10

Mãos na Massa: Salvando os dados

Vamos agora começar a persistência de nossos dados de cursos em arquivos JSON.

1- Vamos começar criando um arquivo chamado `data.js`, que será nosso módulo responsável por persistir os dados. Salve-o na pasta raiz do `alura-timer`:

```
//data.js

module.exports = {

}
```

2- Vamos importar agora o módulo `jsonfile-promised`, que será o módulo que usaremos para persistir os dados. Em seu terminal na pasta do `alura-timer` execute o comando:

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

Em seguida importe-o no topo do `data.js`:

```
//data.js
const jsonfile = require('jsonfile-promised');

module.exports = {

}
```

3- Vamos criar a primeira função do módulo, que será responsável por criar um novo arquivo. Em seu `data.js` crie a função `criaArquivoDeCurso` como abaixo:

```
//data.js
const jsonfile = require('jsonfile-promised');
module.exports = {

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

}
```

4- Para que não criemos arquivos com nomes repetidos, vamos utilizar o submódulo `fs` do node para detectar se um arquivo já existe antes de criá-lo. Importe o `fs` no topo do módulo `data.js`:

```
//data.js
const jsonfile = require('jsonfile-promised');
const fs = require('fs');

module.exports = {
    ...
}
```

5- Vamos criar também a função que salva os dados do curso em um arquivo. Em seu `data.js` crie a função `salvaDados` , que recebe o nome do curso e o tempo estudo como parâmetros:

```
//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)){
            //Salvar Dados
        }else{
            // Criar Arquivo
            this.criaArquivoDeCurso(arquivoDoCurso, {})
                .then(()=>{
                    // Salvar Dados
                })
        }
    },
    criaArquivoDeCurso(nomeArquivo,conteudoArquivo){
        ...
    }

}
```

6- Como precisamos que adicionar o tempo estudado a um arquivo JSON, vamos criar uma nova função especialista nisso, a função `adicionaTempoAoCurso` , que vai receber o caminho para o arquivo do curso e o tempo estudado. Ela vai criar o objeto JSON que será salvo no arquivo e efetivamente realizar a escrita:

```
//data.js
const jsonfile = require('jsonfile-promised');
const fs = require('fs');
module.exports = {

    salvaDados(curso, tempoEstudado){
        ...
    },

    adicionaTempoAoCurso(arquivoDoCurso, tempoEstudado){
        let dados = {
            ultimoEstudo : new Date().toString(),
            tempo: tempoEstudado
        }
```

```
            jsonfile.writeFile(arquivoDoCurso, dados)
                .then(()=>{
                    console.log('Tempo salvo com sucesso')
                }).catch((err) => {
                    console.log(err);
                })
        },

        criaArquivoDeCurso(nomeArquivo,conteudoArquivo){
            ...
        }

    }
```

7- Por último, vamos adicionar um parâmetro opcional na função `adicionaTempoAoCurso` para que nossos JSON sejam escritos com uma identação melhor. Adicione como terceiro parâmetro da função `writeFile` o objeto `{ spaces: 2}` :

```
    //data.js
    const jsonfile = require('jsonfile-promised');
    const fs = require('fs');
    module.exports = {

        salvaDados(curso, tempoEstudado){
            ...
        },

        adicionaTempoAoCurso(arquivoDoCurso, tempoEstudado){
            let dados = {
                ultimoEstudo = new Date().toString(),
                tempo: tempoEstudado
            }

            jsonfile.writeFile(arquivoDoCurso, dados, {spaces: 2})
                .then(()=>{
                    console.log('Tempo salvo com sucesso')
                }).catch((err) => {
                    console.log(err);
                })
        },

        criaArquivoDeCurso(nomeArquivo,conteudoArquivo){
            ...
        }

    }
```

8- Agora vamos chamar nossa função `adicionaTempoAoCurso` nos locais aonde precisamos salvar dados da função `salvaDados` :

```
    //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)){
            //Salvar Dados
            this.adicionaTempoAoCurso(arquivoDoCurso, tempoEstudado);
        }else{
            // Criar Arquivo
            this.criaArquivoDeCurso(arquivoDoCurso, {})
                .then(()=>{
                    // Salvar Dados
                    this.adicionaTempoAoCurso(arquivoDoCurso, tempoEstudado);
                })
        }
    },

    adicionaTempoAoCurso(arquivoDoCurso, tempoEstudado){
        ...
    },

    criaArquivoDeCurso(nomeArquivo,conteudoArquivo){
        ...
    }

}
```

9- Agora basta chamar a nossa função `salvaDados` no processo principal em `main.js` que quando escutarmos o evento de `curso-parado` o nosso módulo de `data.js` já vai ficar responsável por salvar os nosso dados:

Importando o `data.js` no topo do processo principal:

```
//main.js
const { app, BrowserWindow, ipcMain } = require('electron');
const data = require('./data');
```

e em seguida chamando a função salvar no evento `curso-parado`

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

    data.salvaDados(curso,tempoEstudado);
})
```