

DESENVOLVIMENTO

WEB COM GO

CONTATO

eminetto@gmail.com

<https://eltonminetto.dev>

<http://twitter.com/eminetto>

EMENTA

- Introdução a Go
- Projeto de exemplo
 - Hello World
- Estrutura do projeto e entidade
 - Criando serviços
- Trabalhando com banco de dados e usando módulos
 - Iniciando o servidor e criando rotas
 - Implementando Rest e tratando erros
- Gerando HTML usando Templates e arquivos estáticos

INTRODUÇÃO A GO

O QUE É?

UMA LINGUAGEM OPEN SOURCE

Robert Griesemer
(dev Google)

Rob Pike (Unix)

Ken Thompson
(Unix, B, C)



**POR QUE UMA NOVA
LINGUAGEM?**

**MUITOS PROBLEMAS COM SOFTWARE
EM GRANDE ESCALA**

VELOCIDADE DE COMPILAÇÃO

SISTEMAS DISTRIBUÍDOS MULTICORE

OBJETIVOS

SEMÂNTICA SIMPLES

TIPAGEM ESTÁTICA

PROGRAMAÇÃO CONCORRENTE

DIVERTIDA!

**TALK IS CHEAP, SHOW
ME THE CODE!**

PACOTES

```
package main
```

```
import (  
    "fmt"  
    "math"  
)
```

```
func main() {  
    fmt.Printf("Now you have %g problems.", math.Sqrt(7))  
}
```

RESULTADOS MÚLTIPLOS

```
package main

import "fmt"

func swap(x, y string) (string, string) {
    return y, x
}

func main() {
    a, b := swap("hello", "world")
    fmt.Println(a, b)
}
```

ERROS

```
package main

import "github.com/coderockr/nfe/transmitter"

func main() {
    response, err := transmitter.transmit(nfe, xml)
    if err != nil {
        panic("Error ") //tratamento de erro qualquer
    }
    result, err := transmitter.saveData(response, xml)
    if err != nil {
        panic("Error ") //tratamento de erro qualquer
    }
}
```

GOROUTINES

```
package main

import (
    "fmt"
    "time"
)

func say(s string) {
    for i := 0; i < 5; i++ {
        time.Sleep(100 * time.Millisecond)
        fmt.Println(s)
    }
}

func main() {
    go say("world")
    say("hello")
}
```

CANAIS

```
package main

import "fmt"

func sum(s []int, c chan int) {
    sum := 0
    for _, v := range s {
        sum += v
    }
    c <- sum // send sum to c
}

func main() {
    s := []int{7, 2, 8, -9, 4, 0}

    c := make(chan int)
    go sum(s[:len(s)/2], c)
    go sum(s[len(s)/2:], c)
    x, y := <-c, <-c // receive from c

    fmt.Println(x, y, x+y)
}
```

CROSS COMPILATION

```
G00S=darwin GOARCH=amd64 go build gorotines.go
```

```
G00S=windows GOARCH=amd64 go build gorotines.go
```

```
G00S=linux GOARCH=amd64 go build gorotines.go
```

OO via composição e não herança, biblioteca
padrão poderosa, etc.

QUEM ESTÁ USANDO?

Google, Basecamp, Globo.com, Canonical,
DigitalOcean, Dropbox, Github, Heroku,
Medium, Docker, MongoDB, Mozilla, Netflix,
New Relic, New York Times, Resultados Digitais,
Moip, Neoway, Walmart, Code:Nation, etc

<https://github.com/golang/go/wiki/GoUsers>

APLICAÇÕES

APIS

MICROSERVICES

IOT

DATABASES

APLICATIVOS CLI

MATERIAL DE ESTUDO

<http://golang.org/>
<http://tour.golang.org/>
<https://gobyexample.com>
<http://exercism.io/>
<https://github.com/avelino/awesome-go>
<http://asemanago.com.br>
[https://novatec.com.br/livros/linguagem-de-
programacao-go/](https://novatec.com.br/livros/linguagem-de-programacao-go/)
<https://2020.gopherconbr.org>

PROJETO DE EXEMPLO

RESTBEER - API SOBRE 🍺🍺

Múltiplos formatos de resultado (JSON, HTML)

GET `http://localhost:4000/beer`

POST `http://localhost:4000/beer`

GET `http://localhost:4000/beer/1`

PUT `http://localhost:4000/beer/1`

DELETE `http://localhost:4000/beer/1`

HELLO WORLD

<https://github.com/eminetto/pos-web-go>

CLONAR O PROJETO

```
git clone https://github.com/eminetto/pos-web-go.git
```

COMPILAR

```
go build -o bin/pos-web-go main.go
```

COMPILAR E EXECUTAR

```
go run main.go
```

ESTRUTURA DO PROJETO E ENTIDADE

[https://github.com/eminetto/pos-web-go/pull/1/
files](https://github.com/eminetto/pos-web-go/pull/1/files)

git checkout 2-structure

CRIANDO SERVIÇOS

[https://github.com/eminetto/pos-web-go/pull/2/
files](https://github.com/eminetto/pos-web-go/pull/2/files)

```
git checkout 3-service
```


TRABALHANDO COM BANCO DE DADOS E USANDO MÓDULOS

<https://github.com/eminetto/pos-web-go/pull/4/files>

```
git checkout 4-sqlite  
go mod init github.com/eminetto/pos-web-go  
go get github.com/mattn/go-sqlite3
```

INICIANDO O SERVIDOR E CRIANDO ROTAS

[https://github.com/eminetto/pos-web-go/pull/5/
files](https://github.com/eminetto/pos-web-go/pull/5/files)

```
git checkout 5-main  
go run web/main.go
```

IMPLEMENTANDO REST E TRATANDO ERROS

[https://github.com/eminetto/pos-web-go/pull/6/
files](https://github.com/eminetto/pos-web-go/pull/6/files)

```
git checkout 6-rest-crud
```

GERANDO HTML USANDO TEMPLATES E ARQUIVOS ESTÁTICOS

[https://github.com/eminetto/pos-web-go/pull/7/
files](https://github.com/eminetto/pos-web-go/pull/7/files)

```
git checkout 7-templates
```

TRABALHO

Em grupos de no máximo 3 pessoas, implemente as funções `updateBeer` e `removeBeer` do arquivo `web/handlers/beer.go`

O trabalho deve ser entregue na forma de um link para um <https://gist.github.com>, pelo e-mail eminetto@gmail.com

CONTATO

eminetto@gmail.com

<http://eltonminetto.net>

<http://twitter.com/eminetto>