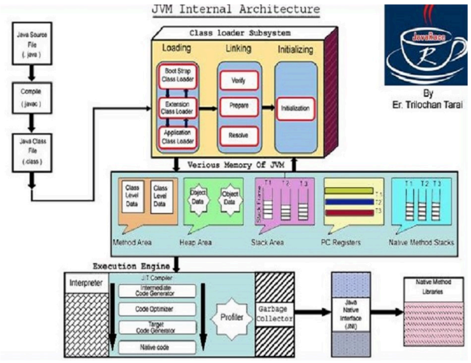
# JVM in detail



- let’s say we have a java source file (i.e. *text.java*)
    * then the java compiler *javac* compiles the *test.java* file and generates *test.class* file
    * this *test.class* file is the *input* of *JVM*
        ** that mean *JVM* takes *.class* file, then **loads** it and **executes** it
- **JVM** consists of 3 modules :
    * **class loader subsystem**
    * **memory area**
    * **execution engine**

## class loader subsystem

- **class loader subsystem** consists of 3 activities :
    * **loading**
    * **linking**
    * **initialising**
        ** that means it is responsible for **loading**, **linking** and **initialising** the *.class* file
- in **loading activity**, there are 3 types of loaders
    * **bootstrap class loader**
    * **extension class loader**
    * **application class loader**
- **bootstrap class loader** is responsible for loading *class* from *bootstrap class path* (*rt.jar*)
    * all the *core java API classes* are loading in *bootstrap class loader*

- **extension class loader** is responsible for loading the *classes* present *inside* the *exe* folder
- **application class loader** is responsible to load *application level class path* such as *enviroment variable* or *enviromental class path*
- among these 3 class loaders