Fibers

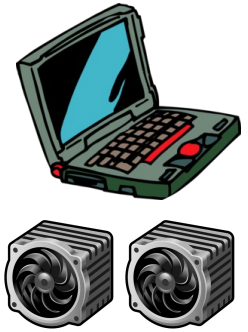
Green threads
Light weight threads
Coroutines

in Java Virtual Machine

Volkan Yazıcı

http://vlkan.com/

Multitasking



You have two CPUs.
You can read and watch **simultaneously**.

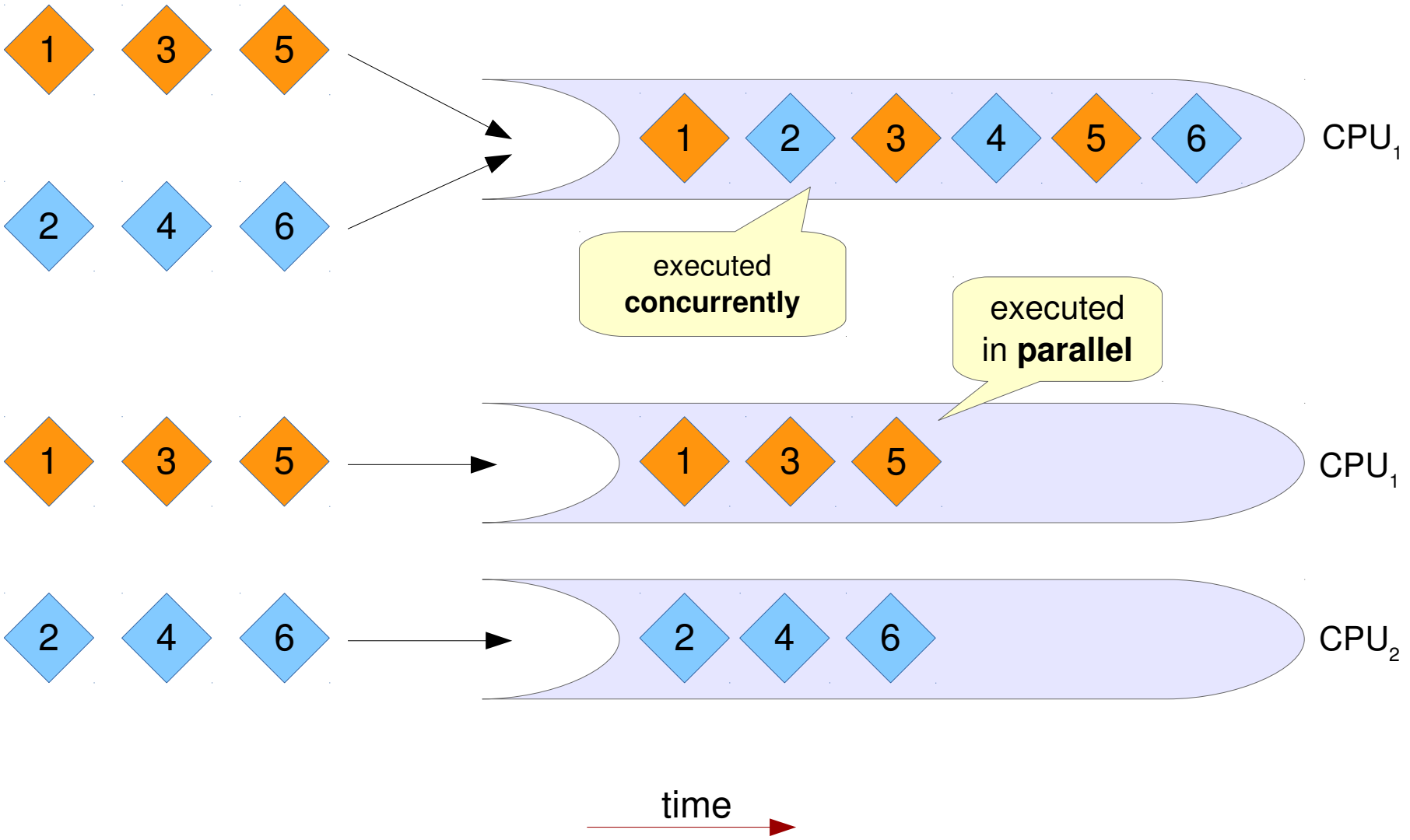
You have **one CPU**.
You can **still** read and watch simultaneously.

How?

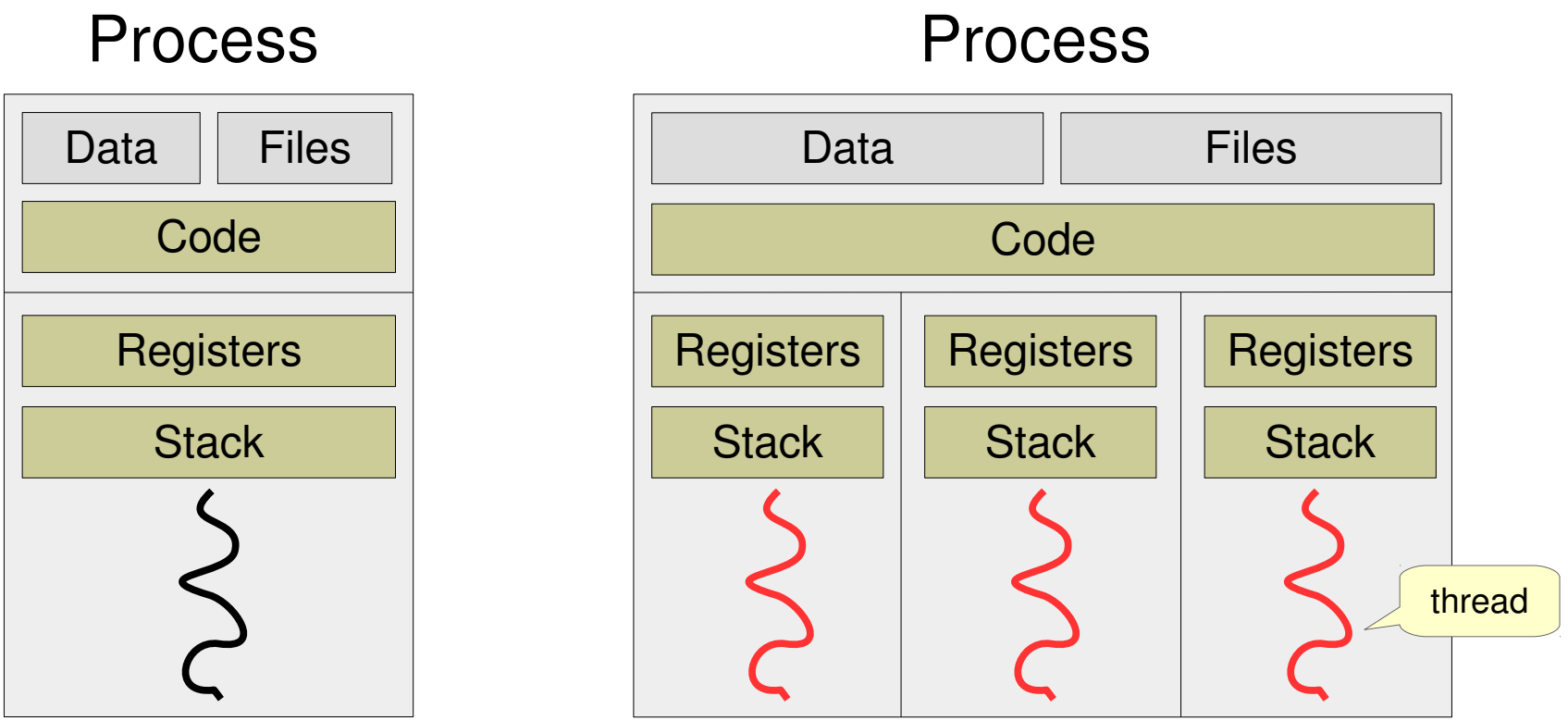
Via multitasking!

But how?

Parallelism vs. Concurrency



What is an OS process and thread?



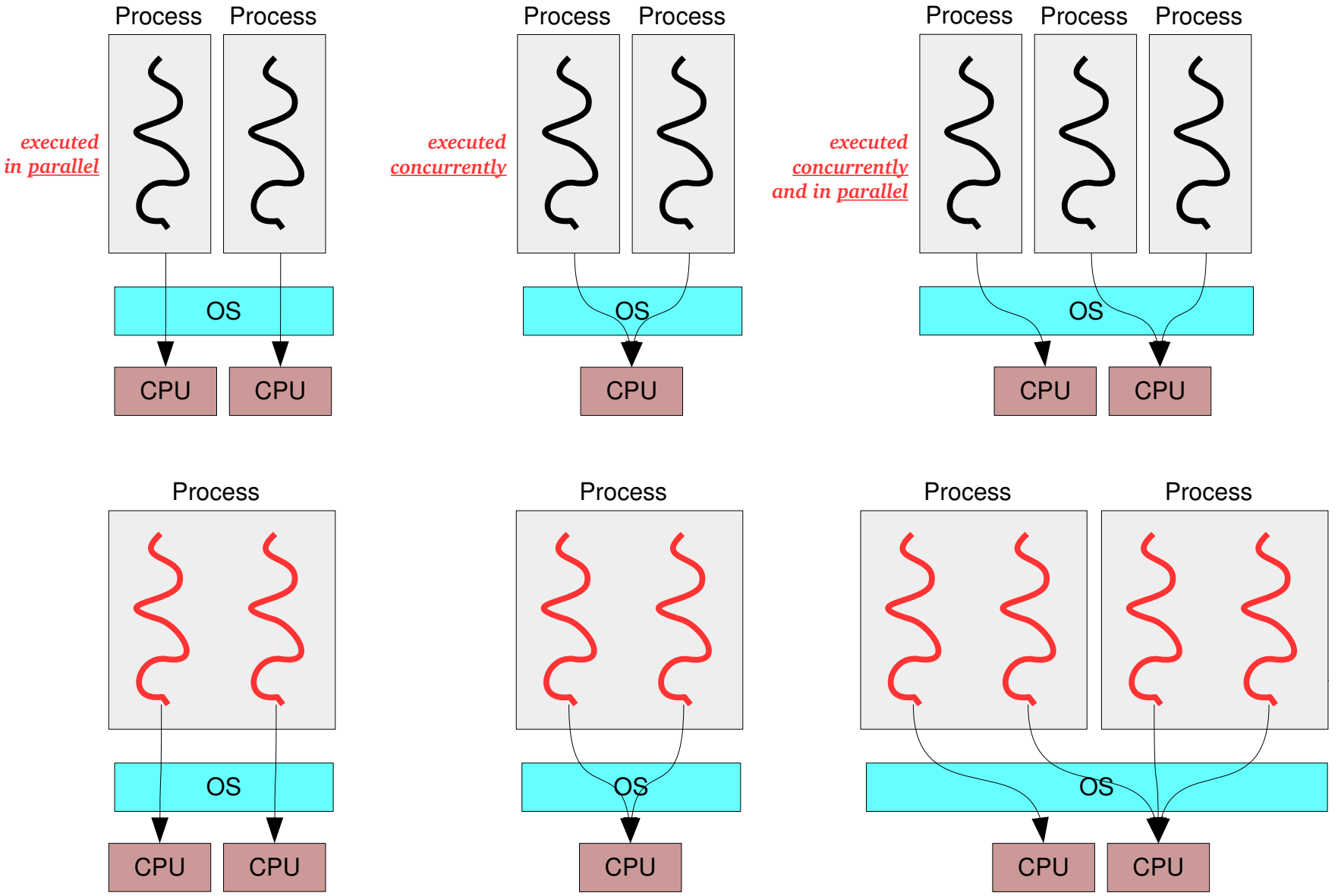
Threads share the same address space

→ consumes less memory

→ spawning is cheaper

→ context switching is cheaper

How does OS map processes/threads to CPUs?

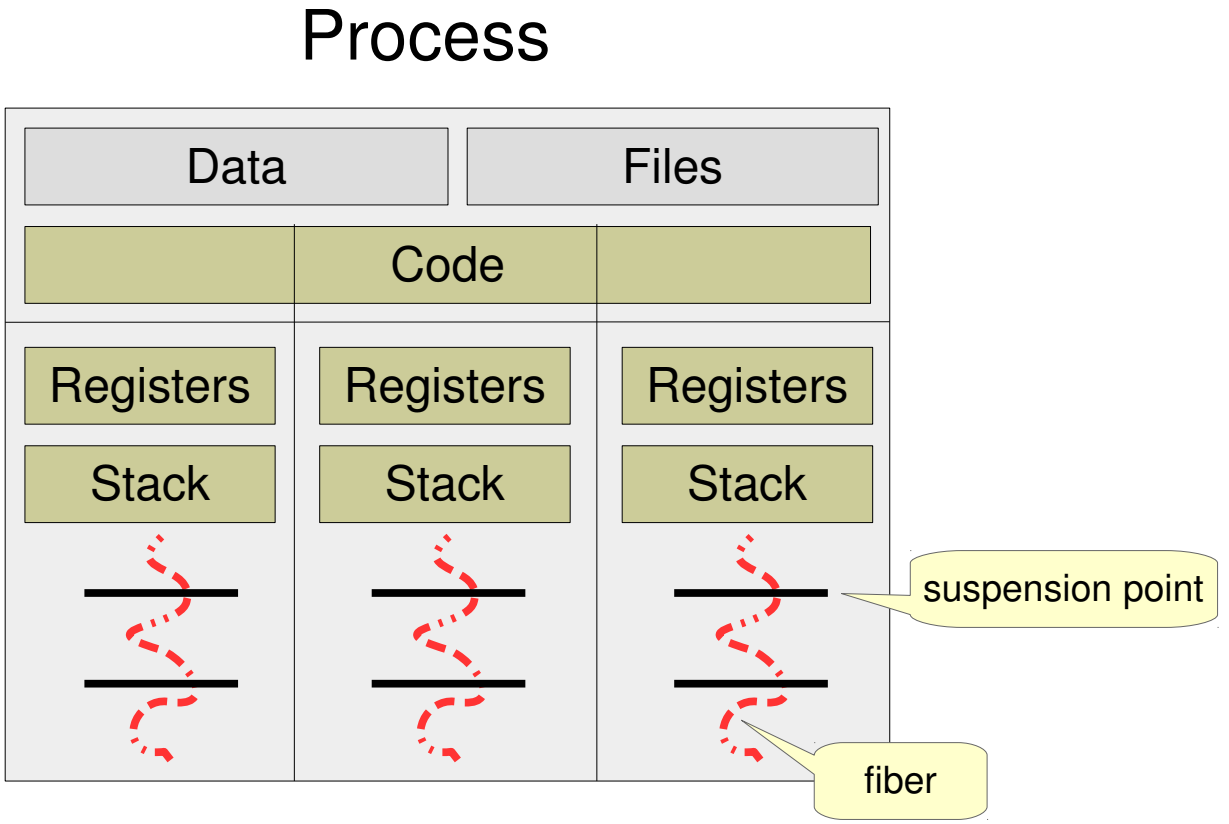


What is a JVM process and thread?

≤1.1 releases were using green threads

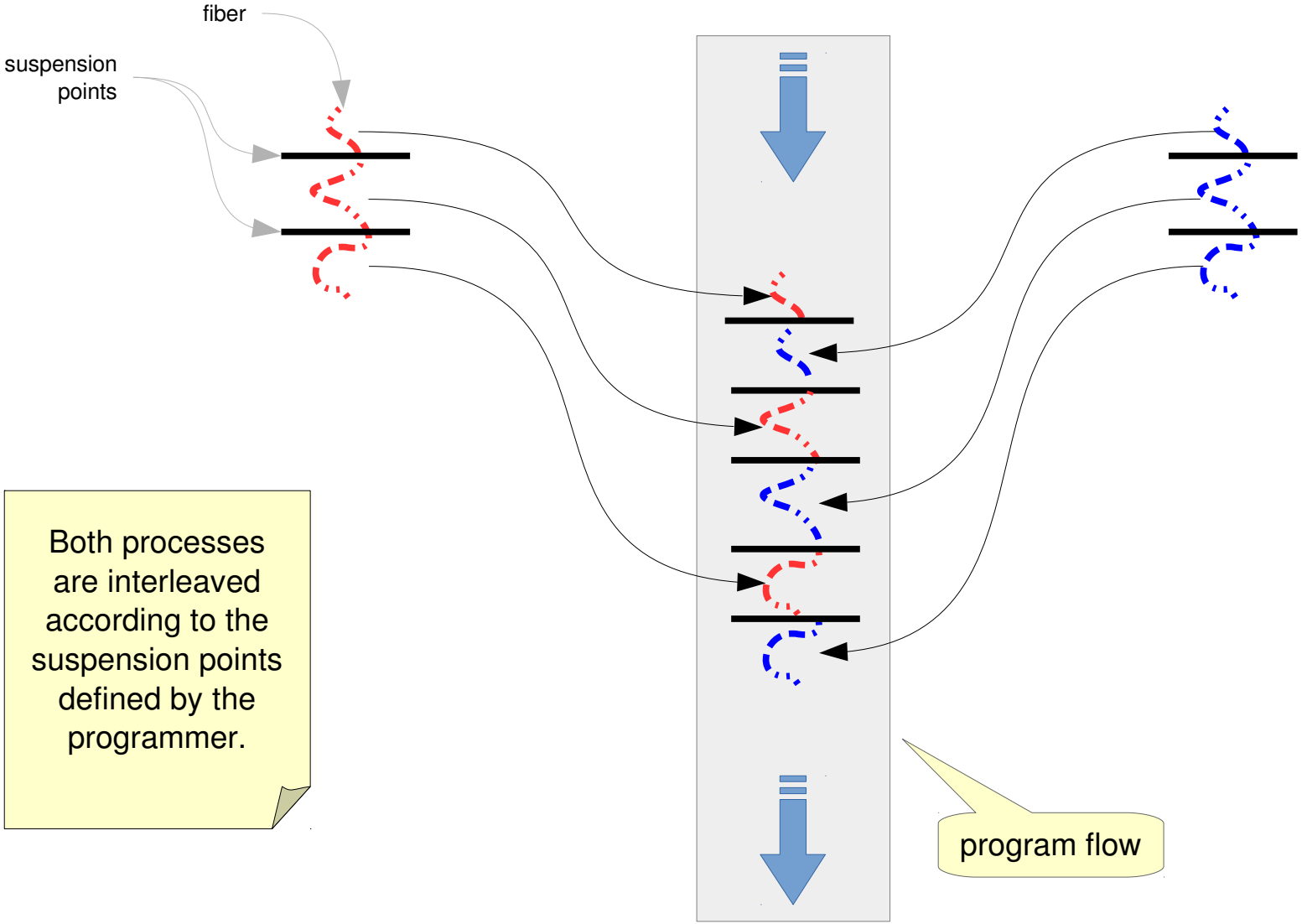
>1.1 releases maps to native OS processes and threads

What is a fiber?



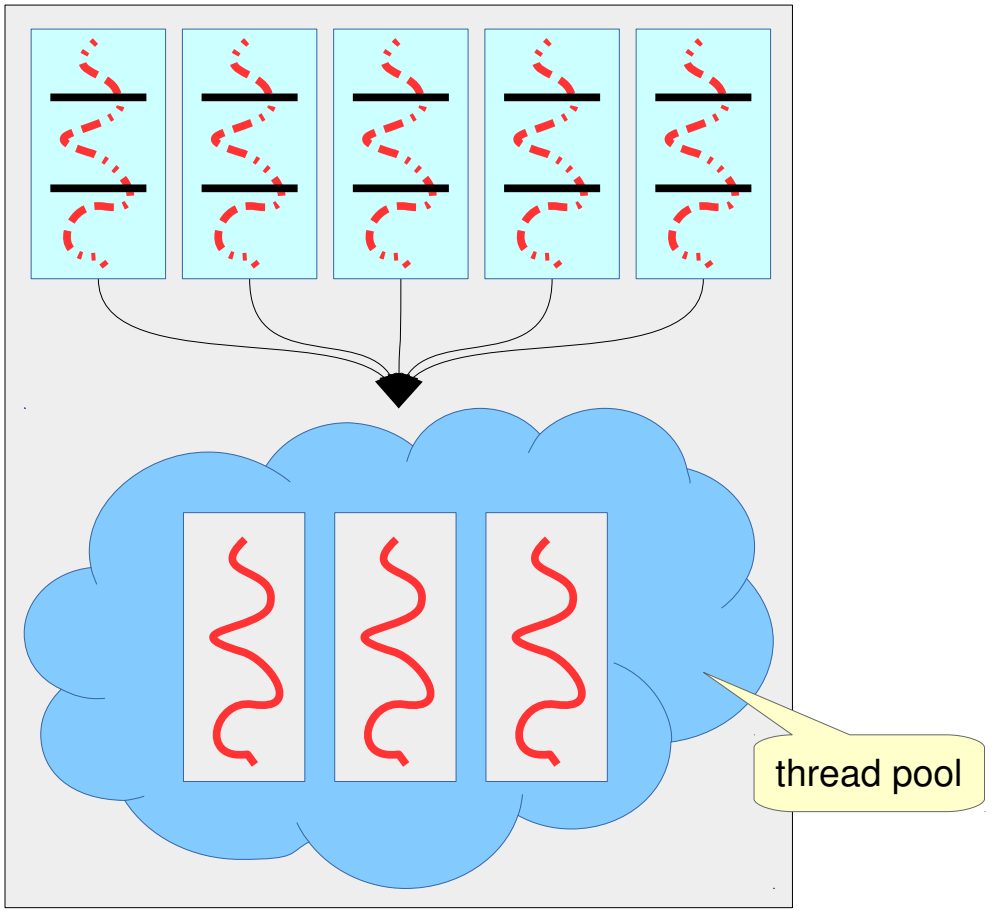
→ preemption entry points are set by the programmer
→ scheduling is performed ***cooperatively*** at the user-level

How do fibers work?



Both processes are interleaved according to the suspension points defined by the programmer.

How are fibers implemented?



When to use fibers?

Fibers are not meant to replace threads.

Fibers should be used ***when its body blocks very often*** waiting on other fibers
→ waiting for messages sent by other fibers on a channel
→ or waiting for the value of a dataflow-variable.

For long-running computations that rarely block, traditional threads are preferable.

Fibers and threads interoperate very well.

Fibers are specially useful ***for replacing callback-ridden asynchronous code***.

Quasar

http://docs.paralleluniverse.co/quasar/

Quasar is a Java library that provides
→ high-performance **lightweight threads**,
→ Go-like **channels**,
→ Erlang-like **actors**,
→ and other asynchronous programming tools.

Quasar fibers rely on bytecode instrumentation.

This can be done
→ at classloading time via a Java Agent,
→ or at compilation time. (Ant/Maven plugins are available.)

Running Quasar Code

dependency:

```
<dependency>
    <groupId>com.vlkan</groupId>
    <artifactId>quasar-maven-plugin</artifactId>
    <version>0.6.2</version>
</dependency>
```

plugin:

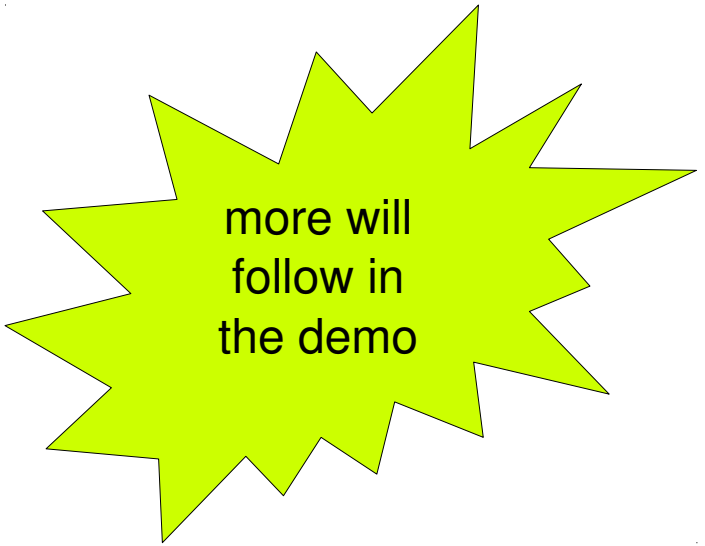
```
<plugin>
    <groupId>com.vlkan</groupId>
    <artifactId>quasar-maven-plugin</artifactId>
    <version>0.6.2</version>
    <configuration>
        <check>true</check>
        <debug>true</debug>
        <verbose>true</verbose>
    </configuration>
    <executions>
        <execution>
            <goals>
                <goal>instrument</goal>
            </goals>
        </execution>
    </executions>
</plugin>
```

or via Java Agent: `java -javaagent:/path/to/quasar-core.jar ...`

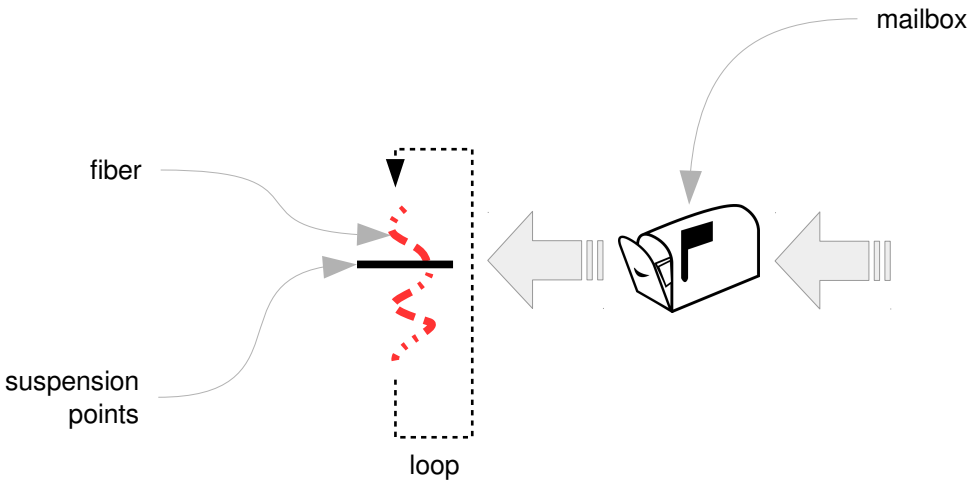
Sample Quasar Fiber

```
Fiber<V> fiber = new Fiber<V>() {

    @Override
    protected V run() throws SuspendExecution, InterruptedException {
        // your code
    }

}.start();

// do other stuff

fiber.join();

V value = fiber.get();
```

Marks a suspension point.



What is an actor?



Performance Benchmark: Thread-Ring

http://vlkan.com/blog/post/2014/09/01/java-fiber-test/

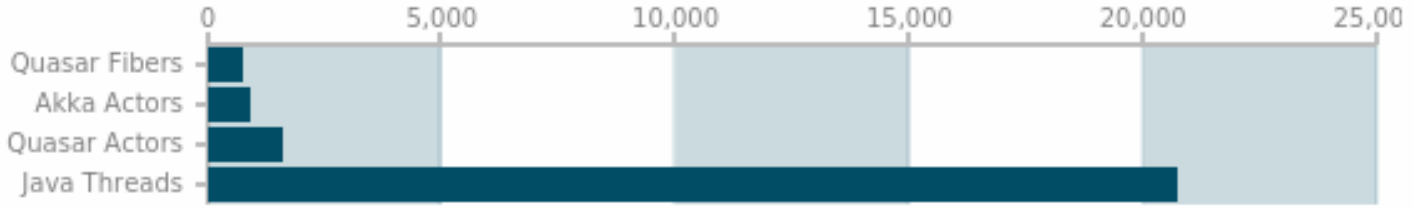
Thread-ring benchmark:

http://www.sics.se/~joe/ericsson/du98024.html

n threads are spawned and connected as a ring structure. Through this ring a message – an integer comprising 4 bytes – is circulated involving ***m*** message passings.

→ Ubuntu GNU/Linux 14.04 (LTS)
→ 6 physical core Intel(R) Xeon(R) E5645 2.40GHz
→ 64-bit Java HotSpot VM (build 1.8.0_20-ea-b05)
→ JMH: 5+5 warmup rounds and 3 JVM forks
→ 503 threads
→ 1e7 message passings
→ Results are in milliseconds

Fiber Impl.	Min.		Avg.	Max.	Stddev.	Confidence Interval		
Quasar Fibers	695.613	29x	721.457	758.231	20.796	99.9%	699.226	743.689
Akka Actors	856.807	23x	911.295	963.403	34.345	99.9%	874.578	948.012
Quasar Actors	1553.224	13x	1606.718	1660.329	35.756	99.9%	1568.492	1644.943
Java Threads	15730.028	1x	20709.233	33084.117	4338.423	99.9%	16071.196	25347.270



JVM Fiber Implementations

Akka

http://akka.io/

Kilim

http://www.malhar.net/sriram/kilim/

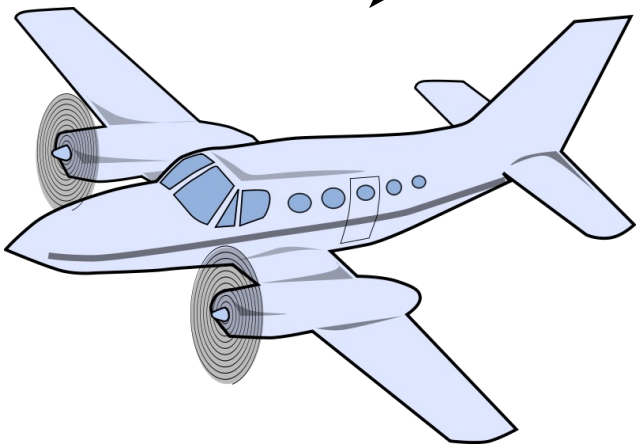
Orbit

http://orbit.bioware.com/

Quasar

http://www.paralleluniverse.co/quasar/

Quasar vs. Akka



Take Away

Multitasking Model	Scheduled by	Memory Model
Process	Kernel	Unshared
Thread	Kernel	Shared
Fiber	Kernel + User	Shared

Questions?