# VDFUNIT – Getting started

## Getting VDFUnit

You can get the latest version (in fact, *all* versions) from <https://bitbucket.org/olaeld/vdfunit>

Use Mercurial to pull from Bitbucket (highly recommended), or use the menu item ‘get source’ on Bitbucket to get VDFUnit.

## Running VDFUnit

From the “VDFUnit Example workspace” folder, start the “VDFUnitDemo” workspace appropriate for your VDF version (sws files are provided starting with version 15.1, but creating one for earlier versions of VDF should not be a problem).

Compile and run the project “VDFUnit TestRunner”, and you’re good to go!

If all goes well, the program should start, the test fixtures should be executed, and the result box should become green. You can run the tests again by pressing the “Run tests” button.

## So I ran the program. Now what?

Now it’s your turn to write tests. Let me walk you through some VDFUnit essentials.

# VDFUnit essentials

## Test fixtures

Tests belong in a *test fixture*. Each test fixture should test a set of related functionality. Ideally, one test fixture should test the functionality of one class.

**cMyClass**

**cMyClassTests**

***One* class => *One* test fixture**

## Tests

A test fixture contains a set of tests.

For each public method of your class, you would normally write at least one test.

## Assertions

A test should either fail or pass. Normally, this is done by assertion. If an assert fails while running a test, this means that the test has failed (the test result box will turn red).

### Assert methods

Procedure Assert Boolean bCondition String sAssertMessage  
Procedure AssertFalse Boolean bCondition String sAssertMessage  
Procedure AssertIAreEqual Integer Expected Integer Actual String sAssertMessage  
Procedure AssertNAreEqual Number Expected Number Actual String sAssertMessage  
Procedure AssertSAreEqual String Expected String Actual String sAssertMessage  
Procedure AssertDTAreEqual DateTime Expected DateTime Actual String sAssertMessage

## An example test fixture

Filename: VDFPatterns\AppSrc\cArrayIteratorIntegerTests.pkg

Use VDFUnit.pkg

Use cArrayIteratorInteger.pkg

Object cArrayIteratorIntegerTests is a cTestFixture

Procedure Setup

Object oIterator is a cArrayIteratorInteger

End\_Object

Integer[] iArray

Integer iCount

For iCount from 0 to 19

Move iCount to iArray[iCount]

Loop

Send AttachArray to oIterator iArray

End\_Procedure

Procedure TearDown

Send Destroy of oIterator

End\_Procedure

Object IteratorReset\_IteratorMoveNext\_CurrentArrayValue\_Returns0 is a cTest

Procedure Test

Send IteratorReset to oIterator

Boolean bFoundFirstElement

Get IteratorMoveNext of oIterator to bFoundFirstElement

Send Assert bFoundFirstElement "First element of iterator not found"

Send AssertIAreEqual 0 (CurrentArrayValue(oIterator(Self)))

End\_Procedure

End\_Object

Object IteratorMoveNext\_20times\_ReturnsFalse is a cTest

Procedure Test

Send IteratorReset to oIterator

Integer iCount

Boolean bFound

For iCount from 1 to 20

Get IteratorMoveNext of oIterator to bFound

Send Assert bFound ("Iterator should contain 20 elements, but found only" ;

\* (iCount - 1))

If (not(bFound)) Procedure\_Return

Loop

Get IteratorMoveNext of oIterator to bFound

Send AssertFalse bFound "Iterator should not contain more than 20 elements"

End\_Procedure

End\_Object

End\_Object

## Running tests

To run the tests, include the test fixture file in oVDFUnitTests.pkg.

VDFUnit detects its presence, and the TestRunner program will execute its tests.