Use Pyramid Like a Pro

Keith Yang

@keitheis

Pyramid™

- WEB APPLICATION FRAMEWORK -

Keith Yang

@keitheis

- architect.py in biideal
- Organizer/Coordinator in Taipei.Py
- Web Lead in PyCon Taiwan 2013

Some fun fact

- New EC Startup 🛒
- Re-built on Nov.2012, Apr. 2013 online
- Agile style 🪄
- Powered by Pyramid

Taipei.py

- Monthly Meetup
- to Weekly Meetup

Taipei.py 的奇幻漂流

Pyramid™

Core

Choice 👈
Like
Human Being

Being What? Alien?

1. Know your future
2. Know your scale
3. Know your tool sets
4. Know your team, goal, and time frames

from future import everything

Choose What? 👈

1. `Web Framework(s)`
2. `SQL` `NoSQL` `ORM` `Query Helper` `ALL REDIS?`
3. `Message Queues` and/or `Background Workers`
4. `Template` `Session` `API?` `RESTful` `RESTlike`

Too many to choose but usually or eventually you have to

Architecture,
Architecture,
Architecture.

That's why Flexibility Counts

The Power of Pyramid (Pylons Project)

1. Flexibility: from minimal to full stack `seriously ready`
2. Views config & varies routes
3. Add-ons and friends
4. Reliability, Agilibility, Simplicity and `Community`
5. `Fun` to learn

Python 3 Ready from Pyramid 1.3

What Pyramid is NOT

Not the first web framework

you should learn together with

your first programming language (e.g. Python).

Rumors

“Use NoSQL and you will be good with schema free.”

Imagine a type free programming language,

which `int(“0”) + str(1)` will be so fine.

Rumors (cont.)

“Why you should use this web framework”

Said, Django, Flask, Web2py, Bottle, Rails, ...

all the excellent choices.

Also known as **「不要欺騙小孩子」** 👆

課本上有寫「政黨體質的健全、民主、清廉與執政能力關係人民的幸福」

✖ 不解釋 。

I say:

“Just use Python”

Fundamental Pyramid School

- Understand Pyramid Request
- Separated logic of views, data and business logics

For any MVC-like Framework

M or C is not for business logic

Fundamental Pyramid School (cont.)

- Super powerful Views
- Flexible Routes
- Context driven

Database: to SQL

- Postgres, professional or enthusiastic
- MariaDB, new MySQL candidates, or lazy at first
- SQLite for development, personal app, or no production-level needed

Database: to NoSQL

- To name a few:
  - MongoDB, Riak, Cassandra, Redis, Hadoop, ...
- When you arrived there you'll learn these class.

  Before that, just keep calm on them.

Database: None

Perfect

No default binding ORM, no bother.

To ORM

- SQLAlchemy ORM
- Read What is Pony ORM?¶
- Django ORM (not really a choice here)
- Peewee Django ORM like
- MongoEngine, MongoKit, Ming, …

Have you tried Taiwan Mango?

Not to ORM

- SQLAlchemy Core
- MoSQL v0.2 by mosky, also on (yesterday v0.1.6)
- Pure SQL Dialects: hard code driver Query

  (sqlite3, psycopg2, mysql-python, ...)
- Pure NoSQL Dialects

Templates: Mako or Jinja2

- Mako Templates - pythonic, or not, depending on you with its nature

  by Mike Bayer (author of Alembic)

- Jinja2

  by Armin Ronacher, also Flask author

  1. Elegant
  2. Strict, more or less, magic

Templates (cont.) - Plim

1. by Maxim Avanov
2. Ported from Slim template
3. Clean HTML usage
4. Built on top of Mako Templates
5. Fun and half hell on debug mode

Guess what? You'll learn how to guess by using it 😐

Templates (cont.) - Plim

```
-for row in ['apple', 'banana', 'pineapple']
  tr
    -for col in ['juice', 'muffin', 'pie']
      td ${row.capitalize()} ${col}
```

That's all. No </blah blah things>.

Templates (cont.) - Chameleon

ZPT, Kid or Genshi lovers should give it a look

```
<tr tal:repeat="row 'apple', 'banana', 'pineapple'">
    <td tal:repeat="col 'juice', 'muffin', 'pie'">
         ${row.capitalize()} ${col}
    </td>
</tr>
```

Too many to introduce (again)

- Celery, Python RQ
- RebbitMQ, ZeroMQ
- Redis, memcached
- Beaker, retools (both by `Ben Bangert`, Pylons 1 author)

Oh, Sentry

- by David Cramer (our Keynote speaker)
- Awesome error events handle
- Host by yourself or getsentry.com

My better practices

- Separate the (DB) models and behaviors
- Reproduce-able development database (Fixtures)
- Development mode optimization
- Plim template

Double-edged swords ✄ use them carefully.

My better practices (cont.)

- Explicit architecture (lib, tasks, schemas ... folders) with explicit nameing
- Focus on a utc module handles datetime & timezone
- Fundamental doc: README and doc/**/*.rst
- Avoid mess sys or deploy stuff in code repository if possible

Minimal Viable Code </>

My worser lesson 1

```python
def ugly_customized_routes(config, resource_route,
    path=None,
    actions="ICRUD",
    resource_name=None):
    ...
```

Not aesthetic route helper

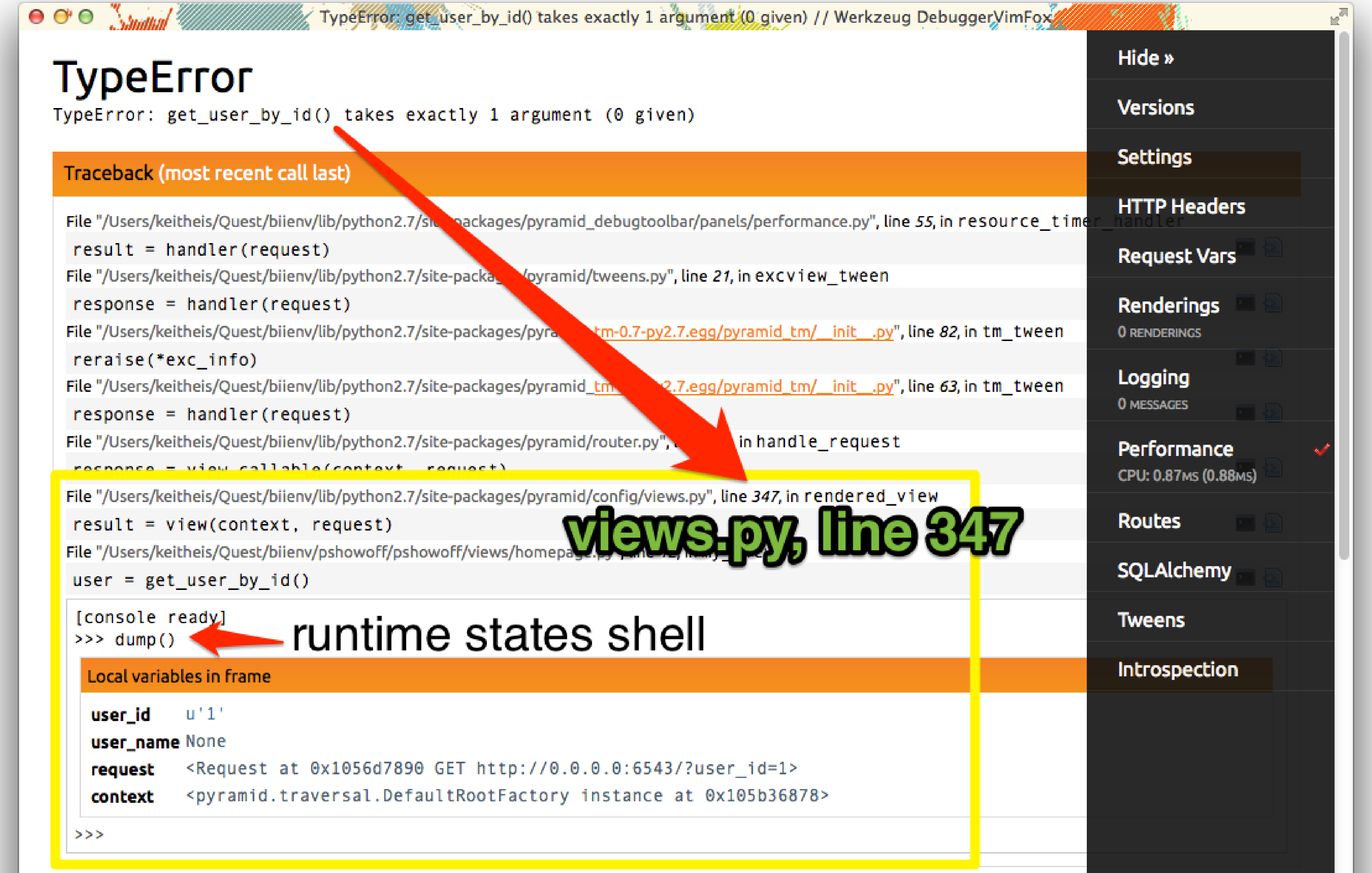
My worser lesson 2

```
print("This is too embarrassing to print")
```

Lacking CI tests 😐

Debug mode

```
if debug_notfound_route:
    from .views.notfound import (
      NotFoundDebugView as NotFoundView
    )
else:
    from .views.notfound import NotFoundView
config.add_notfound_view(NotFoundView)
```



TypeError: get_user_by_id() takes exactly 1 argument (0 given) // Werkzeug DebuggerVimFox

TypeError

```
TypeError: get_user_by_id() takes exactly 1 argument (0 given)
```

Traceback (most recent call last)

```
File "/Users/keitheis/Quest/biienv/lib/python2.7/si  packages/pyramid_debugtoolbar/panels/performance.py", line 55, in resource_timer_handler
 result = handler(request)
File "/Users/keitheis/Quest/biienv/lib/python2.7/site-packa  /pyramid/tweens.py", line 21, in excview_tween
 response = handler(request)
File "/Users/keitheis/Quest/biienv/lib/python2.7/site-packages/pyra  tm-0.7-py2.7.egg/pyramid_tm/__init__.py", line 82, in tm_tween
 reraise(*exc_info)
File "/Users/keitheis/Quest/biienv/lib/python2.7/site-packages/pyramid_tm  2.7.egg/pyramid_tm/__init__.py", line 63, in tm_tween
 response = handler(request)
File "/Users/keitheis/Quest/biienv/lib/python2.7/site-packages/pyramid/router.py",   in handle_request
 response = view_callable(context, request)
File "/Users/keitheis/Quest/biienv/lib/python2.7/site-packages/pyramid/config/views.py", line 347, in rendered_view
 result = view(context, request)
File "/Users/keitheis/Quest/biienv/pshowoff/pshowoff/views/homepa
 user = get_user_by_id()
```

views.py, line 347

```
[console ready]
>>> dump()
```

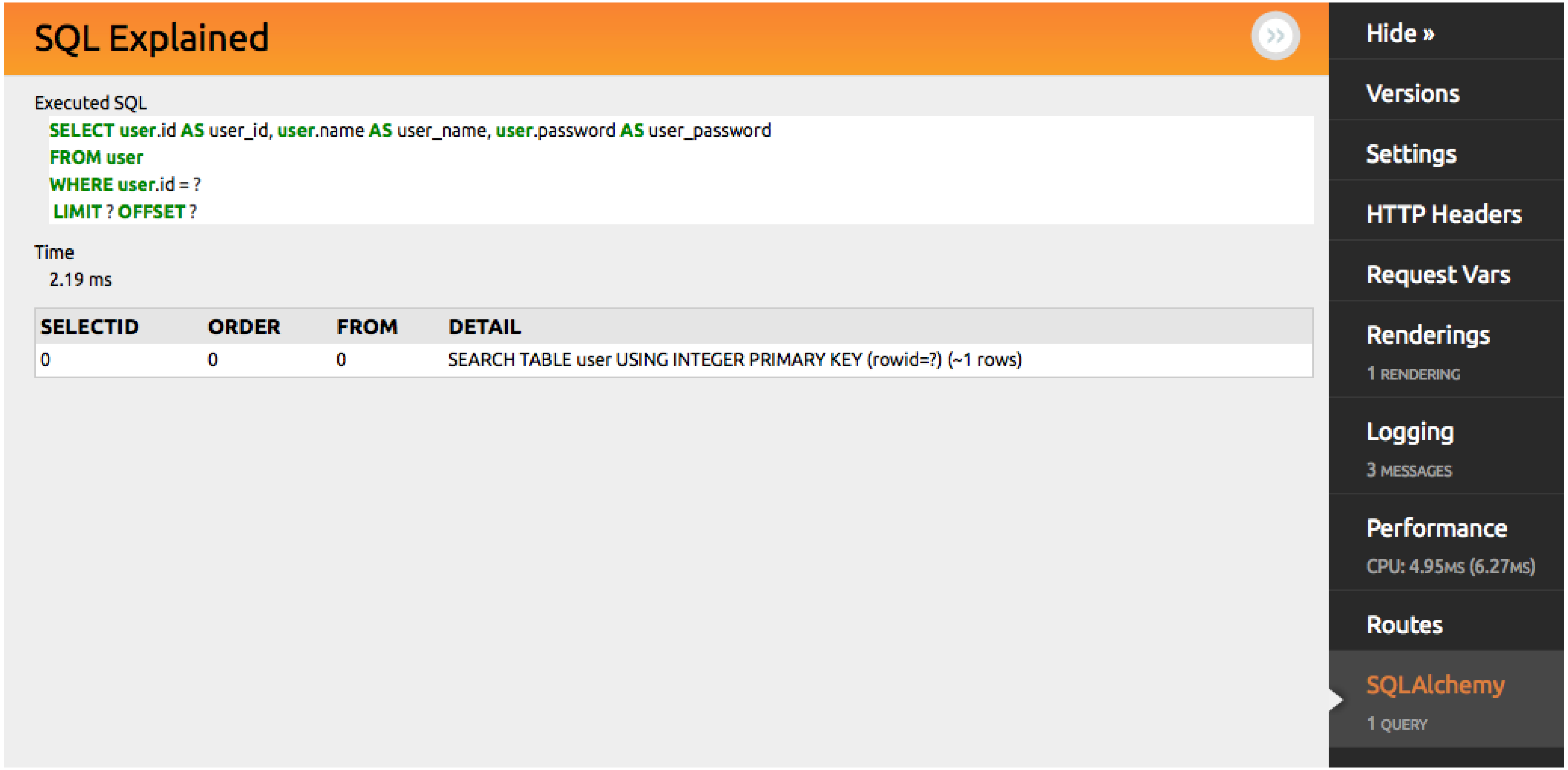
runtime states shell

Local variables in frame

user_id	u'1'
user_name	None
request	<Request at 0x1056d7890 GET http://0.0.0.0:6543/?user_id=1>
context	<pyramid.traversal.DefaultRootFactory instance at 0x105b36878>

```
>>>
```

SQL Master



SQL Explained

Executed SQL

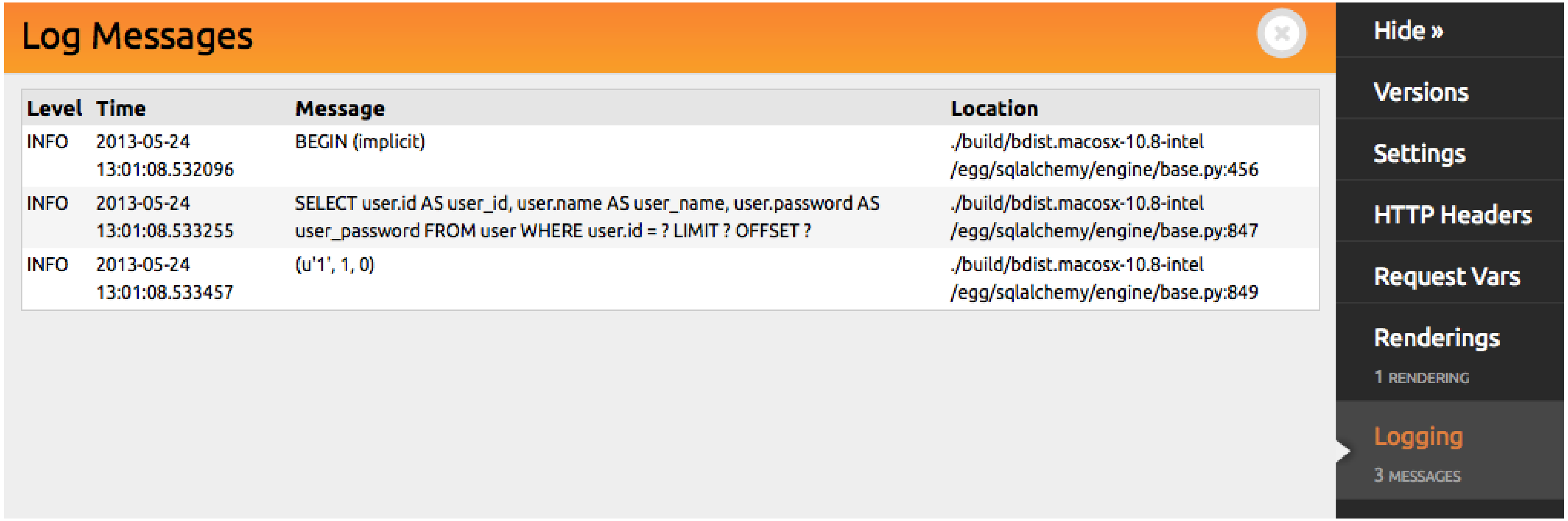
```
SELECT user.id AS user_id, user.name AS user_name, user.password AS user_password
FROM user
WHERE user.id = ?
 LIMIT ? OFFSET ?
```

Time

2.19 ms

SELECTID	ORDER	FROM	DETAIL
0	0	0	SEARCH TABLE user USING INTEGER PRIMARY KEY (rowid=?) (~1 rows)

Logging in the flow



Log Messages

Level	Time	Message	Location
INFO	2013-05-24 13:01:08.532096	BEGIN (implicit)	./build/bdist.macosx-10.8-intel /egg/sqlalchemy/engine/base.py:456
INFO	2013-05-24 13:01:08.533255	SELECT user.id AS user_id, user.name AS user_name, user.password AS user_password FROM user WHERE user.id = ? LIMIT ? OFFSET ?	./build/bdist.macosx-10.8-intel /egg/sqlalchemy/engine/base.py:847
INFO	2013-05-24 13:01:08.533457	(u'1', 1, 0)	./build/bdist.macosx-10.8-intel /egg/sqlalchemy/engine/base.py:849

Debug Debugtoolbar

```
# Disable development performance killer
debugtoolbar.panels =
    pyramid_debugtoolbar.panels.sqla.SQLADebugPanel
    ...
    #pyramid_debugtoolbar.panels.introspection.Introspection
    #pyramid_debugtoolbar.panels.tweens.TweensDebugPanel
    #pyramid_debugtoolbar.panels.routes.RoutesDebugPanel
```

Before your first release

>_ Practice

- pshell (the force on production, use it carefully)
- Play Command-Line Pyramid
- Alembic Database Migration scripts (if using SQLAlchemy)

Save
Your Self
In Time

Feel the power of Alembic

- by Mike Bayer (author of SQLAlchemy)
- Use op and sa instead your defined Models

```python
from alembic import op
import sqlalchemy as sa
```

Because YMWV (Your model WILL vary) between revisions.

```
alembic -c dev.ini
revision --autogenerate
-m "magic hour"
```

Encourage Refactor and Clean Code

- Feature development helps some
- Clean code, testing and refactor help everyone

From Web app to API

- Reusable permission control
- Customized API render (extra work on JSON response)

```
@view_defaults(renderer='api', permission='account')
class APIResourceView(APIBaseView, APIPagingMixin):
    ...
```

From Web app to API (cont.)

- Views for API Forbidden and NotFound

```
config.add_forbidden_view(
    APIForbiddenView, path_info='/api/')
config.add_notfound_view(
    APINotFoundView, path_info='/api/')
```

(no months or years needed for working)

API V2 Routes Ready

```
from .api.routes import api_routes
config.include(api_routes, '/api/v1')
```

```
from .api.v1.routes import api_routes as api_v1_routes
config.include(api_v1_routes, '/api/v1')
from .api.routes import api_routes as api_v2_routes
config.include(api_v2_routes, '/api/v2')
```

“ 哭吧你

現在每個人可獲得1 Pyramid。

更彈性。要怎樣就怎樣。

更多元。做不完的決定。

更多頁。為什麼不立即開始讀讀不完的文件呢？

flickr 梗

Much More

🐦 @pylons

2013 Taiwan

PYCON

Thank You 👍