Django

Web Uygulamaları Geliştirmede Çağdaş bir Yaklaşım

Temel özellikler

- Modüler, akılcı tasarım
- Tekrarsız kodlama (DRY)
- Model – Şablon – Uygulama mimarisi
  - MTV (veya MVC)
  - Herşey doğru yerde
- Genişletilebilir yapı
- Yüksek yönetilebilirlik

Nesne tabanlı modelleme

- Veri modelinizin öğeleri Python nesneleridir
- SQL yazmadan erişim
- Veritabanı soyutlaması
  - MySQL, PostgreSQL, Oracle, SQLite desteği
  - Aynı arayüz ile erişim

Zarif URL'ler

- Okunabilir URL'ler
  - /index.php?cat=123&p=4&sid=qwe123 *yerine*
  - /blog/duyuru/django-semineri/
- URL'lerinizi fonksiyonlarınızla eşleştirin
  - Regular expression kullanarak URL'lerinizi tanımlayın
  - Platform kısıtlaması yok

Şablon Sistemi

- Grafik tasarımın işlevsel taraftan ve içerikten tamamen ayrılması
- Genişletilebilir esnek şablon dili
- Nesne tabanlı yaklaşım

Kaşe Sistemi

- İstenilen çözünürlükte kaşeleme
  - tüm sayfa veya tek bir sorgu veya bir nesne
- İstenilen kaşe sistemi ile entegrasyon
  - Memcached, DB, dosya, vb. ile kaşeleme
  - Aynı arayüz ile erişim

Otomatik Yönetici Arayüzleri

- Veri modelleriniz ile birlikte yönetici arayüzünüz hazır
  - Yayına hazır kalitede içerik ekleme – silme – değiştirme arayüzleri otomatik olarak elinizde

Çok Dil Desteği

- Uygulamalarınızda tercüme edilmesi gereken içeriği geliştirme sırasında işaretleyin
- Django size istediğiniz diller için tercüme dosyaları üretsin
- Uygulamanız dilediğiniz dillerde aynı şekilde çalışsın
  - Tek satır değiştirmenize gerek yok

Geliştirme ortamı

- Django development server
  - Kolayca test edin
  - Başka server kurmanıza gerek yok
- Uygulamanızı debug edin
- Standard IDE ‘lerle uyum

XML Bazlı İçerik Dağıtımı

- RSS ve ATOM beslemeleri
  - Tüm içeriğiniz için RSS ve ATOM beslemelerini birkaç satır kod ile üretin
- Sitemaps
  - Site içeriğini sitemaps formatında oluşturun
  - Esnek ve kolay

İleri özellikler

- Test platformları ile entegrasyon
  - Unit tests
  - Doc tests
- Middleware
  - Uygulamanız ile request / response seviyelerinde etkileşim
- Sinyaller
  - Olay bazlı işlevsellik ile daha gevşek ilişkilendirme

Örnek Blog Uygulaması

- Veri modelleri
- URL tasarımı
- Uygulama katmanı
- Şablonlar

Modeller

```
from django.db import models

class Blog(models.Model):
    title = models.CharField(max_length=100)
    is_featured = models.BooleanField(default=False)

class Entry(models.Model):
    title = models.CharField(max_length=100)
    body = models.TextField()
    blog = models.ForeignKey(Blog)
```

API erişimi

> b = Blog(title=’güncel’)

> b.save()

> e = Entry(title=’ilk yazı’, body=’Django ile hayat daha kolay’, blog=b)

> e.save()

> b = Blog.objects.get(title=’güncel’)

> entries = Entry.objects.filter(blog=b)

> entries = b.entry_set.all()

> entries[0].title

‘ilk yazı’

URL tasarımı

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

urlpatterns = patterns('',
    ('^blog/(?P<blog_id>d+)/$', 'blog.views.entry_list'),
)
```

“View” Fonksiyonları

```
from django.shortcuts import render_to_response

def entry_list(request, blog_id):
    blog = Blog.objects.get(id=blog_id)
    return render_to_response(‘blog/entry_list.html’,
                                        {‘entries’:
blog.entry_set.all()})
```

Şablon

```
{% for entry in entries %}

    <b>{{ entry.blog.title }}:
{{ entry.title }}</b><br>

    {{ entry.body|truncatewords:50 }}

{% endfor %}
```

İletişim

Django

djangoproject.com

Google Groups

django-tr@googlegroups.com

Onur Mat

omat@teknolab.org