目录

[常见组件介绍](#_Toc25802_WPSOffice_Level1) [1](#_Toc25802_WPSOffice_Level1)

[slf4j兼容其他日志](#_Toc13671_WPSOffice_Level1) [2](#_Toc13671_WPSOffice_Level1)

[logback.xml文件位置更改](#_Toc1743_WPSOffice_Level1) [3](#_Toc1743_WPSOffice_Level1)

[日志，是什么、为什么、怎么办](#_Toc32280_WPSOffice_Level1) [3](#_Toc32280_WPSOffice_Level1)

[log4jdbc组件](#_Toc23978_WPSOffice_Level1) [3](#_Toc23978_WPSOffice_Level1)

# 常见组件介绍

目前来讲，我一共见过这么几种日志。

log4j，知道的最早，但至今不会用的玩意 ：）

logback

java.util.logging，是吧，没想到吧，java居然自带一套日志组件

这时候毛糙的少年多半会这样提醒我，喂，处男，你漏掉了slf4j和commons-logging

不会漏的，都射这么多了，有B数的。

不把这些2个东西列出来，是因为这两个jar包都是接口，你单独用slf4j或是commons-logging是没法打日志的。

真正往控制台输出，文件里写东西的是上面那3个家伙。

slf4j或是commons-logging要起效，必须搭配上面那三个中的一个。但也不能乱搭，slf4j的话，可以与那3个里的任一个组合，而commons-logging只能与log4j与java.util.logging组合。

但反过来，上面那3个家伙，不依赖slf4j与commons-logging，单挑一个出来都可以完成打日志的活。

那事情是怎么发展到今天这个局面的呢。

当初Apache开发了log4j，稳定，高效，强。接着Apache极力向sun公司推荐把log4j做为jdk1.4的一部分。sun看完后非常满意，于是拒绝了。

但由于sun自带的日志java.util.logging性能太烂，大家基本上还是在用Apache的log4j。虽说java.util.logging用的人不多，但市面上已经存在了两种工具。有多恶心呢，如果你的项目依赖了另外两个项目，其中一个用log4j，另一个用java.util.logging，你要怎么统一日志输出？

没办法，Apache又开发了commons-logging，这是一套接口，以后大家打日志的时候只用commons-logging包里的接口，然后底层怎么实现，爱用哪个用哪个。

紧接着，在log4j马上要出2的时候，这个组他妈的分裂了。

走的那帮人折腾出了slf4j。。。。。logback谁做的不清楚，但冲着slf4j与logback的组合程度，我觉得和走的那帮人脱不了干系。

slf4j，也是接口，但他屌在兼容一切日志，而且还和spring扯上了关系。真是印证了后来者居上这句话。

# slf4j兼容其他日志

超简单，贴jar包就好了。下面是栗子，slf4j组合logback，并兼容其他日志组件。

<!-- 要如何做到slf4j，兼容所有输出日志工具 -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>${slf4j.version}</version>

</dependency>

<!--commons-logging替换成slf4j-->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>jcl-over-slf4j</artifactId>

<version>${slf4j.version}</version>

</dependency>

<!--log4j替换成slf4j-->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>log4j-over-slf4j</artifactId>

<version>${slf4j.version}</version>

</dependency>

<!-- java.util.logging替换成slf4j -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>jul-to-slf4j</artifactId>

<version>${slf4j.version}</version>

</dependency>

<!-- 本项目采用的日志实现 -->

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>1.2.3</version>

</dependency>

<!--logback spring扩展-->

<dependency>

<groupId>org.logback-extensions</groupId>

<artifactId>logback-ext-spring</artifactId>

<version>0.1.4</version>

</dependency>

# logback.xml文件位置更改

<context-param>

<param-name>logbackConfigLocation</param-name>

<param-value>classpath:config/logback.xml</param-value>

</context-param>

<listener>

<listener-class>ch.qos.logback.ext.spring.web.LogbackConfigListener</listener-class>

</listener>

这样就可以随便找地方放配置文件了。但有一个注意点，这个监听器一定要放在最前面。

# 日志，是什么、为什么、怎么办

日志就是一串文字，存在的意义是让程序员了解一个正在运行的程序，并排查问题。日志需要记录程序执行到哪一步了，有没有出问题。如果有，问题是什么，因为什么引起的，具体在哪个流程发生的。

以上这段话解决了是什么，为什么，怎么办。

# log4jdbc组件

这是依赖，maven库里叫这个名字的jar太多了。

<dependency>

<groupId>org.bgee.log4jdbc-log4j2</groupId>

<artifactId>log4jdbc-log4j2-jdbc4.1</artifactId>

<version>1.16</version>

</dependency>

用接口代理数据源net.sf.log4jdbc.sql.jdbcapi.DataSourceSpy

全部替换成代理数据源

logback.xml新增<logger>