Ruby - Bug #15789

Parse error when numbered parameter is used in a lambda that is a default value of other optarg

04/24/2019 04:18 PM - ibylich (Ilya Bylich)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.7.0dev (2019-04-24 trunk cf930985da) [x86_64-darwin18]	**Backport:**	2.4: UNKNOWN, 2.5: UNKNOWN, 2.6: UNKNOWN

Description

Sorry if the name of the ticket is not desccriptive

While working on backporting these commits into a parser gem:
https://github.com/ruby/ruby/commit/6ca9e7cc0785c33f6d382176dbd79d6c91db72fe
https://github.com/ruby/ruby/commit/ae07b66aaa092c59ac9d544c9b582712290dc357

... I've found a weird case that throws a SyntaxError:

```
> def m(a = ->{@1}); end
SyntaxError ((irb):10: ordinary parameter is defined)
def m(a = ->{@1}); end
             ^~
```

And same errors gets thrown when I pass a lambda with numparams to lambda optarg:

```
> ->(optarg = ->{@1}) {}
SyntaxError ((irb):1: ordinary parameter is defined)
->(optarg = ->{@1}) {}
               ^~
```

I guess the reason for that is that p->max_numparam should be organized as a stack, not a plain shared value.

History

#1 - 05/05/2019 04:28 AM - jeremyevans0 (Jeremy Evans)

- File fix-numbered-parameter-in-optarg-default-value.patch added

Attached is a one-line patch that fixes this issue, hopefully without causing additional issues:

```
m(a = ->{@1}); a end
m.call(1)
# => 1

m2 = ->(a = ->{@1}) {a}
m2.call.call(2)
# => 2

m3 = ->(a: ->{@1}) {a}
m3.call.call(3)
# => 3

def m(a = @1); a end
# SyntaxError ((irb):1: numbered parameter outside block

m2 = ->(a = @1) {a}
# SyntaxError ((irb):1: ordinary parameter is defined)
```

#2 - 05/05/2019 04:47 AM - jeremyevans0 (Jeremy Evans)

- File fix-numbered-parameter-in-optarg-default-value-v2.patch added

jeremyevans0 (Jeremy Evans) wrote:

> Attached is a one-line patch that fixes this issue, hopefully without causing additional issues:

Here's a patch that includes tests for this fix.

#3 - 05/05/2019 07:44 AM - nobu (Nobuyoshi Nakada)

jeremyevans0 (Jeremy Evans) wrote:

> Attached is a one-line patch that fixes this issue, hopefully without causing additional issues:

Thank you, committed the patch.

```
m2 = ->(a = @1) {a}
# SyntaxError ((irb):1: ordinary parameter is defined)
```

This should be the "outside block" error, I think.

#4 - 05/05/2019 11:36 PM - jeremyevans0 (Jeremy Evans)

nobu (Nobuyoshi Nakada) wrote:

> ```
> m2 = ->(a = @1) {a}
> # SyntaxError ((irb):1: ordinary parameter is defined)
> ```
>
> This should be the "outside block" error, I think.

I think the ordinary parameter is defined error makes sense if you consider the block starting at the -> and not the {. Especially when you consider the older lambda/proc syntax uses the same error:

```
lambda { |a=@1| a }
# SyntaxError ((irb):1: ordinary parameter is defined)
```

#5 - 05/06/2019 03:47 PM - nobu (Nobuyoshi Nakada)

jeremyevans0 (Jeremy Evans) wrote:

> I think the ordinary parameter is defined error makes sense if you consider the block starting at the -> and not the {. Especially when you consider the older lambda/proc syntax uses the same error:
>
> ```
> lambda { |a=@1| a }
> # SyntaxError ((irb):1: ordinary parameter is defined)
> ```

I've thought inside braces and outside braces are different.

That is, in this code, the @1 should mean the parameter of the outer proc.

```
proc {->(a = @1) {a}}.call(42)
```

But now I changed the mind.
The scope inside parentheses should be separated from the outside.

#6 - 05/09/2019 03:41 AM - jeremyevans0 (Jeremy Evans)

- Status changed from Open to Closed

nobu committed my fix for this at bb4ac7a6506971dc34b5656f1a69aadc7299fcab

Files

fix-numbered-parameter-in-optarg-default-value.patch	546 Bytes	05/05/2019	jeremyevans0 (Jeremy Evans)
fix-numbered-parameter-in-optarg-default-value-v2.patch	1.42 KB	05/05/2019	jeremyevans0 (Jeremy Evans)