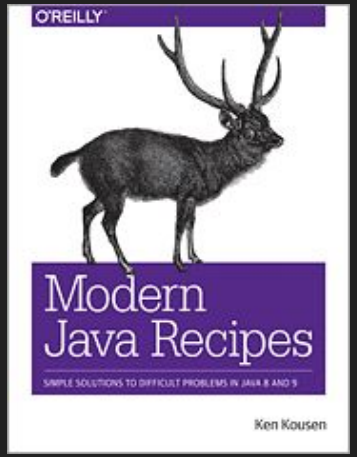
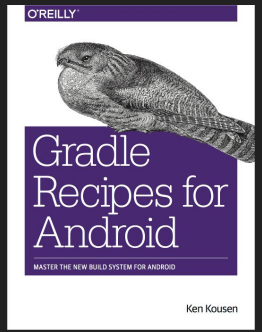
Groovy, Java, and Kotlin

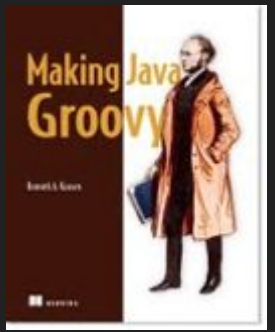
Functional Programming

Contact Info

Ken Kousen
Kousen IT, Inc.
ken.kousen@kousenit.com
http://www.kousenit.com
http://kousenit.org (blog)
@kenkousen







Videos

O'Reilly video courses: See Safari Books Online for details

- Groovy Programming Fundamentals
- Practical Groovy Programming
- Mastering Groovy Programming
- Learning Android
- Practical Android
- Gradle Fundamentals
- Gradle for Android
- Spring Framework Essentials
- Advanced Java Development

Kotlin Certified Training Partner

- Certified by JetBrains for Kotlin

Kotlin Certified Training Partner

- Certified by JetBrains for Kotlin
- (Would do the same thing for Groovy if a program existed :)

Kotlin Certified Training Partner

- Certified by JetBrains for Kotlin
- (Would do the same thing for Groovy if a program existed :)
- (Ditto for Java)

GitHub Survey 2018

https://octoverse.github.com/projects#languages

Fastest Growing Languages

1 → Kotlin

9 → Groovy

Note Java is #2 in Top Languages Over Time

Lambda Expressions

- Java lambda expressions
  - Assigned to Single Abstract Method interfaces
  - Parameter types inferred from context

Functional Programming

Lambda, Method References, and Streams

LambdaDemo.java

MapFilterReduce.java

PrimeChecker.java

StreamsDemo.java

Functional Programming

Groovy uses closures for lambdas

Stream methods are on collections

collect, findAll, inject vs map, filter, reduce

lambdas.groovy

map_filter_reduce.groovy

PrimeCheckerGroovy.groovy

Kotlin Functions

Kotlin defines functions defined with the "fun" keyword

```
fun main(args: Array<String>) { … }
```

If function consists of one statement, can use assignment

```
fun sayHello(name: String) = println("Hello, $name!")
```

(note: semicolons not needed)

Kotlin Functions

Return type shown after signature

```
fun sum(a: Int, b:Int) : Int {
    return a + b
}
```

Simpler:

```
fun sum(a: Int, b: Int) = a + b
```

Return type inferred

(Use "Unit" return type for Java "void")

Functions

Support default parameters

```
fun read(b: Array<Byte>, off: Int = 0, len: Int = b.size) {
    ...
}
```

Override defaults by supplying actual values

Lazy Streams

- Java Streams are lazy
  - Only process as much data as needed
- Groovy and Kotlin collections are not
  - But Groovy can use Java streams and Kotlin has sequences
  - LazyStreams.java
  - LazyStreamsGroovy.groovy
  - LazyStreamsSpec.groovy

Kotlin Sequences

- Methods like map, filter, reduce, and fold are added to collections
- The "`asSequence()`" method converts collection to sequence
  - Like Java streams
  - Evaluated element at a time
  - No data processed unless there is a terminal expression

POGOs vs POJOs

Plain Old Groovy Objects

- private attributes, public methods, public class
- map-based constructor
- generated getters and setters
- @Canonical → toString, equals, hashCode, tuple ctor

POJOs vs POGOs

Sorting streams with POJOs/POGOs

Golfer.java, SortGolfers.java

GroovyGolfer.groovy, sort_groovy_golfers.groovy

GroovyGolferCS.groovy

Kotlin Data Classes

Classes defined using the keyword "data"

```
data class Customer(val name: String, val email: String)
```

(That's the entire class)

Data classes have:

- generated getters and setters
- toString, equals, hashCode
- `copy()` method
- componentN() methods for destructuring

Groovy JDK

Methods added to Java to make it groovier…

Location.java

Geocoder.groovy

GeocoderTest.java

Going Beyond Java

AST transformations: @Memoize, @TailRecursive

AnnotatedFunctions.groovy

UseAnnotatedFunctions.java

AnnotatedFunctionsTest.java

composition.groovy

currying.groovy

Metaprogramming

Groovy can add methods to a class at runtime or compile time

pirate.groovy

Metaprogramming

Kotlin allows for extension methods added at compile time

Yoda.kt

Cat Pictures

Flickr RESTful (sort of) web service

https://www.flickr.com/services/api/flickr.photos.search.html

Parsing JSON

SwingBuilder

Can use Java 8 parallel streams

cat_pictures.groovy

parallel_kitties.groovy

Summary

Use Groovy and Java together
- Each does what it does well

Java functional capabilities help
- Groovy goes beyond them
- Kotlin does too, but has its own quirks

Easy to mix the languages together

All demos (and more) at: https://github.com/kousen/java_groovy_kotlin