Log4j.xml配置日志按级别过滤并将指定级别的日志发送到ActiveMQ

[收藏](javascript:void(0);)

[摆渡者](https://my.oschina.net/itblog/home)

* 发表于 1年前

* 阅读 1870

* 收藏 16

* 点赞 1

* [评论 0](https://my.oschina.net/itblog/blog/536505#comment-list)

[**免费在线直播教学： java    web前端    c++   python   ios！>>>**](http://web.tanzhouedu.com/index/osc.html)  https://my.oschina.net/img/hot3.png

*摘要: 本文介绍使用log4j.xml来配置日志按级别过滤，并将指定的日志输出到ActiveMQ中。*

在之前的一篇博客《[Spring+Log4j+ActiveMQ实现远程记录日志——实战+分析](http://my.oschina.net/itblog/blog/533730)》的评论中，有同学提到这种方式应该要能根据日志级别设置来决定是否发送到mq，不然会大量占用网络资源。于是经过了一番搜索后，实现了这个功能。现在记录在这里。

目标：将debug，info级别的日志输出到本地文件，将warn，error级别的日志输出到ActiveMQ。

说明：本文还是使用之前的两个项目：Product和Logging。

经过一番搜索后，发现log4j还可以按照级别过滤日志，但**过滤只能使用log4j.xml配置**：

Filters can be defined at appender level. For example, to filter only certain levels, the LevelRangeFilter can be used like this:

<appender name="TRACE" class="org.apache.log4j.ConsoleAppender">

    <layout class="org.apache.log4j.PatternLayout">

        <param name="ConversionPattern" value="[%t] %-5p %c - %m%n" />

    </layout>

    <filter class="org.apache.log4j.varia.LevelRangeFilter">

        <!-- 注意这个min和max是相等的 -->

        <param name="levelMin" value="DEBUG" />

        <param name="levelMax" value="DEBUG" />

    </filter>

</appender>

在搜索资料的过程中，也看到了有网友说log4j.properties方式也可以实现按级别过滤日志，具体步骤请参看《[Log4j按级别输出日志到不同文件配置分析](http://blog.csdn.net/projava/article/details/7344839)》。此种方式的缺点是，如果有多个Appender，则需要多个继承的类（每个Appender需要重新定义一个），因此感觉不如log4j.xml方式通过为appender配置filter来的直接。

然后我在这里找到了一份log4j.xml样本：<https://code.google.com/p/log4j-jms-sample/source/browse/trunk/src/main/resources/log4j.xml>，拿过来后，**我只是在jms appender里面增加了一个filter而已**：

<appender name="jms" class="org.apache.log4j.net.JMSAppender">

    <param name="InitialContextFactoryName"

        value="org.apache.activemq.jndi.ActiveMQInitialContextFactory" />

    <param name="ProviderURL" value="tcp://localhost:61616" />

    <param name="TopicBindingName" value="logTopic" />

    <param name="TopicConnectionFactoryBindingName" value="ConnectionFactory" />

    <!-- Only log WARN & ERROR msg -->

    <filter class="org.apache.log4j.varia.LevelRangeFilter">

        <param name="LevelMin" value="WARN" />

        <param name="LevelMax" value="ERROR" />

    </filter>

</appender>

而Product项目的测试代码相当简单：

package com.demo.product;

import org.apache.log4j.Logger;

public class Main{

    public static void main(String[] args) throws Exception {

        Logger logger = Logger.getLogger(Main.class);

        logger.debug("Debug");

        logger.info("Info");

        logger.warn("Warn");

        logger.error("Error");

        logger.fatal("Fatal");

        System.exit(0);

    }

}

我以为就这样配置就能让WARN和ERROR级别的日志输出到jms了，但是我运行的时候却报错了：

javax.jms.JMSException: Wire format negotiation timeout: peer did not send his wire format.

at org.apache.activemq.util.JMSExceptionSupport.create(JMSExceptionSupport.java:62)

at org.apache.activemq.ActiveMQConnection.syncSendPacket(ActiveMQConnection.java:1395)

at org.apache.activemq.ActiveMQConnection.ensureConnectionInfoSent(ActiveMQConnection.java:1481)

at org.apache.activemq.ActiveMQConnection.createSession(ActiveMQConnection.java:323)

at org.apache.activemq.ActiveMQConnection.createTopicSession(ActiveMQConnection.java:1112)

搜索这个问题，有很多人遇到了，[这里](http://activemq.apache.org/javaxjmsjmsexception-wire-format-negociation-timeout-peer-did-not-send-his-wire-format.html)列出了三种可能的原因：

**1. You're connecting to the port not used by ActiveMQ TCP transport**

Make sure to check that you're connecting to the appropriate host:port

**2. You're using log4j JMS appender and doesn't filter out ActiveMQ log messages**

Be sure to read [*How do I use log4j JMS appender with ActiveMQ*](http://activemq.apache.org/how-do-i-use-log4j-jms-appender-with-activemq.html) and more importantly to never send ActiveMQ log messages to JMS appender

**3. Your broker is probably under heavy load (or network connection is unreliable), so connection setup cannot be completed in a reasonable time**

If you experience sporadic exceptions like this, the best solution is to use failover transport, so that your clients can try connecting again if the first attempt fails. If you're getting these kind of exceptions more frequently you can also try extending wire format negotiation period (default 10 sec). You can do that by using wireFormat.maxInactivityDurationInitalDelay property on the connection URL in your client. For example:

tcp://localhost:61616?wireFormat.maxInactivityDurationInitalDelay=30000

第一种情况显然不是。

第三种情况，由于我就一个jms connection，也没有往这个连接发送jms消息，所以不可能负载过重。

第二种情况是不要把activemq的日志发送到JMSAppender了，[*How do I use log4j JMS appender with ActiveMQ*](http://activemq.apache.org/how-do-i-use-log4j-jms-appender-with-activemq.html)  一文中有以下配置：

## Be sure that ActiveMQ messages are not logged to 'jms' appender

log4j.logger.org.apache.activemq=INFO, stdout

 上面的意思是，对于org.apache.activemq包下的INFO级别以上的日志，都输出到stdout appender中。我对比了一下，从拷贝而来的log4j.xml中也包含了类似的配置：

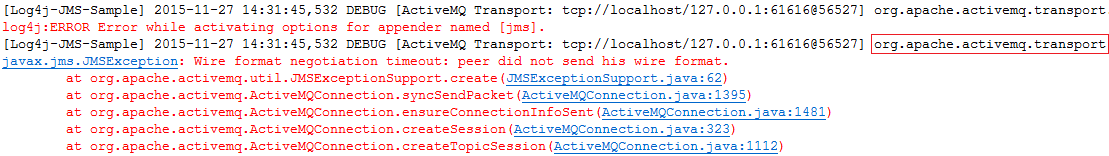
<!-- Be sure that ActiveMQ messages are not logged to 'jms' appender -->

<logger name="org.apache.activemq">

    <appender-ref ref="console" />

</logger>

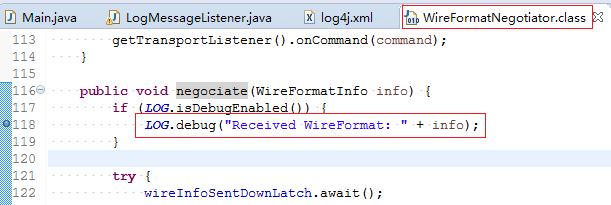
但是为何结果还是这样？几经思考，我重新查看了一下报错的日志：



后面的内容是这样的：

org.apache.activemq.transport.WireFormatNegotiator.negociate-118 | Received WireFormat ...

于是我去找到这个类，在这个negociate方法上打上断点（Maven项目的好处还包括可以自动下载jar包对应版本的源代码），开始调试，然后发现是这一句报错：



然后我想了想能不能不打印这个debug消息呢，于是我在开始的org.apache.activemq包中加上了level限制：

<logger name="org.apache.activemq">

    <level value="INFO" />

    <appender-ref ref="console" />

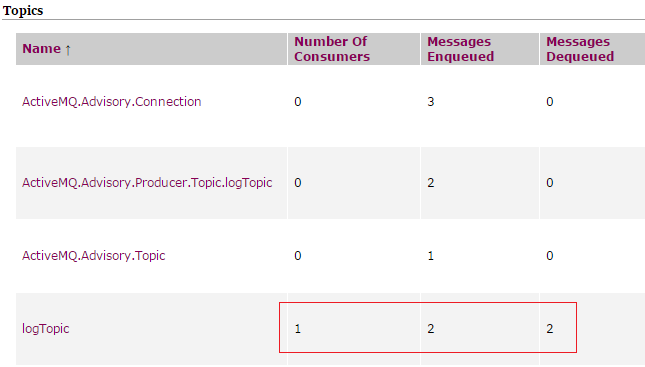
</logger>

这样以后，问题解决。其实，只要我稍微细心一点，可以发现

log4j.logger.org.apache.activemq=INFO, stdout

这个配置**不仅指明了org.apache.activemq包下的日志信息输出到stdout这个appender中，而且还指明了只有INFO以上的级别才能输出**。二者同时指定才能达到目的，这在刚刚的xml文件中也得到体现。

现在，WARN和ERROR级别的日志就可以输出到ActiveMQ了：



在Logging项目中，和之前一样，LogMessageListener也只是简单打印了级别和内容：

public void onMessage(Message message) {

    LoggingEvent event;

    try {

        event = (LoggingEvent)((ActiveMQObjectMessage)message).getObject();

        System.out.println("[" + event.getLevel() + "] | " + event.getMessage());

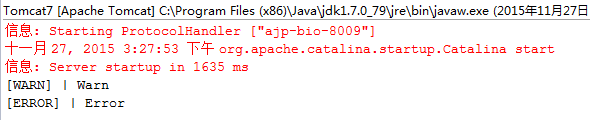
    } catch (JMSException e) {

        e.printStackTrace();

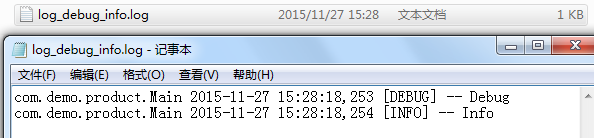
    }

}

从结果中能看到输出的日志级别仅仅包括了WARN和ERROR：



至于剩下的DEBUG和INFO级别的日志，则直接配置输出到RollingFileAppender即可。日志文件的内容也当然和预期一样了：



最后贴出完整的log4j.xml配置内容：

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

<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">

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

    <!-- Console Appender, used to record activemq log. -->

    <appender name="console" class="org.apache.log4j.ConsoleAppender">

        <param name="Target" value="System.out" />

        <layout class="org.apache.log4j.PatternLayout">

            <param name="ConversionPattern"

                value="[Log4j-JMS-Sample] %d{yyyy-MM-dd HH:mm:ss,SSS} %p [%t] %c.%M-%L | %m%n" />

        </layout>

    </appender>

    <!-- File Appender, used to record debug & info log. -->

    <appender name="file" class="org.apache.log4j.RollingFileAppender">

        <param name="File" value="C:\\logs\\log\_debug\_info.log" />

        <param name="Append" value="true" />

        <param name="MaxFileSize" value="500KB" />

        <param name="MaxBackupIndex" value="2" />

        <layout class="org.apache.log4j.PatternLayout">

            <param name="ConversionPattern"

                value="%c %d{ISO8601} [%p] -- %m%n" />

        </layout>

        <!-- Only log DEBUG & INFO msg -->

        <filter class="org.apache.log4j.varia.LevelRangeFilter">

            <param name="LevelMin" value="DEBUG" />

            <param name="LevelMax" value="INFO" />

        </filter>

    </appender>

    <!-- JMS Appender, used to record warn & info log. -->

    <appender name="jms" class="org.apache.log4j.net.JMSAppender">

        <param name="InitialContextFactoryName"

            value="org.apache.activemq.jndi.ActiveMQInitialContextFactory" />

        <param name="ProviderURL" value="tcp://localhost:61616" />

        <param name="TopicBindingName" value="logTopic" />

        <param name="TopicConnectionFactoryBindingName" value="ConnectionFactory" />

        <!-- Only log WARN & ERROR msg -->

        <filter class="org.apache.log4j.varia.LevelRangeFilter">

            <param name="LevelMin" value="WARN" />

            <param name="LevelMax" value="ERROR" />

        </filter>

    </appender>

    <!-- Log in org.apache.activemq are logged to console. -->

    <logger name="org.apache.activemq">

        <level value="INFO" />

        <appender-ref ref="console" />

    </logger>

    <root>

        <priority value="DEBUG" />

        <appender-ref ref="console" />

        <appender-ref ref="jms" />

        <appender-ref ref="file" />

    </root>

</log4j:configuration>

当然，如果希望再把info和debug分开，可以多配置一个fileappender，让每个过滤器的LevelMax和LevelMin的值相等并为它们配置不同的文件即可。

参考：

<http://wiki.apache.org/logging-log4j/Log4jXmlFormat>

<http://wiki.apache.org/logging-log4j/JMSAppender>

<http://activemq.apache.org/javaxjmsjmsexception-wire-format-negociation-timeout-peer-did-not-send-his-wire-format.html>

<http://activemq.apache.org/how-do-i-use-log4j-jms-appender-with-activemq.html>