

django

The Web framework for perfectionists with deadlines

14.12.2009

Bernhard Mäser
bernhard.maeser@scale-it.at

what is Django?

„Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design“.

Django is „pythonic“:

```
#not pythonic:  
  
i = 0  
while i<10:  
    do_something(i)  
    i += 1
```

```
#pythonic:  
  
for i in xrange(10):  
    do_something(i)
```

MTV???

Django is different. The classic Model-View-Controller pattern is called MTV:

- Models: object-orientated encapsulation of data, stored in a database (ORM)
- Template: „blueprints“ for formated output, (x)html in most use cases
- View: the logic-layer (controller); works with data, merges template with data and deliver it to the client

modular design

```
/my_project/..  
    ..guestbook/  
    ..blog/  
    ..myapp/..  
        ..models.py  
        ..views.py  
        ..urls.py  
        ..templates/  
    ..another_app/
```

reuse your apps!

building models

```
# blog/models.py

from django.db import models
from django.contrib.auth.models import User

class BlogPost(model.Models):
    title = models.CharField(max_length=200)
    author = models.ForeignKey(User, blank=True, null=True)
    created = models.DateTimeField(auto_now_add=True)
    text = models.TextField()

    class Meta:
        verbose_name = 'post'
        verbose_name_plural = 'posts'
        db_table = 'blog_posts'
        ordering = ('-created',)

    def __unicode__(self):
        return u'%s' % self.title
```

create templates | 2

Websites base-design (base.html):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/
TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
    <title>Django Demo - {%
        block title %}{%
        endblock %}</title>
    <link rel="stylesheet" type="text/css" href="{{ MEDIA_URL }}css/
screen.css"/>
</head>
<body>
    <div id="header">blaha blah</div>
    <div id="content"> {%
        block content %}{%
        endblock %} </div>
    <div id="footer">blah blah blah</div>
</body>
</html>
```

create templates 2/2

blog-site (blog.html):

```
{% extends "base.html" %}

{% block content %}

    {% for blog in blogs %}

        <h1>{{ blog.title }}</h1>
        <p>{{ blog.text }}</p>
        <p> Written by: {{ blog.user.username }} </p>

    {% endfor %}

{% endblock %}
```

write the views

```
#blog/views.py

from blog.models import BlogPost
from django.template import RequestContext
from django.shortcuts import render_to_response, get_object_or_404

def blog_index(request):
    bloglist = BlogPost.objects.all()
    return render_to_response('blog.html', {'blogs':bloglist, },
                             context_instance=RequestContext(request))

def blog_detail(request, blog_id):
    blog = get_object_or_404(BlogPost, blog_id)
    return render_to_response('blog_detail.html', {'blog':blog, },
                             context_instance=RequestContext(request))

@login_required
def json_blog(request, blog_id):
    blog = get_object_or_404(BlogPost, blog_id)
    return util.json_response(blog)
```

design your URLs

```
# urls.py
from django.conf.urls.defaults import *
from django.conf import settings
from django.contrib import admin

admin.autodiscover()

urlpatterns = patterns('',
    (r'^blog/$', 'myproject.blog.views.blog_index'),
    # pass url-fragments as named parameters
    (r'^blog/(?P<blog_id>\d+)/$', 'myproject.blog.views.blog_detail'),
    (r'^ajaxblog/(?P<blog_id>\d+)/$', 'myproject.blog.views.json_blog'),
    # serve static content:
    (r'^media/(?P<path>>.*)$', 'django.views.static.serve',
        {'document_root': settings.MEDIA_ROOT, 'show_indexes': True}),
    # for everything under /admin/ include the urls of the admin app
    (r'^admin/', include(admin.site.urls)),
)
```

forms 1/2

```
# forms.py

from django import forms
from blog.models import BlogPost

class BlogForm(forms.Form):
    class Meta:
        model = BlogPost
        exclude = ('created',)

## use it in a view and pass it to a template:

myform = BlogForm
return render_to_response('blog_create.html', {'form':myform},
                           context_instance=RequestContext(request))
```

forms 2/2

```
# in template:
```

```
<form action="/blog_save/" method="POST">
    {{ form.as_p }}
<input type="submit" value="Submit" />
</form>
```

```
# results in:
```

```
<form action="/contact/" method="POST">

<p><label for="id_title">Title:</label>
    <input id="id_title" type="text" name="title" maxlength="100" /></p>
<p><label for="text">Text:</label>
    <input type="textarea" name="text" id="id_text" /></p>

<input type="submit" value="Submit" />
</form>

# you can also use form.as_table or form.as_ul !
```

technical stuff I/2

I. official supported databases:

- PostgreSQL
- MySQL
- Oracle
- SQLite

2. „community driven“

- MS SQL
- couchDB
- Google AppEngine
-

technical stuff I/2

I. OS:

- Linux / Unix
- Mac OS
- Windows

2. HTTP-Server

- Apache with mod_python
- all WSGI, FastCGI, SCGI or AJP supporting HTTP-Servers (Apache, nginx, lighttpd,...)

„batteries included“ customizable admin app

Django administration

Welcome, adrian. Documentation / Change password / Log out

Home > Polls > What's up? > History

Change history: What's up?

Date/time	User	Action
Dec. 1, 2007, 3:12 p.m.	adrian	Changed date published.

Django administration

Welcome, adrian. Documentation / Change password / Log out

Home > Polls > Add poll

Add poll

Question:

Date information (Show)

Choice #1
Choice:
Votes:

Choice #2
Choice:
Votes:

Choice #3
Choice:
Votes:

Django administration

Welcome, adrian. Documentation / Change password / Log out

Home > Polls > What's up?

Change poll

Question:

Date published: Date: Today
Time: Now

Django administration

Welcome, adrian. Documentation / Change password / Log out

Site administration

Auth

Groups	<input type="button" value="Add"/> <input type="button" value="Change"/>
Users	<input type="button" value="Add"/> <input type="button" value="Change"/>

Sites

Sites	<input type="button" value="Add"/> <input type="button" value="Change"/>
-------	--------------------------------------------------------------------------

Polls

Polls	<input type="button" value="Add"/> <input type="button" value="Change"/>
-------	--------------------------------------------------------------------------

Recent Actions

My Actions

None available

„batteries included“

- authentication
 - users
 - groups
 - permissions
- security
 - no SQL injection (provides own query language)
 - Cross-Site Request Forgery middleware
 - lots of great tools to prevent session-hijacking, XSS, directory traversal

„batteries included“

- form preview
- form wizards
- caching
 - per site
 - per view
 - template fragments
 - low-level
- humanize
 - eg: „intword“ converts 1000000 to 1.0 million
 - or: „naturalday“ converts 15.12.2009 to tomorrow

„batteries included“

- Sessions
- Signals
 - event driven functions
 - build-in signals (eg: `pre_save`, `post_save`,...)
- Sitemap generation
- Internationalisation
- Jython support
- Syndication feeds (RSS/Atom)
- Pagination
- GREAT community and documentation!

questions

????

next: demo, code, configuration