Ruby - Bug #4487

require_relative fails in an eval'ed file

03/10/2011 03:25 AM - rogerdpack (Roger Pack)

Status:	Closed		
Priority:	Normal		
Assignee:	matz (Yukihiro Matsumoto)		
Target version:			
ruby -v:	-	**Backport:**	

Description

=begin
Hello all.

$cat eval_me1.rb
eval(File.read('eval_me2.rb'), binding, File.expand_path('./eval_me2.rb'))
$cat eval_me2.rb
require_relative 'eval_me1.rb'
$ ruby eval_me1.rb
C:/dev/ruby/faster_require/spec/eval_me2.rb:1:in require_relative': cannot infer basepath (LoadError) from
C:/dev/ruby/faster_require/spec/eval_me2.rb:1:in '
from eval_me1.rb:1:in eval' from eval_me1.rb:1:in '

I suppose was assuming that if eval included a filename, then require_relative would work from within it. Perhaps I am mistaken?
Thanks!
-r
=end

Related issues:

Related to Ruby - Bug #4352: [patch] Fix eval(s, b) backtrace; make eval(s, b...	**Closed**
Related to Ruby - Bug #7391: Allow to use require_relative from eval and irb ...	**Closed**

History

#1 - 06/26/2011 06:27 PM - naruse (Yui NARUSE)

- Status changed from Open to Assigned

- Assignee set to mame (Yusuke Endoh)

#2 - 07/05/2011 02:53 AM - mame (Yusuke Endoh)

- ruby -v changed from ruby 1.9.3dev (2011-03-04 trunk 31024) [i386-mingw32] to -

Hello,

> Â $cat eval_me1.rb
> Â eval(File.read('eval_me2.rb'), binding, File.expand_path('./eval_me2.rb'))
> Â $cat eval_me2.rb
> Â require_relative 'eval_me1.rb'
> Â $ ruby eval_me1.rb
> Â C:/dev/ruby/faster_require/spec/eval_me2.rb:1:in require_relative': cannot infer basepath (LoadError) Â Â Â Â from
> C:/dev/ruby/faster_require/spec/eval_me2.rb:1:in '
> Â Â Â Â from eval_me1.rb:1:in eval' Â Â Â Â from eval_me1.rb:1:in '
>
> I suppose was assuming that if eval included a filename, then require_relative would work from within it. Perhaps I am mistaken?

I think your expectation is reasonable, though I personally dislike
the eval's feature to fake filepath.

The following patch makes require_relative use the given file path.
I'm afraid if I should include this patch in 1.9.3 because I can't
estimate the impact of this patch. What do you think?

diff --git a/vm_eval.c b/vm_eval.c
index 7df7f5f..3710401 100644

```
--- a/vm_eval.c
+++ b/vm_eval.c
@@ -1007,7 +1007,7 @@ eval_string_with_cref(VALUE self, VALUE src,
VALUE scope, NODE *cref, const char
/* make eval iseq */
th->parse_in_eval++;
th->mild_compile_error++;
```

- iseqval

#3 - 07/05/2011 12:52 PM - mame (Yusuke Endoh)

- Assignee changed from mame (Yusuke Endoh) to matz (Yukihiro Matsumoto)

Related to #4352.

I need matz's judgment.

--
Yusuke Endoh mame@tsg.ne.jp

#4 - 07/05/2012 05:07 PM - LTe (Piotr Niełacny)

=begin
If in irb we can execute

(({load("file.rb")}))

why we can't

(({require_relative("file")}))

Ruby just return exception (LoadError: cannot infer basepath). Unfortunately, this is a 'lie' because ruby can recognize basepath.
=end

#5 - 07/05/2012 05:26 PM - shyouhei (Shyouhei Urabe)

=begin

@LTe sorry, I can't get it. load loads from $LOAD_PATH, while require_relative requires from relative path. They are different.

=end

#6 - 07/06/2012 04:04 PM - LTe (Piotr Niełacny)

@shyouhei (Shyouhei Urabe) yes I agree but load method tries to load file from relative path. When load method can't find file in relative path it loads from $LOAD_PATH. The same can be done by require_relative (recognize path).

https://github.com/ruby/ruby/pull/139

#7 - 07/06/2012 05:52 PM - shyouhei (Shyouhei Urabe)

=begin
@LTe I'd rather ask you "require_relative loads something relative from WHAT?"

Obviously it is not relative from your mind :)

Current require_relative loads relative from where the require_relative command is written. So when in IRB sessions, it fails to infer where it is beacuse the command is written in a non-file (console).

OTOH load loads from process PWD, which is possible in IRB.

So the point is, if you want require_relative to work on an IRB session, you have to define "from where require_relative should search relativeness".
=end

#8 - 07/06/2012 06:13 PM - Eregon (Benoit Daloze)

shyouhei (Shyouhei Urabe) wrote:

> Current require_relative loads relative from where the require_relative command is written. So when in IRB sessions, it fails to infer where it is beacuse the command is written in a non-file (console).
>
> OTOH load loads from process PWD, which is possible in IRB.
>
> So the point is, if you want require_relative to work on an IRB session, you have to define "from where require_relative should search

> relativeness".

From the process current working directory I guess, especially since you almost always launch IRB from a terminal.

Personally I'm doing require './myfile' which is not the most elegant, but if you don't have completion in IRB, that's shorter to type.

Otherwise, there's always the option to do irb -I. and use plain require.

#9 - 07/06/2012 06:59 PM - naruse (Yui NARUSE)

Eregon (Benoit Daloze) wrote:

> shyouhei (Shyouhei Urabe) wrote:
>
> > Current require_relative loads relative from where the require_relative command is written. So when in IRB sessions, it fails to infer where it is beacuse the command is written in a non-file (console).
> >
> > OTOH load loads from process PWD, which is possible in IRB.
> >
> > So the point is, if you want require_relative to work on an IRB session, you have to define "from where require_relative should search relativeness".
>
> From the process current working directory I guess, especially since you almost always launch IRB from a terminal.
>
> Personally I'm doing require './myfile' which is not the most elegant, but if you don't have completion in IRB, that's shorter to type.
>
> Otherwise, there's always the option to do irb -I. and use plain require.

require_relative is introduced to avoid accidentally require a malicious file on the current working directory.
So it can't be acceptable.
Use require or load on such case.

#10 - 07/06/2012 07:08 PM - shyouhei (Shyouhei Urabe)

naruse (Yui NARUSE) wrote:

> require_relative is introduced to avoid accidentally require a malicious file on the current working directory.
> So it can't be acceptable.
> Use require or load on such case.

I'm not pretty sure about this. Is there a chance for a (proposed behaviour of) require_relative to require a malicious file on the current directory?

Because you are on an IRB session and intentionally emitting require_relative (not require), I doubt the danger you say.

#11 - 07/06/2012 07:21 PM - naruse (Yui NARUSE)

shyouhei (Shyouhei Urabe) wrote:

> naruse (Yui NARUSE) wrote:
>
> > require_relative is introduced to avoid accidentally require a malicious file on the current working directory.
> > So it can't be acceptable.
> > Use require or load on such case.
>
> I'm not pretty sure about this. Is there a chance for a (proposed behaviour of) require_relative to require a malicious file on the current directory?
>
> Because you are on an IRB session and intentionally emitting require_relative (not require), I doubt the danger you say.

- irb is not the only user of eval.
- A user won't always use require_relative intentionally.
- There is a suitable another way: require './myfile'

With those reason, I don't think require_relative should be changed.

#12 - 07/06/2012 08:06 PM - Eregon (Benoit Daloze)

naruse (Yui NARUSE) wrote:

> Eregon (Benoit Daloze) wrote:

> From the process current working directory I guess, especially since you almost always launch IRB from a terminal.

> require_relative is introduced to avoid accidentally require a malicious file on the current working directory.
> So it can't be acceptable.
> Use require or load on such case.

I see, you're right.
Indeed, with this in mind I think it's not worth changing, and the actual require_relative behavior is clearer (relative to "this file" directory, if there is no accurate "this file", just #raise).

#13 - 07/14/2012 02:59 PM - ko1 (Koichi Sasada)

- Assignee changed from matz (Yukihiro Matsumoto) to mame (Yusuke Endoh)

mame-san, please ask matz.

#14 - 02/02/2013 01:06 PM - mame (Yusuke Endoh)

- Subject changed from require_relative fails in an eval'ed file to require_relative fails in an eval'ed file

- Assignee changed from mame (Yusuke Endoh) to matz (Yukihiro Matsumoto)

- Target version set to 2.6

#15 - 03/23/2013 02:50 PM - Conrad.Irwin (Conrad Irwin)

This bug also affects pry: https://github.com/pry/pry/issues/880. Our use-case is slightly different because we are doing TOPLEVEL_BINDING.eval("some code", "/absolute/path.rb").

I think that when an absolute path is set in eval, then require_relative should use it.

#16 - 08/18/2015 02:28 PM - julik (Julik Tarkhanov)

This is actually very pertinent for Rack as well, because currently config.ru does not support require_relative which is very counterintuitive.

#17 - 12/07/2017 11:49 AM - mame (Yusuke Endoh)

- Status changed from Assigned to Closed

Now, it works. I'm unsure who changed the behavior... Anyway, closing.