PHUNWARE

# Rx Java

"A library for **composing asynchronous** and event-based programs using **observable sequences**"

(Yes! I copied this from the Netflix guys)



© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

# The Big Question?

© 2009-2014 PHUNWARE, INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

# The Big Question?

Do we really need another way of managing concurrency?



© 2009-2014 PHUNWARE, INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

# Unique Problem I Faced

TMDB API-gives all the info but does not have a long movie description

TRAKT API-all YouTube trailer links work, movie description is nice

phunware.com
© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

# Unique Problem I Faced

TMDB API-gives all the info but does not have a long movie description , sometimes youtube trailers are not available

TRAKT API-all YouTube trailer links work , movie description is nice but info is not available for all movies of tmdb, a lot of small issues.

Rotten Tomatoes API- well a movie app without reviews does not sound great at all :-)

FANART API- a huge collection of fan art

*And the movie detail page wanted all of this info!*

phunware.com
© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

# The Solution

Get the data from TMDB API using the IMDB ID of the movie.

Get the data from the Trakt API for the movie using the IMDB ID.

Rotten Tomatoes API- well a movie app without reviews does not sound great at all :-)

FANART API- a huge collection of fan art

***And the movie detail page wanted all of this info!***



© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

# The Solution

Get the data from TMDB API using the IMDB ID of the movie.

Get the data from the Trakt API for the movie using the IMDB ID.

If Trakt has the data available , show the movie description and trailer from the Trakt api because that seems more reliable

If not , use whatever data TMDB gives you.

Will get to the fanart API & rotten tomatoes in a longer presentation some other day :-)



© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

# Retrofit+Rx Java To The Rescue

- Retrofit has support for observables. So let us create the respective retrofit interfaces

```
public interface MovieDetailServiceTrakt {

    @GET("/{apikey}/{movieid}")
```

if not , use whatever data TMDB gives you.

Will get to the fanart API & rotten tomatoes in a longer presentation some other day :-)


© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

# Retrofit+Rx Java To The Rescue

- Retrofit has support for observables. So let us create the respective retrofit interfaces

```
public interface MovieDetailServiceTrakt {

   @GET("/{apikey}/{movieid}")
    Observable<MovieTrakt> getMovieDetails(@Path("movieid")String movieId,@Path("apikey")String
     apikey);
}
```

```
public interface MovieDetailService {
     @GET("/{movieid}")
     Observable<Movie> getMovieDetails(@Path("movieid") String movieId, @Query("api_key") String
apiKey,@Query("append_to_response") String appendToResponse);

}
```


© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

## Get the Observable from TMDB:-

```
Observable<Movie> movieDetailTmdb= MoviApi.getMovieDetailService()
                                   .getMovieDetails(movieId, Config.MDB.getKey(), "trailers");
```

## Get the Observable from TRAKT:-

```
Observable<MovieTrakt> movieDetailTrakt=MoviApi.getMovieDetailServiceTrakt()
                                        .getMovieDetails(movieId, Config.TRAKT.getKey());
```

Have some Rxjava Fun!

© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

## Get the Observable from TMDB:-

```
Observable<Movie> movieDetailTmdb= MoviApi.getMovieDetailService()
                                   .getMovieDetails(movieId, Config.MDB.getKey(), "trailers");
```

## Get the Observable from TRAKT:-

```
Observable<MovieTrakt> movieDetailTrakt=MoviApi.getMovieDetailServiceTrakt()
                                        .getMovieDetails(movieId, Config.TRAKT.getKey());
```

## Have some RxJava Fun!

```
      Observable.zip(movieDetailTmdb,movieDetailTrakt,new Func2<Movie, MovieTrakt,
MovieDetailMerged>() {
            @Override
            public MovieDetailMerged call(final Movie movie, MovieTrakt movieTrakt) {

                MovieDetailMerged movieDetailMerged=new MovieDetailMerged(movie,movieTrakt);
                return movieDetailMerged;
            }
        });
What we have here is  an object which has the data from both TMDB and TRAKT.
```



© 2009-2014 PHUNWARE INC. CONFIDENTIAL & PROPRIETARY

PHUNWARE

### Observer One

```
public class ObserverOne implements
Observer<Movie>{
    @Override
    public void onCompleted() {

    }

    @Override
    public void onError(Throwable e) {

    }

    @Override
    public void onNext(Movie mov) {
        Timber.tag("observerone");
        Timber.v(mov.original_title);
    }
}
```

### Observer Two

```
public class Observertwo implements Observer<Movie> {
    @Override
    public void onCompleted() {

    }

    @Override
    public void onError(Throwable e) {

    }

    @Override
    public void onNext(Movie mov) {
        Timber.tag("observertwo");
        Timber.v(mov.original_title);
    }
}
```

What we have here is  an object which has the data from both TMDB and TRAKT.



© 2009-2014 PBUNWARE INC. CONFIDENTIAL & PROPRIETARY



## Observer One

```
public class ObserverOne implements
Observer<Movie>{
    @Override
    public void onCompleted() {

    }

    @Override
    public void onError(Throwable e) {

    }

    @Override
    public void onNext(Movie mov) {
        Timber.tag("observerone");
        Timber.v(mov.original_title);
    }
}
```

## Observer Two

```
public class Observertwo implements Observer<Movie> {
    @Override
    public void onCompleted() {

    }

    @Override
    public void onError(Throwable e) {

    }

    @Override
    public void onNext(Movie mov) {
        Timber.tag("observertwo");
        Timber.v(mov.original_title);
    }
}
```

```
ObserverOne one=new ObserverOne();
Observertwo two=new Observertwo();
publishSubject=PublishSubject.create();
publishSubject.subscribe(one);
publishSubject.subscribe(two);

movieService.getUpcomingMovie("MYAPIKEY")
          .map(new MovieListToListOfMovieTransform())
          .flatMap(new Func1<List<Movie>, Observable<Movie>>() {
               @Override
               public Observable<Movie> call(List<Movie> movies) {
                    Timber.v(movies.size()+"--"+movies.get(0).original_title);
                    return Observable.from(movies);
               }
          })
.subscribeOn(Schedulers.io())
.observeOn(AndroidSchedulers.mainThread())
.subscribe(publishSubject);
```



© 2009-2014 PBUNWARE INC. CONFIDENTIAL & PROPRIETARY