**Home**

**Java Core**

**Java SE**

**Java EE**

Frameworks

**Servers**

**Coding**

**IDEs**

**Books**

**Videos**

**Certifications**

**Testing**

Home ▸ Frameworks ▸ Spring

**Learn Spring framework:**

- Understand the core of Spring
- Understand Spring MVC
- Understand Spring AOP
- Understand Spring Data JPA
- Spring Dependency Injection (XML)
- Spring Dependency Injection (Annotations)

Spring Dependency Injection (Java config)

- Spring MVC beginner tutorial
- Spring MVC Exception Handling
- Spring MVC and log4j
- Spring MVC Send email
- Spring MVC File Upload
- Spring MVC Form Handling
- Spring MVC Form Validation
- Spring MVC File Download
- Spring MVC JdbcTemplate
- Spring MVC CSV view
- Spring MVC Excel View
- Spring MVC PDF View
- Spring MVC XstlView
- Spring MVC + Spring Data JPA + Hibernate - CRUD
- Spring MVC Security (XML)
- Spring MVC Security (Java config)
- Spring & Hibernate Integration (XML)
- Spring & Hibernate Integration (Java

config)

- Spring & Struts Integration (XML)
- Spring & Struts Integration (Java config)
- 14 Tips for Writing Spring MVC Controller

# How to use log4j in Spring MVC

Written by Nam Ha Minh

Last Updated on 20 June 2019 | Print Email

This tutorial guides you how to setup logging utility log4j for a Spring MVC applicaiton.

You know, Log4j is a popular and widely-used logging framework for Java development. It's pretty easy to setup and use log4j in a Spring MVC application, just by following a few steps described below.

## 1. Adding log4j library jar file

First, download log4j distribution, extract the zip archive and add the `log4j-1.2.17.jar` file into the `WEB-INF\lib` directory of your Spring MVC web application.

If you are using Maven, add the following entry into `pom.xml` file:

```
<dependency>
  <groupId>log4j</groupId>
  <artifactId>log4j</artifactId>
  <version>1.2.17</version>
  <scope>runtime</scope>
</dependency>
```

## 2. Creating log4j configuration file

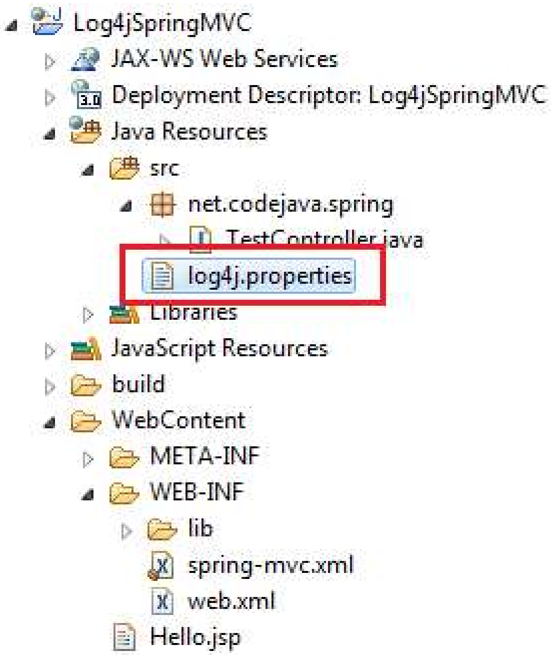
Create a `log4j.properties` file with the following content:

```
# LOG4J configuration
log4j.rootLogger=INFO, Appender1, Appender2

log4j.appender.Appender1=org.apache.log4j.ConsoleAppender
log4j.appender.Appender1.layout=org.apache.log4j.PatternLayout
log4j.appender.Appender1.layout.ConversionPattern=%-7p %d [%t] %c %x - %m%n

log4j.appender.Appender2=org.apache.log4j.FileAppender
log4j.appender.Appender2.File=D:/Logs/SpringMVC.log
log4j.appender.Appender2.layout=org.apache.log4j.PatternLayout
log4j.appender.Appender2.layout.ConversionPattern=%-7p %d [%t] %c %x - %m%n
```

That specifies log level as INFO and configures two appenders: one for console logging and one for file logging (Path of the log file is `D:/Logs/SpringMVC.log`, so make sure the directory exists). If you prefer XML, here’s the equivalent version in XML (`log4j.xml`):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE log4j:configuration PUBLIC "-//log4j/log4j Configuration//EN" "log4

<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

    <appender name="Appender1" class="org.apache.log4j.ConsoleAppender">
       <layout class="org.apache.log4j.PatternLayout">
          <param name="ConversionPattern" value="%-7p %d [%t] %c %x - %m%n"/>
       </layout>
    </appender>

    <appender name="Appender2" class="org.apache.log4j.FileAppender">
       <param name="File" value="D:/Logs/SpringMVC2.log" />
       <layout class="org.apache.log4j.PatternLayout">
          <param name="ConversionPattern" value="%-7p %d [%t] %c %x - %m%n"/>
       </layout>
    </appender>

    <root>
        <priority value="INFO"/>
        <appender-ref ref="Appender1" />
        <appender-ref ref="Appender2" />
    </root>

</log4j:configuration>
```

Then put the `log4j.properties` file or `log4j.xml` file under the root of the classpath (under `WEB-INF\classes` directory of your web application). In Eclipse, put it under `src` directory like in the following screenshot:



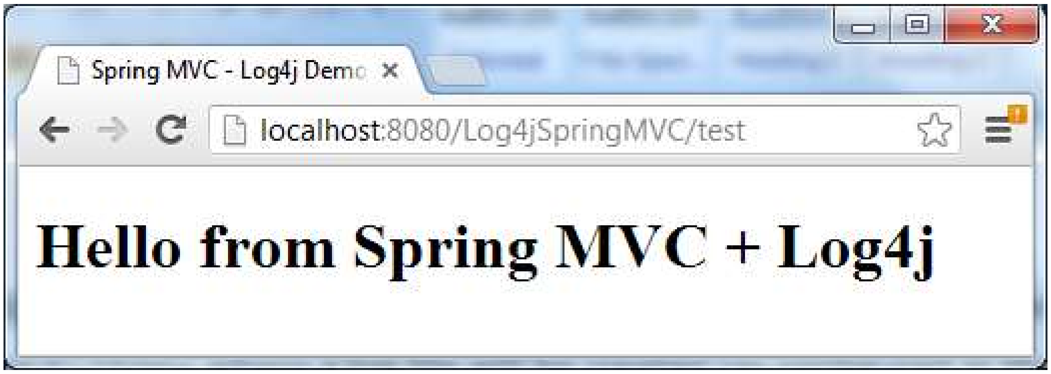
And that's all! Now we can write some code in a Spring controller class to use log4j as follows:

```java
package net.codejava.spring;

import org.apache.log4j.Logger;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

@Controller
public class TestController {

    private Logger logger = Logger.getLogger(TestController.class);

    @RequestMapping(value = "/test", method = RequestMethod.GET)
    public ModelAndView test() {
        ModelAndView model = new ModelAndView("Hello");

        logger.info("This is an info log entry");
        logger.error("This is an error log entry");

        return model;
    }
}
```

Now let's test it out. For the demo application in this tutorial, type the following URL into browser's address bar:

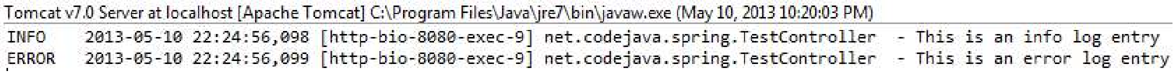
*http://localhost:8080/Log4jSpringMVC/test*

Output in the browser:



Hello from Spring MVC + Log4j

And notice some log entries are appended to the server’s console as follows:



```
Tomcat v7.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jre7\bin\javaw.exe (May 10, 2013 10:20:03 PM)
INFO    2013-05-10 22:24:56,098 [http-bio-8080-exec-9] net.codejava.spring.TestController  - This is an info log entry
ERROR   2013-05-10 22:24:56,099 [http-bio-8080-exec-9] net.codejava.spring.TestController  - This is an error log entry
```

And the log entries are also written to the log in the `D:/Logs/SpringMVC.log file` file.

## Related Log4j Tutorials:

- How to use log4j in Java
- Configure log4j for creating daily rolling log files
- Common conversion patterns for log4j's PatternLayout

## Other Spring Tutorials:

- Understand the core of Spring framework
- Understand Spring MVC
- Spring MVC Form Handling Tutorial
- Spring MVC Form Validation Tutorial
- Spring MVC with JdbcTemplate Example
- Spring MVC + Spring Data JPA + Hibernate - CRUD Example
- 14 Tips for Writing Spring MVC Controller
- Spring and Hibernate Integration Tutorial Part (XML Configuration)

## About the Author:

Nam Ha Minh is certified Java programmer (SCJP and SCWCD). He started programming with Java in the time of Java 1.4 and has been falling in love with Java since then. Make friend with him on Facebook.

**Attachments:**

**Log4jSpringMVC.zip [ ] 3873 kB**

Add comment

| | |
|---|---|
| Name | E-mail |

Comment

500 symbols left

□ Notify me of follow-up comments

□ I'm not a robot

reCAPTCHA
Privacy - Terms

Send

## Comments

1 2

**#6sibani** 2018-10-04 01:26

if I want to have more than one application having different log 4j config files, then how I have to set those file names?

Quote

**#5Chazz** 2016-08-17 01:07

Learn a lot ,Thx it is useful

Quote

**#4karthi** 2016-03-16 06:34

Very useful...thanks a lot...

Quote

**#3koti** 2015-11-11 05:31

log4j added

Quote

**#2koti** 2015-11-11 05:30

log4j

Quote

1 2

Refresh comments list

About CodeJava.net:

CodeJava.net shares Java tutorials, code examples and sample projects for programmers at all levels.
CodeJava.net is created and managed by Nam Ha Minh - a passionate programmer.

About Advertise Contact Terms of Use Privacy Policy Sitemap Newsletter Facebook Twitter YouTube