Procédure :

Mise en place de tests unitaires pour projets PHP

Rennes, le 26/08/2014

FICHE DE CONTROLE ET DIFFUSION DU DOCUMENT

Versions		
Version	**Date**	**Sujet**
1.1	26/08/2014	Mise à jour : Processus d'installation de PHPUnit
1.0	21/08/2014	Initialisation du document

Diffusion			
Nom	**Société**	**Nombre d'exemplaires**	**Mode**
Etienne Méléard	Renater	1	Mail

NIVEAU 1			NIVEAU 2			NIVEAU 3		
Emetteur (Rédacteur)			Vérificateur			Approbateur		
Nom	**Date**	**Visa**	**Nom**	**Date**	**Visa**	**Nom**	**Date**	**Visa**
Damien Bruchet	**26/08/2014**	**DABR**						

I. Table des matières

1) Situation, contexte

Ce document fournit des préconisations quand à la mise en place de tests unitaires pour projets PHP.

2) Prérequis

1. PHP

La version utilisée est la version 5.4.24 (PHP 5 minimum).

2. PHPUnit et PEAR

PHPUnit est un framework de test unitaires open source.

PEAR (PHP Extension and Application Repository) est une collection d'API PHP.

→ Processus d'installation de phpunit

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

Cf. : https://github.com/sebastianbergmann/phpunit/blob/master/README.md

// DEPRECATED (mais fonctionnel) :

- Installation PEAR sur MAC OS :

Cf. http://coolestguidesontheplanet.com/installing-pear-osx-10-9-mavericks-osx10-810-7/

- Installation PHPUnit sur MAC OS :

Cf. http://imzank.com/2012/08/how-to-install-phpunit-with-mamp/

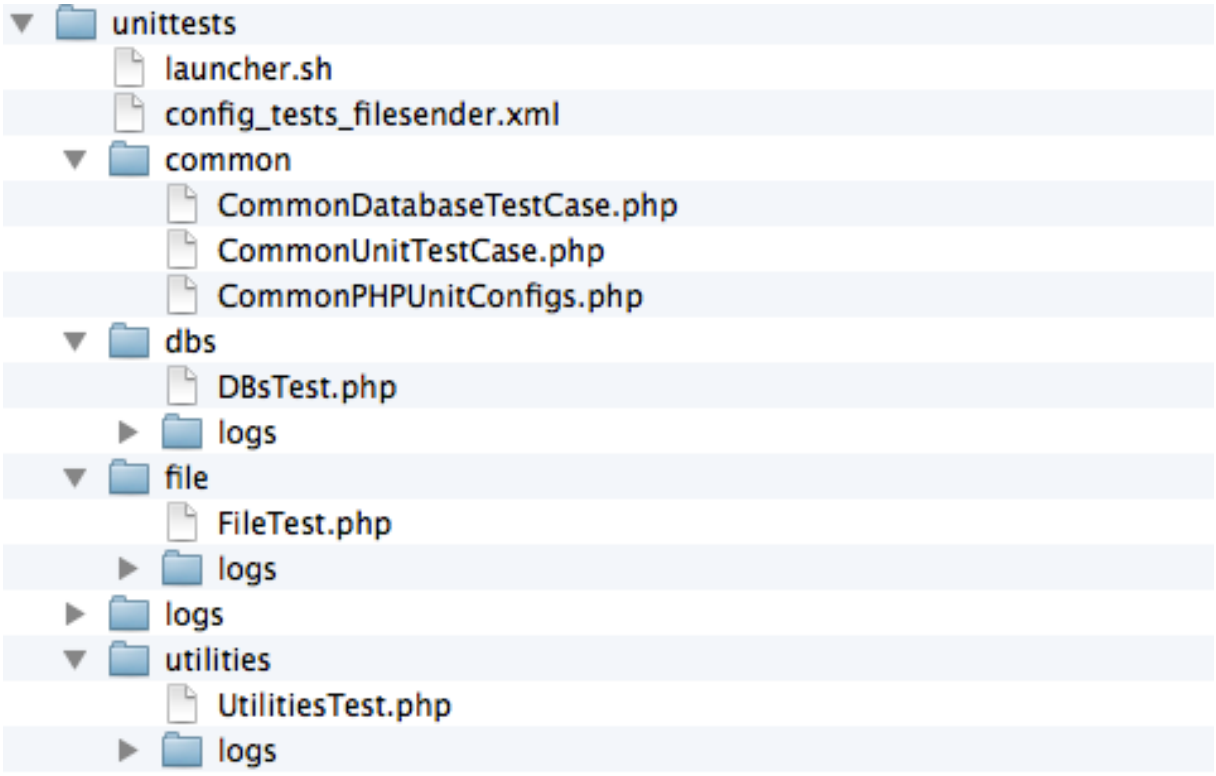
NB : la version de MAC OS utilisé est la 10.9.4.

3) Mise en place des test unitaires

a) Arborescence des fichiers

La mise en place de tests unitaires tels que préconisée dans ce document se doit de suivre l'arborescence définie ci-dessous.

A la racine du projet se trouve un dossier nommé « unittests » contenant :



NB : ce document prend pour exemple le projet filesender-2.0 (http://filesender.org).

- **launcher.sh**

Ce script permet d'automatiser le lancement des tests unitaires (cf. annexe).

→ Cf. partie *d) Exécution* page XX.

- **config_tests_filesender.xml**

Ce fichier contient la configuration optionnelle pour l'exécution des tests unitaires avec PHPUnit.

→ Cf. partie b*) Configuration* page XX.

- **common > CommonDatabaseTestCase.php**

Ce fichier contient 3 éléments :

- ***DBOperations*** : correspond aux différentes opérations CRUD en base de données
- ***DBErrors*** : correspond aux différents codes erreurs liés aux opérations via une basse de données
- ***CommonDatabaseTestCase*** : classe abstraites qui se doit être étendus par chacun des fichiers contenant les tests unitaires liés aux opérations CRUD.

Il contient quatre fonctions abstraites à redéfinir dans les classes filles. Elles correspondent aux opérations CRUD (Create, Read, Update, Delete).

Il contient également deux fonctions formalisant les messages en sortie des tests unitaires.

Il implémente l'interface PHPUnit_Framework_TestCase, nécessaire à l'exécution des tests unitaires avec PHPUnit.

- **common > CommonUnitTestCase.php**

Ce fichier contient une classe abstraite qui se doit d'être étendu par chacun des fichiers contenant les tests unitaires.

Il contient deux fonctions formalisant les messages en sortie des tests unitaires.

Il implémente l'interface PHPUnit_Framework_TestCase, nécessaire à l'exécution des tests unitaires avec PHPUnit.

- **common > CommonPHPUnitConfigs**

Ce fichier contient l'ensemble des configurations, inclusions de fichier, et autres éléments nécessaires au bon déroulement de l'exécution des tests unitaires.

→ Cf. partie b*) Configuration* page XX.

- **example > logs**

Ce dossier contient les fichiers de log et de résultats émis par PHPUnit après l'exécution des tests unitaires.

Les dossiers « ***dbs*** », « ***file*** » et « ***utilities*** » fournis en annexe contiennent des exemples de tests unitaires pour le projet filesender-2.0.

b) Configuration

- Configuration de l'environnement

Cette configuration se doit être écrite dans le fichier ***common > CommonPHPUnitConfigs.php***.

Exemple :

```
date_default_timezone_set("Europe/Paris");

require_once('../../classes/autoload.php');
require_once('../../classes/_includes.php');
```

- Fichier de configuration PHPUnit (xml)

Ce fichier de configuration est un fichier xml interprété par PHPUnit.

Exemple :

```
<phpunit>
 <testsuites>
  <testsuite name="ExampleTestSuite" >
   <file>example/ExampleTest.php</file>
   <file>example/DBsTest.php</file>
  </testsuite>
  <testsuite name="ExampleBDDTestSuite" >
    <file>example/DBsTest.php</file>
  </testsuite>

 </testsuites>
</phpunit>
```

L'exécution de la commande suivante lancera les tests unitaires présent dans les fichiers *example/ExampleTest.php* et *example/DbsTest.php* :

```
./launcher.sh –c configuration.xml
```

c) Développement du test

Ci-dessous un exemple permettant de tester succinctement que la date passé en paramètre est au bon format après l'appel de la fonction Utilities::formatDate($timestamp).

CF. Projet filesender-2.0.

```
require_once dirname(__FILE__).'/../common/CommonUnitTestCase.php';

/**
 * @backupGlobals disabled
 */
class UtilitiesTest extends CommonUnitTestCase {
  protected  $objectUtilities;

  /**
   * Init variables, first function called
   */
  protected function setUp() {
    echo "@ ".date("Y-m-d H:i:s")."\n\n";
  }



  public function testFormatDate(){
    //Test
    $timestamp = strtotime("2014-09-04");

    try{
      $this->assertTrue(Utilities::formatDate($timestamp) == "04-09-2014" ) ;

      $this->displayInfo(get_class(), __FUNCTION__ ,'');
    }catch (PHPUnit_Framework_AssertionFailedError $ex){
      $this->displayError(get_class(), __FUNCTION__, $ex->getMessage());
      throw new PHPUnit_Framework_AssertionFailedError();
    }

    return true;
  }
}
```

d) Exécution

Le fichier « launcher.sh » peut être utilisé pour automatiser l'exécution des tests unitaires.

Usage :

```
./launcher.sh testFolder [testfile.php] [[-c [testconfig.xml]] [-ts [testSuiteName]]]
```

Il est possible d'exécuter tous les tests d'un dossier en exécutant la commande :

```
./launcher.sh dossierTest
```

Pour spécifier un fichier de test particulier, il faut exécuter la commande :

```
./launcher.sh dossierTest fichierTest.php
```

Il est possible d'utiliser un fichier de configuration xml (cf. documentation PHPUnit).

Pour se faire, exécutez la commande suivante :

```
./launcher.sh dossierTest –c configuration.xml
```

Pour exécuter un testsuite défini dans le fichier de configuration xml particulier, exécutez la commande suivante :

```
./launcher.sh dossierTest –c configuration.xml -ts testSuiteName
```