

# 1. 도구 개요

## PMD

소 개	정해진 규칙에 따라 소스코드를 검사해 주고 이에 대한 결과를 report하게 함으로서 코딩 효율을 높여주는 도구		
주요기능	Code Check, Reporting		
카테고리	Quality Management	세부카테고리	정적분석
커버리지	Java, JavaScript, JSP, XML, XSL	도구난이도	하
라이선스형태 / 비용	BSD-style / 무료	사전설치도구	JDK, Eclipse
운영체제	Windows, Linux, Mac OS X, UNIX	도구버전	5.0.0 (2010. 10)
특징	<ul style="list-style-type: none"> <li>• 작성한 코드에 대한 위반사항(위반되는 코딩 스타일, 불필요한 코드)를 찾음</li> <li>• 위반 사항을 명시한 report파일(pmd.xml, cpd.xml file)에 대한 수정이 쉬움</li> <li>• 한 번의 클릭으로 수많은 규칙에 대한 수정이 가능</li> </ul>		
적용회사 / 프로젝트			
관련 도구	Eclipse Metrics, Checkclipse, Checkstyle, CAP, FindBugs, Jamit, Simian		
제작사	maven		
공식 홈페이지	<a href="http://pmd.sourceforge.net/">http://pmd.sourceforge.net/</a>		

## 2. 기능 요약

### PMD

정해진 규칙에 따라 소스코드를 검사해 주고 이에 대한 결과를 report하게 함으로서 코딩 효율을 높여주는 도구

주요기능	지원내용
소스코드 검사범위	프로젝트
대상 언어	Java, JavaScript, XML, XSL, JSP
코드 위배사항 발견	지원(코딩스타일 및 사용되지 않는 코드)
결과 파일 보고	지원
중복코드 검사	지원
우선순위 시각화	지원(5단계)
규칙 설정	지원 (사용자 정의 설정 / 기본 설정)

### 3. 도구 실행 환경

#### PMD

#### JAVA를 지원하는 IDE상에서 설치 및 구현이 가능

- 다양한 OS를 지원
  - Windows : Windows XP / Windows 7 (32, 64-bit 모두 지원)
  - Linux : 32, 64-bit 지원
  - Mac OS X : 32, 64-bit 지원
  - UNIX : 32, 64-bit 지원
- JDK, 자바 기반 도구(IDE 등)이 필요
  - 코드 및 플러그인 형태(여러 JAVA IDE에 최적화된 형태), 도구에 포함되어 있는 형태로 제공

**PMD (Eclipse Plug-in 형태)**

**Java IDE (Eclipse, NetBeans 등)**

**JDK (Java development kit)**

**Windows / Linux / Mac OS / UNIX**

## 4. 도구 설치 방법

세부 목차

PMD

4.1 PMD 다운 및 설치하기

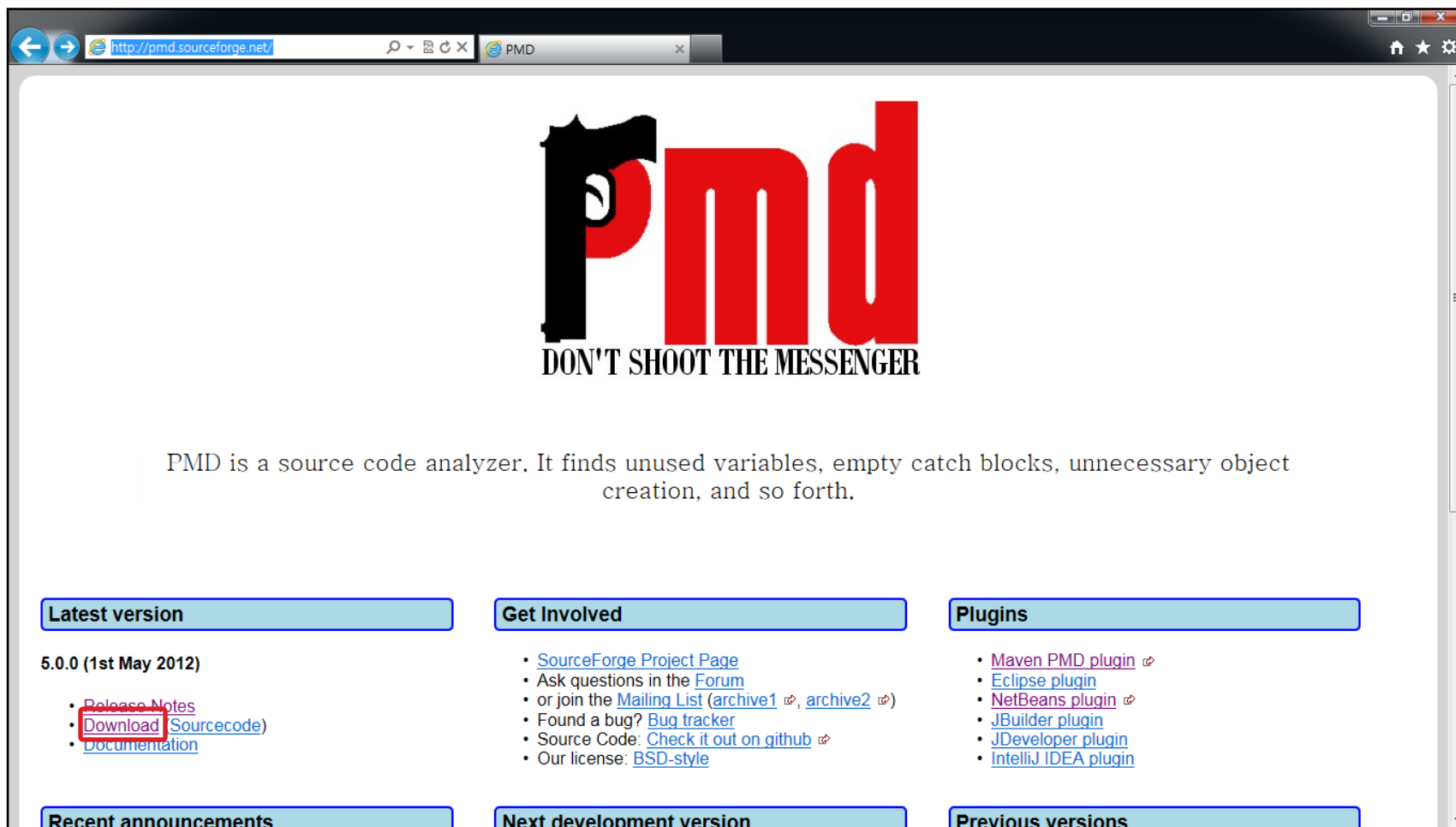
4.2 PMD 설치 확인하기

## 4. 도구 설치 방법

### 4.1 PMD 다운 및 설치하기 (1/2)

PMD

- <http://pmd.sourceforge.net/>에서 PMD 다운
  - 홈페이지에서 Download 클릭

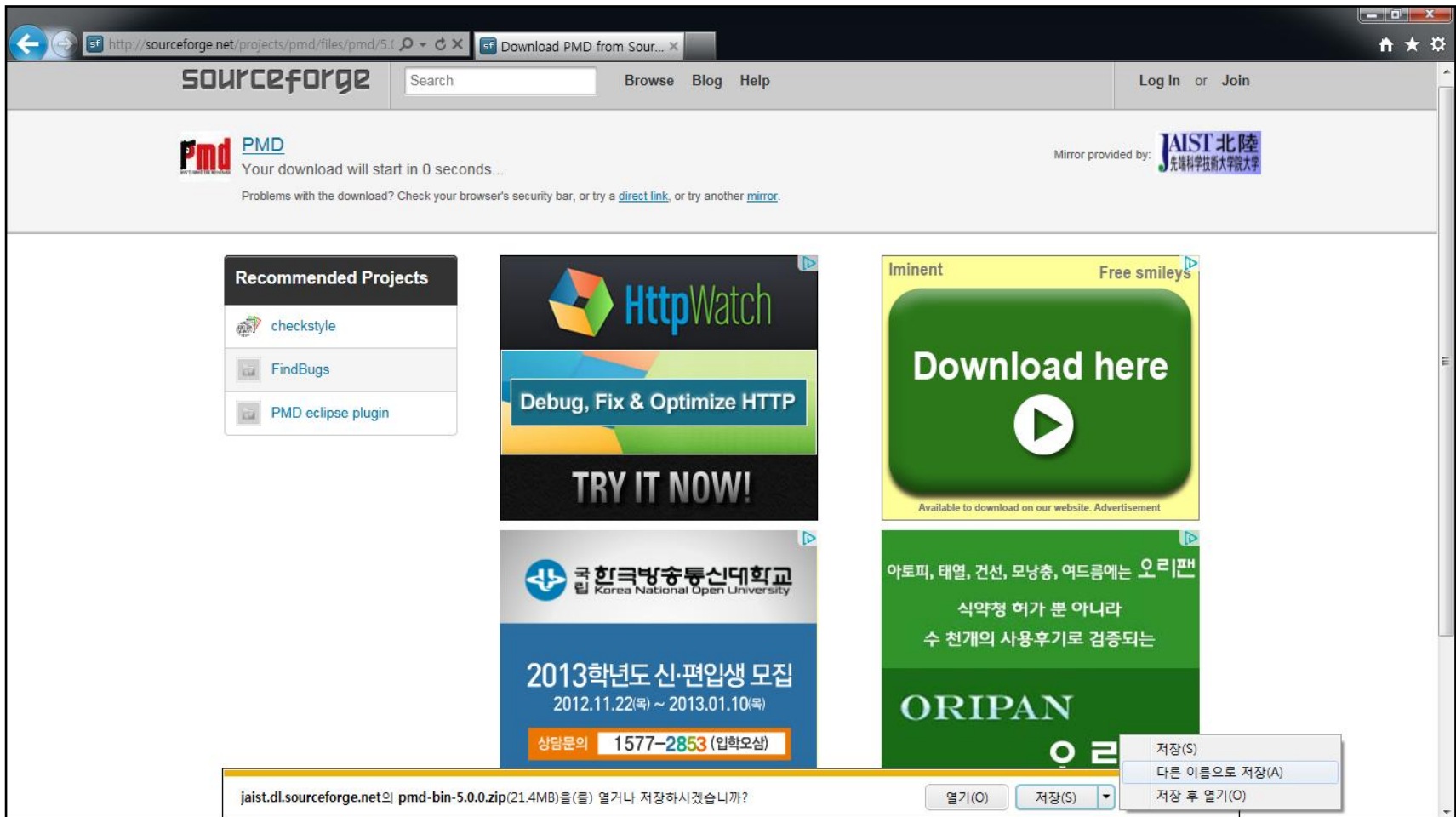


## 4. 도구 설치 방법

### 4.1 PMD 다운 및 설치하기 (2/2)

PMD

- 압축을 풀고, PMD5.0.0을 Eclipse가 설치된 폴더 내의 plugins 폴더에 복사
  - 압축해제 → plugins 폴더에 복사 → Eclipse (재)실행

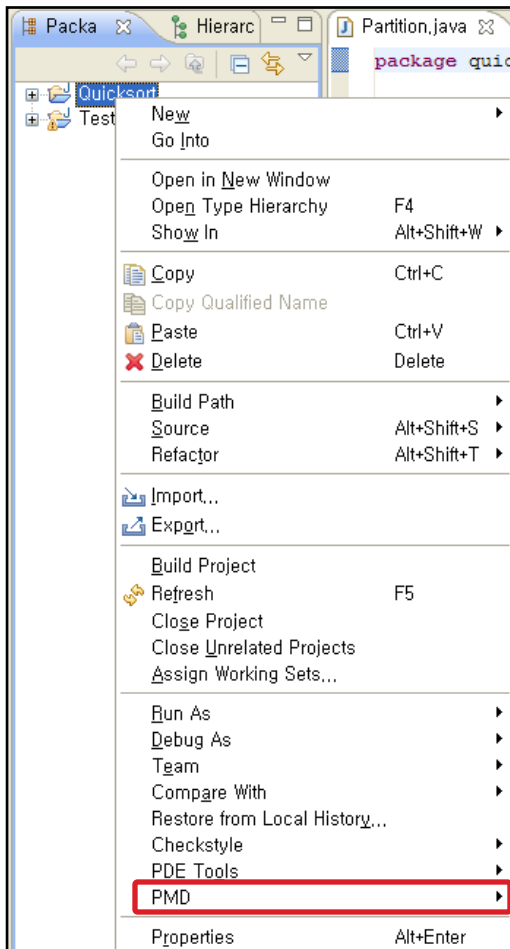


## 4. 도구 설치 방법

### 4.2 PMD 설치 확인하기

PMD

- 설치 확인 하기
  - 프로젝트 폴더 선택 후 마우스 우 클릭 팝업메뉴에 PMD를 확인



## 5. 도구 기능 소개

### 세부목차

### PMD

- 5.1 PMD의 주요 기능
- 5.2 예제소개
- 5.3 PMD를 이용한 코드 검사하기
- 5.4 CPD를 이용한 중복코드 검사하기
- 5.5 검사결과 Reporting하기



## 5. 도구 기능 소개

### 5.1 PMD의 주요 기능

PMD

- PMD의 주요기능 : 소스코드 검사 / CPD적용 / 보고서 작성
  - 프로젝트의 코드를 Rule에 따라 적절하게 구현되어있는지 확인 : 규칙 기반 source code검사
  - 중복 코드의 검색 - CPD(Copy/Paste Detector) : Project → properties
  - PMD와 CPD를 이용하여 check한 결과를 report로 작성

The screenshot displays the PMD and CPD tool outputs. The top left window, 'Violations Overview', shows a list of rules and their violation counts. The top right window, 'pmd-report.txt', shows detailed error messages. The bottom left window, 'cpd-report.txt', shows code snippets. The bottom right window, 'Violations Overview', shows a table of violations.

**[PMD를 이용한 code check]**

**[check결과 reporting]**

Message	Class
Found suspect cut & paste (2 matches, 3 lines)	
lines 29-31 in file Partition.java	src.quicksortmodel.Partition
lines 32-34 in file Partition.java	src.quicksortmodel.Partition

**[CPD를 이용한 중복체크]**

## 5. 도구 기능 소개

### 5.2 예제 소개

#### PMD

- PMD의 기능을 설명하기 위한 예제 : 퀵 정렬(Quick Sort)
  - 퀵 정렬 예제는 분할 알고리즘을 이용하여 구현
  - 정렬 알고리즘 중 가장 우수한 평균 수행속도를 자랑
    - > 분할알고리즘(Partition Algorithm)은 퀵 정렬의 기본 개념
      - » 분할 알고리즘을 기본으로 사용하여 정렬하고자 하는 배열을 2개의 하위 배열로 분할하고 각각의 하위 배열에서는 다시 재귀적으로 자신의 배열을 분할하여 결국 배열을 정렬하는 알고리즘
      - » 멀리 떨어진 자료를 직접적으로 치환함으로써 정렬의 수행속도를 개선
  - 퀵 정렬의 요구사항

배열을 2개의 하위 배열 즉, 왼쪽 하위 배열은 피벗 값보다 작거나 같은 값으로 이루어진 배열, 그리고 오른쪽 하위 배열은 피벗 값보다 큰 값으로 이루어진 배열

왼쪽 하위 배열에 퀵 정렬을 다시 실행

오른쪽 하위 배열에 퀵 정렬을 다시 실행

이 과정을 계속해서 진행하되 서로의 자리 값이 엇갈리면 피벗 값과 왼쪽에서 찾은 값을 바꿈

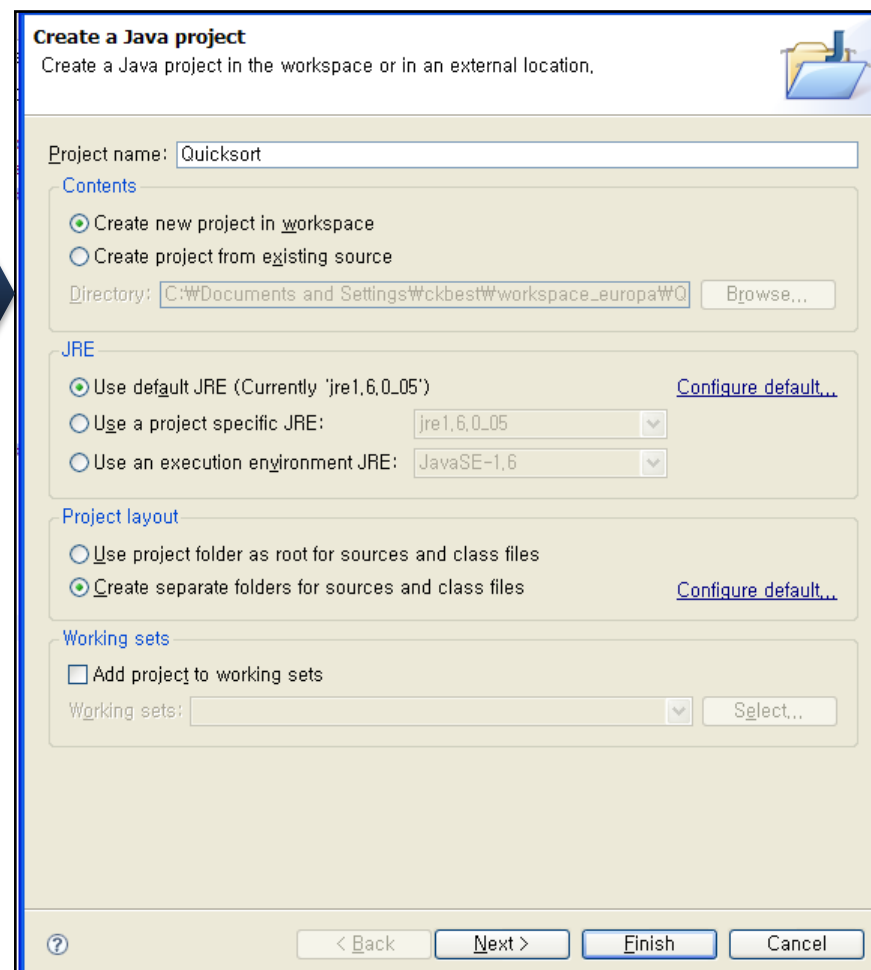
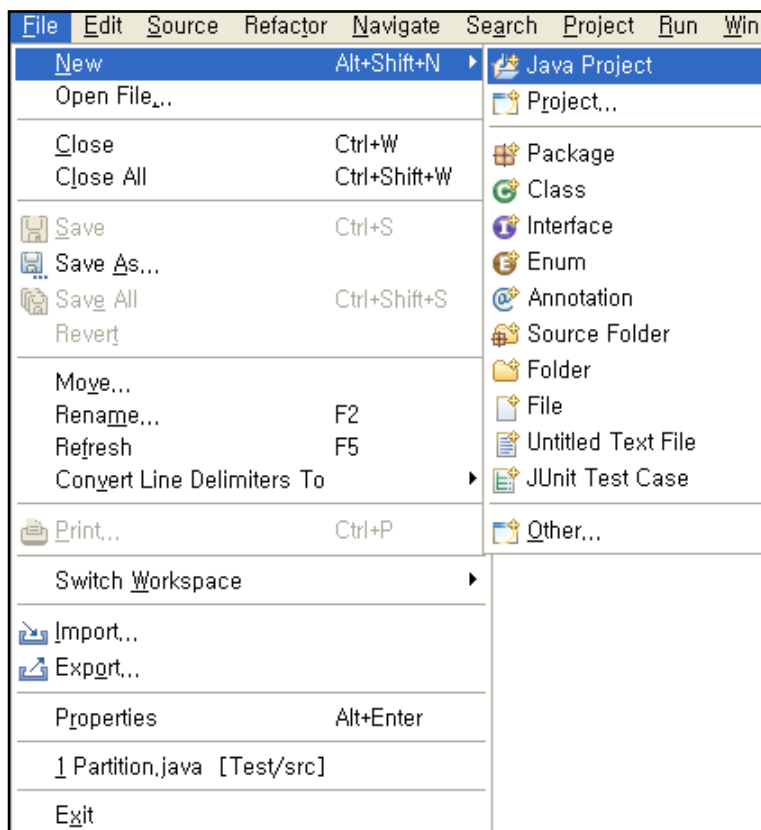
## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (1/9)

PMD

- 프로젝트 생성 : QuickSort

- File → New → Project → Java Project → QuickSort 입력 → Next

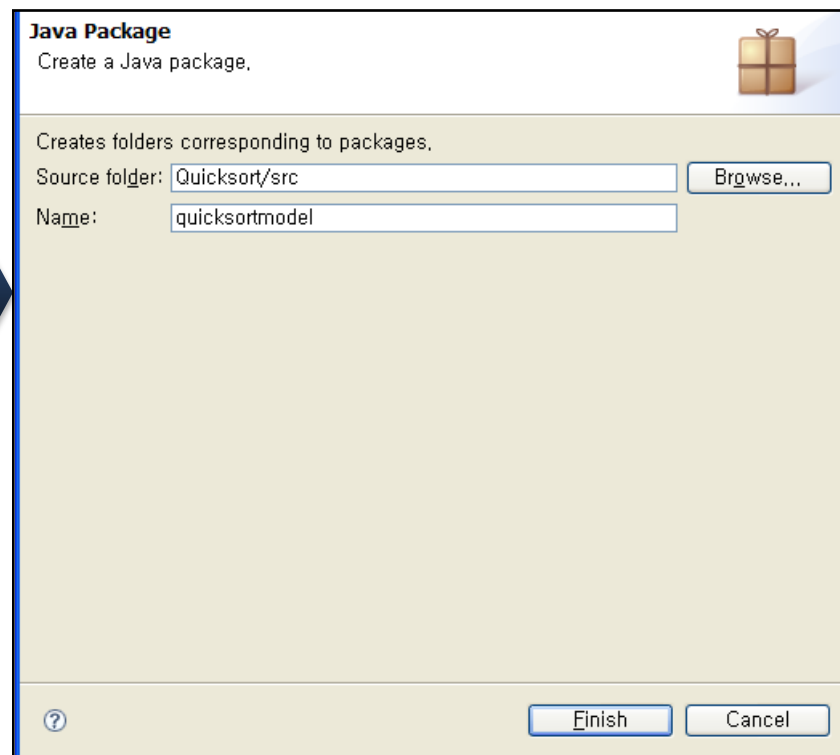
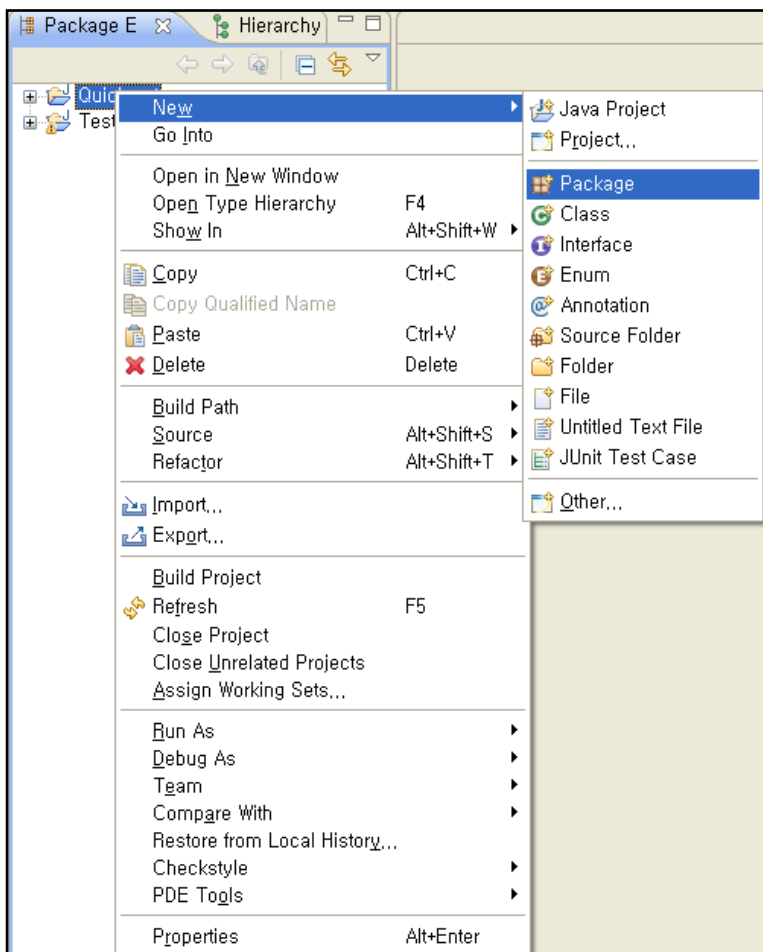


## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (2/9)

PMD

- Quicksortmodel 패키지를 생성
  - 프로젝트 선택 후 마우스 우 클릭 → New → Package → "quicksortmodel" 입력 → Finish

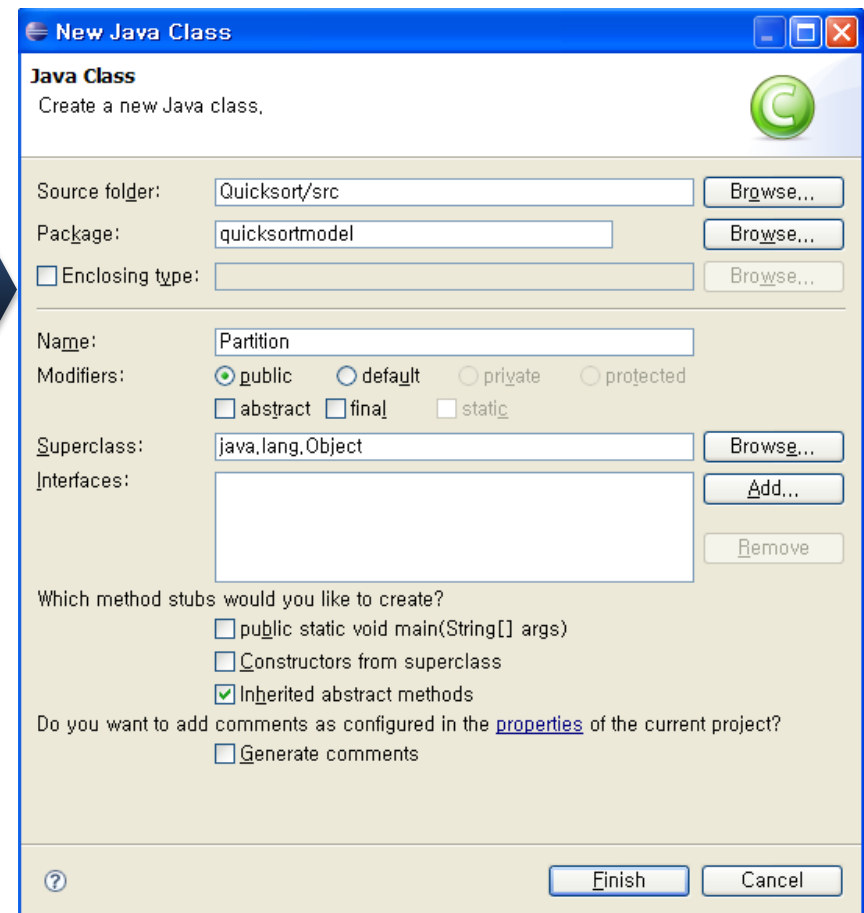
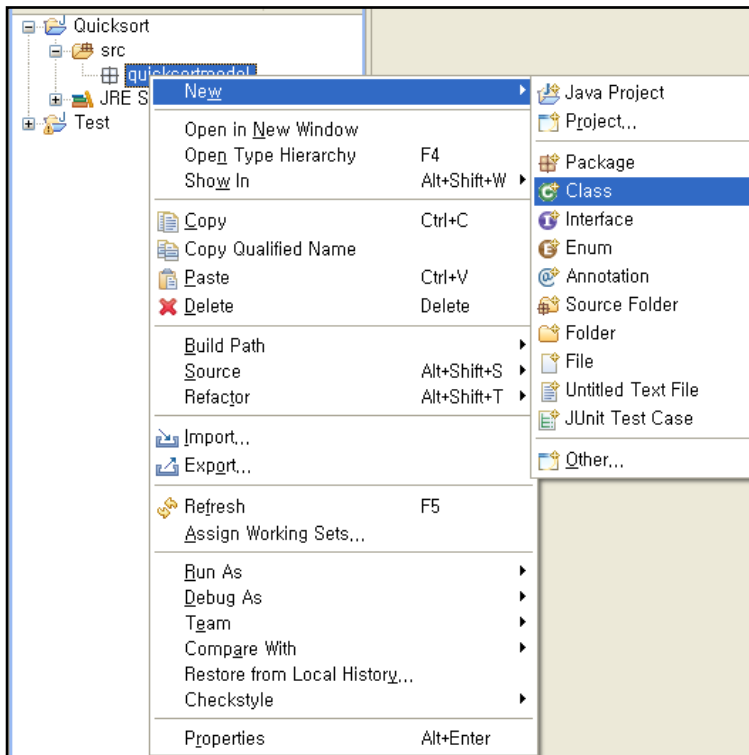


## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (3/9)

PMD

- Quicksortmodel 패키지에 클래스 생성
  - Quicksortmodel 패키지 마우스 우 클릭 → New → Class → "Partition" 입력



## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (4/9)

PMD

- Partition 클래스에 테스트할 quicksort 코드 작성

```
package quicksortmodel;

import java.util.*;

public class Partition {
    static int cnt=1, ind=1;
    public int getRandomNum(int num){
        int result = 0;
        Random rangen = new Random();
        result = rangen.nextInt() % num;

        if (result < 0) result = result * -1;
        return result;
    }
    public int randomDivide(int[] arr, int leftmost, int rightmost){
        Partition middle = new Partition();
        int key;
        if((rightmost-leftmost)<=0)
            return 0;
        else
        {
            key = middle.getRandomNum(rightmost-leftmost+1)+leftmost;
            int pivot = swap(arr, key, rightmost);
            if(ind%10==0){
                ind++;
            }else{
                ind++;
            }
            int i= leftmost-1;
            int j= rightmost;
            while(true){
                while(arr[++i]<pivot);
                while(j>leftmost && arr[--j]>=pivot);
                if(i>=j)
                    break;
                else
                    swap(arr, i, j);
            }
            swap(arr, i, rightmost);
            randomDivide(arr, leftmost, i-1);
            randomDivide(arr, i+1, rightmost);
        }
        return key;
    }
}
```

```
public int swap(int[] arr, int left, int right){
    int temp;
    temp = arr[left];
    arr[left] = arr[right];
    arr[right] = temp;
    return temp;
}

public static void main(String args[]){

    int i, size=1000, left, right, pivot;
    int[] theArray = new int[size];
    Partition middle = new Partition();

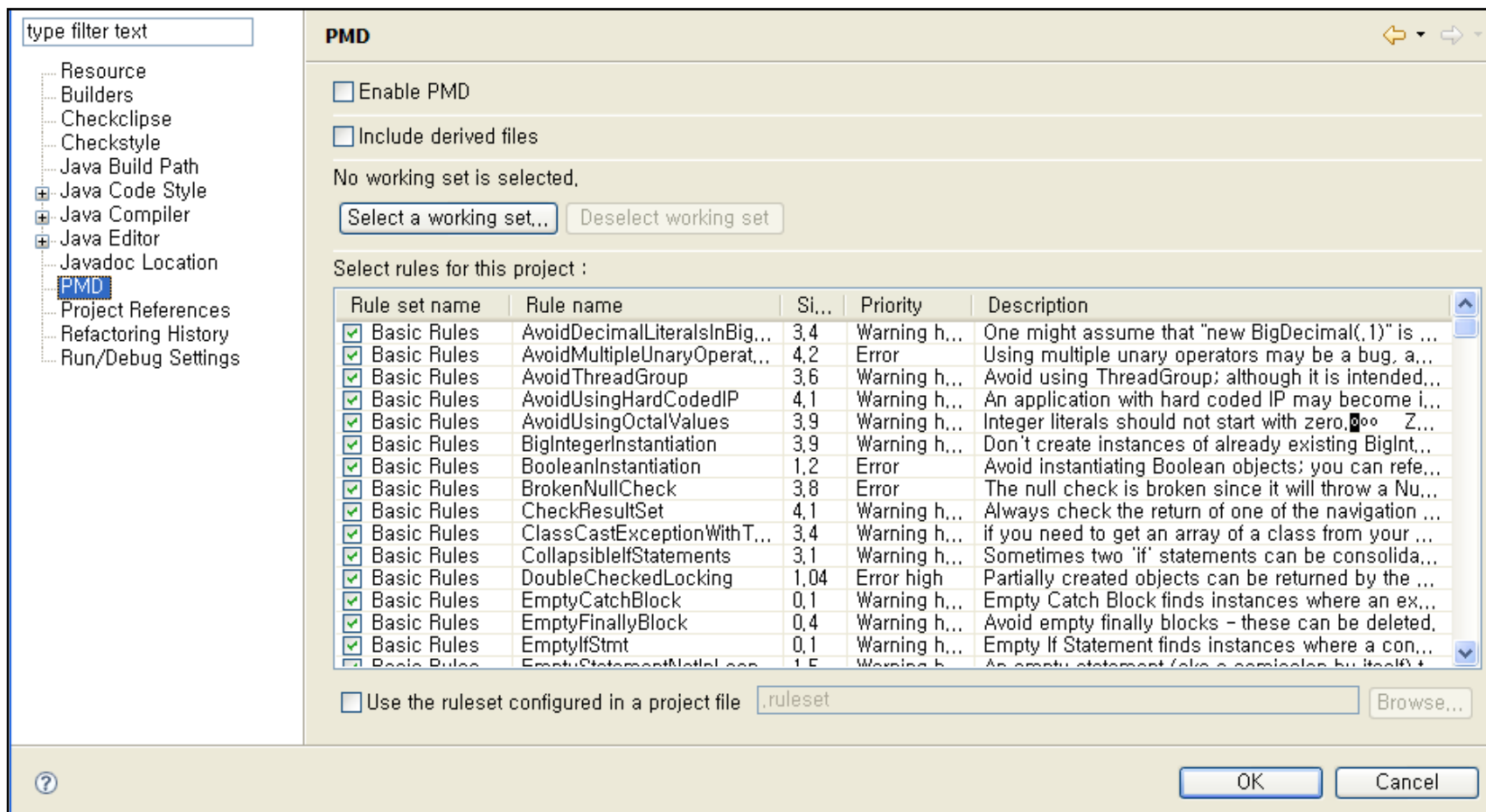
    for(i=0; i<size; i++){
        theArray[i] = middle.getRandomNum(size+1);
    }
    left=0;
    right=size-1;
    pivot = middle.randomDivide(theArray, left, right);
    System.out.print("\n\n<정렬된 후의 값>\n");
    System.out.print("-----\n\t");
    int a=0;
    for(i=0; i<size; i++){
        System.out.print("[ "+(a++)+" ]\n\t");
        int ct = i;
        for(int j=ct; j<ct+20; j++){
            System.out.print(theArray[j]+" \t");
            i++;
        }
        i -= 1;
        System.out.println();
    }
}
```

## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (5/9)

#### PMD

- PMD 기능 및 Rule에 대한 환경 설정
  - QuickSort 프로젝트 선택 → Project → Properties → PMD

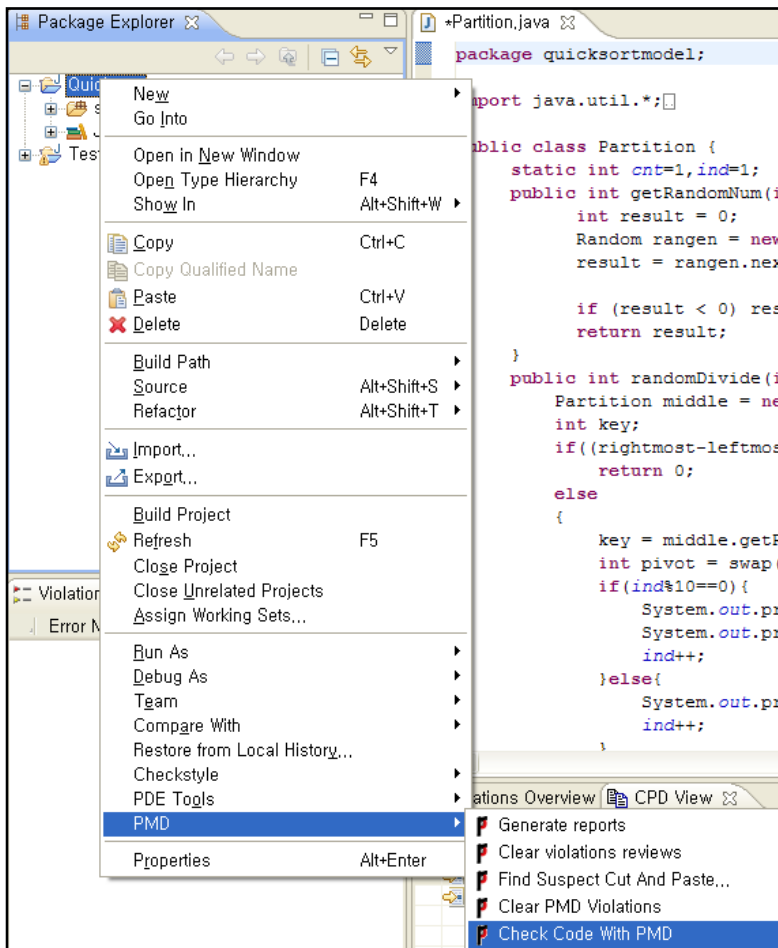


## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (6/9)

#### PMD

- 테스트하고자 하는 프로젝트에 대해 PMD를 실행
  - 프로젝트 마우스 우 클릭 → PMD → Check Code With PMD





## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (7/9)

PMD

#### • 코드 검사의 결과

- Violations Outline : 검사 결과에 대한 에러메시지와 에러가 발생한 해당 라인을 보임
- Violations Overview : 검사 결과에 대한 overview를 보임

Error Message	Line
System.out.print is used	28
System.out.print is used	29
System.out.print is used	32
System.out.print is used	69
System.out.print is used	70
Use explicit scoping instead of t...	7
Parameter 'num' is not assigne...	8
Local variable 'rangen' could b...	10
Avoid using if statements witho...	13
Parameter 'rightmost' is not as...	18
Parameter 'leftmost' is not assi...	18
Parameter 'arr' is not assigned ...	18
Local variable 'middle' could b...	19
Avoid using if...else statements...	22
A method should have only one...	22
Local variable 'pivot' could be d...	26
Avoid variables with short name...	35
Avoid variables with short name...	36
Avoid empty while statements	38
Avoid using while statements w...	38
Avoid using while statements w...	39
Avoid empty while statements	39
Avoid using if...else statements...	41

Element	# Violatio...	# Violations/LOC	# Violations/Meth...	Project
quicksortmodel	88	505,7 / 1000	11,00	Quicksort
Partition.java	44	505,7 / 1000	11,00	Quicksort
SystemPrintln	(max) 5	57,5 / 1000	1,25	Quicksort
DefaultPackage	1	11,5 / 1000	0,25	Quicksort
ShortVariable	(max) 5	57,5 / 1000	1,25	Quicksort
MethodArgumentCouldBeFinal	(max) 5	57,5 / 1000	1,25	Quicksort
DontImportJavaLang	1	11,5 / 1000	0,25	Quicksort
LocalVariableCouldBeFinal	(max) 5	57,5 / 1000	1,25	Quicksort
DataflowAnomalyAnalysis	11	126,4 / 1000	2,75	Quicksort
WhileLoopsMustUseBraces	2	23,0 / 1000	0,50	Quicksort
IfElseStmtsMustUseBraces	3	34,5 / 1000	0,75	Quicksort
EmptyWhileStmt	2	23,0 / 1000	0,50	Quicksort
IfStmtsMustUseBraces	1	11,5 / 1000	0,25	Quicksort
UnusedImports	1	11,5 / 1000	0,25	Quicksort
OnlyOneReturn	1	11,5 / 1000	0,25	Quicksort
ForLoopsMustUseBraces	1	11,5 / 1000	0,25	Quicksort
Partition.java	44	505,7 / 1000	11,00	Quicksort

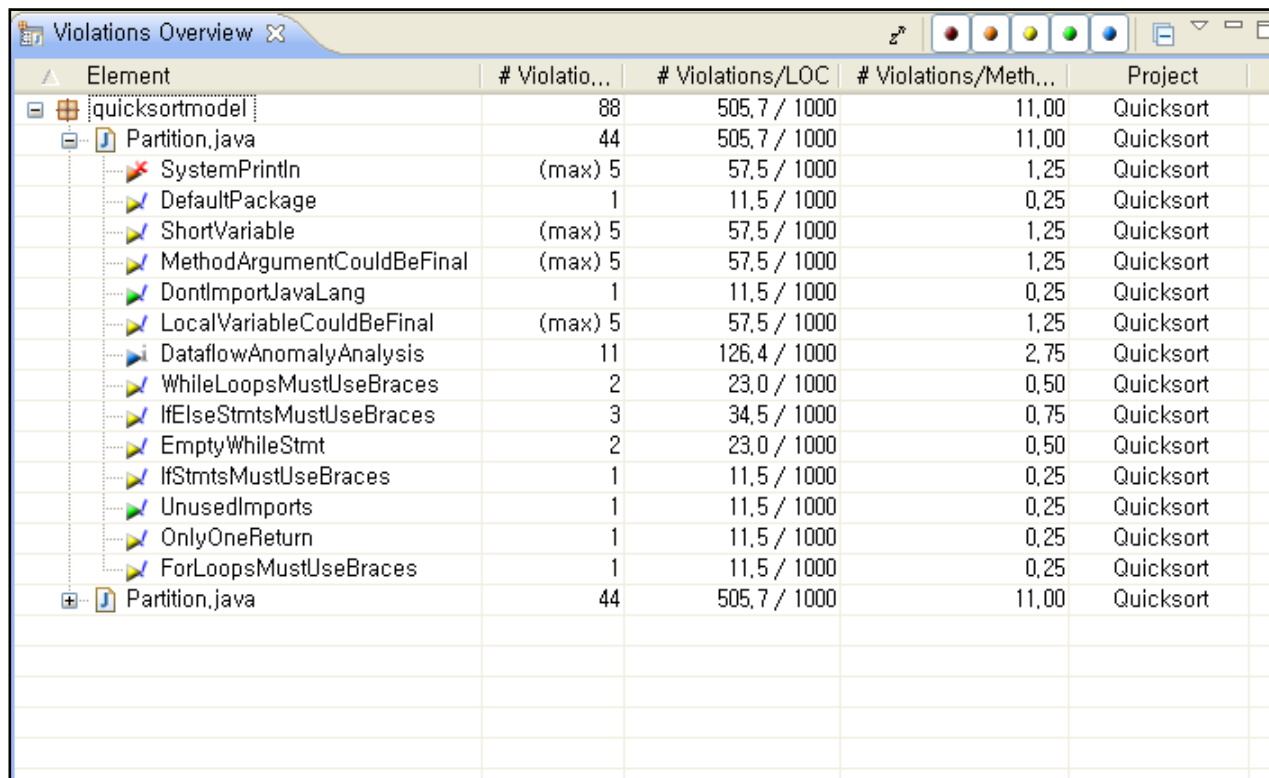
## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (8/9)

#### PMD

#### • Violations Overview

- Element : 우선 순위에 따른 위반사항
- #Violation : 위배된 code 수
- #Violation/LOC : 위치에 따른 위배된 code 수
- #Violation/Method : Method에 따른 위배된 code 수
- Project : 프로젝트 이름



Element	# Violatio...	# Violations/LOC	# Violations/Meth...	Project
quicksortmodel	88	505,7 / 1000	11,00	Quicksort
Partition.java	44	505,7 / 1000	11,00	Quicksort
SystemPrintln	(max) 5	57,5 / 1000	1,25	Quicksort
DefaultPackage	1	11,5 / 1000	0,25	Quicksort
ShortVariable	(max) 5	57,5 / 1000	1,25	Quicksort
MethodArgumentCouldBeFinal	(max) 5	57,5 / 1000	1,25	Quicksort
DontImportJavaLang	1	11,5 / 1000	0,25	Quicksort
LocalVariableCouldBeFinal	(max) 5	57,5 / 1000	1,25	Quicksort
DataflowAnomalyAnalysis	11	126,4 / 1000	2,75	Quicksort
WhileLoopsMustUseBraces	2	23,0 / 1000	0,50	Quicksort
IfElseStmtsMustUseBraces	3	34,5 / 1000	0,75	Quicksort
EmptyWhileStmt	2	23,0 / 1000	0,50	Quicksort
IfStmtsMustUseBraces	1	11,5 / 1000	0,25	Quicksort
UnusedImports	1	11,5 / 1000	0,25	Quicksort
OnlyOneReturn	1	11,5 / 1000	0,25	Quicksort
ForLoopsMustUseBraces	1	11,5 / 1000	0,25	Quicksort
Partition.java	44	505,7 / 1000	11,00	Quicksort

## 5. 도구 기능 소개

### 5.3 PMD를 이용한 코드 검사하기 (9/9)

PMD

- Violations Outline
  - Error Message를 클릭하면 해당코드를 찾아갈 수 있음

The screenshot displays an IDE interface with the following components:

- Package Explorer:** Shows a project named 'Quicksort' with a file 'Partition.java'.
- Partition.java:** Contains the following code:
 

```
public class Partition {
    static int cnt=1, ind=1;
    public int getRandomNum(int num){
        int result = 0;
        Random rangen = new Random();
        result = rangen.nextInt() % num;

        if (result < 0) result = result * -1;
        return result;
    }
    public int randomDivide(int[] arr, int leftmost, int rightmost){
        Partition middle = new Partition();
        int key;
        if((rightmost-leftmost)<=0)
            return 0;
        else
            ...
    }
}
```
- Violations Overview:** A table listing violations across the project.
 

Element	# Violations	# Violations/LOC
quicksortmodel	44	500.0 / 1000
Partition.java	44	500.0 / 1000
SystemPrintln	(max) 5	56.8 / 1000
DefaultPackage	1	11.4 / 1000
MethodArgumentCouldBeFinal	(max) 5	56.8 / 1000
ShortVariable	(max) 5	56.8 / 1000
DontImportJavaLang	1	11.4 / 1000
DataflowAnomalyAnalysis	11	125.0 / 1000
LocalVariableCouldBeFinal	(max) 5	56.8 / 1000
WhileLoopsMustUseBraces	2	22.7 / 1000
IFStmtsMustUseBraces	1	11.4 / 1000
EmptyWhileStmt	2	22.7 / 1000
IFElseStmtsMustUseBraces	3	34.1 / 1000
UnusedImports	1	11.4 / 1000
OnlyOneReturn	1	11.4 / 1000
ForLoopsMustUseBraces	1	11.4 / 1000
- Violations Outline:** A list of error messages with line numbers.
 

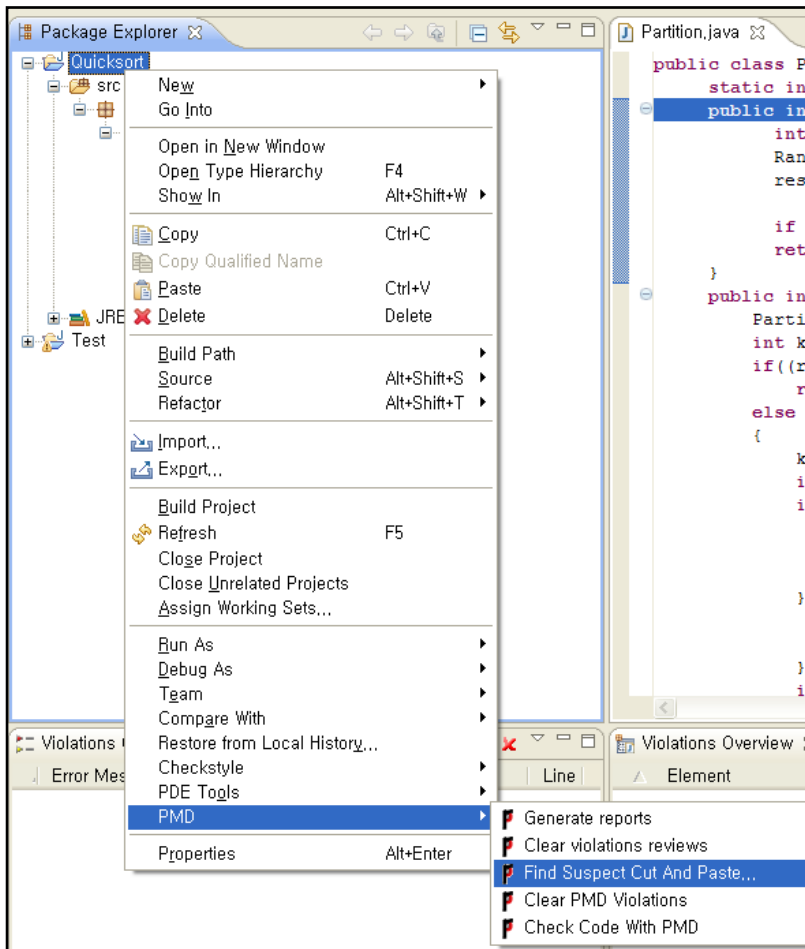
Error Message	Line
System.out.print is used	28
System.out.print is used	29
System.out.print is used	32
System.out.print is used	69
System.out.print is used	70
Use explicit scoping instead of the default package private ...	7
Parameter 'num' is not assigned and could be declared final	8
Local variable 'rangen' could be declared final	10
Avoid using if statements without curly braces	13
Parameter 'leftmost' is not assigned and could be declare...	18
Parameter 'arr' is not assigned and could be declared final	18

## 5. 도구 기능 소개

### 5.4 CPD를 이용한 중복 코드 검사 (1/3)

PMD

- 중복된 코드를 찾을 때 사용 : CPD
  - 프로젝트 마우스 우 클릭 → PMD → Find Suspect Cut And Paste



## 5. 도구 기능 소개

### 5.4 CPD를 이용한 중복 코드 검사 (2/3)

PMD

- 중복 코드 검색 조건 설정 → OK
  - 중복된 코드를 찾고, 프로젝트의 report 폴더에 검색결과를 보고

The screenshot illustrates the PMD (PMD - Plugin for Microsoft Development) interface for running a CPD (Copy/Paste Detection) analysis. On the left, the 'Choose a language for the Copy/Paste detection' dialog is shown, with 'java' selected as the language and '15' as the minimum tile-size. The 'Report' section is checked, and 'Simple Text' is selected as the output format. An arrow points from this dialog to the main IDE window.

The main IDE window displays the 'Package Explorer' on the left, showing the project structure with a 'reports' folder containing 'cpd-report.txt'. The 'Partition.java' file is open in the editor, showing the following code:

```
package quicksortmodel;

import java.util.*;

public class Partition {
    static int cnt=1, ind=1;
    public int getRandomNum(int num) {
        int result = 0;
        Random rangen = new Random();
        result = rangen.nextInt() % num;

        if (result < 0) result = result
        return result;
    }
    public int randomDivide(int[] arr, int
```

At the bottom, the 'Violations Outline' and 'CPD View' panels are visible. The 'Violations Outline' panel shows an 'Error Message' section. The 'CPD View' panel shows a message: 'Found suspect cut & paste (2 matches, 3 lines)'.

## 5. 도구 기능 소개

### 5.4 CPD를 이용한 중복 코드 검사 (3/3)

PMD

- CPD View 창 : 중복된 코드에 대한 정보를 알림
  - Message중 한 개를 클릭하면 해당 코드를 찾아감

The screenshot displays two windows from an IDE. The top-left window is titled 'CPD View' and contains a table with two columns: 'Message' and 'Class'. It lists two messages: 'Found suspect cut & paste (2 matches, 3 lines)' for 'lines 27-29 in file Partition.java' and 'lines 30-32 in file Partition.java', both pointing to the class 'src.quicksortmodel.Partition'. The top-right window is titled 'Partition.java' and shows the source code of the 'randomDivide' method. The code includes a loop that prints the pivot value. The bottom window is another instance of the 'CPD View' window, showing the same list of messages.

Message	Class
Found suspect cut & paste (2 matches, 3 lines)	
lines 27-29 in file Partition.java	src.quicksortmodel.Partition
lines 30-32 in file Partition.java	src.quicksortmodel.Partition

```

public int randomDivide(int[] arr, int leftmost, int rightmost){
    Partition middle = new Partition();
    int key;
    if((rightmost-leftmost)<=0)
        return 0;
    else
    {
        key = middle.getRandomNum(rightmost-leftmost+1)+leftmost;
        int pivot = swap(arr,key,rightmost);
        if(ind%10==0){
            System.out.println();
            System.out.print((cnt++)+"번째: "+pivot+" ");
            ind++;
        }else{
            System.out.print((cnt++)+"번째: "+pivot+" ");
            ind++;
        }
    }
}
  
```

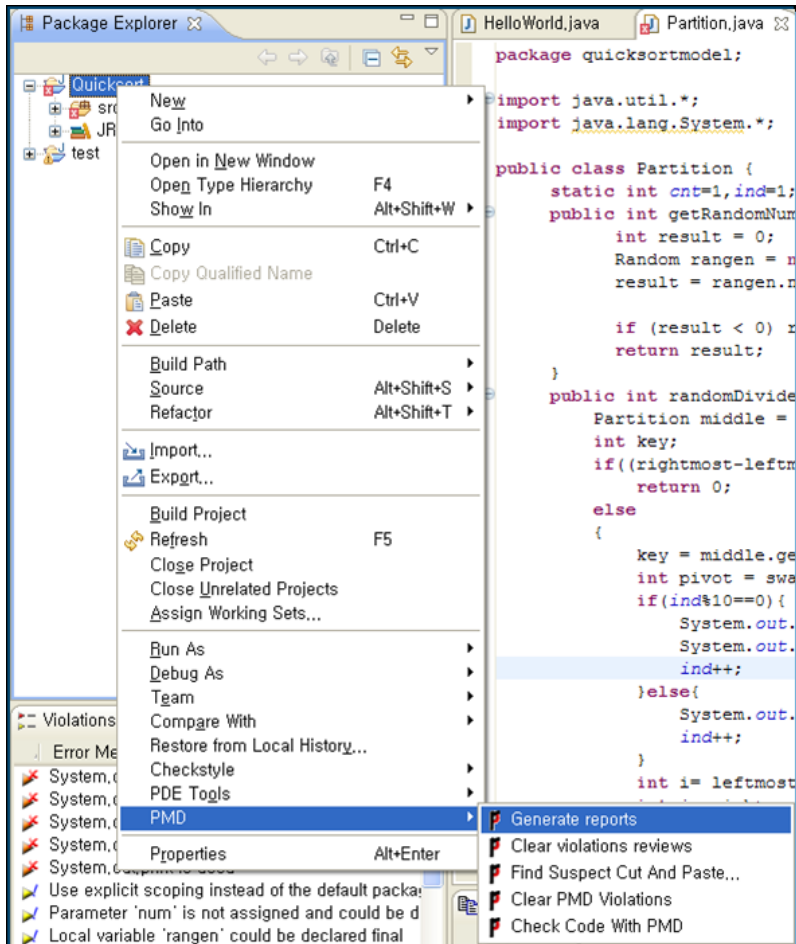
Message	Class
Found suspect cut & paste (2 matches, 3 lines)	
lines 27-29 in file Partition.java	src.quicksortmodel.Partition
lines 30-32 in file Partition.java	src.quicksortmodel.Partition

## 5. 도구 기능 소개

### 5.5 검사 결과 reporting하기 (1/3)

PMD

- 리포팅 하고자 하는 프로젝트에 대해 리포트 생성.
  - 프로젝트 마우스 우 클릭 → PMD → Generate reports

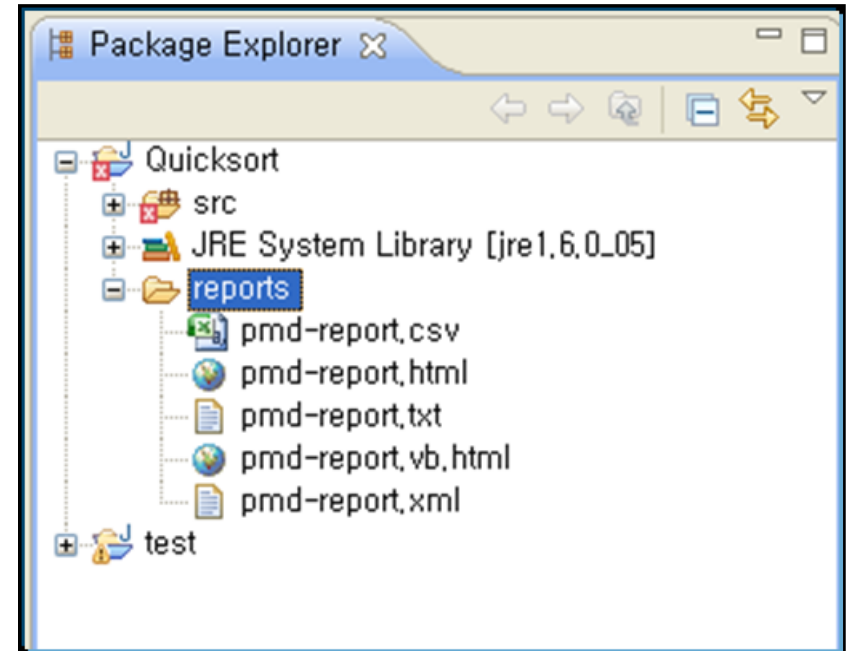
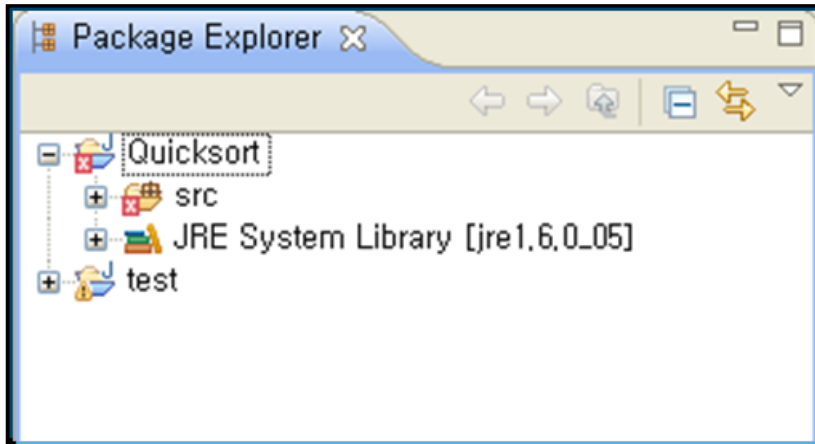


## 5. 도구 기능 소개

### 5.5 검사 결과 reporting하기 (2/3)

PMD

- 프로젝트 폴더에 reports라는 폴더가 있는것을 확인



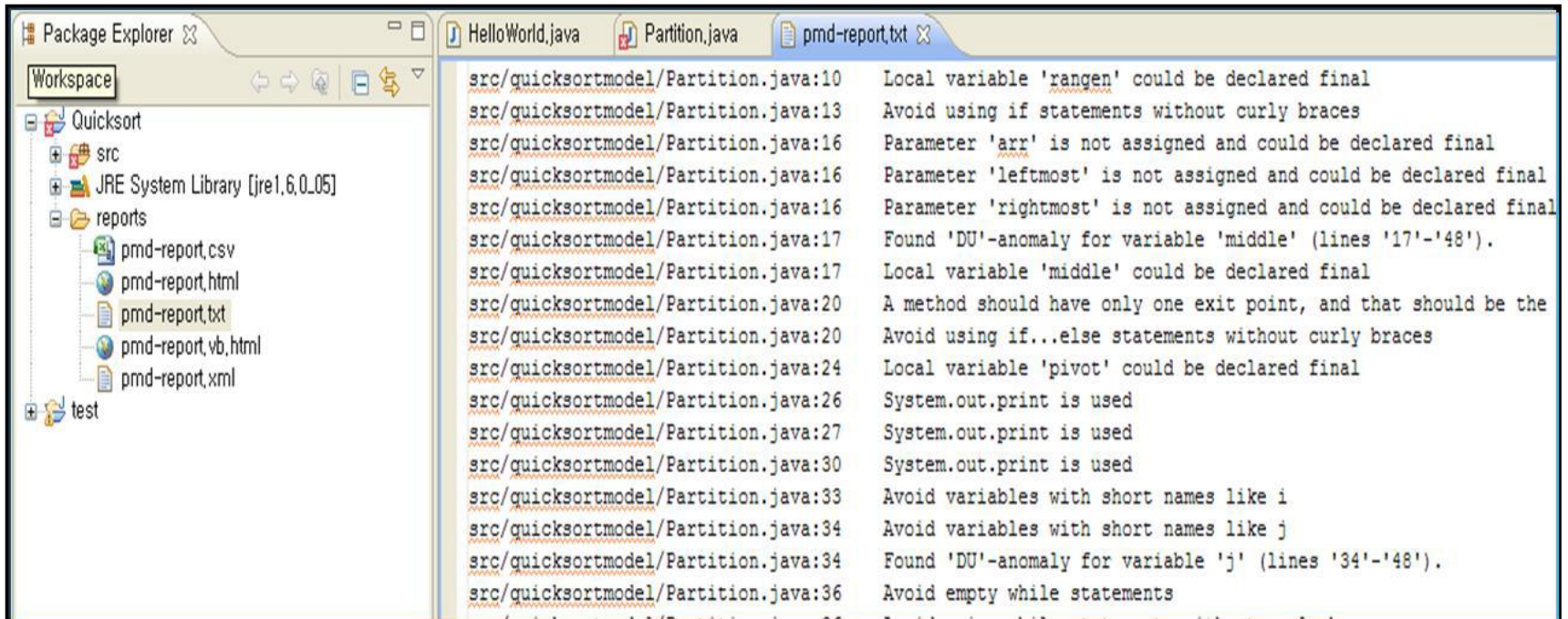


## 5. 도구 기능 소개

### 5.5 검사 결과 reporting하기 (3/3)

PMD

- Reports 폴더에 생성되는 파일
  - Pmd-report.csv : Microsoft Excel 파일
  - Pmd-report.html : html 파일
  - Pmd-report.txt : text 파일
  - Pmd-report.vb.html : VB 파일
  - Pmd-report.xml : xml 파일
  - 각 파일을 클릭하면 각각의 확장자에 따라서 레포팅 된 결과를 보임



## 6. 도구 활용 예제

### 세부 목차

### PMD

- 6.1 예제소개
- 6.2 요구사항 개발
- 6.3 PMD 코드 검사
- 6.4 PMD 코드 검사 결과
- 6.5 PMD 보고서 작성

## 6. 도구 활용 예제

### 6.1 예제소개 (1/2)

PMD

- 예제 시스템 : StringTokenizer
  - StringToken의 PMD 적용 과정

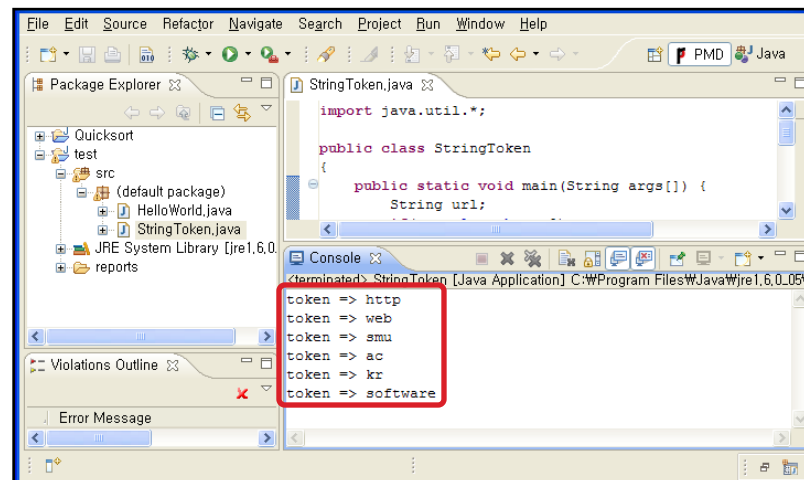
StringToken은 URL의 토큰을 분리해내는 프로그램

Java.util 패키지에 속해있는 StringTokenizer 클래스를 이용하여 토큰을 분리

PMD를 이용하여 소스코드를 검사하고 레포팅

URL이 한 단어씩 분리하여 출력

위반되는 코딩스타일을 사용하여 PMD가 검사할수 있게 함



## 6. 도구 활용 예제

### 6.1 예제소개 (2/2)

PMD

#### StringToken의 구현이슈

작성한 코드를 한눈에 파악하기 어렵고 디버깅이 힘들

잘못된 변수선언과 너무 짧은 변수 명을 사용

코드 검사한 결과를 문서로 남기어 정확한 수정이 필요



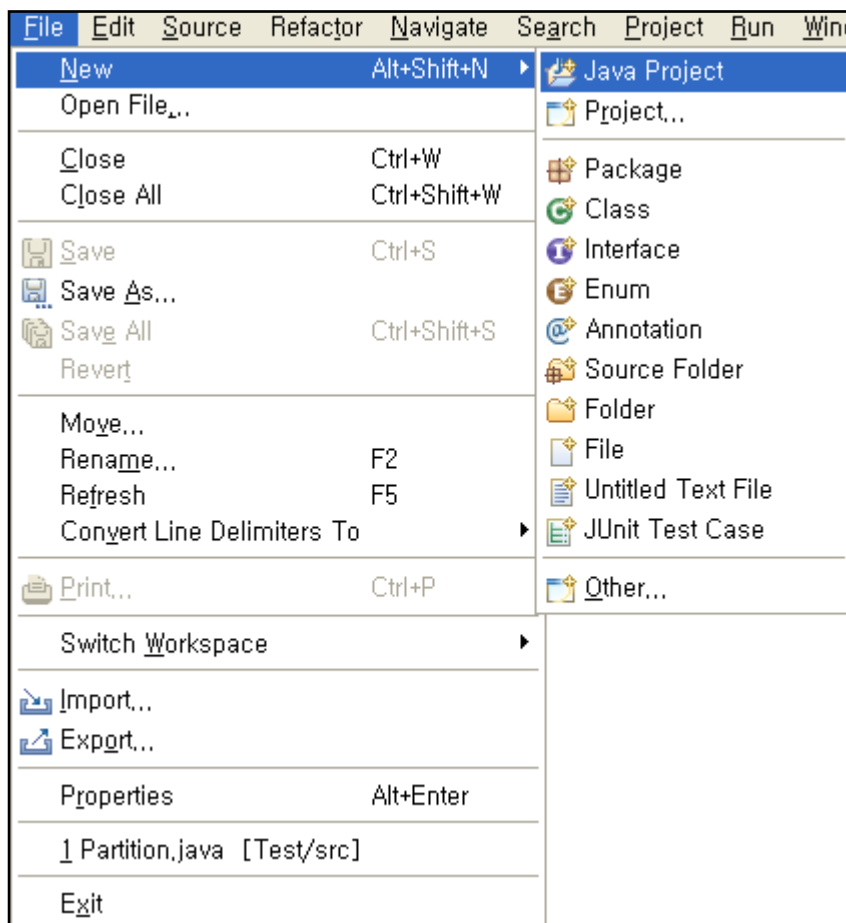
PMD를 도입하여 소스스타일 관리를 진행

## 6. 도구 활용 예제

### 6.2 요구사항 개발 (1/5)

PMD

- StringTokenizer을 위한 코드작성
  - File → New → Project → Java Project

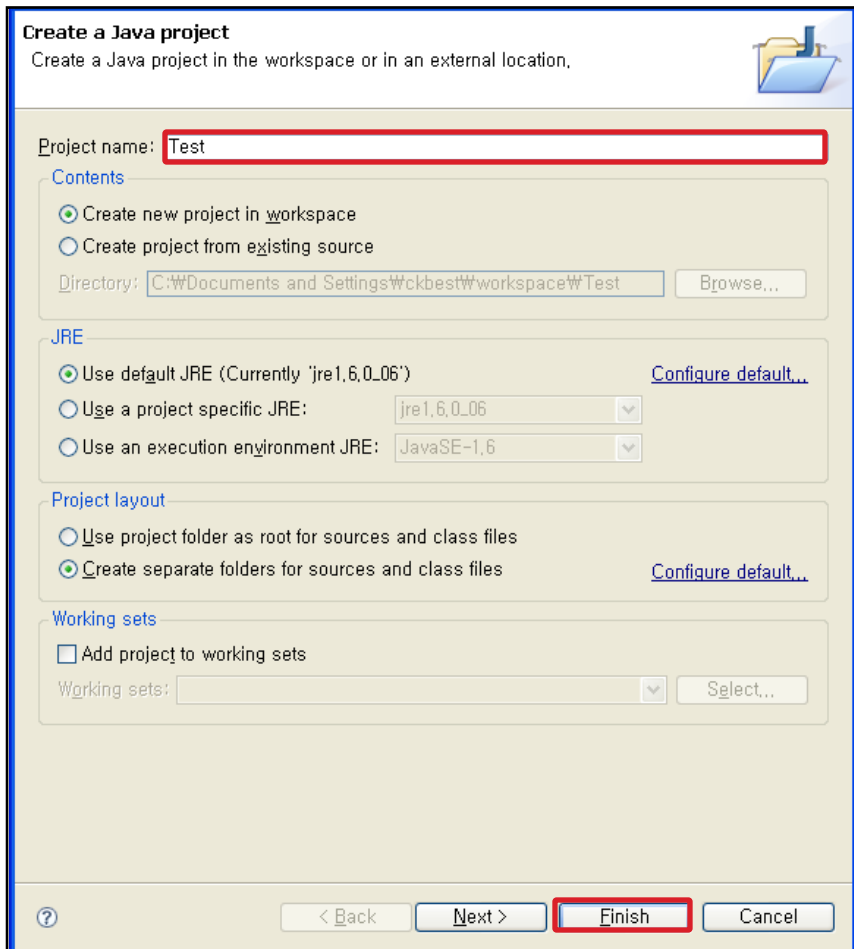


## 6. 도구 활용 예제

### 6.2 요구사항 개발 (2/5)

PMD

- Java project 생성
  - Project이름에 Test 설정 → Finish

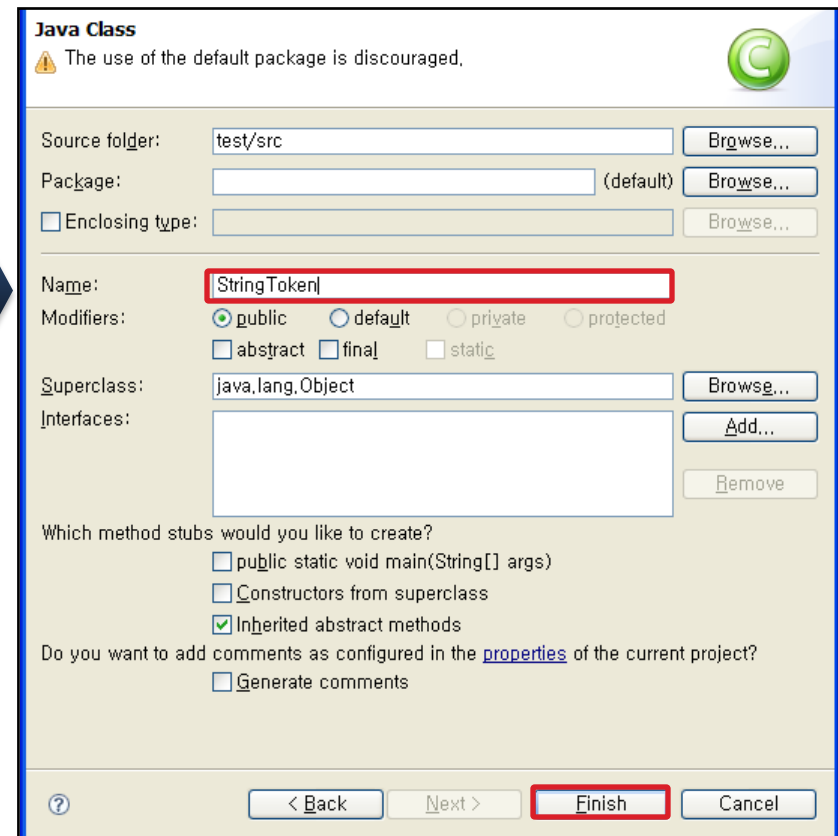
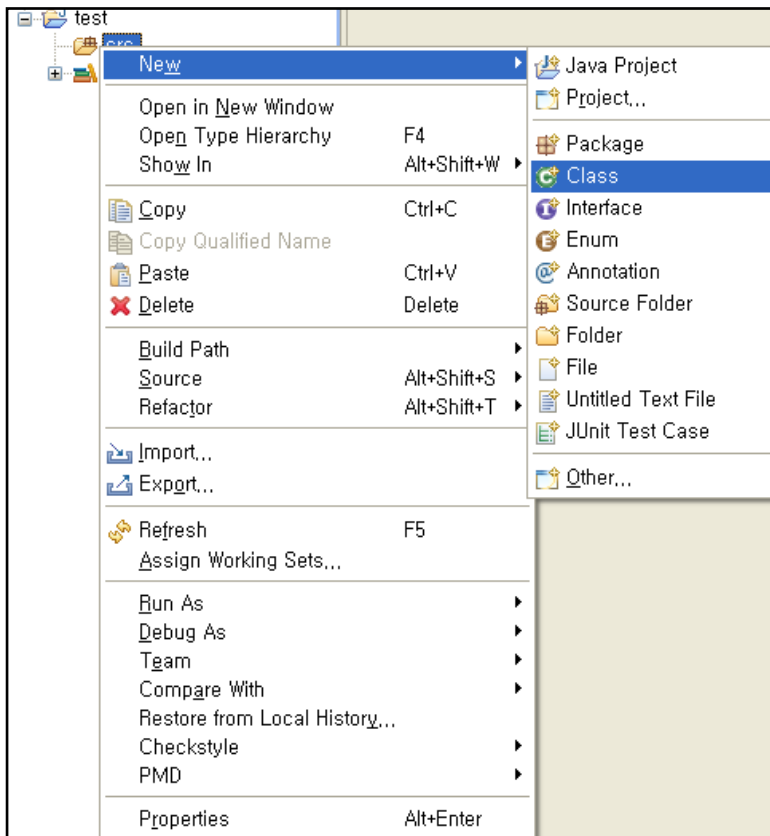


## 6. 도구 활용 예제

### 6.2 요구사항 개발 (3/5)

PMD

- StringToken 코드를 작성하기 위한 클래스 생성
  - Test프로젝트 마우스 우 클릭 → New → Class → Class Name "StringToken"입력 → Finish



## 6. 도구 활용 예제

### 6.2 요구사항 개발 (4/5)

#### PMD

- StringTokenizer 코드 작성
  - StringTokenizer(String str) : 문자열 str을 기본적인 구별자("WtWnWr)로 토큰을 구별을 위한 생성자
  - StringTokenizer(String str, String delim) : 문자열 str을 구별자(delim)로 토큰을 구별을 위한 생성자
  - StringTokenizer(String str, String delim, boolean returnToken)
    - > 문자열 str을 구별자(delim)로 토큰을 구별하고, 구별자를 토큰과 마찬가지로 nextToken()에서 반환할 여부를 returnToken 값으로 기술하는 생성자
  - Boolean hasMoreTokens() : 토큰이 더 이상 있는지에 대한 여부
  - String nextToken() : 다음 토큰 반환
  - String nextToken(String delim) – 새로운 구별자(delim)를 이용하여 다음토큰 반환
  - Int countTokens() –토큰의 개수 반환



```
StringToken.java
import java.util.*;

public class StringTokenizer
{
    public static void main(String args[]) {
        String url;
        if(args.length == 0)
            url = "http://web.smu.ac.kr/software/";
        else
            url = args[0];
        StringTokenizer st = new StringTokenizer(url, "/.~", false);
        while(st.hasMoreTokens()) {
            String token = st.nextToken();
            System.out.println("token => " + token);
        }
    }
}
```

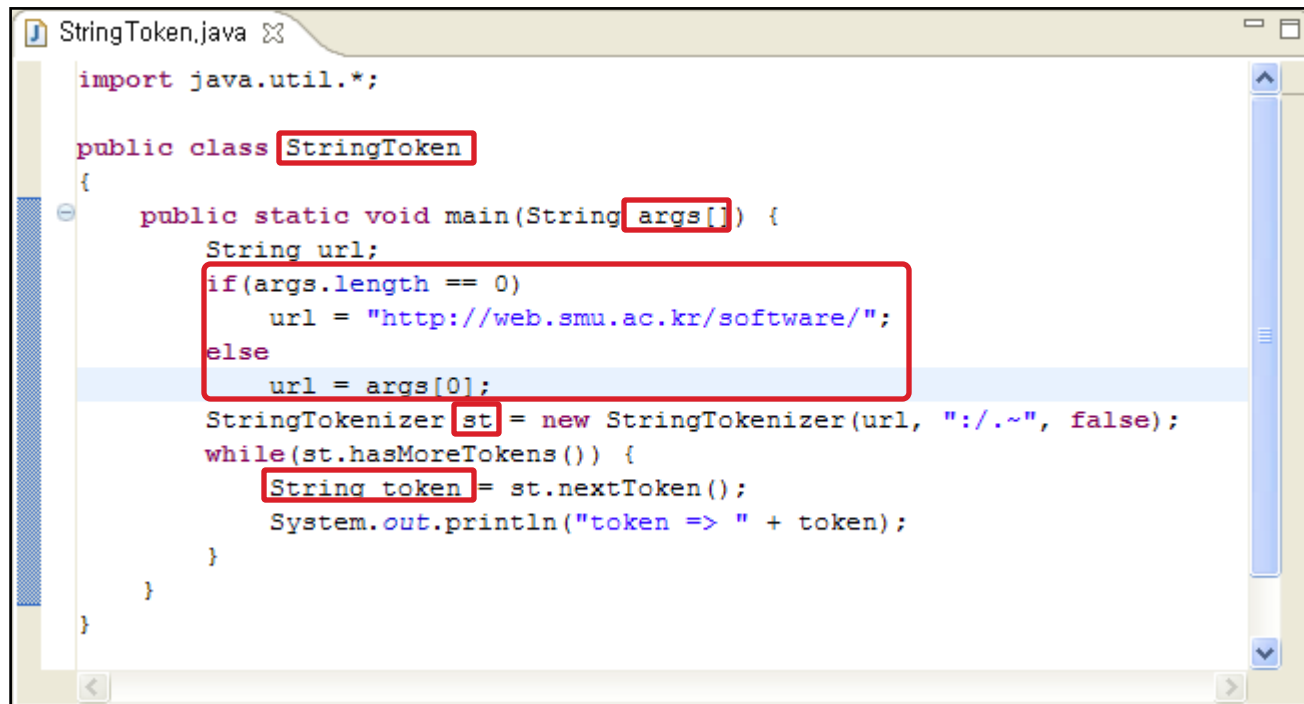


## 6. 도구 활용 예제

### 6.2 요구사항 개발 (5/5)

PMD

- 잘못된 코딩스타일의 사용 : PMD의 코딩스타일 추출 기능 확인을 위한 코드
  - 클래스 이름을 package이름과 동일하지 않게 설정 : 현재package 이름은default입니다
  - 사용되지 않는 파라미터 args를 선언
  - If else문에 중괄호를 사용하지 않음
  - St와 같이 짧은 변수이름을 사용
  - st와 token같이 지역 변수 이름을 final로 선언하지 않음



```
StringToken.java
import java.util.*;

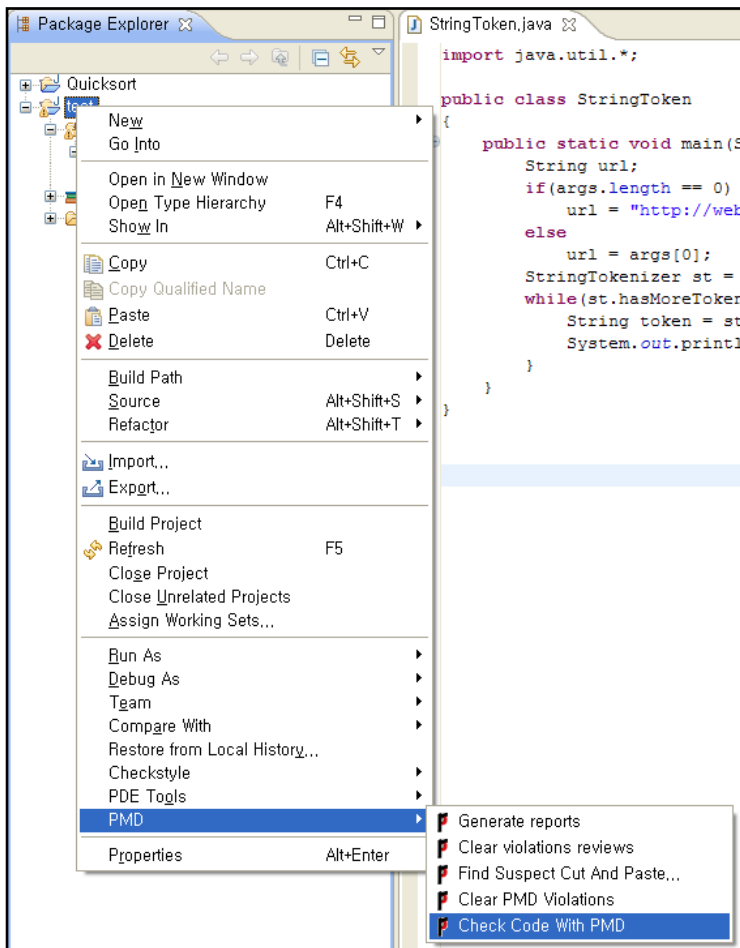
public class StringToken
{
    public static void main(String args[]) {
        String url;
        if(args.length == 0)
            url = "http://web.smu.ac.kr/software/";
        else
            url = args[0];
        StringTokenizer st = new StringTokenizer(url, ":/~", false);
        while(st.hasMoreTokens()) {
            String token = st.nextToken();
            System.out.println("token => " + token);
        }
    }
}
```

## 6. 도구 활용 예제

### 6.3 PMD의 코드 검사

PMD

- 코드 작성 후 PMD를 이용하여 코드 검사
  - 프로젝트 선택 후 마우스 우 클릭 → PMD → Check Code With PMD



## 6. 도구 활용 예제

### 6.4 PMD의 코드 검사 결과

PMD

- PMD의 코드검사 결과
  - 앞서 설정한 잘못된 코딩 스타일 모두 검출

The screenshot shows an IDE with the following components:

- Package Explorer:** Shows a project structure with 'Quicksort', 'test', 'src', '(default package)', 'StringToken.java', 'JRE System Library [jre1.6.0\_05]', and 'reports'.
- StringToken.java:** The code editor displays the following code:
 

```
import java.util.*;

public class StringToken
{
    public static void main(String args[]) {
        String url;
        if(args.length == 0)
            url = "http://web.smu.ac.kr/software/";
        else
            url = args[0];
        StringTokenizer st = new StringTokenizer(url, " / . ~", false);
        while(st.hasMoreTokens()) {
            String token = st.nextToken();
            System.out.println("token => " + token);
        }
    }
}
```
- Violations Outline:** Lists error messages:
  - System.out.print is used
  - All methods are static. Consider using Singleton in
  - All classes and interfaces must belong to a named
  - Parameter 'args' is not assigned and could be dec
  - Avoid using if...else statements without curly brace
  - Avoid using if...else statements without curly brace
  - Local variable 'st' could be declared final
  - Avoid variables with short names like st
  - Local variable 'token' could be declared final
- Violations Overview:** A table summarizing violations across the project.
 

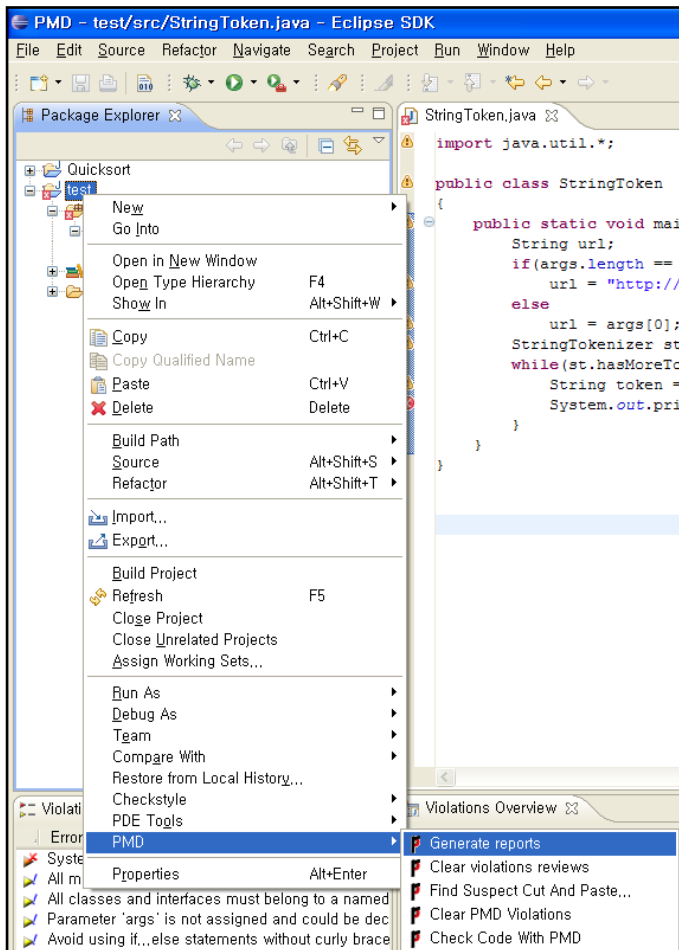
Element	# Violations	# Violations/LOC	# Violations/Method
(default package)	18	692.3 / 1000	9.00
StringToken.java	9	692.3 / 1000	9.00
SystemPrintln	1	76.9 / 1000	1.00
MethodArgumentCouldBeFinal	1	76.9 / 1000	1.00
NoPackage	1	76.9 / 1000	1.00
UseSingleton	1	76.9 / 1000	1.00
IFElseStrmtsMustUseBraces	2	153.8 / 1000	2.00
LocalVariableCouldBeFinal	2	153.8 / 1000	2.00
ShortVariable	1	76.9 / 1000	1.00
StringToken.java	9	692.3 / 1000	9.00

## 6. 도구 활용 예제

### 6.5 PMD보고서 작성 (1/2)

#### PMD

- PMD보고서 작성하기
  - 프로젝트 선택 후 마우스 우 클릭 → PMD → Generate reports

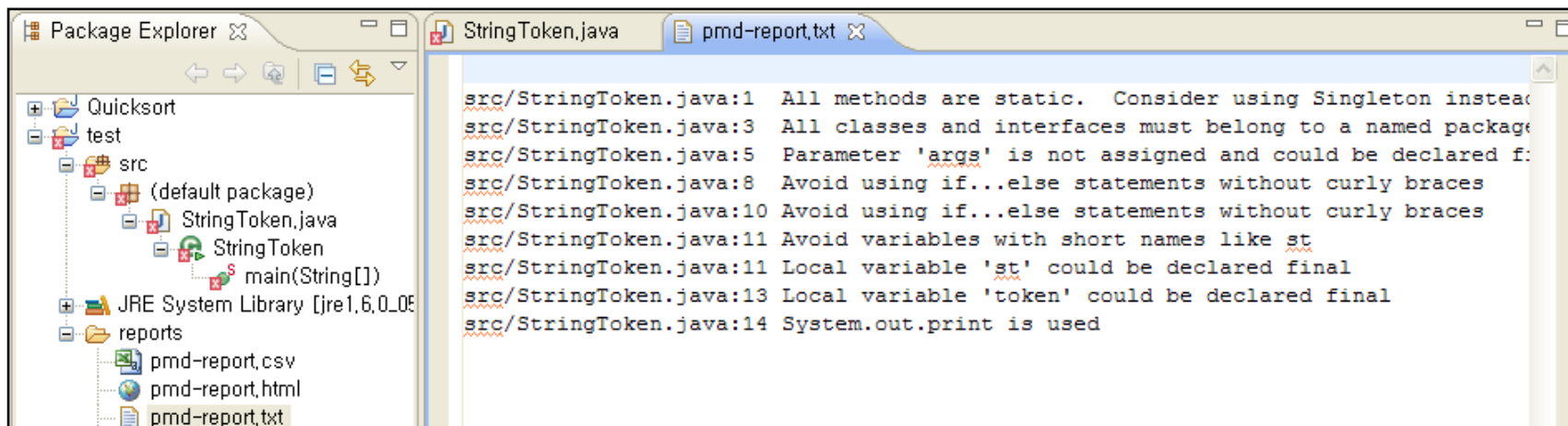


## 6. 도구 활용 예제

### 6.5 PMD보고서 작성 (2/2)

PMD

- PMD보고서 작성 결과
  - Test 프로젝트 폴더에 reports폴더가 생성되며, reports 폴더에 리포트 파일이 생성
  - 잘못된 코딩스타일에 대한 검사 및 보고가 이루어짐



- > 클래스 이름을 package이름과 동일하지 않게 설정
- > 사용되지 않는 파라미터 args를 선언
- > If else문에 중괄호를 사용하지 않음
- > St와 같이 짧은 변수이름을 사용
- > st와 token같이 지역 변수 이름을 final로 선언하지 않음

## 7. FAQ

### PMD

질문1) PMD의 약자는무엇인가요?

➡ 답변1 : PMD는 특정 단어의 약자는 아닙니다. 다음과 같은 여러 가지 의미를 중의적으로 사용하고 있습니다.  
Pretty Much Done / Project Mess Detector / Project Monitoring Directives /  
Project Meets Deadline / Programming Mistake Detector /  
Pounds Mistakes Dead / PMD Meaning Discovery /  
Programs of Mass Destruction / Programming Meticulous coDe

질문2) PMD는 Java에 대한 코드 분석만 지원하나요?

➡ 답변2 : 아닙니다. 2012년 5월 1일 Release된 5.0.0 버전부터 Java를 포함하여 JavaScript, XML, XSL, JSP도 지원합니다.

## 8. 도구 평가

### PMD

- 활용성
  - 개발자 편의대로 규칙을 바꾸어 사용할 수 있으며, 규칙에 따른 코드 관리로 활용하기에 용이
- 범용성
  - 대부분의 자바를 지원하는 도구에서 사용 가능
- 호환성
  - Eclipse위주로 개발된 플러그인이나, 제한적으로 다른 도구에서 사용 가능
- 성능
  - 빠른 작동성능, 도구에 같이 로드 되고, 사용되어도 일반적으로 큰 문제가 없음
- 기타
  - 5단계 UI 디자인을 적용하여, 코딩의 문제점에 대한 단계별 인지가 용이

#### 도구평가 의견

- 정해진 규칙에 따라 코드를 관리하여 코딩 컨벤션을 대신할 수 있음
- 리포팅을 이용하여 잘못 작성된 코드를 쉽게 이해할 수 있음
- 개발자의 편의대로 규칙을 바꿔 원하는 규칙으로 PMD 기능 이용 가능

## 9. 용어 정리

PMD

### 본 매뉴얼에서 사용하고 있는 용어의 정리

CPD	Copy/Paste Detector. 중복된 코드를 검사해주는 PMD의 기능 중 하나
URL	Uniform Resource Identifier. 웹 상에서 서비스를 제공하는 각 서버들에 있는 파일들의 위치를 명시하기 위한 것. 접속해야 될 서비스의 종류, 서버의 위치(도메인네임), 파일의 위치를 포함
코딩스타일	Programming style. 컴퓨터 프로그램의 소스코드를 작성할 때 사용되는 규칙 혹은 가이드라인
코딩 컨벤션	프로그래머가 소스코드를 쉽게 이해하고 유지하는 것을 돕기 위해 따르는 규칙. 컨벤션은 문서화 되어 모든 팀이나 회사 동료들이 따르도록 정규화 될 수 있고, 또한 각 개인의 코딩 습관에 따라 다를 수 있음
토큰	문자열의 분류화된 블록(block). 어휘분석기(lexical analyzer)에서 문자열을 스캐닝한 뒤 토큰화 하면 생성되며, 문자열을 토큰화 할 경우 프로그래머가 정의하기에 따라 여러 가지 방식으로 토큰을 나눔