Product manual for Mimic service

About the product:

Mimic service is a virtual learning service using a web-based platform. The purpose for this mimic service is to simulate the behaviour of any http service. It has different functions that helps users to learn, unlearn, relearn different requests and responses, letting them to reset states as well as terminate the mimic service.

Key Features:

- Learns to respond according to requests, with or without using the form
- Remove the previous saved response
- Auto detects the mime type
- Behaves differently according to the current internal state of the service
- Learns sequence of responses to the same request
- Saves the defined requests and responses
- Can be used by any http application or a web browser
- Learn complex responses faster
- View all responses that Mimic have learned

Installation and Run:

If you are an Window user :
Simply download *mimic.jar* from ITHS and double click *mimc.jar*. If *mimic.jar* file has been updated, then the old version should be replaced with the latest version.

If you are an Linux/Mac user:
Please type: **java -jar mimic.jar**
It uses port 8080 by default, to use another port for example 1235, use the following command: **java -jar mimic.jar -p 1235**

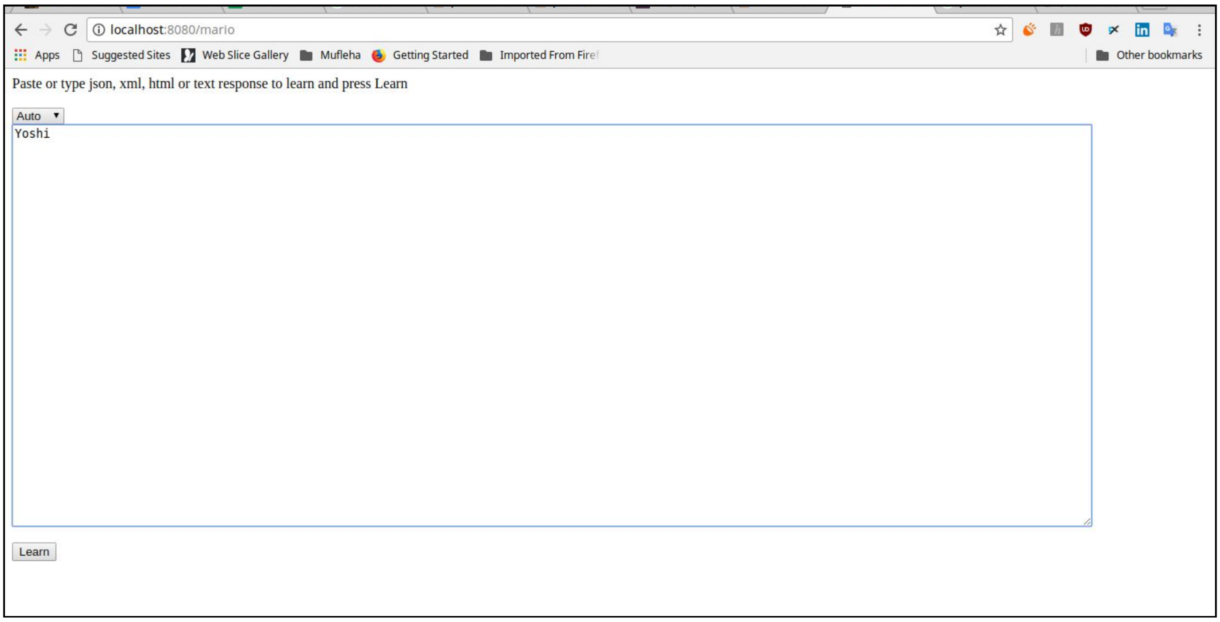
A web browser or any http application can be used to access mimic service.

How to use mimic service:

Learn a response:

To learn a response using the form:

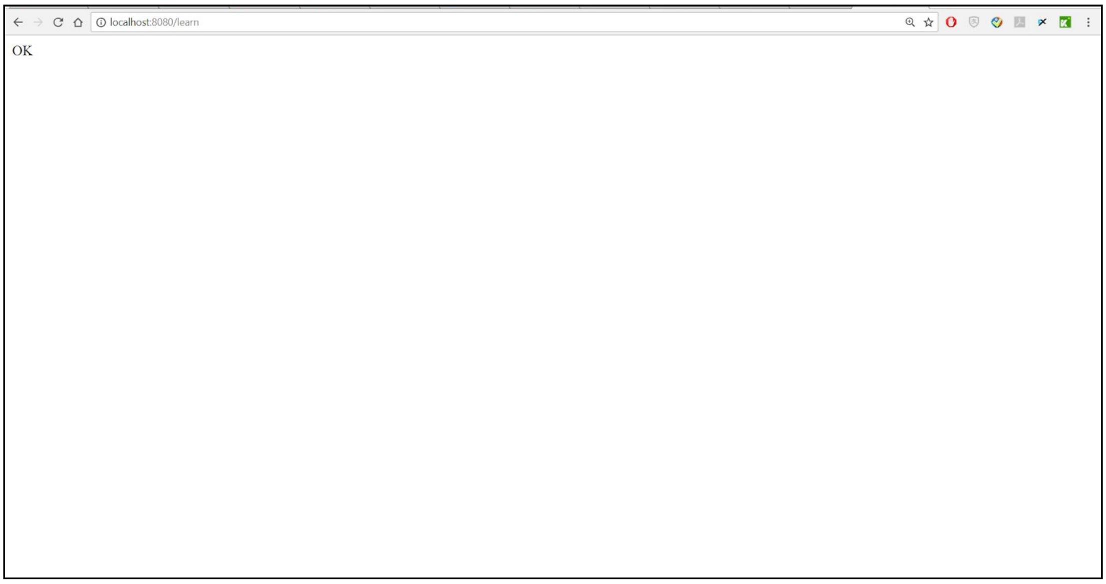
1. Send any http command as request to get a form as response, for example: *http://localhost:8080/Mario*
2. Fill in any response in the form.



Paste or type json, xml, html or text response to learn and press Learn

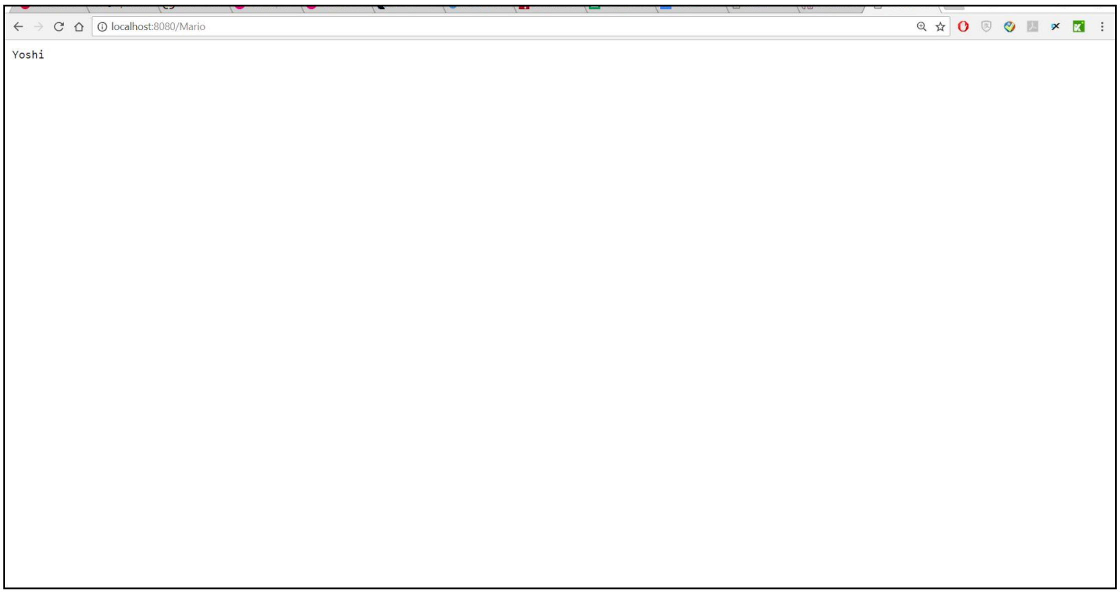
```
Yoshi
```

3. Click ‘Learn’.



OK

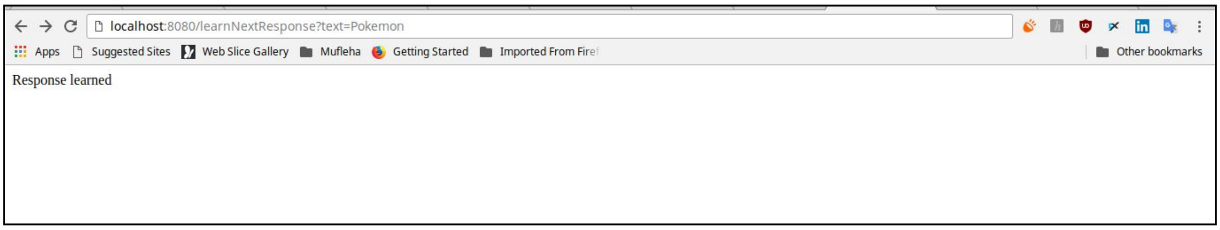
4. Trigger the request you send previously, which is ‘Mario’ to get the response that mimic have learned.



```
Yoshi
```

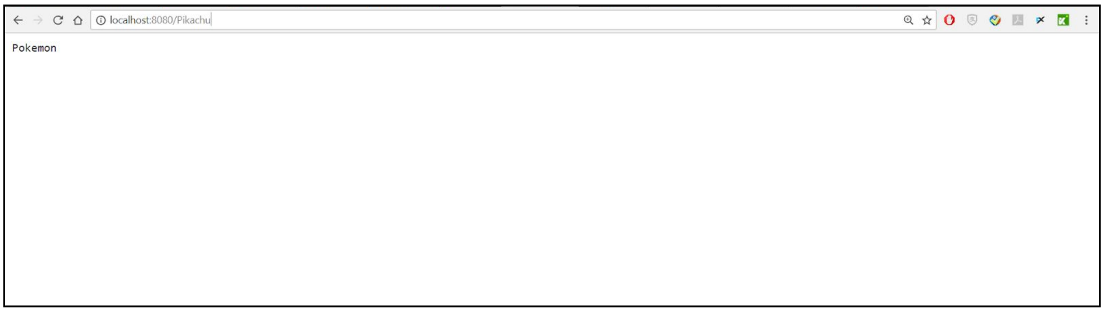
To learn a response without using the form:

1. Define a response before triggering a request to prevent the form from appearing by using the following command: e.g. *http://localhost:8080/learnNextResponse?text=Pokemon*
   Any arguments can be given after ‘text’ as your response.



```
Response learned
```

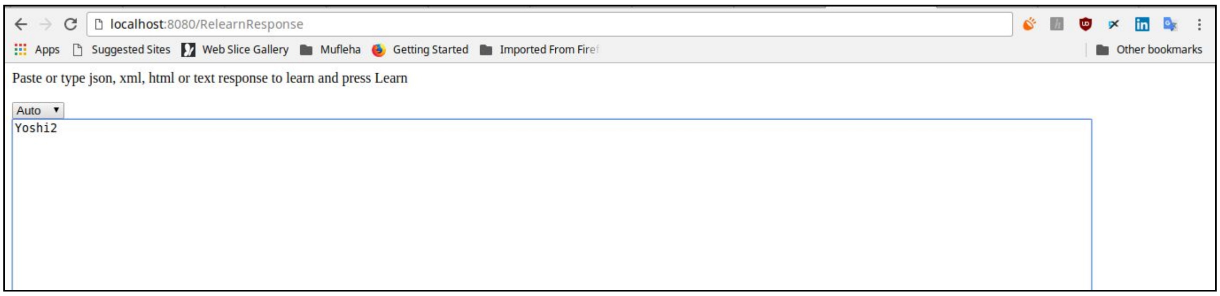
2. Trigger the response by giving an request (could be any request) after you see ’Respponse learned’ showing up:
   Using this format: http://localhost:8080/Pikachu



```
Pokemon
```

Replace a response:

1. To replace a learned response with a new one, use ‘RelearnResponse’, fill in the new response and click “Learn”:



localhost:8080/RelearnResponse

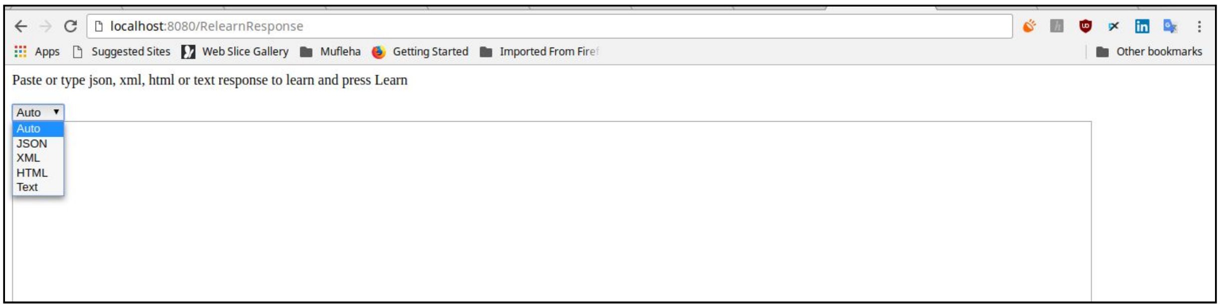
Paste or type json, xml, html or text response to learn and press Learn

Auto

Yoshi2

Specify mime type while learning a response:

1. By default it auto detects the mime type, it can be changed by selecting the mime type from the dropdown list:

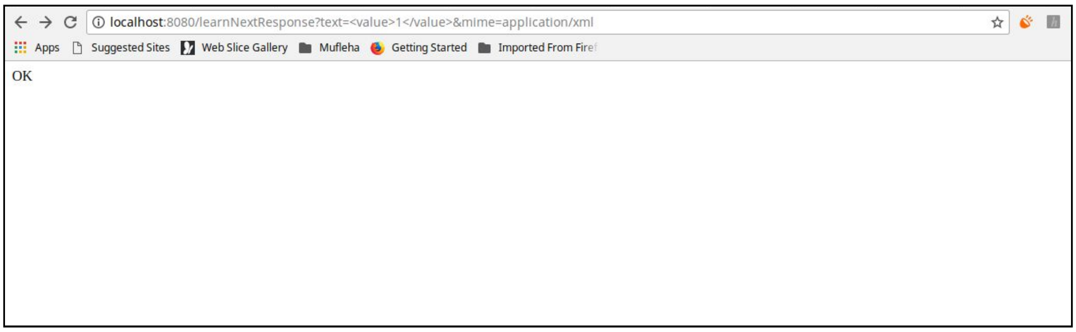


localhost:8080/RelearnResponse

Paste or type json, xml, html or text response to learn and press Learn

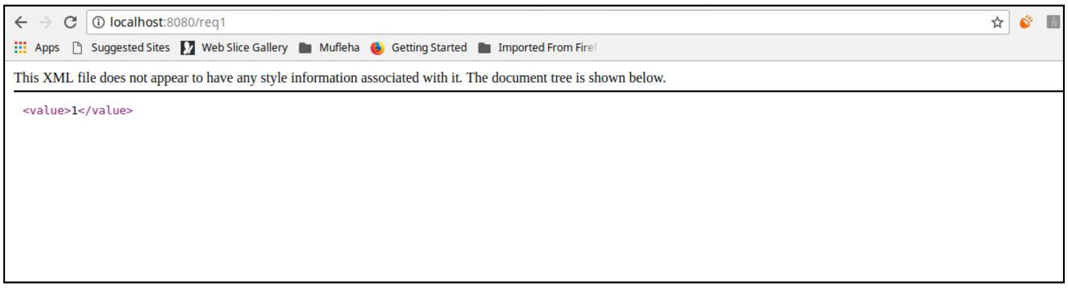
Learn next response by specifying the mime type:

1. Use the following command: http://localhost:8080/learnNextResponse?text=<value>1</value>&mime=application/xml



localhost:8080/learnNextResponse?text=<value>1</value>&mime=application/xml

OK

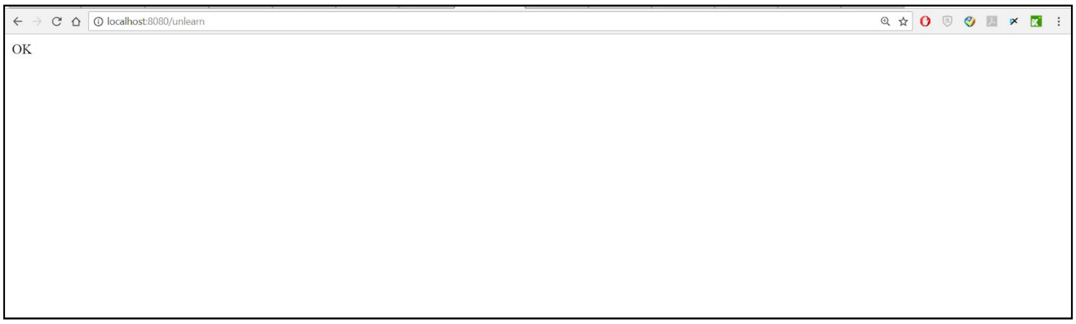


This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<value>1</value>
```

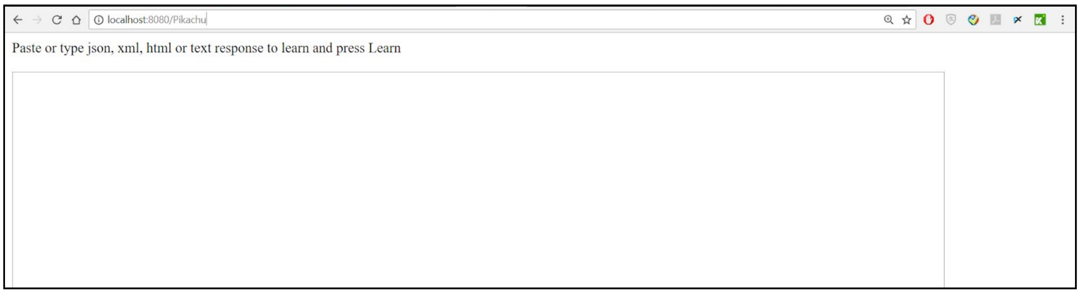
Unlearn a response/request:

To remove a response by applying the 'unlearnResponse' command: http://localhost:8080/unlearnResponse



OK

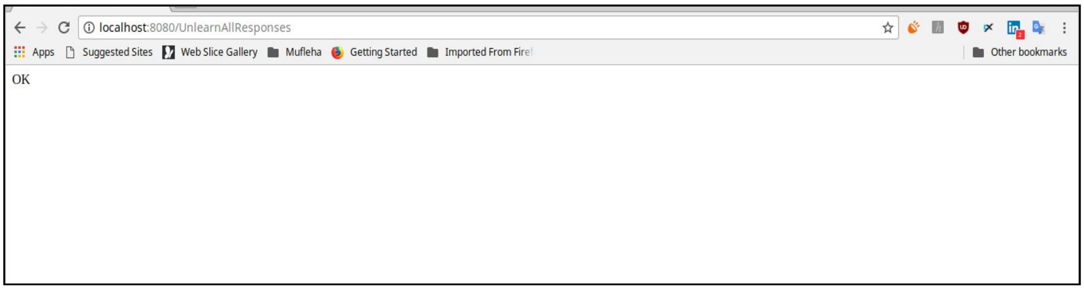
When triggering the previous request 'Pikachu', the corresponding request has been removed:



Paste or type json, xml, html or text response to learn and press Learn

Unlearn all responses using:

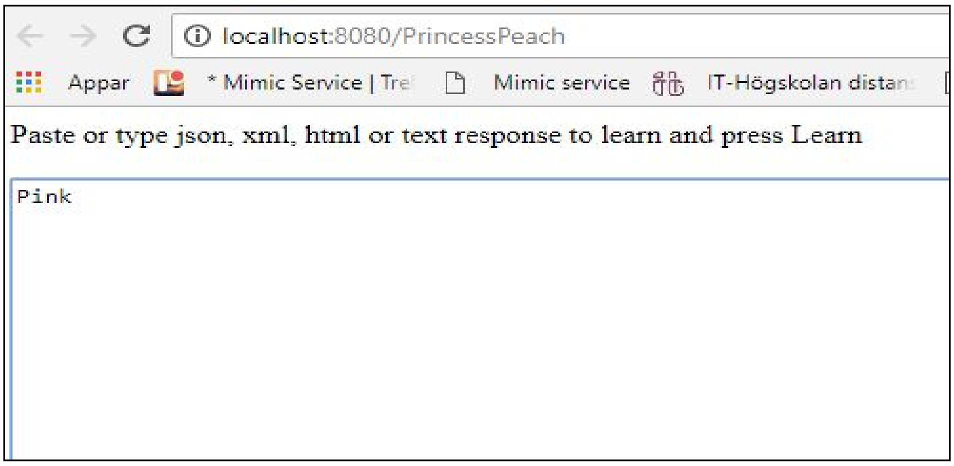
http://localhost:8080/UnlearnAllResponses



OK

Adding multiple responses using the form and the AddResponse command:

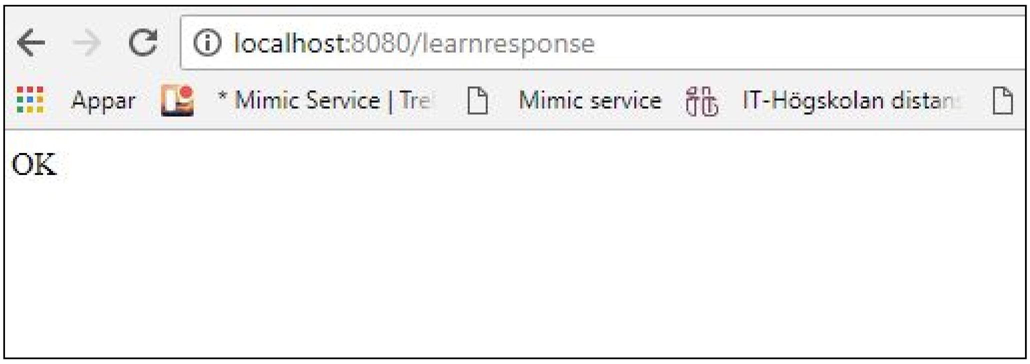
1. Send any http command as request to get a form as response, for example: http://localhost:8080/PrincessPeach



Paste or type json, xml, html or text response to learn and press Learn

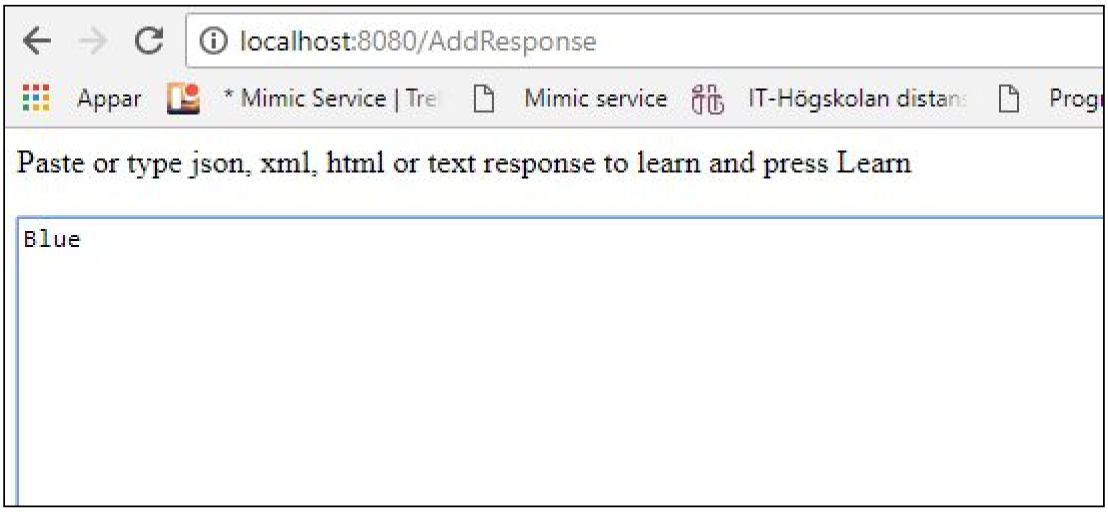
```
Pink
```

2. Enter your choice of response. Press Learn to save the response. Mimic will confirm the learned response with “OK”.



OK

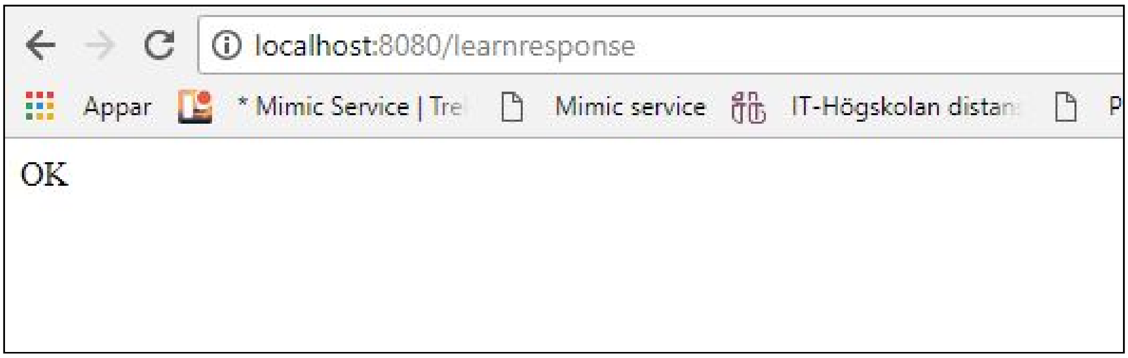
3. Enter http://localhost:8080/AddResponse to add another response.



Paste or type json, xml, html or text response to learn and press Learn

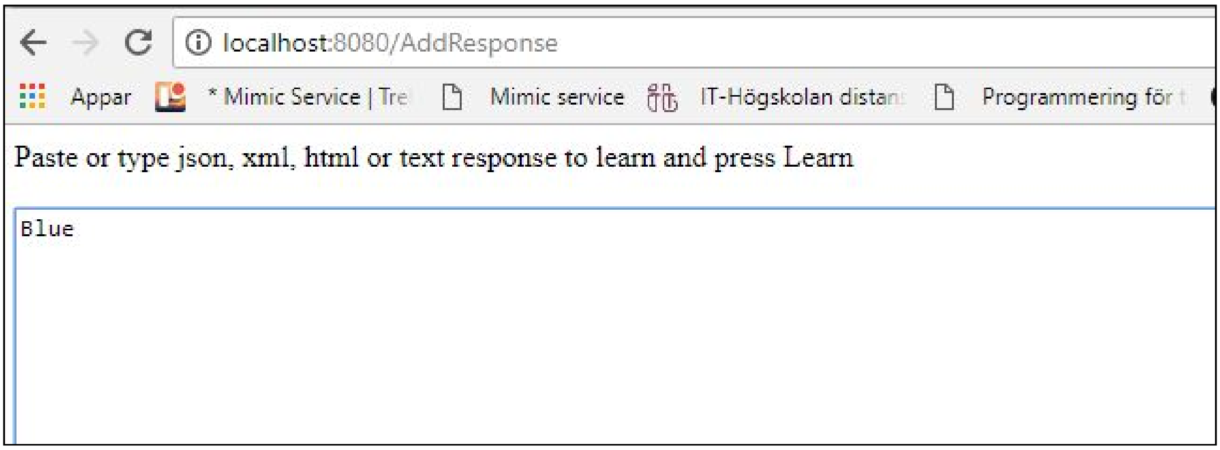
```
Blue
```

4. Enter your choice of response. Press Learn to save the response.



OK

5. Repeat the steps until you've saved all the responses.
   AddResponse works the same as LearnNextResponse:



Paste or type json, xml, html or text response to learn and press Learn

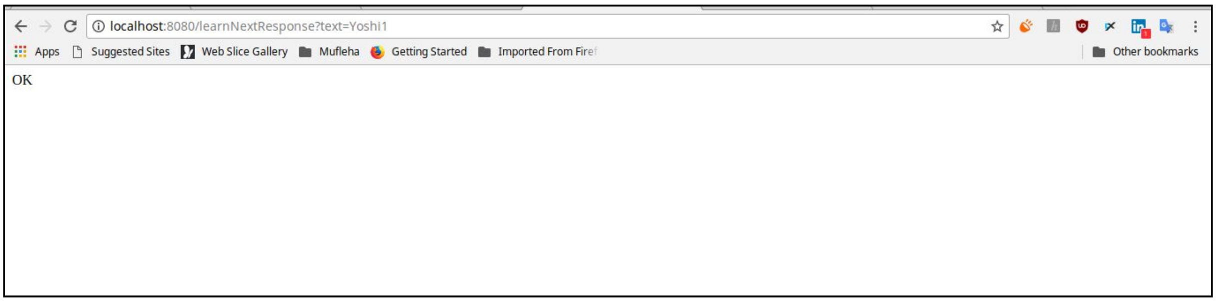
```
Blue
```

Adding multiple responses without using the form:

A request can learn multiple responses using 'learnNextResponse' and by triggering the same request multiple times gives different response until the sequence ends. See following example where a request 'req1' gets three different responses 'Yoshi1', 'Yoshi2' and 'Yoshi3'. Triggering 'req1' three times responds like this:

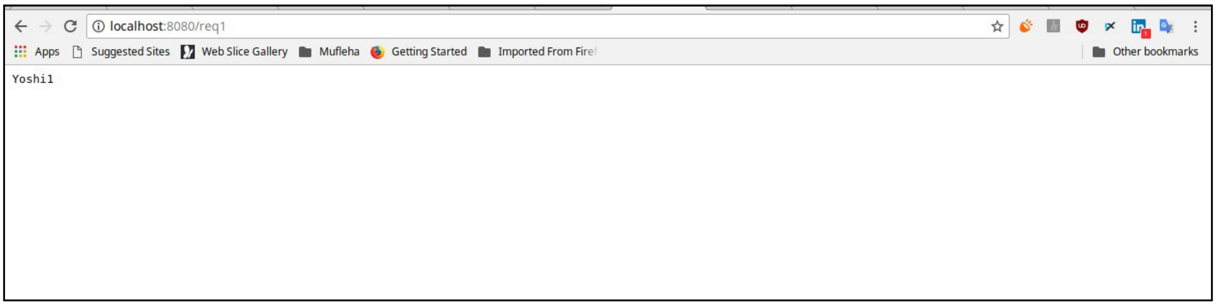
1. req1 Yoshi1
2. req1 Yoshi2
3. req1 Yoshi3

Learn a request:
http://localhost:8080/learnNextResponse?text=Yoshi1



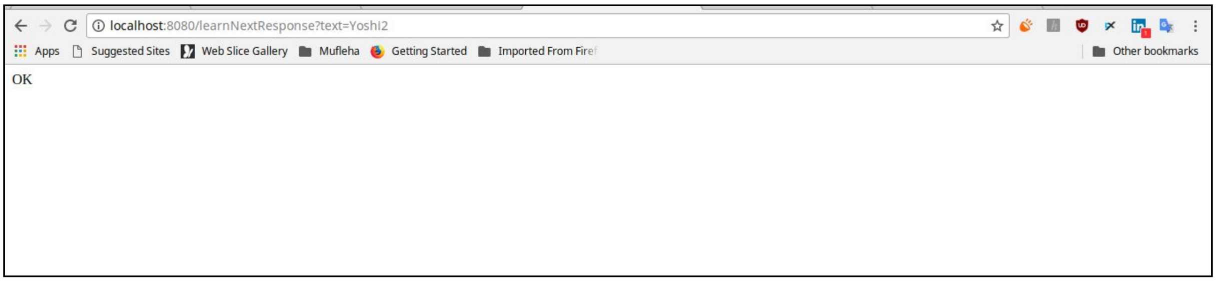
```
OK
```

Assign the learned response to req1:
http://localhost:8080/req1



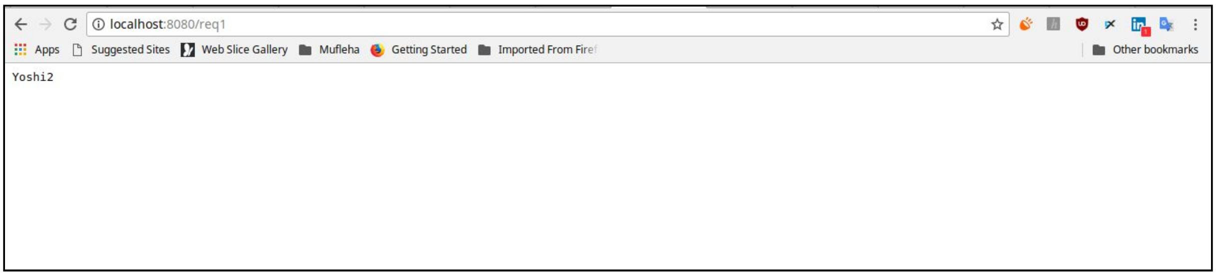
```
Yoshi1
```

Learn another request:
http://localhost:8080/learnNextResponse?text=Yoshi2



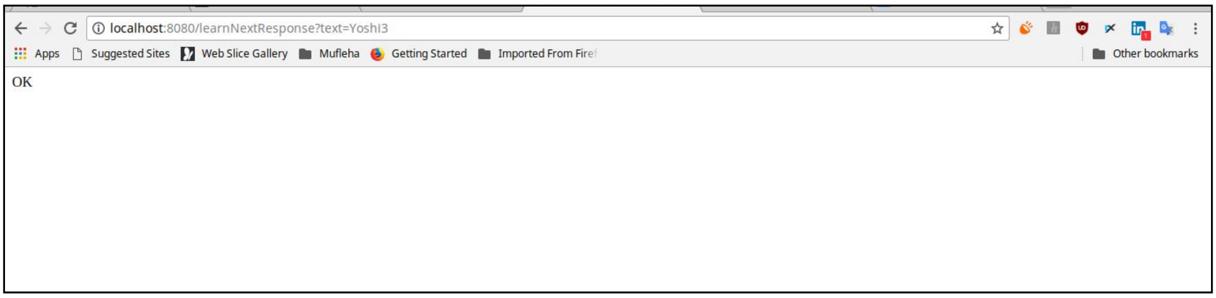
```
OK
```

Assign the learned response to req1 again:
http://localhost:8080/req1



```
Yoshi2
```

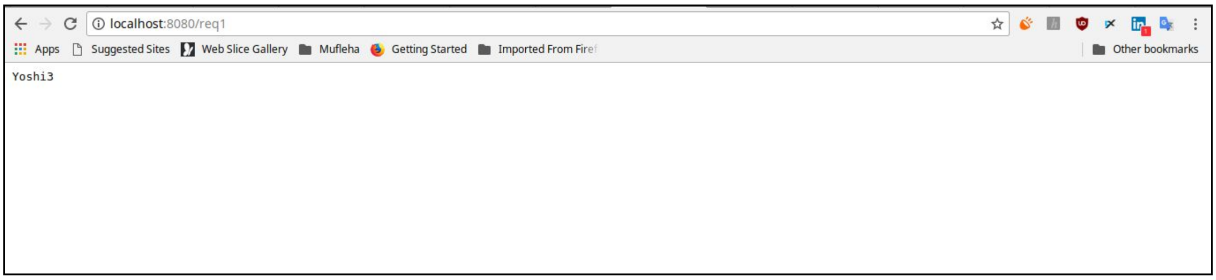
Learn another request:
http://localhost:8080/learnNextResponse?text=Yoshi3



localhost:8080/learnNextResponse?text=Yoshi3

```
OK
```

Assign the learned response to req1 again:
http://localhost:8080/req1

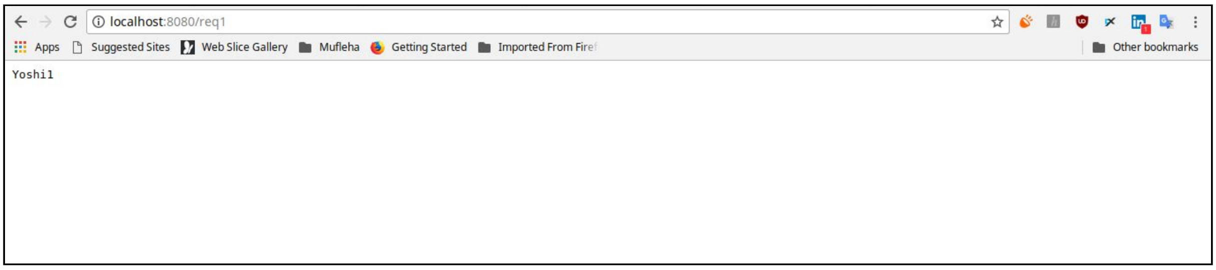


localhost:8080/req1

```
Yoshi3
```

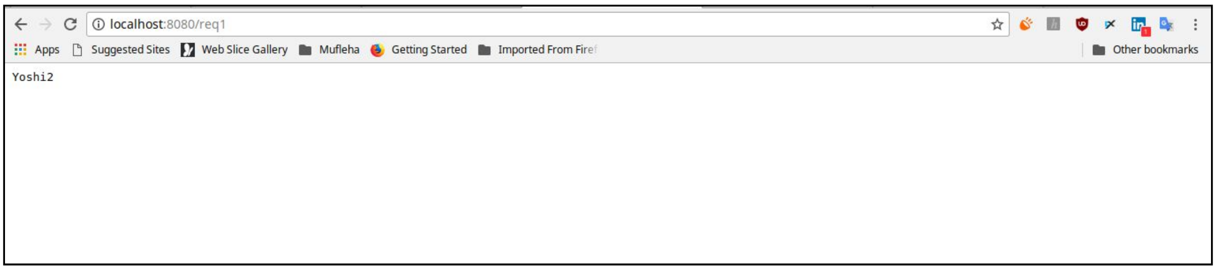
How to check if it responds differently:

Sending ‘req1’ will respond differently until it reaches the last response:



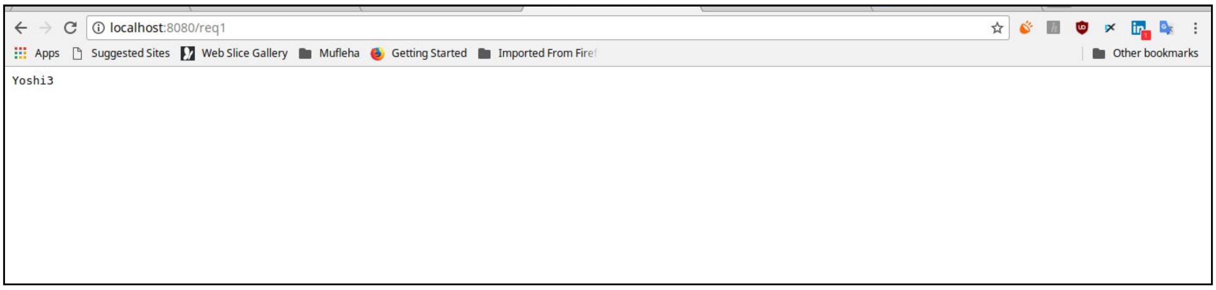
localhost:8080/req1

```
Yoshi1
```



localhost:8080/req1

```
Yoshi2
```



localhost:8080/req1

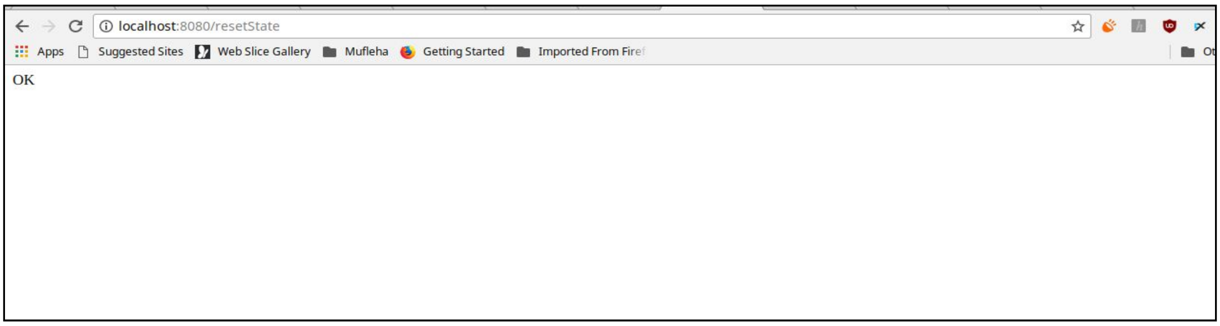
```
Yoshi3
```

Unlearn a response in the middle of the sequence by applying the ‘unlearn’ command:

1. Save several responses as shown in previous step “Adding multiple responses”.
2. Save the responses Yoshi1, Yoshi2, Yoshi3, Yoshi4 and Yoshi5 for request “req1”.
3. Check the response by entering http://localhost:8080/req1 3 times, to stay in Yoshi3 mode.
4. Use unlearn command by entering http://localhost:8080/UnlearnResponse.
5. It removes all the responses after Yoshi3. In this example the responses Yoshi4 and Yoshi5 will be removed.

Reset State:

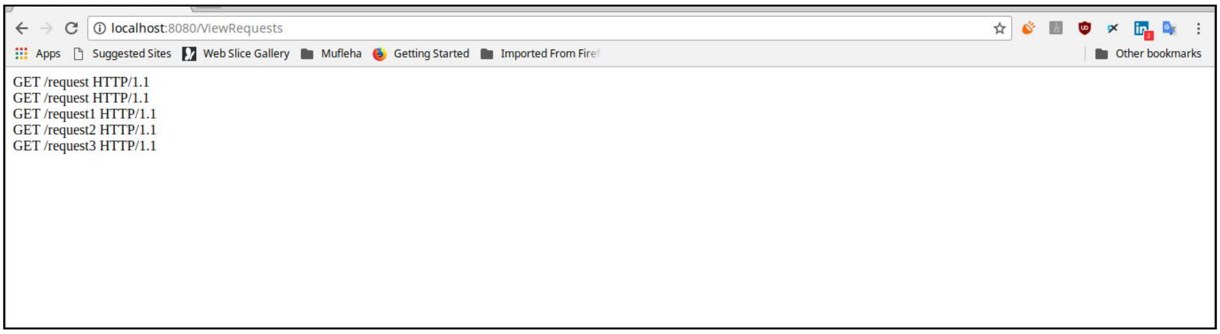
1. Reset the simulated application state (to repeat a sequence again) using: http://localhost:8080/resetState



```
OK
```

View all requests:

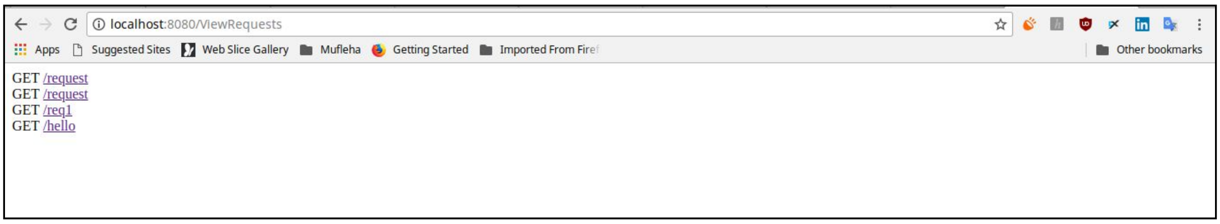
1. To view all the requests use the following command: http://localhost:8080/ViewRequests



```
GET /request HTTP/1.1
GET /request HTTP/1.1
GET /request1 HTTP/1.1
GET /request2 HTTP/1.1
GET /request3 HTTP/1.1
```

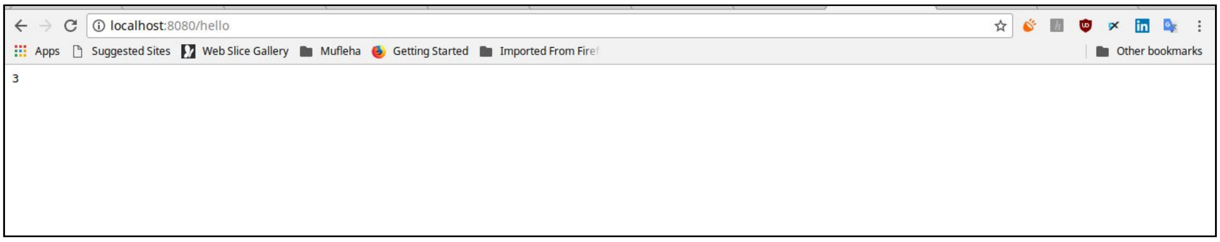
View all responses to the requests:

1. Click on a request from the list to see its responses.



```
GET /request
GET /request
GET /req1
GET /hello
```

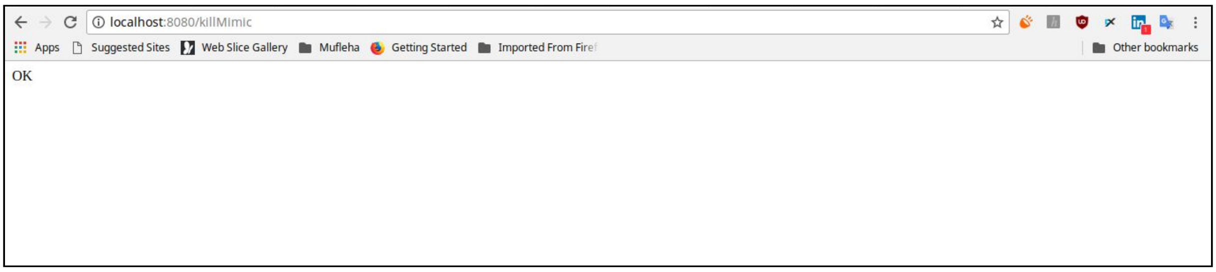
2. Clicking on the last request “hello” in the above example will show its response “3”



```
3
```

Closing the mimic service:

1. Close Mimic and save the learned responses by using the command: http://localhost:8080/killMimic



```
OK
```