

Go语言编程之旅：一起用Go做项目



[Go语言编程之旅：一起用Go做项目_下载链接1](#)

著者:陈剑煜

出版者:电子工业出版社

出版时间:2020-6

装帧:平装

isbn:9787121390746

《Go语言编程之旅：一起用Go做项目》共分为6章，分别是命令行应用、HTTP应用、RPC应用、WebSocket应用、进程内缓存和Go语言中的大杀器，其中前5章是Go语言开发中最常见的项目类型。

《Go语言编程之旅：一起用Go做项目》对项目开发、细节分析、运行时分析等核心内容进行了较为深入的剖析，提供了相对完整的项目实践经验。在项目迭代过程中，难免会遇到问题，因此本书针对Go语言的大杀器（分析工具）及常见问题进行了全面讲解，帮助读者对各类问题进行排查。

本书适合有一定Go语言基础的读者以及相关开发人员阅读。

作者介绍:

陈剑煜（网名：煎鱼）

微信公众号“脑子进煎鱼了”的作者，“Go夜读”SIG小组成员，对Go语言有一定的兴趣和经验。在社区连载过一系列Go语言相关的技术文章，其中“跟煎鱼学Go”系列广受欢迎。

徐新华（网名：polaris）

Go语言中文网站长，目前在北京一家创业公司担任CTO。2011年北京大学硕士毕业，先后在开心网、奇虎360工作。一直从事服务端相关工作，有着丰富的经验。在Go1.0正式发布时开始接触Go语言，并将其用于工作中，。8年来一直致力于推广Go语言在国内的发展，致力构建国内Go语言中文社区。

目录: 第1章 命令行应用：打造属于自己的工具集 1

- 1.1 工具之旅 1
 - 1.1.1 标准库flag 1
 - 1.1.2 初始化项目 1
 - 1.1.3 示例 2
 - 1.1.4 分析 4
 - 1.1.5 定义参数类型 7
 - 1.1.6 小结 8
- 1.2 单词格式转换 8
 - 1.2.1 安装Cobra 8
 - 1.2.2 初始化cmd和word子命令 8
 - 1.2.3 单词转换 9
 - 1.2.4 word子命令 11
 - 1.2.5 验证 12
 - 1.2.6 小结 13
- 1.3 便捷的时间工具 13
 - 1.3.1 获取时间 14
 - 1.3.2 推算时间 14
 - 1.3.3 初始化子命令 15
 - 1.3.4 验证 17
 - 1.3.5 时区问题 17
 - 1.3.6 参考时间的格式 20
 - 1.3.7 小结 20
- 1.4 SQL语句到结构体的转换 20
 - 1.4.1 需要转换的数据结构 21
 - 1.4.2 生成结构体 21
 - 1.4.3 表到结构体的转换 24
 - 1.4.4 初始化子命令 28
 - 1.4.5 验证 30
 - 1.4.6 小结 31
- 第2章 HTTP应用：写一个完整的博客后端 32
 - 2.1 博客之旅 32
 - 2.1.1 gin 32
 - 2.1.2 初始化项目 32
 - 2.1.3 安装gin 32

- 2.1.4 快速启动 33
- 2.1.5 验证 34
- 2.1.6 源码分析 34
- 2.1.7 小结 38
- 2.2 项目设计 39
 - 2.2.1 目录结构 39
 - 2.2.2 数据库 40
 - 2.2.3 创建model 42
 - 2.2.4 路由 43
 - 2.2.5 处理程序 44
 - 2.2.6 启动接入 45
 - 2.2.7 验证 46
 - 2.2.8 小结 46
- 2.3 公共组件 46
 - 2.3.1 错误码标准化 47
 - 2.3.2 配置管理 50
 - 2.3.3 数据库连接 55
 - 2.3.4 日志写入 56
 - 2.3.5 响应处理 62
 - 2.3.6 小结 65
- 2.4 接口文档 65
 - 2.4.1 Swagger简介 65
 - 2.4.2 OpenAPI和Swagger 66
 - 2.4.3 安装 Swagger 66
 - 2.4.4 写入注解 66
 - 2.4.5 生成 68
 - 2.4.6 路由 68
 - 2.4.7 查看接口文档 69
 - 2.4.8 源码分析 70
 - 2.4.9 存在的问题 72
 - 2.4.10 小结 73
- 2.5 接口校验 73
 - 2.5.1 validator介绍 73
 - 2.5.2 业务接口校验 74
 - 2.5.3 国际化处理 75
 - 2.5.4 二次封装 77
 - 2.5.5 验证 78
 - 2.5.6 小结 79
- 2.6 模块开发：标签管理 80
 - 2.6.1 新建model方法 80
 - 2.6.2 处理model回调 81
 - 2.6.3 新建dao方法 84
 - 2.6.4 新建service方法 85
 - 2.6.5 新增业务错误码 86
 - 2.6.6 新增路由方法 87
 - 2.6.7 验证接口 89
 - 2.6.8 发现问题 89
 - 2.6.9 解决问题 90
 - 2.6.10 小结 91
- 2.7 模块开发：文章管理 91
 - 2.7.1 新建model方法 91
 - 2.7.2 新建dao方法 95
 - 2.7.3 新建service方法 98
 - 2.7.4 新增业务错误码 100

- 2.7.5 新增路由方法 101
- 2.7.6 验证接口 103
- 2.7.7 发现问题 103
- 2.7.8 解决问题 104
- 2.8 上传图片和文件服务 105
 - 2.8.1 新增配置 105
 - 2.8.2 上传文件 105
 - 2.8.3 新建service方法 109
 - 2.8.4 新增业务错误码 109
 - 2.8.5 新增路由方法 110
 - 2.8.6 验证接口 111
 - 2.8.7 文件服务 111
 - 2.8.8 源码分析 111
 - 2.8.9 小结 112
- 2.9 API访问控制 112
 - 2.9.1 JWT简介 113
 - 2.9.2 JWT的使用场景 115
 - 2.9.3 安装JWT 115
 - 2.9.4 配置JWT 115
 - 2.9.5 处理JWT令牌 116
 - 2.9.6 获取JWT令牌 118
 - 2.9.7 处理应用中间件 121
 - 2.9.8 小结 123
- 2.10 常见应用中间件 123
 - 2.10.1 访问日志记录 124
 - 2.10.2 异常捕获处理 125
 - 2.10.3 服务信息存储 130
 - 2.10.4 接口限流控制 131
 - 2.10.5 统一超时控制 134
 - 2.10.6 注册中间件 136
- 2.11 链路追踪 137
 - 2.11.1 OpenTracing规范 137
 - 2.11.2 Jaeger的使用 138
 - 2.11.3 在应用中注入追踪 139
 - 2.11.4 验证跟踪情况 141
 - 2.11.5 实现日志追踪 142
 - 2.11.6 实现SQL追踪 145
 - 2.11.7 小结 146
- 2.12 应用配置问题 147
 - 2.12.1 配置读取 147
 - 2.12.2 配置热更新 151
 - 2.12.3 小结 153
- 2.13 编译程序应用 153
 - 2.13.1 编译简介 154
 - 2.13.2 交叉编译 158
 - 2.13.3 编译缓存 158
 - 2.13.4 编译文件大小 159
 - 2.13.5 编译信息写入 160
 - 2.13.6 小结 162
- 2.14 优雅重启和停止 163
 - 2.14.1 遇到的问题 163
 - 2.14.2 解决方案 164
 - 2.14.3 常用的快捷键 164
 - 2.14.4 实现优雅重启和停止 165

- 2.14.5 小结 166
- 2.15 思考 167
 - 2.15.1 总结 167
 - 2.15.2 作业 167
- 第3章 RPC应用：启动你的RPC服务 169
 - 3.1 gRPC和Protobuf 169
 - 3.1.1 gRPC简介 169
 - 3.1.2 Protobuf简介 170
 - 3.1.3 gRPC的优点和缺点 172
 - 3.1.4 小结 174
 - 3.2 Protobuf的使用 174
 - 3.2.1 安装Protobuf 174
 - 3.2.2 初始化Demo项目 175
 - 3.2.3 编译和生成proto文件 176
 - 3.2.4 更多的数据类型支持 178
 - 3.2.5 小结 180
 - 3.3 gRPC的使用 180
 - 3.3.1 安装gRPC 180
 - 3.3.2 gRPC的调用方式 180
 - 3.3.3 Unary和Streaming RPC 187
 - 3.3.4 客户端与服务端是如何交互的 188
 - 3.3.5 小结 193
 - 3.4 运行一个gRPC服务 194
 - 3.4.1 初始化项目 194
 - 3.4.2 编译和生成proto文件 194
 - 3.4.3 编写gRPC方法 196
 - 3.4.4 编写启动文件 198
 - 3.4.5 调试gRPC接口 198
 - 3.4.6 gRPC的错误处理 199
 - 3.4.7 源码分析 204
 - 3.5 gRPC服务间的内调 205
 - 3.5.1 进行gRPC调用 206
 - 3.5.2 grpc.Dial做了什么 206
 - 3.6 提供HTTP接口 209
 - 3.6.1 支持其他协议 209
 - 3.6.2 另起端口监听HTTP 209
 - 3.6.3 在同端口监听HTTP 211
 - 3.6.4 同端口同方法提供双流量支持 213
 - 3.6.5 其他方案 221
 - 3.7 接口文档 221
 - 3.7.1 安装和下载 221
 - 3.7.2 静态资源转换 221
 - 3.7.3 Swagger UI的处理和访问 222
 - 3.7.4 Swagger描述文件的生成和读取 223
 - 3.7.5 查看接口文档 224
 - 3.7.6 小结 224
 - 3.8 gRPC拦截器 225
 - 3.8.1 拦截器的类型 225
 - 3.8.2 客户端和服务端拦截器 225
 - 3.8.3 实现一个拦截器 226
 - 3.8.4 实现多个拦截器 227
 - 3.8.5 “用”多个拦截器 228
 - 3.8.6 常用服务端拦截器 230
 - 3.8.7 常用客户端拦截器 232

- 3.8.8 演示 235
- 3.9 metadata和RPC自定义认证 237
 - 3.9.1 metadata介绍 237
 - 3.9.2 metadata是如何传递的 239
 - 3.9.3 对RPC方法做自定义认证 240
 - 3.9.4 小结 242
- 3.10 链路追踪 242
 - 3.10.1 注入追踪信息 243
 - 3.10.2 初始化Jaeger 243
 - 3.10.3 metadata的读取和设置 244
 - 3.10.4 服务端 245
 - 3.10.5 客户端 246
 - 3.10.6 实现HTTP追踪 247
 - 3.10.7 验证 248
 - 3.10.8 小结 249
- 3.11 gRPC服务注册和发现 249
 - 3.11.1 服务注册和发现 250
 - 3.11.2 gRPC负载均衡策略 251
 - 3.11.3 实现服务注册和发现 254
 - 3.11.4 其他方案 257
- 3.12 实现自定义的protoc插件 257
 - 3.12.1 插件的内部逻辑 258
 - 3.12.2 generator.Plugin接口 259
 - 3.12.3 FileDescriptor属性 259
 - 3.12.4 实现一个简单的自定义插件 262
 - 3.12.5 实现定制化的gRPC自定义插件 265
 - 3.12.6 小结 272
- 3.13 对gRPC接口进行版本管理 272
 - 3.13.1 接口变更 273
 - 3.13.2 可兼容性修改 273
 - 3.13.3 破坏性修改 273
 - 3.13.4 设计gRPC接口 273
 - 3.13.5 版本号管理 274
- 3.14 常见问题讨论 274
 - 3.14.1 Q&A 274
 - 3.14.2 小结 276
- 第4章 WebSocket应用：聊天室 277
 - 4.1 基于 TCP 的聊天室 277
 - 4.1.1 代码实现 277
 - 4.1.2 简单客户端 281
 - 4.1.3 演示 281
 - 4.1.4 改进 282
 - 4.1.5 小结 283
 - 4.2 认识 WebSocket 283
 - 4.2.1 WebSocket简介 283
 - 4.2.2 WebSocket 的优点 284
 - 4.2.3 选择一个合适的库 285
 - 4.2.4 nhooyr.io/websocket的介绍和使用 287
 - 4.2.5 抓包分析协议 289
 - 4.2.6 小结 292
 - 4.3 聊天室需求分析和设计 293
 - 4.3.1 聊天室的主要需求 293
 - 4.3.2 技术选择 294
 - 4.3.3 总体设计思路和流程 294

- 4.4 项目结构组织和基础代码框架 295
 - 4.4.1 项目结构组织 295
 - 4.4.2 基础代码框架 297
- 4.5 核心流程 299
 - 4.5.1 前端关键代码 299
 - 4.5.2 后端流程关键代码 303
 - 4.5.3 小结 310
- 4.6 广播器 311
 - 4.6.1 单例模式 311
 - 4.6.2 广播器的实现 316
- 4.7 非核心功能 325
 - 4.7.1 @提醒功能 325
 - 4.7.2 敏感词处理 328
 - 4.7.3 离线消息处理 332
 - 4.7.4 小结 341
- 4.8 关键性能分析和优化 341
 - 4.8.1 测试工具 341
 - 4.8.2 性能测试 344
 - 4.8.3 小结 350
- 4.9 Nginx部署 350
- 4.10 总结 351
- 第5章 进程内缓存 352
 - 5.1 缓存简介 352
 - 5.2 缓存淘汰算法 353
 - 5.2.1 初始化项目 353
 - 5.2.2 缓存接口 353
 - 5.2.3 FIFO算法 354
 - 5.2.4 LFU算法 360
 - 5.2.5 LRU算法 366
 - 5.3 进程内缓存 368
 - 5.3.1 支持并发读写 368
 - 5.3.2 缓存库主体结构TourCache 369
 - 5.3.3 测试 371
 - 5.4 缓存的性能和优化思路 373
 - 5.4.1 基准测试 373
 - 5.4.2 优化方案 376
 - 5.4.3 小结 378
 - 5.5 高性能缓存库——BigCache 378
 - 5.5.1 简单使用 378
 - 5.5.2 优化技巧 380
 - 5.5.3 小结 386
 - 5.6 进程内缓存的优化版 386
 - 5.6.1 分片技术的应用 386
 - 5.6.2 测试 390
 - 5.6.3 GC耗时验证 391
 - 5.6.4 小结 393
- 第6章 Go语言中的大杀器 394
 - 6.1 Go大杀器之性能剖析PProf（上） 394
 - 6.1.1 PProf简介 394
 - 6.1.2 PProf的使用 395
 - 6.1.3 通过测试用例做剖析 405
 - 6.1.4 通过Lookup写入文件做剖析 407
 - 6.1.5 为什么要初始化net/http/pprof 409
 - 6.1.6 小结 409

6.2 Go大杀器之性能剖析PProf（下）	409
6.2.1 场景	410
6.2.2 措施	410
6.2.3 排查	410
6.2.4 发现根源、解决问题	413
6.2.5 小结	414
6.3 Go大杀器之跟踪剖析trace	414
6.3.1 trace简介	415
6.3.2 实战演练	420
6.3.3 小结	422
6.4 用GODEBUG看调度跟踪	422
6.4.1 GODEBUG基础知识	423
6.4.2 GODEBUG	424
6.4.3 小结	429
6.5 用GODEBUG看GC	429
6.5.1 GC基础知识	429
6.5.2 GODEBUG	430
6.5.3 案例	432
6.5.4 涉及术语	433
6.5.5 小结	433
6.6 Go进程诊断工具gops	433
6.6.1 gops的基本使用	433
6.6.2 常规命令	434
6.6.3 源码分析	437
6.6.4 需要注意的一点	439
6.6.5 小结	439
6.7 公开和发布度量指标	439
6.7.1 expvar标准库	439
6.7.2 Prometheus技术栈	444
6.7.3 小结	446
6.8 逃逸分析	447
6.8.1 思考	447
6.8.2 堆和栈	447
6.8.3 逃逸分析简介	447
6.8.4 需要逃逸分析的原因	448
6.8.5 逃逸分析判断	448
6.8.6 逃逸案例	448
6.6.7 小结	452
附录A Go modules的入门和使用	453
附录B goroutine与panic、recover的小问题	469
附录C Go在容器运行时要注意的细节	477
附录D 让Go“恐慌”的十种方法	479
• • • • •	(收起)

[Go语言编程之旅：一起用Go做项目_下载链接1](#)

标签

Go

评论

[Go语言编程之旅：一起用Go做项目 下载链接1](#)

书评

[Go语言编程之旅：一起用Go做项目 下载链接1](#)