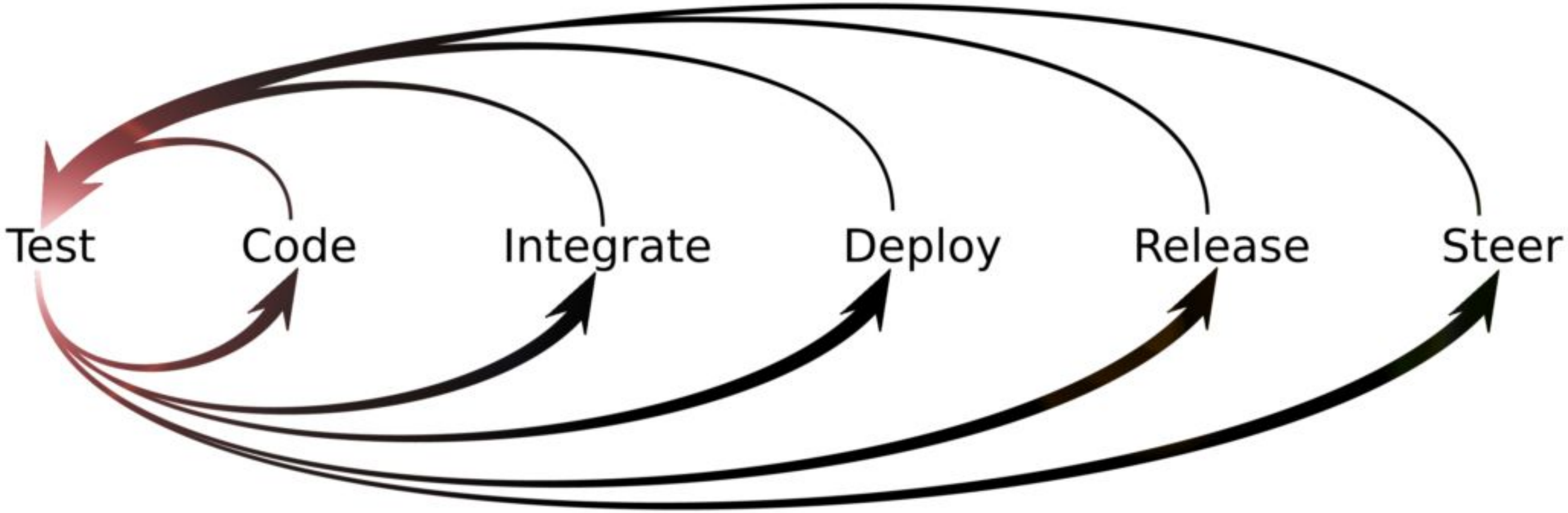
Intro to TDD

TDD Done Right

How should you practise TDD... really?

Talk by André Araújo, October, 2020

Test-Drive all the things!



Source: http://www.zeroplayer.com/tdd.png

Underplayed Premises of TDD:

The **<u>Money</u>** Premise — We're in this for the money.

The <u>Judgment</u> Premise — We'll rely on individuals making local decisions.

The **<u>Correlation</u>** Premise — Internal quality *is* productivity.

he **Chaining** Premise — We'll test mostly in very small parts.

The **Steering** Premise — Tests & testability are first-class design participants.

Source: GeePaw Hill, a software coach, spoke about five underplayed premises of TDD at eXperience Agile 2018

The World's Simplest Instructions for TDD

When you code, alternate these activities:

➔ One

Add a test, get it to **fail**, and **write code** to **pass the test** (DoSimpleThings, CodeUnitT estFirst)

➔ Two

Remove duplication (OnceAndOnlyOnce, DontRepeatYourself, ThreeStrikesAndYou Automate)

Just Before We Get Going...

How to Write a Test List

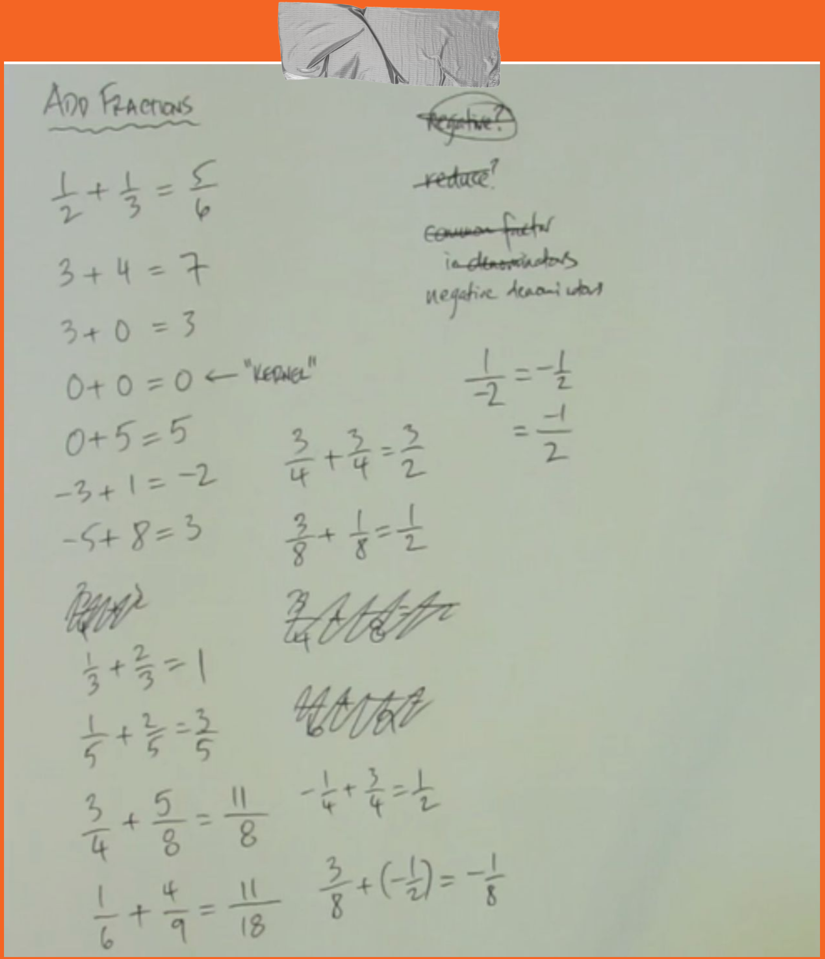
David Allen's slogan:

Your brain is for having ideas, not storing them.

Source: *Getting Things Done*."Task Inbox" as described by David Allen

Source: Test-Driven Development: By Example.

Let's DO IT



ADD FRACTIONS

$\frac{1}{2} + \frac{1}{3} = \frac{5}{6}$

$3 + 4 = 7$

$3 + 0 = 3$

$0 + 0 = 0 \leftarrow$ "KERNEL"

$0 + 5 = 5$

$-3 + 1 = -2$

$-5 + 8 = 3$

[illegible]

$\frac{1}{3} + \frac{2}{3} = 1$

$\frac{1}{5} + \frac{2}{5} = \frac{3}{5}$

$\frac{3}{4} + \frac{5}{8} = \frac{11}{8}$

$\frac{1}{6} + \frac{4}{9} = \frac{11}{18}$

$\frac{3}{4} + \frac{3}{4} = \frac{3}{2}$

$\frac{3}{8} + \frac{1}{8} = \frac{1}{2}$

[illegible]

[illegible]

$-\frac{1}{4} + \frac{3}{4} = \frac{1}{2}$

$\frac{3}{8} + \left(-\frac{1}{2}\right) = -\frac{1}{8}$

~~Negative?~~

~~reduce?~~

~~common factor in denominators~~

negative denominators

$\frac{1}{-2} = -\frac{1}{2}$

$= \frac{-1}{2}$

Some tips!

➔ **Version Control?**
Any time the tests pass 100%, I may commit.

➔ **One more thing**
If you have trouble with time "getting away from you" or spending too much time going down "the wrong path", then try the **Pomodoro Technique.**

Let's CODE

Clone this and follow me:

git@github.com:syngenta-digital/how-should-you-practise-tdd-really.git

Good luck!

- **Wiki Community**, “Test-Driven Development”
- “The Art of Agile Development: Refactoring”. James excerpts their book of the same title, in which they describe their approach to refactoring, including the notion of Shotgun Surgery.
- Andy Hunt and Dave Thomas, The Pragmatic Programmer.
- Liz Keogh, “Step Away From the Tools”. Most of us, when we learn a new technique, want tools to help us get started
- J. B. Rainsberger, “Primitive Obsession Obsession”. I wrote this article to discuss the code smell Primitive Obsession.
- Kent Beck, Test-Driven Development: By Example.
- Pomodoro Technique to transform lives: **http://www.pomodorotechnique.com.**

A special thanks to

➔ **J. B. Rainsberger**

He have helped a major government contractor see how to reduce billions of dollars in erroneous insurance and health benefit claims. Their necessarily-complicated COBOL-based system allows *everyone* to qualify for medical benefits related to child-birth, even if they've never given birth. Over dinner he sketched a plan to replace the most expensive parts of their legacy system gradually and safely, and now they can save their client significant sums of money in weeks instead of months.

All this material is copy by https://www.jbrains.ca/training/ from "The world's best introduction to test-driven-development"