Ruby trunk - Feature #5690

Module#qualified_const_get

11/30/2011 12:10 PM - wycats (Yehuda Katz)

Status:	Closed
Priority:	Normal
Assignee:	tenderlovemaking (Aaron Patterson)
Target version:	

Description

It would be great if there was a way to dynamically load a constant path:

```
module Foo
  module Bar
    module Baz
    end
  end
end

Foo.qualified_const_get("Bar::Baz") => Foo::Bar::Baz
```

Related issues:

Related to Ruby trunk - Feature #5666: Make rb_path2class public	**Closed**	**11/24/2011**
Related to Ruby trunk - Feature #12319: `Module#const_get` does not accept sy...	**Open**	
Has duplicate Ruby trunk - Feature #767: Module#const_get□□□□□□□□□□□□□□□□□□□□□□□□	**Rejected**	**11/21/2008**

History

#1 - 11/30/2011 12:13 PM - naruse (Yui NARUSE)

- Category set to core

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

- Target version set to 2.0.0

#2 - 11/30/2011 03:23 PM - matz (Yukihiro Matsumoto)

- ruby -v changed from ruby 1.9.3p0 (2011-10-30 revision 33570) [x86_64-darwin11.2.0] to -

Hi,

In message "Re: [ruby-core:41404] [ruby-trunk - Bug #5690][Open] Module#qualified_const_get"
on Wed, 30 Nov 2011 12:10:02 +0900, Yehuda Katz wycats@gmail.com writes:

> It would be great if there was a way to dynamically load a constant path:
>
> ```
> module Foo
>   module Bar
>     module Baz
>     end
>   end
> end
>
> Foo.qualified_const_get("Bar::Baz") => Foo::Bar::Baz
> ```

Interesting.

- is there any use-case for this method?
- if Bar is not a class nor module, what kind error should be raised?

- is qualified_const_get an appropriate name for the function?

```
                matz.
```

#3 - 11/30/2011 06:24 PM - matz (Yukihiro Matsumoto)

Hi,

In message "Re: [ruby-core:41408] Re: [ruby-trunk - Bug #5690][Open] Module#qualified_const_get"
on Wed, 30 Nov 2011 16:52:36 +0900, Yehuda Katz wycats@gmail.com writes:

> - is there any use-case for this method?
>
> We use it often in Rails.

Could you show us the concrete example? It is difficult for me to imagine.

> - if Bar is not a class nor module, what kind error should be raised?
>
> TypeError: "(…) is not a class/module"

Understood.

> - is qualified_const_get an appropriate name for the function?
>
> Unknown. It's what we call it in Rails ;)

How about

```
Foo.const_get("Bar::Baz")
```

or

```
Foo.const_get(:Bar, :Baz)
```

?

```
                          matz.
```

#4 - 12/01/2011 06:23 AM - wycats (Yehuda Katz)

Yehuda Katz
(ph) 718.877.1325

2011/11/30 Yukihiro Matsumoto matz@ruby-lang.org

> Hi,
>
> In message "Re: [ruby-core:41408] Re: [ruby-trunk - Bug #5690][Open]
> Module#qualified_const_get"
> on Wed, 30 Nov 2011 16:52:36 +0900, Yehuda Katz wycats@gmail.com
> writes:
>
> |> * is there any use-case for this method?
> |
> |We use it often in Rails.
>
> Could you show us the concrete example? It is difficult for me to
> imagine.

One case is to convert path names (file system paths) into constants.

In general, Rails uses inflections to convert names from different contexts. In the router, for instance, "foo/bar" is used to represent namespaces, which are then converted into "Foo::Bar" and then Foo::Bar.

> |> * if Bar is not a class nor module, what kind error should be raised?
> |>
> |

|TypeError: "(…) is not a class/module"

Understood.

|> * is qualified_const_get an appropriate name for the function?
|>
|
|Unknown. It's what we call it in Rails ;)

How about

Foo.const_get("Bar::Baz")

Sounds good to me. Currently, trying to const_set or const_get a name with
:: in it produces a NameError, so it should not introduces new issues.

or

Foo.const_get(:Bar, :Baz)

This doesn't satisfy the use-case in question.

?

matz.

#5 - 12/01/2011 10:23 AM - Anonymous

On Wed, Nov 30, 2011 at 04:52:36PM +0900, Yehuda Katz wrote:

On Tue, Nov 29, 2011 at 10:04 PM, Yukihiro Matsumoto matz@ruby-lang.orgwrote:

Hi,

In message "Re: [ruby-core:41404] [ruby-trunk - Bug #5690][Open]
Module#qualified_const_get"
on Wed, 30 Nov 2011 12:10:02 +0900, Yehuda Katz wycats@gmail.com
writes:

|It would be great if there was a way to dynamically load a constant path:
|
| module Foo
| module Bar
| module Baz
| end
| end
| end
|
| Foo.qualified_const_get("Bar::Baz") => Foo::Bar::Baz

Interesting.

- is there any use-case for this method?

We use it often in Rails.

When serializing objects to yaml, the fully qualified class name is
emitted. When reviving those objects, the class name comes back as a
string, so psych uses rb_path2class to find the constant.

- if Bar is not a class nor module, what kind error should be raised?

TypeError: "(…) is not a class/module"

Agreed.

- is qualified_const_get an appropriate name for the function? Unknown. It's what we call it in Rails ;)

Yes.

--
Aaron Patterson
http://tenderlovemaking.com/

#6 - 12/04/2011 12:53 PM - naruse (Yui NARUSE)

(2011/12/01 5:59), Yehuda Katz wrote:

> Yehuda Katz
> (ph) 718.877.1325
>
> 2011/11/30 Yukihiro Matsumoto >
>
> ```
> Hi,
>
> In message "Re: [ruby-core:41408] Re: [ruby-trunk - Bug #5690][Open] Module#qualified_const_get"
>    on Wed, 30 Nov 2011 16:52:36 +0900, Yehuda Katz <wycats@gmail.com <mailto:wycats@gmail.com>> writes:
>
> |> * is there any use-case for this method?
> |
> |We use it often in Rails.
>
> Could you show us the concrete example?  It is difficult for me to
> imagine.
> ```
>
> One case is to convert path names (file system paths) into constants.
>
> In general, Rails uses inflections to convert names from different contexts. In the router, for instance, "foo/bar" is used to represent namespaces, which are then converted into "Foo::Bar" and then Foo::Bar.

"concrete exapmle" needs "what is the router?" and "when the router is used?".

--
NARUSE, Yui naruse@airemix.jp

#7 - 02/25/2012 01:53 PM - ko1 (Koichi Sasada)

(2011/11/30 12:10), Yehuda Katz wrote:

> It would be great if there was a way to dynamically load a constant path:
>
> ```
> module Foo
>   module Bar
>     module Baz
>     end
>   end
> end
>
> Foo.qualified_const_get("Bar::Baz") => Foo::Bar::Baz
> ```

Any progress on it?

There are several similar proposals.
I think 2.0 is a good opportunity to finish such discussion.

BTW, is it a "BUG"? (on title)

--
// SASADA Koichi at atdot dot net

#8 - 02/25/2012 07:01 PM - trans (Thomas Sawyer)

Please, can we avoid the use such abstract name.

Is the speed hit too great that Ruby's can't just use #const_get for this as well? I suspect the Rail's method name was chosen simply to avoid a monkey patch.

Of course, in Facets the method is just called #constant. Heaven forbid we use the obvious name ;-)

#9 - 02/25/2012 07:46 PM - fxn (Xavier Noria)

I wrote qualified_const_get in Active Support. The rationale for that name was: 1) I didn't want to touch const_get. const_get is supposed to raise an exception if the argument is not a valid constant name and "Foo::Bar" is not a valid constant name. I didn't want to change that expectation in such a fundamental method. And 2) I wanted a name that was obvious. Given that a Ruby programmer knows const_get, he will instantly know what qualified_const_get is going to do. In that sense I don't think this method is more abstract than const_get, it is in my view rather the natural companion of const_get.

That's for the current name. If this was going to be considered for Ruby 2.0 I think I'd prefer this behavior to be available in const_get itself.

#10 - 02/26/2012 01:04 AM - trans (Thomas Sawyer)

I don't think the name is particularly obvious. Nevertheless, I am hopeful that it won't matter. I think const_get can accommodate. In fact, I think it can even be made to handle:

const_get('Foo', 'Bar::Baz') #=> Foo::Bar::Baz

It is backward compatible, after all.

#11 - 02/26/2012 01:06 AM - trans (Thomas Sawyer)

@Xavier Let me "qualify" that ;-) I think your reasoning made very good sense for adding to ActiveSupport. But becoming part of Ruby, there is the opportunity to do it "right", as your last statement concurs.

#12 - 06/24/2012 03:22 AM - marcandre (Marc-Andre Lafortune)

Is anyone producing a slide-show for this?

#13 - 07/01/2012 01:25 AM - tenderlovemaking (Aaron Patterson)

- File bug5690.pdf added

Adding a slide for this feature.

#14 - 07/01/2012 01:28 AM - tenderlovemaking (Aaron Patterson)

- File deleted (bug5690.pdf)

#15 - 07/01/2012 01:29 AM - tenderlovemaking (Aaron Patterson)

- File bug5690.pdf added

updating the pdf

#16 - 07/01/2012 01:41 AM - marcandre (Marc-Andre Lafortune)

Cool slide :-)

+1 from me, but modifying existing const_get. No incompatibility as "Foo::Bar" are currently invalid constant names. And const_set should be modified as well (raises a Name if the path up to the last constant name does not exist or is not a Module/Class).

#17 - 07/01/2012 03:00 AM - trans (Thomas Sawyer)

I agree with marcandre. Why bother with an additional (and long named) method, when current method can suffice?

#18 - 07/02/2012 01:54 AM - mame (Yusuke Endoh)

Aaron, your slide is received. Thank you.

marcandre (Marc-Andre Lafortune) wrote:

> +1 from me, but modifying existing const_get.

I'll add it verbally during the presentation, if I recall.

--
Yusuke Endoh mame@tsg.ne.jp

#19 - 07/02/2012 11:53 PM - Anonymous

- File noname added

On Mon, Jul 02, 2012 at 01:54:32AM +0900, mame (Yusuke Endoh) wrote:

> Issue #5690 has been updated by mame (Yusuke Endoh).
>
> Aaron, your slide is received. Thank you.
>
> marcandre (Marc-Andre Lafortune) wrote:
>
> > +1 from me, but modifying existing const_get.
>
> I'll add it verbally during the presentation, if I recall.

Thank you!

--
Aaron Patterson
http://tenderlovemaking.com/

#20 - 07/14/2012 03:31 PM - ko1 (Koichi Sasada)

- Tracker changed from Bug to Feature

#21 - 07/23/2012 09:39 PM - mame (Yusuke Endoh)

- Assignee changed from matz (Yukihiro Matsumoto) to tenderlovemaking (Aaron Patterson)

Yehuda Katz and Aaron Patterson,

I'm happy to inform you that matz has accepted your proposal,
as one to extend existing const_get' (notqualified_const_get').

Aaron, could you implement a patch?

--
Yusuke Endoh mame@tsg.ne.jp

#22 - 07/24/2012 10:23 AM - cjheath (Clifford Heath)

Aaron,

On 24/07/2012, at 4:14 AM, Aaron Patterson wrote:

> On Mon, Jul 23, 2012 at 09:39:38PM +0900, mame (Yusuke Endoh) wrote:
>
> > Issue #5690 has been updated by mame (Yusuke Endoh).
> > Assignee changed from matz (Yukihiro Matsumoto) to tenderlovemaking (Aaron Patterson)
> > Yehuda Katz and Aaron Patterson,
> > I'm happy to inform you that matz has accepted your proposal,
> > as one to extend existing const_get' (notqualified_const_get').
> >
> > Aaron, could you implement a patch?
> Yes I will. Thanks! :-)

Please ensure that if the lookup fails, the exception indicates which part of the name caused the failure.

It's waaay past time that the industry moves past "ENOENT: No such file or directory" in its exception reporting :)

Thank-you.

Clifford Heath.

#23 - 10/24/2012 02:40 AM - tenderlovemaking (Aaron Patterson)

- File p2c.diff added

I've written a patch this this feature, and I've attached it here. Will someone review so that we can figure out what needs to change before it's applied? Thank you.

#24 - 10/27/2012 09:11 AM - ko1 (Koichi Sasada)

I got SEGV.
could you run "make test-rubyspec" on your environment?

```
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/const_get_spec.rb:34: [BUG] Segmentation fault
ruby 2.0.0dev (2012-10-27 trunk 37336) [x86_64-linux]

-- Control frame information -----------------------------------------------
c:0031 p:---- s:0108 e:000107 CFUNC  :const_get
c:0030 p:0038 s:0104 e:000103 BLOCK  /mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/const_get_spec.rb:34 [FINISH]
c:0029 p:---- s:0101 e:000100 CFUNC  :instance_eval
c:0028 p:0014 s:0098 e:000097 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:69
c:0027 p:0017 s:0092 e:000091 BLOCK  /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:179 [FINISH]
c:0026 p:---- s:0089 e:000088 IFUNC
c:0025 p:---- s:0087 e:000086 CFUNC  :each
c:0024 p:---- s:0085 e:000084 CFUNC  :all?
c:0023 p:0044 s:0082 e:000081 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:179
c:0022 p:0077 s:0076 e:000075 BLOCK  /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:208
c:0021 p:0004 s:0072 e:000071 BLOCK  /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:239 [FINISH]
c:0020 p:---- s:0070 e:000069 CFUNC  :times
c:0019 p:0017 s:0067 e:000066 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:238
c:0018 p:0011 s:0064 e:000063 BLOCK  /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:200 [FINISH]
c:0017 p:---- s:0061 e:000060 CFUNC  :each
c:0016 p:0091 s:0058 e:000057 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:199
c:0015 p:0087 s:0055 e:000054 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:38
c:0014 p:0030 s:0048 e:000047 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/object.rb:11
c:0013 p:0045 s:0041 e:000c28 TOP    /mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/const_get_spec.rb:4 [FINISH]
c:0012 p:---- s:0039 e:000038 CFUNC  :load
c:0011 p:0014 s:0035 e:000034 BLOCK  /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:57 [FINISH]
c:0010 p:---- s:0033 e:000032 CFUNC  :instance_eval
c:0009 p:0014 s:0030 e:000029 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:69
c:0008 p:0061 s:0024 e:0000e0 BLOCK  /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:57 [FINISH]
c:0007 p:---- s:0021 e:000020 CFUNC  :each
c:0006 p:0042 s:0018 e:001450 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:51
c:0005 p:0015 s:0014 e:000013 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:43
c:0004 p:0047 s:0011 e:000010 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/commands/mspec-run.rb:91
c:0003 p:0077 s:0008 e:000007 METHOD /mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/utils/script.rb:218
c:0002 p:0047 s:0004 e:000ac8 EVAL   /mnt/sdb1/ruby/trunk/spec/mspec/bin/mspec-run:8 [FINISH]
c:0001 p:0000 s:0002 e:0007f8 TOP    [FINISH]

/mnt/sdb1/ruby/trunk/spec/mspec/bin/mspec-run:8:in <main>'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/utils/script.rb:218:inmain'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/commands/mspec-run.rb:91:in run'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:43:inprocess'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:51:in files'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:51:ineach'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:57:in block in files'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:69:inprotect'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:69:in instance_eval'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:57:inblock (2 levels) in files'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:57:in load'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/const_get_spec.rb:4:in'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/object.rb:11:in describe'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:38:indescribe'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:199:in process'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:199:ineach'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:200:in block in process'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:238:inrepeat'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:238:in times'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:239:inblock in repeat'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:208:in block (2 levels) in process'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:179:inprotect'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:179:in all?'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:179:ineach'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/context.rb:179:in block in protect'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:69:inprotect'
/mnt/sdb1/ruby/trunk/spec/mspec/lib/mspec/runner/mspec.rb:69:in instance_eval'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/const_get_spec.rb:34:inblock (2 levels) in '
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/const_get_spec.rb:34:in `const_get'

-- C level backtrace information -------------------------------------------
make: *** [test-rubyspec] □□□□□□□□□□□□□□□
```

#25 - 10/27/2012 11:23 AM - Anonymous

On Sat, Oct 27, 2012 at 09:11:51AM +0900, ko1 (Koichi Sasada) wrote:

> Issue #5690 has been updated by ko1 (Koichi Sasada).

I got SEGV.
could you run "make test-rubyspec" on your environment?

I can't seem to run make test-rubyspec in my environment. It always
does this:

```
[aaron@higgins ruby (trunk)]$ make test-rubyspec
./miniruby -I./lib -I. -I.ext/common  ./tool/runruby.rb --extout=.ext  -- --disable-gems ./spec/mspec/bin/mspec run -B ./spec/default.mspec
ruby 2.0.0dev (2012-10-27 trunk 37339) [x86_64-darwin12.2.0]
/Users/aaron/git/ruby/spec/mspec/lib/mspec/guards/guard.rb:156:in block in os?': uninitialized constant SpecGuard::Config (NameError)
from /Users/aaron/git/ruby/spec/mspec/lib/mspec/guards/guard.rb:155:ineach'
from /Users/aaron/git/ruby/spec/mspec/lib/mspec/guards/guard.rb:155:in any?'
from /Users/aaron/git/ruby/spec/mspec/lib/mspec/guards/guard.rb:155:inos?'
from /Users/aaron/git/ruby/spec/mspec/lib/mspec/guards/platform.rb:22:in `block in match?'
```

I'm not sure why. Anyway, I can run mspec by hand, and this should be
fixed in r37340.

--
Aaron Patterson
http://tenderlovemaking.com/

#26 - 11/02/2012 05:04 AM - tenderlovemaking (Aaron Patterson)

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

I committed this in r37340 and r37335

#27 - 04/26/2016 04:50 AM - nobu (Nobuyoshi Nakada)

- Related to Feature #12319: `Module#const_get` does not accept symbol with nested name added

#28 - 04/26/2016 04:56 AM - nobu (Nobuyoshi Nakada)

- Description updated

Files

noname	500 Bytes	12/01/2011	Anonymous
bug5690.pdf	615 KB	07/01/2012	tenderlovemaking (Aaron Patterson)
noname	500 Bytes	07/02/2012	Anonymous
p2c.diff	4.86 KB	10/24/2012	tenderlovemaking (Aaron Patterson)