

Consumindo e criando
testes para sua **API** like a
boss com **Postman**





Quem sou eu?



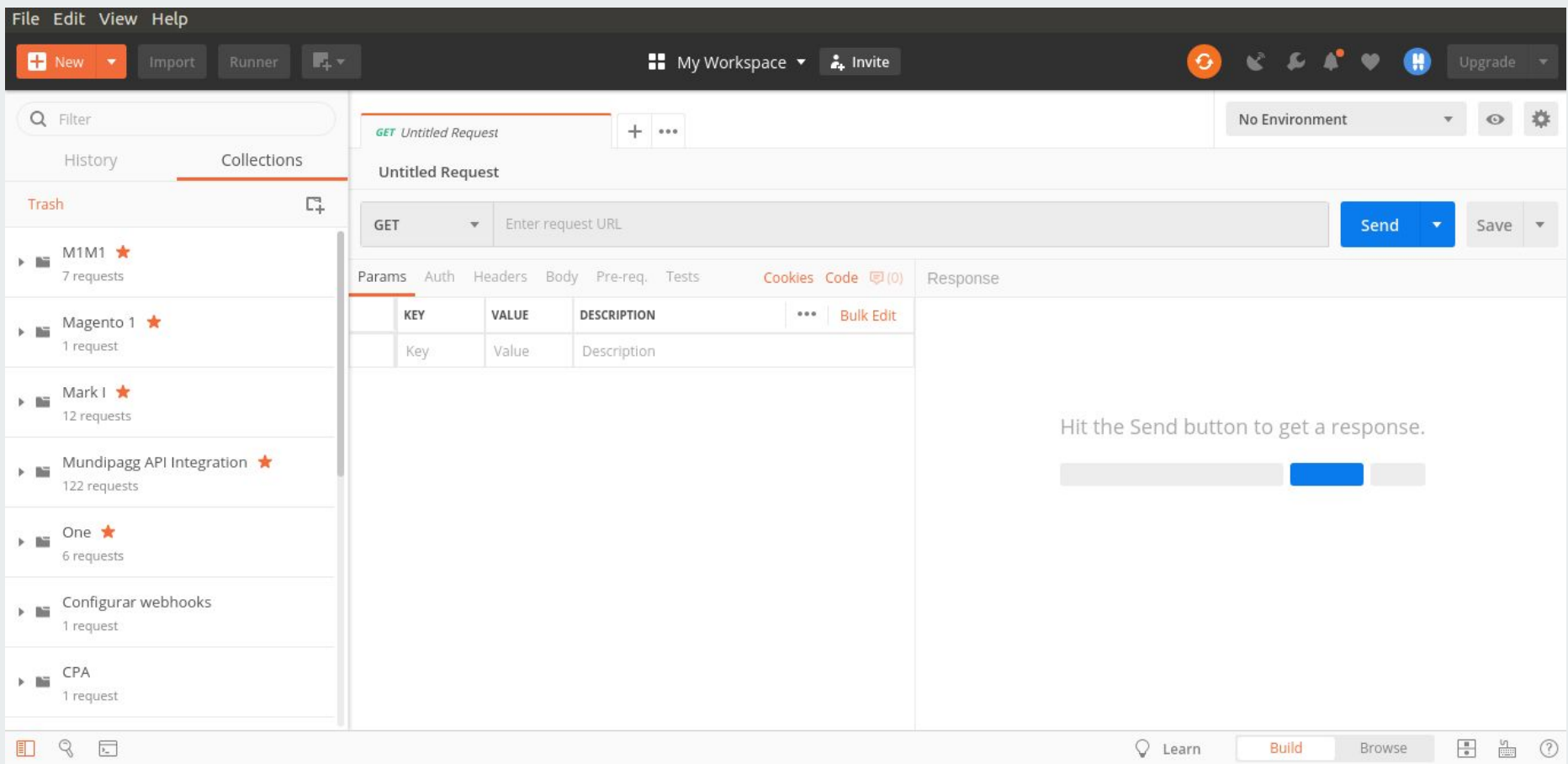
- Cientista da computação e programador PHP há 8 anos.
- Especializado em tecnologias voltadas para e-commerces
- Desenvolvedor na Mundipagg

Agenda



- Interface e requests básicos
- Code snippets
- API do Postman
- Environments
- Testes vs Pre-requests
- Testes
- Pre-requests
- O que dá pra fazer?
- Escrevendo testes
- Runner
- Criando fluxos de execução
- Postman monitor
- Mock Server
- Gerando docs a partir de collections
- Newman

Interface



POSTUntitled Request

+

...

No Environment

Untitled Request

POST

Enter request URL

Send

Save

ParamsAuthHeaders (1)BodyPre-req. TestsCookiesCode (0)

Verbos

none

form-data

x-www-form-urlencoded

raw

binary

JSON (application/json)

Beautify

1

Response

Hit the Send button to get a response.

Body

Response

Interface

Status code

Status: 200 OK

Time: 2015 ms

Size: 403 B

Resultado testes

Save

Download

Body

Cookies

Headers (10)

Test Results (3/4)

Pretty

Raw

Preview

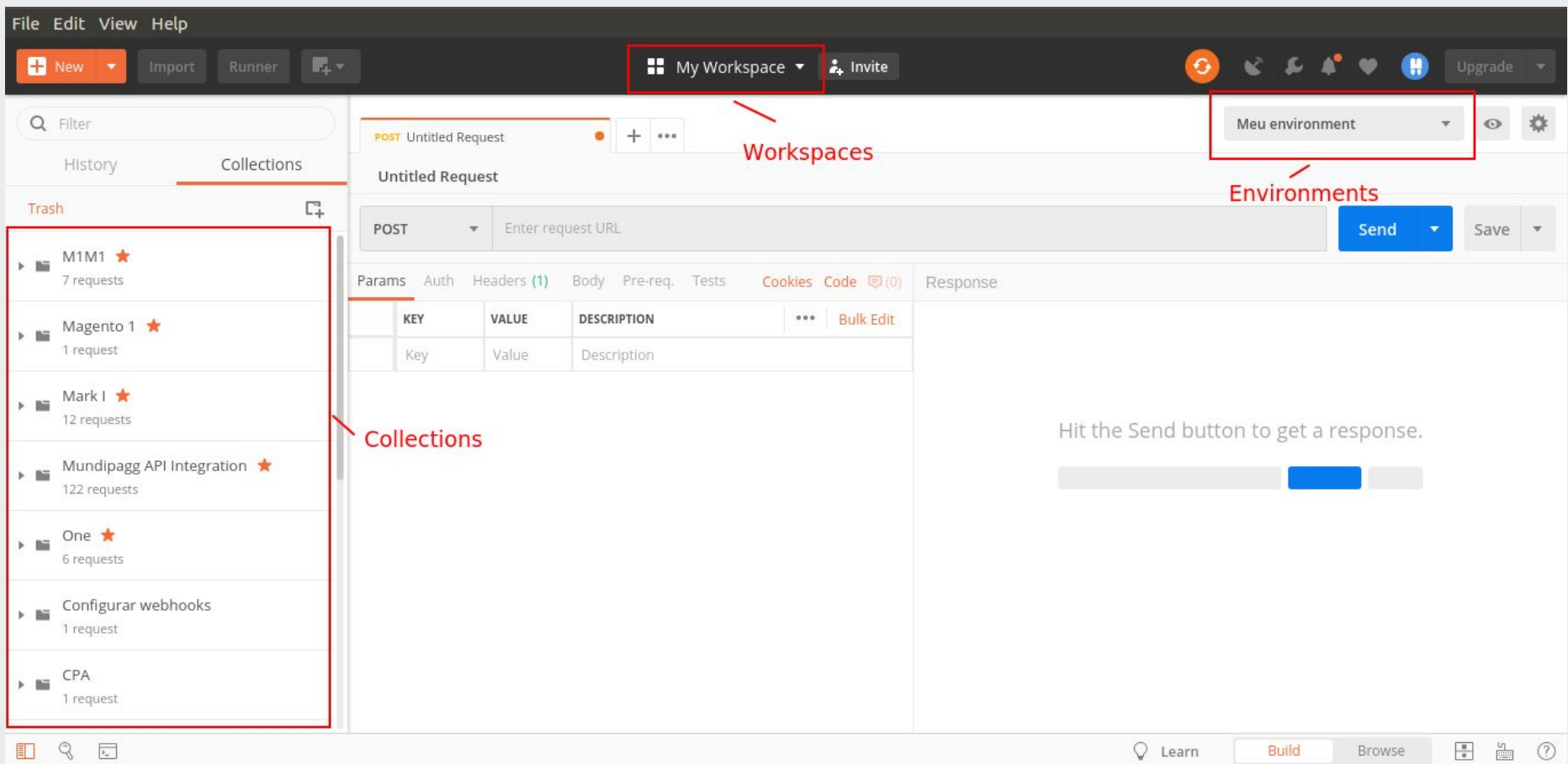
JSON



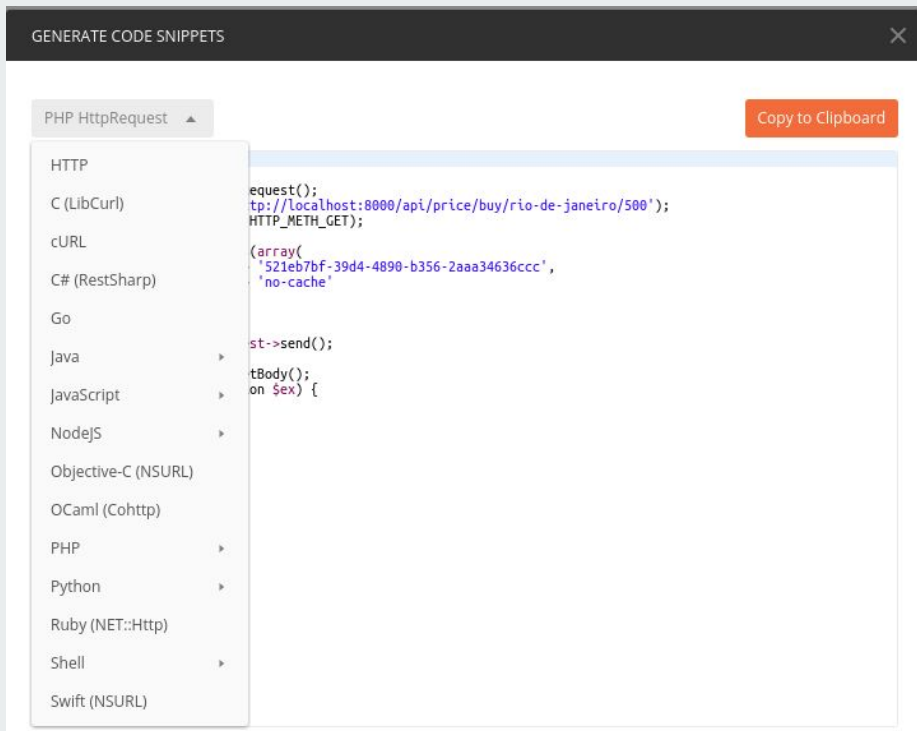
```
1 {  
2   "amout_to_buy": 500,  
3   "original_price": 4.31,  
4   "lower_price": "4.29",  
5   "date_time": "19-10-2019 21:50:07"  
6 }
```

Body da response

Interface




Code snippets



Da aba de requisições para sua linguagem favorita.


Api do Postman


“A API do Postman permite que você acesse programaticamente os dados armazenados na conta do Postman com facilidade.”




Postman API

An API to access your Postman data

 Backup

 Notifications

 0 Existing API keys

Details

Existing API Keys

The Postman API has several endpoints to help you integrate Postman even more deeply with your development toolchain. You can add new [collections](#), update existing collections, update [environments](#), add and run [monitors](#) directly through the API. This API allows you to programmatically access data stored in your Postman account with ease. Get started with the API by clicking the Run in Postman button at the top of the [Postman API documentation](#) page and use the Postman app to send requests.

Available Integrations

Get Postman API Key

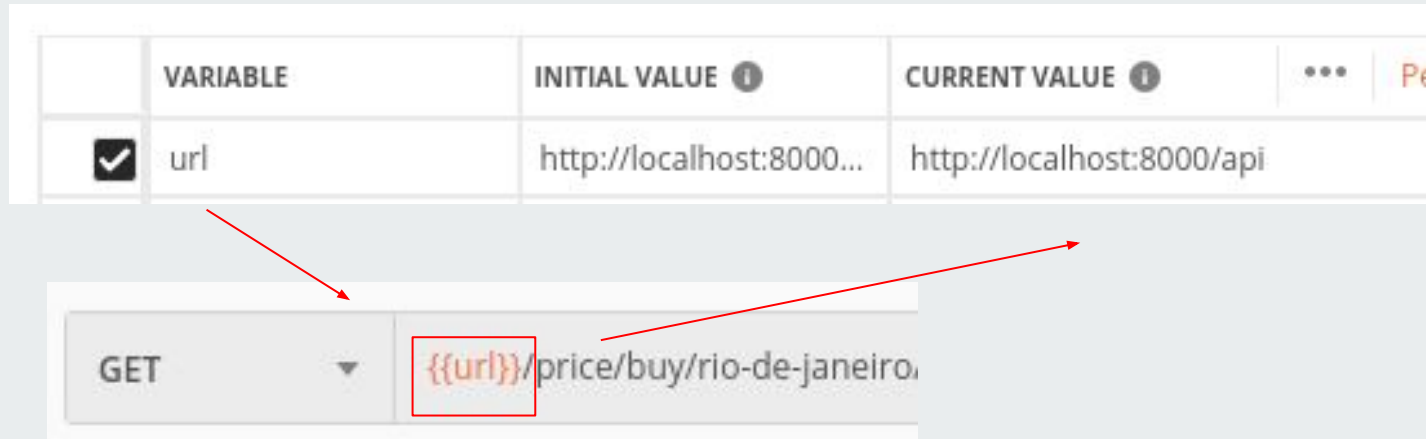
Use the Postman API key to authorize your requests to the Postman API

+ Get API Key

Environments

Permite definir e obter valores em variáveis que podem ser utilizadas em todos os seus requests dentro de uma collection.

Por ex:



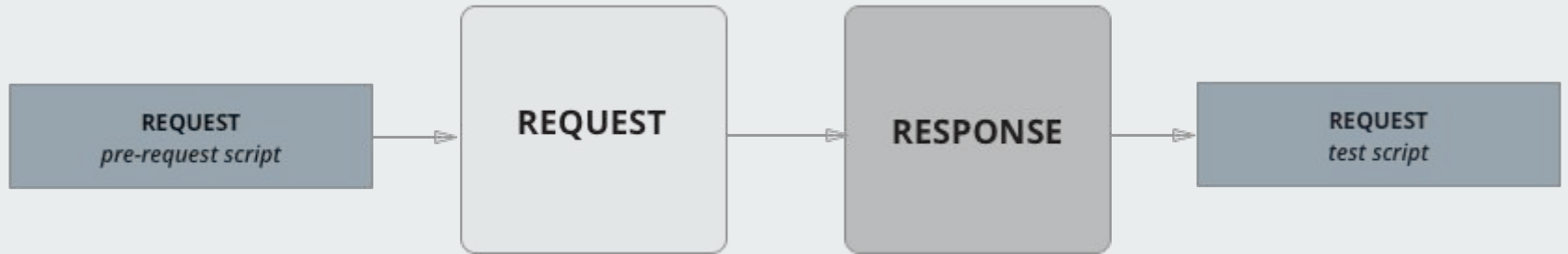
The image shows a screenshot of the Postman interface. At the top, there is a table representing environment variables. The table has four columns: a checkbox, 'VARIABLE', 'INITIAL VALUE', and 'CURRENT VALUE'. The first row shows a checked checkbox, the variable name 'url', and its values 'http://localhost:8000...' and 'http://localhost:8000/api'. Below this table, a red arrow points from the 'url' variable to a request URL field. The request URL field shows 'GET' as the method and the URL 'http://localhost:8000/api/price/buy/rio-de-janeiro...'. A red box highlights the variable reference '{{url}}' in the URL, and another red arrow points from the 'url' variable in the table above to this box.

	VARIABLE	INITIAL VALUE ⓘ	CURRENT VALUE ⓘ	...	Pe
<input checked="" type="checkbox"/>	url	http://localhost:8000...	http://localhost:8000/api		

GET ▼ {{url}}/price/buy/rio-de-janeiro...

Testes vs Pre-requests

A grande diferença está no momento em que são executados. Antes ou depois da request



Testes

Scripts escritos em javascript que são usados para validar a resposta de um request

The screenshot displays the Postman interface with the 'Tests' tab selected. The left pane contains three JavaScript test scripts. The right pane shows the test results, all of which passed.

Test Scripts:

```
1 pm.test("response is ok", function () {  
2   pm.response.to.have.status(200);  
3   pm.response.to.not.be.error;  
4 });  
5  
6 pm.test("Response should be okay to process", function () {  
7   var jsonData = pm.response.json();  
8   amount = pm.environment.get("amount");  
9   amount_to_buy = jsonData.amount_to_buy;  
10  
11   pm.expect(amount_to_buy).to.eql(amount);  
12  
13 });  
14  
15 pm.test("Fill current amount", function () {  
16   var jsonData = pm.response.json();  
17
```

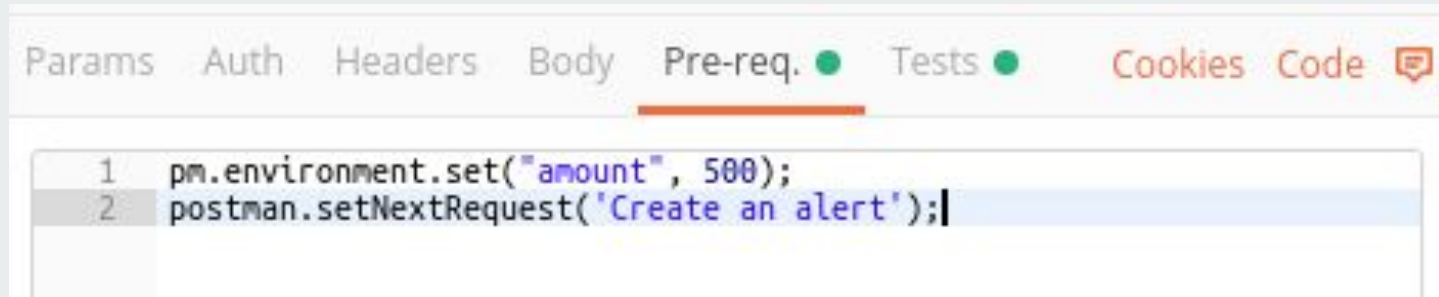
Test Results:

Status	Time	Size
200 OK	2827 ms	403 B

Test Results (4/4)
response is ok
Response should be okay to process
Fill current amount

Pre-requests

- São escritos da mesma maneira que os testes, exceto pela ausência do `pm.response`
- Muito usados para preencher variáveis com timestamp e valores randômicos



The screenshot shows the Postman interface with the 'Pre-req.' tab selected. The code editor contains two lines of JavaScript code:

```
1 pm.environment.set("amount", 500);  
2 postman.setNextRequest('Create an alert');
```

O que dá pra fazer?



- Verificar status code (2xx, 4xx, 5xx, etc)
- Deu 200, mas o teste falhou
- Verificar tempo de resposta
- Pegar/setar valor de uma variável de environment
- Verificar conteúdo retornado no body da response
- Validar header da response
- Enviar uma request via teste

Escrevendo testes



Tipos

- `pm.test`
- `pm.environment`
- `pm.globals`
- `pm.variables`
- `pm.expect`
- `pm.response`
- `pm.sendRequest`

Escrevendo testes

```
1 pm.test("response is ok", function () {
2     pm.response.to.have.status(200);
3     pm.response.to.not.be.error;
4 });
5
6 pm.test("Response should be okay to process", function () {
7     var jsonData = pm.response.json();
8     amount = pm.environment.get("amount");
9     amount_to_buy = jsonData.amout_to_buy;
10
11     pm.expect(amount_to_buy).to.eql(amount);
12
13 });
14
15 pm.test("Fill current amount", function () {
16     var jsonData = pm.response.json();
17
18     price = jsonData.lower_price;
19     pm.environment.set("current_price", price);
20 });
21
22 pm.test("Response time is less than 200ms", function () {
23     pm.expect(pm.response.responseTime).to.be.below(200);
24 });
25
```


Escrevendo testes

```
1  pm.test("response is ok", function () {  
2      pm.response.to.have.status(200);  
3      pm.response.to.not.be.error;  
4  });  
5  
6  pm.test("Response should be okay to process", function () {  
7      var jsonData = pm.response.json();  
8      amount = pm.environment.get("amount");  
9      amount_to_buy = jsonData.amout_to_buy;  
10  
11      pm.expect(amount_to_buy).to.eql(amount);  
12  
13  });  
14
```

Escrevendo testes

```
15 ▾ pm.test("Fill current amount", function () {  
16     var jsonData = pm.response.json();  
17  
18     price = jsonData.lower_price;  
19     pm.environment.set("current_price", price);  
20 });  
21  
22 ▾ pm.test("Response time is less than 200ms", function () {  
23     pm.expect(pm.response.responseTime).to.be.below(200);  
24 });  
25
```

Collection runner

Collection Runner

Quanto tá o dólar ▾

Run In Command Line

Docs

Choose a collection or folder

🔍 Search for a collection or folder

◀ Quanto tá o dólar?

GET Get value by city

GET Get all alerts

POST Create an alert

GET Get alert by ID

PUT Update alert

Environment

Qtod Local ▾

Iterations

1

Delay

0

ms

Log Responses

For all requests ▾

ⓘ

Data

Select File

Recent Runs

🔍 Type to Filter

Import Test Run

■	Quanto tá o dólar? Qtod Local	PASSED Today, 11:23 pm
■	Quanto tá o dólar? Qtod Local	1 FAILED Today, 11:23 pm
■	Quanto tá o dolar? Qtod Local	3 FAILED Yesterday, 9:33 pm
■	Quanto tá o dolar? Qtod Local	PASSED Yesterday, 8:31 pm
■	Quanto tá o dolar? Qtod Local	STOPPED 15 Oct, 2019
■	Quanto tá o dolar? Qtod Local	1 FAILED 15 Oct, 2019
■	Quanto tá o dolar? Qtod Local	PASSED 11 Oct, 2019

Collection runner

Collection Runner

Run Results

Quanto tá o dólar ▾

Run In Command Line

Docs

12
PASSED

0
FAILED

Quanto tá o dólar? Qtod Local
just now

Run Summary ▶

Export Results

Retry

New

Iteration 1

GET

Get value by city

http://localhost:8000/api/...

...anto tá o dólar? / Get value by city

200 OK

1832 ms

97 B

PASS

response is ok

PASS

Response should be okay to process

PASS

Fill current amount

PASS

Response time is less than 200ms

POST

Create an alert

http://localhost:8000/api/...

Quanto tá o dólar? / Create an alert

201 Created

90 ms

223 B

PASS

Alert created

PASS

Response should be okay to process

PASS

Response should be okay to process

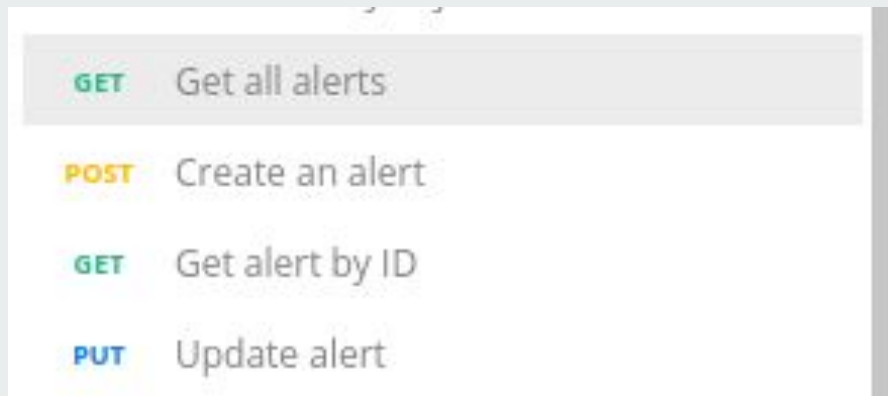
PASS

Response time is less than 200ms

Criando fluxos de execução

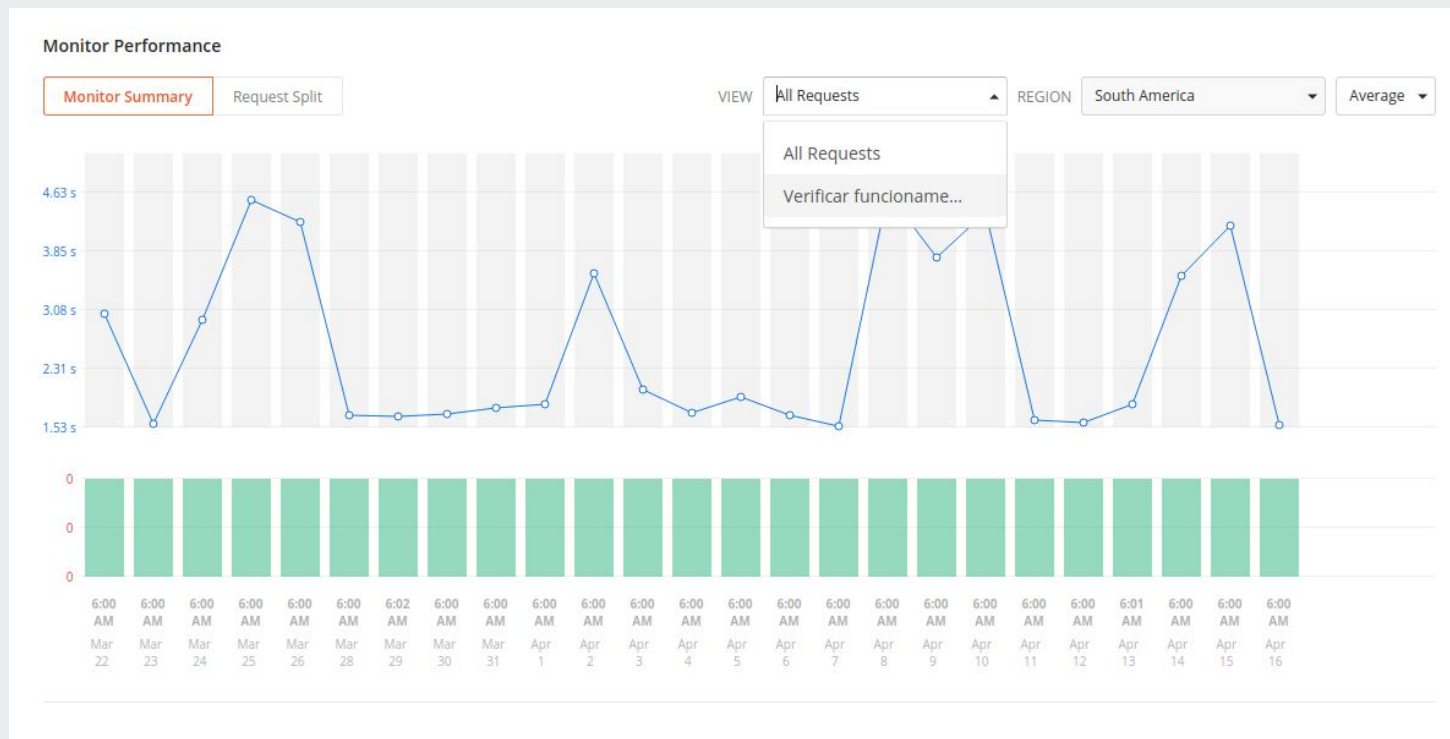
Por padrão, o postman executa as requisições na ordem que elas aparecem dentro da collection, porém através do **setNextRequest** é possível alterar essa ordem

4
1
2
3



No exemplo ao lado eu crio primeiramente um alerta antes de obtê-lo por seu id e atualizá-lo

Postman monitor



Mock Server

O postman mock server permite que você simule o comportamento de sua api baseado nas suas collections

✓

Select requests to mock

2. Configure mock server

3. Next steps

×

Name

Enter a title to describe your requests. This will help you identify your mock server and API collection in Postman.

Use an environment (optional)

Qtod Local

▼

An environment is a group of variables useful for storing and reusing values. [Learn more](#)

☐ Make this mock server private


Calls to a private mock server require a Postman API Key header. [Learn More](#)

Number of calls made to mock servers might be limited by your Postman account. Check your [usage limits](#).

Back

Create

Gerando docs a partir de collections

 **POSTMAN**

Public

Run in Postman

No environment

QUANTO TÁ O DÓLAR?

Introduction

- GET Get value by city
- GET Get all alerts
- POST Create an alert
- GET Get alert by ID
- PUT Update alert
- POST Send Bot Webhook to Qtod
- POST Check webhook
- GET Cron trigger

Quanto tá o dólar?

Api de consulta e criação de alertas de preços do dólar

GET Get value by city

{{url}}/price/buy/rio-de-janeiro/{{amount}}

LanguagePHP

Example Request

Get buy value by city

```
<?php

$curl = curl_init();

curl_setopt_array($curl, array(
  CURLOPT_URL => "{{url}}/price/buy/rio-de-janeiro/{{amount}}",
  CURLOPT_RETURNTRANSFER => true,
  CURLOPT_ENCODING => "",
  CURLOPT_MAXREDIRS => 10,
  CURLOPT_TIMEOUT => 8,
```

Example Response

200 — OK

Newman

```
→ Create product (Bad request)
POST http://localhost:8000/api/products/ [400 Bad Request, 369B, 6ms]
✓ Não conseguiu criar produto com descrição maior igual a 500 caracteres

[] Product / Get
→ Get products
GET http://localhost:8000/api/products/ [200 OK, 1.85KB, 8ms]
✓ Pegou lista de produtos

→ Get product
GET http://localhost:8000/api/products/43 [200 OK, 521B, 8ms]
✓ Retornou produto pelo id
✓ Tempo de resposta menor que 50ms

→ Api root
GET http://localhost:8000/api/ [200 OK, 2.11KB, 5ms]
✓ Api root OK

[] Product / Put
→ Update product
PUT http://localhost:8000/api/products/43 [200 OK, 521B, 37ms]
```

	executed	failed
iterations	1	0
requests	6	0
test-scripts	12	0
prerequisite-scripts	11	0
assertions	7	0
total run duration: 422ms		
total data received: 4.09KB (approx)		
average response time: 23ms		

Rodando os testes de suas collections com o Newman

```
newman run MyCollection.json -e Environment.json
```



Contato



github.com/michelpi



[@Michel_PL](https://twitter.com/Michel_PL)





Útil

- Postman (<https://www.getpostman.com>)
- Docs da api do Postman (<https://docs.api.getpostman.com>)
- Repositório da palestra (<https://github.com/michelp/palestras>)