# Ruby master - Bug #9589

## Stack level too deep during eval causes segmentation fault

03/04/2014 12:16 AM - carlosayam (carlos aya)

| | | | |
|---|---|---|---|
| **Status:** | Closed | | |
| **Priority:** | Normal | | |
| **Assignee:** | | | |
| **Target version:** | | | |
| **ruby -v:** | 1.9.3 | **Backport:** | |

**Description**

The following silly code, which imho should generate a " stack level too deep (SystemStackError)", segfaults

```
$ echo 'eval($)' | ruby -n -e 'eval($)'
```

I thought it may be a bug, not sure.

My version:
carlos-mac$ ruby -v
ruby 1.9.3p327 (2012-11-10) [x86_64-darwin12.2.0]

### History

**#1 - 03/04/2014 03:31 AM - Cezary (Cezary Baginski)**

Not sure if it's related, but I'm getting segfaults instead of SystemStackErrors with recursive lambda calls from:

example_spec.rb:

```
describe Fixnum do
subject { subject }
it { should be }
end
```

by running:

```
rspec example_spec.rb
```

using this version of rspec:

rspec (2.14.1)
rspec-core (2.14.8)

Both on:

ruby 2.1.1p76 (2014-02-24 revision 45161) [x86_64-linux]

and:

ruby 2.2.0dev (2014-03-04 trunk 45264) [x86_64-linux]

but the debug build of 2.2.0 (-O0 and -ggdb3) gives SystemStackError as expected.

(all examples compiled with gcc (Ubuntu/Linaro 4.7.3-1ubuntu1) 4.7.3)

**#2 - 03/05/2014 07:01 AM - carlosayam (carlos aya)**

Interesting, I searched the rspec-core code and it uses instance_eval/class_eval. That prompted me to try these faulty snippets...

```
b = Proc.new do
b.instance_eval(&b)
end
b.instance_eval(&b)
```

and this one too...

```
b = Proc.new do
Proc.class_eval(&b)
```

```
end
Proc.class_eval(&b)
```

Both segfault. I suspect eval, instance_eval and class_eval are not guarded against stack errors (again v.1.9.3 not sure about 2.x).

**#3 - 06/27/2014 08:15 AM - runephilosof (Rune Philosof)**

This bug should be against the newest ruby stable 2.1.2, but I cannot change that.
I just encountered it in a spec that looks like this:

```
require 'spec_helper'

describe "test" do
let(:die) { "this is fine" }
context "with bad context" do
let(:die) { die + ", but this will fail" }
it "it throws segmentation fault" do
expect { die }.to raise_error("Segmentation Fault")
end
end
end
```

This is with ruby 2.1.2 and rspec-core 2.14.8

**#4 - 06/30/2014 08:51 AM - decuplet (Nikita Shilnikov)**

This command fails with segfault on 2.1.1 and 2.1.2:

```
ruby -e 'define_method(:bar) {send(:bar)}; bar'
```

It's OK on 2.0.0 and below.

**#5 - 09/10/2019 03:21 AM - jeremyevans0 (Jeremy Evans)**

*- Backport deleted (1.9.3: UNKNOWN, 2.0.0: UNKNOWN, 2.1: UNKNOWN)*

*- Status changed from Open to Closed*

From my testing with class Object; define_method(:bar) {send(:bar)}; bar end:

1.9-2.1: SystemStackError
2.2-2.4: segfault
2.5-master: SystemStackError

As this issue appears to have been fixed, closing.  If you can reproduce with a currently supported Ruby version, please post back here.