Build a web scraper based on virtual threads!

@David_Vlijmincx
www.davidvlijmincx.com | Oracle ACE

Outline

- Set up
- Threads and virtual threads
- Thread performance (How fast can you scrape)
- Structured concurrency
- Scoped values
- Bonus features ☺ (Diving deeper into the details)
- Show your Scraper/ freestyle assignments (If you want)

What is a web scraper

- A bot
- Downloads web pages to find new web pages
- GOTO 1

What is a web scraper

```
Crawled 2000 web page(s)Total execution time: 1605ms
Throughput: 1246.1059190031153 pages/sec

Process finished with exit code 0
```

Set up of the workshop

- The repo will have a README with assignments
- Every assignment has its own starter branch with the same name

For each assignment we will go through this loop:

- While(thisFormatIsOke)
  - Small presentation
  - Time to code!
  - Small demo

Time for step 1: Project setup

Set up

- SSID: ---
- WIFI password: ---

What You need:

- https://github.com/davidtos/virtual_thread_workshop
- IDE
- Java 21

ChatGPT use at your own risk

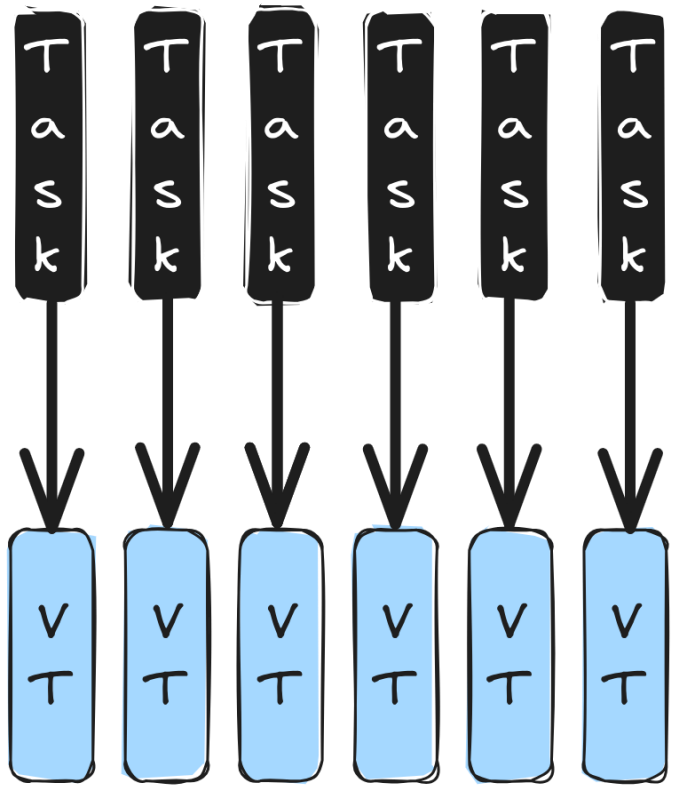
(Platform) Threads

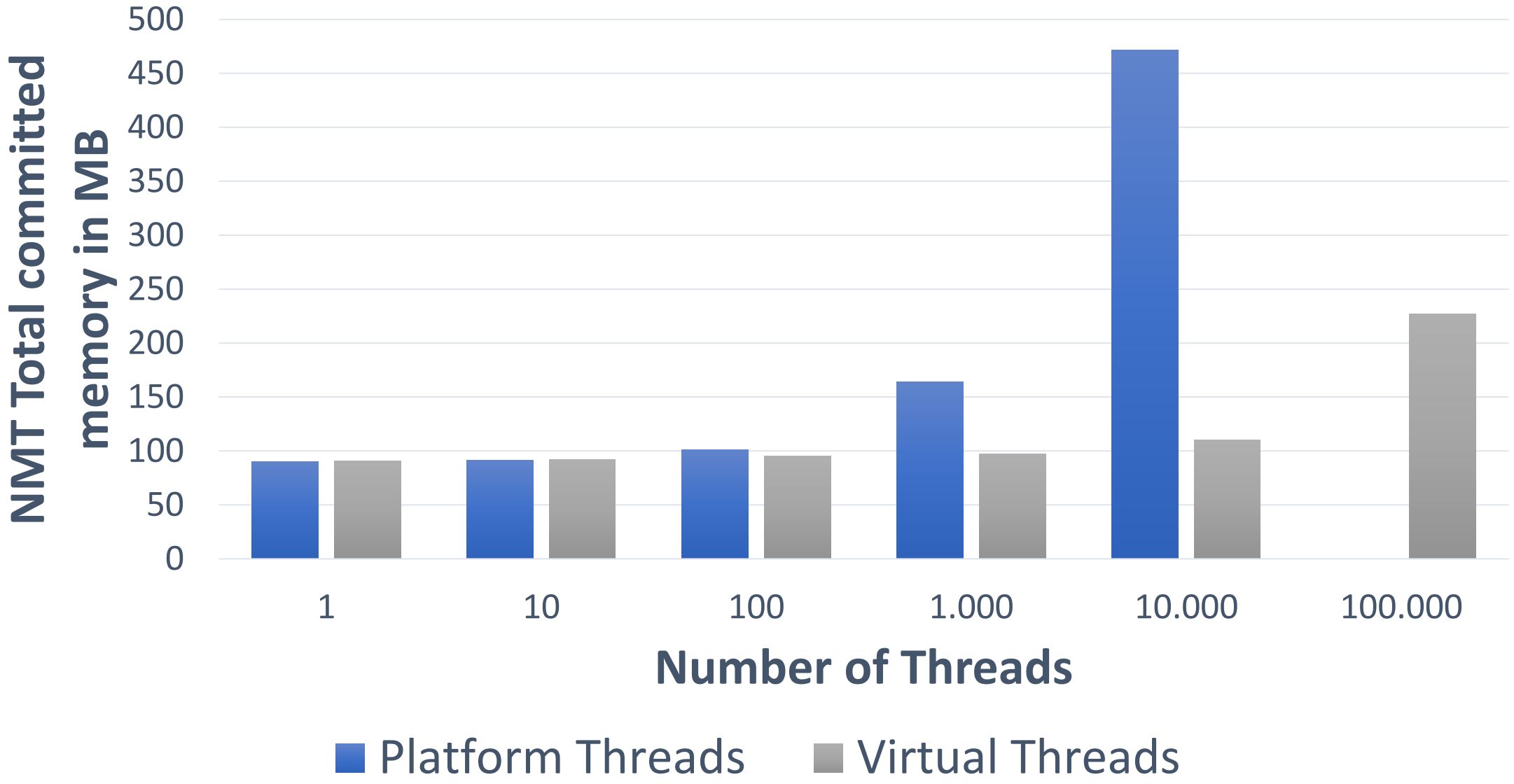
```
var thread = new Thread(task);
thread.start();

var executor = Executors.newFixedThreadPool(
        Runtime.getRuntime().availableProcessors()
)
```

Time for step 2: Adding threads

Virtual Threads

- Alternative implementation of Threads
- The OS does not know about them
  - Concept inside the JVM
  - Stack frames live on the heap
  - Resizable stacks
- You don’t have to allocate a lot of memory at the start
- Cheap to create




NMT Total committed memory in MB
500
450
400
350
300
250
200
150
100
50
0
1
10
100
1.000
10.000
100.000
Number of Threads
Platform Threads
Virtual Threads

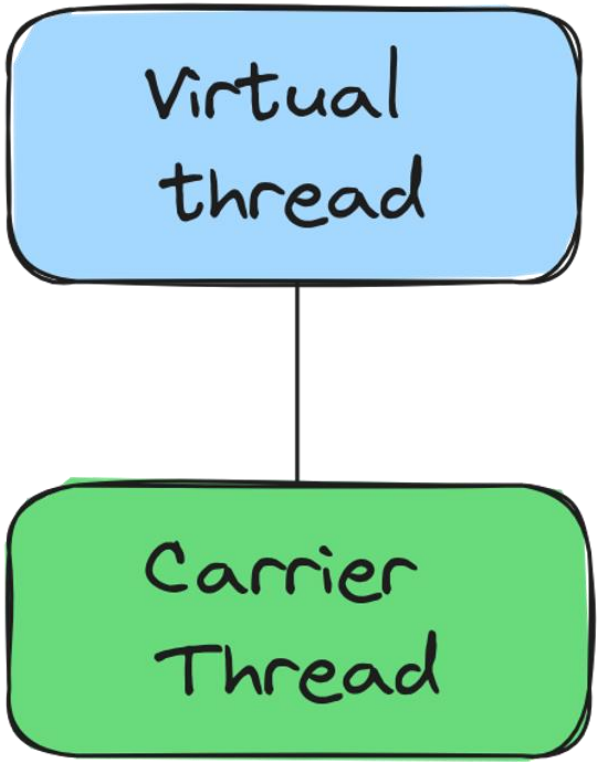
Virtual Threads

Thread.startVirtualThread(RunnableTask);

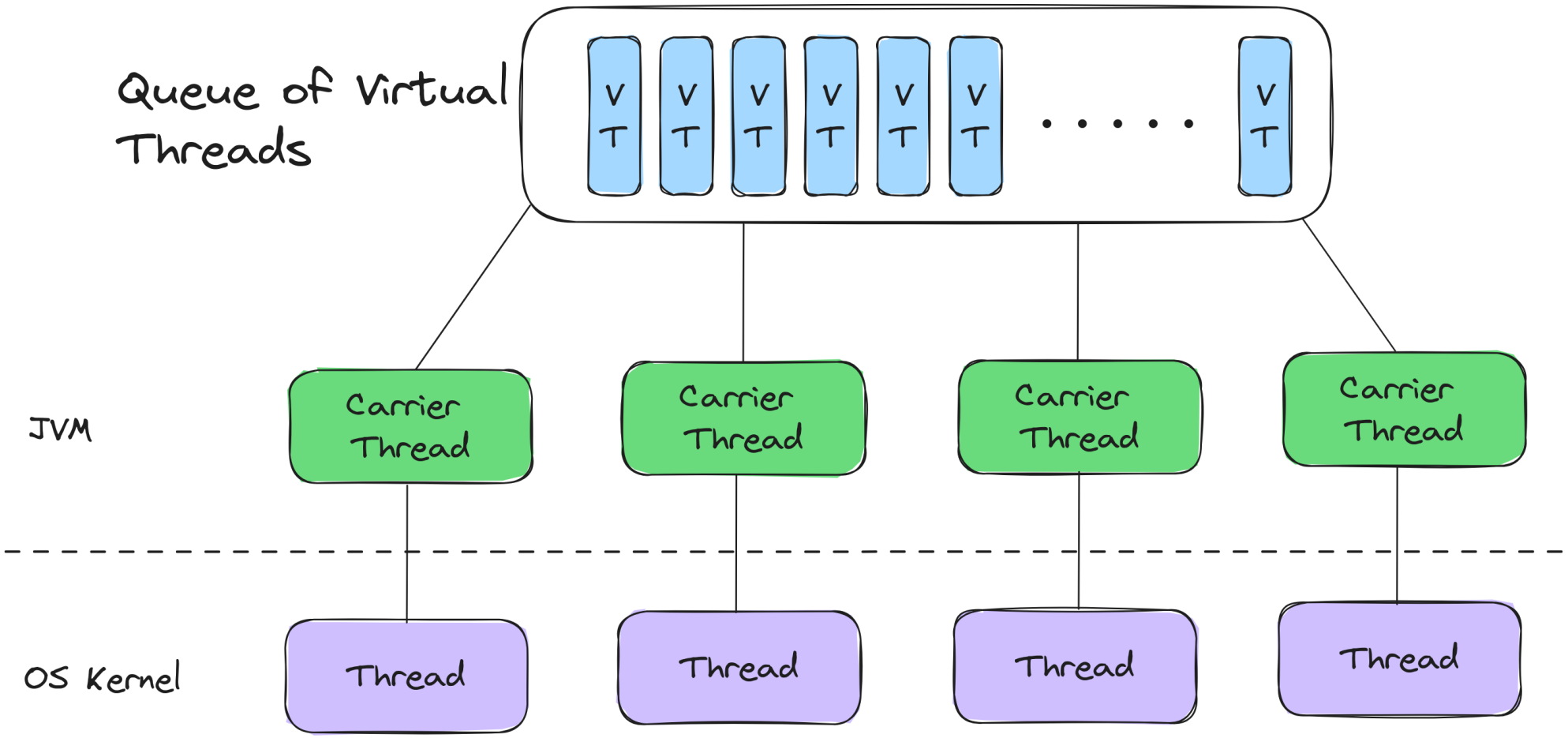
Executors.newVirtualThreadPerTaskExecutor()

Time for step 3: Virtual threads

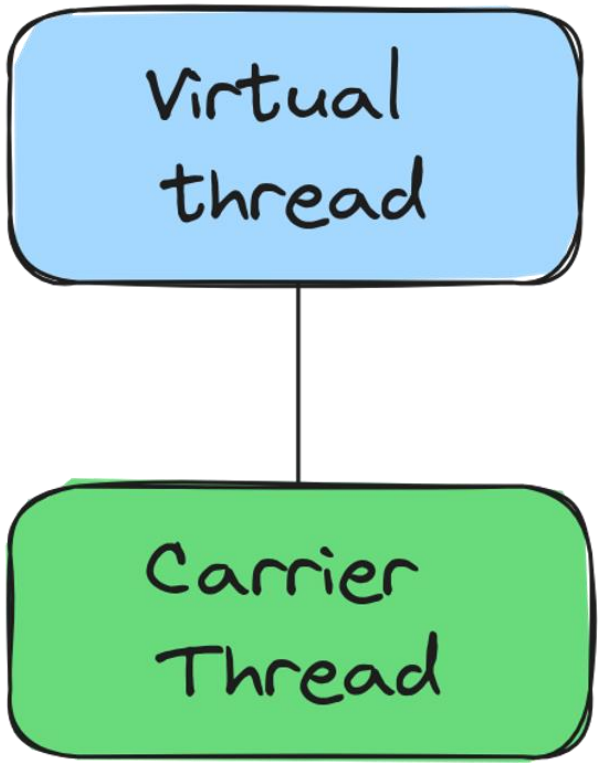
How does it work



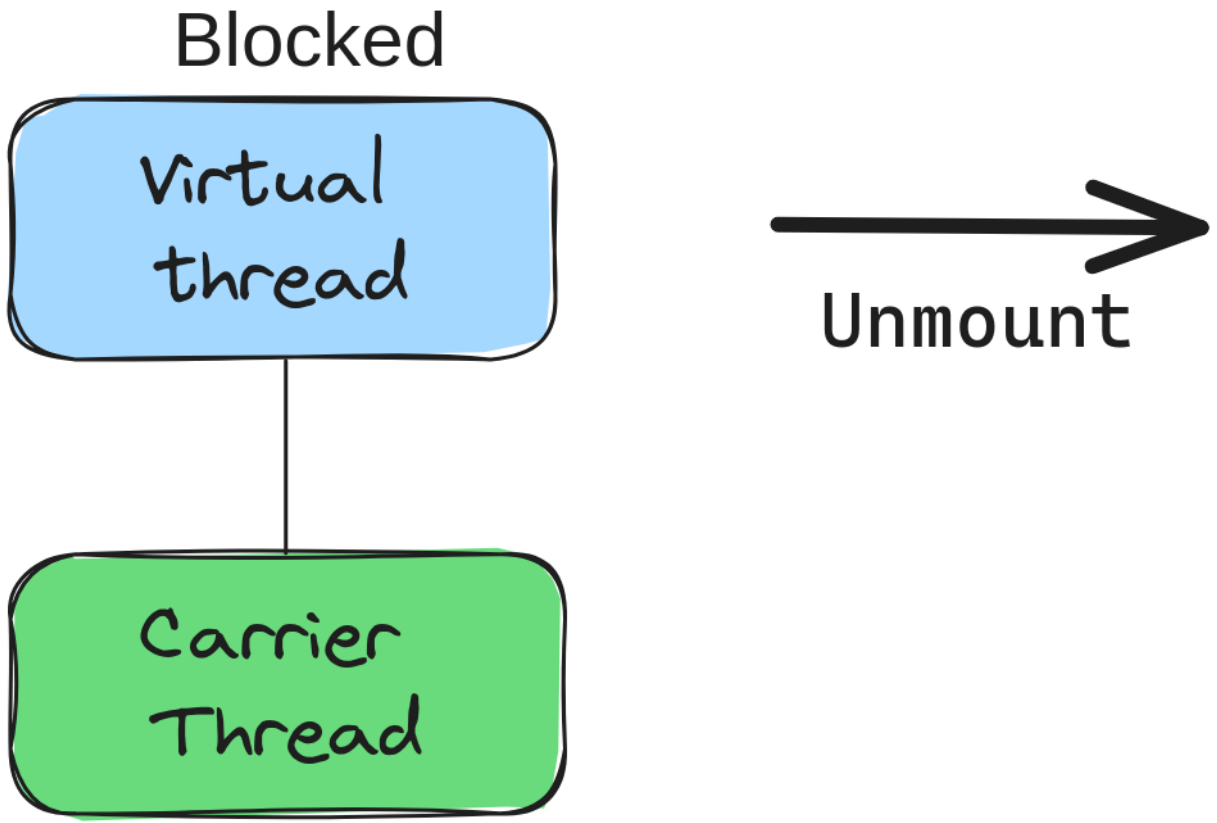
How does it work



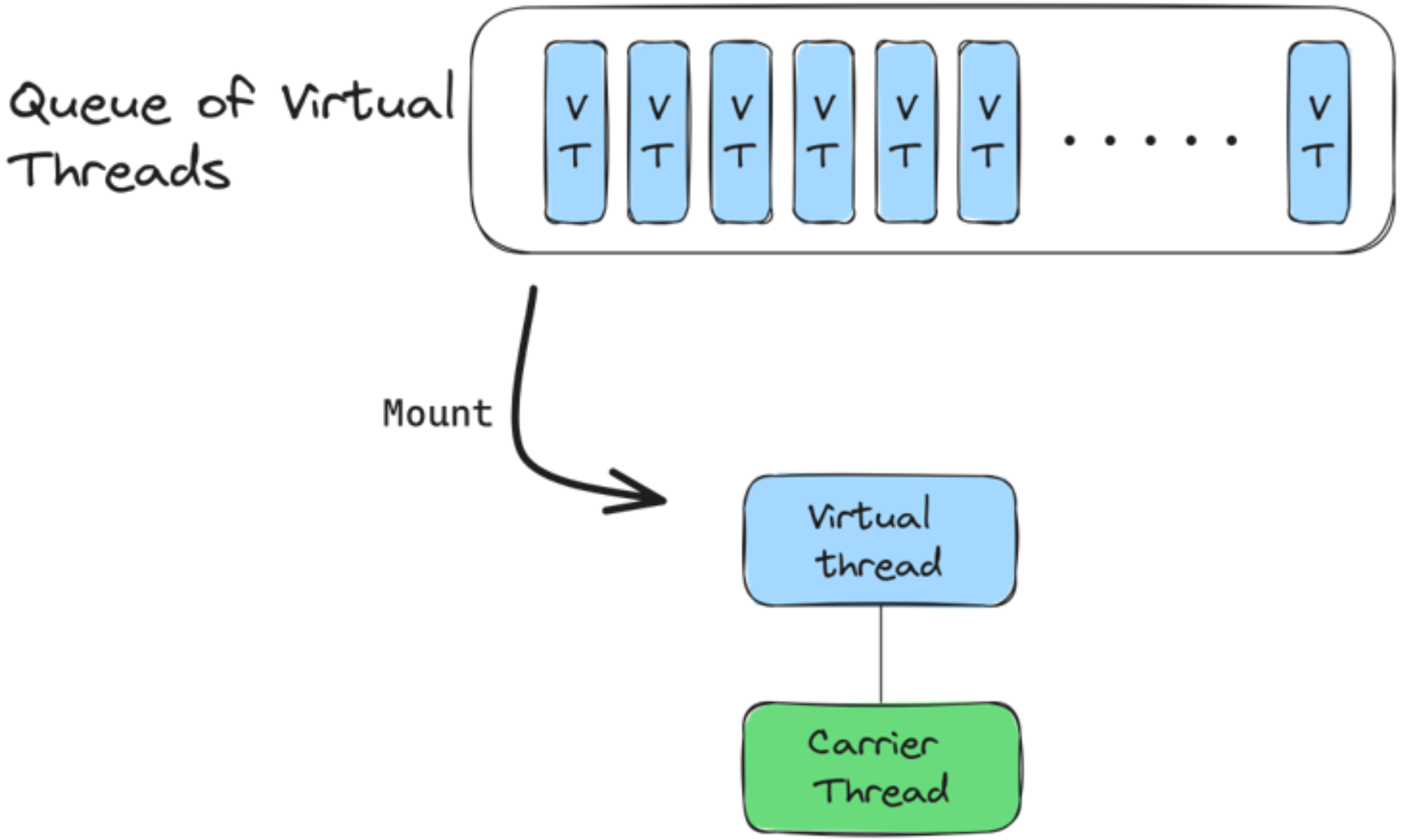
Mounting and unmounting



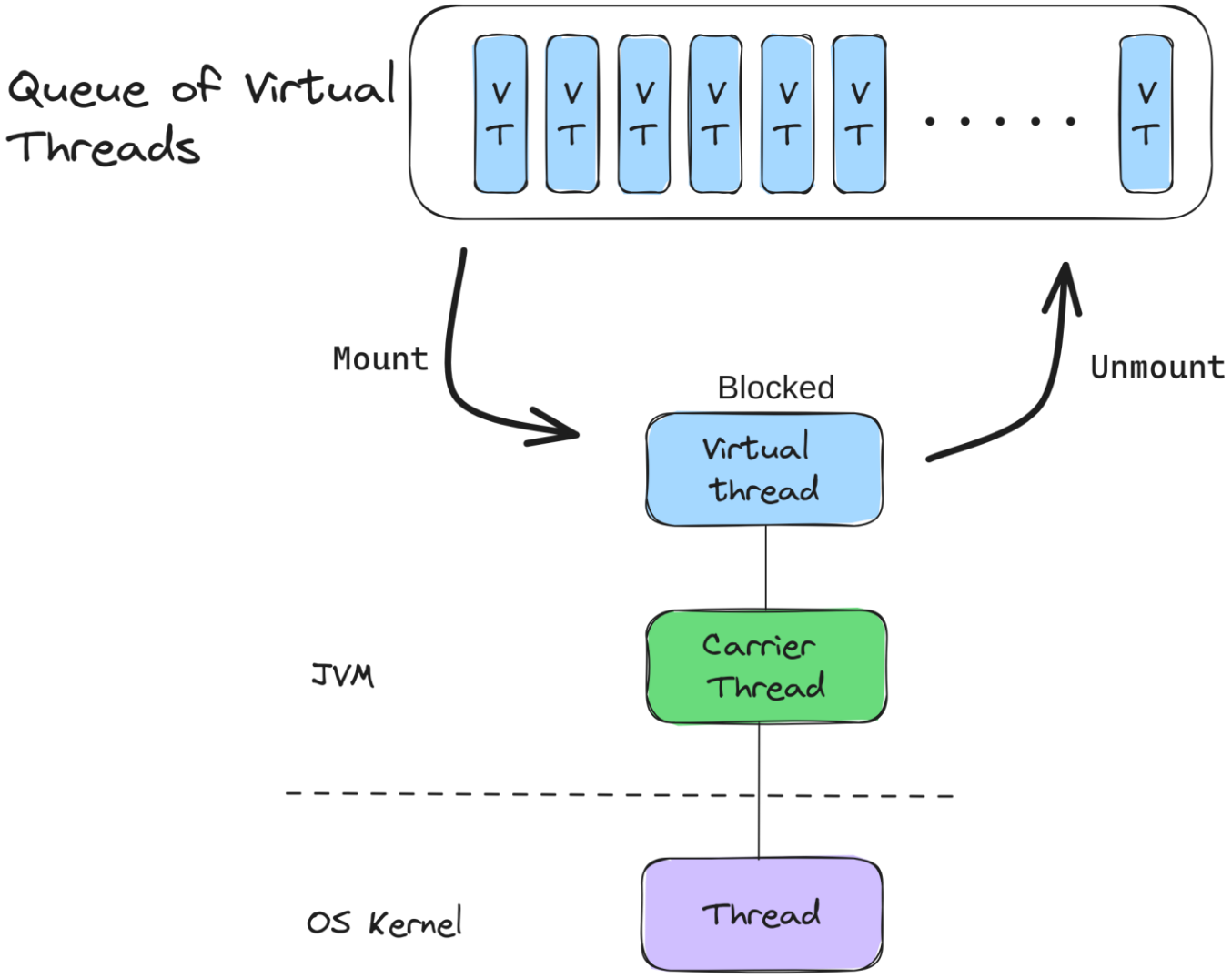
Mounting and unmounting



Mounting and unmounting

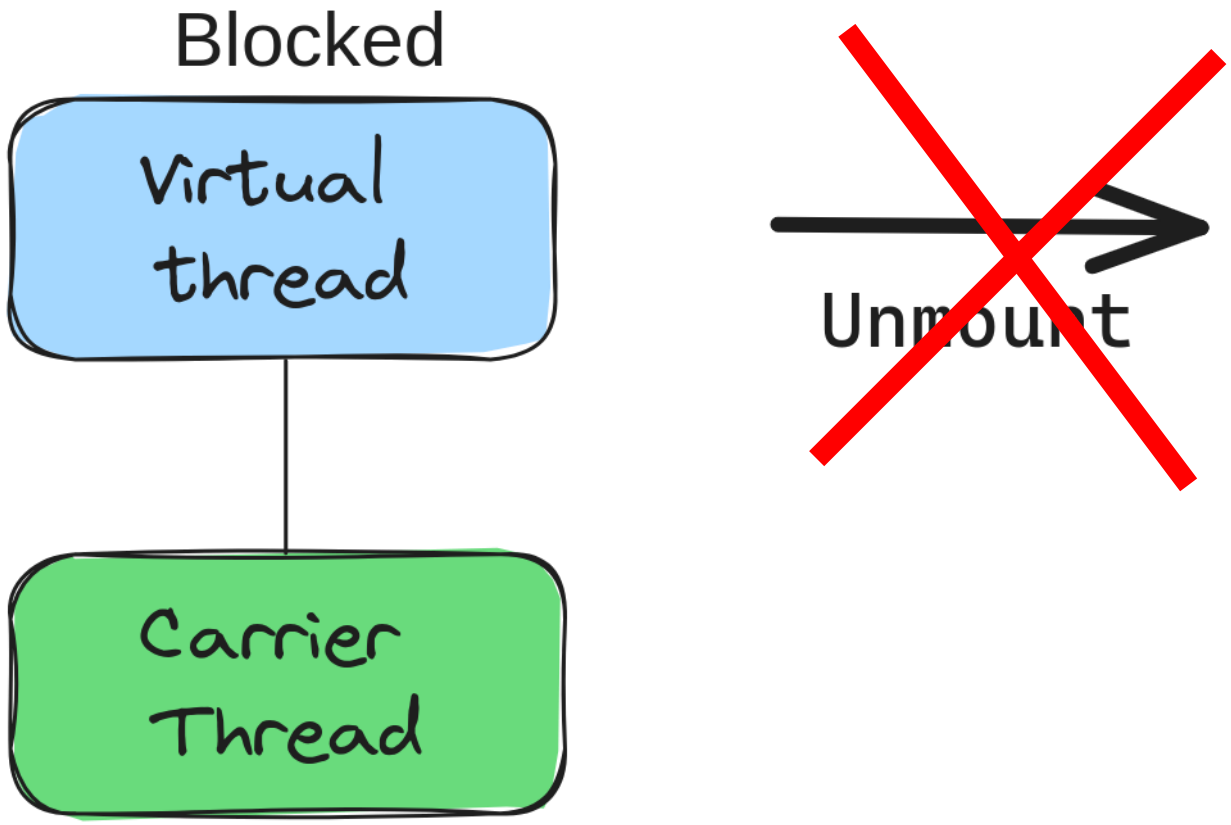


Mounting and unmounting



Time for step 4: Difference in threads

Pinning Virtual Threads



See pinned virtual threads

Run option:

-Djdk.tracePinnedThreads=short

System property:

jdk.tracePinnedThreads=full

Not every method is ready

- Synchronized Block
- Calls to native code (JNI)

Time for step 5: Find the pinned thread

Setting the number of carrier threads

- -Djdk.virtualThreadScheduler.parallelism=1
  Default of the number of carrier threads

- -Djdk.virtualThreadScheduler.maxPoolSize=1
  Maximum number of carrier threads

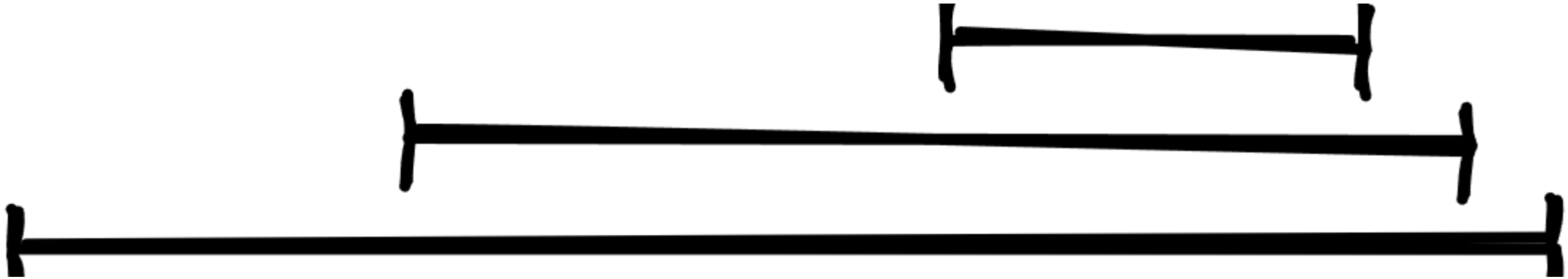
Time for step 6: Set the carrier threads

Time for step 7: Improve performance

(it’s a setup for later)


VIRTUAL THREADS
This is brilliant.
STRUCTURED CONCURRENCY
But I like this.
imgflip.com

Structured concurrency

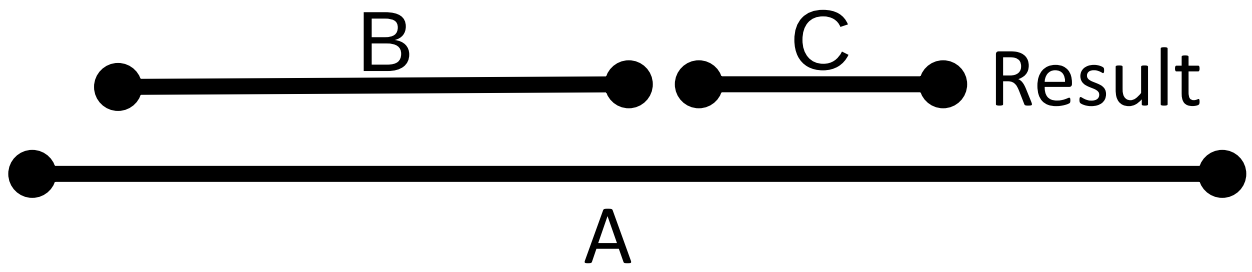


Structured Code

```
public String methodA(){

    String foo = methodB();
    String bar = methodC();

    return foo + bar;
}
```

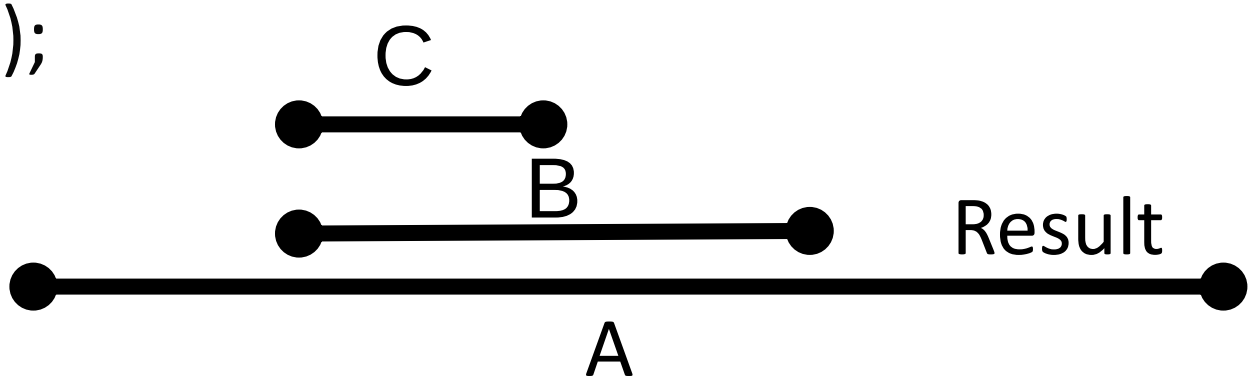
Structured Code

```
public String methodA(){

    String foo = methodB();
    String bar = methodC();

    return foo + bar;
}
```



Implicit relation

```
static public String methodA() throws … {
    Future<String> foo = methodB();
    Future<String> bar = methodC();

    return foo.get() + bar.get();
}
```



Structured concurrency

```
try (var scope = new StructuredTaskScope.ShutdownOnFailure()) {
        scope.fork(…)
}

try (… = new StructuredTaskScope.ShutdownOnSuccess<>()) {
        scope.fork(…)
}
```

Time for step 8 & 9: Structured concurrency

What is a scoped value

- A way to share data between threads
- Use less memory than thread locals
- Bounded context in which a value is known

Creating a Scope

```
class Scrape implements Runnable {
final static ScopedValue<HttpClient> CLIENT =
ScopedValue.newInstance();
….
}
```

Creating a Scope

```
ScopedValue.runWhere(Scrape.CLIENT,
                     VALUE,
                     new Scrape(queue, visited))
```

Time for step 9: Creating a scope

Scope inheritance

- Scoped values are inherited when using a StructuredTaskScope
  - When a scope is active all child threads will have the same scope

- Hint: first create a scope with a given value

Bonus 1: Scope inheritance

Scope rebinding

- It’s a child scope with the same key as the parent scope

- Scope{ key1, value1…
  - Scope{ key1, value2…
    - Scope{key1, value3…

Bonus 2: Rebinding scopes

What does the taskscope do

- Virtual threads behave as one unit of work
- Passes down scope values
- It stops Threads when a criteria is met

Creating your own taskscope

1. Extends the StructuredTaskScope<StartingPoint>

2. Override the handleComplete method

4. Check if the Status if the subtask is Subtask.State.SUCCESS

3. Call shutdown(); when you have the result you want

Bonus 3: Creating your own task scope

Moving logic out of the scope class

- Freestyle assignment
  - You implement it any you want
- It’s about improving the readability of your code
- One suggestion: Try out the Predicate Functional interface

Bonus 4: Moving business logic

Deadlines

- It stops virtual threads when they take to long
- Creating a deadline:

.joinUntil();

- What happens if a thread does not respond to interrupts?
  Maybe its stuck in a while(true)

Bonus 5: Deadlines

Virtual threads with existing executors

- Virtual threads have their own factory
  - Thread.ofVirtual().factory();

- Use this factory so other executors use virtual threads instead of platform threads

Bonus 6: Virtual threads with executors

Limiting the number of requests without pools

- Freestyle challenge to limit the number of requests made.
- Suggestion: A Semaphore is one way of solving this

Side note: You cannot create a pool of virtual thread, or pin a thread in any way

Bonus 7: Limit the number of requests

Someone want to show their Scraper, or freestyle assignment?