```
//LENDO E ESCREVENDO ARQUIVOS NO NODE:


//Existe uma biblioteca core do Node (só relembrando, bibliotecas core, são bibliotecas já embutidas no Node) chamada "fs", que significa "file sistem". Por através dessa biblioteca é possível ler arquivos e pastas no nosso servidor, bem como é possível escrever nesses arquivos.

//O fs é preparado para ler arquivos de forma síncrona - onde o Node pausa o event loop somente para ler o arquivo, o que significa que o nosso sistema fica travado enquanto o arquivo não for lido - e assíncrona - onde o event loop não é pausado, o arquivo é lido enquando o Node vai executando outras tarefas.

//Vejamos como utilizar o "fs", criamos um arquivo "141-Arquivo_JSON.json" para ser lido pelo fs...

//IMPORTANDO BIBLIOTECA FS:
//O primeiro passo é a importação da biblioteca, podemos importá-la diretamente pelo "require()", não precisamos baixá-la, pois já é uma biblioteca core do Node...
const fs = require('fs')
const { type } = require('os')



//ATRIBUIÇÃO DE CAMINHO RELATIVO DO ARQUIVO A SER LIDO:
//A biblioteca fs precisa encontrar o caminho do arquivo, para isso usamos o comando "__dirname" que procurará pelo arquivo no caminho relativo que nós apresentarmos...
const caminho = __dirname + '/141-Arquivo_JSON.json'



//LEITURA FEITA DE FORMA SÍNCRONA:
//Para fazer a leitura síncrona do arquivo usamos o método "readFileSync" da biblioteca fs, esse método recebe como parâmetros o caminho do arquivo e o encoding dele, que nosso caso é o "utf-8"...
const conteudoSyn = fs.readFileSync(caminho, 'utf-8')
console.log('1)', conteudoSyn) //É importante salientar que o método readFileSync traduz o arquivo lido sempre em uma string...
console.log('Tipo de dado:', typeof conteudoSyn) //Se quiséssemos ter acesso aos dados do arquivo JSON teríamos que converter o arquivo para JSON, para então poder retirar os valores á partir dos atributos do objeto. Mas isso serve para qualquer tipo de arquivo
```

```
lido á partir da biblioteca "fs", teríamos que convertê-lo para o tipo de dado que desejássemos, só não teríamos que converter se o dado fosse uma string...



//LEITURA FEITA DE FORMA ASSÍNCRONA:
//Para fazer a leitura assíncrona do arquivo usamos o método "readFile" da biblioteca fs, esse método recebe 3 parâmetros o caminho do arquivo, o encoding dele e uma função callback com 2 parâmetros, onde 1 é o "err" que poderá exibir uma mensagem de erro caso o arquivo não seja encontrado e o outro é uma variável que conterá conteúdo do arquivo que sempre será lido como string...
const conteudoAssyn = fs.readFile(caminho, 'utf-8', (err, conteudo) => { //Veja que a função callback recebe os 2 parâmetros, um de erro e o outro vai trazer o conteúdo do arquivo que estiver no caminho relativo...
    const config = JSON.parse(conteudo) //Como o arquivo é convertido para string, tivemos que usar um JSON.parse() para converter a string para JSON, para que ele podesse ser lido...
    console.log(`\n2) Só fui executado Depois, por que sou assíncrono: ${config.db.host}: ${config.db.port}`) //Aqui nós retiramos os valores contidos em host e port...
})
//Perceba também que o conteúdo assíncrono só é executado depois, por que o Node continua executando os outros comandos no event loop e só quando a leitura do arquivo estiver pronta é que ele será apresentado...



//FACILIDADE DE LEITURA DOS ARQUIVOS JSON:
//Quando desejamos ler um arquivo de extensão ".json" podemos fazer a leitura desse arquivo somente usando o "require()" sem a necessidade de usar a biblioteca "fs". Por isso a utilização de arquivos ".json" é tão incentivada para a interoperabilidade dos dados na web.
const arquivoJson = require('./141-Arquivo_JSON.json') //Veja que bastou apenas chamar pelo arquivo JSON no require()
console.log(`\n3) ${arquivoJson.db.host}: ${arquivoJson.db.port}`) //Podemos acessar seu conteúdo facilmente graças ao JSON também ser um objeto...



//LENDO ARQUIVOS DE UMA PASTA ESPECÍFICA:
//Perceba que vamos usar a biblioteca fs para ler os arquivos que estão na pasta externa a essa pasta atual...
```

```
fs.readdir(__dirname + '/..', (err, arquivos) => { //Para isso usamos o métodos "readdir()", esse método recebe 2 parâmetros que são o caminho do arquivo, por através do método "__dirname", veja que usando "/.." nós saímos de uma pasta, e o outro parâmetro é uma função callback que recebe 2 parâmetros, um para o erro, e outro para os arquivos que se encontram na pasta...
    console.log('\n4) Conteúdo da pasta externa:')
    console.log(arquivos) //Os arquivos também são retornados em formato de um array que contém os nomes dos arquivos e pastas e formato de string...
    console.log('tipo retornado por readdir():', typeof arquivos)
    console.log('tipo dos valores dentro do array:', typeof arquivos[0])
})




//ESCREVENDO E PERSISTINDO EM ARQUIVO:
//Para que possamos escrever e salvar algo no Node temos que usar o método "writeFile()". Esse método recebe 3 parâmetros, primeiro o caminho do arquivo onde desejamos salvar as informações, segundo o tipo de dado que desejamos salvar - nesse quesito podemos usar um parse de acordo com o tipo de dado e terceiro uma função callback que recebe somente a mensagem de erro para que possamos tratar o erro da forma que desejarmos...
const produto = {
    nome: 'Tv',
    preço: 2159.99,
    modelo: 'Smart',
    peso: 2
}
fs.writeFile(__dirname + '/141-Arquivo_JSON_Escrito_por_Atraves_do_Node.json', JSON.stringify(produto), err => {
    console.log(err || '\n5) arquivo salvo!') //Perceba que salvamos o arquivo no formato json e incluímos na callback de erro a opção de mostrar o erro ou de mostrar que o arquivo foi salvo...
})
//Detalhe importante, quando salvamos com writeFile um arquivo que já existe, ele vai salvar por cima do arquivo existente substituindo ele...
```

ARQUIVO JSON LIDO...

```
{
    "db": {
```

```
        "host": "localhost",
        "port": 5432,
        "user": "usuário",
        "pass": "123456"
    }
}
```

ARQUIVO JSON ESCRITO E SALVO...

```
{"nome":"Tv","preço":2159.99,"modelo":"Smart","peso":2}
```

RESULTADO NO CONSOLE...

```
[Running] node "c:\Users\Almoxarifado\Documents\javascript\arquivos_das_aulas\141-Lendo_Arquivos_no_Node.js"
1) {
    "db": {
        "host": "localhost",
        "port": 5432,
        "user": "usuário",
        "pass": "123456"
    }
}
Tipo de dado: string

3) localhost: 5432

4) Conteúdo da pasta externa:
[
  '.git',
  '.gitignore',
  '.vscode',
  'arquivos_das_aulas',
  'consulta_rapida',
  'EXERCICIOS_01',
  'EXERCICIOS_02',
  'node_modules',
```

```
  'package-lock.json'
]
tipo retornado por readdir(): object
tipo dos valores dentro do array: string

5) arquivo salvo!

2) Só fui executado Depois, por que sou assíncrono: localhost: 5432

[Done] exited with code=0 in 0.123 seconds
```