Change NameError#local_variables to return the list of local variables where the method is raised

12/06/2015 06:15 AM - yuki24 (Yuki Nishijima)

Status: Closed
Priority: Normal
Assignee:
Target version:

Description
Sasada-san and I talked about this briefly a few weeks ago, but I also wanted to let others know about this.

This change will make it possible to pull out the list of local variables where the exception is raised without TracePoint. The did_you_mean gem uses TracePoint, and the current code looks like this:

```ruby
# lib/did_you_mean.rb
TracePoint.new(:raise) do |tp|
  e = tp.raised_exception
  if SPELL_CHECKERS.include?(e.class.to_s) && !e.instance_variable_defined?(:@frame_binding) || @frame_binding
    e.instance_variable_set(:@frame_binding, tp.binding)
  end
end.enable
```

# lib/did_you_mean/spell_checkers/name_error_checkers/variable_name_checker.rb
```ruby
def initialize(exception)
  @lvar_names = exception.frame_binding.local_variables
end
```

If we change NameError#local_variables as described it'll look like this:

```ruby
# lib/did_you_mean/spell_checkers/name_error_checkers/variable_name_checker.rb
def initialize(exception)
  @lvar_names = exception.local_variables
end
```

The problem with TracePoint is that it's still a little buggy (also see #11668, #11667) and also makes Ruby slower as reported by Sasada-san. I would like to change the behaviour of NameError#local_variables and remove the use of TracePoint from the gem entirely so we can make the gem much more stable.

Associated revisions
Revision 506b25aa - 12/08/2015 05:27 AM - nobu (Nobuyoshi Nakada)
```plaintext
error.c: name_err_local_variables
```n
- error.c (name_err_local_variables): new method
  NameError#local_variables for internal use only.
  [Feature #11777]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@52942 b2dd03c8-39d4-4d8f-98f8-823fe69b080e

Revision 52942 - 12/08/2015 05:27 AM - nobu (Nobuyoshi Nakada)
```plaintext
error.c: name_err_local_variables
```n
- error.c (name_err_local_variables): new method
  NameError#local_variables for internal use only.
  [Feature #11777]

09/02/2022 1/2
Revision 52942 - 12/08/2015 05:27 AM - nobu (Nobuyoshi Nakada)
error.c: name_err_local_variables

- error.c (name_err_local_variables): new method
  NameError#local_variables for internal use only.
  [Feature #11777]

Revision 52942 - 12/08/2015 05:27 AM - nobu (Nobuyoshi Nakada)
error.c: name_err_local_variables

- error.c (name_err_local_variables): new method
  NameError#local_variables for internal use only.
  [Feature #11777]

Revision 52942 - 12/08/2015 05:27 AM - nobu (Nobuyoshi Nakada)
error.c: name_err_local_variables

- error.c (name_err_local_variables): new method
  NameError#local_variables for internal use only.
  [Feature #11777]

Revision 52942 - 12/08/2015 05:27 AM - nobu (Nobuyoshi Nakada)
error.c: name_err_local_variables

- error.c (name_err_local_variables): new method
  NameError#local_variables for internal use only.
  [Feature #11777]

History

#1 - 12/06/2015 12:43 PM - nobu (Nobuyoshi Nakada)
Do you want to **add** NameError#local_variables?

#2 - 12/07/2015 05:26 AM - matz (Yukihiro Matsumoto)
It's OK to add. But I think we should mark it as 'internal use' in the reference.

Matz.

#3 - 12/08/2015 05:27 AM - nobu (Nobuyoshi Nakada)
- Status changed from Open to Closed

Applied in changeset r52942.

error.c: name_err_local_variables

- error.c (name_err_local_variables): new method
  NameError#local_variables for internal use only.
  [Feature #11777]

#4 - 12/08/2015 12:34 PM - znz (Kazuhiro NISHIYAMA)
NEWS does not mention about this feature.
Is it intended?

#5 - 12/08/2015 03:44 PM - nobu (Nobuyoshi Nakada)
Yes, this method is only for did_you_mean gem.