lysine.dev/okhttp/

wasmo

Deconstructing OkHttp

What’s It?

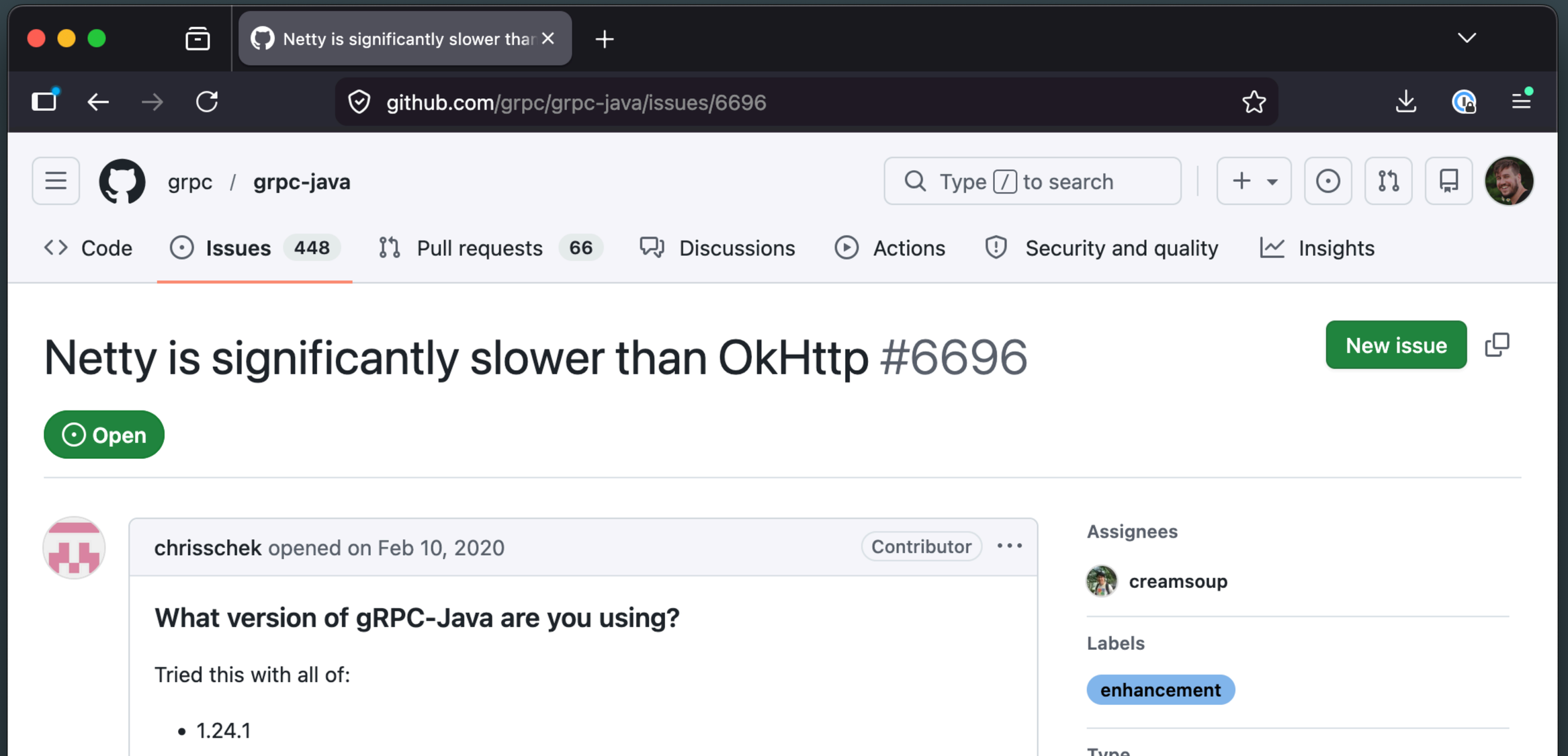
OkHttp is a library for making HTTP calls

Predates ktor

Predates Coroutines

Predates Kotlin

Secure, Fast, & Simple



How do we make it good?
By trying hard

ITEM 1

TaskRunner

Task Scheduling in OkHttp

Connection Pool	connection cleanup
Happy Eyeballs	async connect
HTTP/2	pings, pongs, acknowledgements
WebSocket	pings, pongs, messages
Disk Cache	prune

Case Study: Web Sockets

You can send messages on a web socket

You can configure a periodic ping to keep a web socket alive, perhaps every 5 seconds

The message and the ping write to the same TCP connection

Queueing Writes

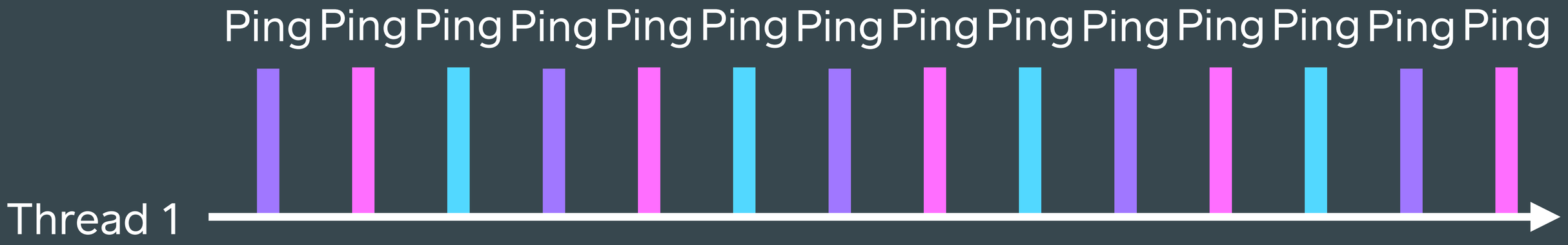
At most one thread can write to a TCP connection at a time

A program may have many web sockets

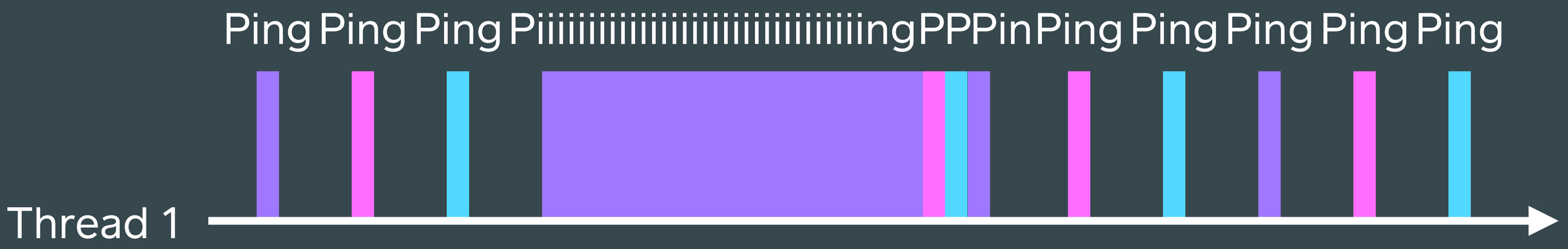
Solution 1

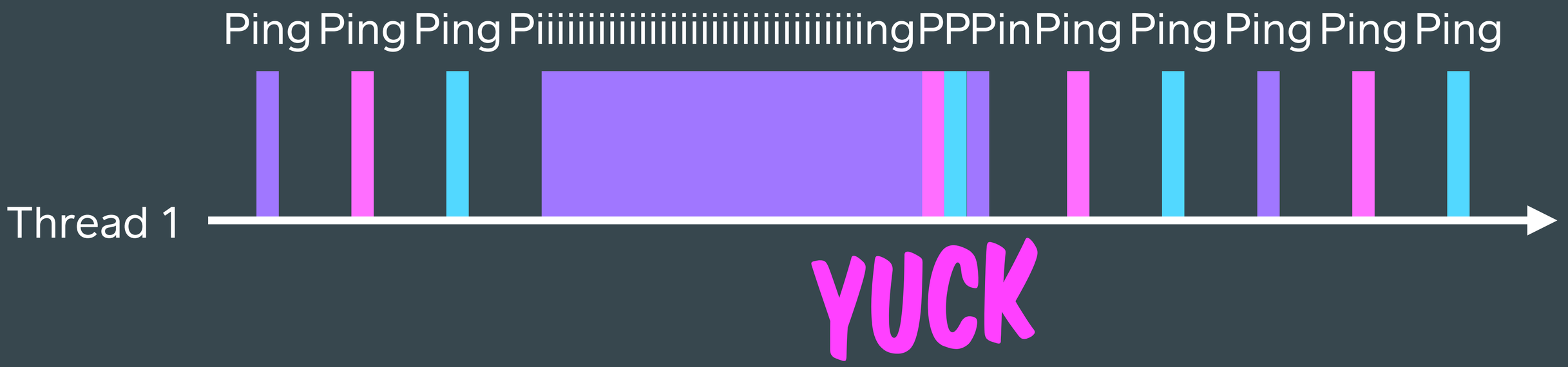
A single shared `ScheduledExecutorService`


Thread 1


Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping
Thread 1


Thread 1

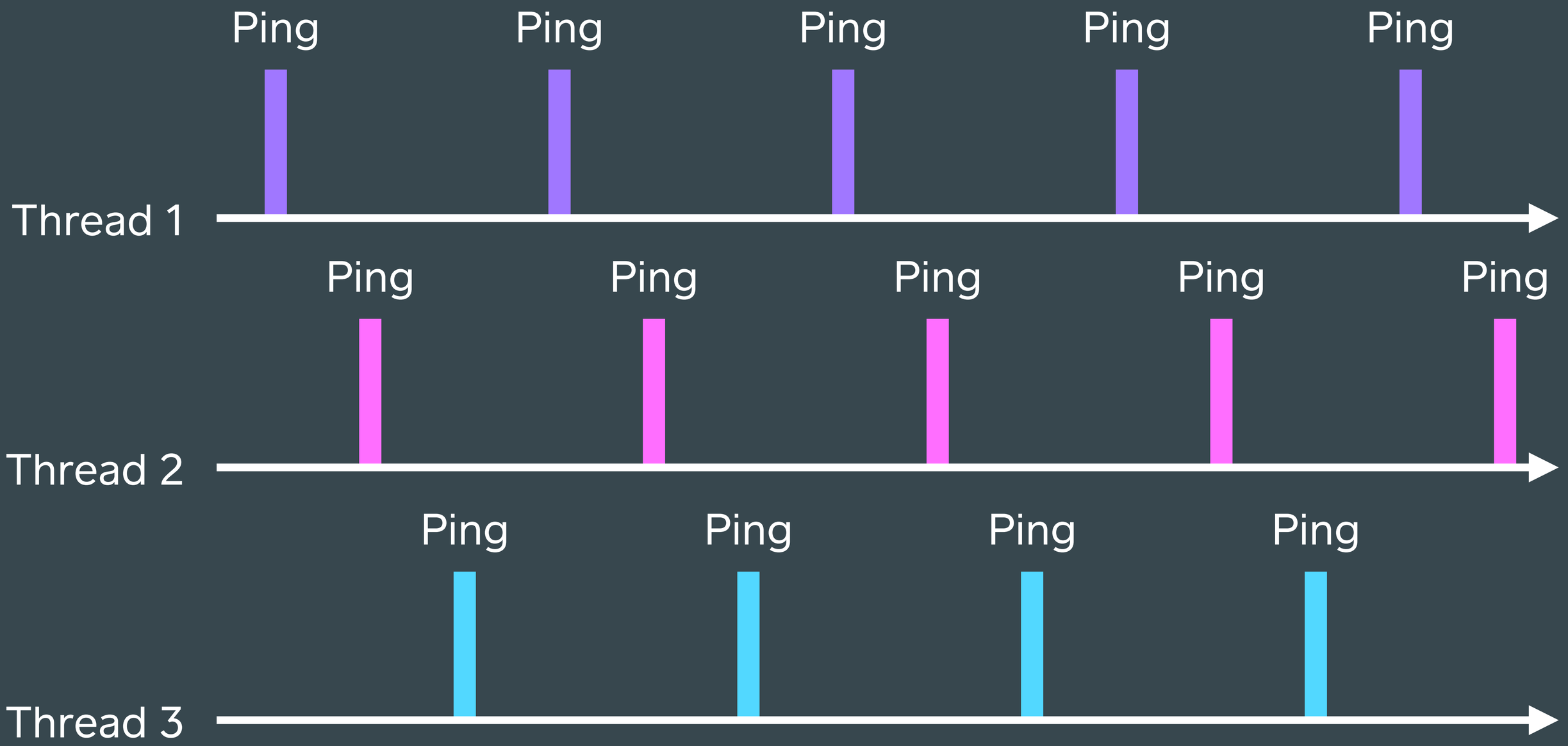

Ping Ping Ping Piiiiiiiiiiiiiiiiiiiiiiiiiiiiiiing PPPin Ping Ping Ping Ping Ping
Thread 1


Ping Ping Ping Piiiiiiiiiiiiiiiiiiiiiiiiiiiiiiing PPPin Ping Ping Ping Ping Ping
Thread 1
YUCK

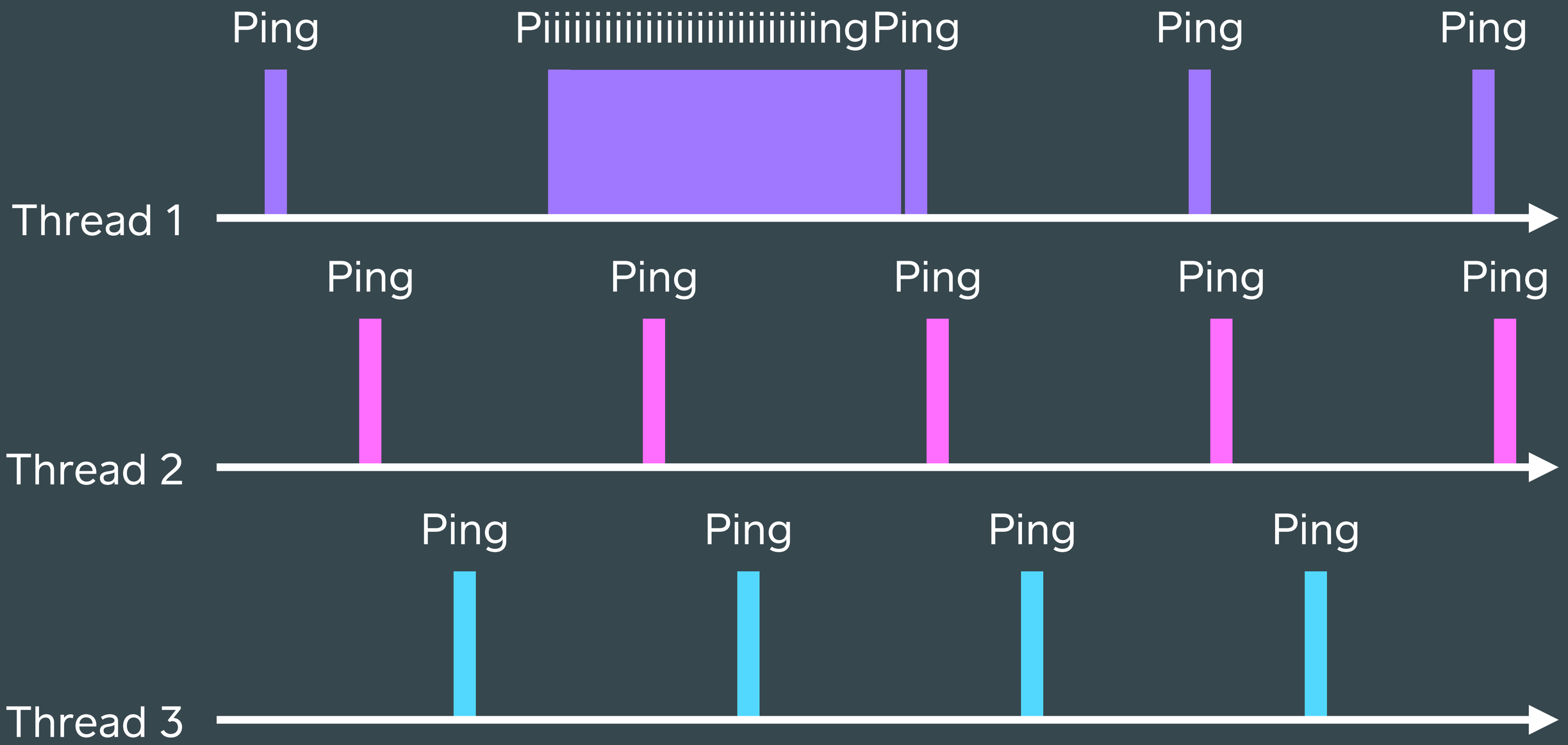
Solution 2

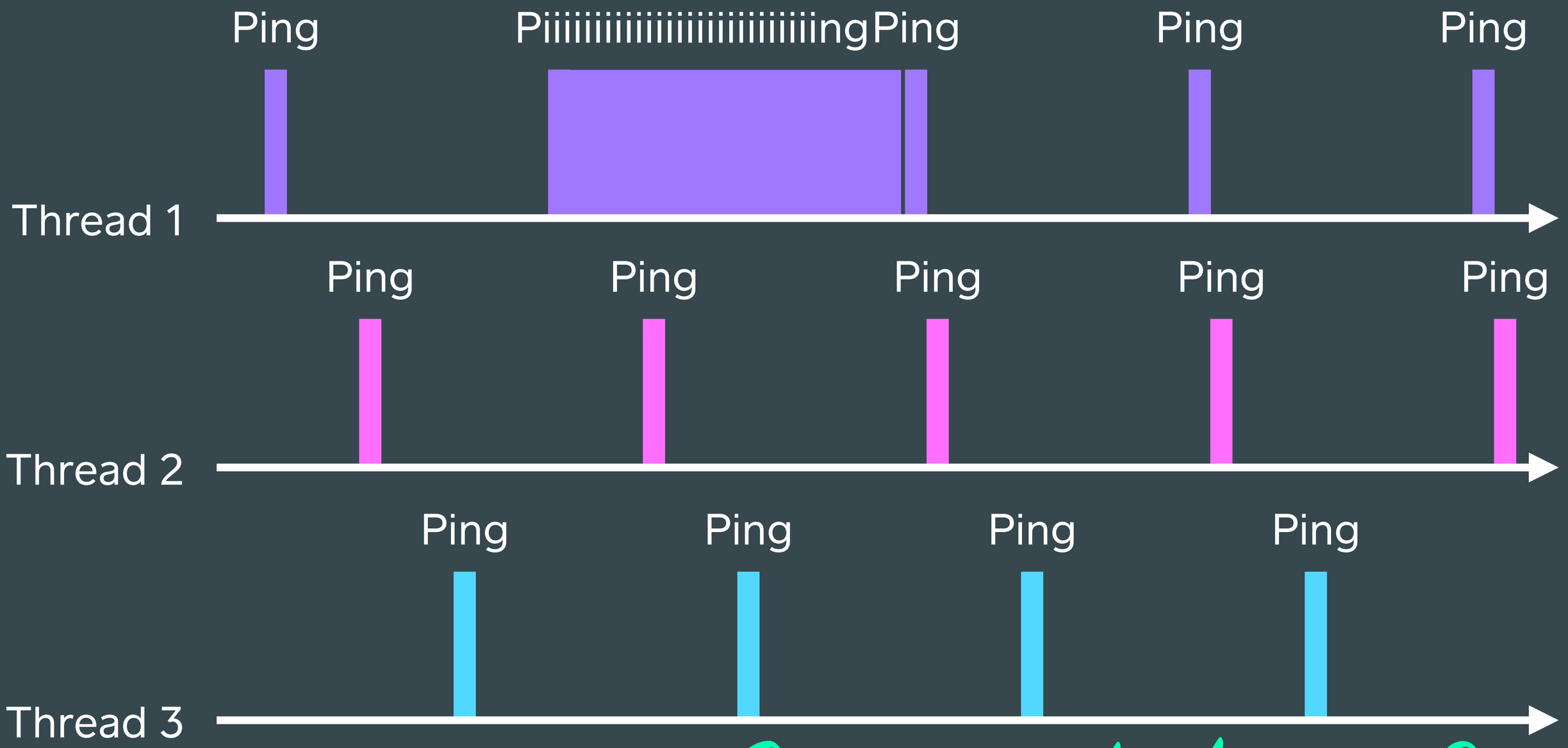
A `ScheduledExecutorService` per web socket


Thread 1
Thread 2
Thread 3


Ping
Ping
Ping
Ping
Ping
Thread 1
Ping
Ping
Ping
Ping
Ping
Thread 2
Ping
Ping
Ping
Ping
Thread 3


Thread 1
Thread 2
Thread 3


Ping
Piiiiiiiiiiiiiiiiiiiiiiiiiiing
Ping
Ping
Ping
Thread 1
Ping
Ping
Ping
Ping
Ping
Thread 2
Ping
Ping
Ping
Ping
Thread 3

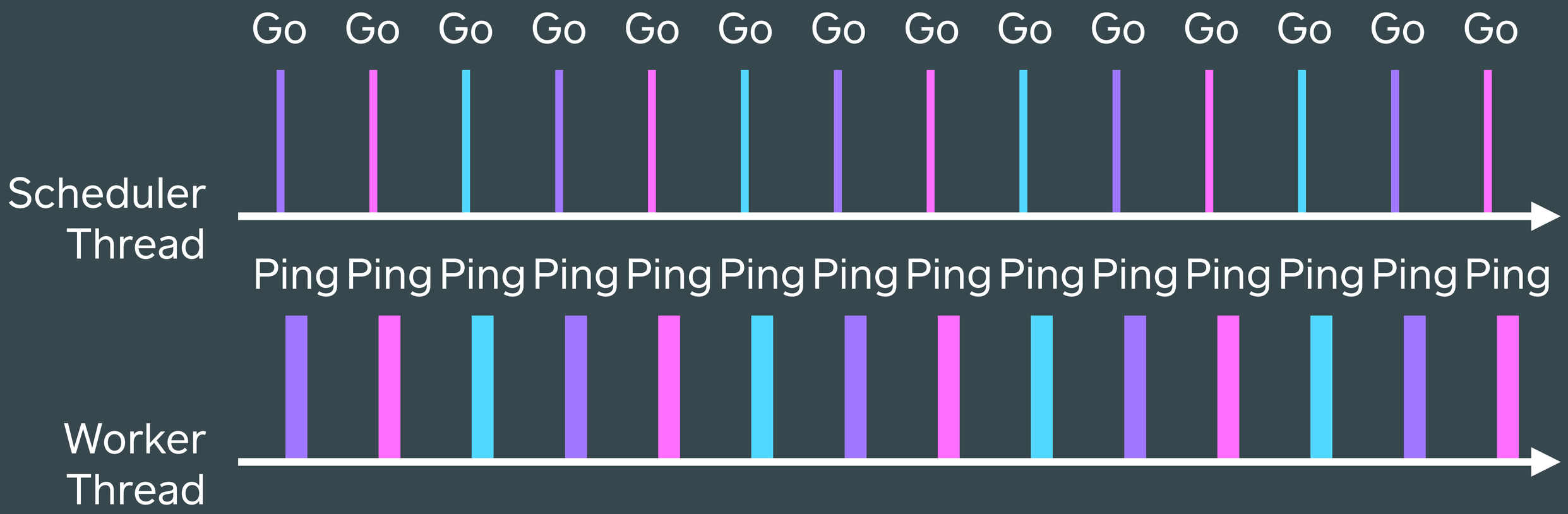


Can we do better?

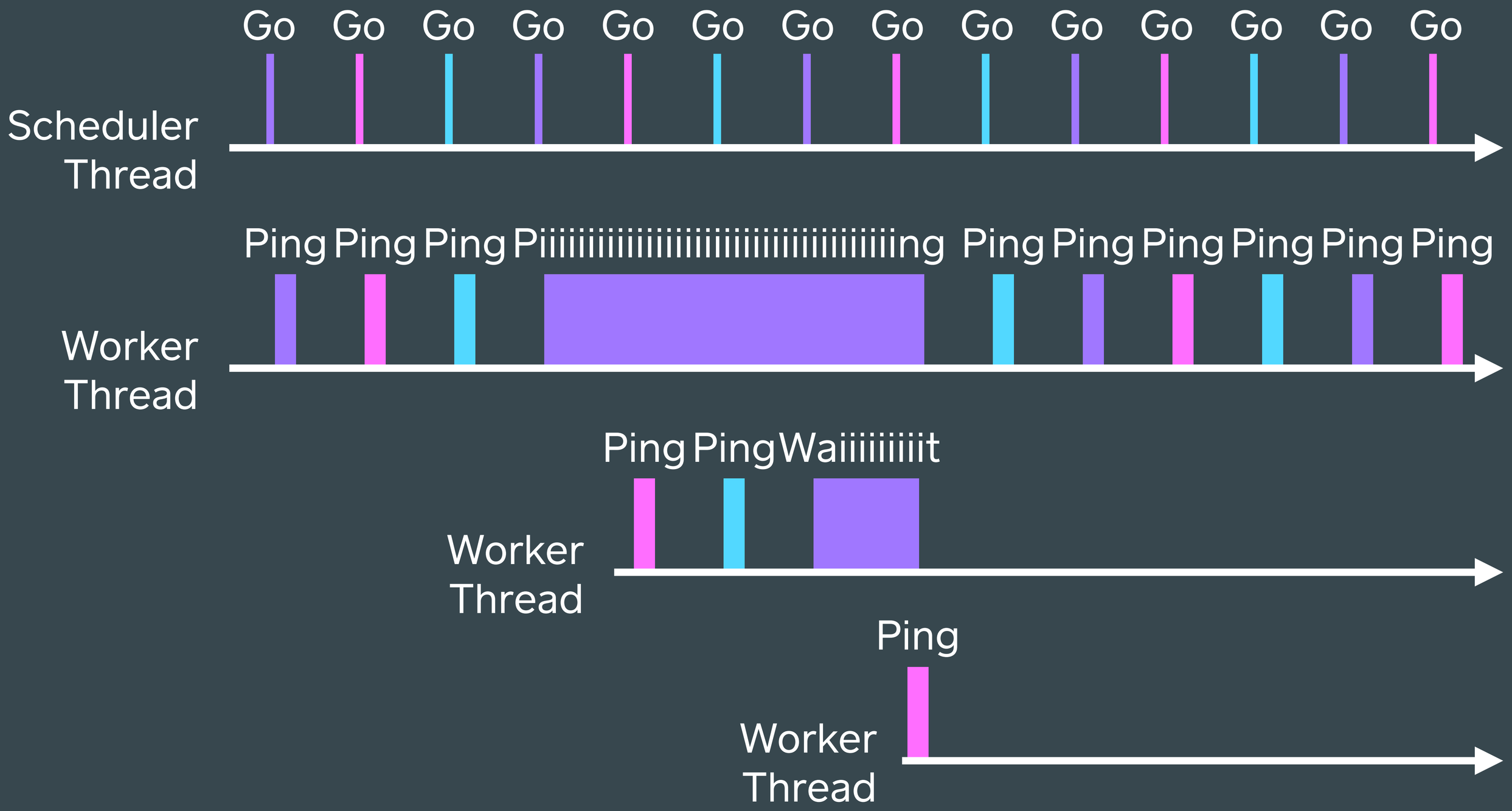
Solution 3

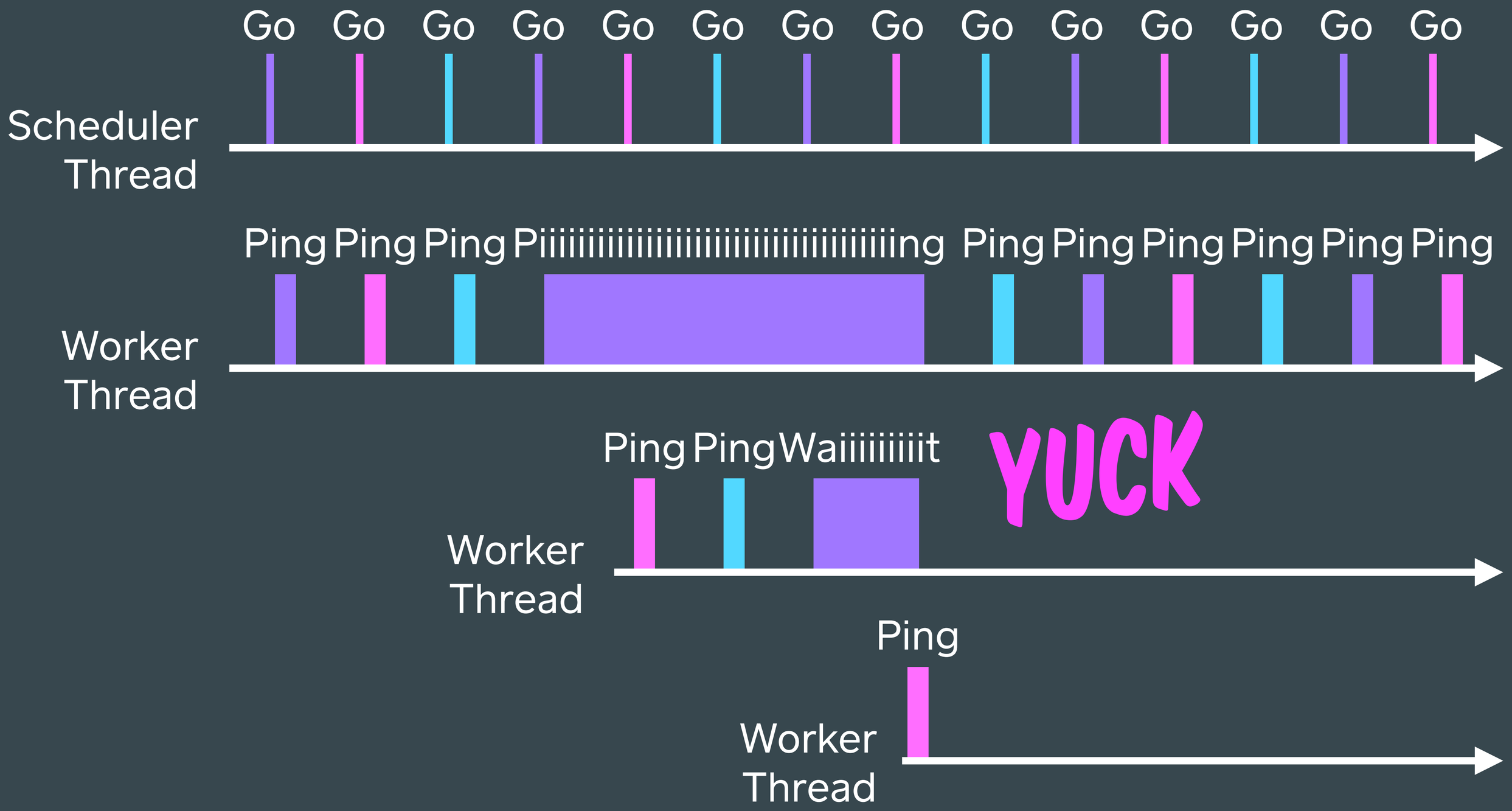
A `ScheduledExecutorService` for scheduling, plus a separate `ThreadPoolExecutorService` for execution


Scheduler
Thread
Worker
Thread


Go Go Go Go Go Go Go Go Go Go Go Go Go Go
Scheduler Thread
Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping Ping
Worker Thread


Scheduler Thread
Worker Thread


Go Go Go Go Go Go Go Go Go Go Go Go Go Go
Scheduler Thread
Ping Ping Ping Piiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiing Ping Ping Ping Ping Ping Ping
Worker Thread
Ping Ping Waiiiiiiiiit
Worker Thread
Ping
Worker Thread


Go Go Go Go Go Go Go Go Go Go Go Go Go Go
Scheduler Thread
Ping Ping Ping Piiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiing Ping Ping Ping Ping Ping Ping
Worker Thread
Ping Ping Waiiiiiiiiit
YUCK
Worker Thread
Ping
Worker Thread

Solution 4: TaskRunner

The ‘coordinator’ thread runs the next scheduled task

When it’s time to run that task, select another thread to be coordinator. This will start a thread if necessary!

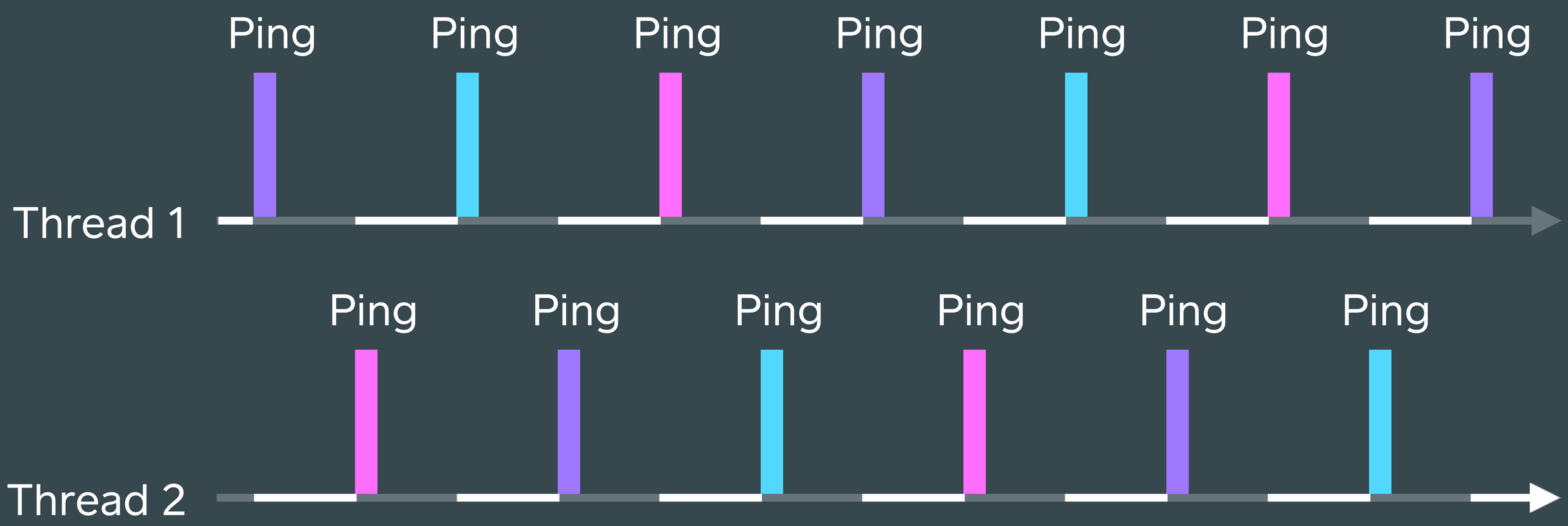
Resourceful

If a connection has a task running, all of its other tasks are ignored by the coordinator

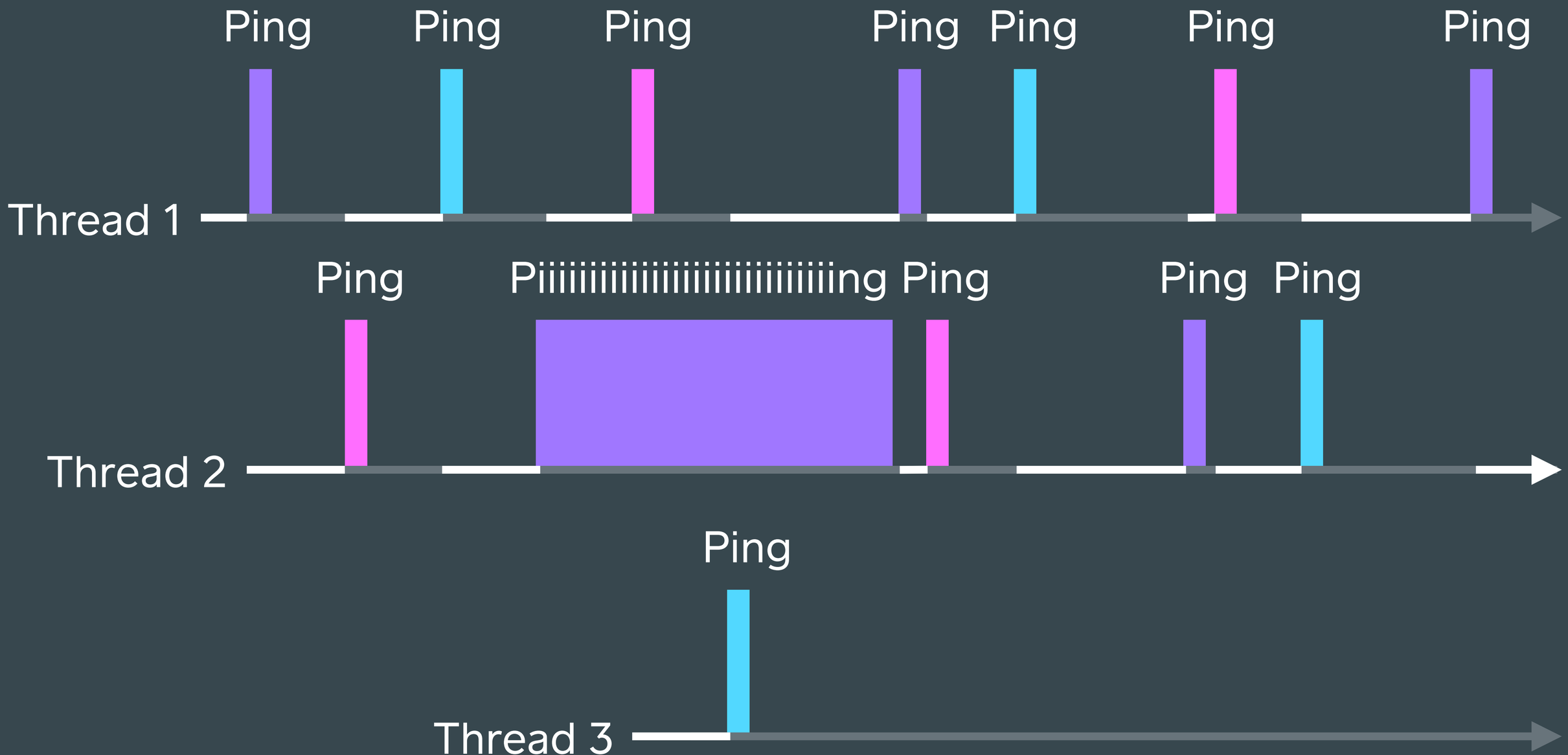
Minimum: zero threads

Maximum: one thread for each connection


Thread 1


Ping
Ping
Ping
Ping
Ping
Ping
Ping
Thread 1
Ping
Ping
Ping
Ping
Ping
Ping
Thread 2


Thread 1


Ping
Ping
Ping
Ping
Ping
Ping
Ping
Thread 1
Ping
Piiiiiiiiiiiiiiiiiiiiiiiiiiiing
Ping
Ping
Ping
Thread 2
Ping
Thread 3

Designed for Testing

We swap it out for `TaskFaker` in our tests

Tests are single-threaded and deterministic

Advice

You can just do things!

`ScheduledExecutorService` ain’t special

Dynamic is Dangerous!

Imagine a program that’s maintaining 1,000 web socket connections, using 1,001 threads

When the network slows down, the thread count goes up!

ITEM 2

Sockets

What’s a Socket?

A TCP connection between a pair of computers

Duplex: both sides transmit to each other simultaneously

Java’s `java.net.Socket`

Where’s it Used?

HTTP/1 and HTTP/2 are *layered* over TCP sockets

Since OkHttp 5.2, in a protocol upgrade

Java's Socket

CONNECT

DISCONNECT

DATA STREAMS

CONNECTION STATE

TCP SETTINGS

SWAP IMPLEMENTATION

```
public class Socket extends Closeable {
    public void connect(SocketAddress endpoint) throws IOException;
    public void connect(SocketAddress endpoint, int timeout) throws IOException;
    public void bind(SocketAddress bindpoint) throws IOException;


    public synchronized void close() throws IOException;
    public void shutdownInput() throws IOException;
    public void shutdownOutput() throws IOException;
    public boolean isConnected();
    public boolean isBound();
    public boolean isClosed();
    public boolean isInputShutdown();
    public boolean isOutputShutdown();


    public InputStream getInputStream() throws IOException;
    public OutputStream getOutputStream() throws IOException;


    public InetAddress getInetAddress();
    public InetAddress getLocalAddress();
    public int getPort();
    public int getLocalPort();
    public SocketAddress getRemoteSocketAddress();
    public SocketAddress getLocalSocketAddress();
    public SocketChannel getChannel();

    public void setTcpNoDelay(boolean on) throws SocketException;
    public boolean getTcpNoDelay() throws SocketException;
    public void setSoLinger(boolean on, int linger) throws SocketException;
    public int getSoLinger() throws SocketException;
    public void sendUrgentData(int data) throws IOException;
    public void setOOBInline(boolean on) throws SocketException;
    public boolean getOOBInline() throws SocketException;
    public synchronized void setSoTimeout(int timeout) throws SocketException;
    public synchronized int getSoTimeout() throws SocketException;
    public synchronized void setSendBufferSize(int size) throws SocketException;
    public synchronized int getSendBufferSize() throws SocketException;
    public synchronized void setReceiveBufferSize(int size) throws SocketException;
    public synchronized int getReceiveBufferSize() throws SocketException;
    public void setKeepAlive(boolean on) throws SocketException;
    public boolean getKeepAlive() throws SocketException;
    public void setTrafficClass(int tc) throws SocketException;
    public int getTrafficClass() throws SocketException;
    public void setReuseAddress(boolean on) throws SocketException;
    public boolean getReuseAddress() throws SocketException;
    public void setPerformancePreferences(int connectionTime, int latency, int bandwidth);
    public <T> java.net.Socket setOption(SocketOption<T> name, T value) throws IOException;
    public <T> T getOption(SocketOption<T> name) throws IOException;
    public Set<SocketOption<?>> supportedOptions();


    public static synchronized void setSocketImplFactory(SocketImplFactory fac) throws IOException;
}
```

Okio's Socket

DATA STREAMS

DISCONNECT

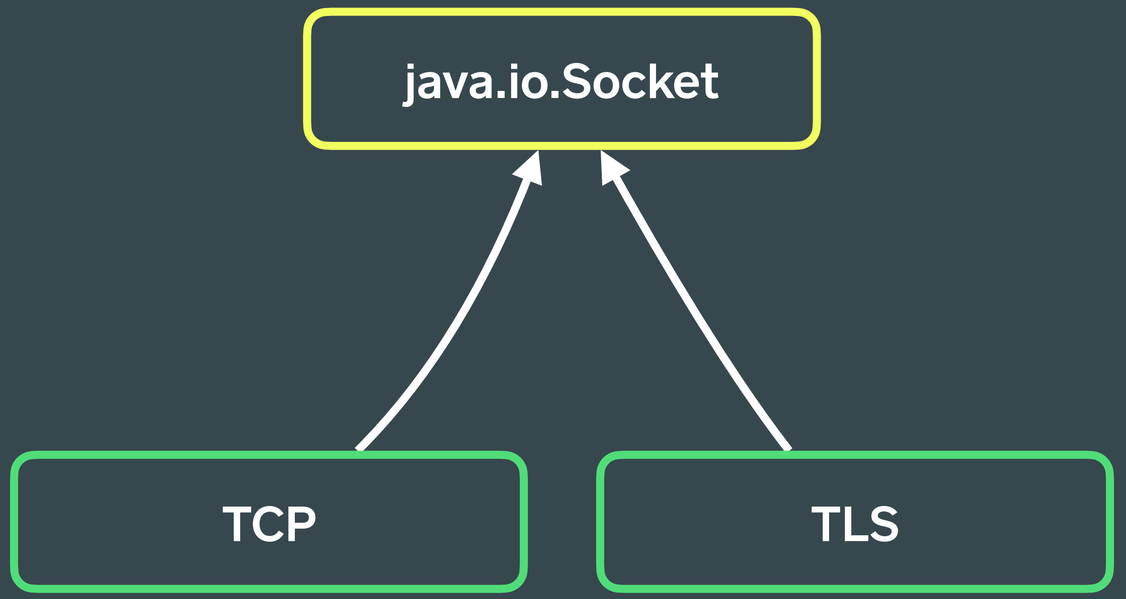
```
interface Socket {
  val source: Source
  val sink: Sink

  fun cancel()
}
```

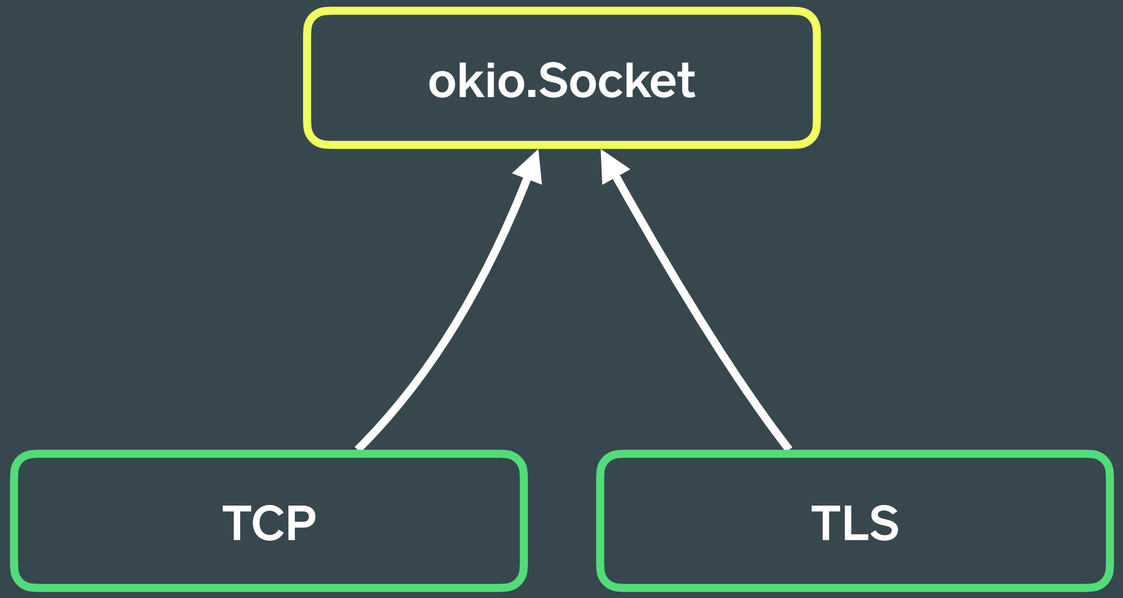
why tho

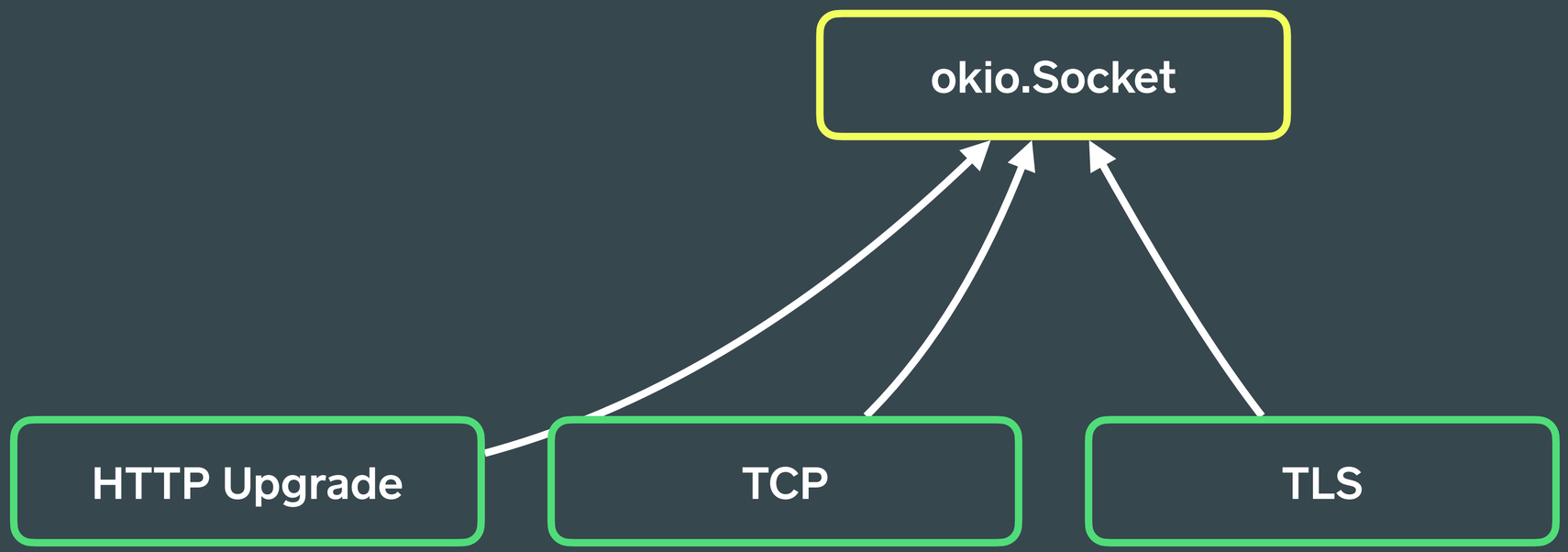
What happens when you add a function to an interface?

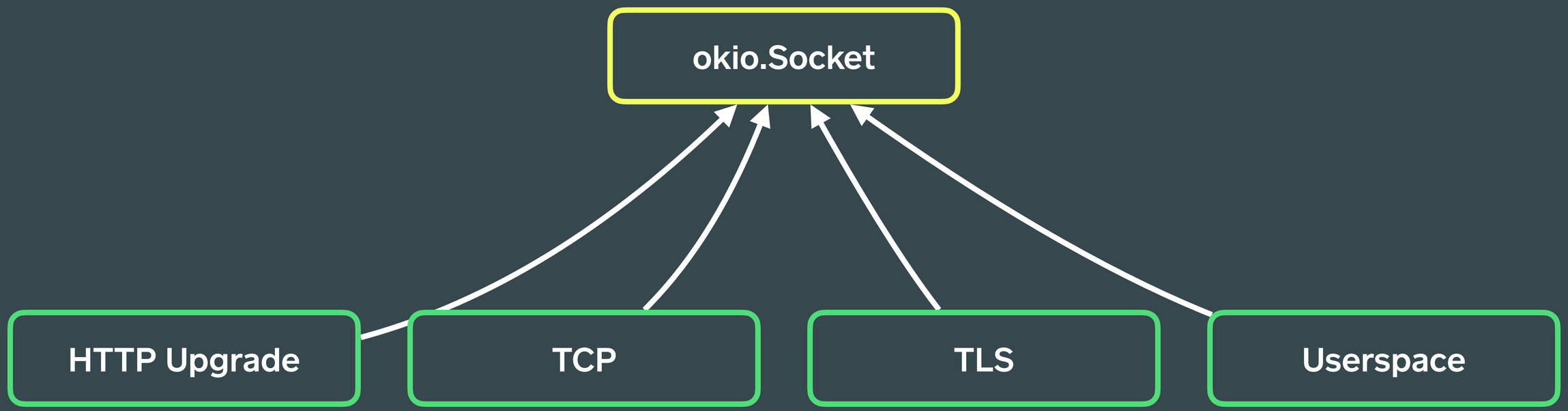
- Widen the capabilities of the callers
- Narrow what implementations are possible

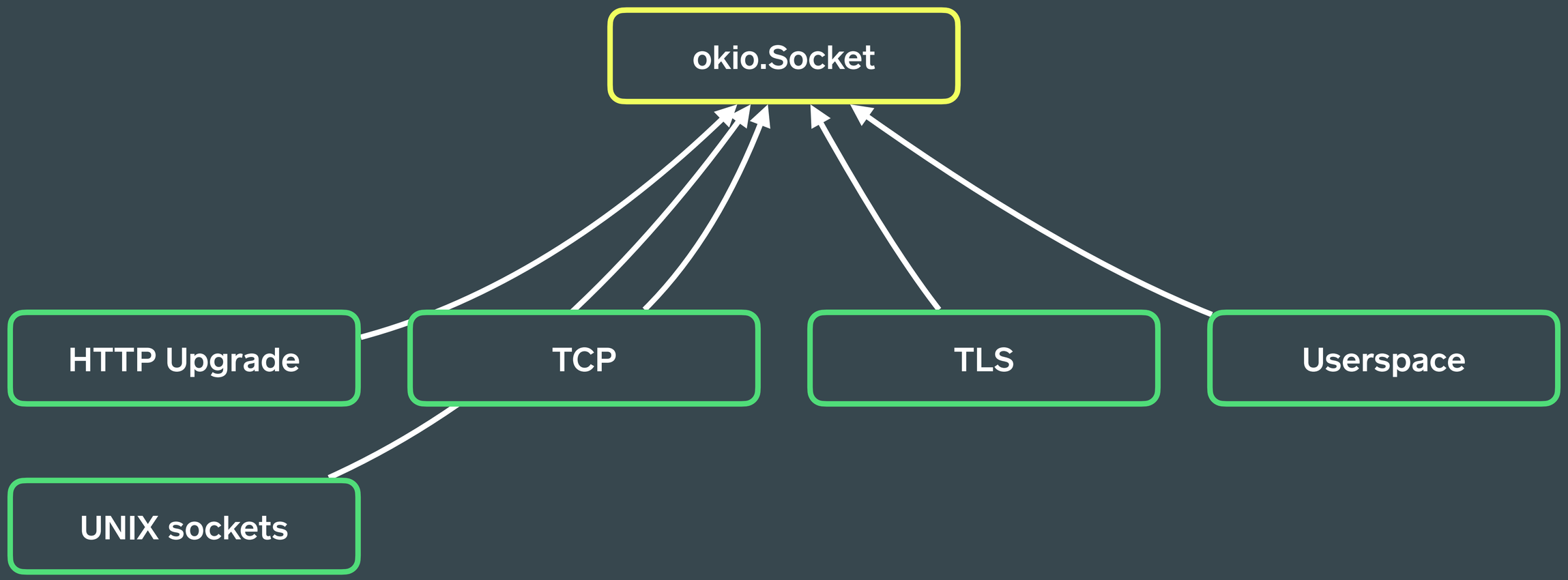
java.io.Socket

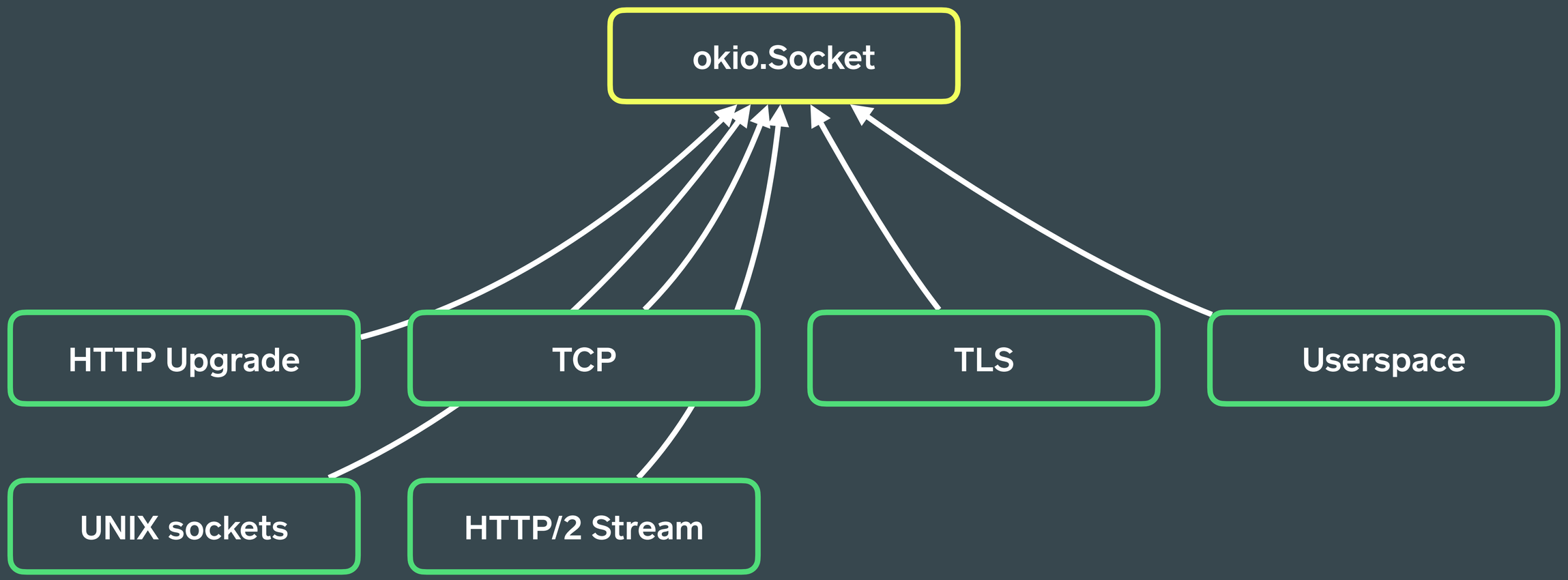

java.io.Socket
TCP
TLS

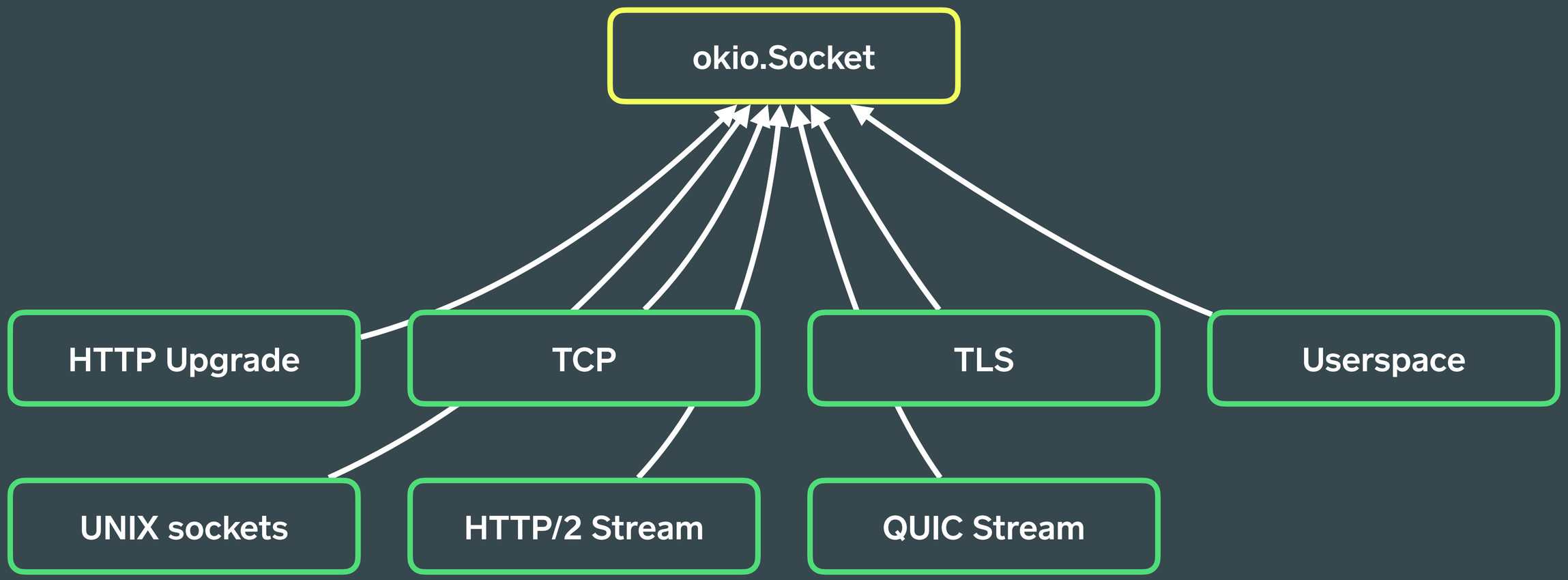

okio.Socket
TCP
TLS

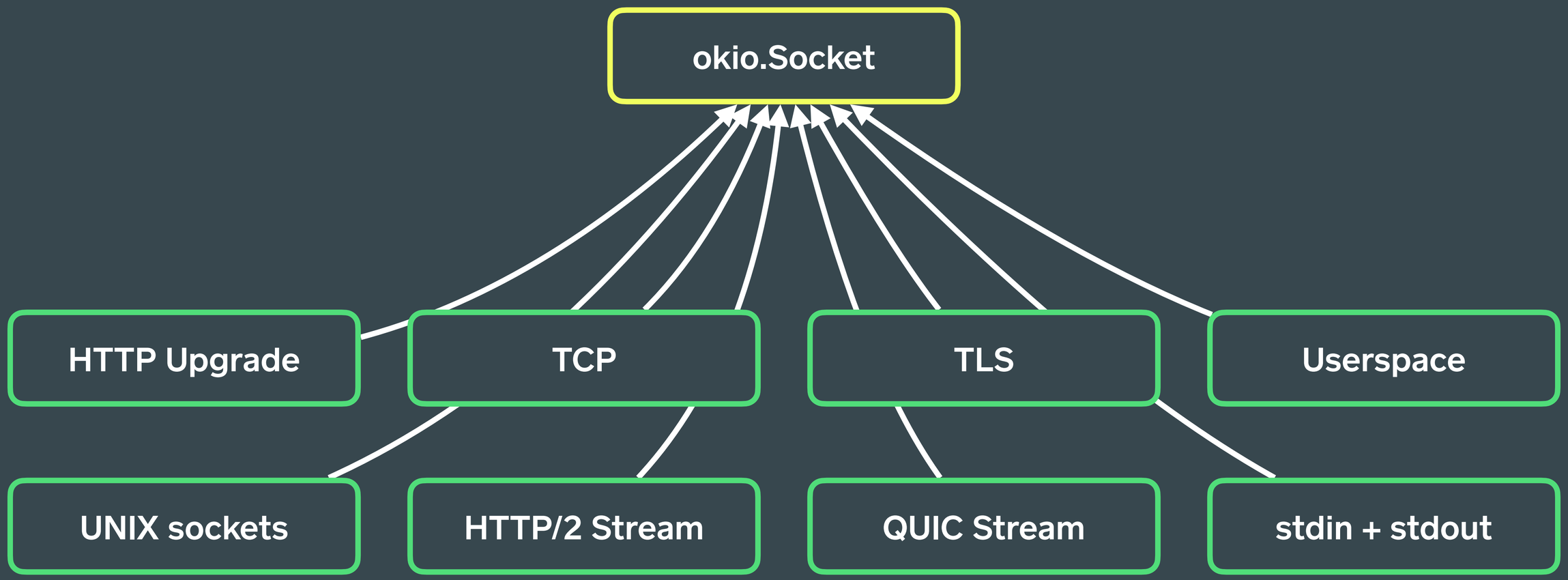

okio.Socket
HTTP Upgrade
TCP
TLS

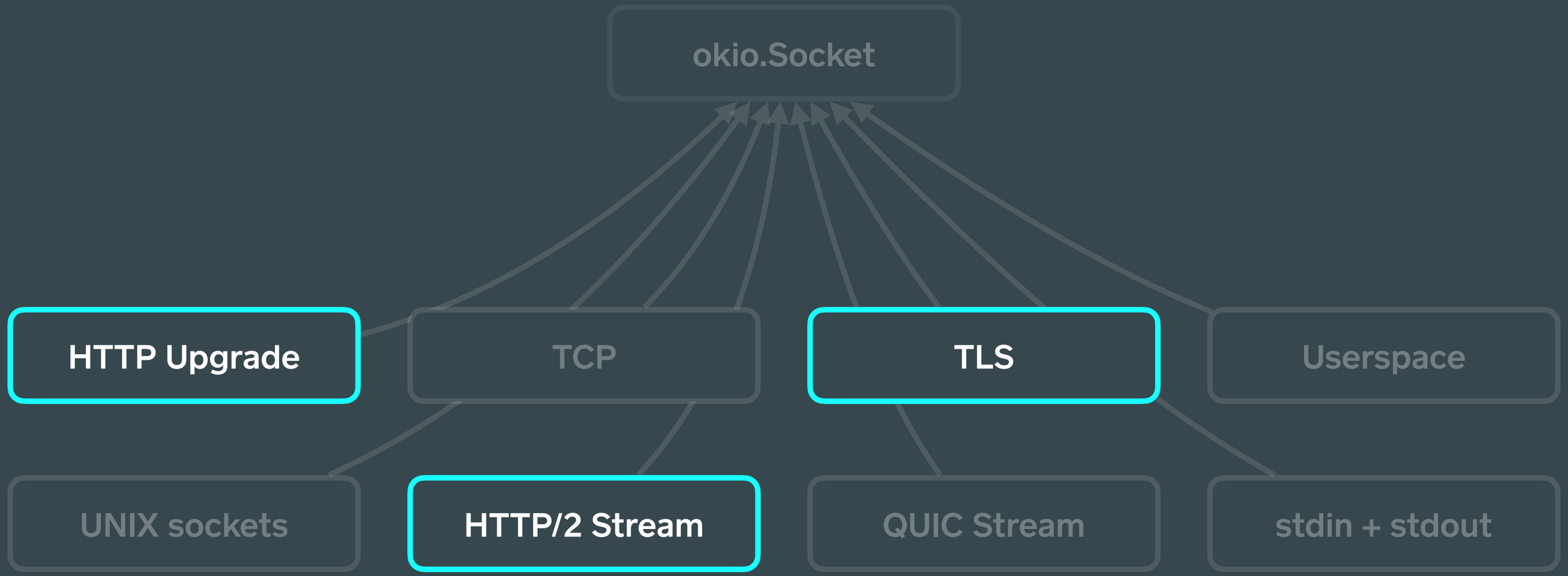

okio.Socket
HTTP Upgrade
TCP
TLS
Userspace


okio.Socket
HTTP Upgrade
TCP
TLS
Userspace
UNIX sockets


okio.Socket
HTTP Upgrade
TCP
TLS
Userspace
UNIX sockets
HTTP/2 Stream


okio.Socket
HTTP Upgrade
TCP
TLS
Userspace
UNIX sockets
HTTP/2 Stream
QUIC Stream


okio.Socket
HTTP Upgrade
TCP
TLS
Userspace
UNIX sockets
HTTP/2 Stream
QUIC Stream
stdin + stdout


okio.Socket
HTTP Upgrade
TCP
TLS
Userspace
UNIX sockets
HTTP/2 Stream
QUIC Stream
stdin + stdout

Cancel vs. Close

Cancel:

Abort an operation because it's no longer needed

Can be asynchronous!

Close:

Release resources after an operation completes

Java's Socket does both?!

Future Plans

OkHttp always uses an operating system socket, even in tests

Switching to `inMemorySocketPair()` could make our tests run faster

And maybe yours too, if you’re using MockWebServer

ITEM 3

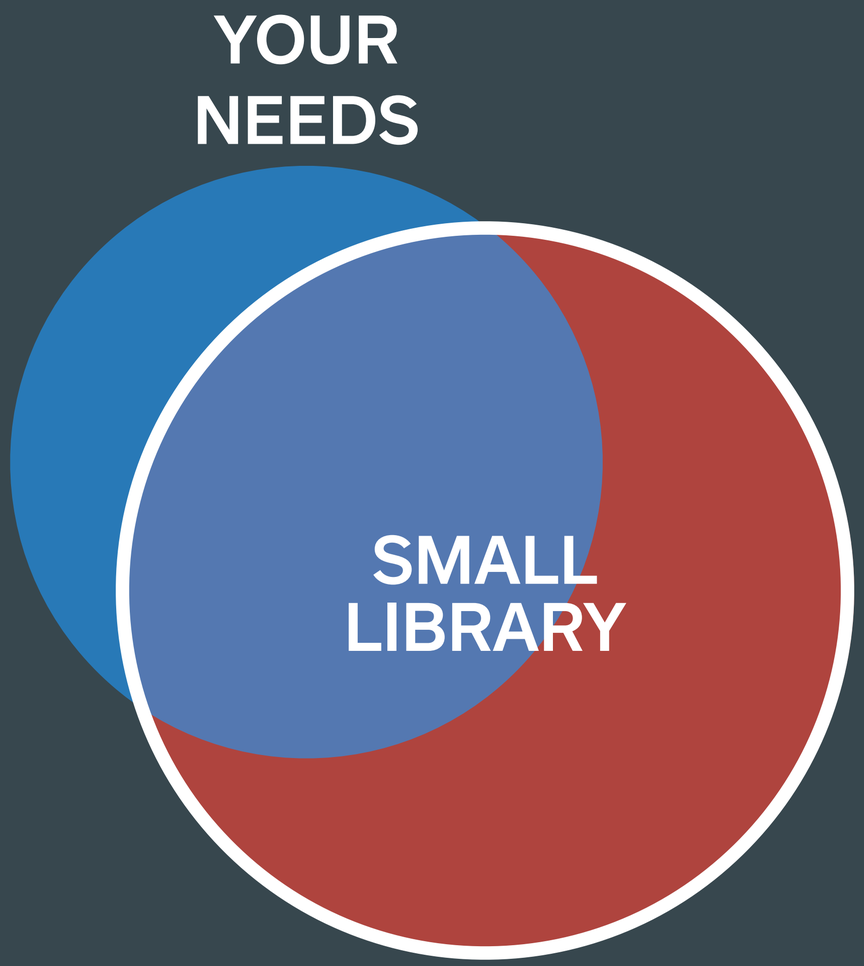
Extensibility

Why?

Writing extensible APIs give us an escape hatch: we don’t need to add a ton of features to serve a ton of users

It’s more fun!

SMALL LIBRARY


YOUR
NEEDS
SMALL
LIBRARY


YOUR NEEDS
JEN'S NEEDS
SMALL LIBRARY


YOUR NEEDS
JEN'S NEEDS
SMALL LIBRARY
MARCOS' NEEDS


YOUR NEEDS
JEN'S NEEDS
SMALL LIBRARY
MATT'S NEEDS
MARCOS' NEEDS


YOUR NEEDS
JEN'S NEEDS
BIG LIBRARY
MATT'S NEEDS
MARCOS' NEEDS


FEATURES
YOU NEED
FEATURES YOU
DON’T NEED

ITEM 4

Interceptors

OkHttp History

2011 Android's HTTP Clients post on the Android Developers Blog

2013 Announcing OkHttp on the Square Engineering Blog

2014 OkHttp 2.2 introduces interceptors
Our API draws on a project called AOP Alliance (2003)

2016 OkHttp is radically refactored around interceptors

Application	headers, logging, etc.
Retry and Follow Up	redirects and auth challenges
Bridge	add cookies, user-agent, gzip
Cache	disk, network, or both
Connect	connect TCP, TLS, Happy Eyeballs
Network	more headers, logging, etc.
Call Server	write request & read response

This is Weird!

We build interceptors as an extension mechanism

Not to serve as our internal architecture

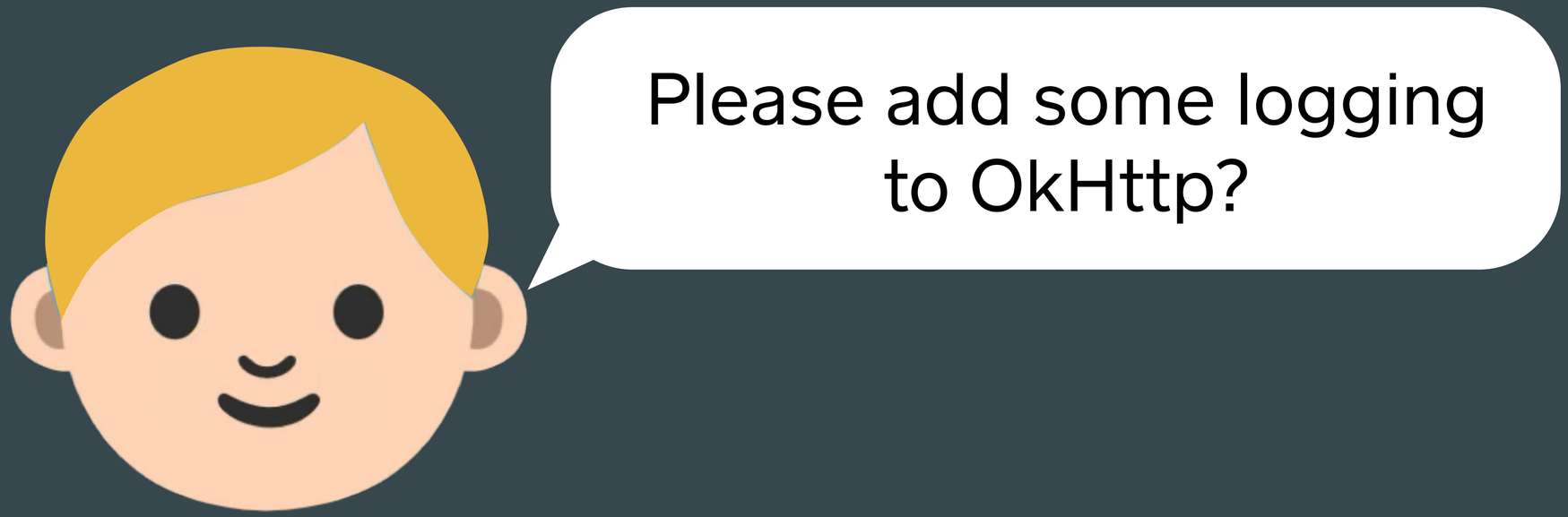
But they ended up being very well suited to that

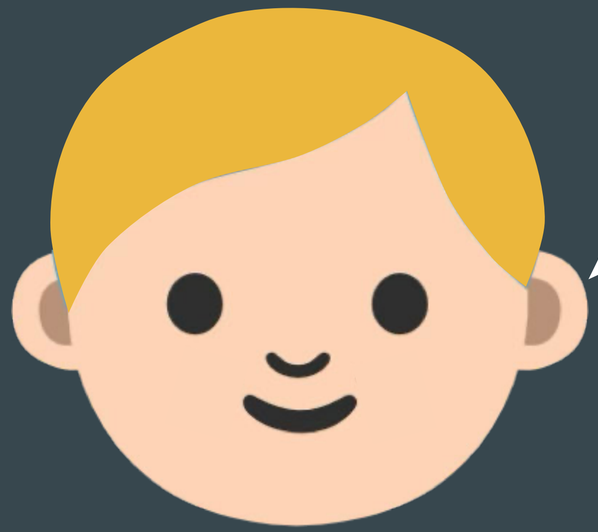
More Powerful Now

```
interface Interceptor {
  ...
  interface Chain {
    ...
    fun withAuthenticator(authenticator: Authenticator): Chain
    fun withCache(cache: Cache?): Chain
    fun withCertificatePinner(certificatePinner: CertificatePinner): Chain
    fun withConnectTimeout(timeout: Int, unit: TimeUnit): Chain
    fun withConnectionPool(connectionPool: ConnectionPool): Chain
    fun withCookieJar(cookieJar: CookieJar): Chain
    fun withDns(dns: Dns): Chain
    fun withHostnameVerifier(hostnameVerifier: HostnameVerifier): Chain
    fun withProxy(proxy: Proxy?): Chain
    fun withProxyAuthenticator(proxyAuthenticator: Authenticator): Chain
    fun withProxySelector(proxySelector: ProxySelector): Chain
    fun withReadTimeout(timeout: Int, unit: TimeUnit): Chain
    fun withRetryOnConnectionFailure(retryOnConnectionFailure: Boolean): Chain
    fun withSocketFactory(socketFactory: SocketFactory): Chain
    fun withSslSocketFactory(sslSocketFactory: SSLSocketFactory?, x509TrustManager: X509TrustManager?): Chain
    fun withWriteTimeout(timeout: Int, unit: TimeUnit): Chain
  }
}
```

ITEM 5

EventListener

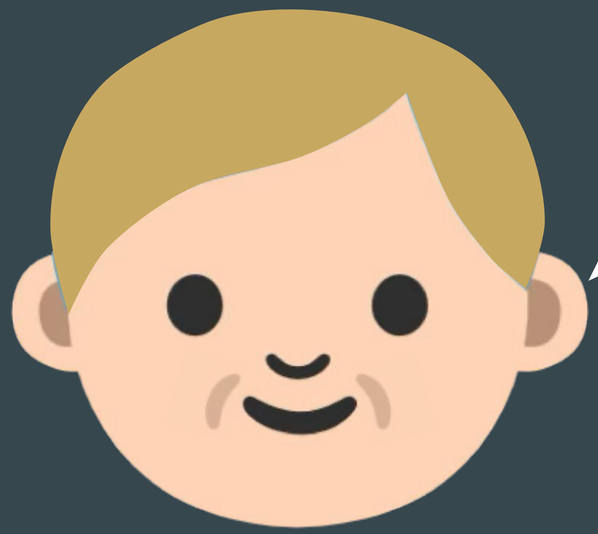

Please add some logging
to OkHttp?



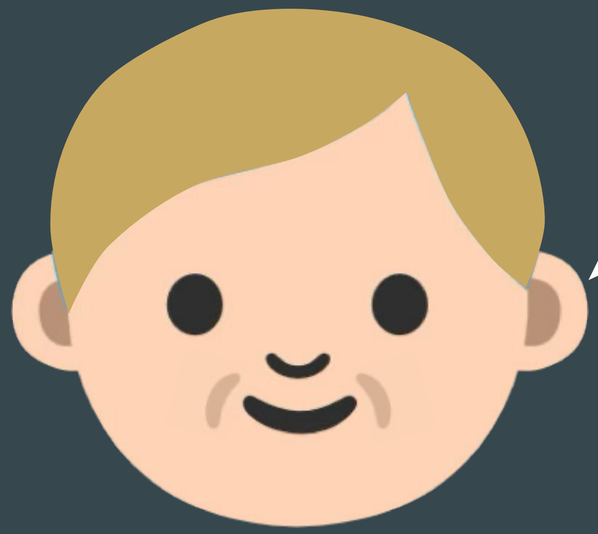
Please add some logging to OkHttp?

Logging? What for

Logging? What for



I’m trying to debug why a response isn’t cached

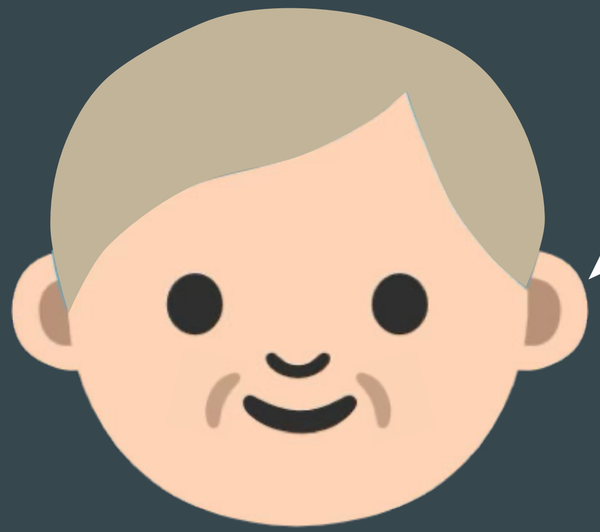


I’m trying to debug why a response isn’t cached

Ahh... you want an observability system?

Ahh... you want an observability system?

As long as I can tell why this response isn’t cached



As long as I can tell why this response isn't cached

We just shipped EventListener. Hooray!

We just shipped EventListener. Hooray!

I retired from programming 3 years ago

First Instinct: Enable Logging

It’d be so easy to add a System Property or Environment Variable that causes OkHttp to write what it’s up to to `System.out`.

But … logging is the worst form of observability!

The Worst Form of Observability?

Invisible to operational health

Invisible to business health

Inadequate context

Expensive

Insecure

Do not log

march 11, 2020 12 mins read start a discussion edit this page

Almost every week I accidentally get into this logging argument. Here's the problem: people tend to log different things and call it a best-practice. And I am not sure why. When I start discussing this with other people I always end up repeating the exact same ideas over and over again.

So. Today I want to criticize the whole logging culture and provide a bunch of alternatives.

Logging does not make sense

Let's start with the most important one. Logging does not make any sense!

Let's review a popular code sample that you can find all across the internet:

```
try:
```

Logging, But Good

```
abstract class EventListener {
  open fun callStart(call: Call)
  open fun callEnd(call: Call)
  open fun callFailed(call: Call, ioe: IOException)

  open fun dnsStart(call: Call, domainName: String)
  open fun dnsEnd(call: Call, domainName: String, inetAddressList: List<InetAddress>)

  open fun requestHeadersStart(call: Call)
  open fun requestHeadersEnd(call: Call, request: Request)

  ...

  open fun canceled(call: Call)
}
```

What’s it for?

Hook it up to Perfetto, Datadog, Open Telemetry, etc.

For operational metrics like *‘P99 response time on images’*

For user journeys like *‘Jenn waited 3.1 seconds for /send-payment’*

For incidents like, *‘TLS handshakes to the Toronto Datacenter failed’*

Refactor OkHttp, Again

Our code was messy!

For example, does OkHttp make a DNS request before it searches the connection pool? Or after? Or both?!

We added events, then we refactored so the events made sense

Advice

Create your own `EventListener` interfaces

Make observability assertions in your test cases!

ITEM 6

Tags

Tags API

```
interface Call {

  fun <T : Any> tag(type: KClass<T>, computeIfAbsent: () -> T): T

  fun <T : Any> tag(type: KClass<T>): T?

  ...
}
```

Use Cases

Observability: Link trace IDs across interceptors and event listeners

Integration: Retrofit attaches an `Invocation` object as a tag

Security: Isolate caches, cookies, or connection pools based on a User tag

Implementation

We want a `Map<KClass<T>, T>`

With concurrent reads & writes

Optimized for very small size

Solution 1

Just use `ConcurrentHashMap`

Solution 1

Just use `ConcurrentHashMap`

Can we do better?

Solution 2

Borrow ideas from `CoroutineContext`!

Two `Tags` classes:

- `EmptyTags`: 0 elements
- `LinkedTags`: A single entry and another `Tags`

```
interface Tags {
  fun <T : Any> plus(key: KClass<T>, value: T?): Tags
  fun <T : Any> get(key: KClass<T>): T?
}

object EmptyTags : Tags {
  ...
}

class LinkedTags<K : Any>(
  val key: KClass<K>,
  val value: K,
  val next: Tags,
) : Tags {
  ...
}
```

```kotlin
internal fun <T : Any> AtomicReference<Tags>.computeIfAbsent(
  type: KClass<T>,
  compute: () -> T,
): T {
  var computed: T? = null

  while (true) {
    val tags = get()

    // If the element is already present. Return it.
    val existing = tags[type]
    if (existing != null) return existing

    if (computed == null) {
      computed = compute()
    }

    // If we successfully add the computed element, we're done.
    val newTags = tags.plus(type, computed)
    if (compareAndSet(tags, newTags)) return computed

    // We lost the race. Possibly to other code that was putting a *different* key. Try again!
  }
}
```

Linked Tags

Single global empty object in the common case

Callsite keeps an `AtomicReference<Tags>`

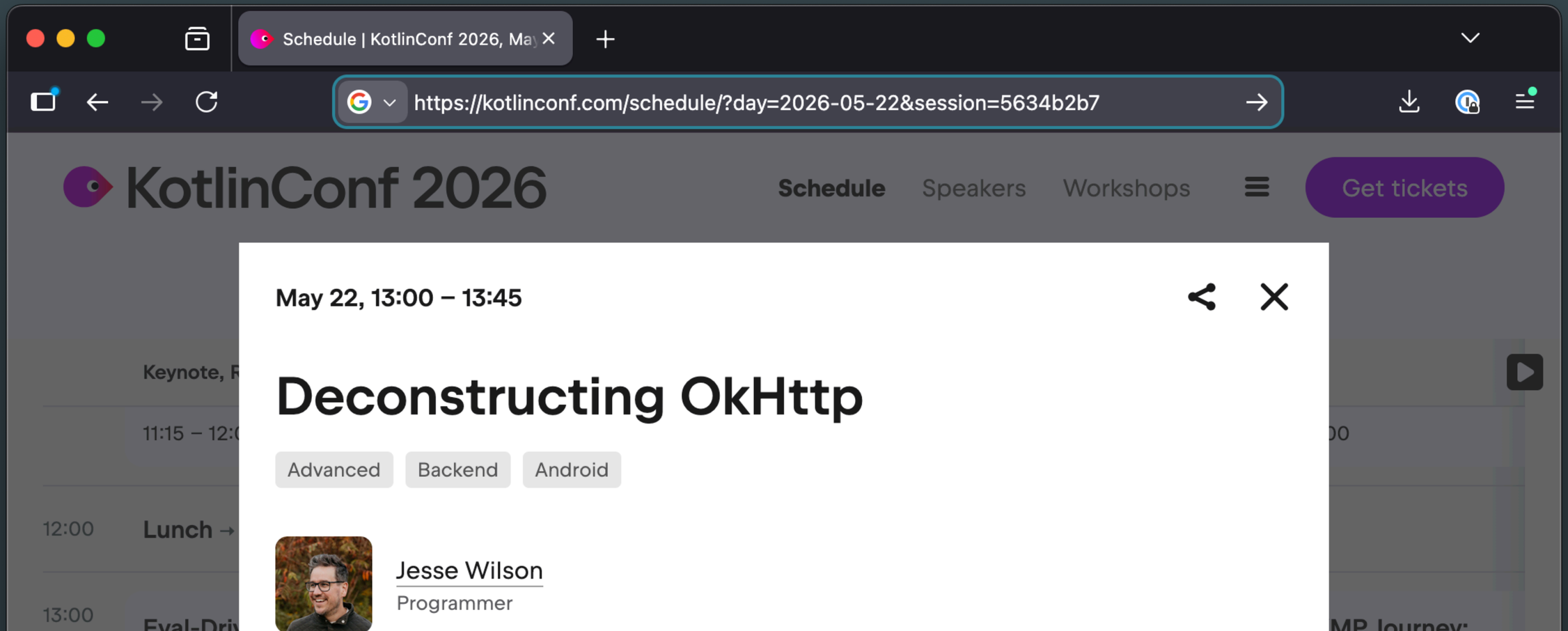
About 150 lines of code!

ITEM 7

HttpUrl

Motivation

https://kotlinconf.com/schedule/?day=2026-05-22&session=5634b2b7



java.net.URL

```
fun buildScheduleUrl(queryParameters: Map<String, String?>): URL {
  val file = buildString {
    append("/schedule/")
    var first = true
    for ((name, value) in queryParameters) {
      if (first) {
        append('?')
        first = false
      } else {
        append('&')
      }
      append(URLEncoder.encode(name, "UTF-8").replace("+", "%20"))
      if (value != null) {
        append('=')
        append(URLEncoder.encode(value, "UTF-8").replace("+", "%20"))
      }
    }
  }

  @Suppress("DEPRECATION") // URI cannot escape query parameters independently.
  return URL("https", "kotlinconf.com", -1, file, null)
}
```

java.net.URL

```
fun buildScheduleUrl(queryParameters: Map<String, String?>): URL {
  val file = buildString {
    append("/schedule/")
    var first = true
    for ((name, value) in queryParameters) {
      if (first) {
        append('?')
        first = false
      } else {
        append('&')
      }
      append(URLEncoder.encode(name, "UTF-8").replace("+", "%20"))
      if (value != null) {
        append('=')
        append(URLEncoder.encode(value, "UTF-8").replace("+", "%20"))
      }
    }
  }

  @Suppress("DEPRECATION") // URI cannot escape query parameters independently.
  return URL("https", "kotlinlang.com", -1, file, null)
}
```

YUCK

YUCK

YUCK

YUCK

YUCK

HttpUrl

```
fun buildScheduleHttpUrl(queryParameters: Map<String, String?>): HttpUrl {
  return HttpUrl.Builder()
    .apply {
      scheme("https")
      host("kotlinconf.com")
      encodedPath("/schedule/")
      for ((name, value) in queryParameters) {
        addQueryParameter(name, value)
      }
    }
    .build()
}
```

What Does It Do?

Encode & Decode URLs

Resolve links

Decode IP addresses

Internationalized Domain Names (IDNs) & Punycode

Top Private Domains

Input	Canonical
http://[0000:0000:0000:0000:0000:0000:0000:0001]/	http://[::1]/
http://[::ffff:c0a8:1fe]/	http://192.168.1.254/
http://%5B%3A%3A1%5D/	http://[::1]/
http://a:b@c:d@e/	http://a:b%40c%3Ad@e/
http://Σ	http://xn--4xa/
http://café.com/	http://xn--caf-dma.com/
http://waSMo.com/	http://wasmo.com/

How Does It Work?

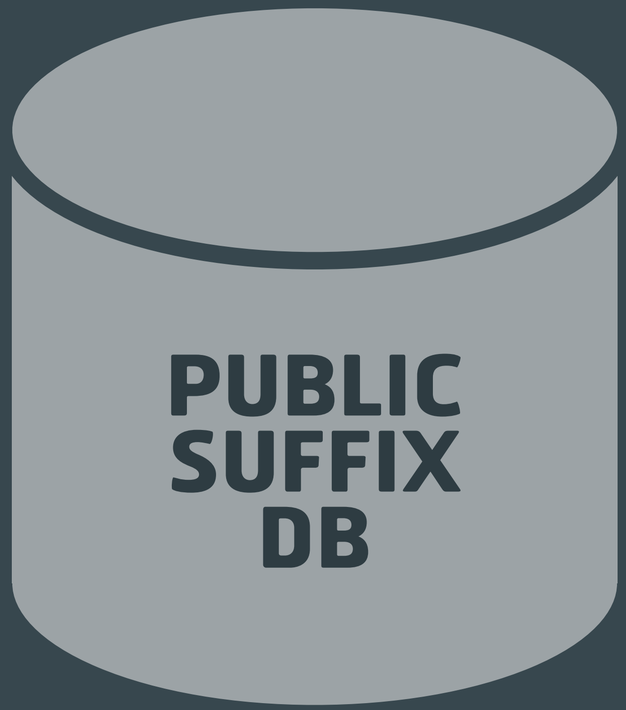
Lots of parsers & formatters

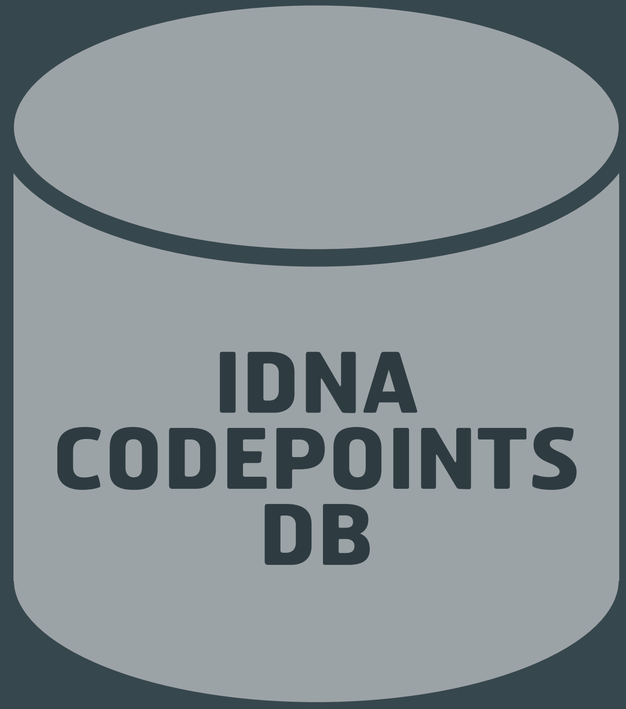
Lots of our own test cases

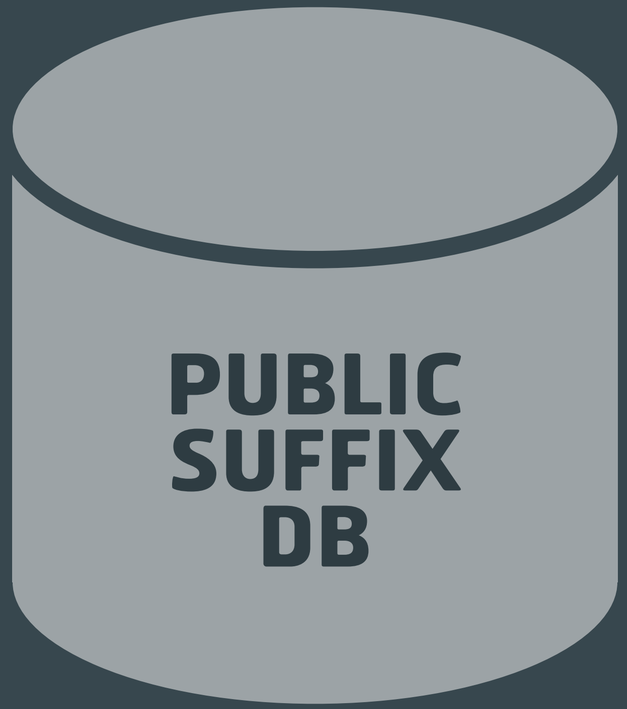
Third-party test suites

Two embedded databases

Embedded Databases?!






PUBLIC
SUFFIX
DB

Public Suffix DB

Which URLs share cookies with which other URLs?

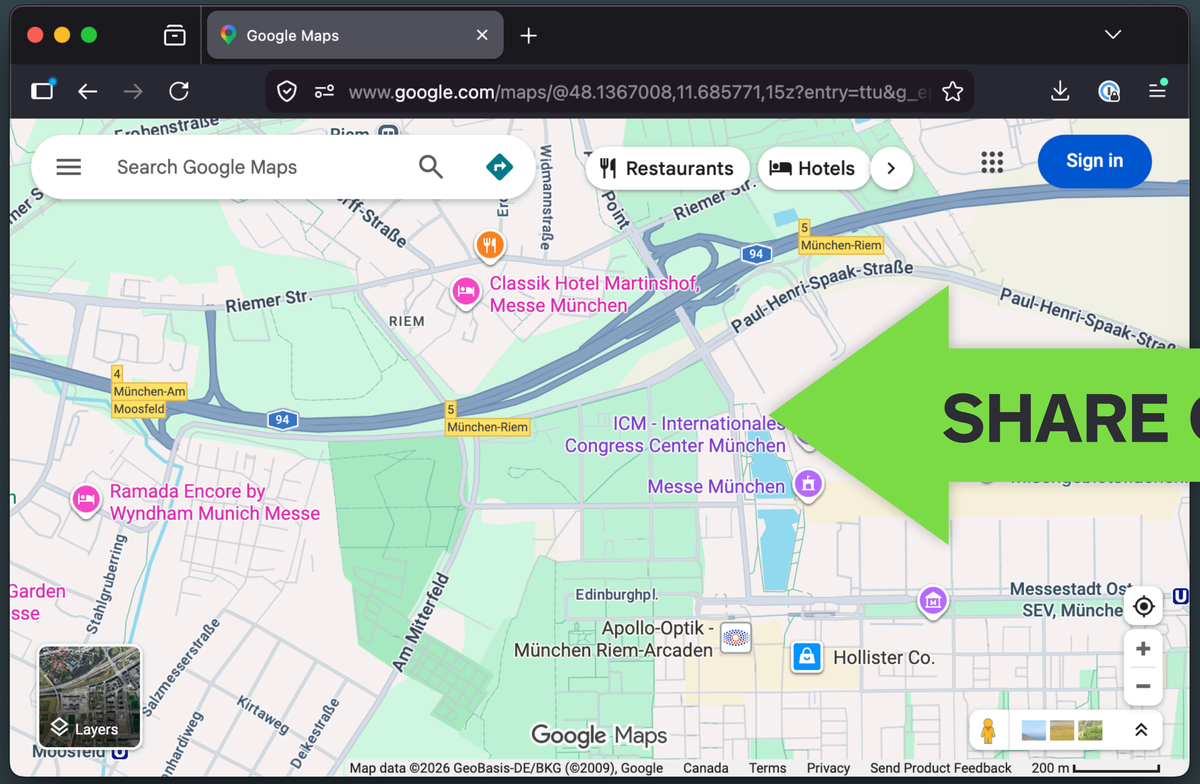
Enables single sign on

But the websites need mutual trust

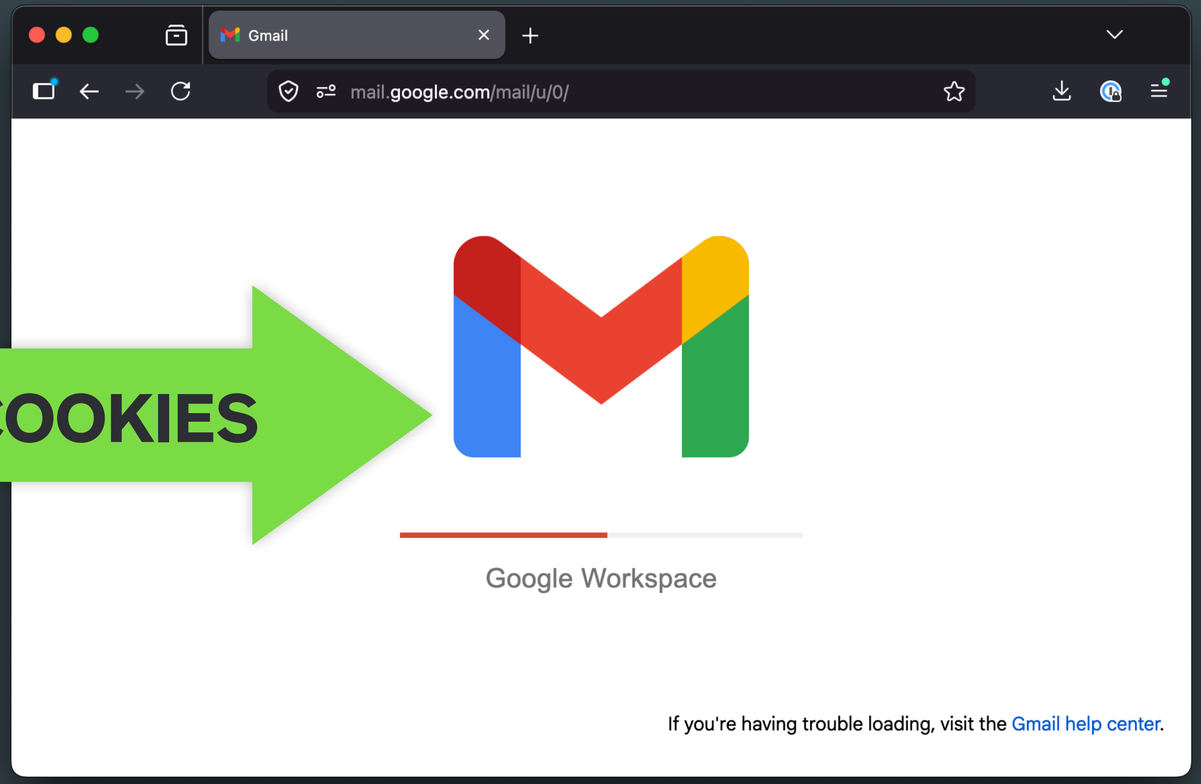
Single Sign On to Google

maps.google.com

mail.google.com



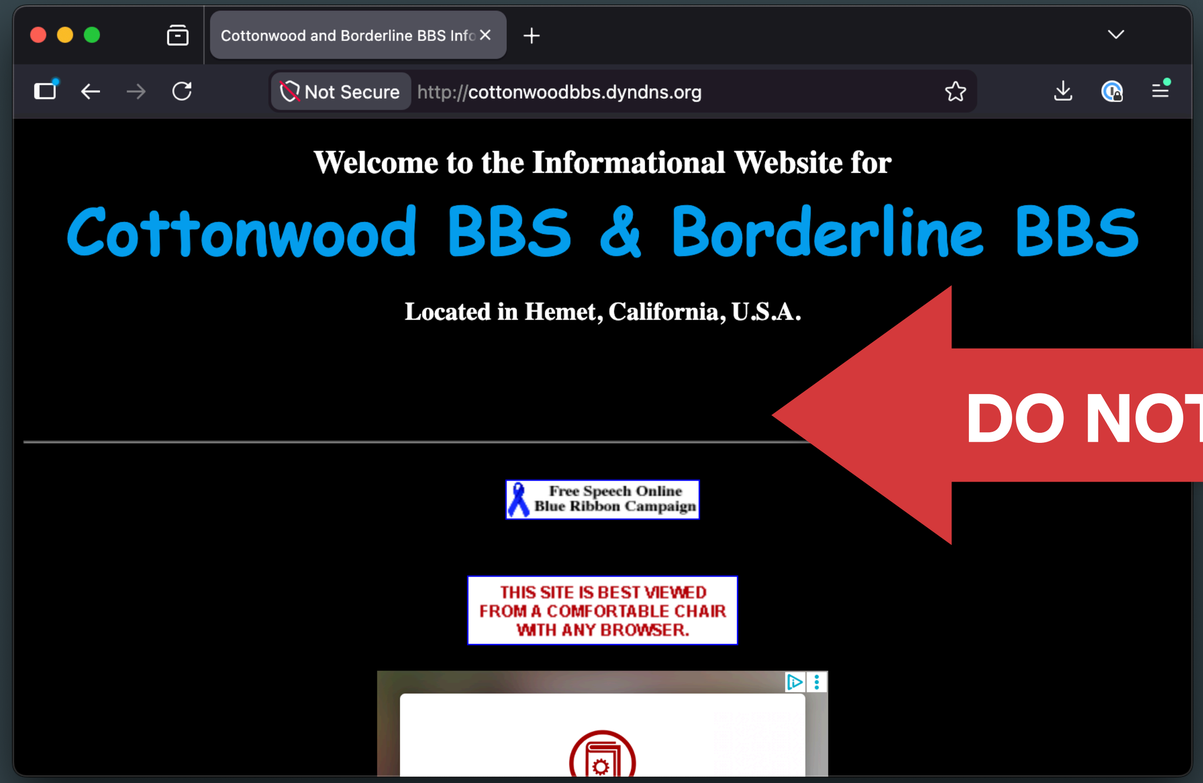
SHARE COOKIES



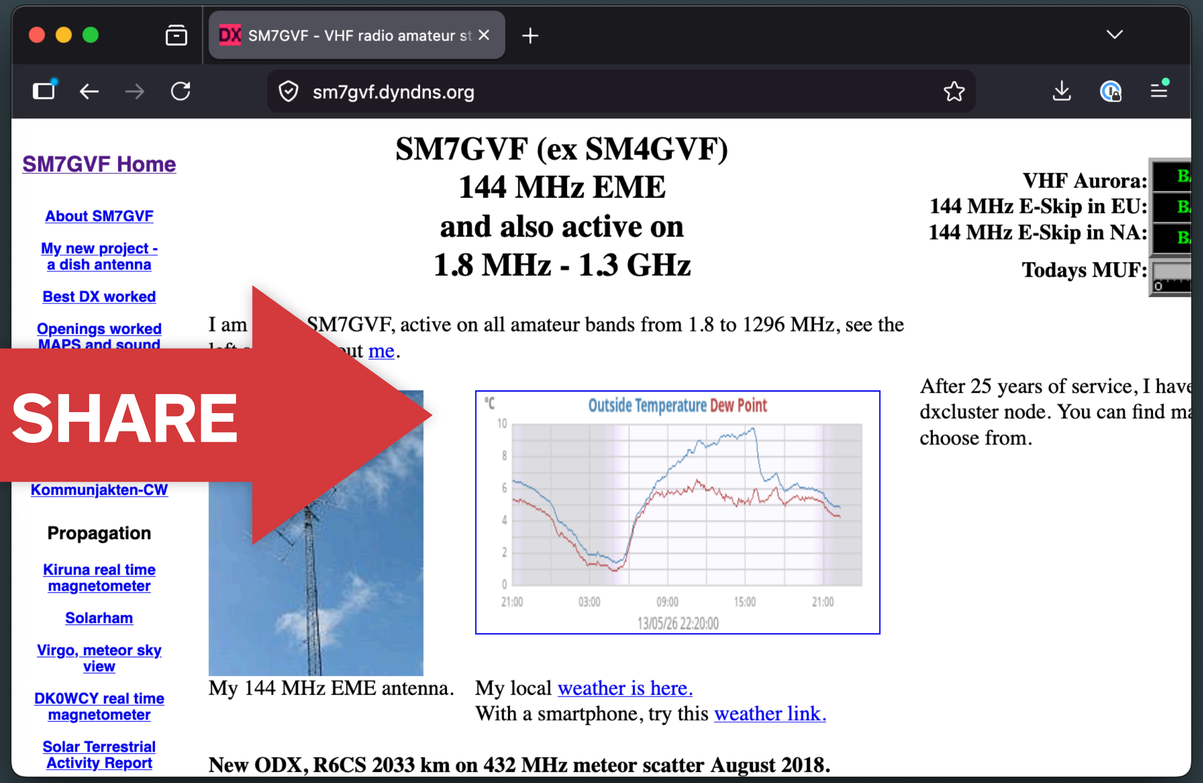
But no sharing on dyndns.org

cottonwoodbbs.dyndns.org

sm7gvf.dyndns.org

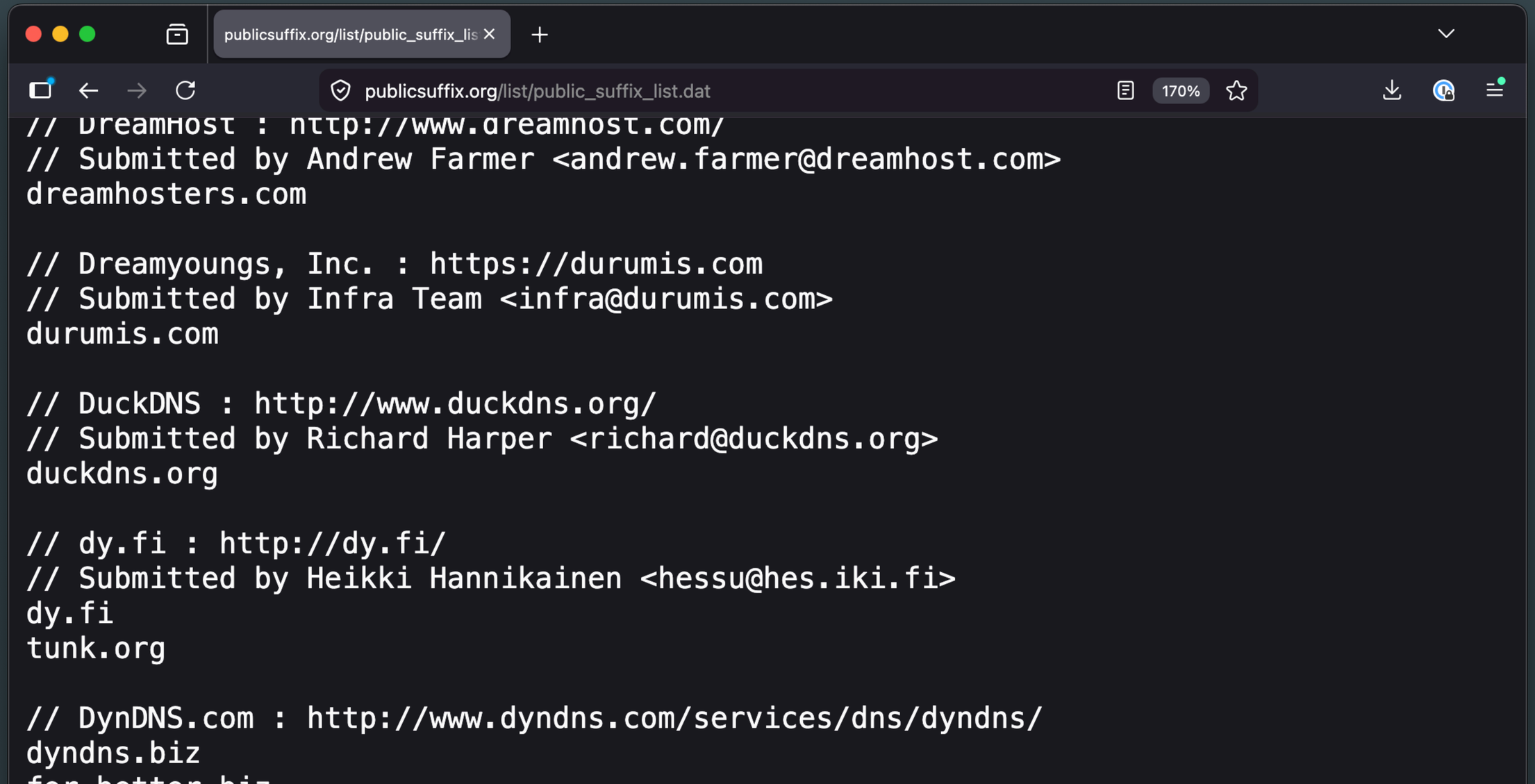


DO NOT SHARE

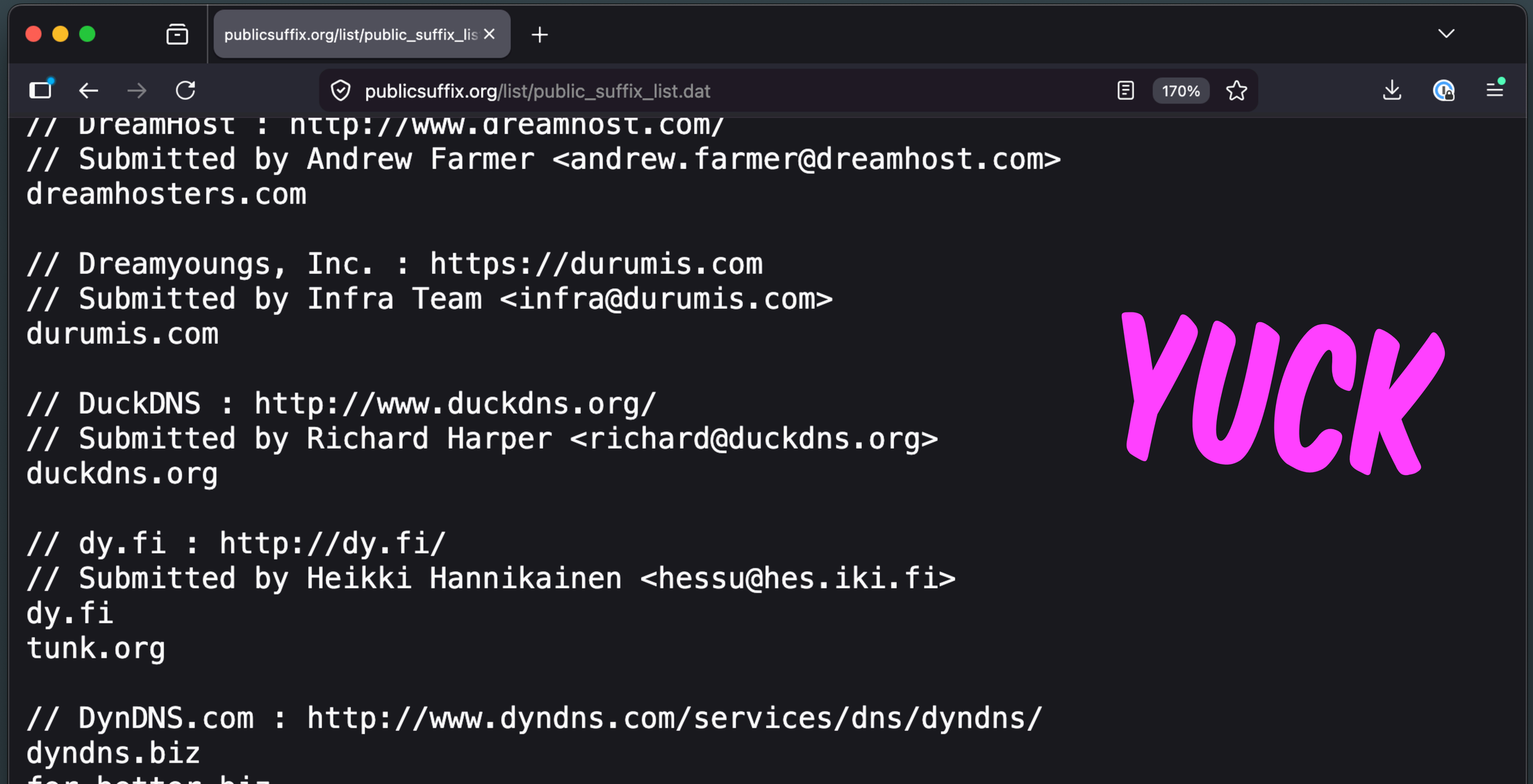


```
// Prints 'session=abcd1234; domain=google.com; path=/'
println(
  Cookie.parse(
    url = "https://login.google.com/".toHttpUrl(),
    setCookie = "session=abcd1234; domain=google.com",
  ),
)

// Prints 'null'
println(
  Cookie.parse(
    url = "https://login.dyndns.org/".toHttpUrl(),
    setCookie = "session=abcd1234; domain=dyndns.org",
  ),
)
```

It’s Just a Giant List of Domains



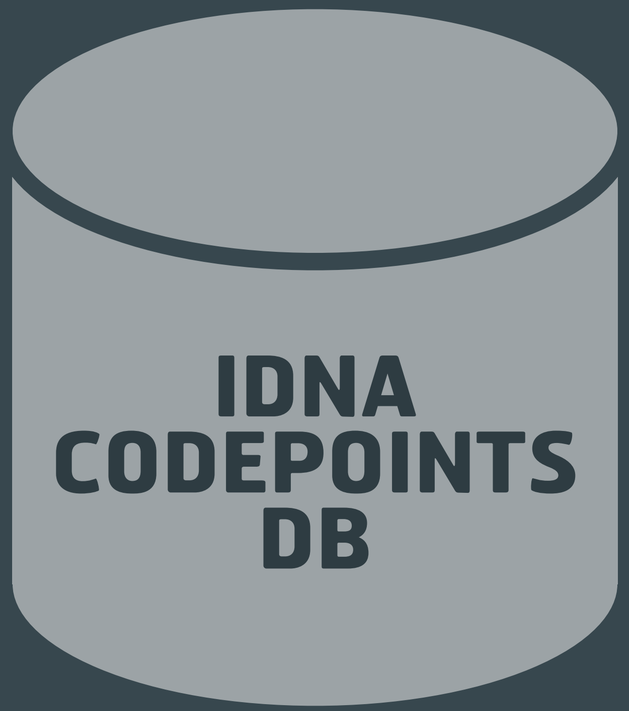
It’s Just a Giant List of Domains



The entire database is in memory as a `ByteArray(132_737)`

Entries are sorted and newline-separated

Lookup by a custom binary search


IDNA
CODEPOINTS
DB

IDNA Codepoints DB

How to process each character in a hostname?

1.1 million codepoints!

Allowed, Disallowed, Ignored, or Mapped

Allowed a â α ㅊ 

Mapped

A → a
Â → â
A → α
㉾ → ㅊ
℠ → sm
™ → tm

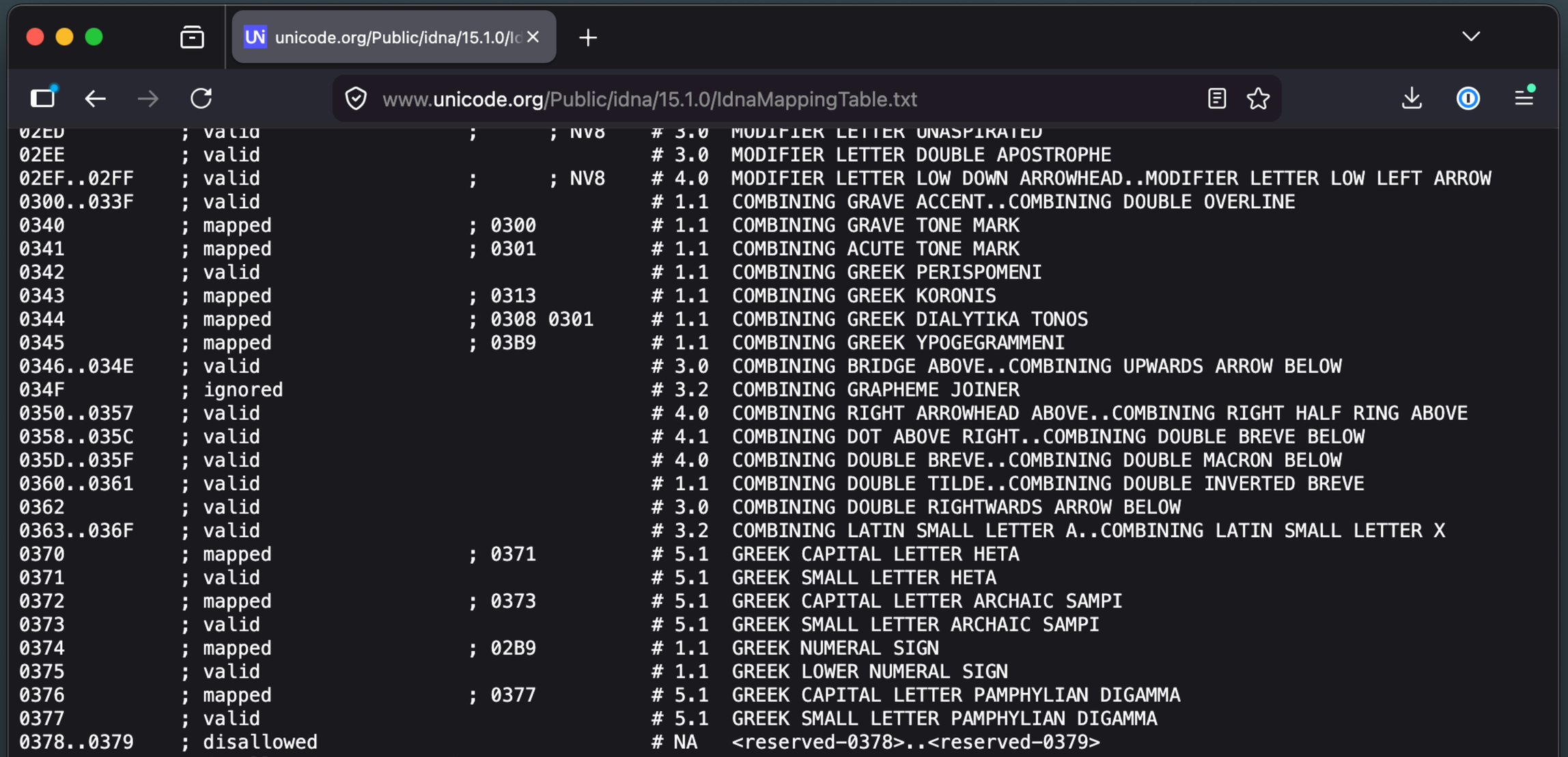
Disallowed Ↄ a.m. ㅢ

Ignored

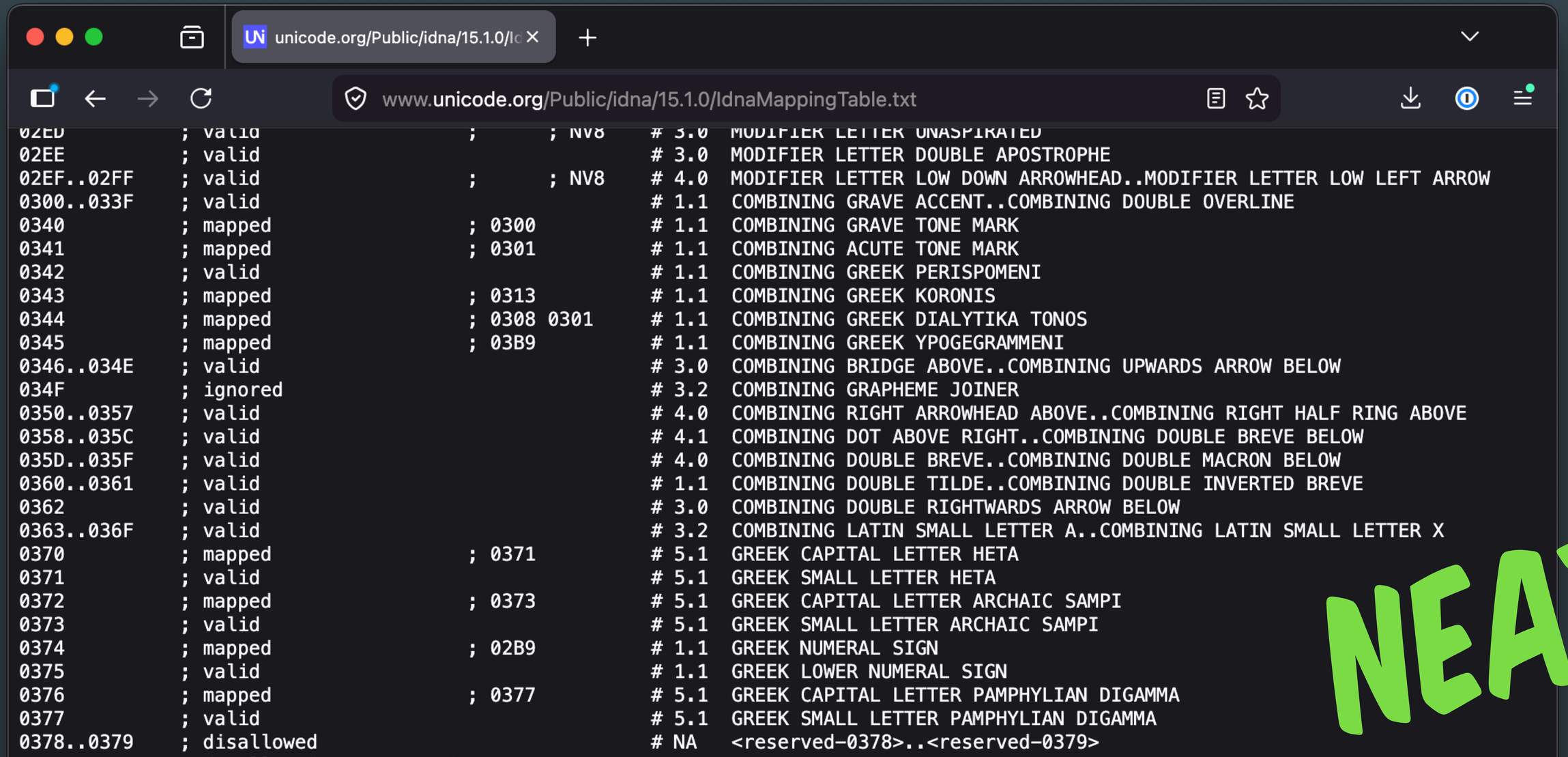
U+200B
zero width space

U+00AD
soft hyphen

It’s Just a Giant List of Code Points



It’s Just a Giant List of Code Points



Solution 1

```
sealed interface Mapping {
  data object Allowed : Mapping
  data class Mapped(val value: ByteString) : Mapping
  data object Disallowed : Mapping
  data object Ignored : Mapping
}

// 1.1 million entries
val codePointToMapping: Map<Int, Mapping> = ...
```

Solution 1

```
sealed interface Mapping {
  data object Allowed : Mapping
  data class Mapped(val value: ByteString) : Mapping
  data object Disallowed : Mapping
  data object Ignored : Mapping
}

// 1.1 million entries
val codePointToMapping: Map<Int, Mapping> = ...
```

Can we do better?

IDNA Mapping Table Goals

Memory compact

Fast to initialize

Fast to query

Compatible with Kotlin/JS, Kotlin/Native, and Kotlin/Wasm

Good News

The spec is in a 875 KiB text file

There’s only about 9,000 lines, because each line expresses a range:

`'a'..'z'` is all allowed
`'\u0080'..'\u009F'` is all disallowed

And these are somewhat repetitive. Mapping A to Z takes 26 lines:

`'A'` maps to `'a'`
`'B'` maps to `'b'`
…

Solution 2

Express 1.1 million code points as 8,153 ranges with these types:

- Ignored
- Allowed
- Disallowed
- Mapped with an delta offset (ie. ‘+ 32’ to map `'A'` to `'a'`)
- Mapped to an inline value (1 or 2 bytes)
- Mapped to a offset in an overflow string

Pack each item into 6 bytes, plus the overflow string (4,719 bytes)

Total size: 54 KiB

OFFSET	MAPPING	MAPPING DATA 1	MAPPING DATA 2
0x0000	allowed		
0x0041	mapped delta	32	
0x005B	allowed		
0080	disallowed		
0x00A0	mapped inline	0x0020	
...			
0x1F14E	mapped external	4204	3
0x1F14F	mapped external	4207	3

Solution 3

The first 3 bytes of each entry are pretty darn repetitive

- Group entries by the first 2 of those bytes
- Build a table of contents for each group

Total size: 39 KiB (28% savings)

Bad News

Shipping a resource file in Kotlin/Multiplatform is difficult

(Compose Multiplatform Resources isn’t everywhere yet)

Solution 4

We’ve been using bytes throughout

If we use only 7 bits of each byte, this data can be a string

```
internal val IDNA_MAPPING_TABLE = IdnaMappingTable(
    sections = "\u0000\u0000\u0000\u0000\u0000\u0001\u0000\u
    ranges = "\u0000x--AP\u0000\u0020[x--\u0000y--\u0020@\u
    mappings = "\u002Ö\u002Ō\u002Ó\u002Ǫ1/41/23/4i̇l\u00b7ʼn
)
```

IDNA Mapping Table

This ‘database’ compiles into a 33 KiB .class file

Plus 8 KiB of code to use it

(Including another custom binary search)

ITEM 8

Cache

Two Jobs

Policy

Which responses to save?

Does a saved response satisfy a request?

Should we check our cached response with the server with `If-None-Match` ? (etags)

Persistence

What happens if the process crashes while we're saving a response?

We don't want to serve that response later

We also don't want to leak storage space

Persistence Strategy

2 files for every saved response:

Metadata a text file with the URL, select request headers, response headers, and server certificates

Body the original response body, possibly gzipped

A journal with file state (dirty, clean, removed), plus access events

```
val cache = Cache(
  fileSystem = FileSystem.SYSTEM,
  directory = "cache".toPath(),
  maxSize = 1024 * 1024 * 10,
)

val client = OkHttpClient.Builder()
  .cache(cache)
  .eventListener(eventListener)
  .build()

val call1 = client.newCall(Request("https://kotlinconf.com/".toHttpUrl()))
println(call1.execute().body.string())

val call2 = client.newCall(Request("https://wasmo.com/".toHttpUrl()))
println(call2.execute().body.string())

val call3 = client.newCall(Request("https://kotlinconf.com/".toHttpUrl()))
println(call3.execute().body.string())
```

Journal

```
libcore.io.DiskLruCache
1
201105
2

DIRTY f712e13336b8831e979afd57f6050144
CLEAN f712e13336b8831e979afd57f6050144 6212 117149
DIRTY d69aa6582489f430cbefa9d1c9da75da
CLEAN d69aa6582489f430cbefa9d1c9da75da 5168 988
READ f712e13336b8831e979afd57f6050144
DIRTY f712e13336b8831e979afd57f6050144
CLEAN f712e13336b8831e979afd57f6050144 6212 117149
```

Metadata

```
https://wasmo.com/
GET
0
HTTP/1.1 200
5
content-type: text/html; charset=UTF-8
date: Thu, 14 May 2026 14:14:38 GMT
content-encoding: gzip
OkHttp-Sent-Millis: 1778768078732
OkHttp-Received-Millis: 1778768078800

TLS_AES_128_GCM_SHA256
2
MIIEVjCCAj6gAwIBAgIQY5WTY8JOcIJxWRi/w9ftVjANBgkqhkiG9w0BAQsFADBPMQswCQYDVQQGEwJVUzEpMC
MIIFazCCA1OgAwIBAgIRAIIQz7DSQONZRGPgu2OCiwAwDQYJKoZIhvcNAQELBQAwTzELMAkGA1UEBhMCVVMxKT
0
TLSv1.3
```

Body

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Wasmo</title>
    <meta name="viewport"
          content="width=device-width, initial-scale=1, minimum-scale=1, maximum-scale
    <meta name="theme-color" content="#ffffff">
    <meta property="og:image" content="https://wasmo.com/assets/og-image.png">
    <meta property="og:image:width" content="1200">
    <meta property="og:image:height" content="630">
    <link href="https://fonts.gstatic.com" rel="preconnect" crossorigin="anonymous">
    <link href="https://fonts.googleapis.com/css2?family=Outfit:wght@100..900&amp;disp
          rel="stylesheet">
    <link href="/favicon.ico" rel="icon" sizes="32x32">
    <link href="/icon.svg" rel="icon" type="image/svg+xml">
    <link href="/apple-touch-icon.png" rel="apple-touch-icon">
    <link href="/assets/Wasmo.css" rel="stylesheet">
```

So So Many Failure Modes

Process crash

Response body truncated

Disk fills up

Delete fails!

Self-Crashing FileSystem

Okio has an in-memory file system for testing, `FakeFileSystem`.
(This lets us test Windows behaviour without using Windows!)

To test our cache's crashes we created `FaultyFileSystem`

```
class FaultyFileSystem(
  delegate: FileSystem,
) : ForwardingFileSystem(delegate) {
  private val immortalPaths = mutableSetOf<Path>()

  fun failDelete(path: Path) {
    immortalPaths += path
  }

  @Throws(IOException::class)
  override fun delete(path: Path, mustExist: Boolean) {
    if (path in immortalPaths) throw IOException("boom!")
    super.delete(path, mustExist)
  }
}
```

Playing Hurt

If the cache can’t write new files, performance degrades:

- Network calls still proceed
- But the cache always misses

Windows Is

Windows refuses to delete files that are open for read

Here's a torture test:

1. Start reading `https://ziglang.org/` from the cache
2. Evict that URL from the cache
3. Crash the process

Windows Is YUCK

Windows refuses to delete files that are open for read

Here's a torture test:

1. Start reading `https://ziglang.org/` from the cache
2. Evict that URL from the cache
3. Crash the process

Advice

Don’t do your own journaling, just use a database

`AtomicFile` is fine

If you are using files, consider Okio’s `FileSystem` so you can test

ITEM 9

TLS & Generating Certificates

Java's TLS APIs are Bad

```
public SslClient createSslContextAndTrustManager(
    long duration,
    String hostname,
    List<String> altNames,
    String serialNumber
 ) throws GeneralSecurityException, IOException {
    Security.addProvider(new BouncyCastleProvider());

    // Subject, public & private keys for this certificate.
    KeyPairGenerator keyPairGenerator = KeyPairGenerator.getInstance("RSA", "BC");
    keyPairGenerator.initialize(1024, new SecureRandom());
    KeyPair heldKeyPair = keyPairGenerator.generateKeyPair();

    X500Principal subject = new X500Principal("CN=" + hostname);

    // Generate & sign the certificate.
    long now = System.currentTimeMillis();
    X509V3CertificateGenerator generator = new X509V3CertificateGenerator();
    generator.setSerialNumber(new BigInteger(serialNumber));
    generator.setIssuerDN(subject);
    generator.setNotBefore(new Date(now));
    generator.setNotAfter(new Date(now + duration));
    generator.setSubjectDN(subject);
    generator.setPublicKey(heldKeyPair.getPublic());
    generator.setSignatureAlgorithm("SHA256WithRSAEncryption");

    ASN1Encodable[] encodableAltNames = new ASN1Encodable[altNames.size()];
    for (int i = 0, size = altNames.size(); i < size; i++) {
```

Crypto Jargon	ASN1, CN, DER, DN, RSA, SSL, TLS, X500, X509V3
Magic Strings	`"BC"`, `"CN"`, `"RSA"`, `"SHA256WithRSAEncryption"`, `"TLS"`
Too Abstract	No `generateRsaKeyPair()`
Side-Effects	Why isn't `init()` part of the constructor?
Dangerous	`keyPairGenerator.initialize(1024)`
Gross Syntax	`GeneralName.iPAddress,` `new X500Principal("CN=" + hostname)`
Incomplete	Need BouncyCastle or OkHttp to create a certificate?

“while everybody talked about the weather, nobody seemed to do anything about it”

– Charles Dudley Warner

OkHttp History

2018 OkHttp 3.11 adds `okhttp-tls`, a new module for testing HTTPS clients and servers

This module depends on Bouncy Castle.

2020 OkHttp 4.9 implements its own certificate encoder, so we no longer depend on Bouncy Castle.

Certificates With OkHttp

```
val heldCertificate = HeldCertificate.Builder()
  .commonName("Wasmo")
  .addSubjectAlternativeName("wasmo.com")
  .build()

val handshakeCertificates = HandshakeCertificates.Builder()
  .addTrustedCertificate(heldCertificate.certificate)
  .build()

val client = OkHttpClient.Builder()
  .sslSocketFactory(
    handshakeCertificates.sslSocketFactory(),
    handshakeCertificates.trustManager
  )
  .build()
```

PEM and DER

```
-----BEGIN CERTIFICATE-----
MIIBKDCB0KADAgECAgEBMAoGCCqGSM49BAMCMBAxDjAMBgNVBAMMBVdhc21vMB4X
DTI2MDUxNjAzMzEzM1oXDTI2MDUxNzAzMzEzM1owEDEOMAwGA1UEAwwFV2FzbW8w
WTATBgcqhkjOPQIBBggqhkjOPQMBBwNCAASCkvH6fAAh0C21XzdGEi9utL/zQRPf
MAZUz4RXLswb9YjSTawf9cxMHg6wb5IKvRysUiqmD1RVcPPUeeke6K7XoxswGTAX
BgNVHREBAf8EDTALggl3YXNtby5jb20wCgYIKoZIzj0EAwIDRwAwRAIgVAq2+fu9
uMDFn7zp/6KzvcwdNpwZrXULL9Cv1EbDVsECIHhuI0wkQGrPzQEKQcQ3nj5IxekT
hYZbC8Zzndh8n8Vz
-----END CERTIFICATE-----
```

Advice

Certificates are scary?

Actually, it’s just their APIs!

The protocols aren’t so bad, and the math is neat

If it’s too hard to test HTTPS, people will not

ITEM 10

Happy Eyeballs

DNS

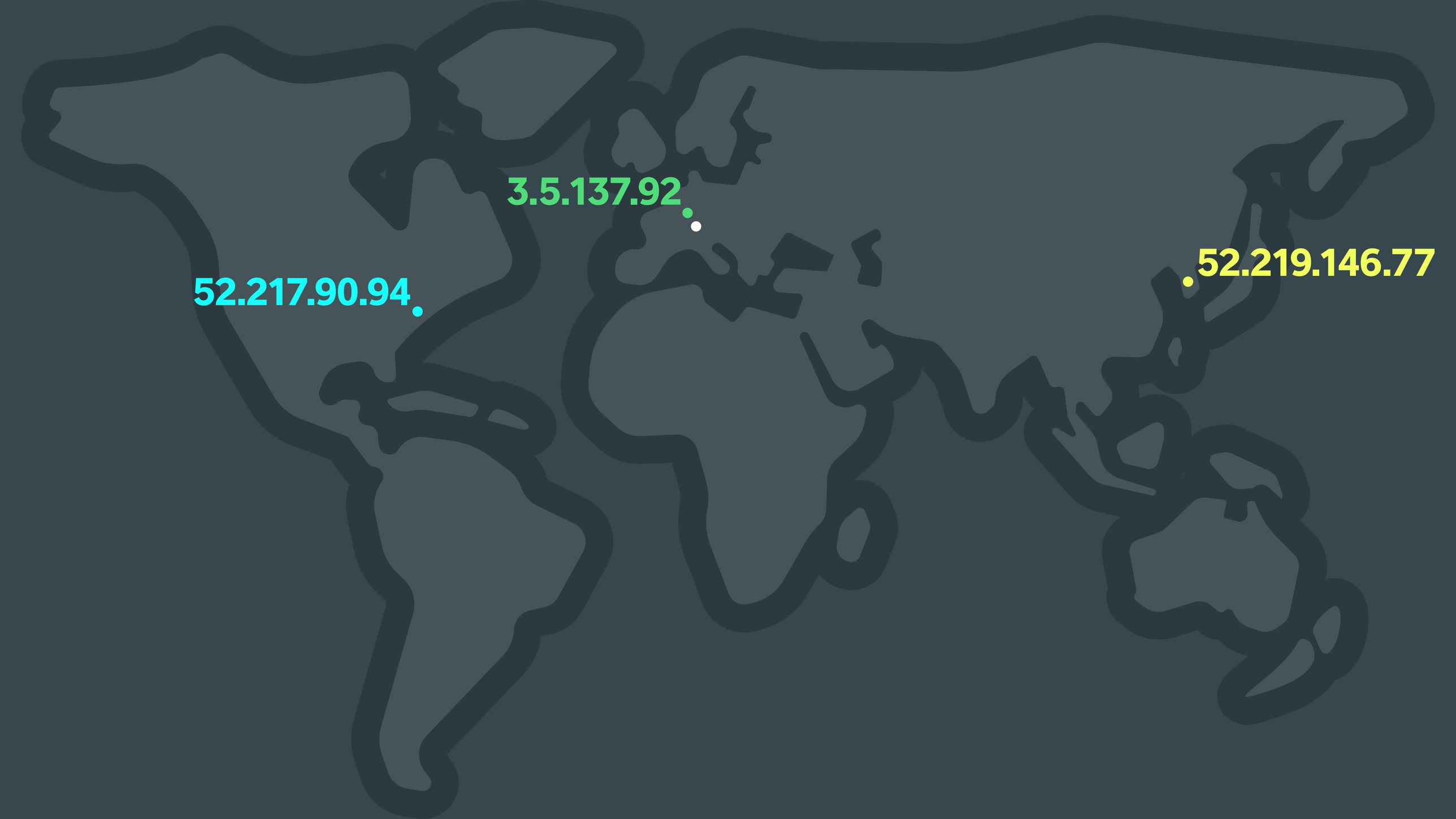
Turns a hostname like `cdn.kotlinconf.com` into an IP address

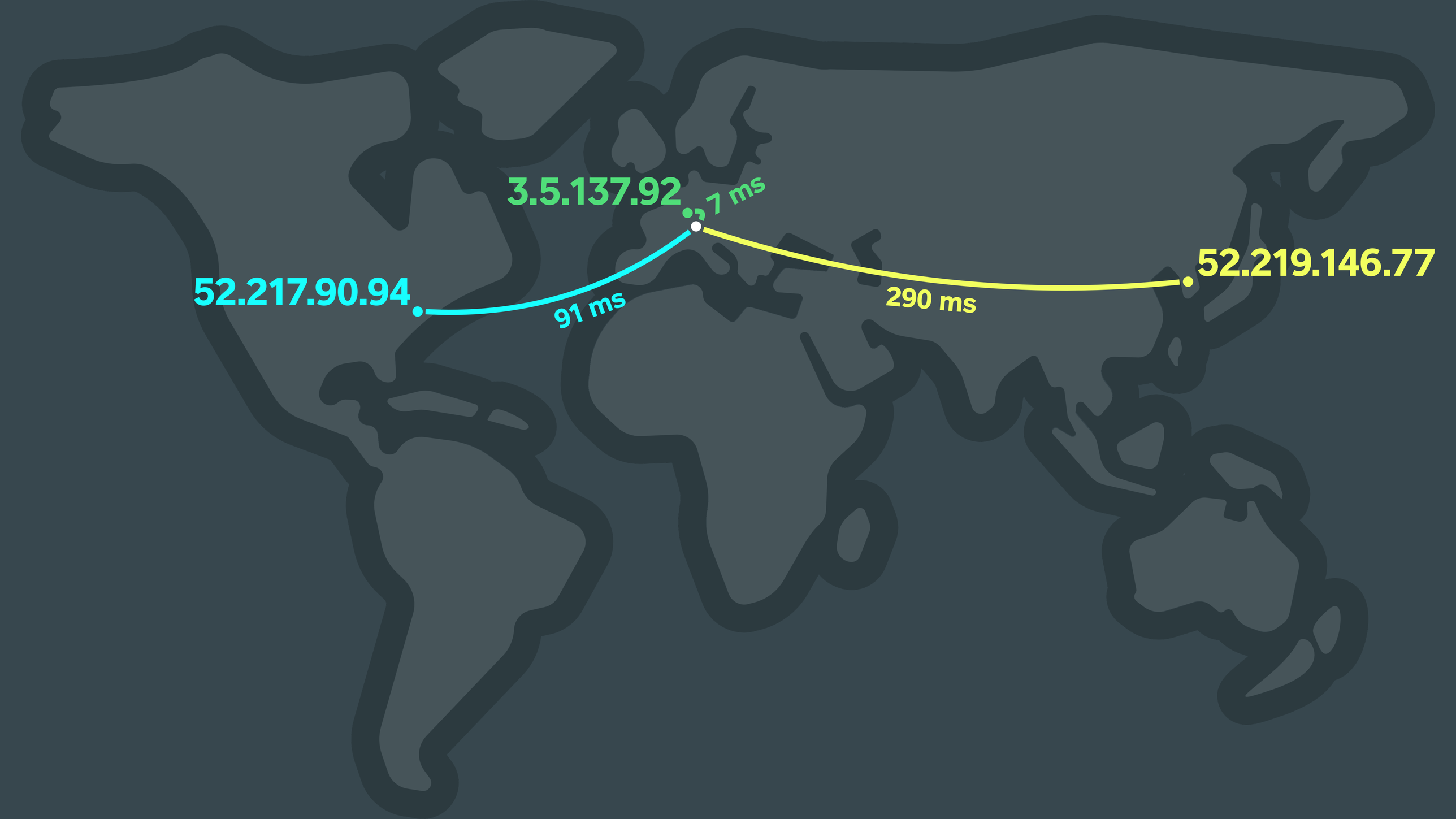
A single DNS name may resolve to multiple IP addresses:

- IPv4 and IPv6
- Multiple geographic locations
- Redundancy

```
for (address in Dns.SYSTEM.lookup("cdn.kotlinconf.com")) {
  println(address)
}
```

```
kotlinconf.com/52.219.146.77
kotlinconf.com/3.5.137.92
kotlinconf.com/52.217.90.94
```


3.5.137.92
52.217.90.94
52.219.146.77


3.5.137.92
7 ms
52.217.90.94
91 ms
290 ms
52.219.146.77

What IP to Connect To?

Attempt each in sequence

If connecting to one IP address times out, try the next

What IP to Connect To?

Attempt each in sequence

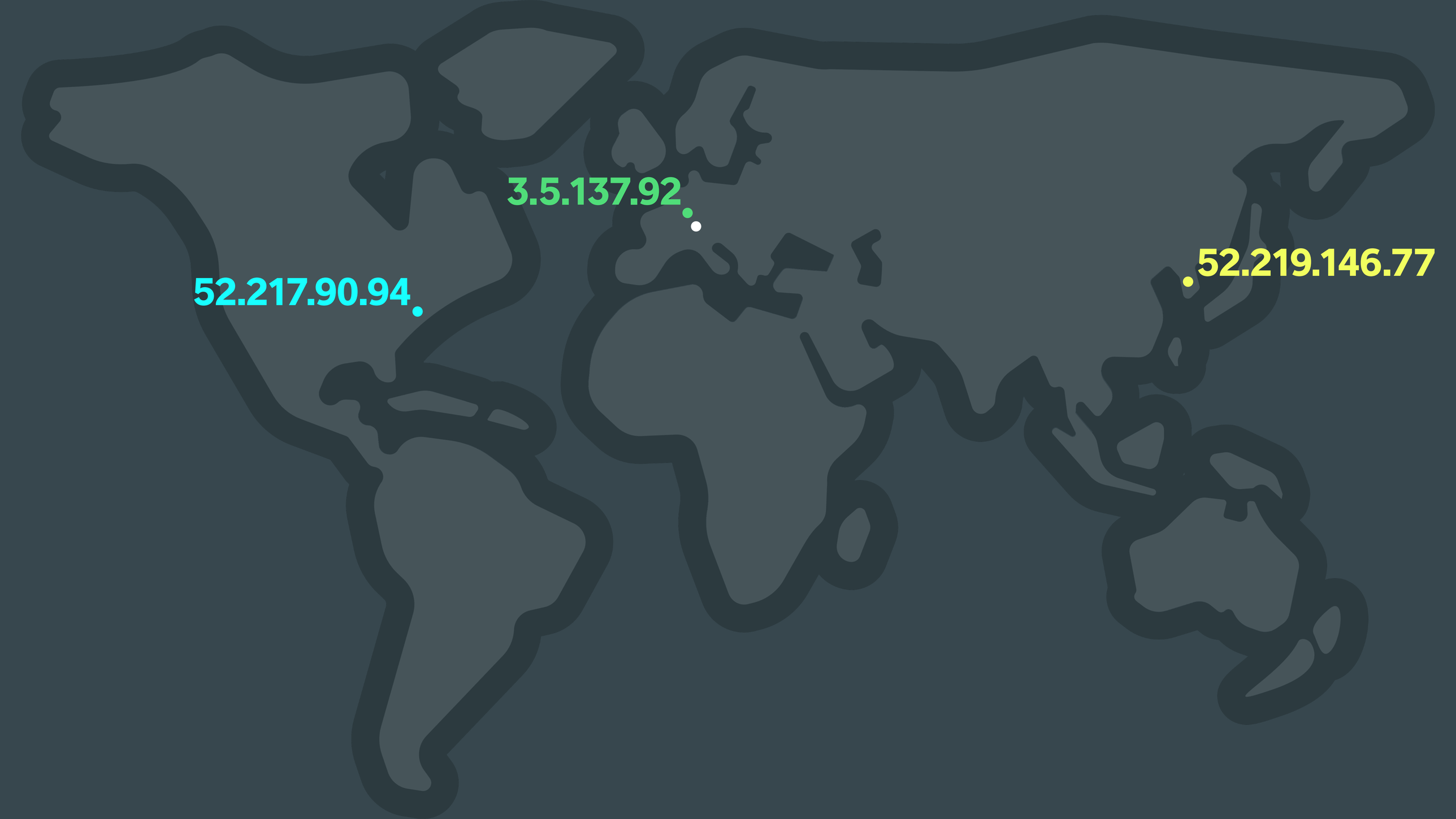
If connecting to one IP address times out, try the next

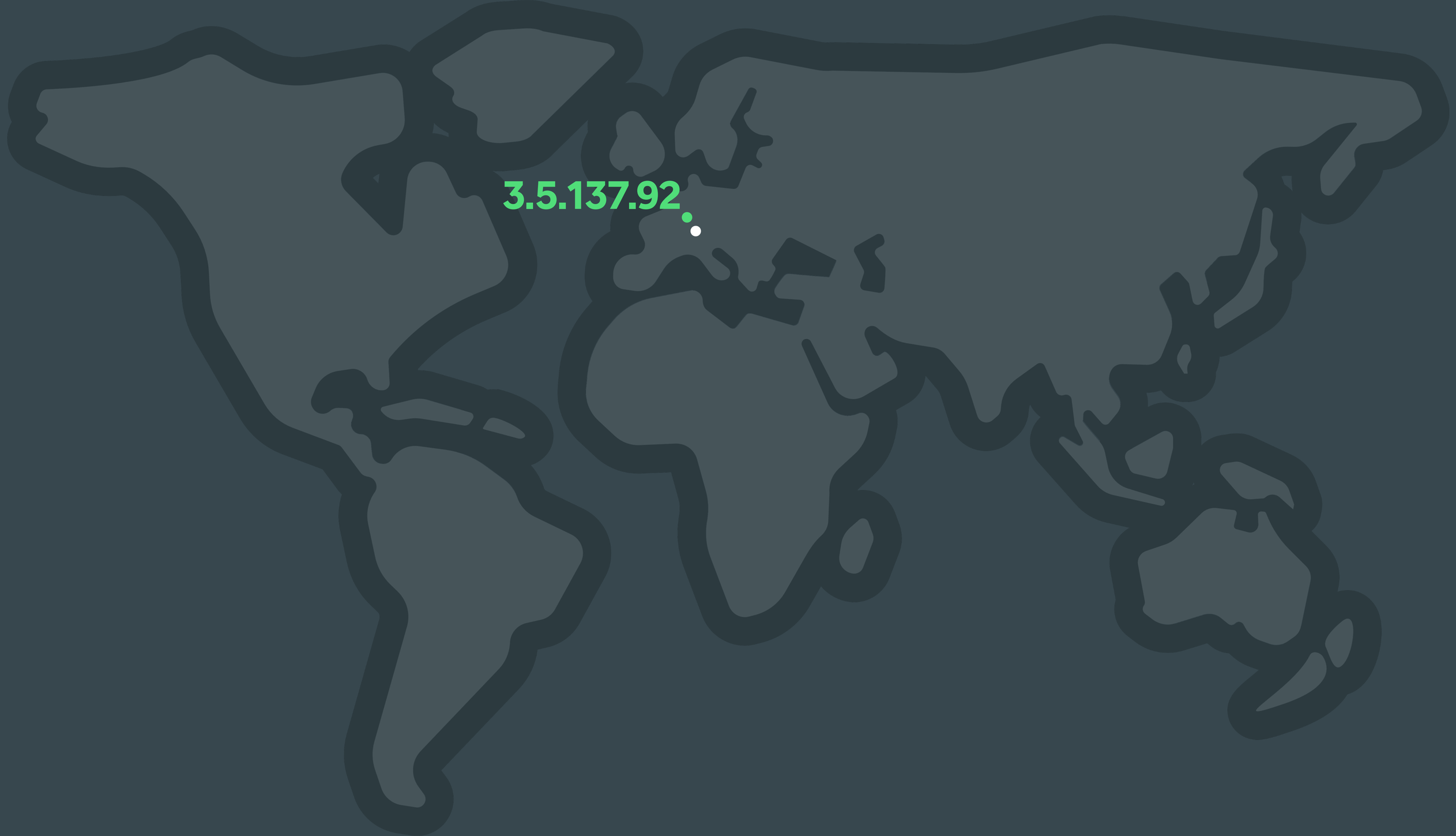
Can we do better?

Happy Eyeballs

Attempt a new connection every 250 ms until one of them succeeds, or they all time out

Why 250 ms? Attempt to balance between latency and resource consumption


3.5.137.92
52.217.90.94
52.219.146.77


3.5.137.92

OkHttp’s Happy Eyeballs

This is a scary feature to roll out

Will we accidentally break the Internet?

- Mobile apps could send 2x or 3x as many SYN packets
- Connecting was single-threaded and now it’s concurrent!

Lots of tests

Next Steps

Every 250 ms is too simple?

Persist race results to inform future connections?

ITEM 11

Upgrade to Kotlin

Java	Kotlin
Builders	Named Parameters
Nullable Everything	Strict Nulls
Declared Exceptions	Hope for the best
java.io	kotlinx.io
Callbacks	Coroutines
SCREAMING_CASE	UpperCamel
Name Every Type	Lambdas

OkHttp 4

```
val client = OkHttpClient()
val call = client.newCall(
  Request.Builder()
    .url("https://www.kotlinconf.com/".toHttpUrl())
    .build(),
)
val response = call.execute()

assertThat(response.body!!.string()).contains("Munich")
```

OkHttp 4

```
val client = OkHttpClient()
val call = client.newCall(
  Request.Builder()
    .url("https://www.kotlinconf.com/".toHttpUrl())
    .build(),
)
val response = call.execute()

assertThat(response.body!!.string()).contains("Munich")
```

YUCK

Multiple Responses?

A single `Call` might include multiple intermediate responses:

- HTTP 301, a redirect
- HTTP 401, an auth challenge
- HTTP 503, a server-recommended retry

These earlier responses are returned in `Response.priorResponse`.

But OkHttp discards the bodies of these responses

In Java, Null is Fine

Java doesn’t mind if you call `call.response.body.string()`

But Kotlin makes you do the `!!` of shame, because `Response.body` is nullable

A stronger type system made our API worse!

A Pragmatic Mitigation

OkHttp 5 added `UnreadableResponseBody`.

It’s non-null, but it throws if you attempt to read it

OkHttp 5

```
val client = OkHttpClient()
val call = client.newCall(
  Request(
    url = "https://www.kotlinconf.com/".toHttpUrl()
  ),
)
val response = call.execute()

assertThat(response.body.string()).contains("Munich")
```

Next Steps

There’s a bunch of backwards-incompatible changes to consider:

- kotlinx.io
- Coroutines in the core module

We aren’t pursuing these yet!

We’d like to make `HttpUrl` and other value types multiplatform

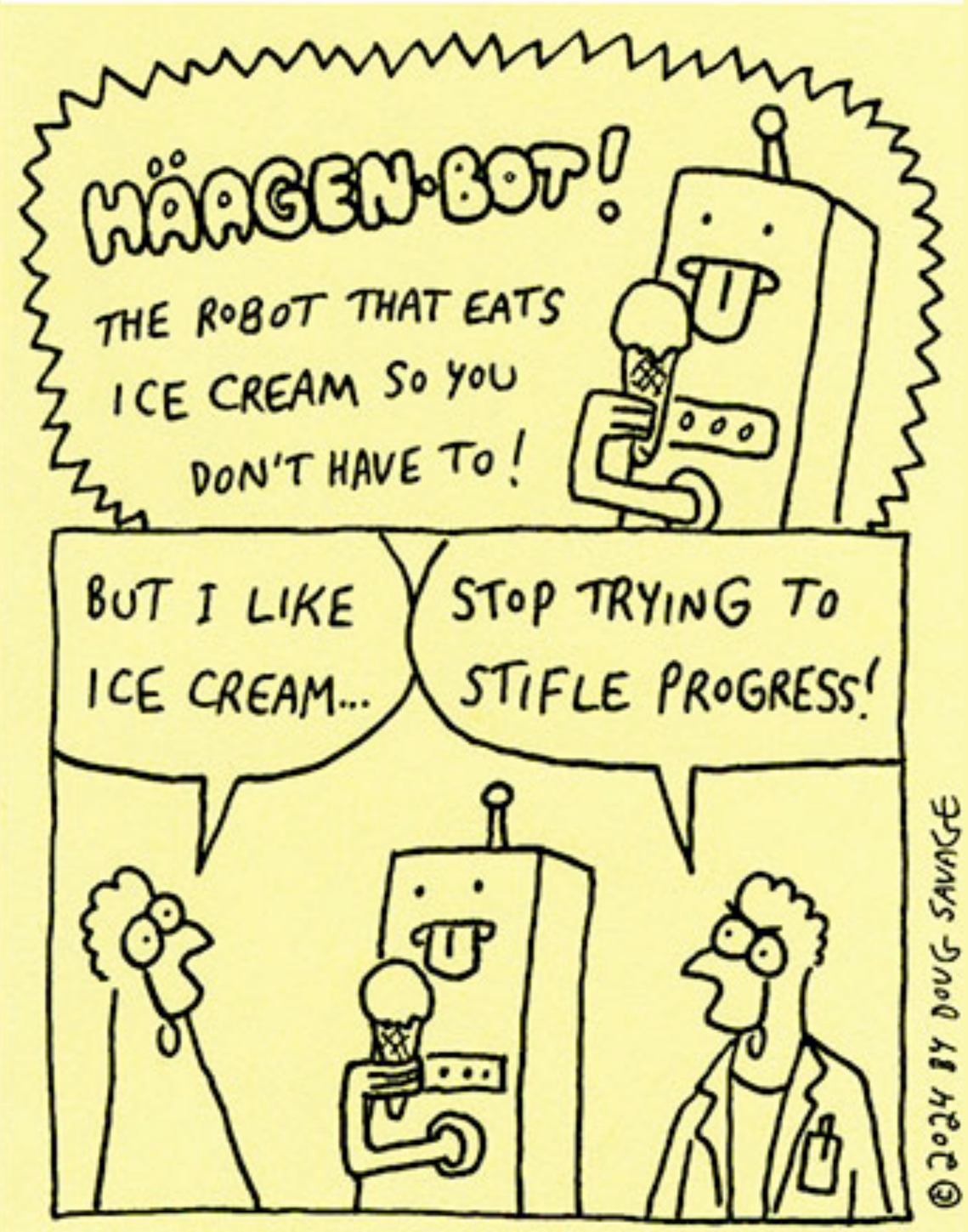
Recap

Programming is Fun

Optimizing code:

- Makes programs faster
- Makes software cheaper to operate
- Saves CO_2

Use tests to solve hard problems



lysine.dev/okhttp/

wasmo

Thanks