Ruby master - Feature #15573
Permit zero step in Numeric#step and Range#step
01/30/2019 03:32 AM - mrkn (Kenta Murata)

Status: Closed
Priority: Normal
Assignee: mrkn (Kenta Murata)
Target version:

Description
Numeric#step disallows zero in the 2nd argument, but it allows zero passed as the value of by: keyword argument. I confirmed that this inconsistency exists since 2.3. I want to allow zero in the 2nd argument, too.

```
>> 1.step(10, by: 0) { break }
=> nil
>> 1.step(10, 0) { break }
```

ArgumentError (step can't be 0)

Moreover, Range#step disallows zero if a block is given. I want to relax also this restriction.

```
>> (1..10).step(0) { break }
```

ArgumentError (step can't be 0)

Associated revisions
Revision f754b422 - 10/23/2020 06:26 AM - Kenta Murata
numeric.c, range.c: prohibit zero step
  • numeric.c: prohibit zero step in Numeric#step
  • range.c: prohibit zero step in Range#step
  • Fix ruby-spec

[Feature #15573]

History
#1 - 01/30/2019 03:40 AM - mrkn (Kenta Murata)
  - Description updated

#2 - 01/30/2019 04:16 AM - shyouhei (Shyouhei Urabe)
mrkn (Kenta Murata) wrote:
  I want to relax also this restriction.

Tell us why? Consistency?

03/12/2022
There is also inconsistency in that, without the block, 0 is allowed in these forms.

```
1.step(10, 0) # => ok
1.step(10, 0){} # => ArgumentError: step can't be 0
(1..10).step(0) # => ok
(1..10).step(0){} # => ArgumentError: step can't be 0
```

Since 0 can already sneak in, disallowing it just when the block is given makes it confusing. I agree that it should be relaxed.

Oops, sorry for the post if this was what mfkn has already pointed out.

---

#4 - 01/30/2019 10:44 AM - shevegen (Robert A. Heiler)

I can't say in which way to change it but I think this is indeed surprising behaviour and should be changed either way towards more consistency if possible.

#5 - 02/07/2019 06:12 AM - mrkn (Kenta Murata)

Tell us why? Consistency?

The reason is that zero step allows when the block is not given.

```
>> 1.step(10, 0).each { break }
=> nil
irb(main):005:0> (1..10).step(0)
=> ((1..10).step(0))
>> (1..10).step(0).each { break }
=> nil
```

#6 - 02/07/2019 07:58 AM - matz (Yukihiro Matsumoto)

For this case, consistency matters. I thought it is better to raise an error on zero step. But mrkn (Kenta Murata) can choose whatever behavior he believes right.

Matz.

#7 - 02/08/2019 01:03 AM - mrkn (Kenta Murata)

- Assignee changed from matz (Yukihiro Matsumoto) to mrkn (Kenta Murata)

#8 - 08/27/2020 10:29 AM - fatkodima (Dima Fatko)

Mathematically, it is allowed that step can be 0, but from practical stand point - this is an infinite loop waiting to be happen.

The original issue was about inconsistencies, so I decided to allow 0 for consistency (mainly because there is an ArithmeticSequence which allows that) and deprecate raising an exception. Or we can rewrite the ArithmeticSequence to raise and deprecate accepting 0 for all methods.

PR: [https://github.com/ruby/ruby/pull/3462](https://github.com/ruby/ruby/pull/3462)

#9 - 08/27/2020 01:59 PM - Dan0042 (Daniel DeLorme)

What is the purpose of a zero step? I cannot understand why/how it would be used.

#10 - 08/27/2020 03:42 PM - fatkodima (Dima Fatko)

Dan0042 (Daniel DeLorme) wrote in #note-9:

What is the purpose of a zero step? I cannot understand why/how it would be used.

This is basically an infinite generator of the same number. I haven't real usecases for this also.

#11 - 08/27/2020 04:12 PM - zverok (Victor Shepelev)

What is the purpose of a zero step? I cannot understand why/how it would be used.

This is basically an infinite generator of the same number. I haven't real usecases for this also.
For edge cases like this, they typically emerge on dynamic calculation (and if the calculation is mathematically sound, they probably shouldn't raise). Like, I dunno, something like (intentionally vague)

\[(\text{lower\_threshold}..\text{upper\_threshold}).\text{step}(\text{avg\_object\_size}).\text{take}(5)\]

...will in edge case produce 5 copies of lower\_threshold, which, depending on application logic, might be a useful result. So, ArgumentError("Nobody on the tracker have seen a sense in this") is probably undesirable.

#12 - 08/27/2020 04:17 PM - fatkodima (Dima Fatko)

\[zverok (Victor Shepelev)\]

Without take (or simply if step will have a block) this will be problematic. But, agreed, good example.

#13 - 10/23/2020 06:27 AM - Anonymous

- Status changed from Open to Closed

Applied in changeset [git|f754b422855131111092c0c147d744775cc4793f].

numeric.c, range.c: prohibit zero step

- numeric.c: prohibit zero step in Numeric#step
- range.c: prohibit zero step in Range#step
- Fix ruby-spec

[Feature #15573]