Final Project – Part B: URL Validator Report
Rishi Bhandarkar, Tom Dale, Matt Walz

When testing manually we entered popular websites. Examples include: http://www.amazon.com, http://www.google.com, http://www.reddit.com, http://www.twitch.tv. When doing this manual testing no bugs were found but this was not a very robust way to test URLs as they can vary from each other so much. To get a better grasp on the different sections of a URL and how they can differ we created partitions to test sections of the URL individually.

For all further testing we used arrays of predetermined strings for each section of the URL we were testing. These pre-determined strings were paired with a Boolean to show if that value should return valid or invalid by the URL validator (similar logic to the Apache Commons isValid() test method). The different test arrays we used to test the different partitions included testSchemes, testDomains, testPorts, testPaths, testQueries, and testFragments.

In testYourFirstPartition() we tested valid and invalid schemes. Some test inputs we knew should fail while others should pass. By checking both we made sure the schemes were being tested correctly by the URL Validator when the rest of the URL was valid.

In testYourSecondPartition() we tested for valid and invalid domains. Same as with the schemes, we tested both domains that should pass and others that should fail as well as checked localhost and hostname when local URLs were allowed by the validator. By testing the domain we were able to find bugs detailed in Bug Report 1, Bug Report 2, and Bug Report 3.

In testYourThirdPartition() we tested valid and invalid ports by using valid URLS and adding the different ports on the end. During the testing of valid and invalid ports we found the bug in Bug Report 4.

In testYourFourthPartition() we tested valid and invalid paths inside the URL. No bugs were found as a result of testing various paths attached at the end of a valid URL.

In testYourFifthPartition() we tested queries appended to valid URLs. This testing uncovered the bug detailed in Bug Report 5.

In testYourSixthPartition() we tested valid and invalid fragments at the end of the URL. Similarly to the path testing, no bugs were found as a result of this test.

By breaking the testing into these six partitions we were able to test each section of a URL independently of each other to isolate any bugs that present in a particular section of the URL which greatly assisted in being able to isolate the segment of code containing the bug.

Our last unit test, testMyIsValid(), combines the premise of all the partitions but instead of checking just one section of the URL at a time (with the rest of the URL being valid), it creates every possible combination of URLs using our test arrays, and tests them. This is much more comprehensive and can find bugs that may only be present when there are combinations inside the URL that can create bugs, rather than isolated in one section of the URL.

For the project itself Matt did the coding in java, creating the test suites and doing the bug reports. The report was handled by Tom and Rishi. We used gmail for our communication, and to divide the work.

Bug Report 1

URL Validator fails to validate local URLs when 'ALLOW_LOCAL_URLS' is set

Submitter

Matt Walz

Date Seen

11 March 2016

Tested with following OS and software

Windows 10

Eclipse - Version: Kepler Service Release 2; Build id: 20140224-0627

Bug Description

When testing isValid() method for UrlValidator.java with UrlValidator.ALLOW_LOCAL_URLS set, the method incorrectly returns false for http://hostname and http://localhost.

Minimized test code to reproduce failure:

```
UrlValidator urlVal = new UrlValidator(null, null,
  UrlValidator.ALLOW_LOCAL_URLS);
assertTrue("http://hostname is valid",
  urlVal.isValid("http://hostname"));
assertTrue("http://localhost is valid",
  urlVal.isValid("http://localhost"));
```

Actual Behavior

Running the code listed above will result in both assertTrue checks failing.

Expected Behavior

I expected both assertTrue checks to pass on the above code with the UrlValidator.ALLOW_LOCAL_URLS' value set.

Troubleshooting/Testing Steps Attempted

I believe I have localized the error to the DomainValidator.java file, line number 139. The NOT (!) should be removed from the beginning of the 'if' statement. Thus, line 139 should read as:

```
if (hostnameRegex.isValid(domain)) {
```

Bug Report 2

URL Validator fails to invalidate IPv4 Addresses with byte values > 255

Submitter

Matt Walz

Date Seen

11 March 2016

Tested with following OS and software

Windows 10

Eclipse - Version: Kepler Service Release 2; Build id: 20140224-0627

Bug Description

When testing isValid() method for UrlValidator.java with UrlValidator.ALLOW_LOCAL_URLS set, the method incorrectly returns true for http://256.256.256.256.

Minimized test code to reproduce failure:

```
UrlValidator urlVal = new UrlValidator();
assertFalse("http://256.256.256.256 is invalid",
  urlVal.isValid("http://256.256.256.256"));
```

Actual Behavior

Running the code listed above will result in the assertFalse check failing.

Expected Behavior

I expected the assertFalse check to fail since the maximum integer value of a byte is 255.

Troubleshooting/Testing Steps Attempted

I believe I have localized the error to the inetAddressValidator.java file, line number 96. The 'return' statement after checking

```
if (iIPSegment > 255)
```

should be a 'return false;' (instead of return true as is currently written).

Bug Report 3

URL Validator fails to validate many two-letter country codes used as internet top level domains

Submitter
Matt Walz

Date Seen
12 March 2016

Tested with following OS and software
Windows 10
Eclipse - Version: Kepler Service Release 2; Build id: 20140224-0627

Bug Description
When testing isValid() method for UrlValidator.java, URLs containing two-letter country codes beginning with the letter j or later (alphabetically speaking) as their domain codes are automatically invalidated.

Minimized test code to reproduce failure:

```
UrlValidator urlVal = new UrlValidator();
assertTrue("http://www.google.mz is valid",
  urlVal.isValid("http://www.google.mz"));
```

Actual Behavior
Running the code listed above will result in the assertTrue check failing.

Expected Behavior
I expected the assertTrue check to pass since ".mz" is the correct domain code for the country of Mozambique.

Troubleshooting/Testing Steps Attempted
I believe I have localized the error to the DomainValidator.java file. Beginning at line 250, an array containing strings of acceptable country codes is created, but the array stops with the string "it" for Italy. This array needs to be updated to contain the complete list of two-letter country domain codes.

Bug Report 4

URL Validator fails to validate valid port values > 999

Submitter

Matt Walz

Date Seen

12 March 2016

Tested with following OS and software

Windows 10

Eclipse - Version: Kepler Service Release 2; Build id: 20140224-0627

Bug Description

When testing isValid() method for UrlValidator.java, URLs containing port values greater than 999 are automatically invalidated.

Minimized test code to reproduce failure:

```
UrlValidator urlVal = new UrlValidator();
assertTrue("http://www.google.com:1023 is valid",
  urlVal.isValid("http://www.google.com:1023"));
```

Actual Behavior

Running the code listed above will result in the assertTrue check failing.

Expected Behavior

I expected the assertTrue check to pass since :1023 is a valid port value (last of the well-known port numbers)

Troubleshooting/Testing Steps Attempted

I believe I have localized the error to the URLValidator.java file, line 158. This line creates a regular expression for ports which is limited to 3 digits. If this was changed to allow 5 digits it would allow the validator to account for all well-known, registered, and dynamic/private port numbers. However a check would still be needed to verify that the port value is less than or equal to 65535 as that is the highest allowed dynamic port value.

Bug Report 5

URL Validator fails to validate URLs with queries

Submitter

Matt Walz

Date Seen

12 March 2016

Tested with following OS and software

Windows 10

Eclipse - Version: Kepler Service Release 2; Build id: 20140224-0627

Bug Description

When testing isValid() method for UrlValidator.java, URLs containing query strings are automatically invalidated.

Minimized test code to reproduce failure:

```
UrlValidator urlVal = new UrlValidator();
assertTrue("http://www.google.com?query=answer",
  urlVal.isValid("http://www.google.com?query=answer"));
```

Actual Behavior

Running the code listed above will result in the assertTrue check failing.

Expected Behavior

I expected the assertTrue check to pass since "`?query=answer`" is a valid query string.

Troubleshooting/Testing Steps Attempted

I believe I have localized the error to the URLValidator.java file, line 446. The NOT (!) should be removed from the beginning of the 'return' statement. Thus, line 446 should read as:

```
return QUERY_PATTERN.matcher(query).matches();
```