Урок Спринт 1.4

Слой middleware

Middleware в веб-сервере — это промежуточный набор программ, которые обрабатывают запросы до того, как они достигнут конечного обработчика, и/или после того, как они были обработаны. `Middleware` может проверять аутентификацию, записывать запросы в лог, обрабатывать ошибки и делать много других полезных вещей. `Middleware` в Go заменяют наследование, трейты и миксины в других языках программирования для выделения общей логики HTTP хендеров.

`Middleware` в Go реализуется как функции: они принимают `http.Handler` и возвращают новый `http.Handler`, который добавляет дополнительную логику к обработке запросов. Например, `middleware` для записи запросов в лог может выглядеть так:

```
func LoggingMiddleware(next http.Handler) http.Handler {
    return http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
        log.Printf("%s %s", r.Method, r.URL.Path)
        next.ServeHTTP(w, r)
    })
}
```

Эта функция принимает `http.Handler` и возвращает новый `http.Handler`, который записывает в журнал каждый запрос перед тем, как передать его следующему обработчику. Чтобы использовать этот `middleware`, мы можем добавить его к нашему обработчику:

```
mux := http.NewServeMux()
mux.HandleFunc("/", HomeHandler)

// Добавляем middleware для записи запросов
handler := LoggingMiddleware(mux)

http.ListenAndServe(":8000", handler)
```

Давайте соберем полноценное `http`-приложение:

```go
package main

import (
    "log"
    "net/http"
    "time"
)

func loggingMiddleware(next http.Handler) http.Handler {
    return http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
        start := time.Now()

        // Логируем информацию о запросе
        log.Printf("Запрос: %s %s", r.Method, r.URL.Path)

        // Передаем управление следующему обработчику
        next.ServeHTTP(w, r)

        // Вычисляем время выполнения запроса
        duration := time.Since(start)
        log.Printf("Время выполнения запроса: %s", duration)
    })
}

func helloHandler(w http.ResponseWriter, r *http.Request) {
    w.Write([]byte("Привет, мир!"))
}

func main() {
    mux := http.NewServeMux()

    // Создаем обработчик для маршрута "/"
    hello := http.HandlerFunc(helloHandler)

    // Применяем logging middleware к обработчику "/"
    mux.Handle("/", loggingMiddleware(hello))

    // Запускаем сервер на порту 8080
    if err := http.ListenAndServe(":8080", mux); err != nil {
        log.Fatal(err)
    }
}
```

Теперь каждый запрос будет записан в журнал перед тем, как он будет обработан нашим обработчиком.

Далее

Справка