Ruby master - Feature #6012

Proc#source_location also return the column

02/14/2012 09:17 AM - rogerdpack (Roger Pack)

Status:	Assigned	
Priority:	Normal	
Assignee:	nobu (Nobuyoshi Nakada)	
Target version:		

Description

As originally suggested in http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/42418

Suggestion/feature request:
have #source_location also return the beginning column where it was defined.
["test.rb", 8, 33]

Thanks!
-roger-

Related issues:

Related to CommonRuby - Feature #8751: Add offsets to method#source_location	**Open**	**08/08/2013**

History

#1 - 02/14/2012 09:17 AM - rogerdpack (Roger Pack)

oops make that a feature request, but I'm unable to edit them myself.
Cheers!
-r

#2 - 02/14/2012 10:06 AM - nahi (Hiroshi Nakamura)

- Tracker changed from Bug to Feature

#3 - 02/26/2012 05:06 AM - ko1 (Koichi Sasada)

- Category set to core

- Assignee set to nobu (Nobuyoshi Nakada)

- Target version set to 2.0.0

#4 - 02/26/2012 11:32 PM - trans (Thomas Sawyer)

Would this effect Method#source_location too?

I'm not sure I am really digging this idea. First of all it means I have to go back and fix some code. Secondly it means I have to always worry about the additional piece of data even though most of the time it doesn't matter. And if the return can vary between 2 or 3 elements that's another thing to worry with.

On the other hand I can understand that it could be useful information in some cases.

In times like this that I think "Embrace the Object".

proc.source_location #=> #<SourceLocation @file="foo.rb" @line=12 @column=14>

And then a few different methods could provide that information in various useful forms.

proc.source_location.to_a #=> ["foo.rb", 12, 14]
proc.source_location.to_s #=> "foo.rb:12"
proc.source_location.values_at(:file, :line) #=> ["foo.rb", 12]

Or what have you.

#5 - 02/26/2012 11:33 PM - trans (Thomas Sawyer)

BTW & OT: When is any one going to explain how we format code examples as monospace text on this site?

#6 - 02/27/2012 05:23 AM - drbrain (Eric Hodel)

On Feb 26, 2012, at 6:33 AM, Thomas Sawyer wrote:

> BTW & OT: When is any one going to explain how we format code examples as monospace text on this site?

Click the RD button and use RD formatting (two spaces).

Here's a bash alias to help, which works for rdoc too.

alias rdindent='pr -l1 -o2'

#7 - 02/27/2012 09:39 AM - trans (Thomas Sawyer)

Thanks Eric! I ((*never*)) noticed that ((%RD%)) "button" before (hardly looks like a button).

Why did it put:

=begin
=end

In the textarea when I clicked on it? ... maybe I'll find out by submitting this...

=begin
What's with the =begin =end?

Testing 1 2 3...

Try ((*em*)) (({code})) ((|ls|)) ((%var%)).
=end

Sorry for the noise.

#8 - 02/27/2012 09:40 AM - trans (Thomas Sawyer)

Well, that failed miserably. LOL :-)

#9 - 03/18/2012 06:46 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

#10 - 10/25/2012 09:58 PM - yhara (Yutaka HARA)

- Target version changed from 2.0.0 to 2.6

#11 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)

- Target version deleted (2.6)

#12 - 12/28/2018 06:05 AM - mame (Yusuke Endoh)

Now the abstract syntax tree has column information, so we can implement this issue. We even add the last point of method.

```
# test.rb
◆def foo # ◆: line 2, column 0
end★     # ★: line 3, column 3
```

```
p method(:foo).source_location #=> ["test.rb", 2, 0, 3, 3]
```

#13 - 01/20/2019 10:14 PM - ioquatix (Samuel Williams)

If changing this API is too complicated due to backwards compatibility, why not introduce new more general API:

```
Method#source -> Source.new(path, line_number, line_count, code, ...)
```

Usage:

```
method.source.code
method.source.path
method.source.location -> [2, 0, 3, 3]
```

Maybe including byte offset and length would also be useful (for seek).

#14 - 02/12/2019 01:24 AM - ioquatix (Samuel Williams)

I also wish there was some meaningful implementation of proc.source.hash that was reasonably consistent across invocations of Ruby. Even if it was just best effort.

#15 - 02/12/2019 02:24 AM - ioquatix (Samuel Williams)

I was playing around with this idea trying to make an implementation of class Source.

Is the source file cached in Ruby? Or should we use File.read to load it into memory?

It seems inefficient for large files, to find line/column offset. It would be nice to have absolute offset to seek to.

Maybe it's possible for source_location to append one more thing - the actual source code - if possible. This would be useful for situations like eval, where you might define something for a path that doesn't actually exist, but the source code is still available.

#16 - 02/12/2019 07:49 AM - duerst (Martin Dürst)

ioquatix (Samuel Williams) wrote:

> I also wish there was some meaningful implementation of proc.source.hash that was reasonably consistent across invocations of Ruby. Even if it was just best effort.

Please make that a separate feature if you are serious about it.