Life hacks: how to write code better and faster

1. Expressions vs statements

In Java, it is helpful to distinguish between two categories: *statements* and *expressions*. A statement is usually said to be *executed*, while an expression is said to be *evaluated*. But that's not the most important thing.

The main difference between a statement and an expression is that evaluating an expression has a *result*. And this result has a type, and it can be assigned to a variable or used in some other expression.

Examples:

Code	**Notes**
`int x;`	Statement
`(a < 10)`	Expression whose type is `boolean`
`i++;`	Expression whose type is the same as the type of the `i` variable
`x = 5;`	Expression whose type is the same as the type of the `x` variable

And what does this give us?

First, we can take advantage of the fact that many statements are actually expressions (meaning they evaluate to a value). For example, code like this will work:

Code	**Notes**

`int x, y, z;`
`x = y = z = 1;`
	`int x, y, z;`
	`x = (y = (z = 1))`

Second, if we want, we can ignore the result of evaluating an expression.

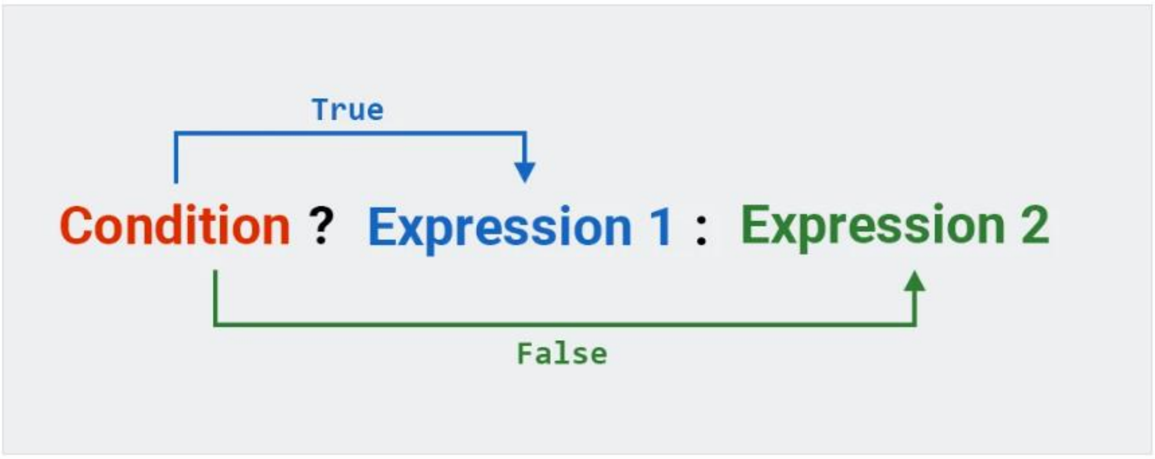
Code	**Code where we ignore the result:**

`int x = scanner.nextInt();`
`boolean m = (5 < 10);`
	`scanner.nextInt();`
	`(5 < 10);`

We ignore the result of evaluating an expression, for example, if the expression involves doing something useful, and this action is what is important to us, not the result itself.

---

2. Ternary operator



This life hack is already more interesting than the previous one. Java has a special *ternary operator*. Its syntax is somewhat similar to the syntax for the `if-else` statement:

```
Condition ? Expression 1 : Expression 2;
```

If *condition* is true, then *Expression 1* is evaluated, otherwise *Expression 2* is evaluated. The condition is followed by a *question mark*, and the two expressions are separated by *colon*.

The main difference between the *ternary operator* and an `if-else` statement is that the ternary operator is an expression, which means we can assign its result to something.

For example, suppose we want to calculate the minimum of two numbers. Using the ternary operator, this code would look like this:

```
int a = 2;
int b = 3;
int min = a < b ?  a : b;
```

Or, let's say you need to assign different values to a variable depending on some condition. How do you do that?

One option is to use an `if-else` statement:

```
int age = 25;
int money;
if (age > 30)
   money = 100;
else
   money = 50;
```

The second option is to use the *ternary operator*, that is, shorthand for the `if-else` statement:

```
int age = 25;
int money = age > 30 ? 100 : 50;
```

So which is better to use — an `if-else` statement or the *ternary operator*? In terms of execution speed, there isn't much difference. This is more a matter of code readability. And this is a very important point: the code must not only work correctly, but also be easy for other programmers to read.

The simplest rule is this: if the code fits on one line, then use the *ternary operator*; but if it does not fit on one line, then it is better to use an `if-else` statement.

3. Comparing real numbers

As mentioned earlier, you can't just grab real numbers and compare them. There is always the possibility that some significant digits may be discarded, causing unexpected side effects.

That's why there is a time-tested approach. If two real numbers differ by a very small value, then they can be considered to be equal. Example:

```java
double a = 1.000001;
double b = 1.000002;
if ( (b - a) < 0.0001 )
   System.out.println("The numbers are equal");
else
   System.out.println("The numbers are not equal");
```

But that's not all we have to worry about, since the difference between the numbers may well turn out to be negative. So for this approach to work, you need to compare not just the difference between the numbers, but the absolute value of the difference between the numbers: `|a-b|`

Java has a method for calculating the absolute value of a number: `Math.abs()`:

```java
int m = Math.abs(value);
```

As a result, the corrected version of our example above will look like this:

```java
double a = 1.000001;
double b = 1.000002;

if ( Math.abs(b - a) < 0.0001 )
   System.out.println("The numbers are equal");
else
   System.out.println("The numbers are not equal");
```