System testing & Static Analysis

2nd Report

Team #1

200711460 이상열

200711470 정재호

201111344 김재엽

201211350 박주광

Contests

1. System Testing

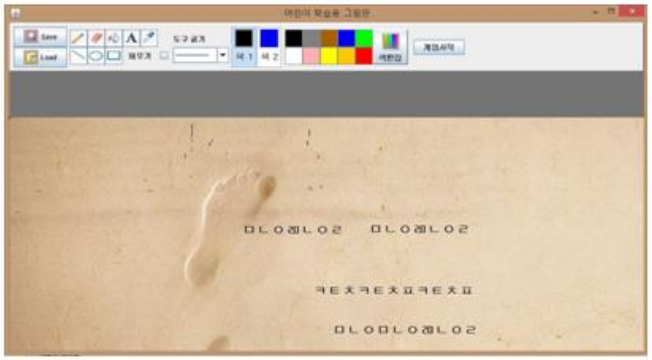
System Testing Result #1

Test Case No. 203

현재 보여지는 화면만 잘라내어 저장하는 현상으로 보여진다. 아래 앞서 <원본 그림>은 9999x9999규격으로 저장한 파일이며 오른쪽 <저장 1>과 <저장 2>는 프로그램을 사용 <원본 그림>을 불러서 프로그램 창의 크기를 다르게 하여 저장 기능을 수행한 것이다.

System Testing Result #2

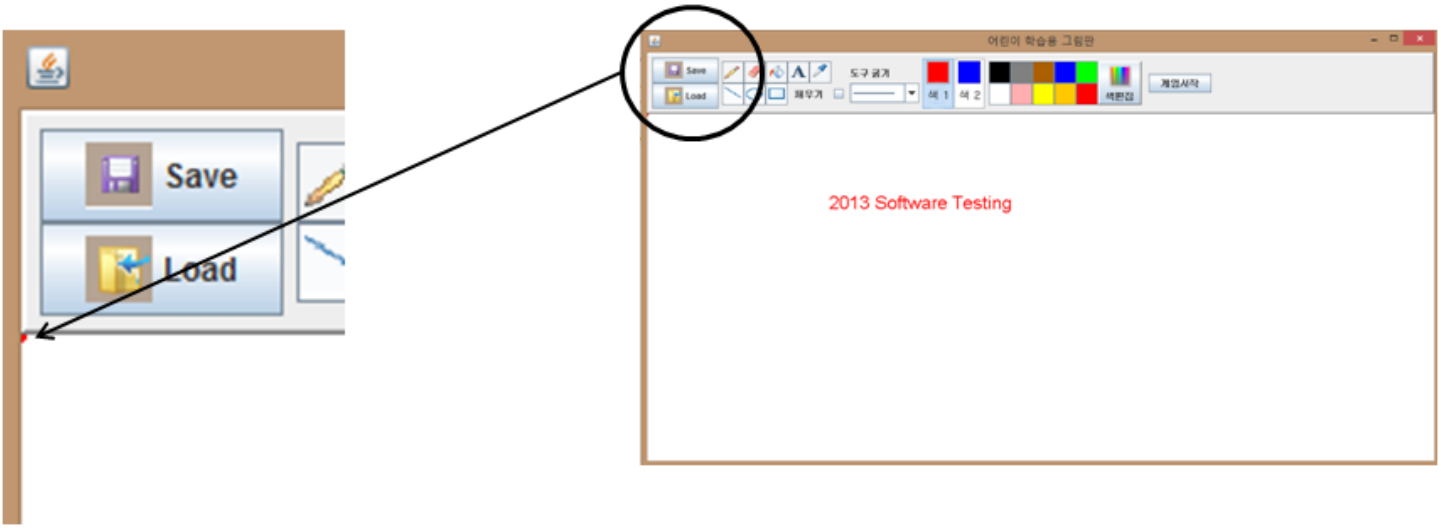
Test Case No. 301, 302, 303



특정 작업(연필, 지우개, 페인트)을 수행하면 화면이 아래로 내려가는 현상

System Testing Result #3

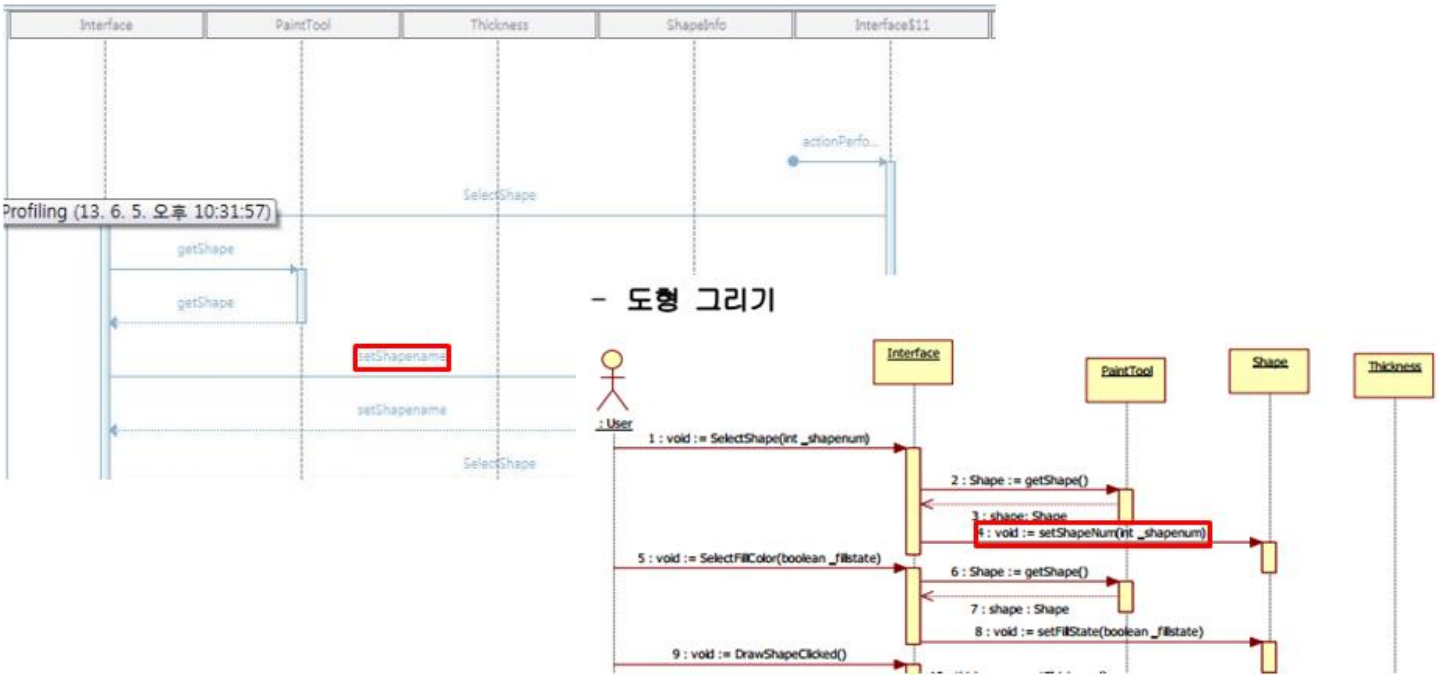
Test Case No. 304



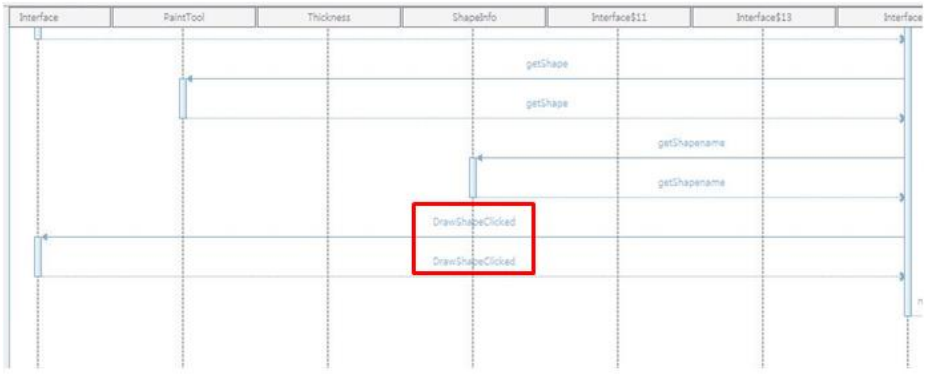
텍스트 입력 시 왼쪽 위편에 텍스트 색상과 같은 색상의 점 발생

2. Static Analysis – Eclipse TPTP

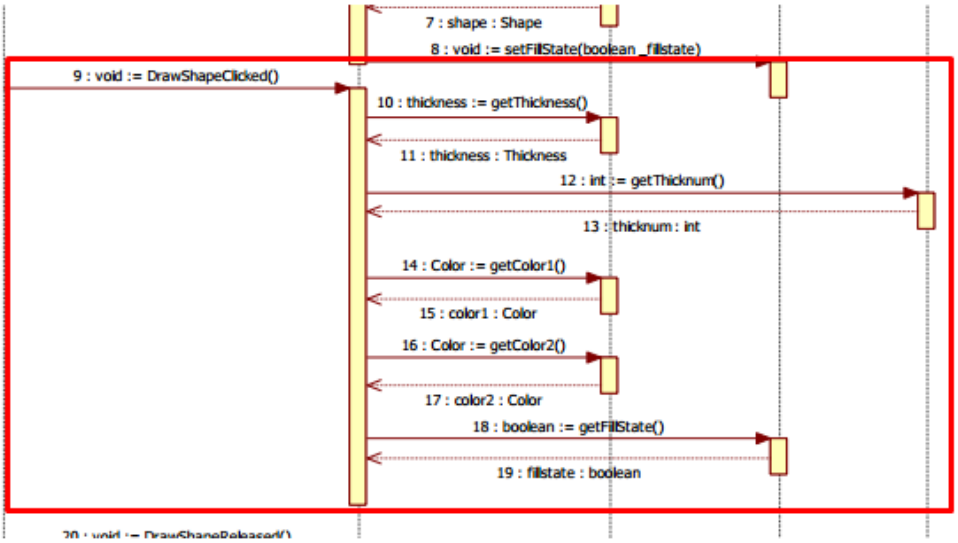
Analysis #1 (도형 그리기)



실제 코드와 문서상 method의 이름이 맞지 않음

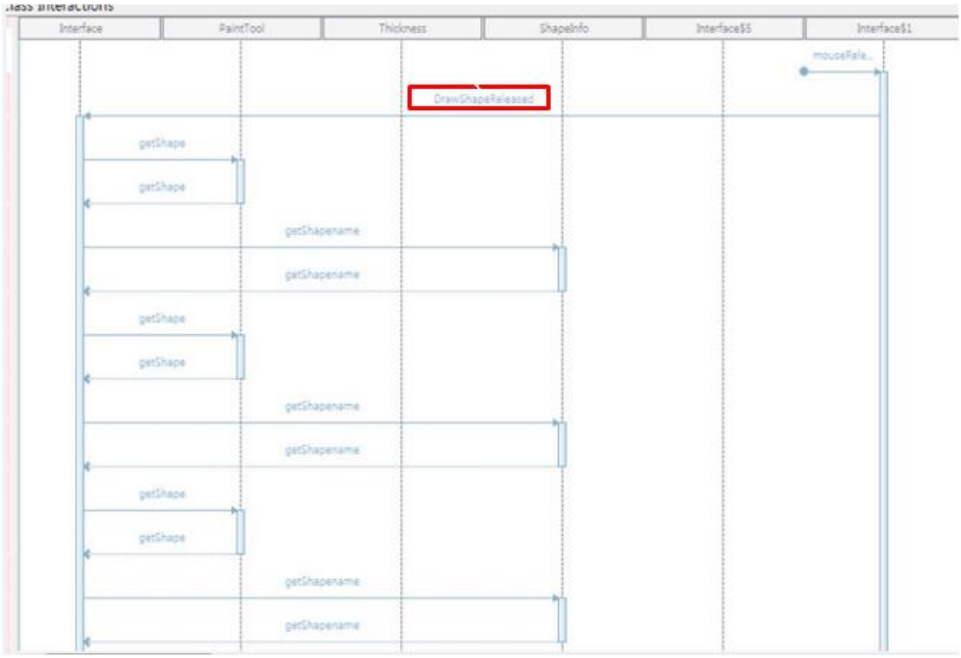


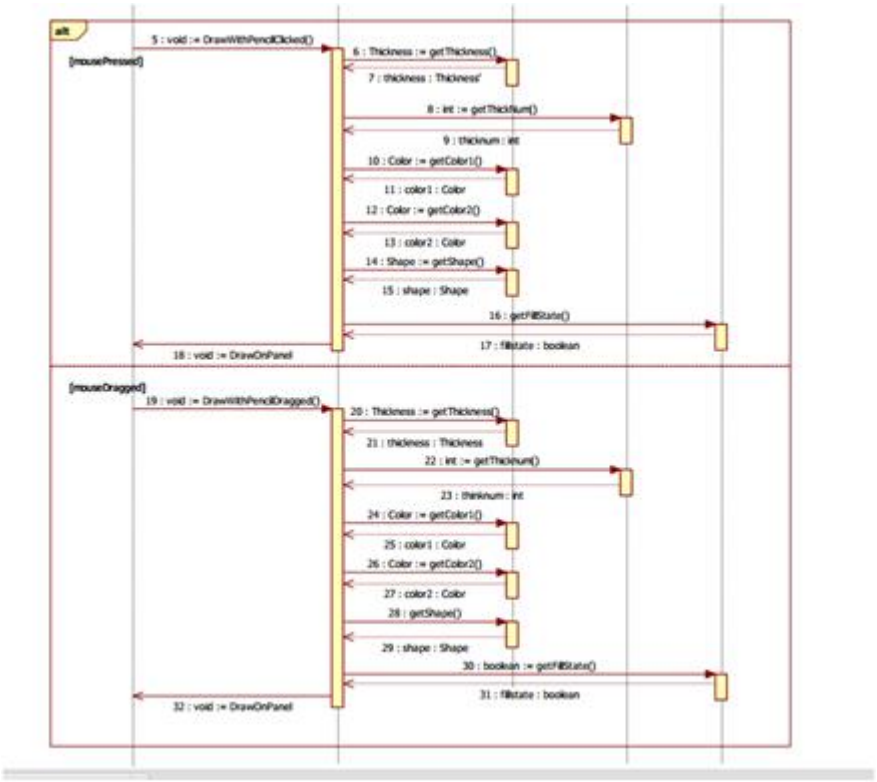

```
public void DrawShapeClicked(MouseEvent arg0) {
    //사각형이나 원일경우 눌렀을 때 좌표를 예쁜 좌표로 지정
    lx=arg0.getX();
    ly=arg0.getY();
}
```



문서상의 method기능과 실제 간 차이 발생

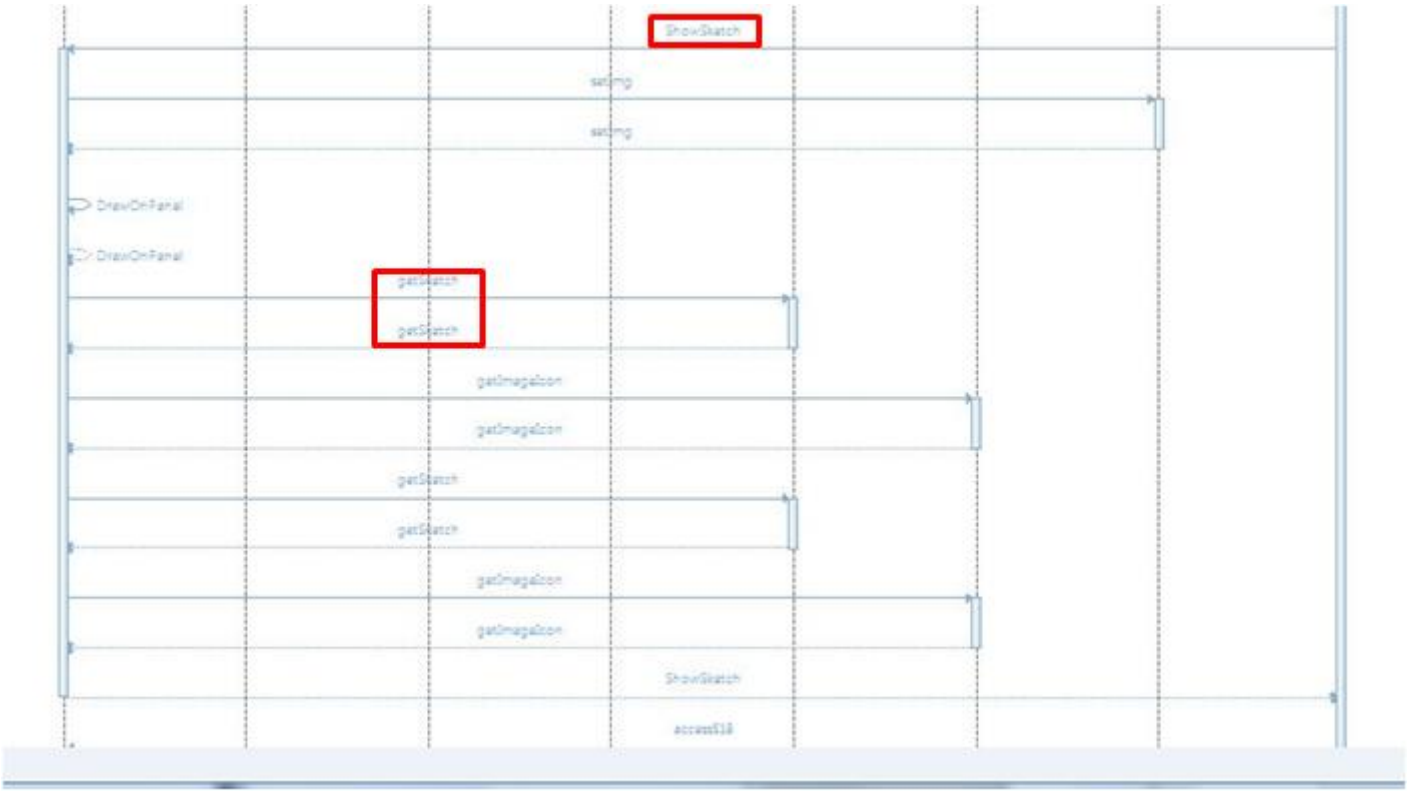
Analysis #2 (연필 그리기)



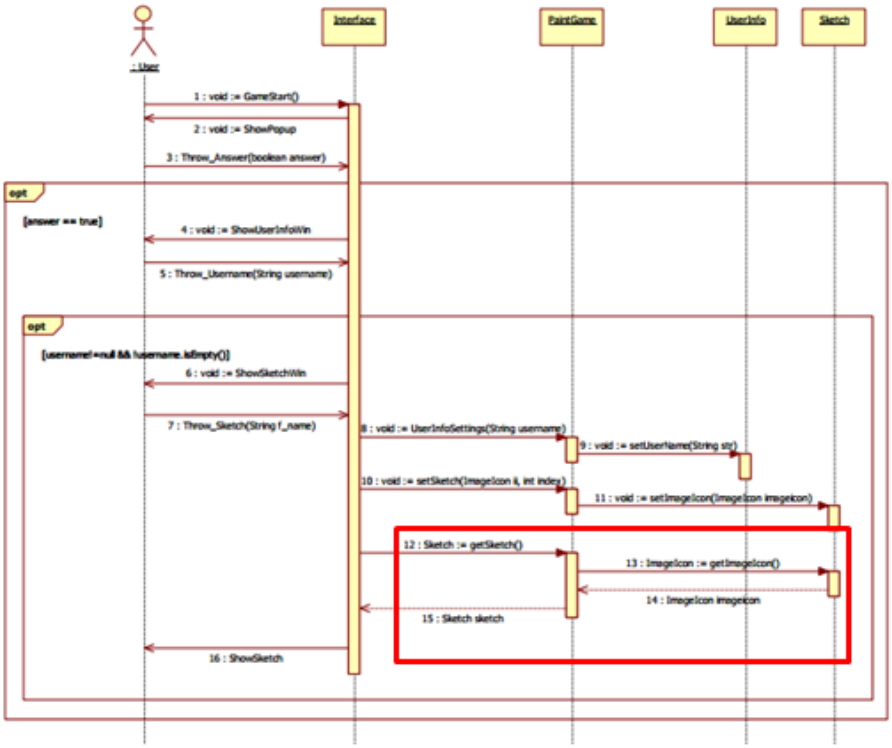


마우스 press와 dragged 부분만 정의되어 있고 release부분은 정의 되어 있지 않음

Analysis #3 (게임 시작하기)

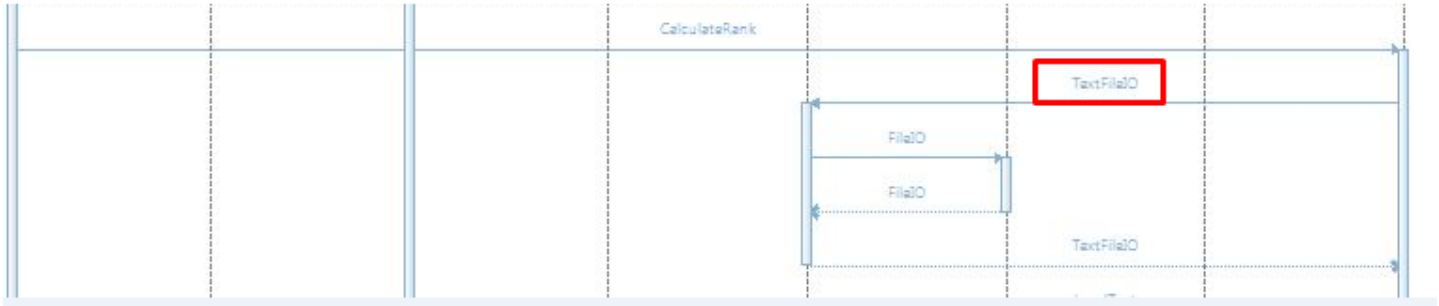


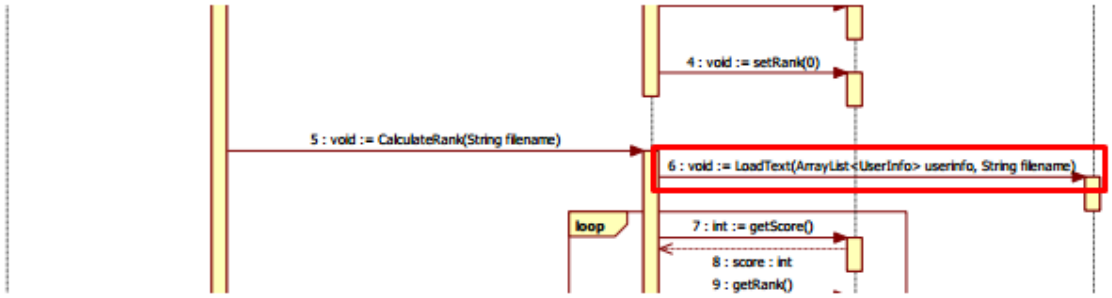
- 게임 시작하기



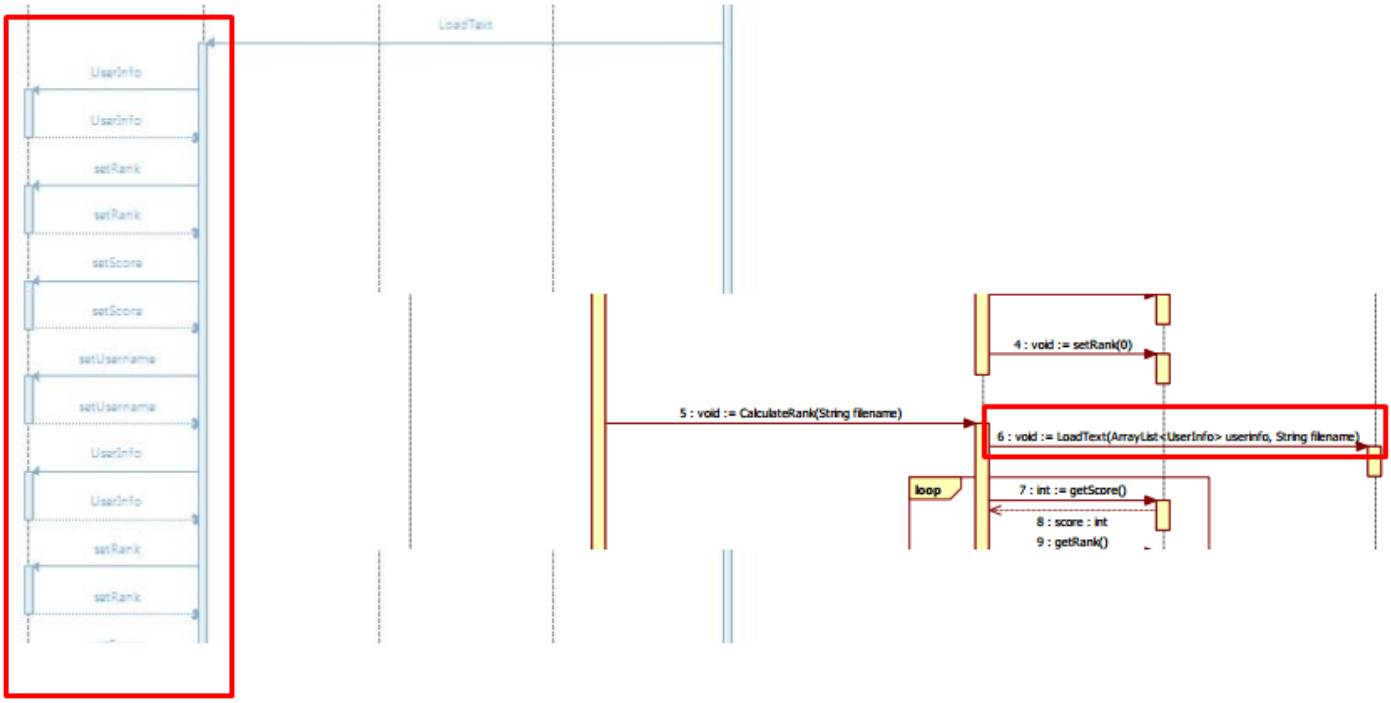
ShowSketch이후에 getSketch가 시작되지만 문서상에는 이와 같이 표현되어 있지 않음

Analysis #4 (채점하기)

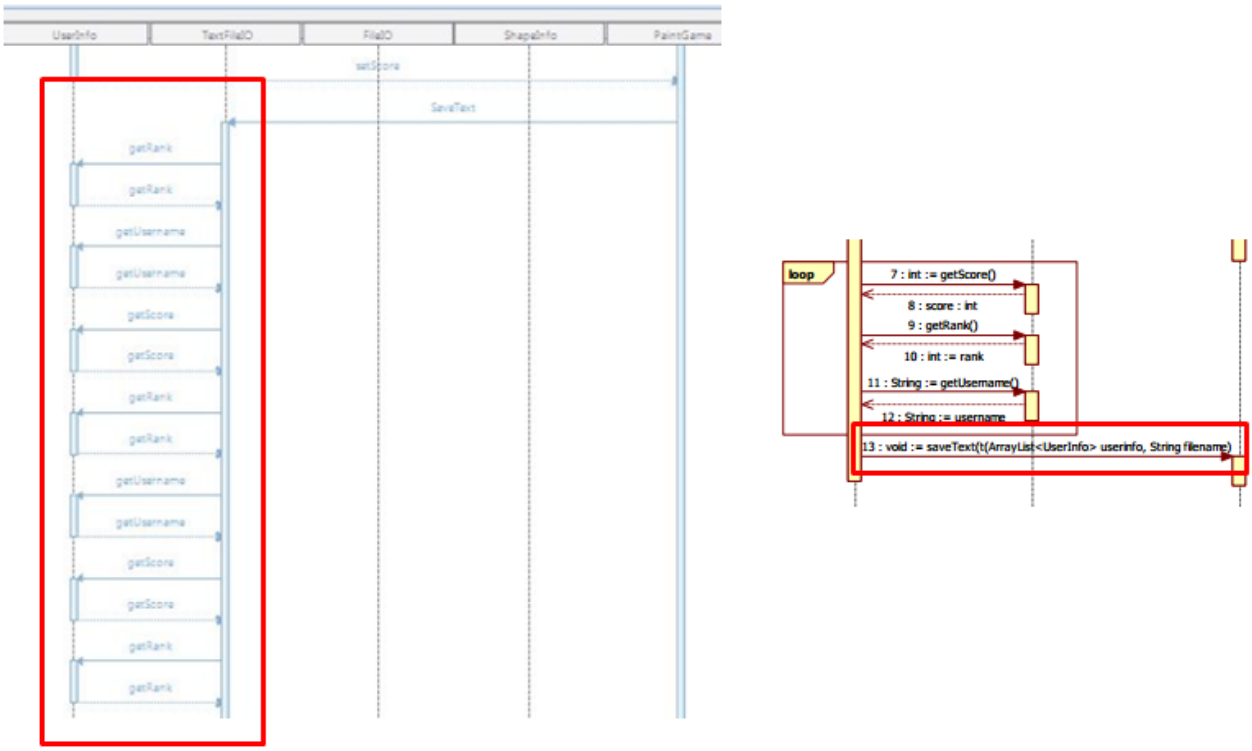




문서상에 TextFileI/O가 빠져있음



LoadText시에도 loop가 돌게 되어 있는데 표현이 안 됨. 밑의 기능은 loop가 돈다는 것을 명시했는데, 이러한 표기법에서 통일성이 요구됨



SaveText역시 LoadText와 통일하게 표기법의 통일성이 요구 됨

2. Static Analysis – SONAR

1. Introduction

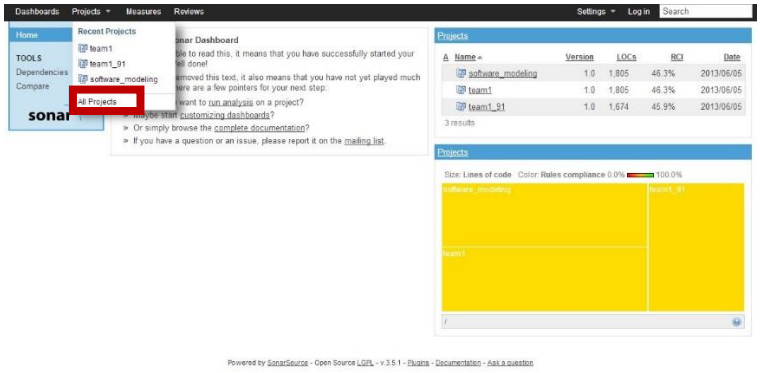
I. Purpose

Introduction을 통해 Sonar 에서 제공하는 Report의 기본 용어들을 설명하고, Sonar Report를 통해 분석대상을 설정한다. 그리고 Analysis 에서는 Sonar Report에서 설정한 대상을 분석하고, 해당 문제를 해결할 방안을 제시한다.

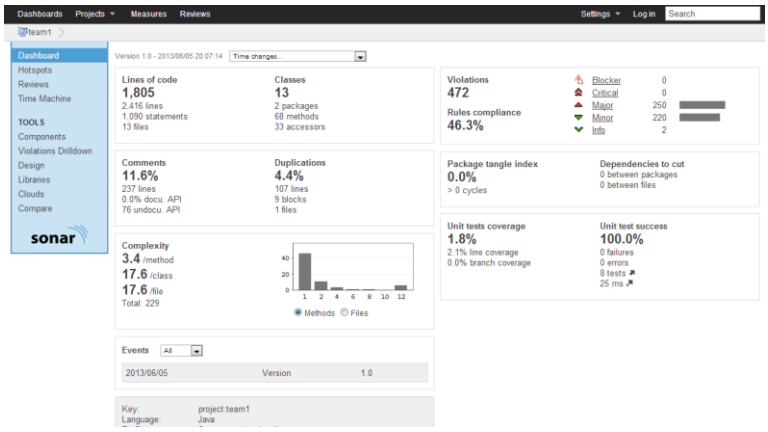
II. How to access sonar

i. http://server.brainguys.net:9000/ 다음 주소로 접속.

ii. Projects 에서 해당 프로젝트(team1)를 클릭하여 분석결과 화면(Dashboard)으로 이동한다.



iii. 다음 화면과 같이 분설결과 화면(Dashboard)을 확인할 수 있음.



III. Metric Definition(용어 설명)

i. Lines of code

해당 프로젝트의 총 라인 수를 의미하며, sonar에서 사용하는 모든 단위의 기본이 된다. 예를 들어 뒤에 나올 Duplications(중복 code)에서 중복된 라인 수와 그 라인수가 전체 프로젝트에서 차지하는 비율을 나타낸다. Statements는 예를 들어 if, while, for문과 같이 JAVA language specification상에서 statements로 정의된 구문의 개수를 의미한다.

ii. Class

클래스의 개수를 의미한다. Sonar에서는 function을 Method와 Accessor로 구분하는데 Accessor는 해당 클래스에 접근할 수 있는 함수들(ex: setter, getter)를 의미하고 Method는 그 외의 함수들을 의미한다.

iii. Comments

주석의 비율 및 라인 수를 나타낸다. Documented API는 라이브러리상에 선언된 주석을 의미하며, Undocumented API는 개발자가 임의로 삽입한 주석을 의미한다.

iv. Duplication

중복된 소스 코드에 대해 라인 수와 블록 수, 그리고 해당 중복이 일어나 파일 개수를 나타내어 준다. 변수에 대해서는 변수의 Type만 구분하고 변수의 값이 같은 것은 체크하지 않는다.

v. Complexity

If, for, while, case, catch, throw, return, &&, ||, ?등과 같은 Keyword가 나타나면 count를 올려서 complexity를 측정한다. 우측의 막대그래프는 복잡도에 따른 Method또는 file, Class의 분포도를 보여준다.

vi. Violation

주로 code quality와 관련된 내용들로서 5단계의 severity(심각도)로 나누어 보여준다. Code standard와 관련된 규칙들로 가독성과 유지보수에 도움이 될 수 있기 때문에, 해당 규칙에 대해 고려해볼 수 있다.

vii. Coverage

Unit test를 통한 coverage를 나타내며, Jacoco나 Clover와 같은 coverage툴을 이용하여 측정한다. 앞에서 이야기한 것처럼 Sonar는 기본단위가 라인 수 이므로 주석이나 빈 공간도 coverage 비율에 포함한다. 그래서 실제 coverage보다 조금 낮게 나오기도 하므로 주의해서 보아야 한다.

2. Sonar Report

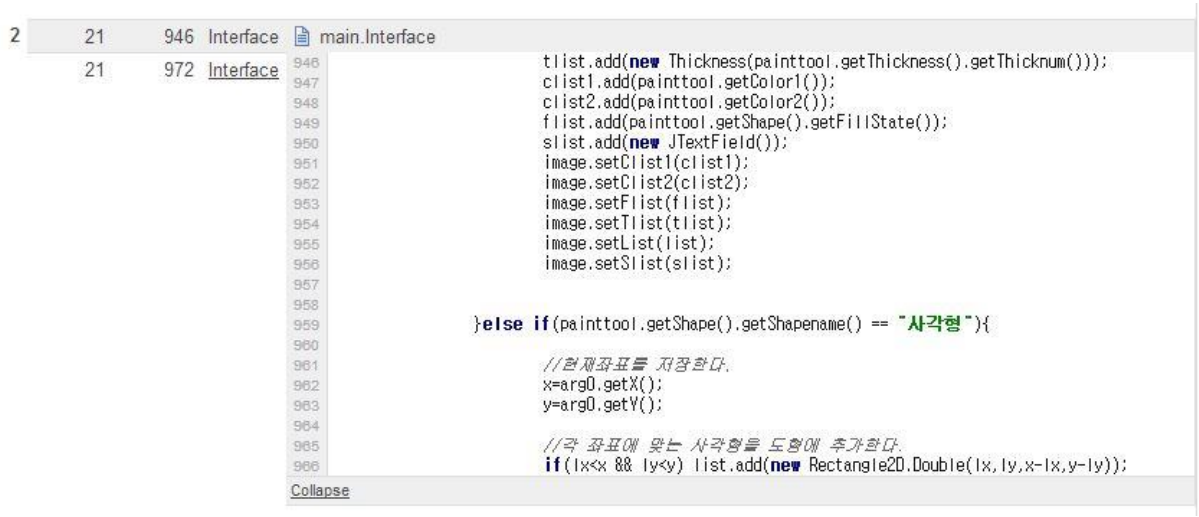
I. Purpose

Sonar Report를 통해 Analysis대상을 미리 제시하여 본 문서를 읽는 사람이 통일된 주제에 초점을 맞추고, Analysis에 대한 근거를 미리 제시한다.

II. Duplication

i. 총 3부분에서 Duplication이 발생하였으며, 해당 Duplication이 발생한 지점을 screen shot과 번호를 붙여 제시하였다.

ii. Number: D001
File: Interface.java
Class: interface
Method: public void DrawShapeReleased(MouseEvent arg0)



다음 두 Duplication는 해당 Method에서 발생하여서 하나의 Duplication으로 묶었다.

iii. Number: D002

File: Interface.java

Class: interface

Method: public void Fill_Color(MouseEvent arg0), public void Load()

다음 두 Method File_color와 Load에서 Duplicaion이 발생하였다.

iv. Number: D003

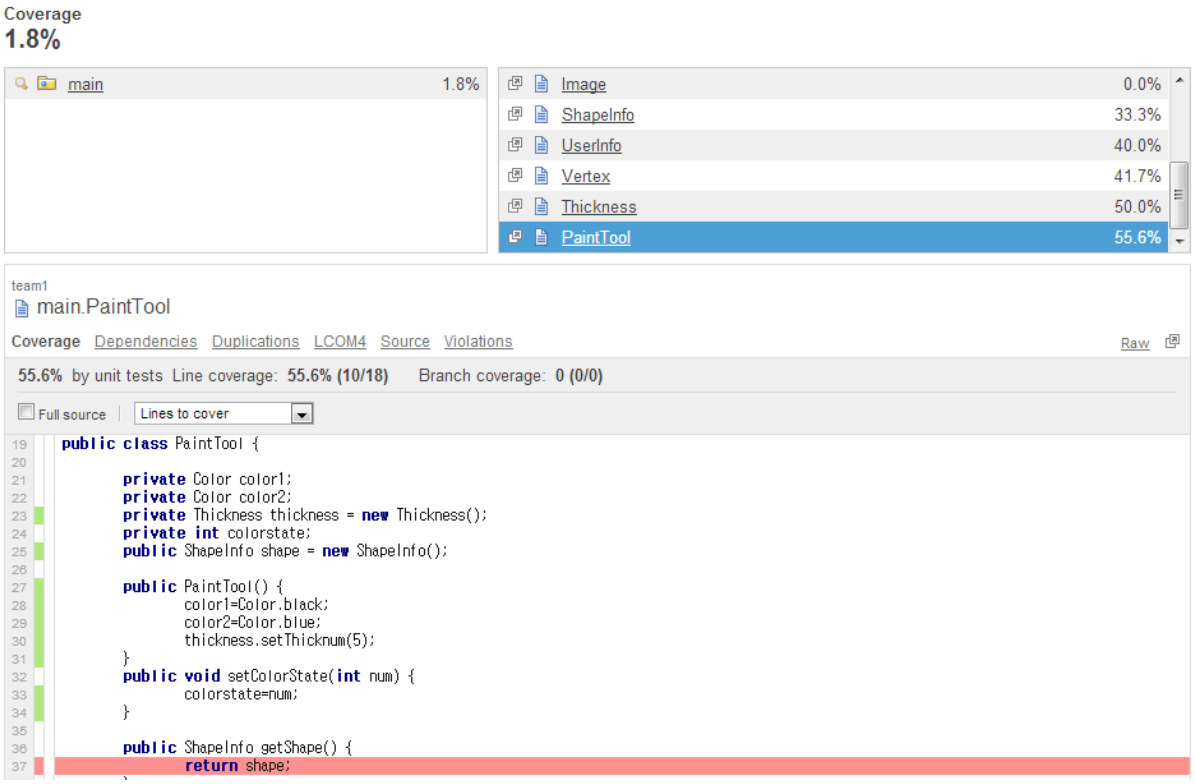
File: Interface.java

Class: interface

Method: public void ShowSketchWin(),public void ShowScoreWin()

다음 두 Method ShowSketchWin과 ShowScoreWin에서 Duplication이 발생하였다.

III. Coverage

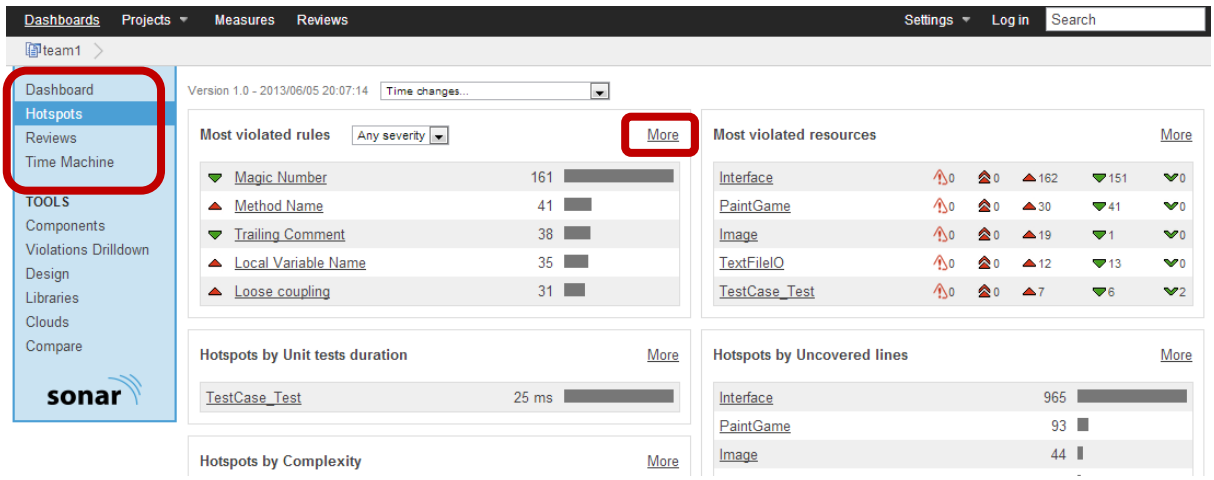


위의 Coverage는 Unit test에 대한 Coverage를 나타낸다. Coverage의 경우 다음과 같이 Unit test가 진행된 부분과 진행되지 않은 부분을 녹색라인과 빨간색 라인으로 구분하여 보여주는 것을 확인할 수 있다.

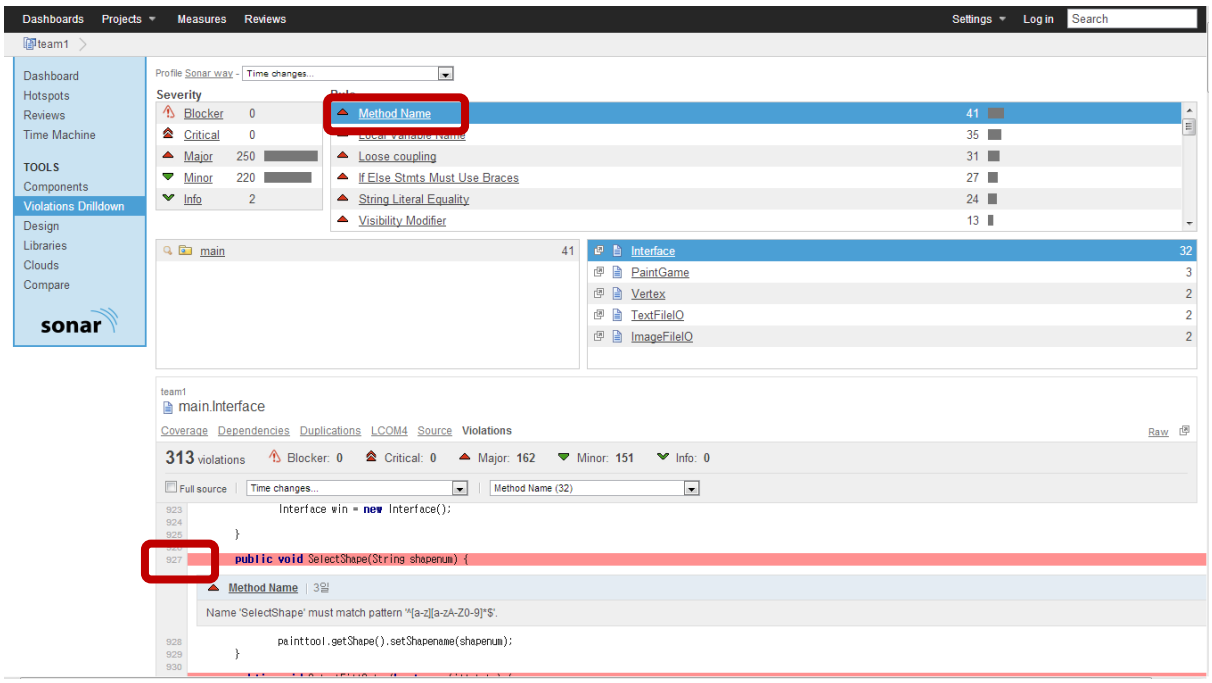
IV. Violation

i. Violation은 Code Standard에 맞추고자 하는데 그 목적이 있으며, 해당 violation이 발생한 지점의 개수와 rule name, 번호를 붙여 제시하였다.

ii. Dashboard아래의 Hotspots을 클릭 후 violated rules의 more button을 누르면 violation에 대한 정보를 한눈에 볼 수 있다.



iii. 해당 violation을 클릭하면 violation이 발생한 지점을 확인할 수 있다.



iv. 이 뒤로부터 Severity에 따라 Violation을 구분하여 violation rule에 대한 설명과 번호를 붙였다.

v. Severity: Major (V001 ~ V027)

Violation 발생 개수: 250

1. Number: V001

   Name: Method Name

   Number of violation: 41

   Description: "^[a-z][a-zA-Z0-9]*$" Method의 이름을 다음 정규식에 따라 작성할 것을 요구하는 rule 이다. 시작은 [a-z]의 소문자로 시작하여 그 뒤로 [a-zA-Z0-9]영어 숫자를 쓸 것을 권유한다.

2. Number: V002

   Name: Local Variable Name

   Number of violation: 35

Description: "^[a-z][a-zA-Z0-9]*$" 지역변수의 이름을 이름을 다음 정규식에 따라 작성할 것을 요구하는 rule이다. 시작은 [a-z]의 소문자로 시작하여 그 뒤로 [a-zA-Z0-9]영어 숫자를 쓸 것을 권유한다.

3. Number: V003
   Name: Loose coupling
   Number of violation: 31
   Description: 해당 프로젝트에서 사용하는 'ArrayList'와 같은 implementation type이 아닌 interface를 사용할 것을 권유한다.

4. Number: V004
   Name: If Else Stmts Must Use Braces
   Number of violation: 27
   Description: if else 구문을 중괄호({,})와 함께 사용할 것을 권장한다.

5. Number: V005
   Name: String Literal Equality
   Number of violation: 24
   Description: 문자열 비교는 '=='이 아니라 equals()를 이용할 것을 권장한다.(문자열 비교 함수 사용을 권장한다는 의미로 해석됩니다.)

6. Number: V006
   Name: Visibility Modifier
   Number of violation: 13
   Description: Protected로 선언된 변수들을 Private으로 선언하고 accessor(setter, getter)등으로 접근할 수 있도록 만들 것을 권장한다.

7. Number: V007
   Name: If Stmts Must Use Braces
   Number of violation: 9
   Description: if 구문을 중괄호({,})와 함께 사용할 것을 권장한다.

8. Number: V008
   Name: Avoid Print Stack Trace
   Number of violation: 9
   Description: printStackTrace()대신에 logger call을 사용할 것을 권장한다.(JAVA에서 제공하는 class 중 Logger라는 클래스에 대한 것으로 보입니다. 메시지가 발생시간을 관리하는 클래스로 보이니다.)

9. Number: V009
   Name: Member name
   Number of violation: 9
   Description: "^[a-z][a-zA-Z0-9]*$" 멤버변수의 이름을 이름을 다음 정규식에 따라 작성할 것을 요구하는 rule이다. 시작은 [a-z]의 소문자로 시작하여 그 뒤로 [a-zA-Z0-9]영어 숫자를 쓸 것을 권유한

다.

10. Number: V010
    Name: Integer Instantiation
    Number of violation: 7
    Description: instantiating Integer objects 대신에 Integer.valueOf()사용할 것을 권장한다.

11. Number: V011
    Name: Hidden Field
    Number of violation: 6
    Description: 해당 변수나 클래스가 filed를 감추어서 발생한다. (정확하게 무엇을 의미하는지는 모르겠으나 간략하게 조사해본 결과, Hidden filed는 보안상 취약점이 될 수 있다고 합니다. 그래서 이러한 부분을 줄일 것을 권유하는 것 같습니다.)

12. Number: V012
    Name: Anon Inner Length
    Number of violation: 5
    Description: Inner class의 길이가 최대길이로 설정된 20보다 길다. (class 내부에 선언된 다른 class의 정의가 길어서 발생하는 거 같습니다. Complexity에는 방영되지 않으나 class의 규모가 필요이상으로 커지는 것을 막기 위해, inner class의 길이를 줄이것을 권유하는 것 같습니다.)

13. Number: V013
    Name: Parameter Name
    Number of violation: 5
    Description: "^[a-z][a-zA-Z0-9]*$" Parameter의 이름을 이름을 다음 정규식에 따라 작성할 것을 요구하는 rule이다. 시작은 [a-z]의 소문자로 시작하여 그 뒤로 [a-zA-Z0-9]영어 숫자를 쓸 것을 권유한다.

14. Number: V014
    Name: Avoid Duplicate Literals
    Number of violation: 5
    Description: 특정 문자열(ex: 게임시작, 지우개)등이 여러 번 나타난다.

15. Number: V015
    Name: Local Final Variable Name
    Number of violation: 3
    Description: "^[a-z][a-zA-Z0-9]*$" final로 선언된 지역변수들의 이름을 이름을 다음 정규식에 따라 작성할 것을 요구하는 rule이다. 시작은 [a-z]의 소문자로 시작하여 그 뒤로 [a-zA-Z0-9]영어 숫자를 쓸 것을 권유한다.

16. Number: V016
    Name: Cyclomatic Complexity
    Number of violation: 3

Description: 해당 Method들의 Cyclomatic Complexity이 20이상으로 나타나므로 이를 줄여야 한다. (Cyclomatic Complexity은 위에서 서술한 것처럼 compelxity를 측정하는 한 지표이다.)

17. Number: V017
    Name: For Loops Must Use Braces
    Number of violation: 3
    Description: for 루프문의 중괄호({,})와 함께 사용할 것을 권장한다.

18. Number: V018
    Name: Simplify Boolean Expression
    Number of violation: 3
    Description: 좀 더 간단히 표현할 수 있는 항목들을 보여준다.

19. Number: V019
    Name: Ncss Method Count
    Number of violation: 2
    Description: Ncss가 최대길이(50)을 넘어갔을 때 발생한다. (NCSS는 'Not Commented Source Statements'로 일정 길이 이상(50)주석이 달리지 않은 Method에 대해서만 발생합니다.)

20. Number: V020
    Name: Unused local variable
    Number of violation: 2
    Description: 사용되지 않는 지역변수들을 보여준다.

21. Number: V021
    Name: Use Index Of Char
    Number of violation: 2
    Description: String.indexOf(char)가 String.indexOF(String)보다 빠르므로 이를 사용할 것을 권장한다.

22. Number: V022
    Name: Avoid commented-out lines of code
    Number of violation: 1
    Description: '//'다음과 함께 사용된 주석의 사용을 제지할 것을 요구한다.

23. Number: V023
    Name: Constructor Calls Overridable Method
    Number of violation: 1
    Description: Overridable Method인 함수 'SelectFileColor'가 Object 생성 중에 호출되는 것을 지적하는 부분이다.

24. Number: V024
    Name: Parameter Assignment
    Number of violation: 1

Description: Parameter로 선언된 변수에 다시 값을 넣지 않는 것을 권유한다.

25. Number: V025
    Name: Unnecessary Local Before Return
    Number of violation: 1
    Description: return이 되는 local변수를 굳이 return하지 않고 내부에서 저장할 것을 권유한다.

26. Number: V026
    Name: Ncss Type Count
    Number of violation: 1
    Description: NCSS가 최대 길이를 넘어간 경우 발생한다.(위에서 나왔던 것과 마찬가지로 NCSS가 일정 길이(이 경우에는 Method가 아니라 file을 체크하므로 200)이상 넘게 되면 발생합니다.)

27. Number: V027
    Name: Unused Private Field
    Number of violation: 1
    Description: 사용하지 않는 private 선언부를 지울 것을 권장한다.

vi. Severity: Minor (V028~V030)
Violation 발생 개수: 220

1. Number: V028
   Name: Magic Number
   Number of violation: 161
   Description: 개발자가 임의로 설정한 수치를 의미한다.

2. Number: V029
   Name: Trailing Comment
   Number of violation: 38
   Description: 코드가 끝나는 줄에 '//주석'다음과 같이 주석이 들어가는 것을 찾아낸다.

3. Number: V030
   Name: Singular Field
   Number of violation: 21
   Description: 지역 변수로 선언될 수 있는 private 멤버들을 찾아낸다.

vii. Severity: Info (V031)
Violation 발생 개수: 2

1. Number: V031
   Name: Unused Imports
   Number of violation: 2
   Description: 참조하지 않는 라이브러리들을 찾아낸다.

3. Analysis

I. Purpose

Sonar Report에서 제시한 Analysis 대상을 분석하여 건의사항과 그 근거를 제시하여, 프로젝트의 완성도를 높인다.

II. Duplication

i. D001

이 Duplication의 경우 주로 도형과 관련된 class의 변수를 설정하거나 출력하는 부분으로 보인다. 따라서 각 도형 클래스 내부에 해당 속성들의 설정해주는 Method를 만들어주거나, 또는 interface 클래스에 해당 도형의 속성을 설정해주는 Method를 만들어 주는 방법들이 있을 것이다. 다만, Method를 만들어 줌으로써 가독성이 떨어지거나 복잡도가 올라갈 수 있으므로, 개발자에 재량에 따라 적절한 방법을 취할 것을 권유한다.

ii. D002

D001의 Duplication과 마찬가지로 비슷한 동작을 하는 부분이 잡혀있다. 위와 같은 방법으로 해결할 수 있다.

iii. D003

D001의 Duplication과 마찬가지로 비슷한 동작을 하는 부분이 잡혀있다. 위와 같은 방법으로 해결할 수 있다.

III. Coverage

i. Coverage가 1.8%밖에 안되어 coverage가 상당히 낮아 보이지만, 실질적으로 accessor(setter, getter)는 굳이 Unit test를 거치지 않아도 괜찮으며, 주석이나 변수 선언부 또한 coverage비율에 포함되므로 coverage가 낮게 나오는 것은 큰 문제가 아니라고 볼 수 있다. 다만, PaintGame.java의 public void CalculateScore(BufferedImage d_img, String filename)의 경우 System testing에서도 정확한 수치를 확인할 방법이 없었으므로 unit test를 한 번 거쳐보는 것이 좋을 것 같다.

IV. Violation

i. Violation의 경우 프로젝트가 실행되는 logic을 고려하지 않고, 단순히 code standard를 지키기 위해 제시된다. 따라서, 실제로 프로그램이 어떻게 돌아가는 지에 따라 violation rule어기게 되더라도 큰 문제는 되지 않는다고 볼 수 있다. 다만, 유지, 보수 및 source code공유를 고려 한다면, V001이나 V004와 같은 부분들도 고려해보아야 한다. 밑에 나열하는 violation은 다음과 같은 취지에서 선정해본 것들이다.

ii. V006, V011, V023: 다음 violation들은 보안과 관계된 부분이므로 수정해볼 수 있다.

iii. V012, V018, V021: 다음 violation들을 지킴으로써 프로그램의 효율성을 올릴 수 있으므로 수정해볼 수 있다.

4. Result

I. Sonar를 통해서 얻은 결과들은 System testing처럼 오류를 잡아내어 수정할 것을 요구하지 않는다. 위의 분석결과와 같이 효율성이나 보안등과 같이 프로그램의 quality를 높이는데 사용할 수 있는 결과들을 제공한다. 따라서, 해당 결과들을 무조건 수용하지 않고 재량에 맞게 선택적으로 수용해야 한다.

II. Duplication의 경우 위에서 말한 것처럼 duplication을 줄임으로써 발생하는 Trade-off를 고려하여 code 수정을 해야한다.

III. Coverage의 경우도 실제 수치보다 Unit test가 거쳐가야 할만한 부분을 찾아내 Unit test를 거치는 것이 바람직하다.

IV. Violation의 경우, Source code의 가독성과 유지 및 보수에 관련된 부분들이 많으므로, 되도록이면 지킬 수 있도록 하는 것이 바람직하다.