IO#read_nonblock raises IOError when called following buffered character IO

06/26/2022 04:35 PM - javanthropus (Jeremy Bopp)

Status: Rejected
Priority: Normal
Assignee: 
Target version: ruby 3.1.2p20 (2022-04-12 revision 4491bb740a) [x64-mingw-ucrt]

Description
The following example code works as expected on Linux but raises an IOError on Windows:

```ruby
r, w = IO.pipe
w.write("foobar")
c = r.getc
r.ungetc(c)
r.read_nonblock(3)
```

The error message is:

```
IOError: byte oriented read for character buffered IO
```

History

#1 - 06/27/2022 12:21 AM - shyouhei (Shyouhei Urabe)

Not looking at the implementation but sounds like an issue of Linux side to me. The error message seems intentional.

#2 - 06/29/2022 05:07 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Rejected

It is an expected behavior. 
On Windows, text/binary conversion is used by default, and ungetc pushes back the character to the conversion buffer in that case. The error message means this situation.

#3 - 06/29/2022 05:12 AM - shyouhei (Shyouhei Urabe)

@nobu (Nobuyoshi Nakada) Yes but wouldn't it be nice if IO#read_nonblock also raises on Linux?

#4 - 06/29/2022 05:21 AM - nobu (Nobuyoshi Nakada)

@shyouhei (Shyouhei Urabe) Even without any conversion?

This is an example on macOS.

```ruby
$ ruby -v
ruby 3.2.0dev (2022-06-28T06:49:05Z master 98bf8c83fa) [x86_64-darwin21]
<internal:io>:63:in `read_nonblock': byte oriented read for character buffered IO (IOError)
  from -e:1:in `<main>'
```

#5 - 06/29/2022 06:09 AM - shyouhei (Shyouhei Urabe)

nobu (Nobuyoshi Nakada) wrote in #note-4:

@shyouhei (Shyouhei Urabe) Even without any conversion?

Yes. "Byte oriented and character oriented IO operations cannot be mixed at all" sounds much simpler than the current (it sometimes raises and sometimes doesn't).

#6 - 06/29/2022 07:39 AM - nobu (Nobuyoshi Nakada)

Do you mean that once character oriented methods are called, all byte oriented methods always fail?
Basically yes. Some edge cases can be considered too (e.g. `IO#flush` could also reset this, `IO#pread` could be special-case allowed (avoiding `IO#seek` is the reason for this method to exist), etc).

There needs to be documentation to cover the corner cases no matter what. Wouldn't it be better to document the current behavior rather than add artificial limitations?

For what it's worth, I was able to reproduce the reported behavior on Linux with a minor modification to the example script:

```python
r, w = IO.pipe
r.set_encoding('utf-8', universal_newline=True)  # Set line ending handling like on Windows
w.write("foobar")
c = r.getc
r.ungetc(c)
r.read_nonblock(3)
```