

Standard exceptions

<code>assert</code>	<code>object</code>	Passes if object is not false or nil	<code>assert [1, 2].include?(2)</code>
<code>assert_in_delta</code>	<code>expected_float, actual_float, delta</code>	Passes if <code>expected_float</code> and <code>actual_float</code> are equal within delta tolerance	<code>assert_in_delta 0.05, (50000.0 / 10**6), 0.00001</code>
<code>assert_send</code>	<code>send_array</code>	Passes if the method <code>send</code> returns a true value <code>send_array</code> is composed of: a receiver, method, arguments to the method	<code>assert_send [[1, 2], :include?, 2]</code>
<code>assert_raise</code>	<code>*args</code>	Passes if the block raises one of the given exceptions	<code>assert_raise RuntimeError, LoadError { raise 'Boom!!!' }</code>
<code>assert_raises</code>	<code>*args, &block</code>	Alias of <code>assert_raise</code>	Will be deprecated in 1.9, and removed in 2.0.
<code>assert_nothing_raised</code>	<code>*args</code>	Passes if block does not raise an exception	<code>assert_nothing_raised { [1, 2].uniq }</code>
<code>assert_throws</code>	<code>expected_symbol, &proc</code>	Passes if the block throws <code>expected_symbol</code>	<code>assert_throws :done { throw :done }</code>
<code>assert_nothing_thrown</code>	<code>&proc</code>	Passes if block does not throw anything	<code>assert_nothing_thrown { [1, 2].uniq }</code>
<code>assert_instance_of</code>	<code>klass, object</code>	Passes if <code>object.instance_of? klass</code>	<code>assert_instance_of String, 'foo'</code>
<code>assert_kind_of</code>	<code>klass, object</code>	Passes if <code>object.kind_of? klass</code>	<code>assert_kind_of Object, 'foo'</code>
<code>assert_respond_to</code>	<code>object, method</code>	Passes if <code>object.respond_to? method</code>	<code>assert_respond_to 'foo_method', :object</code>
<code>assert_match</code>	<code>regexp, string</code>	Passes if <code>string =~ pattern</code>	<code>assert_match(/d+/, 'five, 6, seven')</code>
<code>assert_no_match</code>	<code>regexp, string</code>	Passes if <code>regexp !~ string</code>	<code>assert_no_match(/two/, 'one 2 three')</code>
<code>assert_same</code>	<code>expected, actual</code>	Passes if <code>expected</code> and <code>actual</code> are the same instance	<code>o = Object.new ; assert_same o, o</code>
<code>assert_not_same</code>	<code>expected, actual</code>	Passes if <code>expected</code> and <code>actual</code> are not the same instance	<code>assert_not_same Object.new, Object.new</code>
<code>assert_operator</code>	<code>object1, operator, object2</code>	Passes if <code>object1.send(operator, object2)</code> is true	<code>assert_operator 5, :>=, 4</code>
<code>assert_equal</code>	<code>expected, actual</code>	Passes if <code>expected == actual</code>	<code>assert_equal 'MY STRING', 'my string'.upcase</code>
<code>assert_not_equal</code>	<code>expected, actual</code>	Passes if <code>expected != actual</code>	<code>assert_not_equal 'some string', 5</code>
<code>assert_nil</code>	<code>object</code>	Passes if object is nil	<code>assert_nil [1, 2].uniq!</code>
<code>assert_not_nil</code>	<code>object</code>	Passes if <code>! object.nil?</code>	<code>assert_not_nil '1 two 3'.sub!(/two/, '2')</code>

Test::Rails [if you're using ZenTest products – www.zenspider.com/ZSS/Products/ZenTest]

assert_assigned	ivar, value
assert_content_type	type
assert_flash	key, content
assert_image	src
assert_error_on	field, type
assert_field	form_action, type, model, column, value
assert_input	form_action, type, name, value
assert_label	form_action, name, include_f
assert_links_to	href, content
assert_multipart_form	form_action
assert_select	form_action
assert_submit	form_action, model, colum, options
assert_tag_in_form	form_action, options
assert_empty	object
assert_includes	object, item

Rails exceptions

assert_response	response_code	Passes if response code is response_code	assert_response :success
assert_redirected_to	options	Passes if last action redirect_to matches options	assert_redirected_to :controller => 'blog' assert_redirected_to blog_messages()
assert_template	template_name	Passes if the request was rendered with the appropriate template file	assert_template 'admin/blog'
assert_recognizes	expected_options, path	Passes if the routing of the given path was handled correctly and that the parsed options match	assert_recognizes({:controller => 'items', :action => 'list', :id => '1'}, 'items/list/1')
assert_generates	expected_path, options	Passes if the provided options can be used to generate the provided path. This is the inverse of assert_recognizes	assert_generates("/items/list/1", { :controller => "items", :action => "list", :id => "1" })
assert_routing	path, options, defaults, extras	Passes if the URL generated from options is the same as path, and also that the options recognized from path are the same as options. This essentially combines assert_recognizes and assert_generates into one step.	
assert_tag	options	Passes if there is a tag/node/element in the body of the response that meets all of the given conditions - see doc for further infos about conditions	assert_tag :span, :attributes => { :id => "x" } assert_tag :tag => "span", :descendant => { :tag => "strong" }
assert_no_tag	options	Identical to assert_tag, but asserts that a matching tag does not exist.	
assert_dom_equal	expected, actual		assert_dom_equal('foo', @response.body)
assert_dom_not_equal	expected, actual		
assert_valid	model_instance	Passes if model_instance passes validation	assert_valid customer