参考文档

<https://blog.51cto.com/1197822/2157668>

**<?**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="1800">

<Properties>

*<!-- ==============================================公共配置============================================== -->*

*<!-- 设置日志文件的目录名称 -->*

<property name="logFileName">qfxLog4jDemoLog</property>

*<!-- 日志默认存放的位置,可以设置为项目根路径下,也可指定绝对路径 -->*

*<!-- 存放路径一:通用路径,window平台 -->*

*<!-- <property name="basePath">d:/logs/${logFileName}</property> -->*

*<!-- 存放路径二:web工程专用,java项目没有这个变量,需要删掉,否则会报异常,这里把日志放在web项目的根目录下 -->*

*<!-- <property name="basePath">${web:rootDir}/${logFileName}</property> -->*

*<!-- 存放路径三:web工程专用,java项目没有这个变量,需要删掉,否则会报异常,这里把日志放在tocmat的logs目录下 -->*

<property name="basePath">${sys:catalina.home}/logs/${logFileName}</property>

*<!-- 控制台默认输出格式,"%-5level":日志级别,"%l":输出完整的错误位置,是小写的L,因为有行号显示,所以影响日志输出的性能 -->*

<property name="console\_log\_pattern">%d{yyyy-MM-dd HH:mm:ss.SSS} [%-5level] %l - %m%n</property>

*<!-- 日志文件默认输出格式,不带行号输出(行号显示会影响日志输出性能);%C:大写,类名;%M:方法名;%m:错误信息;%n:换行 -->*

*<!-- <property name="log\_pattern">%d{yyyy-MM-dd HH:mm:ss.SSS} [%-5level] %C.%M - %m%n</property> -->*

*<!-- 日志文件默认输出格式,另类带行号输出(对日志输出性能未知);%C:大写,类名;%M:方法名;%L:行号;%m:错误信息;%n:换行 -->*

<property name="log\_pattern">%d{yyyy-MM-dd HH:mm:ss.SSS} [%-5level] %C.%M[%L line] - %m%n</property>

*<!-- 日志默认切割的最小单位 -->*

<property name="every\_file\_size">20MB</property>

*<!-- 日志默认输出级别 -->*

<property name="output\_log\_level">DEBUG</property>

*<!-- ===========================================所有级别日志配置=========================================== -->*

*<!-- 日志默认存放路径(所有级别日志) -->*

<property name="rolling\_fileName">${basePath}/all.log</property>

*<!-- 日志默认压缩路径,将超过指定文件大小的日志,自动存入按"年月"建立的文件夹下面并进行压缩,作为存档 -->*

<property name="rolling\_filePattern">${basePath}/%d{yyyy-MM}/all-%d{yyyy-MM-dd-HH}-%i.log.gz</property>

*<!-- 日志默认同类型日志,同一文件夹下可以存放的数量,不设置此属性则默认为7个,filePattern最后要带%i才会生效 -->*

<property name="rolling\_max">500</property>

*<!-- 日志默认同类型日志,多久生成一个新的日志文件,这个配置需要和filePattern结合使用;*

*如果设置为1,filePattern是%d{yyyy-MM-dd}到天的格式,则间隔一天生成一个文件*

*如果设置为12,filePattern是%d{yyyy-MM-dd-HH}到小时的格式,则间隔12小时生成一个文件 -->*

<property name="rolling\_timeInterval">12</property>

*<!-- 日志默认同类型日志,是否对封存时间进行调制,若为true,则封存时间将以0点为边界进行调整,*

*如:现在是早上3am,interval是4,那么第一次滚动是在4am,接着是8am,12am...而不是7am -->*

<property name="rolling\_timeModulate">true</property>

*<!-- ============================================Info级别日志============================================ -->*

*<!-- Info日志默认存放路径(Info级别日志) -->*

<property name="info\_fileName">${basePath}/info.log</property>

*<!-- Info日志默认压缩路径,将超过指定文件大小的日志,自动存入按"年月"建立的文件夹下面并进行压缩,作为存档 -->*

<property name="info\_filePattern">${basePath}/%d{yyyy-MM}/info-%d{yyyy-MM-dd}-%i.log.gz</property>

*<!-- Info日志默认同一文件夹下可以存放的数量,不设置此属性则默认为7个 -->*

<property name="info\_max">100</property>

*<!-- 日志默认同类型日志,多久生成一个新的日志文件,这个配置需要和filePattern结合使用;*

*如果设置为1,filePattern是%d{yyyy-MM-dd}到天的格式,则间隔一天生成一个文件*

*如果设置为12,filePattern是%d{yyyy-MM-dd-HH}到小时的格式,则间隔12小时生成一个文件 -->*

<property name="info\_timeInterval">1</property>

*<!-- 日志默认同类型日志,是否对封存时间进行调制,若为true,则封存时间将以0点为边界进行调整,*

*如:现在是早上3am,interval是4,那么第一次滚动是在4am,接着是8am,12am...而不是7am -->*

<property name="info\_timeModulate">true</property>

*<!-- ============================================Warn级别日志============================================ -->*

*<!-- Warn日志默认存放路径(Warn级别日志) -->*

<property name="warn\_fileName">${basePath}/warn.log</property>

*<!-- Warn日志默认压缩路径,将超过指定文件大小的日志,自动存入按"年月"建立的文件夹下面并进行压缩,作为存档 -->*

<property name="warn\_filePattern">${basePath}/%d{yyyy-MM}/warn-%d{yyyy-MM-dd}-%i.log.gz</property>

*<!-- Warn日志默认同一文件夹下可以存放的数量,不设置此属性则默认为7个 -->*

<property name="warn\_max">100</property>

*<!-- 日志默认同类型日志,多久生成一个新的日志文件,这个配置需要和filePattern结合使用;*

*如果设置为1,filePattern是%d{yyyy-MM-dd}到天的格式,则间隔一天生成一个文件*

*如果设置为12,filePattern是%d{yyyy-MM-dd-HH}到小时的格式,则间隔12小时生成一个文件 -->*

<property name="warn\_timeInterval">1</property>

*<!-- 日志默认同类型日志,是否对封存时间进行调制,若为true,则封存时间将以0点为边界进行调整,*

*如:现在是早上3am,interval是4,那么第一次滚动是在4am,接着是8am,12am...而不是7am -->*

<property name="warn\_timeModulate">true</property>

*<!-- ============================================Error级别日志============================================ -->*

*<!-- Error日志默认存放路径(Error级别日志) -->*

<property name="error\_fileName">${basePath}/error.log</property>

*<!-- Error日志默认压缩路径,将超过指定文件大小的日志,自动存入按"年月"建立的文件夹下面并进行压缩,作为存档 -->*

<property name="error\_filePattern">${basePath}/%d{yyyy-MM}/error-%d{yyyy-MM-dd}-%i.log.gz</property>

*<!-- Error日志默认同一文件夹下可以存放的数量,不设置此属性则默认为7个 -->*

<property name="error\_max">100</property>

*<!-- 日志默认同类型日志,多久生成一个新的日志文件,这个配置需要和filePattern结合使用;*

*如果设置为1,filePattern是%d{yyyy-MM-dd}到天的格式,则间隔一天生成一个文件*

*如果设置为12,filePattern是%d{yyyy-MM-dd-HH}到小时的格式,则间隔12小时生成一个文件 -->*

<property name="error\_timeInterval">1</property>

*<!-- 日志默认同类型日志,是否对封存时间进行调制,若为true,则封存时间将以0点为边界进行调整,*

*如:现在是早上3am,interval是4,那么第一次滚动是在4am,接着是8am,12am...而不是7am -->*

<property name="error\_timeModulate">true</property>

*<!-- ============================================控制台显示控制============================================ -->*

*<!-- 控制台显示的日志最低级别 -->*

<property name="console\_print\_level">DEBUG</property>

</Properties>

*<!--定义appender -->*

<appenders>

*<!-- =======================================用来定义输出到控制台的配置======================================= -->*

<Console name="Console" target="SYSTEM\_OUT">

*<!-- 设置控制台只输出level及以上级别的信息(onMatch),其他的直接拒绝(onMismatch)-->*

<ThresholdFilter level="${console\_print\_level}" onMatch="ACCEPT" onMismatch="DENY"/>

*<!-- 设置输出格式,不设置默认为:%m%n -->*

<PatternLayout pattern="${console\_log\_pattern}"/>

</Console>

*<!-- ================================打印root中指定的level级别以上的日志到文件================================ -->*

<RollingFile name="RollingFile" fileName="${rolling\_fileName}" filePattern="${rolling\_filePattern}">

<PatternLayout pattern="${log\_pattern}"/>

<Policies>

<TimeBasedTriggeringPolicy interval="${rolling\_timeInterval}" modulate="${warn\_timeModulate}"/>

<SizeBasedTriggeringPolicy size="${every\_file\_size}"/>

</Policies>

*<!-- 设置同类型日志,同一文件夹下可以存放的数量,如果不设置此属性则默认存放7个文件 -->*

<DefaultRolloverStrategy max="${rolling\_max}" />

</RollingFile>

*<!-- =======================================打印INFO级别的日志到文件======================================= -->*

<RollingFile name="InfoFile" fileName="${info\_fileName}" filePattern="${info\_filePattern}">

<PatternLayout pattern="${log\_pattern}"/>

<Policies>

<TimeBasedTriggeringPolicy interval="${info\_timeInterval}" modulate="${info\_timeModulate}"/>

<SizeBasedTriggeringPolicy size="${every\_file\_size}"/>

</Policies>

<DefaultRolloverStrategy max="${info\_max}" />

<Filters>

<ThresholdFilter level="WARN" onMatch="DENY" onMismatch="NEUTRAL"/>

<ThresholdFilter level="INFO" onMatch="ACCEPT" onMismatch="DENY"/>

</Filters>

</RollingFile>

*<!-- =======================================打印WARN级别的日志到文件======================================= -->*

<RollingFile name="WarnFile" fileName="${warn\_fileName}" filePattern="${warn\_filePattern}">

<PatternLayout pattern="${log\_pattern}"/>

<Policies>

<TimeBasedTriggeringPolicy interval="${warn\_timeInterval}" modulate="${warn\_timeModulate}"/>

<SizeBasedTriggeringPolicy size="${every\_file\_size}"/>

</Policies>

<DefaultRolloverStrategy max="${warn\_max}" />

<Filters>

<ThresholdFilter level="ERROR" onMatch="DENY" onMismatch="NEUTRAL"/>

<ThresholdFilter level="WARN" onMatch="ACCEPT" onMismatch="DENY"/>

</Filters>

</RollingFile>

*<!-- =======================================打印ERROR级别的日志到文件======================================= -->*

<RollingFile name="ErrorFile" fileName="${error\_fileName}" filePattern="${error\_filePattern}">

<PatternLayout pattern="${log\_pattern}"/>

<Policies>

<TimeBasedTriggeringPolicy interval="${error\_timeInterval}" modulate="${error\_timeModulate}"/>

<SizeBasedTriggeringPolicy size="${every\_file\_size}"/>

</Policies>

<DefaultRolloverStrategy max="${error\_max}" />

<Filters>

<ThresholdFilter level="FATAL" onMatch="DENY" onMismatch="NEUTRAL"/>

<ThresholdFilter level="ERROR" onMatch="ACCEPT" onMismatch="DENY"/>

</Filters>

</RollingFile>

</appenders>

*<!--定义logger,只有定义了logger并引入的appender,appender才会生效-->*

<loggers>

*<!-- 设置打印sql语句配置开始,以下两者配合使用,可以优化日志的输出信息,减少一些不必要信息的输出 -->*

*<!-- 设置java.sql包下的日志只打印DEBUG及以上级别的日志,此设置可以支持sql语句的日志打印 -->*

<logger name="java.sql" level="DEBUG" additivity="false">

<appender-ref ref="Console"/>

</logger>

*<!-- 设置org.mybatis.spring包下的日志只打印WARN及以上级别的日志 -->*

<logger name="org.mybatis.spring" level="WARN" additivity="false">

<appender-ref ref="Console"/>

</logger>

*<!-- 设置org.mybatis.spring包下的日志只打印WARN及以上级别的日志 -->*

<logger name="org.springframework" level="WARN" additivity="false">

<appender-ref ref="Console"/>

</logger>

*<!-- 设置org.mybatis.spring包下的日志只打印WARN及以上级别的日志 -->*

<logger name="com.qfx.workflow.service" level="WARN" additivity="false">

<appender-ref ref="Console"/>

</logger>

*<!-- 设置打印sql语句配置结束 -->*

*<!--建立一个默认的root的logger-->*

<root level="${output\_log\_level}">

<appender-ref ref="RollingFile"/>

<appender-ref ref="Console"/>

<appender-ref ref="InfoFile"/>

<appender-ref ref="WarnFile"/>

<appender-ref ref="ErrorFile"/>

</root>

</loggers>

</configuration>

**web.xml文件中配置log4j2的信息**

*<!-- log4j2设置 ,此处配置适用于Servlet的3.0及以上版本使用,3.0以下版本另有配置 -->*

*<!-- 如果log4j2.xml放在classpath目录下(即class目录),纯web项目中,不需要log4j2的任何配置也能启用日志 -->*

*<!-- 如果不是放在classpath目录下,则需要指定配置文件路径 -->*

*<!-- 如果放在classpath下其他的目录中,即使指定路径也无法找到log4j2.xml -->*

<context-param>

<param-name>log4jConfigLocation</param-name>

<param-value>classpath:log4j2.xml</param-value>

*<!-- 配置log4j2.xml文件路径在/WEB-INF/目录下,放置在这里可以被加载到文件 -->*

*<!-- <param-value>/WEB-INF/log4j2.xml</param-value> -->*

</context-param>

<listener>

<listener-class>org.apache.logging.log4j.web.Log4jServletContextListener</listener-class>

</listener>