

Mãos na Massa: Guardando os Logs

Começando deste ponto? Você pode fazer o [DOWNLOAD \(https://s3.amazonaws.com/caelum-online-public/624-golang/06/projetos/alura-golang-stage-fim-cap06.zip\)](https://s3.amazonaws.com/caelum-online-public/624-golang/06/projetos/alura-golang-stage-fim-cap06.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo.

Neste exercício vamos finalizar nosso projeto e escrever a funcionalidade final, que escreve nosso arquivo de logs.

1- Para isto vamos criar a função `registraLog` que deve ser responsável por escrever em um arquivo de textos se o site está online ou não. Crie a função `registraLog`, que recebe o site e um booleano para informar se o site está online ou não.

```
// restante do código omitido

func registraLog(site string, status bool) {

}
```

2- Vamos chamar a função `registraLog` toda vez que um site retornar o status code 200 ou não. Em sua função `testaSite`, adicione duas chamadas a `registraLog`, passando true ou false em caso de sucesso ou não da requisição de teste:

```
// restante do código omitido

func testaSite(site string) {

    //restante da função.

    if resp.StatusCode == 200 {
        fmt.Println("Site:", site, "foi carregado com sucesso!")
        registraLog(site, true)
    } else {
        fmt.Println("Site:", site, "está com problemas. Status Code:", resp.StatusCode)
        registraLog(site, false)
    }
}
```

3- Agora dentro da função `registraLog`, vamos chamar a função `os.OpenFile` com as flags `os.O_CREATE|os.O_RDWR|os.O_APPEND`, para que o arquivo possa ser criado, possa ser escrito e que possamos adicionar múltiplas linhas nele. Vamos detectar se ocorrer algum erro e também não devemos esquecer de fechar o arquivo.

```
//hello.go
// restante do código omitido

func registraLog(site string, status bool) {

    arquivo, err := os.OpenFile("log.txt", os.O_CREATE|os.O_RDWR|os.O_APPEND, 0666)

    if err != nil {
        fmt.Println("Ocorreu um erro:", err)
    }
}
```

```

    }

    arquivo.Close()
}

```

4- Agora como visto no vídeo, vamos utilizar a função `.writeString` de arquivo, junto com a `.Format` do pacote `time` para escrever no arquivo de texto a hora que o log foi registrado e se o site estava online ou não. Como a função `.WriteString` aceita apenas strings, vamos tomar o cuidado de converter o booleano `status` para string também, com a função `strconv.FormatBool`

```

//hello.go
// restante do código omitido

func registraLog(site string, status bool) {

    arquivo, err := os.OpenFile("log.txt", os.O_CREATE|os.O_RDWR|os.O_APPEND, 0666)

    if err != nil {
        fmt.Println("Ocorreu um erro:", err)
    }
    arquivo.WriteString(time.Now().Format("02/01/2006 15:04:05") + " - " + site +
        " - online: " + strconv.FormatBool(status) + "\n")

    arquivo.Close()
}

```

5- Com a escrita dos logs pronta, basta agora criar uma pequena função com os conhecimentos de leitura de arquivos que já temos para criar a função `imprimeLogs`, que vai ler o nosso arquivo de logs e imprimir no nosso terminal:

Não esqueça de importar `io/ioutil`

```

import "io/ioutil"

// restante do código omitido

func imprimeLogs() {

    arquivo, err := ioutil.ReadFile("log.txt")

    if err != nil {
        fmt.Println("Ocorreu um erro:", err)
    }

    fmt.Println(string(arquivo))
}

```

Lembre-se a função `ReadFile` do pacote `ioutil` já realiza o `Open` e `Close` do arquivo por debaixo dos panos para nós, por isso não nos preocupamos com isto aqui`

6- Com a função criada, vamos chama-lá em nosso menu:

```
// restante do código omitido

func main() {
    exibeIntroducao()
    for {
        exibeMenu()
        comando := leComando()

        switch comando {
        case 1:
            iniciarMonitoramento()
        case 2:
            //Chamando aqui
            fmt.Println("Exibindo Logs...")
            imprimeLogs()
        case 0:
            fmt.Println("Saindo do programa")
            os.Exit(0)
        default:
            fmt.Println("Não conheço este comando")
            os.Exit(-1)
        }
    }
}
```

Agora nossa aplicação está pronta, implementamos todas as funções que gostaríamos e agora conseguimos monitorar nossos sites utilizando a linguagem Go.