DIGITALHOUSE

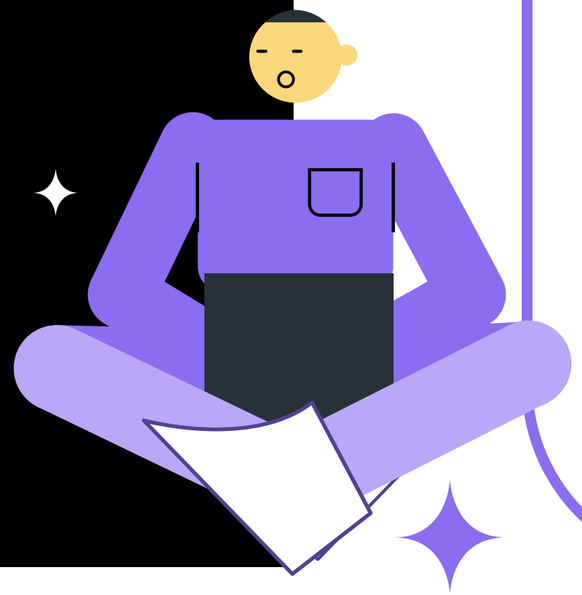
Variables de entorno en Go

Certified Tech Developer
The Ultimate Degree

Índice

01 Godotenv package

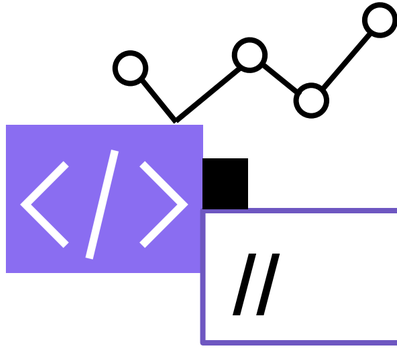
Godotenv es un **package de Go** que sirve **para poder cargar y leer variables** de environment (entorno) **desde un archivo .env**.



Ficheros .env

Para configurar las variables de entorno se utilizan los ficheros **.env**, también llamados coloquialmente como “**dotenv files**” o directamente “**dotfiles**”. De hecho, hoy en día muchas tecnologías soportan este tipo de ficheros: los propios IDEs, Docker y algunos frameworks web son solo unos pocos ejemplos.

En el ecosistema Go también tenemos algunas alternativas para trabajar con este tipo de ficheros. Una de ellas es la librería **godotenv** que se trata de un port de su versión para Ruby.



Instalación

Para instalar el **pkg godotenv**, desde la consola de comandos, ejecutamos la siguiente instrucción:

```
$ go get -u github.com/joho/godotenv
```

Para utilizarlo, en la raíz de nuestro proyecto, se debe crear un archivo con nombre “**.env**”. Por ejemplo:

```
.env
MY_USER=DIGITALHOUSE
MY_PASS=CERTIFIEDTECHDEVELOPER
```

Instalación y uso

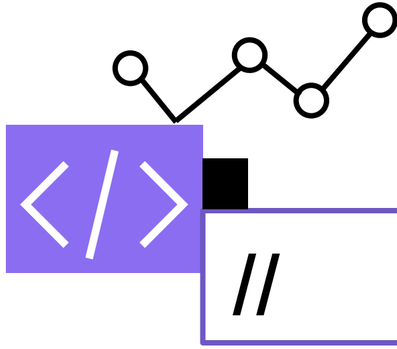
Ya instalado el **godotenv** y creado el archivo **.env**, solo tenemos que importarlo desde nuestra aplicación Go y utilizarlo.

```
{}
```

```go
package main

import (
   "github.com/joho/godotenv"
   "log"
   "os"
)

func main() {
 err := godotenv.Load()
 if err != nil {
   log.Fatal("Error al intentar cargar archivo .env")
 }

 usuario := os.Getenv("MY_USER")
 password := os.Getenv("MY_PASS")
}
```

02

Token en variable de entorno



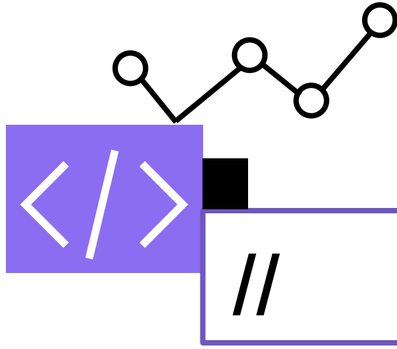
Token en variable de entorno

Implementar **godotenv** al proyecto para poder acceder al archivo **.env** que se utilizará al correr el programa de manera local.

```
$ go get -u github.com/joho/godotenv
```

Crear el archivo **.env** junto al main del proyecto y agregar el token.

```
$ TOKEN=123456
```



Implementar dotenv en main

Se debe implementar la carga del archivo **.env** al inicio de la función **main**. El método **Load** carga el contenido (del archivo **.env**) en la variable de entorno.

```
{}
func main() {

    err := godotenv.Load()
    if err != nil {
        log.Fatal("Error al intentar cargar archivo .env")
    }

}
```

Ejemplo validar token

```
{}
```

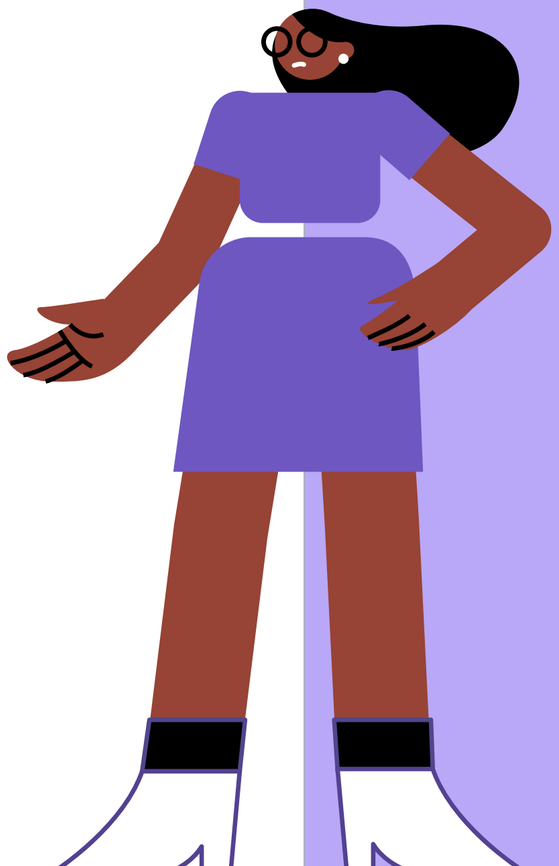
```go
func (c *Product) GetAll() gin.HandlerFunc {
   return func(ctx *gin.Context) {

       token := ctx.GetHeader("token")

       if token != os.Getenv("TOKEN") {
           ctx.JSON(401, gin.H{"error": "token inválido”})
           return
       }

       p, err := c.service.GetAll()
 ...
```

ctx.GetHeader permite tomar variables del header de la petición. No permite tomar variables guardadas en un **.env**.

Conclusiones

En esta presentación terminamos de comprender las variables de entorno.

Además, aprendimos cómo incorporarlas en nuestros programas para poder guardar los tokens y claves de nuestra aplicación.



DIGITALHOUSE

¡Muchas gracias!