Kotlin for Microservices, why?

Mario Fernandez

Lead Developer

ThoughtWorks

Starting point

For our new service, we are thinking of using something different than Java


Java


Scala


JRuby


Kotlin

What do you get with Kotlin

Lean

```
object CountryLanguageMatrix {
    val locales = mapOf(
            Country("de") to listOf(Language("de_DE")),
            Country("us") to listOf(Language("en_US")),
            Country("ca") to listOf(Language("en_US"), Language("fr_FR")
            Country("cn") to listOf(Language("zh_CN"), Language("en_GB")
    )
}
```

```
package my.utils

object Utils

fun String.asStream(): InputStream {
    return Utils.javaClass.classLoader.getResourceAsStream(this)
}

fun InputStream.readTextAndClose(charset: Charset = Charsets.UTF_8) =
    this.bufferedReader(charset).use { it.readText() }
```

```
import my.utils.asStream
import my.utils.readTextAndClose

val jwt = "jwt".asStream().readTextAndClose()
```

Sane defaults

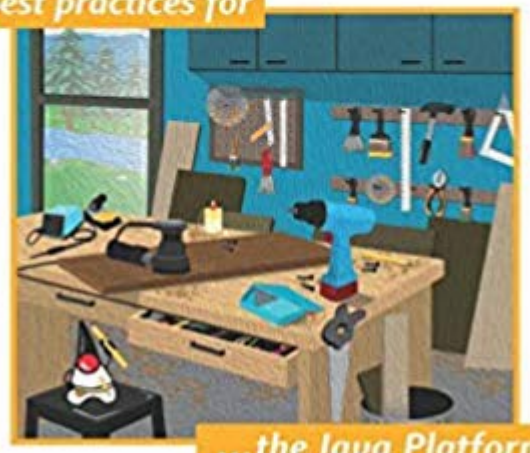
Joshua Bloch

Updated for Java 9

Effective Java

Third Edition

Best practices for



...the Java Platform

Item 15: Minimise mutability

```
data class Stuff(val value: String)

val safeToShare = listOf(
        Stuff("you"),
        Stuff("won't"),
        Stuff("change"),
        Stuff("this"))
```

Item 17: Design and document for inheritance or else prohibit it

```
class NotExtensible
class BadIdea: NotExtensible() // Error !

open class Extensible {
    open fun say() = "hi"
}

class GoodIdea: Extensible() {
    override fun say() = "hello"
}
```

Null safety

The billion dollar mistake

> I call it my billion-dollar mistake. It was the invention of the null reference in 1965. At that time, I was designing the first comprehensive type system for references in an object oriented language (ALGOL W). My goal was to ensure that all use of references should be absolutely safe, with checking performed automatically by the compiler. But I couldn't resist the temptation to put in a null reference, simply because it was so easy to implement. This has led to innumerable errors, vulnerabilities, and system crashes, which have probably caused a billion dollars of pain and damage in the last forty years.

Pitfalls

```
override fun preHandle(
  request: HttpServletRequest,
  response: HttpServletResponse,
  handler: Any): Boolean {
    val value: String = request.getHeader("Header")

    // value is technically of type String!
    // It can actually be null
}
```

```
override fun doFilterInternal(
        request: HttpServletRequest,
        response: HttpServletResponse,
        filterChain: FilterChain) {

    request.getHeader(Headers.AUTHORIZATION)?.let { header ->
        jwt(header)?.let { jwt ->
            authentication(jwt)?.let { auth ->
                val context = SecurityContextHolder.getContext()
                context.authentication = auth
            }
        }

    filterChain.doFilter(request, response)
}
```

And much more

How can I try it?

Seamless transition


spring
boot

AssertJ

```
FROM openjdk:8-jre-alpine3.9

WORKDIR /app
EXPOSE 4003

ENV ENV='dev'

RUN apk add --update --no-cache dumb-init \
  && rm -rf /var/cache/apk/*

COPY build/libs/*.jar app.jar

RUN adduser -D runner

USER runner
ENTRYPOINT ["/usr/bin/dumb-init", "--"]
CMD ["java", "-Dspring.profiles.active=${ENV}", "-jar", "app.jar" ]
```

Kotlin is a gateway drug

New libraries

STRIKT

New paradigms

Conclusion

Links

- https://hceris.com/painless-json-with-kotlin-and-jackson/
- https://hceris.com/mock-verification-in-kotlin/