Abdessamad Tanafaat G.INFO 3

Rendu : TP N° 6: Tests de performance : (Synchronous programming :Platform threads vs. Virtual threads)

Activité 1 : contexte de programmation synchrone :

- Lien Git vers le projet : https://github.com/abdessamadtanafaat/ThreadComparisonSpringBoot

Activité 2 : l'utilisation de l'outil de test de performance « Apache Jmeter» :

platform threads : Cas classique				
Débit moy	Temps de réponse moy	% Erreur	Kb/sec reçus	Kb/sec émis
147.35296241904808	6601	0%	6203,21	17,83

Virtual threads				
Débit moy	Temps de réponse moy	% Erreur	Ko/sec reçus	Ko/sec émis
254.324532745	4763 ms	0%	6507,34	50,6

Activité 3 : l'utilisation de l'outil de test de performance « Apache Benchmark » :

Test 1 :

Test 1 (1000 concurrent requests) :

platform threads					
Time taken for tests	Requests per second	Time per request	Failed requests	Transfer rate	Ko/sec émis
168.438	118.74	8421.877	0%	4985.97 Kb/sec	5103,45

Cas 2 : Threads virtuels

virtual threads					
Time taken for tests	Requests per second	Time per request	Failed requests	Transfer rate	Ko/sec émis
130,333	205,4	6007,455	0%	6034,67	6589

Test 2 :

Test 1 (2000 concurrent requests) :

platform threads					
Time taken for tests	Requests per second	Time per request	Failed requests	Transfer rate	Ko/sec émis
149.135	134.11	14913.542	0%	5631.29	5631.29

virtual threads					
Time taken for tests	Requests per second	Time per request	Failed requests	Transfer rate	Ko/sec émis
135,55	168,3	15912,567	0%	6033,553	6033,553

Résumé des conclusions tirées des tests de performance :

1. ***Impact de la concurrence et de la gestion des threads :***

Platform Threads : Gérés par le système d'exploitation, les threads classiques consomment davantage de mémoire et de ressources à mesure que la concurrence augmente, ce qui peut entraîner une surcharge, des temps de réponse plus longs et un débit moyen plus faible.
Virtual Threads : Légers et gérés par le JDK, ils permettent de gérer efficacement une grande quantité de requêtes simultanées, avec moins de consommation mémoire et une meilleure gestion de la concurrence.

2. ***Meilleure performance sous haute concurrence :***

Avec des niveaux élevés de concurrence (1000 à 2000 requêtes simultanées), les Virtual Threads surpassent les Platform Threads en termes de gestion de la charge, de temps de réponse plus courts et de débit plus élevé.

3. ***Meilleur débit de transfert avec les Virtual Threads :***

Les Virtual Threads ont affiché un débit de transfert plus élevé, même avec une charge plus importante, ce qui montre qu'ils sont capables de traiter plus de requêtes tout en maintenant une faible latence.

4. ***Consommation de ressources et scalabilité :***

Les Virtual Threads consomment moins de mémoire et sont plus scalables, ce qui les rend idéaux pour les applications à forte concurrence.
Les Platform Threads, bien que plus efficaces dans des scénarios à faible concurrence, peuvent devenir moins performants sous des charges élevées en raison de leur gestion plus coûteuse des ressources.

Conclusion globale :

Virtual Threads sont plus adaptés pour les environnements à haute concurrence, offrant des meilleures performances, une scalabilité accrue et une gestion optimisée des ressources.

Platform Threads restent utiles pour des charges plus faibles où les threads natifs sont suffisants, mais les Virtual Threads constituent une option plus performante et scalable pour des applications modernes et complexes nécessitant une gestion efficace de la concurrence et des ressources.