Building RESTful Services with Go and MongoDB

GopherCon India 2015
20 February, 2015

Shiju Varghese

About Me

- Cloud Solutions Architect
- Gopher

@shijucv

https://github.com/shijuvar

Outline

- Building HTTP Servers in Go.
- Working with MongoDB using mgo.
- REST API Demo with Go and MongoDB.

Building HTTP Servers

net/http Package

- Provides HTTP client and server implementations.
- Allows you to build HTTP servers in Go.
- Provides composability and extensibility.

Processing HTTP Requests

- ServeMux - HTTP Request multiplexer (router).
- Handler - Serve HTTP Requests.

http.Handler Interface

```
type Handler interface {
    ServeHTTP(ResponseWriter, *Request)
}
```

- Serve HTTP Requests.
- Handlers are responsible for writing response headers and bodies.

HandlerFunc

```
type HandlerFunc func(ResponseWriter, *Request)
func (f HandlerFunc) ServeHTTP(w ResponseWriter, r *Request) {
  f(w, r)
}
```

An adapter to allow the use of ordinary functions as HTTP handlers.

A Simple HTTP Server

```
package main

import (
    "fmt"
    "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
    fmt.Fprint(w, "Hello, world!")
}

func main() {
    http.HandleFunc("/", handler)
    http.ListenAndServe(":8080", nil)
}
```

Persistence with MongoDB

mgo (mango)

- MongoDB driver for the Go.
- Written in Go.
- Created in 2010.
- Sponsored by MongoDB from 2011.

Connecting to MongoDB

```
session, err := mgo.Dial("localhost")
if err != nil {
     return err
}
```

Dial establishes a new session to the cluster of servers defined by the url parameter.

Working with Collections

Accessing collections

```
c := session.DB(database).C(collection)
```

Querying against collections

```
collection := session.DB("notesdb").C("notes")
var notes []Note
iter := collection.Find(nil).Iter()
    result := Note{}
    for iter.Next(&result) {
        notes = append(notes, result)
    }
```

Demo - REST API with Go and MongoDB

Model

```
type Note struct {
	Id          bson.ObjectId `bson:"_id" json:"id"`
	Title       string        `json:"title"`
	Description string        `json:"description"`
	CreatedOn   time.Time     `json:"createdon"`
}
```

Routing with gorilla/mux Package

```
r := mux.NewRouter()
r.HandleFunc("/api/notes", NotesHandler).Methods("GET")
r.HandleFunc("/api/notes", CreateNoteHandler).Methods("POST")
r.HandleFunc("/api/notes/{id}", UpdateNoteHandler).Methods("PUT")
r.HandleFunc("/api/notes/{id}", DeleteNoteHandler).Methods("DELETE")
```

Handler Functions

```
func CreateNoteHandler(w http.ResponseWriter, r *http.Request) {
}
func NotesHandler(w http.ResponseWriter, r *http.Request) {
}
func UpdateNoteHandler(w http.ResponseWriter, r *http.Request) {
}
func DeleteNoteHandler(w http.ResponseWriter, r *http.Request) {
}
```

Source Code

https://github.com/shijuvar/gophercon-rest-demo

Thank You

Shiju Varghese

Github - https://github.com/shijuvar
Twitter - https://twitter.com/shijucv