flipped-aurora / gin-vue-admin Public

Code Issues 23 Pull requests 2 Discussions Actions Projects ···

main

gin-vue-admin / server / utils / breakpoint_continue.go / <> Jump to

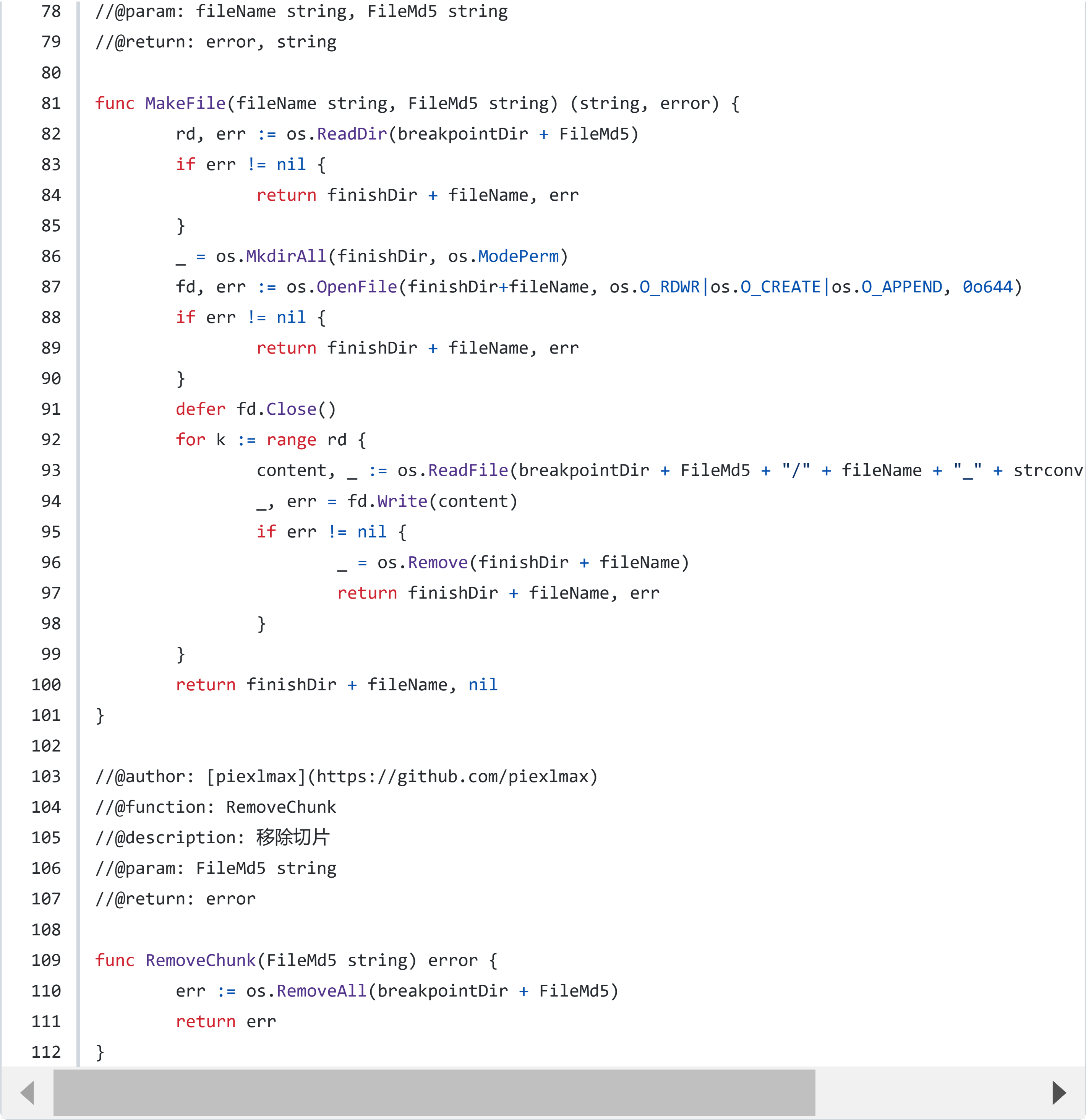
piexlmax 增加自动化代码匹配进入业务库功能 (#1249) ... ✓ History

6 contributors

112 lines (99 sloc) | 3.03 KB

```go
package utils

import (
	"errors"
	"os"
	"strconv"
	"strings"
)

// 前端传来文件片与当前片为什么文件的第几片
// 后端拿到以后比较次分片是否上传 或者是否为不完全片
// 前端发送每片多大
// 前端告知是否为最后一片且是否完成

const (
	breakpointDir = "./breakpointDir/"
	finishDir     = "./fileDir/"
)

//@author: [piexlmax](https://github.com/piexlmax)
//@function: BreakPointContinue
//@description: 断点续传
//@param: content []byte, fileName string, contentNumber int, contentTotal int, fileMd5 string
//@return: error, string

func BreakPointContinue(content []byte, fileName string, contentNumber int, contentTotal int, file
	path := breakpointDir + fileMd5 + "/"
	err := os.MkdirAll(path, os.ModePerm)
```

```go
        if err != nil {
                return path, err
        }
        pathC, err := makeFileContent(content, fileName, path, contentNumber)
        return pathC, err
}

//@author: [piexlmax](https://github.com/piexlmax)
//@function: CheckMd5
//@description: 检查Md5
//@param: content []byte, chunkMd5 string
//@return: CanUpload bool

func CheckMd5(content []byte, chunkMd5 string) (CanUpload bool) {
        fileMd5 := MD5V(content)
        if fileMd5 == chunkMd5 {
                return true // 可以继续上传
        } else {
                return false // 切片不完整，废弃
        }
}

//@author: [piexlmax](https://github.com/piexlmax)
//@function: makeFileContent
//@description: 创建切片内容
//@param: content []byte, fileName string, FileDir string, contentNumber int
//@return: string, error

func makeFileContent(content []byte, fileName string, FileDir string, contentNumber int) (string,
        if strings.Index(fileName, "./") > -1 || strings.Index(FileDir, "./") > -1 {
                return "", errors.New("文件名或路径不合法")
        }
        path := FileDir + fileName + "_" + strconv.Itoa(contentNumber)
        f, err := os.Create(path)
        if err != nil {
                return path, err
        } else {
                _, err = f.Write(content)
                if err != nil {
                        return path, err
                }
        }
        defer f.Close()
        return path, nil
}

//@author: [piexlmax](https://github.com/piexlmax)
//@function: makeFileContent
//@description: 创建切片文件
```



```go
//@param: fileName string, FileMd5 string
//@return: error, string

func MakeFile(fileName string, FileMd5 string) (string, error) {
        rd, err := os.ReadDir(breakpointDir + FileMd5)
        if err != nil {
                return finishDir + fileName, err
        }
        _ = os.MkdirAll(finishDir, os.ModePerm)
        fd, err := os.OpenFile(finishDir+fileName, os.O_RDWR|os.O_CREATE|os.O_APPEND, 0o644)
        if err != nil {
                return finishDir + fileName, err
        }
        defer fd.Close()
        for k := range rd {
                content, _ := os.ReadFile(breakpointDir + FileMd5 + "/" + fileName + "_" + strconv
                _, err = fd.Write(content)
                if err != nil {
                        _ = os.Remove(finishDir + fileName)
                        return finishDir + fileName, err
                }
        }
        return finishDir + fileName, nil
}

//@author: [piexlmax](https://github.com/piexlmax)
//@function: RemoveChunk
//@description: 移除切片
//@param: FileMd5 string
//@return: error

func RemoveChunk(FileMd5 string) error {
        err := os.RemoveAll(breakpointDir + FileMd5)
        return err
}
```