How to make object eligible for garbage collection in Java?

An object is eligible to be garbage collected if its reference variable is lost from the program during execution.Sometimes they are also called **unreachable objects.**

What is reference of an object?

The *new* operator dynamically allocates memory for an object and returns a reference to it. This reference is the address in memory of the object allocated by new. A reference is an address that indicates where an object's variables, methods etc. are stored.

The objects are not actually used when assigned to a variable or passed as an argument to a method . The references to objects are used everywhere. Example:

```
Box mybox =  new Box();   //referencing to object
```

Role of an unreachable objects in java

In java, the memory allocated at runtime i.e. heap area can be made free by the process of garbage collection. It is nothing but just a method of making the memory free which is not being used by the programmer. Only the objects who have no longer reference to them are eligible for garbage collection in java.

Ways to make an object eligible for garbage collection:

Please note that the object can not become a candidate for garbage collection until all references to it are discarded.

1. **Object created inside a method :** When a method is called it goes inside the stack frame. When the method is popped from the stack, all its members dies and if some objects were created inside it then these objects becomes unreachable or anonymous after method

execution and thus becomes eligible for garbage collection

.Example:

```
/* Java program to demonstrate that
objects created inside a method will becomes
eligible for gc after method execution terminate */

class Test
{

    // to store object name
    String obj_name;

    public Test(String obj_name)
    {
        this.obj_name = obj_name;
    }

    static void show()
    {
        //object t1 inside method becomes unreachable when show() removed
        Test t1 = new Test("t1");
        display();

    }
    static void display()
    {
        //object t2 inside method becomes unreachable when display() removed
        Test t2 = new Test("t2");
    }

    // Driver method
    public static void main(String args[])
    {
        // calling show()
        show();

        // calling garbage collector
        System.gc();
    }

    @Override
    /* Overriding finalize method to check which object
    is garbage collected */
    protected void finalize() throws Throwable
    {
        // will print name of object
        System.out.println(this.obj_name + " successfully garbage collected"
    }
}
```

Run on IDE

Output:

```
t1 successfully garbage collected
t2 successfully garbage collected
```

Note : If a method returns the object created inside it and we store this object reference by using a reference-type variable than it is no longer eligible for garbage collection.

2. **Reassigning the reference variable:** When reference id of one object is referenced to reference id of some other object then the previous object has no any longer reference to it and becomes unreachable and thus becomes eligible for garbage collection.Example:

```
/* Java program to demonstrate gc
 when one object referred to other object */

class Test
{
    // to store object name
    String obj_name;

    public Test(String obj_name)
    {
        this.obj_name = obj_name;
    }

    // Driver method
    public static void main(String args[])
    {
        Test t1 = new Test("t1");
        Test t2 = new Test("t2");

        //t1 now referred to t2
        t1 = t2;

        // calling garbage collector
        System.gc();
    }

    @Override
    /* Overriding finalize method to check which object
     is garbage collected */
    protected void finalize() throws Throwable
    {
        // will print name of object
        System.out.println(this.obj_name + " successfully garbage collected"
    }
}
```

Run on IDE

Output:

```
t1 successfully garbage collected
```

3. **Nullifying the reference variable :** When all the reference variables of an object are changed to NULL, it becomes unreachable and thus becomes eligible for garbage collection.Exam

```
/* Java program to demonstrate gc
 when object reference changed to NULL */
```

```
class Test
{
    // to store object name
    String obj_name;

    public Test(String obj_name)
    {
        this.obj_name = obj_name;
    }

    // Driver method
    public static void main(String args[])
    {
        Test t1 = new Test("t1");

        /* t1 being used for some purpose in program */

        /* When there is no more use of t1, make the object
           referred by t1 eligible for garbage collection */
        t1 = null;

        // calling garbage collector
        System.gc();
    }

    @Override
    /* Overriding finalize method to check which object
     is garbage collected */
    protected void finalize() throws Throwable
    {
        // will print name of object
        System.out.println(this.obj_name + " successfully garbage collected"
    }
}
```

Run on IDE

Output:

```
t1 successfully garbage collected
```

4. **Anonymous object** : The reference id of an anonymous object is not stored anywhere. Hence, it becomes unreachable.

   Example:

```
/* Java program to demonstrate gc
 of anonymous objects */

class Test
{
    // to store object name
    String obj_name;

    public Test(String obj_name)
    {
        this.obj_name = obj_name;
    }

    // Driver method
    public static void main(String args[])
    {
        //anonymous object without reference id
        new Test("t1");

        // calling garbage collector
        System.gc();
    }

    @Override
    /* Overriding finalize method to check which object
     is garbage collected */
    protected void finalize() throws Throwable
    {
        // will print name of object
        System.out.println(this.obj_name + " successfully garbage collected"
    }
}
```

Run on IDE

Output:

```
t1 successfully garbage collected
```

Related Article: Island of Isolation

This article is contributed by **Apoorva Singh and Gaurav Miglani**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Recommended Posts:

Memory leaks in Java
Mark-and-Sweep: Garbage Collection Algorithm