Pyramid Royal Documentation

Release 0.8.9.dev0

Hadrien David

November 20, 2015

Royal is a pyramid extension which eases writing RESTful web applications.

It uses pyramid traversal algorithm rather than URL Dispatch as it offers a neat pattern to represent REST resources.

Documentation on read the docs

Source code on github

Overview

1.1 Traversal quick overview

> **Note:** Refer to request processing and traversal chapters in pyramid documentation for details.

Using traversal, requesting `/users/hadrien/photos/` is treated as dict accesses:

```
RootFactory(request)['users']['hadrien']['photos']
```

A root factory object configured using `pyramid.config.Configurator.set_root_factory()` is instantiated for each request.

A traverser function locates the context by traversing resources tree using any existing `__getitem__` on the root object and subobjects.

When any of the `__getitem__` call raises a `KeyError` exception or traverser reach the end of request's `PATH_INFO`, context is found.

Then pyramid's router looks up a view callable using the context.

1.2 Collections and Items

> **Note:** Refer to Creating a Pyramid Project for details on pyramid configuration.

A REST API is a tree of resources accessible via HTTP methods. Here is an example with 3 resources, *user*, *photo* and *user photo*:

```
root
-- photos            Collection of photos         /photos
|   -- {photo_id}    Photo item                   /photos/123/
-- users             Collection of users          /users/
    -- {user_id}     User item                    /users/hadrien/
       -- photos     Collection of user's photos  /users/hadrien/photos
```

Two ways to configure this tree of resources with royal:

1. **Imperative** using `config.add_resource`. By convention, directive will look for classes named `Collection` and/or `Item` in sub modules.

    • `example/resource/__init__.py:`

```python
    def includeme(config):
        config.add_resource('users')
        config.add_resource('users.photos')
        config.add_resource('photos')
```

- example/resource/users.py, example/resource/users_photos.py and example/resource/photos.py:

```python
import royal


class Collection(royal.Collection):
    pass


class Item(royal.Item):
    pass
```

2. **Declarative** using collection_config and item_config decorator:

```python
import royal

def incudeme(config):
    config.scan()


@royal.collection_config('users')
class Users(royal.Collection):

    def index(self, params):
        pass


@royal.item_config('users')
class User(royal.Item):

    def show(self, params):
        pass


@royal.collection_config('users.photos')
class UserPhotos(royal.Collection):
    pass


@royal.item_config('users.photos')
@royal.item_config('photos')
class Photo(royal.Item):
    pass


@royal.collection_config('photos')
class Photos(royal.Collection):
    pass
```

TBD...

Changelog

2.1 Development

2.2 0.8.8 (2015-11-03)

- Permit method override for GET.

2.3 0.8.7 (2015-08-05)

- Make DELETE method return 204 No Content

2.4 0.8.6 (2015-04-15)

- Make dummy testing resources derive from dict.

2.5 0.8.5 (2015-03-11)

- Fix `royal.testing`.

2.6 0.8.3 (2015-03-10)

- Breaking change: Remove `.parent` and `children` property on `royal.resource.Base`
- Add `find_item` and `find_collection`
- Update interfaces.

2.7 0.8.2 (2015-03-06)

- Fix a bug with 2 items being adjacent in hierarchy. The child item was not registered on the good parent.
- Add a default implementation in sqla extension for `Item.replace`.

2.8 0.8.1 (2015-03-05)

- Add sqlalchemy extension.

- Change the way links are added to resource representation.

- Fix issue #6: HTTP 500 error when using POST verb on Item resources.

- Breaking changes:

    - *royal.resource:Base* constructor now takes request as third positional argument.

    - *royal.resource:Base.resource_url* and *Base.url* method signatures change.

- Add *presources* script to print all resources declared with royal.

- Add declarative configuration decorators *collection_config* & *item_config*.

- No more utility class: resource configuration uses pyramid introspectables to store configuration rather than utility internal dict.

- Move *royal.utility* to *royal.directives*

2.9 0.7.4

- Add update_schema and replace_schema to be symetric with create_schema.

2.10 0.7.3

- Started some documentation.

- Set default root factory to `royal.resource.Root`

- Exception view which catches all exception is only added when `debug=false` in application settings.

2.11 0.7.2

- Depends on pyramid without any version spec.

2.12 0.7.1

- Add a default `multipart/form-data` deserializer.

- No more views on `voluptuous.MultipleInvalid` as `voluptuous` has been removed from dependencies.

2.13 0.7

- Breaking changes:

- – Query params are not anymore passed to resources methods via `**kwargs` but as a `dict`. TBD: use a `multidict`.

- – Working on return value of `Collection.create` method. TBD: Add adapters to permit multiple way of calling resource methods from view and adapt renderering

- Add `renderer_adapter` decorator to register adapter via `config.scan`.

- Add renderer adapter to configuration introspectables under *Renderer adapters* category.

- Add `add_deserializer` config directive and decorator `royal.deserializer_config` to add deserializers for specific content_type.

- Remove decorator `log_error_dict` in favor of pyramid_exclog extension.

2.14 0.6

- Breaking changes: TBD

- Move onctuous away in favor of voluptuous

- Added method tunneling to permit `PUT`, and `DELETE` via `POST` methods.

- TBD: adapt how request body is parsed.

2.15 0.5.1

- Add `Base.__getitem__` which gets children from `self.children`

2.16 0.5

- Breaking change: Pages parameters are not anymore page & page_size but offset and limit. It is more developer and db friendly.

2.17 0.2

- onctuous schema checking on resource creation.

2.18 0.1

- Initial version

- royal renderer able to return bson or json.

- royal.includeme adds royal renderer and views.

- royal.resource.PaginatedResult which permits Collection.index to return paginated results.

- royal.views with default views for Collection and Resource.

- CollectionView.index does automatic pagination.