Go语言学习(二)常量

原创 2016年03月23日 21:17:37 572

package main

import "fmt"

//1.定义常量的关键字const

const Pi float64 = 3.14159265358979323846

const zero = 0.0 //无类型浮点常量

const(

size int64 = 1024

eof = -1 //无类型整型常量

)

const u,v float32 = 0 , 3 //u=0.0,v=3.0 常量的多重赋值

const a , b , c = 3, 4, "foo" //a=3,b=4,c="foo" 无类型整型和字符常量

// Go的常量定义可以限定常量类型，但不是必需的。如果定义常量时没有指定类型，那么它

// 与字面常量一样，是无类型常量。

// 常量定义的右值也可以是一个在编译期运算的常量表达式，比如

const mask = 1 << 3

//2.预定义常量

//Go语言预定了这些常量:true,false和iota

# //iota比较特殊,可以被认为是一个可被编译器修改的常量,在每一个const关键字出现时被重置为0,然后再下一个const常量出现之前,每次出现一次iota,其所代表的数字会自增1

const( //iota被重置为0

c01 = iota //c01 == 0

c02 = iota //c02 == 1

c03 = iota //c03 == 2

)

const( //iota被重置为0

a0 = 1 << iota //a0 == 1 (1乘以2的0次幂)

b1 = 1 << iota //b1 == 2 (1乘以2的1次幂)

c1 = 1 << iota //c1 == 4 (1乘以2的2次幂)

)

const( //iota被重置为0

u1 = iota \* 42 //u1 == 0

v1 float64 = iota \* 42 //v1 ==42.0

w1 = iota \* 42 //w1 ==84

)

const x1 = iota //x1==0 因为iota又被重置为0了

const y1 = iota //y1==0 同上

// 如果两个 const 的赋值语句的表达式是一样的，那么可以省略后一个赋值表达式。因此，上

// 面的前两个 const 语句可简写为：

const(

c00 = iota

c11

c22

)

const(

a12 = 1 << iota //a12==1 (iota在每个const开头被重置为0)

b12 //b12==2

c12 //c12==4

)

func main(){

//注意:常量的声明不能重复

fmt.Println("Pi=",Pi)

fmt.Println("zero=",zero)

fmt.Println("size=",size)

fmt.Println("eof=",eof)

fmt.Printf("u=%v,v=%v\n",u,v)

fmt.Printf("a=%v,b=%v,c=%v\n",a,b,c)

fmt.Printf("c01=%v,c02=%v,c03=%v\n",c01,c02,c03)

fmt.Printf("a0=%v,b1=%v,c1=%v\n",a0,b1,c1)

fmt.Printf("u1=%v,v1=%v,w1=%v\n",u1,v1,w1)

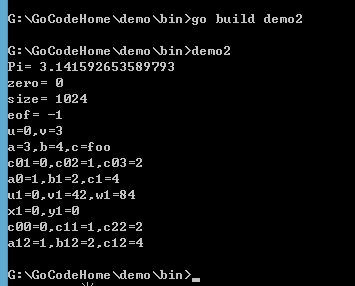
fmt.Printf("x1=%v,y1=%v\n",x1,y1)

fmt.Printf("c00=%v,c11=%v,c22=%v\n",c00,c11,c22)

fmt.Printf("a12=%v,b12=%v,c12=%v\n",a12,b12,c12)

}

运行结果：



文章来源：<http://blog.csdn.net/mchenys/article/details/50966716>

版权声明：本文为博主原创文章，未经博主允许不得转载。