Ruby master - Feature #3346

__DIR__ revisted

05/26/2010 08:45 PM - trans (Thomas Sawyer)

Status:	Closed
Priority:	Normal
Assignee:	yhara (Yutaka HARA)
Target version:	2.0.0

Description

=begin
I'd like to know why **DIR** was rejected? I use File.dirname(**FILE**) all the time, and I frequently see others do so as well. #relative_require is helpful but it covers only one specific use case --and probably not the most common one. I am often using File.dirname(**FILE**) in build scripts, when I am loading examples for tests, and when I load output templates or other pluggable modules that reside relative to my code. For something so common, having to clutter my code with a 22 character sequence, when an perfectly obvious 7 character sequence would do semms very uncharacteristic of Ruby, which is usually quite concise. Indeed, it is not uncommon to see code that defines a constant such as DIR = File.dirname(**FILE**) when it will be used more than once because it quickly becomes an eye-sore. For these reasons I hope you will reconsider the earlier rejection.
=end

Related issues:

Related to Ruby master - Bug #7729: __dir__ returns a absolute dir path	**Rejected**	**01/23/2013**
Is duplicate of Ruby master - Feature #1961: Kernel#__dir__	**Closed**	**08/19/2009**
Has duplicate Ruby master - Bug #7975: Why __dir__, not __DIR__	**Rejected**	**02/27/2013**
Has duplicate Ruby master - Feature #8098: Add __DIR__ to compliment __FILE__	**Rejected**	**03/15/2013**

Associated revisions

Revision 805b08f2 - 11/03/2012 01:37 AM - nari

- eval.c (f_current_dirname): add the new method for Kernel.
  This method almotst same as File.dirname(**FILE**). One
  different behavior is it returns nil when **FILE** returns nil.
  [Feature #3346]

- NEWS: ditto

- test/ruby/test_method.rb: related test.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@37432 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 37432 - 11/03/2012 01:37 AM - nari

- eval.c (f_current_dirname): add the new method for Kernel.
  This method almotst same as File.dirname(**FILE**). One
  different behavior is it returns nil when **FILE** returns nil.
  [Feature #3346]

- NEWS: ditto

- test/ruby/test_method.rb: related test.

Revision 37432 - 11/03/2012 01:37 AM - nari

- eval.c (f_current_dirname): add the new method for Kernel.
  This method almotst same as File.dirname(**FILE**). One

different behavior is it returns nil when **FILE** returns nil. [Feature #3346]

- NEWS: ditto
- test/ruby/test_method.rb: related test.

Revision 37432 - 11/03/2012 01:37 AM - nari

- eval.c (f_current_dirname): add the new method for Kernel. This method almotst same as File.dirname(**FILE**). One different behavior is it returns nil when **FILE** returns nil. [Feature #3346]
- NEWS: ditto
- test/ruby/test_method.rb: related test.

Revision 37432 - 11/03/2012 01:37 AM - nari

- eval.c (f_current_dirname): add the new method for Kernel. This method almotst same as File.dirname(**FILE**). One different behavior is it returns nil when **FILE** returns nil. [Feature #3346]
- NEWS: ditto
- test/ruby/test_method.rb: related test.

Revision 37432 - 11/03/2012 01:37 AM - nari

- eval.c (f_current_dirname): add the new method for Kernel. This method almotst same as File.dirname(**FILE**). One different behavior is it returns nil when **FILE** returns nil. [Feature #3346]
- NEWS: ditto
- test/ruby/test_method.rb: related test.

Revision 37432 - 11/03/2012 01:37 AM - nari

- eval.c (f_current_dirname): add the new method for Kernel. This method almotst same as File.dirname(**FILE**). One different behavior is it returns nil when **FILE** returns nil. [Feature #3346]
- NEWS: ditto
- test/ruby/test_method.rb: related test.

History

#1 - 05/26/2010 08:59 PM - naruse (Yui NARUSE)

=begin
I like **DIR**.
=end

#2 - 05/26/2010 09:13 PM - shyouhei (Shyouhei Urabe)

=begin
Which is "the earlier rejection"? I see nothing (though this request is not accepted yet).
=end

#3 - 05/26/2010 09:17 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:30436] [Feature #3346] **DIR** revisted"
on May.26,2010 20:59:48, redmine@ruby-lang.org wrote:

> I like **DIR**.

I like Kernel#**dir**.

Regards,
--
U.Nakamura usa@garbagecollect.jp

=end

#4 - 05/26/2010 09:53 PM - rkh (Konstantin Haase)

=begin

> I like Kernel#**dir**.

But then there should be Kernel#**file**.

Konstantin
=end

#5 - 05/26/2010 10:03 PM - Eregon (Benoit Daloze)

=begin
Hi,
On 26 May 2010 14:17, U.Nakamura usa@garbagecollect.jp wrote:

> Hello,
>
> In message "[ruby-core:30436] [Feature #3346] **DIR** revisted"
> on May.26,2010 20:59:48, redmine@ruby-lang.org wrote:
>
>> I like **DIR**.
>
> I like Kernel#**dir**.
>
> ## Regards,
>
> U.Nakamura usa@garbagecollect.jp
>
> I prefer **DIR**.

(as a keyword,
because it's clearly the same family with **FILE**,
and it's a constant (so in uppercase))

My first use would be things like Dir[File.join(**DIR**,'*.rb')].each
*{ |f| require f }
That's only because I am sometimes too lazy to "require_relative" all
the files under the same dir.
But maybe that's not the best way to do it, tough.*

Currently, there is also another possibility, which is kind of a hack:
Dir[File.expand_path("../.rb",***FILE***)].each { |f| require f }*

Just some ideas in the air for this case:

- require_relatives
- require_relative '*.rb'
- require_relative '*' (then assume we want only .rb files)
- require_relative (then the meaning would be "require the relative stuff, what is relative to this") The 2nd and 3rd one obviously more flexible as they accept a glob matching pattern.

Anyway, there are still many cases in which **DIR** would be useful,
even if that's kind of low-level stuff.

Regards,
B.D.

=end

#6 - 05/27/2010 01:11 AM - rogerdpack (Roger Pack)

=begin

> Which is "the earlier rejection"? I see nothing (though this request is not accepted yet).

#1961 wasn't rejected, just not accepted/committed yet.

For me, the "ideal" would be a compiler time constant **FILE**, but if it's a method instead, then name it **file** just to not confuse people.

-rp
=end

#7 - 05/27/2010 07:11 AM - matz (Yukihiro Matsumoto)

=begin
Hi,

In message "Re: [ruby-core:30434] [Feature #3346] **DIR** revisted"
on Wed, 26 May 2010 20:45:55 +0900, Thomas Sawyer redmine@ruby-lang.org writes:

|I'd like to know why **DIR** was rejected?

I haven't rejected. I agreed with the basic idea, and proposed to add
Kernel#**dir**; a method not to add a new keyword, a lowercase name to
clarify it's implemented by a method. But some didn't agree with me,
so I haven't added.

```
                            matz.
```

=end

#8 - 05/27/2010 12:04 PM - zenspider (Ryan Davis)

=begin

On May 26, 2010, at 09:11 , Roger Pack wrote:

> #1961 wasn't rejected, just not accepted/committed yet.

wow... it is just like Seattle politics... if we just keep voting OVER and OVER on the same issue, eventually it'll get accepted!

You should run for Seattle City Council.

Don't forget **DIR_DIR** and **DIR_DIR_DIR**... I'd use them lots.

=end

#9 - 08/24/2010 11:49 PM - trans (Thomas Sawyer)

=begin
Good to hear. **dir** works for me if that is deemed better, though I concur that it inclines me to expect **FILE** to be **file** too. Also, I do not recall who suggested it, but an optional join parameter was once suggested too, e.g.

dir('foo')

Which would be equivalent to:

File.expand_path(File.join(File.dirname(**FILE**),'foo'))

Such a parameter would seem more appropriate to a lowercase method name. Also perhaps it would quell the "**DIR_DIR**" gripes, which I don't earnestly get, but if they should be taken seriously...

dir('..')

=end

#10 - 03/25/2012 01:57 PM - mame (Yusuke Endoh)

- Description updated

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

#11 - 03/25/2012 02:27 PM - matz (Yukihiro Matsumoto)

UNIX style directory hierarchy prepare a directory to put everything relate to the app/lib. I hesitate to add a method/directive that encourage file system convention apparently against UNIX style.

#12 - 03/25/2012 10:20 PM - trans (Thomas Sawyer)

Ruby does not provide way to get root of app/lib, and current working directory is not always root of applicable app/lib, so that's not always helpful way.

Sometimes practical choice outweighs theoretical ideal. File.dirname(__FILE__) is exhaustively used idiom.

Consider another use case:

myapp/
lib/
myapp/
template.rb
templates/
foo.erb
bar.erb

In template.rb there will be something like:

def read_template(name)
file = File.dirname(**FILE**) + "/templates/#{name}.rb"
raise "no such template -- #{name}" unless File.exist?(file)
File.read(file)
end

#13 - 06/30/2012 02:13 AM - yhara (Yutaka HARA)

- File 3346.pdf added

Adding presentation slide for the feature request meeting ([ruby-dev:45708])

#14 - 07/02/2012 01:33 AM - mame (Yusuke Endoh)

Yhara-san, your slide is received. Thank you!

But, matz has already accepted this proposal basically.
The remaining issue is its API (including name). Right?
If so, I'm afraid if just listing candidates will make no progress.
It may be good to choose one target yourself and make it authorized by matz.

Anyway, your slide will be presented at the meeting.
Matz may choose one in his preference. Thanks again!

--
Yusuke Endoh mame@tsg.ne.jp

#15 - 07/06/2012 06:53 PM - naruse (Yui NARUSE)

I was against the name **DIR** for years,
but now I'm for **dir** if **file** and **line** method is also added.

#16 - 07/12/2012 05:37 AM - rogerdpack (Roger Pack)

I'm for **dir** or **DIR** since I use File.dirname(**FILE**) a lot. Barely prefer **DIR** since **FILE** already exists so it keeps parity with it.

#17 - 07/24/2012 10:39 PM - mame (Yusuke Endoh)

- Assignee changed from matz (Yukihiro Matsumoto) to nobu (Nobuyoshi Nakada)

Thomas Sawyer and Yutaka Hara,

Matz determined Kernel#**dir** (as a method).

Yhara-san, do you have a patch?

--
Yusuke Endoh mame@tsg.ne.jp

#18 - 08/03/2012 08:05 PM - matz (Yukihiro Matsumoto)

- Status changed from Assigned to Closed

#19 - 08/03/2012 08:22 PM - shyouhei (Shyouhei Urabe)

- Category set to core

- Status changed from Closed to Assigned

- Assignee changed from nobu (Nobuyoshi Nakada) to yhara (Yutaka HARA)

- Target version set to 2.0.0

This one is alive.

#20 - 08/03/2012 08:30 PM - Eregon (Benoit Daloze)

mame (Yusuke Endoh) wrote:

> Thomas Sawyer and Yutaka Hara,
>
> Matz determined Kernel#**dir** (as a method).
>
> Yhara-san, do you have a patch?

Will **dir** always be an absolute path, or just File.dirname(**FILE**)?

File.dirname(**FILE**) is relative (often ".") when **FILE** is $0.
It would mean the path needs to be expanded if it is relative (using File.expand_path), to keep the right location if the current working directory changes.

#21 - 09/14/2012 04:32 PM - rogerdpack (Roger Pack)

definitely recommend full path to avoid possible confusion, like we used to have with 1.8.x not using the full path for $: but that's just my opinion...

#22 - 10/27/2012 11:29 PM - yhara (Yutaka HARA)

mame (Yusuke Endoh) wrote:

> Thomas Sawyer and Yutaka Hara,
>
> Matz determined Kernel#**dir** (as a method).
>
> Yhara-san, do you have a patch?

Sorry, I forgot to reply. I don't have a patch right now.

#23 - 11/01/2012 11:02 PM - authorNari (Narihiro Nakamura)

- File __dir__.patch added

Hi.

yhara (Yutaka HARA) wrote:

> mame (Yusuke Endoh) wrote:
>
>> Thomas Sawyer and Yutaka Hara,
>>
>> Matz determined Kernel#**dir** (as a method).
>>
>> Yhara-san, do you have a patch?
>
> Sorry, I forgot to reply. I don't have a patch right now.

I've created a patch for Kernel#**dir**. How about it?

#24 - 11/03/2012 10:37 AM - authorNari (Narihiro Nakamura)

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r37432.
Thomas, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- eval.c (f_current_dirname): add the new method for Kernel.
  This method almotst same as File.dirname(**FILE**). One
  different behavior is it returns nil when **FILE** returns nil.
  [Feature #3346]
- NEWS: ditto
- test/ruby/test_method.rb: related test.

Files

3346.pdf	41.5 KB	06/30/2012	yhara (Yutaka HARA)
__dir__.patch	1.38 KB	11/01/2012	authorNari (Narihiro Nakamura)