cucumber ltd

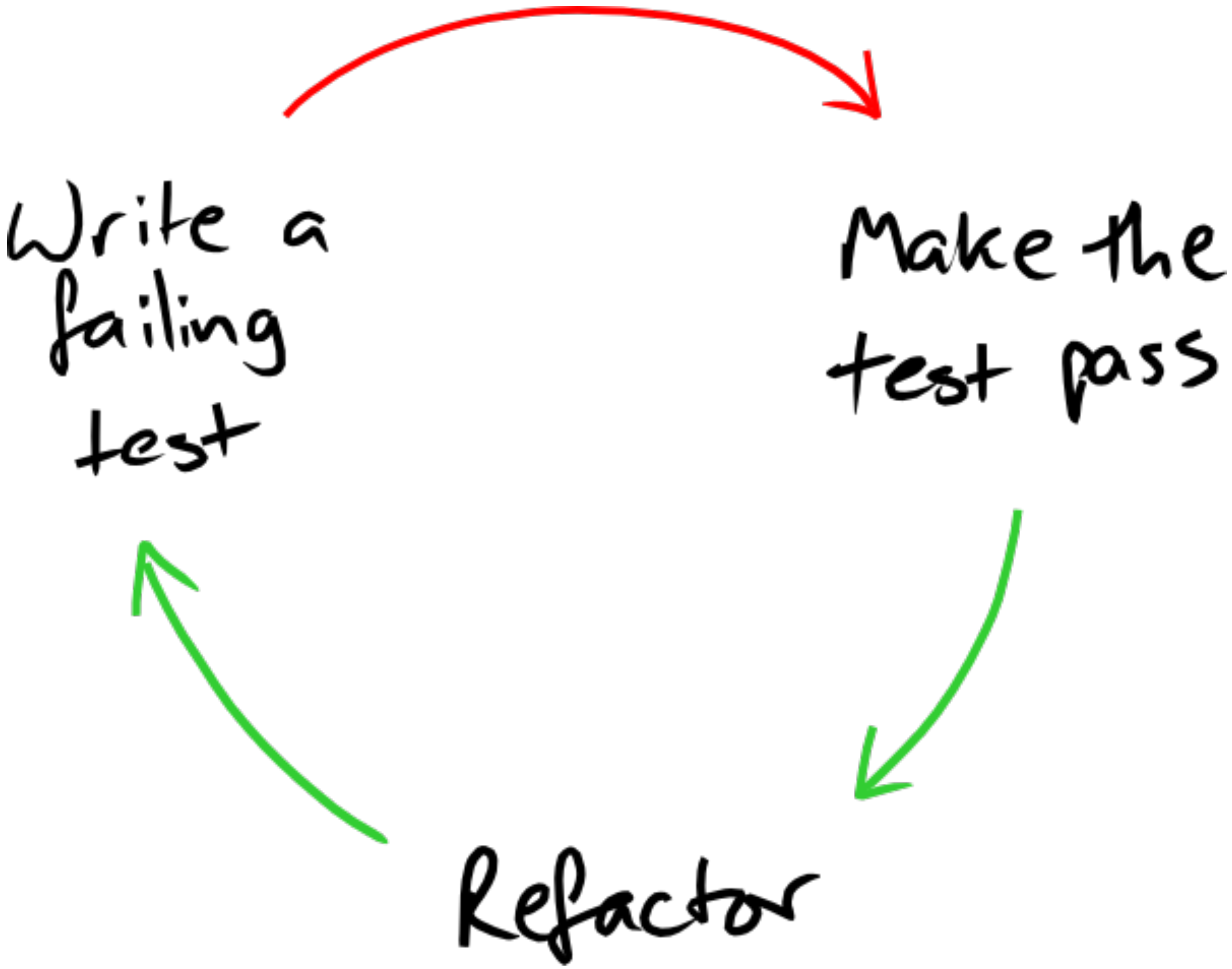
Introduction to TDD and BDD

Seb Rose

@sebrose

http://cucumber.io

- **TDD**
- BDD
- Contradiction
- xDD



From Growing Object-Oriented Software by Nat Pryce and Steve Freeman

http://www.growing-object-oriented-software.com/figures.html

- Write the next **specification**
- Make it **pass** (quickly)
- **Refactor** (until happy)

Refactor

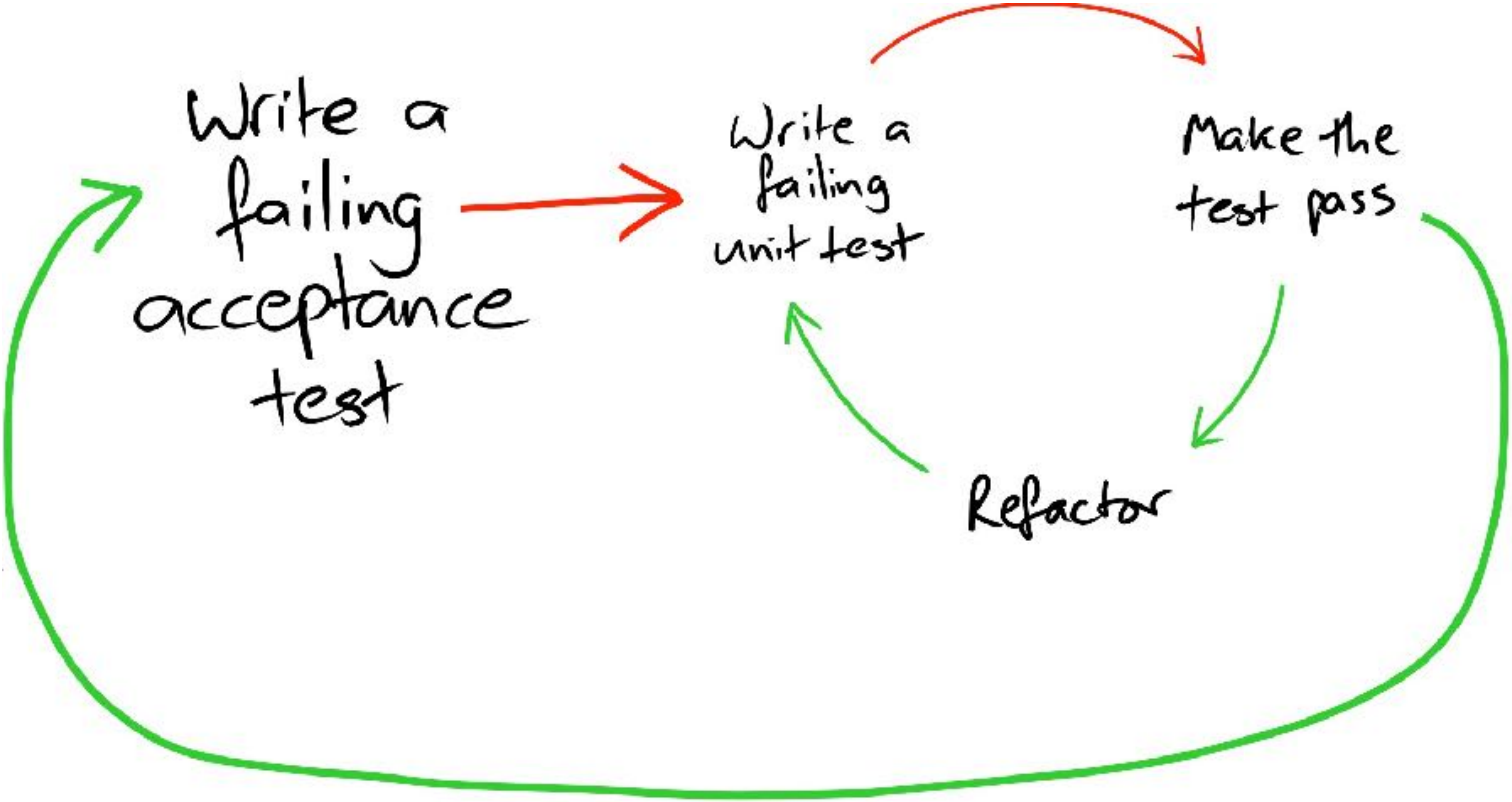
Improve the structure of some code without affecting its externally observable behaviour.

- TDD
- **BDD**
- Contradiction
- xDD

"BDD practitioners **explore, discover, define,** then **drive out** the desired behaviour of software using **conversation, concrete examples,** and **automated tests**."

From Cucumber for Java Book by Seb Rose, Matt Wynne and Aslak Hellesøy
https://pragprog.com/book/srjcuc/the-cucumber-for-java-book

@sebrose http://cucumber.io



From Growing Object-Oriented Software by Nat Pryce and Steve Freeman

http://www.growing-object-oriented-software.com/figures.html

```
Feature: Team Scoring
  Teams start with zero score.
  Correct answer gets points depending on
  how difficult it is.

Scenario: Score starts at 0
  Given I register a team
  Then my score is 0

Scenario: Correct easy answer scores 10
  Given I register a team
  When I submit a correct easy answer
  Then my score is 10

Scenario: Correct hard answer scores 50
  Given I register a team
  When I submit a correct hard answer
  Then my score is 50
```

User Story
Acceptance criteria

- TDD
- BDD
- **Contradiction**
- xDD

BDD

- Behaviour Driven Development

ATDD

- Acceptance Test Driven Development

SbE

- Specification by Example

TDD

- Test Driven Development

What's the difference between TDD, ATDD, BDD and SbE?
They're called different things

http://lizkeogh.com/2011/06/27/atdd-vs-bdd-and-a-potted-history-of-some-related-stuff/

A group of people **specifying** how some software should behave **before** implementing it

- Work from the outside
- Use examples
- Ubiquitous language

- TDD
- BDD
- Contradiction
- **xDD**

First ‘D’ is for Driven

- First specify the behaviour
- Then use the ‘failing’ specification to drive development

Second ‘D’ for Design?

- Listen to your tests
- Testability gets ‘baked in’
- Refactor until it feels right

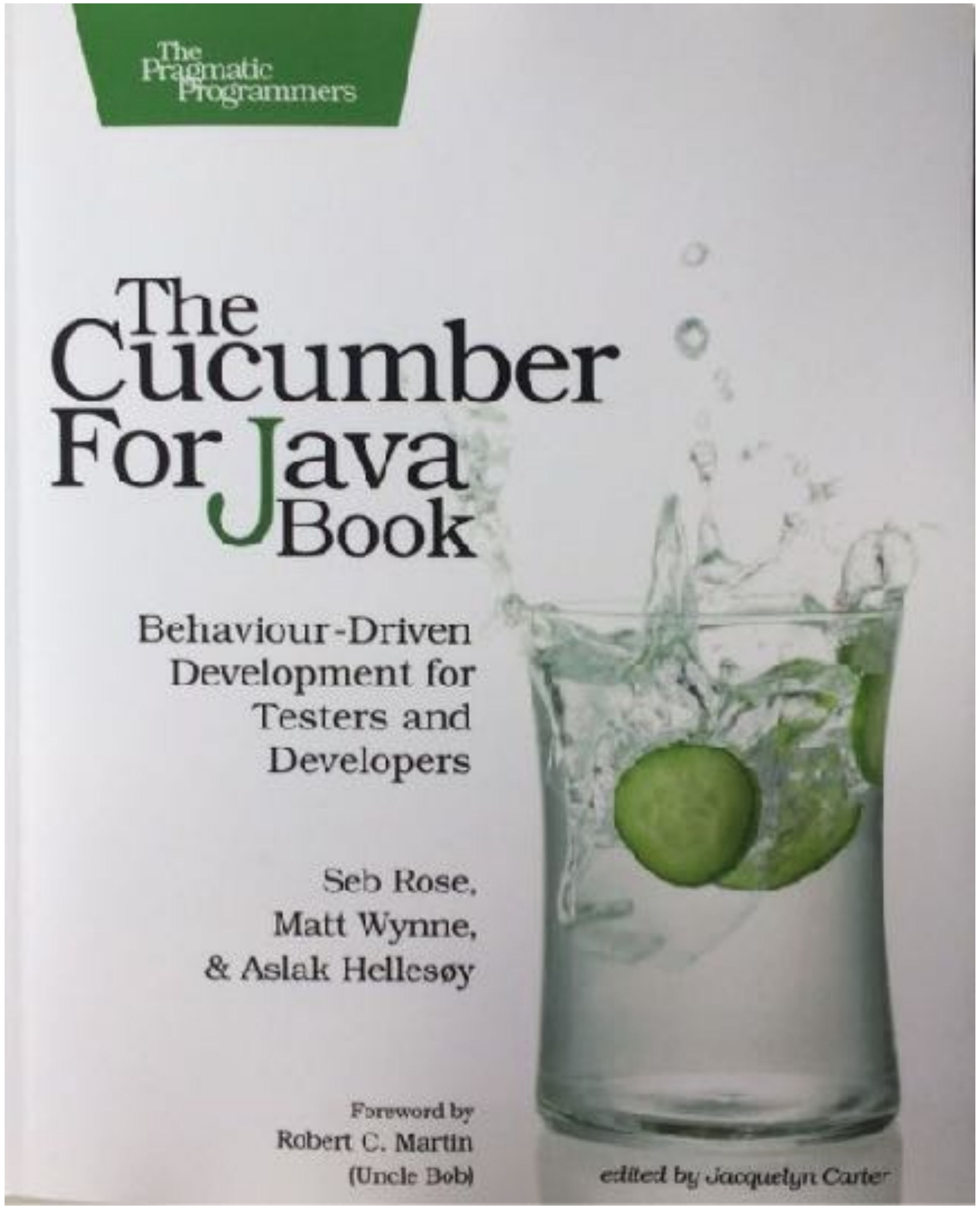
They’re all the same (mostly)

- Outside-in
- Examples
- Ubiquitous language
- Automated verification

The only relevant question:

“**Who**
is
interested
in reading the tests?”

cucumber ltd



The Pragmatic Programmers

The Cucumber For Java Book

Behaviour-Driven Development for Testers and Developers

Seb Rose, Matt Wynne, & Aslak Hellesøy

Foreword by Robert C. Martin (Uncle Bob)

edited by Jacquelyn Carter

Seb Rose

Twitter:	@sebrose
Blog:	http://cucumber.io
E-mail:	seb@cucumber.io