HttpClient

<dependency>  
 <groupId>org.apache.httpcomponents</groupId>  
 <artifactId>httpclient</artifactId>  
 <version>4.5.13</version>  
</dependency>

核心API

HttpClient HttpClients CloseableHttpClient HttpGet HttpPost

发送请求步骤

创建HttpClient对象 创建Http请求对象 调用HttpClient的execute方法发送请求

Get

//http客户端对象，可以发送http请求  
CloseableHttpClient httpClient = HttpClients.createDefault();  
  
//构造Get方式请求  
HttpGet httpGet = new HttpGet("http://localhost:8080/user/shop/status");  
  
//发送请求  
CloseableHttpResponse response = httpClient.execute(httpGet);  
  
//http响应码  
int statusCode = response.getStatusLine().getStatusCode();  
  
//http响应体  
HttpEntity entity = response.getEntity();  
//将响应体转为String字符串  
String body = EntityUtils.toString(entity);  
System.out.println(body);  
  
//关闭资源  
response.close();  
httpClient.close();

Post

CloseableHttpClient httpClient = HttpClients.createDefault();  
//Post方式请求  
HttpPost httpPost = new HttpPost("http://localhost:8080/admin/employee/login");  
  
//构造json数据  
JSONObject jsonObject = new JSONObject();  
jsonObject.put("username","admin");  
jsonObject.put("password", "123456");  
  
//构造请求体  
StringEntity stringEntity = new StringEntity(jsonObject.toString());  
//设置请求编码  
stringEntity.setContentEncoding("utf-8");  
//设置数据类型  
stringEntity.setContentType("application/json");  
//设置当前Post请求的请求体  
httpPost.setEntity(stringEntity);  
  
//发送请求  
CloseableHttpResponse response = httpClient.execute(httpPost);  
  
//http响应码  
int statusCode = response.getStatusLine().getStatusCode();  
  
//http响应体  
HttpEntity entity = response.getEntity();  
//将响应体转为String字符串  
String body = EntityUtils.toString(entity);  
System.out.println(body);  
  
//关闭资源  
response.close();  
httpClient.close();

Http工具类

public class HttpClientUtil {

static final int TIMEOUT\_MSEC = 5 \* 1000;

/\*\*

\* 发送GET方式请求

\* @param url

\* @param paramMap

\* @return

\*/

public static String doGet(String url,Map<String,String> paramMap){

// 创建Httpclient对象

CloseableHttpClient httpClient = HttpClients.createDefault();

String result = "";

CloseableHttpResponse response = null;

try{

URIBuilder builder = new URIBuilder(url);

if(paramMap != null){

for (String key : paramMap.keySet()) {

builder.addParameter(key,paramMap.get(key));

}

}

URI uri = builder.build();

//创建GET请求

HttpGet httpGet = new HttpGet(uri);

//发送请求

response = httpClient.execute(httpGet);

//判断响应状态

if(response.getStatusLine().getStatusCode() == 200){

result = EntityUtils.toString(response.getEntity(),"UTF-8");

}

}catch (Exception e){

e.printStackTrace();

}finally {

try {

response.close();

httpClient.close();

} catch (IOException e) {

e.printStackTrace();

}

}

return result;

}

/\*\*

\* 发送POST方式请求

\* @param url

\* @param paramMap

\* @return

\* @throws IOException

\*/

public static String doPost(String url, Map<String, String> paramMap) throws IOException {

// 创建Httpclient对象

CloseableHttpClient httpClient = HttpClients.createDefault();

CloseableHttpResponse response = null;

String resultString = "";

try {

// 创建Http Post请求

HttpPost httpPost = new HttpPost(url);

// 创建参数列表

if (paramMap != null) {

List<NameValuePair> paramList = new ArrayList();

for (Map.Entry<String, String> param : paramMap.entrySet()) {

paramList.add(new BasicNameValuePair(param.getKey(), param.getValue()));

}

// 模拟表单

UrlEncodedFormEntity entity = new UrlEncodedFormEntity(paramList);

httpPost.setEntity(entity);

}

httpPost.setConfig(builderRequestConfig());

// 执行http请求

response = httpClient.execute(httpPost);

resultString = EntityUtils.toString(response.getEntity(), "UTF-8");

} catch (Exception e) {

throw e;

} finally {

try {

response.close();

} catch (IOException e) {

e.printStackTrace();

}

}

return resultString;

}

/\*\*

\* 发送POST方式请求

\* @param url

\* @param paramMap

\* @return

\* @throws IOException

\*/

public static String doPost4Json(String url, Map<String, String> paramMap) throws IOException {

// 创建Httpclient对象

CloseableHttpClient httpClient = HttpClients.createDefault();

CloseableHttpResponse response = null;

String resultString = "";

try {

// 创建Http Post请求

HttpPost httpPost = new HttpPost(url);

if (paramMap != null) {

//构造json格式数据

JSONObject jsonObject = new JSONObject();

for (Map.Entry<String, String> param : paramMap.entrySet()) {

jsonObject.put(param.getKey(),param.getValue());

}

StringEntity entity = new StringEntity(jsonObject.toString(),"utf-8");

//设置请求编码

entity.setContentEncoding("utf-8");

//设置数据类型

entity.setContentType("application/json");

httpPost.setEntity(entity);

}

httpPost.setConfig(builderRequestConfig());

// 执行http请求

response = httpClient.execute(httpPost);

resultString = EntityUtils.toString(response.getEntity(), "UTF-8");

} catch (Exception e) {

throw e;

} finally {

try {

response.close();

} catch (IOException e) {

e.printStackTrace();

}

}

return resultString;

}

private static RequestConfig builderRequestConfig() {

return RequestConfig.custom()

.setConnectTimeout(TIMEOUT\_MSEC)

.setConnectionRequestTimeout(TIMEOUT\_MSEC)

.setSocketTimeout(TIMEOUT\_MSEC).build();

}

}