Django

9.9.2010 – Jani Roine

Sisältö

Yleiskatsaus

Koodiesimerkkejä

Mallit, URL-ohjain, näkymät, sivupohjat

Lomakkeiden käsittely

Ylläpitosivusto

Keskustelua

Django lyhyesti

Python-ohjelmointikielellä toteutettu ohjelmistokehys verkkosovellusten kehittämistä varten

“The Web framework for perfectionists with deadlines”

Django sai alkunsa Lawrence Journal-World sanomalehden sisäisenä projektina vuonna 2003

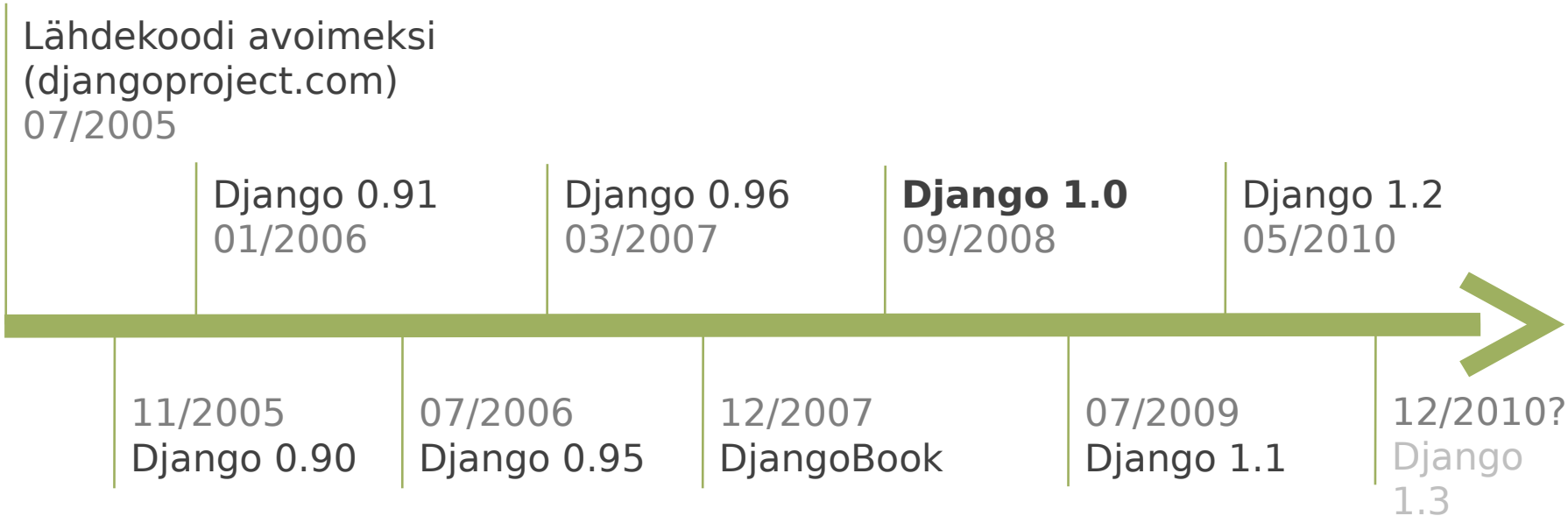
Adrian Holovaty, Simon Willison ja myöhemmin Jakob Kaplan-Moss

Lähdekoodi julkaistiin avoimeksi heinäkuussa 2005

Nimettiin jazz-kitaristi Django Reinhardtin mukaan

BSD-lisenssi

Django 5-v


Lähdekoodi avoimeksi
(djangoproject.com)
07/2005
11/2005
Django 0.90
Django 0.91
01/2006
07/2006
Django 0.95
Django 0.96
03/2007
12/2007
DjangoBook
Django 1.0
09/2008
07/2009
Django 1.1
Django 1.2
05/2010
12/2010?
Django
1.3

Tavoitteet ja periaatteet

Tavoitteena yksinkertaistaa ja nopeuttaa monimutkaisten tietokantapohjaisten verkkosivustojen kehitystä

Vähemmän koodia, vähemmän toistoa

“Avaimet käteen”-periaate

Django perustuu MVC-arkkitehtuuriin (malli-näkymä-ohjain), jonka tarkoituksena on erottaa sovelluksen käyttöliittymä tiedon varastoinnista ja käsittelystä

Djangossa näkymä on osa kontrolleria (model-view-template)

Djangon oppiminen

Python-ohjelmoinnin perusteet

Django on käytännössä kokoelma Python-kielellä toteutettuja kirjastoja

Tietokantojen perusteet

Projektin verkkosivustolla erinomainen tutoriaali sekä dokumentaatio

http://docs.djangoproject.com

Kehittäjien kirjoittama kirja luettavissa verkossa ilmaiseksi

http://djangobook.com

Aktiivinen online-yhteisö

Palvelinvaatimukset

Apache + mod_python tai mod_wsgi

Vaihtoehtoisesti FastCGI:tä tukeva tai mikä tahansa WSGI-yhteensopiva web-palvelin

Django tukee useita tietokantapalvelimia:

PostgeSQL

MySQL

SQLite

Oracle

+ MS SQL, DB2 ja SQL Anywhere liitännät erikseen

Google App Engine sisältää Django 0.96.1-version

Keskeiset ominaisuudet

ORM (object-relational mapper)

mahdollistaa relaatiotietokantataulujen rivien muuntamisen ohjelmiston olioiksi

Säännöllisiä lausekkeita käyttävä URL-ohjain

Sivupohjamoottori

Lomakkeiden käsittely

Kevyt sisäänrakennettu web-palvelin kehitystä ja testausta varten

Tuki sovelluksien internationalisoinnille ja lokalisoinnille

Cache framework

Django-projektin aloitus

python django-admin.py startproject demo

demo/

- __init__.py
- manage.py
- **settings.py**
- urls.py

settings.py

```
...
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': '/path/to/demo/demo.db',
        'USER': '',
        'PASSWORD': '',
        'HOST': '',
        'PORT': '',
    }
}
INSTALLED_APPS = (
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.sites',
    'django.contrib.messages',
    ‘django.contrib.admin’,
    'demo',
)
...
```

python manage.py startapp levylista

models.py

```
# -*- coding: utf-8 -*-

from django.db import models



GENRET = ((1, u'Pop'), (2, u'Jazz'))



class Artisti(models.Model):

    nimi = models.CharField(max_length=100)

    genre = models.PositiveSmallIntegerField(choices=GENRET)

    kotisivu = models.URLField(blank=True)



    def __unicode__(self):

        return self.nimi
```

Tietokantakyselyt

```
>>> from demo.models import Artisti, Julkaisu

>>> Artisti.objects.all()

[]


>>> a = Artisti(nimi="The Cure", genre=1)

>>> a.save()

>>> a.nimi

u'The Cure'

>>> b = Artisti(nimi="The xx", genre=1)

>>> b.save()

>>> Artisti.objects.all()

[<Artisti: The Cure>, <Artisti: The xx>]


>>> j = Julkaisu(artisti=a, nimi="Three Imaginary Boys")

>>> j.save()
```

urls.py

```
from django.conf.urls.defaults import *
from django.contrib import admin
admin.autodiscover()

urlpatterns = patterns('demo.views',
    (r'^$', 'etusivu'),
    (r'^artistit/(?P<artisti_id>\d+)/$', 'artisti'),
    (r'^julkaisut/(?P<julkaisu_id>\d+)/$', 'julkaisu'),
)

urlpatterns += patterns('',
    (r'^admin/', include(admin.site.urls)),
)
```

views.py

```
from django.shortcuts import render_to_response, get_object_or_404

from demo.models import Artisti, Julkaisu


def etusivu(request):
    artistit = Artisti.objects.all()
    return render_to_response('index.html', {'artistit': artistit})


def artisti(request, artisti_id):
    artisti = get_object_or_404(Artisti, pk=artisti_id)
    return render_to_response('artisti.html', {'artisti': artisti})


def julkaisu(request, julkaisu_id):
    julkaisu = get_object_or_404(Julkaisu, pk=julkaisu_id)
    return render_to_response('julkaisu.html', {'julkaisu': julkaisu})
```

Sivupohjamoottori

Sivupohjien avulla voidaan luoda mitä tahansa tekstimuotoisia dokumentteja

Sivupohjissa on tekstiä, muuttujia sekä niihin liittyviä suotimia ja tageja

Muuttujia käsitellään notaatiolla {{ artisti.nimi }}

Suotimia voidaan ketjuttaa muuttujiin

- {{ artisti.nimi|upper }} “The Cure” > “THE CURE”
- {{ artisti.nimi|truncatewords:1 }} “The Cure” > “The …”

Tagien avulla voidaan esimerkiksi tulostaa tekstiä, läpikäydä tietueita ja sisällyttää uusia tiedostoja

- {% if muuttuja %} ja {% else %}
- {% for %}
- {% url %}
- {% comment %}

Sivupohjamoottoria on mahdollista laajentaa luomalla uusia suodattimia ja tageja

Esimerkkejä sivupohjista

templates/base.html

```
<!doctype html>
<html lang="en">
<head>
<title>{% block title %}Demo{% endblock %}</title>
</head>
<body>

 <h1>Demo</h1>

 <div id="content">
  {% block content %}{% endblock %}
 </div>

</body>
</html>
```

templates/index.html

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

{% block content %}
 <ul>
 {% for artisti in artistit %}
  <li>
   <a href="{% url demo.views.artisti artisti.id %}">
    {{ artisti.nimi }}
   </a>
  </li>
 {% endfor %}
 <ul>
{% endblock %}
```

Lomakkeiden käsittely

Djangon Form-luokan tarkoituksena on helpottaa lomakkeiden käsittelyä

Koska lomake on usein samankaltainen malliluokan kanssa, voidaan se johtaa suoraan mallista

```
from django import forms


GENRET = ((1, u'Pop'), (2, u'Jazz'))


class ArtistiForm(forms.Form):
    nimi = forms.CharField()
    genre = forms.ChoiceField(choices=GENRET)
    kotisivu = forms.URLField()
```

```
from django.forms import ModelForm
from demo.models import Artisti


class ArtistiForm(ModelForm):
    class Meta:
        model = Artisti
```

Lomakkeiden käsittely

Lomakkeen esittämiseksi näkymässä lomake annetaan osaksi kontekstia ja lomakkeelta pyydetään sen HTML-esitys

Lomaketta voi tarvittaessa muotoilla itse

```
<form action="{% url demo.views.palaute %}" method="post">
    {{ form }}
    <p><input type="submit" value="Tallenna"></p>
</form>
```

Lomakkeiden käsittely

Esimerkki palautelomakkeen käsittelystä näkymässä

```
def palaute(request):
    if request.method == 'POST':
        form = CommentForm(request.POST)
        if form.is_valid():
            return HttpResponseRedirect('/kiitos/')
    else:
        form = CommentForm()
    return render_to_response(‘lomake.html', { 'form': form, })
```

Ylläpitosivusto

Dajngon ylläpitosivuston avulla voidaan lisätä, selata, muokata ja poistaa sisältöä tietokannasta

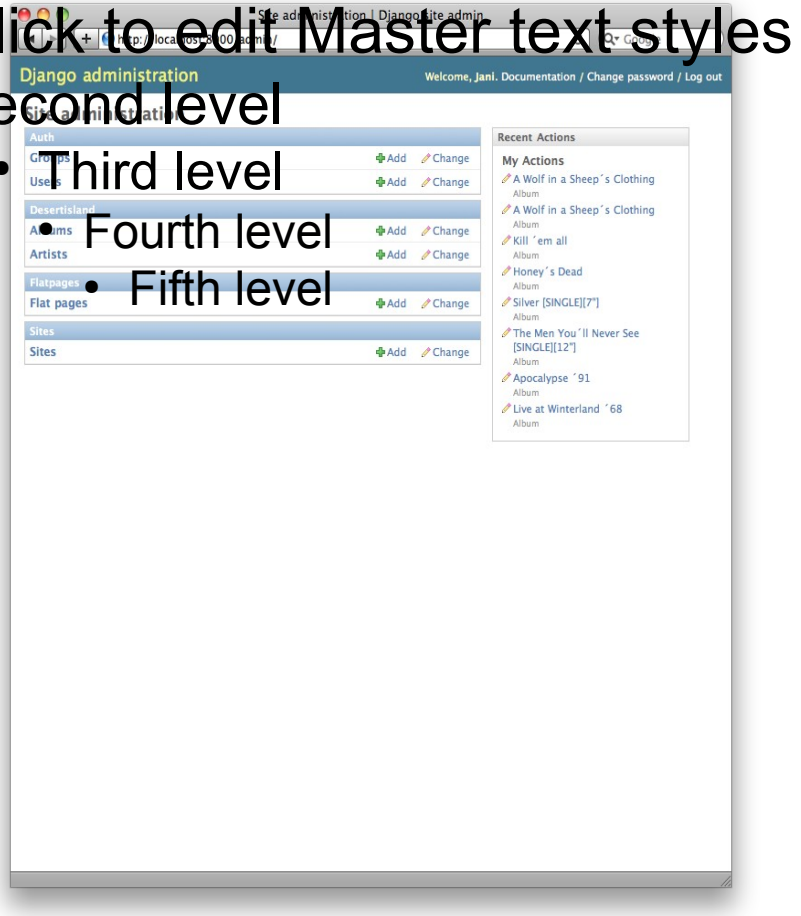
Ominaisuuksia:

Haku

Sisällön suodatus

Uudelleenjärjestys valitun sarakkeen mukaan

Ylläpitosivuston ulkoasua voi muuttaa sivupohjia ja tyylitiedostoa muokkaamalla



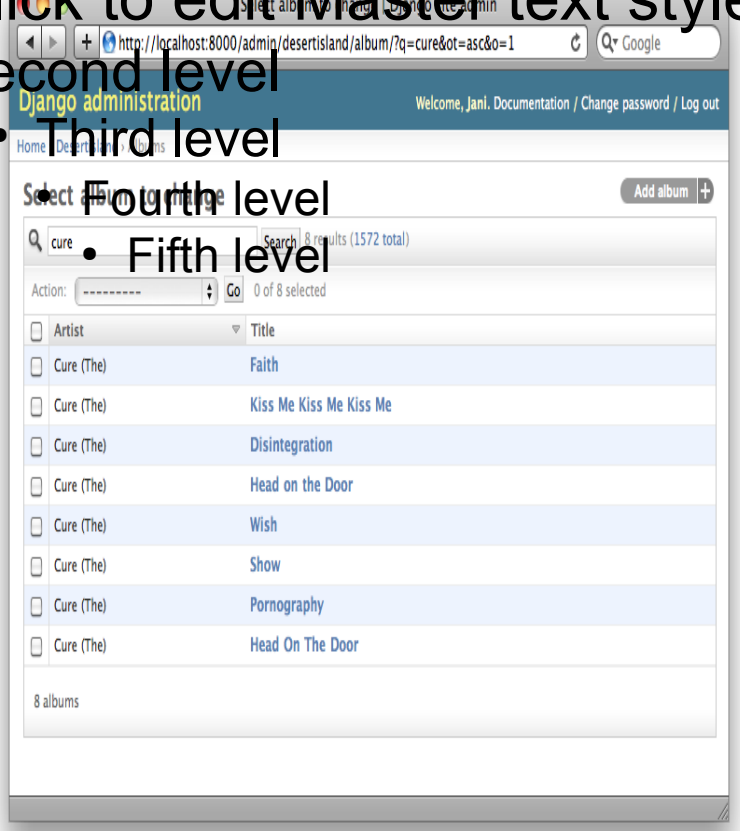
admin.py

```
from django.contrib import admin

from demo.models import Artisti, Julkaisu


class ArtistiAdmin(admin.ModelAdmin):

    pass

admin.site.register(Artisti, ArtistiAdmin)


class JulkaisuAdmin(admin.ModelAdmin):

    list_display = ('artisti', 'nimi')

    list_display_links = ('nimi',)

    search_fields = ['artisti__nimi', 'nimi']

admin.site.register(Julkaisu, JulkaisuAdmin)
```

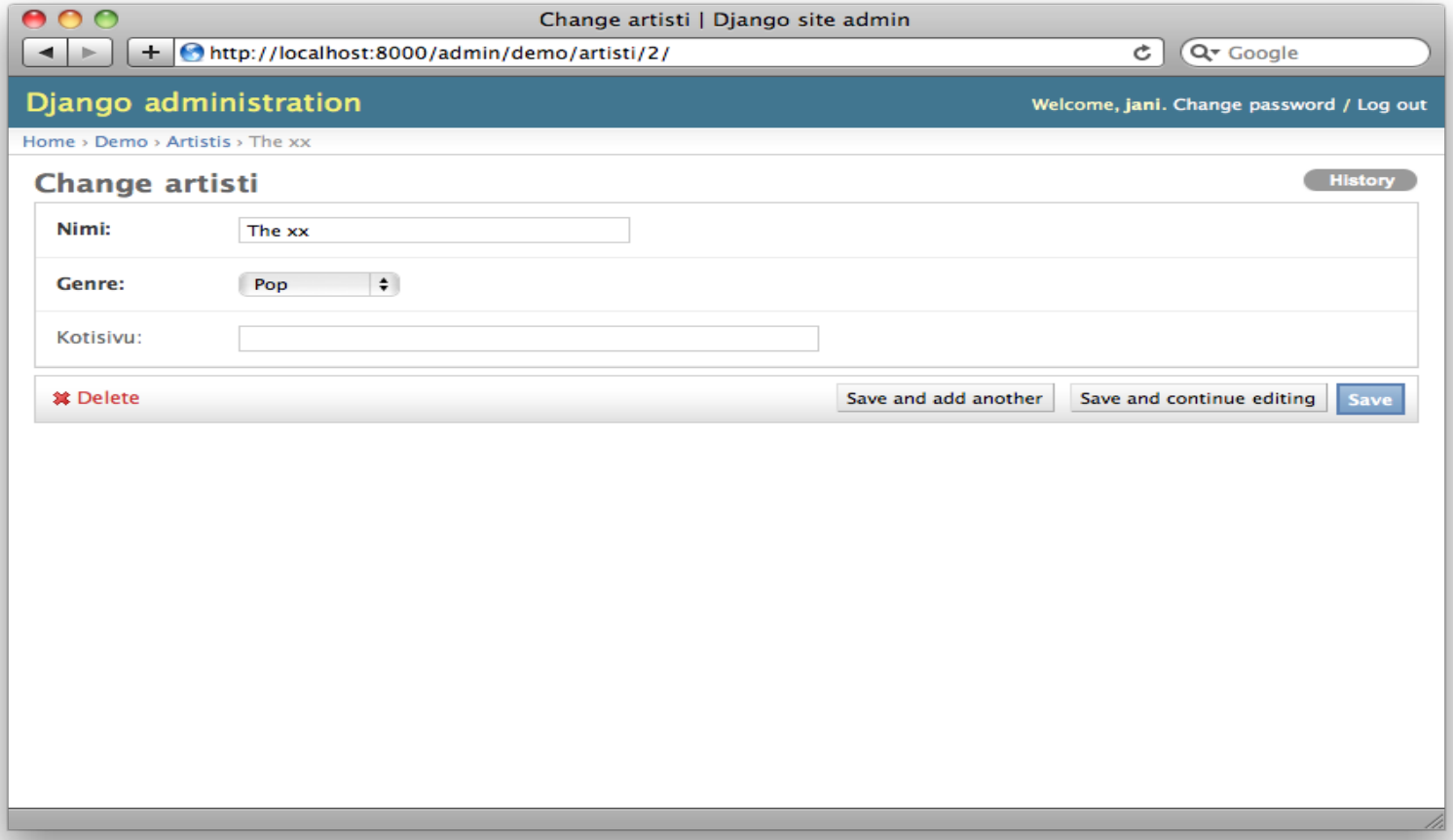
Click to edit Master text styles

Second level

- Third level
- Fourth level
- Fifth level



Ylläpitosivusto



Djangon mukana tulevia sovelluksia

- Admin site
- Authentication
- Flatpages
- Comments
- Pagination
- Syndication feeds (RSS/Atom)

Poimintoja uusista ominaisuuksista

1.0

Uudelleen kirjoitettu ylläpitoliittymä

Parannettu Unicode-tuki

GeoDjango

1.1

ORM-parannuksia

Admin “actions”

1.2

Tuki usean tietokannan käyttämiseen

Mallin validointi

Messages framework

Oikeuksien määrittely objekti-tasolla

Parannuksia sivupohjamoottoriin, mm. “fiksu” if-tagi

Djangolla toteutettuja sivustoja

EveryBlock (www.everyblock.com)

Bitbucket (www.bitbucket.org)

Washington Post

Lawrence.com

...

Lisää osoitteessa www.djangosites.org

Linkkejä

Django (http://djangoproject.com)

The Django Book (http://djangobook.com)

Django Users (http://groups.google.com/group/django-users)

Django Snippets (http://djangosnippets.org)

Django Developers (http://groups.google.com/group/django-developers)