# **Understanding Memory Management**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.

## **The Heap and the Nursery**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations. Java objects reside in an area called the heap. The heap is created when the JVM starts up and may increase or decrease in size while the application runs. When the heap becomes full, garbage is collected. During the garbage collection objects that are no longer used are cleared, thus making space for new objects. The nursery is a part of the heap reserved for allocation of new objects. When the nursery becomes full, garbage is collected by running a special young collection, where all objects that have lived long enough in the nursery are promoted (moved) to the old space, thus freeing up the nursery for more object allocation. When the old space becomes full garbage is collected there, a process called an old collection.

The reasoning behind a nursery is that most objects are temporary and short lived. A young collection is designed to be swift at finding newly allocated objects that are still alive and moving them away from the nursery. Typically, a young collection frees a given amount of memory much faster than an old collection or a garbage collection of a single-generational heap (a heap without a nursery).

## **Object Allocation**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.During object allocation, the JRockit JVM distinguishes between small and large objects.

## **Garbage Collection**

Garbage collection is the process of freeing space in the heap or the nursery for allocation of new objects.

\* The Mark and Sweep Model

\* Generational Garbage Collection

\* Dynamic and Static Garbage Collection Modes

\* Compaction

### **The Mark and Sweep Model**

The mostly concurrent mark phase is divided into four parts:

* Initial marking, where the root set of live objects is identified. This is done while the Java threads are paused.
* Concurrent marking, where the references from the root set are followed in order to find and mark the rest of the live objects in the heap. This is done while the Java threads are running.
* Precleaning, where changes in the heap during the concurrent mark phase are identified and any additional live objects are found and marked. This is done while the Java threads are running.
* Final marking, where changes during the precleaning phase are identified and any additional live objects are found and marked. This is done while the Java threads are paused.

The mostly concurrent sweep phase consists of four parts:

* Sweeping of one half of the heap. This is done while the Java threads are running and are allowed to allocate objects in the part of the heap that isn’t currently being swept.
* A short pause to switch halves.
* Sweeping of the other half of the heap. This is done while the Java threads are running and are allowed to allocate objects in the part of the heap that was swept first.
* A short pause for synchronization and recording statistics

#### **Mostly Concurrent Mark and Sweep**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.

#### **Parallel Mark and Sweep**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.

### **Generational Garbage Collection**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.

### **Dynamic and Static Garbage Collection Modes**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.

### **Compaction**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.

### 

#### **External and Internal Compaction**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.

#### **Sliding Window Schemes**

Memory management is the process of allocating new objects and removing unused objects to make space for those new object allocations.

#### **Compaction Area Sizing**

# **Understanding Threads and Locks**

## **Understanding Threads**

### **Default Stack Size for Java Threads**

## **Understanding Locks**

### **Spinning and Sleeping**

### **Lock Chains**