如何把 RxJava 应用到知乎

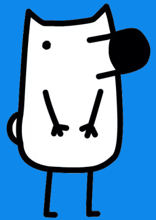
by 何若昕

关于我

何若昕

- 知乎 Android 开发工程师
- 业余独立开发者

目录

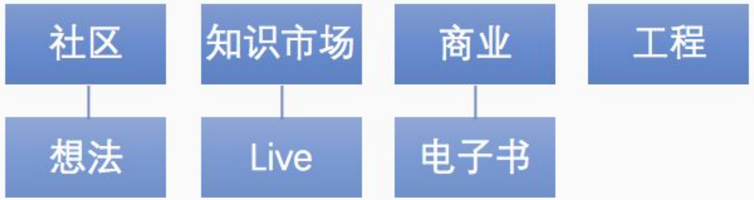


知乎 Android 开发团队的现状

团队概况：

一共 21 名开发工程师

小组划分：



关于 RxJava，整个团队内要求：

人人会读

但不强制写

特例：

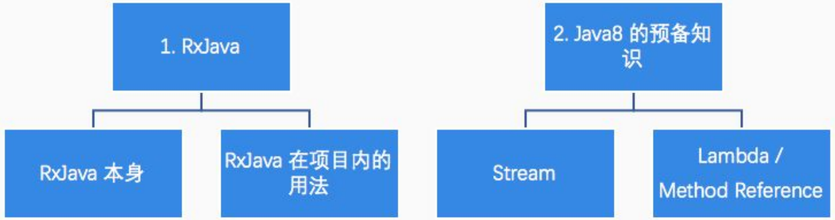
网络请求（Retrofit）只能用 RxJava

事件通知仅使用 RxBus

如何在组内推广？

新人上船：

周期举行「小讲堂」培训



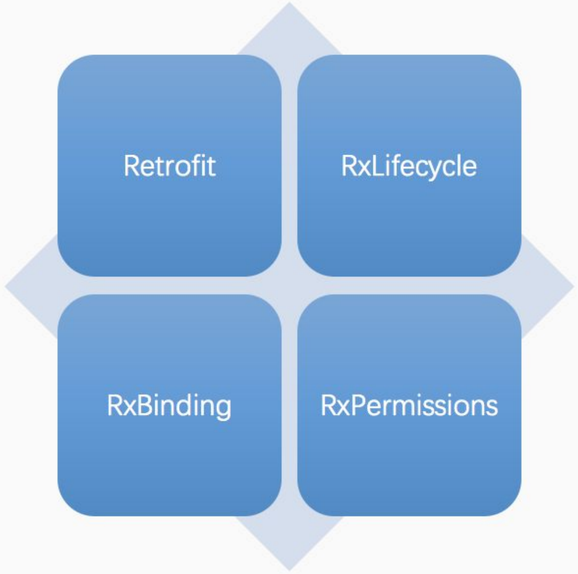
老手进阶：

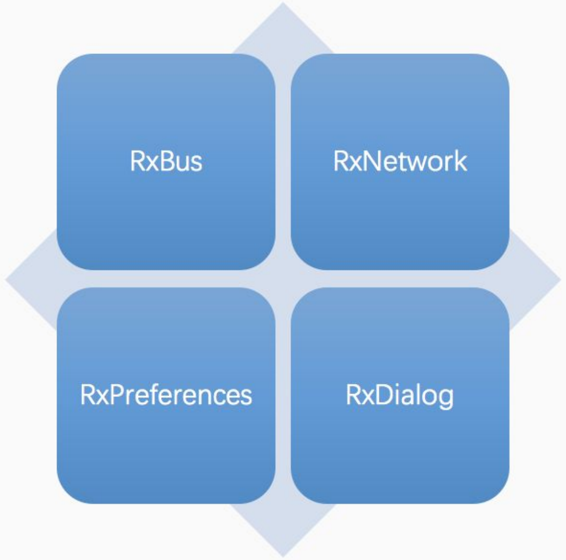
每周一次的分享

wiki 文档

踩坑笔记

现有脚手架


Retrofit
RxLifecycle
RxBinding
RxPermissions


RxBus
RxNetwork
RxPreferences
RxDialog

脚手架：三方库

Retrofit

以 Rx Observable 的方式请求网络

RxPermissions

以 Rx 的方式申请系统权限

RxBinding

提供各种 Android 原生组件的 Rx 绑定

RxLifecycle

将 Fragment/Activity 的生命周期输出为 RxJava 事件

脚手架：三方库 - RxPermissions、RxBinding

```
RxView.clicks(view)
  .throttleFirst(ms, TimeUnit.MILLISECONDS)
  .compose(RxLifecycleAndroid.bindView(view))
  .subscribe(aVoid -> listener.onClick(view));
```

```
new RxPermissions(getActivity())
  .request(Manifest.permission.CAMERA)
  .subscribe(granted -> {
     if (granted) {
        ...
  });
```

脚手架 : 三方库 - Retrofit

```
public interface ProfileService {
        @GET("/people/{member_id}")
        Observable<Response<People>> getPeopleById(@Path("member_id") String pMemberId);
    }

mProfileService = RetrofitInitializer.getDefaultInstance()
  .getRetrofit()
  .create(ProfileService.class);

mProfileService.getPeopleById(memberId)
  .subscribeOn(Schedulers.io())
  .observeOn(AndroidSchedulers.mainThread())
  .subscribe(response -> {
      if (response.isSuccessful()) doSomething(response.body());
  }, Debug::e);
```

脚手架：三方库 - RxLifecycle

它做了两件事：

1. 监听 Fragment/Activity 的生命周期，并输出为 Rx 事件。

2. 通过 bindUtilEvent 来帮助取消耗时超过生命周期长度的订阅，以防止内存泄漏。

```
public class SomeFragment extends RxFragment {
  @Override
  public void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    mProfileService.getPeopleById(memberId)
            .compose(bindUntilEvent(FragmentEvent.DESTROY))
            .subscribeOn(Schedulers.io())
            .observeOn(AndroidSchedulers.mainThread())
            .subscribe(response -> {
                if (response.isSuccessful()) {
                    doSomething(response.body());
                }
            }, Debug::e);
  }
}
```

脚手架：自研小工具

RxBus

类似 EventBus 的，基于 Rx 的事件总线

RxPreferences

对 Preference 读写操作的封装，同时可以将某项值的变化持续输出为 Observable

RxDialog

用 Rx 来处理对话框，将各个 Button 的点击输出为 Observable 事件；onComplete 表示对话框的消失

RxNetwork

将连接断开 wifi/3G 等事件转化为 Rx Observable

脚手架 : 自研小工具 - RxBus

类似 EventBus 的事件通知机制

```
// 发送事件
RxBus.INSTANCE
    .post(new SomeEvent())

// 接收事件
RxBus.INSTANCE
   .toObservable(SomeEvent.class)
   .subscribe(event -> {
       ...
   }, Debug::e);
```

```
public enum  RxBus {
  INSTANCE;
  private PublishSubject<Object> mSubject = PublishSubject.create();
  public void post(Object object) {
      mSubject.onNext(object);
  }
  public <T> Observable<T> toObservable(Class<T> eventType) {
      return mSubject.ofType(eventType);
  }
}
```

脚手架：自研小工具 - RxPreference、RxNetwork

```
RxPreferences.INSTANCE
  .<String>onPreferenceChanged()
  .filter(p -> TextUtils.equals(p.getKey(), SOME_PREFERENCE_KEY))
  .subscribe(p -> {
      ...
  }, Debug::e);

RxNetwork.INSTANCE
  .onConnectionChanged()
  .filter(info -> info.getNetworkType() == ConnectivityManager.TYPE_WIFI)
  .filter(info -> !info.isConnected())
  .observeOn(AndroidSchedulers.mainThread())
  .subscribe(connect -> {
      Debug.d(TAG, "stop video cause wifi disconnected.");
  }, Debug::e);
```

脚手架：自研小工具 - RxDialog

```
new RxDialog(mActivity)
  .title("Dialog Title")
  .message("Dialog Message")
  .positive("OK")
  .negative("Cancel")
  .toObservable()
  .compose(RxUtils.bindLifecycle(mActivity, R.id.DialogId))
  .subscribe(clickEvent -> {
     switch (clickEvent) {
       case PositiveClick:
         break;
       case NegativeClick:
         break;
     }
  }, Debug::e);
```

```
// 当需要时，直接 dispose 订阅即可取消对话框
// 无需到处传递 Dialog 的引用
RxUtils.dispose(mActivity, R.id.DialogId);
```

框架和探索：

MVX 中 X 的生命周期

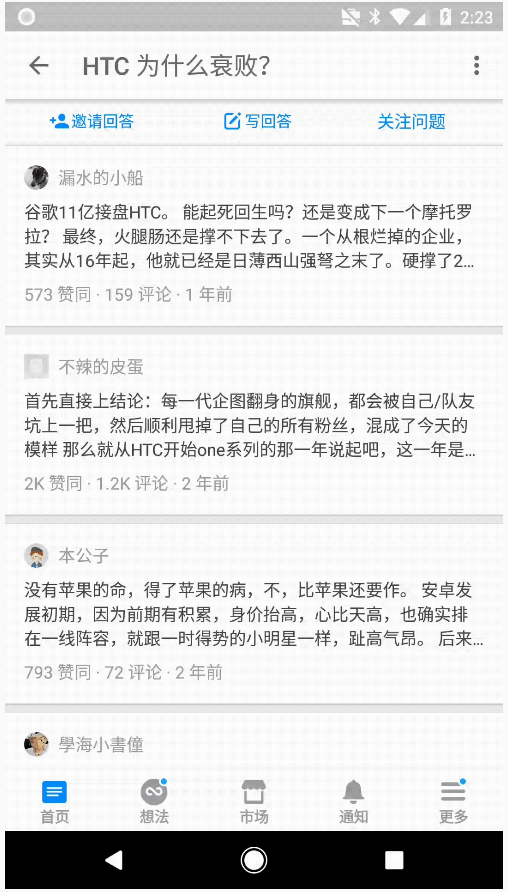
```java
@Override
public void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  List<BaseViewModel> vmList = Lists2.of(
          new SomeVM(),
          new SomeVM1(),
          new SomeVM2()
  );
  StreamSupport.stream(vmList)
          .forEach(vm -> {
              RxFragment.this
                      .lifecycle()
                      .subscribe(vm::syncLifecycleTo);
          });
}
```

```java
public abstract class BaseLifeCycleViewModel extends BaseObservable {
  protected void onCreate() {}
  protected void onCreateView() {}
  ...
  private int mLastOrdinal = -1;

  public void syncLifecycleTo(LifecycleEvent currentEm) {
    int ordinal = currentEm.ordinal();
    if (ordinal == mLastOrdinal) return;
    if (ordinal > mLastOrdinal) {
      RefStreams.of(LifecycleEventMethod.values())
              .skip(mLastOrdinal + 1)
              .filter(em -> em.compareTo(currentEm) <= 0)
              .forEach(em -> em.run(this));
    } else {
      currentEm.run(this);
    }
    mLastOrdinal = ordinal;
  }
}
```

```java
/**
 * CREATE,
 * CREATE_VIEW,
 * START,
 * RESUME,
 * PAUSE,
 * STOP,
 * DESTROY_VIEW,
 * DESTROY,
 */
```

框架探索：连续加载的 RecyclerView


2:23

HTC 为什么衰败?

漏水的小船

谷歌11亿接盘HTC。 能起死回生吗？还是变成下一个摩托罗拉？ 最终，火腿肠还是撑不下去了。一个从根烂掉的企业，其实从16年起，他就已经是日薄西山强弩之末了。硬撑了2...

573 赞同 · 159 评论 · 1 年前

不辣的皮蛋

首先直接上结论：每一代企图翻身的旗舰，都会被自己/队友坑上一把，然后顺利甩掉了自己的所有粉丝，混成了今天的模样 那么就从HTC开始one系列的那一年说起吧，这一年是...

2K 赞同 · 1.2K 评论 · 2 年前

本公子

没有苹果的命，得了苹果的病，不，比苹果还要作。 安卓发展初期，因为前期有积累，身价抬高，心比天高，也确实排在一线阵容，就跟一时得势的小明星一样，趾高气昂。 后来...

793 赞同 · 72 评论 · 2 年前

學海小書僮

1. 从网络（Retrofit）或者数据库（Room）中得到数据（Model）

2. 将 Model 转换为 ViewHolder

3. 显示在界面上

```
Observable.from(...)

.map(...)

subscribe(...);
```

```java
public class SomeVM extends BasePagingRecyclerViewParentViewModel {
  @Override
  public Observable<ZHObjectList> provideSource(Paging paging) {
     return mSomeNetworkApiService.requestDate(paging)
              .map(Response::body)
              .map(ZHObjectList.class::cast);
  }
  @Override
  public Observable<BasePagingRecyclerViewChildViewModel> convertItem(Observable<Object> from) {
     return from .map(SomeModel.class::cast)
              .map(SomeVieModel::wrapModel);
  }
  @Override
  public Observable<BaseRecyclerChildViewModel> convertError(Throwable error) {
     return Observable.just(error).map(ErrorNoticeViewModel::wrapError);
  }
}
```

```java
public class ZHObjectList<T> {
  @JsonProperty("data")
  public List<T> data;


  @JsonProperty("paging")
  public Paging paging;
}
```

Q & A

Thanks

感谢大家