CSCI 5308
Advanced Software Development Concepts

ASSIGNMENT - 2

Banner ID: B00952865

Name: Aditya Maheshbhai Purohit

GitHub - My Forked Repo Link: https://github.com/adityap27/JSON-java

Table of Contents

Choose a Java-based open-source repository.

For this assignment, I have used the same repository as of Assignment-1.

All the repository conditions of assignment-1 are re-shown in this assignment below, for convenience:

I had selected this repository https://github.com/stleary/JSON-java [1]

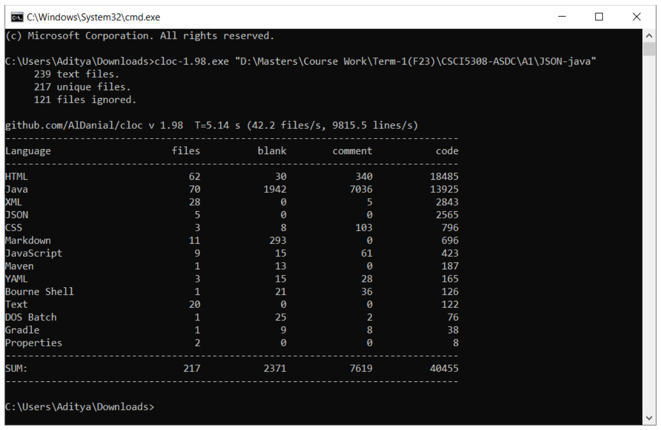
It followed all the mentioned assignment conditions (as of commit: 783d298f990c75fa2de4f458d1647cbf8e46a858)

Condition 1: It must be a maven or gradle-based project.

Answer: Yes, it used both maven and gradle. I will be using maven primarily.

Condition 2: It must have at least 10,000 lines of code

Answer: Yes, it has 13,925 Lines of java code as calculated by "cloc" tool [2].

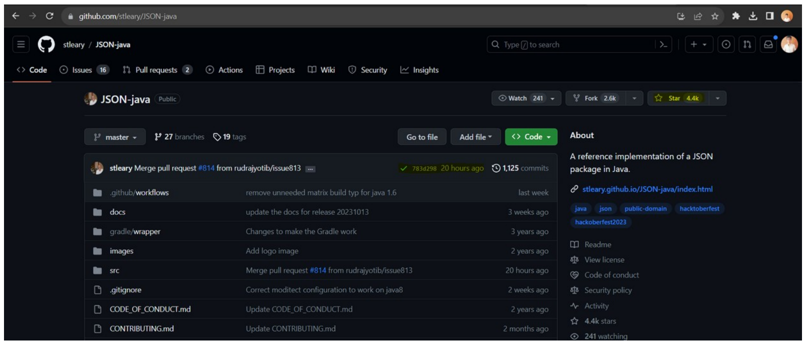


```
(c) Microsoft Corporation. All rights reserved.

C:\Users\Aditya\Downloads>cloc-1.98.exe "D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A1\JSON-java"
     239 text files.
     217 unique files.
     121 files ignored.

github.com/AlDanial/cloc v 1.98  T=5.14 s (42.2 files/s, 9815.5 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
HTML                            62             30            340          18485
Java                            70           1942           7036          13925
XML                             28              0              5           2843
JSON                             5              0              0           2565
CSS                              3              8            103            796
Markdown                        11            293              0            696
JavaScript                       9             15             61            423
Maven                            1             13              0            187
YAML                             3             15             28            165
Bourne Shell                     1             21             36            126
Text                            20              0              0            122
DOS Batch                        1             25              2             76
Gradle                           1              9              8             38
Properties                       2              0              0              8
-------------------------------------------------------------------------------
SUM:                           217           2371           7619          40455
-------------------------------------------------------------------------------

C:\Users\Aditya\Downloads>
```

Figure 1: Lines of java code calculated using cloc tool [2].

Condition 3: It must have at least 50 stars.

Answer: Yes, it has 4.4k stars as shown in **figure 2**.


github.com/stleary/JSON-java

JSON-java Public — Watch 241 · Fork 2.6k · Star 4.4k

master · 27 branches · 19 tags

stleary Merge pull request #814 from rudrajyotib/issue813 — 783d298 20 hours ago — 1,125 commits

.github/workflows	remove unneeded matrix build typ for java 1.6	last week
docs	update the docs for release 20231013	3 weeks ago
gradle/wrapper	Changes to make the Gradle work	3 years ago
images	Add logo image	2 years ago
src	Merge pull request #814 from rudrajyotib/issue813	20 hours ago
.gitignore	Correct moditect configuration to work on java8	2 weeks ago
CODE_OF_CONDUCT.md	Update CODE_OF_CONDUCT.md	2 years ago
CONTRIBUTING.md	Update CONTRIBUTING.md	2 months ago

About

A reference implementation of a JSON package in Java.

stleary.github.io/JSON-java/index.html

java · json · public-domain · hacktoberfest · hackoberfest2023

4.4k stars · 241 watching

Figure 2: Stars of the repository and activeness [1].

Condition 4: It must have tests written using the JUnit framework

Answer: Yes, it uses Junit 4.13.2 and it is mentioned in the pom.xml.

Condition 5: It must not be a tutorial or example repository

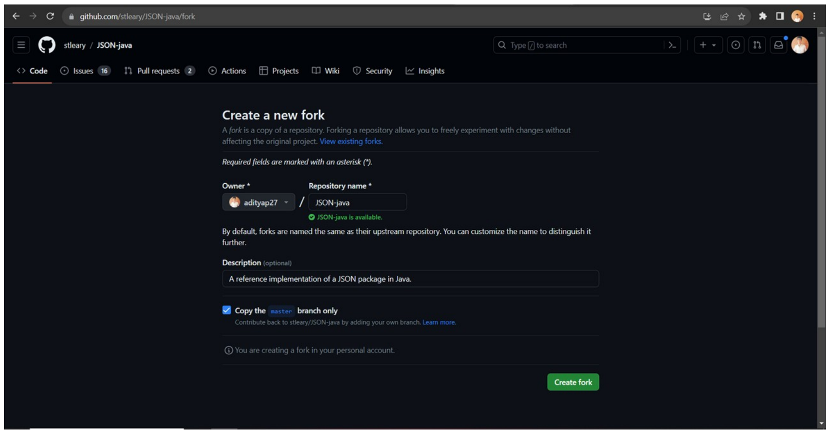
Answer: It is not a tutorial or example repository.

Condition 6: It must be active (at least one commit in the past one year)

Answer: Yes, it is active, as shown in **figure 2**.

Later steps:

1. I forked that repository in my GitHub account. Link: https://github.com/adityap27/JSON-java


github.com/stleary/JSON-java/fork

Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. View existing forks.

Required fields are marked with an asterisk (*).

Owner *: adityap27 / Repository name *: JSON-java

JSON-java is available.

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional): A reference implementation of a JSON package in Java.

[✓] Copy the `master` branch only

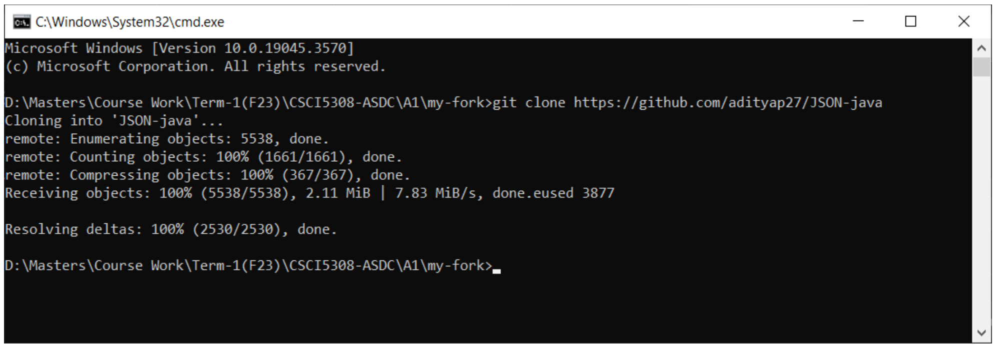
Contribute back to stleary/JSON-java by adding your own branch. Learn more.

You are creating a fork in your personal account.

Create fork

Figure 3: Fork of JSON-java repository [1].

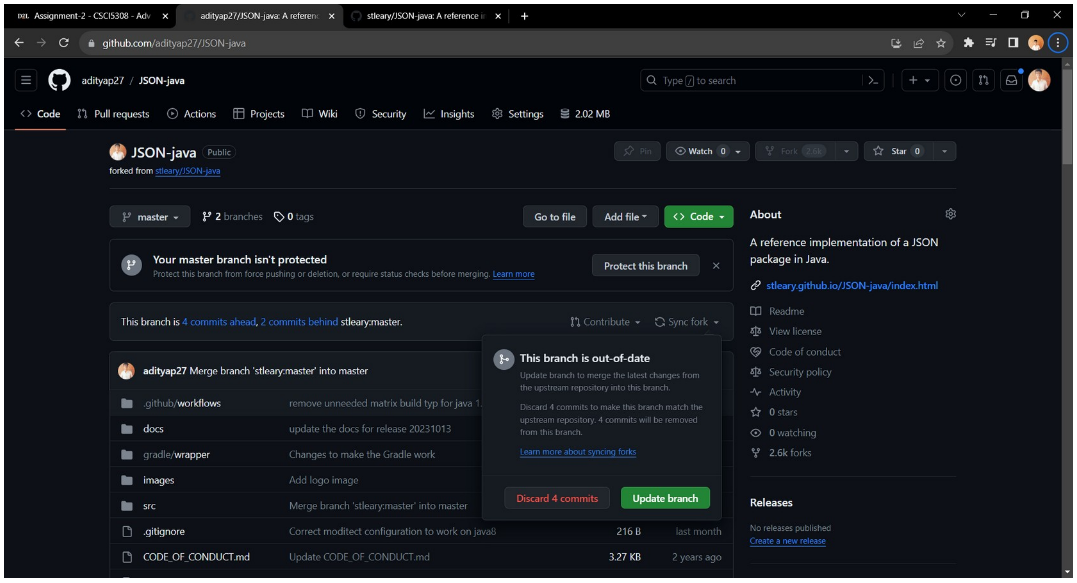
2. I cloned the forked repository in my local machine.



```
Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A1\my-fork>git clone https://github.com/adityap27/JSON-java
Cloning into 'JSON-java'...
remote: Enumerating objects: 5538, done.
remote: Counting objects: 100% (1661/1661), done.
remote: Compressing objects: 100% (367/367), done.
Receiving objects: 100% (5538/5538), 2.11 MiB | 7.83 MiB/s, done.eused 3877

Resolving deltas: 100% (2530/2530), done.

D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A1\my-fork>
```

Figure 4: Clone forked repository into local machine.

3. I used the Sync fork feature to get the new commits from the original repository, as it had new commits after I completed my Assignment-1.



Your master branch isn't protected

Protect this branch from force pushing or deletion, or require status checks before merging. Learn more

This branch is 4 commits ahead, 2 commits behind stleary:master.

This branch is out-of-date

Update branch to merge the latest changes from the upstream repository into this branch.

Discard 4 commits to make this branch match the upstream repository. 4 commits will be removed from this branch.

Learn more about syncing forks

adityap27 Merge branch 'stleary:master' into master

.github/workflows	remove unneeded matrix build typ for java 1		
docs	update the docs for release 20231013		
gradle/wrapper	Changes to make the Gradle work		
images	Add logo image		
src	Merge branch 'stleary:master' into master		
.gitignore	Correct moditect configuration to work on java8	216 B	last month
CODE_OF_CONDUCT.md	Update CODE_OF_CONDUCT.md	3.27 KB	2 years ago

About

A reference implementation of a JSON package in Java.

stleary.github.io/JSON-java/index.html

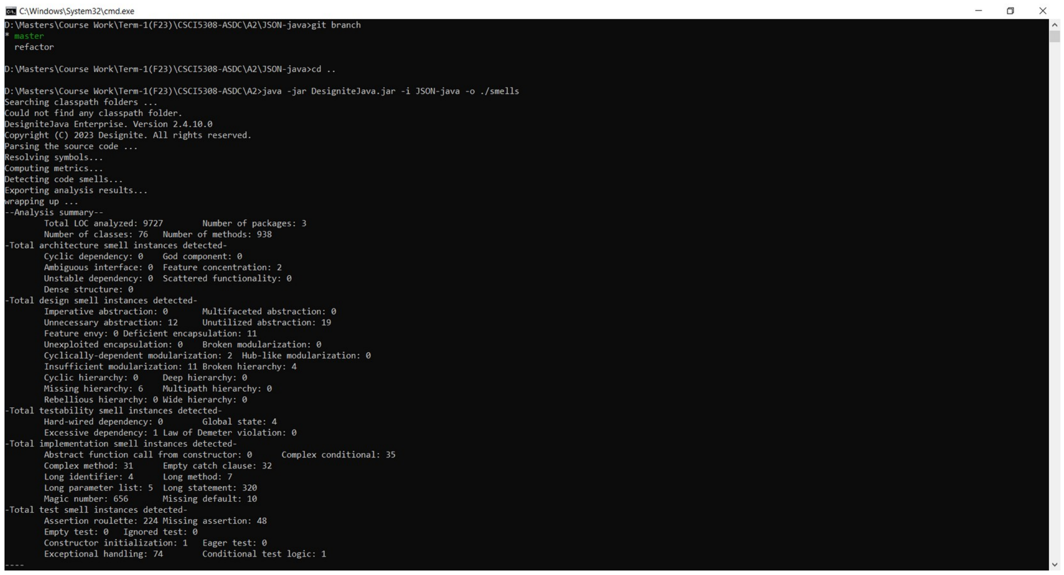
No releases published

Create a new release

Figure 5: Sync with the original repository.

Task 1: Identify Smells

I ran DesigniteJava Enterprise Version 2.4.10.0, to get the csv files of Implementation and Design smells.



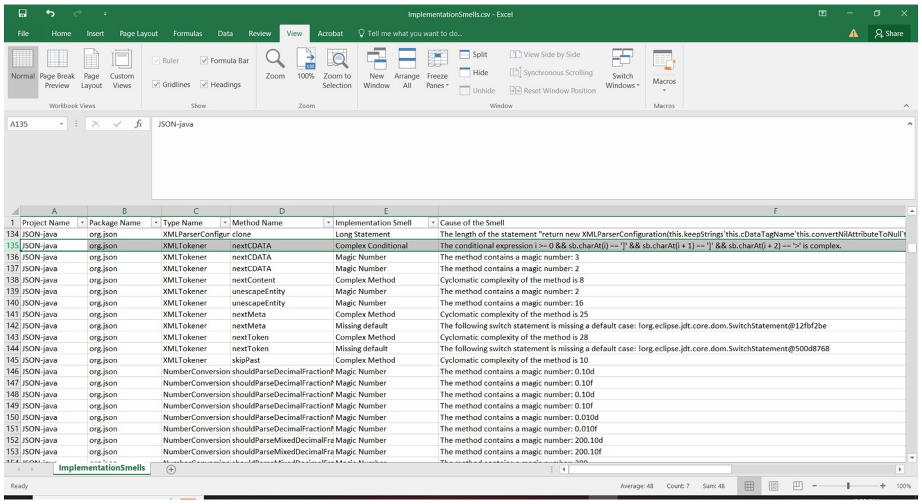
```
D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A2\JSON-java>git branch
* master
  refactor

D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A2\JSON-java>cd ..

D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A2>java -jar DesigniteJava.jar -i JSON-java -o ./smells
Searching classpath folders ...
Could not find any classpath folder.
DesigniteJava Enterprise. Version 2.4.10.0
Copyright (C) 2023 Designite. All rights reserved.
Parsing the source code ...
Resolving symbols...
Computing metrics...
Detecting code smells...
Exporting analysis results...
wrapping up ...
--Analysis summary--
        Total LOC analyzed: 9727        Number of packages: 3
        Number of classes: 76   Number of methods: 938
-Total architecture smell instances detected-
        Cyclic dependency: 0    God component: 0
        Ambiguous interface: 0  Feature concentration: 2
        Unstable dependency: 0  Scattered functionality: 0
        Dense structure: 0
-Total design smell instances detected-
        Imperative abstraction: 0       Multifaceted abstraction: 0
        Unnecessary abstraction: 12     Unutilized abstraction: 19
        Feature envy: 0 Deficient encapsulation: 11
        Unexploited encapsulation: 0    Broken modularization: 0
        Cyclically-dependent modularization: 2  Hub-like modularization: 0
        Insufficient modularization: 11 Broken hierarchy: 4
        Cyclic hierarchy: 0     Deep hierarchy: 0
        Missing hierarchy: 6    Multipath hierarchy: 0
        Rebellious hierarchy: 0 Wide hierarchy: 0
-Total testability smell instances detected-
        Hard-wired dependency: 0        Global state: 4
        Excessive dependency: 1 Law of Demeter violation: 0
-Total implementation smell instances detected-
        Abstract function call from constructor: 0      Complex conditional: 35
        Complex method: 31      Empty catch clause: 32
        Long identifier: 4      Long method: 7
        Long parameter list: 5  Long statement: 320
        Magic number: 656       Missing default: 10
-Total test smell instances detected-
        Assertion roulette: 224 Missing assertion: 48
        Empty test: 0   Ignored test: 0
        Constructor initialization: 1   Eager test: 0
        Exceptional handling: 74        Conditional test logic: 1
----
```

Figure 6: Execute DesigniteJava tool on master branch [3].

Set-1: Implementation Smells

True Positive:

Complex conditional: The conditional expression i >= 0 && sb.charAt(i) == ']' && sb.charAt(i + 1) == ']' && sb.charAt(i + 2) == '>' is complex.

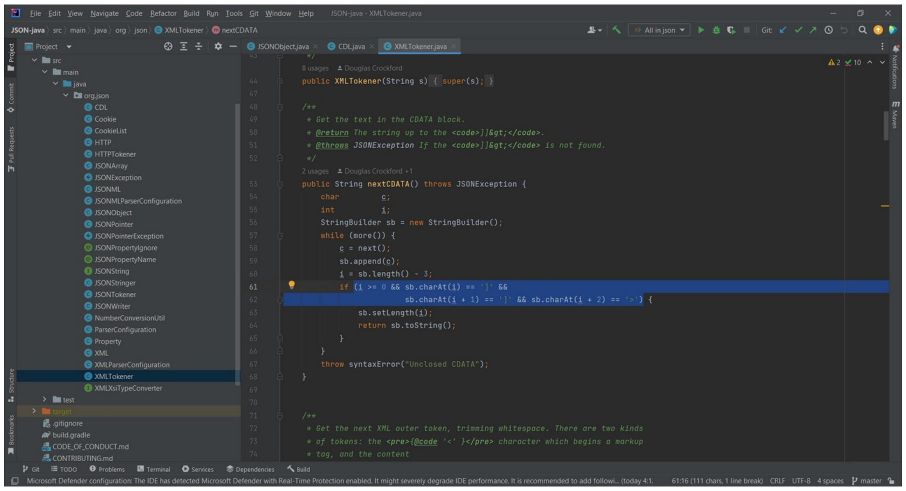


	Project Name	Package Name	Type Name	Method Name	Implementation Smell	Cause of the Smell
134	JSON-java	org.json	XMLParserConfigur	clone	Long Statement	The length of the statement "return new XMLParserConfiguration(this.keepStrings`this.cDataTagName`this.convertNilAttributeToNull`t
135	JSON-java	org.json	XMLTokener	nextCDATA	Complex Conditional	The conditional expression i >= 0 && sb.charAt(i) == ']' && sb.charAt(i + 1) == ']' && sb.charAt(i + 2) == '>' is complex.
136	JSON-java	org.json	XMLTokener	nextCDATA	Magic Number	The method contains a magic number: 3
137	JSON-java	org.json	XMLTokener	nextCDATA	Magic Number	The method contains a magic number: 2
138	JSON-java	org.json	XMLTokener	nextContent	Complex Method	Cyclomatic complexity of the method is 8
139	JSON-java	org.json	XMLTokener	unescapeEntity	Magic Number	The method contains a magic number: 2
140	JSON-java	org.json	XMLTokener	unescapeEntity	Magic Number	The method contains a magic number: 16
141	JSON-java	org.json	XMLTokener	nextMeta	Complex Method	Cyclomatic complexity of the method is 25
142	JSON-java	org.json	XMLTokener	nextMeta	Missing default	The following switch statement is missing a default case: !org.eclipse.jdt.core.dom.SwitchStatement@12fbf2be
143	JSON-java	org.json	XMLTokener	nextToken	Complex Method	Cyclomatic complexity of the method is 28
144	JSON-java	org.json	XMLTokener	nextToken	Missing default	The following switch statement is missing a default case: !org.eclipse.jdt.core.dom.SwitchStatement@500d8768
145	JSON-java	org.json	XMLTokener	skipPast	Complex Method	Cyclomatic complexity of the method is 10
146	JSON-java	org.json	NumberConversion	shouldParseDecimalFraction	Magic Number	The method contains a magic number: 0.10d
147	JSON-java	org.json	NumberConversion	shouldParseDecimalFraction	Magic Number	The method contains a magic number: 0.10f
148	JSON-java	org.json	NumberConversion	shouldParseDecimalFraction	Magic Number	The method contains a magic number: 0.10d
149	JSON-java	org.json	NumberConversion	shouldParseDecimalFraction	Magic Number	The method contains a magic number: 0.10f
150	JSON-java	org.json	NumberConversion	shouldParseDecimalFraction	Magic Number	The method contains a magic number: 0.010d
151	JSON-java	org.json	NumberConversion	shouldParseDecimalFraction	Magic Number	The method contains a magic number: 0.010f
152	JSON-java	org.json	NumberConversion	shouldParseMixedDecimalFra	Magic Number	The method contains a magic number: 200.10d
153	JSON-java	org.json	NumberConversion	shouldParseMixedDecimalFra	Magic Number	The method contains a magic number: 200.10f

Figure 7: True positive implementation smell.

Rationale: There is a combination of 4 conditions which is hard to understand that what exactly this condition means as a whole. This condition is checking whether the index is valid (*>=0*) and whether the 3 consecutive characters in the string from given index, represent end of a CDATA tag or not. (i.e. *]]>*)

Here, there are total 4 conditions as show in figure 8, which can be grouped into a single new method to increase the readability and understandability for other developers. **For eg:** All 4 conditions can be written in a new method called **isEndOfCDATA(StringBuilder sb, int index)** which can improve the readability as the method name suggests what the condition does instead of logically trying to find the relation between 4 conditions in mind while reading the code. This new method can take the string and the starting index to check for the end tag, as arguments.

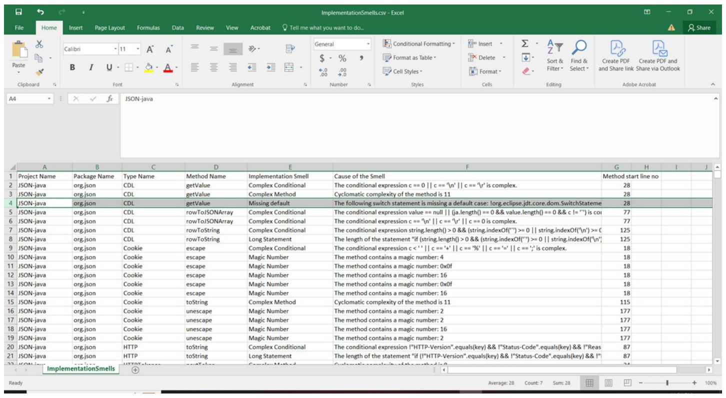


```
    public XMLTokener(String s) { super(s); }

    /**
     * Get the text in the CDATA block.
     * @return The string up to the <code>]]&gt;</code>.
     * @throws JSONException If the <code>]]&gt;</code> is not found.
     */
    public String nextCDATA() throws JSONException {
        char         c;
        int          i;
        StringBuilder sb = new StringBuilder();
        while (more()) {
            c = next();
            sb.append(c);
            i = sb.length() - 3;
            if (i >= 0 && sb.charAt(i) == ']' &&
                        sb.charAt(i + 1) == ']' && sb.charAt(i + 2) == '>') {
                sb.setLength(i);
                return sb.toString();
            }
        }
        throw syntaxError("Unclosed CDATA");
    }


    /**
     * Get the next XML outer token, trimming whitespace. There are two kinds
     * of tokens: the <pre>{@code '<' }</pre> character which begins a markup
     * tag, and the content
```

Figure 8: Code example of true positive implementation smell.

False Positive:

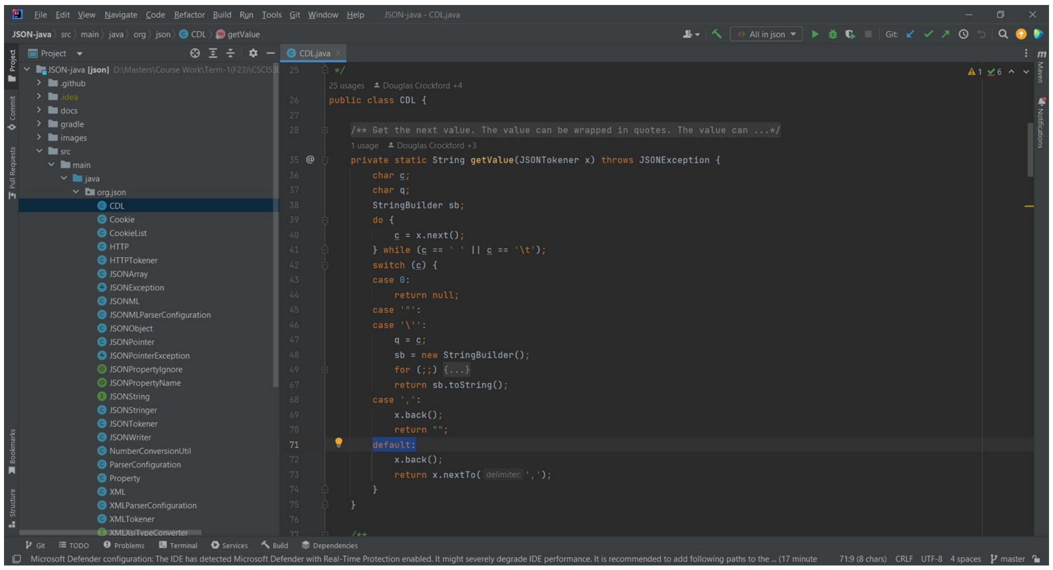
Missing default: The following switch statement is missing a default case: [! org.eclipse.jdt.core.dom.SwitchStatement@4ff17275] in org.json.CDL.**getValue**(..) method



Project Name	Package Name	Type Name	Method Name	Implementation Smell	Cause of the Smell	Method start line no
JSON-java	org.json	CDL	getValue	Complex Conditional	The conditional expression c == 0 || c == '\n' || c == '\r' is complex.	28
JSON-java	org.json	CDL	getValue	Complex Method	Cyclomatic complexity of the method is 11	28
JSON-java	org.json	CDL	getValue	Missing default	The following switch statement is missing a default case: !org.eclipse.jdt.core.dom.SwitchStateme	28
JSON-java	org.json	CDL	rowToJSONArray	Complex Conditional	The conditional expression value == null || (ja.length() == 0 && value.length() == 0 && c != '"') is co	77
JSON-java	org.json	CDL	rowToJSONArray	Complex Conditional	The conditional expression c == '\n' || c == '\r' || c == 0 is complex.	77
JSON-java	org.json	CDL	rowToString	Complex Conditional	The conditional expression string.length() > 0 && (string.indexOf('"') >= 0 || string.indexOf('\n') >= (	125
JSON-java	org.json	CDL	rowToString	Long Statement	The length of the statement "if (string.length() > 0 && (string.indexOf('"') >= 0 || string.indexOf('\n')	125
JSON-java	org.json	Cookie	escape	Complex Conditional	The conditional expression c < ' ' || c == '+' || c == '%' || c == '=' || c == ';' is complex.	18
JSON-java	org.json	Cookie	escape	Magic Number	The method contains a magic number: 4	18
JSON-java	org.json	Cookie	escape	Magic Number	The method contains a magic number: 0x0f	18
JSON-java	org.json	Cookie	escape	Magic Number	The method contains a magic number: 16	18
JSON-java	org.json	Cookie	escape	Magic Number	The method contains a magic number: 0x0f	18
JSON-java	org.json	Cookie	escape	Magic Number	The method contains a magic number: 16	18
JSON-java	org.json	Cookie	toString	Complex Method	Cyclomatic complexity of the method is 11	115
JSON-java	org.json	Cookie	unescape	Magic Number	The method contains a magic number: 2	177
JSON-java	org.json	Cookie	unescape	Magic Number	The method contains a magic number: 2	177
JSON-java	org.json	Cookie	unescape	Magic Number	The method contains a magic number: 16	177
JSON-java	org.json	Cookie	unescape	Magic Number	The method contains a magic number: 2	177
JSON-java	org.json	HTTP	toString	Complex Conditional	The conditional expression !"HTTP-Version".equals(key) && !"Status-Code".equals(key) && !"Reas	87
JSON-java	org.json	HTTP	toString	Long Statement	The length of the statement "if (!"HTTP-Version".equals(key) && !"Status-Code".equals(key) && !"	87

Figure 9: False Positive smell detected by DesigniteJava

Rationale: This implementation smell of “Missing default” is false positive, as the detected method already has a default case for switch case as per source code [org.json.CDL.getValue(..)] as show in figure 10, but the tool is still saying that it is missing default.



JSON-java - CDL.java

```java
*/
25 usages  Douglas Crockford +4
public class CDL {

    /** Get the next value. The value can be wrapped in quotes. The value can ...*/
    1 usage  Douglas Crockford +3
    private static String getValue(JSONTokener x) throws JSONException {
        char c;
        char q;
        StringBuilder sb;
        do {
            c = x.next();
        } while (c == ' ' || c == '\t');
        switch (c) {
        case 0:
            return null;
        case '"':
        case '\'':
            q = c;
            sb = new StringBuilder();
            for (;;) {...}
            return sb.toString();
        case ',':
            x.back();
            return "";
        default:
            x.back();
            return x.nextTo( delimiter: ',');
        }
    }

    /**
```

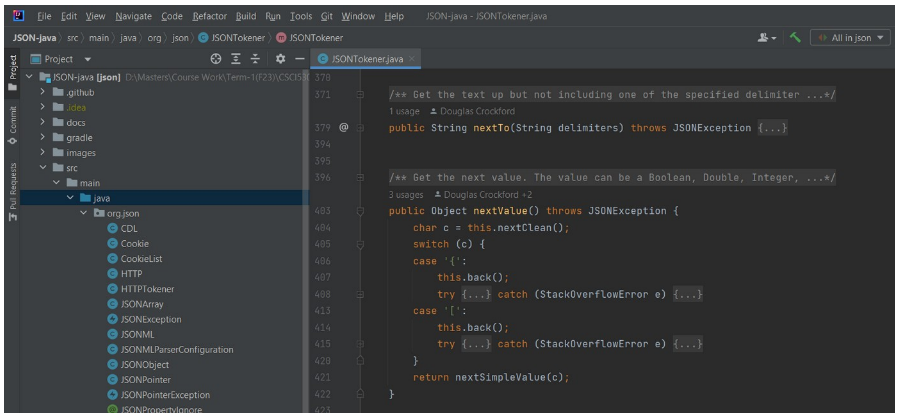
Microsoft Defender configuration: The IDE has detected Microsoft Defender with Real-Time Protection enabled. It might severely degrade IDE performance. It is recommended to add following paths to the ... (17 minute

Figure 10: Presence of default case in CDL.getValue() method's switch case.

False Negative:

Undetected Smell:

Missing Default: For switch statement in nextTo(..) method of org.json.JSONTokener Class.

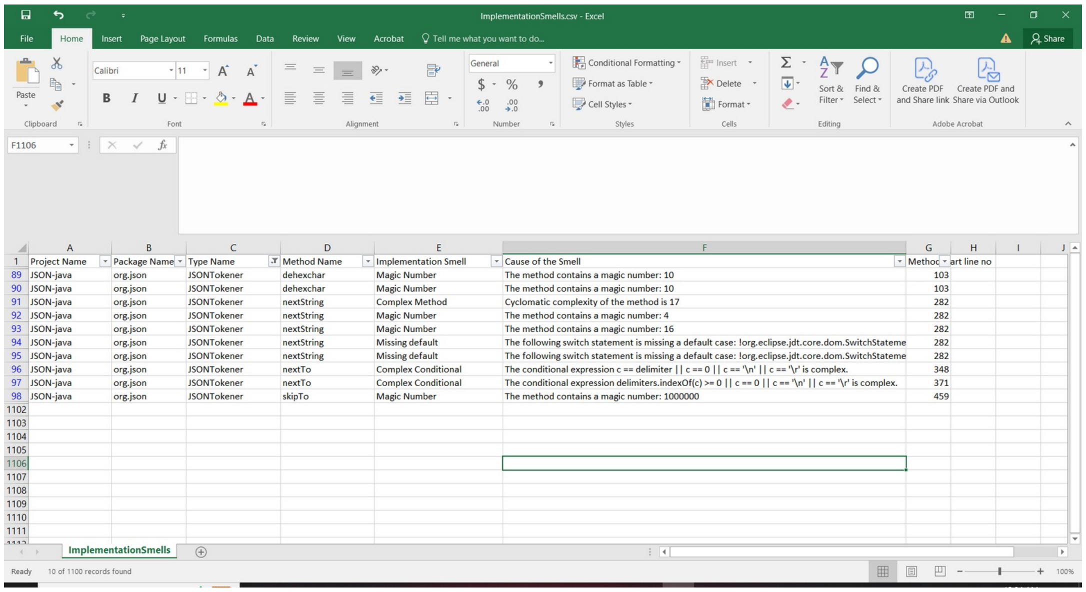


JSON-java - JSONTokener.java

```java
370
371     /** Get the text up but not including one of the specified delimiter ...*/
        1 usage  Douglas Crockford
379 @   public String nextTo(String delimiters) throws JSONException {...}
394
395
396     /** Get the next value. The value can be a Boolean, Double, Integer, ...*/
        3 usages  Douglas Crockford +2
403     public Object nextValue() throws JSONException {
404         char c = this.nextClean();
405         switch (c) {
406         case '{':
407             this.back();
408             try {...} catch (StackOverflowError e) {...}
413         case '[':
414             this.back();
415             try {...} catch (StackOverflowError e) {...}
420         }
421         return nextSimpleValue(c);
422     }
```

Figure 11: False Negative implementation smell.

Rationale: The nextValue method of JSONTokener class of org.json package has a switch statement at line 405, as shown in figure 11. This switch block has no default case and the DesigniteJava tool is unable to detect this. The CSV generated by tool can be seen in Figure 12, for implementation smells only for this class, filtered.



ImplementationSmells.csv - Excel

	Project Name	Package Name	Type Name	Method Name	Implementation Smell	Cause of the Smell	Methoc	art line no
89	JSON-java	org.json	JSONTokener	dehexchar	Magic Number	The method contains a magic number: 10	103	
90	JSON-java	org.json	JSONTokener	dehexchar	Magic Number	The method contains a magic number: 10	103	
91	JSON-java	org.json	JSONTokener	nextString	Complex Method	Cyclomatic complexity of the method is 17	282	
92	JSON-java	org.json	JSONTokener	nextString	Magic Number	The method contains a magic number: 4	282	
93	JSON-java	org.json	JSONTokener	nextString	Magic Number	The method contains a magic number: 16	282	
94	JSON-java	org.json	JSONTokener	nextString	Missing default	The following switch statement is missing a default case: !org.eclipse.jdt.core.dom.SwitchStateme	282	
95	JSON-java	org.json	JSONTokener	nextString	Missing default	The following switch statement is missing a default case: !org.eclipse.jdt.core.dom.SwitchStateme	282	
96	JSON-java	org.json	JSONTokener	nextTo	Complex Conditional	The conditional expression c == delimiter || c == 0 || c == '\n' || c == '\r' is complex.	348	
97	JSON-java	org.json	JSONTokener	nextTo	Complex Conditional	The conditional expression delimiters.indexOf(c) >= 0 || c == 0 || c == '\n' || c == '\r' is complex.	371	
98	JSON-java	org.json	JSONTokener	skipTo	Magic Number	The method contains a magic number: 1000000	459	

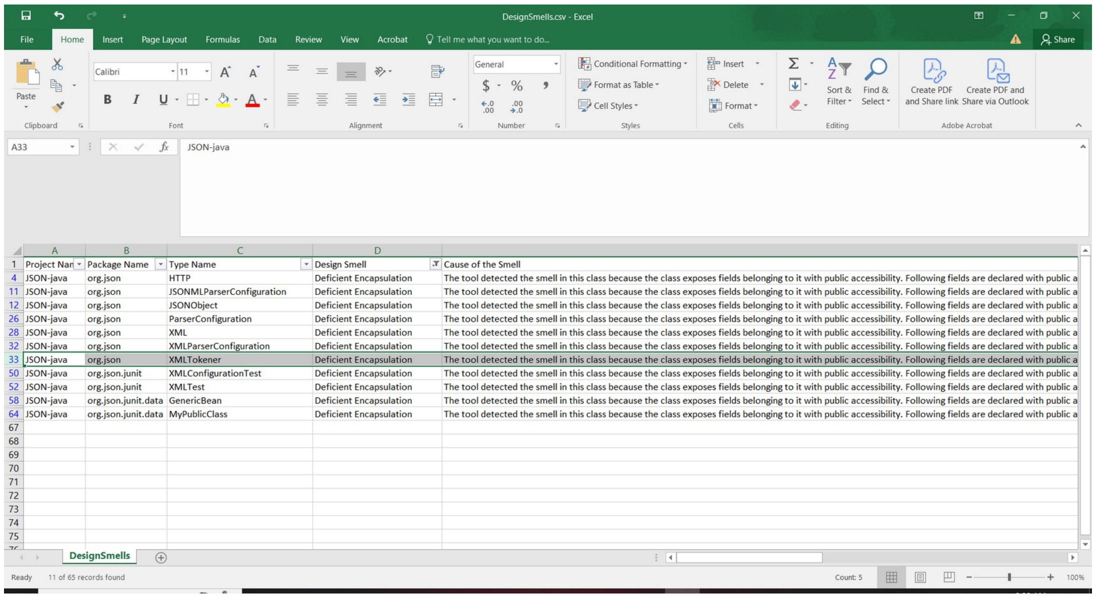
Ready 10 of 1100 records found

Figure 12: False Negative missing default for nextValue method of JSONTokener

Set-2: Design Smells

True Positive:

Deficient Encapsulation: The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public accessibility: entity.



DesignSmells.csv - Excel

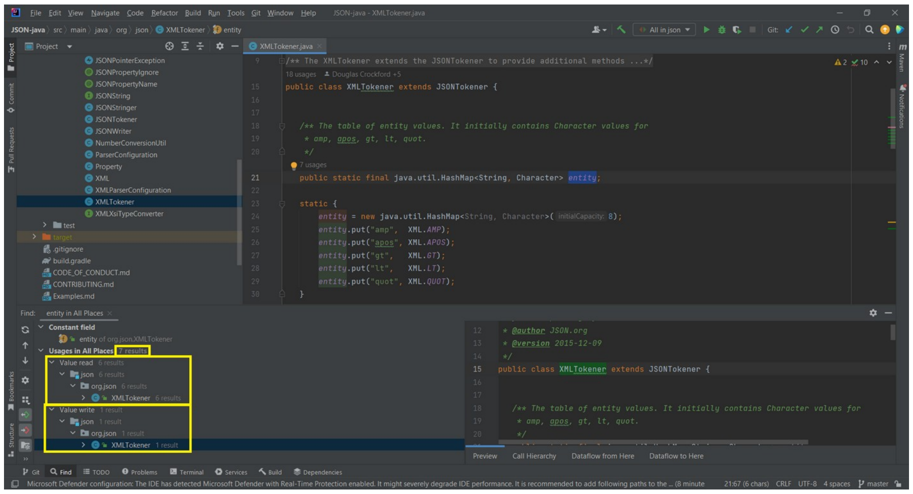
A33 JSON-java

	Project Nan	Package Name	Type Name	Design Smell	Cause of the Smell
4	JSON-java	org.json	HTTP	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
11	JSON-java	org.json	JSONMLParserConfiguration	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
12	JSON-java	org.json	JSONObject	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
26	JSON-java	org.json	ParserConfiguration	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
28	JSON-java	org.json	XML	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
32	JSON-java	org.json	XMLParserConfiguration	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
33	JSON-java	org.json	XMLTokener	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
50	JSON-java	org.json.junit	XMLConfigurationTest	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
52	JSON-java	org.json.junit	XMLTest	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
58	JSON-java	org.json.junit.data	GenericBean	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a
64	JSON-java	org.json.junit.data	MyPublicClass	Deficient Encapsulation	The tool detected the smell in this class because the class exposes fields belonging to it with public accessibility. Following fields are declared with public a

Ready 11 of 65 records found Count: 5

Figure 13: True Positive design smell.

Rationale: The entity field of org.json.XMLTokener class has 7 usages as show in figure 14 and all of them are in the same class, so it would have been better to keep it private instead of public. So here, the declared accessibility of “entity” data member is more permissive than actually required and thus the smell is true positive.

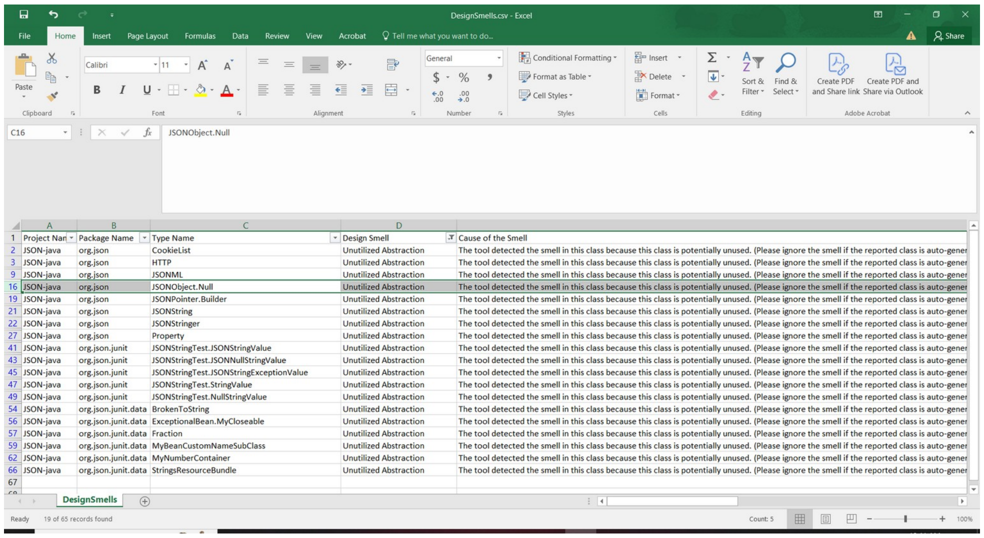


```java
/** The XMLTokener extends the JSONTokener to provide additional methods ...*/
public class XMLTokener extends JSONTokener {


    /** The table of entity values. It initially contains Character values for
     * amp, apos, gt, lt, quot.
     */
    public static final java.util.HashMap<String, Character> entity;

    static {
        entity = new java.util.HashMap<String, Character>( initialCapacity: 8);
        entity.put("amp",  XML.AMP);
        entity.put("apos", XML.APOS);
        entity.put("gt",   XML.GT);
        entity.put("lt",   XML.LT);
        entity.put("quot", XML.QUOT);
    }
```

Figure 14: Code example of true positive design smell.

False Positive:

Unutilized Abstraction: The tool detected the smell in this class (JSONObject.Null) because this class is potentially unused.

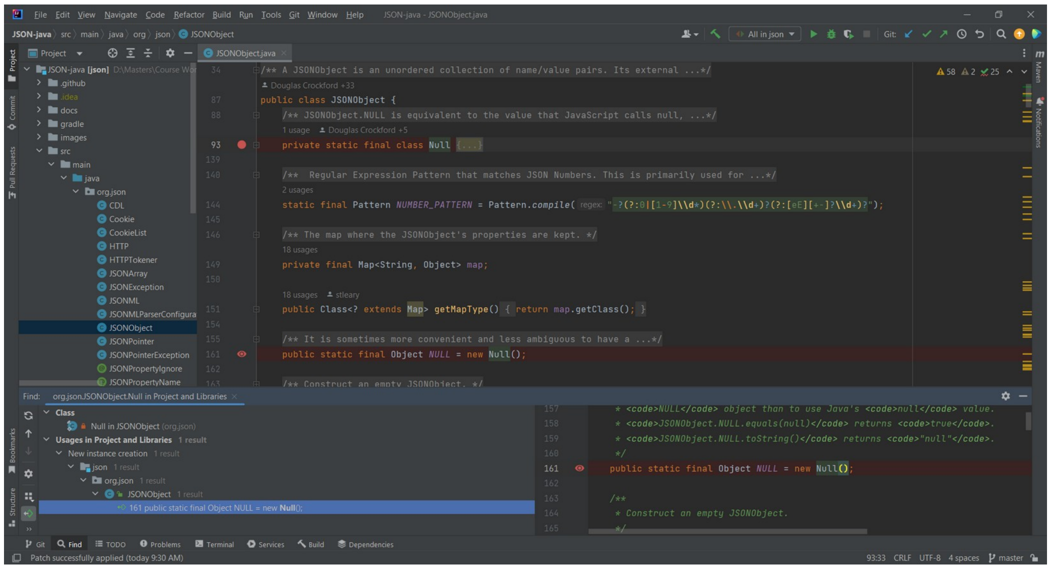


	Project Nan	Package Name	Type Name	Design Smell	Cause of the Smell
2	JSON-java	org.json	CookieList	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
3	JSON-java	org.json	HTTP	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
9	JSON-java	org.json	JSONML	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
16	JSON-java	org.json	JSONObject.Null	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
19	JSON-java	org.json	JSONPointer.Builder	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
21	JSON-java	org.json	JSONString	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
22	JSON-java	org.json	JSONStringer	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
27	JSON-java	org.json	Property	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
41	JSON-java	org.json.junit	JSONStringTest.JSONStringValue	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
43	JSON-java	org.json.junit	JSONStringTest.JSONNullStringValue	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
45	JSON-java	org.json.junit	JSONStringTest.JSONStringExceptionValue	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
47	JSON-java	org.json.junit	JSONStringTest.StringValue	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
49	JSON-java	org.json.junit	JSONStringTest.NullStringValue	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
54	JSON-java	org.json.junit.data	BrokenToString	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
56	JSON-java	org.json.junit.data	ExceptionalBean.MyCloseable	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
57	JSON-java	org.json.junit.data	Fraction	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
59	JSON-java	org.json.junit.data	MyBeanCustomNameSubClass	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
62	JSON-java	org.json.junit.data	MyNumberContainer	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener
66	JSON-java	org.json.junit.data	StringsResourceBundle	Unutilized Abstraction	The tool detected the smell in this class because this class is potentially unused. (Please ignore the smell if the reported class is auto-gener

Figure 15: False Positive design smell detected by DesigniteJava

Rationale: This design smell for static nested class Null inside the JSONObject class is false positive because it is not unused but actually used in the outer class (i.e. JSONObject). Line 93 is where the class is created and line 161 is where the class is being used, as shown in figure 16.

Moreover, the tool asks to ignore this smell if it is unused, but auto-generated and/or used for specific known purpose (i.e. some annotated classes in spring boot). But here, this class is actually used, not auto-generated and not used for specific known purpose but just a general plain java code.



```
/** A JSONObject is an unordered collection of name/value pairs. Its external ...*/
public class JSONObject {
    /** JSONObject.NULL is equivalent to the value that JavaScript calls null, ...*/
    private static final class Null {...}

    /**  Regular Expression Pattern that matches JSON Numbers. This is primarily used for ...*/
    static final Pattern NUMBER_PATTERN = Pattern.compile("-?(?:0|[1-9]\\d*)(?:\\.\\d+)?(?:[eE][+-]?\\d+)?");

    /** The map where the JSONObject's properties are kept. */
    private final Map<String, Object> map;

    public Class<? extends Map> getMapType() { return map.getClass(); }

    /** It is sometimes more convenient and less ambiguous to have a ...*/
    public static final Object NULL = new Null();

    /** Construct an empty JSONObject. */
```

```
Find: org.json.JSONObject.Null in Project and Libraries
Class
    Null in JSONObject (org.json)
Usages in Project and Libraries  1 result
    New instance creation  1 result
        json  1 result
            org.json  1 result
                JSONObject  1 result
                    161 public static final Object NULL = new Null();
```

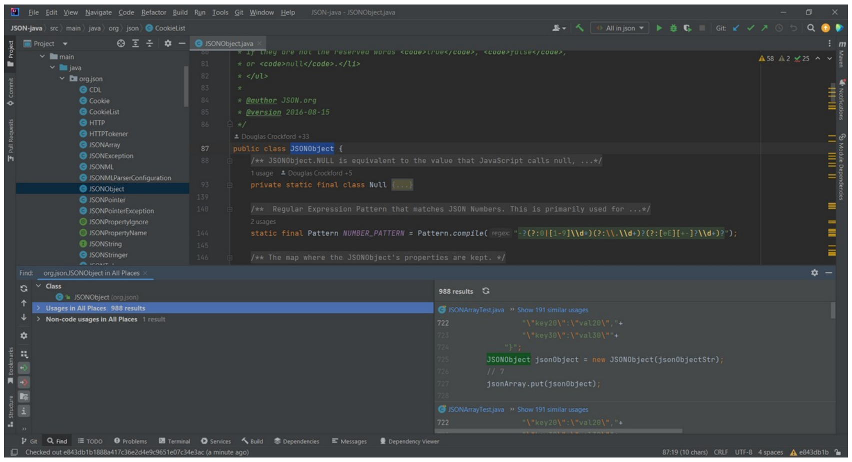
```
     * <code>NULL</code> object than to use Java's <code>null</code> value.
     * <code>JSONObject.NULL.equals(null)</code> returns <code>true</code>.
     * <code>JSONObject.NULL.toString()</code> returns <code>"null"</code>.
     */
    public static final Object NULL = new Null();

    /**
     * Construct an empty JSONObject.
     */
```

```
Patch successfully applied (today 9:30 AM)
```

Figure 16: False Positive design smell.

False Negative:
Undetected Smell:

Hub-Like Modularization: For JSONObject class there are lot of incoming and outgoing dependencies.



```
 * or <code>null</code>.</li>
 * </ul>
 *
 * @author JSON.org
 * @version 2016-08-15
 */
public class JSONObject {
    /** JSONObject.NULL is equivalent to the value that JavaScript calls null, ...*/
    private static final class Null {...}

    /**  Regular Expression Pattern that matches JSON Numbers. This is primarily used for ...*/
    static final Pattern NUMBER_PATTERN = Pattern.compile("-?(?:0|[1-9]\\d*)(?:\\.\\d+)?(?:[eE][+-]?\\d+)?");

    /** The map where the JSONObject's properties are kept. */
```

```
Find: org.json.JSONObject in All Places
Class
    JSONObject (org.json)
> Usages in All Places  988 results
> Non-code usages in All Places  1 result
```

```
988 results
JSONArrayTest.java  » Show 191 similar usages
722                "\"key20\":\"val20\","+
723                "\"key30\":\"val30\""+
724            "}";
725        JSONObject jsonObject = new JSONObject(jsonObjectStr);
726        // 7
727        jsonArray.put(jsonObject);
JSONArrayTest.java  » Show 191 similar usages
722                "\"key20\":\"val20\","+
```

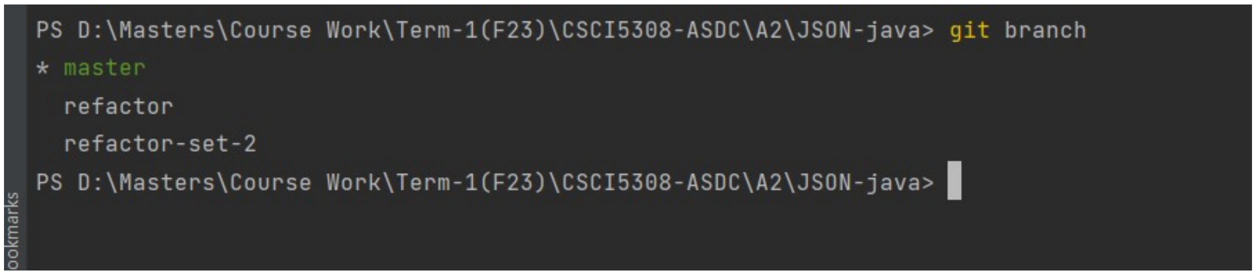
```
Checked out e843db1b1888a417c36e2d4e9c9651e07c34e3ac (a minute ago)
```

Figure 17: False Negative design smell.

Rationale: The org.json.JSONObject has 988 incoming dependencies in the project which refers to a hub-like modularization smell, but the tool is not able to detect this smell and is not present in the design smell CSV.

Task 2: Refactoring.

For set 1 and set 2 refactoring, I created two separate branches from my repository's (forked) master branch. This was done to raise separate pull requests. Both these branches were later on merged with my master branch.



```
PS D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A2\JSON-java> git branch
* master
  refactor
  refactor-set-2
PS D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A2\JSON-java>
```

Figure 18: 2 Branches for 2 different refactoring sets.

Set-1:

1. **Refactoring name: Rename Variable**

   Location

   File: src/main/java/org/json/CookieList.java
   Class: CookieList
   Method: toString()
   Line No: 49, 55, 61

   Explanation: boolean isEndOfPair is more meaningful as compared to just boolean b.

   Link of the files(s) of the previous commit (before refactoring):
   https://github.com/adityap27/JSON-java/blob/e843db1b1888a417c36e2d4e9c9651e07c34e3ac/src/main/java/org/json/CookieList.java

   Link of the files(s) of the commit with refactoring changes (after refactoring):
   https://github.com/adityap27/JSON-java/blob/097a401f3f38f7ac8ec6b4cc28fca1b6486f6d48/src/main/java/org/json/CookieList.java

   Link of commit:
   https://github.com/adityap27/JSON-java/commit/097a401f3f38f7ac8ec6b4cc28fca1b6486f6d48

2. **Refactoring name: Introduce explaining variable**

   Location

   File: src/main/java/org/json/XML.java
   Class: XML
   Method: toString()
   Line No: 850, 852, 854, 857

   Explanation: There are multiple long expressions which are using (indentFactor > 0) ? "\n" : "" sub-expression. indentationSuffix is introduced to replace this sub-expression to improve readability and reduce duplication.

   Link of the files(s) of the previous commit (before refactoring):
   https://github.com/adityap27/JSON-java/blob/097a401f3f38f7ac8ec6b4cc28fca1b6486f6d48/src/main/java/org/json/XML.java

   Link of the files(s) of the commit with refactoring changes (after refactoring):
   https://github.com/adityap27/JSON-java/blob/75419e3f257af9c365b0ef3e5f4428a335564f8d/src/main/java/org/json/XML.java

   Link of commit:
   https://github.com/adityap27/JSON-java/commit/75419e3f257af9c365b0ef3e5f4428a335564f8d

3. **Refactoring name: Decompose complex conditional**

   Location

   File: src/main/java/org/json/NumberConversionUtil.java
   Class: NumberConversionUtil
   Method: stringToNumber()
   Line No: 27, 56, 62

   Explanation: There are some digit-related conditions which may look complex to some developers due to multiple conditional operators.
   Eg: if ((initial >= '0' && initial <= '9') || initial == '-')
   and if(at1 == '0' && at2 >= '0' && at2 <= '9')

   The sub-condition c >= '0' && c <= '9' is moved to a method isNumericChar(...) to reduce the complexity of these conditions and better readability.
   Link of the files(s) of the previous commit (before refactoring):
   https://github.com/adityap27/JSON-java/blob/75419e3f257af9c365b0ef3e5f4428a335564f8d/src/main/java/org/json/NumberConversionUtil.java

Link of the files(s) of the commit with refactoring changes (after refactoring):
https://github.com/adityap27/JSON-java/blob/7f1cb8bf62015016d4b02879b00cc7477b62c570/src/main/java/org/json/NumberConversionUtil.java

Link of commit:
https://github.com/adityap27/JSON-java/commit/7f1cb8bf62015016d4b02879b00cc7477b62c570

Pull Request for Set-1 Refactoring:

https://github.com/stleary/JSON-java/pull/831

PR Status as of Assignment Submission: Approved.

Set-2:

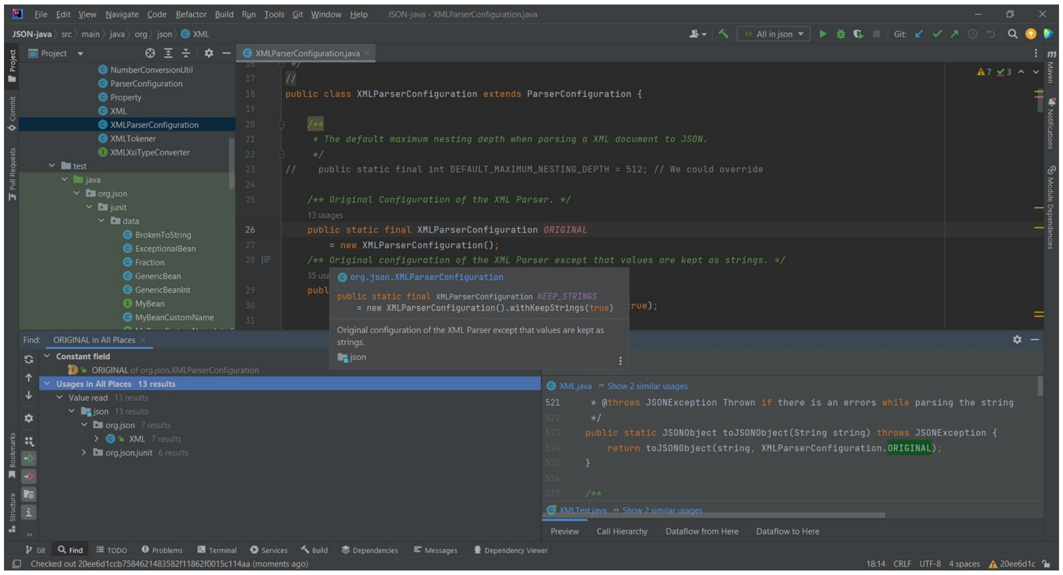
1. **Refactoring name: Move Field**

   Location

   File: src/main/java/org/json/XML.java,
   src/main/java/org/json/XMLParserConfiguration.java,
   src/test/java/org/json/junit/XMLConfigurationTest.java,
   src/test/java/org/json/junit/XMLTest.java

   Moved "ORIGINAL" field from org.json.XMLParserConfiguration to org.json.XML. Also renamed it to ORIGINAL_PARSER_CONFIG for better understanding with respect to new class XML. Other changes are due to the change in Class and renaming of the field.

   Explanation: The field ORIGINAL is used 7 times in production code and all of those usages are in XML class. So it makes more sense to move this field from org.json.XMLParserConfiguration to org.json.XML



```
//
public class XMLParserConfiguration extends ParserConfiguration {

    /**
     * The default maximum nesting depth when parsing a XML document to JSON.
     */
//    public static final int DEFAULT_MAXIMUM_NESTING_DEPTH = 512; // We could override

    /** Original Configuration of the XML Parser. */
    public static final XMLParserConfiguration ORIGINAL
        = new XMLParserConfiguration();
    /** Original configuration of the XML Parser except that values are kept as strings. */
```

```
org.json.XMLParserConfiguration
public static final XMLParserConfiguration KEEP_STRINGS
    = new XMLParserConfiguration().withKeepStrings(true)
```

Original configuration of the XML Parser except that values are kept as strings.

Find: ORIGINAL in All Places

Constant field: ORIGINAL of org.json.XMLParserConfiguration

Usages in All Places 13 results

```
     * @throws JSONException Thrown if there is an errors while parsing the string
     */
    public static JSONObject toJSONObject(String string) throws JSONException {
        return toJSONObject(string, XMLParserConfiguration.ORIGINAL);
    }

    /**
```

Checked out 20ee6d1ccb7584621483582f11862f0015c114aa (moments ago)

Figure 19: move field.

Link of the files(s) of the previous commit (before refactoring):
https://github.com/adityap27/JSON-java/tree/20ee6d1ccb7584621483582f11862f0015c114aa

Link of the files(s) of the commit with refactoring changes (after refactoring):
https://github.com/adityap27/JSON-java/tree/306744f74f4962dca6b13fde3e62ba069d2bd120

Link of commit:
https://github.com/adityap27/JSON-java/commit/306744f74f4962dca6b13fde3e62ba069d2bd120

2. Refactoring name: Change bidirectional association to unidirectional association

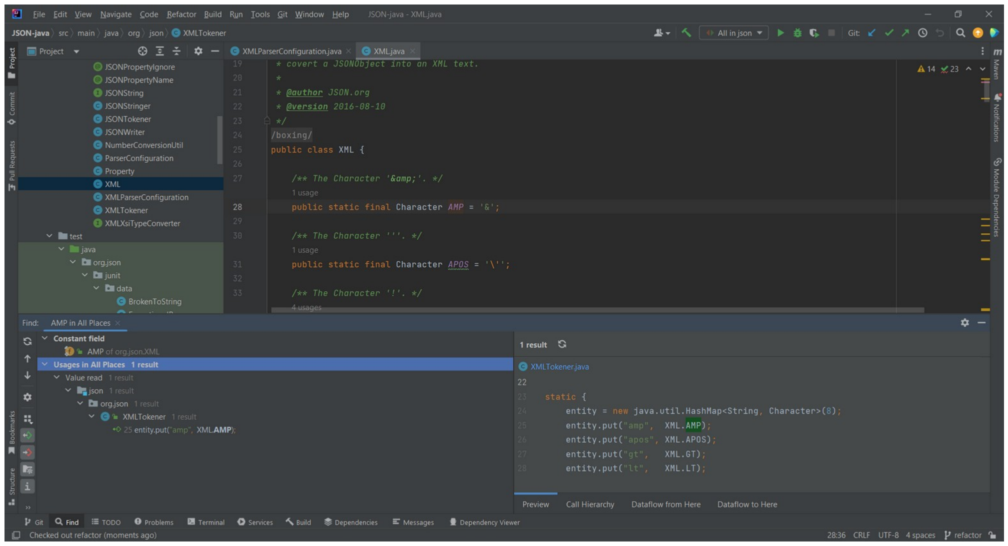
Location

File: src/main/java/org/json/XML.java, src/main/java/org/json/XMLTokener.java, src/main/java/org/json/JSONML.java

The variables in line 25 to 51 of org.json.XML class are moved to the org.json.XMLTokener class to change the bidirectional association to unidirectional. Other class org.json.JSONML is changed due to these field movements.

Explanation: There is a Cyclic-modularization between the XML and XMLTokener classes. This points to the bidirectional association issue where both class know about each other directly and using their fields or method. This can be resolved and converted to unidirectional association by simply moving some fields from XML class to

XMLTokener itself. These fields are actually used only in XMLTokener again, so it makes more sense to move there.

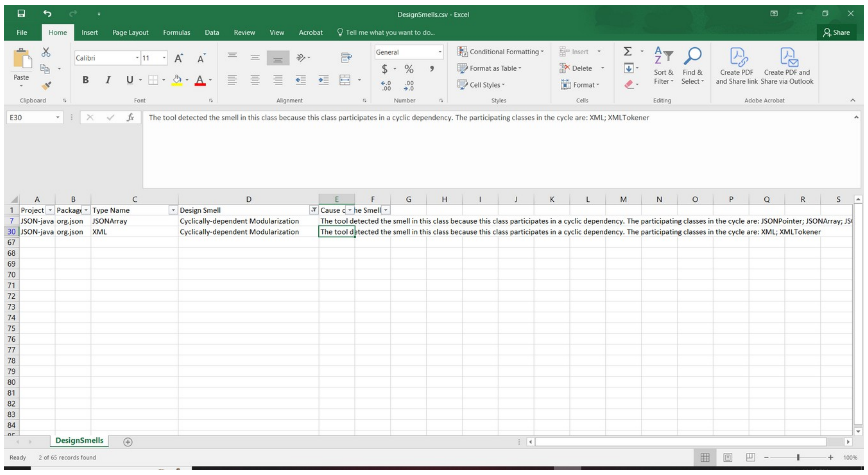


```
 * covert a JSONObject into an XML text.
 *
 * @author JSON.org
 * @version 2016-08-10
 */
/boxing/
public class XML {

    /** The Character '&amp;'. */
    1 usage
    public static final Character AMP = '&';

    /** The Character '''. */
    1 usage
    public static final Character APOS = '\'';

    /** The Character '!'. */
    4 usages
```

```
Find: AMP in All Places
Constant field
    AMP of org.json.XML
Usages in All Places  1 result
    Value read  1 result
        json  1 result
            org.json  1 result
                XMLTokener  1 result
                    25 entity.put("amp", XML.AMP);
```

```
1 result
XMLTokener.java
22
    static {
        entity = new java.util.HashMap<String, Character>(8);
        entity.put("amp",  XML.AMP);
        entity.put("apos", XML.APOS);
        entity.put("gt",   XML.GT);
        entity.put("lt",   XML.LT);
```

Figure 20: Field is only used in XMLTokener. These fields are causing bi-directional association.



E30: The tool detected the smell in this class because this class participates in a cyclic dependency. The participating classes in the cycle are: XML; XMLTokener

	Project	Package	Type Name	Design Smell	Cause of the Smell
7	JSON-java	org.json	JSONArray	Cyclically-dependent Modularization	The tool detected the smell in this class because this class participates in a cyclic dependency. The participating classes in the cycle are: JSONPointer; JSONArray; JS
30	JSON-java	org.json	XML	Cyclically-dependent Modularization	The tool detected the smell in this class because this class participates in a cyclic dependency. The participating classes in the cycle are: XML; XMLTokener

Ready 2 of 65 records found

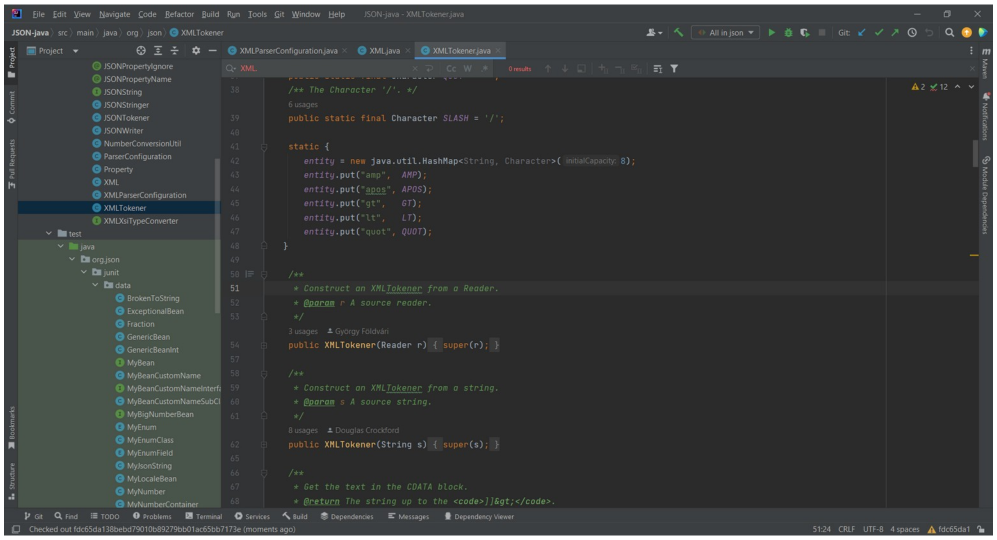
Figure 21: Tool also detects cycle between these 2 classes.

Link of the files(s) of the previous commit (before refactoring):
https://github.com/adityap27/JSON-java/tree/4a468d163ab09687d2333c3d928403c2e7ae6e4b

Link of the files(s) of the commit with refactoring changes (after refactoring):
https://github.com/adityap27/JSON-java/tree/fdc65da138bebd79010b89279bb01ac65bb7173e

Link of commit:
https://github.com/adityap27/JSON-java/commit/fdc65da138bebd79010b89279bb01ac65bb7173e

After refactoring, there are no uses of XML class in XMLTokener class, as shown in figure below.



```
    /** The Character '/'. */
    public static final Character SLASH = '/';

    static {
        entity = new java.util.HashMap<String, Character>( initialCapacity: 8);
        entity.put("amp",  AMP);
        entity.put("apos", APOS);
        entity.put("gt",   GT);
        entity.put("lt",   LT);
        entity.put("quot", QUOT);
    }

    /**
     * Construct an XMLTokener from a Reader.
     * @param r A source reader.
     */
    public XMLTokener(Reader r) { super(r); }

    /**
     * Construct an XMLTokener from a string.
     * @param s A source string.
     */
    public XMLTokener(String s) { super(s); }

    /**
     * Get the text in the CDATA block.
     * @return The string up to the <code>]]&gt;</code>.
```

Figure 22: Removed bi-directional association.

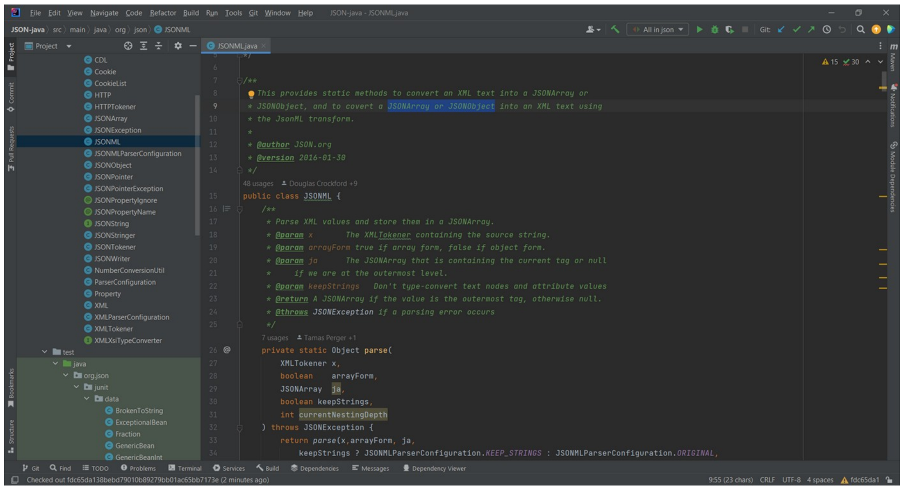
3. Refactoring name: Extract class

Location

File:
src/main/java/org/json/JSONMLArray.java (JSONML.java before),
src/main/java/org/json/JSONMLObject.java (new file),
src/test/java/org/json/junit/JSONMLObjectTest.java (JSONMLObjectTest.java before),

The 7 methods from org.json.JSONML class related to JSONObject are extracted to create a new class called JSONMLObject in the org.json package. Moreover, the generic class is now renamed to JSONMLArray. Other test files are changed due to the renaming and shifting of these methods.

Explanation: The org.json.JSONML violates the single responsibility principle. It handles conversion from XML to JSONArray as well as JSONObject and reverse. This class can be split into 2, using extract class refactoring. This will ensure single responsibility of the classes. I have also modified test cases and the documentation to reflect these changes.



```java
/**
 * This provides static methods to convert an XML text into a JSONArray or
 * JSONObject, and to covert a JSONArray or JSONObject into an XML text using
 * the JsonML transform.
 *
 * @author JSON.org
 * @version 2016-01-30
 */
public class JSONML {
    /**
     * Parse XML values and store them in a JSONArray.
     * @param x       The XMLTokener containing the source string.
     * @param arrayForm true if array form, false if object form.
     * @param ja      The JSONArray that is containing the current tag or null
     *     if we are at the outermost level.
     * @param keepStrings   Don't type-convert text nodes and attribute values
     * @return A JSONArray if the value is the outermost tag, otherwise null.
     * @throws JSONException if a parsing error occurs
     */
    private static Object parse(
        XMLTokener x,
        boolean    arrayForm,
        JSONArray  ja,
        boolean keepStrings,
        int currentNestingDepth
    ) throws JSONException {
        return parse(x,arrayForm, ja,
            keepStrings ? JSONMLParserConfiguration.KEEP_STRINGS : JSONMLParserConfiguration.ORIGINAL,
```

Figure 23: Single class is doing 2 different things.

Link of the files(s) of the previous commit (before refactoring):
https://github.com/adityap27/JSON-java/tree/8124612ac7b01ff7277764d313a422ddede6a5e1

Link of the files(s) of the commit with refactoring changes (after refactoring):
https://github.com/adityap27/JSON-java/tree/027f48f907f44ab388ce350b53b93dc51de1c4ba

Link of commit:
https://github.com/adityap27/JSON-java/commit/027f48f907f44ab388ce350b53b93dc51de1c4ba

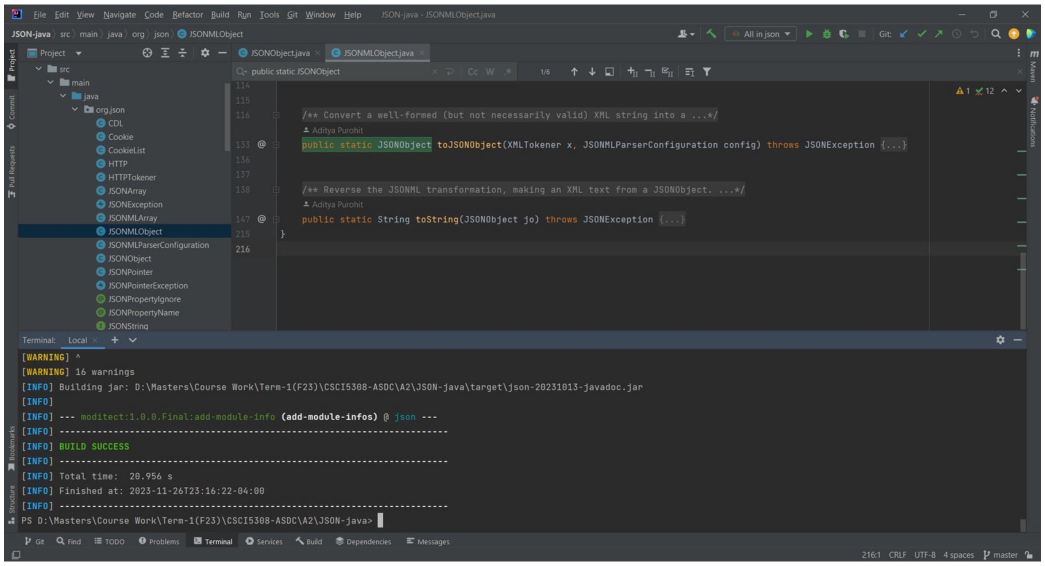
Pull Request for Set-2 Refactoring:

https://github.com/stleary/JSON-java/pull/833

PR Status as of Assignment Submission: **In Review**.

Build Status after all refactoring:

The build is passing after all refactoring applied. “mvn clean package” command was used to check the build status.



```
/** Convert a well-formed (but not necessarily valid) XML string into a ...*/
public static JSONObject toJSONObject(XMLTokener x, JSONMLParserConfiguration config) throws JSONException {...}

/** Reverse the JSONML transformation, making an XML text from a JSONObject. ...*/
public static String toString(JSONObject jo) throws JSONException {...}
}
```

```
[WARNING] ^
[WARNING] 16 warnings
[INFO] Building jar: D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A2\JSON-java\target\json-20231013-javadoc.jar
[INFO]
[INFO] --- moditect:1.0.0.Final:add-module-info (add-module-infos) @ json ---
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  20.956 s
[INFO] Finished at: 2023-11-26T23:16:22-04:00
[INFO] ------------------------------------------------------------------------
PS D:\Masters\Course Work\Term-1(F23)\CSCI5308-ASDC\A2\JSON-java>
```

Figure 24: post-refactoring: build status.

References:

[1] Stleary, “Stleary/JSON-java: A reference implementation of a JSON package in Java.,” *GitHub*. [Online]. Available: https://github.com/stleary/JSON-java. [Accessed Nov. 5, 2023].

[2] AlDanial, “Aldanial/Cloc: Cloc Counts Blank Lines, comment lines, and physical lines of source code in many programming languages.,” *GitHub*. [Online]. Available: https://github.com/AlDanial/cloc. [Accessed Nov. 5, 2023].

[3] “Designite - Reduce Technical Debt of your Software,” Designite-tools.com. [Online]. Available: https://www.designite-tools.com/. [Accessed: 26-Nov-2023].

[4] “IntelliJ IDEA – the leading Java and Kotlin IDE,” *JetBrains*. [Online]. Available: https://www.jetbrains.com/idea/. [Accessed: 26-Nov-2023].

[5] B. Porter, J. van Zyl, and O. Lamy, “Welcome to Apache maven,” *Apache.org*. [Online]. Available: https://maven.apache.org/. [Accessed: 26-Nov-2023].

[6] “Java Archive Downloads - Java SE 17,” *Oracle.com*. [Online]. Available: https://www.oracle.com/java/technologies/javase/jdk17-archive-downloads.html. [Accessed: 26-Nov-2023].