

ImportNew

- [首页](#)
- [所有文章](#)
- [资讯](#)
- [Web](#)
- [架构](#)
- [基础技术](#)
- [书籍](#)
- [教程](#)
- [Java小组](#)
- [工具资源](#)

- 导航条 -

混乱的 Java 日志体系

2016/09/10 | 分类: [基础技术](#) | [0 条评论](#) | 标签: [log](#)

分享到:

9 原文出处: [xirong](#)



一、困扰的疑惑

目前的日志框架有 jdk 自带的 logging, log4j1、log4j2、logback , 这些框架都自己定制了日志 API , 并且有相应的实现; 目前用于实现日志统一的框架 Apache commons-logging、slf4j , 遵循「面向接口编程」的原则, 这两大框架可以让用户在程序运行期间去选择具体的日志实现系统 (log4j1\log4j2\logback等) 来记录日志, 是统一抽象出来的一些接口。

这些日志系统涉及到的繁杂的各种集成 jar 包, 如下:

- log4j、log4j-api、log4j-core
- log4j-1.2-api、log4j-jcl、log4j-slf4j-impl、log4j-jul
- logback-core、logback-classic、logback-access
- commons-logging
- slf4j-api、slf4j-log4j12、slf4j-simple、jcl-over-slf4j、slf4j-jdk14、log4j-over-slf4j、slf4j-jcl

log4j

Apache 的一个开放源代码项目, 通过使用Log4j, 我们可以控制日志信息输送的目的地是控制台、文件、GUI组件、甚至是套接口服务 器、NT的事件记录器、UNIX Syslog守护进程等; 用户也可以控制每一条日志的输出格式; 通过定义每一条日志信息的级别, 用户能够更加细致地控制日志的生成过

程。这些可以通过一个 配置文件来灵活地进行配置，而不需要修改程序代码。最新的目前版本为 log4j 1.2，maven 依赖如下：

```
1 <dependency>
2   <groupId>log4j</groupId>
3   <artifactId>log4j</artifactId>
4   <version>1.2.17</version>
5 </dependency>
```

Logback

Logback 是由 log4j 创始人设计的又一个开源日记组件，Logback 当前分成三个模块：logback-core, logback-classic 和 logback-access。logback-core 是其它两个模块的基础模块，logback-classic 是 log4j 的一个 改良版本。此外 logback-classic 完整实现 SLF4J API 使你可以很方便地更换成其它日记系统如 log4j 或 JDK14 Logging。logback-access 访问模块与 Servlet 容器集成提供通过 Http 来访问日记的功能。maven 最新依赖如下：

```
1 <dependency>
2   <groupId>ch.qos.logback</groupId>
3   <artifactId>logback-core</artifactId>
4   <version>1.1.6</version>
5 </dependency>
6 <dependency>
7   <groupId>ch.qos.logback</groupId>
8   <artifactId>logback-classic</artifactId>
9   <version>1.1.6</version>
10 </dependency>
11 <dependency>
12   <groupId>org.slf4j</groupId>
13   <artifactId>slf4j-api</artifactId>
14   <version>1.7.18</version>
15 </dependency>
```

Logback 作为一个通用可靠、快速灵活的日志框架，将作为 Log4j 的替代和 SLF4J 组成新的日志系统的完整实现。Logback 声称具有极佳的性能，“某些关键操作，比如判定是否记录一条日志语句的操作，其性能得到了显著的提高。这个操作在 Logback 中需要 3 纳秒，而在 Log4J 中则需要 30 纳秒。LogBack 创建记录器（logger）的速度也更快：13 微秒，而在 Log4J 中需要 23 微秒。更重要的是，它获取已存在的记录器只需 94 纳秒，而 Log4J 需要 2234 纳秒，时间减少到了 1/23。跟 JUL 相比的性能提高也是显著的”。另外，Logback 的所有文档是全面免费提供的，不象 Log4J 那样只提供部分免费文档而需要用户去购买付费文档。具体包括：

- 更快的执行速度
- 更充分的测试
- logback-classic 非常自然的实现了 SLF4J
- 使用 XML 配置文件或者 Groovy
- 自动重新载入配置文件
- 优雅地从 I/O 错误中恢复
- 自动清除旧的日志归档文件
- 自动压缩归档日志文件
- 谨慎模式
- Lilith
- 配置文件中的条件处理
- 更丰富的过滤
- Logback-access 模块，提供了通过 HTTP 访问日志的能力，是 logback 不可或缺的组成部分

详细的介绍可以参考：<http://www.oschina.net/translate/reasons-to-prefer-logbak-over-log4j>


越来越多的开源项目依赖 Logback:

- Akka
- Apache Airvata
- Apache Camel
- Apache Cocoon 3
- Apache Jackrabbit
- Apache OpenMeetings
- Apache S4
- Apache Syncope
- Apache Tobago
- Artifactory
- Bootique
- Citizen Intelligence Agency
- dCache
- Dropwizard
- Geomajas
- Gradle
- Grinder
- Gyrex
- JAOP
- JMX Monitor
- JWebUnit
- Lift
- Metrics
- OpenGDA
- OpenRDF
- OpenTSDB
- Play Framework
- PSI probe
- Red5
- Scalate
- Scalatra
- Shibboleth
- Sonar
- SpringSource dm Server
- StreamBase
- Virgo Web Server
- XtremeMP
- Xuggler
- Xwiki
- Zabbix

SLF4J

简单日记门面(Facade) SLF4J 是为各种 logging APIs 提供一个简单统一的接口，从而使得最终用户能够在部署的时候配置自己希望的logging APIs实现。Logging API实现既可以选择直接实现SLF4J接的logging APIs如： NLOG4J、SimpleLogger。也可以通过SLF4J提供的API实现来开发相应的适配器如Log4jLoggerAdapter、JDK14LoggerAdapter。

Apache Common-Logging

目前广泛使用的Java日志门面库。通过动态查找的机制，在程序运行时自动找出真正使用的日志库。但由于它使用了ClassLoader寻找和载入底层的日志库，导致了象OSGi这样的框架无法正常工作，由于其不同的插件使用自己的ClassLoader。OSGi这种机制保证了插件互相独立，然而确使Apache Common-Logging无法工作。

SLF4J 库类似于 Apache Common-Logging。但是，他在编译时静态绑定真正的Log库。使用SLF4J时，如果你需要使用某一种日志实现，那么你必须选择正确的SLF4J的jar包的集合，如此便可以在OSGi中使用了。另外，SLF4J 支持参数化的log字符串，避免了之前为了减少字符串拼接的性能损耗而不得不写的if(logger.isDebugEnabled()), 现在你可以直接写：logger.debug("current user is: {}", user)。拼装消息被推迟到了它能够确定是不是要显示这条消息的时候，但是获取参数的代价并没有幸免。同时，日志中的参数若超过三个，则需要将参数以数组的形式传入，如：

```
1 | Object[] params = {value1, value2, value3};
2 | logger.debug("first value: {}, second value: {} and third value: {}. ", params);
```

现在，Hibernate、Jetty、Spring-OSGi、Wicket和MINA等越来越多的开源项目都已经迁移到了SLF4J，由此可见SLF4J的[影响力](#)不可忽视。

- [Apache ActiveMQ](#)
- [Apache Archiva](#)
- [Apache Camel](#)
- [Apache Directory](#)
- [Apache FTPServer](#)
- [Apache Geronimo](#)
- [Apache Graffito](#)
- [Apache Jackrabbit](#)
- [Apache Mina](#)
- [Apache Qpid](#)
- [Apache ServiceMix](#)
- [Apache Sling](#)
- [Apache Solr](#)
- [Apache Tapestry](#)
- [Apache Wicket](#)
- [Aperture](#)
- [Apogee](#)
- [Artifactory](#)
- [AsyncWeb](#)
- [DbUnit](#)
- [Display tag](#)
- [Ehcache](#)
- [GMaven](#)
- [Gradle](#)
- [GreenMail](#)
- [GumTree](#)
- [H2 Database](#)
- [HA-JDBC](#)
- [Hibernate](#)
- [Igenko](#)
- [Jabsorb](#)
- [Jetty v6](#)
- [jLynx](#)
- [JMesa](#)
- [JODConverter](#)
- [JTrac](#)
- [JWebUnit 2.x](#)
- [JQuantLib](#)
- [LIFERAY](#)
- [Lift](#)
- [log4jdbc](#)
- [Magnolia](#)
- [MRCP4J](#)
- [Mindquarry](#)
- [Mugshot](#)
- [Mule](#)
- [Nexus](#)
- [Novocode](#)
- [NetCDF](#)
- [OpenMeetings](#)
- [OpenRDF](#)
- [Penrose](#)
- [PSI Probe](#)
- [PZFileReader](#)
- [Quartz Scheduler](#)
- [QuickFIX/J](#)
- [Sonar](#)
- [SMSJ](#)
- [Spring-OSGi](#)
- [SpringSource dm Server™](#)
- [StreamBase](#)
- [TimeFinder](#)
- [WTFIGO](#)
- [YASL](#)
- [Xoactory](#)
- [XWiki](#)

二、how to use

介绍完这些日志框架后，那么这些东西怎么用，之间的依赖是什么样子的，还有启动初始化的过程究竟干了什么，怎么找到相应的实现类及配置的？

这四篇文章是我目前在互联网上面发现的最详细的介绍，请参考：



1. [jdk-logging、log4j、logback日志介绍及原理](#) 三个日志系统的实现机制介绍
2. [commons-logging与jdk-logging、log4j1、log4j2、logback的集成原理](#) Apache Commons-logging 通用日志框架与日志系统的机制介绍
3. [SLF4J 与 jdk-logging、log4j1、log4j2、logback的集成原理](#) SLF4J 通用日志框架与具体日志实现系统的机制机制介绍，包括依赖的jar包，jar冲突处理等；
4. [slf4j、jcl、jul、log4j1、log4j2、logback大总结](#) 各个组件的jar包以及目前系统日志需要切换实现方式的方法，推荐阅读。

```

1  <!-- ===== -->
2  <!-- 日志及相关依赖 (用slf4j+logback代替jcl+log4j) -->
3  <!-- ===== -->
4  <dependency>
5  <groupId>org.slf4j</groupId>
6  <artifactId>slf4j-api</artifactId>
7  <version>1.7.7</version>
8  </dependency>
9  <!-- 强制使用 logback的绑定 -->
10 <dependency>
11 <groupId>ch.qos.logback</groupId>
12 <artifactId>logback-core</artifactId>
13 <version>1.1.3</version>
14 </dependency>
15 <dependency>
16 <groupId>ch.qos.logback</groupId>
17 <artifactId>logback-classic</artifactId>
18 <version>1.1.3</version>
19 </dependency>
20 <!-- 强制使用 logback的绑定, 这里去除对log4j 的绑定 -->
21 <dependency>
22 <groupId>org.slf4j</groupId>
23 <artifactId>slf4j-log4j12</artifactId>

```

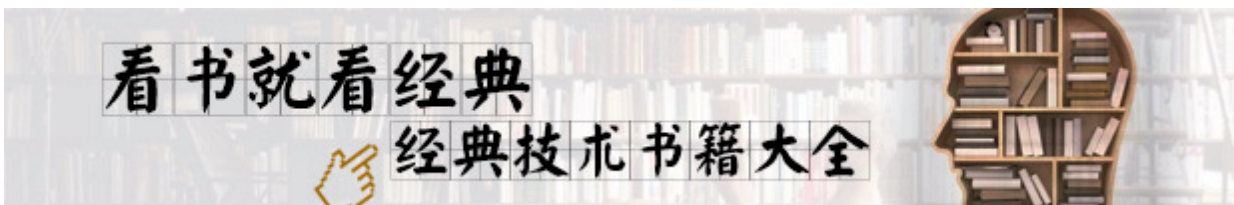
```
24 <version>99.0-does-not-exist</version>
25 </dependency>
26 <!-- slf4j 的桥接器,将第三方类库对 log4j 的调用 delegate 到 slf api 上 -->
27 <!-- 这个桥接器是自己做的,主要是我们依赖的类库存在很多硬编码的引用 -->
28 <dependency>
29 <groupId>org.slf4j</groupId>
30 <artifactId>log4j-over-slf4j</artifactId>
31 <version>1.7.14-SNAPSHOT</version>
32 </dependency>
33 <dependency>
34 <groupId>org.slf4j</groupId>
35 <artifactId>jcl-over-slf4j</artifactId>
36 <version>1.7.7</version>
37 </dependency>
38 <!-- 强制排除 log4j 的依赖,全部 delegate 到 log4j-over-slf4j 上 -->
39 <dependency>
40 <groupId>log4j</groupId>
41 <artifactId>log4j</artifactId>
42 <version>99.0-does-not-exist</version>
43 </dependency>
44 <dependency>
45 <groupId>apache-log4j</groupId>
46 <artifactId>log4j</artifactId>
47 <version>999-not-exist</version>
48 </dependency>
49 <!-- slf logback 配置结束 -->
```

三、结束

通过上述阅读,对「混乱的 java 日志体系」应该有了较清楚的理解,但是真的存在项目需要切换日志系统的情况吗?根据我的理解,很多当你的系统越来越庞大,越来越复杂需要系统[重构](#),解决性能的时候,可以考虑升级你当前的日志配置。如果你是新启用的项目,直接考虑使用Logback + SLF4j 即可。

参考资料

- [Apache logging services](#)
- [LOGBack home page](#)
- [Java日志框架: SLF4J, Apache Common-Logging, Log4J和Logback](#)
- [jdk-logging、log4j、logback日志介绍及原理](#)



相关文章

- [学习笔记: The Log \(我所读过的最好的一篇分布式技术文章\)](#)
- [Android一周开发要闻 2013-2-5](#)
- [SPRING SECURITY JAVA配置: 简介](#)
- [ArrayList vs. LinkedList vs. Vector](#)
- [可提高Java开发性能的5款调试工具](#)
- [Tomcat server.xml配置示例](#)
- [ZooKeeper编程指导](#)

- [JDK动态代理详解](#)
- [Jodd 一款优雅的 Java 工具集](#)
- [大数据搜索选开源还是商业软件? Elasticsearch 对比 Splunk](#)

发表评论

Comment form

Name*

姓名

邮箱*

请填写邮箱

网站 (请以 http://开头)

请填写网站地址

评论内容*

请填写评论内容

提交

(*) 表示必填项

提交评论

还没有评论。

[« JVM类加载的那些事](#)
[深入浅出ThreadLocal »](#)

Search for:

Search

Search

java夜校

Docker

Nginx

Redis

ActiveMQ

Hessian

FastDFS

MyCat

Velocity

SpringBoot

更多前沿技术

- [本周热门文章](#)
- [本月热门](#)
- [热门标签](#)

0 [JDK 10 的 109 项新特性](#)

1 [通向架构师的道路（第二十六天）漫谈...](#)

2 [Java 11 发布线路图：有哪些值得...](#)

3 [在 Java 8 中避免 Null 检查](#)

4 [Java 虚拟机1：什么是 Java](#)

5 [Java代码优化](#)

6 [Java 虚拟机 2：Java 内存区域...](#)

7 [Java 虚拟机 3：常用 JVM 命...](#)

8 [Java 虚拟机 4：内存溢出](#)

9 [Java虚拟机 5：Java垃圾回收（G...](#)



最新评论



Re: [通向架构师的道路（第七天）之漫谈使用 ...](#)

事务是threadLocal 应用最普遍的地方，把spring jdbc 和 hibernate 对... brain



Re: [Java虚拟机学习（1）：体系结构...](#)

MinHeapFreeRatio MaxHeapFreeRatio这两个参数在JDK8中没有起作用 史锦明



Re: [贫血领域模型是如何导致糟糕的软...](#)

1 1



Re: [Java虚拟机学习（7）：对象内存...](#)

请教一下大神，方法区GC在什么时候触发啊 史锦明



Re: [JAVA回调机制\(CallBack\)...](#)

写的真好 helloWorld



Re: [Java并发编程：volatile关键字...](#)

因为这是在两个线程中。发生重排后，语句2先于语句1执行，然后线程2就满足条件执行语句3了，当然语句2... 梦中客



Re: [Java7/8 中的 HashMap 和 Concur...](#)

\ "由于是双倍扩容，迁移过程中，会将原来 table[i] 中的链表的所有节点，分拆到新的数组的 ne... jason



Re: [Java中的线程库 — Quas...](#)

从API易用性和容易理解程度，新手其实更推荐<https://github.com/offbynull...> [Xinyuan.Yan](#)



关于ImportNew

ImportNew 专注于 Java 技术分享。于2012年11月11日 11:11正式上线。是的，这是一个很特别的时刻 :)



ImportNew 由两个 Java 关键字 import 和 new 组成，意指：Java 开发者学习新知识的网站。import 可认为是学习和吸收，new 则可认为是新知识、新技术圈子和新朋友.....



联系我们

Email: ImportNew.com@gmail.com

新浪微博: [@ImportNew](#)

推荐微信号



ImportNew



安卓应用频道



Linux爱好者

反馈建议: ImportNew.com@gmail.com

广告与商务合作QQ: 2302462408

推荐关注

[小组](#) — 好的话题、有启发的回复、值得信赖的圈子

[头条](#) — 写了文章？看干货？去头条！

[相亲](#) — 为IT单身男女服务的征婚传播平台

[资源](#) — 优秀的工具资源导航

[翻译](#) — 活跃 & 专业的翻译小组

[博客](#) — 国内外的精选博客文章

[设计](#) — UI,网页，交互和用户体验

[前端](#) — JavaScript, HTML5, CSS

[安卓](#) — 专注Android技术分享

[iOS](#) — 专注iOS技术分享

[Java](#) — 专注Java技术分享

[Python](#) — 专注Python技术分享

© 2018 ImportNew

