ThoughtWorks®

A Gentle Introduction

TDD IN GO

Practice, libraries and tools for
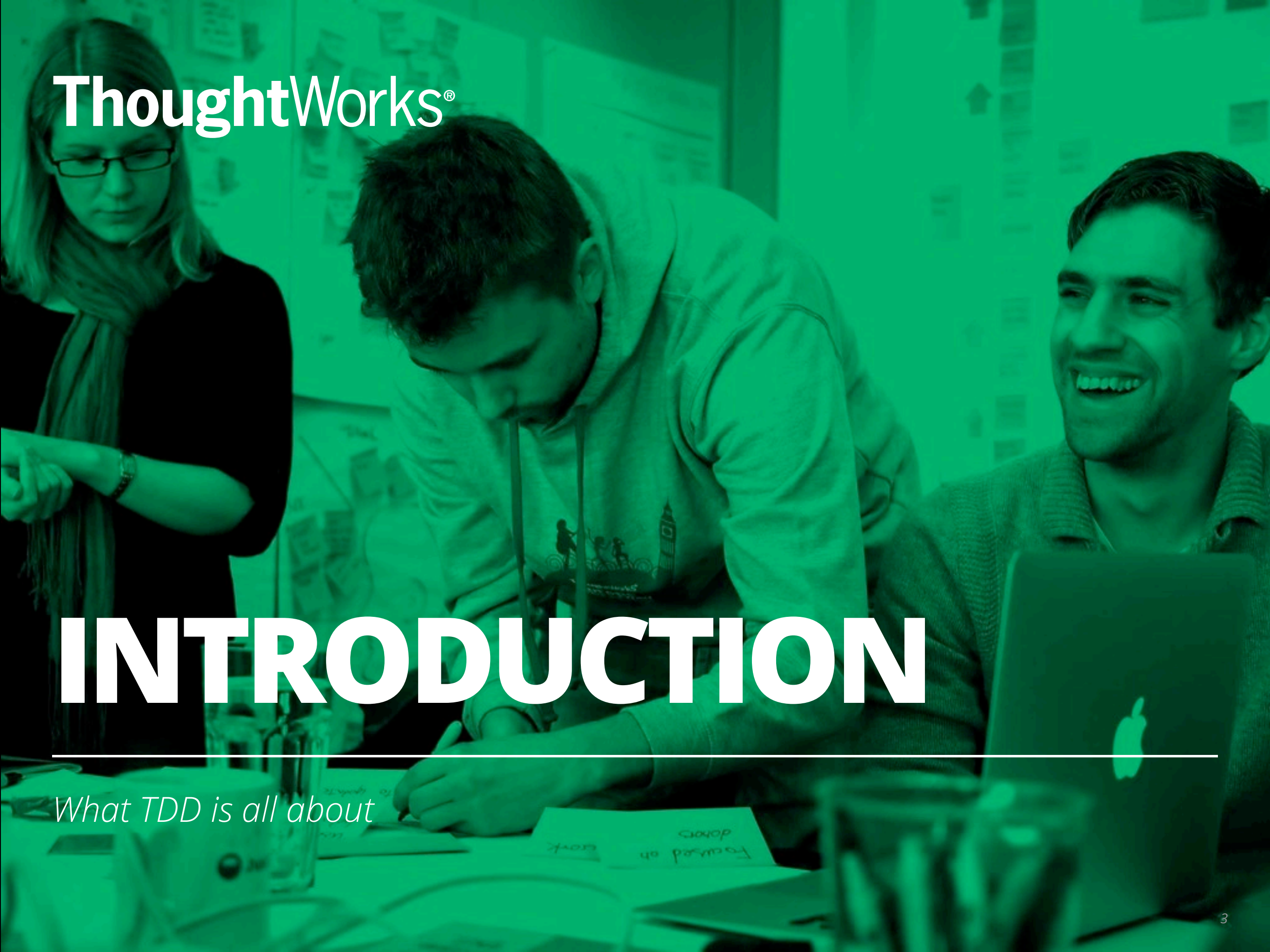*Test-Driven Development in the Go language*

Luciano Ramalho | @standupdev | @ramalhoorg

AGENDA

Brief introduction

Coding Dojo: Runes (randori style)
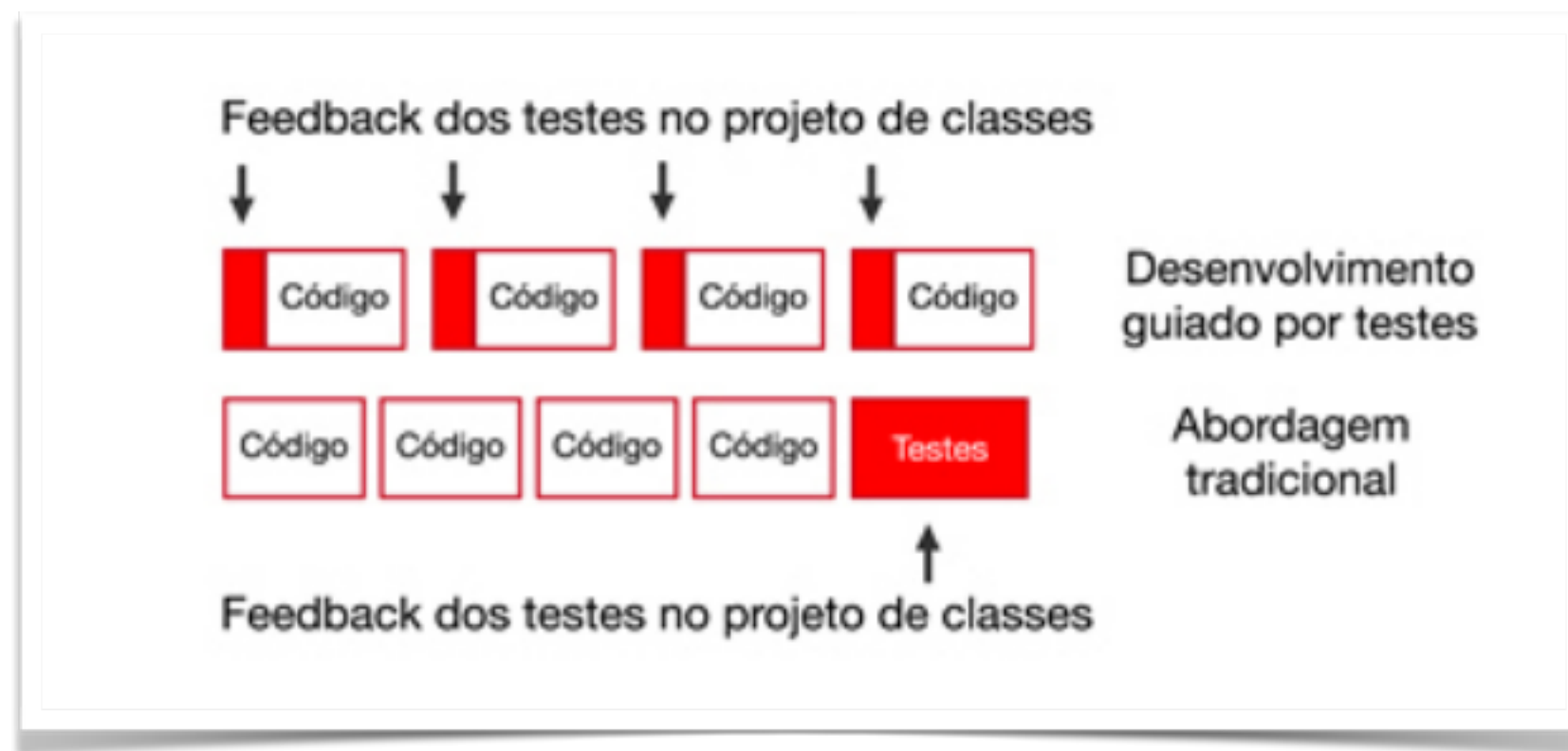
Overview of tools, techniques and libraries

INTRODUCTION

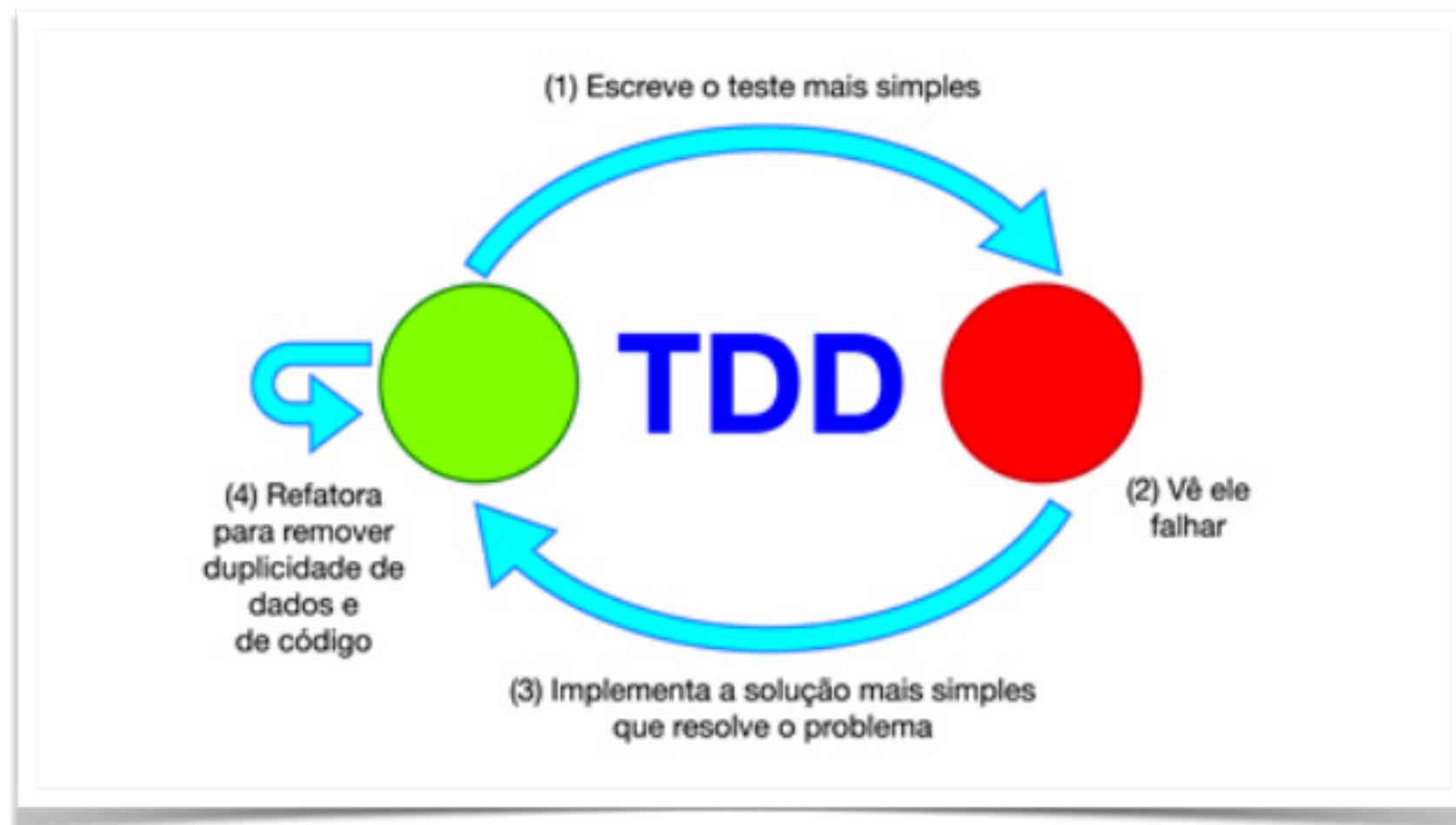*What TDD is all about*

Test-Driven Development|Design

Test-first approach

Source: **Test-Driven Development**, by Hugo Corbucci and Mauricio Aniche

TDD CYCLE

Source: **Test-Driven Development**, by Hugo Corbucci and Mauricio Aniche

TDD BEST-PRACTICE FOR PAIRING: CALL SHOT

HUGO FALA

Uma técnica divertida e muito útil quando se está usando TDD é a de, antes de rodar o teste, narrar o resultado esperado. Algo como:

- Espero que este teste falhe com uma exceção que diz que o método `maior_valor` não existe.

- Agora espero que o teste falhe dizendo que esperava 250 mas devolveu `nil`.

- Agora o teste vai passar.

Apesar de o exercício parecer fútil, falar em voz alta o resultado que esperamos nos ajuda a tomar consciência do nosso erro quando o resultado não bater. Também torna a prática de programação em pares mais divertida e garante que ambas as pessoas no par estejam acompanhando uma a outra. Nesse caso, a pessoa que não escreveu o teste é a que precisa prever o que vai acontecer.
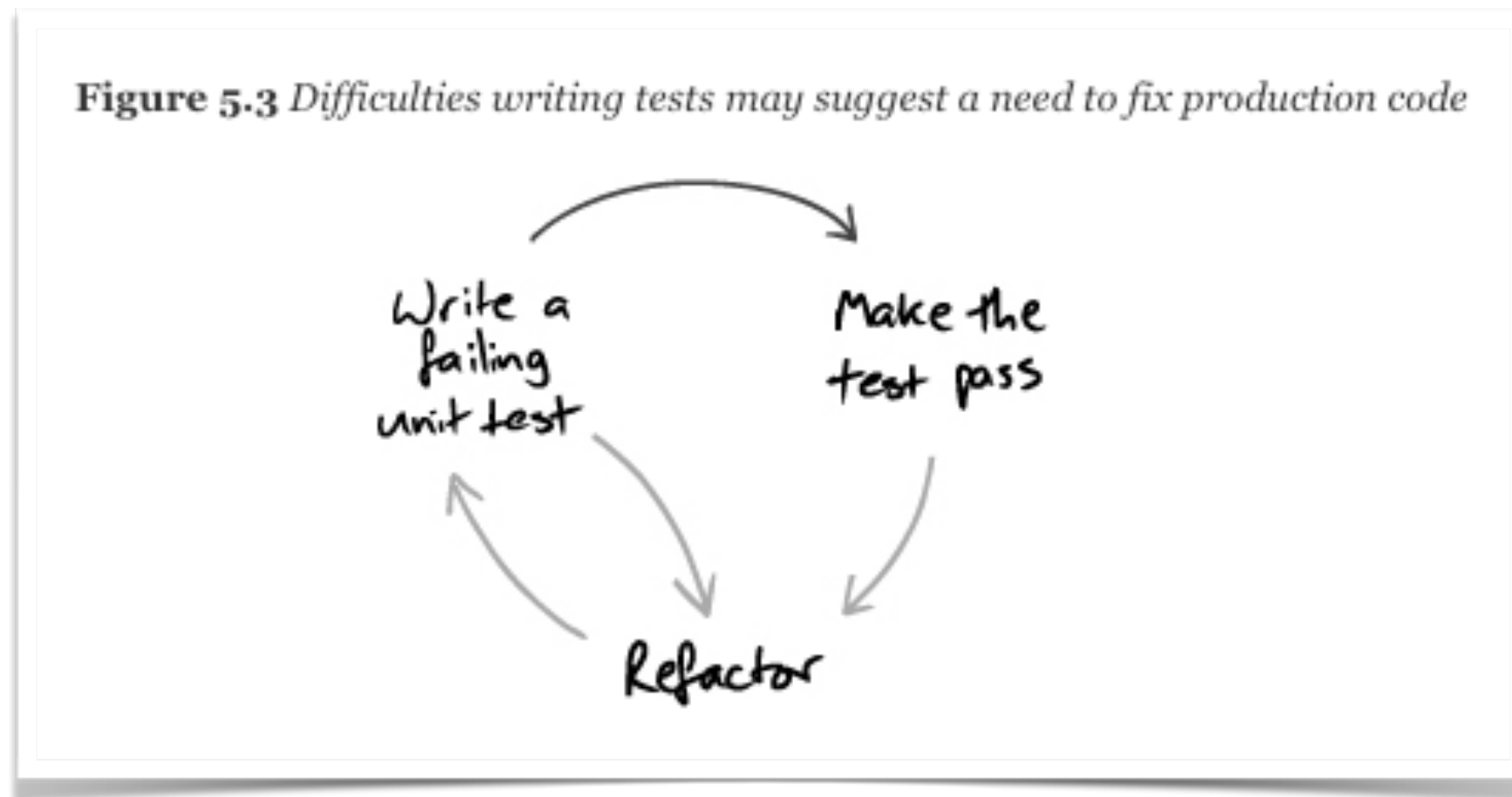
Source: **Test-Driven Development**, by Hugo Corbucci and Mauricio Aniche

TDD CYCLE: IMPROVE FAILING REPORTS

Figure 5.2 *Improving the diagnostics as part of the TDD cycle*

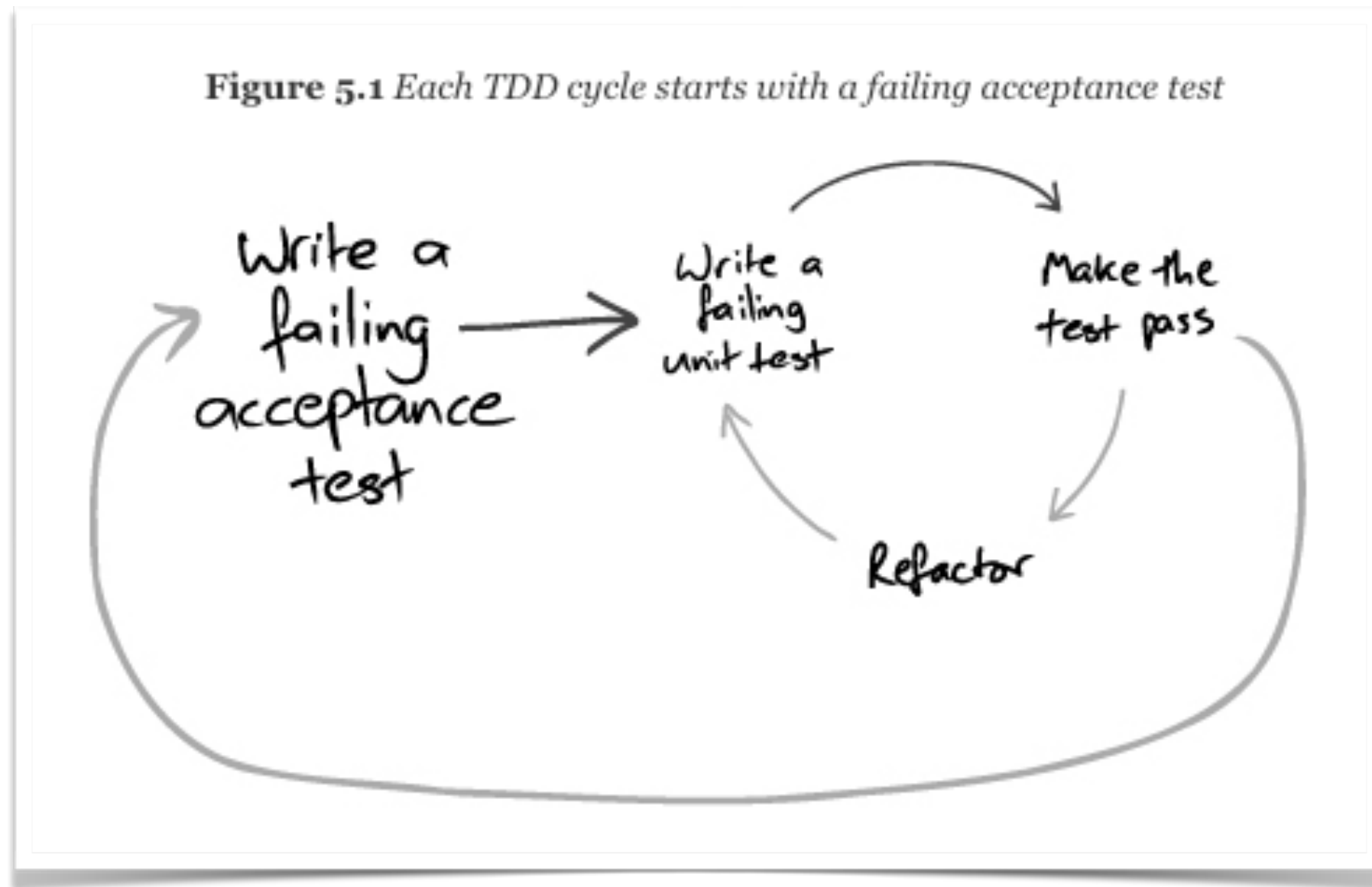Make the diagnostics clear

Make the test pass

Refactor

Write a failing test

Source: **Growing Object-Oriented Software, Guided by Tests**
by Steve Freeman, Nat Pryce

TDD CYCLE: REFACTOR AFTER TEST

Figure 5.3 *Difficulties writing tests may suggest a need to fix production code*

Write a failing unit test → Make the test pass → Refactor

Source: **Growing Object-Oriented Software, Guided by Tests**
by Steve Freeman, Nat Pryce

TDD CYCLES: MOCKIST STYLE

Figure 5.1 *Each TDD cycle starts with a failing acceptance test*

Write a failing acceptance test → Write a failing unit test → Make the test pass → Refactor

Source: **Growing Object-Oriented Software, Guided by Tests**
by Steve Freeman, Nat Pryce

TDD STYLES

Chicago style, a.k.a. "classic"

Mostly inside-out: from unit tests to acceptance tests

London style, a.k.a. "mockist"

Mostly outside-in: from acceptance tests to unit tests

MARTIN FOWLER ON TDD STYLES

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

Translations: French · Italian · Spanish · Portuguese · Korean

Find **similar articles** to this by looking at these tags: popular · testing

Contents

Regular Tests
Tests with Mock Objects
   Using EasyMock
The Difference Between Mocks and Stubs
Classical and Mockist Testing
Choosing Between the Differences
   Driving TDD
   Fixture Setup
   Test Isolation
   Coupling Tests to Implementations
   Design Style
So should I be a classicist or a mockist?

CODING DOJO

Let's practice

CODING DOJO STYLES

Randori

Pairs of pilot and co-pilot alternate in 7-minute intervals

Prepared kata

Experienced coder demonstrates via live coding

Work on small increments

Like 4L on 4WD: no need to engage at all times, but good when the going is tough

At first: practice with the smallest increments you can think

RUNNING EXAMPLE

```
$ runes cat eyes
U+1F638 😸        GRINNING CAT FACE WITH SMILING EYES
U+1F63B 😻        SMILING CAT FACE WITH HEART-SHAPED EYES
U+1F63D 😽        KISSING CAT FACE WITH CLOSED EYES
$ ▊
```

ThoughtWorks®

TOOLS

Libraries and utilities for TDD in Go

LIBRARIES FOR TESTING

Go testing libraries with most Github stars*

5251 ★	stretchr/testify
3685 ★	smartystreets/goconvey
2166 ★	onsi/ginkgo
1452 ★	golang/mock
902 ★	DATA-DOG/go-sqlmock
884 ★	gavv/httpexpect
709 ★	onsi/gomega
575 ★	google/go-cmp
512 ★	franela/goblin
502 ★	h2non/baloo
496 ★	h2non/gock
404 ★	DATA-DOG/godog
387 ★	go-check/check

TECHNIQUES

Coding tips

SIMPLE EXAMPLE-TEST

```
func Example() {
        main()
        // Output:
        // Please provide one or more words to search.
}
```

ANOTHER SIMPLE EXAMPLE-TEST

```go
func ExampleMake() {
        w := []string{"beta", "alpha", "gamma", "beta"}
        s := Make(w...)
        fmt.Println(s)
        // Output: Set{alpha beta gamma}
}
```

EXAMPLE-TEST WITH NON-DETERMINISTIC OUTPUT

```go
func ExampleSet_Channel() {
        set := MakeFromText("beta alpha delta gamma")
        result := []string{}
        for elem := range set.Channel() { // order is undefined
                result = append(result, elem)
        }
        sort.Strings(result) // must sort so example passes
        fmt.Println(result)
        // Output: [alpha beta delta gamma]
}
```

EXAMPLE-TEST WITH FAKE COMMAND-LINE ARGUMENTS

```go
func Example_2WordQuery() { // ❶
        oldArgs := os.Args // ❷
        defer func() { os.Args = oldArgs }()
        os.Args = []string{"", "cat", "smiling"}
        main() // ❸
        // Output:
        // U+1F638      😸      GRINNING CAT FACE WITH SMILING EYES
        // U+1F63A      😺      SMILING CAT FACE WITH OPEN MOUTH
        // U+1F63B      😻      SMILING CAT FACE WITH HEART-SHAPED EYES
}
```

TABLE TEST WITH SUB-TESTS

```go
func TestMake(t *testing.T) {
    testCases := []struct {
        elems   []string
        wantLen int
    }{
        {[]string{}, 0},
        {[]string{"a"}, 1},
        {[]string{"a", "b"}, 2},
        {[]string{"a", "b", "a"}, 2},
    }
    for _, tc := range testCases {
        t.Run(fmt.Sprintf("%v gets %d", tc.elems, tc.wantLen), func(t *testing.T) {
            s := Make(tc.elems...)
            assert.Equal(t, tc.wantLen, s.Len())
        })
    }
}
```

TEST WITH FAKE ENVIRONMENT VARIABLE

```go
func restore(nameVar, value string, existed bool) {
        if existed {
                os.Setenv(nameVar, value)
        } else {
                os.Unsetenv(nameVar)
        }
}

func TestGetUCDPath_isSet(t *testing.T) {
        pathBefore, existed := os.LookupEnv("UCD_PATH")
        defer restore("UCD_PATH", pathBefore, existed)
        ucdPath := fmt.Sprintf("./TEST%d-UnicodeData.txt", time.Now().UnixNano())
        os.Setenv("UCD_PATH", ucdPath)
        got := getUCDPath()
        if got != ucdPath {
                t.Errorf("getUCDPath() [set]\nwant: %q; got: %q", ucdPath, got)
        }
}
```

TEST WITH HTTP SERVER DOUBLE

```go
func TestFetchUCD(t *testing.T) {
    srv := httptest.NewServer(http.HandlerFunc(
            func(w http.ResponseWriter, r *http.Request) {
                    w.Write([]byte(lines3Dto43))
            }))
    defer srv.Close()

    ucdPath := fmt.Sprintf("./TEST%d-UnicodeData.txt", time.Now().UnixNano())
    done := make(chan bool)             // ❶
    go fetchUCD(srv.URL, ucdPath, done) // ❷
    _ = <-done                          // ❸
    ucd, err := os.Open(ucdPath)
    if os.IsNotExist(err) {
            t.Errorf("fetchUCD did not save:%v\n%v", ucdPath, err)
    }
    ucd.Close()
    os.Remove(ucdPath)
}
```

SLOW TEST THAT CAN BE SKIPPED

```go
func TestOpenUCD_remote(t *testing.T) {
	if testing.Short() { // ❶
		t.Skip("skipped test [-test.short option]") // ❷
	}
	ucdPath := fmt.Sprintf("./TEST%d-UnicodeData.txt", time.Now().UnixNano())
	ucd, err := openUCD(ucdPath)
	if err != nil {
		t.Errorf("openUCD(%q):\n%v", ucdPath, err)
	}
	ucd.Close()
	os.Remove(ucdPath)
}
```

ThoughtWorks®

REFERENCES

Where to learn more

REFERENCES

MORE REFERENCES

Books

Kent Beck: **Test Driven Development: By Example** https://tgo.li/2NvBfcX

Steve Freeman, Nat Pryce:
Growing Object-Oriented Software, Guided by Tests https://tgo.li/2tV8QoK

Posts|Videos

Martin Fowler: **Mocks Aren't Stubs** https://tgo.li/2lUqTXv

Martin Fowler: **Is TDD Dead?** https://tgo.li/2IWOAYn

Michael Feathers , Steve Freeman:
Test Driven Development: Ten Years Later https://tgo.li/2KD2Gnm

Stanislav Pankevich:
Notes on "TDD by Example" by Kent Beck https://tgo.li/2ufKWDN

THANK YOU

Let's connect!

Luciano Ramalho
@standupdev | @ramalhoorg
luciano.ramalho@thoughtworks.com

ThoughtWorks®