

# Zero-to-Wiki in 30 minutes

Building a Simple Wiki with Django

Brad Montgomery

<http://bradmontgomery.net/bcmem/>

# What is Django?

- Billed as “The Web framework for perfectionists with deadlines”
- Written in Python
- Adherence to the DRY principle
- Model-View architecture (loosely MVC)
- More at: [djangoproject.com](http://djangoproject.com)

# Purpose?

- Introduce Django (a pretty cool tool)
- Allude to “What” and “How”
- Provide resources to Get Started!

# Example: a Wiki

- We're building a VERY simplistic wiki...
  - Add/Edit pages (including wiki markup)
- Assumptions (stuff you'll need beforehand):
  - Python+Django(v1.0)+Database
  - Some python/command-line familiarity
- Django Parts: Models, Views, Forms, Templates

# Project: Zipi

- Django Project = site-wide settings (zipi)
  - `django-admin.py startproject zipi`
  - Creates files in `zipi/`:
    - `__init__.py`
    - `manage.py`
    - `settings.py`
    - `urls.py`
- NOTE: “zipi” is a python package.

# App: wiki

- From within zipi/
- Django App = Web-based application (wiki)
  - `python manage.py startapp wiki`
  - Creates files in wiki/:
    - `__init__.py`
    - `models.py`
    - `view.py`
  - NOTE: “wiki” is also a python package.

# The Model

- The Definitive source of your Data!
- Usually maps to a Database Table
- Django provides an ORM...
- Consists of Fields and their Behaviors
  - written as a python class, see “models.py”
- Afterwards run: `python manage.py syncdb`

# Automatic Admin

- Django offers add-on “contrib” apps...
- admin: Provides Web-based CRUD interface for users
- “Tie” our model to Django’s admin
- Create a new file in wiki/
  - See “admin.py”

# URLConf

- Django promotes Clean URL's!
- URL Configuration (aka URLConf)
  - Mapping of URL patterns to Python Callback functions (aka views)
  - Based on Regular Expressions
- See "urls.py"

# Views

- Accepts an http request (with optional parameters) and returns an http response.
- Optionally loads a template, and creates a context (a dictionary mapping template variables to python objects)
- See “views.py”

# Forms

- Forms validate and clean input data.
- Derived from a Form class, and consist of Fields, similar to Models.
- ModelForm: Special case of Forms used to add/edit a Django Model.
- See “forms.py”

# Templates

- Django's "output" is "consumed" through templates... (xml, html, csv, json, ... etc)
- Django's template language != python
- Meant to express presentation (not logic)
- Template language consists of:
  - **Tags:** `{% for page in pages %} {% endfor %}`
  - **Variables:** `{{ page }}`
  - **Filters:** `{{page|title }}` **or** `{{t|timesince }}`

# Template Inheritance

- Templates can be **extended**
- **blocks** can be defined/overridden
- Templates can also be **included**
- See “[zipi/templates/](#)”
- (Tags/Filters can be be customized...)

# Resources

- Django Website: <http://djangoproject.com>
- Mailing List:
  - <http://groups.google.com/group/django-users/>
- IRC: #django on irc.freenode.net
- Books? (currently lag behind version 1.0)
- Friends? Python User Groups?