Unit Testing with PHPUnit

# What is it?

Unit Testing or Test Driven Development, is the process of writing automatable tests for your code which allows you to enter data and compare the actual result with an expected result. If your code does not generate the appropriate output then the test fails. Once the test is created it can be run multiple times, allowing you to, Write Once – Test Many.

# Assertion

Unit Tests are run through formulating an *Assertion*. An assertion is where you assert that the input given will provide a specific output, in the context of the function or class that it is being tested with. For example:

function sum($a, $b)   
{

return $a + $b;

}

$this->assertEquals(2, sum(1, 1));

Once the test is run, if the assertion is proven to be correct then the test passes. Otherwise the test fails and where/why the test failed is highlighted – In this example the test assets that the sum of 1+1 equals 2, which is correct. If the *sum()* function did not correctly contain the addition of $a+$b then the assertion would have been proved false, and flags would raise where the code failed. From these failure flags the developer is able to edit the function or class to work correctly.

The most used assertions are:

* assertTrue() : This verifies that a condition is true
* assertFalse() : This verifies that a condition is false
* assertEquals() : This verifies that expected and actual values are equal, the same way as the PHP comparison operator ==
* assertSame() : This is similar to assertEquals() , but it checks whether values are identical, the same way as the === operator
* assertNull() : This verifies that value is null
* assertEmpty() : This verifies that value is empty, but it uses the PHP function empty() , which means empty can be false, null, ‘’, array()

# Importance

Unit tests allow us the confidence in knowing that our code works and can be relied upon in a commercial environment to not fail in its developed task. Breaking code into small manageable units reduces the risk of introducing bugs when pieces of code interact with each other. It is also, a great investment available if you need to expand and modify your application during updates or for refactoring purposes as it naturally lends itself to Object-Oriented code paradigms that aid development efficiency.