Ruby - Bug #10745

Special combinations of parameters in assert_equal (test/unit) may cause exceptions

01/15/2015 04:26 PM - herwinw (Herwin Quarantainenet)

Status:	Rejected		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.1.2p95	**Backport:**	2.0.0: UNKNOWN, 2.1: UNKNOWN, 2.2: UNKNOWN

Description

```
require 'test/unit'
require 'ipaddr'

class TestX < Test::Unit::TestCase
  def test_x
    assert_equal([IPAddr.new('1.2.3.4')], [[1,2,3]])
  end
end
```

This results in the following trace:

```
NoMethodError: undefined method `to_i' for [1, 2, 3]:Array
    /usr/lib/ruby/2.1.0/ipaddr.rb:471:in `initialize'
    /usr/lib/ruby/2.1.0/ipaddr.rb:516:in `new'
    /usr/lib/ruby/2.1.0/ipaddr.rb:516:in `coerce_other'
    /usr/lib/ruby/2.1.0/ipaddr.rb:150:in `=='
    /usr/lib/ruby/2.1.0/test/unit/assertions.rb:250:in `=='
    /usr/lib/ruby/2.1.0/test/unit/assertions.rb:250:in `assert_equal'
```

I don't think the assert_equal is supposed to throw these kind of errors

History

#1 - 01/16/2015 12:09 AM - sawa (Tsuyoshi Sawada)

An interesting case. assert_equal seems to be assuming that the == method on the relevant class (IPAddr#== here) does not raise an error, but that should not be taken for granted. It should be wrapped in a rescue block.

#2 - 01/16/2015 01:12 AM - nobu (Nobuyoshi Nakada)

- Description updated

- Status changed from Open to Rejected

#3 - 01/16/2015 12:03 PM - kou (Kouhei Sutou)

I think that it should be fixed in ipaddr.

The following code should print false instead of raising an exception:

```
% /tmp/local/bin/ruby -v -r ipaddr -e 'puts(IPAddr.new == [])'
ruby 2.3.0dev (2015-01-16 trunk 49282) [x86_64-linux]
/tmp/local/lib/ruby/2.3.0/ipaddr.rb:471:in `initialize': undefined method `to_i' for []:Array (NoMethodError)
 from /tmp/local/lib/ruby/2.3.0/ipaddr.rb:516:in `new'
 from /tmp/local/lib/ruby/2.3.0/ipaddr.rb:516:in `coerce_other'
 from /tmp/local/lib/ruby/2.3.0/ipaddr.rb:150:in `=='
 from -e:1:in `<main>'
```

#4 - 01/16/2015 12:34 PM - Eregon (Benoit Daloze)

Tsuyoshi Sawada wrote:

> An interesting case. assert_equal seems to be assuming that the == method on the relevant class (IPAddr#== here) does not raise an error, but that should not be taken for granted. It should be wrapped in a rescue block.

I disagree, the exception is more useful as an error than just assert_equal returning false.

Kouhei Sutou wrote:

> I think that it should be fixed in ipaddr.

Indeed, the error from IPAddr could be clearer,

#5 - 01/16/2015 10:15 PM - marcandre (Marc-Andre Lafortune)

Benoit Daloze wrote:

> Tsuyoshi Sawada wrote:
>
> > An interesting case. assert_equal seems to be assuming that the == method on the relevant class (IPAddr#== here) does not raise an error, but that should not be taken for granted. It should be wrapped in a rescue block.
>
> I disagree, the exception is more useful as an error than just assert_equal returning false.

Agreed.

> Indeed, the error from IPAddr could be clearer,

Mmm, not quite. Calling == should never raise, so there should not be an error.

#6 - 01/17/2015 11:05 PM - sawa (Tsuyoshi Sawada)

Benoit Daloze wrote:

> I disagree, the exception is more useful as an error than just assert_equal returning false.

According to http://docs.ruby-lang.org/ja/2.1.0/method/Test=3a=3aUnit=3a=3aAssertions/i/assert_equal.html, it should raise MiniTest::Assertion error, which should wrap the message and backtrace from the NoMethodError. It should not raise a NoMethodError. This is a bug of minitest in that it could not properly handle and report the test object's (ipaddr) bug.

After all, test frameworks are for detecting and reporting the test object's bugs. It does not make sense for a test framework to raise an error by itself when the test object has a bug.