


# A primer on Falcon

**Francisco Ros**

Python Murcia

CEEIM, Murcia (Spain), 2016

# about.me

- CEO (= do a little bit of everything) at  **Doalitic**
- PhD in Computer Science
- Coding in Python for some time

 <https://es.linkedin.com/in/franciscojros>

 @fjrosmunoz

 fjros a-t doalitic d-o-t com

# Falcon

- Minimalist open source *web* framework
- Actually, it shines for REST APIs
- Created by Rackspace
- Python 2 and 3
- Not opinionated, you must decide what components (if any) to use: async framework, object-relational mapper, etc
- Easy to use
- High performance

<http://falconframework.org>

# Performance comparison

- CPython 2.7.9
- Multiple URIs
- Query string with percent-encoded characters
- Complex response headers

Framework	Test	req/sec	µs/req	Performance
Falcon (0.3.0)	Extended	14,384	70	5x
Bottle (0.12.8)	Simple	12,583	79	4x
Flask (0.10.1)	Simple	2,837	352	1x

## CPython 3.4.3

Framework	req/sec	µs/req	Performance
Falcon (0.3.0)	19,514	51	7x
Bottle (0.12.8)	10,748	93	4x
Werkzeug (0.10.4)	4,968	201	2x
Flask (0.10.1)	2,961	338	1x
Pecan (0.8.3)	2,763	362	1x

## PyPy 2.5.1

Framework	req/sec	µs/req	Performance
Falcon (0.3.0)	256,417	4	27x
Bottle (0.12.8)	145,411	7	15x
Werkzeug (0.10.4)	44,276	23	5x
Pecan (0.8.3)	16,031	62	2x
Flask (0.10.1)	9,471	106	1x

# Getting started

```
host$ git clone https://github.com/fjros/python_murcia_2016_falcon
host$ cd python_murcia_2016_falcon/
host$ vagrant up
host$ vagrant ssh
falcon$ PYTHONPATH=/vagrant:$PYTHONPATH gunicorn -b 0.0.0.0:8000 music:api
```

# REST 101

URI <i>resource</i>	GET <i>retrieve</i>	POST <i>create</i>	PUT/PATCH <i>update</i>	DELETE <i>remove</i>
/v1/albums	200 OK	201 Created 400 Bad Request		
/v1/albums/{album}	200 OK 404 Not Found			204 No Content 404 Not Found
/v1/songs?album=id	200 OK	201 Created 400 Bad Request		
/v1/songs/{song}	200 OK 404 Not Found			204 No Content 404 Not Found

```
{  
  "id": "album-2",  
  "author": "Gamma Ray",  
  "name": "Heading for tomorrow",  
  "year": 1990  
}
```

```
{  
  "id": "song-2",  
  "name": "Lust for life",  
  "album": "album-2"  
}
```

# Falcon basics

- API
- Routes, Sinks
- Resources
- Responders, Requests, Responses
- Exceptions
- Cookies

# see git repo



# Falcon middleware

- Simple yet powerful
- Useful for many purposes
  - Database sessions
  - Cache connections
  - Token verification/decoding
  - *many more*
- Similar to hooks *@falcon.before* and *@falcon.after*, but middleware applies globally to the entire API

# Example: database middleware

```
# music.py

from sqlalchemy import create_engine
from sqlalchemy.orm import scoped_session, sessionmaker
from middleware.database import SQLAlchemySessionManager

def setup_database_session_factory(uri):
    """Build a SQLAlchemy factory for scoped sessions.

    :param str uri: Database uri.
    :return: Factory for scoped sessions.
    :rtype: `sqlalchemy.orm.session.scoped_session`
    """

    engine = create_engine(uri)
    session_factory = scoped_session(sessionmaker())
    session_factory.configure(bind=engine)
    return session_factory

session_manager = SQLAlchemySessionManager('<your_database_uri>')
api = falcon.API(middleware=session_manager)
```

# Example: database middleware

```
# middleware/database.py
```

```
class SQLAlchemySessionManager:
```

```
    def __init__(self, session_factory):  
        self._session_factory = session_factory
```

```
    def process_request(self, req, resp):  
        req.context['session'] = self._session_factory()
```

```
    def process_response(self, req, resp, resource):  
        self._session_factory.remove()
```

# Falcon testing

```
class TestAlbum(testing.TestBase):

    albums_uri = '/v1/albums'

    def before(self):
        self.api = api

    def after(self):
        pass

    def test_retrieve_albums(self):
        body = self.simulate_request(method='GET', path=self.albums_uri, decode='utf-8')

        self.assertEqual(self.srmock.status, falcon.HTTP_200)
        self.assertIn('application/json', self.srmock.headers_dict['Content-Type'])
        self.assertIsNotNone(body)
        albums = json.loads(body)
        self.assertGreater(len(albums), 0)
        for album in albums:
            self.assertIsInstance(album['id'], str)
            self.assertIsInstance(album['name'], str)
            self.assertIsInstance(album['author'], str)
            self.assertIsInstance(album['year'], int)
```

# see git repo

# Falcon at Doalitic

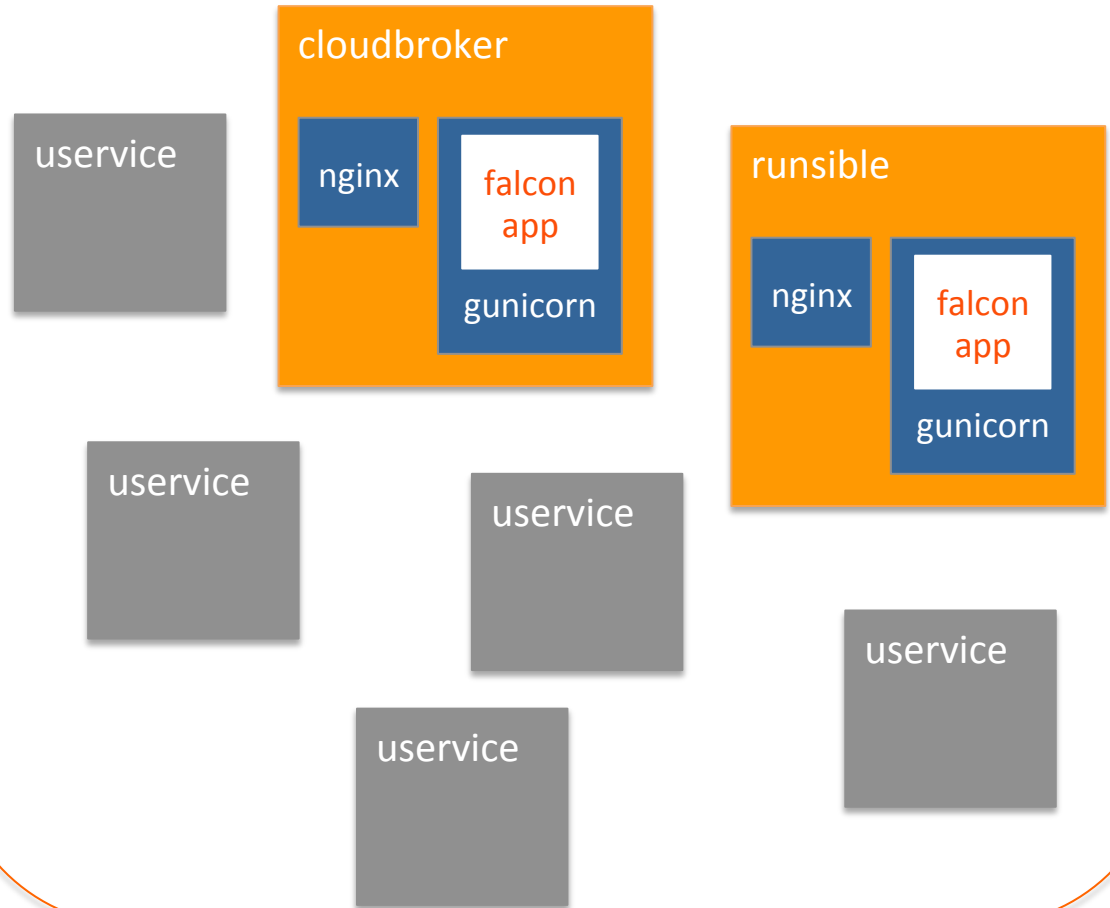
**Brook.io**

*next app*

*yet another app*

API

## Doalitic Core Platform



# Thanks!