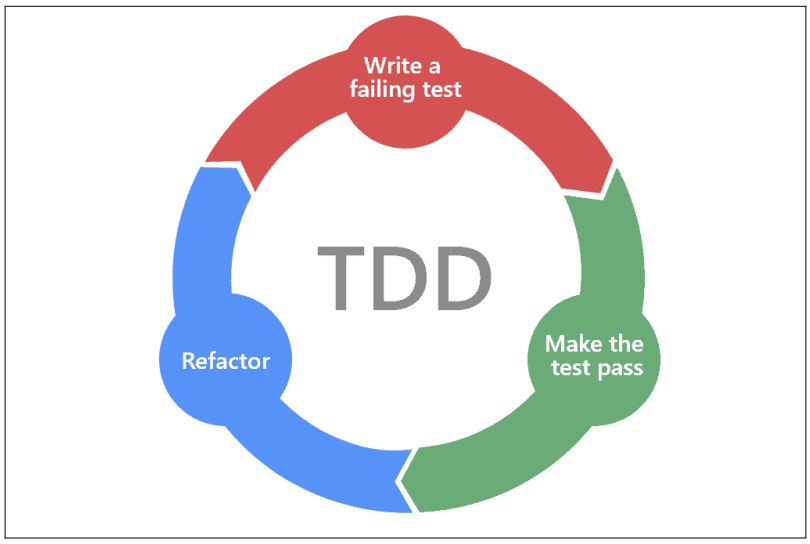
CS3318



Assignment 03 Test Driven Development

18 November 2020
Assignment: **35 marks**
Type: Individual Assignment (code and report)

OVERVIEW

Test Driven Development (TDD) is software development approach in which test cases specify and validate the functionality of the code. In that sense, TDD is a specification technique with a side-effect of ensuring that source code is comprehensively tested. The TDD approach, is described as a cycle of Red-Green-Refactor. Run a test, see it fail (red), implement the simplest code to make the test pass (green), and then refactor the code so your test stays green and your code is improved.

Following a TDD approach requires creating the test class, creating the test methods, and then creating empty implementations of the code that will eventually become production code. An IDE such as IntelliJ IDEA can automate a lot of this process.

In this assignment you will develop an application using TDD in IntelliJ IDEA using Maven and git.

Deliverables

Source and test code, an archive of your git repository and a report submitted via Canvas.

Due Date

Thursday 17th December 2020.

Introduction

To employ TDD effectively you need a program specification. TDD is very good at detailed specification and validation, but does not help with the overall design process. When TDD is used in conjunction with unit testing, the developer writes a single test i.e. unit test and then just enough production code to fulfil that test. Remember that TDD is a thinking process that results in unit tests, while "unit testing" is writing many small tests that each test one aspect of an object's behaviour.

The Assignment Task

In this assignment you will use the TDD methodology to develop a `Colour` class that can represent and operate on colours that can be represented in a variety of colour models. You will use unit testing to implement the TDD process, in particular state testing to test the direct inputs and outputs of a test `Colour` object.

The `Colour` class must implement the following functionality

- To create a `Colour` object you must specify exactly three colour components
- It is not allowed to create a `Colour` object without specifying valid component values
- Valid component values are in the range 0 to 255
- The default colour model is RGB
- A different colour models is specified by using a `String` value to identify the name of the model when creating the `Colour` object
- Colours can be compared, to be equal they must have the same colour model and same component values
- Colours can be added, if they use the same model, by adding each of the components, however the value of each component cannot exceed 255

You will need to decide how your implementation of `Colour` presents this functionality and write appropriate tests. Do not exceed the scope of the specification (i.e. implement unnecessary functionality).

How to Proceed

Create a local git repository for your assignment. At each step of the TDD cycle commit your code (test, implementation, refactoring are three separate commits). The archive of your repository will capture your application of the TDD process.

See https://git-scm.com/docs/git-archive for details on archiving a git repository.

ASSIGNMENT REQUIREMENTS

Create a Maven-based Java project in IntelliJ IDEA, which should have the default structure with folders for production and test code. Include JUnit 5 as a dependency in Maven.

Employ TDD Best practices, which include

- Use descriptive names for test methods
- Write the test before writing the implementation code
- Only write new code when test is failing
- Rerun all tests every time implementation code changes
- All tests should pass before new test is written
- Refactor only after all tests are passing
- Write the simplest code to pass the test
- Write assertions first, act later
- Minimise assertions in each test
- Do not introduce dependencies between tests
- Tests should run fast

OUTCOMES

The assignment outcomes are

- Source and test code and an archive of your git repository.
- A 500 word report describing the TDD process and where you consider the approach was of benefit and detriment to the development process.
- You may also wish to comment on any issues you have identified with the specification.