Exemplo 1

const http = require('http')

const port = 3000

const server = http.createServer((req, res) => {

  res.write('Oi HTTP')

  res.end()

})

server.listen(port, () => {

  console.log(`Servidor rodando na porta: ${port}`)

})

2 – retornando html

import http from 'http'; // Importação no formato ESM

const port = 3000;

// Criação do servidor HTTP

const server = http.createServer((req, res) => {

  res.statusCode = 200; // Define o status HTTP como 200 (OK)

  res.setHeader('Content-Type', 'text/html; charset=utf-8'); // Define o tipo de conteúdo e o charset

  res.end('<h1>Olá, este é o meu primeiro server!</h1>'); // Responde com HTML

});

// Inicia o servidor na porta especificada

server.listen(port, () => {

  console.log(`Servidor rodando na porta: ${port}`);

});

URL

import { URL } from 'url';

const address = 'https://www.meusite.com.br/catalogo?produtos=cadeira';

const parsedUrl = new URL(address);

console.log(parsedUrl.host);

console.log(parsedUrl.pathname);

console.log(parsedUrl.search);

console.log(parsedUrl.searchParams);

console.log(parsedUrl.searchParams.get('produtos'));

HTTP com URL

import http from "http";

import { URL } from "url";

const port = 3000;

const server = http.createServer((req, res) => {

  const urlInfo = new URL(req.url, `http://localhost:${port}`); // Cria um objeto URL com base na requisição

  const name = urlInfo.searchParams.get("name"); // Obtém o parâmetro 'name'

  console.log(name);

  res.statusCode = 200;

  res.setHeader("Content-Type", "text/html; charset=utf-8");

  if (!name) {

    res.end(

      "<h1>Preencha seu nome:</h1><form method='GET'><input type='text' name='name'/><input type='submit' value='Enviar'></form>"

    );

  } else {

    res.end(`<h1>Seja bem-vindo ${name}!</h1>`);

  }

});

server.listen(port, () => {

  console.log(`Servidor rodando na porta: ${port}`);

});

FS

import http from "http";

import fs from "fs";

const port = 3000;

const server = http.createServer((req, res) => {

  fs.readFile("mensagem.html", (err, data) => {

    if (err) {

      res.writeHead(500, { "Content-Type": "text/plain; charset=utf-8" });

      res.write("Erro ao ler o arquivo.");

      return res.end();

    }

    res.writeHead(200, { "Content-Type": "text/html; charset=utf-8" });

    res.write(data);

    res.end();

  });

});

server.listen(port, () => {

  console.log(`Servidor rodando na porta: ${port}`);

});

Escrevendo\_Arquivos

import http from "http";

import fs from "fs";

import { URL } from "url";

const port = 3000;

const server = http.createServer((req, res) => {

  const urlInfo = new URL(req.url, `http://localhost:${port}`); // Cria um objeto URL

  const name = urlInfo.searchParams.get("name"); // Obtém o parâmetro 'name'

  res.statusCode = 200;

  res.setHeader("Content-Type", "text/html; charset=utf-8");

  if (!name) {

    // Lê o arquivo index.html

    fs.readFile("index.html", (err, data) => {

      if (err) {

        res.writeHead(500, { "Content-Type": "text/plain; charset=utf-8" });

        res.write("Erro ao carregar o arquivo.");

        return res.end();

      }

      res.writeHead(200, { "Content-Type": "text/html; charset=utf-8" });

      res.write(data);

      res.end();

    });

  } else {

    // Escreve o nome no arquivo arquivo.txt

    fs.writeFile("arquivo.txt", name, (err) => {

      if (err) {

        res.writeHead(500, { "Content-Type": "text/plain; charset=utf-8" });

        res.write("Erro ao salvar o arquivo.");

        return res.end();

      }

      // Redireciona para a página inicial

      res.writeHead(302, { Location: "/" });

      res.end();

    });

  }

});

server.listen(port, () => {

  console.log(`Servidor rodando na porta: ${port}`);

});

Atualizando Arquivos

import http from "http";

import fs from "fs";

import { URL } from "url";

const port = 3000;

const server = http.createServer((req, res) => {

  const urlInfo = new URL(req.url, `http://localhost:${port}`); // Cria um objeto URL

  const name = urlInfo.searchParams.get("name"); // Obtém o parâmetro 'name'

  res.statusCode = 200;

  res.setHeader("Content-Type", "text/html; charset=utf-8");

  if (!name) {

    // Lê o arquivo index.html

    fs.readFile("index.html", (err, data) => {

      if (err) {

        res.writeHead(500, { "Content-Type": "text/plain; charset=utf-8" });

        res.write("Erro ao carregar o arquivo.");

        return res.end();

      }

      res.writeHead(200, { "Content-Type": "text/html; charset=utf-8" });

      res.write(data);

      res.end();

    });

  } else {

    // Adiciona o nome ao arquivo arquivo.txt

    const nameNewLine = name + "\r\n";

    fs.appendFile("arquivo.txt", nameNewLine, (err) => {

      if (err) {

        res.writeHead(500, { "Content-Type": "text/plain; charset=utf-8" });

        res.write("Erro ao salvar o arquivo.");

        return res.end();

      }

      // Redireciona para a página inicial

      res.writeHead(302, { Location: "/" });

      res.end();

    });

  }

});

server.listen(port, () => {

  console.log(`Servidor rodando na porta: ${port}`);

});

Removendo Arquivos

import fs from 'fs';

fs.unlink('arquivo.txt', (err) => {

  if (err) {

    console.log(err);

    return;

  }

  console.log('Arquivo removido!');

});

Renomeando Arquivos

import fs from 'fs';

fs.rename('arquivo.txt', 'novoarquivo.txt', (err) => {

  if (err) {

    console.log(err);

    return;

  }

  console.log('Arquivo renomeado!');

});

Detalhamento de Arquivos

import fs from 'fs';

fs.stat('novoarquivo.txt', (err, stats) => {

  if (err) {

    console.error(err);

    return;

  }

  console.log(stats.isFile());

  console.log(stats.isDirectory());

  console.log(stats.isSymbolicLink());

  console.log(stats.ctime);

  console.log(stats.size);

});

Roteamento

import http from 'http';

import fs from 'fs';

import { URL } from 'url';

const port = 3000;

const server = http.createServer((req, res) => {

  const q = new URL(req.url, `http://localhost:${port}`);

  const filename = q.pathname.substring(1);

  console.log(filename);

  if (filename.includes('html')) {

    if (fs.existsSync(filename)) {

      fs.readFile(filename, (err, data) => {

        if (err) {

          res.writeHead(500, { 'Content-Type': 'text/html' });

          res.write('<h1>Erro ao ler o arquivo</h1>');

          return res.end();

        }

        res.writeHead(200, { 'Content-Type': 'text/html' });

        res.write(data);

        return res.end();

      });

    } else {

      fs.readFile('404.html', (err, data) => {

        if (err) {

          res.writeHead(500, { 'Content-Type': 'text/html' });

          res.write('<h1>Erro ao ler o arquivo 404</h1>');

          return res.end();

        }

        res.writeHead(404, { 'Content-Type': 'text/html' });

        res.write(data);

        return res.end();

      });

    }

  }

});

server.listen(port, () => {

  console.log(`Servidor rodando na porta: ${port}`);

});

Path

import path from 'path';

const customPath = '/relatorios/Joao/relatorio1.pdf';

console.log(path.dirname(customPath));

console.log(path.basename(customPath));

console.log(path.extname(customPath));

Path Absoluto

import path from 'path';

// path absoluto

console.log(path.resolve('teste.txt'));

// formar path

const midFolder = 'relatorios';

const fileName = 'matheus.txt';

const finalPath = path.join('/', 'arquivos', midFolder, fileName);

console.log(finalPath);

Trabalhando com dirs.

import fs from 'fs';

// Verifica se a pasta não existe

if (!fs.existsSync('./minhapasta')) {

  console.log('Não existe');

}

// Cria a pasta

fs.mkdirSync('minhapasta');

// Verifica se a pasta foi criada

if (fs.existsSync('minhapasta')) {

  console.log('Existe');

}

Módulos OS

import os from 'os';

console.log(os.cpus());

console.log(os.freemem());

console.log(os.homedir());

console.log(os.type());