# Pemrograman Go

**Dr. Bambang Purnomosidi D. P. <bambangpdp@gmail.com>**

**Version 0.0.1-rev-2021-03-17 15:33:21 +0700**

# Daftar Isi

# Preface


Buku ini merupakan buku bebas tentang bahasa pemrograman Go. Go merupakah bahasa pemrograman yang dirancang dan pertama kali diimplementasikan oleh Robert Griesemer, Rob Pike, dan Ken Thompson di Google. Go banyak digunakan pada sistem terdistribusi dan sistem berbasis Cloud. Buku ini dibuat dengan sponsor dari Zimera Systems dan mempunyai lisensi **Creative Commons Attribution-ShareAlike 4.0 International**.

- Lisensi dalam Bahasa Indonesia - https://creativecommons.org/licenses/by-sa/4.0/deed.id.
- Lisensi dalam Bahasa Inggris - https://creativecommons.org/licenses/by-sa/4.0/deed.en.

Secara umum, penggunaan lisensi ini mempunyai implikasi bahwa pengguna materi:

1. Harus memberikan atribusi ke penulis dan sponsor untuk penulisan materi ini.
2. Boleh menggunakan produk yang ada disini untuk keperluan apapun jika point 1 di atas terpenuhi.
3. Boleh membuat produk derivatif dari produk yang ada disini sepanjang perubahan-perubahan yang dilakukan diberitahukan ke kami dan di-share dengan menggunakan lisensi yang sama.

## Atribusi

**Dr. Bambang Purnomosidi D. P.**
***Zimera Systems***
Dusun Medelan, Umbulmartani, Ngemplak
Sleman, DIY
https://www.google.com/maps/place/Zimera+Systems/@-7.6975303,110.43921,17z/data=!3m1!4b1!4m5!3m4!1s0x2e7a5d7cc40e8871:0x2d44da15f0b37

81e!8m2!3d-7.6975303!4d110.4413987
E-mail: zimera-systems@gmail.com

Pembuatan buku ini merupakan hasil pekerjaan kolektif baik secara langsung maupun tidak langsung. Penulis serta kontributor mengucapkan terima kasih untuk Zimera Systems yang telah memberikan **sponsorship** selama penulisan buku ini.

# Bagian I: Go dan Peranti Pengembangan Go

Bagian ini membahas tentang pengenalan Go secara umum dan bagaimana menyiapkan peranti pengembangan jika ingin membuat program meggunakan Go.

# Bab 1. Pengenalan Go

## 1.1. Apa itu Go?

Go adalah nama bahasa pemrograman sekaligus nama implementasi dalam bentuk kompilator (`compiler`). Untuk pembahasan berikutnya, istilah `Go` akan mengacu juga pada spesifikasi bahasa pemrograman serta peranti pengembangannya. Nama yang benar adalah `Go`, bukan `Golang` atau `golang`. Istilah `Golang` atau `golang` muncul karena nama domain `go.org` tidak tersedia saat itu dan mencari sesuatu melalui Google atau mesin pencari lainnya menggunakan kata kunci `Go` tidak menghasilkan hasil yang baik. Dengan demikian, untuk penyebutan di **hashtag** biasanya digunakan `#golang` sehingga mesin pencari bisa mengindeks dan memberikan hasil yang baik. Lihat FAQ tentang Go di https://golang.org/doc/faq#go_or_golang untuk informasi lebih lanjut.

## 1.2. Lisensi Go

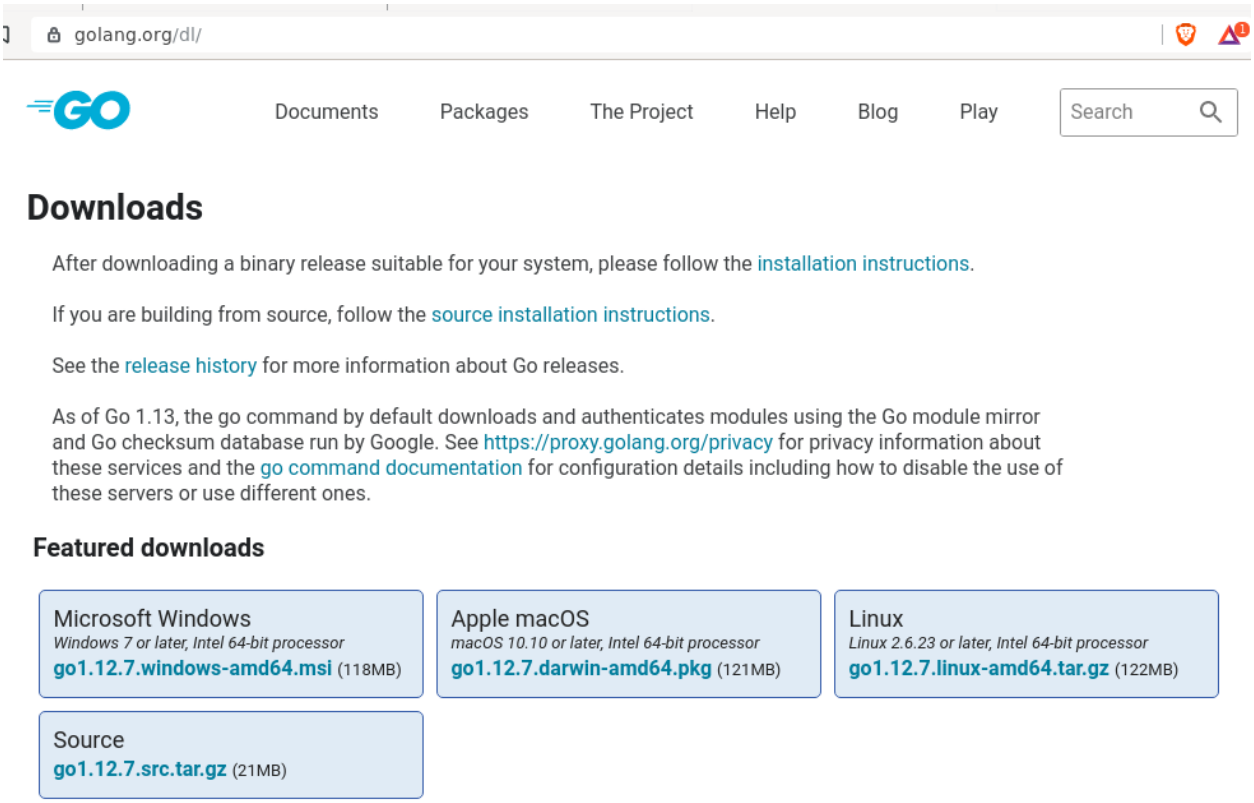
Go didistribusikan dengan menggunakan lisensi modifikasi dari BSD. Lisensi lengkap dari Go bisa diakses di Lisensi Go. Secara umum, penggunaaan lisensi ini mempunyai implikasi sebagai berikut:

- boleh digunakan untuk keperluan komersial maupun non-komersial tanpa batasan
- boleh memodifikasi sesuai keperluan
- boleh mendistribusikan
- boleh memberikan sublisensi ke pihak lain
- boleh memberikan garansi
- tidak boleh menggunakan merk dagang Go
- tanpa jaminan dan jika terjadi kerusakan terkait penggunaan software ini maka pemberi lisensi tidak bisa dituntut
- jika mendistribusikan harus mengikutsertakan pemberitahuan hak cipta.

## 1.3. Instalasi Go

Go tersedia pada berbagai platform. Proyek Go sendiri secara resmi mendukung platform Linux, FreeBSD, MacOSX, dan Windows. Dukungan tersebut merupakan dukungan resmi dan

distribusi `binary executable` dari berbagai platform tersebut tersedia di repository download Go seperti pada gambar berikut:




Dengan dukungan tersebut, Proyek Go akan menerima laporan `bugs` terkait dengan distribusi pada berbagai platform tersebut. Meski demikian, bukan berarti platform-platform lain tidak bisa menggunakan Go karena distribusi dalam bentuk kode sumber tersedia dan telah berhasil dikompilasi ke berbagai platform: NetBSD, OpenBSD, DragonFlyBSD, dan lain-lain. Informasi mengenai platform-platform yang mungkin bisa digunakan oleh Go bisa diperoleh di wiki.

### 1.3.1. Download dan Install Go

Download dan instalasi Go pada tulisan ini adalah download dan instalasi untuk lebih dari satu versi Go dan masing-masing menggunakan workspace sendiri-sendiri. Hal ini disebabkan karena seringkali software yang dibangun ditargetkan untuk lebih dari satu versi, misalnya Go 1.11 ke atas (Go 1.11.x, 1.12.x, dan 1.13.x). Kondisi ini menjadi tidak sederhana karena penulis tidak ingin mencampuraduk kode sumber yang dibuat menggunakan masing-masing versi. Go sendiri menyarankan untuk menggunakan satu workspace untuk semua proyek Go yang kita buat. Satu workspace saja tidak masalah jika hanya mentargetkan satu versi. Di bagian ini penulis akan menjelaskan konfigurasi yang penulis gunakan untuk menangani masalah tersebut.

*Lokasi instalasi Go*

Go akan diinstall di direktori $HOME/software/go-dev-tools/goVERSI

VERSI = versi dari Go yang akan diinstall, misalnya go1.12.7

Lokasi instalasi tersebut digunakan penulis karena penulis mempunyai lebih dari 1 versi Go, jika nanti ada versi lainnya, versi lain tersebut akan di-install (misal versi 1.13.0) di $HOME/software/go-dev-tools/go1.13.0

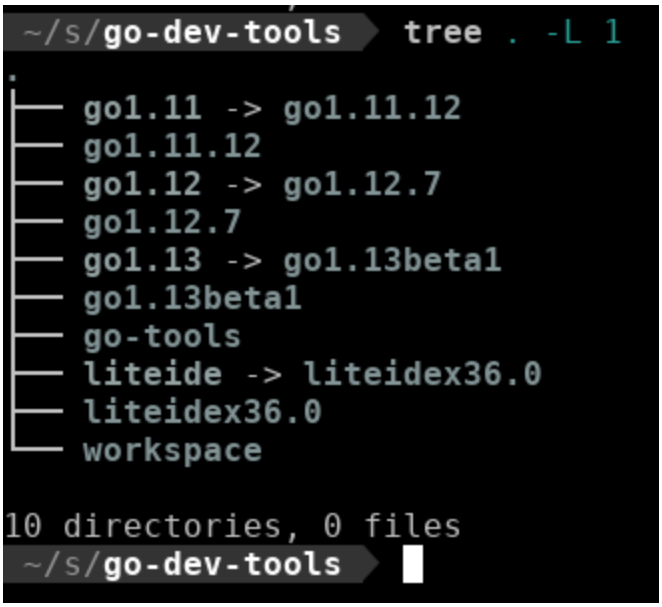
Meski mendukung banyak platform, di buku ini hanya akan dibahas penggunaan Go di platform Linux. Pada dasarnya peranti pengembang yang disediakan sama. Silahkan menyesuaikan dengan platform yang anda gunakan. Untuk instalasi berikut ini, ambil distribusi yang sesuai dengan platform di komputer anda. Untuk pembahasan ini, digunakan `go1.12.7.linux-amd64.tar.gz`. Setelah itu, ikuti langkah-langkah berikut:

```
$ ls -la
total 475520
drwxr-xr-x   4 bpdp bpdp      4096 Jul 21 08:16 ./
drwxr-xr-x  88 bpdp bpdp      4096 Jul 12 14:17 ../
-rw-r--r--   1 bpdp bpdp 127959471 Jul 19 04:41 go1.12.7.linux-
amd64.tar.gz
...
...
$ mkdir -p $HOME/software/go-dev-tools
$ cd $HOME/software/go-dev-tools
$ tar -xvf ~/master/go/go1.12.7.linux-amd64.tar.gz
$ mv go go1.12.7
```

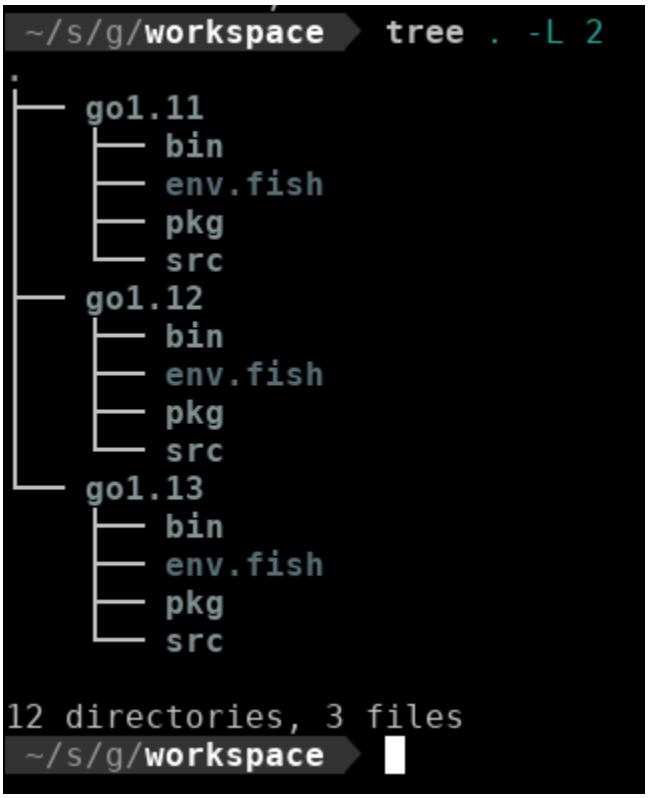
Setelah menjalankan langkah-langkah di atas, Go sudah terinstall di direktori `$HOME/software/go-dev-tools/go1.12.7`

### 1.3.2. Konfigurasi Variabel Lingkungan Sistem Operasi, Compiler Go, dan Workspace

Untuk konfigurasi kompiler, ada tiga langkah yang perlu dilakukan: download, ekstrak pada lokasi tertentu, dan terakhir setting environment variables. Pada konfigurasi ini, compiler dan workspace berada pada `$HOME/software/go-dev-tools/`. Lokasi ini selanjutnya akan kita sebut dengan GODEVTOOLS_HOME. Setelah download dan install compiler Go seperti langkah di atas, buat struktur direktori sebagai berikut (untuk go1.12.7 sudah dibuat dengan cara di atas):




Semua versi Go ada di $GODEVTOOLS_HOME. Direktori workspace digunakan untuk menyimpan kode sumber yang kita buat seesuai dengan versi Go yang kita targetkan. Untuk setiap direktori di workspace, buat struktur dan 1 file env.sh sebagai berikut:




Isi dari file env.sh menyesuaikan shell yang digunakan:

*Bash*

```
export GOPATH=`pwd`
export PATH=$PATH:$GOPATH/bin
```

*Fish*

```
set -x GOPATH (pwd)
set -x PATH $PATH $GOPATH/bin
```

Go menggunakan beberapa variabel lingkungan sistem operasi. Supaya berfungsi dengan baik, tetapkan nilai-nilai variabel lingkungan tersebut di file inisialisasi shell. Jika menggunakan **Fish**, maka inisialisasi tersebut ada di `$HOME/.config/fish/config.fish`. Jika menggunakan **Bash**, maka inisialisasi tersebut diletakkan di `$HOME/.bashrc`). Meski bisa diletakkan pada file tersebut, penulis menyarankan untuk meletakkan pada suatu file text biasa dan kemudian di-**source**. Pada bagian ini, penulis akan meletakkan di file `%HOME/env/fish/go/go1.12`.

```
set GODEVTOOLS_HOME /home/bpdp/software/go-dev-tools

set GO_HOME $GODEVTOOLS_HOME/go1.12
set LITEIDE_HOME $GODEVTOOLS_HOME/liteide
set GOTOOLS $GODEVTOOLS_HOME/go-tools

set -x GOROOT $GO_HOME
set -x GOOS linux
set -x GOARCH amd64
set -x GOHOSTOS linux
set -x GOHOSTARCH amd64

alias godev='cd $GODEVTOOLS_HOME/workspace/go1.12'
alias godevtools='cd $GOTOOLS'

set -x PATH $PATH $GO_HOME/bin $LITEIDE_HOME/bin $GOTOOLS/bin
```

Jika menggunakan **Bash**:

```
GODEVTOOLS_HOME=/home/bpdp/software/go-dev-tools

GO_HOME=$GODEVTOOLS_HOME/go/go1.12.7
LITEIDE_HOME=$GODEVTOOLS_HOME/liteide
GOTOOLS=$GODEVTOOLS_HOME/go-tools

export GOROOT=$GO_HOME
export GOOS=linux
export GOARCH=amd64
export GOHOSTOS=linux
export GOHOSTARCH=amd64

export PATH=$PATH:$GO_HOME/bin:$LITEIDE_HOME/bin:$GOTOOLS:
$GO3RDPARTYTOOLS/bin

alias godev='cd $GODEVTOOLS_HOME/workspace/go1.12'
alias godevtools='cd $GOTOOLS'
```

Dengan memasukkan beberapa variabel lingkungan tersebut ke file, saat kita ingin menggunakan Go, tinggal di - **source** sebagai berikut:

```
$ source ~/env/fish/go/go1.12.7
```

Setelah itu, Go bisa digunakan. Untuk melihat hasil, eksekusi perintah `go env`, hasilnya seharusnya adalah sebagai berikut:

```
$ go env
GOARCH="amd64"
GOBIN=""
GOCACHE="/home/bpdp/.cache/go-build"
GOEXE=""
GOFLAGS=""
GOHOSTARCH="amd64"
GOHOSTOS="linux"
GOOS="linux"
GOPATH="/home/bpdp/go"
GOPROXY=""
GORACE=""
GOROOT="/home/bpdp/software/go-dev-tools/go1.12"
GOTMPDIR=""
GOTOOLDIR="/home/bpdp/software/go-dev-tools/go1.12/pkg/tool/linux_amd64"
GCCGO="gccgo"
CC="gcc"
CXX="g++"
CGO_ENABLED="1"
GOMOD=""
CGO_CFLAGS="-g -O2"
CGO_CPPFLAGS=""
CGO_CXXFLAGS="-g -O2"
CGO_FFLAGS="-g -O2"
CGO_LDFLAGS="-g -O2"
PKG_CONFIG="pkg-config"
GOGCCFLAGS="-fPIC -m64 -pthread -fmessage-length=0 -fdebug-prefix
-map=/tmp/go-build584380045=/tmp/go-build -gno-record-gcc-switches"
$
```

Variabel $GOPATH seharusnya menunjuk ke workspace, baru akan berisi nilai yang benar (bukan $HOME/go) jika sudah men-**source** file `env.sh` di workspace.

Saat bekerja menggunakan Go, pada dasarnya kita akan menemukan berbagai macam proyek yang bisa dikategorikan menjadi 2 berdasarkan output dari proyek tersebut:

1. **Ready-to-use application**: aplikasi yang siap dipakai, biasanya didistribusikan dalam bentuk **binary executable(s)** atau kode sumber seperti nsq, Hugo, dan lain-lain.
2. Pustaka / **library** maupun aplikasi yang kita kembangkan sendiri.

Untuk dua kategori ini, ada dua perlakuan.

**Ready-to-use application**

Untuk kategori ini, siapkan lokasi khusus di media penyimpan untuk menyimpan hasil binary executable, setelah itu set `PATH`, `GOPATH` dan `go get -u -v <repo-url>`. Berikut adalah setting

pada komputer penulis:

```
~/s/g/go-tools  tree . -L 1
.
├── bin
├── env.fish
├── go-pkg-needed.sh
├── pkg
└── src

3 directories, 2 files
~/s/g/go-tools  cat go-pkg-needed.sh
#!/usr/bin/fish
#go get -u -v github.com/nsf/gocode
# diganti:
go get -u -v github.com/stamblerre/gocode
go get -u -v github.com/rogpeppe/godef
go get -u -v golang.org/x/lint/golint
go get -u -v github.com/lukehoban/go-outline
go get -u -v github.com/sqs/goreturns
go get -u -v golang.org/x/tools/...
go get -u -v github.com/uudashr/gopkgs
go get -u -v github.com/newhook/go-symbols
go get -u -v github.com/go-delve/delve/cmd/dlv
go get -u -v github.com/pointlander/peg
go get -u -v github.com/songgao/colorgo
go get -u -v github.com/motemen/gore
go get -u -v github.com/onsi/ginkgo/ginkgo
go get -u -v github.com/onsi/gomega/...
go get -u -v github.com/smartystreets/goconvey
go get -u -v github.com/blynn/nex
go get -u -v github.com/zmb3/gogetdoc
~/s/g/go-tools
```

Isi dari file `go-pkg-needed.sh` adalah sebagai berikut, anda bisa menambah atau mengurangi sesuai kebutuhan:

```
#!/usr/bin/fish

# ganti di atas dengan #!/usr/bin/bash jika anda menggunakan Bash
go get -u -v github.com/stamblerre/gocode
go get -u -v github.com/rogpeppe/godef
go get -u -v golang.org/x/lint/golint
go get -u -v github.com/lukehoban/go-outline
go get -u -v github.com/sqs/goreturns
go get -u -v golang.org/x/tools/...
go get -u -v github.com/uudashr/gopkgs
go get -u -v github.com/newhook/go-symbols
go get -u -v github.com/go-delve/delve/cmd/dlv
go get -u -v github.com/pointlander/peg
go get -u -v github.com/songgao/colorgo
go get -u -v github.com/motemen/gore
go get -u -v github.com/onsi/ginkgo/ginkgo
go get -u -v github.com/onsi/gomega/...
go get -u -v github.com/smartystreets/goconvey
go get -u -v github.com/blynn/nex
go get -u -v github.com/zmb3/gogetdoc
go get -u -v golang.org/x/tools/gopls
```

Dengan konfigurasi seperti itu, kerjakan berikut ini untuk install:

```
$ source env/fish/go/go1.12.7
$ godevtools
$ source env.sh
$ ./go-pkg-needed.sh
```

Perintah `source env.sh` di atas berguna antara lain untuk menetapkan `$GOPATH` ke `$GOTOOLS`. Setelah proses sebentar, hasil **binary executables** akan diletakkan pada $GOTOOLS/bin dan bisa kita jalankan langsung.

jangan meletakkan paket-paket **executables** ini jika $GOPATH belum menunjukkan nilai yang benar karena nanti akan tercampur dengan **binary executables** dari distribusi Go.

**Pustaka / library maupun aplikasi yang kita kembangkan sendiri**

Untuk keperluan ini biasanya kita menggunakan `modules` yang mulai ada pada versi Go 1.11 dan akan stabil pada versi 1.13. Modules ini akan kita bahas tersendiri.

### 1.3.3. Menguji Instalasi Go

Kode sumber Go yang kita buat bisa dijalankan / dieksekusi tanpa harus dikompilasi (jadi seperti script Python atau Ruby) atau bisa juga dikompilasi lebih dulu untuk menghasilkan `binary executable`. Selain menghasilkan `binary executable`, sebenarnya ada paket pustaka yang dimaksudkan untuk digunakan dalam program (disebut sebagai `package`). Package akan dibahas lebih lanjut pada bab-bab berikutnya.

Untuk menguji, buat program sederhana seperti listing `hello.go`. Setelah itu, gunakan `go run namafile.go` untuk menjalankan secara langsung atau dikompilasi lebih dulu dengan `go build namafile.go`.

```
// hello.go
package main

import "fmt"

func main() {
    fmt.Printf("hello, world\n")
}
```

Berikut ini adalah langkah-langkah untuk mengeksekusi `hello.go`:

```
$ go run hello.go
hello, world
$ go build hello.go
$ ls -la
total 1980
drwxr-xr-x 2 bpdp bpdp    4096 Jul 21 10:41 ./
drwxr-xr-x 3 bpdp bpdp    4096 Jul 21 10:40 ../
-rwxr-xr-x 1 bpdp bpdp 2014135 Jul 21 10:41 hello*
-rw-r--r-- 1 bpdp bpdp      86 Jul 21 10:40 hello.go
$ file hello
hello: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically
linked, Go
BuildID=-WQRW-exSunj5kUQwAX9/zYf98wtRiMVNHHsFRqn-/1wN0h--
29c_4cVsSKleo/4LgNCTrEswXoVuMCJgkH, not
stripped
$ strip hello
$ ls -la
total 1400
drwxr-xr-x 2 bpdp bpdp    4096 Jul 21 10:42 ./
drwxr-xr-x 3 bpdp bpdp    4096 Jul 21 10:40 ../
-rwxr-xr-x 1 bpdp bpdp 1420104 Jul 21 10:42 hello*
-rw-r--r-- 1 bpdp bpdp      86 Jul 21 10:40 hello.go
$ ./hello
hello, world
$ file hello
hello: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically
linked, Go
BuildID=-WQRW-exSunj5kUQwAX9/zYf98wtRiMVNHHsFRqn-/1wN0h--
29c_4cVsSKleo/4LgNCTrEswXoVuMCJgkH,
stripped
$
```

## 1.4. Memahami Lingkungan Peranti Pengembangan Go

Saat menginstall Go, kita akan memperoleh 3 buah file `binary executable`:

```
$ pwd
/home/bpdp/software/go-dev-tools/go1.12.7/bin
$ ls -la
total 34744
drwxr-xr-x  2 bpdp bpdp     4096 Jul  9 04:32 ./
drwxr-xr-x 10 bpdp bpdp     4096 Jul  9 04:29 ../
-rwxr-xr-x  1 bpdp bpdp 14617729 Jul  9 04:31 go*
-rwxr-xr-x  1 bpdp bpdp 17422226 Jul  9 04:32 godoc*
-rwxr-xr-x  1 bpdp bpdp  3525802 Jul  9 04:31 gofmt*
$
```

Penjelasan untuk masing-masing akan diuraikan di sub-sub bab berikut.

### 1.4.1. go

`go` merupakan peranti untuk mengelola kode sumber Go yang kita buat. Beberapa argumen dari `go` adalah:

```
$ go version
go version go1.12.7 linux/amd64
$ go
Go is a tool for managing Go source code.

Usage:

    go <command> [arguments]

The commands are:

    bug         start a bug report
    build       compile packages and dependencies
    clean       remove object files and cached files
    doc         show documentation for package or symbol
    env         print Go environment information
    fix         update packages to use new APIs
    fmt         gofmt (reformat) package sources
    generate    generate Go files by processing source
    get         download and install packages and dependencies
    install     compile and install packages and dependencies
    list        list packages or modules
    mod         module maintenance
    run         compile and run Go program
    test        test packages
    tool        run specified go tool
    version     print Go version
    vet         report likely mistakes in packages

Use "go help <command>" for more information about a command.

Additional help topics:

    buildmode   build modes
    c           calling between Go and C
    cache       build and test caching
    environment environment variables
    filetype    file types
    go.mod      the go.mod file
    gopath      GOPATH environment variable
    gopath-get  legacy GOPATH go get
    goproxy     module proxy protocol
    importpath  import path syntax
    modules     modules, module versions, and more
    module-get  module-aware go get
    packages    package lists and patterns
    testflag    testing flags
    testfunc    testing functions

Use "go help <topic>" for more information about that topic.
$
```

### 1.4.2. godoc

godoc merupakan peranti untuk menampilkan dokumentasi paket pustaka standar Go atau menampilkan server untuk dokumentasi Go (mirip seperti yang terdapat pada [website dokumentasi Go](http://golang.org/doc/).

```
$ godoc --help
usage: godoc -http=localhost:6060
  -analysis string
        comma-separated list of analyses to perform (supported: type,
pointer). See http://golang.org/lib/godoc/analysis/help.html
  -goroot string
        Go root directory (default "/home/bpdp/software/go-dev-
tools/go1.12")
  -http string
        HTTP service address (default "localhost:6060")
  -index
        enable search index
  -index_files string
        glob pattern specifying index files; if not empty, the index is
read from these files in sorted order
  -index_interval duration
        interval of indexing; 0 for default (5m), negative to only index
once at startup
  -index_throttle float
        index throttle value; 0.0 = no time allocated, 1.0 = full throttle
(default 0.75)
  -links
        link identifiers to their declarations (default true)
  -maxresults int
        maximum number of full text search results shown (default 10000)
  -notes string
        regular expression matching note markers to show (default "BUG")
  -play
        enable playground
  -templates string
        load templates/JS/CSS from disk in this directory
  -timestamps
        show timestamps with directory listings
  -url string
        print HTML for named URL
  -v    verbose mode
  -write_index
        write index to a file; the file name must be specified with
-index_files
  -zip string
        zip file providing the file system to serve; disabled if empty
$
```

### 1.4.3. gofmt

`gofmt` merupakan peranti untuk mem-format kode sumber dalam bahasa pemrograman Go.

```
$ gofmt --help
usage: gofmt [flags] [path ...]
  -cpuprofile string
        write cpu profile to this file
  -d    display diffs instead of rewriting files
  -e    report all errors (not just the first 10 on different lines)
  -l    list files whose formatting differs from gofmt's
  -r string
        rewrite rule (e.g., 'a[b:len(a)] -> a[b:]')
  -s    simplify code
  -w    write result to (source) file instead of stdout
$
```

Untuk melihat bagaimana `gofmt` bisa digunakan untuk membantu memformat kode sumber, buat kode sumber sederhana berikut ini:

```
// hello-unformatted.go
package main
import "fmt"
func main() {
    fmt.Printf("halo\n") // menampilkan tulisan
       fmt.Printf("dunia")  // ini tulisan baris kedua
}
```

Format file kode sumber di atas sebagai berikut:

```
$ gofmt hello-unformatted.go > hello-formatted.go
```

Hasilnya adalah sebagai berikut:

```go
// hello-formatted.go
package main

import "fmt"

func main() {
    fmt.Printf("halo\n") // menampilkan tulisan
    fmt.Printf("dunia")  // ini tulisan baris kedua
}
```

# Bab 2. IDE untuk Go

## 2.1. Sub Section 1

# Bab 3. Struktur Program Go

## 3.1. Sub Section 1

# Part II: Sintaksis Go

Bagian ini membahas tentang sintaks dari bahasa pemrograman Go.

# Bab 4. Sintaksis Dasar Go

## 4.1. Sub Section 1

# Bab 5. Fungsi

## 5.1. Sub Section 1

# Bab 6. Penanganan Kesalahan

## 6.1. Sub Section 1

# Bab 7. Struktur Data

## 7.1. Sub Section 1

# Bab 8. Konkurensi dan Paralelisme

## 8.1. Sub Section 1

# Part III: Go untuk Kasus Spesifik

Bagian ini membahas tentang penggunaan Go untuk menyelesaikan masalah tertentu dalam pembuatan software. Penyelesaian masalah bisa dilakukan dengan menggunakan pustaka standar maupun pustaka pihak ketiga. Pada bagian ini, pembahasan tidak hanya ditujukan pada pustaka standar tetapi juga pustaka pihak ketiga selama bisa digunakan untuk menyelesaiakn masalah dalam domain masalah tertentu.

# Bab 9. Testing

## 9.1. Sub Section 1

# Bab 10. I/O dan FIlesystems

## 10.1. Sub Section 1

# Bab 11. Akses Basis Data

## 11.1. Sub Section 1

# Bab 12. Sistem Terdistribusi

## 12.1. Sub Section 1

# Bab 13. Aplikasi Web

## 13.1. Sub Section 1