Ruby master - Bug #13320
rescue blocks get an entry in backtrace locations
03/16/2017 06:31 PM - bughit (bug hit)

<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>ko1 (Koichi Sasada)</td>
</tr>
<tr>
<td>Target version:</td>
<td>ruby -v:</td>
</tr>
<tr>
<td></td>
<td>ruby 2.4.0p0 (2016-12-24 revision 57164) [x86_64-linux]</td>
</tr>
<tr>
<td>Backport:</td>
<td>2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN</td>
</tr>
</tbody>
</table>

Description

```ruby
def foo
  puts caller(0)
end

def bar
  raise
  rescue
    foo
  end

  bar

  foo
  rescue in bar
  bar

this does not seem to make sense, conceptually a rescue block is not a separate callable entity, but just a section of a method, why should it get its own entry in the call stack?
```

History

#1 - 05/19/2017 06:26 AM - ko1 (Koichi Sasada)
- Assignee set to ko1 (Koichi Sasada)
- Status changed from Open to Feedback

```ruby
def foo
  puts caller(0)  # (1)
end

def bar
  p 0
  begin  # (3)
    p 1
    raise
    p 2
    rescue
      foo  # (2)
    end
  end

  bar  # (4)

=>
0
1
(1) t.rb:2:in `foo'
(2) t.rb:12:in `rescue in bar'
(3) t.rb:7:in `bar'
(4) t.rb:16:in `<main>,'
```

(3) shows they are in begin/end clause and
(2) shows they are in rescue clause.
does it make sense?

#2 - 07/26/2019 02:54 PM - jeremyevans0 (Jeremy Evans)
- Status changed from Feedback to Closed