Ruby - Bug #6324

Spurious/incorrect warning on File.open with internal_encoding specified in string mode to duplicate external_encoding

04/20/2012 12:53 AM - jrochkind (jonathan rochkind)

Status:	Closed		
Priority:	Normal		
Assignee:	naruse (Yui NARUSE)		
Target version:			
ruby -v:	ruby 1.9.3p125 (2012-02-16 revision 34643) [x86_64-linux]	**Backport:**	

Description

Normally, if you open a File specifying an internal_encoding equivalent to an external_encoding, using the string method, it gives you a warning. *Normally* this warning is possibly appropriate and accurate:

irb(main):018:0* f = File.open("test", "r:cp866:cp866")
(irb):18: warning: Ignoring internal encoding cp866: it is identical to external encoding cp866

However, there is a case where it is NOT. If you have set your Encoding.default_internal

irb(main):019:0> Encoding.default_internal = "UTF-8"
irb(main):020:0> f = File.open("test", "r:cp866:cp866")
(irb):20: warning: Ignoring internal encoding cp866: it is identical to external encoding cp866

In this case, it is neccesary to set the internal_encoding to override the non-nil Encoding.default_internal

The functionality in fact *works* here, we HAVE succesfully over-ridden the default_internal:

irb(main):022:0> p f.internal_encoding
=> nil
irb(main):023:0> f.read.encoding
=> #Encoding:IBM866

So the warning is in fact *wrong*, the :internal_encoding was NOT ignored, it was used as desired. The warning is also unneccesary, what was being done here makes perfect sense, there's no need for a warning.

Note that the named argument approach works differnetly, no warning is output (whether or not you've set Encoding.default_internal).

```
irb(main):024:0> f = File.open("foo", :external_encoding => "cp866", :internal_encoding => "cp866"
)
=> #<File:foo>  # NO WARNING OUTPUT
irb(main):025:0> f.internal_encoding
=> nil # CORRECT
irb(main):026:0> f.read.encoding
=> #<Encoding:IBM866> # CORRECT
```

Both the named argument and the permission-string method seem to work properly and equivalently. But only the permission-string argument method gives you an incorrect warning.

Related issues:

Related to Ruby - Bug #5568: IO#set_encoding ignores internal when the same a...	**Closed**	**11/04/2011**

Associated revisions

Revision afd9ce9d9e00cfc940e4989a34033fa73268d969 - 05/04/2012 04:14 PM - naruse (Yui NARUSE)

- io.c (parse_mode_enc): remove warnings 'Ignoring internal encoding'.
  [ruby-core:44455] [Bug #6324]

- io.c (io_encoding_set): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@35538 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision afd9ce9d - 05/04/2012 04:14 PM - naruse (Yui NARUSE)

- io.c (parse_mode_enc): remove warnings 'Ignoring internal encoding'.
  [ruby-core:44455] [Bug #6324]

- io.c (io_encoding_set): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@35538 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 04/20/2012 04:36 AM - mame (Yusuke Endoh)

- Status changed from Open to Assigned

- Assignee set to duerst (Martin Dürst)

Martin-sensei, Naruse-san, what do you think?

--
Yusuke Endoh mame@tsg.ne.jp

#2 - 04/22/2012 05:34 PM - duerst (Martin Dürst)

- Assignee changed from duerst (Martin Dürst) to naruse (Yui NARUSE)

In my view, we could/should get rid of this warning. See details below.

jrochkind (jonathan rochkind) wrote:

> Normally, if you open a File specifying an internal_encoding equivalent to an external_encoding, using the string method, it gives you a warning. *Normally* this warning is possibly appropriate and accurate:
>
> irb(main):018:0* f = File.open("test", "r:cp866:cp866")
> (irb):18: warning: Ignoring internal encoding cp866: it is identical to external encoding cp866

This is actually misleading. The actual internal encoding (the encoding of the Strings read in) is still cp866, so the information isn't ignored. It's just that the "internal_encoding" slot of the file isn't set because it can be deduced from context.

Also, there's really not much harm if a programmer writes File.open("test", "r:cp866:cp866") instead of just File.open("test", "r:cp866"). The warning may help a programmer who wrongly assumes that there's always a need for giving both encodings, but I think documentation should be enough for this.

> However, there is a case where it is NOT. If you have set your Encoding.default_internal
>
> irb(main):019:0> Encoding.default_internal = "UTF-8"
> irb(main):020:0> f = File.open("test", "r:cp866:cp866")
> (irb):20: warning: Ignoring internal encoding cp866: it is identical to external encoding cp866
>
> In this case, it is neccesary to set the internal_encoding to override the non-nil Encoding.default_internal
>
> The functionality in fact *works* here, we HAVE succesfully over-ridden the default_internal:
>
> irb(main):022:0> p f.internal_encoding
> => nil
> irb(main):023:0> f.read.encoding
> => #Encoding:IBM866
>
> So the warning is in fact *wrong*, the :internal_encoding was NOT ignored, it was used as desired. The warning is also unneccesary, what was being done here makes perfect sense, there's no need for a warning.

There's an additional case where the warning might not make sense. Assume I open files with variable external encodings that I all want to have the same internal encoding:

somehow calculate ext_enc

File.open("test", "r:#{ext_enc}:cp866")

Now such a program will work quite well, without warnings. But occasionally, it will produce a warning, namely when it hits a case with ext_enc == "cp866". One could of course rewrite this as
File.open("test", ext_enc == "cp866" ? "r:cp866": "r:#{ext_enc}:cp866")
but this really seems way too complicated. It would be better to get rid of the warning.

> Note that the named argument approach works differnetly, no warning is output (whether or not you've set Encoding.default_internal).
>
> ```
> irb(main):024:0> f = File.open("foo", :external_encoding => "cp866", :internal_encoding => "cp866")
> => #<File:foo>  # NO WARNING OUTPUT
> irb(main):025:0> f.internal_encoding
> => nil # CORRECT
> irb(main):026:0> f.read.encoding
> => #<Encoding:IBM866> # CORRECT
> ```
>
> Both the named argument and the permission-string method seem to work properly and equivalently. But only the permission-string argument method gives you an incorrect warning.

Another good reason to get rid of the warning.

So I hope my opinion is clear. But it wasn't my idea in the first place, so maybe I'm missing something. I'm switching assignee over to Yui.

#3 - 05/05/2012 01:14 AM - naruse (Yui NARUSE)

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r35538.
jonathan, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- io.c (parse_mode_enc): remove warnings 'Ignoring internal encoding'.
  [ruby-core:44455] [Bug #6324]

- io.c (io_encoding_set): ditto.