Cập nhật tháng 8 năm 2024

[Bài đọc] Quản lý bộ nhớ trong Java

1. Vùng nhớ trong Java:

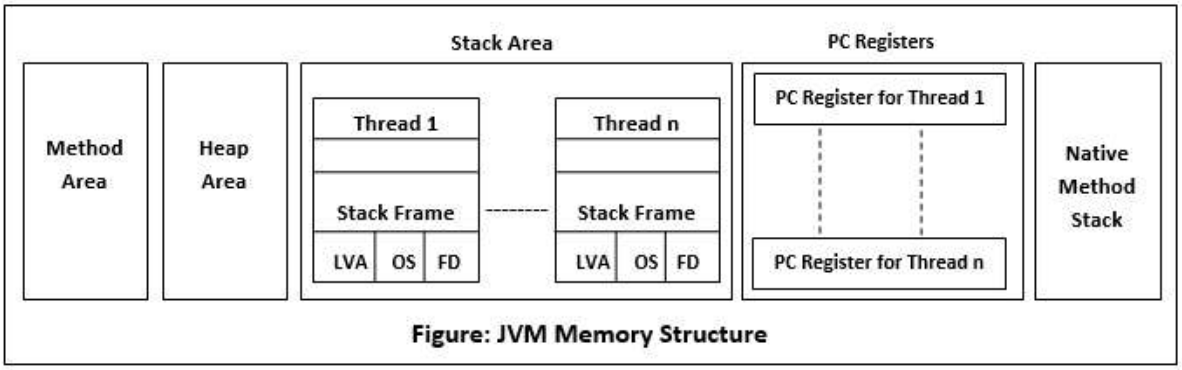

Figure: JVM Memory Structure

- **Code Segment:**

  Đinh nghĩa:

  Code Segment là nơi lưu trữ mã lệnh của chương trình, hay còn gọi là tập hợp các câu lệnh mà CPU thực thi.

  Trong Java:

- Code Segment tương ứng với **Method Area** trong JVM (Java Virtual Machine).
- Chức năng:
  - Lưu thông tin về các **class** đã được tải (metadata).
  - Lưu trữ mã bytecode của các phương thức.
  - Chứa constant pool (bảng hằng số, ví dụ các chuỗi String không thay đổi).
- **Ví dụ:** Khi bạn khai báo final int counter = 0;, biến counter được lưu trong vùng Method Area.

- **Data Segment:**

  Định nghĩa:

  Data Segment chứa các biến toàn cục và các biến tĩnh (static), được cấp phát khi chương trình chạy và tồn tại suốt vòng đời của chương trình.

  Trong Java:

- Biến tĩnh (static) và hằng số (final static) được lưu trong **Method Area**.

- Constant pool của mỗi lớp chứa các giá trị hằng được sử dụng trong chương trình, ví dụ chuỗi "Hello" hoặc số 3.14.

- **Stack Segment:**

  Định nghĩa:

  Stack Segment lưu trữ các biến cục bộ, tham số hàm và địa chỉ trả về của các lời gọi hàm. Stack hoạt động theo cơ chế **LIFO (Last In, First Out)**.

  Trong Java:

- Tương ứng với **JVM Stack**.
- Mỗi luồng (Thread) có một Stack riêng, giúp quản lý dữ liệu độc lập.
- Lưu các biến cục bộ và tham số trong phương thức.
- Nếu bạn khai báo:

  ```
  int x = 5;
  ```

  thì biến x sẽ được lưu trong Stack của luồng hiện tại.

- **Heap Segment:**

  Định nghĩa:

  Heap là vùng nhớ động (Dynamic Memory), nơi lưu trữ dữ liệu được cấp phát tại thời gian chạy bằng các lệnh như malloc() hoặc new.

  Trong Java:

- Tương ứng với **Heap Memory**.
- Dùng để lưu trữ các **đối tượng** và **thực thể của class** được tạo bằng từ khóa new.
- Heap được chia thành:
  - **Young Generation**: Nơi các đối tượng mới được tạo ra (chủ yếu ở vùng **Eden Space**).
  - **Old Generation**: Chứa các đối tượng sống lâu hơn.
  - **Metaspace (Java 8+)**: Lưu metadata về các class thay vì lưu trong Heap.

- **Free Memory:**

  Định nghĩa:

  Phần còn lại của bộ nhớ chưa được sử dụng và có thể cấp phát động.

  Trong Java:

- Tương ứng với vùng Heap chưa được dùng hoặc được giải phóng bởi Garbage Collector.
- **GC (Garbage Collection)** tự động quản lý và thu hồi vùng nhớ không còn được tham chiếu.

2. Quản lý bộ nhớ trong Java:

- Java sử dụng **Garbage Collector** (GC) để tự động thu hồi bộ nhớ của các đối tượng không còn tham chiếu.

- Điều này giúp Java giảm lỗi liên quan đến quản lý bộ nhớ thủ công, nhưng bạn cần đảm bảo sử dụng biến và đối tượng hiệu quả để tránh tình trạng **memory leak** (rò rỉ bộ nhớ).

Link tài nguyên đọc thêm:

- Memories: https://www.javatpoint.com/memory-management-in-java

Danh sách các bài học