MXUnit

LINKS

www.mxunit.org
www.mxunit.org/update

Icon	Action	Shortcut
	Help	F1
	Find Tests	CTRL-F
	Reload Tests	F5
	Run selected Tests	Enter; r
	Run failures	CTRL-Enter; CTRL-R
	Expand/Collapse tests	=
	Select All Tests in Tree	CTRL-A
	Show errors/failures only	f
	Select previously run test	1 through 9; the last 9 tests can be loaded from the history dropdown by hitting its corresponding number on the keyboard
Test(s) selected in test tree		
	View test output in browser	b; F8
File selected in the "Trace" table		
	Open file at error line	Enter; double-click

ADVANCED METHODS

- **expectedExceptions(types)**
- **debug(data)**
- **makePublic(obj, method, [newMethod])**
- **injectMethod(receiver,giver,functionName,functionNameInReceiver)**
- **injectProperty(obj,name,value,scope=variables)**

TEST CASE

```
component extends="mxunit.framework.TestCase"{

   function setup(){}
   function testXXX(){}
   function teardown(){}

}
```

TIPS & TRICKS

- You can right click on any directory, select "Run MXUnit Tests", and it'll find your Test components and run them, ignoring all the others
- If you have your test case open in an Eclipse editor, you can right click in the file, select "Run MXUnit Test", and run the tests in the plugin
- You can find "slow" tests easily by setting the Timeout preference to "1". You can easily change the timeout preference from the little white triangle on the right side of the view
- If you maximize the plugin view, the view will automatically put the "Tag Context" view on the right (i.e. it'll shift from vertical to horizontal)
- debug() and the "b" keyboard shortcut are perhaps your best friends
- If you need to copy the contents of a failure or error message, you can select the failed/errored test in the tree, then select a row in the TagContext panel, right click, and choose "Copy Exception" or "Copy Tag Context"

ASSERTIONS

- **assert(condition,message)**
- **assertTrue(condition,message)**
- **assertFalse(condition,message)**
- **assertEquals(expected,actual,message)**
- **assertNotSame(obj1,obj2,message)**
- **assertSame(obj1,obj2,message)**
- **assertXPath(xpath,data,text,message)**
- **assertIsTypeOf(obj,type)**
- **assertIsXMLDoc(obj)**
- **assertIsArray(obj)**
- **assertIsStruct(obj)**
- **assertIsQuery(obj)**
- **assertIsDefined(obj)**
- **assertIsEmpty(obj)**
- **assertIsEmptyArray(obj)**
- **assertIsEmptyQuery(obj)**
- **assertIsEmptyStruct(obj)**

FAILURES

- **fail(message)**
- **failNotEquals(value,value2,message)**

ANNOTATIONS

- **expectedExceptions** = List of Exception Types

BASE TEST CLASS

- **mxunit.framework.TestCase**

OUTPUT FORMATS

- html
- xml
- query
- junitxml
- array

URL RUNNER

- MyTest.cfc?method=runTestRemote

URL PARAMS

- debug:boolean
- testMethod:string
- output:outputFormat

TESTCASE INTERCEPTORS

- **setup()**
  Called at the beginning of each test method
- **teardown()**
  Called at the end of each test method

REMOTE FACADE

```
component extends="mxunit.framework.RemoteFacade"{

  function actOnTestCase(any TestCase){}

}
```

TEST SUITE

```
testSuite =
createObject("component","mxunit.framework.TestSuite").TestSuite();
testSuite.addAll("mxunit.samples.MyComponentTest");
testSuite.add("mxunit.samples.MyOtherComponentTest","aTestFunctionThatDoes
NotBeginWithTest,anotherTestFunctionThatDoesNotBeginWithTest");
results = testSuite.run();
writeOutput(results.getResultsOutput('html'));
```

DIRECTORY RUNNER

```
runner = createObject("component","mxunit.runner.DirectoryTestSuite");
results = runner.run(directory, recurse, excludes, componentPath);
writeOutput(results.getResultsOutput('html'));
```

ANT TASK

```
<taskdef name="mxunittask" classname="org.mxunit.ant.MXUnitAntTask"
classpath="${mxunit.jar}" />
<mxunittask server="localhost"
            port="8500"
            defaultrunner="/mxunit/runner/HttpAntRunner.cfc"
            outputdir="${output.dir}"
            verbose="true" haltonerror="true"> ...
```