Valderi Leithardt, Dr.

IPW (Class six notes)

Summary

- Review of previous class *
- HTTP
- Pratica
- Exercises

Express.js *(*Review previous class)*

Express é o framework Node mais popular e a biblioteca subjacente para uma série de outros frameworks do Node. O Express oferece soluções para:

✓ Gerir requisições de diferentes verbos HTTP em diferentes URLs;

--> Verbos HTTP: GET, POST, DELETE, PUT, PATCH

Material adaptado conforme descrito em MDN, 2023

✓ Integrar "view engines" para inserir dados nos templates;

✓ Definir as configurações comuns da aplicação web, como a porta a ser usada para conexão e a localização dos modelos que são usados para renderizar a resposta;

✓ Adicionar novos processos de requisição por meio de "middleware" em qualquer ponto da "fila" de requisições.

Tutoriais de instalação

https://expressjs.com/en/starter/installing.html

https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm

https://github.com/programadriano/node-express

https://github.com/expressjs/express

NPM

O npm é o Gestor de Pacotes do Node (Node Package Manager) que vem junto com ele e que é muito útil no desenvolvimento Node.

- É um repositório amplamente usado para a publicação de projetos Node.js de código aberto (open-source). Isso significa que ele é uma plataforma online onde qualquer pessoa pode publicar e compartilhar ferramentas escritas em JavaScript.
- O npm é uma ferramenta de linha de comando que ajuda a interagir com plataformas online, como navegadores e servidores. Essa utilidade auxilia na instalação e desinstalação de pacotes, gerenciamento da versões e gerenciamento de dependências necessárias para executar um projeto.
- Para utilizar os pacotes, o projeto deve conter um arquivo chamado de package.json.
- Dentro do pacote encontrará metadados específicos para os projetos.
- Os metadados ajudam a identificar o projeto e agem como uma base para que os usuários obtenham as informações.

Os metadados mostram alguns aspetos do projeto na seguinte ordem:

1) O nome do projeto;
2) A versão inicial;
3) A descrição;
4) O ponto de entrada;
5) Os comandos de teste;
6) O repositório git;
7) As palavras-chave;
8) A licença;
9) As dependências;
10) As dependências do desenvolvedor (devDependencies).

NPM

Identificar um projeto por meio de metadados:

- ✓ O nome é x,z,y-npm;
- ✓ A versão é 1.0.0;
- ✓ A descrição informa npm guide for beginner;
- ✓ O ponto de entrada do projeto ou o arquivo principal é beginner-npm.js;
- ✓ As palavras chave ou tags para encontrar o projeto no repositório são npm, example e basic;
- ✓ O autor do projeto é X,Y, X;
- ✓ Este projeto está licenciado sob o MIT;
- ✓ As dependências ou outros módulos que esse módulo usa são express 4.16.4.

```
{
  "name": "xyz-npm",
  "version": "1.0.0",
  "description": "npm guide for beginner",
  "main": "beginner-npm.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "npm",
    "example",
    "basic"
  ],
  "author": "X,Y, Z",
  "license": "MIT",
  "dependencies": {
    "express": "^4.16.4"
  }
}
```

* Material adaptado conforme descrito NPM, 2023

HTTP

```
const express = require('express')

const produtos = [
  'Caneta',
  'lapis',
  'Eraser',
  'Cera',
  'Portatil'
]

const buscaProdutos = (produtoBuscado) => produtos.filter(produto => produtoBuscado === produto)

const server = express() /*servidor express*/

server.get('/produtos', (request, response) => {
  response.send(
    'Lista de Produtos:\n' +
    produtos.join('\n'))
})

server.get('/busca', (request, response) => {
  const produtosEncontrados = buscaProdutos(request.query.nomeProduto)

  response.send(
    'Lista de Produtos:\n' +
    produtosEncontrados.join('\n'))
})

server.listen(4000, () => console.log('Acesse localhost:4000'))
/*
npm install express (no terminal VSC)

No Browser http://localhost:4000/produtos
*/
```

File System do Node.js

- O FS (File System) é um módulo integrado do Node.js que fornece uma API para interagir com o sistema de arquivos em que o Node.js está sendo executado. Também permite a leitura, gravação, exclusão e manipulação de arquivos e diretórios.

- É possível criar, abrir, ler, gravar e fechar arquivos, além de manipular diretórios, como criar, renomear e excluir, também inclui recursos para manipulação de fluxos de dados, como a criação de fluxos de leitura e gravação.

- O módulo FS é especialmente útil em aplicativos de servidor, como servidores da web e aplicativos de back-end, que precisam interagir com o sistema de arquivos do servidor para armazenar e recuperar dados.

- A tarefa mais comun do módulo FS é ler o conteúdo de um arquivo. Exemplo utilizando o método fs.readFile():

Neste exemplo, estamos lendo o conteúdo de um arquivo de texto chamado ***arquivo.txt*** usando o método ***fs.readFile()***.

```
const fs = require("fs");
fs.readfile("/caminho/ficheiro.txt", "utf8", (err, data)=> {
if (err) throw err;
console.log(data);
});
```

File System do Node.js

- Além de ler arquivos, podemos usar o módulo FS para criar e escrever em arquivos. Aqui está um exemplo simples de como criar um arquivo e escrever conteúdo nele usando o método fs.writeFile():

```
7
8   const content = "Este é o conteúdo que será escrito no arquivo.";
9
10  fs.writeFile("/caminho/do/novo-arquivo.txt", content, (err) => {
11    if (err) throw err;
12    console.log("O arquivo foi salvo com sucesso!");
13  });
```

- Neste exemplo, esta sendo criado um arquivo chamado novo-arquivo.txt e escrevendo o conteúdo da variável content no arquivo usando o método fs.writeFile().

Fs Promises API – Acesso em (2023)
https://nodejs.org/api/fs.html

Módulo fs do Node.js – Acesso em (2023)
https://nodejs.reativa.dev/0046-node-module-fs/index

Fetch API

- O Fetch API é uma interface JavaScript moderna para fazer requisições HTTP/HTTPS de forma assíncrona. Permitindo os desenvolvedores criarem aplicações web mais interativas e dinâmicas, oferecendo uma maneira mais intuitiva e fácil de realizar chamadas de rede.

- Fetch são as abstrações da Interface do HTTP Request, Response, Headers (en-US), e Body payloads, juntamente com global fetch (en-US) método para iniciar requisições de recursos assíncronos. Como os componentes principais do HTTP são abstraidos como objetos de JavaScript, torna-se fácil APIs fazer uso das funcionalidades.

Material adaptado de MDN (2023).

Import

- A declaração estática import é usada para importar vínculos que são exportados por um outro módulo. Os módulos importados estão em strict mode, declarado como tal ou não. A declaração import não pode ser usada em scripts embutidos, a menos que tal script tenha um type="module". Há também uma função dinâmica import(), que não requer scripts de type="module".

- A compatibilidade com versões anteriores pode ser garantida usando o atributo nomodule na tag de script.

Import (MDN) – Acesso em (2023).

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Statements/import

Import – Acesso em (2023).

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/import

Exercises

- 1º Trabalho part 2 disponibilizado em:

https://github.com/isel-leic-ipw/2324i-IPW-LEIC31D/wiki/IPW_IP-2324-1-A1

Cap 18 – livro ENG: https://eloquentjavascript.net/18_http.htmll

* Leitura https://www.w3.org/TR/webarch/

** Concluir exercícios da aulas anteriores.

References

- https://eloquentjavascript.net/18_http.html
- https://www.tutorialspoint.com/expressjs/index.htm
- https://expressjs.com/en/starter/installing.html
- https://www.w3schools.com/nodejs/nodejs_http.asp
- Também foram realizadas adaptações e modificações com base no material disponibilizado por Professor Luís Falcão, acesso online em: https://github.com/isel-leic-ipw/
- Aulas gravadas Professor Falcão:

- Part 1: https://videoconf-colibri.zoom.us/rec/share/iRDoy6zvDh_y1SPgYjDi2i5KX1JIgpvI5cUEY3mOJzfL7mR6KMOSTsiCwWwwAhvI.QU1pZ87KsebyesZu
- Part 2: https://videoconf-colibri.zoom.us/rec/share/HsomnIdwL49NJs2EfIhiPoSXEoP3gYdrOqFY-sFKHtoM3qJ0rddLzJmWwbmc9mla.H5OI96z0v_HPDTeZ

Valderi Leithardt, Dr.

Professor IPW

valderi.leithardt@isel.pt