TDD

Systembeschreibung

TDD
Systembeschreibung

1. Was ist TDD?

TDD steht für „Test Driven Development“ und ist eine Arbeitsweise, die ein effizientes Programmieren ermöglichen soll, indem die Programmierung in möglichst kleine Arbeitsschritte geteilt wird. Für jeden Schritt wird durch einen entsprechenden Test sichergestellt, dass er korrekt funktioniert, und wenn zu einem späteren Zeitpunkt der Code verändert wird, stellen die bereits implementierten Tests dabei direkt sicher, dass die Funktionalität des Codes nicht beeinträchtigt wurde.

Die Arbeitsweise besteht dabei aus drei Arbeitsschritten, die in festgelegter Reihenfolge beliebig oft wiederholt werden können. Steht die Aufgabenstellung fest, beginnt die Implementierung nicht mit dem Schreiben von Code, sondern mit dem Schreiben eines Tests. Die Frage ist also: Welches ist der denkbar einfachste Input, den die gewünschte Funktion bekommen könnte und was ist der korrekte Return-Wert? Dafür wird der Test geschrieben, der dann notwendigerweise fehlschlägt. Dies ist in TDD in der ersten Phase „RED“ implementiert. Als nächstes wird der Code soweit programmiert, dass er diesen ersten Test besteht (und sonst nichts). Dies erfolgt in der zweiten Phase „GREEN“. Ist dieser Schritt erfolgreich beendet

2. Startmenü

Wird das Programm gestartet, erscheint als erstes das Startmenü, in dem eine Übung und eine Erweiterung ausgesucht werden muss. Erst wenn beide Einstellungen gemacht wurden, ist der Wechsel in die erste Phase RED möglich.

3. Übungen

TDD bietet einige bereits vorgefertigte Aufgabenstellungen, aus denen ein ausgewählt werden muss, um einen TDD-Entwicklungszyklus zu starten. Eine Beschreibung, um welche Aufgabe es sich genau handelt, wird eingeblendet, sobald eine Aufgabe ausgesucht wurde.

4. Erweiterungen

TDD umfasst zwei Erweiterungen, von denen eine ausgewählt werden muss, um einen Entwicklungszyklus zu beginnen:

4.1. Babysteps

Babysteps zwingt den Entwickler, schneller und in kleineren Schritten zu programmieren. Es kann zwischen Zeiteinheiten von zwei oder drei Minuten gewählt werden. Der Entwickler hat dann gemäß seiner Einstellung Zeit, um einen Test bzw. seine Implementierung zu schreiben. Benötigt er länger als die vorgegebene Zeit, wird der Code gelöscht und die Uhr beginnt von neuem zu zählen.

4.2. Tracking

Tracking ist eine Erweiterung, bei der alle Tests/Codeiterationen und Fehler dabei grafisch dargestellt werden. Es werden dabei alle Schritte ab der ersten GREEN-Phase berücksichtigt, da es in der ersten RED-Phase noch keinen nutzbaren Code gibt. Es kann zu jedem Zeitpunkt der Programmierung diese grafische Ausgabe aufgerufen werden.

5. RED

RED ist die erste Phase in der testgetriebenen Entwicklung. In RED wird ein Test für den aktuellen Code geschrieben, der fehlschlagen soll. Es ist wichtig dabei, dass es sich nur um genau einen Test handelt, um die Kleinschrittigkeit der testgetriebenen Entwicklung beizubehalten. Sollte mehr als ein Test fehlschlagen, ist der Wechsel in die nächste Phase nicht möglich. Um dabei strukturiert vorzugehen, ist es empfehlenswert, den Test so zu konzipieren, dass die einfachste Anfrage, die beim aktuellen Stand des Codes zu einem falschen Ergebnis führt, in der aktuellen Iteration von RED implementiert werden sollte. Die Vorgehensweise ist also vom Allgemeinen zum Speziellen.
Wenn dies abgeschlossen ist, kann man mit dem Button „To GREEN“ in die nächste Phase wechseln. Wenn die aktuelle RED-Phase nicht die erste RED-Phase ist, gibt es bei Bedarf auch die Möglichkeit zurück zu RFCTR zu wechseln.

6. GREEN

In GREEN findet die eigentliche Programmierarbeit statt. In der vorhergehenden Phase RED wurde ein einzelner fehlschlagender Test definiert. Es geht nun also darum, den Code so zu erweitern, dass alle vorhergehenden Test sowie der zuletzt aufgenommene problemlos durchlaufen werden.
Wenn dies abgeschlossen ist, kann man mit dem Button „Refactor“ in die nächste Phase wechseln. Dies ist nur möglich, wenn alle Tests bestanden werden. Bei Bedarf gibt es weiterhin auch die Möglichkeit zurück zu RED zu wechseln.

TDD
Systembeschreibung

7. RFCTR

Sind RED und GREEN erledigt und der Code auf einem funktionalen Stand, ist es ein sinnvoller Zeitpunkt, den Code noch einmal zu überdenken. Gibt es eine Möglichkeit, den Code eleganter zu gestalten, ohne dass Funktionalität eingebüßt werden? Gibt es Codedopplungen, die eventuell zusammengefasst werden könnten? In RFCTR werden diese Änderungen implementiert.
Wenn dies abgeschlossen ist, kann man mit dem Button „To RED“ in die nächste RED-Phase wechseln. Dies ist nur möglich, wenn nach wie vor alle Tests bestanden werden. Bei Bedarf gibt es weiterhin auch die Möglichkeit zurück zu GREEN zu wechseln.