Dustan Kasten / 4.19.2011

Introduction to Django

DustanKasten

"Web Designer"

I like people. Therefore I have a user experience bend to my thoughts.

Dustan Kasten / 4.19.2011

Philosophies

MV[C|T]

Applications vs. Projects

DRY

Basic Project

__init__.py	this is for django
manage.py	command-line tools
settings.py	project settings
urls.py	urls are for people, too

Settings.py

```
import os

PROJECT_ROOT = os.path.realpath(
    os.path.dirName(__file__))
```

Settings.py

```
MEDIA_ROOT = os.path.join(
  PROJECT_ROOT, 'media')

MEDIA_URL = '/media/'
```

urls.py

```
url(r'^$',
  direct_to_template('index.html'))
```

urls.py

```
url(r'^$',
    direct_to_template('index.html'))
```

← urls routed via regexsauce

urls.py

```
url(r'^$',  ← urls routed via regexsauce
  direct_to_template('index.html')
  render('index.html')
```

urls.py

```
url(r'^/blog/$',
  include('blog.urls')  ← omg!
```

models.py

```
class Modeling(models.Model):
  text = models.TextField()
  number = models.IntegerField()
  date = models.DateTimeField()
  brother = models.ForeignKey(Brother)
```

models.py

```
class Modeling(models.Model):
  def get_absolute_url(self):
    return self.slug
  def __unicode__(self):
    return self.title
```

forms.py

```
from django import forms
class NewForm(forms.Form):
    text = forms.CharField()
    date = forms.DateField()
    email = forms.EmailField()
```

forms / view

```
def contact(request):
  if request.method == 'POST':
    form = NewForm(request.POST)
    ...
  else:
    form = NewForm()
  return render( ... )
```

forms / view

```
def contact(request):
  if request.method == 'POST':
    ...
    if form.is_valid():
      # process data in form.cleaned_data
      return HttpResponseRedirect( ... )
    ...
```

managers

```
class ModelManager(models.Manager):
  def texted(self):
    return self.filter(text__isnull=True)
```

managers

```
class ModelManager(models.Manager):
  ...
  cursor.execute("""
    # custom SQL query
  """)
  ...
```

views.py

```
def new_view(request):
    # do some awsome logic here
    html = "<html><body> ... </body></html>
    return HttpResponse(html)
```

views.py

HttpRequest and HttpResponse objects

http://docs.djangoproject.com/en/1.3/ref/request-response/

views.py

HttpRequest Attributes

path, path_info, method, encoding, GET, POST, REQUEST, COOKIES, FILES, META, user, session, raw_post_data, urlconf

views.py

HttpRequest Methods

get_host, get_full_path, build_absolute_uri, is_secure, is_ajax, read, readline, readlines, xreadlines, __iter__

views.py

HttpResponse methods

__init__, __setitem__, __selitem__, __getitem__, has_header, set_cookie, delete_cookie, write, flush, tell

views.py

HttpResponse subclasses

Redirect (302), PermanentRedirect (301), NotModified(304), BadRequest (400), NotFound (404), Forbidden(403), NotAllowed (405), Gone (410), ServerError (500)

views.py

django shortcuts

http://docs.djangoproject.com/en1.3/topics/http/shortcuts/

views.py

django shortcuts

render()

redirect()

get_object_or_404()

... there's more too.

views.py

```
def new_view(request):
  article = Article.objects.all()
  return render("template.html",
          { "article" : article, } )
```

templates

This front-end developers dream

http://docs.djangoproject.com/en1.3/ref/templates/

templates

Philosophy

Separation of programming logic and presentation.

No ugly <?php wtf_code_here() ?>

templates

```
{% extends "base.html" %}
{% block content %}
   ...
{% endblock content %}
```

templates

```
{% for a in article %}
  <a href="{{ a.get_absolute_url }}">
    {{ story.headline|upper }}
  </a>
{% endfor %}
```

Resources

doc.djangoproject.com

djangobook.com

code.djangoproject.com/
wiki/DjangoResources

djangoweek.ly

The End

Dustan Kasten
dustan.kasten@gmail.com
@iamdustan