Report

Test Cases

December 20, 2017

Author:
Caroline Nilsson *(cn222nd)*
Daniel Alm Grundström *(dg222dw)*
Term: HT 2017
Course: 2DV610 - Software Testing

Contents

1 Use Case: Start Server

This section provides Test Cases for Use Case 1 starting the server. Below is a list of steps that shall be performed before each Test Case.

- Open the terminal
- Navigate to the .jar file location

1.1 Unavailable socket (Manual)

The system shall notify the administrator if the provided port socket is unavailable.
Test Input
Port socket: 80 *incorrect*
Shared resource: MyWebServer-master 12.15.02/tests/se/lnu/http/resources/inner/
correct

1.1.1 Input

- Navigate to the .jar file location
- input `"java -jar MyWebServer.jar 80 MyWebServer-master 12.15.02/tests/se/lnu/http/resources/inner/"`
- Press enter
- Open a web browser
- Enter `"localhost:80"`
- Press enter

1.1.2 Output

Web Browser

- Display unable to connect to the server

Console

- `"Socket 80 was taken"` shows in console window

1.2 Restriction on shared resource container (Manual)

The system shall give an error when the shared resource container is protected.
Test Input
Port socket: 8080 *correct*
Shared resource: `/var/root/` (MAC) `/root/` (Linux)
incorrect

1.2.1 Input

- Navigate to the .jar file location
- input "`java -jar MyWebServer.jar 8080 /var/www/`"
- Press enter

1.2.2 Output

Console

- "`No access to folder /var/root`" on Mac and "`No access to folder /root/`" on Linux shows in console window

1.3 Access log could not be written (Manual)

The system shall report when the access log could not be written to.
Test Input
Port socket: 8080 *correct*
Shared resource: MyWebServer-master 12.15.02/tests/se/lnu/http/resources/inner/
correct

1.3.1 Input

- Navigate to the .jar file location
- If there is an existing log.txt, remove it
- Enter `touch log.txt`
- Enter `chflags uchg log.txt` on Mac and `chmod 000 log.txt` on Linux
- input "`java -jar MyWebServer.jar 8080 MyWebServer-master 12.15.02/tests/se/lnu/http/resources/inner/`"
- Press enter

1.3.2 Output

Console

- "`Cannot write to server log file log.txt`" shows in console window

1.3.3 After test

- Remove log.txt `chflags nouhcg log.txt && rm -f log.txt` on Mac and `rm -f log.txt` on Linux

1.4 Starting Server (Manual)

An administrator should be able to start the server by running the .jar file and provide port socket and shared resources folder.
Test Input
Port socket: 8080
Shared resource: MyWebServer-master 12.15.02/tests/se/lnu/http/resources/inner/

1.4.1 Input

- Navigate to the .jar file location
- input "`java -jar MyWebServer.jar 8080 MyWebServer-master 12.15.02/tests/se/lnu/http/resources/inner/`"
- Press enter key
- Open a web browser
- Enter "`localhost:8080`"
- Press enter

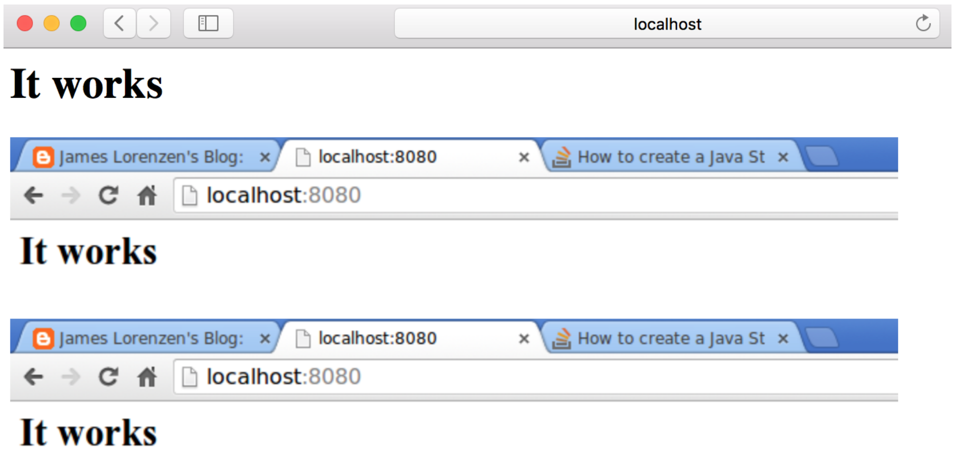
1.4.2 Output

Web Browser

- "It works" is shown on the page *(see figure: 1)*

Console

- "`HTTP Server Started`" is shown in console window



It works

It works

It works

Figure 1: Output for successfully starting server

2 Use Case: Stop Server

Before each test in this section perform test case 1.4 in order to start the server and assure it runs correctly.

2.1 Stopping Server (Manual)

An administrator should be able to stop the server by inputting "stop" into a running server's command line.

2.1.1 Input

- Input "stop" into the server's command line window and press Enter
- Open a web browser and navigate to "http://localhost:8080/"

2.1.2 Output

- A page with the header "Unable to connect" should appear in the web browser.

2.2 Access log written to when server is stopped (Manual)

Make sure that an entry is written to the server's access log when an administrator manually stops the server.

2.2.1 Input

- Input "stop" into server's command line window and press Enter
- Open server access log in a text editor

2.2.2 Output

- The text "HTTP Server stopped" should be displayed on the last line of the access log.

3 Use Case: Request Shared Resource

Before each test in this section perform test case 1.4 in order to start the server and assure it runs correctly.

3.1 Return Response 200 OK (JMeter)

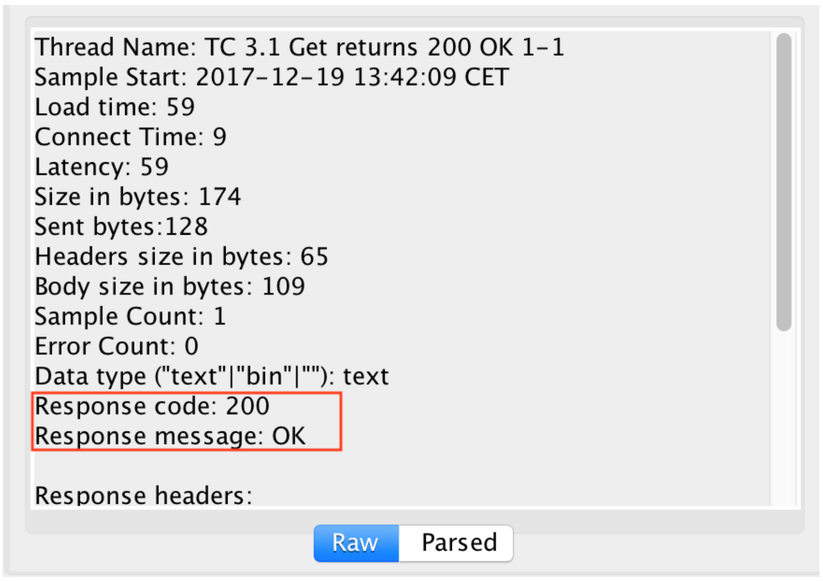
Ensure that the server return HTTP 1.1 response 200 when a request was successful.

3.1.1 Input

- *HTTP Request* GET: `index.html`
- Run test

3.1.2 Output

- Server response with response code 200 *(See fig. 2)*



```
Thread Name: TC 3.1 Get returns 200 OK 1-1
Sample Start: 2017-12-19 13:42:09 CET
Load time: 59
Connect Time: 9
Latency: 59
Size in bytes: 174
Sent bytes:128
Headers size in bytes: 65
Body size in bytes: 109
Sample Count: 1
Error Count: 0
Data type ("text"|"bin"|""): text
Response code: 200
Response message: OK

Response headers:
```

Figure 2: 200 OK

3.2 Return Response 404 NOT FOUND (JMeter)

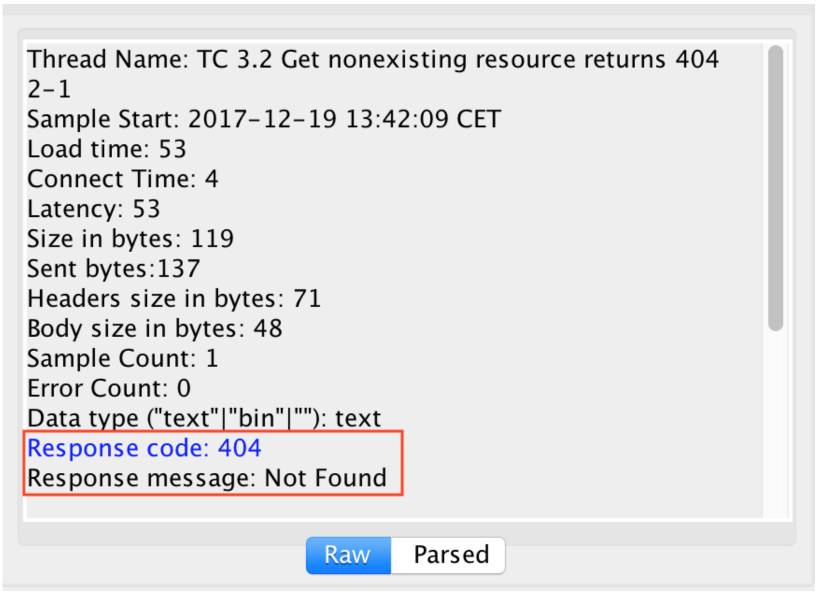
Ensure that the server return HTTP 1.1 response 404 when a requested resource does not exist.

3.2.1 Input

- *HTTP Request* GET: `nonexistingfile.txt`
- Run test

3.2.2 Output

- Server response with response code 404 *(See fig. 3)*



```
Thread Name: TC 3.2 Get nonexisting resource returns 404
2-1
Sample Start: 2017-12-19 13:42:09 CET
Load time: 53
Connect Time: 4
Latency: 53
Size in bytes: 119
Sent bytes:137
Headers size in bytes: 71
Body size in bytes: 48
Sample Count: 1
Error Count: 0
Data type ("text"|"bin"|""): text
Response code: 404
Response message: Not Found
```

Figure 3: 404 Not Found

3.3 Return Response 403 FORBIDDEN (JMeter)

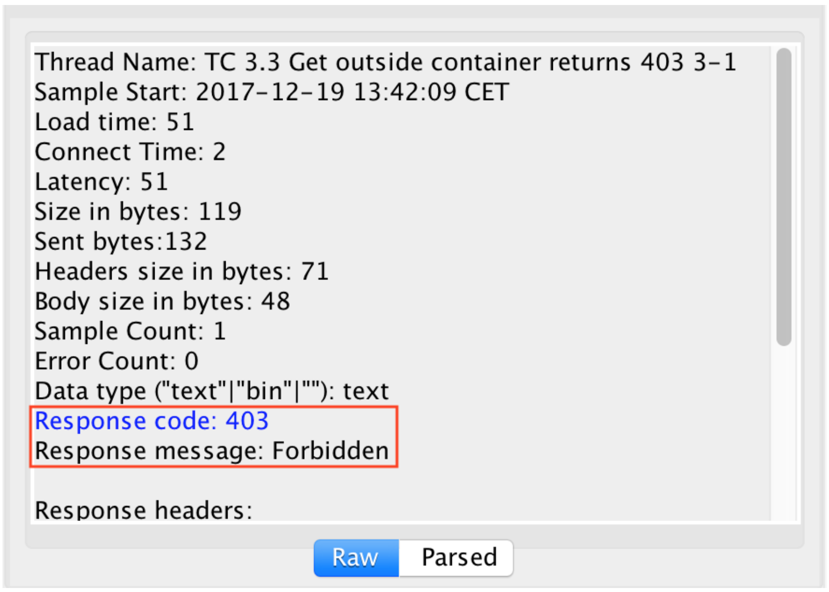
Ensure that the server return HTTP 1.1 response 403 when the requested resource is outside the shared resource container.

3.3.1 Input

- *HTTP Request* GET: `../secret.html`
- Run test

3.3.2 Output

- Server response with response code 403 *(See fig. 4)*



```
Thread Name: TC 3.3 Get outside container returns 403 3-1
Sample Start: 2017-12-19 13:42:09 CET
Load time: 51
Connect Time: 2
Latency: 51
Size in bytes: 119
Sent bytes:132
Headers size in bytes: 71
Body size in bytes: 48
Sample Count: 1
Error Count: 0
Data type ("text"|"bin"|""): text
Response code: 403
Response message: Forbidden

Response headers:
```

Figure 4: 403 Forbidden

3.4 Return Response 400 BAD REQUEST (JMeter)

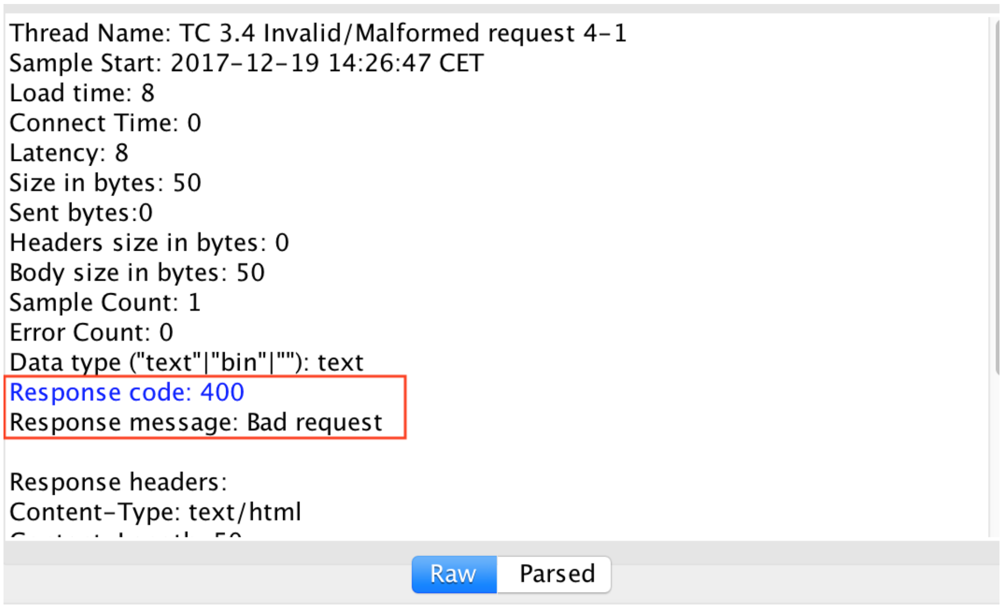
Ensure that the server return HTTP 1.1 response 400 when the URL is malformed.

3.4.1 Input

- *HTTP Request* GET: `%index.html`
- Run test

3.4.2 Output

- Server response with response code 400 *(See fig. 5)*



```
Thread Name: TC 3.4 Invalid/Malformed request 4-1
Sample Start: 2017-12-19 14:26:47 CET
Load time: 8
Connect Time: 0
Latency: 8
Size in bytes: 50
Sent bytes:0
Headers size in bytes: 0
Body size in bytes: 50
Sample Count: 1
Error Count: 0
Data type ("text"|"bin"|""): text
Response code: 400
Response message: Bad request

Response headers:
Content-Type: text/html
```

Figure 5: Bad Request

4 Requirement: 1 Responsive Under High Load

4.1 System response time (JMeter)

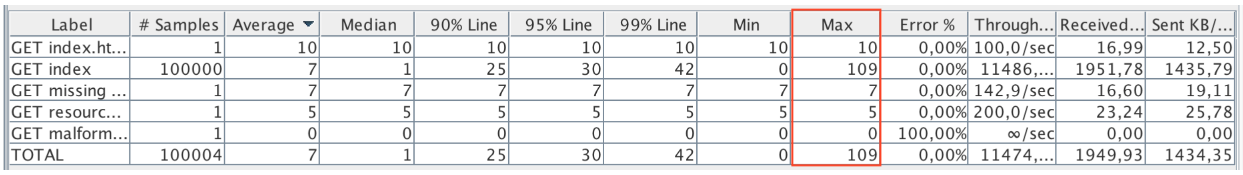
The system should response within 2 seconds when no more than 100 users access the shared resources.

4.1.1 Input

- Perform test case 1.4
- *Thread Group* Users: 100
- *Thread Group* Loop Count: 1000
- *HTTP Request* GET: `index.html`
- Run test

4.1.2 Output

- maximum server response time does not exceed 2000 ms (2 sec) *(See fig. 6)*



Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %	Through...	Received...	Sent KB/...
GET index.ht...	1	10	10	10	10	10	10	10	0,00%	100,0/sec	16,99	12,50
GET index	100000	7	1	25	30	42	0	109	0,00%	11486,...	1951,78	1435,79
GET missing ...	1	7	7	7	7	7	7	7	0,00%	142,9/sec	16,60	19,11
GET resourc...	1	5	5	5	5	5	5	5	0,00%	200,0/sec	23,24	25,78
GET malform...	1	0	0	0	0	0	0	0	100,00%	∞/sec	0,00	0,00
TOTAL	100004	7	1	25	30	42	0	109	0,00%	11474,...	1949,93	1434,35

Figure 6: Maximum Server Response Time

5 Informal Requirement: Usability

5.1 Show Help Text When Faulty Parameter Is Given

The web server should show help text informing the user on how to start the server when the user gives a faulty command line argument.

5.1.1 Input

- Navigate to the MyWebServer.jar folder in the console window
- Input "`java -jar MyWebServer.jar 0`"
- Press enter

5.1.2 Output

- The text "`Enter a valid port 1-65 535 and a optional URL`" is shown in the console window

6 Informal Requirement: Security

6.1 HTTPS

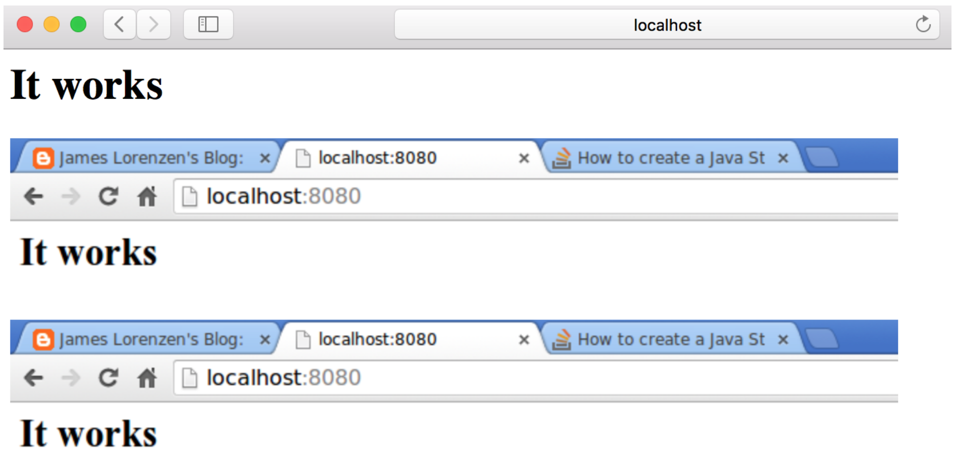
The web server shall use *HTTPS* as standard.

6.1.1 Input

- Perform test case 1.4
- Open a web browser
- Enter "`https://localhost:8080`" into the address bar
- Press enter

6.1.2 Output

- "It works" is shown on the page *(see figure:* **??***)*



It works

It works

It works

Figure 7: Output for successfully starting server