Ruby trunk - Bug #15708

Implicit numbered argument decomposes an array

03/19/2019 11:08 AM - sawa (Tsuyoshi Sawada)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.7.0dev	**Backport:**	2.4: UNKNOWN, 2.5: UNKNOWN, 2.6: UNKNOWN

Description

In the following, @1 refers to the entire item iterated:

```
a = [1, 2, 3]
a.map{|x| x} # => [1, 2, 3]
a.map{@1} # => [1, 2, 3]
```

whereas in the following, @1 refers to the first item achieved by decomposing the item iterated, behaving the same as x given by |(x)| rather than by |x|:

```
a = [[1], [2], [3]]
a.map{|x| x} # => [[1], [2], [3]]
a.map{|(x)| x} # => [1, 2, 3]
a.map{@1} # => [1, 2, 3]
```

Is this intended?

Related issues:

Related to Ruby trunk - Misc #15723: Reconsider numbered parameters	**Feedback**

History

#1 - 03/19/2019 11:26 AM - mame (Yusuke Endoh)

Yes, it is actually intended. @1 is equivalent to |at1, at2, at3, ...|'s at1.

```
a = [1, 2, 3]
a.map{|x,| x} # => [1, 2, 3]
a.map{@1} # => [1, 2, 3]

a = [[1], [2], [3]]
a.map{|x,| x} # => [1, 2, 3]
a.map{@1} # => [1, 2, 3]
```

Honestly, I'm not a fan of the behavior, though.

#2 - 03/19/2019 11:57 AM - shevegen (Robert A. Heiler)

> Honestly, I'm not a fan of the behavior, though.

I think it comes down a lot to personal preferences. For example I actually like being able to use @NUMBER_HERE :) - although in "production" code, I may use the oldschool variant simply because, although it is longer, it is more explicit; but I think it is fine to have the possibility to use @NUMBER if one wants to. My primary reason for liking this, aside from once having suggested something vaguely similar :P, is that it reminds me of $1 $2 etc...

None of these will win in a beauty contest, but e. g. $1 $2 is very simple to type and also simple to remember - we just point to the matching capture group. (I actually also "manually" count within the code comment for regexes, e. g. I have lines like:

```
when /(.)(.)(.)/ # $3
```

or something like that, just so that I can quickly see which capture group I was interested in. Other personal preferences may include things such as endless range :D - or other changes. Anyway I digress.

I think the primary part of sawa's comment is to clarify on the behaviour, and ideally note it down as the correct behaviour too. Ruby users may otherwise be confused (and defined behaviour is also easier to test against).

Mame said that the behaviour is as expected; this is probably the correct answer. I would suggest to also include one of sawa's example in the official documentation - this may help avoid confusion for other ruby folks, when ruby 2.7 is released come xmas.

#3 - 03/19/2019 11:14 PM - hsbt (Hiroshi SHIBATA)

Sorry, your inconvenience experience. We have an issue of bugs.ruby-lang.org. I fixed it on this morning(JST).
I removed duplicated issues and copy from them.

nobu (Nobuyoshi Nakada) said:

Intended.

It equals

```
a.map{|x,| x} # => [1, 2, 3]
```

#4 - 04/08/2019 06:13 PM - Eregon (Benoit Daloze)

As I said in #15723,

I believe the |x,| behavior for @1 can only be considered a bug.

It prevents array_of_arrays.each { p @1 } to work correctly.
Why would we want to prevent that and make this pattern not general, dangerous, inconsistent and unusable for nested arrays?
This doesn't make any sense to me.

How can this be intended?
It ignores elements and make one of the simplest use of @1 wrong.

```
array_of_arrays = [[1,2], [3,4]]
array_of_arrays.each { p @1 }
# => 1
# => 3
```

The same happens for every block with @1 which passed value happens to be an Array.

This kind of behavior is what I learned in programming languages classes as a design flaw,
because it cannot handle properly elements independent of their representation.

#5 - 04/08/2019 06:35 PM - Eregon (Benoit Daloze)

FWIW, I would bet >99% of Rubyists would agree this is a bug: https://twitter.com/eregontp/status/1115318993299083265

#6 - 04/09/2019 03:18 PM - dgutov (Dmitry Gutov)

This is what happens when one syntactic sugar(*) collides with another.

(*) a.map { |x,| x } being a shorthand for a.map { |(x)| x } , and sometimes not, depending on the runtime values.

Neither of these are good, IMO (one for consistency and strong typing, and a lot of people have already expressed their feelings about the other).

It's too late to get rid of the first one, I think. But we can still reverse the decision on the new one.

#7 - 04/09/2019 07:41 PM - Eregon (Benoit Daloze)

For some reason, a reply on the tracker seems to have been lost, or removed.
I think it is highly relevant, so I'll quote it here:

sholden (Scott Holden) wrote:

> This is definitely not the behavior I would expect. In everything that I've seen, developers are describing the feature such that
>
> ```
> a.map{|x| x} == a.map{ @1 }
> ```

> This would be a very surprising behavior for people to stumble upon.

#8 - 04/09/2019 07:45 PM - Eregon (Benoit Daloze)

FWIW, the replies on my tweet above is some good sign that very few Rubyists expect this behavior and it breaks the basics assumptions of how the feature can be used.

#9 - 04/11/2019 09:42 PM - jeremyevans0 (Jeremy Evans)

- File single-implicit-arg-no-destructure.diff added

Attached is a patch that will turn off destructuring if the only implicit block variable is @1:

```
# equivalent to proc{|x| x}
proc{@1}.call([1,2])
# => [1, 2]

# equivalent to proc{|_,x| x}
proc{@2}.call([1,2])
# => 2

# equivalent to proc{|x,y| y; x}
proc{@2; @1}.call([1,2])
# => 1
```

I think this results in semantics that most people would expect, even if they don't like the implicit block argument syntax.

#10 - 04/12/2019 09:59 AM - nobu (Nobuyoshi Nakada)

jeremyevans0 (Jeremy Evans) wrote:

> Attached is a patch that will turn off destructuring if the only implicit block variable is @1:

```
-   args->nd_ainfo->rest_arg = excessed_comma;
+   if (max_numparam > 1) {
+       args->nd_ainfo->rest_arg = excessed_comma;
+   }
```

It can be done by just removing the line, regardless max_numparam.

#11 - 04/16/2019 12:20 AM - nobu (Nobuyoshi Nakada)

- Related to Misc #15723: Reconsider numbered parameters added

Files

single-implicit-arg-no-destructure.diff	471 Bytes	04/11/2019	jeremyevans0 (Jeremy Evans)