Garbage collector :

- Garbage collector is destroys the objects.
- main objective of Garbage collector is to free the heap memory by destroying the unreachable objects
-

* unreachable objects ?

An object is said to be unreachable iff it does't contain any reference to it. The unreachable objects are eligible for garbage collection.

Four ways to make Object to be eligible for GC

1. Nullifying the reference variable ⇒ ref var=null;
2. Re - Assigning the reference variable ⇒ r1 = r2;
   ↑ gc
3. object created inside method : Out of scope then gc.
4. Island of Isolation.

→ it may not destroy immediately by garbage collector whenever Jvm runs Garbage collector program, then only object will be destroyed. But, when Jvm runs Garbage collector we can not except.

Two ways to run Garbage Collector

① using System.gc(); with System.runFinalization();

② Runtime.getRuntime().gc();

- Before destroying an object Garbage Collector calls finalize() method on the object to perform cleanup activities. once finalize() method completes Garbage collector destroys that object.

Syntax:

```
Protected void finalize() throws Throwable.
```

```
Jvm → gc → finalize();
```

- finalize method called by Garbage Collector not Jvm
- finalize method implementation is empty in object class
- finalize method never invoked more than once for any given object
- if an uncaught exception thrown by finalize() method the exception is Ignored and finalization of that object terminate.

Is land of Isolation

- Object1 refferences object2 and object2 refferen
object1 and neither object1 nor object2 is referenced by any other objects that's an island of isolation.

- If Both objects are made null then they are eligible for garbage collection.

```
obj1 = obj2 ;
obj2 = obj1 ;
```

→ neither of these to referred by other

make them NULL

↓

GC → Eligible