Daniel Bonnin
Victor Hernandez
Billy Kerns

CS362: Final Project, Part B

This file contains bug reports and documentation of the testing we performed for part B of the final project.

Manual Testing

Methodology of testing:

Manual

Manually tested urls:	Expected	Actual
http://www.amazon.com	True	True
http://www.amazon.org:22	True	True
http://adasd.amazon.com/test1	True	True
http://www.amazon.zw	True	False
http://www.amazon.gov?action=view True	False	

Note on http://www.amazon.zw:

This bug was discovered while looking over the code. We did not just luck into manually testing that url and finding the bug.

Partition Testing

Methodology of testing:

Partition

How did we partition the inputs?

Our partitions were authority, scheme, port, path, and query. For each partition we had a known valid url that would test only that particular partition with a series of valid and invalid parts. For example for the authority partition the url being test would look like this: validScheme + validAuthority||invalidAuthority + validPort + validPath + validQuery.

Debugging

Did you use any of Agan's principle in debugging URLValidator?:

Yes. We used Agan's principles to find bugs in URLValidator. The first rule is to understand the system and the first thing we did was study how URLValidatorTest worked by manually going through the source code. Secondly, we were asked to write our own tests to debug URLValidator and In doing so, we spent some time studying the isValid method, and where the URL validity checking methods came from. By developing these tests, we further explored the system and were able to identify and localize bugs. Agan's second rule is to "Make it fail". In debugging, we marked the known valid and invalid URL components that resulted in failures. We then stimulated these failures by manually testing similar URL components and made notes of the results. These results were used then used to develop program based testing. As we were not tasked with fixing the bugs, many of Agan's Principles were not directly applicable. However we were able to verify that bugs were found by indirectly applying Agan's Principles such as divide and conquer, and by working on one bug at a time to more efficiently work as a group.

Test Names

1. testYourFirstPartition()

Valid schemes tested:	Expected	Actual
http://	True	True
ftp://	True	True
https://	True	True

Invalid schemes tested:		
https	False	False
data://	False	False
zzzzz://	False	False
ftp:/	False	False
ftp	False	False
ftp:://	False	False
ftp:	False	False
https:///	False	False
https:$/	False	False
https//:	False	False

2. testYourSecondPartition()

Valid authorities tested:	Expected	Actual
www.google.com	True	True
google.com	True	True
google.org	True	True
255.com	True	True
google.gov	True	True
google.edu	True	True

Invalid authorities tested:		
256.256.256.256	False	True
1.2.3.4.5	False	False
.1.2.3.4	False	False
go.a1a	False	False
go.1aa	False	False
.aaa	False	False
aaa	False	False
aaa.	False	False
1.2.3	False	False
empty string	False	False

3. testYourThirdPartition()

Valid ports tested:	Expected	Actual
:22	True	True
:65535	True	False
:0	True	True
:6566	True	False
:1	True	True
:12	True	True
:123	True	True
:1234	True	False

:12345	True	False
:11111	True	False

Invalid ports tested:

:123456	False	False
:1b3	False	False
:b21	False	False
:.11111	False	False
:ljljl	False	False
:-1	False	False
:-200	False	False
:-b.	False	False
:1234567	False	False
:-0	False	False

4. testYourFourthPartiton()

Valid paths tested:	Expected	Actual
/test1	True	True
/t123	True	True
/$23	True	True
/test1/	True	True
test1/file	True	True
/java/java_object_classes	True	True
/courses/1555028/assignments/6594488	True	True
/wiki/Uniform_Resource_Identifier	True	True
/search	True	True
/r/cscareerquestions	True	True

Invalid paths tested:

/..	False	False
/../	False	False
/..//file	False	False
/test1//file	False	False
///////	False	False
456	False	False
abc	False	False
_$	False	False
_#	False	False
///^	False	False

5. testYourFithPartition()

Valid queries tested:	Expected	Actual
?action=view	True	False
?action=edit&mode=up	True	False
?newwindow=1&q=url+query	True	False
?module_item_id=16435218	True	False
?some_action=Some_thiNG	True	False
?1111=22222	True	False
?royals=world_series_champs	True	False
?ideas=NoNe	True	False
?last_ONE=false	True	False
?LAST_one=true	True	False

==
Team Collaboration
==
How did we collaborate?:

We primarily used email to communicate, divvy up tasks, and to share updates. In addition to this we used Github to work on the files involved with this project. We didn't really have set tasks for each member. Instead we each looked at what needed to be done and did it. After sharing what we had completed the other members would go over offering their input on anything that needed to be changed.

==
Bug Reports
==
Bug report 1:
Name:
  Valid country domains failing validation.
File:
  DomainValidator.java
Line:
  358
Severity:
  MEDIUM
Priority:
  MEDIUM
Reported Date:
  25NOV2015
Reason:
  Countries missing from COUNTRY_CODE_TLD in DomainValidator.java
Status:
  OPEN
Description:
  All countries after Italy (alphabetically) are omitted from COUNTRY_CODE_TLD array.
Steps to Reproduce:
  Call isValidAuthority("www.msn.<countryCode>"); where countryCode is
  the top level domain for any country after Italy alphabetically.
Expected Results:
  Should return true, but will return false.
Debugging details:
  Manual testing revealed this bug. One of the partitions obvious to our
  team was the top-level domain since they are many and often are obscure.
  Used the eclipse debugger to localize the fault to isValidCountryCodeTld(),
  which references the COUNTRY_CODE_TLD array.

Bug report 2:
Name:
  Valid query causes false invalid
File:
  UrlValidator.java
Line:
  446
Severity:
  HIGH

Priority:
  High
Reported Date:
  27NOV2015
Reason:
  isValidQuery() boolean return value is negated to opposite value.
Status:
  Open
Description:
  Query regex pattern matches nothing. Only returns true on empty query.
Steps to Reproduce:
  call isValid("http://www.google.com:22/test1?action=view");
Expected Results:
  Should return true.
Debugging details:
  We tested known correct/incorrect queries appended to known correct URL strings on the isValid() function as part of testing the query partition of URLs.

  We only found this bug due to isValid() failing with known correct queries.

  Using reserved characters as test negative query strings was problematic. The URL regex will process the query as part of the path unless its first character is a '?'. Also, the regex for checking query strings in URLValidator matches all characters except newline, so it does not reject reserved characters.

  It took a code inspection to localize the fault to line 446 in URLValidator.java: return !QUERY_PATTERN.matcher(query).matches();

  Removing the negation operator fixed the bug, though incorrect queries are still interpreted as part of the path if the '?' is omitted.

<u>Bug report 3:</u>
Name:
  Port numbers higher than 99 are invalidated
File:
  UrlValidator.java
Line:
  158
Severity:
  MEDIUM
Priority:
  MEDIUM
Reported Date:
  28NOV2015
Reason:
  PORT_REGEX invalidates port numbers with > 3 digits.
Status:

Open
Description:
The isValidAuthority() function returns false
due to PORT_REGEX failing to match > 3 digits.
Steps to Reproduce:
Call UrlValidator.isValid("http://www.amazon.com:2222");
Expected Results:
Should return true. Current bug results in this returning false.
Debugging details:
In our test of the port "partition", we established the
upper boundary of port numbers as the max 16 bit unsigned int,
which is 65535.

It seemed that the length of the port string was the issue.

Using the Eclipse debugger, we localized the fault to
the PORT_REGEX on line 158 of UrlValidator.java, which
enforces a maximum of 3 digits.

Correcting this regex to allow 5 digits would validate
all valid ports, but also permit some false positives (eg. 65535 < portNum < 99999).

This may or may not be out of scope for our concerns, however.

<u>Bug report 4:</u>
Name:
inValid ip address is validated
File:
InetAddressValidator.java
Line:
96
Severity:
MEDIUM
Priority:
MEDIUM
Reported Date:
5DEC2015
Reason:
Incorrect return value in segment value check.Validates all 3 digit ip segments.
Status:
Open
Description:
Boolean return value is flipped.
Steps to Reproduce:
call isValid("http://999.999.999.999");
Expected Results:
Should return false.
Debugging details:
We included the ip address option in our covering array options set as a negative test.
Our oracle asserted that the test should return invalid, so the validation alerted us to this bug.
Using the debugger, we stepped through the execution until noticing the fault in InetAddressValidator.java.