Step 1:   
在pom.xml文件删除log4j的依赖，加入logback的依赖，下面为所有logback的依赖jar包：

<dependency> <groupId>org.springframework</groupId> <artifactId>spring-context</artifactId> <version>${spring.version}</version> <!-- 排除commmons-logging --> <exclusions> <exclusion> <groupId>commons-logging</groupId> <artifactId>commons-logging</artifactId> </exclusion> </exclusions> </dependency> <!-- https://mvnrepository.com/artifact/org.slf4j/slf4j-api --> <dependency> <groupId>org.slf4j</groupId> <artifactId>slf4j-api</artifactId> <version>1.7.25</version> </dependency> <!-- https://mvnrepository.com/artifact/ch.qos.logback/logback-classic --> <dependency> <groupId>ch.qos.logback</groupId> <artifactId>logback-classic</artifactId> <version>1.2.3</version> <!--<scope>test</scope>--> </dependency> <!-- https://mvnrepository.com/artifact/org.logback-extensions/logback-ext-spring --> <dependency> <groupId>org.logback-extensions</groupId> <artifactId>logback-ext-spring</artifactId> <version>0.1.4</version> </dependency> <!-- https://mvnrepository.com/artifact/org.slf4j/jcl-over-slf4j --> <dependency> <groupId>org.slf4j</groupId> <artifactId>jcl-over-slf4j</artifactId> <version>1.7.25</version> </dependency>

Step 2:   
在mybatis-config.xml文件里把原来的“LOG4J“改为“SLF4J“，因为logback是对slf4j的完美实现，并加入如下代码：

<?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> <settings> <!--<setting name="logImpl" value="LOG4J" />--> <!--<setting name="logImpl" value="STDOUT\_LOGGING" />--> <setting name="logImpl" value="SLF4J" /> <!-- 在所有的SQL映射路径上加上“dao”这个前缀 --> <setting name="logPrefix" value="dao."/> </settings> </configuration>

Step 3:   
在web.xml文件中把log4j的相关配置替换成logback的配置：

<context-param> <param-name>logbackConfigLocation</param-name> <param-value>classpath:logback.xml</param-value> </context-param> <listener> <listener-class>ch.qos.logback.ext.spring.web.LogbackConfigListener</listener-class> </listener>

Step 4:   
配置logback.xml：

<?xml version="1.0" encoding="UTF-8" ?> <!-- logback整合日志--> <configuration scan="true" scanPeriod="60 seconds" debug="false"> <include resource="org/springframework/boot/logging/logback/base.xml"/> <!-- 定义日志文件 输入位置 --> <property name="log\_dir" value="${LOG\_PATH}"/> <!--<property name="log\_dir" value="/opt/Log/prom\_prize" />--> <!-- 日志最大的历史 30天 --> <property name="maxHistory" value="30"/> <!-- ConsoleAppender 控制台输出日志 --> <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender"> <!-- 对日志进行格式化 --> <encoder> <pattern>%d{yyyy-MM-dd HH:mm:ss} [%thread] %-5level %logger -%msg%n</pattern> </encoder> </appender> <!-- ERROR级别日志 --> <!-- 滚动记录文件，先将日志记录到指定文件，当符合某个条件时，将日志记录到其他文件 RollingFileAppender--> <appender name="ERROR" class="ch.qos.logback.core.rolling.RollingFileAppender"> <!-- 过滤器，只记录WARN级别的日志 --> <filter class="ch.qos.logback.classic.filter.LevelFilter"> <level>ERROR</level> <onMatch>ACCEPT</onMatch> <onMismatch>DENY</onMismatch> </filter> <!-- 最常用的滚动策略，它根据时间来制定滚动策略.既负责滚动也负责出发滚动 --> <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> <!--日志输出位置 可相对、和绝对路径 --> <fileNamePattern>${log\_dir}/%d{yyyy-MM-dd}/error-log.log</fileNamePattern> <!-- 可选节点，控制保留的归档文件的最大数量，超出数量就删除旧文件假设设置每个月滚动，且<maxHistory>是6， 则只保存最近6个月的文件，删除之前的旧文件。注意，删除旧文件是，那些为了归档而创建的目录也会被删除--> <maxHistory>${maxHistory}</maxHistory> </rollingPolicy> <!-- 按照固定窗口模式生成日志文件，当文件大于20MB时，生成新的日志文件。窗口大小是1到3，当保存了3个归档文件后，将覆盖最早的日志。 <rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy"> <fileNamePattern>${log\_dir}/%d{yyyy-MM-dd}/.log.zip</fileNamePattern> <minIndex>1</minIndex> <maxIndex>3</maxIndex> </rollingPolicy> --> <!-- 查看当前活动文件的大小，如果超过指定大小会告知RollingFileAppender 触发当前活动文件滚动 <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy"> <maxFileSize>5MB</maxFileSize> </triggeringPolicy> --> <encoder> <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger - %msg%n</pattern> </encoder> </appender> <!-- WARN级别日志 appender --> <appender name="WARN" class="ch.qos.logback.core.rolling.RollingFileAppender"> <!-- 过滤器，只记录WARN级别的日志 --> <filter class="ch.qos.logback.classic.filter.LevelFilter"> <level>WARN</level> <onMatch>ACCEPT</onMatch> <onMismatch>DENY</onMismatch> </filter> <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> <!-- 按天回滚 daily --> <fileNamePattern>${log\_dir}/%d{yyyy-MM-dd}/warn-log.log </fileNamePattern> <!-- 日志最大的历史 60天 --> <maxHistory>${maxHistory}</maxHistory> </rollingPolicy> <encoder> <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger - %msg%n</pattern> </encoder> </appender> <!-- INFO级别日志 appender --> <appender name="INFO" class="ch.qos.logback.core.rolling.RollingFileAppender"> <!-- 过滤器，只记录INFO级别的日志 --> <filter class="ch.qos.logback.classic.filter.LevelFilter"> <level>INFO</level> <onMatch>ACCEPT</onMatch> <onMismatch>DENY</onMismatch> </filter> <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> <!-- 按天回滚 daily --> <fileNamePattern>${log\_dir}/%d{yyyy-MM-dd}/info-log.log </fileNamePattern> <!-- 日志最大的历史 60天 --> <maxHistory>${maxHistory}</maxHistory> </rollingPolicy> <encoder> <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger - %msg%n</pattern> </encoder> </appender> <!-- DEBUG级别日志 appender --> <appender name="DEBUG" class="ch.qos.logback.core.rolling.RollingFileAppender"> <!-- 过滤器，只记录DEBUG级别的日志 --> <filter class="ch.qos.logback.classic.filter.LevelFilter"> <level>DEBUG</level> <onMatch>ACCEPT</onMatch> <onMismatch>DENY</onMismatch> </filter> <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> <!-- 按天回滚 daily --> <fileNamePattern>${log\_dir}/%d{yyyy-MM-dd}/debug-log.log </fileNamePattern> <!-- 日志最大的历史 60天 --> <maxHistory>${maxHistory}</maxHistory> </rollingPolicy> <encoder> <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger - %msg%n</pattern> </encoder> </appender> <!-- TRACE级别日志 appender --> <appender name="TRACE" class="ch.qos.logback.core.rolling.RollingFileAppender"> <!-- 过滤器，只记录ERROR级别的日志 --> <filter class="ch.qos.logback.classic.filter.LevelFilter"> <level>TRACE</level> <onMatch>ACCEPT</onMatch> <onMismatch>DENY</onMismatch> </filter> <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> <!-- 按天回滚 daily --> <fileNamePattern>${log\_dir}/%d{yyyy-MM-dd}/trace-log.log </fileNamePattern> <!-- 日志最大的历史 60天 --> <maxHistory>${maxHistory}</maxHistory> </rollingPolicy> <encoder> <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger - %msg%n</pattern> </encoder> </appender> <!--输出到mysql数据库的appender配置 --> <appender name="db" class="ch.qos.logback.classic.db.DBAppender"> <connectionSource class="ch.qos.logback.core.db.DriverManagerConnectionSource"> <driverClass>com.mysql.jdbc.Driver</driverClass> <url>jdbc:mysql://120.0.0.1:3306/logback\_member?characterEncoding=utf8</url> <user>root</user> <password>123456789</password> </connectionSource> </appender> <logger name="org.mybatis" level="debug" /> <logger name="ch.qos.logback"/> <!-- 就是这个监控了mybatis日志输出，配合上面的“dao” --> <logger name="dao" level="DEBUG"/> <root level="INFO"> <!-- 控制台输出 --> <appender-ref ref="STDOUT" /> <!--<!– 文件输出 –>--> <appender-ref ref="ERROR" /> <appender-ref ref="INFO" /> <appender-ref ref="WARN" /> <appender-ref ref="DEBUG" /> <appender-ref ref="TRACE" /> </root> </configuration>