Aula 01: Leitura de arquivos em Node JS

Sumário

Introdução ao Node JS

O que é Node.js?

Node.js é um ambiente de execução JavaScript assíncrono e baseado em eventos, construído sobre o motor V8 do Google Chrome. Ele permite que os desenvolvedores usem JavaScript no lado do servidor para construir aplicações web rápidas e escaláveis. Node.js é amplamente utilizado para construir servidores web, APIs, aplicativos de tempo real, entre outras coisas.

Como o Node.js funciona?

Node.js utiliza o motor V8 do Google Chrome, que é responsável por compilar e executar o código JavaScript. O V8 é conhecido por sua alta performance e eficiência, permitindo que Node.js processe operações de I/O de forma assíncrona. Isso significa que, ao invés de bloquear a execução enquanto espera por uma operação de I/O (como leitura de arquivos, consultas de banco

de dados, etc.), o Node.js continua a executar outras tarefas e usa um sistema de callbacks para lidar com a resposta quando a operação de I/O estiver concluída.

Vantagens do Node.js

- **Alta Performance**: O motor V8 compila JavaScript para código de máquina, resultando em execução rápida.
- **Escalabilidade**: A natureza assíncrona e baseada em eventos do Node.js permite a criação de aplicativos escaláveis.
- **Unificação da Linguagem**: Os desenvolvedores podem usar JavaScript tanto no front-end quanto no back-end, promovendo consistência.

Leitura Síncrona de Arquivos

```
const fs = require('fs');

try {
    // Passo como parametro o caminho do arquivo e o encoding
    // Caso der erro de leitura, será lençado um Error
    const data = fs.readFileSync('caminho/do/arquivo.txt', 'u
    console.log(data);
} catch (err) {
    console.error(err);
}
```

real_names_and_grades.txt

Escreva um programa que leia o conteúdo de um arquivo chamado `notas.txt` e exiba o conteúdo no console. O arquivo deve ser lido de forma síncrona.

conteudo2.txt

Crie um programa que leia dois arquivos (`arquivo1.txt` e `arquivo2.txt`) de forma síncrona e escreva o conteúdo

conteudo.txt

combinado no console. E conte quantas palavras tem nos dois arquivos.

Leitura Assíncrona de Arquivos

```
const fs = require('fs');

// Aqui preciso passar três parâmetros para a função
// Os já mencionados
// Mais a função de callback que será responsável por ler o a
fs.readFile('caminho/do/arquivo.txt', 'utf8', (err, data) =>
    if (err) {
        console.error(err);
        return;
    }
    console.log(data);
});
```

Escreva um programa que leia o conteúdo de um arquivo chamado `tarefas.txt` e exiba o conteúdo no console. O arquivo deve ser lido de forma assíncrona.

daily_tasks.txt

Modifique o programa anterior para contar e exibir o número de tarefas no arquivo `tarefas.txt`.

Usando Promises para Leitura de Arquivos

```
const fs = require('fs').promises;

async function lerArquivo() {
    try {
        const data = await fs.readFile('caminho/do/arquivo.tx
        console.log(data);
    } catch (err) {
        console.error(err);
    }
}

lerArquivo();
```

Leitura de Arquivos em Partes (Streams)

```
const fs = require('fs');

// crio o strem de leitura e passo dois parâmetros
// caminho do arquivo e objeto com opções
// nas opções posso definir algumas coisas como
// encoding
// tamanho do chunk (highWaterMark) em bytes
const stream = fs.createReadStream('caminho/do/arquivo.txt',

// Stream dispara alguns eventos no ciclo de vida de leitura
stream.on('data', (chunk) => {
    console.log(chunk);
});

stream.on('end', () => {
    console.log('Leitura concluída.');
});

stream.on('error', (err) => {
    console.error(err);
});
```

livro.txt

Escreva um programa que leia o conteúdo de um arquivo chamado `livro.txt` usando streams e exiba cada pedaço de dados no console.

Modifique o programa anterior para contar o número de pedaços lidos e exibir esse número no final.

Leitura de Arquivos JSON

```
const fs = require('fs').promises;

async function lerArquivoJSON() {
    try {
            // Leio o arquivo texto normalmente e no final tr
            // usando JSON.parse
        const data = await fs.readFile('caminho/do/arquivo.js
        const jsonData = JSON.parse(data);
        console.log(jsonData);
    } catch (err) {
        console.error(err);
    }
}

lerArquivoJSON();
```

Escreva um programa que leia um arquivo JSON chamado `dados.json` e exiba o conteúdo no console.

Observando Mudanças em Arquivos

```
const fs = require('fs');
```

```
fs.watch('caminho/do/arquivo.txt', (eventType, filename) => {
    if (filename) {
        console.log(`Arquivo modificado: ${filename}`);
        console.log(`Tipo de evento: ${eventType}`);
    }
});
```