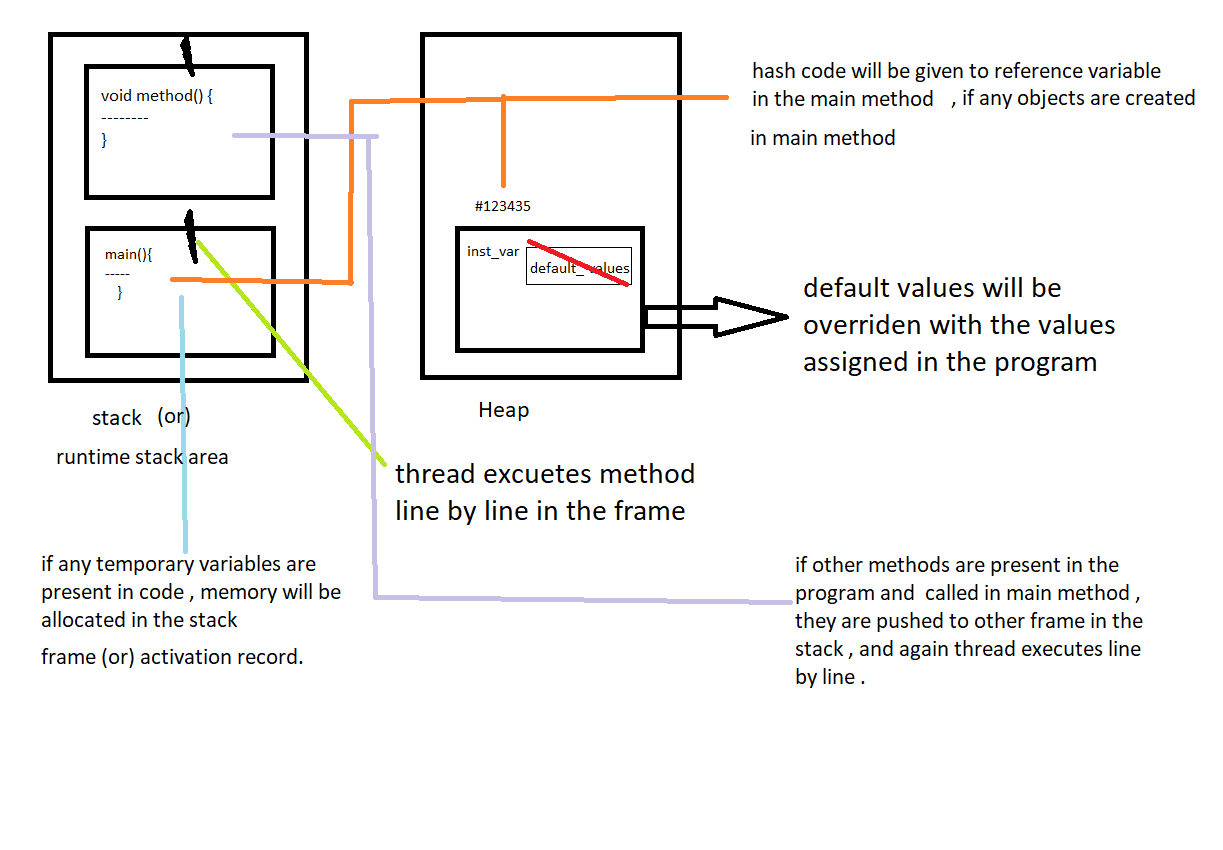
Methods

Method has 4 parts

1. Name
2. Input(parameter)
3. Body
4. Return type

In a method there is some task and if I want that task to be executed , then that method has to be brought up into the stack (or) Runtime stack area

Every thing done here is with respect to byte code.



Process :

Check jvm data area picture in previous file

* When the program executes first main method pushes into the stack frame
* Then a thread executes the main method line by line in the activation record
* If any object is present , first R.H.S side is executed in the object .
* Then memory will be created in the heap for instance variables and default values are assigned to them w.r.t to their data types .
* The hash code assigned memory space where the instance variables are stored is sent to reference variable in main method ( how hash code sits on reference variable process will be given in constructor)
* If any instance methods present in the program are called in the main method , that method is brought up into another stack frame in run time stack area.
* That method is also executed line by line by a thread.
* After the execution of that whole method , it will be pushed out (added) from the run time stack area.
* After execution of all the lines in the main method , it will also be popped out (deleted)
* So there is no reference variable in the stack to hold hash code
* Now garbage collector will check for unreferenced objects and deletes them .

Note : if the method returns anything it is optional whether to collect it or not

Eg: Method\_With\_Return\_Type\_And\_Parameters

If method is declared with parameters, arguments should be passed compulsory.

Note : // Nested methods are not supported in java . it is supported until java 7 or older version , from java 8 you can achieve it by lambda expression