链滴

安卓 OKHttp 的使用

作者：xynling

原文链接：https://ld246.com/article/1542776726696

来源网站：链滴

一，OKHttp介绍

okhttp是一个第三方类库，用于android中请求网络。

这是一个开源项目,是安卓端最火热的轻量级框架,由移动支付Square公司贡献(该公司还贡献了Picass和LeakCanary) 。用于替代HttpUrlConnection和Apache HttpClient(android API23 里已移除HttpClient)。

okhttp有自己的官网，官网网址：OKHttp官网

如果想了解原码可以在github上下载，地址是：https://github.com/square/okhttp

在AndroidStudio中使用不需要下载jar包，直接添加依赖即可：

compile ‘com.squareup.okhttp3:okhttp:3.4.1’

下面对以OKHttp3来详细介绍OKHttp的使用方法。

二，get请求的使用方法

使用OKHttp进行网络请求支持两种方式，一种是同步请求，一种是异步请求。下面分情况进行介绍。

1，get的同步请求

对于同步请求在请求时需要开启子线程，请求成功后需要跳转到UI线程修改UI。

使用示例如下：

```
public void getDatasync(){    new Thread(new Runnable() {        @Override        public void run() {            try {                OkHttpClient client = new OkHttpClient();//创建OkHttpClient对象        Request request = new Request.Builder()                .url("http://www.baidu.com")//请接口。如果需要传参拼接到接口后面。                .build();//创建Request 对象                Response response = null;                response = client.newCall(request).execute();//得到Response 对象                if (response.isSuccessful()) {                    Log.d("kwwl","response.code()=="+response.code());                Log.d("kwwl","response.message()=="+response.message());                    Log.d("wwl","res=="+response.body().string());                //此时的代码执行在子线程，修改UI的操作请用handler跳转到UI线程。                }            } catch (Exception e) {                e.printStackTrace();            }        }    }).start();}
```

此时打印结果如下：

response.code()==200；

response.message()==OK；

res=={“code”:200,”message”:success};

注意事项：

1，Response.code是http响应行中的code，如果访问成功则返回200.这个不是服务器设置的，而是http协议中自带的。res中的code才是服务器设置的。注意二者的区别。

2，response.body().string()本质是输入流的读操作，所以它还是网络请求的一部分，所以这行代码须放在子线程。

3，response.body().string()只能调用一次，在第一次时有返回值，第二次再调用时将会返回null。

因是：response.body().string()的本质是输入流的读操作，必须有服务器的输出流的写操作时客户端读操作才能得到数据。而服务器的写操作只执行一次，所以客户端的读操作也只能执行一次，第二次返回null。

2，get的异步请求

这种方式不用再次开启子线程，但回调方法是执行在子线程中，所以在更新UI时还要跳转到UI线程中。

使用示例如下：

```
private void getDataAsync() {    OkHttpClient client = new OkHttpClient();    Request request = new Request.Builder()            .url("http://www.baidu.com")            .build();    client.newCall(request).enqueue(new Callback() {        @Override        public void onFailure(Call call, IOException ) {        }        @Override        public void onResponse(Call call, Response response) throws IOException {            if(response.isSuccessful()){//回调的方法执行在子线程。                Log.d("kwwl", 获取数据成功了");                Log.d("kwwl","response.code()=="+response.code());                Log.d("kwwl","response.body().string()=="+response.body().string());            }        }    });}
```

异步请求的打印结果与注意事项与同步请求时相同。最大的不同点就是异步请求不需要开启子线程，equeue方法会自动将网络请求部分放入子线程中执行。

注意事项：

1，回调接口的onFailure方法和onResponse执行在子线程。

2，response.body().string()方法也必须放在子线程中。当执行这行代码得到结果后，再跳转到UI线修改UI。

三，post请求的使用方法

Post请求也分同步和异步两种方式，同步与异步的区别和get方法类似，所以此时只讲解post异步请的使用方法。

使用示例如下：

```
private void postDataWithParame() {    OkHttpClient client = new OkHttpClient();//创建OkHttClient对象。    FormBody.Builder formBody = new FormBody.Builder();//创建表单请求体    forBody.add("username","zhangsan");//传递键值对参数    Request request = new Request.Builder(//创建Request 对象。            .url("http://www.baidu.com")            .post(formBody.build())//传请求体            .build();    client.newCall(request).enqueue(new Callback() {。。。});//回调方法的用与get异步请求相同，此时略。}
```

看完代码我们会发现：post请求中并没有设置请求方式为POST，回忆在get请求中也没有设置请求方为GET，那么是怎么区分请求方式的呢？重点是Request.Builder类的post方法，在Request.Builder象创建最初默认是get请求，所以在get请求中不需要设置请求方式，当调用post方法时把请求方式修为POST。所以此时为POST请求。

四，POST请求传递参数的方法总结

在post请求使用方法中讲了一种传递参数的方法，就是创建表单请求体对象，然后把表单请求体对象为post方法的参数。post请求传递参数的方法还有很多种，但都是通过post方法传递的。下面我们看下Request.Builder类的post方法的声明：

```
public Builder post(RequestBody body)
```

由方法的声明可以看出，post方法接收的参数是RequestBody 对象，所以只要是RequestBody 类以子类对象都可以当作参数进行传递。FormBody就是RequestBody 的一个子类对象。

1，使用FormBody传递键值对参数

这种方式用来上传String类型的键值对

使用示例如下：

```
private void postDataWithParame() {    OkHttpClient client = new OkHttpClient();//创建OkHtt
Client对象。    FormBody.Builder formBody = new FormBody.Builder();//创建表单请求体    for
Body.add("username","zhangsan");//传递键值对参数    Request request = new Request.Builder(
//创建Request 对象。            .url("http://www.baidu.com")            .post(formBody.build())//传
请求体            .build();    client.newCall(request).enqueue(new Callback() {。。。});//此处省略回
方法。}
```

2，使用RequestBody传递Json或File对象

RequestBody是抽象类，故不能直接使用，但是他有静态方法create，使用这个方法可以得到Reques
Body对象。

这种方式可以上传Json对象或File对象。

上传json对象使用示例如下：

```
OkHttpClient client = new OkHttpClient();//创建OkHttpClient对象。MediaType JSON = Media
ype.parse("application/json; charset=utf-8");//数据类型为json格式，String jsonStr = "{\"usern
me\":\"lisi\",\"nickname\":\"李四\"}";//json数据.RequestBody body = RequestBody.create(JSON
 josnStr);Request request = new Request.Builder()        .url("http://www.baidu.com")        .post
body)        .build();client.newCall(request).enqueue(new Callback() {。。。});//此处省略回调方
。
```

上传File对象使用示例如下：

```
OkHttpClient client = new OkHttpClient();//创建OkHttpClient对象。MediaType fileType = Med
aType.parse("File/*");//数据类型为json格式，File file = new File("path");//file对象.RequestBody
ody = RequestBody.create(fileType , file );Request request = new Request.Builder()        .url("ht
p://www.baidu.com")        .post(body)        .build();client.newCall(request).enqueue(new Callba
k() {。。。});//此处省略回调方法。
```

3，使用MultipartBody同时传递键值对参数和File对象

这个字面意思是多重的body。我们知道FromBody传递的是字符串型的键值对，RequestBody传递是多媒体，那么如果我们想二者都传递怎么办？此时就需要使用MultipartBody类。

使用示例如下：

```
OkHttpClient client = new OkHttpClient();MultipartBody multipartBody =new MultipartBody.
uilder()        .setType(MultipartBody.FORM)        .addFormDataPart("groupId",""+groupId)//
加键值对参数        .addFormDataPart("title","title")        .addFormDataPart("file",file.getName(),
equestBody.create(MediaType.parse("file/*"), file))//添加文件        .build();final Request request
= new Request.Builder()        .url(URLContant.CHAT_ROOM_SUBJECT_IMAGE)        .post(multip
rtBody)        .build();client.newCall(request).enqueue(new Callback() {。。。});
```

4，自定义RequestBody实现流的上传

在上面的分析中我们知道，只要是RequestBody类以及子类都可以作为post方法的参数，下面我们自定义一个类，继承RequestBody，实现流的上传。

使用示例如下：

首先创建一个RequestBody类的子类对象：

```
RequestBody body = new RequestBody() {    @Override    public MediaType contentType() {    return null;    }    @Override    public void writeTo(BufferedSink sink) throws IOException {/重写writeTo方法        FileInputStream fio= new FileInputStream(new File("fileName"));        byte ] buffer = new byte[1024*8];        if(fio.read(buffer) != -1){            sink.write(buffer);        }    }};
```

然后使用body对象：

```
 OkHttpClient client = new OkHttpClient();//创建OkHttpClient对象。Request request = new Request.Builder()        .url("http://www.baidu.com")        .post(body)        .build();client.newCall(request).enqueue(new Callback() {。。。});
```

以上代码的与众不同就是body对象，这个body对象重写了write方法，里面有个sink对象。这个是Okio包中的输出流，有write方法。使用这个方法我们可以实现上传流的功能。

使用RequestBody上传文件时，并没有实现断点续传的功能。我可以使用这种方法结合RandomAccessFile类实现断点续传的功能。

五，设置请求头

OKHttp中设置请求头特别简单，在创建request对象时调用一个方法即可。

使用示例如下：

```
Request request = new Request.Builder()            .url("http://www.baidu.com")            .header("User-Agent", "OkHttp Headers.java")            .addHeader("token", "myToken")            .uild();
```

其他部分代码略。

六，下载文件

在OKHttp中并没有提供下载文件的功能，但是在Response中可以获取流对象，有了流对象我们就可自己实现文件的下载。代码如下：

这段代码写在回调接口CallBack的onResponse方法中：

```
try{    InputStream  is = response.body().byteStream();//从服务器得到输入流对象    long sum = 0    File dir = new File(mDestFileDir);    if (!dir.exists()){        dir.mkdirs();    }    File file = new File(ir, mdestFileName);//根据目录和文件名得到file对象    FileOutputStream  fos = new FileOutputStream(file);    byte[] buf = new byte[1024*8];    int len = 0;    while ((len = is.read(buf)) != -1){ fos.write(buf, 0, len);    }    fos.flush();    return file; }
```

七，对于OKHttp的使用封装

由于okhttp是偏底层的网络请求类库，返回结果的回调方法仍然执行在子线程中，需要自己跳转到UI

程，使用麻烦。为了使用方便需要对OKHttp进行再次封装。对于OKHttp的封装首推的就是hongyan大神的OKHttpUtils。我个人在看过OKHttp的原码和借鉴各大神的封装源码后封装了一套自己的OKHttpUtils。这套OKHttpUtils最大的优点是简单和便于使用，这是我项目中实际用的网络请求工具类，全可以说拿来即用。而且代码简单，可供学习使用。

github的地址是：https://github.com/guozhengXia/OkHttpUtils

封装的功能有：

- 一般的get请求
- 一般的post请求
- 上传单个文件(包含进度)
- 上传list集合文件
- 上传map集合文件
- 文件下载(包含进度)
- 图片下载(实现了图片的压缩)

请大家多多支持，多多提出宝贵意见