Broken Links Documentation

Badoi Mircea Aurelian

April 29, 2020

1 Problem Statement

Design an application which takes an URL and displays all the links found on the page. Application could also shows a sorted list alphabetically of the links, a sorted list by time response and as well as filters the link by status DOWN(broken).

2 Problem Solving

In order to find the broken links, we have followed the next steps:
-collect the links found on the given page
-send request to the server for all links found and read the response code
-based on response from server, the links are categorized by DOWN or UP

For sorting we use sort method of ArrayList by writing a comparator which suits our needs.

Filtering it is just a traverse of the list of links. We keep just the links with status DOWN.

3 Application outline

3.1 The high level architectural overview of the application

Application is divided in 3 main folders.
-first one (src) contains the interfaces and classes for solving the requirements
-the second one (test) has the JUnit Tests for:

-FilterListOfLinksByDown class

-LinkTypeBrokenCheck class

-SortListOfLinksByName class

-SortListOfLinksByTimeResponse class.
-the third one (GUI) where we build the interface with the user using Java Swing. It also contains MainFrameController class which controls the app.

3.2 The specification of the input data format

Input data is given by the user as a valid URL. In case of an invalid link, the application displays "ERROR...TRY AGAIN".

3.3 The specification of the output data format

The output represent a list of outgoing links, the time response and the status of them.

3.4 The list of all the files in the application and their description

Classes and Interfaces:
-src folder:

-WebLinkType class it's a type of data. It is composed of an enum status (statusLink), a string (url) and an int(timeResponseServer)

-FilterListOfLinksByAction interface has one function filterBy

-FilterListOfLinksByDown implements the upper interface. Function filterBy takes a list of WebLinkType and returns an ArrayList with the DOWN links

-FindLinkAction interface has one function findAllLinks

-FindLinks implements the interface FindLinkAction. Function findAllLinks takes a String and returns an ArrayList of URLs found on a page(for this we use JSoup).

-LinkTypeAction interface has two functions, isLinkBroken and responseTimeServer

-LinkTypeBrokenCheck implements LinkTypeAction. Function isLinkBroken takes an URL type element and a WebLinkType, try to connect to the server, acknowledge the response and set the status of WebLinkType element.

-SortOnListOfLinksAction interface has one function sortBy.

-SortListOfLinksByName implements SortOnListOfLinksAction. Function sortBy takes a list and sort alphabetically it.

-SortListOfLinksByTimeResponse implements SortOnListOfLinksAction. Function sortBy takes a list and sort by time response it.
-test folder:

-FilterByDownStatusTest class

-LinkTypeBrokenCheckTest class

-SortListByNamesTest class

-SortListByTimeResponseTest class.
-GUI folder:

-Main class

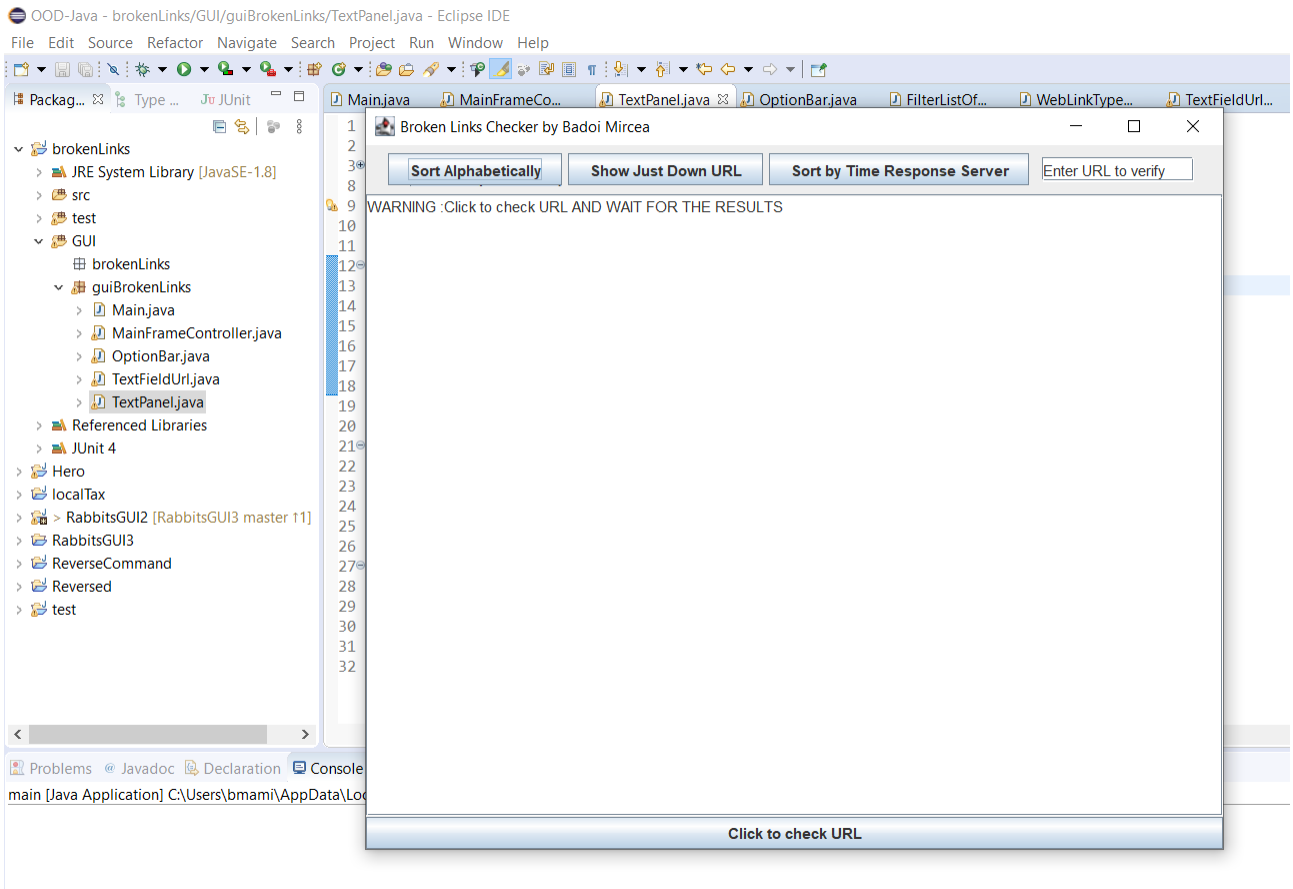
-MainFrameController class set the preconditions of the program and build the necessary components of app

-OptionBar class makes the tool bar with three buttons, two for sorting and one to filter. It contains also a text field where you introduced the URL

-TextFieldUrl class contains the text field of the tool bar

-TextPanel class represents the display area of the app

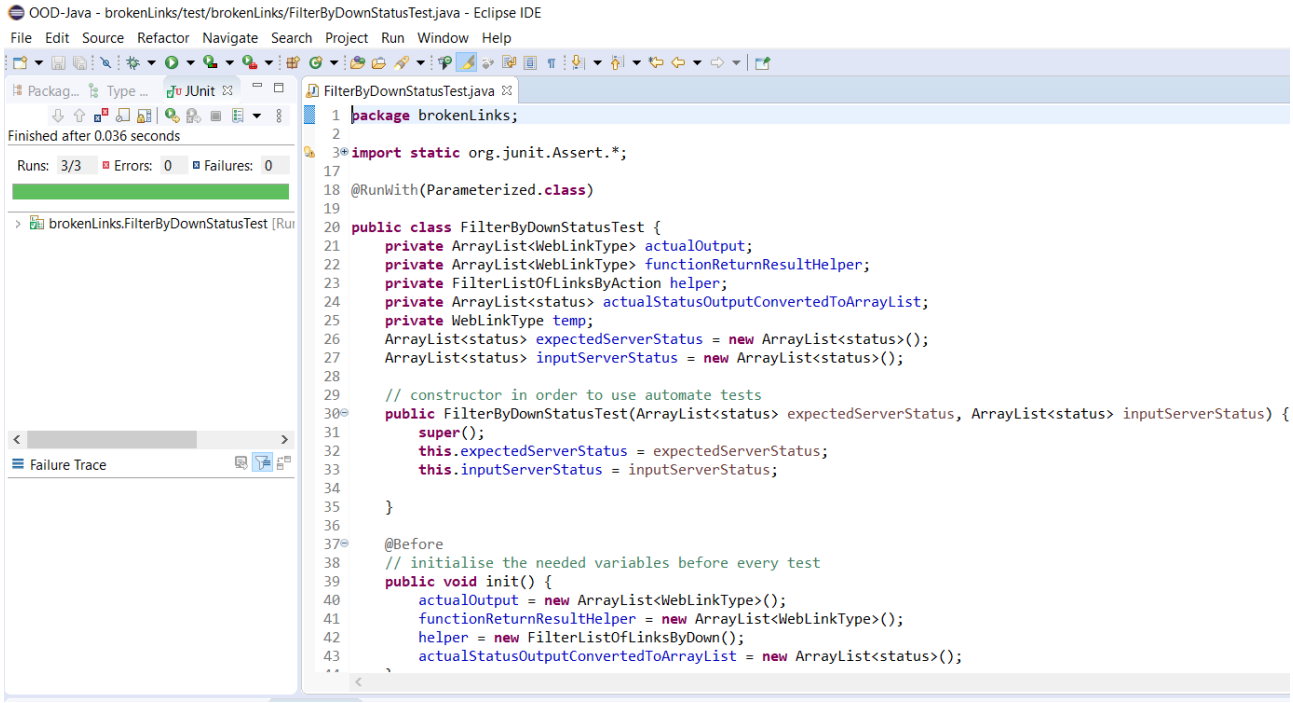
3.5 Application design



The user is suppose to enter the URL in the upper right text field and click the Click to check URL button from the bottom of the app.

4 Experimental data

The tests were made with JUnit for the important classes of the app.



OOD-Java - brokenLinks/test/brokenLinks/LinkTypeBrokenCheckTest.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Packag... Type ... JUnit

Finished after 5.068 seconds

Runs: 2/2 Errors: 0 Failures: 0

brokenLinks.LinkTypeBrokenCheckTest [F

Failure Trace

FilterByDownStatusTest.java LinkTypeBrokenCheckTest.java

```java
package brokenLinks;

import static org.junit.Assert.*;

@RunWith(Parameterized.class)

public class LinkTypeBrokenCheckTest {
    private LinkTypeAction test;
    private ArrayList<String> expectedResponsFromServer = new ArrayList<String>();;
    private ArrayList<String> actualResponsFromServer;
    private ArrayList<String> inputLinksToCheck = new ArrayList<String>();;

    // constructor in order to use automate tests
    public LinkTypeBrokenCheckTest(ArrayList<String> expectedResponsFromServer, ArrayList<String> inputLinksToCheck)
        super();
        this.expectedResponsFromServer = expectedResponsFromServer;
        this.inputLinksToCheck = inputLinksToCheck;

    }

    @Before
    // initialise the needed variables before every test
    public void init() {
        test = new LinkTypeBrokenCheck();
        actualResponsFromServer = new ArrayList<String>();
    }

    @Parameters
    // parameters(input and output)for automate tests
    public static Collection<Object[]> data() {
```

OOD-Java - brokenLinks/test/brokenLinks/SortListByNamesTest.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Packag... Type ... JUnit

Finished after 0.022 seconds

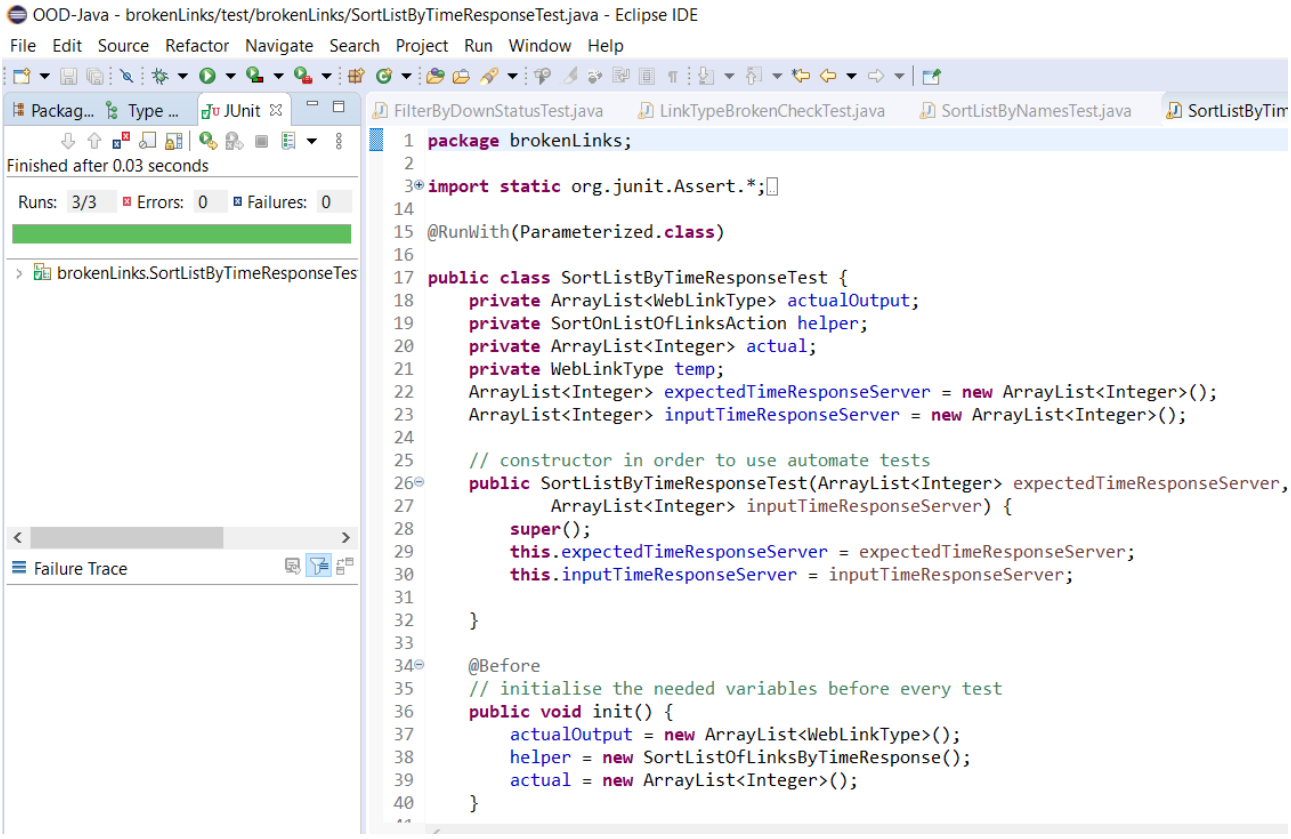
Runs: 2/2 Errors: 0 Failures: 0

brokenLinks.SortListByNamesTest [Runne

Failure Trace

FilterByDownStatusTest.java LinkTypeBrokenCheckTest.java SortListByNamesTest.java

```java
package brokenLinks;

import static org.junit.Assert.*;

@RunWith(Parameterized.class)

public class SortListByNamesTest {
    private ArrayList<WebLinkType> actualOutput;
    private SortOnListOfLinksAction helper;
    private ArrayList<String> actualConvertedToArrayList;
    private WebLinkType temp;
    ArrayList<String> expectedOrderOfNames = new ArrayList<String>();
    ArrayList<String> inputOrderOfNames = new ArrayList<String>();

    // constructor in order to use automate tests
    public SortListByNamesTest(ArrayList<String> expectedOrderOfNames, ArrayList<String> inputOrderOfNames)
        super();
        this.expectedOrderOfNames = expectedOrderOfNames;
        this.inputOrderOfNames = inputOrderOfNames;

    }

    @Before
    // initialise the needed variables before every test
    public void init() {
        actualOutput = new ArrayList<WebLinkType>();
        helper = new SortListOfLinksByName();
        actualConvertedToArrayList = new ArrayList<String>();
    }
```



5 Conclusion

Working at this application helped me gain more experience in JUnit, Java Swing and also it helped me understand how it works. I managed to make me study more on Java programming language and gain more codding ability experience.
From my point of view, the program I implemented, as well as, the tools I used for this app, are correct and both compute the right answer.

References

[1] https://jsoup.org/?fbclid=IwAR1zWNpsbhz4iU2KzWYO5UomM0LUOIT3cZE5TbiDZYeVK17W9XCnwG4uxS8, accessed in April 2020.

[2] geeksforgeeks.org/?fbclid=IwAR0Y2jkrkTRgwGEuLZxznVgZHwJybaqvFaUG1KKBTbiKxMk___ _rH1z7xb4, accessed in April 2020.

[3] https://www.javatpoint.com/java-swing?fbclid=IwAR2nqA8PeQDXvupzmEpJNEm6pWlwcQI8Q5stKF688Nu3GtvNz accessed in April 2020.