

## Mãos na Massa: Organização e Fluxo

Começando deste ponto? Você pode fazer o [DOWNLOAD \(https://s3.amazonaws.com/caelum-online-public/624-golang/02/projetos/alura-golang-stage-fim-cap02.zip\)](https://s3.amazonaws.com/caelum-online-public/624-golang/02/projetos/alura-golang-stage-fim-cap02.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo.

Neste capítulo vamos determinar para qual fluxo do nosso programa o usuário deve seguir a partir do comando que ele escolheu, e além disso vamos começar a organizar nosso código em pequenas funções.

1- O primeiro passo vai ser criar um `switch` com todas as opções que oferecemos ao usuário até agora, e um caso de `default` para quando o usuário colocar algum comando desconhecido:

```
//hello.go

package main

import "fmt"

func main() {
    // ... restante da função main
    var comando int
    fmt.Scan(&comando)

    fmt.Println("O valor da variável comando é:", comando)

    switch comando {
    case 1:
        fmt.Println("Monitorando...")
    case 2:
        fmt.Println("Exibindo Logs...")
    case 0:
        fmt.Println("Saindo do programa...")
    default:
        fmt.Println("Não conheço este comando")
    }
}
```

*Aqui colocamos uma opção no switch para cada item do menu que tínhamos no menu anteriormente*

2- Agora vamos começar a organizar nosso código em pequenas funções, cada uma com sua responsabilidade. A primeira a ser criada será a função que vai exibir a introdução de boas vindas para o usuário. Extraí este código da função `main` e coloque na função `exibeIntroducao()`:

```
//hello.go

func main() {

    exibeIntroducao()

    fmt.Println("1- Iniciar Monitoramento")
    fmt.Println("2- Exibir Logs")
}
```

```
    fmt.Println("0- Sair do Programa")

    // ... restante da função main
}
func exibeIntroducao(){
    nome := "Douglas"
    versao := 1.1
    fmt.Println("Olá, sr(a).", nome)
    fmt.Println("Este programa está na versão", versao)
}
```

*Não esqueça de chamar a função `exibeIntroducao()` na função `main`.*

3- Vamos extrair também o código que exibe o menu de opções do usuário para uma função externa, chamada de `exibeMenu` :

```
func main() {

    exibeIntroducao()
    exibeMenu()

    var comando int
    fmt.Scan(&comando)
    fmt.Println("O valor da variável comando é:", comando)
    //... restante da função

}

func exibeIntroducao() {
    //... restante da função
}

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

*Não esqueça de chamar a função `exibeMenu()` na função `main`.*

4- Por último, vamos criar uma função responsável por ler os comandos do usuário e exportar esta função da `main` também. A função `leComando` deve retornar um `int` com o comando lido pelo usuário:

```
func main() {

    exibeIntroducao()
    exibeMenu()
    comando := leComando()

    switch comando {
    //... restante da função
    }

}
```

```
func exibeIntroducao() {  
    //... restante da função  
}  
  
func exibeMenu(){  
    //... restante da função  
}  
  
func leComando() int {  
    var comandolido int  
    fmt.Scan(&comandolido)  
    fmt.Println("O valor da variável comando é:", comandolido)  
  
    return comandolido;  
}
```

*Não esqueça de capturar o comando lido na função main, para que ele possa ser utilizado pelo switch*

5- Vamos por último adicionar nos itens finais do switch um ponto de saída para o nosso script, retornando para o sistema operacional se tudo correu bem ou não. Importe o pacote `os` do Go e utilize a função `Exit()` para informar o status de saída do programa:

```
package main  
  
import "fmt"  
import "os"  
  
func main() {  
  
    exibeIntroducao()  
    exibeMenu()  
    comando := leComando()  
  
    switch comando {  
    case 1:  
        fmt.Println("Monitorando...")  
    case 2:  
        fmt.Println("Exibindo Logs...")  
    case 0:  
        fmt.Println("Saindo do programa...")  
        os.Exit(0)  
    default:  
        fmt.Println("Não conheço este comando")  
        os.Exit(-1)  
    }  
}  
  
//... restante do arquivo
```

Pronto, nosso código está mais organizado e agora só falta implementar as funções de cada um dos casos do nosso switch.