

Software Verification

2nd Testing

T 3

200711453 류진렬

200711477 황진수

200711428 박기성

201360220 황 민

200312489 유현덕

Contents

- Category Partitioning Testing
- Pairwise Combinational Testing
- Static Analysis
 - ❖ Clover
 - ❖ Sonar
 - ❖ PMD

Category Partitioning Testing

Category Partitioning

▪ Step 1. Choosing categories

- 요구사항 명세서의 Functional Requirement를 기반으로 항목을 분류하고 unit들을 정의

분류	항목
시스템	파일 입출력
그리기	색상
	그리기 도구
	선 두께
효과	특수 효과

Category Partitioning Testing

- Step 2. Identify Representative Value

- 파일 입출력
 - 파일 저장하기
 - 파일 불러오기

- 색상
 - 선택 안 함(default)
 - 색 편집
 - 색 선택

Category Partitioning Testing

- Step 2. Identify Representative Value
- 그리기 도구
 - 연필
 - 지우개
 - 선 그리기
 - 색 채우기
 - 브러시 – normal
 - 브러시 – special1
 - 브러시 – special2
 - 도형 – 원
 - 도형 – 사각형

Category Partitioning Testing

- Step 2. Identify Representative Value
- 선 두께
 - 1
 - 5
 - 10
 - 15
 - 20
- 특수 효과
 - 선택 안 함
 - 초기화
 - 데칼코마니
 - 점선으로 변경
 - 그림자 효과

Category Partitioning

- **Step 3. Generate Test Case Specifications**
- 파일 I/O (2) * 그리기도구 (9) * 색상 (3) * 선두께 (5) * 특수효과 (5) = 1350 개의 test cases.
- Property Constraints와 Single Constraints 정의

Category Partitioning

▪ Step 3. Generate Test Case Specifications

▪ Property Constraints – property 설정

✓ 그리기 도구 항목

- 색채우기 [property DT]
- 연필 [property DT]
- 지우개 [property DTWD]
- 도형(원) [property DT] [property DTWD]
- 도형(사각형) [property DT] [property DTWD]
- 브러시(노말) [property DT] [property DTWD]
- 브러시(special1) [property DT] [property DTWD]
- 브러시(special2) [property DT] [property DTWD]
- 선 [property DT] [property DTWD]

Category Partitioning

▪ Step 3. Generate Test Case Specifications

▪ Property Constraints – if property 설정

✓ 색상 항목

- default [if DT]
- 색편집 [if DT]
- 색선택 [if DT]

✓ 선두께 항목

- 1 [if DTWD]
- 5 [if DTWD]
- 10 [if DTWD]
- 15 [if DTWD]
- 20 [if DTWD]

Category Partitioning

▪ Step 3. Generate Test Case Specifications

- Single Constraints (combinational test case들 중에서 특정 카테고리의 value가 반복적으로 수행될 필요 없이 단 한 번만 포함되어 테스트되면 되는 경우)

- ✓ 특수효과 항목

- 데칼코마니 [single]
- 점선으로 변경 [single]
- 그림자 효과 [single]
- 초기화 [single]

Category Partitioning

▪ Step 3. Generate Test Case Specifications

▪ Property Constraints와 Single Constraints 적용 후 카테고리

- | | | |
|-------------------|--|-------------------|
| ▪ 파일 I/O | ▪ 그리기 도구 | ▪ 선두께 |
| ▪ 저장 | ▪ 색채우기 [property DT] | ▪ 1 [if DTWD] |
| ▪ 불러오기 | ▪ 연필 [property DT] | ▪ 5 [if DTWD] |
| ▪ 특수효과 | ▪ 지우개 [property DTWD] | ▪ 10 [if DTWD] |
| ▪ 데칼코마니(single) | ▪ 도형-사각형 [property DT] [property DTWD] | ▪ 15 [if DTWD] |
| ▪ 점선으로 변경(single) | ▪ 도형-원 [property DT] [property DTWD] | ▪ 20 [if DTWD] |
| ▪ 그림자효과(single) | ▪ 브러시-노말 [property DT] [property DTWD] | ▪ 색상 |
| ▪ 초기화(single) | ▪ 브러시-special1 [property DT] [property DTWD] | ▪ Default [if DT] |
| ▪ 효과 없음(single) | ▪ 브러시-special2 [property DT] [property DTWD] | ▪ 색편집 [if DT] |
| | ▪ 선 [property DT] [property DTWD] | ▪ 색선택 [if DT] |

▪ Property Constraints와 Single Constraints 적용 후 테스트 케이스 수

$$\checkmark (2 * 2 * 3 * 1 * 1) + (2 * 1 * 1 * 5 * 1) + (2 * 6 * 3 * 5 * 1) + 4$$

$$= 206 \text{ 가지.}$$

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
1	1.1.1.1.5	색채우기를 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
2	1.1.2.1.5	색채우기를 선택하고 색 편집으로 임의의 색을 선택하고, 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
3	1.1.3.1.5	색채우기를 선택하고 색 선택으로 선택한 색상으로, 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
4	1.2.1.1.5	연필을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
5	1.2.2.1.5	연필을 선택하고 색 편집으로 임의의 색을 선택하고, 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
6	1.2.3.1.5	연필을 선택하고 색 선택으로 선택한 색상으로, 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
7	1.3.1.1.5	지우개를 선택하고 default 색상으로 선 두께 1 짜리 지우개로 그림을 아무런 효과 없이 지우고 저장한다.	Pass
8	1.3.1.2.5	지우개를 선택하고 default 색상으로 선 두께 5 짜리 지우개로 그림을 아무런 효과 없이 지우고 저장한다.	Pass
9	1.3.1.3.5	지우개를 선택하고 default 색상으로 선 두께 10 짜리 지우개로 그림을 아무런 효과 없이 지우고 저장한다.	Pass
10	1.3.1.4.5	지우개를 선택하고 default 색상으로 선 두께 15 짜리 지우개로 그림을 아무런 효과 없이 지우고 저장한다.	Pass
11	1.3.1.5.5	지우개를 선택하고 default 색상으로 선 두께 20 짜리 지우개로 그림을 아무런 효과 없이 지우고 저장한다.	Pass
12	1.4.1.1.5	도형 - 원을 선택하고 default 색상으로 선 두께 1 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
13	1.4.1.2.5	도형 - 원을 선택하고 default 색상으로 선 두께 5 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
14	1.4.1.3.5	도형 - 원을 선택하고 default 색상으로 선 두께 10 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
15	1.4.1.4.5	도형 - 원을 선택하고 default 색상으로 선 두께 15 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
16	1.4.1.5.5	도형 - 원을 선택하고 default 색상으로 선 두께 20 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
17	1.4.2.1.5	도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 1 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
18	1.4.2.2.5	도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 5 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
19	1.4.2.3.5	도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 10 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
20	1.4.2.4.5	도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 15 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
21	1.4.2.5.5	도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 20 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
22	1.4.3.1.5	도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 1 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
23	1.4.3.2.5	도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 5 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
24	1.4.3.3.5	도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 10 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
25	1.4.3.4.5	도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 15 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
26	1.4.3.5.5	도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 20 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
27	1.5.1.1.5	도형 - 사각형을 선택하고 default 색상으로 선 두께 1 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
28	1.5.1.2.5	도형 - 사각형을 선택하고 default 색상으로 선 두께 5 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
29	1.5.1.3.5	도형 - 사각형을 선택하고 default 색상으로 선 두께 10 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
30	1.5.1.4.5	도형 - 사각형을 선택하고 default 색상으로 선 두께 15 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
31	1.5.1.5.5	도형 - 사각형을 선택하고 default 색상으로 선 두께 20 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
32	1.5.2.1.5	도형 - 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 1 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
33	1.5.2.2.5	도형 - 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 5 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
34	1.5.2.3.5	도형 - 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 10 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
35	1.5.2.4.5	도형 - 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 15 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
36	1.5.2.5.5	도형 - 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 20 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
37	1.5.3.1.5	도형 - 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 1 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
38	1.5.3.2.5	도형 - 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 5 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
39	1.5.3.3.5	도형 - 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 10 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
40	1.5.3.4.5	도형 - 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 15 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
41	1.5.3.5.5	도형 - 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 20 짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
42	2.1.1.1.5	파일을 불러온 후 색채우기를 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
43	2.1.2.1.5	파일을 불러온 후 색채우기를 선택하고 색 편집으로 임의의 색을 선택하고, 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
44	2.1.3.1.5	파일을 불러온 후 색채우기를 선택하고 색 선택으로 선택한 색상으로, 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
45	2.2.1.1.5	파일을 불러온 후 연필을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
46	2.2.2.1.5	파일을 불러온 후 연필을 선택하고 색 편집으로 임의의 색을 선택하고, 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
47	2.2.3.1.5	파일을 불러온 후 연필을 선택하고 색 선택으로 선택한 색상으로, 선 두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
48	2.3.1.1.5	파일을 불러온 후 지우개를 선택하고 default 색상으로 선 두께 1 짜리 지우개로 그림을 아무런 효과 없이 지운다.	Pass
49	2.3.1.2.5	파일을 불러온 후 지우개를 선택하고 default 색상으로 선 두께 5 짜리 지우개로 그림을 아무런 효과 없이 지운다.	Pass
50	2.3.1.3.5	파일을 불러온 후 지우개를 선택하고 default 색상으로 선 두께 10 짜리 지우개로 그림을 아무런 효과 없이 지운다.	Pass
51	2.3.1.4.5	파일을 불러온 후 지우개를 선택하고 default 색상으로 선 두께 15 짜리 지우개로 그림을 아무런 효과 없이 지운다.	Pass
52	2.3.1.5.5	파일을 불러온 후 지우개를 선택하고 default 색상으로 선 두께 20 짜리 지우개로 그림을 아무런 효과 없이 지운다.	Pass
53	2.4.1.1.5	파일을 불러온 후 도형 - 원을 선택하고 default 색상으로 선 두께 1 짜리 그림을 아무런 효과 없이 그린다.	Pass
54	2.4.1.2.5	파일을 불러온 후 도형 - 원을 선택하고 default 색상으로 선 두께 5 짜리 그림을 아무런 효과 없이 그린다.	Pass
55	2.4.1.3.5	파일을 불러온 후 도형 - 원을 선택하고 default 색상으로 선 두께 10 짜리 그림을 아무런 효과 없이 그린다.	Pass
56	2.4.1.4.5	파일을 불러온 후 도형 - 원을 선택하고 default 색상으로 선 두께 15 짜리 그림을 아무런 효과 없이 그린다.	Pass
57	2.4.1.5.5	파일을 불러온 후 도형 - 원을 선택하고 default 색상으로 선 두께 20 짜리 그림을 아무런 효과 없이 그린다.	Pass
58	2.4.2.1.5	파일을 불러온 후 도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 1 짜리 그림을 아무런 효과 없이 그린다.	Pass
59	2.4.2.2.5	파일을 불러온 후 도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 5 짜리 그림을 아무런 효과 없이 그린다.	Pass
60	2.4.2.3.5	파일을 불러온 후 도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 10 짜리 그림을 아무런 효과 없이 그린다.	Pass
61	2.4.2.4.5	파일을 불러온 후 도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 15 짜리 그림을 아무런 효과 없이 그린다.	Pass
62	2.4.2.5.5	파일을 불러온 후 도형 - 원을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 20 짜리 그림을 아무런 효과 없이 그린다.	Pass
63	2.4.3.1.5	파일을 불러온 후 도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 1 짜리 그림을 아무런 효과 없이 그린다.	Pass
64	2.4.3.2.5	파일을 불러온 후 도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 5 짜리 그림을 아무런 효과 없이 그린다.	Pass
65	2.4.3.3.5	파일을 불러온 후 도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 10 짜리 그림을 아무런 효과 없이 그린다.	Pass
66	2.4.3.4.5	파일을 불러온 후 도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 15 짜리 그림을 아무런 효과 없이 그린다.	Pass
67	2.4.3.5.5	파일을 불러온 후 도형 - 원을 선택하고 색 선택으로 선택한 색상으로 선 두께 20 짜리 그림을 아무런 효과 없이 그린다.	Pass
68	2.5.1.1.5	파일을 불러온 후 도형 - 사각형을 선택하고 default 색상으로 선 두께 1 짜리 그림을 아무런 효과 없이 그린다.	Pass
69	2.5.1.2.5	파일을 불러온 후 도형 - 사각형을 선택하고 default 색상으로 선 두께 5 짜리 그림을 아무런 효과 없이 그린다.	Pass

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
70	2.5.1.3.5	파일을 불러온 후 도형 – 사각형을 선택하고 default 색상으로 선 두께 10 짜리 그림을 아무런 효과 없이 그린다.	Pass
71	2.5.1.4.5	파일을 불러온 후 도형 – 사각형을 선택하고 default 색상으로 선 두께 15 짜리 그림을 아무런 효과 없이 그린다.	Pass
72	2.5.1.5.5	파일을 불러온 후 도형 – 사각형을 선택하고 default 색상으로 선 두께 20 짜리 그림을 아무런 효과 없이 그린다.	Pass
73	2.5.2.1.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 1 짜리 그림을 아무런 효과 없이 그린다.	Pass
74	2.5.2.2.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 5 짜리 그림을 아무런 효과 없이 그린다.	Pass
75	2.5.2.3.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 10 짜리 그림을 아무런 효과 없이 그린다.	Pass
76	2.5.2.4.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 15 짜리 그림을 아무런 효과 없이 그린다.	Pass
77	2.5.2.5.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 편집으로 임의의 색을 선택하고 선 두께 20 짜리 그림을 아무런 효과 없이 그린다.	Pass
78	2.5.3.1.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 1 짜리 그림을 아무런 효과 없이 그린다.	Pass
79	2.5.3.2.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 5 짜리 그림을 아무런 효과 없이 그린다.	Pass
80	2.5.3.3.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 10 짜리 그림을 아무런 효과 없이 그린다.	Pass
81	2.5.3.4.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 15 짜리 그림을 아무런 효과 없이 그린다.	Pass
82	2.5.3.5.5	파일을 불러온 후 도형 – 사각형을 선택하고 색 선택으로 선택한 색상으로 선 두께 20 짜리 그림을 아무런 효과 없이 그린다.	Pass
83	1.6.1.1.5	브러시-노말을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
84	1.6.1.2.5	브러시-노말을 선택하고 default 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
85	1.6.1.3.5	브러시-노말을 선택하고 default 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
86	1.6.1.4.5	브러시-노말을 선택하고 default 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
87	1.6.1.5.5	브러시-노말을 선택하고 default 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
88	1.6.2.1.5	브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
89	1.6.2.2.5	브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
90	1.6.2.3.5	브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
91	1.6.2.4.5	브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
92	1.6.2.5.5	브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
93	1.6.3.1.5	브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
94	1.6.3.2.5	브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
95	1.6.3.3.5	브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
96	1.6.3.4.5	브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
97	1.6.3.5.5	브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
98	1.7.1.1.5	브러시-special1을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
99	1.7.1.2.5	브러시-special1을 선택하고 default 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
100	1.7.1.3.5	브러시-special1을 선택하고 default 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
101	1.7.1.4.5	브러시-special1을 선택하고 default 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
102	1.7.1.5.5	브러시-special1을 선택하고 default 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
103	1.7.2.1.5	브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
104	1.7.2.2.5	브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
105	1.7.2.3.5	브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
106	1.7.2.4.5	브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
107	1.7.2.5.5	브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
108	1.7.3.1.5	브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
109	1.7.3.2.5	브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
110	1.7.3.3.5	브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
111	1.7.3.4.5	브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
112	1.7.3.5.5	브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
113	2.6.1.1.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
114	2.6.1.2.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 default 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
115	2.6.1.3.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 default 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
116	2.6.1.4.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 default 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
117	2.6.1.5.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 default 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
118	2.6.2.1.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
119	2.6.2.2.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
120	2.6.2.3.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
121	2.6.2.4.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
122	2.6.2.5.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색편집으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
123	2.6.3.1.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
124	2.6.3.2.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
125	2.6.3.3.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
126	2.6.3.4.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
127	2.6.3.5.5	기존의 이미지를 불러와서 브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
128	2.7.1.1.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
129	2.7.1.2.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 default 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
130	2.7.1.3.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 default 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
131	2.7.1.4.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 default 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
132	2.7.1.5.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 default 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
133	2.7.2.1.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
134	2.7.2.2.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
135	2.7.2.3.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
136	2.7.2.4.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
137	2.7.2.5.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색편집으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
138	2.7.3.1.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
139	2.7.3.2.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
140	2.7.3.3.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
141	2.7.3.4.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
142	2.7.3.5.5	기존의 이미지를 불러와서 브러시-special1을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
143	1.8.1.1.5	브러시 special2을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
144	1.8.1.2.5	브러시 special2을 선택하고 default 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
145	1.8.1.3.5	브러시 special2을 선택하고 default 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
146	1.8.1.4.5	브러시 special2을 선택하고 default 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
147	1.8.1.5.5	브러시 special2을 선택하고 default 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
148	1.8.2.1.5	브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
149	1.8.2.2.5	브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
150	1.8.2.3.5	브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
151	1.8.2.4.5	브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
152	1.8.2.5.5	브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
153	1.8.3.1.5	브러시 special2을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
154	1.8.3.2.5	브러시 special2을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
155	1.8.3.3.5	브러시 special2을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
156	1.8.3.4.5	브러시 special2을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
157	1.8.3.5.5	브러시 special2을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
158	1.9.1.1.5	선을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
159	1.9.1.2.5	선을 선택하고 default 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
160	1.9.1.3.5	선을 선택하고 default 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
161	1.9.1.4.5	선을 선택하고 default 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
162	1.9.1.5.5	선을 선택하고 default 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
163	1.9.2.1.5	선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
164	1.9.2.2.5	선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
165	1.9.2.3.5	선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
166	1.9.2.4.5	선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
167	1.9.2.5.5	선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
168	1.9.3.1.5	선을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
169	1.9.3.2.5	선을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
170	1.9.3.3.5	선을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
171	1.9.3.4.5	선을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
172	1.9.3.5.5	선을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
173	2.8.1.1.5	기존의 그림을 불러와서 브러시 special2을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
174	2.8.1.2.5	기존의 그림을 불러와서 브러시 special2을 선택하고 default 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
175	2.8.1.3.5	기존의 그림을 불러와서 브러시 special2을 선택하고 default 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
176	2.8.1.4.5	기존의 그림을 불러와서 브러시 special2을 선택하고 default 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
177	2.8.1.5.5	기존의 그림을 불러와서 브러시 special2을 선택하고 default 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
178	2.8.2.1.5	기존의 그림을 불러와서 브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
179	2.8.2.2.5	기존의 그림을 불러와서 브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
180	2.8.2.3.5	기존의 그림을 불러와서 브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
181	2.8.2.4.5	기존의 그림을 불러와서 브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
182	2.8.2.5.5	기존의 그림을 불러와서 브러시 special2을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
183	2.8.3.1.5	기존의 그림을 불러와서 브러시 special2을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
184	2.8.3.2.5	기존의 그림을 불러와서 브러시 special2을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass

Category Partitioning – Test Cases

No.	Test Case No.	설명	결과
185	2.8.3.3.5	기존의 그림을 불러와서 브러시 special2을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
186	2.8.3.4.5	기존의 그림을 불러와서 브러시 special1을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
187	2.8.3.5.5	기존의 그림을 불러와서 브러시 special1을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
188	2.9.1.1.5	기존의 그림을 불러와서 선을 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
189	2.9.1.2.5	기존의 그림을 불러와서 선을 선택하고 default 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
190	2.9.1.3.5	기존의 그림을 불러와서 선을 선택하고 default 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
191	2.9.1.4.5	기존의 그림을 불러와서 선을 선택하고 default 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
192	2.9.1.5.5	기존의 그림을 불러와서 선을 선택하고 default 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
193	2.9.2.1.5	기존의 그림을 불러와서 선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
194	2.9.2.2.5	기존의 그림을 불러와서 선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
195	2.9.2.3.5	기존의 그림을 불러와서 선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
196	2.9.2.4.5	기존의 그림을 불러와서 선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
197	2.9.2.5.5	기존의 그림을 불러와서 선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
198	2.9.3.1.5	기존의 그림을 불러와서 선을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그린다.	Pass
199	2.9.3.2.5	기존의 그림을 불러와서 선을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
200	2.9.3.3.5	기존의 그림을 불러와서 선을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 아무런 효과 없이 그린다.	Pass
201	2.9.3.4.5	기존의 그림을 불러와서 선을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 아무런 효과 없이 그린다.	Pass
202	2.9.3.5.5	기존의 그림을 불러와서 선을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그린다.	Pass
203	2.5.2.3.1	기존의 그림을 불러와서 도형-네모를 선택하고 색편집으로 선택한 임의의 색상으로 선두께 10짜리 그림을 그린 후 데칼코마니 효과를 적용한다.	Pass
204	2.9.2.4.2	기존의 그림을 불러와서 선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 15짜리 그림을 그린 후 점선으로 변경 효과를 적용한다.	Pass
205	2.8.1.1.3	기존의 그림을 불러와서 브러시 special2을 선택하고 default 색상으로 선두께 1짜리 그림을 그린 후 그림자 효과를 적용한다.	Pass
206	1.4.3.5.4	도형-원을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 그린 후 초기화 효과를 적용하여 저장한다.	Pass

Pairwise Combination Testing

Pairwise Combination Testing

- Property constraints와 Single constraints 적용 후 test cases =
File I/O (2) * 그리기도구(9) * 색상(3) * 선 두께(5) * 특수효과(1) = 274가지의
test case들을 PICT(pairwise combination test tool)을 사용하여 추림

입력
TC.txt
문서

결과
res.txt
문서

Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

```
C:\Users\WJin>cd ..
C:\Users>cd ..
C:\>cd Program Files
C:\Program Files>cd PICT
C:\Program Files\PICT>pic t c.txt > res.txt
Input Error: Parameter/value type mismatch: IF [그리기] = "색채우기" THEN [선두께] = "1";
C:\Program Files\PICT>pic t c.txt > res.txt
C:\Program Files\PICT>
```

커맨드를 이용하여
Pairwise combination
Test case 생성

Pairwise Combination Testing

- 206가지의 test case들이 45가지로 추려짐
(IF 조건문을 사용하여 Property constraints를 반영시킬 수 있다.)

TC.txt 의 내용 :

파일IO : 저장하기, 불러오기

그리기 : 연필, 색채우기, 지우개, 동그라미, 네모, 브러시N,
브러시S1, 브러시S2, 선

색상 : 기본, 색편집, 색선택

선두께 : 1, 5, 10, 15, 20

```
IF [그리기] = "지우개" THEN [색상] = "기본";
```

```
IF [그리기] = "색채우기" THEN [선두께] = 1;
```

```
IF [그리기] = "연필" THEN [선두께] = 1;
```

Pairwise Combination Testing – Test Cases

No.	Test Case No.	설명	결과
1	1.1.1.1.5	색채우기를 선택하고 default 색상으로 선두께 1짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
2	1.9.2.5.5	선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 20짜리 그림을 아무런 효과 없이 그리고 저장한다.	Pass
3	2.3.1.3.5	기존의 그림을 불러와서 지우개를 선택하고 선두께 10짜리로 그림을 지운다.	Pass
4	2.7.3.2.5	기존의 그림을 불러와서 브러시 S1을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
5	2.8.2.2.5	기존의 그림을 불러와서 브러시 S2를 선택하고 색편집으로 선택한 임의의 색상으로 선두께 5짜리 그림을 아무런 효과 없이 그린다.	Pass
6	1.6.3.4.5	브러시 노말을 선택하고 색선택으로 선택한 색상으로 선두께 15짜리 그림을 그린 후 저장한다.	Pass
7	1.6.2.3.5	브러시 노말을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 10짜리 그림을 그린 후 저장한다.	Pass
8	1.7.2.1.5	브러시 S1을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 1짜리 그림을 그린 후 저장한다.	Pass
9	2.6.1.2.5	기존의 그림을 불러와서 브러시 노말을 선택하고 default 색상으로 선두께 5짜리 그림을 그린다.	Pass
10	2.6.3.1.5	기존의 그림을 불러와서 브러시 노말을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 그린다.	Pass
11	2.4.1.4.5	기존의 그림을 불러와서 도형-원을 선택하고 default 색상으로 선두께 15짜리 그림을 그린다.	Pass
12	1.4.3.5.5	도형-원을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 그리고 저장한다.	Pass
13	2.5.2.1.5	기존의 그림을 불러와서 도형-사각형을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 10짜리 그림을 그린다.	Pass
14	1.7.1.3.5	브러시 S1을 선택하고 default 색상을 선택한 후 선두께 10짜리 그림을 그린 후 저장한다.	Pass
15	1.3.1.4.5	지우개를 선택하고 선두께 15짜리로 그림을 지우고 저장한다.	Pass
16	1.4.3.3.5	도형-원을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 그린 후 저장한다.	Pass
17	1.7.2.4.5	브러시 S1을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 15짜리 그림을 그린 후 저장한다.	Pass
18	2.7.1.5.5	기존의 그림을 불러와서 브러시 S1을 선택하고 default 색상으로 선두께 20짜리 그림을 그린다.	Pass
19	1.8.3.3.5	브러시 S2를 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 그리고 저장한다.	Pass
20	2.8.1.1.5	기존의 그림을 불러와서 브러시 S2를 선택하고 default 색상으로 선두께 1짜리 그림을 그린다.	Pass
21	1.3.1.2.5	지우개를 선택하고 선두께 5짜리로 그림을 지우고 저장한다.	Pass
22	2.9.1.3.5	기존의 그림을 불러와서 선을 선택하고 default 색상으로 선두께 10짜리 그림을 그린다.	Pass
23	2.4.2.2.5	기존의 그림을 불러와서 도형-원을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 5짜리 그림을 그린다.	Pass

Pairwise Combination Testing – Test Cases

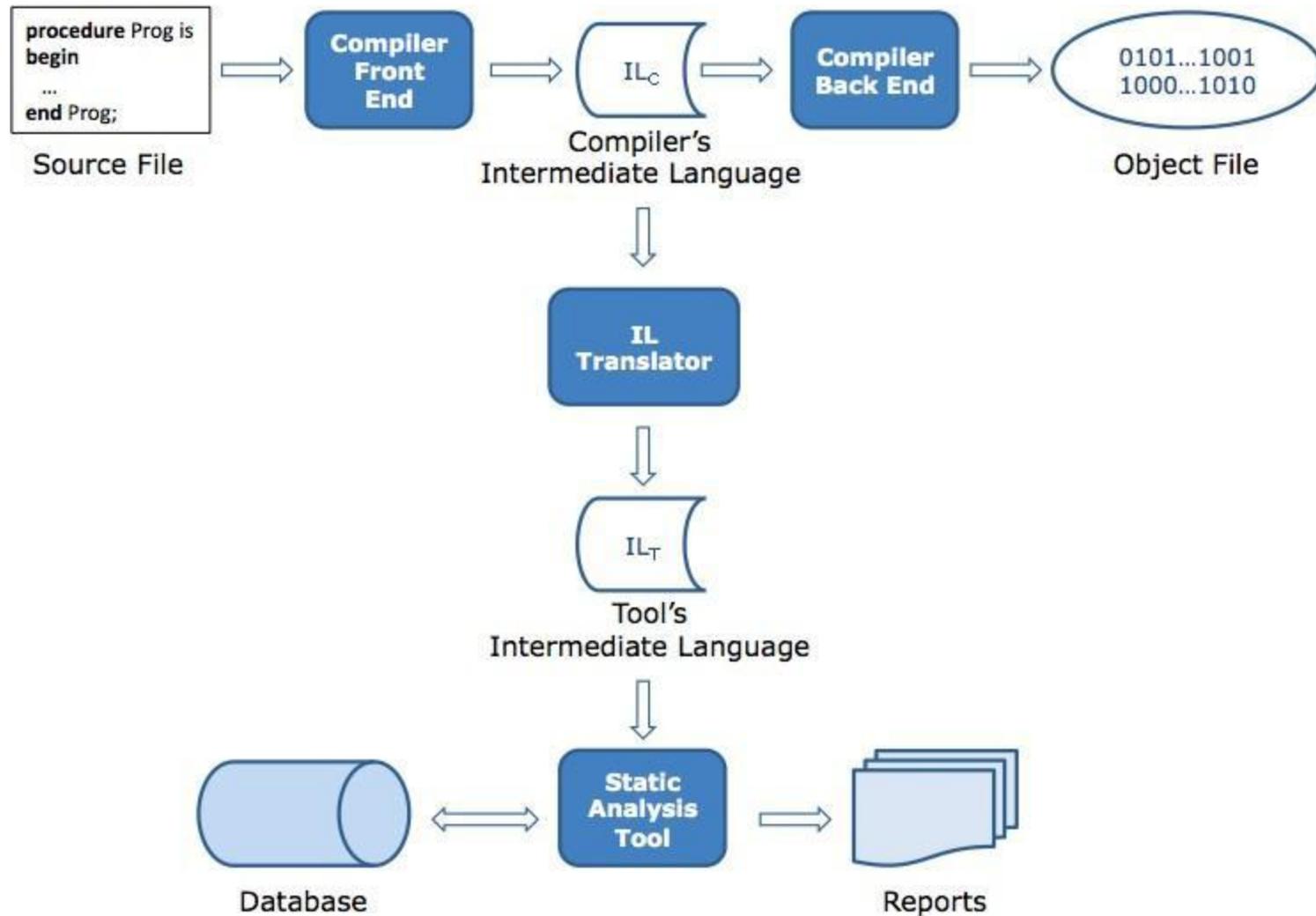
No.	Test Case No.	설명	결과
24	2.8.1.4.5	기존의 그림을 불러와서 브러시 S2를 선택하고 default 색상으로 선두께 15짜리 그림을 그린다.	Pass
25	2.9.3.2.5	기존의 그림을 불러와서 선을 선택하고 색선택으로 선택한 색상으로 선두께 5짜리 그림을 그린다.	Pass
26	1.5.1.4.5	도형-사각형을 선택하고 default 색상으로 선두께 15짜리 그림을 그리고 저장한다.	Pass
27	1.3.1.5.5	지우개를 선택하고 선두께 20짜리로 그림을 지우고 저장한다.	Pass
28	2.4.2.1.5	기존의 그림을 불러와서 도형-원을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 1짜리 그림을 그린다.	Pass
29	2.9.1.1.5	기존의 그림을 불러와서 선을 선택하고 default 색상으로 선두께 1짜리 그림을 그린다.	Pass
30	1.6.3.5.5	브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 그리고 저장한다.	Pass
31	1.8.2.5.5	브러시S2를 선택하고 색편집으로 선택한 임의의 색상으로 선두께 20짜리 그림을 그리고 저장한다.	Pass
32	2.9.2.4.5	기존의 그림을 불러와서 선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 15짜리 그림을 그린다.	Pass
33	1.2.2.1.5	연필을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 1짜리 그림을 그리고 저장한다.	Pass
34	2.5.3.5.5	기존의 그림을 불러와서 도형-사각형을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 그린다.	Pass
35	2.2.1.1.5	기존의 그림을 불러와서 연필을 선택하고 default 색상으로 선두께 1짜리 그림을 그린다.	Pass
36	1.5.2.2.5	도형-사각형을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 5짜리 그림을 그린다.	Pass
37	2.1.2.1.5	기존의 그림을 불러와서 색채우기를 선택하고 색편집으로 선택한 임의의 색상으로 선두께 1짜리 그림을 그린다.	Pass
38	2.2.3.1.5	기존의 그림을 불러와서 연필을 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 그린다.	Pass
39	1.3.1.1.5	지우개를 선택하고 1짜리 두께로 그림을 지운 후 저장한다.	Pass
40	1.5.3.3.5	브러시-노말을 선택하고 색선택으로 선택한 색상으로 선두께 10짜리 그림을 그리고 저장한다.	Pass
41	1.1.3.1.5	색채우기를 선택하고 색선택으로 선택한 색상으로 선두께 1짜리 그림을 그리고 저장한다.	Pass
42	2.5.2.3.1	기존의 그림을 불러와서 도형-네모를 선택하고 색편집으로 선택한 임의의 색상으로 선두께 10짜리 그림을 그린 후 데칼코마니 효과를 적용한다.	Pass
43	2.9.2.4.2	기존의 그림을 불러와서 선을 선택하고 색편집으로 선택한 임의의 색상으로 선두께 15짜리 그림을 그린 후 점선으로 변경 효과를 적용한다.	Pass
44	2.8.1.1.3	기존의 그림을 불러와서 브러시 special2을 선택하고 default 색상으로 선두께 1짜리 그림을 그린 후 그림자 효과를 적용한다.	Pass
45	1.4.3.5.4	도형-원을 선택하고 색선택으로 선택한 색상으로 선두께 20짜리 그림을 그린 후 초기화 효과를 적용하여 저장한다.	Pass

Static Analysis

Static Analysis

- 어떤 프로그램을 분석할 때 그 프로그램을 실행시키지 않고 그 자체를 분석하는 것. 프로그램에 내재한 논리적 오류는 보통 프로그램을 실행하여 확인하지 않으면 찾기가 힘들지만, 정적 분석은 이러한 오류를 찾아내는 데 도움을 줄 수 있다.
- inspection, walkthrough와 같은 수동적인 방법과, 정적 분석 도구를 활용하는 방법이 있다.

Static Analysis Procedure



Static Analysis Type

분류	구분	내용
코드 유형	규칙준수	프로그램 개발 언어별 코딩표준 및 주석 규칙 준수 여부 검사
	중복코드	프로그램 내에 중복으로 사용된 소스코드가 있는지 검사
	복잡성	소스코드의 내부에 if문과 같은 분기문 등의 복잡도를 측정 (Cyclomatic Complexity)
결함 유형	Memory leak	프로그램 코드 내에서 메모리 사용이 완료된 후 해제가 되지 않고 계속 점유하여 발생하는 문제 등
	Buffer Overflow / Buffer Overrun	프로그램 코드 내에서 메모리를 다루는 중 오류 발생하여 잘못된 동작하는 문제
	NullPointerException	프로그램 코드 내에서 객체를 선언하지 않거나, 해제 후에 사용하려고 할 때 발생하는 문제

Static Analysis Tools

제품명	설명
SourceMonitor	프로그램 소스 복잡도 분석 도구로 다양한 언어(C,C++,Java 등) 지원 http://www.campwoodsw.com/sourcemonitor.html
PMD	자바 프로그램 언어에 대한 소스 코드 검사 도구 http://pmd.sourceforge.net/
FindBugs	자바 프로그램 언어에 대한 소스 코드 오류 분석 도구 http://findbugs.sourceforge.net/
CppCheck	C/C++ 프로그램 언어에 대한 오류 분석 도구 http://cppcheck.sourceforge.net/
Sonar	다양한 언어(C/C++, Java, PHP 등) 소스코드 분석 도구이며, 플러그인을 추가하여 통합 분석 가능 http://www.sonarsource.org/
CheckStyle	자바 프로그램에 대한 코딩 표준 준수 여부 검사 도구 http://checkstyle.sourceforge.net/
N'SIQ CppStyle	C/C++ 프로그램 언어에 대한 코딩 표준 준수 여부 검사 도구 http://dev.naver.com/projects/nsiqcppstyle/

Clover

Clover

- 총 206개의 Test case들을 모두 실행해보고 난 후 Code coverage 확인

The screenshot displays the Clover integration in an IDE. The top part shows the source code for `DrawingObject.java` with line numbers 80-87. The bottom part shows the 'Coverage Explorer' window with a table of metrics for 'Application classes'.

Elem	Cov%	Av Me Cpx	Cpx
modling	83.9%	2.7	530.0
(default package)	83.9%	2.7	530.0
DrawEllipse.java	78.6%	1.3	18.0
DrawingObject.java	89.9%	1.3	25.0
DrawLine.java	78.6%	1.3	18.0
DrawRectangle.java	78.6%	1.3	18.0
DrawShape.java	0.0%	-	0.0
DrawTool.java	84.5%	3.3	247.0
FileIO.java	78.9%	3.6	18.0
Screen.java	96.2%	1.8	79.0
SpecialEffect.java	72.2%	8.9	107.0

Metrics for: DrawingObject.java

Structure	Test Executions
Packages: -	Executed Tests: 0
Files: -	Passes: 0
Classes: 1	Fails: 0
Methods: 19	Errors: 0
Statements: 48	
Branches: 12	

Source

LOC:	116	NC LOC:	95
Total Cmp:	25	Cmp Density:	0.!
Avg Method Cmp:	1.3		

Clover

- 총 206개의 Test case들을 모두 실행해보고 난 후 Code coverage 확인
 - Interface Class의 경우 회색
 - 선정된 206개의 Test case별로 모든 기능들을 수행하여 테스트해보았지만 100%의 Code coverage를 달성할 수 없었음.
 - 수행될 수 없는 예외처리 사항들에 대한 코드들이 포함되었을 것이라고 예상.
 - 시스템 테스팅만을 수행한 것이므로 코드에 대한 내용을 분석하지 못해서 구체적으로 어떤 기능에 대한 부분이 cover되지 않았는지 알 수가 없었음.

```
626 public double getsx(){ return sx; }
280 public double getsy(){ return sy; }
453 public double getex(){ return ex; }
280 public double getey(){ return ey; }

0 public void setisex(){
0     this.isex = 1;
0 }
0 public int getisex() {
0     return isex;
0 }

7573 public double getx(int index){
7573     return x[index];
7573 }
6874 public double gety(int index){
6874     return y[index];
```

Clover

- Treemap Report

Refresh

Console Error Log Problems Javadoc Declaration Properties LogCat Coverage Explo... Test Run Explorer Test Contributi... Clover Dashboa...

Show: Application classes

Elem	Cov%	Av Me Cpx	Cpx
modling	83.9%	2.7	530.0
(default package)	83.9%	2.7	530.0
DrawEllipse.java	78.6%	1.3	18.0
DrawingObject.java	89.9%	1.3	25.0
DrawLine.java	78.6%	1.3	18.0
DrawRectangle.java	78.6%	1.3	18.0
DrawShape.java	0.0%	-	0.0
DrawTool.java	84.5%	3.3	247.0
FileIO.java	78.9%	3.6	18.0
Screen.java	96.2%	1.8	79.0
SpecialEffect.java	72.2%	8.9	107.0

Metrics for: DrawLine.java

Structure

- Packages: -
- Files: -
- Classes: 1
- Methods: 14
- Statements: 34
- Branches: 8

Test Executions

- Executed Tests: 0
- Passes: 0
- Fails: 0
- Errors: 0

Source

- LOC: 88
- Total Cmp: 18
- Avg Method Cmp: 1.3
- NC LOC: 80
- Cmp Density: 0.:

Clover

- Cloud Report

The screenshot displays the Clover Cloud Report interface. At the top, a breadcrumb trail shows the project structure: DrawEllipse, DrawLine, DrawRectangle, DrawTool, DrawingObject, FileIO, Screen, and SpecialEffect. Below this, there are navigation tabs for 'Project risks' and 'Quick wins', a 'Project modling' dropdown, a checkbox for 'Include classes from sub-packages', and a 'Refresh' button.

The main content area is divided into two sections. On the left, a tree view shows the project structure with a 'Show: Application classes' dropdown. The right section, titled 'Metrics for: DrawLine.java', provides a detailed table of coverage metrics for each class in the project.

Elem	Cov%	Av Me Cpx	Cpx
modling	83.9%	2.7	530.0
(default package)	83.9%	2.7	530.0
DrawEllipse.java	78.6%	1.3	18.0
DrawingObject.java	89.9%	1.3	25.0
DrawLine.java	78.6%	1.3	18.0
DrawRectangle.java	78.6%	1.3	18.0
DrawShape.java	0.0%	-	0.0
DrawTool.java	84.5%	3.3	247.0
FileIO.java	78.9%	3.6	18.0
Screen.java	96.2%	1.8	79.0
SpecialEffect.java	72.2%	8.9	107.0

Additional metrics for DrawLine.java are shown on the right:

- Structure:** Packages: -, Files: -, Classes: 1, Methods: 14, Statements: 34, Branches: 8
- Test Executions:** Executed Tests: 0, Passes: 0, Fails: 0, Errors: 0
- Source:** LOC: 88, Total Cmp: 18, Avq Method Cmp: 1.3, NC LOC: 80, Cmp Density: 0.5

Clover

- Clover Dashboard

The screenshot displays the Clover Dashboard within an IDE. The dashboard is organized into several sections:

- Coverage:** Shows 10 classes with 1,674 / 1,996 elements covered, resulting in 83.9% coverage. A progress bar indicates this percentage.
- Test Results:** Shows 0 / 0 tests completed in 0 seconds.
- Most Complex Packages:** Lists the package 'default-pkg' with 530 elements and 83.9% coverage.
- Most Complex Classes:** Lists the top 5 most complex classes:
 1. 84.3% DrawTool (244)
 2. 72.2% SpecialEffect (107)
 3. 96.2% Screen (79)
 4. 89.9% DrawingObject (25)
 5. 78.6% DrawEllipse (18)
- Top 8 Project Risks:** Lists the top 8 risks: SpecialEffect, FileIO, DrawRectangle, DrawTool, DrawLine, DrawingObject, Screen, and DrawEllipse.
- Least Tested Methods:** Lists the top 4 least tested methods:
 1. 0% DrawTool.filldecalcolor(int,int,int,int,Color) : BufferedImage (13)
 2. 0% Screen.r() : void (1)
 3. 0% SpecialEffect.getdecalflag() : int (1)
 4. 0% SpecialEffect.getlighton() : boolean (1)

Clover

- Coverage Report(PDF)

coverage.pdf - Adobe Reader

파일(F) 편집(E) 보기(V) 창(W) 도움말(H)

1 / 1 143%

도구 서명 주

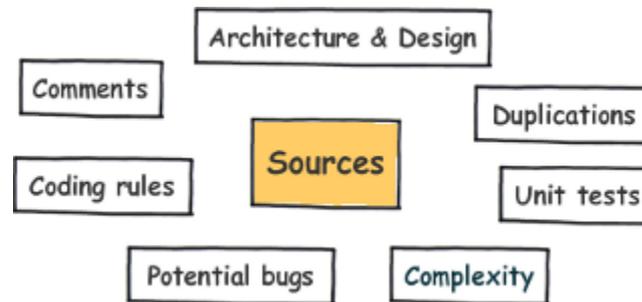
Clover Coverage Report		project stats:			
modling Coverage Report		LOC:	2,724	Methods:	197
Coverage timestamp: 6 6 2013 18:57:15 KST		NCLOC:	2,262	Classes:	10
		Files:	9	Pkgs:	1

	Branch	Stmt	Method	Total
Clover database 6 6 2013 17:04:49 KST	71.8%	87.6%	84.8%	83.9% 
Packages	Branch	Stmt	Method	Total
default-pkg	71.8%	87.6%	84.8%	83.9% 

Sonar

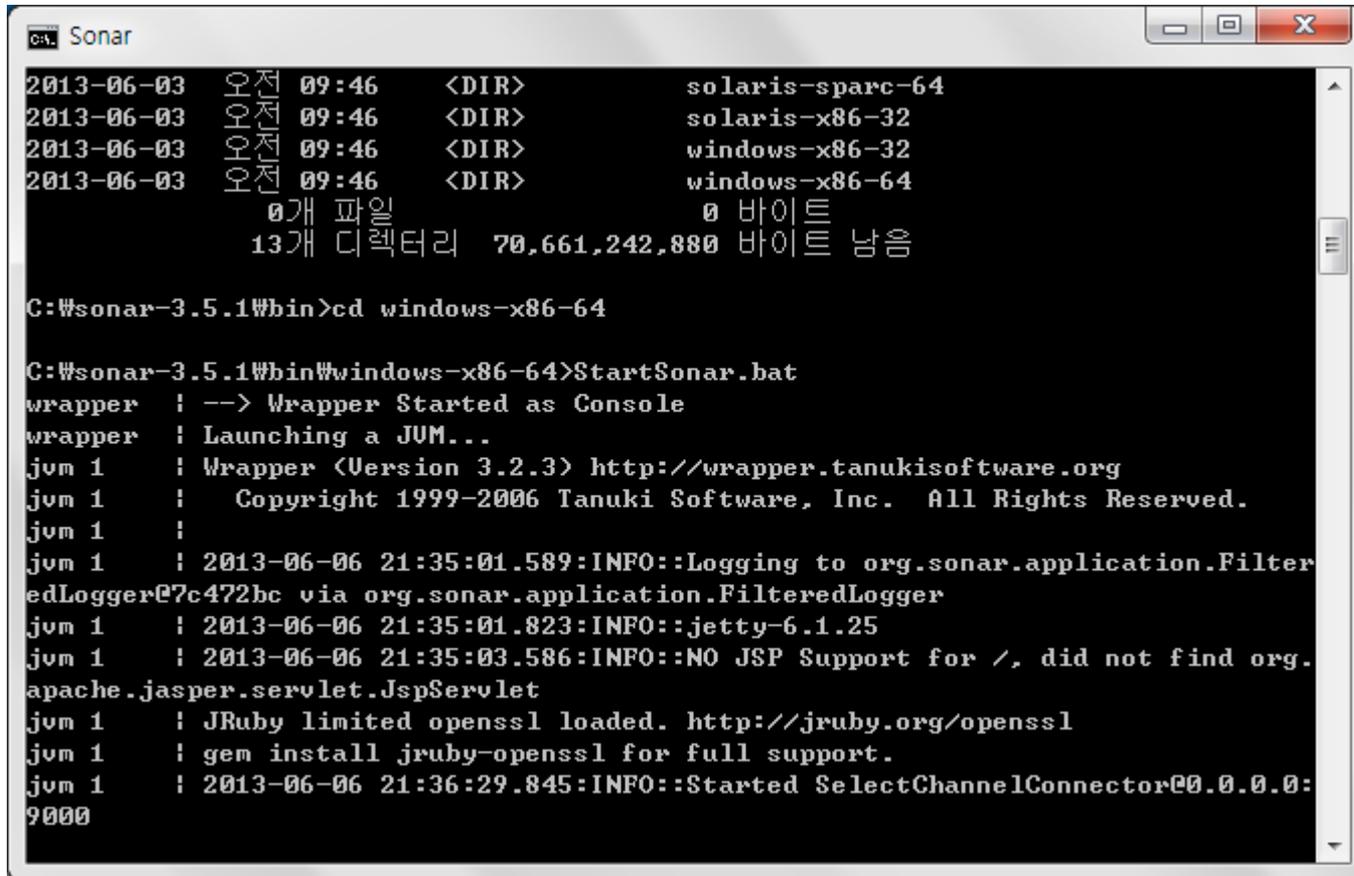
Sonar

- Sonar는 정적분석 도구로 소스코드에 대한 전반적인 품질을 확보할 수 있도록 정보를 제공하는 통합 플랫폼
- Server / Client 구조
- C/C++/Java 등 15개 이상의 다양한 프로그램 언어 지원
- 플러그인 설치로 다양한 도구와 유연한 통합
- 웹 기반 애플리케이션으로 다양한 결과를 서버에서 통합 관리 용이



Sonar

- Sonar Server 구동



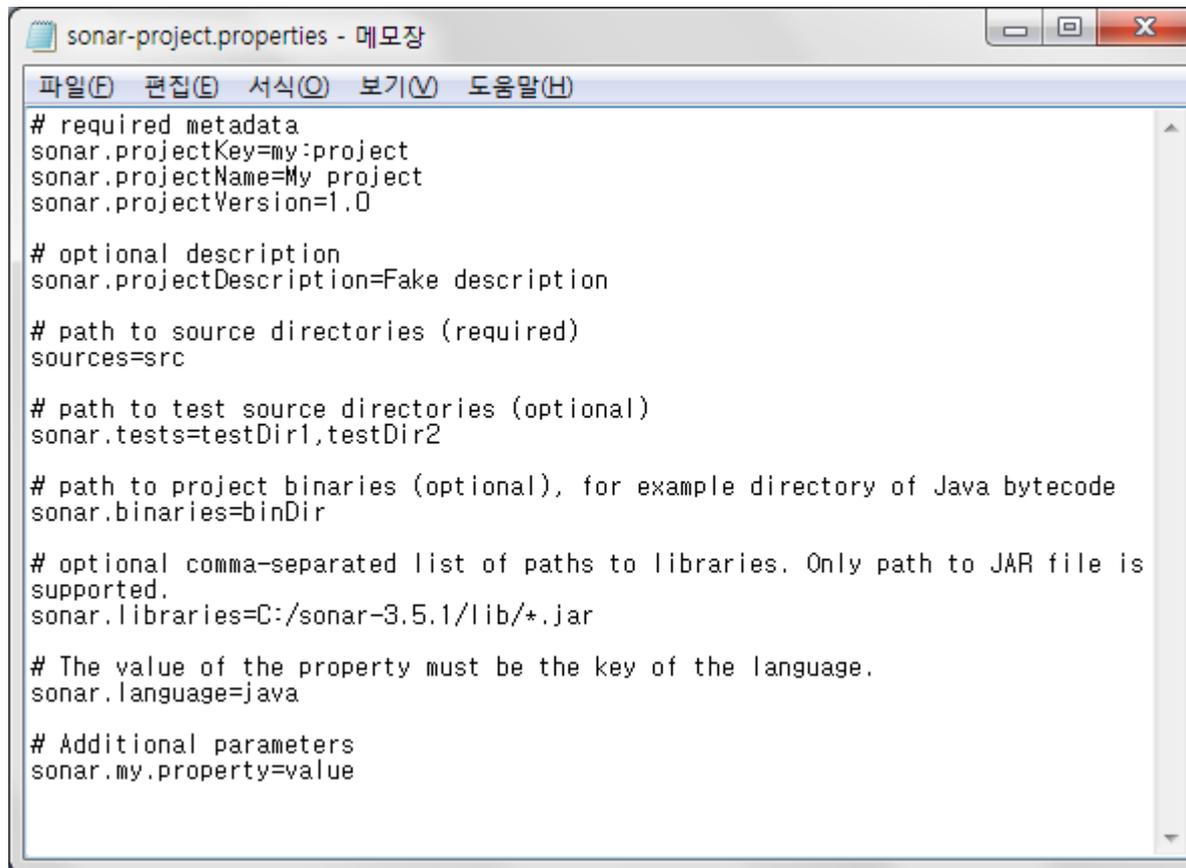
```
ca. Sonar
2013-06-03 오전 09:46 <DIR> solaris-sparc-64
2013-06-03 오전 09:46 <DIR> solaris-x86-32
2013-06-03 오전 09:46 <DIR> windows-x86-32
2013-06-03 오전 09:46 <DIR> windows-x86-64
      0개 파일      0 바이트
      13개 디렉터리 70,661,242,880 바이트 남음

C:\sonar-3.5.1\bin>cd windows-x86-64

C:\sonar-3.5.1\bin\windows-x86-64>StartSonar.bat
wrapper | --> Wrapper Started as Console
wrapper | Launching a JVM...
jvm 1 | Wrapper (Version 3.2.3) http://wrapper.tanukisoftware.org
jvm 1 | Copyright 1999-2006 Tanuki Software, Inc. All Rights Reserved.
jvm 1 |
jvm 1 | 2013-06-06 21:35:01.589:INFO::Logging to org.sonar.application.Filtered
edLogger@7c472bc via org.sonar.application.FilteredLogger
jvm 1 | 2013-06-06 21:35:01.823:INFO::jetty-6.1.25
jvm 1 | 2013-06-06 21:35:03.586:INFO::NO JSP Support for /, did not find org.
apache.jasper.servlet.JspServlet
jvm 1 | JRuby limited openssl loaded. http://jruby.org/openssl
jvm 1 | gem install jruby-openssl for full support.
jvm 1 | 2013-06-06 21:36:29.845:INFO::Started SelectChannelConnector@0.0.0.0:
9000
```

Sonar

- 프로젝트 최상위 경로에 sonar-project.properties 파일 작성



```
sonar-project.properties - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
# required metadata
sonar.projectKey=my:project
sonar.projectName=My project
sonar.projectVersion=1.0

# optional description
sonar.projectDescription=Fake description

# path to source directories (required)
sources=src

# path to test source directories (optional)
sonar.tests=testDir1,testDir2

# path to project binaries (optional), for example directory of Java bytecode
sonar.binaries=binDir

# optional comma-separated list of paths to libraries. Only path to JAR file is
supported.
sonar.libraries=C:/sonar-3.5.1/lib/*.jar

# The value of the property must be the key of the language.
sonar.language=java

# Additional parameters
sonar.my.property=value
```

Sonar

■ Sonar Runner 실행화면

```

C:\Windows\system32\cmd.exe
C:\Users\JJS\Desktop\modling>sonar
C:\Users\JJS\Desktop\modling>sonar-runner-1.3
Runner configuration file: C:\Users\JJS\Desktop\modling\sonar-runner-1.3
Project configuration file: C:\Users\JJS\Desktop\modling\sonar-runner-1.3
Runner version: 1.3
Java version: 1.7.0_17, vendor: Oracle Corporation
OS name: "Windows 7", version: "6.0.6002", arch: "amd64"
Server: http://localhost:9000
Work directory: C:\Users\JJS\Desktop\modling\sonar-runner-1.3\work
21:38:35.160 INFO   .s.b.b.BatchSet
21:38:35.660 INFO   o.s.h.c.File
21:38:35.784 INFO   atchPluginRepos
21:38:38.171 INFO   .s.b.b.TaskCon
21:38:39.372 INFO   .b.h.JdbcDriver
21:38:39.388 INFO   .b.ProjectExcL
21:38:39.388 WARN   .c.p.DefaultDat
ion purpose only
21:38:39.388 INFO   o.s.c.p.Dat
://localhost/sonar
21:38:39.840 INFO   actDatabaseConn
21:38:46.268 INFO   .s.b.s.ScanCont
21:38:46.283 INFO   .b.b.ProjectSet
21:38:48.140 INFO   .s.b.ProfilePro
uage=java1
21:38:48.202 INFO   s.f.ExclusionF
21:38:48.202 INFO   s.f.ExclusionF
21:38:48.280 INFO   nPluginsConfig
21:38:48.654 INFO   org.sonar
03)
21:38:48.717 INFO   org.sonar
sis of 2013-06-03 13:01:03.486)
21:38:48.764 INFO   org.sonar
ysis of 2013-06-03 13:01:03.486)
21:38:48.982 INFO   s.f.FileSystem
21:38:48.982 INFO   s.f.FileSystem
ng\sonar
21:38:48.982 INFO   s.f.FileSystem
ng\src
21:38:48.982 INFO   s.f.FileSystem
ng\binDir
21:38:48.982 INFO   s.f.FileSystem
: ko_KR
21:38:49.029 INFO   p.PhasesTimeProfiler
21:38:49.419 INFO   p.PhasesTimeProfiler
21:38:49.434 INFO   p.PhasesTimeProfiler
21:38:49.746 INFO   o.s.java.JavaSquid
21:38:51.572 INFO   o.s.java.JavaSquid
21:38:51.696 INFO   p.PhasesTimeProfiler
21:38:51.696 INFO   p.PhasesTimeProfiler
21:38:51.696 INFO   s.p.s.SurefireSensor
onar\build\surefire-reports
21:38:51.696 INFO   p.PhasesTimeProfiler
21:38:51.696 INFO   p.PhasesTimeProfiler
21:38:51.696 INFO   o.s.p.cpd.CpdSensor
21:38:51.696 INFO   s.p.c.i.IndexFactory
21:38:52.180 INFO   p.PhasesTimeProfiler
21:38:52.180 INFO   p.PhasesTimeProfiler
21:38:52.180 INFO   org.sonar.INFO
21:38:52.242 INFO   ckstyleConfiguration
Desktop\modling\sonar\checkstyle.xml
21:38:54.816 INFO   org.sonar.INFO
21:38:54.832 INFO   p.PhasesTimeProfiler
21:38:54.832 INFO   p.PhasesTimeProfiler
21:38:54.832 INFO   org.sonar.INFO
21:38:54.848 INFO   o.s.p.p.PmdTemplate
21:38:54.894 INFO   p.p.PmdConfiguration
modling\sonar\pmd.xml
21:38:59.637 INFO   org.sonar.INFO
21:38:59.730 INFO   p.PhasesTimeProfiler
21:38:59.730 INFO   p.PhasesTimeProfiler
21:38:59.730 INFO   p.PhasesTimeProfiler
21:39:00.183 INFO   p.PhasesTimeProfiler
21:39:00.183 INFO   p.PhasesTimeProfiler
21:39:00.230 INFO   p.PhasesTimeProfiler
21:39:00.230 INFO   p.PhasesTimeProfiler
21:39:00.245 INFO   p.PhasesTimeProfiler
21:39:00.245 INFO   p.PhasesTimeProfiler
21:39:00.261 INFO   p.PhasesTimeProfiler
21:39:00.261 INFO   p.PhasesTimeProfiler
21:39:00.276 INFO   o.s.p.j.JaCoCoPlugin
output directory does not exist: C:\Users\
21:39:00.276 INFO   p.PhasesTimeProfiler
21:39:00.822 INFO   p.PhasesTimeProfiler
21:39:06.267 INFO   s.c.c.ScanGraphStore - Persist graphs of components
21:39:06.360 INFO   .b.p.UpdateStatusJob - ANALYSIS SUCCESSFUL, you can browse ht
tp://localhost:9000
21:39:06.360 INFO   b.p.PostJobsExecutor - Executing post-job class org.sonar.plu
gins.core.batch.IndexProjectPostJob
21:39:06.438 INFO   b.p.PostJobsExecutor - Executing post-job class org.sonar.plu
gins.dbcleaner.ProjectPurgePostJob
21:39:06.454 INFO   .p.d.p.KeepOneFilter - -> Keep one snapshot per day between 2
013-05-09 and 2013-06-05
21:39:06.454 INFO   .p.d.p.KeepOneFilter - -> Keep one snapshot per week between
2012-06-07 and 2013-05-09
21:39:06.454 INFO   .p.d.p.KeepOneFilter - -> Keep one snapshot per month between
2008-06-12 and 2012-06-07
21:39:06.454 INFO   .d.p.DeleteAllFilter - -> Delete data prior to: 2008-06-12
21:39:06.470 INFO   o.s.c.purge.PurgeDao - -> Clean My project [id=1]
21:39:06.470 INFO   o.s.c.purge.PurgeDao - <- Clean snapshot 8
Total time: 36.894s
Final Memory: 15M/367M
C:\Users\JJS\Desktop\modling>

```

Sonar

- Sonar Dashboard

The screenshot displays the Sonar Dashboard for a project named 'My project'. The browser address bar shows 'http://localhost:9000/dashboard/index/1'. The dashboard is divided into several sections:

- Project Information (Highlighted):** A red box highlights the 'Lines of code' (2,352) and 'Classes' (11) section. The text '프로젝트의 기본적인 정보' (Basic information of the project) is written in red above this section.
- Violations:** Shows 1,007 total violations with a 42.9% rules compliance rate. A legend indicates the severity distribution: 0 Blocker, 0 Critical, 179 Major, 807 Minor, and 21 Info.
- Hotspots by Lines of code:** A table lists hotspots: DrawTool (902), Screen (524), SpecialEffect (402), DrawingObject (95), and SpecialEffectTest (90).
- Comments:** 7.9% (203 lines, 0.0% docu. API, 154 undocu. API).
- Duplications:** 23.1% (659 lines, 30 blocks, 6 files).
- Complexity:** 4.1 /method, 47.5 /class, 52.3 /file. Total: 523. A bar chart shows complexity distribution for Methods and Files.
- Most violated rules:** Magic Number (671), Trailing Comment (126), If Else Stmts Must Use Braces (30), Visibility Modifier (24), and Avoid Duplicate Literals (16).
- Most violated resources:** A table showing violation counts for DrawTool, Screen, SpecialEffect, FileIO, and SpecialEffectTest across various rule categories.
- Unit tests coverage:** 0 tests.

Sonar

- Sonar Dashboard

No coding standards and Potential bugs

The screenshot shows the Sonar Dashboard for a project named 'My project'. The browser address bar indicates the URL is `http://localhost:9000/dashboard/index/1`. The dashboard is divided into several sections:

- Summary Metrics:**
 - Lines of code:** 2,352 (2,857 lines, 1,382 statements, 10 files)
 - Classes:** 11 (1 package, 127 methods, 36 accessors)
 - Violations:** 1,007 (Rules compliance: 42.9%)
 - Hotspots by Lines of code:** DrawTool (902), Screen (524), SpecialEffect (402), DrawingObject (95), SpecialEffectTest (90)
 - Comments:** 7.9% (203 lines, 0.0% docu. API, 154 undocu. API)
 - Duplications:** 23.1% (659 lines, 30 blocks, 6 files)
 - Complexity:** 4.1 /method, 47.5 /class, 52.3 /file (Total: 523)
- Violations Breakdown:**
 - Blocker: 0
 - Critical: 0
 - Major: 179
 - Minor: 807
 - Info: 21
- Most violated rules:**
 - Magic Number: 671
 - Trailing Comment: 126
 - If Else Stmt Must Use Braces: 30
 - Visibility Modifier: 24
 - Avoid Duplicate Literals: 16
- Most violated resources:**
 - DrawTool: 0 Blocker, 0 Critical, 73 Major, 457 Minor, 4 Info
 - Screen: 0 Blocker, 0 Critical, 44 Major, 152 Minor, 0 Info
 - SpecialEffect: 0 Blocker, 0 Critical, 34 Major, 138 Minor, 4 Info
 - FileIO: 0 Blocker, 0 Critical, 20 Major, 6 Minor, 0 Info
 - SpecialEffectTest: 0 Blocker, 0 Critical, 5 Major, 31 Minor, 2 Info
- Unit tests coverage:** 0 tests
- Unit test success:** 0 tests

Sonar

- Sonar Dashboard

My project
DrawTool

Duplications Source Violations Raw

534 violations Blocker: 0 Critical: 0 Major: 73 Minor: 457 Info: 4

Full source | Time changes... | If Else Stmts Must Use Braces (16)

```
648     {
649         btnNewButton.setIcon(new ImageIcon("res\\select_rec.png"));
650     }
651     else
652         btnNewButton.setIcon(new ImageIcon("res\\rectan.jpg"));
```

If Else Stmts Must Use Braces | 3일

Avoid using if...else statements without curly braces

```
653
654     frame.getContentPane().add(btnNewButton);
655
656     JButton btnNewButton_1 = new JButton("");
```

```
667
668     if (figureShape=="circle") {
669         btnNewButton_1.setIcon(new ImageIcon("res\\select_circle.png"));
670     }
671     else
672         btnNewButton_1.setIcon(new ImageIcon("res\\circle.jpg"));
```

If Else Stmts Must Use Braces | 3일

Avoid using if...else statements without curly braces

```
672
673     frame.getContentPane().add(btnNewButton_1);
674
675     frame.addWindowListener(new WindowListener(){
```

```
748     }
749 });
750 btnNewButton.setBounds(0, 28, 120, 25);
751 if (lineThickness == 5)
```

Sonar

▪ Sonar Dashboard

The screenshot shows the Sonar Dashboard for a project named 'My project'. The browser address bar indicates the URL is `http://localhost:9000/dashboard/index/1`. The dashboard is divided into several sections:

- Summary Metrics:**
 - Lines of code: 2,352 (2,857 lines, 1,382 statements, 10 files)
 - Classes: 11 (1 package, 127 methods, 36 accessors)
 - Violations: 1,007 (Rules compliance: 42.9%)
 - Comments: 7.9% (203 lines, 0.0% docu. API, 154 undocu. API)
 - Duplications: 23.1% (659 lines, 30 blocks, 6 files)
 - Complexity: 4.1 /method, 47.5 /class, 52.3 /file (Total: 523)
- Hotspots by Lines of code:** A table listing hotspots with their line counts and bar charts. The top items are:

Hotspot	Lines of code
DrawTool	902
Screen	524
SpecialEffect	402
DrawingObject	95
SpecialEffectTest	90
- Most violated rules:** A list of rules with their violation counts and severity levels.

Rule	Count
Magic Number	671
Trailing Comment	126
If Else Stmts Must Use Braces	30
Visibility Modifier	24
Avoid Duplicate Literals	16
- Most violated resources:** A table showing the number of violations for each resource across different severity levels.

Resource	Blocker	Critical	Major	Minor	Info
DrawTool	0	0	73	457	4
Screen	0	0	44	152	0
SpecialEffect	0	0	34	138	4
FileIO	0	0	20	6	0
SpecialEffectTest	0	0	5	31	2
- Unit tests coverage:** Shows 0 tests passed.

클래스별 라인 수

Sonar

▪ Sonar Dashboard

The screenshot shows the Sonar Dashboard for a project named 'My project'. The browser address bar indicates the URL is `http://localhost:9000/dashboard/index/1`. The dashboard is divided into several sections:

- Summary Metrics:**
 - Lines of code:** 2,352 (2,857 lines, 1,382 statements, 10 files)
 - Classes:** 11 (1 package, 127 methods, 36 accessors)
 - Violations:** 1,007 (Rules compliance: 42.9%)
 - Hotspots:** 7 (Hotspots by Lines of code)
 - Comments:** 7.9% (203 lines, 0.0% docu. API, 154 undocu. API)
 - Duplications:** 23.1% (659 lines, 30 blocks, 6 files)
 - Complexity:** 4.1 /method, 47.5 /class, 52.3 /file (Total: 523)
- Hotspots by Lines of code:** A table listing hotspots: DrawTool (902), Screen (524), SpecialEffect (402), DrawingObject (95), and SpecialEffectTest (90).
- Most violated rules:** A list of rules with their counts: Magic Number (671), Trailing Comment (126), If Else Stmt Must Use Braces (30), Visibility Modifier (24), and Avoid Duplicate Literals (16).
- Most violated resources:** A table showing violations for resources: DrawTool, Screen, SpecialEffect, FileIO, and SpecialEffectTest.
- Unit tests coverage:** 0 tests.

A red box highlights the 'Comments' section, and red Korean text '주석 라인 수' (Number of comment lines) is overlaid on the 'Duplications' section.

Sonar

- Sonar Dashboard

Dashboard
Hotspots
Reviews
Time Machine

TOOLS
Components
Violations Drilldown
Libraries
Clouds
Compare

sonar

http://localhost:9000/drilldown/measures/1?metric= My project

Comments (%)

7.9%

[default] 7.9%

File	Comments (%)
DrawShape	0.0%
SpecialEffectTest	0.0%
SpecialEffect	6.1%
DrawTool	6.7%
DrawEllipse	8.0%
DrawLine	8.0%

My project
SpecialEffect

Duplications Source Violations Raw

Lines:	451	Statements:	294	Comments (%):	6.1%	Public documented API (%):	0.0%	Classes:	1
Lines of code:	402	Complexity:	108	Comment lines:	26	Public undocumented API:	8		
Methods:	7	Complexity /method:	15.4			Public API:	8		
Accessors:	5								

```

1 import java.awt.AWTException;
2 import java.awt.Color;
3 import java.awt.Rectangle;
4 import java.awt.Robot;
5 import java.awt.image.BufferedImage;
6 import java.util.ArrayList;
7
8
9 //
10 // Generated by StarUML(tm) Java Add-In
11 //
12 // @ Project : Untitled
13 // @ File Name : SpecialEffect.java
14 // @ Date : 2013-05-05
15 // @ Author :
16 //
17 //
18
19 public class SpecialEffect {
20     private int lightxy; //0000 00x0 g00
21     private boolean islighton = false; //0000 000
  
```

Sonar

- Sonar Dashboard

The screenshot shows the Sonar Dashboard for a project named 'My project'. The browser address bar indicates the URL is `http://localhost:9000/dashboard/index/1`. The dashboard is divided into several sections:

- Summary Metrics:**
 - Lines of code:** 2,352 (2,857 lines, 1,382 statements, 10 files)
 - Classes:** 11 (1 package, 127 methods, 36 accessors)
 - Violations:** 1,007 (Rules compliance: 42.9%)
 - Hotspots:** 11
 - Comments:** 7.9% (203 lines, 0.0% docu. API, 154 undocu. API)
 - Complexity:** 4.1 /method, 47.5 /class, 52.3 /file (Total: 523)
- Hotspots by Lines of code:** A table listing hotspots with their line counts:

DrawTool	902
Screen	524
SpecialEffect	402
DrawingObject	95
SpecialEffectTest	90
- Duplications:** 23.1% (659 lines, 30 blocks, 6 files). This section is highlighted with a red box and has the Korean text '중복 코드 라인 수' (Duplicate code line count) written over it.
- Most violated rules:**
 - Magic Number: 671
 - Trailing Comment: 126
 - If Else Stmt Must Use Braces: 30
 - Visibility Modifier: 24
 - Avoid Duplicate Literals: 16
- Most violated resources:** A table showing violations for different resources:

DrawTool	0	0	73	457	4
Screen	0	0	44	152	0
SpecialEffect	0	0	34	138	4
FileIO	0	0	20	6	0
SpecialEffectTest	0	0	5	31	2
- Unit tests coverage:** 0 tests

Sonar

- Sonar Dashboard

The screenshot displays the Sonar Dashboard interface. At the top, a browser window shows the URL `http://localhost:9000/drilldown/measures/1?highlight` and the project name 'My project'. The left sidebar contains navigation links: Dashboard, Hotspots, Reviews, Time Machine, TOOLS, Components, Violations Drilldown, Libraries, Clouds, and Compare. The main content area is titled 'Duplicated lines (%)' and shows a total of 23.1% duplicated lines, with a drilldown on 659 duplicated lines. A table lists the following categories and their counts:

Category	Count
DrawTool	417
DrawEllipse	59
Screen	52
DrawRectangle	48
SpecialEffect	46
DrawLine	37

Below this, the 'My project' section shows a detailed view for 'DrawTool' with a duplication rate of 38.2%. It indicates 1,092 total lines, 417 duplicated lines, and 13 duplicated blocks. A table lists the duplicated blocks:

Blocks	Nb Lines	From line	File	Details
2	73	122	DrawTool	DrawTool
	72	141	DrawTool	<pre> 122 case 1: 123 if (sx <= ex && sy <= ey) { //1 124 list.add(new DrawLine (sx+1, sy-1, ex+1, ey-1, color.black, 1)); 125 list.add(new DrawLine (sx-1, sy+1, ex-1, ey+1, color.black, 1)); 126 } 127 else if (sx>=ex && sy >= ey) { //2 </pre>
				Expand
2	73	224	DrawTool	DrawTool
	72	243	DrawTool	<pre> 224 case 1: 225 if (sx <= ex && sy <= ey) { //1 226 list.add(new DrawLine (sx+1, sy-1, ex+1, ey-1, color, 1)); 227 list.add(new DrawLine (sx-1, sy+1, ex-1, ey+1, color, 1)); 228 } 229 else if (sx>=ex && sy >= ey) { //2 </pre>
				Expand

Sonar

- Sonar Dashboard

0.0% docu. API
154 undocu. API

30 blocks
6 files

Complexity
4.1 /method
47.5 /class
52.3 /file
 Total: 523

● Methods ○ Files

Events: All

2013/06/06	Version	1.0
Fake description		
Key:	my:project	
Language:	Java	
Profile:	Sonar way (version 1)	
Alerts:	RSS Feed	

Bad distribution of complexity

DrawTool	↑	↓	▲	▼	↕
Screen	0	0	44	152	0
SpecialEffect	0	0	34	138	4
FileIO	0	0	20	6	0
SpecialEffectTest	0	0	5	31	2

Unit tests coverage: -

Unit test success: 0 tests

Size: Lines of code Color: Rules compliance 0.0% 100.0%

[default]

Sonar

- Sonar Dashboard

The screenshot shows the Sonar Dashboard for a project named "Sonar - My project". The browser address bar indicates the URL is `http://localhost:9000/dashboard/index/1`. The dashboard is divided into several sections:

- Summary Metrics:** 0.0% docu. API, 154 undocu. API, 30 blocks, 6 files.
- Complexity:** 4.1 /method, 47.5 /class, 52.3 /file, Total: 523. A bar chart shows complexity levels for methods and files.
- Events:** A table with columns for date (2013/06/06), version (1.0), and other details.
- Project Information:** Fake description, Key: my:project, Language: Java, Profile: Sonar way (version 1), Alerts: RSS Feed.
- Quality Gate:** A table listing quality gates with their status (pass/fail) and counts for errors, warnings, and failures.

Quality Gate	Errors	Warnings	Failures	Pass	Fail
DrawTool	0	0	73	457	4
Screen	0	0	44	152	0
SpecialEffect	0	0	34	138	4
FileIO	0	0	20	6	0
SpecialEffectTest	0	0	5	31	2
- Unit tests coverage:** 0 tests.
- Unit test success:** 0 tests.
- Treemap:** A treemap visualization showing the distribution of lines of code and rules compliance. The treemap is currently empty, showing only a "(default)" label. A legend indicates "Size: Lines of code" and "Color: Rules compliance 0.0% - 100.0%".

Sonar

- Sonar Dashboard

Sonar - Windows Internet Explorer

My project
 DrawTool

[Duplications](#) [Source](#) [Violations](#) [Raw](#)

Lines:	1,092	Statements:	508	Comments (%):	6.7%	Public documented API (%):	0.0%	Classes:	2
Lines of code:	902	Complexity:	245	Comment lines:	65	Public undocumented API:	40		
Methods:	26	Complexity /method:	9.4			Public API:	40		
Accessors:	6								

```

1  import java.awt.AWTException;
2  import java.awt.BasicStroke;
3  import java.awt.Color;
4  import java.awt.Graphics;
5  import java.awt.Graphics2D;
6  import java.awt.Panel;
7  import java.awt.Rectangle;
8  import java.awt.Robot;
9  import java.awt.Shape;
10 import java.awt.event.ActionEvent;
11 import java.awt.event.ActionListener;
12 import java.awt.event.AdjustmentEvent;
13 import java.awt.event.AdjustmentListener;
14 import java.awt.event.WindowEvent;
15 import java.awt.event.WindowListener;
16 import java.awt.image.BufferedImage;
17 import java.io.File;
18 import java.io.IOException;
19 import java.util.ArrayList;
20
21 import javax.imageio.ImageIO;
22 import javax.swing.ImageIcon;
23 import javax.swing.JButton;
24 import javax.swing.JComponent;
25 import javax.swing.JFrame;
26 import javax.swing.JPanel;
27 import javax.swing.JScrollBar;
28 import javax.swing.JTextPane;
29
30 //
31 //
32 //  Generated by StarUML(tm) Java Add-In
33 //
34 //  @ Project : Untitled
35 //  @ File Name : DrawTool.java
36 //  @ Date : 2013-05-05
37 //  @ Author :
38 //

```

PMD

What is PMD?

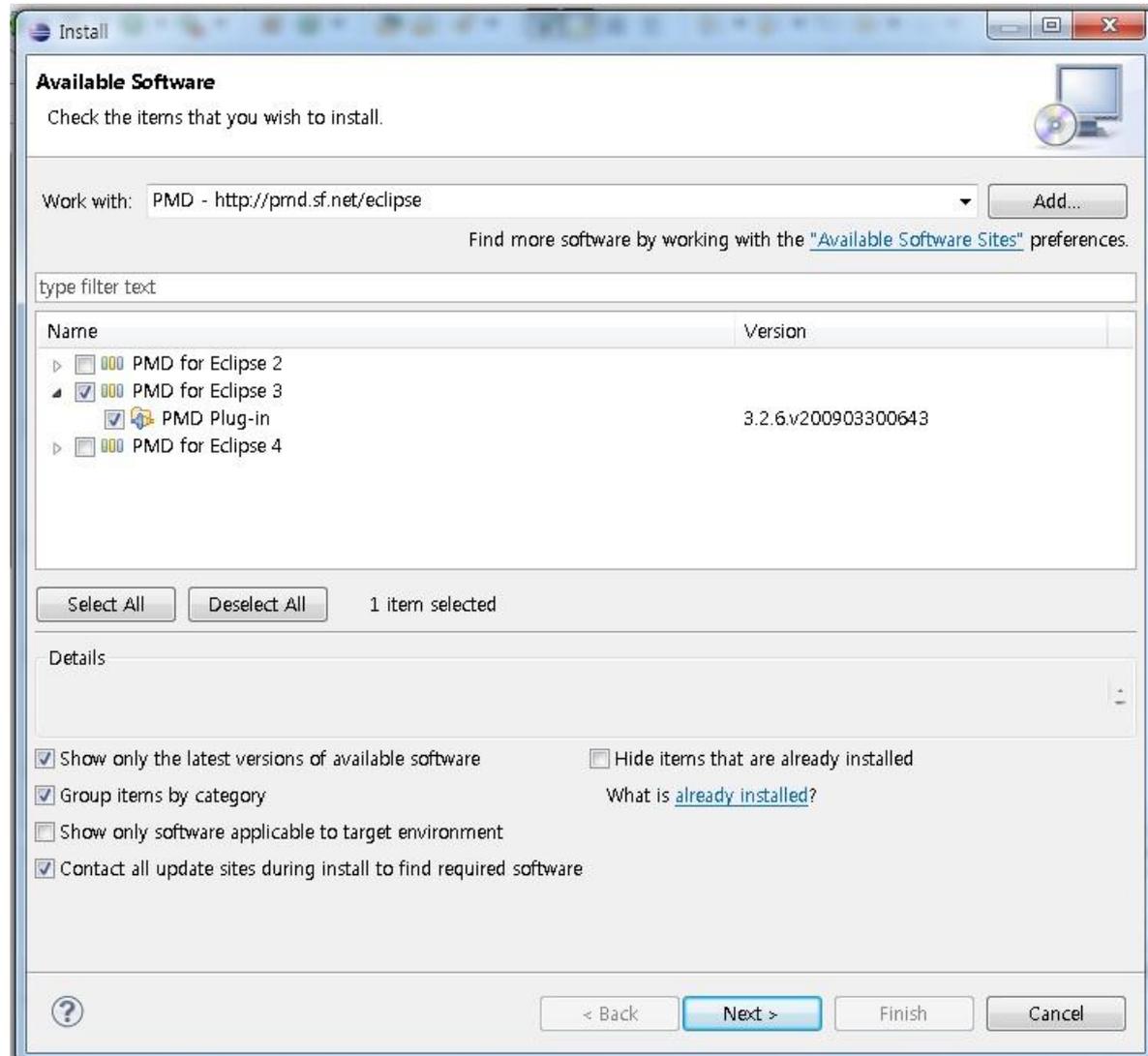
- 룰 기반의 정적 자바소스 코드 분석기
- CPD(Copy and Paste Detector)를 포함
- Java, JavaScript, XML, XSL 언어에 사용
- 이클립스 plug-in 형태로 제공
- 지정된 Rule set에 따라 코드를 검사하고 priority에 따라 경고
(High Error – Error – High Warning – Warning – Information)

Use PMD to find..

- 잠재적 버그
- 사용되지 않은 코드
(unused variable, empty catch block)
- 복잡한 코드
(Cyclomatic complexity 계산, 불필요한 객체 생성 탐지)
- 중복된 코드 (CPD)

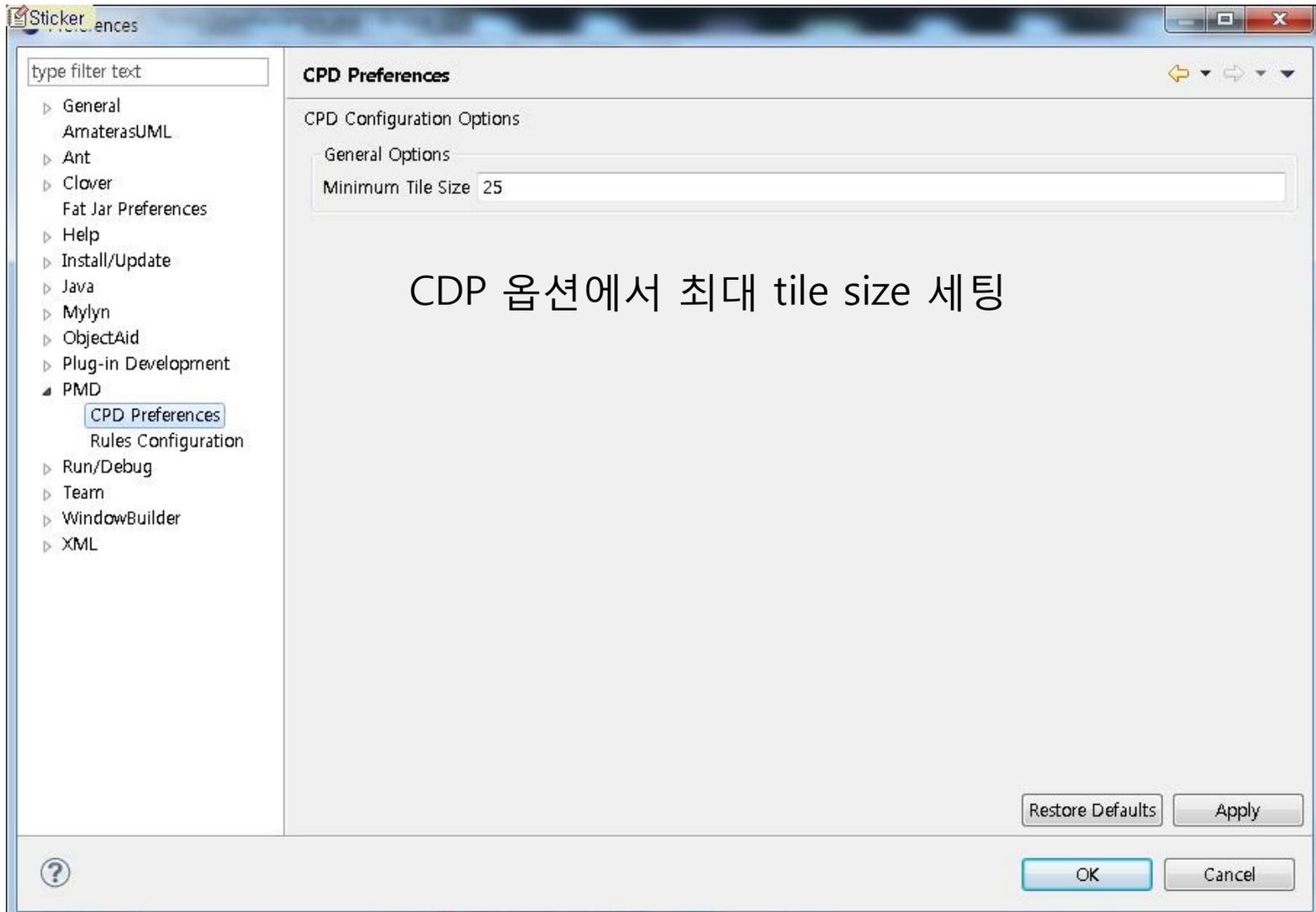
PMD in practice

Plug-in 설치



PMD in practice

Windows - Preference



PMD in practice

Windows - Preference

The screenshot shows the PMD Rules Configuration dialog box in an IDE. The dialog is titled "Rules Configuration" and contains a table of rules. The "Code Size Rules" rule is selected, and its configuration is shown in the "PMD Plugin" dialog box.

Rule set name	Rule name	Severity	Priority	Description
Design Rules	SwitchDensity	Warning	1.02	A high ratio of statements to labels in a switch stat...
Basic Rules	ForLoopShouldBeWhileLoop	Warning	1.02	Some for loops can be simplified to while loops...
Import Statemen...	ImportFromSamePackage	Warning	1.02	No need to...
Controversial Rul...	NullAssignment	Warning	1.02	Assigning...
Controversial Rul...	UnusedModifier	Warning	1.02	Fields in in...
Controversial Rul...	AssignmentInOperand	Warning	1.03	Avoid assign...
Code Size Rules	CyclomaticComplexity	Warning	1.03	Complexity...
Design Rules	ConstructorCallsOverridabl...	Error	1.04	Calling ove...
Design Rules	AccessorClassGeneration	Warning	1.04	Instantiatio...
Basic Rules	DoubleCheckedLocking	Error	1.04	Partially cre...
Coupling Rules	CouplingBetweenObjects	Warning	1.04	This rule co...
Coupling Rules	ExcessiveImports	Warning	1.04	A high num...
JUnit Rules	JUnitAssertionsShouldInclu...	Warning	1.04	JUnit assert...
Controversial Rul...	AtLeastOneConstructor	Warning	1.04	Each class...
Code Size Rules	ExcessivePublicCount	Warning	1.04	A large num...
Design Rules	SimplifyBooleanExpressions	Warning	1.05	Avoid unn...
Basic Rules	ReturnFromFinallyBlock	Warning	1.05	Avoid retur...
Design Rules	FinalFieldCouldBeStatic	Warning	1.1	If a final fi...

The "PMD Plugin" dialog box shows the following configuration for the "Code Size Rules" plugin:

- RuleSet name: Code Size Rules
- Rule Reference:
- Since: 1.03
- Rule name: CyclomaticComplexity
- XPath rule:
- Rule implementation class: net.sourceforge.pmd.rules.CyclomaticComplexity
- Message: The {0} "{1}" has a Cyclomatic Complexity of {2}.
- Priority: Warning high
- Uses Type Resolution:
- Uses DFA:
- Description: Complexity is determined by the number of decision points in a method plus one for the method entry. The decision points are 'if', 'while', 'for', and 'case labels'. Generally, 1-4 is low complexity, 5-7 indicates moderate complexity, 8-10 is high complexity.
- External Info URL: <http://pmd.sourceforge.net/rules/codesize.html#CyclomaticComplexity>
- Examples:


```
// Cyclomatic Complexity = 12
public class Foo {
    1 public void example() {
    2     if (a == b) {
```

PMD in practice

Rule Configuration Example

- Code Size Rule (ex: TooManyMethods) ->
 - 함수가 너무 많다는 것은 리팩토링의 고려 요소
 - 하지만 이러한 룰은 제약사항이 될 수 있다.
- Code Size Rule (ex: CyclomaticComplexity) ->
 - complexity는 하나의 method entry + method 안에서의 decision point 수. 이것을 찾아주는 룰
 - 1~4가 low complexity, 5~7 moderate complexity, 8~10 high complexity, 11+ very high complexity

PMD in practice

중복된 코드 확인 - CPD

The screenshot shows the Eclipse IDE with the PMD CPD (Copy-Paste Detector) view open. A red box highlights a code block in `SpecialEffect.java` (lines 220-234) that is identified as a suspect cut & paste. The CPD View at the bottom lists several instances of 'Found suspect cut & paste' across different files, including `DrawTool.java` and `SpecialEffect.java`.

음영 안의 코드 영역이 다른 클래스에서도 중복되어 사용되었다는 의미

Message	Class
Found suspect cut & paste (2 matches,35 lines)	
lines 926-960 in file DrawTool.java	src.DrawTool
lines 1016-1050 in file DrawTool.java	src.DrawTool
Found suspect cut & paste (2 matches,15 lines)	
lines 220-234 in file SpecialEffect.java	src.SpecialEffect
lines 242-256 in file SpecialEffect.java	src.SpecialEffect
Found suspect cut & paste (3 matches,47 lines)	
Found suspect cut & paste (2 matches,45 lines)	
Found suspect cut & paste (2 matches,45 lines)	
Found suspect cut & paste (2 matches,7 lines)	
Found suspect cut & paste (2 matches,12 lines)	

PMD in practice

Rule Based code Check

The screenshot shows the Eclipse IDE with the PMD plugin. The main editor displays the code for `SpecialEffect.java`. The `Violations Outline` window is open, showing a list of errors. The `Violations Overview` table is also visible, showing the total number of violations for the project.

# Violations	# Violations/LOC	# Violati...	Project
520	207.9 / 1000	3.25	modling

The `Violations Outline` window shows the following errors:

- All classes and interfaces must belong to...
- The class 'SpecialEffect' has a Cyclomatic Complexity of 10 (Highest = 38).
- Found non-transient, non-static member...
- Private field 'islighton' could be made fin...
- Found non-transient, non-static member...
- Avoid using implementation types like 'A...
- Avoid using implementation types like 'A...
- Found non-transient, non-static member...
- Avoid using implementation types like 'A...
- Avoid using implementation types like 'A...
- Found non-transient, non-static member...

소프트웨어 모델링 수업 3조 프로젝트에 적용 결과 520개의 violation 탐지.
(기본적으로 설정되어 있는 rule set들이 너무 많다.)

Rule에 기반한 violation들을 표시해준다.

PMD in practice

Rule Based code Check

The screenshot shows the Eclipse IDE with the PMD Plugin configuration dialog box open. The dialog is titled "PMD Plugin" and shows the following details for the "LocalVariableCouldBeFinal" rule:

- RuleSet name: Optimization Rules
- Since: 2.2
- Rule name: LocalVariableCouldBeFinal
- Rule implementation class: net.sourceforge.pmd.rules.optimization.LocalVariableCouldBeFinal
- Message: Local variable "{0}" could be declared final
- Priority: Warning high
- Description: A local variable assigned only once can be declared final.
- External Info URL: <http://pmd.sourceforge.net/rules/optimizations.html#LocalVariableCouldBeFinal>
- Examples:


```
public class Bar {
    public void foo () {
        String a = "a"; //if a will not be assigned again
        final String b = "b";
    }
}
```

The "Violations Outline" window shows a list of violations for the file "SpecialEffect.java":

Error Message	Line
Local variable 'height' could be declared final	44
Local variable 'height' could be declared final	36
Found non-transient, non-static member. Please mark as transi...	20
Found non-transient, non-static member. Please mark as transi...	26
Found non-transient, non-static member. Please mark as transi...	25
Found non-transient, non-static member. Please mark as transi...	27
Found non-transient, non-static member. Please mark as transi...	21
Found 'DU'-anomaly for variable 'ty' (lines '216'-'306').	216
Found 'DU'-anomaly for variable 'tx' (lines '215'-'306').	215
Found 'DU'-anomaly for variable 'tempy' (lines '238'-'306').	238
Found 'DU'-anomaly for variable 'tempy' (lines '212'-'306').	212

The "Violations" window shows a table with columns: s/LOC, # Violati..., and Project. The data row shows: 1000, 3.25, and modling.

Violation의 상세 정보를 확인할 수 있다.

PMD in practice

- 기본적으로 셋팅된 불필요한 rule들이 너무 많아 필요하다고 판단되는 몇 가지만 적용

1. Basic Rules

(ex) empty catch block과 같은 코드가 있을 경우 알려주는 Rule

2. Naming Rules

(ex)클래스 안에 생성자가 아닌 메소드에 그 클래스의 이름과 같은 이름으로 선언 시 warning을 발생해주는 naming과 관련된 Rule

3. Code size Rules

(ex)CyclomaticComplexity 확인, 너무 많은 메소드 사용 시 경고 등의 Rule

4. Optimization Rules

(ex)empty string을 더한 경우와 같은 rule

```
String s = "" + 123; // bad
```

```
String t = Integer.toString(456); // ok
```

PMD in practice

Rule Based code Check

새로운 rule 적용 후 60개의 violation 발견

Element	# Violations	# Violations/LOC	# Violations/...	Project
(default package)	60	23.1 / 1000	0.36	modling
DrawEllipse.java	1	12.8 / 1000	0.07	modling
DrawingObject.java	1	10.4 / 1000	0.05	modling
DrawLine.java	1	12.7 / 1000	0.07	modling
DrawRectangle.java	1	12.2 / 1000	0.07	modling
DrawShape.java	1	83.3 / 1000	0.09	modling
DrawTool.java	17	17.3 / 1000	0.59	modling
FileO.java	2	21.3 / 1000	0.40	modling
EmptyCatchBlock	1	10.6 / 1000	0.20	modling
NoPackage	1	10.6 / 1000	0.20	modling
FileO.java	2	21.3 / 1000	0.40	modling
Screen.java	15	26.3 / 1000	0.50	modling
ShortMethodName	1	1.8 / 1000	0.03	modling
TooManyMethods	1	1.8 / 1000	0.03	modling
TooManyFields	1	1.8 / 1000	0.03	modling
LongVariable	1	1.8 / 1000	0.03	modling
CyclomaticComplexity	2	3.5 / 1000	0.07	modling
NoPackage	1	1.8 / 1000	0.03	modling

Violations Overview

Console

Error Message

- This class has too many methods, consider refact
- This class has a bunch of public methods and att
- The method 'tempaddlist' has a Cyclomatic Corn
- The method 'spBrush2Cal' has a Cyclomatic Corr
- The method 'spBrush1Cal' has a Cyclomatic Corr
- The method 'addlist' has a Cyclomatic Complexit
- The class 'DrawTool' has a Cyclomatic Complexit
- Avoid really long methods.
- Avoid really long methods.
- Avoid really long classes.

PMD in practice

Rule Based code Check

▶ DrawTool.java	17	17.3 / 1000	0.59	modling
▶ FileIO.java	2	21.3 / 1000	0.40	modling
▶ EmptyCatchBlock	1	10.6 / 1000	0.20	modling
▶ NoPackage	1	10.6 / 1000	0.20	modling
▶ FileIO.java	2	21.3 / 1000	0.40	modling
▶ Screen.java	15	26.3 / 1000	0.50	modling
▶ ShortMethodName	1	1.8 / 1000	0.03	modling
▶ TooManyMethods	1	1.8 / 1000	0.03	modling
▶ TooManyFields	1	1.8 / 1000	0.03	modling
▶ LongVariable	1	1.8 / 1000	0.03	modling
▶ CyclomaticComplexity	2	3.5 / 1000	0.07	modling
▶ NoPackage	1	1.8 / 1000	0.03	modling

패키지 네임을 설정하지 않았거나 빈 CatchBlock 사용, 너무 많은 메소드 사용에 대한 경고, 설정한 CyclomaticComplexity 증가에 대한 경고를 해주고 있다. 이를 확인하고 수정하면서 더 안정적이고 효율적인 코딩 습관을 갖게 해주는 데 도움이 된다.

PMD in practice

Generate Reports

The screenshot displays a web browser window with a PMD report titled "PMD report" and "Problems found". The report lists 24 items with columns for #, File, Line, and Problem. The problems include issues like "All classes and interfaces must belong to a named package" and "Avoid really long classes".

#	File	Line	Problem
1	src/DrawEllipse.java	4	All classes and interfaces must belong to a named package
2	src/DrawLine.java	4	All classes and interfaces must belong to a named package
3	src/DrawRectangle.java	5	All classes and interfaces must belong to a named package
4	src/DrawShape.java	4	All classes and interfaces must belong to a named package
5	src/DrawTool.java	1	This class has a bunch of public methods and attributes
6	src/DrawTool.java	40	All classes and interfaces must belong to a named package
7	src/DrawTool.java	40	Avoid really long classes.
8	src/DrawTool.java	40	The class 'DrawTool' has a Cyclomatic Complexity of 9 (Highest = 46).
9	src/DrawTool.java	40	This class has too many methods, consider refactoring it.
10	src/DrawTool.java	90	The method 'tempaddlist' has a Cyclomatic Complexity of 19.
11	src/DrawTool.java	120	The method 'spBrush2Cal' has a Cyclomatic Complexity of 46.
12	src/DrawTool.java	219	The method 'spBrush1Cal' has a Cyclomatic Complexity of 46.
13	src/DrawTool.java	317	The method 'addlist' has a Cyclomatic Complexity of 27.
14	src/DrawTool.java	384	Avoid really long methods.
15	src/DrawTool.java	732	Avoid really long methods.
16	src/DrawTool.java	896	All classes and interfaces must belong to a named package
17	src/DrawTool.java	926	Avoid instantiating new objects inside loops
18	src/DrawTool.java	940	Avoid instantiating new objects inside loops
19	src/DrawTool.java	944	Avoid instantiating new objects inside loops
20	src/DrawTool.java	948	Avoid instantiating new objects inside loops
21	src/DrawTool.java	952	Avoid instantiating new objects inside loops
22	src/DrawingObject.java	18	All classes and interfaces must belong to a named package
23	src/FileIO.java	24	All classes and interfaces must belong to a named package
24	src/FileIO.java	72	Avoid empty catch blocks

Below the report, a file explorer window shows the directory structure: J:\workspace\modling\reports. A red box highlights the generated report files: cpd-report.txt, pmrd-report.csv, pmrd-report.html, pmrd-report.txt, pmrd-report.vb.html, and pmrd-report.xml.

Generate Reports 기능을 통해서 Rule 기반 코드 검사를 한 결과를 XML, txt, HTML 등의 Report로 만들어낼 수 있다.

After Using PMD

- 코드 작성시 피해야 할 패턴들을 학습하게 해준다.
- 기본적으로 세팅되어 있는 룰이 너무 많아

코드 작성의 제약 사항이 많아질 수 있다.

-> 따라서 진행하는 프로젝트 특성에 따라

필요한 룰셋들만을 설정하여 사용하는 것이 좋다.

- Rule 기반 검사, 중복된 코드 확인 등으로
refactoring에 유용하게 사용될 수 있는 code inspection tool이다.

References

- <http://blog.naver.com/dive2net?Redirect=Log&logNo=40062353250>
- <http://bcho.tistory.com/156>
- <http://okjsp.tistory.com/1165642997>
- http://bullseye.com/coverage.html#basic_path
- <http://blog.daum.net/aqua0405/5558428>
- http://alica_park.blog.me/30120565913
- http://www.sten.or.kr/bbs/board.php?bo_table=test_story
- <http://docs.codehaus.org/>
- www.honamsw.com/?bid=pds&m=bbs&uid=38
- <http://pmd.sf.net/eclipse>
- <http://merds.tistory.com/160>
- <http://blog.daum.net/zfan11/7786936>
- <http://pmd.sourceforge.net/>