TDD de 0 a 100 (o casi)

Raúl Villares (@RaulVillaresBg)
Alberto de la Cruz (@AlbertodlCG)

¿ Quienes somos ?

aleasoluciones

¿ Por qué Python ?

- Curva de aprendizaje
- Módulos para “hablar” con electrónica de red (pexpect, pysnmp)
- Opensource friendly
- Disponible “Out of the Box” en todos los sistemas operativos que usamos
- La conjunción de todas las anteriores

Resumen del taller

- Objetivos:
  - Ir un paso más allá de los test unitarios y hablar de dobles y tests de integración
  - Compartir las herramientas/técnicas que utilizamos en Alea
- Estructura:
  - Un poco de teoría de TDD
  - Ejercicio 1
  - Otro poco de teoría de dobles
  - Ejercicio 2 y 3
  - Conclusión y cierre

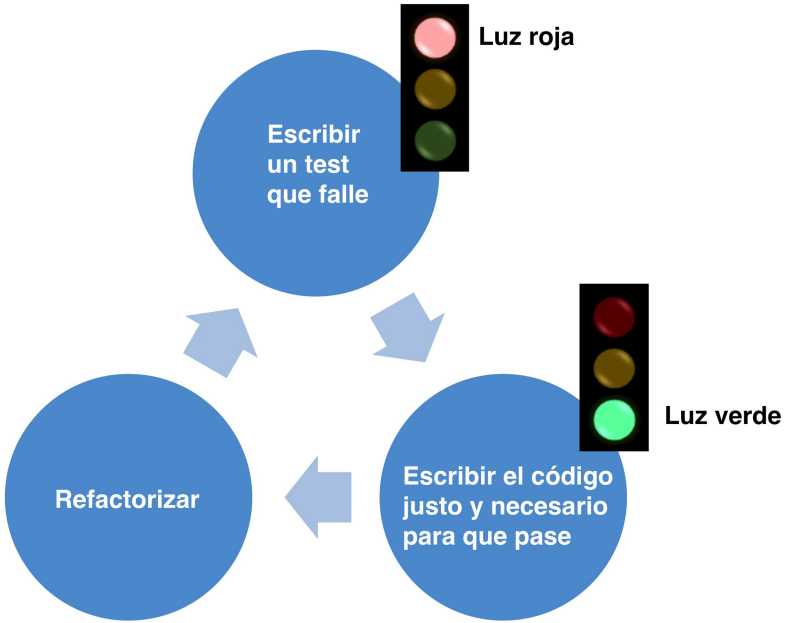
Temperatura TDD

- ¿ Quién es la primera vez que lo escucha ?
- ¿ A quién le suena ?
- ¿ Quién practica ?

- Antes del taller ➜ un poco teoría

Un poco de teoría: TDD

- TDD ➔ Test Driven Development



Buenas prácticas

- AAA / Given-When-Then ➜ Arrange (Preparar), Act (Actuar), Assert (Afirmar)
- Mantén cada test enfocado sólo en el resultado necesario para validar ese test.
- Trata tus tests con el mismo respeto que a tu código de producción.
- Comparte las prácticas que usas en los tests con el resto de tu equipo de manera que todos puedan evolucionar y mantener los tests de similar manera.

Antipatrones

- Tests que dependan del estado de tu sistema o de tests previamente ejecutados.
- Dependencia entre tests.
- Orden de ejecución.
- Tests frágiles.
- Testear detalles de implementación. Se testea el QUÉ, no el CÓMO.
- Tests lentos de ejecutar.

¿ Por qué lo usamos en Alea ?

- Sólo el código necesario (YAGNI), sin complejidades innecesarias.
- Foco en la funcionalidad
- Ayuda a tener código desacoplado
- Favorece que puedas aplicar ciertos patrones de diseño
- Software funcionando (cumple lo que dicen los tests)
- Tranquilidad ante los cambios (malla de seguridad)
- Documentación ejecutable
- Si no haces los tests al principio, probablemente nunca los hagas :)

Herramientas que usamos (1ª parte)

- Mamba
- https://github.com/nestorsalceda/mamba

```python
from mamba import description, context, it

with description('checking if string contains a character'):
    with context('given a character and a string'):
        with context('when the string contains the character'):
            with it('returns true'):
                character = 'C'
                string = 'PyConES2018'

                assert(character in string)

        with context('when the string does NOT contain the character'):
            with it('return false'):
                character = 'x'
                string = 'PyConES2018'

                assert(character not in string)
```

Herramientas que usamos (1ª parte)

- Expects
- https://github.com/jaimegildesagredo/expects

```
from mamba import description, context, it
from expects import expect, contain

with description('checking if string contains a character'):
    with context('given a character and a string'):
        with context('when the string contains the character'):
            with it('returns true'):
                character = 'C'
                string = 'PyConES2018'

                expect(string).to(contain(character))

        with context('when the string does NOT contain the character'):
            with it('return false'):
                character = 'x'
                string = 'PyConES2018'

                expect(string).not_to(contain(character))
```

Taller - Presentación

- BezOS: el sistema operativo con el que Jeff pretende aniquilar a los Mac.
- ¿ Cómo se divide el taller ?
  - La gran prueba de entrada (1 iteración)
  - El encargo de Jeff (2 iteraciones)

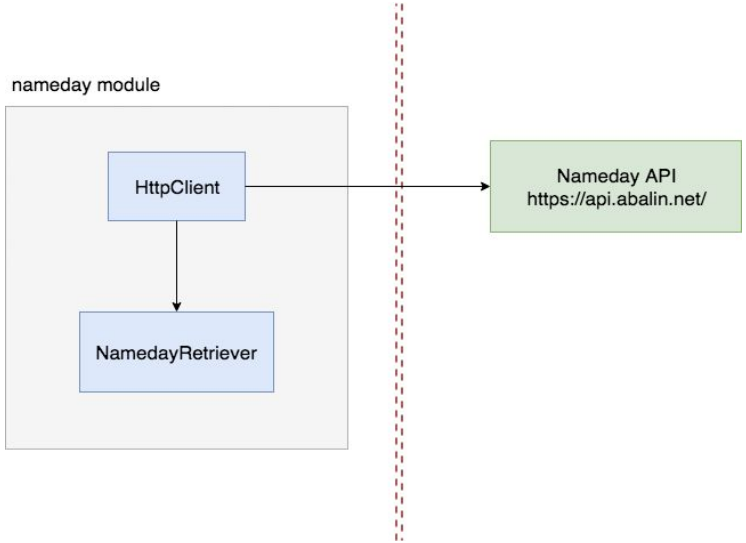
La gran prueba de entrada

- Tests unitarios: se encargan de testear un módulo por sí mismo, sin interacción con el resto de módulos. FIRST.
- Serie de Fibonacci
  - Enunciado: queremos una función que dado un índice, devuelva el valor asociado a la serie de fibonacci (para el 0 devuelve 0, para el 1 devuelve 1 y para el resto la suma de los 2 anteriores).

0 1 1 2 3 5 8 13 21
34 55 89 144 233 377
610 987 1597 2584....

El encargo de Jeff

- Enunciado: se pretende desarrollar un componente del BezOS que devuelva los santos del día para un país en concreto. Para ello, se utiliza el API proporcionado por https://api.abalin.net/



Dobles

- El nombre viene de los dobles de las películas de acción
- Nos permiten no tener que usar código de producción
- Parte fundamental de los tests de integración
- Buscamos testear la interacción entre módulos

Tipos de dobles

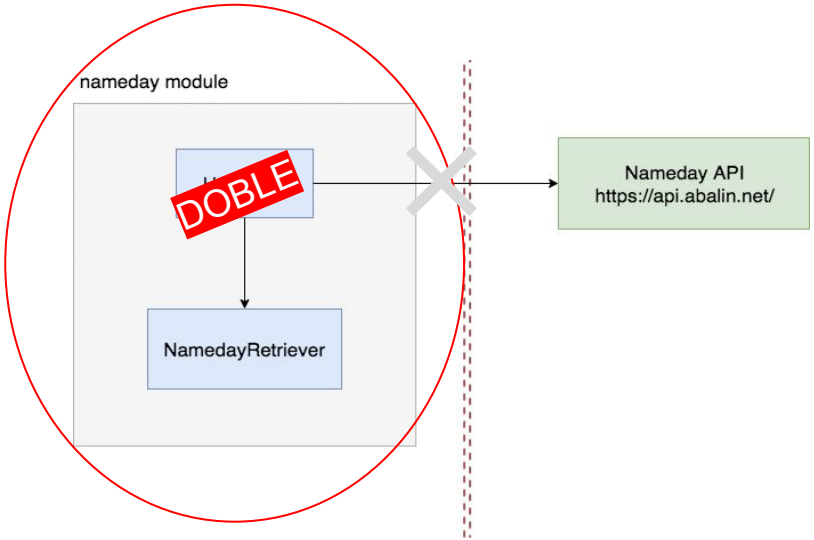
- Dummy: un doble que no hace nada pero es necesario que exista.
- Fake: es un doble totalmente funcional pero con “atajos”. Por ejemplo, una base de datos en memoria.
- **Stub**: te dice lo que quieres oír. Por ejemplo: cuando llames al método add con los valores 2 y 4, devuelve 6.
  - Nosotros solemos usarlo con “when”
- **Spy**: recuerda cualquier cosa que le pasa. Por ejemplo: es muy útil para testear la colaboración entre módulos.
  - Nosotros solemos usarlo con “have_been_called(_with)”
- Mock: simula exactamente el orden de las llamadas.

Herramientas

- Doublex:
  - https://github.com/davidvilla/python-doublex
  - https://python-doublex.readthedocs.io/en/latest/
- Doublex-expects:
  - Matcher para doublex
  - https://github.com/jaimegildesagredo/doublex-expects

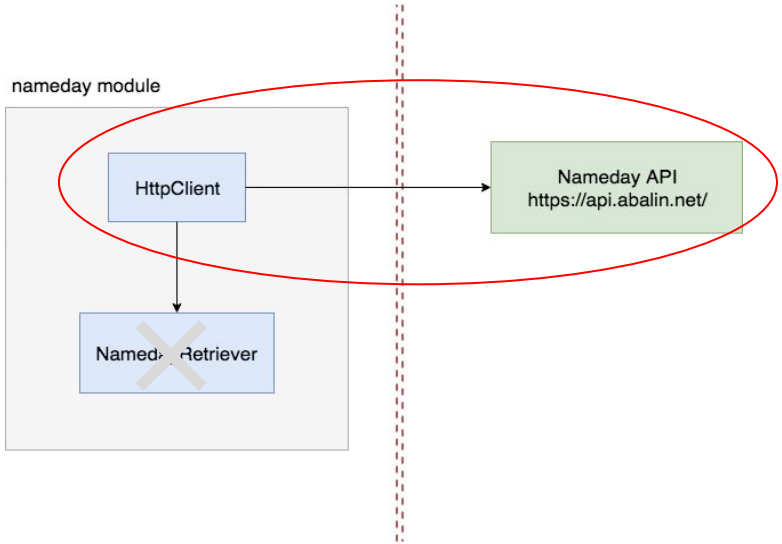
El encargo de Jeff: 1ª iteración

- Enunciado: se pretende desarrollar un sistema que devuelva los santos del día para un país en concreto. Para ello, se utiliza el API proporcionado por https://api.abalin.net/
- En esta iteración, el objetivo es centrarse en el módulo NamedayRetriever, doblando el httpclient para simular la conexión al API. Dada la salida que te ofrece el API al consultar por España (https://api.abalin.net/get/today?country=es) , obtener un string con los nombres separados por comas.



El encargo de Jeff: 2ª iteración

- Enunciado: se pretende desarrollar un sistema que devuelva los santos del día para un país en concreto. Para ello, se utiliza el API proporcionado por https://api.abalin.net/
- En esta iteración, el objetivo testear el contrato con el API, realizando llamadas reales desde el httpclient.



Enlaces

- http://www.codemanship.co.uk/tdd_jasongorman_codemanship.pdf
- https://librosweb.es/libro/tdd/
- https://docs.google.com/document/d/1tJSfw5nvkSB_cpAqBleHeoStcs0gPVg7uGrWxYmingI/edit#heading=h.498fp6zdnxpr
- https://martinfowler.com/tags/test%20categories.html

Domingo 7 a las 9:30, Charla **Mocks, fakes, dummies, stubs and spies: Successfully isolating the snake** por **Mario Corchero**
https://2018.es.pycon.org/talk/mocks-fakes-dummies-stubs-and-spies-successfully-isolating-the-snake