Introduction aux Threads Virtuels en SpringBoot 3 à travers une démonstration

BreizhCamp 2024 – Emmanuelle Rouillé

*Le monde est tel que nous le façonnons

The world is how we shape it*

Breizh C@mp
La conférence à l'Ouest

sopra steria

Emmanuelle Rouillé

Architecte

Intervenante

Architecte chez Sopra Steria

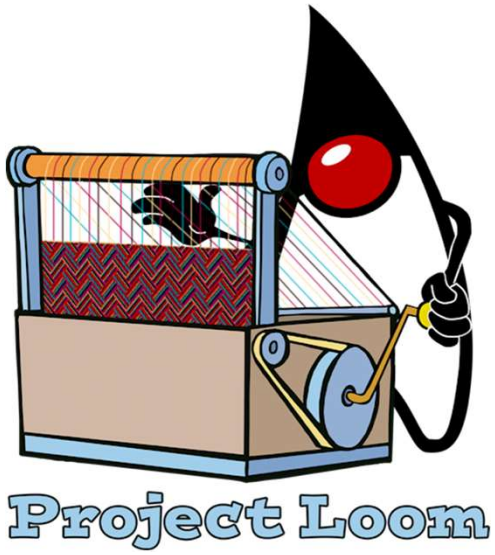
- Technologies principales : Spring/Java, Cloud
- Secteur Télécom

Introduction

Objectif du quickie

- Faire découvrir les threads virtuels et le support Spring Boot 3 à travers une démonstration

- Threads virtuels
  - Une des features principales du projet Loom
  - JEP 444 (https://openjdk.org/jeps/444)
  - Release JDK 21, first preview JDK 19, second preview JDK 20



Pour aller plus loin

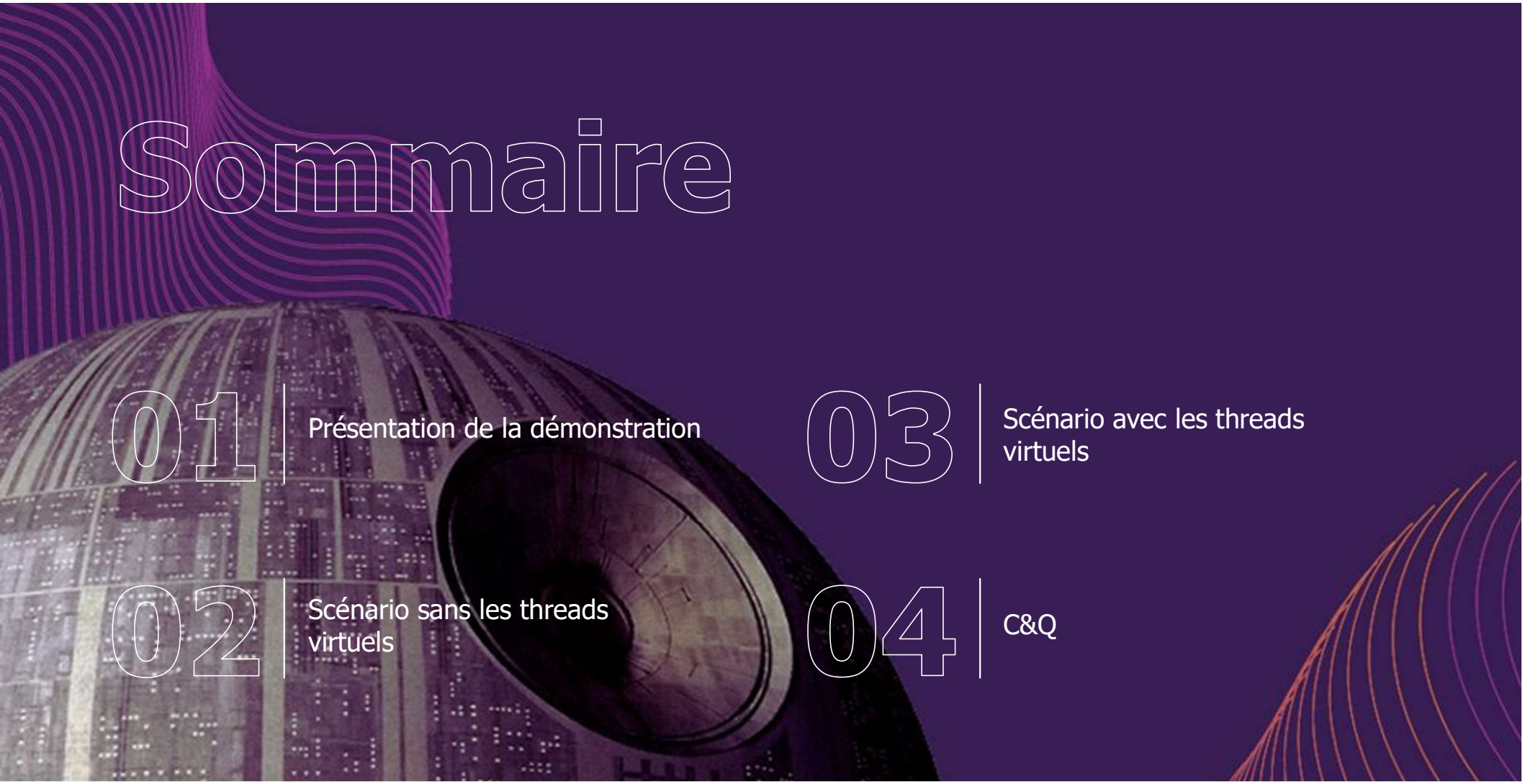
- José Paumard, Rémi Forax
Devoxx 2022
Loom nous protègera-t-il du braquage temporel ?

- José Paumard
Devoxx 2023
Programmation concurrente et asynchrone: Loom en Java 20 et 21

- Daria Hervieux
BreizhCamp 2023
Spring et Virtual Threads : est-il possible d'optimiser les performances de votre application ?

- Thomas Piscitelli, Alexandre Thomazo
BreizhCamp 2023
LOOM, la météorite sur la JVM ?

Sommaire

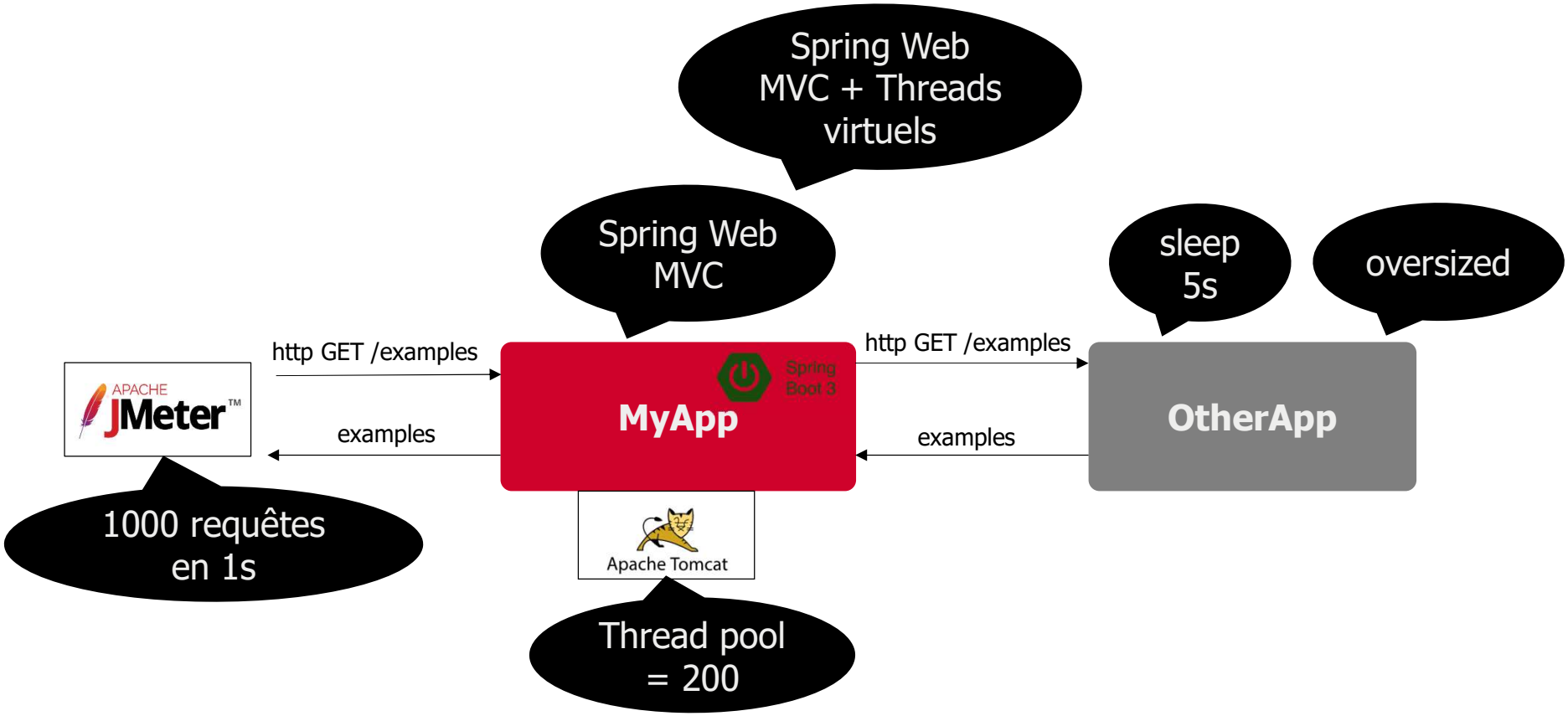


Breizh C@mp
La conférence à l'Ouest

sopra steria

Présentation de la démonstration

Présentation de la démonstration

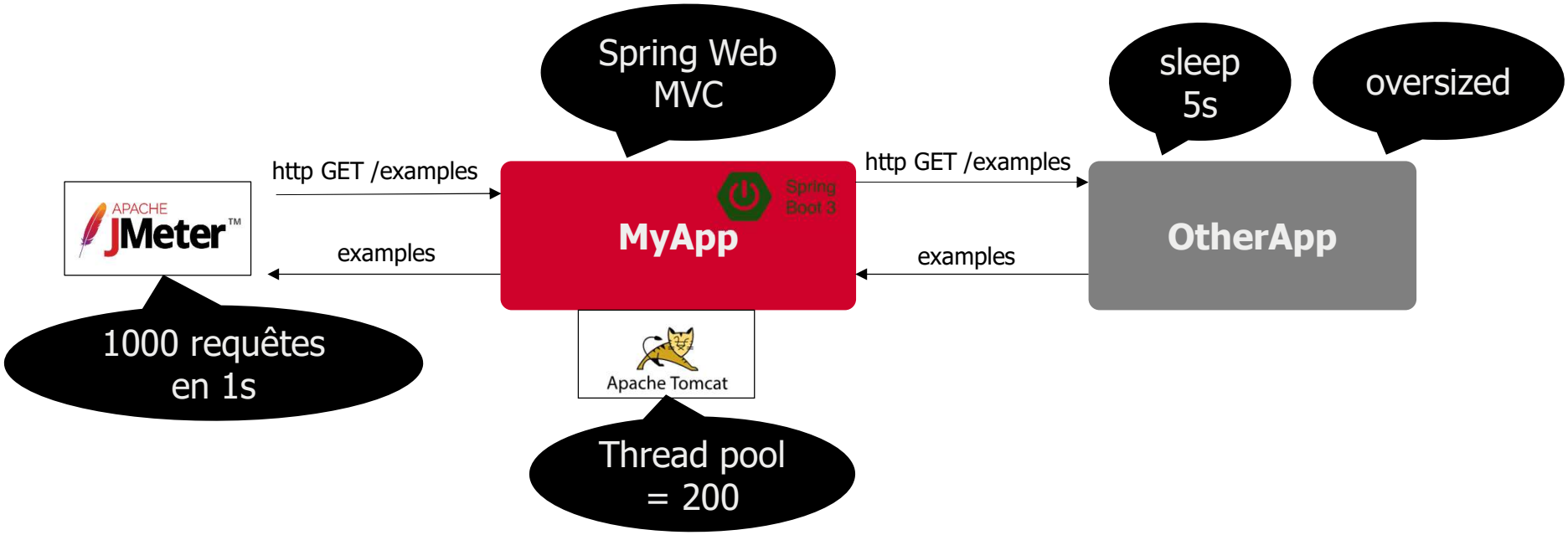


Scénario sans les threads virtuels

Sans les threads virtuels

Démonstration

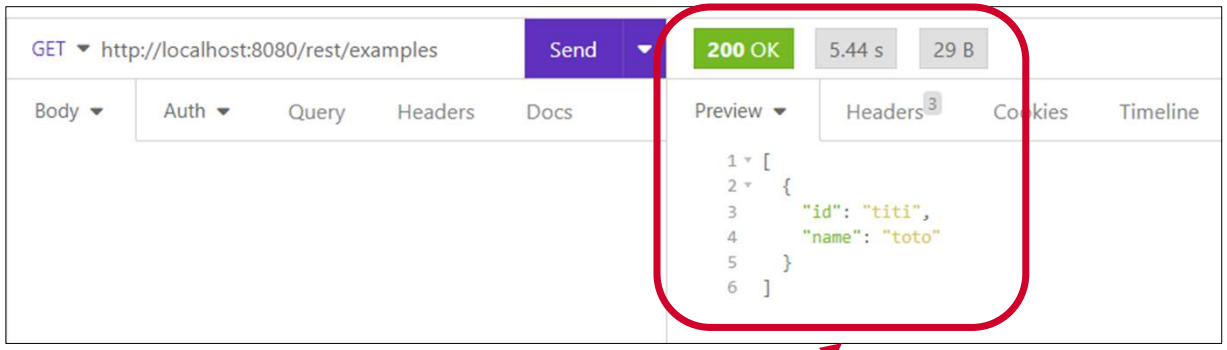
- Rappel du scénario



Sans les threads virtuels

Démonstration

- Comportement observé pour 1 requête



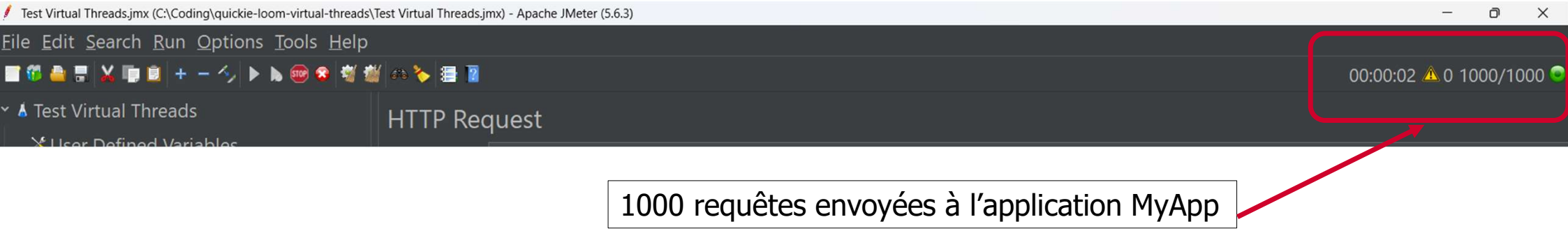
Liste d'exemples retournée au bout de ~5s

- Possibilité de vérifier dans les logs applicatifs le type de threads utilisés
  - Ici des threads plateforme : `Thread[#50,http-nio-8080-exec-1,5,main]`

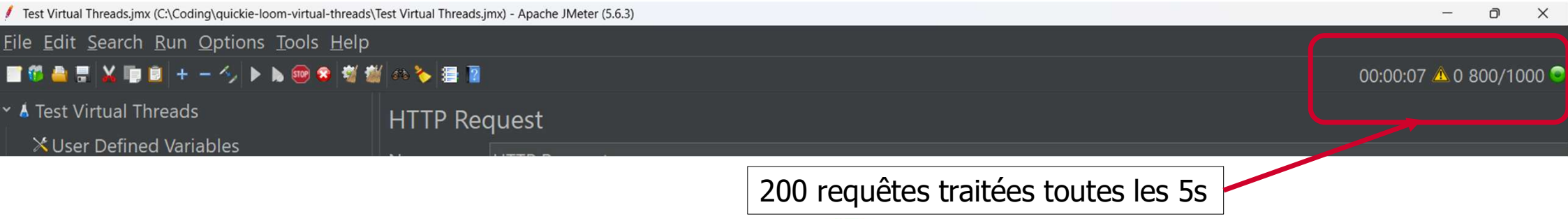
Sans les threads virtuels

Démonstration

- Comportement observé lors du test de charge



1000 requêtes envoyées à l'application MyApp

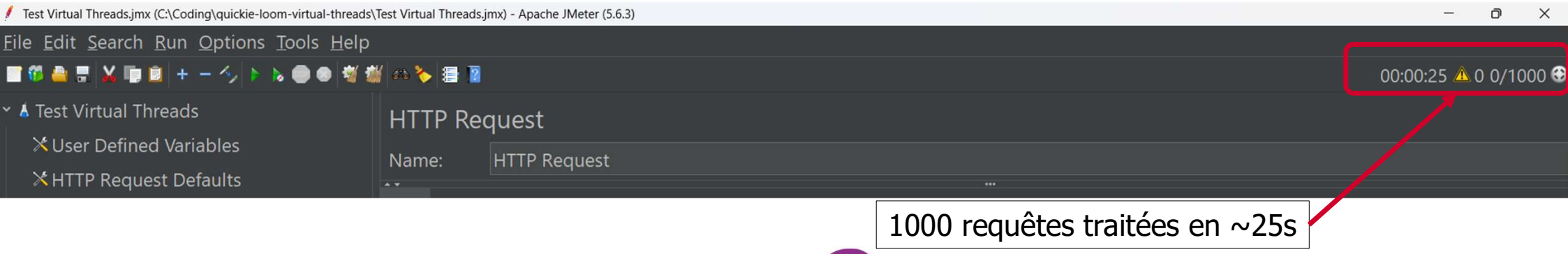


200 requêtes traitées toutes les 5s

Sans les threads virtuels

Démonstration

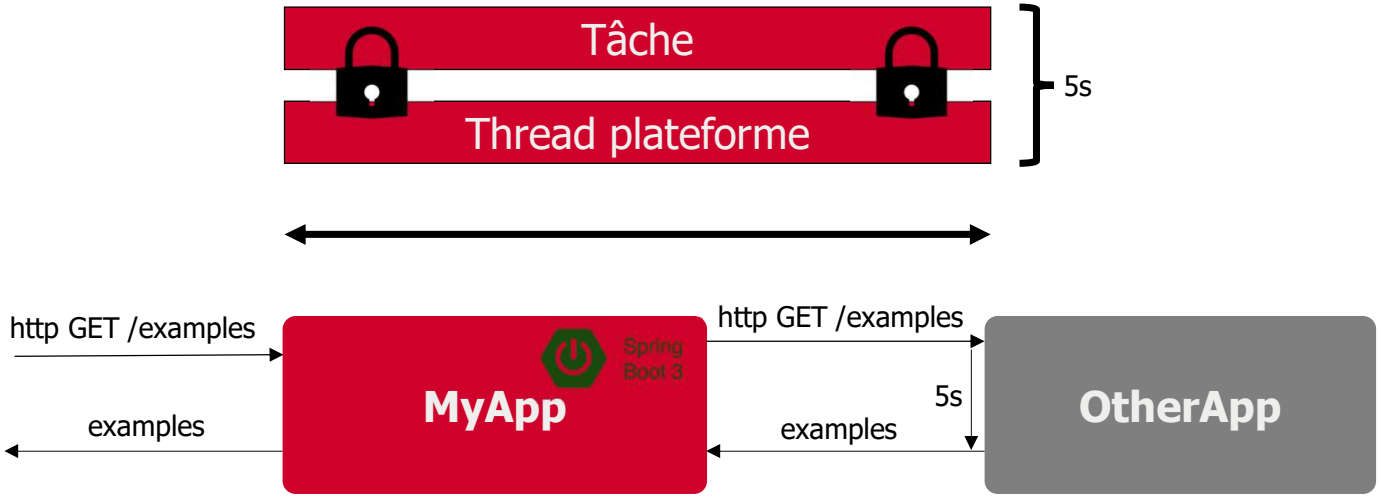
- Comportement observé lors du test de charge



Sans les threads virtuels

Explication

- Exécution synchrone bloquante
- Couplage fort entre une tâche et un thread plateforme



Scénario avec les threads virtuels

Avec les threads virtuels

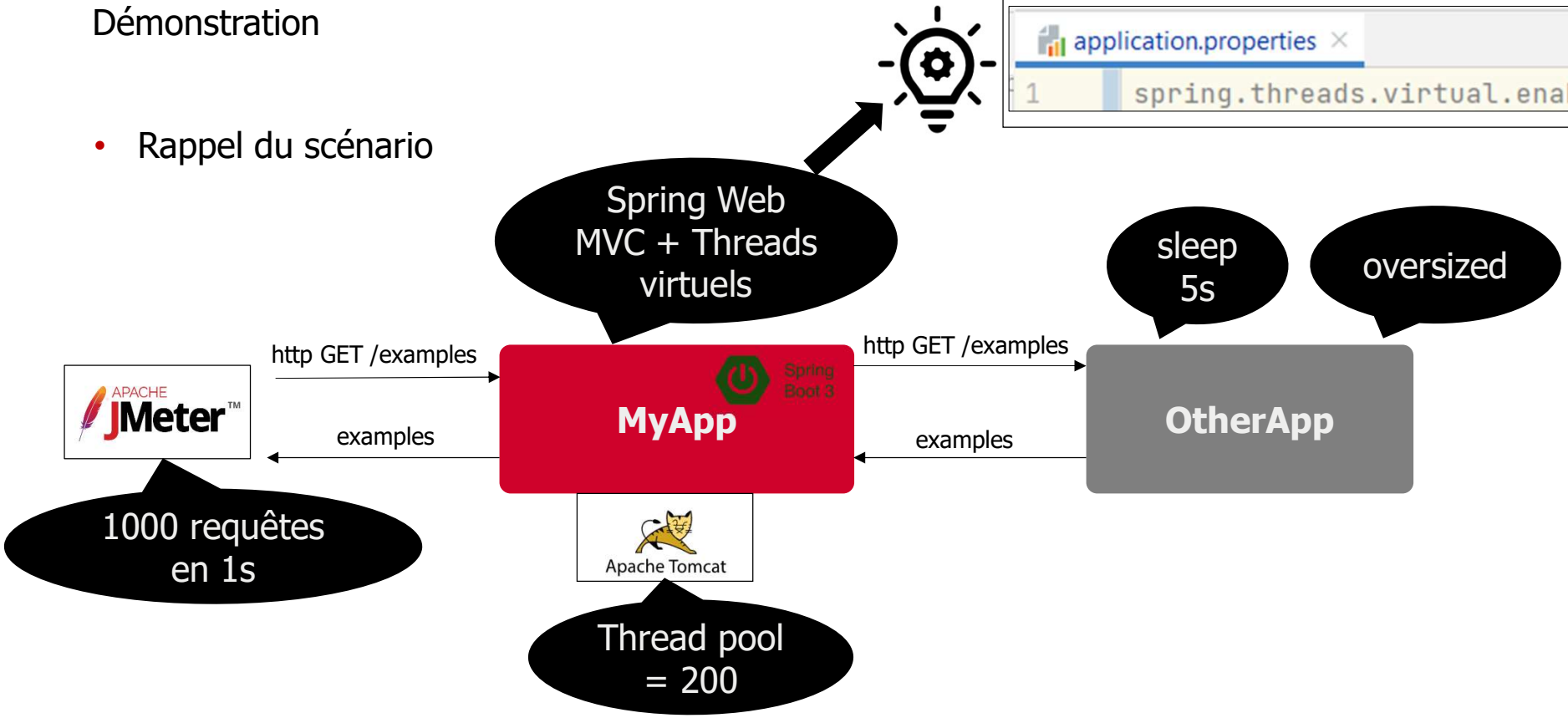
Démonstration

- Rappel du scénario

Même application que pour la programmation synchrone, juste une propriété Spring à changer :

application.properties

```
1  spring.threads.virtual.enabled=true
```



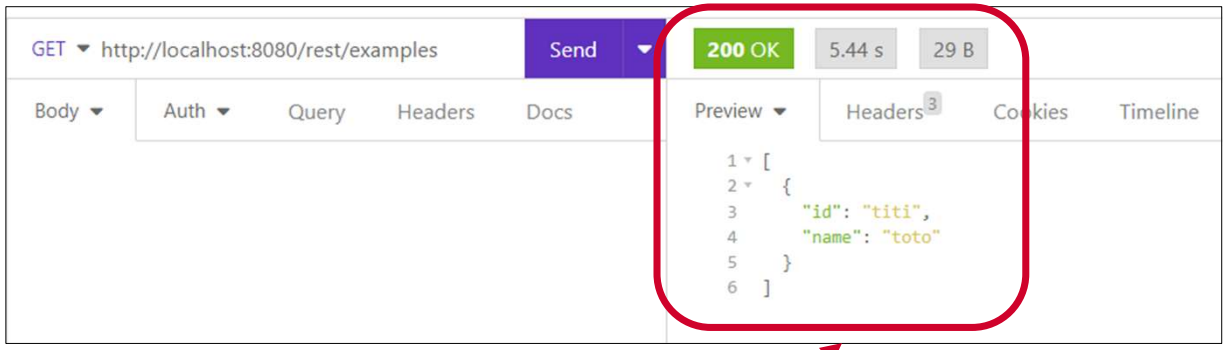
Breizh C@mp La conférence à l'Ouest

sopra steria

Avec les threads virtuels

Démonstration

- Comportement observé pour 1 requête



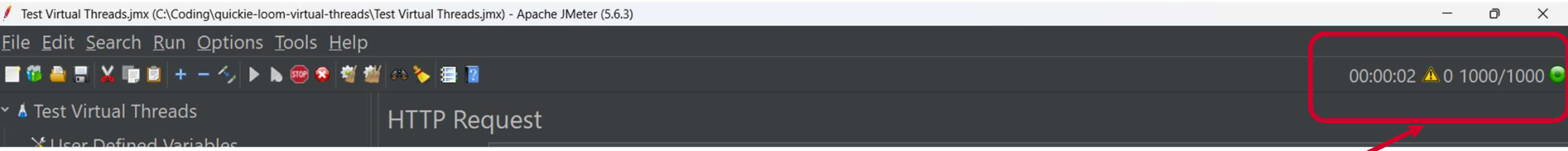
Liste d'exemples retournée au bout de ~5s

- Possibilité de vérifier dans les logs applicatifs le type de threads utilisés
  - Ici des threads virtuels : `VirtualThread[#2894,tomcat-handler-979]/runnable@ForkJoinPool-1-worker-4`

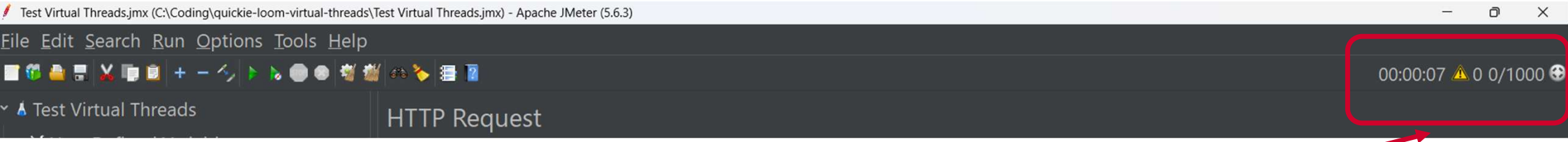
Avec les threads virtuels

Démonstration

- Comportement observé lors du test de charge



1000 requêtes envoyées à l'application MyApp



1000 requêtes traitées en ~7s

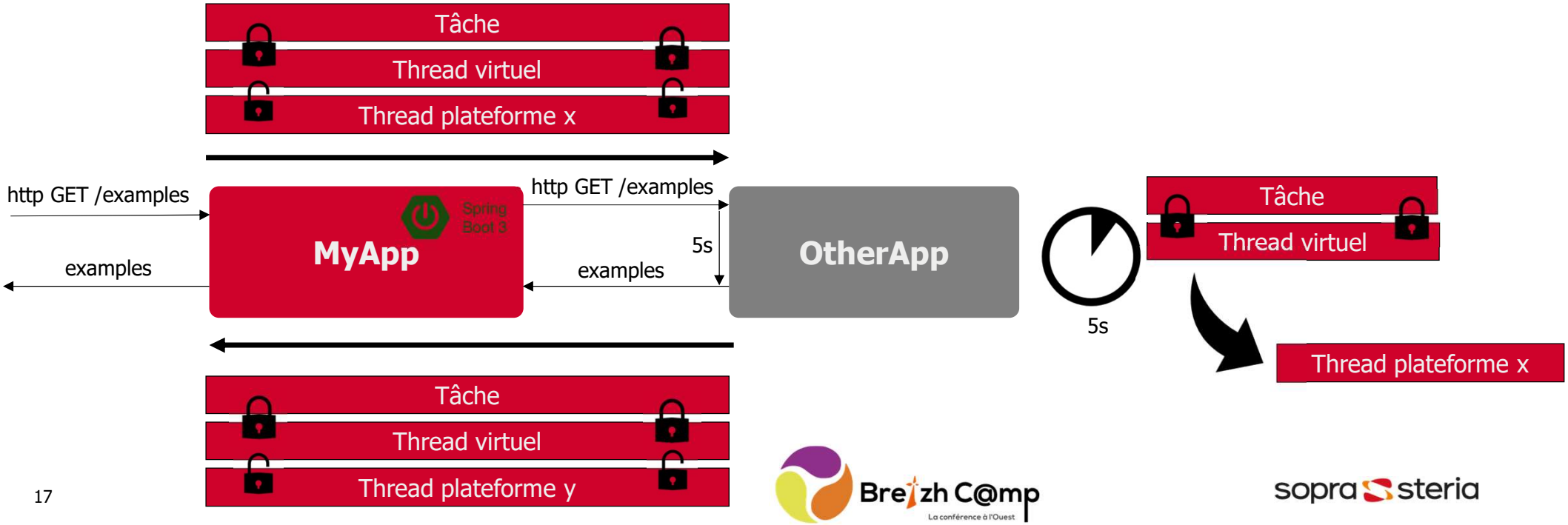
Breizh C@mp La conférence à l'Ouest

sopra steria

Avec les threads virtuels

Explication

- Exécution asynchrone non bloquante
- Couplage faible entre un thread virtuel et un thread plateforme



C&Q

Breizh C@mp
La conférence à l'Ouest

sopra steria

Conclusion

Les threads virtuels

- Les avantages de l'exécution asynchrone non bloquante sans changer le code

- Bonnes pratiques
  - Pas de gestion de pools de threads pour les threads virtuels
  - Moins performant que des threads plateforme pour les tâches *CPU intensives*
  - *Semaphores* pour éviter de « bombarder » les applications externes
  - Limitation sur les blocks *synchronized* (utiliser *reentrant lock*) et certaines librairies utilisant du code natif
  - …
- Documentation Java
  - https://docs.oracle.com/en/java/javase/21/core/virtual-threads.html#GUID-2BCFC2DD-7D84-4B0C-9222-97F9C7C6C521

Conclusion

Le support SpringBoot 3

- À partir de la version 3.2.3
  - Documentation
    - https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/

7.8. Task Execution and Scheduling

In the absence of an `Executor` bean in the context, Spring Boot auto-configures an `AsyncTaskExecutor`. When virtual threads are enabled (using Java 21+ and `spring.threads.virtual.enabled` set to `true`) this will be a `SimpleAsyncTaskExecutor` that uses virtual threads. Otherwise, it will be a `ThreadPoolTaskExecutor` with sensible defaults. In either case, the auto-configured executor will be automatically used for:

- asynchronous task execution (`@EnableAsync`)
- Spring for GraphQL's asynchronous handling of `Callable` return values from controller methods
- Spring MVC's asynchronous request processing
- Spring WebFlux's blocking execution support

Feedbacks

Merci

emmanuelle.rouille@soprasteria.com

https://github.com/emmanuellerouille/virtual-threads

*Le monde est tel que nous le façonnons

The world is how we shape it*

Breizh C@mp
La conférence à l'Ouest

sopra steria