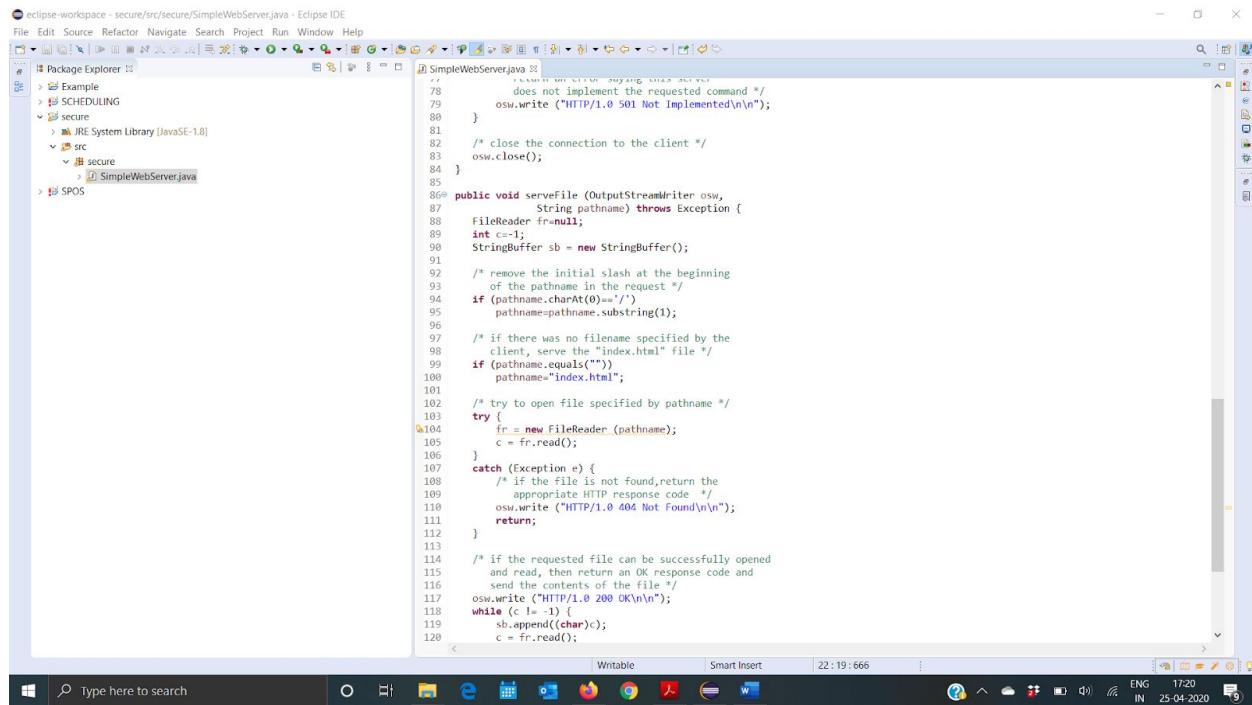


Name: Samiksha Dharmadhikari  
Id : 1001740496

## Part 1:

### Manual analysis:

We can observe that on line 104, fr is never closed. FileReader should be closed in order to remove the problem.



The screenshot shows the Eclipse IDE interface with the file "SimpleWebServer.java" open in the editor. The code implements a web server that reads files from the "src" directory. On line 104, a FileReader object "fr" is created and used to read the file specified by the pathname. However, there is no explicit code to close this reader object. The rest of the code handles the response and connection closure.

```
78     does not implement the requested command */
79     osw.write ("HTTP/1.0 501 Not Implemented\n\n");
80 }
81 /* close the connection to the client */
82 osw.close();
83 }
84 }
85
86 public void serveFile (OutputStreamWriter osw,
87                      String pathname) throws Exception {
88     FileReader fr=null;
89     int c=-1;
90     StringBuffer sb = new StringBuffer();
91
92     /* remove the initial slash at the beginning
93      of the pathname in the request */
94     if (pathname.charAt(0)=='/')
95         pathname=pathname.substring(1);
96
97     /* if there was no filename specified by the
98      client, serve the "index.html" file */
99     if (pathname.equals(""))
100         pathname="index.html";
101
102     /* try to open file specified by pathname */
103     try {
104         fr = new FileReader_(pathname);
105         c = fr.read();
106     }
107     catch (Exception e) {
108         /* if the file is not found,return the
109          appropriate HTTP response code */
110         osw.write ("HTTP/1.0 404 Not Found\n\n");
111         return;
112     }
113
114     /* if the requested file can be successfully opened
115      and read, then return an OK response code and
116      send the contents of the file */
117     osw.write ("HTTP/1.0 200 OK\n\n");
118     while (c != -1) {
119         sb.append((char)c);
120         c = fr.read();
121     }
122 }
```

### Tool Choices/Versions

1. Spot bugs - plugin 3.1.5
2. SonarLint version - 5.1

### Tool 1 : spot bugs

#### Steps :

1. We need to install this tool into the eclipse, we do this by searching it in Eclipse Marketplace.
2. Right click on the project and click properties.
3. We need to set the rank, confidence, and Bug Categories.
4. We can see the bugs identified in bugs explorer and its information in bug info.

Test 1 :

Rank: 15

Minimum Confidence to report: Medium

Reported Bug Categories(Visible): Bad Practice, Correctness, Performance, Dodgy Code, Multithreaded Correctness

Bugs: 2( 1 high confidence, 1 medium confidence)

Screenshot for test 1:

The screenshot shows the Eclipse IDE interface with a Java file named `SimpleWebServer.java` open. A context menu is displayed over the following line of code:

```
        /* remove the initial slash at the beginning
         * of the pathname in the request */
        if (0 == '/') {
            pathname = pathname.substring(1);
```

The context menu includes options like Example, SCHEDULING, New, Go Into, Open in New Window, Open type Hierarchy, Show In, Copy, Copy Qualified Name, Paste, Delete, Remove from Context, Build Path, Source, Refactor, Import..., Export..., SpotBugs, Refresh, Close Project, Close Unrelated Projects, Assign Working Sets..., Coverage As, Run As, Debug As, Restore from Local History..., Team, Compare With, Configure, Validate, Properties, and Alt+Enter.

The Java code in the editor is as follows:

```
StringBuffer sb = new StringBuffer();
sb.append("HTTP/1.0 200 OK\r\n");
sb.append("Content-Type: text/html\r\n");
sb.append("Content-Length: " + pathname.length() + "\r\n");
sb.append("\r\n");
sb.append(pathname);
System.out.println(sb.toString());
sb.setLength(0);
```

The status bar at the bottom shows the date as 25-04-2020 and the time as 17:27.

**Eclipse IDE - Java EE Perspective**

**Properties for secure**

**SpotBugs**

Enable project specific settings [Configure Workspace Settings](#)

Run automatically,  (also on full build), analysis effort [Default](#)

[Reporter Configuration](#) [Filter files](#) [Plugins and misc. Settings](#) [Detector configuration](#)

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

Reported (visible) bug categories:

- Bad practice
- Malicious code vulnerability
- Correctness
- Security
- Dodgy code
- Experimental
- Multithreaded correctness
- Internationalization

Mark bugs with ... rank as:

Scariest:	<a href="#">Warning</a>
Scary:	<a href="#">Warning</a>
Troubling:	<a href="#">Warning</a>
Of concern:	<a href="#">Warning</a>

[Restore Defaults](#) [Apply and Close](#) [Cancel](#)

**SimpleWebServer.java**

```

1 package secure;
2
3 import java.io.*;
4 import java.net.*;
5 import java.util.*;
6
7 public class SimpleWebServer {
8     public void serveFile (OutputStream osu,
9             String pathname) throws Exception {
10        FileReader fr=null;
11        int c=-1;
12        StringBuffer sb = new StringBuffer();
13
14        /* remove the initial slash at the beginning
15         * of the pathname in the request */
16        if (pathname.charAt(0)=="/")
17            pathname=pathname.substring(1);
18
19        /* if there was no filename specified by the
20         * client, serve the "index.html" file */
21        if (pathname.equals(""))
22            pathname="index.html";
23
24        /* try to open file specified by pathname */
25        try {
26            fr = new FileReader (pathname);
27            c = fr.read();
28        }
29        catch (Exception e) {
30            /* if the file is not found, return the
31             * appropriate HTTP response code */
32            osu.write ("HTTP/1.0 404 Not Found\n\n");
33            return;
34        }
35
36        /* if the requested file can be successfully opened
37         * and read, then return an OK response code and
38         * send the contents of the file */
39        osu.write ("HTTP/1.0 200 OK\n\n");
40        while (c != -1) {
41            sb.append((char)c);
42            c = fr.read();
43        }
44        osu.write (sb.toString());
45
46    }
47
48    /* This method is called when the program is run from
49     * the command line. */
50
51    public static void main (String argv[])
52        throws Exception {
53        /* Create a SimpleWebServer object, and run it */
54        SimpleWebServer sws = new SimpleWebServer();
55        sws.run();
56    }
57
58 }

```

**Secure**

Type here to search

File Edit Source Refactor Navigate Search Project Run Window Help

SimpleWebServer.java

```

1 package secure;
2
3 import java.io.*;
4 import java.net.*;
5 import java.util.*;
6
7 public class SimpleWebServer {
8     public void serveFile (OutputStream osu,
9             String pathname) throws Exception {
10        FileReader fr=null;
11        int c=-1;
12        StringBuffer sb = new StringBuffer();
13
14        /* remove the initial slash at the beginning
15         * of the pathname in the request */
16        if (pathname.charAt(0)=="/")
17            pathname=pathname.substring(1);
18
19        /* if there was no filename specified by the
20         * client, serve the "index.html" file */
21        if (pathname.equals(""))
22            pathname="index.html";
23
24        /* try to open file specified by pathname */
25        try {
26            fr = new FileReader (pathname);
27            c = fr.read();
28        }
29        catch (Exception e) {
30            /* if the file is not found, return the
31             * appropriate HTTP response code */
32            osu.write ("HTTP/1.0 404 Not Found\n\n");
33            return;
34        }
35
36        /* if the requested file can be successfully opened
37         * and read, then return an OK response code and
38         * send the contents of the file */
39        osu.write ("HTTP/1.0 200 OK\n\n");
40        while (c != -1) {
41            sb.append((char)c);
42            c = fr.read();
43        }
44        osu.write (sb.toString());
45
46    }
47
48    /* This method is called when the program is run from
49     * the command line. */
50
51    public static void main (String argv[])
52        throws Exception {
53        /* Create a SimpleWebServer object, and run it */
54        SimpleWebServer sws = new SimpleWebServer();
55        sws.run();
56    }
57
58 }

```

Type here to search

File Edit Source Refactor Navigate Search Project Run Window Help

**eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE**

File Edit Source Refactor Navigate Search Project Run Window Help

Bug Explorer (2) Of Concern (2)

- High confidence (1)
  - Write to static field from instance method (1)
    - Write to static field secure.SimpleWebServer.dServerSocket
- Normal confidence (1)

```

1 package secure;
2
3 import java.io.*;
4 import java.net.*;
5 import java.util.*;
6
7 public class SimpleWebServer {
8     ...
9     public void serveFile(OutputStreamWriter osw,
10         String pathname) throws Exception {
11         FileReader fr=null;
12         int c=-1;
13         StringBuffer sb = new StringBuffer();
14
15         /* remove the initial slash at the beginning
16         of the pathname in the request */
17         if (pathname.charAt(0)== '/')
18             pathname=pathname.substring(1);
19
20         /* if there was no filename specified by the
21         client, serve the "index.html" file */
22         if (pathname.equals(""))
23             pathname="index.html";
24
25         /* try to open file specified by pathname */
26         try {
27             fr = new FileReader (pathname);
28             c = fr.read();
29         }
30         catch (Exception e) {
31             /* if the file is not found,return the
32             appropriate HTTP response code */
33             osw.write ("HTTP/1.0 404 Not Found\r\n\r\n");
34             return;
35         }
36
37         /* if the requested file can be successfully opened
38         and read, then return an OK response code and
39         send the contents of the file */
40         osw.write ("HTTP/1.0 200 OK\r\n\r\n");
41         while (c != -1) {
42             sb.append((char)c);
43             c = fr.read();
44         }
45         osw.write (sb.toString());
46     }
47
48     /* This method is called when the program is run from
49     the command line. */
50 }
51
52 /* This method is called when the program is run from
53 the command line. */
54
55 public static void main(String[] args) {
56     SimpleWebServer server = new SimpleWebServer();
57     server.start();
58 }
59
60 
```

Write to static field secure.SimpleWebServer.dServerSocket from instance method new.secure.SimpleWebServer() [Of Concern(15); High confidence]

Type here to search

Bug Info (30)

SimpleWebServer.java: 30

Navigation

Write to static field secure.SimpleWebServer.dServerSocket from instance method new.secure.SimpleWebServer()

Bug: Write to static field secure.SimpleWebServer.dServerSocket from instance method new.secure.SimpleWebServer()

This instance method writes to a static field. This is tricky to get correct if multiple instances are being manipulated, and generally bad practice.

Rank: Of Concern (15), confidence: High

Pattern: ST\_WRITE\_TO\_STATIC\_FROM\_INSTANCE\_METHOD

Type: ST, Category: STYLE (Dodgy code)

---

XML output:

```

<BugInstance type="ST_WRITE_TO_STATIC_FROM_INSTANCE_METHOD" priority="1" rank="15">
    <Class classname="secure.SimpleWebServer">
        <SourceLine classname="secure.SimpleWebServer" sourcefile="SimpleWebServer.java" start="30" end="30" startB="30" endB="30" signature="()V" signatureB="()V" />
    </Class>
    <Method classname="secure.SimpleWebServer" name="<init>" signatures="()V" signaturesB="()V" start="29" end="31" startB="29" endB="31" />
    <Field classname="secure.SimpleWebServer" name="dServerSocket" signature="Lsecure/Socket;" signatureB="Lsocket/;" start="30" end="30" startB="30" endB="30" />
    <SourceLine classname="secure.SimpleWebServer" sourcefile="SimpleWebServer.java" start="30" end="30" startB="30" endB="30" />
</BugInstance>

```

---

**eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE**

File Edit Source Refactor Navigate Search Project Run Window Help

Bug Explorer (2) Of Concern (2)

- High confidence (1)
  - Write to static field from instance method (1)
    - Write to static field secure.SimpleWebServer.dServerSocket
- Normal confidence (1)
  - Dereference of the result of readLine() without nullcheck
    - Dereference of the result of readLine() without nullcheck

```

1 package secure;
2
3 import java.io.*;
4 import java.net.*;
5 import java.util.*;
6
7 public class SimpleWebServer {
8     ...
9     public void processRequest(Socket s) throws Exception {
10         /* used to read data from the client */
11         BufferedReader br =
12             new InputStreamReader (s.getInputStream());
13
14         /* used to write data to the client */
15         OutputStreamWriter osw =
16             new OutputStreamWriter (s.getOutputStream());
17
18         /* read the HTTP request from the client */
19         String request = br.readLine();
20
21         String command = null;
22         String pathname = null;
23
24         /* parse the HTTP request */
25         StringTokenizer st =
26             new StringTokenizer (request, " ");
27
28         command = st.nextToken();
29         pathname = st.nextToken();
30
31         if (command.equals("GET")) {
32             /* if the request is a GET,
33             try to respond with the file
34             the user is requesting */
35             serveFile (osw,pathname);
36         }
37         else {
38             /* if the request is a NOT a GET,
39             return an error saying this server
40             does not implement the requested command */
41             osw.write ("HTTP/1.0 501 Not Implemented\r\n\r\n");
42         }
43
44         /* close the connection to the client */
45         osw.close();
46     }
47
48     /* This method is called when the program is run from
49     the command line. */
50 }
51
52 /* This method is called when the program is run from
53 the command line. */
54
55 public static void main(String[] args) {
56     SimpleWebServer server = new SimpleWebServer();
57     server.start();
58 }
59
60 
```

Dereference of the result of readLine() without nullcheck in secure.Server.processRequest(Socket) [Of Concern(15); Normal confidence]

Type here to search

Bug Info (64)

SimpleWebServer.java: 64

Navigation

Dereference of the result of readLine() without nullcheck in secure.SimpleWebServer.processRequest(Socket)

Value loaded from request

Bug: Dereference of the result of readLine() without nullcheck in secure.SimpleWebServer.processRequest(Socket)

The result of invoking readLine() is dereferenced without checking to see if the result is null. If there are no more lines of text to read, readLine() will return null and dereferencing that will generate a null pointer exception.

Rank: Of Concern (15), confidence: Normal

Pattern: NP\_DEREFERENCE\_OF\_READLINE\_VALUE

Type: NP, Category: STYLE (Dodgy code)

---

XML output:

```

<BugInstance type="NP_DEREFERENCE_OF_READLINE_VALUE" priority="2" rank="15">
    <Class classname="secure.SimpleWebServer">
        <SourceLine classname="secure.SimpleWebServer" sourcefile="SimpleWebServer.java" start="64" end="64" startB="64" endB="64" signature="()V" signatureB="()V" />
    </Class>
    <Method classname="secure.SimpleWebServer" name="processRequest" signature="()V" start="64" end="64" startB="64" endB="64" />
    <SourceLine classname="secure.SimpleWebServer" sourcefile="SimpleWebServer.java" start="64" end="64" startB="64" endB="64" />
    <LocalVariable name="request" register="4" pc="49" role="LOCAL_VARIABLE_VALUE" start="64" end="64" startB="64" endB="64" />
    <SourceLine classname="secure.SimpleWebServer" sourcefile="SimpleWebServer.java" start="64" end="64" startB="64" endB="64" />
</BugInstance>

```

## Test 2:(Most aggressive mode)

Rank: 20

Minimum Confidence to report: Low

Reported Bug Categories(visible): Bad practice, Malicious code vulnerability, Correctness, Performance, Security, Dodgy Code, Experimental, Multithreaded correctness, Internationalization

Bugs: 8(4 high confidence, 3 normal confidence, 1 low confidence)

## ScreenShots for test 2:

The screenshot displays two instances of the Eclipse IDE interface, both showing the same Java code for a simple web server and its associated bug analysis results.

**Top Window (Bug Analysis Results):**

- Bug Explorer:** Shows 8 bugs categorized as secure, including 6 of concern (High confidence, Reliance on default encoding, Write to static field from instance method), 1 normal confidence, and 1 low confidence.
- Code Editor:** The `SimpleWebServer.java` file contains the following code:

```
1 package secure;
2
3 import java.io.*;
4
5 public class SimpleWebServer {
6     private static final int PORT = 8080;
7
8     private static ServerSocket dServerSocket;
9
10    public SimpleWebServer() throws Exception {
11        dServerSocket = new ServerSocket(PORT);
12    }
13
14    public void run() throws Exception {
15        while(true) {
16            /* wait for a connection from a client */
17            Socket s = dServerSocket.accept();
18
19            /* then process the client's request */
20            processRequest(s);
21        }
22    }
23
24    /* Reads the HTTP request from the client, and
25     * responds with the file the user requested or
26     * a HTTP error code. */
27    public void processRequest(Socket s) throws Exception {
28        /* used to read data from the client */
29        BufferedReader br =
30            new BufferedReader (
31                new InputStreamReader (s.getInputStream()));
32
33        /* used to write data to the client */
34        OutputStreamWriter osw =
35            new OutputStreamWriter (s.getOutputStream());
36
37        /* read the HTTP request from the client */
38        String request = br.readLine();
39
40        String command = null;
41        String pathname = null;
42
43        /* parse the HTTP request */
44        StringTokenizer st =
45            new StringTokenizer(request, " ");
46    }
47}
```
- Bug Info:** A detailed view of a bug entry for a static field write from an instance method.
  - SimpleWebServer.java:30**
  - Bug:** Write to static field secure.SimpleWebServer.dServerSocket from instance method new.secure.SimpleWebServer()
  - Description:** This instance method writes to a static field. This is tricky to get correct if multiple instances are being manipulated, and generally bad practice.
  - Rank:** Of Concern (15), **confidence:** High
  - Pattern:** ST\_WRITE\_TO\_STATIC\_FROM\_INSTANCE\_METHOD
  - Type:** ST, **Category:** STYLE (Dodgy code)
- XML output:** A snippet of XML representing the bug instance details.

**Bottom Window (Search Results):**

- Bug Explorer:** Shows 8 bugs categorized as secure, including 6 of concern (Reliance on default encoding, Found reliance on default encoding in secure.SimpleWebServer), 1 normal confidence, and 1 low confidence.
- Code Editor:** The `SimpleWebServer.java` file contains the same code as the top window.
- Bug Info:** A detailed view of a bug entry for found reliance on default encoding in the `processRequest` method.
  - SimpleWebServer.java:50**
  - Bug:** Found reliance on default encoding in secure.SimpleWebServer.processRequest(Socket): new java.io.InputStreamReader(InputStream)
  - Description:** Found a call to a method which will perform a byte to String (or String to byte) conversion, and will assume that the default platform encoding is suitable. This will cause the application behaviour to vary between platforms. Use an alternative API and specify a charset name or Charset object explicitly.
  - Rank:** Of Concern (19), **confidence:** High
  - Pattern:** DM\_DEFAULT\_ENCODING
  - Type:** Dm, **Category:** I18N (Internationalization)
- XML output:** A snippet of XML representing the bug instance details.

**eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE**

File Edit Source Refactor Navigate Search Project Run Window Help

Bug Explorer Package Explorer

```

22  /* Run the HTTP server on this TCP port. */
23  private static final int PORT = 8080;
24
25  /* The socket used to process incoming connections
26  from web clients */
27  private static ServerSocket dServerSocket;
28
29  public SimpleWebServer() throws Exception {
30     dServerSocket = new ServerSocket(PORT);
31 }
32
33  public void run() throws Exception {
34     while (true) {
35         /* wait for a connection from a client */
36         Socket s = dServerSocket.accept();
37
38         /* then process the client's request */
39         processRequest(s);
40     }
41 }
42
43  /* Reads the HTTP request from the client, and
44  responds with the file the user requested or
45  a HTTP error code. */
46  public void processRequest(Socket s) throws Exception {
47      /* used to read data from the client */
48      BufferedReader br =
49          new BufferedReader (
50              new InputStreamReader (s.getInputStream()));
51
52      /* used to write data to the client */
53      OutputStreamWriter osw =
54          new OutputStreamWriter (s.getOutputStream());
55
56      /* read the HTTP request from the client */
57      String request = br.readLine();
58
59      String command = null;
60      String pathname = null;
61
62      /* parse the HTTP request */
63      StringTokenizer st =
64          new StringTokenizer (request, " ");

```

Found reliance on default encoding in secure.SimpleWebServer.putStreamWriter(OutputStream) [Of Concern(19), High confidence] | Writable

Bug Info

SimpleWebServer.java: 54

- Navigation
- Called reliance on default encoding in secure.SimpleWebServer.processRequest(Socket): new java.io.OutputStreamWriter(OutputStream)

Bug: Found reliance on default encoding in secure.SimpleWebServer.processRequest(Socket): new java.io.OutputStreamWriter(OutputStream)

Found a call to a method which will perform a byte to String (or String to byte) conversion, and will assume that the default platform encoding is suitable. This will cause the application behaviour to vary between platforms. Use an alternative API and specify a charset name or Charset object explicitly.

Rank: Of Concern (19), confidence: High  
Pattern: DM\_DEFAULT\_ENCODING  
Type: Dim, Category: I18N (Internationalization)

XML output:

```

<BugInstance type="DM_DEFAULT_ENCODING" priority="1" rank="19" abbrev="Dim">
    <SourceLine classname="secure.SimpleWebServer">
        <SourceLine classname="secure.SimpleWebServer" sourcefile="SimpleWebServer.java" startB="46" endB="54" startE="46" endE="54" startB2="46" endB2="54" startE2="46" endE2="54" startB3="46" endB3="54" startE3="46" endE3="54" startB4="46" endB4="54" startE4="46" endE4="54" startB5="46" endB5="54" startE5="46" endE5="54" startB6="46" endB6="54" startE6="46" endE6="54" startB7="46" endB7="54" startE7="46" endE7="54" startB8="46" endB8="54" startE8="46" endE8="54" startB9="46" endB9="54" startE9="46" endE9="54" startB10="46" endB10="54" startE10="46" endE10="54" startB11="46" endB11="54" startE11="46" endE11="54" startB12="46" endB12="54" startE12="46" endE12="54" startB13="46" endB13="54" startE13="46" endE13="54" startB14="46" endB14="54" startE14="46" endE14="54" startB15="46" endB15="54" startE15="46" endE15="54" startB16="46" endB16="54" startE16="46" endE16="54" startB17="46" endB17="54" startE17="46" endE17="54" startB18="46" endB18="54" startE18="46" endE18="54" startB19="46" endB19="54" startE19="46" endE19="54" startB20="46" endB20="54" startE20="46" endE20="54" startB21="46" endB21="54" startE21="46" endE21="54" startB22="46" endB22="54" startE22="46" endE22="54" startB23="46" endB23="54" startE23="46" endE23="54" startB24="46" endB24="54" startE24="46" endE24="54" startB25="46" endB25="54" startE25="46" endE25="54" startB26="46" endB26="54" startE26="46" endE26="54" startB27="46" endB27="54" startE27="46" endE27="54" startB28="46" endB28="54" startE28="46" endE28="54" startB29="46" endB29="54" startE29="46" endE29="54" startB30="46" endB30="54" startE30="46" endE30="54" startB31="46" endB31="54" startE31="46" endE31="54" startB32="46" endB32="54" startE32="46" endE32="54" startB33="46" endB33="54" startE33="46" endE33="54" startB34="46" endB34="54" startE34="46" endE34="54" startB35="46" endB35="54" startE35="46" endE35="54" startB36="46" endB36="54" startE36="46" endE36="54" startB37="46" endB37="54" startE37="46" endE37="54" startB38="46" endB38="54" startE38="46" endE38="54" startB39="46" endB39="54" startE39="46" endE39="54" startB40="46" endB40="54" startE40="46" endE40="54" startB41="46" endB41="54" startE41="46" endE41="54" startB42="46" endB42="54" startE42="46" endE42="54" startB43="46" endB43="54" startE43="46" endE43="54" startB44="46" endB44="54" startE44="46" endE44="54" startB45="46" endB45="54" startE45="46" endE45="54" startB46="46" endB46="54" startE46="46" endE46="54" startB47="46" endB47="54" startE47="46" endE47="54" startB48="46" endB48="54" startE48="46" endE48="54" startB49="46" endB49="54" startE49="46" endE49="54" startB50="46" endB50="54" startE50="46" endE50="54" startB51="46" endB51="54" startE51="46" endE51="54" startB52="46" endB52="54" startE52="46" endE52="54" startB53="46" endB53="54" startE53="46" endE53="54" startB54="46" endB54="54" startE54="46" endE54="54" startB55="46" endB55="54" startE55="46" endE55="54" startB56="46" endB56="54" startE56="46" endE56="54" startB57="46" endB57="54" startE57="46" endE57="54" startB58="46" endB58="54" startE58="46" endE58="54" startB59="46" endB59="54" startE59="46" endE59="54" startB60="46" endB60="54" startE60="46" endE60="54" startB61="46" endB61="54" startE61="46" endE61="54" startB62="46" endB62="54" startE62="46" endE62="54" startB63="46" endB63="54" startE63="46" endE63="54" startB64="46" endB64="54" startE64="46" endE64="54" startB65="46" endB65="54" startE65="46" endE65="54" startB66="46" endB66="54" startE66="46" endE66="54" startB67="46" endB67="54" startE67="46" endE67="54" startB68="46" endB68="54" startE68="46" endE68="54" startB69="46" endB69="54" startE69="46" endE69="54" startB70="46" endB70="54" startE70="46" endE70="54" startB71="46" endB71="54" startE71="46" endE71="54" startB72="46" endB72="54" startE72="46" endE72="54" startB73="46" endB73="54" startE73="46" endE73="54" startB74="46" endB74="54" startE74="46" endE74="54" startB75="46" endB75="54" startE75="46" endE75="54" startB76="46" endB76="54" startE76="46" endE76="54" startB77="46" endB77="54" startE77="46" endE77="54" startB78="46" endB78="54" startE78="46" endE78="54" startB79="46" endB79="54" startE79="46" endE79="54" startB80="46" endB80="54" startE80="46" endE80="54" startB81="46" endB81="54" startE81="46" endE81="54" startB82="46" endB82="54" startE82="46" endE82="54" startB83="46" endB83="54" startE83="46" endE83="54" startB84="46" endB84="54" startE84="46" endE84="54" startB85="46" endB85="54" startE85="46" endE85="54" startB86="46" endB86="54" startE86="46" endE86="54" startB87="46" endB87="54" startE87="46" endE87="54" startB88="46" endB88="54" startE88="46" endE88="54" startB89="46" endB89="54" startE89="46" endE89="54" startB90="46" endB90="54" startE90="46" endE90="54" startB91="46" endB91="54" startE91="46" endE91="54" startB92="46" endB92="54" startE92="46" endE92="54" startB93="46" endB93="54" startE93="46" endE93="54" startB94="46" endB94="54" startE94="46" endE94="54" startB95="46" endB95="54" startE95="46" endE95="54" startB96="46" endB96="54" startE96="46" endE96="54" startB97="46" endB97="54" startE97="46" endE97="54" startB98="46" endB98="54" startE98="46" endE98="54" startB99="46" endB99="54" startE99="46" endE99="54" startB100="46" endB100="54" startE100="46" endE100="54" startB101="46" endB101="54" startE101="46" endE101="54" startB102="46" endB102="54" startE102="46" endE102="54" startB103="46" endB103="54" startE103="46" endE103="54" startB104="46" endB104="54" startE104="46" endE104="54" startB105="46" endB105="54" startE105="46" endE105="54" startB106="46" endB106="54" startE106="46" endE106="54" startB107="46" endB107="54" startE107="46" endE107="54" startB108="46" endB108="54" startE108="46" endE108="54" startB109="46" endB109="54" startE109="46" endE109="54" startB110="46" endB110="54" startE110="46" endE110="54" startB111="46" endB111="54" startE111="46" endE111="54" startB112="46" endB112="54" startE112="46" endE112="54" startB113="46" endB113="54" startE113="46" endE113="54" startB114="46" endB114="54" startE114="46" endE114="54" startB115="46" endB115="54" startE115="46" endE115="54" startB116="46" endB116="54" startE116="46" endE116="54" startB117="46" endB117="54" startE117="46" endE117="54" startB118="46" endB118="54" startE118="46" endE118="54" startB119="46" endB119="54" startE119="46" endE119="54" startB120="46" endB120="54" startE120="46" endE120="54" startB121="46" endB121="54" startE121="46" endE121="54" startB122="46" endB122="54" startE122="46" endE122="54" startB123="46" endB123="54" startE123="46" endE123="54" startB124="46" endB124="54" startE124="46" endE124="54" startB125="46" endB125="54" startE125="46" endE125="54" startB126="46" endB126="54" startE126="46" endE126="54" startB127="46" endB127="54" startE127="46" endE127="54" startB128="46" endB128="54" startE128="46" endE128="54" startB129="46" endB129="54" startE129="46" endE129="54" startB130="46" endB130="54" startE130="46" endE130="54" startB131="46" endB131="54" startE131="46" endE131="54" startB132="46" endB132="54" startE132="46" endE132="54" startB133="46" endB133="54" startE133="46" endE133="54" startB134="46" endB134="54" startE134="46" endE134="54" startB135="46" endB135="54" startE135="46" endE135="54" startB136="46" endB136="54" startE136="46" endE136="54" startB137="46" endB137="54" startE137="46" endE137="54" startB138="46" endB138="54" startE138="46" endE138="54" startB139="46" endB139="54" startE139="46" endE139="54" startB140="46" endB140="54" startE140="46" endE140="54" startB141="46" endB141="54" startE141="46" endE141="54" startB142="46" endB142="54" startE142="46" endE142="54" startB143="46" endB143="54" startE143="46" endE143="54" startB144="46" endB144="54" startE144="46" endE144="54" startB145="46" endB145="54" startE145="46" endE145="54" startB146="46" endB146="54" startE146="46" endE146="54" startB147="46" endB147="54" startE147="46" endE147="54" startB148="46" endB148="54" startE148="46" endE148="54" startB149="46" endB149="54" startE149="46" endE149="54" startB150="46" endB150="54" startE150="46" endE150="54" startB151="46" endB151="54" startE151="46" endE151="54" startB152="46" endB152="54" startE152="46" endE152="54" startB153="46" endB153="54" startE153="46" endE153="54" startB154="46" endB154="54" startE154="46" endE154="54" startB155="46" endB155="54" startE155="46" endE155="54" startB156="46" endB156="54" startE156="46" endE156="54" startB157="46" endB157="54" startE157="46" endE157="54" startB158="46" endB158="54" startE158="46" endE158="54" startB159="46" endB159="54" startE159="46" endE159="54" startB160="46" endB160="54" startE160="46" endE160="54" startB161="46" endB161="54" startE161="46" endE161="54" startB162="46" endB162="54" startE162="46" endE162="54" startB163="46" endB163="54" startE163="46" endE163="54" startB164="46" endB164="54" startE164="46" endE164="54" startB165="46" endB165="54" startE165="46" endE165="54" startB166="46" endB166="54" startE166="46" endE166="54" startB167="46" endB167="54" startE167="46" endE167="54" startB168="46" endB168="54" startE168="46" endE168="54" startB169="46" endB169="54" startE169="46" endE169="54" startB170="46" endB170="54" startE170="46" endE170="54" startB171="46" endB171="54" startE171="46" endE171="54" startB172="46" endB172="54" startE172="46" endE172="54" startB173="46" endB173="54" startE173="46" endE173="54" startB174="46" endB174="54" startE174="46" endE174="54" startB175="46" endB175="54" startE175="46" endE175="54" startB176="46" endB176="54" startE176="46" endE176="54" startB177="46" endB177="54" startE177="46" endE177="54" startB178="46" endB178="54" startE178="46" endE178="54" startB179="46" endB179="54" startE179="46" endE179="54" startB180="46" endB180="54" startE180="46" endE180="54" startB181="46" endB181="54" startE181="46" endE181="54" startB182="46" endB182="54" startE182="46" endE182="54" startB183="46" endB183="54" startE183="46" endE183="54" startB184="46" endB184="54" startE184="46" endE184="54" startB185="46" endB185="54" startE185="46" endE185="54" startB186="46" endB186="54" startE186="46" endE186="54" startB187="46" endB187="54" startE187="46" endE187="54" startB188="46" endB188="54" startE188="46" endE188="54" startB189="46" endB189="54" startE189="46" endE189="54" startB190="46" endB190="54" startE190="46" endE190="54" startB191="46" endB191="54" startE191="46" endE191="54" startB192="46" endB192="54" startE192="46" endE192="54" startB193="46" endB193="54" startE193="46" endE193="54" startB194="46" endB194="54" startE194="46" endE194="54" startB195="46" endB195="54" startE195="46" endE195="54" startB196="46" endB196="54" startE196="46" endE196="54" startB197="46" endB197="54" startE197="46" endE197="54" startB198="46" endB198="54" startE198="46" endE198="54" startB199="46" endB199="54" startE199="46" endE199="54" startB200="46" endB200="54" startE200="46" endE200="54" startB201="46" endB201="54" startE201="46" endE201="54" startB202="46" endB202="54" startE202="46" endE202="54" startB203="46" endB203="54" startE203="46" endE203="54" startB204="46" endB204="54" startE204="46" endE204="54" startB205="46" endB205="54" startE205="46" endE205="54" startB206="46" endB206="54" startE206="46" endE206="54" startB207="46" endB207="54" startE207="46" endE207="54" startB208="46" endB208="54" startE208="46" endE208="54" startB209="46" endB209="54" startE209="46" endE209="54" startB210="46" endB210="54" startE210="46" endE210="54" startB211="46" endB211="54" startE211="46" endE211="54" startB212="46" endB212="54" startE212="46" endE212="54" startB213="46" endB213="54" startE213="46" endE213="54" startB214="46" endB214="54" startE214="46" endE214="54" startB215="46" endB215="54" startE215="46" endE215="54" startB216="46" endB216="54" startE216="46" endE216="54" startB217="46" endB217="54" startE217="46" endE217="54" startB218="46" endB218="54" startE218="46" endE218="54" startB219="46" endB219="54" startE219="46" endE219="54" startB220="46" endB220="54" startE220="46" endE220="54" startB221="46" endB221="54" startE221="46" endE221="54" startB222="46" endB222="54" startE222="46" endE222="54" startB223="46" endB223="54" startE223="46" endE223="54" startB224="46" endB224="54" startE224="46" endE224="54" startB225="46" endB225="54" startE225="46" endE225="54" startB226="46" endB226="54" startE226="46" endE226="54" startB227="46" endB227="54" startE227="46" endE227="54" startB228="46" endB228="54" startE228="46" endE228="54" startB229="46" endB229="54" startE229="46" endE229="54" startB230="46" endB230="54" startE230="46" endE230="54" startB231="46" endB231="54" startE231="46" endE231="54" startB232="46" endB232="54" startE232="46" endE232="54" startB233="46" endB233="54" startE233="46" endE233="54" startB234="46" endB234="54" startE234="46" endE234="54" startB235="46" endB235="54" startE235="46" endE235="54" startB236="46" endB236="54" startE236="46" endE236="54" startB237="46" endB237="54" startE237="46" endE237="54" startB238="46" endB238="54" startE238="46" endE238="54" startB239="46" endB239="54" startE239="46" endE239="54" startB240="46" endB240="54" startE240="46" endE240="54" startB241="46" endB241="54" startE241="46" endE241="54" startB242="46" endB242="54" startE242="46" endE242="54" startB243="46" endB243="54" startE243="46" endE243="54" startB244="46" endB244="54" startE244="46" endE244="54" startB245="46" endB245="54" startE245="46" endE245="54" startB246="46" endB246="54" startE246="46" endE246="54" startB247="46" endB247="54" startE247="46" endE247="54" startB248="46" endB248="54" startE248="46" endE248="54" startB249="46" endB249="54" startE249="46" endE249="54" startB250="46" endB250="54" startE250="46" endE250="54" startB251="46" endB251="54" startE251="46" endE251="54" startB252="46" endB252="54" startE252="46" endE252="54" startB253="46" endB253="54" startE253="46" endE253="54" startB254="46" endB254="54" startE254="46" endE254="54" startB255="46" endB255="54" startE255="46" endE255="54" startB256="46" endB256="54" startE256="46" endE256="54" startB257="46" endB257="54" startE257="46" endE257="54" startB258="46" endB258="54" startE258="46" endE258="54" startB259="46" endB259="54" startE259="46" endE259="54" startB260="46" endB260="54" startE260="46" endE260="54" startB261="46" endB261="54" startE261="46" endE261="54" startB262="46" endB262="54" startE262="46" endE262="54" startB263="46" endB263="54" startE263="46" endE263="54" startB264="46" endB264="54" startE264="46" endE264="54" startB265="46" endB265="54" startE265="46" endE265="54" startB266="46" endB266="54" startE266="46" endE266="54" startB267="46" endB267="54" startE267="46" endE267="54" startB268="46" endB268="54" startE268="46" endE268="54" startB269="46" endB269="54" startE269="46" endE269="54" startB270="46" endB270="54" startE270="46" endE270="54" startB271="46" endB271="54" startE271="46" endE271="54" startB272="46" endB272="54" startE272="46" endE272="54" startB273="46" endB273="54" startE273="46" endE273="54" startB274="46" endB274="54" startE274="46" endE274="54" startB275="46" endB275="54" startE275="46" endE275="54" startB276="46" endB276="54" startE276="46" endE276="54" startB277="46" endB277="54" startE277="46" endE277="54" startB278="46" endB278="54" startE278="46" endE278="54" startB279="46" endB279="54" startE279="46" endE279="54" startB280="46" endB280="54" startE280="46" endE280="54" startB281="46" endB281="54" startE281="46" endE281="54" startB282="46" endB282="54" startE282="46" endE282="54" startB283="46" endB283="54" startE283="46" endE283="54" startB284="46" endB284="54" startE284="46" endE284="54" startB285="46" endB285="54" startE285="46" endE285="54" startB286="46" endB286="54" startE286="46" endE286="54" startB287="46" endB287="54" startE287="46" endE287="54" startB288="46" endB288="54" startE288="46" endE288="54" startB289="46" endB289="54" startE289="46" endE289="54" startB290="46" endB290="54" startE290="46" endE290="54" startB291="46" endB291="54" startE291="46" endE291="54" startB292="46" endB292="54" startE292="46" endE292="54" startB293="46" endB293="54" startE293="46" endE293="54" startB294="46" endB294="54" startE294="46" endE294="54" startB295="46" endB295="54" startE295="46" endE295="54" startB296="46" endB296="54" startE296="46" endE296="54" startB297="46" endB297="54" startE297="46" endE297="54" startB298="46" endB298="54" startE298="46" endE298="54" startB299="46" endB299="54" startE299="46" endE299="54" startB300="46" endB300="54" startE300="46" endE300="54" startB301="46" endB301="54" startE301="46" endE301="54" startB302="46" endB302="54" startE302="46" endE302="54" startB303="46" endB303="54" startE303="46" endE303="54" startB304="46" endB304="54" startE304="46" endE304="54" startB305="46" endB305="54" startE305="46" endE305="54" startB306="46" endB306="54" startE306="46" endE306="54" startB307="46" endB307="54" startE307="46" endE307="54" startB308="46" endB308="54" startE308="46" endE308="54" startB309="46" endB309="54" startE
```

eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE

```

1 package secure;
2
3 import java.io.*;
4 import java.net.*;
5
6 public class SimpleWebServer {
7     public void serveFile (OutputStreamWriter osw,
8             String pathname) throws Exception {
9         FileReader fr=null;
10        int c=-1;
11        StringBuffer sb = new StringBuffer();
12
13        /* remove the initial slash at the beginning
14         * of the pathname in the request */
15        if (pathname.charAt(0)=='/')
16            pathname=pathname.substring(1);
17
18        /* if there was no filename specified by the
19         * client, serve the "index.html" file */
20        if (pathname.equals(""))
21            pathname="index.html";
22
23        /* try to open file specified by pathname */
24        try {
25            fr = new FileReader (pathname);
26            c = fr.read();
27        }
28        catch (Exception e) {
29            /* if the file is not found,return the
30             * appropriate HTTP response code */
31            osw.write ("HTTP/1.0 404 Not Found\n\n");
32            return;
33        }
34
35        /* if the requested file can be successfully opened
36         * and read, then return an OK response code and
37         * send the contents of the file */
38        os.write ("HTTP/1.0 200 OK\n\n");
39        while (c != -1) {
40            sb.append((char)c);
41            c = fr.read();
42        }
43        osw.write (sb.toString());
44    }
45
46    /* This method is called when the program is run from
47     * the command line. */
48    public static void main (String argv[]) throws Exception {
49    }
50}

```

Bug Explorer | Package Explorer | SimpleWebServer.java | Bug Info | Smart Insert | 107: 57 [53] | ... | Type here to search | 17:43 | ENG | IN | 25-04-2020 | 10

**Bug:** Exception is caught when Exception is not thrown in secure.SimpleWebServer.serveFile(OutputStreamWriter, String)

This method uses a try-catch block that catches Exception objects, but Exception is not thrown within the try block, and RuntimeException is not explicitly caught. It is a common bug pattern to say `try { ... } catch (Exception e) { something }` as a shorthand for catching a number of types of exception each of whose catch blocks is identical, but this construct also accidentally catches RuntimeException as well, masking potential bugs.

A better approach is to either explicitly catch the specific exceptions that are thrown, or to explicitly catch RuntimeException exception, rethrow it, and then catch all non-Runtime Exceptions, as shown below:

```

try {
    ...
} catch (RuntimeException e) {
    throw e;
} catch (Exception e) {
    ... deal with all non-runtime exceptions ...
}

```

**Rank:** Of Concern (20), confidence: Low  
**Pattern:** REC\_CATCH\_EXCEPTION  
**Type:** REC, **Category:** STYLE (Dodgy code)

**XML output:**

```

<BugInstance type="REC_CATCH_EXCEPTION" priority="3" rank="20" abbrev="REC">
    <Class classname="secure.SimpleWebServer">
        <SourceLine classname="secure.SimpleWebServer" sourcefile="SimpleWebServer.java" start="86" end="107" startOffset="86" endOffset="107" type="Method"/>
        <SourceLine classname="secure.SimpleWebServer" name="serveFile" signature="(Lsecure/SecureWebServer;Ljava/io/OutputStreamWriter;Ljava/lang/String;)V" start="86" end="107" startOffset="86" endOffset="107" type="Method"/>
    </Class>
</BugInstance>

```

eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE

```

1 package secure;
2
3 import java.io.*;
4 import java.net.*;
5
6 public class SimpleWebServer {
7     public void serveFile (OutputStreamWriter osw,
8             String pathname) throws Exception {
9         FileReader fr=null;
10        int c=-1;
11        StringBuffer sb = new StringBuffer();
12
13        /* remove the initial slash at the beginning
14         * of the pathname in the request */
15        if (pathname.charAt(0)=='/')
16            pathname=pathname.substring(1);
17
18        /* if there was no filename specified by the
19         * client, serve the "index.html" file */
20        if (pathname.equals(""))
21            pathname="index.html";
22
23        /* try to open file specified by pathname */
24        try {
25            fr = new FileReader (pathname);
26            c = fr.read();
27        }
28        catch (Exception e) {
29            /* if the file is not found,return the
30             * appropriate HTTP response code */
31            osw.write ("HTTP/1.0 404 Not Found\n\n");
32            return;
33        }
34
35        /* if the requested file can be successfully opened
36         * and read, then return an OK response code and
37         * send the contents of the file */
38        os.write ("HTTP/1.0 200 OK\n\n");
39        while (c != -1) {
40            sb.append((char)c);
41            c = fr.read();
42        }
43        osw.write (sb.toString());
44    }
45
46    /* This method is called when the program is run from
47     * the command line. */
48    public static void main (String argv[]) throws Exception {
49    }
50}

```

Bug Explorer | Package Explorer | SimpleWebServer.java | Bug Info | Smart Insert | 107: 57 [53] | ... | Type here to search | 17:43 | ENG | IN | 25-04-2020 | 10

**Bug:** Found reliance on default encoding in secure.SimpleWebServer.serveFile(OutputStreamWriter, String): new java.io.FileReader(String)

Called method new java.io.FileReader(String)

Found a call to a method which will perform a byte to String (or String to byte) conversion, and will assume that the default platform encoding is suitable. This will cause the application behaviour to vary between platforms. Use an alternative API and specify a charset name or Charset object explicitly.

**Rank:** Of Concern (19), confidence: High  
**Pattern:** DM\_DEFAULT\_ENCODING  
**Type:** Dm, **Category:** I18N (Internationalization)

**XML output:**

```

<BugInstance type="DM_DEFAULT_ENCODING" priority="1" rank="19" abbrev="Dm">
    <Class classname="secure.SimpleWebServer">
        <SourceLine classname="secure.SimpleWebServer" sourcefile="SimpleWebServer.java" start="86" end="107" startOffset="86" endOffset="107" type="Method"/>
        <Method classname="secure.SimpleWebServer" name="serveFile" signature="(Lsecure/SecureWebServer;Ljava/io/OutputStreamWriter;Ljava/lang/String;)V" start="86" end="107" startOffset="86" endOffset="107" type="Method"/>
    </Class>
</BugInstance>

```

```

eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Bug Explorer Package Explorer
SimpleWebServer.java
85  public void serveFile (OutputStreamWriter osw,
86      String pathname) throws Exception {
87      FileReader fr=null;
88      int c=-1;
89      StringBuffer sb = new StringBuffer();
90      /* remove the initial slash at the beginning
91         of the pathname in the request */
92      if (pathname.charAt(0)=='/')
93          pathname=pathname.substring(1);
94
95      /* if there was no filename specified by the
96         client, serve the "index.html" file */
97      if (pathname.equals(""))
98          pathname="index.html";
99
100     /* try to open file specified by pathname */
101     try {
102         fr = new FileReader (pathname);
103         c = fr.read();
104     } catch (Exception e) {
105         /* if the file is not found,return the
106            appropriate HTTP response code */
107         osw.write ("HTTP/1.0 404 Not Found\n\n");
108         return;
109     }
110
111     /* if the requested file can be successfully opened
112        and read, then return an OK response code and
113        send the contents of the file */
114     osw.write ("HTTP/1.0 200 OK\n\n");
115     while (c != -1) {
116         sb.append((char)c);
117         c = fr.read();
118     }
119     osw.write (sb.toString());
120
121 }
122
123 /*
124 * This method is called when the program is run from
125 * the command line.
126 */
127 public static void main (String argv[]) throws Exception {
128
129 }

```

**Bug Info:**

- SimpleWebServer.java: 104**
- Navigation:** Found reliance on default encoding in secure.SimpleWebServer.serveFile(OutputStreamWriter, String): new java.io.FileReader
- Bug:** Found reliance on default encoding in secure.SimpleWebServer.serveFile(OutputStreamWriter, String): new java.io.FileReader(String)
- Description:** Found a call to a method which will perform a byte to String (or String to byte) conversion, and will assume that the default platform encoding is suitable. This will cause the application behaviour to vary between platforms. Use an alternative API and specify a charset name or Charset object explicitly.
- Rank:** Of Concern (19), confidence: High
- Pattern:** DM\_DEFAULT\_ENCODING
- Type:** Dim, Category: I18N (Internationalization)

**XML output:**

```

<BugInstance type="DM_DEFAULT_ENCODING" priority="1" rank="19" abbrev="Dim">
    <SourceLine classname="secure.SimpleWebServer">
        <Method classname="secure.SimpleWebServer" name="serveFile" signature="(Lsecure.SimpleWebServer;Ljava/io/OutputStreamWriter;Ljava/lang/String;)V" start="86" end="123" static="1"/>
        <SourceLine classname="secure.SimpleWebServer" start="104" end="104" static="1"/>
        <Method classname="java.io.FileReader" name="<init>" signature="()V" start="104" end="104" static="1"/>
    </SourceLine>
</BugInstance>

```

### Test 3:

Rank: 1

Minimum Confidence to report: High

Reported Bug Categories(visible): Bad Practice, Correctness, Performance, Dodgy Code, Multithreaded Correctness

Bugs: 0

Screenshot for test 3:

**eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE**

**File Edit Source Refactor Navigate Search Project Run Window Help**

**Bug Explorer Package Explorer**

**SimpleWebServer.java**

```

1 package secure;
2
3 import java.io.*;
4
5 public class SimpleWebServer {
6
7     /* Run the HTTP server on this TCP port. */
8     private static final int PORT = 8080;
9
10    /* The socket used to process incoming connections
11       from web clients */
12    private static ServerSocket dserverSocket;
13
14    public SimpleWebServer () throws Exception {
15        dserverSocket = new ServerSocket (PORT);
16    }
17
18    public void run() throws Exception {
19        while (true) {
20            /* wait for a connection from a client */
21            Socket s = dserverSocket.accept();
22
23            /* then process the client's request */
24            processRequest(s);
25        }
26    }
27
28    /* Reads the HTTP request from the client, and
29       responds with the file the user requested or
30       a HTTP error code */
31    public void processRequest(Socket s) throws Exception {
32        /* used to read data from the client */
33        BufferedReader br =
34            new BufferedReader (
35                new InputStreamReader (s.getInputStream()));
36
37        /* used to write data to the client */
38        OutputStreamWriter osw =
39            new OutputStreamWriter (s.getOutputStream());
40
41        /* write response back to the client */
42        osw.write ("HTTP/1.1 200 OK\r\n");
43        osw.write ("Content-Type: text/html\r\n");
44        osw.write ("\r\n");
45        osw.write ("Secure Web Server");
46        osw.flush();
47    }
48
49    BufferedReader br =
50        new BufferedReader (
51            new InputStreamReader (s.getInputStream()));
52
53    /* used to write data to the client */
54    OutputStreamWriter osw =
55        new OutputStreamWriter (s.getOutputStream());

```

**Properties**

**Alt+Enter**

**Type here to search**

**File Edit Source Refactor Navigate Search Project Run Window Help**

**Bug Explorer Package Explorer**

**Properties for secure**

**SpotBugs**

**Enable project specific settings**

**Run automatically, (also on full build), analysis effort Default**

**Reporter Configuration Filter files Plugins and misc. Settings Detector configuration**

**Minimum rank to report: (1 is most severe, 20 is least)**

**Minimum confidence to report: High**

**Reported (visible) bug categories**

- Bad practice
- Malicious code vulnerability
- Correctness
- Performance
- Security
- Dodgy code
- Experimental
- Multithreaded correctness
- Internationalization

**Mark bugs with ... rank as:**

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

**Restore Defaults**

**Apply and Close Cancel**

**Type here to search**

**File Edit Source Refactor Navigate Search Project Run Window Help**

**Bug Explorer Package Explorer**

**SimpleWebServer.java**

```

1 package secure;
2
3 import java.io.*;
4
5 public class SimpleWebServer {
6
7     /* Run the HTTP server on this TCP port. */
8     private static final int PORT = 8080;
9
10    /* The socket used to process incoming connections
11       from web clients */
12    private static ServerSocket dserverSocket;
13
14    public SimpleWebServer () throws Exception {
15        dserverSocket = new ServerSocket (PORT);
16    }
17
18    public void run() throws Exception {
19        while (true) {
20            /* wait for a connection from a client */
21            Socket s = dserverSocket.accept();
22
23            /* then process the client's request */
24            processRequest(s);
25        }
26    }
27
28    /* Reads the HTTP request from the client, and
29       responds with the file the user requested or
30       a HTTP error code */
31    public void processRequest(Socket s) throws Exception {
32        /* used to read data from the client */
33        BufferedReader br =
34            new BufferedReader (
35                new InputStreamReader (s.getInputStream()));
36
37        /* used to write data to the client */
38        OutputStreamWriter osw =
39            new OutputStreamWriter (s.getOutputStream());
40
41        /* write response back to the client */
42        osw.write ("HTTP/1.1 200 OK\r\n");
43        osw.write ("Content-Type: text/html\r\n");
44        osw.write ("\r\n");
45        osw.write ("Secure Web Server");
46        osw.flush();
47    }
48
49    BufferedReader br =
50        new BufferedReader (
51            new InputStreamReader (s.getInputStream()));
52
53    /* used to write data to the client */
54    OutputStreamWriter osw =
55        new OutputStreamWriter (s.getOutputStream());

```

**ENG IN 17:48 25-04-2020**

The screenshot shows the Eclipse IDE interface with the title "eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE". The left sidebar displays the Package Explorer with a tree structure showing packages like "Example", "SCHEDULING", "secure", and "src", which contains "SimpleWebServer.java". The main editor window shows the Java code for "SimpleWebServer.java". The code is as follows:

```
1 package secure;
2
3 import java.io.*;
4
5 public class SimpleWebServer {
6
7     /* Run the HTTP server on this TCP port. */
8     private static final int PORT = 8080;
9
10    /* The socket used to process incoming connections
11       from web clients */
12    private static ServerSocket dServerSocket;
13
14    public SimpleWebServer () throws Exception {
15        dServerSocket = new ServerSocket (PORT);
16    }
17
18    public void run() throws Exception {
19        while (true) {
20            /* wait for a connection from a client */
21            Socket s = dServerSocket.accept();
22
23            /* then process the client's request */
24            processRequest(s);
25        }
26    }
27
28    /* Reads the HTTP request from the client, and
29       responds with the file the user requested or
30       a HTTP error code */
31    public void processRequest(Socket s) throws Exception {
32        /* used to read data from the client */
33        BufferedReader br =
34            new BufferedReader (
35                new InputStreamReader (s.getInputStream()));
36
37        /* used to write data to the client */
38        OutputStreamWriter osw =
39            new OutputStreamWriter (s.getOutputStream());
40
41    }
42
43    /* used to read data from the client */
44    /* used to write data to the client */
45}
46
47
48
49
50
51
52
53
54
55
```

## Tool 2: Sonar Lint

### Steps:

1. We need to install this tool into the eclipse, we do this by searching it in Eclipse Marketplace.
2. We need to set the preferences for this tool to info, warnings and errors before each time we test.
3. After setting so run the sonar lint to get the sonar lint report.

### Test 1:

Severity of SonarLint markers: Info

Code Smell:

- 1) Number of major Code Smells: 4
- 2) Number of minor Code Smells: 1
- 3) Bugs(Blocker Issue):2

### Screenshots for Test 1:

eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE

```
1 package secure;
2
3 import java.io.*;
4
5 public class SimpleWebServer {
6
7     /* Run the HTTP server on this TCP port. */
8     private static final int PORT = 8080;
9
10    /* The socket used to process incoming connections
11       from web clients */
12    private static ServerSocket dserverSocket;
13
14    public SimpleWebServer () throws Exception {
15        dserverSocket = new ServerSocket (PORT);
16    }
17
18    public void run() throws Exception {
19        while (true) {
20            /* wait for a connection from a client */
21            Socket s = dserverSocket.accept();
22
23            /* then process the client's request */
24            processRequest(s);
25        }
26    }
27
28    /* Reads the HTTP request from the client, and
29       responds with the file the user requested or
30       a HTTP error code */
31    public void processRequest(Socket s) throws Exception {
32        /* used to read data from the client */
33        BufferedReader br =
34            new BufferedReader (
35                new InputStreamReader (s.getInputStream()));
36
37        /* used to write data to the client */
38        OutputStreamWriter osw =
39            new OutputStreamWriter (s.getOutputStream());
40
41    }
42
43    /* used to read the file requested by the client */
44    public void serveFile(OutputStreamWriter osw,
45        String pathname) throws IOException {
46        /* read the file */
47        String requestLine = br.readLine();
48        String command = requestLine.substring(0, 1);
49        String path = requestLine.substring(1);
50
51        /* parse the command */
52        StringTokenizer st = new StringTokenizer(path);
53        String commandName = st.nextToken();
54
55        /* read the file */
56        String content = null;
57        String fileContent = null;
58
59        String commandPathname = commandName + pathname;
60
61        /* if the command is GET */
62        if (command.equals("GET")) {
63            /* read the file */
64            fileContent = readFile(commandPathname);
65
66            /* command = pathname */
67            command = pathname;
68
69            if (command.equals("GET")) {
70                /* if the file exists */
71                if (fileContent != null) {
72                    /* serve the file */
73                    serveFile (osw,pathname);
74                } else {
75                    /* if the request is a NOT a GET,
76                     * return an error saying this server
77                     * does not implement the requested command */
78                    osw.write ("HTTP/1.0 501 Not Implemented\n\n");
79                }
80            }
81
82            /* close the connection to the client */
83            osw.close();
84        }
85
86    }
87
88    public void serveFile (OutputStreamWriter osw,
89        String pathname) throws IOException {
90        /* read the file */
91        String fileContent = readFile(pathname);
92
93        /* write the file content to the client */
94        osw.write(fileContent);
95
96    }
97
98    /* read the file */
99    private String readFile(String pathname) throws IOException {
100        File file = new File(pathname);
101        FileInputStream fis = new FileInputStream(file);
102        BufferedReader fr =
103            new BufferedReader (
104                new InputStreamReader (fis));
105
106        String fileContent = fr.readText();
107
108        fr.close();
109        fis.close();
110
111        return fileContent;
112    }
113}
```

eclipse-workspace - secure/src/secure/SimpleWebServer.java - Eclipse IDE

```
1 package secure;
2
3 import java.io.*;
4
5 public class SimpleWebServer {
6
7     /* Run the HTTP server on this TCP port. */
8     private static final int PORT = 8080;
9
10    /* The socket used to process incoming connections
11       from web clients */
12    private static ServerSocket dserverSocket;
13
14    public SimpleWebServer () throws Exception {
15        dserverSocket = new ServerSocket (PORT);
16    }
17
18    public void run() throws Exception {
19        while (true) {
20            /* wait for a connection from a client */
21            Socket s = dserverSocket.accept();
22
23            /* then process the client's request */
24            processRequest(s);
25        }
26    }
27
28    /* Reads the HTTP request from the client, and
29       responds with the file the user requested or
30       a HTTP error code */
31    public void processRequest(Socket s) throws Exception {
32        /* used to read data from the client */
33        BufferedReader br =
34            new BufferedReader (
35                new InputStreamReader (s.getInputStream()));
36
37        /* used to write data to the client */
38        OutputStreamWriter osw =
39            new OutputStreamWriter (s.getOutputStream());
40
41    }
42
43    /* used to read the file requested by the client */
44    public void serveFile(OutputStreamWriter osw,
45        String pathname) throws IOException {
46        /* read the file */
47        String requestLine = br.readLine();
48        String command = requestLine.substring(0, 1);
49        String path = requestLine.substring(1);
50
51        /* parse the command */
52        StringTokenizer st = new StringTokenizer(path);
53        String commandName = st.nextToken();
54
55        /* read the file */
56        String content = null;
57        String fileContent = null;
58
59        String commandPathname = commandName + pathname;
60
61        /* if the command is GET */
62        if (command.equals("GET")) {
63            /* read the file */
64            fileContent = readFile(commandPathname);
65
66            /* command = pathname */
67            command = pathname;
68
69            if (command.equals("GET")) {
70                /* if the file exists */
71                if (fileContent != null) {
72                    /* serve the file */
73                    serveFile (osw,pathname);
74                } else {
75                    /* if the request is a NOT a GET,
76                     * return an error saying this server
77                     * does not implement the requested command */
78                    osw.write ("HTTP/1.0 501 Not Implemented\n\n");
79                }
80            }
81
82            /* close the connection to the client */
83            osw.close();
84        }
85
86    }
87
88    public void serveFile (OutputStreamWriter osw,
89        String pathname) throws IOException {
90        /* read the file */
91        String fileContent = readFile(pathname);
92
93        /* write the file content to the client */
94        osw.write(fileContent);
95
96    }
97
98    /* read the file */
99    private String readFile(String pathname) throws IOException {
100        File file = new File(pathname);
101        FileInputStream fis = new FileInputStream(file);
102        BufferedReader fr =
103            new BufferedReader (
104                new InputStreamReader (fis));
105
106        String fileContent = fr.readText();
107
108        fr.close();
109        fis.close();
110
111        return fileContent;
112    }
113}
```

SonarLint dialog box open over the code, showing SonarLint settings.

Severity of SonarLint markers: Info

Test file regular expressions:

Outline view on the right side of the IDE.

**Screenshot 1 (Top): Eclipse IDE - SonarLint Context Menu**

The screenshot shows the Eclipse IDE interface with the following details:

- File Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Left Sidebar:** Package Explorer (secure), SCHEDULING, secure (src, New, Go Into, Open in New Window, Open Type Hierarchy, Show In, Copy, Paste, Cut, Delete, Remove from Context, Build Path, Source, Refactor, Import..., Export..., SpotBugs, Refresh, Close Project, Close Unrelated Projects, Assign Working Sets..., Coverage As, Run As, Debug As, Restore from Local History..., Team, Compare With, Configure, SonarLint, Validate, Properties).
- Middle Area:** SimpleWebServer.java code editor. The code handles HTTP requests and serves files. A SonarLint context menu is open over the code, with 'Analyze changed files' and 'Analyze' selected.
- Right Area:** Outline view showing the project structure.
- Bottom:** Status bar showing ENG IN 18:54 25-04-2020.

**Screenshot 2 (Bottom): Eclipse IDE - SonarLint Report**

The screenshot shows the Eclipse IDE interface with the following details:

- File Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Left Sidebar:** Package Explorer (secure, JRE System Library [JavaSE-1.8], src, secure, SimpleWebServer.java, SPOS).
- Middle Area:** SimpleWebServer.java code editor. The code handles HTTP requests and serves files.
- Bottom Area:** SonarLint Report view showing 7 items with resource, date, and description columns.
- Bottom:** Status bar showing ENG IN 18:54 25-04-2020.

Resource	Date	Description
SimpleWebSe		Move the array designator from the variable to the type.
SimpleWebSe		Define and throw a dedicated exception instead of using a generic one.
SimpleWebSe		Define and throw a dedicated exception instead of using a generic one.
SimpleWebSe		Remove this assignment of "dServerSocket".
SimpleWebSe		Replace the synchronized class "StringBuffer" by an unsynchronized one such as "StringBuilder".
SimpleWebSe		Add an end condition to this loop.
SimpleWebSe		Use try-with-resources or close this "fileReader" in a "finally" clause.

## Screen shot for test 2:

### Test 2:

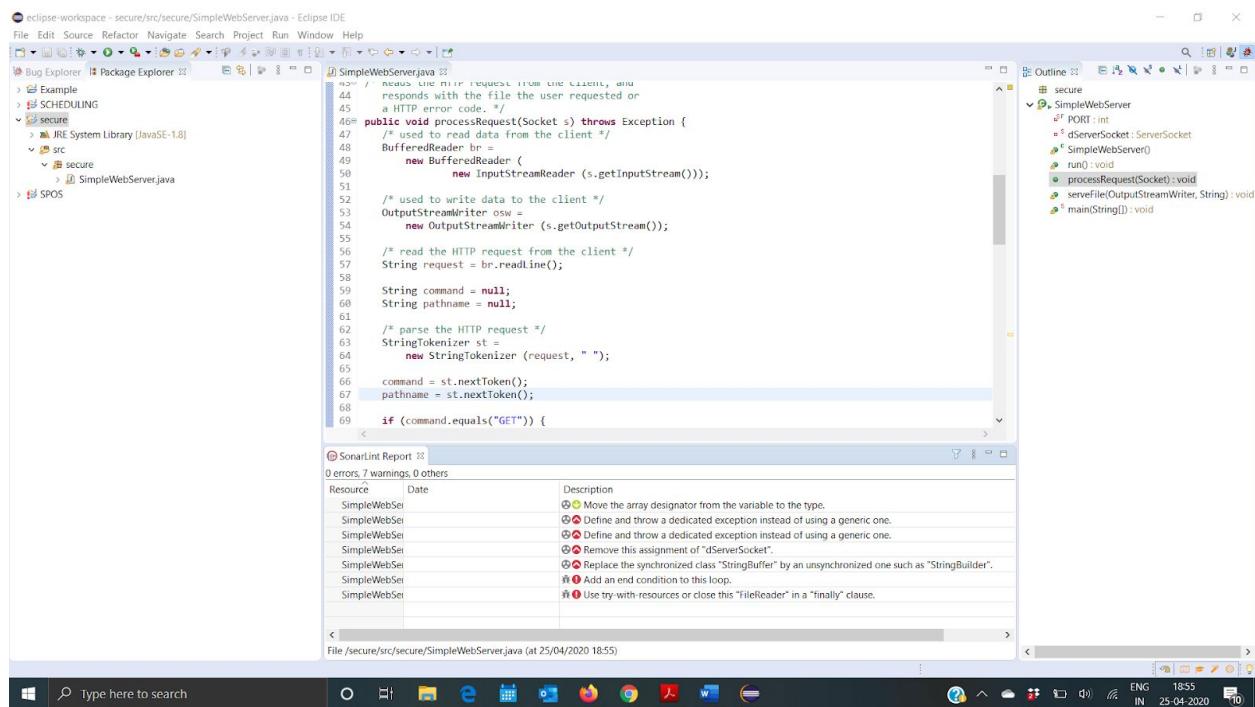
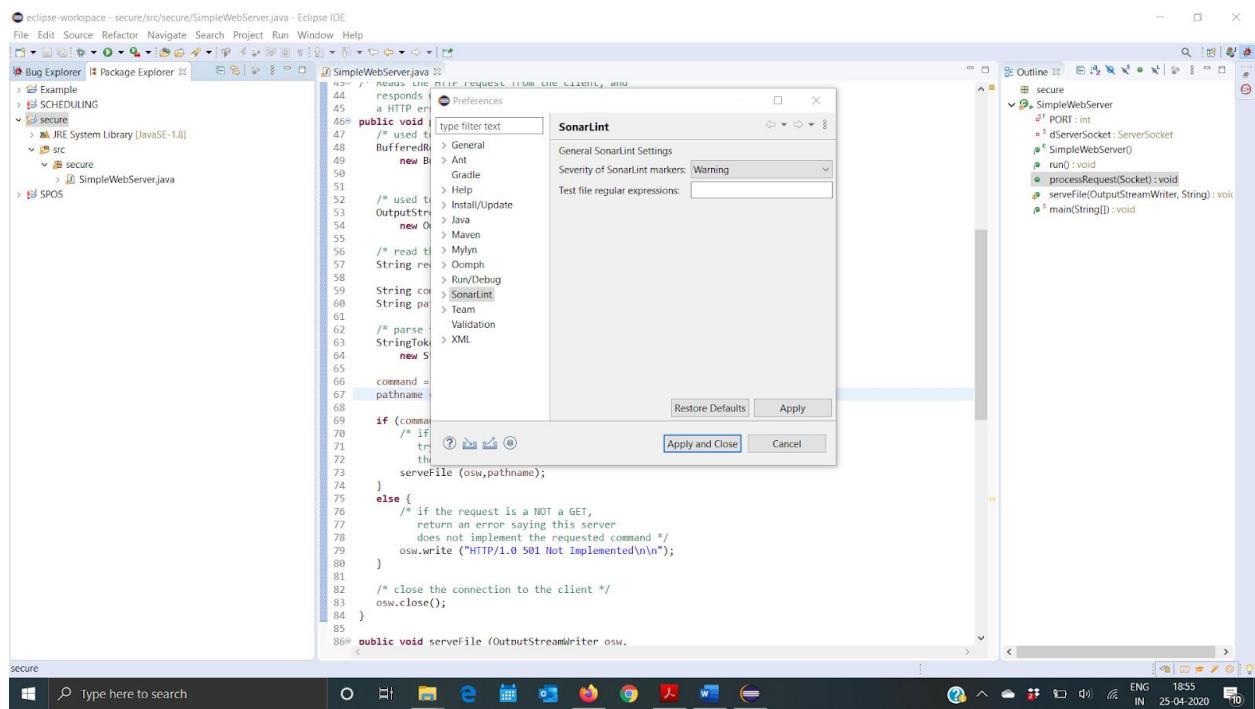
Severity of SonarLint markers: Warnings

Code Smell:

1) Number of major Code Smells: 4

2) Number of minor Code Smells: 1

### 3) Bugs(Blocker Issue):2



Screenshot for test 3:

Test 3:

Severity of Sonarlint markers: Errors

Code Smell:

- 1)Number of major Code Smells: 4
- 2) Number of minor Code Smells: 1
- 3) Bugs(Blocker Issue):2

Screenshot of Eclipse IDE showing SonarLint analysis results for SimpleWebServer.java.

The code in SimpleWebServer.java handles an incoming HTTP request from a client. It reads the file requested by the client and responds with a file or an error code. The code uses BufferedReader and BufferedWriter to read and write data to the client. It also uses StringTokenizer to parse the HTTP request.

**SonarLint Settings:**

- Severity of SonarLint markers: Error
- Test file regular expressions: None

**SonarLint Report:**

0 errors, 7 warnings, 0 others

Resource	Date	Description
SimpleWebSei		Move the array designator from the variable to the type.
SimpleWebSei		Define and throw a dedicated exception instead of using a generic one.
SimpleWebSei		Define and throw a dedicated exception instead of using a generic one.
SimpleWebSei		Remove this assignment of "dServerSocket".
SimpleWebSei		Replace the synchronized class "StringBuffer" by an unsynchronized one such as "StringBuilder".
SimpleWebSei		Add an end condition to this loop.
SimpleWebSei		Use try-with-resources or close this "FileReader" in a "finally" clause.

Screenshot of Eclipse IDE showing SonarLint analysis results for SimpleWebServer.java.

The code in SimpleWebServer.java handles an incoming HTTP request from a client. It reads the file requested by the client and responds with a file or an error code. The code uses BufferedReader and BufferedWriter to read and write data to the client. It also uses StringTokenizer to parse the HTTP request.

**SonarLint Report:**

7 errors, 0 warnings, 0 others

Resource	Date	Description
SimpleWebSei		Move the array designator from the variable to the type.
SimpleWebSei		Define and throw a dedicated exception instead of using a generic one.
SimpleWebSei		Define and throw a dedicated exception instead of using a generic one.
SimpleWebSei		Remove this assignment of "dServerSocket".
SimpleWebSei		Replace the synchronized class "StringBuffer" by an unsynchronized one such as "StringBuilder".
SimpleWebSei		Add an end condition to this loop.
SimpleWebSei		Use try-with-resources or close this "FileReader" in a "finally" clause.

## Comparison/Contrast Tools

Does the tool analyze source or binary as input?

SonarLint - Source code

SpotBugs - Binary

Which category of tools is it?

SonarLint - Bug finding

SpotBugs - Bug finding

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

We can see in the below screenshots the finding on line 64 by SpotBugs is not reported by SonarLint which is StringTokenizer.

The screenshot shows the Eclipse IDE interface with the Java code for `SimpleWebServer.java`. The code handles HTTP requests and responses. On line 64, there is a call to `StringTokenizer st = new StringTokenizer(request, " ")`. This line is highlighted with a red box, indicating it is flagged as a bug by SpotBugs. The 'Bug Info' view on the right provides detailed information about this finding:

- Rank:** Of Concern (15)
- confidence:** Normal
- Pattern:** NP\_DEREference\_OF\_READLINE\_VALUE
- Type:** NP, Category: STYLE (Dodgy code)

The message in the 'Bug Info' view states: "The result of invoking readLine() is dereferenced without checking to see if the result is null. If there are no more lines of text to read, readLine() will return null and dereferencing that will generate a null pointer exception."

```
43* /* Reads the HTTP request from the client, and
44    responds with the file the user requested or
45    a HTTP error code. */
46* public void processRequest(Socket s) throws Exception {
47    /* used to read data from the client */
48    BufferedReader br =
49        new BufferedReader (
50            new InputStreamReader (s.getInputStream()));
51
52    /* used to write data to the client */
53    OutputStream osw =
54        new OutputStreamWriter (s.getOutputStream());
55
56    /* read the HTTP request from the client */
57    String request = br.readLine();
58
59    String command = null;
60    String pathname = null;
61
62    /* parse the HTTP request */
63    StringTokenizer st =
64        new StringTokenizer (request, " ");
65
66    command = st.nextToken();
67    pathname = st.nextToken();
68
69    if (command.equals("GET")) {
70        /* if the request is a GET,
71         * try to respond with the file
72         * the user is requesting */
73        serveFile (osw,pathname);
74    } else {
75        /* if the request is a NOT a GET,
76         * return an error saying this server
77         * does not implement the requested command */
78        osw.write ("HTTP/1.0 501 Not Implemented\n");
79    }
80
81    /* close the connection to the client */
82    osw.close();
83
84}
```

The screenshot shows the Eclipse IDE interface with the following details:

- File Explorer:** Shows the project structure with a file named `SimpleWebServer.java`.
- Code Editor:** Displays the Java code for `SimpleWebServer`. Line 30 is highlighted.
- Bug Explorer:** Shows a warning for line 30: "Write to static field secure.SimpleWebServer.dServerSocket from instance method new secure.SimpleWebServer()".
- SonarLint Report:** Shows 7 warnings, including:
  - Line 29: Move the array designator from the variable to the type.
  - Line 30: Define and throw a dedicated exception instead of using a generic one.
  - Line 31: Define and throw a dedicated exception instead of using a generic one.
  - Line 32: Remove this assignment of "dServerSocket".
  - Line 33: Replace the synchronized class "StringBuffer" by an unsynchronized one such as "StringBuilder".
  - Line 34: Add an end condition to this loop.
  - Line 35: Use try-with-resources or close this "fileReader" in a "finally" clause.
- Outline View:** Shows the class structure with methods `dServerSocket`, `run()`, `processRequest(Socket)`, `servefile(OutputStreamWriter, String)`, and `main(String[])`.
- Taskbar:** Shows the Windows taskbar with various pinned icons.

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

We can see from the below screenshot, both Spotbugs and SonarLint show an issue on line 30 which is `dServerSocket`.

The screenshot shows the Eclipse IDE interface with the following details:

- File Explorer:** Shows the project structure with a file named `SimpleWebServer.java`.
- Bug Explorer:** Shows two findings for line 30:
  - Of Concern (8):** Reliance on default encoding (4), Write to static field from instance method (1), Write to static field `secure.SimpleWebServer.dServerSocket` (1).
  - Normal confidence (3):**
  - Low confidence (1):**
- Bug Info:** Details for the finding at line 30:
  - Bug:** Write to static field `secure.SimpleWebServer.dServerSocket` from instance method `new secure.SimpleWebServer()`
  - Description:** This instance method writes to a static field. This is tricky to get correct if multiple instances are being manipulated, and generally bad practice.
  - Rank:** Of Concern (15), **confidence:** High
  - Pattern:** ST\_WRITE\_TO\_STATIC\_FROM\_INSTANCE\_METHOD
  - Type:** ST, **Category:** STYLE (Dodgy code)
- XML output:** Shows the XML representation of the bug instance.
- Taskbar:** Shows the Windows taskbar with various pinned icons.

