Cheatography

Postman Script 2 Cheat Sheet

by Giang.nd2508 via cheatography.com/134606/cs/27869/

Response Example

```
let res = {
        " sto re": {
                " boo k": [
                        {
                                " cat ego ry": " -
ref ere nce ",
                                " aut hor ": " -
Nigel Rees",
                                " tit le": " -
Sayings of the Centur y",
                                " pri ce": 8.95
                        },
                        {
                                " cat ego ry": " -
fic tio n",
                                " aut hor ": " -
Evelyn Waugh",
                                " tit le": " Sword
of Honour ",
                                " pri ce": 12.99
                        },
                        {
                                " cat ego ry": " -
fic tio n",
                                " aut hor ": " -
Herman Melvil le",
                                " tit le": "Moby
Dick",
                                " isb n": " 0-5 -
53- 213 11- 3",
                                " pri ce": 8.99
                        },
                        {
                                " cat ego ry": " -
fic tio n",
                                " aut hor ": "J. R.
R. Tolkie n",
                                " tit le": "The
Lord of the Rings",
                                " isb n": " 0-3 -
95- 193 95- 8",
                                " pri ce": 22.99
                        }
```

Response Example (cont)

```
>       ],
      "bicycle": {
        "color": "red",
        "price": 19.95
      }
    }
};
```

Extract data

```
let firstAuthor = res.store.book[0].author; //
Nigel Rees
let lastAuthor = res.st ore.bo ok[ res.st ore.bo -
ok.l ength - 1].author; // J. R. R. Tolkien
let firstBook = res.st ore.bo ok[0];
/*{
    cat egory: 'refer ence',
    author: 'Nigel Rees',
    title: 'Sayings of the Century',
    price: 8.95
}
*/
let books = res.st ore.book;
let titleList = [];
books.f or Eac h(( ele ment) => {
       if (eleme nt.p rice < 15 && elemen t.c -
ategory === " fic tio n") {
                tit leL ist.pu sh( ele men t.t -
itle);
       }
});
// [ 'Sword of Honour', 'Moby Dick' ]
let bookList = _.filt er( books, (element) => {
        return elemen t.price < 15 && elemen t.c -
ategory === " fic tio n";
});
```

Extract data (cont)

```
> let titles = _.map(bookList, _.property("title"));
//or
let titles = _.map(bookList, "title");
console.log(titles);
// [ 'Sword of Honour', 'Moby Dick' ]
let isbn = [];
books.forEach((element) => {
   if (element.isbn !== undefined) {
      isbn.push(element.isbn);
   }
});
//[ '0-553-21311-3', '0-395-19395-8' ]
```

Working with time

```
var moment = require('moment');
now = moment();

// 07/05/2021
consol e.l og( now.fo rma t("D D/M M/Y YYY "));
// tăng 1 ngày và 1 giờ so với thời điểm hiện tại
next_1 day _1hour = now.add(1, " d").a dd(1, " -
h");
consol e.l og( nex t_1 day _1h our.fo rma t("D -
D/M M/YYYY HH:mm: ss"));
// giảm 1 ngày so với thời điểm hiện tại
yesterday = now.su btr act(1, " day s");
consol e.l og( yes ter day.fo rma t("D D/M M/Y -
YYY "));
//mili second
consol e.l og( mom ent ().v al ueO f());
//second
consol e.l og( mom ent ().u ni x());
```

Branching

```
// Call request A
postma n.s etN ext Req ues t("A ");
// Stop
postma n.s etN ext Req ues t(n ull);
```

Assertion

```
pm.test("name of the test", () => {
      //code to make assertion
});
//skip test
pm.tes t.s kip ("name of the test", () => {
      //code to make assertion
});
```

Get request information

```
request.name
//request A
reques t.url
//http s:/ /po stm an- ech o.c om/post
reques t.m ethod
//POST
reques t.h eaders
/*
{
    con ten t-type: " app lic ati on/ jso n",
    use r-a gent: " Pos tma nRu nti me/ 7.2 8.4 ",
    accept: /*",
    host: " pos tma n-e cho.co m",
   ...
}
```

Get request information (cont)

```
> */
request.headers['content-type']
//application/json
request.headers.cookie
//sails.sid=s%3A__IlvuQ-_d3y8a2MCRW_ZmNWay5zb_OC.sMfNnx
//Get body in json
pm.request.body.raw
/*
{
   phone: "0931831823",
   code: "402839ouroiewr"
}
*/
//Get body in urlencoded
pm.request.body.urlencoded
//a=1&b=2
//Get body in form-data
pm.request.body.formdata.get("name")
```

Chaijs

expect(object1).keyword(object2)

Chain Keyword: to, be, been, is, that, which, and, has, have, with, at, of, same, but, does, still, also

Support Keyword: **.not**, .deep, .nested, .own, .ordered, .any, .all

Action Keyword: **.a(type[, msg])** .include(val[, msg]) .ok **.true** .false .null .undefined .nan .exist .empty .equal(val[, msg]) **.eql(obj[, msg])** .above(n[, msg]) .least(n[, msg]) .below(n[, msg]) .most(n[, msg] .....

Example

```
pm.expect(pm.response.code).eql(200);
pm.exp ect (re s.s tor e.b ook [0].au tho r).e ql -
("Ni gel  Re es");
let fi rst Aut hor  =  res.st ore.bo ok[ 0].a -
uthor;
pm.exp ect (fi rst Aut hor ).t o.e ql( " Nig el  -
Ree s");
pm.exp ect (re s.s tor e.b ook.le ngt h).a bo -
ve(3);;
let te stQ uan tit y =  re s.s tor e.b ook.le ngt -
h > 3;
pm.exp ect (te stQ uan tit y).t rue;
const expect = requir e("c hai " ).e xpect;
let obj = {
        " cat ego ry": " ref ere nce ",
        " aut hor ": " Nigel Rees",
        " tit le": " Sayings of the Centur y",
        " pri ce": 8.95
}
pm.tes t("test first book", () =>{
       let firstBook = res.st ore.bo ok[0];
        exp ect (fi rst Boo k).e ql (obj);
});
let books = res.st ore.book;
let titles = _.map( books, _.prop ert y("t itl -
e"));
/*
[
        " Sayings of the Centur y",
        " Sword of Honour ",
        "Moby Dick",
        "The Lord of the Rings"
]
*/
expect (ti tle s).i nc lud e("The Lord of the
Rings");
```

Example (cont)

```
> titles.forEach((e) => expect(e).eql("The Lord of the Rings"));
titles.forEach((e) => expect(e).string("o"));
```

Check schema

```
//response
{
    " id": 1,
    " fir stN ame ": " Ver non ",
    " las tNa me": " Har per ",
    " ema il": " ege sta s.r hon cus.Pr oin @ma -
ssa Qui squ epo rtt ito r.o rg",
    " pro gra mme ": " Fin ancial Analys is",
    " cou rse s": [
        " Acc oun tin g",
        " Sta tis tic s"
   ]
}
//Test
const schema = {
    " typ e": " obj ect ",
    " pro per tie s": {
        " id": {"ty pe": " num ber "},
        " fir stN ame ": {"ty pe": " str ing "},
        " las tNa me": {"ty pe": " str ing "},
        " ema il": {"ty pe": " str ing "},
        " pro gra mme ": {"ty pe": " str ing "},
        " cou rse s": {"ty pe": " arr ay"}
   },
// Field nào là bắt buộc thì liệt kê ở phần require
    " req uir e": ["id ", " cou rse s"]
};
```

Check schema (cont)

```
> pm.test("Validate schema", () => {
   pm.response.to.have.jsonSchema(schema);
});
```

Newman

```
const newman = require("newman");
const moment = requir e("m ome nt- tim ezo ne");
let date_time = moment ().t z( " Asi a/H o_C hi_ -
Min h").f or mat ("DD -MM -YY YY_ HH- mm- ss");
let fileName = `repor t-c las s1- ${d ate _ti me}`;
newman.run(
       {
                col lec tion: requir e("./ co -
lle cti ons /cl ass 1.p ost man _co lle cti on.j -
so n"),
                env iro nment: requir e("./ en -
v/D eve lop men t.p ost man _en vir onm ent.js -
on"),
                rep orters: ["cl i", " htm lex -
tra "],
                rep orter: {
                        htm lextra: {
                                export: `./tes -
t/r epo rt/ ${f ile Nam e}.h tml`
                        }
                }
       },
        (err, summary) => {
               if (err) throw err;
                con sol e.l og( sum mar y.r un.f -
ai lures);
       }
);
```

Get response information

```
pm.response.code
//200
pm.res pon se.s tatus
//OK
pm.res pon se.r es pon seTime
//295
pm.res pon se.r es pon seSize
//633
//Get body in string
pm.res pon se.t ext()
//or
respon seBody
// {"ar gs": {}, " dat a":{ " pho ne": " 093 -
183 " ,"co de": " 402 8ew r"}} "
//Get body in json
pm.res pon se.j son()
//or
JSON.p ars e(r esp ons eBody)
/*
{
    " arg s": {},
    " dat a": {
        " pho ne": " 093 183 182 3",
        " cod e": " 402 839 our oie wr"
   }
}
*/
//Get response headers
respon seH eaders
/*
```

Get response information (cont)

```
> {
  "Date": "Wed, 13 Jul 2022 02:00:26 GMT",
  "Content-Type": "application/json; charset=utf-8",
  "Content-Length": "588",
  "Connection": "keep-alive",
  "ETag": "W/\"24c-5jASVJWFOfv/weD+EO/h0FSWRLk\"",
  "Vary": "Accept-Encoding",
  "set-cookie": "sails.sid=s%3ATHifsul4oQ2kju1Pf2KjQ; Path=/;
HttpOnly"
}
*/
```