Response Example

```
let res = {
    "store": {
        "book": [
            {
                "category": "reference",
                "author": "Nigel Rees",
                "title": "Sayings of the Century",
                "price": 8.95
            },
            {
                "category": "fiction",
                "author": "Evelyn Waugh",
                "title": "Sword of Honour",
                "price": 12.99
            },
            {
                "category": "fiction",
                "author": "Herman Melville",
                "title": "Moby Dick",
                "isbn": "0-553-21311-3",
                "price": 8.99
            },
            {
                "category": "fiction",
                "author": "J. R. R. Tolkien",
                "title": "The Lord of the Rings",
                "isbn": "0-395-19395-8",
                "price": 22.99
            }
        ],
        "bicycle": {
            "color": "red",
            "price": 19.95
        }
    }
};
```

Extract data

```
let firstAuthor = res.store.book[0].author; //
Nigel Rees
let lastAuthor = res.store.book[res.store.book.l-
ength - 1].author; // J. R. R. Tolkien
let firstBook = res.store.book[0];
/*{
  category: 'reference',
  author: 'Nigel Rees',
  title: 'Sayings of the Century',
  price: 8.95
}
*/
let books = res.store.book;
let titleList = [];
books.forEach((element) => {
    if (element.price < 15 && element.category ===
"fiction") {
        titleList.push(element.title);
    }
});
// [ 'Sword of Honour', 'Moby Dick' ]
let bookList = _.filter(books, (element) => {
    return element.price < 15 && element.category
=== "fiction";
});
let titles = _.map(bookList, _.property("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' ]
```

Cheatography

Postman Script 2 Cheat Sheet

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

Working with time

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

// 07/05/2021
console.log(now.format("DD/MM/YYYY"));
// tăng 1 ngày và 1 giờ so với thời điểm hiện tại
next_1day_1hour = now.add(1, "d").add(1, "h");
console.log(next_1day_1hour.format("DD/MM/YYYY
HH:mm:ss"));
// giảm 1 ngày so với thời điểm hiện tại
yesterday = now.subtract(1, "days");
console.log(yesterday.format("DD/MM/YYYY"));
//milisecond
console.log(moment().valueOf());
//second
console.log(moment().unix());
```

Branching

```
// Call request A
postman.setNextRequest("A");
// Stop
postman.setNextRequest(null);
```

Assertion

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

Get request information

```
request.name
//request A
request.url
//https://postman-echo.com/post
request.method
//POST
request.headers
```

Get request information (cont)

```
/*
{
  content-type: "application/json",
  user-agent: "PostmanRuntime/7.28.4",
  accept: "/",
  host: "postman-echo.com",
  ...
}
*/
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
```

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.expect(res.store.book[0].author).eql("Nigel Re-
es");
let firstAuthor = res.store.book[0].author;
pm.expect(firstAuthor).to.eql("Nigel Rees");
pm.expect(res.store.book.length).above(3);;
let testQuantity = res.store.book.length > 3;
pm.expect(testQuantity).true;
const expect = require("chai").expect;
let obj = {
    "category": "reference",
    "author": "Nigel Rees",
    "title": "Sayings of the Century",
    "price": 8.95
}
pm.test("test first book", () =>{
    let firstBook = res.store.book[0];
    expect(firstBook).eql(obj);
});
let books = res.store.book;
let titles = _.map(books, _.property("title"));
/*
[
    "Sayings of the Century",
    "Sword of Honour",
    "Moby Dick",
    "The Lord of the Rings"
]
*/
expect(titles).include("The Lord of the Rings");
titles.forEach((e) => expect(e).eql("The Lord of
the Rings"));
titles.forEach((e) => expect(e).string("o"));
```

Check schema

```
//response
{
  "id": 1,
  "firstName": "Vernon",
  "lastName": "Harper",
  "email": "egestas.rhoncus.Proin@massaQuisquepo-
rttitor.org",
  "programme": "Financial Analysis",
  "courses": [
    "Accounting",
    "Statistics"
  ]
}
//Test
const schema = {
  "type": "object",
  "properties": {
    "id": {"type": "number"},
    "firstName": {"type": "string"},
    "lastName": {"type": "string"},
    "email": {"type": "string"},
    "programme": {"type": "string"},
    "courses": {"type": "array"}
  },
// Field nào là bắt buộc thì liệt kê ở phần
require
  "require": ["id", "courses"]
};

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

Newman

```
const newman = require("newman");
const moment = require("moment-timezone");
let date_time = moment().tz("Asia/Ho_Chi_Minh").f-
ormat("DD-mm-YYYY_HH-mm-ss");
let fileName = `report-class1-${date_time}`;
newman.run(
    {
```

Newman (cont)

```
        collection: require("./collections/class-1.postman_collection.json"),
        environment: require("./env/Development.p-ostman_environment.json"),
        reporters: ["cli", "htmlextra"],
        reporter: {
            htmlextra: {
                export: `./test/report/${fileNam-e}.html`
            }
        }
    },
    (err, summary) => {
        if (err) throw err;
        console.log(summary.run.failures);
    }
);
```

Get response information

```
pm.response.code
//200
pm.response.status
//OK
pm.response.responseTime
//295
pm.response.responseSize
//633
//Get body in string
pm.response.text()
//or
responseBody
// {"args":{},"data":{"phone":"093183","code":"-4028ewr"}}"
//Get body in json
pm.response.json()
//or
JSON.parse(responseBody)
/*
{
  "args": {},
  "data": {
```

Get response information (cont)

```
    "phone": "0931831823",
    "code": "402839ouroiewr"
  }
}
*/
//Get response headers
responseHeaders
/*
{
  "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%3ATHifsul4oQ2kju1P-f2KjQ; Path=/; HttpOnly"
}
*/
```