Go Academy

#3 Logging and vendoring

www.3fs.si

Overview

- Vendoring
- Logging
- Go kit
- Refactoring

Vendoring

- vendor folder
- github.com/kardianos/govendor
- github.com/golang/dep
  - official experiment, but not yet the official tool

Vendoring - govendor

- go get -u github.com/kardianos/govendor
- `$ govendor init` # creates vendor/vendor.json
- `$ govendor fetch` …
- `$ govendor sync` # restore dependencies from vendor/vendor.json

Logging

- Unstructured

```
// Unstructured
log.Printf(“HTTP server listening on %s”, addr)

// Output:
// HTTP server listening on 127.0.0.1:80
```

- Structured

```
// Structured
logger.Log(“transport”, “HTTP”, “addr”, addr, “msg”, “listening”)

// Output:
// transport=HTTP addr=127.0.0.1:80 msg=listening
```

Logging - structured

- Package github.com/go-kit/kit/log

```
// Logger is the fundamental interface for all log operations. Log creates a
// log event from keyvals, a variadic sequence of alternating keys and values.
// Implementations must be safe for concurrent use by multiple goroutines. In
// particular, any implementation of Logger that appends to keyvals or
// modifies or retains any of its elements must make a copy first.
type Logger interface {
    Log(keyvals ...interface{}) error
}
```

Logging - structured

- Contextual

```
var logger log.Logger
logger = log.NewLogfmtLogger(log.NewSyncWriter(os.Stderr))
logger = log.With(logger, "instance_id", 123)

logger.Log("msg", "starting")
NewWorker(log.With(logger, "component", "worker")).Run()
NewSlacker(log.With(logger, "component", "slacker")).Run()

// Output:
// instance_id=123 msg=starting
// instance_id=123 component=worker msg=running
// instance_id=123 component=slacker msg=running
```

Logging - structured

- Logfmt

```
var logger log.Logger
logger = log.NewLogfmtLogger(log.NewSyncWriter(os.Stderr))
logger = log.With(logger, "ts", log.DefaultTimestampUTC, "caller", log.DefaultCaller)

logger.Log("msg", "hello")

// Output:
// ts=2016-01-01T12:34:56Z caller=main.go:15 msg=hello
```

- JSON

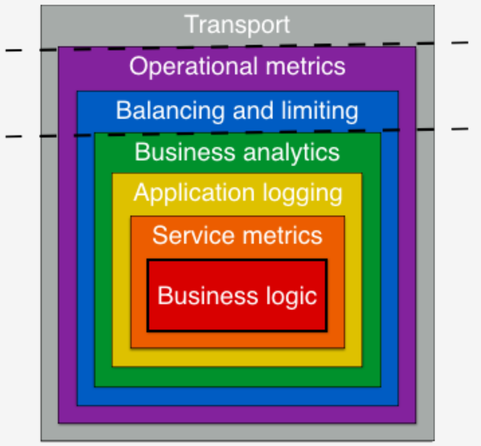
```
var logger log.Logger
logger = log.NewJSONLogger(log.NewSyncWriter(os.Stderr))
logger = log.With(logger, "ts", log.DefaultTimestampUTC, "caller", log.DefaultCaller)

logger.Log("msg", "hello")

// Output:
// {“ts”:“2016-01-01T12:34:56Z”,“caller”:”main.go:15”,”msg”:”hello”}
```

Go kit

- https://gokit.io
- is a collection of Go packages (libraries) that help you build robust, reliable, maintainable microservices
- It was originally conceived by Peter Bourgon

Go kit - architecture

- Transport
- Endpoint
- Service



https://gokit.io/faq/onion.png

Go kit - architecture

- Middlewares
  - enforce a strict separation of concerns

```
// service middleware
logger := log.NewLogger(...)

var service todo.Service    // interface
service = todo.NewService() // concrete struct
service = todo.NewLoggingMiddleware(logger)(service)
```

Refactoring

- Demo

www.3fs.si