

GO CHEAT SHEET

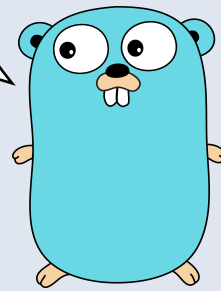


WHAT IS GO?

The Go programming language, also referred to as [Golang](#), was released in 2009. Go is an open source programming language that makes it easy to build simple, reliable, and efficient software.

Go is derived from the C programming language. More so, it offers additional capabilities such as garbage collection, native-style concurrency, and several other native features that allow developers to avoid writing lengthy code to handle networked applications or memory leaks.

Did you know?
JFrog Xray and
JFrog CLI are
written in GO!



HOW TO INSTALL GO IN 3 STEPS

1. Extract the archive you downloaded into /usr/local, which will create a Go tree in /usr/local/go.
Important: Back up any data before extracting the download. This step will remove any previous installation that may be in /usr/local/go.

For example, run the following command as root or through sudo:

```
$ rm -rf /usr/local/go && tar -C /usr/local -xzf go1.16.4.linux-amd64.tar.gz
```

2. Add /usr/local/go/bin to the PATH environment variable.
You can do this by adding the following line to your \$HOME/.profile or /etc/profile (for a system-wide installation):

```
$ export PATH=$PATH:/usr/local/go/bin
```

Note: Changes made to a profile file may not apply until the next time you log into your computer. To apply the changes immediately, just run the shell commands directly or execute them from the profile using a command such as:
source \$HOME/.profile.

3. Verify that you've installed Go by opening a command prompt and typing the following command:

```
$ go version
```

The command will print the installed Go version of.

Learn more about different [installations types](#)

GO PACKAGE MANAGER

The initial prototype vgo was [announced](#) in February 2018. Go included support for [versioned modules](#) since version 1.11. In July 2018, versioned modules [landed](#) in the main Go repository.

Today, Go modules is a collection of Go packages stored in a file tree under the \$GOPATH/pkg folder, including a go.mod file at its root. This file defines the module's path, which is also the import path used for your project, and its dependency requirements, which are the other modules needed for a successful build.

SIMPLE HELLO WORLD EXAMPLE

Step 1: Create a new file called hello.go, with the following:

```
package main

import "fmt"
func main() {
    message := greetMe("world")
    fmt.Println(message)
}
func greetMe(name string) string {
    return "Hello, " + name + "!"
}
```

Step 2: Compile it by running the following command:

```
$ go build
```

Get started with [A Tour of Go](#)

CODING IN GO

Variables

```
Variable declaration
var msg string
msg = "Hello"
```

Constants

```
const Phi = 1.618
```

Strings

```
str := "Frog"
```

Numbers

```
num := 3 // int

num := 3. // float64
num := 3 + 4i // complex128
num := byte('a') // byte (alias for uint8)
```

Arrays

```
// var numbers [3]int
numbers := [...]int{0, 0, 0}
```

Pointers

```
func main () {
    b := *getPointer()
    fmt.Println("Value is", b)
```

GO & JFROG

JFrog offers a unique solution for building, storing and [managing your Go packages](#) in secure, private Go registries. As Go continues to evolve, so will JFrog, maintaining a high level of support for the Go development community.

[Read more about Go repositories](#)

