Team Silver
Benjamin Trettin
Christian Shinkle
Kendall Berner
William Fuhrmann

Code Smell Report

PMD:

After running the PMD test, we have been able to identify a total of 936 problems found. Looking through the report, we found many of the problems were similar but in different locations throughout the code. Some minor problems that the PMD found that we did not see as a serious or necessary issue to address was that of java code conventions such as using proper if()....else() blocks. Another minor error that came up dozens of times was

We have come up with a list of more serious code smells that we would like to further address and see if they need to be fixed.

Code Smell: Empty Method Body
Code Involved: Many methods in testng2\src\main\java\org\testng\TestNG.java and Users\cshin\git\testng2\src\main\java\org\testng\internal\ClonedMethod.java
Why it is a problem: We still need to look into the reasons why these empty methods are being created and what purpose they hold given that they are empty. The problem is it seems to be redundant and if these methods are there for a purpose, then there should be a comment at the beginning of the class to better understand their purpose.

Code Smell: Avoid reassigning parameters
Code Involved:testng2\src\main\java\org\testng\AssertJUnit.java and testng2\src\main\java\org\testng\internal\annotations\JDK15TagFactory.java as well as spread throughout many other classes individually.
Why it is a problem: Reassigning values to incoming parameters is not recommended. Using temporary local variables instead would help with memory leak. The problem has enough instances(approximately 21) that the issue should at least be looked at.

Code Smell:A class which only has private constructors should be final
Code Involved: testng2\src\main\java\org\testng\log4testng\Logger.java and testng2\src\main\java\org\testng\internal\PackageUtils.java as well as a few others.
Why it is a problem: A class with only private constructors should be final, unless the private constructor is invoked by a inner class.

Code Smell: Overridable method 'initTestClassesAndInstances' called during **object construction**
Code Involved:testng2\src\main\java\org\testng\reporters\EmailableReporter2.java and most of the problems come from this particular class.

Why it is a problem: Calling overridable methods during construction poses a risk of invoking methods on an incompletely constructed object and can be difficult to debug. It may leave the sub-class unable to construct its superclass or forced to replicate the construction process completely within itself, losing the ability to call super(). If the default constructor contains a call to an overridable method, the subclass may be completely uninstantiable.

CPD:

The CPD report we used had a token minimum of 100 in order to really see the duplicated code that was more seriously needing fixed. After our CPD report, we found very few cases that indeed actually had duplicated code with more than 100 tokens.

```
Found a 21 line (171 tokens) duplication in the following files:
Starting at line 1090 of
C:\Users\cshin\git\testng2\src\main\java\org\testng\Assert.java
Starting at line 1121 of
C:\Users\cshin\git\testng2\src\main\java\org\testng\Assert.java
```

The 171 tokens that he is duplicating is the first main chunk of two assertEqual methods. The only difference between the two is the second method is an assertEquals Deep. What he could do in order to fix this issue is create a helper method that runs these 171 tokens and has both his two methods call that helper method at the beginning.

The second and last duplicated code that the CPD program found was a little under 20 lines.

```
Found a 17 line (108 tokens) duplication in the following files:
Starting at line 70 of
C:\Users\cshin\git\testng2\src\main\java\org\testng\reporters\jq\SuitePane
l.java
Starting at line 200 of
C:\Users\cshin\git\testng2\src\main\java\org\testng\reporters\JqReporter.j
ava
```

This code appears to be in a similar boat as the first CPD, in that there is some duplicated code within two methods that could potentially be reduced down to a helper method. The one glaring issue with doing that with this code though, is that the two method are not in the same class or even in the same package. With this little of duplicated code, in may not be necessary to create a helper method or even viable with how the packages are set up.