目录

[Android框架 2](#_Toc516333452)

[Retrofit（http） 2](#_Toc516333453)

[RxJava 3](#_Toc516333454)

[Retrofit + RxJava 4](#_Toc516333455)

# Android框架

## Retrofit（http）

<https://www.jianshu.com/p/308f3c54abdd>

1. **public** **class** Example01 {
2. **public** **interface** BlogService {
3. @GET("blog/{id}") //这里的{id} 表示是一个变量
4. Call<ResponseBody> getBlog(/\*\* 这里的id表示的是上面的{id} \*/@Path("id") **int** id);
5. }
7. **public** **static** **void** main(String[] args) **throws** IOException {
8. Retrofit retrofit = **new** Retrofit.Builder()
9. .baseUrl("http://localhost:4567/")
10. .build();
12. BlogService service = retrofit.create(BlogService.**class**);
13. Call<ResponseBody> call = service.getBlog(2);
14. // 用法和OkHttp的call如出一辙
15. // 不同的是如果是Android系统回调方法执行在主线程
16. call.enqueue(**new** Callback<ResponseBody>() {
17. @Override
18. **public** **void** onResponse(
19. Call<ResponseBody> call, Response<ResponseBody> response) {
20. **try** {
21. System.out.println(response.body().string());
22. } **catch** (IOException e) {
23. e.printStackTrace();
24. }
25. }
27. @Override
28. **public** **void** onFailure(Call<ResponseBody> call, Throwable t) {
29. t.printStackTrace();
30. }
31. });
32. }
33. }

## RxJava

<https://blog.csdn.net/sqm8822/article/details/79142458>

1. Observable.from(folders)
2. .flatMap(**new** Func1<File, Observable<File>>() {
3. @Override
4. **public** Observable<File> call(File file) {
5. **return** Observable.from(file.listFiles());
6. }
7. })
8. .filter(**new** Func1<File, Boolean>() {
9. @Override
10. **public** Boolean call(File file) {
11. **return** file.getName().endsWith(".png");
12. }
13. })
14. .map(**new** Func1<File, Bitmap>() {
15. @Override
16. **public** Bitmap call(File file) {
17. **return** getBitmapFromFile(file);
18. }
19. })
20. .subscribeOn(Schedulers.io())
21. .observeOn(AndroidSchedulers.mainThread())
22. .subscribe(**new** Action1<Bitmap>() {
23. @Override
24. **public** **void** call(Bitmap bitmap) {
25. imageCollectorView.addImage(bitmap);
26. }
27. });

## Retrofit + RxJava

<https://www.jianshu.com/p/94f6370ed18d?from=timeline&isappinstalled=1>

先设计interface

1. **public** **interface** GetWeatherService {
2. @GET("weather\_mini")
3. Observable<WeatherEntity> getRxMessage(@Query("city") String city);
4. }

再调用

1. Retrofit retrofit = **new** Retrofit.Builder()
2. .baseUrl("http://wthrcdn.etouch.cn/")//基础URL
3. .addConverterFactory(GsonConverterFactory.create())//设置 Json 转换器
4. .addCallAdapterFactory(RxJavaCallAdapterFactory.create())//RxJava 适配器
5. .build();
7. GetWeatherService weatherService = retrofit.create(GetWeatherService.**class**);
9. weatherService.getRxMessage("北京")
10. .subscribeOn(Schedulers.io())//IO线程加载数据
11. .observeOn(AndroidSchedulers.mainThread())//主线程显示数据
12. .subscribe(**new** Subscriber<WeatherEntity>() {
13. @Override
14. **public** **void** onCompleted() {
16. }
18. @Override
19. **public** **void** onError(Throwable e) {
21. }
23. @Override
24. **public** **void** onNext(WeatherEntity weatherEntity) {
25. Log.e(TAG,"RxJava + Retrofit= " + weatherEntity.getData().getGanmao());
26. }
27. });