# 1、概念

HttpClient 是Apache Jakarta Common 下的子项目，可以用来提供高效的、最新的、功能丰富的支持 HTTP 协议的客户端编程工具包，并且它支持 HTTP 协议最新的版本和建议。

# 2、功能：

（1）实现了所有HTTP的方法（GET POST PUT DELETE等）

（2）支持自动转向

（3）支持HTTPS协议

（4）支持代理服务器等

# 3、使用方法

使用HttpClient发送请求、接收响应很简单，一般需要如下几步：

1. 创建HttpClient对象。

2、创建请求方法的实例，并指定请求的URL，如果需要发送GET请求，创建HttpGet对象，如果需要发送POST请求，创建HttpPost对象。

3、如果需要发送请求参数，可以调用HttpGet、HttpPost共同的setParams(HttpParams params)方法来添加请求参数，对于HttpPost对象而言，也可调用setEntity(HttpEntity entity)方法来设置请求参数。

4、调用HttpClient对象的execute(HttpUriRequest request)发送请求，该方法返回一个HttpResponse对象。

5、调用HttpResponse对象的getAllHeaders()、getHeaders(String name)等方法可获取服务器的响应头，调用HttpResponse的getEntity()方法可获取HttpEntity对象，该对象包装了服务器的响应内容，程序可通过该对象获取服务器的响应内容。

6、释放连接。无论执行方法是否成功，都必须释放连接。

示例：



# 3、读取网页（HTTP/HTTPS）中的内容

下面是我们给出的一个简单的例子用来访问某个页面

/\*\*

\*最简单的HTTP客户端,用来演示通过GET或者POST方式访问某个页面

\*@authorLiudong

\*/

public class SimpleClient {

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

{

HttpClient client = new HttpClient();

// 设置代理服务器地址和端口

//client.getHostConfiguration().setProxy("proxy\_host\_addr",proxy\_port);

// 使用 GET 方法 ，如果服务器需要通过 HTTPS 连接，那只需要将下面 URL 中的 http 换成 https

HttpMethod method=new GetMethod("http://java.sun.com");

//使用POST方法

//HttpMethod method = new PostMethod("http://java.sun.com");

client.executeMethod(method);

//打印服务器返回的状态

System.out.println(method.getStatusLine());

//打印返回的信息

System.out.println(method.getResponseBodyAsString());

//释放连接

method.releaseConnection();

}

}

在这个例子中首先创建一个HTTP客户端(HttpClient)的实例，然后选择提交的方法是GET或者POST，最后在HttpClient实例上执行提交的方法，最后从所选择的提交方法中读取服务器反馈回来的结果。这就是使用HttpClient的基本流程。其实用一行代码也就可以搞定整个请求的过程，非常的简单！

# 4、使用POST方式提交数据（httpClient3）

httpclient使用了单独的一个HttpMethod子类来处理文件的上传，这个类就是MultipartPostMethod，该类已经封装了文件上传的细节，我们要做的仅仅是告诉它我们要上传文件的全路径即可，下面这里将给出关于两种模拟上传方式的代码

## （1）模拟上传url文件（该方式也适合做普通的post请求）

/\*\*

\* 上传url文件到指定URL

\* @param fileUrl 上传图片url

\* @param postUrl 上传路径及参数,注意有些中文参数需要使用预先编码 eg : URLEncoder.encode(appName, "UTF-8")

\* @return

\* @throws IOException

\*/

public static String doUploadFile(String postUrl) throws IOException {

if(StringUtils.isEmpty(postUrl))

return null;

String response = "";

PostMethod postMethod = new PostMethod(postUrl);

try {

HttpClient client = new HttpClient();

client.getHttpConnectionManager().getParams()

.setConnectionTimeout(50000);// 设置连接时间

int status = client.executeMethod(postMethod);

if (status == HttpStatus.SC\_OK) {

InputStream inputStream = postMethod.getResponseBodyAsStream();

BufferedReader br = new BufferedReader(new InputStreamReader(

inputStream));

StringBuffer stringBuffer = new StringBuffer();

String str = "";

while ((str = br.readLine()) != null) {

stringBuffer.append(str);

}

response = stringBuffer.toString();

} else {

response = "fail";

}

} catch (Exception e) {

e.printStackTrace();

} finally {

// 释放连接

postMethod.releaseConnection();

}

return response;

}

## （2）模拟文件上传到指定位置

/\*\*

\* 上传文件到指定URL

\* @param file

\* @param url

\* @return

\* @throws IOException

\*/

public static String doUploadFile(File file, String url) throws IOException {

String response = "";

if (!file.exists()) {

return "file not exists";

}

PostMethod postMethod = new PostMethod(url);

try {

//----------------------------------------------

// FilePart：用来上传文件的类,file即要上传的文件

FilePart fp = new FilePart("file", file);

Part[] parts = { fp };

// 对于MIME类型的请求，httpclient建议全用MulitPartRequestEntity进行包装

MultipartRequestEntity mre = new MultipartRequestEntity(parts,

postMethod.getParams());

postMethod.setRequestEntity(mre);

//---------------------------------------------

HttpClient client = new HttpClient();

client.getHttpConnectionManager().getParams()

.setConnectionTimeout(50000);// 由于要上传的文件可能比较大 , 因此在此设置最大的连接超时时间

int status = client.executeMethod(postMethod);

if (status == HttpStatus.SC\_OK) {

InputStream inputStream = postMethod.getResponseBodyAsStream();

BufferedReader br = new BufferedReader(new InputStreamReader(

inputStream));

StringBuffer stringBuffer = new StringBuffer();

String str = "";

while ((str = br.readLine()) != null) {

stringBuffer.append(str);

}

response = stringBuffer.toString();

} else {

response = "fail";

}

} catch (Exception e) {

e.printStackTrace();

} finally {

// 释放连接

postMethod.releaseConnection();

}

return response;

}

# 常用的类

RequestBuilder：这个类是HttpUriRequest实例的创建者

StringEntity：一个自包含的、可重复的实体，从字符串中获取其内容。