**Object class**

|  |
| --- |
| Object class is the Base class for every class in Java and it is existed in java.lang package. |

**Object class methods**

|  |
| --- |
| public class java.lang.Object  {  public java.lang.Object();  public final native java.lang.Class<?> getClass();  public native int hashCode();  public boolean equals(java.lang.Object);  protected native java.lang.Object clone() throws java.lang.CloneNotSupportedException;  public java.lang.String toString();  public final native void notify();  public final native void notifyAll();  public final void wait() throws java.lang.InterruptedException;  public final native void wait(long) throws java.lang.InterruptedException;  public final void wait(long, int) throws java.lang.InterruptedException;  protected void finalize() throws java.lang.Throwable;  } |

**Calling Base class hashCode() method**

|  |
| --- |
| **package** p1;  **public** **class** One  {  **int** a,b;  One(**int** a,**int** b)  { **this**.a=a;  **this**.b=b;  }  }  **­­**  **package** p1;  **public** **class** Main1  {  **public** **static** **void** main(String[] args)  {  One o1=**new** One(100, 200);  System.***out***.println(o1.hashCode());  }  }  **Output:**  1175962212 |

**hashCode() method of Object class**

|  |
| --- |
| 1. This method returns hash code value which is generated by applying hashing algorithm on the address of the object. |

**toString() method of Object class**

|  |
| --- |
| 1. It is a method which is invoked automatically whenever you print the object. 2. This method returns fully qualified and class name along with @ symbol and hexadecimal format of hashcode value given by hashCode() method. |

**Why we override hashCode() method of Object class in child classes?**

|  |
| --- |
| We override to generate hash code value based on object state not based on object address. |

**Why we override toString() method?**

|  |
| --- |
| We override toString() method to return object state |

**Example on overriding hashCode() method and toString() method**

**One.java**

|  |
| --- |
| **package** p1;  **import** java.util.Objects;  **public** **class** One  {  **int** a,b;  One(**int** a,**int** b)  { **this**.a=a;  **this**.b=b;  }    @Override  **public** **int** hashCode() //o1  {  **int** hv=Objects.*hash*(a, b); //Objects.hash(this.a, this.b);  **return** hv;  }  @Override  **public** String toString()  {  **return** "p1.One[a="+a+",b="+b+"]";  }  } |

**Main1.java**

|  |
| --- |
| **package** p1;  **public** **class** Main1  {  **public** **static** **void** main(String[] args)  {  One o1=**new** One(100, 200);  One o2=**new** One(100, 200);  System.***out***.println("o1 hashcode:\t"+o1.hashCode());  System.***out***.println("o2 hashcode:\t"+o2.hashCode());  **if**(o1.hashCode()==o2.hashCode())  { System.***out***.println("object states are same");  }**else**  { System.***out***.println("object states are different");  }  System.***out***.println("o1 object state:\t"+o1);  System.***out***.println("o2 object state:\t"+o2);  }  }  **Output:**  o1 hashcode: 4261  o2 hashcode: 4261  object states are same  o1 object state: p1.One[a=100,b=200]  o2 object state: p1.One[a=100,b=200] |

What is the use of “public boolean equals (Object obj)” method of Object class

|  |
| --- |
| 1. It compares two objects addresses and returns true if addresses are same otherwise returns false. 2. No need of equals() method of object class because we can do the same by using comparison operator (==). |

**Example on overriding equals() method**

|  |
| --- |
| **package** p1;  **import** java.util.Objects;  **public** **class** One  { **int** a,b;  One(**int** a,**int** b)  { **this**.a=a;  **this**.b=b;  }  @Override  **public** **int** hashCode() {  **return** Objects.*hash*(a, b);  }  @Override  **public** String toString()  { **return** "p1.One[a="+a+",b="+b+"]";  }  @Override  **public** **boolean** equals(Object obj) //o1 here this means o1  {  One other = (One) obj; //down casting  //other=o2  **return** **this**.a == other.a && **this**.b == other.b;  }  }  **Main1.java**  **package** p1;  **public** **class** Main1  {  **public** **static** **void** main(String[] args)  {  One o1=**new** One(100, 200);  One o2=**new** One(100, 200);    System.***out***.println("object state comparision:\t"+ o1.equals(o2) );  System.***out***.println("address comparison:\t"+(o1==o2));  }  }  **Output:**  object state comparision: true  address comparison: false |