package xhu.edu.socket3;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.io.OutputStream;

import java.io.OutputStreamWriter;

import java.io.UnsupportedEncodingException;

import java.net.ServerSocket;

import java.net.Socket;

/\*

\* 模拟web服务器Tomcat向浏览器端发送静态资源。

\* 步骤：

\* 监听指定端口，等待连接

\* 获取连接成功的输入输出流

\* 解析请求行，获取资源路径

\* 按照协议格式返回资源

\*

\*/

public class Server {

public static void main(String[] args) throws IOException {

ServerSocket serverSocket = new ServerSocket(1000);

while (true) {

// 等待客户连接，该方法为阻塞方法

Socket scSocket = serverSocket.accept();

new Thread(new Runnable() {

// 获取套接字的输入输出流

InputStream in = scSocket.getInputStream();

OutputStream out = scSocket.getOutputStream();

@Override

public void run() {

// 解析资源路径

byte[] repath = new byte[1024];

try {

in.read(repath);

} catch (IOException e2) {

// TODO Auto-generated catch block

e2.printStackTrace();

}

String[] reStrings = new String(repath).split(" ");

InputStream flie = null;

try {

// 获取当前项目路径然后连接上请求路径就能找到相应的资源路径

flie = new FileInputStream(System.getProperty("java.class.path") + reStrings[1]);

} catch (FileNotFoundException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

// 如果找不到资源路径

if (flie == null) {

try {

out.write("HTTP/1.1 404 BAD\n".getBytes());

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

} else {

try {

out.write("HTTP/1.1 200 OK\n".getBytes());

out.write("content-type:image/jpeg\n".getBytes());

out.write("\n".getBytes());

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

/\*

\* 读一行文字。 一行被认为是由换行符（'\ n'），回车符（'\ r'），回车符后紧跟换行符或到达文件结尾的任何一个终止（EOF）。 结果

\* 包含行内容的字符串，不包括任何行终止字符;如果在未读取任何字符的情况下到达流末尾，则返回null

\*/

byte[] b = new byte[1024];

try {

while (flie.read(b) != -1) {

out.write(b);

}

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}).start();

}

}

}

运行截图：



总结：

纯Java项目并不会像Javaweb项目，能直接将请求路径进行映射，但是即便是再复杂的项目也是建立在Java项目的基础上的，所以我们能自己进行处理，然后映射到正确的位置，所以我们需要借助一些Java的工具方法进行获取到我们当前的项目路径，再通过连接请求路径进行寻求资源。System.getProperty("java.class.path")，该方法就能获取到我们平时想要的路径。