GO

BOOTCAMP

Clase en vivo

//Go Bases

IT BOARDING

BOOTCAMP

Objetivos de esta clase

- Conocer el paquete **fmt**
- Conocer el paquete **os**
- Conocer el paquete **io**

Índice

IT BOARDING
BOOTCAMP

1 | Repaso

IT BOARDING

BOOTCAMP

Paquete fmt

Verbos de impresión más utilizados

%v	valor en formato estándar
%T	tipo de dato del valor a imprimir
%t	bool
%s	string
%f	punto flotante
%d	entero decimal
%**b**	un entero binario
%o	octal
%c	imprime caracteres
%p	dirección de memoria

.Print( )

En este ejemplo definimos e inicializamos las variables “**name**” y “**age**”.
Ya que la función Print( ) recibe n número de parámetros y de cualquier tipo, podemos pasarle estas variables, y concatenarlas con cadenas definidas dentro de la función.
“**\n**” es un carácter de escape que indica un salto de línea.

{}

```
name, age := "Kim", 22
fmt.Print(name, " is ", age, " years old.\n")
```

.Sprint( )

La función **Sprint(a ...interface{ }) string** toma como parámetro n variables de cualquier tipo y las formatea como un string de manera estándar según su tipo.
El valor de retorno es la concatenación de estas variables condensadas en un string.

En este ejemplo usamos la función **Sprint( )** para guardar en una variable la concatenación de todas las variables pasadas por parámetro y luego usamos la función **Print( )** para imprimir la variable generada.

```
{}
name, age := "Kim", 22
res := fmt.Sprint(name, " is ", age, " years old.\n")
fmt.Print(res)
```

Paquete os

.ReadFile( )

La función **ReadFile(filename string)** recibe como parámetro la dirección y nombre del archivo en formato texto y nos devuelve el contenido del archivo en bytes o un error en caso que lo haya.

```
{}  data, err := os.ReadFile("./myFile.txt")
```

.Exit( )

La función **Exit(code int)** hace que el programa termine inmediatamente con el código de estado dado. Convencionalmente, el código cero indica éxito, el no cero es un **error**. Las funciones diferidas no se ejecutan.

```
{}    os.Exit(1)
```

Paquete io

ReadAll

ReadAll(r Reader) lee desde r hasta un error o EOF y devuelve los datos que leyó y un error si lo hubiera.

```
{}
r := strings.NewReader("some io.Reader stream to be read")

b, err := io.ReadAll(r);
```

Copy

Copy(dst Writer, src Reader) tiene un comportamiento similar a **ReadAll** con la particularidad de que ya incluye el destino(**dst**).

```
{}
r := strings.NewReader("some io.Reader stream to be read")

_, err := io.Copy(os.Stdout, r)
```

WriteString

La función **WriteString(w Writer, s string)** que toma una cadena de texto y un Writer, escribe el contenido de s a w, que acepta una slice de bytes.

```
{}  io.WriteString(os.Stdout, "Hello world!")
```

2 | Puesta en común

IT BOARDING

BOOTCAMP

Puesta en común

Conclusiones

En esta clase trabajamos sobre el paquete **fmt**, que nos permite formatear y mostrar datos en las entradas y salidas.

Trabajamos sobre el paquete **os**, permitiéndonos interactuar con el sistema operativo.

Por último aplicamos el paquete **io**, para realizar operaciones genéricas de entrada y salida

Gracias.

IT BOARDING

BOOTCAMP