

JUnit

See [Unit Testing](#) for more info on how to utilize the methods of the JUnit class.

Method Summary

void	assertEquals (expected, actual) Asserts that two values are equal.
void	assertEquals (message, expected, actual) Asserts that two values are equal.
void	assertFalse (boolean_condition) Asserts that a condition is false.
void	assertFalse (message, boolean_condition) Asserts that a condition is false.
void	assertFloatEquals (expectedFloat, actualFloat, tolerance) Asserts that two floating point values are equal to within a given tolerance.
void	assertFloatEquals (message, expectedFloat, actualFloat, tolerance) Asserts that two floating point values are equal to within a given tolerance.
void	assertMatches (regularExpression, actualString) Asserts that a regular expression matches a string.
void	assertMatches (message, regularExpression, actualString) Asserts that a regular expression matches a string.
void	assertNotNull (object) Asserts that an object is not null.
void	assertNotNull (message, object) Asserts that an object is not null.
void	assertNotSame (notExpected, actual) Asserts that two values are not the same.
void	assertNotSame (message, notExpected, actual) Asserts that two values are not the same.
void	assertNotUndefined (definedObject) Asserts that an object is not undefined.
void	assertNotUndefined (message, definedObject) Asserts that an object is not undefined.
void	assertNull (nullValue) Asserts that an object is null.
void	assertNull (message, nullValue) Asserts that an object is null.
void	assertSame (expected, actual) Asserts that two values are the same.
void	assertSame (message, expected, actual) Asserts that two values are the same.
void	assertTrue (boolean_condition) Asserts that a condition is true.
void	assertTrue (message, boolean_condition) Asserts that a condition is true.
void	assertUndefined (undefinedValue) Asserts that an object is undefined.
void	assertUndefined (message, undefinedValue) Asserts that an object is undefined.
void	fail (message) Fails a test.
void	fail (message, instanceOfCallStack) Fails a test.
void	fail (message, instanceOfCallStack, userMessage) Fails a test.

Method Details

assertEquals

void **assertEquals** (expected, actual)
Asserts that two values are equal. AssertionError is thrown if the actual value does not match the regular expression.

Parameters

{Object} expected - the expected value.
{Object} actual - the actual value.

Returns

void

Sample

```
// Asserts that two values are equal. AssertionError is thrown if the actual value does not match the
regular expression.
jsunit.assertEquals("Solution name test", "someSolution", application.getSolutionName());
jsunit.assertEquals("Simple math test", 2, 1 + 1);
```

assertEqualsvoid **assertEquals** (message, expected, actual)

Asserts that two values are equal. AssertionError is thrown if the actual value does not match the regular expression.

Parameters

{String} message - The test description/message.
{Object} expected - the expected value.
{Object} actual - the actual value.

Returns

void

Sample

```
// Asserts that two values are equal. AssertionError is thrown if the actual value does not match the
regular expression.
jsunit.assertEquals("Solution name test", "someSolution", application.getSolutionName());
jsunit.assertEquals("Simple math test", 2, 1 + 1);
```

assertFalsevoid **assertFalse** (boolean_condition)

Asserts that a condition is false. AssertionError is thrown if the evaluation was not false.

Parameters

{Boolean} boolean_condition - the actual value.

Returns

void

Sample

```
// Asserts that a condition is false. AssertionError is thrown if the evaluation was not false.
jsunit.assertFalse("False test", application.isLastPrintPreviewPrinted());
```

assertFalsevoid **assertFalse** (message, boolean_condition)

Asserts that a condition is false. AssertionError is thrown if the evaluation was not false.

Parameters

{String} message - The test description/message.
{Boolean} boolean_condition - the actual value.

Returns

void

Sample

```
// Asserts that a condition is false. AssertionError is thrown if the evaluation was not false.
jsunit.assertFalse("False test", application.isLastPrintPreviewPrinted());
```

assertFloatEqualsvoid **assertFloatEquals** (expectedFloat, actualFloat, tolerance)

Asserts that two floating point values are equal to within a given tolerance. AssertionError is thrown if the expected value is not within the tolerance of the actual one.

Parameters

`{Number}` expectedFloat - the expected value.
`{Number}` actualFloat - the actual value.
`{Number}` tolerance - tolerance when comparing.

Returns

void

Sample

```
// Asserts that two floating point values are equal to within a given tolerance. AssertionError is
thrown if the expected value is not within the tolerance of the actual one.
jsunit.assertFloatEquals("Float equals test", 3.12, 3.121, 0.0015);
```

assertFloatEquals

void **assertFloatEquals** (message, expectedFloat, actualFloat, tolerance)

Asserts that two floating point values are equal to within a given tolerance. AssertionError is thrown if the expected value is not within the tolerance of the actual one.

Parameters

`{String}` message - The test description/message.
`{Number}` expectedFloat - the expected value.
`{Number}` actualFloat - the actual value.
`{Number}` tolerance - tolerance when comparing.

Returns

void

Sample

```
// Asserts that two floating point values are equal to within a given tolerance. AssertionError is
thrown if the expected value is not within the tolerance of the actual one.
jsunit.assertFloatEquals("Float equals test", 3.12, 3.121, 0.0015);
```

assertMatches

void **assertMatches** (regularExpression, actualString)

Asserts that a regular expression matches a string. AssertionError is thrown if the expected value is not the actual one.

Parameters

`{Object}` regularExpression - the regular expression used for matching.
`{String}` actualString - the actual value to be matched.

Returns

void

Sample

```
// Asserts that a regular expression matches a string. AssertionError is thrown if the expected value
is not the actual one.
jsunit.assertMatches(new RegExp("gr(a|e)y"), "gray");
```

assertMatches

void **assertMatches** (message, regularExpression, actualString)

Asserts that a regular expression matches a string. AssertionError is thrown if the expected value is not the actual one.

Parameters

`{String}` message - The test description/message.
`{Object}` regularExpression - the regular expression used for matching.
`{String}` actualString - the actual value to be matched.

Returns

void

Sample

```
// Asserts that a regular expression matches a string. AssertionError is thrown if the expected value
is not the actual one.
jsunit.assertMatches("Match test", new RegExp("gr(a|e)y"), "gray");
```

assertNotNull

void **assertNotNull** (object)

Asserts that an object is not null. AssertionError is thrown if the object is not null.

Parameters

{Object} object - the actual value.

Returns

void

Sample

```
// Asserts that an object is not null. AssertionError is thrown if the object is not null.
var a; // this is undefined, not null
jsunit.assertNotNull(a);
```

assertNotNull

void **assertNotNull** (message, object)

Asserts that an object is not null. AssertionError is thrown if the object is not null.

Parameters

{String} message - The test description/message.

{Object} object - the actual value.

Returns

void

Sample

```
// Asserts that an object is not null. AssertionError is thrown if the object is not null.
var a; // this is undefined, not null
jsunit.assertNotNull("Not null test", a);
```

assertNotSame

void **assertNotSame** (notExpected, actual)

Asserts that two values are not the same. AssertionError is thrown if the expected value is the actual one.

Parameters

{Object} notExpected - the value that is not expected.

{Object} actual - the actual value.

Returns

void

Sample

```
// Asserts that two values are not the same. AssertionError is thrown if the expected value is the
actual one.
var a = new Date(1990, 1, 1);
var b = new Date(1990, 1, 1);
jsunit.assertNotSame(a, b);
jsunit.assertEquals("But equals", a, b);
```

assertNotSame

void **assertNotSame** (message, notExpected, actual)

Asserts that two values are not the same. AssertionError is thrown if the expected value is the actual one.

Parameters

{String} message - The test description/message.

{Object} notExpected - the value that is not expected.

{Object} actual - the actual value.

Returns

void

Sample

```
// Asserts that two values are not the same. AssertionError is thrown if the expected value is the actual one.
var a = new Date(1990, 1, 1);
var b = new Date(1990, 1, 1);
jsunit.assertNotSame("Not same test", a, b);
jsunit.assertEquals("But equals", a, b);
```

assertNotUndefined

void **assertNotUndefined** (definedObject)

Asserts that an object is not undefined. AssertionError is thrown if the object is undefined.

Parameters

{Object} definedObject - the actual value.

Returns

void

Sample

```
// Asserts that an object is not undefined. AssertionError is thrown if the object is undefined.
var a = 0;
jsunit.assertNotUndefined(a);
```

assertNotUndefined

void **assertNotUndefined** (message, definedObject)

Asserts that an object is not undefined. AssertionError is thrown if the object is undefined.

Parameters

{String} message - The test description/message.

{Object} definedObject - the actual value.

Returns

void

Sample

```
// Asserts that an object is not undefined. AssertionError is thrown if the object is undefined.
var a = 0;
jsunit.assertNotUndefined("Not undefined test", a);
```

assertNull

void **assertNull** (nullValue)

Asserts that an object is null. AssertionError is thrown if the object is not null.

Parameters

{Object} nullValue - the actual value.

Returns

void

Sample

```
// Asserts that an object is null. AssertionError is thrown if the object is not null.
jsunit.assertNull("Null test", null);
```

assertNull

void **assertNull** (message, nullValue)

Asserts that an object is null. AssertionError is thrown if the object is not null.

Parameters

{String} message - The test description/message.

{Object} nullValue - the actual value.

Returns

void

Sample

```
// Asserts that an object is null. AssertionError is thrown if the object is not null.
jsunit.assertNull("Null test", null);
```

assertSame

void **assertSame** (expected, actual)

Asserts that two values are the same. AssertionError is thrown if the expected value is not the actual one.

Parameters

{Object} expected - the expected value.
 {Object} actual - the actual value.

Returns

void

Sample

```
// Asserts that two values are the same. AssertionError is thrown if the expected value is not the
actual one.
var a = new Date(1990, 1, 1);
var b = a;
jsunit.assertSame(a, b);
```

assertSame

void **assertSame** (message, expected, actual)

Asserts that two values are the same. AssertionError is thrown if the expected value is not the actual one.

Parameters

{String} message - The test description/message.
 {Object} expected - the expected value.
 {Object} actual - the actual value.

Returns

void

Sample

```
// Asserts that two values are the same. AssertionError is thrown if the expected value is not the
actual one.
var a = new Date(1990, 1, 1);
var b = a;
jsunit.assertSame("Same test", a, b);
```

assertTrue

void **assertTrue** (boolean_condition)

Asserts that a condition is true. AssertionError is thrown if the evaluation was not true.

Parameters

{Boolean} boolean_condition - the actual value.

Returns

void

Sample

```
// Asserts that a condition is true. AssertionError is thrown if the evaluation was not true.
jsunit.assertTrue("True test", application.isLastPrintPreviewPrinted());
```

assertTrue

void **assertTrue** (message, boolean_condition)

Asserts that a condition is true. AssertionError is thrown if the evaluation was not true.

Parameters

{String} message - The test description/message.
 {Boolean} boolean_condition - the actual value.

Returns

void

Sample

```
// Asserts that a condition is true. AssertionError is thrown if the evaluation was not true.
jsunit.assertTrue("True test", application.isLastPrintPreviewPrinted());
```

assertUndefinedvoid **assertUndefined** (undefinedValue)

Asserts that an object is undefined. AssertionError is thrown if the object is defined.

Parameters{[Object](#)} undefinedValue - the actual value.**Returns**

void

Sample

```
// Asserts that an object is undefined. AssertionError is thrown if the object is defined.
jsunit.assertUndefined("Undefined test", thisIsUndefined);
```

assertUndefinedvoid **assertUndefined** (message, undefinedValue)

Asserts that an object is undefined. AssertionError is thrown if the object is defined.

Parameters{[String](#)} message - The test description/message.{[Object](#)} undefinedValue - the actual value.**Returns**

void

Sample

```
// Asserts that an object is undefined. AssertionError is thrown if the object is defined.
jsunit.assertUndefined(thisIsUndefined);
```

failvoid **fail** (message)

Fails a test. AssertionError is always thrown.

Parameters{[String](#)} message - The test description/message. This is usually the only parameter specified when calling this method.**Returns**

void

Sample

```
// Fails a test. AssertionError is always thrown.
jsunit.fail("Fail test");
jsunit.fail("test", null, "Fail"); // 2nd param is not used in Servoy, params 3 and 1 get merged to form a
message. The result is the same as in the line above.
```

failvoid **fail** (message, instanceOfCallStack)

Fails a test. AssertionError is always thrown.

Parameters{[String](#)} message - The test description/message. This is usually the only parameter specified when calling this method.{[Object](#)} instanceOfCallStack - an internal JSUnit call stack. Use null for this if you want to get to the next optional parameter. Usually not specified.**Returns**

void

Sample

```
// Fails a test. AssertionError is always thrown.
jsunit.fail("Fail test");
jsunit.fail("test", null, "Fail"); // 2nd param is not used in Servoy, params 3 and 1 get merged to form a
message. The result is the same as in the line above.
```

fail

void **fail** (message, instanceOfCallStack, userMessage)

Fails a test. AssertionError is always thrown.

Parameters

{[String](#)} message - The test description/message. This is usually the only parameter specified when calling this method.

{[Object](#)} instanceOfCallStack - an internal JSUnit call stack. Use null for this if you want to get to the next optional parameter. Usually not specified.

{[String](#)} userMessage - a user message. Usually not specified.

Returns

void

Sample

```
// Fails a test. AssertionError is always thrown.
jsunit.fail("Fail test");
jsunit.fail("test", null, "Fail"); // 2nd param is not used in Servoy, params 3 and 1 get merged to form a
message. The result is the same as in the line above.
```