Go 语言从入门到项目实战

第7章 接口

7.1 接口概述

- 接口实际上也是一种数据类型，它更为抽象，它把所有的具有共性的方法定义在一起，这些方法只有函数声明，没有具体的函数体，任何其他类型只要实现了接口中定义好的这些方法，那么就说这个类型实现（Implement）了这个接口；
- 这种只做函数声明的做法为任何实现该接口的类型定义了“行为规范”，接口更关心“行为”，实现接口的类型要遵循这些预先定义好的“行为”，并结合自身的特点实现它们。

7.2 接口的使用

- 声明格式

```
type interface_name interface{
        function_name(params) return_types
        function_name(params) return_types
        function_name(params) return_types
        ...
}
```

7.2 接口的使用

- 示例

```
//叫动作接口，几乎所有动物都应实现该接口
type Sayer interface {
    //叫动作
    Say()
}
```

```
//定义猫类型
type Cat struct{}
//猫的动作
func (catInstance Cat) Say() string { return "喵喵喵" }
//定义狗类型
type Dog struct{}
//狗的动作
func (dogInstance Dog) Say() string { return "汪汪汪
" }
```

7.2 接口的使用

- 示例

```
var anyAnimalSayer Sayer
c := Cat{}
anyAnimalSayer=c
anyAnimalSayer.say()
d := Dog{}
anyAnimalSayer=d
anyAnimalSayer.say()
```

7.2 接口的使用

- 接口的嵌套

```
type Sayer interface {
        say()
}
type Runner interface {
        run()
}
type Sleeper interface {
        sleep()
}
```

```
//动作接口，所有动物都应实现该接口
type Action interface {
        Sayer
        Runner
        Sleeper
}
```

7.2 接口的使用

- 接口的嵌套

```
type Cat struct{}
func (catInstance Cat) say() {
        fmt.Println("喵喵喵")
}
func (catInstance Cat) run() {
        fmt.Println("奔跑中")
}
func (catInstance Cat) sleep() {
        fmt.Println("睡眠中")
}
```

```
func main() {
        var anyAnimalActions Action
        catOne := Cat{}
        anyAnimalActions=&catOne
        anyAnimalActions.say()
        anyAnimalActions.run()
        anyAnimalActions.sleep()
}
```

7.3 空接口

- 实现某些其它编程语言中的“范型”。
- 声明格式：

```
type EmptyInterface interface {}
```

7.3 空接口

- 示例

```
type EmptyInterface interface {}
func main() {
        var emptyInterface EmptyInterface
        strVar:="我是三酷猫"
        emptyInterface= strVar numVar:=18
        emptyInterface= numVar     boolVar:=true
        emptyInterface=boolVar}
```

7.3 空接口

- 示例

```
type EmptyInterface interface {
}
func main() {
        var animalInfo=make(map[string]EmptyInterface)
        animalInfo["name"]="三酷猫"
        animalInfo["age"]=3
        animalInfo["married"]=false
        fmt.Println(animalInfo)
}
```

7.4 类型断言

- 格式

  x.(T)

- 示例

```
type EmptyInterface interface {}
func main() {
        var emptyInterface EmptyInterface
        str:="我是三酷猫"
        emptyInterface=str
        val, boolVal :=emptyInterface.(string)
if boolVal {
                fmt.Println("emptyInterface保存了字符串类型数据：",val)
        }else{
                fmt.Println("emptyInterface保存的不是字符串类型数据。")
        }
}
```