Demo36-封装HttpURLConnection和OkHttp

1. 封装HttpURLConnection

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import okhttp3.Callback;

import okhttp3.OkHttpClient;

import okhttp3.Request;

import okhttp3.Response;

public class HttpUtil {

public static void sendHttpGET(final String address, final HttpCallbackListener listener) {

new Thread(new Runnable() {

@Override

public void run() {

HttpURLConnection connection = null;

try {

URL url = new URL(address);

connection = (HttpURLConnection) url.openConnection();

connection.setRequestMethod("GET");

connection.setConnectTimeout(8000);

connection.setReadTimeout(8000);

connection.setDoInput(true);

connection.setDoOutput(true);

InputStream in = connection.getInputStream();

BufferedReader reader = new BufferedReader(new InputStreamReader(in));

StringBuilder response = new StringBuilder();

String line;

while ((line = reader.readLine()) != null) {

response.append(line);

}

if (listener != null) {

// 回调onFinish()方法

listener.onFinish(response.toString());

}

} catch (Exception e) {

if (listener != null) {

// 回调onError()方法

listener.onError(e);

}

} finally {

if (connection != null) {

connection.disconnect();

}

}

}

}).start();

}

}

在子线程用回调来处理返回的数据或者错误，然后重写onFinish可以使用Handler或者runOnUiThread返回到主线程更新UI

public interface HttpCallbackListener {

void onFinish(String response);

void onError(Exception e);

}

Post请求

public void loginByPost(String userName, String userPass) {

try {

// 请求的地址

String spec = "http://172.16.237.200:8080/video/login.do";

// 根据地址创建URL对象

URL url = new URL(spec);

// 根据URL对象打开链接

HttpURLConnection urlConnection = (HttpURLConnection) url

.openConnection();

// 设置请求的方式

urlConnection.setRequestMethod("POST");

// 设置请求的超时时间

urlConnection.setReadTimeout(5000);

urlConnection.setConnectTimeout(5000);

// 传递的数据

String data = "username=" + URLEncoder.encode(userName, "UTF-8")

+ "&userpass=" + URLEncoder.encode(userPass, "UTF-8");

// 设置请求的头

urlConnection.setRequestProperty("Connection", "keep-alive");

// 设置请求的头

urlConnection.setRequestProperty("Content-Type",

"application/x-www-form-urlencoded");

// 设置请求的头

urlConnection.setRequestProperty("Content-Length",

String.valueOf(data.getBytes().length));

// 设置请求的头

urlConnection

.setRequestProperty("User-Agent",

"Mozilla/5.0 (Windows NT 6.3; WOW64; rv:27.0) Gecko/20100101 Firefox/27.0");

urlConnection.setDoOutput(true); // 发送POST请求必须设置允许输出

urlConnection.setDoInput(true); // 发送POST请求必须设置允许输入

//setDoInput的默认值就是true

//获取输出流

OutputStream os = urlConnection.getOutputStream();

os.write(data.getBytes());

os.flush();

if (urlConnection.getResponseCode() == 200) {

// 获取响应的输入流对象

InputStream is = urlConnection.getInputStream();

// 创建字节输出流对象

ByteArrayOutputStream baos = new ByteArrayOutputStream();

// 定义读取的长度

int len = 0;

// 定义缓冲区

byte buffer[] = new byte[1024];

// 按照缓冲区的大小，循环读取

while ((len = is.read(buffer)) != -1) {

// 根据读取的长度写入到os对象中

baos.write(buffer, 0, len);

}

// 释放资源

is.close();

baos.close();

// 返回字符串

final String result = new String(baos.toByteArray());

// 通过runOnUiThread方法进行修改主线程的控件内容

LoginActivity.this.runOnUiThread(new Runnable() {

@Override

public void run() {

// 在这里把返回的数据写在控件上 会出现什么情况尼

tv\_result.setText(result);

}

});

} else {

System.out.println("链接失败.........");

}

} catch (Exception e) {

e.printStackTrace();

}

}

}

封装OkHttp

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import okhttp3.Callback;

import okhttp3.OkHttpClient;

import okhttp3.Request;

import okhttp3.Response;

public class HttpUtil {

public static void sendOkHttpRequest(final String address, final okhttp3.Callback callback) {

OkHttpClient client = new OkHttpClient();

Request request = new Request.Builder()

.url(address)

.build();

client.newCall(request).enqueue(callback);

}

}

调用示例：

HttpUtil.sendOkHttpRequest(“https//www.baidu.com”,new okhttp3.Callback() {

@Override

public void onResponse(Call call,Response response)throws IOException{

String responseData=response.body().string();

}

@Override

public void onFailure(Call call,IOException e){

}

});