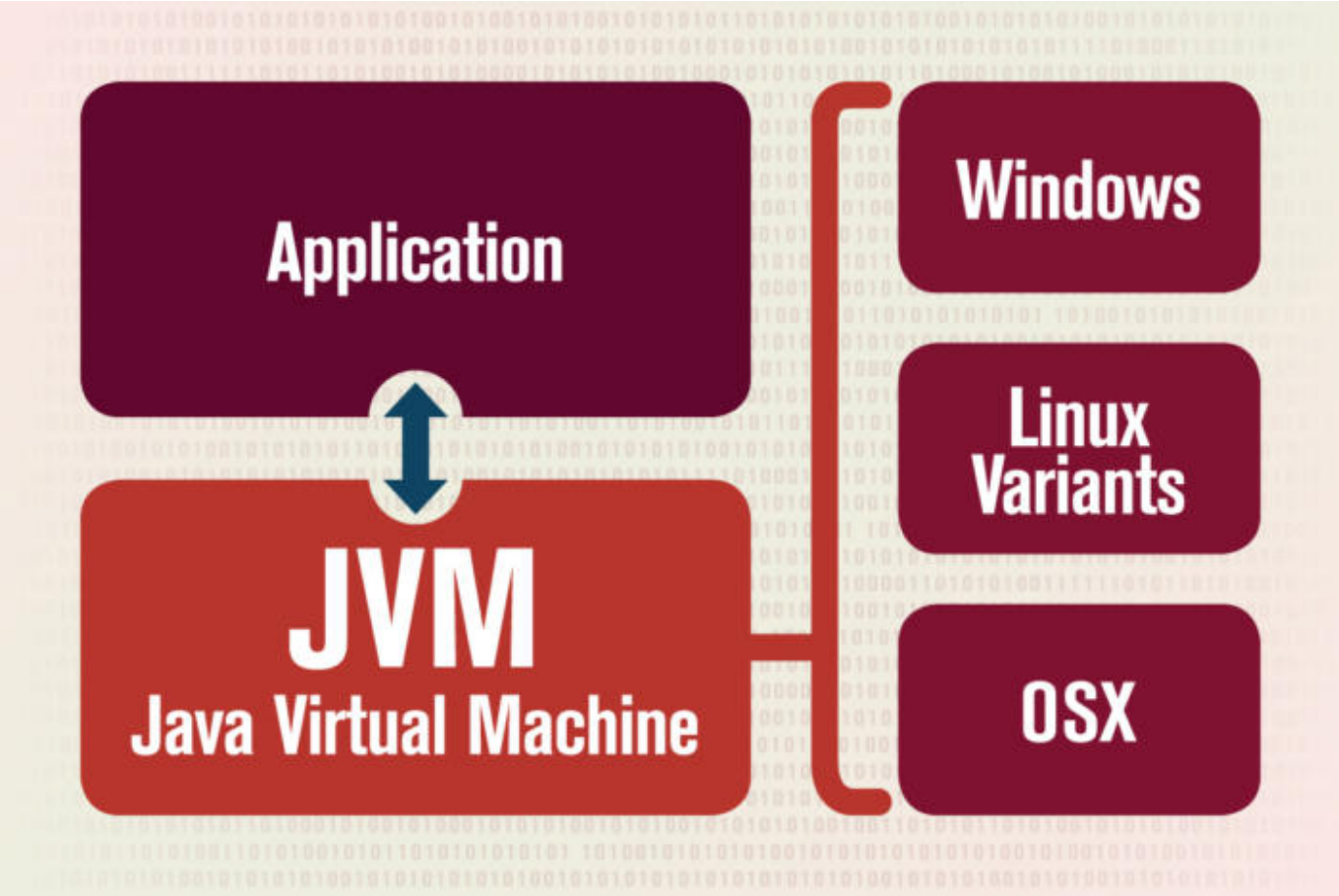
자바 가상 머신(Java Virtual Machine)

시스템 메모리를 관리하면서, 자바 기반 애플리케이션을 위해 이식 가능한 실행 환경을 제공함



JVM은, 다른 프로그램을 실행시키는 것이 목적이다.

갖춘 기능으로는 크게 2가지로 말할 수 있다.

1. 자바 프로그램이 어느 기기나 운영체제 상에서도 실행될 수 있도록 하는 것
2. 프로그램 메모리를 관리하고 최적화하는 것

> JVM은 코드를 실행하고, 해당 코드에 대해 런타임 환경을 제공하는 프로그램에 대한 사양임

개발자들이 말하는 JVM은 보통 어떤 기기상에서 실행되고 있는 프로세스, 특히 자바 앱에 대한 리소스를 대표하고 통제하는 서버를 지칭한다.

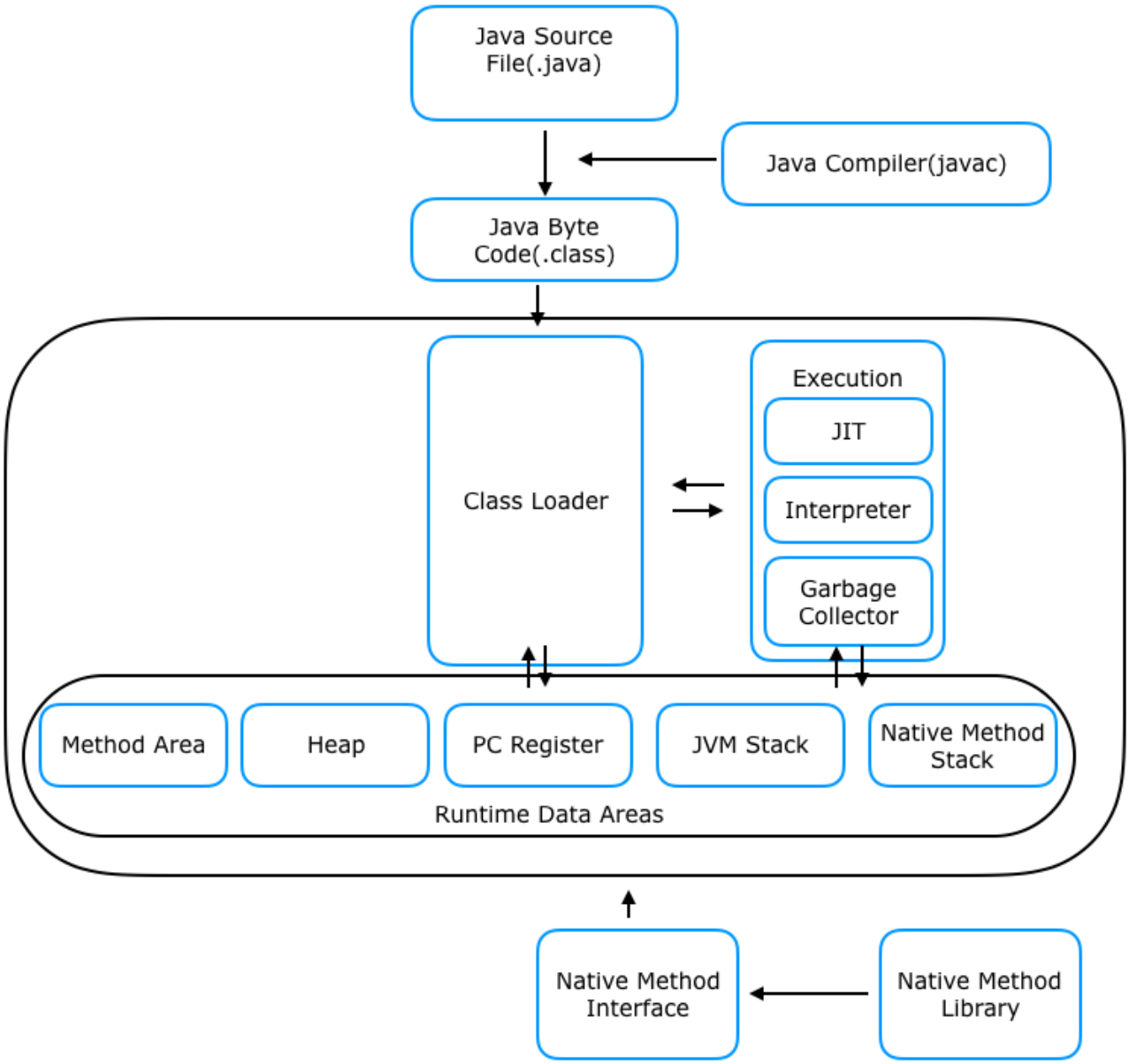
자바 애플리케이션을 클래스 로더를 통해 읽어들이고, 자바 API와 함께 실행하는 역할. JAVA와 OS 사이에서 중개자 역할을 수행하여 OS에 구애받지 않고 재사용을 가능하게 해준다.

JVM에서의 메모리 관리

JVM 실행에 있어서 가장 일반적인 상호작용은, 힙과 스택의 메모리 사용을 확인하는 것

실행 과정

1. 프로그램이 실행되면, JVM은 OS로부터 이 프로그램이 필요로하는 메모리를 할당받음. JVM은 이 메모리를 용도에 따라 여러 영역으로 나누어 관리함
2. 자바 컴파일러(JAVAC)가 자바 소스코드를 읽고, 자바 바이트코드(.class)로 변환시킴
3. 변경된 class 파일들을 클래스 로더를 통해 JVM 메모리 영역으로 로딩함
4. 로딩된 class파일들은 Execution engine을 통해 해석됨
5. 해석된 바이트 코드는 메모리 영역에 배치되어 실질적인 수행이 이루어짐. 이러한 실행 과정 속 JVM은 필요에 따라 스레드 동기화나 가비지 컬렉션 같은 메모리 관리 작업을 수행함



자바 컴파일러

자바 소스코드(.java)를 바이트 코드(.class)로 변환시켜줌

클래스 로더

JVM은 런타임시에 처음으로 클래스를 참조할 때 해당 클래스를 로드하고 메모리 영역에 배치시킴. 이 동적 로드를 담당하는 부분이 바로 클래스 로더

Runtime Data Areas

JVM이 운영체제 위에서 실행되면서 할당받는 메모리 영역임

총 5가지 영역으로 나누어짐 : PC 레지스터, JVM 스택, 네이티브 메서드 스택, 힙, 메서드 영역

(이 중에 힙과 메서드 영역은 모든 스레드가 공유해서 사용함)

PC 레지스터 : 스레드가 어떤 명령어로 실행되어야 할지 기록하는 부분(JVM 명령의 주소를 가짐)

스택 Area : 지역변수, 매개변수, 메서드 정보, 임시 데이터 등을 저장

네이티브 메서드 스택 : 실제 실행할 수 있는 기계어로 작성된 프로그램을 실행시키는 영역

힙 : 런타임에 동적으로 할당되는 데이터가 저장되는 영역. 객체나 배열 생성이 여기에 해당함

(또한 힙에 할당된 데이터들은 가비지컬렉터의 대상이 됨. JVM 성능 이슈에서 가장 많이 언급되는 공간임)

메서드 영역 : JVM이 시작될 때 생성되고, JVM이 읽은 각각의 클래스와 인터페이스에 대한 런타임 상수 풀, 필드 및 메서드 코드, 정적 변수, 메서드의 바이트 코드 등을 보관함

가비지 컬렉션(Garbage Collection)

자바 이전에는 프로그래머가 모든 프로그램 메모리를 관리했음 하지만, 자바에서는 `JVM`이 프로그램 메모리를 관리함!

JVM은 가비지 컬렉션이라는 프로세스를 통해 메모리를 관리함. 가비지 컬렉션은 자바 프로그램에서 사용되지 않는 메모리를 지속적으로 찾아내서 제거하는 역할을 함.

실행순서 : 참조되지 않은 객체들을 탐색 후 삭제 → 삭제된 객체의 메모리 반환 → 힙 메모리 재사용