Go

ODDS | Thaibev

Day 3

Coverage

- Workshop (Privilege)
- Mongo-driver
- Gorm
- Dockerfile for Deployment
- Preview Project

Workshop (Coupon)

Initial Project

```
$ mkdir myapp && cd myapp
$ go mod init myapp
$ go get github.com/labstack/echo/v4
```

Mongo-driver

The Go driver lets you connect to and communicate with MongoDB clusters from a Go application.

Add MongoDB as a Dependency

Use go get to add the Go driver as a dependency.

```
$ go get go.mongodb.org/mongo-driver/mongo
```

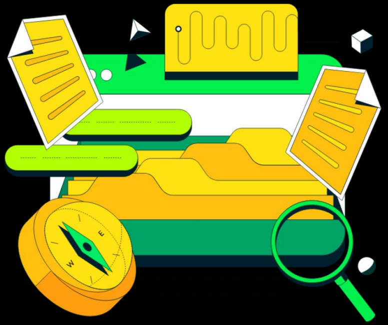
Create a MongoDB

```
$ docker run -d --name some-mongo \
    -e MONGO_INITDB_ROOT_USERNAME=mongoadmin \
    -e MONGO_INITDB_ROOT_PASSWORD=secret \
    -e MONGO_INITDB_DATABASE=privilege \
    -p 27017:27017 \
    mongo
```

MongoDB Compass Download

MongoDB Compass is a powerful GUI for querying, aggregating, and analyzing your MongoDB data in a visual environment.

Compass is free to use and source available, and can be run on macOS, Windows, and Linux.



https://www.mongodb.com/try/download/compass

Insert Master Data

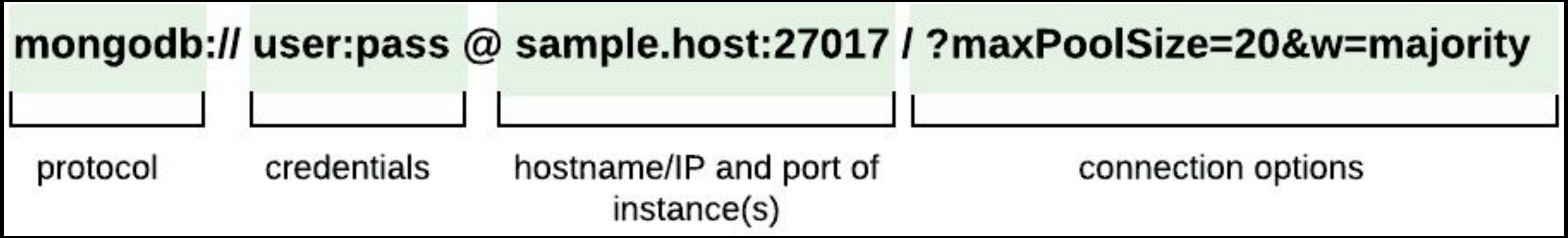
```
$ ./script/create_catalogs

$ ./script/create_category

$ ./script/create_type
```

Connection URI

The connection URI provides a set of instructions that the driver uses to connect to a MongoDB deployment. It instructs the driver on how it should connect to MongoDB and how it should behave while connected. The following example explains each part of a sample connection URI:


mongodb:// user:pass @ sample.host:27017 / ?maxPoolSize=20&w=majority
protocol
credentials
hostname/IP and port of instance(s)
connection options

https://www.mongodb.com/docs/drivers/go/current/fundamentals/connection/#std-label-golang-connection-guide

Usage

To get started with the driver, import the mongo package

```
import (
    "go.mongodb.org/mongo-driver/mongo"
    "go.mongodb.org/mongo-driver/mongo/options"
    "go.mongodb.org/mongo-driver/mongo/readpref"
)
```

Usage

create a mongo.Client with the Connect function:

```
ctx, cancel := context.WithTimeout(context.Background(),
               10*time.Second)
defer cancel()
client, err := mongo.Connect(ctx,
options.Client().ApplyURI("mongodb://root:password@localhost:27017/?
authSource=admin"))
```

Usage

Make sure to defer a call to Disconnect after instantiating your client:

```go
defer func() {
    if err = client.Disconnect(ctx); err != nil {
        panic(err)
    }
}()
```

Query Your MongoDB from Your Application

```
coll := client.Database("privilege").Collection("catalog")

title := "Vintage Point"

var result bson.M

err = coll.FindOne(context.TODO(), bson.D{{"title", title}}).Decode(&result)

if err == mongo.ErrNoDocuments {

    fmt.Printf("No document was found with the title %s\n", title)

    return

}

if err != nil {

   panic(err)

}
```

Gorm

The fantastic ORM library for Golang aims to be developer friendly.

Gorm

```
$ go get -u gorm.io/gorm

$ go get -u gorm.io/driver/postgres
```

Gorm

```
dsn := "host=localhost user=gorm password=gorm
dbname=gorm port=9920 sslmode=disable
TimeZone=Asia/Shanghai"

db, err := gorm.Open(postgres.Open(dsn),

&gorm.Config{})
```

Dockerfile for Deployment

```
FROM golang:1.19-alpine

ARG env
ENV env=$env

WORKDIR /app


COPY go.mod ./
COPY go.sum ./
RUN go mod download
COPY ./ .
COPY ./config-${env}.yaml config.yaml


RUN go mod tidy
RUN go build -o privilege
ENV TZ=Asia/Bangkok


EXPOSE 8000
ENTRYPOINT [ "./privilege" ]
```

Preview Project