



# Scalable Web Service With Golang

## sesi 1

---

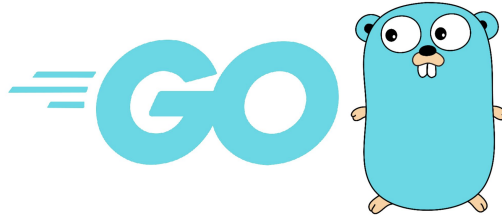




# Go Programming<sup>+</sup> Introduction

## Go Programming Introduction - Sesi 1

# Introduction



Go adalah sebuah bahasa pemrograman open-source yang dibuat di Google pada tahun 2007 oleh Robert Griesemer, Rob Pike, dan Ken Thompson. Berikut merupakan kelebihan dari bahasa Go:

- Clean And Simple: Go membuat segala sesuatu menjadi kompak dan indah. Kita dapat melakukan banyak hal hanya dalam beberapa baris saja.
- Concurrent: Go dapat menjalankan fungsi-fungsi dengan ringan seperti thread dan dapat berkomunikasi antara fungsi yang satu dan fungsi yang lain.
- Fast: Proses compile di Go dapat dilakukan dengan cepat seperti proses compile pada bahasa pemrograman C. Selain itu, Go di desain untuk komputer *multi-core*.

## Go Programming Introduction - Sesi 1

# Installation

### #Windows

- Dapatkan installer dari Go pada link berikut <https://golang.org/dl/>. Jangan lupa untuk memilih installer untuk sistem operasi Windows.
- Setelah selesai terdownload, maka kalian perlu menjalankan installernya. Setelah itu klik next hingga proses instalasi selesai. Pada dasarnya, jika kalian tidak merubah path pada saat instalasi, Go akan ter-install di C:\go. Path tersebut secara otomatis akan didaftarkan dalam PATH environment variable.
- Bukalah Command Prompt / CMD, kemudian eksekusi perintah berikut untuk mengecek versi Go.



Jika versi dari bahasa Go telah keluar pada CMD, maka itu menandakan bahwa kalian telah berhasil menginstall Go

## Go Programming Introduction - Sesi 1

# Installation

#MacOS

- Dapatkan installer dari Go pada link berikut <https://golang.org/dl/>. Jangan lupa untuk memilih installer untuk sistem operasi Mac. Kemudian ikutilah instruksi dari proses installasinya.
- Jika proses instalasi selesai, maka jalankanlah perintah dibawah ini pada terminal



- Jika versi dari bahasa Go telah keluar pada terminal, maka itu menandakan bahwa kalian telah berhasil menginstall Go.

## Go Programming Introduction - Sesi 1

# Installation

#MacOS

- Tambahkan path binary Go ke PATH environment variable.



```
$ echo "export PATH=$PATH:/usr/local/go/bin" >> ~/.bash_profile
$ source ~/.bash_profile
```

## Go Programming Introduction - Sesi 1

# Installation

#Linux

- Dapat versi terbaru dari Go melalui link berikut <https://golang.org/dl/>.
- Kemudian buka terminal dan arahkan kepada direktori home dengan menjalankan perintah `cd ~`.
- Unduh versi terbaru dari Go. Misalkan contohnya pada saat ini versi terbarunya adalah (1.16.7). Jalankan perintah berikut:

```
curl -O https://golang.org/dl/go1.16.7.linux-amd64.tar.gz
```

- Extract file arsip nya kepada `/usr/local` dengan perintah berikut:

```
sudo tar -xzvf go1.16.7.linux-amd64.tar.gz -C /usr/local
```



## Go Programming Introduction - Sesi 1

# Installation

#Linux

- Tambahkan path binary Go ke PATH environment variable.

```
echo "export PATH=$PATH:/usr/local/go/bin" >> ~/.bashrc  
source ~/.bashrc
```

- Lihat versi Go dengan perintah berikut:

```
go version
```

- Jika versi dari bahasa Go telah keluar pada terminal, maka itu menandakan bahwa kalian telah berhasil menginstall Go.



## Go Root & Go Path

- Setelah kita selesai menginstall Go, secara otomatis akan muncul environment variable GOROOT. Isi dari variabel ini adalah lokasi dimana Go ter-install.
- Sebagai contoh di Windows, ketika Go sudah terinstall di C:\go, maka path tersebut akan menjadi isi dari GOROOT.
- Go mempunyai aturan sendiri dalam mengatur workspace dari seluruh codingan Go kita.
- Pada umumnya, programmer Go meletakkan seluruh codingan Go nya pada satu Go workspace.
- Go Workspace adalah suatu direktori pada file system kita yang dimana path kepada file system tersebut telah di simpan didalam environment variable bernama GOPATH.



## Setup Go Workspace

- Go Workspace akan mempunyai 3 buah sub folder yaitu
  - bin: Folder yang berisikan executable file hasil dari build.
  - pkg: Folder yang berisikan file-file hasil dari proses kompilasi.
  - src: Folder yang berisikan project Go disimpan.
- Pada dasarnya, Go Workspace akan berada didalam sub direktori bernama go, yang terdapat di dalam direktori home, atau \$HOME/go, yang dimana \$HOME adalah sebuah variable yang menyimpan direktori home seperti /home/john
- Bagi pengguna Windows, tambahkan path dari sub direktori go tersebut kepada path variable dengan nama GOPATH.
- Bagi pengguna Mac OS, export path kepada ~/.bash\_profile. Untuk Linux, export ke ~/.bashrc.

```
$ echo "export GOPATH=$HOME/Documents/go" >> ~/.bash_profile
$ source ~/.bash_profile
```

## Go Programming Introduction - Sesi 1

# Setup Go Workspace

- Untuk memeriksa apakah path sudah terdaftar dengan benar, maka dapat menjalankan perintah echo \$GOPATH.

```
└─ echo $GOPATH  
  
/Users/sofyan/go
```

- Jika GOPATH sudah terdaftar dengan benar, maka kita perlu menyiapkan 3 buah sub folder di dalam direktori/folder go tersebut.

```
> bin  
> pkg  
> src
```

## Go Modules

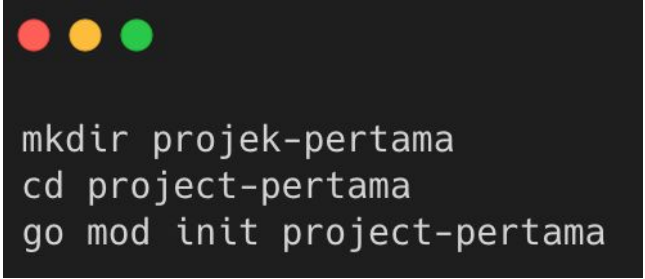
- Go Modules adalah sebuah manajemen dependensi dari bahasa Go. Seperti halnya npm pada bahasa Javascript atau composer pada bahasa PHP.
- Di dukung pertama kali pada Go versi 1.11, tetapi finalisasi terakhir dari implementasi nya terjadi pada Go versi 1.13.
- Digunakan untuk menginisialisasi sebuah projek, sekaligus melakukan manajemen terhadap 3rd party atau library/package lain yang digunakan.
- Go Modules digunakan melalui CLI.
- Dengan menggunakan Go Modules, maka kita dapat membuat project Go di luar dari Go Workspace.



## Go Programming Introduction - Sesi 1

# Initializing Project

Jalankan perintah di bawah ini untuk menginisialisasi project menggunakan Go Modules.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains three lines of white text representing terminal commands.

```
mkdir projek-pertama  
cd project-pertama  
go mod init project-pertama
```

Perintah `go mod init <nama-project>` digunakan untuk membuat Go Modules atau menginisialisasi sebuah project. Nama dari Go Modules lebih baik disamakan dengan nama dari folder projectnya, tetapi jika tidak disamakan maka tidak menjadi masalah.

## Go Programming Introduction - Sesi 1

# Command

Pengembangan aplikasi Go tak jauh dari hal-hal yang berbau CLI atau Command Line Interface. Proses inisialisasi proyek, kompilasi, testing, eksekusi program, semuanya dilakukan lewat command line.

Command `go mod init` digunakan untuk inisialisasi proyek pada Go (menggunakan Go Modules). Untuk nama proyek bisa menggunakan nama apapun, tapi pada umumnya adalah disamakan dengan nama direktori.

Nama proyek ini penting karena nantinya berpengaruh pada import path sub packages yang ada dalam proyek tersebut.

```
mkdir <nama-project>
cd <nama-project>
go mod init <nama-project>
```



## Go Programming Introduction - Sesi 1

### Command (go run)

Command `go run` digunakan untuk eksekusi file program (file ber-ekstensi `.go`). Cara penggunaannya dengan menuliskan command tersebut diikuti argumen nama file.

Berikut adalah contoh penerapan `go run` untuk eksekusi file program `main.go` yang tersimpan di path `project-pertama` yang path tersebut sudah diinisialisasi menggunakan perintah `go mod init`.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) at the top left. It contains two lines of text: `cd project-pertama` and `go run main.go`.


```
cd project-pertama
go run main.go
```

Command `go run` hanya bisa digunakan pada file yang nama package-nya adalah `main`

## Go Programming Introduction - Sesi 1

### Command (go run)

Jika ada banyak file yang package-nya main dan file-file tersebut berada pada satu direktori level dengan file utama, maka eksekusinya adalah dengan menuliskan semua file sebagai argument command go run.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The text 'go run main.go library.go' is displayed in a light gray monospaced font.

```
go run main.go library.go
```



## Go Programming Introduction - Sesi 1

### Command (go build)

Command ini digunakan untuk mengkompilasi file program.

Sebenarnya ketika eksekusi program menggunakan perintah `go run`, terjadi proses kompilasi juga. File hasil kompilasi akan disimpan pada folder temporary untuk selanjutnya langsung dieksekusi.

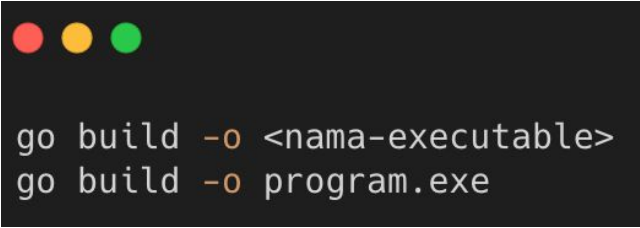
Berbeda dengan `go build`, command ini menghasilkan file executable atau binary pada folder yang sedang aktif.

```
05/03/2020 03:01 PM <DIR> .
05/03/2020 03:01 PM <DIR> ..
05/03/2020 02:18 PM      32 go.mod
05/03/2020 02:50 PM      79 main.go
05/03/2020 02:57 PM      70 main_test.go
05/03/2020 03:01 PM 2,142,720 project-pertama.exe
          4 File(s)      2,142,901 bytes
          2 Dir(s)  21,342,564,352 bytes free
```

Pada contoh di atas, projek program-pertama di-build, menghasilkan file baru pada folder yang sama, yaitu `program-pertama.exe`, yang kemudian dieksekusi. Secara bawaan nama projek akan otomatis dijadikan nama binary.

## Command (go build)

Untuk nama executable sendiri bisa diubah menggunakan flag -o. Contoh:



```
go build -o <nama-executable>  
go build -o program.exe
```

## Go Programming Introduction - Sesi 1

### Command (go get)

Command go get digunakan untuk men-download package.

Sebagai contoh saya ingin men-download package Gin-Gonic untuk Go pada projek project-pertama.

```
└─ go get github.com/gin-gonic/gin  
go: downloading github.com/gin-gonic/gin v1.7.4  
go get: added github.com/gin-gonic/gin v1.7.4
```

Pada contoh di atas, [github.com/gin-gonic/gin](https://github.com/gin-gonic/gin) adalah URL package Gin-Gonic yang merupakan sebuah web framework pada bahasa Go. Package yang sudah terunduh tersimpan dalam temporary folder yang ter-link dengan project folder dimana command go get dieksekusi, menjadikan projek tersebut bisa meng-import package terunduh.



## Go Programming Introduction - Sesi 1

### Command (go get)

Untuk mengunduh dependensi versi terbaru, gunakan flag -u pada command go get, misalnya:



```
go get -u github.com/gin-gonic/gin
```

Command go get harus dijalankan dalam folder project. Jika dijalankan di-luar project maka akan diunduh ke pada GOPATH.

## Go Programming Introduction - Sesi 1

# Create Your First Project With Go

Dari Penjelasan yang sudah didapatkan coba lakukan inisialisasi project GO.

Hasil Akhir:

```
▼ ○ project-pertama
  -go go.mod 1
```

ASK : BAGAIMANA CARANYA SUPAYA HASILNYA SEPERTI INI ?

```
[ .../project-pertama
  go run main.go
Hello World
  .../project-pertama
```

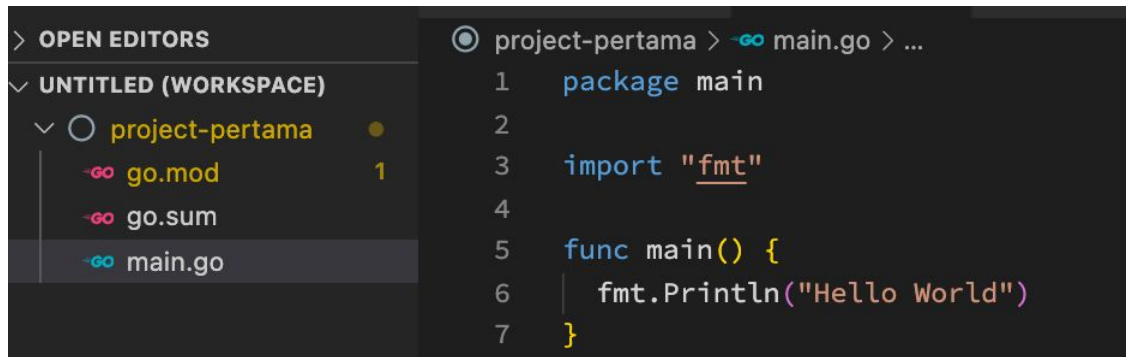
# Create Your First Project With Go

### JAWABAN : MENYIAPKAN FILE PROGRAM

File program disini maksudnya adalah file yang isinya source code Go. File ini berekstensi .go.

Di dalam project yang telah dibuat, siapkan sebuah file dengan nama bebas, yang jelas harus ber-ekstensi .go. Pada contoh ini penulis menggunakan nama file main.go.

Pembuatan file program bisa dilakukan lewat CLI atau browser, atau juga lewat editor. Pastikan file dibuat dalam projek folder.



The screenshot shows a code editor interface. On the left, the 'OPEN EDITORS' sidebar displays a project named 'project-pertama' with three files: 'go.mod', 'go.sum', and 'main.go'. The 'main.go' file is selected. The main editor area shows the content of 'main.go', which is a simple Go program that prints 'Hello World'.

```
project-pertama > main.go > ...
1 package main
2
3 import "fmt"
4
5 func main() {
6     fmt.Println("Hello World")
7 }
```

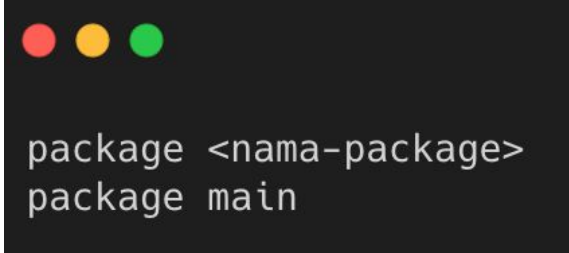


## Create Your First Project With Go

### Keyword Package

Setiap file program harus memiliki package. Setiap project harus ada minimal satu file dengan nama package main. File yang ber-package main, akan di eksekusi pertama kali ketika program di jalankan.

Cara menentukan package dengan menggunakan keyword package, berikut adalah contoh penulisannya.

A dark-themed terminal window with three colored window control buttons (red, yellow, green) in the top-left corner. It contains two lines of Go code: `package <nama-package>` and `package main`.

```
package <nama-package>
package main
```

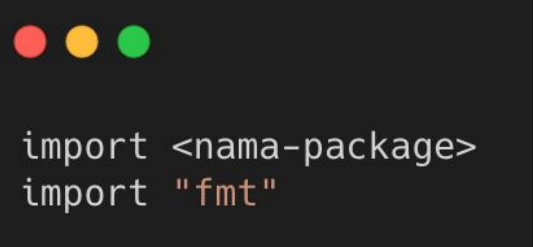
## Create Your First Project With Go

### Keyword import

Keyword import digunakan untuk meng-import atau memasukan package lain kedalam file program, agar isi dari package yang di-import bisa dimanfaatkan.

Package fmt merupakan salah satu package bawaan yang disediakan oleh Go, isinya banyak fungsi untuk keperluan I/O yang berhubungan dengan text.

Berikut adalah skema penulisan keyword import:



```
import <nama-package>  
import "fmt"
```





## Create Your First Project With Go

### Main function

Dalam sebuah proyek harus ada file program yang didalamnya berisi sebuah function bernama *main*. Function tersebut harus berada di file yang package-nya bernama *main*.

Function *main* adalah yang dipanggil pertama kali pada saat eksekusi program.

```
func main() {  
  
}
```



# Create Your First Project With Go

`fmt.Println()`

Function *fmt.Println* digunakan untuk memunculkan text ke layar (pada konteks ini, terminal atau CMD). Di program pertama yang telah kita buat, fungsi ini memunculkan tulisan Hello world.

Skema penulisan keyword `fmt.Println()` bisa dilihat pada contoh berikut.

```
fmt.Println("<isi-pesan>")  
fmt.Println("Hello World")
```

Function *fmt.Println* berada dalam package *fmt*, maka untuk menggunakannya perlu package tersebut untuk di-import terlebih dahulu.

Function *fmt.Println* dapat menampung parameter yang tidak terbatas jumlahnya. Semua data parameter akan dimunculkan dengan pemisah tanda spasi.

```
fmt.Println("Hello World", "Message", "From", "Go")
```



## Create Your First Project With Go

```
fmt.Print()
```

Function *fmt.Print* ini memiliki peran yang sama seperti function *fmt.Println*, pembedanya function *fmt.Print* tidak menghasilkan baris baru di akhir outputnya.

Perbedaan lainnya adalah, nilai pada parameter-parameter yang dimasukkan ke fungsi tersebut digabungkan tanpa pemisah. Tidak seperti pada function *fmt.Println* yang nilai paremeternya digabung menggunakan penghubung spasi.



## Create Your First Project With Go

`fmt.Print()`

```
fmt.Println("Airell Jordan")
fmt.Println("Airell", "Jordan")

fmt.Print("Airell Jordan\n")
fmt.Print("Airell", " Jordan\n")
fmt.Print("Airell", " ", "Jordan\n")
```

Kode di atas menunjukkan perbedaan antara *fmt.Println* dan *fmt.Print*. Output yang dihasilkan oleh 5 statement di atas adalah sama, meski cara yang digunakan berbeda.

Bila menggunakan *fmt.Println* tidak perlu menambahkan spasi di tiap kata, karena fungsi tersebut akan secara otomatis menambahkannya di sela-sela nilai. Berbeda dengan *fmt.Print*, perlu ditambahkan spasi, karena fungsi ini tidak menambahkan spasi di sela-sela nilai parameter yang digabungkan.



# Create Your First Project With Go

Comment / KOMENTAR

Komentar Multiline

```
package main

import "fmt"

func main() {

    //komentar kode
    //menghasilkan pesan Hello World

    fmt.Println("Hello World")

    // fmt.Println("Baris ini tidak akan dieksekusi")
}
```



# Create Your First Project With Go

Comment / KOMENTAR

Komentar Inline

```
package main

import "fmt"

func main() {

    /*
    komentar kode
    menghasilkan pesan Hello World
    */

    fmt.Println("Hello World")

    // fmt.Println("Baris ini tidak akan dieksekusi")
}
```

