# Celestra Assert plugin cheatsheet – v6.1.2 – https://github.com/Serrin/Celestra/

| Plugin properties | Description |
|---|---|
| `assert(condition [, message | error]);` | `celestra.assert(condition [, message | error]);` |
| `assert.strict = true` | Default value is true. If this property is true, then always the strict equality is used in the assert.equal();, assert.notEqual();, assert.deepEqual();, assert.notDeepEqual(); |
| `assert.deepEqual(value1, value2 [, message | error]);` | `celestra.deepStrictEqual(value1, value2 [, message | error]);`<br>*or*<br>`celestra.assertDeepEqual(value1, value2 [, message | error]);` |
| `assert.doesNotMatch(string, regexp[, message | error]);` | `celestra.assertDoesNotMatch(string, regexp[, message | error]);` |
| `assert.equal(value1, value2 [, message | error]);` | `celestra.assertStrictEqual(value1, value2 [, message | error]);`<br>*or*<br>`celestra.assertEqual(value1, value2 [, message | error]);` |
| `assert.fail(message OR error);` | `celestra.assertFail(message OR error);` |
| `assert.isFalse(condition [, message | error]);` | `celestra.assertFalse(condition [, message | error]);` |
| `assert.is(value, expectedType [, message | error]);` | `celestra.assertIs(value, expectedType [, message | error]);` |
| `assert.isNot(value, expectedType [, message | error]);` | `celestra.assertIsNot(value, expectedType [, message | error]);` |
| `assert.isNotNullish(value [, message | error]);` | `celestra.assertIsNotNullish(value [, message | error]);` |
| `assert.isNullish(value [, message | error]);` | `celestra.assertIsNullish(value [, message | error]);` |
| `assert.isTrue(condition [, message | error]);` | `celestra.assertTrue(condition [, message | error]);` |
| `assert.match(string, regexp[, message | error]);` | `celestra.assertMatch(string,regexp[, message | error]);` |
| `assert.notDeepEqual(value1, value2 [, message | error]);` | `celestra.assertNotDeepStrictEqual(value1, value2 [, message | error]);`<br>*or*<br>`celestra.assertNotDeepEqual(value1, value2 [, message | error]);` |
| `assert.notEqual(value1, value2 [, message | error]);` | `celestra.assertNotStrictEqual(value1, value2 [, message | error]);`<br>*or*<br>`celestra.assertNotEqual(value1, value2 [, message | error]);` |
| `assert.notStrictEqual(value1, value2 [, message | error]);` | `celestra.assertNotStrictEqual(value1, value2 [, message | error]);` |
| `assert.ok(condition [, message | error]);` | `celestra.assert(condition [, message | error]);` |
| `assert.strictEqual(value1, value2 [, message | error]);` | `celestra.assertStrictEqual(value1, value2 [, message | error]);` |
| `assert.Throws(callback [, message | error]);` | `celestra.assertThrows(callback [, message | error]);` |