DT : 26/11/2020

JVM Architecture:(JVM Internals)

=>JVM stands for 'Java Virtual Machine' and which is used to execute Java Byte Code.

=>Virtual machine means the S/W component which internally having the behaviour like machine.

=>JVM internally divided into three parts:

1.Class Loader SubSystem

2.Runtime DataArea

3.Execution Engine

1.Class Loader SubSystem:

=>Class Loader SubSystem will load Java Byte Code on to JVM,in this process Class Loader SubSystem uses the following components:

(a)Loader

(b)Linker

(c)Initiate

(a)Loader:

=>Loader will load the required files into current running program.

=>According to JavaLang the required files are available in three locations(JavaLib,ext folder and classpath).

=>To load the required files from three locations,the loader internally uses the following SubLoaders:

(i)BootStrap Class Loader

=>BootStrap CL will load the required files from the JavaLib.

Exp:

"System" and "String" classes are loaded from JavaLib.

(ii)Extention Class Loader:

=>Extention Class Loader will load the required files from the "ext" folder.

C:\Program Files\Java\jdk1.8.0_251\jre\lib\ext

(iii)Application Class Loader:

=>Application CL will load the required files from "classpath"

(b)Linker:

=>Linker will link the loaded files into current running program where they are needed,in this process the Linker internally uses the following components:

(i)Verify

(ii)Prepare

(iii)Resolve

(i)Verify:

=>Verify component will perform verification process,in this process the component will check the loaded and required files are same or not.

(ii)Prepare:

=>Prepare component will perform decoding process,in this it

identify the programming components.(Variable,method,...)

(iii)Resolve:

=>Resolve component will check the programming components are static or NonStatic based on 'static' keyword.

Exp:

In the above program,

```
  static   :   main()
NonStatic  :   a,b,c,add()
```

Note:

=>Based on 'static' keyword the programming components are categorized into two types:

(a)static programming components

(b)NonStatic programming components

(a)static programming components:

=>The programming components which are declared with static keyword are known as Static programming components.

=>These Static programming components will get the memory within the class while class loading and access with class_name.

(b)NonStatic programming components:

=>The programming components which are declared without static keyword are known as NonStatic programming components.

=>These NonStatic programming components will get the memory within the object while object creation and access with Object_name.

--

(c)Initiate:

=>Initiate component will perform initialization process and in this process one memory is created known as "Runtime Data Area".

2.Runtime DataArea:

=>Runtime Data Area internally divided into the following blocks:

(a)Method Area

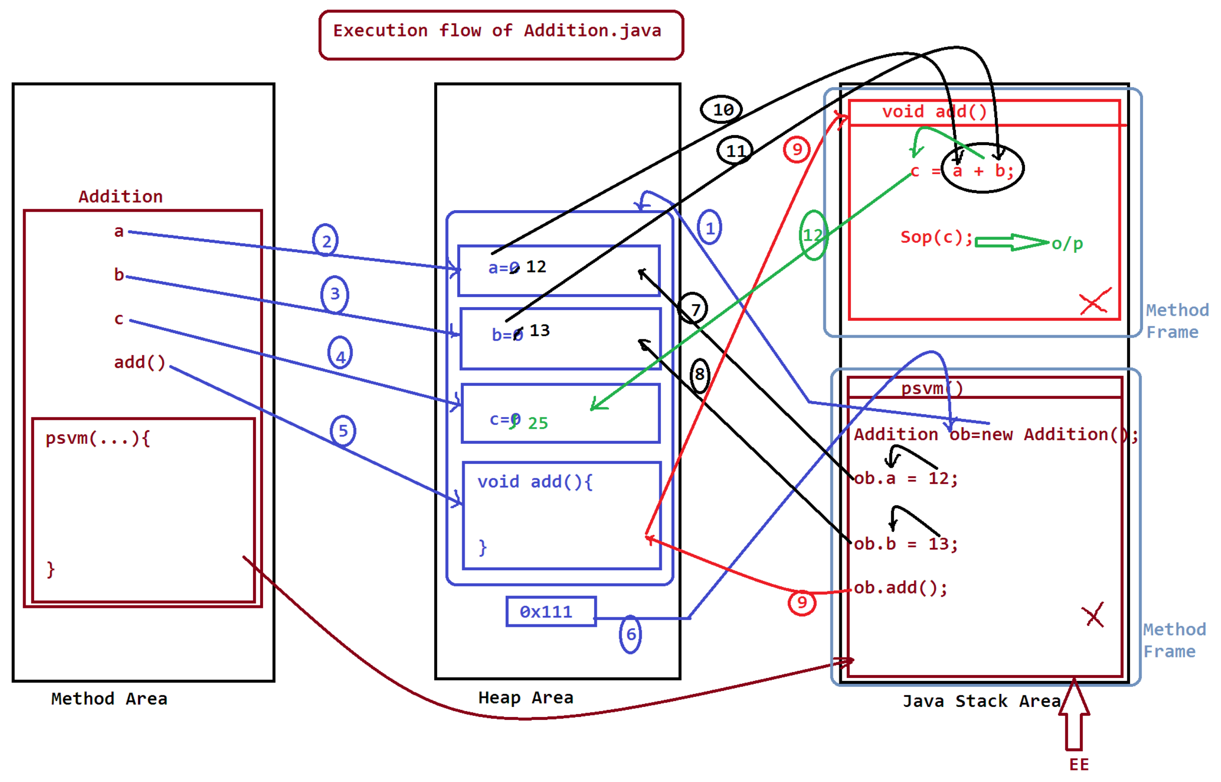
(b)Heap Area

(c)Java Stack Area

(d)PC Register Area

(d)Native Method Area

Dt : 27/11/2020



(a)Method Area:

=>The memory block where the class is loaded is known as Method Area.

=>while class loading the static members of the class will get the memory within the class,in this process main() will get the memory within the class.

=>Once main() get the memory within the class then it is automatically copied on to Java StackArea.

=>The ExecutionEngine(Execution Control) will detect main() method from JavaStackArea and starts the execution process.

(b)Heap Area:

=>The memory block where the objects are created is known as Heap Area.

Execution Behaviour of 'new' keyword:

=>"new" keyword will specify the execution control to create reference part of Heap Area.
=>"new" keyword will specify the execution control to check the required class is available on method_area or not.
=>If the class is available on Method_area then take the nonstatic members of the class and allocate memory at reference.
=>Once all the NonStatic members got the memory at reference then load the reference on to reference variable or Object_name.

--

Note:
=>In the process of constructing JavaAppl we use one MainClass and can have any number of SubClasses.

MainClass : which is holding main() method
SubClass : which is holding variables and methods without main() method

Exp program2:
wap to display Employee data?

EmpDetails

  =>empId,empName,empDesg,empSal

  =>void getEmpDetails()

EmpAddress

  =>hNo,sName,city,pinCode

  =>void getEmpAddress()

Employee

 =>public static void main(String[] args)

/*program to display Employee Data*/

```
import java.lang.String;
import java.lang.System;
class EmpDetails //SubClass
{
        String empId,empName,empDesg;
        int empSal;
        void getEmpDetails()
        {
System.out.println("EmpId:"+empId);
System.out.println("EmpName:"+empName);
System.out.println("EmpDesg:"+empDesg);
System.out.println("EmpSal:"+empSal);
```

```
        }
}
class EmpAddress //SubClass
{
        String hNo,sName,city;
        int pinCode;
        void getEmpAddress()
        {
System.out.println("HNo:"+hNo);
System.out.println("sName:"+sName);
System.out.println("City:"+city);
System.out.println("pinCode:"+pinCode);
        }
}
class Employee //MainClass
{
        public static void main(String[] args)
        {
EmpDetails ed = new EmpDetails();
EmpAddress ea = new EmpAddress();

ed.empId = "A121";
ed.empName = "Raj";
ed.empDesg = "SE";
ed.empSal = 30000;
```

```
ea.hNo = "12-34/h";

ea.sName = "SR Nagar";

ea.city = "Hyd";

ea.pinCode = 612345;


ed.getEmpDetails();

ea.getEmpAddress();

        }

}
```