Java memory area

Java virtual machine utilizes four different memory areas to store java class members:

1. Class Area
2. Heap area
3. Method area
4. Stack area

Class Area

Class area is used to store the static members of the class. Static members include both static member variable and static member functions.

Heap Area

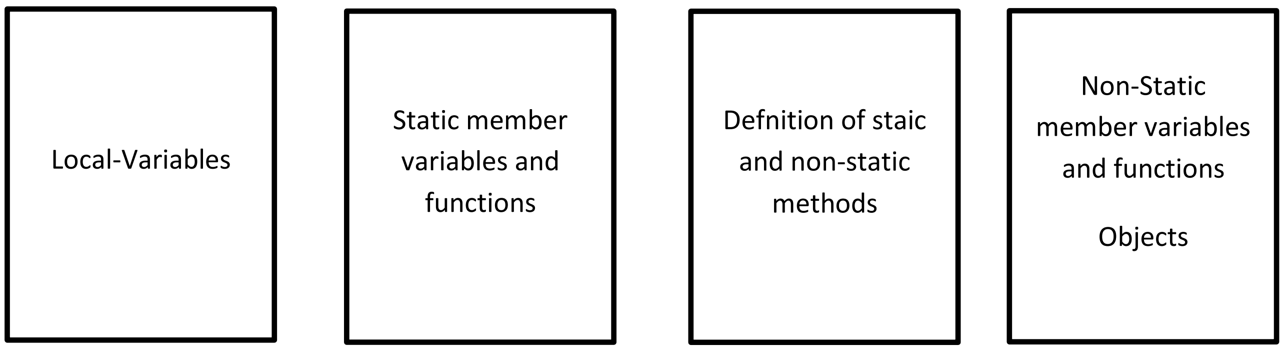
Heap area is used to store the non-static members of the class. Non-Static members include both non-static member variable and non-static member functions.
The objects are stored inside the heap area.

Method Area

Method area is used to store the definition of the method. Declaration of the method will be stored either in class area or in heap area.
If method is static declaration part of method will be stored in class area and for non-static declaration part will be stored in heap memory.

Stack Area

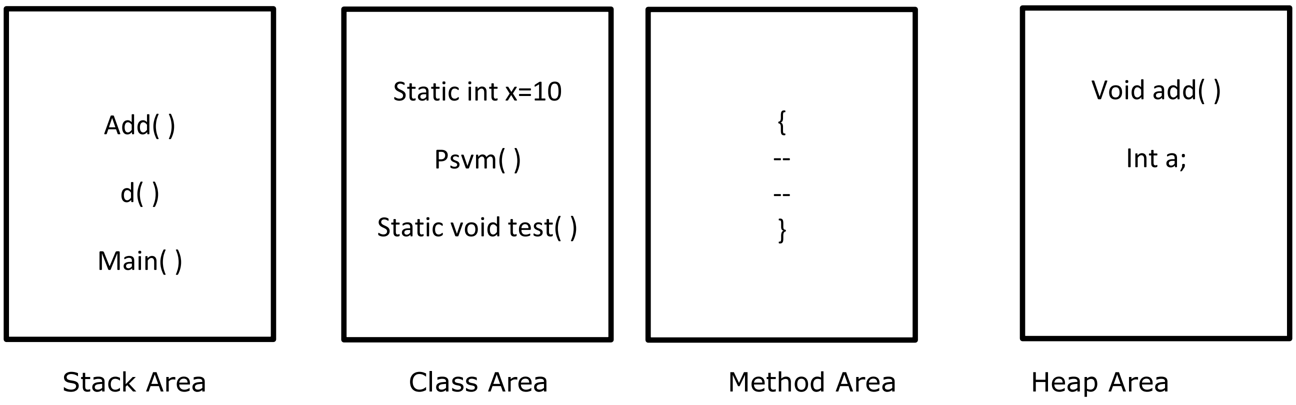
Stack area is also called as an execution area. In java if we call any method the definition of method will be loaded to the stack area of execution.
The local variables of the program will be stored in stack area.


Local-Variables
Static member variables and functions
Defnition of staic and non-static methods
Non-Static member variables and functions
Objects

```
Class Demo
{
Static int x=10;

Void add()
{
Int a;
System.out.println(a);
}
Static void test()
{
System.out.println(x);
}

Public static void main(String args[])
{
Demo d=new Demo();
d.add();
}
}
```



```
class Java_Memory1
{
int i;
int j;

public static void main(String args[])
{

Java_Memory1 m1=new Java_Memory1();

m1.i=10;
m1.j=20;


Java_Memory1 m2=m1;

m2.i=30;
m2.j=40;

System.out.println(m1.i);
System.out.println(m1.j);

}
}

// 30, 40;
```

Since m2 is assigned to m1, address is copied into m1 refernce , any changes made to the variable will be reflected.

```
class Java_Memory2
{
int i;
int j;

public static void main(String args[])
{
Java_Memory2 m1=new Java_Memory2();
m1.i=10;
m1.j=20;

Java_Memory2 m2=new Java_Memory2();

m2.i=30;
m2.j=40;

System.out.println(m1.i);
System.out.println(m1.j);

}
}
```

Two different objects are created hence two different values are printed.