codecademy

Learn Go: fmt Package

Go Fmt .Print() and .Println()

The Go `fmt` package supports two closely-related functions for formatting a string to be displayed on the terminal. `.Print()` accepts strings as arguments and concatenates them without any spacing. `.Println()`, on the other hand, adds a space between strings and appends a new line to the concatenated output string.

```
fmt.Print("I", "am", "cool")
// Iamcool
fmt.Println("I", "am",
"cool")
// I am cool
```

Go Fmt .Printf() Function

The Go `.Printf()` function in `fmt` provides custom formatting of a string using one or more verbs. A verb is a placeholder for a named value (constant or variable) to be formatted according to these conventions:

- `%v` represents the named value in its default format
- `%d` expects the named value to be an integer type
- `%f` expects the named value to be a float type
- `%T` represents the type for the named value

The first argument for `.Printf()` is the string with verb(s) followed by one or more named values corresponding to the verb(s). Unlike `.Println()`, `.Printf()` does not append a newline to the formatted string.

```
name := "Leslie"
fmt.Printf("My name is %v", name)
// My name is Leslie

age := 34
fmt.Printf("I am %d years old", age)
// I am 34 years old

fmt.Printf("%v is of type %T", name,
name)
// Leslie is of type string
```

Go Fmt .Scan() Function

In Go, fmt's Scan() method allows users to input information. The function accepts an argument of an address to be scanned into.

```
var number int
fmt.Println("What is your favorite number?")
fmt.Scan(&number)
fmt.Printf("Your favorite number is %d.\n", number)
```

Print Share