# **Java深入 - logback的配置和使用**

原创 2014年11月07日 13:42:26

* 54400

### 1. logback介绍

Logback是由log4j创始人设计的又一个开源日志组件。logback当前分成三个模块：logback-core,logback- classic和logback-access。logback-core是其它两个模块的基础模块。logback-classic是log4j的一个 改良版本。此外logback-classic完整实现SLF4J API使你可以很方便地更换成其它日志系统如log4j或JDK14 Logging。logback-access访问模块与Servlet容器集成提供通过Http来访问日志的功能。

### 2. maven依赖

**[html]** [view plain](http://blog.csdn.net/initphp/article/details/40891821" \o "view plain) [copy](http://blog.csdn.net/initphp/article/details/40891821" \o "copy)

1. <!-- logback+slf4j -->
2. **<dependency>**
3. **<groupId>**org.slf4j**</groupId>**
4. **<artifactId>**slf4j-api**</artifactId>**
5. **<version>**1.6.0**</version>**
6. **<type>**jar**</type>**
7. **<scope>**compile**</scope>**
8. **</dependency>**
9. **<dependency>**
10. **<groupId>**ch.qos.logback**</groupId>**
11. **<artifactId>**logback-core**</artifactId>**
12. **<version>**0.9.28**</version>**
13. **<type>**jar**</type>**
14. **</dependency>**
15. **<dependency>**
16. **<groupId>**ch.qos.logback**</groupId>**
17. **<artifactId>**logback-classic**</artifactId>**
18. **<version>**0.9.28**</version>**
19. **<type>**jar**</type>**
20. **</dependency>**

如果你没有使用maven，那么你自己去下载jar包吧...

### 3. 配置和使用

#### 1. 日志使用

我们使用org.slf4j.LoggerFactory，就可以直接使用日志了。

**[java]** [view plain](http://blog.csdn.net/initphp/article/details/40891821" \o "view plain) [copy](http://blog.csdn.net/initphp/article/details/40891821" \o "copy)

1. **protected** **final** Logger       logger = LoggerFactory.getLogger(**this**.getClass());

使用：

**[java]** [view plain](http://blog.csdn.net/initphp/article/details/40891821" \o "view plain) [copy](http://blog.csdn.net/initphp/article/details/40891821" \o "copy)

1. @Controller
2. @RequestMapping(value = "")
3. **public** **class** IndexController **extends** BaseController {
5. /\*\*
6. \* Success
7. \* @param response
8. \* @throws IOException
9. \*/
10. @RequestMapping(value = "")
11. @ResponseBody
12. **public** **void** hello(HttpServletResponse response) **throws** IOException {
13. logger.debug("DEBUG TEST 这个地方输出DEBUG级别的日志");
14. logger.info("INFO test 这个地方输出INFO级别的日志");
15. logger.error("ERROR test 这个地方输出ERROR级别的日志");
16. }
18. }

#### 2. 在控制台输出特定级别的日志

logback的配置文件都放在/src/main/resource/文件夹下的logback.xml文件中。其中logback.xml文件就是logback的配置文件。只要将这个文件放置好了之后，系统会自动找到这个配置文件。

下面的配置中，我们输出特定的ERROR级别的日志：

**[html]** [view plain](http://blog.csdn.net/initphp/article/details/40891821" \o "view plain) [copy](http://blog.csdn.net/initphp/article/details/40891821" \o "copy)

1. **<?xml** version="1.0"**?>**
2. **<configuration>**
4. <!-- ch.qos.logback.core.ConsoleAppender 控制台输出 -->
5. **<appender** name="console" class="ch.qos.logback.core.ConsoleAppender"**>**
6. **<encoder>**
7. **<pattern>**[%-5level] %d{HH:mm:ss.SSS} [%thread] %logger{36} - %msg%n**</pattern>**
8. **</encoder>**
9. **</appender>**
11. <!-- 日志级别 -->
12. **<root>**
13. **<level** value="error" **/>**
14. **<appender-ref** ref="console" **/>**
15. **</root>**
17. **</configuration>**

结果只在控制台输出ERROR级别的日志。

#### 3. 设置输出多个级别的日志

**[html]** [view plain](http://blog.csdn.net/initphp/article/details/40891821" \o "view plain) [copy](http://blog.csdn.net/initphp/article/details/40891821" \o "copy)

1. **<?xml** version="1.0"**?>**
2. **<configuration>**
4. <!-- ch.qos.logback.core.ConsoleAppender 控制台输出 -->
5. **<appender** name="console" class="ch.qos.logback.core.ConsoleAppender"**>**
6. **<encoder>**
7. **<pattern>**[%-5level] %d{HH:mm:ss.SSS} [%thread] %logger{36} - %msg%n**</pattern>**
8. **</encoder>**
9. **</appender>**
11. <!-- 日志级别 -->
12. **<root>**
13. **<level** value="error" **/>**
14. **<level** value="info" **/>**
15. **<appender-ref** ref="console" **/>**
16. **</root>**
18. **</configuration>**

设置两个level，则可以输出 ERROR和INFO级别的日志了。

#### 4. 设置文件日志

**[html]** [view plain](http://blog.csdn.net/initphp/article/details/40891821" \o "view plain) [copy](http://blog.csdn.net/initphp/article/details/40891821" \o "copy)

1. **<?xml** version="1.0"**?>**
2. **<configuration>**
4. <!-- ch.qos.logback.core.ConsoleAppender 控制台输出 -->
5. **<appender** name="console" class="ch.qos.logback.core.ConsoleAppender"**>**
6. **<encoder>**
7. **<pattern>**[%-5level] %d{HH:mm:ss.SSS} [%thread] %logger{36} - %msg%n
8. **</pattern>**
9. **</encoder>**
10. **</appender>**
12. <!-- ch.qos.logback.core.rolling.RollingFileAppender 文件日志输出 -->
13. **<appender** name="file"
14. class="ch.qos.logback.core.rolling.RollingFileAppender"**>**
15. **<Encoding>**UTF-8**</Encoding>**
16. **<File>**/home/test.log**</File>**
17. **<rollingPolicy** class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"**>**
18. **<FileNamePattern>**/home/test-%d{yyyy-MM-dd}.log
19. **</FileNamePattern>**
20. **<MaxHistory>**10**</MaxHistory>**
21. **<TimeBasedFileNamingAndTriggeringPolicy**
22. class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP"**>**
23. **<MaxFileSize>**5MB**</MaxFileSize>**
24. **</TimeBasedFileNamingAndTriggeringPolicy>**
25. **</rollingPolicy>**
26. **<layout** class="ch.qos.logback.classic.PatternLayout"**>**
27. **<pattern>**[%-5level] %d{HH:mm:ss.SSS} [%thread] %logger{36} - %msg%n
28. **</pattern>**
29. **</layout>**
30. **</appender>**
32. <!-- 日志级别 -->
33. **<root>**
34. <!-- 定义了ERROR和INFO级别的日志，分别在FILE文件和控制台输出 -->
35. **<level** value="error" **/>**
36. **<level** value="info" **/>**
37. **<appender-ref** ref="file" **/>**
38. **<appender-ref** ref="console" **/>**
39. **</root>**

42. **</configuration>**

#### 5. 精确设置每个包下面的日志

**[html]** [view plain](http://blog.csdn.net/initphp/article/details/40891821" \o "view plain) [copy](http://blog.csdn.net/initphp/article/details/40891821" \o "copy)

1. **<logger** name="com.xxx" additivity="false"**>**
2. **<level** value="info" **/>**
3. **<appender-ref** ref="file" **/>**
4. **<appender-ref** ref="console" **/>**
5. **</logger>**