Giới thiệu JDK, JRE và JVM trong Java

Hiểu rõ sự khác nhau giữa JDK, JRE và JVM là điều khá quan trọng trong Java. Ở đây, chúng tôi trình bày miêu tả ngắn gọn về JVM, để biết thêm chi tiết về nó, bạn truy cập vào chương tiếp theo. Đầu tiên chúng ta tìm hiểu sự khác nhau cơ bản giữa JDK, JRE và JVM.

Sơ lược về JVM

JVM (viết tắt của Java Virtual Machine) là một thiết bị trừu tượng. Nó cung cấp môi trường runtime mà trong đó Java Bytecode có thể được thực thi.

JVM là có sẵn cho nhiều nền tảng. JVM, JRE và JDK là phụ thuộc nền tảng, bởi vì cấu hình của mỗi OS là khác nhau. Nhưng, Java là độc lập nền tảng.

JVM là gì?

- Là một Specification nơi sự làm việc của Java Virtual Machine được xác định. Nhưng các nhà cung cấp trình triển khai là độc lập với việc lựa chọn thuật toán. Trình triển khai của nó được cung cấp bởi Sun và các một số công ty khác.
- Là một trình triển khai. Trình triển khai của nó được biết đến như là JRE.
- Là Runtime Instance. Bất cứ khi nào bạn viết các lệnh Java trên Command Prompt để chạy các lớp Java thì khi đó sự thể hiện (instance) của JVM được tạo.
- Cung cấp môi trường runtime

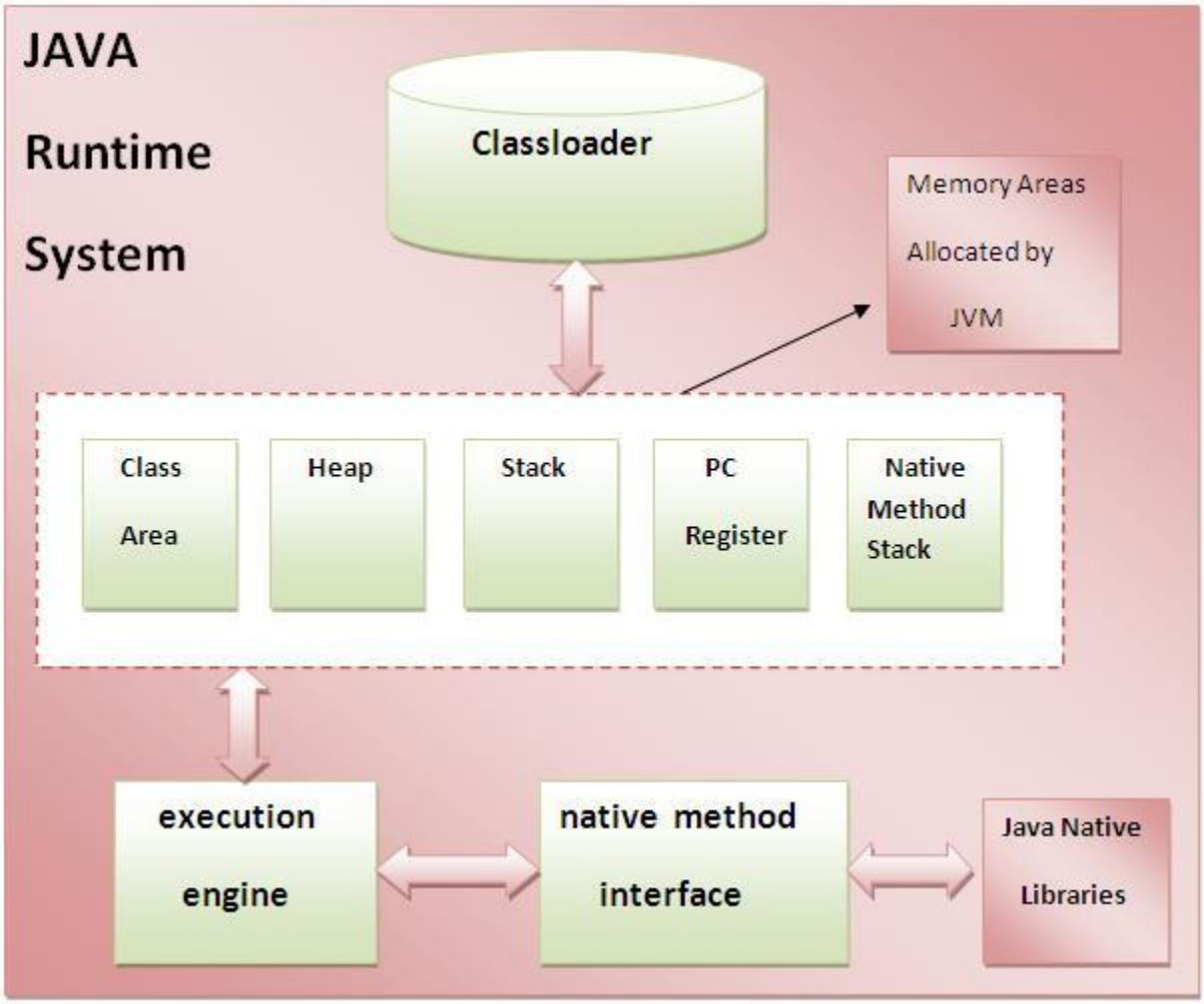
JVM thực hiện các tác vụ chính sau:

- Tải code
- Kiểm tra code
- Thực thi code
- Cung cấp môi trường runtime

JVM cung cấp các định nghĩa cho: Khu vực bộ nhớ, định dạng class file, thiết lập Register, Heap cho Trình dọn rác và các báo cáo lỗi nghiêm trọng (Fatal Error), …

Cấu trúc nội tại của JVM

Bạn theo dõi sơ đồ sau để hiểu cấu trúc nội tại của JVM. Nó gồm Classloader, Memory Area (khu vực bộ nhớ), Excetution Engine (phương tiện thực thi), …



- **Classloader**: Là một hệ thống con của JVM được sử dụng để tải class file.
- **Class (method) Area**: Lưu trữ cấu trúc mỗi lớp, chẳng hạn như hằng, trường, dữ liệu phương thức, code của phương thức, …
- **Heap**: Nó là khu vực dữ liệu runtime mà trong đó đối tượng được cấp phát.
- **Stack**: Stack trong Java lưu giữ các Frame. Nó giữ các biến cục bộ và các kết quả cục bộ, và thực hiện một phần nhiệm vụ trong phần triệu hồi và trả về phương thức. Mỗi Thread có một Stack riêng, được tạo tại cùng thời điểm với Thread.

Một Frame mới được tạo mỗi khi một phương thức được triệu hồi và bị hủy khi lời triệu hồi phương thức là kết thúc.

- **Program Counter Register**: Nó chứa địa chỉ của chỉ lệnh JVM hiện tại đang được thực thi.
- **Native Method Stack**: Bao gồm tất cả các phương thức tự nhiện được sử dụng trong ứng dụng.
- **Execution Engine**: Phần này bao gồm:

  Một bộ xử lý ảo Virtual Processor

  Một trình thông dịch Interpreter. Đọc Bytecode Stream sau đó thực thi các chỉ thị.

  Just-In-Time (JIT) Compiler: được sử dụng để cải thiện hiệu suất. JIT biên dịch các phần của Bytecode mà có cùng tính năng tại cùng một thời điểm, và vì thế giảm lượng thời gian cần thiết để biên dịch. Ở đây khái niệm Compiler là một bộ biên dịch tập chỉ thị của JVM thành tập chỉ thị của một CPU cụ thể.

Giới thiệu JRE

JRE (là viết tắt của Java Runtime Environment) được sử dụng để cung cấp môi trường runtime. Nó là trình triển khai của JVM. JRE bao gồm tập hợp các thư viện và các file khác mà JVM sử dụng tại runtime. Trình triển khai của JVM cũng được công bố bởi các công ty khác ngoài Sun Micro Systems.

Sơ lược về JDK

JDK (là viết tắt của Java Development Kit) bao gồm JRE và các Development Tool.