Optimize Proc#call

10/05/2015 09:32 PM - ko1 (Koichi Sasada)

Status: Closed
Priority: Normal
Assignee: ko1 (Koichi Sasada)

Description
Some years ago, I added optimized Proc#call that skips construction of a method frame for Proc#call and call block directly. That time, test-all shows some failures.

However, I tried it now it doesn't show any errors. So that I will enable it.

This change introduces incompatibilities.

(1) Backtrace doesn't show Proc#call line.

    -> {
        puts caller(0)
    }.call

    # current
test.rb:4:in `block in <main>'
test.rb:5:in `call'
test.rb:5:in `<main>'

    # optimized
    ../../trunk/test.rb:4:in `block in <main>'
    ../../trunk/test.rb:5:in `<main>'

(2) TracePoint ignores Proc#call call.

    TracePoint.new({'call'=>{|tp| p tp}).enable
    ->{}.call

    # current
    #<TracePoint:c_call `call'@test.rb:2>
    #<TracePoint:b_call@test.rb:2>

    # optimized
    #<TracePoint:b_call@../../trunk/test.rb:2>

Let me know if you have trouble with these incompatibilities.

BTW, Object#send has already same technique (and behavior with (1), (2)).

Related issues:
Related to Ruby master - Bug #12332: [PATCH] proc.c: fix RDoc of Proc#===/call...

Associated revisions
Revision 0018a711 - 10/05/2015 09:34 PM - ko1 (Koichi Sasada)

- proc.c: enable optimization of Proc#call. [Feature #11569]
- NEWS: write about this optimization and incompatibilities.
- test/ruby/test_backtrace.rb: catch up this fix.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@52050 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 52050 - 10/05/2015 09:34 PM - ko1 (Koichi Sasada)
• proc.c: enable optimization of Proc#call. [Feature #11569]
• NEWS: write about this optimization and incompatibilities.
• test/ruby/test_backtrace.rb: catch up this fix.

Revision 52050 - 10/05/2015 09:34 PM - ko1 (Koichi Sasada)

• proc.c: enable optimization of Proc#call. [Feature #11569]
• NEWS: write about this optimization and incompatibilities.
• test/ruby/test_backtrace.rb: catch up this fix.

Revision 52050 - 10/05/2015 09:34 PM - ko1 (Koichi Sasada)

• proc.c: enable optimization of Proc#call. [Feature #11569]
• NEWS: write about this optimization and incompatibilities.
• test/ruby/test_backtrace.rb: catch up this fix.

Revision 52050 - 10/05/2015 09:34 PM - ko1 (Koichi Sasada)

• proc.c: enable optimization of Proc#call. [Feature #11569]
• NEWS: write about this optimization and incompatibilities.
• test/ruby/test_backtrace.rb: catch up this fix.

Revision 3044a0bc - 05/15/2016 09:11 PM - normal

proc.c: fix RDoc of Proc#===/call/yield[]

[Bug #12332]

Since r52050 ("proc.c: enable optimization of Proc#call") for [Feature #11569], we need to maintain this documentation in a way RDoc comprehends. This is probably not worth fixing in RDoc itself since this uses a non-standard internal C API which is subject to change without notice.

ref:
http://mid.gmane.org/20160429212836.GA16605@dcvr.yhbt.net
http://mid.gmane.org/1461959651.806728.670.51764@mail.rambler.ru
http://blade.nagaokaut.ac.jp/ruby/ruby-talk/455458

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@55010 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 55010 - 05/15/2016 09:11 PM - normalperson (Eric Wong)

proc.c: fix RDoc of Proc#===/call/yield[]

[Bug #12332]

Since r52050 ("proc.c: enable optimization of Proc#call") for [Feature #11569], we need to maintain this documentation in a way RDoc comprehends. This is probably not worth fixing in RDoc itself since this uses a non-standard internal C API which is subject to change without notice.

ref:
http://mid.gmane.org/20160429212836.GA16605@dcvr.yhbt.net
http://mid.gmane.org/1461959651.806728.670.51764@mail.rambler.ru
http://blade.nagaokaut.ac.jp/ruby/ruby-talk/455458

Revision 55010 - 05/15/2016 09:11 PM - normal

proc.c: fix RDoc of Proc#===/call/yield[]

[Bug #12332]

Since r52050 ("proc.c: enable optimization of Proc#call")
for [Feature #11569], we need to maintain this documentation in a way RDoc comprehends. This is probably not worth fixing in RDoc itself since this uses a non-standard internal C API which is subject to change without notice.

ref:
http://mid.gmane.org/20160429212836.GA16605@dcvr.yhbt.net
http://mid.gmane.org/1461959651.806728.670.51764@mail.rambler.ru
http://blade.nagaokaut.ac.jp/ruby/ruby-talk/435458

Revision 55010 - 05/15/2016 09:11 PM - normal
proc.c: fix RDoc of Proc#===/call/yield/[]

[Bug #12332]

Since r52050 ("proc.c: enable optimization of Proc#call") for [Feature #11569], we need to maintain this documentation in a way RDoc comprehends. This is probably not worth fixing in RDoc itself since this uses a non-standard internal C API which is subject to change without notice.

ref:
http://mid.gmane.org/20160429212836.GA16605@dcvr.yhbt.net
http://mid.gmane.org/1461959651.806728.670.51764@mail.rambler.ru
http://blade.nagaokaut.ac.jp/ruby/ruby-talk/435458

Revision 55010 - 05/15/2016 09:11 PM - normal
proc.c: fix RDoc of Proc#===/call/yield/[]

[Bug #12332]

Since r52050 ("proc.c: enable optimization of Proc#call") for [Feature #11569], we need to maintain this documentation in a way RDoc comprehends. This is probably not worth fixing in RDoc itself since this uses a non-standard internal C API which is subject to change without notice.

ref:
http://mid.gmane.org/20160429212836.GA16605@dcvr.yhbt.net
http://mid.gmane.org/1461959651.806728.670.51764@mail.rambler.ru
http://blade.nagaokaut.ac.jp/ruby/ruby-talk/435458

History

#1 - 10/05/2015 09:35 PM - ko1 (Koichi Sasada)
  - Status changed from Open to Closed

Applied in changeset r52050.

- proc.c: enable optimization of Proc#call. [Feature #11569]
- NEWS: write about this optimization and incompatibilities.
- test/ruby/test_backtrace.rb: catch up this fix.

#2 - 06/08/2016 05:06 AM - usa (Usaku NAKAMURA)
  - Related to Bug #12332: [PATCH] proc.c: fix RDoc of Proc#===/call/yield/[] added