

4.Proxy

October 27, 2019

1 Proxy

Este notebook apresenta os seguintes tópicos:

- Section 1.1 - Introdução
- Section 1.2 - Servidor de proxy

1.1 Introducao

Existe muita informação disponível em repositórios software.

A seguir temos uma *screenshot* do repositório `gems-uff/sapos`.

The screenshot displays the GitHub interface for the repository `gems-uff/sapos`. At the top, the repository name is shown with icons for Watch (6), Star (18), and Fork (11). Below this is a navigation bar with tabs for Code, Issues (39), Pull requests (2), Actions, Projects (0), Wiki, Security, and Insights. The main content area shows the repository's description: "SAPOS main goal is to ease the management of information related to graduate programs such as enrollments, courses, advisement, scholarships, requirements, among others." Below the description are statistics: 1,391 commits, 4 branches, 144 releases, 11 contributors, and the MIT license. A bar chart shows the commit activity over time. Below the statistics are buttons for "Branch: master", "New pull request", "Create new file", "Upload files", "Find file", and "Clone or download". The commit history is shown below, with the latest commit by Carlos-Eduardo-Cabral-da-Cunha (#300) updating the devise gem from 4.6.2 to 4.7.1 due to a security vulnerability. Other commits include fixing scholarship_duration boundaries validation, upgrading rails and gems, adding tests to the spec, adapting migration files, and resizing the rubymine logo.

Commit	Message	Time
app	#293 fixing scholarship_duration boundaries validation in scholarship...	5 months ago
bin	Upgrading rails and gems	5 years ago
config	#293 adding tests to spec of scholarship to validate scholarship_dura...	5 months ago
db	Adapting migration files to work with new Rails version	2 years ago
doc	Resizing again the rubymine logo.	2 years ago

Nessa imagem, vemos a organização e nome do repositório

gems-uff / **sapos** Watch 6 Star 18 Fork 11

Code Issues 39 Pull requests 2 Actions Projects 0 Wiki Security Insights

SAPOS main goal is to ease the management of information related to graduate programs such as enrollments, courses, advisement, scholarships, requirements, among others.

1,391 commits 4 branches 144 releases 11 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

Carlos-Eduardo-Cabral-da-Cunha #300 updating devise gem from 4.6.2 to 4.7.1 due security vulnerability Latest commit d3e4307 on Sep 13

app	#293 fixing scholarship_duration boundaries validation in scholarship...	5 months ago
bin	Upgrading rails and gems	5 years ago
config	#293 adding tests to spec of scholarship to validate scholarship_dura...	5 months ago
db	Adapting migration files to work with new Rails version	2 years ago
doc	Resizing again the rubymine logo.	2 years ago

Estrelas, forks, watchers

gems-uff / **sapos** Watch 6 Star 18 Fork 11

Code Issues 39 Pull requests 2 Actions Projects 0 Wiki Security Insights

SAPOS main goal is to ease the management of information related to graduate programs such as enrollments, courses, advisement, scholarships, requirements, among others.

1,391 commits 4 branches 144 releases 11 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

Carlos-Eduardo-Cabral-da-Cunha #300 updating devise gem from 4.6.2 to 4.7.1 due security vulnerability Latest commit d3e4307 on Sep 13

app	#293 fixing scholarship_duration boundaries validation in scholarship...	5 months ago
bin	Upgrading rails and gems	5 years ago
config	#293 adding tests to spec of scholarship to validate scholarship_dura...	5 months ago
db	Adapting migration files to work with new Rails version	2 years ago
doc	Resizing again the rubymine logo.	2 years ago

Número de issues e pull requests

gems-uff / **sapos** Watch 6 Star 18 Fork 11

[Code](#) [Issues 39](#) [Pull requests 2](#) [Actions](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#)

SAPOS main goal is to ease the management of information related to graduate programs such as enrollments, courses, advisement, scholarships, requirements, among others.

1,391 commits 4 branches 144 releases 11 contributors MIT

Branch: master [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download](#)

Carlos-Eduardo-Cabral-da-Cunha #300 updating devise gem from 4.6.2 to 4.7.1 due security vulnerability Latest commit d3e4307 on Sep 13

app	#293 fixing scholarship_duration boundaries validation in scholarship...	5 months ago
bin	Upgrading rails and gems	5 years ago
config	#293 adding tests to spec of scholarship to validate scholarship_dura...	5 months ago
db	Adapting migration files to work with new Rails version	2 years ago
doc	Resizing again the rubymine logo.	2 years ago

Número de commits, branches, releases, contribuidores e licença

gems-uff / **sapos** Watch 6 Star 18 Fork 11

[Code](#) [Issues 39](#) [Pull requests 2](#) [Actions](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#)

SAPOS main goal is to ease the management of information related to graduate programs such as enrollments, courses, advisement, scholarships, requirements, among others.

1,391 commits 4 branches 144 releases 11 contributors MIT

Branch: master [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download](#)

Carlos-Eduardo-Cabral-da-Cunha #300 updating devise gem from 4.6.2 to 4.7.1 due security vulnerability Latest commit d3e4307 on Sep 13

app	#293 fixing scholarship_duration boundaries validation in scholarship...	5 months ago
bin	Upgrading rails and gems	5 years ago
config	#293 adding tests to spec of scholarship to validate scholarship_dura...	5 months ago
db	Adapting migration files to work with new Rails version	2 years ago
doc	Resizing again the rubymine logo.	2 years ago

Arquivos

gems-uff / sapos

Watch 6 Star 18 Fork 11

Code Issues 39 Pull requests 2 Actions Projects 0 Wiki Security Insights

SAPOS main goal is to ease the management of information related to graduate programs such as enrollments, courses, advisement, scholarships, requirements, among others.

1,391 commits 4 branches 144 releases 11 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

Carlos-Eduardo-Cabral-da-Cunha #300 updating devise gem from 4.6.2 to 4.7.1 due security vulnerability Latest commit d3e4307 on Sep 13

app	#293 fixing scholarship_duration boundaries validation in scholarship...	5 months ago
bin	Upgrading rails and gems	5 years ago
config	#293 adding tests to spec of scholarship to validate scholarship_dura...	5 months ago
db	Adapting migration files to work with new Rails version	2 years ago
doc	Resizing again the rubymine logo.	2 years ago

Mensagem e data dos commits que alteraram esses arquivos por último

gems-uff / sapos

Watch 6 Star 18 Fork 11

Code Issues 39 Pull requests 2 Actions Projects 0 Wiki Security Insights

SAPOS main goal is to ease the management of information related to graduate programs such as enrollments, courses, advisement, scholarships, requirements, among others.

1,391 commits 4 branches 144 releases 11 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

Carlos-Eduardo-Cabral-da-Cunha #300 updating devise gem from 4.6.2 to 4.7.1 due security vulnerability Latest commit d3e4307 on Sep 13

app	#293 fixing scholarship_duration boundaries validation in scholarship...	5 months ago
bin	Upgrading rails and gems	5 years ago
config	#293 adding tests to spec of scholarship to validate scholarship_dura...	5 months ago
db	Adapting migration files to work with new Rails version	2 years ago
doc	Resizing again the rubymine logo.	2 years ago

Podemos extrair informações de repositórios de software de 3 formas:

- Crawling do site do repositório
- APIs que fornecem dados
- Diretamente do sistema de controle de versões

Neste minicurso abordaremos as 3 maneiras, porém daremos mais atenção a APIs do GitHub e extração direta do Git.

1.2 Servidor de proxy

Servidores de repositório costumam limitar a quantidade de requisições que podemos fazer.

Em geral, essa limitação não afeta muito o uso esporádico dos serviços para mineração. Porém, quando estamos desenvolvendo algo, pode ser que passemos do limite com requisições repetidas.

Para evitar esse problema, vamos configurar um servidor de proxy simples em flask.

Quando estamos usando um servidor de proxy, ao invés de fazermos requisições diretamente ao site de destino, fazemos requisições ao servidor de proxy, que, em seguida, redireciona as requisições para o site de destino.

Ao receber o resultado da requisição, o proxy faz um cache do resultado e nos retorna o resultado.

Se uma requisição já tiver sido feita pelo servidor de proxy, ele apenas nos retorna o resultado do cache.

1.2.1 Implementação do Proxy

A implementação do servidor de proxy está no arquivo `proxy.py`. Como queremos executar o proxy em paralelo ao notebook, o servidor precisa ser executado externamente.

Entretanto, o código do proxy será explicado aqui.

Começamos o arquivo com os imports necessários.

```
import hashlib
import requests
import simplejson
import os
import sys
from flask import Flask, request, Response
```

A biblioteca `hashlib` é usada para fazer hash das requisições. A biblioteca `requests` é usada para fazer requisições ao GitHub. A biblioteca `simplejson` é usada para transformar requisições e respostas em JSON. A biblioteca `os` é usada para manipular caminhos de diretórios e verificar a existência de arquivos. A biblioteca `sys` é usada para pegar os argumentos da execução. Por fim, `flask` é usada como servidor.

Em seguida, definimos o site para qual faremos proxy, os headers excluídos da resposta recebida, e criamos um app pro Flask. Note que `SITE` está sendo definido como o primeiro argumento da execução do programa ou como `https://github.com/`, caso não haja argumento.

```
if len(sys.argv) > 1:
    SITE = sys.argv[1]
else:
    SITE = "https://github.com/"
EXCLUDED_HEADERS = ['content-encoding', 'content-length', 'transfer-encoding', 'connection']

app = Flask(__name__)
```

Depois, definimos uma função para tratar todas rotas e métodos possíveis que o servidor pode receber.

```
METHODS = ['GET', 'POST', 'PATCH', 'PUT', 'DELETE']
@app.route('/', defaults={'path': ''}, methods=METHODS)
```

```
@app.route('/<path:path>', methods=METHODS)
def catch_all(path):
```

Dentro desta função, definimos um dicionário de requisição com base na requisição que foi recebida pelo flask.

```
request_dict = {
    "method": request.method,
    "url": request.url.replace(request.host_url, SITE),
    "headers": {key: value for (key, value) in request.headers if key != 'Host'},
    "data": request.get_data(),
    "cookies": request.cookies,
    "allow_redirects": False
}
```

Nesta requisição, substituímos o host pelo site de destino.

Em seguida, convertemos o dicionário para JSON e calculamos o hash SHA1 do resultado.

```
request_json = simplejson.dumps(request_dict, sort_keys=True)
sha1 = hashlib.sha1(request_json.encode("utf-8")).hexdigest()
path_req = os.path.join("cache", sha1 + ".req")
path_resp = os.path.join("cache", sha1 + ".resp")
```

No diretório `cache` armazenamos arquivos `{sha1}.req` e `{sha1}.resp` com a requisição e resposta dos resultados em cache.

Com isso, ao receber uma requisição, podemos ver se `{sha1}.req` existe. Se existir, podemos comparar com a nossa requisição (para evitar conflitos). Por fim, se forem iguais, podemos retornar a resposta que está em cache.

```
if os.path.exists(path_req):
    with open(path_req, "r") as req:
        req_read = req.read()
        if req_read == request_json:
            with open(path_resp, "r") as dump:
                response = simplejson.load(dump)
            return Response(
                response["content"],
                response["status_code"],
                response["headers"]
            )
```

Se a requisição não estiver em cache, transformamos o dicionário da requisição em uma requisição do `requests` para o GitHub, excluimos os headers populadas pelo `flask` e criamos um JSON para a resposta.

```
resp = requests.request(**request_dict)
headers = [(name, value) for (name, value) in resp.raw.headers.items()
            if name.lower() not in EXCLUDED_HEADERS]
response = {
    "content": resp.content,
    "status_code": resp.status_code,
```

```

        "headers": headers
    }
    response_json = simplejson.dumps(response, sort_keys=True)

```

Depois disso, salvamos a resposta no cache e retornamos ela para o cliente original.

```

with open(path_resp, "w") as dump:
    dump.write(response_json)
with open(path_req, "w") as req:
    req.write(request_json)
return Response(
    response["content"],
    response["status_code"],
    response["headers"]
)

```

No fim do script, iniciamos o servidor.

```

if __name__ == '__main__':
    app.run(debug=True)

```

1.2.2 Uso do Proxy

Execute a seguinte linha em um terminal:

```
python proxy.py
```

Agora, toda requisição que faríamos a github.com, passaremos a fazer a localhost:5000. Por exemplo, ao invés de acessar `https://github.com/gems-uff/sapos`, acessaremos `http://localhost:5000/gems-uff/sapos`

1.2.3 Requisição com requests

A seguir fazemos uma requisição com requests para o proxy.

```
[1]: SITE = "http://localhost:5000/" # Se não usar o proxy, alterar para https://
↳ github.com/
```

```
[2]: import requests

response = requests.get(SITE + "gems-uff/sapos")
response.headers['server'], response.status_code

```

```
[2]: ('GitHub.com', 200)
```

Podemos que o resultado foi obtido do GitHub e que a requisição funcionou, dado que o resultado foi 200.

Continua: [5.Crawling.pdf](#)