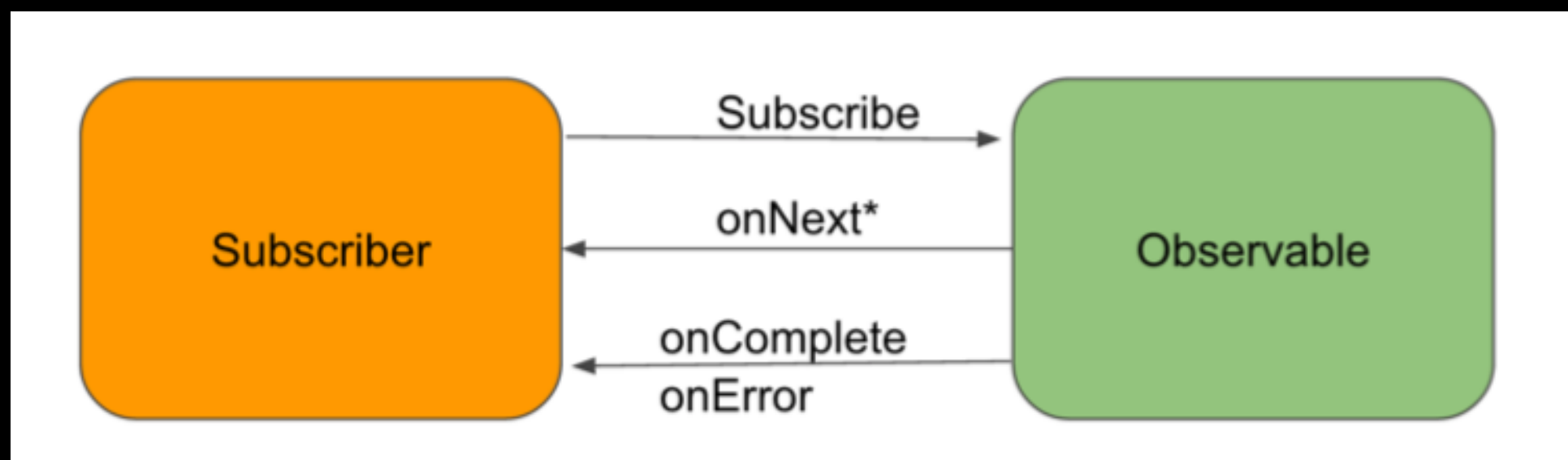


RxJava 2 + Retrofit 2

**Реактивное
программирование?**

Базовые типы

- Observable - источник данных
- Subscriber - наблюдатель



Retrofit 2

```
public interface WeatherApi {  
  
    @GET("data/2.5/forecast/daily")  
    Observable<WeatherData> getWeatherByCity(  
        @Query("q") String city,  
        @Query("cnt") String dayCount,  
        @Query("APPID") String key,  
        @Query("units") String units);  
}
```

Класс App

```
public class App extends Application {  
  
    private static final String WEATHER_BASE_URL = "http://api.openweathermap.org";  
  
    public static WeatherApi mWeatherApi;  
  
    @Override  
    public void onCreate() {  
        super.onCreate();  
  
        Retrofit retrofit = new Retrofit.Builder()  
            .baseUrl(WEATHER_BASE_URL)  
            .addConverterFactory(GsonConverterFactory.create())  
            .addCallAdapterFactory(RxJava2CallAdapterFactory.create())  
            .build();  
  
        mWeatherApi = retrofit.create(WeatherApi.class);  
    }  
}
```

jsonschema2pojo

jsonschema2pojo

[Donate](#) [Why?](#) [Star](#) 3,228 [Share](#) [Tweet](#)

Generate Plain Old Java Objects from JSON or JSON-Schema.

```
1 {
2   "city": {
3     "id": 524901,
4     "name": "Moscow",
5     "coord": {
6       "lon": 37.6156,
7       "lat": 55.7522
8     },
9     "country": "RU",
10    "population": 0
11  },
12  "cod": "200",
13  "message": 0.0226914,
14  "cnt": 14,
15  "list": [
16    {
17      "dt": 1507453200,
18      "temp": {
19        "day": 13.95,
20        "min": 8.26,
21        "max": 14.31,
22        "night": 8.26,
23        "eve": 11.77,
24        "morn": 11
25      },
26      "pressure": 1011.45,
27      "humidity": 98,
28      "weather": [
29        {
30          "id": 500,
31          "main": "Rain",
32          "description": "light rain",
33          "icon": "10d"
34        }
35      ],
36      "speed": 2.69,
37      "deg": 184,
38      "clouds": 0,
39      "rain": 1.29
40    },
41    {
42      "dt": 1507539600,
43      "temp": {
44        "day": 12.05,
45        "min": 11.15,
46        "max": 12.76,
47        "night": 12.14,
48        "eve": 12.17,
49        "morn": 11.15
50      },
51      "pressure": 1001.18,
52      "humidity": 93,
53      "weather": [
54        {
55          "id": 501,
56          "main": "Rain"
```

Package

Class name

Target language:
☒ Java ☐ Scala

Source type:
☐ JSON Schema ☒ JSON
☐ YAML Schema ☐ YAML

Annotation style:
☐ Jackson 2.x ☐ Jackson 1.x
☒ Gson ☐ Moshi ☐ None

☐ Generate builder methods

☐ Use primitive types

☐ Use long integers

☒ Use double numbers

☐ Use Joda dates

☐ Use Commons-Lang3

☒ Include getters and setters

☐ Include constructors

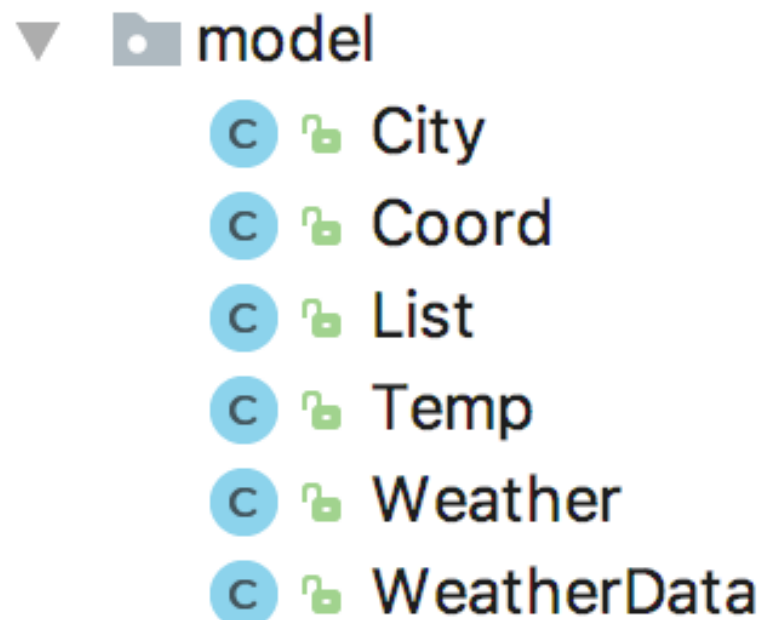
☐ Include and

☒ Include

☐ Include JSR-303 annotations

☐ Allow additional properties

Как это выглядит в проекте



model

- City
- Coord
- List
- Temp
- Weather
- WeatherData



```
1 package ru.belogurovdev.lab6.model;
2
3 import ...
4
5
6 public class Temp {
7
8     @SerializedName("day")
9     @Expose
10    private Double day;
11    @SerializedName("min")
12    @Expose
13    private Double min;
14    @SerializedName("max")
15    @Expose
16    private Double max;
17    @SerializedName("night")
18    @Expose
19    private Double night;
20    @SerializedName("eve")
21    @Expose
22    private Double eve;
23    @SerializedName("morn")
24    @Expose
25    private Double morn;
26
27    public Double getDay() { return day; }
28
29
30    public void setDay(Double day) { this.day = day; }
31
32
33    public Double getMin() { return min; }
34
35
36    public void setMin(Double min) { this.min = min; }
37
38
39    public Double getMax() { return max; }
40
41
42    public void setMax(Double max) { this.max = max; }
43
44
45    public Double getNight() { return night; }
46
47
48    public void setNight(Double night) { this.night = night; }
49
50
51
52
53
54
55
56
57
58
```


RxJava 2

```
Observable<WeatherData> weatherData = mWeatherApi.getWeatherByCity( city: "Moscow", count: "5", KEY);
weatherData
    .subscribeOn(Schedulers.io())                // Subscribe выполняется в отдельном потоке
    .observeOn(AndroidSchedulers.mainThread()) // Observe выполняется в главном потоке
    .subscribe(new Observer<WeatherData>() {
        @Override
        public void onSubscribe(Disposable d) {

        }

        @Override
        public void onNext(WeatherData weatherData) {

        }

        @Override
        public void onError(Throwable e) {

        }

        @Override
        public void onComplete() {

        }
    });
```

RxJava 2 + Java 8

```
Observable<WeatherData> weatherData = mWeatherApi.getWeatherByCity( city: "Moscow", count: "5", KEY);
weatherData
    .subscribeOn(Schedulers.io())           // Subscribe выполняется в отдельном потоке
    .observeOn(AndroidSchedulers.mainThread()) // Observe выполняется в главном потоке
    .subscribe(
        weather -> {
            hideLoad();
            mAdapter.setWeatherList(weather.getList());
        },
        error -> {
            hideLoad();
            Toast.makeText( context: this, error.getMessage(), Toast.LENGTH_SHORT).show();
        }
    );
```

ЗАВИСИМОСТИ

```
// Retrofit
implementation 'com.squareup.retrofit2:retrofit:2.3.0'
implementation 'com.squareup.retrofit2:converter-gson:2.3.0'

// RxJava 2
implementation 'io.reactivex.rxjava2:rxjava:2.1.5'
implementation 'io.reactivex.rxjava2:rxandroid:2.0.1'
implementation 'com.squareup.retrofit2:adapter-rxjava2:2.3.0'
```

Проект

Ссылка на Github
(кликабельно)

**Спасибо за
внимание!**