Ruby trunk - Feature #4532

[PATCH] add IO#pread and IO#pwrite methods

03/28/2011 02:06 PM - normalperson (Eric Wong)

Status:	Closed
Priority:	Normal
Assignee:	nobu (Nobuyoshi Nakada)
Target version:	2.6

Description

=begin
These methods are useful for safe/concurrent file I/O in
multi-thread/process environments and also fairly standard
nowadays especially in systems supporting pthreads.

pread() is already used internally for IO.copy_stream
=end

Associated revisions

Revision 8109114b - 04/03/2017 12:10 AM - nobu (Nobuyoshi Nakada)

Add IO#pread and IO#pwrite methods

These methods are useful for safe/concurrent file I/O in
multi-thread/process environments and also fairly standard
nowadays especially in systems supporting pthreads.

Based on patches by Avseyev sergey.avseyev@gmail.com at
[ruby-core:79290]. [Feature #4532]

- configure.in: check for pwrite(2). pread() is already used
  internally for IO.copy_stream.
- io.c: implement wrappers for pread(2) and pwrite(2) and expose
  them in IO.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@58240 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 58240 - 04/03/2017 12:10 AM - nobu (Nobuyoshi Nakada)

Add IO#pread and IO#pwrite methods

These methods are useful for safe/concurrent file I/O in
multi-thread/process environments and also fairly standard
nowadays especially in systems supporting pthreads.

Based on patches by Avseyev sergey.avseyev@gmail.com at
[ruby-core:79290]. [Feature #4532]

- configure.in: check for pwrite(2). pread() is already used
  internally for IO.copy_stream.
- io.c: implement wrappers for pread(2) and pwrite(2) and expose
  them in IO.

Revision 58240 - 04/03/2017 12:10 AM - nobu (Nobuyoshi Nakada)

Add IO#pread and IO#pwrite methods

These methods are useful for safe/concurrent file I/O in
multi-thread/process environments and also fairly standard
nowadays especially in systems supporting pthreads.

Based on patches by Avseyev sergey.avseyev@gmail.com at
[ruby-core:79290]. [Feature #4532]

- configure.in: check for pwrite(2). pread() is already used internally for IO.copy_stream.
- io.c: implement wrappers for pread(2) and pwrite(2) and expose them in IO.

Revision 58240 - 04/03/2017 12:10 AM - nobu (Nobuyoshi Nakada)

Add IO#pread and IO#pwrite methods

These methods are useful for safe/concurrent file I/O in multi-thread/process environments and also fairly standard nowadays especially in systems supporting pthreads.

Based on patches by Avseyev sergey.avseyev@gmail.com at [ruby-core:79290]. [Feature #4532]

- configure.in: check for pwrite(2). pread() is already used internally for IO.copy_stream.
- io.c: implement wrappers for pread(2) and pwrite(2) and expose them in IO.

History

#1 - 03/28/2011 09:23 PM - kosaki (Motohiro KOSAKI)

2011/3/28 Eric Wong normalperson@yhbt.net:

> Issue #4532 has been reported by Eric Wong.
>
> ---
>
> Feature #4532: [PATCH] add IO#pread and IO#pwrite methods
> http://redmine.ruby-lang.org/issues/4532
>
> Author: Eric Wong
> Status: Open
> Priority: Normal
> Assignee:
> Category: core
> Target version: 1.9.x
>
> These methods are useful for safe/concurrent file I/O in multi-thread/process environments and also fairly standard nowadays especially in systems supporting pthreads.
>
> pread() is already used internally for IO.copy_stream

Do we really need to introduce new method? Why can't we overload IO.read and IO.write?
too complex?

I agree "offset" argument is useful. But I'm not convinced this API design is best. The description
is too quiet.

Ok, back to meta reviewing comments. All new API proposal need to explain why this way
is best and need to persuade matz.

#2 - 03/29/2011 03:23 AM - normalperson (Eric Wong)

KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

> 2011/3/28 Eric Wong normalperson@yhbt.net:
>
> > Issue #4532 has been reported by Eric Wong.
> >
> > ---
> >
> > Feature #4532: [PATCH] add IO#pread and IO#pwrite methods
> > http://redmine.ruby-lang.org/issues/4532

Author: Eric Wong
Status: Open
Priority: Normal
Assignee:
Category: core
Target version: 1.9.x

These methods are useful for safe/concurrent file I/O in
multi-thread/process environments and also fairly standard
nowadays especially in systems supporting pthreads.

pread() is already used internally for IO.copy_stream

Do we really need to introduce new method? Why can't we overload
IO.read and IO.write?
too complex?

IO#read and IO#write take userspace buffers into account which
makes no sense with pread/pwrite.

I considered overloading IO#sysread and IO#syswrite, but it would be
hard for users to determine whether offset is supported on their
platform.

New methods means IO.method_defined? and IO#respond_to? to be used.

I'm not a fan of throwing NotImplementedError and faking with lseek() +
read()/write() to be even worse since it loses the atomicity guarantee.

I also considered putting the new methods in File instead of IO, but
sysseek is an IO method so I put it in IO.

I agree "offset" argument is useful. But I'm not convinced this API
design is best. The description
is too quiet.

Ok, back to meta reviewing comments. All new API proposal need to
explain why this way
is best and need to persuade matz.

Thanks again for your time!

--
Eric Wong

#3 - 03/29/2011 09:23 PM - kosaki (Motohiro KOSAKI)

2011/3/29 Eric Wong normalperson@yhbt.net:

KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

2011/3/28 Eric Wong normalperson@yhbt.net:

Issue #4532 has been reported by Eric Wong.

Feature #4532: [PATCH] add IO#pread and IO#pwrite methods
http://redmine.ruby-lang.org/issues/4532

Author: Eric Wong
Status: Open
Priority: Normal
Assignee:
Category: core
Target version: 1.9.x

These methods are useful for safe/concurrent file I/O in
multi-thread/process environments and also fairly standard
nowadays especially in systems supporting pthreads.

pread() is already used internally for IO.copy_stream

> Do we really need to introduce new method? Why can't we overload
> IO.read and IO.write?
> too complex?

IO#read and IO#write take userspace buffers into account which
makes no sense with pread/pwrite.

userspace buffer is implementation detail. no?

And, If pread is always behave as binary mode read method, your
documentation is much misleading. IMHO.

- f = File.new("testfile")
- f.pread(16, 0) #=> "This is line one"

> I considered overloading IO#sysread and IO#syswrite, but it would be
> hard for users to determine whether offset is supported on their
> platform.

?? Why?
Do you have any example?

> New methods means IO.method_defined? and IO#respond_to? to be used.

OK, fair point.

> I'm not a fan of throwing NotImplementedError and faking with lseek() +
> read()/write() to be even worse since it loses the atomicity guarantee.

I disagree. I dislike following part of your patch.

```
#ifdef HAVE_PREAD
rb_define_method(rb_cIO, "pread", rb_io_pread, -1);
#endif
```

This is very wrong style for new method. Eventually, *all* users need to call method_defined? before pread. Just NotImplementedError (or lseek emulation) makes much simpler script.

> I also considered putting the new methods in File instead of IO, but
> sysseek is an IO method so I put it in IO.

I agree IO is better.

#4 - 03/30/2011 03:23 AM - normalperson (Eric Wong)

KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

> 2011/3/29 Eric Wong normalperson@yhbt.net:
>
> > KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:
> >
> > > Do we really need to introduce new method? Why can't we overload
> > > IO.read and IO.write?
> > > too complex?
> >
> > IO#read and IO#write take userspace buffers into account which
> > makes no sense with pread/pwrite.
>
> userspace buffer is implementation detail. no?

It's a very important detail. Different processes don't share userspace buffers and Ruby should be able to share buffers with others (non-Ruby processes), so we must only use the kernel cache.

> And, If pread is always behave as binary mode read method, your
> documentation is much misleading. IMHO.
>
> - f = File.new("testfile")
> - f.pread(16, 0) #=> "This is line one"

Yes, I just copied docs from sysread/syswrite :x I will update them
in the next patch I post.

> > I'm not a fan of throwing NotImplementedError and faking with lseek() +
> > read()/write() to be even worse since it loses the atomicity guarantee.
>
> I disagree. I dislike following part of your patch.
>
> #ifdef HAVE_PREAD
> rb_define_method(rb_cIO, "pread", rb_io_pread, -1);
> #endif
>
> This is very wrong style for new method. Eventually, *all* users need to call
> method_defined? before pread. Just NotImplementedError (or lseek emulation)
> makes much simpler script.

OK, shall I resubmit with adding optional offset argument to
sysread/syswrite and raise NotImplementedError?

Thank you again for your comments!

--
Eric Wong

#5 - 04/04/2011 07:12 PM - kosaki (Motohiro KOSAKI)

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

Hi

I personally still dislike pread/pwrite method name, but I give up to argue it because I'm worry about making endless discussion. Let's assign this ticket to matz and hear his opinions. :)

#6 - 10/27/2012 05:58 AM - ko1 (Koichi Sasada)

- Description updated

- Assignee changed from matz (Yukihiro Matsumoto) to kosaki (Motohiro KOSAKI)

- Target version changed from 2.0.0 to 2.6

kosaki-san, could you talk with matz about this ticket?
Please change target to 2.0 if you success to persuade matz.

#7 - 01/27/2017 06:08 PM - avsej (Sergey Avseyev)

Hi everyone, any plans to include pread/pwrite in near future?

#8 - 01/27/2017 08:31 PM - avsej (Sergey Avseyev)

- File 0001-Add-IO-pread-and-IO-pwrite-methods.patch added

I rebased the patch against current trunk, and also made some improvements:

- raise NotImplementedError on platforms, which do not support pread/pwrite
- improved documentation
- fix argument order for IO#pwrite to be consistent with pwrite(2) and IO#pread, the offset should go last
- update tests and function names to follow the same style as other code

#9 - 01/27/2017 08:49 PM - avsej (Sergey Avseyev)

- File 0001-Add-IO-pread-and-IO-pwrite-methods-v3.patch added

The same patch as above, but with typo fixes

#10 - 01/27/2017 08:51 PM - avsej (Sergey Avseyev)

- File deleted (0001-Add-IO-pread-and-IO-pwrite-methods-v3.patch)

#11 - 01/27/2017 08:52 PM - avsej (Sergey Avseyev)

- File 0001-Add-IO-pread-and-IO-pwrite-methods-v3.patch added

#12 - 03/13/2017 08:37 AM - matz (Yukihiro Matsumoto)

- Assignee changed from kosaki (Motohiro KOSAKI) to nobu (Nobuyoshi Nakada)

Agreed. pread/pwrite can be useful in some cases.

Matz.

#13 - 04/03/2017 12:10 AM - nobu (Nobuyoshi Nakada)

- Status changed from Assigned to Closed

Applied in changeset trunk|r58240.

Add IO#pread and IO#pwrite methods

These methods are useful for safe/concurrent file I/O in
multi-thread/process environments and also fairly standard
nowadays especially in systems supporting pthreads.

Based on patches by Avseyev sergey.avseyev@gmail.com at
[ruby-core:79290]. [Feature #4532]

- configure.in: check for pwrite(2). pread() is already used
  internally for IO.copy_stream.
- io.c: implement wrappers for pread(2) and pwrite(2) and expose
  them in IO.

Files

0001-add-IO-pread-and-IO-pwrite-methods.patch	6.22 KB	03/28/2011	normalperson (Eric Wong)
0001-Add-IO-pread-and-IO-pwrite-methods.patch	6.8 KB	01/27/2017	avsej (Sergey Avseyev)
0001-Add-IO-pread-and-IO-pwrite-methods-v3.patch	6.98 KB	01/27/2017	avsej (Sergey Avseyev)