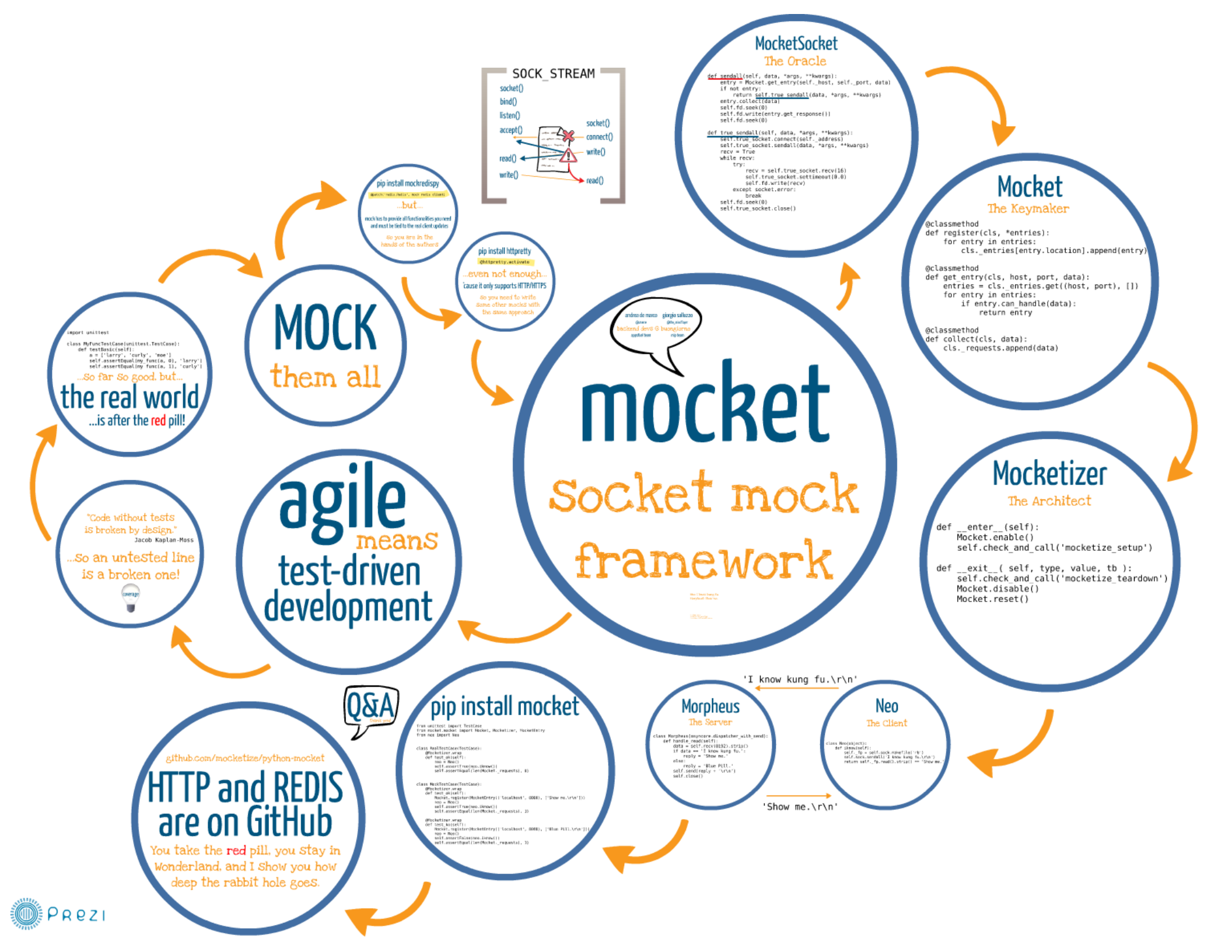

MocketSocket
The Oracle
Mocket
The Keymaker
Mocketizer
The Architect
Morpheus
The Server
Neo
The Client
'I know kung fu.\r\n'
'Show me.\r\n'
SOCK_STREAM
socket()
bind()
listen()
accept()
read()
write()
socket()
connect()
write()
read()
pip install mockredispy
...but...
mock has to provide all functionalities you need and must be tied to the real client updates
So you are in the hands of the authors
pip install httpretty
...even not enough...
'cause it only supports HTTP/HTTPS
So you need to write same other mocks with the same approach
MOCK
them all
mocket
Socket mock framework
agile
means
test-driven
development
"Code without tests is broken by design."
Jacob Kaplan-Moss
...So an untested line is a broken one!
...So far So good. but...
the real world
...is after the red pill!
pip install mocket
Q&A
github.com/mocketize/python-mocket
HTTP and REDIS are on GitHub
You take the red pill, you stay in Wonderland, and I show you how deep the rabbit hole goes.
Prezi

...even not enough...

'cause it only supports HTTP/HTTPS

So you need to write some other mocks with the same approach

andrea de marco @z4erre appsfuel team

giorgio salluzzo @the_mindflayer mip team

backend devs @ buongiorno

mocket

socket mock framework

Neo: I know kung fu.
Morpheus: Show me.

```
def get_
    entr
    for


@classme
def coll
    cls.
```

```
def _
    M
    s

def _
    s
    M
    M
```

Neo: I know kung fu.

Morpheus: Show me.


andrea de marco
@z4erre
giorgio salluzzo
@the_mindflayer
backend devs @ buongiorno
appsfuel team
mip team


agile
means
test-driven
development

"Code without tests
is broken by design."

Jacob Kaplan-Moss

...So an untested line
is a broken one!

```
import unittest

class MyFuncTestCase(unittest.TestCase):
    def testBasic(self):
        a = ['larry', 'curly', 'moe']
        self.assertEqual(my_func(a, 0), 'larry')
        self.assertEqual(my_func(a, 1), 'curly')
```

...So far So good, but...

the real world

...is after the red pill!

pip install mockredispy

```
@patch('redis.Redis', mock_redis_client)
```

...but...

mock has to provide all functionalities you need
and must be tied to the real client updates

So you are in the
hands of the authors

pip install httpretty

```
@httpretty.activate
```

...even not enough...

'cause it only supports HTTP/HTTPS

So you need to write
Some other mocks with
the same approach

...even not enough...

'cause it only supports HTTP/HTTPS

So you need to write some other mocks with the same approach

andrea de marco @z4erre appsfuel team

giorgio salluzzo @the_mindflayer mip team

backend devs @ buongiorno

mocket

socket mock framework

Neo: I know kung fu.
Morpheus: Show me.

```
def get_
    entr
    for


@classme
def coll
    cls.
```

```
def _
    M
    s

def _
    s
    M
    M
```

Neo: Whoa. Déjà vu.
Trinity: What did you just say?
Neo: Nothing. Just had a little déjà vu.

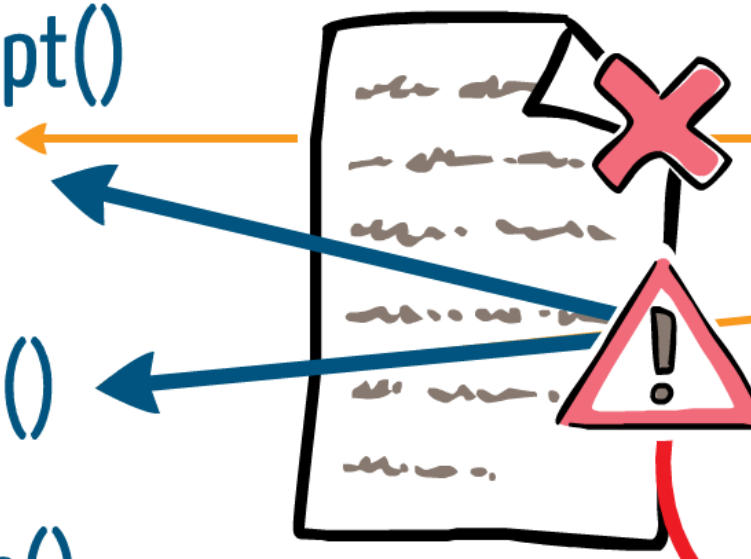
SOCK_STREAM

socket()

bind()

listen()

accept()



read()

write()

socket()

connect()

write()

read()

MocketSocket

The Oracle

```
def sendall(self, data, *args, **kwargs):
    entry = Mocket.get_entry(self._host, self._port, data)
    if not entry:
        return self.true_sendall(data, *args, **kwargs)
    entry.collect(data)
    self.fd.seek(0)
    self.fd.write(entry.get_response())
    self.fd.seek(0)

def true_sendall(self, data, *args, **kwargs):
    self.true_socket.connect(self._address)
    self.true_socket.sendall(data, *args, **kwargs)
    recv = True
    while recv:
        try:
            recv = self.true_socket.recv(16)
            self.true_socket.settimeout(0.0)
            self.fd.write(recv)
        except socket.error:
            break
    self.fd.seek(0)
    self.true_socket.close()
```

```
@classme
def regi
    for
```

Mocket

The Keymaker

```
@classmethod
def register(cls, *entries):
    for entry in entries:
        cls._entries[entry.location].append(entry)

@classmethod
def get_entry(cls, host, port, data):
    entries = cls._entries.get((host, port), [])
    for entry in entries:
        if entry.can_handle(data):
            return entry

@classmethod
def collect(cls, data):
    cls._requests.append(data)
```

Mocketizer

The Architect

```
def __enter__(self):
    Mocket.enable()
    self.check_and_call('mocketize_setup')

def __exit__( self, type, value, tb ):
    self.check_and_call('mocketize_teardown')
    Mocket.disable()
    Mocket.reset()
```

'I know kung fu.\r\n'

Morpheus

The Server

```
class Morpheus(asyncore.dispatcher_with_send):
    def handle_read(self):
        data = self.recv(8192).strip()
        if data == 'I know kung fu.':
            reply = 'Show me.'
        else:
            reply = 'Blue Pill.'
        self.send(reply + '\r\n')
        self.close()
```

Neo

The Client

```
class Neo(object):
    def iknow(self):
        self._fp = self.sock.makefile('rb')
        self.sock.sendall('I know kung fu.\r\n')
        return self._fp.read().strip() == 'Show me.'
```

'Show me.\r\n'

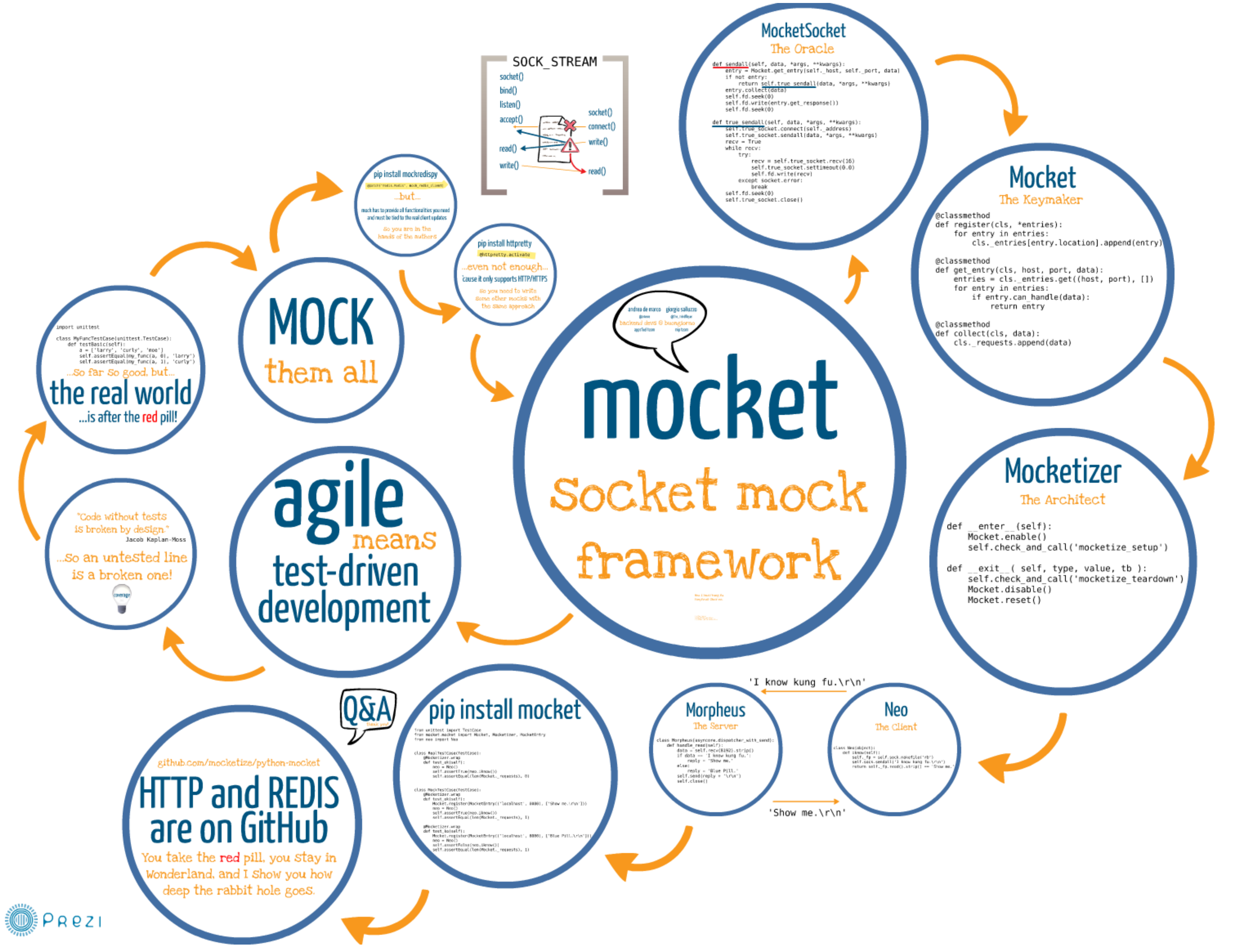
pip install mocket

```
from unittest import TestCase
from mocket.mocket import Mocket, Mocketizer, MocketEntry
from neo import Neo


class RealTestCase(TestCase):
    @Mocketizer.wrap
    def test_ok(self):
        neo = Neo()
        self.assertTrue(neo.iknow())
        self.assertEqual(len(Mocket._requests), 0)


class MockTestCase(TestCase):
    @Mocketizer.wrap
    def test_ok(self):
        Mocket.register(MocketEntry(('localhost', 8080), ['Show me.\r\n']))
        neo = Neo()
        self.assertTrue(neo.iknow())
        self.assertEqual(len(Mocket._requests), 1)

    @Mocketizer.wrap
    def test_ko(self):
        Mocket.register(MocketEntry(('localhost', 8080), ['Blue Pill.\r\n']))
        neo = Neo()
        self.assertFalse(neo.iknow())
        self.assertEqual(len(Mocket._requests), 1)
```

github.com/mocketize/python-mocket

HTTP and REDIS are on GitHub

You take the red pill, you stay in Wonderland, and I show you how deep the rabbit hole goes.



mocket
Socket mock framework
andrea de marco
giorgio salluzzo
backend devs @ buongiorno
SOCK_STREAM
socket()
bind()
listen()
accept()
read()
write()
socket()
connect()
write()
read()
MocketSocket
The Oracle
def sendall(self, data, *args, **kwargs):
entry = Mocket.get_entry(self._host, self._port, data)
if not entry:
return self.true_sendall(data, *args, **kwargs)
entry.collect(data)
self.fd.seek(0)
self.fd.write(entry.get_response())
self.fd.seek(0)
def true_sendall(self, data, *args, **kwargs):
self.true_socket.connect(self._address)
self.true_socket.sendall(data, *args, **kwargs)
recv = True
while recv:
try:
recv = self.true_socket.recv(16)
self.true_socket.settimeout(0.0)
self.fd.write(recv)
except socket.error:
break
self.fd.seek(0)
self.true_socket.close()
Mocket
The Keymaker
@classmethod
def register(cls, *entries):
for entry in entries:
cls._entries[entry.location].append(entry)
@classmethod
def get_entry(cls, host, port, data):
entries = cls._entries.get((host, port), [])
for entry in entries:
if entry.can_handle(data):
return entry
@classmethod
def collect(cls, data):
cls._requests.append(data)
Mocketizer
The Architect
def __enter__(self):
Mocket.enable()
self.check_and_call('mocketize_setup')
def __exit__(self, type, value, tb):
self.check_and_call('mocketize_teardown')
Mocket.disable()
Mocket.reset()
'I know kung fu.\r\n'
Morpheus
The Server
Neo
The Client
'Show me.\r\n'
pip install mocket
Q&A
github.com/mocketize/python-mocket
HTTP and REDIS are on GitHub
You take the red pill, you stay in Wonderland, and I show you how deep the rabbit hole goes.
agile means test-driven development
"Code without tests is broken by design."
Jacob Kaplan-Moss
...so an untested line is a broken one!
coverage
import unittest
...so far so good, but...
the real world
...is after the red pill!
MOCK them all
pip install mockredispy
...but...
So you are in the hands of the authors
pip install httpretty
httpretty.activate
...even not enough...
'cause it only supports HTTP/HTTPS
So you need to write some other mocks with the same approach
Prezi