CODEGYM

Mentor-supported training program from Codegym

Java developer in 12 months

MODULE 2. JAVA CORE

Lesson #7

Casting

CODEGYM

Lesson plan

- instanceof operator
- casting
- switch expression

MODULE 2. JAVA CORE

CODEGYM

instanceof operator

instanceof checks whether an object is and object of a specific class.

It is used in this form: «object» instanceof «class».

Code	Description
`Object o = new Integer(3);` `boolean isInt = o instanceof Integer;`	IsInt will be true. The object referred to by variable o is an object of the Integer class.
`Object o = "Java";` `boolean isInt = o instanceof Integer;`	IsInt will be false. The object referred to by variable o is not of the Integer class, it's an object of the String class.

CODEGYM

instanceof operator respects inheritance and interfaces

instanceof is used as: a instanceof B.

The instanceof operator will return true if:

1. Variable **a** stores a reference of type **B**.
2. Variable **a** stores a reference to an object whose class is inherited from **B**.
3. Variable **a** stores a reference to an object which implements interface **B**.

Otherwise instanceof will return false.

CODEGYM

Casting

The class contains all the methods of the class from which it was inherited, which means that an object of this class can be stored in a variable of any of its parent types.

If, as a result of the assignment, we move up the inheritance chain (to the Object type), then this is known as widening of the type (also known as upward conversion or upcasting), and if we move down to the type of an object, then this is narrowing of the type (also known as downward conversion or downcasting).

Moving up the inheritance chain is called widening because it leads to a more general type. But at the same time, the ability to call methods that were added to the class during inheritance is lost.

When narrowing the type, we need to use the type conversion operator, that is, we perform an explicit conversion.

CODEGYM

Switch expression

Switch statement

Java has 2 switch variants — switch statement and switch expression.

Switch expression is available from Java 14, having two previous modifications in versions 12 and 13.

Here's how the good old switch looks prior to Java 12:

```
switch (product) {
        case "Apple":
            productType = "Fruit";
            break;
        case "Peach":
            productType = "Fruit";
            break;
        case "Raspberry":
            productType = "Berry";
            break;
        default:
            productType = "other";
            break;
    }
```

CODEGYM

Switch expression

Using the arrow syntax from functional programming makes it possible to return values without the break keyword, and in general, the result of switch execution can now be saved to a variable or returned via return.

```
return switch (product) {
        case "Apple", "Peach" -> "Fruit";
        case "Raspberry" -> "Berry";
        default -> "other";
    };
```

CODEGYM

In cases where you need to return not only a result, but also have several lines of code, switch will look as follows:

```
…
        switch (product) {
        case "Apple", "Peach" -> {
            System.out.println("This is fruit");
            break "Fruit";
        }
…
```

CODEGYM

The yield keyword is the only “upgrade” to switch in Java 13 — it replaced the break keyword.

```
…
switch (product) {
        case "Apple", "Peach" -> {
            System.out.println("This is fruit");
            yield "Fruit";
        }
…
```

The main difference between yield and break is that break returned control from a case — a command, but yield returns the result of the entire switch, playing the role of an internal return-a.

CODEGYM

instanceof in Java 14

Instanceof operator changed in Java 14 and now it can be used like this:

```
if(o instanceof String s) {
        s.toLowerCase();
    }
```

This came to replace the old and not a very pretty version, where in addition to checking for instanceof, it is also necessary to cast the variable to a specific type.

```
if(s instanceof String) {
        ((String) s).toLowerCase();
    }
```

CODEGYM

Homework

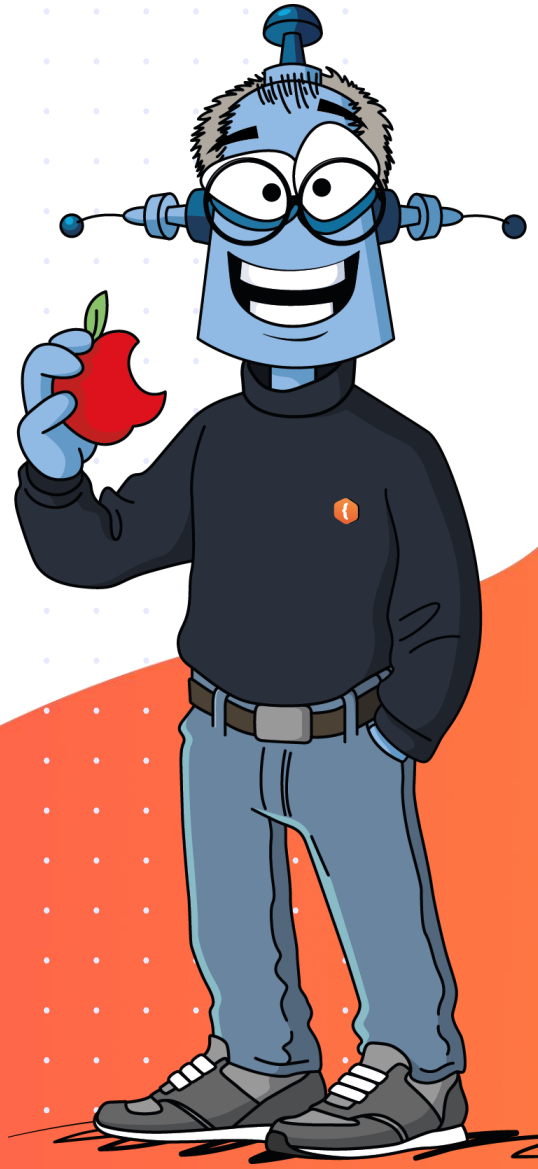
Module 2

Level 7 Casting

MODULE 2. JAVA CORE

CODEGYM

Questions and Answers



MODULE 2. JAVA CORE