06

Exibindo o menu

Transcrição

Agora, após as mensagens de boas vindas e de versão do programa, queremos dar as opções para o usuário, se ele quer monitorar os sites, exibir os logs, etc, e vamos mostrar tudo isso em um menu!

Disponibilizaremos o menu imprimindo as suas opções para o usuário, ele irá digitar a opção desejada, e nós capturaremos o que foi digitado. Então, primeiramente devemos imprimir as opções:

```
package main

import "fmt"

func main() {
    nome := "Douglas"
    idade := 24
    versao := 1.1
    fmt.Println("Olá sr.", nome, "sua idade é", idade)
    fmt.Println("Este programa está na versão", versao)

    fmt.Println("1- Iniciar Monitoramento")
    fmt.Println("2- Exibir Logs")
    fmt.Println("0- Sair do Programa")
}
```

Capturando a entrada do usuário

Para capturar o que o usuário escrever, existe a função `Scanf`, também do pacote `fmt`. Ela recebe dois argumentos, um modificador (o que queremos receber como entrada, um inteiro, string, etc) e um ponteiro para a variável que guardará a entrada do usuário.

Então, vamos declarar essa variável:

```
package main

import "fmt"

func main() {
    nome := "Douglas"
    idade := 24
    versao := 1.1
    fmt.Println("Olá sr.", nome, "sua idade é", idade)
    fmt.Println("Este programa está na versão", versao)

    fmt.Println("1- Iniciar Monitoramento")
    fmt.Println("2- Exibir Logs")
    fmt.Println("0- Sair do Programa")
```

```
    var comando int
}
```

E agora nós passamos para a função `Scanf` o valor `"%d"` , que significa que esperamos receber um número inteiro, e a variável `comando` :

```
package main

import "fmt"

func main() {
    nome := "Douglas"
    idade := 24
    versao := 1.1
    fmt.Println("Olá sr.", nome, "sua idade é", idade)
    fmt.Println("Este programa está na versão", versao)

    fmt.Println("1- Iniciar Monitoramento")
    fmt.Println("2- Exibir Logs")
    fmt.Println("0- Sair do Programa")

    var comando int
    fmt.Scanf("%d", &comando)
}
```

Daqui a pouco falaremos sobre o `&` , que se encontra à frente da variável `comando` . Ou seja, a função salvará na variável o que o usuário digitar. E para verificar se a entrada está mesmo sendo salva na variável `comando` , vamos imprimi-la:

```
package main

import "fmt"

func main() {
    nome := "Douglas"
    idade := 24
    versao := 1.1
    fmt.Println("Olá sr.", nome, "sua idade é", idade)
    fmt.Println("Este programa está na versão", versao)

    fmt.Println("1- Iniciar Monitoramento")
    fmt.Println("2- Exibir Logs")
    fmt.Println("0- Sair do Programa")

    var comando int
    fmt.Scanf("%d", &comando)

    fmt.Println("O comando escolhido foi", comando)
}
```

Ao executar o programa, ele fica travado esperando que digitemos algo. Ao digitar, temos a seguinte saída:

```
alura@alura-01:~/go/src/hello$ go run hello.go
Olá sr. Douglas sua idade é 24
```

```
Este programa está na versão 1.1
1- Iniciar Monitoramento
2- Exibir Logs
0- Sair do Programa
2
O comando escolhido foi 2
```

Entendendo o ponteiro

Sobre o `&` visto antes, ele significa o **endereço da variável** que queremos salvar a entrada, pois a função `Scanf` não espera uma variável, e sim o seu endereço, um ponteiro para a variável.

A variável nada mais é do que uma "caixa", onde guardamos dados. Essa "caixa" está em algum lugar da memória do nosso computador, e esse lugar, o endereço da nossa "caixa", é o que chamamos de **ponteiro**.

Para descobrir o endereço da variável, basta colocar o `&` à frente dela.

Devemos especificar o que esperamos ser digitado?

A variável `comando` é do tipo inteiro, logo, só pode receber números inteiros. Se sabemos disso, por que devemos especificar que esperamos receber um número inteiro na função `Scanf`, através do modificador `"%d"`? Na verdade, nós não precisamos.

Alguém do Go também pensou nisso, por isso criou a função `Scan` (sem a letra `f`). Nela, nós não precisamos especificar o modificador que esperamos:

```
package main

import "fmt"

func main() {
    nome := "Douglas"
    idade := 24
    versao := 1.1
    fmt.Println("Olá sr.", nome, "sua idade é", idade)
    fmt.Println("Este programa está na versão", versao)

    fmt.Println("1- Iniciar Monitoramento")
    fmt.Println("2- Exibir Logs")
    fmt.Println("0- Sair do Programa")

    var comando int
    fmt.Scan(&comando)

    fmt.Println("O comando escolhido foi", comando)
}
```

Ou seja, estamos esperando um dado, e quando o recebermos, colocamos dentro de uma variável inteira. Como a variável é inteira, a função sabe que deve esperar, receber valores inteiros.

Ao executar o programa, caso digitemos um valor não-inteiro, como uma letra, por exemplo, não será guardado nenhum valor no endereço da variável `comando`, logo ela continuará com o valor padrão atribuído a ela no momento da sua

inicialização, que é 0. Ou seja, o ideal seria que detectássemos quando um valor não-inteiro fosse digitado.

É isso que veremos no próximo capítulo.