

Go – Do básico ao desafio técnico



Conteúdo Programático

1. Introdução ao Golang
2. Fundamentos da Linguagem
3. Controle de Fluxo
4. Funções e Métodos
5. Estruturas de Dados
6. Manipulação de Erros
7. Pacotes e Modularização
8. Boas Práticas e Idiomas
9. Trabalho com APIs e Bibliotecas Externas
10. Projeto Teste
11. Conclusão e Próximos Passos



Introdução

O que é Golang?

História e Filosofia

Instalação e Configuração do Ambiente

Configuração de IDEs e Ferramentas (vscode)



Leonardo Neves

Desenvolvedor de Software - Viceri, aluno
de Sistemas de Informação

Stack profissional atual: .NET/Angular



João Esperandio

Engenheiro de Software Sênior - Visa Pismo

Stack profissional atual: Go



O que é Golang?

- **Go** é uma linguagem de programação de código aberto.
- **Sintaxe simples e clara:** Projetada para ser de fácil leitura e escrita, facilitando o desenvolvimento.
- **Compilação nativa:** Ao contrário de linguagens como Java, Go é compilada diretamente em código de máquina, similar a C e C++.
- **Tipagem estática:** Possui tipagem estática, com inferência de tipos que torna o código mais conciso e seguro.
- **Escalabilidade:** Ideal para construir sistemas de grande escala, com alto desempenho.
- **Foco em sistemas e servidores:** Embora seja uma linguagem de propósito geral, Go é otimizada para programação de sistemas, servidores e serviços na nuvem.
- **Suporte a concorrência:** Oferece excelente suporte para o desenvolvimento de programas concorrentes e distribuídos.
- **Garbage collector eficiente:** Gerenciamento automático de memória, simplificando o desenvolvimento e evitando vazamentos de memória.
- **Biblioteca padrão robusta:** Vem com uma rica coleção de bibliotecas padrão, prontas para uso em várias aplicações.
- **Multiplataforma:** O compilador Go está disponível em várias plataformas, incluindo Linux, macOS, Windows, BSD e Unix.



História e Filosofia

Go é uma linguagem de programação criada pelo Google em 2007. Ela foi projetada para ser eficiente, fácil de usar e produtiva, ideal para sistemas grandes e com muitos processos simultâneos.

A necessidade de uma linguagem mais simples que o C++, com compilação e execução eficientes, em um contexto de processadores com muitos núcleos e maior segurança de memória, motivou sua criação.

O projeto, iniciado por Robert Griesemer, Rob Pike e Ken Thompson, tornou-se open source em 2009 e, após três anos de aprimoramentos, a versão 1.0 foi lançada em 2012, supervisionado por Russ Cox.

[Documentário completo](#)



História e Filosofia

Provérbios do Go: Simples e Impactantes


- Clareza é melhor que esperteza.
- Quanto maior a interface, mais fraca é a abstração.
- `interface{}` não diz nada.
- O valor zero deve ser útil.
- Erros são valores.
- Concorrência não é o mesmo que paralelismo.
- Não entre em pânico.



Em 2022, a NSA (National Security Agency, Estados Unidos) recomenda o desenvolvimento de software em linguagens *memory-safe*

NSA urges orgs to use memory-safe programming languages

C/C++ on the bench, as US snoop HQ puts its trust in Rust, C#, Go, Java, Ruby, Swift

 [Laura Dobberstein](#)

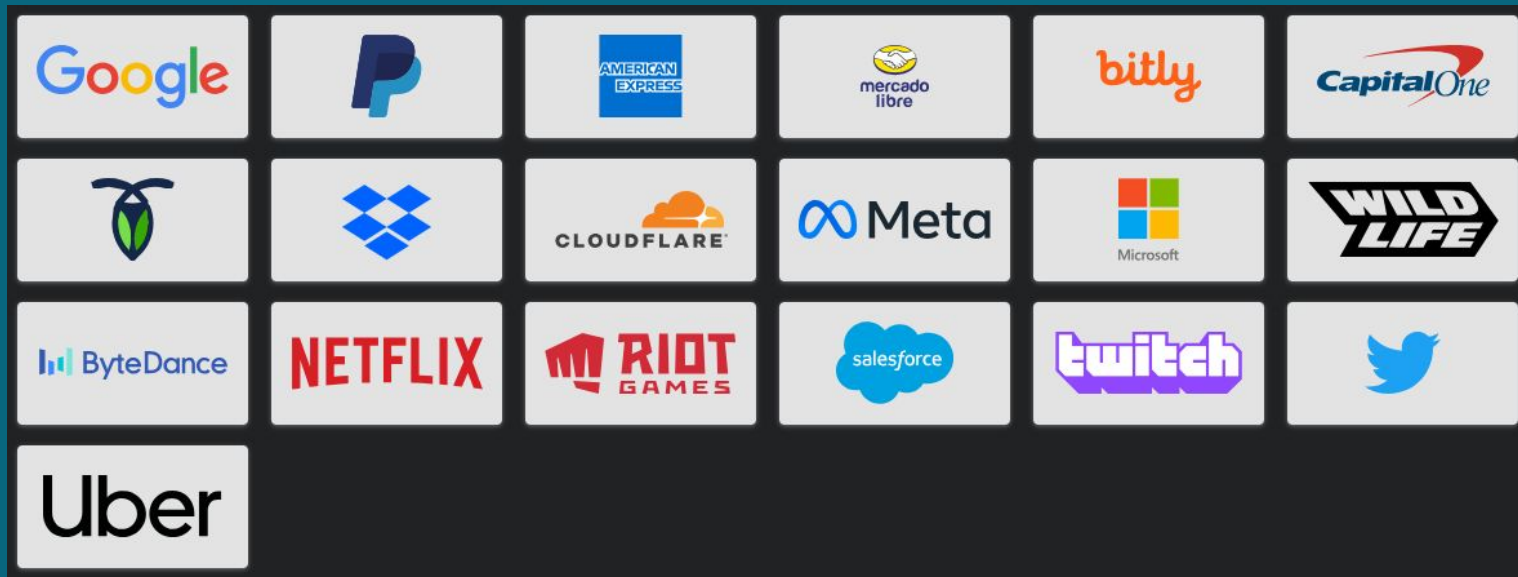
Fri 11 Nov 2022 // 11:35 UTC

<https://www.nsa.gov/Press-Room/News-Highlights/Article/Article/3215760/nsa-releases-guidance-on-how-to-protect-against-software-memory-safety-issues/>



Empresas que usam Go

<https://go.dev/>

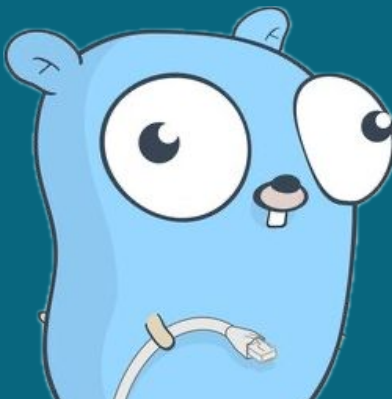


Feito pra ser confortável!

Código em C++

```
1 pilha.cpp
2 #include <stdio.h>
3
4 struct Pilha {
5     int info;
6     Pilha *prox;
7 };
8
9 Pilha *criaPilha(int info) {
10     Pilha *pilha = new Pilha;
11     pilha->info = info;
12     return pilha;
13 }
14
15 void empilha(Pilha pilha, int info) {
16     Pilha *novoNo = criaPilha(info);
17     novoNo->prox = &pilha;
18     pilha = *novoNo;
19 }
20
21 int desempilha(Pilha &pilha) {
22     int info = pilha.info;
23     pilha = *pilha.prox;
24     return info;
25 }
26
27 char isEmpty(Pilha *pilha) {
28     return pilha == NULL;
29 }
30
```

"Vocês vão me adorar!"



Código em Go

```
1 pilha.go
2 package main
3
4 type Pilha struct {
5     info int
6     prox *Pilha
7 }
8
9 func criaPilha(topo int) *Pilha {
10     pilha := new(Pilha)
11     pilha.info = topo
12     return pilha
13 }
14
15 func empilha(topo int, p **Pilha) {
16     novoNo := criaPilha(topo)
17     novoNo.prox = *p
18     *p = novoNo
19 }
20
21 func desempilha(p **Pilha) int {
22     arvore := (*p).info
23     *p = (*p).prox
24     return arvore
25 }
26
27 func isEmpty(p *Pilha) bool {
28     return p == nil
29 }
30
```

Instalação e Configuração do Ambiente

<https://go.dev/doc/install>

