Introdução ao Go

Aula 12: Conversões de Tipo & Atribuição Explícita de Tipo

Nessa aula, o professor abordou o conceito de **conversão de tipo** e como **atribuir tipos explícitos** a variáveis em Go para evitar conversões desnecessárias, o que melhora a clareza e a legibilidade do código.

Problema com Tipos Inferidos

Em Go, quando você cria uma variável e atribui um valor a ela, o compilador tenta **inferir** automaticamente o tipo da variável com base no valor atribuído. Por exemplo:

```go
var investmentAmount = 1000
```

Neste caso, o Go infere que `investmentAmount` é do tipo `int` porque o valor atribuído, `1000`, é um número inteiro.

No entanto, esse processo de inferência pode causar problemas se você quiser realizar operações com valores de tipos diferentes, como `int` e `float64`. Em alguns casos, você precisará **converter explicitamente** os tipos de dados, o que pode deixar o código mais complexo e difícil de ler.

Atribuição Explícita de Tipo

Uma forma de evitar a necessidade de conversões de tipo em cálculos, como no caso do valor do investimento e do número de anos, é **atribuir explicitamente o tipo** à variável.

Por exemplo, se sabemos que o valor do investimento e o número de anos precisam ser tratados como `float64` (números com ponto flutuante), podemos fazer a atribuição explícita logo ao criar a variável:

```go

```

```
var investmentAmount float64 = 1000
var years float64 = 10
```

Dessa forma, essas variáveis são tratadas internamente como `float64`, evitando a necessidade de conversões de tipo durante o cálculo. O código se torna mais limpo e legível:

go

```
futureValue := investmentAmount * math.Pow(1+expectedReturnRate/100, years)
```

Agora, não há mais necessidade de fazer as conversões de `int` para `float64`, o que simplifica o código e melhora sua leitura.

Vantagens da Atribuição Explícita

- **Redução de complexidade**: Ao definir explicitamente os tipos, você evita conversões de tipo durante o cálculo, o que simplifica o código.
- **Clareza**: O código fica mais claro, pois você deixa explícito que essas variáveis devem ser tratadas como números de ponto flutuante (`float64`), independentemente de seu valor inicial.

Quando Usar Conversão de Tipo

Em alguns casos, você pode ter variáveis que devem ser usadas como `int` em algumas operações e como `float64` em outras. Nesses casos, você ainda precisará realizar **conversões de tipo explícitas**. Mas, quando possível, **definir explicitamente o tipo da variável** pode evitar essa necessidade.

Exemplo de conversão de tipo:

go

```
var intValue int = 5
var floatValue float64 = float64(intValue) // Conversão explícita
```

Resumo da Aula

- **Inferência de Tipo**: Go tenta automaticamente inferir o tipo de uma variável com base no valor atribuído, mas isso nem sempre é o desejado.
- **Atribuição Explícita de Tipo**: Para evitar conversões desnecessárias, você pode atribuir explicitamente o tipo de uma variável.
- **Conversões de Tipo**: Em alguns casos, quando os tipos de variáveis precisam ser usados de maneira diferente, você pode precisar converter explicitamente entre tipos, como de `int` para `float64`.

Se precisar de mais ajuda ou explicações, estou à disposição!