Ruby master - Feature #14136

Implement #empty? on more classes

11/28/2017 03:12 PM - mikegee (Michael Gee)

Status:	Open	
Priority:	Normal	
Assignee:		
Target version:		

Description

Hi Ruby Friends!

Rubocop prefers #empty? over length == 0 and size == 0, which is great for String, Array, Hash, etc. It would be nice if more classes implemented #empty? for consistency.

See related discussion at https://github.com/bbatsov/rubocop/issues/2841.

I started this work at https://github.com/ruby/ruby/pull/1759

Thanks!

History

#1 - 11/28/2017 04:23 PM - shevegen (Robert A. Heiler)

The issue discussion there is very long.

For the purpose of this thread here, can you list which specific classes
should have ".empty?()"?

String, Array, Hash have this method so far.

I can understand to some extent that **Tempfile** should have this behaviour.

From the thread I see that you also want to include **StringIO** and
File::Stat, that is, to have a **.empty?** method.

I do not have any real particular contra opinion; neither do I have a
particular pro opinion. I just would like to have this all in the thread
here, for ease of discussion on MRI. (The rubocop discussion appears to
be more akin to rubocop itself; whereas on MRI one ultimately has to
convince matz and respectively the ruby core team).

Also it may be useful to state why .empty? may be useful for these
classes, for real use cases; I mean, I get the point of avoiding
(.size == 0) checks but I think the MRI team also wants to know
how frequent the usage is.

For String, Array and Hash, that use case is obviously well covered.

#2 - 11/29/2017 12:34 AM - mikegee (Michael Gee)

Sorry, that Rubocop issue does have a bunch of unrelated discussion. I should have summarized the parts I was referring to. Thanks for your feedback.

The discussion began because a user reported Rubocop complaining about this code:

File.stat(manifest_file).size == 0

Rubocop would prefer that written as File.stat(manifest_file).empty? (Because Rubocop assumes objects with #size and #length also have #empty?, like String, Array, and Hash do.) But, File::Stat does not have an #empty? method, so the suggestion raises NoMethodError.

I agree that the change suggested by Rubocop would improve this code's clarity. The problem is that not all classes with #size and #length also have #empty?.

I claim that adding #empty? to all these classes improves clarity without any significant downside.

I implemented #empty? on the 3 classes mentioned by Rubocop users in that issue.

Thank you for your consideration.

#3 - 11/29/2017 06:52 AM - nobu (Nobuyoshi Nakada)

mikegee (Michael Gee) wrote:

> The discussion began because a user reported Rubocop complaining about this code:
>
> File.stat(manifest_file).size == 0
>
> Rubocop would prefer that written as File.stat(manifest_file).empty? (Because Rubocop assumes objects with #size and #length also have #empty?, like String, Array, and Hash do.) But, File::Stat does not have an #empty? method, so the suggestion raises NoMethodError.

It's a Rubocop's issue.
I'd suggest File.empty? instead.

#4 - 11/29/2017 09:32 AM - Hanmac (Hans Mackowiak)

nobu (Nobuyoshi Nakada) wrote:

> I'd suggest File.empty? instead.

File.empty? might not always work because you might want lstat or other stat objects

but File::Stat#size? might be interesting, it does return nil on empty size

mikegee (Michael Gee) i think you want this: File::Stat#zero?

#5 - 11/29/2017 02:42 PM - mikegee (Michael Gee)

There seems to be some confusion about what I'm asking for here. I know how to use these classes to make my code work. I'm not asking for help using the existing methods.

I am proposing that all classes that implement #size or #length should also implement #empty? to let developers write clearer code.

#6 - 11/29/2017 09:33 PM - phluid61 (Matthew Kerwin)

mikegee (Michael Gee) wrote:

> I am proposing that all classes that implement #size or #length should also implement #empty? to let developers write clearer code.

This is one of the Rubocop cops I always disable, because I don't find #empty? conceptually clearer (or necessarily even accurate) unless that's what I wrote in the first place.

File.stat is a perfect example: the status object isn't empty. Adding this method would make Ruby code *less* clear, more idiosyncratic.

#7 - 09/12/2018 05:09 AM - bozhidar (Bozhidar Batsov)

phluid61 (Matthew Kerwin) wrote:

> mikegee (Michael Gee) wrote:
>
>> I am proposing that all classes that implement #size or #length should also implement #empty? to let developers write clearer code.
>
> This is one of the Rubocop cops I always disable, because I don't find #empty? conceptually clearer (or necessarily even accurate) unless that's what I wrote in the first place.
>
> File.stat is a perfect example: the status object isn't empty. Adding this method would make Ruby code *less* clear, more idiosyncratic.

Yeah, in this case I'd argue that it's better to use some top-level methods of File instead, but in general every object that has the notion of size should also have the option of emptiness. That's common sense and not adhering it to in the default API simply frustrates Ruby developers everywhere.

#8 - 09/12/2018 05:12 AM - bozhidar (Bozhidar Batsov)

bozhidar (Bozhidar Batsov) wrote:

> phluid61 (Matthew Kerwin) wrote:
>
>> mikegee (Michael Gee) wrote:

> > > I am proposing that all classes that implement #size or #length should also implement #empty? to let developers write clearer code.
> >
> > This is one of the Rubocop cops I always disable, because I don't find #empty? conceptually clearer (or necessarily even accurate) unless that's what I wrote in the first place.
> >
> > File.stat is a perfect example: the status object isn't empty. Adding this method would make Ruby code *less* clear, more idiosyncratic.
>
> Yeah, in this case I'd argue that it's better to use some top-level methods of File instead, but in general every object that has the notion of size should also have the option of emptiness. That's common sense and not adhering it to in the default API simply frustrates Ruby developers everywhere.

Also I'm curious who'd claim that adding an empty method doesn't make sense for something like Tempfile or StringIO.

#9 - 09/12/2018 05:44 AM - jeremyevans0 (Jeremy Evans)

bozhidar (Bozhidar Batsov) wrote:

> > Yeah, in this case I'd argue that it's better to use some top-level methods of File instead, but in general every object that has the notion of size should also have the option of emptiness. That's common sense and not adhering it to in the default API simply frustrates Ruby developers everywhere.

I don't think it's necessarily common sense. It makes sense for collections to have an empty? method. However, not all objects with a size method should necessarily have an implementation of empty?. Pants#size and Horse#size are both methods that could make sense, but Pants#empty? and Horse#empty? may not.

> Also I'm curious who'd claim that adding an empty method doesn't make sense for something like Tempfile or StringIO.

Tempfile#empty? and StringIO#empty? should only be defined if File#empty? is defined, because both Tempfile and StringIO should try to implement the File API to the extent that doing so makes sense. I'm not sure whether File#empty? makes sense. Some people may consider a file of non-zero length with all "\0" or " " bytes to be considered empty. But I guess the same argument could be made that an array of all nil values could be considered empty by some developers.

I will say I haven't been frustrated by the lack of an empty? method on any of the classes being discussed. I'm also of the opinion that adding methods to core/stdlib classes just to appease a static code analyzer is a bad idea. If empty? should be added to any classes, each case should be discussed individually on its own merits, with reasoning given describing why empty? makes semantic sense for the class.

#10 - 10/27/2018 07:20 AM - bozhidar (Bozhidar Batsov)

jeremyevans0 (Jeremy Evans) wrote:

> bozhidar (Bozhidar Batsov) wrote:
>
> > > Yeah, in this case I'd argue that it's better to use some top-level methods of File instead, but in general every object that has the notion of size should also have the option of emptiness. That's common sense and not adhering it to in the default API simply frustrates Ruby developers everywhere.
>
> I don't think it's necessarily common sense. It makes sense for collections to have an empty? method. However, not all objects with a size method should necessarily have an implementation of empty?. Pants#size and Horse#size are both methods that could make sense, but Pants#empty? and Horse#empty? may not.
>
> > Also I'm curious who'd claim that adding an empty method doesn't make sense for something like Tempfile or StringIO.
>
> Tempfile#empty? and StringIO#empty? should only be defined if File#empty? is defined, because both Tempfile and StringIO should try to implement the File API to the extent that doing so makes sense. I'm not sure whether File#empty? makes sense. Some people may consider a file of non-zero length with all "\0" or " " bytes to be considered empty. But I guess the same argument could be made that an array of all nil values could be considered empty by some developers.
>
> I will say I haven't been frustrated by the lack of an empty? method on any of the classes being discussed. I'm also of the opinion that adding methods to core/stdlib classes just to appease a static code analyzer is a bad idea. If empty? should be added to any classes, each case should be discussed individually on its own merits, with reasoning given describing why empty? makes semantic sense for the class.

Yeah, I completely agree with you. Excellent point about using size in different contexts! Generally I'm not trying to suggest changes for the sake of making RuboCop happy, but for the sake of having more consistent and pleasant APIs.