

QA automatizando testes de API com Robot Framework

**Automatizando Testes de APIs REST com Robot Framework: Guia Prático**

**[[](https://www.linkedin.com/in/chmulato/)](https://www.linkedin.com/in/chmulato/)**

# **[Christian Mulato](https://www.linkedin.com/in/chmulato/)**

Desenvolvedor Java Sênior | Especialista em Back-end | Jakarta, Spring Boot, REST APIs, Docker | Engenheiro Químico

# 6 de agosto de 2025

**A Rotina de Ana, a QA que Mudou Tudo**

Ana era QA sênior numa startup de ***fintech*** em crescimento acelerado. Todos os dias, ela chegava no escritório às 8h e já sabia o que a esperava: uma lista interminável de endpoints para testar manualmente.

“GET /users/123”,

“POST /transactions”,

“PUT /accounts/456”

… Ana abria o Postman, configurava headers, montava payloads JSON, executava requests e anotava resultados numa planilha. Eram mais de 50 endpoints, e cada deploy significava testar tudo novamente.

**O pior?** Sexta-feira às 18h, deploy em produção. Ana ficava até tarde validando cada endpoint crítico, um por um, enquanto o time de desenvolvimento já havia ido embora. Se algo falhasse, rollback e fim de semana trabalhando.

Um dia, após mais um deploy tardio que se estendeu pela madrugada, Ana decidiu: “Tem que ter um jeito melhor.”

Foi então que ela descobriu o Robot Framework. Em duas semanas, Ana automatizou todos os testes de API. O que antes levava 4 horas, agora executava em 15 minutos. Deploys de sexta viraram rotina tranquila, e Ana recuperou seus fins de semana.

Hoje, Ana é referência em automação na empresa. E tudo começou com uma simples keyword:

“GET Deve Retornar 200”

O Robot Framework oferece uma abordagem baseada em palavras-chave para automação de testes de **APIs RESTful**, combinando simplicidade de sintaxe com funcionalidades robustas. Este guia demonstra como implementar testes automatizados legíveis e extensíveis.

# **Requisitos Técnicos**

# **Ambiente necessário:**

# Python 3.8 ou superior

* pip (gerenciador de pacotes Python)

# Robot Framework

# Requests Library

# **Configuração do Ambiente**

pip install robotframework

pip install robotframework-requests

# **Arquitetura do Projeto**

tests/

├── resources/

│ └── keywords.robot

├── suites/

│ └── api\_tests.robot

└── variables/

└── config.robot

# **Configuração de Variáveis**

# **Arquivo: variables/config.robot**

\*\*\* Variables \*\*\*

# ${BASE\_URL}

https://api.seuprojeto.com/v1

# **Implementação de Keywords Reutilizáveis**

# **Arquivo: resources/keywords.robot**

\*\*\* Settings

\*\*\* Library    RequestsLibrary

\*\*\* Keywords

\*\*\* Iniciar Sessão na API     Create Session    api    ${BASE\_URL}

GET Deve Retornar 200     [Arguments]    ${endpoint}     ${response}=

GET On Session    api    ${endpoint}     Should Be Equal As Integers

${response.status\_code}    200

# **Casos de Teste**

# **Arquivo: suites/api\_tests.robot**

\*\*\* Settings \*\*\*

# Resource    ../variables/config.robot

# Resource    ../resources/keywords.robot

# Suite Setup    Iniciar Sessão na API

\*\*\* Test Cases \*\*\*

# Testar Endpoint de Status

# GET Deve Retornar 200    /status

# Testar Endpoint de Usuário

# GET Deve Retornar 200    /users/1

# **Execução dos Testes**

robot -d results tests/suites/api\_tests.robot

O comando gera um relatório HTML detalhado em results/report.html com status de execução e métricas de performance.

# **Vantagens da Implementação**

* **Legibilidade:** Sintaxe próxima à linguagem natural, facilitando compreensão por equipes não técnicas
* **Modularidade:** Keywords reutilizáveis reduzem duplicação de código

# **Integração:** Compatibilidade nativa com pipelines CI/CD

# **Relatórios:** Documentação automática de resultados

# **Integração com CI/CD**

# **Exemplo de GitHub Actions:**

name: API Tests

on: [push]

# jobs:

# test:

runs-on: ubuntu-latest

# steps:

- uses: actions/checkout@v2

- name: Set up Python

uses: actions/setup-python@v2

# with:

python-version: 3.11

- name: Install dependencies

run: |

pip install robotframework robotframework-requests

- name: Run tests

run: robot tests/suites/api\_tests.robot

# **Conclusão**

O **Robot Framework** oferece uma solução robusta para automação de testes de APIs REST, combinando sintaxe acessível com funcionalidades empresariais. A implementação de testes automatizados resulta em maior confiabilidade, redução de tempo de validação e melhoria contínua da qualidade do software.

# **Principais benefícios alcançados:**

* Redução significativa no tempo de execução de testes

# Padronização de processos de validação

# Documentação automática de cenários de teste

# Integração seamless com ferramentas de desenvolvimento

# **Recursos Adicionais:**

# Documentação oficial: [**Robot Framework**](https://robotframework.org/)

# Biblioteca de requisições: Requests Library