PHPUnit Cheat Sheet

Version 0.1 based on PHPUnit 3.6

ASSERTIONS

Basics

assertEmpty
assertEquals
assertFalse
assertGreaterThan
assertGreaterThanOrEqual
assertInternalType
assertLessThan
assertLessThanOrEqual
assertNotEmpty
assertNotEquals
assertNotInternalType
assertNotNull
assertNotSame
assertNull
assertSame
assertTrue

Objects

assertInstanceOf
assertObjectHasAttribute
assertObjectNotHasAttribute
assertNotInstanceOf

Classes

assertClassHasAttribute
assertClassHasStaticAttribute
assertClassNotHasAttribute
assertClassNotHasStaticAttribute

Classes and Objects

assertAttributeContains
assertAttributeContainsOnly
assertAttributeEmpty
assertAttributeEquals
assertAttributeGreaterThan
assertAttributeGreaterThanOrEqual
assertAttributeInstanceOf
assertAttributeInternalType
assertAttributeLessThan
assertAttributeLessThanOrEqual
assertAttributeNotContains
assertAttributeNotContainsOnly
assertAttributeNotEmpty
assertAttributeNotEquals
assertAttributeNotInstanceOf
assertAttributeNotInternalType
assertAttributeNotSame
assertAttributeSame

Arrays and Traversable Objects

assertArrayHasKey
assertArrayNotHasKey
assertContains
assertContainsOnly
assertCount
assertNotContains
assertNotContainsOnly
assertNotCount

Strings

assertNotRegExp
assertRegExp
assertSelectEquals
assertSelectCount
assertSelectRegExp
assertStringEndsNotWith
assertStringEndsWith
assertStringEqualsFile
assertStringMatchesFormat
assertStringMatchesFormatFile
assertStringNotEqualsFile
assertStringNotMatchesFormat
assertStringNotMatchesFormatFile
assertStringStartsNotWith
assertStringStartsWith

XML and HTML

assertEqualXMLStructure
assertNotTag
assertTag
assertXmlFileEqualsXmlFile
assertXmlFileNotEqualsXmlFile
assertXmlStringEqualsXmlFile
assertXmlStringEqualsXmlString
assertXmlStringNotEqualsXmlFile
assertXmlStringNotEqualsXmlString

Files

assertFileEquals
assertFileExists
assertFileNotEquals
assertFileNotExists

Others

assertThat

MATCHERS for EXPECTS method

any
at
atLeastOnce
exactly
never
once

CONSTRAINTS for WITH method

Commutation

logicalAnd
logicalNot
logicalOr
logicalXor

Basics

anything
arrayHasKey
contains
equalTo
greaterThan
greaterThanOrEqual
identicalTo
isFalse
isNull
isTrue
lessThan
lessThanOrEqual

Classes & Objects

attribute
attributeEqualTo
classHasAttribute
classHasStaticAttribute
hasAttribute
isInstanceOf
isType

String

matchesRegularExpression
stringContains
stringEndsWith
stringStartsWith

Others

fileExists

RETURNS for WILL method

onConsecutiveCalls
returnArgument
returnCallback
returnValue
throwException

"getMock()" method

```
/**
 * @return PHPUnit_Framework_
 *         MockObject_MockObject
 */
public function getMock(
    $originalClassName,
    $methods = array(),
    array $arguments = array(),
    $mockClassName = '',
    $callOriginalConstructor = TRUE,
    $callOriginalClone = TRUE,
    $callAutoload = TRUE
)
```

Template methods

```
pub static fn setUpBeforeClass()
pro fn setUp()
pro fn assertPreConditions()
pro fn assertPostConditions()
pro fn tearDown()
pub static fn tearDownAfterClass()
pro fn onNotSuccessfulTest()
```

Utilities

```
$this->fail()
$this->markTestIncomplete()
$this->markTestSkipped()
$this->expectOutputString()
$this->setExpectedException()
```

Annotations

```
/**
 * @expectedException <exceptionName>
 * @dataProvider <methodName>
 * @depends <methodName>
 */

<exceptionName> can be:
PHPUnit_Framework_Error
PHPUnit_Framework_Warning
```

Example with a Mock object

```
<?php
require_once 'SomeClass.php';

class StubTest extends PHPUnit_Framework_TestCase
{
    public function testStub()
    {
        // Create a stub for the SomeClass class
        $stub = $this->getMock('SomeClass', array(‘doSomething’) );

        // Configure the stub
        $stub->expects( $this->once() )
             ->method( 'doSomething' )
             ->with( $this->equalTo('bar') )
             ->will( $this->returnValue('foo') );

        // Calling $stub->doSomething() will now return 'foo'
        $this->assertEquals('foo', $stub->doSomething('bar'));
    }
}
```

Testing Exceptions

```
/**
 * @expectedException MyException
 */
public function testThrowsAnException()
{
    $stub = $this->getMock(‘stdClass’);
    $stub->expects( $this->any() )
         ->method(‘push’)
         ->will( $this->throwException( new MyException ) );
    $stub->push(42);
}
```