Desenvolvendo APIs usando middlewares

Elton Minetto

@eminetto
http://eltonminetto.net
http://asemanago.com.br
elton@planrockr.com

Http Is The Foundation Of The Web

Um cliente manda uma request

```
r.Method - HTTP method (GET, POST, PUT, PATCH, DELETE etc.)
r.URL.Path - Request path (/things/123)
r.URL.String() - Full URL
r.URL.Query() - Query parameters (q=something&p=2)
r.Body - io.ReadCloser of the request body
```

O servidor retorna uma response

```
type ResponseWriter interface {

  Header() Header
  Write([]byte) (int, error)
  WriteHeader(int)

}
```

Middlewares

Between the request and response

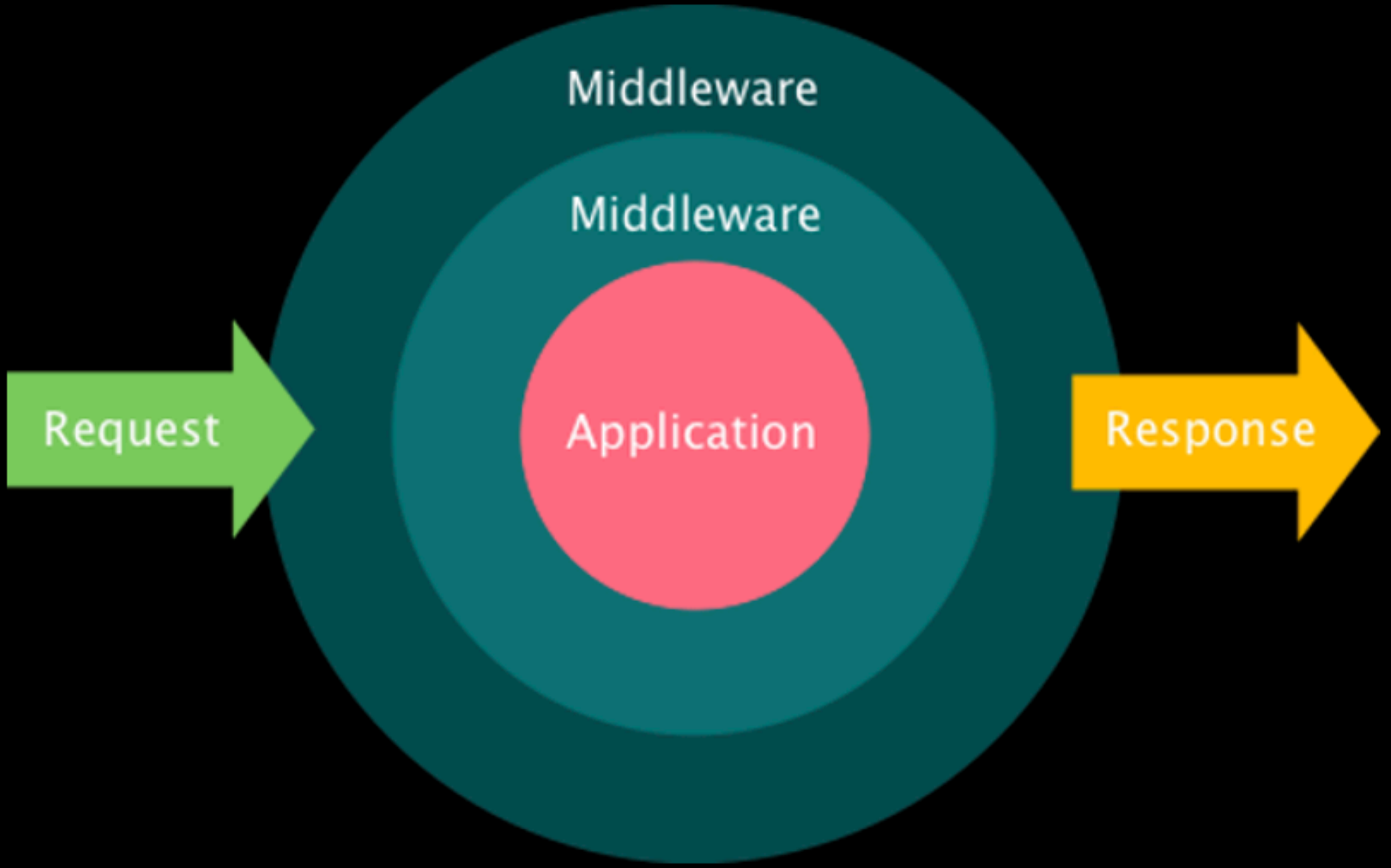
Enrico Zimuel

HTTP middleware is not for your Domain work. The middleware is a path in to, and out of, the core Domain.

Paul M. Jones

Run code before and after handler code

Mat Ryer


Middleware
Middleware
Application
Request
Response

Exemplos

```go
func middlewareOne(next http.Handler) http.Handler {
  return http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
    log.Println("Executing before middlewareOne")
    next.ServeHTTP(w, r)
    log.Println("Executing after middlewareOne")
  })
}
```

```go
func middlewareTwo(next http.Handler) http.Handler {
  return http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
    log.Println("Executing before middlewareTwo")
    if r.URL.Path != "/" {
      return
    }
    next.ServeHTTP(w, r)
    log.Println("Executing after middlewareTwo")
  })
}

func final(w http.ResponseWriter, r *http.Request) {
  log.Println("Executing finalHandler")
  w.Write([]byte("OK"))
}
```

```
func main() {
    finalHandler := http.HandlerFunc(final)
    http.Handle("/", middlewareOne(middlewareTwo(finalHandler)))
    http.ListenAndServe(":8000", nil)
}
```

https://github.com/justinas/alice

```
func main() {
    finalHandler := http.HandlerFunc(final)
    chain := alice.New(middlewareOne, middlewareTwo).Then(finalHandler)
    http.ListenAndServe(":8000", chain)
}
```

```
package main

import (
    "github.com/justinas/alice"
    "net/http"
    "planrockr"
)

func getCurrentSubscription(w http.ResponseWriter, r *http.Request) { ...
}

func main() {
    chain := alice.New(planrockr.Auth,
    planrockr.GetUserByToken).
    Then(http.HandlerFunc(getCurrentSubscription))
    http.ListenAndServe(":8000", chain)
}
```

```go
package main

import (
    "github.com/justinas/alice"
    "net/http"
    "planrockr"
)

func processEvent(next http.Handler) http.Handler { ...
}

func main() {
    chain := alice.New(processEvent,
    planrockr.ValidateHookData).
    Then(http.HandlerFunc(planrockr.Enqueue))
    http.ListenAndServe(":8000", chain)
}
```

Links

https://gist.github.com/eminetto/e3bab34426ac9a0b83a538a0e421bbc8

Contato

@eminetto
http://eltonminetto.net
http://asemanago.com.br
elton@planrockr.com