Django

CGI и FastCGI

- CGI (Common Gateway Interface) - интерфейс связи Web приложения с Web сервером
  - входные параметры: environment + STDIN
  - вызод: STDOUT

- FastCGI - развитие CGI
  - использует socket-ы для связи
  - один скрипт обрабатывает несколько запросов

История

1. Напишите Web приложение с нуля
2. Напишите еще одно Web приложение с нуля
3. Осознайте, что приложения 1 и 2 имеют много общего кода
4. Проведите рефакторинг - выделите общий код
5. Повторите шаги 2 - 4 много раз
6. Вы сделали еще один Web Framework

Устновка

- Python
- Django
- База данных и Python привязки

see http://www.djangobook.com/en/2.0/chapter02.html

Django project

- Создание проекта:

```
django-admin.py startproject example
```

- Получаем в каталоге example:
  - manage.py - скрипт для управления проектом
  - каталог с настройками проекта (example):
    - settings.py
    - urls.py
    - wsgi.py
- Создание приложения в проекте (не обязательно):

```
python manage.py startapp books
```

- Запускаем сервер:

```
python manage.py startserver
```

Hello, World!

- Создаем скрипт views.py (имя опять не важно):

```
from django.http import HttpResponse

def hello(request):
    return HttpResponse("Hello, World!")
```

- Добавляем в список обработчиков (urls.py):

```
from example.views import hello

urlpatterns = patterns("",
    url(r'^hello/$', hello),
)
```

urls

- Создаем еще один view:

```
from django.http import Http404
from datetime import datetime, timedelta

def time(request, argument):
    offset = int(argument)
    html = “<html><body>In %s hour(s), it will be %s.
</body></html>”
    return HttpResponse(html % (offset, datetime.now() +
timedelta(hours=offset)))
```

- Связываем view с url-ом:

```
from example.views import hello, time
urlpatterns = patterns(“”,
    url(r’^hello/$’, hello),
    url(r’^timeplus/(\d{1,2})/$’, time),
)
```

Шаблоны

```
page_template = """
<html><body>
  <p>Dear {{ person }},</p>

  {% if verbose %}
    <p>Verbose</p>
  {% endif %}

  <ul>
    {% for value in list %}
      <li>{{ value }}</li>
    {% endfor %}
  </ul>

</body></html>
"""
```

Шаблоны

- Пример использования шаблона:

```
from django.template import Template, Context

def template(request, name, verbose):
    template = Template(page_template)
    page = template.render(Context({'person' : name, 'verbose' :
verbose, 'list' : [1, 2, 3, 4]}))
    return HttpResponse(page)
```

- Определяем url:

```
from django.conf.urls import patterns, include, url
from example.views import hello, time, template

urlpatterns = paterrns('',
    url(r'^hello/$', hello),
    url(r'^timeplus/(\d{1,2})/$', time),
    url(r'^template/([a-z]+)/(verbose/)?$', template),
)
```

База данных

- Создаем модель (например, в файле models.py):

```
from django.db import models
class Book(models.Model):
    title = models.CharField(max_length=100)
    authors = models.ManyToManyField(Author)
    publisher = models.ForeignKey(Publisher)
    pubdate = models.DateField()
```

- Включаем приложение в настройках:

```
INSTALLED_APPS = (
    . . .,
    'example',
    . . .,
)
```

- Проверяем модели и создаем таблицы:

```
python manage.py validate   # проверка моделей проекта
python manage.py syncdb     # создание таблиц в базе данных
```

Создание объектов

```
>>> from example.models import Publisher
>>>
>>> first = Publisher(name='Apress', address='2855 Telegraph
Avenue', city='Berkley', state_province='CA', country='USA',
website='http://www.apress.com/ ')
>>> first.save()
>>>
>>> second = Publisher(name="O'Reilly", address='10 Fawcett St.',
city='Cambridge', state_province='MA', country='USA', website='
http://www.oreilly.com/ ')
>>> second.save()
>>>
>>> Publisher.objects.all()

[<Publisher: Publisher object>, <Publisher: Publisher object>]
```

__unicode__

- Добавляем строковое представление:

```
from django.db import models
class Publisher(models.Model):
    . . .
    . . .
    def __unicode__(self):
        return self.name
```

- Проверяем:

```
>>> from example.models import Publisher
>>> Publisher.objects.all()
[<Publisher: Apress>, <Publisher: O'Reilly>]
```

Фильтры

- Метод get возвращает один объект:
  - исключение MultipleObjectsReturned, если объектов много
  - исключение Publisher.DoesNotExist (или ObjectDoesNotExist), если объект не существует

- Метод filter возвращает QuerySet

```
>>> from example.models import Publisher
>>>
>>> Publisher.objects.get(name='Apress')
<Publisher: Apress>
>>>
>>> Publisher.objects.filter(name__contains='press')
[<Publisher: Apress>]
>>>
>>> Publisher.objects.filter(name__contains='r', country='USA'
[<Publisher: Apress>, <Publisher: O'Reilly>]
```

Обновление и удаление

- Обновление:

```
>>> Publisher.objects.filter(id=42).update(name=”Apress
Publishing”)
0
>>> Publisher.objects.all().update(country=’U. S. A.’)
2
```

- Удаление:

```
>>> Publisher.objects.filter(id=42).delete()
>>> Publisher.objects.all().delete()
>>> Publisher.objects.all()
[]
```