test.rb

ary = []

t = Thread.new {
begin
  sleep
  ensure
  begin
    ary << Thread.current.status
    sleep
    ensure
    ary << Thread.current.status
  end
end
}

sleep 0.01

begin
  t.kill
  sleep 0.01
  t.raise
end

expected result
["aborting", "aborting"]

actual result
["aborting", "run"]

reason

rb_threadptr_execute_interrupt() overrides th->status unconditionally.

Associated revisions

Revision 9cbf4732 - 11/28/2012 08:31 AM - kosaki (Motohiro KOSAKI)

- vm_core.h (enum rb_thread_status): remove THREAD_TO_KILL
- vm_core.h (struct rb_thread_struct): add to_kill field
- thread.c (terminate_i): convert THREAD_TO_KILL to to_kill.
- thread.c (rb_threadptr_to_kill): ditto.
- thread.c (rb_thread_kill): ditto.
- thread.c (rb_thread_wakeup_alive): ditto.
Revision 37931 - 11/28/2012 08:31 AM - kosaki (Motohiro KOSAKI)

- vm_core.h (enum rb_thread_status): remove THREAD_TO_KILL
- vm_core.h (struct rb_thread_struct): add to_kill field
- thread.c (terminate_i): convert THREAD_TO_KILL to to_kill.
- thread.c (rb_threadptr_to_kill): ditto.
- thread.c (rb_thread_kill): ditto.
- thread.c (rb_thread_wakeup_alive): ditto.
- thread.c (thread_list_i): ditto.
- thread.c (static const char): ditto.
- thread.c (thread_status_name): ditto.
- thread.c (rb_thread_status): ditto.
- thread.c (rb_thread_inspect): ditto.

- vm_backtrace.c (thread_backtrace_to_ary): ditto.

- thread.c (rb_threadptr_execute_interrupts): fix thread status overwritten issue. [Bug #7450] [ruby-core:50249]

- test/ruby/test_thread.rb (test_thread_status_raise_after_kill): test for the above.
- test/ruby/test_thread.rb (test_status_and_stop_p): remove Thread.control_interrupt unsafe test. Thread#kill no longer changes thread status. Instead of, Thread#kill receiver changes their own status when receiving kill signal.

Revision 37931 - 11/28/2012 08:31 AM - kosaki (Motohiro KOSAKI)

- vm_core.h (enum rb_thread_status): remove THREAD_TO_KILL
- vm_core.h (struct rb_thread_struct): add to_kill field
- thread.c (terminate_i): convert THREAD_TO_KILL to to_kill.
- thread.c (rb_threadptr_to_kill): ditto.
- thread.c (rb_thread_kill): ditto.
- thread.c (rb_thread_wakeup_alive): ditto.
- thread.c (thread_list_i): ditto.
- thread.c (static const char): ditto.
- thread.c (thread_status_name): ditto.
- thread.c (rb_thread_status): ditto.
- thread.c (rb_thread_inspect): ditto.
vm_backtrace.c (thread_backtrace_to_ary): ditto.

thread.c (rb_threadptr_execute_interrups): fix thread status overwritten issue. [Bug #7450] [ruby-core:50249]

test/ruby/test_thread.rb (test_thread_status_raise_after_kill): test for the above.

test/ruby/test_thread.rb (test_thread_status_in_trap): test for thread status in trap.

test/ruby/test_thread.rb (test_status_and_stop_p): remove Thread.control_interrupt unsafe test. Thread#kill no longer changes thread status. Instead of, Thread#kill receiver changes their own status when receiving kill signal.

Revision 37931 - 11/28/2012 08:31 AM - kosaki (Motohiro KOSAKI)

vm_core.h (enum rb_thread_status): remove THREAD_TO_KILL
vm_core.h (struct rb_thread_struct): add to_kill field
thread.c (terminate_i): convert THREAD_TO_KILL to to_kill.
thread.c (rb_threadptr_to_kill): ditto.
thread.c (rb_thread_kill): ditto.
thread.c (rb_thread_wakeup_alive): ditto.
thread.c (thread_list_i): ditto.
thread.c (static const char): ditto.
thread.c (thread_status_name): ditto.
thread.c (rb_thread_status): ditto.
thread.c (rb_thread_inspect): ditto.

vm_backtrace.c (thread_backtrace_to_ary): ditto.

thread.c (rb_threadptr_execute_interrups): fix thread status overwritten issue. [Bug #7450] [ruby-core:50249]

test/ruby/test_thread.rb (test_thread_status_raise_after_kill): test for the above.

test/ruby/test_thread.rb (test_thread_status_in_trap): test for thread status in trap.

test/ruby/test_thread.rb (test_status_and_stop_p): remove Thread.control_interrupt unsafe test. Thread#kill no longer changes thread status. Instead of, Thread#kill receiver changes their own status when receiving kill signal.

Revision 37931 - 11/28/2012 08:31 AM - kosaki (Motohiro KOSAKI)

vm_core.h (enum rb_thread_status): remove THREAD_TO_KILL
vm_core.h (struct rb_thread_struct): add to_kill field
thread.c (terminate_i): convert THREAD_TO_KILL to to_kill.
thread.c (rb_threadptr_to_kill): ditto.
thread.c (rb_thread_kill): ditto.
thread.c (rb_thread_wakeup_alive): ditto.
thread.c (thread_list_i): ditto.
thread.c (static const char): ditto.
thread.c (thread_status_name): ditto.
thread.c (rb_thread_status): ditto.
thread.c (rb_thread_inspect): ditto.

vm_backtrace.c (thread_backtrace_to_ary): ditto.

thread.c (rb_threadptr_execute_interrups): fix thread status overwritten issue. [Bug #7450] [ruby-core:50249]

test/ruby/test_thread.rb (test_thread_status_raise_after_kill): test for the above.
• test/ruby/test_thread.rb (test_thread_status_in_trap): test for thread status in trap.

• test/ruby/test_thread.rb (test_status_and_stop_p): remove Thread.control_interrupt unsafe test. Thread#kill no longer changes thread status. Instead of, Thread#kill receiver changes their own status when receiving kill signal.

Revision 37931 - 11/28/2012 08:31 AM - kosaki (Motohiro KOSAKI)

• vm_core.h (enum rb_thread_status): remove THREAD_TO_KILL
• vm_core.h (struct rb_thread_struct): add to_kill field
• thread.c (terminate_i): convert THREAD_TO_KILL to to_kill.
• thread.c (rb_threadptr_to_kill): ditto.
• thread.c (rb_thread_kill): ditto.
• thread.c (rb_thread_wakeup_alive): ditto.
• thread.c (thread_list_i): ditto.
• thread.c (static const char): ditto.
• thread.c (thread_status_name): ditto.
• thread.c (rb_thread_status): ditto.
• thread.c (rb_thread_inspect): ditto.

• vm_backtrace.c (thread_backtrace_to_ary): ditto.

• thread.c (rb_threadptr_execute_interrupts): fix thread status overwritten issue. [Bug #7450] [ruby-core:50249]

• test/ruby/test_thread.rb (test_thread_status_raise_after_kill): test for the above.

• test/ruby/test_thread.rb (test_thread_status_in_trap): test for thread status in trap.

• test/ruby/test_thread.rb (test_status_and_stop_p): remove Thread.control_interrupt unsafe test. Thread#kill no longer changes thread status. Instead of, Thread#kill receiver changes their own status when receiving kill signal.

Revision 37931 - 11/28/2012 08:31 AM - kosaki (Motohiro KOSAKI)

• vm_core.h (enum rb_thread_status): remove THREAD_TO_KILL
• vm_core.h (struct rb_thread_struct): add to_kill field
• thread.c (terminate_i): convert THREAD_TO_KILL to to_kill.
• thread.c (rb_threadptr_to_kill): ditto.
• thread.c (rb_thread_kill): ditto.
• thread.c (rb_thread_wakeup_alive): ditto.
• thread.c (thread_list_i): ditto.
• thread.c (static const char): ditto.
• thread.c (thread_status_name): ditto.
• thread.c (rb_thread_status): ditto.
• thread.c (rb_thread_inspect): ditto.

• vm_backtrace.c (thread_backtrace_to_ary): ditto.

• thread.c (rb_threadptr_execute_interrupts): fix thread status overwritten issue. [Bug #7450] [ruby-core:50249]

• test/ruby/test_thread.rb (test_thread_status_raise_after_kill): test for the above.

• test/ruby/test_thread.rb (test_thread_status_in_trap): test for thread status in trap.

• test/ruby/test_thread.rb (test_status_and_stop_p): remove Thread.control_interrupt unsafe test. Thread#kill no longer changes thread status. Instead of, Thread#kill receiver changes their own status when receiving kill signal.
This issue was solved with changeset r37931.
Motohiro, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- vm_core.h (enum rb_thread_status): remove THREAD_TO_KILL
- vm_core.h (struct rb_thread_struct): add to_kill field
- thread.c (terminate_i): convert THREAD_TO_KILL to to_kill.
- thread.c (rb_threadptr_to_kill): ditto.
- thread.c (rb_thread_kill): ditto.
- thread.c (rb_thread_wakeup_alive): ditto.
- thread.c (thread_list_i): ditto.
- thread.c (static const char): ditto.
- thread.c (thread_status_name): ditto.
- thread.c (rb_thread_status): ditto.
- thread.c (rb_thread_inspect): ditto.

- vm_backtrace.c (thread_backtrace_to_ary): ditto.

- thread.c (rb_threadptr_execute_interrupts): fix thread status overwritten issue. [Bug #7450] [ruby-core:50249]

- test/ruby/test_thread.rb (test_thread_status_raise_after_kill): test for the above.


- test/ruby/test_thread.rb (test_status_and_stop_p): remove Thread.control_interrupt unsafe test. Thread#kill no longer changes thread status. Instead of, Thread#kill receiver changes their own status when receiving kill signal.