**Android reactive programming with RxJava. Programowanie reaktywne – co to jest, przykłady implementacji RxJava w Androidzie.**

**Michał Parysz**

**Anna Wyrwał**

**How to start with RxJava + RxAndroid**

1. Add proper dependencies to build.gradle

**compile 'io.reactivex.rxjava2:rxandroid:2.0.1'  
 compile 'io.reactivex.rxjava2:rxjava:2.0.8'**

apply plugin: 'com.android.application'  
  
android {  
 compileSdkVersion 27  
 defaultConfig {  
 applicationId "com.example.anwyr1.rxjavasample"  
 minSdkVersion 21  
 targetSdkVersion 27  
 versionCode 1  
 versionName "1.0"  
 testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"  
 }  
 buildTypes {  
 release {  
 minifyEnabled false  
 proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'  
 }  
 }  
}  
  
dependencies {  
 **compile 'io.reactivex.rxjava2:rxandroid:2.0.1'  
 compile 'io.reactivex.rxjava2:rxjava:2.0.8'**  
 implementation fileTree(dir: 'libs', include: ['\*.jar'])  
 implementation 'com.android.support:appcompat-v7:27.1.1'  
 implementation 'com.android.support.constraint:constraint-layout:1.1.0'  
 implementation 'com.android.support:design:27.1.1'  
 testImplementation 'junit:junit:4.12'  
 androidTestImplementation 'com.android.support.test:runner:1.0.2'  
 androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'  
}

1. Add support for Java 1.8 (support for lambda expressions)

compileOptions {  
 targetCompatibility 1.8  
 sourceCompatibility 1.8  
 }

android {  
 compileSdkVersion 27  
 defaultConfig {  
 applicationId "com.example.anwyr1.rxjavasample"  
 minSdkVersion 21  
 targetSdkVersion 27  
 versionCode 1  
 versionName "1.0"  
 testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"  
 }  
 buildTypes {  
 release {  
 minifyEnabled false  
 proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'  
 }  
 }  
  
 **compileOptions {  
 targetCompatibility 1.8  
 sourceCompatibility 1.8  
 }**  
   
}

1. Create function that returns List of Strings, e.g.

private List<String> getStrings() {  
 ArrayList<String> strings = new ArrayList<>();  
 for (int i = 0 ; i < 10; ++i) {  
 strings.add("Hello world, that's RxJava in use! Generated string nr: " + i);  
 }  
 return strings;  
}

1. Create two textviews e.g. with ids TextView1 and TextView2
2. Declare created text views in android code

TextView textView1 = findViewById(R.id.*textView1*);  
TextView textView2 = findViewById(R.id.*textView2*);

1. Create observable that takes items from created function

Observable<String> observable = Observable.*fromIterable*(getStrings());

1. Tell the created observable to take string items with 1 second delays (using (s -> Observable.*just*(s).delay(1, TimeUnit.*SECONDS*))). Subscribe the observable with:
   1. textView1::setText – for each emitted element set the textView1 content to its value
   2. System.*err*::println – in case of any errors display them
   3. () -> textView2.setText(R.string.*RxJavaSuccess* – when emitting elements is finished, set the textView2 content to “All strings emitted”

observable.concatMap(s -> Observable.*just*(s).delay(1, TimeUnit.*SECONDS*))  
 .subscribe(textView1::setText, System.*err*::println, () -> textView2.setText(R.string.*RxJavaSuccess*));

That’s it! Your first app which use Reactive Programming is READY!

You can find ready sources for that tutorial under:

<https://github.com/Anna9701/RxJavaSample>