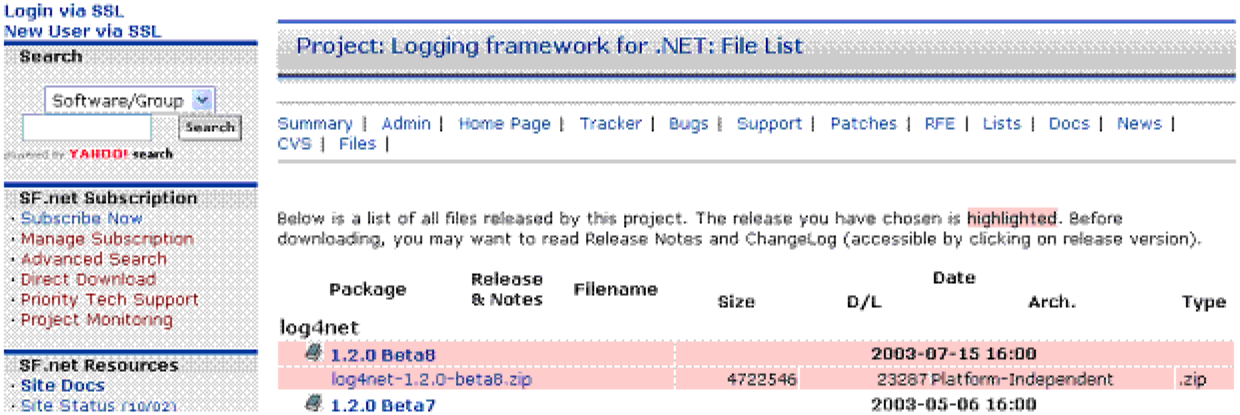
Creating the Sample Application

[Back To Top]

[Download Code]

Step 1: Download log4net

Visit logging.apache.org/log4net and select “download at sourceforge.net” from the navigation menu. You will see the SourceForge page as shown in Figure 1. Download the file named 1.2.0 Beta8 (log4net-1.2.0-beta8.zip) and unzip this archive to your C:\ directory.



Project: Logging framework for .NET: File List

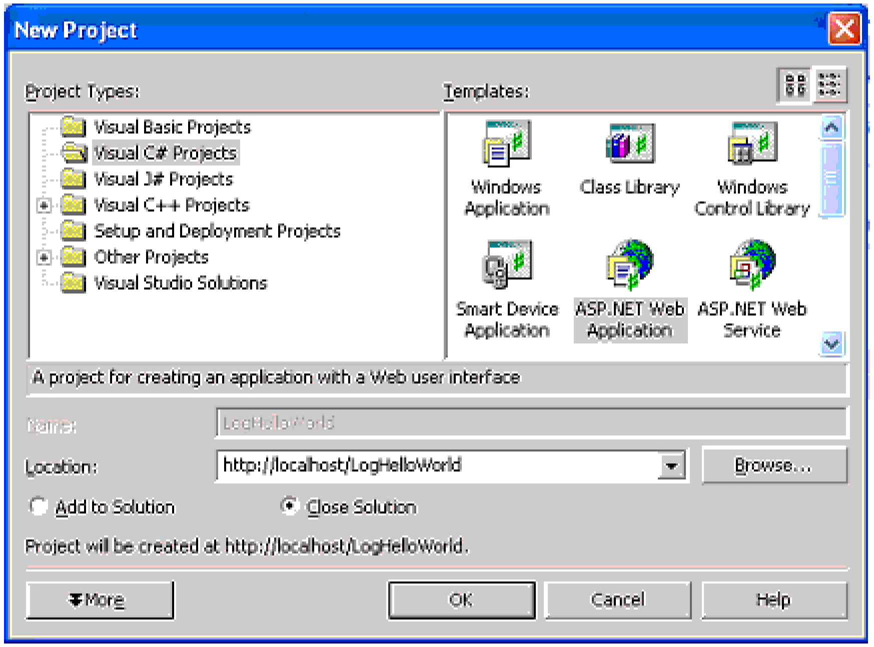
Below is a list of all files released by this project. The release you have chosen is highlighted. Before downloading, you may want to read Release Notes and ChangeLog (accessible by clicking on release version).

Package	Release & Notes	Filename	Date	Size	D/L	Arch.	Type
log4net							
	1.2.0 Beta8		2003-07-15 16:00				
		log4net-1.2.0-beta8.zip		4722546	23287	Platform-Independent	.zip
	1.2.0 Beta7		2003-05-06 16:00				

Figure 1

Step 2: Create a new ASP.NET Application

Start Visual Studio .NET and create a new Web Application project named LogHelloWorld, as shown in Figure 2.



A project for creating an application with a Web user interface

Name: LogHelloWorld

Location: http://localhost/LogHelloWorld

○ Add to Solution ● Close Solution

Project will be created at http://localhost/LogHelloWorld.

Figure 2

Click OK. In the Solution Explorer window, right-click on References and select Add a Reference. Select Browse and navigate to C:\log4net-1.2.0-beta8\bin\net. There, select the subfolder matching the version of .NET you are using. Inside its \release subfolder, select the log4net.dll file.

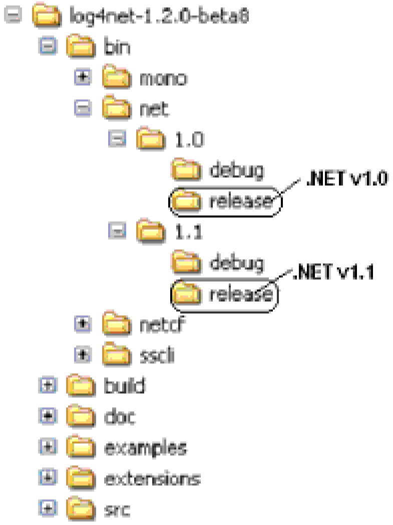


Figure 3

After you have selected the file, click OK, and you will see that the log4net assembly has been added as a reference.

[Register](#)
[Edit My Profile](#)
[Author List](#)
[Write for Us](#)
[About AspAlliance](#)
[Contact Us](#)
[Privacy Policy](#)
[Link To Us](#)
[Advertise](#)

Subscribe
[Free Newsletter](#)
[Newsletter Archive](#)
[RSS Syndication](#)
RSS

.NET Tutorials
[Learn .NET](#)
[Learn WCF](#)
[Learn WPF](#)
[Learn ASP.NET](#)
[Learn AJAX](#)
[Learn Silverlight](#)
[Learn Visual Studio](#)
[Learn ADO.NET](#)
[Learn LINQ](#)
[Learn C# (CSharp)](#)
[Learn VB.NET](#)
[Learn Web Services](#)
[Learn Controls](#)
[Learn BizTalk](#)
[Learn SharePoint](#)
[Learn Mobile](#)
[Learn SQL](#)
[Learn SQL Reporting](#)
[Learn Windows Forms](#)
[Learn XML](#)
[Learn Crystal Reports](#)
[Learn FarPoint](#)
[Learn DevExpress](#)
[Examples](#)
[ASP.NET 2.0](#)

Configuring the Sample Application

[Back To Top]

[Download Code]

Step 3: Add Assembly Information

Open the AssemblyInfo.cs file and add the following line:

```
[assembly: log4net.Config.DOMConfigurator()]
```

The above line is needed because the log4net configuration can be configured using assembly-level attributes rather than specified programmatically. If you are using your own configuration file rather than Web.config then you will have to specify it here along with its extension. You also have the option to set the Watch flag to true, which will make log4net reload the config each time the file is modified. So it will look like the following.

```
[assembly:
log4net.Config.DOMConfigurator(ConfigFile="filename",ConfigFileExtension="log4net",Watch=true)]
```

Step 4: Add Configuration Information

Open the Web.config file and add the following code within its configuration tags. It is important that the configSections node is placed immediately after the opening configuration tag, as shown below.

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
    <section name="log4net"
             type="log4net.Config.Log4NetConfigurationSectionHandler, log4net-net-1.0"
    />
  </configSections>

  <!-- This section contains the log4net configuration settings -->
  <log4net>
    <!-- Define some output appenders -->
    <appender name="ConsoleAppender" type="log4net.Appender.ConsoleAppender">
      <layout type="log4net.Layout.PatternLayout">
        <param name="ConversionPattern" value="%d [%t] %-5p %c [%x] &lt;%X{auth}&gt; - %m%n" />
      </layout>
    </appender>
    <!-- RollingFileAppender looks after rolling over files by size or date -->
    <appender name="RollingFileAppender" type="log4net.Appender.RollingFileAppender">
      <param name="File" value="C:\\log\\RollingLogHelloWorld.log" />
      <param name="AppendToFile" value="true" />
      <param name="MaxSizeRollBackups" value="10" />
      <param name="MaximumFileSize" value="1000" />
      <param name="RollingStyle" value="Size" />
      <param name="StaticLogFileName" value="true" />
      <layout type="log4net.Layout.PatternLayout">
        <param name="ConversionPattern" value="%d [%t] %-5p %-45c [%x] - %m%n" />
      </layout>
    </appender>
    <!-- FileAppender appends to a log and it is manually managed or size -->
    <appender name="FileAppender" type="log4net.Appender.FileAppender">
      <param name="File" value="LogHelloWorld.log" />
      <!-- Example using environment variables in params -->
      <!-- <param name="File" value="${TMP}\\ApplicationKit.log" /> -->
      <param name="AppendToFile" value="true" />
```

Examples

ASP Tutorials
Learn ASP
Learn VBScript
Learn JScript
Learn SQL
Learn XML

Software Resources
Shopping Cart
Ecommerce
Charts and Dashboards

Other Resources
Learn Java
Learn Oracle
Opinion / Editorial
Crystal Reports
Alliance
WPF Resources
AJAX Resources
Silverlight Resources

Free Tools
Cache Manager
SimpleCMS

Reviews
Book Reviews
Product Reviews
Expert Advice

Books
ASP.NET Developer's Cookbook
Sample Chapters
Book Reviews

Community
Developer Jobs
Resource Directory

```
      <layout type="log4net.Layout.PatternLayout">
        <param name="ConversionPattern" value="%d [%t] %-5p %c [%x] - %m%n" />
      </layout>
    </appender>
    <!-- Setup the root category, add the appenders and set the default level -->
    <root>
      <level value="INFO" />
      <appender-ref ref="ConsoleAppender" />
    </root>
    <!-- ApplicationKit category - the presentation UI -->
    <logger name="WebForm1">
      <level value="INFO" />
      <appender-ref ref="FileAppender" />
    </logger>
  </log4net>
</configuration>
```

Towards the end of the configuration file above, please note the logger node with its name attribute set to WebForm1. This identifies the name of our class. For every class in which you want to use logging, you must add an additional logger node to the log4net section of the Web.config file.

In the logger node above, the level is set to INFO, so all messages that are of lower priority than INFO will not be logged for this WebForm1 class.

Running the Sample Application

[Back To Top]

[Download Code]

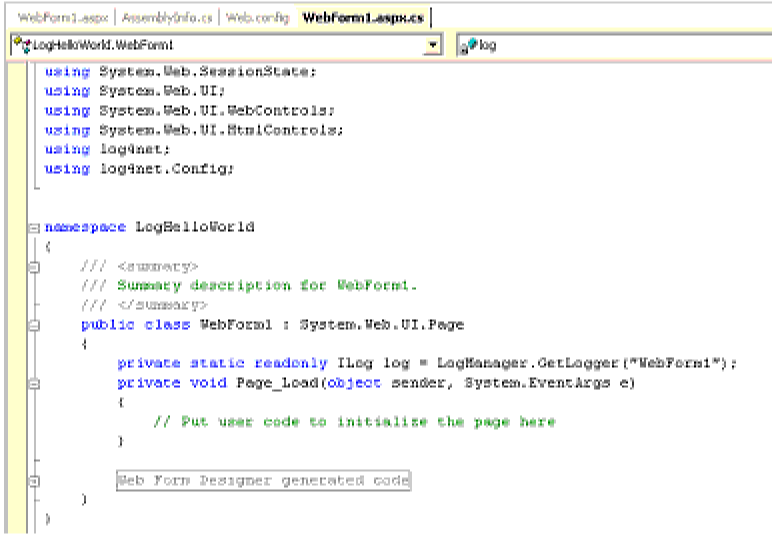
Step 5: Add a Log Object

Add the following directives to the top of the WebForm1.aspx.cs code-behind file.

```
using log4net;
using log4net.Config;
```

Add the following line in the WebForm1 class; its placement is shown in Figure 4 below.

```
private static readonly ILog log = LogManager.GetLogger("WebForm1");
```


WebForm1.aspx | AssemblyInfo.cs | Web.config | WebForm1.aspx.cs
LogHelloWorld.WebForm1
log

```
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
using log4net;
using log4net.Config;


namespace LogHelloWorld
{
    /// <summary>
    /// Summary description for WebForm1.
    /// </summary>
    public class WebForm1 : System.Web.UI.Page
    {
        private static readonly ILog log = LogManager.GetLogger("WebForm1");
        private void Page_Load(object sender, System.EventArgs e)
        {
            // Put user code to initialize the page here
        }

        Web Form Designer generated code
    }
}
```

Figure 4

Step 6: Use the Log Object

Our application is now ready to log messages from the WebForm1 class. To test this, drag a Button control from the toolbox onto WebForm1.

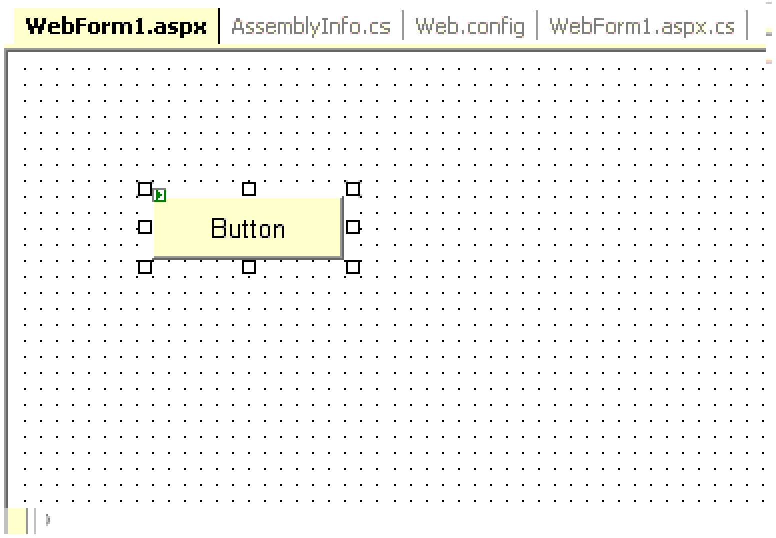


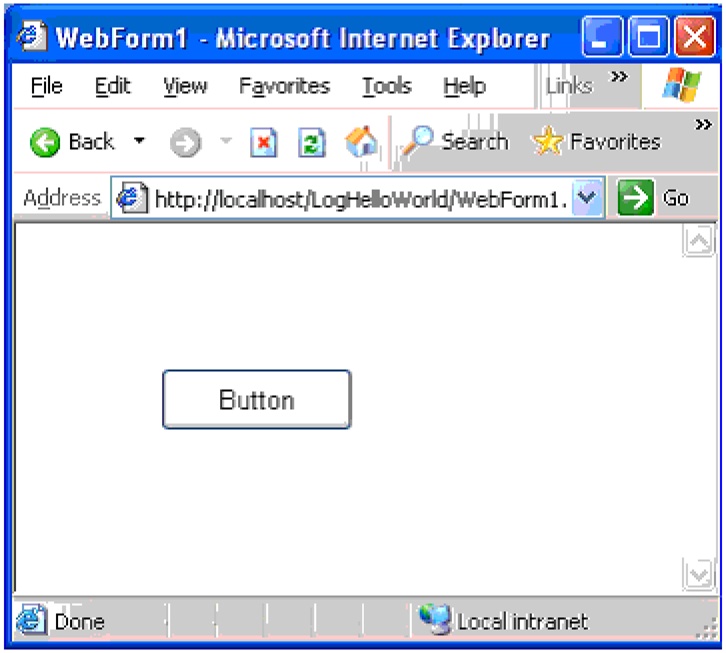
Figure 5

Double-click the Button and add the following code to the Button1_Click event handler.

```
log.Info("Hello World, I am a logger");
```

Step 7: Run the Application

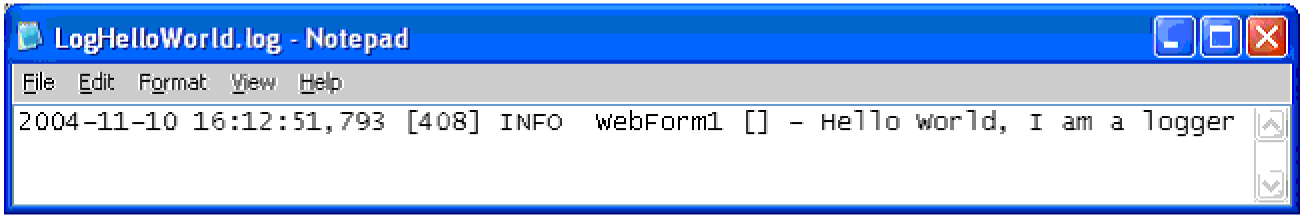
We are now ready to test our application and its logging component. To launch the application, press F5. A web page similar to the following should appear in your browser:


WebForm1 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links
Back
Search
Favorites
Go
Button
Done
Local intranet

Address: http://localhost/LogHelloWorld/WebForm1.

Figure 6

Click on the button to trigger a postback, during which log4net will record our message. Using Notepad, open the file named LogHelloWorld.log in the application's \bin\Debug directory. All going well, you will see a log similar to that shown below.



```
2004-11-10 16:12:51,793 [408] INFO  WebForm1 [] - Hello World, I am a logger
```

Figure 7

Conclusion

I hope that this article has demonstrated how easy it is to add the log4net component to your ASP.NET application. For more information, please visit Apache's log4net homepage:

http://logging.apache.org/log4net/

Article Feedback

Title: ____________________

Name: ____________________

Url: ____________________ *(Optional)*

Comment:

____

Please add 4 and 1 and type the answer here: ____________

Enter Comment

User Comments

Title: Excelent Artical
Name: Mallik
Date: 7/2/2009 6:15:27 AM
Comment:
This is superb article for Log4Net, I want to know about log4net more and more, can you please give me some URLs or send meterials on Log4Net? My mail id is rao.net4@gmail.com

Thanks

Title: Great Article
Name: Mahanth
Date: 6/19/2008 12:08:36 PM
Comment:
This is a very Great article to get a complete knowledge of log4net.
Thanks for the article.

Title: missing step in application
Name: a

Date: 2/6/2008 4:15:02 AM
Comment:
The one part missing in the application is a call to log4net.Config.XmlConfigurator.Configure();

before calling GetLogger
log = LogManager.GetLogger("WebForm1");

Title: no logging bwing done
Name: Priya Hurry
Date: 12/25/2007 5:08:08 AM
Comment:
Hello...I have done exactly what you have been doing...I have tried with the sample application as well...but not log is being changed...
When I debug the application, I can see that the isDebugEnabled,isInfoEnabled,isWarnEnabled,isErrorEnabled,isFatalEnabled is false....Do u think that this is the problem??

Pls reply me on prya.hurry@dcdm.intnet.mu....thanx

Title: Untitled
Name: Harry
Date: 8/17/2007 8:56:59 PM
Comment: