Continuous Delivery ltd

TDD & BDD
Design Through Testing

Exercise 3
Dependency Injection

Overview

To test anything, we need to find points at which we can measure it in some way. This is an important idea in TDD. By starting with a test, we amplify our desire, our need, to make our code more easily testable. This is a very good thing, because that has the side-effect of improving the quality of our design.

Dependency Injection is one very clear way in which this is true. Dependencies that are provided to your code always mean that your code is more flexible, because, by definition, we can change the dependency that we inject, so our code is less-coupled to that specific dependency. It also means that our code is testable, where otherwise it may not be.

Dependency Injection is an important idea in design as well as in TDD!

In this exercise, we will improve the String Calculator that we created earlier. We will add the ability to the calculator to render a result into a nice readable form, and add the ability to display that result.

Goals of the Exercise

To practice ***RED, GREEN, REFACTOR!***

Use ***Dependency Injection*** to improve the testability of our code.

Exercise

Add the ability to display results from the String Calculator when an addition is performed.

When the calculator is called with valid inputs, format the results and display them.

For example:

> given ***add("1,2")***
>
> expect ***"1 + 2 = 3"*** to be displayed.

Advice

Remember this is an exercise in Dependency Injection, so treat the Display as a dependency and inject it! Use that to test your results.

Remember The Rules of the Game!

Make progress in very small steps.

The goal is to practice TDD, more that it is to solve the problem. Don't rush ahead to solve the problem and forget to test.

Don't add new code, unless you have a failing test that makes you.

Always refactor on a passing test, and run the tests after each, small change to verify your changes.