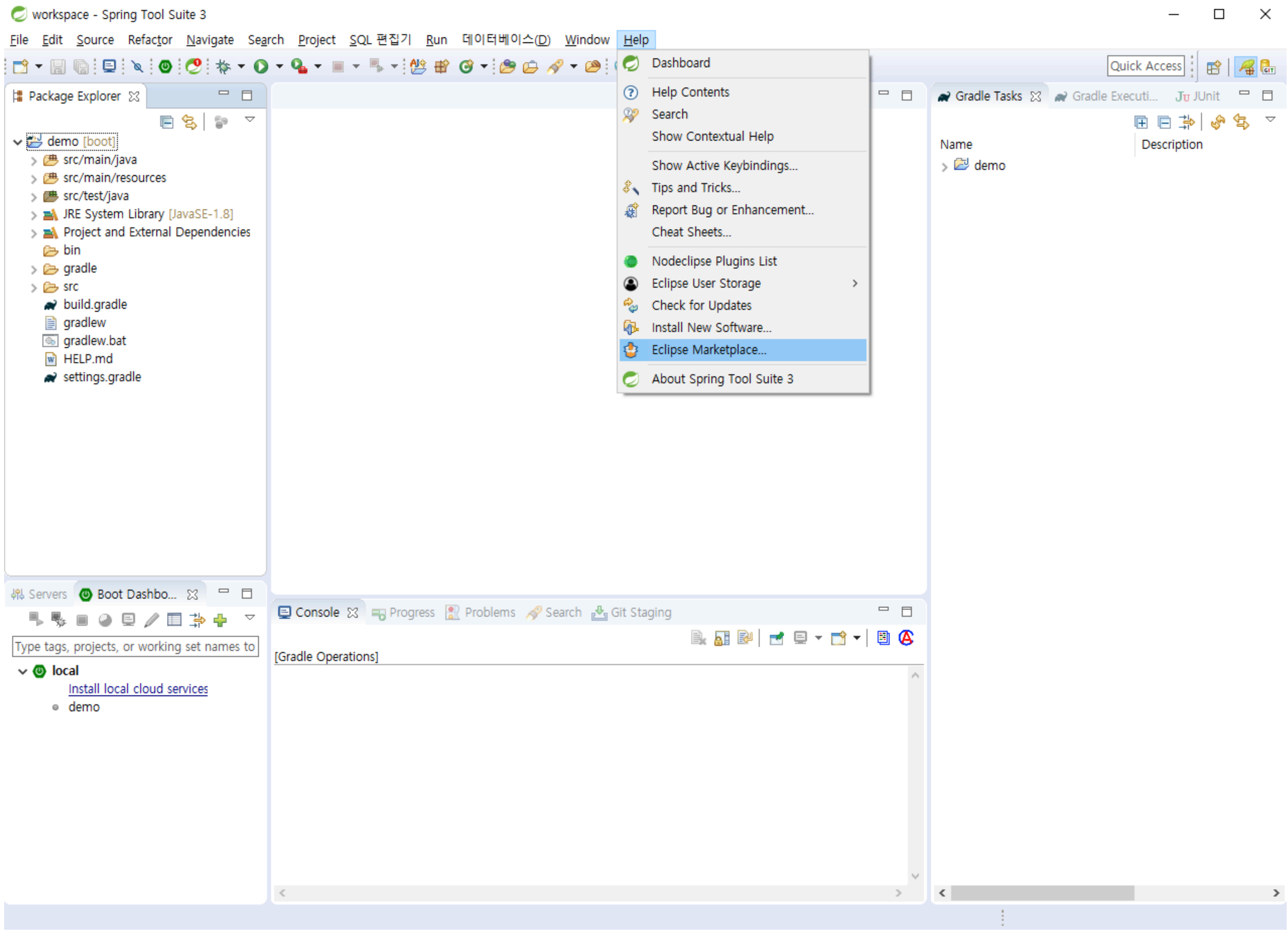
# 개요

큰 규모의 프로젝트에서는 개발자가이드나 코드리뷰 만으로 소스코드 내에 산재되어 있는 잠재적인 문제점을 모두 찾아내기 어렵다. 따라서 소스코드 전체를 일괄로 스캔하여 문제가 되는 패턴을 자동으로 찾아서 수정할 수 있는 도구의 도입은 반드시 필요하다.

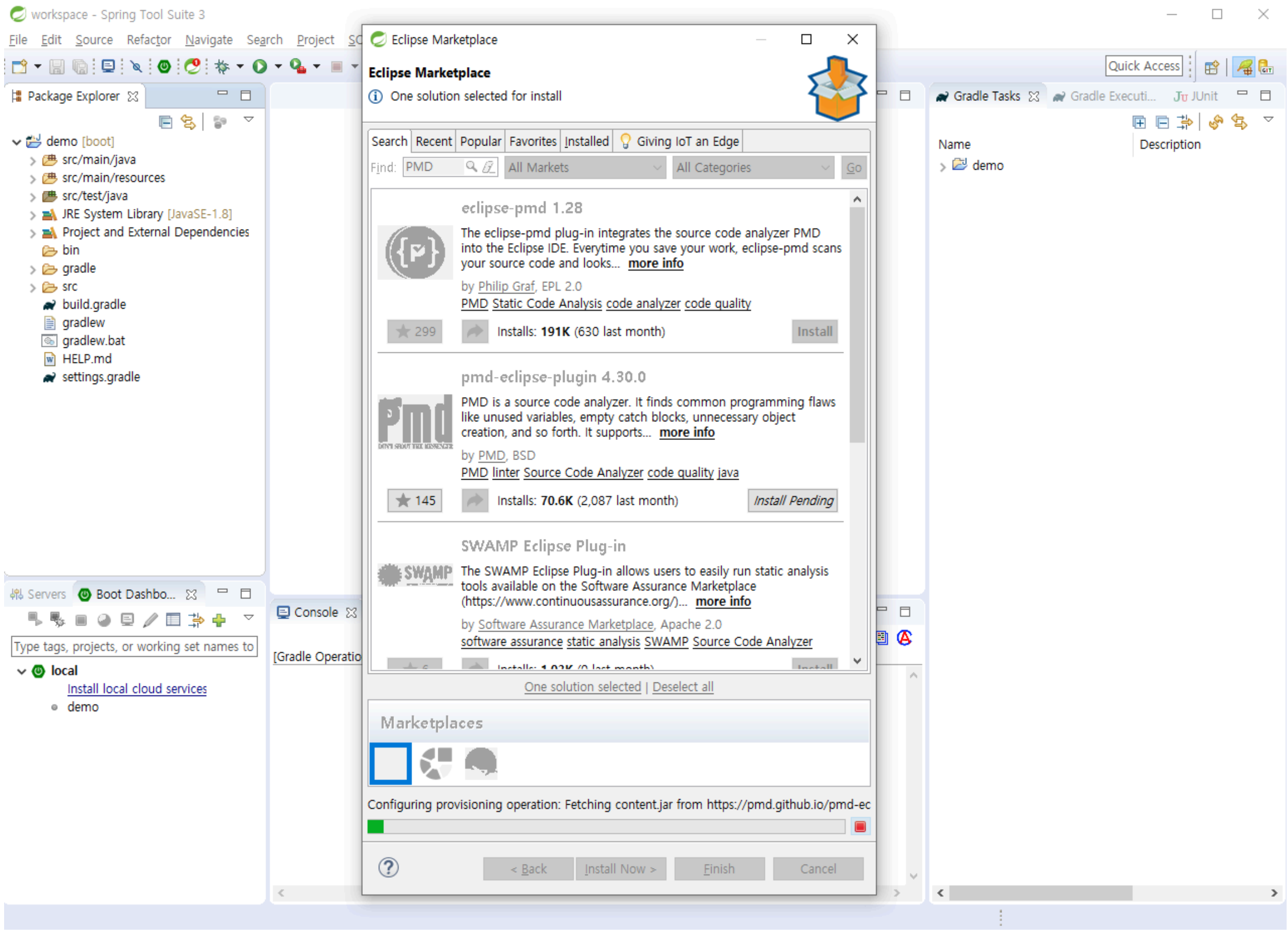
대표적인 정적분석 도구 중 하나인 PMD에 대해 알아보도록 하자. PMD, SpotBugs(FindBugs), SonarQube 등 정적분석도구는 소스코드에 대한 inspection을 강화하여 이후 발생가능한 이슈들을 개발단계에서 점검하는 도구이다.

# STS(Spring-Tool-Suit) PMD Plugin 적용

1) Help > Eclipse Marketplace



2) Search > PMD > PMD Install



Eclipse Marketplace

One solution selected for install

eclipse-pmd 1.28

The eclipse-pmd plug-in integrates the source code analyzer PMD into the Eclipse IDE. Everytime you save your work, eclipse-pmd scans your source code and looks... more info

by Philip Graf, EPL 2.0

PMD Static Code Analysis code analyzer code quality

299 Installs: **191K** (630 last month) Install

pmd-eclipse-plugin 4.30.0

PMD is a source code analyzer. It finds common programming flaws like unused variables, empty catch blocks, unnecessary object creation, and so forth. It supports... more info

by PMD, BSD

PMD linter Source Code Analyzer code quality java

145 Installs: **70.6K** (2,087 last month) *Install Pending*

SWAMP Eclipse Plug-in

The SWAMP Eclipse Plug-in allows users to easily run static analysis tools available on the Software Assurance Marketplace (https://www.continuousassurance.org/)... more info

by Software Assurance Marketplace, Apache 2.0

software assurance static analysis SWAMP Source Code Analyzer

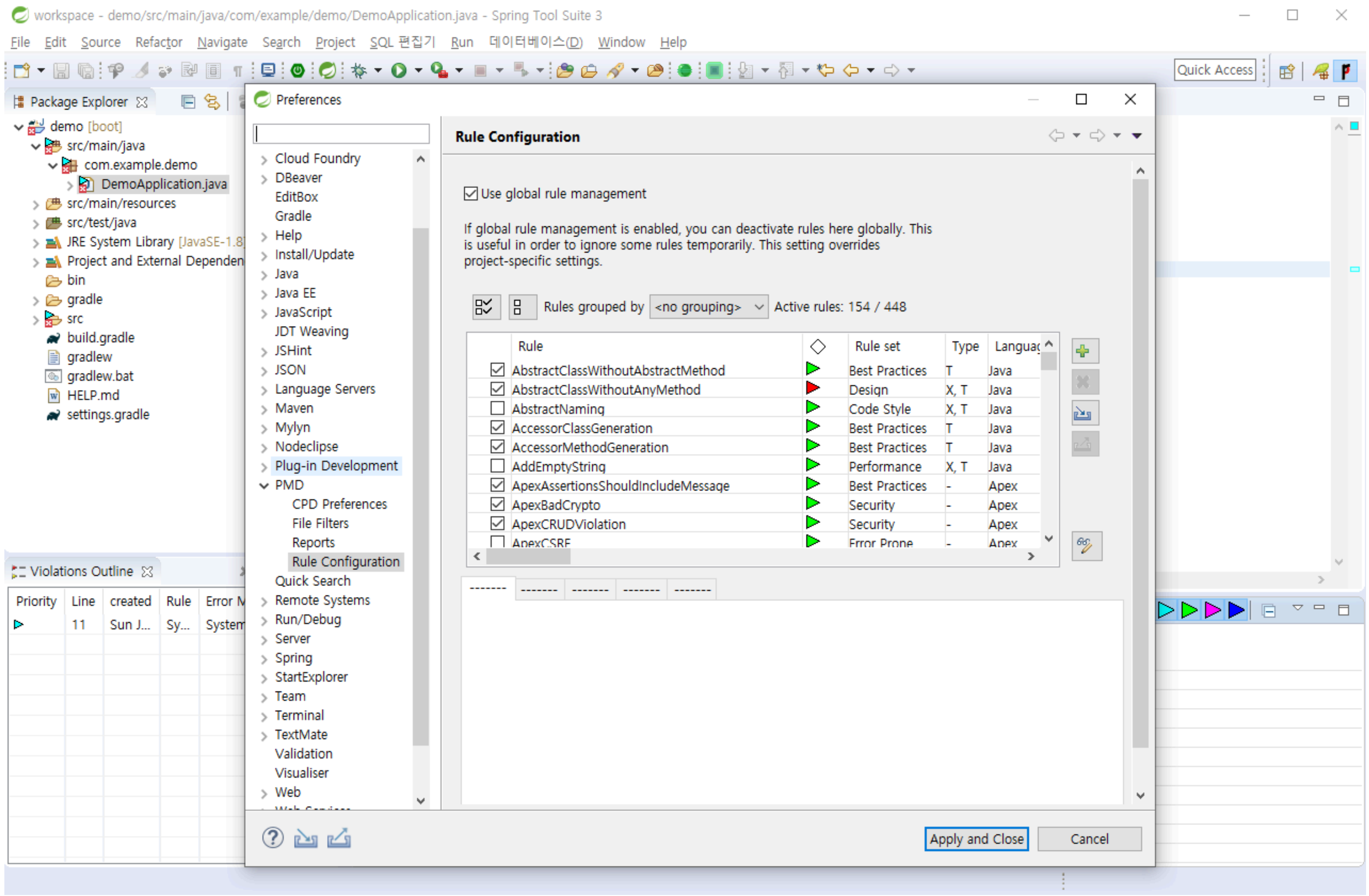
One solution selected | Deselect all

Configuring provisioning operation: Fetching content.jar from https://pmd.github.io/pmd-ec

한 차례 리스타트 후 pmd 정적분석 도구를 사용할 수 있다.

## PMD 활용

1) PMD Ruleset 커스터마이징



| Priority | Line | created | Rule | Error M |
|---|---|---|---|---|
| ▷ | 11 | Sun J... | Sy... | System |

**Rule Configuration**

☑ Use global rule management

If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

Rules grouped by <no grouping> Active rules: 154 / 448

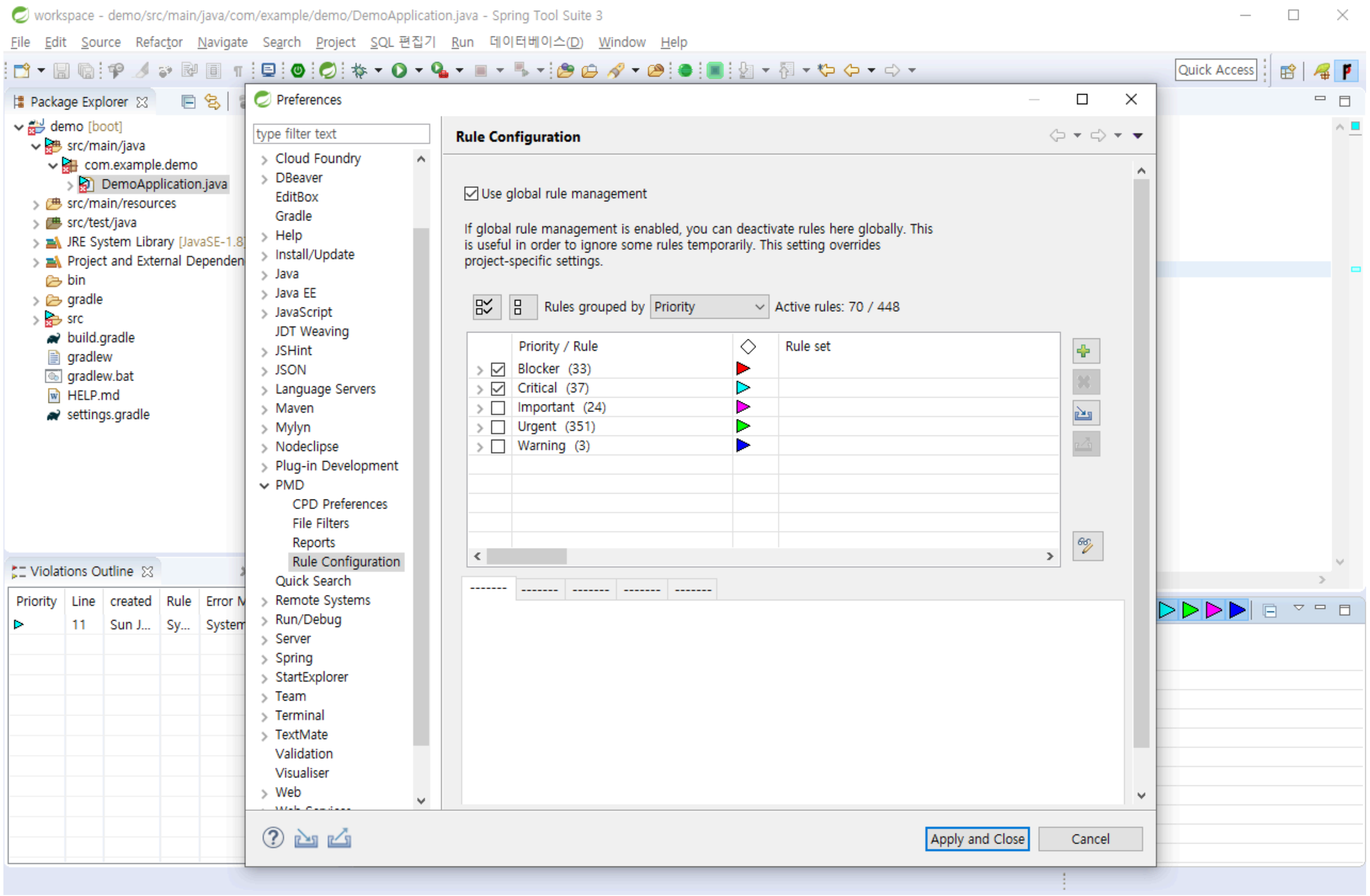
| | Rule | ◇ | Rule set | Type | Languag |
|---|---|---|---|---|---|
| ☑ | AbstractClassWithoutAbstractMethod | ▶ | Best Practices | T | Java |
| ☑ | AbstractClassWithoutAnyMethod | ▶ | Design | X, T | Java |
| ☐ | AbstractNaming | ▶ | Code Style | X, T | Java |
| ☑ | AccessorClassGeneration | ▶ | Best Practices | T | Java |
| ☑ | AccessorMethodGeneration | ▶ | Best Practices | T | Java |
| ☐ | AddEmptyString | ▶ | Performance | X, T | Java |
| ☑ | ApexAssertionsShouldIncludeMessage | ▶ | Best Practices | - | Apex |
| ☑ | ApexBadCrypto | ▶ | Security | - | Apex |
| ☑ | ApexCRUDViolation | ▶ | Security | - | Apex |
| ☐ | ApexCSRF | ▶ | Error Prone | - | Apex |

window > Preferences > PMD > Rule Configuration 접속 시 위와 같이 Default Rule을 확인할 수 있다.

RuleSet은 여러 형태로 커스터마이징이 가능하다. <no grouping>으로 되어 있는 콤보박스를 선택하여 정렬하며 RuleSet을 선택할 수 있다. Default로 등록되어 있는 Active Rule 448개 모두를 등록할 경우 개발자는 개발하기 이전에 inspection의 늪에 허덕여야 할 것이다.

PMD는 총 5단계의 Inspection Level을 관리하고 있다.

- Blocker : High Level의 Inspection으로 반드시 적용되어야 할 Rule (빨간색)
- Critical : Medium High Level의 Inspection으로 컴파일 에러가 발생하는 Rule 등급 (하늘색)
- Urgent : Medium Level의 Inspection으로 권고하는 Rule (초록색)
- Important : Medium Low Level의 Inspection (보라색)
- Warning : Low Level의 Inspection (파란색)



workspace - demo/src/main/java/com/example/demo/DemoApplication.java - Spring Tool Suite 3

Preferences

**Rule Configuration**

☑ Use global rule management

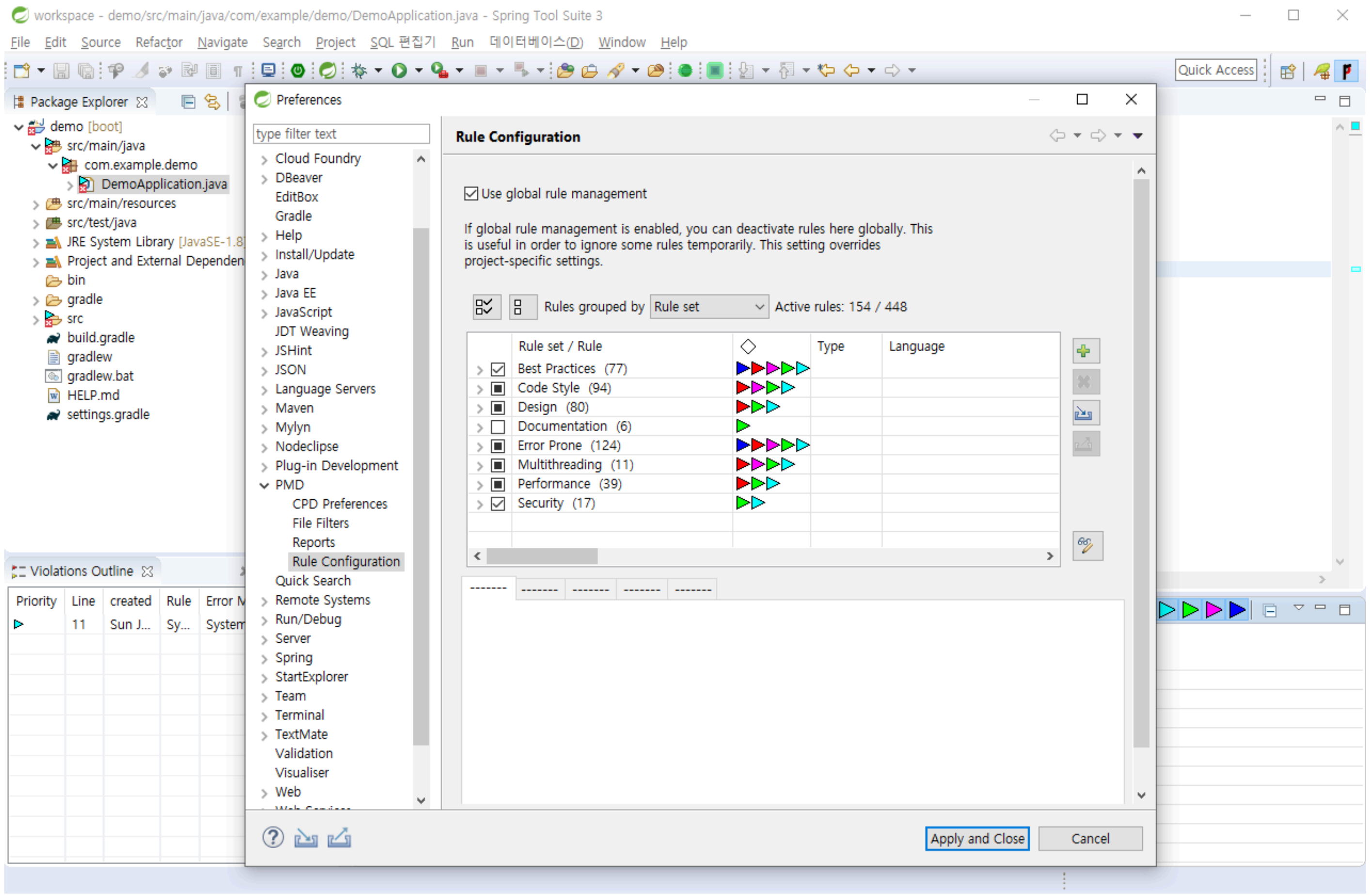
If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

Rules grouped by Priority    Active rules: 70 / 448

| Priority / Rule | ◇ | Rule set |
|---|---|---|
| ☑ Blocker (33) | ▶ | |
| ☑ Critical (37) | ▶ | |
| ☐ Important (24) | ▶ | |
| ☐ Urgent (351) | ▶ | |
| ☐ Warning (3) | ▶ | |

개인적으로 추천하는 RuleSet은 두가지 형태의 조합이다. 먼저, Priority 기준으로 Blocker와 Critical을 선택한다. 전체 448개 중 70개가 그 대상이다. (버전별 상이할 수 있음)

다음으로 Rule Set을 선택하여, Best Practices와 Security 전체를 선택한다. 이때, Best Practices 전체 적용에 부담이 있을 경우에는 +로 Security만 포함하여 적용하도록 한다.



workspace - demo/src/main/java/com/example/demo/DemoApplication.java - Spring Tool Suite 3

Preferences

**Rule Configuration**

☑ Use global rule management

If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

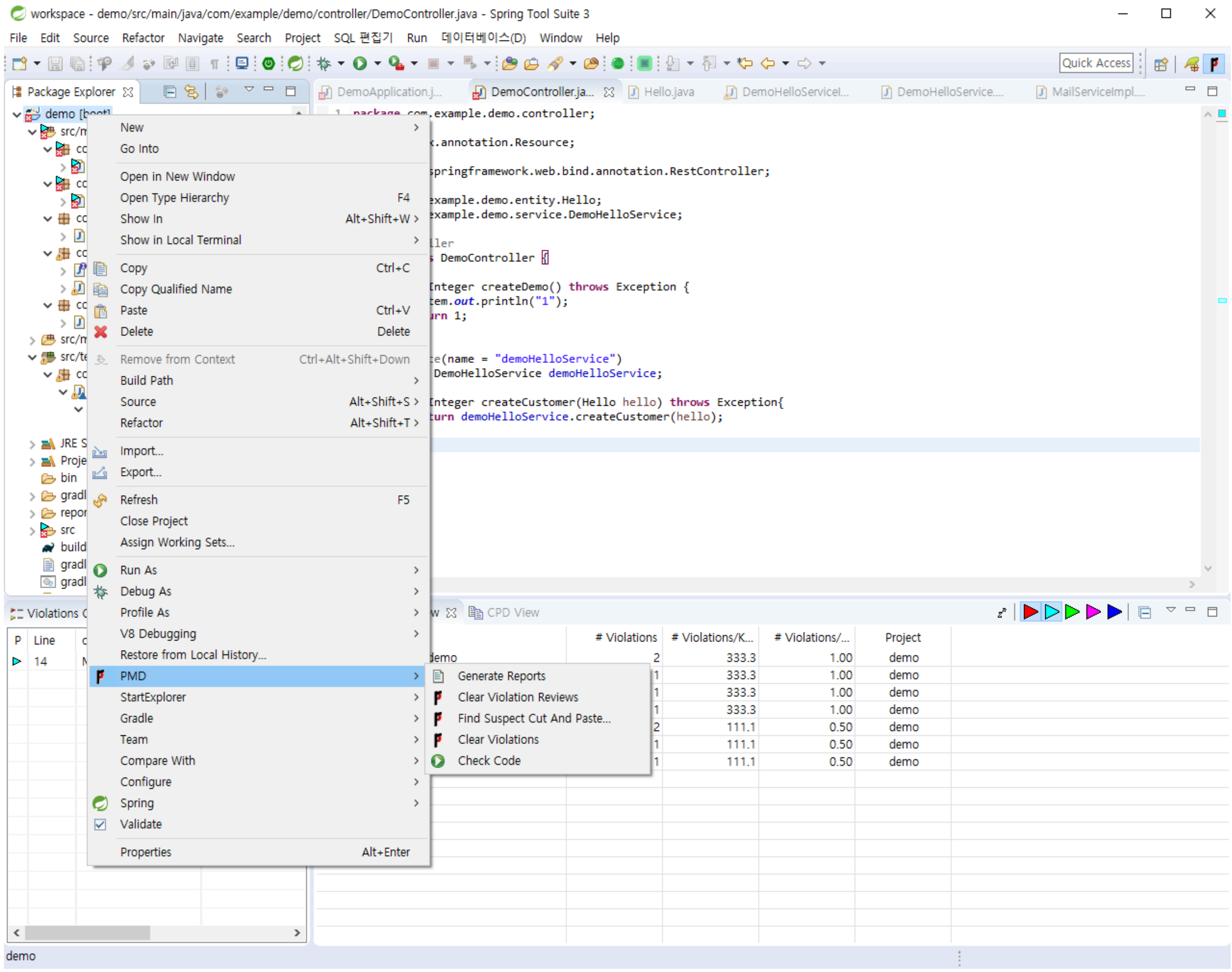
Rules grouped by Rule set    Active rules: 154 / 448

| Rule set / Rule | ◇ | Type | Language |
|---|---|---|---|
| ☑ Best Practices (77) | ▶▶▶▶▶ | | |
| ▣ Code Style (94) | ▶▶▶▶ | | |
| ▣ Design (80) | ▶▶▶ | | |
| ☐ Documentation (6) | ▶ | | |
| ▣ Error Prone (124) | ▶▶▶▶▶ | | |
| ▣ Multithreading (11) | ▶▶▶▶ | | |
| ▣ Performance (39) | ▶▶▶ | | |
| ☑ Security (17) | ▶▶ | | |

이 경우 전체 448개 중 154개의 Ruleset이 등록된 상태가 된다.

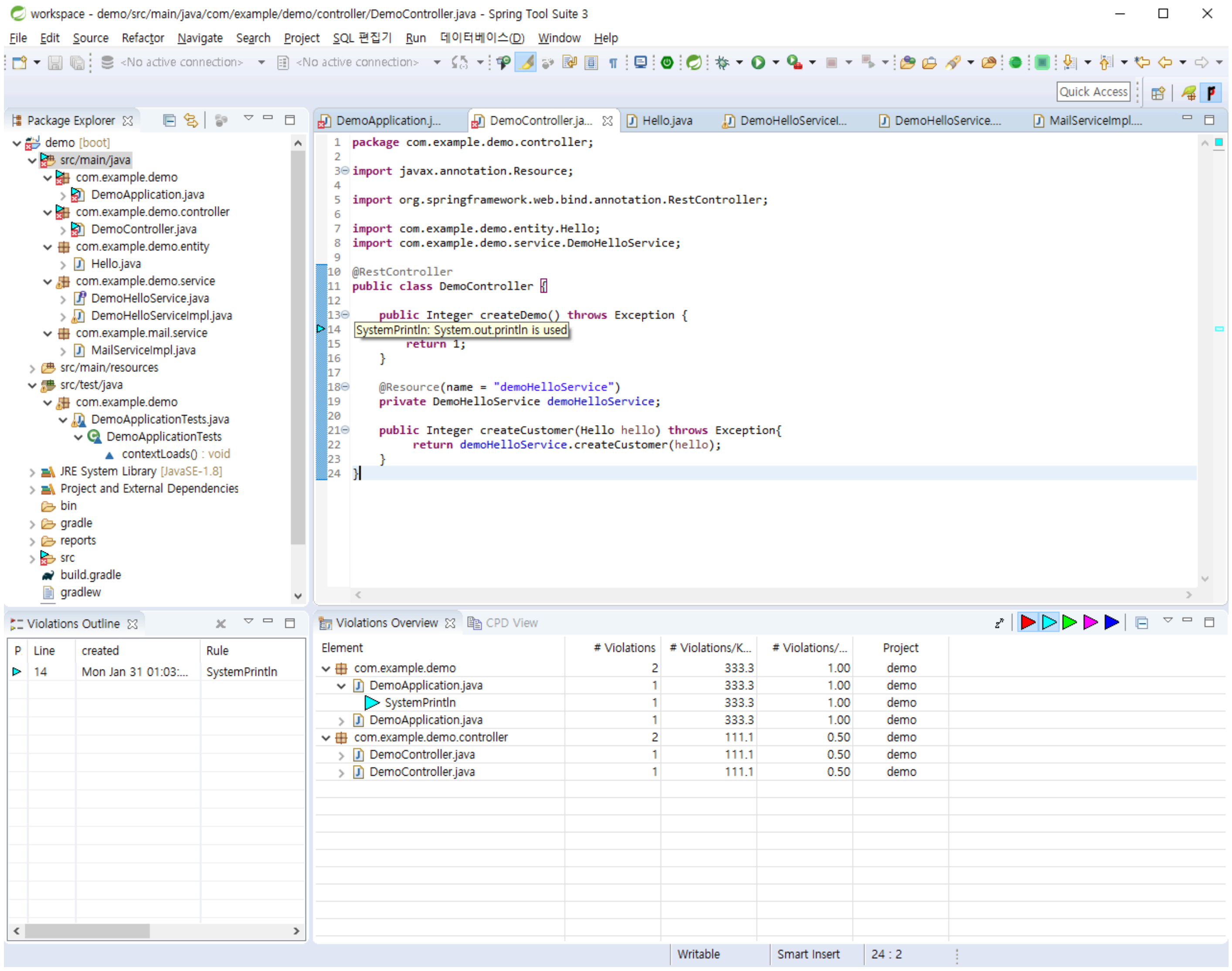
등록이 완료되면, 코드 검사를 실행한다. PMD는 아래와 같은 기능 목록을 지원한다.

- Clear Violation Reviews : Violation Review 초기화
- Find Suspect Cut And Paste Detector : 중복 코드 체크
- Clear Violations : Violation 초기화
- Check Code : 코드 Inspection



2) Check Code

먼저 개발된 소스코드의 Inspection을 실행해 보도록 하자. src/main/java > 우클릭 > PMD > Check Code 클릭 시 선택된 소스 코드 하위를 포함하여 검증한다. 이로인해 대체로 src/main/java를 선택한 후 check를 수행한다.

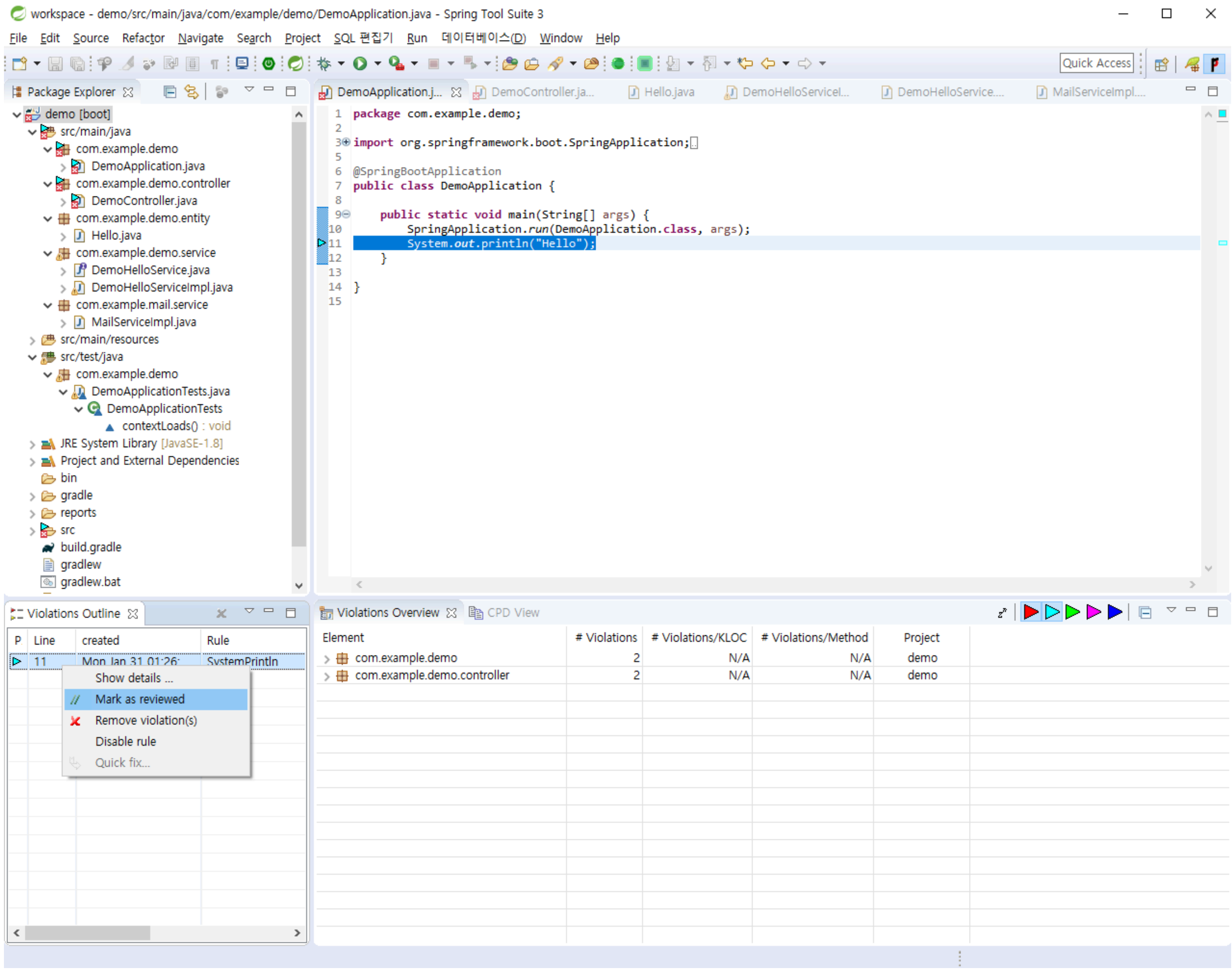


```java
package com.example.demo.controller;

import javax.annotation.Resource;

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

import com.example.demo.entity.Hello;
import com.example.demo.service.DemoHelloService;

@RestController
public class DemoController {

    public Integer createDemo() throws Exception {

        return 1;
    }

    @Resource(name = "demoHelloService")
    private DemoHelloService demoHelloService;

    public Integer createCustomer(Hello hello) throws Exception{
         return demoHelloService.createCustomer(hello);
    }
}
```

| P | Line | created | Rule |
|---|---|---|---|
| ▷ | 14 | Mon Jan 31 01:03:... | SystemPrintln |

| Element | # Violations | # Violations/K... | # Violations/... | Project |
|---|---|---|---|---|
| com.example.demo | 2 | 333.3 | 1.00 | demo |
| DemoApplication.java | 1 | 333.3 | 1.00 | demo |
| SystemPrintln | 1 | 333.3 | 1.00 | demo |
| DemoApplication.java | 1 | 333.3 | 1.00 | demo |
| com.example.demo.controller | 2 | 111.1 | 0.50 | demo |
| DemoController.java | 1 | 111.1 | 0.50 | demo |
| DemoController.java | 1 | 111.1 | 0.50 | demo |

위와 같이 Violation에 걸린 항목들을 확인할 수 있다. PMD 퍼스펙티브를 열거나 Violations Overview 뷰를 열면 간단한 통계를 볼 수 있다. 또한 색깔로 구분된 Violation Level을 기준으로 기본 선택되어 있는 화면에서 선택을 해제하여 위반 우선 순위별로 확인 가능하고 각 파일의 위반사항 갯수, 라인당 위반 수, 메소드당 위반 수를 볼 수 있다. 가장 많이 걸리는 부분인 SystemPrintLn에 대한 예를 위와 같이 확인할 수 있다.



workspace - demo/src/main/java/com/example/demo/DemoApplication.java - Spring Tool Suite 3

```java
package com.example.demo;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
        System.out.println("Hello");
    }

}
```

Violations Outline

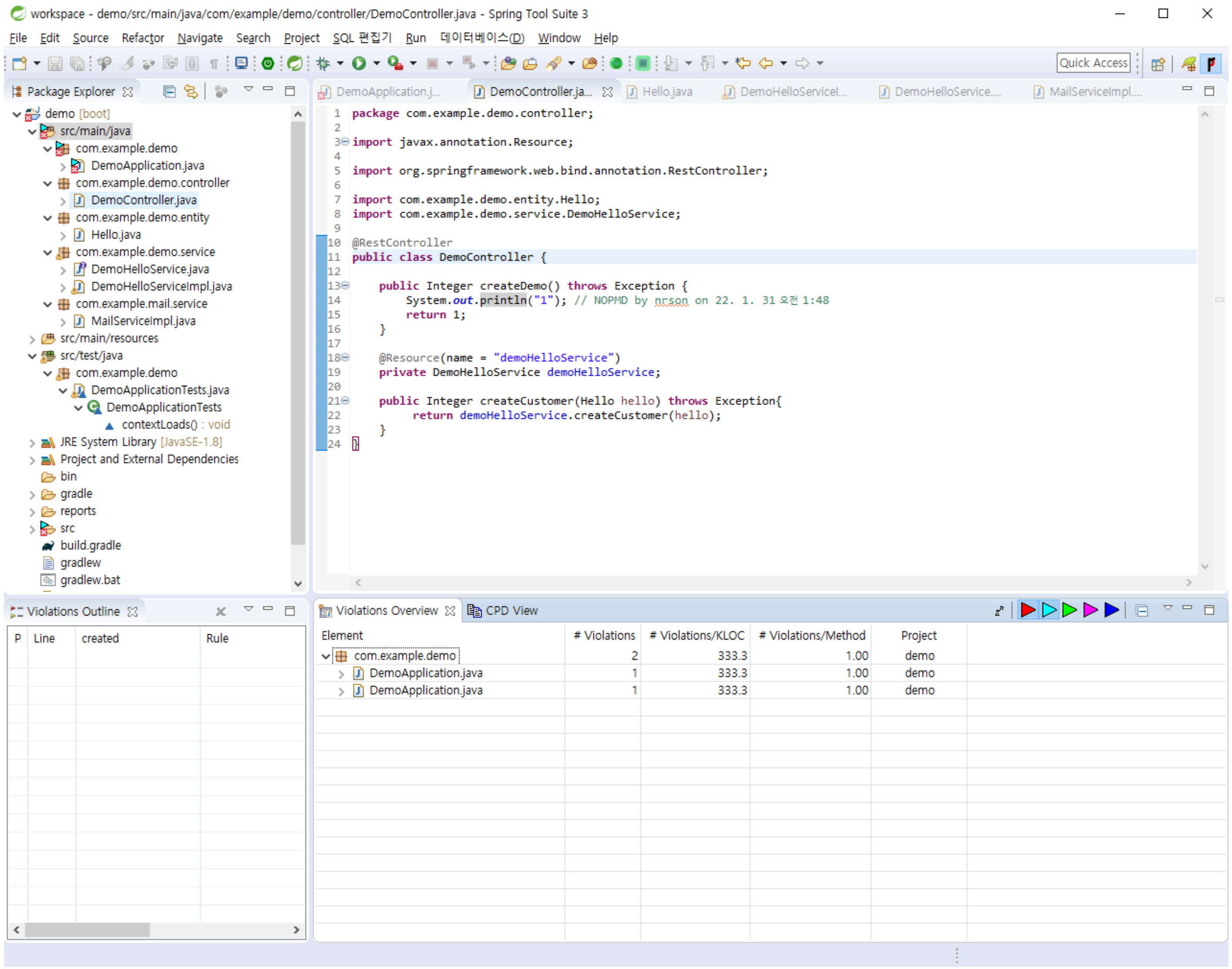
| P | Line | created | Rule |
|---|---|---|---|
|  | 11 | Mon Jan 31 01:26: | SystemPrintln |

Violations Overview

| Element | # Violations | # Violations/KLOC | # Violations/Method | Project |
|---|---|---|---|---|
| com.example.demo | 2 | N/A | N/A | demo |
| com.example.demo.controller | 2 | N/A | N/A | demo |

Violations Outline 뷰에서 Violation을 예외처리하는 Mark as reviewed를 선택할 수 있다. 이 경우 아래와 같이 주석이 달리며, // NOPMD가 포함되어 있는 라인은 PMD Violation에서 예외처리된다.

// NOPMD by nrson on 22. 1. 31 오전 1:48



workspace - demo/src/main/java/com/example/demo/controller/DemoController.java - Spring Tool Suite 3

```java
package com.example.demo.controller;

import javax.annotation.Resource;

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

import com.example.demo.entity.Hello;
import com.example.demo.service.DemoHelloService;

@RestController
public class DemoController {

    public Integer createDemo() throws Exception {
        System.out.println("1"); // NOPMD by nrson on 22. 1. 31 오전 1:48
        return 1;
    }

    @Resource(name = "demoHelloService")
    private DemoHelloService demoHelloService;

    public Integer createCustomer(Hello hello) throws Exception{
         return demoHelloService.createCustomer(hello);
    }
}
```

Violations Overview

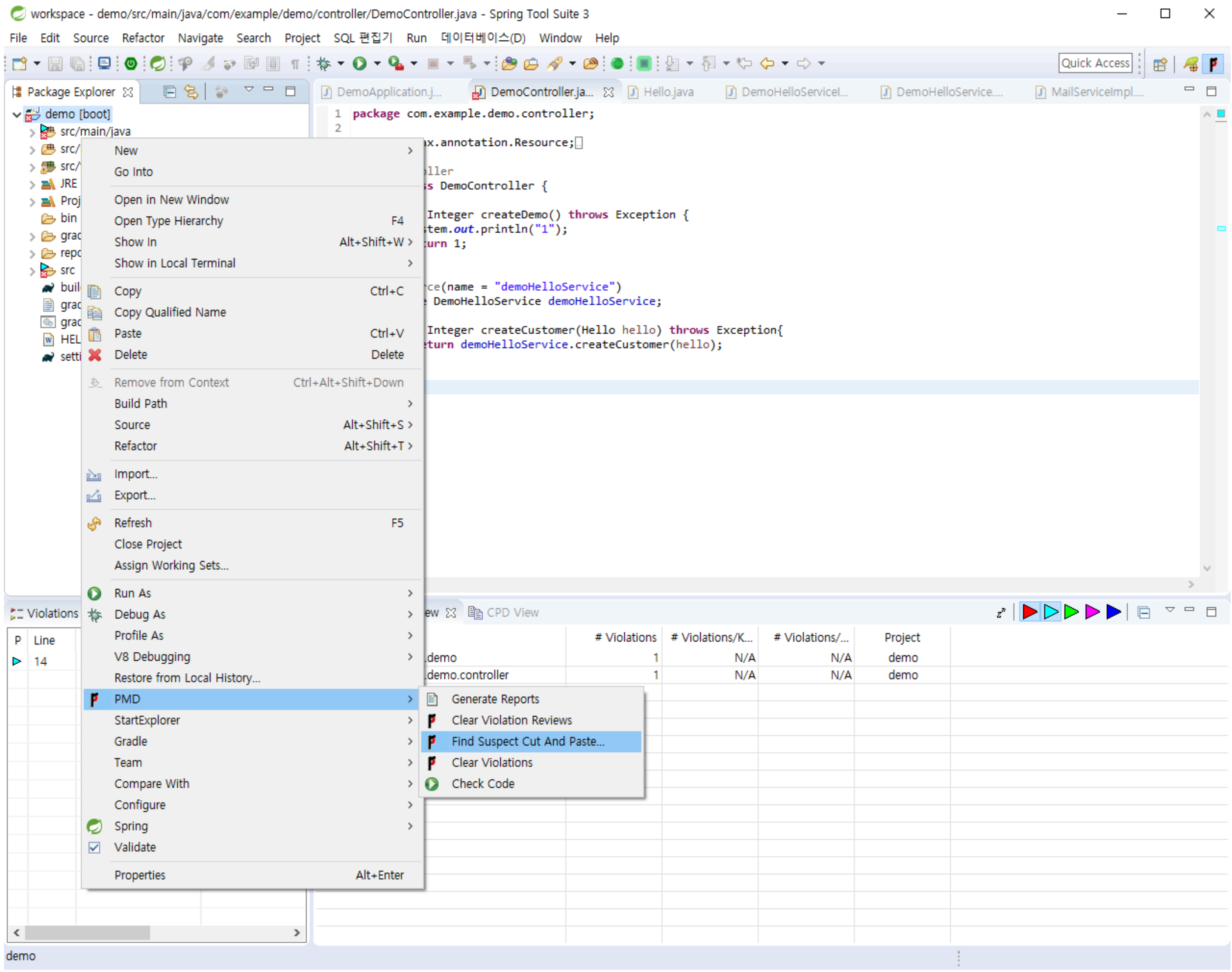
| Element | # Violations | # Violations/KLOC | # Violations/Method | Project |
|---|---|---|---|---|
| com.example.demo | 2 | 333.3 | 1.00 | demo |
| DemoApplication.java | 1 | 333.3 | 1.00 | demo |
| DemoApplication.java | 1 | 333.3 | 1.00 | demo |

위와 같이 Marked 된 Violation은 주석과 함께 제외된 것을 확인할 수 있다. (DemoController가 Violations Overview에서 제거됨)

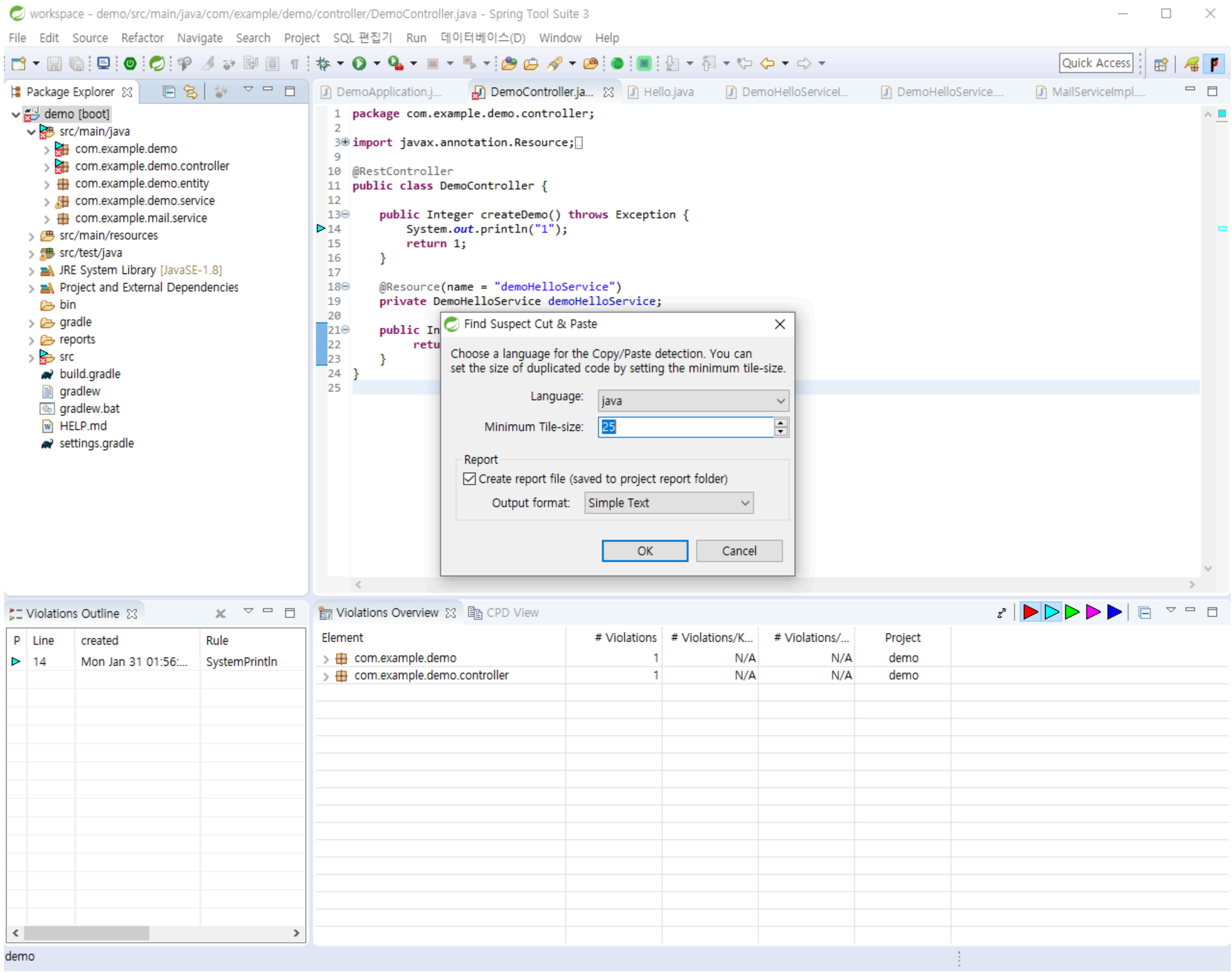
반대로 Clear Violation Review를 통해 주석을 제거하고 다시 Violation 상태로 돌릴 수 있다.

3) Find Suspect Cut And Paste Detector

CPD는 PMD의 기능 중 하나인 중복코드 검사 기능이다. window > Preferences > PMD > CPD preference에서 기본 설정(Minimum Tile Size)을 구성할 수 있다.



proejct 이름 > 우클릭 > PMD > Find Suspect Cut And Paste 클릭 시 하위 중복코드 여부를 검증한다.



```
package com.example.demo.controller;

import javax.annotation.Resource;

@RestController
public class DemoController {

    public Integer createDemo() throws Exception {
        System.out.println("1");
        return 1;
    }

    @Resource(name = "demoHelloService")
    private DemoHelloService demoHelloService;
```

Find Suspect Cut & Paste

Choose a language for the Copy/Paste detection. You can set the size of duplicated code by setting the minimum tile-size.

Language: java

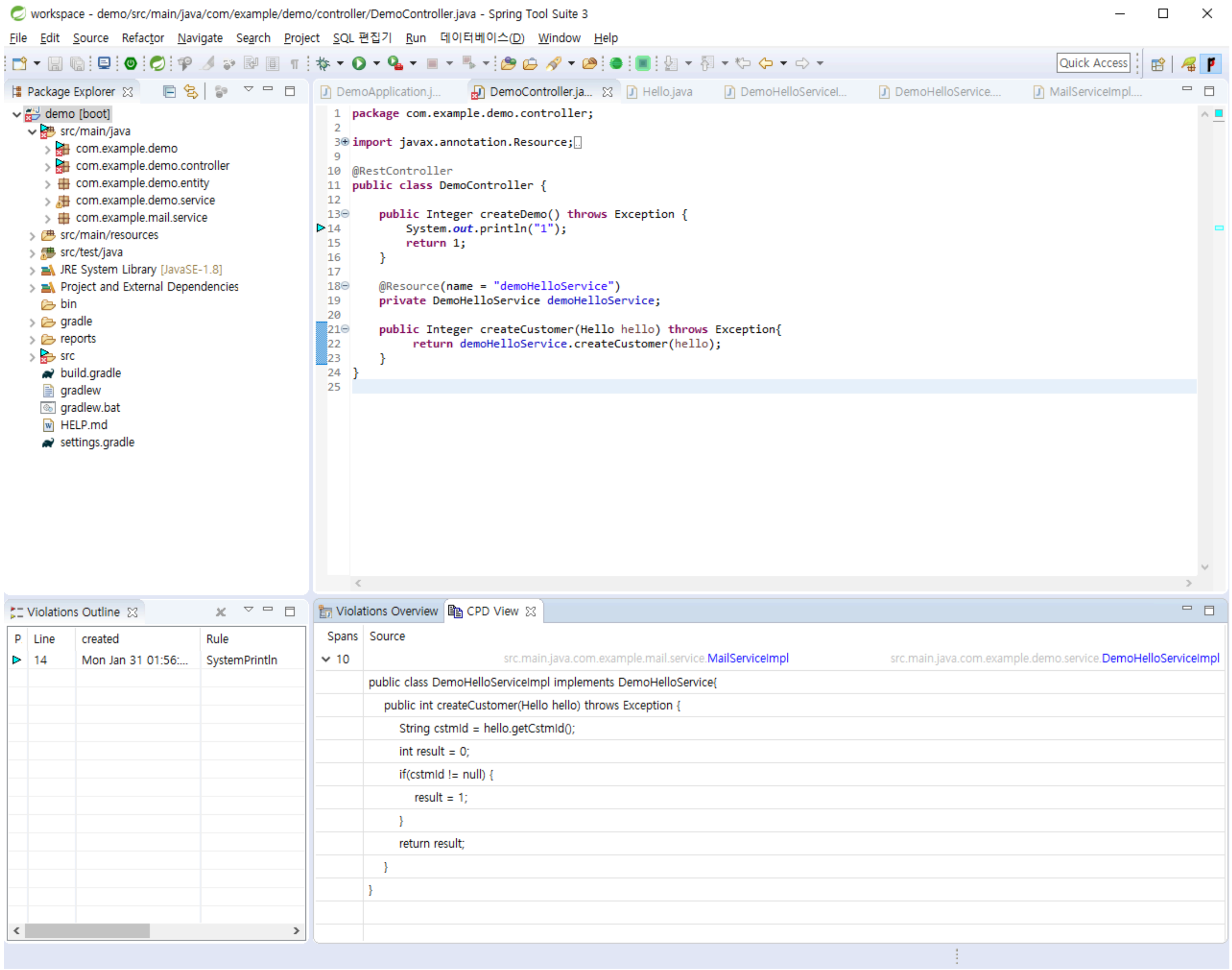
Minimum Tile-size: 25

Report

☑ Create report file (saved to project report folder)

Output format: Simple Text

위와 같이 Minimum Tile-size를 지정하고 OK를 클릭하면,


workspace - demo/src/main/java/com/example/demo/controller/DemoController.java - Spring Tool Suite 3
File Edit Source Refactor Navigate Search Project SQL 편집기 Run 데이터베이스(D) Window Help
Quick Access
Package Explorer
demo [boot]
src/main/java
com.example.demo
com.example.demo.controller
com.example.demo.entity
com.example.demo.service
com.example.mail.service
src/main/resources
src/test/java
JRE System Library [JavaSE-1.8]
Project and External Dependencies
bin
gradle
reports
src
build.gradle
gradlew
gradlew.bat
HELP.md
settings.gradle
DemoApplication.j... DemoController.ja... Hello.java DemoHelloServiceI... DemoHelloService.... MailServiceImpl....
Violations Outline
Violations Overview
CPD View

```java
package com.example.demo.controller;

import javax.annotation.Resource;

@RestController
public class DemoController {

    public Integer createDemo() throws Exception {
        System.out.println("1");
        return 1;
    }

    @Resource(name = "demoHelloService")
    private DemoHelloService demoHelloService;

    public Integer createCustomer(Hello hello) throws Exception{
         return demoHelloService.createCustomer(hello);
    }
}
```

| P | Line | created | Rule |
|---|---|---|---|
| ▷ | 14 | Mon Jan 31 01:56:... | SystemPrintln |

| Spans | Source | |
|---|---|---|
| 10 | src.main.java.com.example.mail.service.MailServiceImpl | src.main.java.com.example.demo.service.DemoHelloServiceImpl |

```java
public class DemoHelloServiceImpl implements DemoHelloService{
    public int createCustomer(Hello hello) throws Exception {
        String cstmId = hello.getCstmId();
        int result = 0;
        if(cstmId != null) {
            result = 1;
        }
        return result;
    }
}
```

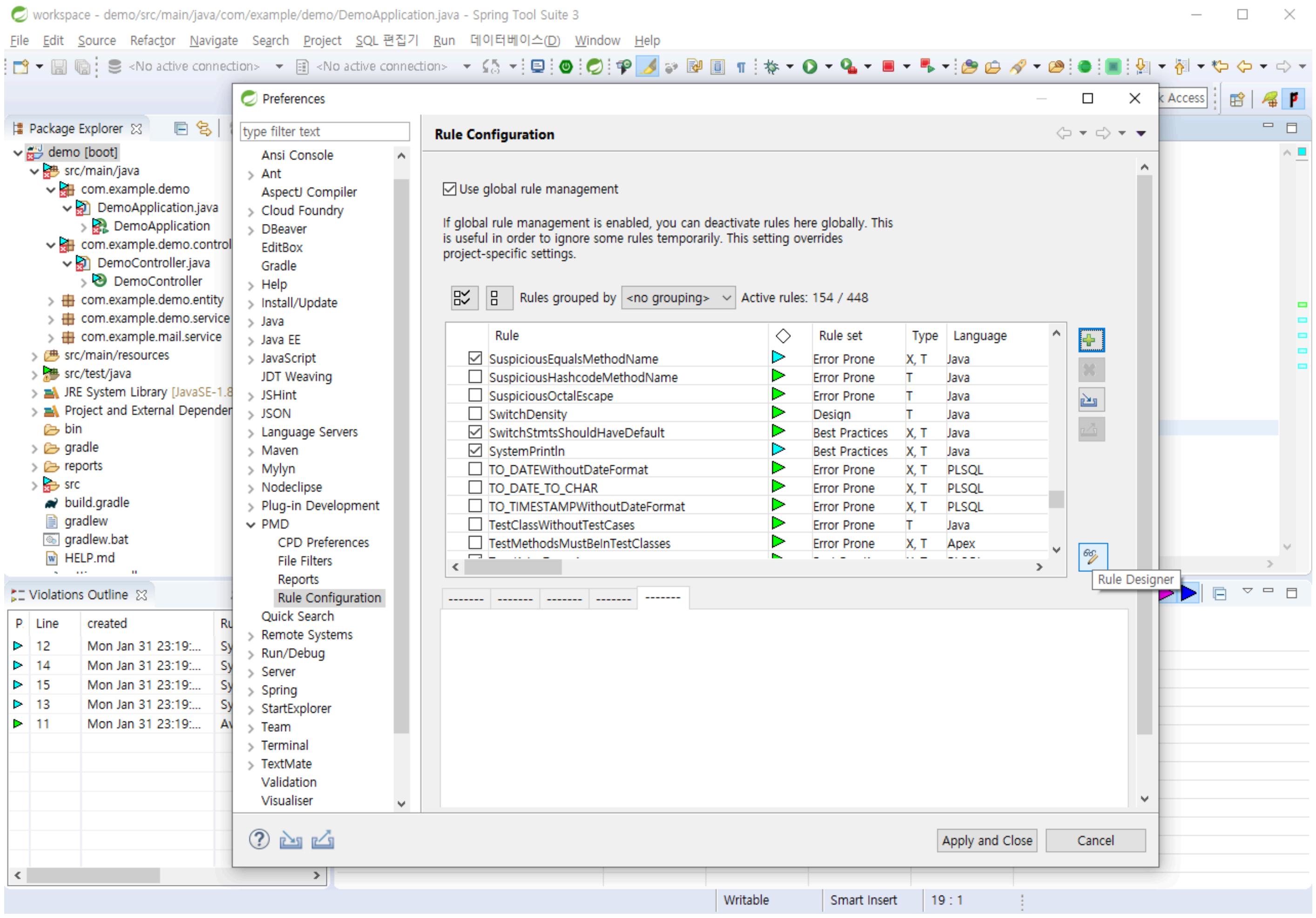
CPD View 탭을 통해 대상 클래스 목록과 중복 코드를 확인할 수 있다.

4) Add rule

새로운 Rule을 추가하는 방법에 대해 알아보자. Rule은 Java Code를 직접 작성하여 추가하는 방법과 XPath를 사용하여 Rule을 추가하는 방법이 있다. 이번 포스팅에서는 XPath를 이용하여 추가하는 방법에 대해 알아보자.

추가할 Rule은 Best Practices에 default로 추가되어 있는 SystemPrintln Rule을 커스터마이징하여 myManagedPrintln을 만들어보도록 하자. SystemPrintln은 System.out.print로 시작하는 구문 모두에 Critical Alert을 날리지만, 커스터마이징하여 생성한 myManagedPrintln은 System.out.print는 허용하고, System.out.println만 Critical Alert으로 날리는 룰이다.

먼저 Rule Designer를 실행하여 원하는 Rule이 정상 동작하는지 확인해 보도록 하자.



Rule Configuration

☑ Use global rule management

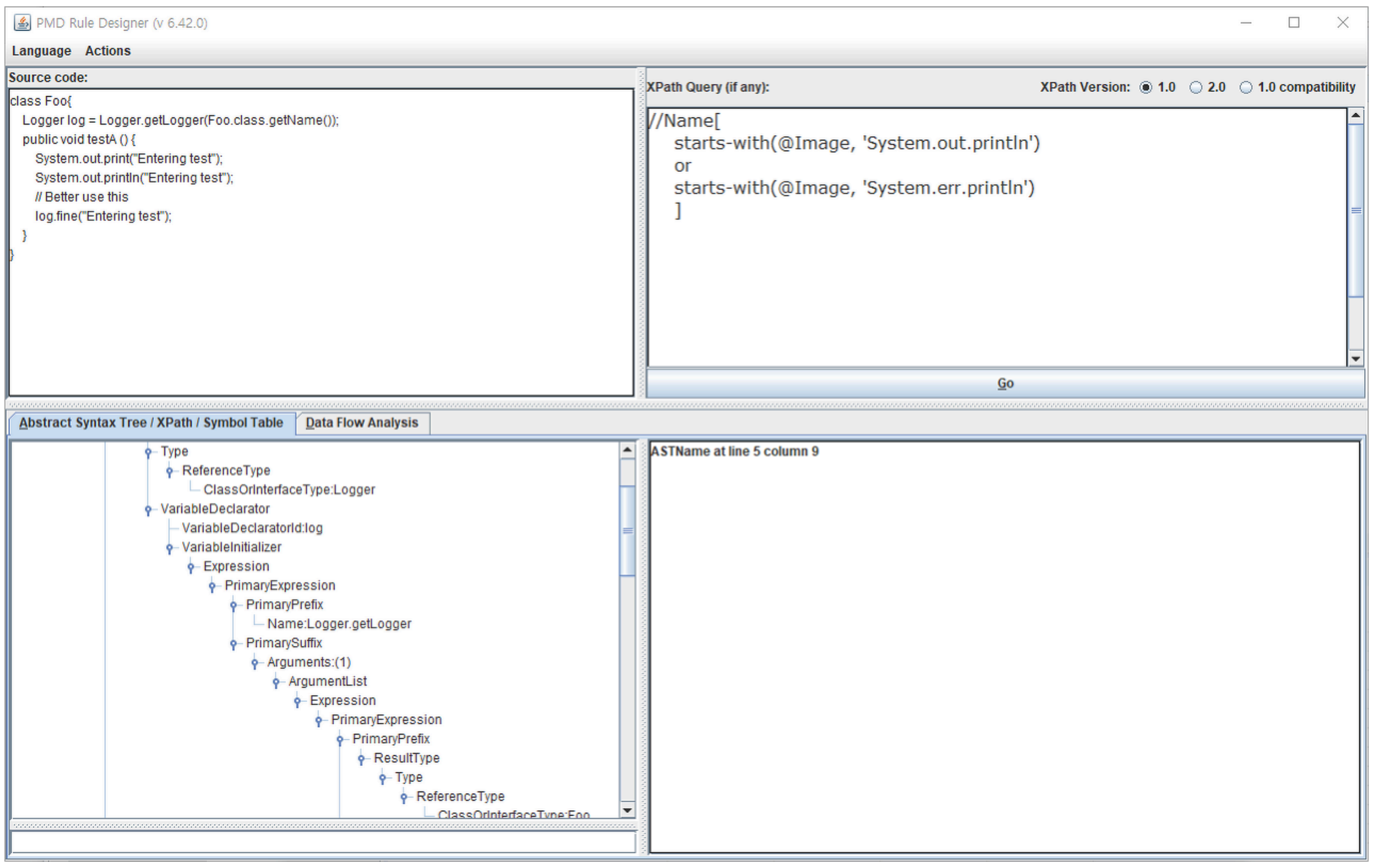
If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

Rules grouped by <no grouping> Active rules: 154 / 448

| | Rule | ◇ | Rule set | Type | Language |
|---|---|---|---|---|---|
| ☑ | SuspiciousEqualsMethodName | ▷ | Error Prone | X, T | Java |
| ☐ | SuspiciousHashcodeMethodName | ▷ | Error Prone | T | Java |
| ☐ | SuspiciousOctalEscape | ▷ | Error Prone | T | Java |
| ☐ | SwitchDensity | ▷ | Design | T | Java |
| ☑ | SwitchStmtsShouldHaveDefault | ▷ | Best Practices | X, T | Java |
| ☑ | SystemPrintln | ▷ | Best Practices | X, T | Java |
| ☐ | TO_DATEWithoutDateFormat | ▷ | Error Prone | X, T | PLSQL |
| ☐ | TO_DATE_TO_CHAR | ▷ | Error Prone | X, T | PLSQL |
| ☐ | TO_TIMESTAMPWithoutDateFormat | ▷ | Error Prone | X, T | PLSQL |
| ☐ | TestClassWithoutTestCases | ▷ | Error Prone | T | Java |
| ☐ | TestMethodsMustBeInTestClasses | ▷ | Error Prone | X, T | Apex |

| P | Line | created |
|---|---|---|
| ▷ | 12 | Mon Jan 31 23:19:... |
| ▷ | 14 | Mon Jan 31 23:19:... |
| ▷ | 15 | Mon Jan 31 23:19:... |
| ▷ | 13 | Mon Jan 31 23:19:... |
| ▷ | 11 | Mon Jan 31 23:19:... |

Window > Preferences > PMD > Rule Configuration > 오른쪽 중간 쯤 Rule Designer를 클릭한다.


PMD Rule Designer (v 6.42.0)
Language
Actions
Go
Abstract Syntax Tree / XPath / Symbol Table
Data Flow Analysis

Source code:

```
class Foo{
  Logger log = Logger.getLogger(Foo.class.getName());
  public void testA () {
    System.out.print("Entering test");
    System.out.println("Entering test");
    // Better use this
    log.fine("Entering test");
  }
}
```

XPath Query (if any): XPath Version: ● 1.0 ○ 2.0 ○ 1.0 compatibility

```
//Name[
    starts-with(@Image, 'System.out.println')
    or
    starts-with(@Image, 'System.err.println')
    ]
```

```
Type
  ReferenceType
    ClassOrInterfaceType:Logger
VariableDeclarator
  VariableDeclaratorId:log
  VariableInitializer
    Expression
      PrimaryExpression
        PrimaryPrefix
          Name:Logger.getLogger
        PrimarySuffix
          Arguments:(1)
            ArgumentList
              Expression
                PrimaryExpression
                  PrimaryPrefix
                    ResultType
                      Type
                        ReferenceType
                          ClassOrInterfaceType:Foo
```

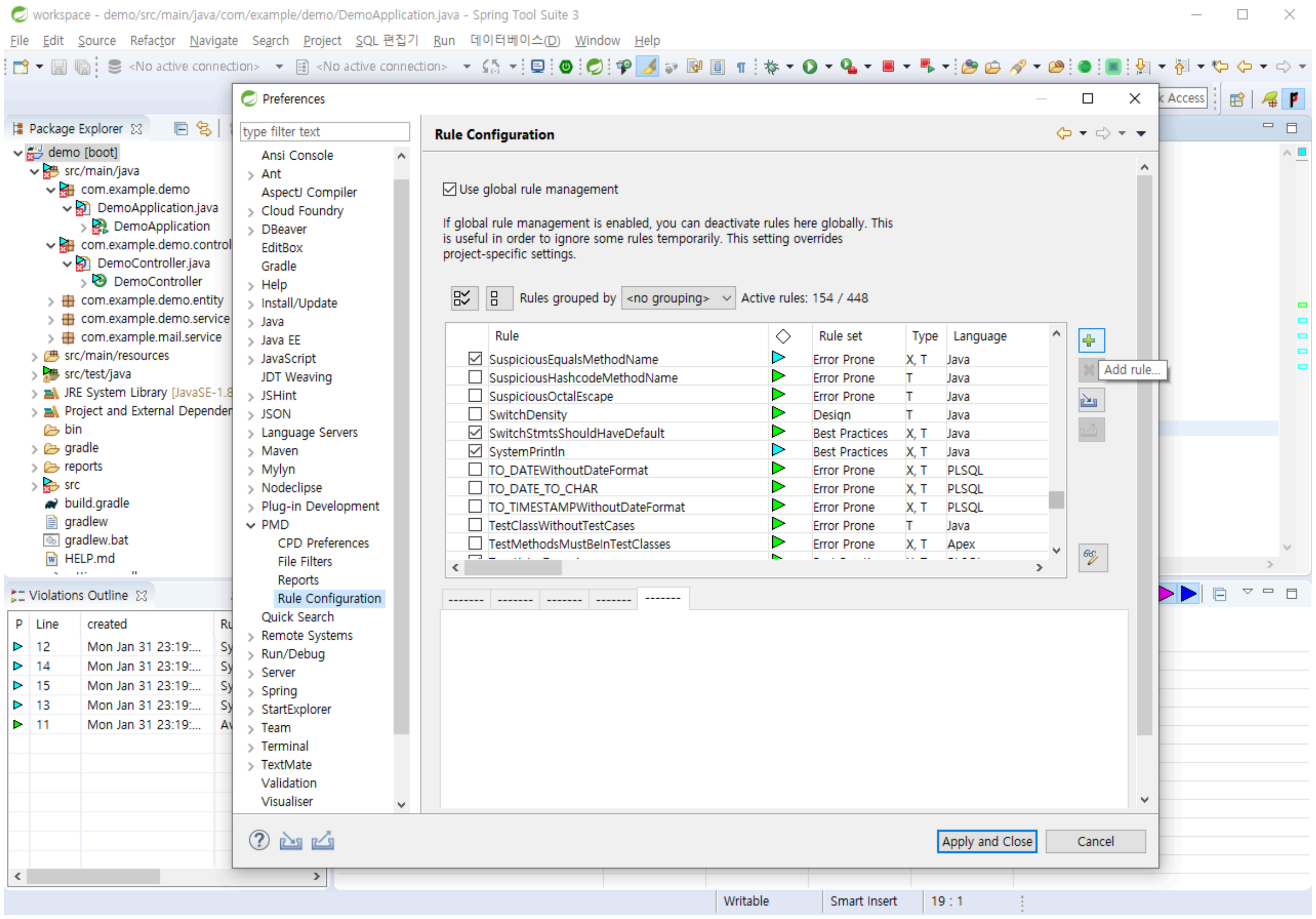
ASTName at line 5 column 9

화면에는 아래 네가지 구성요소로 이루어져 있다.

- Source Code : 테스트를 진행할 샘플 소스코드를 작성한다.
- XPath Query : Violation을 정의하는 XPath식을 작성한다.
- Abstract Syntax Tree / XPath / Symbol Table (Data Flow Analysis) : Source Code를 기준으로 AST 또는 Data Flow가 자동 작성된다.
- Result Window : Violation 결과가 출력된다.

위 샘플을 보면 System.out.print와 System.out.println이 사용되고 있으며, 이 중 System.out.println이 Violation에 걸려 Result Window에 하나의 결과가 출력되는 것을 확인할 수 있다.

테스트가 완료되면, Rule을 등록한다.



Rule Configuration

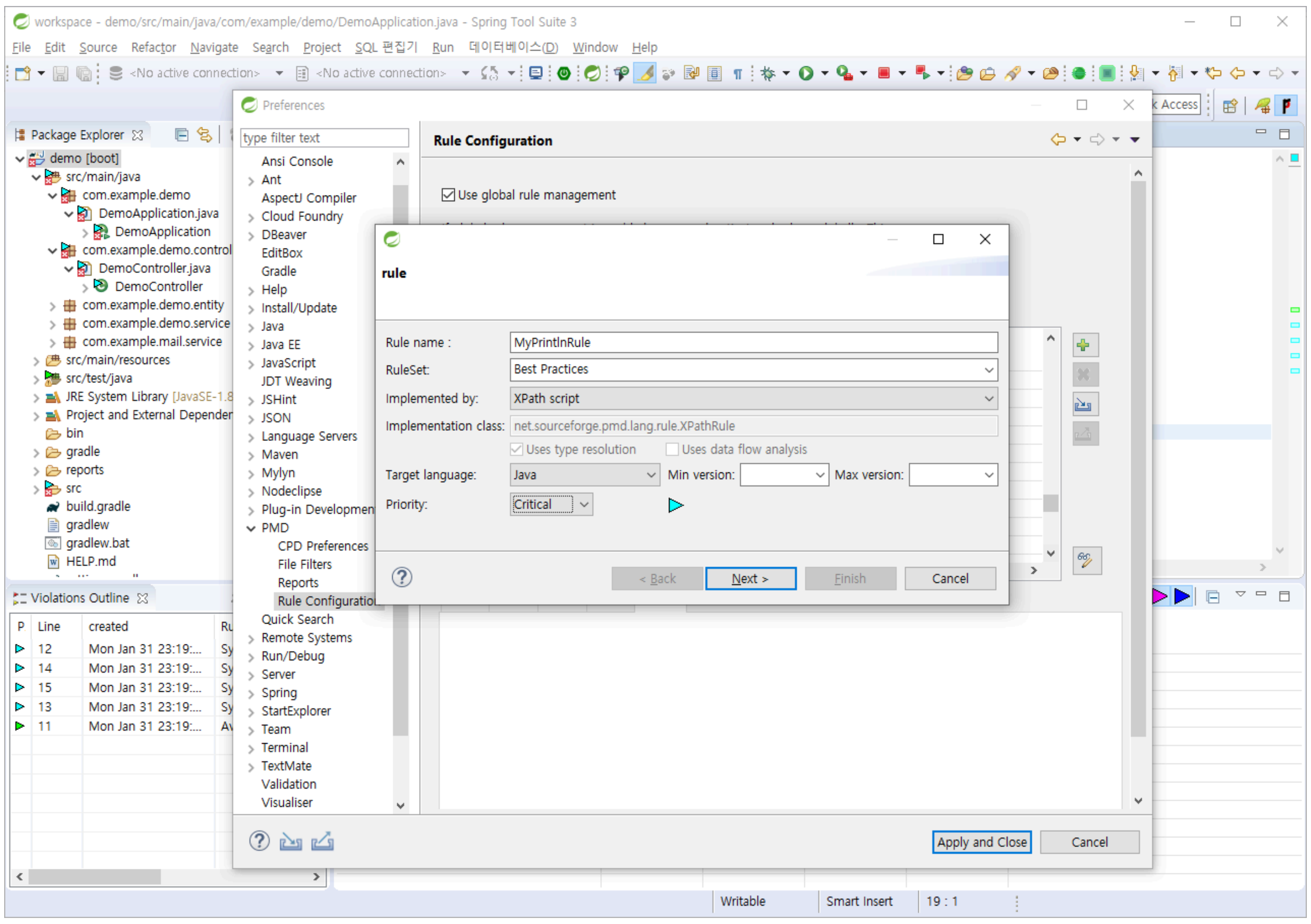
☑ Use global rule management

If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

Rules grouped by <no grouping> Active rules: 154 / 448

| | Rule | ◇ | Rule set | Type | Language |
|---|---|---|---|---|---|
| ☑ | SuspiciousEqualsMethodName | ▷ | Error Prone | X, T | Java |
| ☐ | SuspiciousHashcodeMethodName | ▷ | Error Prone | T | Java |
| ☐ | SuspiciousOctalEscape | ▷ | Error Prone | T | Java |
| ☐ | SwitchDensity | ▷ | Design | T | Java |
| ☑ | SwitchStmtsShouldHaveDefault | ▷ | Best Practices | X, T | Java |
| ☑ | SystemPrintln | ▷ | Best Practices | X, T | Java |
| ☐ | TO_DATEWithoutDateFormat | ▷ | Error Prone | X, T | PLSQL |
| ☐ | TO_DATE_TO_CHAR | ▷ | Error Prone | X, T | PLSQL |
| ☐ | TO_TIMESTAMPWithoutDateFormat | ▷ | Error Prone | X, T | PLSQL |
| ☐ | TestClassWithoutTestCases | ▷ | Error Prone | T | Java |
| ☐ | TestMethodsMustBeInTestClasses | ▷ | Error Prone | X, T | Apex |

Window > Preferences > PMD > Rule Configuration > 오른쪽 상단의 Add rule을 클릭한다.



workspace - demo/src/main/java/com/example/demo/DemoApplication.java - Spring Tool Suite 3
Preferences
Rule Configuration
☑ Use global rule management
rule
< Back
Next >
Finish
Cancel
Apply and Close
Cancel

Rule name : MyPrintlnRule

RuleSet: Best Practices

Implemented by: XPath script

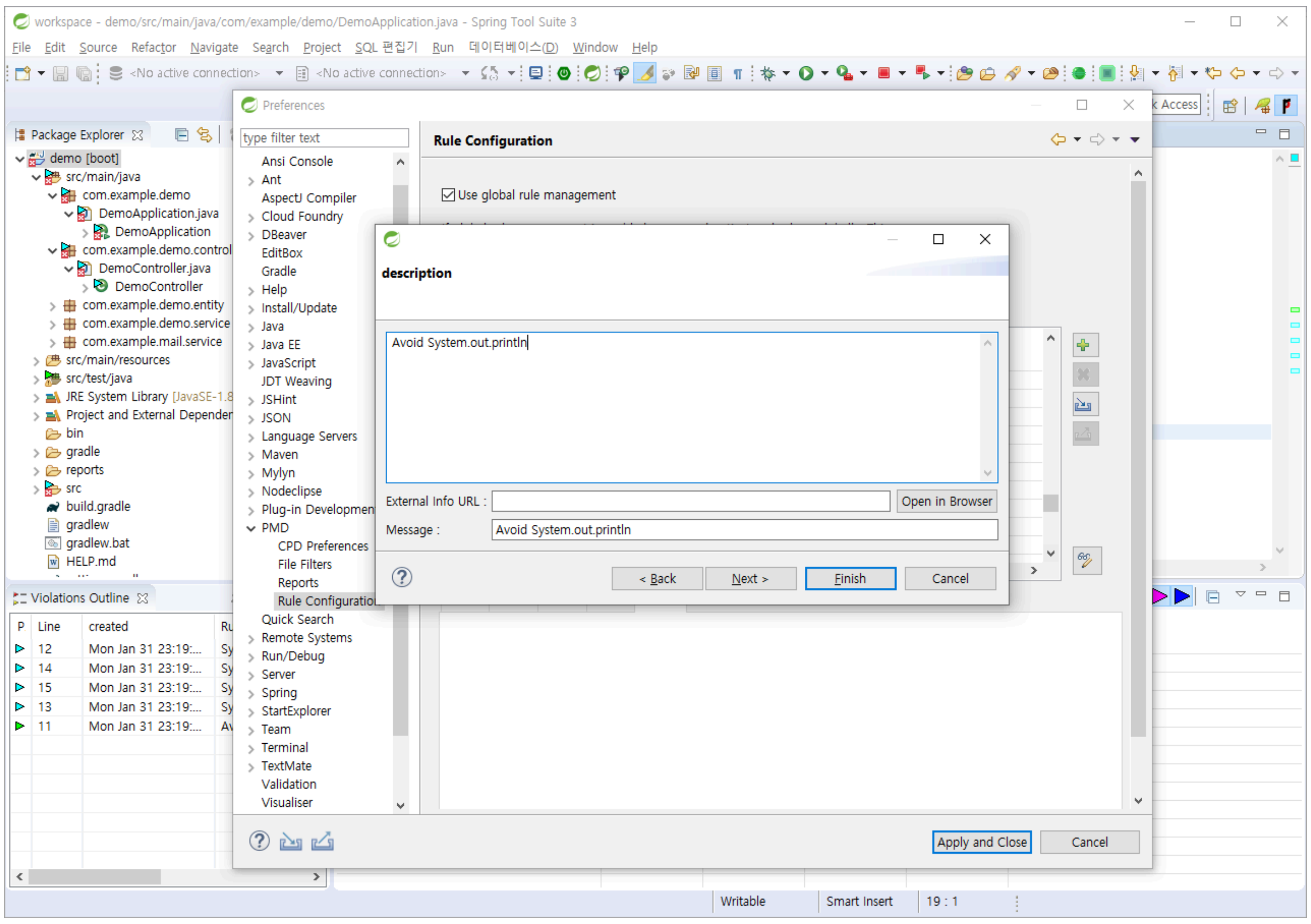
Implementation class: net.sourceforge.pmd.lang.rule.XPathRule

☑ Uses type resolution ☐ Uses data flow analysis

Target language: Java Min version: Max version:

Priority: Critical

- Rule name : Rule 이름 지정 (MyPrintlnRule)
- RuleSet : RuleSet 중 대상을 지정 (Best Practices)
- Implemented by : 앞서 Rule 추가를 위한 두가지 방식(Java Class or XPath script) 중 하나를 선택 (XPath script)
- Implementation class : XPath script의 경우 자동 매핑되며, Java Class의 경우 impl class 지정
- Target language : Language, min version, max version 지정 (Java)
- Priority : 5가지 단계 중 지정 (Critical)



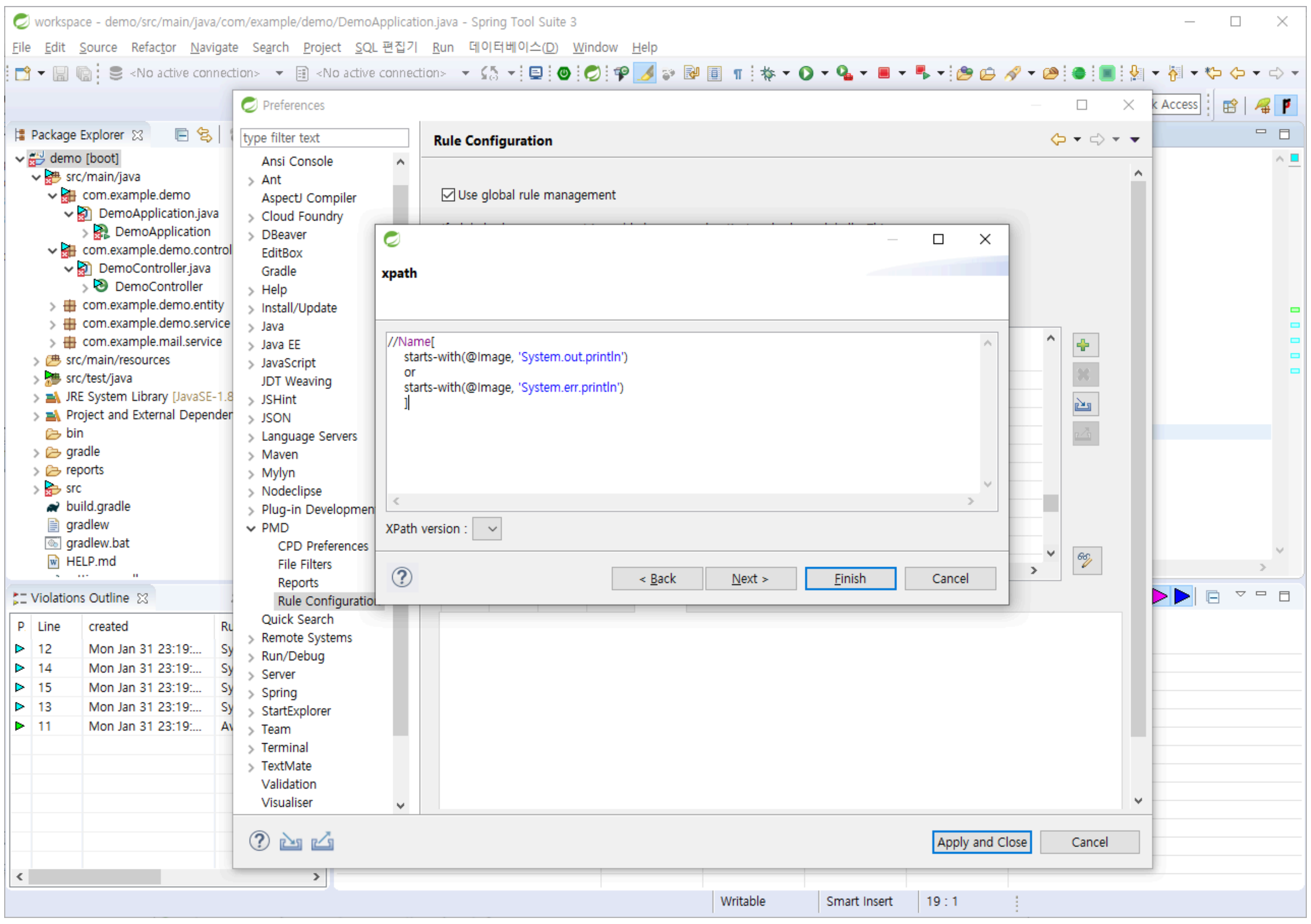
workspace - demo/src/main/java/com/example/demo/DemoApplication.java - Spring Tool Suite 3
Preferences
Rule Configuration
☑ Use global rule management
description
Open in Browser
< Back
Next >
Finish
Cancel
Apply and Close
Cancel

Avoid System.out.println

External Info URL :

Message : Avoid System.out.println

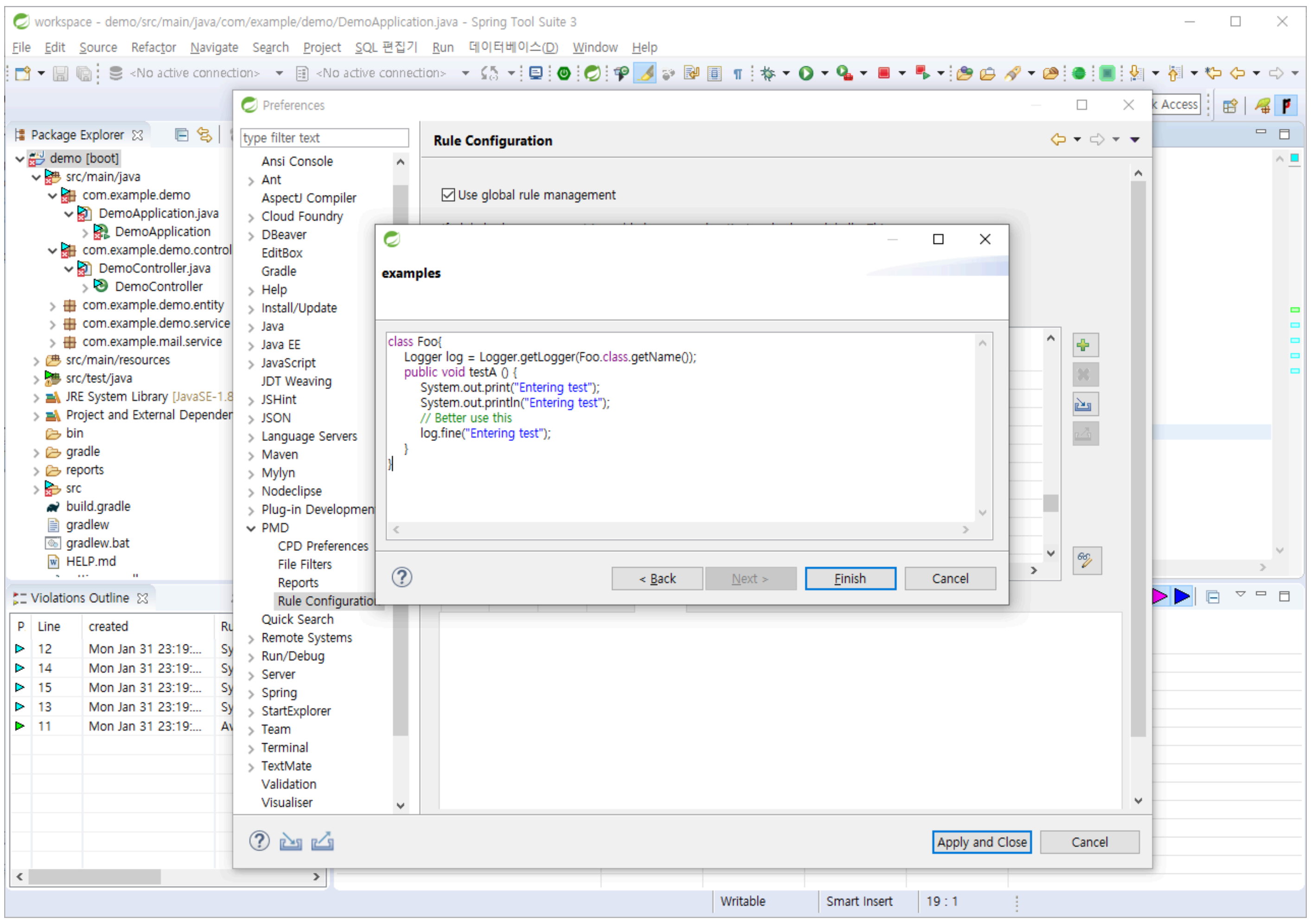
- Window : Description 작성
- Message : 참고 문구



```
//Name[
  starts-with(@Image, 'System.out.println')
  or
  starts-with(@Image, 'System.err.println')
]
```

- Window : XPath 식 작성

```
//Name[
    starts-with(@Image, 'System.out.println')
    or
    starts-with(@Image, 'System.err.println')
    ]
```

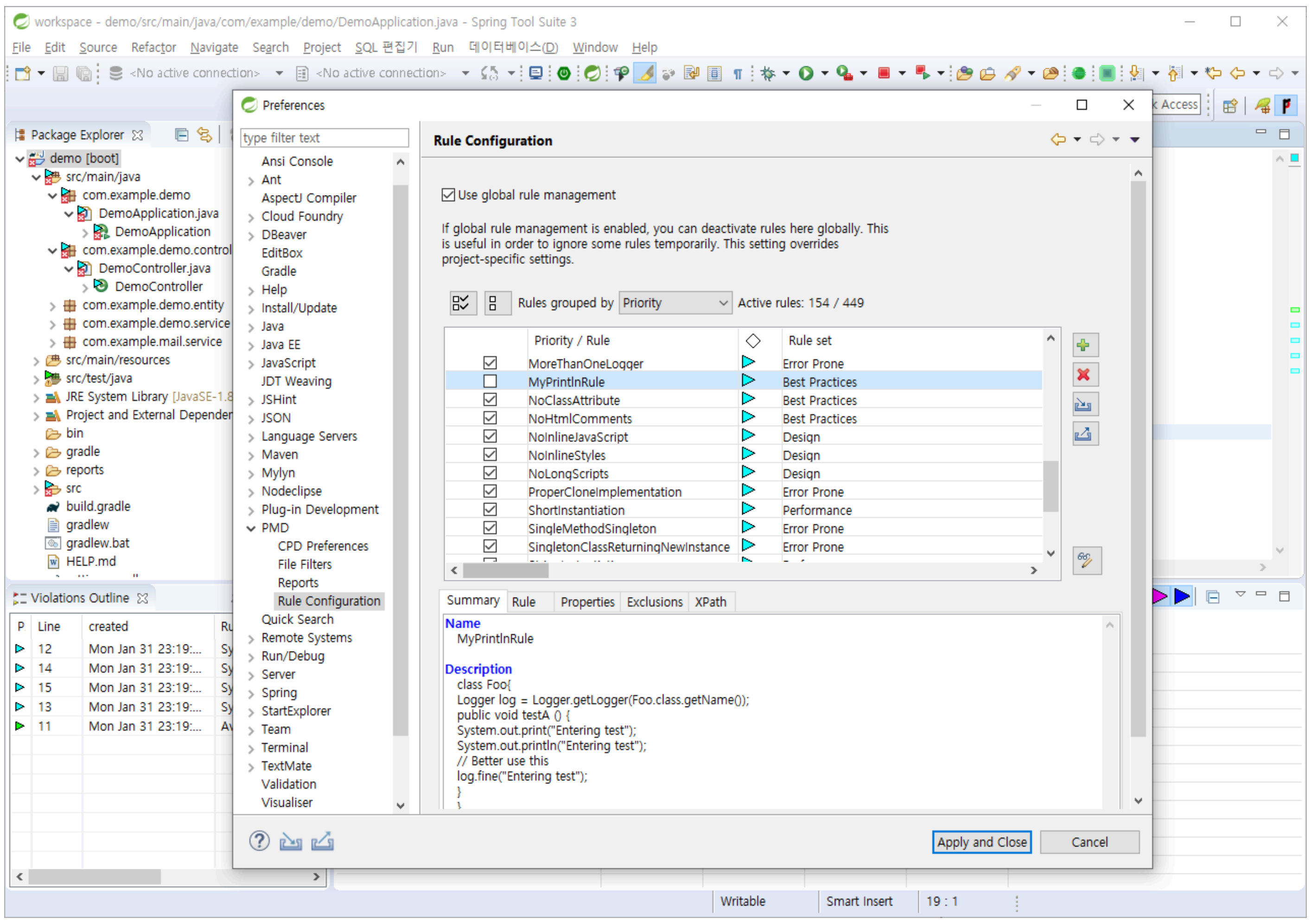


```
class Foo{
  Logger log = Logger.getLogger(Foo.class.getName());
  public void testA () {
    System.out.print("Entering test");
    System.out.println("Entering test");
    // Better use this
    log.fine("Entering test");
  }
}
```

- Window : Sample Code 작성

```
class Foo{
    Logger log = Logger.getLogger(Foo.class.getName());
    public void testA () {
        System.out.println("Entering test");
        // Better use this
        log.fine("Entering test");
    }
}
```

Finish를 클릭하면, 아래와 같이 Rule이 추가된 것을 확인할 수 있다.



Rule Configuration

☑ Use global rule management

If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

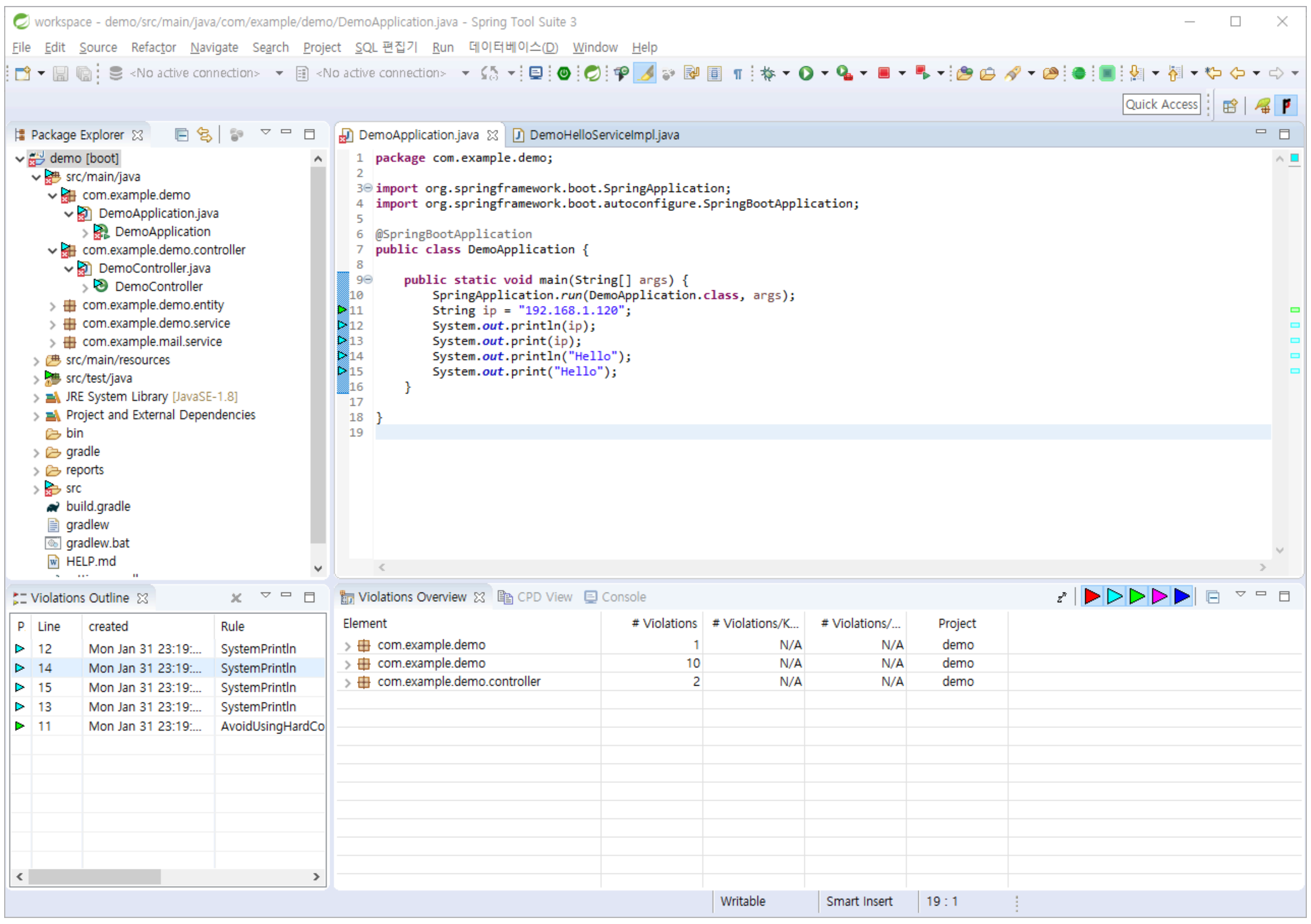
Rules grouped by Priority   Active rules: 154 / 449

| | Priority / Rule | Rule set |
|---|---|---|
| ☑ | MoreThanOneLogger | Error Prone |
| □ | MyPrintlnRule | Best Practices |
| ☑ | NoClassAttribute | Best Practices |
| ☑ | NoHtmlComments | Best Practices |
| ☑ | NoInlineJavaScript | Design |
| ☑ | NoInlineStyles | Design |
| ☑ | NoLongScripts | Design |
| ☑ | ProperCloneImplementation | Error Prone |
| ☑ | ShortInstantiation | Performance |
| ☑ | SingleMethodSingleton | Error Prone |
| ☑ | SingletonClassReturningNewInstance | Error Prone |

Name
MyPrintlnRule

Description
```
class Foo{
Logger log = Logger.getLogger(Foo.class.getName());
public void testA () {
System.out.print("Entering test");
System.out.println("Entering test");
// Better use this
log.fine("Entering test");
}
}
```

동작을 검증하기에 앞서 SystemPrintln Rule이 적용된 상태로 현재 소스코드의 Inspection을 실행해 보도록 하자.



```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
        String ip = "192.168.1.120";
        System.out.println(ip);
        System.out.print(ip);
        System.out.println("Hello");
        System.out.print("Hello");
    }

}
```

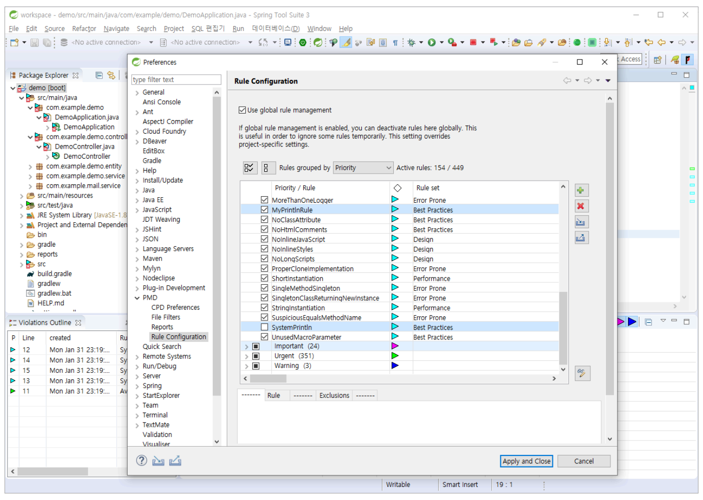
Violations Outline

| P | Line | created | Rule |
|---|---|---|---|
| ▷ | 12 | Mon Jan 31 23:19:... | SystemPrintln |
| ▷ | 14 | Mon Jan 31 23:19:... | SystemPrintln |
| ▷ | 15 | Mon Jan 31 23:19:... | SystemPrintln |
| ▷ | 13 | Mon Jan 31 23:19:... | SystemPrintln |
| ▷ | 11 | Mon Jan 31 23:19:... | AvoidUsingHardCo |

Violations Overview

| Element | # Violations | # Violations/K... | # Violations/... | Project |
|---|---|---|---|---|
| com.example.demo | 1 | N/A | N/A | demo |
| com.example.demo | 10 | N/A | N/A | demo |
| com.example.demo.controller | 2 | N/A | N/A | demo |

위와 같이 System.out.println, System.out.print 모두 Critical Inspection Alert이 나타나는 것을 확인할 수 있다. 그럼 이제 추가한 Rule을 Test하기 위해 SystemPrintln은 Deselect하고, 추가한 MyPrintlnRule을 Select한 후 Check Code를 실행해 보자.



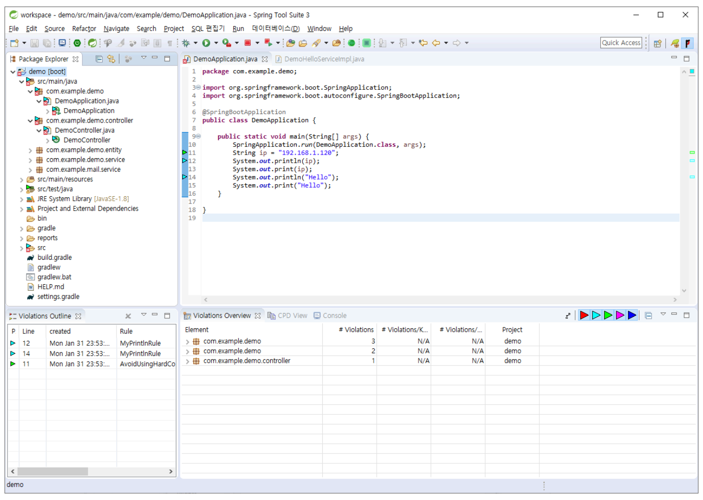
Rule Configuration

☑ Use global rule management

If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

Rules grouped by Priority   Active rules: 154 / 449

| | Priority / Rule | Rule set |
|---|---|---|
| ☑ | MoreThanOneLogger | Error Prone |
| ☑ | MyPrintlnRule | Best Practices |
| ☑ | NoClassAttribute | Best Practices |
| ☑ | NoHtmlComments | Best Practices |
| ☑ | NoInlineJavaScript | Design |
| ☑ | NoInlineStyles | Design |
| ☑ | NoLongScripts | Design |
| ☑ | ProperCloneImplementation | Error Prone |
| ☑ | ShortInstantiation | Performance |
| ☑ | SingleMethodSingleton | Error Prone |
| ☑ | SingletonClassReturningNewInstance | Error Prone |
| ☑ | StringInstantiation | Performance |
| ☑ | SuspiciousEqualsMethodName | Error Prone |
| □ | SystemPrintln | Best Practices |
| ☑ | UnusedMacroParameter | Best Practices |
| ■ | Important (24) | |
| ■ | Urgent (351) | |
| ■ | Warning (3) | |



workspace - demo/src/main/java/com/example/demo/DemoApplication.java - Spring Tool Suite 3

```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
        String ip = "192.168.1.120";
        System.out.println(ip);
        System.out.print(ip);
        System.out.println("Hello");
        System.out.print("Hello");
    }

}
```

Violations Outline

| P | Line | created | Rule |
|---|---|---|---|
| ▷ | 12 | Mon Jan 31 23:53:... | MyPrintlnRule |
| ▷ | 14 | Mon Jan 31 23:53:... | MyPrintlnRule |
| ▷ | 11 | Mon Jan 31 23:53:... | AvoidUsingHardCo |

Violations Overview

| Element | # Violations | # Violations/K... | # Violations/... | Project |
|---|---|---|---|---|
| com.example.demo | 3 | N/A | N/A | demo |
| com.example.demo | 2 | N/A | N/A | demo |
| com.example.demo.controller | 1 | N/A | N/A | demo |

위와 같이 Println만 Inspection Alert이 나타나는 것을 확인할 수 있다. 이와 같이 룰을 커스터마이징하여 나만의 룰셋을 등록관리할 수 있다.

추가로 IPv4에 대한 하드코딩 제한 Rule - MyAvoidIPv4HardCodingRule, 주민번호에 대한 하드코딩 제한 Rule - MyAvoidRRNumHardCodingRule을 생성하고 적용해 보자.

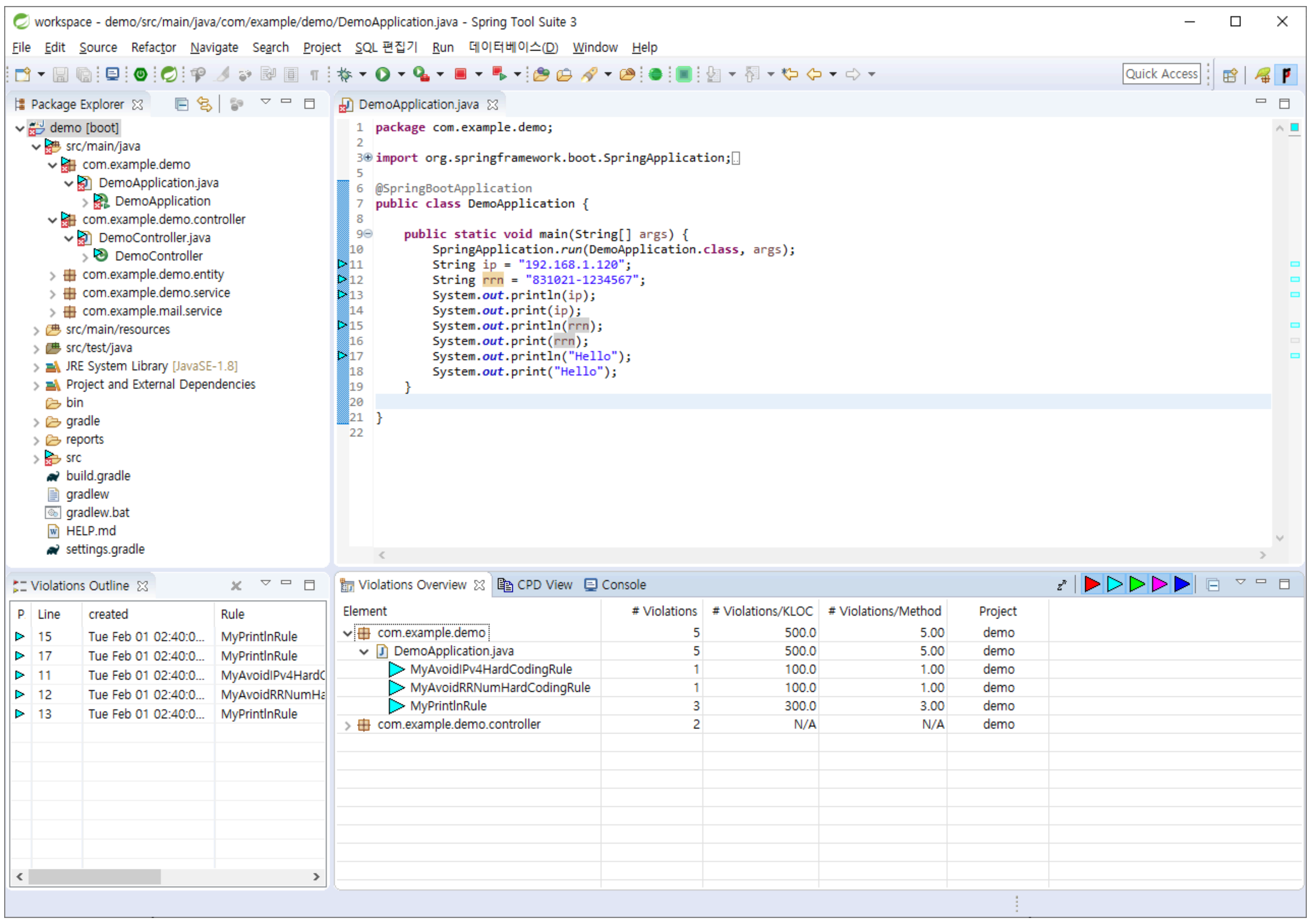
[MyAvoidIPv4HardCodingRule]

```
[XPath]
//PrimaryExpression/PrimaryPrefix/Literal
 [matches(@Image,
  '(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)₩.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)'
 )]

 [Example]
 class DemoApplication {

        public static void main(String[] args) {
                SpringApplication.run(DemoApplication.class, args);
                String ip = "192.168.1.120";
                System.out.println(ip);
                System.out.print(ip);
        }
}
```

[MyAvoidRRNumHardCodingRule]

```
[XPath]
//PrimaryExpression/PrimaryPrefix/Literal
 [matches(@Image,
  '(?:[0-9]{2}(?:0[1-9]|1[0-2])(?:0[1-9]|[1,2][0-9]|3[0,1]))-[1-4][0-9]{6}'
 )]

[Example]
class DemoApplication {

        public static void main(String[] args) {
                SpringApplication.run(DemoApplication.class, args);
                String rrn = "831021-1234567";
                System.out.println(rrn);
                System.out.print(rrn);
        }
}
```

결과는 아래와 같다.



workspace - demo/src/main/java/com/example/demo/DemoApplication.java - Spring Tool Suite 3

```java
package com.example.demo;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
        String ip = "192.168.1.120";
        String rrn = "831021-1234567";
        System.out.println(ip);
        System.out.print(ip);
        System.out.println(rrn);
        System.out.print(rrn);
        System.out.println("Hello");
        System.out.print("Hello");
    }

}
```

Violations Outline

| P | Line | created | Rule |
|---|---|---|---|
| ▷ | 15 | Tue Feb 01 02:40:0... | MyPrintlnRule |
| ▷ | 17 | Tue Feb 01 02:40:0... | MyPrintlnRule |
| ▷ | 11 | Tue Feb 01 02:40:0... | MyAvoidIPv4HardC |
| ▷ | 12 | Tue Feb 01 02:40:0... | MyAvoidRRNumHa |
| ▷ | 13 | Tue Feb 01 02:40:0... | MyPrintlnRule |

Violations Overview

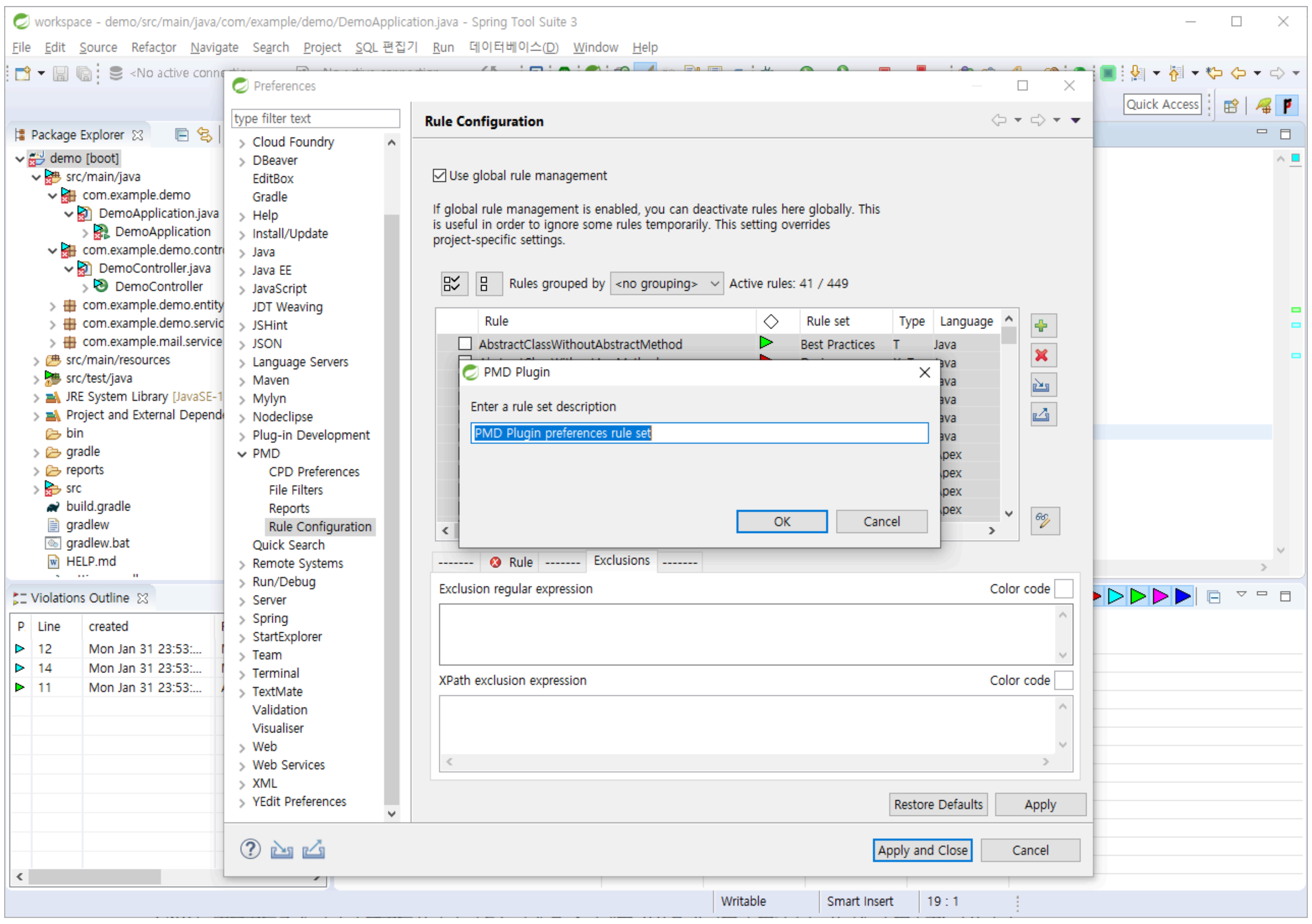
| Element | # Violations | # Violations/KLOC | # Violations/Method | Project |
|---|---|---|---|---|
| com.example.demo | 5 | 500.0 | 5.00 | demo |
| DemoApplication.java | 5 | 500.0 | 5.00 | demo |
| MyAvoidIPv4HardCodingRule | 1 | 100.0 | 1.00 | demo |
| MyAvoidRRNumHardCodingRule | 1 | 100.0 | 1.00 | demo |
| MyPrintlnRule | 3 | 300.0 | 3.00 | demo |
| com.example.demo.controller | 2 | N/A | N/A | demo |

이와 같이 XPath 식을 추가하여 손쉽게 Rule을 생성할 수 있다. (XPath Rule)

5) Import rule set / Export selected rules

이제 생성한 Ruleset을 다른 개발자나 다른 환경으로 이관하기 위해 Export / Import하는 과정에 대해 알아보도록 하자.

먼저, Export Selected rules이다. Export는 앞에 체크되어 있는 Rule을 기준으로 추출하는 것이 아닌 선택되어진 Rule을 추출하는 것에 유의하여 Ruleset 파일을 생성한다.



Rule Configuration

☑ Use global rule management

If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

Rules grouped by <no grouping> Active rules: 41 / 449

| | Rule | | Rule set | Type | Language |
|---|---|---|---|---|---|
| □ | AbstractClassWithoutAbstractMethod | ▷ | Best Practices | T | Java |

PMD Plugin

Enter a rule set description

PMD Plugin preferences rule set

참고 차 전체 Ruleset 파일을 업로드 해두도록 하자.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ruleset xmlns="http://pmd.sourceforge.net/ruleset/2.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         name="pmd-test-ruleset"
         xsi:schemaLocation="http://pmd.sourceforge.net/ruleset/2.0.0 https://pmd.sourceforge.io/ruleset_2_0_0.xsd">
   <description>PMD Plugin preferences rule set</description>
   <rule ref="category/java/bestpractices.xml/AbstractClassWithoutAbstractMethod"/>
   <rule ref="category/java/design.xml/AbstractClassWithoutAnyMethod"/>
   <rule ref="category/java/codestyle.xml/AbstractNaming"/>
   <rule ref="category/java/bestpractices.xml/AccessorClassGeneration"/>
```

```
<rule ref="category/java/bestpractices.xml/AccessorMethodGeneration"/>
<rule ref="category/java/performance.xml/AddEmptyString"/>
<rule ref="category/apex/bestpractices.xml/ApexAssertionsShouldIncludeMessage"/>
<rule ref="category/apex/security.xml/ApexBadCrypto"/>
<rule ref="category/apex/security.xml/ApexCRUDViolation"/>
<rule ref="category/apex/errorprone.xml/ApexCSRF"/>
<rule ref="category/apex/security.xml/ApexDangerousMethods"/>
<rule ref="category/apex/documentation.xml/ApexDoc"/>
<rule ref="category/apex/security.xml/ApexInsecureEndpoint"/>
<rule ref="category/apex/security.xml/ApexOpenRedirect"/>
<rule ref="category/apex/security.xml/ApexSOQLInjection"/>
<rule ref="category/apex/security.xml/ApexSharingViolations"/>
<rule ref="category/apex/security.xml/ApexSuggestUsingNamedCred"/>
<rule ref="category/apex/bestpractices.xml/ApexUnitTestClassShouldHaveAsserts"/>
<rule ref="category/apex/bestpractices.xml/ApexUnitTestMethodShouldHaveIsTestAnnotation"/>
<rule ref="category/apex/bestpractices.xml/ApexUnitTestShouldNotUseSeeAllDataTrue"/>
<rule ref="category/apex/security.xml/ApexXSSFromEscapeFalse"/>
<rule ref="category/apex/security.xml/ApexXSSFromURLParam"/>
<rule ref="category/java/performance.xml/AppendCharacterWithChar"/>
<rule ref="category/java/bestpractices.xml/ArrayIsStoredDirectly"/>
<rule ref="category/ecmascript/codestyle.xml/AssignmentInOperand"/>
<rule ref="category/java/errorprone.xml/AssignmentInOperand"/>
<rule ref="category/java/errorprone.xml/AssignmentToNonFinalStatic"/>
<rule ref="category/java/codestyle.xml/AtLeastOneConstructor"/>
<rule ref="category/java/errorprone.xml/AvoidAccessibilityAlteration"/>
<rule ref="category/java/performance.xml/AvoidArrayLoops"/>
<rule ref="category/java/errorprone.xml/AvoidAssertAsIdentifier"/>
<rule ref="category/xsl/performance.xml/AvoidAxisNavigation"/>
<rule ref="category/java/errorprone.xml/AvoidBranchingStatementAsLastInLoop"/>
<rule ref="category/java/performance.xml/AvoidCalendarDateCreation"/>
<rule ref="category/java/errorprone.xml/AvoidCallingFinalize"/>
<rule ref="category/java/design.xml/AvoidCatchingGenericException"/>
<rule ref="category/java/errorprone.xml/AvoidCatchingNPE"/>
<rule ref="category/java/errorprone.xml/AvoidCatchingThrowable"/>
<rule ref="category/apex/performance.xml/AvoidDebugStatements"/>
<rule ref="category/java/errorprone.xml/AvoidDecimalLiteralsInBigDecimalConstructor"/>
<rule ref="category/apex/design.xml/AvoidDeeplyNestedIfStmts"/>
<rule ref="category/java/design.xml/AvoidDeeplyNestedIfStmts"/>
<rule ref="category/vm/design.xml/AvoidDeeplyNestedIfStmts"/>
<rule ref="category/apex/errorprone.xml/AvoidDirectAccessTriggerMap"/>
<rule ref="category/apex/performance.xml/AvoidDmlStatementsInLoops"/>
<rule ref="category/java/codestyle.xml/AvoidDollarSigns"/>
<rule ref="category/java/errorprone.xml/AvoidDuplicateLiterals"/>
<rule ref="category/java/errorprone.xml/AvoidEnumAsIdentifier"/>
<rule ref="category/java/errorprone.xml/AvoidFieldNameMatchingMethodName"/>
<rule ref="category/java/errorprone.xml/AvoidFieldNameMatchingTypeName"/>
<rule ref="category/java/performance.xml/AvoidFileStream"/>
<rule ref="category/java/codestyle.xml/AvoidFinalLocalVariable"/>
<rule ref="category/apex/bestpractices.xml/AvoidGlobalModifier"/>
<rule ref="category/apex/errorprone.xml/AvoidHardcodingId"/>
<rule ref="category/java/errorprone.xml/AvoidInstanceofChecksInCatchClause"/>
<rule ref="category/java/performance.xml/AvoidInstantiatingObjectsInLoops"/>
<rule ref="category/java/errorprone.xml/AvoidLiteralsInIfCondition"/>
<rule ref="category/apex/bestpractices.xml/AvoidLogicInTrigger"/>
<rule ref="category/java/errorprone.xml/AvoidLosingExceptionInformation"/>
<rule ref="category/java/bestpractices.xml/AvoidMessageDigestField"/>
<rule ref="category/java/errorprone.xml/AvoidMultipleUnaryOperators"/>
<rule ref="category/apex/errorprone.xml/AvoidNonExistentAnnotations"/>
<rule ref="category/java/codestyle.xml/AvoidPrefixingMethodParameters"/>
<rule ref="category/java/bestpractices.xml/AvoidPrintStackTrace"/>
<rule ref="category/java/codestyle.xml/AvoidProtectedFieldInFinalClass"/>
<rule ref="category/java/codestyle.xml/AvoidProtectedMethodInFinalClassNotExtending"/>
<rule ref="category/java/bestpractices.xml/AvoidReassigningCatchVariables"/>
<rule ref="category/java/bestpractices.xml/AvoidReassigningLoopVariables"/>
<rule ref="category/java/bestpractices.xml/AvoidReassigningParameters"/>
<rule ref="category/vm/bestpractices.xml/AvoidReassigningParameters"/>
<rule ref="category/java/design.xml/AvoidRethrowingException"/>
<rule ref="category/apex/performance.xml/AvoidSoqlInLoops"/>
<rule ref="category/apex/performance.xml/AvoidSoslInLoops"/>
<rule ref="category/java/bestpractices.xml/AvoidStringBufferField"/>
<rule ref="category/java/multithreading.xml/AvoidSynchronizedAtMethodLevel"/>
<rule ref="category/plsql/codestyle.xml/AvoidTabCharacter"/>
<rule ref="category/java/multithreading.xml/AvoidThreadGroup"/>
<rule ref="category/java/design.xml/AvoidThrowingNewInstanceOfSameException"/>
<rule ref="category/java/design.xml/AvoidThrowingNullPointerException"/>
<rule ref="category/java/design.xml/AvoidThrowingRawExceptionTypes"/>
<rule ref="category/ecmascript/errorprone.xml/AvoidTrailingComma"/>
<rule ref="category/java/design.xml/AvoidUncheckedExceptionsInSignatures"/>
<rule ref="category/java/bestpractices.xml/AvoidUsingHardCodedIP"/>
<rule ref="category/java/codestyle.xml/AvoidUsingNativeCode"/>
<rule ref="category/java/errorprone.xml/AvoidUsingOctalValues"/>
<rule ref="category/java/performance.xml/AvoidUsingShortType"/>
<rule ref="category/java/multithreading.xml/AvoidUsingVolatile"/>
<rule ref="category/ecmascript/bestpractices.xml/AvoidWithStatement"/>
```

```
<rule ref="category/java/errorprone.xml/BeanMembersShouldSerialize"/>
<rule ref="category/java/performance.xml/BigIntegerInstantiation"/>
<rule ref="category/java/codestyle.xml/BooleanGetMethodName"/>
<rule ref="category/java/performance.xml/BooleanInstantiation"/>
<rule ref="category/java/errorprone.xml/BrokenNullCheck"/>
<rule ref="category/java/performance.xml/ByteInstantiation"/>
<rule ref="category/java/errorprone.xml/CallSuperFirst"/>
<rule ref="category/java/codestyle.xml/CallSuperInConstructor"/>
<rule ref="category/java/errorprone.xml/CallSuperLast"/>
<rule ref="category/java/bestpractices.xml/CheckResultSet"/>
<rule ref="category/java/errorprone.xml/CheckSkipResult"/>
<rule ref="category/java/errorprone.xml/ClassCastExceptionWithToArray"/>
<rule ref="category/apex/codestyle.xml/ClassNamingConventions"/>
<rule ref="category/java/codestyle.xml/ClassNamingConventions"/>
<rule ref="category/java/design.xml/ClassWithOnlyPrivateConstructorsShouldBeFinal"/>
<rule ref="category/java/errorprone.xml/CloneMethodMustBePublic"/>
<rule ref="category/java/errorprone.xml/CloneMethodMustImplementCloneable"/>
<rule ref="category/java/errorprone.xml/CloneMethodReturnTypeMustMatchClassName"/>
<rule ref="category/java/errorprone.xml/CloneThrowsCloneNotSupportedException"/>
<rule ref="category/java/errorprone.xml/CloseResource"/>
<rule ref="category/plsql/codestyle.xml/CodeFormat"/>
<rule ref="category/apex/design.xml/CognitiveComplexity"/>
<rule ref="category/java/design.xml/CognitiveComplexity"/>
<rule ref="category/java/design.xml/CollapsibleIfStatements"/>
<rule ref="category/vm/design.xml/CollapsibleIfStatements"/>
<rule ref="category/java/documentation.xml/CommentContent"/>
<rule ref="category/java/codestyle.xml/CommentDefaultAccessModifier"/>
<rule ref="category/java/documentation.xml/CommentRequired"/>
<rule ref="category/java/documentation.xml/CommentSize"/>
<rule ref="category/java/errorprone.xml/CompareObjectsWithEquals"/>
<rule ref="category/java/errorprone.xml/ComparisonWithNaN"/>
<rule ref="category/java/codestyle.xml/ConfusingTernary"/>
<rule ref="category/java/performance.xml/ConsecutiveAppendsShouldReuse"/>
<rule ref="category/java/performance.xml/ConsecutiveLiteralAppends"/>
<rule ref="category/ecmascript/bestpractices.xml/ConsistentReturn"/>
<rule ref="category/java/bestpractices.xml/ConstantsInInterface"/>
<rule ref="category/java/errorprone.xml/ConstructorCallsOverridableMethod"/>
<rule ref="category/java/codestyle.xml/ControlStatementBraces"/>
<rule ref="category/java/design.xml/CouplingBetweenObjects"/>
<rule ref="category/apex/design.xml/CyclomaticComplexity"/>
<rule ref="category/java/design.xml/CyclomaticComplexity"/>
<rule ref="category/plsql/design.xml/CyclomaticComplexity"/>
<rule ref="category/java/design.xml/DataClass"/>
<rule ref="category/java/errorprone.xml/DataflowAnomalyAnalysis"/>
<rule ref="category/apex/bestpractices.xml/DebugsShouldUseLoggingLevel"/>
<rule ref="category/java/bestpractices.xml/DefaultLabelNotLastInSwitchStmt"/>
<rule ref="category/java/codestyle.xml/DefaultPackage"/>
<rule ref="category/java/errorprone.xml/DetachedTestCase"/>
<rule ref="category/java/errorprone.xml/DoNotCallGarbageCollectionExplicitly"/>
<rule ref="category/java/design.xml/DoNotExtendJavaLangError"/>
<rule ref="category/java/errorprone.xml/DoNotExtendJavaLangThrowable"/>
<rule ref="category/java/errorprone.xml/DoNotHardCodeSDCard"/>
<rule ref="category/java/errorprone.xml/DoNotTerminateVM"/>
<rule ref="category/java/errorprone.xml/DoNotThrowExceptionInFinally"/>
<rule ref="category/java/multithreading.xml/DoNotUseThreads"/>
<rule ref="category/java/multithreading.xml/DontCallThreadRun"/>
<rule ref="category/java/codestyle.xml/DontImportJavaLang"/>
<rule ref="category/java/errorprone.xml/DontImportSun"/>
<rule ref="category/jsp/bestpractices.xml/DontNestJsfInJstlIteration"/>
<rule ref="category/java/errorprone.xml/DontUseFloatTypeForLoopIndices"/>
<rule ref="category/java/bestpractices.xml/DoubleBraceInitialization"/>
<rule ref="category/java/multithreading.xml/DoubleCheckedLocking"/>
<rule ref="category/java/codestyle.xml/DuplicateImports"/>
<rule ref="category/jsp/codestyle.xml/DuplicateJspImports"/>
<rule ref="category/apex/performance.xml/EagerlyLoadedDescribeSObjectResult"/>
<rule ref="category/apex/errorprone.xml/EmptyCatchBlock"/>
<rule ref="category/java/errorprone.xml/EmptyCatchBlock"/>
<rule ref="category/java/errorprone.xml/EmptyFinalizer"/>
<rule ref="category/java/errorprone.xml/EmptyFinallyBlock"/>
<rule ref="category/vm/errorprone.xml/EmptyForeachStmt"/>
<rule ref="category/apex/errorprone.xml/EmptyIfStmt"/>
<rule ref="category/java/errorprone.xml/EmptyIfStmt"/>
<rule ref="category/vm/errorprone.xml/EmptyIfStmt"/>
<rule ref="category/java/errorprone.xml/EmptyInitializer"/>
<rule ref="category/java/codestyle.xml/EmptyMethodInAbstractClassShouldBeAbstract"/>
<rule ref="category/apex/errorprone.xml/EmptyStatementBlock"/>
<rule ref="category/java/errorprone.xml/EmptyStatementBlock"/>
<rule ref="category/java/errorprone.xml/EmptyStatementNotInLoop"/>
<rule ref="category/java/errorprone.xml/EmptySwitchStatements"/>
<rule ref="category/java/errorprone.xml/EmptySynchronizedBlock"/>
<rule ref="category/java/errorprone.xml/EmptyTryBlock"/>
<rule ref="category/apex/errorprone.xml/EmptyTryOrFinallyBlock"/>
<rule ref="category/apex/errorprone.xml/EmptyWhileStmt"/>
<rule ref="category/java/errorprone.xml/EmptyWhileStmt"/>
```

```
<rule ref="category/ecmascript/errorprone.xml/EqualComparison"/>
<rule ref="category/java/errorprone.xml/EqualsNull"/>
<rule ref="category/java/design.xml/ExceptionAsFlowControl"/>
<rule ref="category/apex/design.xml/ExcessiveClassLength"/>
<rule ref="category/java/design.xml/ExcessiveClassLength"/>
<rule ref="category/java/design.xml/ExcessiveImports"/>
<rule ref="category/java/design.xml/ExcessiveMethodLength"/>
<rule ref="category/plsql/design.xml/ExcessiveMethodLength"/>
<rule ref="category/plsql/design.xml/ExcessiveObjectLength"/>
<rule ref="category/plsql/design.xml/ExcessivePackageBodyLength"/>
<rule ref="category/plsql/design.xml/ExcessivePackageSpecificationLength"/>
<rule ref="category/apex/design.xml/ExcessiveParameterList"/>
<rule ref="category/java/design.xml/ExcessiveParameterList"/>
<rule ref="category/plsql/design.xml/ExcessiveParameterList"/>
<rule ref="category/apex/design.xml/ExcessivePublicCount"/>
<rule ref="category/java/design.xml/ExcessivePublicCount"/>
<rule ref="category/vm/design.xml/ExcessiveTemplateLength"/>
<rule ref="category/plsql/design.xml/ExcessiveTypeLength"/>
<rule ref="category/java/codestyle.xml/ExtendsObject"/>
<rule ref="category/apex/codestyle.xml/FieldDeclarationsShouldBeAtStart"/>
<rule ref="category/java/codestyle.xml/FieldDeclarationsShouldBeAtStartOfClass"/>
<rule ref="category/apex/codestyle.xml/FieldNamingConventions"/>
<rule ref="category/java/codestyle.xml/FieldNamingConventions"/>
<rule ref="category/java/design.xml/FinalFieldCouldBeStatic"/>
<rule ref="category/java/codestyle.xml/FinalParameterInAbstractMethod"/>
<rule ref="category/java/errorprone.xml/FinalizeDoesNotCallSuperFinalize"/>
<rule ref="category/java/errorprone.xml/FinalizeOnlyCallsSuperFinalize"/>
<rule ref="category/java/errorprone.xml/FinalizeOverloaded"/>
<rule ref="category/java/errorprone.xml/FinalizeShouldBeProtected"/>
<rule ref="category/java/bestpractices.xml/ForLoopCanBeForeach"/>
<rule ref="category/plsql/codestyle.xml/ForLoopNaming"/>
<rule ref="category/java/codestyle.xml/ForLoopShouldBeWhileLoop"/>
<rule ref="category/java/bestpractices.xml/ForLoopVariableCount"/>
<rule ref="category/apex/codestyle.xml/ForLoopsMustUseBraces"/>
<rule ref="category/ecmascript/codestyle.xml/ForLoopsMustUseBraces"/>
<rule ref="category/java/codestyle.xml/ForLoopsMustUseBraces"/>
<rule ref="category/apex/codestyle.xml/FormalParameterNamingConventions"/>
<rule ref="category/java/codestyle.xml/FormalParameterNamingConventions"/>
<rule ref="category/java/codestyle.xml/GenericsNaming"/>
<rule ref="category/ecmascript/bestpractices.xml/GlobalVariable"/>
<rule ref="category/java/design.xml/GodClass"/>
<rule ref="category/java/bestpractices.xml/GuardLogStatement"/>
<rule ref="category/java/security.xml/HardCodedCryptoKey"/>
<rule ref="category/java/errorprone.xml/IdempotentOperations"/>
<rule ref="category/java/codestyle.xml/IdenticalCatchBranches"/>
<rule ref="category/apex/codestyle.xml/IfElseStmtsMustUseBraces"/>
<rule ref="category/ecmascript/codestyle.xml/IfElseStmtsMustUseBraces"/>
<rule ref="category/java/codestyle.xml/IfElseStmtsMustUseBraces"/>
<rule ref="category/apex/codestyle.xml/IfStmtsMustUseBraces"/>
<rule ref="category/ecmascript/codestyle.xml/IfStmtsMustUseBraces"/>
<rule ref="category/java/codestyle.xml/IfStmtsMustUseBraces"/>
<rule ref="category/jsp/security.xml/IframeMissingSrcAttribute"/>
<rule ref="category/java/design.xml/ImmutableField"/>
<rule ref="category/java/errorprone.xml/ImplicitSwitchFallThrough"/>
<rule ref="category/java/errorprone.xml/ImportFromSamePackage"/>
<rule ref="category/apex/errorprone.xml/InaccessibleAuraEnabledGetter"/>
<rule ref="category/java/performance.xml/InefficientEmptyStringCheck"/>
<rule ref="category/java/performance.xml/InefficientStringBuffering"/>
<rule ref="category/ecmascript/errorprone.xml/InnaccurateNumericLiteral"/>
<rule ref="category/java/security.xml/InsecureCryptoIv"/>
<rule ref="category/java/errorprone.xml/InstantiationToGetClass"/>
<rule ref="category/java/performance.xml/InsufficientStringBufferDeclaration"/>
<rule ref="category/java/performance.xml/IntegerInstantiation"/>
<rule ref="category/pom/errorprone.xml/InvalidDependencyTypes"/>
<rule ref="category/java/errorprone.xml/InvalidLogMessageFormat"/>
<rule ref="category/java/bestpractices.xml/JUnit4SuitesShouldUseSuiteAnnotation"/>
<rule ref="category/java/bestpractices.xml/JUnit4TestShouldUseAfterAnnotation"/>
<rule ref="category/java/bestpractices.xml/JUnit4TestShouldUseBeforeAnnotation"/>
<rule ref="category/java/bestpractices.xml/JUnit4TestShouldUseTestAnnotation"/>
<rule ref="category/java/bestpractices.xml/JUnit5TestShouldBePackagePrivate"/>
<rule ref="category/java/bestpractices.xml/JUnitAssertionsShouldIncludeMessage"/>
<rule ref="category/java/errorprone.xml/JUnitSpelling"/>
<rule ref="category/java/errorprone.xml/JUnitStaticSuite"/>
<rule ref="category/java/bestpractices.xml/JUnitTestContainsTooManyAsserts"/>
<rule ref="category/java/bestpractices.xml/JUnitTestsShouldIncludeAssert"/>
<rule ref="category/java/bestpractices.xml/JUnitUseExpected"/>
<rule ref="category/jsp/errorprone.xml/JspEncoding"/>
<rule ref="category/java/errorprone.xml/JumbledIncrementer"/>
<rule ref="category/java/design.xml/LawOfDemeter"/>
<rule ref="category/plsql/codestyle.xml/LineLength"/>
<rule ref="category/java/codestyle.xml/LinguisticNaming"/>
<rule ref="category/java/bestpractices.xml/LiteralsFirstInComparisons"/>
<rule ref="category/java/codestyle.xml/LocalHomeNamingConvention"/>
<rule ref="category/java/codestyle.xml/LocalInterfaceSessionNamingConvention"/>
```

```
   <rule ref="category/java/codestyle.xml/LocalVariableCouldBeFinal"/>
   <rule ref="category/apex/codestyle.xml/LocalVariableNamingConventions"/>
   <rule ref="category/java/codestyle.xml/LocalVariableNamingConventions"/>
   <rule ref="category/java/errorprone.xml/LoggerIsNotStaticFinal"/>
   <rule ref="category/java/design.xml/LogicInversion"/>
   <rule ref="category/java/performance.xml/LongInstantiation"/>
   <rule ref="category/java/codestyle.xml/LongVariable"/>
   <rule ref="category/java/bestpractices.xml/LooseCoupling"/>
   <rule ref="category/java/design.xml/LoosePackageCoupling"/>
   <rule ref="category/java/codestyle.xml/MDBAndSessionBeanNamingConvention"/>
   <rule ref="category/java/codestyle.xml/MIsLeadingVariableName"/>
   <rule ref="category/java/codestyle.xml/MethodArgumentCouldBeFinal"/>
   <rule ref="category/apex/codestyle.xml/MethodNamingConventions"/>
   <rule ref="category/java/codestyle.xml/MethodNamingConventions"/>
   <rule ref="category/java/bestpractices.xml/MethodReturnsInternalArray"/>
   <rule ref="category/apex/errorprone.xml/MethodWithSameNameAsEnclosingClass"/>
   <rule ref="category/java/errorprone.xml/MethodWithSameNameAsEnclosingClass"/>
   <rule ref="category/java/errorprone.xml/MisplacedNullCheck"/>
   <rule ref="category/plsql/codestyle.xml/MisplacedPragma"/>
   <rule ref="category/java/bestpractices.xml/MissingOverride"/>
   <rule ref="category/java/errorprone.xml/MissingSerialVersionUID"/>
   <rule ref="category/java/errorprone.xml/MissingStaticMethodInNonInstantiatableClass"/>
   <rule ref="category/xml/errorprone.xml/MistypedCDATASection"/>
   <rule ref="category/java/design.xml/ModifiedCyclomaticComplexity"/>
   <rule ref="category/java/errorprone.xml/MoreThanOneLogger"/>
   <rule ref="category/java/design.xml/MutableStaticState"/>
   <rule class="net.sourceforge.pmd.lang.rule.XPathRule" deprecated="false" dfa="false"
         language="java"
         message="Avoid System.out.println"
         name="MyPrintlnRule"
         typeResolution="true">
      <description>class Foo{&#xD;
    Logger log = Logger.getLogger(Foo.class.getName());&#xD;
    public void testA () {&#xD;
        System.out.print("Entering test");&#xD;
        System.out.println("Entering test");&#xD;
        // Better use this&#xD;
        log.fine("Entering test");&#xD;
    }&#xD;
}</description>
      <priority>2</priority>
      <properties>
         <property name="xpath">
            <value>//Name[&#xD;
    starts-with(@Image, 'System.out.println')&#xD;
    or&#xD;
    starts-with(@Image, 'System.err.println')&#xD;
    ]</value>
         </property>
      </properties>
   </rule>
   <rule ref="category/java/design.xml/NPathComplexity"/>
   <rule ref="category/plsql/design.xml/NPathComplexity"/>
   <rule ref="category/apex/design.xml/NcssConstructorCount"/>
   <rule ref="category/java/design.xml/NcssConstructorCount"/>
   <rule ref="category/java/design.xml/NcssCount"/>
   <rule ref="category/apex/design.xml/NcssMethodCount"/>
   <rule ref="category/java/design.xml/NcssMethodCount"/>
   <rule ref="category/plsql/design.xml/NcssMethodCount"/>
   <rule ref="category/plsql/design.xml/NcssObjectCount"/>
   <rule ref="category/apex/design.xml/NcssTypeCount"/>
   <rule ref="category/java/design.xml/NcssTypeCount"/>
   <rule ref="category/jsp/bestpractices.xml/NoClassAttribute"/>
   <rule ref="category/ecmascript/codestyle.xml/NoElseReturn"/>
   <rule ref="category/jsp/bestpractices.xml/NoHtmlComments"/>
   <rule ref="category/vm/design.xml/NoInlineJavaScript"/>
   <rule ref="category/jsp/design.xml/NoInlineScript"/>
   <rule ref="category/jsp/design.xml/NoInlineStyleInformation"/>
   <rule ref="category/vm/design.xml/NoInlineStyles"/>
   <rule ref="category/jsp/bestpractices.xml/NoJspForward"/>
   <rule ref="category/jsp/design.xml/NoLongScripts"/>
   <rule ref="category/java/codestyle.xml/NoPackage"/>
   <rule ref="category/jsp/design.xml/NoScriptlets"/>
   <rule ref="category/jsp/security.xml/NoUnsanitizedJSPExpression"/>
   <rule ref="category/java/errorprone.xml/NonCaseLabelInSwitchStatement"/>
   <rule ref="category/java/errorprone.xml/NonStaticInitializer"/>
   <rule ref="category/java/multithreading.xml/NonThreadSafeSingleton"/>
   <rule ref="category/java/errorprone.xml/NullAssignment"/>
   <rule ref="category/apex/codestyle.xml/OneDeclarationPerLine"/>
   <rule ref="category/java/bestpractices.xml/OneDeclarationPerLine"/>
   <rule ref="category/java/codestyle.xml/OnlyOneReturn"/>
   <rule ref="category/apex/performance.xml/OperationWithLimitsInLoop"/>
   <rule ref="category/java/performance.xml/OptimizableToArrayCall"/>
   <rule ref="category/apex/errorprone.xml/OverrideBothEqualsAndHashcode"/>
```

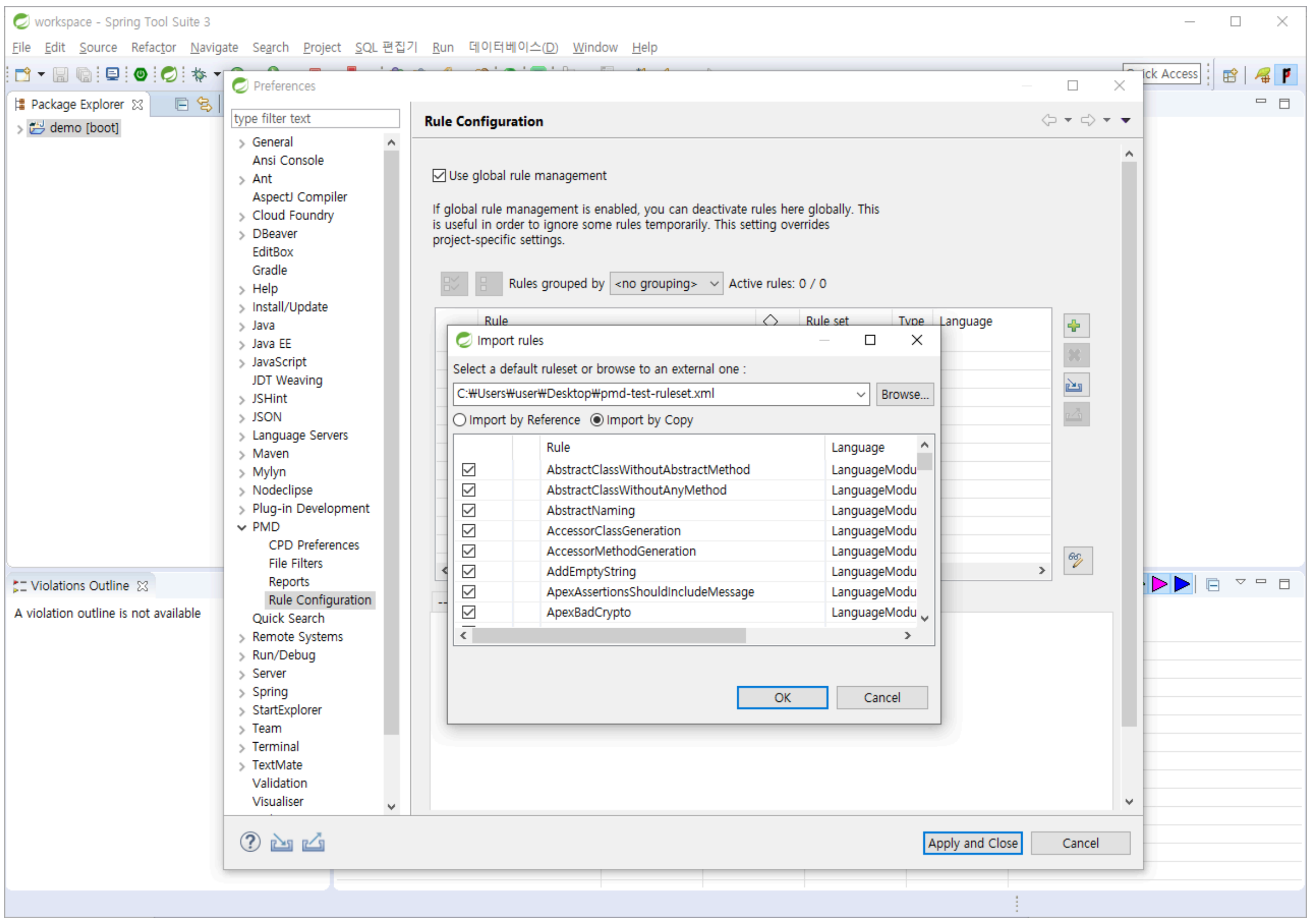
```
<rule ref="category/java/errorprone.xml/OverrideBothEqualsAndHashcode"/>
<rule ref="category/java/codestyle.xml/PackageCase"/>
<rule ref="category/java/bestpractices.xml/PositionLiteralsFirstInCaseInsensitiveComparisons"/>
<rule ref="category/java/bestpractices.xml/PositionLiteralsFirstInComparisons"/>
<rule ref="category/java/codestyle.xml/PrematureDeclaration"/>
<rule ref="category/java/bestpractices.xml/PreserveStackTrace"/>
<rule ref="category/java/bestpractices.xml/PrimitiveWrapperInstantiation"/>
<rule ref="category/pom/errorprone.xml/ProjectVersionAsDependencyVersion"/>
<rule ref="category/java/errorprone.xml/ProperCloneImplementation"/>
<rule ref="category/java/errorprone.xml/ProperLogger"/>
<rule ref="category/apex/codestyle.xml/PropertyNamingConventions"/>
<rule ref="category/java/performance.xml/RedundantFieldInitializer"/>
<rule ref="category/java/codestyle.xml/RemoteInterfaceNamingConvention"/>
<rule ref="category/java/codestyle.xml/RemoteSessionInterfaceNamingConvention"/>
<rule ref="category/java/bestpractices.xml/ReplaceEnumerationWithIterator"/>
<rule ref="category/java/bestpractices.xml/ReplaceHashtableWithMap"/>
<rule ref="category/java/bestpractices.xml/ReplaceVectorWithList"/>
<rule ref="category/java/errorprone.xml/ReturnEmptyArrayRatherThanNull"/>
<rule ref="category/java/errorprone.xml/ReturnEmptyCollectionRatherThanNull"/>
<rule ref="category/java/errorprone.xml/ReturnFromFinallyBlock"/>
<rule ref="category/ecmascript/bestpractices.xml/ScopeForInVariable"/>
<rule ref="category/java/codestyle.xml/ShortClassName"/>
<rule ref="category/java/performance.xml/ShortInstantiation"/>
<rule ref="category/java/codestyle.xml/ShortMethodName"/>
<rule ref="category/java/codestyle.xml/ShortVariable"/>
<rule ref="category/java/design.xml/SignatureDeclareThrowsException"/>
<rule ref="category/java/errorprone.xml/SimpleDateFormatNeedsLocale"/>
<rule ref="category/java/bestpractices.xml/SimplifiableTestAssertion"/>
<rule ref="category/java/design.xml/SimplifiedTernary"/>
<rule ref="category/java/design.xml/SimplifyBooleanAssertion"/>
<rule ref="category/java/design.xml/SimplifyBooleanExpressions"/>
<rule ref="category/java/design.xml/SimplifyBooleanReturns"/>
<rule ref="category/java/design.xml/SimplifyConditional"/>
<rule ref="category/java/performance.xml/SimplifyStartsWith"/>
<rule ref="category/java/errorprone.xml/SingleMethodSingleton"/>
<rule ref="category/java/errorprone.xml/SingletonClassReturningNewInstance"/>
<rule ref="category/java/design.xml/SingularField"/>
<rule ref="category/java/errorprone.xml/StaticEJBFieldShouldBeFinal"/>
<rule ref="category/apex/design.xml/StdCyclomaticComplexity"/>
<rule ref="category/java/design.xml/StdCyclomaticComplexity"/>
<rule ref="category/java/errorprone.xml/StringBufferInstantiationWithChar"/>
<rule ref="category/java/performance.xml/StringInstantiation"/>
<rule ref="category/java/performance.xml/StringToString"/>
<rule ref="category/java/codestyle.xml/SuspiciousConstantFieldName"/>
<rule ref="category/java/errorprone.xml/SuspiciousEqualsMethodName"/>
<rule ref="category/java/errorprone.xml/SuspiciousHashcodeMethodName"/>
<rule ref="category/java/errorprone.xml/SuspiciousOctalEscape"/>
<rule ref="category/java/design.xml/SwitchDensity"/>
<rule ref="category/java/bestpractices.xml/SwitchStmtsShouldHaveDefault"/>
<rule ref="category/java/bestpractices.xml/SystemPrintln"/>
<rule ref="category/plsql/errorprone.xml/TO_DATEWithoutDateFormat"/>
<rule ref="category/plsql/errorprone.xml/TO_DATE_TO_CHAR"/>
<rule ref="category/plsql/errorprone.xml/TO_TIMESTAMPWithoutDateFormat"/>
<rule ref="category/java/errorprone.xml/TestClassWithoutTestCases"/>
<rule ref="category/apex/errorprone.xml/TestMethodsMustBeInTestClasses"/>
<rule ref="category/plsql/bestpractices.xml/TomKytesDespair"/>
<rule ref="category/java/performance.xml/TooFewBranchesForASwitchStatement"/>
<rule ref="category/apex/design.xml/TooManyFields"/>
<rule ref="category/java/design.xml/TooManyFields"/>
<rule ref="category/plsql/design.xml/TooManyFields"/>
<rule ref="category/java/design.xml/TooManyMethods"/>
<rule ref="category/plsql/design.xml/TooManyMethods"/>
<rule ref="category/java/codestyle.xml/TooManyStaticImports"/>
<rule ref="category/java/documentation.xml/UncommentedEmptyConstructor"/>
<rule ref="category/java/documentation.xml/UncommentedEmptyMethodBody"/>
<rule ref="category/java/errorprone.xml/UnconditionalIfStatement"/>
<rule ref="category/java/codestyle.xml/UnnecessaryAnnotationValueElement"/>
<rule ref="category/ecmascript/codestyle.xml/UnnecessaryBlock"/>
<rule ref="category/java/errorprone.xml/UnnecessaryBooleanAssertion"/>
<rule ref="category/java/errorprone.xml/UnnecessaryCaseChange"/>
<rule ref="category/java/codestyle.xml/UnnecessaryCast"/>
<rule ref="category/java/codestyle.xml/UnnecessaryConstructor"/>
<rule ref="category/java/errorprone.xml/UnnecessaryConversionTemporary"/>
<rule ref="category/java/codestyle.xml/UnnecessaryFullyQualifiedName"/>
<rule ref="category/java/codestyle.xml/UnnecessaryImport"/>
<rule ref="category/java/codestyle.xml/UnnecessaryLocalBeforeReturn"/>
<rule ref="category/java/codestyle.xml/UnnecessaryModifier"/>
<rule ref="category/ecmascript/codestyle.xml/UnnecessaryParentheses"/>
<rule ref="category/java/codestyle.xml/UnnecessaryReturn"/>
<rule ref="category/java/performance.xml/UnnecessaryWrapperObjectCreation"/>
<rule ref="category/ecmascript/codestyle.xml/UnreachableCode"/>
<rule ref="category/java/multithreading.xml/UnsynchronizedStaticDateFormatter"/>
<rule ref="category/java/multithreading.xml/UnsynchronizedStaticFormatter"/>
<rule ref="category/java/bestpractices.xml/UnusedAssignment"/>
```

```
   <rule ref="category/java/bestpractices.xml/UnusedFormalParameter"/>
   <rule ref="category/java/bestpractices.xml/UnusedImports"/>
   <rule ref="category/apex/bestpractices.xml/UnusedLocalVariable"/>
   <rule ref="category/java/bestpractices.xml/UnusedLocalVariable"/>
   <rule ref="category/vm/bestpractices.xml/UnusedMacroParameter"/>
   <rule ref="category/java/errorprone.xml/UnusedNullCheckInEquals"/>
   <rule ref="category/java/bestpractices.xml/UnusedPrivateField"/>
   <rule ref="category/java/bestpractices.xml/UnusedPrivateMethod"/>
   <rule ref="category/java/performance.xml/UseArrayListInsteadOfVector"/>
   <rule ref="category/java/performance.xml/UseArraysAsList"/>
   <rule ref="category/java/bestpractices.xml/UseAssertEqualsInsteadOfAssertTrue"/>
   <rule ref="category/java/bestpractices.xml/UseAssertNullInsteadOfAssertTrue"/>
   <rule ref="category/java/bestpractices.xml/UseAssertSameInsteadOfAssertTrue"/>
   <rule ref="category/java/bestpractices.xml/UseAssertTrueInsteadOfAssertEquals"/>
   <rule ref="category/ecmascript/bestpractices.xml/UseBaseWithParseInt"/>
   <rule ref="category/java/bestpractices.xml/UseCollectionIsEmpty"/>
   <rule ref="category/xsl/codestyle.xml/UseConcatOnce"/>
   <rule ref="category/java/multithreading.xml/UseConcurrentHashMap"/>
   <rule ref="category/java/errorprone.xml/UseCorrectExceptionLogging"/>
   <rule ref="category/java/codestyle.xml/UseDiamondOperator"/>
   <rule ref="category/java/errorprone.xml/UseEqualsToCompareStrings"/>
   <rule ref="category/java/performance.xml/UseIOStreamsWithApacheCommonsFileItem"/>
   <rule ref="category/java/performance.xml/UseIndexOfChar"/>
   <rule ref="category/java/errorprone.xml/UseLocaleWithCaseConversions"/>
   <rule ref="category/java/multithreading.xml/UseNotifyAllInsteadOfNotify"/>
   <rule ref="category/java/design.xml/UseObjectForClearerAPI"/>
   <rule ref="category/java/errorprone.xml/UseProperClassLoader"/>
   <rule ref="category/java/codestyle.xml/UseShortArrayInitializer"/>
   <rule ref="category/java/bestpractices.xml/UseStandardCharsets"/>
   <rule ref="category/java/performance.xml/UseStringBufferForStringAppends"/>
   <rule ref="category/java/performance.xml/UseStringBufferLength"/>
   <rule ref="category/java/bestpractices.xml/UseTryWithResources"/>
   <rule ref="category/java/codestyle.xml/UseUnderscoresInNumericLiterals"/>
   <rule ref="category/java/design.xml/UseUtilityClass"/>
   <rule ref="category/java/bestpractices.xml/UseVarargs"/>
   <rule ref="category/java/errorprone.xml/UselessOperationOnImmutable"/>
   <rule ref="category/java/design.xml/UselessOverridingMethod"/>
   <rule ref="category/java/codestyle.xml/UselessParentheses"/>
   <rule ref="category/java/codestyle.xml/UselessQualifiedThis"/>
   <rule ref="category/java/performance.xml/UselessStringValueOf"/>
   <rule ref="category/apex/codestyle.xml/VariableNamingConventions"/>
   <rule ref="category/java/codestyle.xml/VariableNamingConventions"/>
   <rule ref="category/vf/security.xml/VfCsrf"/>
   <rule ref="category/vf/security.xml/VfHtmlStyleTagXss"/>
   <rule ref="category/vf/security.xml/VfUnescapeEl"/>
   <rule ref="category/java/bestpractices.xml/WhileLoopWithLiteralBoolean"/>
   <rule ref="category/apex/codestyle.xml/WhileLoopsMustUseBraces"/>
   <rule ref="category/ecmascript/codestyle.xml/WhileLoopsMustUseBraces"/>
   <rule ref="category/java/codestyle.xml/WhileLoopsMustUseBraces"/>
</ruleset>
```

중간 부분에 추가한 MyPrintlnRule이 아래와 같이 등록되어 있는 것을 볼 수 있다.

```
<rule class="net.sourceforge.pmd.lang.rule.XPathRule" deprecated="false" dfa="false"
         language="java"
         message="Avoid System.out.println"
         name="MyPrintlnRule"
         typeResolution="true">
      <description>class Foo{&#xD;
    Logger log = Logger.getLogger(Foo.class.getName());&#xD;
    public void testA () {&#xD;
        System.out.print("Entering test");&#xD;
        System.out.println("Entering test");&#xD;
        // Better use this&#xD;
        log.fine("Entering test");&#xD;
    }&#xD;
}</description>
      <priority>2</priority>
      <properties>
         <property name="xpath">
            <value>//Name[&#xD;
    starts-with(@Image, 'System.out.println')&#xD;
    or&#xD;
    starts-with(@Image, 'System.err.println')&#xD;
    ]</value>
         </property>
      </properties>
   </rule>
```

Import는 반대로 추출한 Rule을 반영하는 과정이다. 대체로 현재 반영되어 있는 Rule과 중복을 방지하기 위해 전체 Rule을 삭제하고, 반영하는 것을 추천한다.



**Rule Configuration**

☑ Use global rule management

If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

Rules grouped by <no grouping> Active rules: 0 / 0

Import rules

Select a default ruleset or browse to an external one :

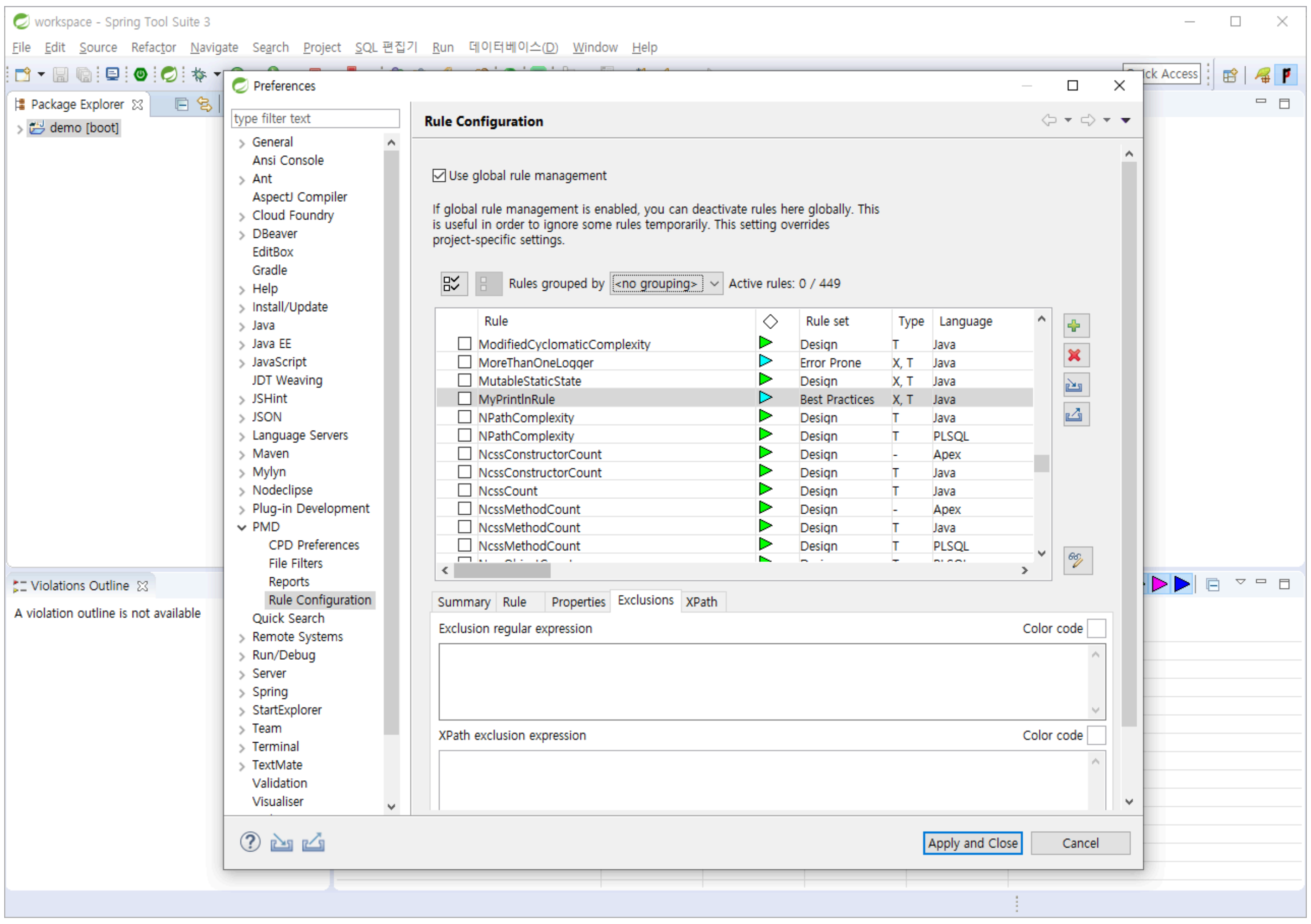
C:₩Users₩user₩Desktop₩pmd-test-ruleset.xml | Browse...

○ Import by Reference ● Import by Copy

| | Rule | Language |
|---|---|---|
| ☑ | AbstractClassWithoutAbstractMethod | LanguageModu |
| ☑ | AbstractClassWithoutAnyMethod | LanguageModu |
| ☑ | AbstractNaming | LanguageModu |
| ☑ | AccessorClassGeneration | LanguageModu |
| ☑ | AccessorMethodGeneration | LanguageModu |
| ☑ | AddEmptyString | LanguageModu |
| ☑ | ApexAssertionsShouldIncludeMessage | LanguageModu |
| ☑ | ApexBadCrypto | LanguageModu |

OK Cancel

위와 같이 Export selected rules를 통해 추출한 ruleset 파일을 반영하고, import by copy를 선택한 후 OK를 클릭한다.



**Rule Configuration**

☑ Use global rule management

If global rule management is enabled, you can deactivate rules here globally. This is useful in order to ignore some rules temporarily. This setting overrides project-specific settings.

Rules grouped by <no grouping> Active rules: 0 / 449

| | Rule | ◇ | Rule set | Type | Language |
|---|---|---|---|---|---|
| ☐ | ModifiedCyclomaticComplexity | ▶ | Design | T | Java |
| ☐ | MoreThanOneLogger | ▶ | Error Prone | X, T | Java |
| ☐ | MutableStaticState | ▶ | Design | X, T | Java |
| ☐ | MyPrintlnRule | ▶ | Best Practices | X, T | Java |
| ☐ | NPathComplexity | ▶ | Design | T | Java |
| ☐ | NPathComplexity | ▶ | Design | T | PLSQL |
| ☐ | NcssConstructorCount | ▶ | Design | - | Apex |
| ☐ | NcssConstructorCount | ▶ | Design | T | Java |
| ☐ | NcssCount | ▶ | Design | T | Java |
| ☐ | NcssMethodCount | ▶ | Design | - | Apex |
| ☐ | NcssMethodCount | ▶ | Design | T | Java |
| ☐ | NcssMethodCount | ▶ | Design | T | PLSQL |

Exclusion regular expression Color code □

XPath exclusion expression Color code □

위와 같이 MyPrintlnRule을 포함한 Ruleset이 적용된 것을 확인할 수 있다.