Ruby master - Feature #10506
ImprvEnumeator.any? behavior
11/13/2014 12:40 PM - thomas.sevestre (Thomas Sevestre)

Status: Rejected
Priority: Normal
Assignee: matz (Yukihiro Matsumoto)
Target version: 3.0

Description
IMHO the API Enumerable.any? behavior could be improved in the following manner:

any? { |obj| block } -> true or false
any? -> an_enumerator

This would allow to do things like

[].any?.with_index { |obj, index| }

And it would be more consistent with collect and other Enumerable functions:

collect { |obj| block } -> array
collect -> an_enumerator

History
#1 - 11/13/2014 02:21 PM - marcandre (Marc-Andre Lafortune)
- Category set to core
- Assignee set to matz (Yukihiro Matsumoto)
- Target version set to 3.0

It probably should have returned an enumerator, but that wasn't as systematic then.

The problem with your proposal is breaking compatibility. I'm quite certain this will be rejected, but I'll let Matz do it.

In the meantime, know that it is easy to get what you want in most cases.

For example:

[].each_with_index.any?{ |obj, index| }

You can also use to_enum to get what you want:

[].to_enum(:any?).with_index{ |obj, index| }

#2 - 11/13/2014 03:26 PM - Eregon (Benoit Daloze)
- Status changed from Open to Rejected

In other words, any? already has behavior when there is no block:
It uses an identity block { |e| e } as said in the documentation.
So it checks if there is any truthy member in the enumerable.

Changing it is unbearable for compatibility, so I close the issue.
Many users already use it for !enum.empty?, although it is not semantically the same.
We would not want to potentially break every use of any? with no block.
If you insist, please propose a nice transition path, but I fear it would take very long to keep some kind of compatibility (first deprecate, then remove, wait a while and add it with new behavior).