Ruby master - Feature #17147

New method to get frozen strings from String objects

09/04/2020 07:36 AM - tagomoris (Satoshi TAGOMORI)

Status:	Feedback
Priority:	Normal
Assignee:	
Target version:	

Description

Object deserializer (like JSON, MessagePack) instantiates many String objects (as keys of Hash objects), and many of those are in a set of names. (So the total number of keys is not infinite.)
In such use-case, the object deserializer is generating many string object instances. Those are impacting the VM performance (mainly for GC pressure), especially in the case when those objects keep staying in memory for a long time.

If we can de-duplicate those instances at the instantiation, we can reduce the performance impact of object instantiation. It can be achieved if we have C API to generate frozen strings.
On the other hand, if we have Ruby methods to get frozen strings from strings, we can implement object deserializer in Ruby. It should be valuable for many Ruby users because of MJIT optimization in the future (And that method can be used from C ext modules too).

So, in general, a Ruby method to get frozen (de-duplicated) strings will be valuable and can improve the Ruby performance so much. Deserializers (JSON, MessagePack) are used everywhere.

Related issues:

Related to Ruby master - Feature #13381: [PATCH] Expose rb_fstring and its fa...	**Closed**
Is duplicate of Ruby master - Feature #13077: [PATCH] introduce String#fstrin...	**Closed**

History

#1 - 09/04/2020 07:39 AM - tagomoris (Satoshi TAGOMORI)

I don't care of the name of that method, but here's some example if the discussion stops without options:

- String#frozen_string
- String#as_frozen_string
- ObjectSpace.get_frozen_string(str)

#2 - 09/04/2020 07:56 AM - shyouhei (Shyouhei Urabe)

Understand the needs. Not sure if what is needed is actually the concept called “frozen” though.

#3 - 09/04/2020 09:11 AM - byroot (Jean Boussier)

tagomoris (Satoshi TAGOMORI) I've been advocating for exposing the fstring family of function exactly for this. We load a lot of data from flat files, and it cost us a lot of memory alloc and then CPU to deduplicate them. And I was planning to submit patches to message pack once these API would be available.

However on the Ruby side, I'm not sure what your proposal do differently from String#-@.

#4 - 09/04/2020 09:58 AM - naruse (Yui NARUSE)

- Is duplicate of Feature #13077: [PATCH] introduce String#fstring method added

#5 - 09/04/2020 10:00 AM - naruse (Yui NARUSE)

- Related to Feature #13381: [PATCH] Expose rb_fstring and its family to C extensions added

#6 - 09/04/2020 10:43 AM - naruse (Yui NARUSE)

- Status changed from Open to Closed

The feature is provided by -str.

#7 - 09/05/2020 12:10 AM - shyouhei (Shyouhei Urabe)

- Status changed from Closed to Feedback

Not also sure if String#-@ saves the OP’s situation, though. The method dedups string contents but has nothing to do with GC pressures.

Can you test if String#-@ works?

#8 - 09/06/2020 03:43 PM - byroot (Jean Boussier)

> Not also sure if String#-@ saves the OP’s situation, though

String#-@ doesn't as it's too late (the string was allocated already). But exposing rb_fstring() would, at in some specific use cases it could drastically reduce allocations.

#9 - 09/07/2020 01:41 AM - tagomoris (Satoshi TAGOMORI)

Thank you for the beedbacks! I missed considering about String#-@ method. It looks worth to try, so I'll evaluate that option on the workload of msgpack-ruby (and Fluentd possibly).