

## Ruby master - Feature #3222

### Can bignums have singleton class & methods?

04/30/2010 07:07 AM - marcandre (Marc-Andre Lafortune)

<b>Status:</b>	Closed	
<b>Priority:</b>	Normal	
<b>Assignee:</b>	ko1 (Koichi Sasada)	
<b>Target version:</b>	2.0.0	
<b>Description</b>		
<pre>=begin Fixing up the rubyspecs led me to the following:  bn = 1 &lt;&lt; 100 class &lt;&lt; bn def foo 42 end end # =&gt; TypeError: can't define singleton method "foo" for Bignum  bn.define_singleton_method(:foo){42} # =&gt; TypeError: can't define singleton method "foo" for Bignum  On the other hand...  module Bar def foo 42 end end class &lt;&lt; bn include Bar end bn.foo # =&gt; 42  If Ruby won't allow singleton methods for Bignum, then shouldn't it disallow access to the singleton class completely?  See also issue <a href="#">#601</a> =end</pre>		
<b>Related issues:</b>		
Related to Ruby master - Bug #601: an instance of Bignum can have singleton m...	<b>Closed</b>	<b>09/25/2008</b>
Related to Ruby master - Feature #6936: Forbid singleton class and instance v...	<b>Closed</b>	<b>08/27/2012</b>

#### Associated revisions

##### Revision f3e5f2cd - 10/28/2012 12:03 AM - ko1 (Koichi Sasada)

- bignum.c (bignew\_1): Bignum instances are frozen. Feature #3222
- include/ruby/ruby.h: Fixnum instances are also frozen.
- class.c (singleton\_class\_of): check Bignum before singleton cheking.
- test/ruby/test\_bignum.rb: add a test.
- test/ruby/test\_fixnum.rb: ditto.
- test/ruby/marshaltestlib.rb, test/ruby/test\_eval.rb, test/ruby/test\_object.rb: catch up above changes.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@37348 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

##### Revision 37348 - 10/28/2012 12:03 AM - ko1 (Koichi Sasada)

- bignum.c (bignew\_1): Bignum instances are frozen. Feature #3222
- include/ruby/ruby.h: Fixnum instances are also frozen.
- class.c (singleton\_class\_of): check Bignum before singleton cheking.
- test/ruby/test\_bignum.rb: add a test.

- test/ruby/test\_fixnum.rb: ditto.
- test/ruby/marshaltestlib.rb, test/ruby/test\_eval.rb, test/ruby/test\_object.rb: catch up above changes.

#### Revision 37348 - 10/28/2012 12:03 AM - ko1 (Koichi Sasada)

- bignum.c (bignew\_1): Bignum instances are frozen. Feature #3222
- include/ruby/ruby.h: Fixnum instances are also frozen.
- class.c (singleton\_class\_of): check Bignum before singleton cheking.
- test/ruby/test\_bignum.rb: add a test.
- test/ruby/test\_fixnum.rb: ditto.
- test/ruby/marshaltestlib.rb, test/ruby/test\_eval.rb, test/ruby/test\_object.rb: catch up above changes.

#### Revision 37348 - 10/28/2012 12:03 AM - ko1 (Koichi Sasada)

- bignum.c (bignew\_1): Bignum instances are frozen. Feature #3222
- include/ruby/ruby.h: Fixnum instances are also frozen.
- class.c (singleton\_class\_of): check Bignum before singleton cheking.
- test/ruby/test\_bignum.rb: add a test.
- test/ruby/test\_fixnum.rb: ditto.
- test/ruby/marshaltestlib.rb, test/ruby/test\_eval.rb, test/ruby/test\_object.rb: catch up above changes.

#### Revision 37348 - 10/28/2012 12:03 AM - ko1 (Koichi Sasada)

- bignum.c (bignew\_1): Bignum instances are frozen. Feature #3222
- include/ruby/ruby.h: Fixnum instances are also frozen.
- class.c (singleton\_class\_of): check Bignum before singleton cheking.
- test/ruby/test\_bignum.rb: add a test.
- test/ruby/test\_fixnum.rb: ditto.
- test/ruby/marshaltestlib.rb, test/ruby/test\_eval.rb, test/ruby/test\_object.rb: catch up above changes.

#### Revision 37348 - 10/28/2012 12:03 AM - ko1 (Koichi Sasada)

- bignum.c (bignew\_1): Bignum instances are frozen. Feature #3222
- include/ruby/ruby.h: Fixnum instances are also frozen.
- class.c (singleton\_class\_of): check Bignum before singleton cheking.
- test/ruby/test\_bignum.rb: add a test.
- test/ruby/test\_fixnum.rb: ditto.
- test/ruby/marshaltestlib.rb, test/ruby/test\_eval.rb, test/ruby/test\_object.rb: catch up above changes.

#### Revision 37348 - 10/28/2012 12:03 AM - ko1 (Koichi Sasada)

- bignum.c (bignew\_1): Bignum instances are frozen. Feature #3222
- include/ruby/ruby.h: Fixnum instances are also frozen.
- class.c (singleton\_class\_of): check Bignum before singleton cheking.
- test/ruby/test\_bignum.rb: add a test.
- test/ruby/test\_fixnum.rb: ditto.
- test/ruby/marshaltestlib.rb, test/ruby/test\_eval.rb, test/ruby/test\_object.rb: catch up above changes.

## History

---

### #1 - 04/30/2010 02:20 PM - shyouhei (Shyouhei Urabe)

```
=begin
Seems like a bug to me.
=end
```

### #2 - 05/03/2010 05:11 PM - mame (Yusuke Endoh)

- Assignee set to matz (Yukihiko Matsumoto)
- Target version changed from 1.9.2 to 2.0.0

```
=begin
Hi,
```

I think this is not a bug.

It is still possible to define method to Bignum by redefining singleton\_method\_added first:

```
x = (1 << 64)
```

```
def x.singleton_method_added(x)
end
def x.foo
p 1
end
x.foo #=> 1
```

Matz actually fixed [#601](#), but he was not keen to fix this kind of issue. [ruby-dev:36778]

In fact, the above loophole is indicated by matz himself.

I agree that Ruby should disallow access to the singleton class completely, but I don't know if it is easy or not.

Anyway, we should discuss this matter towards 1.9.3 or later.

I move this ticket to Feature tracker.

--

Yusuke Endoh [mame@tsg.ne.jp](mailto:mame@tsg.ne.jp)

=end

### **#3 - 09/14/2010 04:00 PM - shyouhei (Shyouhei Urabe)**

- Status changed from Open to Assigned

=begin

=end

### **#4 - 10/27/2012 05:42 AM - ko1 (Koichi Sasada)**

- Description updated

How about "Freeze" all of Bignum instance?

It is big change, but I believe no impact on it.

Related ticket:

[ruby-dev:46081] [ruby-trunk - Feature [#6936](#)][Assigned] Forbid singleton class and instance variables for float

(how to add related ticket into redmine system?)

### **#5 - 10/27/2012 10:20 PM - Eregon (Benoit Daloze)**

ko1 (Koichi Sasada) wrote:

How about "Freeze" all of Bignum instance?

It is big change, but I believe no impact on it.

It makes sense to me to have them frozen, but I think we would need to freeze all Numeric instances as well for consistency (currently, Fixnum, Rational and Complex can have ivars).

### **#6 - 10/27/2012 10:29 PM - ko1 (Koichi Sasada)**

(2012/10/27 22:20), Eregon (Benoit Daloze) wrote:

It makes sense to me to have them frozen, but I think we would need to freeze all Numeric instances as well for consistency (currently, Fixnum, Rational and Complex can have ivars).

Float is frozen because of introduction of Flonum technique.

I think Fixnum should be also frozen.

I'm not sure about complex and rational. I don't think inconsistent because we can make your own Numeric classes (and we can't force it frozen). But I have no objection.

I want to ask real users of Numeric classes.

--

// SASADA Koichi at atdot dot net

**#7 - 10/27/2012 11:25 PM - matz (Yukihiro Matsumoto)**

- Assignee changed from matz (Yukihiro Matsumoto) to ko1 (Koichi Sasada)

Accepted.

Matz.

**#8 - 10/28/2012 05:53 AM - ko1 (Koichi Sasada)**

(2012/10/27 23:25), matz (Yukihiro Matsumoto) wrote:

Issue [#3222](#) has been updated by matz (Yukihiro Matsumoto).

Assignee changed from matz (Yukihiro Matsumoto) to ko1 (Koichi Sasada)

Accepted.

Should I commit Fixnum and Bignum freezing patch (\*1)?

\*1: <http://www.atdot.net/sp/view/4qjkcm/readonly>

I got several errors on test-all caused by Fixnum freezing (no effect on Bignum freezing at test-all). I want to make clear that Fixnum can be frozen or not (Ruby programmers expect Fixnum is mutable?).

```
[ 3393/11368] TestEval#test_instance_eval_block_basic = 0.00 s
1) Error:
test_instance_eval_block_basic(TestEval):
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:133:in instance_variable_set'
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:133:inblock in forall_TYPE'
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:132:in each'
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:132:inforall_TYPE'
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:191:in `test_instance_eval_block_basic'
```

```
[ 3398/11368] TestEval#test_instance_eval_string_basic = 0.00 s
2) Error:
test_instance_eval_string_basic(TestEval):
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:133:in instance_variable_set'
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:133:inblock in forall_TYPE'
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:132:in each'
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:132:inforall_TYPE'
/mnt/sdb1/ruby/trunk/test/ruby/test_eval.rb:163:in `test_instance_eval_string_basic'
```

```
[ 6110/11368] TestMarshal#test_fixnum_ivar = 0.00 s
3) Error:
test_fixnum_ivar(TestMarshal):
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:186:in remove_instance_variable'
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:186:inblock in test_fixnum_ivar'
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:186:in instance_eval'
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:186:inensure in test_fixnum_ivar'
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:186:in `test_fixnum_ivar'
```

```
[ 6111/11368] TestMarshal#test_fixnum_ivar_self = 0.00 s
4) Error:
test_fixnum_ivar_self(TestMarshal):
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:194:in remove_instance_variable'
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:194:inblock in test_fixnum_ivar_self'
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:194:in instance_eval'
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:194:inensure in test_fixnum_ivar_self'
/mnt/sdb1/ruby/trunk/test/ruby/marshaltestlib.rb:194:in `test_fixnum_ivar_self'
```

```
[ 6695/11368] TestObject#test_freeze_immediate = 0.00 s
5) Failure:
test_freeze_immediate(TestObject) [/mnt/sdb1/ruby/trunk/test/ruby/test_object.rb:65]:
expected but was
.
```

The following log is from rubyspec.

```

1)
A singleton class has class Bignum as the superclass of a Bignum instance ERROR
TypeError: can't define singleton
/mnt/sdb1/ruby/trunk/spec/rubyspec/language singleton_class_spec.rb:79:in singleton_class'
/mnt/sdb1/ruby/trunk/spec/rubyspec/language singleton_class_spec.rb:79:inblock (3 levels) in '
/mnt/sdb1/ruby/trunk/spec/rubyspec/language singleton_class_spec.rb:4:in ``

2)
Kernel#instance_variables immediate values returns the correct array if an instance variable is added ERROR
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/instance_variables_spec.rb:23:in instance_variable_set'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/instance_variables_spec.rb:23:inblock (4 levels) in '
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/instance_variables_spec.rb:4:in ``

3)
Kernel#taint has no effect on immediate values ERROR
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/taint_spec.rb:37:in taint'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/taint_spec.rb:37:inblock (3 levels) in '
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/taint_spec.rb:36:in each'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/taint_spec.rb:36:inblock (2 levels) in '
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/taint_spec.rb:4:in ``

4)
Kernel#untrusted? has no effect on immediate values ERROR
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/untrusted_spec.rb:21:in untrust'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/untrusted_spec.rb:21:inblock (3 levels) in '
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/kernel/untrusted_spec.rb:4:in ``

5)
Module#attr_accessor allows creating an attr_accessor on an immediate class ERROR
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/attr_accessor_spec.rb:36:in block (2 levels) in <top (required)>'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/attr_accessor_spec.rb:4:in'

6)
Module#attr_reader allows for adding an attr_reader to an immediate ERROR
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/attr_reader_spec.rb:32:in instance_variable_set'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/attr_reader_spec.rb:32:inblock (2 levels) in '
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/attr_reader_spec.rb:4:in ``

7)
Module#attr_writer allows for adding an attr_writer to an immediate ERROR
RuntimeError: can't modify frozen Fixnum
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/attr_writer_spec.rb:32:in block (2 levels) in <top (required)>'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/module/attr_writer_spec.rb:4:in'

8)
String#% taints result for %s when argument is tainted ERROR
RuntimeError: can't modify frozen Float
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/string/modulo_spec.rb:654:in taint'
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/string/modulo_spec.rb:654:inblock (2 levels) in '
/mnt/sdb1/ruby/trunk/spec/rubyspec/core/string/modulo_spec.rb:4:in ``

```

Thanks,  
Koichi

--  
// SASADA Koichi at atdot dot net

**#9 - 10/28/2012 06:45 AM - Eregon (Benoit Daloze)**

ko1 (Koichi Sasada) wrote:

(2012/10/27 22:20), Eregon (Benoit Daloze) wrote:

It makes sense to me to have them frozen, but I think we would need to freeze all Numeric instances as well for consistency (currently, Fixnum, Rational and Complex can have ivars).

Float is frozen because of introduction of Flonum technique.  
I think Fixnum should be also frozen.

I'm not sure about complex and rational. I don't think inconsistent because we can make your own Numeric classes (and we can't force it frozen). But I have no objection.

I think it's to custom Numeric subclasses authors to chose for them to be frozen or not, but I think having all core Numeric subclasses instances frozen would make sense. These are anyway already immutable for their functionality, it seems weird to allow some kind of changes like singleton methods and ivars.

But you're right, as Rational and Complex are composed of other types, it seems less important for them to follow Fixnum/Bignum/Float (and BigDecimal?).

However, freezing Fixnum would likely affect more code (notably because their instances are supposed to be unique in MRI).

So theoretically numeric primitives should always be totally immutable, but on the other hand I like having them as full-fledged objects (that is I dislike having an object/type distinction) and freezing them reduces freedom.

```
class Fixnum; def fib; @fib ||= (self-1).fib + (self-2).fib; end; end; [0,1].each { |i| i.instance_eval { @fib = i } }
=> [0, 1]
42.fib
=> 267914296
```

This is fun but evil ...

#### #10 - 10/28/2012 06:59 AM - ko1 (Koichi Sasada)

(2012/10/28 6:45), Eregon (Benoit Daloze) wrote:

```
class Fixnum; def fib; @fib ||= (self-1).fib + (self-2).fib; end; end; [0,1].each { |i| i.instance_eval { @fib = i } }
=> [0, 1]
42.fib
=> 267914296
```

This is fun but evil ...

I completely agree with that. I've forgot such pleasure!!

--  
// SASADA Koichi at atdot dot net

#### #11 - 10/28/2012 08:23 AM - Anonymous

Hi,

In message "Re: [ruby-core:48497] Re: [ruby-trunk - Feature #3222] Can bignums have singleton class & methods?" on Sun, 28 Oct 2012 05:33:45 +0900, SASADA Koichi [ko1@atdot.net](mailto:ko1@atdot.net) writes:

|> Accepted.

| Should I commit Fixnum and Bignum freezing patch (\*1)?

| \*1: <http://www.atdot.net/sp/view/4qjkcmm/readonly>

| I got several errors on test-all caused by Fixnum freezing (no effect on Bignum freezing at test-all). I want to make clear that Fixnum can be frozen or not (Ruby programmers expect Fixnum is mutable?).

Ah, some programs might expect modifying instance variables of fixnums. But I think you can ignore such programs for most of the case. Try it.

matz.

#### #12 - 10/28/2012 08:23 AM - ko1 (Koichi Sasada)

(2012/10/28 8:13), Yukihiro Matsumoto wrote:

Ah, some programs might expect modifying instance variables of fixnums. But I think you can ignore such programs for most of the case. Try it.

Roger, boss.

--

// SASADA Koichi at atdot dot net

**#13 - 10/28/2012 09:03 AM - ko1 (Koichi Sasada)**

- *Status changed from Assigned to Closed*

- *% Done changed from 0 to 100*

This issue was solved with changeset r37348.

Marc-Andre, thank you for reporting this issue.

Your contribution to Ruby is greatly appreciated.

May Ruby be with you.

- 
- `bignum.c` (`bignew_1`): Bignum instances are frozen. Feature [#3222](#)
  - `include/ruby/ruby.h`: Fixnum instances are also frozen.
  - `class.c` (`singleton_class_of`): check Bignum before singleton cheking.
  - `test/ruby/test_bignum.rb`: add a test.
  - `test/ruby/test_fixnum.rb`: ditto.
  - `test/ruby/marshaltestlib.rb`, `test/ruby/test_eval.rb`, `test/ruby/test_object.rb`: catch up above changes.