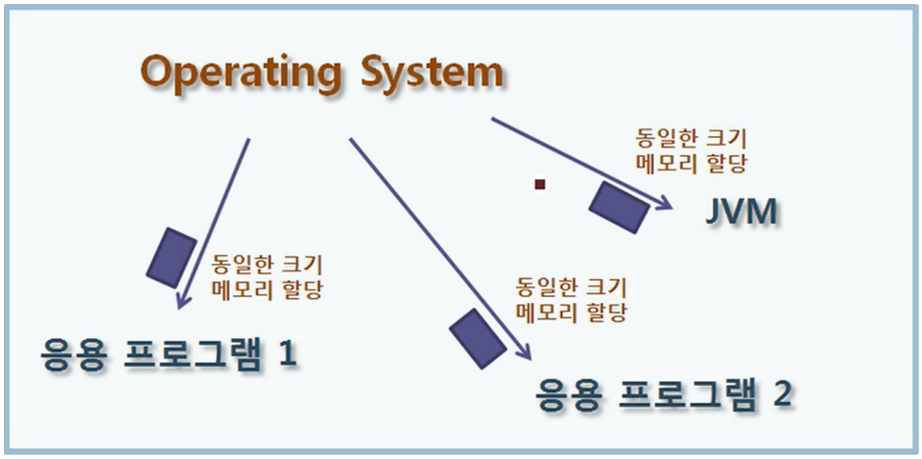
Java Memory Model - JVM의 Memory 관리

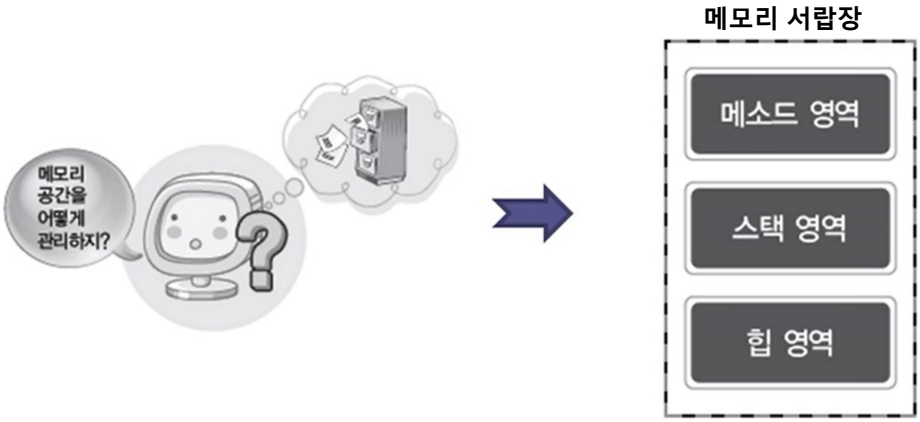
Java II 파트에서 인스턴스의 참조타입 변수에 대해서 학습하면서 Java의 메모리 모델을 간단히 살펴봤었다.
이번 시간에 조금 더 자세히 살펴보고 정리하자

JVM은 운영체제 위에서 동작한다.

- 가상 메모리 기술 : 운영체제가 JVM을 포함한 모든 응용 프로그램에게 동일한 크기의 메모리 공간을 할당할 수 있다.
  제한된 메모리 공간을 가상으로 분할하여 확장하는 기술
- JVM은 운영체제로 부터 할당받은 메모리 공간을 이용해 JVM 자신도 실행해야 하고, 자바 프로그램도 실행해야 한다.
- Java Memory Model : 운영체제로 부터 할당받은 메모리 공간을 이용해 JVM 내에서 별도의 메모리 공간을 구성한다.



JVM의 메모리 살림살이



- JVM이 논리적으로 데이터 특성에 따라 영역을 구분해 놓는다. 이로 인해 Memory Access의 속도가 빨라진다!

- 특성에 따른 분류
  - Method Area : 메소드의 바이트코드, static 변수
  - Stack Area : 지역변수, 매개변수
  - Heap Area : 인스턴스

Method Area (Class LOAD 시점의 공통된 특성)

- 메소드의 자바 바이트코드는 메소드 영역에 저장된다.
- Class Member (static)변수도 메소드 영역에 저장된다.

- JVM은 실행파일을 전부 메모리에 올려 실행하지 않고, 각각의 Class가 필요한 순간에 LOAD한다.

Class의 정의와 더불어 함께 LOAD 되는 데이터들

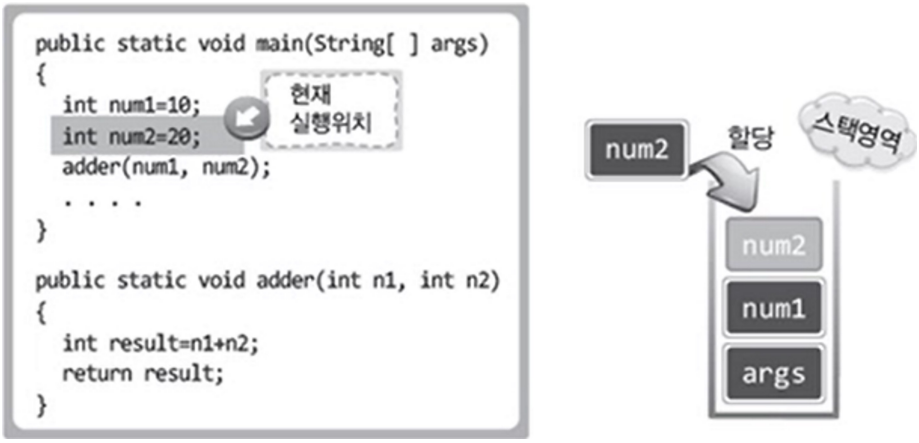
- 프로그램의 흐름을 구성하는 메소드 코드 LOAD
- Class의 Member인 static 변수 LOAD

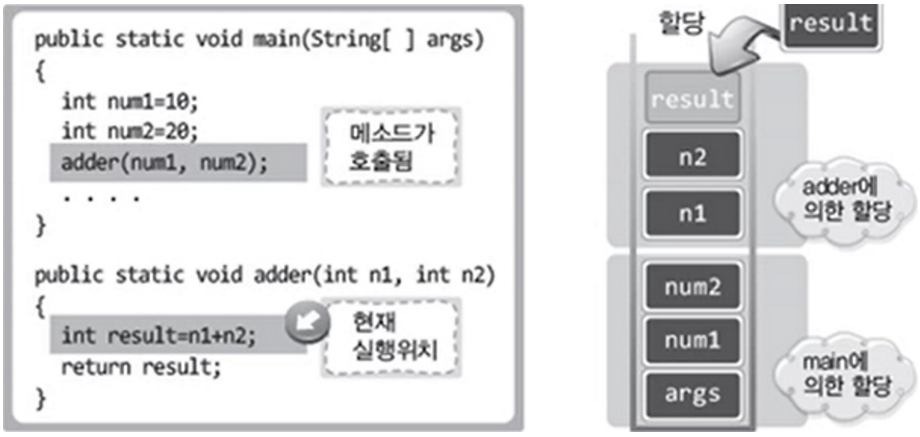
Stack Area (임시 할당과 소멸)

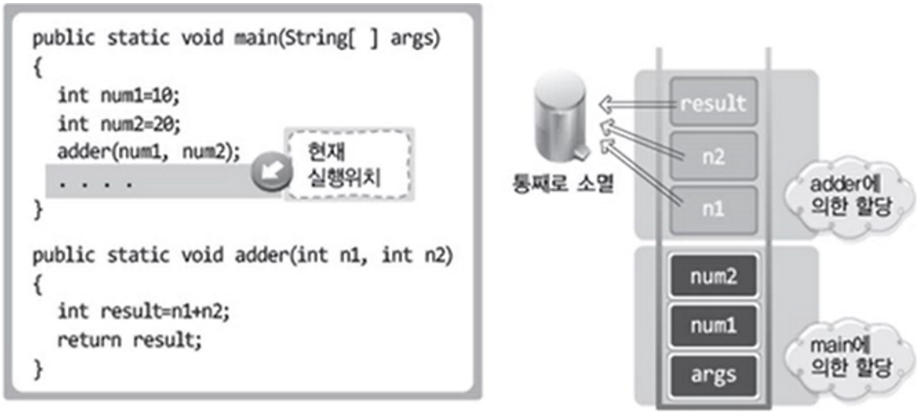
- 매개변수, 지역변수가 할당되는 공간
- 메소드의 실행을 위한 메모리 공간

Stack Flow

- 스택에 데이터가 할당되는 것을 PUSH, 소멸되는 것을 POP이라고 한다.
- 스택에 할당된 지역변수는 해당 메소드가 종료되면 소멸된다.
- 할당 및 소멸의 특성이 접시 쌓는 것과 유사하여 스택이라고 이름이 지어졌다.







Heap Area (인스턴스를 위한 영역)

- 인스턴스가 생성되는 메모리공간
- JVM에 의한 메모리 공간의 인스턴스 정리 (Garbage Collection)이 이루어 지는 공간

- 인스턴스의 할당은 프로그래머가, 소멸은 JVM이 담당
- Garbage Collection의 탐색 효율성을 위해 Heap Area에 인스턴스를 할당
- 정리 기준 : 참조변수에 의한 참조가 이루어지지 않는 인스턴스

인스턴스의 소멸시기

```
public static void main(String[] args){
     String str1 = new String("My String");
     String str2 = new String("Your String");

     . . . . .
     str1 = null;
     str2 = null;

     . . . . .
}
```