Web Projects

in Python

Architecture | Organisation | Best practices

http://samuelfuentes.com

- Samuel Fuentes

- Web projects since 2003
- Python/Django since 2009

- @samufuentes
- http://samuelfuentes.com

Contents

Deployment

- Environments:
    - <u>Development</u>
        - Code managing
            - VCS
        - Packages managing
            - virtualenv
    - Staging
    - Testing
    - Production

Virtualenv

- python-setuptools
- python-pip
- python-virtualenv
- pip install virtualenvwrapper

- .bashrc
  - export PIP_RESPECT_VIRTUALENV=true
  - export WORKON_HOME=$HOME/ve
  - source /usr/local/bin/virtualenvwrapper.sh

Virtualenv II

- mkvirtualenv [--no-site-packages] [--python=]
    - workon
    - pip install [-r requirements.txt]
    - add2virtualenv
    - cdsitepackages
    - lssitepackages
    - cdvirtualenv
    - pip freeze > requirements.txt
- deactivate
- rmvirtualenv

Version Control

- VCS: Subversion
- DVCS: Git, Mercurial, Bazaar

GIT

- http://git-scm.com
- https://github.com/

GIT

- git init
- git add .
- git commit -am ”Initial commit”
- git checkout -b newfeature master
- git checkout master
- git merge newfeature

- https://code.djangoproject.com/wiki/CollaborateOnGithub

MVC

- Model
- View
- Controller

Web frameworks

- Grok · http://grok.zope.org/
- Pyramid · http://pylonsproject.org/
- Turbogears · http://turbogears.com/
- Django · https://www.djangoproject.com/

The web framework for perfectionists with deadlines

Django

- https://www.djangoproject.com/download/
- git clone git@github.com:django/django.git

- https://docs.djangoproject.com/en/1.3/intro/tutorial0
- git clone git://github.com/samufuentes/django-Tutorial

- MVC → MTV

Django – A web project

mysite/

- polls/
- templates/
- __init__.py
- manage.py
- settings.py
- urls.py

polls/

- admin.py
- __init__.py
- models.py
- tests.py
- urls.py
- views.py

Django (Model)

- python manage.py sql polls
- python manage.py syncdb

- models.py

Django (View)

- URL mapping
    - urls.py (regular expression, Python callback function [, optional dictionary])
    - Hierarchical urls (r'^polls/', include('polls.urls'))
- views.py

Django (Template)

- templates/
    - 404.html
    - 500.html
    - admin/
        - base_site.html
    - polls/
        - index.html
        - detail.html
        - results.html

?

Testing

- Testing a web project is really complex
    - Python code
    - DB
    - Request / Response
    - Frontend (Template rendering, JavaScript)
    - Load
    - Emails
- Your web framework comes preloaded with tests and utilities → Use them!

Testing code

- unittest
- doctests
- python [-Wall] manage.py test [--failfast] [app[.class[.test]]]

Testing with DB

- Use test db
- Create db once
- Transaction → rollback
- Fixtures
  - python manage.py dumpdata [--indent 2] polls

Testing views

- Send forged request and check responses
- https://docs.djangoproject.com/en/dev/topics/testing/#m
- Example:
  - `c = Client()`
  - `response = c.post("/polls/1/vote/", {'choice': '1'})`
  - `self.assertEqual(response.status.code, 302)`

Testing frontend

- Selenium http://seleniumhq.org/
- IDE, client, RC, Grid
- RC in Java, client in Python
  - `from selenium import selenium`
  - `self.selenium = selenium("localhost", 4444, "*chrome", "http://127.0.0.1:8000/")`
  - `        sel.open("/polls/")`
  - `        try: self.failUnless(sel.is_text_present("exact:Best football team?"))`
  - `        except AssertionError, e: self.verificationErrors.append(str(e))`
- Framebuffer Xvfb, DISPLAY
- Different browsers

?

Deployment production

- Environments:
  - Development
  - Staging
  - Testing
  - Production
    - Own infrastructure
    - Cloud (AWS with boto)
    - Others

Deployment production II

- Nginx http://nginx.org/
    - nginx -s reload
- gunicorn http://gunicorn.org/
    - gunicorn_django [-w X]
    - supervisord http://supervisord.org/
    - upstart http://upstart.ubuntu.com/

Deployment production III

- Cloud: Rackspace, Linode, AWS, ElasticHosts
- AWS with boto http://boto.cloudhackers.com/
  - `import boto`
  - `conn = boto.connect_ec2()`
  - `image = conn.get_image(ami_id)`
  - `reservation = image.run()`
  - `image.stop()`
- Automatization
  - Fabric http://fabfile.org

Deployment production IV

- Last trend: cloud over another layer
  - Djangozoom http://djangozoom.com
  - ep.io http://www.ep.io/
  - gondor.io https://gondor.io/
- Still in beta

Cache

- Caching
  - Hash tables
  - memcached
  - Upstream cache (ISP proxy, client browser)
  - Others (proxy cache, cdn)
- Cache breaking

Cache II

- Storage:
  - memcached
  - DB driven
  - filesystem
  - in-memory file
- What to save
  - site
  - view
  - template
  - low-level

Cache III

```
from django.core.cache import cache
current_site = Site.objects.get_current()
cache_key = "cat-%s" % current_site.id
existing_tree = cache.get(cache_key, None)
if existing_tree is None:
    # some code
    cache.set(cache_key, existing_tree)
```

Cache IV

- memcached http://memcached.org/
- CACHES = {
  - 'default': {
  - 'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
  - 'LOCATION': [
  - '172.19.26.240:11211',
  - '172.19.26.242:11211',
  -]
  - }
- }

Cache V

- Big guns
    - Squid
    - CDN

Cache breaking

- Techniques
    - Unique id for static elements
    - New version = new id
    - Cache forever
- django compressor
  http://django_compressor.readthedocs.org

On the safe side

- Security
  - Web applications
    - SQL injection
    - XSS
    - CSRF
    - SSL
  - Infrastructure
    - Firewall, IDS
    - Server hardening

On the safe side II

- Logging
- Backups
- Monitoring
    - monit http://mmonit.com/monit/
    - Munin http://munin-monitoring.org/
    - Nagios http://www.nagios.org/
    - Pingdom http://pingdom.com/
- Benchmarking

?