链滴

beego 学习

作者：yunshang
原文链接：https://ld246.com/article/1516286324445
来源网站：链滴

beego 一种MVC架构的web框架，网上评价:设计繁复。优点:简单，国人开人，文档，资料多。

orm

- 支持 Go 的所有类型存储
- 轻松上手，采用简单的 CRUD 风格
- 自动 Join 关联表
- 跨数据库兼容查询
- 允许直接使用 SQL 查询 / 映射
- 严格完整的测试保证 ORM 的稳定与健壮

```
func init() {
  // 注册数据库
  orm.RegisterDataBase("default", "mysql", "root:root@/my_db?charset=utf8", 30)

// 创建数据库，第三个参数是是否检测数据库重复，报错
  orm.RunSyncdb("default", false, true)
}
```

ORM 接口使用

```
  var o Ormer
  o = orm.NewOrm() // 创建一个 Ormer
  // NewOrm 的同时会执行 orm.BootStrap (整个 app 只执行一次)，用以验证模型之间的定义并缓
。
  var r RawSeter
  r = o.Raw("UPDATE user SET name = ? WHERE name = ?", "testing", "slene")
  // Raw 函数，返回一个 RawSeter 用以对设置的 sql 语句和参数进行操作

}
```

context 模块

上下文模块主要是针对 HTTP 请求中，request 和 response 的进一步封装，他包括用户的输入和输，用户的输入即为 request，context 模块中提供了 Input 对象进行解析，用户的输出即为 respons，context 模块中提供了 Output 对象进行输出。

golang strconv包

strconv实现与基本数据类型的字符串表示形式的转换

```
  package main

  import (
      "fmt"
      "strconv"
  )

  func main() {
```

```
    // ParseBool 将字符串转换为布尔值
    fmt.Println(strconv.ParseBool("1")) // true
    fmt.Println(strconv.ParseBool("TRue"))
    // false strconv.ParseBool: parsing "TRue": invalid syntax
    fmt.Println(strconv.ParseBool("0")) // false
    fmt.Println(strconv.ParseBool("FALse"))
    // false strconv.ParseBool: parsing "FAlse": invalid syntax

    // FormatBool 将布尔值转换为字符串 "true" 或 "false"
    fmt.Println(strconv.FormatBool(0 > 1)) // false

    // ParseInt 将字符串转换为 int 类型
    // Atoi 相当于 ParseInt(s, 10, 0)
    // 通常使用这个函数，而不使用 ParseIn
    fmt.Println(strconv.Atoi("2147483647"))
    // 2147483647
    fmt.Println(strconv.Atoi("2147483648"))
    // 2147483647 strconv.ParseInt: parsing "2147483648": value out of range

    // ParseFloat 将字符串转换为浮点数
    s := "0.12345678901234567890"
    f, err := strconv.ParseFloat(s, 32)
    fmt.Println(f, err)         // 0.12345679104328156
}
```

casbin

casbin是一个用Go语言打造的轻量级开源访问控制框架，casbin采用了元模型的设计思想，支持多经典的访问控制方案，如基于角色的访问控制RBAC、基于属性的访问控制ABAC等。

casbin的主要特性包括

- 支持自定义请求的格式，默认的请求格式为{subject, object, action}；
- 具有访问控制模型model和策略policy两个核心概念；
- 支持RBAC中的多层角色继承，不止主体可以有角色，资源也可以具有角色；
- 支持超级用户，如root或Administrator，超级用户可以不受授权策略的约束访问任意资源；
- 支持多种内置的操作符，如keyMatch，方便对路径式的资源进行管理，如/foo/*** ar可以映射到/oo；

casbin不做的事情

- 身份认证authentication（即验证用户的用户名、密码），casbin只负责访问控制。应该有其他专的组件负责身份认证，然后由casbin进行访问控制，二者是相互配合的关系；
- 管理用户列表或角色列表。casbin认为由项目自身来管理用户、角色列表更为合适，casbin假设所策略和请求中出现的用户、角色、资源都是合法有效的。