HTTP编程

笔记本：	HTTP协议		
创建时间：	2018/5/15 20:04	**更新时间：**	2018/5/16 14:53
作者：	ly		
标签：	HTTP/1.1 200 OK \r\n		

1，返回固定数据给浏览器

```
import socket


def main():
    """返回固定数据"""
    # 创建套接字
    tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    # 端口复用
    tcp.setsocketopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    # 绑定端口
    tcp.bind(("", 8888))
    # 设为被动监听
    tcp.listen(128)
    # 重复接收，发送数据
    while True:
        # 取出客户，创建新套接字
        new_socket, cli_addr = tcp.accept()
        print(cli_addr, "连上来咯！")
        # 接收数据
        recv_data = new_socket.recv(1024)
        print(cli_addr, ">>>>>>", recv_data.decode())

        # 响应包体
        send_data = "HTTP/1.1 200 OK\r\n"
        send_data += "\r\n"   # 空行
        send_data += "hello"
        # 发送数据
        new_socket.send(send_data.encode())
        关闭服务套接字
        new_socket.close()

    # 关闭监听套接字
    tcp.close()

if __name__ == "__main__":
    main()
```

2，解析用户的请求，返回特定的页面给客户浏览器(支持默认index页面）

```
import socket
import re


def handle_client(new_socket):
   """接收 并返回固定数据"""
   # 接收数据
   recv_data = new_socket.recv(1024).decode()
   print("*" * 40)
   print("recv_data = ", recv_data)
   # 接收到浏览器后面的用户请求内容：http://127.0.0.1:8888/index.html
   # 实际：已经经过浏览器包装HTTP格式的请求报文  下面是解析这些报文
   # 以行为单位切割
   recv_list = recv_data.splitlines()
   # GET /index.html HTTP/l.1   ====> recv_list[0]
   # [^/]+(/[^ ]+).*
```

```
    # 请求头部
    # 空行
    # 请求包体

    # 通过正则匹配请求的index.html
    print(recv_list[0])
    firstLine = recv_list[0]

    fileName = re.match(r"[^/]+(/[^ ]*)", firstLine).group(1)
    print(fileName)

    # 默认网页，当用户/后面没有内容匹配到只有/时打开index.html
    if fileName == "/"
        fileName = "/index.html"

    # 组成路径    ./  当前路径下查找html文件夹
    # 本页代码 的路径 有一个html文件夹，要访问的页面在html页面下面
    filePath = "./html" + fileName
    print(filePath)

    if
    try:
        with open(filePath, "rb") as f:
            content = f.read()
    except Exception as ret:
        # 没有这个请求的资源 404
        send_data = "HTTP/1.1 404 NOT FOUND\r\n"
        send_data += "\r\n"
        content = "file not found".encode()
    else:
        # 有这个请求对应的资源 200
        # 构造适合网页格式的响应体
        send_data = "HTTP/1.1 200 OK\r\n"
        send_data += "\r\n"
        # send_data += "ok"
        # 发送数据
    new_socket.send(send_data.encode())
    new_socket.send(content)

    new_socket.close()


def main():
    """返回固定数据"""
    # 创建套接字
    tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # 端口复用
    tcp.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

    # 端口绑定
    tcp.bind(("", 8888))

    # 监听
    tcp.listen(128)

    while True:
        # 取出客户 返回服务套接字
        new_socket, cli_addr = tcp.accept()
        print("%s连接上来咯！" % cli_addr[0])

        handle_client(new_socket)
```

```
    # 关闭监听套接字
    tcp.close()


if __name__ == '__main__':
    main()
```

3，多任务协程版 打补丁：只要有阻塞（I/O大部分操作），就算不是gevent，sleep()也会自动切换

```
import socket
import re
import gevent


# 导入gevent包中的monkey包
from gevent import monkey
# 打补丁  固定书写
monkey.patch_all()

def handle_client(new_socket):
    """接收 并返回固定数据"""
    # 接收数据
    recv_data = new_socket.recv(1024).decode()
    print("*" * 40)
    print("recv_data = ", recv_data)
    # 接收到浏览器后面的用户请求内容：http://127.0.0.1:8888/index.html
    # 实际：已经经过浏览器包装HTTP格式的请求报文  下面是解析这些报文
    # 以行为单位切割
    recv_list = recv_data.splitlines()
    # GET /index.html HTTP/l.1   =====> recv_list[0]
    # [^/]+(/[^ ]+).*
    # 请求头部
    # 空行
    # 请求包体

    # 通过正则匹配请求的index.html
    print(str(recv_list[0]))
    firstLine = recv_list[0]

    fileName = re.match(r"[^/]+(/[^ ]*)", firstLine).group(1)
    print(fileName)

    # 支持默认路径
    if fileName == "/":
        fileName = "/index.html"

    # 组成路径    ./   当前路径下查找html文件夹
    # 本页代码 的路径 有一个html文件夹，要访问的页面在html页面下面
    filePath = "./html1" + fileName
    print(filePath)


    try:
        with open(filePath, "rb") as f:
            content = f.read()
    except Exception as ret:
        # 没有这个请求的资源 404
        send_data = "HTTP/1.1 404 NOT FOUND\r\n"
        send_data += "\r\n"
```

```
        content = "file not found".encode()
    else:
        # 有这个请求对应的资源 200
        # 构造适合网页格式的响应体
        send_data = "HTTP/1.1 200 OK\r\n"
        send_data += "\r\n"
        # send_data += "ok"
        # 发送数据
    new_socket.send(send_data.encode())
    new_socket.send(content)

    new_socket.close()


def main():
    """返回固定数据"""
    # 创建套接字
    tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # 端口复用
    tcp.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

    # 端口绑定
    tcp.bind(("", 8888))

    # 监听
    tcp.listen(128)

    while True:
        # 取出客户 返回服务套接字
        new_socket, cli_addr = tcp.accept()
        print("%s连接上来咯！" % cli_addr[0])

        # handle_client(new_socket)

        # 创建gevent对象，指定协程处理函数
        gevent.spawn(handle_client, new_socket)

        # 关闭监听套接字
    tcp.close()


if __name__ == '__main__':
    main()
```