

website of fake API

<https://jsonplaceholder.typicode.com>

website of json formatter

<https://jsonformatter.org/>

PRODUCTS

Type of API : GET

URL: <https://jsonplaceholder.typicode.com/posts>

Test Script for Get Posts

```
pm.test("Status code is 200", () => {
```

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

```
});
```

```
pm.test("Response has correct id, title, and body", () => {
```

```
  const response = pm.response.json();
```

```
  pm.expect(response).to.be.an("array");
```

```
  response.forEach((item) => {
```

```
    pm.expect(item.id).to.exist;
```

```
    pm.expect(item.id).to.be.a("number");
```

```
    pm.expect(item.title).to.exist;
```

```
    pm.expect(item.title).to.be.a("string");
```

```
    pm.expect(item.body).to.exist;
```

```
    pm.expect(item.body).to.be.a("string");
```

```
  });
```



Result of test script ;

Test Results

200 OK • 62 ms • 8.01 KB

Filter Results

PASSED

Status code is 200

PASSED

Response has correct id, title, and body

Type of API : GET

URL: <https://jsonplaceholder.typicode.com/posts/1/comments>

Test script for GET posts/comments

```
pm.test("Status code is 200", function () {
```

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

```
});
```

```
pm.test("All comments have correct postId, id, name, email, and body", function () {
```

```
  const response = pm.response.json();
```

```
  pm.expect(response).to.be.an('array').that.is.not.empty;
```

```
  response.forEach(comment => {
```

```
    pm.expect(comment).to.have.property('postId').that.is.a('number').and.equal(1);
```

```
    pm.expect(comment).to.have.property('id').that.is.a('number');
```

```
    pm.expect(comment).to.have.property('name').that.is.a('string');
```

```
    pm.expect(comment).to.have.property('email').that.is.a('string');
```

```
    pm.expect(comment).to.have.property('body').that.is.a('string');
```

```
  });
```

```
});
```

Result of test script

Filter Results ▾

PASSED Status code is 200

PASSED All comments have correct postId, id, name, email, and body

Type of API: GET

URL: <https://jsonplaceholder.typicode.com/comments?postId=1>

test script for Get comments?postId=1

```
// Test for status code
```

```
pm.test("Status code is 200", function () {
```

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

```
});
```

```
// Test for response time
```

```
pm.test("Response time is less than 300ms", function () {
```

```
  pm.expect(pm.response.responseTime).to.be.below(300);
```

```
});
```

```
// Test for response body properties
```

```
pm.test("Response body has 'postId', 'id', 'name', 'email', and 'body' properties", function () {
```

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

```
  pm.expect(jsonData).to.be.an('array').that.is.not.empty;
```

```
  pm.expect(jsonData[0]).to.have.property('postId');
```

```
  pm.expect(jsonData[0]).to.have.property('id');
```

```
  pm.expect(jsonData[0]).to.have.property('name');
```

```
  pm.expect(jsonData[0]).to.have.property('email');
```

```
  pm.expect(jsonData[0]).to.have.property('body');
```

```
});
```

Result of test script:

Test Results

↺

200 OK

•

186 ms

•

1.74 KB

•

🌐

|

📄 Save Response

⋮

Filter Results

↺

PASSED

Status code is 200

PASSED

Response time is less than 300ms

PASSED

Response body has 'postId', 'id', 'name', 'email', and 'body' properties

Type of API : POST

URL: <https://jsonplaceholder.typicode.com/comments>

Test script for Post comments

```
// Test for status code
```

```
pm.test("Status code is 201", function () {
```

```
    pm.response.to.have.status(201);
```

```
});
```

```
// Test for response time
```

```
pm.test("Response time is less than 500ms", function () {
```

```
    pm.expect(pm.response.responseTime).to.be.below(500);
```

```
});
```

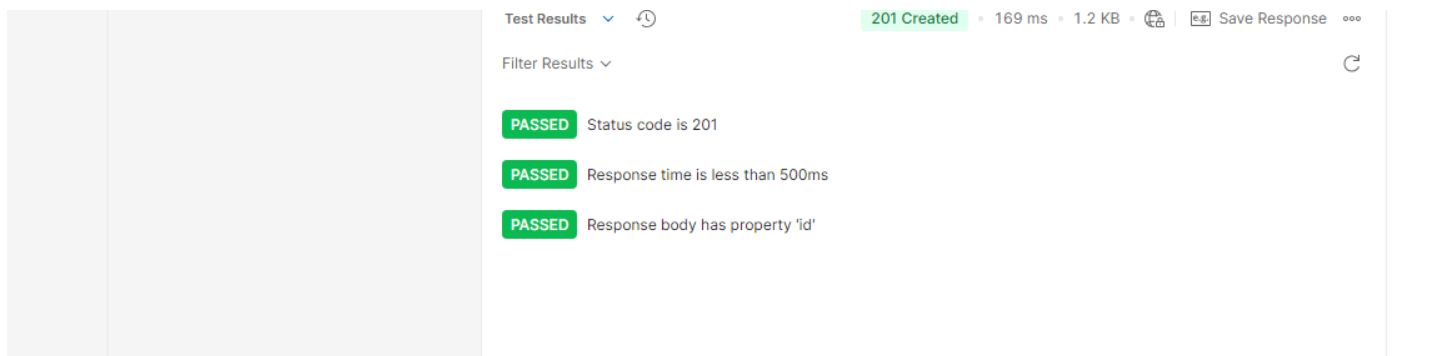
```
// Test for response body properties
```

```
pm.test("Response body has property 'id'", function () {
```

```
    pm.expect(pm.response.json()).to.have.property('id');
```

```
});
```

Result of test script:



The screenshot displays the Postman Test Results interface. At the top, it shows 'Test Results' with a dropdown arrow and a refresh icon. To the right, a green status bar indicates '201 Created' with details: '169 ms', '1.2 KB', a globe icon, and a 'Save Response' button with a three-dot menu. Below this, a 'Filter Results' dropdown is visible. The test results are listed in a table with three rows, each showing a green 'PASSED' status and the test description:

Status	Test Description
PASSED	Status code is 201
PASSED	Response time is less than 500ms
PASSED	Response body has property 'id'

Type of API : PUT

URL: <https://jsonplaceholder.typicode.com/posts/1>

Test script for PUT posts

```
// Ensure the status code is 200
```

```
pm.test("Status code is 200", function () {
```

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

```
});
```

```
// Validate the response time
```

```
pm.test("Response time is less than 500ms", function () {
```

```
  pm.expect(pm.response.responseTime).to.be.below(500);
```

```
});
```

```
// Validate the response body
```

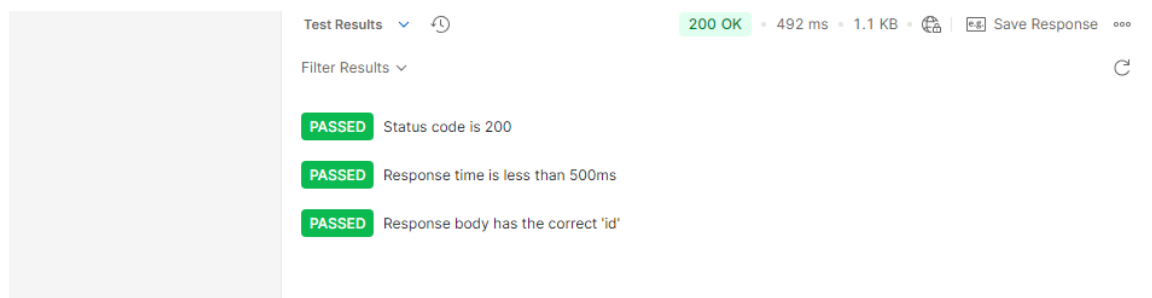
```
pm.test("Response body has the correct 'id'", function () {
```

```
  const responseBody = pm.response.json();
```

```
  pm.expect(responseBody.id).to.eql(1);
```

```
});
```

Result of test script:



The screenshot displays the Postman Test Results interface. On the left, there is a large grey rectangular area representing the test results. On the right, the 'Test Results' section is expanded, showing a summary of the test run. The summary indicates a status of '200 OK', a response time of '492 ms', and a response size of '1.1 KB'. Below the summary, there are three individual test results, each marked as 'PASSED' in a green box. The tests are: 'Status code is 200', 'Response time is less than 500ms', and 'Response body has the correct 'id''.

Test Results
200 OK • 492 ms • 1.1 KB
Filter Results
PASSED Status code is 200
PASSED Response time is less than 500ms
PASSED Response body has the correct 'id'

Type of API : DELETE

URL: <https://jsonplaceholder.typicode.com/posts/1>

Test script of Delete posts

```
pm.test("Status code is 200", function () {
```

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

```
});
```

```
pm.test("Response time is less than 500ms", function () {
```

```
  pm.expect(pm.response.responseTime).to.be.below(500);
```

```
});
```

Result of test script:

