**java.lang package – Part-03**

* **equals() method:**

We can use equals method, to check equality of two objects.

If our class doesn’t contain equals method, then Object class equals() method will be executed.

class Test{

String name;

int rollNo;

Test(String name, int rollNo){

this.name=name;

this.rollNo= rollNo;

}

public static void main(String[] args){

Student s1 = new Student(“Durga”, 101);

Student s2 = new Student(“Ravi”, 102);

Student s3 = new Student(“Durga”, 101);

Student s4 = s1;

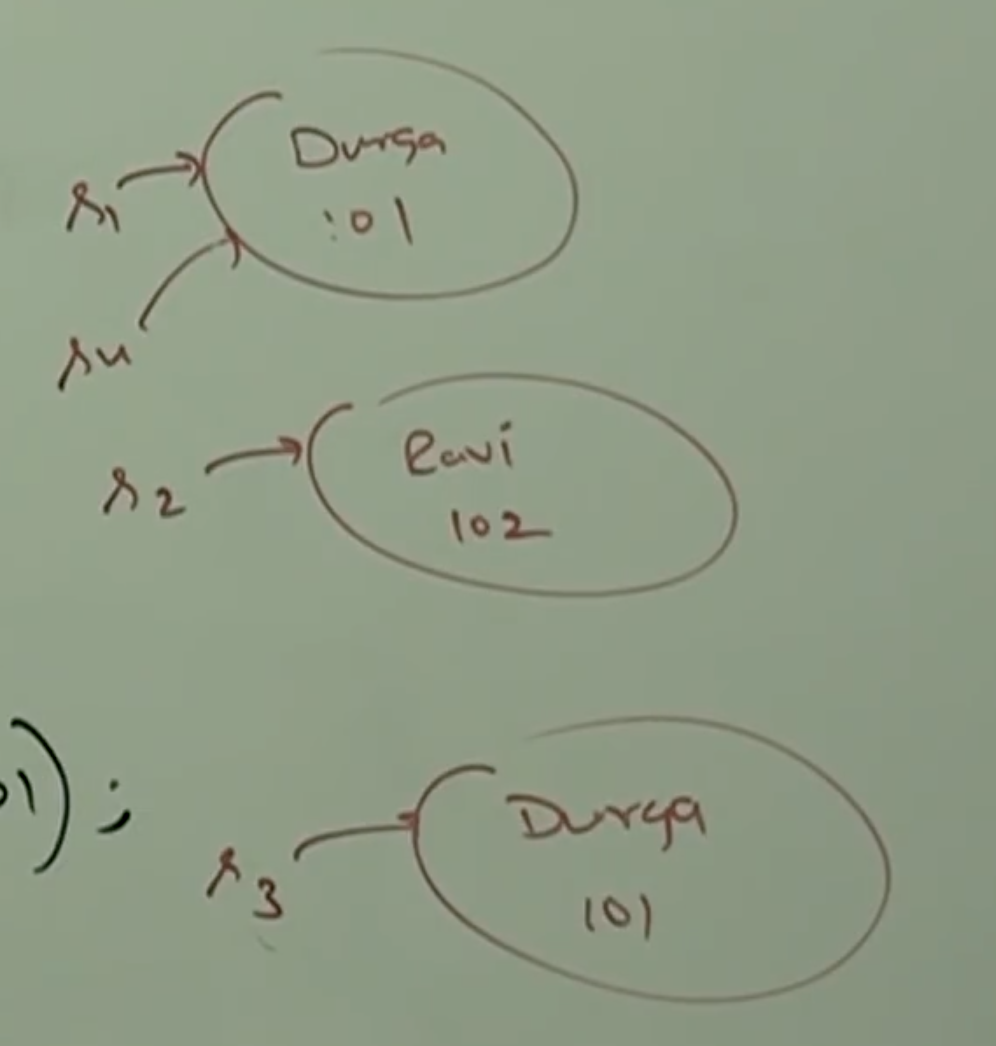
System.out.println(s1.equals(s2)); //false

System.out.println(s1.equals(s3)); //false

System.out.println(s1.equals(s4)); // true

}

}



**Note:**

In the above example, object class equals method got executed which is meant for reference comparison (address comparison) that is if two references pointing to the same object then only .equals() method returns true.

Based on our requirement we can override equals() method for content comparison.

While overriding equals() method for content comparison, we have to take care about the following.

1. What is the meaning of equality. That is (whether we have to check only names or only rollNumber or both).
2. If we are passing different type of object, our equals() method shouldn’t raise ClassCastException. That is, we have to handle ClassCastException to return false.
3. If we are passing null argument then our equals() method shouldn’t raise NullPointerException. That is, we have to handle NullPointerException to return false.

The following is the proper way of overriding equals() method for Student class content comparison.

Example:

public boolean equals(Object obj){

try{

String name1 = this.name;

String rollNo1 = this.rollNo;

Student s = (Student) obj; // Throws CCE

String name2 = s.name; // Throws NPE

String rollNo2 = s.rollNo;

If(name1.equals(name2) && rollNo1 == rollNo2){

return true;

} else{

return false;

}

} catch(ClassCastException cce){

return false;

} catch(NullPointerException npe){

return false;

}

}

s1.equals(s2);

s1 🡪 Current Object, so this.name and this.rollNo;

s2 🡪 The object passed for comparison

Student s1 = new Student(“durga”, 101);

Student s2 = new Student(“ravi”, 102);

Student s3 = new Student(“durga”, 101);

Student s4 = s1;

System.out.println(s1.equals(s2)); //false

System.out.println(s1.equals(s3)); //true

System.out.println(s1.equals(s4)); // true

System.out.println(s1.equals(“durga”)); //false

System.out.println(s1.equals(null)); //false

* **Simplified Version of equals():**

public boolean equals(Object obj){

try{

Student s = Student(obj);

if(name.equals(s.name) && rollno == s.rollno){

return true;

} else{

return false;

}

} catch(ClassCastException cce){

return false;

} catch(NullPointerException npe){

return false;

}

}

* **More simplified version of equals() method:**

public boolean equals(Object obj){

if(obj instanceof Student){

Student s = (Student) obj;

if(name.equals(s.name) && rollno == s.rollno){

return true;

} else{

return false;

}

}

return false;

}

**Note:**

To make above equals() methods more efficient we have to write the following code at the beginning inside equals() method.

if(this == obj)

return true;

According to this, if both references pointing to the same object then without performing any comparison .equals() method returns true directly.

public boolean equals(Object obj){

if(this == obj){

return true;

}

if(obj instanceof Student){

Student s = (Student) obj;

if(name.equals(s.name) && rollno == s.rollno){

return true;

} else{

return false;

}

}

return false;

}

* **equals() of String and StringBuffer class:**

String:

String s1 = new String(“durga”);

String s2 = new String(“durga”);

System.out.println(s1 == s2);

System.out.println(s1.equals(s2));

Note:

In String class .equals() method is overridden for content comparison. Hence, eventhough objects are different if content is same then .equals() returns true.

StringBuffer:

StringBuffer sb1 = new StringBuffer(“durga”);

StringBuffer sb2 = new StringBuffer(“durga”);

System.out.println(sb1 == sb2);

System.out.println(sb1.equals(sb2));

Note:

In StringBuffer .equals() method is not overridden for content comparison. Hence, if objects are different .equals() method returns false, eventhough content is same.