Asyncio community one year later

EuroPython 2015, Bilbao

redhat

Victor Stinner
vstinner@redhat.com

Victor Stinner

- Python core developer since 5 years
- Senior Software Engineer at Red Hat
- Port OpenStack to Python 3
- Working remotely from South of France

Agenda

- Community
- Asyncio clients
- Asyncio servers
- Trollius
- Benchmarks
- How can you help?

Asyncio launch

- Python 3.4.0: March 2014
- Almost “naked”: very few libraries

Community

- python-tulip mailing list (Google Group)
- #asyncio IRC channel on Freenode
- Python bug tracker (bugs.python.org)
- More and more conferences!

aiohttp

- Most famous and successful library
- HTTP client and server
- HTTPS
- Client and server Websocket
- https://aiohttp.rtfd.org/

aiohttp client example

```
@asyncio.coroutine
def fetch_page(url):
    req = yield from aiohttp.request('GET', url)
    assert req.status == 200
    return (yield from req.read())
```

SQL drivers

- MySQL: aiomysql
- PostgreSQL: aiopg (based on psycopg2)

aiopg example

```
dsn = 'dbname=aiopg host=127.0.0.1 user=…'

@asyncio.coroutine
def go():
    pool = yield from aiopg.create_pool(dsn)
    with (yield from pool.cursor()) as cur:
        yield from cur.execute("SELECT 1")
        row = yield from cur.fetchone()
        assert row == (1,)
```

ORM

- peewee: peewee-async
- SQLAlchemy: aiopg.sa

Key-value stores

- memcached: aiomemcache
- redis: aioredis
- redis: asyncio-redis

aioredis example

```
@asyncio.coroutine
def wait_each_command():
   foo = yield from redis.get('foo')
   bar = yield from redis.incr('bar')
   return foo, bar
```

aioredis pipeline example

```
@asyncio.coroutine
def pipelined():
    get = redis.get('foo')
    incr = redis.incr('bar')
    foo, bar = yield from asyncio.gather(get,
                                         incr)
    return foo, bar
```

NoSQL

- CouchDB: aiocouchdb
- MongoDB: asyncio-mongodb

Clients

- DNS: aiodns (async resolver)
- IRC: bottom
- IRC: irc3
- SSH: AsyncSSH
- XMPP (Jabber): slixmpp

More clients

- AMI: panoramisk (AMI and FastAGI)
- AMQP: aioamqp
- ElasticSearch: aioes
- Etcd: aioetcd
- Google Hangouts: hangups

Web frameworks

- aiopyramid
- aiowsgi
- interest
- Muffin
- nacho
- Pulsar
- rainfall
- Vase

- API hour

Websockets

- aiohttp.web
- AutobahnPython
- websockets
- WebSocket-for-Python

redhat.

python™

Servers

- FastAGI (Asterisk): panoramisk
- IRC: irc3d
- HTTP: aiohttp
- SSH: AsyncSSH

aiohttp server

```
@asyncio.coroutine
def hello(request):
    return web.Response(body=b"Hello, world")

app = web.Application()
app.router.add_route('GET', '/', hello)
```

aiohttp server

```
@asyncio.coroutine
def stats(request):
    body = yield from get_stats()
    return web.Response(body=body)

app.router.add_route('GET', '/stats', stats)
```

Unit tests

- asynctest: for unittest
- pytest-asyncio: for pytest

- aiotest (test asyncio implementation)

Trollius

- Trollius is the Python 2 port of asyncio
- Work on Python 2.6 – 3.5
- Use "`yield From(...)`" instead of "`yield from ...`"
- Only a few asyncio libraries are compatible with trollius
- Only use it if you cannot port your application to Python 3

redhat.

python™

Benchmarks

- Ludovic Gasc ran benchmark on Flask, Django and API Hour (asyncio)
- API Hour is as fast or much faster
- Best case: API-Hour handles 5x more requests per second
- JSON serialization: 400k req/s API-Hour vs 70-79k for Django-Flask
- Details: http://blog.gmludo.eu/

redhat

python™

How can you help?

- Need tutorials and more documentation
- Port more stdlib modules to Python 3: ftplib, poplib, imaplib, nntplib, smtplib, telnetlib, xmlrpclib, etc.
- Interoperability with Twisted

Questions ?

http://www.asyncio.org/
github.com/python/asyncio/wiki/ThirdParty

Contact :
vstinner@gmail.com