

Understanding Nginx

Modules Development and Architecture Resolving

深入理解Nginx

模块开发与架构解析



陶辉 著



机械工业出版社
China Machine Press

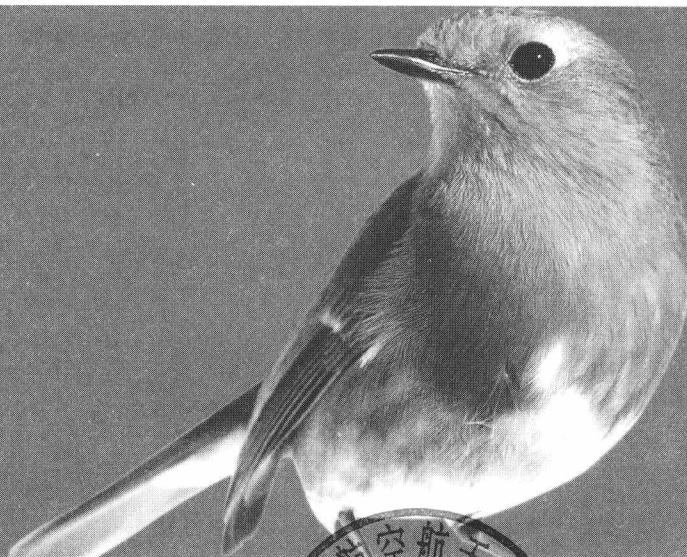
013028432

TP368.5
75

Understanding Nginx
Modules Development and Architecture Resolving

深入理解Nginx

模块开发与架构解析



陶辉 著



机械工业出版社
China Machine Press



北航

C1634909

TP368.5

75

204890810

图书在版编目 (CIP) 数据

深入理解 Nginx: 模块开发与架构解析 / 陶辉著. —北京: 机械工业出版社, 2013.3

ISBN 978-7-111-41478-0

I. 深… II. 陶… III. Web 服务器 IV. TP393.09

中国版本图书馆 CIP 数据核字 (2013) 第 033020 号

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问 北京市展达律师事务所

本书是阿里巴巴资深 Nginx 技术专家呕心沥血之作, 是作者多年的经验结晶, 也是目前市场上唯一一本通过还原 Nginx 设计思想, 剖析 Nginx 架构来帮助读者快速高效开发 HTTP 模块的图书。

本书首先通过介绍官方 Nginx 的基本用法和配置规则, 帮助读者了解一般 Nginx 模块的用法, 然后重点介绍如何开发 HTTP 模块 (含 HTTP 过滤模块) 来得到定制的 Nginx, 其中包括开发一个功能复杂的模块所需要了解的各种知识, 如 Nginx 的基础数据结构、配置项的解析、记录日志的工具以及 upstream、subrequest 的使用方法等。在此基础上, 综合 Nginx 框架代码分析 Nginx 的架构, 介绍其设计理念 and 技巧, 进一步帮助读者自由、有效地开发出功能丰富、性能一流的 Nginx 模块。

机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑: 马 超

藁城市京瑞印刷有限公司印刷

2013 年 4 月第 1 版第 1 次印刷

186mm×240 mm·36.5 印张

标准书号: ISBN 978-7-111-41478-0

定价: 89.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzjsj@hzbook.com



前言

为什么要写这本书

当我试图在产品的关键位置设计一个高性能 Web 服务器时，我选择使用成熟的 Nginx。选择它的理由为：首先，它对服务器性能上的挖掘已经达到了很高水平，它能尽量使不同的硬件（包括网卡、硬盘、不同的 CPU 核心）并发运行，同时软件中又没有阻塞进程使之睡眠的代码，从性能上来说，它可以挑战任何服务器。其次，完全基于事件驱动的服务器开发效率往往很不理想，它们要处理的事件过于底层化、细节化，这使得各功能模块无法聚焦于业务，最终产品的功能都较为单一，不会有丰富的可选功能。但 Nginx 却不然，由于它在软件架构上具有优秀的设计，使得 Nginx 完全由许多简单的模块构成，各模块（特别是 HTTP 模块）不用介入底层细节，在尽享分阶段、无阻塞的事件驱动架构下，可以专注于业务功能的实现，这样最终为 Nginx 带来了大量的官方、第三方的功能模块，使得功能同样强大的 Nginx 在产品核心位置上足以担当重任，经受住海量请求的考验。

当 Nginx 已有模块提供的功能不能完全实现我的所有业务需求时，我可以在 Nginx 的后端再搭建一个实现了缺失功能的非 Nginx 服务器，将 Nginx 无法实现的请求反向代理到这台服务器上处理。但这样也有一定的弊端，首先增大了处理请求的开销，其次后端服务器的设计仍然制约着总体性能（它依然需要解决 Nginx 解决过的无阻塞问题，那样才能像 Nginx 一样高效），这样做仅适用于对性能要求不高的场景。唯有开发一个实现了所需功能的自定义 Nginx 模块嵌入到 Nginx 代码中，才能让自己的业务像 Nginx 一样充分挖掘服务器的硬件资源，及时地响应百万级别的并发 TCP 连接。

当我在开发 Nginx 模块之前，试图在市面上找到一本关于 Nginx 模块开发的书籍（无论是中文还是英文）时却一无所获。我只能找到如何使用 Nginx 及其已有模块的书籍。为了开发 Nginx 模块，我只能通过阅读 Nginx 极度缺少注释的源代码，并分析各种官方 Nginx 模块来逐步还原其设计思想，反复尝试、验证着怎样的模块能够使用 Nginx 的基础架构，和丰富的跨平台工具方法，同时符合 Nginx 设计思想，使 Nginx 拥有媲美 Linux 内核的一流效率。这个过程耗费了我很多的精力，因此，我希望今后的 Nginx 使用者、开发者在遇到同样的

问题时，不至于还要很痛苦地阅读源代码来找到模块开发方法，而是简单地按照章节查阅本书，就可以快速找到怎样简单、高效地开发 Nginx 模块，把精力放在业务的实现上。这是我写这本书的第一个目的。

当我们产品中运行的 Nginx 出现了问题时，往往是通过找到错误的配置项、使用方式来解决的，这样也的确能够修复大部分问题。但是更深层次的问题，或者是使用场景比较偏僻，抑或是 Nginx 自身代码考虑得不够全面时，这些问题往往只能由那些花费大量精力研究 Nginx 源代码的工程师来解决。我写作本书的第二个目的是希望通过透彻地解析 Nginx 架构，帮助读者深入理解 Nginx，既能够正确地使用它，也能在它出现任何问题时找到根本原因，进而用最合适的方法修复或者回避问题。

Nginx 是一个优秀的事件驱动框架，虽然它在 HTTP 的处理上非常出色，但它绝不仅仅用于 Web 服务器。Nginx 非常适合开发在传输层以 TCP 对外提供服务的服务器程序。基于 Nginx 框架开发程序有 5 个优势：

- 1) Nginx 将网络、磁盘及定时器等异步事件的驱动都做了非常好的封装，基于它开发将可以忽略这些事件处理的细节。

- 2) Nginx 封装了许多平台无关的接口、容器，适用于跨平台开发。

- 3) 优秀的模块化设计，使得开发者可以轻易地复用各种已有的模块，其中既包括基本的读取配置、记录日志等模块，也包括处理请求的诸如 HTTP、mail 等高级功能模块。

- 4) Nginx 是作为服务器来设计其框架的，因此，它在服务器进程的管理上相当出色，基于它开发服务器程序可以轻松地实现程序的动态升级，子进程的监控、管理，配置项的动态修改生效等。

- 5) Nginx 充分考虑到各操作系统所擅长的“绝活”，能够使用特殊的系统调用更高效地完成任务时，绝不会去使用低效的通用接口。尤其对于 Linux 操作系统，Nginx 不遗余力地做了大量优化。

当我们期望编写一款能够以低负载处理高并发请求并且主要处理基于 TCP 的服务器程序时，推荐选择 Nginx，它可能会带给我们意外的惊喜。这本书的第三部分，将通过分析 Nginx 的内部架构，帮助读者了解怎样基于 Nginx 开发高效的 TCP 服务器程序：通过开发一种新的模块类型，实现一种新的功能框架来提供极佳的扩展性，使得功能子模块仅关注于业务的开发，忽视底层事件的处理。这是我写作本书的第三个目的。

除了这 3 个主要目的外，我还希望通过这本书向大家展示 Nginx 在服务器开发上的许多巧妙设计，它们或在抽象设计上精妙，或通过操作系统精确、节省地使用硬件资源，这些细节之处的设计都体现了 Igor Sysoev 的不凡功底。即使我们完全不使用 Nginx，学习这些技巧也将有助于我们服务器编程水平的提升。

读者对象

本书适合以下读者阅读。

- 对 Nginx 及如何将它搭建成一个高性能的 Web 服务器感兴趣的读者。
- 希望通过开发特定的 HTTP 模块实现高性能 Web 服务器的读者。
- 希望了解 Nginx 的架构设计，学习怎样充分使用服务器上的硬件资源的读者。
- 了解如何快速定位、修复 Nginx 中深层次 Bug 的读者。
- 希望利用 Nginx 提供的框架，设计出任何基于 TCP 的、无阻塞的、易于扩展的服务器的读者。

背景知识

如果仅希望了解怎样使用已有的 Nginx 功能搭建服务器，那么阅读本书不需要什么先决条件。但如果希望通过阅读本书的第二、第三部分，来学习 Nginx 的模块开发和架构设计技巧，则必须了解 C 语言的基本语法。在阅读本书第三部分时，需要读者对 TCP 有一个基本的了解，同时对 Linux 操作系统也应该有简单的了解。

如何阅读本书

我很希望将本书写成一本“step by step”式（循序渐进式）的书籍，因为这样最能节省读者的时间，然而，由于 3 个主要写作目的想解决的问题都不是那么简单，所以这本书只能做一个折中的处理。

在第一部分的前两章中，将只探讨如何使用 Nginx 这一问题。阅读这一部分的读者不需要了解 C 语言，就可以学习如何部署 Nginx，学习如何向其中添加各种官方、第三方的功能模块，如何通过修改配置文件来更改 Nginx 及各模块的功能，如何修改 Linux 操作系统上的参数来优化服务器性能，最终向用户提供企业级的 Web 服务器。这一部分介绍配置项的方式，更偏重于带领对 Nginx 还比较陌生的读者熟悉它，通过了解几个基本 Nginx 模块的配置修改方式，进而使读者可以通过查询官网、第三方网站来了解如何使用所有 Nginx 模块的用法。

在第二部分的第 3 章～第 7 章中，都是以例子来介绍 HTTP 模块的开发方式的，这里有些接近于“step by step”的学习方式，我在写作这一部分时，会通过循序渐进的方式使读者能够快速上手，同时会穿插着介绍其常见用法的基本原理。

在第三部分，将开始介绍 Nginx 的完整框架，阅读到这里时将会了解第二部分中 HTTP 模块为何以此种方式开发，同时将可以轻易地开发出 Nginx 模块。这一部分并不仅仅满足于

阐述 Nginx 架构，而是会探讨其为何如此设计，只有这样才能抛开 HTTP 框架、邮件代理框架，实现一种新的业务框架、一种新的模块类型。

对于 Nginx 的使用还不熟悉的读者应当从第 1 章开始学习，前两章将帮助你快速了解 Nginx。

使用过 Nginx，但对如何开发 Nginx 的 HTTP 模块不太了解的读者可以直接从第 3 章开始学习，在这一章阅读完后，即可编写一个功能大致完整的 HTTP 模块。然而，编写企业级的模块必须阅读完第 4 章才能做到，这一章将会介绍编写产品线上服务器程序时必备的 3 个手段。第 5 章举例说明了两种编写复杂 HTTP 模块的方式，在第三部分会对这两种方式有进一步的说明。第 6 章介绍一种特殊的 HTTP 模块——HTTP 过滤模块的编写方法。第 7 章探讨基础容器的用法，这同样是复杂模块的必备工具。

如果读者对于普通 HTTP 模块的编写已经很熟悉，想深入地实现更为复杂的 HTTP 模块，或者想了解邮件代理服务器的设计与实现，或者希望编写一种新的处理其他协议的模块，或者仅仅想了解 Nginx 的架构设计，都可以直接从第 8 章开始学习，这一章会从整体上系统介绍 Nginx 的模块式设计。第 9 章的事件框架是 Nginx 处理 TCP 的基础，这一章无法跳过。阅读第 8、第 9 章时可能会遇到许多第 7 章介绍过的容器，这时可以回到第 7 章查询其用法和意义。第 10 章~第 12 章介绍 HTTP 框架，通过这 3 章的学习会对 HTTP 模块的开发有深入的了解，同时可以学习 HTTP 框架的优秀设计。第 13 章简单地介绍了邮件代理服务器的设计，它近似于简化版的 HTTP 框架。第 14 章介绍了进程间同步的工具。

为了不让读者陷入代码的“汪洋大海”中，在本书中大量使用了图表，这样可以使读者快速、大体地了解流程和原理。关键地方会直接给出代码，并添加注释加以说明。希望这种方式能够帮助读者减少阅读花费的时间，更快、更好地把握 Nginx，同时深入到细节中。

在本书开始写作时，由于 Nginx 的最新稳定版本是 1.0.14，所以本书是基于此版本来编写的。截止到本书编写完成时，Nginx 的稳定版本已经上升到了 1.2.4。但这不会对本书的阅读造成困扰，因为本书主要是在介绍 Nginx 的基本框架代码，以及怎样使用这些框架代码开发新的 Nginx 模块，而不是介绍 Nginx 的某些功能。在这些基本框架代码中，Nginx 一般不会做任何改变，否则已有的大量 Nginx 模块将无法工作，这种损失也是不可承受的。而且，Nginx 框架为具体的功能模块提供了足够的灵活性，修改功能时很少需要修改框架代码。

Nginx 是跨平台的服务器，然而这本书将只针对最常见的 Linux 操作系统进行分析，这样做一方面是篇幅所限，另一方面则是本书的写作目的主要在于告诉读者如何基于 Nginx 编写代码，而不是怎样在一个具体的操作系统上修改配置来使用 Nginx。因此，即使本书以 Linux 系统为代表讲述 Nginx，也不会影响使用其他操作系统的读者阅读，因为操作系统的差别对阅读本书的影响实在是非常小。

勘误和支持

由于作者的水平有限，加之编写的时间也很仓促，书中难免会出现一些错误或者不准确的地方，恳请读者批评指正。为此，我特意创建了一个在线支持与应急方案的二级站点：<http://nginx.weebly.com>。读者可以将书中的错误发布在 Bug 勘误表页面中，同时如果你遇到任何问题，也可以访问 Q&A 页面，我将尽量在线上为读者提供最满意的解答。书中的全部源文件都将发布在这个网站上，我也会将相应的功能更新及时发布出来。如果你有更多的宝贵意见，也欢迎你发送邮件至我的邮箱 russelltao@foxmail.com，期待能够听到读者的真挚反馈。

致谢

我首先要感谢 Igor Sysoev，他在 Nginx 设计上展现的功力令人折服，正是他的工作成果才让本书的诞生有了意义。

lisa 是机械工业出版社华章公司的优秀编辑，非常值得信任。在这半年的写作过程中，她花费了很多时间、精力来阅读我的书稿，指出了许多文字和格式上的错误，她提出的建议大大提高了本书的可读性。

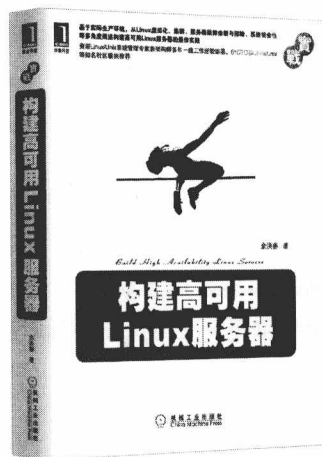
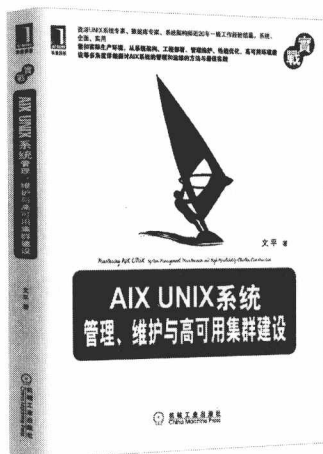
在这半年时间内，一边工作一边写作给我带来了很大的压力，所以我要感谢我的父母在生活上对我无微不至的照顾，使我可以全力投入到写作中。繁忙的工作之余，写作又占用了休息时间的绝大部分，感谢我的太太对我的体谅和鼓励，让我始终以高昂的斗志投入到本书的写作中。

感谢我工作中的同事们，正是在与他们一起工作的日子里，我才不断地对技术有新的感悟；正是那些充满激情的岁月，才使得我越来越热爱服务器技术的开发。

谨以此书，献给我最亲爱的家人，以及众多热爱 Nginx 的朋友。

陶 辉

推荐阅读



AIX UNIX系统：管理、维护与高可用集群建设

作者：文平 ISBN：978-7-111-35951-7 定价：79.00元

资深UNIX系统专家、数据库专家、

系统架构师近20年一线工作经验结晶，系统、全面、实用

紧扣实际生产环境，从系统架构、工程部署、管理维护、性能优化、高可用环境建设等多角度详细探讨AIX系统的管理和运维的方法与最佳实践

构建高可用Linux服务器

作者：余洪春 ISBN：978-7-111-36055-1 定价：79.00元

基于实际生产环境，从Linux虚拟化、集群、服务器故障诊断与排除、系统安全性等多角度阐述构建高可用Linux服务器的最佳实践

资深Linux/Unix系统管理专家兼架构师多年一线工作经验结晶，51CTO和ChinaUnix等知名社区联袂推荐



北航

C1634909



前 言

第一部分 Nginx 能帮我们做什么

第 1 章 研究 Nginx 前的准备工作 / 2

- 1.1 Nginx 是什么 / 2
- 1.2 为什么选择 Nginx / 4
- 1.3 准备工作 / 7
 - 1.3.1 Linux 操作系统 / 7
 - 1.3.2 使用 Nginx 的必备软件 / 7
 - 1.3.3 磁盘目录 / 8
 - 1.3.4 Linux 内核参数的优化 / 9
 - 1.3.5 获取 Nginx 源码 / 11
- 1.4 编译安装 Nginx / 11
- 1.5 configure 详解 / 11
 - 1.5.1 configure 的命令参数 / 12
 - 1.5.2 configure 执行流程 / 18
 - 1.5.3 configure 生成的文件 / 22
- 1.6 Nginx 的命令行控制 / 24
- 1.7 小结 / 27

第 2 章 Nginx 的配置 / 28

- 2.1 运行中的 Nginx 进程间的关系 / 28
- 2.2 Nginx 配置的通用语法 / 31
 - 2.2.1 块配置项 / 31
 - 2.2.2 配置项的语法格式 / 32
 - 2.2.3 配置项的注释 / 33
 - 2.2.4 配置项的单位 / 33
 - 2.2.5 在配置中使用变量 / 33
- 2.3 Nginx 服务的基本配置 / 34
 - 2.3.1 用于调试进程和定位问题的配置项 / 34
 - 2.3.2 正常运行的配置项 / 36
 - 2.3.3 优化性能的配置项 / 38
 - 2.3.4 事件类配置项 / 39
- 2.4 用 HTTP 核心模块配置一个静态 Web 服务器 / 41
 - 2.4.1 虚拟主机与请求的分发 / 42
 - 2.4.2 文件路径的定义 / 45
 - 2.4.3 内存及磁盘资源的分配 / 48
 - 2.4.4 网络连接的设置 / 50
 - 2.4.5 MIME 类型的设置 / 53
 - 2.4.6 对客户端请求的限制 / 54
 - 2.4.7 文件操作的优化 / 55
 - 2.4.8 对客户端请求的特殊处理 / 57
 - 2.4.9 ngx_http_core_module 模块提供的变量 / 59
- 2.5 用 HTTP proxy module 配置一个反向代理服务器 / 60
 - 2.5.1 负载均衡的基本配置 / 62
 - 2.5.2 反向代理的基本配置 / 64
- 2.6 小结 / 68

第二部分 如何编写 HTTP 模块

第 3 章 开发一个简单的 HTTP 模块 / 70

- 3.1 如何调用 HTTP 模块 / 70
- 3.2 准备工作 / 72

- 3.2.1 整型的封装 / 72
- 3.2.2 ngx_str_t 数据结构 / 73
- 3.2.3 ngx_list_t 数据结构 / 73
- 3.2.4 ngx_table_elt_t 数据结构 / 77
- 3.2.5 ngx_buf_t 数据结构 / 77
- 3.2.6 ngx_chain_t 数据结构 / 79
- 3.3 如何将自己的 HTTP 模块编译进 Nginx / 79
 - 3.3.1 config 文件的写法 / 80
 - 3.3.2 利用 configure 脚本将定制的模块加入到 Nginx 中 / 80
 - 3.3.3 直接修改 Makefile 文件 / 84
- 3.4 HTTP 模块的数据结构 / 85
- 3.5 定义自己的 HTTP 模块 / 88
- 3.6 处理用户请求 / 92
 - 3.6.1 处理方法的返回值 / 92
 - 3.6.2 获取 URI 和参数 / 95
 - 3.6.3 获取 HTTP 头部 / 98
 - 3.6.4 获取 HTTP 包体 / 101
- 3.7 发送响应 / 102
 - 3.7.1 发送 HTTP 头部 / 102
 - 3.7.2 将内存中的字符串作为包体发送 / 104
 - 3.7.3 经典的“Hello World”示例 / 106
- 3.8 将磁盘文件作为包体发送 / 107
 - 3.8.1 如何发送磁盘中的文件 / 107
 - 3.8.2 清理文件句柄 / 110
 - 3.8.3 支持用户多线程下载和断点续传 / 111
- 3.9 用 C++ 语言编写 HTTP 模块 / 112
 - 3.9.1 编译方式的修改 / 112
 - 3.9.2 程序中的符号转换 / 114
- 3.10 小结 / 114

第 4 章 配置、error 日志和请求上下文 / 115

- 4.1 http 配置项的使用场景 / 115
- 4.2 怎样使用 http 配置 / 117
 - 4.2.1 分配用于保存配置参数的数据结构 / 117

- 4.2.2 设定配置项的解析方式 / 119
- 4.2.3 使用 14 种预设方法解析配置项 / 125
- 4.2.4 自定义配置项处理方法 / 136
- 4.2.5 合并配置项 / 137
- 4.3 HTTP 配置模型 / 140
 - 4.3.1 解析 HTTP 配置的流程 / 141
 - 4.3.2 HTTP 配置模型的内存布局 / 144
 - 4.3.3 如何合并配置项 / 147
 - 4.3.4 预配置项处理方法的工作原理 / 149
- 4.4 error 日志的用法 / 150
- 4.5 请求的上下文 / 155
 - 4.5.1 上下文与全异步 Web 服务器的关系 / 155
 - 4.5.2 如何使用 HTTP 上下文 / 156
 - 4.5.3 HTTP 框架如何维护上下文结构 / 157
- 4.6 小结 / 158

第 5 章 访问第三方服务 / 159

- 5.1 upstream 的使用方式 / 160
 - 5.1.1 ngx_http_upstream_t 结构体 / 163
 - 5.1.2 设置 upstream 的限制性参数 / 164
 - 5.1.3 设置需要访问的第三方服务器地址 / 165
 - 5.1.4 设置回调方法 / 166
 - 5.1.5 如何启动 upstream 机制 / 166
- 5.2 回调方法的执行场景 / 167
 - 5.2.1 create_request 回调方法 / 167
 - 5.2.2 reinit_request 回调方法 / 169
 - 5.2.3 finalize_request 回调方法 / 170
 - 5.2.4 process_header 回调方法 / 171
 - 5.2.5 rewrite_redirect 回调方法 / 172
 - 5.2.6 input_filter_init 与 input_filter 回调方法 / 172
- 5.3 使用 upstream 的示例 / 173
 - 5.3.1 upstream 的各种配置参数 / 174
 - 5.3.2 请求上下文 / 175
 - 5.3.3 在 create_request 方法中构造请求 / 176

- 5.3.4 在 `process_header` 方法中解析包头 / 177
- 5.3.5 在 `finalize_request` 方法中释放资源 / 180
- 5.3.6 在 `ngx_http_mytest_handler` 方法中启动 upstream / 181
- 5.4 subrequest 的使用方式 / 183
 - 5.4.1 配置子请求的处理方式 / 183
 - 5.4.2 实现子请求处理完毕时的回调方法 / 184
 - 5.4.3 处理父请求被重新激活后的回调方法 / 185
 - 5.4.4 启动 subrequest 子请求 / 185
- 5.5 subrequest 执行过程中的主要场景 / 186
 - 5.5.1 如何启动 subrequest / 186
 - 5.5.2 如何转发多个子请求的响应包体 / 188
 - 5.5.3 子请求如何激活父请求 / 192
- 5.6 subrequest 使用的例子 / 193
 - 5.6.1 配置文件中子请求的设置 / 194
 - 5.6.2 请求上下文 / 194
 - 5.6.3 子请求结束时的处理方法 / 195
 - 5.6.4 父请求的回调方法 / 196
 - 5.6.5 启动 subrequest / 197
- 5.7 小结 / 198

第 6 章 开发一个简单的 HTTP 过滤模块 / 199

- 6.1 过滤模块的意义 / 199
- 6.2 过滤模块的调用顺序 / 200
 - 6.2.1 过滤链表是如何构成的 / 200
 - 6.2.2 过滤链表的顺序 / 203
 - 6.2.3 官方默认 HTTP 过滤模块的功能简介 / 204
- 6.3 HTTP 过滤模块的开发步骤 / 206
- 6.4 HTTP 过滤模块的简单例子 / 207
 - 6.4.1 如何编写 config 文件 / 208
 - 6.4.2 配置项和上下文 / 208
 - 6.4.3 定义 HTTP 过滤模块 / 210
 - 6.4.4 初始化 HTTP 过滤模块 / 211
 - 6.4.5 处理请求中的 HTTP 头部 / 212
 - 6.4.6 处理请求中的 HTTP 包体 / 213
- 6.5 小结 / 214

第7章 Nginx 提供的高级数据结构 / 215

- 7.1 Nginx 提供的高级数据结构概述 / 215
- 7.2 ngx_queue_t 双向链表 / 217
 - 7.2.1 为什么设计 ngx_queue_t 双向链表 / 217
 - 7.2.2 双向链表的使用方法 / 217
 - 7.2.3 使用双向链表排序的例子 / 219
 - 7.2.4 双向链表是如何实现的 / 221
- 7.3 ngx_array_t 动态数组 / 222
 - 7.3.1 为什么设计 ngx_array_t 动态数组 / 223
 - 7.3.2 动态数组的使用方法 / 223
 - 7.3.3 使用动态数组的例子 / 225
 - 7.3.4 动态数组的扩容方式 / 226
- 7.4 ngx_list_t 单向链表 / 226
- 7.5 ngx_rbtree_t 红黑树 / 227
 - 7.5.1 为什么设计 ngx_rbtree_t 红黑树 / 227
 - 7.5.2 红黑树的特性 / 228
 - 7.5.3 红黑树的使用方法 / 230
 - 7.5.4 使用红黑树的简单例子 / 233
 - 7.5.5 如何自定义添加成员方法 / 234
- 7.6 ngx_radix_tree_t 基数树 / 236
 - 7.6.1 ngx_radix_tree_t 基数树的原理 / 236
 - 7.6.2 基数树的使用方法 / 238
 - 7.6.3 使用基数树的例子 / 239
- 7.7 支持通配符的散列表 / 240
 - 7.7.1 ngx_hash_t 基本散列表 / 240
 - 7.7.2 支持通配符的散列表 / 243
 - 7.7.3 带通配符散列表的使用例子 / 250
- 7.8 小结 / 254

第三部分 深入 Nginx

第8章 Nginx 基础架构 / 256

- 8.1 Web 服务器设计中的关键约束 / 256

- 8.2 Nginx 的架构设计 / 259
 - 8.2.1 优秀的模块化设计 / 259
 - 8.2.2 事件驱动架构 / 263
 - 8.2.3 请求的多阶段异步处理 / 264
 - 8.2.4 管理进程、多工作进程设计 / 267
 - 8.2.5 平台无关的代码实现 / 268
 - 8.2.6 内存池的设计 / 268
 - 8.2.7 使用统一管道过滤器模式的 HTTP 过滤模块 / 268
 - 8.2.8 其他一些用户模块 / 269
- 8.3 Nginx 框架中的核心结构体 ngx_cycle_t / 269
 - 8.3.1 ngx_listening_t 结构体 / 269
 - 8.3.2 ngx_cycle_t 结构体 / 271
 - 8.3.3 ngx_cycle_t 支持的方法 / 273
- 8.4 Nginx 启动时框架的处理流程 / 275
- 8.5 worker 进程是如何工作的 / 278
- 8.6 master 进程是如何工作的 / 281
- 8.7 小结 / 286

第 9 章 事件模块 / 287

- 9.1 事件处理框架概述 / 287
- 9.2 Nginx 事件的定义 / 290
- 9.3 Nginx 连接的定义 / 293
 - 9.3.1 被动连接 / 294
 - 9.3.2 主动连接 / 297
 - 9.3.3 ngx_connection_t 连接池 / 298
- 9.4 ngx_events_module 核心模块 / 300
 - 9.4.1 如何管理所有事件模块的配置项 / 301
 - 9.4.2 管理事件模块 / 303
- 9.5 ngx_event_core_module 事件模块 / 305
- 9.6 epoll 事件驱动模块 / 310
 - 9.6.1 epoll 的原理和用法 / 311
 - 9.6.2 如何使用 epoll / 313
 - 9.6.3 ngx_epoll_module 模块的实现 / 315
- 9.7 定时器事件 / 323

- 9.7.1 缓存时间的管理 / 324
- 9.7.2 缓存时间的精度 / 326
- 9.7.3 定时器的实现 / 327
- 9.8 事件驱动框架的处理流程 / 328
 - 9.8.1 如何建立新连接 / 329
 - 9.8.2 如何解决“惊群”问题 / 330
 - 9.8.3 如何实现负载均衡 / 333
 - 9.8.4 post 事件队列 / 334
 - 9.8.5 ngx_process_events_and_timers 流程 / 335
- 9.9 文件的异步 I/O / 338
 - 9.9.1 Linux 内核提供的文件异步 I/O / 339
 - 9.9.2 ngx_epoll_module 模块中实现的针对文件的异步 I/O / 342
- 9.10 小结 / 346

第 10 章 HTTP 框架的初始化 / 347

- 10.1 HTTP 框架概述 / 348
- 10.2 管理 HTTP 模块的配置项 / 351
 - 10.2.1 管理 main 级别下的配置项 / 352
 - 10.2.2 管理 server 级别下的配置项 / 354
 - 10.2.3 管理 location 级别下的配置项 / 357
 - 10.2.4 不同级别配置项的合并 / 362
- 10.3 监听端口的管理 / 367
- 10.4 server 的快速检索 / 369
- 10.5 location 的快速检索 / 371
- 10.6 HTTP 请求的 11 个处理阶段 / 372
 - 10.6.1 HTTP 处理阶段的普适规则 / 374
 - 10.6.2 NGX_HTTP_POST_READ_PHASE 阶段 / 376
 - 10.6.3 NGX_HTTP_SERVER_REWRITE_PHASE 阶段 / 378
 - 10.6.4 NGX_HTTP_FIND_CONFIG_PHASE 阶段 / 379
 - 10.6.5 NGX_HTTP_REWRITE_PHASE 阶段 / 379
 - 10.6.6 NGX_HTTP_POST_REWRITE_PHASE 阶段 / 379
 - 10.6.7 NGX_HTTP_PREACCESS_PHASE 阶段 / 379
 - 10.6.8 NGX_HTTP_ACCESS_PHASE 阶段 / 380
 - 10.6.9 NGX_HTTP_POST_ACCESS_PHASE 阶段 / 380