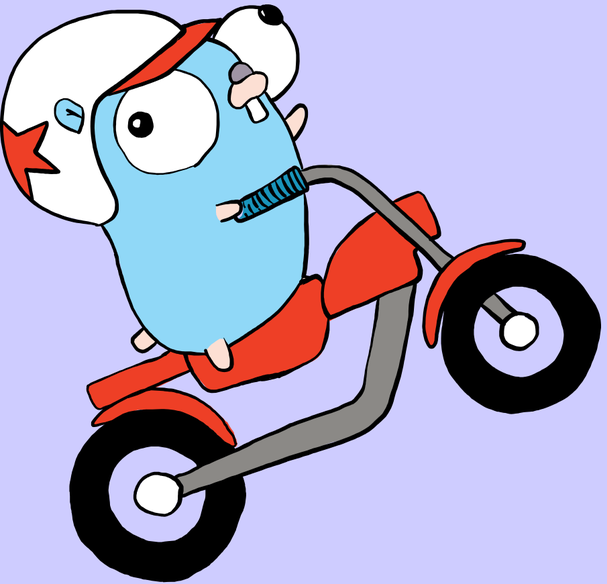
GO

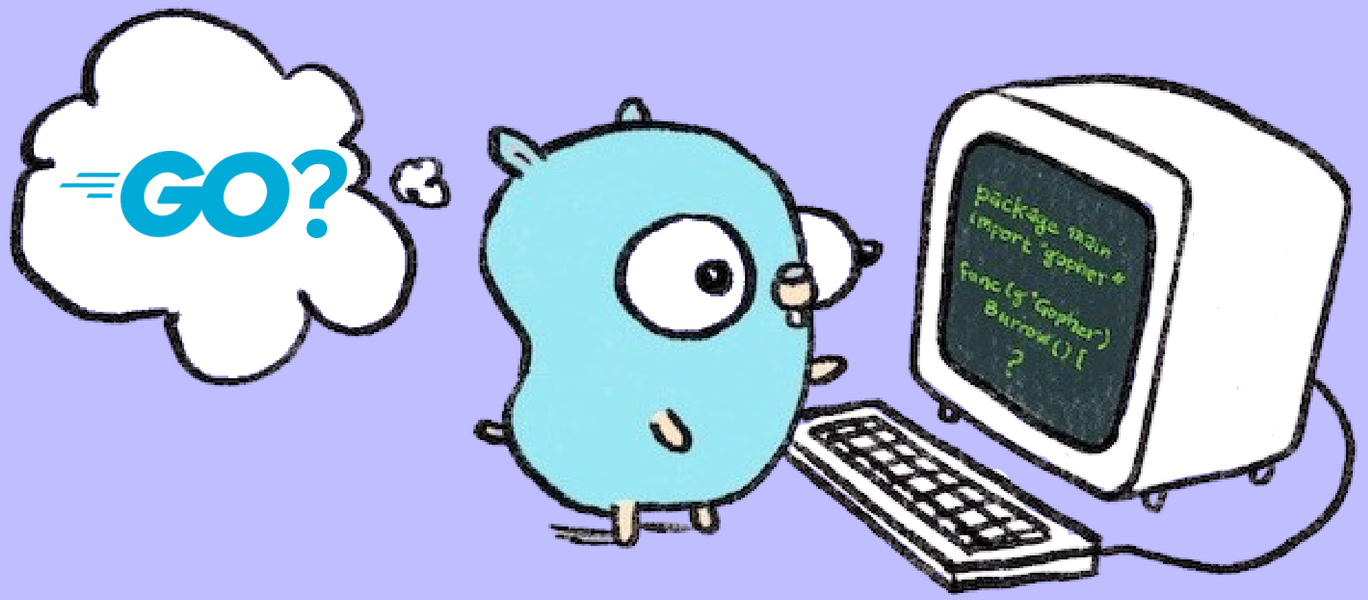
20211 황다인

Index

1. What is GO
2. Basics



What is GO



What is GO

What is Go

로버트 그리즈머

롭 파이크

케네스 톰슨

세 사람 모두 C언어를 매우 싫어했음

What is GO

C언어의 복잡함을 제거

↓

What is GO

break	default	func	interface	select
case	defer	go	map	struct
chan	else	goto	package	switch
const	fallthrough	if	range	type
continue	for	import	return	var

Google

AMERICAN EXPRESS

mercado libre

bitly

Capital One

CLOUDFLARE

Meta

Microsoft

WILD LIFE

ByteDance

NETFLIX

RIOT GAMES

salesforce

twitch

Uber

What is GO

BUT...

What is GO

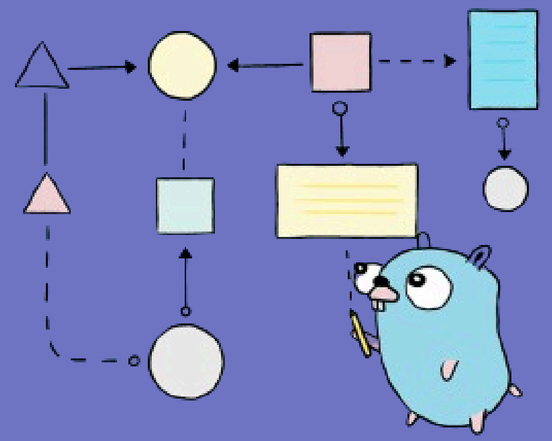
C

GO

What is GO

https://go.dev/play

Basics



main()

```go
pakage main

import (
    "fmt"
)

func main() {
    fmt.Print("Hello Go")
}
```

print

fmt:Formatted I/O
≈ stdio.h(C언어)

```
# 기본 출력 함수
fmt.Print()

# 공백 포함 출력 함수
fmt.Println()

# 포맷 지정 출력 함수
fmt.Printf()
```

variable

상수는 var 대신 const

```
# 어느 위치에서도 사용 가능

var (name) (type)

var (name1), (name2) (type)

var (name) = (value)
```

variable

```
var (
    i int
    b bool
    s string
)

# 함수 안에서만 사용 가능

(name) := (value)
```

data type

```
var integer int # (int / int8 / int16 / int32 / ... )

var str string

var float float64 # (float32 / float 64)

var boolean bool

var byt byte # (= uint8)

var run rune # (= int32)

var complex complex128 # (complex64, complex128)
```

구분	연산자	연산	피연산자 타입
사칙 연산과 나머지	+	덧셈	정수, 실수, 복소수, 문자열
	-	뺄셈	정수, 실수, 복소수
	*	곱셈	정수, 실수, 복소수
	/	나눗셈	정수, 실수, 복소수
	%	나머지	정수
비트 연산	&	AND 비트 연산	정수
	|	OR 비트 연산	정수
	^	XOR 비트 연산	정수
	&^	비트 클리어	정수
시프트 연산	<<	왼쪽 시프트	정수 << 양의 정수
	>>	오른쪽 시프트	정수 >> 양의 정수

if

```
if v < 0 { # 조건은 bool값으로

}

if v := 10; v > 0 { #v는 if 외부에서는 사용이 불가

}
```

if

```
if v < 0 {

} else if v < 1 {

} else {

}
```

switch

```
switch {
    case v == 1:

    case v == 2:
        fallthrough #case를 종료X
    default:

}
```

for

```
for i := 0; i < 10; i++ {
        sum += i
}

sum = 1
for sum < 1000 {
    sum += sum
}

for {}
```

pointer

```
i, j := 123, 456

p := &i

q := j

fmt.Print(*p, q)
```

& → 포인터 생성

pointer

```
i, j := 123, 456

p := &i

q := j

fmt.Print(*p, q)
```

* → 값 가져오기

pointer

```
fmt.Println(p, *p)
```

console

```
0x34d050c5a130 123
```

struct

```
type name struct {
    X int
    Y int
}

var v1 = name{1, 2}

var v2 = name{X: 1, Y: 2}
```

array

```
# 선언과 초기화를 따로

var array [n](type)
array[0] = 1

#선언과 초기화를 같이

array1 := [n](type){value}
```

slice

```go
s := []int{1, 2, 3, 4, 5}

s1 := []struct {
    i int
    s string
  }{
    {1, "가"},
    {2, "나"},
  }
```

slice

```
                                ↱ capacity
s3 := make([]int, 0, 5)
                               ↳ length

s3 = append(s3, 1, 2, 3)


fmt.Print(s3[0:2])
```

map

```
#일반적 생성
var m map[string]string

#make로 생성
m1 := make(map[string]string)

m["a"] = "a"
m1["a"] = "a"
```

map

```
#값 확인
m[key]

#삭제
delete(m, key)

#키 존재 여부 확인
_, ok := m[key]
```

function

```go
func f1(x int, y int) int {
    return x + y
}

func f2(x, y string) (s string) {
    return
}
```

method

```go
type struc struct {
    X int
    Y int
}

func (s struc) fuc1() int {
    return s.X + s.Y
}
```

method

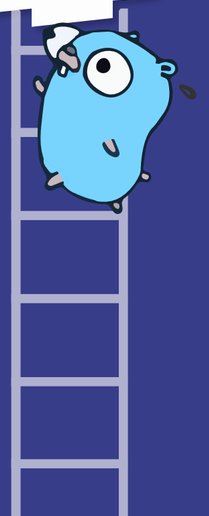
```
type F float64

        ↱ 원본 X
func (f F) fuc2() (s string) {
    if f > 0 {
        s = "양수"
    } else {
        s = "음수"
    }
    return
}
```

method

```
func (f *F) fuc3(float float64) {
    *f = F(float)
}
```

Thank You



여담으로
← 이 친구의 이름은
고퍼로 Gopher
(흙파는쥐류) 이다