

<<深入分析Java Web技术内幕>>

图书基本信息

书名：<<深入分析Java Web技术内幕>>

13位ISBN编号：9787121179907

10位ISBN编号：7121179903

出版时间：2012-9

出版时间：电子工业出版社

作者：许令波

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<深入分析Java Web技术内幕>>

### 内容概要

《深入分析Java Web技术内幕》围绕Java Web相关技术从三方面全面深入地进行阐述。首先介绍前端知识，主要介绍Java Web开发中涉及的一些基本知识，包括Web请求过程、HTTP协议、DNS技术和CDN技术。其次深入介绍Java技术，包括I/O技术、中文编码问题、Javac编译原理、class文件结构解析、ClassLoader工作机制及JVM的内存管理等。最后介绍Java服务端技术，主要包括Servlet、Session与Cookie、Tomcat与Jetty服务器、Spring容器、Ibatis框架和Velocity框架等原理介绍。本书不仅介绍这些技术和框架的工作原理，而且结合示例来讲解，通过通俗易懂的文字和丰富生动的配图，让读者充分并深入理解它们的内部工作原理，同时还结合了设计模式来介绍这些技术背后的架构思维。

## <<深入分析Java Web技术内幕>>

### 作者简介

许令波，毕业于合肥工业大学，获计算机硕士学位。

热爱Java Web技术，关注服务端性能优化，热衷开源技术的研究和分享，曾获developerWorks最佳作者称号。

2009年进入淘宝工作，目前从事模板渲染框架与MVC框架的开发与应用、Java Web的性能优化、高访问量系统静态化和商品详情系统的业务改造等工作。

## 书籍目录

第1章 深入Web请求过程1 1.1 B/S网络架构概述2 1.2 如何发起一个请求4 1.3 HTTP协议解析6 1.3.1 查看HTTP信息的工具8 1.3.2 浏览器缓存机制9 1.4 DNS域名解析12 1.4.1 DNS域名解析过程12 1.4.2 跟踪域名解析过程15 1.4.3 清除缓存的域名18 1.4.4 几种域名解析方式19 1.5 CDN工作机制20 1.5.1 CDN架构20 1.5.2 负载均衡21 1.6 总结24 第2章 深入分析Java/I/O的工作机制25 2.1 Java的I/O类库的基本架构25 2.1.1 基于字节的I/O操作接口26 2.1.2 基于字符的I/O操作接口27 2.1.3 字节与字符的转化接口28 2.2 磁盘I/O工作机制29 2.2.1 几种访问文件的方式29 2.2.2 Java访问磁盘文件33 2.2.3 Java序列化技术34 2.3 网络I/O工作机制36 2.3.1 TCP状态转化37 2.3.2 影响网络传输的因素39 2.3.3 JavaSocket的工作机制39 2.3.4 建立通信链路40 2.3.5 数据传输41 2.4 NIO的工作方式41 2.4.1 BIO带来的挑战41 2.4.2 NIO的工作机制42 2.4.3 Buffer的工作方式45 2.4.4 NIO的数据访问方式47 2.5 I/O调优49 2.5.1 磁盘I/O优化49 2.5.2 TCP网络参数调优50 2.5.3 网络I/O优化52 2.6 设计模式解析之适配器模式56 2.6.1 适配器模式的结构56 2.6.2 Java/I/O中的适配器模式57 2.7 设计模式解析之装饰器模式57 2.7.1 装饰器模式的结构58 2.7.2 Java/I/O中的装饰器模式58 2.8 适配器模式与装饰器模式的区别59 2.9 总结59 第3章 深入分析JavaWeb中的中文编码问题60 3.1 几种常见的编码格式60 3.1.1 为什么要编码60 3.1.2 如何“翻译”61 3.2 Java中需要编码的场景63 3.2.1 I/O操作中存在的编码63 3.2.2 内存操作中的编码65 3.3 Java中如何编解码66 3.3.1 按照ISO—8859—1编码68 3.3.2 按照GB2312编码69 3.3.3 按照GBK编码70 3.3.4 按照UTF—16编码70 3.3.5 按照UTF—8编码71 3.3.6 UTF—8编码代码片段71 3.3.7 几种编码格式的比较73 3.4 JavaWeb中涉及的编解码73 3.4.1 URL的编解码75 3.4.2 HTTPHeader的编解码78 3.4.3 POST表单的编解码78 3.4.4 HTTPBODY的编解码79 3.5 JS中的编码问题80 3.5.1 外部引入JS文件80 3.5.2 JS的URL编码81 3.5.3 其他需要编码的地方83 3.6 常见问题分析83 3.6.1 中文变成了看不懂的字符83 3.6.2 一个汉字变成一个问号84 3.6.3 一个汉字变成两个问号84 3.6.4 一种不正确的正确编码85 3.7 总结86 第4章 Javac编译原理87 4.1 Javac是什么88 4.2 Javac编译器的基本结构88 4.3 Javac工作原理分析90 4.3.1 词法分析器91 4.3.2 语法分析器98 4.3.3 语义分析器103 4.3.4 代码生成器113 4.4 设计模式解析之访问者模式116 4.4.1 访问者模式的结构117 4.4.2 Javac中访问者模式的实现118 4.5 总结119 第5章 深入class文件结构120 5.1 JVM指令集简介120 5.1.1 类相关的指令122 5.1.2 方法的定义123 5.1.3 属性的定义124 5.1.4 其他指令集125 5.2 class文件头的表示形式133 5.3 常量池137 5.3.1 UTF8常量类型140 5.3.2 Fieldref、Methodref常量类型141 5.3.3 Class常量类型141 5.3.4 NameAndType常量类型142 5.4 类信息142 5.5 Fields和Methods定义143 5.6 类属性描述147 5.7 Javap生成的class文件结构148 5.7.1 LineNumberTable150 5.7.2 LocalVariableTable151 5.8 总结153 第6章 深入分析ClassLoader工作机制154 6.1 ClassLoader类结构分析155 6.2 ClassLoader的等级加载机制156 6.3 如何加载class文件159 6.3.1 加载字节码到内存159 6.3.2 验证与解析161 6.3.3 初始化Class对象161 6.4 常见加载类错误分析161 6.4.1 ClassNotFoundException161 6.4.2 NoClassDefFoundError162 6.4.3 UnsatisfiedLinkError163 6.4.4 ClassCastException164 6.4.5 ExceptionInInitializerError165 6.5 常用的ClassLoader分析166 6.6 如何实现自己的ClassLoader170 6.6.1 加载自定义路径下的class文件170 6.6.2 加载自定义格式的class文件172 6.7 实现类的热部署174 6.8 Java应不应该动态加载类176 6.9 总结177 第7章 JVM体系结构与工作方式178 7.1 JVM体系结构178 7.1.1 何谓JVM178 7.1.2 JVM体系结构详解181 7.2 JVM工作机制183 7.2.1 机器如何执行代码183 7.2.2 JVM为何选择基于栈的架构184 7.2.3 执行引擎的架构设计185 7.2.4 执行引擎的执行过程186 7.2.5 JVM方法调用栈191 7.3 总结195 第8章 JVM内存管理196 8.1 物理内存与虚拟内存197 8.2 内核空间与用户空间198 8.3 Java中哪些组件需要使用内存199 8.3.1 Java堆199 8.3.2 线程199 8.3.3 类和类加载器200 8.3.4 NIO200 8.3.5 JNI201 8.4 JVM内存结构201 8.4.1 PC寄存器202 8.4.2 Java栈202 8.4.3 堆203 8.4.4 方法区203 8.4.5 运行时常量池204 8.4.6 本地方法栈204 8.5 JVM内存分配策略204 8.5.1 通常的内存分配策略205 8.5.2 Java中内存分配详解205 8.6 JVM内存回收策略210 8.6.1 静态内存分配和回收210 8.6.2 动态内存分配和回收211 8.6.3 如何检测垃圾211 8.6.4 基于分代的垃圾收集算法213 8.7 内存问题分析222 8.7.1 GC日志分析222 8.7.2 堆快照文件分析225 8.7.3 JVMCrash日志分析225 8.8 实例1231 8.9 实例2233 8.10 实例3235 8.11 总结240 第9章 Servlet工作原理分析241 9.1 从Servlet容器说起241 9.1.1 Servlet容器的启动过程242 9.1.2 Web应用的初始化工作245 9.2 创建Servlet实例247 9.2.1 创建Servlet对象248 9.2.2 初始化Servlet248 9.3 Servlet体系结构250 9.4 Servlet如何工作253 9.5 Servlet中的Listener255 9.6 Filter如何工作257 9.7 Servlet中的url—pattern259 9.8 总结260 第10章 深

## &lt;&lt;深入分析Java Web技术内幕&gt;&gt;

入理解Session与Cookie261 10.1 理解Cookie262 10.1.1 Cookie属性项262 10.1.2 Cookie如何工作263 10.1.3 使用Cookie的限制266 10.2 理解Session267 10.2.1 Session与Cookie267 10.2.2 Session如何工作268 10.3 Cookie安全问题271 10.4 分布式Session框架272 10.4.1 存在哪些问题272 10.4.2 可以解决哪些问题273 10.4.3 总体实现思路273 10.5 Cookie压缩278 10.6 表单重复提交问题280 10.7 总结281 第11章 Tomcat的系统架构与设计模式282 11.1 Tomcat总体设计282 11.1.1 Tomcat总体结构283 11.1.2 Connector组件289 11.1.3 Servlet容器Container294 11.1.4 Tomcat中的其他组件305 11.2 Tomcat中的设计模式305 11.2.1 门面设计模式305 11.2.2 观察者设计模式307 11.2.3 命令设计模式309 11.2.4 责任链设计模式310 11.3 总结312 第12章 Jetty的工作原理解析313 12.1 Jetty的基本架构313 12.1.1 Jetty的基本架构简介313 12.1.2 Handler的体系结构315 12.2 Jetty的启动过程316 12.3 接受请求317 12.3.1 基于HTTP协议工作317 12.3.2 基于AJP工作319 12.3.3 基于NIO方式工作322 12.4 处理请求323 12.5 与Jboss集成326 12.6 与Tomcat的比较327 12.6.1 架构比较327 12.6.2 性能比较328 12.6.3 特性比较328 12.7 总结329 第13章 Spring框架的设计理念与设计模式分析330 13.1 Spring的骨骼架构330 13.1.1 Spring的设计理念331 13.1.2 核心组件如何协同工作332 13.2 核心组件详解333 13.2.1 Bean组件333 13.2.2 Context组件335 13.2.3 Core组件336 13.2.4 Ioc容器如何工作338 13.3 Spring中AOP特性详解348 13.3.1 动态代理的实现原理348 13.3.2 SpringAOP如何实现351 13.4 设计模式解析之代理模式354 13.4.1 代理模式原理354 13.4.2 Spring中代理模式的实现354 13.5 设计模式解析之策略模式357 13.5.1 策略模式原理357 13.5.2 Spring中策略模式的实现358 13.6 总结358 第14章 SpringMVC工作机制与设计模式360 14.1 SpringMVC的总体设计360 14.2 Control设计365 14.2.1 HandlerMapping初始化366 14.2.2 HandlerAdapter初始化368 14.2.3 Control的调用逻辑369 14.3 Model设计370 14.4 View设计371 14.5 框架设计的思考373 14.5.1 为什么需要框架373 14.5.2 需要什么样的框架373 14.5.3 框架设计的原则374 14.5.4 “指航灯” 374 14.5.5 最基本的原则374 14.6 设计模式解析之模板模式375 14.6.1 模板模式的结构375 14.6.2 SpringMVC中的模板模式示例376 14.7 总结377 第15章 深入分析Ibatis框架之系统架构与映射原理378 15.1 Ibatis框架主要的类层次结构378 15.2 Ibatis框架的设计策略379 15.3 Ibatis框架的运行原理381 15.4 示例383 15.5 Ibatis对SQL语句的解析385 15.6 数据库字段映射到Java对象386 15.7 示例运行的结果388 15.8 设计模式解析之简单工厂模式388 15.8.1 简单工厂模式的实现原理388 15.8.2 Ibatis中的简单工厂模式示例389 15.9 设计模式解析之工厂模式390 15.9.1 工厂模式的实现原理390 15.9.2 Ibatis中的工厂模式示例391 15.10 总结392 第16章 Velocity工作原理解析394 16.1 Velocity总体架构395 16.2 JTree渲染过程解析398 16.2.1 #set语法402 16.2.2 Velocity的方法调用403 16.2.3 #if、#elseif和#else语法406 16.2.4 #foreach语法407 16.2.5 #parse语法409 16.3 事件处理机制410 16.4 常用优化技巧413 16.4.1 减少树的总节点数量413 16.4.2 减少渲染耗时的节点数量413 16.5 与JSP比较414 16.5.1 JSP渲染机制414 16.5.2 Velocity与JSP420 16.6 设计模式解析之合成模式420 16.6.1 合成模式的结构420 16.6.2 Velocity中合成模式的实现421 16.7 设计模式解析之解释器模式422 16.7.1 解释器模式的结构422 16.7.2 Velocity中解释器模式的实现423 16.8 总结423 第17章 Velocity优化实践424 17.1 现实存在的问题424 17.2 优化的理论基础425 17.2.1 程序语言的三角形结构425 17.2.2 数据结构减少抽象化426 17.2.3 简单的程序复杂化426 17.2.4 减少翻译的代价427 17.2.5 变的转化为不变427 17.3 一个高效的模板引擎的实现思路427 17.3.1 vm模板如何被编译429 17.3.2 方法调用的无反射优化436 17.3.3 字符输出改成字节输出439 17.4 优化的成果440 17.4.1 char转成byte440 17.4.2 无反射执行441 17.5 其他优化手段442 17.6 总结442

## &lt;&lt;深入分析Java Web技术内幕&gt;&gt;

## 章节摘录

版权页：插图：1.类加载器 在深入分析ClassLoader时我们详细分析了ClassLoader的工作机制，这里需要说明的是，每一个被JVM装载的类型都有一个对应的java.lang.Class类的实例来表示该类型，该实例可以唯一表示被JVM装载的class类，要求这个实例和其他类实例一样都存放在Java的堆中。

2.执行引擎 执行引擎是JVM的核心部分，执行引擎的作用就是解析JVM字节码指令，得到执行结果。在《Java虚拟机规范》中详细地定义了执行引擎遇到每条字节码指令时应该处理什么，并且应该得到什么结果。

但是并没有规定执行引擎应该如何或采取什么方式处理而得到这个结果。

因为执行引擎具体采取什么方式由JVM的实现厂家自己去实现，是直接解释执行还是采用JIT技术转成本地代码去执行，还是采用寄存器这个芯片模式去执行都可以。

所以执行引擎的具体实现有很大的发挥空间，如SUN的hotspot是基于栈的执行引擎，而Google的Dalvik是基于寄存器的执行引擎。

执行引擎也就是执行一条条代码的一个流程，而代码都是包含在方法体内的，所以执行引擎本质上就是执行一个个方法所串起来的流程，对应到操作系统中一个执行流程是一个Java进程还是一个Java线程呢？

很显然是后者，因为一个Java进程可以有多个同时执行的执行流程。

这样说来每一个Java线程就是一个执行引擎的实例，那么一个JVM实例中就会同时有多个执行引擎在工作，这些执行引擎有的在执行用户的程序，而有的在执行JVM内部的程序（如Java垃圾收集器）。

3.Java内存管理 执行引擎在执行一段程序时需要存储一些东西，如操作码需要的操作数，操作码的执行结果需要保存。

class类的字节码还有类的对象等这些信息都需要在执行引擎执行之前就准备好。

从图7—2中可以看出一个JVM实例会有一个方法区、Java堆、Java栈、PC寄存器和本地方法区。

其中方法区和Java堆是所有线程共享的，也就是可以被所有的执行引擎实例访问。

每一个新的执行引擎实例被创建时会为这个执行引擎创建一个Java栈和一个PC寄存器，如果当前正在执行一个Java方法，那么当前的这个Java栈中保存的是该线程中方法调用的状态，包括方法的参数、方法的局部变量、方法的返回值以及运算的中间结果等。

而PC寄存器会指向即将执行的下一条指令。



## <<深入分析Java Web技术内幕>>

### 编辑推荐

《深入分析Java Web技术内幕》不仅介绍这些技术和框架的工作原理，而且结合示例来讲解，通过通俗易懂的文字和丰富生动的配图，让读者充分并深入理解它们的内部工作原理，同时还结合了设计模式来介绍这些技术背后的架构思维。

## <<深入分析Java Web技术内幕>>

### 名人推荐

这是一本有关Java的书，里面讲述的大量基础知识对前端开发工程师也非常有帮助。

比如中文编码章节，作者以一个实践者的身份详细阐述了编码问题的方方面面。

总之，这是一本用心的书，是实践者的思考和总结。

国内目前很少看到这类书籍，强烈推荐从事Web开发人员购买阅读并实践之。

——王保平，开源前端类库KISSY、SeaJS作者 作者在淘宝做了很多Java Web方面的改造项目，在Java Web的相关技术上有深入的掌握，并积累了丰富的经验。

在这本书中作者不仅向读者展示了这类大改造项目所需的知识。

还展示了Java Web更为全景的技术知识体系，值得Java Web开发人员阅读。

——林昊，淘宝资深技术专家、China OSGi User Group总监 从第一次拜读相关内容开始，就可以感觉到作者并不是简单地讲述一个技术或者概念，他的分析和讲解十分深入，并且可以很好地聚焦读者的思路，尤其是在Java Web、Servlet规范及字符串处理方面，都有很优秀的内容。

在众多向developerWorks投稿的国内作者中，无论从文章的质量看，还是从内容的选题方向看，作者的文章都可称是上乘之作。

同时，他的多篇文章还得到了广大网站读者的好评，其访问量、评分及评论的数量均名列前茅。

——刘达



<<深入分析Java Web技术内幕>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>