PRIMEIROS PASSOS COM POSTMAN

UM GUIA PRÁTICO E DIDÁTICO
PARA QUEM QUER APRENDER A TESTAR APIs DO ZERO

POR KAREN.K

O que é o Postman?

O **Postman** é uma ferramenta usada para **testar APIs** (interfaces de programação de aplicações).

Em termos simples: ele permite que você **envie requisições** para um servidor e veja as **respostas** — tudo isso sem precisar escrever código do zero.

> É como um "WhatsApp de QAs e Devs": você manda uma **mensagem (requisição)** para o servidor, e ele responde com uma **mensagem (resposta)**.

Por que usar?

- Permite **validar se uma API está funcionando corretamente**.
- É intuitivo e visual (ótimo pra quem está começando).
- Facilita testes **funcionais, exploratórios e automatizados**.
- Gera **collections** organizadas de testes, com histórico e documentação.
- Permite **colaboração** entre QAs e Devs.

Entendendo o que é uma API

Antes de tudo, entenda o que você está testando.

Uma **API** (Application Programming Interface) é como um **cardápio**: ela lista o que o sistema oferece (endereços, ações, parâmetros) para que outras aplicações possam se comunicar.

Exemplo simples:

- O aplicativo do iFood se comunica com as APIs dos restaurantes.
- O Postman é o "garçom" que faz esse pedido de forma manual para testar se o sistema responde certo.

Principais tipos de requisição

No Postman, você verá diferentes **métodos HTTP** — eles indicam o tipo de ação que você quer fazer com a API.

Pense neles como **verbos de ação**:

Método	Significado	Exemplo	Analogia
GET	Buscar dados	Buscar lista de usuários	“Quero ver o cardápio.”
POST	Enviar/criar dados	Criar um novo usuário	“Quero adicionar um novo prato.”
PUT	Atualizar dados existentes	Atualizar nome do usuário	“Quero mudar o nome do prato.”
PATCH	Atualizar parcialmente	Alterar só o e-mail	“Trocar só um ingrediente.”
DELETE	Excluir dados	Apagar usuário	“Remover um prato do cardápio.”

Explorando o Postman pela primeira vez

1. **Crie uma conta gratuita** em Postman: The World's Leading API Platform | Sign Up for Free .
2. Clique em **Workspaces → Create Workspace** (você pode criar um espaço pessoal para seus testes).
3. Dentro do workspace, crie uma **Collection** — pense nela como uma "pasta" para agrupar suas requisições.
4. Dentro da collection, clique em **New Request**.

Fazendo seu primeiro teste (GET)

Cenário: Buscar usuários na API pública ServeRest

URL: *https://serverest.dev/usuarios*

Passos:

1. No Postman, escolha o método **GET**.
2. Cole a URL acima.
3. Clique em **Send**.

Resultado esperado:

Abaixo aparecerá um **JSON** com lista de usuários.

💬 *Explicando:*

- **URL:** o "endereço" da API.
- **Headers:** informações extras da requisição (ex: tipo de conteúdo).
- **Body:** o corpo da requisição (usado em POST, PUT, PATCH).
- **Response:** é a resposta da API.

Enviando dados (POST)

Cenário: Criar um novo usuário

URL: https://serverest.dev/usuarios

Método: **POST**

1. Selecione **POST**.

2. Vá até a aba **Body → Raw → JSON**.
3. Cole o corpo da requisição:

```
{
"nome": "Karen QA",
"email": "karenqa@test.com",
"password": "1234",
"administrador": "true"
}
```

1. Clique em **Send**.
2. **Status 201 Created** indica sucesso!

💬 *Explicando:*

- 201 significa “criado com sucesso”.
- Você acabou de inserir um novo dado no banco via API!

Outros exemplos de requisições

PUT — Atualizando dados

```
PUT <https://serverest.dev/usuarios/123abc>
{
  "nome": "Karen QA Atualizada"
}
```

DELETE — Excluindo um registro

```
DELETE <https://serverest.dev/usuarios/123abc>
```

Resposta esperada:

```
{
```

`"message": "Registro excluído com sucesso"`

`}`

Entendendo o código de status (Status Code)

Quando você clica em **Send**, a resposta vem com um **número** — esse é o **status da requisição**.

É como o humor da API naquele momento.

Código	Significado	Interpretação
200	OK	Tudo certo
201	Created	Novo recurso criado
400	Bad Request	Erro no envio dos dados
401	Unauthorized	Falta autenticação
404	Not Found	Endpoint não existe
500	Internal Server Error	Erro no servidor

Autenticação e Tokens

Algumas APIs exigem login antes de acessar endpoints protegidos.

1. Faça login com um **POST** (ex: /login)
2. Pegue o token retornado na resposta.
3. Vá em **Headers** → adicione:
4. `Authorization: Bearer seu_token_aqui`
5. Teste os outros endpoints normalmente.

💬 Isso garante que só usuários autenticados possam manipular dados sensíveis.

Collections e Environments

Para deixar tudo organizado:

- **Collections:** guardam suas requisições agrupadas.
- **Environments:** armazenam variáveis, como URLs e tokens.

Exemplo:

`{{base_url}} = <https://serverest.dev>`

`{{token}} = seu_token_aqui`

Assim, se mudar o ambiente (ex: de teste pra produção), basta alterar uma variável!

Criando Variáveis de Ambiente

As **variáveis** servem para armazenar dados reutilizáveis, como URLs, tokens e IDs.

- Vá em **Environments → Create Environment**.
- Crie variáveis, por exemplo:

`base_url = <https://serverest.dev>`

`token = seu_token_aqui`

Agora, em vez de escrever tudo manualmente, use:

`{{base_url}}/usuarios`

Vantagens:

- Facilita mudar ambientes (teste → produção).
- Evita erros e repetições.
- Mantém o workspace limpo e dinâmico.

Automatizando Testes no Postman

Você pode escrever **scripts de teste** simples em JavaScript na aba **Tests**, por exemplo:

```
pm.test("Status code is 200", function () {
    pm.response.to.have.status(200);
});
```

Ou:

```
pm.test("O campo 'email' existe", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.email).to.exist;
});
```

Esses scripts validam automaticamente se as respostas estão corretas!

Checklist de Boas Práticas

- Use variáveis para URLs, tokens e IDs.
- Mantenha collections organizadas.
- Adicione descrições claras em cada requisição.
- Escreva pequenos testes automáticos.
- Use environments diferentes (dev, staging, produção).
- Versione suas collections.
- Teste casos negativos (erros esperados).

“Dominar o Postman é entender o diálogo entre sistemas.

Quanto mais você explora, mais você entende a alma do produto.

E um QA que entende o produto, testa com propósito.”