In Java 8 streams the intermediate operations are not evaluated until a terminal operation is invoked.

Each intermediate operation creates a new stream, stores the provided operation/function, and returns the new stream. When a terminal operation is called, the traversal of streams begins and the associated function is performed one by one on each element.

Lazy evaluation example

Let's look at an example to understand how lazy evaluation helps immensely. In the below example, we have a list of 20 integers. We need to get the first number we encounter that is greater than 5 and is divisible by 3 in this list.

```
import java.util.Arrays;
import java.util.List;
import java.util.Optional;

public class LazyEvaluationDemo {

    public static void main(String args[]) {
        List<Integer> data = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,

        Optional<Integer> number = data.stream()
                .filter(num -> num > 5)
                .filter(num -> num % 3 == 0)
                .findFirst();

        System.out.println(number.get());
    }
}
```

Run Save Reset

The terminal operation in the above example is `findFirst()`. This means we only need one element which matches all the given criteria in the `filter()` method.

Let's see how this is evaluated.

1. Pick the first element, i.e., 1. It is less than 5.
2. Pick the second element, i.e., 2. It is less than 5.
3. Pick the third element, i.e., 3. It is less than 5.
4. Pick the fourth element, i.e., 4. It is less than 5.
5. Pick the fifth element, i.e., 5. It is equal to 5.
6. Pick the sixth element, i.e., 6. It is greater than 5 so move to the next operation which is again a filter.
7. Six is divisible by 3, so move to the next operation which is `findFirst`. Hence, 6 is returned as the result.

Let's add some print statements to the above example and see how the elements are processed.

```
import java.util.Arrays;
import java.util.List;
import java.util.Optional;

public class LazyEvaluationDemo {

    public static void main(String args[]) {
        List<Integer> data = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,

        Optional<Integer> number = data.stream()
                .filter(num -> {
                    System.out.println("Processing first filter for number " + num);
                    return num > 5;
                })
                .filter(num -> {
                    System.out.println("Processing second filter for number " + num);
                    return num % 3 == 0;
                })
                .findFirst();

        System.out.println(number.get());
    }
}
```

Run Save Reset