

Django: unchained

by Serena Simkus

Abstract

This talk aims to provide an introduction to what Diango is, where it came from, and some of the cool features it offers. I will show through examples some of what Django's ORM, template language, and admin page look like and what they do, and how to use them.

Disclaimer

I have used Django for a total of two projects, plus following the tutorial, and the simple demo I made for this presentation. I am not claiming to be a Django master.

What is Django?

- Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design.
- free and open source web application framework

Where did Django come from?

- developed from a online-news operation
- primary goal is to "ease the creation of complex, database-driven websites"
- lets you build high-performing, elegant Web applications quickly
- written in Python, 2005

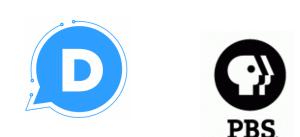
Django...

- adheres to the DRY principle
- emphasizes reusability
- focuses on automating as much as possible
- provides an optional admin account

These companies use Django:

- Pinterest
- Instagram
- Mozilla Foundation
- The Washington Times
- <u>Disqus</u>
- Public Broadcasting Services (PBS)
- OpenStack







Cool Django Features

- Object-relational mapper (ORM)
- Automatic admin interface
- Elegant URL design
- Template language
- Cache system
- Internationalization

Django's ORM

Object-relational mapping:

A model is the single, definitive source of information about your data. It contains the essential fields and behaviors of the data you're storing. Generally, each model maps to a single database table.

Django's ORM

The basics:

- Each model is a Python class that subclasses django.db.models.Model
- Each attribute of the model represents a database field
- With all of this, Django gives you an automatically-generated database-access API

Django's ORM

from django.db import models

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

Equivalent table created in SQL:

CREATE TABLE myapp_person ("id" serial NOT NULL PRIMARY KEY, "first_name" varchar(30) NOT NULL, "last_name" varchar(30) NOT NULL

);

Django's Admin Interface

As simple as configuring your settings, and creating a user:

python manage.py createsuperuser
 Fill in username and password
 And then go to your_domain/admin

Django's Template Language

• Templates

- \circ simply a text file
- contains variables and tags
- base.html

• Variables

- 0 {{ variable }}
- evaluates variable and replaces with result
- alphanumeric characters and underscore only
- dot notation to access attributes of a variable

Django's Template Language

• Filters

- modify variables for display by using filters
- apply filters with a pipe (|), filters can be chained
- examples:
 - {{ list|join:", " }}
 - { { value | length }}
 - { { value | striptags }}
 - { name lower }}

Django's Template Language

• Tags

- create text in the output, load external information
- control flow, loops or logic
- for, if elif else, block, and extends
- Comments
- Template inheritance
 - o {% extends "base.html" %}
 - o {% block content %}Content here{% endblock %}

Django vs. Flask

- not a fair comparison, depends on what you want to do with either of them
- Flask is simpler, Django more complicated
- Flask has less built-in stuff, Django has more features and is more powerful and robust
- Flask is pure Python, Django has its own 'pseudo' language
- Flask is easy to get started, Django requires a little learning before starting

Summary

- Django is awesome!!
- ORM can help save time and makes CRUD operations a lot nicer and easier
- Admin account makes interacting with the page content super simple
- Template language makes working with models and page content easier and more integrated, template tags and logic make HTML less ugly
- Great documentation, easy to learn, and makes web development faster and easier for most people

Additional Reading/References

- <u>https://www.djangoproject.com/</u>
- <u>https://docs.djangoproject.com/en/1.7/intro/tutorial01/</u>
- http://www.fullstackpython.com/django.html
- <u>https://docs.djangoproject.com/en/1.7/ref/templates/api/</u>
- <u>http://c2.com/cgi/wiki?DontRepeatYourself</u>
- (funny) video comparing Django and Flask:
 <u>https://www.youtube.com/watch?v=v2g_jp9sfEo</u>
- (funny) video comparing Django and Rails:
 - o <u>https://www.youtube.com/watch?v=_gjycsotm6M</u>
- <u>https://bernardopires.com/2014/03/rails-vs-django-an-in-</u> <u>depth-technical-comparison/</u>