Golang Tutorial #2

Hands on

golang coding convention

- camelCase
- official blog
- package name 單數名詞
- package 內部 variable name 不要重複 prefix, e.g., `chubby.ChubbyFile`
- error variable naming: prefix with `err` or `Err`
- Named Result Parameters 個人喜好

golang package

- namespace
  - shared namespace inside package, global var/func/...
- dependency
  - import by package
- only package main is entry
- godoc split page by package
- test split py package

golang package alias

- https://github.com/golang/go/wiki/CodeReviewComments#imports

```
import   "lib/math"         // math.Sin
import M "lib/math"         // M.Sin
import . "lib/math"         // Sin    like import math.*
import _ "lib/math"         // only do the init() for math
```

golang package template/example

- example: awesome
  - server application
  - cmd tools
  - library
- project layout

go example on go routines, channels 用法

- https://blog.golang.org/go-concurrency-patterns-timing-out-and
- https://blog.golang.org/pipelines
- https://talks.golang.org/2012/concurrency.slide

Gin async fast timeout

```
ctx.SetTimeout(timeout)
go func() {
    c.Next()
    finish()
}()
<-ctx.Done()
switch ctx.Err() {
// fast return to release http resource, actual handler is
case context.DeadlineExceeded:
    GinError(c, ErrTimeout)
    return
default:
    // do nothing, common path
}
```

resource pool (queue/ ring buffer)

```
c := make(chan *Session, dbi.MaxConn)

// fetch resource, blocking call
session = <-p.c:

// put back to pool after used
p.c <- session
```

go mod

- `go mod init github.com/xxx/xxx`

```
module gitlab.com/eagleraYp/lotushouse
require (
        cloud.google.com/go v0.34.0 // indirect
        firebase.google.com/go v3.5.0+incompatible
        github.com/googleapis/gax-go v2.0.2+incompatible //
```

- build `go build ./...`
  - `go build, go test` will automatically add new dependencies(updating go.mod and downloading the new dependencies).
- create vendor `go mod vendor`

go profiling

- https://github.com/davecheney/gophercon2018-performance-tuning-workshop
- `go tool pprof`
- http pprof
-
```
go tool pprof -http=":8011"
http://localhost:10201/debug/pprof/profile?
seconds=30
```

HTTP pprof example

```
import (
        "log"
        "net/http"
        _ "net/http/pprof"
)

// ActivateProfile runs the profiling endpoint
func ActivateProfile() {
        log.Println("Start profiling")
        go http.ListenAndServe(":10201", http.DefaultServeM
}
```

Extend reading

- Error handling and Go
- Go Errors
- Visualize go routines