Ruby trunk - Feature #5009

Bang method (!) consistency in Ruby 2 API

07/10/2011 06:07 AM - sunaku (Suraj Kurapati)

Status:	Closed
Priority:	Normal
Assignee:	
Target version:	Next Major

Description

=begin
Hello,

In Ruby 2, please use bang methods (those that end with !) consistently in the API. For example, the Ruby 1 API inconsistently names the following "destructive" methods plainly (without a bang):

- Array#push
- Array#pop
- Array#shift
- Array#unshift
- Array#concat
- Array#delete
- Array#clear

Please convert these into bang methods in the Ruby 2 API.

Thanks for your consideration.
=end

History

#1 - 07/10/2011 06:12 AM - adgar (Michael Edgar)

This is a common misconception about the use of bang methods in Ruby. Bang does not indicate that a method mutates its receiver, merely that it should be used with caution.

A good writeup on this can be found here: http://dablog.rubypal.com/2007/8/15/bang-methods-or-danger-will-rubyist

From the conclusion:

> If you let go of the notion that ! means destructive, and if you think through the naming and pairing of dangerous and non-dangerous methods, you'll see how valuable a flag the ! can be.

#2 - 07/10/2011 06:20 AM - antares (Michael Klishin)

If this is a proposal for renaming aforementioned methods, it will break a lot of existing code. If the idea is to add aliases, it will introduce Ruby 2-specific methods that will not be used for a really long time by developers who want to maintain compatibility with multiple Ruby versions.

#3 - 07/10/2011 04:51 PM - naruse (Yui NARUSE)

-1
ActiveRecord's save/save! is also a good example.

#4 - 07/12/2011 03:06 PM - duerst (Martin Dürst)

Just for the record, there is one method that I think we should add a bang if we still could, and this is String#force_encoding. The reason isn't that it changes String itself (which it does), but that I have come across quite some places where I use string.dup.force_encoding. It would have been best to use force_encoding! for what's now force_encoding, and force_encoding for the copying variant. But I agree with both Michaels and Yui that it's too late to change.

#5 - 07/12/2011 03:16 PM - naruse (Yui NARUSE)

Martin Dürst wrote:

> Just for the record, there is one method that I think we should add a bang if we still could, and this is String#force_encoding. The reason isn't that it changes String itself (which it does), but that I have come across quite some places where I use string.dup.force_encoding. It would have

been best to use force_encoding! for what's now force_encoding, and force_encoding for the copying variant. But I agree with both Michaels and Yui that it's too late to change.

About String#force_encoding, I think add an alias named String#encoding= may be good.
But it's still under consideration.

#6 - 07/15/2011 05:00 AM - sunaku (Suraj Kurapati)

Interesting, the Rails' save/save! example makes sense.

And why worry about keeping backwards compatibility?

Ruby 2 can break it, in the interest of removing cruft.

#7 - 07/16/2011 12:29 AM - naruse (Yui NARUSE)

- Status changed from Open to Closed

Close this; force_encoding is another problem.