大厂面试课之Go语言

第33课：上传文件

同学们好！在上节课，我们完成了基于FastDFS和Nginx的分布式文件存储服务器的搭建，并通过FastDFS客户机的上传文件命令，成功地将一副本地图片上传至FastDFS服务器。那么，我们如何通过Go语言程序实现类似的文件上传功能呢？这就是这节课我们要解决的问题。

安装FastDFS客户端SDK

在Go程序中上传文件到FastDFS

我们首先需要安装FastDFS客户机面向Go语言的开发工具包，即SDK，然后编写一个基于Go语言的测试程序，将一副本地图片上传至FastDFS服务器。

安装FastDFS客户端SDK

我们先来安装FastDFS客户端SDK。

知识讲解

安装fdfs_client

- https://github.com/tedcy/fdfs_client
  fdfs_client-master.zip
  GOPATH\src\github.com\tedcy\fdfs_client

我们可以从GitHub上下载FastDFS客户端SDK，并解压到正确的路径下。

配置fdfs_client

知识讲解

- 将GOPATH\src\github.com\tedcy\fdfs_client目录下的fdfs.conf文件复制到GOPATH\src\iHome\front-end\conf目录下，编辑修改之：

```
tracker_server=192.168.0.111:22122
maxConns=100
```

- 如果使用Windows操作系统，该文件必须使用PC（DOS）格式保存

之后将FastDFS客户机配置文件，复制到front-end工程目录下的conf子目录中，并对该文件做一些修改，指定最大连接数和跟踪服务器的IP地址及端口。注意，如果我们使用的是Windows操作系统，该文件必须保存为DOS格式。

在Go程序中上传文件到FastDFS

下面我们将尝试使用Go语言，编写一个上传文件到FastDFS的测试程序。

GOPATH/src/iHome/front-end/test/fastdfs.go

项目实战

```
package main

import (
    "fmt"
    "github.com/tedcy/fdfs_client"
    "io"
    "os"
)

func main() {
    // 创建FastDFS客户机
    client, err := fdfs_client.NewClientWithConfig("../conf/fdfs.conf")
    defer client.Destory()
    if err != nil {
```


GOPATH/src/iHome/front-end/test/fastdfs.go

项目实战

```
        fmt.Println("fdfs_client.NewClientWithConfig错误:", err)
        return
    }

    // 打开文件
    file, err := os.Open("../../user-avatars/user01.png")
    defer file.Close()
    if err != nil {
        fmt.Println("os.Open错误:", err)
        return
    }

    // 读取文件内容
    content, err := io.ReadAll(file)
```


GOPATH/src/iHome/front-end/test/fastdfs.go

项目实战

```
    if err != nil {
        fmt.Println("io.ReadAll错误:", err)
        return
    }

    // 上传数据到FastDFS
    fileid, err := client.UploadByBuffer(content, "png")
    if err != nil {
        fmt.Println("Client.UploadByBuffer错误:", err)
        return
    }

    fmt.Println(fileid)
}
```


我们在front-end工程目录下的test子目录中创建一个名为fastdfs.go的文件：

```go
package main

import (
    "fmt"
    "github.com/tedcy/fdfs_client"
    "io"
    "os"
)

func main() {
    // 创建FastDFS客户机
    client, err := fdfs_client.NewClientWithConfig("../conf/fdfs.conf")
    defer client.Destory()
    if err != nil {
        fmt.Println("fdfs_client.NewClientWithConfig错误:", err)
        return
    }

    // 打开文件
    file, err := os.Open("../../user-avatars/user01.png")
    defer file.Close()
    if err != nil {
        fmt.Println("os.Open错误:", err)
        return
    }

    // 读取文件内容
    content, err := io.ReadAll(file)
    if err != nil {
        fmt.Println("io.ReadAll错误:", err)
        return
    }

    // 上传数据到FastDFS
    fileid, err := client.UploadByBuffer(content, "png")
    if err != nil {
        fmt.Println("Client.UploadByBuffer错误:", err)
        return
    }

    fmt.Println(fileid)
}
```

我们首先以FastDFS客户机配置文件的路径为参数，调用FastDFS客户端SDK的NewClientWithConfig函数，创建一个FastDFS客户机对象。打开本地图片文件，并读取其中的图像数据，传给FastDFS客户机对象的UploadByBuffer方法，同时指定文件的扩展名。如果不发生错误的话，我们的图片已经上传到FastDFS服务器中了。FastDFS客户机对象的UploadByBuffer方法成功返回字符串形式的文件凭证，我们将其打印出来。

项目实战

在虚拟机中启动FastDFS和Nginx

```
$ sudo fdfs_trackerd /etc/fdfs/tracker.conf
$ sudo fdfs_storaged /etc/fdfs/storage.conf
$ sudo /usr/local/nginx/sbin/nginx
```

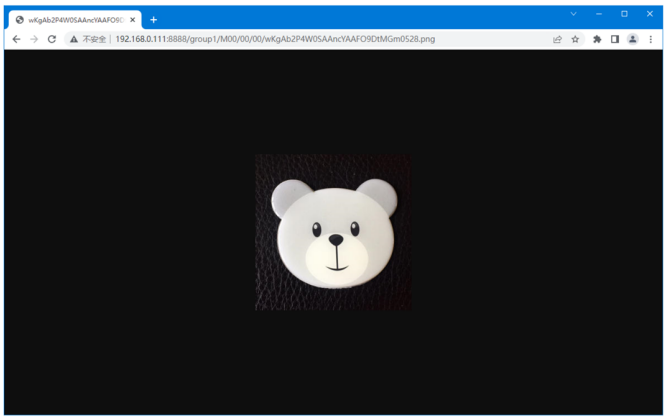
运行测试程序，输出文件凭证

```
> go run fastdfs.go
group1/M00/00/00/wKgAb2P4W0SAAncYAAFO9DtMGm0528.png
```


在VMware中启动Ubuntu虚拟机，在Ubuntu虚拟机中启动FastDFS的跟踪服务器和存储服务器，以及Nginx服务器。直接在test目录下通过“go run”命令启动测试程序，打印文件凭证。

项目实战

用浏览器访问：http://<Nginx服务器地址:端口号>/<文件凭证>



- `http://192.168.0.111:8888/group1/M00/00/00/wKgAb2P4W0SAAncYAAFO9DtMGm0528.png`

打开浏览器，地址栏中输入包含文件凭证的URL，浏览器窗口中应显示出我们用Go语言代码上传的图片。

更多精彩，敬请期待

谢谢大家，我们下节课再见！