er/zap，支持日志记录
+ goner/gin，集成gin框架，提供HTTP请求参数的依赖注入
+ goner/viper，用于解析多种配置文件
+ ...

小试牛刀

下面使用**Gone**来编写一个简单的Web服务，更多例子在[快速开始](http://cxyroad.com/ "https://goner.fun/zh/quick-start/") 和[example](http://cxyroad.com/ "https://github.com/gone-io/gone/tree/main/example")。

```
package main

import (
"fmt"
"github.com/gone-io/gone"
"github.com/gone-io/gone/goner"
)

// 实现一个Goner，什么是Goner? => https://goner.fun/zh/guide/core-concept.html#goner-%E9%80%9D%E8%80%85
type controller struct {
gone.Flag //goner 标记，匿名嵌入后，一个结构体就实现了Goner
gone.RouteGroup `gone:"gone-gin-router"` //注入根路由
}

// 实现 Mount 方法，挂载路由；框架会自动执行该方法
func (ctr *controller) Mount() gone.GinMountError {

// 定义请求结构体
type Req struct {
Msg string `json:"msg"`
}

/ 标签标记为需要依赖注入
}

func (b *Boss) Do() {
```

```
for _, w := range b.workers {
w.Do()
}
}

// WorkerX 实现 Worker 接口
type WorkerX struct {
gone.Flag
}

func (w *WorkerX) Do() {
fmt.Println("WorkerX Do")
}

func main() {
gone.
//Goner启动前的准备工作
Prepare(func(cemetery gone.Cemetery) error {
cemetery.
//注册Boss
Bury(&Boss{}).

//注册多个Worker Goners
Bury(&WorkerX{}).
Bury(&WorkerX{}).
Bury(&WorkerX{})
return nil
}).
//Goner启动
Run(func(boss *Boss) { //Run 方法中的参数被自动注入
boss.Do()
})
}

```

了解更多，请阅读 [开发指南](http://cxyroad.com/
"https://goner.fun/zh/guide/")

关于Logo

Golang的吉祥物是一只可爱的地鼠，Gone的Logo是从它衍生出来，加上翅膀加上光圈，是一只天使地鼠，我们感觉这很复活Gone鬼故事的气质。

原文链接: https://juejin.cn/post/7382044195534569511