

Programação Web Back-end


Middleware e validação de parâmetros

Adriano Rivolli

rivolli@utfpr.edu.br

Universidade Tecnológica Federal do Paraná (UTFPR)
Câmpus Cornélio Procopio
Departamento de Computação

Conteúdo

- 
- 1 Middlewares
 - 2 Validação de parâmetros

Middleware

Introdução aos Middlewares

- Uma forma de encapsular funcionalidades as requisições HTTP
- Uma função que recebe 3 (ou 4) argumentos
 - ▶ Objeto de erro (opcional)
 - ▶ Objeto de requisição
 - ▶ Objeto de resposta
 - ▶ Função `next()`

Pipeline

■ Os *middlewares* precisam ser registrados

- ▶ `app.use(meu_middleware)`
- ▶ `app.use(meu_middleware1, meu_middleware2)`
- ▶ `app.use(funcao_retorna_middleware())`
- ▶ `router.use((req, res, next) => {})`
- ▶ `app.use("/minha-rota", meu_middleware)`

■ A ordem de registro importa

■ É possível registrar *middlewares* para rotas e grupos de rotas

■ A função `next()` aciona a próxima função

- ▶ Para encerrar a requisição basta não chamar a função

Exemplo: criação e registro




■ Função de log:

```
app.use(function (req, res, next) {  
  console.log(Date.now(), - ", req.url)  
  next()  
})
```

■ Executado a todas as requisições

Aplicações

- 
- Validação de dados
 - Conversão de tipos
 - Tratamento de erros
 - Controle de acesso
 - Controle de logs
 - Sobrescrita de rotas


Middlewares disponíveis

- Body parser
- Arquivos estáticos
- Cookies e sessões

Body Parser

- Permite receber o uso dos dados enviados por POST
 - ▶ `app.use(express.urlencoded({extended: true}))`
- Faz o *parse* dos dados JSON convertendo-os em um objeto JavaScript acessível via **req.body**
 - ▶ `app.use(express.json())`

Servindo arquivos estáticos

- 
- Disponibiliza todos os arquivos de uma pasta
`app.use(express.static("public"))`
 - Cria um caminho virtual para acessar os arquivos
`app.use("/static", express.static("public"))`

Validação de parâmetros

Boas práticas

- **Sempre validar os dados recebidos pelo servidor**
- Uso de *middlewares* é uma boa alternativa
 - ▶ Valida os dados e mostra uma mensagem de erro no caso de inconformidades
 - ▶ Transforma e trata os tipos adequados
 - ▶ Garante unicidade dos dados e outras restrições

Validação manual

- Verifica os dados obrigatórios
- Verifica se o domínio/tipo dos valores estão corretos
 - ▶ Exemplo: valores mínimo/máximo
- Consulta o banco de dados para checagens adicionais

Uso de frameworks

■ Joi

▶ <https://joi.dev/api/>

■ Express validator

▶ <https://express-validator.github.io/docs/>