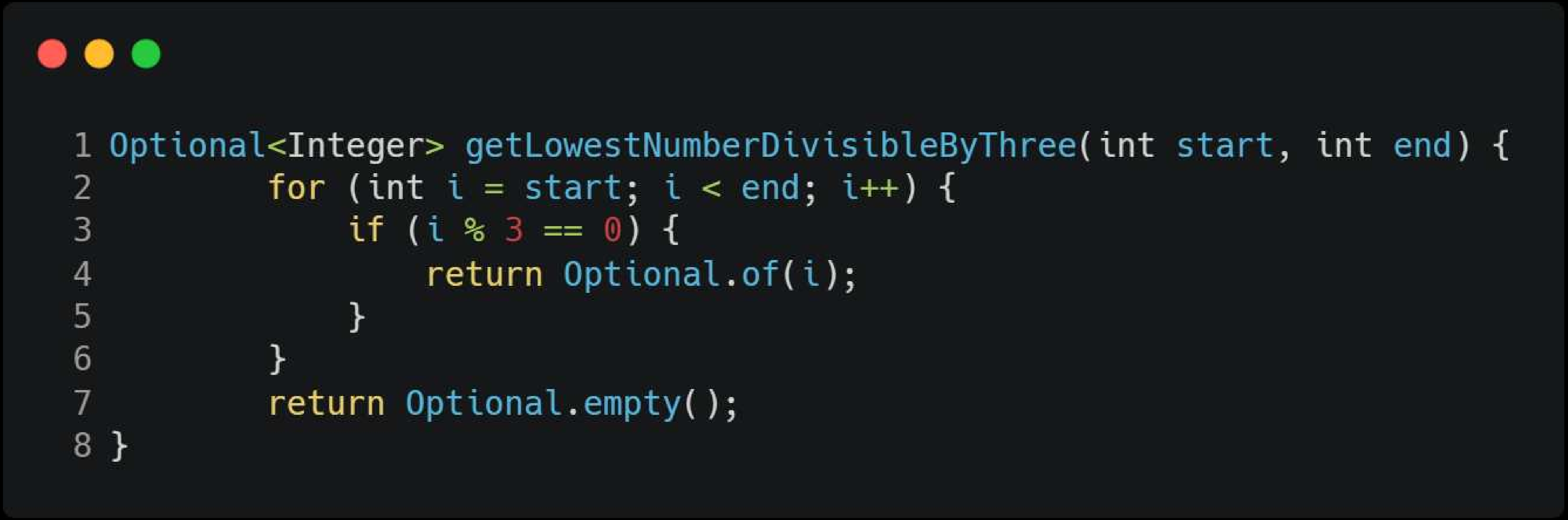
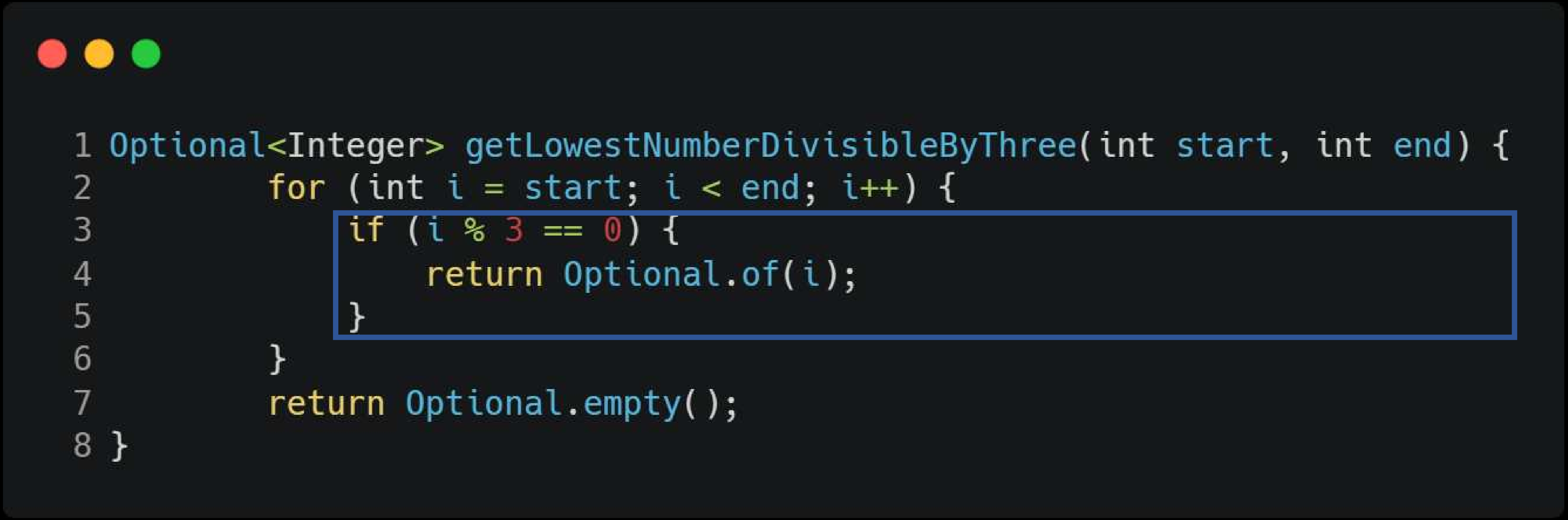
Object Calisthenics
-
Applying it blindly and seeing what happens

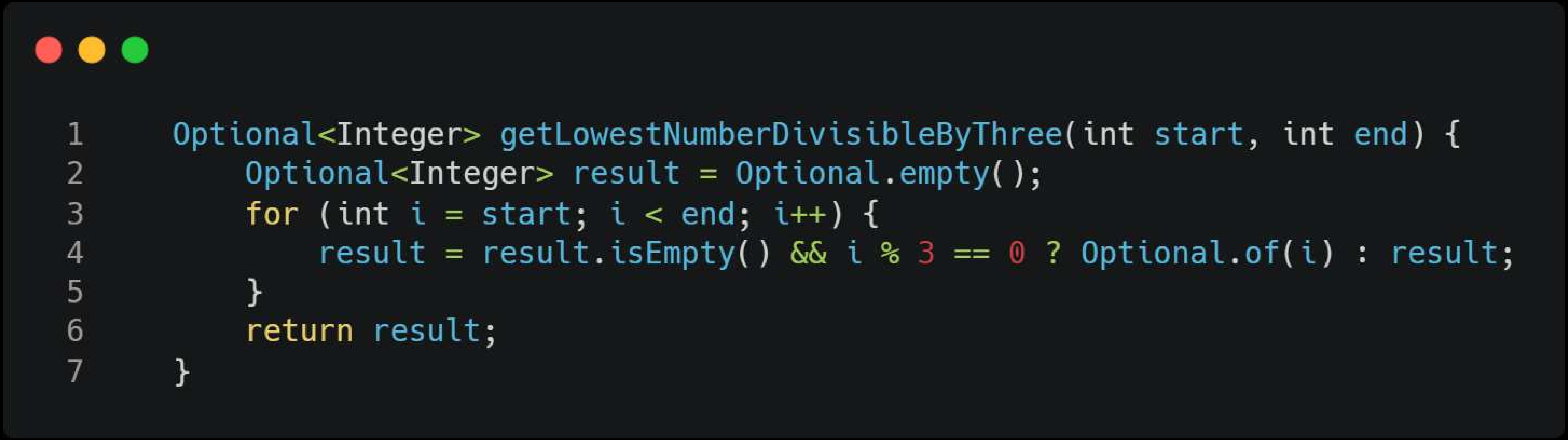


```java
Optional<Integer> getLowestNumberDivisibleByThree(int start, int end) {
        for (int i = start; i < end; i++) {
            if (i % 3 == 0) {
                return Optional.of(i);
            }
        }
        return Optional.empty();
}
```

Rule 1
-
One level of indentation per method

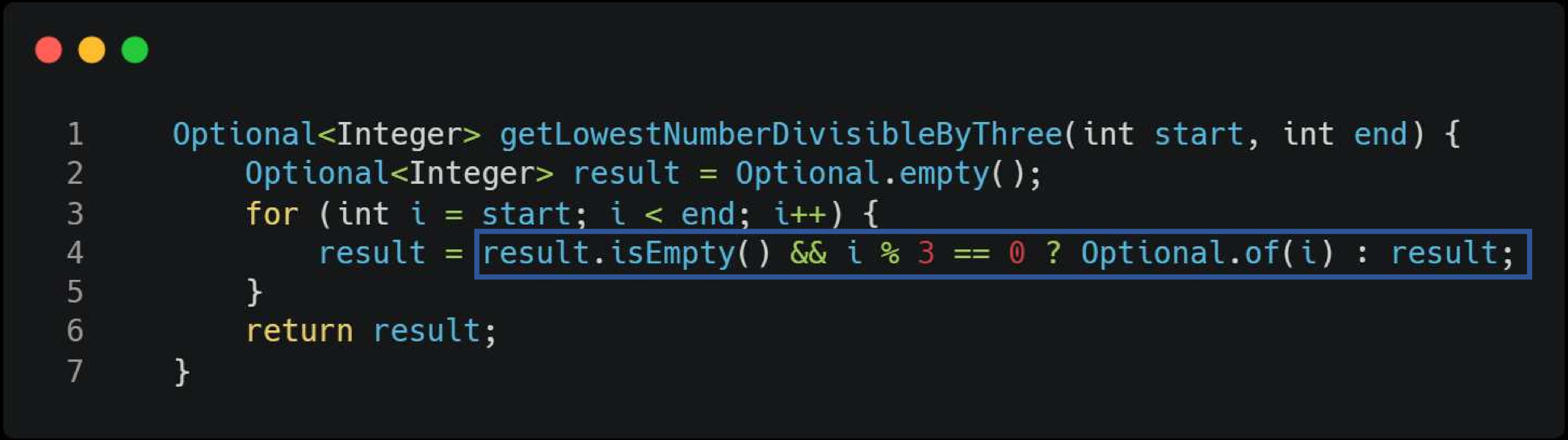


```java
Optional<Integer> getLowestNumberDivisibleByThree(int start, int end) {
        for (int i = start; i < end; i++) {
            if (i % 3 == 0) {
                return Optional.of(i);
            }
        }
        return Optional.empty();
}
```

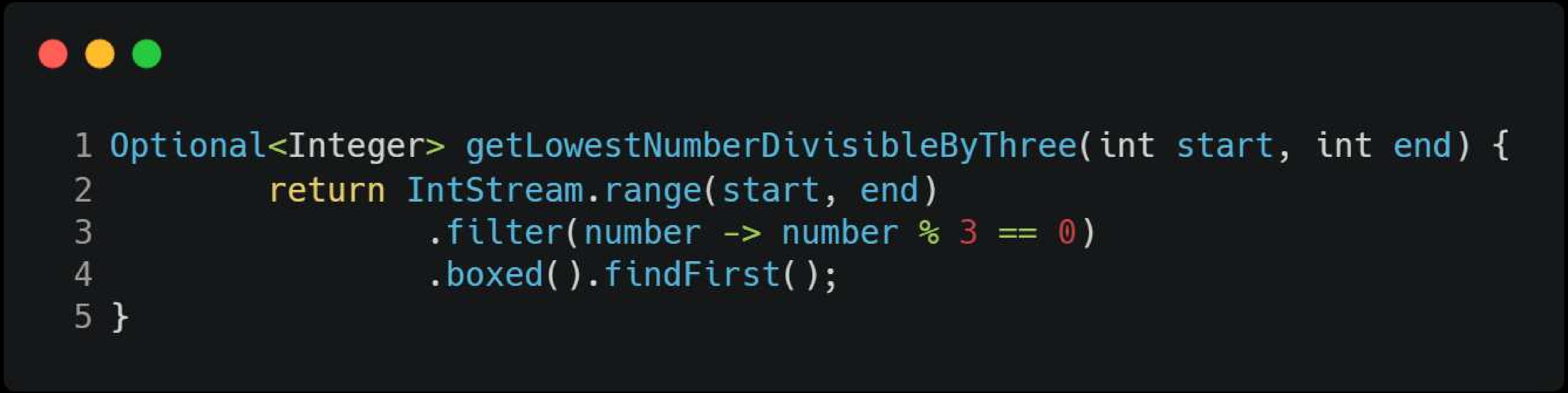


```java
Optional<Integer> getLowestNumberDivisibleByThree(int start, int end) {
    Optional<Integer> result = Optional.empty();
    for (int i = start; i < end; i++) {
        result = result.isEmpty() && i % 3 == 0 ? Optional.of(i) : result;
    }
    return result;
}
```

Rule 2
-
Don't use the ELSE keyword



```java
Optional<Integer> getLowestNumberDivisibleByThree(int start, int end) {
    Optional<Integer> result = Optional.empty();
    for (int i = start; i < end; i++) {
        result = result.isEmpty() && i % 3 == 0 ? Optional.of(i) : result;
    }
    return result;
}
```

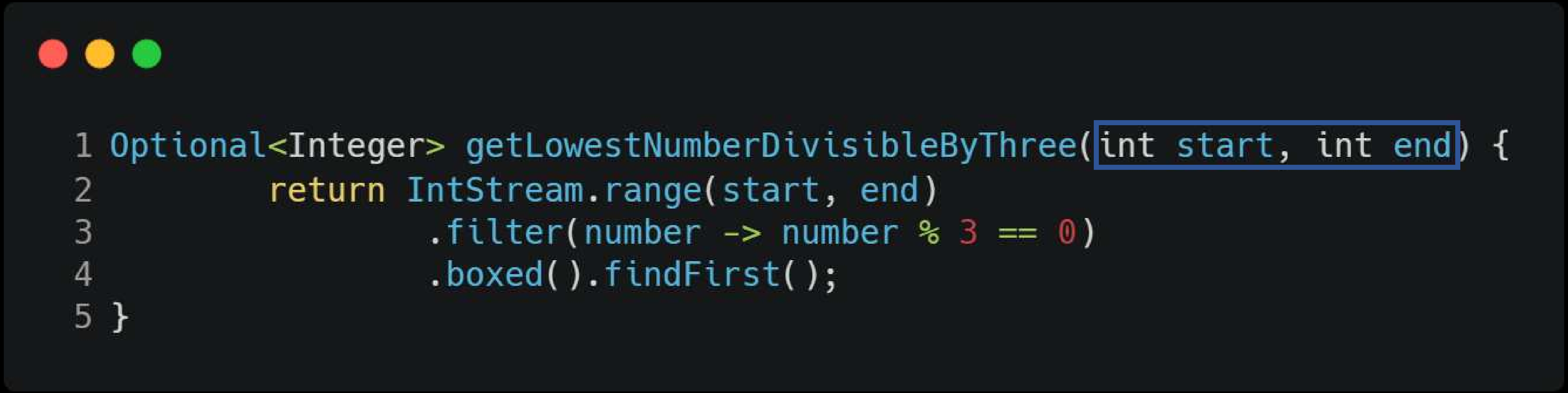


```java
Optional<Integer> getLowestNumberDivisibleByThree(int start, int end) {
        return IntStream.range(start, end)
                .filter(number -> number % 3 == 0)
                .boxed().findFirst();
}
```

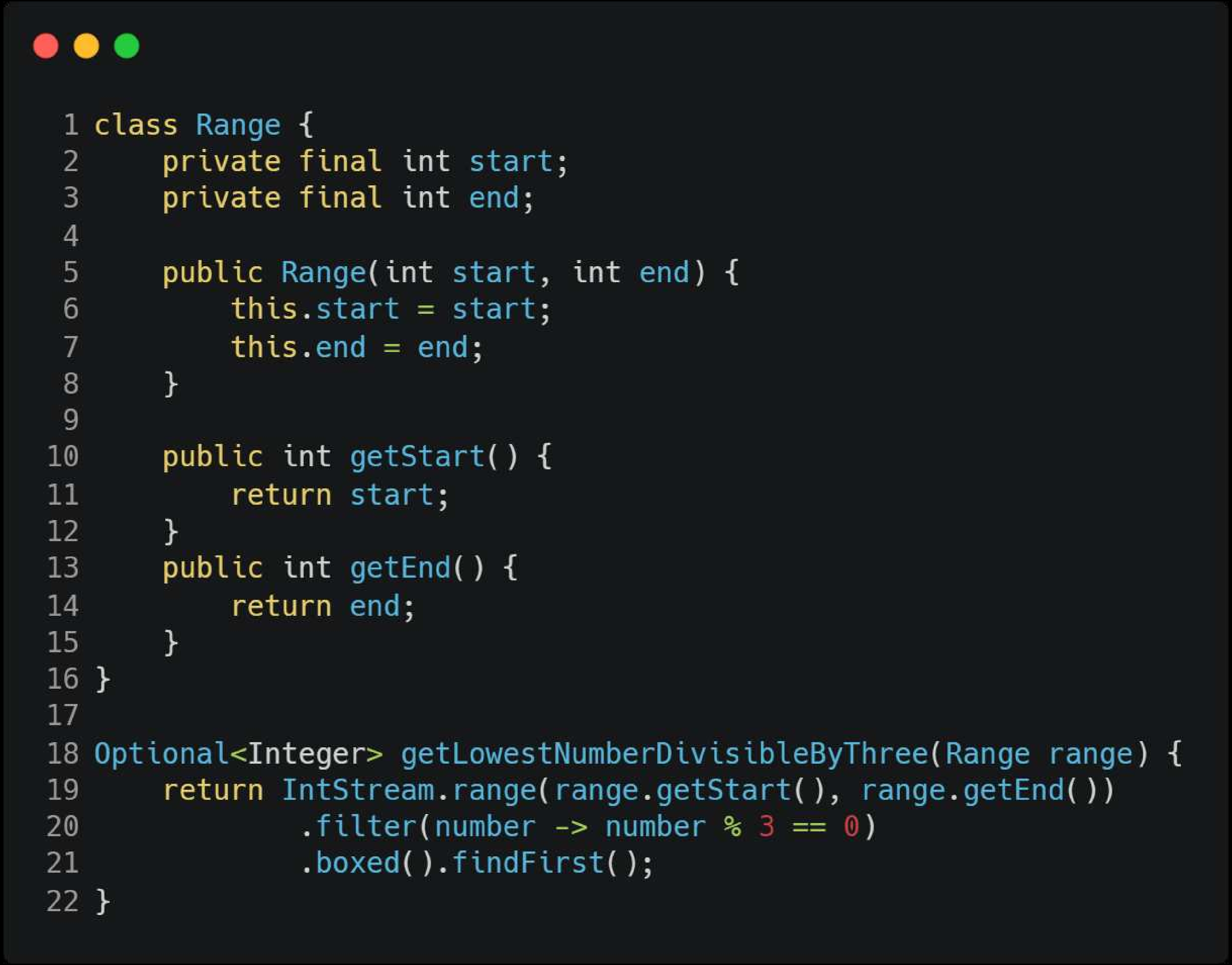
Rule 3

-

Wrap all primitives and Strings in classes



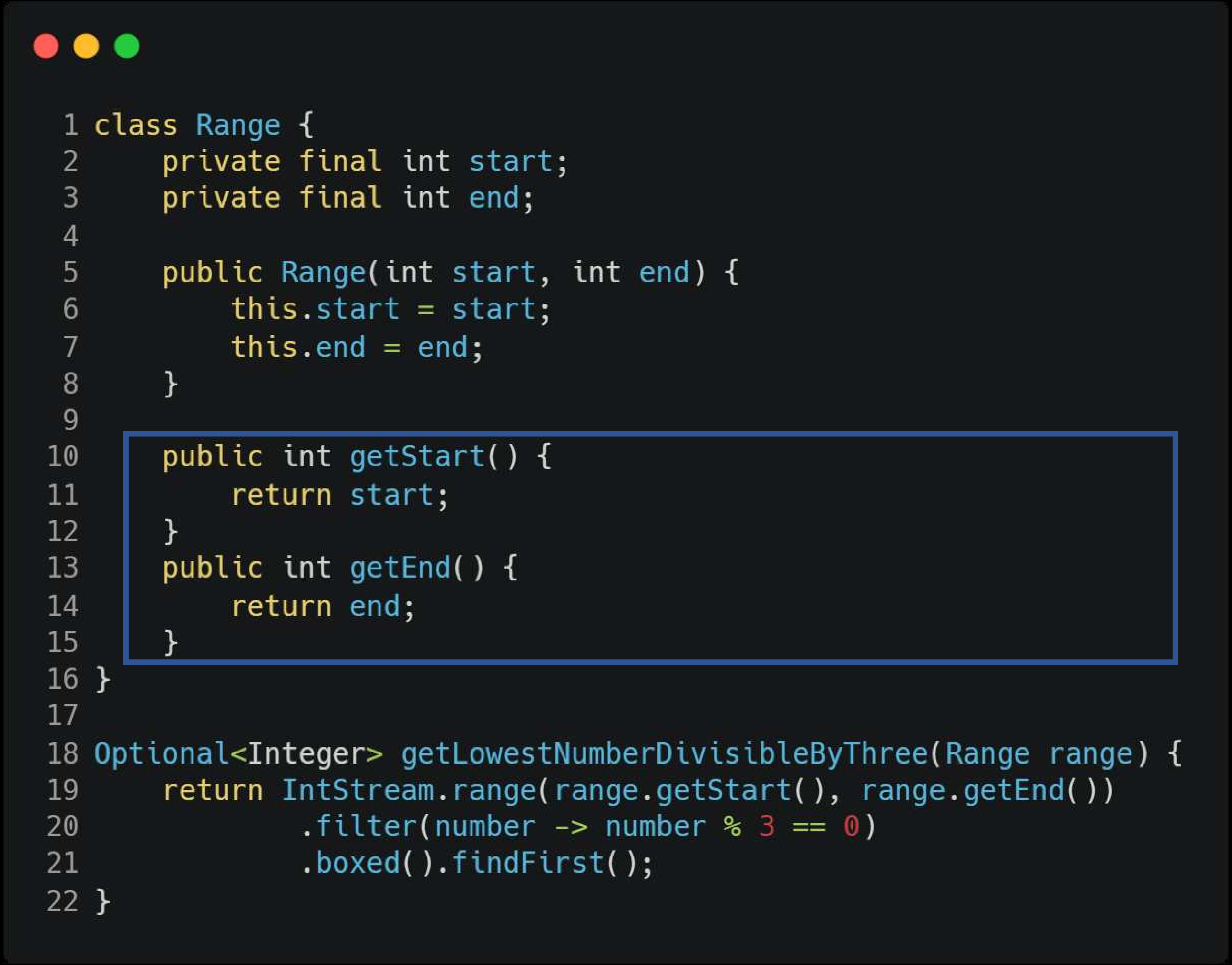
```java
Optional<Integer> getLowestNumberDivisibleByThree(int start, int end) {
        return IntStream.range(start, end)
                .filter(number -> number % 3 == 0)
                .boxed().findFirst();
}
```



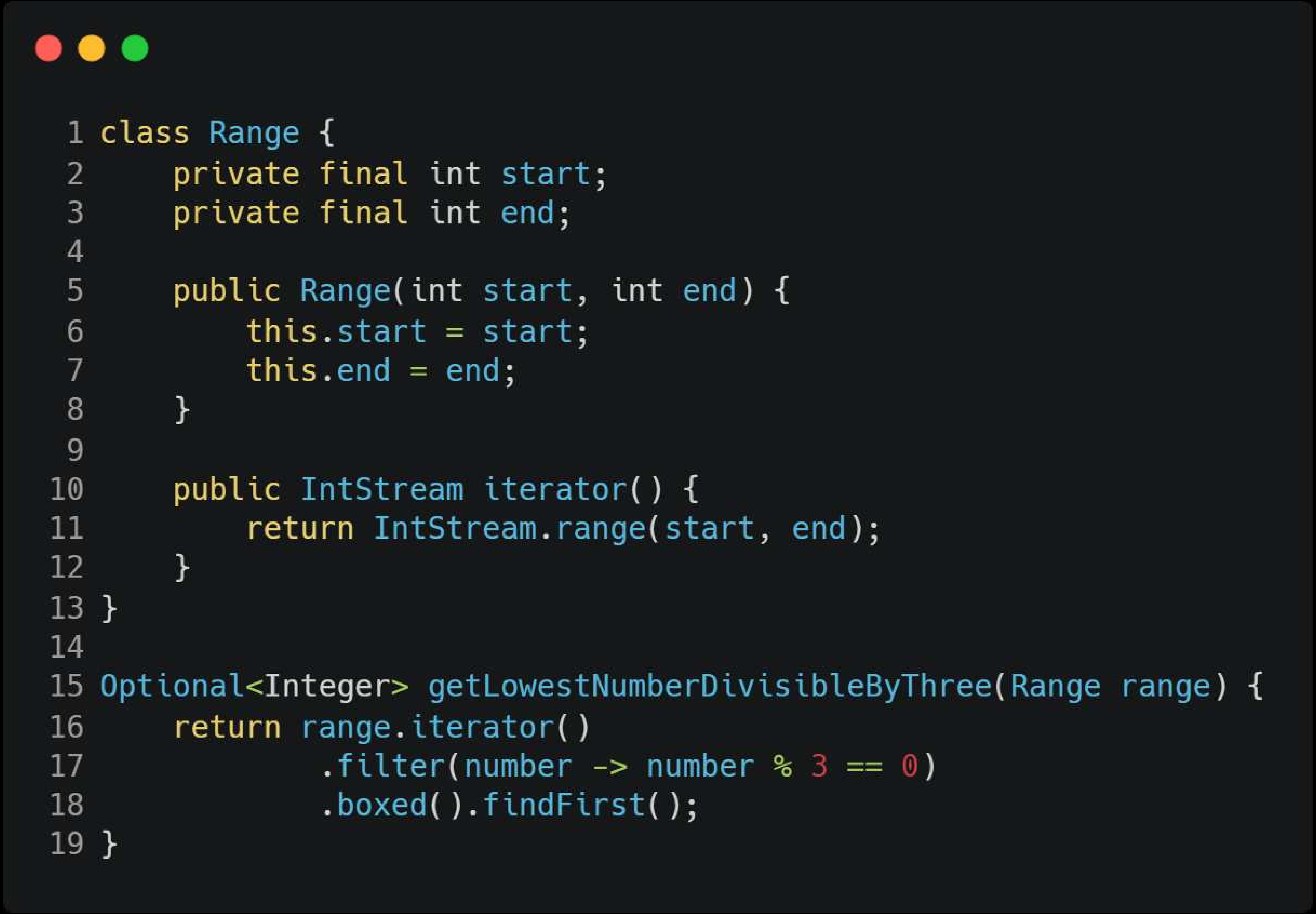
```java
class Range {
    private final int start;
    private final int end;

    public Range(int start, int end) {
        this.start = start;
        this.end = end;
    }

    public int getStart() {
        return start;
    }
    public int getEnd() {
        return end;
    }
}

Optional<Integer> getLowestNumberDivisibleByThree(Range range) {
    return IntStream.range(range.getStart(), range.getEnd())
            .filter(number -> number % 3 == 0)
            .boxed().findFirst();
}
```

4. First class collections.
5. One dot per line.
6. Don't abbreviate.
7. Keep all classes less than 50 lines.
8. No classes with more than two instance variables.

Rule 9
-
No getters or setters.



```java
class Range {
    private final int start;
    private final int end;

    public Range(int start, int end) {
        this.start = start;
        this.end = end;
    }

    public int getStart() {
        return start;
    }
    public int getEnd() {
        return end;
    }
}

Optional<Integer> getLowestNumberDivisibleByThree(Range range) {
    return IntStream.range(range.getStart(), range.getEnd())
            .filter(number -> number % 3 == 0)
            .boxed().findFirst();
}
```

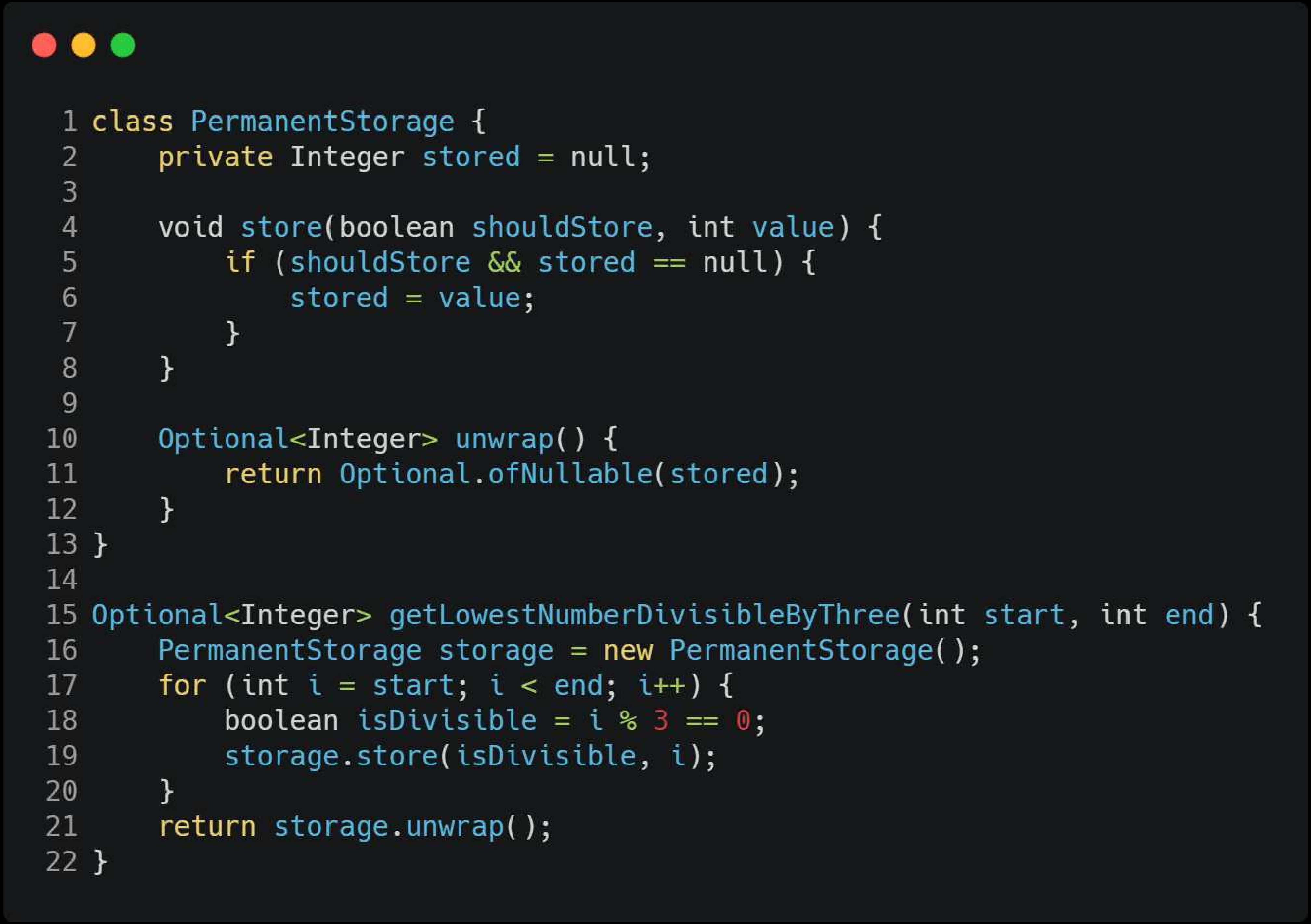


```java
class Range {
    private final int start;
    private final int end;

    public Range(int start, int end) {
        this.start = start;
        this.end = end;
    }

    public IntStream iterator() {
        return IntStream.range(start, end);
    }
}

Optional<Integer> getLowestNumberDivisibleByThree(Range range) {
    return range.iterator()
            .filter(number -> number % 3 == 0)
            .boxed().findFirst();
}
```

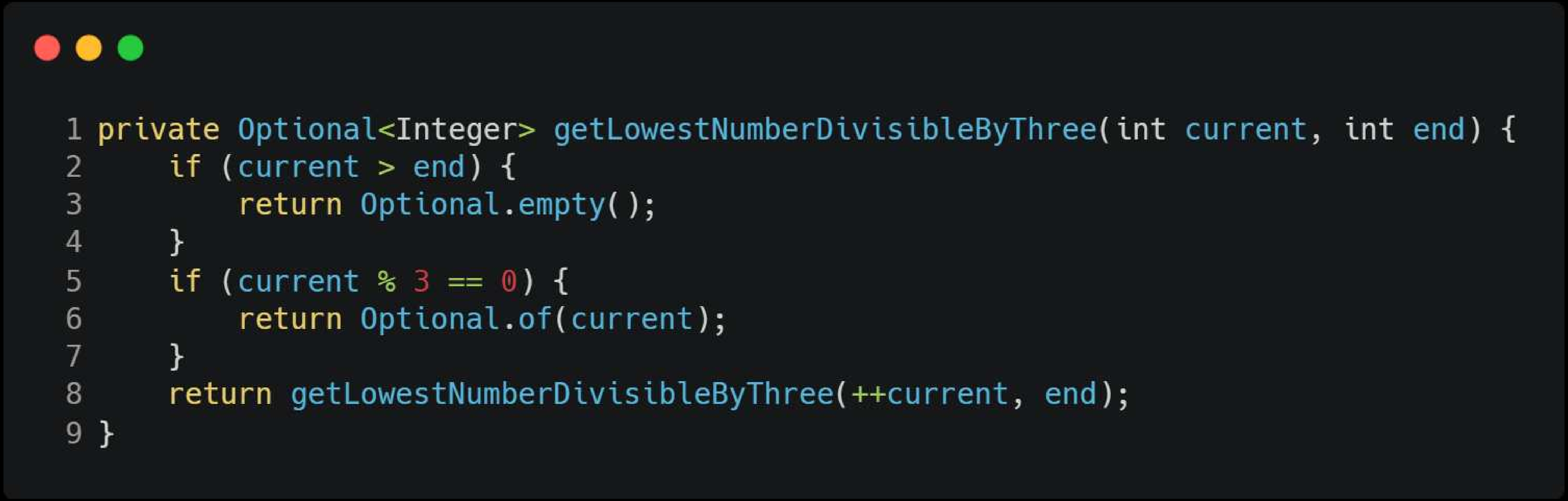
Other Solutions

-

but not better ones



```java
class PermanentStorage {
    private Integer stored = null;

    void store(boolean shouldStore, int value) {
        if (shouldStore && stored == null) {
            stored = value;
        }
    }

    Optional<Integer> unwrap() {
        return Optional.ofNullable(stored);
    }
}

Optional<Integer> getLowestNumberDivisibleByThree(int start, int end) {
    PermanentStorage storage = new PermanentStorage();
    for (int i = start; i < end; i++) {
        boolean isDivisible = i % 3 == 0;
        storage.store(isDivisible, i);
    }
    return storage.unwrap();
}
```



```java
private Optional<Integer> getLowestNumberDivisibleByThree(int current, int end) {
    if (current > end) {
        return Optional.empty();
    }
    if (current % 3 == 0) {
        return Optional.of(current);
    }
    return getLowestNumberDivisibleByThree(++current, end);
}
```

Feelings?

-

good and bad ones

Rule 8

-

No classes with more than two instance variables – how?

Where to add the age field?

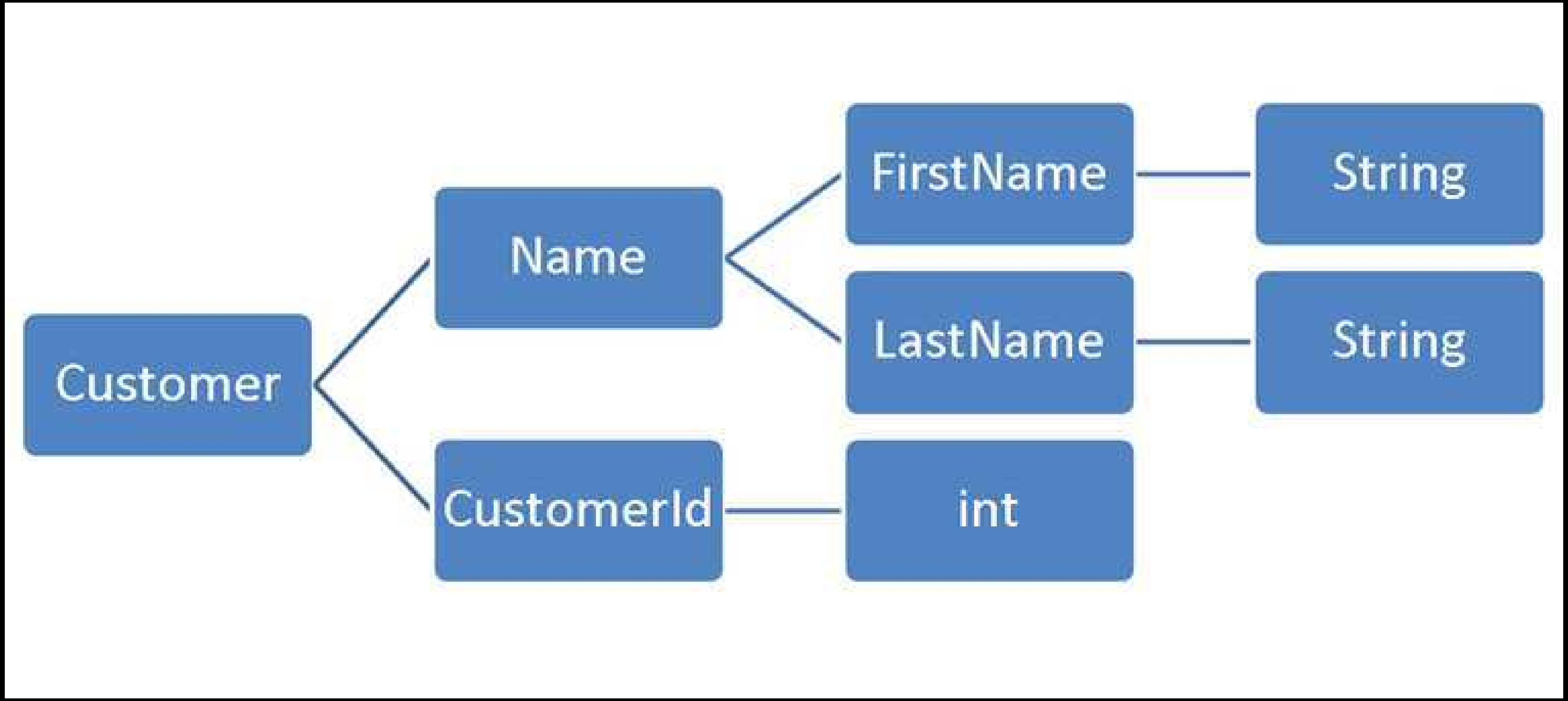


Image: https://williamdurand.fr/2013/06/03/object-calisthenics/

Thank You
-
all for being awesome

By:
Donato Wolfisberg
donato@wolfisberg.dev