中山大学数据科学与计算机学院软件工程（移动信息工程）本科生实验报告

移动应用开发实验报告

姓名	学号	班级	电话	邮箱	日期
蔡烨	15352010	1501班	15989010669	caiy29@mail2.sysu.edu.cn	2017/12/24

1. 实验题目

Retrofit + RxJava + OkHttp实现网络请求

2. 实现内容

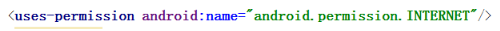
- 学习使用retrofit实现网络请求
- 学习RxJava中observable的使用
- 复习同步异步概念

3. 实验过程

1. 写三个xml文件：activity_main.xml，cardview.xml，details.xml。其中只有cardview是新接触的，但是也并不抽象，不难使用。还有设置TextView只有一行，并且省略号在结尾处。

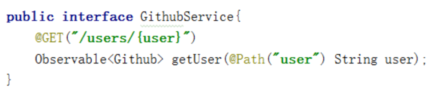
```
android:maxLines="1"
android:ellipsize="end"
```

2. 注册网络访问权限；定义两个model类：Github, Repos，一个适配器CardAdapter，在前面的实验中都已经写过，不再赘述。

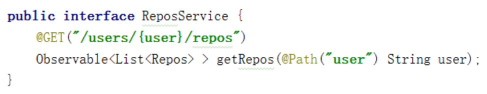


```
<uses-permission android:name="android.permission.INTERNET"/>
```

3. 定义两个访问接口 API interface，提供URL，返回RxJava中的observable类型，需注意的是repos的接口返回是的一个列表。Retrofit通过给访问接口方法添加相应的注解来表示该方法对应于HTTP的哪种请求。



```
public interface GithubService{
    @GET("/users/{user}")
    Observable<Github> getUser(@Path("user") String user);
}
```



```
public interface ReposService {
    @GET("/users/{user}/repos")
    Observable<List<Repos> > getRepos(@Path("user") String user);
}
```

4. 构造retrofit对象实现网络访问。Retrofit是对OkHttp的封装，提供了使用注解更简单的构建各种请求，配置各种参数的方式。但本质发起网络请求的还是OkHttp。

```java
public class ServiceFactory {
    //负责发起网络请求，维护网络连接
    private static OkHttpClient createOkHttp(){
        OkHttpClient okHttpClient = new OkHttpClient.Builder()
                .connectTimeout(10,  TimeUnit.SECONDS) //连接超时
                .readTimeout(30,TimeUnit.SECONDS) //读超时
                .writeTimeout(10,TimeUnit.SECONDS) //写超时
                .build();
        return okHttpClient;
    }
    //将网络传输的数据转换为可用的model对象，提供简单的数据处理方式
    public static Retrofit createRetrofit(String baseurl){
        return new Retrofit.Builder()
                .baseUrl(baseurl)
                .addConverterFactory(GsonConverterFactory.create())
                .addCallAdapterFactory(RxJavaCallAdapterFactory.create())
                .client(createOkHttp())
                .build();
    }
}
```

5. MainActivity.java

为recyclerview配置adapter

```java
recyclerView = (RecyclerView) findViewById(R.id.recycler);
cards = new ArrayList<Github>();
cardAdapter = new CardAdapter(MainActivity.this,cards);
recyclerView.setLayoutManager(new LinearLayoutManager(this));
//recyclerView.setAdapter(cardAdapter);
//有动画的适配器
ScaleInAnimationAdapter animationAdapter = new ScaleInAnimationAdapter(cardAdapter);
animationAdapter.setDuration(700);
recyclerView.setAdapter(animationAdapter);
recyclerView.setItemAnimator(new OvershootInLeftAnimator());
```

cardItem的点击事件

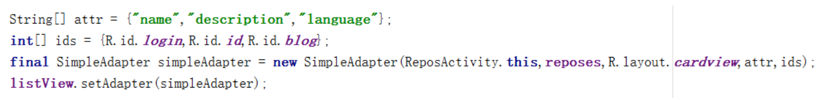
```java
cardAdapter.setOnItemClickListener(new CardAdapter.OnItemClickListener() {
    @Override
    public void onClick(int position) {
        Intent intent = new Intent(MainActivity.this,ReposActivity.class);
        intent.putExtra("login",cards.get(position).getLogin());
        startActivity(intent);
    }
    @Override
    public void onLongClick(int position) {
        cards.remove(position);
//      cardAdapter.notifyItemRemoved(position);
        cardAdapter.notifyItemRemoved(position);
        if(position != cards.size()){
            cardAdapter.notifyItemRangeChanged(position,cards.size()-position);
        }
    }
});
```

fetch按钮的点击事件：调用相应的接口函数获取数据

```java
fetch.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        progressBar.setVisibility(View.VISIBLE);
        Retrofit retrofit = ServiceFactory.createRetrofit(BASE_URL);
        GithubService service = retrofit.create(GithubService.class);
        String search = input.getText().toString();
        service.getUser(search) //获取observable对象
                .subscribeOn(Schedulers.newThread()) //请求在新的线程中执行
                .observeOn(AndroidSchedulers.mainThread())
                .subscribe(new Subscriber<Github>(){
                    @Override
                    public void onCompleted() {//请求结束时调用的回调函数
                        System.out.print("完成传输");
                        progressBar.setVisibility(View.INVISIBLE);
                    }
                    @Override
                    public void onError(Throwable e) {//请求出现错误时的回调函数
                        Toast.makeText(MainActivity.this,e.hashCode()+"请确认你的搜索用户存在",Toast.LENGTH_SHORT).show();
                        Log.e("Github-Demo",e.getMessage());
                        progressBar.setVisibility(View.INVISIBLE);
                    }
                    @Override
                    public void onNext(Github github) {//每一次收到数据时的回调函数
                        if(github != null)
                            cards.add(github);
                        cardAdapter.notifyDataSetChanged();
```

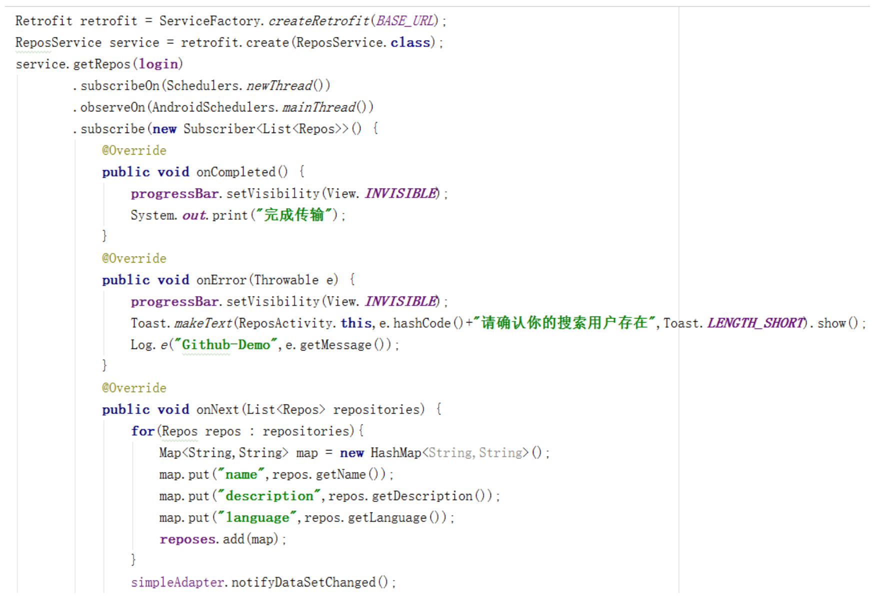
6. ReposActivity.java

为listView设置adapter，使用的是与recyclerview一样的layout：cardview.xml



```
String[] attr = {"name","description","language"};
int[] ids = {R.id.login,R.id.id,R.id.blog};
final SimpleAdapter simpleAdapter = new SimpleAdapter(ReposActivity.this,reposes,R.layout.cardview,attr,ids);
listView.setAdapter(simpleAdapter);
```

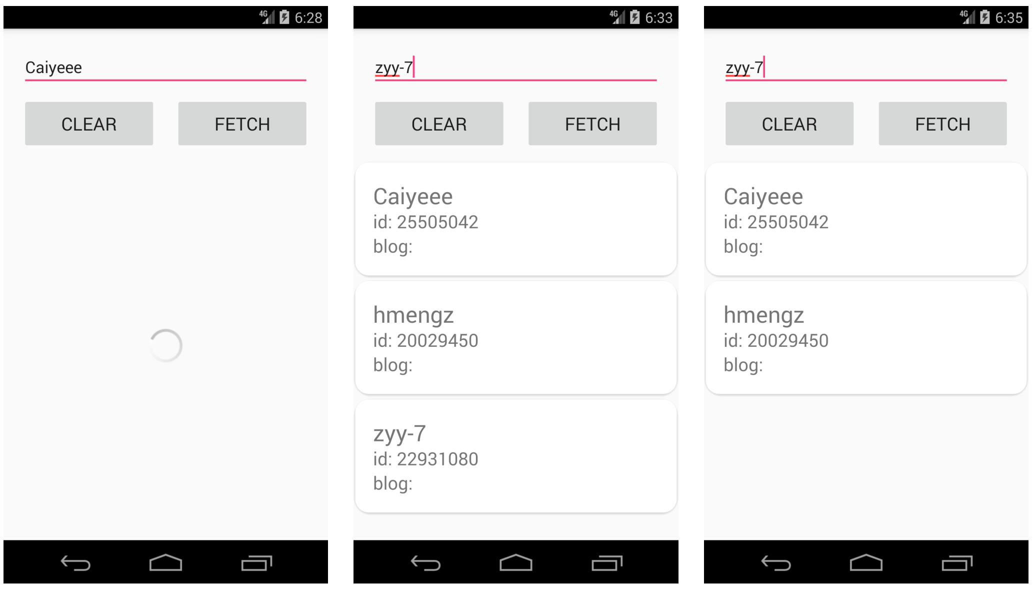
调用接口函数，获取repository数据



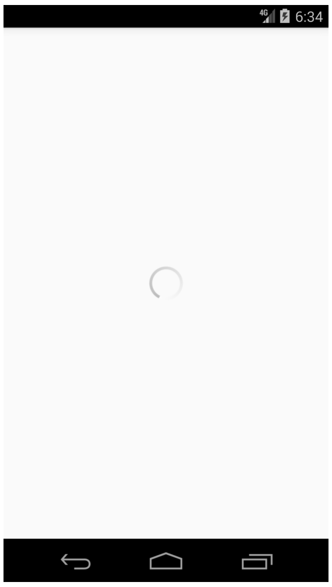
```
Retrofit retrofit = ServiceFactory.createRetrofit(BASE_URL);
ReposService service = retrofit.create(ReposService.class);
service.getRepos(login)
        .subscribeOn(Schedulers.newThread())
        .observeOn(AndroidSchedulers.mainThread())
        .subscribe(new Subscriber<List<Repos>>() {
            @Override
            public void onCompleted() {
                progressBar.setVisibility(View.INVISIBLE);
                System.out.print("完成传输");
            }
            @Override
            public void onError(Throwable e) {
                progressBar.setVisibility(View.INVISIBLE);
                Toast.makeText(ReposActivity.this,e.hashCode()+"请确认你的搜索用户存在",Toast.LENGTH_SHORT).show();
                Log.e("Github-Demo",e.getMessage());
            }
            @Override
            public void onNext(List<Repos> repositories) {
                for(Repos repos : repositories){
                    Map<String,String> map = new HashMap<String,String>();
                    map.put("name",repos.getName());
                    map.put("description",repos.getDescription());
                    map.put("language",repos.getLanguage());
                    reposes.add(map);
                }
                simpleAdapter.notifyDataSetChanged();
```

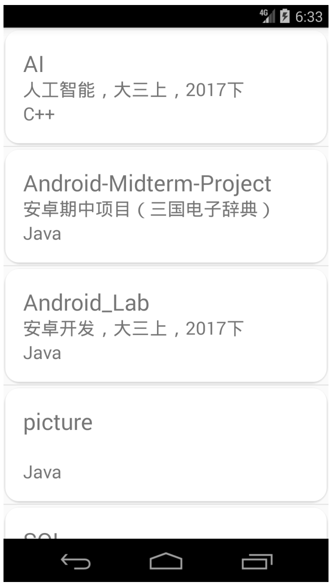
4. 实验结果截图

从左到右：搜索等待、搜索、长按删除后



从左到右：进入repos页面等待、repos页面、clear后





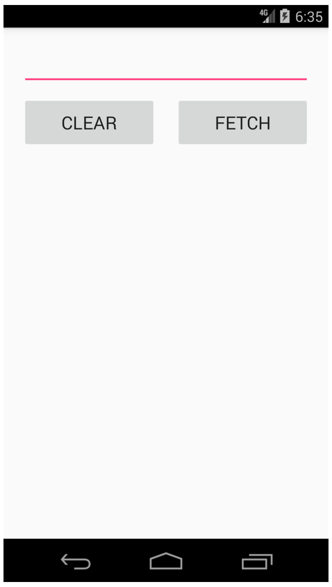
AI
人工智能，大三上，2017下
C++

Android-Midterm-Project
安卓期中项目（三国电子辞典）
Java

Android_Lab
安卓开发，大三上，2017下
Java

picture

Java



5. 实验思考及感想

本次试验虽然TA说很简单，代码都给出来了，但是因为这个内容感觉比较抽象比较难懂，所以还是纠结了许久。经过这次实验，懂得了如何设置TextView只有一行并且用...来代替，学会了使用cardview，知道了怎样让app利用OkHttp、Retrofit连接到网络并且使用get来获取数据。