
mbtest Documentation

Release 2.5.0

Simon Brunning

Jun 30, 2021

CONTENTS:

1	Guide	3
1.1	Use with Docker	3
1.2	Extra	4
1.3	TODO	4
2	API Reference	5
2.1	The <i>mbtest.server</i> module	5
2.2	The <i>mbtest.imposters.imposters</i> module	8
2.3	The <i>mbtest.imposters.stubs</i> module	10
2.4	The <i>mbtest.imposters.predicates</i> module	11
2.5	The <i>mbtest.imposters.responses</i> module	14
2.6	The <i>mbtest.imposters.behaviors.copy</i> module	17
2.7	The <i>mbtest.imposters.behaviors.lookup</i> module	17
2.8	The <i>mbtest.imposters.behaviors.using</i> module	18
2.9	The <i>mbtest.matchers</i> module	20
2.10	The <i>mbtest.imposters.base</i> module	23
3	Indices and tables	25
4	Installation	27
5	Usage	29
6	Indices and tables	31
	Python Module Index	33
	Index	35

Opinionated Python wrapper & utils for the [Mountebank](#) over the wire test double tool.
Includes [pytest](#) fixture and [PyHamcrest](#) matchers.

(Work in progress)

1.1 Use with Docker

If you want to use your own mountebank service instance ([Docker](#), for example) you have **no need to use npm** requirements.

```
docker run -p 2525:2525 -p IMPOSTER_PORT:IMPOSTER_PORT -d bbyars/mountebank
```

You can do like this in your `[conftest.py]`:

```
import pytest
from mbtest.server import MountebankServer

@pytest.fixture(scope="session")
def mock_server():
    return MountebankServer(port=2525, host="localhost")
```

Don't forget to open docker ports for mountebank (default 2525) and for each of its imposters.

```
from mbtest.imposters import Imposter, Predicate, Response, Stub

imposter = Imposter(
    Stub(
        Predicate(path="/test") & Predicate(query={}) & Predicate(method="GET"),
        Response(body="sausages")
    ),
    record_requests=True,
    port=IMPOSTER_PORT)

with mock_server(imposter) as ms:
    response = requests.get(f"{imposter.url}/test")
    # Check your request
    assert_that(imposter, had_request().with_path("/test").and_method("GET"))
```

If you don't specify a port for the Imposter it will be allocated randomly.

1.2 Extra

You can combine your Predicates with `&(and)`, `|(or)`.

1.3 TODO

- Basics
 - Server options
 - * Executing
 - * Existing server, e.g. docker
 - Running locally, against existing server (e.g. docker)
- Stubs, predicates, responses
 - And and or
 - Options
 - Injection
- Stubbing vs. Mocking
 - Assertions and matchers
- Proxies
 - Record/Playback
- SMTP

API REFERENCE

2.1 The *mbtest.server* module

`mbtest.server.mock_server(request, executable=PosixPath('node_modules/.bin/mb'), port=2525, timeout=5, debug=True, allow_injection=True, local_only=True, data_dir='.mbdb')`

Pytest fixture, making available a mock server, running one or more imposters, one for each domain being mocked.

Use in a pytest conftest.py fixture as follows:

```
@pytest.fixture(scope="session")
def mock_server(request):
    return server.mock_server(request)
```

Test will look like:

```
def test_an_imposter(mock_server):
    imposter = Imposter(Stub(Predicate(path='/test'),
                             Response(body='sausages')),
                        record_requests=True)

    with mock_server(imposter) as s:
        r = requests.get(f"{imposter.url}/test")

        assert_that(r, is_response().with_status_code(200).and_body("sausages"))
        assert_that(s, had_request(path='/test', method="GET"))
```

Parameters

- **request** (FixtureRequest) – Request for a fixture from a test or fixture function.
- **executable** (Union[str, Path]) – Alternate location for the Mountebank executable.
- **port** (int) – Server port.
- **timeout** (int) – specifies how long to wait for the Mountebank server to start.
- **debug** (bool) – Start the server in debug mode, which records all requests. This needs to be *True* for the `mbtest.matchers.had_request()` matcher to work.
- **allow_injection** (bool) – Allow JavaScript injection. If *True*, *local_only* should also be *True*, as per Mountebank security.
- **local_only** (bool) – Accept request only from localhost.

- **data_dir** (Optional[str]) – Persist all operations to disk, in this directory.

Return type `ExecutingMountebankServer`

Returns Mock server.

class `mbtest.server.MountebankServer`(*port*, *scheme*='http', *host*='localhost', *imposters_path*='imposters')

Allow addition of imposters to an already running Mountebank mock server.

Test will look like:

```
def test_an_imposter(mock_server):
    mb = MountebankServer(1234)
    imposter = Imposter(Stub(Predicate(path='/test'),
                             Response(body='sausages')),
                        record_requests=True)

    with mb(imposter):
        r = requests.get(f"{imposter.url}/test")

        assert_that(r, is_response().with_status_code(200).and_body("sausages"))
        assert_that(imposter, had_request(path='/test', method="GET"))
```

Imposters will be torn down when the *with* block is exited.

Parameters

- **port** (int) – Server port.
- **scheme** (str) – Server scheme, if not *http*.
- **host** (str) – Server host, if not *localhost*.
- **imposters_path** (str) – Imposters path, if not *imposters*.

add_imposters(*definition*)

Add imposters to Mountebank server.

Parameters definition (`Union[Imposter, Iterable[Imposter]]`) – One or more Imposters.

Return type `None`

add_impostor(*definition*)

Add single imposter to Mountebank server.

Parameters definition – One or more Imposters.

delete_imposters()

Delete all impostors from server.

Return type `None`

delete_impostor(*imposter*)

Delete impostor from server.

get_actual_requests()

Return type `Sequence[Request]`

property `server_url`: `furl.furl.furl`

Return type `furl`

query_all_imposters()

Yield all imposters running on the server, including those defined elsewhere.

Return type `Sequence[Imposter]`

import_running_imposters()

Replaces all running imposters with those defined on the server

Return type `None`

get_running_imposters()

Returns all imposters that the instance is aware of

Return type `Sequence[Imposter]`

```
class mbtest.server.ExecutingMountebankServer(executable=PosixPath('node_modules/.bin/mb'),
                                              port=2525, timeout=5, debug=True,
                                              allow_injection=True, local_only=True,
                                              data_dir='.mbdb')
```

A Mountebank mock server, running one or more imposters, one for each domain being mocked.

Test will look like:

```
def test_an_imposter(mock_server):
    mb = ExecutingMountebankServer()
    imposter = Imposter(Stub(Predicate(path='/test'),
                              Response(body='sausages')),
                        record_requests=True)

    with mb(imposter) as s:
        r = requests.get(f"{imposter.url}/test")

        assert_that(r, is_response().with_status_code(200).and_body("sausages"))
        assert_that(s, had_request(path='/test', method="GET"))

    mb.close()
```

The mountebank server will be started when this class is instantiated, and needs to be closed if it's not to be left running. Consider using the `mock_server()` pytest fixture, which will take care of this for you.

Parameters

- **executable** (`Union[str, Path]`) – Optional, alternate location for the Mountebank executable.
- **port** (`int`) – Server port.
- **timeout** (`int`) – How long to wait for the Mountebank server to start.
- **debug** (`bool`) – Start the server in debug mode, which records all requests. This needs to be `True` for the `mbtest.matchers.had_request()` matcher to work.
- **allow_injection** (`bool`) – Allow JavaScript injection. If `True`, `local_only` should also be `True`, as per Mountebank security.
- **local_only** (`bool`) – Accept request only from localhost.
- **data_dir** (`Optional[str]`) – Persist all operations to disk, in this directory.

running: `Set[int] = {}`

start_lock = `<unlocked _thread.lock object>`

`close()`

Return type `None`

exception `mbtest.server.MountebankException`

Exception using Mountebank server.

exception `mbtest.server.MountebankPortInUseException`

Mountebank server failed to start - port already in use.

exception `mbtest.server.MountebankTimeoutError`

Mountebank server failed to start in time.

2.2 The *mbtest.imposters.imposters* module

```
class mbtest.imposters.imposters.Imposter(stubs, port=None, protocol=<Protocol.HTTP: 'http'>,
                                           name=None, default_response=None, record_requests=True,
                                           mutual_auth=False, key=None, cert=None)
```

Represents a `Mountebank imposter`. Think of an imposter as a mock website, running a protocol, on a specific port. Required behaviors are specified using stubs.

Parameters

- **stubs** (`Union[Stub, Iterable[Stub]]`) – One or more Stubs.
- **port** (`Optional[int]`) – Port.
- **protocol** (`Protocol`) – Protocol to run on.
- **name** (`Optional[str]`) – Imposter name - useful for interactive exploration of imposters on `http://localhost:2525/imposters`
- **default_response** (`Optional[HttpResponse]`) – The default response to send if no predicate matches.
- **record_requests** (`bool`) – Record requests made against this imposter, so they can be asserted against later.
- **mutual_auth** (`bool`) – Server will request a client certificate.
- **key** (`Optional[str]`) – SSL server certificate.
- **cert** (`Optional[str]`) – SSL server certificate.

```
class Protocol(value)
```

Imposter `Protocol`.

```
HTTP = 'http'
```

```
HTTPS = 'https'
```

```
SMTP = 'smtp'
```

```
TCP = 'tcp'
```

```
property url: furl.furl.furl
```

Return type `furl`

```
as_structure()
```

Converted to a JSON serializable structure.

Return type `Any`

Returns Structure suitable for JSON serialisation.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type *Imposter*

Returns Converted object.

get_actual_requests()

Return type *Sequence[Request]*

attach(*host, port, server_url*)

Attach imposter to a running MB server.

Return type *None*

property attached: *bool*

Imposter is attached to a running MB server.

Return type *bool*

property configuration_url: *furl.furl.furl*

Return type *furl*

query_all_stubs()

Return all stubs running on the impostor, including those defined elsewhere.

Return type *List[Stub]*

playback()

Return type *List[Stub]*

add_stubs(*definition, index=None*)

Add one or more stubs to a running impostor.

Return type *None*

add_stub(*definition, index=None*)

Add a stub to a running impostor. Returns index of new stub.

Return type *int*

delete_stub(*index*)

Remove a stub from a running impostor.

Return type *Stub*

class `mbtest.imposters.imposters.Request`

static `from_json(json)`

Return type *Request*

class `mbtest.imposters.imposters.HttpRequest`(*method, path, query, headers, body, **kwargs*)

```
static from_json(json)
```

Return type [*HttpRequest*](#)

```
class mbtest.imposters.imposters.Address(address, name)
```

property address

Alias for field number 0

property name

Alias for field number 1

```
class mbtest.imposters.imposters.SentEmail(from_, to, cc, bcc, subject, text, **kwargs)
```

```
static from_json(json)
```

Return type [*SentEmail*](#)

```
mbtest.imposters.imposters.smtp_imposter(name='smtp', record_requests=True)
```

Canned SMTP server imposter.

Return type [*Imposter*](#)

2.3 The *mbtest.imposters.stubs* module

```
class mbtest.imposters.stubs.Stub(predicates=None, responses=None)
```

Represents a [Mountebank stub](#). Think of a stub as a behavior, triggered by a matching predicate.

Parameters

- **predicates** ([Union](#)[[BasePredicate](#), [Iterable](#)[[BasePredicate](#)], [None](#)]) – Trigger this stub if one of these predicates matches the request
- **responses** ([Union](#)[[BaseResponse](#), [Iterable](#)[[BaseResponse](#)], [None](#)]) – Use these response behaviors (in order)

as_structure()

Converted to a JSON serializable structure.

Return type [*Any*](#)

Returns Structure suitable for JSON serialisation.

```
classmethod from_structure(structure)
```

Converted from a JSON serializable structure.

Parameters **structure** ([Any](#)) – JSON structure to be converted.

Return type [*Stub*](#)

Returns Converted object.

```
class mbtest.imposters.stubs.AddStub(stub=None, index=None)
```

Represents a [Mountebank add stub request](#) <<http://www.mbtest.org/docs/api/overview#add-stub>>. To add new stab to an existing imposter.

Parameters

- **index** (`Optional[int]`) – The index in imposter stubs array. If you leave off the index field, the stub will be added to the end of the existing stubs array.
- **stub** (`Optional[Stub]`) – The stub that will be added to the existing stubs array

as_structure()

Converted to a JSON serializable structure.

Return type `Any`

Returns Structure suitable for JSON serialisation.

static from_structure(structure)

Converted from a JSON serializable structure.

Parameters **structure** (`Any`) – JSON structure to be converted.

Return type `AddStub`

Returns Converted object.

2.4 The *mbtest.imposters.predicates* module

class `mbtest.imposters.predicates.BasePredicate`

classmethod from_structure(structure)

Converted from a JSON serializable structure.

Parameters **structure** (`Any`) – JSON structure to be converted.

Return type `BasePredicate`

Returns Converted object.

class `mbtest.imposters.predicates.LogicallyCombinablePredicate`

class `mbtest.imposters.predicates.Predicate(path=None, method=None, query=None, body=None, headers=None, xpath=None, operator=<Operator.EQUALS: 'equals'>, case_sensitive=True)`

Represents a [Mountebank predicate](#). A predicate can be thought of as a trigger, which may or may not match a request.

Parameters

- **path** (`Union[str, furl, None]`) – URL path.
- **method** (`Optional[Method]`) – HTTP method.
- **query** (`Optional[Mapping[str, Union[str, int, bool]]]`) – Query arguments, keys and values.
- **body** (`Union[str, Any, None]`) – Body text. Can be a string, or a JSON serialisable data structure.
- **headers** (`Optional[Mapping[str, str]]`) – Headers, keys and values.
- **xpath** (`Optional[str]`) – xpath query
- **operator** (`Operator`) –
- **case_sensitive** (`bool`) –

exception InvalidPredicateOperator

class Method(*value*)

Predicate HTTP method.

DELETE = 'DELETE'

GET = 'GET'

HEAD = 'HEAD'

POST = 'POST'

PUT = 'PUT'

PATCH = 'PATCH'

class Operator(*value*)

Predicate operator.

EQUALS = 'equals'

DEEP_EQUALS = 'deepEquals'

CONTAINS = 'contains'

STARTS_WITH = 'startsWith'

ENDS_WITH = 'endsWith'

MATCHES = 'matches'

EXISTS = 'exists'

classmethod has_value(*name*)

Return type *bool*

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod from_structure(*structure*)

Converted from a JSON serializable structure.

Parameters **structure** (*Any*) – JSON structure to be converted.

Return type *Predicate*

Returns Converted object.

fields_from_structure(*inner*)

fields_as_structure()

class mbtest.imposters.predicates.AndPredicate(*left, right*)

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type *AndPredicate*

Returns Converted object.

class `mbtest.imposters.predicates.OrPredicate(left, right)`

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type *OrPredicate*

Returns Converted object.

class `mbtest.imposters.predicates.NotPredicate(inverted)`

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type *NotPredicate*

Returns Converted object.

class `mbtest.imposters.predicates.TcpPredicate(data)`

Represents a [Mountebank TCP predicate](#). A predicate can be thought of as a trigger, which may or may not match a request.

Parameters `data` (*str*) – Data to match the request.

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type *TcpPredicate*

Returns Converted object.

class `mbtest.imposters.predicates.InjectionPredicate`(*inject*)

Represents a [Mountebank injection predicate](#). A predicate can be thought of as a trigger, which may or may not match a request.

Injection requires Mountebank version 2.0 or higher.

Parameters `inject` (*str*) – JavaScript function to inject.

classmethod `from_structure`(*structure*)

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type [InjectionPredicate](#)

Returns Converted object.

2.5 The `mbtest.imposters.responses` module

class `mbtest.imposters.responses.BaseResponse`

classmethod `from_structure`(*structure*)

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type [BaseResponse](#)

Returns Converted object.

class `mbtest.imposters.responses.HttpResponse`(*body=""*, *status_code=200*, *headers=None*, *mode=None*)

Represents a [Mountebank HTTP response](#).

Parameters

- **body** ([Union](#)[*str*, *Any*]) – Body text for response. Can be a string, or a JSON serialisable data structure.
- **status_code** ([Union](#)[*int*, *str*]) – HTTP status code
- **headers** ([Optional](#)[[Mapping](#)[*str*, *str*]]) – Response HTTP headers
- **mode** ([Optional](#)[*Mode*]) – Mode - text or binary

property `body`: *str*

Return type *str*

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod `from_structure`(*inner*)

Converted from a JSON serializable structure.

Parameters `structure` – JSON structure to be converted.

Return type [HttpResponse](#)

Returns Converted object.

```
class mbtest.imposters.responses.Response(body="", status_code=200, wait=None, repeat=None,
                                         headers=None, mode=None, copy=None, decorate=None,
                                         lookup=None, shell_transform=None, *,
                                         http_response=None)
```

Represents a Mountebank 'is' response behavior.

Parameters

- **body** (`Union[str, Any]`) – Body text for response. Can be a string, or a JSON serialisable data structure.
- **status_code** (`Union[int, str]`) – HTTP status code
- **wait** (`Union[int, str, None]`) – Add latency, in ms.
- **repeat** (`Optional[int]`) – Repeat this many times before moving on to next response.
- **headers** (`Optional[Mapping[str, str]]`) – Response HTTP headers
- **mode** (`Optional[Mode]`) – Mode - text or binary
- **copy** (`Optional[Copy]`) – Copy behavior
- **decorate** (`Optional[str]`) – Decorate behavior.
- **lookup** (`Optional[Lookup]`) – Lookup behavior
- **shell_transform** (`Union[str, Iterable[str], None]`) – shellTransform behavior
- **http_response** (`Optional[HttpResponse]`) – HTTP Response Fields - use this **or** the body, status_code, headers and mode fields, not both.

```
class Mode(value)
```

An enumeration.

```
TEXT = 'text'
```

```
BINARY = 'binary'
```

```
as_structure()
```

Converted to a JSON serializable structure.

Return type `Any`

Returns Structure suitable for JSON serialisation.

```
classmethod from_structure(structure)
```

Converted from a JSON serializable structure.

Parameters **structure** (`Any`) – JSON structure to be converted.

Return type `Response`

Returns Converted object.

```
property body
```

```
property status_code
```

```
property headers
```

```
property mode
```

```
class mbtest.imposters.responses.TcpResponse(data)
```

```
as_structure()
```

Converted to a JSON serializable structure.

Return type [Any](#)

Returns Structure suitable for JSON serialisation.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` ([Any](#)) – JSON structure to be converted.

Return type [TcpResponse](#)

Returns Converted object.

class `mbtest.imposters.responses.Proxy`(*to*, *wait=None*, *inject_headers=None*, *mode=<Mode.ONCE: 'proxyOnce'>*, *predicate_generators=None*)

Represents a [Mountebank proxy](#).

Parameters `to` ([Union](#)[[furl](#), [str](#)]) – The origin server, to which the request should proxy.

class `Mode`(*value*)

Defines the replay behavior of the proxy.

`ONCE = 'proxyOnce'`

`ALWAYS = 'proxyAlways'`

`TRANSPARENT = 'proxyTransparent'`

as_structure()

Converted to a JSON serializable structure.

Return type [Any](#)

Returns Structure suitable for JSON serialisation.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` ([Any](#)) – JSON structure to be converted.

Return type [Proxy](#)

Returns Converted object.

class `mbtest.imposters.responses.PredicateGenerator`(*path=False*, *query=False*, *operator=<Operator.EQUALS: 'equals'>*, *case_sensitive=True*)

as_structure()

Converted to a JSON serializable structure.

Return type [Any](#)

Returns Structure suitable for JSON serialisation.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` ([Any](#)) – JSON structure to be converted.

Return type [PredicateGenerator](#)

Returns Converted object.

class `mbtest.imposters.responses.InjectionResponse`(*inject*)

Represents a [Mountebank injection response](#).

Injection requires Mountebank version 2.0 or higher.

Parameters `inject` (*str*) – JavaScript function to inject .

classmethod `from_structure`(*structure*)

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type [InjectionResponse](#)

Returns Converted object.

2.6 The `mbtest.imposters.behaviors.copy` module

class `mbtest.imposters.behaviors.copy.Copy`(*from_, into, using*)

Represents a [copy behavior](#).

Parameters

- **from** – The name of the request field to copy from, or, if the request field is an object, then an object specifying the path to the request field.
- **into** (*str*) – The token to replace in the response with the selected request value.
- **using** (*Using*) – The configuration needed to select values from the response.

as_structure()

Converted to a JSON serializable structure.

Return type [Any](#)

Returns Structure suitable for JSON serialisation.

classmethod `from_structure`(*structure*)

Converted from a JSON serializable structure.

Parameters `structure` (*Any*) – JSON structure to be converted.

Return type [Copy](#)

Returns Converted object.

2.7 The `mbtest.imposters.behaviors.lookup` module

class `mbtest.imposters.behaviors.lookup.Lookup`(*key, datasource_path, datasource_key_column, into*)

Represents a [lookup behavior](#).

Parameters

- **key** (*Key*) – How to select the key from the request.
- **datasource_path** (*Union[str, Path]*) – The path to the data source.
- **datasource_key_column** (*str*) – The header of the column to match against the key.
- **into** (*str*) – The token to replace in the response with the selected request value.

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod from_structure(structure)

Converted from a JSON serializable structure.

Parameters **structure** (*Any*) – JSON structure to be converted.

Return type *Lookup*

Returns Converted object.

class `mbtest.imposters.behaviors.lookup.Key`(*from_, using, index=0*)

The information on how to select the key from the request.

Parameters

- **from** – The name of the request field to copy from, or, if the request field is an object, then an object specifying the path to the request field.
- **using** (*Using*) – The configuration needed to select values from the response
- **index** (*int*) – Index of the item from the result array to be selected.

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod from_structure(structure)

Converted from a JSON serializable structure.

Parameters **structure** (*Any*) – JSON structure to be converted.

Return type *Key*

Returns Converted object.

2.8 The *mbtest.imposters.behaviors.using* module

class `mbtest.imposters.behaviors.using.Using`(*method, selector*)

How to select values from the response.

Parameters

- **method** (*Method*) – The method used to select the value(s) from the request.
- **selector** (*str*) – The selector used to select the value(s) from the request.

class `Method`(*value*)

An enumeration.

REGEX = 'regex'

XPATH = 'xpath'

JSONPATH = 'jsonpath'

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod from_structure(structure)

Converted from a JSON serializable structure.

Parameters **structure** (*Any*) – JSON structure to be converted.

Return type *Using*

Returns Converted object.

class `mbtest.imposters.behaviors.using.UsingRegex(selector, ignore_case=False, multiline=False)`

Select values from the response using a regular expression.

Parameters

- **selector** (*str*) – The selector used to select the value(s) from the request.
- **ignore_case** (*bool*) – Uses a case-insensitive regular expression
- **multiline** (*bool*) – Uses a multiline regular expression

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod from_structure(structure)

Converted from a JSON serializable structure.

Parameters **structure** (*Any*) – JSON structure to be converted.

Return type *UsingRegex*

Returns Converted object.

class `mbtest.imposters.behaviors.using.UsingXPath(selector, ns=None)`

Select values from the response using an xpath expression.

Parameters

- **selector** (*str*) – The selector used to select the value(s) from the request.
- **ns** (*Optional[Mapping[str, str]]*) – The ns object maps namespace aliases to URLs

as_structure()

Converted to a JSON serializable structure.

Return type *Any*

Returns Structure suitable for JSON serialisation.

classmethod from_structure(structure)

Converted from a JSON serializable structure.

Parameters **structure** (*Any*) – JSON structure to be converted.

Return type *UsingXPath*

Returns Converted object.

class `mbtest.imposters.behaviors.using.UsingJsonpath(selector)`

Select values from the response using a jsonpath expression.

Parameters `selector` (`str`) – The selector used to select the value(s) from the request.

classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure` – JSON structure to be converted.

Return type `UsingJsonpath`

Returns Converted object.

2.9 The *mbtest.matchers* module

```
mbtest.matchers.had_request(method=<hamcrest.core.core.isanything.IsAnything object>,
                             path=<hamcrest.core.core.isanything.IsAnything object>,
                             query=<hamcrest.core.core.isanything.IsAnything object>,
                             headers=<hamcrest.core.core.isanything.IsAnything object>,
                             body=<hamcrest.core.core.isanything.IsAnything object>,
                             times=<hamcrest.core.core.isanything.IsAnything object>)
```

Mountebank server has recorded call matching.

Build criteria with *with_* and *and_* methods:

```
assert_that(server, had_request().with_path("/test").and_method("GET"))
```

Available attributes as per parameters.

Parameters

- **method** (`Union[str, Matcher[str]]`) – Request’s method matched...
- **path** (`Union[furl, str, Matcher[Union[furl, str]]]`) – Request’s path matched...
- **query** (`Union[Mapping[str, str], Matcher[Mapping[str, str]]]`) – Request’s query matched...
- **headers** (`Union[Mapping[str, str], Matcher[Mapping[str, str]]]`) – Request’s headers matched...
- **body** (`Union[str, Matcher[str]]`) – Request’s body matched...
- **times** (`Union[int, Matcher[int]]`) – Request’s number of times called matched...

Return type `Matcher[Union[Imposter, MountebankServer]]`

```
class mbtest.matchers.HadRequest(method=<hamcrest.core.core.isanything.IsAnything object>,
                                 path=<hamcrest.core.core.isanything.IsAnything object>,
                                 query=<hamcrest.core.core.isanything.IsAnything object>,
                                 headers=<hamcrest.core.core.isanything.IsAnything object>,
                                 body=<hamcrest.core.core.isanything.IsAnything object>,
                                 times=<hamcrest.core.core.isanything.IsAnything object>)
```

Mountebank server has recorded call matching

Parameters

- **method** (`Union[str, Matcher[str]]`) – Request’s method matched...
- **path** (`Union[furl, str, Matcher[Union[furl, str]]]`) – Request’s path matched...

- **query** (`Union[Mapping[str, str], Matcher[Mapping[str, str]]]`) – Request’s query matched...
- **headers** (`Union[Mapping[str, str], Matcher[Mapping[str, str]]]`) – Request’s headers matched...
- **body** (`Union[str, Matcher[str]]`) – Request’s body matched...
- **times** (`Union[int, Matcher[int]]`) – Request’s number of times called matched...

describe_to(*description*)

Generates a description of the object.

The description may be part of a description of a larger object of which this is just a component, so it should be worded appropriately.

Parameters **description** (`Description`) – The description to be built or appended to.

Return type `None`

static append_matcher_description(*field_matcher, field_name, description*)

Return type `None`

describe_mismatch(*actual, description*)

Generates a description of why the matcher has not accepted the item.

The description will be part of a larger description of why a matching failed, so it should be concise.

This method assumes that `matches(item)` is `False`, but will not check this.

Parameters

- **item** – The item that the `Matcher` has rejected.
- **mismatch_description** – The description to be built or appended to.

Return type `None`

with_method(*method*)

and_method(*method*)

with_path(*path*)

and_path(*path*)

with_query(*query*)

and_query(*query*)

with_headers(*headers*)

and_headers(*headers*)

with_body(*body*)

and_body(*body*)

with_times(*times*)

and_times(*times*)

```
mbtest.matchers.email_sent(to=<hamcrest.core.core.isanything.IsAnything object>,
                           subject=<hamcrest.core.core.isanything.IsAnything object>,
                           body_text=<hamcrest.core.core.isanything.IsAnything object>)
```

Mountebank SMTP server was asked to sent email matching:

Parameters

- **to** (`Union[str, Matcher[str]]`) – Email’s to field matched...
- **subject** (`Union[str, Matcher[str]]`) – Email’s subject field matched...
- **body_text** (`Union[str, Matcher[str]]`) – Email’s body matched...

Return type `Matcher[Union[Imposter, MountebankServer]]`

```
class mbtest.matchers.EmailSent(to=<hamcrest.core.core.isanything.IsAnything object>,
                               subject=<hamcrest.core.core.isanything.IsAnything object>,
                               body_text=<hamcrest.core.core.isanything.IsAnything object>)
```

Mountebank SMTP server was asked to sent email matching:

Parameters

- **to** (`Union[str, Matcher[str]]`) – Email’s to field matched...
- **subject** (`Union[str, Matcher[str]]`) – Email’s subject field matched...
- **body_text** (`Union[str, Matcher[str]]`) – Email’s body matched...

describe_to(*description*)

Generates a description of the object.

The description may be part of a description of a larger object of which this is just a component, so it should be worded appropriately.

Parameters **description** (`Description`) – The description to be built or appended to.

Return type `None`

describe_mismatch(*actual*, *description*)

Generates a description of why the matcher has not accepted the item.

The description will be part of a larger description of why a matching failed, so it should be concise.

This method assumes that `matches(item)` is `False`, but will not check this.

Parameters

- **item** – The item that the `Matcher` has rejected.
- **mismatch_description** – The description to be built or appended to.

Return type `None`

static get_sent_email(*actual*)

Return type `Sequence[SentEmail]`

get_matching_emails(*sent_email*)

Return type `Sequence[SentEmail]`

2.10 The *mbtest.imposters.base* module

class `mbtest.imposters.base.JsonSerializable`

Object capable of being converted to a JSON serializable structure (using `as_structure()`) or from such a structure ((using `from_structure()`).

abstract `as_structure()`

Converted to a JSON serializable structure.

Return type `Any`

Returns Structure suitable for JSON serialisation.

abstract classmethod `from_structure(structure)`

Converted from a JSON serializable structure.

Parameters `structure (Any)` – JSON structure to be converted.

Return type `JsonSerializable`

Returns Converted object.

static `add_if_true(dictionary, key, value)`

Return type `None`

`set_if_in_dict(dictionary, key, name)`

Return type `None`

class `mbtest.imposters.base.Injecting(inject)`

as_structure()

Converted to a JSON serializable structure.

Return type `Any`

Returns Structure suitable for JSON serialisation.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

INSTALLATION

Install from [Pypi](#) as usual, using `pip` , `tox`, or `setup.py`.

Also requires [Mountebank](#) to have been installed:

```
$ npm install mountebank@2.4 --production
```


USAGE

A basic example:

```
import requests
from hamcrest import assert_that
from brunns.matchers.response import is_response
from mbtest.matchers import had_request
from mbtest.imposters import Imposter, Predicate, Response, Stub

def test_request_to_mock_server(mock_server):
    # Set up mock server with required behavior
    imposter = Imposter(Stub(Predicate(path="/test"),
                             Response(body="sausages")))

    with mock_server(imposter):
        # Make request to mock server - exercise code under test here
        response = requests.get(f"{imposter.url}/test")

        assert_that("We got the expected response",
                     response, is_response().with_status_code(200).and_body("sausages"))
        assert_that("The mock server recorded the request",
                     imposter, had_request().with_path("/test").and_method("GET"))
```

Needs a pytest fixture, most easily defined in `conftest.py`:

```
import pytest
from mbtest import server

@pytest.fixture(scope="session")
def mock_server(request):
    return server.mock_server(request)
```


INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

m

- `mbtest.imposters.base`, 23
- `mbtest.imposters.behaviors.copy`, 17
- `mbtest.imposters.behaviors.lookup`, 17
- `mbtest.imposters.behaviors.using`, 18
- `mbtest.imposters.imposters`, 8
- `mbtest.imposters.predicates`, 11
- `mbtest.imposters.responses`, 14
- `mbtest.imposters.stubs`, 10
- `mbtest.matchers`, 20
- `mbtest.server`, 5

INDEX

A

`add_if_true()` (*mbtest.imposters.base.JsonSerializable* static method), 23

`add_imposters()` (*mbtest.server.MountebankServer* method), 6

`add_impостor()` (*mbtest.server.MountebankServer* method), 6

`add_stub()` (*mbtest.imposters.imposters.Imposter* method), 9

`add_stubs()` (*mbtest.imposters.imposters.Imposter* method), 9

`Address` (class in *mbtest.imposters.imposters*), 10

`address` (*mbtest.imposters.imposters.Address* property), 10

`AddStub` (class in *mbtest.imposters.stubs*), 10

`ALWAYS` (*mbtest.imposters.responses.Proxy.Mode* attribute), 16

`and_body()` (*mbtest.matchers.HadRequest* method), 21

`and_headers()` (*mbtest.matchers.HadRequest* method), 21

`and_method()` (*mbtest.matchers.HadRequest* method), 21

`and_path()` (*mbtest.matchers.HadRequest* method), 21

`and_query()` (*mbtest.matchers.HadRequest* method), 21

`and_times()` (*mbtest.matchers.HadRequest* method), 21

`AndPredicate` (class in *mbtest.imposters.predicates*), 12

`append_matcher_description()` (*mbtest.matchers.HadRequest* static method), 21

`as_structure()` (*mbtest.imposters.base.Injecting* method), 23

`as_structure()` (*mbtest.imposters.base.JsonSerializable* method), 23

`as_structure()` (*mbtest.imposters.behaviors.copy.Copy* method), 17

`as_structure()` (*mbtest.imposters.behaviors.lookup.Key* method), 18

`as_structure()` (*mbtest.imposters.behaviors.lookup.Lookup* method), 17

`as_structure()` (*mbtest.imposters.behaviors.using.Using* method), 18

`as_structure()` (*mbtest.imposters.behaviors.using.UsingRegex* method), 19

`as_structure()` (*mbtest.imposters.behaviors.using.UsingXPath* method), 19

`as_structure()` (*mbtest.imposters.imposters.Imposter* method), 8

`as_structure()` (*mbtest.imposters.predicates.AndPredicate* method), 12

`as_structure()` (*mbtest.imposters.predicates.NotPredicate* method), 13

`as_structure()` (*mbtest.imposters.predicates.OrPredicate* method), 13

`as_structure()` (*mbtest.imposters.predicates.Predicate* method), 12

`as_structure()` (*mbtest.imposters.predicates.TcpPredicate* method), 13

`as_structure()` (*mbtest.imposters.responses.HttpResponse* method), 14

`as_structure()` (*mbtest.imposters.responses.PredicateGenerator* method), 16

`as_structure()` (*mbtest.imposters.responses.Proxy* method), 16

`as_structure()` (*mbtest.imposters.responses.Response* method), 15

`as_structure()` (*mbtest.imposters.responses.TcpResponse* method), 15

`as_structure()` (*mbtest.imposters.stubs.AddStub* method), 11

`as_structure()` (*mbtest.imposters.stubs.Stub* method), 10

`attach()` (*mbtest.imposters.imposters.Imposter* method), 9

`attached` (*mbtest.imposters.imposters.Imposter* property), 9

B

`BasePredicate` (class in *mbtest.imposters.predicates*), 11

`BaseResponse` (class in *mbtest.imposters.responses*), 14

`BINARY` (*mbtest.imposters.responses.Response.Mode* attribute), 15

`body` (*mbtest.imposters.responses.HttpResponse* property), 14

body (*mbtest.imposters.responses.Response* property), 15

C

close() (*mbtest.server.ExecutingMountebankServer* method), 7

configuration_url (*mbtest.imposters.imposters.Imposter* property), 9

CONTAINS (*mbtest.imposters.predicates.Predicate.Operator* attribute), 12

Copy (class in *mbtest.imposters.behaviors.copy*), 17

D

DEEP_EQUALS (*mbtest.imposters.predicates.Predicate.Operator* attribute), 12

DELETE (*mbtest.imposters.predicates.Predicate.Method* attribute), 12

delete_imposters() (*mbtest.server.MountebankServer* method), 6

delete_impostor() (*mbtest.server.MountebankServer* method), 6

delete_stub() (*mbtest.imposters.imposters.Imposter* method), 9

describe_mismatch() (*mbtest.matchers.EmailSent* method), 22

describe_mismatch() (*mbtest.matchers.HadRequest* method), 21

describe_to() (*mbtest.matchers.EmailSent* method), 22

describe_to() (*mbtest.matchers.HadRequest* method), 21

E

email_sent() (in module *mbtest.matchers*), 21

EmailSent (class in *mbtest.matchers*), 22

ENDS_WITH (*mbtest.imposters.predicates.Predicate.Operator* attribute), 12

EQUALS (*mbtest.imposters.predicates.Predicate.Operator* attribute), 12

ExecutingMountebankServer (class in *mbtest.server*), 7

EXISTS (*mbtest.imposters.predicates.Predicate.Operator* attribute), 12

F

fields_as_structure() (*mbtest.imposters.predicates.Predicate* method), 12

fields_from_structure() (*mbtest.imposters.predicates.Predicate* method), 12

from_json() (*mbtest.imposters.imposters.HttpRequest* static method), 9

from_json() (*mbtest.imposters.imposters.Request* static method), 9

from_json() (*mbtest.imposters.imposters.SentEmail* static method), 10

from_structure() (*mbtest.imposters.base.JsonSerializable* class method), 23

from_structure() (*mbtest.imposters.behaviors.copy.Copy* class method), 17

from_structure() (*mbtest.imposters.behaviors.lookup.Key* class method), 18

from_structure() (*mbtest.imposters.behaviors.lookup.Lookup* class method), 18

from_structure() (*mbtest.imposters.behaviors.using.Using* class method), 19

from_structure() (*mbtest.imposters.behaviors.using.UsingJsonpath* class method), 20

from_structure() (*mbtest.imposters.behaviors.using.UsingRegex* class method), 19

from_structure() (*mbtest.imposters.behaviors.using.UsingXpath* class method), 19

from_structure() (*mbtest.imposters.imposters.Imposter* class method), 9

from_structure() (*mbtest.imposters.predicates.AndPredicate* class method), 12

from_structure() (*mbtest.imposters.predicates.BasePredicate* class method), 11

from_structure() (*mbtest.imposters.predicates.InjectionPredicate* class method), 14

from_structure() (*mbtest.imposters.predicates.NotPredicate* class method), 13

from_structure() (*mbtest.imposters.predicates.OrPredicate* class method), 13

from_structure() (*mbtest.imposters.predicates.Predicate* class method), 12

from_structure() (*mbtest.imposters.predicates.TcpPredicate* class method), 13

from_structure() (*mbtest.imposters.responses.BaseResponse* class method), 14

from_structure() (*mbtest.imposters.responses.HttpResponse* class method), 14

from_structure() (*mbtest.imposters.responses.InjectionResponse* class method), 17

from_structure() (*mbtest.imposters.responses.PredicateGenerator* class method), 16

from_structure() (*mbtest.imposters.responses.Proxy* class method), 16

from_structure() (*mbtest.imposters.responses.Response* class method), 15

from_structure() (*mbtest.imposters.responses.TcpResponse* class method), 16

from_structure() (*mbtest.imposters.stubs.AddStub* static method), 11

from_structure() (*mbtest.imposters.stubs.Stub* class method), 10

G

GET (*mbtest.imposters.predicates.Predicate.Method* attribute), 12

get_actual_requests() (*mbtest.imposters.imposters.Imposter* method), 9

get_actual_requests() (*mbtest.server.MountebankServer* method), 6

get_matching_emails() (*mbtest.matchers.EmailSent* method), 22

get_running_imposters() (*mbtest.server.MountebankServer* method), 7

get_sent_email() (*mbtest.matchers.EmailSent* static method), 22

H

had_request() (in module *mbtest.matchers*), 20

HttpRequest (class in *mbtest.imposters.imposters*), 9

HttpResponse (class in *mbtest.imposters.responses*), 14

HTTPS (*mbtest.imposters.imposters.Imposter.Protocol* attribute), 8

HEAD (*mbtest.imposters.predicates.Predicate.Method* attribute), 12

headers (*mbtest.imposters.responses.Response* property), 15

HTTP (*mbtest.imposters.imposters.Imposter.Protocol* attribute), 8

HttpRequest (class in *mbtest.imposters.imposters*), 9

HttpResponse (class in *mbtest.imposters.responses*), 14

HTTPS (*mbtest.imposters.imposters.Imposter.Protocol* attribute), 8

I

import_running_imposters() (*mbtest.server.MountebankServer* method), 7

Imposter (class in *mbtest.imposters.imposters*), 8

Imposter.Protocol (class in *mbtest.imposters.imposters*), 8

Injecting (class in *mbtest.imposters.base*), 23

InjectionPredicate (class in *mbtest.imposters.predicates*), 13

InjectionResponse (class in *mbtest.imposters.responses*), 16

J

JSONPATH (*mbtest.imposters.behaviors.using.Using.Method* attribute), 18

JsonSerializable (class in *mbtest.imposters.base*), 23

K

Key (class in *mbtest.imposters.behaviors.lookup*), 18

L

LogicallyCombinablePredicate (class in *mbtest.imposters.predicates*), 11

Lookup (class in *mbtest.imposters.behaviors.lookup*), 17

M

MATCHES (*mbtest.imposters.predicates.Predicate.Operator* attribute), 12

mbtest.imposters.base module, 23

mbtest.imposters.behaviors.copy module, 17

mbtest.imposters.behaviors.lookup module, 17

mbtest.imposters.behaviors.using module, 18

mbtest.imposters.imposters module, 8

mbtest.imposters.predicates module, 11

mbtest.imposters.responses module, 14

mbtest.imposters.stubs module, 10

mbtest.matchers module, 20

mbtest.server module, 5

mock_server() (in module *mbtest.server*), 5

mode (*mbtest.imposters.responses.Response* property), 15

module

- mbtest.imposters.base*, 23
- mbtest.imposters.behaviors.copy*, 17
- mbtest.imposters.behaviors.lookup*, 17
- mbtest.imposters.behaviors.using*, 18
- mbtest.imposters.imposters*, 8
- mbtest.imposters.predicates*, 11
- mbtest.imposters.responses*, 14
- mbtest.imposters.stubs*, 10
- mbtest.matchers*, 20
- mbtest.server*, 5

MountebankException, 8

MountebankPortInUseException, 8

MountebankServer (class in *mbtest.server*), 6

MountebankTimeoutError, 8

N

name (*mbtest.imposters.imposters.Address* property), 10

NotPredicate (class in *mbtest.imposters.predicates*), 13

O

ONCE (*mbtest.imposters.responses.Proxy.Mode* attribute), 16

OrPredicate (class in *mbtest.imposters.predicates*), 13

P

PATCH (*mbtest.imposters.predicates.Predicate.Method* attribute), 12

playback() (*mbtest.imposters.imposters.Imposter* method), 9

POST (*mbtest.imposters.predicates.Predicate.Method* attribute), 12

Predicate (class in *mbtest.imposters.predicates*), 11

Predicate.InvalidPredicateOperator, 11

Predicate.Method (class *mbtest.imposters.predicates*), 12

Predicate.Operator (class *mbtest.imposters.predicates*), 12

PredicateGenerator (class *mbtest.imposters.responses*), 16

Proxy (class in *mbtest.imposters.responses*), 16

Proxy.Mode (class in *mbtest.imposters.responses*), 16

PUT (*mbtest.imposters.predicates.Predicate.Method* attribute), 12

Q

query_all_imposters() (*mbtest.server.MountebankServer* method), 6

query_all_stubs() (*mbtest.imposters.imposters.Imposter* method), 9

R

REGEX (*mbtest.imposters.behaviors.using.Using.Method* attribute), 18

Request (class in *mbtest.imposters.imposters*), 9

Response (class in *mbtest.imposters.responses*), 14

Response.Mode (class in *mbtest.imposters.responses*), 15

running (*mbtest.server.ExecutingMountebankServer* attribute), 7

S

SentEmail (class in *mbtest.imposters.imposters*), 10

server_url (*mbtest.server.MountebankServer* property), 6

set_if_in_dict() (*mbtest.imposters.base.JsonSerializable* method), 23

SMTP (*mbtest.imposters.imposters.Imposter.Protocol* attribute), 8

smtp_imposter() (in *mbtest.imposters.imposters* module), 10

start_lock (*mbtest.server.ExecutingMountebankServer* attribute), 7

STARTS_WITH (*mbtest.imposters.predicates.Predicate.Operator* attribute), 12

status_code (*mbtest.imposters.responses.Response* property), 15

Stub (class in *mbtest.imposters.stubs*), 10

T

TCP (*mbtest.imposters.imposters.Imposter.Protocol* attribute), 8

TcpPredicate (class in *mbtest.imposters.predicates*), 13

TcpResponse (class in *mbtest.imposters.responses*), 15

TEXT (*mbtest.imposters.responses.Response.Mode* attribute), 15

in TRANSPARENT (*mbtest.imposters.responses.Proxy.Mode* attribute), 16

U

url (*mbtest.imposters.imposters.Imposter* property), 8

Using (class in *mbtest.imposters.behaviors.using*), 18

Using.Method (class *mbtest.imposters.behaviors.using*), 18 in

UsingJsonpath (class *mbtest.imposters.behaviors.using*), 19 in

UsingRegex (class in *mbtest.imposters.behaviors.using*), 19

UsingXPath (class in *mbtest.imposters.behaviors.using*), 19

W

with_body() (*mbtest.matchers.HadRequest* method), 21

with_headers() (*mbtest.matchers.HadRequest* method), 21

with_method() (*mbtest.matchers.HadRequest* method), 21

with_path() (*mbtest.matchers.HadRequest* method), 21

with_query() (*mbtest.matchers.HadRequest* method), 21

with_times() (*mbtest.matchers.HadRequest* method), 21

X

XPATH (*mbtest.imposters.behaviors.using.Using.Method* attribute), 18