Ruby master - Bug #18577

Range#include? returns wrong result for beginless range with exclusive string end

02/08/2022 04:33 PM - takaram (Takuya Aramaki)

Status: Closed
Priority: Normal
Assignee: 
Target version: 
ruby -v: ruby 3.1.0p0 (2021-12-25 revision fb4df44d16) [x86_64-linux]
Backport: 2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: UNKNOWN, 3.1: UNKNOWN

Description
ExCLUSIVE range should not include the end value, but (...'z').include? 'z' returns true. member? and === behave in the same way, while cover? does not.
This happens when the range is beginless and its end is string.

irb(main):001:0> (...'z').include? 'z'
=> true
irb(main):002:0> (...'z').member? 'z'
=> true
irb(main):003:0> (...'z') === 'z'
=> true
irb(main):004:0> (...'z').cover? 'z'
=> false

Associated revisions
Revision fd710d7e - 02/10/2022 03:47 AM - jeremyevans (Jeremy Evans)
Fix Range#include? for beginless exclusive string ranges
Previously, include? would return true for the end of the range, when it should return false because the range is exclusive.

Research and Analysis by Victor Shepelev.
Fixes [Bug #18577]

History
#1 - 02/09/2022 10:56 AM - zverok (Victor Shepelev)
It seems that the cause is unnecessary (?) specialization of Range#include? for String arguments.
I believe string ranges should just fallback to the default implementation:

(...Date.today) === Date.today
# => false
(...Date.today).cover? Date.today
# => false
(...Date.today).include? Date.today
# in `each': can't iterate from NilClass (TypeError)

The removal of specialization will also fix current (and possible future) discrepancies of Range#cover? and Range#=== for String ranges.
If it is not acceptable (some cases of backward compatibility?), this statement should be fixed.
cc jeremyevans0 (Jeremy Evans)

#2 - 02/09/2022 07:07 PM - jeremyevans0 (Jeremy Evans)
Zverok (Victor Shepelev) wrote in #note-1:
It seems that the cause is unnecessary (?) specialization of Range#include? for String arguments.
I believe string ranges should just fallback to the default implementation:

(...Date.today) === Date.today
# => false
The removal of specialization will also fix current (and possible future) discrepancies of Range#cover? and Range### for String ranges.

If you want to remove the specialization, that's something that probably should be discussed at a developer meeting. I am against removing it. Not because I think the specialization is a good idea, but because I don't think it's worth breaking backwards compatibility (raising TypeError for previously working code).

If it is not acceptable (some cases of backward compatibility?), this statement should be fixed.

I submitted a pull request to fix this: [https://github.com/ruby/ruby/pull/5541](https://github.com/ruby/ruby/pull/5541)

#3 - 02/10/2022 03:47 AM - jeremyevans (Jeremy Evans)
- Status changed from Open to Closed

Applied in changeset [git|fd710d7e9995679db80b7adf35bbda2cd4db90c6](https://github.com/ruby/ruby/tree/fd710d7e9995679db80b7adf35bbda2cd4db90c6)

Fix Range#include? for beginless exclusive string ranges

Previously, include? would return true for the end of the range, when it should return false because the range is exclusive.

Research and Analysis by Victor Shepelev.

Fixes [Bug #18577](https://github.com/ruby/ruby/issues/18577)