



Go语言入门到精通

江洲老师云课堂

—— 主讲：江洲老师 ——

文件的写入

目录



PART 01
文件的创建



PART 02
WriteString()
方法



PART 03
Write()方法



PART 04
WriteAt()方法



文件的创建

1.文件的创建

将数据存储到文件之前，先要创建文件，GO语言中提供了一个**Create()函数专门创建文件。**

该函数在创建文件时，首先会判断要创建的文件是否存在，如果不存在，则创建，如果存在，会先将文件中已有的数据清空。

同时，当文件创建成功后，该文件会默认打开，所以不用在执行打开操作，可以直接向该文件中写入数据。

创建文件的步骤：

- (1)导入“os”包，创建文件，读写文件的函数都在os包
- (2)指定创建的文件存放路径以及文件名
- (3)执行Create()函数，进行文件创建
- (4)关闭文件



文件的创建

```
import "os"

func CreateFile(path string) {
    //创建文件，返回两个值，一是创建的文件，二是错误信息
    f, err := os.Create(path)
    if err != nil { //如果有错误，打印错误，同时返回
        fmt.Println("err=", err)
        return
    }
    defer f.Close() //在退出整个函数时，关闭文件
}

func main() {
    var filePath = "a.txt"
    CreateFile(filePath)
}
```

执行以上代码后，可以在程序文件存放的目录中，看到有一个a.txt的文件。
注意：在创建的文件时，注意需要判断是否出现异常，同时要注意defer的应用



WriteString()方法


1.WriteString()方法



WriteString()方法

文件打开以后，向文件中写数据，可以使用WriteString()方法。

```
func CreateFile(path string) {  
    //创建文件，返回两个值，一是创建的文件，二是错误信息  
    f, err := os.Create(path)  
    if err != nil { //如果有错误，打印错误，同时返回  
        fmt.Println("err=", err)  
        return  
    }  
    for i := 1; i < 10; i++ {  
        f.WriteString("Hello World")  
    }  
    defer f.Close() //在退出整个函数时，关闭文件  
}
```



写入文件

WriteString()方法默认返回两个参数，第一个参数，指的是写入文件的数据长度，第二个参数记录的是错误信息。

WriteString()方法默认写到文件中的数据是不换行的。



Write()方法

1.Write()方法



Write()方法

除了使用WriteString()函数向文件中写入数据意外，还可以使用Write()函数，如下所示：

```
func CreateFile(path string) {  
    //创建文件，返回两个值，一是创建的文件，二是错误信息  
    f, err := os.Create(path)  
    if err != nil { //如果有错误，打印错误，同时返回  
        fmt.Println("err=", err)  
        return  
    }  
    var str string  
    for i := 1; i < 10; i++ {  
        // "i = 1\n", 这个字符串存储在str中  
        str = fmt.Sprintf("i = %d\n", i+1)  
        buf := []byte(str)  
        n, err := f.Write(buf)  
        if err != nil {  
            fmt.Println(err)  
        }  
        fmt.Println(n)  
    }  
    defer f.Close() //在退出整个函数时，关闭文件  
}
```

注意的是，使用Write()函数写数据时，参数为字节切片，所以需要将字符串转换成字节切片。该方法返回的也是写入文件数据的长度。



WriteAt()方法

1. WriteAt()方法

WriteAt()方法

第三种写入的方式是使用WriteAt()函数，在指定的位置写入数据。
如下所示：

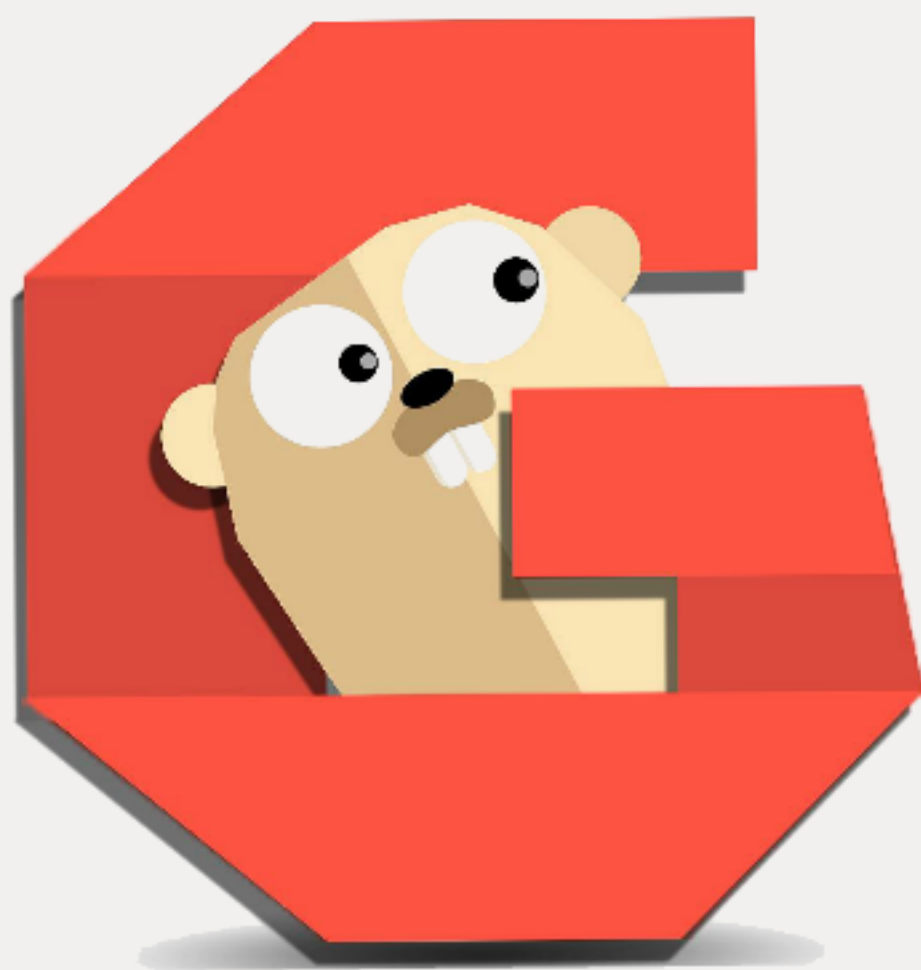
```
func CreateFile(path string) {  
    //创建文件，返回两个值，一是创建的文件，二是错误信息  
    f, err := os.Create(path)  
    if err != nil { //如果有错误，打印错误，同时返回  
        fmt.Println("err=", err)  
        return  
    }  
    var str string  
    var a int  
    for i := 0; i < 10; i++ {  
        // "i = 1\n", 这个字符串存储在str中  
        str = fmt.Sprintf("i = %d\n", i+1)  
        buf := []byte(str)  
        // 查找文件末尾的偏移量  
        n, _ := f.Seek(0, os.SEEK_END)  
  
        // 从末尾的偏移量开始写入内容  
        a, err = f.WriteAt([]byte(buf), n)  
        fmt.Println(a)  
    }  
    defer f.Close() //在退出整个函数时，关闭文件  
}
```

以上程序中Seek()函数返回值存储到变量n中，值为文件末尾的位置。WriteAt()也返回的是写入的数据长度。



实例演示





Go语言入门到精通

江洲老师云课堂

—— 主讲：江洲老师 ——

感谢您的聆听和观看