Ruby trunk - Misc #12013

io/wait: allow to wait on readable and writable

01/21/2016 04:53 PM - chucke (Tiago Cardoso)

Status:	Closed	
Priority:	Normal	
Assignee:	nobu (Nobuyoshi Nakada)	

Description

If I have a socket and I want to wait for both read and write events, IO.select is my only co-pilot:

```
IO.select([mysock],[mysock])
```

the beautiful thing about the #wait_readable and #wait_writable methods is that I can have a friendlier way to compose sockets for other event loops which monkey-patching IO.select. One example is celluloid-io, which has its own wrappers around the network sockets classes.

But I think there is a limitation when I want to listen for both reads and writes. See both examples below:

```
IO.select([mysock],[mysock], nil, 30)
# as opposed to
require 'io/wait'
mysock.wait_readable(30) && mysock.wait_writable(30)
```

in the second example, I can wait potentially 60 seconds, instead of the 30 from the first example.

I'm not sure which API it should be, my main reference is the celluloid io reactor api in this case:

```
mysock.wait(:r)
mysock.wait(:w)
mysock.wait(:rw)
```

drawback: there is already a wait method, so backwards compatibility would be gone. or would it? Current arity is 0, which means, one could still alias it to #wait_readable if no argument is passed.

Associated revisions

Revision b58fac9a - 01/24/2016 07:55 AM - nobu (Nobuyoshi Nakada)

wait readable/writable

- ext/io/wait/wait.c (io_wait_readwrite): [EXPERIMENTAL] allow to wait for multiple modes, readable and writable, at once. the arguments may change in the future. [Feature #12013]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@53642 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 53642 - 01/24/2016 07:55 AM - nobu (Nobuyoshi Nakada)

wait readable/writable

- ext/io/wait/wait.c (io_wait_readwrite): [EXPERIMENTAL] allow to wait for multiple modes, readable and writable, at once. the arguments may change in the future. [Feature #12013]

Revision 53642 - 01/24/2016 07:55 AM - nobu (Nobuyoshi Nakada)

wait readable/writable

- ext/io/wait/wait.c (io_wait_readwrite): [EXPERIMENTAL] allow to wait for multiple modes, readable and writable, at once. the arguments may change in the future. [Feature #12013]

Revision 53642 - 01/24/2016 07:55 AM - nobu (Nobuyoshi Nakada)

wait readable/writable

- ext/io/wait/wait.c (io_wait_readwrite): [EXPERIMENTAL] allow to wait for multiple modes, readable and writable, at once. the arguments may change in the future. [Feature #12013]

Revision 53642 - 01/24/2016 07:55 AM - nobu (Nobuyoshi Nakada)

wait readable/writable

- ext/io/wait/wait.c (io_wait_readwrite): [EXPERIMENTAL] allow to wait for multiple modes, readable and writable, at once. the arguments may change in the future. [Feature #12013]

History

#1 - 01/22/2016 05:44 AM - nobu (Nobuyoshi Nakada)

- Description updated

- Status changed from Open to Feedback

- Assignee set to nobu (Nobuyoshi Nakada)

What about:

```
mysock.wait(30, to: [:read, :write])
```

or

```
mysock.wait(30, to: :readwrite)
```

#2 - 01/22/2016 09:57 AM - chucke (Tiago Cardoso)

If the option would be for more verbosity, I'd opt for :readable and :writable, but tell me what you think that it's more readable:

```
socket.wait(:readable, writable)
socket.wait(:read, write)
socket.wait(:rw)

socket.wait(:readable_writable)
socket.wait([:readable, writable]) # allocates one array
```

There are a lot of alternatives, I'd go for less verbosity (:r, :w, :rw), but don't have a strong opinion about the best API.

#3 - 01/24/2016 07:54 AM - nobu (Nobuyoshi Nakada)

- Status changed from Feedback to Closed

Applied in changeset r53642.

---

wait readable/writable

- ext/io/wait/wait.c (io_wait_readwrite): [EXPERIMENTAL] allow to wait for multiple modes, readable and writable, at once. the arguments may change in the future. [Feature #12013]

#4 - 01/24/2016 08:41 AM - normalperson (Eric Wong)

Regarding r53642, how about we limit the symbols to just :r/:w/:rw?

Fewer ways to accomplish the same thing reduces cognitive overhead
for code reviewers; and smaller object size helps everyone.

```
--- a/ext/io/wait/wait.c
+++ b/ext/io/wait/wait.c
@@ -172,30 +172,12 @@ wait_mode_sym(VALUE mode)
     if (mode == ID2SYM(rb_intern("r"))) {
    return RB_WAITFD_IN;
     }
-    if (mode == ID2SYM(rb_intern("read"))) {
-   return RB_WAITFD_IN;
-    }
-    if (mode == ID2SYM(rb_intern("readable"))) {
-   return RB_WAITFD_IN;
-    }
     if (mode == ID2SYM(rb_intern("w"))) {
    return RB_WAITFD_OUT;
     }
-    if (mode == ID2SYM(rb_intern("write"))) {
-   return RB_WAITFD_OUT;
-    }
-    if (mode == ID2SYM(rb_intern("writable"))) {
```

```
-    return RB_WAITFD_OUT;
-     }
      if (mode == ID2SYM(rb_intern("rw"))) {
     return RB_WAITFD_IN|RB_WAITFD_OUT;
      }
-     if (mode == ID2SYM(rb_intern("read_write"))) {
-    return RB_WAITFD_IN|RB_WAITFD_OUT;
-     }
-     if (mode == ID2SYM(rb_intern("readable_writable"))) {
-    return RB_WAITFD_IN|RB_WAITFD_OUT;
-     }
      rb_raise(rb_eArgError, "unsupported mode: %"PRIsVALUE, mode);
      return 0;
 }
```

I even prefer just supporting wait(:rw) since we already have
"wait_readable"/"wait_writable" methods, but :r and :w may make
sense for consistency.

#5 - 01/25/2016 11:11 AM - chucke (Tiago Cardoso)

Agreeing with Eric Wong, but apart from that, awesome! Thx Nobu! :)

#6 - 01/25/2016 12:58 PM - chucke (Tiago Cardoso)

nobu (Nobuyoshi Nakada), I was thinking whether this shouldn't be extended to the #ready? method. This by default checks whether the descriptor has something to read, but I saw that the source code calls the internal C method rb_io_check_readable. Since there is already a method rb_io_check_writable, wouldn't it make sense to make the #ready method receive an argument? (:r/:w/:rw or :read/:write/:readwrite depending of the final API).

I'd open a new issue if I get your approval.