IMMOBILIEN SCOUT 24

RxJava & Coroutines

A Practical Analysis

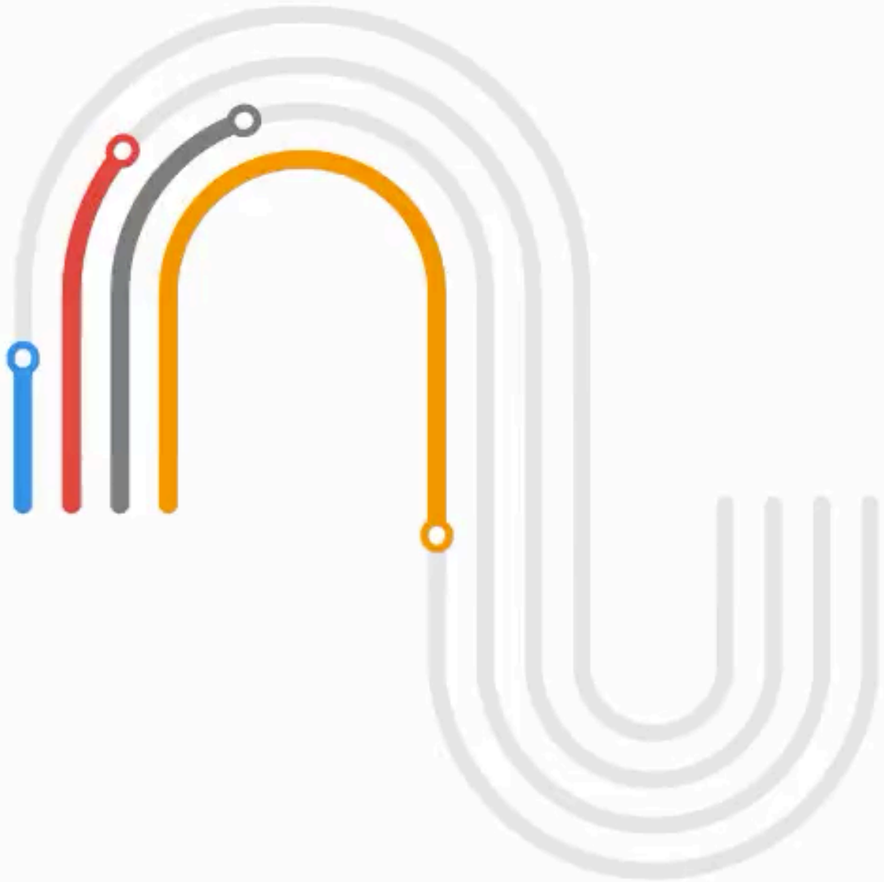
@askashdavies

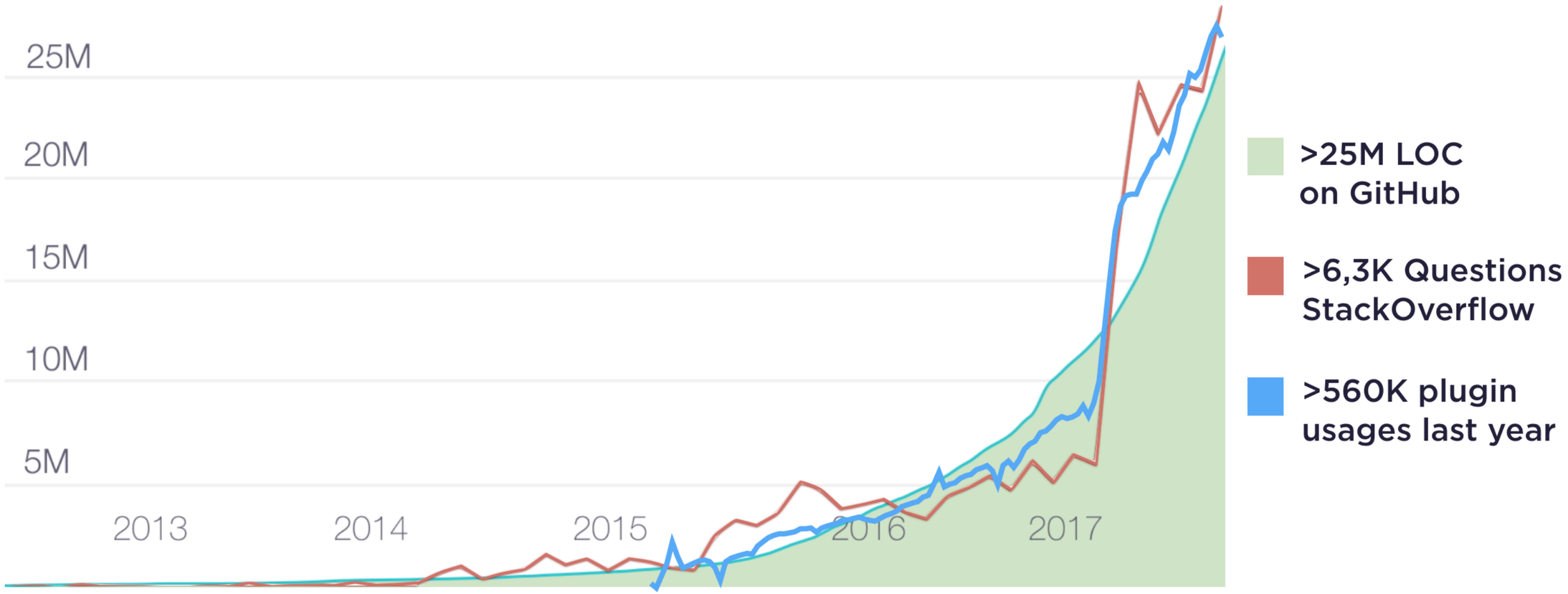
8%
2015

20%
2016

24%
2017 (Pre IO)

48%
2017 (Post IO)





blog.jetbrains.com/kotlin/2016/02/kotlin-1-0-released-pragmatic-language-for-jvm-and-android/

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()
```

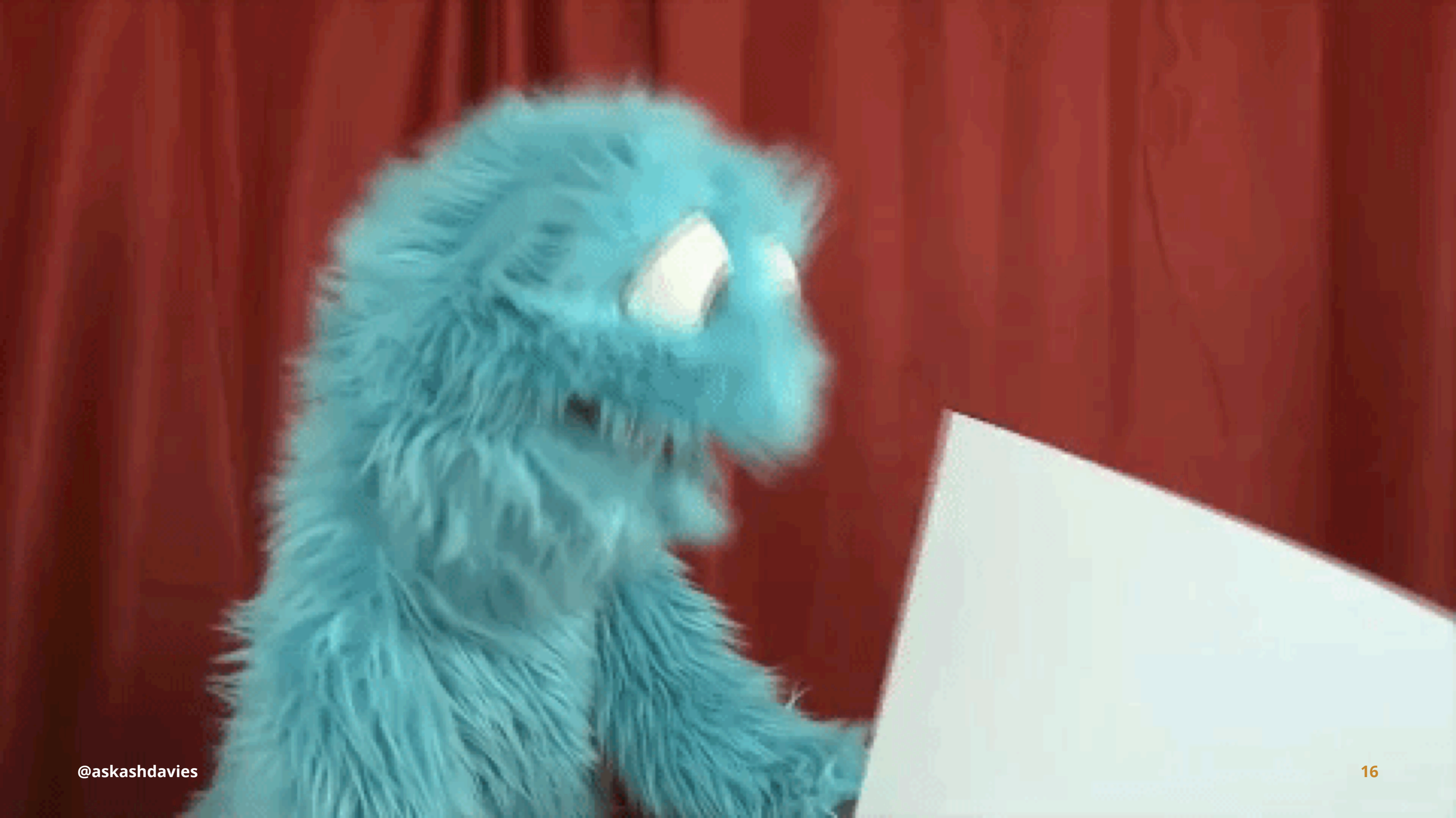
Coroutine Builders

```
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 // ⚠️
```

```
@ObsoleteCoroutinesApi // ⚠️
```

Annotations

```
@ExperimentalCoroutinesApi // ⚠️

@ObsoleteCoroutinesApi // ⚠️

@InternalCoroutinesApi // ☠️
```

💪 Coroutines

Native first-party library

😊 Easy-to-use


suspend fun

Dispatchers

History of Android

Background Processes

Background Processes

 `Runnable` / `Handler`

Background Processes

 `AsyncTask`

Background Processes

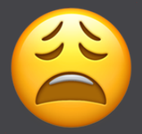
 `IntentService`

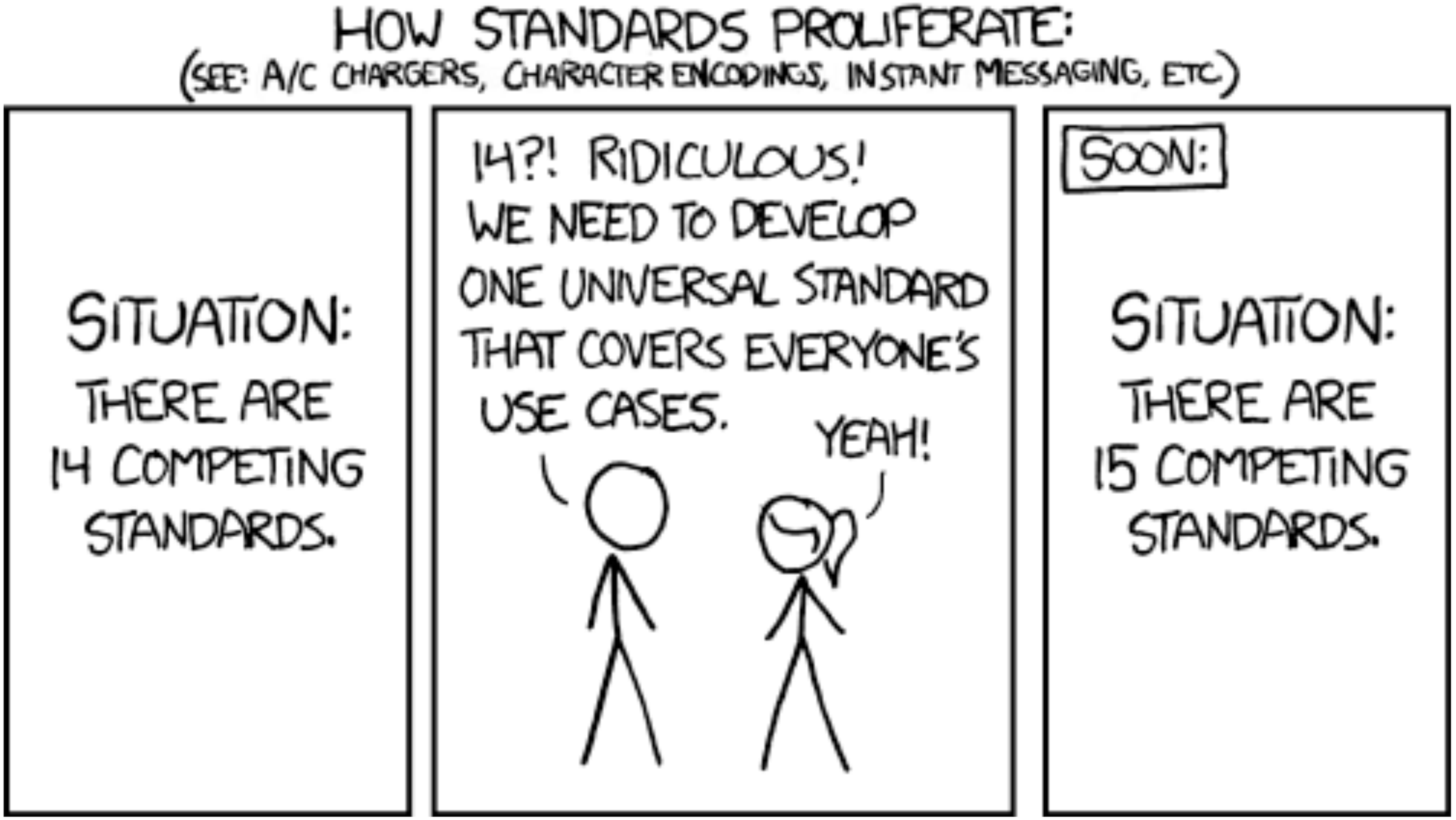
Background Processes

 `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.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

RxJava to the rescue

😮 Reactive

RxFingerprint: Android Fingerprint Authentication and Encryption in RxJava2

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

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.

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.

RxFingerprint: Android Fingerprint Authentication and Encryption in RxJava2

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

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.

Rx Preferences

Reactive `SharedPreferences` for Android.

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.

RxMocks Apache License 2.0

Mocks/Assertions for RxJava testing

RxPaparazzo

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.

downloads 245/month

Android Arsenal RxCache

RxCache

中文文档

Android Arsenal RxActivityResult

chat on gitter

RxActivityResult

RxBinding

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

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.

RxAndroidBle build passing maven-central v1.7.1

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.

RxFingerprint: Android Fingerprint Authentication and Encryption in RxJava2

RxTuples2

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

For the RxJava 1.X version, please go to RxTuples.

Reactive Wrapper for Java8 WatchService

nakes it easy to:

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

Rx Preferences

Reactive `SharedPreferences` for Android.

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.

RxMocks

Apache License 2.0

ocks/Assertions fo

Reactive Fit API Library for Android

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

RxPaparazzo

RxCupboard

RxCupboard brings the excellent Android Cupboard library into the world of RxJava. Usin `Flowable` s, you can fluently store and retrieve streams of POJOs from your database.

RxPermissions

RxFileObserver

Android Arsenal RxFileObserver

Reactive wrapper around Android's FileObserver

ownloads 245/month

ndroid Arsenal RxCache

JitPack 0.10.2 nug

This library allows tl

RxBonjour

RxActivityResult

build failing

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

RxCache

中文文档

For a more reactive approach go here.

RxBinding

RxActivityResult

RxLoader

License Apache 2.0 API 16+ JCenter 2.1.0

An Android Loader that wraps an RxJava Observable.

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

RxAndroidBle

build passing maven-central v1.7.1

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 anin

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 functio

For the RxJava 1.X version, please go to RxTuples.

Reactive Wrapper for Java8 WatchService

nakes it easy to:

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

RxPalette

RxJava binding APIs for the Palette Android library.

Rx Preferences

Reactive `SharedPreferences` for Android.

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.

RxMocks

Apache License 2.0

RxPaparazzo

RxCupboard

RxCupboard brings the excellent Android Cupboard library into the world of RxJava. Usin `Flowable` s, you can fluently store and retrieve streams of POJOs from your database.

ocks/Assertions fo

Reactive Fit API Library for Android

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

RxPermissions

JitPack 0.10.2 nuget v0.10.2

This library allows the usage

RxFileObserver

Android Arsenal RxFileObserver

Reactive wrapper around Android's FileObserver

downloads 245/month

Android Arsenal RxCache

RxBonjou

build failing

Reactive Wearable API Library for Android

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

Arsenal RxActivityResult

gitter

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

RxCache

中文文档

For a more reactive approach go here.

RxLoader

License Apache 2.0 API 16+ JCenter 2.1.0

RxBinding

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

RxActivityResult

RxRelay

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

Basically: A `Subject` except without

an RxJava Observable.

Android RxFirebase

build error coverage unknown bintray invalid license apache 2.0

RxJava wrapper for use with the Android Firebase client

RxAndroidBle

build passing maven

RxLifecycle

Android RxTasks

build passing coverage unknown Download 2.0.0 license apache 2.0

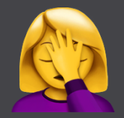
ibrary allows one to automatically complete sequences based on a second lifecycle stream.

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


build
passing

👍 Asynchronous APIs
Synchronous APIs 👎

From \ To	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

"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 */
  }
}
```

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()
```

ViewModel.viewModelScope

androidx.lifecycle:2.1.0-alpha**

Channels 

Roman Elizarov

Cold flows, hot channels

```
Flow.collect {
   it.emit("Hello World!")
}
```

medium.com/@elizarov/cold-flows-hot-channels-d74769805f9

RxJava: Complex Business Logic

Asynchronicity Comparison

Manuel Vivo (@manuelvicnt)

Is Coroutines a replacement for RxJava?

Maybe...

Not really...

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?

 Yes!

How could I migrate to Coroutines?

🤝 Migration Policy

Retrofit Services

Retrofit2 Coroutines Adapter

```
com.jakewharton.retrofit:retrofit2-kotlin-coroutines-adapter:+
```

```
interface UserService {

  @GET("/user")
  fun getUser(): Deferred<User>
}

val retrofit = Retrofit.Builder()
    .baseUrl("https://example.com/")
    .addCallAdapterFactory(CoroutineCallAdapterFactory())
    .build()

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

```
interface UserService {

  @GET("/user")
  fun getUser(): Deferred<User>
}

val retrofit = Retrofit.Builder()
    .baseUrl("https://example.com/")
    .addCallAdapterFactory(CoroutineCallAdapterFactory())
    .build()

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

Retrofit

First-party "suspend" support

Coroutines RxJava2

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

Name	Scope	Description
`rxCompletable`	`CoroutineScope`	Cold completable that starts coroutine on subscribe
`rxMaybe`	`CoroutineScope`	Cold maybe that starts coroutine on subscribe
`rxSingle`	`CoroutineScope`	Cold single that starts coroutine on subscribe
`rxObservable`	`ProducerScope`	Cold observable that starts coroutine on subscribe
`rxFlowable`	`ProducerScope`	Cold observable that starts coroutine on subscribe with **backpressure** support

```
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)
  }
```

kotlin.github.io/kotlinx.coroutines/kotlinx-coroutines-rx2/kotlinx.coroutines.rx2/io.reactivex.-observable-source/index.html

Exceptions?

Exceptions

```
observable.subscribe(
  onSuccess = { it: T -> },
  onError = { it: Throwable -> } // Gotta catch em all!
)

launch {
  try {
    doSomethingDangerous()
  } catch(exception: DangerousException) {
    // Specific exception caught!
  }
}
```

Exceptions

```
val handler = CoroutineExceptionHandler { _, exception ->
  println("Caught $exception")
}

val job = GlobalScope.launch(handler) {
  throw AssertionError()
}

join(job)

// Caught java.lang.AssertionError
```

😅 Conclusion

IMMOBILIEN SCOUT 24

Cheers! 🍻

bit.ly/rxjava-coroutines-prague