



Guilherme Louro



Publicado em:

Mon 02 March 2015

[←Home](#)

## Configurando ambiente Django com Apache e mod\_wsgi

// Tags [python](#) [django](#) [apache](#) [mod\\_wsgi](#) [virtualenv](#) [virtualenvwrapper](#)

### Entendendo a necessidade

Muitas vezes encontramos dificuldade em colocar nossas aplicações para funcionar em um servidor devido ao pouco conhecimento em infraestrutura, principalmente aqueles que vieram do php, onde, subir um site e já o ver funcionando no ambiente final se trata apenas de subir os arquivos para a pasta **www** e pronto, certo? Não, não é bem por aí ...

Normalmente quando configuramos a hospedagem de um domínio através de um software de gestão de alojamento web (*cpanel é o mais conhecido*) automaticamente o sistema configura o VirtualHost específico para o seu domínio cadastrado, já direcionando a path para a sua pasta **www** ou **public\_html**. Mas como isso é feito? Não entrarei em detalhes de como o cpanel funciona, mas irei demonstrar aqui como configuramos um servidor com [apache](#) para receber nossa aplicação.

### Mas por que o Apache?

A partir do momento que eu mudei meu foco, saindo do PHP para trabalhar com **Python**, eu acabei "abandonando" o Apache para trabalhar com Nginx. Porém, me deparei com um projeto onde tinha que funcionar em uma Hospedagem compartilhada na qual só funciona o apache. Como não vi nada relacionado a essa configuração aqui no **Pythonclub**, achei que seria útil para muitos que podem cair em uma situação parecida com a minha, ou simplesmente prefira usar o Apache do que o Nginx.

Caso o seu interesse seja mesmo usar o Nginx (~~peço para por aqui~~), acho ótimo!!! Te dou todo apoio e ainda te indico um ótimo post para isso, do nosso amigo [Igor Santos](#).

- [Configurando um servidor de produção para aplicações Python](#) (Nginx)

Agora, chega de conversa e vamos ao que interessa.

### Como fazer?

Existem várias maneiras de se fazer o Django trabalhar com apache, uma delas é a combinação Apache + [mod\\_wsgi](#) e será dessa forma que faremos. Com mod\_wsgi podemos implementar qualquer aplicação Python que suporte a interface **Python WSGI**.

#### Instalando alguns pacotes necessários

Antes de mais nada, atualize a lista de pacotes

```
$ sudo apt-get update
```

#### Apache + mod\_wsgi

```
$ sudo apt-get install apache2 libapache2-mod-wsgi
```

#### Python setup tools + pip

```
$ sudo apt-get install python-setuptools  
$ sudo apt-get install python-pip
```

Topo

[OPEN CHAT](#)

## Vamos testar o WSGI?

Vamos fazer um teste com uma aplicação simples em python.

Começe criando um diretório na raiz do apache (*DocumentRoot*)

```
$ sudo mkdir /var/www/wsgi_test
```

Em seguida vamos criar nossa app de teste ...

```
$ sudo vim /var/www/app.wsgi
```

... e escrever nossa app python compatível com WSGI

```
def application(environ, start_response):
    status = '200 OK'
    output = 'Hello World!'
    response_headers = [('Content-type', 'text/plain'),
                        ('Content-Length', str(len(output)))]
    start_response(status, response_headers)
    return [output]
```

Vamos criar agora um host para usar como nosso domínio da aplicação teste

```
$ sudo vim /etc/hosts
```

Adicione esta linha ao seu arquivo hosts

```
127.0.0.1 wsgi_test
```

E vamos configurar nosso VirtualHost no Apache.

```
$ sudo vim /etc/apache2/sites-available/wsgi_test
```

```
<VirtualHost *:80>
    ServerName wsgi_test
    DocumentRoot /var/www/wsgi_test
    <Directory /var/www/wsgi_test>
        Order allow,deny
        Allow from all
    </Directory>
    WSGIScriptAlias / /var/www/wsgi_test/app.wsgi
</VirtualHost>
```

Ative-o

```
$ sudo a2ensite wsgi_test
```

Obs: esse comando cria um link simbólico do wsgi\_test para a pasta sites-enabled. Você pode fazer isso manualmente.

Reinicie o apache:

```
$ sudo service apache2 reload
```

Feito isso abra o ~~internet explorer~~ seu navegador preferido e acesse [http://wsgi\\_test](http://wsgi_test). Se você está vendo a mensagem "Hello World" pode comemorar, o wsgi está funcionando com o apache.

## Configurando Django com WSGI

Até o momento entendemos como funciona a configuração do apache para receber uma aplicação Python com WSGI. Podemos usar essa ideia para qualquer aplicação python, porém veremos como fica essa configuração para trabalhar com **Django**.

Criando o ambiente

Topo

OPEN CHAT

É sempre bom trabalharmos com ambientes virtuais em nossas aplicações python, para isso temos o [virtualenv](#). Eu, particularmente, prefiro usar o [VirtualenvWrapper](#), que separa os ambientes virtuais das aplicações. Caso você não conheça, indico o post do [Arruda](#) que foi o que me guiou quando comecei a usar. Usando [VirtualEnvWrapper](#)

No meu caso usei o virtualenvwrapper e meu filesystem é o seguinte:

```
+-- /home/guilouro
|   +-- .virtualenvs  #[Ambientes]
|   +-- www           #[Projetos]
```

O Virtualenvwrapper criará meus projetos dentro de `www` e os ambientes em `.virtualenvs`. Mas para que isso aconteça temos que adicionar algumas linhas em nosso `~/.bashrc`

```
# adicione no final do arquivo ~/.bashrc
# ...
export PROJECT_HOME=~/.www
export WORKON_HOME=~/.virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

Atualize com o comando:

```
source ~/.bashrc
```

#### Criando nosso projeto

```
$ mkproject wsgi
```

Com as configurações anteriores o virtualenvwrapper já irá ativar o ambiente e levar você para a pasta do projeto. Mas para ativar é muito simples, basta usar:

```
$ workon wsgi
```

Com o nosso ambiente virtual criado partiremos então para a criação do nosso projeto django. Utilizarei a versão mais atual até o momento, nesse caso 1.7

```
$ pip install django
```

Não entrarei em detalhes para a configuração inicial do django, portanto irei usar um template que eu criei para a inicialização dos meus projeto. Criamos então o projeto dessa forma:

```
# startproject pelo template
$ django-admin.py startproject --templatehttps://github.com/guilouro/django-boilerplate/archive/master.zip wsgitest .
# instala os pacotes
$ pip install -r requirements.txt
# faz a migração
$ python manage.py migrate
```

Você encontra esse template aqui -> [django-boilerplate](#)

#### Criando um site no apache para o projeto

Primeiramente, vamos criar um domínio fictício para responder com o nosso projeto ao ser acessado.

```
$ sudo vim /etc/hosts

127.0.0.1    djangowsgi.com
```

Agora vamos configurar o apache:

```
$ sudo vim /etc/apache2/sites-available/djangowsgi
```

```
WSGIDaemonProcess djangowsgi.com python-path=/home/guilouro/www/wsgi:/home/guilouro/.virtualenvs/wsgi/lib/python2.7/site-packages
WSGIProcessGroup djangowsgi.com
```

```
<VirtualHost *:80>
    ServerName djangowsgi.com
    WSGIScriptAlias / /home/guilouro/www/wsgi/wsgitest/wsgi.py

    <Directory /home/guilouro/www/wsgi>
        <Files wsgi.py>
            Order allow,deny
            Allow from all
        </Files>
    </Directory>

    Alias /media/ /home/guilouro/www/wsgi/media/
    Alias /static/ /home/guilouro/www/wsgi/static/

    <Directory /home/guilouro/www/wsgi/static>
        Order allow,deny
        Allow from all
    </Directory>

    <Directory /home/guilouro/www/wsgi/media>
        Order allow,deny
        Allow from all
    </Directory>

</VirtualHost>
```

[Topo](#)[OPEN CHAT](#)

Reinicie novamente o apache:

```
$ sudo service apache2 reload
```

Explicarei agora um pouco do que foi usado nessa configuração

**WSGIScriptAlias:** é a url na qual você estará servindo sua aplicação (/ indica a url raiz), e a segunda parte é a localização de um "arquivo WSGI".

### Modo daemon

O modo *daemon* é o modo recomendado para a execução do mod\_wsgi(em plataformas não-windows). Ele gera um processo independente que lida com as solicitações e pode ser executado como um usuário diferente do servidor web. Um dos pontos positivos dele é que a cada alteração em seu projeto você não precisa restartar o apache, basta executar um `touch` no seu arquivo `wsgi.py`

#### Directivas para o daemon

**WSGIDaemonProcess:** Foi atribuido a ele o nosso servername e utilizamos `python-path` por se tratar de um projeto que esta em ambiente virtual. Passamos então nossas paths nele.

**WSGIProcessGroup:** Atribuímos o servername a ele

### Testando a aplicação

Agora acesse <http://djangowsgi.com> e corre para o abraço.

Espero que tenha ficado claro. Qualquer dúvida ou problema deixe nos comentários e vamos juntos tentar resolver .

#### Referências:

- modwsgiwiki - <https://code.google.com/p/modwsgi/wiki/>
- Blogalizado - [http://www.blogalizado.com.br/deploy-de-aplicacao-django-no-apache-com-mod\\_wsgi/](http://www.blogalizado.com.br/deploy-de-aplicacao-django-no-apache-com-mod_wsgi/)
- Documentação oficial do django - <https://docs.djangoproject.com/en/1.7/howto/deployment/wsgi/modwsgi/>



"Configurando ambiente Django com Apache e mod\_wsgi" de "Guilherme Louro" está licenciado com uma Licença Creative Commons - Atribuição-NãoComercial-SemDerivações 4.0 Internacional .

Compartilhar 13

Compartilhar

Tweetar

## 18 Comentários Pythonclub Blog

Entrar

Recomendar 1

Compartilhar

Ordenar por Mais votados



Participe da discussão...

FAZER LOGIN COM

OU REGISTRE-SE NO DISQUS

Nome

Topo

OPEN CHAT



Everton Massucatto • 3 anos atrás

Guilherme, primeiramente parabéns pelo post! Tenho apenas uma observação a fazer. Ao executar o comando:

```
sudo a2ensite wsgi_test
```

Gera o erro dizendo que o site wsgi\_teste não existe. Para conseguir habilitar o site criei o arquivo wsgi\_test com a extensão .conf (wsgi\_test.conf)

Ficando:

```
sudo vim /etc/apache2/sites-available/wsgi_test.conf
```

Após essa alteração o site foi habilitado com sucesso.

Novamente parabéns! Li outros posts no site e todos são muito bem elaborados.

Abs :)

3 ^ | v • Responder • Compartilhar



**Guilherme Louro** → Everton Massucatto • 2 anos atrás

Muito obrigado pela observação Everton! Vou verificar o porque deste erro.

^ | v • Responder • Compartilhar ›



**Silas Vasconcelos** • 3 anos atrás

Brother muito legal, sempre vejo pouca coisa sobre wsgi no apache, mais acho que todo programador deve saber, pois como as vezes o cliente já tem comprado a hospedagem em algum servidor compartilhado, melhor já saber como fazer do que saber dizer que não é possível ou impossível, novamente parabéns!

2 ^ | v • Responder • Compartilhar ›



**Guilherme Louro** → Silas Vasconcelos • 3 anos atrás

Obrigado Silas!! Esse foi o principal motivo que me fez pesquisar o funcionamento dele junto ao apache. Apesar de não ser o mais indicado, é sim, importante que todo programador saiba e entenda o seu funcionamento. Obrigado pelo feedback amigo!! abraços

^ | v • Responder • Compartilhar ›



**Carlos Jr.** • 2 anos atrás

Boa tarde Guilherme, seu tutorial é perfeito e 100% funcional, porém estou me deparando com o seguinte erro ao acessar o link da minha aplicação :

Forbidden

You don't have permission to access /  
on this server.

preciso da sua ajuda pra solução do erro. não acho solução satisfatoria na internet!

1 ^ | v • Responder • Compartilhar ›



**Alezy Oliveira** → Carlos Jr. • 2 anos atrás

Também tive essa dificuldade :-(

1 ^ | v • Responder • Compartilhar ›



**Henrique Marti** → Carlos Jr. • um ano atrás

Adicionei essa linha:

```
<VirtualHost *:80>
    ServerName.djangowsgi.com
    WSGIScriptAlias / /home/henrique/www/wsgi/wsgitest/wsgi.py

    <Directory /home/henrique/www/wsgi>
        <Files wsgi.py>
            Require all granted
            Order deny,allow
            Allow from all
        </Files>
    </Directory>

    Alias /media/ /home/henrique/www/wsgi/media/
    Alias /static/ /home/henrique/www/wsgi/static/

    <Directory /home/henrique/www/wsgi/static>
        Order allow,deny
        Allow from all
    </Directory>

    <Directory /home/henrique/www/wsgi/media>
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

^ | v • Responder • Compartilhar ›

Topo

OPEN CHAT



**Roberto Ludwig** • 3 anos atrás

Parabéns pelo post, foi muito útil!

1 ^ | v • Responder • Compartilhar ›



**Bruno Moreno** • 3 anos atrás

Mandou muito bem irmão, parabéns!

1 ^ | v • Responder • Compartilhar ›



**Guilherme Louro** → Bruno Moreno • 3 anos atrás

Obrigado meu camarada!!

^ | v • Responder • Compartilhar ›




**Willian Justen de Vasconcellos** • 3 anos atrás

Muito bem explicado, funcionou certinho no mac, a diferença ficou pela instalação dos pacotes com o brew e para o sistema de pastas.

Parabéns =)

1 ^ | v • Responder • Compartilhar ›


**Guilherme Louro**  Willian Justen de Vasconcellos • 3 anos atrás

Show mlk!! Obrigado

1   • Responder • Compartilhar ›**César Javier Fois** • um mês atrás

Muito bom o tutorial, uma consideração que pode ajudar a outros no teste WSGI, para que funcione o comando "

sudo a2ensite wsgi\_test" o nome do arquivo deve terminar com ".conf", não sou nenhum experto somente que estava travado ai.

  • Responder • Compartilhar ›**Andre Nunes** • 9 meses atrás

127.0.0.1 wsgi\_test


uma duvida para responder um ip publico eu tiro o 127.0.0.1 e coloco meu IP publico

  • Responder • Compartilhar ›**Leonardo Claro** • um ano atrás


Percebi outro problema, o apache não esta conseguindo carregar a parte /static/ (tudo da pasta assets):

[authz\_core:error] [pid 21315] [client 127.0.0.1:47000] AH01630: client denied by server configuration: /home/leonardo/www/wsgi/static

Não está carregando os JavaScripts, CSSs, e etc dentro do assets ( com alias do conteudo /static/ )

  • Responder • Compartilhar ›**Leonardo Claro**  Leonardo Claro • um ano atrásDescobri como resolver, ter que dar "workon nome\_projeto" e depois executar o `manage.py` dessa forma`"./manage.py collectstatic"`  • Responder • Compartilhar ›**Leonardo Claro** • um ano atrás

Galera, eu criei outro projeto com 1 virtualenv separado, seguindo os passos abaixo de "Configurando Django com WSGI", se eu deixar ambos projetos habilitado no apache, apenas o ultimo que eu fiz que é exibido corretamente, o anterior fica com erro interno 500, alguem passou por isso?

  • Responder • Compartilhar ›**Leonardo Claro**  Leonardo Claro • um ano atrásJá resolvi, a parte do WSGIDaemonProcess e do WSGIProcessGroup tem que ficar dentro da tag `<virtualhost *:80="">`, ai 1 não irá impactar no outro virtual host.  • Responder • Compartilhar ›

## TAMBÉM EM PYTHONCLUB BLOG

**Abrangência de Listas e Dicionários**

3 comentários • um ano atrás



Hygor Vinicius — muito bom parabéns ....

**What the Flask? pt 4 - Extensões para o Flask**

3 comentários • 10 meses atrás



Régis Silva — Bruno Rocha is back! uhhuuuuuu

**Peewee - Um ORM Python minimalista**

1 comentário • 7 meses atrás

Topo



Hygor Vinicius — parabéns pelo tutorial

OPEN CHAT

**Paralelismo em Python usando concurrent.futures**

12 comentários • 2 anos atrás



Anderson Marques — Primeiro parabéns pelo post José. Usar ThreadPoolExecutor e ProcessPoolExecutor é melhor que usar a

...

 Inscreva-se  Adicione o Disqus no seu site  Adicionar Disqus  Adicionar  Privacidade

[Topo](#)

[OPEN CHAT](#)

