HTTP COM NODE JS

Douglas Nassif Roma Junior

/douglasjunior

/in/douglasjunior

douglasjunior.me

nassifrroma@gmail.com

Slides: https://git.io/vdn2c

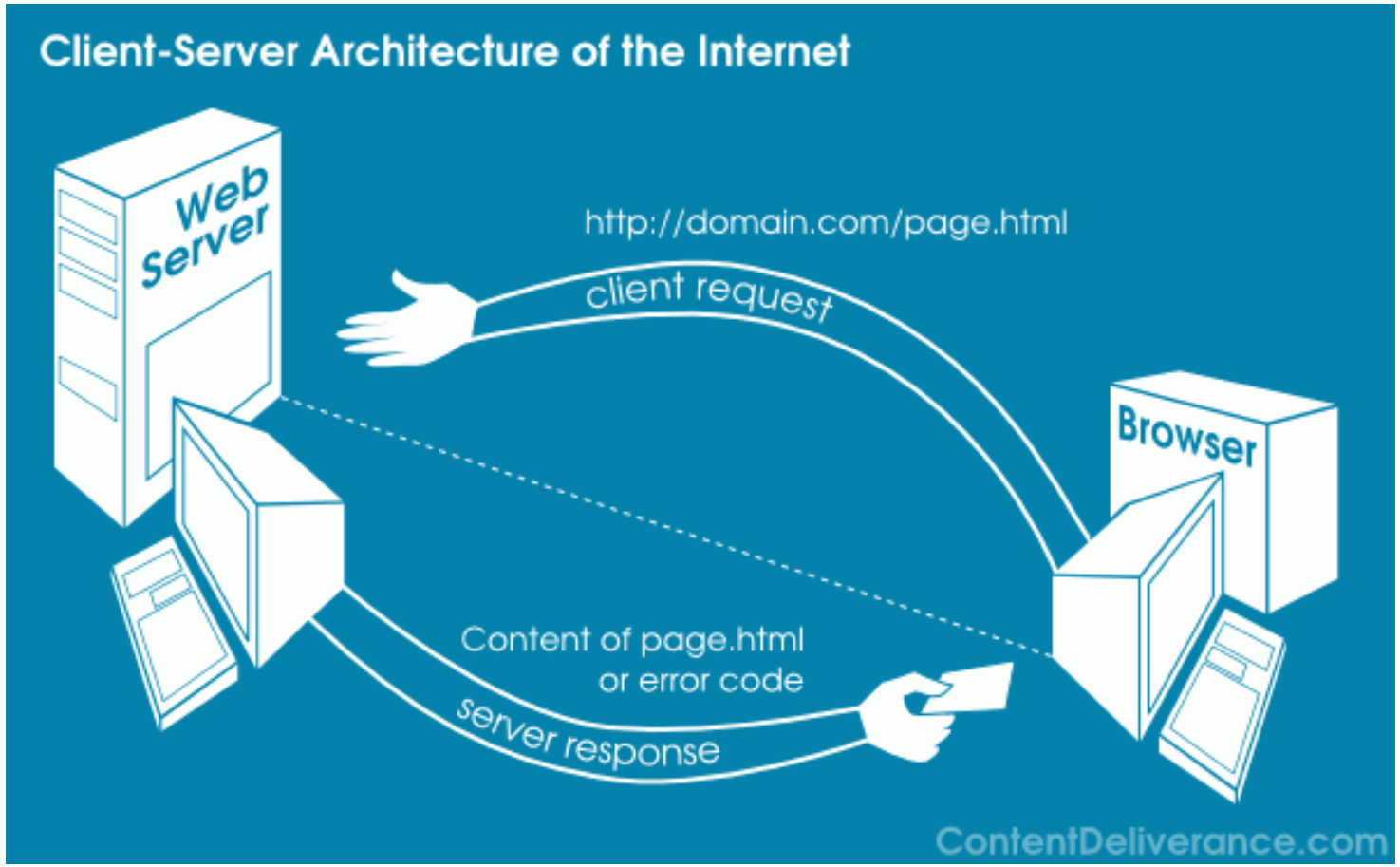
AGENDA

- Protocolo HTTP
- HTTP com Node JS
- Express JS
- Criação de rotas
- Middlewares
- Express Valdiator
- Express Generator
- Referências

PROTOCOLO HTTP

http://arquivo.devmedia.com.br/REVISTAS/java/imagens/117/3/1.png

PROTOCOLO HTTP



http://contentdeliverance.com/cms-school/wp-content/uploads/2011/05/client-server-diagram-internet.png

HTTP COM NODE JS

- Importe o módulo `http` e inicie o serviço na porta desejada.

```
const http = require('http');
const porta = 3000;

http.createServer((request, response) => {

  response.setHeader('content-type', 'text/html; charset=utf-8');
  response.writeHead(200);
  response.write('Olá Node JS!');
  response.end();

}).listen(porta, () => {
  console.log('Servidor iniciado na porta:', porta);
});
```

- Abra no navegador `http://localhost:3000`

CRIAÇÃO DE ROTAS

- As rotas são as URLs que a aplicação será capaz de responder.

```
const express = require('express');
const server = express();
const porta = 3000;

server.get('/usuarios/:usuarioId/', (request, response) => {
  const usuarioId = request.params.usuarioId;
  response.status(200).send("Listando usuários: " + usuarioId);
})

server.post('/usuarios/', (request, response) => {
  let usuario = "";
  request.on('data', (chunk) => { usuario += chunk });
  request.on('end', () => {
    console.log('Usuários recebido:', usuario);
    response.status(201).send();
  });
})

server.listen(porta, () => {
  console.log('Servidor iniciado na porta:', porta);
})
```

MIDDLEWARES

- Middlewares atuam como mediadores que interceptam as requisições para realizar algum tipo de tarefa.
  - Por exemplo, validação, autenticação, body-parser e etc.

```
const bodyParser = require('body-parser');
server.use(bodyParser.json());
server.use(bodyParser.urlencoded({ extended: false }));
```

```
server.use((request, response, next) => {
  if (usuarioAutenticado) {
    next();
  } else {
    response.status(403).send();
  }
});
```

EXPRESS VALIDATOR

- Express Validator é um middleware que facilita a validação do corpo da requisição e parâmetros da URL.

```
server.post('/usuarios/', (request, response, next) => {
  request.check({
    nome: {
      in: "body",
      notEmpty: true,
      errorMessage: "Informe o nome do usuário."
    }
  });
  request.getValidationResult()
    .then((result) => {
      if (result.isEmpty()) {
        next();
      } else {
        response.status(400).json(result.array());
      }
    }).catch(next);
}, (request, response) => {
  let usuario = request.body;
  console.log('Usuários recebido:', usuario);
  response.status(201).send();
});
```

EXPRESS GENERATOR

- Facilita a criação e organização inicial do projeto. Instale a dependência globalmente:

```
$ npm install express-generator -g
```

- Para criar um novo projeto:

```
$ express meu-projeto
```

- Instale as dependências do novo projeto:

```
$ cd meu-projeto
$ npm install
```

REFERÊNCIAS

- DevMedia: Como funcionam as aplicações Web - http://www.devmedia.com.br/como-funcionam-as-aplicacoes-web/25888
- Node JS HTTP - https://nodejs.org/api/http.html#http_class_http_server
- Express JS - http://expressjs.com/pt-br/
- Express Validator - https://github.com/ctavan/express-validator
- Express Generator - http://expressjs.com/pt-br/starter/generator.html
- Conteúdo do treinamento - https://git.io/vdn2c

DÚVIDAS?

Douglas Nassif Roma Junior

/douglasjunior

/in/douglasjunior

douglasjunior.me

nassifrroma@gmail.com

Slides: https://git.io/vdn2c