

Figure 12.2. Test::Unit assertions

`assert(boolean, [message])`
 Fails if *boolean* is false or nil.

`assert_nil(obj, [message])`
`assert_not_nil(obj, [message])`
 Expects *obj* to be (not) nil.

`assert_equal(expected, actual, [message])`
`assert_not_equal(expected, actual, [message])`
 Expects *obj* to equal/not equal *expected*, using `==`.

`assert_in_delta(expected_float, actual_float, delta, [message])`
 Expects that the actual floating-point value is within *delta* of the expected value.

`assert_raise(Exception, ...) { block }`
`assert_nothing_raised(Exception, ...) { block }`
 Expects the block to (not) raise one of the listed exceptions.

`assert_instance_of(klass, obj, [message])`
`assert_kind_of(klass, obj, [message])`
 Expects *obj* to be a kind/instance of *klass*.

`assert_respond_to(obj, message, [message])`
 Expects *obj* to respond to *message* (a symbol).

`assert_match(regexp, string, [message])`
`assert_no_match(regexp, string, [message])`
 Expects *string* to (not) match *regexp*.

`assert_same(expected, actual, [message])`
`assert_not_same(expected, actual, [message])`
 Expects *expected*.`equal?(actual)`.

`assert_operator(obj1, operator, obj2, [message])`
 Expects the result of sending the message *operator* to *obj1* with parameter *obj2* to be true.

`assert_throws(expected_symbol, [message]) { block }`
 Expects the block to throw the given symbol.

`assert_send(send_array, [message])`
 Sends the message in *send_array*[1] to the receiver in *send_array*[0], passing the rest of *send_array* as arguments. Expects the return value to be true.

`flunk(message="Flunked")`
 Always fail.