



Unit testing with PHPUnit

Duration: 1 day - 7 hours

Price : 590€ / person

Code: PHPUNITC1

GOALS

In one day, you will learn how to install phpunit, write and execute your first unit tests suites thanks to the phpunit framework.

PRE-REQUISITES: Good knowledge of PHP.

PUBLIC: Developers

FULL PROGRAM

INTRODUCTION TO UNIT TEST

- What are unit tests ?
- Pros and cons
- The FIRST rule

INTRODUCTION TO PHP UNIT

- Introduction to PHP Unit
- Open Source PHP projects use PHP Unit
- IDEs integration
- PHPUnit installation from PEAR
- **Lab :** Install PHPUnit with PEAR

WRITE YOUR FIRST ASSERTIONS

- Introduction to the Command Line Interface
- Write a first tests suite
- Run the test suite from the command line tool
- **Lab:** Write and run a PHPUnit tests suite from the command line

INITIALIZE AND RESET A TEST

- Setup a tests suite with the setup() method
- Reset a testing environment with the tearDown() method
- Introduction to fixtures
- **Lab:** setup and reset a tests suite with setup() and tearDown() methods

TESTING EXCEPTIONS

- Write unit tests for methods, which can handle exceptions
- Testing exception with the setExpectedException() assertion method
- Testing exception by using the @expectedException annotation
- Testing exception thanks to a try { } catch code block
- **Lab:** implement unit tests for a method that can throw an exception

TEST DRIVEN DEVELOPMENT

- Understanding the test driven development (TDD) methodology
- Main advantages of the TDD approach
- Implement code thanks to a test driven development approach
- **Lab:** fix and document a bug thanks to a TDD approach

DATA PROVIDERS AND FIXTURES

- Understanding the data providers and fixtures concepts
- Create a new data provider method to run a test with several values
- **Lab:** implement a data provider method to test a method's edge cases

ADVANCED TESTING PRACTICES

- Testing exceptions
- Using data providers to create fixtures
- Assertions to test a class and its attributes
- Assertions to test Boolean values
- Assertions to test string values
- Assertions to test XML values

CODE COVERAGE ANALYSIS

- Understanding the code coverage concept
- Check XDebug extension is installed on the web server
- Generate a complete code coverage report from the command line interface
- Analyzing and understanding generated reports and statistics
- Understanding and using the special `@covers` annotation
- Ignore code blocks from coverage process thanks to `@codeCoverageIgnore(Start|End)` annotation
- **Lab:** Implement new methods and their tests to validate the code coverage rate

USING MOCK OBJECTS

- Discovering and creating mock objects
- Test data model coupled to mock objects
- Check code coverage rate thanks to advanced unit tests
- **Lab:** implement mock objects to simulate real objects behaviors

GENERATING REPORTS

- Export results as XML
- Export results as JSON
- Export results as TAP
- Check code coverage rate thanks to advanced unit tests
- **Lab :** export results as XML, JSON and TAP

MONITORING THE TESTS SUITE WITH JENKINS

- Discovering the continuous integration practice
- Installing and running a continuous integration tool
- **Lab :** install and running Jenkins CI