

IMMOBILIEN
SCOUT24

RxJava & Coroutines

A Practical Analysis

Droidcon Lisbon 



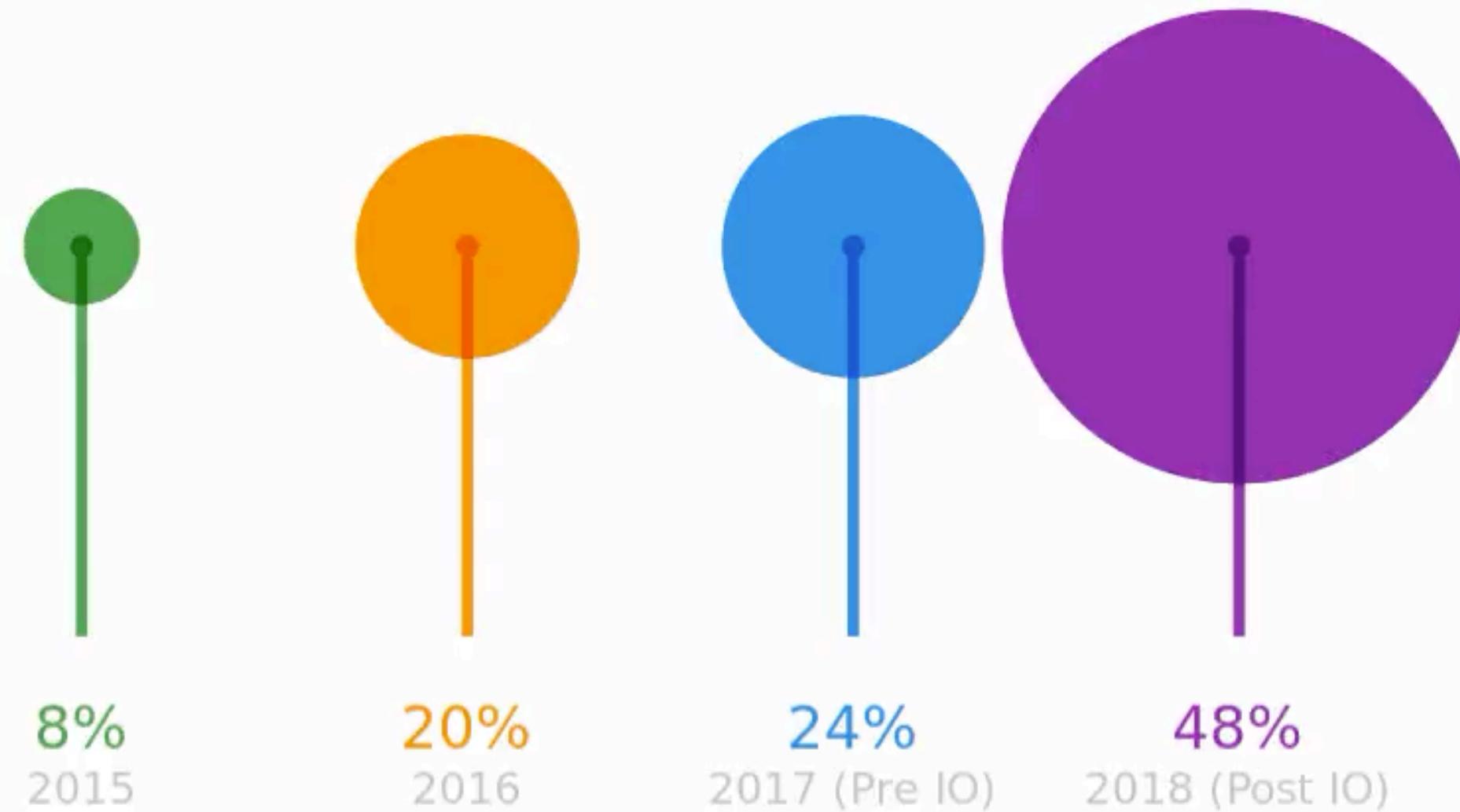
@askashdavies

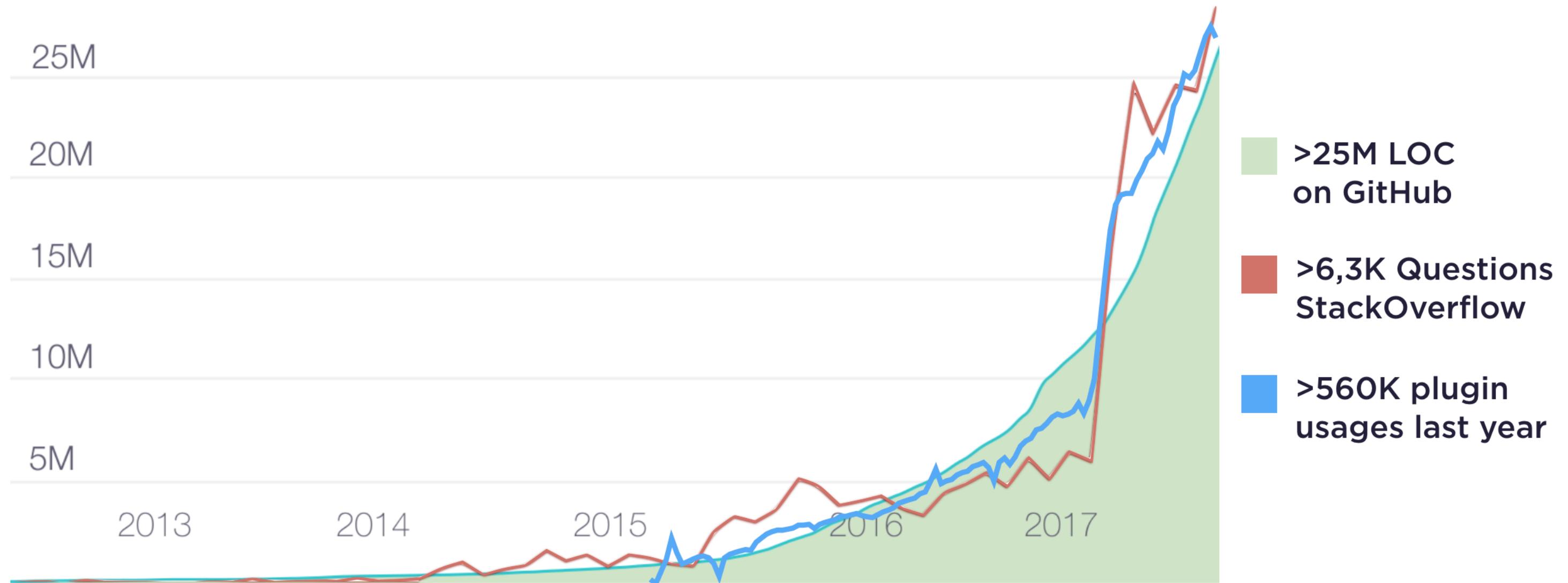






Kotlin Adoption (2018)







Coroutines

```
fun main() {  
    GlobalScope.launch {  
        delay(1000L)  
        println("World!")  
    }  
    println("Hello,")  
    Thread.sleep(2000L)  
}
```

```
// Hello,  
// World!
```

```
fun main() {  
    GlobalScope.launch {  
        delay(1000L)  
        println("World!")  
    }  
    println("Hello,")  
    Thread.sleep(2000L)  
}
```

```
// Hello,  
// World!
```

```
fun main() {  
    GlobalScope.launch {  
        delay(1000L)  
        println("World!")  
    }  
    println("Hello,")  
    Thread.sleep(2000L)  
}
```

```
// Hello,  
// World!
```

Coroutine Builders

Coroutine Builders

```
val deferred: Deferred<String> = async { "Hello World!" }
```

Coroutine Builders

```
val deferred: Deferred<String> = async { "Hello World!" }  
val result: String = deferred.await()
```

Coroutine Builders

```
val deferred: Deferred<String> = async { "Hello World!" }  
val result: String = deferred.await()  
  
val job: Job = launch { "Hello World!" }
```

Coroutine Builders

```
val deferred: Deferred<String> = async { "Hello World!" }
```

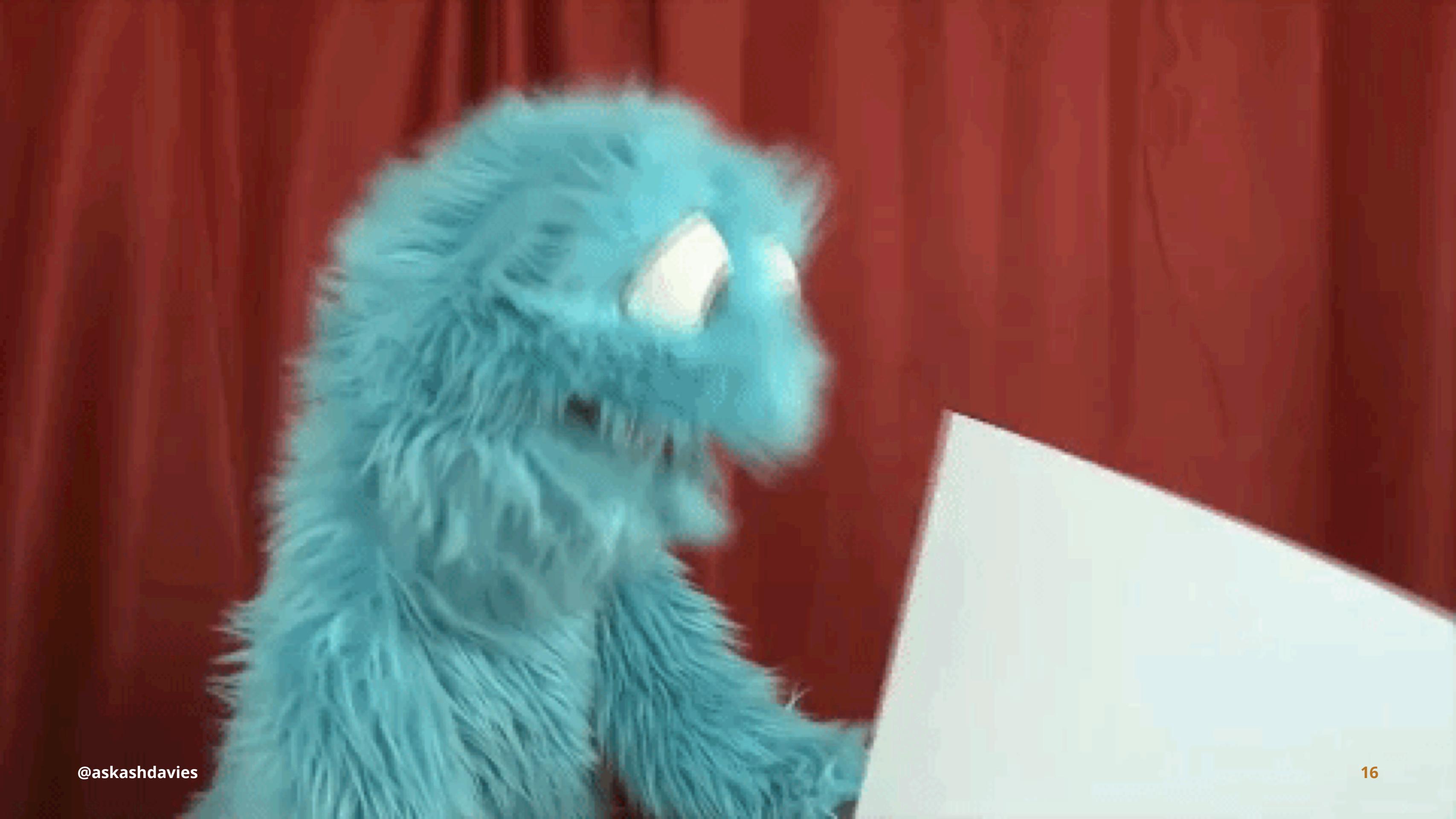
```
val result: String = deferred.await()
```

```
val job: Job = launch { "Hello World!" }
```

```
job.join()
```



Stability



@Annotations



(Here be dragons)

Annotations

@ExperimentalCoroutinesApi // !

Annotations

`@ExperimentalCoroutinesApi // !`

`@FlowPreview // !`

Annotations

`@ExperimentalCoroutinesApi // !`

`@FlowPreview // !`

`@ObsoleteCoroutinesApi // !`

Annotations

`@ExperimentalCoroutinesApi // !`

`@FlowPreview // !`

`@ObsoleteCoroutinesApi // !`

`@InternalCoroutinesApi // 💀`

Annotations @Experimental



Coroutines



Native first-party
library



Easy-to-use



suspend fun



History of Android

Background Processes

Background Processes



Runnable / Handler

Background Processes



AsyncTask

Background Processes



IntentService

Background Processes



Loader<T>

Background Processes



WorkManager



HOW STANDARDS PROLIFERATE:

(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)

SITUATION:
THERE ARE
14 COMPETING
STANDARDS.

14?! RIDICULOUS!
WE NEED TO DEVELOP
ONE UNIVERSAL STANDARD
THAT COVERS EVERYONE'S
USE CASES.



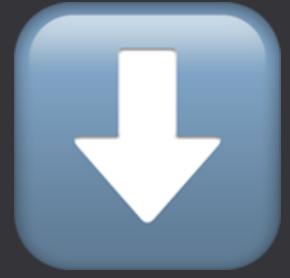
SOON:

SITUATION:
THERE ARE
15 COMPETING
STANDARDS.

RxJava to the rescue



Chained operations



Abstract threading



Reactive



RxFingerprint: Android Fingerprint Authentication and Encryption in RxJava2

RxTuples2

RxTuples2 is a library to smooth RxJava2 usage by adding simple Tuple creation functions.

For the RxJava 1.X version, please go to [RxTuples](#).

RxFingerprint wraps the Android Fingerprint APIs (introduced in Android Marshmallow) and makes it easy to:

RxMocks

[Apache License 2.0](#)

Mocks/Assertions for RxJava testing

RxPermissions

[JitPack](#) 0.10.2 [nuget](#) v0.10.2 ▾ 1.2k [build](#) passing

This library allows the usage of RxJava with the new Android M permission model.

RxLoader

[License](#) Apache 2.0 [API](#) 16+ [JCenter](#) 2.1.0

An Android Loader that wraps an RxJava Observable.

RxAndroidBle

[build](#) passing [maven-central](#) v1.7.1

RxAnimations

[Download 0.9.1](#) [API 15+](#)

RxAnimations is a library with the main goal to make android animations more solid and cohesive.

RxTuples2

RxTuples2 is a library to smooth RxJava2 usage by adding simple Tuple creation functions.

For the RxJava 1.X version, please go to [RxTuples](#).

[Android Arsenal](#) [RxPaparazzo](#)

RxJava extension for Android to take photos using the camera, select files or photos from the device and optionally crop or rotate any selected images.

RxPaparazzo

RxPermissions

[JitPack 0.10.2](#) [nuget v0.10.2](#) [build 1.2k](#) [build passing](#)

This library allows the usage of RxJava with the new Android M permission model.

RxBinding

RxJava binding APIs for Android UI widgets from the platform and support libraries.

RxAndroidBle

[build passing](#)

[maven-central v1.7.1](#)

RxFingerprint: Android Fingerprint Authentication and Encryption in RxJava2

RxFingerprint wraps the Android Fingerprint APIs (introduced in Android Marshmallow) and makes it easy to:

Rx Preferences

Reactive [SharedPreferences](#) for Android.

RxMocks

[Apache License 2.0](#)

Mocks/Assertions for RxJava testing

[downloads 245/month](#)

[Android Arsenal](#) [RxCache](#)

RxCache

[中文文档](#)

RxLoader

For a more reactive approach go [here](#).

[License Apache 2.0](#) [API 16+](#) [JCenter 2.1.0](#)

An Android Loader that wraps an RxJava Observable.

RxLifecycle

This library allows one to automatically complete sequences based on a second lifecycle stream.

This capability is useful in Android, where incomplete subscriptions can cause memory leaks.

RxAnimations

Download 0.9.1 API 15+

RxAnimations is a library with the main goal to make ar

RxPresso build passing Download 0.2.0 license unknown

Easy Espresso UI testing for Android applications using RxJava.

RxTuples2

RxTuples2 is a library to smooth RxJava2 usage by adding simple Tuple creation

For the RxJava 1.X version, please go to [RxTuples](#).

Android Arsenal RxPaparazzo

RxJava extension for Android to take photos using the camera, select files or photos from the device and optionally crop or rotate any selected images.

RxCupboard

RxCupboard brings the excellent Android [Cupboard](#) library into the world of RxJava. Using Flowable s, you can fluently store and retrieve streams of POJOs from your database.

RxPermissions

JitPack 0.10.2



nug

RxBonjour

This library allows t

build failing

A reactive wrapper around network service discovery functionalities for Kotlin and Java.

RxBinding

RxJava binding APIs for Android UI widgets from the platform and support libraries.

RxAndroidBle build passing maven-central v1.7.1

RxFingerprint: Android Fingerprint Authentication and Encryption in RxJava2

makes it easy to:

Reactive Wrapper for Java8 WatchService

RxFileWatcher allows you to observe directories (recursively or not) for file system events with a RxJava observable based on the JDK WatchService, but it is much more convenient.

Rx Preferences

Reactive SharedPreferences for Android.

RxMocks

Apache License 2.0

ocks/Assertions fo

RxFileObserver

build passing codecov unknown Download 2.0.1 Android Arsenal RxFit API 9+

downloads 245/month

Android Arsenal RxFileObserver

Android Arsenal RxCache

Reactive wrapper around Android's FileObserver

RxCache

中文文档

RxLoader

For a more reactive approach go [here](#).

License Apache 2.0 API 16+ JCenter 2.1.0

An Android Loader that wraps an RxJava Observable.

RxLifecycle

This library allows one to automatically complete sequences based on a second lifecycle stream.

This capability is useful in Android, where incomplete subscriptions can cause memory leaks.

RxAnimations

Download 0.9.1 API 15+

RxAnimations is a library with the main goal to make android anim

RxPresso build passing Download 0.2.0 license unknown

Easy Espresso UI testing for Android applications using RxJava.

RxFingerprint: Android Fingerprint Authentication and Encryption in RxJava2

RxTuples2

RxTuples2 is a library to smooth RxJava2 usage by adding simple Tuple creation function

For the RxJava 1.X version, please go to [RxTuples](#).

Android Arsenal RxPaparazzo

RxJava extension for Android to take photos using the camera, select files or photos from the device and optionally crop or rotate any selected images.

RxPaparazzo

RxPermissions

JitPack 0.10.2 nuget v0.10.2

RxBonjour

This library allows the usage

Reactive Wearable API Library for Android

build passing codecov unknown Download 1.3.0 Android Arsenal RxWear API 9+

build failing

RxMocks

Apache License 2.0

ocks/Assertions fo

RxFileObserver

build passing codecov unknown Download 2.0.1 Android Arsenal RxFit API 9+

downloads 245/month

Android Arsenal RxFileObserver

Reactive wrapper around Android's FileObserver

Android Arsenal RxCache

RxCache

中文文档

For a more reactive approach go [here](#).

RxBinding

RxJava binding APIs for Android UI widgets from the platform and support libraries.

RxLoader

License Apache 2.0 API 16+ JCenter 2.1.0

an RxJava Observable.

RxActivityResult

Relays are RxJava types which are both an Observable and a Consumer.

Android RxFirebase

RxRelay

build error coverage unknown bintray invalid license apache 2.0

RxAndroidBle

build passing maven

Basically: A Subject except without

RxLifecycle

RxJava wrapper for use with the Android Firebase client

Android RxTasks

build passing coverage unknown Download 2.0.0 license apache 2.0

library allows one to automatically complete sequences based on a second lifecycle stream.

capability is useful in Android, where incomplete subscriptions can cause memory leaks.

build passing

 Asynchronous APIs
Synchronous APIs 

To From	Flowable	Observable	Maybe	Single	Completable
Flowable		toObservable()	reduce() elementAt() firstElement() lastElement() singleElement()	scan() elementAt() first()/firstOrError() last()/lastOrError() single/singleOrError() all()/any()/count() (and more)	ignoreElements()
Observable	toFlowable()		reduce() elementAt() firstElement() lastElement() singleElement()	scan() elementAt() first()/firstOrError() last()/lastOrError() single/singleOrError() all()/any()/count() (and more)	ignoreElements()
Maybe	toFlowable()	toObservable()		toSingle() sequenceEqual()	toCompletable()
Single	toFlowable()	toObservable()	toMaybe()		toCompletable()
Completable	toFlowable()	toObservable()	toMaybe()	toSingle() toSingleDefault()	



Observable

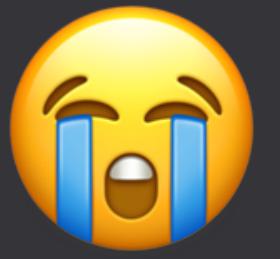
```
.fromIterable(resourceDraft.getResources())
.flatMap(resourceServiceApiClient::createUploadContainer)
.zipWith(Observable.fromIterable(resourceDraft.getResources()), Pair::create)
.flatMap(uploadResources())
.toList()
.toObservable()
.flatMapMaybe(resourceCache.getResourceCachedItem())
.defaultIfEmpty(Resource.getDefaultItem())
.flatMap(postResource(resourceId, resourceDraft.getText(), currentUser, resourceDraft.getIntent()))
.observeOn(AndroidSchedulers.mainThread())
.subscribeOn(Schedulers.io())
.subscribe(
    resource -> repository.setResource(resourceId, resource, provisionalResourceId),
    resourceUploadError(resourceId, resourceDraft, provisionalResourceId)
);
```



Anchorman: The Legend of Ron Burgundy (DreamWorks Pictures)



Hidden complexity



Hidden gotchas



Memory footprint



Steep learning curve



#RxMustDie

pca.st/7IJG

@askashdavies



*"If all you have is a hammer,
everything looks like a nail"*

— Abraham Maslow, **The Psychology of Science, 1966**

Coroutine Use Cases

Network Call Handling

```
// RxJava2: Single<T>  
fun getUser(): Single<User> = Single.fromCallable { /* ... */ }
```

```
// Coroutines: T  
suspend fun getUser(): User = /* ... */
```

Cache Retrieval

```
// RxJava2: Maybe<T>  
fun getUser(): Maybe<User> = Maybe.fromCallable { /* ... */ }
```

```
// Coroutines: T?  
suspend fun getUser(): User? = /* ... */
```

Background Task Handling

```
// RxJava2: Completable  
fun storeUser(user: User): Completable.fromCallable { /* ... */ }
```

```
// Coroutines: Unit  
suspend fun storeUser(user: User) { /* ... */ }
```

Thread Handling

```
// RxJava2  
getUser()  
    .subscribeOn(Schedulers.io())  
    .observeOn(AndroidSchedulers.mainThread())  
    .subscribe { /* Do something */ }  
  
// Coroutines  
launch(Dispatchers.Main) {  
    withContext(Dispatchers.IO) {  
        val user = getUser()  
        /* Do something */  
    }  
}
```



FlatMap

```
// RxJava2
getUser()
.flatMap { doSomethingWithUser(it) }
.subscribeOn(Schedulers.computation())
.observeOn(AndroidSchedulers.mainThread())
.subscribe { /* Do something else */ }
```

```
// Coroutines
launch(Dispatchers.Main) {
    val user = getUser()
    val smth = doSomethingWithUser(user)

    /* Do something else */
}
```

Callback Consumption

```
// RxJava2
fun getSingle(): Single = Single.create<T> { emitter ->
    doSomethingAsync(object: Callback<T> {
        override fun onComplete(result: T) = emitter.onSuccess(result)
        override fun onException(exception: Exception) = emitter.onError(exception)
    })
}

// Coroutines
suspend fun getCoroutine() : T = suspendCoroutine { continuation ->
    doSomethingAsync(object : Callback<T> {
        override fun onComplete(result: T) = continuation.resume(result)
        override fun onException(exception: Exception) = continuation.resumeWithException(exception)
    })
}
```

Task Cancellation

```
// RxJava2  
val disposable: Disposable = Single  
    .create { /* Do something */ }  
    .subscribe { /* Do something else */ }  
disposable.dispose()
```

```
// Coroutine  
val parent: Job = Job()  
launch(Dispatchers.Main + parent) { /* Do something */ }  
parent.cancelChildren()
```



Backpressure



Backpressure

```
launch {  
    channel.send("Hello")  
}
```

ViewModel.viewModelScope
androidx.lifecycle:2.1.0

Channels



Roman Elizarov

Cold flows, hot channels

```
flow {  
    emit("Hello")  
    emit("World")  
}
```



Flow



```
Observable.create { emitter: ObservableEmitter<String> ->  
    emitter.onNext("Hello")  
    emitter.onNext("World")  
}
```

Flow



```
flow { collector: FlowCollector<String> ->  
    collector.emit("Hello")  
    collector.emit("World")  
}
```

Flow



```
flow { emit("Hello world") }

.onEach { delay(500) }

.map { it.length }

.collect { println("Length: $it") }
```

Testing

```
@Test
```

```
fun testFoo() = runBlockingTest {  
    val actual = foo()  
    // ...  
}
```

```
suspend fun foo() {  
    delay(1_000)  
    // ...  
}
```



RxJava: Complex Business Logic

Asynchronicity Comparison

Manuel Vivo (@manuelvicnt)

Is Coroutines a replacement for RxJava?

Maybe...

Should I migrate to Coroutines?

Probably not...

"If it ain't broke, don't fix it"

— Bert Lance, Nation's Business, 1977



Did "you" migrate to Coroutines?



A woman in a green military uniform, including a beret and a name tag that reads "ASHLEY", stands next to a sign. The sign has large white letters on a dark background that read "HELPS WEAR".

HELPS
WEAR

How could I migrate to Coroutines?



Migration Policy

Retrofit Services

Retrofit 2.6.0

Built-in suspend support

```
interface UserService {  
  
    @GET("/user")  
    suspend fun getUser(): User  
}
```

```
GlobalScope.launch {  
  
    val user = retrofit  
        .create<UserService>() // >= 2.5.0  
        .getUser()  
}
```

Coroutines RxJava2

org.jetbrains.kotlinx:kotlinx-coroutines-rx2:+


```
val service: UserService = retrofit.create() // >= 2.5.0
```

GlobalScope

```
.rxSingle { service.getUser() }  
.observeOn(AndroidSchedulers.mainThread())  
.subscribeOn(Schedulers.io())  
.subscribe(  
    { /* Do something with user */ },  
    { /* Handle error ... probably */ }  
)
```

Completable.await()

```
GlobalScope.launch {  
    Completable  
        .complete()  
        .await()  
}
```

Maybe.await()

```
GlobalScope.launch {  
    val result: String? = Maybe  
        .fromCallable { null as String? }  
        .await()  
  
    // result == null  
}
```

Observable.await...

```
val observable = Observable.just(1, 2, 3, 4)
```

```
// Await first item
```

```
val item = observable.awaitFirst()
```

```
// Print each item
```

```
observable.consumeEach(::println)
```

```
// Consume all items
```

```
observable
    .openSubscription()
    .consume {
        println(it.size)
    }
```

Exceptions



Conclusion

IMMOBILIEN
SCOUT24

Cheers! 🍻



@askashdavies



97