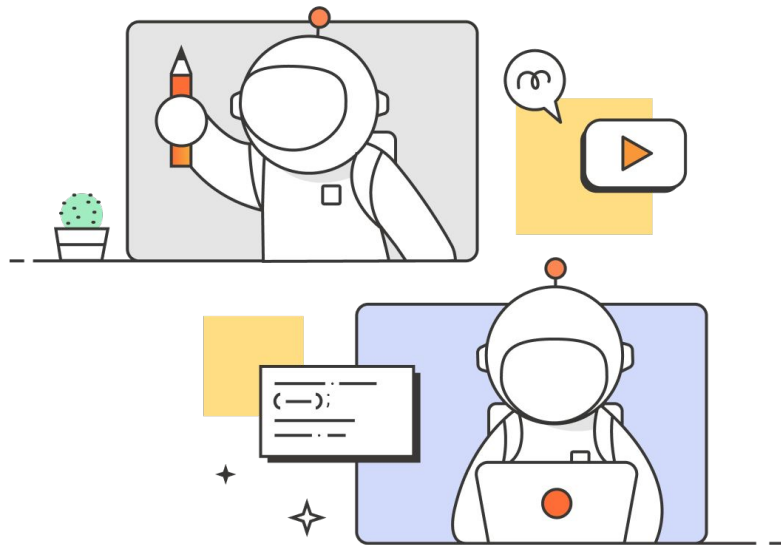




How to Run Tests in Postman



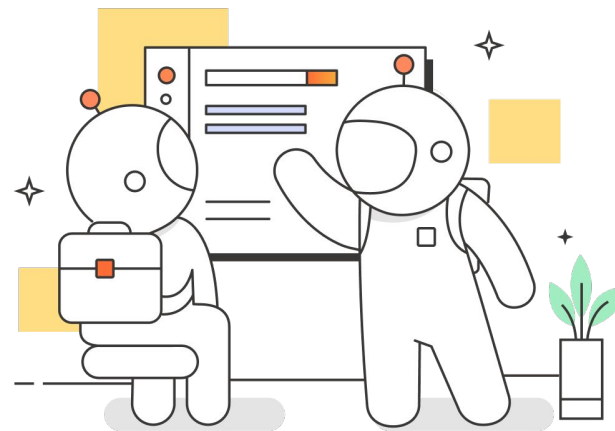


Topics Covering

- 1 Basics of Testing APIs

- 2 Automated Testing

- 3 Overview of Advanced Testing Options





Testing APIs: Sending Requests

A core component of Postman that allows you to test an API to see how it works and what it will return

Postman API / Postman Echo GET

GET

Params • Auth Headers (8) Body Pre-req. Tests Settings [Cookies](#)

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	id	123			
<input checked="" type="checkbox"/>	type	VIP			

Body Cookies (1) Headers (7) Test Results Security 200 OK 465 ms 853 B [Save Response](#)

Pretty Raw Preview Visualize JSON

```
1 {
2   "args": {
3     "id": "123",
4     "type": "VIP"
5   },
6   "headers": {
7     "x-forwarded-proto": "https",
8     "x-forwarded-port": "443",
9     "host": "postman-echo.com",
10    "x-amzn-trace-id": "Root=1-632a3c57-3c0f306e028d5632546fa347",
11    "user-agent": "PostmanRuntime/7.29.2",
12    "accept": "*/*",
13    "cache-control": "no-cache",
14    "postman-token": "315c86c5-b435-48f1-978c-a77a17a21766",
15    "accept-encoding": "gzip, deflate, br"
```



Testing APIs: Pre-Requests and Test Scripts

Pre-request scripts

- Javascript code that executes before the request is sent
- [Use Cases](#)

Tests

- Javascript code that executes *after* you receive the response
- Help to confirm your API is working as expected
- Add `.log()` to help with [debugging](#)

The screenshot shows the Postman interface for a GET request to `https://www.postman-echo.com/get`. The 'Tests' tab is active, displaying the following JavaScript code:

```
1 // use the `pm.*` API to write your test
2 // the `pm.test()` method accepts two parameters
3 // the first parameter is the name of your test (be descriptive!)
4 // the second parameter is a function. If any assertions within the
  function fails, the test fails
5
6 pm.test("Name of the first test", function() {
7   // make an assertion
8   // if your assertion throws an error, this test will fail
9   pm.response.to.have.status(404);
10 });
11
12 pm.test("Name of the second test", function() {
13   // make an assertion
14   // if your assertion throws an error, this test will fail
15   pm.response.to.have.status(200);
16 });
```

On the right side, there is a sidebar with a note: "Test scripts are written in JavaScript, and are run after the response is received. [Learn more about tests scripts](#)". Below this are 'SNIPPETS' such as "Get an environment variable", "Get a global variable", "Get a variable", "Get a collection variable", "Set an environment variable", "Set a global variable", "Set a collection variable", and "Clear an environment variable".

At the bottom, the 'Test Results (1/2)' section shows:

- FAIL** Name of the first test | AssertionError: expected response to have status code 404 but got 200
- PASS** Name of the second test



Testing APIs: Variables and Environments

Variables allow for storage and reuse of values in Postman.

- **Scopes:** Global, Collection, Environment, Data, Local
- **Types:** Default, Secret

The screenshot shows a GET request in Postman with the URL `https://postman-echo.com/get?var={{my_variable}}`. A dropdown menu is open, showing the variable `my_variable` selected. The menu also displays the variable's initial value (`Hello`), current value (`Hello`), and scope (`Global`). Below the dropdown, a table shows the variable `var` with a checked checkbox and the placeholder `{{my_variable}}`.

KEY	VALUE
<input checked="" type="checkbox"/> var	{{my_variable}}

Environments: group sets of variables together, manage access for collaboration

The screenshot shows the 'Testing Env' environment configuration in Postman. The table lists variables with their types, initial values, and current values.

VARIABLE	TYPE	INITIAL VALUE	CURRENT VALUE	Persist All	Reset All
<input checked="" type="checkbox"/> base_url	default	https://postman-echo.com/	https://postman-echo.com/		
<input checked="" type="checkbox"/> environment_id	default	1850107-47d15285-1259-4103-a8...	1850107-47d15285-1259-4103-a8c7-46554a3...		
<input checked="" type="checkbox"/> auth_key	secret	*****	*****		
<input checked="" type="checkbox"/> api_secret	secret	*****	*****		
Add a new variable					

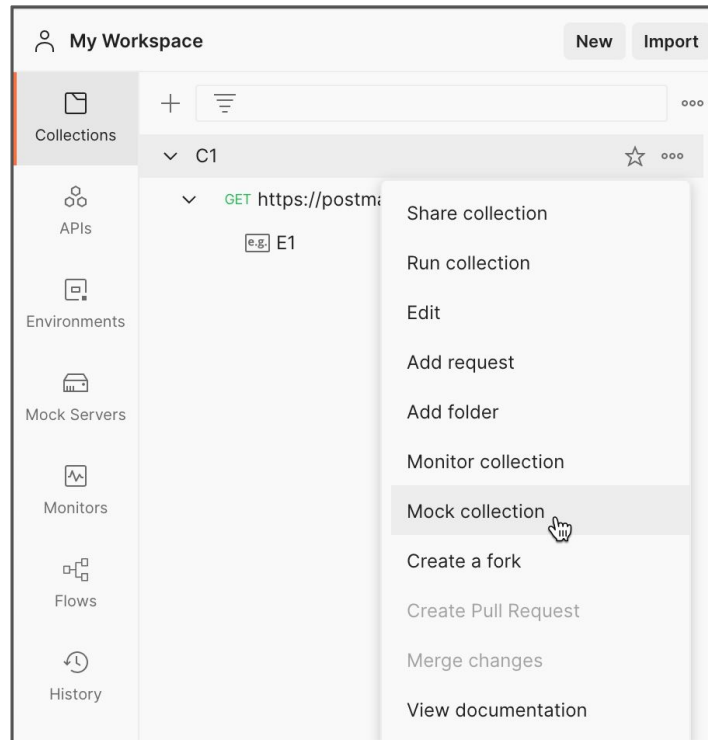
Demo





Testing APIs: Mock servers

- Reduce time debugging during API lifecycle
- Allows you to see what an API would deliver in prod by mocking the request and example responses
- Allows for collaboration between backend and front end teams before API is in prod
- Test without using sensitive data using dummy data





Automating Test: Collection Runner

- **Collection Runner** allows you to run API requests in a collection in sequence
- This enables you to:
 - Automate your testing
 - Schedule runs with **monitors**
 - Perform smoke tests

Widget Collection - Run results

Run on Today, 10:04:59 - [View Run History](#)

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	none	1	321ms	4	14 ms

All Tests Passed (2) Failed (2) Skipped (0) [View Summary](#)

Iteration 1

- GET** GET widget example.com / GET widget 200 OK 16 ms 1.005 KB
Pass Status code is 200
- GET** GET widgets example.com / GET widgets 200 OK 7 ms 1.005 KB
Fail Status code name has string | AssertionError: expected response to have status reason 'Created' but got 'OK'
- POST** POST widget example.com / POST widget 200 OK 17 ms 1.557 KB
Pass Response time is less than 200ms
- DELETE** DELETE widget example.com / DELETE widget 405 Method Not Allowed 15 ms 226 B
Fail Status code is 200 | AssertionError: expected response to have status code 200 but got 405



Automating Test: CI Integrations w/ Newman and Postman CLI

- **Newman**: programmatically run collections and [integrate with your CI](#)
- **Postman CLI**
 - secure CLI
 - run collections
 - check API specifications against your Governance and Security rules

```
ln ~/desktop/newman-run
+ newman run postman-echo.json
```

The image shows a terminal window with a dark background. The title bar at the top right says "Terminal". The prompt is "ln ~/desktop/newman-run" and the command being executed is "newman run postman-echo.json".



Automating Tests: Monitors

Monitors: automatically run your tests for endpoints and APIs at regular intervals to validate:

- HTTP Response Codes
- Latency





Testing APIs: Additional Resources

Consumer Driven Contract Testing

- https://www.youtube.com/watch?v=Ynfr-y_1WRs&t=1035s
- <https://www.postman.com/events/intergalactic/a-pi-contract-testing/>

More Testing Content

- [Shift Left in Postman](#)
- [Test examples in Postman](#) - A public workspace containing many more examples of crafting tests within Postman.
- [Continuous testing with Postman](#) - A previously aired webinar covering other advanced testing workflows.

The screenshot shows the Postman interface for a GET request to `https://www.postman-echo.com/get`. The `Tests` tab is active, displaying two test scripts:

```
1 // use the `pm.*` API to write your test
2 // the `pm.test()` method accepts two parameters
3 // the first parameter is the name of your test (be descriptive!)
4 // the second parameter is a function. If any assertions within the
  function fails, the test fails
5
6 pm.test("Name of the first test", function() {
7   // make an assertion
8   // if your assertion throws an error, this test will fail
9   pm.response.to.have.status(404);
10 });
11
12 pm.test("Name of the second test", function() {
13   // make an assertion
14   // if your assertion throws an error, this test will fail
15   pm.response.to.have.status(200);
16 });
```

Below the code, the `Test Results (1/2)` section shows the following:

Result	Test Name	Message
FAIL	Name of the first test	AssertionError: expected response to have status code 404 but got 200
PASS	Name of the second test	

The status bar at the bottom indicates a 200 OK response with 90 ms latency and 718 B body size.



Thank for participating in our Webinar!

If you have any follow-up questions, please book some time with the Product Advocacy team on our [Calendly](#) or email us at productadvocacy@postman.com.

