URLValidator.txt

Team:
Alex Rappa
Joshua Hesseltine

Class: CS 362
Date: 08/11/2016
Description: Final Project: Part B

Summary: The goal of the Final Project Part B is to develop a testing suite to discover any bugs within the buggy version of URLValidator that has been provided, focusing on the isValid() method. Given an input string, this method returns true if the string is a valid URL or false if the string is an invalid URL.

Outline:

1. Manual Testing

The manual tests were written to test the isValid() function with URL's which are known to be valid or invalid. 29 tests covered 13 valid URL's and 16 invalid URL's.

Valid URL examples:
http://www.google.com
www.google.com
255.255.255.255
https://go.cc:80/test1

Invalid URL examples:
://www.google.com
google.com

256.256.256.256
http/www.google.com

Results: 27/29 passed; 2/29 failed

[Line 54] UrlValidatorTest.java:
Input: “www.google.com”
Expected: true
Output: false

[Line 61] UrlValidatorTest.java:
Input: “255.255.255.255”
Expected: true
Output: false

Our manual testing discovered a bug for the following inputs: “www.google.com” & “255.255.255.255”. The isValid() function incorrectly returned false for these valid URL’s, indicating that a bug is present for valid URL’s with no explicit URL Scheme. We’ll document this bug as Bug #1: Empty Scheme.

2. Partition Testing

These tests partition the input domain by the components of a URL: URL = scheme + authority + port + path + query. Within each partitioned test, all components remain unchanged except for the single component being tested. This strategy aims to identify errors in the isValid() method by isolating each URL component.

a. First Partition: URL Scheme

Different URL schemes were tested in the format: “<scheme>www.google.com” 13 tests covered 5 valid URL’s and 8 invalid URL’s.

Results: 14/15 passed; 1/15 failed

[Line 96] UrlValidatorTest.java:
Input: “www.google.com”
Expected: true
Output: false

When the URL Scheme is not explicit (when <scheme> = “”) the isValid() method incorrectly returns false. This bug was first discovered in Manual Testing and defined as Bug #1: Empty Scheme. All other URL scheme tests returned their expected values.

b. Second Partition: URL Authority

Different URL authorities were tested in the format: “http://<authority>”
15 tests covered 7 valid URL’s and 8 invalid URL’s.

Results: 13/15 passed; 2/15 failed

[Line 121] UrlValidatorTest.java:
  Input: “http://256.256.256.256”
  Expected: false
  Output: true

[Line 122] UrlValidatorTest.java:
  Input: “http://999.999.999.999”
  Expected: false
  Output: true

The isValid() method incorrectly returned true for the invalid URL’s listed above. It failed to recognize the inherent IP decimal limit of 0-255. It did correctly return false for IP address “1000.999.999.999” indicating that the isValid() method may only be checking for a 0-3 digit range. We’ll document this bug as Bug #2: IP Range.

C. Third Partition: URL Port

Different URL ports were tested in the format: “http://google.com:<port>”
11 tests covered 5 valid URL’s and 6 invalid URL’s.

Results: 9/11 passed; 2/11 failed

[Line 136] UrlValidatorTest.java:
  Input: “http://www.google.com:1000”
  Expected: true
  Output: false

[Line 139] UrlValidatorTest.java:
  Input: “http://www.google.com:65535”
  Expected: true
  Output: false

The isValid() method incorrectly returned false for port = 1000, 65535. It did however return true for port = 999, indicating that the isValid() method may only be checking if the port number is a decimal in range 0-999. We’ll document this bug as Bug #3: Port Range.

d. Fourth Partition: URL Path

Different URL paths were tested in the format:
“http://www.google.com<path>”
11 tests covered 4 valid URL’s and 7 invalid URl’s.

Results: 9/11 passed; 2/11 failed

[Line 165] UrlValidatorTest.java:
Input: “http://www.google.com/maps/baps/..slaps”
Expected: false
Output: true

[Line 166] UrlValidatorTest.java:
Input: “http://www.google.com/maps/moremaps/#../mymaps”
Expected: false
Output: true

The isValid() method incorrectly returned true for these two invalid URL paths. It did however return false for path “/maps/..baps”, indicating the isValid() method may stop checking for valid paths after two directories into the path. We’ll document this bug as Bug #4: Path Tree Depth.

e. Fifth Partition: URL Query

Different URL queries were tested in the format:
“http://www.google.com<query>”
8 tests covered 5 valid URL’s and 3 invalid URL’s.

Results:

[Lines 174-178] UrlValidatorTest.java:
Input: “http://www.google.com/something?id=1000”
Input: “http://www.google.com/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8#q=foo%20bar”
Input: “http://www.google.com/?q=foo%20bar”
Input: “http://www.google.com/?id=foo+bar#q=foo+bar”
Input: “http://www.google.com/?action=view”
Expected: true
Output: false

[Lines 180] UrlValidatorTest.java:
Input: “http://www.google.com/something*id=1000”
Expected: false
Output: true

The isValid() method incorrectly returned false for the valid URL’s listed above. Additionally, it incorrectly returned true for one invalid URL query. This indicates that the isValid() method does not correctly validate URL queries, and incorrectly returns false for URL’s that contain a “?” character. We’ll document this bug as Bug #5: Query Character “?”.

3. Iterative Based Testing

The main test methods used for iterative based testing were our partitioning cases:
testYourFirstPartition()
testYourSecondPartition()
testYourThirdPartition()
testYourFourthPartition()
testYourFifthPartition()

These tests each built upon a common string/URL for each call to print to the console. Most used a known valid domain and iterated over various inputs in each partition to check for consistency and validity. We knew, what was and was not valid for each case giving better fidelity to the test case output.

4. Randomized Testing

For the randomized unit test case we created an array of valid URLs of the same universe and looped through the array and called the isValid() method. This seemed appropriate since we created a unit test case to test the isValid() function and not the partitions which are covered seperately.

Due to already having an understanding of how the URLs are built from partitioning based testing, that is: schemes, ports, authority, queries, paths and domains we decided to focus on the Extra Credit Randomize() problem for the URL validator by randomizing each partition and inserting it into a random tester, much the same way we built the testIsValid() method.

The core imported libraries were:
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Random;

The URL builder library allows the seeded random partitions to be random, with testing dependency being only 'valid' URLS. Meaning the randomization of URLS only generate valid strings. Incorporating another test case where the URLs generated randomly create any random string value could be a valuable test case for future development.

5. Bug Report Summary

a. Bug #1: Empty Scheme

Description: isValid() fails to return true for valid URL's when the URL
scheme is left empty (scheme = “”).

Input to produce failure:
    isValid(“www.google.com”) = false
    isValid(“255.255.255.255”) = false

b. Bug #2: IP Range

Description: isValid() fails to check for IP ranges: [0-255].[0-255].[0-255].[0-255]; Returns true for IP's outside of valid range.

Input to produce failure:
    isValid("256.256.256.256") = true
    isValid("999.999.999.999") = true

c. Bug #3: Port Range

Description: isValid() fails for any port numbers > 999 in a valid URL.

Input to produce failure:
    isValid("http://www.google.com:1000") = false
    isValid("http://www.google.com:65535") = false

d. Bug #4: Path Tree Depth

Description: isValid() fails to detect errors in URL's path beyond 2 directories i.e path = /<depth1>/<depth2>/<error_in_depth3>…

Input to produce failure:
isValid("http://www.google.com/maps/baps/..slaps") = true
isValid("http://www.google.com/maps/moremaps/#../mymaps") = true

e. Bug #5: Query Character "?"

Description: isValid() fails for any URL containing a query with character "?". Additionally, queries without "?" character are incorrectly return true.

Input to produce failure:
        isValid("http://www.google.com/something?id=1000") = false
        isValid("http://www.google.com/?q=foo%20bar") = false
        isValid("http://www.google.com/?action=view") = false
    isValid("http://www.google.com/something*id=1000") = true

6. Team Collaboration
Our team managed to effectively share project responsibilities and communicate throughout the class. We used Google Chat as our primary means of communication and would check in each week to discuss our goals, current progress, and pertinent deadlines. We used Github as our collaboration tool to maintain version control. We had no issues despite the inherent space and time limitations in any online program. Each member successfully contributed to the Final project.