kratos框架介绍及示例

by liwenzhou.com

资料

kratos官方资料

https://github.com/go-kratos/kratos

https://go-kratos.dev/

复习资料

Protocol Buffers V3语法

前置依赖

推荐安装并使用最新稳定版本的Go。

推荐国内同学配置国内的GOPROXY节点。

```
go env -w GO111MODULE=on
go env -w GOPROXY=https://goproxy.cn,direct
```

安装CLI脚手架工具

https://go-kratos.dev/docs/getting-started/usage

kratos 是与 Kratos 框架配套的脚手架工具，kratos 能够

- 通过模板快速创建项目
- 快速创建与生成 protoc 文件
- 使用开发过程中常用的命令
- 极大提高开发效率，减轻心智负担

```
go install github.com/go-kratos/kratos/cmd/kratos/v2@latest
```

快速开始

创建项目

1. 创建一个名为helloworld的项目

```
kratos new helloworld
```

上面的命令会从github拉取项目目录模板，国内网络可能会失败。如果失败使用以下命令指定从gitee拉取。

```
# 如在国内环境拉取失败，可 -r 指定源
kratos new helloworld -r https://gitee.com/go-kratos/kratos-layout.git
```

2. 进入项目目录

```
cd helloworld
```

3. 拉取项目依赖

```
go mod download
```

生成代码

生成所有proto源码、wire等等

```
go generate ./...
```

运行

使用kratos运行项目

```
kratos run
```

或者手动编译后执行

```
# 编译
go build -o ./bin/ ./...
# 执行
./bin/helloworld -conf ./configs/config.yaml
```

输出

```
# 输出
INFO msg=config loaded: config.yaml format: yaml # 默认载入 configs/config.yaml 配置文件
INFO msg=[gRPC] server listening on: [::]:9000 # gRPC服务监听 9000 端口
INFO msg=[HTTP] server listening on: [::]:8000 # HTTP服务监听 8000 端口
```

项目的目录结构

```
.
├── Dockerfile
├── LICENSE
├── Makefile
├── README.md
├── api // 下面维护了微服务使用的proto文件以及根据它们所生成的go文件
|   └── helloworld
|       └── v1
|           ├── error_reason.pb.go
|           ├── error_reason.proto
|           ├── greeter.pb.go
|           ├── greeter.proto
|           ├── greeter_grpc.pb.go
|           └── greeter_http.pb.go
├── bin // 编译好的二进制可执行文件存放目录
|   └── helloworld
├── cmd // 整个项目启动的入口文件
|   └── helloworld
|       ├── main.go
|       ├── wire.go
|       └── wire_gen.go
├── configs // 配置文件目录
|   └── config.yaml
├── go.mod
├── go.sum
├── internal  // 该服务所有不对外暴露的代码，通常的业务逻辑都在这下面，使用internal避免错误引用
|   ├── biz
|   |   ├── README.md
|   |   ├── biz.go
|   |   └── greeter.go
|   ├── conf // 内部使用的config的结构定义，使用proto格式生成
|   |   ├── conf.pb.go
|   |   └── conf.proto
|   ├── data // 业务数据访问，包含 cache、db 等封装，实现了 biz 的 repo 接口。
|   |   ├── README.md
|   |   ├── data.go
|   |   └── greeter.go
|   ├── server // http和grpc实例的创建和配置
|   |   ├── grpc.go
```

```
|   |   ├── http.go
|   |   └── server.go
|   └── service // 实现了 api 定义的服务层，格式化输出数据&协同各类 biz 交互，但是不应处理复杂逻辑
|       ├── README.md
|       ├── greeter.go
|       └── service.go
├── openapi.yaml
└── third_party // api 依赖的第三方proto
    ├── README.md
    ├── errors
    |   └── errors.proto
    ├── google
    |   ├── api
    |   |   ├── annotations.proto
    |   |   ├── client.proto
    |   |   ├── field_behavior.proto
    |   |   ├── http.proto
    |   |   └── httpbody.proto
    |   └── protobuf
    |       ├── any.proto
    |       ├── api.proto
    |       ├── compiler
    |       |   └── plugin.proto
    |       ├── descriptor.proto
    |       ├── duration.proto
    |       ├── empty.proto
    |       ├── field_mask.proto
    |       ├── source_context.proto
    |       ├── struct.proto
    |       ├── timestamp.proto
    |       ├── type.proto
    |       └── wrappers.proto
    ├── openapi
    |   └── v3
    |       ├── annotations.proto
    |       └── openapi.proto
    └── validate
        ├── README.md
        └── validate.proto
```

依赖注入工具wire

安装wire

依赖注入工具，必须安装。

```
go install github.com/google/wire/cmd/wire@latest
```