Ruby trunk - Feature #2975

Kernel.warn always writes despite $VERBOSE value

03/17/2010 04:28 AM - DanRathbun (Dan Rathbun)

Status:	Closed
Priority:	Normal
Assignee:	
Target version:	2.0.0

Description

=begin
Kernel.warn always writes despite $VERBOSE value

The RDoc say:
http://www.ruby-doc.org/core/classes/Kernel.html#M005921
warn(msg) => nil
Display the given message (followed by a newline) on STDERR unless warnings are disabled (for example with the -W0 flag).

(from Ruby.h, ver 1.8.6, line 562..564, Language="C")
void rb_warning _((const char*, ...)); /* reports if `-w' specified */
void rb_sys_warning _((const char*, ...)); /* reports if `-w' specified /
void rb_warn __((const char, ...)); /* reports always */*

In ver 1.8.6, in practice Kernel.warn (and by inclusion,) anyobject's instance method warn, outputs no matter what the setting of $VERBOSE. It then appears to work as though it's calling rb_warn in the core, but the RDoc seems to say it should act like it's calling rb_warning in the core.

This has forced me to make my 'warn' calls work the way they should, by doing this:

```
# Send warning only if in Verbose mode
warn('My Informational Message') if $VERBOSE

# Send warning unless in Silent mode
warn('My Important Message') unless $VERBOSE.nil?

# Send warning no matter what Verbose mode
warn('My Critical Message that MUST be displayed!')
```

I'm tired of this workaround:
I think we need something like this overrride, where I define 3 methods,
warn, warn?, warn!
and in addition, I have them return a boolean result (the original warn just returned nil.)

```
#!ruby warn_ovr.rb
# Make Warnings work the way they should.
# by: Dan Rathbun - 16 MAR 2010 - Palm Bay, FL, USA
# TERMS: Public Domain, Take it, Use it, Abuse it!

module Kernel

# alias the old warn method
alias_method(:old_warn,:warn)

# warn! will always send to $stderr
# regardless of $VERBOSE setting
def warn!(msg)
unless msg.is_a?(String)
raise(TypeError,'String argument expected.',caller(1))
end
$stderr.write(msg + "\n")
return true # no IO error occured
end

# warn will now send to $stderr
```

```
# ONLY if $VERBOSE is not Silent mode (nil)
def warn(msg)
unless msg.is_a?(String)
raise(TypeError,'String argument expected.',caller(1))
end
unless $VERBOSE.nil?
$stderr.write(msg + "\n")
return true
else
return false
end
end

# warn? will send to $stderr
# ONLY if $VERBOSE is in Verbose mode (true)
def warn?(msg)
unless msg.is_a?(String)
raise(TypeError,'String argument expected.',caller(1))
end
if $VERBOSE
$stderr.write(msg + "\n")
return true
else
# We return false if $VERBOSE is nil or false
return false
end
end

end # Kernel
```

NOTE: The above override script does not override the Module method Kernel.warn, only the instance method. (I could have just run the script in the ObjectSpace without the Kernel wrapper, and added the methods to class Object. Their use either way is the same.)
=end

History

#1 - 03/17/2010 06:42 AM - nobu (Nobuyoshi Nakada)

- Category changed from core to doc

- ruby -v set to all

=begin
It's a document issue.

```
$ ruby -w0 -e 'p $VERBOSE'
true
$ ruby -W0 -e 'p $VERBOSE'
nil
```

=end

#2 - 03/17/2010 07:56 PM - DanRathbun (Dan Rathbun)

=begin
On 03/17/2010 06:42 AM - Nobuyoshi Nakada
--> said: "It's a document issue."

I agree.. that there IS an existing doc issue.

But that's only a SMALL part of the issue.

Proposing:
(A) Change and adition of module Kernel warning methods:
(1) Change name of Kernel.warn to Kernel.warn! [rb_warn_always, see (B)(3)]
Add Typechecking, and return boolean true result.
(2) Add 2 methods to module Kernel, with Typechecking, and return boolean result.
(a) Kernel.warn rb_warn_not_nil, see (B)(1) Kernel.warn? [rb_warn_verbose, see (B)(2)]

(B) C core function wrappers (work as per Ruby given above):
(1) Kernel.warn BOOL rb_warn_not_nil

(2) Kernel.warn? BOOL rb_warn_verbose   wrapper for VOID rb_warning
(3) Kernel.warn! BOOL rb_warn_always   wrapper for VOID rb_warn

The ALTERNATIVE, is to add a 2nd parameter to the existing Kernel.warn method:
(and revise as by combining the 3 Ruby methods given above.)
warn(message, warnIf = :always)
warnIf may also be :notnil | :verbose
=end

#3 - 04/04/2010 02:05 AM - znz (Kazuhiro NISHIYAMA)

- Category set to doc

- Target version set to 2.0.0

=begin

=end

#4 - 03/18/2012 03:36 PM - nahi (Hiroshi Nakamura)

- Description updated

- Status changed from Open to Closed

I mark as Closed since it seems we already have it in docs.

Please file a new ticket for the proposed API change since it's dated...