confine-orm Documentation

Release 0.1

Marc Aymerich

Sep 27, 2017

Contents

Confine-orm (aka *CONFINE Object REST Mapper* or *CONFINE Object Resource Mapper*) is a high level Python library for easily interacting with CONFINE REST API using object oriented concepts.

CHAPTER 1

Installation

pip install confine-orm

CHAPTER 2

Design Considerations

The main goal of this library is to provide fast and easy access to CONFINE REST API: Open a Python interpreter and start interacting right away.

To achieve this goal we have borrowed some ideas from traditional SQL object relational mappers and applied them to an hypermedia-driven resource-oriented architecture (CONFINE).

This library has been heavily inspired by Django's ORM implementation; also using an Active Record like pattern, plus concurrency based on asynchronous non-blocking I/O and caching based on Identity Mapping.

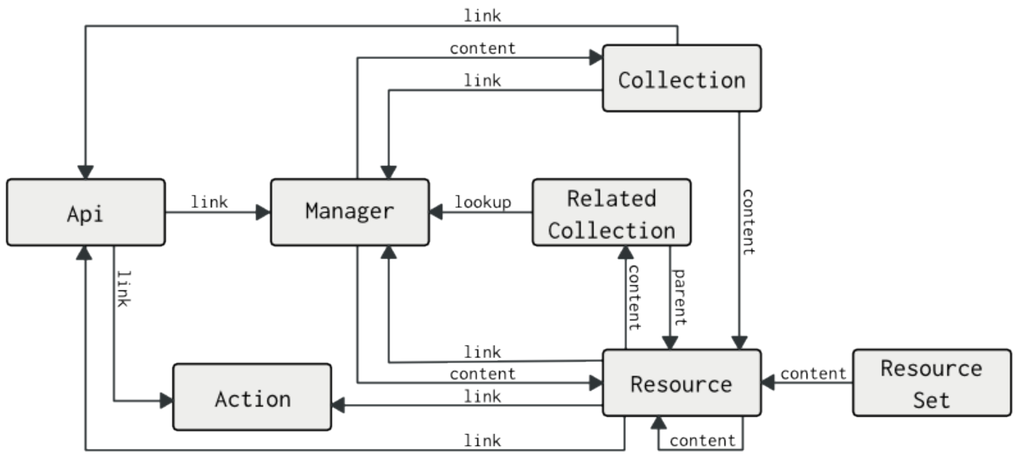
Confine-orm leverages the HATEOAS discoverability of CONFINE's API; rather than relying on a predefined knowledge, resources and methods are autodiscovered on the fly, while browsing the API.

Our hope is that you can use this library to produce brief, readable and efficient code in a fun and effortless way :)

CHAPTER 3

Object Oriented Modeling

The following diagram illustrates the classes used to model CONFINE REST API and how they relate to each other.



- `Api` is a good starting point for browsing. It represents the `Base` resource of an API and it has `links` to `Manager` and `Action`. These relations are discovered from the Link header of the HTTP response.
- `Manager` is used for accessing linked `Collections` and `Resources`.
- `Actions` are used to represent action-like endpoints (i.e. `do-get-auth-token`). Actions are special methods that are not expressed using HTTP verbs (`POST`, `GET`, etc).
- `Collection` is a list of uniform resources; all resources share the same media type (i.e. all registered nodes). For convinience it maintains a `link` back to its manager in order to proxy its methods.
- `Resource` is a local representation of a remote resource (i.e. a node), basically a `Resource` is an object with a URI. It may contain nested resources, or `RelatedCollections`.
- `RelatedCollection` is a subcollection that all its resources are related to the same `parent` (i.e. all slivers of a particular node). A `RelatedCollection` is able to construct a `lookup` for discovering its related

  `Manager`, therefore it is able to proxy its methods.

- `ResourceSet` is a set container that can be used to perform concurrent operations over a set of non-uniform resources.

CHAPTER 4

Interface Examples

```
>>> # Not much is needed
>>> from orm.api import Api
```

```
>>> # Api gives the entry point
>>> controller = Api('http://127.0.0.1/api/')
>>> controller.retrieve()
>>> print controller
{
    "testbed_params": {
        "mgmt_ipv6_prefix": "fd65:fc41:c50f::/48",
        "priv_ipv4_prefix_dflt": "192.168.241.0/24",
        "sliver_mac_prefix_dflt": "0x54c0"
    },
    "confine_params": {
        "priv_ipv6_prefix": "fdbd:e804:6aa9::/48",
        "debug_ipv6_prefix": "fd5f:eee5:e6ad::/48"
    },
    "uri": "http://127.0.0.1:8888/api/"
}
```

```
>>> # Linked collections can be easily accessed
>>> nodes = controller.nodes.retrieve()
```

```
>>> # Actions are autodiscovered and are callables
>>> controller.get_auth_token(username='vct', password='vct')
{"token": "0affff8bde9f13e0f807713616f173b1b4e2c98d"}
```

```
>>> # Filtering can go accross relationships
>>> nodes = nodes.filter(group__name='vct').exclude(id=12)
>>> nodes
[<orm.resources.Resource: http://127.0.0.1:8888/api/nodes/1413>,
 <orm.resources.Resource: http://127.0.0.1:8888/api/nodes/1414>]
```

```
>>> # Get a particular node from the collection
>>> node = nodes.get(id=1413)
>>> print node
{
    "sliver_pub_ipv6": "none",
    "sliver_pub_ipv4": "dhcp",
    "set_state": "debug",
    "group": {
        "uri": "http://127.0.0.1:8888/api/groups/2"
    },
    "sliver_mac_prefix": null,
    "description": "",
    "uri": "http://127.0.0.1:8888/api/nodes/1413",
    "mgmt_net": {
        "backend": "tinc_client",
        "tinc_client": {
            "island": null,
            "pubkey": null,
            "name": "node_1413"
        },
        "native": null,
        "addr": "fd65:fc41:c50f:585::2",
        "tinc_server": null
    },
    "priv_ipv4_prefix": null,
    "slivers": [
        {
            "uri": "http://127.0.0.1:8888/api/slivers/69"
        }
    ],
    "cert": null,
    "id": 1413,
    "local_iface": "eth0",
    "sliver_pub_ipv4_range": "#8",
    "boot_sn": 0,
    "arch": "i686",
    "properties": {},
    "direct_ifaces": [
        "eth1",
        "eth2"
    ],
    "name": "sadsadsad"
}
```

```
>>> # When relations are followed confine-orm automatically retrieves missing pieces
>>> node.group.uri
'http://127.0.0.1:8888/api/groups/2'
>>> node.group.name
'vct'
```

```
>>> # Authentication is easy
>>> node.reboot()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/home/vct/confine-orm/orm/managers.py", line 27, in __call__
    self.api.validate_response(response, valid_codes)
  File "/home/vct/confine-orm/orm/api.py", line 185, in validate_response
```

```
    raise self.ResponseStatusError(msg % context)
orm.api.ResponseStatusError: [http://127.0.0.1:8888/api/nodes/1413/ctl/reboot/]:
    403 FORBIDDEN (!= 200, 201, 202) Authentication credentials were not provided.
>>> controller.login(username='vct', password='vct')
>>> node.reboot()
{"detail": "Node instructed to reboot"}
```

```
>>> # Related collections automatically provide parent resource when needed, like on␣
↪create()
>>> slice = controller.slices.retrieve()[0]
>>> node.slivers.create(slice=slice)
<orm.resources.Resource: http://127.0.0.1:8888/api/slivers/70>
```

```
>>> # All methods over a list of resources perform concurrently
>>> node.slivers.update(set_state='registered')
```

```
>>> # Resources have save() and delete() methods
>>> node.name = 'ApeNode'
>>> node.save()
>>> node.delete()
```

```
>>> # and discovered actions as well
>>> slice.upload_overlay(open('/tmp/overlay.tgz', 'rb'))
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search