|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | http://chaijs.com/public/img/chai-logo.png | **Chai Assertion Library** Cheat Sheet – Assert Interface | | | |
| |  |  | | --- | --- | | Install |  | | npm install chai | |  |  |  | | --- | --- | | Include |  | | var assert = require( 'chai' ).assert ; | |  |  |  | | --- | --- | | Package.json |  | | "devDependencies": {  "chai": "\*",  "mocha": "\*" } | |  |  |  | | --- | --- | | Usage |  | | describe( 'feature', function() {  it( 'test', function( done ) {  assert.equal( 1, 2 ) ;  done() ;  }  } | | | |  |  | | --- | --- | | API Reference |  | | assert( expression, message ) Evaluate *expression*, if false then fail with message | | | .fail( actual, expected, [message], [operator] )  Throw a failure | | | .ok / .notOk( object, [message] )  Asserts that *object* is / is not truthy | | | .equal / .notEqual( actual, expected, [message] )  Asserts non-strict equality of *actual* and *expected* | | | .strictEqual / .notStrictEqual( actual, expected, [message] )  Asserts strict equality of *actual* and *expected* | | | .deepEqual / .notDeepEqual( actual, expected, [message] )  Asserts deep equality of *actual* and *expected* | | | .isTrue / .isFalse( value, [message] )  Asserts that *value* is / is not true | | | .isAbove / .isBelow( value1, value2, [message] )  Asserts that *value1* is above / below *value2* | | | .isNull / .isNotNull( value, [message] ) Asserts that *value* is / is not null | | | .isUndefined / .isDefined( value, [message] )  Asserts that *value* is / is not undefined | | | .isFunction / .isNotFunction( value, [message] )  Asserts that *value* is / is not a function | | | .isObject / .isNotObject( value, [message] )  Asserts that *value* is / is not an object | | | .isArray / .isNotArray( value, [message] )  Asserts that *value* is / is not an array | | | .isString / .isNotString( value, [message] )  Asserts that *value* is / is not a string | | | .isNumber / .isNotNumber( value, [message] )  Asserts that *value* is / is not a number | | | .isBoolean / .isNotBoolean( value, [message] )  Asserts that *value* is / is not a boolean | | | .typeOf / .notTypeOf( value, name, [message] )  Asserts that type of *value* is / is not *name* | | | |  |  | | --- | --- | | API Reference |  | | .instanceOf / .notInstanceOf( object, constructor, [message] )  Assets that *value* is / is not an instance of *constructor* | | | .include / .notInclude( haystack, needle, [message] )  Asserts that array or string *haystack* includes / does not include value *needle* | | | .match / .notMatch( value, regexp, [message] )  Asserts that *value* matches / does not match regular expression *regexp* | | | .property / .notProperty( object, property, [message] )  Asserts that *object* has / does not have property named *property* | | | .deepProperty / .notDeepProperty( object, property, [message] )  Asserts that *object* has / does not have property *property* (which may use dot/bracket notation for deep reference) | | | .propertyVal / .propertyNotVal( object, property, value, [message] )  Asserts that *object* has / does not have property *property* with value *value* | | | .deepPropertyVal / .deepPropertyNotVal( object, property, value, [message] )  Asserts that *object* has / does not have property *property* (which may use dot/bracket notation for deep reference) with value *value* | | | .lengthOf( object, length, [message] )  Asserts that *object* has *length* property with expected value | | | .sameMembers( array1, array2, [message] )  Asserts that *array1* and *array2* have the same members, order is not taken into account | | | .sameDeepMembers( array1, array2, [message] )  Asserts that *array1* and *array2* have the same members using deep-equality checking, order is not taken into account | | | .includeMembers( array1, array2, [message] )  Asserts that *array1* contains all elements from *array2*, order is not taken into account | | |
|  | | |