Gestion de la mémoire en Java

On Heap VS Off Heap

Sommaire

1. Introduction historique
2. Apparition de la Off Heap
3. Les différences apportées
4. Comment utiliser la Off Heap
5. Conclusion

1. Introduction

- On Heap : Object myObject = new Object();

- Problèmes avec les architectures : compromis entre mémoire du heap et mémoire du Garbage Collector.

- Apparition de la Off Heap avec Java 5.0 pour résoudre ces problèmes.

2. Apparition de la Off Heap

- Avantages de la Off Heap : évolutivité de la taille mémoire, Garbage Collector, pas de dédoublement, meilleurs démarrages…
- Stockage de la Off Heap : sérialisé.

Tout ce qui n’est pas dans la Heap est dans la Off Heap.

3. Les différences apportées

On Heap : peu de gestion; Off Heap : beaucoup de gestion.

Off Heap :

- Plus lente mais peut utiliser des données plus volumineuses;
- Plus facile que les autres stockages de données sur disque dur.
- Temps de démarrage plus rapides.

4. Comment utiliser la Off Heap

Pour créer un objet sur la Off Heap il faut pouvoir accéder, gérer et référencer la mémoire. Pour cela nous utilisons :

- Java Native Access (JNA)

- La classe ByteBuffer

5. Conclusion