Desafio Rocky

Manual de documentação

Autor: Paulo Cesar Ferreira Junior

Sorocaba-SP

07/2020

Sumário

1 Explicando as funcionalidades

Para a execução das funcionalidades foi utilizado a linguagem JavaScript, utilizado como referência o site: https://nodejs.org/docs/latest-v13.x/api/

1.1 Leitura do arquivo Json

Foi criado uma variável chamada banco e importado a ela o documento broke-database.json, utilizando a função nativa require.

```
let banco = require('./broken-database.json');
```

1.2 Correção do nome dos produtos

Feito a criação da função corrigirCaracter que recebe um array de objetos json chamado banco e corrigi os caracteres que foram alterados para o campo name.

Utilizando a tabela Unicode para fazer a comparação dos caracteres que foram alterados e com o auxílio da função replace, trocando para o estado original.

```
function corrigirCaracter(banco) {
    try {
        banco.forEach(n => {
            n.name = n.name.replace(/\u00DF/g, "b");
            n.name = n.name.replace(/\u00F8/g, "o");
            n.name = n.name.replace(/\u00E6/g, "a");
            n.name = n.name.replace(/\u00A2/g, "c");
        });
    } catch (ex) {
        console.log('Problema ao receber o banco de arquivos json: ', ex);
    }
}
```

1.3 Correção do preço dos produtos

Feito a criação da função corrigirPreco que recebe um array de objetos json chamado banco e corrigi os preços, alterando o campo price do tipo de string para number, apenas para os campos prices que estão como string.

Utilizado como validação um if que valida se o campo price é igual ao tipo string, se for verdadeira o campo é alterado para o tipo number.

```
function corrigirPreco(banco) {
    try {
        banco.forEach(n => {
            if (typeof n.price == 'string') {
                n.price = Number(n.price);
            }
        });
    } catch (ex) {
        console.log('Problema ao receber o banco de arquivos json: ', ex);
    }

}
```

1.4 Correção da quantidade dos produtos

Feito a função corrigirQuantidade que recebe um array de objetos json chamado de banco e cria o campo quantity para os objetos json que não tem este campo.

O array banco é percorrido e a cada iteração é feito uma pergunta através de if para saber se aquele campo não existe, se não ele não existir é criado para aquele objeto json o campo quantity e atribui o valor 0 ao mesmo.

```
function corrigirQuantidade(banco) {
    try {
        banco.forEach(n => {
            if (n.quantity === undefined) {
                n.quantity = 0;
            }
        });
    } catch (ex) {
        console.log('Problema ao receber o banco de arquivos json: ', ex);
    }

}
```

1.5 Exportando o objeto json para um arquivo

Feito a função exportarArquivo que recebe um array de objetos json chamado de banco e cria um arquivo.json no mesmo diretório que o script encontra-se.

Para isto foi utilizado a função writeFile da classe fs nativa do nodejs, que permite fazer ações como leitura, escrita, exclusão e criação de arquivos, caso a haja algum erro no momento da exportação a writeFile interrompe o procedimento e imprime a exceção no console.

```
function exportarArquivo(banco) {
    banco = JSON.stringify(banco);
    fs.writeFile('./saida.json', banco, (err) => {
        if (err) {
            console.log(err);
        }
    });
}
```

1.6 Impressão da lista com o nome dos produtos

Feito a função imprimeNome que recebe um array de objetos json chamado de banco e cria uma lista com o nome dos produtos, ordenados por category e id.

Para isto foi utilizado a função sort que ordena o array banco pelo o campo category e id, assim que o array de objetos json estiver ordenado é utilizado a função map que passa o nome dos produtos para um array chamado lista e o mesmo é impresso.

```
function imprimeNome(banco) {
    try {
        banco.sort(function(a, b) {
            return a.category < b.category ? -1 : a.category > b.category ? 1 : (a.id < b.id ? -1 : a.id > b.id ? 1 : 0);
        });
        try {
            let lista = banco.map(n => {
                return n.name;
            });
            console.log('Lista de Produtos: ', lista);
        } catch (ex) {
            console.log('Houve um problema ao gerar a lista', ex);
        }
    } catch (ex) {
        console.log('Problema ao receber o banco de arquivos json: ', ex);
    }
}
```

1.7 Calculando o valor total dos produtos por categoria

Feito uma função calculaEstoque que recebe um array de objetos json chamado banco, cria um array estoque contendo os campos category que são preenchidos com as categorias (Acessórios, Eletrodomésticos, Eletrônicos e Panelas) e valor é atribuído o 0.

O banco é percorrido e o dentro desta iteração é percorrido o array estoque e feito a pergunta com if que se o campo category for igual ao campo category do banco, caso seja a multiplicação do campo price com o campo quantity do array banco é somada ao campo valor do array estoque, feito para cada iteração até percorre o último objeto do array estoque e depois até percorrer o cada objeto do array banco.

```
function calculaEstoque(banco) {
    try {
        let estoque = [{
                category: 'Acessórios',
                valor: 0
            }, {
                category: 'Eletrodomésticos',
                valor: 0
            }, {
                category: 'Eletrônicos',
                valor: 0
            }, {
                category: 'Panelas',
                valor: 0
            }

        ];

        banco.forEach(n => {
            estoque.forEach(e => {
                if (n.category == e.category) {
                    e.valor = e.valor + (n.price * n.quantity);
                }
            })
        });

        return estoque;
    } catch (ex) {
        console.log('Problema ao receber o banco de arquivos json: ', ex);
    }

}
```

1.8 Utilização das funções

Para a utilização das funções, foram feitas as chamadas em sequências ao final do escopo da última função.

```
corrigirCaracter(banco);
corrigirPreco(banco);
corrigirQuantidade(banco);
imprimeNome(banco);
console.log('Estoque total: ', calculaEstoque(banco));
exportarArquivo(banco);
```

2 Escolha da Linguagem

As duas linguagens são fascinantes e confesso que gosto do python comecei a estudar um pouco tempo, e, como realizei meu trabalho de conclusão do curso com nodejs já havia um contato com javaScript, tive que aprender algumas coisas novas para realizar este teste, contudo gostei muito do resultado.

Por este motivo escolhi fazer com javaScript, mas nada emplacaria em fazer em python.

3 Tratamento de erros

Para tratar e evitar erros, ame algumas funções foram utilizado ifs e para quase todas as funções foram utilizados os comando try e catch que são blocos para tratar exceções caso ocorra, apenas a função exportarArquivo que não foi necessário a utilização do escopo try e catch pois a função writeFile tem a tratativa de erros nativa.

4 Outros

Gostaria de ter feito em python pois quando comecei a estudar a linguagem fiquei muito interessado contudo achei que não daria tempo de aprender o suficiente para este desafio, utilizei algumas informações do canal Programador a Bordo pelo o youtube para entender melhor a utilização das funções foreach, map e utilização da operação ternária.

Fiquei muito feliz com este desafio, por ter visto e aprendido coisas novas e espero ter pelo conseguido realizar o esperado.