

# PRAKTIKUM 04: UNIT TESTING DENGAN PHPUNIT PART 1

by Nurul Firdaus, S.Kom., M.Inf.

## TUJUAN PEMBELAJARAN

Mahasiswa memahami dasar-dasar pengujian sistem pada komponen sistem

Mahasiswa memahami pengujian komponen sistem

Mahasiswa memahami dan menerapkan pengujian unit/unit testing yang merupakan bagian dari komponen sistem

Mahasiswa memahami dan menerapkan unit testing dengan PHPUnit



## REFERENSI

### Link Penting

<https://phpunit.de/>

<https://phpunit.de/getting-started/phpunit-9.html>

<https://phpunit.readthedocs.io/en/9.3/index.html>

<https://github.com/arseto/php-test-example>

<https://pear.php.net/>

### Buku Referensi PHPUnit

Bergmann, S., 2005. *PHPUnit Pocket Guide: Test-Driven Development in PHP*. " O'Reilly Media, Inc."

### Link materi

<http://www.computesta.com/blog/2010/09/phpunit-pengecekan-coding-yang-lebih-pro/#.X47jWO0xXIU>

## BASIC ILMU PENGETAHUAN

1. PHP Programming Language
2. Object Oriented Programming (OOP)
3. CLI Command Windows / Linux

#### 4. Composer

Composer is an *application-level package manager* for the *PHP Programming Language* that provides a standard format for managing dependencies of PHP software and required *libraries*. (Wikipedia)

## MATERI

Umumnya sebagai programmer PHP, kita mencoba input-input form dengan nilai-nilai tertentu. Kemudian misalnya ternyata hasilnya tidak sesuai maka menggunakan `echo`, `print_r` atau mungkin `var_dump` dan didampingi fungsi `die()` untuk mencari kesalahan. Tetapi ada juga programmer yang levelnya lebih tinggi dari programmer biasa. Mereka akan menulis beberapa baris coding dan kemudian menjalankan skrip kode untuk class/fungsi tertentu tersebut untuk melihat hasilnya sesuai harapan atau tidak. Itulah unit test. Pengetesan pada class atau fungsi API.

Kali ini kami akan memperkenalkan salah satu framework untuk testing unit di PHP yakni PHPUnit. Framework ini diciptakan oleh Sebastian Bergmann, seorang pioner di bidang quality assurance dalam proyek PHP.

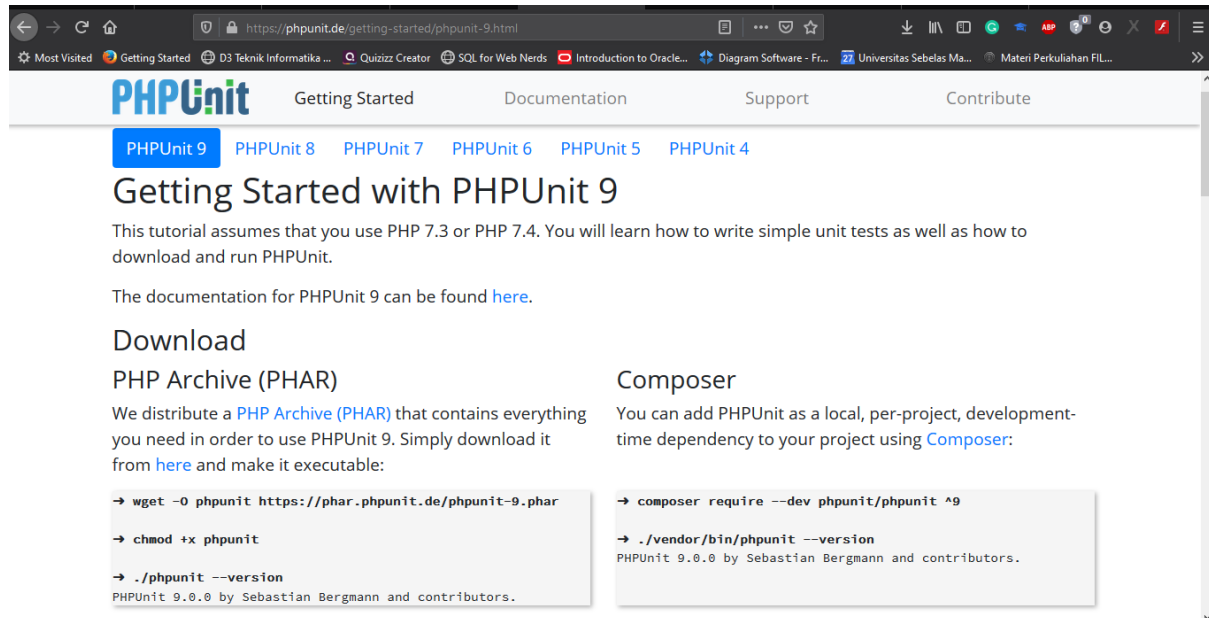
Alur pekerjaan programmer pada umumnya adalah membuat terlebih dahulu class, kemudian testing, setelah itu baru digunakan. Dengan PHPUnit maka akan sedikit berbeda alurnya. Berikut merupakan gambaran alur testing unit menggunakan framework PHPUnit :

1. mendesain class/fungsi
2. membuat sebuah rangkaian nilai yang ingin dites
3. mengimplementasikan class/fungsi
4. menjalankan rangkaian test
5. memperbaiki kesalahan atau error dan kembali ke langkah ke 4

Langkah-langkah ini mungkin kelihatannya memerlukan waktu yang sangat banyak, tetapi sebenarnya tidaklah demikian. Faktanya untuk membuat sebuah rangkaian test dengan PHPUnit cukup beberapa menit dan untuk menjalankan test cukup beberapa detik.

# PRAKTIKUM

1. Buka link berikut <https://phpunit.de/getting-started/phpunit-9.html>



2. Ketikkan perintah berikut di cmd `composer require --dev phpunit/phpunit ^9` dan cek file `composer.json` (*Pastikan isi file `composer.json` kurang lebih seperti dibawah ini*)

```
1 {
2     "name": "lenovo/latihan-php-unit",
3     "description": "Hanya untuk latihan composer dan unit testing dengan php unit",
4     "type": "project",
5     "license": "GPL",
6     "authors": [
7         {
8             "name": "Nurul Firdaus",
9             "email": "nurul.firdaus@staff.uns.ac.id"
10        }
11    ],
12    "autoload": {
13        "classmap": [
14            "src/"
15        ]
16    },
17    "require": {
18        "nesbot/carbon": "^2.41"
19    },
20    "require-dev": {
21        "phpunit/phpunit": "^9.4"
22    }
23 }
```

3. Ketikkan perintah berikut `./vendor/bin/phpunit --version` untuk memastikan versi PHPUnit yang dipakai (***pengguna windows pastikan untuk memakai backslash [ \ ], pengguna linux memakai slash [ / ]***)

```
PHPUnit 9.4.1 by Sebastian Bergmann and contributors.
D:/#26w62f6r eanujj 3050/Bahar V]9r/bn9KtjKw Bf/J9tj9n-bpb-nuit> ./vendor/bin/phpunit --version
```

4. Buatlah dua folder (folder **src** dan **tests**) di dalam folder **latihan-php-unit**

2020 > Bahan Ajar > Praktikum RPL > latihan-php-unit

Name	Date modified	Type	Size
src	10/20/2020 4:58 PM	File folder	
tests	10/20/2020 4:58 PM	File folder	
vendor	10/15/2020 8:56 AM	File folder	
composer.json	10/20/2020 4:21 PM	JSON File	1 KB
composer.lock	10/15/2020 10:28 AM	LOCK File	82 KB
tes_carbon.php	10/15/2020 9:07 AM	PHP File	1 KB

5. Buat file **Email.php** didalam folder **src**

```
Email.php
1 <?php declare(strict_types=1);
2 final class Email
3 {
4     private $email;
5
6     private function __construct(string $email)
7     {
8         $this->ensureIsValidEmail($email);
9
10        $this->email = $email;
11    }
12
13    public static function fromString(string $email): self
14    {
15        return new self($email);
16    }
17
18    public function __toString(): string
19    {
20        return $this->email;
21    }
22
23    private function ensureIsValidEmail(string $email): void
24    {
25        if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
26            throw new InvalidArgumentException(
27                sprintf(
28                    '"%s" is not a valid email address',
29                    $email
30                )
31            );
32        }
33    }
34 }
```

6. Buat file **EmailTest.php** didalam folder **tests**

```
EmailTest.php
1 <?php declare(strict_types=1);
2 use PHPUnit\Framework\TestCase;
3
4 final class EmailTest extends TestCase
5 {
6     public function testCanBeCreatedFromValidEmailAddress(): void
7     {
8         $this->assertInstanceOf(
9             Email::class,
10            Email::fromString('user@example.com')
11        );
12    }
13
14    public function testCannotBeCreatedFromInvalidEmailAddress(): void
15    {
16        $this->expectException(InvalidArgumentException::class);
17
18        Email::fromString('invalid');
19    }
20
21    public function testCanBeUsedAsString(): void
22    {
23        $this->assertEquals(
24            'user@example.com',
25            Email::fromString('user@example.com')
26        );
27    }
28 }
```

7. Ketikkan perintah **composer dump-autoload** untuk men-generate files **autoload** dan perintah **./vendor/bin/phpunit tests** untuk melakukan pengujian/testing

```
D:\#Semester Ganjil 2020\Bahan Ajar\Praktikum RPL\latihan-php-unit> composer dump-autoload
Generating autoload files
Generated autoload files

D:\#Semester Ganjil 2020\Bahan Ajar\Praktikum RPL\latihan-php-unit> ./vendor/bin/phpunit tests
PHPUnit 9.4.1 by Sebastian Bergmann and contributors.

...                                     3 / 3 (100%)

Time: 00:00.040, Memory: 4.00 MB

OK (3 tests, 3 assertions)
```

8. Ketikkan perintah **./vendor/bin/phpunit** untuk melihat alternatif output yang didasarkan pada gagasan bahwa nama pengujian dapat digunakan untuk mendokumentasikan perilaku yang diverifikasi oleh pengujian.

```
D:\#Semester Ganjil 2020\Bahan Ajar\Praktikum RPL\latihan-php-unit> ./vendor/bin/phpunit --testdox tests
PHPUnit 9.4.1 by Sebastian Bergmann and contributors.

Email
  ✔ Can be created from valid email address
  ✔ Cannot be created from invalid email address
  ✔ Can be used as string

Time: 00:00.078, Memory: 4.00 MB

OK (3 tests, 3 assertions)
```

9. Buatlah file **phpunit.xml** seperti gambar berikut :

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <phpunit backupGlobals="false"
3           backupStaticAttributes="false"
4           bootstrap="vendor/autoload.php"
5           colors="true"
6           convertErrorsToExceptions="true"
7           convertNoticesToExceptions="true"
8           convertWarningsToExceptions="true"
9           processIsolation="false"
10          stopOnFailure="false"
11          verbose="true">
12    <testsuites>
13      <testsuite name="Unit Test">
14        <directory suffix="Test.php">./tests/</directory>
15      </testsuite>
16    </testsuites>
17  </phpunit>
```

## TUGAS

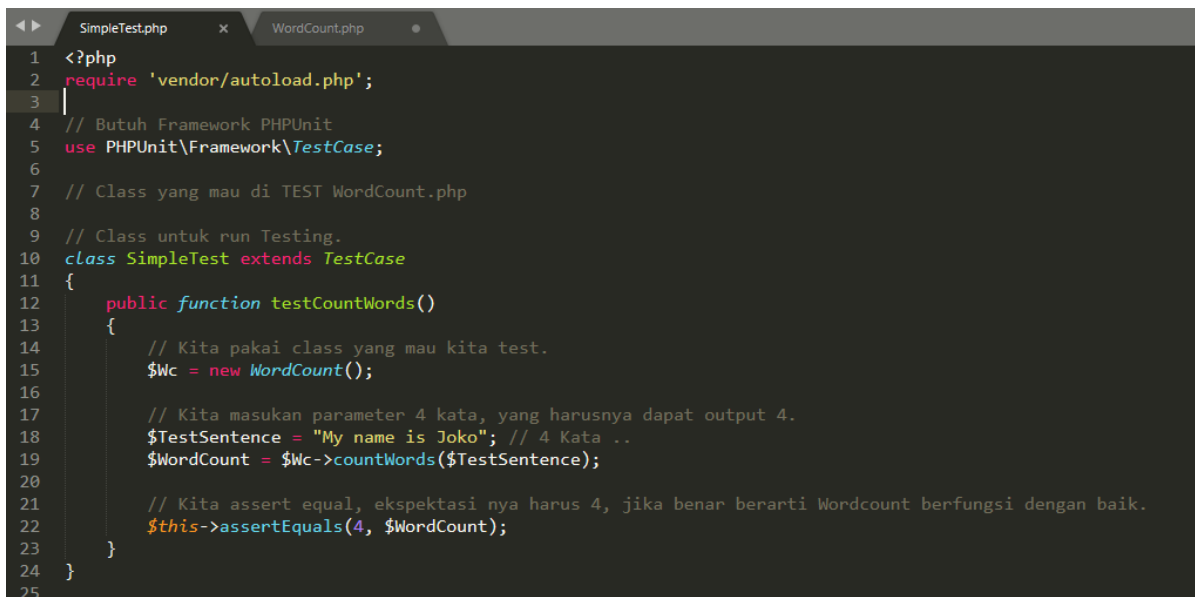
1. Jelaskan maksud unit testing dengan menggunakan PHPUnit pada file **Email.php** dan **EmailTest.php** di praktikum diatas!
2. Implementasikan dan jelaskan example/contoh pada sub-bab **Test Dependencies** dan **Data providers**!

<https://phpunit.readthedocs.io/en/9.3/writing-tests-for-phpunit.html#test-dependencies>

<https://phpunit.readthedocs.io/en/9.3/writing-tests-for-phpunit.html#data-providers>

3. Buatlah file **src\WordCount.php** dan **tests\SimpleTest.php**

```
SimpleTest.php x WordCount.php
1  <?php
2  // File : WordCount.php
3  class WordCount
4  {
5      public function countWords($sentence)
6      {
7          return count(explode(" ", $sentence));
8      }
9  }
10 ?>
```



```

1 <?php
2 require 'vendor/autoload.php';
3
4 // Butuh Framework PHPUnit
5 use PHPUnit\Framework\TestCase;
6
7 // Class yang mau di TEST WordCount.php
8
9 // Class untuk run Testing.
10 class SimpleTest extends TestCase
11 {
12     public function testCountWords()
13     {
14         // Kita pakai class yang mau kita test.
15         $wc = new WordCount();
16
17         // Kita masukan parameter 4 kata, yang harusnya dapat output 4.
18         $testSentence = "My name is Joko"; // 4 Kata ..
19         $wordCount = $wc->countWords($testSentence);
20
21         // Kita assert equal, ekspektasi nya harus 4, jika benar berarti Wordcount berfungsi dengan baik.
22         $this->assertEquals(4, $wordCount);
23     }
24 }
25

```

- a. Test dengan menggunakan perintah `vendor/bin/phpunit tests\SimpleTest.php` .Jelaskan hasilnya!
- b. Ubahlah `$testSentence` dengan nama  
Cth : `$testSentence = "My name is Nurul Firdaus"`  
Kemudian test menggunakan perintah `vendor/bin/phpunit tests\SimpleTest.php` .  
Jelaskan hasilnya!
- c. Ubahlah `$this->assertEquals(5, $wordCount);`  
Kemudian test menggunakan perintah `vendor/bin/phpunit tests\SimpleTest.php` .  
Jelaskan hasilnya!
- d. Jelaskan maksud unit testing dengan menggunakan PHPUnit pada file **WordCount.php** dan **SimpleTest.php**