Ruby - Bug #17014

Range#minmax returns incorrect results on non-numeric exclusive ranges

07/05/2020 04:42 PM - sambostock (Sam Bostock)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.7.1	**Backport:**	2.5: DONTNEED, 2.6: DONTNEED, 2.7: DONE

Description

The implementation of Range#minmax added in d5c60214c45 causes the following incorrect behaviour:

```
('a'...'c').minmax # => ["a", ["a", "b"]]
```

instead of

```
('a'...'c').minmax # => ["a", "b"]
```

Cause

This is because the C implementation of Range#minmax (range_minmax) directly delegates to the C implementation of Range#min (range_min) and Range#max (range_max), without changing the execution context.

Range#max's C implementation (range_max), when given a non-numeric exclusive range, delegates to super, which is meant to call Enumerable#max. However, because range_max is called directly by range_minmax, super calls Enumerable#minmax instead, causing the incorrect nesting.

Resolution

- ruby/ruby#3285 fixed this bug by explicitly calling Range#min and Range#max, instead of delegating **directly** to range_min and range_max
- ruby/ruby#3286 followed up by replacing rb_intern("min") and rb_intern("max") in the new implementation with statics id_min and id_max
- ruby/ruby#3290 follows up by extracting range_min_internal and range_max_internal from range_min and range_max, and calling those directly from range_minmax

Associated revisions

Revision d24cce8e7f48b0b45f726f5f1ac7ff796f46ba72 - 07/19/2020 03:16 AM - nagachika (Tomoyuki Chikanaga)

merge revision(s) bf1a6771f305ea286a3ae575676924551c03e857,c1463625555b061a2b94c3b6c5581730b482a285: [Backport #17012] [Backport #17014]

```
    Fix non-numeric exclusive Range#minmax bug

    The implementation of Range#minmax added in d5c60214c45 causes the
    following incorrect behaviour:

      ('a'...'c').minmax => ["a", ["a", "b"]]

    instead of

      ('a'...'c').minmax => ["a", "b"]

    This is because the C implementation of Range#minmax (range_minmax)
    directly delegates to the C implementation of Range#min (range_min) and
    Range#max (range_max), without changing the execution context.

    Range#max's C implementation (range_max), when given a non-numeric
    exclusive range, delegates to super, which is meant to call
    Enumerable#max. However, because range_max is called directly by
    range_minmax, super calls Enumerable#minmax instead, causing the
    incorrect nesting.
```

```
    Perhaps it is possible to change the execution context in an optimized
    manner, but the simplest solution seems to be to just explicitly
    delegate from Range#minmax to Range#min and Range#max.

    Use static variables in Range#minmax
```

Revision d24cce8e7f48b0b45f726f5f1ac7ff796f46ba72 - 07/19/2020 03:16 AM - nagachika (Tomoyuki Chikanaga)

merge revision(s) bf1a6771f305ea286a3ae575676924551c03e857,c1463625555b061a2b94c3b6c5581730b482a285: [Backport #17012] [Backport #17014]

```
    Fix non-numeric exclusive Range#minmax bug

    The implementation of Range#minmax added in d5c60214c45 causes the
    following incorrect behaviour:

      ('a'...'c').minmax => ["a", ["a", "b"]]

    instead of

      ('a'...'c').minmax => ["a", "b"]

    This is because the C implementation of Range#minmax (range_minmax)
    directly delegates to the C implementation of Range#min (range_min) and
    Range#max (range_max), without changing the execution context.

    Range#max's C implementation (range_max), when given a non-numeric
    exclusive range, delegates to super, which is meant to call
    Enumerable#max. However, because range_max is called directly by
    range_minmax, super calls Enumerable#minmax instead, causing the
    incorrect nesting.

    Perhaps it is possible to change the execution context in an optimized
    manner, but the simplest solution seems to be to just explicitly
    delegate from Range#minmax to Range#min and Range#max.

    Use static variables in Range#minmax
```

Revision d24cce8e - 07/19/2020 03:16 AM - nagachika (Tomoyuki Chikanaga)

merge revision(s) bf1a6771f305ea286a3ae575676924551c03e857,c1463625555b061a2b94c3b6c5581730b482a285: [Backport #17012] [Backport #17014]

```
    Fix non-numeric exclusive Range#minmax bug

    The implementation of Range#minmax added in d5c60214c45 causes the
    following incorrect behaviour:

      ('a'...'c').minmax => ["a", ["a", "b"]]

    instead of

      ('a'...'c').minmax => ["a", "b"]

    This is because the C implementation of Range#minmax (range_minmax)
    directly delegates to the C implementation of Range#min (range_min) and
    Range#max (range_max), without changing the execution context.

    Range#max's C implementation (range_max), when given a non-numeric
    exclusive range, delegates to super, which is meant to call
    Enumerable#max. However, because range_max is called directly by
    range_minmax, super calls Enumerable#minmax instead, causing the
    incorrect nesting.

    Perhaps it is possible to change the execution context in an optimized
    manner, but the simplest solution seems to be to just explicitly
    delegate from Range#minmax to Range#min and Range#max.

    Use static variables in Range#minmax
```

History

#1 - 07/05/2020 07:53 PM - jeremyevans0 (Jeremy Evans)

- Status changed from Open to Closed

- Backport changed from 2.5: UNKNOWN, 2.6: UNKNOWN, 2.7: UNKNOWN to 2.5: UNKNOWN, 2.6: REQUIRED, 2.7: REQUIRED

#2 - 07/05/2020 07:54 PM - jeremyevans0 (Jeremy Evans)

- Backport changed from 2.5: UNKNOWN, 2.6: REQUIRED, 2.7: REQUIRED to 2.5: DONTNEED, 2.6: DONTNEED, 2.7: REQUIRED

#3 - 07/19/2020 03:16 AM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 2.5: DONTNEED, 2.6: DONTNEED, 2.7: REQUIRED to 2.5: DONTNEED, 2.6: DONTNEED, 2.7: DONE

ruby_2_7 d24cce8e7f48b0b45f726f5f1ac7ff796f46ba72 merged revision(s)
bf1a6771f305ea286a3ae575676924551c03e857,c1463625555b061a2b94c3b6c5581730b482a285.