Ruby master - Feature #11302

Dir.entries and Dir.foreach without [".", ".."]

06/24/2015 06:24 AM - naruse (Yui NARUSE)

Status:	Closed	
Priority:	Normal	
Assignee:	matz (Yukihiro Matsumoto)	
Target version:		

Description

Dir.entries returns an array of its content with "." and "..".
But as far as I met, almost all cases don't need them.

How about adding such new method or options?

Related issues:

Related to Ruby master - Feature #13789: Dir - methods	**Rejected**
Related to Ruby master - Feature #13969: Dir#each_child	**Closed**

Associated revisions

Revision 944c455b - 05/25/2017 02:50 AM - nobu (Nobuyoshi Nakada)

dir.c: Dir.each_child and Dir.children

- dir.c (dir_s_each_child, dir_s_children): Dir.each_child and Dir.children which are similar to Dir.foreach and Dir.entries respectively, except to exclude "." and "..". [Feature #11302]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@58879 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 58879 - 05/25/2017 02:50 AM - nobu (Nobuyoshi Nakada)

dir.c: Dir.each_child and Dir.children

- dir.c (dir_s_each_child, dir_s_children): Dir.each_child and Dir.children which are similar to Dir.foreach and Dir.entries respectively, except to exclude "." and "..". [Feature #11302]

Revision 58879 - 05/25/2017 02:50 AM - nobu (Nobuyoshi Nakada)

dir.c: Dir.each_child and Dir.children

- dir.c (dir_s_each_child, dir_s_children): Dir.each_child and Dir.children which are similar to Dir.foreach and Dir.entries respectively, except to exclude "." and "..". [Feature #11302]

Revision 58879 - 05/25/2017 02:50 AM - nobu (Nobuyoshi Nakada)

dir.c: Dir.each_child and Dir.children

- dir.c (dir_s_each_child, dir_s_children): Dir.each_child and Dir.children which are similar to Dir.foreach and Dir.entries respectively, except to exclude "." and "..". [Feature #11302]

Revision b2996b30 - 09/15/2017 05:00 PM - naruse (Yui NARUSE)

Find.find -> Use Dir.children instead of Dir.entries

Dir.children is available since Feature #11302.
Find.find can use of the new list (having no '.' neither '..' entries),
making now superflous an if statement.

This change can improve the performance of Find.find when the path
has lots of entries (thousands?).

https://bugs.ruby-lang.org/issues/11302
patched by Espartaco Palma esparta@gmail.com
https://github.com/ruby/ruby/pull/1697 fix GH-1697
[Feature #13896]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@59926 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 59926 - 09/15/2017 05:00 PM - naruse (Yui NARUSE)

Find.find -> Use Dir.children instead of Dir.entries

Dir.children is available since Feature #11302.
Find.find can use of the new list (having no '.' neither '..' entries),
making now superflous an if statement.

This change can improve the performance of Find.find when the path
has lots of entries (thousands?).

https://bugs.ruby-lang.org/issues/11302
patched by Espartaco Palma esparta@gmail.com
https://github.com/ruby/ruby/pull/1697 fix GH-1697
[Feature #13896]

Revision 59926 - 09/15/2017 05:00 PM - naruse (Yui NARUSE)

Find.find -> Use Dir.children instead of Dir.entries

Dir.children is available since Feature #11302.
Find.find can use of the new list (having no '.' neither '..' entries),
making now superflous an if statement.

This change can improve the performance of Find.find when the path
has lots of entries (thousands?).

https://bugs.ruby-lang.org/issues/11302
patched by Espartaco Palma esparta@gmail.com
https://github.com/ruby/ruby/pull/1697 fix GH-1697
[Feature #13896]

Revision 59926 - 09/15/2017 05:00 PM - naruse (Yui NARUSE)

Find.find -> Use Dir.children instead of Dir.entries

Dir.children is available since Feature #11302.
Find.find can use of the new list (having no '.' neither '..' entries),
making now superflous an if statement.

This change can improve the performance of Find.find when the path
has lots of entries (thousands?).

https://bugs.ruby-lang.org/issues/11302
patched by Espartaco Palma esparta@gmail.com
https://github.com/ruby/ruby/pull/1697 fix GH-1697
[Feature #13896]

Revision 37c08fad - 11/07/2018 03:55 PM - hsbt (Hiroshi SHIBATA)

Dir.children is available since Feature #11302. FileUtils uses
Dir.each on an internal method encapsulated on a private class
Entry_#entry, having no '.' neither '..' entries would make
now superfluous a chained reject filtering.

This change can improve the performance of these FileUtils
methods when the provided path covers thousands of files or
directories:

- chmod_R
- chown_R
- remove_entry
- remove_entry_secure
- rm_r
- remove_dir
- copy_entry

Related: Feature #13896 https://bugs.ruby-lang.org/issues/13896

[Feature #14109][Fix GH-1754]

Co-Authored-By: esparta esparta@gmail.com

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65610 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 65610 - 11/07/2018 03:55 PM - hsbt (Hiroshi SHIBATA)

Dir.children is available since Feature #11302. FileUtils uses
Dir.each on an internal method encapsulated on a private class
Entry_#entry, having no '.' neither '..' entries would make
now superfluous a chained reject filtering.

This change can improve the performance of these FileUtils
methods when the provided path covers thousands of files or
directories:

- chmod_R
- chown_R
- remove_entry
- remove_entry_secure
- rm_r
- remove_dir
- copy_entry

Related: Feature #13896 https://bugs.ruby-lang.org/issues/13896

[Feature #14109][Fix GH-1754]

Co-Authored-By: esparta esparta@gmail.com

Revision 65610 - 11/07/2018 03:55 PM - hsbt (Hiroshi SHIBATA)

Dir.children is available since Feature #11302. FileUtils uses
Dir.each on an internal method encapsulated on a private class
Entry_#entry, having no '.' neither '..' entries would make
now superfluous a chained reject filtering.

This change can improve the performance of these FileUtils
methods when the provided path covers thousands of files or
directories:

- chmod_R
- chown_R
- remove_entry
- remove_entry_secure
- rm_r
- remove_dir
- copy_entry

Related: Feature #13896 https://bugs.ruby-lang.org/issues/13896

[Feature #14109][Fix GH-1754]

Co-Authored-By: esparta esparta@gmail.com

History

#1 - 06/24/2015 06:41 AM - nobu (Nobuyoshi Nakada)

Candidates for the methods or options?

I prefer a same option for both methods, but no concrete idea.

#2 - 06/24/2015 03:44 PM - red (Arnaud Rouyer)

Nobuyoshi Nakada wrote:

> Candidates for the methods or options?
>
> I prefer a same option for both methods, but no concrete idea.

Dir.foreach and Dir.entries both support a second hash argument for options: as of 2.2.2, the docs only mention the :encoding key in the options hash.

Basing myself on the GNU ls util, I propose supporting an :ignore key in the optional hash argument.

We could have an API similar to this:

```
$ ls -a
.
```

```
..
.hidden_file
directory
file.bin

$ irb

irb:001> Dir.entries('.')
 => [".", "..", ".hidden_file", "directory", "file.bin"]
irb:002> Dir.entries('.', ignore: :almost_all)  # almost_all option name taken from GNU ls option name
 => [".hidden_file", "directory", "file.bin"]   # http://git.savannah.gnu.org/cgit/coreutils.git/tree/src/ls.c
#n4784
irb:003> Dir.entries('.', ignore: :directories)
=> [".hidden_file", "file.bin"]
irb:004> Dir.entries('.', ignore: :hidden)
=> ["directory", "file.bin"]

# Fancy proposal
irb:005> Dir.entries('.', ignore: /o/)
=> [".", "..", ".hidden_file", "file.bin"]
```

Edit after a closer look:

Internally, Dir.entries and Dir.foreach both call Dir.new and use the resulting Dir object's #to_a and #each methods respectively to return an array/an enumerator. If we want to go down this path, these options have to be supported in Dir.new, stored in the Dir instance and reused in the #each enumerator to filter out ignored entries.

#3 - 03/19/2017 05:10 AM - olivierlacan (Olivier Lacan)

red (Arnaud Rouyer) wrote:

> Basing myself on the GNU ls util, I propose supporting an :ignore key in the optional hash argument.

I very much like this. I just ran into this issue myself today having to remove . and .. from Dir.entries output.

I don't think the ignore option accepting a regex is fancy at all, it makes a ton of sense. An array should also be acceptable considering that Dir.entries('.', ignore: %w[. ..]) would become equivalent to:

```
Dir.entries('.') - %w[. ..]
```

I find it quite elegant, and certainly a lot more discoverable than GNU ls style arguments. :-)

#4 - 04/19/2017 02:06 AM - nobu (Nobuyoshi Nakada)

red (Arnaud Rouyer) wrote:

> Basing myself on the GNU ls util, I propose supporting an :ignore key in the optional hash argument.
>
> ```
> irb:002> Dir.entries('.', ignore: :almost_all)  # almost_all option name taken from GNU ls option name
> ```

ignore: :almost_all seems like that almost all files will be ignored and only '.' and '..' will be returned.

#5 - 05/19/2017 09:01 AM - akr (Akira Tanaka)

There is Pathname#children and Pathname#each_child.

How about Dir.children and Dir.each_child ?

#6 - 05/19/2017 09:06 AM - shyouhei (Shyouhei Urabe)

+1 for Dir.children

#7 - 05/19/2017 09:08 AM - matz (Yukihiro Matsumoto)

Sounds good.

Matz.

#8 - 05/25/2017 02:50 AM - nobu (Nobuyoshi Nakada)

- Status changed from Assigned to Closed

Applied in changeset trunk|r58879.

dir.c: Dir.each_child and Dir.children

- dir.c (dir_s_each_child, dir_s_children): Dir.each_child and Dir.children which are similar to Dir.foreach and Dir.entries respectively, except to exclude "." and "..". [Feature #11302]

#9 - 09/14/2017 09:05 AM - Eregon (Benoit Daloze)

- Related to Feature #13789: Dir - methods added

#10 - 09/14/2017 09:07 AM - Eregon (Benoit Daloze)

The decision here is surprising given https://bugs.ruby-lang.org/issues/13789#note-3
but nevertheless I'm very happy this got accepted.

#11 - 10/04/2017 11:11 AM - znz (Kazuhiro NISHIYAMA)

- Related to Feature #13969: Dir#each_child added