Plugin and Play

PLUGINS THAT SUPPORT THE FULL SPECTRUM OF KOTLIN FEATURES

15-9-2022 by Simone de Gijt

OPENVALUE

Introduction

Simone de Gijt

2016	- Kotlin 1.0 released - Started working in IT
2017	
2018	
2019	Google announces that Kotlin is the preferred language of Android app developers
2020	
2021	- Signs contract @ OpenValue - Starts working with Kotlin
…	

Twitter: @de_gijt

What to expect?

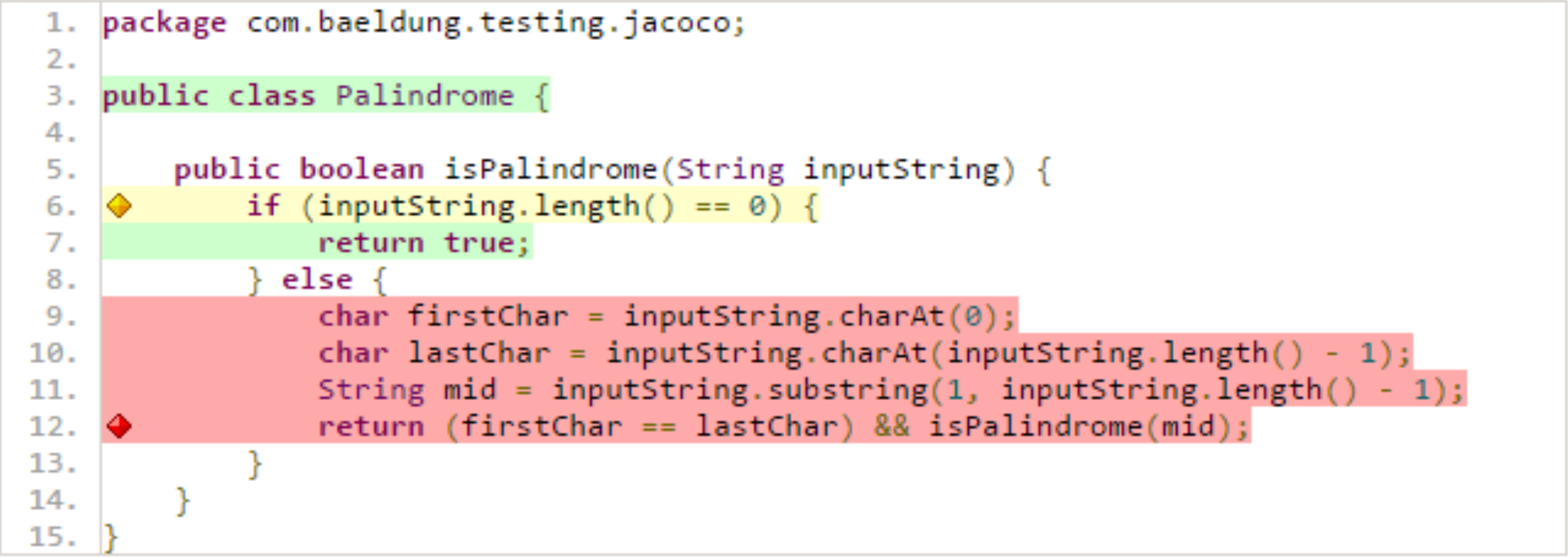
- **Kover**
  - Code covarage
  - Options
  - Advantages / Disadvantages
  - Demo of usages / configuration
- **Ktlint**
  - Linter
  - Options
  - Advantages / Disadvantages
  - Demo of usages / configuration
- **Detekt**
  - Code smells
  - Options
  - Advantages / Disadvantages
  - Demo of usages / configuration

Kover

{kof · fer}

Kover - Gradle plugin for Kotlin code coverage agents: IntelliJ and JaCoCo.

Code coverage

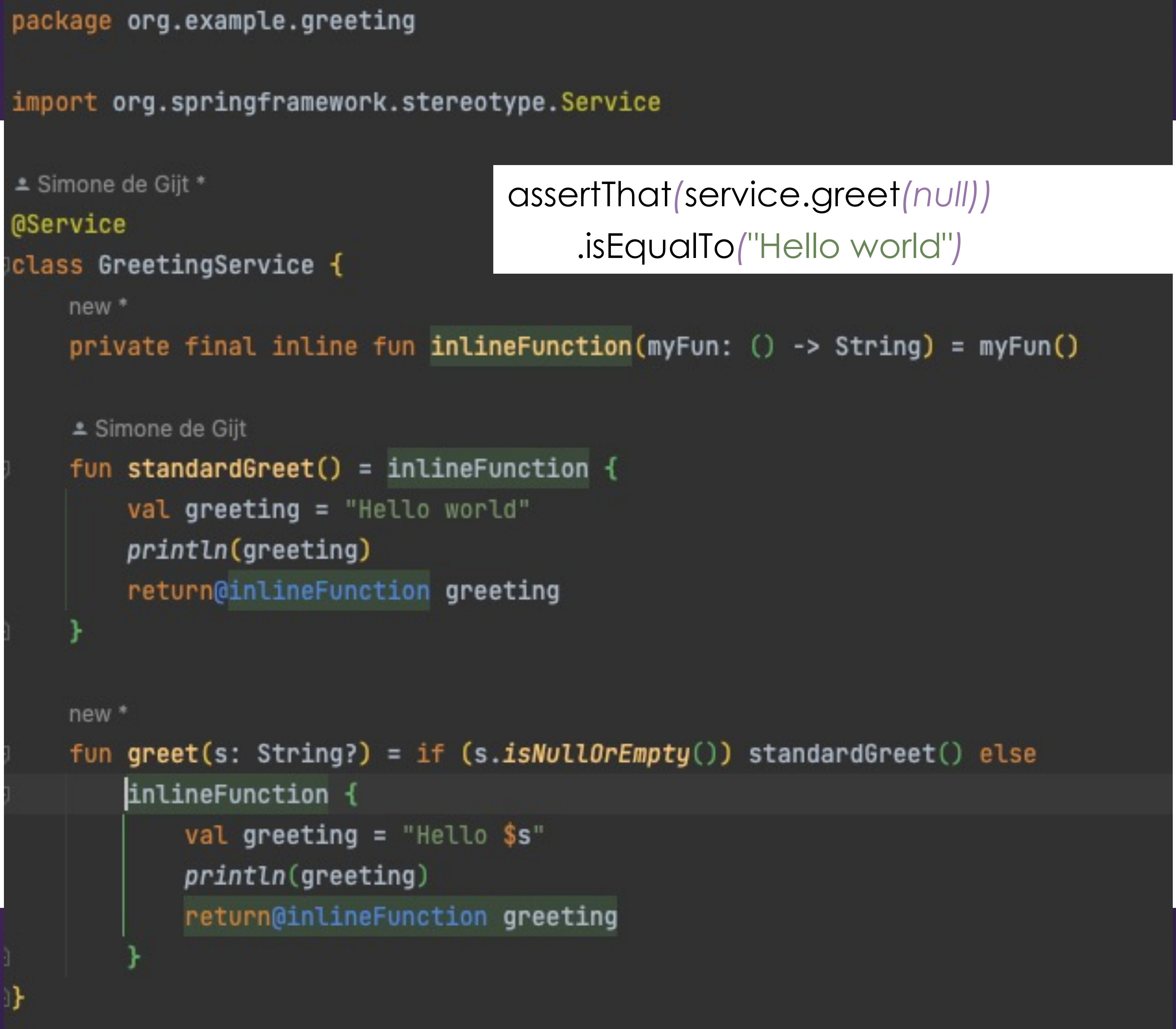


```
package com.baeldung.testing.jacoco;

public class Palindrome {

    public boolean isPalindrome(String inputString) {
        if (inputString.length() == 0) {
            return true;
        } else {
            char firstChar = inputString.charAt(0);
            char lastChar = inputString.charAt(inputString.length() - 1);
            String mid = inputString.substring(1, inputString.length() - 1);
            return (firstChar == lastChar) && isPalindrome(mid);
        }
    }
}
```

Options

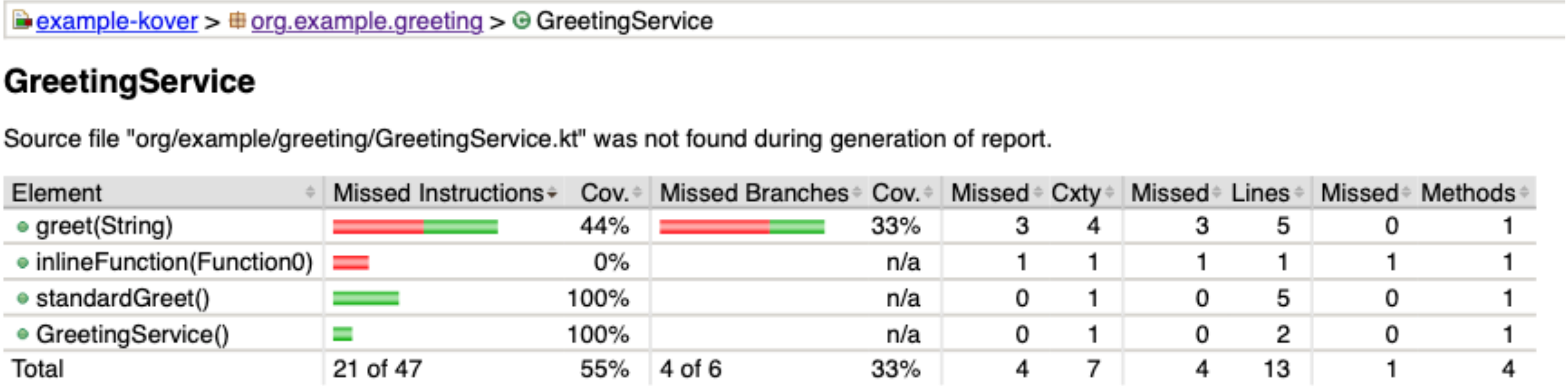
- **IntelliJ IDEA**
  - Available in IntelliJ IDEA
  - Handles Kotlin well
  - Only available in IntelliJ IDEA

- **JaCoCo**
  - Existing tool
  - Well-known and widely recognized format
  - Has some quirks regarding Kotlin



```
package org.example.greeting

import org.springframework.stereotype.Service

Simone de Gijt *
@Service
class GreetingService {
    new *
    private final inline fun inlineFunction(myFun: () -> String) = myFun()

    Simone de Gijt
    fun standardGreet() = inlineFunction {
        val greeting = "Hello world"
        println(greeting)
        return@inlineFunction greeting
    }

    new *
    fun greet(s: String?) = if (s.isNullOrEmpty()) standardGreet() else
        inlineFunction {
            val greeting = "Hello $s"
            println(greeting)
            return@inlineFunction greeting
        }
}
```

```
assertThat(service.greet(null))
        .isEqualTo("Hello world")
```



example-kover > org.example.greeting > GreetingService

GreetingService

Source file "org/example/greeting/GreetingService.kt" was not found during generation of report.

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
● greet(String)	🟥🟩	44%	🟥🟩	33%	3	4	3	5	0	1
● inlineFunction(Function0)	🟥	0%		n/a	1	1	1	1	1	1
● standardGreet()	🟩	100%		n/a	0	1	0	5	0	1
● GreetingService()	🟩	100%		n/a	0	1	0	2	0	1
Total	21 of 47	55%	4 of 6	33%	4	7	4	13	1	4

Advantages Kover

- Provided by JetBrains
- Fully integrated with the Gradle toolchain and multiplatform projects.
- Kotlin Android support without the need to divide into build types and flavours
- Customizable filters for instrumented classes
- Handles specific Kotlin features

Disadvantages

- Still in an experimental state
- Only available for Gradle (not Maven)
- Engine Jacoco doesn’t support Kotlin Directory Structure

Demo

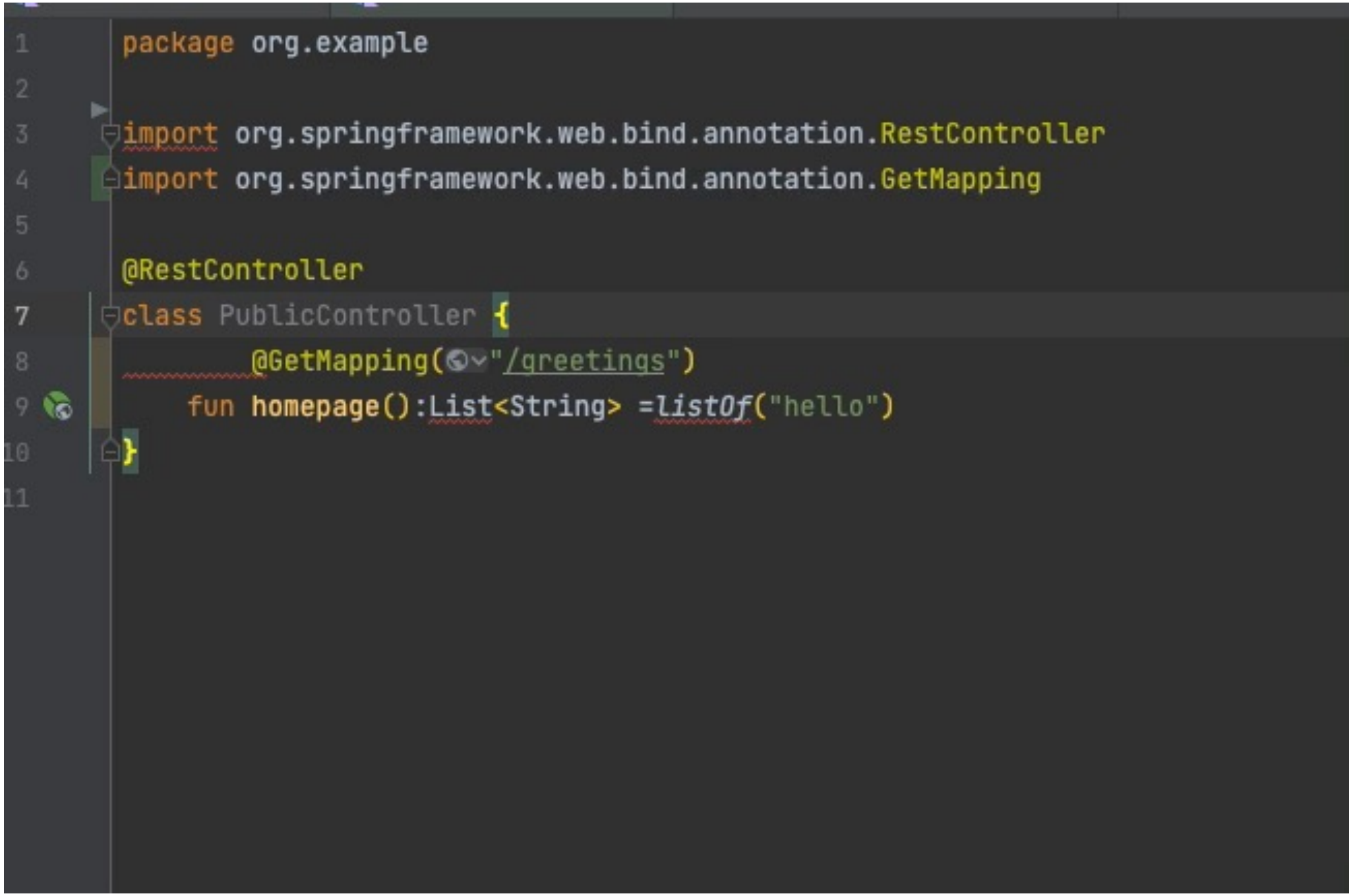
https://github.com/SimonedeGijt/example-kover

Ktlint

{ki̱tty · lint}

Ktlint - An anti-bikeshedding Kotlin linter with built-in formatter

Linter



```
package org.example

import org.springframework.web.bind.annotation.RestController
import org.springframework.web.bind.annotation.GetMapping

@RestController
class PublicController {
        @GetMapping("/greetings")
    fun homepage():List<String> =listOf("hello")
}
```

Advantages

- It saves time!
- It saves energy
- It simplifies your process

Plugin options

- Pinterest:
  - **Static code analysis tool** that is used to analyse the Kotlin code for you.
  - Follows the **official Kotlin code style**

- JLLeitschuh:
  - **Gradle wrapper** over the pinterest project
  - Adds **special tasks**

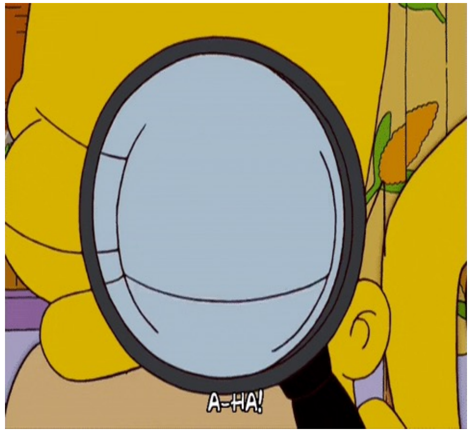
- GantSign:
  - **Maven wrapper** over the pinterest project

Demo

https://github.com/SimonedeGijt/example-ktlint

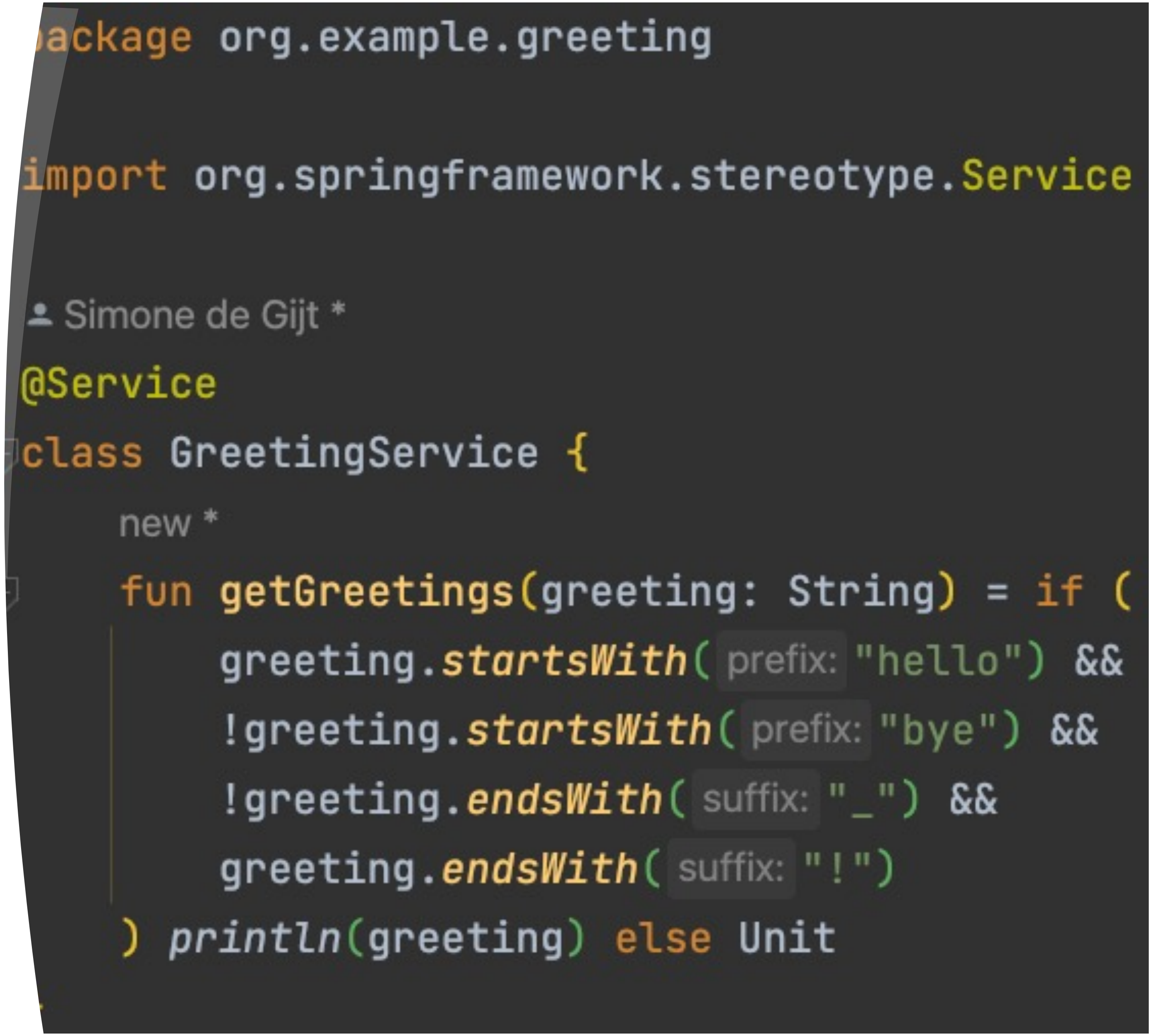
Detekt

{de·tect}



Detekt - A static code analysis tool for the Kotlin programming language.

Code smells



```
package org.example.greeting

import org.springframework.stereotype.Service

Simone de Gijt *
@Service
class GreetingService {
    new *
    fun getGreetings(greeting: String) = if (
        greeting.startsWith( prefix: "hello") &&
        !greeting.startsWith( prefix: "bye") &&
        !greeting.endsWith( suffix: "_") &&
        greeting.endsWith( suffix: "!")
    ) println(greeting) else Unit
```

Options

- **Sonarqube**
  - Very nice reports that integrate with Gitlab
  - You get a lot of extra’s..
  - Community and paid versions
- **Megalinter**
  - Free
  - Not Kotlin specific

Advantages

- Highly configurable rule sets
- Specification of quality gates which will break your build
- Gradle plugin for code analysis via Gradle builds
- SonarQube & IntelliJ integratIons
- Third party integrations for Maven, Bazel and Github Actions (Docker based and Javascript based)

Disadvantages

- Doesn’t support other languages.

Demo

https://github.com/SimonedeGijt/example-detekt

Sources

- Kover – The Code Coverage Plugin
- https://kotlinlang.org/docs/multiplatform.html
- https://www.geeksforgeeks.org/code-formatting-in-kotlin-using-ktlint/
- https://blog.mindorks.com/code-formatting-in-kotlin-using-ktlint
- https://github.com/Kotlin/kotlinx-kover
- https://github.com/JLLeitschuh/ktlint-gradle
- https://github.com/pinterest/ktlint
- https://github.com/gantsign/ktlint-maven-plugin
- https://github.com/detekt/detekt
- https://megalinter.github.io/latest/installation/

?

Questions

Contact

Slideshare:

Twitter:
@de_gijt

LinkedIn:
Simone de Gijt