

Django

The Next Generation ~~Bounty Hunt...err~~
Web Application Framework

My Experience

- I am NOT an experienced web application developer.
- Mac pundit, UNIX guru
- I HATE web browser overload.
 - Desktop Apps rule.
 - Web Apps have their place.

So Why Bother

- Like I said, web apps have their place
 - A web browser provides ability to view, edit and modify fairly linear documents.
 - Google Apps is an abuse.
 - We all have web browsers, and the web is the generic starting point for providing information via

What I have I tried

- PHP
 - Yuck.
 - Inelegant.
 - Boring, but functional.
 - Lots of CMSs built on it.



Zend
The *php* Company

Plone

- An interesting platform - Python based.
- Complicated as all heck.
- Breaks at the slightest breeze.
- Result: I hated using it.



TurboGears

- Finally, a framework I thought would beat everything
- Pre-built parts - widgets, easy templating.
- Result: Widgets never really worked for me!



and now, Django.

- Django designed for a publishing environment.
- Prebuilt 'apps' for back-end admin, RSS, comments, registration, etc.
- Excellent documentation.
- Simple, but



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

- Object-relational mapper
- Automatic admin interface
- Elegant URL design
- Template system
- Cache system
- Internationalization

Object-relational mapper

- Each attribute of the model represents a database field.
- Model metadata goes in an inner class named Meta.
- Metadata used for admin site goes into an inner class named Admin.
- Django gives you an automatically-generated database-access API.

```
from django.db import models

class Person(models.Model):
    first_name = models.CharField(max_length=30)
    last_name = models.CharField(max_length=30)
```

Valid Database Types:
postgresql_psycopg2
postgresql
mysql
sqlite3
ado_mssql

Automatic admin interface

- The tedious problem of providing an admin interface already solved.
- Helps provide a clear distinction between content publishers and the public.



Elegant URL design

```
from django.conf.urls.defaults import *

urlpatterns = patterns('',
    (r'^articles/2003/$', 'news.views.special_case_2003'),
    (r'^articles/(\d{4})/$', 'news.views.year_archive'),
    (r'^articles/(\d{4})/(\d{2})/$', 'news.views.month_archive'),
    (r'^articles/(\d{4})/(\d{2})/(\d+)/$', 'news.views.article_detail'),
)
```

- “Cruff-free URLs”
- Mappings done with RegEx to callback functions.
- On match, passes a request object and capture URL parameters.
- Easy reference in templates for perfect URLs

Template system

- Uses their own templating system
- Template is simply a text file, not limited to XML, HTML. Can be used for any text format (e.g. e-mails, CSV)

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

{% block title %}{{ section.title }}{% endblock %}

{% block content %}
<h1>{{ section.title }}</h1>

{% for story in story_list %}
<h2>
  <a href="{{ story.get_absolute_url }}">
    {{ story.headline|upper }}
  </a>
</h2>
<p>{{ story.tease|truncatewords:"100" }}</p>
{% endfor %}
{% endblock %}
```


Sites that use Django

- lawrence.com - An internationally renowned local-entertainment site
- washingtonpost.com - The Washington Post's growing selection of innovative Web database applications.
- chicagocrime.org - A freely browsable database of crimes reported in Chicago.
- LJWorld.com - An industry-leading newspaper site.
- Tabblo - An innovative photo-sharing site, with a narrative twist.
- Toronto Life - Toronto's city magazine.
- lawrencechamber.com - A Chamber of Commerce site that doesn't suck.

Fin

- Django adheres to the DRY principle: “Don’t Repeat Yourself”
- Its strengths:
 - Pre-built ‘applications’
 - Powerful templating
 - Flexible database abstraction

Fin

- Weakness
 - No official AJAX support yet.
 - Designed to be tightly integrated with Django's templating system.
 - It's the web.