尚马教育 JAVA 高级课程

日志组件

文档编号：C08

创建日期： 2017-07-07

最后修改日期：2019-10-18

版 本 号：V3.0

电子版文件名：尚马教育-第三阶段-8.日志组件.docx

文档修改记录：

更新日期	更新作者	更新说明	版本号
2017-08-10	张元林	初始版本	V1.0
2018-08-01	王绍成	Mybatis 版本更新	V2.0
2019-10-18	冯勇涛	课件格式以及课程深度加深	V3.0

目录

1. 日志介绍

1.1. 什么是日志

日志是软件应用必备的组件，是程序 debug，或是数据收集管理的重要依据，方便我们监测生产环境的变量值变化以及代码运行轨迹。本课程主要用来学习实际开发中常用的日志组件。

主要是为了方便我们监测生产环境的变量值变化以及代码运行轨迹等。

这些记录会被输出到我们指定的位置形成文件,帮助我们分析错误以及用户请求轨迹。

2. 常用日志组件：

2.1. Log4j 与 log4j2.x

Log4j 是目前最常用的一种日志组件，它本身是 Apache 的一个开放源代码项目。通过使用 Log4j，我们可以控制日志信息输送的目的地是控制台、文件、数据库等；我们也可以控制每一条日志的输出格式；通过定义每一条日志信息的级别，我们能够更加细致地控制日志的生成过程。

Log4j 有 7 种不同的 log 级别，按照等级从低到高依次为：TRACE>DEBUG>INFO>WARN>ERROR>FATAL>OFF。如果配置为 OFF 级别，表示关闭 log。

Log4j 目前两个版本，log4j 与 log4j2。Log4j2 的前身是 log4j,它吸收了 logback 的设计,重新推出的一款新组件,且使用.xml/.json 文件来替换了之前.properties 来配置。Log4j2 采用异步日志器,基于 LMAX Disruptor 库，相比较 log4j 高出 10 倍吞吐量。

Log4j 支持两种格式的配置文件：properties 和 xml。包含三个主要的组件：Logger、appender、Layout。

2.2. 日志门面

2.2.1. Common-logging

Common-logging 简称 JCL,Common-logging 是 Apache 提供的一个通用日志 API，可以让应用程序不再依赖于具体的日志实现工具。该日志接口对其它一些日志工具，包括 Log4J、Avalon LogKit、JUL 等，进行了简单的包装，可以让应用程序在运行时，直接将日志适配到对应的日志实现工具中。

Common-logging 通过动态查找的机制，在程序运行时自动找出真正使用的日志库。这一点与 slf4j 不同，slf4j 是在编译时静态绑定真正的 Log 实现库。

2.2.2. Slf4j

SLF4J 全称 The Simple Logging Facade for Java，简单日志门面，这个不是具体的日志解决方案，而是通过门面模式提供一些 Java Logging API，类似于 JCL。

使用 SLF4J 日志门面时，如果需要使用某一种日志实现，则需要选择正确的 SLF4J 的 jar 包（称为日志桥接）。SLF4J 提供了统一的记录日志的接口，只要按照其提供的方法记录即可，最终日志的格式、记录级别、输出方式等通过具体日志系统的配置来实现，因此可以在应用中灵活切换日志系统。

3. Log4j2 环境搭建

3.1. Log4j2 的 jar 包

3.2. Log4j2 的 xml 配置

```
<?xml version="1.0" encoding="UTF-8"?>

<configuration>

  <appenders>

    <!-- 控制台输出 -->

    <console name="Console" target="SYSTEM_OUT">

        <PatternLayout

            pattern="%d{HH:mm:ss.SSS} [%t] %-5level %class %L %M -- %msg%n" />

    </console>


    <!-- fileName: 输出路径 filePattern: 命名规则 -->

    <RollingFile name="RollingFileDebug"

        fileName="D:/logs/debug.log"

        filePattern="D:/logs/$${date:yyyy-MM-dd}/debug-%d{yyyy-MM-dd}-%i.log">

        <Filters>

            <ThresholdFilter level="DEBUG" />

            <ThresholdFilter level="INFO" onMatch="DENY"

                onMismatch="NEUTRAL" />

        </Filters>

        <!-- 输出格式 -->
```

```
        <PatternLayout
             pattern="%d{HH:mm:ss.SSS} [%t] %-5level %class{36} %L %M - %msg%n" />
        <Policies>
        <!-- 单个日志文件的大小限制 -->
        <SizeBasedTriggeringPolicy size="100 MB" />
        </Policies>
        <!-- 最多保留 20 个日志文件 -->
        <DefaultRolloverStrategy max="20" />
    </RollingFile>

    <RollingFile name="RollingFileInfo"
        fileName="D:/logs/info.log"
        filePattern="D:/logs/$${date:yyyy-MM-dd}/info-%d{yyyy-MM-dd}-%i.log">
        <Filters>
             <ThresholdFilter level="INFO" />
             <ThresholdFilter level="WARN" onMatch="DENY"
                  onMismatch="NEUTRAL" />
        </Filters>
        <!-- 输出格式 -->
        <PatternLayout
             pattern="%d{HH:mm:ss.SSS} %-5level %class{36} %L %M - %msg%xEx%n" />
        <Policies>
             <!-- SizeBasedTriggeringPolicy 单个文件的大小限制 -->
             <SizeBasedTriggeringPolicy size="100 MB" />
        </Policies>
        <!-- DefaultRolloverStrategy 同一个文件下的最大文件数 -->
        <DefaultRolloverStrategy max="20" />
    </RollingFile>
```

```
<RollingFile name="RollingFileWarn"
     fileName="D:/logs/warn.log"
     filePattern="D:/logs/$${date:yyyy-MM}/warn-%d{yyyy-MM-dd}-%i.log">
     <Filters>
          <ThresholdFilter level="WARN" />
          <ThresholdFilter level="ERROR" onMatch="DENY"
               onMismatch="NEUTRAL" />
     </Filters>
     <PatternLayout
          pattern="[%d{HH:mm:ss:SSS}] [%p] - %l - %m%n" />
     <Policies>
          <!--<TimeBasedTriggeringPolicy modulate="true" interval="1"/> -->
          <SizeBasedTriggeringPolicy size="100 MB" />
     </Policies>
     <!--最多保留 20 个日志文件 -->
     <DefaultRolloverStrategy max="20" min="0" />
</RollingFile>

<RollingFile name="RollingFileError"
     fileName="D:/logs/error.log"
     filePattern="D:/logs/$${date:yyyy-MM}/error-%d{yyyy-MM-dd}-%i.log">
     <Filters>
          <ThresholdFilter level="ERROR" />
          <ThresholdFilter level="FATAL" onMatch="DENY"
               onMismatch="NEUTRAL" />
     </Filters>
     <PatternLayout
          pattern="[%d{HH:mm:ss:SSS}] [%p] - %l - %m%n" />
     <Policies>
```

```
            <!--<TimeBasedTriggeringPolicy modulate="true" interval="1"/> -->
            <SizeBasedTriggeringPolicy size="100 MB" />
        </Policies>
        <!--最多保留 20 个日志文件 -->
        <DefaultRolloverStrategy max="20" min="0" />
    </RollingFile>
</appenders>

<loggers>
    <root level="debug">
        <appender-ref ref="Console"/>
        <appender-ref ref="RollingFileDebug"/>
        <appender-ref ref="RollingFileInfo"/>
        <appender-ref ref="RollingFileWarn"/>
        <appender-ref ref="RollingFileError"/>
    </root>

    <logger name="org.springframework" level="error"></logger>
    <logger name="org.mybatis.spring" level="error"></logger>
    <logger name="org.apache.ibatis" level="error"></logger>

    <!--过滤掉 spring 和 mybatis 的一些无用的 debug 信息 -->
    <!-- <AsyncLogger name="org.springframework" level="error" includeLocation="true">
        <AppenderRef ref="RollingFileError"></AppenderRef>
    </AsyncLogger>
    <AsyncLogger name="org.mybatis" level="error" includeLocation="true">
        <AppenderRef ref="RollingFileError"></AppenderRef>
    </AsyncLogger>
    <AsyncLogger name="com.alibaba.druid" level="error"  includeLocation="true">
```

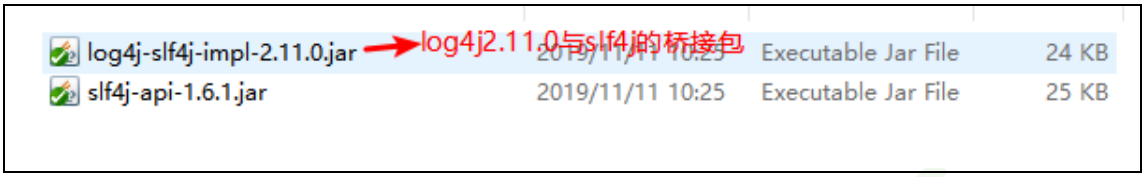
```
        <AppenderRef ref="RollingFileError"></AppenderRef>
    </AsyncLogger>
    <AsyncLogger name="org.apache.ibatis" level="error"  includeLocation="true">
        <AppenderRef ref="RollingFileError"></AppenderRef>
    </AsyncLogger>

    <AsyncRoot level="debug" includeLocation="true">
        <appender-ref ref="Console"/>
        <appender-ref ref="RollingFileDebug"/>
        <appender-ref ref="RollingFileWarn"/>
          <appender-ref ref="RollingFileError"/>
          <appender-ref ref="RollingFileFatal"/>
    </AsyncRoot>    -->

  </loggers>
</configuration>
```

3.3. Log4j2 的使用

```
private Logger l = LogManager.getLogger(SysuserHandler.class);
l.debug("---debug 信息--");
l.info("---info 信息--");
l.warn("---warn 信息--");
l.error("---error 信息--");
```

4. Slf4j 使用

4.1. Slf4j 的 jar

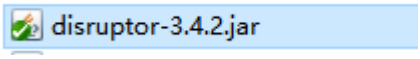


4.2. Slf4j 的使用

```
private  Logger l = LoggerFactory.getLogger(SysuserHandler.class);

l.debug("---debug 信息--");

l.info("---info 信息--");

l.warn("---warn 信息--");

l.error("---error 信息--");
```

5. Log4j2 的异步日志

5.1. 异步组件 jar



5.2. Xml 配置

```
<AsyncLogger name="org.springframework" level="error" includeLocation="true">
          <AppenderRef ref="RollingFileError"></AppenderRef>
     </AsyncLogger>
     <AsyncLogger name="org.mybatis" level="error" includeLocation="true">
          <AppenderRef ref="RollingFileError"></AppenderRef>
     </AsyncLogger>
     <AsyncLogger name="com.alibaba.druid" level="error"  includeLocation="true">
          <AppenderRef ref="RollingFileError"></AppenderRef>
     </AsyncLogger>
     <AsyncLogger name="org.apache.ibatis" level="error"  includeLocation="true">
          <AppenderRef ref="RollingFileError"></AppenderRef>
     </AsyncLogger>

     <AsyncRoot level="debug" includeLocation="true">
          <appender-ref ref="Console"/>
          <appender-ref ref="RollingFileDebug"/>
          <appender-ref ref="RollingFileWarn"/>
            <appender-ref ref="RollingFileError"/>
            <appender-ref ref="RollingFileFatal"/>
     </AsyncRoot>
```

6. 其他

PatternLayout 格式化符号说明：

%p 或 %level：输出日志信息的优先级，即 DEBUG，INFO，WARN，ERROR，FATAL。

%d：输出日志时间点的日期或时间，默认格式为 ISO8601，也可以在其后指定格式，如：%d{yyyy/MM/dd HH:mm:ss,SSS}。

%r：输出自应用程序启动到输出该 log 信息耗费的毫秒数。

%t：输出产生该日志事件的线程名。

%class：输出日志信息所属的类目，通常就是所在类的全名。

%M：输出产生日志信息的方法名。

%F：输出日志消息产生时所在的文件名称。

%L：输出代码中的行号。

%m 或%msg 或%message:：输出代码中指定的具体日志信息。

%n：输出一个回车换行符，Windows 平台为“rn”，Unix 平台为“n”。

%x：输出和当前线程相关联的 NDC(嵌套诊断环境)，尤其用到像 java servlets 这样的多客户多线程的应用中。

%%：输出一个“%”字符。

另外，还可以在%与格式字符之间加上修饰符来控制其最小长度、最大长度、和文本的对齐方式。如：

1)%-20：“-”号表示左对齐,不满足 20 个字符则以空格代替。

2)%.30：指定输出 category 的名称，最大的长度是 30，如果 category 的名称长度大于 30 的话，就会将左边多出的字符截掉，但小于 30 的话也不会补空格。