

Project Loom:

A Summary For Busy Developers

Michael Easter

@codetojoy

Project Loom

~~Fibers~~ and Continuations



OpenJDK

[Installing](#)
[Contributing](#)
[Sponsoring](#)
[Developers' Guide](#)
[Vulnerabilities](#)
[JDK GA/EA Builds](#)

[Mailing lists](#)
[Wiki](#) · [IRC](#)

[Bylaws](#) · [Census](#)
[Legal](#)

[JEP Process](#)

[Source code](#)
Mercurial
GitHub

[Tools](#)
Mercurial
Git
jtreg harness

[Groups](#)
(overview)

JDK 19

This release will be the Reference Implementation of version 19 of the Java SE Platform.

Status

The main-line code repository is open for bug fixes, small enhancements, and JEPs as proposed and tracked via the JEP Process.

Schedule

2022/06/09	Rampdown Phase One (fork from main line)
2022/07/21	Rampdown Phase Two
2022/08/11	Initial Release Candidate
2022/08/25	Final Release Candidate
2022/09/20	General Availability

OpenJDK

[Installing](#)
[Contributing](#)
[Sponsoring](#)
[Developers' Guide](#)
[Vulnerabilities](#)
[JDK GA/EA Builds](#)

[Mailing lists](#)
[Wiki](#) · [IRC](#)

[Bylaws](#) · [Census](#)
[Legal](#)

[JEP Process](#)

[Source code](#)
Mercurial
GitHub

[Tools](#)
Mercurial
Git
jtreg harness

[Groups](#)
(overview)

JDK 19

This release will be the Reference Implementation of version 19 of the Java SE Platform.

Status

The main-line code repository is open for bug fixes, small enhancements, and JEPs as proposed and tracked via the JEP Process.

Schedule

2022/06/09	Rampdown Phase One (fork from main line)
2022/07/21	Rampdown Phase Two
2022/08/11	Initial Release Candidate
2022/08/25	Final Release Candidate
2022/09/20	General Availability



JEP 425: Virtual Threads (Preview)

OpenJDK

JEP 428: Structured Concurrency (Incubator)

Virtual Threads

Platform threads

```
14 void createManyThreads(int numThreads) throws Exception {  
15     Thread thread = null;  
16  
17     for (int i = 0; i < numThreads; i++) {  
18         thread = new Thread(Runner::doWork);  
19         thread.start();  
20     }  
21  
22     thread.join();  
23 }
```

Platform threads

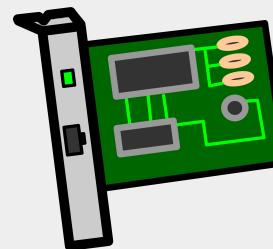
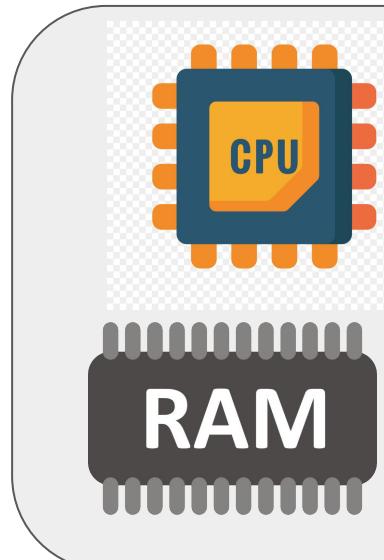
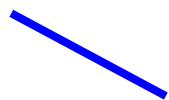
```
TRACER working t: Thread[#4092,Thread-4064,5,main]
TRACER working t: Thread[#4093,Thread-4065,5,main]
TRACER working t: Thread[#4094,Thread-4066,5,main]
Exception in thread "main" java.lang.OutOfMemoryError:
unable to create native thread:
possibly out of memory or process/resource limits reached
at java.base/java.lang.Thread.start0(Native Method)
at java.base/java.lang.Thread.start(Thread.java:1531)
at net.codetojoy.Runner.createManyThreads(Runner.java:19)
```

Virtual threads

```
14 void createManyThreads(int numThreads) throws Exception {  
15     Thread thread = null;  
16  
17     for (int i = 0; i < numThreads; i++) {  
18         thread = Thread.startVirtualThread(Runner::doWork);  
19     }  
20  
21     thread.join();  
22 }
```

Virtual threads

```
TRACER working t: VirtualThread[#400029]/Runnable@ForkJoinPool-1-worker-1
TRACER working t: VirtualThread[#400031]/Runnable@ForkJoinPool-1-worker-6
TRACER working t: VirtualThread[#400032]/Runnable@ForkJoinPool-1-worker-2
TRACER working t: VirtualThread[#400030]/Runnable@ForkJoinPool-1-worker-5
TRACER working t: VirtualThread[#400033]/Runnable@ForkJoinPool-1-worker-1
TRACER working t: VirtualThread[#400034]/Runnable@ForkJoinPool-1-worker-8
TRACER working t: VirtualThread[#400036]/Runnable@ForkJoinPool-1-worker-2
```













OS (Kernel)





JVM



OS (Kernel)



JVM

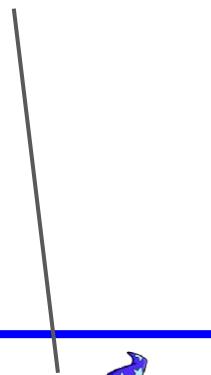


OS (Kernel)



Scheduler

JVM



OS (Kernel)



Scheduler

Heap

JVM



OS (Kernel)



Scheduler

Heap

Continuation

JVM



OS (Kernel)



Project Loom

~~Fibers~~ and Continuations





Project Loom

~~Fibers~~ and Continuations





Project Loom

~~Fibers~~ and Continuations



Pull Request

openjdk/jdk Public

Watch 298 Fork 3.6k Starred 13.2k

Code Pull requests 213 Security Insights

8284161: Implementation of Virtual Threads (Preview) #8166

Closed AlanBateman wants to merge 23 commits into `openjdk:master` from `AlanBateman:JDK-8284161`

Conversation 230 Commits 23 Checks 12 Files changed 1,133 +95,870 -8,270

Changes from all commits File filter Conversations Review changes

0 / 1133 files viewed

Pull Request

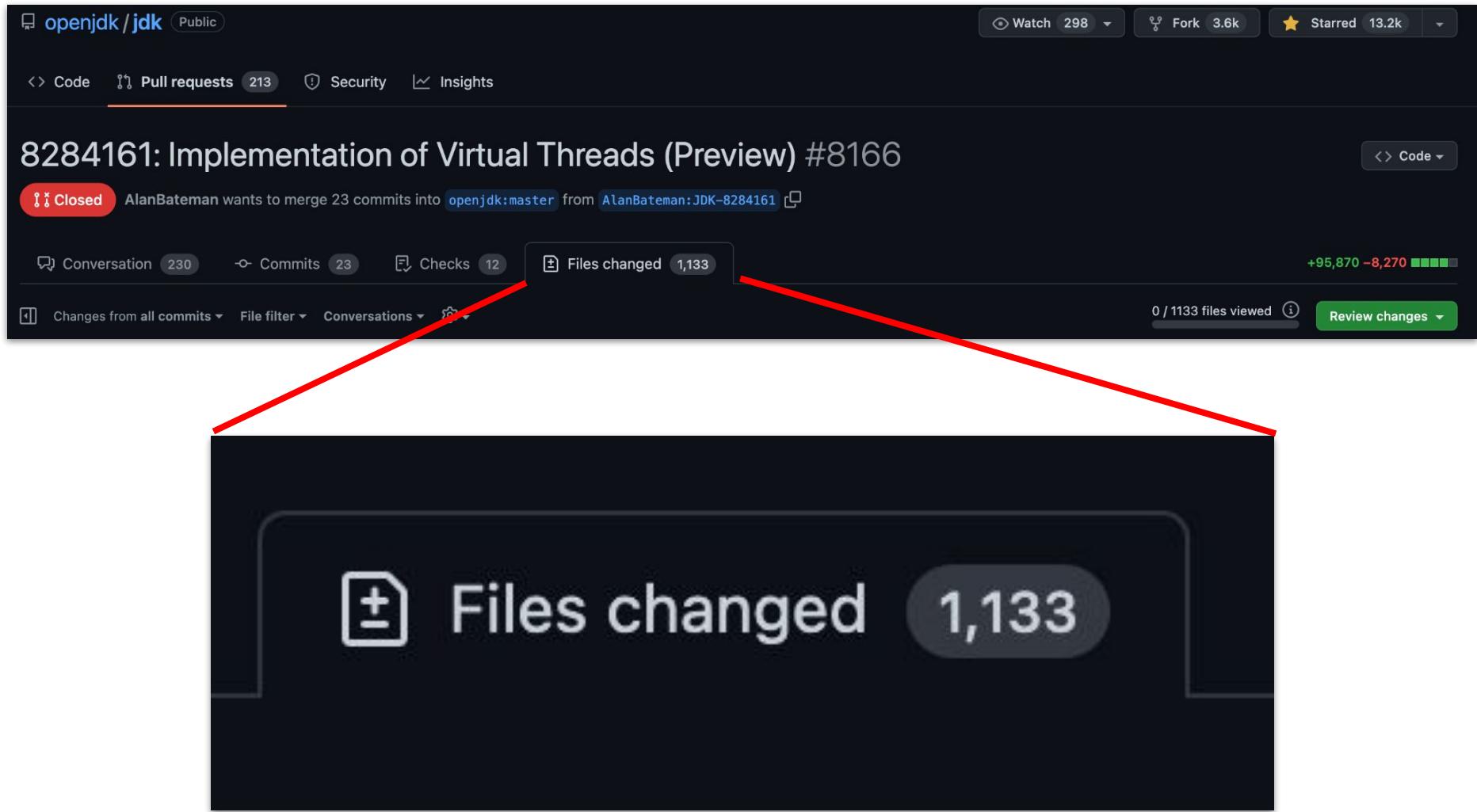
Screenshot of a GitHub Pull Request page for issue #8166.

The pull request is titled "8284161: Implementation of Virtual Threads (Preview) #8166". It is marked as "Closed" and shows AlanBateman merging 23 commits from "openjdk:master" into "AlanBateman:JDK-8284161".

Key statistics displayed on the page include:

- Conversation: 230
- Commits: 23
- Checks: 12
- Files changed: 1,133 (highlighted by a red box and arrow)
- Changes from all commits: 0 / 1133 files viewed
- Review changes button

A large red arrow points from the "Files changed" section at the top of the page down to the "Files changed" summary box at the bottom.



Pull Request

openjdk/jdk Public

<> Code Pull requests 213 Security Insights

8284161: Implementation of Virtual Threads (Preview) #8166

Closed AlanBateman wants to merge 23 commits into `openjdk:master` from `AlanBateman:JDK-8284161` ⚙

Conversation 230 Commits 23 Checks 12 Files changed 1,133 +95,870 -8,270

Changes from all commits ▾ File filter ▾ Conversations ▾ ⚙ 0 / 1133 files viewed ⓘ Review changes ▾



Pull Request

openjdk/jdk Public

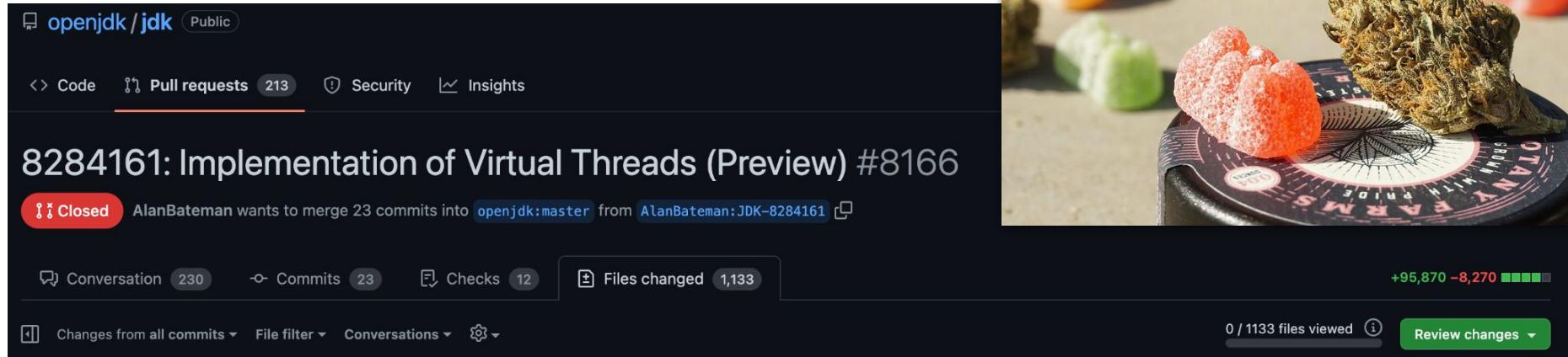
<> Code Pull requests 213 Security Insights

8284161: Implementation of Virtual Threads (Preview) #8166

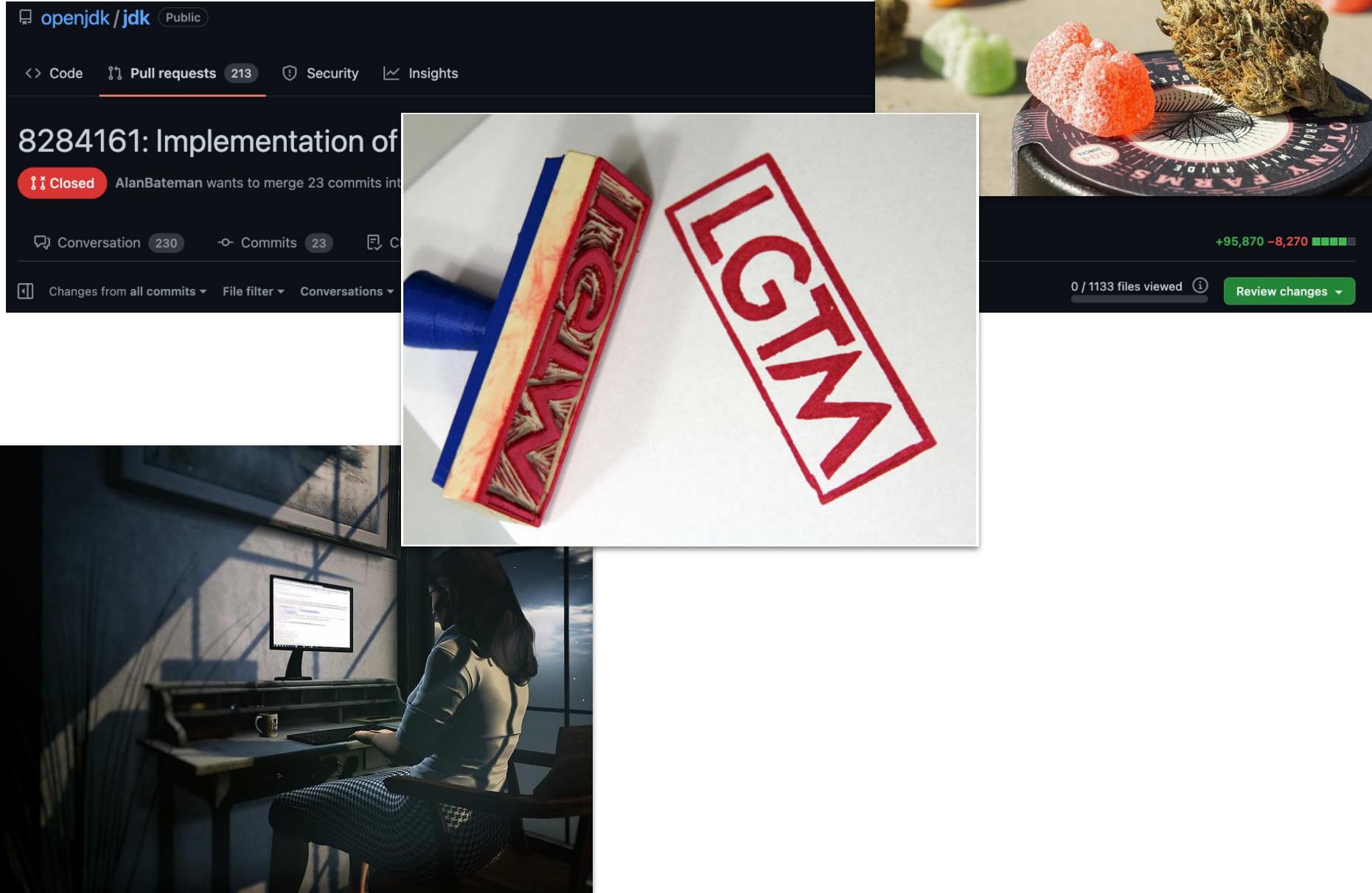
Closed AlanBateman wants to merge 23 commits into `openjdk:master` from `AlanBateman:JDK-8284161`

Conversation 230 Commits 23 Checks 12 Files changed 1,133 +95,870 -8,270

Changes from all commits ▾ File filter ▾ Conversations ▾ Review changes ▾



Pull Request



Codes like sync, scales like async

```
49 void run() throws Exception {  
50     try (var executor = Executors.newVirtualThreadPerTaskExecutor()) {  
51         int numTasks = 10;  
52         for (int i = 0; i < numTasks; i++) {  
53             executor.submit(new MyTask(i, database));  
54         }  
55     }  
56 }
```

Codes like sync, scales like async

```
30 class MyTask implements Runnable {  
31     @Override  
32     public void run() {  
33         var user = database.findUser(id);  
34         // ...  
35     }
```

Pros

JVM



Pros

sync

JVM



Pros

sync

scale

JVM



Pros

sync

scale

JVM

simple



Pros

sync

scale

JVM



simple

debug,
tooling,
monitoring

Loom C5M

:≡ README.md

Project Loom C5M

Project Loom C5M is an experiment to achieve 5 million persistent connections each in client and server Java applications using OpenJDK Project Loom virtual threads.

Hype



r/java

Posts



Posted by u/Carlislee 1 year ago 



Loom cant come fast enough

Complete throwaway post but I am super excited for project Loom. Tons of complexity is about to be thrown out the window when it comes to WebFlux, CompletableFuture, complex Flow<> objects....

Hype



joshlemer · 1 yr. ago

Couldn't agree more! What if the whole reactive programming paradigm (futures, observables, ...) turns out to be a temporary fad because we just didn't have the tools to simply program with light threads?

 28 

[Give Award](#) [Share](#) [Report](#) [Save](#)

Hype



african_or_european · 1 yr. ago

I hate reactive java so much and I've fought it tooth and nail at work, but was never able to clearly describe why it's so terrible besides basically "... just look at it!" I'm absolutely going to be using this.

 13 

[Give Award](#) [Share](#) [Report](#) [Save](#)

“Loom will kill Reactive programming”

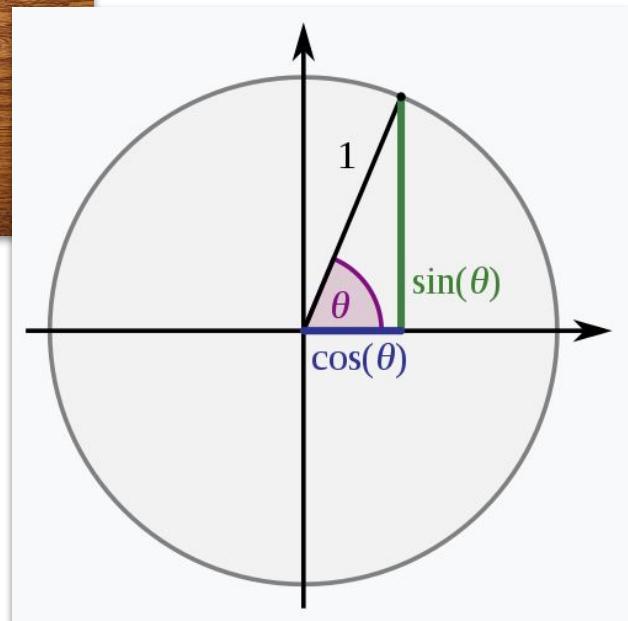


OSSA ROMANI PRES.
IN CEMET. S. HELENAE
VIA LABIC.
SUB PRÆPOSITO LAPIDE
REPERTA
HIC SITA SUNT
A. D. MDCCXLIX

OSSA ROMANI PRESA
IN COEMET^A S^A HELENAE
VIA LABIC^A
SUB PR^APOSITO LAPIDE
REPERTA
HIC SITA SUNT
A^D MDCCXLIX



OSSA ROMANI PRESA
IN COEMET^A S^A HELENAE
VIA LABIC^A
SUB PR^APOSITO LAPIDE
REPERTA
HIC SITA SUNT
A^D MDCCXLIX



Reactive

:-|



Oleh Dokuka

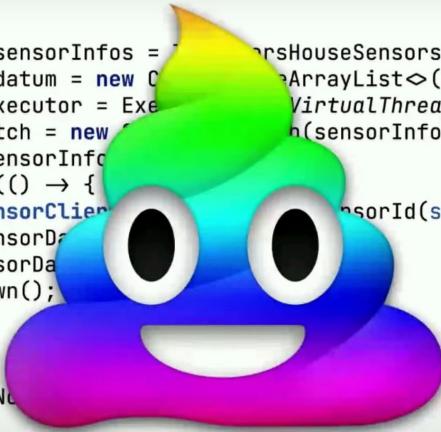
:-|



:-|



```
List<SensorInfo> sensorInfos = ...  
List<SensorData> datum = new ConcurrentSkipListArrayList<>();  
ExecutorService executor = Executors.newVirtualThreadPerTaskExecutor();  
CountDownLatch latch = new CountDownLatch(sensorInfos.size());  
for (SensorInfo sensorInfo : sensorInfos) {  
    executor.submit(() -> {  
        double t = sensorClient.readDouble(sensorId(sensorInfo.id()));  
        SensorData sensorData = new SensorData(t);  
        datum.add(sensorData);  
        latch.countDown();  
    });  
}  
latch.await(1000, TimeUnit.MILLISECONDS);  
executor.shutdownNow();
```



A large, colorful emoji of a smiling face, transitioning through the colors of a rainbow (red, orange, yellow, green, blue, purple). It has large, expressive eyes and a wide, open-mouthed smile.



ReactiveX

An API for asynchronous programming
with observable streams

Choose your platform

Languages

- Java: [RxJava](#)
- JavaScript: [RxJS](#)
- C#: [Rx.NET](#)
- C#(Unity): [UniRx](#)
- Scala: [RxScala](#)
- Clojure: [RxClojure](#)
- C++: [RxCpp](#)
- Lua: [RxLua](#)
- Ruby: [Rx.rb](#)
- Python: [RxPY](#)
- Go: [RxGo](#)
- Groovy: [RxGroovy](#)
- JRuby: [RxJRuby](#)
- Kotlin: [RxKotlin](#)
- Swift: [RxSwift](#)
- PHP: [RxPHP](#)
- Elixir: [reaxive](#)
- Dart: [RxDart](#)

ReactiveX

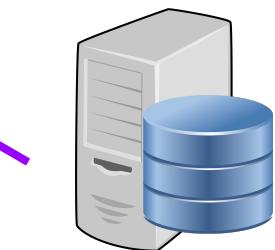
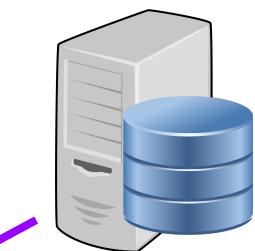
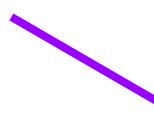
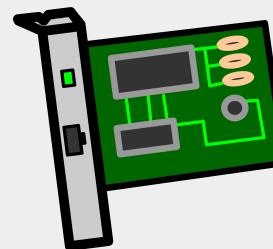
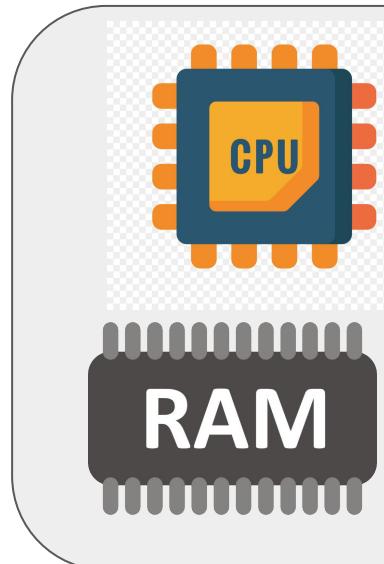
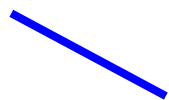
An API for asynchronous programming
with observable streams

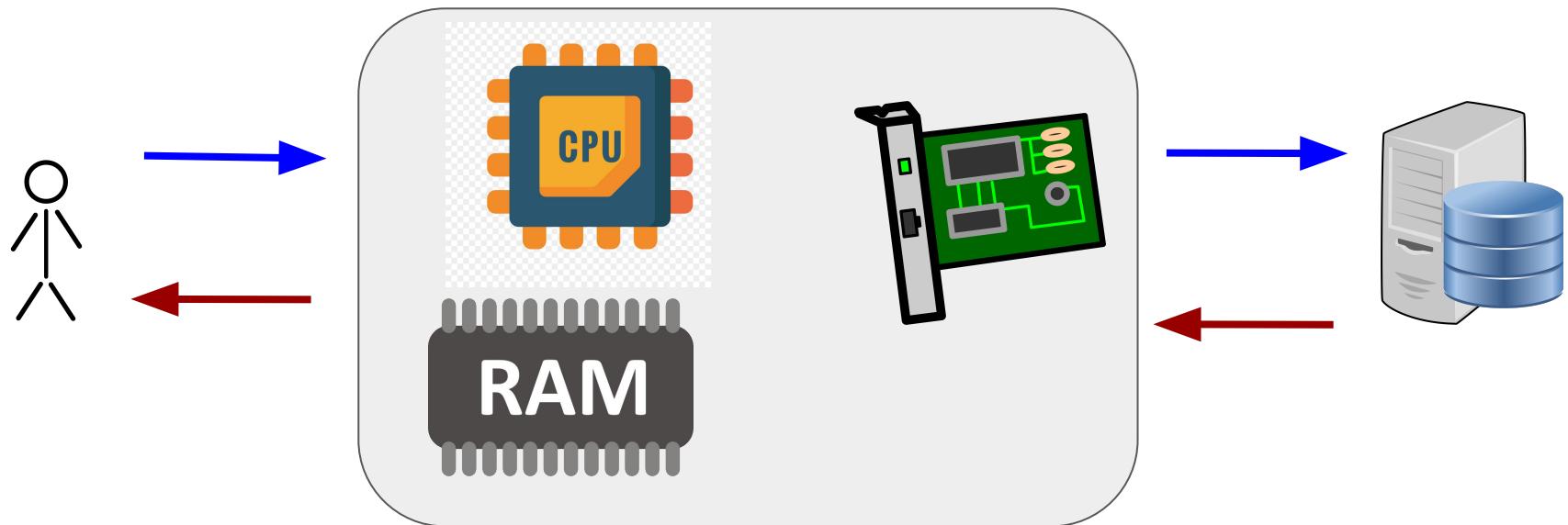
Choose your platform

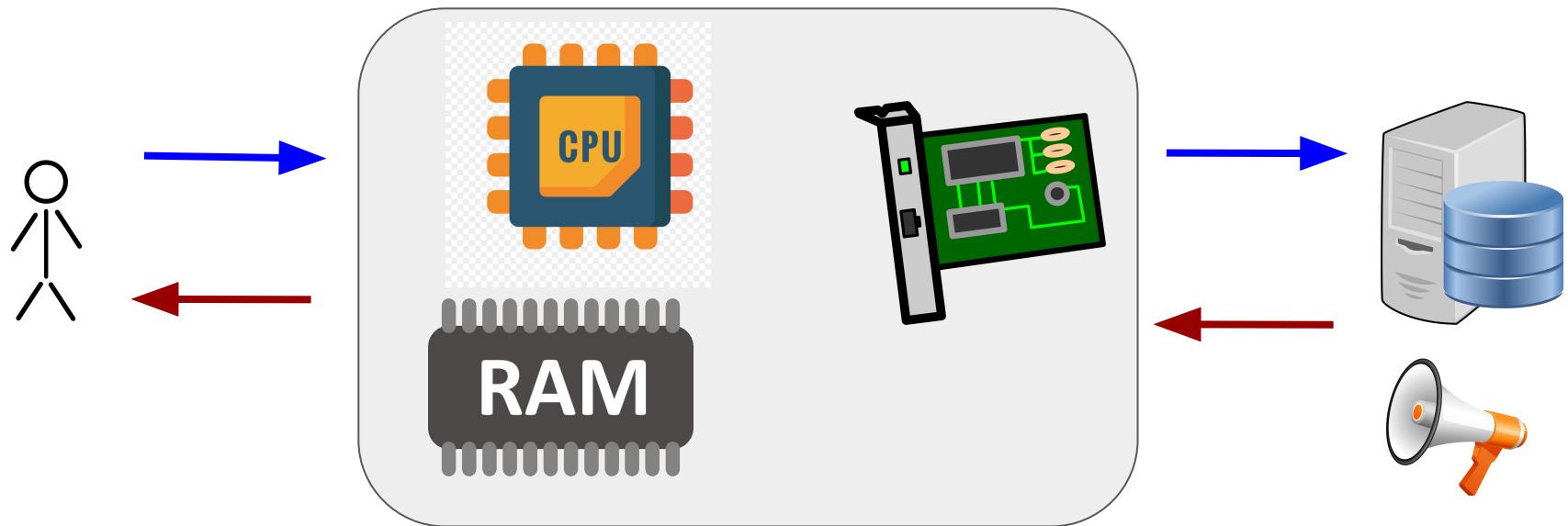
The Reactive Manifesto

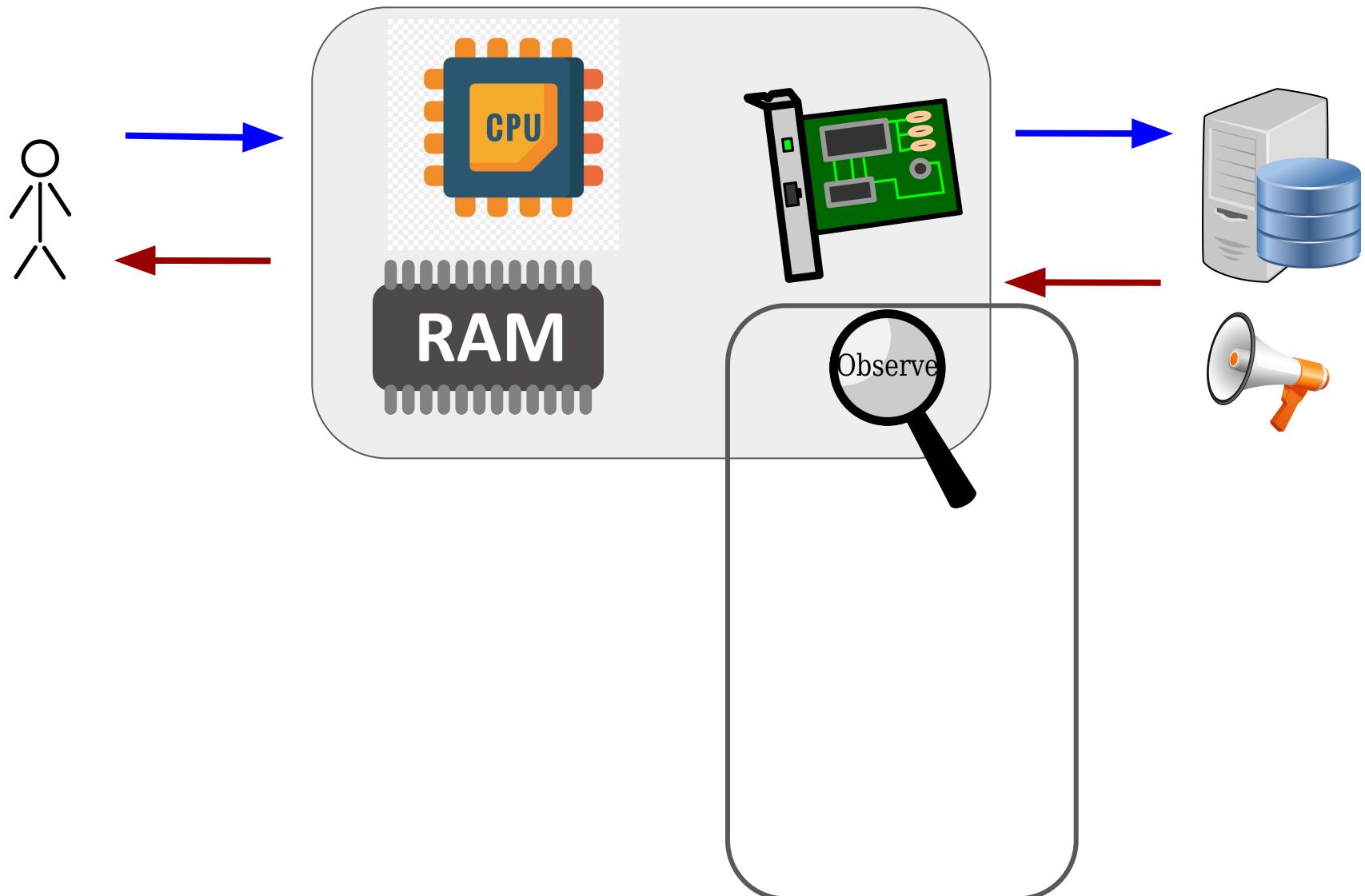
Published on September 16 2014. (v2.0)

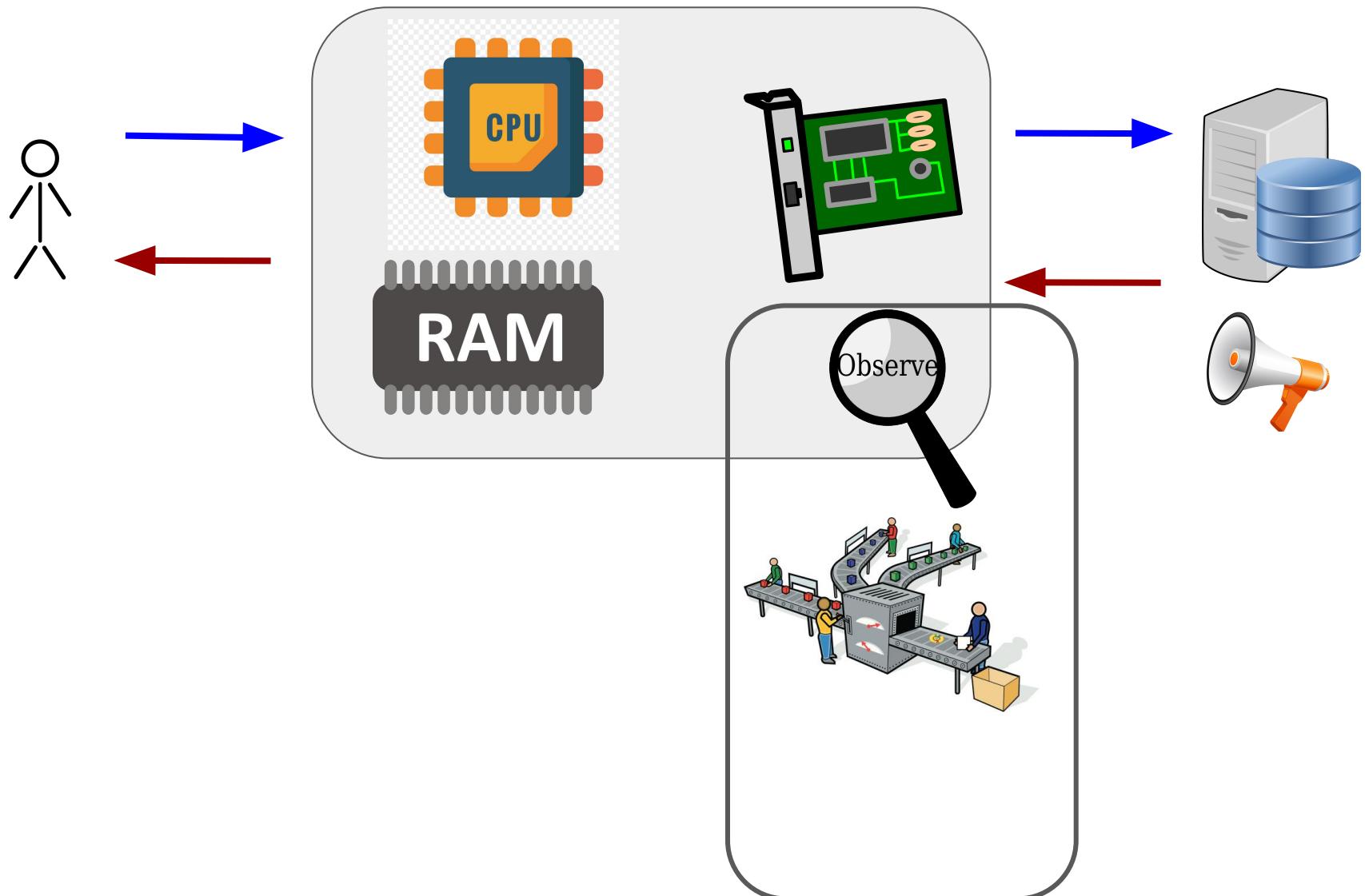


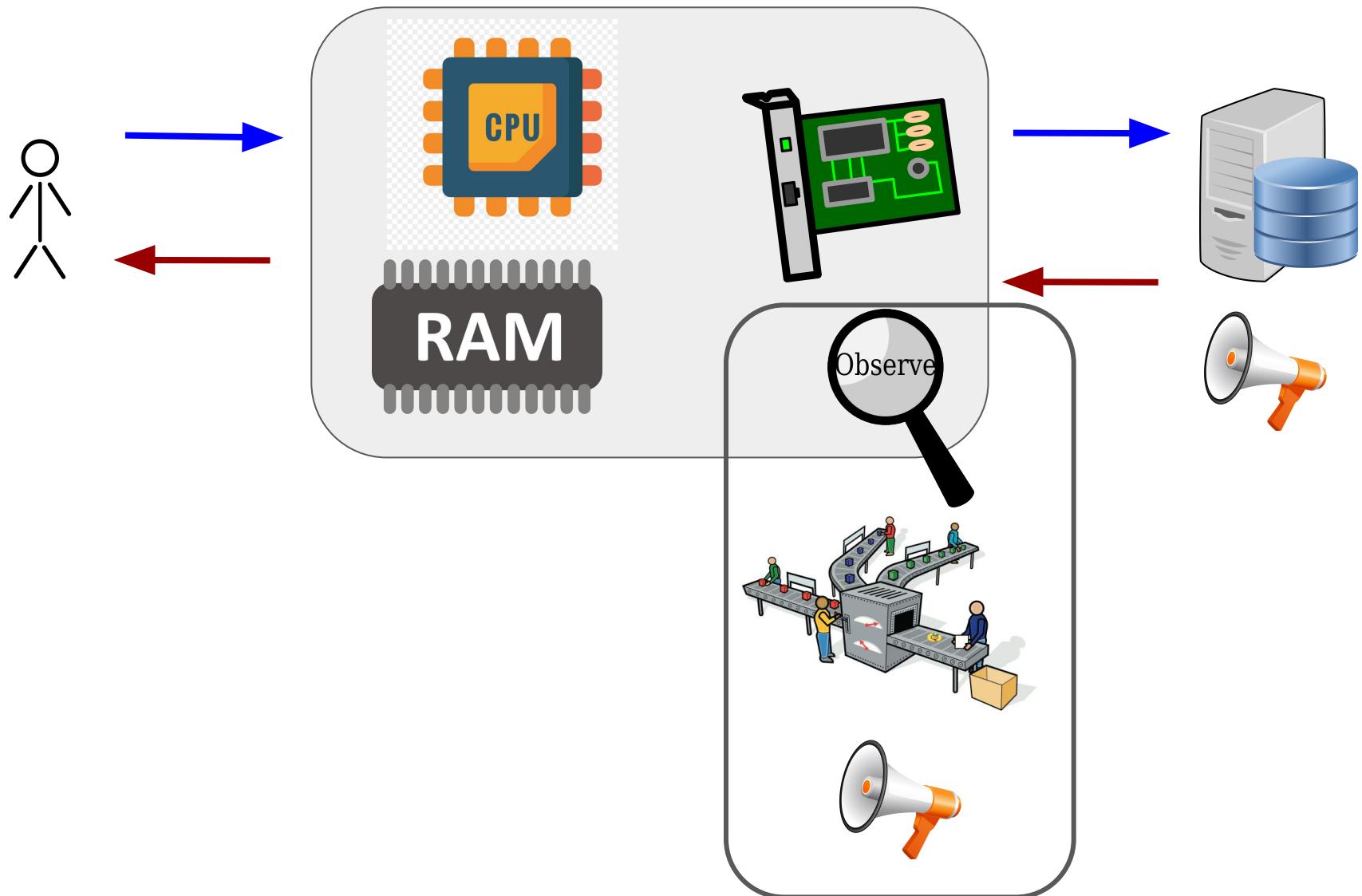


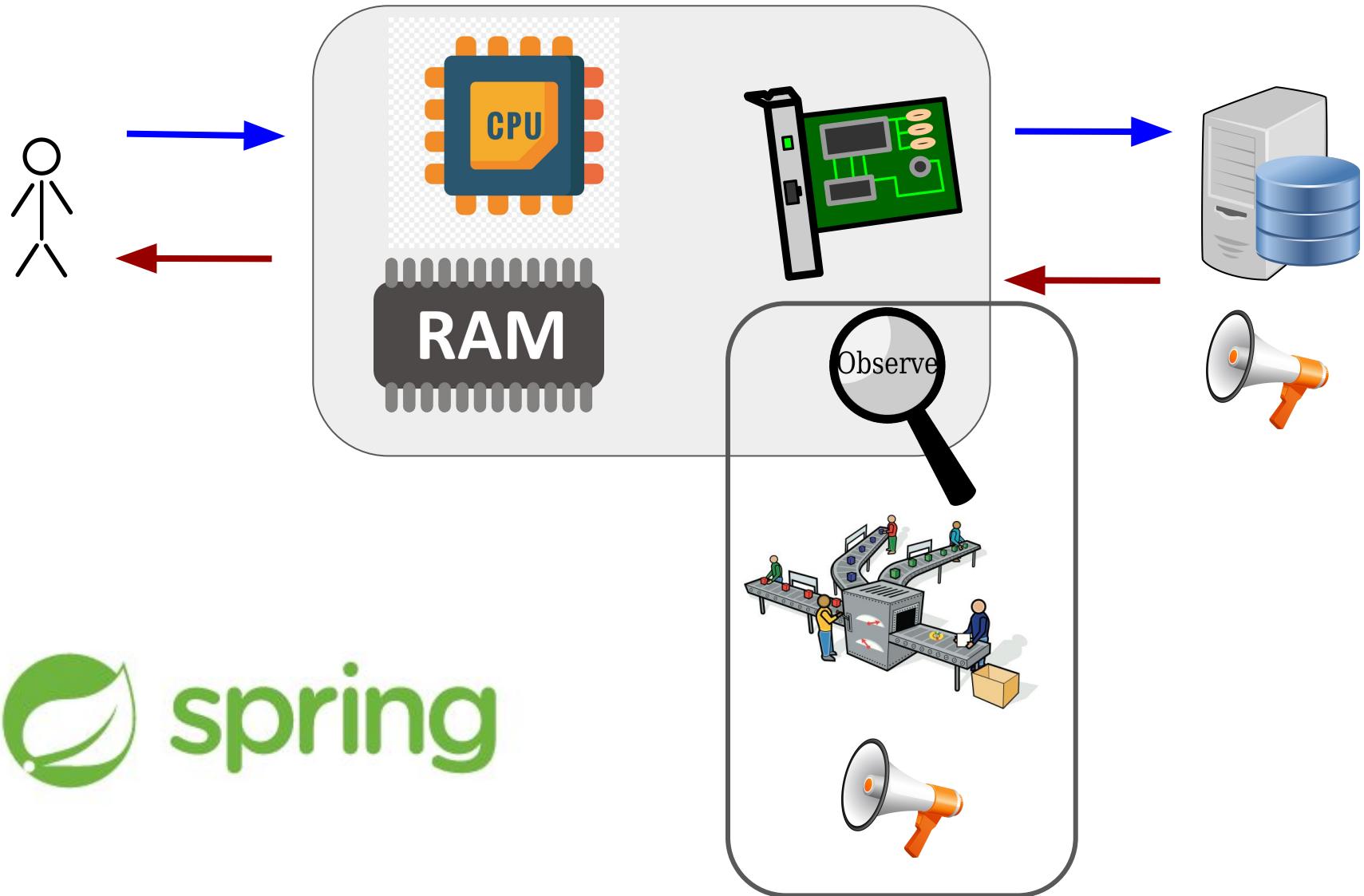












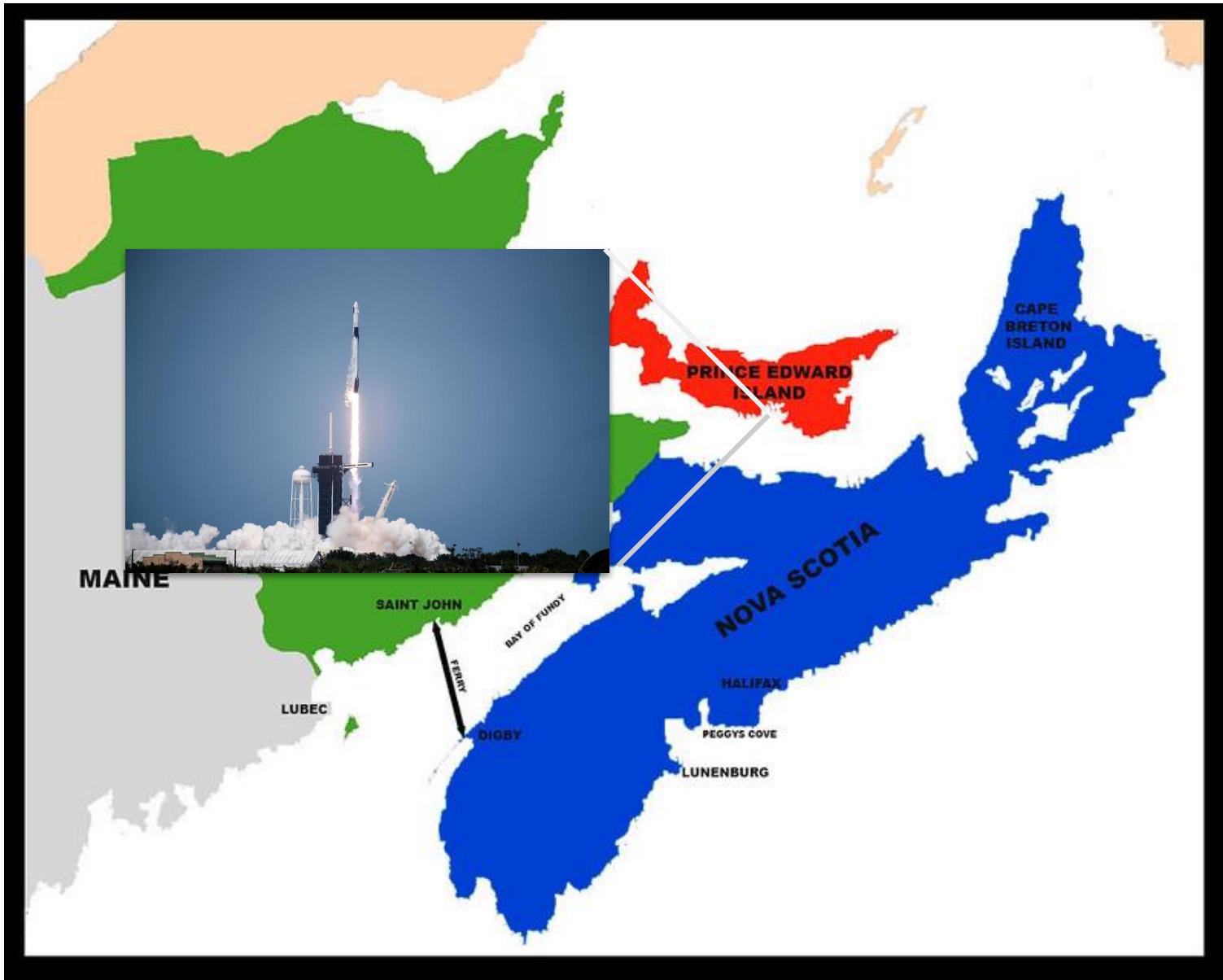
FAANG

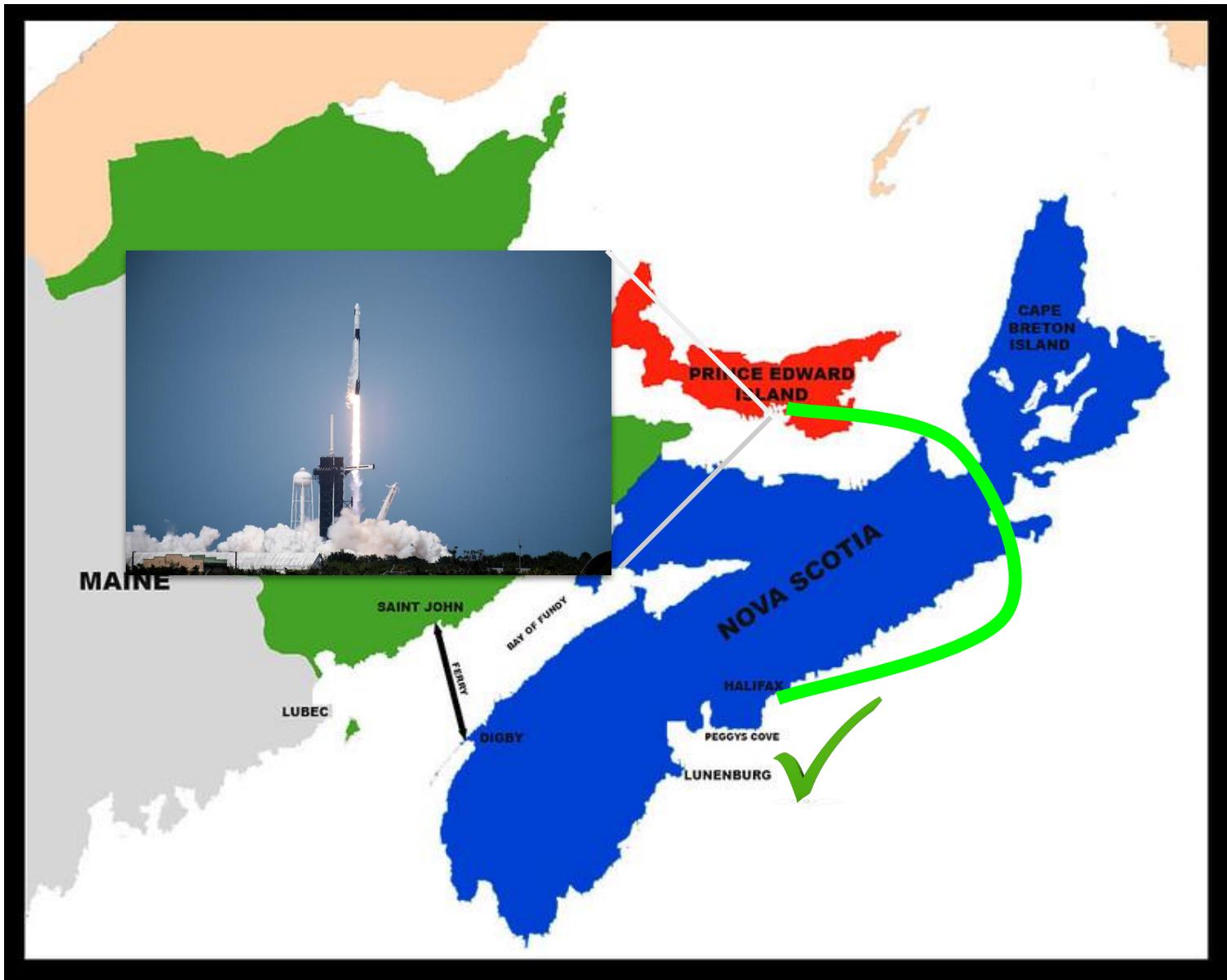


~~FAANG~~ -> MAANA







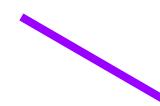
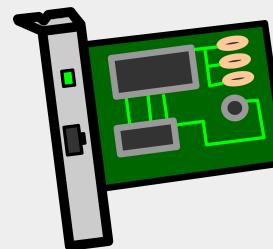
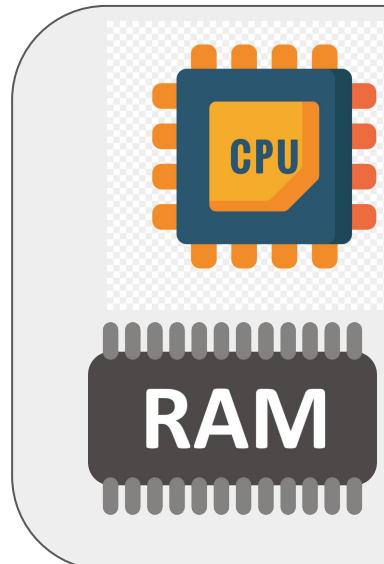
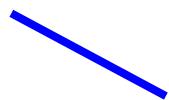


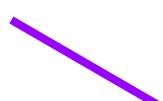
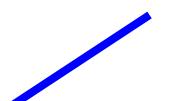
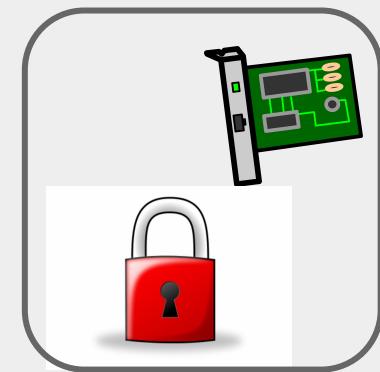
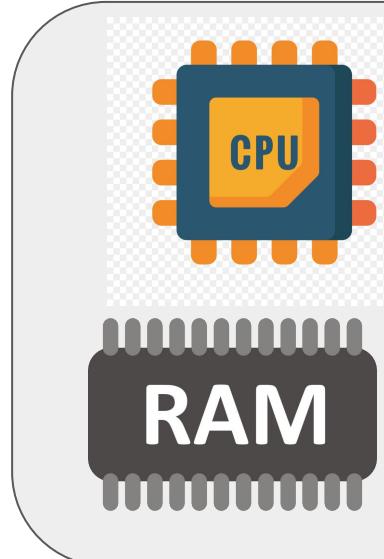
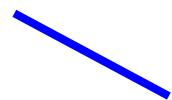
Cons

scale

JVM







Locks

```
45  class MyTask implements Runnable {  
46      @Override  
47      public void run() {  
48          try {  
49              var database = pool.acquireDatabase();  
50              var user = database.findUser(id);  
51              // ...  
52          } catch (Exception ex) {  
53              // #yolo  
54          } finally {  
55              pool.releaseDatabase();  
56          }  
57      }  
58  }
```

Locks

```
30  class ConnectionPool {  
31      private static final int MAX_AVAIL = 2;  
32      private final Semaphore available = new Semaphore(MAX_AVAIL);  
33      private final Database database = new Database();  
34  
35      Database acquireDatabase() throws InterruptedException {  
36          available.acquire();  
37          return database;  
38      }  
39  
40      void releaseDatabase() {  
41          available.release();  
42      }  
43  }
```

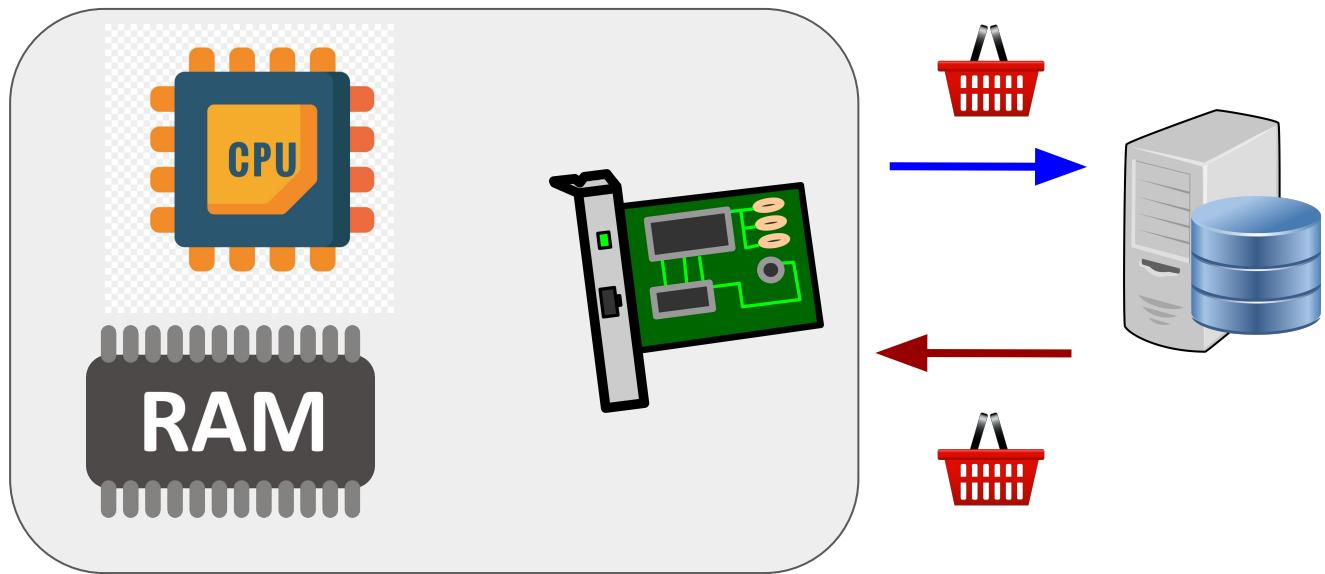
Cons

JVM



scale

back
pressure







☰ README.md

Project Loom C5M

Project Loom C5M is an experiment to achieve 5 million parallel client and server Java applications using [OpenJDK Project Loom](#).



☰ README.md

Project Loom C5M

Project Loom C5M is an experiment to achieve 5 million p
client and server Java applications using [OpenJDK Project](#)





☰ README.md

Project Loom C5M

Project Loom C5M is an experiment to achieve 5 million parallel client and server Java applications using OpenJDK Project Loom.



Structured Concurrency

```
1 i=0;  
2 i=i+1;  
3 PRINT i; "squared=";i*i;  
4 IF i>=100 THEN GOTO 6;  
5 GOTO 2;  
6 PRINT "Program Completed.";  
7 END
```

STRUCTURED PROGRAMMING AND PROBLEM-SOLVING WITH PASCAL

Richard B. Kleburz

`Z := 1;`

`while N > 0`

`if N is odd`

`then`

`Z := Z * E;`

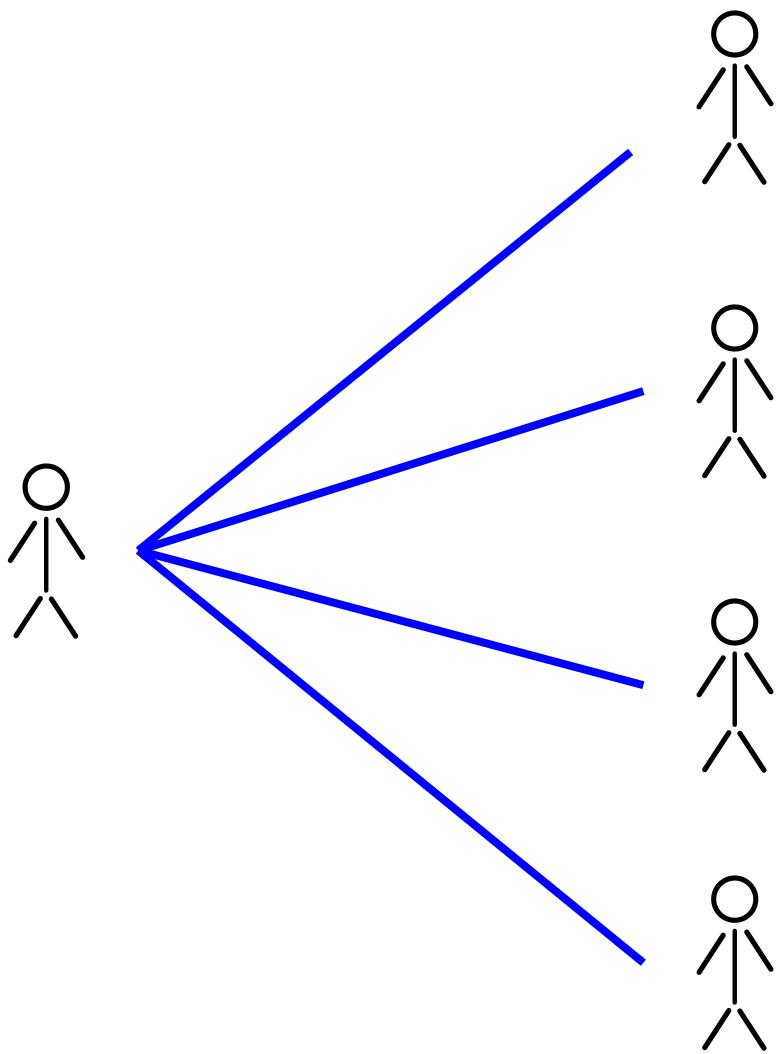
`N := N - 1;`

`else`

`E := E * E;`

`N := N div 2;`

`result is Z`



njs blog

[My homepage](#)

| [Blog archive](#)

WED 25 APRIL 2018

Notes on structured concurrency, or:

njs blog

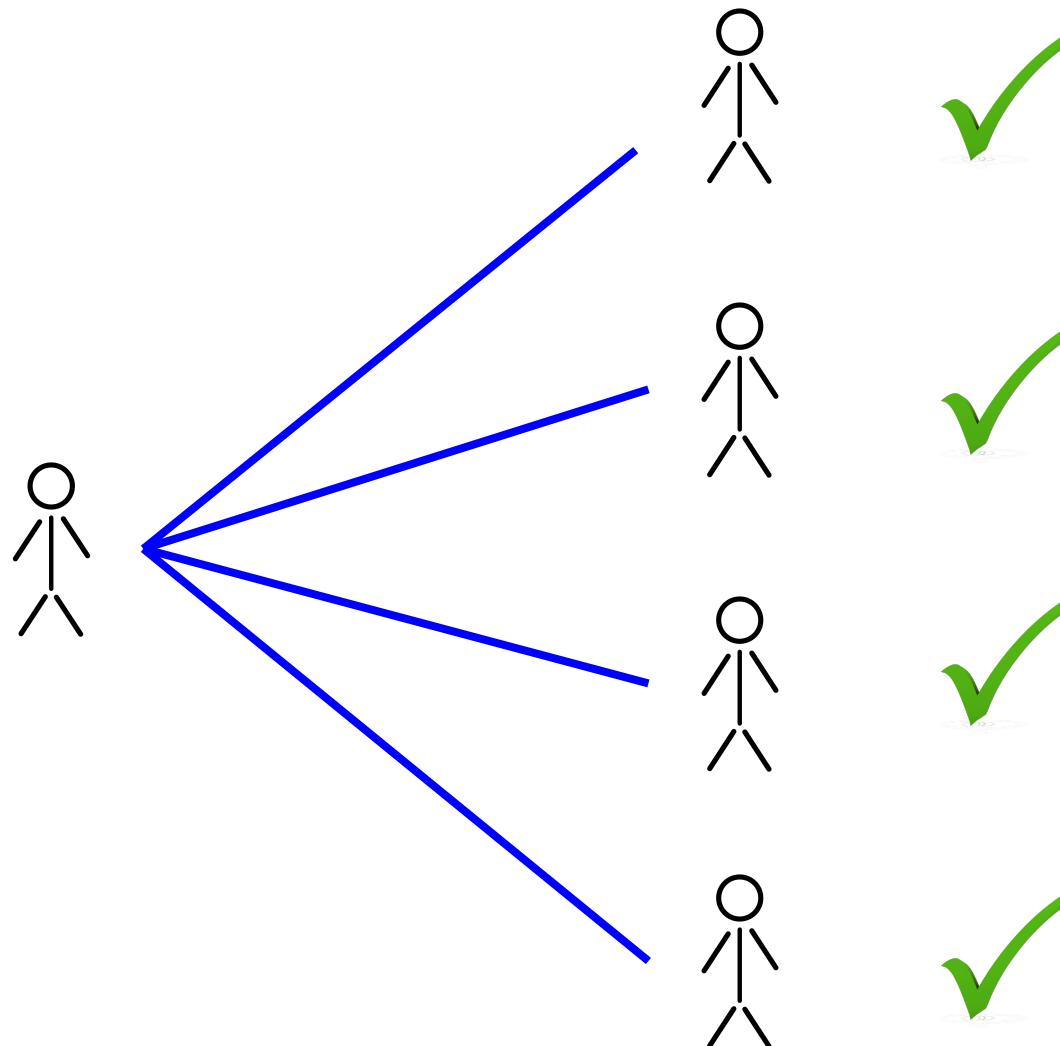
[My homepage](#)

| [Blog archive](#)

WED 25 APRIL 2018

Notes on structured concurrency, or: Go statement considered harmful

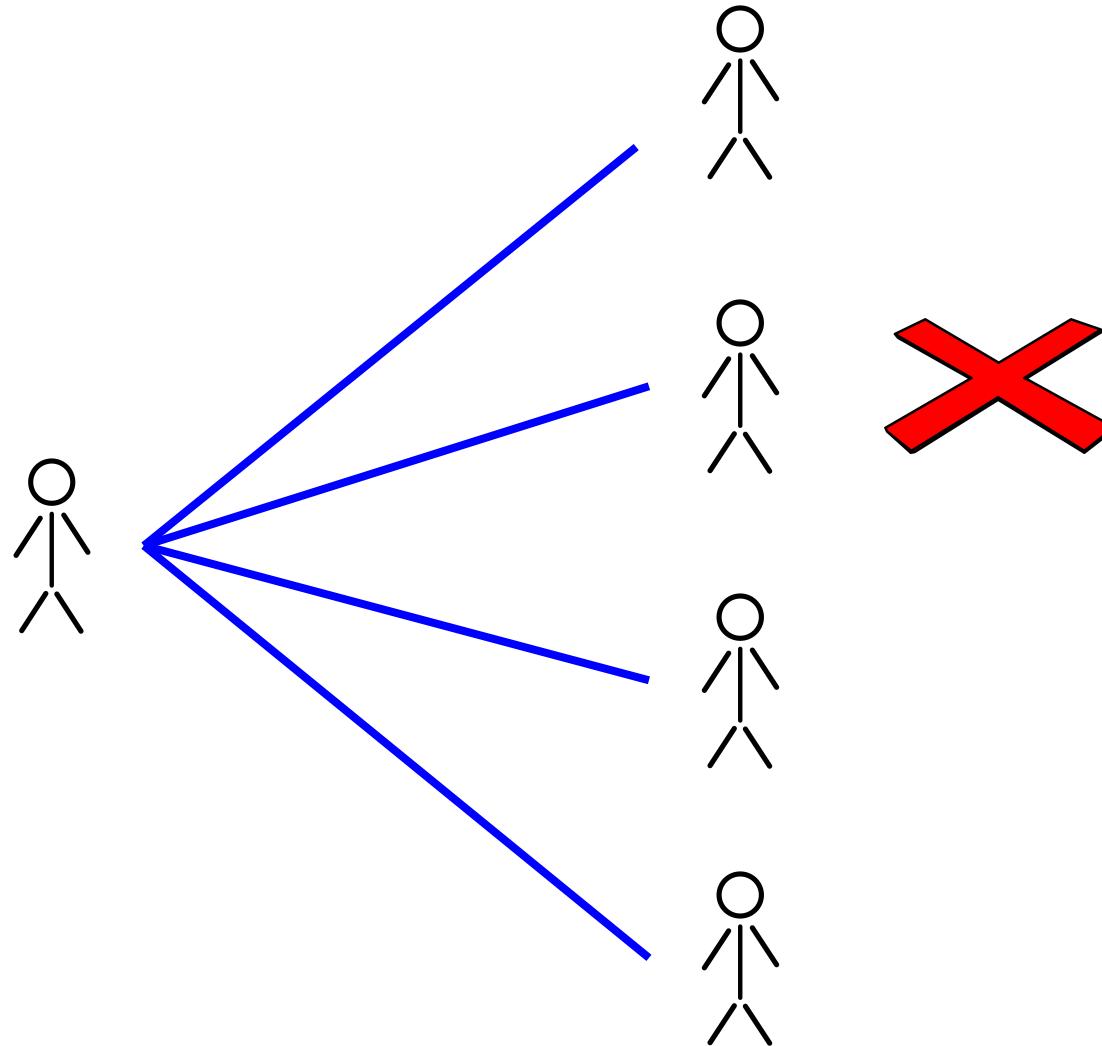
Invoke All



Invoke All

```
35 ~ String run() throws Exception {  
36     try (var scope = new StructuredTaskScope.ShutdownOnFailure()) {  
37         Future<String> foo = scope.fork(() -> taskFoo());  
38         Future<String> bar = scope.fork(() -> taskBar());  
39  
40         scope.join();  
41  
42         return foo.resultNow() + " " + bar.resultNow();  
43     }  
44 }
```

Invoke All



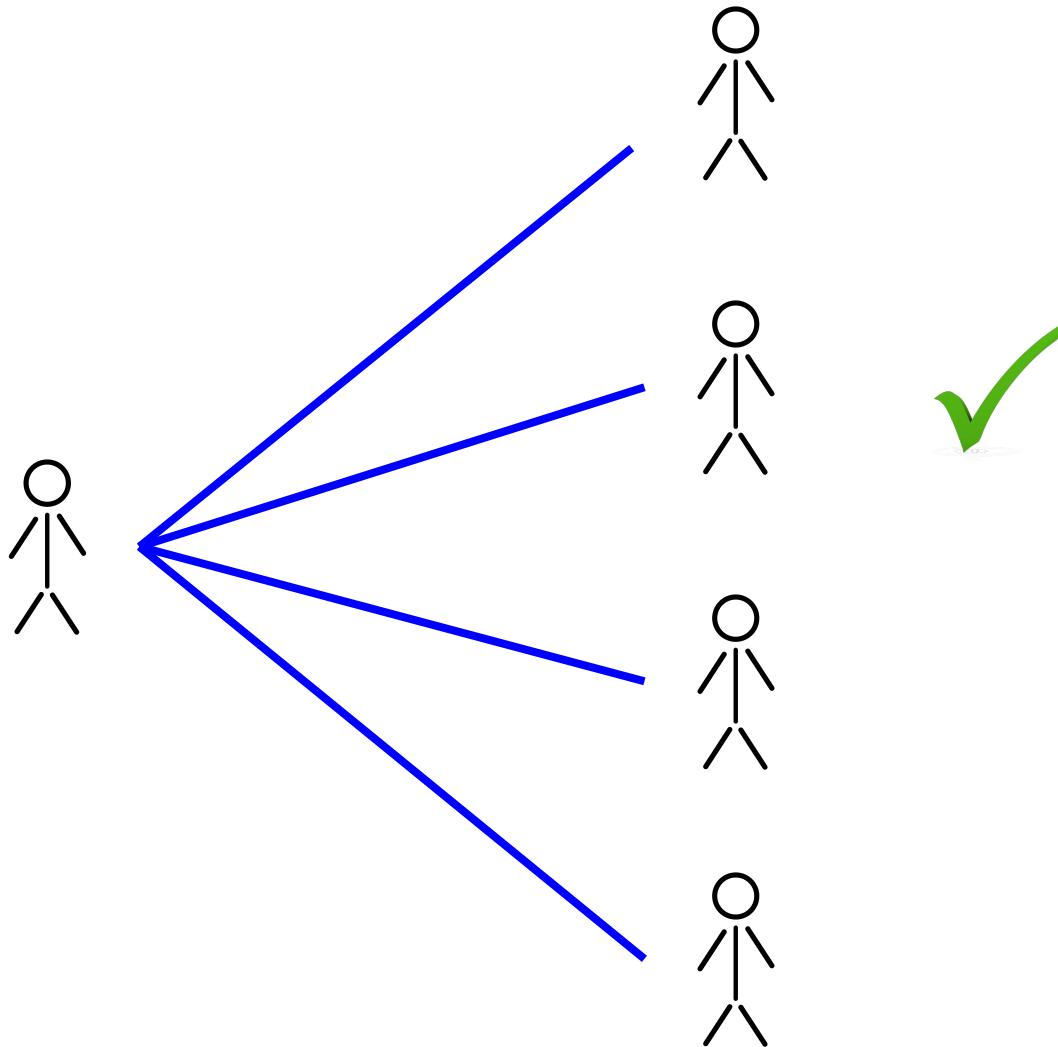
Invoke All

```
35 String run() throws Exception {  
36     try (var scope = new StructuredTaskScope.ShutdownOnFailure()) {  
37         Future<String> foo = scope.fork(() -> taskFoo());  
38         Future<String> bar = scope.fork(() -> taskBar());  
39  
40         scope.join();  
41         scope.throwIfFailed();  
42  
43         return foo.resultNow() + " " + bar.resultNow();  
44     }  
45 }
```

JVM



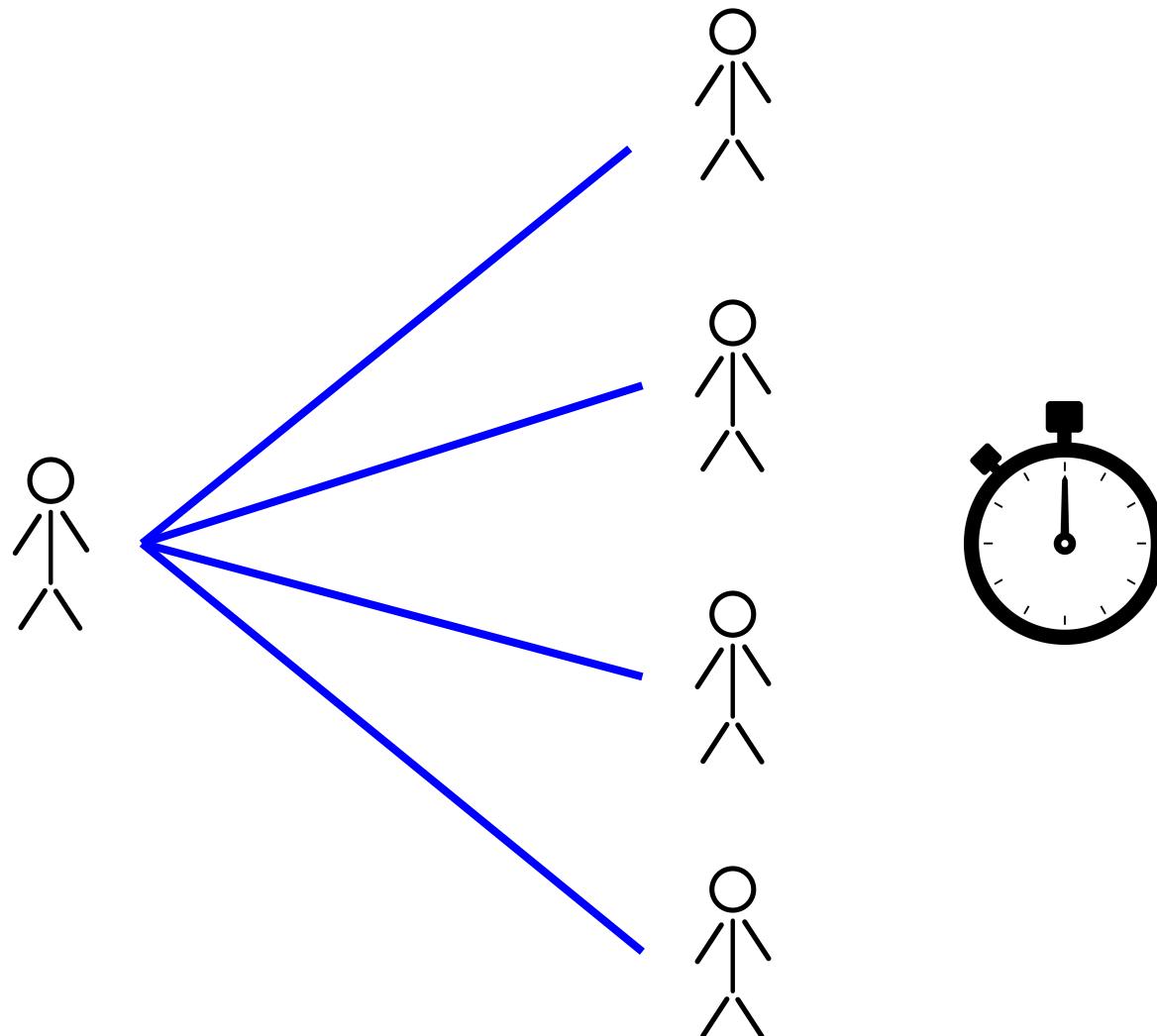
Invoke Any



Invoke Any

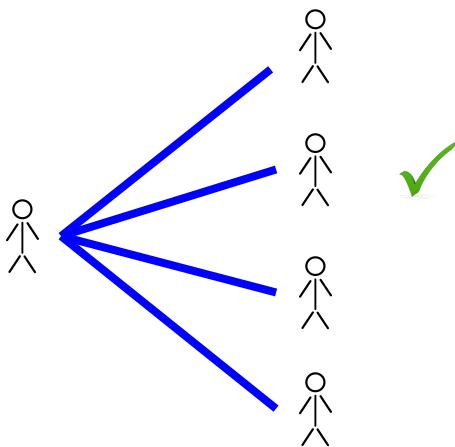
```
35 String run() throws Exception {
36     try (var scope =
37             new StructuredTaskScope.ShutdownOnSuccess<String>()) {
38         Future<String> foo = scope.fork(() -> taskFoo());
39         Future<String> bar = scope.fork(() -> taskBar());
40
41         scope.join();
42
43         return scope.result();
44     }
45 }
```

Deadline



Deadline

```
35 | String run() throws Exception {
36 |     try (var scope =
37 |             new StructuredTaskScope.ShutdownOnSuccess<String>()) {
38 |         Future<String> foo = scope.fork(() -> taskFoo());
39 |         Future<String> bar = scope.fork(() -> taskBar());
40 |
41 |         var deadline = Instant.now().plusSeconds(secondsToAdd: 2);
42 |         scope.joinUntil(deadline);
43 |
44 |         return scope.result();
45 |     }
46 | }
```



☰ README.md

Project Loom C5M

Project Loom C5M is an experiment to achieve 5 million parallel client and server Java applications using OpenJDK Project Loom.



Code, Credits, Links

Available [here](#) on GitHub

