

# Test-Driven Development dengan PHPUnit

Putera Kahfi

[www.codejunior.net](http://www.codejunior.net)

email : puterakahfi@gmail.com

---

## Pendahuluan

Quality merupakan aspek penting dalam pengembangan perangkat lunak, software yang tidak berkualitas akan menyebabkan banyak problem, mulai dari sisi pengembangan sampai pada aspek pemeliharaan.

Bagaimana menghasilkan software yang berkualitas bukan perkara yang mudah, karena banyak aspek-aspek yang perlu diperhatikan. Salah satu aspek tersebut adalah aspek pengujian (testing). Pengujian sendiri terbagi menjadi beberapa level ( unit, integration, system) dan beberapa jenis pengujian ( target testing, Objective testing, Acceptance testing, ..) <sup>1</sup>

### 1. Kapan melakukan pengujian

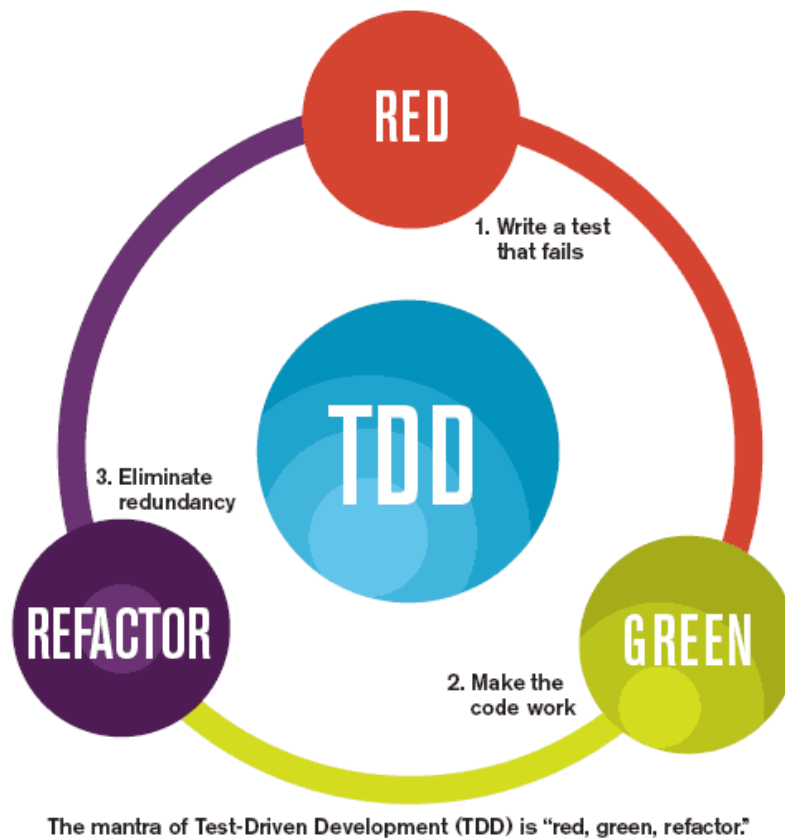
Pengujian bisa dilakukan pada saat awal development(TDD), pada saat proses development di kerjakan, dan pada waktu proses development selesai. Idealnya proses testing dilakukan sejak awal proses development, dengan cara ini maka kebutuhan, problem2x akan lebih cepat diketahui dari pada melakukan testing pada saat proses development selesai.

### 2. Test-Driven Development (TDD)

*Test-driven Development* merupakan salah satu metodologi pengujian *software*. Dimana pada tahap awal yang dibuat adalah skema testingnya terlebih dahulu, artinya kita membuat skema test terlebih dahulu sebelum merancang kode yang *concrete*.

---

<sup>1</sup> Untuk informasi detail bisa diakses di website <http://www.computer.org/portal/web/swebok>



**Gambar 1. Siklus TDD**

Jika kita perhatikan gambar diatas, ada 3 point penting yaitu red, green, *refactor*, apa yang dimaksud dengan red, green dan *refactor* ?

#### **A. Red**

Red mengindikasikan test yang gagal, karena memang pada TDD tes akan di setup ke gagal, untuk mengindikasikan bahwa kita baru membuat sebuah skema testing dan belum membuat implementasi dari testingnya

## B. Green

Green mengindikasikan kita sudah membuat code yang concrete dan sudah melewati berhasil melewati fase testing.

## C. Refactor

Refactor merupakan proses mengubah internal code tanpa mengubah behaviour dari code tersebut, Refactor bertujuan agar code yang di tulis lebih efisien, tidak berulang-ulang dan dinamis. Dengan kata lain refactor membuat code menjadi lebih mudah di gunakan dan di modifikasi<sup>2</sup>

### 2.1 Kelebihan dan kekurangan TDD

#### Kelebihan :

- mengurangi waktu *debugging*<sup>3</sup>
- meningkatkan kualitas *software*
- kemungkinan-kemungkinan kesalahan software lebih cepat terdeteksi

#### Kekurangan

- Untuk scope *software* yang lebih besar, akan memerlukan lebih banyak waktu
- Jika terjadi perubahan business logic, maka *test case* yang saling berhubungan harus di *maintenance* atau di pastikan sesuai dengan *business logic* yang baru

## 3. Instalasi phpunit

---

<sup>2</sup> Silahkan akses <http://refactoring.com/> untuk info lebih detail tentang refactoring

<sup>3</sup> Debugging merupakan proses menelusuri kesalahan program

PHPUnit adalah sebuah programmer-oriented testing framework untuk bahasa pemrograman php, merupakan bagian dari xUnit untuk unit testing framework <sup>4</sup> di buat oleh Sebastian Bergmann<sup>5</sup>, official web untuk PHPUnit adalah <http://phpunit.de/>, versi terbaru adalah phpunit 4.0 untuk versi stable dan phpunit 4.1 untuk versi beta

Untuk instalasi PHPUnit cukup mudah, ada banyak cara yang bisa dilakukan, bisa melalui phar, pear atau composer. Disini hanya akan di jelaskan bagaimana instalasi PHPUnit di linux menggunakan phar<sup>6</sup>.

Tahap-tahap instalasi phar adalah, buka terminal/console ketikkan beberapa baris perintah dibawah ini:

```
→ wget https://phar.phpunit.de/phpunit.phar  
→ chmod +x phpunit.phar  
→ mv phpunit.phar /usr/local/bin/phpunit  
→ phpunit --version
```

PHPUnit 4.0.0 by Sebastian Bergmann.

#### Keterangan:

- *download* phpunit.phar menggunakan wget
- rubah hak akses file agar bisa di eksekusi
- pindahkan ke *directory* /usr/local/bin dengan nama phpunit (agar bisa langsung di eksekusi dari mana saja)
- untuk melihat versi gunakan phpunit --version

<sup>4</sup> Diterjemahkan secara bebas dari <http://phpunit.de>

<sup>5</sup> <http://sebastian-bergmann.de>

<sup>6</sup> Untuk instalasi lainnya silahkan baca di <http://phpunit.de/manual/4.0/en/installation.html>

#### 4. TDD dengan phpunit

untuk mengimplementasikan TDD dengan phpunit maka kita ambil sebuah contoh user, katakanlah ada class yang namanya User yang mempunyai method untuk menambah user, melihat detail user

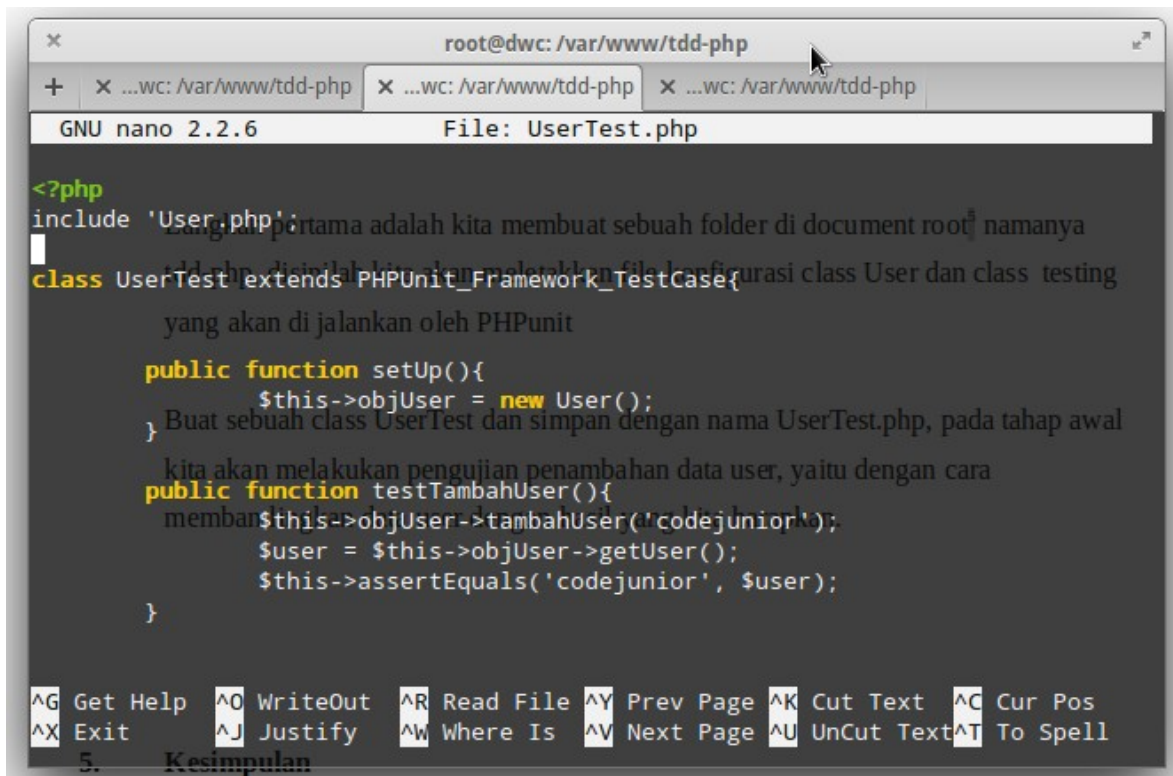
Seperti yang telah di jelaskan sebelumnya TDD dimulai dengan menuliskan kode testing yang gagal terlebih dahulu.

Langkah pertama adalah kita membuat sebuah folder di document root<sup>7</sup> namanya tdd-php, disini kita akan meletakkan file konfigurasi class User dan class testing yang akan di jalankan oleh PHPunit

Buat sebuah class UserTest dan simpan dengan nama UserTest.php, pada tahap awal kita akan melakukan pengujian penambahan data user, yaitu dengan cara membandingkan data user dengan hasil yang kita harapkan.

---

<sup>7</sup> Lokasi document root bisa berbeda-beda, pada contoh ini lokasinya adalah di /var/www/



```
<?php
include 'User.php';

class UserTest extends PHPUnit_Framework_TestCase{
    public function setUp(){
        $this->objUser = new User();
    }
    public function testTambahUser(){
        $this->objUser->tambahUser('codejunior');
        $user = $this->objUser->getUser();
        $this->assertEquals('codejunior', $user);
    }
}
```

**Gambar 2. Kode testing untuk class User**

Langkah selanjutnya adalah membuat sebuah class User dan simpan dengan nama User.php buatlah class yang kosong dan belum ada methodnya

setelah itu tahap selanjutnya adalah melakukan pengujian (**Red**), yaitu untuk menguji apakah testing jalan dan failed

```
dwc@dwc:/var/www/tdd-php$ phpunit UserTest.php
```

PHPUnit 4.0.14 by Sebastian Bergmann.

PHP Fatal error: Call to undefined method User::tambahUser() in /var/www/tdd-php/UserTest.php on line 12

PHP Stack trace:

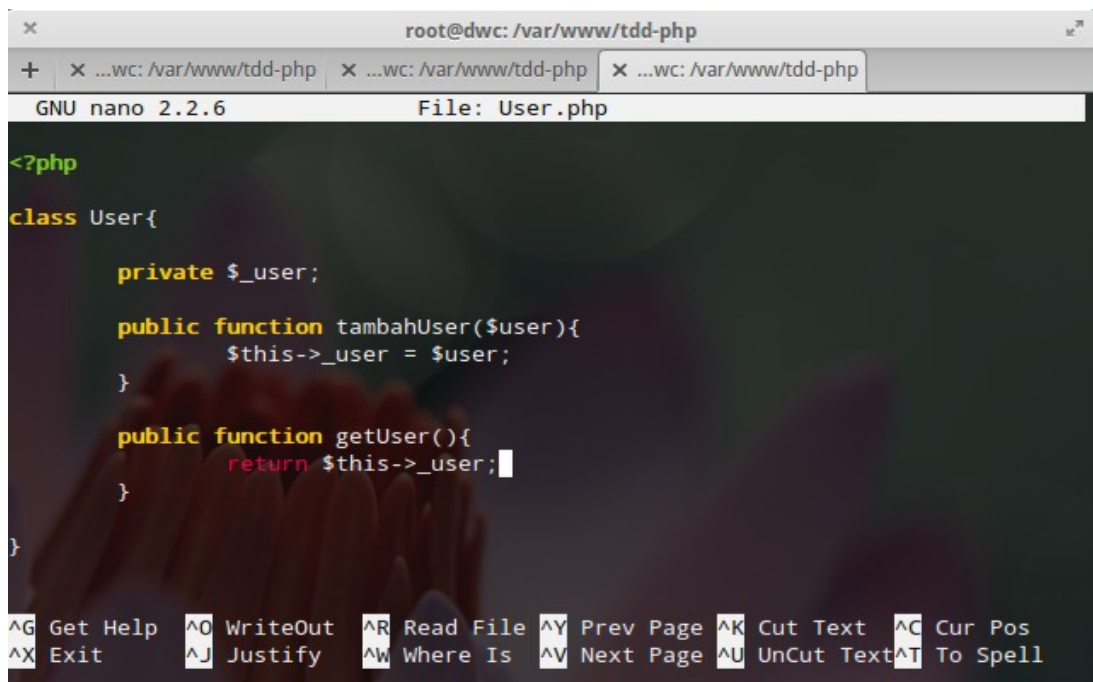
PHP 1. {main}() /usr/local/bin/phpunit:0

PHP 2. PHPUnit\_TextUI\_Command::main() /usr/local/bin/phpunit:583

PHP 3. PHPUnit\_TextUI\_Command->run() phar:///usr/local/bin/phpunit/phpunit/TextUI/Command.php:132

PHP 4. PHPUnit\_TextUI\_TestRunner->doRun() ...

pada saat testing maka dia akan menampilkan pesan PHP Fatal error: Call to undefined method User::tambahUser() in /var/www/tdd-php/UserTest.php on line 12 yang artinya kita belum membuat method addUser, tambahkan dua method untuk menambah user dan mengambil detail user



```
root@dwc: /var/www/tdd-php
GNU nano 2.2.6 File: User.php

<?php
class User{

    private $_user;

    public function tambahUser($user){
        $this->_user = $user;
    }

    public function getUser(){
        return $this->_user;
    }

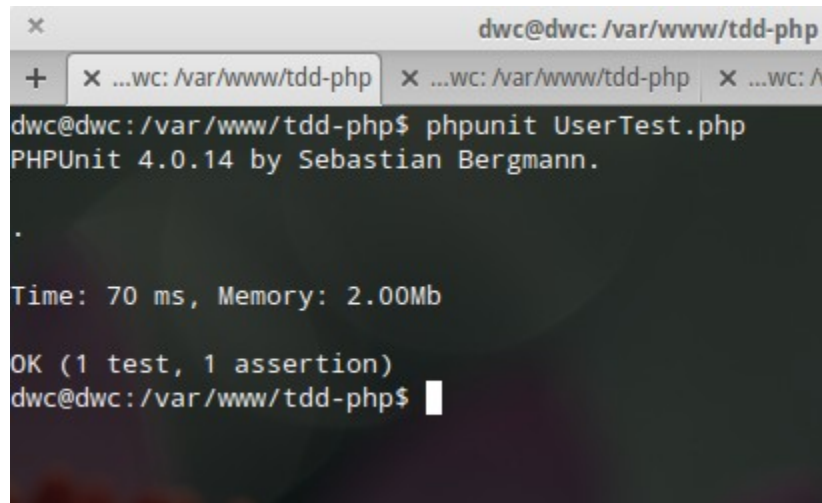
}

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page  ^U UnCut Text ^T To Spell
```

**Gambar 3. Kode untuk class User**

Langkah selanjutnya adalah melakukan pengujian (Green) untuk memastikan testing berhasil , dengan mengetikkan perintah

**# phpunit UserTest.php**



```
dwc@dwc: /var/www/tdd-php
dwc@dwc: /var/www/tdd-php$ phpunit UserTest.php
PHPUnit 4.0.14 by Sebastian Bergmann.

.

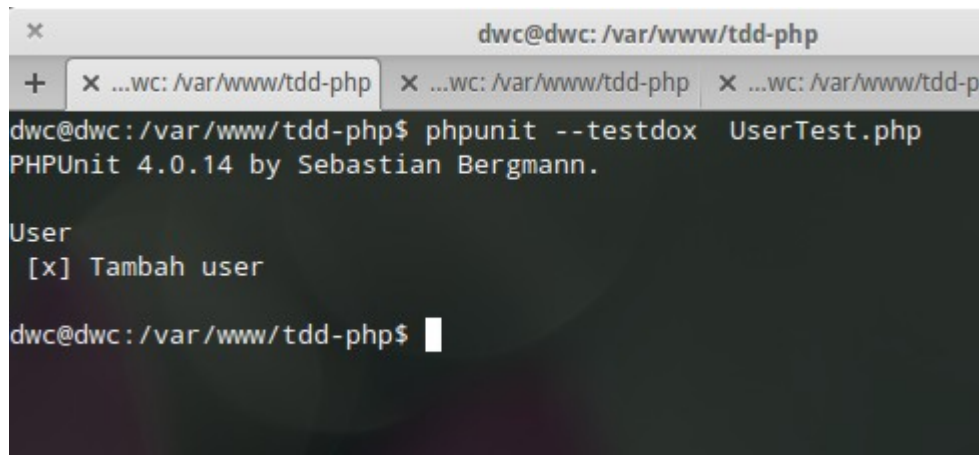
Time: 70 ms, Memory: 2.00Mb

OK (1 test, 1 assertion)
dwc@dwc: /var/www/tdd-php$
```

Gambar 4. Pengujian class User

maka anda akan mendapatkan keterangan bahwa 1 test sudah berhasil. Untuk melihat detail testing yang berhasil gunakan **-testdox** untuk perintahnya menjadi :

**phpunit -testdox UserTest.php**



```
dwc@dwc: /var/www/tdd-php
dwc@dwc: /var/www/tdd-php$ phpunit --testdox UserTest.php
PHPUnit 4.0.14 by Sebastian Bergmann.

User
[x] Tambah user

dwc@dwc: /var/www/tdd-php$
```

Gambar 5. Pengujian dengan testdox



Setelah kita cek ternyata pada saat menambah user data yang di inputkan tidak boleh kosong, maka disini kita akan melakukan refactoring (perubahan code tanpa mengubah behaviournya, data akan tetap bertambah saat data yang di inputkan tidak kosong)

lakukan modifikasi untuk class User menjadi:

```
<?php  
  
class User{  
  
    private $_user;  
    private $_pesan;  
  
    public function tambahUser($user){  
        if($user=='') {  
            $this->setPesan('nama tidak boleh kosong');  
        }else{  
            $this->_user = $user;  
        }  
    }  
  
    public function getUser(){  
        return $this->_user;  
    }  
  
    public function setPesan($pesan){  
        $this->_pesan = $pesan;  
    }  
  
    public function getPesan(){  
        return $this->_pesan;  
    }  
}
```

Kemudian kita buat dua skema testing yaitu testing saat data berhasil dan testing saat data yang di inputkan kosong

```
<?php

include 'User.php';

class UserTest extends PHPUnit_Framework_TestCase{

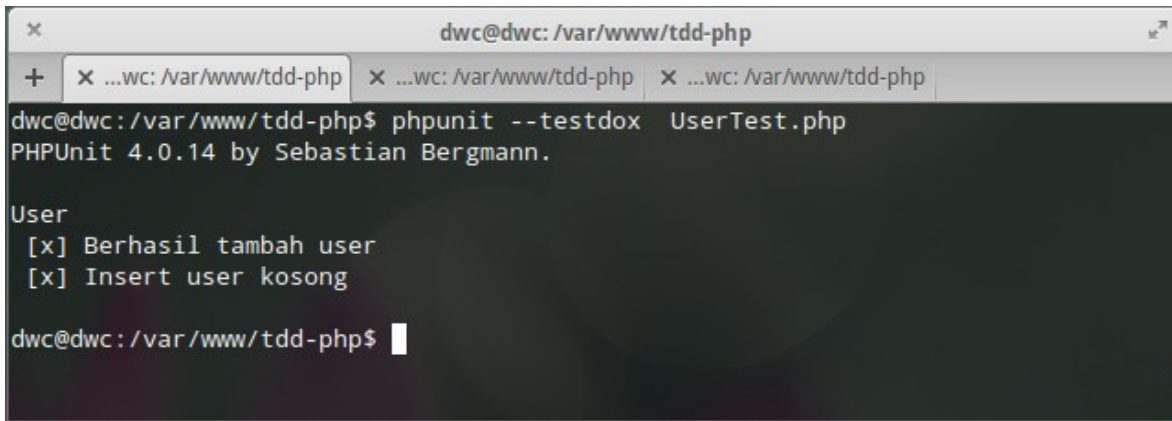
    public function setUp(){
        $this->objUser = new User();
    }

    public function testBerhasilTambahUser(){
        $this->objUser->tambahUser('codejunior');
        $user = $this->objUser->getUser();
        $this->assertEquals('codejunior', $user);
    }

    public function testInsertUserKosong(){
        $this->objUser->tambahUser('');
        $pesan = $this->objUser->getPesan();
        $this->assertEquals('nama tidak boleh kosong', $pesan);
    }

}
```

Kemudian kita lakukan pengujian lagi, maka akan di dapati dua hasil pengujian

A terminal window titled 'dwc@dwc: /var/www/tdd-php' with three tabs. The command 'phpunit --testdox UserTest.php' has been executed. The output shows 'PHPUnit 4.0.14 by Sebastian Bergmann.' followed by a section header 'User'. Below this, two test results are listed: '[x] Berhasil tambah user' and '[x] Insert user kosong'. The prompt 'dwc@dwc: /var/www/tdd-php\$' is visible at the bottom.

```
dwc@dwc: /var/www/tdd-php$ phpunit --testdox UserTest.php
PHPUnit 4.0.14 by Sebastian Bergmann.

User
[x] Berhasil tambah user
[x] Insert user kosong

dwc@dwc: /var/www/tdd-php$
```

**Gambar 6. Pengujian setelah refactoring**

## 5. Kesimpulan

Contoh diatas merupakan contoh sederhana tentan TDD dengan PHPunit, anda bisa mengembangkannya lebih lanjut, seperti jika data sudah ada maka tampilkan pesan data sudah ada, menambahkan email, no telpon, validasi jika email salah Dan lain-lain.

TDD merupakan sebuah metodologi testing yang memungkinkan sebuah agar sebuah software sudah melewati fase testing di awal, dengan cara ini maka problem-problem yang mungkin terjadi pasca produksi bisa di ketahui sejak awal, dan ketika terjadi perubahan maka dapat di deteksi effect dari perubahan tersebut sehingga tidak menimbulkan riffle effect, keretakan *software (fragile)*, *hard code* yang membuat proses *development*, *maintenance software* menjadi rumit dan memakan waktu.