Testing with Test Doubles

Budapest, 25. May 2021

Lapos Zsófia

in lapos-zsófia

Content

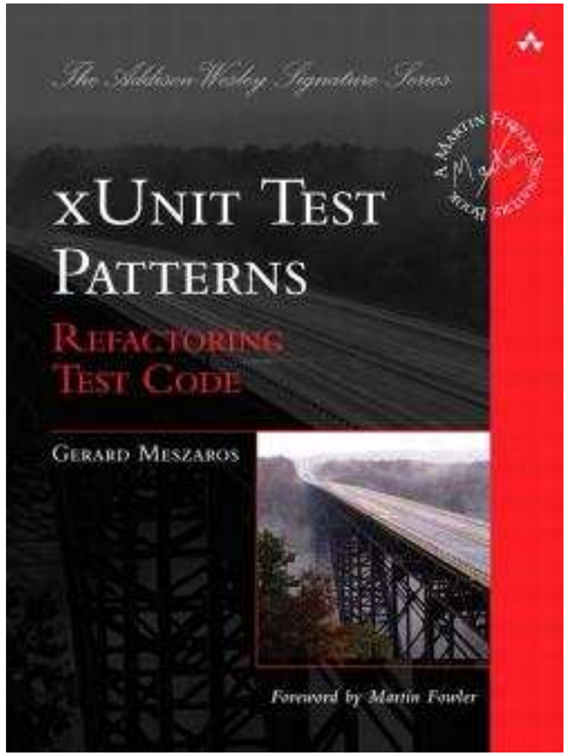
- Definition
- When & How?
- Types Of Test Doubles
- Conclusion

Definition

- In automated testing it is common to use objects that **look** and **behave** like their production equivalents but are actually simplified.
- This reduces complexity, allows to verify code independently from the rest of the system and sometimes it is even necessary to execute self validating tests at all.

A **Test Double** is a generic term used for these objects.

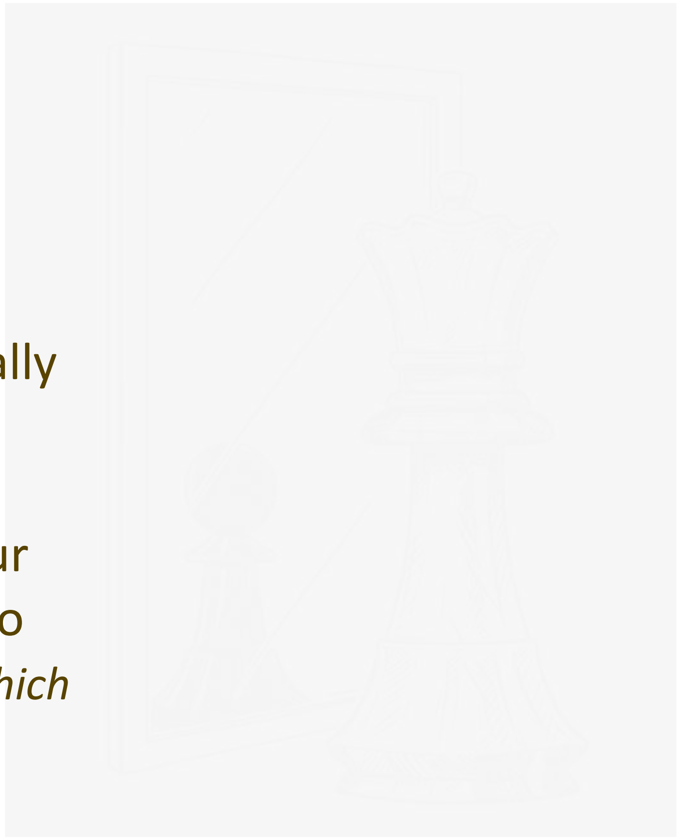
The term **Test Double** comes from Gerard Meszaros's xUnit Test Patterns book.
He describes them as:
"any object or component that we install in place of the real component for the express purpose of running a test".


The Addison-Wesley Signature Series
A Martin Fowler Signature Book
xUnit Test Patterns
Refactoring Test Code
Gerard Meszaros
Foreword by Martin Fowler

When & How?

- When practicing TDD, it’s important to ensure that our unit tests actually test just a single unit (*often a single class*) of our codebase.

- Test doubles help isolate our unit tests, they can also help speed up our tests by avoiding costly or slow processes, such as emitting a request to an actual API (*which you may or may not own*) or querying a database (*which may contain production data or need to be seeded*).



A unit of software

- Can be a **query** → returns a response, free of side effects
- Can be a **command** → changes the state of a system, but do not returns a value

xUnit Test Patterns: Test Double

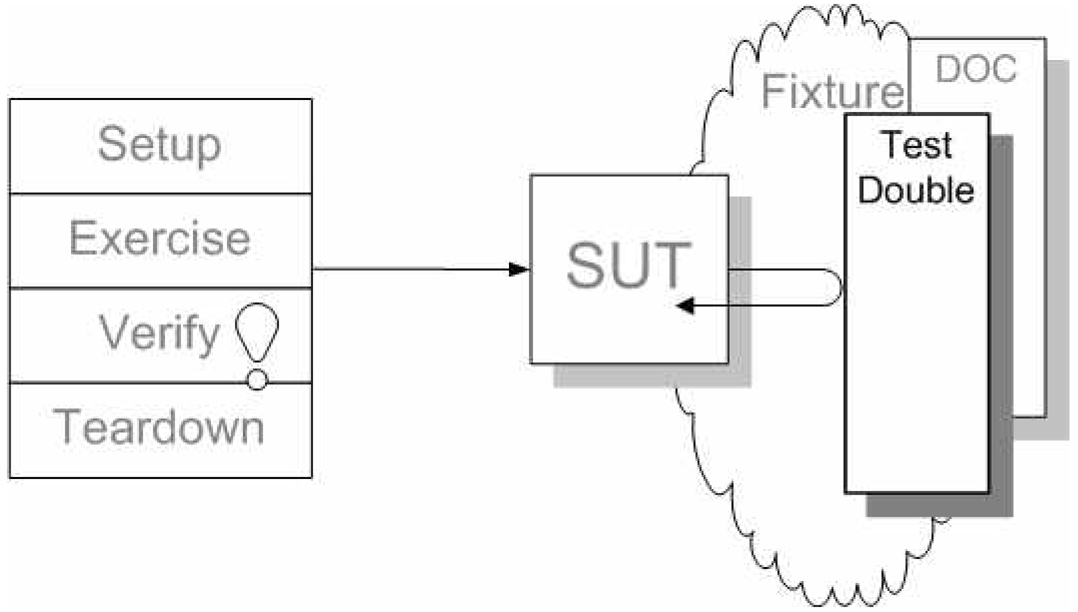
In Gerard's book the term **Test Double** appears as a Superclass:

→ SUT: System under test (=UUT)
→ DOC: Depended-on Component

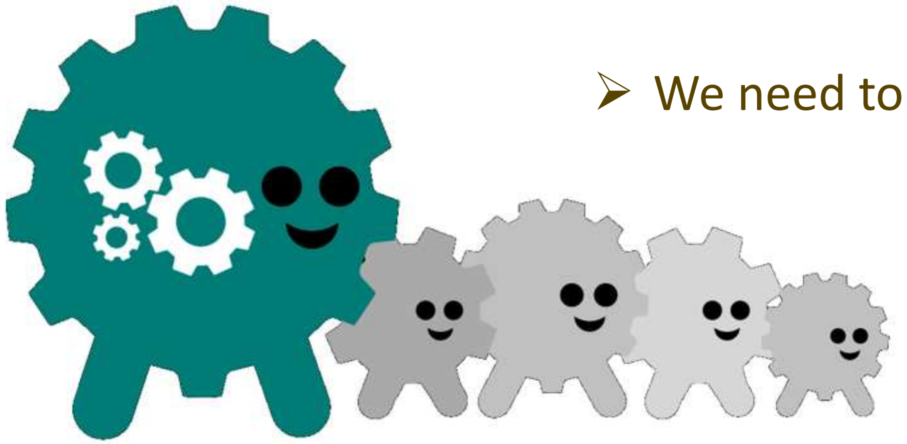
When do we use them?

→Slow tests
→DOC is
- not available
- not under test control
- has side-effects

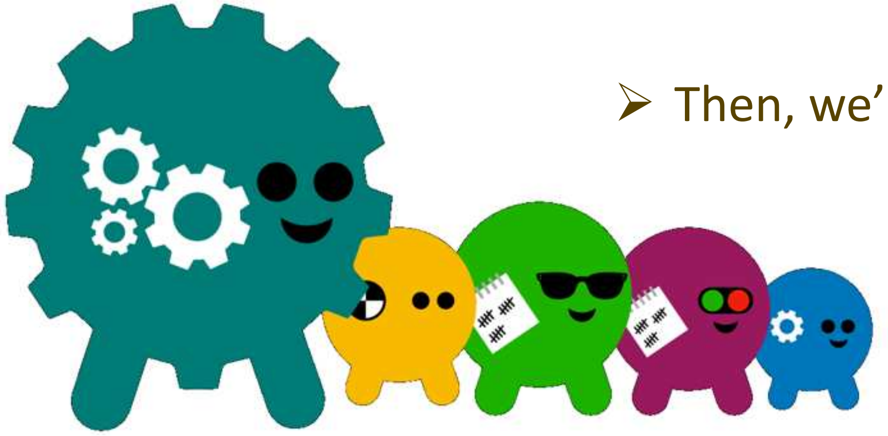
Solution: Replace DOC with a **Double**



- We want to test an object that uses collaborators…

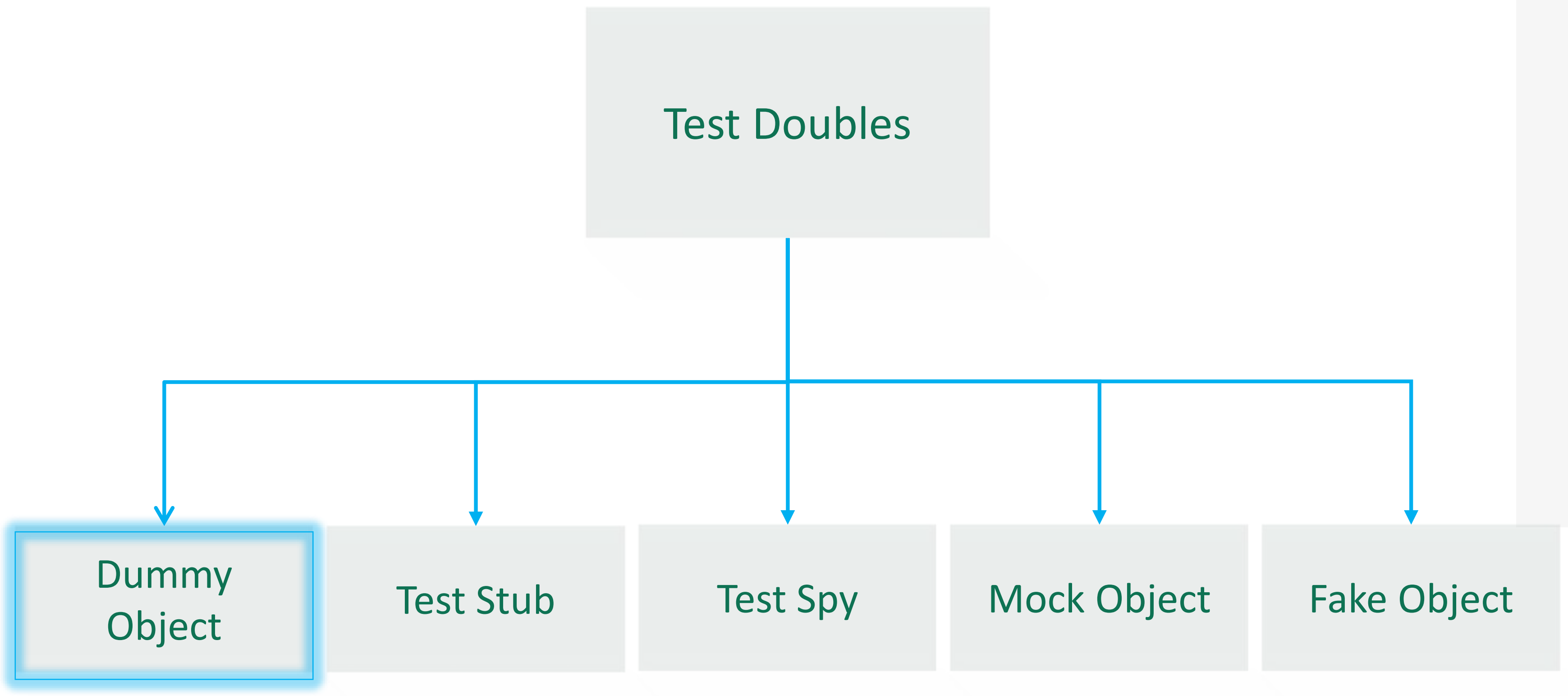


- We need to isolate its behavior from that of its collaborators



- Then, we'll need… **Test Doubles**!

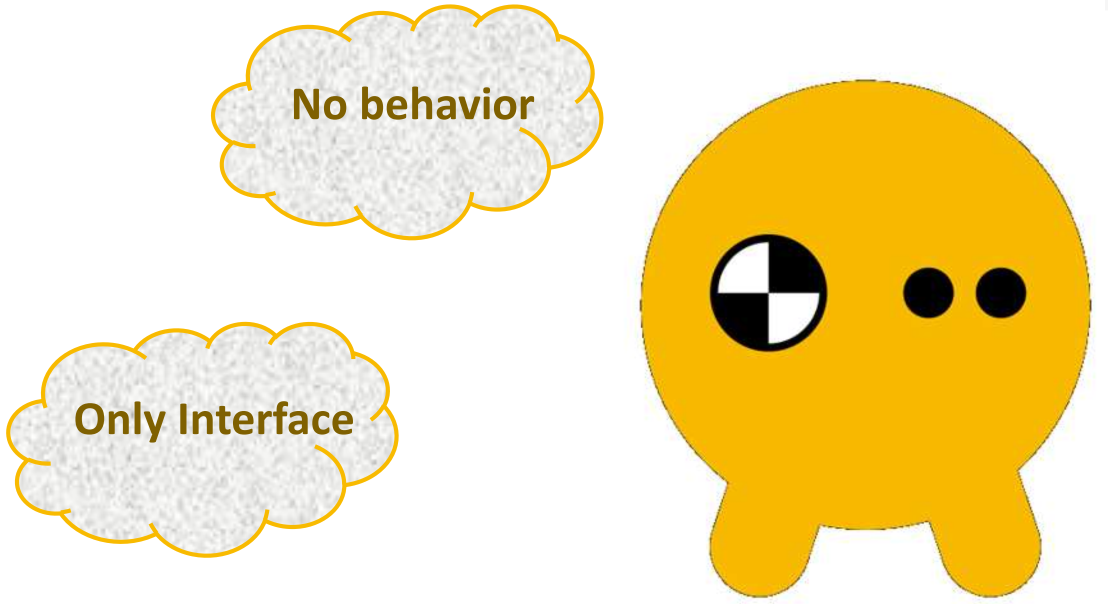
Types Of Test Doubles



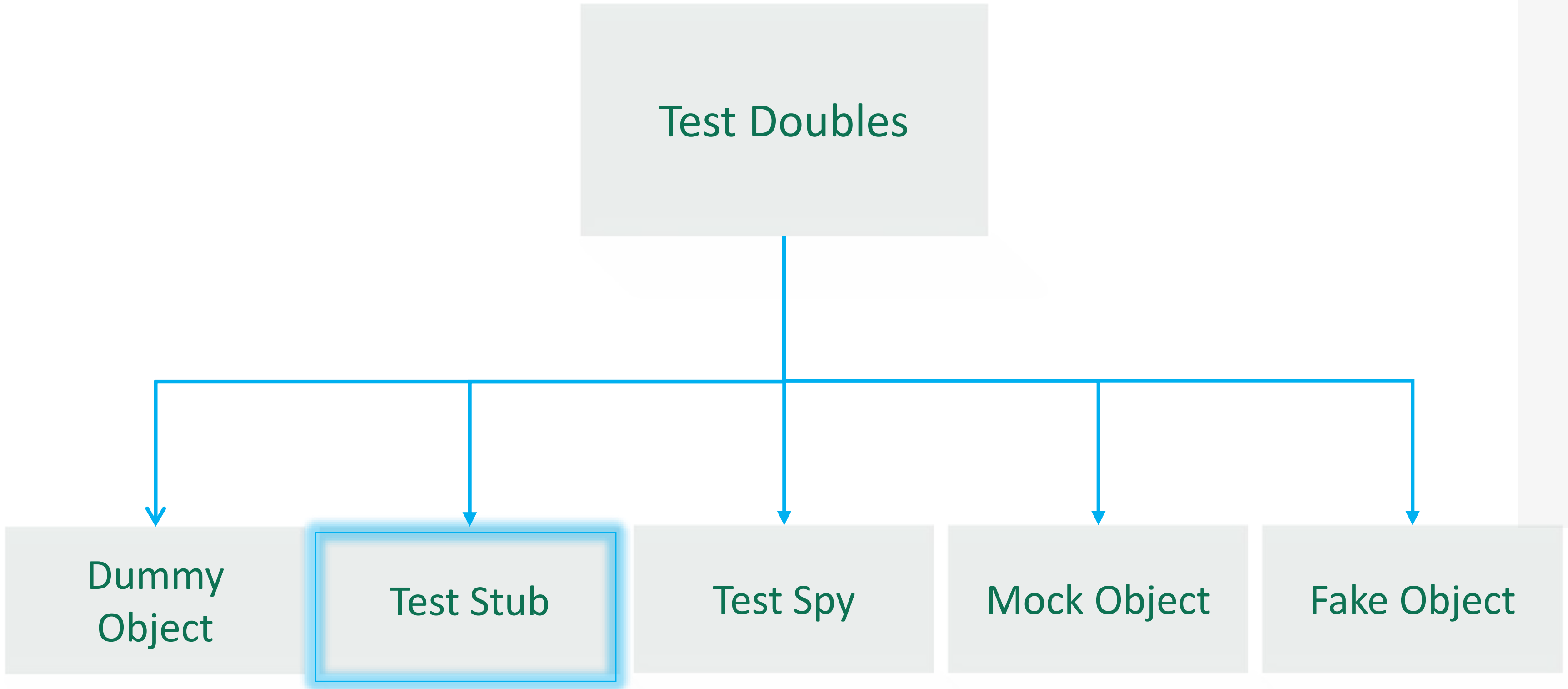
Meszaros classify several types of doubles according to the specific testing perspective

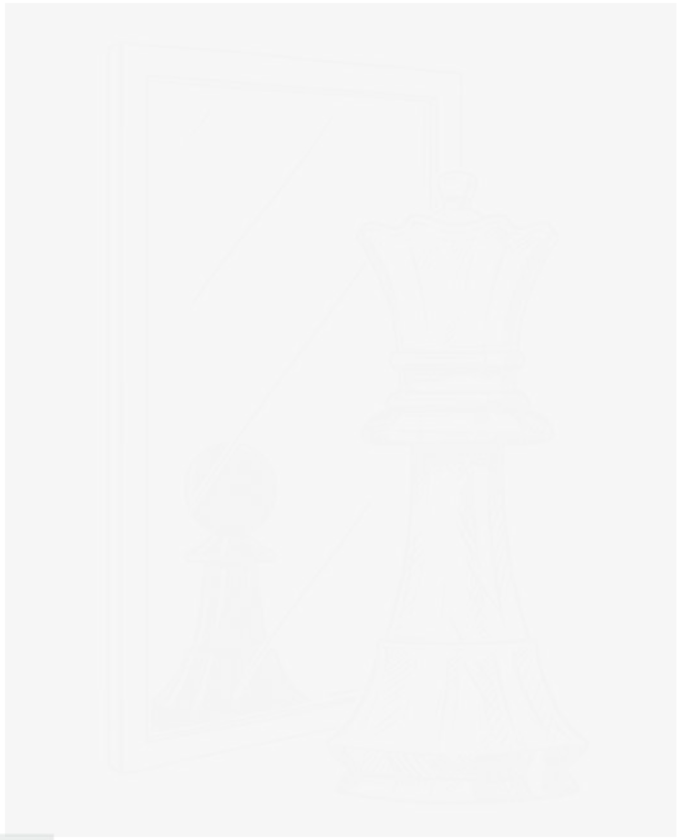
Dummy Object

- Some method signatures of the SUT may require objects as parameters
- The dummies are objects that our System depends on, but they are never used
- It contains next to nothing, basically just enough to get our code to compile
- We don’t care about them because they are irrelevant in the test scope



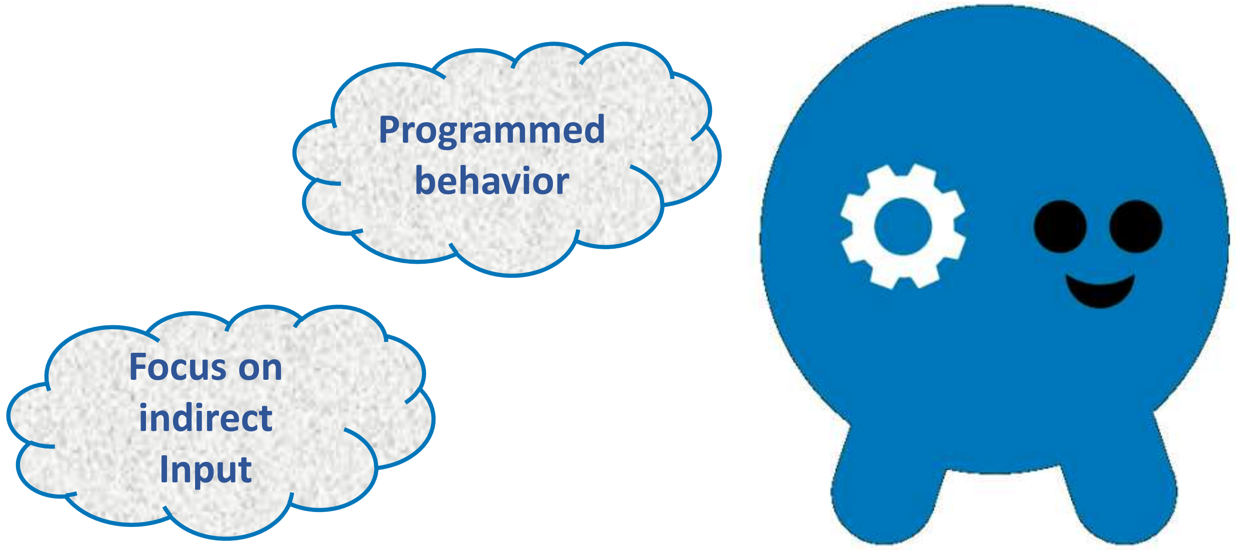
Types Of Test Doubles



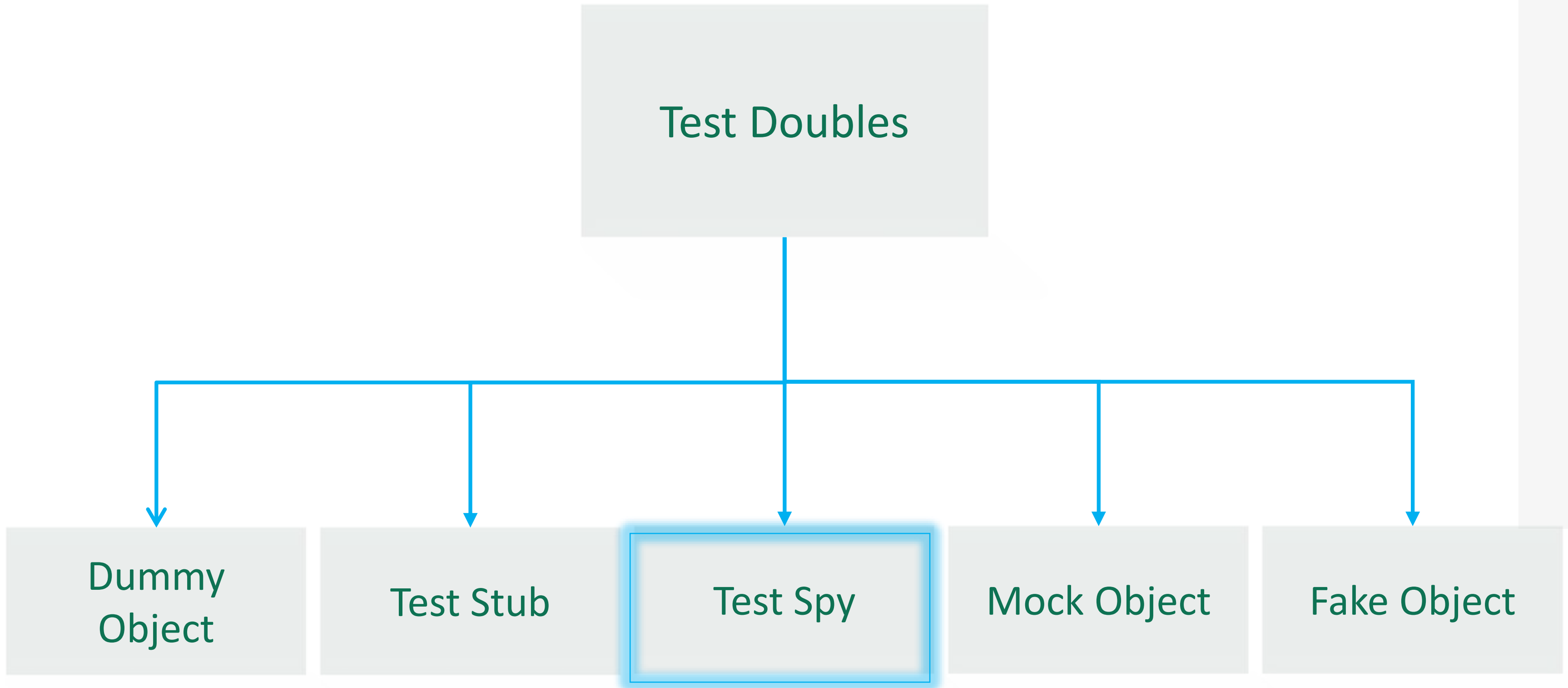


Test Stub

- A stub is a function that replaces a real implementation of an existing function
- Can be told to return a specified fake value when a given method is called
- If our test subject requires a companion object to provide some sort of data, we can use a stub to "stub out" that data source and return consistent fake data in our test setup
- Stubs are often configured at the beginning of a test with the values the programmer wishes to be returned

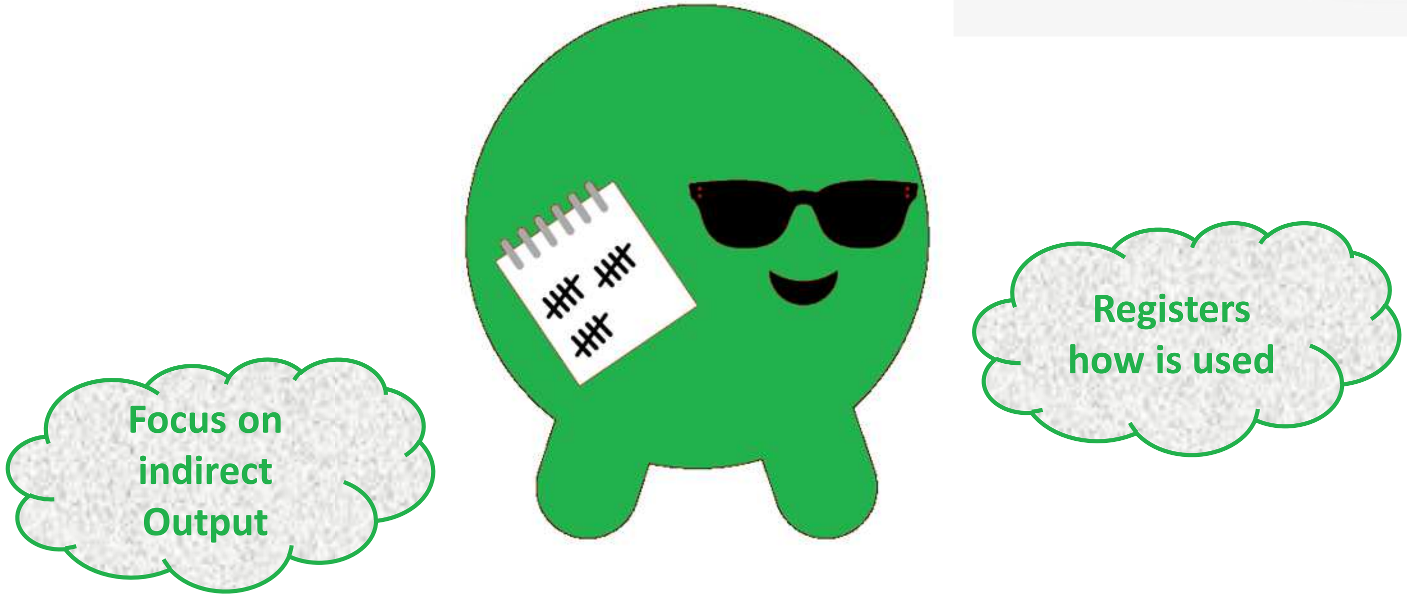


Types Of Test Doubles

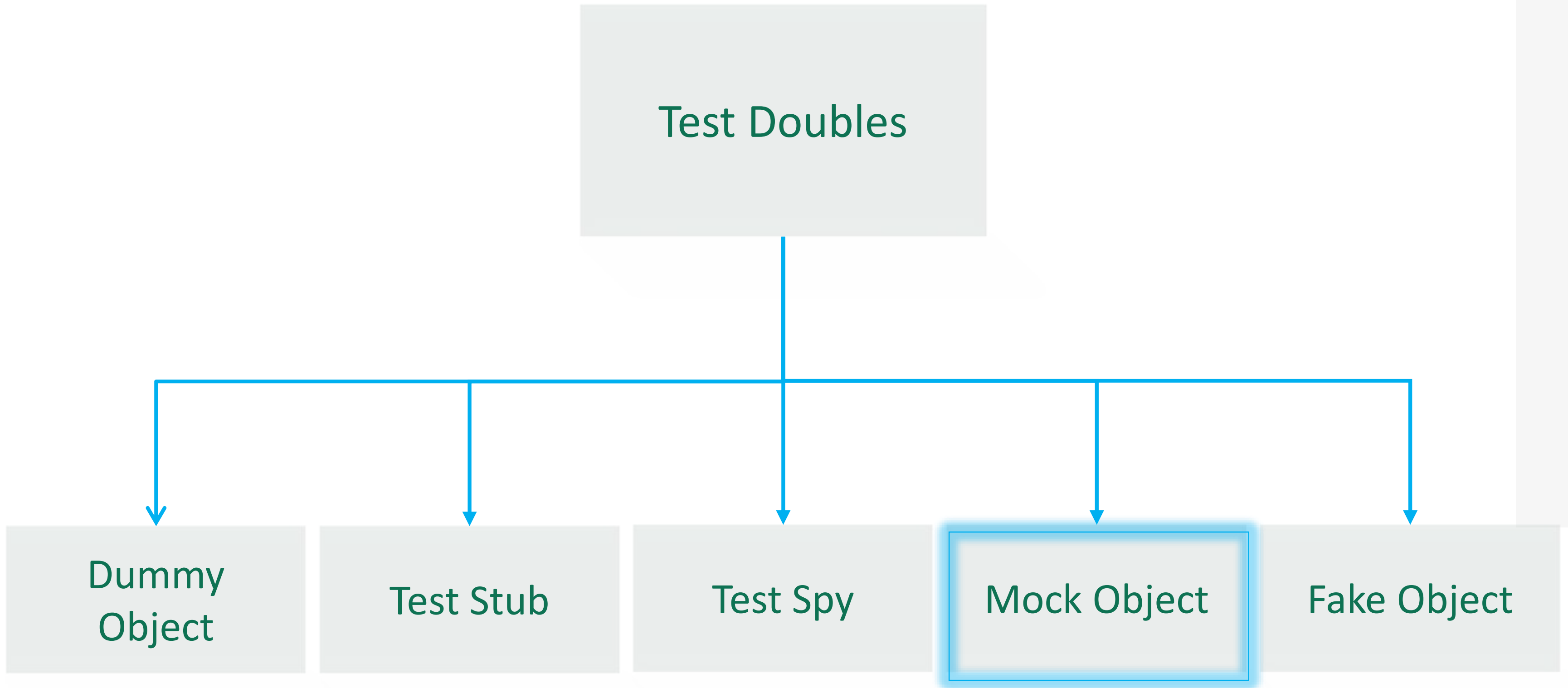


Test Spy

- To get good enough visibility of the indirect outputs of the SUT, we may have to replace some of the context with something we can use to capture these outputs
- A Test Spy can:
  - record the parameters passed to it
  - verify the order in which DOC methods were called
- Does not fail, it merely records interactions
- Is inspected after the test execution in order to verify that indirect output was correct

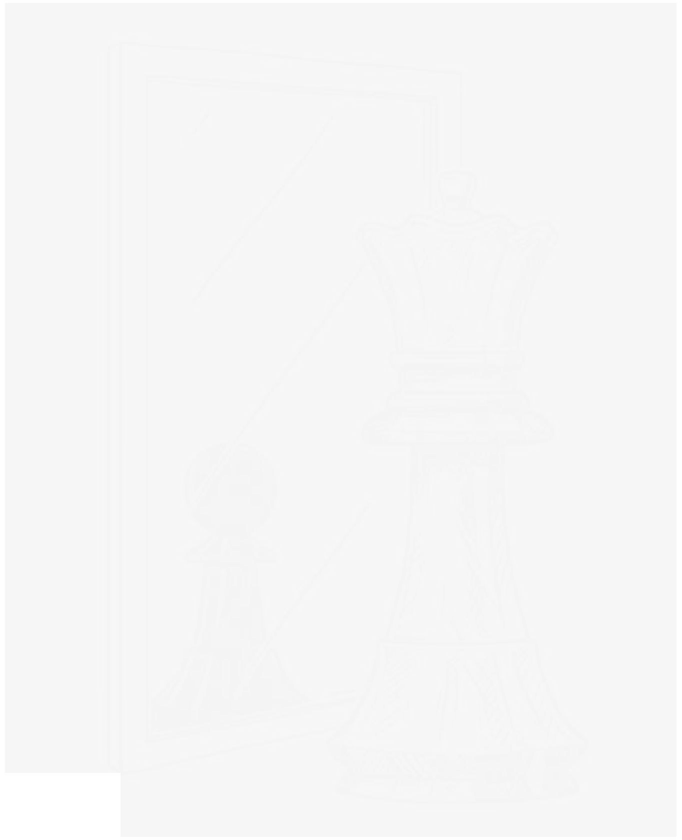


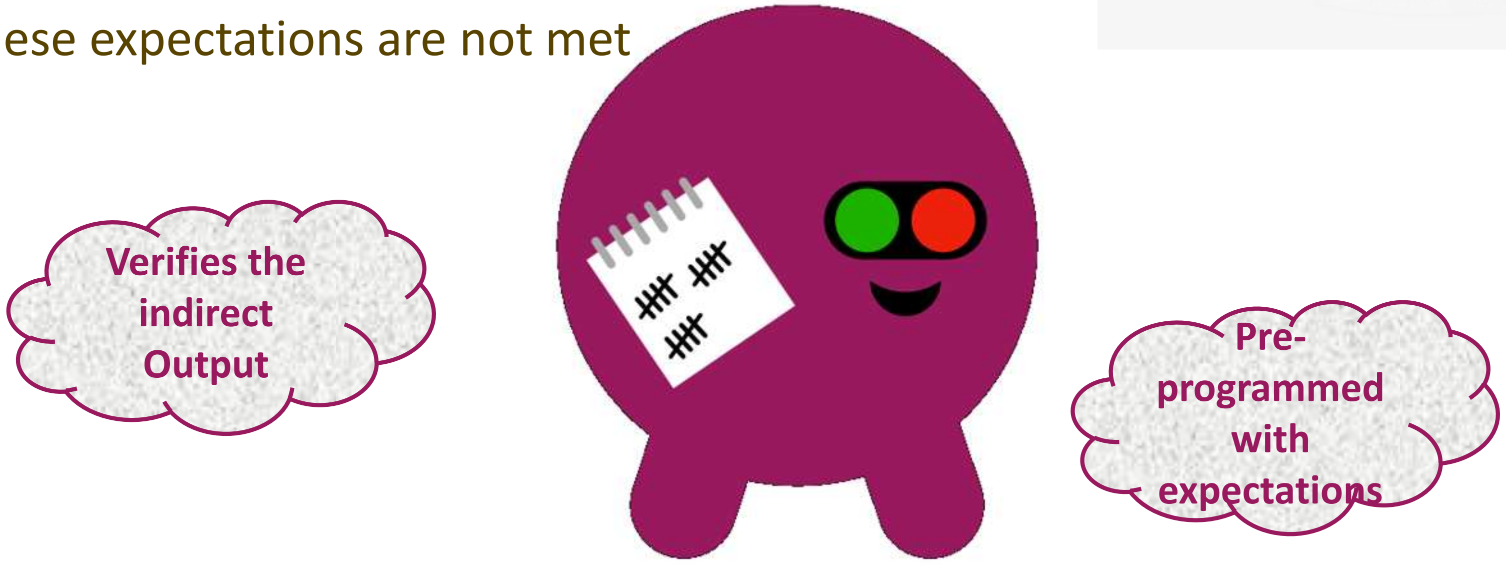
Types Of Test Doubles



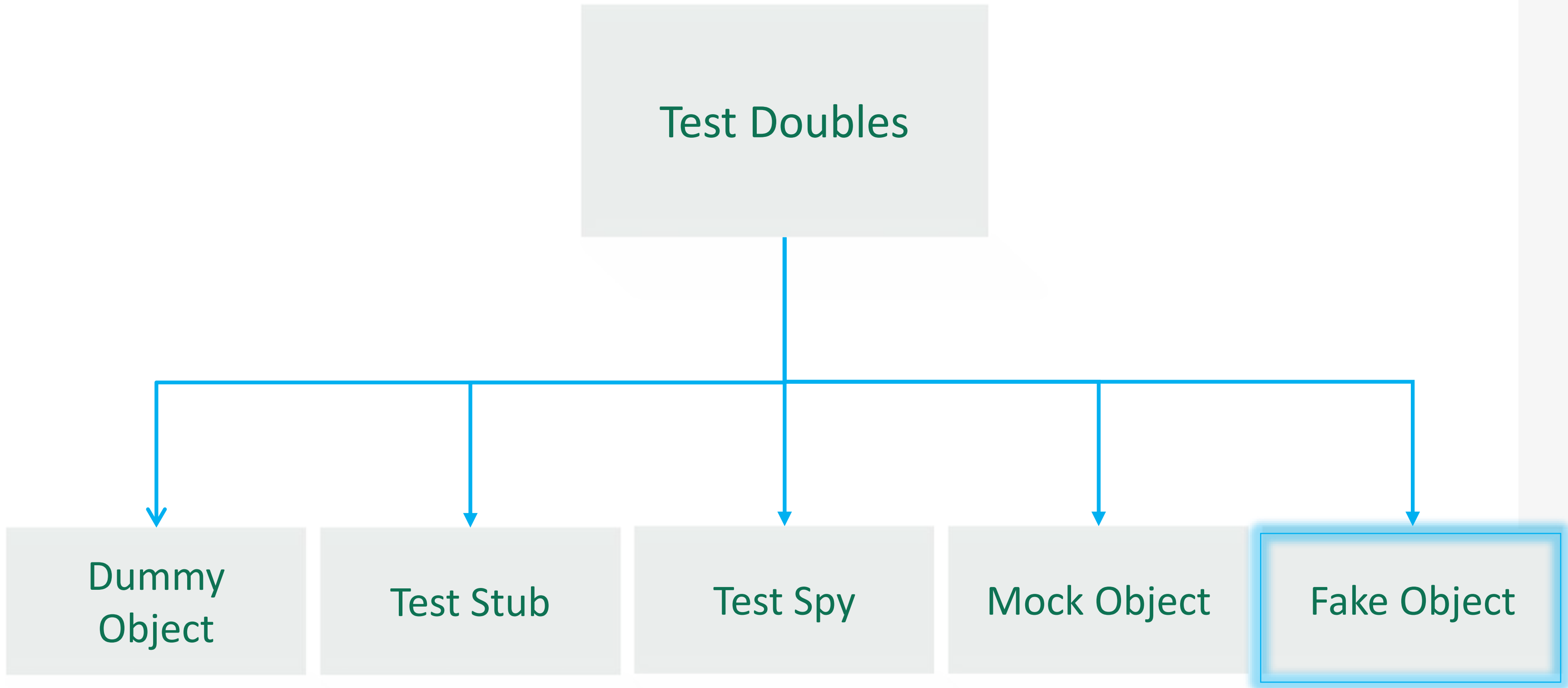
Mock Object

- Pre-programmed objects with expectations which form a specification of the calls they are expected to receive
- Mocks are useful for testing interactions with objects that should not be accessed from a unit test: web service, file system, UI.
  - Define Mock object with same interface as DOC
  - Configure mock with **expectations**
    - values to return (like test stub)
    - the methods that must be called
  - The mock will **fail** if these expectations are not met



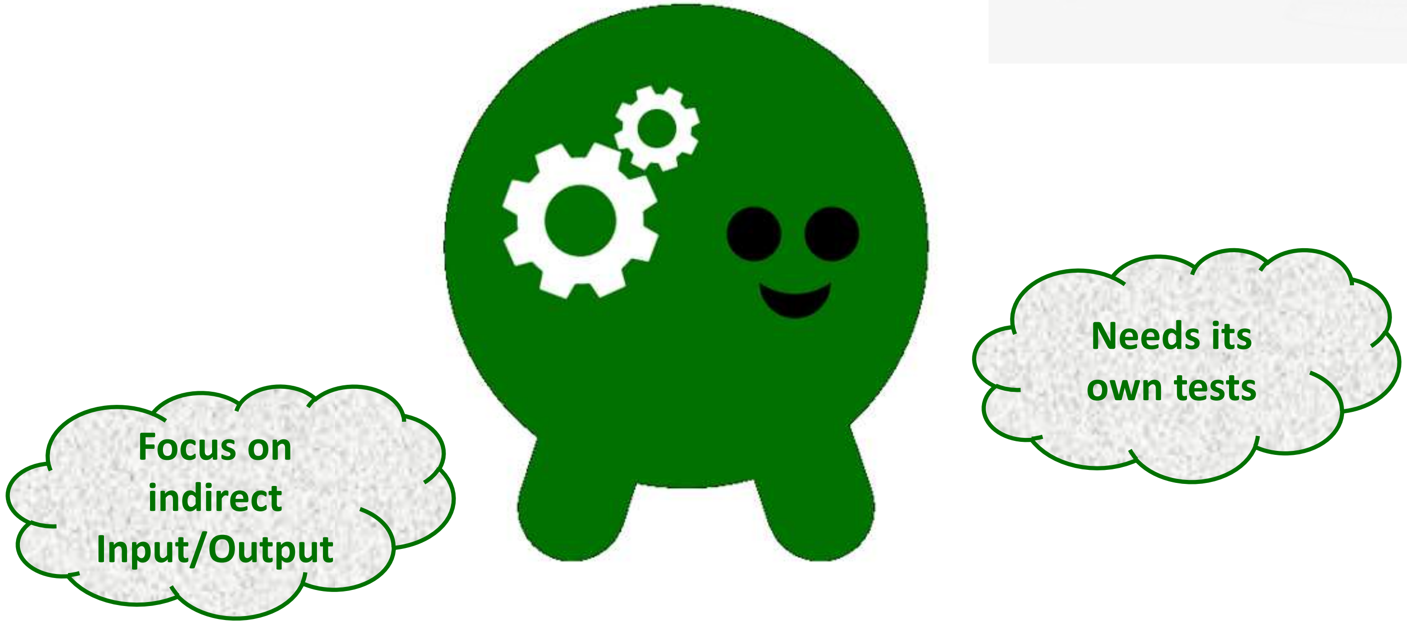

Verifies the indirect Output
Pre-programmed with expectations

Types Of Test Doubles

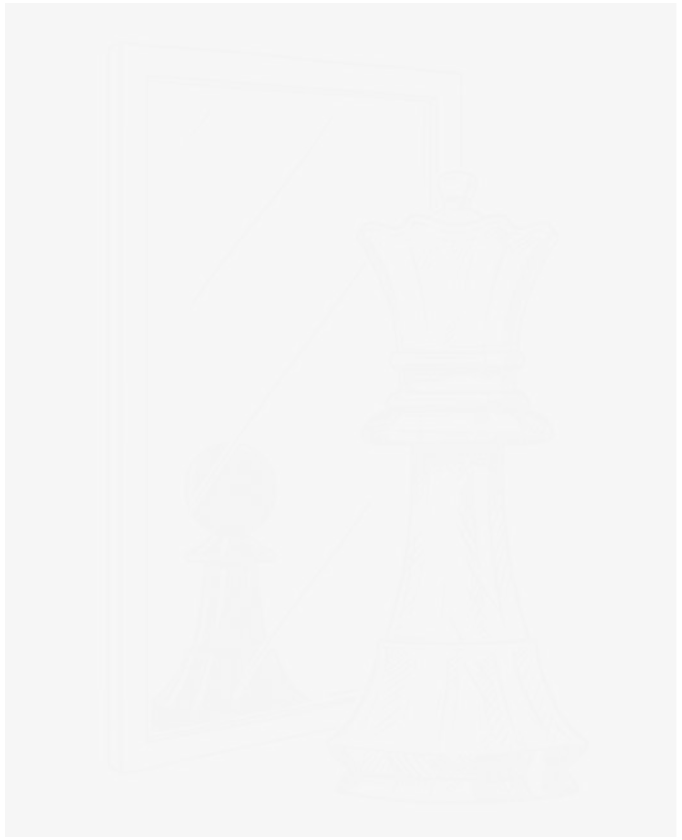


Fake Object

- Fake objects actually have working implementations, but usually take some shortcut which makes them not suitable for production
- Are used when we want to test an infrastructural class, in other words, fakes are for the classes which are beyond our application limit
- Typically, it implements the same functionality as the real DOC but in a much simpler way.

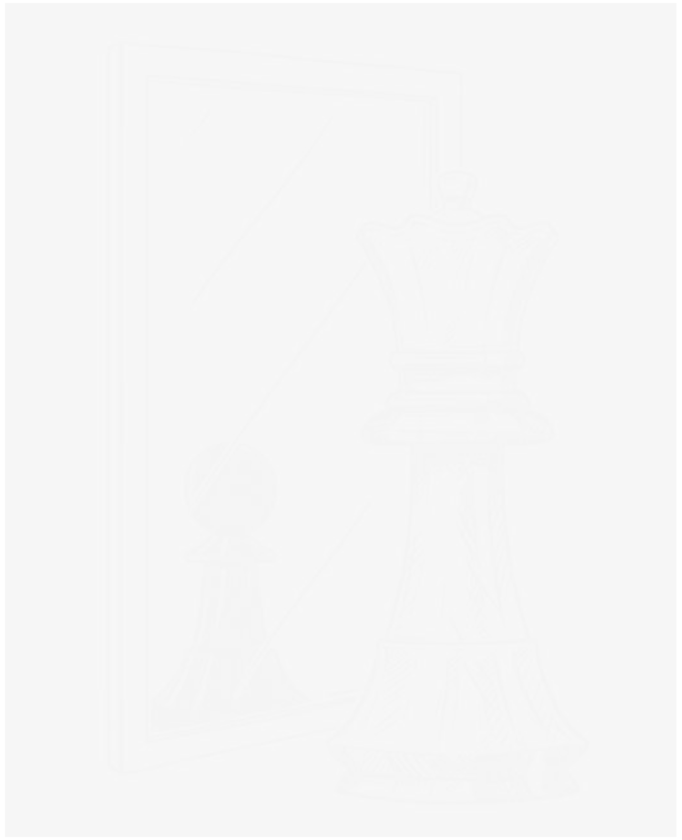


Conclusion



- Tests are software, too. They also can have bugs, and cause making changes hard, when we write too many, too detailed tests.
- Test Doubles are an integral part of unit testing. Mocks, Stubs and Dummies are all useful tools to have, and understanding the difference is important.
- Each replaces a real object in the test environment, but the behavior can be quite different.
- We must know the scope of the code we are going to test to get coupled as little as possible.

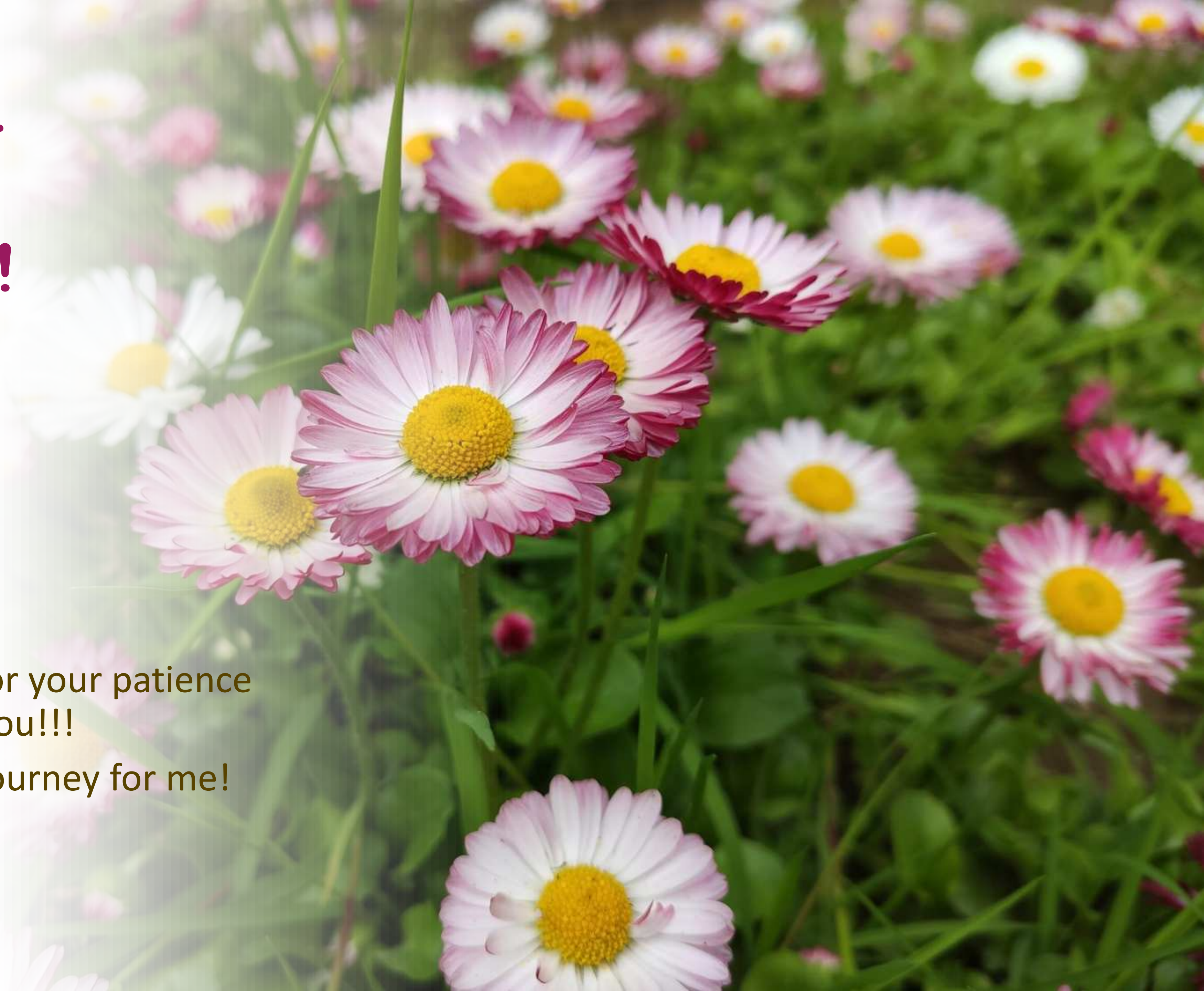
References:



Gerard Meszaros: **xUnit Test Patterns - Refactoring Test Code**
http://xunitpatterns.com/Test%20Double.html
Martin Fowler: **Mocks Aren't Stubs**
https://martinfowler.com/articles/mocksArentStubs.html
Michal Lipski : **Test Doubles** — Fakes, Mocks and Stubs
https://blog.pragmatists.com/test-doubles-fakes-mocks-and-stubs-1a7491dfa3da
Fran Iglesias: **Test Doubles** – The motion pictures
https://speakerdeck.com/franiglesias/tests-doubles-the-motion-picture

Cartoons borrowed from: https://speakerdeck.com/franiglesias/tests-doubles-the-motion-picture

Thank you for your attention!



THANKS TO:

- Alcor Academy and CSS
- all participants of the course for your patience with me, I learned a lot from you!!!
- It was awesome, an amazing journey for me!