

Test

Driven

Development

Trainer

Inhaltsverzeichnis

Einführung.....	2
Was ist TDDT?	2
Start.....	2
Auswahl der Aufgabe	3
Auswahl des Katalogs	4
Start der Aufgabe (Programmierungsumgebung).....	6
Erweiterungen.....	9
Babysteps	9
ATDD	9
Social-Media.....	10
Aufbau eines XML-Aufgaben-Dokuments	11

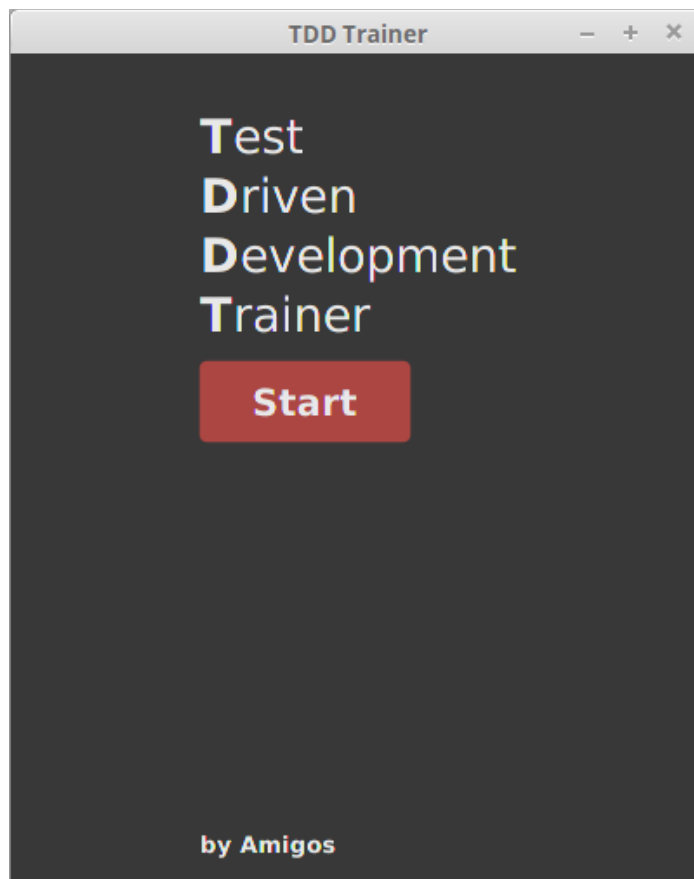
Einführung

Was ist TDDT?

Die Anwendung TDDT (TDD Trainer) soll Ihnen in den ersten Semestern helfen, die Technik der testgetriebenen Entwicklung (test driven development, TDD) zu üben. Bei der testgetriebenen Entwicklung handelt es sich um eine Technik zur Softwareentwicklung, bei der ein Test vor dem zu testenden Code geschrieben wird. Sollten Ihre Kenntnisse über TDD etwas eingerostet sein, ist der Text von Frank Westphal (<http://www.frankwestphal.de/TestgetriebeneEntwicklung.html>) sehr zu empfehlen. TDDT soll den Prozess abbilden und Sie führen.

Start

Beim Start des Programms erscheint folgendes Fenster:

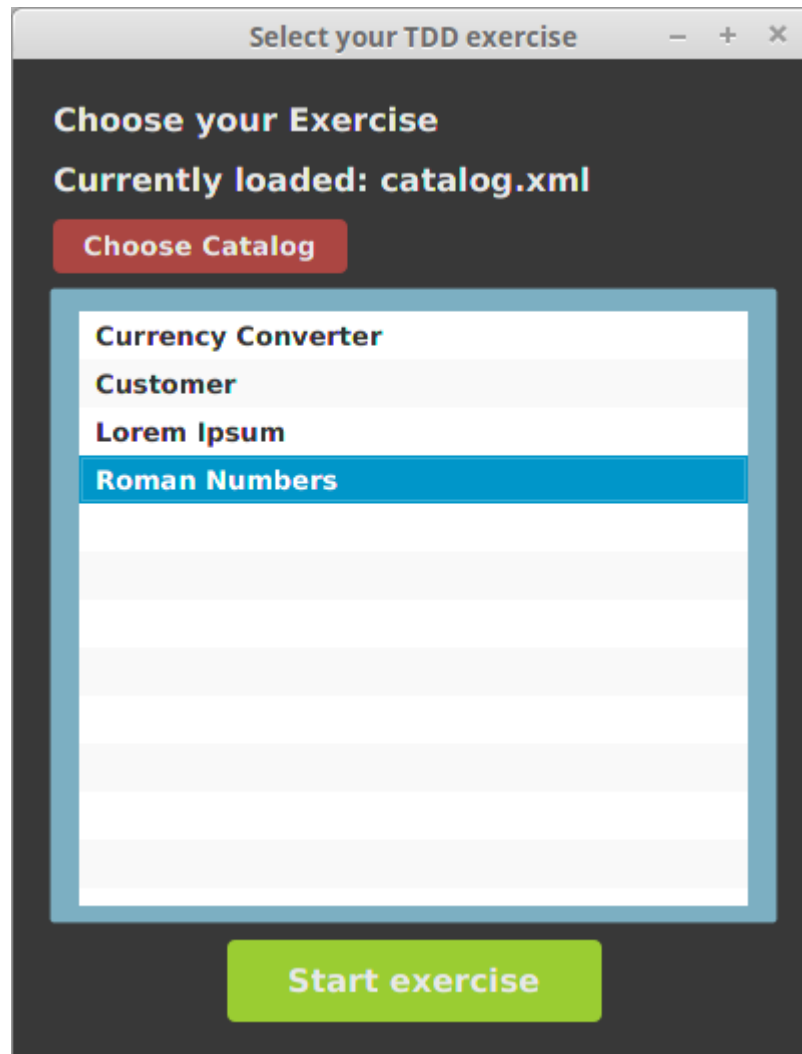


1 Startbildschirm

Klicken Sie auf den „Start“-Button um zur Aufgabenauswahl zu gelangen.

Auswahl der Aufgabe

Sie sollten sich jetzt in diesem Auswahl-Fenster befinden:



2 Aufgabenauswahl

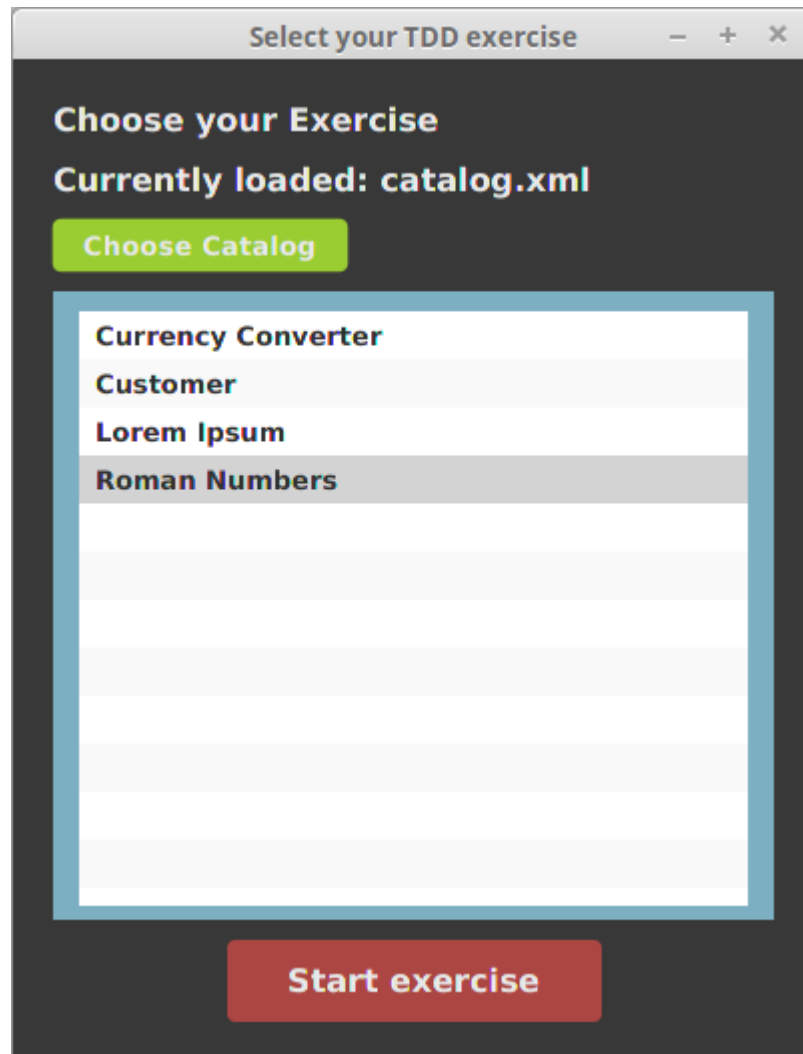
Um eine Aufgabe auszuwählen und zu starten müssen Sie auf die Aufgabe und danach auf den „Start exercise“-Button klicken, alternativ reicht es auch wenn sie einen Doppelklick auf die gewünschte Aufgabe ausführen.

Standardgemäß wird der Katalog mit dem Namen „catalog.xml“ geladen. Sollten Sie mal keine Aufgabe angezeigt bekommen, dann könnte das entweder daran liegen dass der Katalog leer ist oder die Datei nicht existiert. Oder wollen Sie einen anderen Katalog laden?

In dem Fall lesen Sie sich den Punkt „Auswahl des Katalogs“ durch.

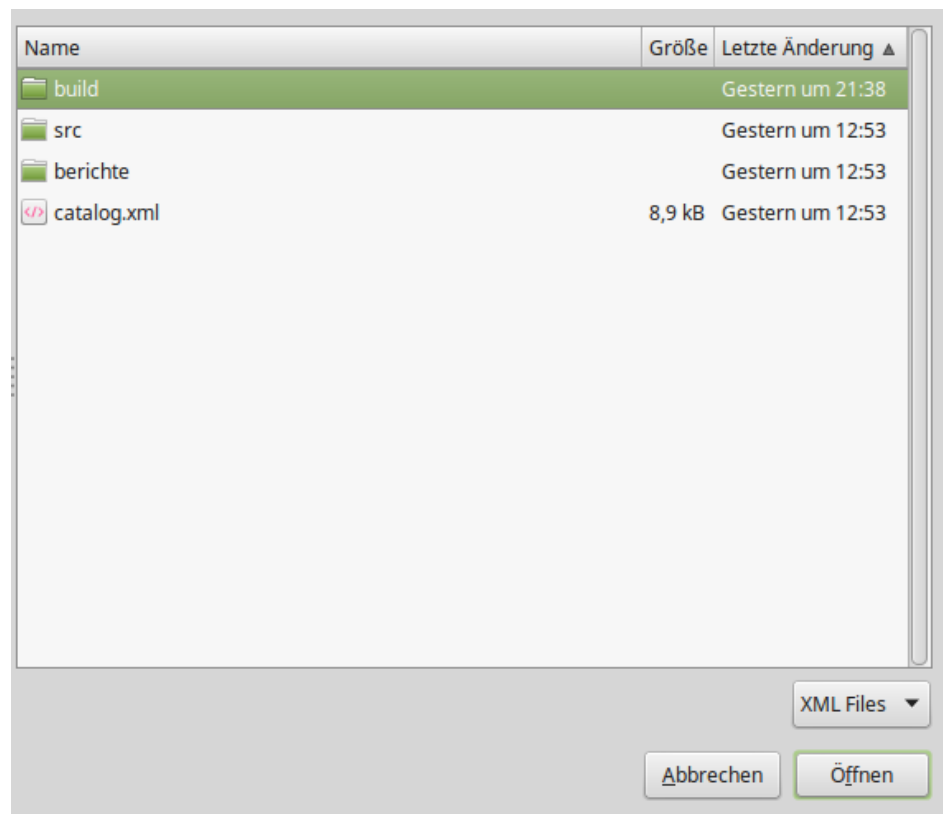
Auswahl des Katalogs

Um einen anderen Katalog auszuwählen müssen Sie auf den „Choose Catalog“-Button klicken.



3 Aufgabenauswahl

Es öffnet sich dann das folgende Fenster:

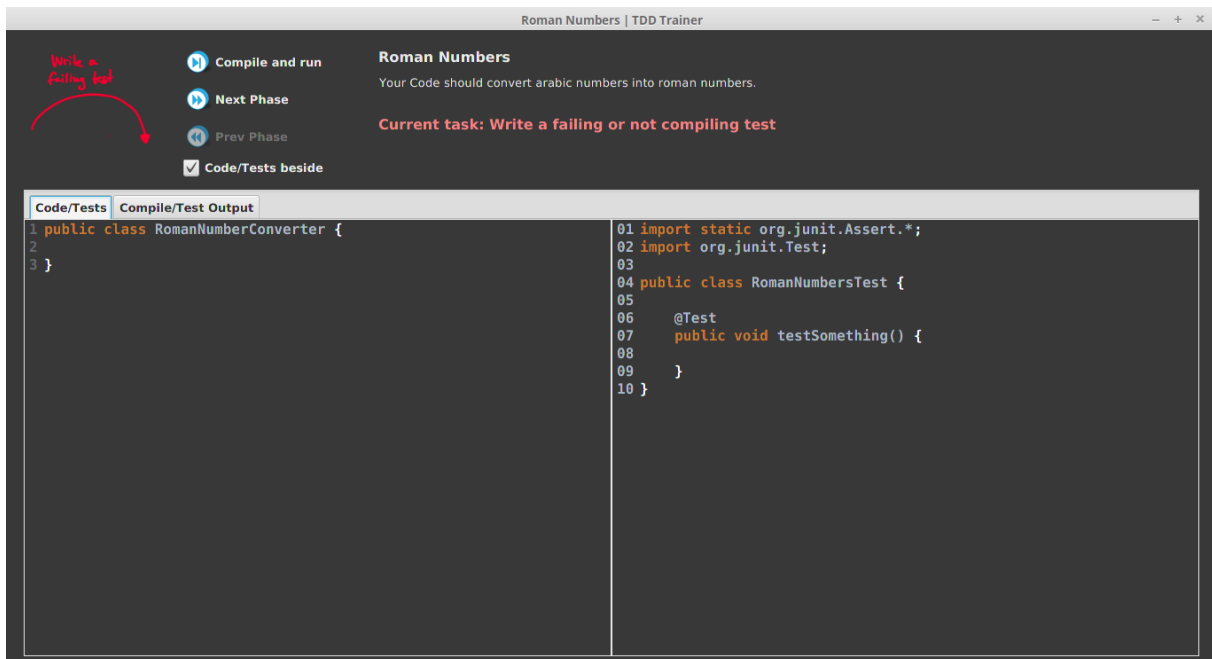


4 XML-Datei Auswahl (Katalog)

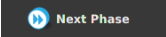
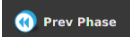
In diesem Fenster können Sie dann die XML-Datei auswählen, die die Aufgaben enthält.

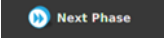
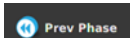
Start der Aufgabe (Programmierungsumgebung)

Nachdem Sie die Aufgabe ausgesucht haben, öffnet sich folgendes Fenster:

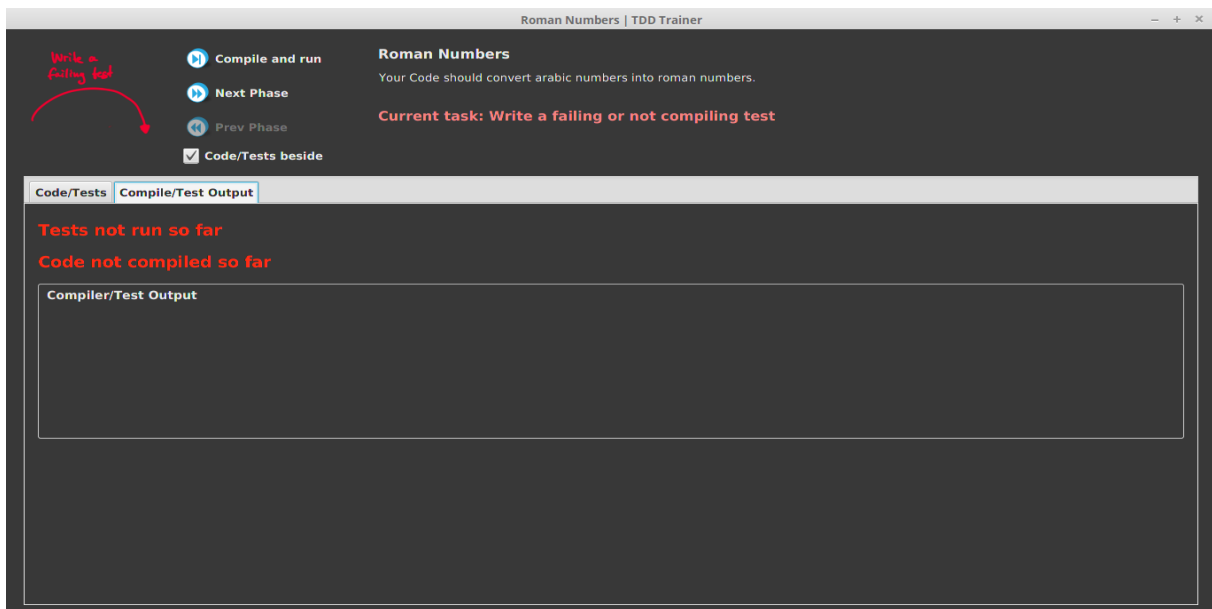


5 Programmierungsumgebung

Auf der linken Seite befindet sich der Code auf der rechten Seite die Tests. Ihre aktuelle Aufgabe steht immer unter der Aufgabenstellung. Wenn Sie meinen dass Sie die Aufgabe erfüllt haben dann drücken Sie auf  wenn es kompilierbar und die Aufgabe erfüllt ist dann leitet Sie das Programm zur nächsten Aufgabe. Wenn Sie einen Schritt zurückgehen möchten dann müssen Sie  drücken.

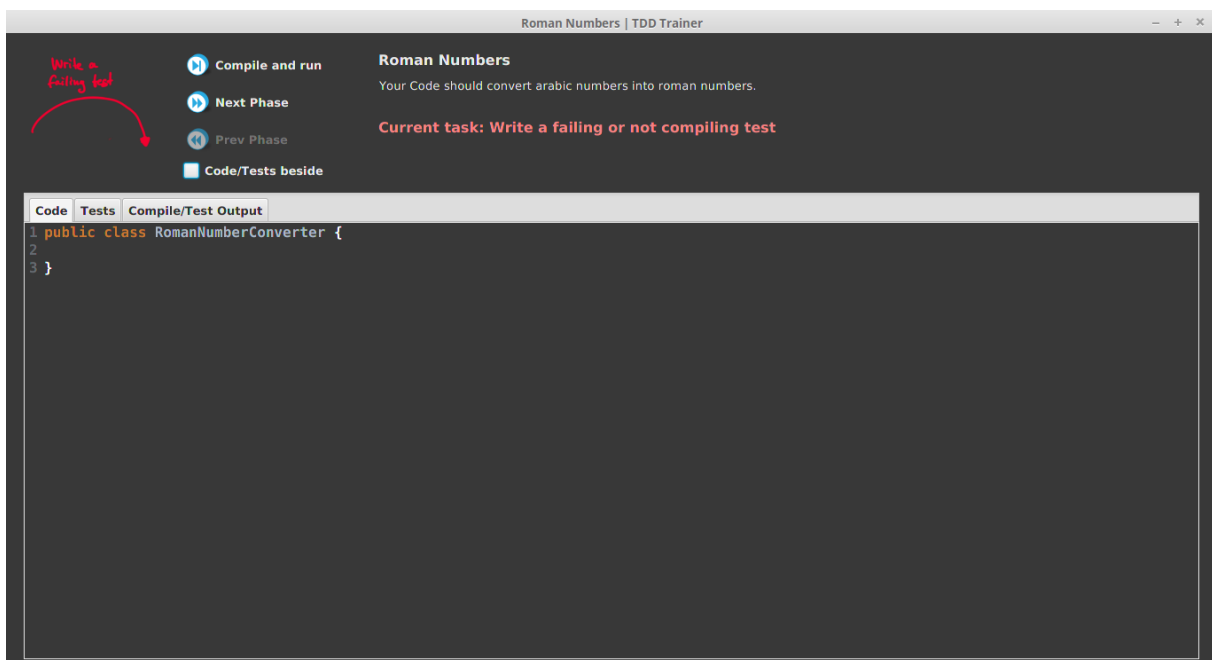
Um die Formatierungen des Codes brauchen Sie sich keine Gedanken zu machen, denn der Code wird nach der Eingabe von STRG + LEERTASTE oder  oder  selbständig formatiert. Außerdem wird nach der Eingabe von „{“ + ENTER direkt das „}“ hinzugefügt.

Um es übersichtlicher zu halten, wird der Output in einem separaten Tab angezeigt:

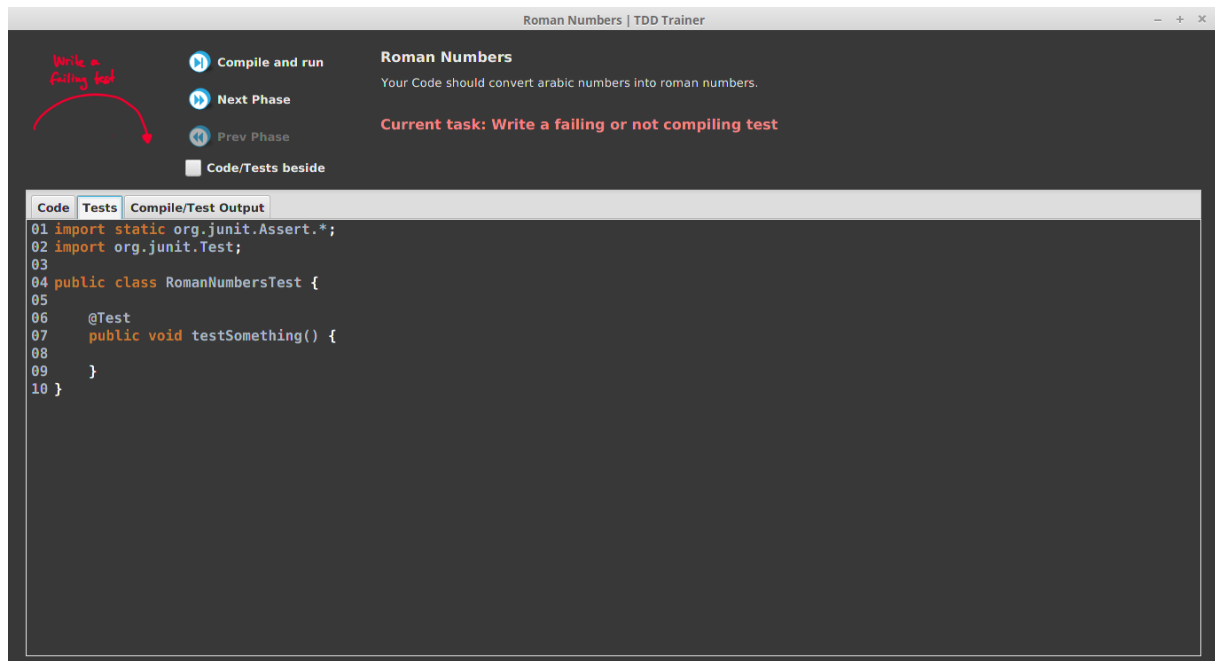


6 Output Bereich

Ihnen gefällt die Aufteilung des Codes und Tests nicht dann nehmen Sie einfach den Haken aus der „Code/Tests beside“-Checkbox raus und dann sieht das ganze so aus:



7 Code separat



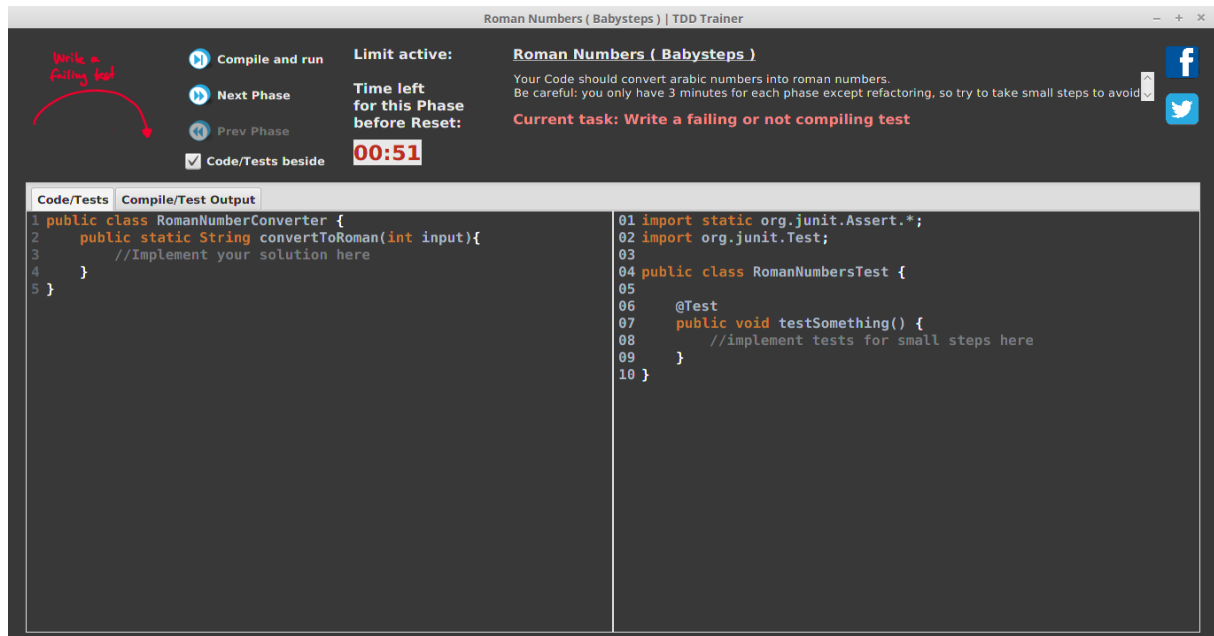
8 Tests separat

Erweiterungen

Babysteps

Wenn für eine Übung Babysteps eingeschaltet sind, wird die Zeit, die Sie in den Phasen **RED** und **GREEN** haben, limitiert. Ist die Zeit abgelaufen, wird der neue Test/Code gelöscht und es wird in die vorangegangene Phase zurückgewechselt. Babysteps haben den Sinn Ihnen die Entwicklung in kleineren Schritten nahezulegen/anzutrainieren. Die Zeit für **REFACTOR** ist nicht limitiert.

Die Zeit wird Ihnen dann wie folgt dargestellt:



9 Babysteps Zeit

ATDD

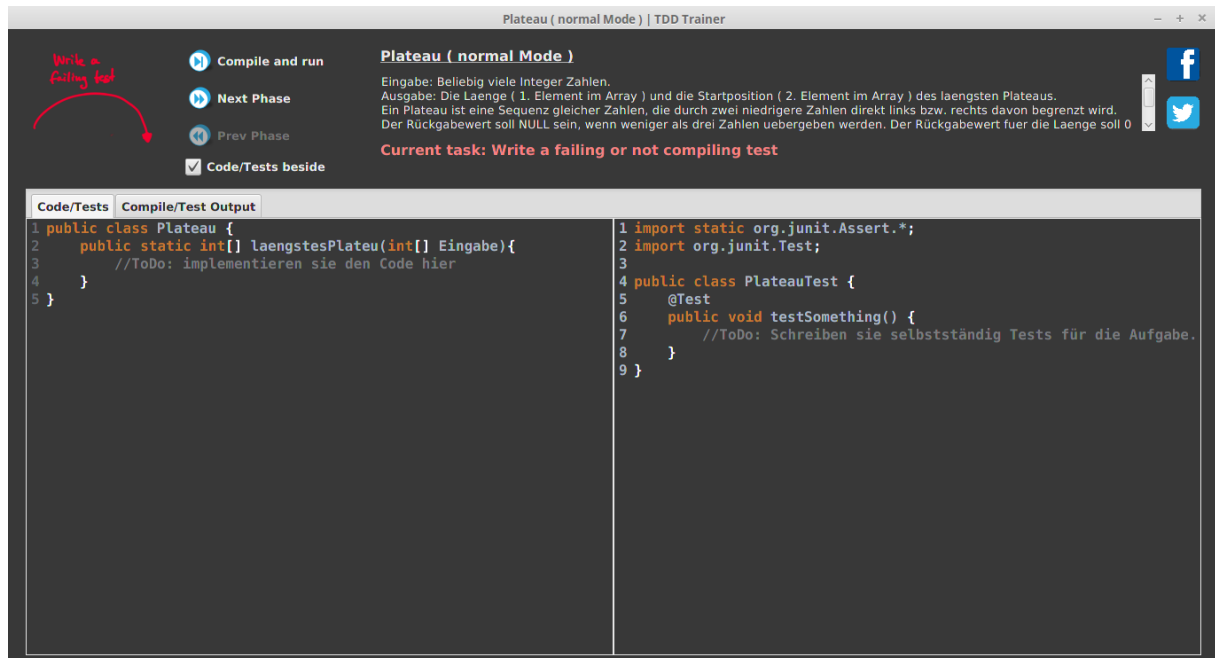
Bei ATDD gibt es zusätzlich zu den Unit-Tests auch noch Akzeptanztests. Ein Akzeptanztest ist ein Test, der **GREEN** wird, wenn ein Feature vollständig implementiert wurde. Der Arbeitsfluss ändert sich dadurch folgendermaßen:

- Als erstes wird ein Akzeptanztest geschrieben. Dieser Test bleibt solange rot, bis ein Feature vollständig implementiert wurde.
- Dann startet der normale TDD Zyklus. Das Feature wird mit Unit-Tests schrittweise entwickelt, bis alle Tests (also auch der Akzeptanztest) grün werden.
- Dann wird der nächste Akzeptanztest geschrieben.

Social-Media

Wenn Sie Ihren Lernerfolg teilen möchten dann können sie das mit den Social-Media Buttons machen. Einfach drauf drücken und Sie werden sofort auf die entsprechende Social-Media Seite weitergeleitet.

Die Social-Media Buttons stehen in oben rechts zur Verfügung:



10 Social-Media Buttons

Aufbau eines XML-Aufgaben-Dokuments

Ein XML-Dokument, das als Aufgabenkatalog fungiert, sollte folgenden Aufbau haben:

```
<exercises>

  <exercise name="Name der Aufgabe">

    <description>

      Hier kann man eine explizite Aufgabenstellung oder Ähnliches angeben

    </description>

    <class name="Beispielklasse">

public class Beispielklasse {

  // Code soll hier implementiert werden

}

    </class>

    <test name="Beispieltestklasse">

public class Beispieltestklasse {

  @Test

  public class beispieltest1() {

    //

  }

}

    </test>

    <options>

      <option name="babysteps" value="120">

      <option name="atdd" value="">

    </options>

  </exercise>

<exercise>

  <!-- Weitere Aufgabe -->

</exercise>

</exercises>
```

Eine Option ist dabei nur dann anzugeben, wenn diese Option aktiviert sein soll. Ist keine Option zu aktivieren, so trägt man `<options />` ein.