

PROGRAMAÇÃO _

FRONT-END _

DATA SCIENCE _

INTELIGÊNCIA ARTIFICIAL _

DEVOPS _

UX & DESIGN _

MOBILE _

INOVAÇÃO & GESTÃO _

Artigos > **Programação**

Dotenv: gerenciando variáveis de ambiente



**Gabrielle Ribeiro
Gomes**
20/01/2023

COMPARTILHE

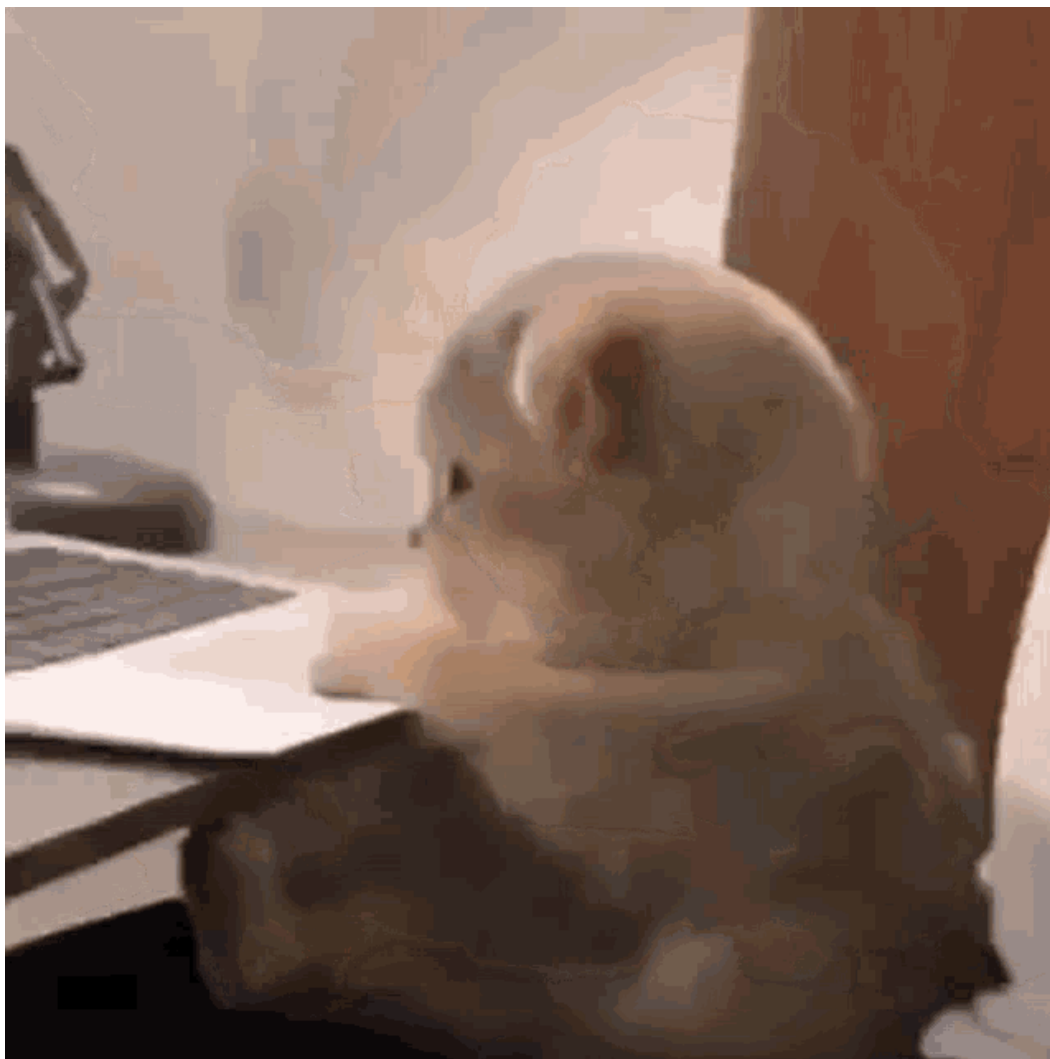




Introdução

O dia a dia de pessoas desenvolvedoras é cheio de situações desafiadoras.

Por exemplo, você já esteve no meio do desenvolvimento de uma aplicação e precisou conectar em outro banco de dados que não era o seu banco local? Ou subiu um código no Github e com o acesso ao banco local e acabou sobrescrevendo o acesso ao banco de dados e a aplicação caiu?



Fonte: media.tenor.com

Situações como essas são comuns quando estamos desenvolvendo projetos que possuem diversos ambientes sendo utilizados, tais como ambientes de desenvolvimento, testes e produção.

Com isso, se faz necessário o uso de diferentes variáveis de ambiente. Caso você queira se aprofundar mais sobre os diferentes ambientes, temos um artigo aqui na Alura sobre o assunto: [O que são ambientes?](#)



MATRICULE-SE

E como podemos lidar melhor com essas questões relacionadas a variáveis de ambiente dentro de projetos Node.js?

Existe uma ferramenta que pode nos auxiliar com isso: o pacote Dotenv. E, para conhecermos mais sobre esse assunto, neste artigo você verá:

- O que é o Dotenv;
- Como instalar o Dotenv;
- Como utilizar o Dotenv para facilitar o uso e gestão de variáveis de ambiente no desenvolvimento de suas aplicações com o Node.js.



O que é o Dotenv

Antes de entendermos o que é o Dotenv e o que ele faz, você se lembra o que são variáveis de ambiente?

As variáveis de ambiente guardam os dados de configuração do sistema. Elas geralmente armazenam informações sensíveis como, por exemplo, chaves de APIs, credenciais de acesso a bancos de dados, localização de arquivos, portas HTTPs, etc.

Já que as variáveis de ambiente contêm informações privadas, é importante que elas não fiquem expostas. Sendo assim, uma boa prática é mantê-las separadas do código.

O Dotenv é um pacote que serve justamente para gerenciar as variáveis de ambiente dentro de um projeto Node.js. Essa ferramenta armazena a configuração dessas variáveis em um ambiente separado do código da aplicação.



MATRICULE-SE

Manter um arquivo de configuração nos traz facilidades no desenvolvimento de aplicações, pois permite que os valores das variáveis de ambiente sejam facilmente alterados nos diferentes ambientes utilizados, sem a necessidade de alterar o código. Se faz necessário apenas alterar os valores das variáveis de ambiente no arquivo que as armazena.

Instalando o Dotenv

Imagine que você está desenvolvendo uma aplicação utilizando o Node.js e precisa acessar e manipular informações sobre os clientes em um banco de dados MySQL. Como as credenciais de acesso ao banco de dados são informações privadas, é uma boa ideia mantê-las separadas do código. Assim, você decide fazer a gestão dessas variáveis de ambiente através do Dotenv.

Mas então, como instalamos o Dotenv no nosso projeto?

Para instalar o Dotenv através do gerenciador de pacotes npm abra o terminal na pasta do seu projeto Node.js e digite o comando a seguir:



```
npm install dotenv
```

Para verificar se o Dotenv foi instalado corretamente faça o seguinte: ainda no terminal, digite o seguinte comando:



```
npm view dotenv version
```

Após digitar o comando, você poderá ver qual a **versão** instalada no seu projeto. Seu terminal vai exibir algo semelhante a imagem a seguir:

```
PS D:\Alura\artigo-dotenv> npm view dotenv version  
1.1.4
```

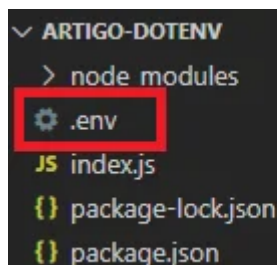
Além disso, o Dotenv deverá aparecer como uma das dependências do seu projeto dentro do arquivo `package.json`.

```
"dependencies": {  
  "dotenv": "^16.0.3"  
}
```

Agora que finalizamos a instalação do pacote, podemos seguir com a configuração e uso do Dotenv em nosso projeto. Vamos lá?!

Criando um arquivo `.env`

Com o pacote Dotenv instalado em seu projeto, o primeiro passo para utilizá-lo é criar um arquivo chamado `.env` dentro do diretório raiz da sua aplicação. Esse arquivo é responsável por guardar as variáveis de ambiente do seu projeto. A estrutura de arquivos do seu projeto ficará semelhante a imagem a seguir:



Vamos adicionar as credenciais de acesso ao seu banco de dados no arquivo `.env`. O formato utilizado neste arquivo é `NOME_VARIAVEL=VALOR_VARIAVEL`, sendo uma variável para cada linha do arquivo.

Desse modo, as nossas variáveis de ambiente referentes às credenciais de acesso ao banco de dados são: `HOST`, `USER`, `PASSWORD`, `DATABASE`.

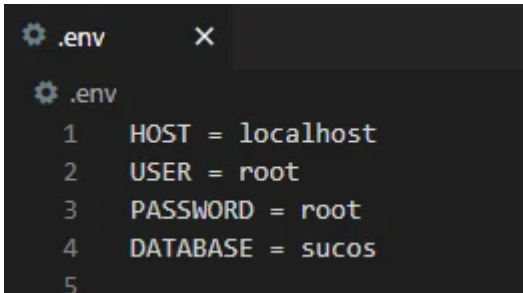
- `HOST`: que armazenará qual o ambiente que estamos utilizando, no nosso caso será o `localhost`, já que estamos no nosso computador local;
- `USER`: que armazenará qual o usuário vai acessar o banco de dados, no nosso caso é o `root`;

**alura**

- PASSWORD: que guardará qual a senha do usuário, no nosso caso a senha também é *root*;
- DATABASE: define qual base de dados será utilizada, no nosso projeto é a base *sucos*.

MATRICULE-SE

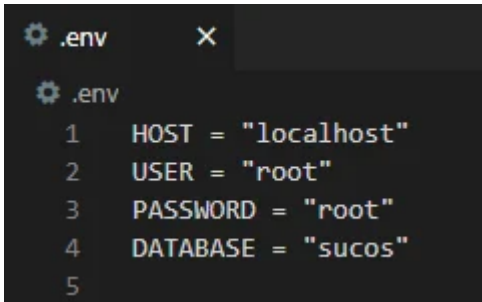
Ao adicionar essas variáveis e seus valores referentes, nosso arquivo `.env` ficará da seguinte maneira. Ah! E não se esqueça de salvar o arquivo. Confira a seguir:



```
.env
1  HOST = localhost
2  USER = root
3  PASSWORD = root
4  DATABASE = sucos
5
```

A propósito, nos arquivos `.env` o uso de aspas é opcional nos valores das variáveis, mesmo quando se trata de strings.

Caso você resolva adicionar aspas nos valores das strings, o arquivo continuará funcionando normalmente. Apesar de opcional, o uso das aspas pode ser interessante para manter o arquivo mais organizado. Confira na imagem abaixo:



```
.env
1  HOST = "localhost"
2  USER = "root"
3  PASSWORD = "root"
4  DATABASE = "sucos"
5
```

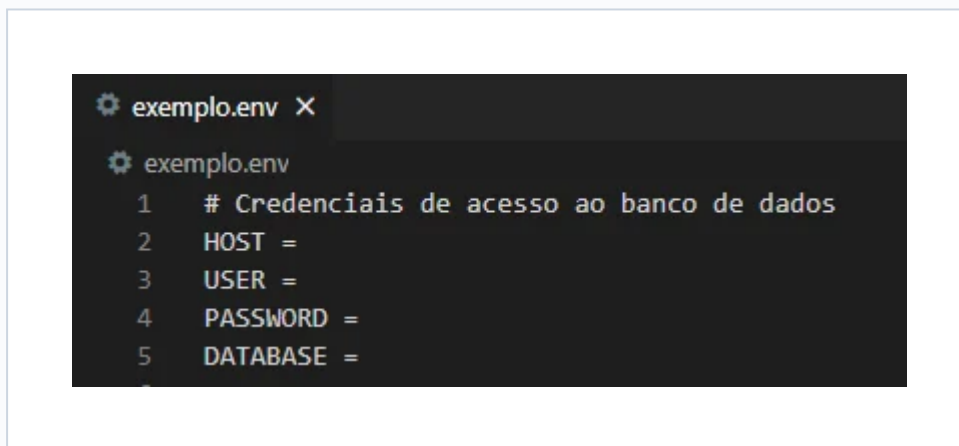
Note que o texto no arquivo se encontra em uma única cor. Caso a quantidade de variáveis de ambiente cresça, o que é comum no desenvolvimento de software, esse padrão monocromático dificulta a leitura e identificação das variáveis e seus valores.

Assim, caso você utilize como IDE o VS Code, saiba que ele fornece algumas extensões para facilitar o uso da ferramenta Dotenv como, por exemplo, a extensão DotENV que serve para trazer cores ao arquivo `.env`, facilitando assim sua leitura. Mas, calma lá! Falaremos com mais detalhes sobre o uso de extensões do VS Code para o Dotenv mais adiante nesse artigo.

É muito importante que você **não envie o arquivo `.env` para o repositório do git** que você estiver utilizando. Em um sistema git, você pode adicionar o `.env` no `.gitignore`. Assim, o arquivo `.env` será automaticamente ignorado nos commits.

Uma prática muito comum é criar um arquivo de exemplo do `.env` que contenha o nome de todas as variáveis de ambiente e enviar esse arquivo para o Github. Dessa forma, outras pessoas que contribuírem com o desenvolvimento do projeto podem facilmente preencher o seu próprio `.env` com as informações do seu próprio ambiente, sem que nenhuma variável seja esquecida.

Nesse arquivo de exemplo também podemos adicionar comentários explicando qual o propósito de cada variável de ambiente. Assim, vamos nomear nosso arquivo de exemplo como `exemplo.env`, conforme mostra a imagem a seguir:



```
exemplo.env X
exemplo.env
1  # Credenciais de acesso ao banco de dados
2  HOST =
3  USER =
4  PASSWORD =
5  DATABASE =
```

Agora que temos nosso arquivo `.env` criado, como podemos utilizar os valores das variáveis salvas nele em nosso projeto?

Falaremos disso na sequência, bora lá!

Como utilizar as variáveis de ambiente

[MATRICULE-SE](#)

O nosso arquivo com as variáveis de ambiente está criado, mas nosso projeto ainda não tem acesso a essas variáveis. Então precisamos modificar o código com o arquivo e utilizar as credenciais guardadas nas variáveis para acessar nosso banco de dados.

Para isso, temos o pacote Dotenv que instalamos. Ele será responsável pela leitura e configuração das variáveis de ambiente.

Como em qualquer outro módulo do Node.js, é necessário importar o módulo para conseguir usar suas funcionalidades. Por isso, a primeira coisa que precisamos fazer é realizar a importação do Dotenv, através de uma chamada do `require` do módulo `dotenv`. Assim, adicionamos o seguinte trecho em nosso código:

```
require('dotenv').config()
```

Agora que importamos o módulo, podemos utilizá-lo dentro da nossa aplicação. A próxima etapa é acessar nossas variáveis de ambiente. Fazemos isso através do `process.env`. O formato usado para acessar as variáveis de ambiente é `process.env.NOME_VARIAVEL`.

O `process` é um recurso que garante que as variáveis de ambiente utilizadas existam somente dentro da aplicação que estamos utilizando, para que não haja conflito caso nosso computador esteja rodando mais de uma aplicação Node.js ao mesmo tempo. Mesmo que essas diferentes aplicações tenham variáveis de ambiente com nomes iguais. Você pode verificar mais detalhes sobre o `process` na [documentação do Node.js](#).

Vamos nos conectar ao banco de dados utilizando as credenciais que estão nas variáveis de ambiente do nosso arquivo `.env`. Para lidar com as questões relacionadas ao banco de dados MySQL em nossa aplicação, vamos utilizar o módulo `mysql2` do npm:

```
require('dotenv').config()
const mysql = require('mysql2')

const connection = mysql.createConnection({
  host: process.env.HOST,
  user: process.env.USER,
```




Desse modo, conseguimos realizar a conexão no banco de dados e estamos prontos para utilizar os dados armazenados no banco em nossa aplicação.

Assim, você agora é capaz de gerenciar suas variáveis de ambiente usando o Dotenv. Isso, sem dúvidas, vai facilitar sua rotina como pessoa desenvolvedora. Mas será que isso é tudo que podemos fazer com o Dotenv?

Existem outras ferramentas que podem melhorar ainda mais o uso do Dotenv. Por exemplo, as extensões do VSCode. Vamos saber mais no próximo tópico?

Dicas VSCode

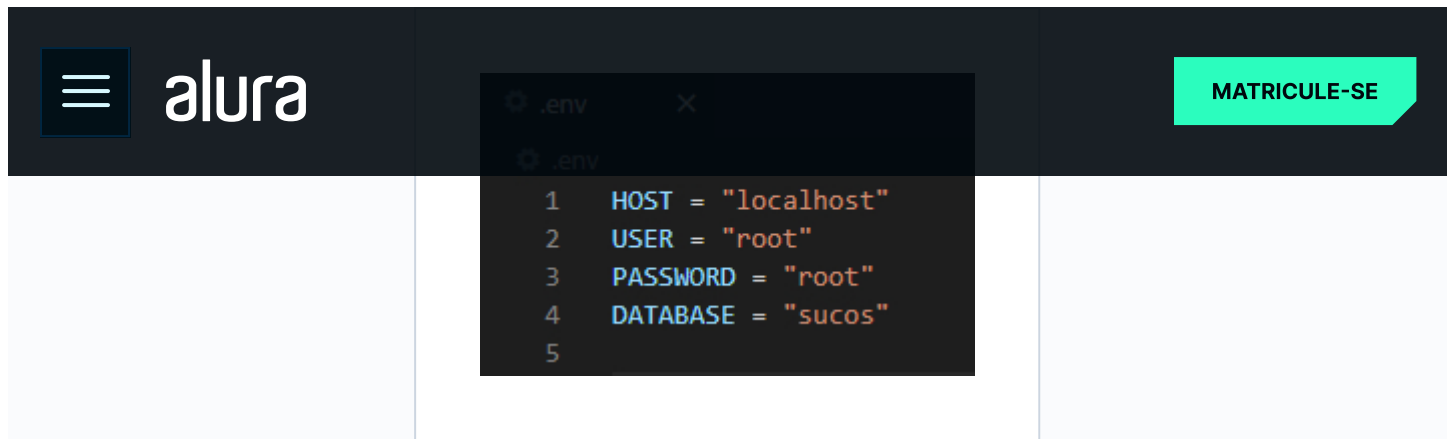
O VSCode possui algumas extensões que podem melhorar o uso do módulo Dotenv.

Lembra que o texto em nosso arquivo `.env` possuía uma única cor, o que dificultava sua leitura? Uma solução para isso é a extensão DotENV. Essa extensão colore o texto do arquivo `.env` e destaca os diferentes elementos da sua estrutura.

Você pode instalar essa extensão por meio do link a seguir:

- [Instalação da extensão DotENV](#)

Após a instalação da extensão, nosso arquivo `.env` ficou com uma aparência melhor. Agora as variáveis e seus valores têm cores diferentes, o que torna mais fácil a leitura e identificação dos elementos do arquivo, como se pode conferir na imagem abaixo:



Imagine que nosso projeto Node.js está crescendo e acabou sendo necessário utilizar vários ambientes, pois agora temos um ambiente para desenvolvimento, um para testes e outro para produção. Dessa forma, as variáveis de ambiente para cada um desses cenários muda e você precisa realizar a alteração do arquivo `.env` sempre que trocar de ambiente. Existe uma extensão do VSCode, chamada `.ENV Switcher`, que pode nos ajudar com isso.

Essa extensão serve para auxiliar o uso de diferentes arquivos de configuração `.env`, para que seja possível navegar entre os ambientes sem precisar alterar os valores em um único arquivo.

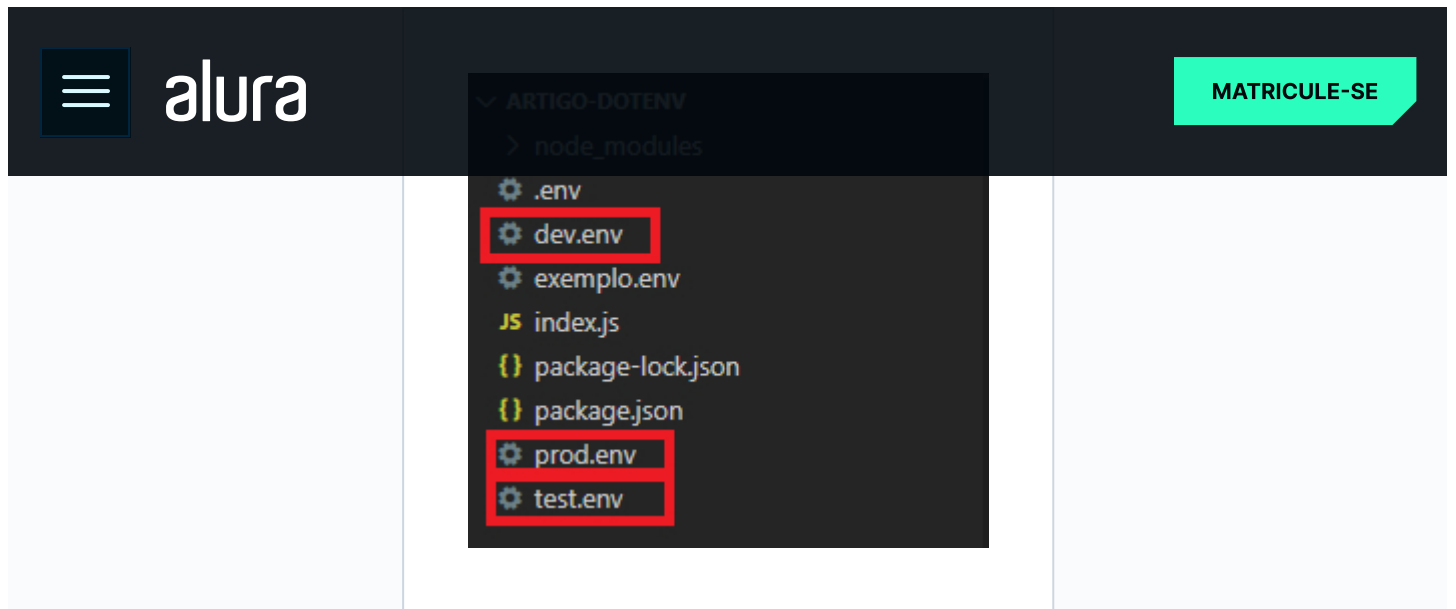
Para instalar essa extensão, use o link a seguir:

- [Instalação .ENV Switcher](#)

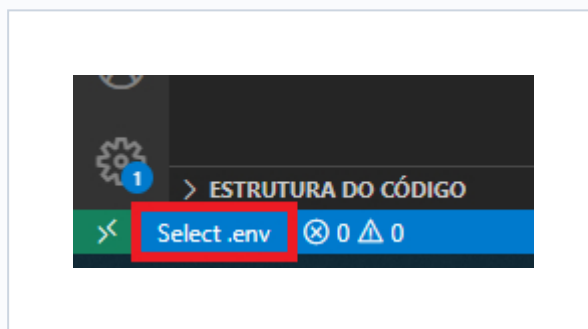
Com essa extensão nós criamos um arquivo `.env` para cada ambiente que estamos utilizando, com as suas respectivas variáveis de ambiente. Assim, teremos três novos arquivos:

- `dev.env` : para o ambiente de desenvolvimento;
- `test.env` : para o ambiente de testes;
- `prod.env` : para o ambiente de produção.

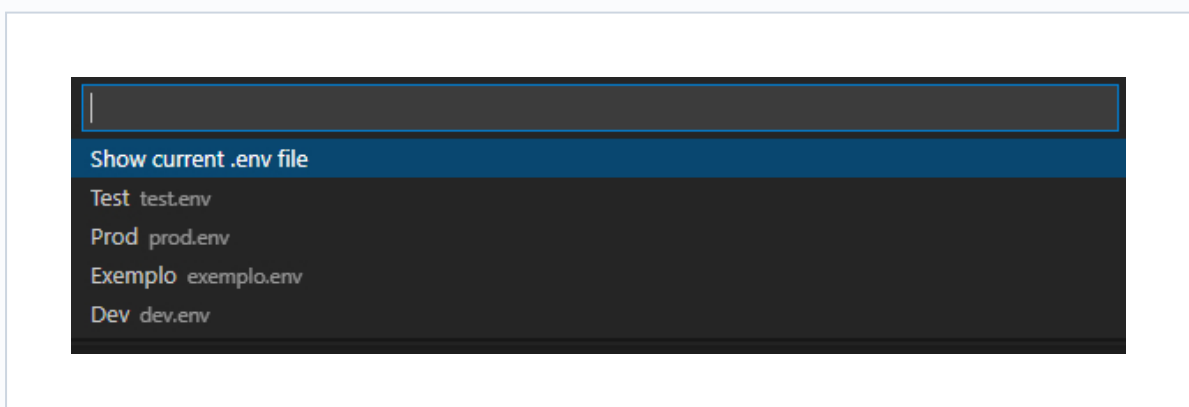
Confira a seguir:



Com a extensão .ENV Switcher instalada, para alterar qual ambiente estamos utilizando, basta clicar na opção escrito *Select .env* que aparecerá no menu inferior esquerdo do VSCode, conforme imagem abaixo:



Na sequência, vamos selecionar qual arquivo `.env` queremos utilizar no menu que será aberto na parte superior do VSCode. Confira:



Assim, podemos facilmente alterar nossas variáveis de ambiente entre os ambientes sem precisar fazer alterações no arquivo `.env`. Lembre-se de adicionar todos esses arquivos no `.gitignore`, pois não queremos colocar no Github os dados privados desses arquivos.



Conclusão

Neste artigo, aprendemos o que é o Dotnet, como instalá-lo e como utilizá-lo para otimizar nossa gestão de variáveis de ambiente em um projeto Node.js. Criamos um arquivo `.env` para guardar as credenciais de acesso a um banco de dados MySQL. E depois acessamos o valor dessas variáveis e ambiente em nossa aplicação. Além disso, também vimos como utilizar extensões do VSCode para tornar mais prático o uso do pacote Dotenv.

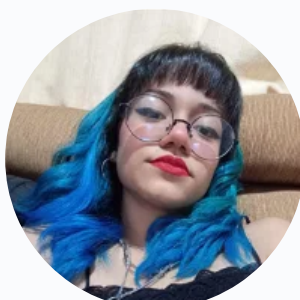
Assim, você agora é capaz de gerenciar suas variáveis de ambiente em seus projetos Node.js de forma mais segura e fácil através do pacote Dotnet.

Referências

- [Dotenv tutorial](#)
- [Tornando sua aplicação Node JS mais segura com Dotenv!](#)

Confira neste artigo:

- [Introdução](#)
- [O que é o Dotenv](#)
- [Instalando o Dotenv](#)
- [Criando um arquivo .env](#)
- [Como utilizar as variáveis de ambiente](#)
- [Dicas VSCode](#)
- [Conclusão](#)
- [Referências](#)



**alura****Gabrielle Ribeiro Gomes****MATRICULE-SE**

Gabrielle é estudante de Engenharia de Software na Universidade de Brasília - UnB. Faz parte do Scuba Team da Alura atuando principalmente com Python. É apaixonada por programação, robótica, machine learning e gatos.

[← Artigo Anterior](#)[Próximo Artigo →](#)[**Cronograma de eventos no Discord \(23-27\) de janeiro**](#)[**Cronograma de eventos no Discord \(16-20\) de janeiro**](#)

Veja outros artigos sobre [Programação](#)

Quer mergulhar em tecnologia e aprendizagem?

Receba a newsletter que o nosso CEO escreve pessoalmente, com insights do mercado de trabalho, ciência e desenvolvimento de software

ME INSCREVA**alura****Nossas redes e apps**

**alura****MATRICULE-SE**

Institucional

[Sobre nós](#)[Trabalhe conosco](#)[Para Empresas](#)[Para Escolas](#)[Política de Privacidade](#)[Compromisso de Integridade](#)[Termos de Uso](#)[Status](#)

A Alura

[Formações](#)[Como Funciona](#)[Todos os cursos](#)[Depoimentos](#)[Instrutores\(as\)](#)[Dev em <T>](#)[Luri by ChatGPT](#)

Conteúdos

[Alura Cases](#)[Imersões](#)[Artigos](#)[Podcasts](#)[Artigos de educação corporativa](#)

Fale Conosco

[Email e telefone](#)[Perguntas frequentes](#)

Novidades e Lançamentos

ENVIAR



CURSOS

Cursos de Programação

Lógica | Python | PHP | Java | .NET | Node JS | C | Computação | Jogos | IoT

Cursos de Front-end

HTML, CSS | React | Angular | JavaScript | jQuery

Cursos de Data Science

Ciência de dados | BI | SQL e Banco de Dados | Excel | Machine Learning | NoSQL | Estatística

Cursos de Inteligência Artificial

IA para Programação | IA para Dados

Cursos de DevOps

AWS | Azure | Docker | Segurança | IaC | Linux

Cursos de UX & Design

Usabilidade e UX | Vídeo e Motion | 3D

Cursos de Mobile

React Native | Flutter | iOS e Swift | Android, Kotlin | Jogos

Cursos de Inovação & Gestão

Métodos Ágeis | Softskills | Liderança e Gestão | Startups | Vendas