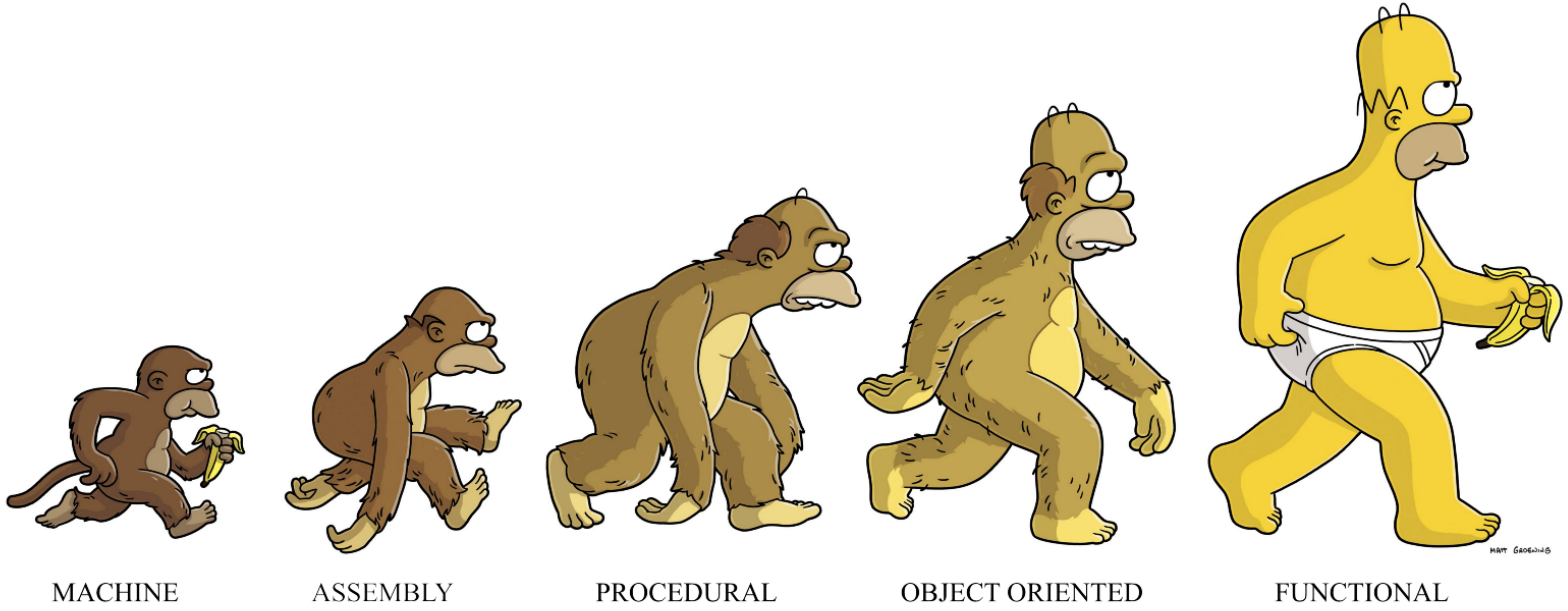
Reactive Programming

With Spring Framework 5

What is Functional Programming

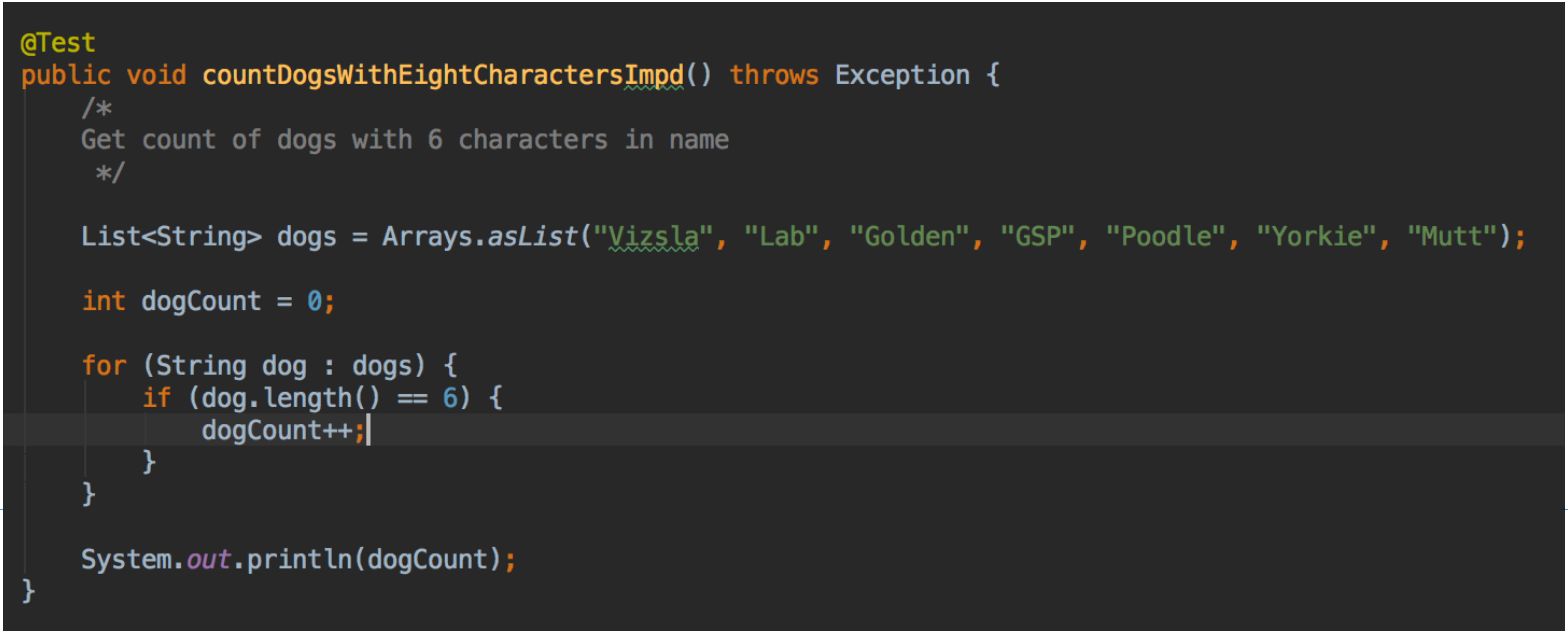
Credit Charles Scalfani



Imperative Programming

- “In computer science, imperative programming is a programming paradigm that uses statements that change a program's state. In much the same way that the imperative mood in natural languages expresses commands, an imperative program consists of commands for the computer to perform.” Source: Wikipedia

Imperative Example

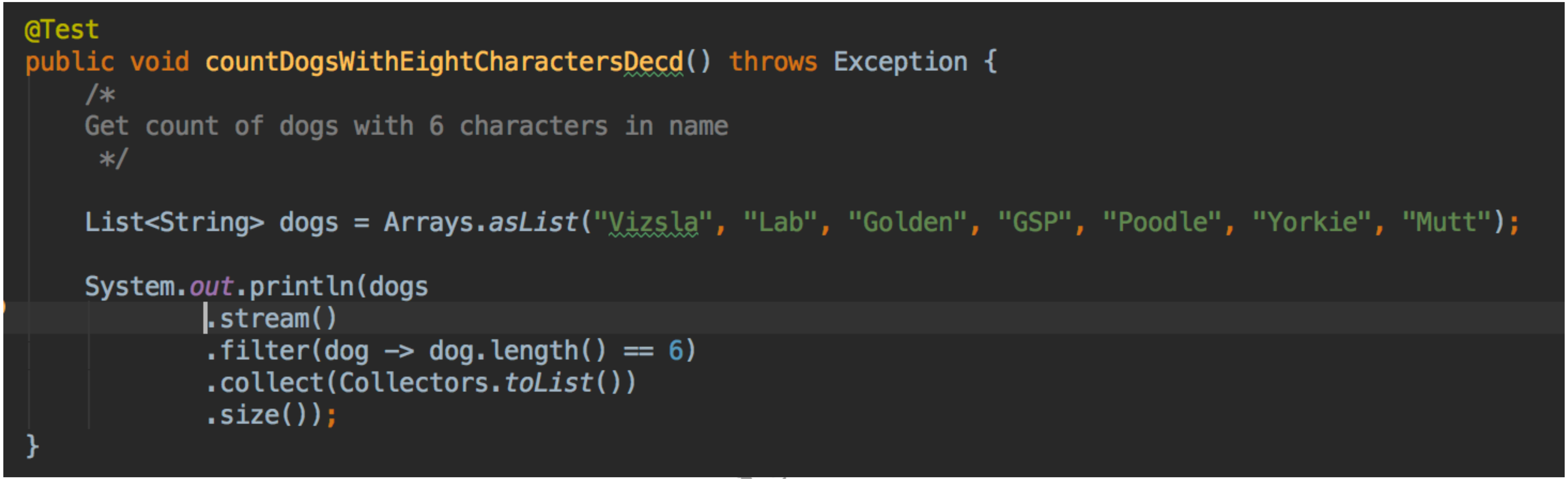


```java
@Test
public void countDogsWithEightCharactersImpd() throws Exception {
    /*
    Get count of dogs with 6 characters in name
     */

    List<String> dogs = Arrays.asList("Vizsla", "Lab", "Golden", "GSP", "Poodle", "Yorkie", "Mutt");

    int dogCount = 0;

    for (String dog : dogs) {
        if (dog.length() == 6) {
            dogCount++;
        }
    }

    System.out.println(dogCount);
}
```

Functional Programming

- “In computer science, functional programming is a programming paradigm - a style of building the structure and elements of computer programs - that treats computation as the evaluation of mathematical functions and avoids changing-state and mutable data.” Source: Wikipedia

Functional Example



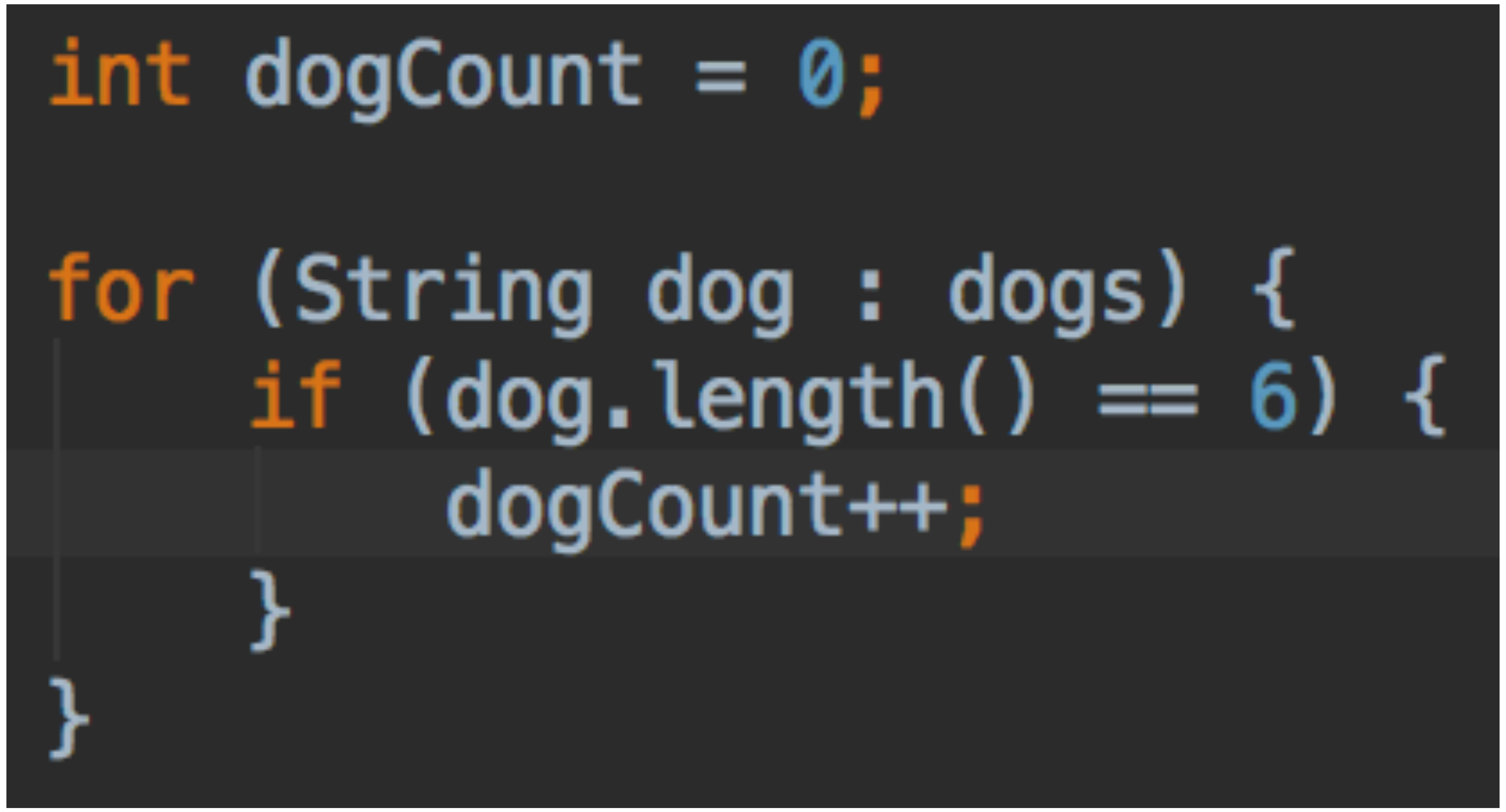
```java
@Test
public void countDogsWithEightCharactersDecd() throws Exception {
    /*
    Get count of dogs with 6 characters in name
     */

    List<String> dogs = Arrays.asList("Vizsla", "Lab", "Golden", "GSP", "Poodle", "Yorkie", "Mutt");

    System.out.println(dogs
            .stream()
            .filter(dog -> dog.length() == 6)
            .collect(Collectors.toList())
            .size());
}
```

Imperative vs Functional

Imperative	Declarative
How to do it	What to do
Mutable	Immutable (Transforms)
Has side effects	No side effects
Pass Objects	Can also pass functions
Hard to Compose	Functional Composition
Not Threadsafe	Threadsafe

Mutability

- Mutability are objects which can change
- Immutable objects cannot change



```
int dogCount = 0;

for (String dog : dogs) {
    if (dog.length() == 6) {
        dogCount++;
    }
}
```

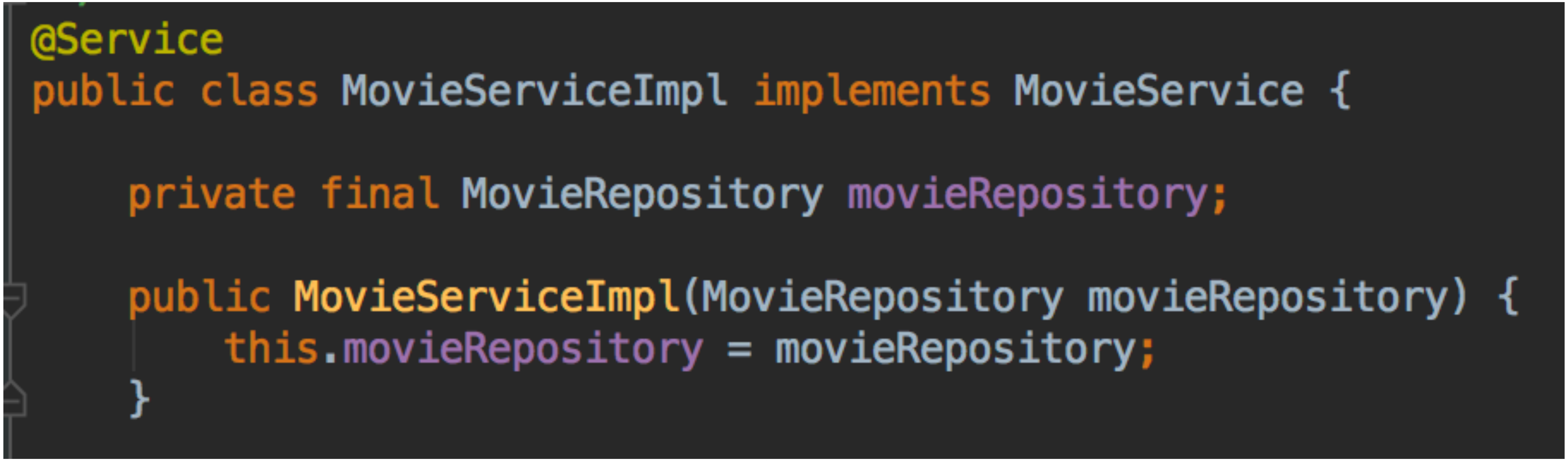
Mutable vs Immutable

- Mutable objects can be error prone and hard to understand
- Immutable objects are easier to use
- Immutable objects are thread safe
- Mutable objects open the door to concurrency problems

Final Variables in Java

- Once initialized, the variable cannot be re-assigned
- Final variables in Java can still be mutated
- BUT - state of the object can change, if properties are not final

Use of Final in Spring



```
@Service
public class MovieServiceImpl implements MovieService {

    private final MovieRepository movieRepository;

    public MovieServiceImpl(MovieRepository movieRepository) {
        this.movieRepository = movieRepository;
    }
```

2021

Java Functions

- Remember in Java - “Everything is an Object”
- Java Function interface was introduced in Java 8
- Java Functions:
  - Can be passed objects
  - Can create objects
  - Can return objects

Java Functional Interfaces

- Added to package java.util.function
- Functional Interfaces rely on using Generics
- **Function<I, O>** - accepts type I, returns type O
- **Predicate<T>** - Accepts type T, returns boolean
- **Supplier<T>** - Accepts no input, returns type T
- **Consumer<T>** - accepts type T, returns no output

SPRING FRAMEWORK

GURU