# Test-Driven Development training

## Things you can or should do while waiting on class start:

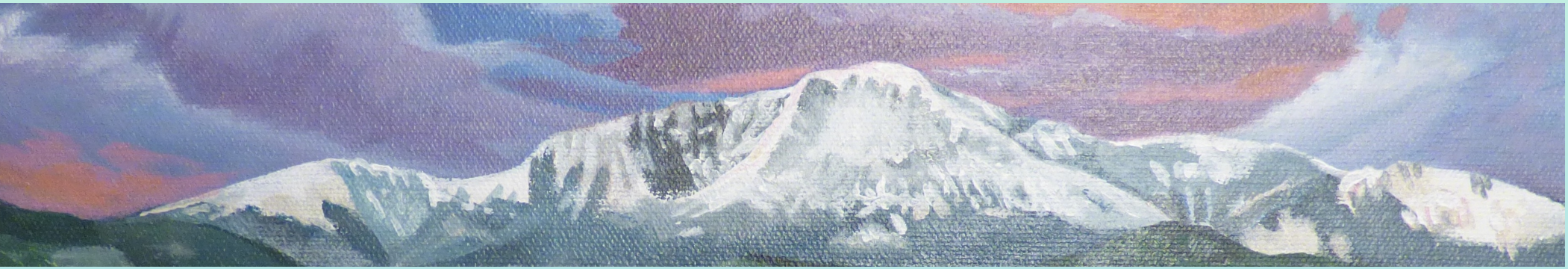
- Install mob tool.
  See https://github.com/remotemobprogramming/mob under "how to install." (Use git bash under Windows.)
- `git clone https://github.com/jlangr/LibraryNet2020`
- Load solution.
- Make sure you can run the tests.
- Either install Kahoot! on your phone, or navigate to kahoot.it.
- Email me your cell # at jeff@langrsoft.com (for disconnect purposes).

Jeff Langr / @jlangr

ICON
AGILITY SERVICES

# Test-Driven Development Foundations v1.3

# TDD Foundations in C#



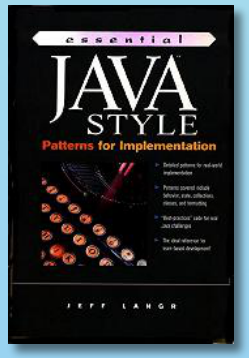

## Jeff Langr

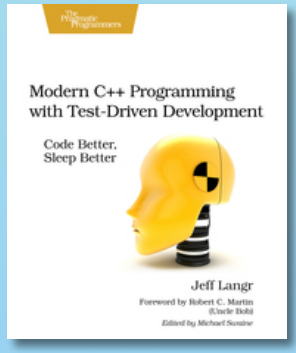
jeff@langrsoft.com

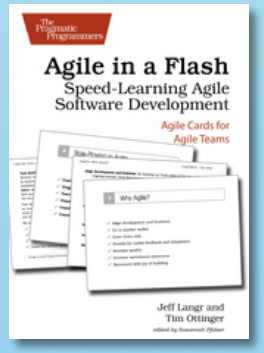
@jlangr

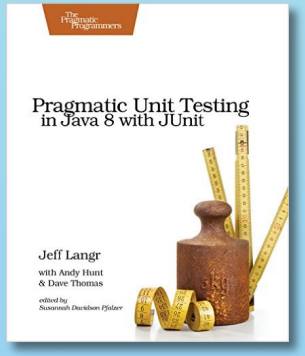

LANGR | SOFTWARE SOLUTIONS

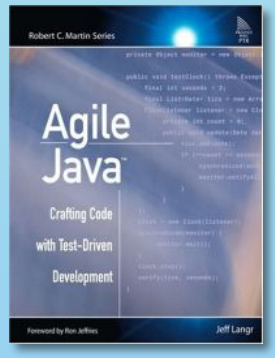
http://langrsoft.com

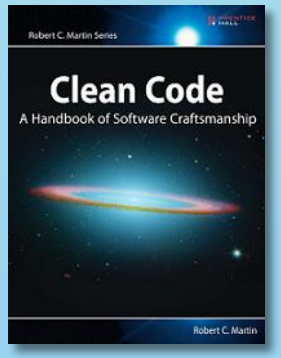


Kahoot!

# What's a unit test look like?

```
public class TestClassName
{
    [XUnit.Fact]
    public void DescriptionForTest()
    {
        // Arrange;

        // Act;

        // Assert;
    }
}
```

The AAA mnemonic is courtesy of Bill Wake.

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# What do assertions look like?

```
Assert.False(condition);

Assert.Equal("something", text);

Assert.NotEqual("something else", text);

Assert.Contains("tech", "technology"); // also DoesNotContain

Assert.Matches(".*thing$", "something");

Assert.Throws<DivideByZeroException>(() ⇒ 4 / zero);

Assert.Empty(someEnumerable); // also NotEmpty

Assert.IsType<Auto>(obj);

Assert.Collection(new List<int> {2, 4},

    n ⇒ Assert.Equal(2, n),

    n ⇒ Assert.Equal(4, n));

Assert.All(new List<string> {"a", "ab", “abc"}, s ⇒ s.StartsWith("a"));
```

**Where’s the docs?**

- Use your editor’s intellisense.
- Browse XUnit source.
- Consider https://fluentassertions.com

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# A sample unit test

```
[Fact]
public void IdlesEngineWhenStarted()
{
    var auto = new Auto();
    auto.DepressBrake();

    auto.PressStartButton();

    Assert.InRange(auto.RPM, 950, 1100);
}
```

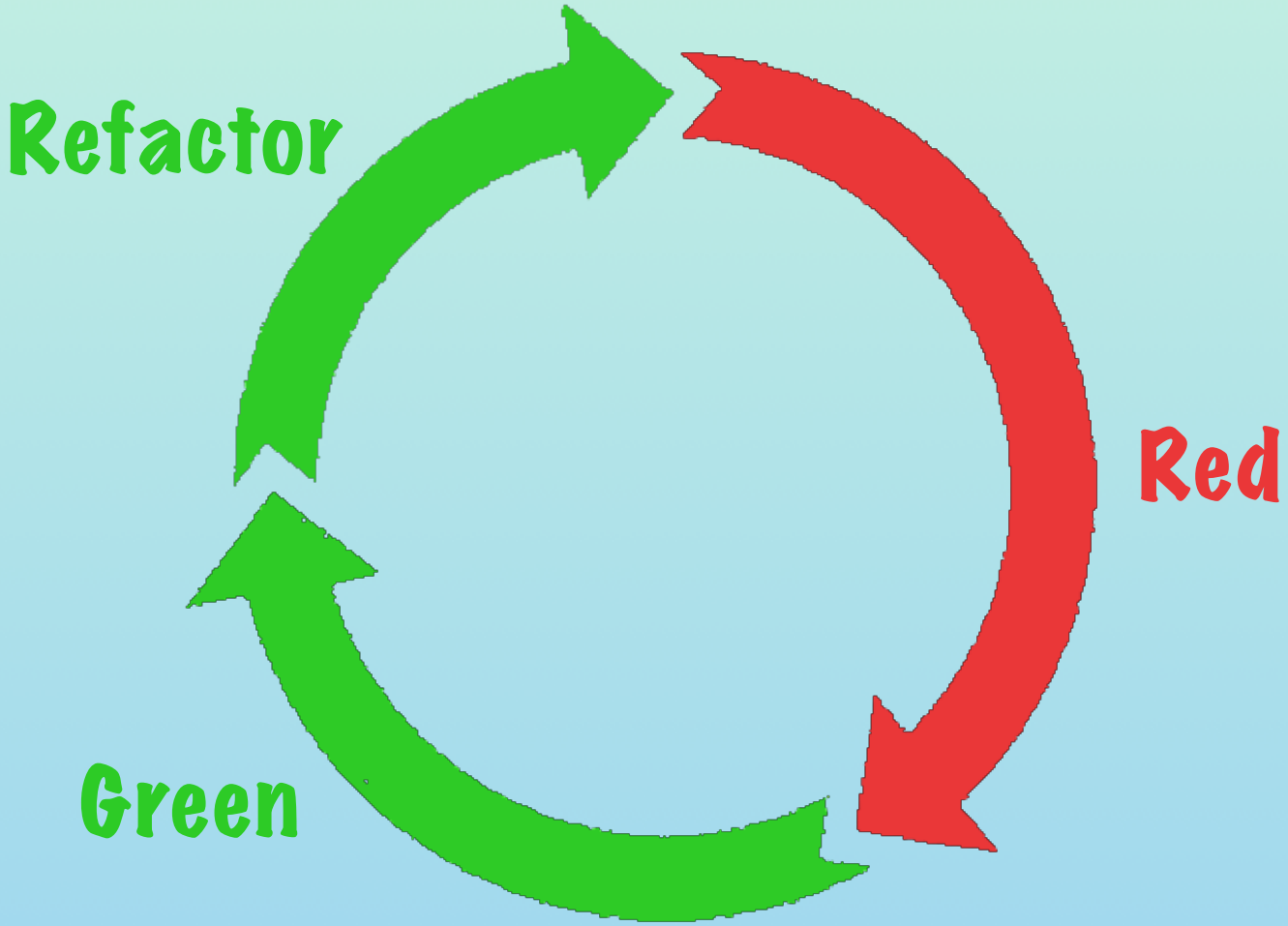

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Exercise: Simple outcomes

Flesh out the tests in ./LibraryTests/Util/BasicsTest.cs

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

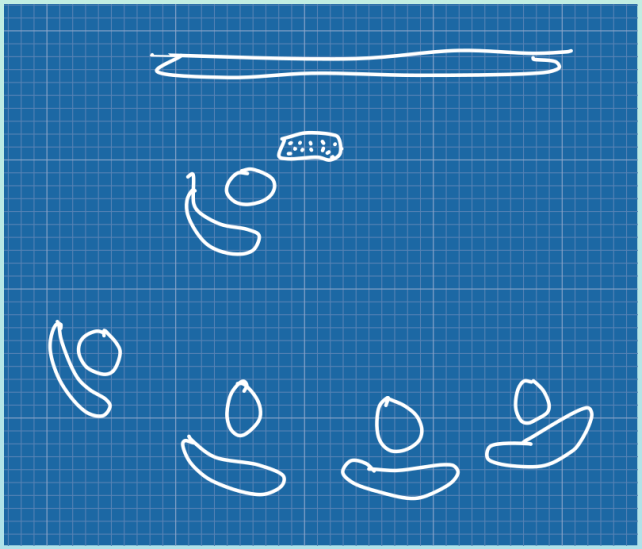
# Test-Driven Development




*An incremental design technique*

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# A Tale of Two Mob Programming Rules



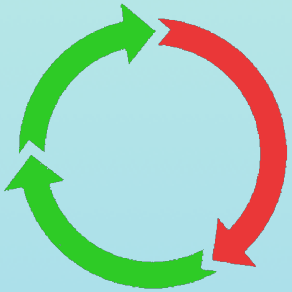
*Once upon a time, there existed a quaint notion of an "office," a physical place where people could undistance and work together…*

- Strong-style navigation by mob
- Short, timed rotations

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

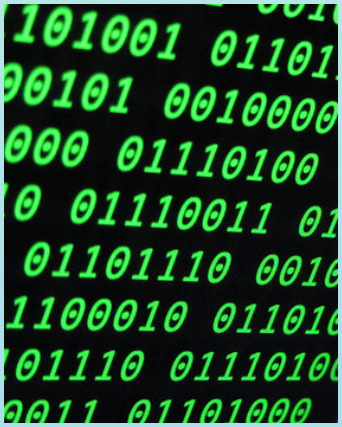
# Exercise: (Test)-Code-Refactor

See LibraryTest/Util/**NameNormalizerTest.cs**



Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Why???



Keep yer code clean

Document choices

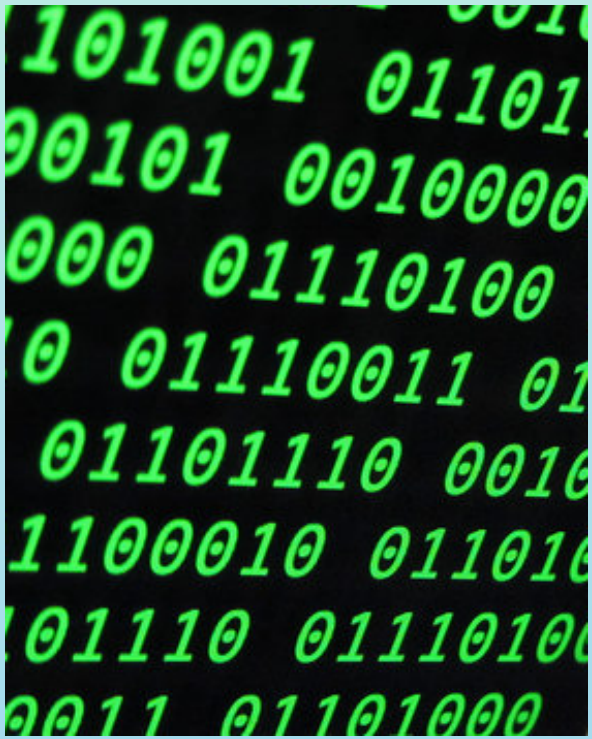
Gate defects

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.



"BinaryCode," courtesy https://www.flickr.com/photos/132889348@N07/
License: https://creativecommons.org/licenses/by-sa/2.0/

# Core TDD themes



- Drive *behavior*
- Specify by example
- Grow in small increments
- Stick to the cycle
  - Always see red
  - Always refactor

***It's just code!***

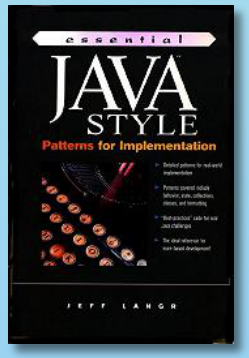

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

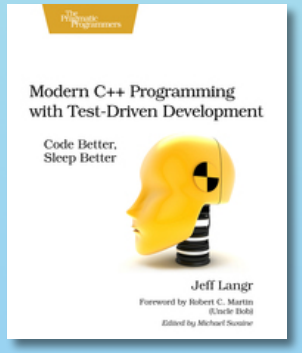
"BinaryCode," courtesy https://www.flickr.com/photos/132889348@N07/
License: https://creativecommons.org/licenses/by-sa/2.0/

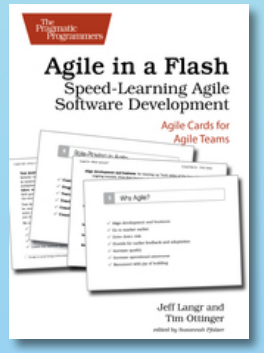
Kahoot!

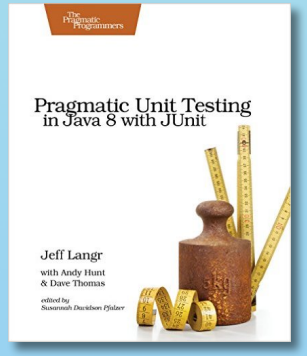
# Stepping back: Writing tests

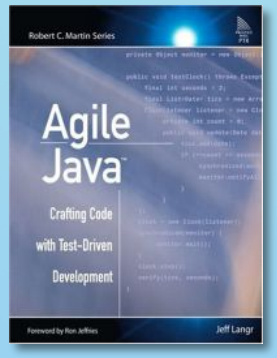
LANGR | SOFTWARE SOLUTIONS

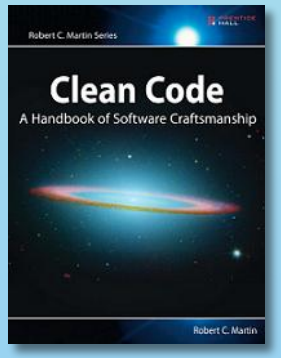


# Recent file list

Create a list of recently-opened files (max: 5) for use in a GUI application.

What tests do we need?

```
public class RecentFileList {
   public void Open(string filename) { … }
   public IList<string> Items { get; private set; }
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Ordering tests

- **Z**ero
- **O**ne
- **M**any
- **B**oundaries
- **I**nterface definition
- **E**xceptional behavior
- **S**imple Solutions & Scenarios

**ZOMBIES!**

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

http://blog.wingman-sw.com/archives/677

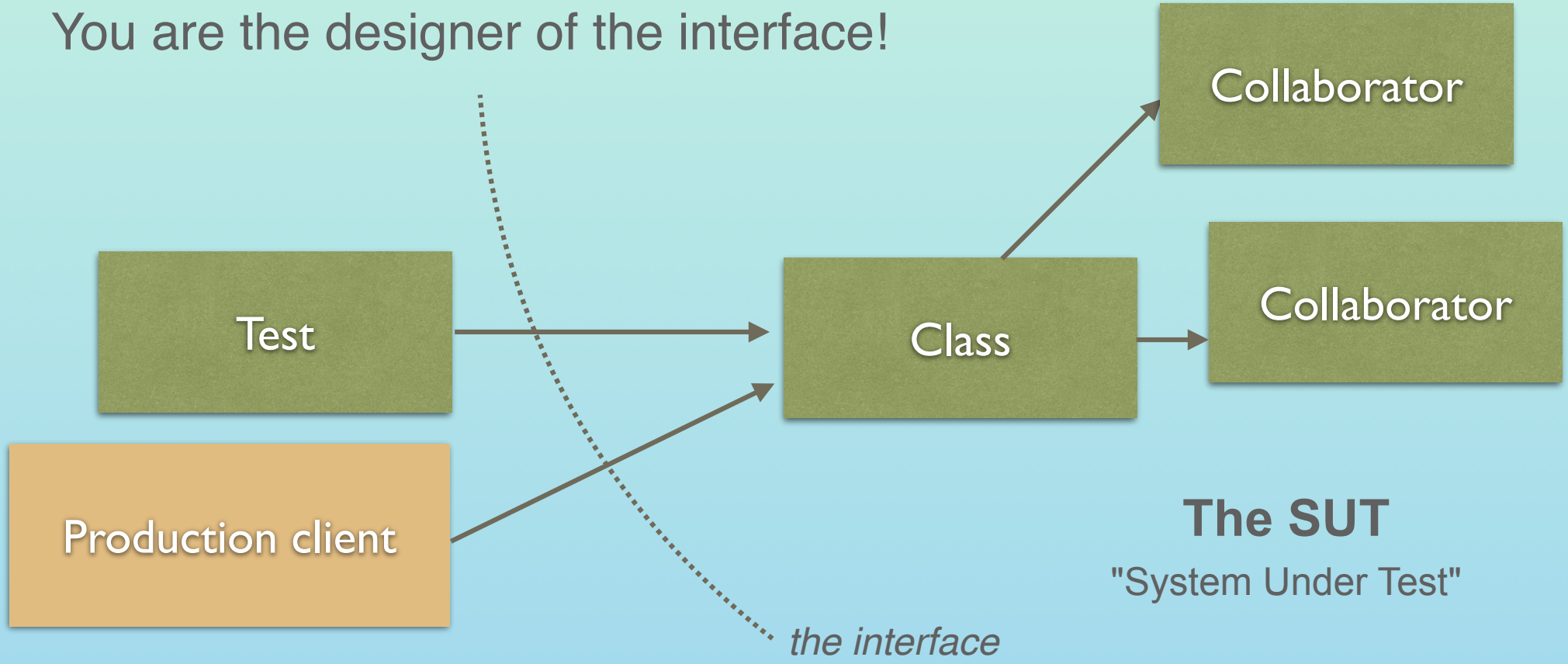
# Test lists

- **Z**: No files opened
- **O**: One file opened
- **M**: Multiple files opened
- **B**: One more than capacity opened
- **E**: File opened twice

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Your test is a client

You are the designer of the interface!




**The SUT**

"System Under Test"

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Structuring the test

*"sifts duplicate filename to top of recently-used list"*

| | |
|---|---|
| **Arrange** | Add file with name `opened-A.txt`<br>Add file with name `opened-B.txt`<br>Add file with name `opened-C.txt` |
| **Act** | Add file with name `opened-A.txt` |
| **Assert** | Expect recently-used list to equal:<br>`opened-A.txt`<br>`opened-C.txt`<br>`opened-B.txt` |

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# AAA is a visual mnemonic

Don't do this:

```
[Fact]
public void AAAIsAVisualMnemonic()
{
    // arrange
    var x = new Something();
    x.MoreSetup();
    x.EvenMoreSetup();

    // act
    x.DoStuff();

    // assert
    Assert.True(x.SomeProperty);
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Let space do the talking

```
[Fact]
public void AAAIsAVisualMnemonic()
{
    var x = new Something();
    x.MoreSetup();
    x.EvenMoreSetup();

    x.DoStuff();

    Assert.True(x.SomeProperty);
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Assert-first

Consider working from the outcome.

*"sifts duplicate filename to top of recently-used list"*

```
Assert.Equal(new List[]{
  duplicateOfFirstAdded, "3rd", "2nd" },
  recentlyUsedList.Items);
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Assert-first

Consider working from the outcome.

*"sifts duplicate filename to top of recently-used list"*

```
recentlyUsedList.Open(duplicateOfFirstAdded);

Assert.Equal(new List[]{
  duplicateOfFirstAdded, "3rd", "2nd" },
  recentlyUsedList.Items);
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Assert-first

Consider working from the outcome.

*"sifts duplicate filename to top of recently-used list"*

```
var recently UsedList = new RecentlyUsedList();
recentlyUsedList.Add("1st");
recentlyUsedList.Add("2nd");
recentlyUsedList.Add("3rd");
var duplicateOfFirstFileAdded = "1st";

recentlyUsedList.Open(duplicateOfFirstFileAdded);

Assert.Equal(new List[]{
  duplicateOfFirstAdded, "3rd", "2nd" },
  recentlyUsedList.Items)
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Exercise: Stock portfolio

- is it empty or not?
- what is the count of unique symbols?
- given a symbol and # of shares, make a purchase
- how many shares exist for a given symbol?
- given a symbol and # of shares, sell the shares
- throw an exception when selling too many shares

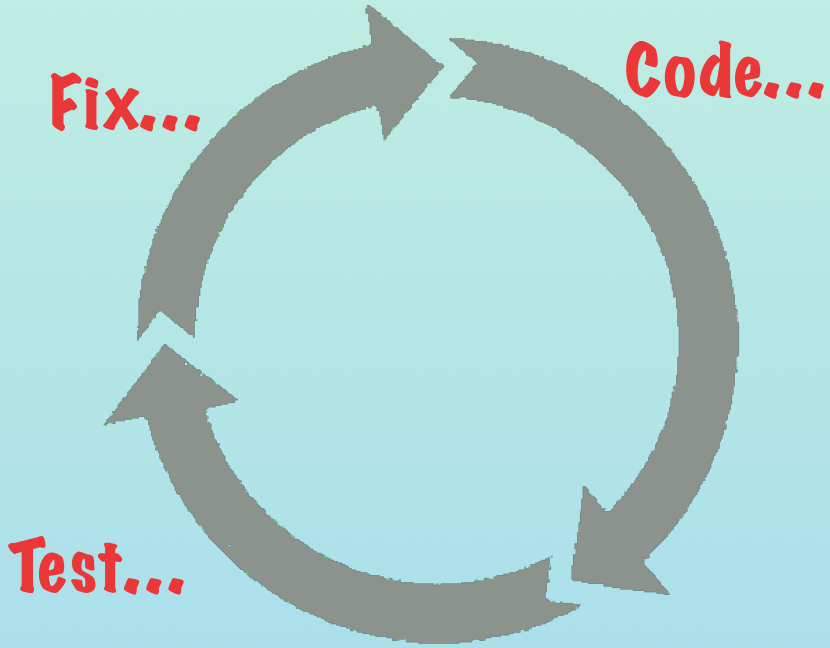
"Tokyo Stock Exchange," courtesy https://www.flickr.com/photos/31029865@N06/
License: https://creativecommons.org/licenses/by/2.0/

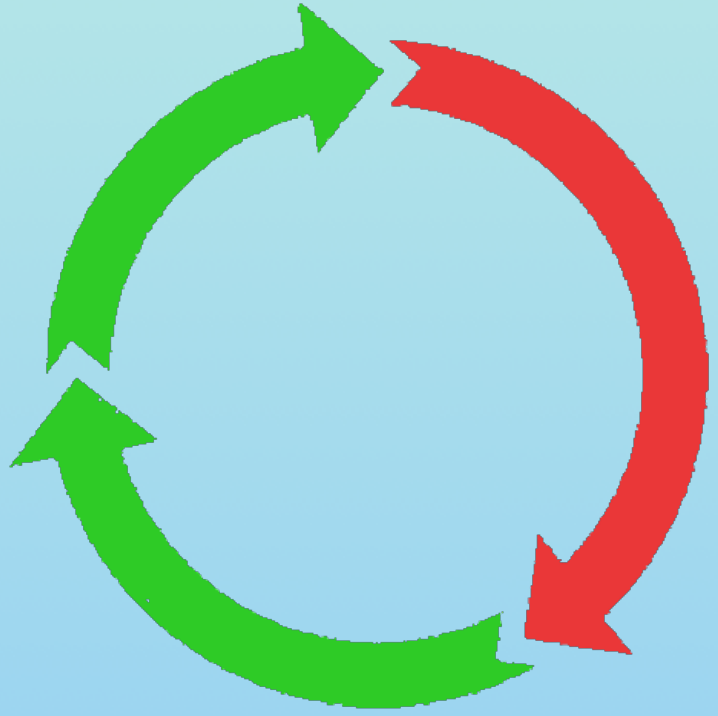
***** Remember! *****

**Red:** Ensure test fails
**Green:** Implement no more code than necessary
**Refactor:** Clarify and eliminate all duplication

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# TAD** and TDD: What's the difference?

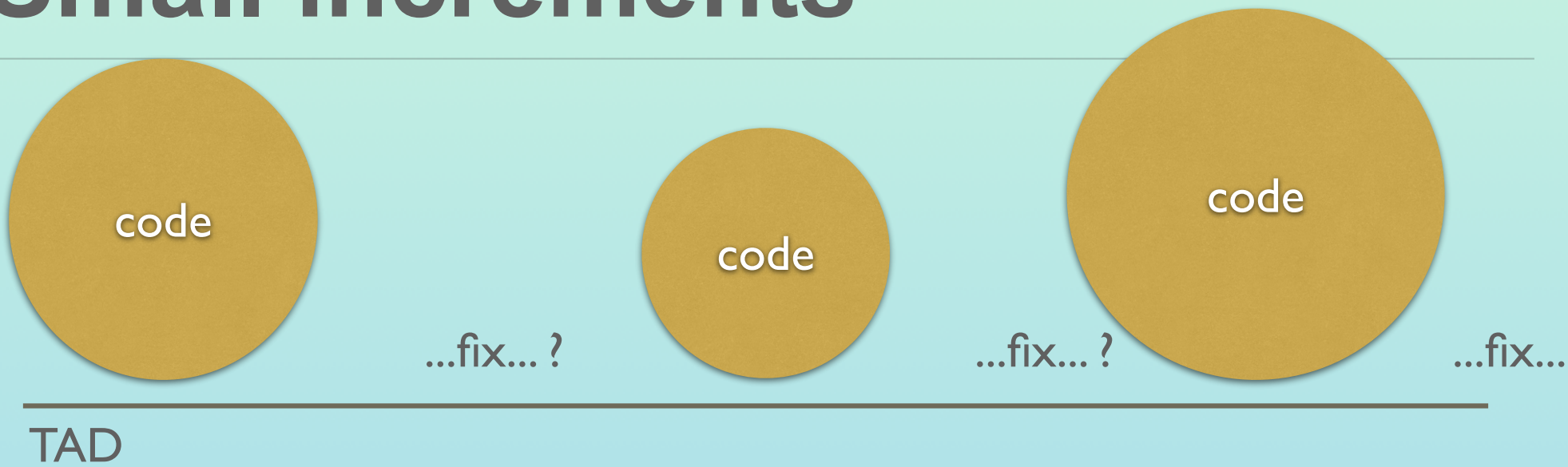


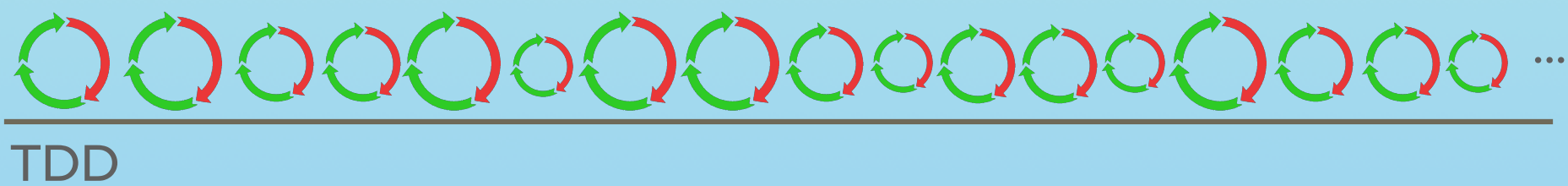
"Test-After Development"

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Small increments



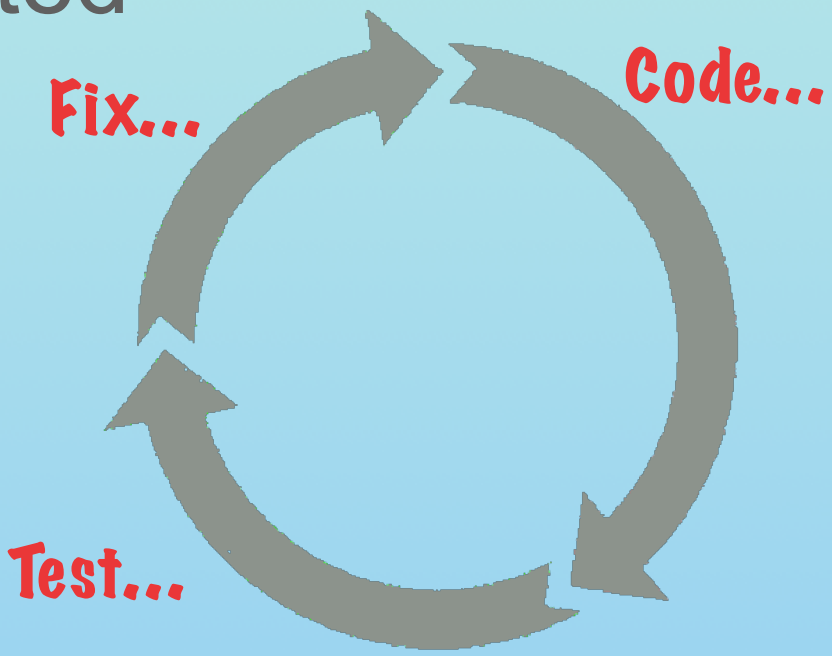

*vs.*




Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Test-After Development (TAD)

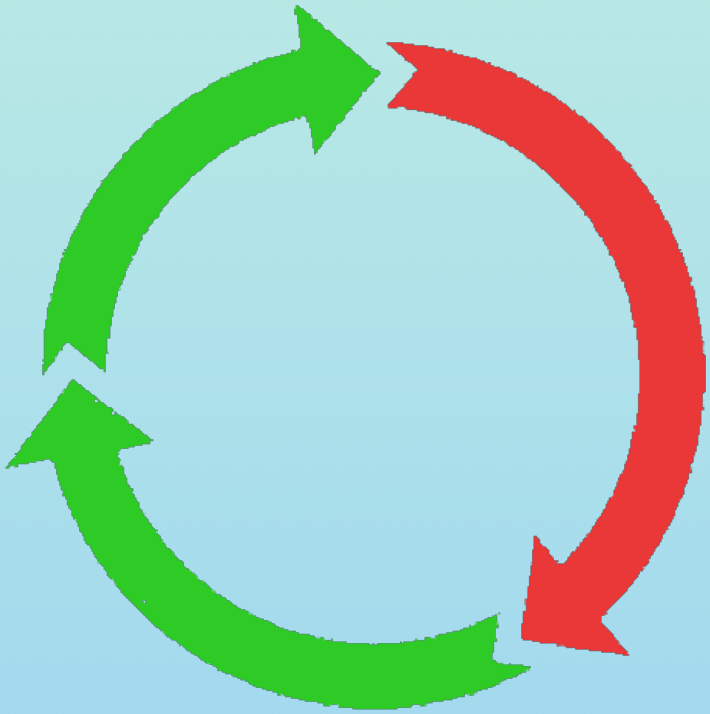
- Some refactoring accommodated
- Coverage: ~70%
- Design not usually impacted
- Some defect reduction
- Separate task?




Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Test-Driven Development (TDD)

- Continual refactoring
- Coverage for all intended features
- Incremental design shaping
- Significant defect reduction
- Minimized debugging
- Integral part of coding process
- Clarify / document needs / choices
- Continual forward progress
- Consistent pacing
- Continual feedback / learning
- Sustainable



Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

Kahoot!

# Katas

http://codekata.com

**Other sites:**

https://exercism.io

http://cyber-dojo.org

https://sites.google.com/site/tddproblems/

https://www.codewars.com

https://github.com/jlangr/name-normalizer

https://github.com/emilybache

https://projecteuler.net/archives


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"Kata," courtesy https://www.flickr.com/photos/superwebdeveloper/
License: https://creativecommons.org/licenses/by/2.0/

# Roman numeral converter

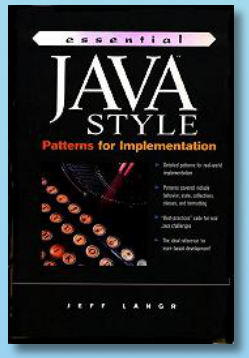
Given a positive integer
from 1 up to 4000,
answer its Roman equivalent

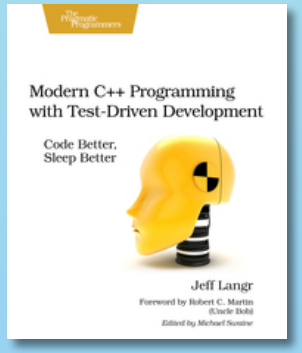
Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

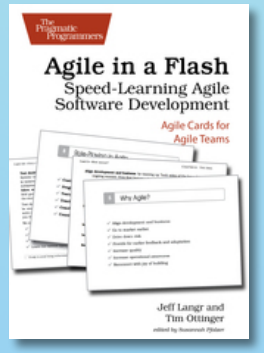
"Kata," courtesy https://www.flickr.com/photos/superwebdeveloper/
License: https://creativecommons.org/licenses/by/2.0/

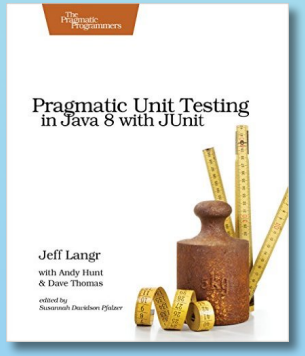
# Where does TDD fit?

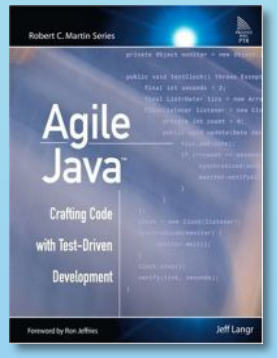
LANGR | SOFTWARE SOLUTIONS

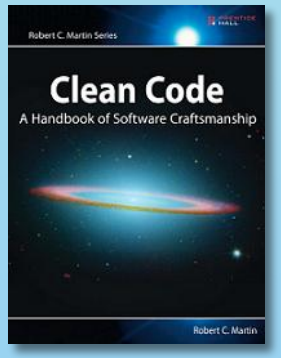


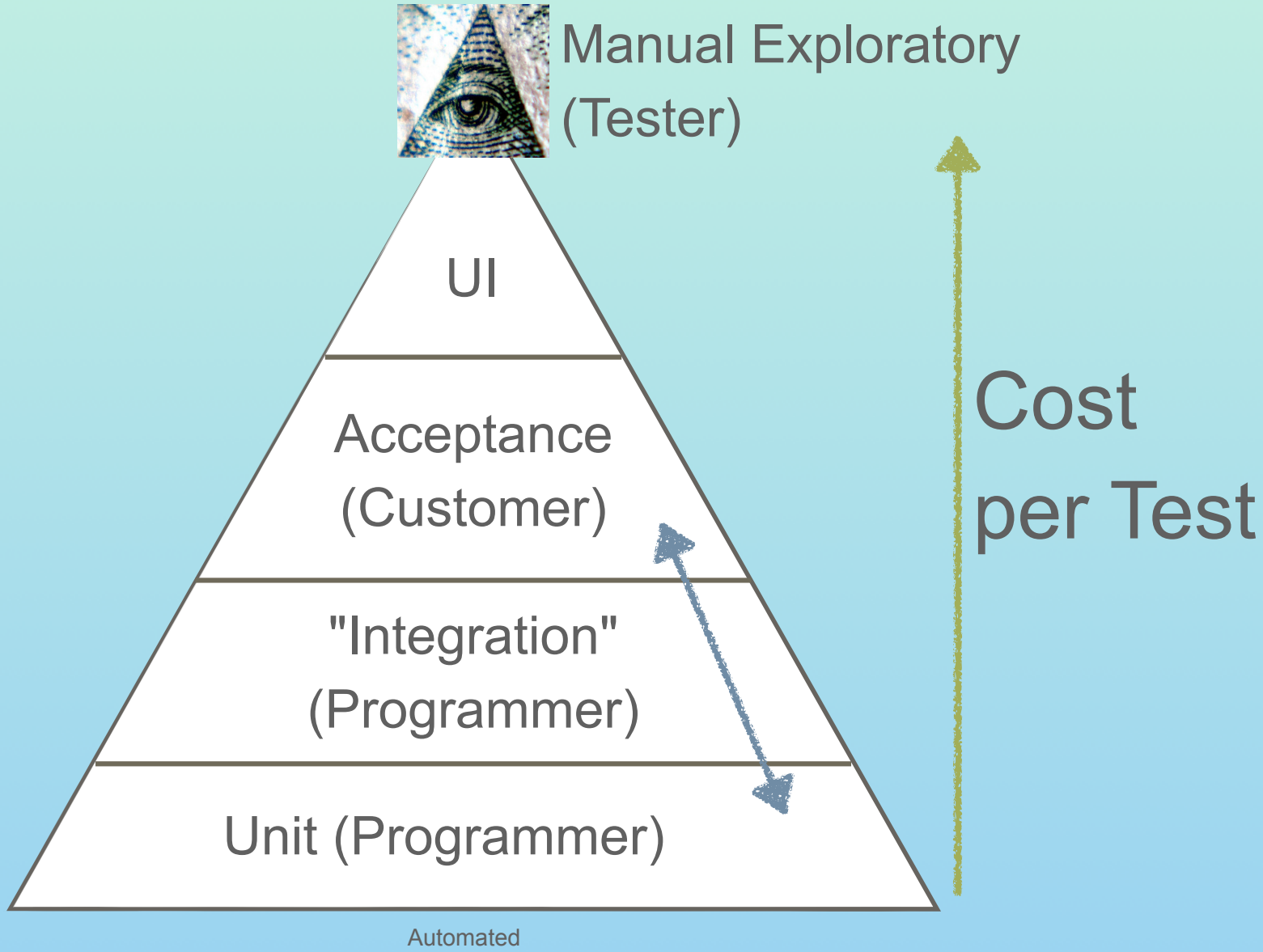



# Unit testing is but a slice of the whole



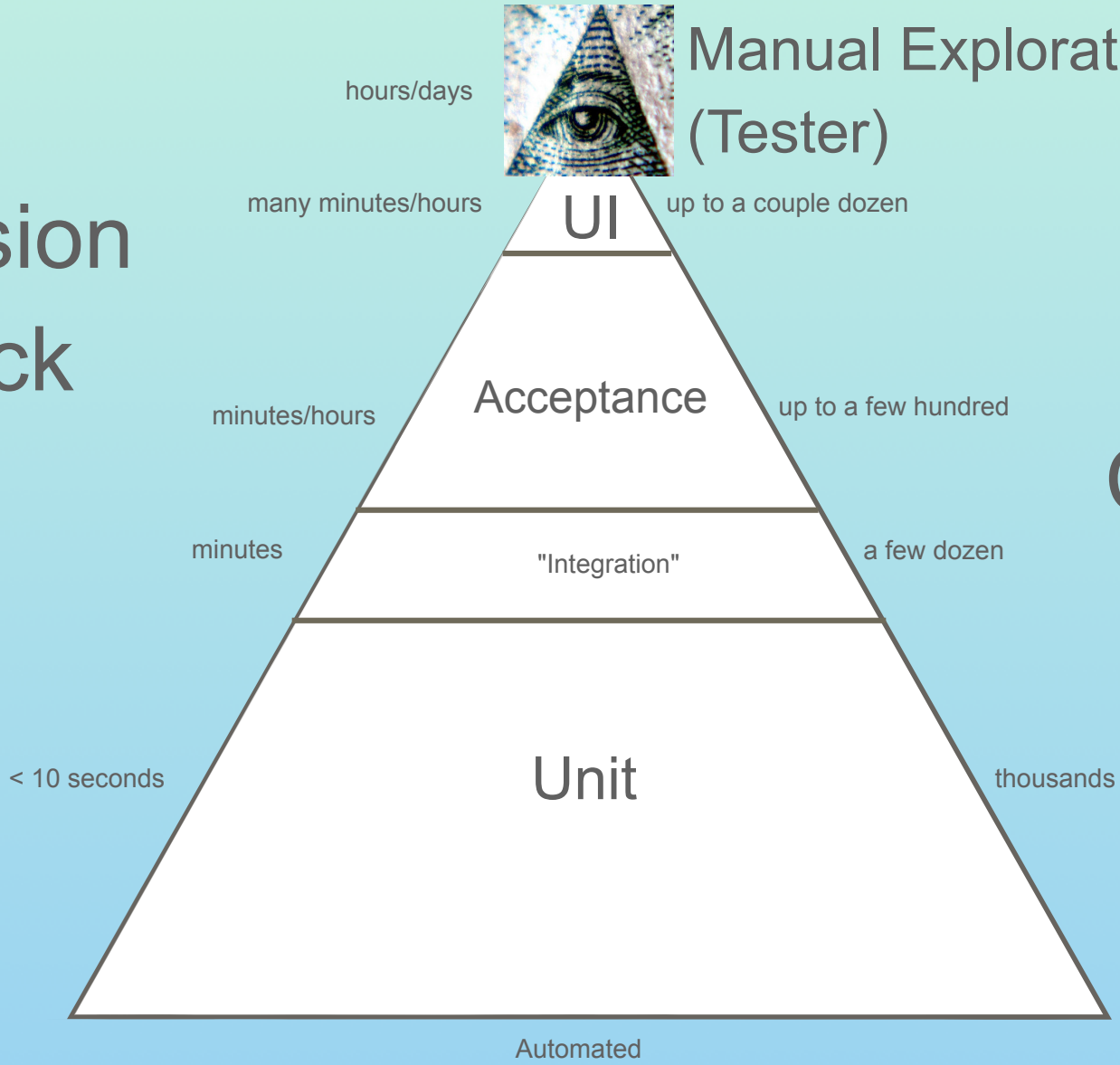


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

Eye image: Frank Boston, https://www.flickr.com/photos/fixersphotos/4019797096/
License: https://creativecommons.org/licenses/by/2.0/

# Right-sizing the pyramid

Regression Feedback




Count


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

Eye image: Frank Boston, https://www.flickr.com/photos/fixersphotos/4019797096/
License: https://creativecommons.org/licenses/by/2.0/

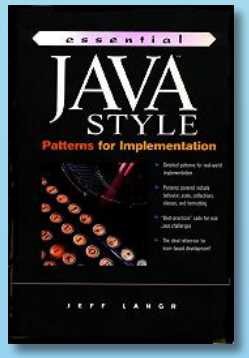
# Feedback questions

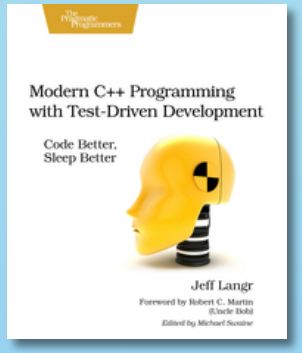
| | |
|---|---|
| :-) | What have I learned of value? |
| %-I | What, if anything, am I unclear about regarding the discussion / exercises? |
| ?:-/ | What, if anything, did I find disturbing / disconcerting? |

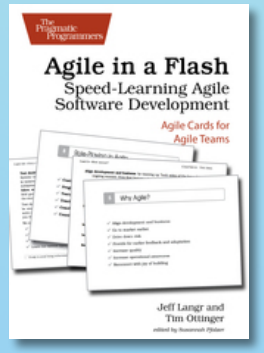
Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

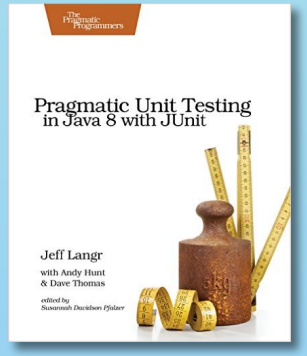
# Documenting unit behaviors

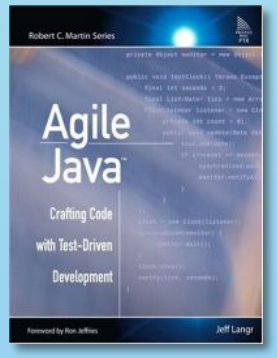
LANGR | SOFTWARE SOLUTIONS

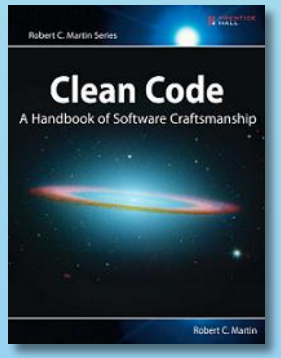


# Tests as documents

```
public class AutoTest
{
    [Fact]
    public void IdlesEngineWhenStarted()
    {
        var auto = new Auto();
        auto.DepressBrake();

        auto.PressStartButton();

        Assert.InRange(
            auto.RPM, 950, 1100);
    }
}
```

what we're describing

the generalized behavior it supports

an example of that behavior

***Seek to first understand through the tests.***

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Behavior-driven naming

- Various forms:
  - *(Given)-When-Then:*
    WhenConditionThenSomethingHoldsTrue
  - *Result-Condition:*
    SomethingHoldsTrueWhenCondition
  - *Verb-Condition:*
    DoesSomethingWhenCondition

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Iterative naming

- Re-consider name continually

[Fact] public void **Something**() {}

[Fact] public void **EngineStart**() {}

[Fact] public void **EngineStartIdleSpeed**() {}

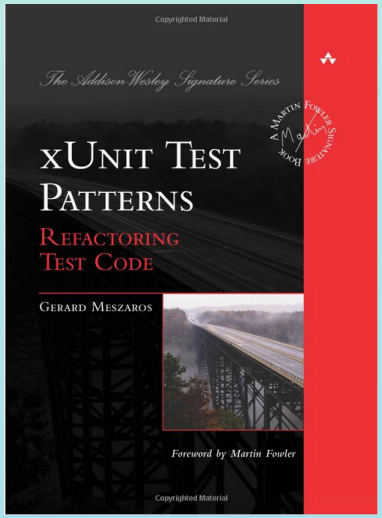
[Fact] public void **HasLowIdleOnEngineStart**() {}

- Review often and holistically!

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Sustainable tests

*The long answer:*




*A quicker route:*

- Single behavior tests
- AAA
- Correlate results with context
- Test abstraction

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Exercise: Test smells

Find and fix test smells.
Paraphrase cleaned tests
to your pair / mob.

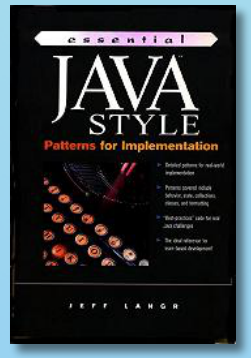
Further instructions: LibraryTests/LibraryTest/Models/**HoldingTest.cs**

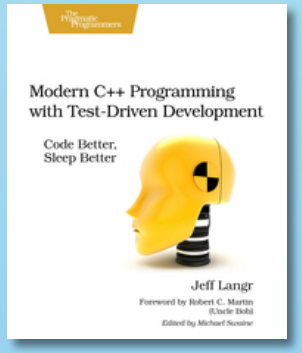

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

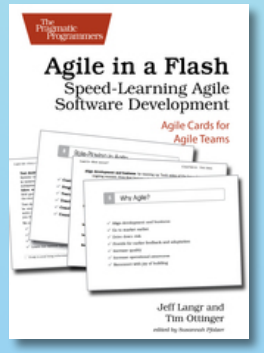
"skunk_dog," courtesy https://www.flickr.com/photos/istolethetv/
License: https://creativecommons.org/licenses/by/2.0/

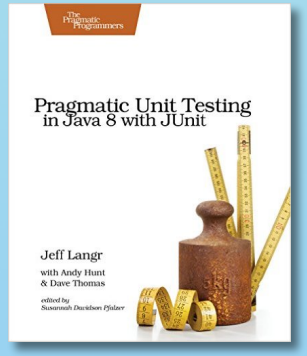
Kahoot!

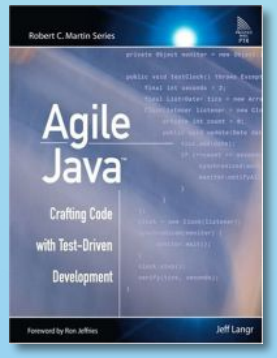
# Continual design

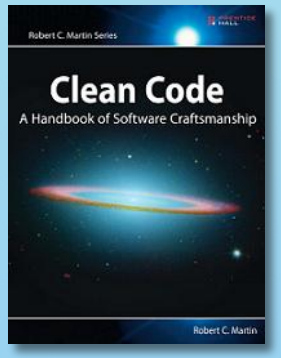


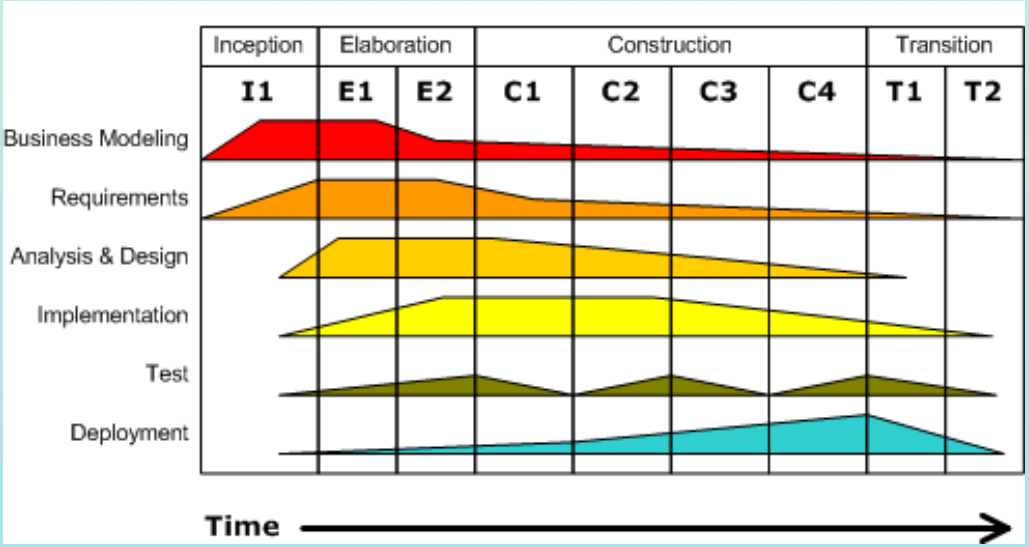



# Doing software development

- Activities:
  - Analysis, design, coding, testing, review, documentation, planning, deployment, ...

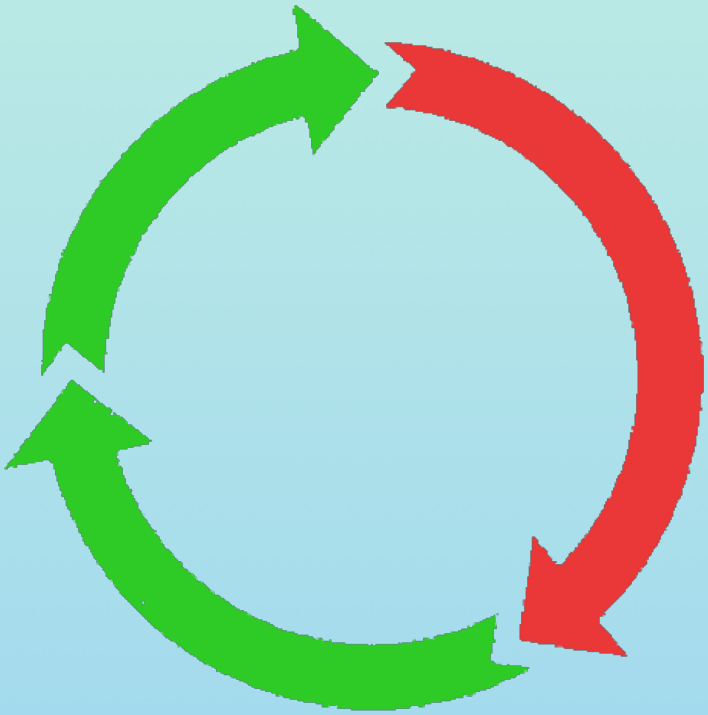



- Agile:
  - *All activities all the time*

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# TDD is continual ...

- Testing
- Coding
- Documentation
- Review
- Design

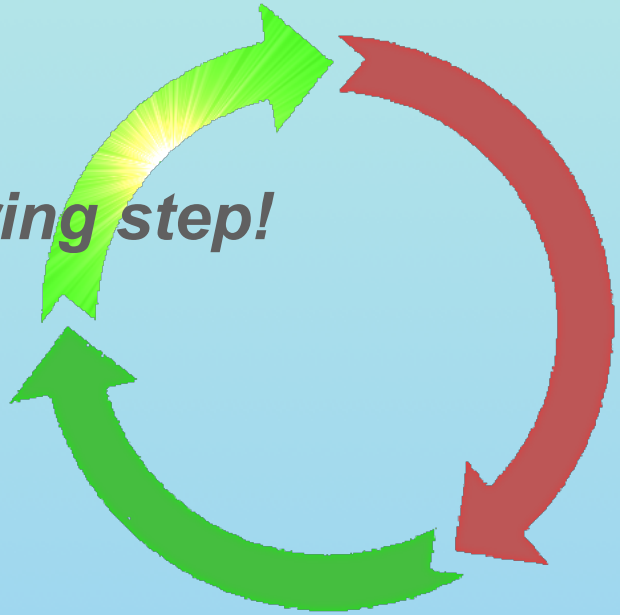


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Continual design

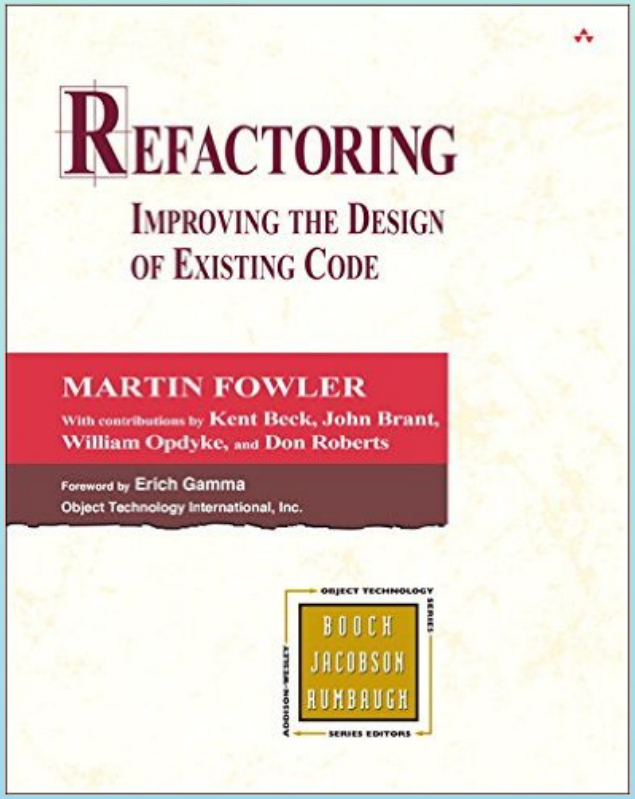
- Always retain an optimal design
- Entropy is unavoidable

***Never skimp on the refactoring step!***

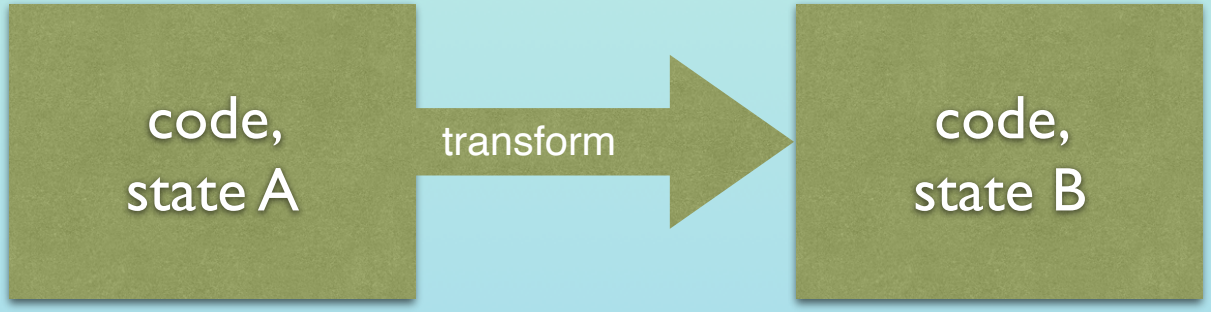


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Refactoring



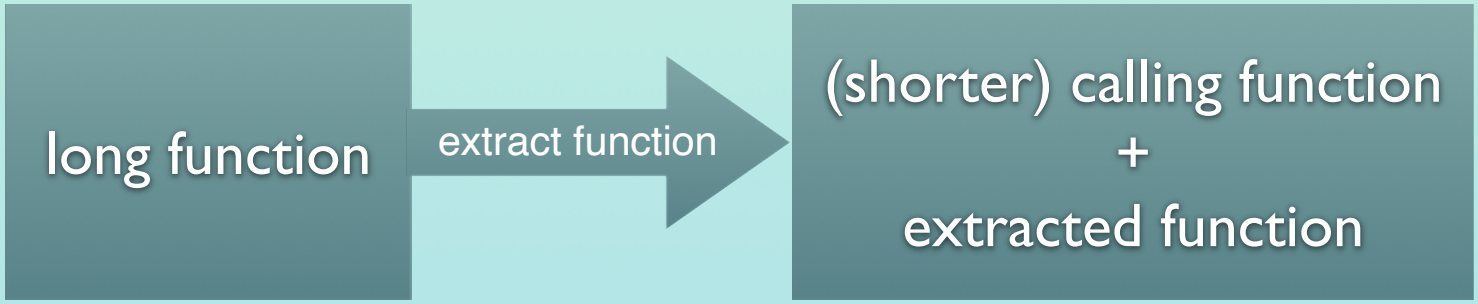

*1999 / 2019*




Same "externally recognized" behavior


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# For *(almost)* every transform...



Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Code smells

An aroma?

- Comments
- Duplicate Code
- Feature Envy
- Large Module
- Long Function
- Long Parameter List
- Primitive Obsession
- Shotgun Surgery
- Speculative Generality
- Switch Statements
- **. . . and many more**

…or a stink?

*"A complete code smells reference:"*
https://github.com/lee-dohm/code-smells

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Exercise: Code smells

Identify code smells.

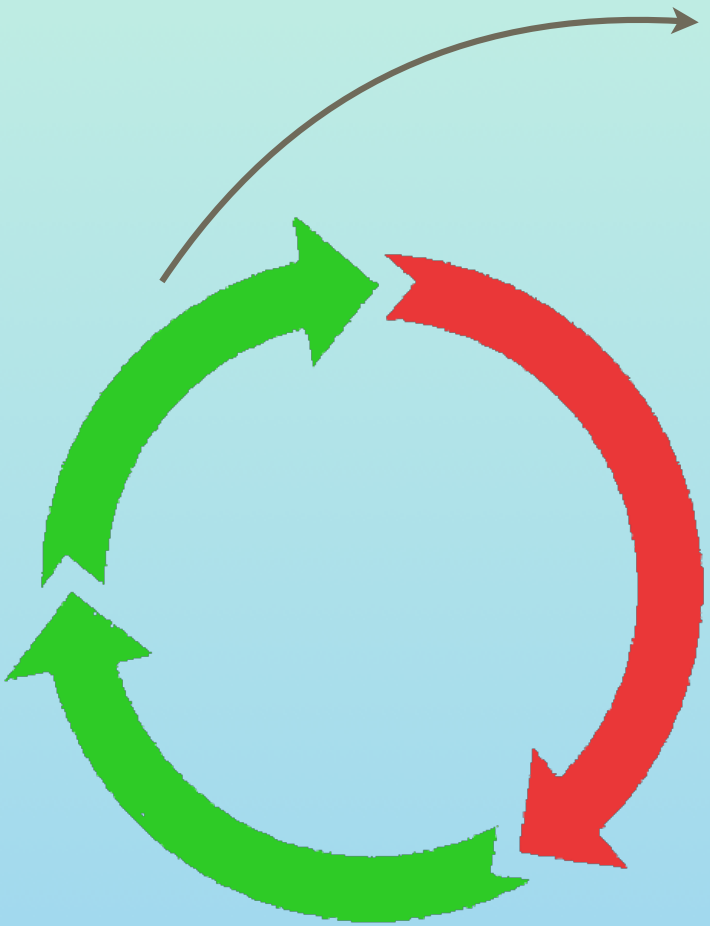
See ./src/pos/routes/checkout.js

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.



"skunk_dog," courtesy https://www.flickr.com/photos/istolethetv/
License: https://creativecommons.org/licenses/by/2.0/

# Refactoring opportunities



small cleanup
verify
small cleanup
verify
...

**One thing at a time!**

Break things once in a while.

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Renaming

"Lake Louise Train Station Restaurant - Bread and Butter," courtesy https://www.flickr.com/photos/elsiehui/
License: https://creativecommons.org/licenses/by/2.0/


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Refactoring: Extract method

```
public void UglyMethod()
{
   // stuff a
   ...
   // stuff b
   ...
   // some smaller behavior
   X x = new X();
   x.DoStuff();
   ...
   // stuff c
   ...
}
```

```
public void UglyMethod()
{
   // stuffs a & b
   ...
  DoSomeSmallerBehavior();
   // stuff c
   ...
}

public void DoSomeSmallerBehavior()
{
   X x = new X();
   x.DoStuff();
   ...
}
```

**Why?**

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Single-line extract?

```
catch(DiagException e)
{
    var errMsg = $"{DateTime.Now}: {System}-{Module} " +
                 $"{LogLevel.Error} {e.Message.Truncate(80)}";
    Log(errMsg);
    throw new ApplicationException(errMsg, e);
}
```

```
catch(DiagException e) {
    var errMsg = FormattedError(e, Module, System);
    Log(errMsg);
    throw new ApplicationException(errMsg, e);
}

private string FormattedError(DiagException e, string module, string system)
{
    return $"{DateTime.Now}: {system}-{module} " +
           $"{LogLevel.Error} {e.Message.Trunc(80)}";
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Exercise: Extract method

In Library/Scanner/ScanStation.cs, focus on extracts, renames, and eliminating lies in `AcceptBarcode`

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Code smell: Feature envy

```
public class WrongPlace
{
   public void Client()
   {
      var x = EnviousMethod();
      ...
   }

   private int EnviousMethod()
   {
      var there = new GreenerGrass();
      var result = there.DoStuff(this.SomeValue);
      there.DoMoreStuff(result);
      return there.Answer();
   }
}
```

*Why is this a problem?*

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Refactoring: move method

```
public class WrongPlace {
   public void Client() {
      var x = new GreenerGrass().HomebodyMethod(this.SomeValue);
      ...
   }
}

public class GreenerGrass {
   public int HomebodyMethod(int someValue) {
      var result = DoStuff(someValue);
      DoMoreStuff(result);
      return Answer();
   }
   ...
}
```

*What must you also do when you move a method?*

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Exercise: Move method

- Create new homes for ScanStation code as appropriate.
- Write tests to document the moved behavior.

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

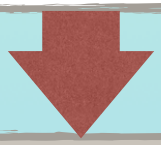
# Refactoring-inhibiting temp

```
public void CompleteSale() {
    total = 0.0m;
    var totalOfDiscountedItems = 0.0m;
    foreach (Purchase item in purchases) {
        var message = "item:" + item.Description;
        if (item.IsDiscountable) {
            var discounted = item.Amount * (1.0m - MemberDiscount);
            totalOfDiscountedItems += discounted;
            message = "item: " + item.Description + " price: " + item.Amount +
                      " discounted price: " + discounted;
            total += discounted;
        }
        else {
            total += item.Amount;
            message = "item: " + item.Description + " price: " + item.Amount;
        }
        DisplayDevice.AppendMessage(message);
    }
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Refactoring: Replace temp with query

```
var discounted = item.Amount * (1.0m - MemberDiscount);
totalOfDiscountedItems += discounted;
message = "item: " + item.Description + " price: " + item.Amount +
          " discounted price: " + discounted;
total += discounted;
```



```
totalOfDiscountedItems += Discounted(item);
message = "item: " + item.Description + " price: " + item.Amount +
          " discounted price: " + Discounted(item);
total += Discounted(item);
// ...

private decimal Discounted(Purchase item)
{
    return item.Amount * (1.0m - MemberDiscount);
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Replace temp with query

## Steps

- Make temp const
- Extract Method on the right-hand-side
- Replace temp references with the query
- Remove temp

Why?

Concerns?

Can we go the other way?

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Old-school design planning

“Big Design Up-Front” (BDUF)


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"UML Diagram," courtesy https://www.flickr.com/photos/acehoward/
License: https://creativecommons.org/licenses/by/2.0/

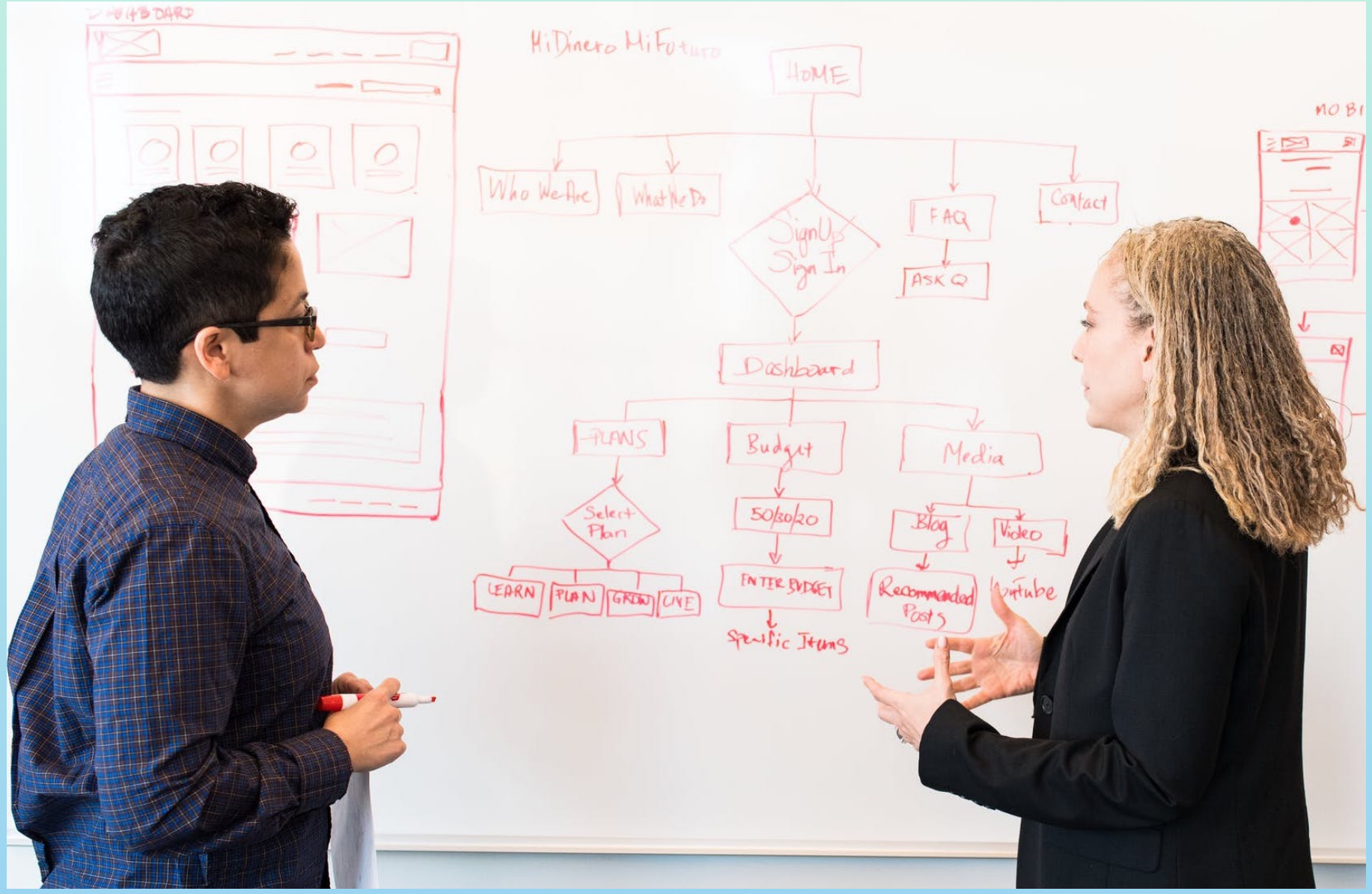
# Design occurs at every possible juncture

- Project
  - Release
    - Iteration
      - Day
        - Task
          - Test


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"UML Diagram," courtesy https://www.flickr.com/photos/acehoward/
License: https://creativecommons.org/licenses/by/2.0/

# This is design




Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Premature Generalization

- Need might never materialize
- Details might change
- Interim complexity $

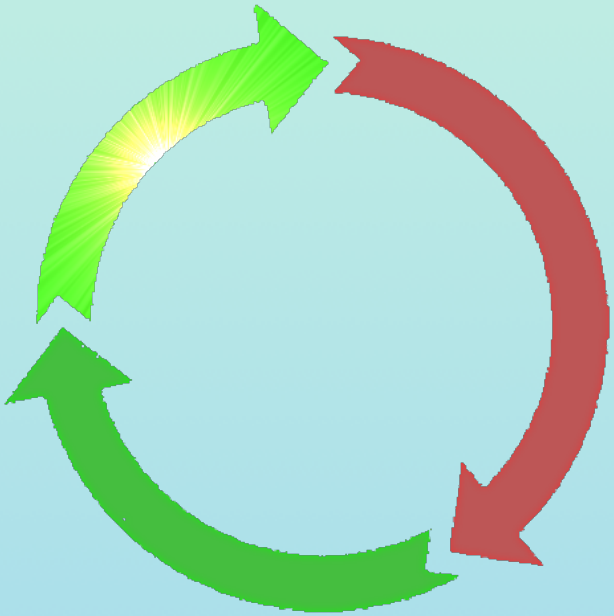
Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"crystal," courtesy https://www.flickr.com/photos/bb_matt/
License: https://creativecommons.org/licenses/by/2.0/

# Refactoring Guidelines

Never skip!

Single-goal



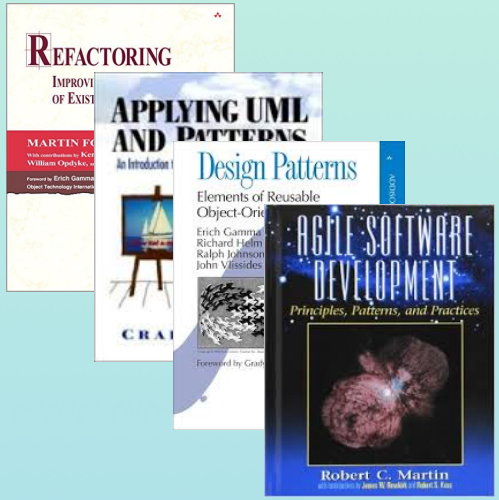
Run *all* tests

It's your responsibility

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Design Drivers / Guidelines

- Code smells
- SOLID
- Design patterns
- GRASP
- Simple design
- ...



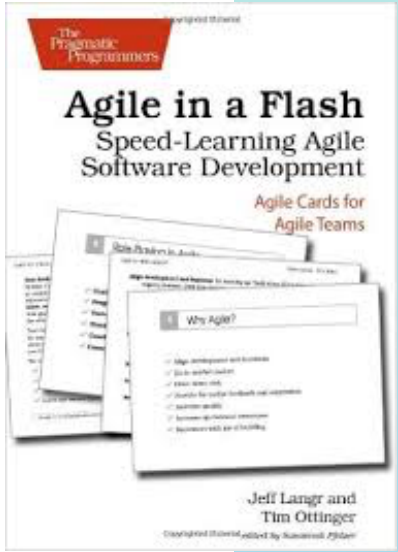

**It's all good!**

*Does everything point to the same place?*

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

## 41 Build Superior Systems with Simple Design

- All tests must pass
- No code is duplicated
- Code is self-explanatory
- No superfluous parts exist




Kent Beck's (ordered) rules for emergent design

# Exercise: Simple design rules

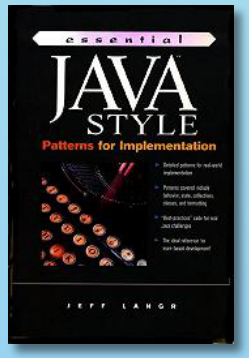
- In ./src/pos/routes/checkout.js
  - Stamp out duplication!
  - Strive for rapid readability
  - Have you gone too far?

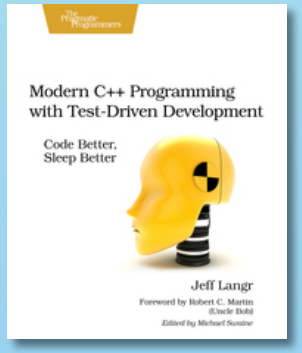
Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

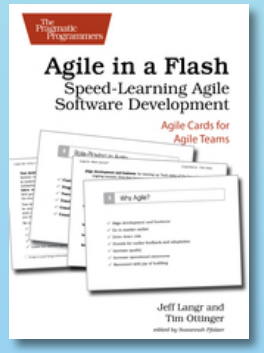
Kahoot!

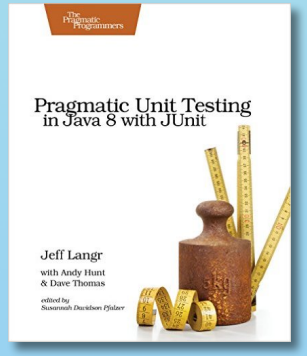
# Test doubles

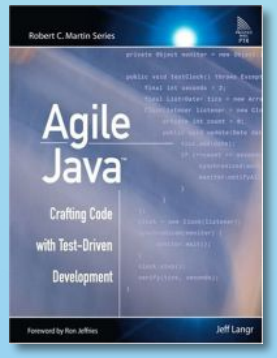
LANGR | SOFTWARE SOLUTIONS

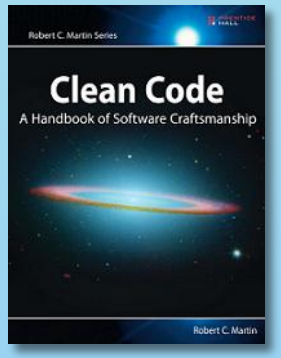


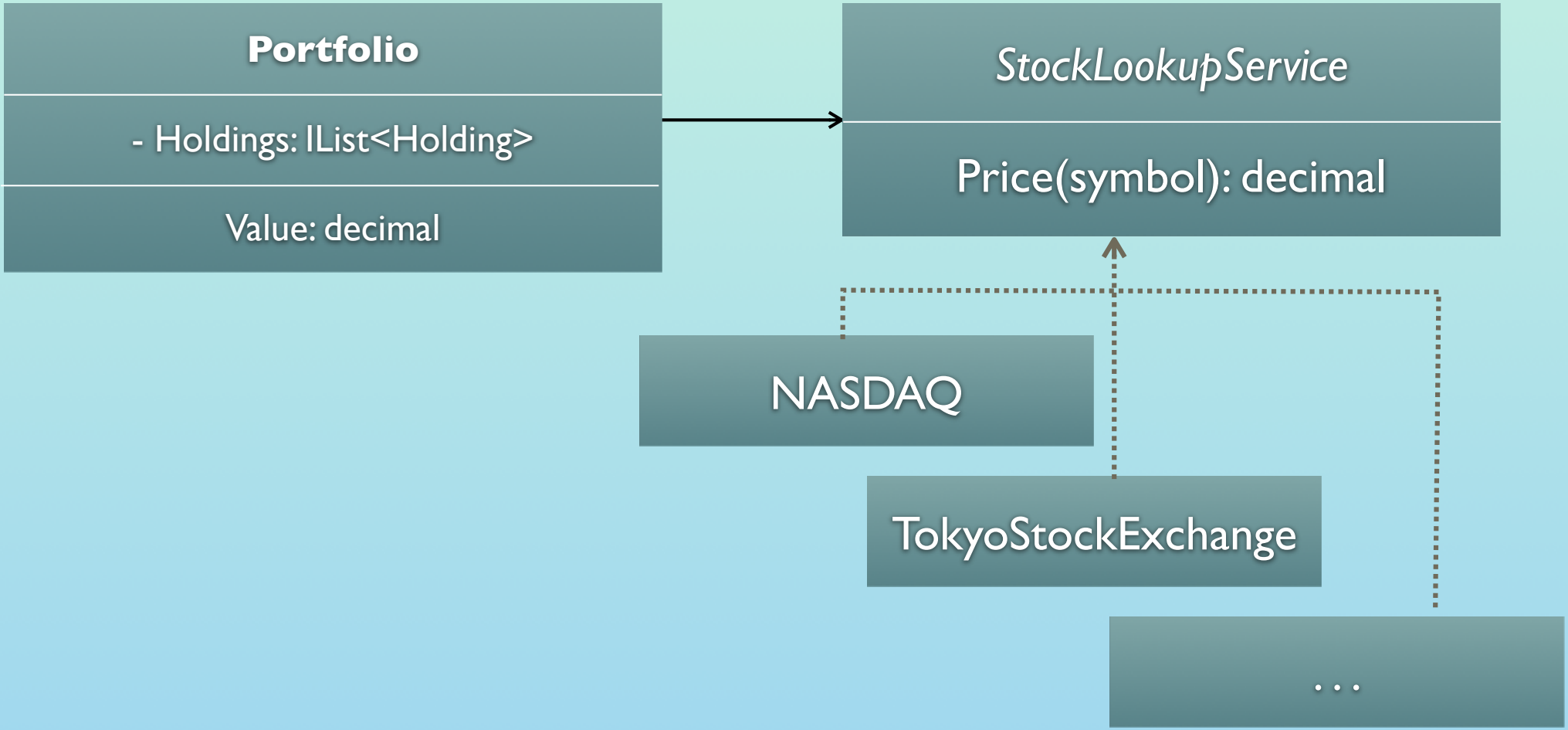



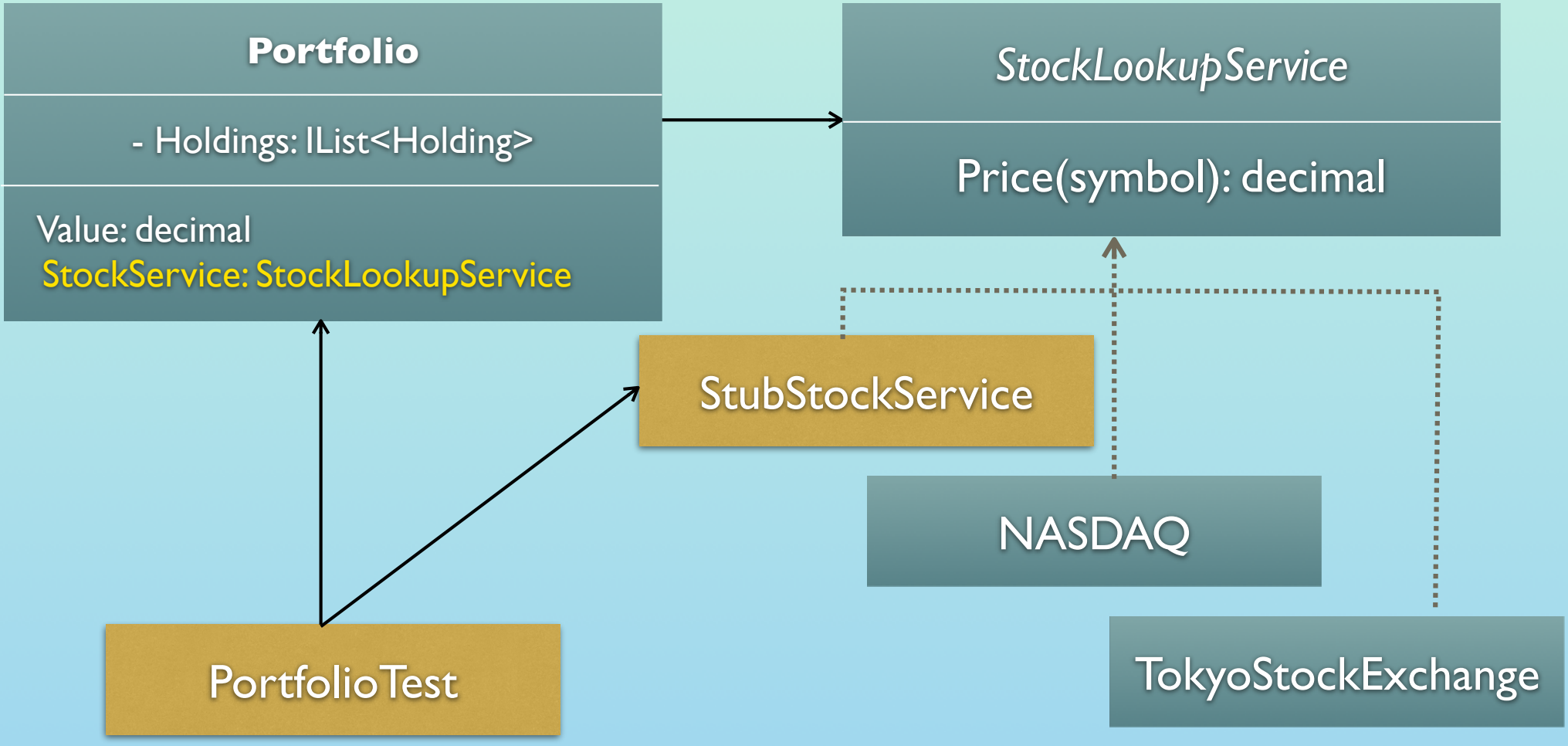
# Testing challenge: Portfolio





Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

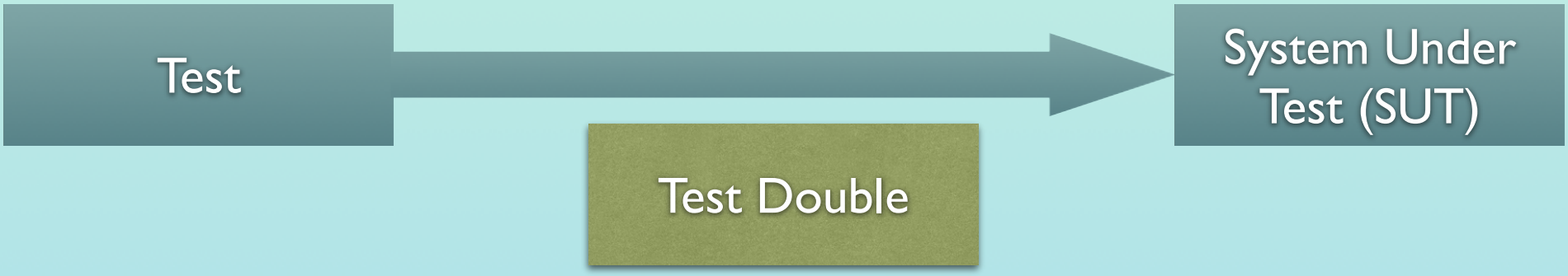
# Using a test double




Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Demo: portfolio value

# Injecting a test double



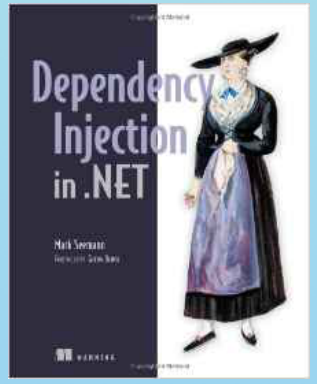

Constructor / setter / property

Factory

Factory method override

Injection tool

Consider:





Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Exercise: hand-crafted test double

Change PatronsService to do a credit check on patron create.

```
public interface ICreditCheck
{
   int CreditScore(string ssn);
}
```

1. Add a social security # property to Patron.
2. Don’t add patron if credit score < 630.
3. Create a production service to use a (temporary) check that always returns 800.

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

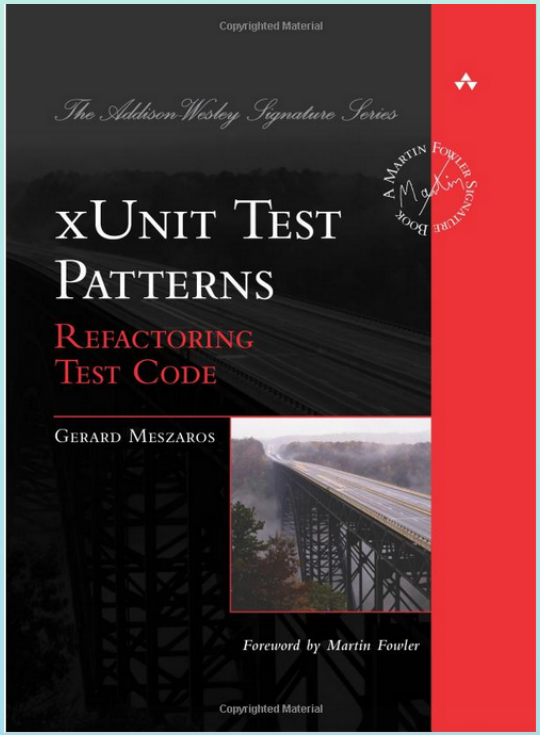
# Test double terms

**Stub:** dumb emulation

**Spy:** captures values to verify*

**Mock:** self-verifies

**Fake:** whole collaborator emulation




* Caution: Tool implementation of spies vary as to whether the real method gets called or not.


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Mock tools

Rhino Mocks

TypeMock Isolator

Moq

NMock

*History suggests more & changes coming!*

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Moq: A simple stub

https://github.com/moq/moq4

an expectation

```
var mock = new Moq.Mock<IList<string>>();
mock.Setup(l ⇒ l.Count)
    .Returns(42);
IList<string> list = mock.Object;

Assert.That(list.Count, Is.EqualTo(42));
```

https://github.com/Moq/moq4/wiki/Quickstart

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Moq: Expectations specify args

```
mock.Setup(l ⇒ l[15]).Returns(“1500");
mock.Setup(l ⇒ l[16]).Returns("1600");

Assert.Equal("1500", list[15]);
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Moq: Retrieve the Mock manager object

```
private IList<string> list;

public MiscTest()
{
    var mock = new Mock<IList<string>>();
    list = mock.Object;
}

[Fact]
public void moqRetrieveMock()
{
    Mock.Get(list).Setup(l ⇒ l.Count).Returns(42);
    Assert.Equal(42, list.Count);
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Exercise: Moq

Replace your hand-crafted CreditCheck test double by using Moq.

Use Mock.Get to eliminate mock construction details from the test.

Make sure the CreditScore method is overridable!

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Moq: Strict mocks

```
var mock = new Mock<ICreditCheck>(MockBehavior.Strict);
mock.Setup(creditCheck ⇒ creditCheck.CreditScore("9999"))
    .Returns(ICreditCheck.GoodCreditThreshold - 1);
service.CreditCheck = mock.Object;

var id = service.Create(new Patron {Name = "a", SSN = “9999"});

Assert.Null(service.FindById(id));
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

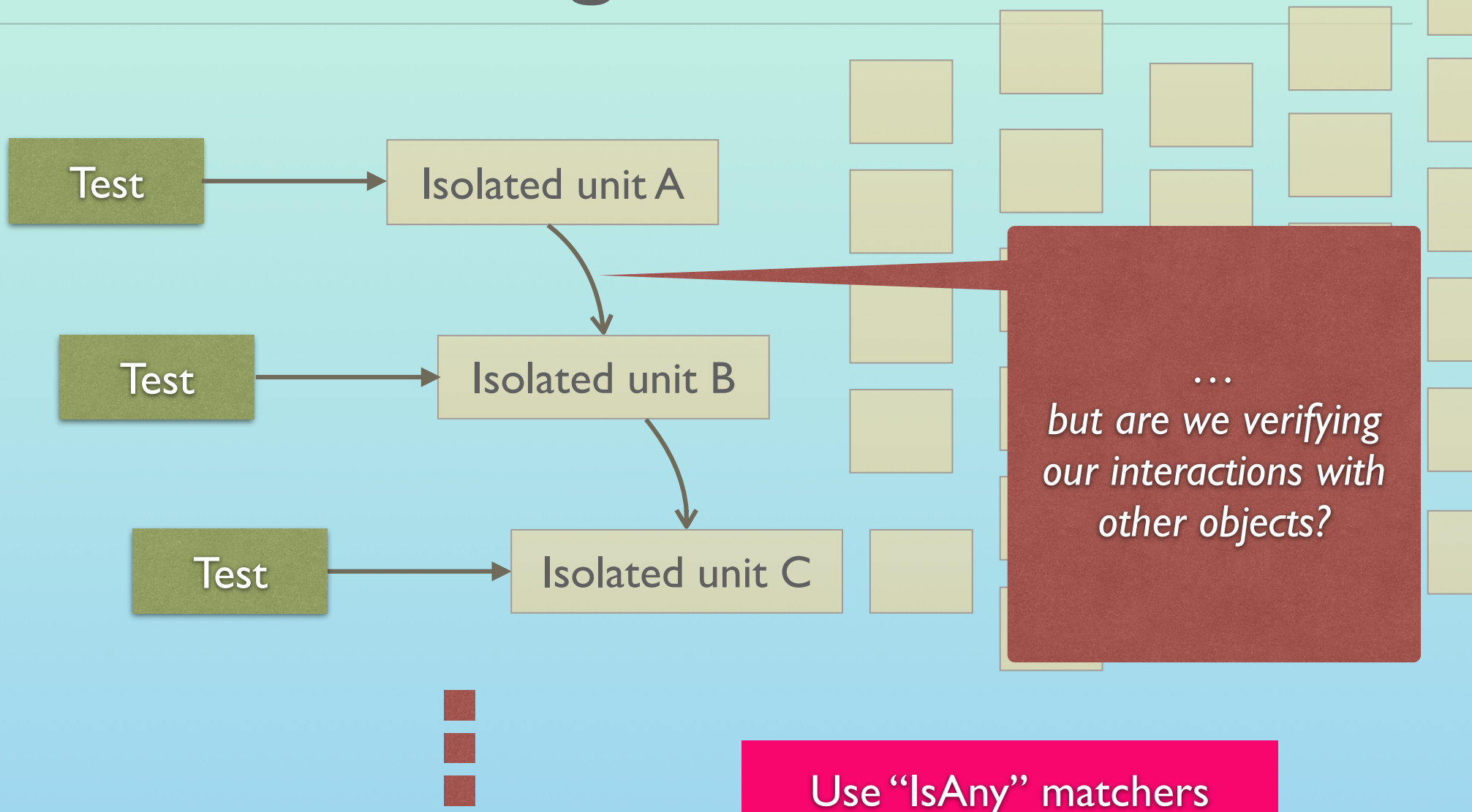
# Moq: Argument matchers

```
var mock = new Mock<IDictionary<object, string>>();
var dictionary = mock.Object;

mock.Setup(d ⟹ d[It.IsAny<string>()])
    .Returns("a fish");

Assert.Equal("a fish", dictionary["smelt"]);
Assert.Equal("a fish", dictionary["trout"]);
Assert.NotEqual("a fish", dictionary[42]);
```

**Also:** `IsInRange`, `IsRegex`, `IsNotNull`, `IsNotIn` (a sequence)

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

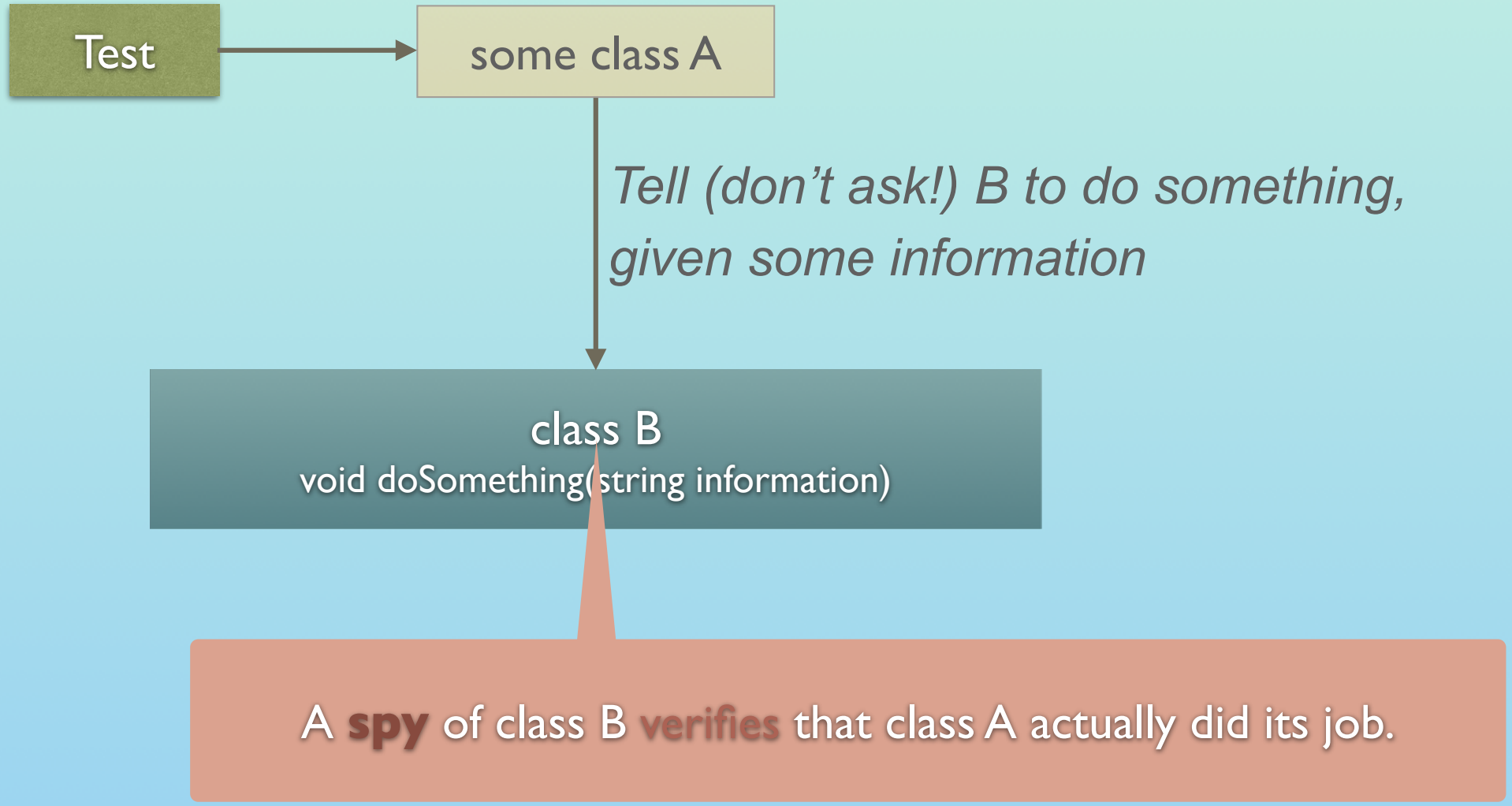
# “Unit” testing




Use “IsAny” matchers sparingly!


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# “Tell, Don’t Ask”





Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Test-driving interactions

We have this already:

```
public class DictionaryService
{
    public Auditor EventAuditor { get; set; }
    public virtual void Add(string word, string definition)
    {
        Words[word] = definition;
        EventAuditor.Log($"adding {word}:{definition}");
    }
}
```

We want to test-drive this.

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Moq: Spy via mock verify

```
[Fact]
public void LogsAuditRecordOnAdd()
{
   var auditorSpy = new Mock<Auditor>(MockBehavior.Strict); // same old,
   dictionaryService.EventAuditor = auditorSpy.Object;      // same old

   dictionaryService.Add("dog", "a canine");

   auditorSpy.Verify(auditor ⇒ auditor.Log("adding dog:a canine”));
}
```


Copyright © Langr Software Solutions, Inc. 2020.  All rights reserved.

# Spy / verify exercise

Verify that an email is sent when a would-be patron is declined for bad credit.

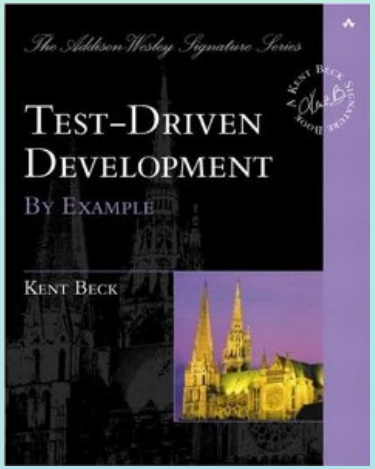
```
public interface IEmail
{
    void Send(
      string recipient, string sender, string title, string content);
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Schools of mock

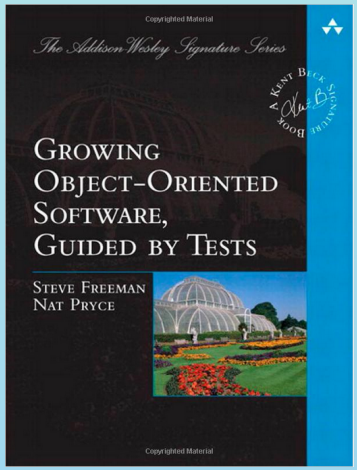
Classic ("Detroit")

- algorithmic approach
- verification of state
- tests drive code from specific to general




London

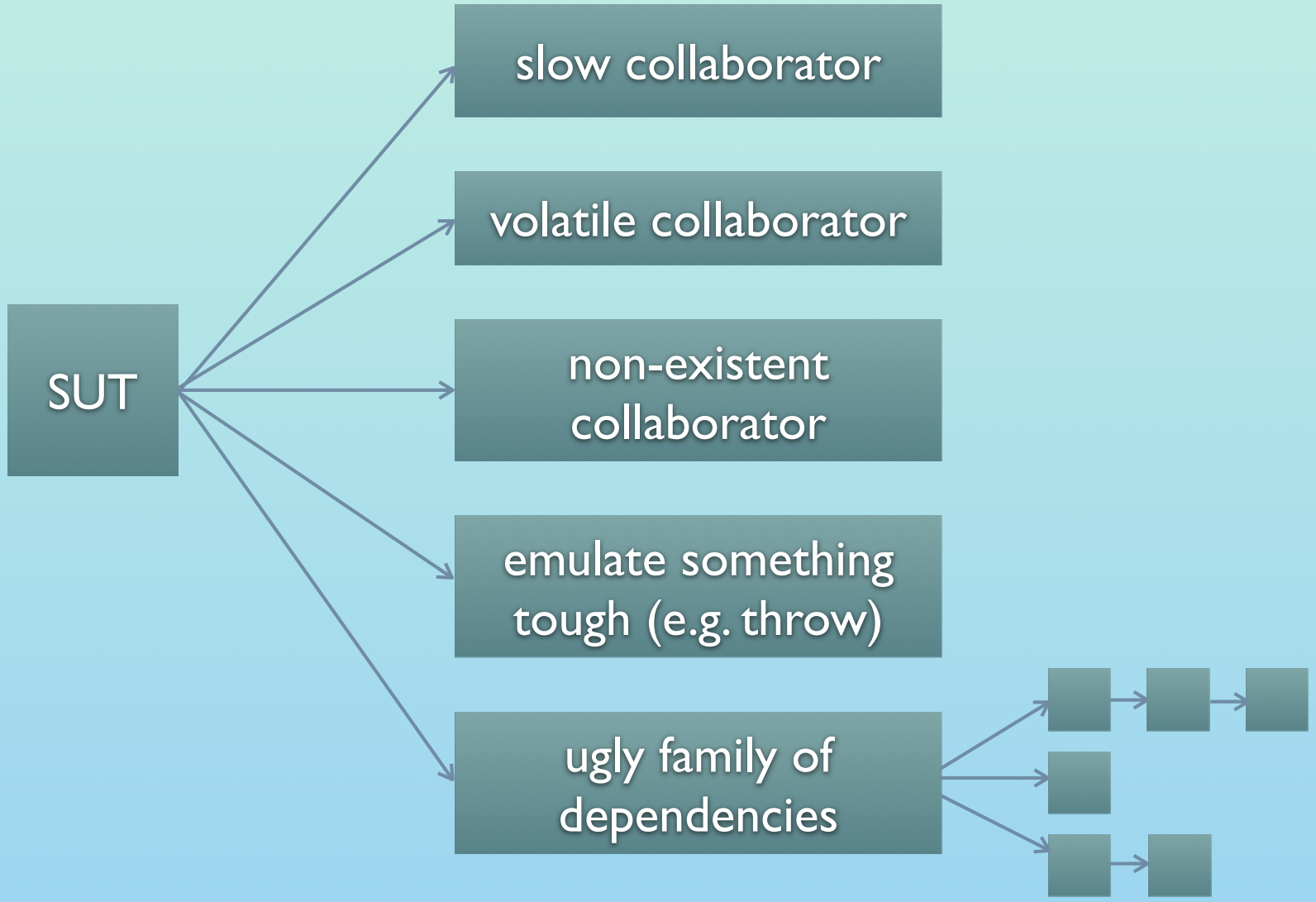
- roles, responsibilities, and interactions
- "Fake it until you make it"
- drive incrementally from the outside in




**Summarized well in Martin Fowler's "Mocks Aren't Stubs"**


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Classic school: When to mock




Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# The pragmatic mocker

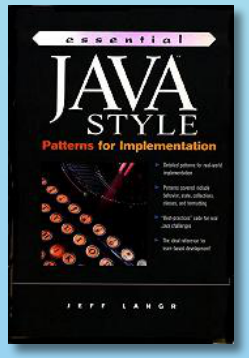
- Writes integration tests where mocks are used
- Watches coupling between tests & implementation
- Seeks to mock only direct collaborators
- Avoids fakes (and if necessary, test-drives 'em!)
- Isolates, minimizes the use of test doubles
  - But prefers testability

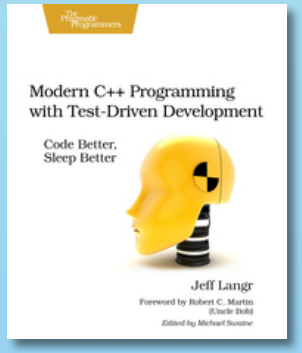

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

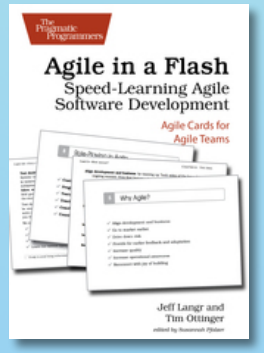
"Jester," courtesy https://www.flickr.com/photos/mrpolyonymous/
License: https://creativecommons.org/licenses/by/2.0/

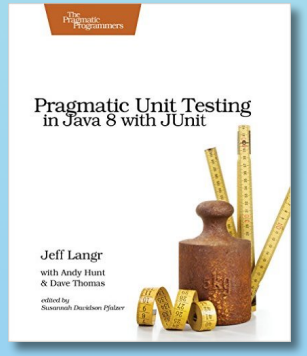
# Moq: Miscellany

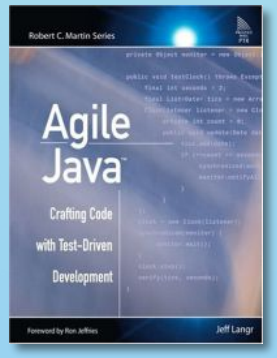
LANGR | SOFTWARE SOLUTIONS

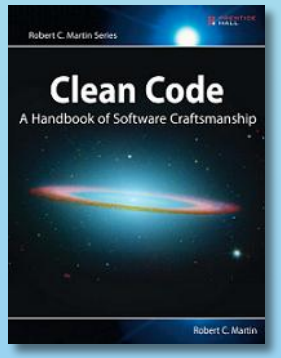


# Moq: Mock creation via `Mock.Of`

```
public interface Cache
{
    string LookUp(string key);
}
```

```
[Fact]
public void MockOfSyntax()
{
    var cache = Mock.Of<Cache>(c => c.LookUp("smelt") == "a fish");

    Assert.Equal(cache.LookUp("smelt"), "a fish");
}
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Emulating an exception throw

```
mock.Setup( ... ).Throws<InvalidOperationException>();

mock.Setup( ... ).Throws(new Exception(“ ... "));
```

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Moq: Arg matcher with predicate

```
mock
    .Setup(d ⇒ d[It.Is<string>(s ⇒ s.Last() == 's')])
    .Returns("maybe plural");
mock
    .Setup(d ⇒ d[It.Is<string>(s ⇒ s.Last() ≠ 's')])
    .Returns("maybe singular");

Assert.Equal("maybe plural", dictionary["dogs"]);
Assert.Equal("maybe singular", dictionary["trout"]);
```


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Moq: Return transformed arg capture

```
mock.Setup(x ⇒ x.StringStuff(It.IsAny<string>()))
    .Returns((string arg) ⇒ arg.ToLower());
```


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Moq: Callbacks

```
var mock = new Mock<DictionaryService>();
var i = 0;
var definitions = new string[] { "a fish", "did smell" };
mock.Setup(d ⇒ d.LookUp("smelt"))
    .Returns(() ⇒ definitions[i])
    .Callback(() ⇒ i++);
var dict = mock.Object;

Assert.That(dict.LookUp("smelt"), Is.EqualTo("a fish"));
Assert.That(dict.LookUp("smelt"), Is.EqualTo("did smell”));
```

*Clever!*

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# TDD wrap-up exercise

- New scanner mode: Inventory reconciliation
- Triggered by special patron ID (99999999)
- AcceptBarcode collects scanned materials
- CompleteCheckout produces a report
  - and returns to "regular" state

Work outside in:

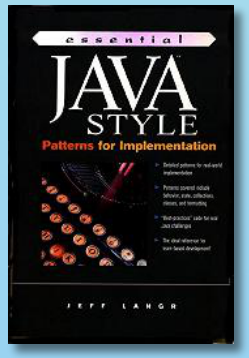
- Stub collaborators until ready to build them out

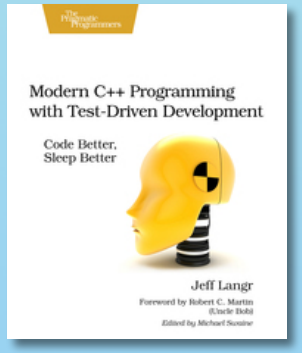
Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

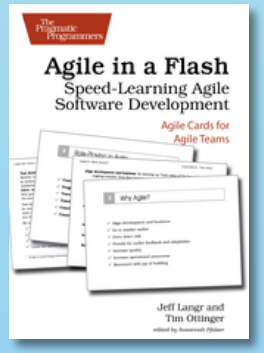
"barcode_scan3_code128.png," courtesy https://www.flickr.com/photos/mdornseif/
License: https://creativecommons.org/licenses/by-sa/2.0/

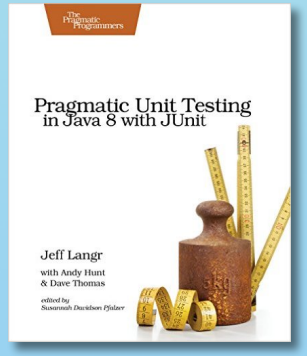
# Sustaining TDD

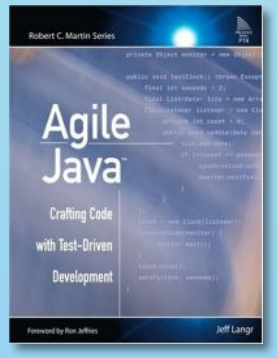
LANGR | SOFTWARE SOLUTIONS

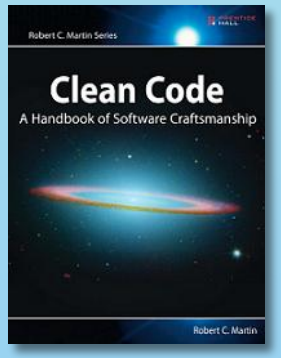


# Next week

- Establish team policies & standards around TDD
- “Stop the line” mentality for CI build
- Reviews must include the tests
- Consider pairing / mobbing


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"Kickoff," courtesy https://www.flickr.com/photos/edrost88/
License: https://creativecommons.org/licenses/by/2.0/

# Growing TDD

- Sharing sessions
- Challenges / contests
- Coaches & champions
- "Stop the line" mentality
- Influential, dynamic metrics
- More mobbing

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"Kickoff," courtesy https://www.flickr.com/photos/10209472@N03/
License: https://creativecommons.org/licenses/by-sa/2.0/

# Sustaining TDD

- Daily stand-down / discussions / retros
- "Why" reminders
- Review standards regularly
- Keep tests fast!
- Always use tests as an entry point
- Refactor, refactor, refactor
  - Tests too!


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"Infinity!" courtesy https://www.flickr.com/photos/thecampbell/
License: https://creativecommons.org/licenses/by/2.0/

# Mastering TDD

TDD is a skill.

*Practice it (really!).*

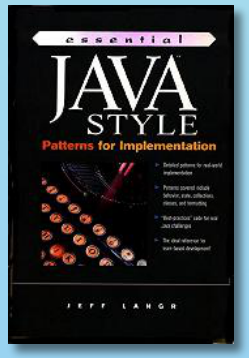

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

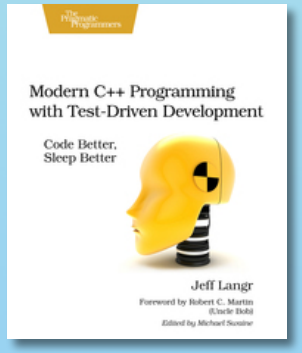
"Beautiful and dusty," courtesy https://www.flickr.com/photos/80435089@N08/
License: https://creativecommons.org/licenses/by/2.0/

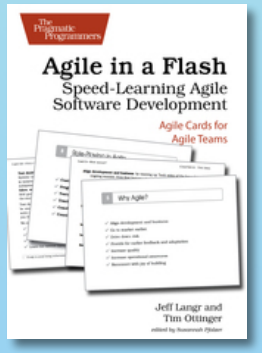
# Miscellaneous Topics

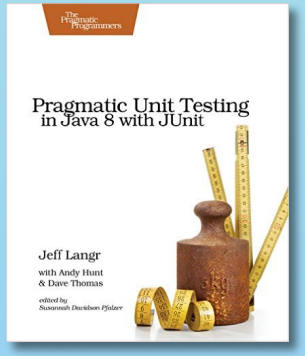
ICON
AGILITY SERVICES

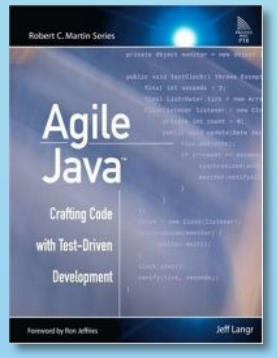
LANGR | SOFTWARE SOLUTIONS

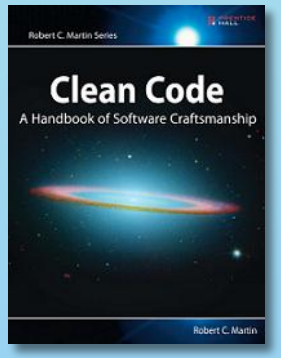


# Facts vs. theories

The stated "theory:"

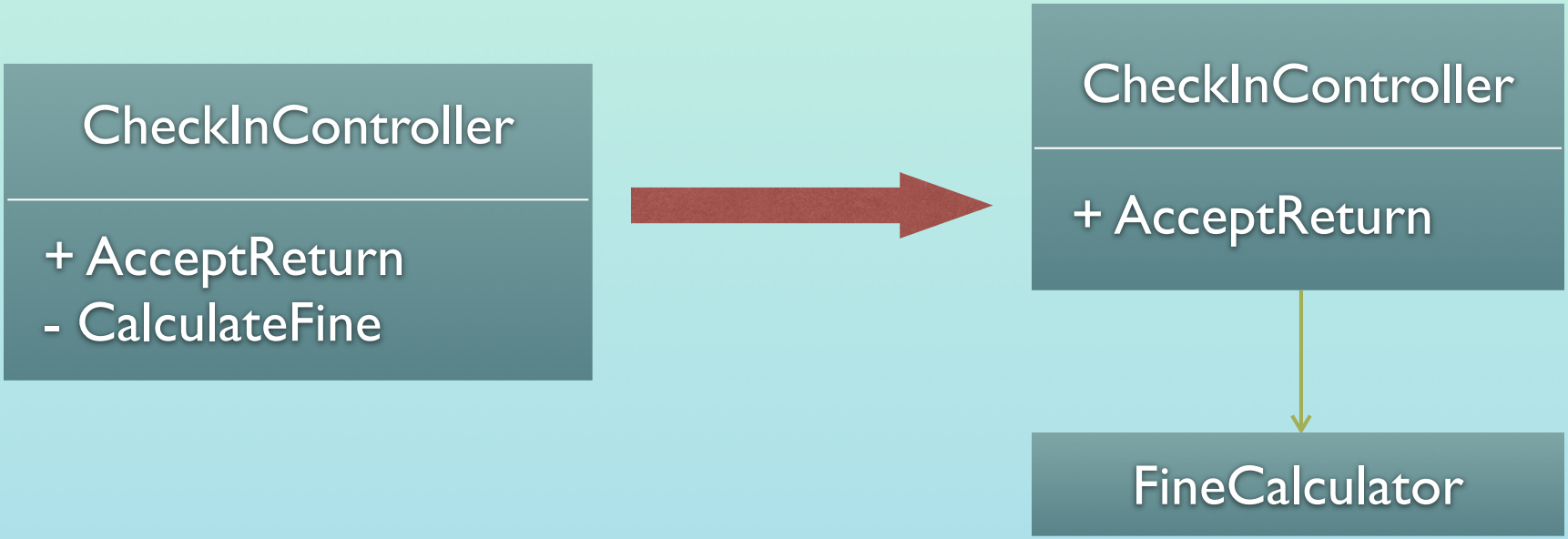
*"Facts are tests which are always true. They test invariant conditions. Theories are tests which are only true for a particular set of data."*

The pragmatic use:

```
[Theory]
[InlineData(0, 0, 0)]
[InlineData(1, 19, 1)]
[InlineData(10, 20, 10)]
[InlineData(55, 88, 11)]
public void PrimeFactors(ulong x, ulong y, ulong expected)
{
    Assert.Equal(expected, GCD(x, y));
}
```


Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Testing Private Behavior?




Or simply relax access.

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Rules of Ten

1

Green & clean in < 10 (minutes)

Delete and repeat if you're late!

... taking smaller steps this time.

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"Egg Timer," courtesy https://www.flickr.com/photos/engendercdp/ spikeheap/
License: https://creativecommons.org/licenses/by/2.0/

# Rules of Ten

Debate stops at 10 (minutes)

"Show me."

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

"Egg Timer," courtesy https://www.flickr.com/photos/engendercdp/ spikeheap/
License: https://creativecommons.org/licenses/by/2.0/

# Rules of Ten

3

All unit tests run in < 10 (seconds).

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.



"Egg Timer," courtesy https://www.flickr.com/photos/engendercdp/ spikeheap/
License: https://creativecommons.org/licenses/by/2.0/

# Fixing a Slow Test Run

Warn on tests < ***n*** *ms*

Fail on tests > ***n*** ms

*Running a subset of tests increases risk.*

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Uncle Bob's TPP

Transformations have a preferred ordering, which if maintained by ordering of tests, prevents "long outages" in TDD.

The Priority List

(**{}->nil)** no code at all->code that employs nil
**(nil->constant)**
**(constant->constant+)** a simple constant to a more complex constant
**(constant->scalar)** replacing a const. with a variable or an argument
**(statement->statements)** adding more unconditional statements.
**(unconditional->if)** splitting the execution path
**(scalar->array)**
**(array->container)**
**(statement->tail-recursion)**
**(if->while)**
**(statement->recursion)**
**(expression->function)** replacing expression w/ a function or algorithm
**(variable->assignment)** replacing the value of a variable.
**(case)** adding a case (or else) to an existing switch or if

"As the tests get more specific, the code gets more generic."

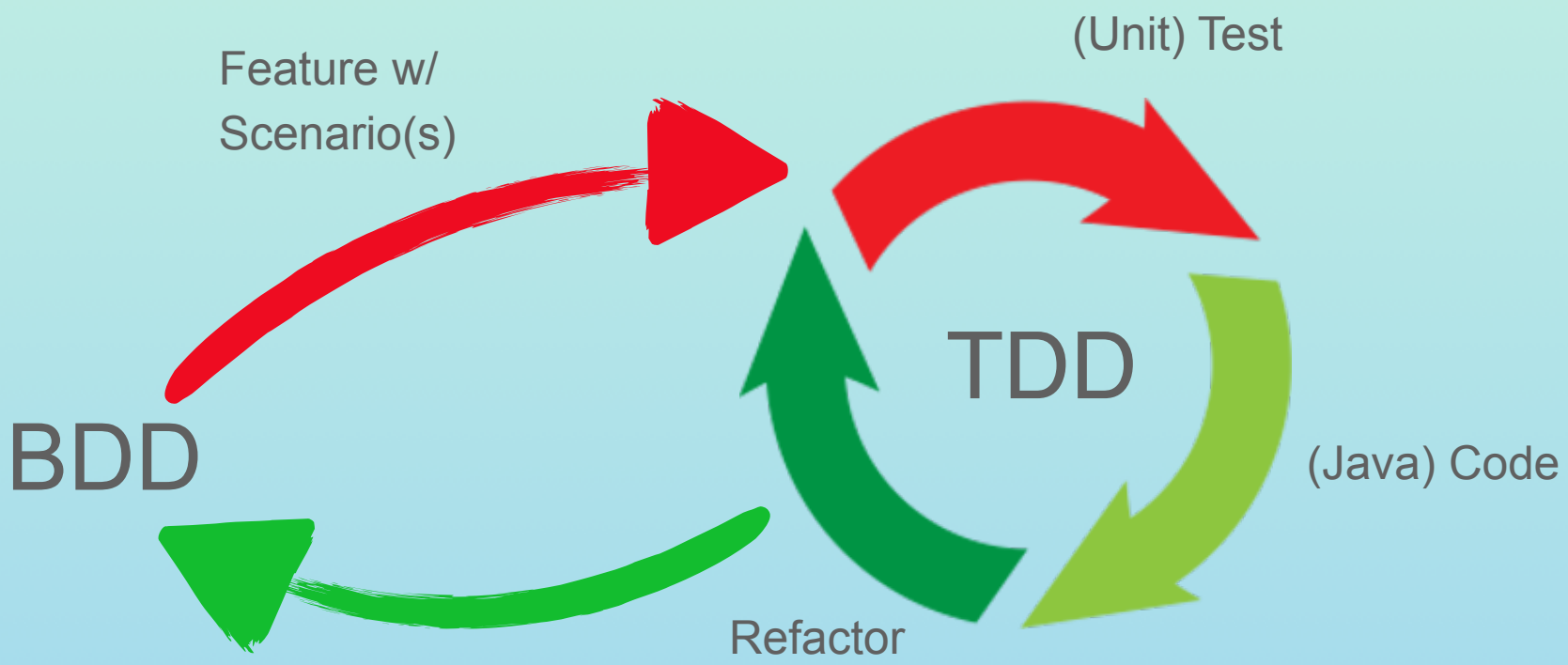
http://thecleancoder.blogspot.com/2011/02/fib-t-p-premise.html

https://8thlight.com/blog/uncle-bob/2013/05/27/TheTransformationPriorityPremise.html

Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

*Per and updated by Uncle Bob.*

# Behavior-Driven Development (BDD)




Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# References / Reading List


[Beck2002] Beck, Kent. **Test-Driven Development: By Example**. Addison-Wesley, 2002.

[Feathers2005] Feathers, Michael. **Working Effectively With Legacy Code**. Prentice Hall, 2005.

[Fowler1999] Fowler, Martin. **Refactoring: Improving the Design of Existing Code**. Addison-Wesley, 1999.

[Freeman, 2009]. Freeman, S.; Pryce, N. **Growing Object-Oriented Software, Guided By Tests**. Addison-Wesley, 2009.

[Langr2005] Langr, Jeff. **Agile Java: Crafting Code With Test-Driven Development**. Prentice Hall, 2005.

[Langr2011] Langr, Jeff and Ottinger, Tim. **Agile in a Flash**. Pragmatic Programmers, 2011.

[Langr2013] Langr, Jeff. **Modern C++ Programming With Test-Driven Development**, Pragmatic Programmers, 2013.

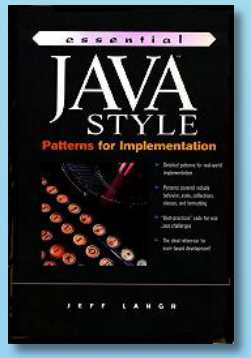
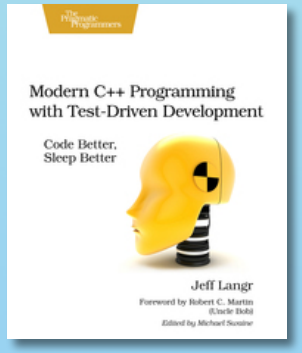
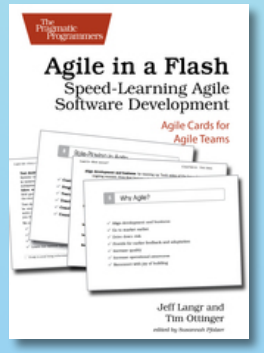
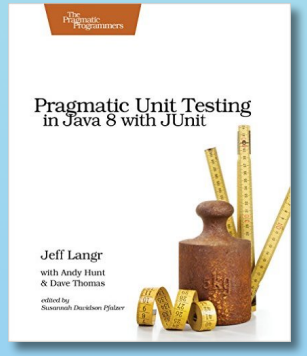
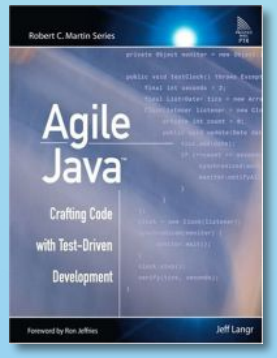
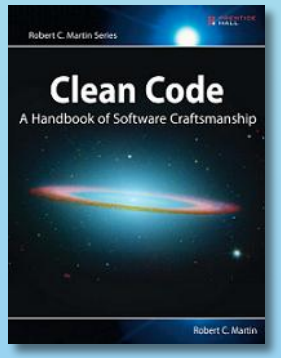
[Langr2015] Langr, Jeff, et. al. **Pragmatic Unit Testing in Java 8 with JUnit.** Pragmatic Programmers, 2015.

[Martin2002] Martin, Robert C. **Agile Software Development: Principles, Patterns, and Practices**. Prentice Hall, 2002.

[Meszaros2007] Meszaros, Gerard. **xUnit Test Patterns: Refactoring Test Code**. Addison-Wesley, 2007.



Copyright © Langr Software Solutions, Inc. 2020. All rights reserved.

# Thank you.

LANGR | SOFTWARE SOLUTIONS