ROCKY

FULL DIGITAL PERFORMANCE

Projeto Classificatório

Processo seletivo - Web Development

Diogo Rodrigo Espíndola Franco

Explicação do algoritmo

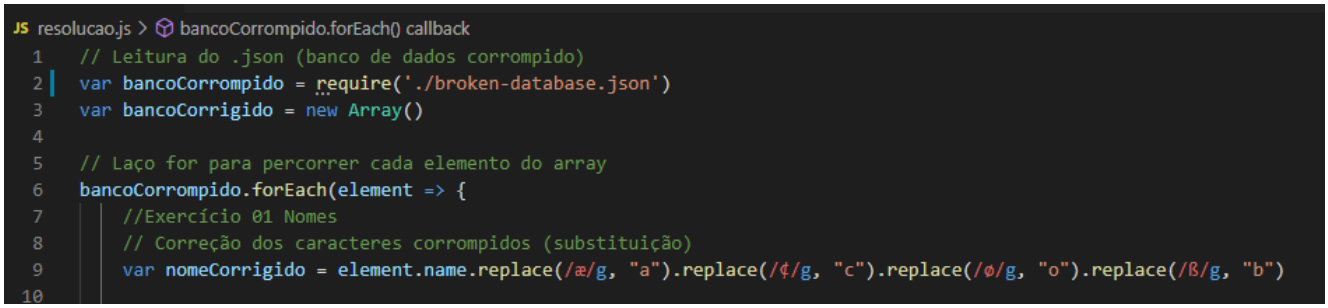
Sorocaba, 21 de junho de 2021

Problemas detectados no banco de dados corrompido

1. **Nomes**
   Todos os nomes de produto tiveram alguns caracteres modificados, houve substituição de todos os "a" por "æ", "c" por "¢", "o" por "ø", "b" por "ß". É preciso reverter essas substituições para recuperar os nomes originais:

Resolução:

Criei uma nova array bancoCorrigido para receber a correção, utilizei o forEach para percorrer cada elemento do array e o replace para realizar as substituições, segue exemplo abaixo:



```
JS resolucao.js > bancoCorrompido.forEach() callback
1   // Leitura do .json (banco de dados corrompido)
2   var bancoCorrompido = require('./broken-database.json')
3   var bancoCorrigido = new Array()
4
5   // Laço for para percorrer cada elemento do array
6   bancoCorrompido.forEach(element => {
7       //Exercício 01 Nomes
8       // Correção dos caracteres corrompidos (substituição)
9       var nomeCorrigido = element.name.replace(/æ/g, "a").replace(/¢/g, "c").replace(/ø/g, "o").replace(/ß/g, "b")
10
```

- **forEach()** - permite executar uma função para cada item de um array
- **replace** - replace significa substituir e é isso o que o método replace() faz, ou seja, substitui um trecho de uma string por outro e retorna a operação em uma nova string

Referências:

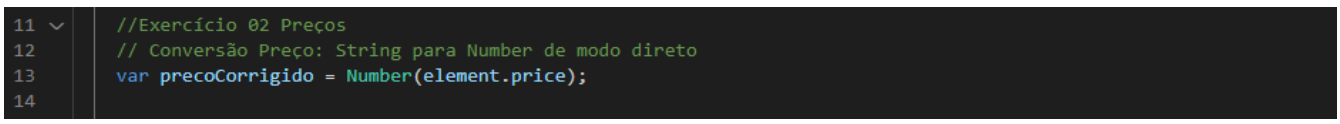
forEach: (https://www.devmedia.com.br/javascript-foreach-executando-uma-funcao-para-cada-elemento-de-um-array/39808)

replace: https://blog.betrybe.com/javascript/javascript-replace/

2. **Preços**
   Os preços dos produtos devem ser sempre do tipo number, mas alguns deles estão no tipo string. É necessário transformar as strings novamente em number.

Resolução:

No exemplo abaixo utilizei o recurso de conversão de string em number, segue exemplo abaixo:



```
11      //Exercício 02 Preços
12      // Conversão Preço: String para Number de modo direto
13      var precoCorrigido = Number(element.price);
14
```

Referências:

Após várias pesquisas recorri auxilio a um mentor, no qual após realizar uma análise em meu código me pontuou onde poderia ser realizado melhorias, garantindo a funcionabilidade.

3. **Quantidades**
   Nos produtos onde a quantidade em estoque era zero, o atributo "quantity" sumiu.
   Ele precisa existir em todos os produtos, mesmo naqueles em que o estoque é 0.

Resolução:

No exemplo abaixo utilizei condicional if e else, criei uma variável quantidadeCorrigida que receberá o valor 0 caso a quantidade for null, ou seja, vazio. Caso contrário receberá o valor estipulado pelo sistema, segue exemplo abaixo:

```
15    //Exercício 03 Quantidades
16    // Volta o valor 0 onde o estoque estiver vazio
17    var quantidadeCorrigida
18    if (element.quantity == null) {
19        quantidadeCorrigida = 0
20    }
21    else {
22        quantidadeCorrigida = element.quantity
23    }
24
```

Referências:

Nesse caso eu já havia visto condicional em C, C++ e Java.

Recuperação dos dados originais do banco de dados

A) Ler o Arquivo Json:

Resolução:

No exemplo abaixo criei uma variável bancoCorrompido para receber os dados do json. Utilizei o recurso **require** para importar o arquivo /broken-database.json.

```
JS resolucao.js > ...
1   // Leitura do .json (banco de dados corrompido)
2   var bancoCorrompido = require('./broken-database.json')
```

- **Require:** O **require** é uma forma que foi desenvolvida para **Node.js** de importar e exportar módulos em uma aplicação

Referências:

https://www.horadecodar.com.br/2020/09/08/diferencas-entre-import-e-require-javascript/

E) Exportar um arquivo JSON com o banco corrigido:

Resolução:

No exemplo abaixo utilizei o argumento **JSON.stringify()** permite que o conteúdo seja exibido no formato correto e permite ao usuário definir o espaçamento que se adapta à sua legibilidade.
File Save utilizado para salvar o arquivo na pasta e criar um novo banco de dados saida.json

```
// Conversão de array para json Pretty-printing (melhorar a aparência)
//stringfy = Transforma o array em uma string do tipo .json
const bancoCorrigidoJsonArray = JSON.stringify(bancoCorrigido, null, 4)
//fs = File save para salvar o arquivo na pasta e criar um novo banco de dados
const fs = require('fs')
fs.writeFileSync('saida.json', bancoCorrigidoJsonArray)
```

Referências:

https://www.delftstack.com/pt/howto/javascript/json-stringify-pretty/

Validação do banco de dados corrigido

A) Uma função que imprime a lista com todos os nomes dos produtos, ordenados primeiro por categoria em ordem alfabética e ordenados por id em ordem crescente. Obs: é apenas uma saída, ordenada pelos dois fatores citados acima.

Resolução:

No exemplo abaixo utilizei uma função para **ordenar** as categorias e os id's, ela faz a comparação e atribui -1 ou +1 para ordenar mediante o resultado da condicional **if**.

```
//função utilizada para ordenação de categoria e id.
function CompareCategoria( a, b ) {
  //https://stackoverflow.com/questions/1129216/sort-array-of-objects-by-string-property-value
  if ( a.category < b.category ){
    return -1;
  }
  if ( a.category > b.category ){
    return 1;
  }
  return 0;
}

function CompareId( a, b ) {
  //https://stackoverflow.com/questions/1129216/sort-array-of-objects-by-string-property-value
  if ( a.id < b.id ){
    return -1;
  }
  if ( a.id > b.id ){
    return 1;
  }
  return 0;
}
```

Referências:

- https://stackoverflow.com/questions/1129216/sort-array-of-objects-by-string-property-value
- *Após várias pesquisas recorri auxilio a um mentor, no qual após realizar uma análise em meu código me pontuou onde poderia ser realizado melhorias, garantindo a funcionabilidade.*

B) Uma função que calcula qual é o valor total do estoque por categoria, ou seja,a soma do valor de todos os produtos em estoque de cada categoria, considerando a quantidade de cada produto.

Resolução:

No exemplo abaixo utilizei **.filter** para realizar um filtro, **.map** para retornar a quantidade e **.reduce** para realizar a soma de modo acumulativo uma em cima da outra.

```
73 //Retorna a lista de categoria distinta(um de cada) -
74 //https://stackoverflow.com/questions/15125920/how-to-get-distinct-values-from-an-array-of-objects-in-javascript
75 const categoriasDistintas = [...new Set(banco.map(item => item.category))];
76
77 categoriasDistintas.forEach(categoria => {
78   //https://medium.com/poka-techblog/simplify-your-javascript-use-map-reduce-and-filter-bd02c593cc2d
79   const totalCategoria = banco
80   // Filtra
81   .filter(item => item.category === categoria)
82   //Map = Retorna quantidade
83   .map(item => item.quantity)
84   //Reduce Soma de maneira acumulativa - uma em cima da outra.
85   .reduce((acumulado, quantidade) => acumulado + quantidade, 0);
86
87   console.log("Categoria: " + categoria + " - Total: " + totalCategoria)
88
89 });
```

Referências:

- https://medium.com/poka-techblog/simplify-your-javascript-use-map-reduce-and-filter-bd02c593cc2d
- https://stackoverflow.com/questions/15125920/how-to-get-distinct-values-from-an-array-of-objects-in-javascrip
- *Após várias pesquisas recorri auxilio a um mentor, no qual após realizar uma análise em meu código me pontuou onde poderia ser realizado melhorias, garantindo a funcionabilidade.*

O porque da escolha da linguagem?

A linguagem JavaScript foi definida pela empresa ROCKY FULL DIGITAL PERFORMANCE (Recrutadora)

Primeiramente gostaria de agradecer a oportunidade e dizer que sou muito grato em poder participar dessa etapa nesse processo seletivo. Confesso que foi desafiador e que no início não sabia nem por onde começar pois foi meu primeiro teste, porém a determinação e a força de vontade em aprender e concluir esse desafio falou mais alto do que qualquer dificuldade. Recorri ao Google, Youtube e a um mentor para aprender e executar o teste, confesso que aprendi muito e acabei me identificando ainda mais com a área de desenvolvimento, independentemente do resultado desta etapa será um aprendizado que levarei comigo para o resto da minha vida.