VIT-AP UNIVERSITY

Course code : **CSE2005**

Course title : **Object Oriented Programming**

Finalize() method

Objectives

This session will give the knowledge about

- Get basic information about garbage collection
- Define finalize method

VIT-AP UNIVERSITY

Garbage Collection

Garbage Collection- Introduction

You have created objects of classes in last several programs. What do you think will happen to the memory occupied by these object when the program finishes?

Once the program completes,

These objects become garbage…

Now what to do with these garbage??? We need to clean it up!!!

Java's Cleanup Mechanism – The Garbage Collector

Java has its own Garbage Collector

Consider the following:

Test test=new Test();

Here when the object is created, memory is allocated for this object.

test=null; => When you execute this, the reference is deleted but the memory occupied by the object is not released.

We need to release the memory occupied by the 'test' object.

Java's Cleanup Mechanism – The Garbage Collector

- Objects on the heap must be de-allocated or destroyed, and their memory released for later reallocation
- Java handles object de-allocation automatically through Garbage Collection
- Objects which are occupying memory but not referenced will be reclaimed

Java's Cleanup Mechanism – The Garbage Collector

- The Garbage Collection is done automatically by JVM.
- If we need to manually do the garbage collection, we can use the following method:
  - Runtime rs = Runtime.getRuntime();
  - rs.gc();
- The gc() method is used to manually run the garbage collection.

Example

```
package vit.demo;
import java.util.*;
public class Main {
   public static void main(String s[]) throws Exception {
            Runtime rs = Runtime.getRuntime();
            System.out.println("memory before Garbage Collection = " + rs.freeMemory());
            rs.gc();
            System.out.println("memory after Garbage Collection = " + rs.freeMemory());
   }
}
```

Here the rs.freeMemory() method will give the free memory available in the system. Try this program and observe the results…

The finalize( ) Method

Often, an object needs to perform some action when it is destroyed

The action could pertain to:

- releasing a file handle
- reinitializing a variable, such as a counter

Java's answer is a mechanism called finalization

By using finalization, you can define specific actions that will occur when an object is just about to be reclaimed by the garbage collector

The finalize( ) Method

- To add a finalizer to a class, you simply define the finalize( ) method
- The Java runtime calls that method whenever it is about to recycle an object of that class
- Inside the finalize( ) method, you will specify those actions that must be performed before an object is destroyed

Example

```
package vit.demo;
import java.util.*;
public class Main {
   public static int count;
   public Main() {
          count++;
   }
   public static void main(String args[]) {
          Main ob1 = new Main();
          System.out.println("Number of objects :" + Main.count);
          Main ob2 = new Main();
          System.out.println("Number of objects :" + Main.count);
```

Example

```
        Runtime r = Runtime.getRuntime();
        ob1 = null;
        ob2 = null;
        r.gc();
    }
    protected void finalize() {
        System.out.println("Program    about to termonate");
        Main.count--;
        System.out.println("Number of objects: " + Main.count);
    }
}
```

```
Number of objects :1
Number of objects :2
Program about to termonate
Number of objects: 1
Program about to termonate
Number of objects: 0
```

Summary

We have discussed about

- Get basic information about garbage collection
- Define finalize method