

Acessando arquivos .env n 💆 🛈 🛅 🔘 🔊 🔍











Inscreva-se!

NODEJS JAVASCRIPT DEVELOPMENT

Acessando arquivos .env nativamente com Node.js

Já pensou carregar as suas variáveis de ambiente diretamente no Node.js? A versão 20.6 trás uma novidade incrível!

Compartilhe!



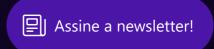














Começando a semana com uma notícia incrível e uma das novidades mais aguardadas do Node.js em anos, o suporte **nativo** ao uso de arquivos de ambiente, os chamados **dotenvs** ou **env-config**.

dotenv

Um dos pacotes mais comuns e mais utilizados em toda a comunidade, não só do JavaScript, é o pacote doteny. Ele faz algo que é super simples mas ao mesmo tempo essencial: carrega variáveis de ambiente na memória do runtime.

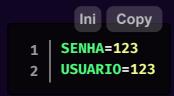
Variáveis de ambiente são variáveis especiais que carregam dados protegidos que não podem ser expostos no código e ficam apenas disponíveis no ambiente na qual elas estão inseridas, por exemplo, uma VM ou uma função serverless.

Eu escrevi bastante sobre elas nesse artigo – inclusive usando a antiga forma de carregar as variáveis, com o dotenv, então é até bacana para dar uma olhada em como o Node evoluiu desde lá – mas, essencialmente, variáveis de ambiente são o local ideal para armazenar usuários e senhas de bancos de dados, ou até mesmo



que é muito mais difícil do que ter acesso somente ao código.

Essas variáveis ficam armazenadas em arquivos que chamamos de do tenv, ou .env, que são arquivos definidos no formato INI ou bash dependendo do runtime que você está usando (para o dotenv seria o INI), então um arquivo de ambiente poderia ser assim:



Porém a gente precisava sempre de um módulo para poder carregar essas variáveis, até a chegada do Deno.

Deno

O Deno mudou bastante a forma que runtimes olham para esses arquivos porque ele permite que você carregue diretamente da sua standard library, ou seja, ao invés de termos que instalar um módulo a parte, podemos usar o que é nativo do runtime:

```
TypeScript Copy

// app.ts
import { load } from "https://deno.land/std@0.203.0/dotenv/mod.ts";

console.log(await load({export: true})); // { GREETING: "hello world" }

console.log(Deno.env.get("GREETING")); // hello world
```

Node 20

O Node 20.6 traz <u>uma mudança</u> fundamental que vai além até do que o Deno já fez, **agora é possível ler arquivos .env direto do runtime**.

Ao invés de termos que fazer:

```
Lucas
santos
         Acessando arquivos .env n 💆 🚺 🫅 🔊 🔊
                                                                    Inscreva-se!
                                                           Log in
            const {DB_HOST, DB_PORT, DB_USER, DB_PASS} = process.env
        2
            const db = require('alguma-lib-de-banco-de-dados')
        3
        4
            db.connect({
        5
                host: DB_HOST,
        6
                port: DB_PORT,
        7
                user: DB_USER,
        8
        9
                pass: DB_PASS
            })
       10
```

Podemos simplesmente utilizar o comando **node --env-file=<caminho> index.js** e todas as variáveis presentes no nosso arquivo definido no caminho serão carregadas no nosso **process.env**. Por exemplo, imagine um arquivo **.env** da seguinte forma:

```
Ini Copy

1 | DB_PASS=pass
2 | DB_USER=root
3 | DB_PORT=5432
4 | DB_HOST=localhost
```

Quando executarmos o nosso Node com o comando **node** --**env-file =.env index.js**, poderemos utilizar a variável **process.env.DB_PASS**, e ela vai ter o valor de **pass**, assim como as demais variáveis que estarão perfeitamente setadas na nossa aplicação para uso.

Mas essa não é a única novidade.

Node options

O Node.js permite que você crie uma variável de ambiente chamada NO DE_OPTIONS, ela permite que você defina todas as opções que seriam passadas ao node no comando principal diretamente na variável, ou seja, se quisermos, por exemplo, iniciar o Node com um inspetor (usando a flag --inspect) podemos utilizar: NODE_OPTIONS='--inspect' e agora todas as execuções de node index.js vão ser como se fossem escritas node --inspect index.js



opções do Node (exceto, claro, o próprio --env-file) em uma variável de ambiente:

```
1 | NODE_OPTIONS='--inspect --loader=tsx'
```

Como eu mostrei no meu artigo sobre TSX, podemos carregar arquivos TypeScript nativamente usando um loader, a ainda por cima podemos carregar o nosso arquivo de ambiente direto, com o comando acima vamos poder rodar node --env-file=.env index.ts, por exemplo.

Conclusão

Essa funcionalidade é gigante! Enquanto tira um pacote da nossa lista de dependências, ela ainda permite deixar o nosso código mais simples e direto, mas lembre-se que ela só está disponível a partir da versão **20.6**.

Compartilhe!



Assine a newsletter!

Receba conteúdos exclusivos diretamente no seu email!

Digite seu email...

Inscreva-se!

Você também pode gostar

Lucas santos

Acessando arquivos .env n 💆 🚺 🛅 🔘 🔊









Inscreva-se!

A versão 21.2 do Node é a mais especial de todas!

24 Nov 2023 – 6 minutos de leitura

Novidades do Node 21!

10 Nov 2023 – 3 minutos de leitura

Novidades do TS 5.3 - Beta!

1 Nov 2023 – 3 minutos de leitura

Como rodar TypeScript nativamente no Node.js com TSX

4 Out 2023 – 3 minutos de leitura

A novidades do TS 5.3

30 Ago 2023 – 4 minutos de leitura

Veja todos os 32 posts -



NODEJS

A versão 21.2 do Node é a mais especial de todas!

Vem comigo entender o que mudou na versão 21.2 do Node.js, por que essa é uma das versões mais especiais de todas! E o que isso significa!

24 NOV 2023 • 6 MINUTOS DE LEITURA



NODEJS

Novidades do Node 21!

Saiba tudo sobre as principais novidades do Node 21 e 21.1 que estão chegando por aí no Node.js!

10 NOV 2023 • 3 MINUTOS DE LEITURA

