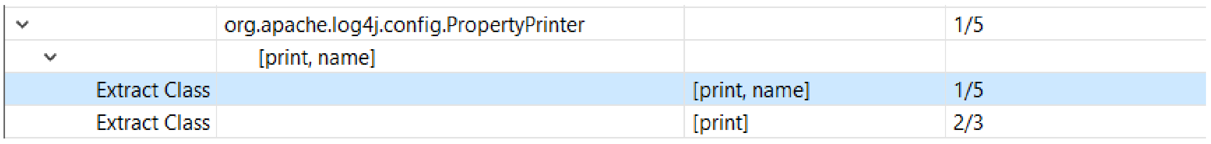
DSCI 644 Assignment 2 Akshat Garg(ag2193@rit.edu)

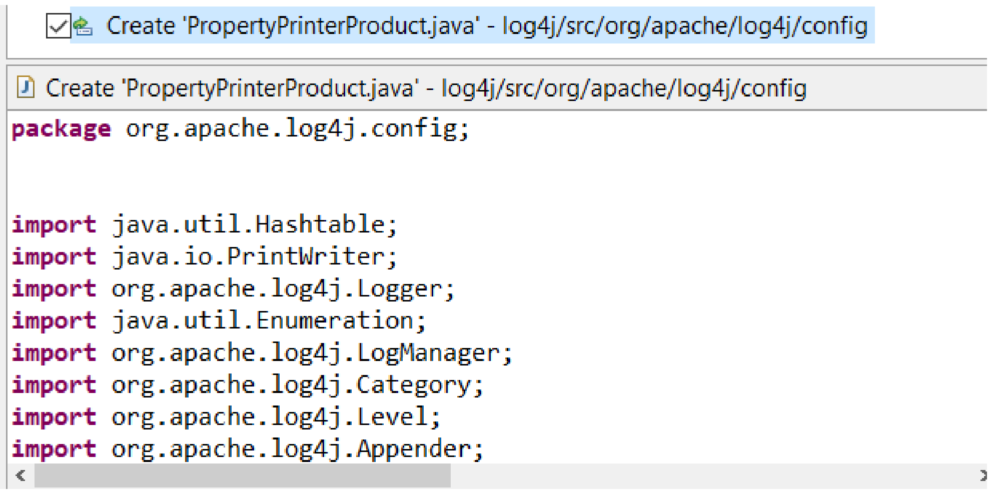
Part 1

God Class



⌄	org.apache.log4j.config.PropertyPrinter		1/5
⌄	[print, name]		
Extract Class		[print, name]	1/5
Extract Class		[print]	2/3

Flaw Instance for God class for 1st instance

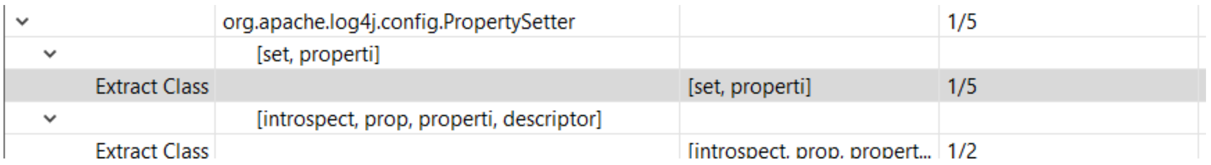


Create 'PropertyPrinterProduct.java' - log4j/src/org/apache/log4j/config

Create 'PropertyPrinterProduct.java' - log4j/src/org/apache/log4j/config

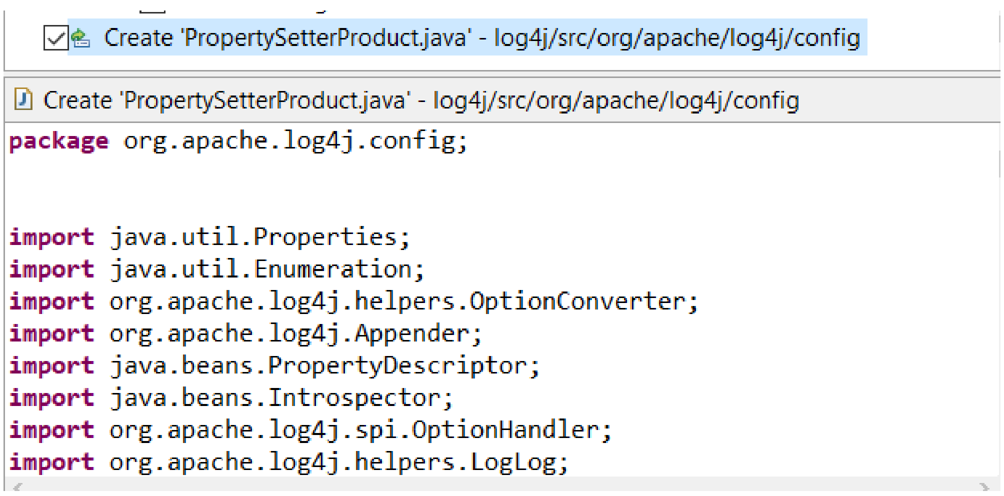
```
package org.apache.log4j.config;


import java.util.Hashtable;
import java.io.PrintWriter;
import org.apache.log4j.Logger;
import java.util.Enumeration;
import org.apache.log4j.LogManager;
import org.apache.log4j.Category;
import org.apache.log4j.Level;
import org.apache.log4j.Appender;
```

This were the refactoring operation and results for 1st instance of God class.



⌄	org.apache.log4j.config.PropertySetter		1/5
⌄	[set, properti]		
Extract Class		[set, properti]	1/5
⌄	[introspect, prop, properti, descriptor]		
Extract Class		[introspect, prop, propert...	1/2

Flaw Instance for God class for 2nd instance

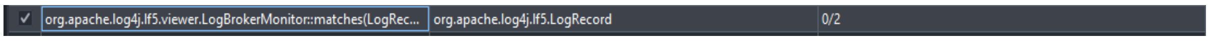

Create 'PropertySetterProduct.java' - log4j/src/org/apache/log4j/config

Create 'PropertySetterProduct.java' - log4j/src/org/apache/log4j/config

```
package org.apache.log4j.config;


import java.util.Properties;
import java.util.Enumeration;
import org.apache.log4j.helpers.OptionConverter;
import org.apache.log4j.Appender;
import java.beans.PropertyDescriptor;
import java.beans.Introspector;
import org.apache.log4j.spi.OptionHandler;
import org.apache.log4j.helpers.LogLog;
```

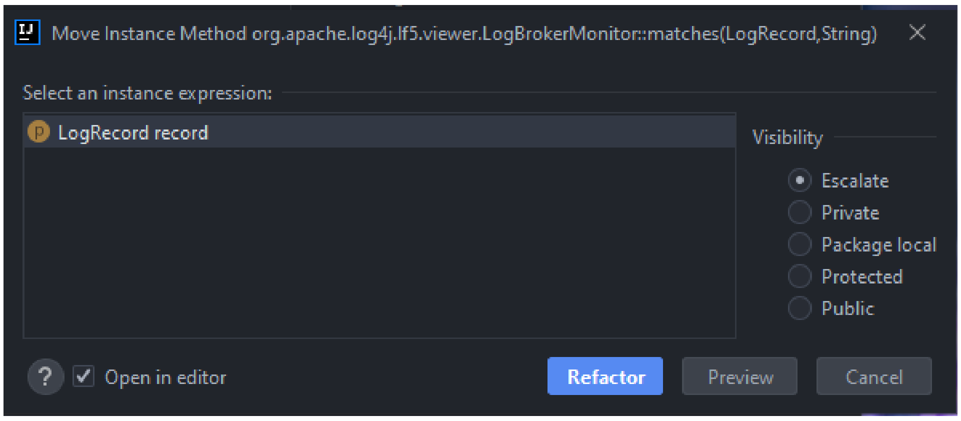
This were the refactoring operation and results for 2nd instance of God class.

Feature Envy



✓	org.apache.log4j.lf5.viewer.LogBrokerMonitor::matches(LogRec...	org.apache.log4j.lf5.LogRecord	0/2

Flaw instance for the feature envy for the 1st instance.


Move Instance Method org.apache.log4j.lf5.viewer.LogBrokerMonitor::matches(LogRecord,String)

Select an instance expression:

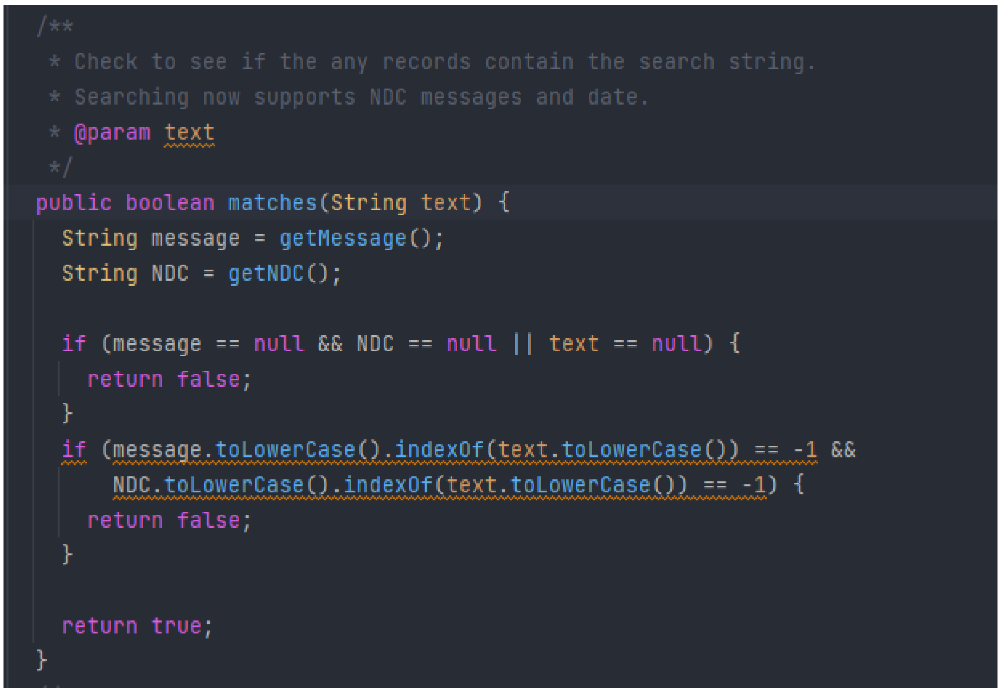
LogRecord record

Visibility

[✓ Escalate] Private Package local Protected Public

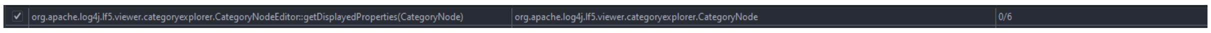
✓ Open in editor

Refactoring operation that was performed on 1st instance.



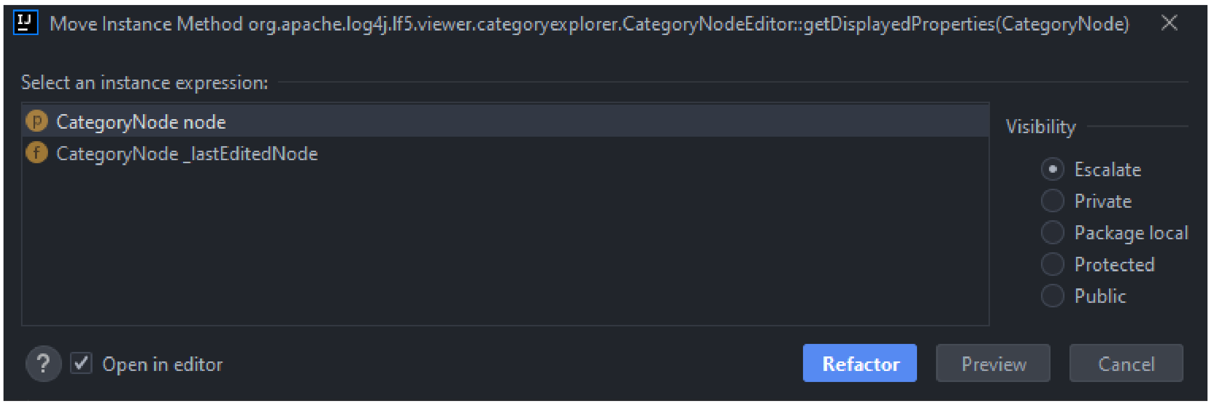
```java
/**
 * Check to see if the any records contain the search string.
 * Searching now supports NDC messages and date.
 * @param text
 */
public boolean matches(String text) {
  String message = getMessage();
  String NDC = getNDC();

  if (message == null && NDC == null || text == null) {
    return false;
  }
  if (message.toLowerCase().indexOf(text.toLowerCase()) == -1 &&
      NDC.toLowerCase().indexOf(text.toLowerCase()) == -1) {
    return false;
  }

  return true;
}
```

After refactoring was completed, here is the changes in code for 1st instance.



✓ org.apache.log4j.lf5.viewer.categoryexplorer.CategoryNodeEditor::getDisplayedProperties(CategoryNode) | org.apache.log4j.lf5.viewer.categoryexplorer.CategoryNode | 0/6

Flaw instance for the feature envy for the 2nd instance.



Move Instance Method org.apache.log4j.lf5.viewer.categoryexplorer.CategoryNodeEditor::getDisplayedProperties(CategoryNode)

Select an instance expression:

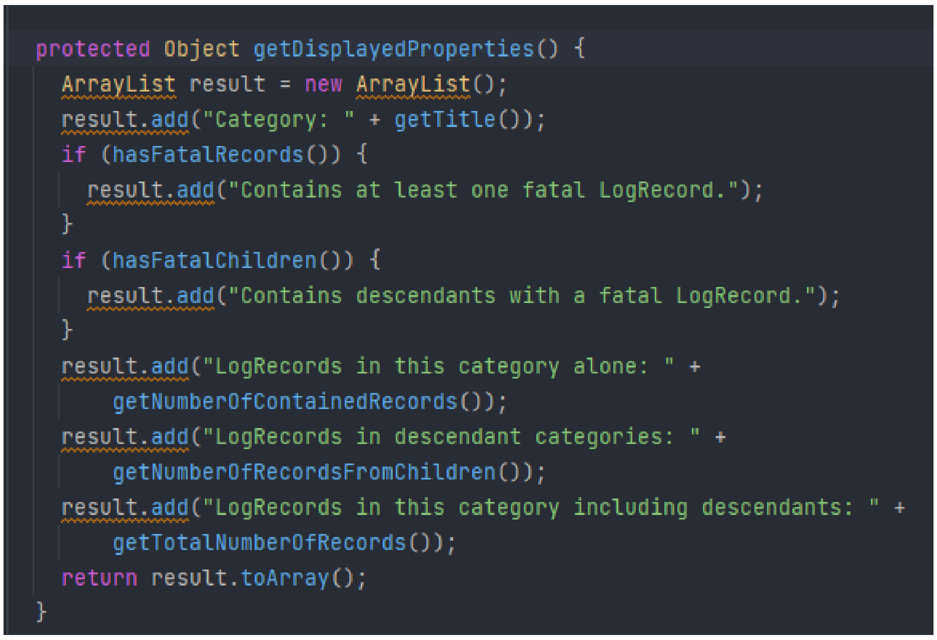
- CategoryNode node
- CategoryNode _lastEditedNode

Visibility

- [✓ Escalate]
- Private
- Package local
- Protected
- Public

✓ Open in editor

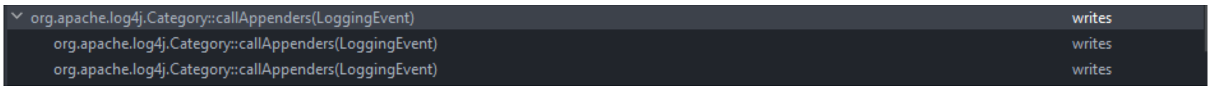
Refactoring operation that was performed on 2nd instance. It was performed on CategoryNode node.



```java
protected Object getDisplayedProperties() {
    ArrayList result = new ArrayList();
    result.add("Category: " + getTitle());
    if (hasFatalRecords()) {
      result.add("Contains at least one fatal LogRecord.");
    }
    if (hasFatalChildren()) {
      result.add("Contains descendants with a fatal LogRecord.");
    }
    result.add("LogRecords in this category alone: " +
        getNumberOfContainedRecords());
    result.add("LogRecords in descendant categories: " +
        getNumberOfRecordsFromChildren());
    result.add("LogRecords in this category including descendants: " +
        getTotalNumberOfRecords());
    return result.toArray();
}
```

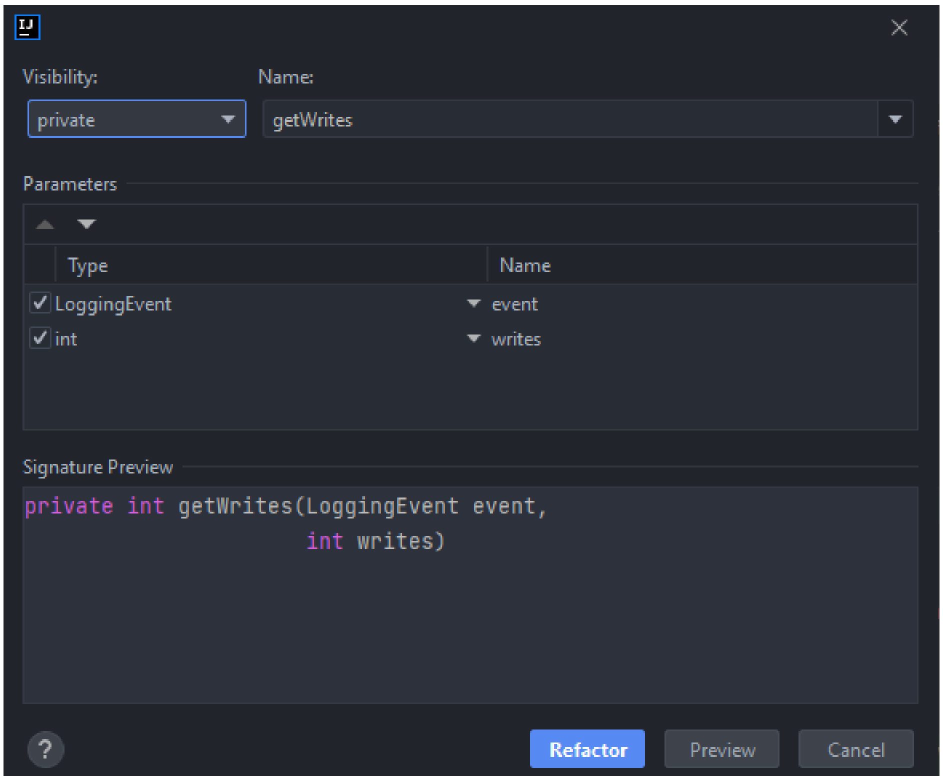
After refactoring was completed, here is the changes in code for 2nd instance.

Long Method



```
org.apache.log4j.Category::callAppenders(LoggingEvent)        writes
    org.apache.log4j.Category::callAppenders(LoggingEvent)    writes
    org.apache.log4j.Category::callAppenders(LoggingEvent)    writes
```

Flaw instance for the Long Method for the 1st instance. It was performed on 1st one.



Visibility: private

Name: getWrites

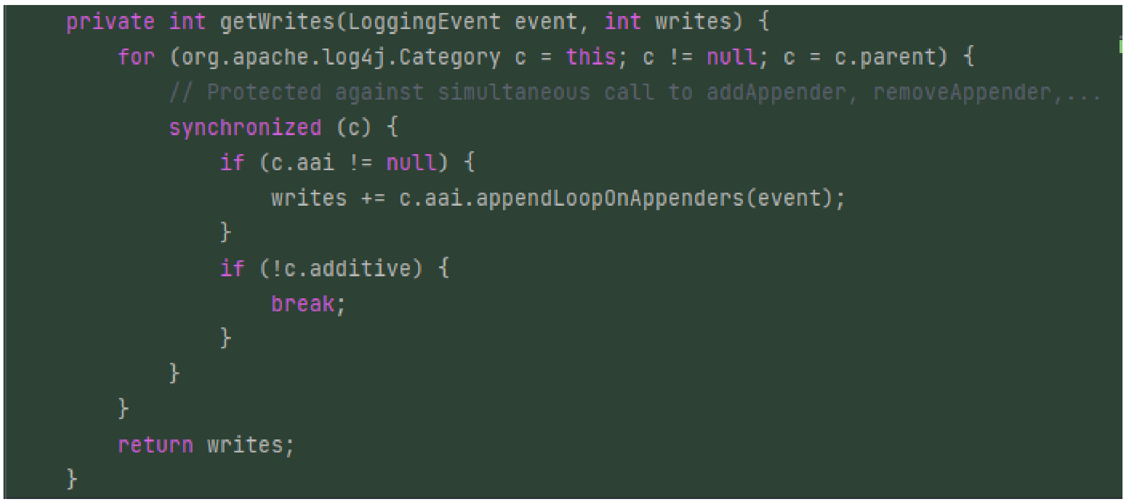
Parameters

Type	Name
✓ LoggingEvent	event
✓ int	writes

Signature Preview

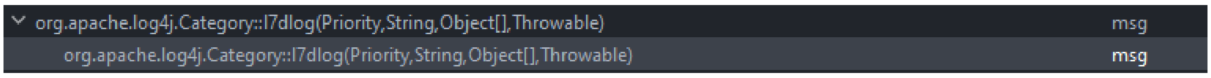
```java
private int getWrites(LoggingEvent event,
                      int writes)
```

Refactoring operation that was performed on 1st instance.



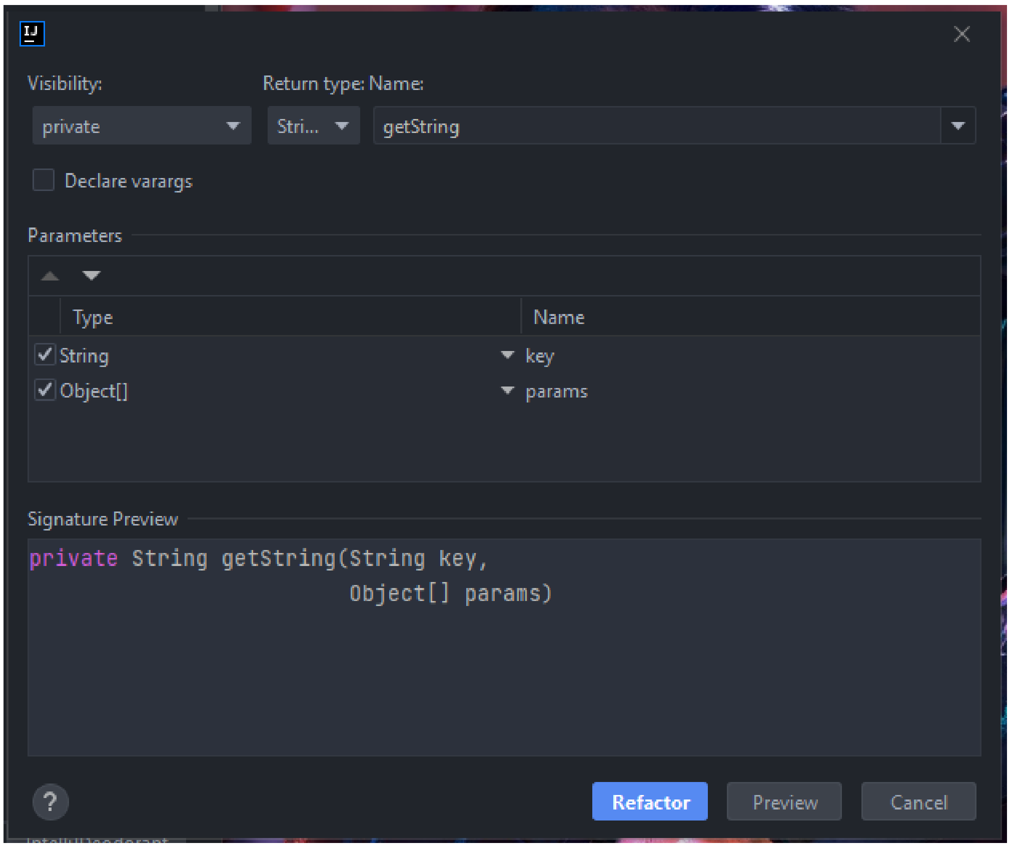
```
private int getWrites(LoggingEvent event, int writes) {
    for (org.apache.log4j.Category c = this; c != null; c = c.parent) {
        // Protected against simultaneous call to addAppender, removeAppender,...
        synchronized (c) {
            if (c.aai != null) {
                writes += c.aai.appendLoopOnAppenders(event);
            }
            if (!c.additive) {
                break;
            }
        }
    }
    return writes;
}
```

After refactoring was completed, here is the changes in code for 1st instance.



```
org.apache.log4j.Category::l7dlog(Priority,String,Object[],Throwable)    msg
    org.apache.log4j.Category::l7dlog(Priority,String,Object[],Throwable)    msg
```

Flaw instance for the Long Method for the 2nd instance.



Visibility: private
Return type: Stri...
Name: getString
Declare varargs

Parameters

Type	Name
✓ String	key
✓ Object[]	params

Signature Preview

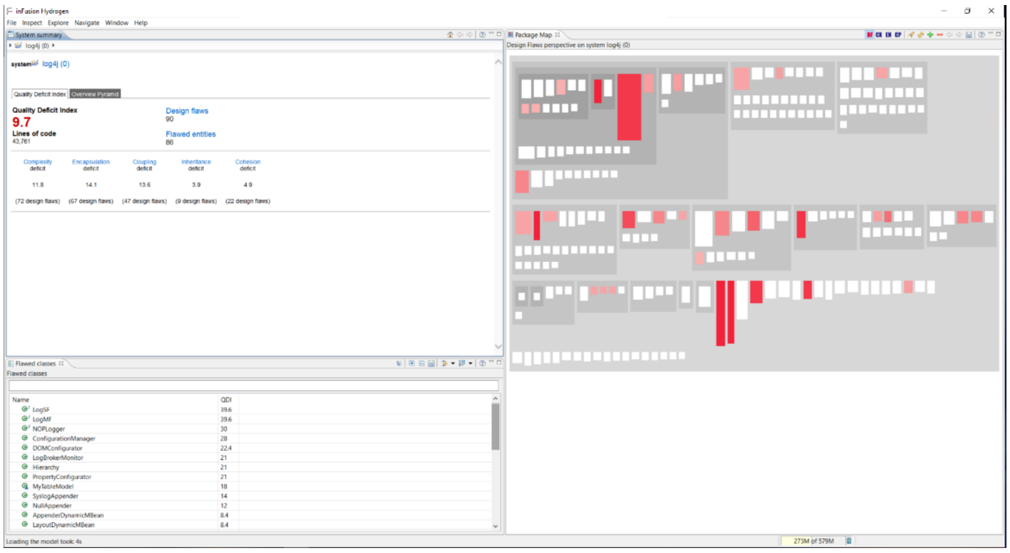
```
private String getString(String key,
                         Object[] params)
```

Refactoring operation that was performed on 2nd instance.

```
private String getString(String key, Object[] params) {
    String pattern = getResourceBundleString(key);
    String msg;
    if (pattern == null)
        msg = key;
    else
        msg = java.text.MessageFormat.format(pattern, params);
    return msg;
}
```

After refactoring was completed, here is the changes in code for 2nd instance.

inFusion



Design Flaws perspective on system log4j (0)

system log4j (0)

Quality Deficit Index | Overview Pyramid

Quality Deficit Index
9.7
Lines of code
43,761

Design flaws
90
Flawed entities
86

Complexity deficit	Encapsulation deficit	Coupling deficit	Inheritance deficit	Cohesion deficit
11.8	14.1	13.6	3.9	4.9
(72 design flaws)	(67 design flaws)	(47 design flaws)	(9 design flaws)	(22 design flaws)

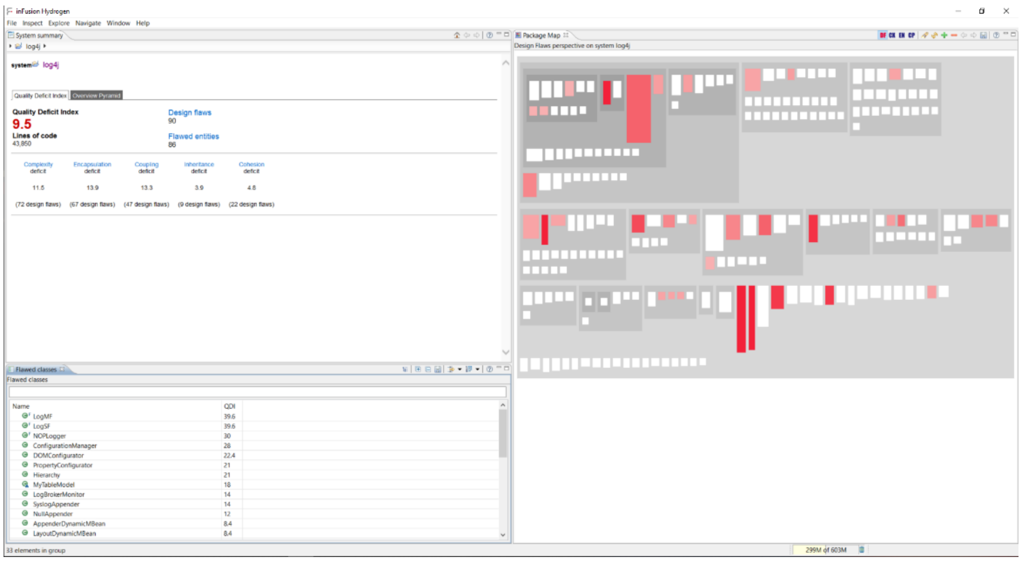
Flawed classes

Name	QDI
LogSF	39.6
LogMF	39.6
NOPLogger	30
ConfigurationManager	28
DOMConfigurator	22.4
LogBrokerMonitor	21
Hierarchy	21
PropertyConfigurator	21
MyTableModel	18
SyslogAppender	14
NullAppender	12
AppenderDynamicMBean	8.4
LayoutDynamicMBean	8.4

Loading the model took: 4s

273M of 579M

Before the refactoring applied, we had an overall quality deficit index as 9.7.



Design Flaws perspective on system log4j

system log4j

Quality Deficit Index | Overview Pyramid

Quality Deficit Index
9.5
Lines of code
43,850

Design flaws
90
Flawed entities
86

Complexity deficit	Encapsulation deficit	Coupling deficit	Inheritance deficit	Cohesion deficit
11.5	13.9	13.3	3.9	4.8
(72 design flaws)	(67 design flaws)	(47 design flaws)	(9 design flaws)	(22 design flaws)

Flawed classes

Name	QDI
LogMF	39.6
LogSF	39.6
NOPLogger	30
ConfigurationManager	28
DOMConfigurator	22.4
PropertyConfigurator	21
Hierarchy	21
MyTableModel	18
LogBrokerMonitor	14
SyslogAppender	14
NullAppender	12
AppenderDynamicMBean	8.4
LayoutDynamicMBean	8.4

33 elements in group

299M of 603M

After the refactoring applied, we had an overall quality deficit index as 9.5.

JDeodorant

About the experience with using JDeodorant, it was really positive, it really made the job easy to refactor as it auto refactors for us. Only flaw I faced was long methods were only available in IntelliJ and God Class were only available in Eclipse.