Ruby - Feature #8929

CSV.foreach(filename) without block returns failing Enumerator

09/20/2013 06:24 AM - martinjos (Martin Sidaway)

Status:	Closed	
Priority:	Normal	
Assignee:	JEG2 (James Gray)	
Target version:		

Description

CSV.foreach(filename) {|entry| p entry } => works
CSV.foreach(filename).to_a => fails

It gives the following error:

IOError: closed stream
from /home/martin/.rvm/rubies/ruby-2.0.0-p247/lib/ruby/2.0.0/csv.rb:1776:in gets' from
/home/martin/.rvm/rubies/ruby-2.0.0-p247/lib/ruby/2.0.0/csv.rb:1776:in block in shift'
from /home/martin/.rvm/rubies/ruby-2.0.0-p247/lib/ruby/2.0.0/csv.rb:1774:in loop' from
/home/martin/.rvm/rubies/ruby-2.0.0-p247/lib/ruby/2.0.0/csv.rb:1774:in shift'
from /home/martin/.rvm/rubies/ruby-2.0.0-p247/lib/ruby/2.0.0/csv.rb:1716:in each' from (irb):7:in each'
from (irb):7:in `to_a'
(...)

Associated revisions

Revision 28204c67ed6e55309cdf05e5e20f1f7e59e85e96 - 10/04/2013 12:32 AM - nobu (Nobuyoshi Nakada)

csv.rb: foreach enumerator

- lib/csv.rb (CSV.foreach): support enumerator. based on a patch by
  Hanmac (Hans Mackowiak) at [ruby-core:57643]. [ruby-core:57283]
  [Feature #8929]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@43135 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 28204c67 - 10/04/2013 12:32 AM - nobu (Nobuyoshi Nakada)

csv.rb: foreach enumerator

- lib/csv.rb (CSV.foreach): support enumerator. based on a patch by
  Hanmac (Hans Mackowiak) at [ruby-core:57643]. [ruby-core:57283]
  [Feature #8929]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@43135 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 09/20/2013 07:04 AM - martinjos (Martin Sidaway)

(I thought it had failed to send this message...)

#2 - 09/20/2013 07:28 AM - martinjos (Martin Sidaway)

I can see that it is tricky, because a normal enumerator created by calling CSV#each shouldn't necessarily close the file (as the user may want to call #rewind), but when you use CSV.foreach you have no reference to the CSV object, and you don't want the file handle to be left open.

However, I don't see why, in principle, CSV.foreach(filename) as an Enumerator shouldn't be supported. It has clear, well-defined semantics: it just has to close the file automatically after yielding the last entry.

Also, this would be consistent with, amongst other things, File.foreach(filename).

The present behaviour occurs because self.foreach() uses the block form of self.open(), which ensures the csv object is closed after yielding it. It then calls csv.each(&block) inside the open block, which is okay if a block was passed in to self.foreach, but if not, the no-block form of each() simply calls to_enum (from Object). This enumerator then gets passed out of the open block (becoming invalidated in the process), and out of self.foreach.

In other words:

class CSV

def self.foreach(path, options = Hash.new, &block)
open(path, options) do |csv|
csv.each(&block)
end
end

def self.open(*args)
(...)
if block_given?
begin
yield csv
ensure
csv.close
end
else
csv
end
end

def each
if block_given?
while row = shift
yield row
end
else
to_enum
end
end

end

A solution might be to create an alternative version of CSV#each that calls CSV#close after the last entry:

def each_closing
if block_given?
begin
while row = shift
yield row
end
ensure
close
end
else
to_enum(**method**) # **method** returns :each_closing
end
end

Then CSV.foreach could be:

def self.foreach(path, options = Hash.new, &block)
open(path, options).each_closing(&block)
end

#3 - 09/20/2013 05:48 PM - jwille (Jens Wille)

=begin
why not simply return an Enumerator from foreach?

class CSV

```
def self.foreach(path, options = Hash.new, &block)
  if block_given?
    open(path, options) do |csv|
      csv.each(&block)
    end
  else
    to_enum(__method__, path, options)
  end
end
```

end

which yields

ary = []; CSV.foreach(filename) {|entry| ary << entry }

ary == CSV.foreach(filename).to_a #=> true
=end

#4 - 09/20/2013 11:16 PM - zzak (zzak _)

- Category set to lib

- Status changed from Open to Feedback

- Assignee set to JEG2 (James Gray)

#5 - 10/04/2013 02:13 AM - Hanmac (Hans Mackowiak)

another sample where the difference is shown:

CSV.foreach('test.csv').with_index { |csv,i| p i } #<< fails
CSV.to_enum(:foreach,'test.csv').with_index { |csv,i| p i } # works

i think the code from jwillie would be nearly the best, but i would use this:

return to_enum(**method**, path, options) unless block_given?

#6 - 10/04/2013 09:30 AM - nobu (Nobuyoshi Nakada)

- Tracker changed from Bug to Feature

#7 - 10/04/2013 09:32 AM - nobu (Nobuyoshi Nakada)

- Status changed from Feedback to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r43135.
Martin, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

---

csv.rb: foreach enumerator

- lib/csv.rb (CSV.foreach): support enumerator. based on a patch by Hanmac (Hans Mackowiak) at [ruby-core:57643]. [ruby-core:57283] [Feature #8929]