Go语言入门到精通

江洲老师云课堂

主讲：江洲老师

文件的读取

目 录

Read 读取文件

1.Read 读取文件

Read 读取文件

如果文件已经存在，并且也已经有数据了，那么可以直接读取该文件中的内容。

读取文件的基本流程如下：

（1）打开要读取的文件

（2）对文件进行读取

（3）关闭文件

在向文件中写数据的时候，使用的是Write,那么读取文件中的数据，使用的是Read。

关于Read( )函数的使用如下：

```go
func ReadFile(filePath string) {
    //打开文件
    f, err := os.Open(filePath)
    if err != nil {
        fmt.Println("err = ", err)
        return
    }
    //关闭文件
    defer f.Close()
    buf := make([]byte, 1024*2) //2k大小
    //n代表从文件读取内容的长度
    n, err1 := f.Read(buf)
    if err1 != nil && err1 != io.EOF { //文件出错，同时没有到结尾
        fmt.Println("err1 = ", err1)
        return
    }

    fmt.Println("buf = ", string(buf[:n]))
}
```

Read 读取文件

Open( )是打开文件，与OpenFile( )的区别是，Open( )只有读的权限。

在使用Read( )函数读取文件中的内容时，需要一个切片类型，而定义切片时类型为字符数组，将文件中的内容保存在切片中，同时除了对其判断是否出错时以外，还要判断是否到文件末尾(这里需要导入io包)。

Read( )函数返回的是从文件中读取的数据的长度，最后，输出切片中存储的文件数据。

注意，读取的是从最开始到整个数据长度，因为有可能存储到切片中的数据达不到切片的总长度（也就是切片是2k,但是从文件中读取的数据有可能只有1k）

读取步骤

上面我们是将文件的内容全部读取出来，然后存放在切片中，我们也可以每次只读取一行数据。
这需要用到bufio包中的ReadBytes函数。具体如下：

1：打开文件

```
//打开文件
f, err := os.Open(path)
if err != nil {
    fmt.Println("err = ", err)
    return
}
```

2：创建缓冲区

在使用ReadBytes( )函数读取数据时，需要用到缓冲区,所谓缓冲区就是存储数据的区域，也就是先将从文件中读取的数据存储在该区域内，然后在将区域中的数据取出来，写到磁盘上。提供缓冲区的原因是：
为了缓和 CPU 与 磁盘设备之间速度不匹配矛盾。文件缓冲区是用以暂时存放读写期间的文件数据而在内存区预留的一定空间。

3：循环读取文件中的内容，直到文件末尾位置。

```
buf := make([]byte, 4*1024) //4k大小的缓冲
    n, err := file.Read(buf)
    if err != nil && err != io.EOF { //EOF是文件 结束标志位
        fmt.Println("文件读取错误！", err)
        return err
    }
```

4：最后关闭文件

```
//关闭文件
defer f.Close()
```

实例演示

江洲老师云课堂

主讲：江洲老师

感谢您的聆听和观看