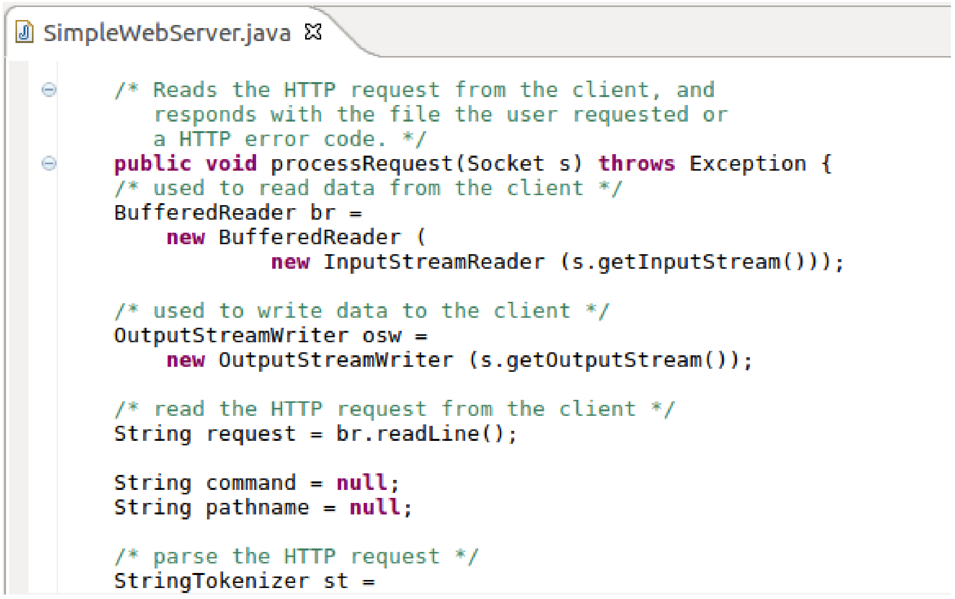
Assignment 10 :- Understanding and Using Static Code Analysis Tools

Name :- Aparna Krishna Bhat

ID :- 1001255079

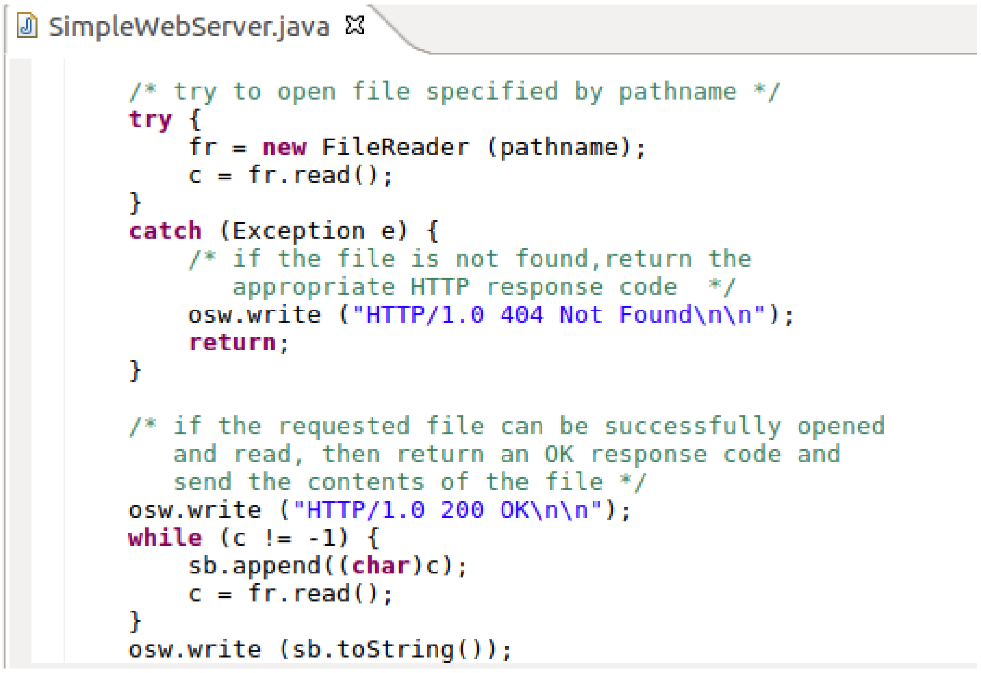
<u>Part 1</u>

The aim of this task is to use research tools like SpotBugs and SonarLint to examine a web server code.



SimpleWebServer.java

```java
/* Reads the HTTP request from the client, and
   responds with the file the user requested or
   a HTTP error code. */
public void processRequest(Socket s) throws Exception {
/* used to read data from the client */
BufferedReader br =
    new BufferedReader (
            new InputStreamReader (s.getInputStream()));

/* used to write data to the client */
OutputStreamWriter osw =
    new OutputStreamWriter (s.getOutputStream());

/* read the HTTP request from the client */
String request = br.readLine();

String command = null;
String pathname = null;

/* parse the HTTP request */
StringTokenizer st =
```



SimpleWebServer.java

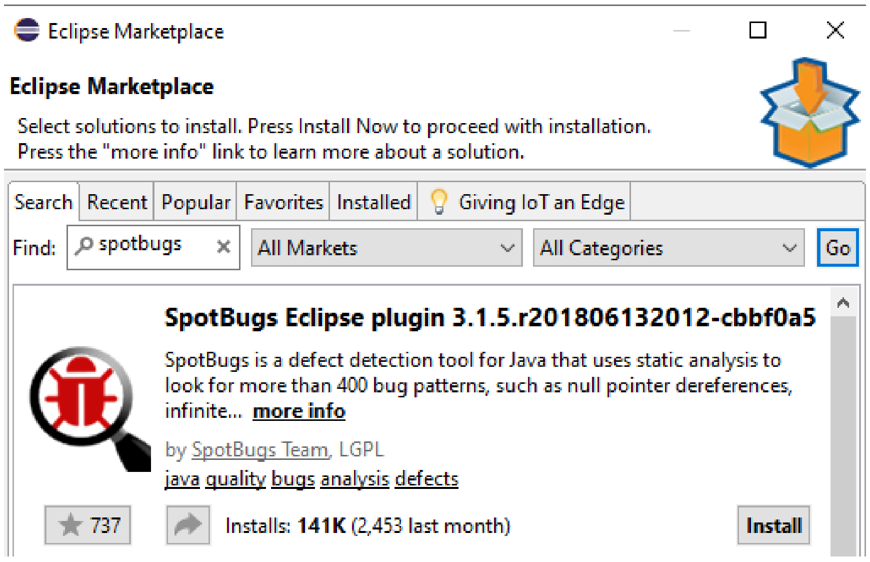
```java
/* try to open file specified by pathname */
try {
    fr = new FileReader (pathname);
    c = fr.read();
}
catch (Exception e) {
    /* if the file is not found,return the
       appropriate HTTP response code  */
    osw.write ("HTTP/1.0 404 Not Found\n\n");
    return;
}

/* if the requested file can be successfully opened
   and read, then return an OK response code and
   send the contents of the file */
osw.write ("HTTP/1.0 200 OK\n\n");
while (c != -1) {
    sb.append((char)c);
    c = fr.read();
}
osw.write (sb.toString());
```

Manual Analysis :- The code is initially examined to see if any issues can be found. We can see from the code for SimpleWebServer.java that the BufferedReader object br generates an InputStream and uses it but does not close it. Close the streams at all times. While a FileReader object has been developed, it has not been closed. There may be performance problems, garbage value selection, and incorrect undesired output if these two artifacts are not closed.

Tool Choices/Versions

SpotBugs and SonarLint are the two resources I've selected.

1) **SpotBugs Eclipse plugin**



Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

SpotBugs Eclipse plugin 3.1.5.r201806132012-cbbf0a5

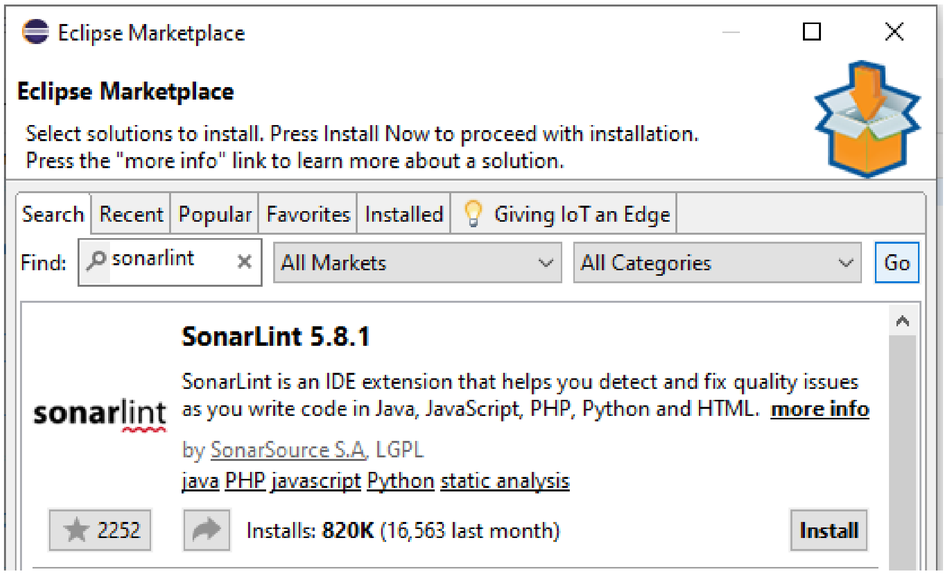
SpotBugs is a defect detection tool for Java that uses static analysis to look for more than 400 bug patterns, such as null pointer dereferences, infinite... more info

by SpotBugs Team, LGPL

java quality bugs analysis defects

737 Installs: **141K** (2,453 last month)

2) **SonarLint Eclipse plugin 5.8.1**



Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

SonarLint 5.8.1

SonarLint is an IDE extension that helps you detect and fix quality issues as you write code in Java, JavaScript, PHP, Python and HTML. more info

by SonarSource S.A, LGPL

java PHP javascript Python static analysis

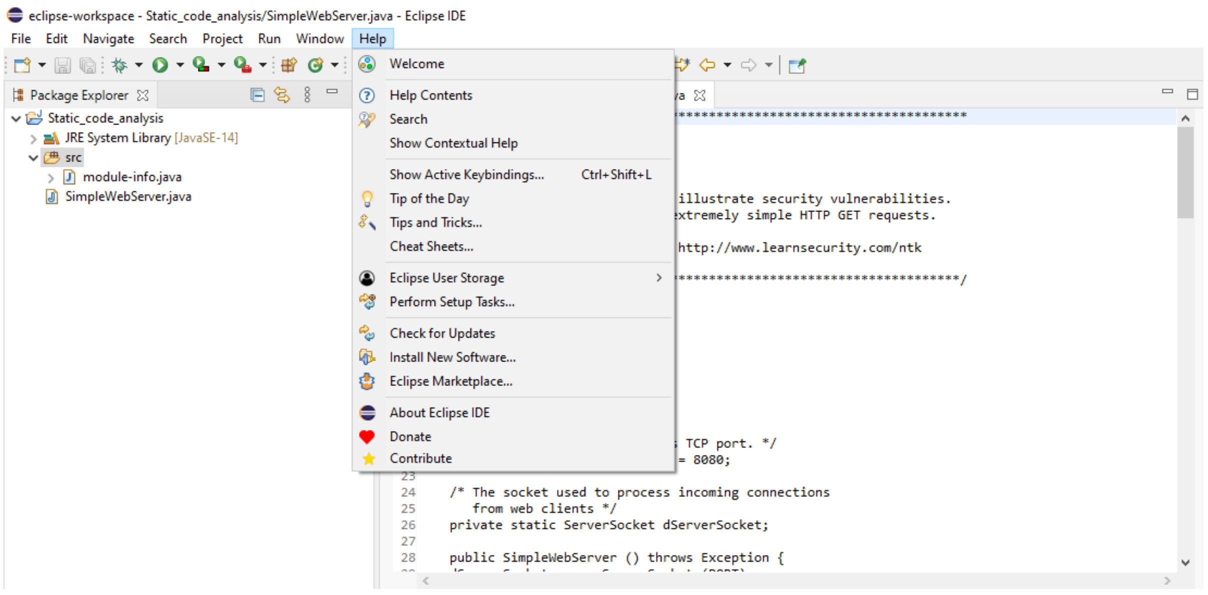
2252 Installs: **820K** (16,563 last month)

Tool Invocation Process

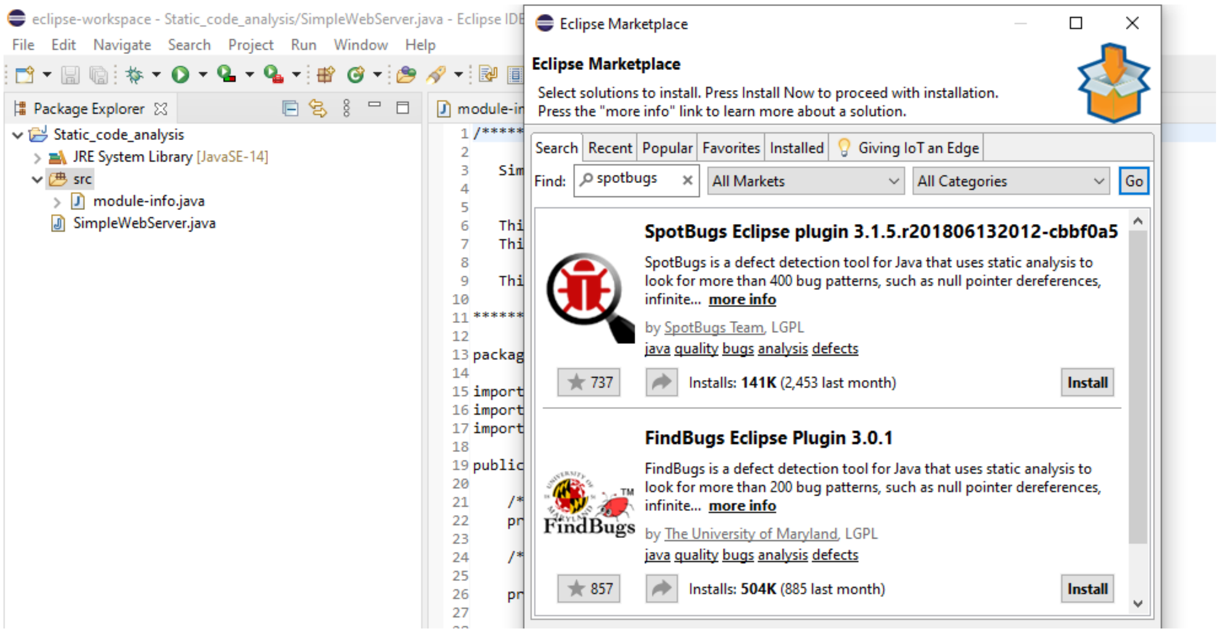
A) SpotBugs

Go to Help in the Eclipse IDE and look for Elipse Marketplace. Then look for SpotBugs and download it.

Home -> Help -> Eclipse Marketplace -> Search Spotbugs -> Install



```
****************************************
illustrate security vulnerabilities.
xtremely simple HTTP GET requests.
http://www.learnsecurity.com/ntk
**************************************/
 TCP port. */
= 8080;

    /* The socket used to process incoming connections
       from web clients */
    private static ServerSocket dServerSocket;

    public SimpleWebServer () throws Exception {
```


eclipse-workspace - Static_code_analysis/SimpleWebServer.java - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Package Explorer
Static_code_analysis
JRE System Library [JavaSE-14]
src
module-info.java
SimpleWebServer.java
Eclipse Marketplace
Search Recent Popular Favorites Installed Giving IoT an Edge
All Markets
All Categories
Go

Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Find: spotbugs

SpotBugs Eclipse plugin 3.1.5.r201806132012-cbbf0a5

SpotBugs is a defect detection tool for Java that uses static analysis to look for more than 400 bug patterns, such as null pointer dereferences, infinite... more info

by SpotBugs Team, LGPL

java quality bugs analysis defects

737 Installs: 141K (2,453 last month) Install

FindBugs Eclipse Plugin 3.0.1

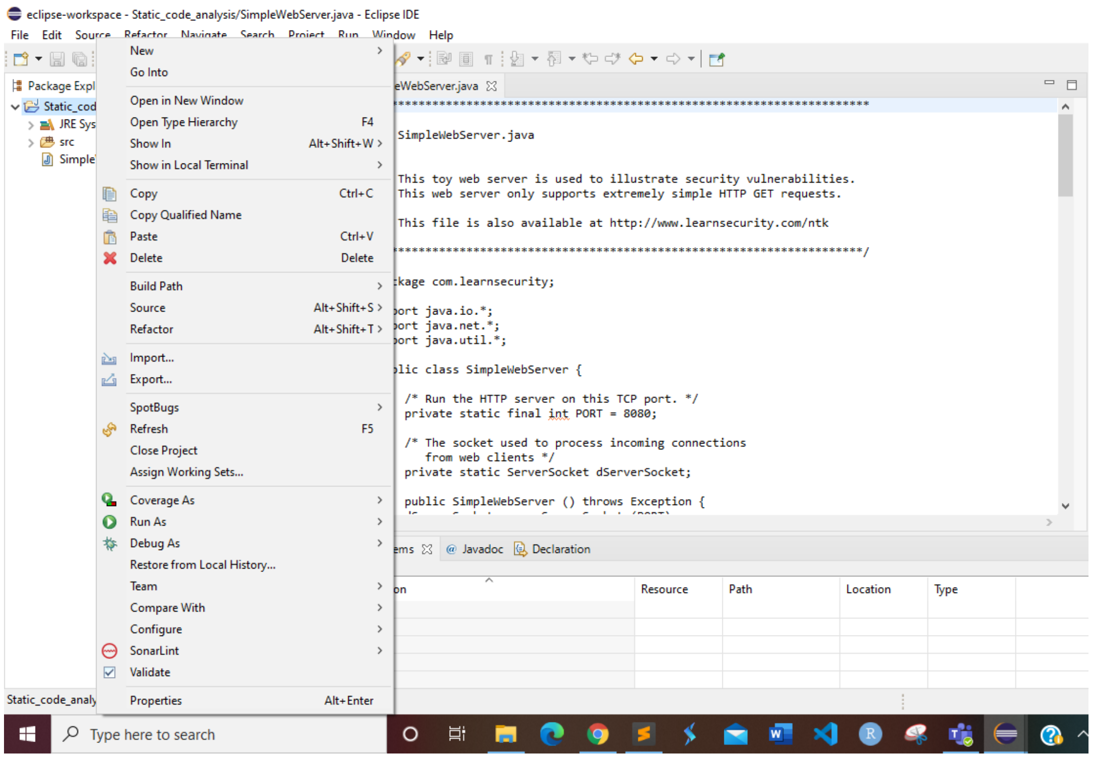
FindBugs is a defect detection tool for Java that uses static analysis to look for more than 200 bug patterns, such as null pointer dereferences, infinite... more info

by The University of Maryland, LGPL

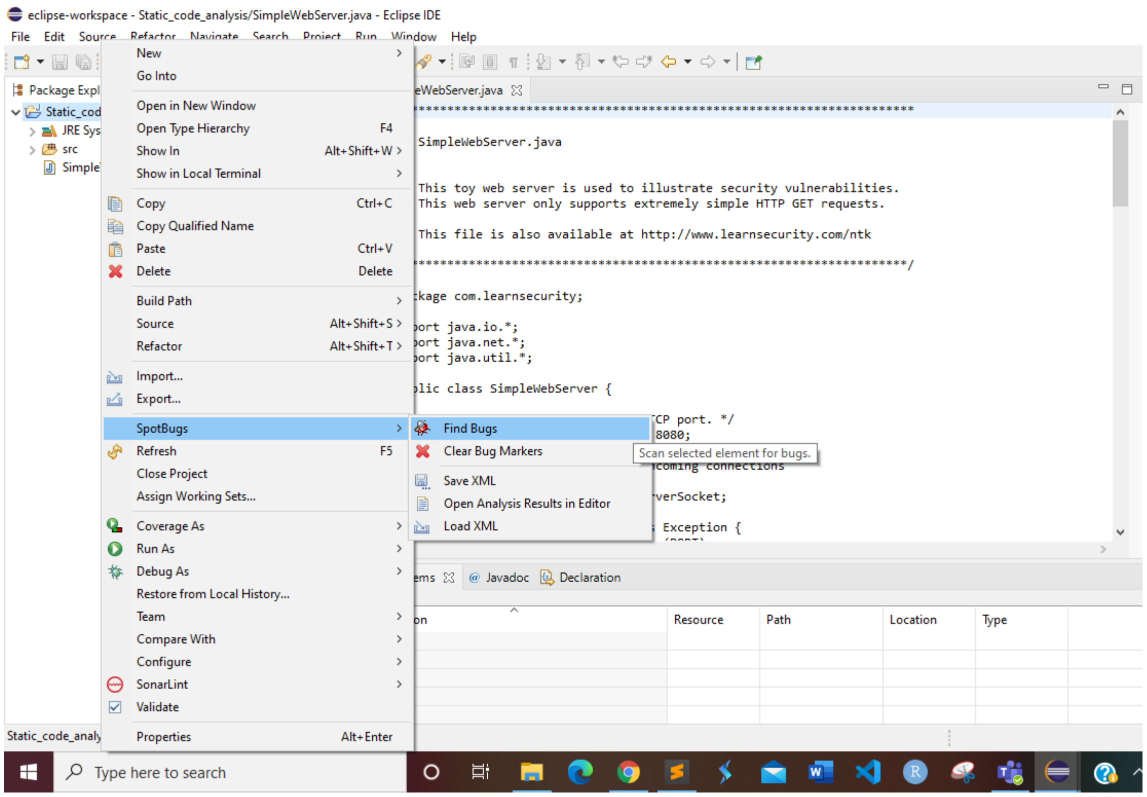
java quality bugs analysis defects

857 Installs: 504K (885 last month) Install

Right-click on the project in which the SimpleWebServer.java file is located. SpotBugs is an option that we can see. We get Find Bugs if we press that. We will discover all of the bugs in the software by clicking on Find Bugs.

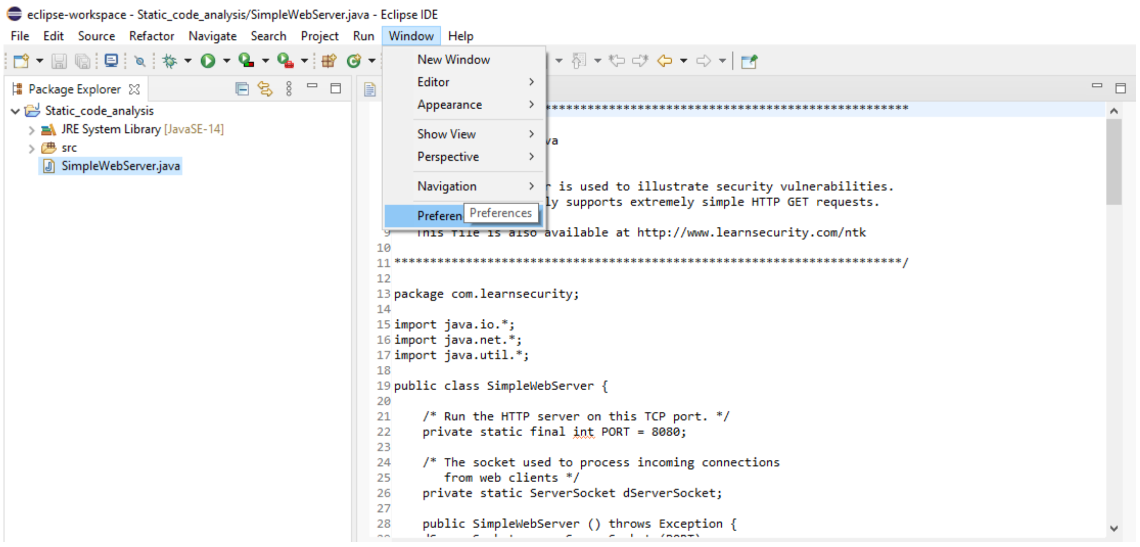


```
*********************************************************************
SimpleWebServer.java

This toy web server is used to illustrate security vulnerabilities.
This web server only supports extremely simple HTTP GET requests.

This file is also available at http://www.learnsecurity.com/ntk

*********************************************************************/
kage com.learnsecurity;

ort java.io.*;
ort java.net.*;
ort java.util.*;

lic class SimpleWebServer {

   /* Run the HTTP server on this TCP port. */
   private static final int PORT = 8080;

   /* The socket used to process incoming connections
      from web clients */
   private static ServerSocket dServerSocket;

   public SimpleWebServer () throws Exception {
```

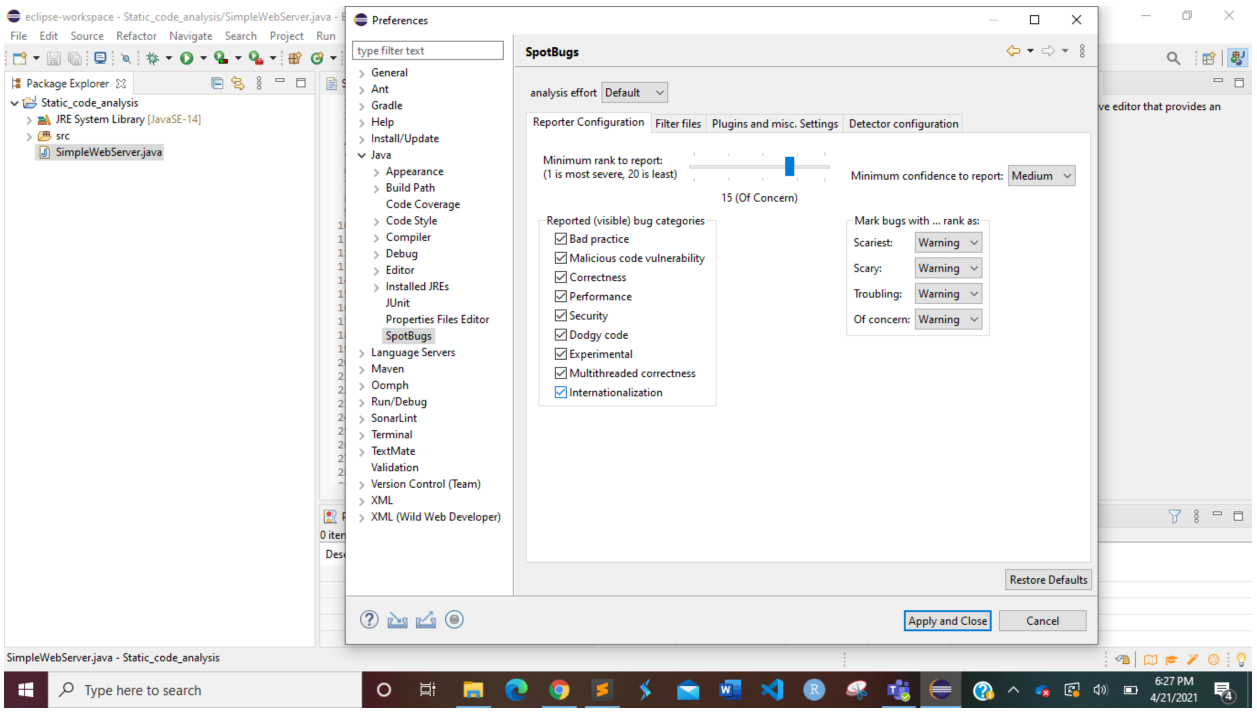


```
*********************************************************************
SimpleWebServer.java

This toy web server is used to illustrate security vulnerabilities.
This web server only supports extremely simple HTTP GET requests.

This file is also available at http://www.learnsecurity.com/ntk

*********************************************************************/
kage com.learnsecurity;

ort java.io.*;
ort java.net.*;
ort java.util.*;

lic class SimpleWebServer {

                        CP port. */
                        8080;

                        coming connections

                        verSocket;

                        Exception {
```

We go to Window -> Preferences in the Eclipse IDE. Then, in Preferences, go to Java -> SpotBugs and make sure that all of the check boxes for Reported Bug categories in Reported Configuration and Detector Configuration are checked. Using SpotBugs, we want to find all sorts of bugs in the software.


eclipse-workspace - Static_code_analysis/SimpleWebServer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Static_code_analysis
JRE System Library [JavaSE-14]
src
SimpleWebServer.java
New Window
Editor
Appearance
Show View
Perspective
Navigation
Preferences

```
******************************************************
a

r is used to illustrate security vulnerabilities.
ly supports extremely simple HTTP GET requests.
This file is also available at http://www.learnsecurity.com/ntk

*********************************************************************/

package com.learnsecurity;

import java.io.*;
import java.net.*;
import java.util.*;

public class SimpleWebServer {

    /* Run the HTTP server on this TCP port. */
    private static final int PORT = 8080;

    /* The socket used to process incoming connections
       from web clients */
    private static ServerSocket dServerSocket;

    public SimpleWebServer () throws Exception {
```



SpotBugs

analysis effort: Default

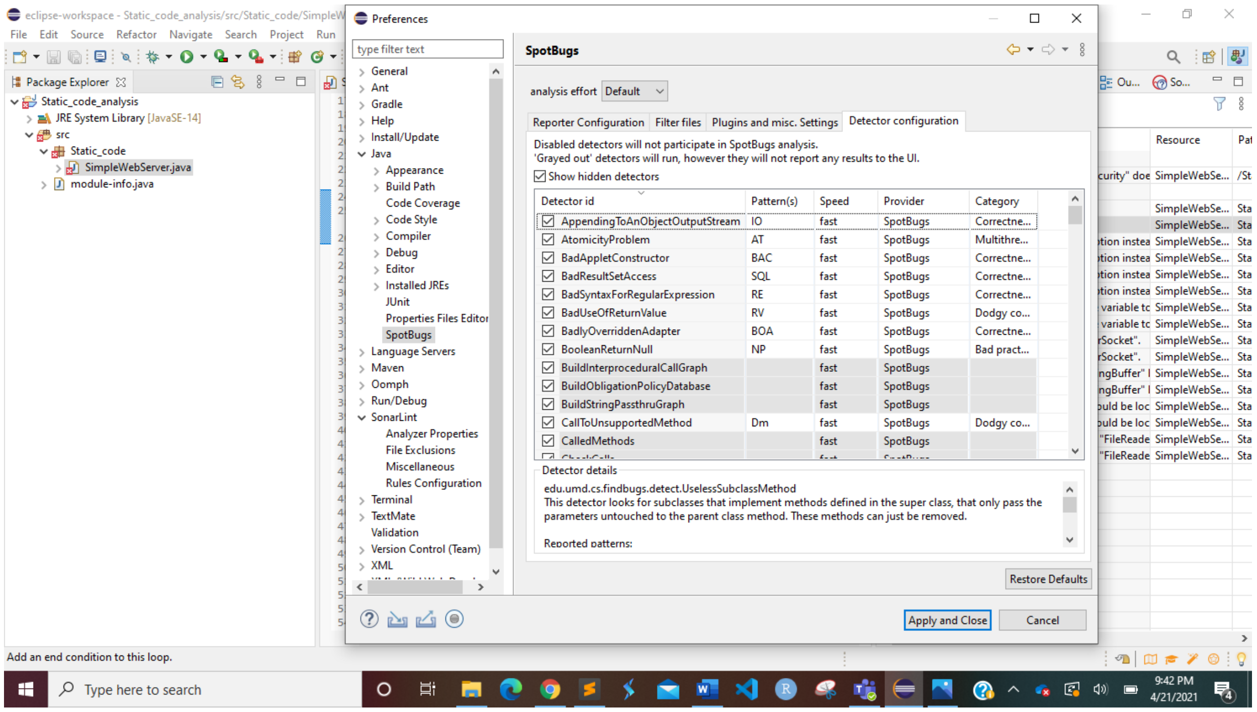
Minimum rank to report:
(1 is most severe, 20 is least)
15 (Of Concern)

Minimum confidence to report: Medium

Reported (visible) bug categories
- [x] Bad practice
- [x] Malicious code vulnerability
- [x] Correctness
- [x] Performance
- [x] Security
- [x] Dodgy code
- [x] Experimental
- [x] Multithreaded correctness
- [x] Internationalization

Mark bugs with ... rank as:

Scariest:	Warning
Scary:	Warning
Troubling:	Warning
Of concern:	Warning



SpotBugs

analysis effort: Default

Detector configuration

Disabled detectors will not participate in SpotBugs analysis.
'Grayed out' detectors will run, however they will not report any results to the UI.

☑ Show hidden detectors

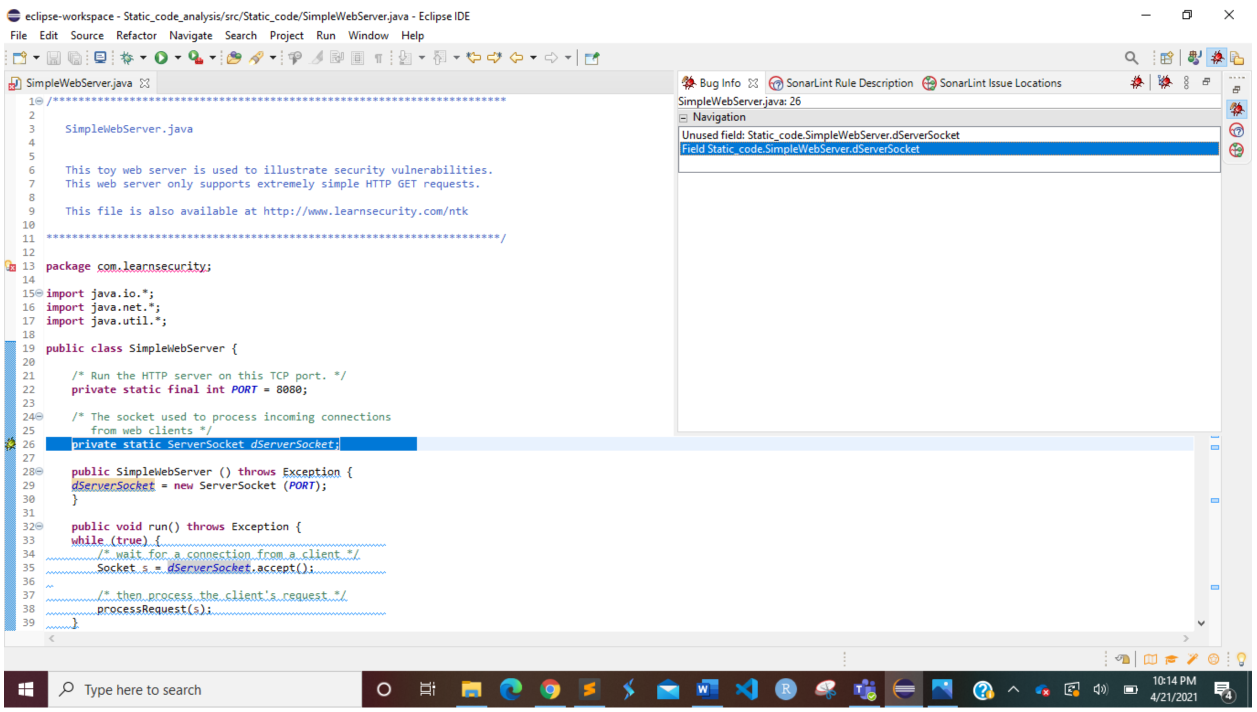
Detector id	Pattern(s)	Speed	Provider	Category
☑ AppendingToAnObjectOutputStream	IO	fast	SpotBugs	Correctne...
☑ AtomicityProblem	AT	fast	SpotBugs	Multithre...
☑ BadAppletConstructor	BAC	fast	SpotBugs	Correctne...
☑ BadResultSetAccess	SQL	fast	SpotBugs	Correctne...
☑ BadSyntaxForRegularExpression	RE	fast	SpotBugs	Correctne...
☑ BadUseOfReturnValue	RV	fast	SpotBugs	Dodgy co...
☑ BadlyOverriddenAdapter	BOA	fast	SpotBugs	Correctne...
☑ BooleanReturnNull	NP	fast	SpotBugs	Bad pract...
☑ BuildInterproceduralCallGraph		fast	SpotBugs	
☑ BuildObligationPolicyDatabase		fast	SpotBugs	
☑ BuildStringPassthruGraph		fast	SpotBugs	
☑ CallToUnsupportedMethod	Dm	fast	SpotBugs	Dodgy co...
☑ CalledMethods		fast	SpotBugs	

Detector details

edu.umd.cs.findbugs.detect.UselessSubclassMethod
This detector looks for subclasses that implement methods defined in the super class, that only pass the parameters untouched to the parent class method. These methods can just be removed.

Reported patterns:

Next, by right-clicking on SimpleWebServer.java and selecting SpotBugs -> Find Bugs, we can now find all the bugs.



eclipse-workspace - Static_code_analysis/src/Static_code/SimpleWebServer.java - Eclipse IDE

```
1  /***********************************************************************
2
3     SimpleWebServer.java
4
5
6     This toy web server is used to illustrate security vulnerabilities.
7     This web server only supports extremely simple HTTP GET requests.
8
9     This file is also available at http://www.learnsecurity.com/ntk
10
11 ***********************************************************************/
12
13 package com.learnsecurity;
14
15 import java.io.*;
16 import java.net.*;
17 import java.util.*;
18
19 public class SimpleWebServer {
20
21     /* Run the HTTP server on this TCP port. */
22     private static final int PORT = 8080;
23
24     /* The socket used to process incoming connections
25        from web clients */
26     private static ServerSocket dServerSocket;
27
28     public SimpleWebServer () throws Exception {
29     dServerSocket = new ServerSocket (PORT);
30     }
31
32     public void run() throws Exception {
33     while (true) {
34         /* wait for a connection from a client */
35         Socket s = dServerSocket.accept();
36
37         /* then process the client's request */
38         processRequest(s);
39     }
```

Bug Info

SimpleWebServer.java: 26

Navigation

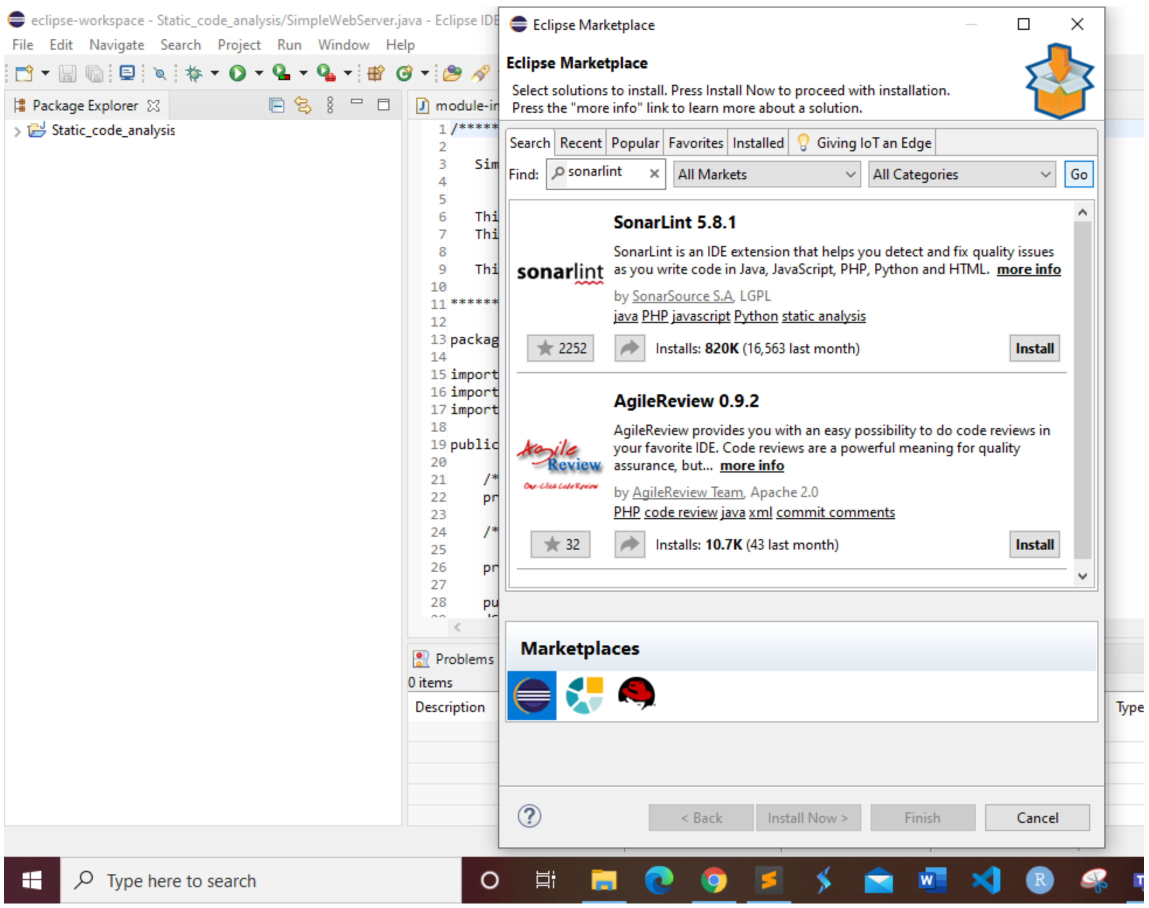
Unused field: Static_code.SimpleWebServer.dServerSocket
Field Static_code.SimpleWebServer.dServerSocket

dServerSocket has been declared as a static variable on line 26. This leads to a bug on line 29, where we attempt to write the instance method new ServerSocket (PORT) to a static variable. A static field is written by the instance process. If several instances are being exploited, this is difficult to get correct, and it's usually not a good a practice.

B) SonarLint

Navigate to Help in the Eclipse IDE and look for Eclipse Marketplace. Then search for SonarLint and install it.

Help -> Eclipse MarketPlace -> Search Sonarlint -> Install


Eclipse Marketplace

Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Search | Recent | Popular | Favorites | Installed | Giving IoT an Edge

Find: sonarlint | All Markets | All Categories | Go

SonarLint 5.8.1

SonarLint is an IDE extension that helps you detect and fix quality issues as you write code in Java, JavaScript, PHP, Python and HTML. **more info**

by SonarSource S.A, LGPL
java PHP javascript Python static analysis

2252 | Installs: **820K** (16,563 last month) | **Install**

AgileReview 0.9.2

AgileReview provides you with an easy possibility to do code reviews in your favorite IDE. Code reviews are a powerful meaning for quality assurance, but... **more info**

by AgileReview Team, Apache 2.0
PHP code review java xml commit comments

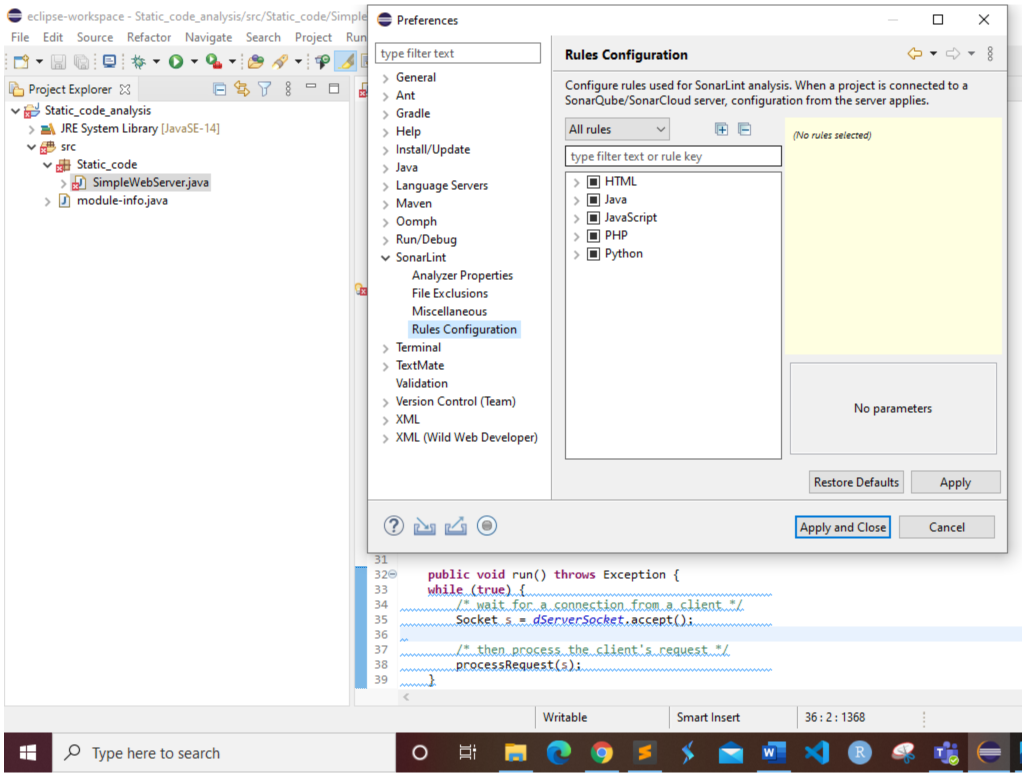
32 | Installs: **10.7K** (43 last month) | **Install**

Marketplaces

< Back | Install Now > | Finish | Cancel

Right-click on the project containing the SimpleWebServer.java package. SonarLint is a choice we can see. We have two choices when we press that: Analyze and Exclude. We will find all of the program's bugs by clicking on Analyze.

We go to Window -> Preferences in the Eclipse IDE. Then, in Preferences, go to Java -> SonarLint and double-check that all of the Java check boxes in Rules Configuration are checked. We want to use SonarLint to find all sorts of bugs in the software.


Rules Configuration

Configure rules used for SonarLint analysis. When a project is connected to a SonarQube/SonarCloud server, configuration from the server applies.

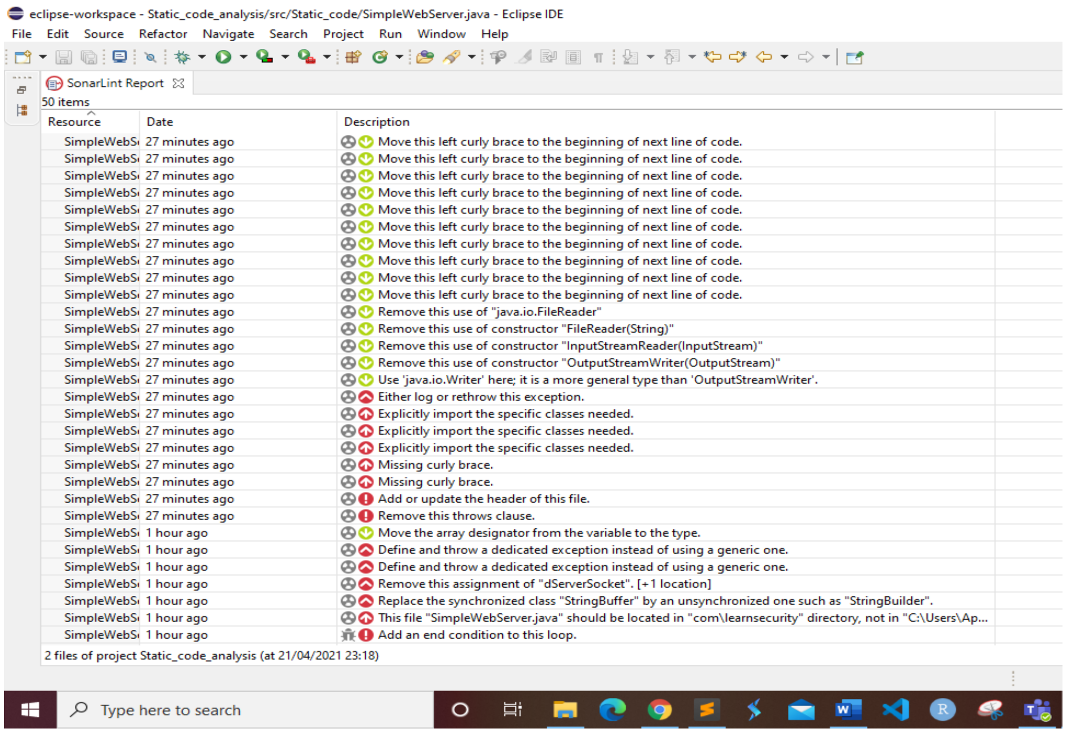
All rules

(No rules selected)

No parameters

```
public void run() throws Exception {
while (true) {
    /* wait for a connection from a client */
    Socket s = dServerSocket.accept();

    /* then process the client's request */
    processRequest(s);
}
```

We'll be able to find all types of bugs in the entire software once all of the rules are installed during SonarLint review. This search is performed to ensure that the program does not contain any bugs. By right-clicking on SimpleWebServer.java and selecting SonarLint -> Analyze, we can now find all of the bugs.


eclipse-workspace - Static_code_analysis/src/Static_code/SimpleWebServer.java - Eclipse IDE

SonarLint Report

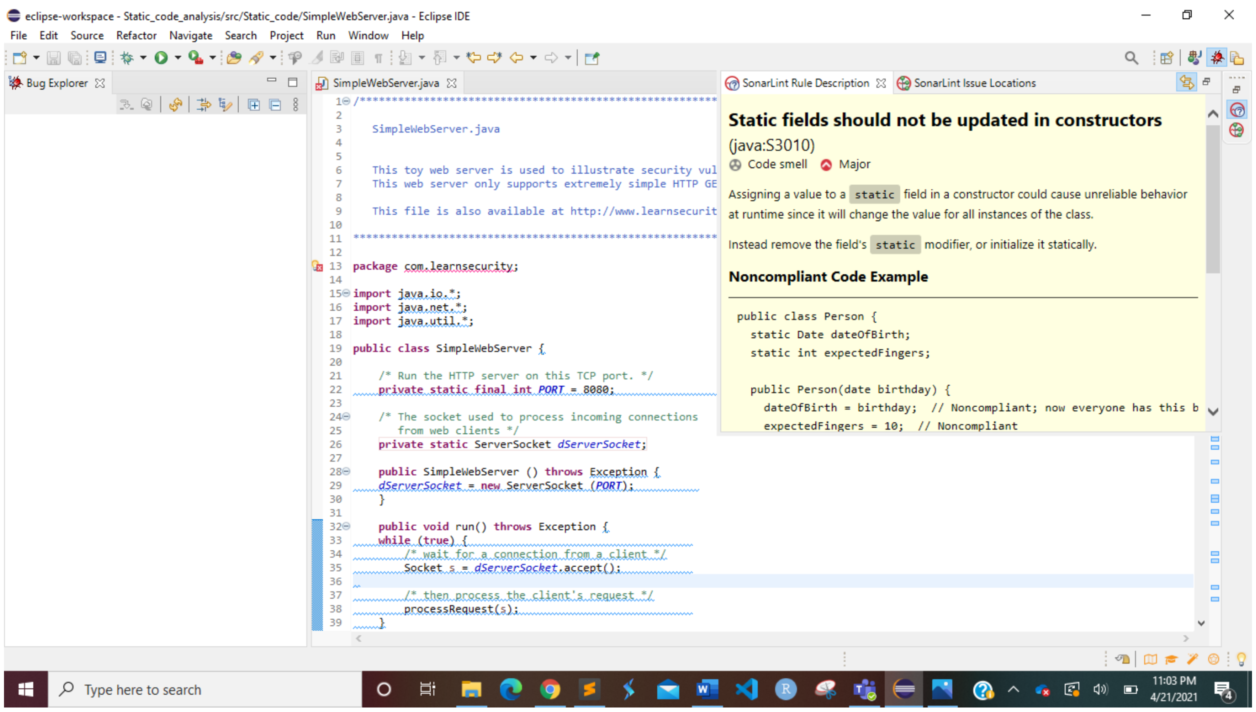
50 items

Resource	Date	Description
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Move this left curly brace to the beginning of next line of code.
SimpleWebS	27 minutes ago	Remove this use of "java.io.FileReader"
SimpleWebS	27 minutes ago	Remove this use of constructor "FileReader(String)"
SimpleWebS	27 minutes ago	Remove this use of constructor "InputStreamReader(InputStream)"
SimpleWebS	27 minutes ago	Remove this use of constructor "OutputStreamWriter(OutputStream)"
SimpleWebS	27 minutes ago	Use 'java.io.Writer' here; it is a more general type than 'OutputStreamWriter'.
SimpleWebS	27 minutes ago	Either log or rethrow this exception.
SimpleWebS	27 minutes ago	Explicitly import the specific classes needed.
SimpleWebS	27 minutes ago	Explicitly import the specific classes needed.
SimpleWebS	27 minutes ago	Explicitly import the specific classes needed.
SimpleWebS	27 minutes ago	Missing curly brace.
SimpleWebS	27 minutes ago	Missing curly brace.
SimpleWebS	27 minutes ago	Add or update the header of this file.
SimpleWebS	27 minutes ago	Remove this throws clause.
SimpleWebS	1 hour ago	Move the array designator from the variable to the type.
SimpleWebS	1 hour ago	Define and throw a dedicated exception instead of using a generic one.
SimpleWebS	1 hour ago	Define and throw a dedicated exception instead of using a generic one.
SimpleWebS	1 hour ago	Remove this assignment of "dServerSocket". [+1 location]
SimpleWebS	1 hour ago	Replace the synchronized class "StringBuffer" by an unsynchronized one such as "StringBuilder".
SimpleWebS	1 hour ago	This file "SimpleWebServer.java" should be located in "com\learnsecurity" directory, not in "C:\Users\Ap...
SimpleWebS	1 hour ago	Add an end condition to this loop.

2 files of project Static_code_analysis (at 21/04/2021 23:18)

With the SonarLint tool, several new bugs and suggestions have been given and discovered. Here are a few of them:

1)


eclipse-workspace - Static_code_analysis/src/Static_code/SimpleWebServer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Bug Explorer
SimpleWebServer.java
SonarLint Rule Description
SonarLint Issue Locations
Type here to search
11:03 PM
4/21/2021

```
/*****************************************************************

   SimpleWebServer.java


   This toy web server is used to illustrate security vul
   This web server only supports extremely simple HTTP GE

   This file is also available at http://www.learnsecurit

*****************************************************************

package com.learnsecurity;

import java.io.*;
import java.net.*;
import java.util.*;

public class SimpleWebServer {

    /* Run the HTTP server on this TCP port. */
    private static final int PORT = 8080;

    /* The socket used to process incoming connections
       from web clients */
    private static ServerSocket dServerSocket;

    public SimpleWebServer () throws Exception {
    dServerSocket = new ServerSocket (PORT);
    }

    public void run() throws Exception {
    while (true) {
        /* wait for a connection from a client */
        Socket s = dServerSocket.accept();

        /* then process the client's request */
        processRequest(s);
    }
```

Static fields should not be updated in constructors
(java:S3010)
Code smell Major

Assigning a value to a `static` field in a constructor could cause unreliable behavior at runtime since it will change the value for all instances of the class.

Instead remove the field's `static` modifier, or initialize it statically.

Noncompliant Code Example

```
public class Person {
  static Date dateOfBirth;
  static int expectedFingers;

  public Person(date birthday) {
    dateOfBirth = birthday;  // Noncompliant; now everyone has this b
    expectedFingers = 10;  // Noncompliant
```

2) Explicitly import the particular classes required – Because import java.util.* is a very common import and all libraries are imported, it is not a best practice or a safe practice. As a result, always define the exact library.
3) Missing curly brace – We must always use curly braces after if (condition) to define the boundaries or block of statements for the action to be taken when the condition is valid. Otherwise, only pone line is called operation after this condition.
4) Replace the synchronized "StringBuffer" class with an unsynchronized "StingBuilder" class – The methods in String Buffer are synchronized where possible, so that all operations on any given instance behave as if they happened in a sequential order that matches the order of the method calls made by each of the threads involved.

<u>Comparison/Contrast Tools</u>

Does the tool analyze source or binary as input?

SpotBugs, which is a continuation of FindBugs, analyzes binary data while others focus on the source code. It examines Java bytecode for bugs. SonarLint is an IDE extension that assists you in detecting and correcting code quality issues as you write it. SonarLint squiggles errors like a spell checker, allowing them to be corrected before committing code. The source is analyzed by

SonarLint. It simplifies the management of code. It ensures that the system is easy to manage and safe.

Which category of tools is it?

SpotBugs is a Java program that detects potential bugs. There are four levels of potential errors: ***scariest, scary, troubling, and of concern***. This is a warning to the developer about the potential effect or severity of their work.

SpotBugs belongs to the following tool categories:

1) Type checking
2) Bug detection
3) Security assessment

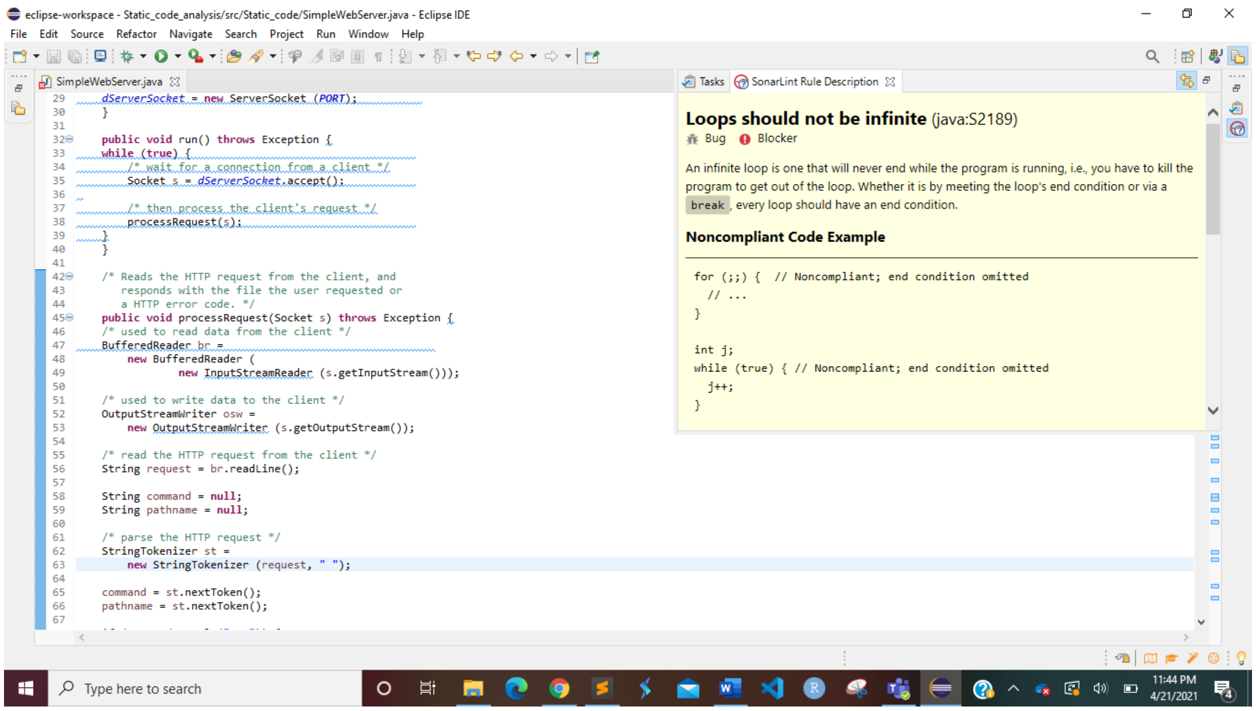
SonarLint is an IDE extension that assists you in detecting and resolving code quality issues as you write. It squiggles errors in code, just like a spell checker, so they can be corrected until the code is committed.

SonarLint is a tool that belongs to the following group:

1) Style checking
2) Bug finding
3) Security review

Show an example (if one exists) of a finding that is reported by one tool and not others?

As previously mentioned, SonarLint detects some new bugs in the software. The exception below is one of the findings recorded in SonarLint but not in SpotBugs. Add an end-condition to this loop – Since there is no condition to end the while(true) loop in line 33, it can go on indefinitely.


eclipse-workspace - Static_code_analysis/src/Static_code/SimpleWebServer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
SimpleWebServer.java
Tasks
SonarLint Rule Description
Type here to search
11:44 PM
4/21/2021

```
29      dServerSocket = new ServerSocket (PORT);
30      }
31
32      public void run() throws Exception {
33      while (true) {
34          /* wait for a connection from a client */
35          Socket s = dServerSocket.accept();
36
37          /* then process the client's request */
38          processRequest(s);
39      }
40      }
41
42      /* Reads the HTTP request from the client, and
43         responds with the file the user requested or
44         a HTTP error code. */
45      public void processRequest(Socket s) throws Exception {
46      /* used to read data from the client */
47      BufferedReader br =
48          new BufferedReader (
49                  new InputStreamReader (s.getInputStream()));
50
51      /* used to write data to the client */
52      OutputStreamWriter osw =
53          new OutputStreamWriter (s.getOutputStream());
54
55      /* read the HTTP request from the client */
56      String request = br.readLine();
57
58      String command = null;
59      String pathname = null;
60
61      /* parse the HTTP request */
62      StringTokenizer st =
63          new StringTokenizer (request, " ");
64
65      command = st.nextToken();
66      pathname = st.nextToken();
67
```

Loops should not be infinite (java:S2189)
Bug Blocker

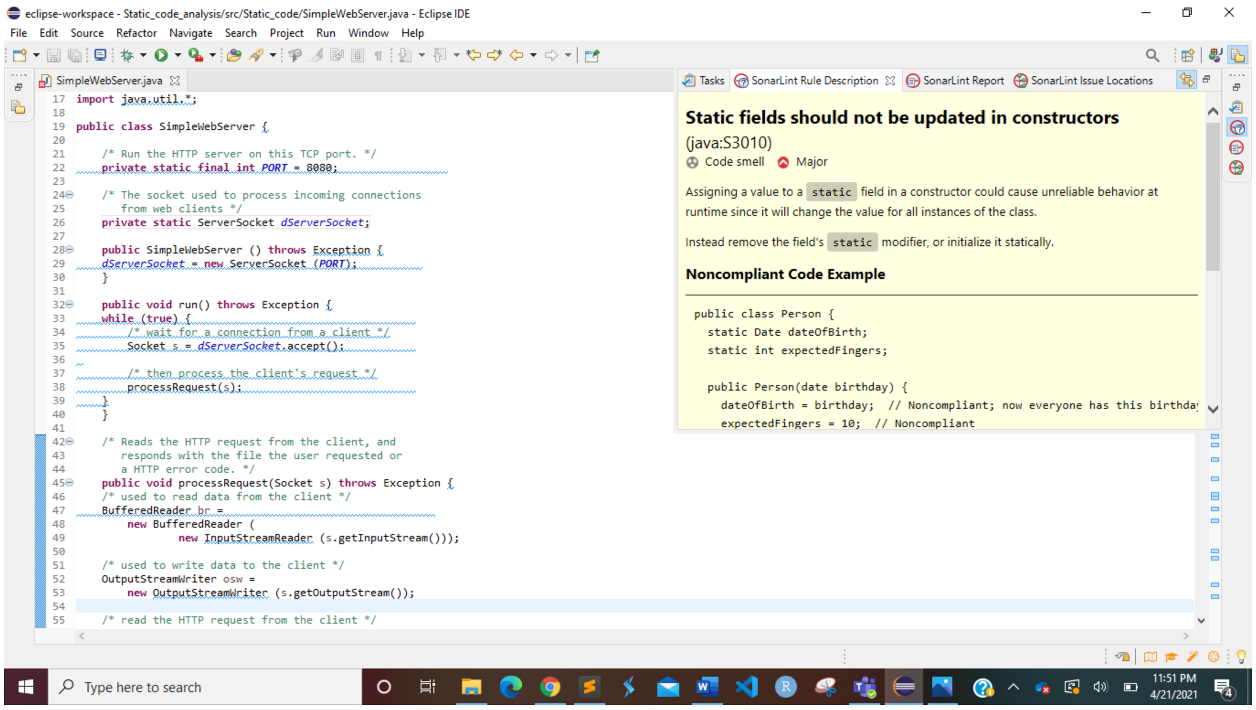
An infinite loop is one that will never end while the program is running, i.e., you have to kill the program to get out of the loop. Whether it is by meeting the loop's end condition or via a `break`, every loop should have an end condition.

Noncompliant Code Example

```
for (;;) {  // Noncompliant; end condition omitted
  // ...
}

int j;
while (true) { // Noncompliant; end condition omitted
  j++;
}
```

Show an example (if one exists) of a finding reported by multiple tools?

Remove this dServerSocket assignment from SonarLint. This bug also refers to the line 29 that was previously mentioned.


eclipse-workspace - Static_code_analysis/src/Static_code/SimpleWebServer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
SimpleWebServer.java
Tasks
SonarLint Rule Description
SonarLint Report
SonarLint Issue Locations
Type here to search
11:51 PM
4/21/2021

```
17  import java.util.*;
18
19  public class SimpleWebServer {
20
21      /* Run the HTTP server on this TCP port. */
22      private static final int PORT = 8080;
23
24      /* The socket used to process incoming connections
25         from web clients */
26      private static ServerSocket dServerSocket;
27
28      public SimpleWebServer () throws Exception {
29      dServerSocket = new ServerSocket (PORT);
30      }
31
32      public void run() throws Exception {
33      while (true) {
34          /* wait for a connection from a client */
35          Socket s = dServerSocket.accept();
36
37          /* then process the client's request */
38          processRequest(s);
39      }
40      }
41
42      /* Reads the HTTP request from the client, and
43         responds with the file the user requested or
44         a HTTP error code. */
45      public void processRequest(Socket s) throws Exception {
46      /* used to read data from the client */
47      BufferedReader br =
48          new BufferedReader (
49                  new InputStreamReader (s.getInputStream()));
50
51      /* used to write data to the client */
52      OutputStreamWriter osw =
53          new OutputStreamWriter (s.getOutputStream());
54
55      /* read the HTTP request from the client */
```

Static fields should not be updated in constructors (java:S3010)
Code smell Major

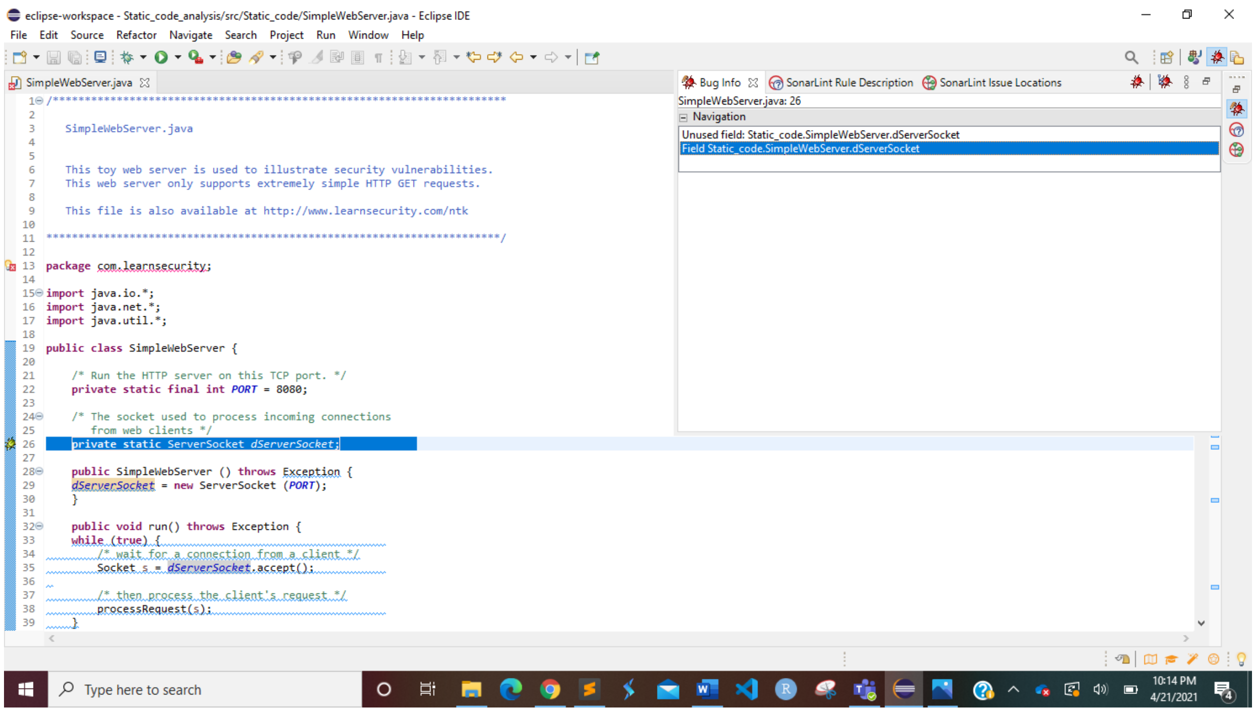
Assigning a value to a `static` field in a constructor could cause unreliable behavior at runtime since it will change the value for all instances of the class.

Instead remove the field's `static` modifier, or initialize it statically.

Noncompliant Code Example

```
public class Person {
  static Date dateOfBirth;
  static int expectedFingers;

  public Person(date birthday) {
    dateOfBirth = birthday;  // Noncompliant; now everyone has this birthda
    expectedFingers = 10;  // Noncompliant
```

Write to a static field from an instance method in SpotBugs. dServerSocket has been declared as a static variable on line 26. This leads to a bug on line 29, where we attempt to write the instance method new ServerSocket (PORT) to a static variable. A static field is written by the instance process. If several instances are being exploited, this is difficult to get correct, and it's usually poor practice.

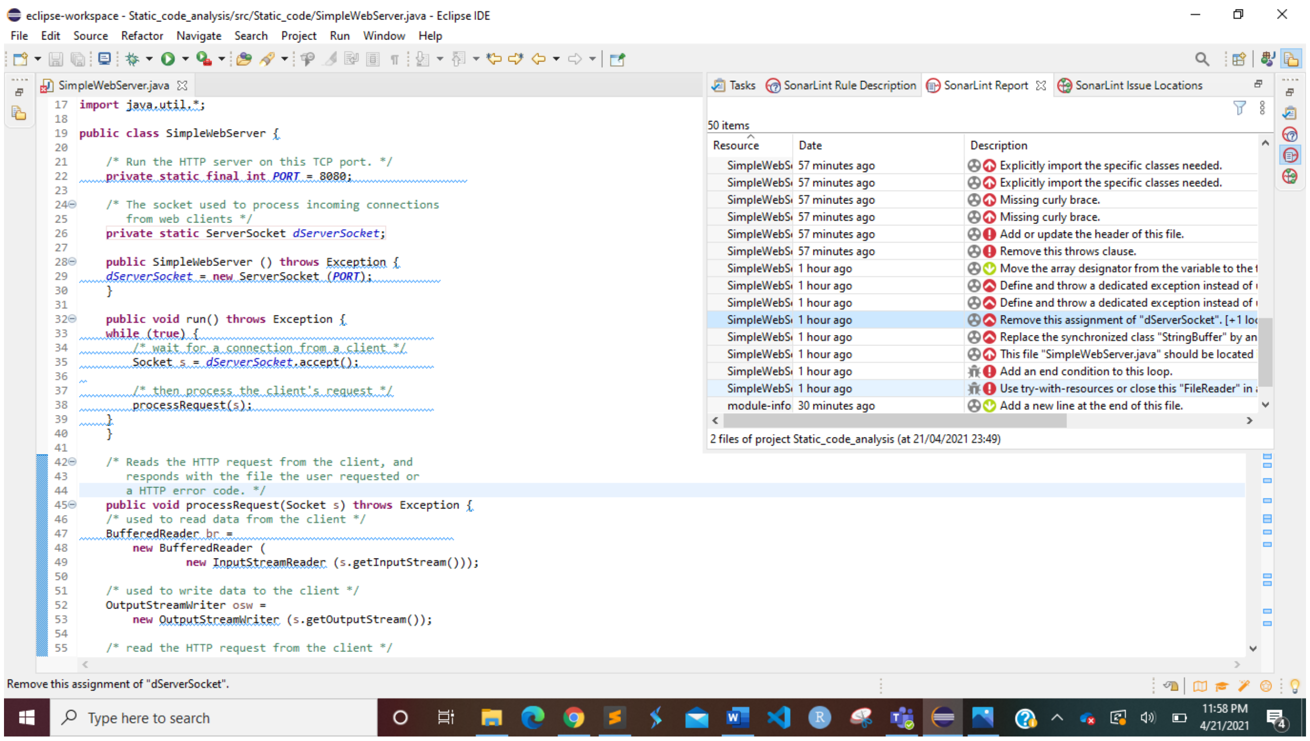

eclipse-workspace - Static_code_analysis/src/Static_code/SimpleWebServer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
SimpleWebServer.java
Bug Info
SonarLint Rule Description
SonarLint Issue Locations
Type here to search
10:14 PM
4/21/2021

```
 1 /************************************************************************
 2
 3    SimpleWebServer.java
 4
 5
 6    This toy web server is used to illustrate security vulnerabilities.
 7    This web server only supports extremely simple HTTP GET requests.
 8
 9    This file is also available at http://www.learnsecurity.com/ntk
10
11 ************************************************************************/
12
13 package com.learnsecurity;
14
15 import java.io.*;
16 import java.net.*;
17 import java.util.*;
18
19 public class SimpleWebServer {
20
21     /* Run the HTTP server on this TCP port. */
22     private static final int PORT = 8080;
23
24     /* The socket used to process incoming connections
25        from web clients */
26     private static ServerSocket dServerSocket;
27
28     public SimpleWebServer () throws Exception {
29     dServerSocket = new ServerSocket (PORT);
30     }
31
32     public void run() throws Exception {
33     while (true) {
34         /* wait for a connection from a client */
35         Socket s = dServerSocket.accept();
36
37         /* then process the client's request */
38         processRequest(s);
39     }
```

```
SimpleWebServer.java: 26
Navigation
Unused field: Static_code.SimpleWebServer.dServerSocket
Field Static_code.SimpleWebServer.dServerSocket
```

For the known flaw in the code used, document which tools reported it (true negative) and which tools did not (false positive)?

The known error, as discovered during the manual study, is that the FileReader object has been generated but not closed. Always closing the streams is a good coding practice to have because it prevents us from acting strangely. The exception "Use try-with-resources or close this FileReader in a finally clause" was used by SonarLint to find this bug. This error was not found by SpotBugs.


eclipse-workspace - Static_code_analysis/src/Static_code/SimpleWebServer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
SimpleWebServer.java
Tasks
SonarLint Rule Description
SonarLint Report
SonarLint Issue Locations
Type here to search
11:58 PM
4/21/2021

```java
import java.util.*;

public class SimpleWebServer {

    /* Run the HTTP server on this TCP port. */
    private static final int PORT = 8080;

    /* The socket used to process incoming connections
       from web clients */
    private static ServerSocket dServerSocket;

    public SimpleWebServer () throws Exception {
    dServerSocket = new ServerSocket (PORT);
    }

    public void run() throws Exception {
    while (true) {
        /* wait for a connection from a client */
        Socket s = dServerSocket.accept();

        /* then process the client's request */
        processRequest(s);
    }
    }

    /* Reads the HTTP request from the client, and
       responds with the file the user requested or
       a HTTP error code. */
    public void processRequest(Socket s) throws Exception {
    /* used to read data from the client */
    BufferedReader br =
        new BufferedReader (
                new InputStreamReader (s.getInputStream()));

    /* used to write data to the client */
    OutputStreamWriter osw =
        new OutputStreamWriter (s.getOutputStream());

    /* read the HTTP request from the client */
```

50 items

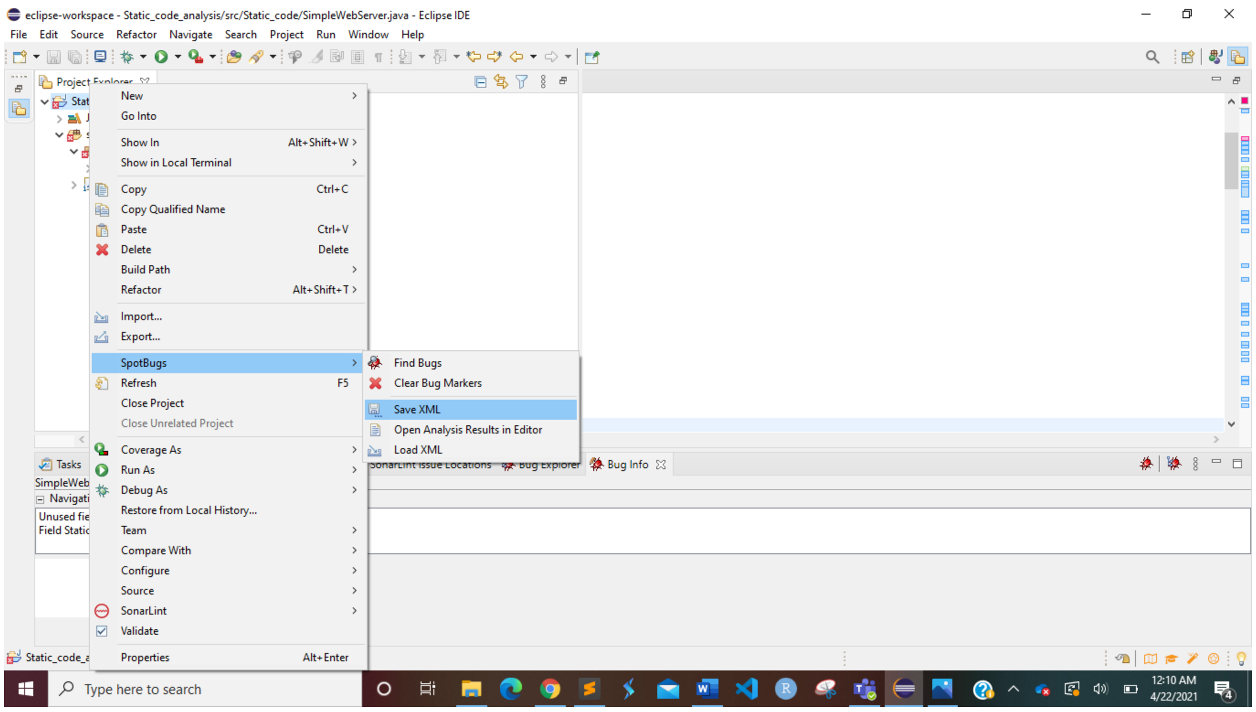
Resource	Date	Description
SimpleWebS	57 minutes ago	Explicitly import the specific classes needed.
SimpleWebS	57 minutes ago	Explicitly import the specific classes needed.
SimpleWebS	57 minutes ago	Missing curly brace.
SimpleWebS	57 minutes ago	Missing curly brace.
SimpleWebS	57 minutes ago	Add or update the header of this file.
SimpleWebS	57 minutes ago	Remove this throws clause.
SimpleWebS	1 hour ago	Move the array designator from the variable to the t
SimpleWebS	1 hour ago	Define and throw a dedicated exception instead of
SimpleWebS	1 hour ago	Define and throw a dedicated exception instead of
SimpleWebS	1 hour ago	Remove this assignment of "dServerSocket". [+1 loc
SimpleWebS	1 hour ago	Replace the synchronized class "StringBuffer" by an
SimpleWebS	1 hour ago	This file "SimpleWebServer.java" should be located
SimpleWebS	1 hour ago	Add an end condition to this loop.
SimpleWebS	1 hour ago	Use try-with-resources or close this "FileReader" in
module-info	30 minutes ago	Add a new line at the end of this file.

2 files of project Static_code_analysis (at 21/04/2021 23:49)

Remove this assignment of "dServerSocket".

I have copy pasted the issues found using SonarLint in an excel file. The file is attached in the submission folder.

As seen in the screenshot, we can save the SpotBugs report as an XML file. This XML file is attached to the submission of the task.



Results

The results in the raw format, excel sheet, xlm format have been provided in the submission folder. The best screenshots have been included in the report raw_result.pdf contains more images including the ones that have not been used.

Part 2

Buggy Java Code

The code provided is some code written in Java.

```
package com.aa.lookahead.one.common;


import java.util.ArrayList;

import java.util.List;


public class Sample {


        /**

         * @param args

         */

        public static void main(String[] args) {

                // TODO Auto-generated method stub


                ArrayList<Integer> list = null;

                for(int i = 0 ; i <12 ;i++)

                {

                        list.add(i);

                }


                int[] strs = new int[10];
                for (int i = 0 ; i < strs.length+1 ; i++)

                {

                        strs[i] = i+1;

                }


                System.out.println(list.size());

                System.out.println(strs[10]);


        }


}
```

Manual Analysis

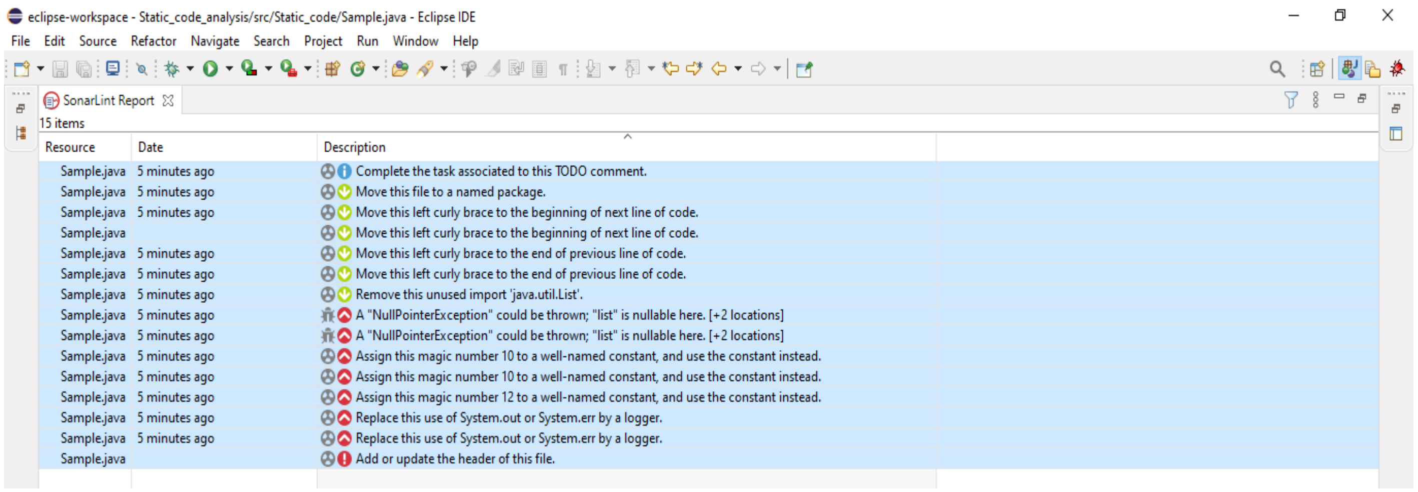
By just looking at the code, we can identify that

1) At Line 14 Array list has been initialized as ***null***. So, it will give null pointer exception error
2) At Line 21 ***i < strs.length+1*** will give array out of bound exception as strs is initialized as empty array.

SpotBugs report

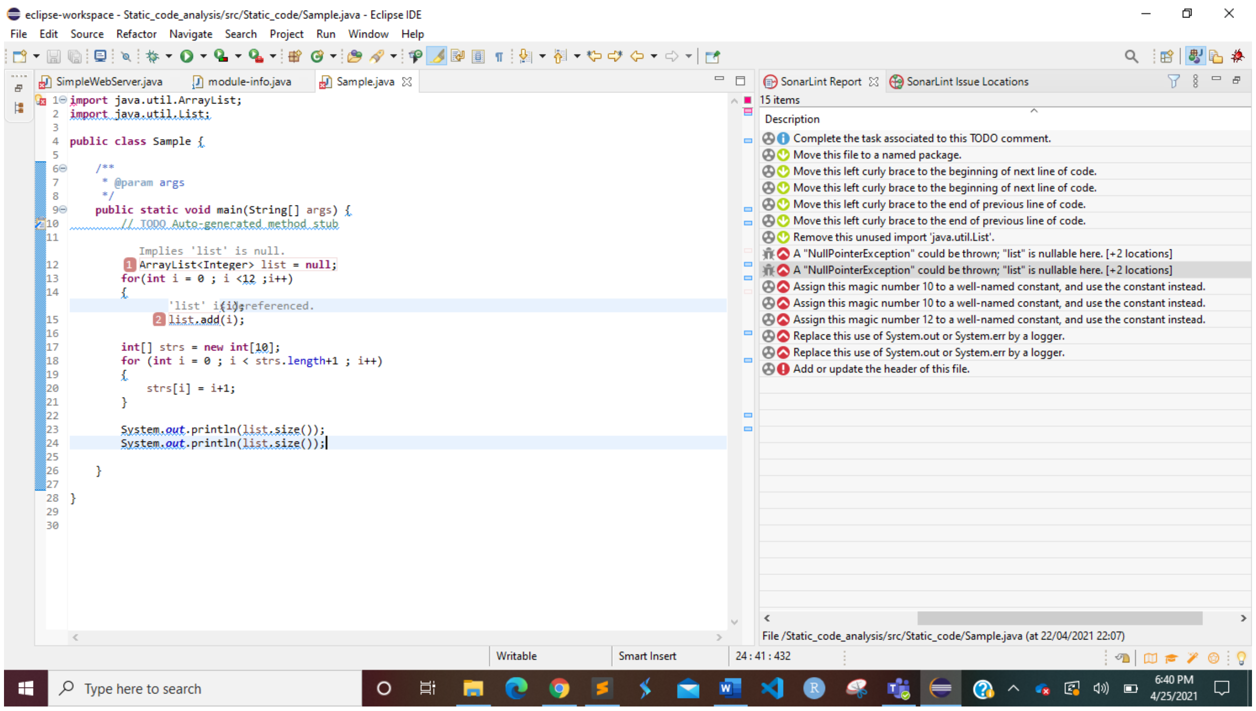
Could not find any bugs in spotbugs

SonarLint report


eclipse-workspace - Static_code_analysis/src/Static_code/Sample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
SonarLint Report

15 items

Resource	Date	Description
Sample.java	5 minutes ago	Complete the task associated to this TODO comment.
Sample.java	5 minutes ago	Move this file to a named package.
Sample.java	5 minutes ago	Move this left curly brace to the beginning of next line of code.
Sample.java		Move this left curly brace to the beginning of next line of code.
Sample.java	5 minutes ago	Move this left curly brace to the end of previous line of code.
Sample.java	5 minutes ago	Move this left curly brace to the end of previous line of code.
Sample.java	5 minutes ago	Remove this unused import 'java.util.List'.
Sample.java	5 minutes ago	A "NullPointerException" could be thrown; "list" is nullable here. [+2 locations]
Sample.java	5 minutes ago	A "NullPointerException" could be thrown; "list" is nullable here. [+2 locations]
Sample.java	5 minutes ago	Assign this magic number 10 to a well-named constant, and use the constant instead.
Sample.java	5 minutes ago	Assign this magic number 10 to a well-named constant, and use the constant instead.
Sample.java	5 minutes ago	Assign this magic number 12 to a well-named constant, and use the constant instead.
Sample.java	5 minutes ago	Replace this use of System.out or System.err by a logger.
Sample.java	5 minutes ago	Replace this use of System.out or System.err by a logger.
Sample.java		Add or update the header of this file.



eclipse-workspace - Static_code_analysis/src/Static_code/Sample.java - Eclipse IDE

```
 1 import java.util.ArrayList;
 2 import java.util.List;
 3
 4 public class Sample {
 5
 6     /**
 7      * @param args
 8      */
 9     public static void main(String[] args) {
10         // TODO Auto-generated method stub
11
           Implies 'list' is null.
12         1 ArrayList<Integer> list = null;
13         for(int i = 0 ; i <12 ;i++)
14         {
                 'list' is dereferenced.
15             2 list.add(i);
16
17         int[] strs = new int[10];
18         for (int i = 0 ; i < strs.length+1 ; i++)
19         {
20             strs[i] = i+1;
21         }
22
23         System.out.println(list.size());
24         System.out.println(list.size());
25
26     }
27
28 }
29
30
```

SonarLint Report

15 items

Description
Complete the task associated to this TODO comment.
Move this file to a named package.
Move this left curly brace to the beginning of next line of code.
Move this left curly brace to the beginning of next line of code.
Move this left curly brace to the end of previous line of code.
Move this left curly brace to the end of previous line of code.
Remove this unused import 'java.util.List'.
A "NullPointerException" could be thrown; "list" is nullable here. [+2 locations]
A "NullPointerException" could be thrown; "list" is nullable here. [+2 locations]
Assign this magic number 10 to a well-named constant, and use the constant instead.
Assign this magic number 10 to a well-named constant, and use the constant instead.
Assign this magic number 12 to a well-named constant, and use the constant instead.
Replace this use of System.out or System.err by a logger.
Replace this use of System.out or System.err by a logger.
Add or update the header of this file.

File /Static_code_analysis/src/Static_code/Sample.java (at 22/04/2021 22:07)

Writable | Smart Insert | 24 : 41 : 432

SonarLint showed the Null pointer exception at two locations at Line 12 and at Line 15. The initial manual analysis for identifying the null pointer exception was correct.

Fixes

In order to fix the above buggy java code

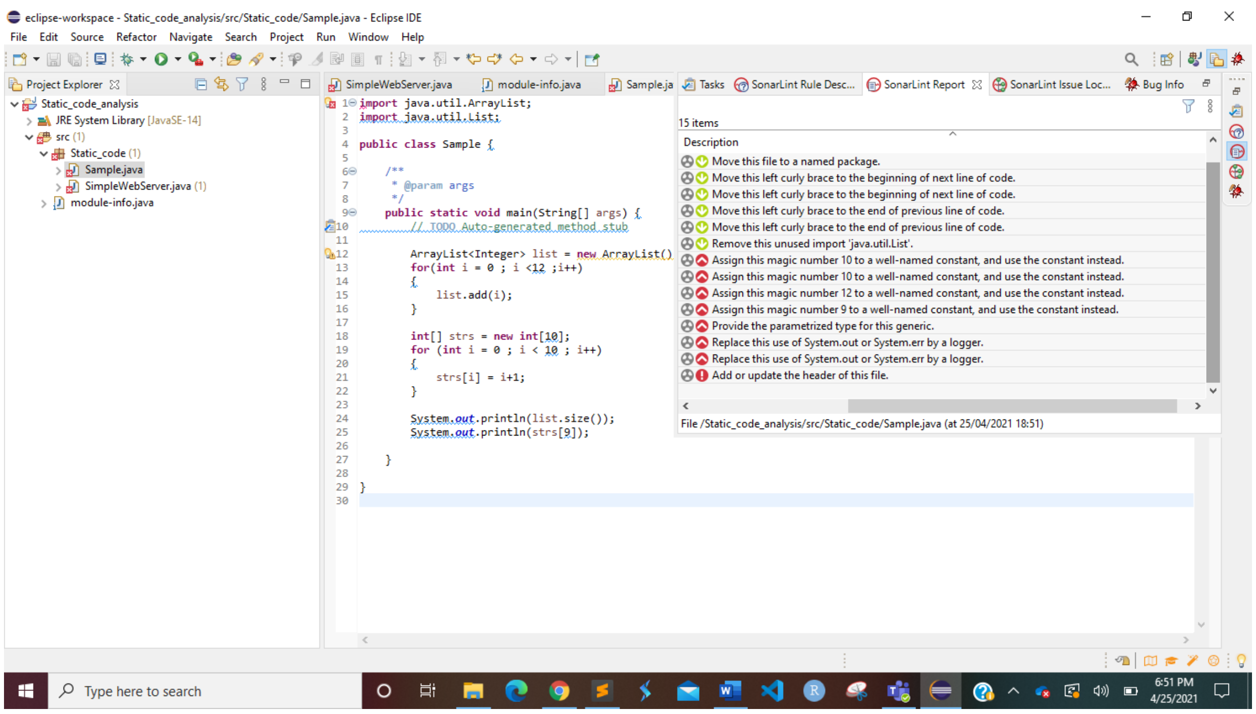
1) At line 12 instead of ArrayList<Integer> list =**null**
   Initialize it as ArrayList<Integer> list = **new** ArrayList();
   This would fix the null pointer exception error

2) At line 19 instead of **for** (**int** i = 0 ; i < 10 ; i++)
   Change to **for** (**int** i = 0 ; i < 10 ; i++)
   This would fix the the issue.

Fixed Code

```
import java.util.ArrayList;
import java.util.List;

public class Sample {

        /**
         * @param args
         */
        public static void main(String[] args) {
                // TODO Auto-generated method stub
                ArrayList<Integer> list = new ArrayList();
                for(int i = 0 ; i <12 ;i++)
                {
                        list.add(i);
                }

                int[] strs = new int[10];
                for (int i = 0 ; i < 10 ; i++)
                {
                        strs[i] = i+1;
                }

                System.out.println(list.size());
                System.out.println(strs[9]);

        }

}
```

SonarLint report after Fixes



eclipse-workspace - Static_code_analysis/src/Static_code/Sample.java - Eclipse IDE

```
import java.util.ArrayList;
import java.util.List;

public class Sample {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub

        ArrayList<Integer> list = new ArrayList()
        for(int i = 0 ; i <12 ;i++)
        {
            list.add(i);
        }

        int[] strs = new int[10];
        for (int i = 0 ; i < 10 ; i++)
        {
            strs[i] = i+1;
        }

        System.out.println(list.size());
        System.out.println(strs[9]);

    }

}
```

SonarLint Report

15 items

Description
- Move this file to a named package.
- Move this left curly brace to the beginning of next line of code.
- Move this left curly brace to the beginning of next line of code.
- Move this left curly brace to the end of previous line of code.
- Move this left curly brace to the end of previous line of code.
- Remove this unused import 'java.util.List'.
- Assign this magic number 10 to a well-named constant, and use the constant instead.
- Assign this magic number 10 to a well-named constant, and use the constant instead.
- Assign this magic number 12 to a well-named constant, and use the constant instead.
- Assign this magic number 9 to a well-named constant, and use the constant instead.
- Provide the parametrized type for this generic.
- Replace this use of System.out or System.err by a logger.
- Replace this use of System.out or System.err by a logger.
- Add or update the header of this file.

File /Static_code_analysis/src/Static_code/Sample.java (at 25/04/2021 18:51)

Null pointer exceptions have been fixed.