常量与变量

变量

标识符

标识符

在编程语言中标识符就是程序员定义的具有特殊意义的词，比如变量名、常量名、函数名等等。Go语言中标识符由字母数字和_（下划线）组成，并且只能以字母和_开头。举几个例子：`abc`，`_`，`_123`，`a123`。

关键字

关键字是指编程语言中预先定义好的具有特殊含义的标识符。关键字和保留字都不建议用作变量名。

Go语言中有25个关键字：

```
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
```

此外，Go语言中还有37个保留字。

```
Constants:    true  false  iota  nil

    Types:    int  int8  int16  int32  int64
              uint  uint8  uint16  uint32  uint64  uintptr
              float32  float64  complex128  complex64
              bool  byte  rune  string  error

Functions:   make  len  cap  new  append  copy  close  delete
```

go语言中的变量先声明再使用。

```go
package main

import (
    "fmt"
)

var id string = "2315925647" //对变量id进行声明 单独声明

var (
    name string = "wwy"
    age  int    = 20
    isOk bool   = true
)

func main() {
    fmt.Println(id, name, age, isOk)
}
```

注意：Go语言中声明的变量必须使用，否则就编译不过

运行结果

```
>4 go 设置调用>
2315925647 wwy 20 true

进程 已完成，退出代码为 0
```

```go
package main

import (
    "fmt"
)

var id string = "2315925647" //对变量id进行声明 单独声明

var (
    name string = "wwy"
    age  int    = 20
    isOk bool   = true
)

func main() {
    fmt.Println(id, name, age, isOk) //打印之后会有换行
    fmt.Printf("name:%s", name)      //占位符，使用name替换%s,没有换行
    fmt.Print(name, id)              //直接输出打印的内容没有换行
}
```

简短变量声明（只能在函数中使用）

```
s3 := "hahaha"
//简短变量声明
fmt.Println(s3)
```

简短变量声明只能在函数中使用，不能在全局中使用

匿名变量

```
func foo() (int, string) {
    return 10, "wwy"
}
func main() {
    x, _ := foo()
    _, y := foo()
    fmt.Println("x=", x)
    fmt.Println("y=", y)
}
```

当有些变量不想要时，将不想要的变量赋值给匿名变量。匿名变量不分配内存。

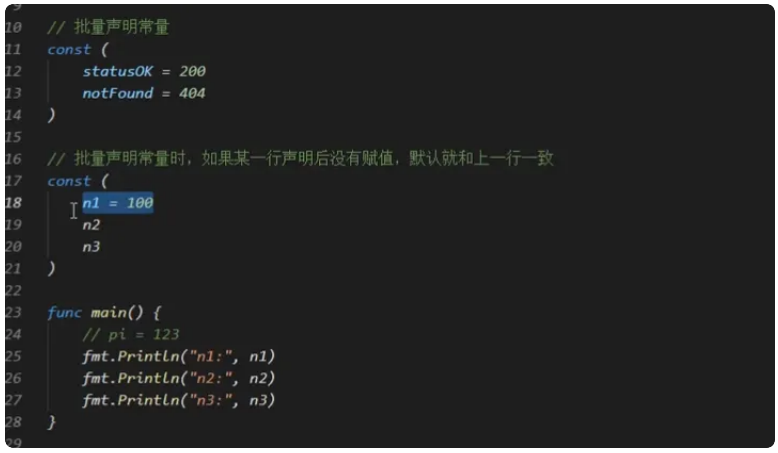
运行结果：

```
<4 go 设置调用>
x= 10
y= wwy

进程 已完成，退出代码为 0
```

常量（const）

```
const  pi =3.141592653589793   1 个用法

func main() {
    pi=123
}
```

这里pi是定义的一个常量，pi被定义后不能赋值。



```go
// 批量声明常量
const (
    statusOK = 200
    notFound = 404
)

// 批量声明常量时，如果某一行声明后没有赋值，默认就和上一行一致
const (
    n1 = 100
    n2
    n3
)

func main() {
    // pi = 123
    fmt.Println("n1:", n1)
    fmt.Println("n2:", n2)
    fmt.Println("n3:", n3)
}
```

iota（常量计数）

```go
const (
    n1 = iota
    n2
    n3
)

func main() {
    fmt.Println(n1, n2, n3)
}
```

运行结果:

```
<4 go 设置调用>
0 1 2

进程 已完成，退出代码为 0
```

当中间出现一个变量插队

```go
const ( //iota类似枚举
    n1 = iota
    n2=100
    n3=iota
    n4
    n5
)
```

```
0 100 2 3 4

进程 已完成，退出代码为 0
```

多个常量声明在同一行

```go
const ( //iota类似枚举
    d1, d2 = iota + 1, iota + 2
    d3, d4 = iota + 1, iota + 2
)

func main() {
    fmt.Println(d1, d2, d3, d4)
}
```

```
<4 go 设置调用>
1 2 2 3

进程 已完成，退出代码为 0
```

定义数量级

```go
const (
    _ = 0   = iota 无用法
    KB = 1024 = 1 << (10 * iota) 无用法
    MB = 1048576 = 1 << (10 * iota) 无用法
    GB = 1073741824 = 1 << (10 * iota) 无用法
    TB = 1099511627776 = 1 << (10 * iota) 无用法
    PB = 1125899906842624 = 1 << (10 * iota) 无用法
)
```