



Test Patterns



CAUTION

Testar nem sempre é **simples**

Exemplo

Deve fazer um pedido com 3 itens em dólares

Dado um novo pedido com 3 itens associados, um Livro de \$50,00, um CD de \$20,00 e um DVD de \$30,00

Quando o pedido for realizado

Então deve ser retornado uma confirmação do pedido contendo o código, juntamente com o total do pedido de R\$550,00, se a cotação do dólar for R\$5,50 e o status aguardando pagamento



O que fazer quando **existem entradas e saídas indiretas no componente testado?**

Um **test double** é um padrão que tem o objetivo de substituir
um DOC (depended-on component) em um determinado tipo
de teste por motivos de performance ou segurança



CAUTION

Nem tudo é **mock**

The Addison-Wesley Signature Series



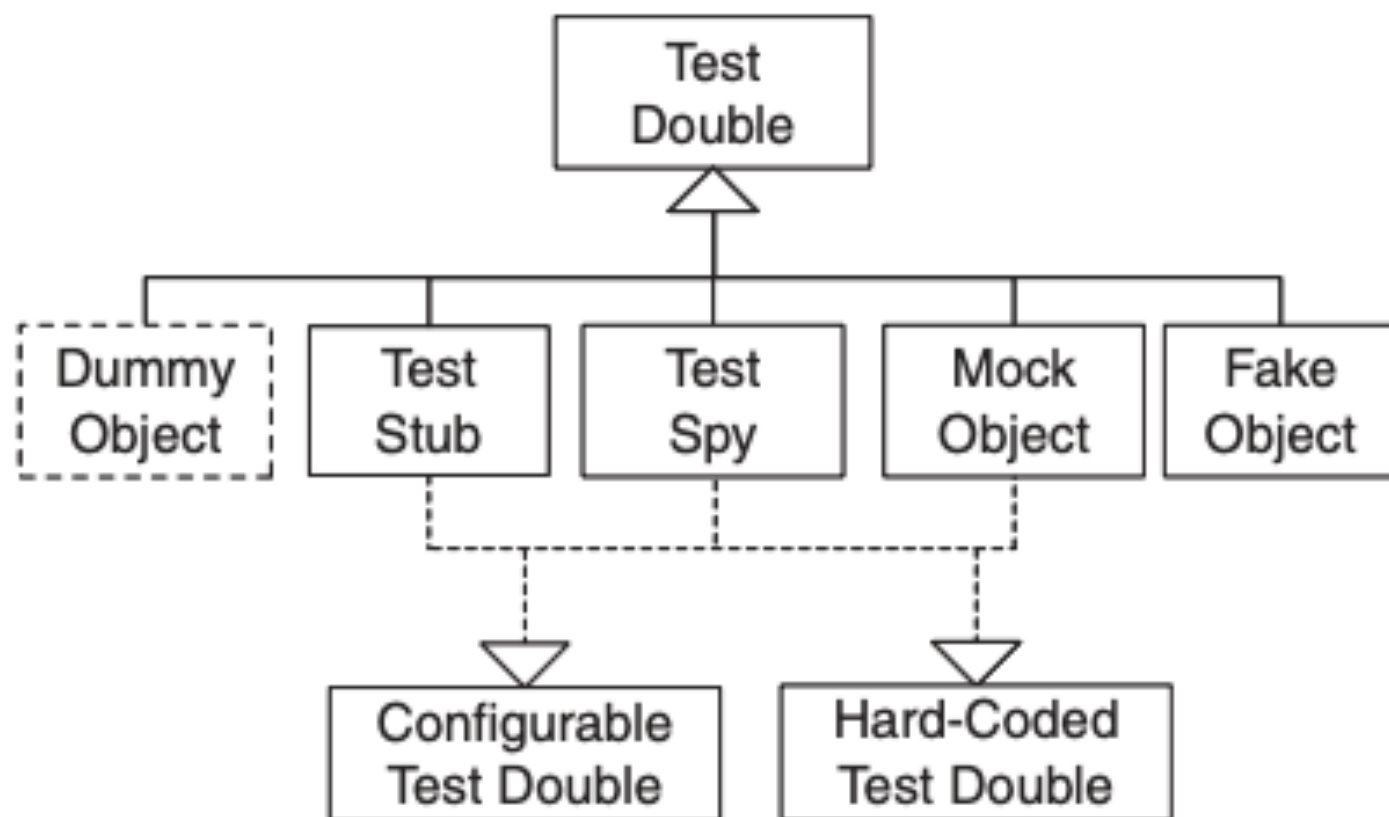
xUNIT TEST PATTERNS

REFACTORING
TEST CODE

GERARD MESZAROS



Foreword by Martin Fowler



Test Patterns

Dummy: Objetos que criamos apenas para completar a lista de parâmetros que precisamos passar para invocar um determinado método

Test Patterns

Dummy: Objetos que criamos apenas para completar a lista de parâmetros que precisamos passar para invocar um determinado método

Stubs: Objetos que retornam respostas prontas, definidas para um determinado teste, por questão de performance ou segurança (exemplo: quando eu executar o método fazer pedido preciso que o método pegar cotação do dólar retorne R\$3,00)

Test Patterns

Dummy: Objetos que criamos apenas para completar a lista de parâmetros que precisamos passar para invocar um determinado método

Stubs: Objetos que retornam respostas prontas, definidas para um determinado teste, por questão de performance ou segurança (exemplo: quando eu executar o método fazer pedido preciso que o método pegar cotação do dólar retorne R\$3,00)

Spies: Objetos que "espionam" a execução do método e armazenam os resultados para verificação posterior (exemplo: quando eu executar o método fazer pedido preciso saber se o método enviar email foi invocado internamente e com quais parâmetros)

Test Patterns

Dummy: Objetos que criamos apenas para completar a lista de parâmetros que precisamos passar para invocar um determinado método

Stubs: Objetos que retornam respostas prontas, definidas para um determinado teste, por questão de performance ou segurança (exemplo: quando eu executar o método fazer pedido preciso que o método pegar cotação do dólar retorne R\$3,00)

Spies: Objetos que "espionam" a execução do método e armazenam os resultados para verificação posterior (exemplo: quando eu executar o método fazer pedido preciso saber se o método enviar email foi invocado internamente e com quais parâmetros)

Mocks: Objetos similares a stubs e spies, permitem que você diga exatamente o que quer que ele faça e o teste vai quebrar se isso não acontecer

Test Patterns

Dummy: Objetos que criamos apenas para completar a lista de parâmetros que precisamos passar para invocar um determinado método

Stubs: Objetos que retornam respostas prontas, definidas para um determinado teste, por questão de performance ou segurança (exemplo: quando eu executar o método fazer pedido preciso que o método pegar cotação do dólar retorne R\$3,00)

Spies: Objetos que "espionam" a execução do método e armazenam os resultados para verificação posterior (exemplo: quando eu executar o método fazer pedido preciso saber se o método enviar email foi invocado internamente e com quais parâmetros)

Mocks: Objetos similares a stubs e spies, permitem que você diga exatamente o que quer que ele faça e o teste vai quebrar se isso não acontecer

Fake: Objetos que tem implementações que simulam o funcionamento da instância real, que seria utilizada em produção (exemplo: uma base de dados em memória)

Mocks Aren't Stubs


martinfowler.com

Refactoring Agile Architecture About Thoughtworks

Mocks Aren't Stubs

The term 'Mock Objects' has become a popular one to describe special case objects that mimic real objects for testing. Most language environments now have frameworks that make it easy to create mock objects. What's often not realized, however, is that mock objects are but one form of special case test object, one that enables a different style of testing. In this article I'll explain how mock objects work, how they encourage testing based on behavior verification, and how the community around them uses them to develop a different style of testing.

02 January 2007



Martin Fowler

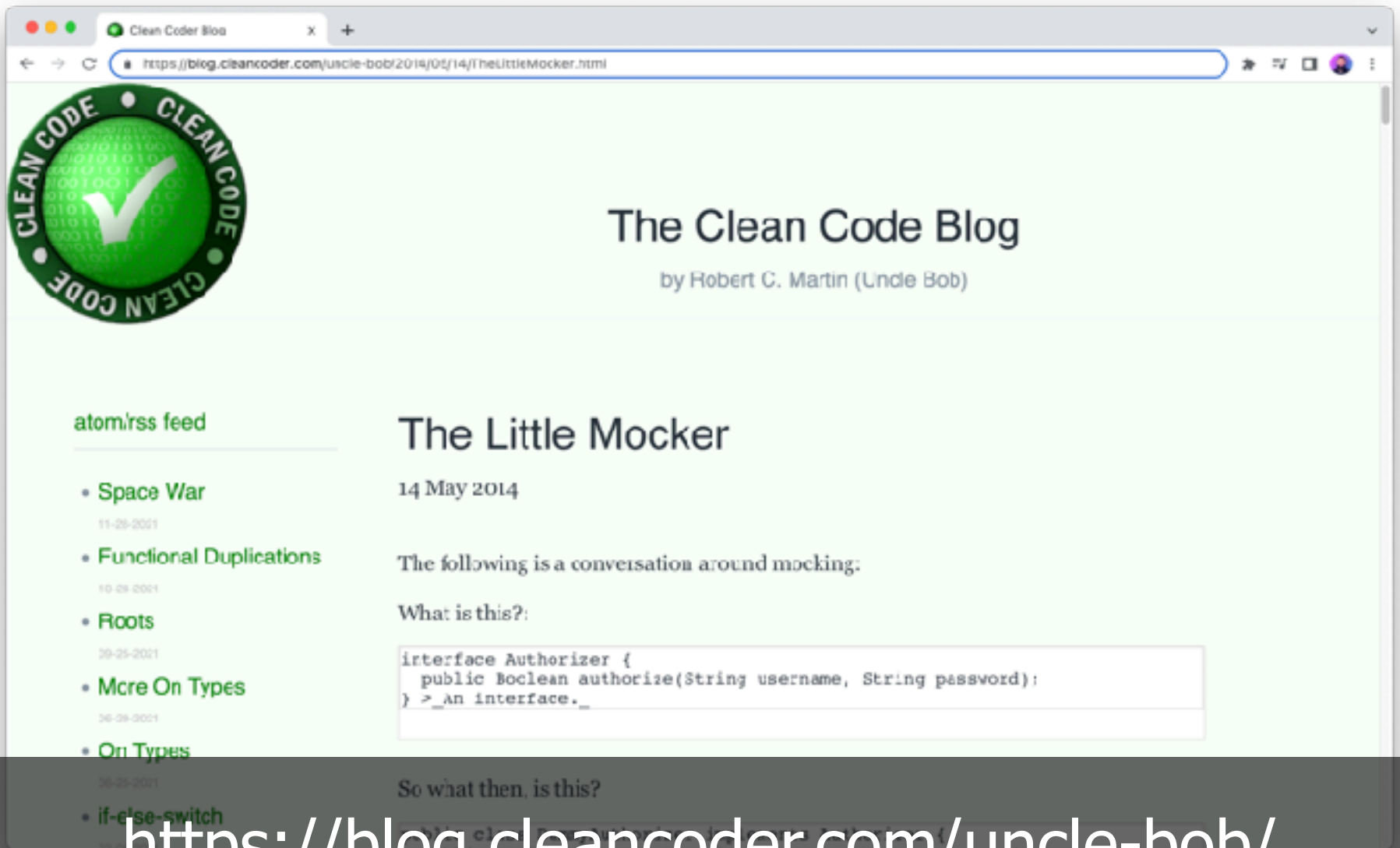
POPULAR
TESTING

CONTENTS

- Regular Tests
- Tests with Mock Objects
 - Using EasyMock
- The Difference Between Mocks and Stubs
- Classical and Mocklet Testing
- Choosing Between the Differences
 - Driving TDD
 - Fixture Setup
 - Test Isolation
 - Coupling Tests to Implementations
 - Design Style
 - Should I have a class or a mock?

Table of Contents

<https://martinfowler.com/articles/mocksArentStubs.html>



<https://blog.cleancoder.com/uncle-bob/2014/05/14/TheLittleMocker.html>