布尔型

布尔型的值只可以是常量 true 或者 false。

两个类型相同的值可以使用相等 `==` 或者不等 `!=` 运算符来进行比较并获得一个布尔型的值。

当相等运算符两边的值是完全相同的值的时候会返回 true，否则返回 false，并且只有在两个的值的类型相同的情况下才可以使用。

布尔型常用在**条件判断语句**，或者**循环语句**。也可以用在**逻辑表达式**中。

注意：

- 布尔类型变量的**默认值为false**。
- Go 语言中不允许将整型强制转换为布尔型。
- 布尔型无法参与数值运算，也无法与其他类型进行转换。
- Go语言中不能用0和非0表示真假

一个简单的例子：`var b bool = true`。

条件判断示例：

```
package main

import "fmt"

func main() {
    //满分100，60分为及格
    score := 50
    if score >= 60 {    //此处50分不大于60分所以是假FALSE
        fmt.Println("恭喜您及格了！")
    } else if score > 100 {
        fmt.Println("您老越界了！")
    } else {
        fmt.Println("哎呀！咋考的！")
    }
}

#结果
哎呀！咋考的！
```

循环语句示例：

```
package main

import "fmt"

func main() {
    total := 10
    for i := 0; i < total; i++ { //此处 i<total 当i大于等于10时为FALSE
        fmt.Printf("i: %v\n", i)
    }
}
```

```
#结果
i: 0
i: 1
i: 2
i: 3
i: 4
i: 5
i: 6
i: 7
i: 8
i: 9
```

逻辑表达式示例：

```
package main

import "fmt"

func main() {
    age := 18
    gender := "男"

    if age >= 18 && gender == "男" { //此处&&和 两侧都为真才是TRUE，只要有一个判断为假就是FALSE
        fmt.Println("您已是成年男子")
    }

    if age >= 18 || gender == "男" { //此处||和 一侧为真就是是TRUE，两侧都为假就是FALSE
        fmt.Println("您已成年")
    }

    var a bool = true
    var b bool = false
    if !(a && b) { //逻辑! NOT运算符。如果条件为TRUE,则逻辑NOT条件FALSE，否则为TRUE
        fmt.Printf("条件为 true\n")
    }
}

#结果
您已是成年男子
您已成年
条件为 true
```