计划使用go实现一个dhcp的server，因此学习总结一下go的基本语法和功能。

1：go程序的基本组成部分

|  |  |
| --- | --- |
| 包声明 | 在源文件的第一行指明当前文件属于哪个包。  每个go项目都有一个main包用于程序的入口  格式类似如下：packet main |
| 引入包 | 引入第三方或标准包，例如import “fmt” |
| 函数 | 定义格式如下：  func main() 其中func是关键字 |
| 变量 | 标识符或变量是大小字母开头，则可以被外部的包使用，小写则只能被所属包使用 |
| 表达式及代码语句 | 一行代表一个语句结束 |
| 注释 | 与C++的块注释相同 /\* \*/ |
| 标识符 | 用来给变量，函数等命名  1：以数字开头的标识符无效  2：关键字也无效  3：表达式也无效 |

2：go的关键字



3：go的数据类型

总共提供了布尔类型，数字类型，字符串类型，派生类型4大类。

|  |  |  |
| --- | --- | --- |
| 布尔类型 | bool | var b bool = true |
| 数字类型 | **uint8** | Var a uint8=10 |
|  | **uint16** | Var a uint16=11 |
|  | **uint32** |  |
|  | **uint64** |  |
|  | **int8** |  |
|  | **int16** |  |
|  | **int32** |  |
|  | **int64** |  |
|  | **float32** |  |
|  | **float64** |  |
|  | **complex64** 32 位实数和虚数 |  |
|  | **complex128** 64 位实数和虚数 |  |
|  | **byte** |  |
|  | **rune** |  |
|  | **uint 与CPU有关系** |  |
|  | **Int 与CPU有关系** |  |
|  | **uintptr 与CPU有关系** |  |
| 字符串类型 | Go的字符串是由单个字节连接起来的。Go语言的字符串的字节使用UTF-8编码标识Unicode文本 |  |
| 派生类型 | * (a) 指针类型（Pointer） * (b) 数组类型 * (c) 结构化类型(struct) * (d) Channel 类型 * (e) 函数类型 * (f) 切片类型 * (g) 接口类型（interface） * (h) Map 类型 |  |

Go的变量定义

格式：var identifier type

例如：var uint8 b=3;

第二种定义方式 var x=y

例如：var a=3 a的类型进行自动推断

第三种方式：x :=y

例如：a :=3 类型自动推断

多个变量的声明：

var name1,name2,name3 type

例如： var var1,var2,var3 uint8;

因式分解方式的声明方式：

Var (

Name1 type

Name2 type

) 这类方式一般用于定义全局信息

并行赋值，形式如下：a, b, c := 5, 7, "abc"

右边的值被依次赋值给左边的变量。

Go中的常量，常量只能是布尔，数字和字符型。

定义格式：const name type=xxx

例如：const b string = “123”

特殊常量iota, 是一个被编译器使用的常量，在const出现时，该值归位为0，否则每使用一次，该值增加1.

Go的函数定义

格式：

func function\_name( [parameter list] ) [return\_types] {

函数体

}

函数定义解析：

* func：函数由 func 开始声明
* function\_name：函数名称，函数名和参数列表一起构成了函数签名。
* parameter list：参数列表，参数就像一个占位符，当函数被调用时，你可以将值传递给参数，这个值被称为实际参数。参数列表指定的是参数类型、顺序、及参数个数。参数是可选的，也就是说函数也可以不包含参数。
* return\_types：返回类型，函数返回一列值。return\_types 是该列值的数据类型。有些功能不需要返回值，这种情况下 return\_types 不是必须的。
* 函数体：函数定义的代码集合。

例如：

func max(num1, num2 int) int {

/\* 声明局部变量 \*/

var result int

if (num1 > num2) {

result = num1

} else {

result = num2

}

return result

}