

Binary operator

This lesson explores the Binary operator functional interface in Java.

We'll cover the following ^

- BinaryOperator
 - Example

BinaryOperator<T>

`BinaryOperator<T>` is a functional interface that inherits from `BiFunction<T, T, T>` interface. The `BinaryOperator<T>` interface takes only one parameter as compared to `BiFunction<T, T, T>`, which takes three parameters(two operands type and one result type).

Both the input objects and the result are of the same type in `BinaryOperator<T>`.

Below are the few interfaces that come under the `BinaryOperator<T>` category.

<code>BinaryOperator<T></code>	Represents an operation upon two operands of the same type, producing a result of the same type as the operands (reference type)	<code>T apply(T t, T u)</code>
<code>DoubleBinaryOperator</code>	Accepts two double-value operands and produces a double-value result	<code>double applyAsDouble(double left, double right)</code>
<code>IntBinaryOperator</code>	Accepts two int-value	<code>int applyAsInt(int</code>

<code>IntBinaryOperator</code>	Accepts two int-value operands and produces an int-value result	<code>int applyAsInt(int left, int right)</code>
<code>LongBinaryOperator</code>	Accepts two long-value operands and produces a long-value result.	<code>long applyAsLong(long left, long right)</code>

Example

```
import java.util.function.BinaryOperator;

public class BinaryOperatorDemo {

    public static void main(String args[]) {
        Person person1 = new Person("Alex", 23);
        Person person2 = new Person("Daniel", 56);
        BinaryOperator<Person> operator = (p1, p2) -> {
            p1.name = p2.name;
            p1.age = p2.age;
            return p1;
        };

        operator.apply(person1, person2);
        System.out.println("Person Name: " + person1.getName() + " Person Age: " + person1.getAge());
    }
}

class Person {
    String name;
    int age;

    Person() {
    }

    Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }
}
```

```
return age;
```

```
}
```

```
}
```



1

What is the purpose of the `BooleanSupplier` function interface? Choose all that apply.

2

What is the return type of a lambda expression?



We need to override which `Predicate` method in Java 8?

[Retake Quiz](#)

In the next lesson, we will discuss the capturing lambda expressions .