IMMOBILIEN SCOUT 24

Refactoring Legacy Code with Kotlin & Coroutines

Kotlin Everywhere: Hamburg

@askashdavies

Java 

Streams ⇝


SO?
@askashdavies

Kotlin 

Null 

```
if (foo == null) {
  bar();
}
```

Offensive Code < 🖕 >

Urgency 🚨

Kotlin?

Kotlin 💪

Google IO 2018 🥇

Google IO 2019 🥇 🎖️

Kotlin

@askashdavies

Libraries

Anko

RxKotlin

Kotlin Extensions for RxJava

RxDownload

kotlinx.coroutines

JB official | Download 0.10.8 | build passing | license Apache License 2.0

Ktor

Anko

RxKotlin

KAndroid

JB official | latest version v1.3.50

Kotlin Extensions for RxJava

Kotlin/Native

Kotter Knife

RxDownload

language kotlin | RxJava 2.0

kotlinx.coroutines

JB official | license Apache License 2.0 | Download 1.3.0

kotlin-core

This p
take a

JB official | Download 0.10.8 | build passing | license Apache License 2.0

Gradle Kotlin DSL Samples

build passing | licen

Ktor

official | Download 1.2.3 | TeamCity Build | license Apache License 2.0

Anko

Download 0.8.8 | build passing | issue resolution 9 d | Android Arsenal KAndroid | AndroidWeekly #148

RxKotlin

KAndroid

JB official | latest version v1.3.50

Kotlin Extensions for RxJava

Kotlin/Native

kotlin 1.0.2 | maven-central v2.0.0-ALPHA-03 | build passing | issues 13 open | license MIT | chat kotlin slack

Notice: Kodein and Injekt, much of the same

Kotter Knife

RxDownload

language kotlin | RxJava 2.0

kotlinx.coroutines

JB official | license Apache License 2.0 | Download 1.3.0

kotlin 1.3.10 | maven-central v2.6.0 | build passing | issues 1 open | license MIT | chat kotlin slack #kohesive

klutter


kotlin-core
Gradle Kotlin DSL Samples
build passing
This p
take a
JB official Download 0.10.8 build passing license Apache License 2.0
Ktor
Ankc
KotlinTest
Download 0.8.8 build passing issue resolution
KAndroid
JB official latest version v1.3.50
Kotlin Extensions for RxJava
Kotter Knife
Kotlin/Native
kotlin 1.0.2 maven-central v2.0.0-ALPHA-03 build passing issues 13 open license MIT chat kotlin slack
Notice:
Download 2.3.0 Android Arsenal LastAdapter License Apache 2.0 chat on gitter
RxDownloac
LastAdapter
Kanary
languag
powered by vok build passing gitter join chat tag v0.7.1 maven central 0.7.1
Welcome to Vaadin-On-Kotlin
kotlin 1.3.10 maven-central v2.6.0 build passing issues 1 open license MIT chat kotlin slack #kotlinslack
KANARY
klutter


build
passing


@askashdavies

Kotlin 🤔

Idiomacy


@askashdavies

Idiomatic Code

Idiomatic Code

- Consistent, easier to read
- Less cognitive load
- Less ambiguity
- Function > Style

Code Style

- kotlinlang.org/docs/reference/coding-conventions.html
- android.github.io/kotlin-guides/style.html

Refactoring Legacy Code


^⌥⇧K


US FIGURE SKATING
@askashdavies


@askashdavies

```
public class BadJavaActivity extends Activity {

  @Inject Dependency dependency;
}
```

General Assumptions 👮

```
class BadKotlinActivity : Activity() {

  @Inject var dependency: Dependency? = null
}
```

```
class SlightlyLessBadKotlinActivity : Activity() {

  @Inject internal lateinit var dependency: Dependency
}
```

UninitializedPropertyAccessException!

```
lateinit var file: File

if (::file.isInitialized) { /* ... */ }
```

Data Classes

```java
public class User {

  private String firstName;
  private String lastName;

  public User(String firstName, String lastName) {
    this.firstName = firstName;
    this.lastName = lastName;
  }

  public String getFirstName() {
    return firstName;
  }

  public void setFirstName(String firstName) {
    this.firstName = firstName;
  }

  public String getLastName() {
    return lastName;
  }

  public void setLastName(String lastName) {
    this.lastName = lastName;
  }

  @Override
  public boolean equals(Object o) {
    if (this == o) {
      return true;
    }
    if (o == null || getClass() != o.getClass()) {
      return false;
    }
    User user = (User) o;
    return Objects.equals(firstName, user.firstName) &&
        Objects.equals(lastName, user.lastName);
  }

  @Override
  public int hashCode() {
    return Objects.hash(firstName, lastName);
  }
}
```

```
class User(var firstName: String?, var lastName: String?) {

  override fun equals(o: Any?): Boolean {
    if (this === o) {
      return true
    }
    if (o == null || javaClass != o.javaClass) {
      return false
    }
    val user = o as User?
    return firstName == user!!.firstName && lastName == user.lastName
  }

  override fun hashCode(): Int {
    return Objects.hash(firstName, lastName)
  }
}
```

```
data class User(val firstName: String, val lastName: String)
```

```
data class User(val firstName: String, val lastName: String? = null)
```

```
@NotNull
public final User copy(@Nullable String firstName, @Nullable String lastName) {
  return new User(firstName, lastName);
}
```

Kotlin

- Singleton objects
- String interpolation
- Elvis operator 🦹
- Destructuring
- Extension functions
- Scoping functions

Maintaining History

Maintaining History

- Change extension `.java` -> `.kt`
- First commit
- Apply Kotlin conversion
- Second commit

Maintaining History

Refactoring

Refactoring

Refactoring

SOLID

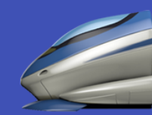
- Single-responsibility
- Open-closed
- Liskov substitution
- Interface segregation
- Dependency inversion

Refactoring

Single Responsibility

Perspective 

Asynchronicity

Concurrency is Hard 

Thread

```
new Thread(() -> {
  foo();
}).start();
```

CompletableFuture

```
CompletableFuture.supplyAsync(this::findSomeData)
   .thenApply(this:: intReturningMethod)
   .thenAccept(this::notify);
```

RxJava

```
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)
  );
```

@askashdavies

What If? 🤔

Coroutines

@askashdavies

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

// Hello,
// World!
```

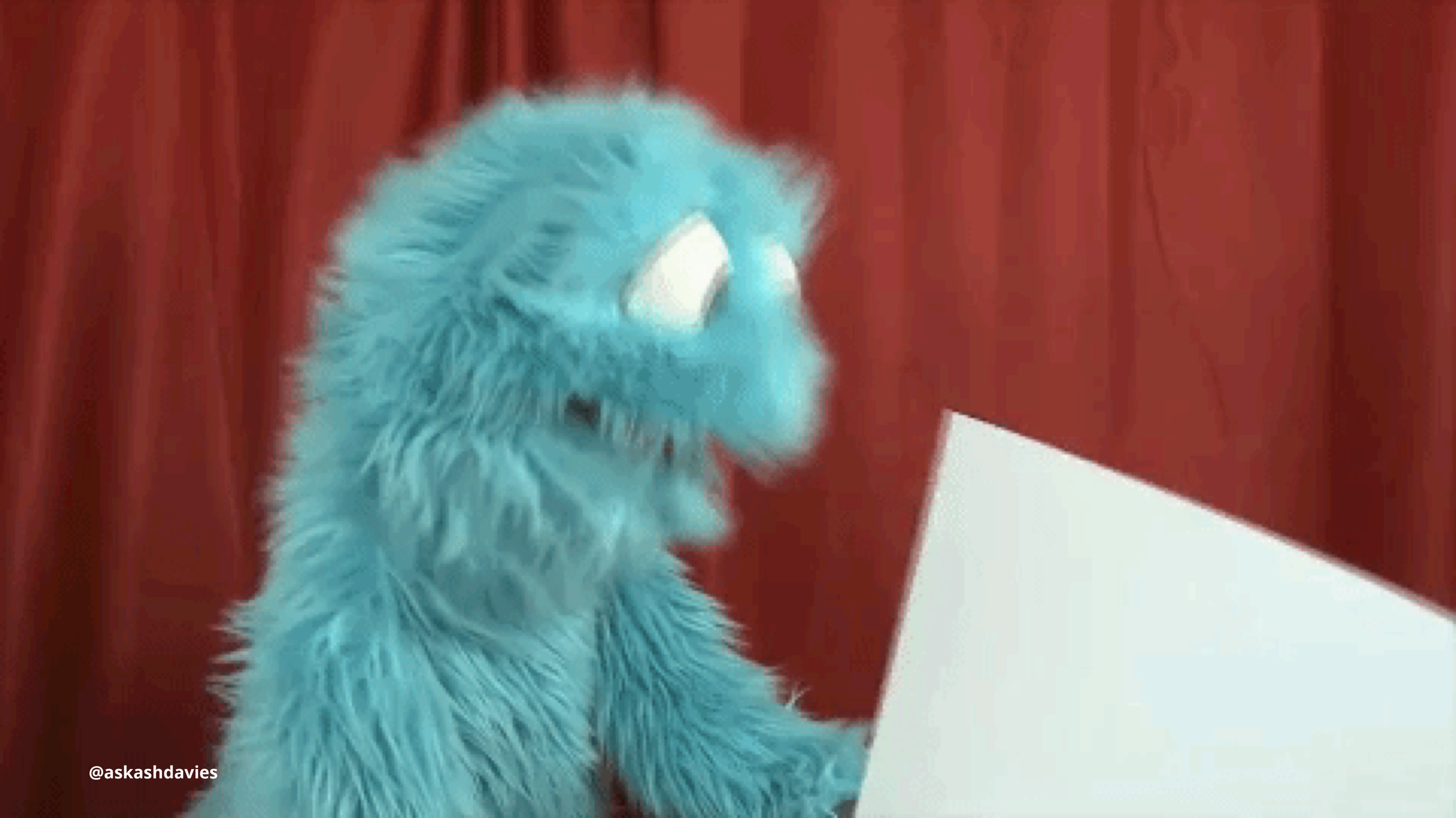
```
fun main() {
    GlobalScope.launch {
        delay(1000L)
        println("World!")
    }
    println("Hello,")
    Thread.sleep(2000L)
}

// Hello,
// World!
```

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

// Hello,
// World!
```

Stability


@askashdavies

@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

Efficient 

😊 Easy-to-use

👌 suspend fun

👌 Suspend

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

// Hello,
// World!
```

👌 Suspend

```
fun main() {
  GlobalScope.launch {
    doWorld()
    println("Hello,")
    Thread.sleep(2000L)
  }
}

suspend fun doWorld() {
  delay(1000L)
  println("World!")
}

// Hello,
// World!
```

👌 Suspend

```
fun main() {
  GlobalScope.launch {
    doWorld()
    println("Hello,")
    Thread.sleep(2000L)
  }
}

suspend fun doWorld() {
  withContext(Dispatchers.IO) {
    delay(1000L)
    println("World!")
  }
}

// Hello,
// World!
```

Dispatchers

Dispatchers

- Default
- IO
- Main
  - Android (Main Thread Dispatcher)
  - JavaFx (Application Thread Dispatcher)
  - Swing (Event Dispatcher Thread)
- Unconfined

Testing

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

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

Thanks!

@askashdavies