Programme : Bsc
Subject : Computer Science
Semester : V
Paper Code : CSC 106
Paper Title : Object Oriented Programming
Unit V : The run-time structure :Objects
Module Name : The component-level approach , Automatic memory management
Module No : 14
Name of the Presenter: Mr. Filipe Rodrigues

Lecture Notes

The concept of Destructors

- ✓ a special method that gets called automatically as soon as the life-cycle of an object is finished
- ✓ called to de-allocate and free memory.
- ✓ No arguments can be passed in a destructor.
- ✓ Overloading is not allowed.
- ✓ The following tasks get executed when a destructor is called.
  - Closing all the database connections or files
  - Releasing all the network resources
  - Recovering the heap space allocated during the lifetime of an object
- ✓ In Java , the finalize method performs the function of a Destructor

Garbage collection in Java

Whenever you run a java program, JVM creates three threads.

- ✓ 1) main thread – execute the main method
- ✓ 2) Thread Scheduler(background) - Task is to schedule threads
- ✓ 3) Garbage Collector Thread(background). – Task is to collect abandoned object from the heap memory.
  - Abandoned objects- don't have live references
  - Garbage collector calls finalize method before sweeping out abandoned objects.
  - After the finalize method is called , object is destroyed from the memory.

finalize() method in Java

- ✓ Finalize() is the method of Object class.
- ✓ This method is called just before an object is garbage collected.
- ✓ finalize() method overrides to dispose system resources, perform clean-up activities and minimize memory leaks.

```
public class FinalizeExample {
    public static void main(String[] args)
  {
      FinalizeExample obj = new FinalizeExample();
      obj = null;          // calling garbage collector
       System.gc();
       System.out.println("end of garbage collection");

  }
  @Override
   protected void finalize()
  {
     System.out.println("finalize method called");
   }
}
```

O/P: end of garbage collection
finalize method called

Memory Leaks

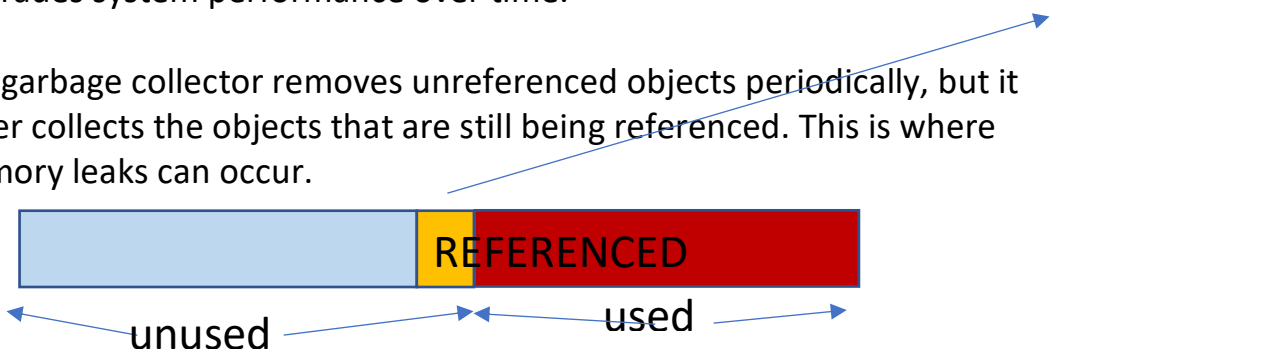
A Memory Leak is a situation when there are objects present in the heap that are no longer used, but the garbage collector is unable to remove them from memory and, thus they are unnecessarily maintained.

A memory leak is bad because it:

blocks memory resources

degrades system performance over time.

The garbage collector removes unreferenced objects periodically, but it never collects the objects that are still being referenced. This is where memory leaks can occur.



The need for automatic memory management

A good O-O environment should offer an automatic memory management mechanism

- detect and reclaim unreachable objects,
- allowing application developers to concentrate on their job .

Object memory Reclamation

Memory of reclaimed object added to free cell list.

Or the associated memory is returned to the Operating system.

For the subsequent object creation, Free List is first searched

If empty request O.S. for memory

Object cloning in Java

- ✓ The object cloning is a way to create exact copy of an object. The clone() method of Object class is used to clone an object.
- ✓ The java.lang.Cloneable interface must be implemented by the class whose object clone we want to create.
- ✓ If we don't implement Cloneable interface, clone() method generates CloneNotSupportedException.
- ✓ Steps involved:
  - ✓ Define a parent class,
  - ✓ implement Cloneable interface
  - ✓ provide the definition of the clone() method

Object cloning in Java:an example

```
class Student implements Cloneable{
int rollno;
String name;

Student(int rollno,String name){
this.rollno=rollno;
this.name=name;
}

public Object clone()throws CloneNotSupportedException{
return super.clone();
}
```

```
public static void main(String args[]){
try{
Student s1=new Student(1011,”Jim");
Student s2=(Student)s1.clone();
System.out.println(s1.rollno+" "+s1.name);
System.out.println(s2.rollno+" "+s2.name);
}
catch(CloneNotSupportedException c){}
}
}
```

Output: 1011 Jim

1011 Jim