链滴

Retrofit 初体验

作者：flowaters

原文链接：https://ld246.com/article/1532448767494

来源网站：链滴

背景

如果看到了OkHttp, RxJava相关文章的话，一定会同时看到Retrofit。

OkHttp是对http客户端进行了封装，RxJava是反应式宣言，那Retrofit是什么呢?

于是这便是本文的写作背景。

第一个例子

Retrofit的一个用法是直接把Http请求封装为Java对象。举个例子就明白了。

封装请求参数

原始的url为:

```
curl -s "http://ip.taobao.com/service/getIpInfo2.php?ip=119.75.213.61"
{"code":0,"data":{"ip":"119.75.213.61","country":"中国","area":"","region":"北京","city":"北京","c
unty":"XX","isp":"电信","country_id":"CN","area_id":"","region_id":"110000","city_id":"110100","
ounty_id":"xx","isp_id":"100017"}}
```

下面开始变魔术，将url网络调用变为java方法调用，调用方对网络基本无感知。

IPService.java

```
package testRetrofit.ip;

import okhttp3.ResponseBody;
import retrofit2.Call;
import retrofit2.http.GET;
import retrofit2.http.Query;

public interface IPService {
    @GET("service/getIpInfo2.php")
    Call<ResponseBody> getip(@Query("ip") String ip);

}
```

MainIp.java

```
package testRetrofit.ip;

import java.io.IOException;

import okhttp3.ResponseBody;
import retrofit2.Call;
import retrofit2.Retrofit;

public class MainIp {
    public static void main(String[] args) throws IOException {
        Retrofit retrofit = new Retrofit.Builder().baseUrl("http://ip.taobao.com/").build();
        IPService service = retrofit.create(IPService.class);
```

```
        // 这里是调用方, 仅仅能看到execute()了一下
        Call<ResponseBody> repos = service.getip("119.75.213.61");
        ResponseBody repo = repos.execute().body();
        System.out.println(repo.string());

    }
}
```

运行结果

{"code":0,"data":{"ip":"119.75.213.61","country":"中国","area":"","region":"北京","city":"北京","county":"XX","isp":"电信","country_id":"CN","area_id":"","region_id":"110000","city_id":"110100","ounty_id":"xx","isp_id":"100017"}}

在本例中，通过接口注解，调用java方法，Retrofit自动将其转化为http请求参数，去请求服务器，返回结果。

本例主要演示参数的封装。

封装运行结果

上面的例子中，返回的结果为ResponseBody，还需要自己处理。在本例中，通过定义一个IP类，来接返回IP类对象。具体如下：

IPService2.java

```
package testRetrofit.ip2;

import retrofit2.Call;
import retrofit2.http.GET;
import retrofit2.http.Query;

public interface IPService2 {
    @GET("service/getIpInfo2.php")
    Call<Result<IP>> getip(@Query("ip") String ip);
}
```

Result.java

```
package testRetrofit.ip2;

public class Result<T> {
    public int code;
    public T data;

    @Override
    public String toString() {
        return "Result [code=" + code + ", data=" + data + "]";
    }
}
```

IP.java

```
package testRetrofit.ip2;

public class IP {
    public String ip;
    public String country;
    public String area;
    public String region;
    public String city;

    public String county;
    public String isp;
    public String country_id;
    public String area_id;
    public String region_id;

    public String city_id;
    public String county_id;
    public String isp_id;

    @Override
    public String toString() {
        return "IP [ip=" + ip + ", country=" + country + ", area=" + area + ", region=" + region +
", city=" + city
                + ", county=" + county + ", isp=" + isp + ", country_id=" + country_id + ", area_id="
+ area_id
                + ", region_id=" + region_id + ", city_id=" + city_id + ", county_id=" + county_id + ",
isp_id="
                + isp_id + "]";
    }
}
```

MainIp2.java

```
package testRetrofit.ip2;

import java.io.IOException;

import retrofit2.Call;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class MainIp2 {
    public static void main(String[] args) throws IOException {
        Retrofit retrofit = new Retrofit.Builder().baseUrl("http://ip.taobao.com/")
                .addConverterFactory(GsonConverterFactory.create()).build();

        IPService2 service = retrofit.create(IPService2.class);

        // 这里是调用方, 仅仅能看到execute()了一下，返回了IP对象结果。
        Call<Result<IP>> repos = service.getip("119.75.213.61");
        Result<IP> repo = repos.execute().body();
        System.out.println(repo.toString());
    }
}
```

运行结果

Result [code=0, data=IP [ip=119.75.213.61, country=中国, area=, region=北京, city=北京, county=XX, isp=电信, country_id=CN, area_id=, region_id=110000, city_id=110100, county_id=xx, ip_id=100017]]

到此为止，完成了一个入门使用说明。但是实际情况中，对于输入和输出可能有更复杂的情况。这时怎么用retrofit来处理呢?

输入和输出

输入和输出可能有多种情况。

输入

输入参数的指定方法，可以通过url，可以通过body，可以通过表单、分块表单，可以通过Header等。

这里仅罗列一下，需要的时候可以去[1]中来找示例，都不复杂。

请求方法:

- @GET
- @POST
- @PUT
- @DELETE
- @PATCH
- @HEAD
- @OPTIONS
- @HTTP

请求标记:

- @FormUrlEncoded: 表单中的field
- @Multipart: 分传上传
- @Streaming: 流形式返回

请求参数:

- @Headers: Header控制
- @Header: Header控制
- @HeaderMap: Header控制
- @Body: HTTP请求body
- @Field: 表单中的field
- @FieldMap: 表单中的field
- @Part: 分传上传
- @PartMap: 分传上传

- @Path: 通过路径传递参数，如"users/{user}/repos"
- @Query:
- @QueryMap:
- @Url:

输出

输出的结果可以是Json格式的，可以是Protobuf格式的，可以是XML格式的，可以是Scalars格式的也可以是用户自定义格式的。

前面的例子即是Json格式的。

- Json适配器：提供了Gson, Jackson和Moshi三种。
- Protobuf适配器：提供了Protobuf和Wire两种。
- XML适配器：Simple XML

自定义Converter

如果返回的结果不是标准的Json格式，而是自定义的格式，又想返回成Java对象直接使用，怎么办呢?

这时可以通过自定义Converter，来实现定制输出结果类。

下面举个自定义格式的例子：

通过IPResultConverter和IPResultConverterFactory来解析返回的IP Json，输出{国家}{省}{市}

IPResultConverter.java

```
package testRetrofit.ip3;

import java.io.IOException;
import java.util.Map;

import com.google.gson.Gson;

import okhttp3.ResponseBody;
import retrofit2.Converter;

public class IPResultConverter implements Converter<ResponseBody, String> {

    @Override
    public String convert(ResponseBody value) throws IOException {
        // 将responseBody转换为了string
        Map result = new Gson().fromJson(value.string(), Map.class);
        if (result.containsKey("data")) {
            Map dataMap = (Map) result.get("data");
            return dataMap.get("country") + "_" + dataMap.get("region") + "_" + dataMap.get("cit
");
        }
        return null;
```

```
    }
}
```

IPResultConverterFactory.java

```
package testRetrofit.ip3;

import java.io.IOException;
import java.util.Map;

import com.google.gson.Gson;

import okhttp3.ResponseBody;
import retrofit2.Converter;

public class IPResultConverter implements Converter<ResponseBody, String> {

    @Override
    public String convert(ResponseBody value) throws IOException {
        Map result = new Gson().fromJson(value.string(), Map.class);
        if (result.containsKey("data")) {
            Map dataMap = (Map) result.get("data");
            return dataMap.get("country") + "_" + dataMap.get("region") + "_" + dataMap.get("cit
");
        }
        return null;
    }
}
```

IPService3.java

```
package testRetrofit.ip3;

import retrofit2.Call;
import retrofit2.http.GET;
import retrofit2.http.Query;

public interface IPService3 {
    @GET("service/getIpInfo2.php")
    Call<String> getip(@Query("ip") String ip);

}
```

MainIp3.java

```
package testRetrofit.ip3;

import java.io.IOException;

import retrofit2.Call;
import retrofit2.Retrofit;

public class MainIp3 {
    public static void main(String[] args) throws IOException {
        Retrofit retrofit = new Retrofit.Builder().baseUrl("http://ip.taobao.com/")
```

```
                .addConverterFactory(new IPResultConverterFactory()).build();

        IPService3 service = retrofit.create(IPService3.class);

        Call<String> repos = service.getip("119.75.213.61");
        String repo = repos.execute().body();
        System.out.println(repo);
    }
}
```

结果

```
中国_北京_北京
```

自定义CallAdapter

自定义CallAdapter是做什么的呢?

默认情况下，我们返回的是一个Call<?>的结果。如果我们不想要Call<?>，而是想要Observable<?>或者想要LiveData<?>呢?

retrofit允许通过自定义CallAdapter，来对Call<?>再做一次转换。

如果看了前面的反应式宣言的话，这里这么做的目的就很好理解了。

同样举个例子，其中没有详细来写，仅做示例：

IPService4.java

```
package testRetrofit.ip4;

import okhttp3.ResponseBody;
import retrofit2.http.GET;
import retrofit2.http.Query;
import rx.Observable;

public interface IPService4 {
    @GET("service/getIpInfo2.php")
    Observable<ResponseBody> getip(@Query("ip") String ip);

}
```

MainIp4.java

```
package testRetrofit.ip4;

import java.io.IOException;

import okhttp3.ResponseBody;
import retrofit2.Retrofit;
import retrofit2.adapter.rxjava.RxJavaCallAdapterFactory;
import rx.Observable;
import rx.Subscriber;
import rx.schedulers.Schedulers;
```

```
public class MainIp4 {
    public static void main(String[] args) throws IOException {
        Retrofit retrofit = new Retrofit.Builder().baseUrl("http://ip.taobao.com/")
                .addCallAdapterFactory(RxJavaCallAdapterFactory.create()).build();

        IPService4 service = retrofit.create(IPService4.class);

        Observable<ResponseBody> repos = service.getip("119.75.213.61");

        repos.observeOn(Schedulers.io()).subscribe(new Subscriber<ResponseBody>() {

            @Override
            public void onCompleted() {
            }

            @Override
            public void onError(Throwable error) {
            }

            @Override
            public void onNext(ResponseBody body) {
                try {
                    System.out.println(body.string());
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });

        try {
            Thread.sleep(10000L);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

结果

{"code":0,"data":{"ip":"119.75.213.61","country":"中国","area":"","region":"北京","city":"北京","c
unty":"XX","isp":"电信","country_id":"CN","area_id":"","region_id":"110000","city_id":"110100","
ounty_id":"xx","isp_id":"100017"}}

参考

1. Retrofit官方文档
2. 你真的会用Retrofit2吗?Retrofit2完全教程