Golang map 详解

主讲教师：（大地）
合作网站：www.itying.com （IT 营）
我的专栏：https://www.itying.com/category-79-b0.html

1、map 的介绍

map 是一种无序的基于 key-value 的数据结构，Go 语言中的 map 是引用类型，必须初始化才能使用。

Go 语言中 map 的定义语法如下：

```
map[KeyType]ValueType
```

其中:

- KeyType:表示键的类型。
- ValueType:表示键对应的值的类型。

map 类型的变量默认初始值为 nil，需要使用 make()函数来分配内存。语法为：

make: 用于 slice，map，和 channel 的初始化

```
make(map[KeyType]ValueType, [cap])
```

其中 cap 表示 map 的容量，该参数虽然不是必须的。

注意：获取 map 的容量不能使用 cap, cap 返回的是数组切片分配的空间大小，根本不能用于 map。要获取 map 的容量，可以用 len 函数。

2、map 基本使用

map 中的数据都是成对出现的，map 的基本使用示例代码如下：

```
func main() {
    scoreMap := make(map[string]int, 8)
    scoreMap["张三"] = 90
    scoreMap["小明"] = 100
    fmt.Println(scoreMap)
    fmt.Println(scoreMap["小明"])
    fmt.Printf("type of a:%T\n", scoreMap)
}
```

输出：

```
map[小明:100 张三:90]
100
type of a:map[string]int
```

map 也支持在声明的时候填充元素，例如：

```
func main() {
    userInfo := map[string]string{
        "username": "IT 营小王子",
        "password": "123456",
    }
    fmt.Println(userInfo) //
}
```

3、判断某个键是否存在

Go 语言中有个判断 map 中键是否存在的特殊写法，格式如下:

```
value, ok := map 对象[key]
```

举个例子：

```
func main() {
    scoreMap := make(map[string]int)
    scoreMap["张三"] = 90
```

```
    scoreMap["小明"] = 100
    // 如果 key 存在 ok 为 true,v 为对应的值；不存在 ok 为 false,v 为值类型的零值
    v, ok := scoreMap["张三"]
    if ok {
        fmt.Println(v)
    } else {
        fmt.Println("查无此人")
    }
}
```

4、map 的遍历

Go 语言中使用 for range 遍历 map。

```
func main() {
    scoreMap := make(map[string]int)
    scoreMap["张三"] = 90
    scoreMap["小明"] = 100
    scoreMap["娜扎"] = 60
    for k, v := range scoreMap {
        fmt.Println(k, v)
    }
}
```

但我们只想遍历 key 的时候，可以按下面的写法：

```
func main() {
    scoreMap := make(map[string]int)
    scoreMap["张三"] = 90
    scoreMap["小明"] = 100
    scoreMap["娜扎"] = 60
    for k := range scoreMap {
        fmt.Println(k)
    }
}
```

注意： 遍历 map 时的元素顺序与添加键值对的顺序无关。

5、使用 delete()函数删除键值对

使用 delete()内建函数从 map 中删除一组键值对，delete()函数的格式如下：

delete(map 对象，key)

其中，

- map 对象:表示要删除键值对的 map 对象
- key:表示要删除的键值对的键

示例代码如下：

```
func main(){
    scoreMap := make(map[string]int)
    scoreMap["张三"] = 90
    scoreMap["小明"] = 100
    scoreMap["娜扎"] = 60
    delete(scoreMap, "小明")//将小明:100 从 map 中删除
    for k,v := range scoreMap{
        fmt.Println(k, v)
    }
}
```

6、【案例】按照指定顺序遍历 map

```
func main() {
    rand.Seed(time.Now().UnixNano()) //初始化随机数种子

    var scoreMap = make(map[string]int, 200)

    for i := 0; i < 100; i++ {
        key := fmt.Sprintf("stu%02d", i) //生成 stu 开头的字符串
        value := rand.Intn(100)          //生成 0~99 的随机整数
        scoreMap[key] = value
    }
    //取出 map 中的所有 key 存入切片 keys
    var keys = make([]string, 0, 200)
    for key := range scoreMap {
        keys = append(keys, key)
    }
    //对切片进行排序
```

```
    sort.Strings(keys)
    //按照排序后的 key 遍历 map
    for _, key := range keys {
        fmt.Println(key, scoreMap[key])
    }
}
```

7、元素为 map 类型的切片

下面的代码演示了切片中的元素为 map 类型时的操作：

```
func main() {
    var mapSlice = make([]map[string]string, 3)
    for index, value := range mapSlice {
        fmt.Printf("index:%d value:%v\n", index, value)
    }
    fmt.Println("after init")
    // 对切片中的 map 元素进行初始化
    mapSlice[0] = make(map[string]string, 10)
    mapSlice[0]["name"] = "小王子"
    mapSlice[0]["password"] = "123456"
    mapSlice[0]["address"] = "海淀区"
    for index, value := range mapSlice {
        fmt.Printf("index:%d value:%v\n", index, value)
    }
}
```

8、值为切片类型的 map

下面的代码演示了 map 中值为切片类型的操作：

```
func main() {
    var sliceMap = make(map[string][]string, 3)
    fmt.Println(sliceMap)
    fmt.Println("after init")
    key := "中国"
    value, ok := sliceMap[key]
    if !ok {
        value = make([]string, 0, 2)
    }
```

```
    value = append(value, "北京", "上海")
    sliceMap[key] = value
    fmt.Println(sliceMap)
}
```

9、练习题

写一个程序，统计一个字符串中每个单词出现的次数。比如：”how do you do”中 how=1 do=2 you=1。

```
    var wordMap = make(map[string]int)
    var str = "how do you do"
    var arrSlice = strings.Split(str, " ")
    for _, word := range arrSlice {
        wordMap[word]++
    }
    fmt.Println(wordMap)
```