在Web Server开发的天空自由翱翔

Broadview
www.broadview.com.cn

Nginx 模块开发指南

使用C++11和Boost程序库

罗剑锋 著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Nginx 模块开发指南

使用C++11和Boost程序库

罗剑锋 著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

Nginx 是由俄罗斯工程师 Igor Sysoev 开发的一个高性能 Web 服务器，运行效率远超传统的 Apache、Tomcat，是世界第二大 Web 服务器，被国内外诸多顶级互联网公司采用。

Nginx 的一个突出特点是其灵活优秀的模块化架构，可以在不修改核心的前提下增加任意功能，自 2004 年发布至今，已经拥有百余个官方及非官方的功能模块（如 fastcgi、memcached、mysql 等），使得 Nginx 成长为了一个近乎“全能”的服务器软件。

Nginx 以纯 C 语言实现，开发扩展功能模块也大多使用 C 语言，但由于 C 语言固有的过程式特性，编写、调试代码都较麻烦——特别是对于 Nginx 的初学者。本书深入源码，详细解析了模块体系、配置指令、HTTP 框架等 Nginx 核心运行机制，并在此基础上讲解如何使用 C++和 Boost 程序库来开发 Nginx 模块，充分利用现代 C++里的大量新特性和库组件，让 Nginx 的模块开发变得更加便捷、轻松和愉快。

本书结构严谨、脉络清晰、论述精确、详略得当，值得广大软件开发工程师、系统运维工程师和编程爱好者拥有。

图书在版编目（CIP）数据

Nginx 模块开发指南：使用 C++11 和 Boost 程序库 / 罗剑锋著. —北京：电子工业出版社，2015.11
ISBN 978-7-121-27294-3

Ⅰ. ①N… Ⅱ. ①罗… Ⅲ. ①互联网络－网络服务器－程序设计－指南②C 语言－程序设计－指南
Ⅳ.①TP368.5-62②TP312-62

中国版本图书馆 CIP 数据核字(2015)第 227682 号

策划编辑：孙学瑛
责任编辑：安　娜
印　　刷：三河市双峰印刷装订有限公司
装　　订：三河市双峰印刷装订有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787×980　1/16　印张：23.25　　字数：438 千字
版　　次：2015 年 11 月第 1 版
印　　次：2015 年 11 月第 1 次印刷
印　　数：3000 册　　定价：79.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前言

缘起

最早接触 Nginx 大概是在 2011 年，面对着一个全新的 Web 服务器，和大多数人一样最初我也是一片茫然，能找到的参考资料十分有限，安装、配置、运行几乎都是“摸着石头过河”，犯过许多低级错误。

随着对 Nginx 逐渐熟悉，它的高并发处理能力给我留下了深刻的印象，作为一个开源软件的爱好者，很自然地想要探究一下它的内部工作原理。我由此开始了对 Nginx 源码的钻研之路，中间经过了很多的艰辛曲折，走过了不少的弯路。

我最常用的工作语言是 C++，所以在阅读 Nginx 源码时也总以 C++的面向对象方式来思考和理解，以对象作为切入点记笔记、画 UML：从最简单的 ngx_str_t、ngx_array_t 入手，然后到 ngx_request_t、ngx_upstream_t 等复杂的结构，再围绕着这些对象研究相关的功能函数和处理流程，梳理代码逻辑的同时也摸索着使用 C++编写 Nginx 模块的方法，逐渐积累了一些用起来颇为顺手的小工具——当然还是比较初级的形式。

去年年中的某个时刻，我被调到了新的工作岗位，需要重度使用 Nginx 开发，这让我以前的零散积累终于有了用武之地，那段时间里使用 C++陆续做了很多东西，也借着机会重新优化了原有的工具代码。

繁忙的工作之余，我有了种进一步整理经验的迫切感，因为只有系统完整地分享这些知识，才能让更多的人使用 C++来基于 Nginx 二次开发，让 Nginx 更好地为网络世界服务，于是今年年初终于动起了写作的念头。

经过大半年的努力，现在这本书终于呈现在了读者面前，结构上基本反映了我学习研究

Nginx 时的心路历程，从最初的“一无所知”起步，慢慢地深入到定制开发模块的层次，希望能与读者“心有戚戚焉”。

Nginx 随感

毫无疑问，Nginx 是目前这个星球上所能获得的最强劲的 Web 服务器（没有之一），同时也是目前最成熟、最优秀的 TCP/HTTP 服务器开发框架。

Nginx 资源消耗低，并发处理性能高，配置灵活，能够连接 CGI、PHP、MySQL、Memcached 等多种后端，还有着出色的负载均衡能力，可以整合封装各种 service，构建稳定高效的服务。如今 Nginx 已经成为了网站架构里不可或缺的关键组件，广泛应用于国内外许多大型 IT 企业内部。每一个繁忙的网站背后，可能都有 Nginx 默默工作的身影。

在 Nginx 出现之前，使用 C/C++开发 Web 服务器是项比较“痛苦”的工作，虽然有很多网络程序库可以使用（例如 ACE、asio、libevent、thrift 等），但它们通常只关注较底层的基础功能实现，离成熟的“框架”相距甚远，不仅开发过程烦琐低效，而且程序员还必须要处理配置管理、进程间通信、协议解析等许多 Web 服务之外的其他事情，才能开发出一个较为完善的服务器程序。但即使开发出了这样的服务器，通常性能上也很难得到保证，会受到程序库和开发者水平等因素的限制——很长一段时间里，C/C++在 Web 服务器领域都没有大展拳脚的机会。

Nginx 的横空出世为 Web 服务器开辟了一个崭新的天地，它搭建了一个高性能的服务器开发框架，而且是一个完整的、全功能的服务器。模块化的架构设计很好地分离了底层支撑模块和上层逻辑模块，底层模块处理了配置、并发等服务器的外围功能，核心支撑模块定义了主体的 TCP/HTTP 处理框架。开发者可以把大部分精力集中在上层的业务功能实现上，再也不必去为其他杂事而分心，提高了软件的开发效率。

在 Nginx 框架里 C/C++程序员可以尽情发挥自己的专长，充分利用 Nginx 无阻塞处理的优势，打造出高质量的 Web 应用服务器，与其他系统一较高下。

Nginx 和 C/C++

Igor Sysoev 选择用 C 语言（准确地说是 ANSI C）来实现 Nginx 肯定是经过了认真的考虑。

作为与 UNIX 一同诞生的编程语言，C 语言一直是系统级编程的首选。和其他高级语言相比，它简单可靠，更接近计算机底层硬件，运行效率更高。指针更是 C 语言的一大特色，善用

指针能够完成许多其他语言无法完成的工作。

以C语言实现的Nginx没有“虚拟机”的成本，省略了不必要的中间环节，直接操纵计算机硬件，从根本上提高了Web服务器的处理能力。虽然C语言不直接支持面向对象，但Nginx灵活运用了指针，采用结构体+函数指针的形式，达到了同样的效果，从而使软件拥有了良好的结构。

C++是仅次于C的系统级编程语言，在兼容C的同时又增加了类、异常、模板等新特性，还支持面向对象、泛型、函数式、模板元等多种编程范式，可以说是计算机语言里的一个“庞然大物”。C++的特性很多，有的也很好用，但总体上的确比较复杂，易学难精，容易被误用和滥用，导致低效、难维护的代码，我想这可能是Igor Sysoev放弃使用C++的一个重要原因。

另一个可能的原因是C语言本身已经非常稳定，几十年来没有太大的变动，各个系统里都支持得非常好。而C++在1998年才有了第一个标准，而且现在还在发展之中，语言特性还不够稳定（例如export、register等曾经的关键字在C++11里就已经被废弃），许多编译器对C++的支持程度也有差异，这与Nginx的高可移植性目标明显不符。

但C++毕竟还是有很多的优点，类可以更好地封装信息、异常简化了错误处理、模板能够在编译期执行类型计算。在C++11标准颁布之后C++更是几乎变成了一门“全新”的语言，auto/decltype/nullptr/noexcept等新关键字增强了语言的描述能力，标准库也扩充了相当多的组件，易用性和稳定性都大大提升。

在Nginx里使用C++时要对C++的长处和不足有清醒的认识，避免多层次继承、虚函数等影响效率的编程范式，只使用经过充分验证的、能够切实提高开发效率和性能的语言特性和库，避免华而不实的技术炫耀，尽量做到像Nginx源码那样质朴踏实。只有这样，才能够发挥出1+1>2的作用，让Nginx从C++中得到更进一步的发展动力。

致谢

首先当然要感谢Nginx的作者Igor Sysoev，没有他就不会有如此优秀的Web服务器，也就不会有本书的诞生。

亲情永远是人生命中最值得珍惜的部分，我要感谢父母多年来的养育之恩和“后勤”工作，感谢妻子在生活中的陪伴（但因为频繁的加班常常缺少相处的时间），感谢即将上小学的女儿，愿你们能够永远幸福快乐。

我还要感谢工作中共同奋斗拼搏的诸多同事，从他们那里我获取到了很多的经验和知识，度过了很多难忘的岁月，和他们在一起让工作超越了谋生的手段变成了一种享受。

最后感谢读者选择本书，希望读者能够在阅读过程中有所收获，在 Nginx 开发过程中获得乐趣。

您的朋友 罗剑锋

2015 年 8 月 18 日 于 北京 酒仙桥

目录

第 0 章 导读 1

0.1 关于本书 1
0.2 读者对象 2
0.3 读者要求 3
0.4 运行环境 3
0.5 本书的结构 4
0.6 如何阅读本书 6
0.7 本书的源码 6

第 1 章 Nginx 入门 7

1.1 关于 Nginx 7
1.1.1 历史 8
1.1.2 特点 8
1.1.3 进程模型 9
1.1.4 版本 10
1.2 安装 Nginx 11
1.2.1 准备工作 11
1.2.2 快速安装 11
1.2.3 运行命令 12
1.2.4 验证安装 13
1.2.5 定制安装 14
1.3 配置 Nginx 15
1.3.1 配置文件格式 16
1.3.2 进程配置 17
1.3.3 运行日志配置 18
1.3.4 http 配置 18
1.3.5 server 配置 19
1.3.6 location 配置 20
1.3.7 文件访问配置 21
1.3.8 upstream 配置 22
1.3.9 变量 22
1.4 总结 24

第 2 章 Nginx 开发准备 25

2.1 开发环境 25
2.1.1 C++标准 25
2.1.2 Boost 程序库 26
2.2 Nginx 的目录结构 26
2.3 Nginx 源码的特点 27
2.3.1 代码风格 28
2.3.2 代码优化 28
2.3.3 面向对象思想 28
2.4 在 Nginx 里使用 C++ 29
2.4.1 实现原则 29
2.4.2 代码风格 29
2.4.3 编译脚本 30

2.5 基本的C++包装类 32
2.5.1 类定义 32
2.5.2 构造和析构 33
2.5.3 成员函数 33
2.6 总结 34

第3章 Nginx基础设施 35

3.1 头文件 35
3.2 整数类型 36
3.2.1 标准整数类型 36
3.2.2 自定义整数类型 36
3.2.3 无效值 37
3.2.4 C++封装 38
3.3 错误处理 41
3.3.1 错误码定义 41
3.3.2 C++异常 41
3.4 内存池 44
3.4.1 结构定义 44
3.4.2 操作函数 45
3.4.3 C++封装 45
3.4.4 清理机制 48
3.4.5 C++内存分配器 50
3.5 字符串 52
3.5.1 结构定义 52
3.5.2 操作函数 53
3.5.3 C++封装 55
3.6 时间与日期 58
3.6.1 时间结构定义 58
3.6.2 时间操作函数 58
3.6.3 日期结构定义 59
3.6.4 日期操作函数 60
3.6.5 C++封装时间 61
3.6.6 C++封装日期 62
3.7 运行日志 64
3.7.1 结构定义 65
3.7.2 操作函数 65
3.7.3 C++封装 66
3.8 总结 68

第4章 Nginx高级数据结构 71

4.1 动态数组 71
4.1.1 结构定义 71
4.1.2 操作函数 73
4.1.3 C++封装 73
4.2 单向链表 76
4.2.1 结构定义 77
4.2.2 操作函数 78
4.2.3 C++迭代器 79
4.2.4 C++封装链表 81
4.3 双端队列 84
4.3.1 结构定义 84
4.3.2 操作函数 85
4.3.3 C++节点 87
4.3.4 C++迭代器 89
4.3.5 C++封装队列 90
4.4 缓冲区 95
4.4.1 结构定义 95
4.4.2 操作函数 97
4.4.3 C++封装 98
4.5 数据块链 100
4.5.1 结构定义 100
4.5.2 操作函数 101
4.5.3 C++节点 101
4.5.4 C++迭代器 103
4.5.5 C++封装数据块链 105
4.6 键值对 107
4.6.1 简单键值对 107
4.6.2 散列表键值对 107

4.7 总结 108

第 5 章 Nginx 开发综述 111

5.1 最简单的 Nginx 模块 111
5.1.1 模块设计 112
5.1.2 配置解析 112
5.1.3 处理函数 114
5.1.4 模块集成 116
5.1.5 编译脚本和命令 117
5.1.6 测试验证 118
5.2 Nginx 开发基本流程 119
5.2.1 设计 119
5.2.2 开发 119
5.2.3 编译 120
5.2.4 测试验证 120
5.2.5 调优 121
5.3 Nginx 的模块 121
5.3.1 模块的数据结构 121
5.3.2 模块的种类 123
5.3.3 http 模块 124
5.3.4 模块的类图 124
5.3.5 模块的组织形式 125
5.4 Nginx 的配置 127
5.4.1 结构定义 128
5.4.2 配置解析的基本流程 131
5.4.3 配置数据的存储模型 133
5.4.4 访问配置数据 138
5.4.5 确定配置数据的位置 138
5.4.6 配置解析函数 140
5.4.7 配置数据的合并 141
5.4.8 配置指令的类型 142
5.5 Nginx 模块源码分析 143
5.5.1 ngx_core_module 143
5.5.2 ngx_errlog_module 146
5.6 Nginx 的编译脚本 147
5.6.1 运行机制 147
5.6.2 使用的变量 148
5.7 C++封装模块信息 149
5.7.1 NgxModuleConfig 149
5.7.2 NgxModule 153
5.8 C++封装配置解析 155
5.8.1 NgxCommand 155
5.8.2 NgxTake 155
5.8.3 NGX_MODULE_NULL 157
5.9 C++开发 Nginx 模块 158
5.9.1 C++模块的基本组成 158
5.9.2 配置信息类 159
5.9.3 业务逻辑类 160
5.9.4 模块集成类 162
5.9.5 实现源文件 164
5.9.6 增加更多功能 164
5.10 总结 165

第 6 章 Nginx HTTP 框架综述 169

6.1 HTTP 框架简介 169
6.1.1 模块分类 169
6.1.2 处理流程 170
6.1.3 请求结构体 172
6.1.4 请求的处理阶段 173
6.1.5 请求的环境数据 175
6.2 HTTP 处理引擎 176
6.2.1 处理函数原型 176
6.2.2 处理函数的存储方式 176
6.2.3 内容处理函数 177
6.2.4 引擎的数据结构 178
6.2.5 引擎的初始化 179
6.2.6 引擎的运行机制 181
6.2.7 日志阶段的处理 182

6.3 HTTP 过滤引擎 183
6.3.1 过滤函数原型 183
6.3.2 过滤函数链表 184
6.3.3 过滤函数的顺序 185
6.3.4 过滤链表的运行机制 187
6.3.5 请求体过滤 188
6.4 Nginx 模块源码分析 188
6.4.1 ngx_http_static_module 188
6.4.2 ngx_http_not_modified_filter_module 189
6.5 C++封装 190
6.5.1 NgxModuleCtx 191
6.5.2 NgxHttpCoreModule 193
6.5.3 NgxFilter 195
6.6 总结 197

第 7 章 Nginx HTTP 请求处理 199

7.1 HTTP 状态码 199
7.2 请求结构体 200
7.3 请求行 201
7.3.1 请求方法 201
7.3.2 协议版本号 202
7.3.3 资源标识符 202
7.4 请求头 203
7.5 请求体 204
7.5.1 结构定义 205
7.5.2 操作函数 205
7.6 响应头 206
7.6.1 结构定义 206
7.6.2 操作函数 207
7.7 响应体 207
7.8 C++封装 208
7.8.1 NgxHeaders 208
7.8.2 NgxRequestBody 211
7.8.3 NgxRequest 212
7.8.4 NgxResponse 213
7.9 开发 HTTP 处理模块 216
7.9.1 模块设计 216
7.9.2 配置信息类 217
7.9.3 业务逻辑类 217
7.9.4 模块集成类 219
7.9.5 实现源文件 221
7.9.6 编译脚本 221
7.9.7 测试验证 221
7.10 开发 HTTP 过滤模块 222
7.10.1 模块设计 222
7.10.2 配置信息类 222
7.10.3 环境数据类 223
7.10.4 业务逻辑类 223
7.10.5 模块集成类 227
7.10.6 实现源文件 228
7.10.7 编译脚本 228
7.10.8 测试验证 228
7.11 总结 229

第 8 章 Nginx HTTP 请求转发 231

8.1 upstream 框架简介 231
8.1.1 工作原理 232
8.1.2 请求结构体 233
8.1.3 上游结构体 234
8.1.4 上游配置参数 236
8.2 upstream 运行机制 237
8.2.1 回调函数 237
8.2.2 初始化 upstream 239
8.2.3 设置 upstream 239
8.2.4 启动 upstream 241
8.2.5 处理 upstream 数据 241
8.3 load-balance 运行机制 242

8.3.1 结构定义 243
8.3.2 初始化模块入口 247
8.3.3 初始化 IP 地址列表 248
8.3.4 初始化算法 250
8.3.5 执行算法 251
8.4 Nginx 模块源码分析 251
8.4.1 ngx_http_memcached_module . 251
8.4.2 ngx_http_upstream_ip_hash_module 254
8.5 C++封装 257
8.5.1 NgxUpstream 257
8.5.2 NgxUpstreamHelper 259
8.5.3 NgxHttpUpstreamModule 261
8.5.4 NgxLoadBalance 263
8.6 开发 upstream 模块 264
8.6.1 模块设计 264
8.6.2 配置信息类 264
8.6.3 业务逻辑类 265
8.6.4 模块集成类 268
8.6.5 实现源文件 269
8.6.6 编译脚本 269
8.6.7 测试验证 269
8.7 开发 load-balance 模块 270
8.7.1 模块设计 271
8.7.2 配置信息类 271
8.7.3 业务逻辑类 271
8.7.4 模块集成类 273
8.7.5 实现源文件 275
8.7.6 编译脚本 275
8.7.7 测试验证 275
8.8 总结 276

第 9 章 Nginx HTTP 子请求 277

9.1 子请求简介 277
9.1.1 工作原理 278
9.1.2 请求结构体 279
9.1.3 回调函数 280
9.1.4 待处理请求链表 282
9.1.5 子请求存储结构 282
9.2 子请求运行机制 283
9.2.1 创建子请求 283
9.2.2 处理引擎 287
9.2.3 数据整理 288
9.3 C++封装 290
9.3.1 NgxSubRequestHandler 290
9.3.2 NgxSubRequest 292
9.4 数据回传模块 293
9.4.1 模块设计 293
9.4.2 环境数据类 293
9.4.3 业务逻辑类 295
9.4.4 模块集成类 297
9.4.5 编译脚本 298
9.5 在模块里使用子请求 299
9.5.1 模块设计 299
9.5.2 配置信息类 299
9.5.3 业务逻辑类 300
9.5.4 测试验证 303
9.6 总结 304

第 10 章 Nginx 变量 305

10.1 结构定义 305
10.1.1 变量值 305
10.1.2 变量访问对象 306
10.1.3 变量的存储 307
10.1.4 请求结构体 307
10.2 运行机制 308
10.2.1 注册变量 308
10.2.2 获取变量 309

10.3 C++封装 310
10.3.1 NgxVariableValue 310
10.3.2 NgxVariable 311
10.3.3 NgxVariables 312
10.3.4 NgxVarManager 313
10.3.5 NgxVariableValueProxy 314
10.4 在模块里使用变量 315
10.4.1 添加变量 316
10.4.2 读写变量 317
10.5 总结 317

第 11 章 Nginx 辅助设施 319

11.1 摘要算法 319
11.1.1 MD5 319
11.1.2 SHA-1 320
11.1.3 MurmurHash 321
11.1.4 C++封装 321
11.2 编码和解码 324
11.2.1 CRC 校验 324
11.2.2 Base64 编码解码 325
11.2.3 URI 编码解码 326
11.2.4 HTML 和 JSON 编码 327
11.3 正则表达式 327
11.4 共享内存 328
11.4.1 结构定义 328
11.4.2 操作函数 329
11.4.3 C++共享内存 329
11.5 定时器 331
11.5.1 结构定义 331
11.5.2 操作函数 332
11.5.3 C++定时器事件 332
11.5.4 C++定时器工厂 333
11.6 总结 335

第 12 章 Nginx 与设计模式 337

12.1 设计模式简介 337
12.2 框架级别的模式 337
12.3 业务级别的模式 339
12.4 代码级别的模式 340
12.5 总结 342

第 13 章 结束语 343

13.1 本书的遗憾 343
13.2 下一步 344
13.3 临别赠言 345

附录 A 推荐书目 347

附录 B GDB 调试简介 349

附录 C Nginx 的字符串格式化 351

附录 D Nginx 里的 void*变量 353

附录 E Nginx C++模块简介 355

附录 F Nginx Lua 模块简介 357

第 0 章

导读

0.1 关于本书

Nginx[①]是由俄罗斯工程师 Igor Sysoev 开发的一个高性能 Web 服务器，各方面的表现均远超传统的 Apache，已经应用于诸多顶级互联网公司，为全世界数以亿计的网民提供着出色的服务。根据某权威公司分析统计，现在它已是市场份额第二大的 Web 服务器，并且仍在快速增长。

除了最引人注目的高性能和高稳定性外，Nginx 的另一个突出特点是高扩展性，其灵活优秀的模块化架构允许在不修改核心的前提下增加任意功能。自 2004 年正式发布以来，Nginx 已经拥有了百余个官方及非官方的功能模块（如 fastcgi、memcache、mysql、lua 等），这使得 Nginx 超越了一般意义上的 Web 服务器，成为了一个近乎“全能”的应用服务器。

Nginx 以纯 C 语言实现，故开发扩展功能模块也大多使用 C 语言。但由于 C 语言固有的“中级语言”特性，难以实现良好的软件结构，编写、调试代码都比较麻烦——特别是对于初学者。本书将在解析 Nginx 源码的基础上详细讲解使用 C++和 Boost 程序库来开发 Nginx 模块，充分利用现代 C++的新特性和标准库/Boost 库，让 Nginx 的模块开发变得更加便捷和轻松愉快。

---

① 似乎很多人（包括 Nginx 官网）都会孜孜不倦地提醒 Nginx 的正确发音，所以作者在这里也不能“免俗”：Nginx 的正确发音应该是“engine eks”。不过作者（还有周围的许多同事）却更愿意像 UNIX/Linux 那样称它为“engine ks”——虽然这是一个“错误”的发音，但却简洁明快。

0.2 读者对象

本书适合以下五类读者：

- 基于 Nginx 进行二次开发的软件工程师；
- Nginx 运维工程师；
- Web 服务器开发者；
- 对 Nginx 架构、内部实现感兴趣的程序员；
- C/C++、计算机编程爱好者和在校学生。

Nginx 采用进程池、事件驱动等方式来支持海量并发连接的处理，搭建了一个高性能高稳定性的服务器框架，在这个框架之内可以利用 Nginx 内部的各种构件编写模块，自由实现所需的业务功能。因此，国内外都有很多个人和企业以 Nginx 为平台进行二次开发，编写模块甚至直接定制 Nginx，进而提高网站的整体服务能力。本书由浅入深地详细介绍了 Nginx 模块开发的全过程，并且分析了 Nginx 的内部运行机制，能够帮助软件工程师较快地熟悉 Nginx 原理，迅速地投入到实际开发工作中。

Nginx 项目十分活跃，版本更新较快，新的功能不断增加，这使得市面上的相关书籍和网络资料常常会变得“过时”——只言片语、语焉不详或者答非所问，Nginx 运维工程师很可能会遇到运行、配置出现问题却无法解决的尴尬局面。然而 Nginx 是开源的，一切问题都可以在源码中找到答案。但十余万行的 Nginx 源码又有如汪洋大海，如果没有经验丰富的引航员指路，工程师难免会迷失方向，上下求索而不得。本书正是担当了这样的引航员角色，细致地分析了 Nginx 的配置解析功能和 HTTP 处理流程，让运维工程师可以快速定位配置指令和它的实现逻辑，深层次地做好 Nginx 的运维工作。

对于那些自行开发 Web 服务器的程序员来说，Nginx 也是一个非常有意义的借鉴品。它采用 epoll/kqueue 等异步调用削减了系统成本，充分挖掘了计算机硬件的性能，让 CPU、磁盘、网卡等设备并发运行，比传统的多进程、多线程方式的服务器运行得更快，服务能力更强。本书深入剖析了 Nginx 源码，解析了进程模型、模块架构等机制，学习借鉴这些大师级的代码，无疑会让自己的开发功力更上一层楼，写出更好的软件产品。

Nginx 虽然功能如此强大，但它的实现却非常的清晰易读：文件组织良好，代码格式优美，内部隐藏的大量编程技巧、设计思想和架构更是值得仔细研究的无价之宝。对于每一位喜好编程、准备或者正在投身于软件/互联网行业的人来说，本书都是一个很好的起点，可以从零起步窥探到 Nginx 内部的真正精髓，学习到各种实用的技术和知识，增加自身的“含金量”。

本书还大量应用了 C++11 标准和 Boost 程序库，使用了 lambda 表达式、模板元编程等

许多高级特性，可以看作是现代 C++编程范式的一次成功实践，读者可以接触到当前 C++最新、最前沿的技术。

0.3 读者要求

本书要求读者基本了解 C/C++和网络编程知识。

Nginx 本身是用 C 实现的，而本书使用 C++作为编程语言，所以读者应该具备足够的 C/C++知识，例如宏、指针、类型转换、封装、继承、异常等 C/C++特性，如果已有实际编程经验则更好。

在基本的 C++之上，本书还使用了 C++11 标准和 Boost 程序库来提高开发效率，要求读者对模板、泛型编程和 C++标准库有一定的了解，能够运用 vector、list 等常见的泛型容器和迭代器。

Nginx 是一个 Web 服务器，所以熟悉网络编程对于学习本书是非常有益的。但由于 Nginx 封装了很多网络通信的底层细节，所以本书不要求读者对网络编程技术有很深刻的认识，但至少应该对 TCP 和 HTTP 通信协议有所了解。

此外，因为 Nginx 大多运行在 Linux 服务器上，读者还应该再学习一些 UNIX/Linux 和 Shell 编程的相关知识，才能理解 Nginx 的编译脚本和运行维护。

附录 A 列出了一些技术书籍，涵盖了大部分本书要求掌握的 C/C++语言和网络编程技术知识，建议读者阅读本书时参考。

0.4 运行环境

Nginx 可以跨平台编译和运行，支持 Linux、FreeBSD、OS X、Windows 等多种操作系统。但就目前市场来看，Linux 是应用最普及的服务器操作系统，故本书的开发环境选用 Linux。

Linux 有很多的发行版本，企业中使用较多的是偏重于稳定性的 CentOS 和偏重于易用性的 Ubuntu，出于个人喜好的原因本书选择了后者（请 CentOS 支持者见谅）。

下面是本书使用的具体环境：

- 操作系统：Ubuntu 14.04（Linux 3.13.0）；
- Shell ：系统自带的 dash（注意不是 bash）；
- 编译器 ：系统自带的 GCC 4.8.2（支持 C++11 标准）；

0.5 本书的结构

Nginx是一个非常复杂的系统，简单的循序渐进方式难以透彻地讲解清楚，也不利于学习和掌握模块开发知识，所以本书结合“知其所以然”再“知其然”的方式组织全书的章节：首先介绍Nginx的基本知识作为入门，然后由浅入深地解析Nginx的源码和架构，理解了内部运行机制后再介绍如何开发Nginx模块。

学习Nginx开发不研究其内部工作原理是不行的，但解析得太深会导致源码太多，文字过于晦涩难懂；介绍得太浅又不能达到“知其所以然”的效果。所以本书只以中等深度研究Nginx源码和框架，讲解关键的流程、原理和函数调用，但不涉及过于底层的实现细节。每个章节首先学习结构定义等静态模型，然后研究工作原理、运行机制等动态模型，最后结合Nginx官方源码，通过模块实例讲解开发要点，这种方式能够较好地覆盖Nginx开发的各个方面。

全书共分14章，各章的内容简介如下。

- 第0章：导读

也就是读者正在阅读的这一章，介绍本书的基本内容和一些注意事项。

- 第1章：Nginx入门

本章简要介绍Nginx的历史、特点，以及如何编译、安装和配置Nginx，可以当作是一本简明Nginx使用手册。

- 第2章：Nginx开发准备

本章是开发Nginx模块前的准备工作，介绍本书使用的C++11标准和Boost程序库，还有Nginx源码的目录结构、基本的代码风格和特点。针对Nginx源码的特点提出了C++的解决方案，实现了一个对Nginx数据结构的C++封装类。

- 第3章：Nginx基础设施

剖析Nginx这样复杂的系统必须从最底层的基础设施开始，本章首先介绍了Nginx框架里基本的整数类型和错误码，然后再研究内存池、字符串、时间日期和运行日志，同时使用C++11标准进行了面向对象的封装，打造出方便易用的基础工具类。

- 第4章：Nginx高级数据结构

本章研究ngx_array_t、ngx_list_t、ngx_queue_t、ngx_buf_t等各种高级Nginx数据结构，它们类似C++的标准容器，在Nginx框架里经常出现，必须很好地理解并掌握它们的特性和用法，C++封装能够让这些数据结构更容易使用。