## Ruby master - Bug #14998

### Race conditions in MonitorMixin when interrupted

08/16/2018 12:51 PM - Eregon (Benoit Daloze)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>shugo (Shugo Maeda)</td>
</tr>
<tr>
<td>Target version:</td>
<td>ruby -v: ruby 2.6.0dev (2018-08-16 trunk 64394) [x86_64-linux]</td>
</tr>
<tr>
<td>Backport:</td>
<td>2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN</td>
</tr>
</tbody>
</table>

### Description

From [https://bugs.ruby-lang.org/issues/14859#note-9](https://bugs.ruby-lang.org/issues/14859#note-9)

The code of MonitorMixin#wait is:

```ruby
def wait(timeout = nil)
  @monitor.__send__(:mon_check_owner)
  count = @monitor.__send__(:mon_exit_for_cond)
  begin
    @cond.wait(@monitor.instance_variable_get(:@mon_mutex), timeout)
    return true
  ensure
    # What if Thread.raise happens here?
    @monitor.__send__(:mon_enter_for_cond, count)
  end
end
```

Probably this code needs to carefully use Thread.handle_interrupt.

### Associated revisions

Revision 6ec1720a - 11/28/2018 01:51 AM - shugo (Shugo Maeda)

lib/monitor.rb: avoid race conditions by Thread.handle_interrupt

Suggested by Benoit Daloze. [ruby-core:88502] [Bug #14998]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66061 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66061 - 11/28/2018 01:51 AM - shugo (Shugo Maeda)

lib/monitor.rb: avoid race conditions by Thread.handle_interrupt

Suggested by Benoit Daloze. [ruby-core:88502] [Bug #14998]

Revision 66061 - 11/28/2018 01:51 AM - shugo (Shugo Maeda)

lib/monitor.rb: avoid race conditions by Thread.handle_interrupt

Suggested by Benoit Daloze. [ruby-core:88502] [Bug #14998]

### History

#1 - 11/19/2018 08:57 AM - shugo (Shugo Maeda)

Eregon (Benoit Daloze) wrote:

From [https://bugs.ruby-lang.org/issues/14859#note-9](https://bugs.ruby-lang.org/issues/14859#note-9)

The code of MonitorMixin#wait is:

```ruby
def wait(timeout = nil)
  @monitor.__send__(:mon_check_owner)
  count = @monitor.__send__(:mon_exit_for_cond)
  begin
    @cond.wait(@monitor.instance_variable_get(:@mon_mutex), timeout)
    return true
  ensure
    # What if Thread.raise happens here?
    @monitor.__send__(:mon_enter_for_cond, count)
  end
end
```
ensure

# What if Thread#raise happens here?
@monitor.__send__(:mon_enter_for_cond, count)
end
end

Probably this code needs to carefully use Thread.handle_interrupt.

How about the following code?

def wait(timeout = nil)
  Thread.handle_interrupt { Exception => :never }
do
    count = @monitor.__send__(:mon_exit_for_cond)
  begin
    Thread.handle_interrupt { Exception => :immediate }
do
      @cond.wait(@monitor.instance_variable_get(:@mon_mutex), timeout)
    end
    return true
ensure
  @monitor.__send__(:mon_enter_for_cond, count)
end
end

#2 - 11/28/2018 01:51 AM - shugo (Shugo Maeda)
- Status changed from Open to Closed

Applied in changeset trunk|r66061.

lib/monitor.rb: avoid race conditions by Thread.handle_interrupt

Suggested by Benoit Daloze. [ruby-core:88502] [Bug #14998]