

SYSTEM TEST & STATIC ANALYSIS

2013 SPRING SOFTWARE VERIFICATION

TEAM 1

200711460 이상열

200711470 정재호

201111344 김재엽

201211350 박주광

CONTENTS

System Test

Static Analysis

Customer's View

Reference

Functional Analysis

Functional Analysis

System Testing

System Test Result (cont.)

202.306	아무거나 불러온 후 직선을 그린다.	Pass
202.307.131	아무거나 불러온 후 채우기를 선택하고 동그라미를 그린다.	Pass
202.307.132	아무거나 불러온 후 채우기를 선택하지 않고 동그라미를 그린다.	Pass
202.308.131	아무거나 불러온 후 채우기를 선택하고 네모를 그린다.	Pass
202.308.132	아무거나 불러온 후 채우기를 선택하지 않고 네모를 그린다.	Pass
203.411.401.301	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 연필로 그린다.	Pass
203.411.401.301.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 연필로 그리고 채점한다.	Pass
203.411.401.302	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 지우개로 지운다.	Pass
203.411.401.302.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 지우개로 지우고 채점한다.	Pass
203.411.401.303	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 색 채우기를 한다.	Pass
203.411.401.303.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 색 채우기를 하고 채점한다.	Pass
203.411.401.304.411	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 텍스트를 선택하고 아무것도 입력하지 않는다.	Pass
203.411.401.304.411.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 텍스트를 선택하고 아무것도 입력하지 않고 채점한다.	Pass
203.411.401.304.412	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 텍스트를 선택하고 많이 입력한다.	Pass
203.411.401.304.412.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 텍스트를 선택하고 많이 입력한 후 채점한다.	Pass
203.411.401.305	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 스포이드를 사용한다.	Pass
203.411.401.305.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 스포이드를 사용하고 채점한다.	Pass
203.411.401.306	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 직선을 그린다.	Pass

System Testing

System Test Result (cont.)

203.411.401.306.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 직선을 그리고 채점한다.	Pass
203.411.401.307.131	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 채우기를 선택하고 원을 그린다.	Pass
203.411.401.307.131.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 채우기를 선택하고 원을 그린 후 채점한다.	Pass
203.411.401.307.132	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 채우기를 선택하지 않고 원을 그린다.	Pass
203.411.401.307.132.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 채우기를 선택하지 않고 원을 그린 후 채점한다.	Pass
203.411.401.308.131	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 채우기를 선택하고 네모를 그린다.	Pass
203.411.401.308.131.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 채우기를 선택하고 네모를 그린 후 채점한다.	Pass
203.411.401.308.132	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 채우기를 선택하지 않고 네모를 그린다.	Pass
203.411.401.308.132.204	게임시작 후 이름에 아무것도 입력하지 않고 확인버튼을 누르고 채우기를 선택하지 않고 네모를 그린 후 채점한다.	Pass
203.411.402.301	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 연필로 그린다.	Pass
203.411.402.302	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 지우개로 지운다.	Pass
203.411.402.303	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 색 채우기를 한다.	Pass
203.411.402.304.411	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 텍스트를 선택하고 아무것도 입력하지 않는다.	Pass
203.411.402.304.412	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 텍스트를 선택하고 많이 입력한다.	Pass
203.411.402.305	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 스포이드를 사용한다.	Pass
203.411.402.306	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 직선을 그린다.	Pass

System Testing

System Test Result (cont.)

203.411.402.307.131.308	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 채우기를 선택하고 네모를 그린다.	Pass
203.411.402.307.132.308	게임시작 후 이름에 아무것도 입력하지 않고 x버튼을 누르고 채우기를 선택하지 않고 네모를 그린다.	Pass
203.412.401.301	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 연필로 그린다.	Pass
203.412.401.301.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 연필로 그리고 채점한다.	Pass
203.412.401.302	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 지우개로 지운다.	Pass
203.412.401.302.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 지우개로 지우고 채점한다.	Pass
203.412.401.303	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 색 채우기를 한다.	Pass
203.412.401.303.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 색 채우기를 하고 채점한다.	Pass
203.412.401.304.411	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 텍스트를 선택하고 아무것도 입력하지 않는다.	Pass
203.412.401.304.411.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 텍스트를 선택하고 아무것도 입력하지 않고 채점한다.	Pass
203.412.401.304.412	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 텍스트를 선택하고 많이 입력한다.	Pass
203.412.401.304.412.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 텍스트를 선택하고 많이 입력한 후 채점한다.	Pass
203.412.401.305	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 스포이드를 사용한다.	Pass
203.412.401.305.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 스포이드를 사용하고 채점한다.	Pass
203.412.401.306	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 직선을 그린다.	Pass
203.412.401.306.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 직선을 그리고 채점한다.	Pass
203.412.401.307.131	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 채우기를 선택하고 원을 그린다.	Pass

System Testing

System Test Result (cont.)

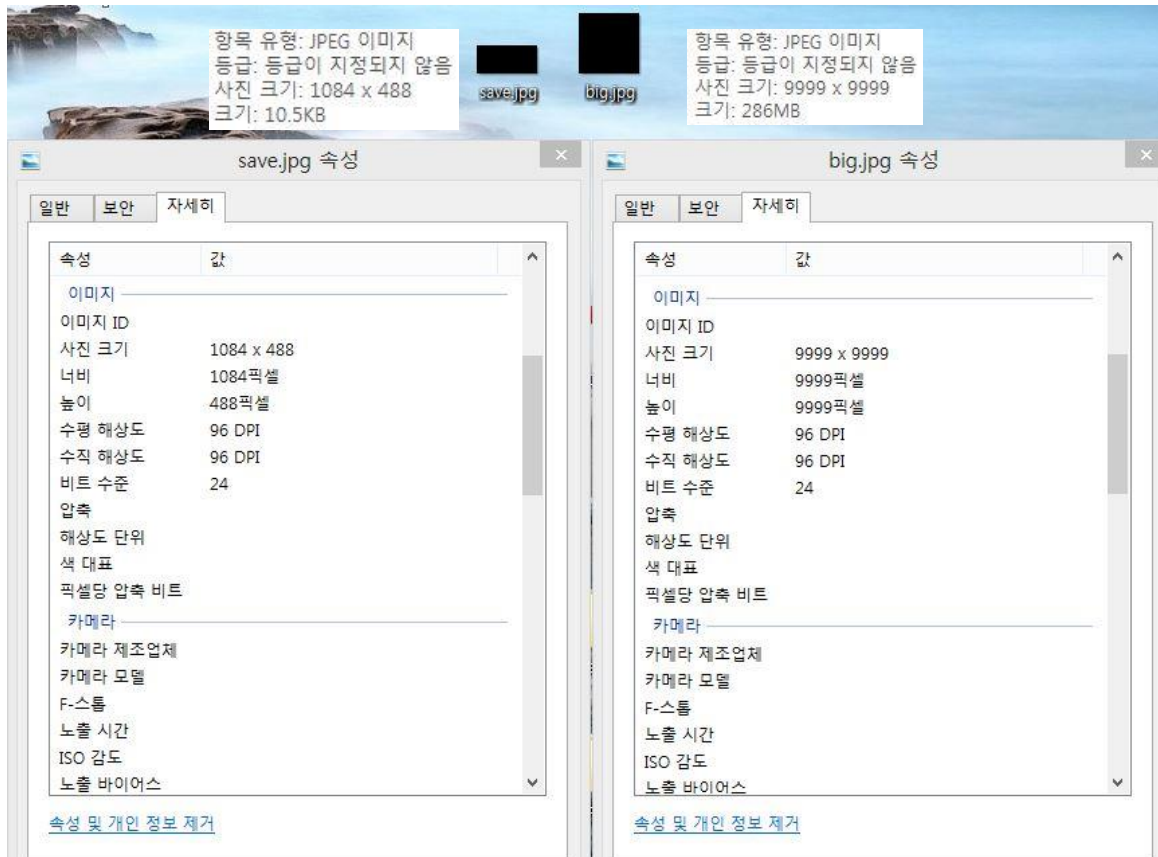
203.412.401.307.131.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 채우기를 선택하고 원을 그린 후 채점한다.	Pass
203.412.401.307.132	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 채우기를 선택하지 않고 원을 그린다.	Pass
203.412.401.307.132.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 채우기를 선택하지 않고 원을 그린 후 채점한다.	Pass
203.412.401.308.131	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 채우기를 선택하고 네모를 그린다.	Pass
203.412.401.308.131.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 채우기를 선택하고 네모를 그린 후 채점한다.	Pass
203.412.401.308.132	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 채우기를 선택하지 않고 네모를 그린다.	Pass
203.412.401.308.132.204	게임시작 후 임의의 이름을 입력하고 확인버튼을 누르고 채우기를 선택하지 않고 네모를 그린 후 채점한다.	Pass
203.412.402.301	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 연필로 그린다.	Pass
203.412.402.302	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 지우개로 지운다.	Pass
203.412.402.303	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 색 채우기를 한다.	Pass
203.412.402.304.411	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 텍스트를 선택하고 아무것도 입력하지 않는다.	Pass
203.412.402.304.412	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 텍스트를 선택하고 많이 입력한다.	Pass
203.412.402.305	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 스포이드를 사용한다.	Pass
203.412.402.306	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 직선을 그린다.	Pass
203.412.402.307.131.308	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 채우기를 선택하고 네모를 그린다.	Pass
203.412.402.307.132.308	게임시작 후 임의의 이름을 입력하고 x버튼을 누르고 채우기를 선택하지 않고 네모를 그린다.	Pass
203.412.401.204.402	게임시작 후 임의의 이름을 입력하고 채점후 x버튼을 누른다.	Pass
203.412.304.412.204.304	게임시작 후 임의의 이름을 입력하고 텍스트 박스를 쓰고 채점 한 뒤 텍스트 박스를 누른다.	Pass
304.412.203.304	텍스트 입력 후 게임 상태에서 텍스트박스를 누른다.	Pass

-> 92개의 Test Case 모두 통과

System Testing

Brute Force Testing Result #1

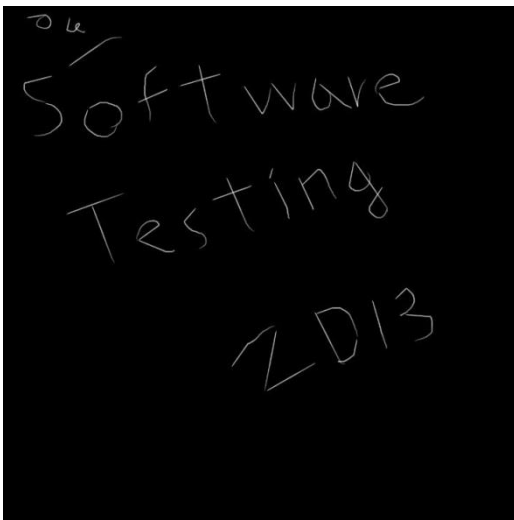
♪ 크기가 큰 jpg 파일을 불러오면 임의로 파일을 잘라내는 현상



System Testing

Brute Force Testing Result #1 (cont.)

- 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.



<저장 1>



<저장 2>

System Testing

Brute Force Testing Result #2

- ♪ 특정 작업을 수행하면 화면이 아래로 내려가는 현상
(특정 작업은 연필, 지우개, 페인트 등을 칭함)

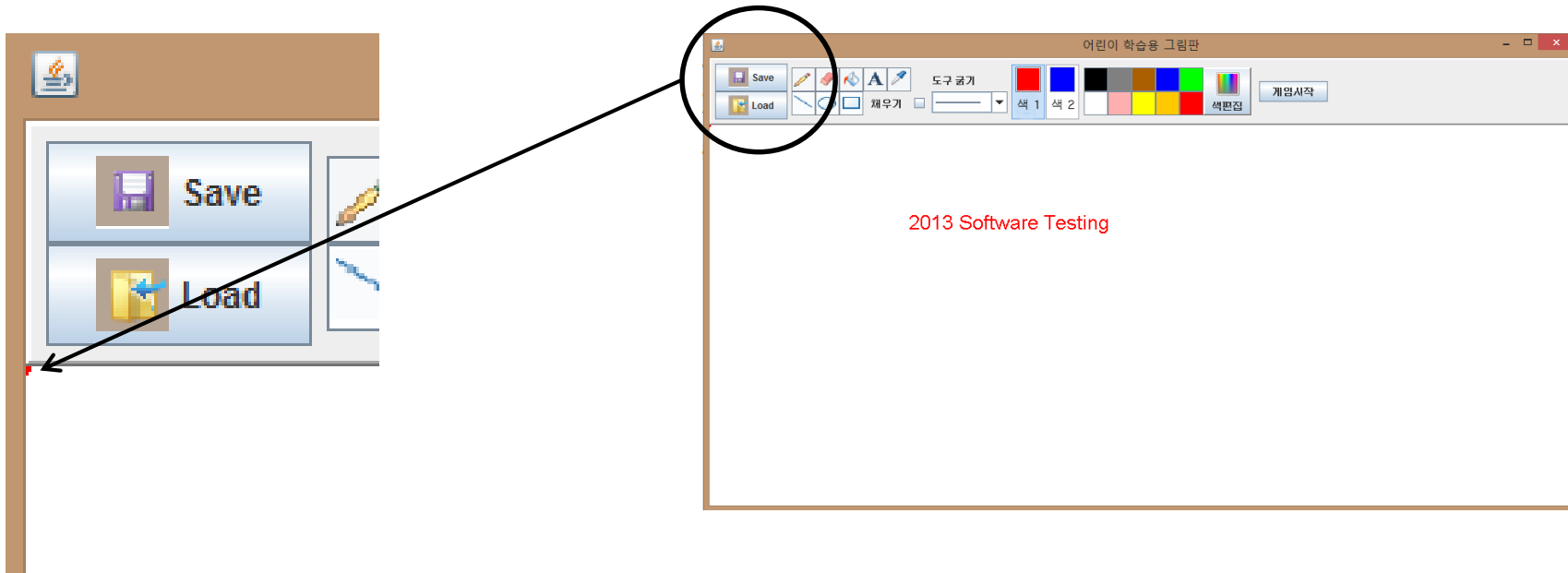


- > 똑같은 작업을 시행해도 발생할 때가 있고 발생하지 않을 경우가 있다.
4대의 노트북에서 동일한 현상을 목격. 정확한 원인 규명은 하지 못함.

System Testing

Brute Force Testing Result #3

♪ 텍스트 입력 시 왼쪽 위편에 텍스트 색상과 같은 색상의 점 발생



Static Analysis

Static Analysis

Static Analysis

Sonar + CruiseControl

<http://docs.codehaus.org/display/SONAR/Installing+and+Configuring+Sonar+Ant+Task>
에서 ant-task.jar 파일을 받아 적절한 위치에 저장

Sonar와 연계 할 ant build file을 다음과 같이 수정하여 연동을 함 (build.xml)

```
<property name="sonar.jdbc.url" value="jdbc:mysql://localhost:3306/sonar?useUnicode=true&characterEncoding=utf8" />
<property name="sonar.jdbc.username" value="sonar" />
<property name="sonar.jdbc.password" value="sonar123" />
<property name="sonar.sourceEncoding" value="euc_kr" />
<property name="sonar.projectKey" value="project:team1" />
<property name="sonar.projectName" value="team1" />
<property name="sonar.projectVersion" value="1.0" />
<property name="sonar.language" value="java" />
<property name="sonar.sources" value="PaintGame/src" />
<property name="sonar.binaries" value="target" />
```


Static Analysis

Sonar + CruiseControl (cont.)

```
<taskdef uri="antlib:org.sonar.ant" resource="org/sonar/ant/antlib.xml">
  <classpath path="/var/www/sonar/lib/sonar-ant-task-2.1.jar" />
</taskdef>

<target name="sonar">
  <sonar:sonar xmlns:sonar="antlib:org.sonar.ant"/>
</target>
```

ant 명령어로 sonar target을 지정하여 테스트

 <u>software modeling</u>	1.0	1,805	46.3%	18:30
--	-----	-------	-------	-------

성공적으로 작동 했을 시엔 위와 같이 프로젝트가 추가됨

Static Analysis

Sonar + CruiseControl (cont.)

Build Output

```
Buildfile: projects/software_modeling/build.xml
ccAntProgress -- clean
[delete] Deleting directory /var/www/cruisecontrol/projects/software_modeling/target
ccAntProgress -- compile
[mkdir] Created dir: /var/www/cruisecontrol/projects/software_modeling/target/classes
[javac] Compiling 12 source files to /var/www/cruisecontrol/projects/software_modeling/target/classes
[javac] Note: /var/www/cruisecontrol/projects/software_modeling/PaintGame/src/main/Interface.java uses or overrides a deprecated API.
[javac] Note: Recompile with -Xlint:deprecation for details.
[javac] Note: /var/www/cruisecontrol/projects/software_modeling/PaintGame/src/main/PaintGame.java uses unchecked or unsafe operations.
[javac] Note: Recompile with -Xlint:unchecked for details.
ccAntProgress -- test
[mkdir] Created dir: /var/www/cruisecontrol/projects/software_modeling/target/test-classes
[javac] Compiling 1 source file to /var/www/cruisecontrol/projects/software_modeling/target/test-classes
[javac] Note: /var/www/cruisecontrol/projects/software_modeling/PaintGame/src/testcase/TestCase_Test.java uses unchecked or unsafe operations.
[javac] Note: Recompile with -Xlint:unchecked for details.
[mkdir] Created dir: /var/www/cruisecontrol/projects/software_modeling/target/test-results
[junit] Running testcase.TestCase_Test
[junit] Testsuite: testcase.TestCase_Test
[junit] Tests run: 8, Failures: 0, Errors: 0, Time elapsed: 0.263 sec
[junit] Tests run: 8, Failures: 0, Errors: 0, Time elapsed: 0.263 sec
[junit]
ccAntProgress -- jar
[jar] Building jar: /var/www/cruisecontrol/projects/software_modeling/target/software_modeling.jar
ccAntProgress -- zip
[zip] Building zip: /var/www/cruisecontrol/projects/software_modeling/target/software_modeling.zip
ccAntProgress -- init
ccAntProgress -- sonar
[sonar:sonar] Apache Ant version 1.7.0 compiled on December 13 2006
[sonar:sonar] Sonar Ant Task version: 2.1
[sonar:sonar] Loaded from: file:/var/www/sonar/lib/sonar-ant-task-2.1.jar
[sonar:sonar] INFO: Default locale: "ko_KR", source code encoding: "euc_kr"
[sonar:sonar] INFO: Work directory: /var/www/cruisecontrol/projects/software_modeling/.sonar
[sonar:sonar] INFO: Sonar Server 3.5.1
[sonar:sonar] 18:30:30.193 INFO - Load batch settings
[sonar:sonar] 18:30:30.701 INFO - User cache: /home/papimomi/.sonar/cache
[sonar:sonar] 18:30:30.729 INFO - Install plugins
[sonar:sonar] 18:30:31.675 INFO - ----- Executing Project Scan
[sonar:sonar] 18:30:32.279 INFO - Install JDBC driver
[sonar:sonar] 18:30:32.300 INFO - Apply project exclusions
[sonar:sonar] 18:30:32.304 INFO - Create JDBC datasource for jdbc:mysql://localhost:3306/sonar?useUnicode=true&characterEncoding=utf8
[sonar:sonar] 18:30:33.262 INFO - Initializing Hibernate
[sonar:sonar] 18:30:47.840 INFO - ----- Inspecting software_modeling
[sonar:sonar] 18:30:47.895 INFO - Load module settings
[sonar:sonar] 18:30:48.951 INFO - Quality profile : [name=Sonar way,language=java]
[sonar:sonar] 18:30:49.006 INFO - Excluded tests:
[sonar:sonar] 18:30:49.007 INFO - **/package-info.java
[sonar:sonar] 18:30:49.489 INFO - Configure Maven plugins
[sonar:sonar] 18:30:49.684 INFO - Compare to previous analysis
[sonar:sonar] 18:30:49.776 INFO - Compare over 5 days (2013-05-31)
[sonar:sonar] 18:30:49.792 INFO - Compare over 30 days (2013-05-06)
[sonar:sonar] 18:30:50.177 INFO - Base dir: /var/www/cruisecontrol/projects/software_modeling
[sonar:sonar] 18:30:50.178 INFO - Working dir: /var/www/cruisecontrol/projects/software_modeling/.sonar
```

CruiseControl에서 모든 작업을 처리

Static Analysis

Sonar + Jacoco + JUnit

jacoco란 clover와 같은 java code coverage library

sonar + clover도 가능하지만 License Key가 없어서 free library인 jacoco 사용

www.eclemma.org/jacoco에서 다운로드 받아서 적당한 곳에 저장

Sonar와 연계 할 ant build file을 다음과 같이 수정하여 연동을 함 (build.xml)

```
<property name="sonar.dynamicAnalysis" value="reuseReports" />
<property name="sonar.core.coveragePlugin" value="jacoco" />
<property name="sonar.surefire.reportsPath" value="target/test-results" />
<property name="sonar.jacoco.reportPath" value="target/test-results/jacoco.exec" />
```


Static Analysis

Sonar + Jacoco + Junit (cont.)

```
<taskdef uri="antlib:org.jacoco.ant" resource="org/jacoco/ant/antlib.xml">
  <classpath path="/var/www/sonar/lib/jacocoant.jar" />
</taskdef>
```

```
    <jacoco:coverage destfile="target/test-results/jacoco.exec" xmlns:jacoco="antlib:org.jacoco.
ant">
      <junit fork="yes" haltonfailure="no" printsummary="on">
        <classpath >
          <pathelement location="target/classes" />
          <pathelement location="lib/junit-4.4.jar" />
          <pathelement location="target/test-classes" />
        </classpath>
        <formatter type="brief" usefile="false" />
        <formatter type="xml" />
        <batchtest todir="target/test-results" >
          <fileset dir="target/test-classes" includes="**/*Test.class" />
        </batchtest>
      </junit>
    </jacoco:coverage>
  </target>
```

Static Analysis

Sonar + Jacoco + Junit (cont.)

```
main.Interface
Coverage Dependencies Duplications LCOM Source Violations
0.0% by unit tests Line coverage: 0.0% (0/965) Branch coverage: 0.0% (0/140)
Full source Lines to cover
70 //
71
72 public class Interface extends JComponent {
73     private PaintTool painttool = new PaintTool();
74     private PaintGame paintgame = new PaintGame();
75     private Image image = new Image();
76
77     //화면 출력에 필요한 리스트 변수들
78     //가위칼라, 그림 블러링 등 리스트 관련 변수들은 초기화 꼭 해야함!
79     private ArrayList<Shape> list = new ArrayList<Shape>();
80     private ArrayList<Thickness> tlist = new ArrayList<Thickness>(); // 굵기
81     private ArrayList<Color> clist1 = new ArrayList<Color>(); // 색상
82     private ArrayList<Color> clist2 = new ArrayList<Color>(); // 색상
83     private ArrayList<Boolean> flist = new ArrayList<Boolean>(); // 색 채우기 유무
84     private ArrayList<JTextField> slist = new ArrayList<JTextField>(); // 텍스트 입력
85     private ArrayList<Vertex> vlist = new ArrayList<Vertex>(); // 직선
86
87     private JFrame frame;
88
89     //버튼
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

Unit tests coverage
1.8%
2.1% line coverage
0.0% branch coverage

Unit test success
100.0%
0 failures
0 errors
8 tests
25 ms

Static Analysis

Sonar TroubleShooting

```
Caused by: java.lang.IllegalStateException: The folder 'target/*.jar' does not exist for 'project:team1' (base directory = /var/www/cruisecontrol/projects/team1)
```

위와 같은 에러가 난다면 ant build파일에서 sonar.binaries 부분이 잘못설정 된 것 예제에는 <property name="sonar.binaries" value="build/*.jar" />와 같이 돼 있는데 이렇게 하면 위와 같이 에러가 난다. 그냥 폴더명만 써주면 해결. 위와 같은 경우엔 'target'

```
[sonar:sonar] 20:43:41.551 INFO - Project coverage is set to 0% as build output directory does not exist: /var/www/cruisecontrol/projects/team1/.sonar/build/classes
```

jacoco를 설정했음에도 위와 같이 잘못된 build output directory를 찾는다면 ant build파일에서 sonar와 관련된 property를 맨 위에 올려두고 선처리를 하도록 하면 정상적으로 찾아

```
[sonar:sonar] 20:07:29.351 INFO - Analysing /var/www/cruisecontrol/projects/team1/target/test-results/jacoco.exec
```

Static Analysis

Sonar TroubleShooting (cont.)

•TroubleShooting (Cont.)

```
Caused by: org.sonar.api.utils.SonarException: Unable to read and import the source file : '/var/www/cruisecontrol/projects/team1/PaintGame/src/main/PaintGame.java' with the charset : 'UTF-8'.
```

위 에러는 ant build에서 해당 소스코드의 encoding을 설정하지 않았을 때 발생하는 오류

```
<property name="sonar.sourceEncoding" value="UTF-8" />
```

위와 같은 property를 넣어서 처리를 해주면 해결 됨

<sonar:sonar />, <jacoco:coverage /> 를 찾지 못한다면 namespace를 설정하지 않아서 생기는 문제이다. 다음과 같이 수정하면 해결됨

```
<sonar:sonar xmlns:sonar="antlib:org.sonar.ant" />  
<jacoco:coverage xmlns:jacoco="antlib:org.jacoco.ant" />
```

Static Analysis

Sonar - Dashboard

Version 1.0 - 2013/06/04 20:31:51

Time changes... 

Lines of code

1,805

2,416 lines
1,090 statements
13 files

Classes

13

2 packages
68 methods
33 accessors

Violations

472

Rules compliance

46.3%

	Blocker	0	
	Critical	0	
	Major	250	
	Minor	220	
	Info	2	

Comments

11.6%

237 lines
0.0% docu. API
76 undocu. API

Duplications

4.4%

107 lines
9 blocks
1 files

Unit tests coverage

-

Unit test success

0 tests

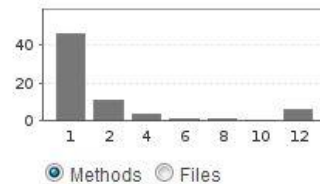
Complexity

3.4 /method

17.6 /class

17.6 /file

Total: 229



Static Analysis

Sonar – Dashboard(cont.)

- Lines of code
 - 해당 프로젝트의 총 라인 수
 - Statements : Number of statements as defined in the Java Language Specification but without block definitions (ex: if, while, etc.)
- Classes
 - 클래스의 개수
 - Number of methods/functions.
 - Accessors (ex: getter, setter)

Lines of code

1,805
2,416 lines
1,090 statements
13 files

Classes

13
2 packages
68 methods
33 accessors

Comments

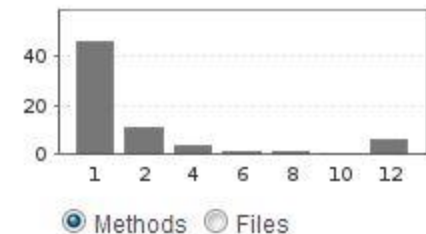
11.6%
237 lines
0.0% docu. API
76 undocu. API

Duplications

4.4%
107 lines
9 blocks
1 files

Complexity

3.4 /method
17.6 /class
17.6 /file
Total: 229



Static Analysis

Sonar – Dashboard(cont.)

- Comments
 - 주석의 비율 및 라인 수
 - Documented API는 라이브러리 상에 선언된 주석을 의미한다
- Duplication
 - 중복된 라인 수 또는 블록 수
 - 해당 중복이 발생한 file의 개수

Lines of code

1,805
2,416 lines
1,090 statements
13 files

Classes

13
2 packages
68 methods
33 accessors

Comments

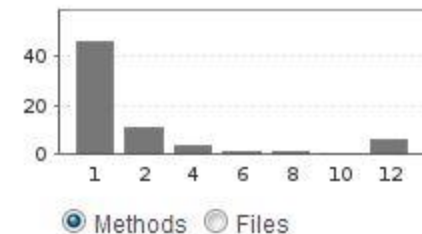
11.6%
237 lines
0.0% docu. API
76 undocu. API

Duplications

4.4%
107 lines
9 blocks
1 files

Complexity

3.4 /method
17.6 /class
17.6 /file
Total: 229



Static Analysis

Sonar – Dashboard(cont.)

- Example of duplication
 - File: Interface.java
 - Class: interface
 - Method: `public void DrawShapeReleased(MouseEvent arg0)`
- 중복이 발생하는 부분을 줄여서 라인 수를 줄이고 효율을 높일 수 있지만, 가독성이 떨어져 유지보수에 어려움이 발생할 수 있다.
- Trade-off 를 고려하여 적절한 대안을 제시할 수 있다.

```
946 Interface main.Interface
972 interface
946 t list.add(new Thickness(painttool.getThickness().getThicknum()));
947 c list1.add(painttool.getColor1());
948 c list2.add(painttool.getColor2());
949 f list.add(painttool.getShape().getFillState());
950 s list.add(new JTextField());
951 image.setClist1(c list1);
952 image.setClist2(c list2);
953 image.setFlist(f list);
954 image.setTlist(t list);
955 image.setList(l list);
956 image.setSlist(s list);
957
958
959 }else if(painttool.getShape().getShapename() == "사각형"){
960
961 //좌표표를 저장한다.
962 x=arg0.getX();
963 y=arg0.getY();
964
965 //각 좌표에 맞는 사각형을 도형에 추가한다.
966 if(1<x && 1<y) l list.add(new Rectangle2D.Double(1x,x-1x,y-1y));
Collapse
```

```
946 Interface main.Interface
972 interface
973
974
975 t list.add(new Thickness(painttool.getThickness().getThicknum()));
976 c list1.add(painttool.getColor1());
977 c list2.add(painttool.getColor2());
978 f list.add(painttool.getShape().getFillState());
979 s list.add(new JTextField());
980 //image클래스에 각 변수를 세팅
981 image.setClist1(c list1);
982 image.setClist2(c list2);
983 image.setFlist(f list);
984 image.setTlist(t list);
985 image.setList(l list);
986 image.setSlist(s list);
987
988 }else if(painttool.getShape().getShapename() == "원"){
989
990 //좌표 좌표값 저장
991 x=arg0.getX();
992 y=arg0.getY();
993
994 //좌표에 맞는 원을 그려준다.
995 if(1<x && 1<y) l list.add(new Ellipse2D.Double(1x,1y,x-1x,y-1y));
Collapse
```


Static Analysis

SONAR – Dashboard(cont.)

- Complexity
 - Keyword에 따라 complexity 가 증가한다.(ex: if, for, while, case, catch, throw, return, &&, ||, ?)
 - Method, class, file의 평균 complexity를 출력한다.
 - 막대그래프는 복잡도에 따른 method 또는 file이나 class의 분포도를 보여준다.

Lines of code

1,805

2,416 lines
1,090 statements
13 files

Classes

13

2 packages
68 methods
33 accessors

Comments

11.6%

237 lines
0.0% docu. API
76 undocu. API

Duplications

4.4%

107 lines
9 blocks
1 files

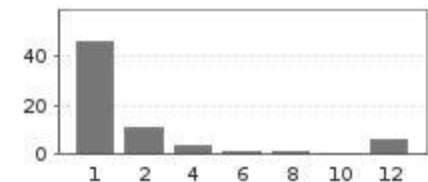
Complexity

3.4 /method

17.6 /class

17.6 /file

Total: 229



Methods Files

Static Analysis

Sonar – Dashboard(cont.)

- Example of Complexity
 - File: interface.java
 - Complexity : 165
 - Complexity/method : 4.9

Complexity /method
3.4

main	3.7
testcase	1.0

Interface	4.9
PaintGame	3.9
TextFileIO	3.5
ImageFileIO	2.5
Image	2.0
ShapeInfo	1.0

software_modeling
main.Interface

Duplications Source Violations Raw

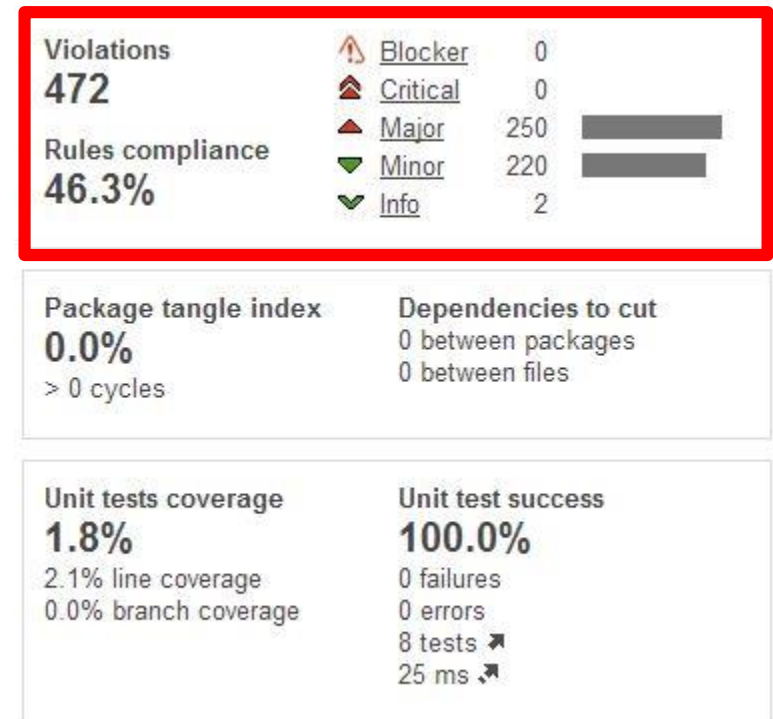
Lines:	1,717	Statements:	873	Comments (%):	12.2%	Public documented API (%):	0.0%
Lines of code:	1,291	Complexity:	165	Comment lines:	180	Public undocumented API:	38
Methods:	34	Complexity /method:	4.9			Public API:	38
Accessors:	0					Classes:	1

```
1 package main;
2 import java.awt.AWTException;
3 import java.awt.BorderLayout;
4 import java.awt.Color;
5 import java.awt.Cursor;
6 import java.awt.Dimension;
7 import java.awt.FlowLayout;
8 import java.awt.Font;
9 import java.awt.GridBagConstraints;
10 import java.awt.GridBagLayout;
11 import java.awt.GridLayout;
12 import java.awt.Insets;
13 import java.awt.Point;
```

Static Analysis

Sonar – Dashboard(cont.)

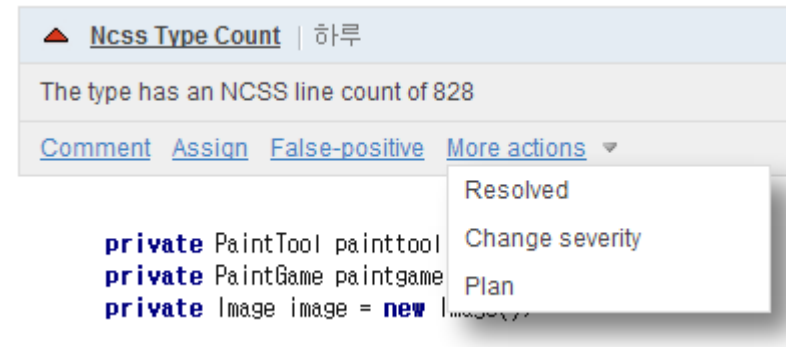
- Violations
 - 주로 code quality와 관련된 내용들이 다.
 - 각 항목들을 severity에 따라 5개의 레벨로 나누어 보여준다.
 - 각 항목마다 severity설정을 바꾸어 줄 수 있으며 comment를 달아줄 수도 있다.



Static Analysis

Sonar – Dashboard(cont.)

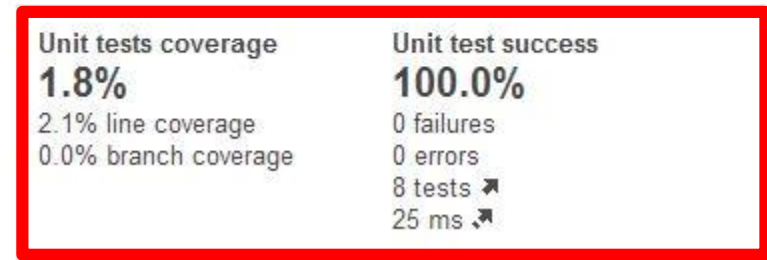
- Example of Violations
 - 해당 violation의 상태에 따라 comment, assign, false-positive등과 같은 피드백을 줄 수 있다.
 - False-positive는 해당 결과가 잘 못 나왔을 경우에 대한 설명을 쓸 수 있다.
 - Change severity는 severity를 변경하고 왜 변경하였는지에 대한 설명을 쓸 수 있다.



Static Analysis

Sonar – Dashboard(cont.)

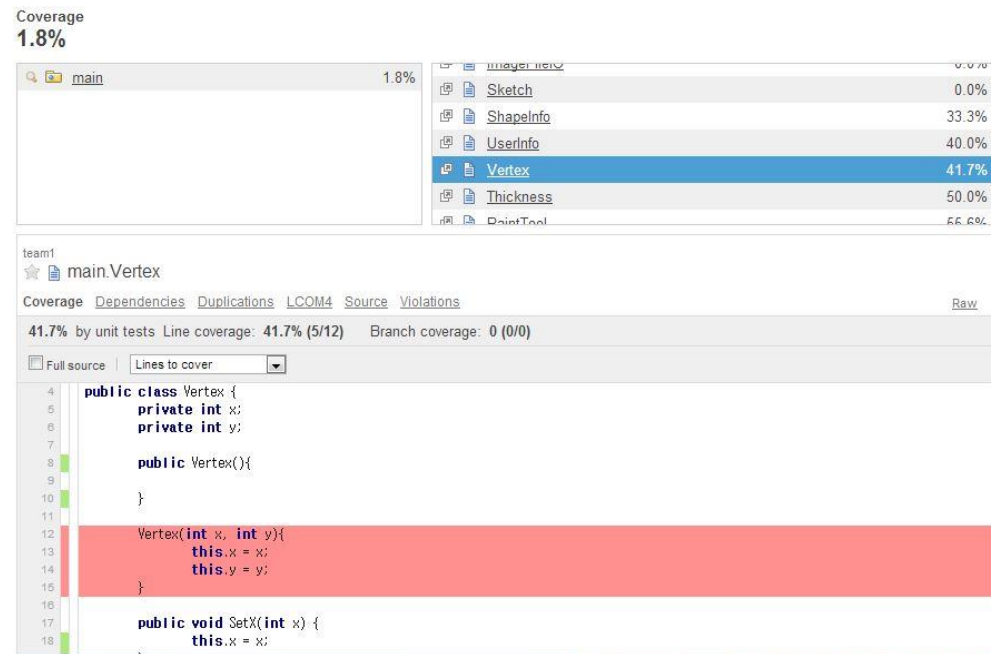
- Coverage
 - Unit test결과를 통한 coverage를 나타낸다.
 - Clover나 Jacoco와 같은 coverage측정 툴과 같이 사용하여 측정한다.
 - Unit test success는 unit test case의 성공률을 나타낸다.
- 다음 coverage분석을 통해 해당 팀의 unit test coverage가 현저히 낮은 것을 확인할 수 있다.



Static Analysis

Sonar – Dashboard(cont.)

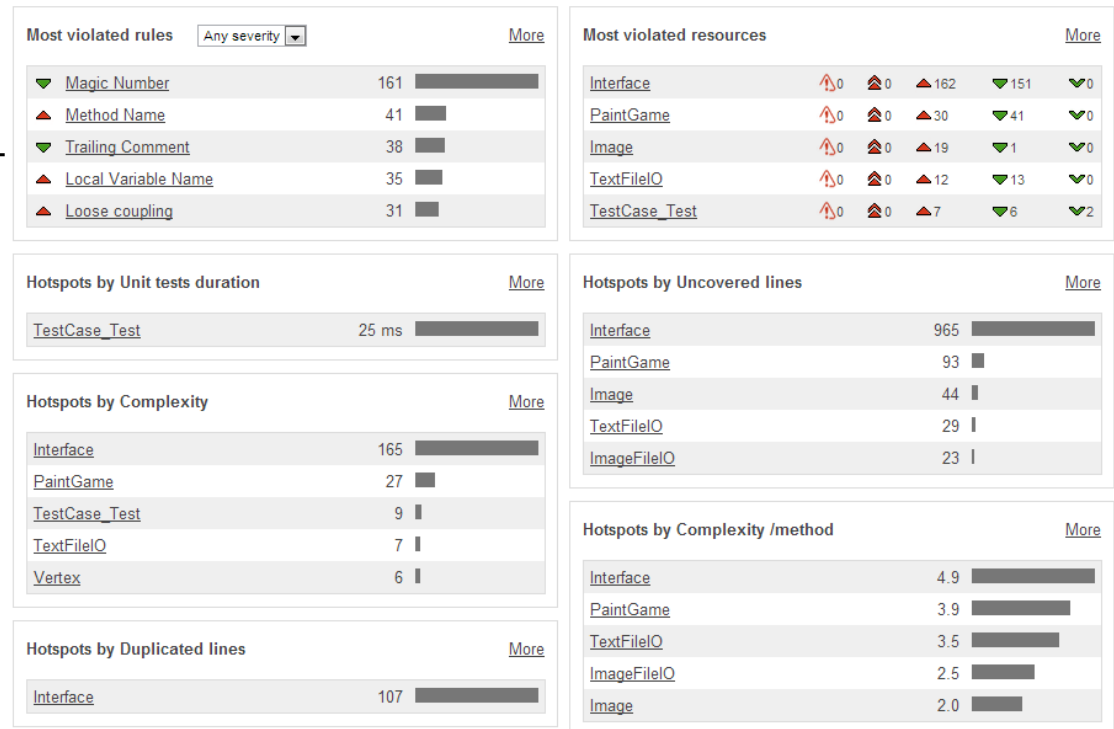
- Example of coverage
 - 오른쪽 상단에 각 파일 마다 coverage를 나타낸다.
 - 좌측 라인에 초록색 라인이 있는 부분은 unit test가 거친 부분이고, 빨간색으로 나타난 부분은 unit test를 거치지 않은 부분이다.



Static Analysis

Sonar – Hotspot

- Violation, duplication, complexity 등과 같이 이슈가 되는 부분에서 수치가 높게 나타나 있는 순서대로 내림차순으로 나타내어 보여준다.



Static Analysis

Sonar – Quality Profiles

- Setting -> quality profiles를 선택하였을 때 나타나는 페이지이다.
- 모든 violation에 대해 관리할 수 있으며, severity를 바꾸어 주거나 해당 violation을 체크할 것인지 안 할 것인지에 대해 선택할 수 있다.
- 좀 더 확장하여, 문자열에 대한 정규식이나 묵인할 개수에 대한 설정도 바꿀 수 있다.

Quality Profiles > Java > Sonar way >

Coding rules Alerts Projects Permalinks Profile inheritance Changelog

Name/Key Repository Severity Status Search

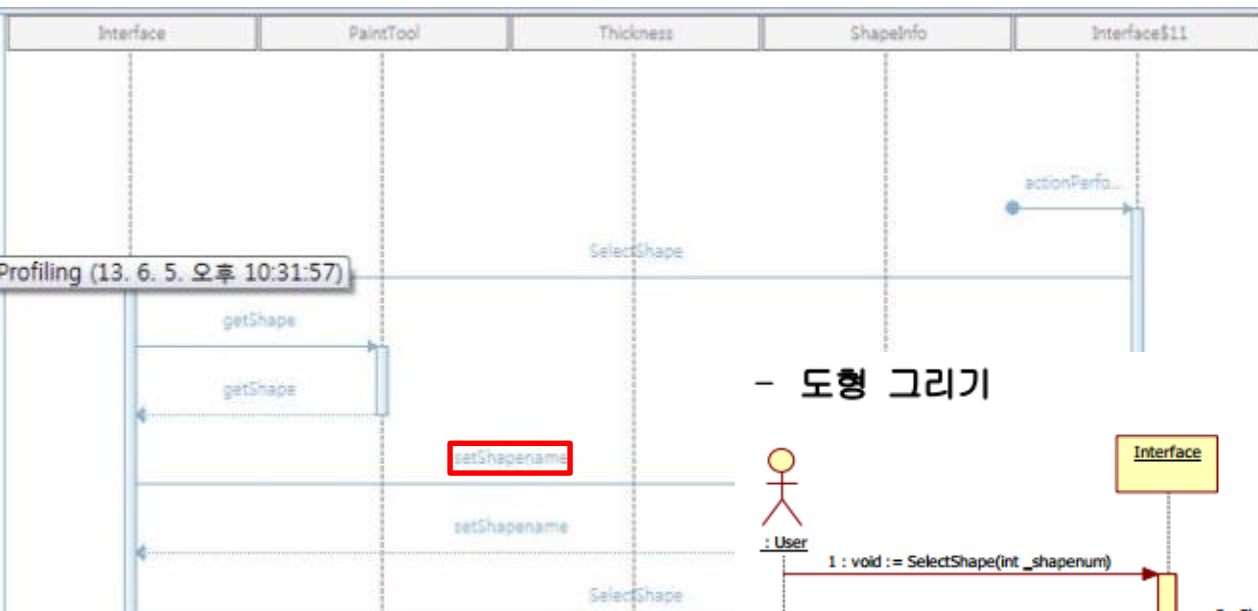
Any Checkstyle Common Sonar Findbugs PMD PMD Unit Tests Any Blocker Critical Major Minor Info Any Active Inactive

Download Bulk Change: [dropdown]

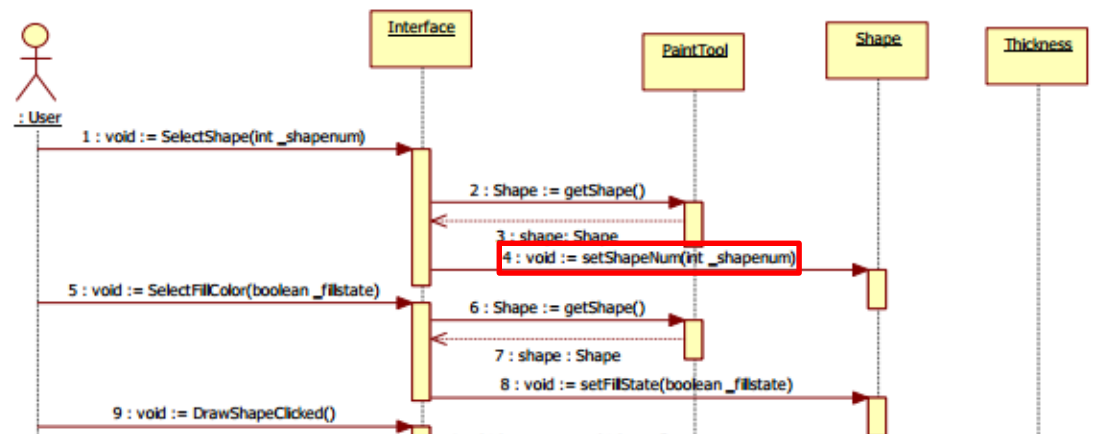
Active/Severity	Name [expand / collapse]
<input checked="" type="checkbox"/> Major	Anon Inner Length
<input checked="" type="checkbox"/> Major	Avoid Array Loops
<input checked="" type="checkbox"/> Major	Avoid Assert As Identifier
<input checked="" type="checkbox"/> Major	Avoid Calling Finalize
<input checked="" type="checkbox"/> Major	Avoid Catching NPE
<input checked="" type="checkbox"/> Critical	Avoid Catching Throwable
<input checked="" type="checkbox"/> Major	Avoid commented-out lines of code
<input checked="" type="checkbox"/> Major	Avoid cycle between java packages
<input checked="" type="checkbox"/> Major	Avoid Decimal Literals In Big Decimal Constructor
<input checked="" type="checkbox"/> Major	Avoid Duplicate Literals

Static Analysis

Eclipse TPTP – 도형 그리기

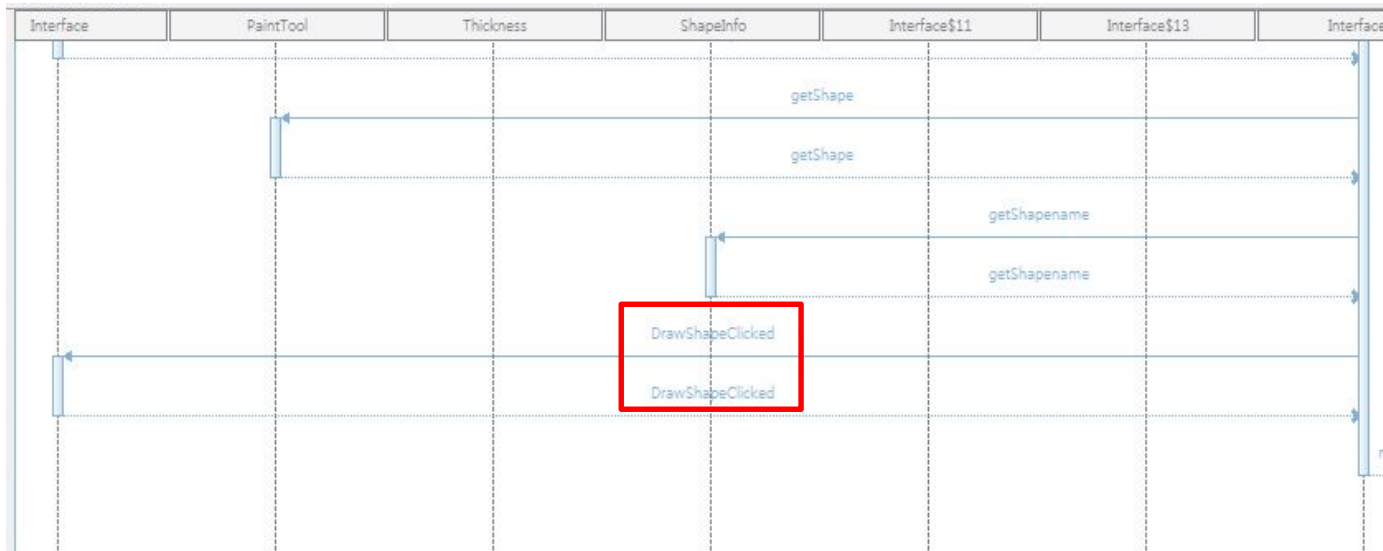


- 도형 그리기



Static Analysis

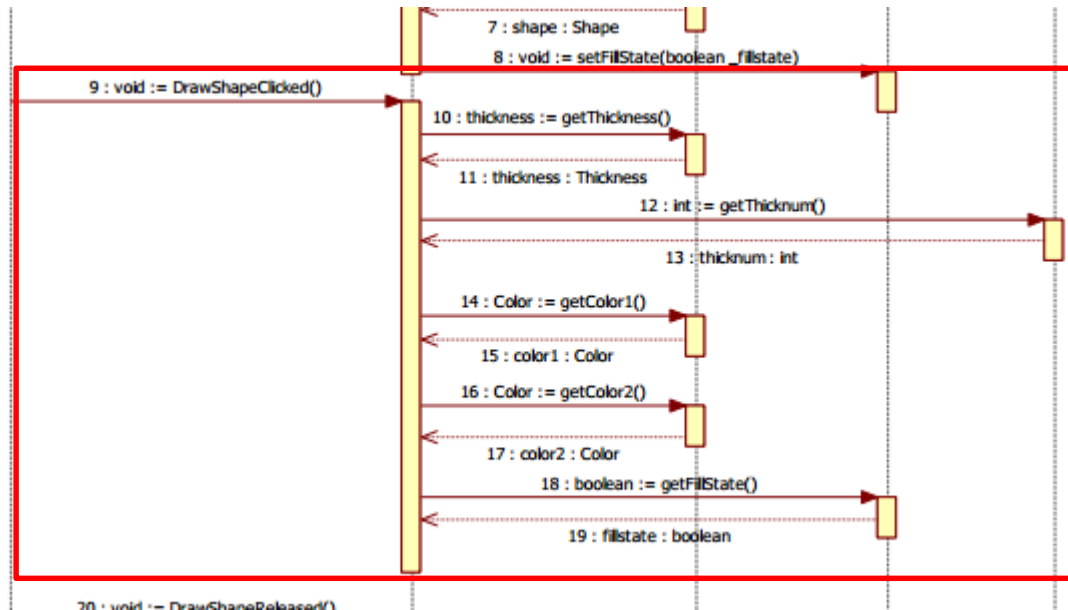
Eclipse TPTP – 도형 그리기 (cont.)



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

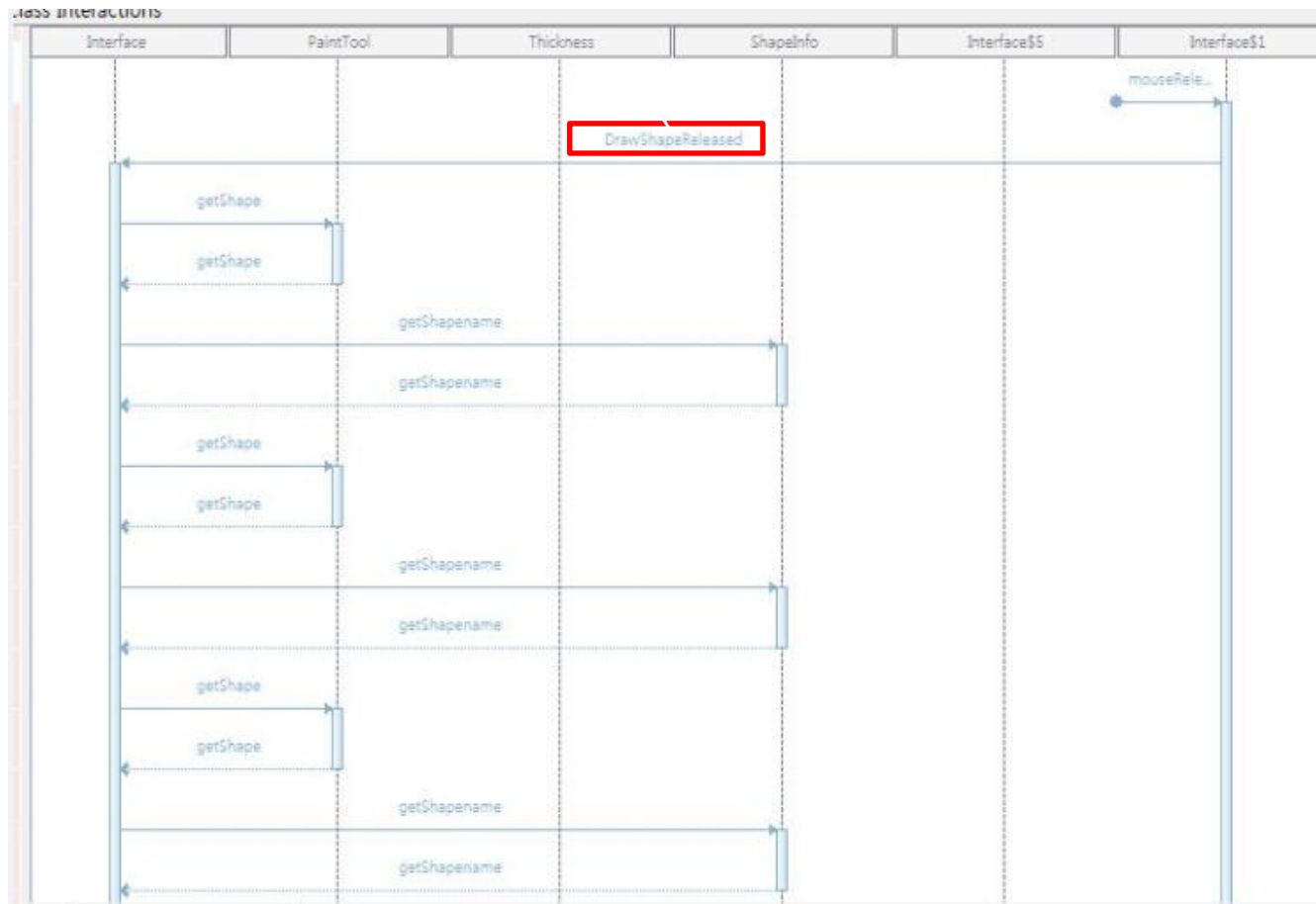
Static Analysis

Eclipse TPTP – 도형 그리기 (cont.)



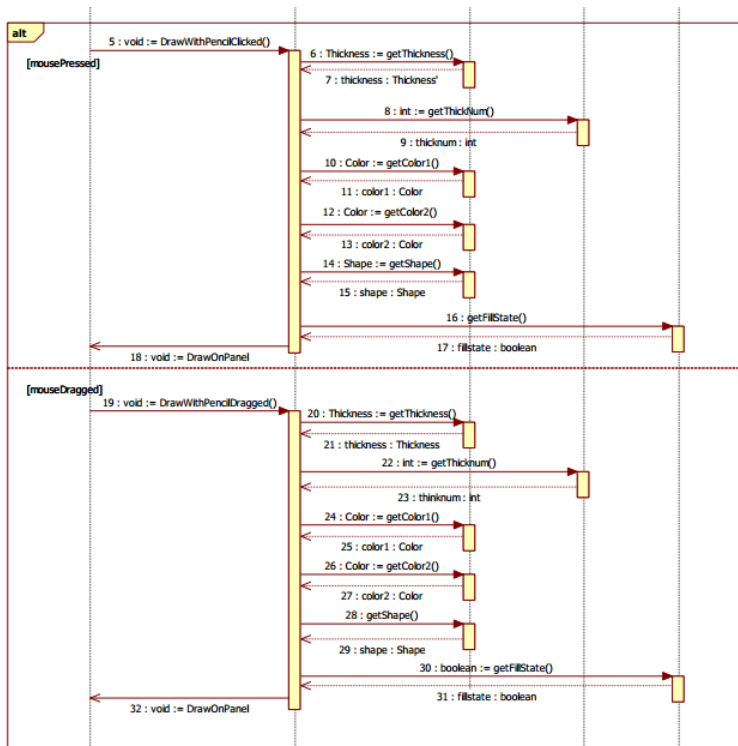
Static Analysis

Eclipse TPTP – 연필 그리기



Static Analysis

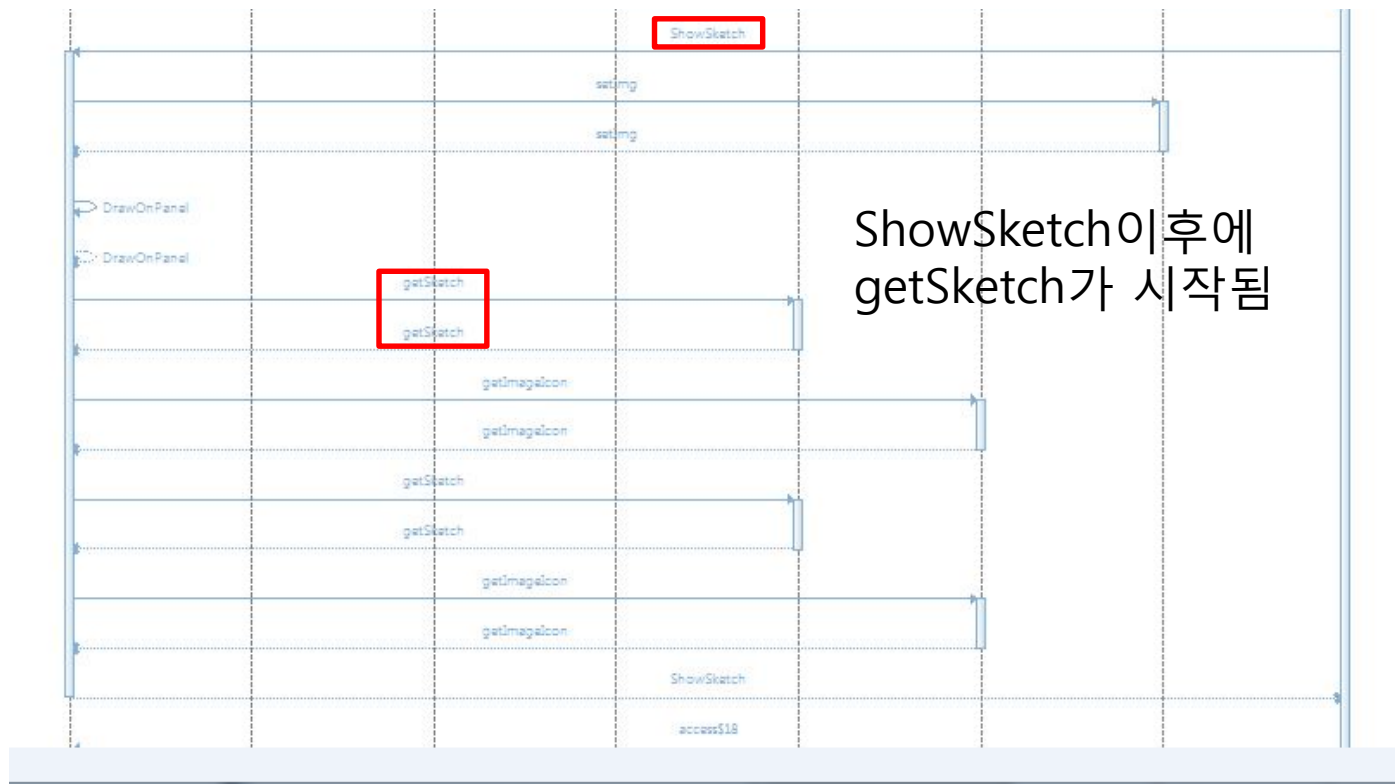
Eclipse TPTP – 연필 그리기 (cont.)



마우스 press와 dragged 부분만 정의되어 있고 release부분은 정의되어 있지 않음

Static Analysis

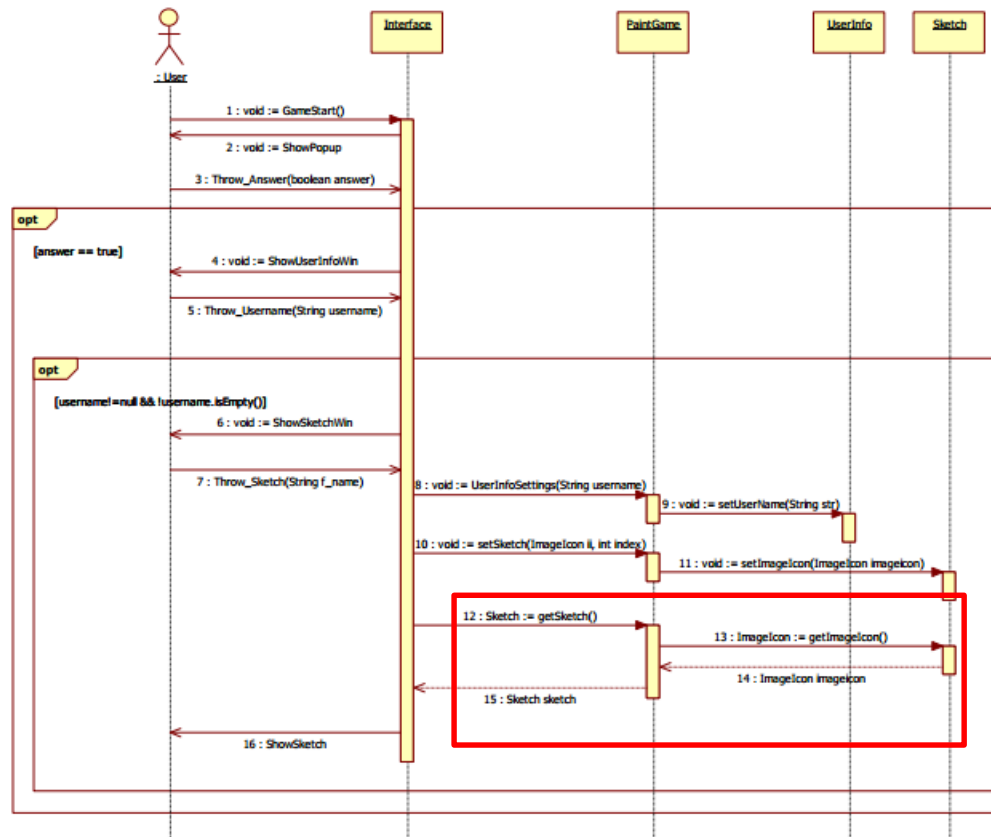
Eclipse TPTP - 게임 시작하기



Static Analysis

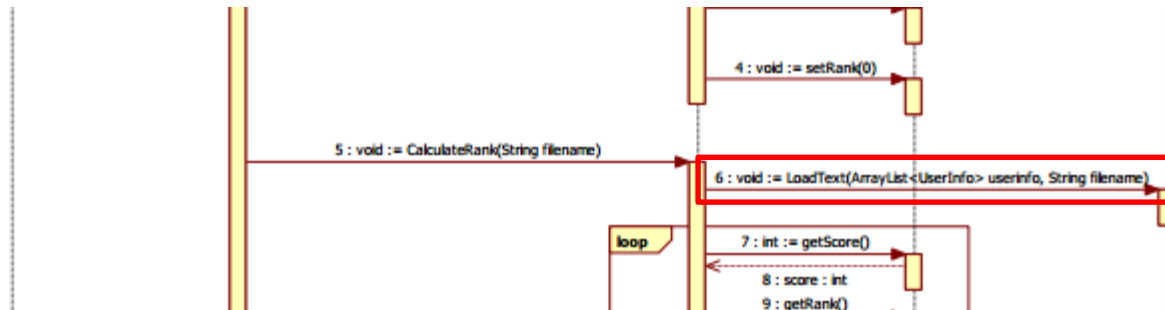
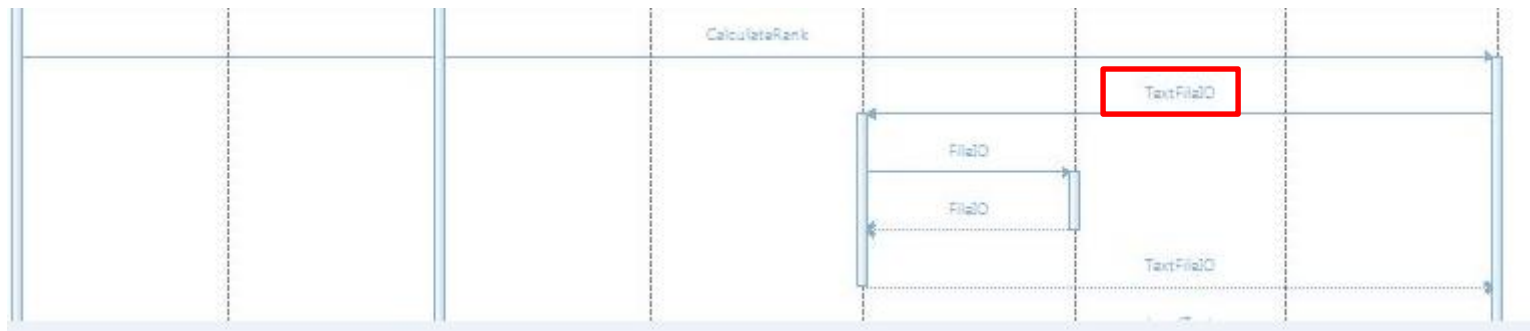
Eclipse TPTP - 게임 시작하기 (cont.)

- 게임 시작하기



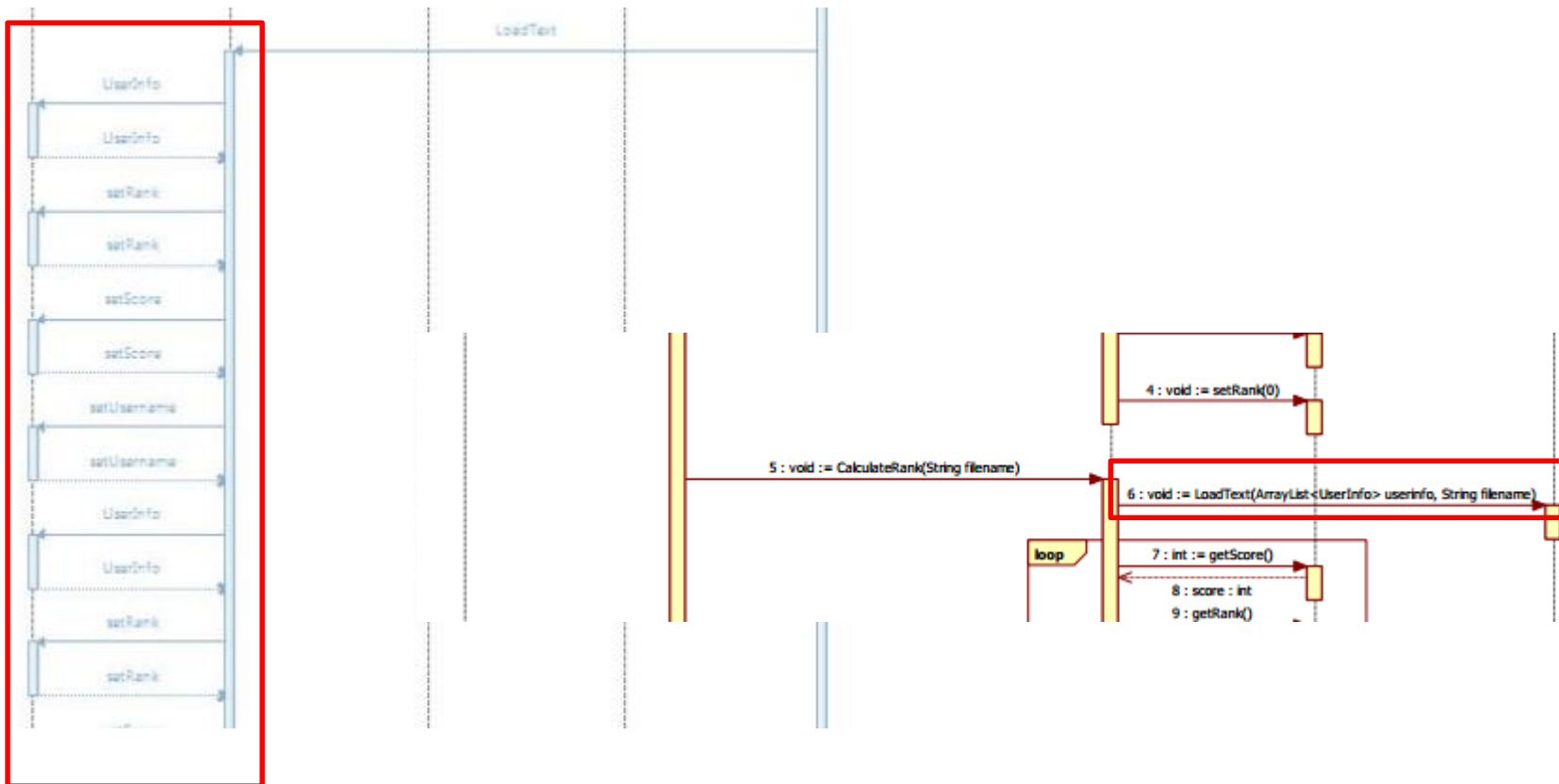
Static Analysis

Eclipse TPTP – 채점하기



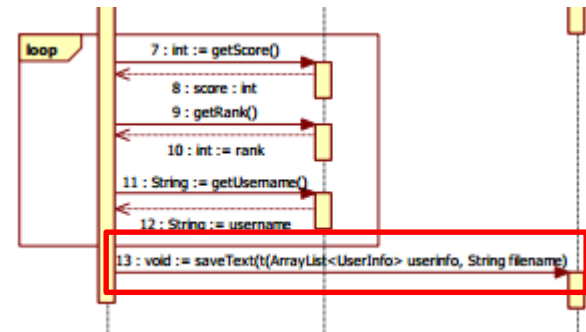
Static Analysis

Eclipse TPTP – 채점하기 (cont.)



Static Analysis

Eclipse TPTP – 채점하기 (cont.)



Reference

Reference

Reference

Address

- Sonar
 - <http://www.sonarsource.org/>
 - <http://docs.codehaus.org/>
- TPTP
 - <http://antop.tistory.com/135>
 - <http://javacan.tistory.com/entry/125>

Q & A

Q & A