Reactive programming

Lessons learned

Tomasz Nurkiewicz

You must be *this* tall to practice reactive programming

> *[…] a very particular set of skills, skills […] acquired over a very long career. Skills that make me a nightmare for people like you*

Liam Neeson on reactive programming

Who am I?

I built complex reactive systems

* not really proud about the “*complex*” part

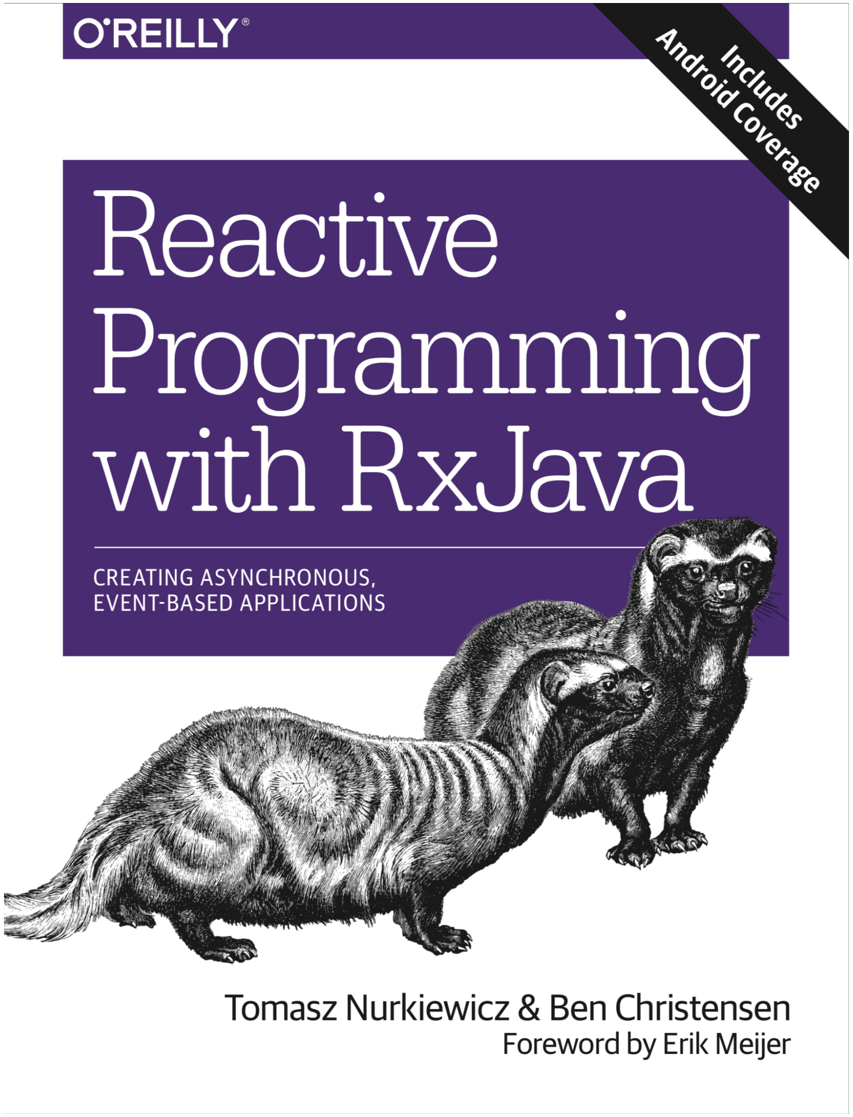
1000+ Akka cluster nodes

Tens of thousands of RPS
on a single node

Beat C10k problem

...and C100k

I wrote a book with the word "*reactive*" in the title


O'REILLY
Includes Android Coverage
Reactive Programming with RxJava
CREATING ASYNCHRONOUS, EVENT-BASED APPLICATIONS
Tomasz Nurkiewicz & Ben Christensen
Foreword by Erik Meijer

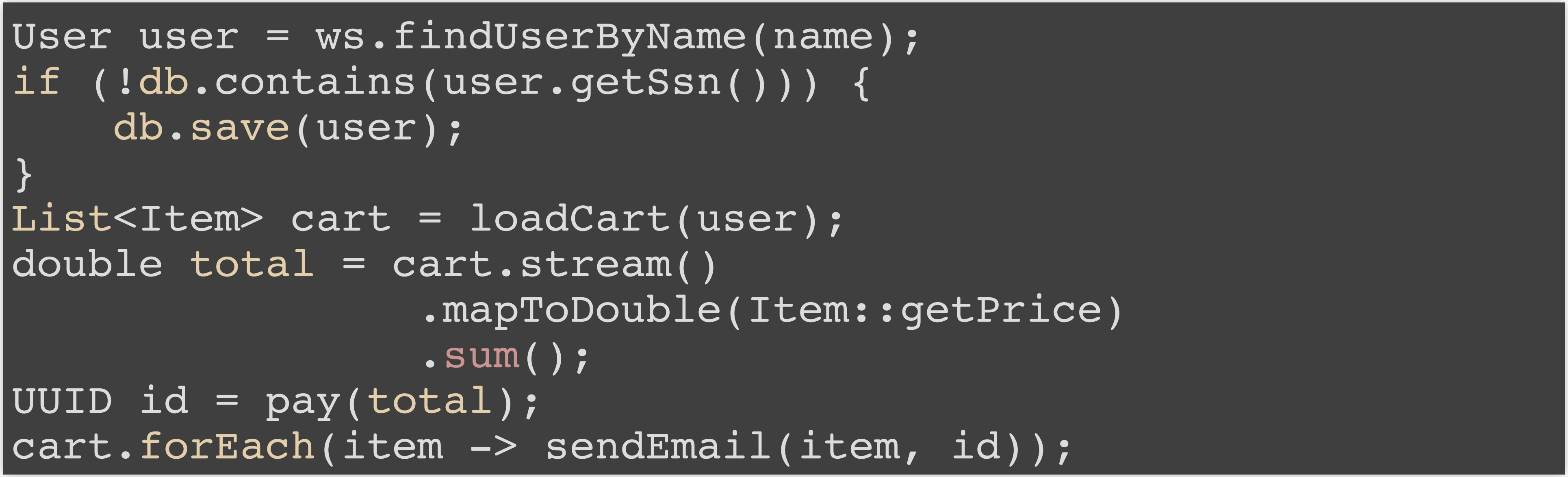
May you live in interesting times

Chinese curse

> *May you support interesting codebase*

me

1. Fetch user by name from a web service
2. If not yet in database, store it
3. Load shopping cart for user
4. Count total price of items
5. Make single payment
6. Send e-mail for each individual item, together with payment ID



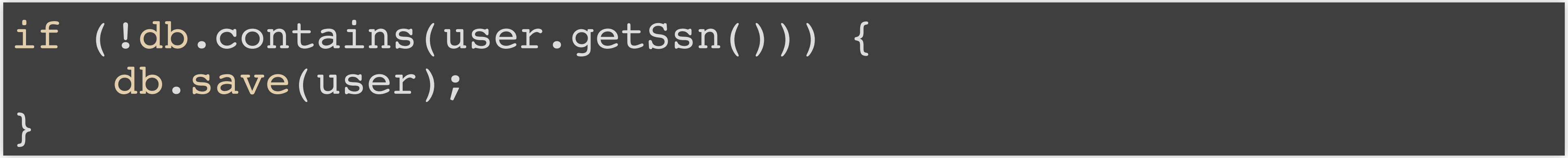
```
User user = ws.findUserByName(name);
if (!db.contains(user.getSsn())) {
    db.save(user);
}
List<Item> cart = loadCart(user);
double total = cart.stream()
                .mapToDouble(Item::getPrice)
                .sum();
UUID id = pay(total);
cart.forEach(item -> sendEmail(item, id));
```

```
     User  user = ws.findUserByName(name)
```

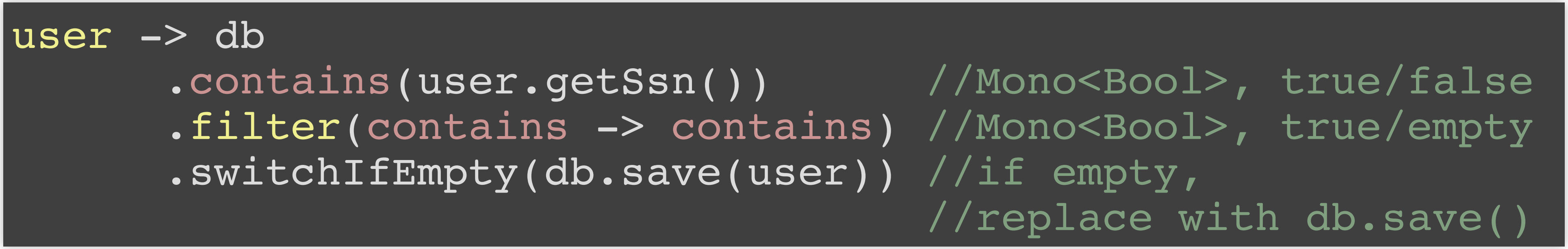
```
Mono<User> user = ws.findUserByName(name)
```

```
     boolean  contains = db.contains(user.getSsn())
```

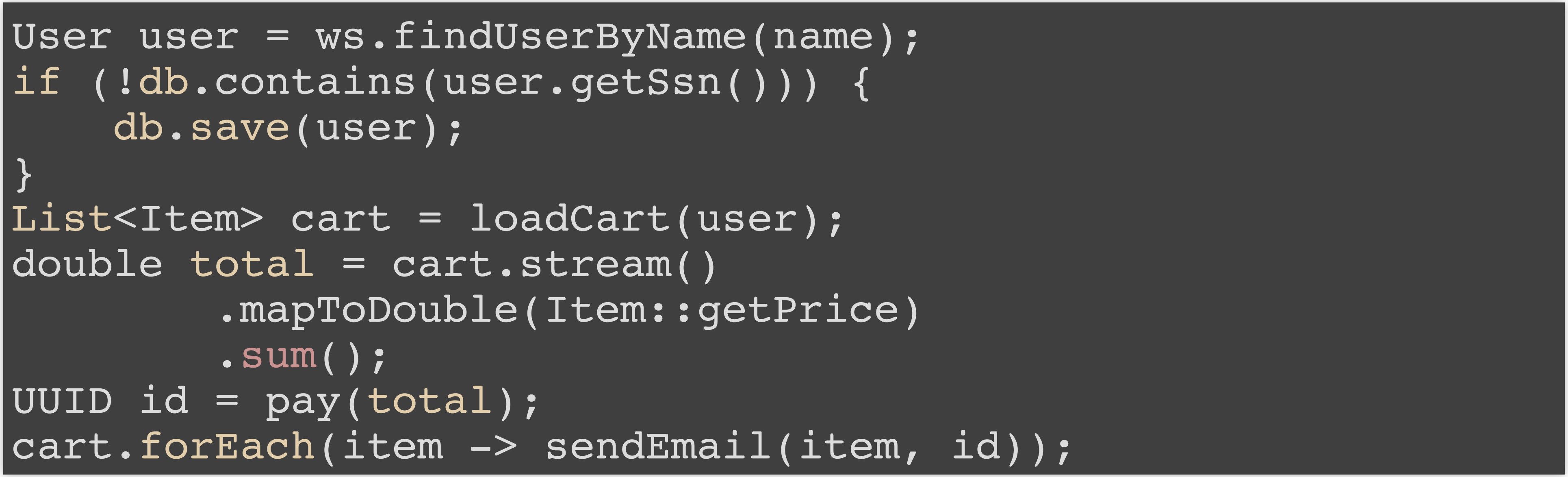
```
Mono<Boolean> contains = db.contains(user.getSsn())
```



```
if (!db.contains(user.getSsn())) {
    db.save(user);
}
```

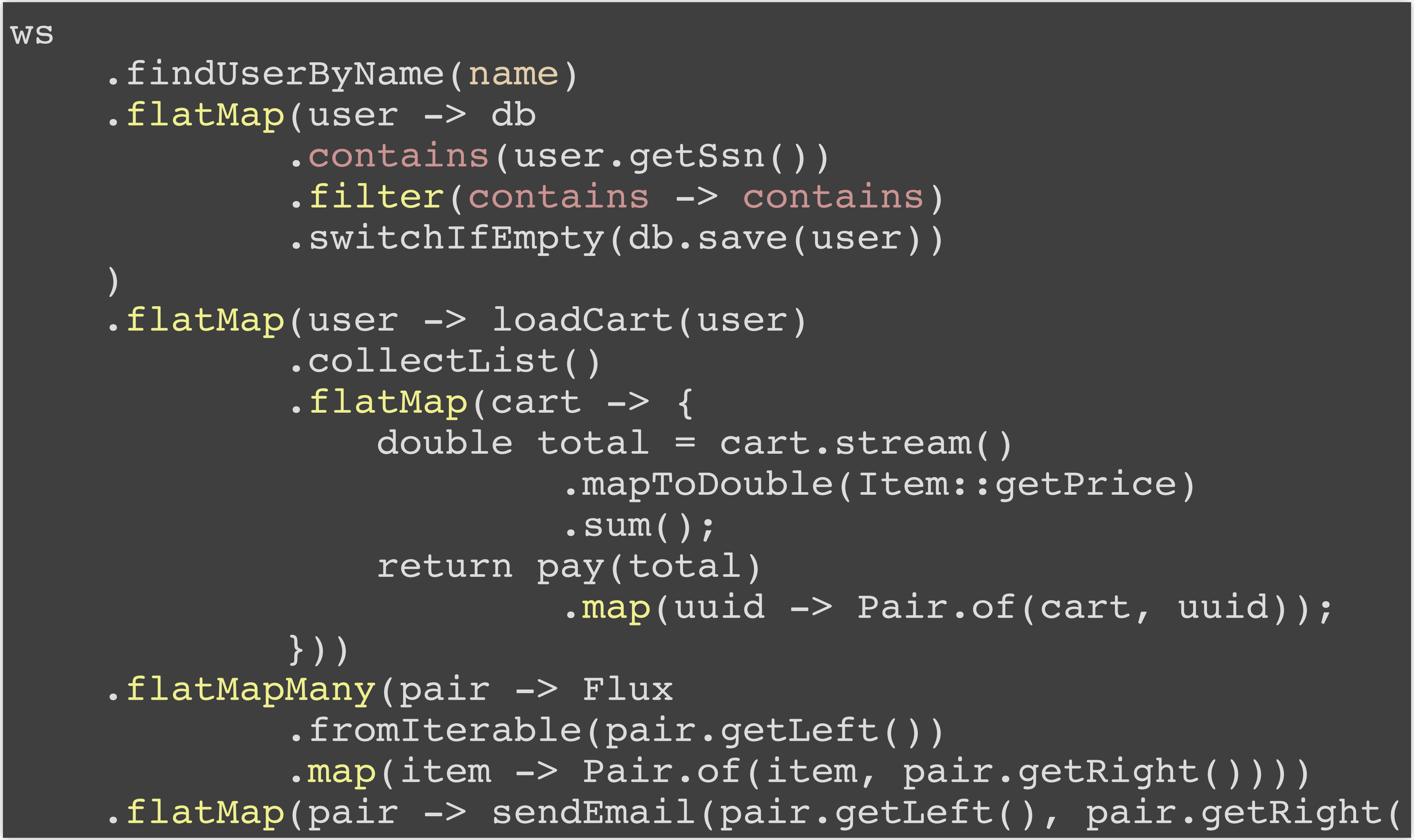


```
user -> db
      .contains(user.getSsn())      //Mono<Bool>, true/false
      .filter(contains -> contains) //Mono<Bool>, true/empty
      .switchIfEmpty(db.save(user)) //if empty,
                                    //replace with db.save()
```

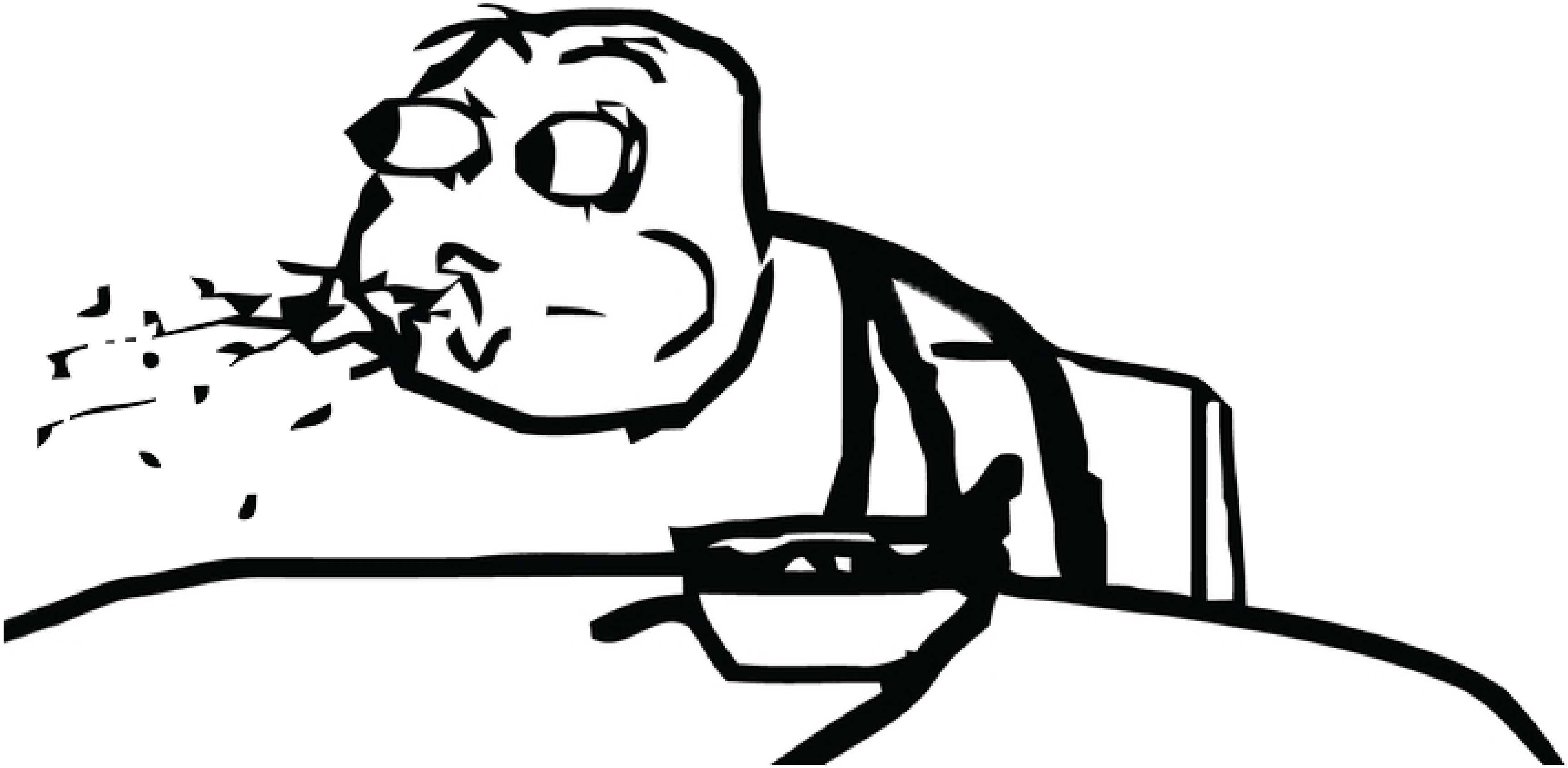


```
User user = ws.findUserByName(name);
if (!db.contains(user.getSsn())) {
    db.save(user);
}
List<Item> cart = loadCart(user);
double total = cart.stream()
        .mapToDouble(Item::getPrice)
        .sum();
UUID id = pay(total);
cart.forEach(item -> sendEmail(item, id));
```

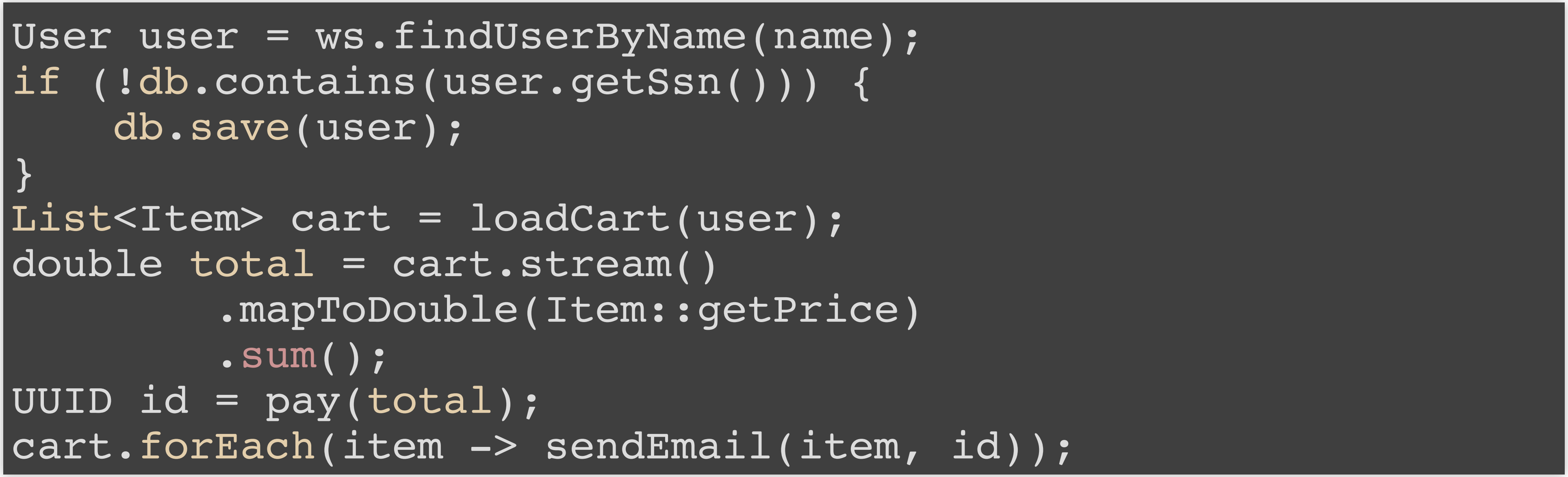
Now, take a deep breath…



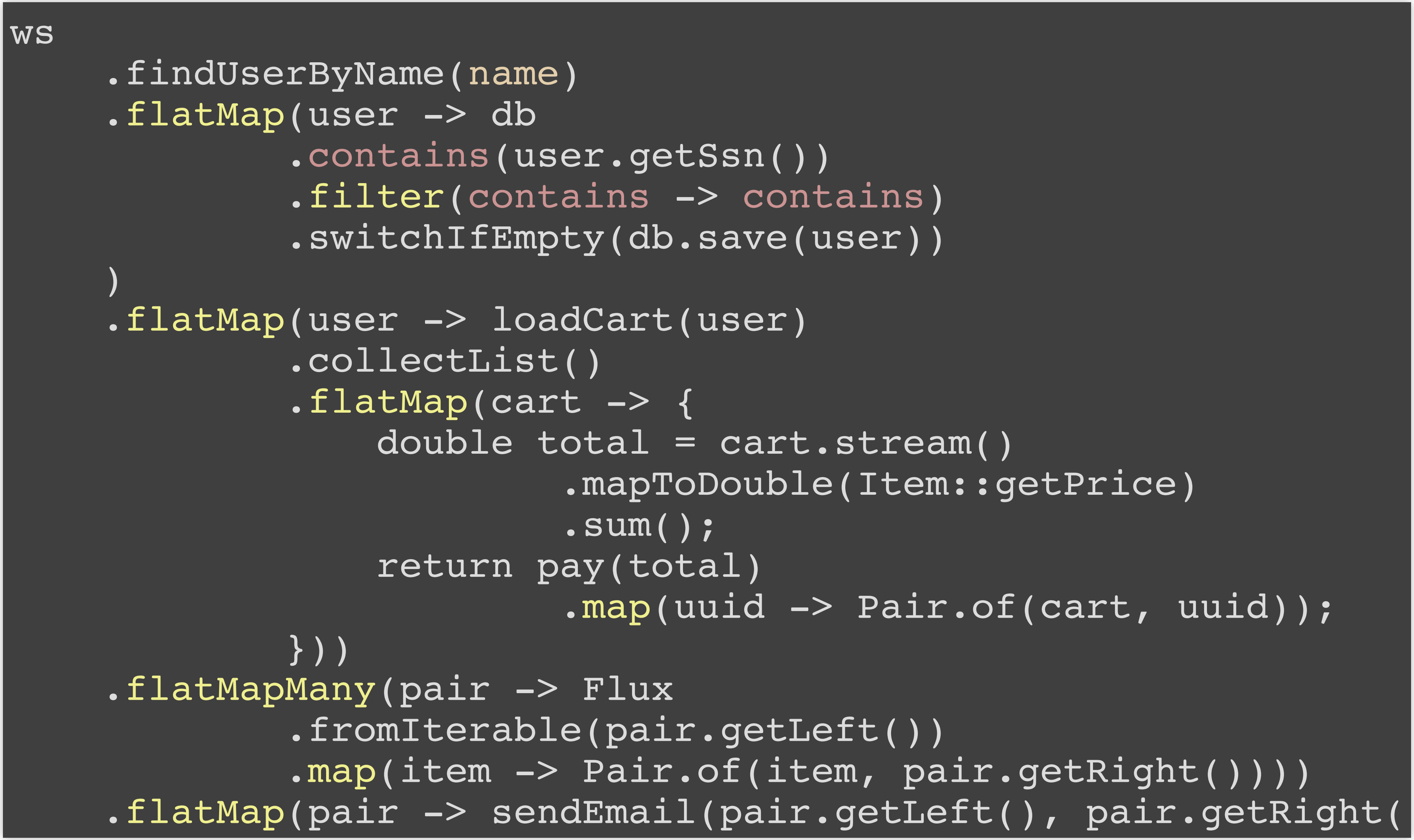
```
ws
    .findUserByName(name)
    .flatMap(user -> db
            .contains(user.getSsn())
            .filter(contains -> contains)
            .switchIfEmpty(db.save(user))
    )
    .flatMap(user -> loadCart(user)
            .collectList()
            .flatMap(cart -> {
                double total = cart.stream()
                        .mapToDouble(Item::getPrice)
                        .sum();
                return pay(total)
                        .map(uuid -> Pair.of(cart, uuid));
            }))
    .flatMapMany(pair -> Flux
            .fromIterable(pair.getLeft())
            .map(item -> Pair.of(item, pair.getRight())))
    .flatMap(pair -> sendEmail(pair.getLeft(), pair.getRight(
```



That escalated quickly



```java
User user = ws.findUserByName(name);
if (!db.contains(user.getSsn())) {
    db.save(user);
}
List<Item> cart = loadCart(user);
double total = cart.stream()
        .mapToDouble(Item::getPrice)
        .sum();
UUID id = pay(total);
cart.forEach(item -> sendEmail(item, id));
```



```
ws
    .findUserByName(name)
    .flatMap(user -> db
            .contains(user.getSsn())
            .filter(contains -> contains)
            .switchIfEmpty(db.save(user))
    )
    .flatMap(user -> loadCart(user)
            .collectList()
            .flatMap(cart -> {
                double total = cart.stream()
                        .mapToDouble(Item::getPrice)
                        .sum();
                return pay(total)
                        .map(uuid -> Pair.of(cart, uuid));
            }))
    .flatMapMany(pair -> Flux
            .fromIterable(pair.getLeft())
            .map(item -> Pair.of(item, pair.getRight())))
    .flatMap(pair -> sendEmail(pair.getLeft(), pair.getRight(
```

Ubiquitous language?

Are Semigroup, Monoid, Monad, Functor, Kleisli, and Yoneda pervasive in your domain model?

www.innoq.com/en/blog/the-language-of-maths-is-not-the-language-of-your-business/

cont.

> *Unless your core domain is mathematics, category theory is not the language used by your domain experts.*

www.innoq.com/en/blog/the-language-of-maths-is-not-the-language-of-your-business/

cont.

> *Good luck getting your domain experts to understand the language introduced by that abstraction.*

www.innoq.com/en/blog/the-language-of-maths-is-not-the-language-of-your-business/

Are `Mono` and `Flux` pervasive in your domain model?

Unless your core domain is infectious diseases, Reactor is not the language used by your domain experts.

en.wikipedia.org/wiki/Infectious_mononucleosis

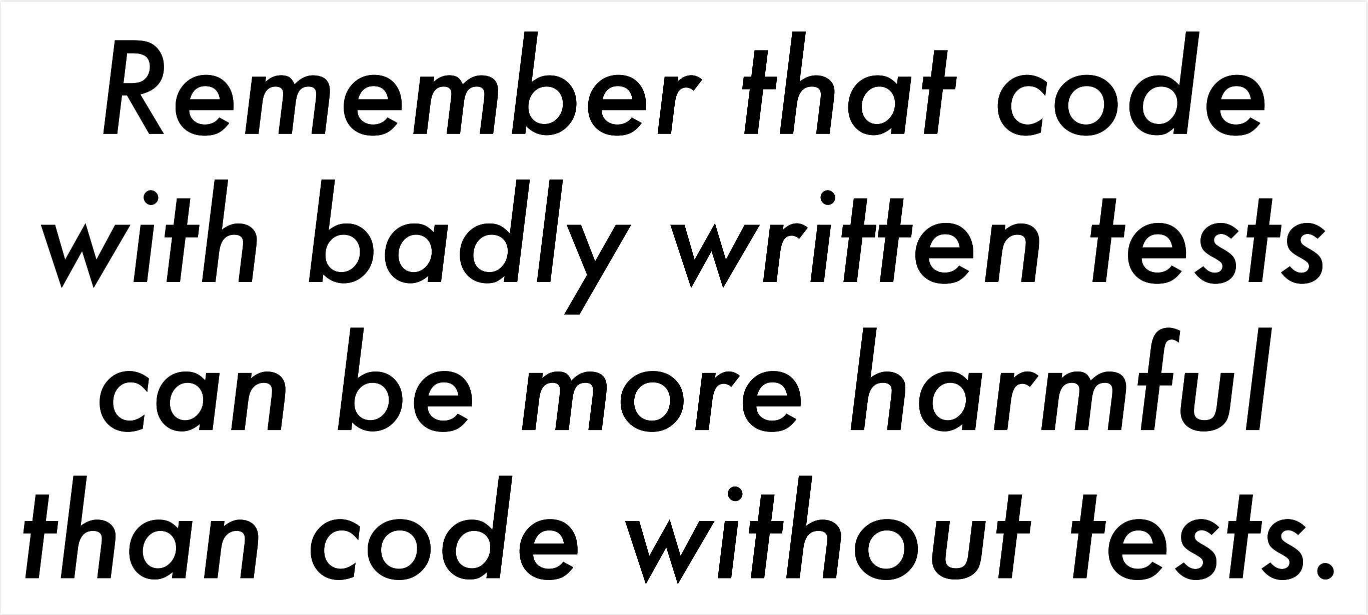
en.wikipedia.org/wiki/Dysentery

Universal measure of code quality?

“Simple”?

“Monad transformers are reducing boilerplate”

“Tested?”



Remember that code with badly written tests can be more harmful than code without tests.

hackernoon.com/few-simple-rules-for-good-coding-my-15-years-experience-96cb29d4acd9

- open/closed
- SOLID
- high cohesion
- low coupling
- cyclomatic complexity
- DRY
- ...

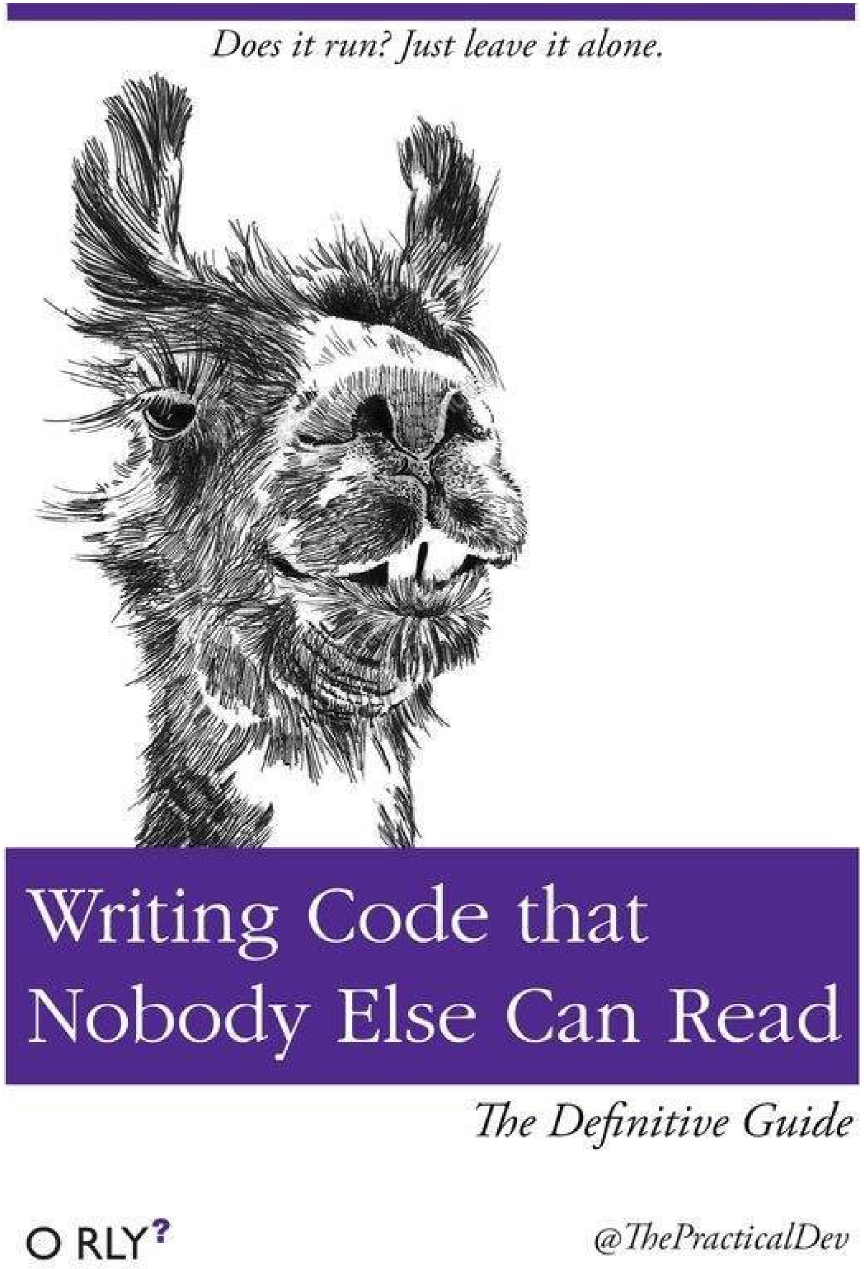
Boring

Implementation transparency

I don't care about patterns, frameworks, syntax

10x developer?

Enable 10 other developers



Does it run? Just leave it alone.

Writing Code that Nobody Else Can Read

The Definitive Guide

O RLY?

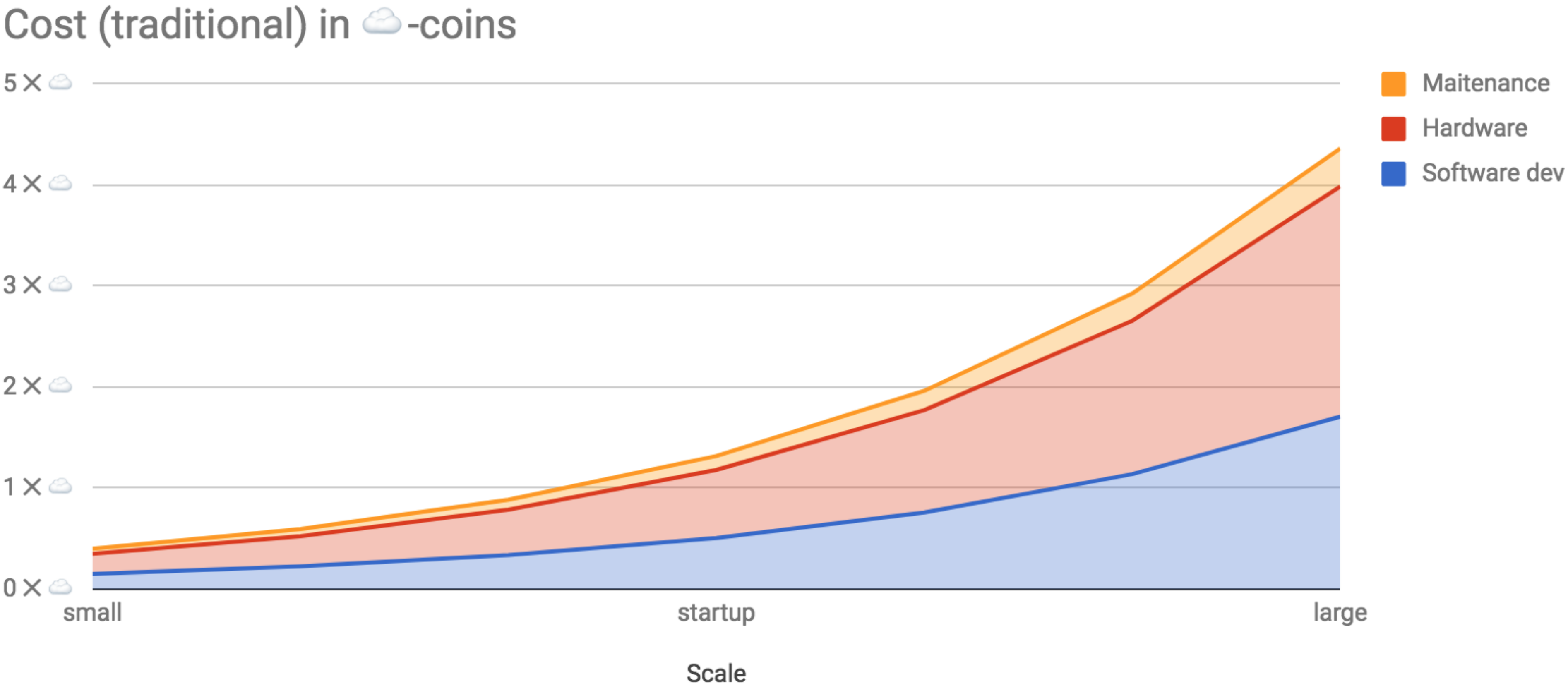
@ThePracticalDev

What are you optimizing?

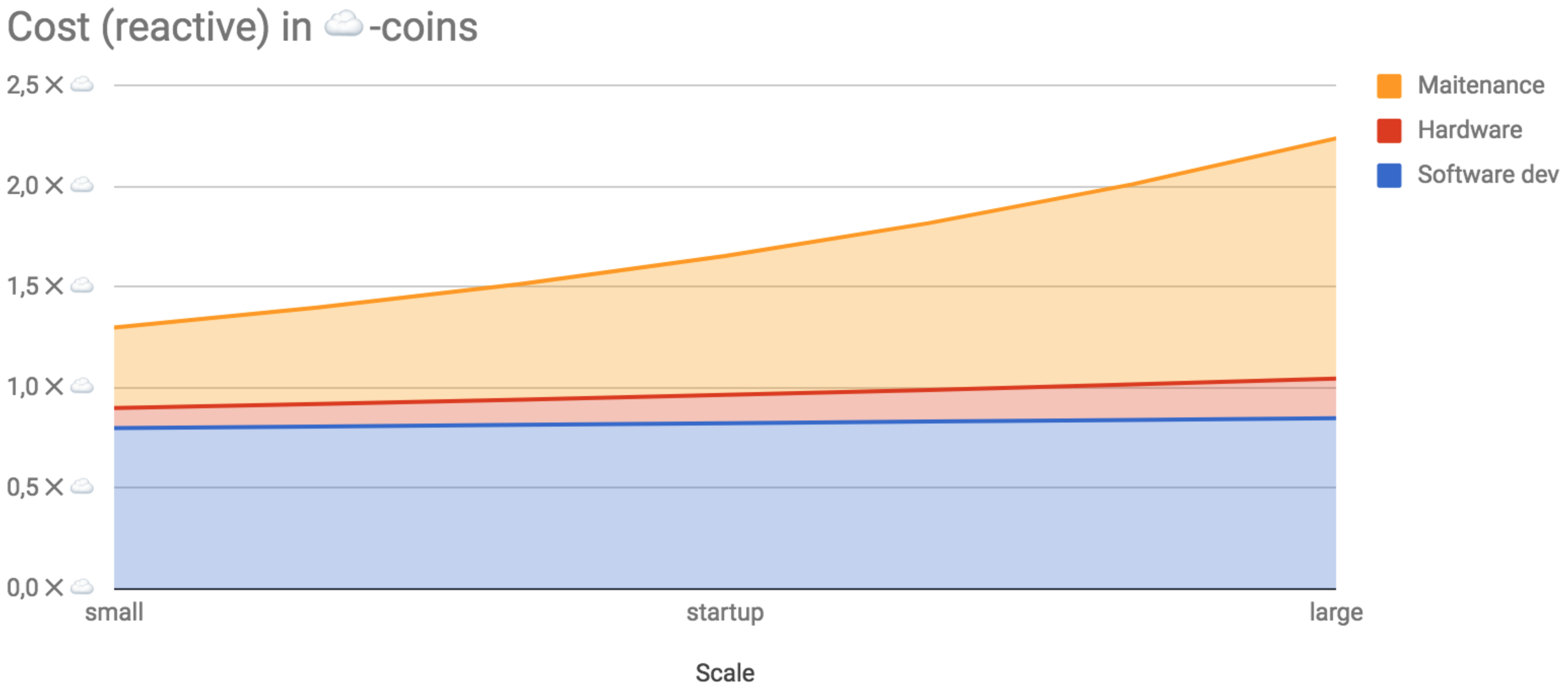
Costs

- Software development
- Hardware
- Maintenance

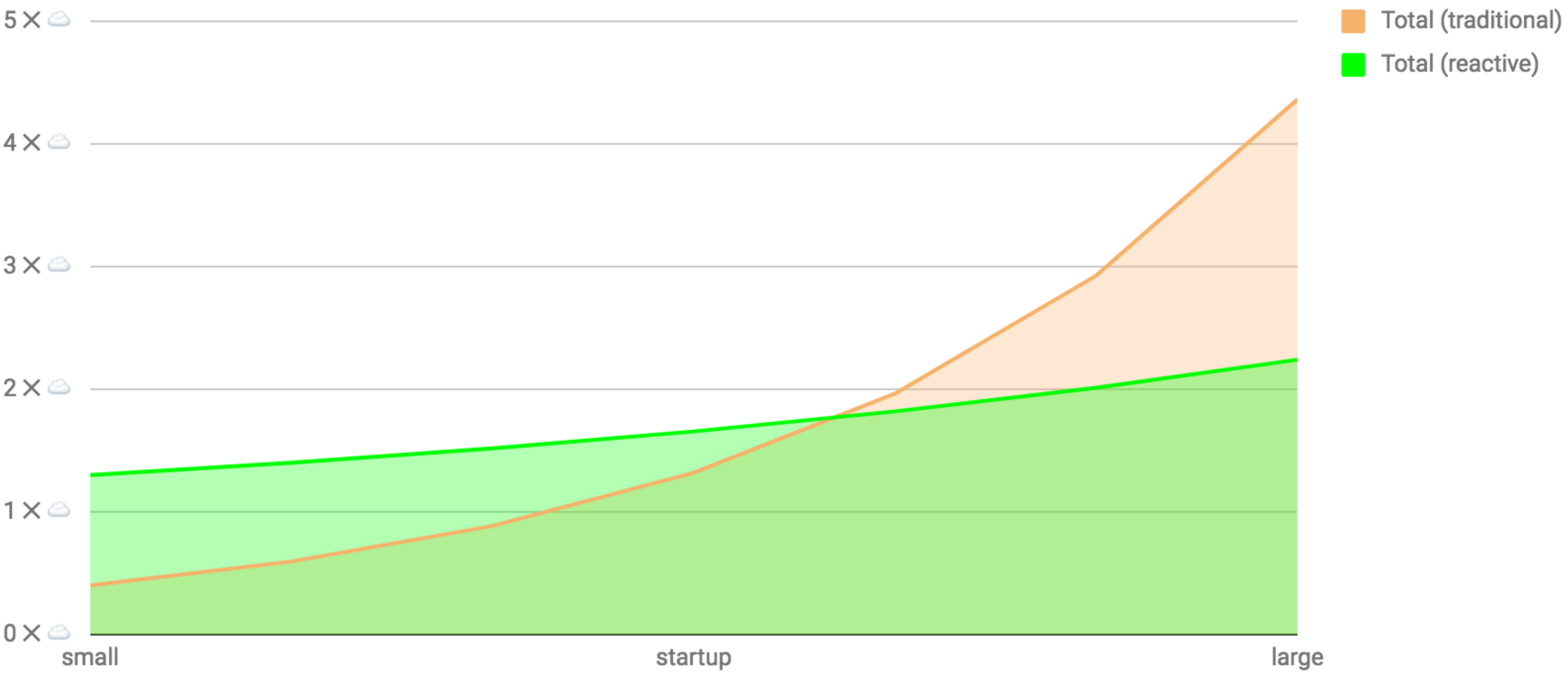
Blocking code



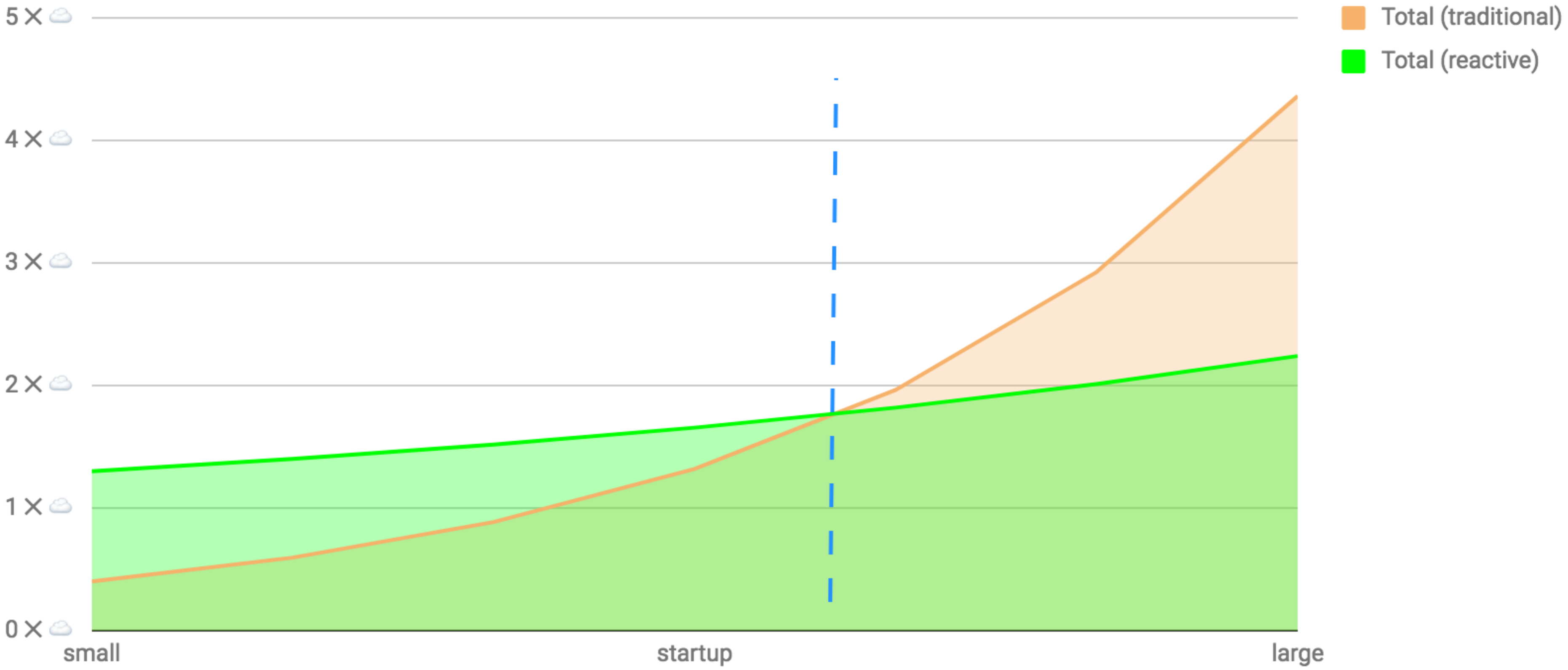
Reactive code



Tipping point?



The NETFLIX point



Are you NETFLIX?

No.

Little's law

$$L = \lambda W$$

en.wikipedia.org/wiki/Little's_law

Tomcat, 100 threads (L), 100 ms/request (W)

1K request/second (λ)

on a laptop

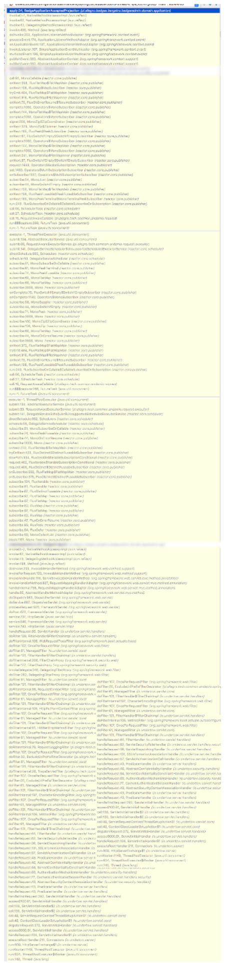
λ confirmed!

Ever heard of space-time trade off?

What about human-hardware trade off?

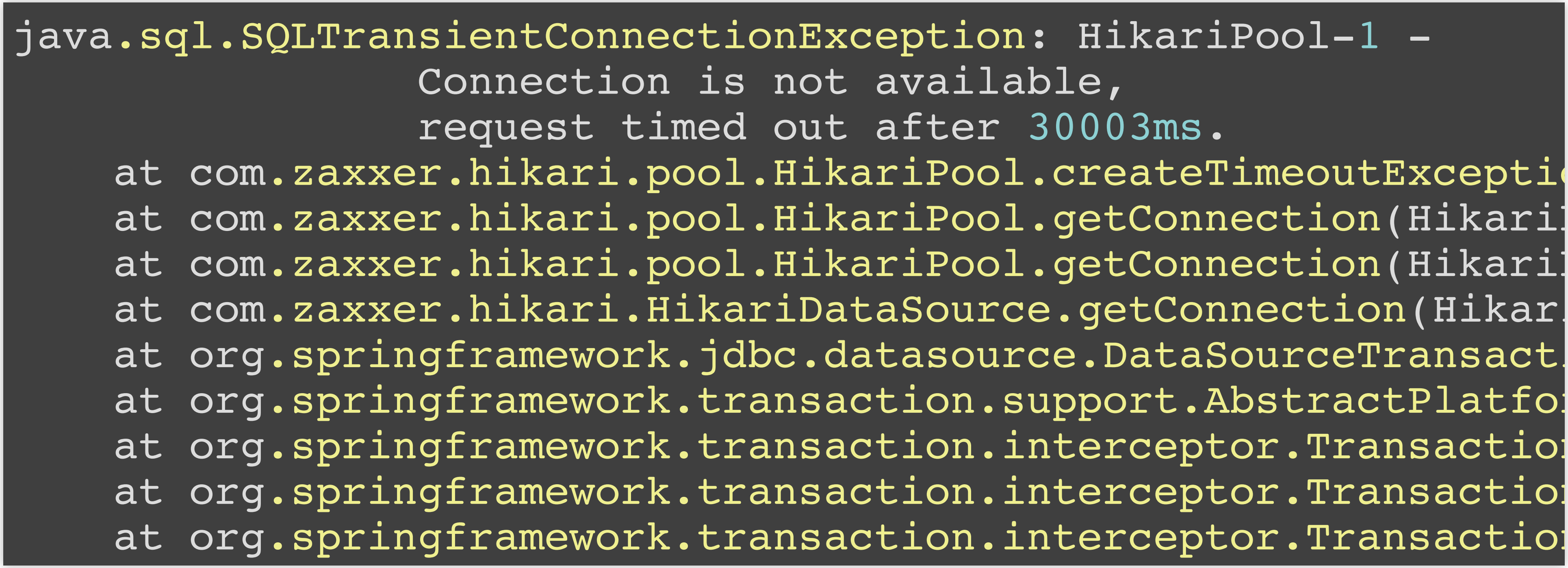
Let's talk about maintenance

I miss you, stack trace…



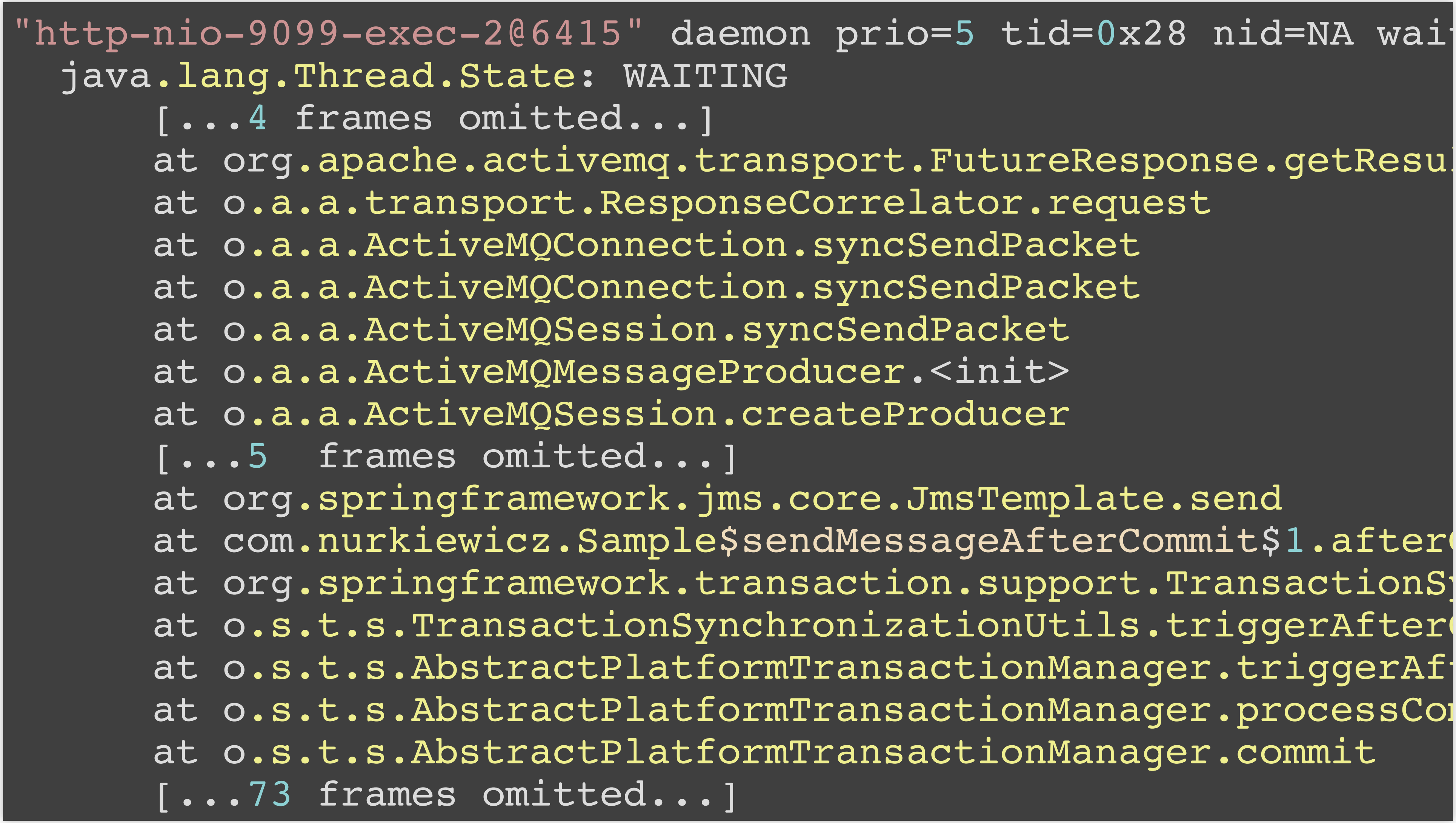
```
apply:24, BadgeApplicationAcceptedProjection (pl.allegro.badges.bargeine.badgesdomain.domain.application)
invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
invoke:62, NativeMethodAccessorImpl (sun.reflect)
invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
invoke:498, Method (java.lang.reflect)
doInvoke:253, ApplicationListenerMethodAdapter (org.springframework.context.event)
processEvent:174, ApplicationListenerMethodAdapter (org.springframework.context.event)
onApplicationEvent:137, ApplicationListenerMethodAdapter (org.springframework.context.event)
invokeListener:127, SimpleApplicationEventMulticaster (org.springframework.context.event)
multicastEvent:139, SimpleApplicationEventMulticaster (org.springframework.context.event)
publishEvent:383, AbstractApplicationContext (org.springframework.context.support)
publishEvent:350, AbstractApplicationContext (org.springframework.context.support)
[redacted]
call:91, MonoCallable (reactor.core.publisher)
onNext:358, FluxFlatMap$FlatMapMain (reactor.core.publisher)
onNext:108, FluxMap$MapSubscriber (reactor.core.publisher)
tryEmit:484, FluxFlatMap$FlatMapMain (reactor.core.publisher)
onNext:916, FluxFlatMap$FlatMapInner (reactor.core.publisher)
onNext:79, FluxOnErrorResume$ResumeSubscriber (reactor.core.publisher)
complete:1092, Operators$MonoSubscriber (reactor.core.publisher)
onNext:144, MonoFlatMap$FlatMapMain (reactor.core.publisher)
complete:1092, Operators$MonoSubscriber (reactor.core.publisher)
signal:556, MonoZip$ZipCoordinator (reactor.core.publisher)
onNext:329, MonoZip$ZipInner (reactor.core.publisher)
onNext:185, FluxPeek$PeekSubscriber (reactor.core.publisher)
onNext:87, FluxSwitchIfEmpty$SwitchIfEmptySubscriber (reactor.core.publisher)
complete:1092, Operators$MonoSubscriber (reactor.core.publisher)
onNext:144, MonoFlatMap$FlatMapMain (reactor.core.publisher)
complete:1092, Operators$MonoSubscriber (reactor.core.publisher)
onNext:241, MonoFlatMap$FlatMapInner (reactor.core.publisher)
onNext:87, FluxSwitchIfEmpty$SwitchIfEmptySubscriber (reactor.core.publisher)
request:1649, Operators$ScalarSubscription (reactor.core.publisher)
set:1463, Operators$MultiSubscriptionSubscriber (reactor.core.publisher)
onSubscribe:1337, Operators$MultiSubscriptionSubscriber (reactor.core.publisher)
subscribe:54, MonoJust (reactor.core.publisher)
subscribe:44, MonoSwitchIfEmpty (reactor.core.publisher)
onNext:150, MonoFlatMap$FlatMapMain (reactor.core.publisher)
onNext:198, FluxPeekFuseable$PeekFuseableSubscriber (reactor.core.publisher)
onNext:185, MonoPeekTerminal$MonoTerminalPeekSubscriber (reactor.core.publisher)
run:243, FluxSubscribeOnCallable$CallableSubscribeOnSubscription (reactor.core.publisher)
call:50, SchedulerTask (reactor.core.scheduler)
call:27, SchedulerTask (reactor.core.scheduler)
call:19, RequestAwareCallable (pl.allegro.tech.common.andamio.request)
run$$$capture:266, FutureTask (java.util.concurrent)
run:-1, FutureTask (java.util.concurrent)
execute:-1, ThreadPoolExecutor (java.util.concurrent)
submit:134, AbstractExecutorService (java.util.concurrent)
submit:33, RequestAwareExecutorService (pl.allegro.tech.common.andamio.request.executor)
submit:141, DelegateServiceScheduler$UnsupportedScheduledExecutorService (reactor.core.scheduler)
directSchedule:662, Schedulers (reactor.core.scheduler)
schedule:59, DelegateServiceScheduler (reactor.core.scheduler)
subscribe:51, MonoSubscribeOnCallable (reactor.core.publisher)
subscribe:61, MonoPeekTerminal (reactor.core.publisher)
subscribe:74, MonoPeekFuseable (reactor.core.publisher)
subscribe:60, MonoFlatMap (reactor.core.publisher)
subscribe:60, MonoFlatMap (reactor.core.publisher)
subscribe:3008, Mono (reactor.core.publisher)
onComplete:76, FluxSwitchIfEmpty$SwitchIfEmptySubscriber (reactor.core.publisher)
onComplete:1120, Operators$MonoSubscriber (reactor.core.publisher)
subscribe:56, MonoSupplier (reactor.core.publisher)
subscribe:44, MonoSwitchIfEmpty (reactor.core.publisher)
subscribe:71, MonoPeek (reactor.core.publisher)
subscribe:3008, Mono (reactor.core.publisher)
subscribe:100, MonoZip$ZipCoordinator (reactor.core.publisher)
subscribe:128, MonoZip (reactor.core.publisher)
subscribe:60, MonoFlatMap (reactor.core.publisher)
subscribe:44, MonoOnErrorResume (reactor.core.publisher)
subscribe:3008, Mono (reactor.core.publisher)
onNext:372, FluxFlatMap$FlatMapMain (reactor.core.publisher)
tryEmit:484, FluxFlatMap$FlatMapMain (reactor.core.publisher)
onNext:916, FluxFlatMap$FlatMapInner (reactor.core.publisher)
onNext:79, FluxOnErrorResume$ResumeSubscriber (reactor.core.publisher)
onNext:198, FluxPeekFuseable$PeekFuseableSubscriber (reactor.core.publisher)
run:243, FluxSubscribeOnCallable$CallableSubscribeOnSubscription (reactor.core.publisher)
call:50, SchedulerTask (reactor.core.scheduler)
call:27, SchedulerTask (reactor.core.scheduler)
call:19, RequestAwareCallable (pl.allegro.tech.common.andamio.request)
run$$$capture:266, FutureTask (java.util.concurrent)
run:-1, FutureTask (java.util.concurrent)
execute:-1, ThreadPoolExecutor (java.util.concurrent)
submit:134, AbstractExecutorService (java.util.concurrent)
submit:33, RequestAwareExecutorService (pl.allegro.tech.common.andamio.request.executor)
submit:141, DelegateServiceScheduler$UnsupportedScheduledExecutorService (reactor.core.scheduler)
directSchedule:662, Schedulers (reactor.core.scheduler)
schedule:59, DelegateServiceScheduler (reactor.core.scheduler)
subscribe:51, MonoSubscribeOnCallable (reactor.core.publisher)
subscribe:74, MonoPeekFuseable (reactor.core.publisher)
subscribe:44, MonoOnErrorResume (reactor.core.publisher)
subscribe:3008, Mono (reactor.core.publisher)
onNext:372, FluxFlatMap$FlatMapMain (reactor.core.publisher)
tryOnNext:423, FluxDistinct$DistinctFuseableSubscriber (reactor.core.publisher)
slowPath:534, FluxIterable$IterableSubscriptionConditional (reactor.core.publisher)
request:462, FluxIterable$IterableSubscriptionConditional (reactor.core.publisher)
request:466, FluxDistinct$DistinctFuseableSubscriber (reactor.core.publisher)
onSubscribe:222, FluxFlatMap$FlatMapMain (reactor.core.publisher)
onSubscribe:375, FluxDistinct$DistinctFuseableSubscriber (reactor.core.publisher)
subscribe:124, FluxIterable (reactor.core.publisher)
subscribe:61, FluxIterable (reactor.core.publisher)
subscribe:67, FluxDistinctFuseable (reactor.core.publisher)
subscribe:97, FluxFlatMap (reactor.core.publisher)
subscribe:97, FluxFlatMap (reactor.core.publisher)
subscribe:62, FluxMap (reactor.core.publisher)
subscribe:97, FluxFlatMap (reactor.core.publisher)
subscribe:62, FluxMap (reactor.core.publisher)
subscribe:47, FluxOnErrorResume (reactor.core.publisher)
subscribe:83, FluxPeek (reactor.core.publisher)
subscribe:54, FluxDefer (reactor.core.publisher)
subscribe:59, MonoCollectList (reactor.core.publisher)
block:1161, Mono (reactor.core.publisher)
[redacted]
invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
invoke:62, NativeMethodAccessorImpl (sun.reflect)
invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
invoke:498, Method (java.lang.reflect)
doInvoke:205, InvocableHandlerMethod (org.springframework.web.method.support)
invokeForRequest:133, InvocableHandlerMethod (org.springframework.web.method.support)
invokeAndHandle:116, ServletInvocableHandlerMethod (org.springframework.web.servlet.mvc.method.annotation)
invokeHandlerMethod:827, RequestMappingHandlerAdapter (org.springframework.web.servlet.mvc.method.annotation)
handleInternal:738, RequestMappingHandlerAdapter (org.springframework.web.servlet.mvc.method.annotation)
handle:85, AbstractHandlerMethodAdapter (org.springframework.web.servlet.mvc.method)
doDispatch:963, DispatcherServlet (org.springframework.web.servlet)
doService:897, DispatcherServlet (org.springframework.web.servlet)
processRequest:970, FrameworkServlet (org.springframework.web.servlet)
doPost:872, FrameworkServlet (org.springframework.web.servlet)
service:707, HttpServlet (javax.servlet.http)
service:846, FrameworkServlet (org.springframework.web.servlet)
service:790, HttpServlet (javax.servlet.http)
handleRequest:85, ServletHandler (io.undertow.servlet.handlers)
doFilter:129, FilterHandler$FilterChainImpl (io.undertow.servlet.handlers)
doFilterInternal:108, WebRequestTraceFilter (org.springframework.boot.actuate.trace)
doFilter:107, OncePerRequestFilter (org.springframework.web.filter)
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.servlet.handlers)
doFilterInternal:208, FilterChainProxy (org.springframework.security.web)
doFilter:177, FilterChainProxy (org.springframework.security.web)
invokeDelegate:346, DelegatingFilterProxy (org.springframework.web.filter)
doFilter:262, DelegatingFilterProxy (org.springframework.web.filter)
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.ser
doFilterInternal:99, RequestContextFilter (org.springframe
doFilter:107, OncePerRequestFilter (org.springframework.w
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.ser
doFilterInternal:105, HttpPutFormContentFilter (org.spring
doFilter:107, OncePerRequestFilter (org.springframework.w
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.ser
doFilterInternal:81, HiddenHttpMethodFilter (org.springfra
doFilter:107, OncePerRequestFilter (org.springframework.w
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.ser
doFilterInternal:79, RequestLoggingFilter (pl.allegro.tech.c
doFilter:107, OncePerRequestFilter (org.springframework.w
doFilter:33, ExcludeUrlPrefixFilterDecorator (pl.allegro.tech
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.ser
doFilterInternal:26, TraceServletFilter (pl.allegro.tech.comm
doFilter:107, OncePerRequestFilter (org.springframework.w
doFilter:33, ExcludeUrlPrefixFilterDecorator (pl.allegro.tech
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.ser
doFilterInternal:197, CharacterEncodingFilter (org.springfra
doFilter:107, OncePerRequestFilter (org.springframework.w
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.ser
doFilterInternal:106, MetricsFilter (org.springframework.bo
doFilter:107, OncePerRequestFilter (org.springframework.w
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.ser
handleRequest:84, FilterHandler (io.undertow.servlet.hand
handleRequest:62, ServletSecurityRoleHandler (io.underto
handleRequest:36, ServletDispatchingHandler (io.undertow
handleRequest:131, SSLInformationAssociationHandler (io.
handleRequest:57, ServletAuthenticationCallHandler (io.un
handleRequest:43, PredicateHandler (io.undertow.server.h
handleRequest:46, AbstractConfidentialityHandler (io.unde
handleRequest:64, ServletConfidentialityConstraintHandle
handleRequest:60, AuthenticationMechanismsHandler (io.undertow.security.handlers)
handleRequest:77, CachedAuthenticatedSessionHandler (io.undertow.servlet.handlers.security)
handleRequest:43, AbstractSecurityContextAssociationHandler (io.undertow.security.handlers)
handleRequest:43, PredicateHandler (io.undertow.server.handlers)
handleRequest:43, PredicateHandler (io.undertow.server.handlers)
handleFirstRequest:292, ServletInitialHandler (io.undertow.servlet.handlers)
access$100:81, ServletInitialHandler (io.undertow.servlet.handlers)
call:138, ServletInitialHandler$2 (io.undertow.servlet.handlers)
call:135, ServletInitialHandler$2 (io.undertow.servlet.handlers)
call:48, ServletRequestContextThreadSetupAction$1 (io.undertow.servlet.core)
call:43, ContextClassLoaderSetupAction$1 (io.undertow.servlet.core)
dispatchRequest:272, ServletInitialHandler (io.undertow.servlet.handlers)
access$000:81, ServletInitialHandler (io.undertow.servlet.handlers)
handleRequest:104, ServletInitialHandler$1 (io.undertow.servlet.handlers)
executeRootHandler:211, Connectors (io.undertow.server)
run:809, HttpServerExchange$1 (io.undertow.server)
runWorker:1149, ThreadPoolExecutor (java.util.concurrent)
run:624, ThreadPoolExecutor$Worker (java.util.concurrent)
run:748, Thread (java.lang)
```

```
doFilter:107, OncePerRequestFilter (org.springframework.web.filter)
doFilter:33, ExcludeUrlPrefixFilterDecorator (pl.allegro.tech.common.andamio.spring
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.servlet.handlers)
doFilterInternal:197, CharacterEncodingFilter (org.springframework.web.filter)
doFilter:107, OncePerRequestFilter (org.springframework.web.filter)
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.servlet.handlers)
doFilterInternal:106, MetricsFilter (org.springframework.boot.actuate.autoconfigure
doFilter:107, OncePerRequestFilter (org.springframework.web.filter)
doFilter:61, ManagedFilter (io.undertow.servlet.core)
doFilter:131, FilterHandler$FilterChainImpl (io.undertow.servlet.handlers)
handleRequest:84, FilterHandler (io.undertow.servlet.handlers)
handleRequest:62, ServletSecurityRoleHandler (io.undertow.servlet.handlers.securi
handleRequest:36, ServletDispatchingHandler (io.undertow.servlet.handlers)
handleRequest:131, SSLInformationAssociationHandler (io.undertow.servlet.handle
handleRequest:57, ServletAuthenticationCallHandler (io.undertow.servlet.handlers
handleRequest:43, PredicateHandler (io.undertow.server.handlers)
handleRequest:46, AbstractConfidentialityHandler (io.undertow.security.handlers)
handleRequest:64, ServletConfidentialityConstraintHandler (io.undertow.servlet.ha
handleRequest:60, AuthenticationMechanismsHandler (io.undertow.security.handl
handleRequest:77, CachedAuthenticatedSessionHandler (io.undertow.servlet.hand
handleRequest:43, AbstractSecurityContextAssociationHandler (io.undertow.secur
handleRequest:43, PredicateHandler (io.undertow.server.handlers)
handleRequest:43, PredicateHandler (io.undertow.server.handlers)
handleFirstRequest:292, ServletInitialHandler (io.undertow.servlet.handlers)
access$100:81, ServletInitialHandler (io.undertow.servlet.handlers)
call:138, ServletInitialHandler$2 (io.undertow.servlet.handlers)
call:135, ServletInitialHandler$2 (io.undertow.servlet.handlers)
call:48, ServletRequestContextThreadSetupAction$1 (io.undertow.servlet.core)
call:43, ContextClassLoaderSetupAction$1 (io.undertow.servlet.core)
dispatchRequest:272, ServletInitialHandler (io.undertow.servlet.handlers)
access$000:81, ServletInitialHandler (io.undertow.servlet.handlers)
handleRequest:104, ServletInitialHandler$1 (io.undertow.servlet.handlers)
executeRootHandler:211, Connectors (io.undertow.server)
run:809, HttpServerExchange$1 (io.undertow.server)
runWorker:1149, ThreadPoolExecutor (java.util.concurrent)
run:624, ThreadPoolExecutor$Worker (java.util.concurrent)
run:748, Thread (java.lang)
```

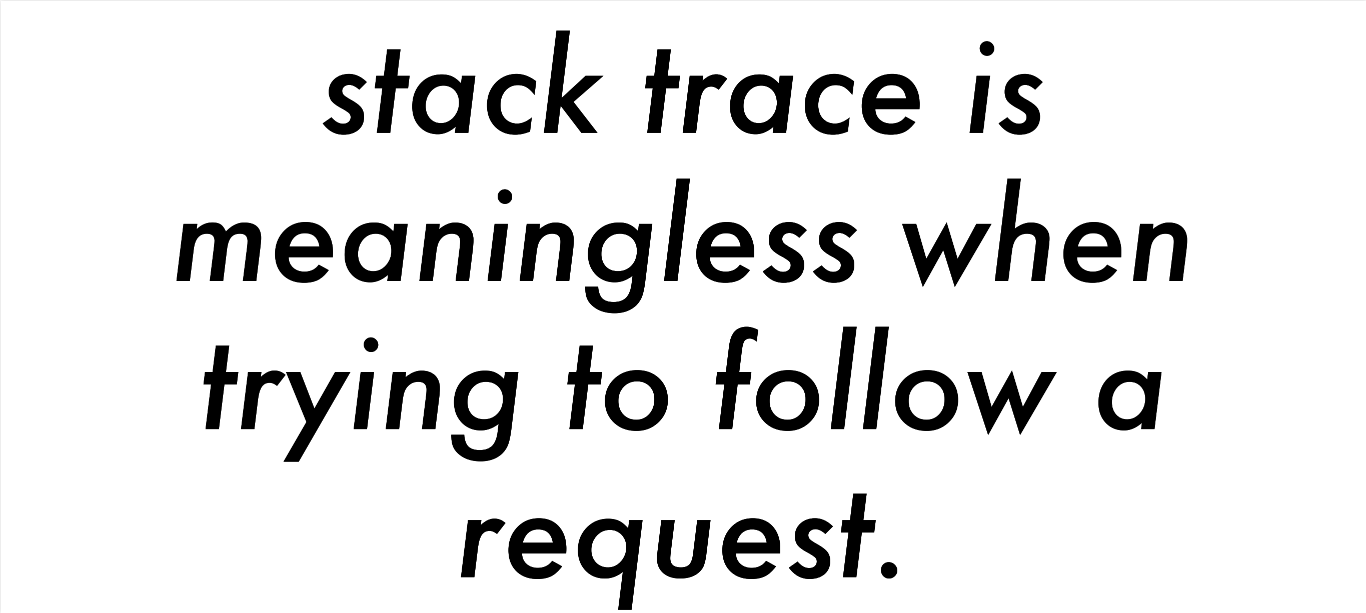


```
java.sql.SQLTransientConnectionException: HikariPool-1 -
                Connection is not available,
                request timed out after 30003ms.
    at com.zaxxer.hikari.pool.HikariPool.createTimeoutExcepti
    at com.zaxxer.hikari.pool.HikariPool.getConnection(Hikari
    at com.zaxxer.hikari.pool.HikariPool.getConnection(Hikari
    at com.zaxxer.hikari.HikariDataSource.getConnection(Hikar
    at org.springframework.jdbc.datasource.DataSourceTransact
    at org.springframework.transaction.support.AbstractPlatfo
    at org.springframework.transaction.interceptor.Transactio
    at org.springframework.transaction.interceptor.Transactio
    at org.springframework.transaction.interceptor.Transactio
```

www.nurkiewicz.com/2017/03/beware-of-slow-transaction-callbacks-in.html



```
"http-nio-9099-exec-2@6415" daemon prio=5 tid=0x28 nid=NA wai
  java.lang.Thread.State: WAITING
      [...4 frames omitted...]
      at org.apache.activemq.transport.FutureResponse.getResu
      at o.a.a.transport.ResponseCorrelator.request
      at o.a.a.ActiveMQConnection.syncSendPacket
      at o.a.a.ActiveMQConnection.syncSendPacket
      at o.a.a.ActiveMQSession.syncSendPacket
      at o.a.a.ActiveMQMessageProducer.<init>
      at o.a.a.ActiveMQSession.createProducer
      [...5  frames omitted...]
      at org.springframework.jms.core.JmsTemplate.send
      at com.nurkiewicz.Sample$sendMessageAfterCommit$1.after
      at org.springframework.transaction.support.TransactionSy
      at o.s.t.s.TransactionSynchronizationUtils.triggerAfter
      at o.s.t.s.AbstractPlatformTransactionManager.triggerAf
      at o.s.t.s.AbstractPlatformTransactionManager.processCo
      at o.s.t.s.AbstractPlatformTransactionManager.commit
      [...73 frames omitted...]
```



stack trace is meaningless when trying to follow a request.

medium.com/netflix-techblog/zuul-2-the-netflix-journey-to-asynchronous-non-blocking-systems-45947377fb5c

> *It is difficult to follow a request as events and callbacks are processed [...]*

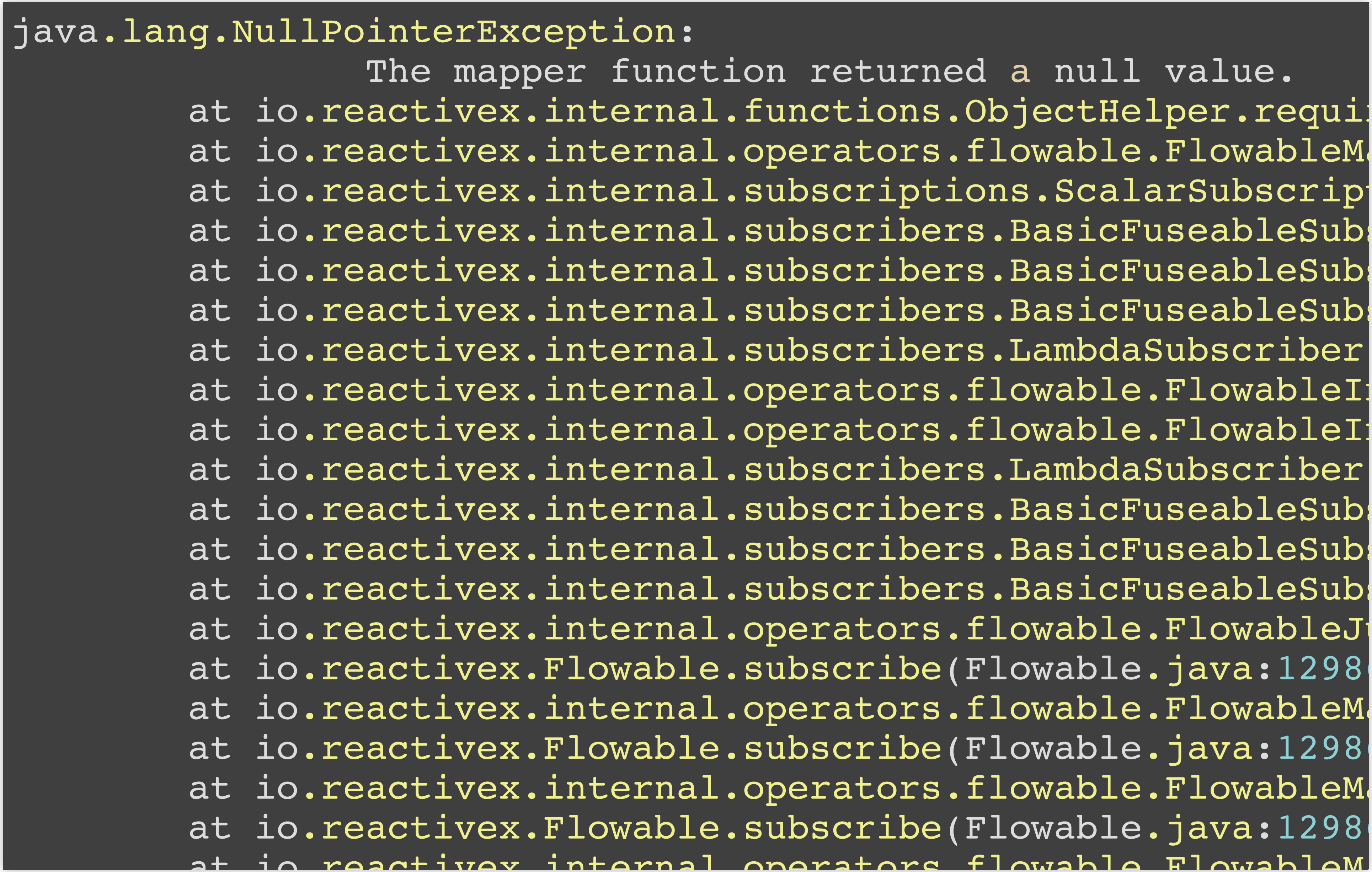
medium.com/netflix-techblog/zuul-2-the-netflix-journey-to-asynchronous-non-blocking-systems-45947377fb5c

> […] unhandled exceptions, and incorrectly handled state changes […] These types of issues have proven to be quite difficult to debug

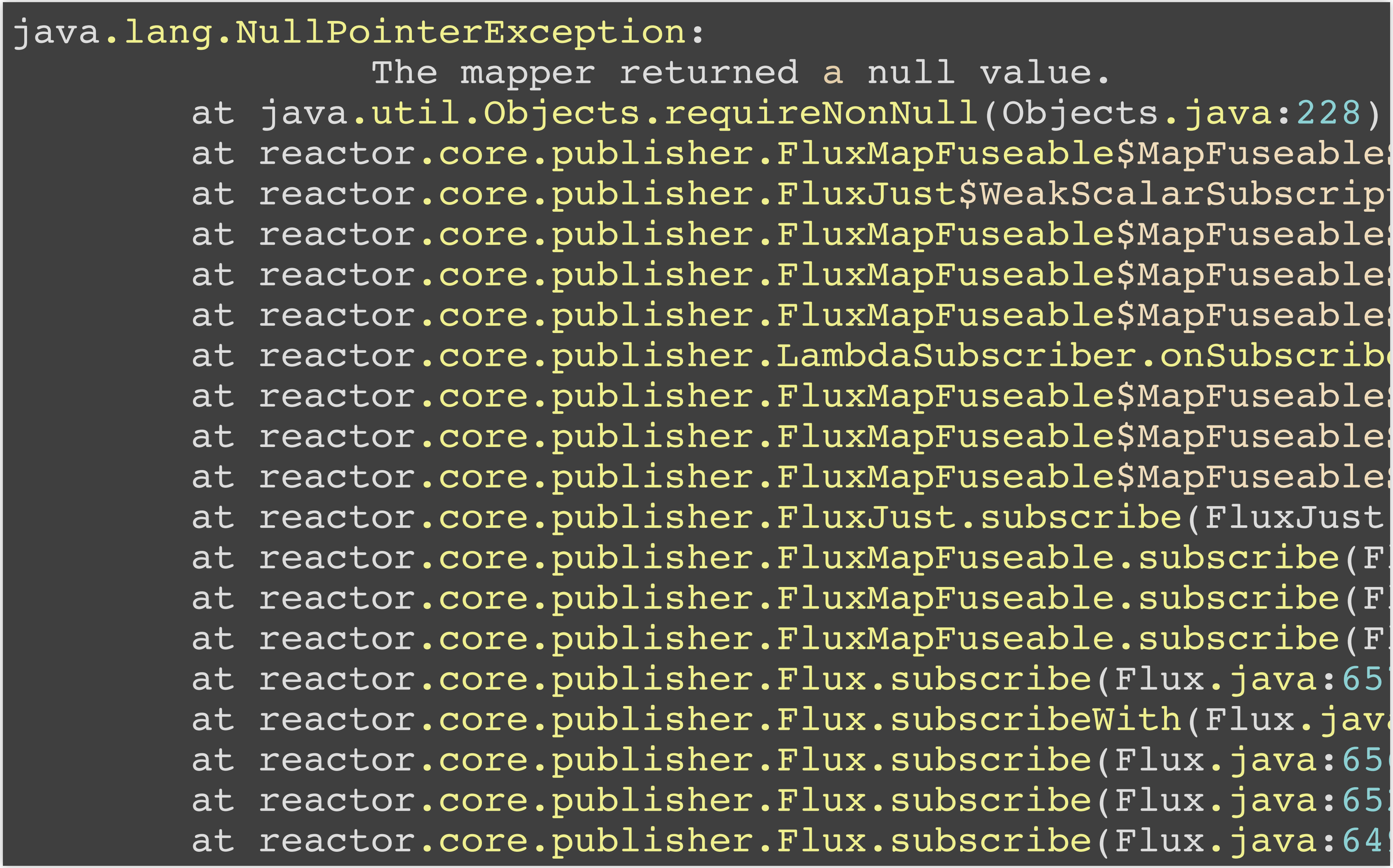
medium.com/netflix-techblog/zuul-2-the-netflix-journey-to-asynchronous-non-blocking-systems-45947377fb5c

Exceptions from hell

```
flux
        .map(this::foo)
        .map(this::bar)
        .map(this::buzz)
```

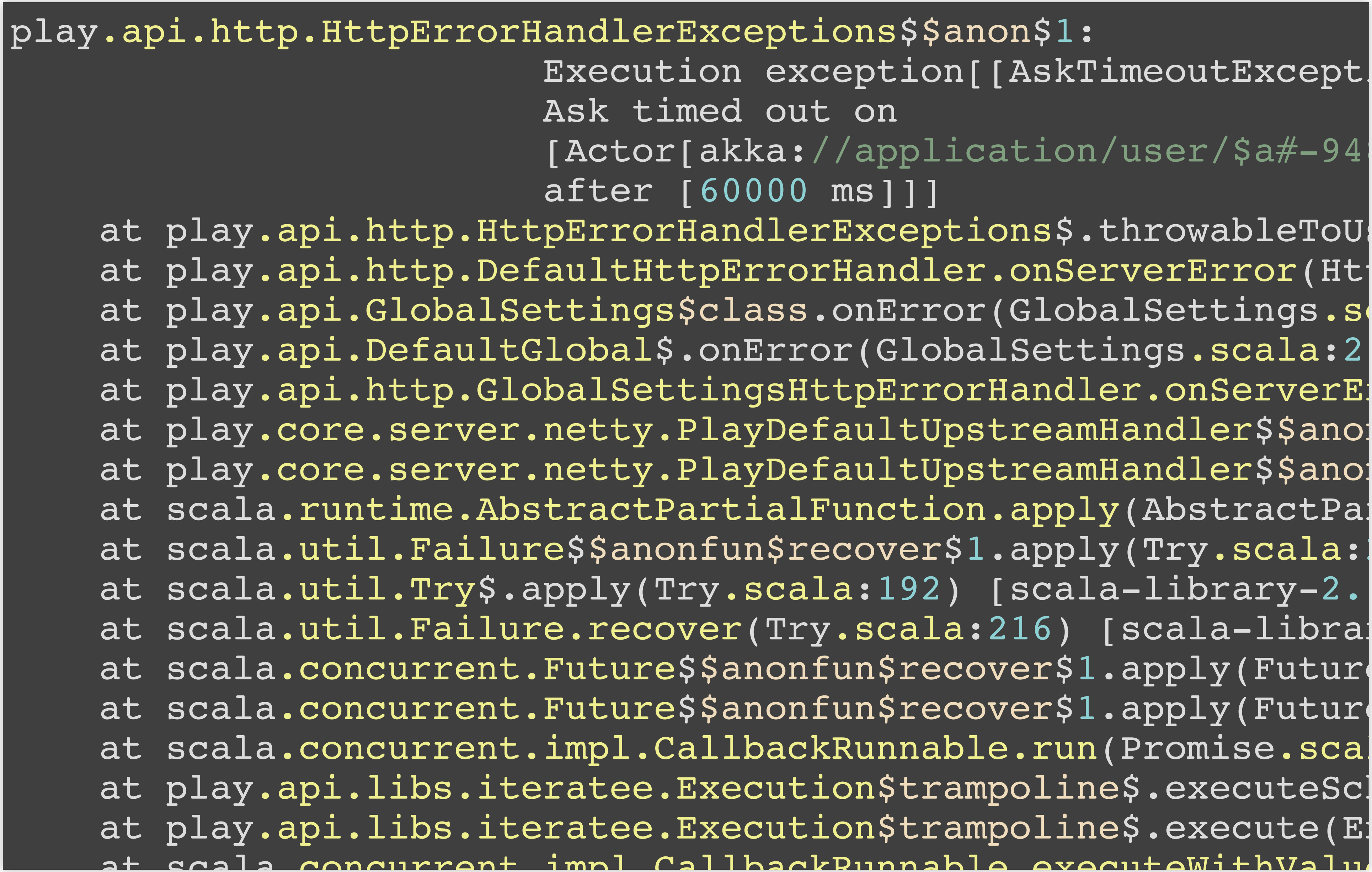


```
java.lang.NullPointerException:
                The mapper function returned a null value.
        at io.reactivex.internal.functions.ObjectHelper.requi
        at io.reactivex.internal.operators.flowable.FlowableMa
        at io.reactivex.internal.subscriptions.ScalarSubscript
        at io.reactivex.internal.subscribers.BasicFuseableSubs
        at io.reactivex.internal.subscribers.BasicFuseableSubs
        at io.reactivex.internal.subscribers.BasicFuseableSubs
        at io.reactivex.internal.subscribers.LambdaSubscriber
        at io.reactivex.internal.operators.flowable.FlowableIn
        at io.reactivex.internal.operators.flowable.FlowableIn
        at io.reactivex.internal.subscribers.LambdaSubscriber
        at io.reactivex.internal.subscribers.BasicFuseableSubs
        at io.reactivex.internal.subscribers.BasicFuseableSubs
        at io.reactivex.internal.subscribers.BasicFuseableSubs
        at io.reactivex.internal.operators.flowable.FlowableJu
        at io.reactivex.Flowable.subscribe(Flowable.java:1298
        at io.reactivex.internal.operators.flowable.FlowableMa
        at io.reactivex.Flowable.subscribe(Flowable.java:1298
        at io.reactivex.internal.operators.flowable.FlowableMa
        at io.reactivex.Flowable.subscribe(Flowable.java:1298
        at io.reactivex.internal.operators.flowable.FlowableM
```



```
java.lang.NullPointerException:
                The mapper returned a null value.
        at java.util.Objects.requireNonNull(Objects.java:228)
        at reactor.core.publisher.FluxMapFuseable$MapFuseable
        at reactor.core.publisher.FluxJust$WeakScalarSubscrip
        at reactor.core.publisher.FluxMapFuseable$MapFuseable
        at reactor.core.publisher.FluxMapFuseable$MapFuseable
        at reactor.core.publisher.FluxMapFuseable$MapFuseable
        at reactor.core.publisher.LambdaSubscriber.onSubscrib
        at reactor.core.publisher.FluxMapFuseable$MapFuseable
        at reactor.core.publisher.FluxMapFuseable$MapFuseable
        at reactor.core.publisher.FluxMapFuseable$MapFuseable
        at reactor.core.publisher.FluxJust.subscribe(FluxJust
        at reactor.core.publisher.FluxMapFuseable.subscribe(F
        at reactor.core.publisher.FluxMapFuseable.subscribe(F
        at reactor.core.publisher.FluxMapFuseable.subscribe(F
        at reactor.core.publisher.Flux.subscribe(Flux.java:65
        at reactor.core.publisher.Flux.subscribeWith(Flux.jav
        at reactor.core.publisher.Flux.subscribe(Flux.java:65
        at reactor.core.publisher.Flux.subscribe(Flux.java:65
        at reactor.core.publisher.Flux.subscribe(Flux.java:64
```

`TimeoutException`



```
play.api.http.HttpErrorHandlerExceptions$$anon$1:
                        Execution exception[[AskTimeoutExcept
                        Ask timed out on
                        [Actor[akka://application/user/$a#-94
                        after [60000 ms]]]
    at play.api.http.HttpErrorHandlerExceptions$.throwableToU
    at play.api.http.DefaultHttpErrorHandler.onServerError(Ht
    at play.api.GlobalSettings$class.onError(GlobalSettings.s
    at play.api.DefaultGlobal$.onError(GlobalSettings.scala:2
    at play.api.http.GlobalSettingsHttpErrorHandler.onServerE
    at play.core.server.netty.PlayDefaultUpstreamHandler$$ano
    at play.core.server.netty.PlayDefaultUpstreamHandler$$ano
    at scala.runtime.AbstractPartialFunction.apply(AbstractPa
    at scala.util.Failure$$anonfun$recover$1.apply(Try.scala:
    at scala.util.Try$.apply(Try.scala:192) [scala-library-2.
    at scala.util.Failure.recover(Try.scala:216) [scala-libra
    at scala.concurrent.Future$$anonfun$recover$1.apply(Futur
    at scala.concurrent.Future$$anonfun$recover$1.apply(Futur
    at scala.concurrent.impl.CallbackRunnable.run(Promise.sca
    at play.api.libs.iteratee.Execution$trampoline$.executeSc
    at play.api.libs.iteratee.Execution$trampoline$.execute(E
    at scala.concurrent.impl.CallbackRunnable.executeWithValu
```

github.com/CogComp/open-eval/issues/186

BEST
PRACTICES

The best reactive system is single-threaded (!)

- Reactor pattern
- Actors
- SPSC queues
- Rx/Reactor operators

Order is no longer guaranteed

`parallel()` vs
`flatMap()` vs
`concatMap()` vs
`concatMapEager()`

DDoS was never that simple!

```
flatMap(..., 128)
```

Monitoring

Staged event-driven architecture

an approach to software architecture that decomposes a complex, event-driven application into a set of stages connected by queues

en.wikipedia.org/wiki/Staged_event-driven_architecture

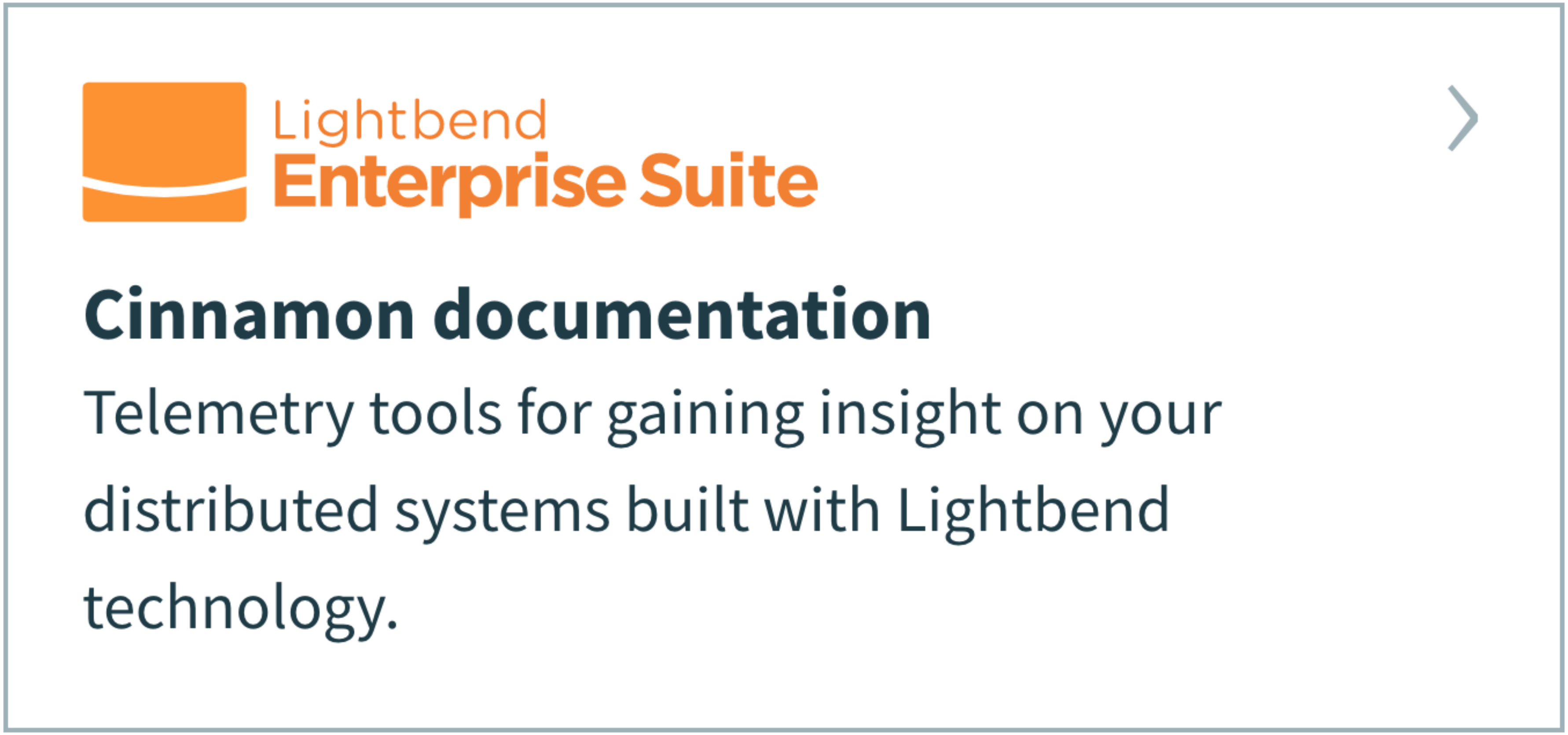
Watch out for your queues

Why no `mailboxSize` in Akka 2?

slow, inaccurate, impossible, silly

letitcrash.com/post/17707262394/why-no-mailboxsize-in-akka-2

Lightbend Telemetry 💰



Lightbend
Enterprise Suite

Cinnamon documentation

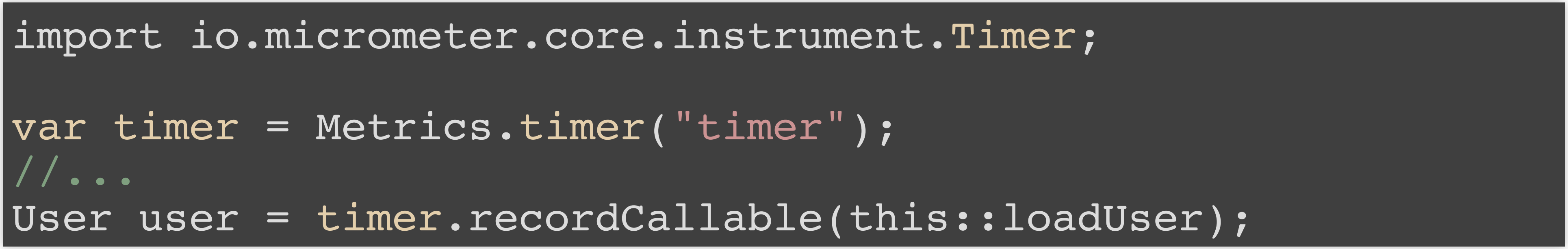
Telemetry tools for gaining insight on your distributed systems built with Lightbend technology.

Mailbox size (counter)
statistics for actor mailbox sizes

developer.lightbend.com/docs/cinnamon/2.5.x/instrumentations/akka/akka.html

Timing things

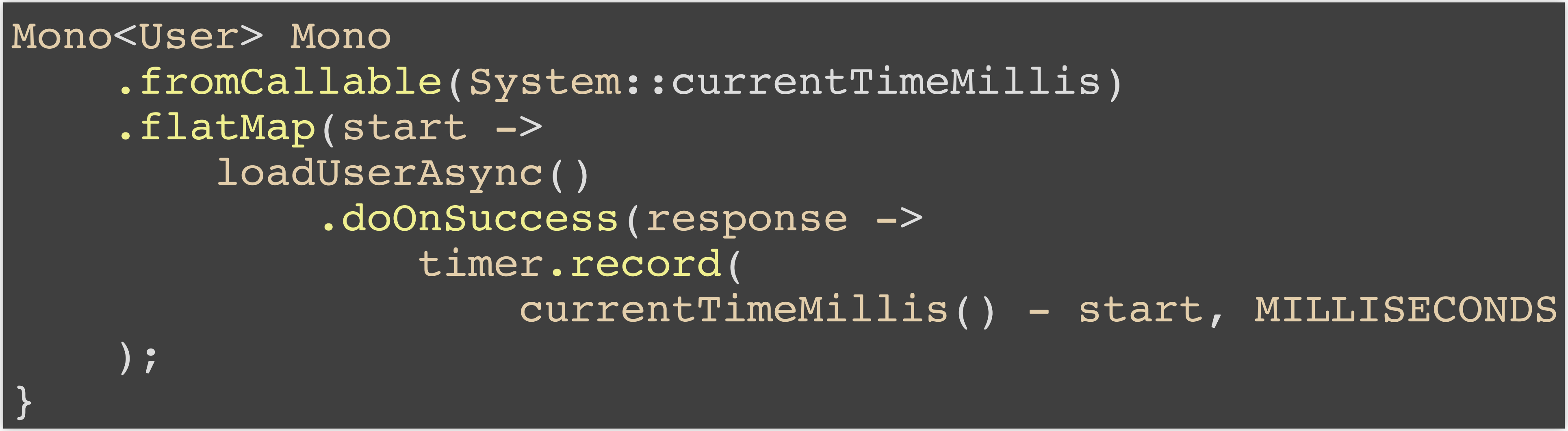
Timers are a bit counterintuitive to get right



```java
import io.micrometer.core.instrument.Timer;

var timer = Metrics.timer("timer");
//...
User user = timer.recordCallable(this::loadUser);
```

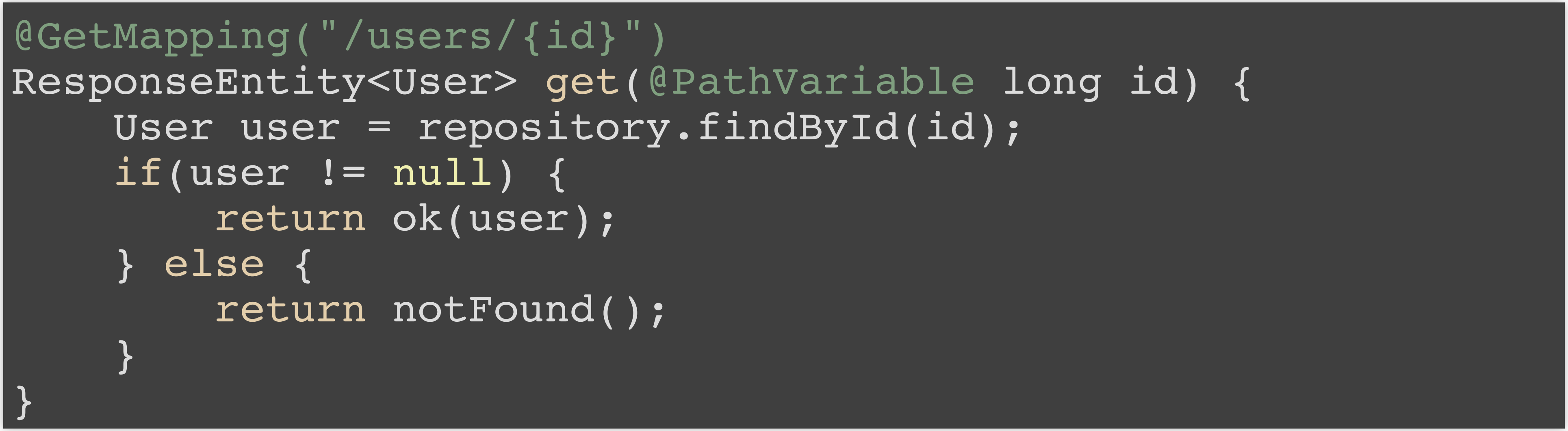
```java
Mono<User> user = timer.recordCallable(this::loadUserAsync);
```



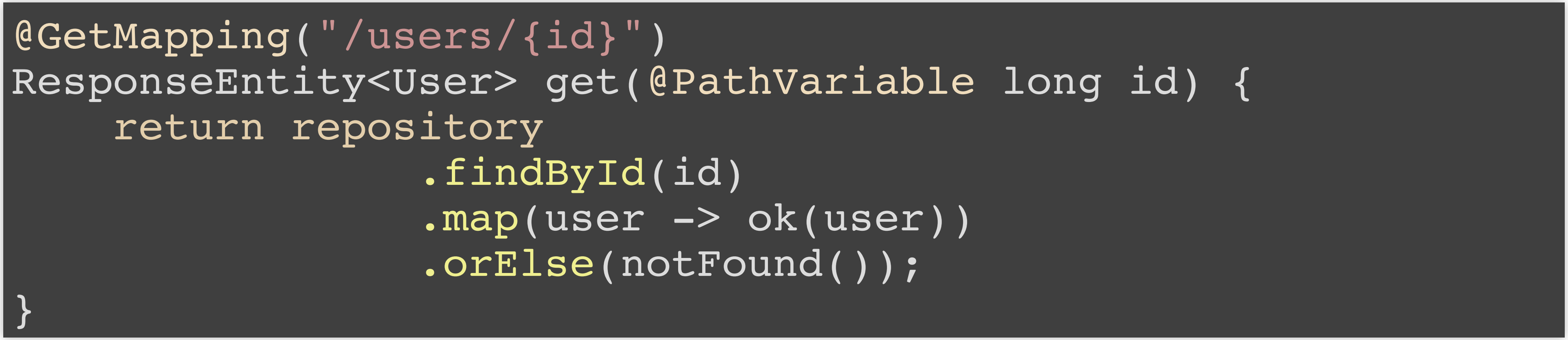
```
Mono<User> Mono
    .fromCallable(System::currentTimeMillis)
    .flatMap(start ->
        loadUserAsync()
            .doOnSuccess(response ->
                timer.record(
                    currentTimeMillis() - start, MILLISECONDS
    );
}
```

Key Takeaways

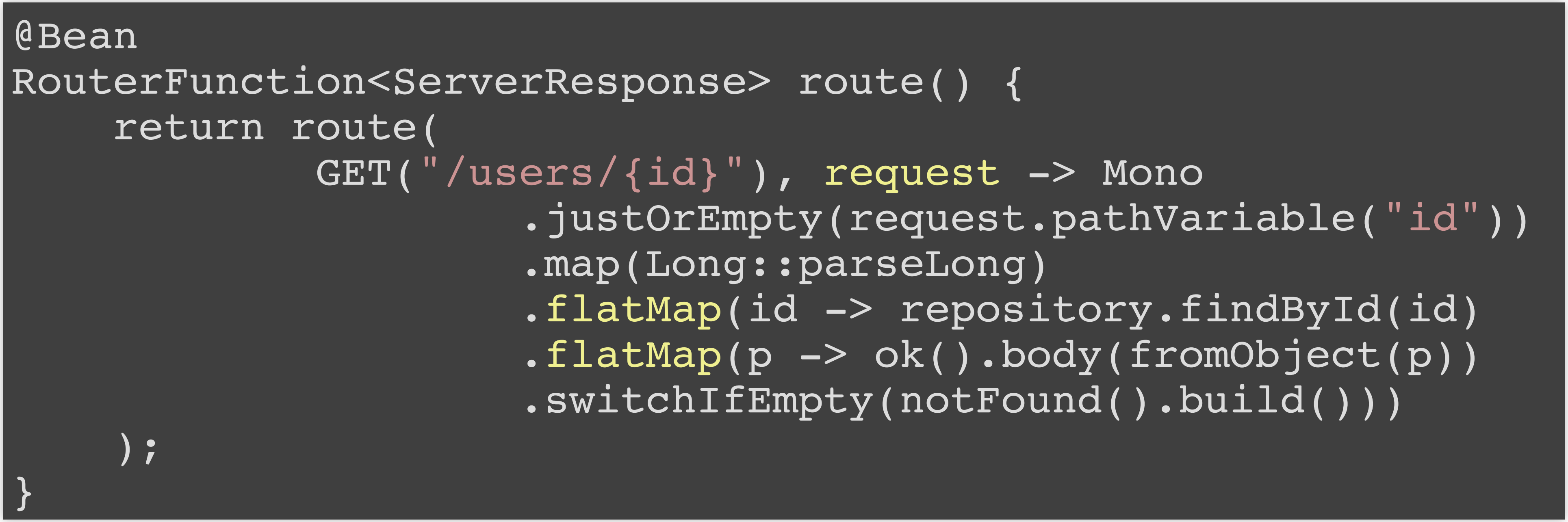
Spring Boot 2 changes everything



```
@GetMapping("/users/{id}")
ResponseEntity<User> get(@PathVariable long id) {
    User user = repository.findById(id);
    if(user != null) {
        return ok(user);
    } else {
        return notFound();
    }
}
```



```java
@GetMapping("/users/{id}")
ResponseEntity<User> get(@PathVariable long id) {
    return repository
                .findById(id)
                .map(user -> ok(user))
                .orElse(notFound());
}
```



```
@Bean
RouterFunction<ServerResponse> route() {
    return route(
            GET("/users/{id}"), request -> Mono
                    .justOrEmpty(request.pathVariable("id"))
                    .map(Long::parseLong)
                    .flatMap(id -> repository.findById(id)
                    .flatMap(p -> ok().body(fromObject(p))
                    .switchIfEmpty(notFound().build()))
    );
}
```

Reactive programming is awesome

Seriously

the explosion of latency-inducing microservices

www.infoq.com/articles/Designing-Implementing-Using-Reactive-APIs

Are you really benefitting?

Need or desire?

Make a conscious decision

Thank you!

@tnurkiewicz

nurkiewicz.github.io/talks/2018/reactive-lessons

Image sources

1. http://disney.wikia.com/wiki/Statler_and_Waldorf
2. https://twitter.com/nixcraft/status/9746458539767
3. www.pngmart.com/image/46149
4. http://half-life.wikia.com/wiki/File:HL3_logo.svg
5. http://dilbert.com/strip/2008-09-03
6. https://www.3djuegos.com/comunidad-foros/tema/45105428/0/nintendo-switch-publica-su-especificaciones-tecnicas-al-completo/