自：https://blog.csdn.net/zjq852533445/article/details/78320012

**log4j2打印Mybatis执行的SQL语句及SQL语句的执行时间**

2017年10月23日 16:18:12 [烟花苏柳](https://me.csdn.net/zjq852533445) 阅读数：3296 标签： [mybatis](http://so.csdn.net/so/search/s.do?q=mybatis&t=blog)[log4j2](http://so.csdn.net/so/search/s.do?q=log4j2&t=blog) 更多

个人分类： [ssm](https://blog.csdn.net/zjq852533445/article/category/7044101)[mybatis](https://blog.csdn.net/zjq852533445/article/category/7236306)[log4j2](https://blog.csdn.net/zjq852533445/article/category/7243661)

## 有关于log4j2的详细配置方式，建议大家观看[log4j2配置文件详解](http://www.cnblogs.com/hafiz/p/6170702.html)，这里不做介绍

### 一、首先如需使用log4j2，得引入maven依赖

<!-- log4j2 -->

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-api</artifactId>

<version>${log4j2.version}</version>

</dependency>

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-core</artifactId>

<version>${log4j2.version}</version>

</dependency>

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-web</artifactId>

<version>${log4j2.version}</version>

</dependency>

注：本人使用的是2.7版本的log4j2

### 二、然后在项目的classpath下编写log4j2.xml

log4j2在启动的时候会默认加载名为log4j2.xml的配置文件

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

<Configuration status="OFF" monitorInterval="1800">

<properties>

<!-- log打印到本地的路径 -->

<property name="LOG\_HOME">D:/log4j2/mybatis/genertor/logs/</property>

<property name="ERROR\_LOG\_FILE\_NAME">error</property>

</properties>

<Appenders>

<!-- 控制台打印日志 -->

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

<PatternLayout pattern="%d %-5p (%F:%L) - %m%n" />

</Console>

<!-- 日志信息输出到文件配置 -->

<RollingRandomAccessFile name="ErrorLog"

fileName="${LOG\_HOME}/${ERROR\_LOG\_FILE\_NAME}.log"

filePattern="${LOG\_HOME}/${ERROR\_LOG\_FILE\_NAME}.log.%d{yyyy-MM-dd}.gz">

<PatternLayout

pattern="%d %-5p (%F:%L) - %m%n"/>

<Policies>

<!-- TimeBasedTriggeringPolicy指定的size是1，结合起来就是1天生成一个新文件。如果filePattern改成%d{yyyy-MM-dd HH}.gz,此时最小粒度为小时，则每一个小时生成一个文件。 -->

<TimeBasedTriggeringPolicy/>

<!-- 指定当文件体积大于size指定的值时，触发Rolling -->

<SizeBasedTriggeringPolicy size="100 MB"/>

</Policies>

<!-- 指定最多保存的文件个数 -->

<DefaultRolloverStrategy max="20"/>

</RollingRandomAccessFile>

<param name="Encoding" value="UTF-8" />

</Appenders>

<Loggers>

<!-- 3rdparty Loggers -->

<logger name="org.springframework.core" level="info">

</logger>

<logger name="org.springframework.beans" level="info">

</logger>

<logger name="org.springframework.context" level="info">

</logger>

<logger name="org.springframework.web" level="info">

</logger>

<logger name="org.springframework.test.context.junit4.SpringJUnit4ClassRunner" level="info">

</logger>

<!-- 下面的logger需要修改为你的项目根路径，如com.sd -->

<logger name="com.stu" level="debug" includeLocation="true" additivity="false">

<appender-ref ref="ErrorLog"/>

<appender-ref ref="Console"/>

</logger>

<!-- 将druid连接池的sql语句打印到日志文件中 -->

<logger name="druid.sql.Statement" level="debug" additivity="false">

<appender-ref ref="Console"/>

<appender-ref ref="RollingFileName"/>

</logger>

<root level="info" includeLocation="true">

<appender-ref ref="ErrorLog"/>

<appender-ref ref="Console"/>

</root>

</Loggers>

</Configuration>

### 三、在web.xml中配置log4j2的加载启动

<!-- log4j2-begin -->

<listener>

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

</listener>

<filter>

<filter-name>log4jServletFilter</filter-name>

<filter-class>org.apache.logging.log4j.web.Log4jServletFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>log4jServletFilter</filter-name>

<url-pattern>/\*</url-pattern>

<dispatcher>REQUEST</dispatcher>

<dispatcher>FORWARD</dispatcher>

<dispatcher>INCLUDE</dispatcher>

<dispatcher>ERROR</dispatcher>

</filter-mapping>

<!-- log4j2-end -->

注：加上上述配置后，就能在项目启动是加载log4j2

### 四、在Mybatis的配置文件中，加入上述配置

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

<!DOCTYPE configuration

PUBLIC "-//mybatis.org//DTD Config 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-config.dtd">

<configuration>

<!-- 使用log4j2打印查询语句 -->

<settings>

<setting name="logImpl" value="LOG4J2" />

</settings>

</configuration>

注：settings配置完毕后，即可实现SQL语句的打印！控制台打印如下结果：

1. 2017-10-23 16:38:01,021 DEBUG (BaseJdbcLogger.java:145) - ==> Preparing: SELECT count(\*) FROM t\_user
2. 2017-10-23 16:38:01,056 DEBUG (BaseJdbcLogger.java:145) - ==> Parameters:
3. 2017-10-23 16:38:01,084 DEBUG (BaseJdbcLogger.java:145) - <== Total: 1

### 五、实现Mybatis官方提供的拦截器，用于记录SQL语句的执行时间

package com.stu.interceptor;

import java.sql.Statement;

import java.util.Properties;

import org.apache.ibatis.executor.statement.StatementHandler;

import org.apache.ibatis.plugin.Interceptor;

import org.apache.ibatis.plugin.Intercepts;

import org.apache.ibatis.plugin.Invocation;

import org.apache.ibatis.plugin.Plugin;

import org.apache.ibatis.plugin.Signature;

import org.apache.ibatis.session.ResultHandler;

/\*\*

\* Sql执行时间记录拦截器

\*/

@Intercepts({@Signature(type = StatementHandler.class, method = "query", args = {Statement.class, ResultHandler.class}),

@Signature(type = StatementHandler.class, method = "update", args = {Statement.class}),

@Signature(type = StatementHandler.class, method = "batch", args = { Statement.class })})

public class SqlCostInterceptor implements Interceptor {

@Override

public Object intercept(Invocation invocation) throws Throwable {

long startTime = System.currentTimeMillis();

try {

return invocation.proceed();

} finally {

long endTime = System.currentTimeMillis();

long sqlCost = endTime - startTime;

System.out.println("执行耗时 : [" + sqlCost + "ms ] ");

}

}

@Override

public Object plugin(Object target) {

return Plugin.wrap(target, this);

}

@Override

public void setProperties(Properties properties) {

}

}

注：Interceptor接口是Mybatis官方提供的拦截接口，创建一个类实现该接口并重写其三个方法并将该类配置在Mybatis的配置文件中，即可拦截SQL语句的执行过程

### 六、将手动编写的拦截器配置在Mybatis配置文件中：

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

<!DOCTYPE configuration

PUBLIC "-//mybatis.org//DTD Config 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-config.dtd">

<configuration>

<!-- 使用log4j2打印查询语句 -->

<settings>

<setting name="logImpl" value="LOG4J2" />

</settings>

<typeAliases>

<package name="com.stu.pojo" />

</typeAliases>

<plugins>

<!-- 拦截器配置 -->

<plugin interceptor="com.stu.interceptor.SqlCostInterceptor" />

</plugins>

</configuration>

### 七、测试SQL的控制台打印结果

1. 2017-10-23 16:38:01,091 DEBUG (BaseJdbcLogger.java:145) - ==> Preparing: select id, username, password, state, isdel, add\_time, money, left\_money from t\_user limit ?,?
2. 2017-10-23 16:38:01,093 DEBUG (BaseJdbcLogger.java:145) - ==> Parameters: 0(Integer), 15(Integer)
3. 2017-10-23 16:38:01,128 DEBUG (BaseJdbcLogger.java:145) - <== Total: 15
4. 执行耗时 : [36ms ]

写在最后：如需转载，请注明出处，如果有什么问题，欢迎在评论区询问