Documentação - Postman Automatizado

Guia Completo de Uso e Configuração

Versão 1.0

Índice

1. Objetivo do Programa

O Postman Automatizado é uma aplicação desktop desenvolvida em Python que automatiza o envio de requisições HTTP para APIs. O programa monitora diretórios específicos, processa arquivos JSON e os envia como requisições POST para um endpoint configurado, facilitando a integração entre sistemas através de uma interface gráfica moderna e intuitiva.

2. Bibliotecas Utilizadas

Biblioteca	Função
ttkbootstrap	Framework para criação de interfaces gráficas modernas com tema escuro
requests	Biblioteca para envio de requisições HTTP
watchdog	Monitoramento de diretórios e detecção de eventos de criação de arquivos
pystray	Criação de ícones na bandeja do sistema (system tray)
pillow	Manipulação de imagens para o ícone da aplicação
pyinstaller	Criação de executáveis standalone
pywin32	Integração com o sistema Windows para verificação de instâncias múltiplas

3. Consumo de Recursos

O consumo de recursos do Postman Automatizado varia de acordo com a configuração utilizada:

- **Processamento em Lote:** Quanto maior o número de requisições simultâneas (definido na "Fila de processamento"), maior será o consumo de memória e CPU
- **Execução Automática:** Intervalos muito curtos de verificação podem aumentar o consumo de CPU
- **Monitoramento Contínuo:** Mantém um processo ativo em segundo plano, com consumo mínimo de recursos quando ocioso
- **Requisitos Mínimos:** Python 3.6+, 100MB de RAM livre, conexão com internet para envio das requisições

4. Aba de Configurações

4.1. Campos de Configuração

• **URL Base:** Endereço principal da API que receberá as requisições POST

• **Autenticação:**

- **Username/Password:** Credenciais para autenticação básica HTTP

- **Token/Access Token:** Token para autenticação via Bearer token

- **Client ID/Secret:** Credenciais para autenticação OAuth

• **Timeout:** Tempo máximo (em segundos) que o sistema aguardará por uma resposta da API antes de considerar a requisição como falha. Valores recomendados entre 30 e 300 segundos, dependendo da complexidade da API.

• **Fila de processamento:** Número que define:

1. Quantas requisições serão enviadas simultaneamente quando o modo "Requisições em Lote" estiver ativado

2. Complemento que será adicionado à URL base (ex: se URL base é "https://api.exemplo.com/" e fila é "10", a URL final será "https://api.exemplo.com/10")

5. Aba de Cadastro de Caminhos

5.1. Como Cadastrar Caminhos

1. Defina um nome para identificar o caminho
2. Selecione o diretório de entrada onde os arquivos JSON serão colocados
3. Selecione o diretório de saída onde os arquivos processados serão movidos
4. Configure o tempo de execução (em segundos) para verificação automática
5. Configure as flags conforme necessário
6. Clique em "Salvar Caminhos"

5.2. Funcionalidade da Flag Gatilho

Quando ativada, o processamento só iniciará quando um arquivo específico chamado **gatilho_ini.json** for detectado no diretório de entrada. Este arquivo funciona como um "sinal verde" para iniciar o processamento dos demais arquivos.

• **gatilho_ini.json:** Enviado no início do processamento. Após ser processado, o sistema processa todos os outros arquivos JSON no diretório.

• **gatilho_fim.json:** Enviado ao final do processamento, após todos os outros arquivos terem sido processados. Serve como confirmação de que o lote foi completamente processado.

Esta funcionalidade é útil para garantir que todos os arquivos necessários já estejam no diretório antes de iniciar o processamento, evitando processamento parcial de lotes.

5.3. Requisições em Lote

Quando ativada, esta opção permite o envio simultâneo de múltiplas requisições, aumentando significativamente a velocidade de processamento. O número de requisições simultâneas é determinado pelo valor configurado no campo "Fila de processamento" na aba Configurações.

Considerações importantes:

- Maior paralelismo = maior velocidade, mas também maior consumo de recursos
- Recomenda-se ajustar o valor de acordo com a capacidade do seu sistema e da API de destino
- Valores muito altos podem sobrecarregar a API ou causar erros de timeout

5.4. Execução Automática

Quando ativada, o sistema verificará periodicamente o diretório de entrada em busca de novos arquivos para processar. O intervalo entre verificações é definido pelo campo "Tempo de execução" em segundos.

Recomendações:

- Valores muito baixos podem sobrecarregar o sistema e a API
- Valores recomendados: entre 30 e 300 segundos (0,5 a 5 minutos)
- Para processamento em tempo real, considere usar valores menores com a flag de gatilho ativada

6. Aba de Logs

A aba de Logs exibe informações detalhadas sobre o processamento, incluindo:

- Início e fim do monitoramento
- Detalhes de cada requisição enviada (URL, tempo de resposta, status)
- Erros que possam ocorrer durante o processamento
- Confirmação de arquivos processados e movidos
- Tempo de execução de cada requisição

Os logs são salvos diariamente em arquivos na pasta "log" com o formato "log_DDMMYY.txt" para referência futura e diagnóstico de problemas.

7. Criação de Executável

O aplicativo pode ser distribuído como um arquivo executável (.exe) para Windows, facilitando a instalação e uso sem necessidade de configurar o ambiente Python. Para criar o executável:

1. Execute o arquivo **criar_executavel.bat** com um duplo clique
2. Aguarde o processo ser concluído (pode demorar alguns minutos)
3. O executável será criado na pasta **dist** dentro do diretório do projeto

O executável gerado inclui todas as dependências necessárias e pode ser distribuído para outros computadores Windows sem necessidade de instalação adicional.

8. Exemplos de Uso

Na pasta **exemplos** estão disponíveis modelos de arquivos que podem ser utilizados como referência:

- **exemplo_requisicao.json:** Exemplo de corpo de requisição
- **gatilho_ini.json:** Modelo de arquivo para iniciar o processamento
- **gatilho_fim.json:** Modelo de arquivo para finalizar o processamento

9. Dicas e Solução de Problemas

- Certifique-se de que os diretórios de entrada e saída existam antes de iniciar o monitoramento
- Os arquivos são processados em ordem alfabética pelo nome
- Após o processamento, os arquivos são movidos para o diretório de saída com um timestamp no nome

• Se ocorrerem erros nas requisições, verifique os logs para mais detalhes

• A aplicação pode ser minimizada para a bandeja do sistema (system tray) para continuar o monitoramento em segundo plano