

Ruby master - Feature #9634

[PATCH]Symbol GC

03/13/2014 08:02 AM - authorNari (Narihiro Nakamura)

Status:	Closed
Priority:	Normal
Assignee:	matz (Yukihiro Matsumoto)
Target version:	2.2.0

Description

I've written a patch to collect most symbols.

PATCH: https://github.com/authorNari/ruby/compare/4a91fb7a45f0e3c...symbol_gc.patch

Summary

- Most symbols in Ruby level are GC-able generated by #to_sym, #intern, etc..
- Exclude a symbol which is translated ID in C-level from GC-able symbols
- Keep Ruby's C extension compatibility
- Pass make test-all

Benchmark

A benchmark program is here.

```
obj = Object.new
100_000.times do |i|
  obj.respond_to?("sym#{i}".to_sym)
end
GC.start
puts "symbol : #{Symbol.all_symbols.size}"

% time RBENV_VERSION=ruby-r45059 ruby -v /tmp/a.rb
ruby 2.2.0dev (2014-02-20 trunk 45059) [x86_64-linux]
symbol : 102416
0.24s user 0.01s system 91% cpu 0.272 total

% time RBENV_VERSION=symgc ruby -v /tmp/a.rb
ruby 2.2.0dev (2014-02-20 trunk 45059) [x86_64-linux]
symbol : 2833
0.21s user 0.01s system 90% cpu 0.247 total
```

The total number of symbols is declined.
The total time of symgc version is improved because Full GC pressure has been reduced.

The result of make benchmark.

<https://gist.github.com/authorNari/9359704>

There is no significant slowdown.

(I would welcome to try an additional benchmark and report)

Implementation Detail

I classify Dynamic symbol and Static symbol.

- Static symbol
 - Generated by rb_itnern()
 - A sequential unique number as in the past.
 - Not GC-able

symbol static symbol dynamic symbol

- static symbol

- rb_itern
- GC
- 147 ID

- dynamic symbol

- Ruby #to_sym, #intern
- RVALUE
- GC
- C ID SYM2ID pindown GC
- Ruby ID pindown

GC Heroku

GC Heroku

Associated revisions

Revision 90b70738 - 03/26/2014 04:57 AM - nari

- parse.y: support Symbol GC. [ruby-trunk Feature #9634]
See this ticket about Symbol GC.
- include/ruby/ruby.h:
Declare few functions.
 - rb_sym2id: almost same as old SYM2ID but support dynamic symbols.
 - rb_id2sym: almost same as old ID2SYM but support dynamic symbols.
 - rb_sym2str: almost same as rb_id2str(SYM2ID(sym)) but not pin down a dynamic symbol. Declare a new struct.
 - struct RSymbol: represents a dynamic symbol as object in Ruby's heaps. Add few macros.
 - STATIC_SYM_P: check a static symbol.
 - DYNAMIC_SYM_P: check a dynamic symbol.
 - RSYMBOL: cast to RSymbol
- gc.c: declare RSymbol. support T_SYMBOL.
- internal.h: Declare few functions.
 - rb_gc_free_dsymbol: free up a dynamic symbol. GC call this function at a sweep phase.
 - rb_str_dynamic_intern: convert a string to a dynamic symbol.
 - rb_check_id_without_pindown: not pinning function.
 - rb_sym2id_without_pindown: ditto.
 - rb_check_id_cstr_without_pindown: ditto.
- string.c (Init_String): String#intern and String#to_sym use rb_str_dynamic_intern.
- template/id.h.templ: use LSB of ID as a flag for determining a static symbol, so we shift left other ruby_id_types.
- string.c: use rb_sym2str instead rb_id2str(SYM2ID(sym)) to avoid pinning.

- load.c: use `xx_without_pindown` function at creating temporary ID to avoid pinning.
- object.c: ditto.
- sprintf.c: ditto.
- struct.c: ditto.
- thread.c: ditto.
- variable.c: ditto.
- vm_method.c: ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@45426 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 45426 - 03/26/2014 04:57 AM - nari

- parse.y: support Symbol GC. [ruby-trunk Feature #9634]
See this ticket about Symbol GC.
- include/ruby/ruby.h:
 - Declare few functions.
 - `rb_sym2id`: almost same as old `SYM2ID` but support dynamic symbols.
 - `rb_id2sym`: almost same as old `ID2SYM` but support dynamic symbols.
 - `rb_sym2str`: almost same as `rb_id2str(SYM2ID(sym))` but not pin down a dynamic symbol. Declare a new struct.
 - struct `RSymbol`: represents a dynamic symbol as object in Ruby's heaps. Add few macros.
 - `STATIC_SYM_P`: check a static symbol.
 - `DYNAMIC_SYM_P`: check a dynamic symbol.
 - `RSYMBOL`: cast to `RSymbol`
- gc.c: declare `RSymbol`. support `T_SYMBOL`.
- internal.h: Declare few functions.
 - `rb_gc_free_dsymbols`: free up a dynamic symbol. GC call this function at a sweep phase.
 - `rb_str_dynamic_intern`: convert a string to a dynamic symbol.
 - `rb_check_id_without_pindown`: not pinning function.
 - `rb_sym2id_without_pindown`: ditto.
 - `rb_check_id_cstr_without_pindown`: ditto.
- string.c (`Init_String`): `String#intern` and `String#to_sym` use `rb_str_dynamic_intern`.
- template/id.h.templ: use LSB of ID as a flag for determining a static symbol, so we shift left other `ruby_id_types`.
- string.c: use `rb_sym2str` instead `rb_id2str(SYM2ID(sym))` to avoid pinning.
- load.c: use `xx_without_pindown` function at creating temporary ID to avoid pinning.
- object.c: ditto.
- sprintf.c: ditto.
- struct.c: ditto.
- thread.c: ditto.
- variable.c: ditto.
- vm_method.c: ditto.

- parse.y: support Symbol GC. [ruby-trunk Feature #9634]
See this ticket about Symbol GC.
- include/ruby/ruby.h:
Declare few functions.
 - rb_sym2id: almost same as old SYM2ID but support dynamic symbols.
 - rb_id2sym: almost same as old ID2SYM but support dynamic symbols.
 - rb_sym2str: almost same as rb_id2str(SYM2ID(sym)) but not pin down a dynamic symbol. Declare a new struct.
 - struct RSymbol: represents a dynamic symbol as object in Ruby's heaps. Add few macros.
 - STATIC_SYM_P: check a static symbol.
 - DYNAMIC_SYM_P: check a dynamic symbol.
 - RSYMBOL: cast to RSymbol
- gc.c: declare RSymbol. support T_SYMBOL.
- internal.h: Declare few functions.
 - rb_gc_free_dsymbols: free up a dynamic symbol. GC call this function at a sweep phase.
 - rb_str_dynamic_intern: convert a string to a dynamic symbol.
 - rb_check_id_without_pindown: not pinning function.
 - rb_sym2id_without_pindown: ditto.
 - rb_check_id_cstr_without_pindown: ditto.
- string.c (Init_String): String#intern and String#to_sym use rb_str_dynamic_intern.
- template/id.h.templ: use LSB of ID as a flag for determining a static symbol, so we shift left other ruby_id_types.
- string.c: use rb_sym2str instead rb_id2str(SYM2ID(sym)) to avoid pinning.
- load.c: use xx_without_pindown function at creating temporary ID to avoid pinning.
- object.c: ditto.
- sprintf.c: ditto.
- struct.c: ditto.
- thread.c: ditto.
- variable.c: ditto.
- vm_method.c: ditto.

- parse.y: support Symbol GC. [ruby-trunk Feature #9634]
See this ticket about Symbol GC.
- include/ruby/ruby.h:
Declare few functions.
 - rb_sym2id: almost same as old SYM2ID but support dynamic symbols.
 - rb_id2sym: almost same as old ID2SYM but support dynamic symbols.
 - rb_sym2str: almost same as rb_id2str(SYM2ID(sym)) but not pin down a dynamic symbol. Declare a new struct.
 - struct RSymbol: represents a dynamic symbol as object in Ruby's heaps. Add few macros.
 - STATIC_SYM_P: check a static symbol.
 - DYNAMIC_SYM_P: check a dynamic symbol.
 - RSYMBOL: cast to RSymbol

- gc.c: declare RSymbol. support T_SYMBOL.
- internal.h: Declare few functions.
 - rb_gc_free_dsymbols: free up a dynamic symbol. GC call this function at a sweep phase.
 - rb_str_dynamic_intern: convert a string to a dynamic symbol.
 - rb_check_id_without_pindown: not pinning function.
 - rb_sym2id_without_pindown: ditto.
 - rb_check_id_cstr_without_pindown: ditto.
- string.c (Init_String): String#intern and String#to_sym use rb_str_dynamic_intern.
- template/id.h.templ: use LSB of ID as a flag for determining a static symbol, so we shift left other ruby_id_types.
- string.c: use rb_sym2str instead rb_id2str(SYM2ID(sym)) to avoid pinning.
- load.c: use xx_without_pindown function at creating temporary ID to avoid pinning.
- object.c: ditto.
- sprintf.c: ditto.
- struct.c: ditto.
- thread.c: ditto.
- variable.c: ditto.
- vm_method.c: ditto.

Revision 45426 - 03/26/2014 04:57 AM - nari

- parse.y: support Symbol GC. [ruby-trunk Feature #9634]
See this ticket about Symbol GC.
- include/ruby/ruby.h:
 - rb_sym2id: almost same as old SYM2ID but support dynamic symbols.
 - rb_id2sym: almost same as old ID2SYM but support dynamic symbols.
 - rb_sym2str: almost same as rb_id2str(SYM2ID(sym)) but not pin down a dynamic symbol. Declare a new struct.
 - struct RSymbol: represents a dynamic symbol as object in Ruby's heaps. Add few macros.
 - STATIC_SYM_P: check a static symbol.
 - DYNAMIC_SYM_P: check a dynamic symbol.
 - RSYMBOL: cast to RSymbol
- gc.c: declare RSymbol. support T_SYMBOL.
- internal.h: Declare few functions.
 - rb_gc_free_dsymbols: free up a dynamic symbol. GC call this function at a sweep phase.
 - rb_str_dynamic_intern: convert a string to a dynamic symbol.
 - rb_check_id_without_pindown: not pinning function.
 - rb_sym2id_without_pindown: ditto.
 - rb_check_id_cstr_without_pindown: ditto.
- string.c (Init_String): String#intern and String#to_sym use rb_str_dynamic_intern.
- template/id.h.templ: use LSB of ID as a flag for determining a static symbol, so we shift left other ruby_id_types.

- string.c: use rb_sym2str instead rb_id2str(SYM2ID(sym)) to avoid pinning.
- load.c: use xx_without_pindown function at creating temporary ID to avoid pinning.
- object.c: ditto.
- sprintf.c: ditto.
- struct.c: ditto.
- thread.c: ditto.
- variable.c: ditto.
- vm_method.c: ditto.

Revision 45426 - 03/26/2014 04:57 AM - nari

- parse.y: support Symbol GC. [ruby-trunk Feature #9634]
See this ticket about Symbol GC.
- include/ruby/ruby.h:
 - Declare few functions.
 - rb_sym2id: almost same as old SYM2ID but support dynamic symbols.
 - rb_id2sym: almost same as old ID2SYM but support dynamic symbols.
 - rb_sym2str: almost same as rb_id2str(SYM2ID(sym)) but not pin down a dynamic symbol. Declare a new struct.
 - struct RSymbol: represents a dynamic symbol as object in Ruby's heaps. Add few macros.
 - STATIC_SYM_P: check a static symbol.
 - DYNAMIC_SYM_P: check a dynamic symbol.
 - RSYMBOL: cast to RSymbol
- gc.c: declare RSymbol. support T_SYMBOL.
- internal.h: Declare few functions.
 - rb_gc_free_dsymbols: free up a dynamic symbol. GC call this function at a sweep phase.
 - rb_str_dynamic_intern: convert a string to a dynamic symbol.
 - rb_check_id_without_pindown: not pinning function.
 - rb_sym2id_without_pindown: ditto.
 - rb_check_id_cstr_without_pindown: ditto.
- string.c (Init_String): String#intern and String#to_sym use rb_str_dynamic_intern.
- template/id.h.templ: use LSB of ID as a flag for determining a static symbol, so we shift left other ruby_id_types.
- string.c: use rb_sym2str instead rb_id2str(SYM2ID(sym)) to avoid pinning.
- load.c: use xx_without_pindown function at creating temporary ID to avoid pinning.
- object.c: ditto.
- sprintf.c: ditto.
- struct.c: ditto.
- thread.c: ditto.
- variable.c: ditto.
- vm_method.c: ditto.

- parse.y: support Symbol GC. [ruby-trunk Feature #9634]
See this ticket about Symbol GC.
- include/ruby/ruby.h:
Declare few functions.
 - rb_sym2id: almost same as old SYM2ID but support dynamic symbols.
 - rb_id2sym: almost same as old ID2SYM but support dynamic symbols.
 - rb_sym2str: almost same as rb_id2str(SYM2ID(sym)) but not pin down a dynamic symbol. Declare a new struct.
 - struct RSymbol: represents a dynamic symbol as object in Ruby's heaps. Add few macros.
 - STATIC_SYM_P: check a static symbol.
 - DYNAMIC_SYM_P: check a dynamic symbol.
 - RSYMBOL: cast to RSymbol
- gc.c: declare RSymbol. support T_SYMBOL.
- internal.h: Declare few functions.
 - rb_gc_free_dsymbols: free up a dynamic symbol. GC call this function at a sweep phase.
 - rb_str_dynamic_intern: convert a string to a dynamic symbol.
 - rb_check_id_without_pindown: not pinning function.
 - rb_sym2id_without_pindown: ditto.
 - rb_check_id_cstr_without_pindown: ditto.
- string.c (Init_String): String#intern and String#to_sym use rb_str_dynamic_intern.
- template/id.h.templ: use LSB of ID as a flag for determining a static symbol, so we shift left other ruby_id_types.
- string.c: use rb_sym2str instead rb_id2str(SYM2ID(sym)) to avoid pinning.
- load.c: use xx_without_pindown function at creating temporary ID to avoid pinning.
- object.c: ditto.
- sprintf.c: ditto.
- struct.c: ditto.
- thread.c: ditto.
- variable.c: ditto.
- vm_method.c: ditto.

History

#1 - 03/13/2014 09:41 AM - rosenfeld (Rodrigo Rosenfeld Rosas)

Wow, great work! Congrats :-)

#2 - 03/13/2014 02:20 PM - ktsj (Kazuki Tsujimoto)

- File test-all_segfault.log added

make test-all sometimes causes segmentation fault.
I attached the backtrace log.

#3 - 03/14/2014 05:02 AM - authorNari (Narihiro Nakamura)

- Description updated

#4 - 03/14/2014 05:08 AM - authorNari (Narihiro Nakamura)

Kazuki Tsujimoto wrote:

make test-all sometimes causes segmentation fault.
I attached the backtrace log.

Thank you! I fixed it and rebased.

<https://github.com/authorNari/ruby/commit/9cd060aab6ca9cf55971b8d8881b30f0204f71be>

https://github.com/authorNari/ruby/compare/4a91fb7a45f0e3c...symbol_gc

#5 - 03/14/2014 06:38 AM - normalperson (Eric Wong)

Cool! I benchmarked your original version and it didn't notice obvious regressions.

I noticed `rb_check_id_without_pindown` still takes a volatile arg. Is this for GC-safety? Can we encourage `RB_GC_GUARD` instead for new APIs? volatile is not always enough, and tends to generate bad code. I realize this was probably for consistency with the old `rb_check_id` function.

#6 - 03/14/2014 10:07 AM - ktsj (Kazuki Tsujimoto)

Narihiro Nakamura wrote:

Thank you! I fixed it and rebased.

<https://github.com/authorNari/ruby/commit/9cd060aab6ca9cf55971b8d8881b30f0204f71be>

https://github.com/authorNari/ruby/compare/4a91fb7a45f0e3c...symbol_gc

New `symbol_gc` branch works fine. Thanks!

#7 - 03/15/2014 04:21 AM - authorNari (Narihiro Nakamura)

Eric Wong wrote:

volatile is not always enough, and tends to generate bad code.

It make sense for me.

I've removed the volatile declaration of `rb_check_id_without_pindown`.

<https://github.com/authorNari/ruby/commit/5d5f9a63cc059433aa304a4af5>

#8 - 03/26/2014 04:57 AM - Anonymous

- Status changed from Open to Closed

- % Done changed from 0 to 100

Applied in changeset r45426.

-
- `parse.y`: support Symbol GC. [ruby-trunk Feature [#9634](#)]
See this ticket about Symbol GC.
 - `include/ruby/ruby.h`:
Declare few functions.
 - `rb_sym2id`: almost same as old `SYM2ID` but support dynamic symbols.
 - `rb_id2sym`: almost same as old `ID2SYM` but support dynamic symbols.
 - `rb_sym2str`: almost same as `rb_id2str(SYM2ID(sym))` but not pin down a dynamic symbol. Declare a new struct.
 - struct `RSymbol`: represents a dynamic symbol as object in Ruby's heaps. Add few macros.
 - `STATIC_SYM_P`: check a static symbol.
 - `DYNAMIC_SYM_P`: check a dynamic symbol.
 - `RSYMBOL`: cast to `RSymbol`
 - `gc.c`: declare `RSymbol`. support `T_SYMBOL`.
 - `internal.h`: Declare few functions.

- rb_gc_free_dsymbol: free up a dynamic symbol. GC call this function at a sweep phase.
 - rb_str_dynamic_intern: convert a string to a dynamic symbol.
 - rb_check_id_without_pindown: not pinning function.
 - rb_sym2id_without_pindown: ditto.
 - rb_check_id_cstr_without_pindown: ditto.
- string.c (Init_String): String#intern and String#to_sym use rb_str_dynamic_intern.
 - template/id.h.tmpl: use LSB of ID as a flag for determining a static symbol, so we shift left other ruby_id_types.
 - string.c: use rb_sym2str instead rb_id2str(SYM2ID(sym)) to avoid pinning.
 - load.c: use xx_without_pindown function at creating temporary ID to avoid pinning.
 - object.c: ditto.
 - sprintf.c: ditto.
 - struct.c: ditto.
 - thread.c: ditto.
 - variable.c: ditto.
 - vm_method.c: ditto.

Files

test-all_segfault.log	10 KB	03/13/2014	ktsj (Kazuki Tsujimoto)
-----------------------	-------	------------	-------------------------