# 自定义存储系统

之前文件上传时，django通过FileSystemStorage类将文件上传到media\_root指定的目录下，现在要用fsatdfs进行远程存储，定义的类必须是storage类的子类，存储类必须实现\_open()和\_save()方法。

1、可能项目中多个应用都要用到上传存储功能，定义一个工具类，utils-->fdfs-->storage.py:

from django.core.files.storage import Storage

from fdfs\_client.client import Fdfs\_client

class FDFSStorage(Storage):

‘’’fastdfs文件存储类’’’

def \_open(self, name, mode=’rb’):

修改client.conf

base\_path = 日志文件路径，电脑上必须有

tracker\_server = 所在电脑地址（不能是本地环回）

‘’’打开文件时使用’’’

pass

def \_save(self, name, content):

‘’’保存文件时使用’’’

# name:选择的上传文件的名字

# content:包含你上传文件内容的file对象

# 创建一个Fdfs\_client对象

client = Fdfs\_client(‘./utils/fdfs/client.conf’)

# 根据文件内容上传文件到fdfs

res = client.upload\_by\_buffer(content.read()) # 返回字典

if res.get(‘Status’) != ‘Upload successed.’:

raise Exception(‘上传文件失败’)

filename = res.get(‘Remote file\_id’)

return filename

def exists(self, name):

ip和client.conf路径可以在项目settings.py中自定义配置，然后导入

‘’’django判断文件名是否可用’’

return False

def url(self, name):

‘’’返回文件url路径’’’

return ‘http://ip和端口号/’ + name # 返回路径存于数据表中没有后缀名

2、在项目settings.py中配置：

DEFAULT\_FILE\_STORAGE = 'utils.fdfs.storage.FDFSStorage'

3、前端调用图片时，goods.image返回的是fdfs返回的id，而goods.image.url则返回上面拼接的 ‘http://ip和端口号/’ + name