Ruby - Bug #12814

Pathname#each_child(false) produces unusable file paths

10/06/2016 11:38 AM - tao (Mr. Tao)

Status:	Rejected		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.3.1p112 (2016-04-26 revision 54768) [x86_64-darwin15]	**Backport:**	2.1: UNKNOWN, 2.2: UNKNOWN, 2.3: UNKNOWN

Description

I tried to iterate over items in a directory with each_child testing each item whether it is a directory or not with directory?. This was working just fine until I added **false** as an argument to each_child.

```
Pathname.new('.').each_child { |f| p f.realpath } # works as expected
Pathname.new('.').each_child(false) { |f| p f.realpath } # throws an error
```

As per Ruby doc "By default, the yielded pathnames will have enough information to access the files.", however pathnames yielded with with_directory=false are completely useless as file paths.

History

#1 - 10/06/2016 12:56 PM - nobu (Nobuyoshi Nakada)

- Description updated

- Status changed from Open to Feedback

I can't reproduce it.
What exception raised exactly?

#2 - 10/06/2016 02:10 PM - tao (Mr. Tao)

Sorry, I over edited my post. Argument to pathname is some existing folder, even '..' produces the error.

```
>> Pathname.new('..').each_child(false) { |f| p f.realpath }

Errno::ENOENT: No such file or directory @ realpath_rec - <some_path_here>
 from (irb):10:in `realpath'
 from (irb):10:in `realpath'
 from (irb):10:in `block in irb_binding'
 from /usr/local/Cellar/ruby/2.3.1/lib/ruby/2.3.0/pathname.rb:490:in `each'
 from /usr/local/Cellar/ruby/2.3.1/lib/ruby/2.3.0/pathname.rb:490:in `each_child'
 from (irb):10
 from /usr/local/bin/irb:11:in `<main>'
```

#3 - 10/11/2016 01:12 AM - shyouhei (Shyouhei Urabe)

It's true you can't infer a realpath from with_directory=false-yielded file paths. Yes. But isn't it intentional? You requested to cut where the path was from. Pretty natural you can no longer infer its origin.

The RDoc says it guarantees to include info enough to infer the realpath "By default", and you are requesting something not default. I think the behavour is by design.

#4 - 10/15/2016 04:29 PM - tao (Mr. Tao)

I see… Still I can't decide whether it is for good or not. What is the purpose of such a path then? Shouldn't a String object instead of a Pathname be returned in such a case?

In my scenario I traverse directories and process some of them. If each_child(false) still contained enough information about it's relative position in directory tree it would allow me make code below somewhat simpler and more readable. Perhaps augmenting the Pathname class with this information might be beneficial.

```
…
    user_path.each_child do |mailbox_path|
        process_mailbox mailbox_path if mailbox_path.directory?
…
def process_mailbox(mailbox_path)
```

```
…
    mailbox_path.find do |f|
          if f.directory?
            process_folder(f)
            next
…
def process_folder(folder)
    @user.mailboxes.find_or_create_by_path((folder.relative_path_from @user_path).to_s.split(File::SEPARATOR))
…
```

#5 - 11/05/2016 07:25 AM - akr (Akira Tanaka)

- Status changed from Feedback to Rejected

Pathname#each_child is designed that easier access for child files by default (with_directory=true).
But it is also possible to obtain filenames only (with_directory=false) for applications which
remember the directory in the application (or application don't need the directory).

If you need directory in pathname, you can use with_directory=true.
If you don't want directory in pathname, you can use with_directory=false.