HashCode and Equals in an Object

How to write hashcode for an object

**public** **enum** EmpType {

***PERMANENT***,***TEMPORARY***

}

**import** java.util.Arrays;

**public** **class** Employee {

**private** String stringVal;

**private** **byte** byteVal; **C BLISSED FB**

**private** **short** shortVal;

**private** **int** intVal;

**private** **float** floatVal;

**private** **double** doubleVal;

**private** **long** longVal;

**private** EmpType enumType; // Permanent or Temporary

**private** **char** ch;

**private** **byte**[] byteArrVal;

**private** String[] strArrVal;

@Override

**public** **int** hashCode() {

**final** **int** prime = 31;

**int** result = 1;

result = prime \* result + Arrays.*hashCode*(byteArrVal);

result = prime \* result + byteVal;

result = prime \* result + ch;

**long** temp;

temp = Double.*doubleToLongBits*(doubleVal);

result = prime \* result + (**int**) (temp ^ (temp >>> 32));

result = prime \* result + ((enumType == **null**) ? 0 : enumType.hashCode());

result = prime \* result + Float.*floatToIntBits*(floatVal);

result = prime \* result + intVal;

result = prime \* result + (**int**) (longVal ^ (longVal >>> 32));

result = prime \* result + shortVal;

result = prime \* result + Arrays.*hashCode*(strArrVal);

result = prime \* result + ((stringVal == **null**) ? 0 : stringVal.hashCode());

**return** result;

}

@Override

**public** **boolean** equals(Object obj) {

**if** (**this** == obj)

**return** **true**;

**if** (obj == **null**)

**return** **false**;

**if** (getClass() != obj.getClass())

**return** **false**;

Employee other = (Employee) obj;

**if** (!Arrays.*equals*(byteArrVal, other.byteArrVal))

**return** **false**;

**if** (byteVal != other.byteVal)

**return** **false**;

**if** (ch != other.ch)

**return** **false**;

**if** (Double.*doubleToLongBits*(doubleVal) != Double.*doubleToLongBits*(other.doubleVal))

**return** **false**;

**if** (enumType != other.enumType)

**return** **false**;

**if** (Float.*floatToIntBits*(floatVal) != Float.*floatToIntBits*(other.floatVal))

**return** **false**;

**if** (intVal != other.intVal)

**return** **false**;

**if** (longVal != other.longVal)

**return** **false**;

**if** (shortVal != other.shortVal)

**return** **false**;

**if** (!Arrays.*equals*(strArrVal, other.strArrVal))

**return** **false**;

**if** (stringVal == **null**) {

**if** (other.stringVal != **null**)

**return** **false**;

} **else** **if** (!stringVal.equals(other.stringVal))

**return** **false**;

**return** **true**;

}

}

# **Key points for Hashcode**

**final int prime = 31;**

**int result = 1;**

**Arrays.*hashCode*(byteArrVal); //For byte[]**

**prime \* result + ch; // For char type**

//For double type value

**long temp = Double.*doubleToLongBits*(doubleVal);**

**result = prime \* result + (int) (temp ^ (temp >>> 32));**

//For enum

**result = prime \* result + ((enumType == null) ? 0 : enumType.hashCode());**

**Always remember an enum gives a random value for hashcode.**

//For float type value

**result = prime \* result + Float.*floatToIntBits*(floatVal);**

//For long type value

**result = prime \* result + (int) (longVal ^ (longVal >>> 32));**

//For short type value

**result = prime \* result + shortVal;**

//For String[]

**Arrays.*hashCode*(strArrVal);**

//For String

**result = prime \* result + ((stringVal == null) ? 0 : stringVal.hashCode());**

**//For Boolean**

**result = prime \* result + (boolean ? 1231 : 1237);**

# **Key Points for equals()**

**public** **boolean** equals(Object obj) {

**Always do the first operation**

1. **if (this == obj) return true;**
2. **if (obj == null) return false;**
3. **if (getClass() != obj.getClass()) return false;**

**Second try to compare each field**

1. Employee other = (Employee) obj;
2. For byte[]

**if (!Arrays.equals(byteArrVal, other.byteArrVal)) return false;**

1. For byte value

**if (byteVal != other.byteVal) return false;**

1. For char type value

**if (ch != other.ch) return false;**

1. For double type value

**if(Double.doubleToLongBits(doubleVal) != Double.doubleToLongBits(other.doubleVal))**

**return false;**

1. For float type value

**if(Float.floatToIntBits(floatVal) != Float.floatToIntBits(other.floatVal))**

**return false;**

1. For enum type

**if (enumType != other.enumType) return false;**

1. For int type value

**if (intVal != other.intVal) return false;**

1. For long type value

**if (longVal != other.longVal) return false;**

1. For short type value

**if (shortVal != other.shortVal) return false;**

1. For String[] type

**if (!Arrays.equals(strArrVal, other.strArrVal)) return false;**

1. For normal String value

**if (stringVal == null) {**

**if (other.stringVal != null)**

**return false;**

**} else if (!stringVal.equals(other.stringVal))**

**return false;**

**Points to Remember**

For Any other Array, use the below

**if (!Arrays.equals(ArrVal, other.ArrVal)) return false;**

For Double type value

**if (Double.doubleToLongBits(doubleVal) != Double.doubleToLongBits(other.doubleVal))**

**return false;**

For float type value

**if(Float.floatToIntBits(floatVal) != Float.floatToIntBits(other.floatVal))**

**return false;**

For char type value

**if (ch != other.ch) return false;**

For enum type value

**if (enumType != other.enumType) return false;**

Reference

The hashcode and equals generated by Intellij Idea is given below for the above class.

**public int** hashCode() {  
 **int** result;  
 **long** temp;  
 result = **stringVal** != **null** ? **stringVal**.hashCode() : 0;  
 result = 31 \* result + (**int**) **byteVal**;  
 result = 31 \* result + (**int**) **shortVal**;  
 result = 31 \* result + **intVal**;  
 result = 31 \* result + (**floatVal** != +0.0f ? Float.*floatToIntBits*(**floatVal**) : 0);  
 temp = Double.*doubleToLongBits*(**doubleVal**);  
 result = 31 \* result + (**int**) (temp ^ (temp >>> 32));  
 result = 31 \* result + (**int**) (**longVal** ^ (**longVal** >>> 32));  
 result = 31 \* result + (**enumType** != **null** ? **enumType**.hashCode() : 0);  
 result = 31 \* result + (**int**) **ch**;  
 result = 31 \* result + (**byteArrVal** != **null** ? Arrays.*hashCode*(**byteArrVal**) : 0);  
 result = 31 \* result + (**strArrVal** != **null** ? Arrays.*hashCode*(**strArrVal**) : 0);  
 **return** result;  
}

**public boolean** equals(Object o) {  
 **if** (**this** == o) **return true**;  
 **if** (o == **null** || getClass() != o.getClass()) **return false**;  
  
 Employee employee = (Employee) o;  
  
 **if** (**byteVal** != employee.**byteVal**) **return false**;  
 **if** (**shortVal** != employee.**shortVal**) **return false**;  
 **if** (**intVal** != employee.**intVal**) **return false**;  
 **if** (Float.*compare*(employee.**floatVal**, **floatVal**) != 0) **return false**;  
 **if** (Double.*compare*(employee.**doubleVal**, **doubleVal**) != 0) **return false**;  
 **if** (**longVal** != employee.**longVal**) **return false**;  
 **if** (**ch** != employee.**ch**) **return false**;  
 **if** (**stringVal** != **null** ? !**stringVal**.equals(employee.**stringVal**) : employee.**stringVal** != **null**) **return false**;  
 **if** (**enumType** != employee.**enumType**) **return false**;  
 **if** (!Arrays.*equals*(**byteArrVal**, employee.**byteArrVal**)) **return false**;  
 *// Probably incorrect - comparing Object[] arrays with Arrays.equals* **return** Arrays.*equals*(**strArrVal**, employee.**strArrVal**);  
  
}

**Example on Negative HashCode**

**class Emp {  
  
 private int sal = Integer.*MAX\_VALUE*;*//It can be MIN\_VALUE* private String name;  
  
 public Emp( String name ) {  
 this.name = name;  
 }**

**}**

**It will give negative hashcode as shown below.**

Emp emp = **new** Emp(**"a"**);  
System.***out***.print(**"Emp HashCode :::"**+emp.hashCode());