Ruby - Bug #21324

Namespace loads RubyGems in root Namespace but it should not

05/10/2025 12:36 PM - Eregon (Benoit Daloze)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 3.5.0dev (2025-05-10T07:50:29Z namespace-on-read-.. bd4f57f96b) +PRISM [x86_64-linux]	**Backport:**	3.2: UNKNOWN, 3.3: UNKNOWN, 3.4: UNKNOWN

Description

RubyGems has tons of mutable state, isn't core library and isn't "builtin classes" either, so it should not be in root Namespace, but it is currently:

```
$ RUBY_NAMESPACE=1 ruby -ve 'ns = Namespace.new; p ns::Gem.equal?(Gem)'
ruby 3.5.0dev (2025-05-10T07:50:29Z namespace-on-read-.. bd4f57f96b) +PRISM [x86_64-linux]
ruby: warning: Namespace is experimental, and the behavior may change in the future!
See doc/namespace.md for know issues, etc.
true
```

A concrete example of what breaks most likely due to this:

```
$ gem i delegate:0.3.1
$ RUBY_NAMESPACE=1 ruby -ve 'require "delegate"; p Delegator::VERSION; ns = Namespace.new; File.wr
ite "ns.rb", "gem %{delegate}, %{0.3.1}; require :delegate.to_s; p Delegator::VERSION"; ns.require
 "./ns"'
ruby 3.5.0dev (2025-05-10T07:50:29Z namespace-on-read-.. bd4f57f96b) +PRISM [x86_64-linux]
ruby: warning: Namespace is experimental, and the behavior may change in the future!
See doc/namespace.md for know issues, etc.
"0.4.0"
/home/eregon/prefix/ruby-master/lib/ruby/3.5.0+0/rubygems/specification.rb:2232:in 'Gem::Specifica
tion#check_version_conflict': can't activate delegate-0.3.1, already activated delegate-0.4.0 (Gem
::LoadError)
 from /home/eregon/prefix/ruby-master/lib/ruby/3.5.0+0/rubygems/specification.rb:1383:in 'Gem::Spe
cification#activate'
 from /home/eregon/prefix/ruby-master/lib/ruby/3.5.0+0/rubygems/core_ext/kernel_gem.rb:62:in 'bloc
k in Kernel#gem'
 from /home/eregon/prefix/ruby-master/lib/ruby/3.5.0+0/rubygems/core_ext/kernel_gem.rb:62:in 'Thre
ad::Mutex#synchronize'
 from /home/eregon/prefix/ruby-master/lib/ruby/3.5.0+0/rubygems/core_ext/kernel_gem.rb:62:in 'Kern
el#gem'
 from /home/eregon/ns.rb:1:in '<top (required)>'
 from -e:1:in 'Namespace#require'
 from -e:1:in '<main>'
```

i.e. ns sees delegate-0.4.0 was loaded in main namespace but it shouldn't.

Previously mentioned in https://bugs.ruby-lang.org/issues/21311#note-36

Related issues:

Related to Ruby - Bug #21323: irb fails to start with Namespace	**Closed**
Related to Ruby - Bug #21339: Namespace: `RubyVM::InstructionSequence.load_is...	**Assigned**

History

#1 - 05/10/2025 12:36 PM - Eregon (Benoit Daloze)

- Related to Bug #21323: irb fails to start with Namespace added

#2 - 05/10/2025 12:38 PM - Eregon (Benoit Daloze)

- Description updated

#3 - 05/11/2025 01:13 AM - mame (Yusuke Endoh)

- Assignee set to prism

#4 - 05/11/2025 03:44 AM - mame (Yusuke Endoh)

- Assignee deleted (prism)

#5 - 05/13/2025 06:47 AM - jas (Jasveen Sandral)

I've identified the issue and prepared a fix.

The problem is that RubyGems is being loaded into the root namespace during Ruby's boot process, and then all user namespaces inherit this same Gem constant through the Copy-on-Write mechanism. This causes all namespaces to share the same RubyGems environment and state, leading to version conflicts.

The fix involves:

1. In rb_initialize_main_namespace():
    - Detecting if the Gem constant exists in Object
    - Removing it explicitly with rb_const_remove()
    - Loading a fresh RubyGems environment
2. In namespace_initialize() (for Namespace.new):
    - Getting the namespace's own Object constant
    - Checking and removing any inherited Gem constant
    - Loading a clean RubyGems environment

This ensures each namespace (main and optional) gets its own isolated RubyGems state, preventing version conflicts when loading gems across namespaces.

I've created a PR with these changes: https://github.com/ruby/ruby/pull/13313

#6 - 05/13/2025 11:40 AM - Eregon (Benoit Daloze)

Loading RubyGems creates more constants than just Gem:

```
$ ruby -v --disable-gems -e 'a=Object.constants; require "rubygems"; b=Object.constants; p b-a'
ruby 3.4.1 (2024-12-25 revision 48d4efcb85) +PRISM [x86_64-linux]
[:Monitor, :MonitorMixin, :Gem, :CROSS_COMPILING, :RbConfig, :RUBYGEMS_ACTIVATION_MONITOR]
```

It seems very messy to try to remove that from the root namespace.

The clean solution seems obvious: gem_prelude.rb shouldn't be loaded in the root namespace, but only in user namespaces.

#7 - 05/15/2025 10:48 AM - Eregon (Benoit Daloze)

- Related to Bug #21339: Namespace: `RubyVM::InstructionSequence.load_iseq` isn't called for the root namespace added

#8 - 06/03/2025 05:47 AM - hsbt (Hiroshi SHIBATA)

- Tags set to namespace