Java static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your JAVA code

should not be used in hashes
Bug

Floating point numbers should not be tested for equality
Bug

Increment (++) and decrement (--) operators should not be used in a method call or mixed with other operators in an expression
Code Smell

Limited dependence should be placed on operator precedence
Code Smell

Custom getter method should not be used to override record's getter behavior
Code Smell

Tests should use fixed data instead of randomized data
Code Smell

Spring's ModelAndViewAssert assertions should be used instead of other assertions
Code Smell

Lambdas should not have too many lines
Code Smell

"@EnableAutoConfiguration" should be fine-tuned
Code Smell

Enum values should be compared with "=="
Code Smell

Spring components should use constructor injection
Code Smell

"@RequestMapping" methods should not be "private"

Analyze your code

Code Smell   Major   spring owasp

A method with a `@RequestMapping` annotation part of a class annotated with `@Controller` (directly or indirectly through a meta annotation - `@RestController` from Spring Boot is a good example) will be called to handle matching web requests. That will happen even if the method is `private`, because Spring invokes such methods via reflection, without checking visibility.

So marking a sensitive method `private` may seem like a good way to control how such code is called. Unfortunately, not all Spring frameworks ignore visibility in this way. For instance, if you've tried to control web access to your sensitive, `private`, `@RequestMapping` method by marking it `@Secured` ... it will still be called, whether or not the user is authorized to access it. That's because AOP proxies are not applied to private methods.

In addition to `@RequestMapping`, this rule also considers the annotations introduced in Spring Framework 4.3: `@GetMapping`, `@PostMapping`, `@PutMapping`, `@DeleteMapping`, `@PatchMapping`.

Noncompliant Code Example

```
@RequestMapping("/greet", method = GET)
private String greet(String greetee) {  // Noncompliant
```

Compliant Solution

```
@RequestMapping("/greet", method = GET)
public String greet(String greetee) {
```

See

- OWASP Top 10 2017 Category A6 - Security Misconfiguration

Available In:
sonarlint | sonarcloud | sonarqube

Regex patterns should not be created needlessly

Code Smell

Track uses of disallowed constructors

Code Smell

Java 8's "Files.exists" should not be used

Code Smell

"Optional" should not be used for parameters

Code Smell

Tests should be kept in a dedicated