Pylons

A Modern Python Web Framework

www.pylonshq.com

James Gardner

EuroPython 3rd July 2006

www.3aims.com

Pylons is Rails-Like

- Lot of frameworks are compared to rails
- Pylons Routes and WebHelpers packages were ported straight from Rails so if you've used Rails, Pylons will be very familiar
- Pylons uses very similar principles to Django and TurboGears so if you've used those it will be familiar too
- Pylons is opinionated but lets you disagree

This talk is about Pylons 0.9

ingredients

0.9 Ingredients

- Automatic Installer
  - Easy install and Setuptools
- Automatic Generation of Starter Template
  - Paste Script
  - paster create –template=pylons ProjectName

- Stand-alone Server
  - Paste
  - paster serve development.ini
- Many Deployment Options
  - Provided by WSGI: CGI/FastCGI/SCGI/AJP/mod_python/Windows Services/IIS/etc...
- Clean and Simple URL Mapping
  - Routes (direct port from Rails)

- Sessions and Caching
  - Beaker (a WSGI refactor of some Myghty components)
- Templating
  - Myghty/Kid/Cheetah (via Pylons Buffet)
- Helper Functions and AJAX
  - WebHelpers
- JSON
  - simplejson and a Pylons decorator

- Database Integration
  - SQLObject, SQLAlchemy, anything else
- Interactive Debugging & Traceback Emails
  - Paste
- Unit Testing
  - nose
- Setup & configuration
  - Paste

- Documentation
  - Buildutils and Pudge
- Internationalisaton
  - Python gettext
- Error documents
  - Paste

everything
you need

Pylons Extensions

- Trivial to add middleware such as OpenID Authentication, authenticated session tickets, error handling and many others
- Trivial to add more helper functions
- You can add new project templates using paste script
- You can easily add different template engines

Arguably the most flexible framework

getting started

Installation

- Stable version
  - easy_install -f http://pylonshq.com/download/ "Pylons[test,pudge,kid,cheetah]==0.8.2"
- Living on the edge
  - easy_install -U http://pylonshq.com/svn/Pylons/trunk

If you want to understand Pylons, install it for yourself and explore the project directories

Creating a New Project

Pylons makes heavy use of Paste

> paster create -template=pylons helloworld

```
Selected and implied templates:
  Pylons#pylons  Pylons application template

Variables:
  egg_info_dir:  helloworld.egg-info
  package:       helloworld
  project:       helloworld
Creating template pylons
Creating directory .\helloworld
  Recursing into +egg_info_dir+
    Creating .\helloworld\helloworld.egg-info/
...
```

Giving it a go

> paster serve development.ini

Starting subprocess with file monitor
Starting server in PID 2068.
serving on 0.0.0.0:5000
....

You should see the file displayed in the public/html directory

demo

1. Hello Wold
Paster create
Paster serve
Default page

Pylons Basics

Routes (we'll use this in the next demo)

```
map.connect(':controller/:action/:id')
```

```
from helloworld.lib.base import *

class PeopleController(BaseController):
    def view(self, id):
        return response('Person is %s'%id)
```

Pylons objects (use Paste registry)

```
c,  h, request, response, session
 render, render_response
```

Getting Form Variables

```
def index(self):
    return response(
        'The value is  '+request.params['name']
    )
```

Hello World!

```
> paster controller hello
Creating ... helloworld\controllers\hello.py
Creating ... helloworld\tests\functional\test_hello.py
```

```
from helloworld.lib.base import *
class HelloController(BaseController):
    def index(self):
        return response('Hello World')
```

Using a Template

controllers/hello.py

```
class HelloController(BaseController):
    def index(self):
        return response('Hello World')
    def serverinfo(self):
        c.name = 'The Black Knight'
        return render_response('/serverinfo.myt')
```

Using a Template

templates/serverinfo.myt

```
<p>Hi, here's the server environment: <br />
<% str(request.environ) %></p>

<p>here's the URL you called: <% h.url_for() %></p>

<p>and here's the name you set: <% c.name %></p>
```

demo

2. Hello Wold
hello/index
hello/serverinfo
map.connect('', controller='hello', action='index')

Testing

tests/functional/test_hello.py

```
from helloworld.tests import *

class TestHelloController(TestController):
    def test_index(self):
        response = self.app.get(url_for(controller='hello'))
        # Test response...
        assert 'Hello' in response
```

```
> easy_install nose
> nosetests
```

Shell

> paster shell

Pylons Interactive Shell

...

```
>>> h.url_for(controller='hello')
'/'
>>> app.get('/')
<Response 200 OK 'Hello World'>
>>> assert 'Hello' in _
>>> assert 'Goodbye' in _
Traceback (most recent call last):
  File "<console>", line 1, in ?
AssertionError
```

full example

Rails Flickr – On Pylons!

http://www.rubyonrails.org/screencasts

Flickr is an online photo application

```
paster create --template=pylons imagesearch
cd imagesearch
paster controller flickr
cd imagesearch/lib
wget http://jamesclarke.info/projects/flickr/flickr.py
mv flickr.py flickrapi.py
cd ../../
paster serve --reload development.ini
```

Setup the JavaScripts and autohandler

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
        "http://www.w3.org/TR/html4/strict.dtd">

<html>
<head>
<title>Flickr!</title>
<% h.javascript_include_tag('/javascripts/effects.js', builtins=True)
    %>
<% h.stylesheet_link_tag('/flickr.css') %>
</head>
<body>
% m.call_next()
</body>
</html>
```

Write the controller

```
from imagesearch.lib.base import *
import flickrapi
flickrapi.API_KEY = "Your key here!"

class FlickrController(BaseController):
    def index(self):
        return render_response('/flickr.myt')

    def search(self):
        photos = flickrapi.photos_search(
            tags=request.params['tags'],
            per_page=24
        )
        c.photos = [photo.getURL(size="Small", urlType='source') \
           for photo in photos ]
        return render_response('photos.myt')
```

Main Template

```
<% h.form_remote_tag(url=h.url(action="search"),
      update="photos",
      complete=h.visual_effect("Blind_down", "photos"),
      loading=h.update_element_function("spinner",
         content="loading.."),
      loaded=h.update_element_function("spinner", content=""))
   %>
<div id="spinner"></div>
<fieldset>
       <label for="tags">Tags:</label>
        <% h.text_field("tags") %>
        <% h.submit("Find") %>
</fieldset>
<div id="photos" style="dislay:none"></div>
<% h.end_form() %>
```

Template Fragment

```
% for photo in c.photos:
        <img class="photo" src="<% photo %>">
% #end for
```

CSS

Same as the Rails CSS file!

Routes

We'll use the default

Quick Recap

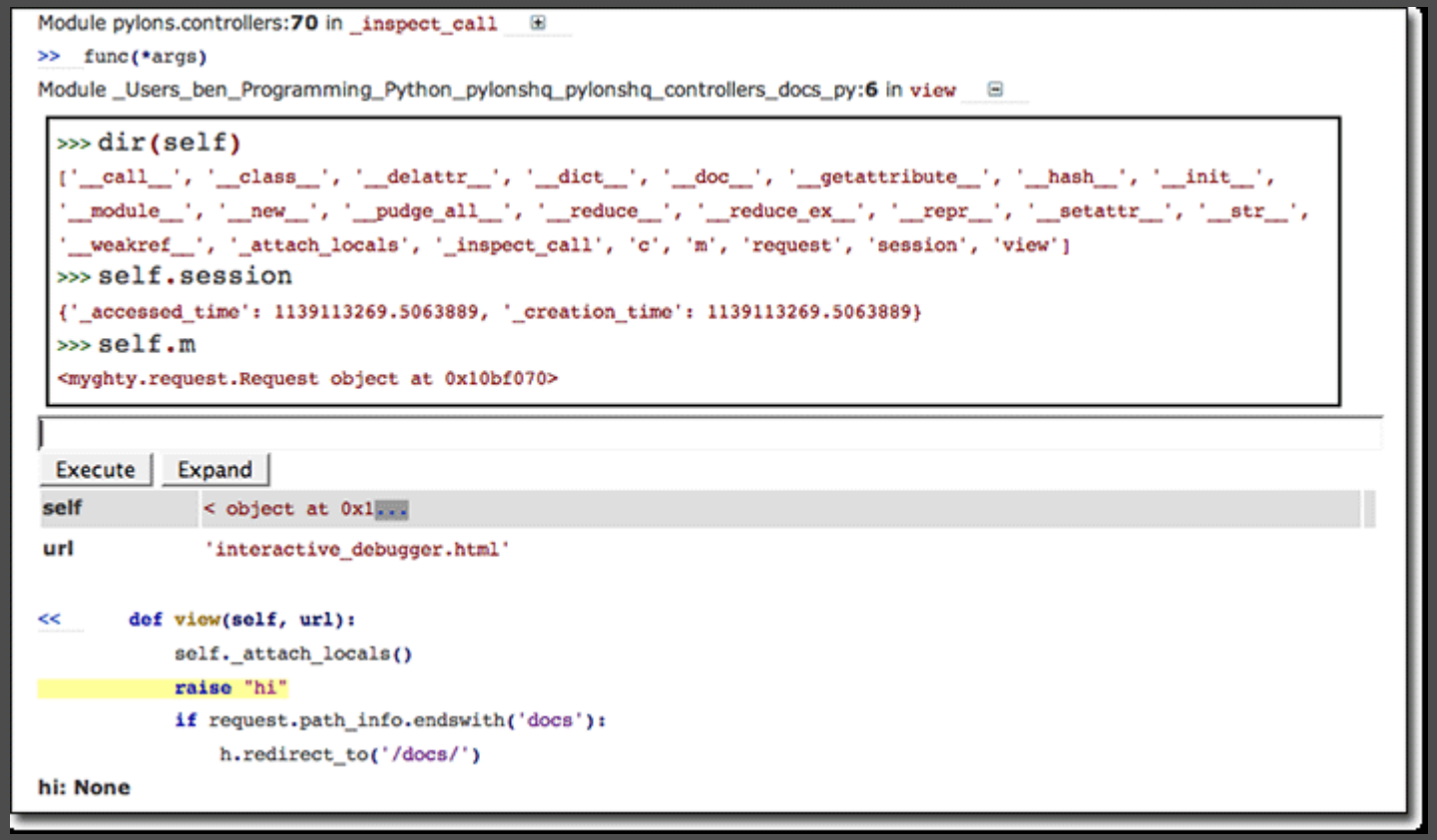
- Installed a Flickr library
- Written a controller with index() and search() methods
- Written a main template linking to the JavaScripts we need
- Created a template fragment to generate HTML to return to the browser via AJAX
- Added the necessary CSS

demo

flickr

Interactive Debugging

- Massively speeds up debugging
- One example of how middleware really helps



demo

Why Should You Use Pylons For Your Next Project?

- Really simple request mechanism
- Solid platform for expansion (eg with helpers, WSGI middleware etc)
- Flexible, eg template choice, model choice, middleware choice
- Source code is well documented and easy to read
- If you know Python you'll find Pylons natural and easy

What We Haven't Seen

- Using SQLAlchemy as a model
- Using Paste, Easy Install and web_setup.py to package and distribute an application for easy configuration by your users
- Building forms with FormBuild

Find out at the Web Framework Shootout Tomorrow

questions?

james@pythonweb.org

thank
you