

Faculty of Engineering and Applied Science

University of Ontario Institute of Technology

**SOFE 3980U**

**Winter 2018**

**Project Team 14**

Software Quality Project

jEdit Bugs

**Faculty Supervisor: Dr. Akramul Azim**

**April 6, 2018**

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Student ID** | **Signature** |
| Saleh Nawar | 100536488 |  |
| Fawwaz Khayyat | 100568635 |  |
| Anwar Almukhtar | 100481975 |  |
| Christian MacLeod | 100560504 |  |

**Introduction:**

The objective of the project is to improve the quality of the jEdit text editor. jEdit is a mature programmer text editor written in java that contains source code editing, search and replace, file management , and customization. Some of the advantages of jEdit is that it is highly customizable and offers a variety of plugins. However, due to being developed using Java, it can be be demanding in terms of memory usage.

**Project Details:**

Findbugs which is an open source software for static code analysis in java program that scans for bugs / errors in the code was used .

Findbug reported 660 bugs in different categories , and around 85 bugs hits were addressed and fixed.

Statics of the bugs/errors:

|  |  |
| --- | --- |
| Bad practice | 234 bugs |
| Dodgy code | 157 bugs |
| Malicious code vulnerability | 94 bugs |
| Performance | 101 bugs |
| Internationalization | 35 bugs |
| Multithreaded correctness | 25 bugs |
| Correctness | 10 bugs |
| Experimental | 4 bugs |

Also, Junit was used in order not to break the relationship between the functions and classes by following these steps:

* Write test cases.
* Test.
* Modify.
* Test again.

**Test Cases Design:**

Due to the time constraint of the project, the team decided to focus their efforts on developing two test cases for the project. These test cases were designed with the purpose of ensuring that alterations to specific functions do not damage the code. Moreover, due to the nature of the test cases, the team decided not to use any tools to generate test data, instead, specific data were used as input for the the test cases.

**Test Suite 1:**

This test suite includes 10 test cases. The test cases were made to test the void function *attribute*. This function sets the values of local variables. However, the local variables could not be accessed from the test class because they are private. Moreover, this function is a void function that do not return a value, we added some getters to the class to test that the function is assigning the value to the correct local variable.

The main purpose of the test cases is to make sure the code did not break after the modification. Therefore, the test cases were written first and runned. After the test cases passed, the code got modified. Then, the test cases ran again. Finally, All the test cases passed.

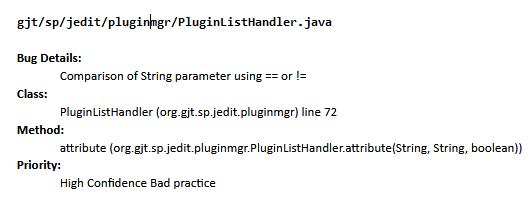


Figure 1: attribute() bug report

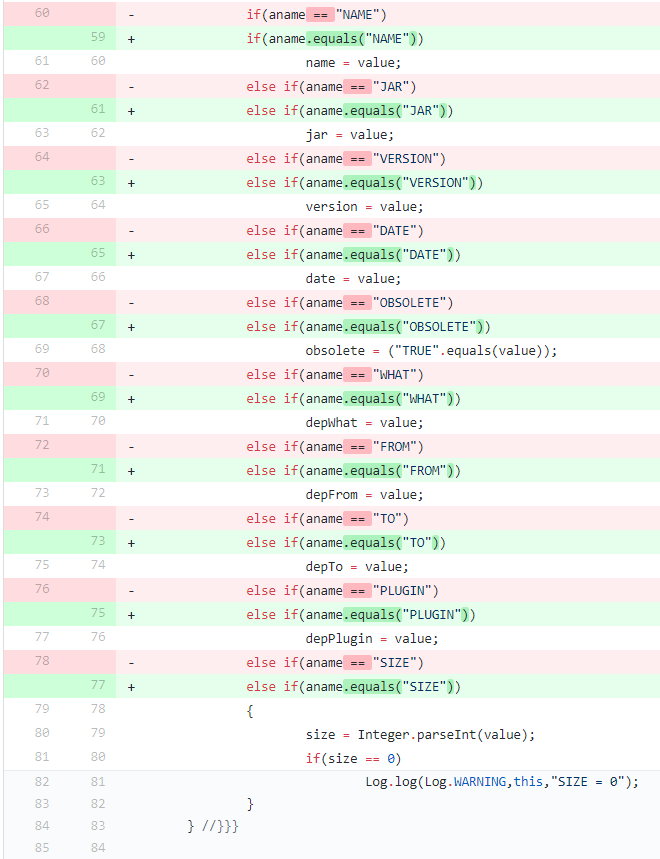


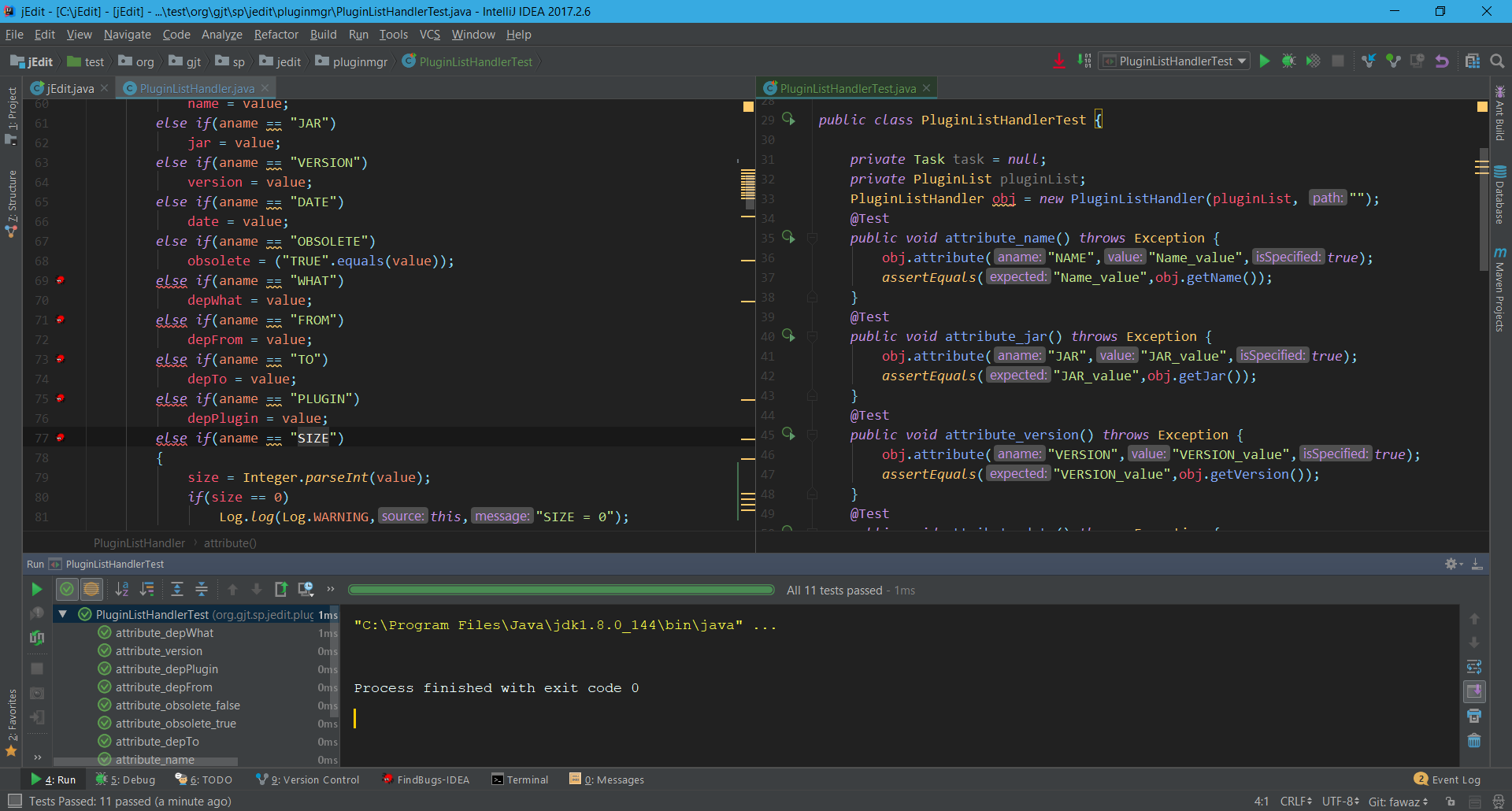
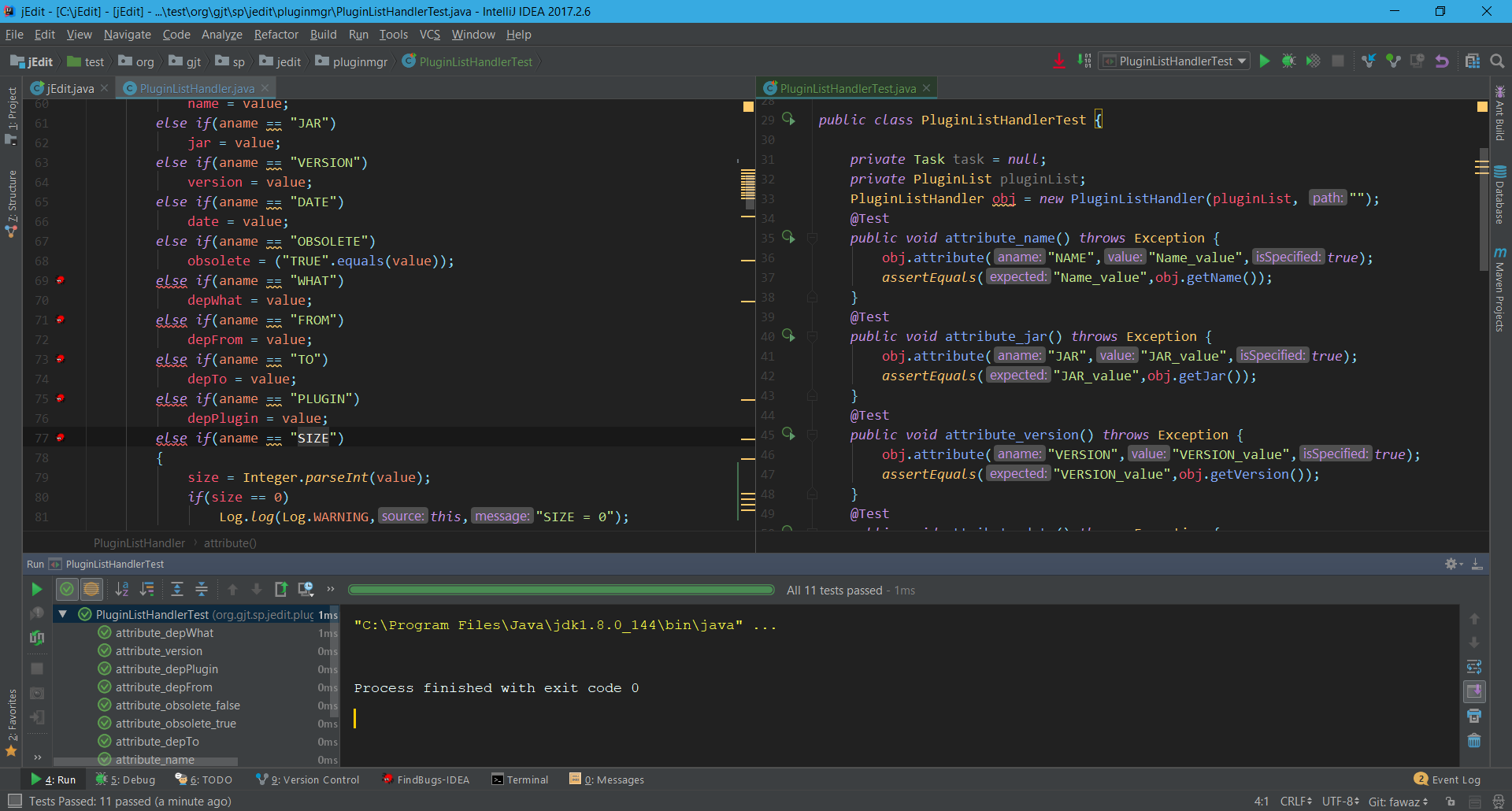
Figure 2: attribute() bug fixes

Figure 3: PluginListHandlerTest class

Figure 4: Test case results

**Test case 2:**

This test case deals with the getRegisters() function of the Registers.java class. Due to the fact the getter function exposed the array registers, a second function, getRegistersCpy(), that returns a copy of the array was made to address this security concern. Afterwards, a test case was made that insures that both functions return identical arrays of the same length.

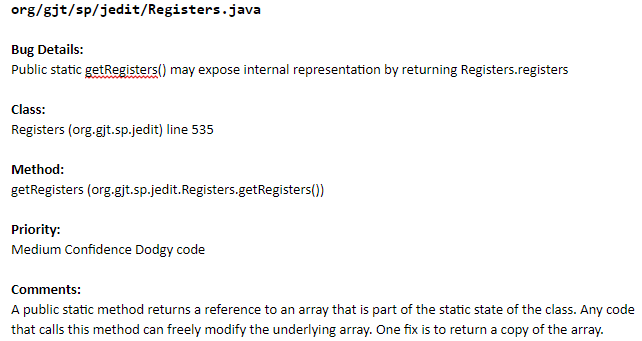
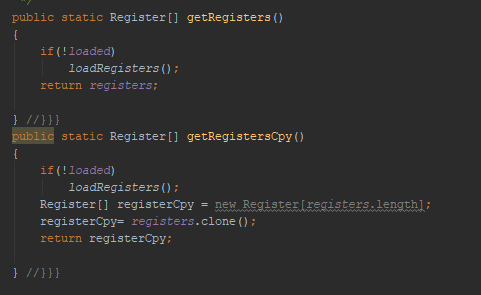


Figure 5: getRegisters bug report

Figure 6: getRegisters and getRegistersCpy

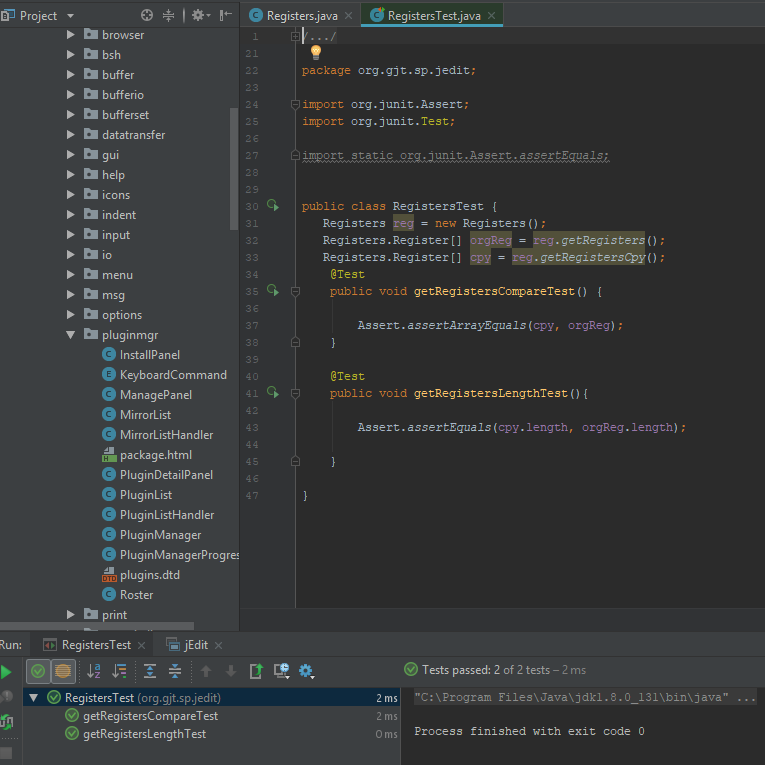


Figure 7: RegistersTest class and test case results

**Challenges:**

Many of the non-void methods in the application were declared private, making them difficult to test without exposing the internals of the system.

Additionally, the software is already highly stable. Version 4.3p9 launched in early 2007, and it has been receiving regular updates since. The majority of the open issues on their bugtracker are either unreproducible, or marked as corrected in an upcoming release.

**Lessons Learned:**

* Improve the quality of the code
* Unit tests should be developed before & during the early development of the code, as post release testing leads to a decrease in initial quality, and a reduced ability to obtain full test coverage.

**Implementation:**

The implementation of fixes to Jedit were made using a variety of tools:

- Findbugs was used for identifying potential bugs.

-intelliJ IDEA was used as IDE to run the project.

-Github was used as source control.

The following section will provide examples of bug fixes made by the team.

**gjt/sp/util/StringList.java**

**Bug Details:**

#### Return value of split(String, Object) ignored, but method has no side effect

**Class:**

#### StringList (org.gjt.sp.util) line 146

**Method:**

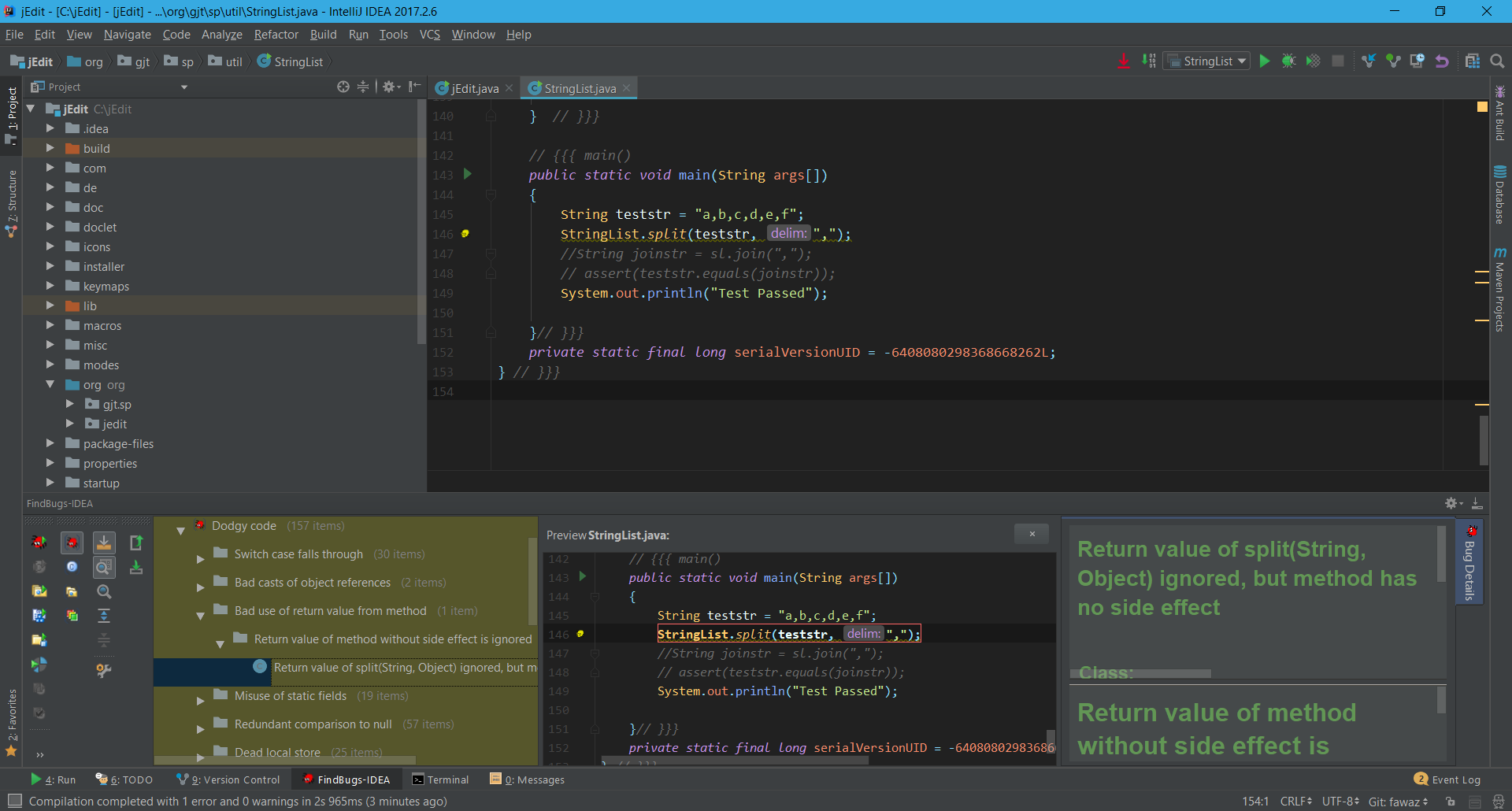
#### main (org.gjt.sp.util.StringList.main(String[]))

**Priority:**

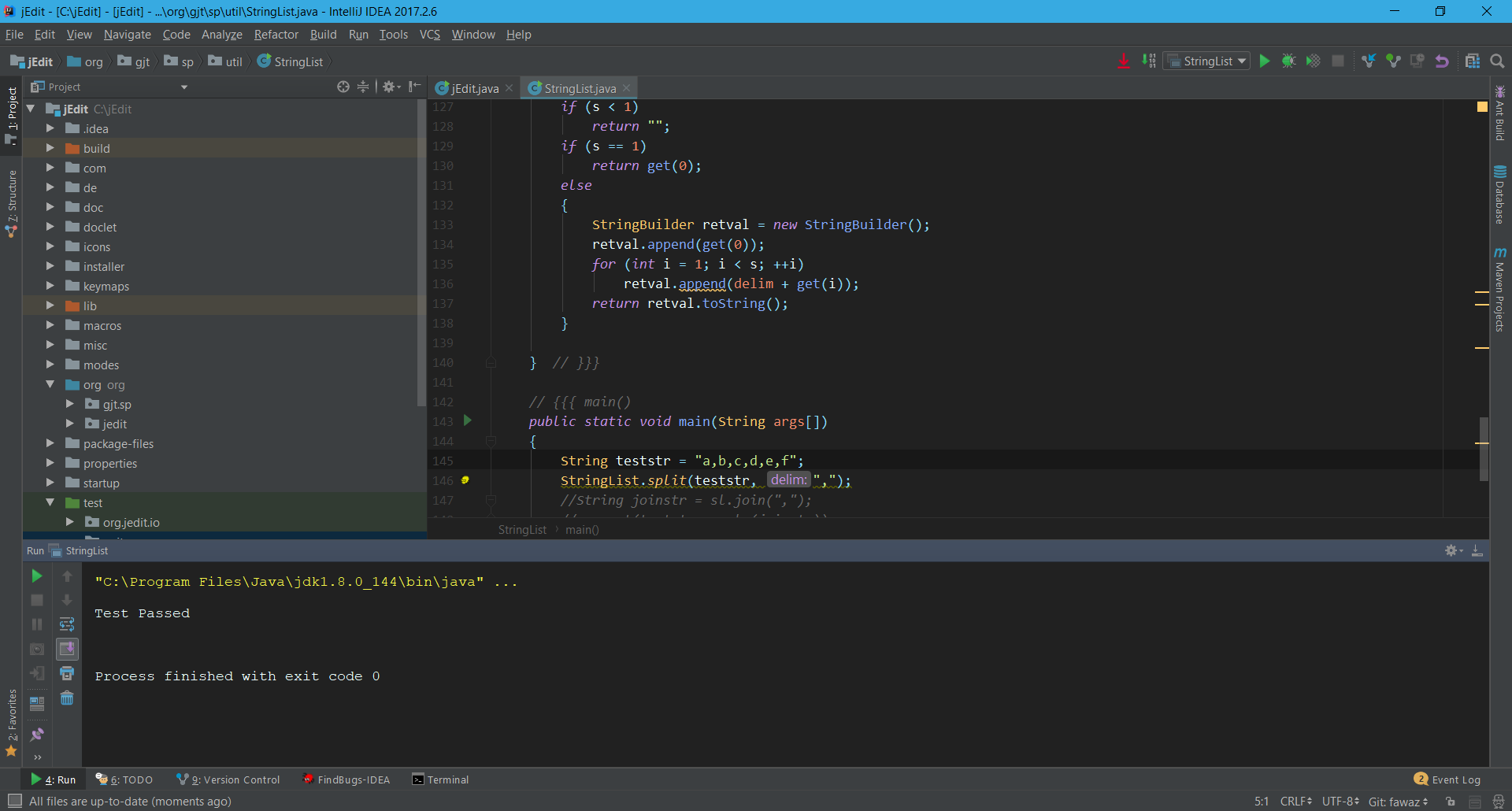
#### Medium Confidence Dodgy code

**Comments:**

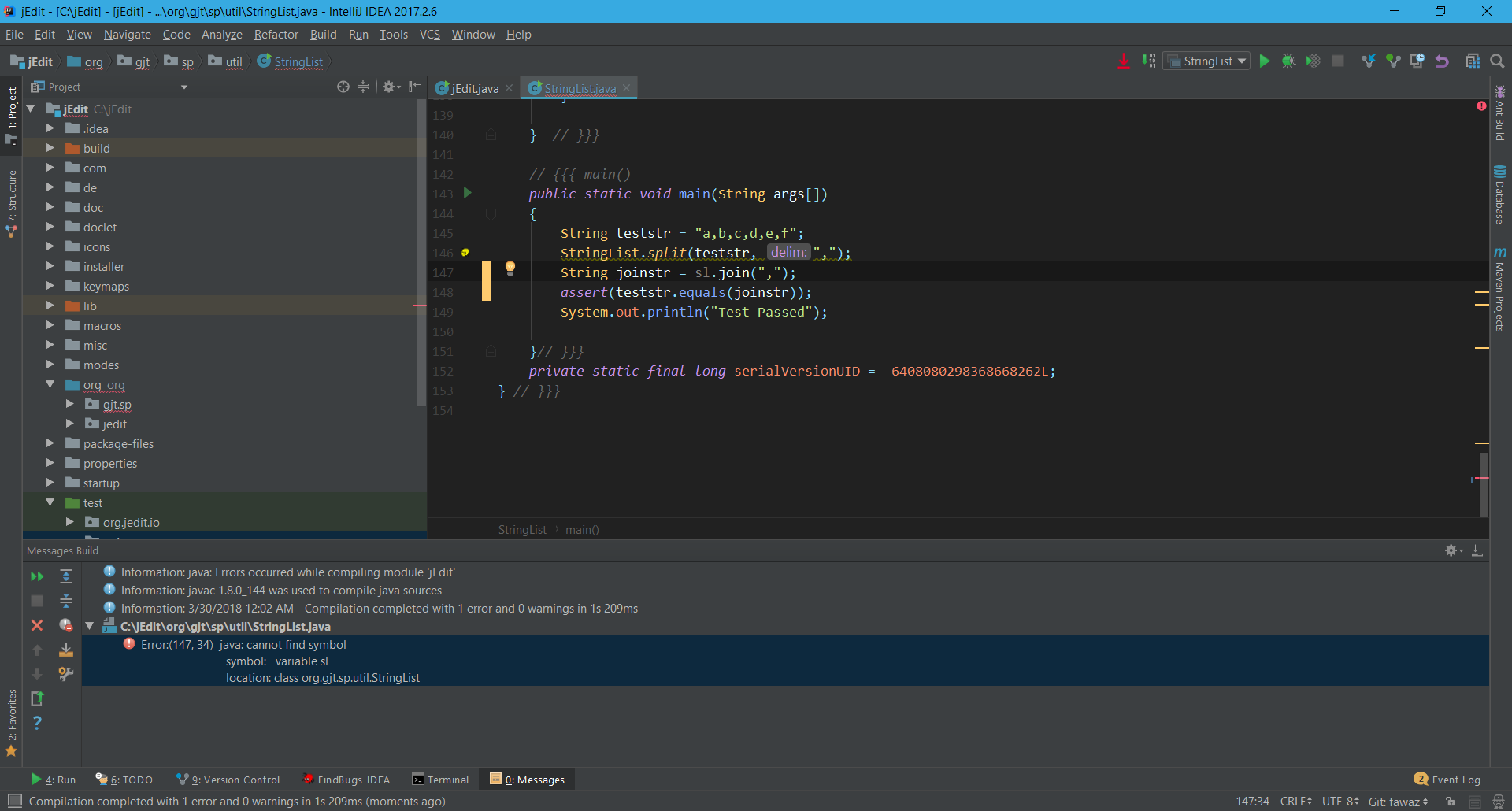
The code in the *main* function is used to test *split* and *join* functions. The code run without error because part of the code is commented. If we uncomment the code, it will produce errors. That is because of missing sl variable storing the return value of the splitted string..



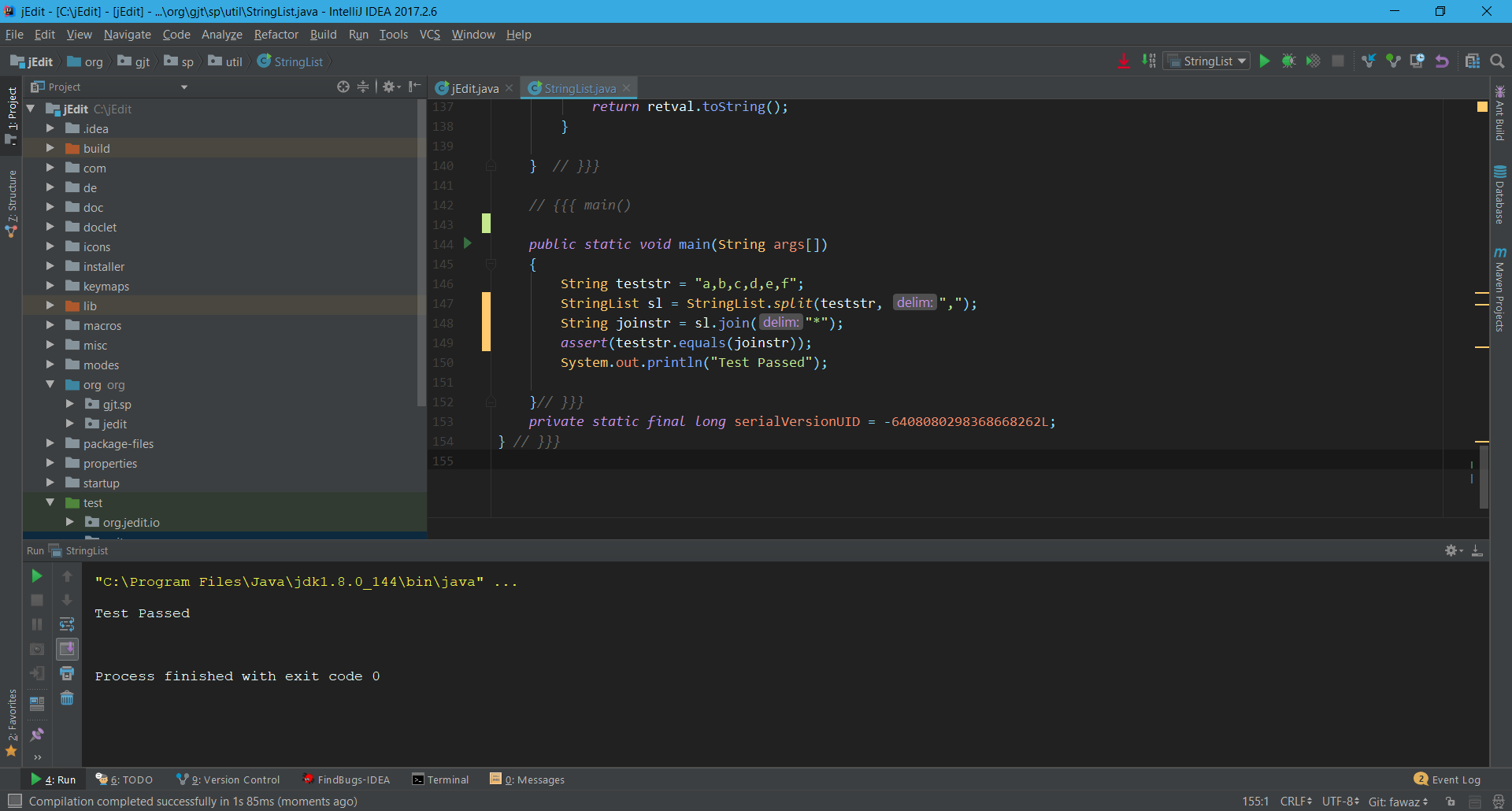
**Original Code**



**Running Original Code**



**Running Uncommented Code**



**Running Fixed Code**

**gjt/sp/jedit/pluginmgr/PluginListHandler.java**

**Bug Details:**

#### Comparison of String parameter using == or !=

**Class:**

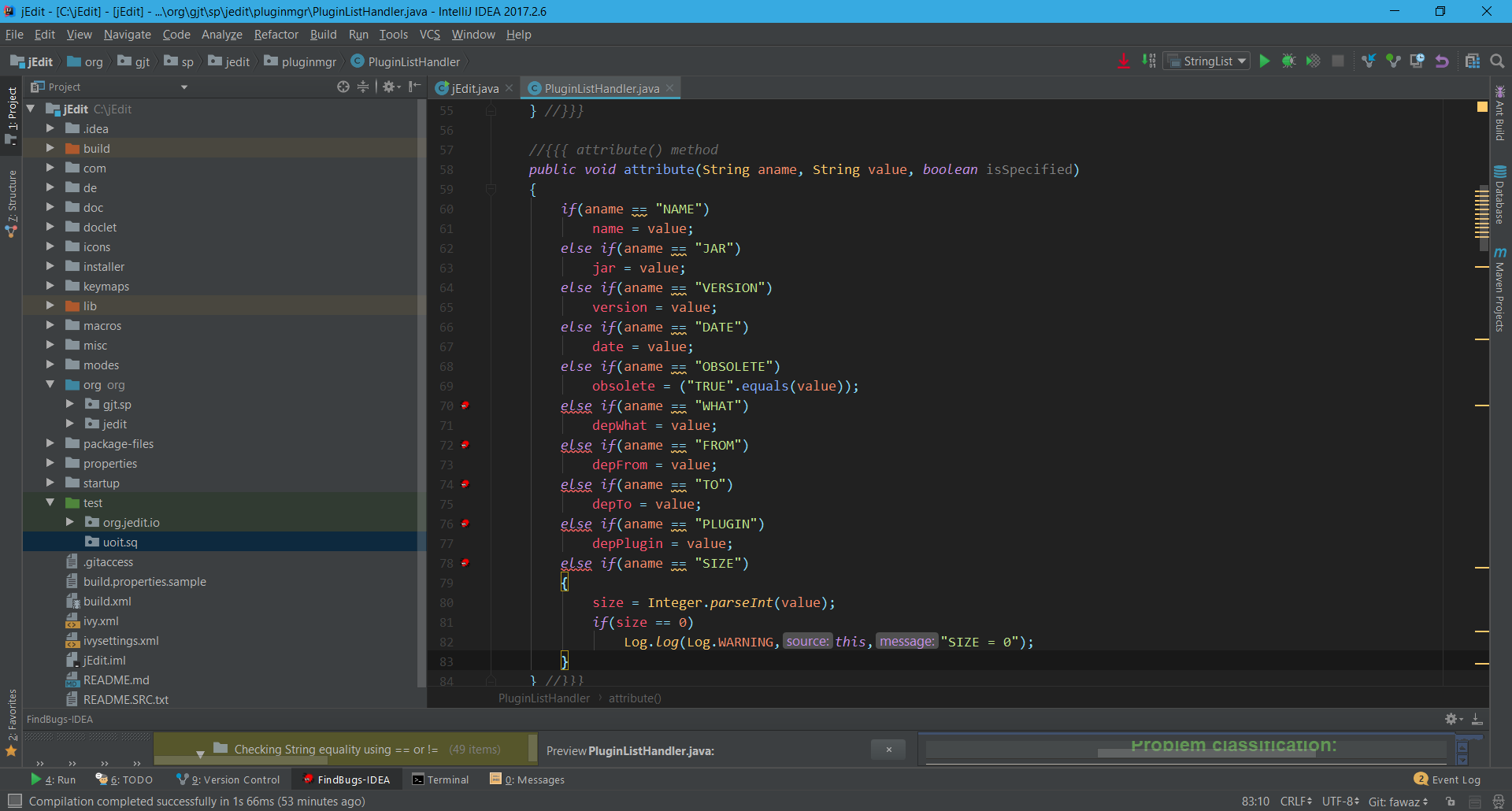
#### PluginListHandler (org.gjt.sp.jedit.pluginmgr) line 72

**Method:**

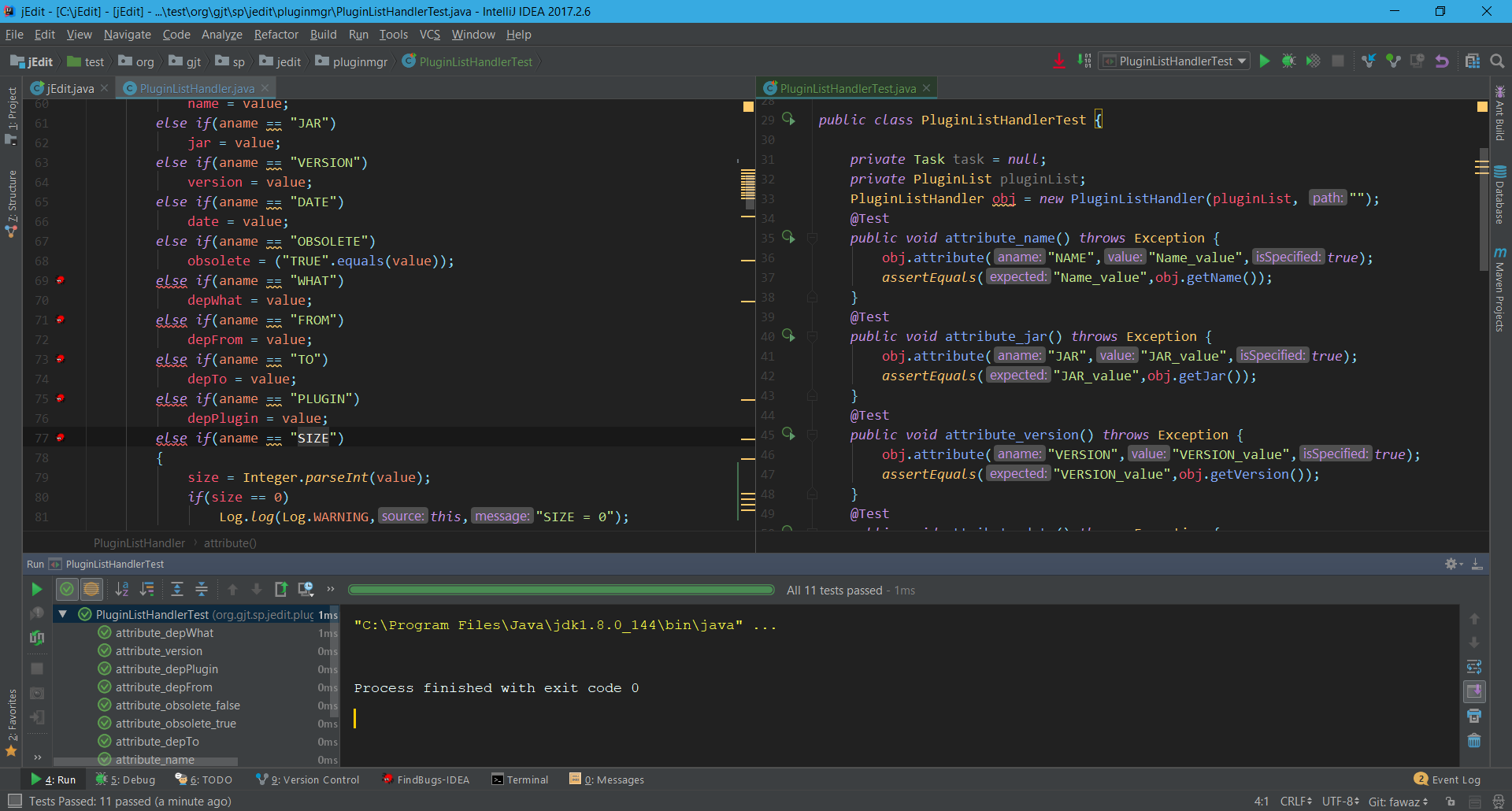
#### attribute (org.gjt.sp.jedit.pluginmgr.PluginListHandler.attribute(String, String, boolean))

**Priority:**

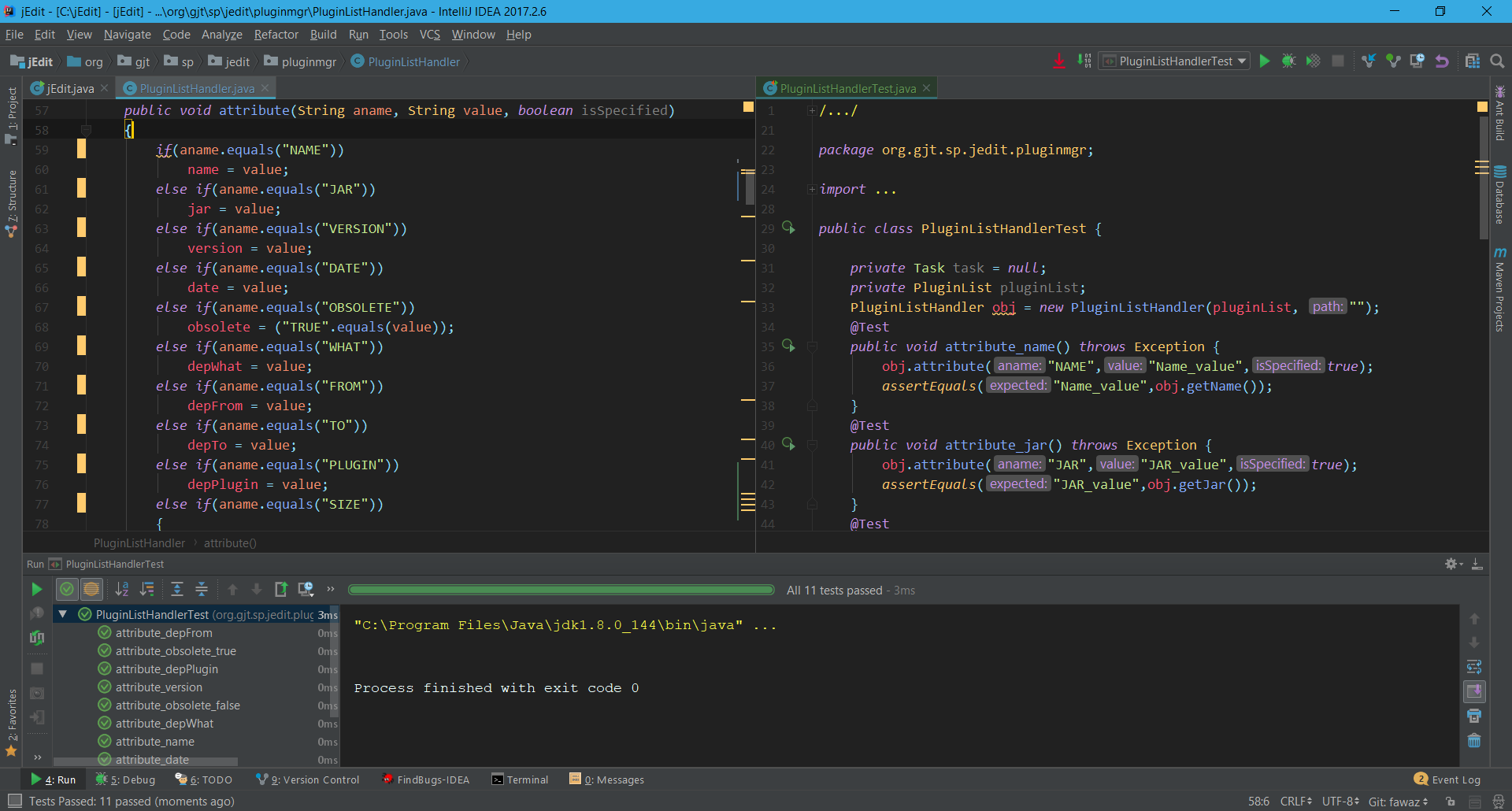
#### High Confidence Bad practice



**Original Code**



**Test Cases Passed Before Code Modification**



**Test Cases Passed After Modifying the Code**

**gjt/sp/jedit/browser/VFSDirectoryEntryTableModel.java**

**Bug Details:**

#### VFSDirectoryEntryTableModel$EntryCompare implements Comparator but not Serializable

**Class:**

#### VFSDirectoryEntryTableModel$EntryCompare (org.gjt.sp.jedit.browser) lines 412-482

**Method:**

#### attribute (org.gjt.sp.jedit.pluginmgr.PluginListHandler.attribute(String, String, boolean))

**Priority:**

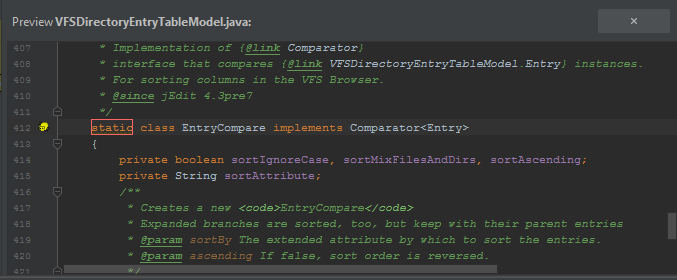
#### Medium Confidence Bad practice

**Problem classification:**

#### Bad practice (Incorrect definition of Serializable class) SE\_COMPARATOR\_SHOULD\_BE\_SERIALIZABLE (Comparator doesn't implement Serializable)

**Notes:**

In class org.gjt.sp.jedit.browser.VFSDirectoryEntryTableModel$EntryCompare ComparatorIdiom (Se)



**gjt/sp/jedit/bufferset/BufferSet.java**

**Bug Details:**

#### BufferSet$NameSorter implements Comparator but not Serializable

**Class:**

#### BufferSet$NameSorter (org.gjt.sp.jedit.bufferset) lines 364-374

**Priority**:

#### Medium Confidence Bad practice

**Problem classification:**

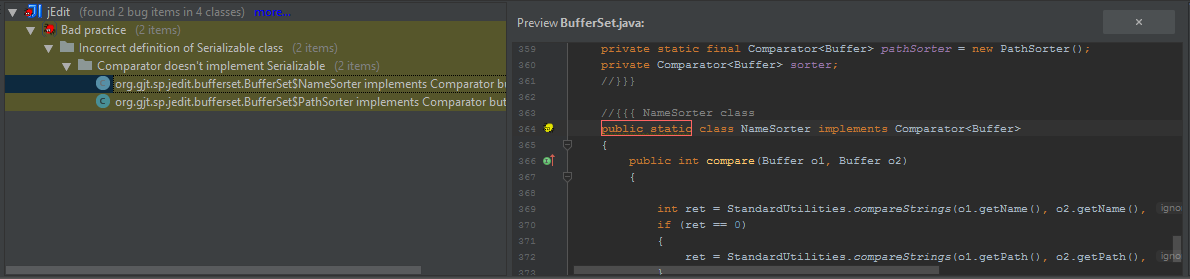
#### Bad practice (Incorrect definition of Serializable class)

#### SE\_COMPARATOR\_SHOULD\_BE\_SERIALIZABLE (Comparator doesn't implement Serializable)

**Notes:**

#### In class org.gjt.sp.jedit.bufferset.BufferSet$NameSorter

#### ComparatorIdiom (Se)



**gjt/sp/jedit/print/PrinterDialog.java**

**Bug Details:**

#### PrinterDialog.ALL isn't final but should be

**Class:**

#### PrinterDialog (org.gjt.sp.jedit.print) line 78

**Field:**

#### ALL

**Priority:**

#### High Confidence Malicious code vulnerability

**Problem classification:**

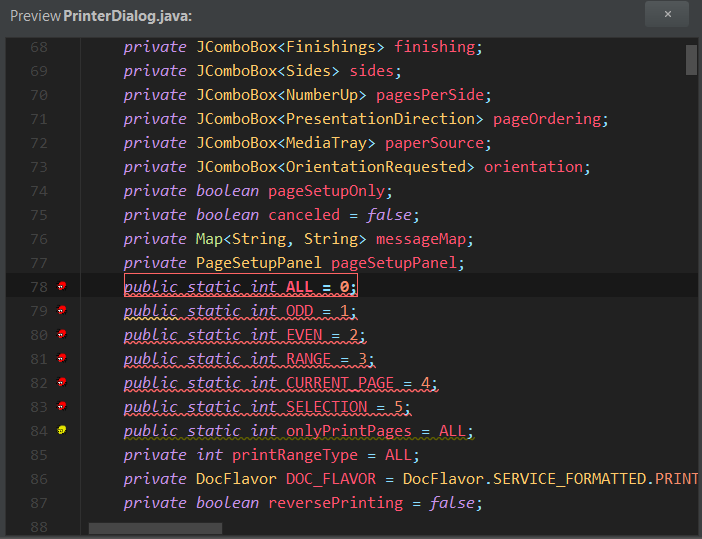
#### Malicious code vulnerability (Mutable static field)

#### MS\_SHOULD\_BE\_FINAL (Field isn't final but should be)

**Notes:**

Field org.gjt.sp.jedit.print.PrinterDialog.ALL

MutableStaticFields (MS)



**gjt/sp/jedit/jEdit.java**

**Bug Details:**

#### initUserProperties() may fail to clean up java.io.InputStream

**Class:**

#### jEdit (org.gjt.sp.jedit) line 3787

**Method**:

#### initUserProperties (org.gjt.sp.jedit.jEdit.initUserProperties())

**Priority:**

#### Medium Confidence Experimental

**Problem classification:**

#### Experimental (Unsatisfied obligation to clean up stream or resource)

#### OBL\_UNSATISFIED\_OBLIGATION (Method may fail to clean up stream or resource)

**Notes:**

Obligation to clean up resource created at jEdit.java:[line 3787] is not discharged

Reference type java.io.InputStream

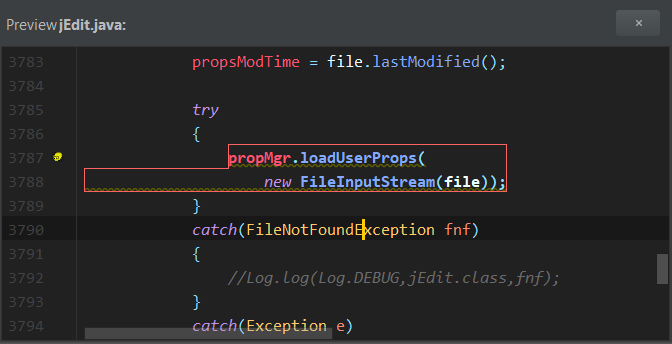
1 instances of obligation remaining

Path continues at jEdit.java:[line 3797]

Path continues at jEdit.java:[line 3799]

Remaining obligations: {InputStream x 1}

FindUnsatisfiedObligation (OBL)



**gjt/sp/jedit/View.java**

**Bug Details:**

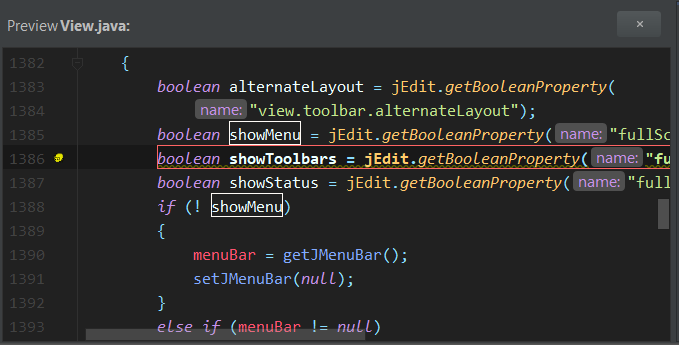
#### Dead store to showToolbars

**Class:**

#### View (org.gjt.sp.jedit) line 1386

**Method:**

#### updateFullScreenProps (org.gjt.sp.jedit.View.updateFullScreenProps())



**gjt/sp/jedit/SplitConfigParser.java**

**Bug Details:**

#### Should SplitConfigParser$BufferSet be a \_static\_ inner class?

**Class:**

#### SplitConfigParser$BufferSet (org.gjt.sp.jedit) lines 204-288

**Priority:**

#### Medium Confidence Performance

**Problem classification:**

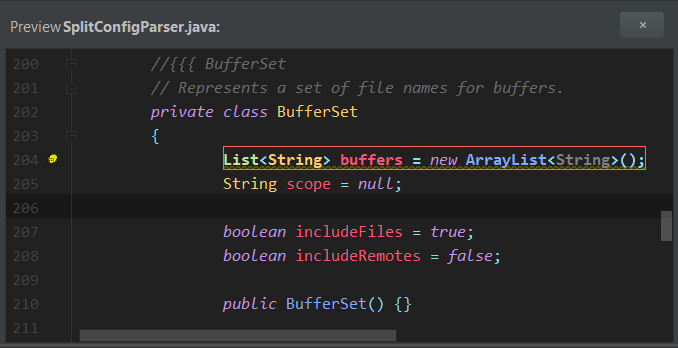
#### Performance (Inner class could be made static)

#### SIC\_INNER\_SHOULD\_BE\_STATIC (Should be a static inner class)

**Notes:**

#### In class org.gjt.sp.jedit.SplitConfigParser$BufferSet

#### UnreadFields (NP|SIC|SS|ST|UrF|UuF|UwF)



**gjt/sp/jedit/EditPane.java**

### 

### Bug Details:

#### Switch statement found where default case is missing

### Class:

#### EditPane$StatusHandler (org.gjt.sp.jedit) lines 1194-1207

### Method:

#### statusChanged (org.gjt.sp.jedit.EditPane$StatusHandler.statusChanged(TextArea, int, boolean))

### Priority:

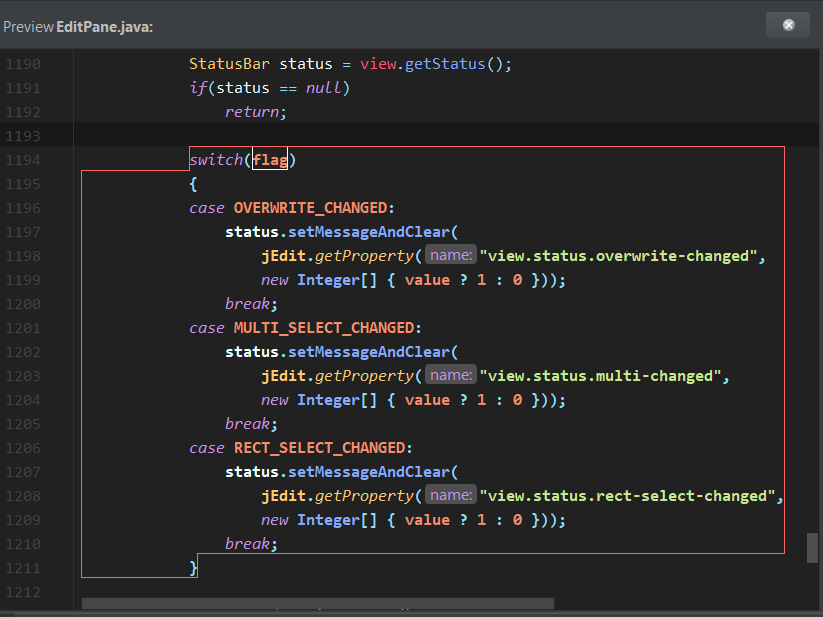
#### Medium Confidence Dodgy code

### Problem classification:

#### Dodgy code (Switch case falls through)

#### SF\_SWITCH\_NO\_DEFAULT (Switch statement found where default case is missing)

#### SwitchFallthrough (SF)



**org/gjt/sp/jedit/Registers.java**

**Bug Details:**

#### Public static getRegisters() may expose internal representation by returning Registers.registers

**Class:**

#### Registers (org.gjt.sp.jedit) line 535

**Method:**

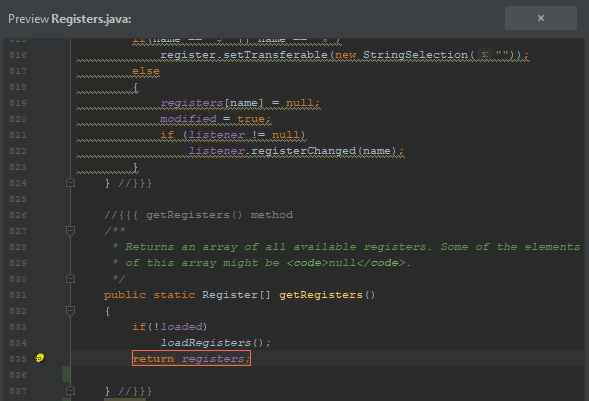
#### getRegisters (org.gjt.sp.jedit.Registers.getRegisters())

**Priority:**

#### Medium Confidence Dodgy code

**Comments:**

A public static method returns a reference to an array that is part of the static state of the class. Any code that calls this method can freely modify the underlying array. One fix is to return a copy of the array.



|  |  |
| --- | --- |
| Bug: | Call to static DateFormat (Sun Bug #6231579 and Sun Bug #6178997) |
| Locations: | org.gjt.sp.jedit.io.FileVFS$LocalFile.getExtendedAttribute(String)  org.gjt.sp.util.Log.\_log(int, String, String) |
| Classification: | High Confidence Multithreaded Correctness |
| Original Code: |  |
| New Code: |  |
| Fix Description: | Encapsulated the DateFormat objects in java’s Synchronized() block; which ensures only one thread can access the object at a time |

**FindBugs Results**

|  |  |
| --- | --- |
| Before fixes | After fixes |
|  |  |