

Golang для начинающих



Клестов Дима,
Golang-разработчик ООО «Инносети»

https://t.me/MaJloe_3Jlo
<https://github.com/MaJloe3Jlo>

Результаты занятия 1

```
func greeting(w http.ResponseWriter, r *http.Request) {
    name := r.URL.Query().Get("name")
    age := r.URL.Query().Get("age")
    fmt.Fprintf(w, "<h1>Hello! My name is %s. My age is %s</h1> ", name, age)
}
```

```
func greetings(w http.ResponseWriter, r *http.Request) {
    queryValues := r.URL.Query()
    name, age := queryValues.Get("name"), queryValues.Get("age")

    response := fmt.Sprintf("<h1>Hello! My name is %s.</h1> <h2>My age is %s years.</h2>", name, age)
    w.Write([]byte(response))
}
```

```
func handler(w http.ResponseWriter, r *http.Request) {
    q := r.URL.Query()
    age := q.Get("age")
    if age == "" {
        age = "unknown"
    }
    fmt.Fprintf(w, "Hello %s You age is %s", q.Get("name"), age)
}
```

```
func greetings(w http.ResponseWriter, r *http.Request) {
    name := r.URL.Query().Get("name")
    age := r.URL.Query().Get("age")

    if name == "" && age == "" {
        fmt.Fprintf(w, "<h1>Введите параметры name и age</h1>")
    } else if age == "" {
        fmt.Fprintf(w, "<h1>Hello! My name is %s.", name)
    } else if name == "" && age != "" {
        fmt.Fprintf(w, "<h1>Hello! I am %s years old.</h1>", age)
    } else {
        fmt.Fprintf(w, "<h1>Hello! My name is %s. I am %s years old.</h1>", name, age)
    }
}
```

```
func greetings(w http.ResponseWriter, r *http.Request) {
    params := r.URL.Query()
    var ret = "Hello!"
    var _name, namePresent = params["name"]
    var _age, agePresent = params["age"]
    if namePresent {
        ret = ret + " My name is " + strings.Join(_name, " ")
    }
    if agePresent {
        ret = ret + " My age is " + strings.Join(_age, " ")
    }
    fmt.Fprint(w, "<H1>"+ret+ "</H1>")
}
```

<https://play.golang.org/>

The Go Playground

Run

Format

☐ Imports

Share

Hello, playground

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 func main() {
8     fmt.Println("Hello, playground")
9 }
10
```

На прошлом занятии:

- 1) Общая информация о Go
- 2) Первая программа на Go
- 3) Домашнее задание



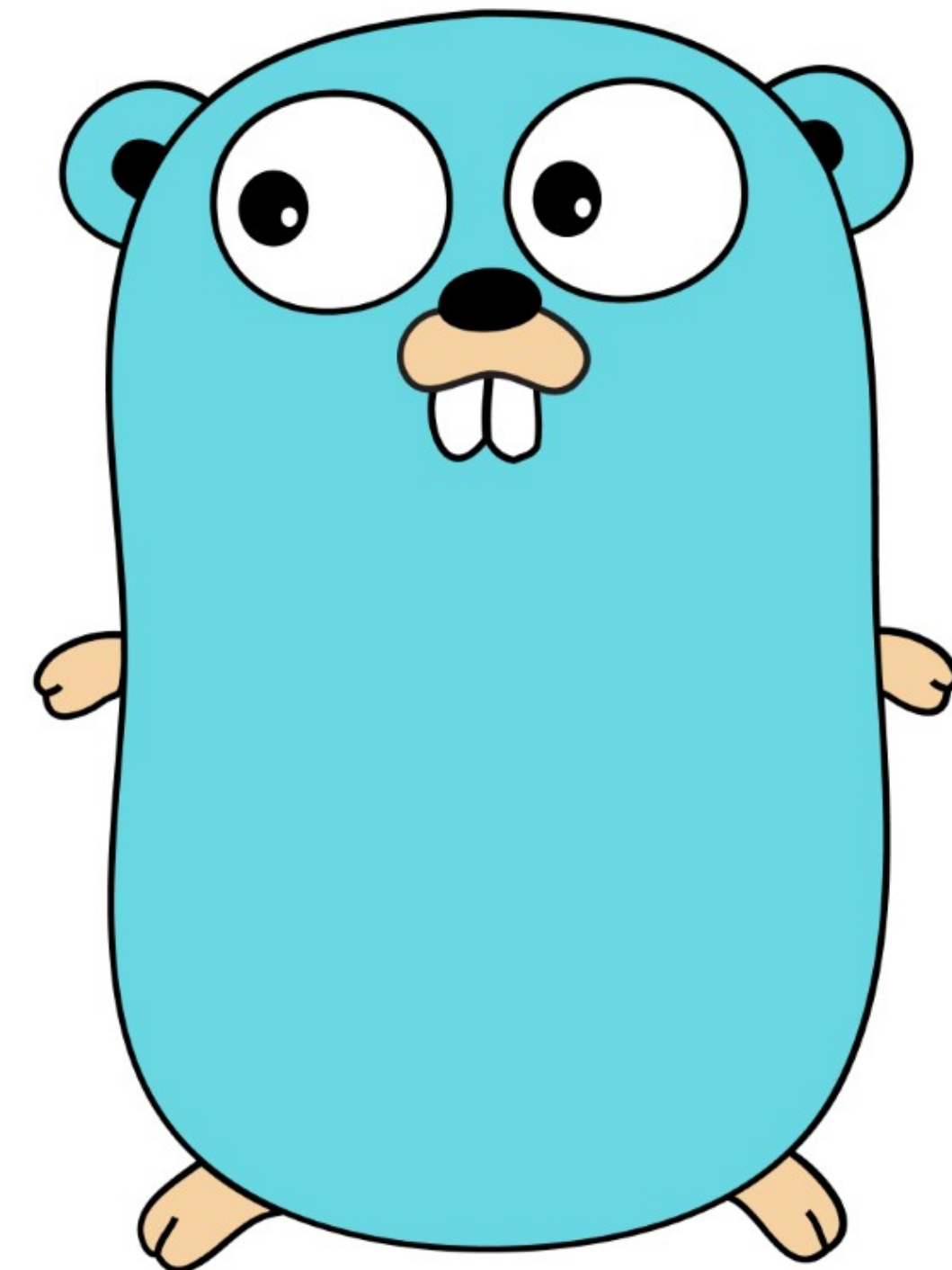
План занятия на сегодня:

- 1) Типы данных в Go
- 2) Переменные, константы и операции с ними
- 3) Области видимости в Go
- 4) Домашняя работа

Статическая типизация - все переменные имеют определенный тип, и этот тип не может быть изменен.

Встроенные типы:

- ✓ Числа
- ✓ Строки
- ✓ Логический



Типы данных. Числа

Числа:

- ✓ Целочисленные
- ✓ С плавающей точкой

Целые числа — это числа, которые не имеют вещественной части: 1, 2, 3.. и до бесконечности.

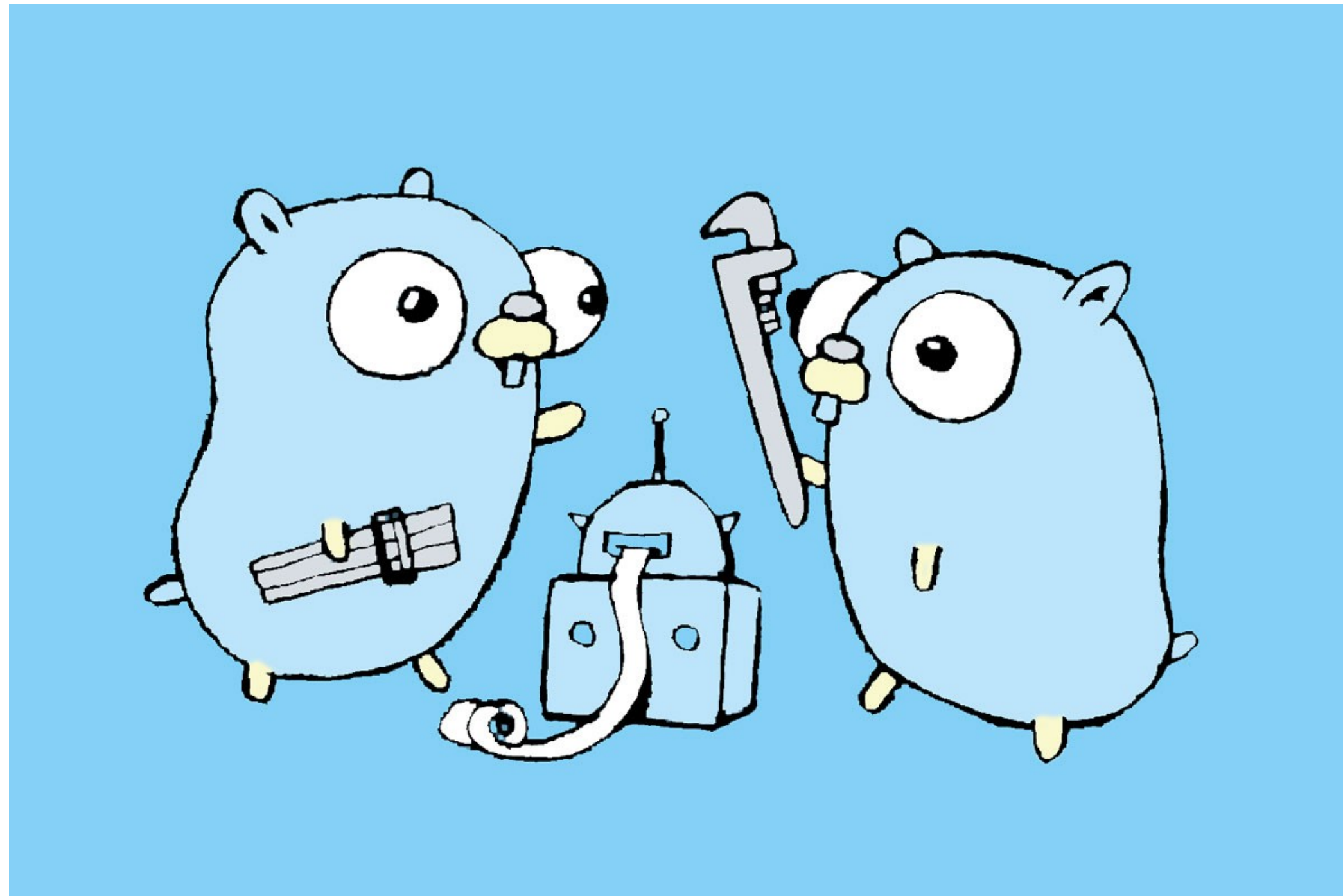
- ✓ Uint означает «unsigned integer» (беззнаковое целое)
- ✓ Int означает «signed integer» (знаковое целое)

Также существует 3 машинно-зависимых целочисленных типа: uint, int и uintptr.

- ✓ Просто используйте тип int или uint.

Целочисленные типы данных	
int8	[-128..127] в памяти: 1 байт (8 бит)
int16	[-32768..32767] в памяти: 2 байта (16 бит)
int32	[-2147483648..2147483647] в памяти: 4 байта (32 бита)
int64	[-9 223 372 036 854 775 808 .. 9 223 372 036 854 775 807] в памяти: 8 байт (64 бита)
uint8	[0..255], в памяти: 1 байт
uint16	[0..65535], в памяти: 2 байта
uint32	[0..4294967295], в памяти: 4 байта
uint64	[0..18446744073709551615], в памяти: 8 байт
byte (синоним uint8)	[0..255], в памяти: 1 байт
rune (синоним int32)	[-2147483648..2147483647], в памяти: 4 байта

Числа с плавающей точкой и операции



Числа с плавающей точкой — это числа, которые содержат вещественную часть (вещественные числа) (1.234, 123.4, 0.00001234)

Запомнить:

- ✓ Числа с плавающей точкой неточны
- ✓ Числа с плавающей точкой имеют определенный размер (32 бита или 64 бита)

Виды типов чисел с плавающей точкой:

- ✓ `float32` и `float64` - вещественные числа с одинарной и двойной точностью
- ✓ `complex64` и `complex128` - комплексные числа (числа с мнимой частью)
- ✓ Если Вы хотите работать с более точными числами используйте `float64`

Операции с числами

+	сложение	$1.0 + 1.0 = 2$
-	вычитание	$2 - 3 = -1$
*	умножение	$4 * 5 = 20$
/	деление	$9 / 3 = 3$
%	остаток от деления	$20 \% 3 = 2$

Типы данных. Строки

Строка — это последовательность символов определенной длины, используемая для представления текста.

- В Go строки можно создать с помощью:
- Двойных кавычек ("Hello world") - не могут содержать новые строки, но поддерживают \n — новая строка, \t — табуляция.
 - Апострофов(`Hello world`) - могут содержать новые строки.



Операции со строками		
len	Количество занимаемых байт	len("Hello") = 5
	Доступ к символу по индексу	"Hello"[1] = "e"
+	Конкатенация (сложение)	"Hello " + "world " = "Hello world"

bool

Булевый тип - bool - это специальный тип, используемый для представления истинности и ложности.

Переменная типа bool может принимать только два значения: true или false.

Логические операторы		
&&	И	true && false = false
	ИЛИ	true false = true
!	НЕ	!true = false

Переменные

Переменная - это именованный участок памяти в компьютере, который хранит некоторое значение (данные).

Полный вид:

var number int = 32

служебное слово | имя переменной | тип данных | начальное значение

- Имя переменной может представлять собой любой набор алфавитных, числовых символов и знака _;
- Первый символ обязательно алфавитный символ или _.

Служебные слова, которые нельзя использовать в качестве имен переменных: break, case, chan, const, continue, default, defer, else, fallthrough, for, func, go, goto, if, import, interface, map, package, range, return, select, struct, switch, type, var

Без начального значения:

```
var word string
```

,начальное значение будет задано автоматически

Короткая форма:

```
check := true
```

,тип будет определен автоматически (bool)

Без типа :

```
var myCar = «Audi A7»
```

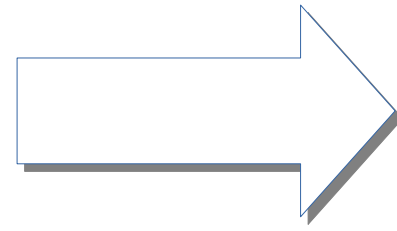
,тип будет определен автоматически (string)



Переменные

Объединение объявления:

```
var x int  
var y int  
var z int
```



```
var x, y, z int
```

Блок объявления:

```
var (  
    name string  
    age int = 31  
    old bool  
    weight = 90  
    voice string = «Hello»  
)
```

Массовое определение:

```
a, b, c := 1, 3, 5
```

Изменение значения
объявленной переменной:

```
var a int  
a = 10
```



Константы - хранят некоторые данные, значения которых нельзя изменить.

- ✓ Уменьшается количество работы во время компиляции
- ✓ Упрощается поиск ошибок

const number int = 32
служебное слово | имя константы | тип данных | значение

или

const number = 32
служебное слово | имя константы | значение

Константы

Блок объявления:

```
const (  
    a int = 45  
    b float32 = 3.3  
)
```

Дублирование значения константы:

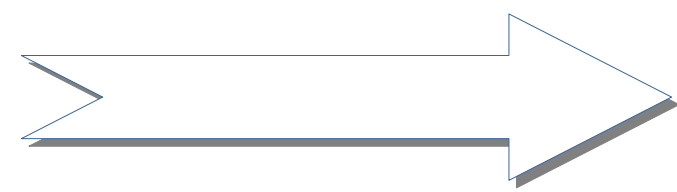
```
package main  
import "fmt"  
const (  
    A int = 46  
    B  
    C float32 = 3.1  
    D  
)  
func main() {  
    fmt.Println(A, B, C, D)  
    // Вывод: 46 46 3.1 3.1  
}
```



Идентификатор `iota`

```
const (  
    January = 1  
    February = 2  
    March = 3  
    April = 4  
    May = 5  
    June = 6  
    July = 7  
    August = 8  
    September = 9  
    October = 10  
    November = 11  
    December = 12  
)
```

```
fmt.Println(June)  
// Вывод: 6
```



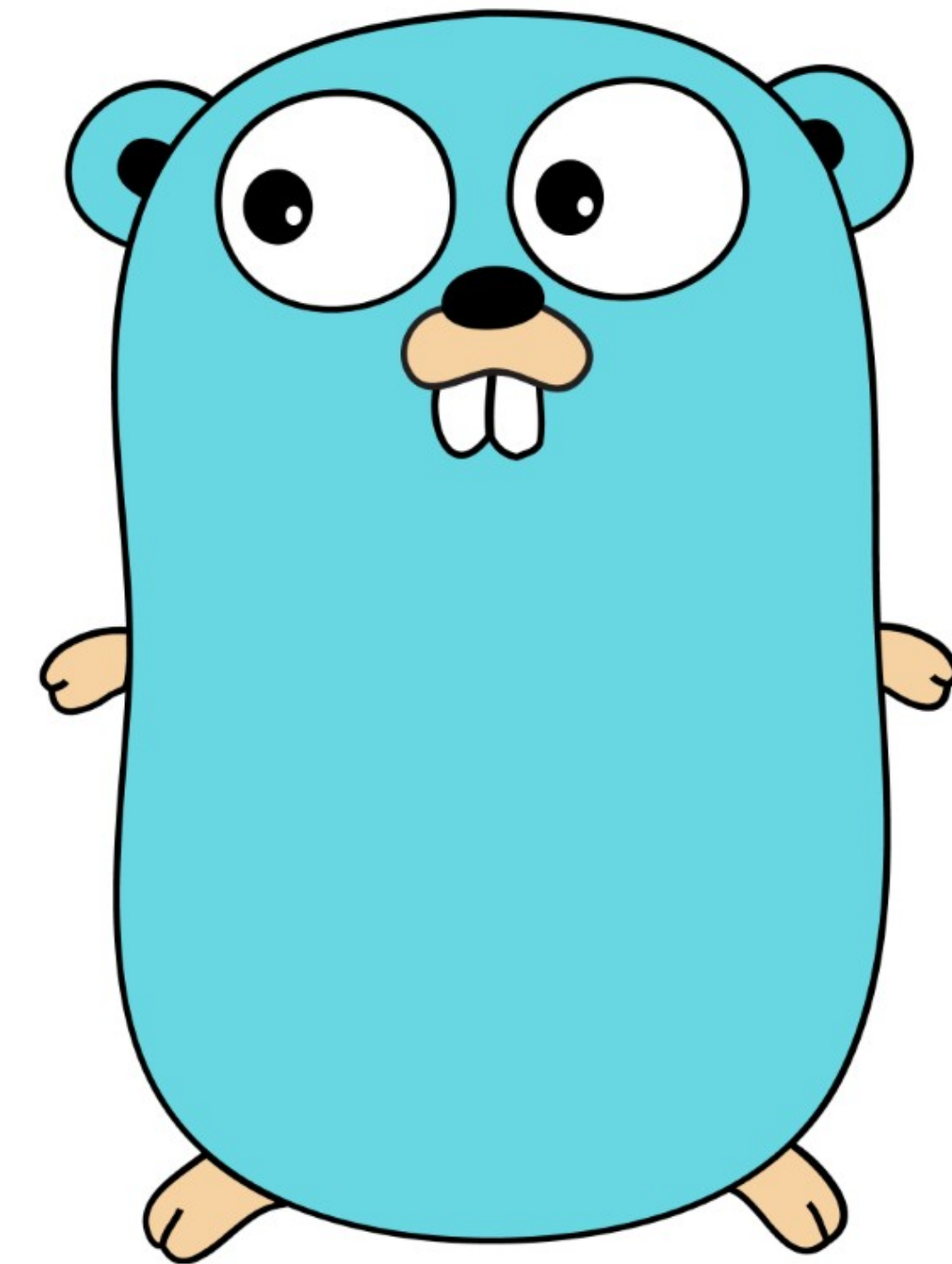
```
const (  
    January = 1 + iota  
    February  
    March  
    April  
    May  
    June  
    July  
    August  
    September  
    October  
    November  
    December  
)
```



Области видимости

Область видимости (англ. `scope`) в программировании — часть программы, в пределах которой идентификатор, объявленный как имя некоторой программной сущности (обычно — переменной, типа данных или функции), остаётся связанным с этой сущностью, то есть позволяет посредством себя обратиться к ней.

- Идентификатор объекта «виден» в определённом месте программы, если в данном месте по нему можно обратиться к данному объекту.
- За пределами области видимости тот же самый идентификатор может быть связан с другой переменной или функцией, либо быть свободным (не связанным ни с какой из них).
- Область видимости может, но не обязана совпадать с областью существования объекта, с которым связано имя.



Практическая часть

Задание здесь:
https://bit.ly/gostudy_2

На следующем занятии:

- 1) Управляющие конструкции (условия, конструкции, циклы)
- 2) Срезы, массивы, мапы



Благодарю за внимание!!!

