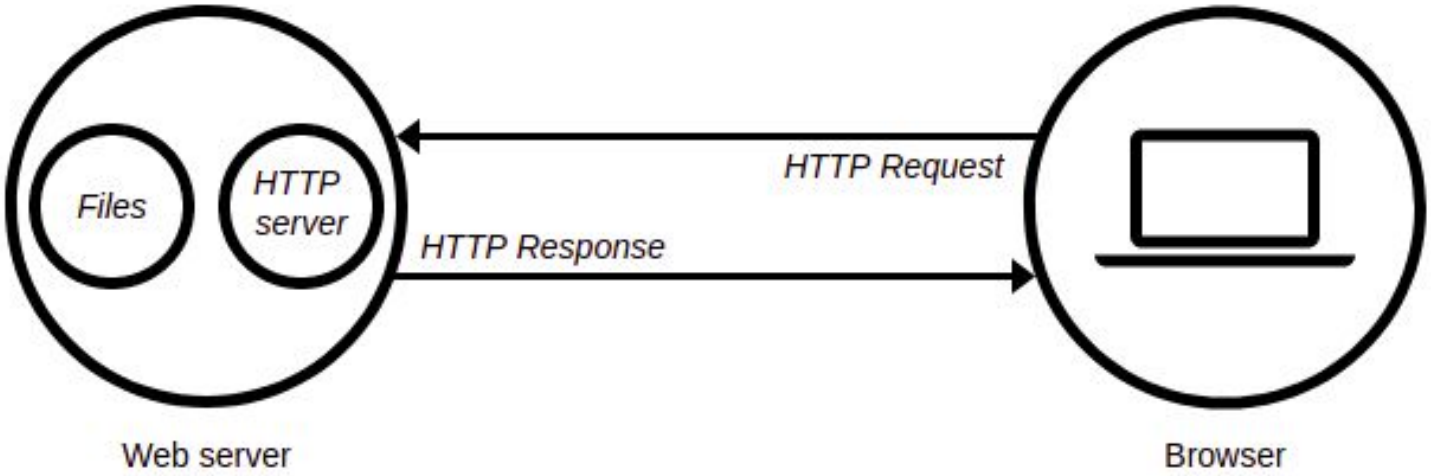
Webserver

Job Queue System

Esquema de Distribuição Web Server



O que é um servidor web (web server)? <https://developer.mozilla.org/pt-BR/docs/Learn/Common_questions/o_que_e_um_web_server>

Servidor Web

- **Estático**
  - envia seus arquivos tal como foram criados e armazenados (hospedados) ao navegador.
- **Dinâmico**:
  - consiste em um servidor web estático com **software adicional**, mais comumente um servidor de aplicações (application server) e um banco de dados (database)

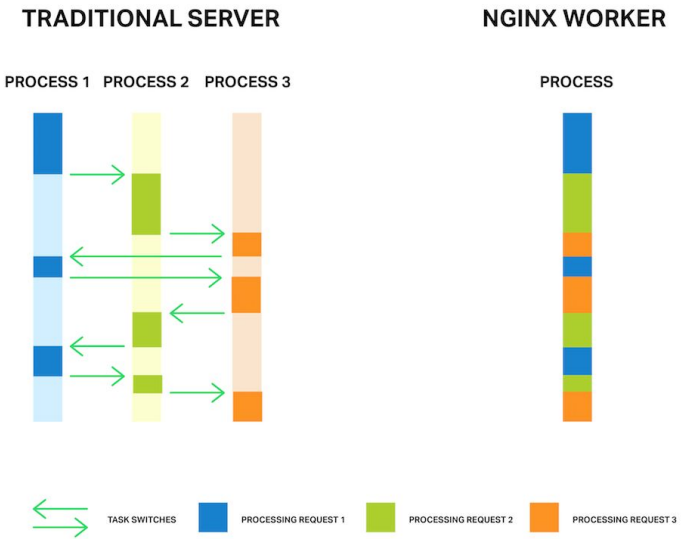
O que é um servidor web (web server)? <https://developer.mozilla.org/pt-BR/docs/Learn/Common_questions/o_que_e_um_web_server>

Web Servers

What are the best web servers? <https://binge.co/what-are-the-best-web-servers**>**

Servidor HTTP Apache

Autor	Rob McCool
Desenvolvedor	Apache Software Foundation
Plataforma	Multiplataforma
Modelo do desenvolvimento	Software livre
Lançamento	1995 (23–24 anos)[1]
Versão estável	2.4.41 (14 de agosto de 2019; há 2 meses[2])
Mercado-alvo	Servidores
Linguagem	C, XML, Forth[3]
Sistema operacional	Tipo Unix, Windows
Gênero(s)	Servidor Web
Licença	Licença Apache
Estado do desenvolvimento	Ativo
Tamanho	8 MB (tarball do fonte)
Página oficial	httpd.apache.org



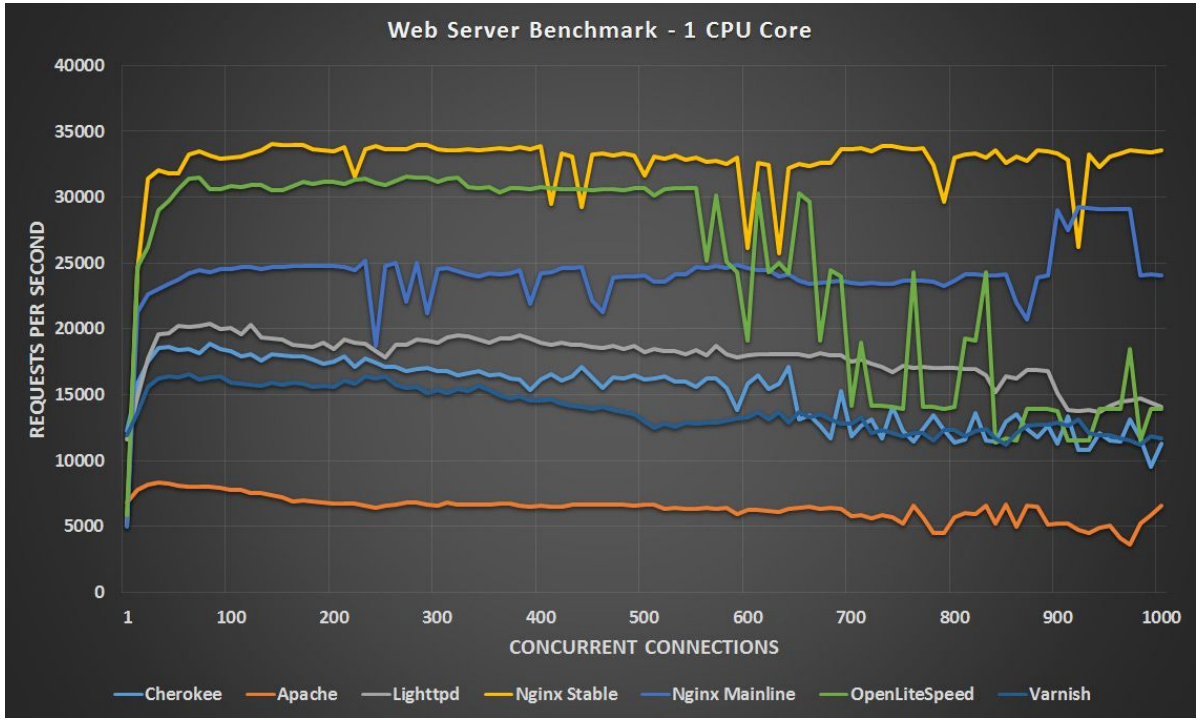
Nginx

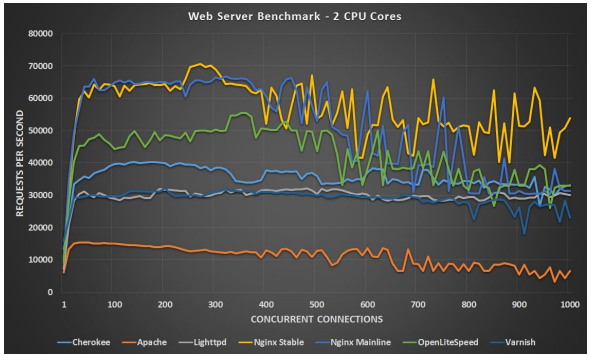
Logotipo do Nginx

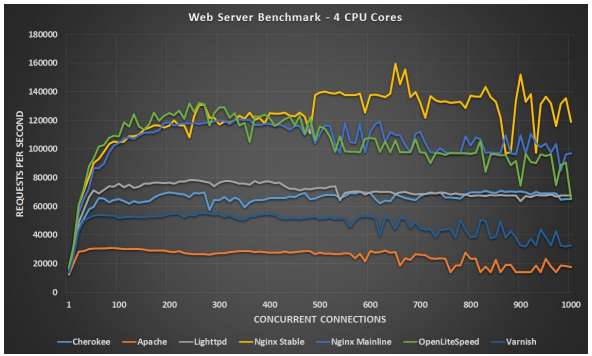
Desenvolvedor	Igor Sysoev
Plataforma	Multi plataforma
Modelo do desenvolvimento	Software livre
Lançamento	4 de outubro de 2004 (15 anos)
Versão estável	1.16.1[1] (13 de agosto de 2019; há 2 meses)
Versão em teste	1.17.4[1] (24 de setembro de 2019; há 30 dias)
Mercado-alvo	Servidores
Linguagem	C, Perl, XML[2]
Sistema operacional	BSD, HP-UX, IBM AIX, Linux, macOS, Solaris, Windows, e outros tipo Unix
Gênero(s)	Servidor HTTP e IMAP/POP3
Licença	BSD
Estado do desenvolvimento	Ativo
Tamanho	1 MB (tarball do fonte)
Página oficial	nginx.org

Introduction to the NGINX JavaScript Module <https://www.nginx.com/blog/introduction-nginscript/**>**

Comparativo de Desempenho







Benchmark de desempenho de servidores Web Linux - Resultados de 2016: <https://www.rootusers.com/linux-web-server-performance-benchmark-2016-results/>
Nginx vs Apache: Confronto Entre Servidores Web: <https://kinsta.com/pt/blog/nginx-vs-apache/>

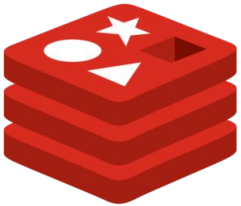
Job Queue System (batch queue)

O uso de uma fila de lotes oferece os seguintes benefícios:

- compartilhamento de recursos do computador entre muitos usuários;
- muda o processamento do trabalho para quando o computador está menos ocupado;
- evita ociosidade dos recursos de computação sem supervisão humana minuto a minuto;
- permite alta utilização 24 horas de recursos de computação caros.

Job queue: <https://en.wikipedia.org/wiki/Job_queue>

Queuing Systems



redis

RabbitMQ

Apache ACTIVEMQ

amazon SQS

Amazon MQ

Serviço gerenciado de agente de mensagens para Apache ActiveMQ

HELIX

Top 6 Queuing Systems for Backend Developers: <https://geekflare.com/queuing-systems-for-backend-developers/>

REDIS

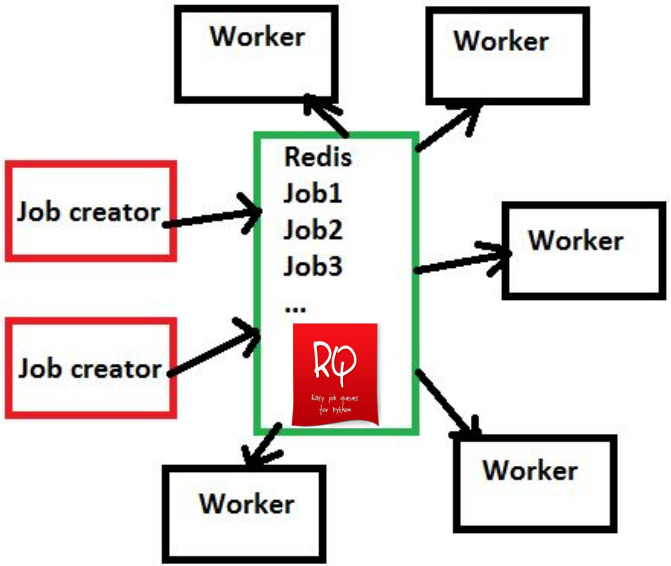
O Redis é uma estrutura de armazenamento de dados em memória de código aberto (licenciado pela BSD), usado como banco de dados, cache e intermediário de mensagens

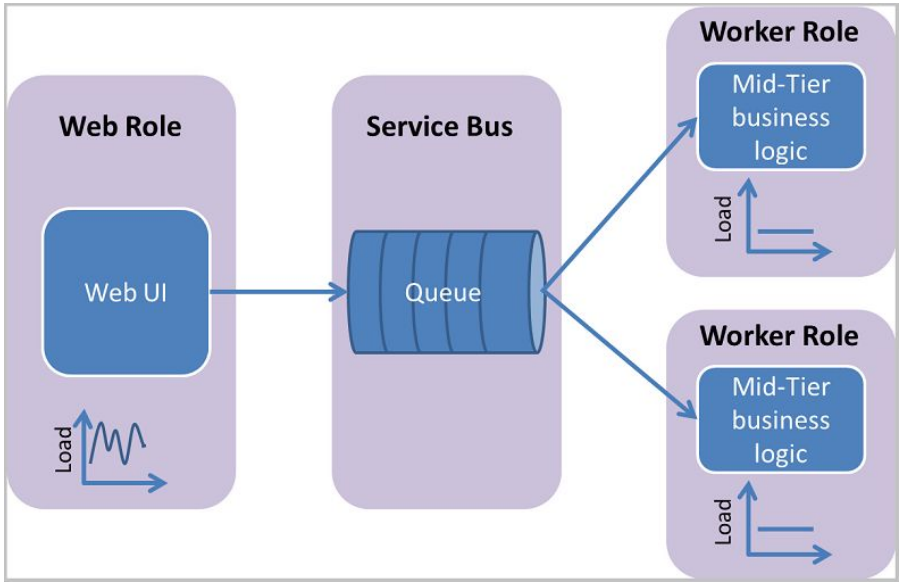
RQ: esasy job queuer for python: <https://python-rq.org/>

redis

RQ: Redis Queue

O RQ (Redis Queue) é uma biblioteca Python simples para enfileirar tarefas e processá-las em segundo plano com os trabalhadores.





RQ: esasy job queuer for python: <https://python-rq.org/> <https://github.com/rq/rq> <https://www.digitalocean.com/community/tutorials/how-to-install-and-use-redis>

Por Que o Redis é Útil?

O desempenho e as funções embarcadas fazem com que o Redis seja melhor que os bancos de dados tradicionais. Os usos típicos do Redis são:

- **Caching –** Sua capacidade melhorada de manter o dado em um disco faz com que ele seja uma alternativa superior às soluções tradicionais de caching.
- **Queuing –** Pode ser usado para fazer uma lista de espera de tarefas em plano de fundo.
- **Countering –** Permite a criação simples e a implementação de contadores sem precisar ler dados ou fazer uploads de esquemas para o banco de dado Redis. Os contadores do Redis vão se manter consistentes.
- **Publishing and Subscribing –** Os usuários podem distribuir dados usando o paradigma Publish/Subscribe.

Começando

Primeiro, execute um servidor Redis. Você pode usar um existente. Para colocar trabalhos em filas, você não precisa fazer nada de especial, basta definir sua função normalmente longa ou de bloqueio:

```
import requests

def count_words_at_url(url):
    resp = requests.get(url)
    return len(resp.text.split())
```

Em seguida, crie uma fila RQ:

```
from redis import Redis
from rq import Queue

q = Queue(connection=Redis())
```

E enfileire a chamada de função:

```
from my_module import count_words_at_url
result = q.enqueue(
             count_words_at_url, 'http://nvie.com')
```

Executar o *Worker*

Para começar a executar chamadas de função enfileiradas em segundo plano, inicie um *worker* no diretório do seu projeto:

```
$ rq worker
*** Listening for work on default
Got count_words_at_url('http://nvie.com') from default
Job result = 818
*** Listening for work on default
```

Prática

https://drive.google.com/drive/folders/1GALyU7A3zyxIUSGe27K-yZZrS_Vfgrpv?usp=sharing