

TALLER · VIERNES 2 · 09:30h

Crea una API REST con Go



Esteban Dorado

@Mr_Esti

\$> t3chfest

1 y 2 de marzo

Build REST API with Golang

Esteban Dorado



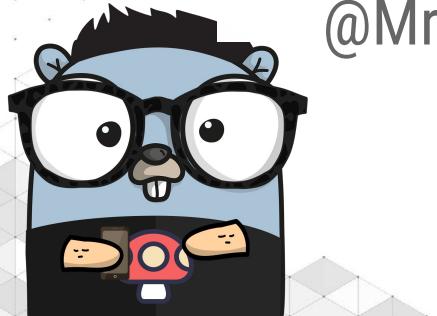
\$> t3chfest





B E E V A

@Mr_Esti



Esteban Dorado



Pet servers

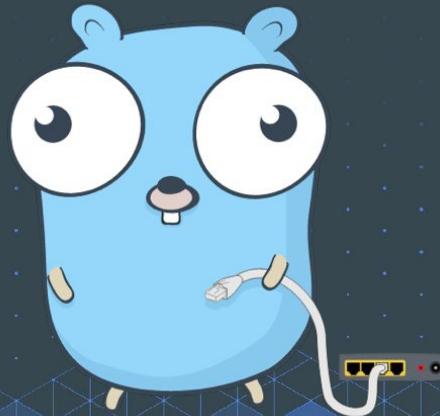


Cattle servers





REST API



What is REST?

- What does it stand for?

Representational **S**tate **T**ransfer

- What is it?

A style of software architecture for distributed systems

- Who/Where/When?

Came about in 2000 doctoral dissertation of Roy Fielding

REST Protocol

Describes six (6) constraints:

1. Uniform Interface
2. Cacheable
3. Client-Server
4. Stateless
5. Code on Demand
6. Layered System

What is RESTful API?

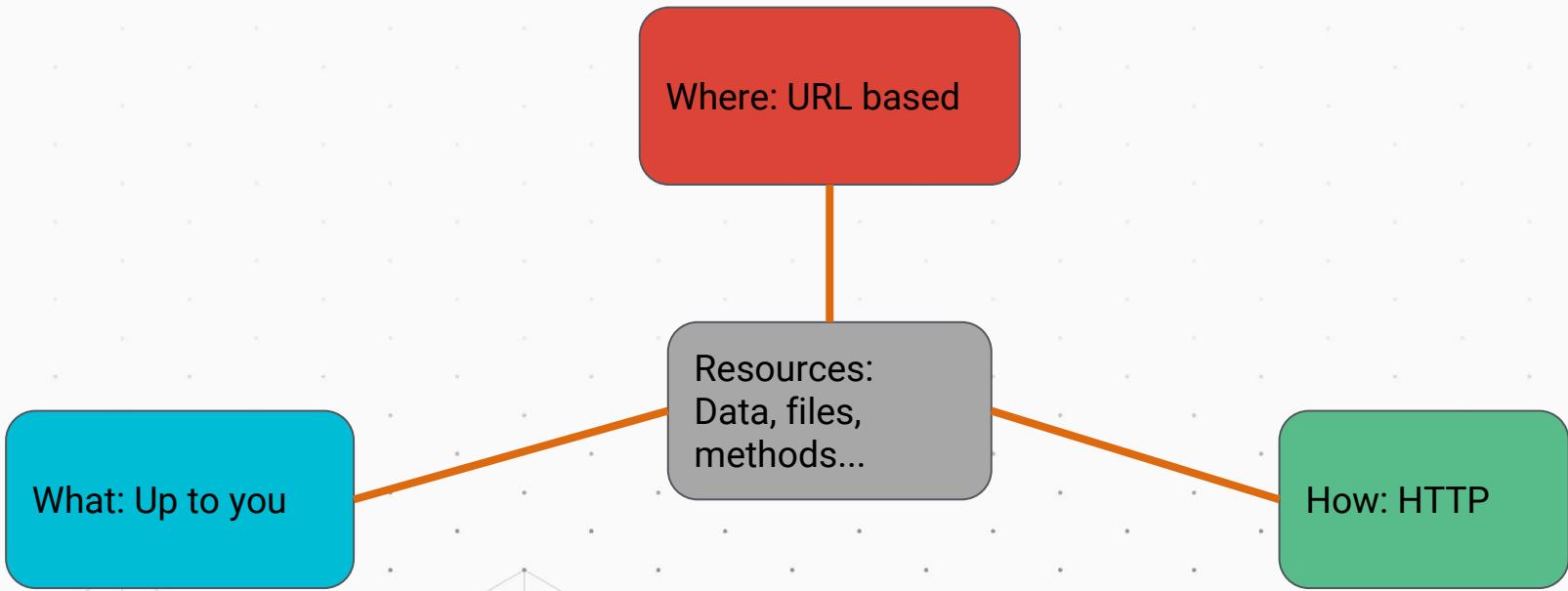
A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data.

A RESTful API -- also referred to as a RESTful web service -- is based on REST technology

PROs of RESTful API?

REST technology is generally preferred to the more robust **Simple Object Access Protocol (SOAP)** technology because REST leverages **less bandwidth**, making it more suitable for internet usage.

REST - Core



Example URIs

- GET

<http://www.example.com/v1/events/123818237>

- POST

<http://www.example.com/v1/events>

- DELETE

<http://www.example.com/v1/events/123818237>

- PUT

<http://www.example.com/v1/events/123818237>

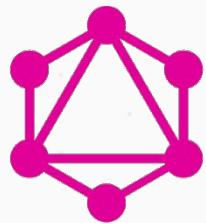
HTTP methods

HTTP Verb	CRUD	HTTP Codes
POST	Create	201 (Created), 404 (Not Found), 409 (Conflict)
GET	Read	200 (OK), 404 (Not Found)
PUT	Update/Replace	405 (Method Not Allowed), 200 (OK), 204 (No Content), 404 (Not Found)
PATCH	Update/Modify	405 (Method Not Allowed), 200 (OK), 204 (No Content), 404 (Not Found)
DELETE	Delete	405 (Method Not Allowed), 200 (OK), 404 (Not Found)

Resources in REST?

- Resources can be served in different representations:
 - XML, JSON, HTML, etc.
- Content negotiation methods
- Headers:
 - Accept or Content-Type
- Query parameters
 - GET /v1/users/10543?format=json
- Uri extension
 - GET /v1/users/10543.xml

Alternatives of REST API



GraphQL

<http://graphql.org>



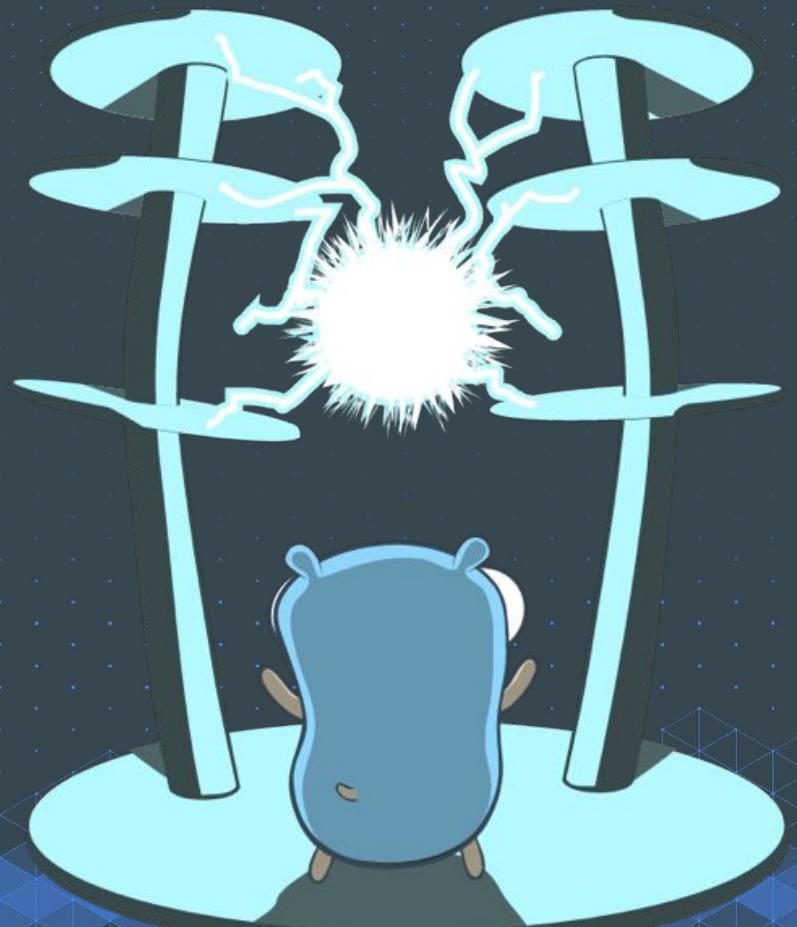
<https://grpc.io>





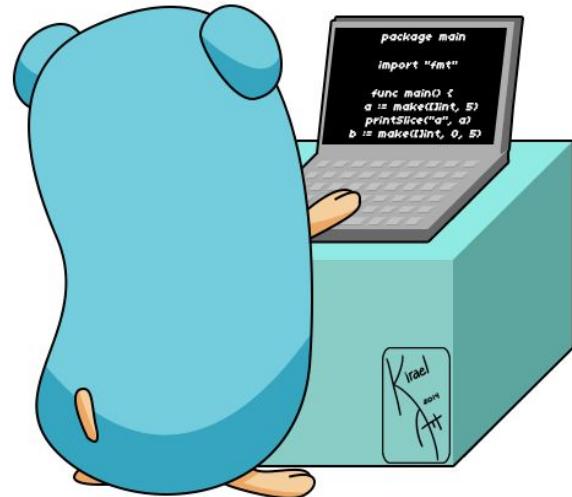


Let's build your REST API





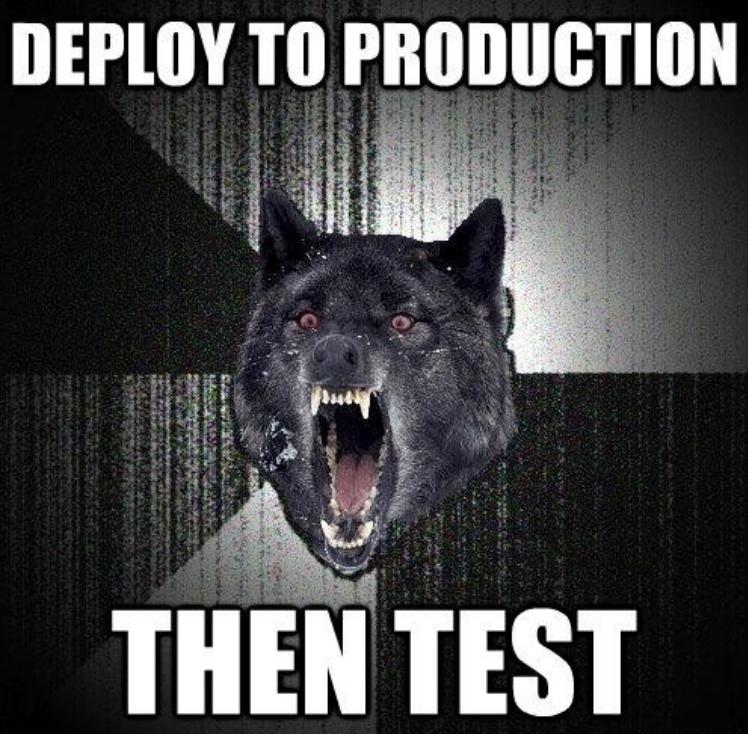
**EVERYBODY
PANIC!!!**



API

https://github.com/mresti/go_rest_api

DEPLOY TO PRODUCTION



THEN TEST

quickmeme.com

**HAVING ISSUES WITH
YOUR CODE?**

**LOOK IT UP ON
STACKOVERFLOW**

quickmeme.com

Mistakes were made. Code was shipped.



Undocumented Spaghetti Code

The Practical Guide

O RLY?

@ThePracticalDev

api.go

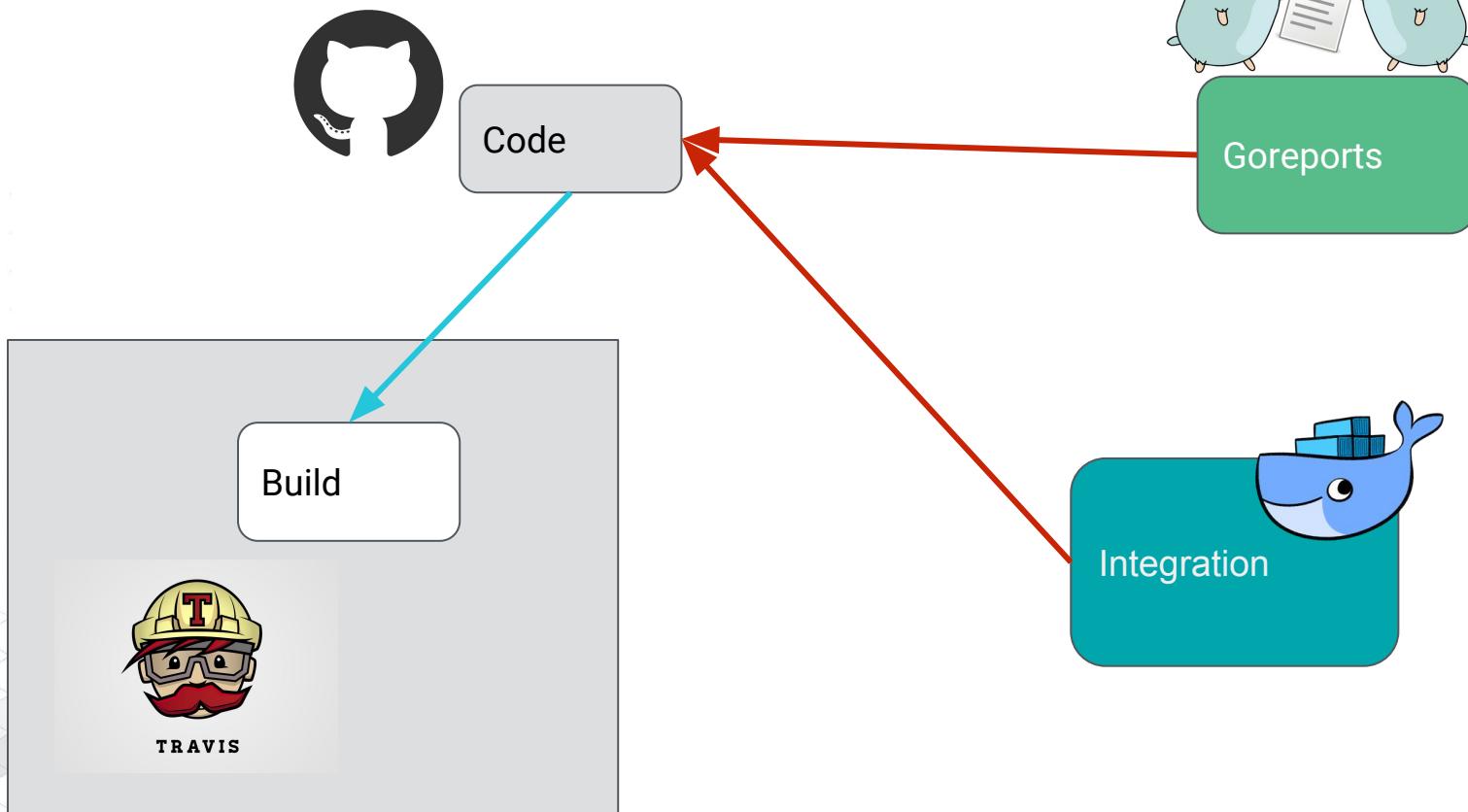
Add line

```
func Handlers() *http.ServeMux {  
    mux := http.NewServeMux()  
    mux.HandleFunc("/favicon.ico", func(_ http.ResponseWriter, _  
        *http.Request) {})  
    mux.HandleFunc("/", count)  
    mux.HandleFunc("/stats", stats)  
  
    return mux  
}
```



SHUT UP AND TAKE MY MONEY

Pipeline: API in golang



API: Makefile

```
# Project specific variables
```

```
PROJECT=api
```

```
OS=$(shell uname)
```

```
GOARCH = amd64
```

```
# GO env
```

```
GOPATH=$(shell pwd)
```

```
GO=go
```

```
GOCMD=GOPATH=$(GOPATH) $(GO)
```

```
GOBUILD = $(GOCMD) build
```

```
# Build the project
```

```
.PHONY: all
```

```
all: build
```

API: Makefile

```
.PHONY: build
```

```
build: format test compile
```

```
.PHONY: compile
```

```
compile: darwin linux windows
```

```
.PHONY: format
```

```
format:
```

```
    @for gofile in $$($(find ./src -name "*.go")); do \
        echo "formatting" $$gofile; \
        gofmt -w $$gofile; \
    done
```

```
.PHONY: test
```

```
test:
```

```
    $(GOCMD) test -v -race ./src/...
```

API: Makefile

```
.PHONY: run

run:
    $(GOCMD) run ./src/main.go

multi: build darwin linux windows
darwin:
    GOOS=darwin GOARCH=${GOARCH} $(GOBUILD) -o bin/$(PROJECT)_darwin
src/main.go
linux:
    GOOS=linux GOARCH=${GOARCH} $(GOBUILD) -o bin/$(PROJECT)_linux src/main.go
windows:
    GOOS=windows GOARCH=${GOARCH} $(GOBUILD) -o bin/$(PROJECT)_windows.exe
src/main.go
```

API: .travis.yml

```
language: go
go_import_path: github.com/mresti/go_rest_api
go:
  - "1.10"
  - 1.9.x
  - tip
matrix:
  allow_failures:
    - go: tip
  fast_finish: true
sudo: false
install:
  - go get -v github.com/alecthomas/gometalinter
  - gometalinter --install
```

API: .travis.yml

```
script:  
  - export PATH=$PATH:$HOME/gopath/bin  
  - export GORACE="halt_on_error=1"  
  - test -z "$(gometalinter --disable-all --enable=gofmt --enable=golint  
    --enable=vet --enable=gosimple --enable=unconvert  
    --deadline=4m ./spew | tee /dev/stderr)"  
  - make      # Run format code, run tests and build code  
  
deploy:  
  provider: script  
  script: curl -X POST https://goreportcard.com/checks -F  
    'repo=github.com/mresti/go_rest_api'  
  on:  
    branch: master
```

Go Report Card | Go project cc X Esteban

← → ↻ Es seguro | https://goreportcard.com/report/github.com/mresti/go_rest_api

Go Report Card [github.com/mresti/go_rest_api](#)

High Scores GitHub About

Report for [github.com/mresti/go_rest_api](#)

A+ Excellent! Found 3 issues across 6 files

Results
gofmt 100%
go_vet 100%
gocyclo 100%
golint 50%
license 100%
ineffassign 100%
misspell 100%

Last refresh: now

[Refresh now](#)

gofmt 100%

Gofmt formats Go programs. We run `gofmt -s` on your code, where `-s` is for the "simplify" command

No problems detected. Good job!

go_vet 100%

`go vet` examines Go source code and reports suspicious constructs, such as `Printf` calls whose arguments do not align with the format string.

No problems detected. Good job!

gocyclo 100%

`Gocyclo` calculates cyclomatic complexities of functions in Go source code. The cyclomatic complexity of a function is calculated according to the following rules: 1 is the base complexity of a function +1 for each 'if', 'for', 'case', '&&' or '||'. Go Report Card warns on functions with cyclomatic complexity > 15.

No problems detected. Good job!

https://goreportcard.com/report/github.com/mresti/go_rest_api

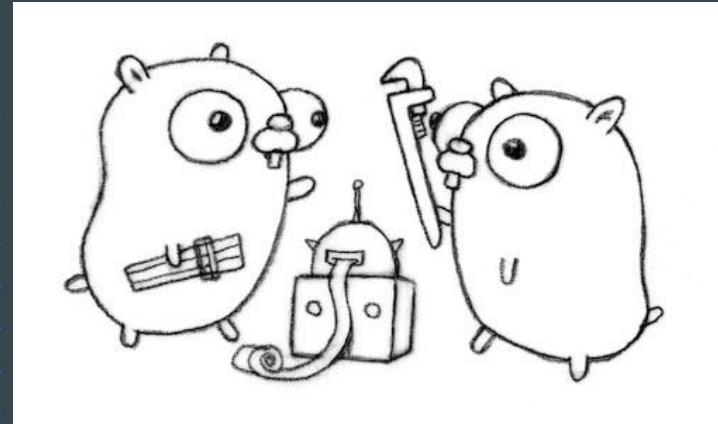
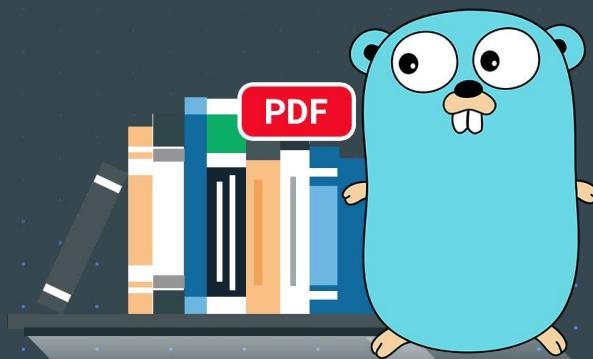
API: dockerfile

```
FROM golang:1.10-alpine
ENV PATH /api:$PATH
ARG API_PORT
ENV PORT $API_PORT
WORKDIR /api
# Copy binary titan
ADD ./bin/api_linux /api/api
# Modified files for titan
RUN chmod 555 /api/api
# Expose ports
EXPOSE $PORT
# Run Titan
CMD api -port $PORT
```

API: docker-compose.yml

```
version: "2.1"
services:
  api:
    build:
      context: .
    args:
      API_PORT: "8081"
  ports:
    - "8081:8081"
```

Research & Development



Beers with friends...





Q&A



<thank-you>

@Mr_Esti