Group Members: Rhett Guyer, Allyce McWhorter, Adrienne Pereira

CS 362

06-Dec-2015

Final Project Part B

We performed manual testing as well as partitioning to find bugs in the isValid function. Our unit test combines the 3 partitions we created and calls them to determine what URLs cause the isValid function to give a false answer.

**Manual Testing:**

For manual testing we ran tests of different variations of URLs those we knew were valid and those that should have been invalid to determine if bugs existed. Examples of the some of the URLs we tested are:

• http://www.google.com.us

• http://www.amazon.com

• http://www.google.com.u

• ftp://user:password@google.com

Our manual testing found Bug #1 in the attached bug report. When testing the URL http://www.domain.edu:80/path?query\_string, we expected the validator to validate the URL as true, however it considered it invalid. We introduced print statements in the URLValidator file to tell which of the isValid if statements contained the error. We narrowed it down to the statement: if (!isValidQuery(urlMatcher.group(PARSE\_URL\_QUERY))). Using the debugger in eclipse we traced the issue further to line 446 of URLValidator.java where the return value for the isValidQuery function is !QUERY\_PATTERN.matcher(query).matches() when is should be QUERY\_PATTERN.matcher(query).matcher().

**Input Partitioning:**

For input partitioning we tested by partitioning schemes, domains and ports.

Our scheme partitioning was based on valid schemes that were found at: <http://en.wikipedia.org/wiki/URI_scheme>. We found no bugs related to schemes.

For the domain partitioning we tested the Google URL and appended valid domains(including country codes to them. We got the list of domains from <http://en.wikipedia.org/wiki/List_of_Internet_top-level_domains>. Examples of our URLs are:

<http://www.google.com>

<http://www.google.edu>

<http://www.google.gov>

<http://www.google.int>

<http://www.google.mil>

<http://www.google.ao>

<http://www.google.aq>

<http://www.google.ar>

Domain partitioning found Bug #2. While testing the URL<http://www.google.us> we expected the validator to tell us the URL was valid, however the result of the call to isValid was that the URL was invalid. Using our print statements we added to the URLValidator file, we traced the error to the if (!isValidAuthority(authority)) statement within the isValid function. Running the eclipse debugger we were able to determine the URL couldn’t be validated based on the hostname (which we expected) or the IP address. The hostname could not be validated because the isValidTld(String tld) function in DomainValidator.java returned false instead of the value true that we were expecting. A review of this function helped us to determine that valid country codes were missing from the TLD lists (Bug #2).

Port partitioning found Bug #3. For the port partitioning we chose random numbers to represent IP addresses and the port numbers then combined them to form a URL. Examples of our URLs are:

<http://43.161.199.53:22029>

<http://121.203.38.71:18653>

<http://166.178.47.188:62392>

<http://191.89.15.67:34090>

http://689.614.1121.1197:17940

<http://213.206.225.56:50162>

For most of these URLs, we expected them to validate as true, however they were considered false. These errors also tracked to the if (!isValidAuthority(authority)) in the isValid function. Running the debugger we traced the issue to the if (!PORT\_PATTERN.matcher(port).matches()) statement in the isValidAuthority function. The port in our URL did not match the PORT\_PATTERN that was pre-defined. The debugger did not find the bug specifically, but after reviewing the results of our tests, we noted that ports over 1000 were considered invalid. This combined with the issue with PORT\_PATTERN, helped us determine that the statement “private static final String PORT\_REGEX = "^:(\\d{1,3})$";” contained the bug. The 3 denotes the number of digits the port number can contain. Since valid port numbers go as high as 65,535 this value should be 5 instead. We note that allowing the port number to use 5 digits means that port number 99,999 would then be listed as valid. In addition to the previously mentioned fix an additional section of code should be added to ensure the port number does not exceed 65,535.

**Team Work:**

Our team met via Google Hangouts multiple times to discuss the project. During our first meeting we went through the URLValidator to understand how it worked and the scope of the final project. During this time we also completed part A of the project and setup a time for us to meet again to discuss part B.

The next time we met, all of us had a chance to look over the project requirements and the project in Java. We were also able to perform some manual tests by this time. Our goal of this meeting was to divide the work for each team member and setup another call so we could combine the work we did individually. During the call we discussed our results from the manual tests. We then combined these tests into one manual test and split up the remaining items for the project. Two members were responsible for designing the partitions and the unit test while the last team member was to write up the report.

Once the project was finished we felt like we worked well together as a team. Everyone was able to pull their own weight and help the team reach its goal. Google Hangouts proved to be a very useful tool as we were able to accomplish tasks much faster than we would have been able to via email or a chat program.

**Bug Report:**

Results of our testing can be found in the file URLValidatorTestResults.txt. The bug report can be found below.

Bug Name: Invalid Query Pattern Matching

Bug ID: #1

Date: 12/6/15

Type: Bug

Priority: HIGH

Description:

The URLValidator fails to properly validate URLs containing queries. The bug can be found in line 446 of URLValidator.java in the isValidQuery function as the return value returns the opposite of the correct value.

Bug Location: Line 446 of URLValidator.java.

Bug Name: Omission of TLDs

Bug ID: #2

Date: 12/6/15

Type: Bug

Priority: HIGH

Description:

The URLValidator fails to properly validate URLs containing accurate country codes. The COUNTRY\_CODE\_TLDS lists beginning at line 248 of DomainValidator.java does not contain the full list of applicable country codes.

Location: Missing TLDs should be added to the COUNTRY\_CODE\_TLD\_LIST in DomainValidator.java

Bug Name: Improper port parameters

Bug ID: #3

Date: 12/6/15

Type: Bug

Priority: HIGH

Description: The URLValidator fails to properly validate URLs with valid port numbers greater than 999. The bug can be traced to Line 158 of URLValidator.java which uses 3 as the number of digits for the port numbers as opposed to 5.

Location: 158 in UrlValidator.java

Bug Name: IP Address Segments Greater Than 255 are Allowed

Bug ID: #3

Date: 12/6/15

Type: Bug

Priority: HIGH

Description: The URLValidator fails to return a result of False when an IP segment of an IP address exceeds 255. For example 255.255.255.500 is a vaild IP address according to the vaidator but in reality this is not a valid address.

Location: Line 94 in InetAddressValidator.java