Feb 5th, 9:00 AM - 11:00 AM

Moving to a Sustainable Web Development Environment for Library Web Applications

Anjanette Young
University of Washington

Moving to a sustainable web development environment for library web applications.

Agenda

1. Background about our Department
2. Theory of Application Design. MVC, MTV
3. Benefits of porting to Django
4. Case Study: PCAD (Some coding talk)
5. Other projects ported to Django.

Django Rocks!

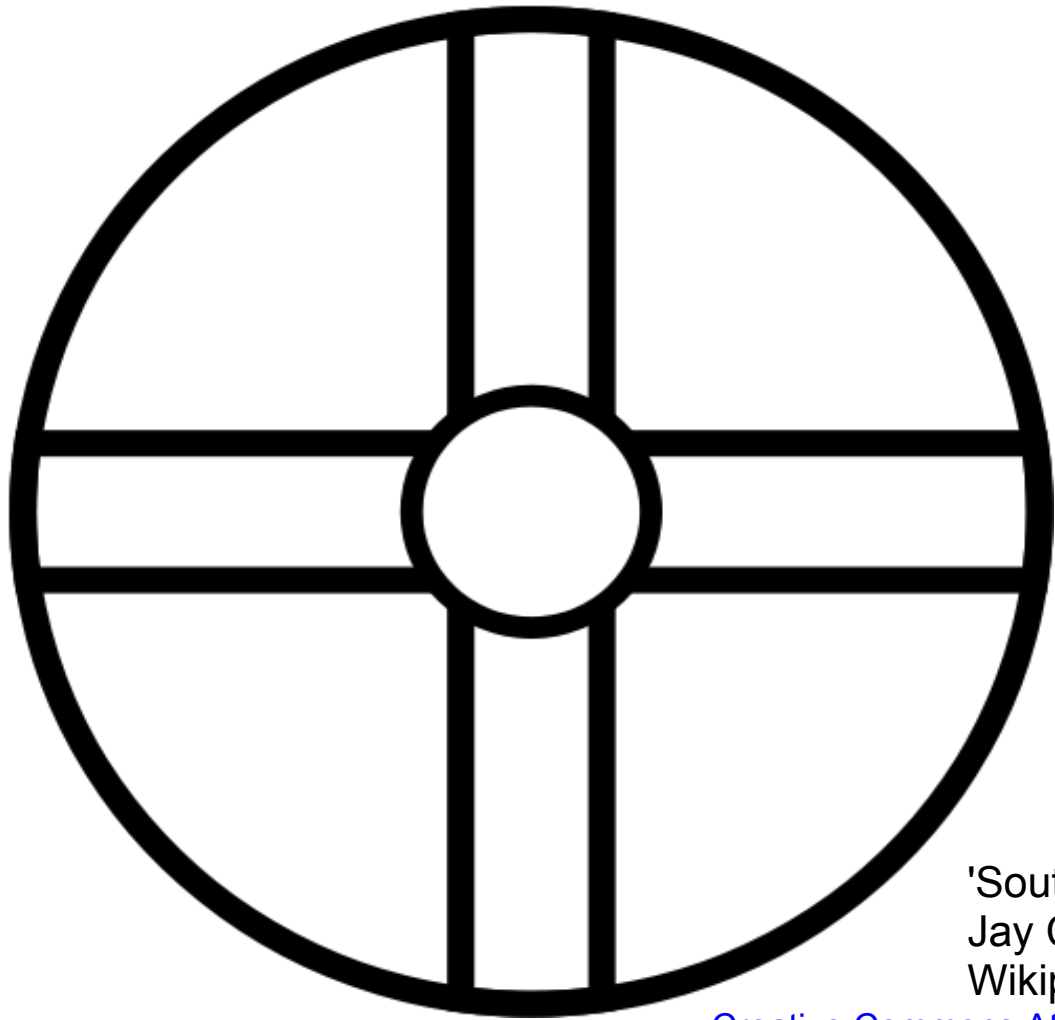
Agenda

Why does Django Rock?

How can we rock with Django?

Django rocks . . .

because it makes maintenance and development of web applications less esoteric.



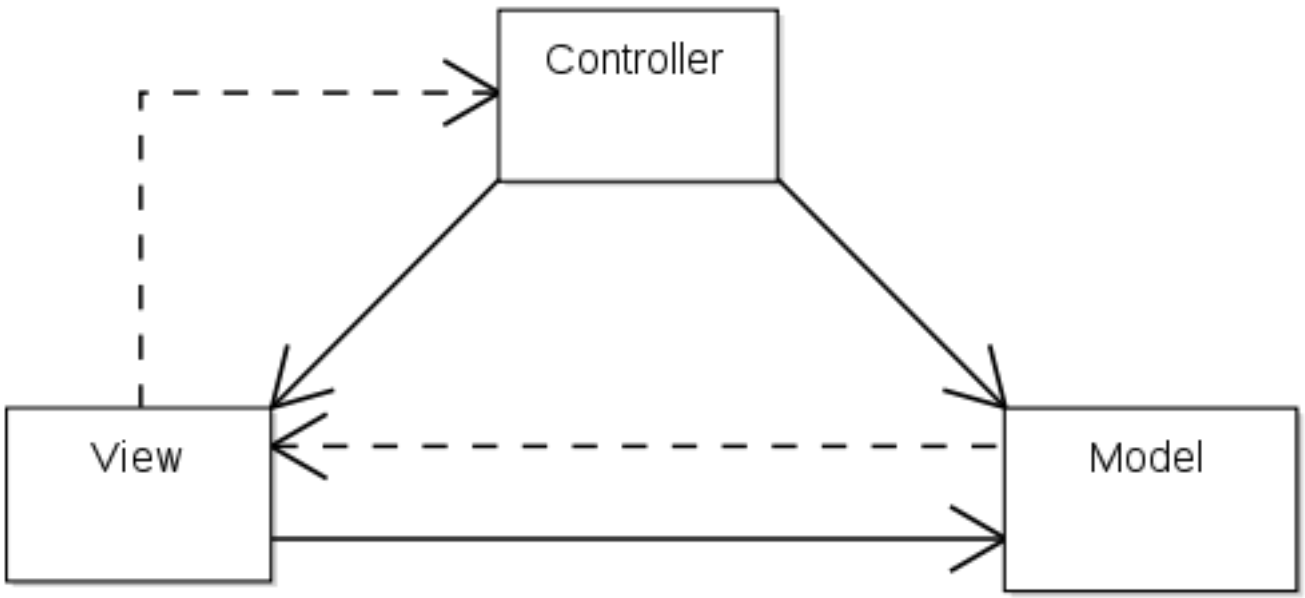
'Southern Cult Solar Cross', Jay Carriker, JCarriker, Wikipedia.

Django rocks . . .

because programmers and developers actually want to learn to use the framework.

Django rocks . . .

because it implements the Model-View-Controller Application Design Pattern, sanely.



Django rocks . . .

because their documentation rocks.

- grows as a single versioned entity.
- clearly states the version of Django that the page references.
- contains their design philosophy, best practices, new features in that version.
- Django Book.

Django rocks . . .

because it decreases the amount of useless knowledge in my head.

David J. Fred, Dfred, Wikipedia.

How to rock with Django

Porting Apps.

No cycle of Learning -> Forgetting -> Relearning.

No re-write of the entire application.
Minimal changes to the existing database.

Django is CRUD +

Create new records

Read existing records

Update existing records

Delete existing records

Browse List

Detailed Record View

Sortable Columns

Pagination

Simple Search

Port to Django?

1. Projects with simple Authorization requirements.
2. Projects with simple workflow.
3. Projects with simple database schemas.

*Django is great as a base for complex applications. Complexity will require code.

Pacific Coast Architecture Database

1. Simple authentication and authorization needs.

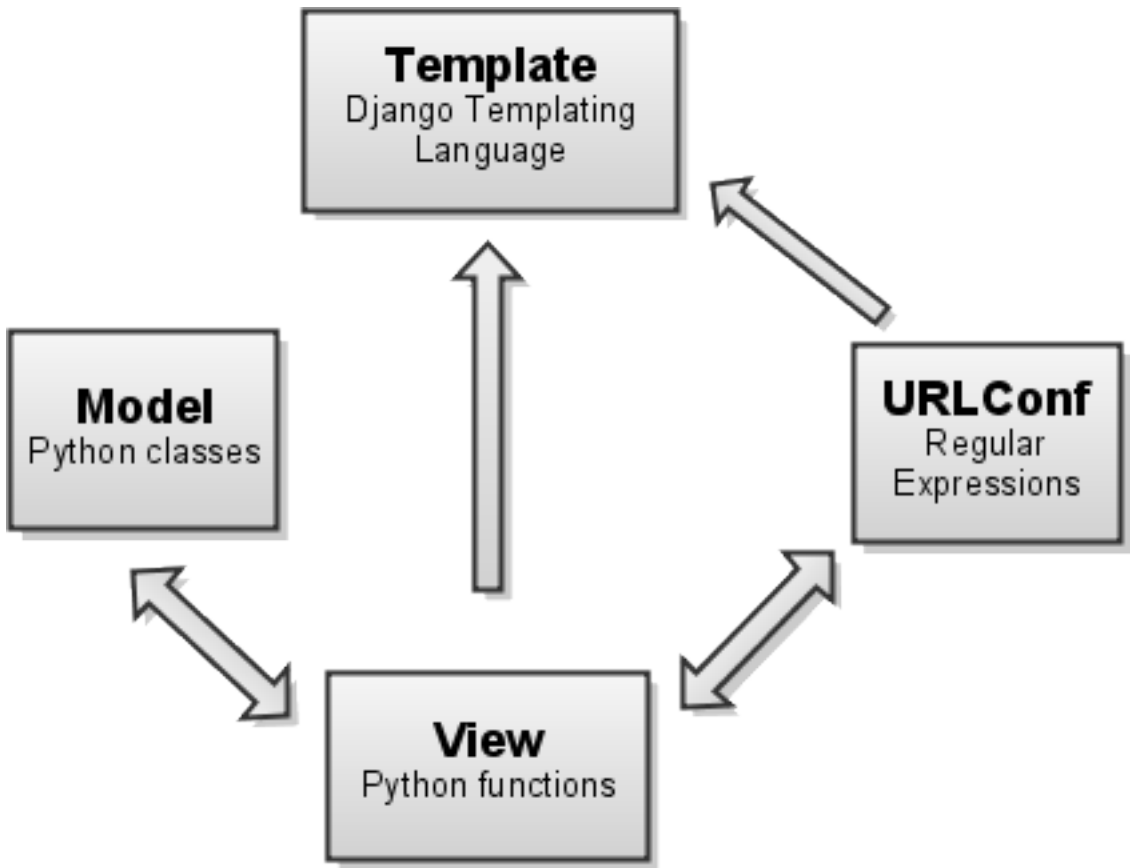
2. Frequent requests for changes to data fields
3. Frequent requests for additional data fields.

4. Frequent requests for new features.

5. Fairly simple database schema.

MTV

Model - Template - View (and URLConf)



Google Sites - Task List

Task List

Add item | Customize this list

Showing **91** items

Owner	Category	Description	Resolution	Completion Date	Milestone Target
Sort	Sort	Sort	Sort	Sort	Sort
Anj	CSS	Tab Navigation from listamatic			3
Anj	CSS	jQuery Image rotator of new public layout			3
Anj	Views	Advanced Searching			3
Anj	Admin	Add city to multiple select display of tbllocations	In models change def __unicode__(self):	June 16, 2009	2
Anj	Admin	Add middlename and namesuffix to multiple select of tblpartners	In models change def __unicode__(self):	June 16, 2009	2
Anj	Admin	Change tlbstructure filter from entrydate to editdate		June 16, 2009	2
Anj	Admin	Add id # to tblstructures browse list		June 16, 2009	2
Anj	Admin	Add publication id number to tblpublications browse		June 16, 2009	2
Anj	DB Cleanup	CDM images need scaling info added to the url.	update tblimages set url = (url || '&DMSCALE=80& DMWIDTH=0& DMHEIGHT=0&DMX=0& DMY=0')	June 22, 2009	2
Anj	Models	Change tblstructures:assessor# from textfield to charfield		June 16, 2009	2
Anj	Models	Admin throws error about datetime.date for tblpas	Change DateTimeField to DateField		2
Anj	Models	structures do not show up on admin or public images detail page.	Added m2m Tblstructures field to Tblimages.	June 19, 2009	2

Creating the Models

1. Clean the existing database.
2. Run inspectdb - creates models.py
3. Clean resulting models.py file.
4. Run syncdb - creates rest of the db tables.

```
$ python manage.py inspectdb > models.py
```

inspectdb is a Django tool that roughly translates an existing database into Django Models.

Cleaning up the Database

Before running inspectdb:

1. All Primary Keys are named 'id'.
2. All Foreign Keys are named '[tablename]_id'
3. If using Postgresql, name all sequences: '[tablename]_id_seq'

Cleaning up models.py

After running inspectdb:

1. PK field types from IntegerField to AutoField.

2. Re-Order Classes so FK's refer to Classes that already exist.

3. Add missing FK's and ManyToMany Fields.

As a general rule, place FK's in the Classes/Objects in which they will most logically be found.

Django rocks . . .

because it creates all the CRUD for you.

Create - Detailed Record Form

Read - Browse List + Detailed Record

Update - Populated Detailed Record Form

Delete - Browse List + Delete Form

Setup CRUD interface

Install and configure the Administration module.

1. Edit settings.py to install.
2. Edit urls.py. Uncomment a couple of lines.
3. Create a file named admin.py

Django rocks because of this admin interface.

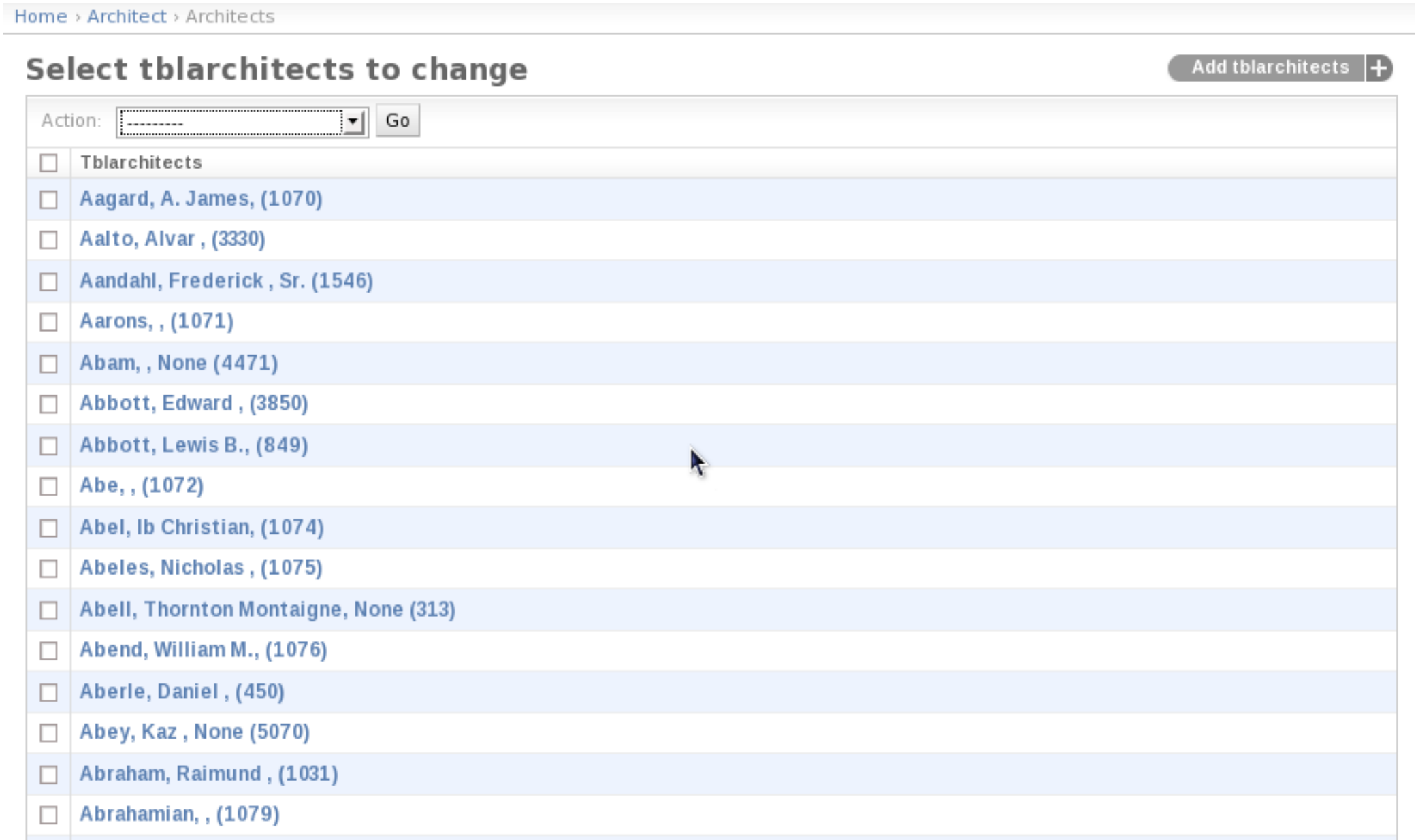
- Almost no programmer intervention needed.
- No Web developer or designer intervention needed.

admin.py - Plain

```
admin.site.register(Architect)
```

* Architect is the name of the Model

Admin Module: Default Browse List

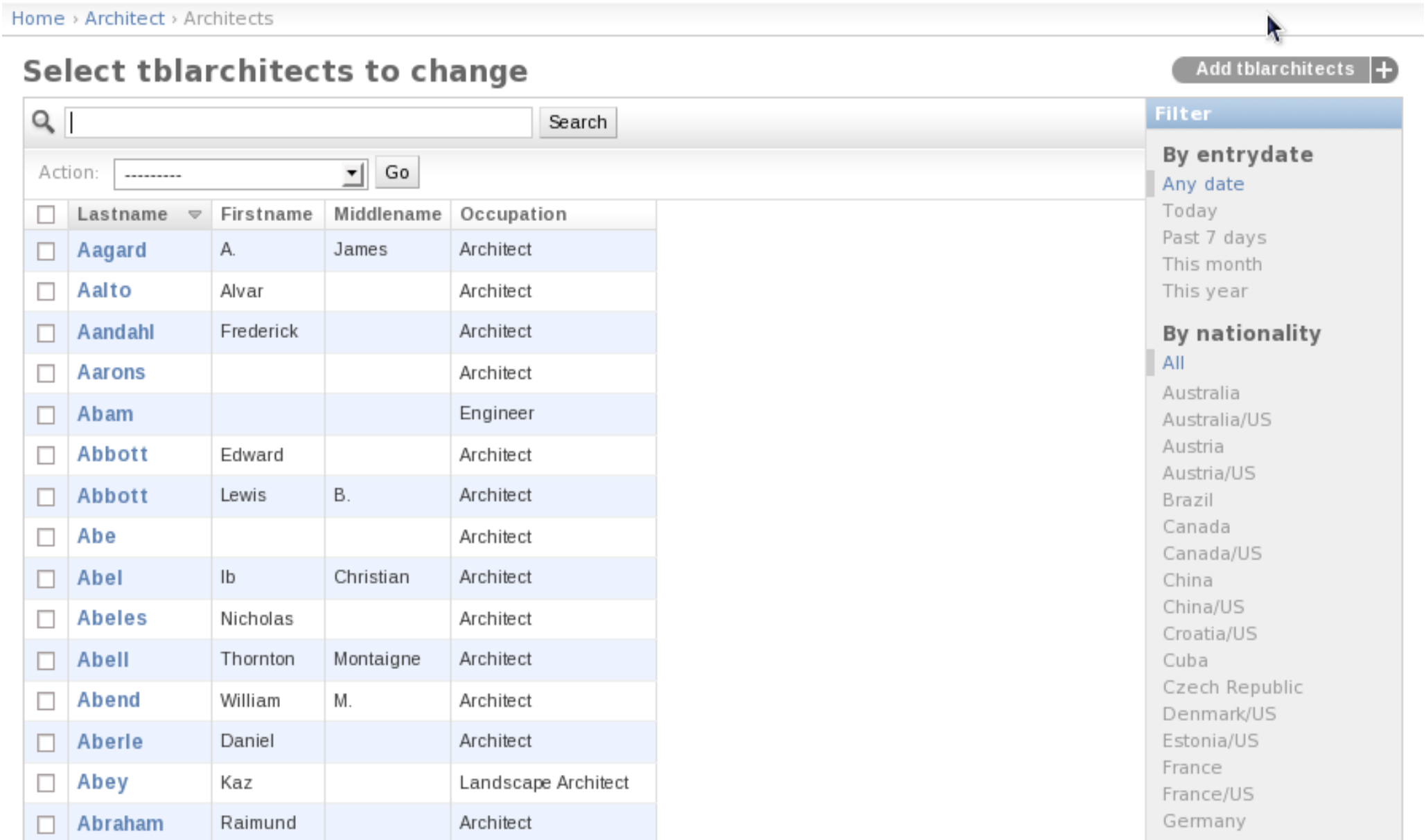


admin.py - Fancy

```
class ArchitectAdmin(admin.ModelAdmin):
    search_fields = ['lastname', 'firstname']
    list_display = ('lastname', 'firstname', 'middlename',
                            'occupation')
    ordering = ('lastname', 'firstname')
    list_filter = ('entrydate', 'nationality', 'occupation')


admin.site.register(Architect, ArchitectAdmin)
```

Admin Module: Fancy Browse List



Creating Public Views

Generic Views "provide easy interfaces to perform the most common tasks developers encounter."

1. Redirect to a different page or render a given template.

2. List and Detail pages for objects.

3. Present date-based objects in year/month/day archive pages, associated detail, and “latest” pages.

http://www.djangobook.com.

Example of simple Generic View

```
architect_list_info = {
    'queryset': Architect.objects.all(),
    }
```

Detailed Record View

Pacific Coast Architecture Database (PCAD)

- Architects
- Structures
- Images
- Partners
- Exhibits
- Publications

Architects

Adler, Jonathan

ID:	4903
Full Name:	Jonathan Adler
Occupation:	Interior Designer
Biographical Information:	Work History: Principal, Jonathan Adler Enterprises, LLC, New York, NY.
Countries:	United States
Structures:	2133 Fillmore Street Store, San Francisco, CA - (11857) 8125 Melrose Avenue Store, Los Angeles, CA - (11856) Parker Palm Springs, Palm Springs, CA - (11855)
Partners:	Adler, Jonathan, Enterprises, LLC (3619)

Extended Generic View Bonus

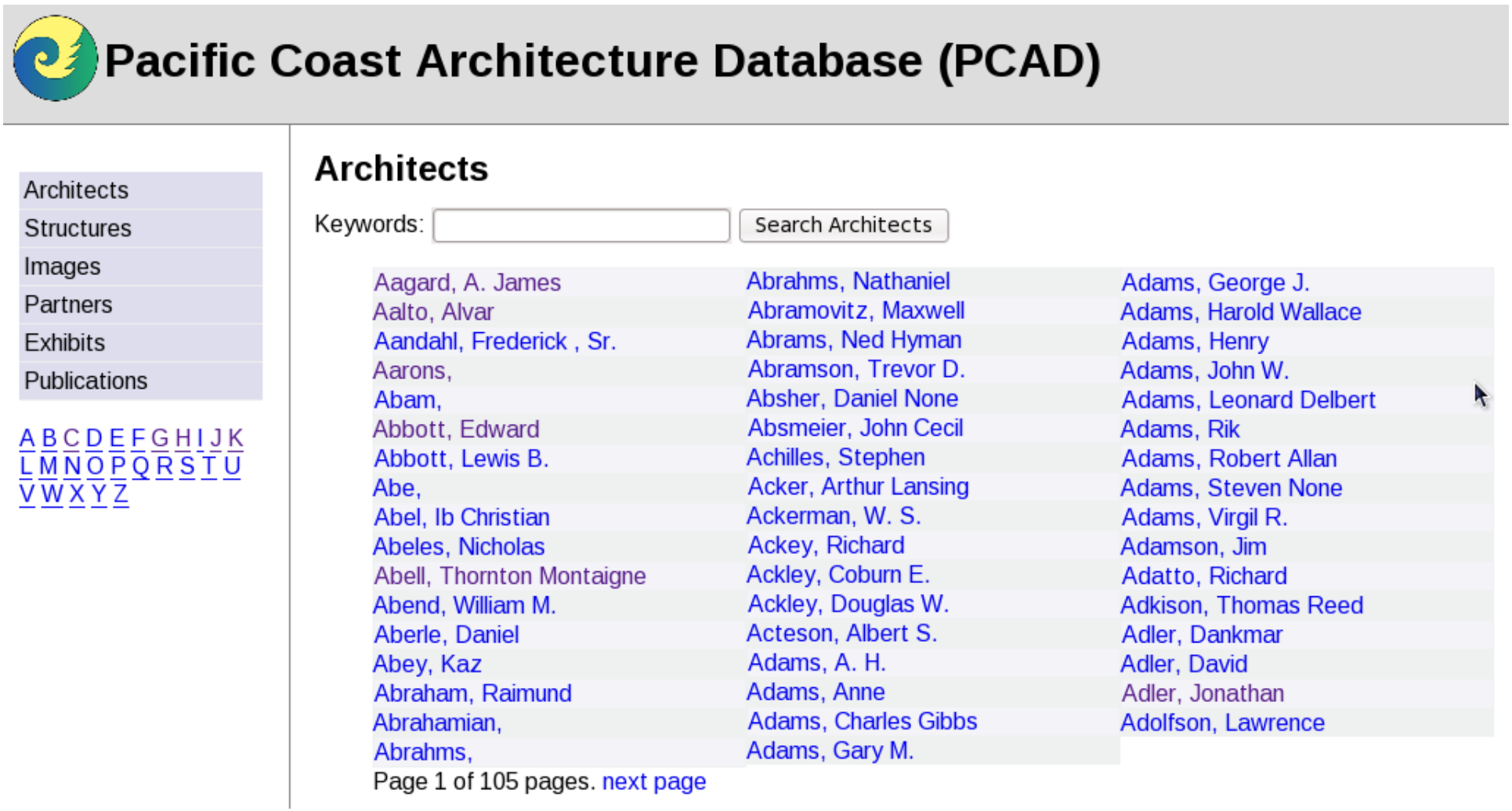
Addable parameters to Generic Views.

allow_empty
context_processors
extra_context
mimetype
paginate_by
queryset
template_loader
template_name
template_object_name

Example of extended Generic View

```
architect_list_info = {
    'queryset': Architect.objects.all(),
    'paginate_by': 50,
    'extra_context': {'alphabet': map(chr, range(65, 91))}
    }
```

Browse List w/extended Generic View Goodness



View example

```
def architects_alpha_list(request, alpha):
    architect_alpha_list = Architect.objects.filter(lastname__startswith = alpha)
    return object_list(request, queryset=architect_alpha_list,
                       paginate_by=50,
                       extra_context={'alphabet':map(chr, range(65, 91))})
```

Django rocks . . .

because it provides Generic Views for Templates.

Browse Lists
Detailed Records
Pagination
Easy customization

Create Public Pages: Templates

The Django templating language: HTML+

1. Recognizes all the basic HTML tags.
2. Allows for inclusions and extensions of templates.
3. Contains a large selection of built-in filters for modifying the output of variables and methods
4. Contains programmatic constructs for presentation, but is not actual python.

List cribbed from Jeff Croft.

http://jeffcroft.com/blog/2007/jan/11/django-and-mtv/

Django Template

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

{% block extrahead %}
<script src="/path/to/query.js" type="text/javascript"></script>
{% endblock %}

{% block title %}PCAD{% endblock %}
```

Django Template - Generic View

```
{% block content %}

<ul>
  {% for a in object_list %}
      <li><a href="/architect/architects/{{ a.id }}">
              {{a.lastname }}, {{a.firstname}}</a>
      </li>
  {% endfor %}
</ul>
  {% else %}
      <p>No list of Architects available.</p>
  {% endif %}

{% endblock %}
```

Django Template - Pagination

```
{% if is_paginated %}
    Page {{ page_obj.number }} of {{ paginator.num_pages }} pages.

   {% if page_obj.has_previous %}
      <a href="./?page={{ page_obj.previous_page_number}}">
          previous page</a> |
   {% endif %}

   {% if page_obj.has_next %}
      <a href="./?page={{ page_obj.next_page_number }}">
          next page</a>
   {% endif %}

{% endif %}
```

Other Projects

Myanmar: SQL Server + ASP -> Postgresql + Django

Sung Dynasty: SQL Server + ASP -> Postgresql + Django

Korean TOC: Flat file + Perl -> Postgresql + Django

Digital Registry Form: SQL Server + ASP -> SQL Server + Django

eReserves form: SQL Server + ASP -> SQL Server + Django

Why does Django Rock?

Django saves time and resources by enabling the delegation of tasks and generalizing the knowledge needed to work with web applications.

Django encourages good practices through it's design.

Django provides great documentation so you don't have to.

How can we rock with Django?

By using the framework for new web application development
By porting old applications to the framework.

1. Clean up the database.
2. Run inspectdb to create models.py
3. Edit the Models (the python classes) so that syncdb will actually work.
4. Install and configure the Administrative module.
5. Use Generic Views and Extended Generic Views for public views (to pipe contexts to the Templates.)
6. Install and use Flatpages for static content pages.

Bibliography

Jeff Croft, "Django and MTV," http://jeffcroft.com/blog/2007/jan/11/django-and-mtv/

"Django appears to be a MVC framework, but you call the Controller the “view”, and the View the “template”. How come you don’t use the standard names?", Django Documentation: FAQ, http://docs.djangoproject.com/en/dev/faq/general/
"MTV Development Pattern," Django Book, 2ed., http://www.djangobook.com/en/beta/chapter05/#cn16

"Design Philosophies," Django Documentation, http://docs.djangoproject.com/en/dev/misc/design-philosophies/


django
The web framework for
ponies with magical powers.