128 == 128 -> false

128 outside cached range

Java create two separate Integer cached Objects

x = y is false, point to the DIFFERENT OBJECT

1==1 -> true

1 inside cached range

Java reuse the Integer cached object.

x = y is true, point to the SAME OBJECT

**Reasons about why:**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

== is to compare object references, see whether if two variables point to the same object in Memory location.

(Object locates different memory location. Different objects means different location)

.equals() compares the values inside those objects.

Java caches from -128 to 127. Outside this range must be different objects comparison.

This can help save memory.

You can even customize this range by setting the JVM argument -XX:AutoBoxCacheMax=size, but by default, it’s fixed at 127.

So , when compare -128 to 127 , it can use ==

But when out of range, it need .equals()

You can use System.identityHashCode() to see the memory address’s hash code.

Int a = 128

Int b = 128

Println(System.identityHashCode(a)) 212628335

Println(System.identityHashCode(b)) 2111991224

Different !!

Int c = 1

Int d = 2

Println(System.identityHashCode(c)) 292938459

Println(System.identityHashCode(d)) 292938459

Same !!

Reference:

https://medium.com/@tecnicorabi/why-1-1-is-true-but-128-128-is-false-in-java-4ea544e83eef