# log4j2

2021年6月8日 13:23

**依赖**

```
<dependency>
        <groupId>org.apache.logging.log4j</groupId>
        <artifactId>log4j-api</artifactId>
        <version>2.11.1</version>
    </dependency>
    <dependency>
        <groupId>org.apache.logging.log4j</groupId>
        <artifactId>log4j-core</artifactId>
        <version>2.11.1</version>
    </dependency>
```

**简单样例**

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--
   status="warn" 日志框架本身的输出日志级别，可以修改为debug
   monitorInterval="5" 自动加载配置文件的间隔时间，不低于 5秒；生产环境中修改配置文件，是热更新，无需重启应用
 -->
<configuration status="warn" monitorInterval="5">
   <!--
      集中配置属性进行管理
      使用时通过:${name}
   -->
   <properties>
      <property name="LOG_HOME">./logs</property>
   </properties>

   <!-- 日志处理 -->
   <Appenders>
      <!-- 控制台输出 appender，SYSTEM_OUT输出黑色，SYSTEM_ERR输出红色 -->
      <Console name="Console" target="SYSTEM_OUT">
         <PatternLayout pattern="%d{HH:mm:ss.SSS} [%t] [%-5level] %c{36}:%L --- %m%n" />
      </Console>
```

```
        <!-- 日志文件输出 appender -->
        <File name="file" fileName="${LOG_HOME}/myfile.log">
            <PatternLayout pattern="[%d{yyyy-MM-dd HH:mm:ss.SSS}] [%-5level] %l %c{36} - %m%n" />
        </File>

        <!-- 使用随机读写流的日志文件输出 appender，性能提高 -->
        <RandomAccessFile name="accessFile" fileName="${LOG_HOME}/myAcclog.log">
            <PatternLayout pattern="[%d{yyyy-MM-dd HH:mm:ss.SSS}] [%-5level] %l %c{36} - %m%n" />
        </RandomAccessFile>

        <!-- 按照一定规则拆分的日志文件的appender -->  <!-- 拆分后的文件 -->
        <!-- filePattern="${LOG_HOME}/$${date:yyyy-MM-dd}/myrollog-%d{yyyy-MM-dd-HH-mm}-%
i.log"> -->
        <RollingFile name="rollingFile" fileName="${LOG_HOME}/myrollog.log"
                     filePattern="${LOG_HOME}/$${date:yyyy-MM-dd}/myrollog-%d{yyyy-MM-dd-HH-
mm}-%i.log">
            <!-- 日志级别过滤器 -->
            <ThresholdFilter level="debug" onMatch="ACCEPT" onMismatch="DENY" />
            <!-- 日志消息格式 -->
            <PatternLayout pattern="[%d{yyyy-MM-dd HH:mm:ss.SSS}] [%-5level] %l %c{36} - %msg%n"
/>
            <Policies>
                <!-- 在系统启动时，出发拆分规则，生产一个新的日志文件 -->
                <OnStartupTriggeringPolicy />
                <!-- 按照文件大小拆分，10MB -->
                <SizeBasedTriggeringPolicy size="10 MB" />
                <!-- 按照时间节点拆分，规则根据filePattern定义的 -->
                <TimeBasedTriggeringPolicy />
            </Policies>
            <!-- 在同一个目录下，文件的个限定为 30个，超过进行覆盖 -->
            <DefaultRolloverStrategy max="30" />
        </RollingFile>
    </Appenders>

    <!-- logger 定义 -->
    <Loggers>
        <!-- 使用 rootLogger 配置 日志级别 level="trace" -->
        <Root level="trace">
            <!-- 指定日志使用的处理器 -->
            <!-- <AppenderRef ref="Console" />-->
            <AppenderRef ref="Console" />
        </Root>
```

```
    </Loggers>
</configuration>
```

**异步日志**

1. **AsyncAppender【生产上几乎不使用，因为性能低下】**

```
<?xml version="1.0" encoding="UTF-8" ?>
<configuration status="warn" monitorInterval="5">

    <properties>
        <property name="LOG_HOME">./logs</property>
    </properties>

    <Appenders>
        <!-- 日志文件输出 appender -->
        <File name="file" fileName="${LOG_HOME}/myfile.log">
            <!--<PatternLayout pattern="[%d{yyyy-MM-dd HH:mm:ss.SSS}] [%-5level] %l %c{36} - %m%n"
/>-->
            <PatternLayout pattern="%d %p %c{1.} [%t] %m%n" />
        </File>

        <Async name="Async">
            <AppenderRef ref="file" />
        </Async>
    </Appenders>

    <Loggers>
        <Root level="trace">
            <AppenderRef ref="Async" />
        </Root>
    </Loggers>

</configuration>
```

2. **AsyncLogger 【生产上用得多，因为性能高】**

添加额外依赖

```
<!-- 异步日志依赖 -->
        <dependency>
            <groupId>com.lmax</groupId>
            <artifactId>disruptor</artifactId>
```

```
        <version>3.3.4</version>
    </dependency>
```

需要添加一个配置文件log4j2.component.properties，内容如下
Log4jContextSelector=org.apache.logging.log4j.core.async.AsyncLoggerContextSelector

```
<!-- 自定义 logger 对象
        includeLocation="false" 关闭日志记录的行号信息，开启的话会严重影响异步输出的性能
        additivity="false" 不再继承 rootlogger对象
     -->
    <AsyncLogger name="com.log" level="trace" includeLocation="false" additivity="false">
        <AppenderRef ref="Async" />
    </AsyncLogger>
```

**完整用例**

```
<?xml version="1.0" encoding="UTF-8"?>
<!--日志级别以及优先级排序: OFF > FATAL > ERROR > WARN > INFO > DEBUG > TRACE > ALL -->
<!--Configuration后面的status，这个用于设置log4j2自身内部的信息输出，可以不设置，当设置成trace时，你会看到log4j2内部各种详细输出-->
<!--monitorInterval：Log4j能够自动检测修改配置 文件和重新配置本身，设置间隔秒数-->
<configuration status="WARN" monitorInterval="30">
    <!--先定义所有的appender-->
    <appenders>
    <!--这个输出控制台的配置-->
        <console name="Console" target="SYSTEM_OUT">
        <!--输出日志的格式-->
            <PatternLayout pattern="[%d{HH:mm:ss:SSS}] [%p] - %l - %m%n"/>
        </console>
    <!--文件会打印出所有信息，这个log每次运行程序会自动清空，由append属性决定，这个也挺有用的，适合临时测试用-->
    <File name="log" fileName="log/test.log" append="false">
      <PatternLayout pattern="%d{HH:mm:ss.SSS} %-5level %class{36} %L %M - %msg%xEx%n"/>
    </File>
    <!-- 这个会打印出所有的info及以下级别的信息，每次大小超过size，则这size大小的日志会自动存入按年份-月份建立的文件夹下面并进行压缩，作为存档-->
        <RollingFile name="RollingFileInfo" fileName="${sys:user.home}/logs/info.log"
                     filePattern="${sys:user.home}/logs/$${date:yyyy-MM}/info-%d{yyyy-MM-dd}-%i.log">
            <!--控制台只输出level及以上级别的信息（onMatch），其他的直接拒绝（onMismatch）-->
            <ThresholdFilter level="info" onMatch="ACCEPT" onMismatch="DENY"/>
            <PatternLayout pattern="[%d{HH:mm:ss:SSS}] [%p] - %l - %m%n"/>
            <Policies>
```

```
            <TimeBasedTriggeringPolicy/>
            <SizeBasedTriggeringPolicy size="100 MB"/>
        </Policies>
    </RollingFile>
    <RollingFile name="RollingFileWarn" fileName="${sys:user.home}/logs/warn.log"
                 filePattern="${sys:user.home}/logs/$${date:yyyy-MM}/warn-%d{yyyy-MM-dd}-%i.log">
        <ThresholdFilter level="warn" onMatch="ACCEPT" onMismatch="DENY"/>
        <PatternLayout pattern="[%d{HH:mm:ss:SSS}] [%p] - %l - %m%n"/>
        <Policies>
            <TimeBasedTriggeringPolicy/>
            <SizeBasedTriggeringPolicy size="100 MB"/>
        </Policies>
    <!-- DefaultRolloverStrategy属性如不设置，则默认为最多同一文件夹下7个文件，这里设置了20 -->
        <DefaultRolloverStrategy max="20"/>
    </RollingFile>
    <RollingFile name="RollingFileError" fileName="${sys:user.home}/logs/error.log"
                 filePattern="${sys:user.home}/logs/$${date:yyyy-MM}/error-%d{yyyy-MM-dd}-%i.log">
        <ThresholdFilter level="error" onMatch="ACCEPT" onMismatch="DENY"/>
        <PatternLayout pattern="[%d{HH:mm:ss:SSS}] [%p] - %l - %m%n"/>
        <Policies>
            <TimeBasedTriggeringPolicy/>
            <SizeBasedTriggeringPolicy size="100 MB"/>
        </Policies>
    </RollingFile>
  </appenders>
  <!--然后定义logger，只有定义了logger并引入的appender，appender才会生效-->
  <loggers>
    <!--过滤掉spring和mybatis的一些无用的DEBUG信息-->
    <logger name="org.springframework" level="INFO"></logger>
    <logger name="org.mybatis" level="INFO"></logger>
    <root level="all">
        <appender-ref ref="Console"/>
        <appender-ref ref="RollingFileInfo"/>
        <appender-ref ref="RollingFileWarn"/>
        <appender-ref ref="RollingFileError"/>
    </root>
  </loggers>
</configuration>
```

**参考资料**:

https://blog.csdn.net/weixin_32265569/article/details/110723441

https://www.cnblogs.com/hafiz/p/6170702.html