Ruby - Bug #21280

StringIO#set_encoding warns when backed by chilled string literal

04/22/2025 12:39 AM - jeremyevans0 (Jeremy Evans)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:		**Backport:**	3.2: UNKNOWN, 3.3: UNKNOWN, 3.4: UNKNOWN

Description

StringIO#set_encoding changes the underlying string encoding if the string is not frozen, but does not change the underlying string encoding if the string is frozen. In Ruby 3.4, this results in a warning for chilled literal strings:

```
$ ruby34 -w -r stringio -e "StringIO.new('').set_encoding('binary')"
-e:1: warning: literal string will be frozen in the future (run with --debug-frozen-string-literal
 for more information)
```

I believe Ruby should emit this warning only for cases that will break when string literals are frozen. This is not one of those cases, so Ruby should not emit the warning. To avoid emitting the warning, I think StringIO#set_encoding should not set the encoding on the underlying string for chilled literal strings. I submitted a pull request to avoid changing the encoding of the underlying string: https://github.com/ruby/stringio/pull/128

However, @rhenium (Kazuki Yamaguchi) said that he thought the encoding of the underlying string should still be changed, and the warning should be emitted (https://github.com/ruby/stringio/pull/128#issuecomment-2818362875).

For some history, before Ruby 2.3, StringIO#set_encoding used to always set the encoding of the underlying string. This was changed in #11827, when the encoding was not set on the underlying string if the string was frozen (3e1c01ae463a8c9d8bbe9050251a2538ddb0292f).

In https://bugs.ruby-lang.org/issues/11827#note-3, @nurse wrote:

> Away from the case and thinking ideal behavior, StringIO should be a view of given source string and set_encoding shouldn't change source encoding.
> But I'm not sure that it is worth breaking the compatibility.

I think this means that ideally, absent backwards compatibility issues, StringIO#set_encoding should never change the underlying string encoding.

In https://bugs.ruby-lang.org/issues/11827#note-4, @shugo (Shugo Maeda) gave an example from @nobu (Nobuyoshi Nakada) that open-uri depends on the current behavior:

```
      enc = Encoding::ASCII_8BIT unless enc
      if self.respond_to? :force_encoding
        self.force_encoding(enc)
      elsif self.respond_to? :string
        self.string.force_encoding(enc)
      else # Tempfile
        self.set_encoding enc
      end
    end
```

However, as StringIO#string is defined, this will call self.string.force_encoding(enc) and not self.set_encoding enc, so I'm not sure why a change to String#set_encoding would affect the behavior of this example.

@rhenium (Kazuki Yamaguchi) pointed out that this issue affects StringIO#binmode and StringIO#set_encoding_by_bom as well as StringIO#encoding.

How do we want to handle this case? Should this result in a warning (current behavior), or is it safe to avoid changing the encoding of the underlying string for chilled strings (as is done for frozen strings)?

History

#1 - 04/23/2025 05:34 AM - byroot (Jean Boussier)

> Should this result in a warning (current behavior), or is it safe to avoid changing the encoding of the underlying string for chilled strings (as is done for frozen strings)?

I think the current behavior is correct in the sense that "chilled strings" are mutable strings, hence should behave just like a mutable string. In this case it means changing the encoding.

Chilled strings should also emit a warning when they are mutated to signal that the behavior will change if --enable-frozen-string-literal is turned on, which it does right now.

Based on that I think the current behavior is correct, this warning is correctly warning you action is needed here.

#2 - 04/23/2025 07:18 AM - jeremyevans0 (Jeremy Evans)

byroot (Jean Boussier) wrote in #note-1:

> Chilled strings should also emit a warning when they are mutated to signal that the behavior will change if --enable-frozen-string-literal is turned on, which it does right now.

But behavior in this case will not change if --enable-frozen-string-literal is turned on, because StringIO#set_encoding does not set the encoding of the underlying string for frozen strings.

> Based on that I think the current behavior is correct, this warning is correctly warning you action is needed here.

But no action is needed. Things will automatically work when strings move from chilled by default to frozen by default, because StringIO#set_encoding will not try to change the encoding of the underlying string in that case.

#3 - 04/23/2025 07:25 AM - byroot (Jean Boussier)

> But behavior in this case will not change

Maybe I'm missing something, but I think the behavior will change:

```
>> StringIO.new("").set_encoding(Encoding::BINARY).string.encoding
(irb):7: warning: literal string will be frozen in the future (run with --debug-frozen-string-literal for more
 information)
=> #<Encoding:BINARY (ASCII-8BIT)>
>> StringIO.new(+"").set_encoding(Encoding::BINARY).string.encoding
=> #<Encoding:BINARY (ASCII-8BIT)>
>> StringIO.new("".freeze).set_encoding(Encoding::BINARY).string.encoding
=> #<Encoding:UTF-8>
```

#4 - 04/23/2025 02:19 PM - jeremyevans0 (Jeremy Evans)

byroot (Jean Boussier) wrote in #note-3:

> > But behavior in this case will not change
>
> Maybe I'm missing something, but I think the behavior will change:
>
> ```
> >> StringIO.new("").set_encoding(Encoding::BINARY).string.encoding
> (irb):7: warning: literal string will be frozen in the future (run with --debug-frozen-string-literal for
> more information)
> => #<Encoding:BINARY (ASCII-8BIT)>
> >> StringIO.new(+"").set_encoding(Encoding::BINARY).string.encoding
> => #<Encoding:BINARY (ASCII-8BIT)>
> >> StringIO.new("".freeze).set_encoding(Encoding::BINARY).string.encoding
> => #<Encoding:UTF-8>
> ```

The full sentence was:

> But behavior in this case will not change if --enable-frozen-string-literal is turned on, because StringIO#set_encoding does not set the encoding

> of the underlying string for frozen strings.

With the patch, StringIO#set_encoding treats a chilled string as a frozen string, so behavior when --enable-frozen-string-literal is turned on does not change. The present situation is that --enable-frozen-string-literal changes behavior of StringIO#set_encoding.

#5 - 04/23/2025 02:22 PM - byroot (Jean Boussier)

Perhaps it's the jetlag talking, but I think the current behavior is desirable.

Chilled string as designed should behave like mutable strings, and warn when a frozen one would have behaved differently. Isn't that the current behavior?

#6 - 04/23/2025 02:37 PM - jeremyevans0 (Jeremy Evans)

byroot (Jean Boussier) wrote in #note-5:

> Chilled string as designed should behave like mutable strings, and warn when a frozen one would have behaved differently. Isn't that the current behavior?

In this case, a frozen string would not behave differently, since StringIO#set_encoding treats frozen strings differently than it does unfrozen strings.

The purpose of chilled strings is to warn users when code will break when the strings are frozen. Since code will not break in this case, issuing a warning seems like a bad idea, as it will prompt code changes when none are actually needed.

#7 - 04/23/2025 02:39 PM - byroot (Jean Boussier)

> Since code will not break in this case

That is debatable. It won't raise an exception yes, but the behavior will change as you can see in the IRB session I shared.

#8 - 04/26/2025 06:11 PM - Eregon (Benoit Daloze)

I think just removing the warning is not right here (also explicitly testing for chilled strings feels dirty).

I think callers of StringIO.new should not pass a frozen String, at least if any StringIO "write" method is used, and I consider StringIO#set_encoding to be a "write" method (since currently in the non-frozen cases it does mutate the String).
Fundamentally the behavior of setting the encoding of the String only if it's not frozen for StringIO#set_encoding is inconsistent/broken/brittle, it might very much break things because AFAIK it affects the encoding of the string returned by StringIO#string.

So from that I can see two solutions:

- StringIO#set_encoding never sets the String encoding, but that is a potentially breaking change (though it would address the fundamentals, so I think worth a try).
- People should not use frozen strings with StringIO if they then use "write" methods on the StringIO. I think we should warn for both frozen and chilled strings in StringIO#set_encoding, to tell users they shouldn't pass a frozen string + call StringIO#set_encoding (they can probably set the encoding when creating the StringIO instead, or pass a non-frozen String).

#9 - 05/03/2025 06:58 AM - jeremyevans0 (Jeremy Evans)

It seems like most developers are against treating chilled strings differently than unfrozen strings in this case. I can understand that, since that's usually the best course of action.

As an alternative, we could remove the setting of the underlying string encoding in all cases. Then behavior is the same for chilled strings, unfrozen strings, and frozen strings. @naruse (Yui NARUSE) already mentioned that this was ideal in https://bugs.ruby-lang.org/issues/11827#note-3, but wasn't sure if it was worth breaking backwards compatibility. I think avoiding a bogus warning for chilled strings is worth it, assuming there isn't significant breakage elsewhere. The example in https://bugs.ruby-lang.org/issues/11827#note-4 shouldn't be affected, since as I mentioned earlier, StringIO takes the elsif self.respond_to? :string branch, not the else branch.

I've submitted a pull request to implement this: https://github.com/ruby/stringio/pull/132 . I think either approach is fine, as long as the bogus warning is avoided.

#10 - 05/09/2025 03:28 AM - mame (Yusuke Endoh)

Discussed at the dev meeting. @matz (Yukihiro Matsumoto) said that he prefers not to issue warnings that could be false positives, even if they are true positives in many cases.

The code StringIO.new("").set_encoding('binary') will work even in future as long as StringIO is used read-only. Therefore, this warning could be a false positive, and it would be nice to stop this warning in StringIO#set_encoding. In code that needs to be changed (i.e., the StringIO is used as read-write mode), it is expected that a warning will be finally issued by StringIO#write or something after a while, even if we don't issue the warning in StringIO#set_encoding.

#11 - 05/09/2025 10:28 AM - Eregon (Benoit Daloze)

https://github.com/ruby/stringio/pull/132 is a much cleaner solution, was it considered?
(https://bugs.ruby-lang.org/issues/21281 doesn't have the log yet so I can't check that, EDIT found it:
https://github.com/ruby/dev-meeting-log/blob/master/2025/DevMeeting-2025-05-08.md)

#12 - 05/13/2025 03:22 AM - jeremyevans0 (Jeremy Evans)

- Status changed from Open to Closed

Fixed by 18d395e0784401585b5c14300e689de55e208647