Unit Testing with Python

Module 5: Test Doubles

Emily Bache
http://coding-is-like-cooking.info
emily@bacheconsulting.com

pluralsight
hardcore developer training

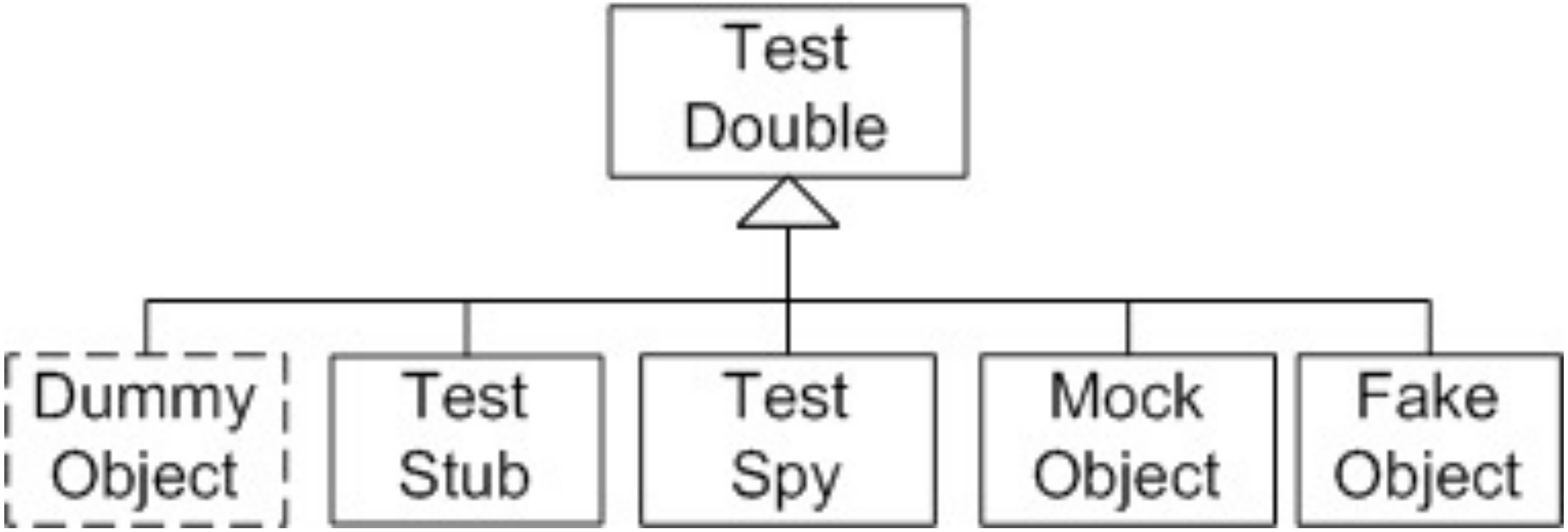
Module Overview

- What is a Test Double?
- Different kinds of Test Double
- Why use Test Doubles?
- using Monkeypatching to insert Test Doubles

Test Double

- like “Stunt Doubles” who stand in for actors in films
- class under test doesn’t know it isn’t talking to the real object
- Allow you to control what happens to your class under test

Different kinds of test double



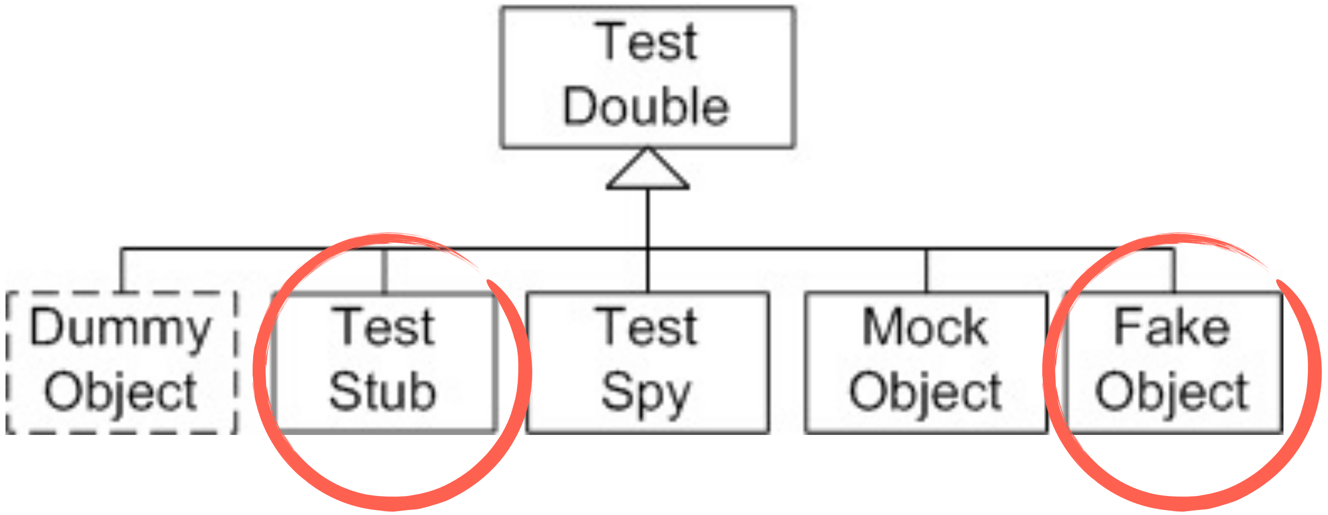
from http://xunitpatterns.com

see also: https://en.wikipedia.org/wiki/Test_Double

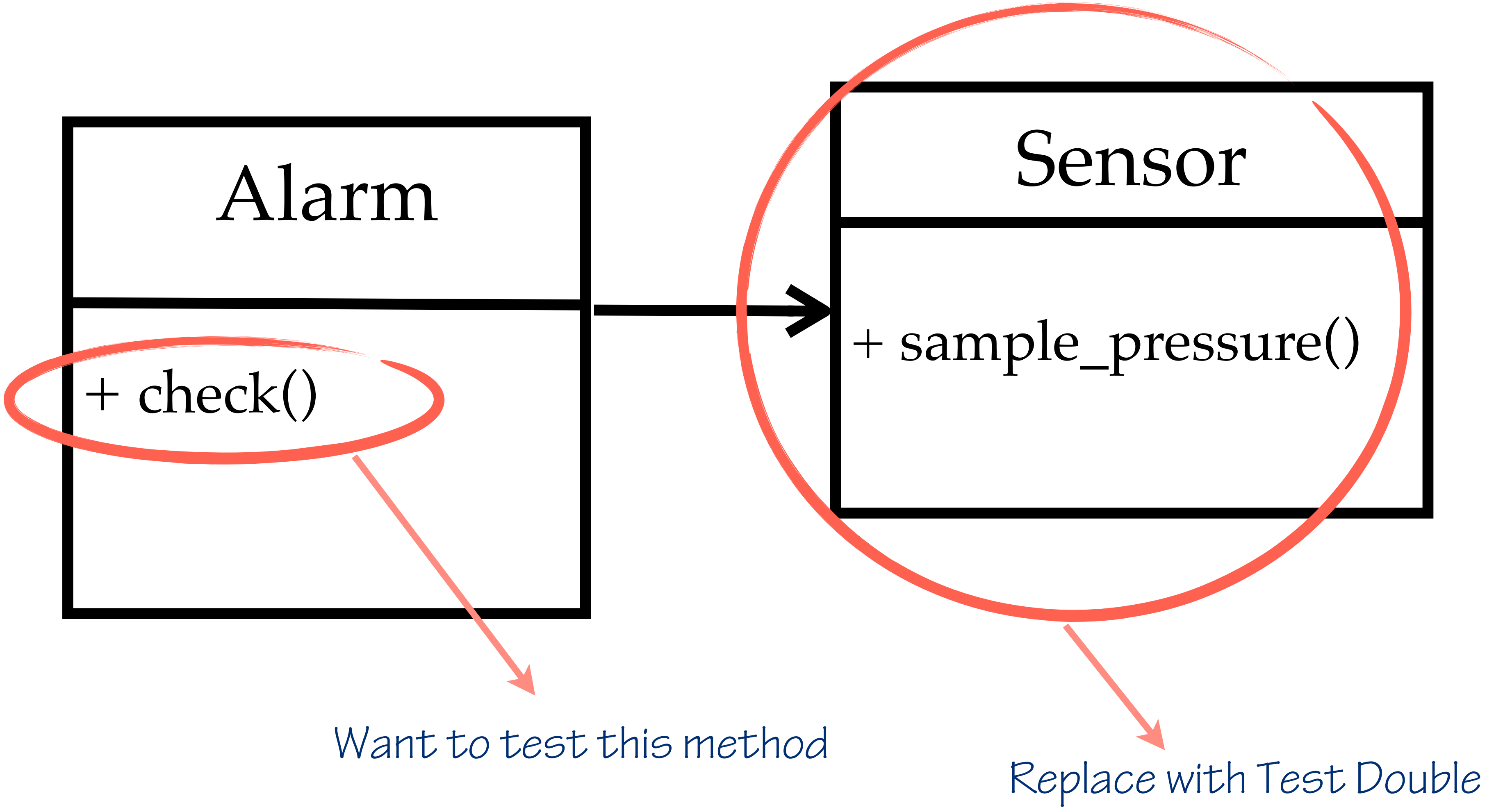
Section outline

- Stub
- Fake
- Mock
- Test Spy
- Dummy Object

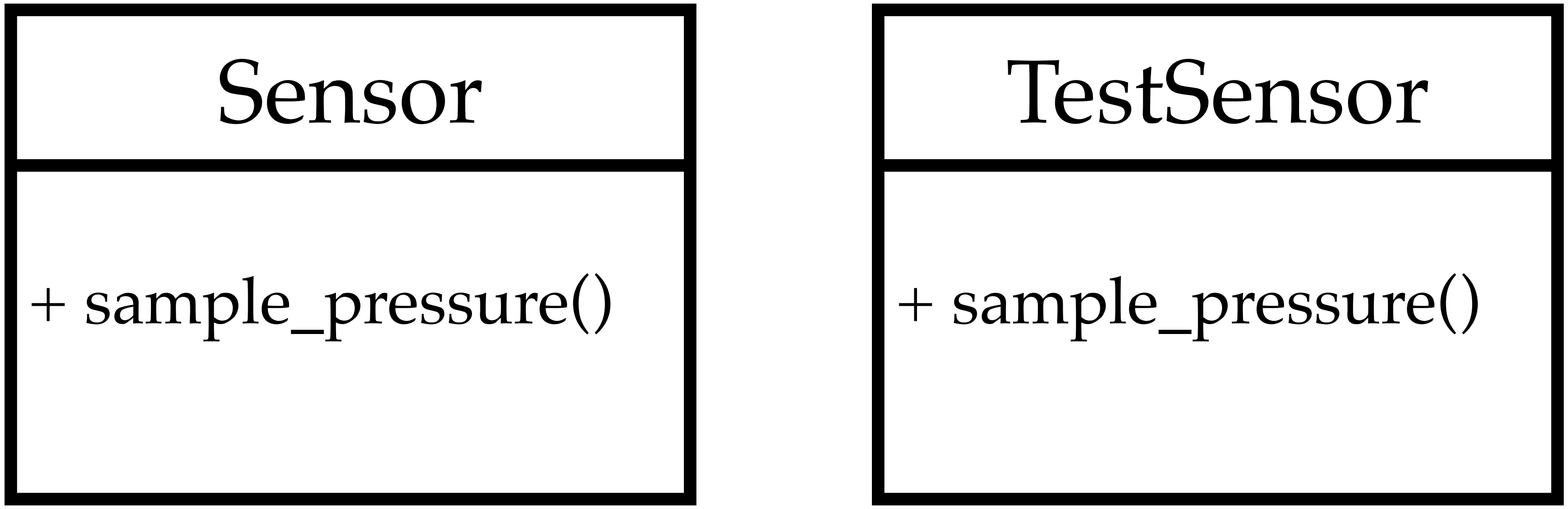
Test Doubles



Racing Car Example



Stub

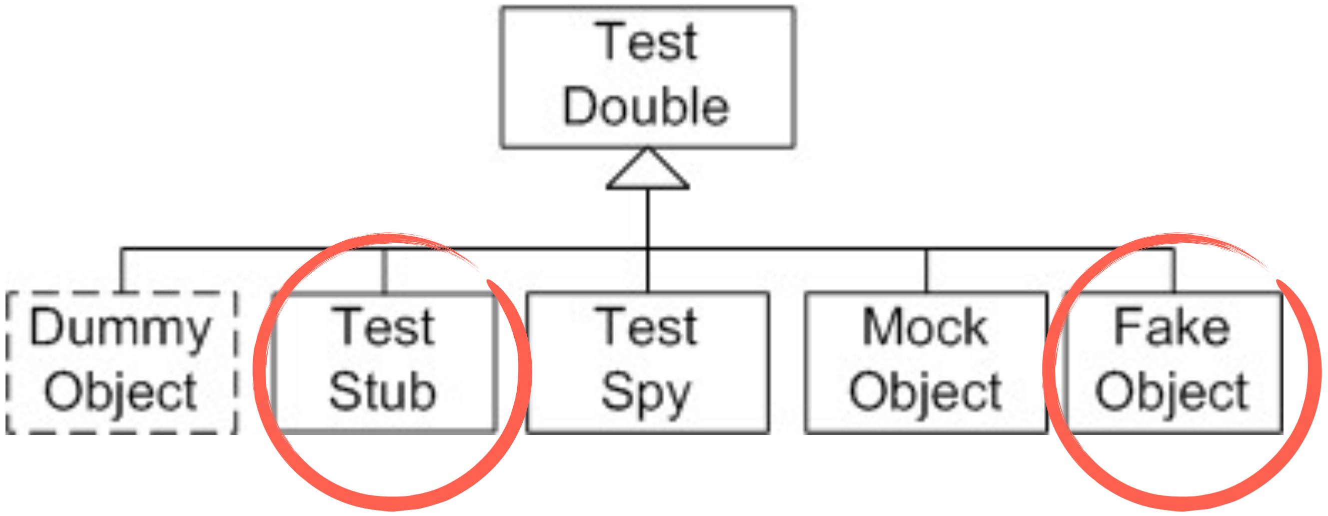


Same interface

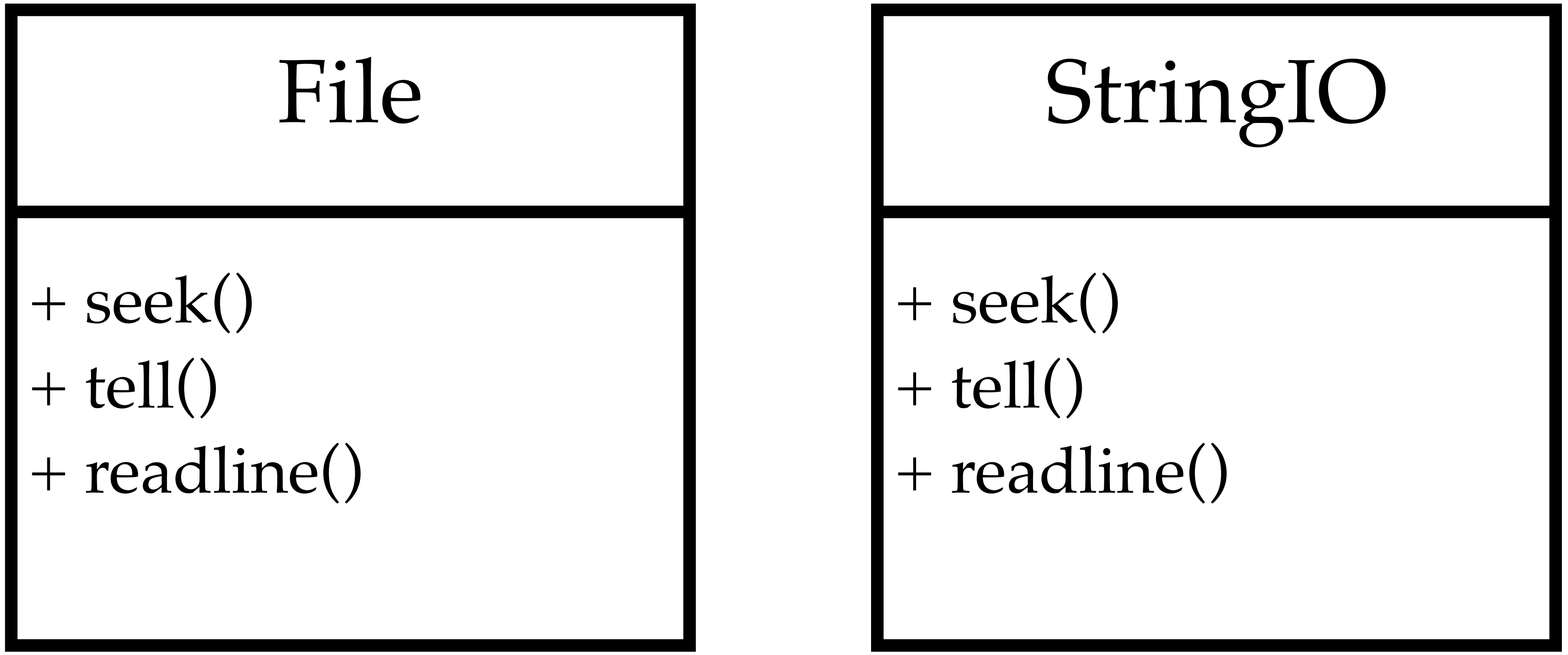
Stub has no logic or advanced behaviour

a Stub is not the same as a Mock!

Test Doubles

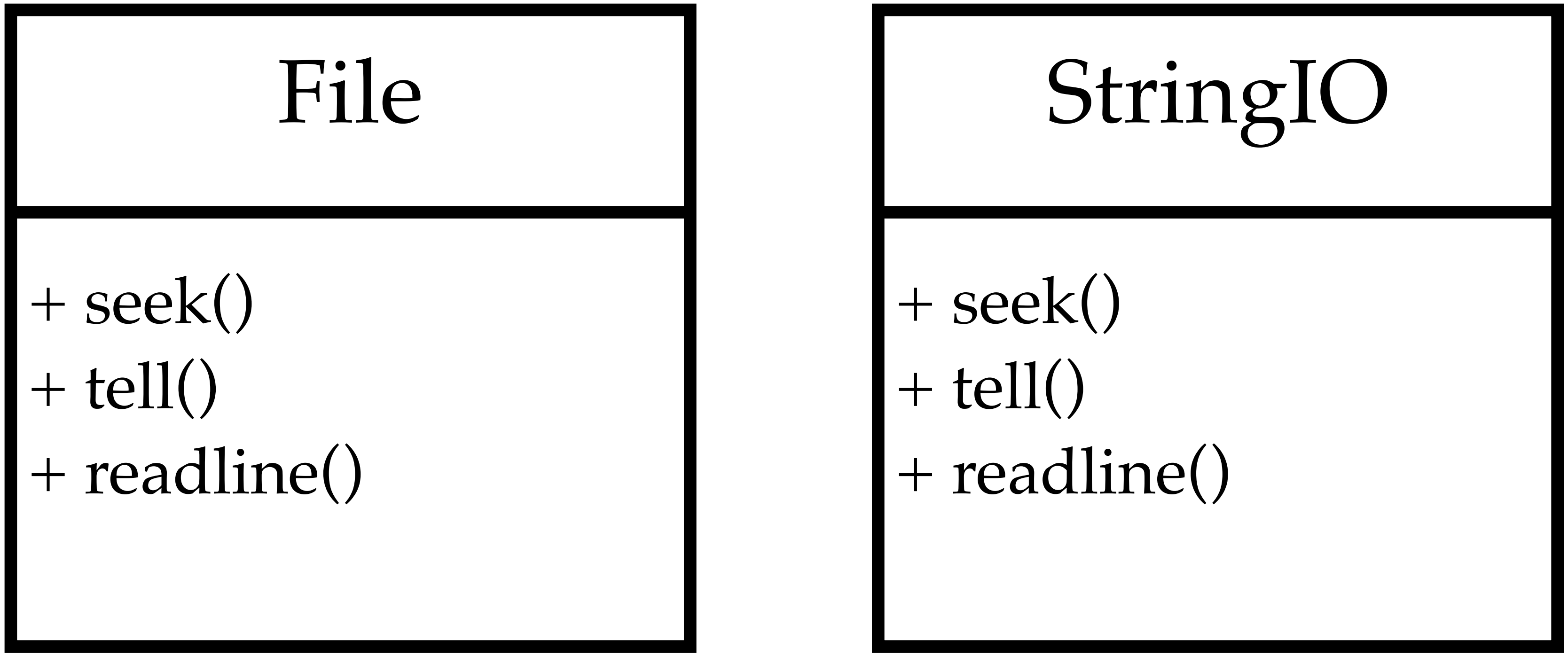


Fake



Same interface

Fake has logic and behaviour but is unsuitable for production


File
+ seek()
+ tell()
+ readline()
StringIO
+ seek()
+ tell()
+ readline()

Common things to replace with Fakes

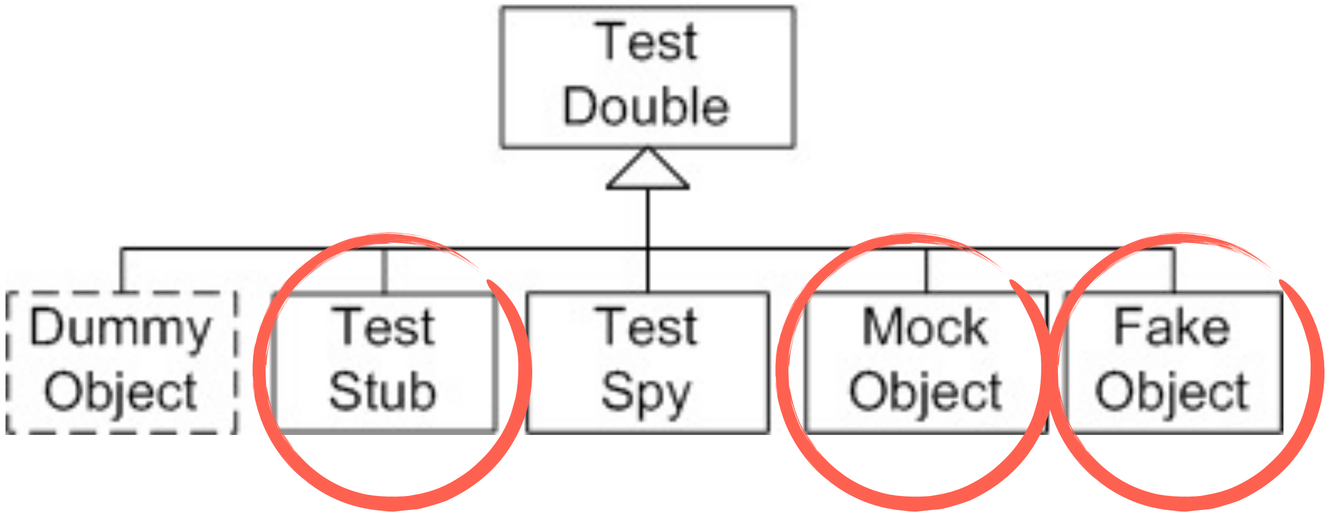
File

Database

WebServer

icons: Double-J Design, FatCow

Test Doubles

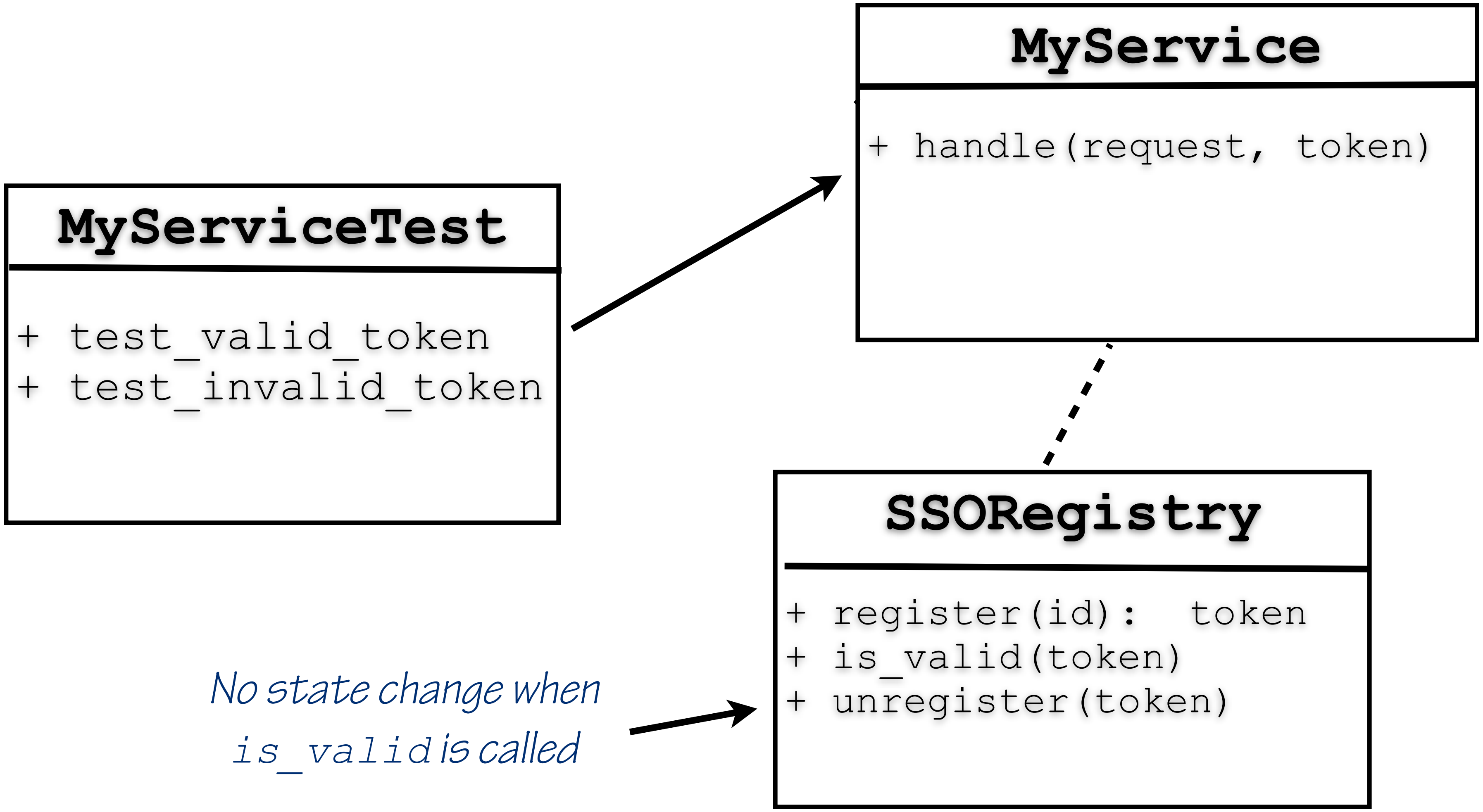


Three kinds of Assert

- Check the return value or an exception
- Check a state change (use a public API)
- Check a method call (use a mock or spy)

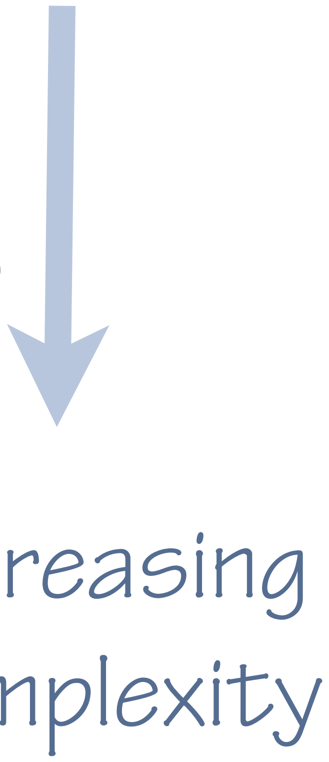
Increasing complexity

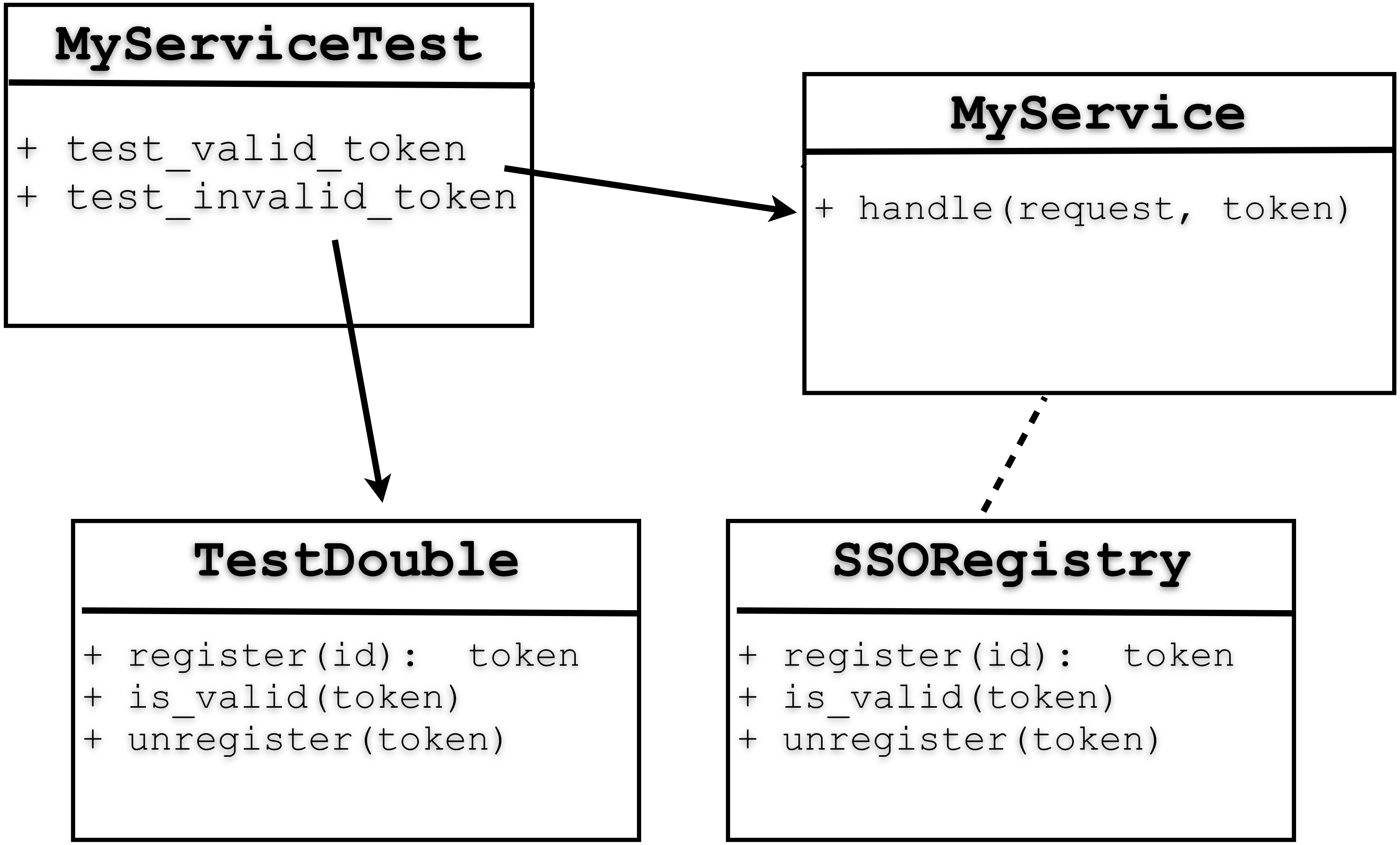
Interaction Testing



Three kinds of Assert

- Check the return value or an exception
- Check a state change (use a public API)
- Check a method call (use a mock or spy)



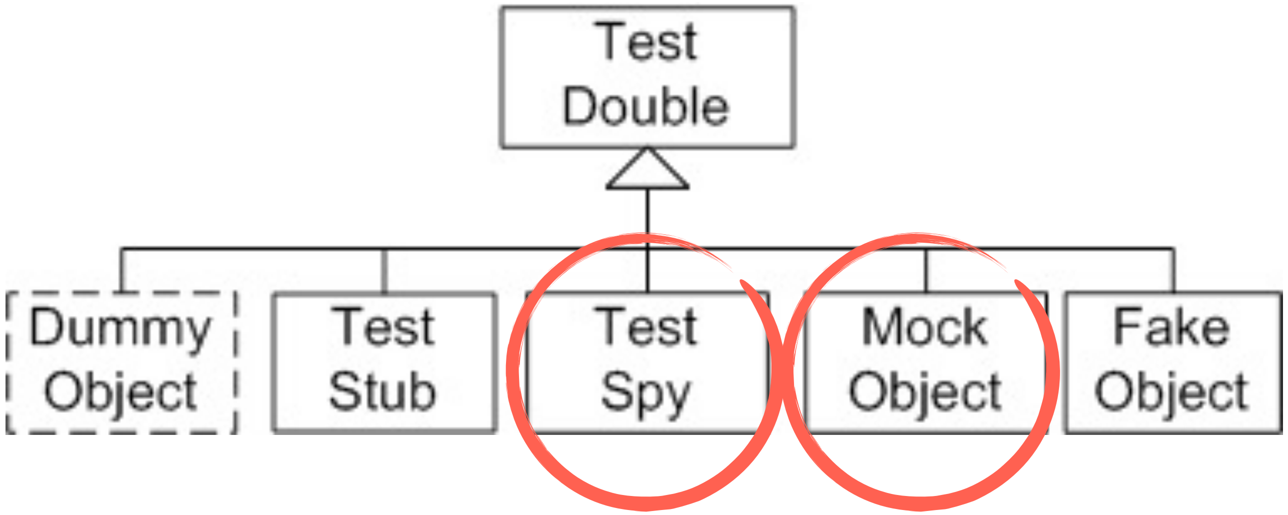

MyServiceTest
+ test_valid_token
+ test_invalid_token
MyService
+ handle(request, token)
TestDouble
+ register(id): token
+ is_valid(token)
+ unregister(token)
SSORegistry
+ register(id): token
+ is_valid(token)
+ unregister(token)

Three kinds of Assert

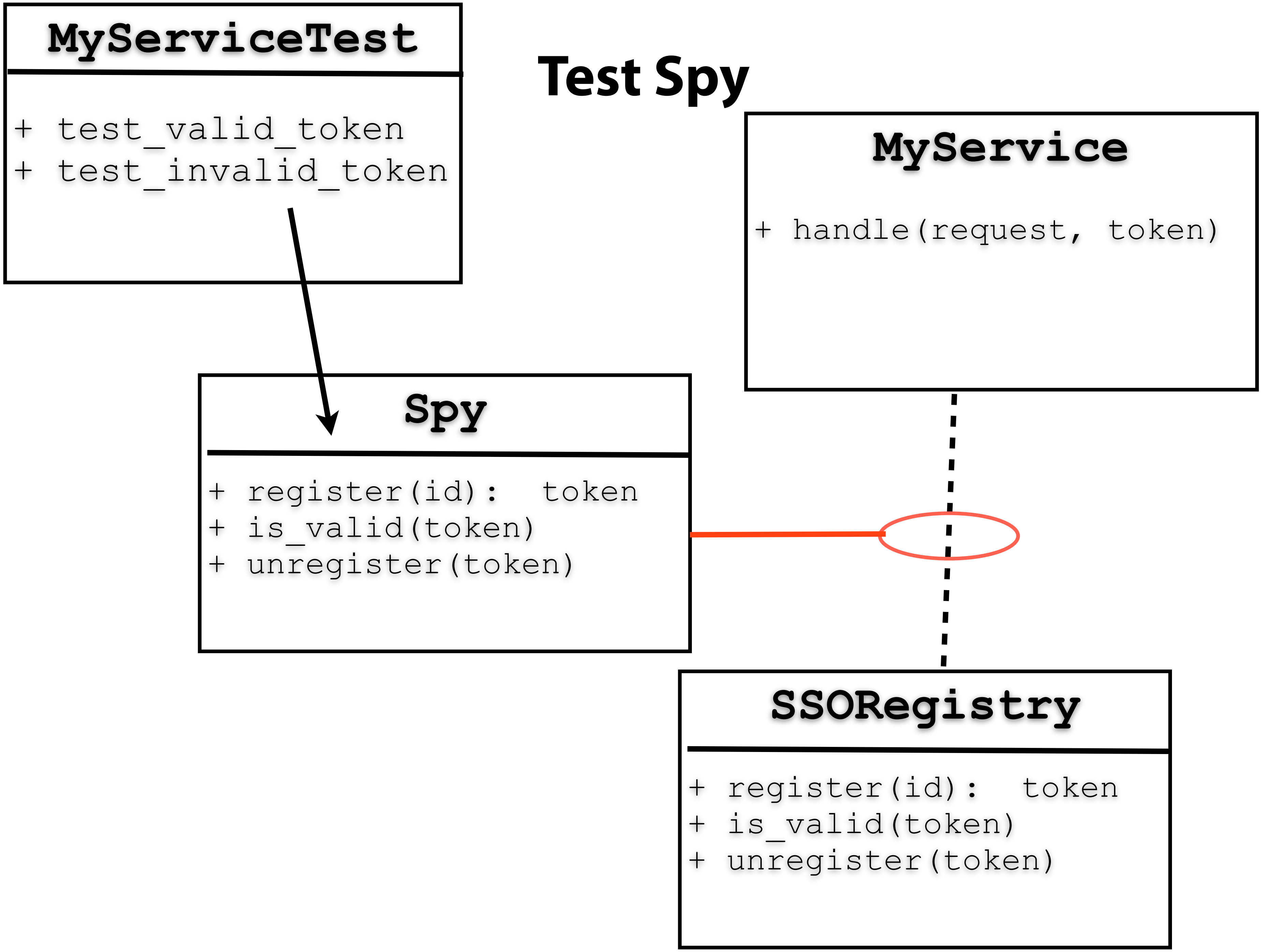
- Check the return value or an exception
- Check a state change (use a public API)
- Check a method call (use a mock or spy)

Increasing complexity

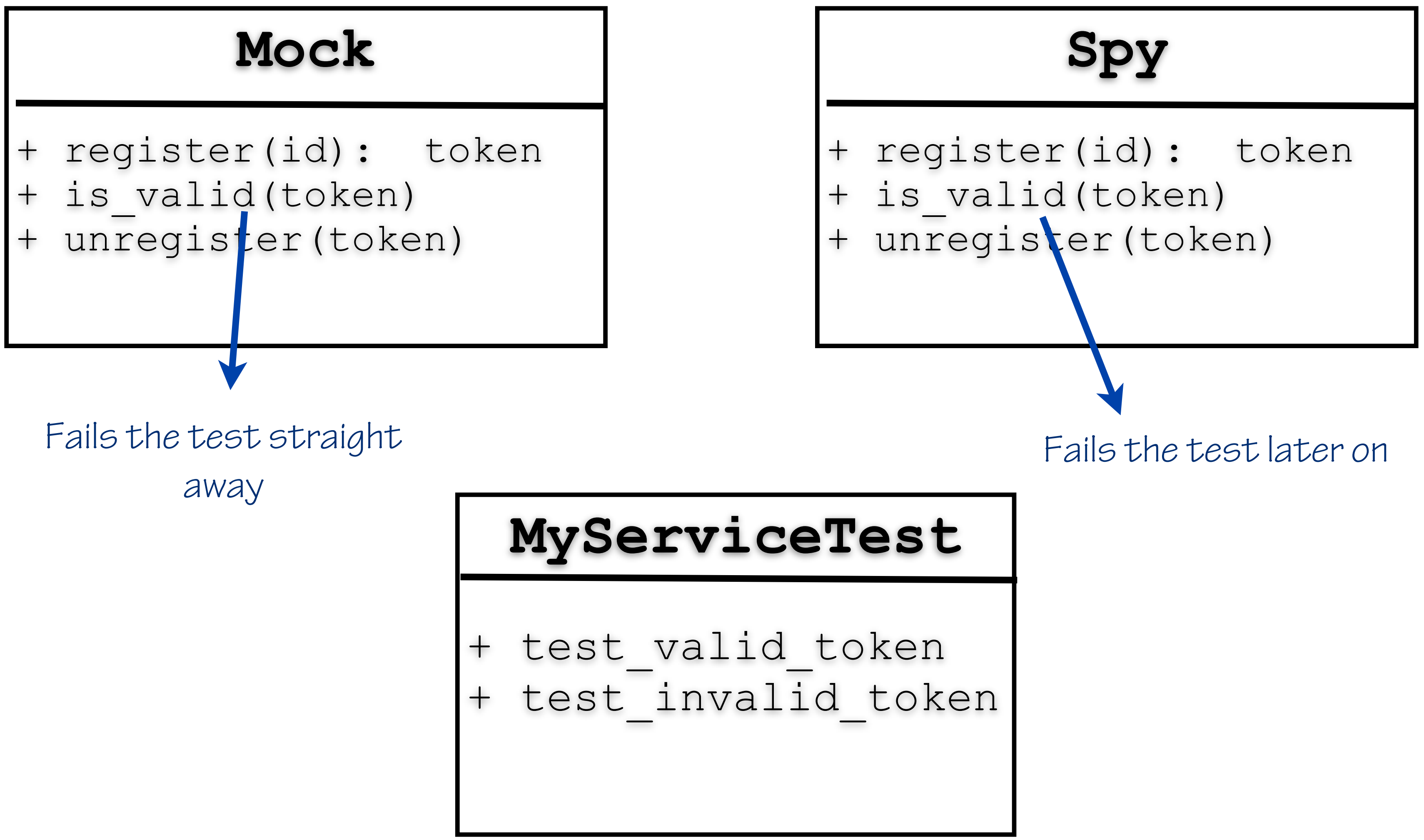
Test Doubles



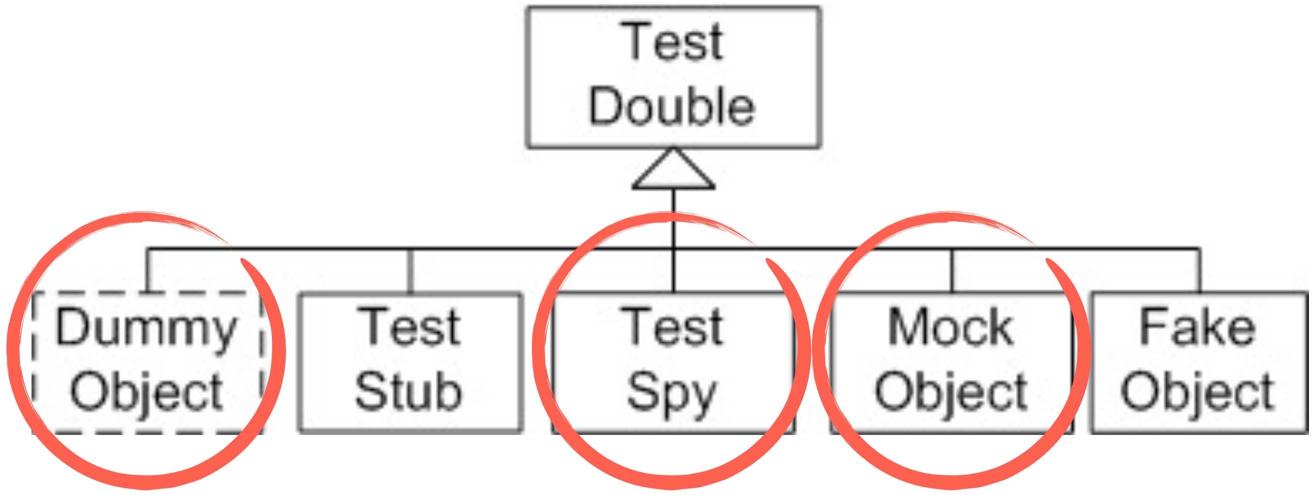
Test Spy


MyServiceTest
+ test_valid_token
+ test_invalid_token
MyService
+ handle(request, token)
Spy
+ register(id): token
+ is_valid(token)
+ unregister(token)
SSORegistry
+ register(id): token
+ is_valid(token)
+ unregister(token)

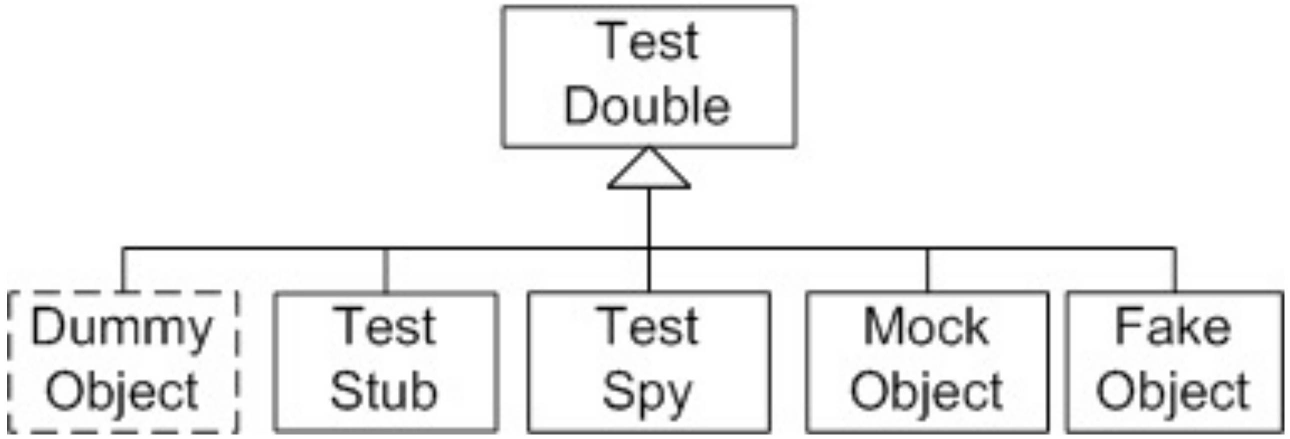
Test Spy


Mock
+ register(id): token
+ is_valid(token)
+ unregister(token)
Fails the test straight away
Spy
+ register(id): token
+ is_valid(token)
+ unregister(token)
Fails the test later on
MyServiceTest
+ test_valid_token
+ test_invalid_token

Test Doubles



What’s the difference?



- A **Stub** returns a hard coded answer to any query. It contains no logic.
- A **Fake** is a real implementation, but simpler - like a big complicated Stub.
- A **Mock** is as a Stub, and additionally verifies interactions.
- A **Test Spy** lets you query afterwards to find out what happened.
- A **Dummy** is something you use when the interface requires an argument which isn't needed for the test.

What is Unit Testing For?

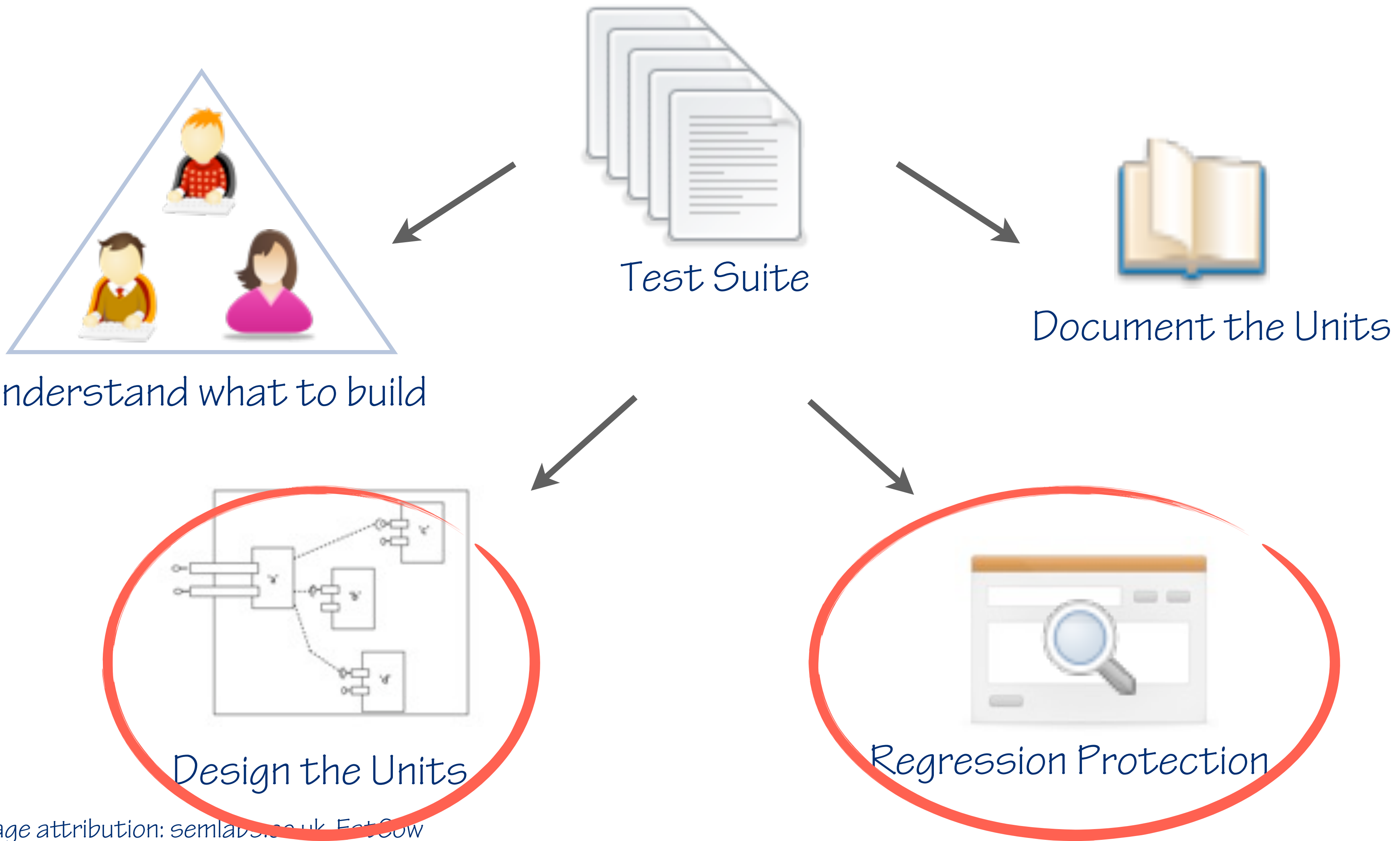
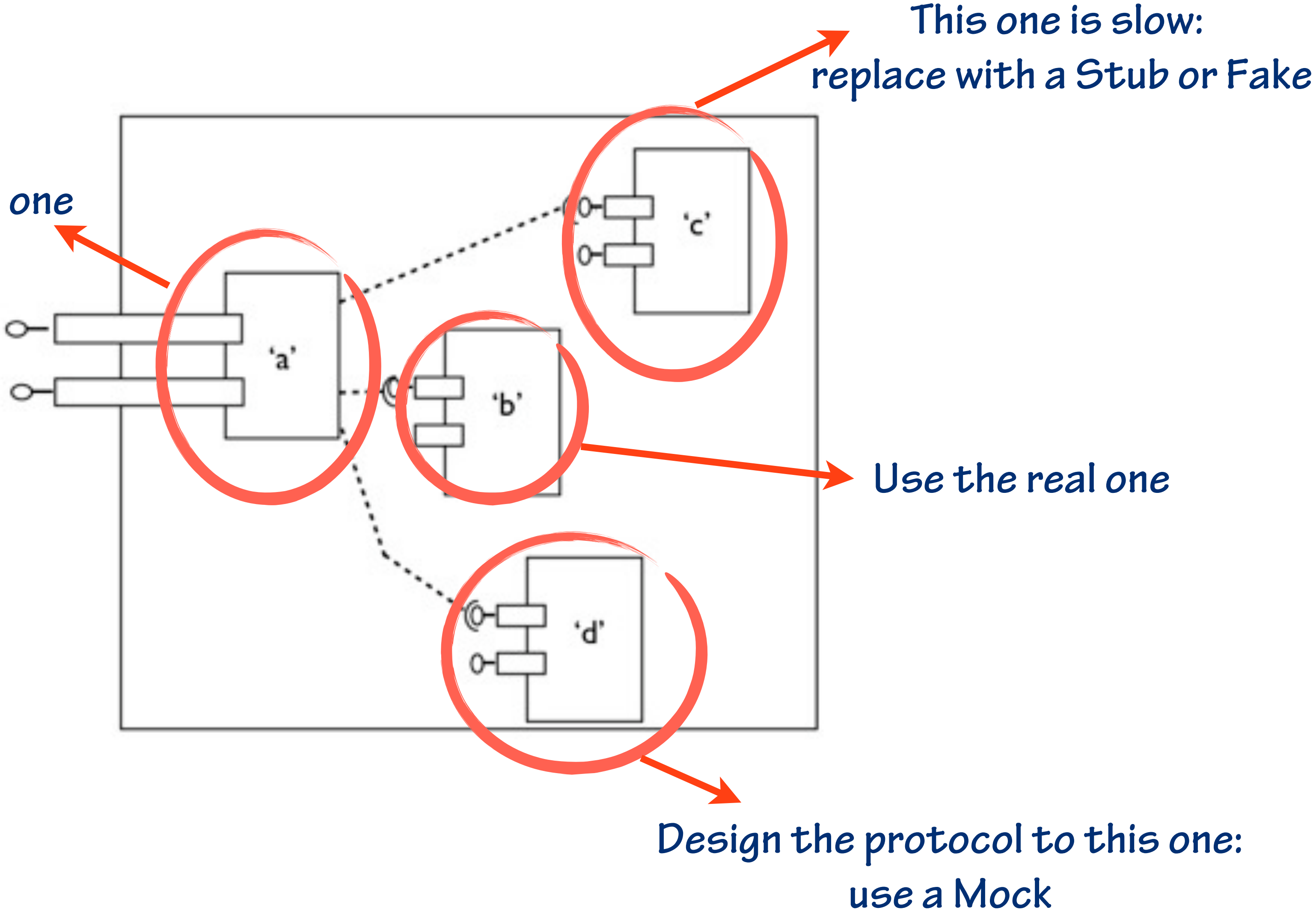


Image attribution: semlabs.co.uk, FatCow

Isolation


This one is slow:
replace with a Stub or Fake
Test This one
'a'
'b'
'c'
'd'
Use the real one
Design the protocol to this one:
use a Mock

Monkeypatching

- Changing code at runtime!

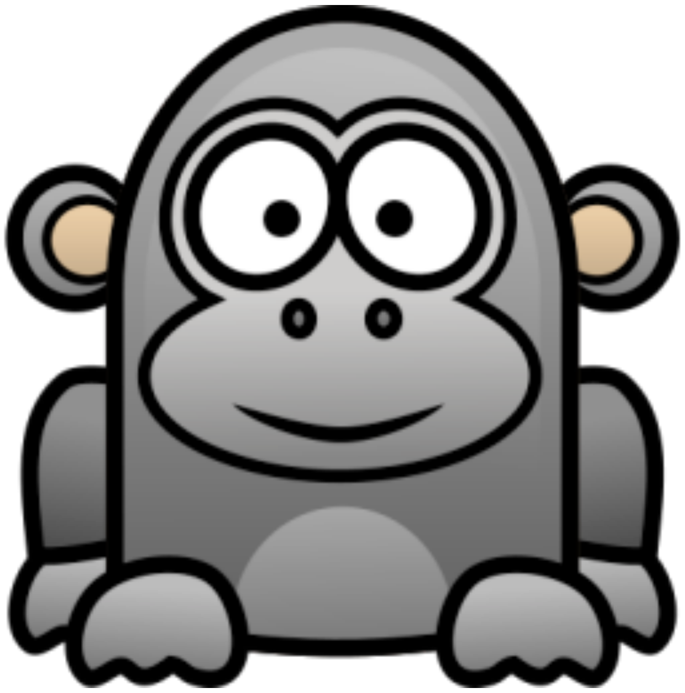


Image by Martin Berube

Module Review

- What is a Test Double?
  - Stubs
  - Fakes
  - Mocks
  - Spies
  - Dummy Objects
- Why use Test Doubles?
  - isolation
  - speed
  - design
- Monkeypatching
  - to insert a test double