WSGI

a interface para aplicações web python

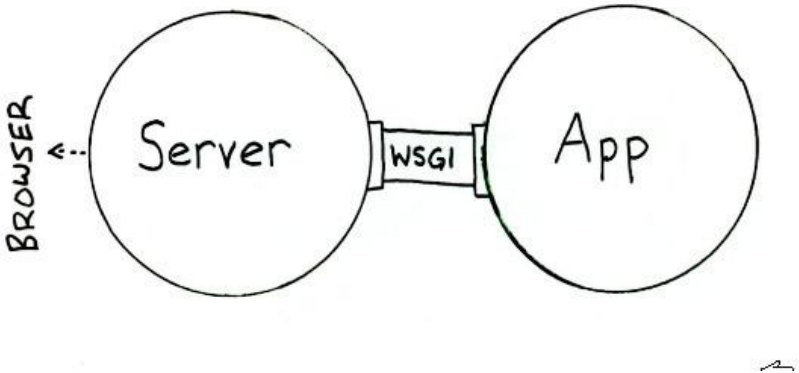
Introdução

Muitas aplicações
Muitos frameworks
Muitos servidores

O que é WSGI?

PEP 333

“This document specifies a proposed standard interface between web servers and Python web applications or frameworks, to promote web application portability across a variety of web servers.”



Aplicação WSGI

```
def applicacao_wsgi(ambiente, resposta_inicio):
    resposta_inicio('200 OK', [('Content-type', 'text/html')])
    return ['<html><body>Hello World!</body></html>']
```

Middleware

Aplicação para o servidor, e servidor para a aplicação

- Lidar com os valores vindos do ambiente
- Mudar o status
- Tratar erros
- Modificar headers
- Definir resposta
- Multiplicidade de aplicações e frameworks

Middleware - exemplo

```
class MiddlewareExemplo:
    def __init__(self, app):
        self.app = app
    def __call__(ambiente, resposta_inicio):

        def resposta_inicio_modificada(status, headers):
            if status[:3] == '200':
                status = '500'
            return resposta_inicio(status, headers)
        return self.app(ambiente, resposta_inicio_modificada)

application = MiddlewareExemplo(application)
```

Por que usar WSGI?

Padrão

Processos separados

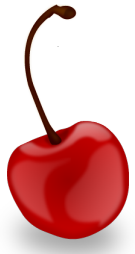
Possibilidade de migração

Exemplos

Frameworks

WEB2PY

django

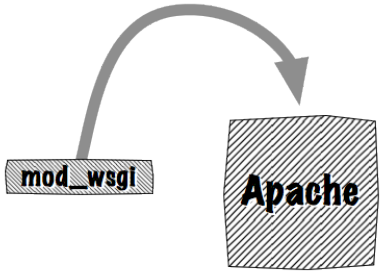


Bottle

webapp2

Servidores

uWSGI



GAE

webapp2

Alta escalabilidade

Não-relacional

Estrutura básica de uma aplicação:

app.yaml
main.html
main.py

Django

Objeto-relacional

Sistema de templates

Estrutura básica de uma aplicação:

project

__init__.py
urls.py
settings.py
manage.py
app

__init__.py
models.py
views.py
tests.py

GAE to Django

Models

```python
# App Engine Datastore model, "webapp/main.py"
from google.appengine.ext import db

class Greeting(db.Model):
    author = db.UserProperty()
    content = db.TextProperty()
    date = db.DateTimeProperty(auto_now_add=True)
```

```python
# Corresponding Django model, "django-guestbook/guestbook/models.py"
from django.db import models
from django.contrib.auth.models import User

class Greeting(models.Model):
    author = models.ForeignKey(User, null=True, blank=True)
    content = models.TextField()
    date = models.DateTimeField(auto_now_add=True)
```

GAE to Django

GET Request

```python
# App Engine GET request handler, "webapp/main.py"
import os

from google.appengine.api import memcache, users
from google.appengine.ext import webapp
from google.appengine.ext.webapp.template import render

class MainHandler(webapp.RequestHandler):
    def get(self):
        user = users.get_current_user()
        greetings = memcache.get('greetings')
        if not greetings:
            greetings = Greeting.all().order('-date').fetch(10)
            memcache.add('greetings', greetings)
        context = {
            'user':      user,
            'greetings': greetings,
            'login':     users.create_login_url(self.request.uri),
            'logout':    users.create_logout_url(self.request.uri),
        }
        tmpl = os.path.join(os.path.dirname(__file__), 'index.html')
        self.response.out.write(render(tmpl, context))
```

```python
# Corresponding Django view, "django-guestbook/guestbook/views.py"
from django.core.cache import cache
from django.views.generic.simple import direct_to_template
from guestbook.forms import CreateGreetingForm
from guestbook.models import Greeting

def list_greetings(request):
    greetings = cache.get(MEMCACHE_GREETINGS)
    if greetings is None:
        greetings = Greeting.objects.all().order_by('-date')[:10]
        cache.add(MEMCACHE_GREETINGS, greetings)
    return direct_to_template(request, 'guestbook/index.html',
        {'greetings': greetings, 'form': CreateGreetingForm()})
```

GAE to Django

URL Routes

```
# App Engine URL configuration
- url: /.*
  script: main.py
```

```
# Django URL configuration, "django-guestbook/guestbook/urls.py"
from django.conf.urls.defaults import *

urlpatterns = patterns('guestbook.views',
    (r'^$',     'list_greetings'),
    (r'^sign$', 'create_greeting'),
)
```

Migrando aplicações

app.yaml

```
application: APP_NAME
version: 1
runtime: python
api_version: 1

handlers:
- url: .*
  script: main.py
```

>> manage.py deploy

Dúvidas?

Referências

- http://legacy.python.org/dev/peps/pep-0333/
- https://cloud.google.com/appengine/articles/django-nonrel
- http://indico.cern.ch/event/44/session/9/material/slides/0?contribId=104
- https://www.djangoproject.com/
- http://docs.python-guide.org/en/latest/scenarios/web/