03

A biblioteca jsonfile

Transcrição

Como estamos trabalhando com o Electron, que por sua vez roda em cima do Node.js, se quisermos fazer a persistência do curso e do seu tempo estudado, existem vários bancos de dados que possuem compatibilidade com o Node.js, como MySQL, MongoDB, entre vários outros. Para isso, bastaria utilizar o driver do banco de dados para implementar a persistência dos dados.

Mas como a integração com o banco de dados é fora do escopo do nosso curso, e salvar esses dados no banco é "matar mosca com canhão", faremos uma coisa mais simples para salvar esses dados: vamos salvá-los em pequenos arquivos JSON, que ficarão na própria aplicação.

Para isso, dentro da pasta **alura-timer**, vamos criar a pasta **data**, que será onde salvaremos os arquivos JSON dos nossos cursos, ou seja, para cada curso teremos um arquivo JSON, e dentro dele os seus dados, como o tempo estudado. Salvando os dados em arquivos JSON, também conseguimos facilmente consumir esses dados em outra aplicação.

Criando um arquivo JSON

Para implementar essa persistência, primeiramente devemos ter alguém responsável por criar e preencher os arquivos JSON. Para isso, criaremos um novo módulo, o **data.js**, dentro da pasta **alura-timer**. Esse módulo possuirá as funções responsáveis por criar e escrever os arquivos JSON dos cursos.

Se o **data.js** será um módulo, já podemos escrever o `module.exports` . Dentro dele, já implementaremos a primeira função, `criaArquivoDeCurso()`

```
// data.js

module.exports = {
    criaArquivoDeCurso() {

    }
}
```

Se ainda não houver o arquivo JSON do curso criado, essa função será a responsável por criá-lo. Como criar e escrever em arquivo JSON é um recurso muito feito em Node, já há uma biblioteca que nos auxilia para tal, o nome dela é **jsonfile**. Para utilizá-la, assim como com as bibliotecas que utilizamos anteriormente, precisamos instalá-la, dentro da pasta **alura-timer**, pelo terminal, executamos:

```
npm install jsonfile-promised --save
```

Vamos utilizar o **jsonfile-promised**, justamente porque essa versão trabalha com ***promisses***, nos facilitando com as operações assíncronas. Com a biblioteca instalada, podemos importá-la:

```
// data.js

const jsonfile = require('jsonfile-promised');
```

```
module.exports = {
    criaArquivoDeCurso() {

    }
}
```

Para criar o arquivo JSON, podemos utilizar a função `writeFile`, que recebe dois parâmetros, o caminho do arquivo e o seu conteúdo. Para utilizar esses dois parâmetros, vamos recebê-los na função `criaArquivoDeCurso`:

```
// data.js

const jsonfile = require('jsonfile-promised');

module.exports = {
    criaArquivoDeCurso(nomeArquivo, conteudoArquivo) {
        jsonfile.writeFile(nomeArquivo, conteudoArquivo);
    }
}
```

E como estamos trabalhando com o **jsonfile-promised**, o código acima é uma *promisse*, logo já podemos encadear o `.then`, que imprimirá uma mensagem de sucesso caso tudo ocorra perfeitamente. Caso ocorra um erro, vamos imprimi-lo, chamando o `.catch`:

```
// data.js

const jsonfile = require('jsonfile-promised');

module.exports = {
    criaArquivoDeCurso(nomeArquivo, conteudoArquivo) {
        jsonfile.writeFile(nomeArquivo, conteudoArquivo)
                .then(() => {
                    console.log('Arquivo Criado');
                }).catch((err) => {
                    console.log(err);
                });
    }
}
```

Resta agora chamar essa função, lá no **main.js**.

Utilizando o módulo

No **main.js**, para utilizar o módulo, devemos importá-lo e no momento em que o curso for parado (lembrando que estamos escutando o evento `curso-parado`), nós salvamos os dados:

```
// main.js

const data = require('./data');

// restante do código comentado
```

```
ipcMain.on('curso-parado', (event,curso, tempoEstudado) => {
    console.log(`O curso ${curso} foi estudado por ${tempoEstudado}`);
    data.salvaDados(curso, tempoEstudado);
});
```

Agora, no **data.js**, criamos a função `salvaDados`. Nessa função, nós verificamos se o arquivo JSON do curso já existe, se sim, nós salvamos os dados nele, se não, nós criamos-o. Algo como:

```
// data.js

const jsonfile = require('jsonfile-promised');

module.exports = {
    salvaDados(curso, tempoEstudado) {
        if() {
            // Salva os dados
        } else {
            // Cria o arquivo e salva os dados nele
        }
    },
    criaArquivoDeCurso(nomeArquivo, conteudoArquivo){
        jsonfile.writeFile(nomeArquivo,conteudoArquivo)
                .then(() => {
                    console.log('Arquivo Criado')
                }).catch((err) => {
                    console.log(err);
                });
    }
}
```

Mas como vemos se um arquivo existe?

Verificando a existência de um arquivo

Para verificar a existência de um arquivo, o próprio Node já possui o módulo ***File System*** (**fs**), que pode realizar essa verificação para nós. Então vamos importá-lo:

```
// data.js

const jsonfile = require('jsonfile-promised');
const fs = require('fs');

// restante do código comentado
```

Com esse módulo, chamando a sua função `existsSync` e passando como parâmetro o caminho completo do arquivo, conseguimos realizar essa verificação. Mas qual o caminho completo do arquivo? É o diretório atual (`__dirname`), dentro da pasta **data**, mais o nome do curso com a extensão `.json`. Vamos criar uma variável com esse caminho, e passá-la para a função `existsSync` :

```
// data.js

const jsonfile = require('jsonfile-promised');
const fs = require('fs');

module.exports = {
    salvaDados(curso, tempoEstudado) {
        let arquivoDoCurso = __dirname + '/data/'+ curso + '.json';
        if(fs.existsSync(arquivoDoCurso)) {
            // Salva os dados
        } else {
            // Cria o arquivo e salva os dados nele
        }
    },
    criaArquivoDeCurso(nomeArquivo, conteudoArquivo){
        jsonfile.writeFile(nomeArquivo,conteudoArquivo)
                .then(() => {
                    console.log('Arquivo Criado')
                }).catch((err) => {
                    console.log(err);
                });
    }
}
```

Agora que estamos fazendo essa verificação de existência do arquivo do curso, se ele existir, basta salvá-lo, se não, nós chamamos a função **`criaArquivoDeCurso`** do módulo, passando o arquivo do curso e como conteúdo um objeto vazio, por enquanto. Além disso, vamos fazer com que a função **`criaArquivoDeCurso`** retorne a *promisse*, para que possamos encadear um **`.then`** e salvar os dados do curso:

```
// data.js

const jsonfile = require('jsonfile-promised');
const fs = require('fs');

module.exports = {
    salvaDados(curso, tempoEstudado) {
        let arquivoDoCurso = __dirname + '/data/'+ curso + '.json';
        if(fs.existsSync(arquivoDoCurso)) {
            // Salva os dados
        } else {
            this.criaArquivoDeCurso(arquivoDoCurso,{})
                .then(() => {
                    // Salva os dados
                });
        }
    },
    criaArquivoDeCurso(nomeArquivo, conteudoArquivo){
        return jsonfile.writeFile(nomeArquivo,conteudoArquivo)
                .then(() => {
                    console.log('Arquivo Criado')
                }).catch((err) => {
                    console.log(err);
                });
    }
}
```

Com esse código, já conseguimos criar o arquivo do curso, restando apenas salvar os dados dentro dele.