Ruby master - Feature #11955

Expose Object that Receives logs in Logger

01/05/2016 09:59 PM - schneems (Richard Schneeman)

Status:	Assigned
Priority:	Normal
Assignee:	sonots (Naotoshi Seo)
Target version:	

Description

I need to be able to perform logic based on the destination of a current logger, this is currently not possible without instance_variable_get. Why would you need to see what destination a logger is going to? There is a common pattern in long lived programs like webservers. You want logs on disk for later reference, but you also want them in STDOUT to make development and debugging easier. Rails does this in development mode. Since there is no way to see if a logger is already going to STDOUT, it gets extended to log to STDOUT so logs show up twice.

While that example was complicated the logic I want is very simple: if you have a logger that is logging to STDOUT, do nothing, otherwise log to STDOUT and current logger. You cannot do this today without exposing the destination of the logger. This patch exposes the logger destination and allows us to write code like this:

```
def make_sure_logging_to_stdout(logger)
  unless logger.destination == STDOUT   # <==== Cannot do this today
    stdout_logger = ::Logger.new(STDOUT)
    logger.extend(Module do
      def add((*args, &block)
        stdout_logger.add(*args, &block)
        super(*args, &block)
      end
    end)
  end
end

logger = Logger.new(STDOUT)
make_sure_logging_to_stdout(logger)

logger.fatal("An error has occured")
```

We should be able to inspect the destination of a logger, this patch enables this functionality.

History

#1 - 01/06/2016 04:27 PM - schneems (Richard Schneeman)

Here is a patch to Rails that could benefit from standardizing access to the logger destination object: https://github.com/rails/rails/pull/22933

#2 - 01/07/2016 05:44 AM - nobu (Nobuyoshi Nakada)

- Description updated

Your example doesn't seem to able to get rid of adding stdout_logger twice or more, even with logger.destination.

Maybe won't it be better to do in LogDevice layer?

#3 - 01/08/2016 05:00 PM - schneems (Richard Schneeman)

> Your example doesn't seem to able to get rid of adding stdout_logger twice or more, even with logger.destination.

It took a long time to write that example to be short, maybe I missed some details. A real world example is in that linked pull request, you can see the comparison I implemented https://github.com/rails/rails/blob/76c385709c873a7105e3a267d84c4e70417a15e2/activesupport/lib/active_support/logger.rb#L8-L17 this is where I would like to use a public interface instead of instance_variable_get.

LogDevice already exposes the destination, but Logger does not expose the LogDevice object. I did not know if there was a reason people did not

what to provide access to LogDevice. Would you prefer that I submitted a patch to expose the LogDevice instead?

#4 - 01/19/2016 08:03 PM - schneems (Richard Schneeman)

- File ruby-changes.patch added

Nobu, I've added a new patch that would expose the LogDevice object in a Logger instance. This would be acceptable for my needs.

#5 - 01/20/2016 12:27 AM - hsbt (Hiroshi SHIBATA)

- Status changed from Open to Assigned

- Assignee set to sonots (Naotoshi Seo)

#6 - 03/15/2016 04:42 PM - schneems (Richard Schneeman)

Anything else that needs to be done for this patch?

#7 - 04/18/2016 01:16 PM - sonots (Naotoshi Seo)

I am wondering of the interface yet.

Users pass an io object to Logger constructor as logdev like Logger.new(logdev), so getting the io object from Logger#logdev seems natural. However,

```
attr_reader :logdev
```

returns a LogDevice instance rather than io object, which seems natural from the view of source codes.

#8 - 06/20/2016 01:45 PM - schneems (Richard Schneeman)

Sorry for the delay, bugs.ruby-lang.org does not send me emails for some reason.

The first patch I attached returned @logdev.dev which is the IO object. It was discussed that to do this Logger must know too much about the logdev interface and it would be simpler to expose the logdev instead.

Either approach will work for my use cases. Now that 2.4.0 preview is out is there a feature freeze? Is there any chance this will come out in time for christmas?

#9 - 11/09/2017 11:35 AM - tensho (Andrew Babichev)

So will **#logdev** will be exposed in Logger or Logger::LogDevice?

Files

ruby-changes.patch	1.04 KB	01/05/2016	schneems (Richard Schneeman)
ruby-changes.patch	2.17 KB	01/19/2016	schneems (Richard Schneeman)