android 介绍Retrofit的简单使用

Retrofit与okhttp共同出自于Square公司，retrofit就是对okhttp做了一层封装。把网络请求都交给给了Okhttp，我们只需要通过简单的配置就能使用retrofit来进行网络请求了，其主要作者是Android大神[JakeWharton](https://github.com/JakeWharton" \t "http://cache.baiducontent.com/_blank)。

导包：

compile 'com.squareup.retrofit2:retrofit:2.0.0-beta4'//Retrofit2所需要的包

compile 'com.squareup.retrofit2:converter-gson:2.0.0-beta4'//ConverterFactory的Gson依赖包

compile 'com.squareup.retrofit2:converter-scalars:2.0.0-beta4'//ConverterFactory的String依赖包

\*这里需要值得注意的是导入的retrofit2包的版本必须要一致，否则就会报错。

首先定义我们请求的Api，我们假设是这样的

http://106.3.227.33/pulamsi/mobileLogin/submit.html

与Okhttp不同的是，Retrofit需要定义一个接口，用来返回我们的Call对象，这里示范的是Post请求：

public interface RequestServes {

@POST("mobileLogin/submit.html")

Call<String> getString(@Query("loginname") String loginname, @Query("nloginpwd") String nloginpwd);}

Retrofit提供的请求方式注解有@GET和@POST，参数注解有@PATH和@Query等，我们只介绍常用的;前两个顾名思义就是定义你的请求方式Get or Post，后面的@PATH指的是通过参数填充完整的路径，一般用法：

@GET("{name}")  
Call<User> getUser(@Path("name") String username);

这里的参数username会被填充至{name}中，形成完整的Url请求地址，{name}相当于一个占位符；

@Query就是我们的请求的键值对的设置，我们构建Call对象的时候会传入此参数，

@POST("mobileLogin/submit.html")

Call<String> getString(@Query("loginname") String loginname, @Query("nloginpwd") String nloginpwd);

这里@Query("loginname")就是键，后面的loginname就是具体的值了，值得注意的是Get和Post请求，都是这样填充参数的；

接口写完了之后我们需要来定义Retrofit对象来进行请求了；

创建一个Retrofit对象

Retrofit retrofit = new Retrofit.Builder()

.baseUrl("http://106.3.227.33/pulamsi/")

//增加返回值为String的支持

.addConverterFactory(ScalarsConverterFactory.create())

//增加返回值为Gson的支持(以实体类返回)

.addConverterFactory(GsonConverterFactory.create())

//增加返回值为Oservable<T>的支持

.addCallAdapterFactory(RxJavaCallAdapterFactory.create())

.build();

这里的baseUrl加上之前@POST("mobileLogin/submit.html")定义的参数形成完整的请求地址；

addConverterFactory(ScalarsConverterFactory.create())的意思是构建了一个返回支持，如果你的Call对象的泛型接收另外的格式需要添加另外的支持，上述代码已经列出；

接下来我们用这个Retrofit对象创建一个RequestSerives接口对象，也就是我们之前定义的那个接口，并且得到我们的Call对象；

RequestSerives requestSerives = retrofit.create(RequestSerives.class);//这里采用的是Java的动态代理模式

Call<String> call = requestSerives.getString("userName", "1234");//传入我们请求的键值对的值

利用得到的Call对象，然后我们就发出网络请求了：

call.enqueue(new Callback<String>() {

@Override

public void onResponse(Call<String> call, Response<String> response){

Log.e("===","return:"response.body().toString());

}

@Override

public void onFailure(Call<String> call, Throwable t) {Log.e("===","失败");}

});