Postman CHEAT SHEET

About

Thank you for downloading this cheat sheet. This guide refers to the Postman App, not the Chrome extension. Please report any problems with it.

Postman Cheat Sheet is based on the official Postman documentation and own experience.

For a detailed documentation on each feature, check out https://www.getpostman.com/docs.

Variables

SETTING

Global

```
pm.globals.set("myVariable", MY_VALUE);
```

Environment

```
pm.environment.set("myVar", MY_VAL);
```

Data variables

Can only be set from a CSV or a JSON file.

GETTING

in scripts (pre-request, tests)

Global

```
pm.globals.get("myVariable");
```

Environment

```
pm.environment.get("myVariable");
```

Data variables

```
pm.iterationData.get("myVariable");
```

Any variable type

```
pm.variables.get("myVariable");
```

in the request builder

Syntax: {{myVariable}}

Request URL: http://{{domain}}/users/{{userId}}

Headers (key, value): X-{{myHeaderName}}

Body: { "id": "{{userId}}", "name": "John Doe"}

REMOVING

Global

```
pm.globals.unset("myVariable");
pm.globals.clear();
```

Environment

```
pm.environment.unset("myVariable");
pm.environment.clear();
```

DYNAMIC VARIABLES

Experimental. Can only be used in request builder. Only one value is generated per request.

{{$guid}} - global unique identifier. Example output: d96d398a-b655-4638-a6e5-40c0dc282fb7

{{$timestamp}} - current timestamp. Example output: 1507370977

{{$randomInt}} - random integer between 0 and 1000. Example output: 567

All dynamic variables can be combined with strings, in order to generate dynamic / unique data. Example body:

```
{"name": "John Doe", "email":
"john.doe.{{$timestamp}}@example.com"}
```

LOGGING / DEBUGGING VARIABLES

Open Postman Console and use console.log in your script. Example:

```
var myVar = pm.globals.get("myVar");
console.log(myVar);
```

Workflows

for Collection Runner / Newman

Set which will be the next request to be executed:

```
postman.setNextRequest("Request 2");
```

Stop executing requests / stop the collection run:

```
postman.setNextRequest(null);
```

External libraries

tv4 - JSON schema validator

```
var jsonSchema = { "items": {    "type": "boolean" }};
var jsonData1 = [true, false];
var jsonData2 = [true, "John"];

pm.test('Schema is valid', function() {
        pm.expect(tv4.validate(data1, schema))
                .to.be.true;
        pm.expect(tv4.validate(data2, schema))
                .to.be.true;
        console.log("Validation failed: ", tv4.error);
});
```

CryptoJS - cryptographic algorithms

Generate SHA-1 hash:

```
CryptoJS.SHA1("Message").toString();
```

Assertions

Note: You need to add any of the assertions inside a pm.test callback. Example

```
pm.test("Status code is 200", function () {
        pm.response.to.have.status(200);
});
```

Status code

Check if status code is 200:

```
pm.response.to.have.status(200);
```

Checking multiple status codes:

```
pm.expect(pm.response.code).to.be.oneOf([201,202]);
```

Response time

Response time below 9ms:

```
pm.expect(pm.response.responseTime).to.be.below(9);
```

Headers

Header exists:

```
pm.response.to.have.header("X-Cache");
```

Header has value:

```
pm.expect(pm.response.headers.get(X-Cache '))
        .to.eql(HIT');
```

Cookies

Cookie exists:

```
pm.expect(pm.cookies.has('sessionId')).to.be.true;
```

Cookie has value:

```
pm.expect(pm.cookies.get('sessionId')).to.eql('ad3s3');
```

Body

Any / HTML responses

Exact body match:

```
pm.response.to.have.body("OK");
pm.response.to.have.body("{"success"=true}");
```

Partial body match / body contains:

```
pm.expect(pm.response.text())
        .to.include("Order placed.");
```

JSON responses

Parse body (need for all assertions):

```
var jsonData = pm.response.json();
```

Simple value check:

```
pm.expect(jsonData.age).to.eql(30);
pm.expect(jsonData.name).to.eql("John");
```

Nested value check:

```
pm.expect(jsonData. products.0.category)
        .to.eql("Detergent");
```

XML responses

Convert XML body to JSON:

```
var jsonData = xml2Json(responseBody);
```

Note: see assertions for JSON responses.

Postman Sandbox

pm

This is the object containing the script that is running, can set variables and has access to a read-only copy of the request or response.

pm.globals

Handle global variables.
Available methods: has, get, set, unset, clear, toObject.

```
pm.globals.get("myVariable");
```

pm.environment

Handle environment variables.
Available methods: has, get, set, unset, clear, toObject.

```
pm.environment.set("myVar", "MY_VALUE");
```

pm.sendRequest

Send simple HTTP(S) GET requests from scripts:

```
pm.sendRequest('http://example.com',
function (err, res) {
    console.log(err ? err : res.json());
});
```

Full-option HTTP(S) request

```
const postRequest = {
    url: 'http://example.com',
    method: 'POST',
    header: 'X-Foo:foo',
    body: {
        mode: 'raw',
        raw: JSON.stringify({ name: 'John' })
    }
};
pm.sendRequest(postRequest, function (err, res) {
    console.log(err ? err : res.json());
});
```

Postman Echo

Helper API for testing requests. Read more at: https://docs.postman-echo.com.

Get Current UTC time in pre-request script:

```
pm.sendRequest('https://postman-
echo.com/time/now', function (err, res) {
    if (err) {      console.log(err);    }
    else {
        var currentTime = res.stream.toString();
        console.log(currentTime);
        pm.environment.set("currentTime", currentTime);
}});
```