Ruby - Bug #21032

`Module#autoload?` is slow when `$LOAD_PATH` contains a relative path

01/13/2025 05:35 PM - byroot (Jean Boussier)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:		**Backport:**	3.1: WONTFIX, 3.2: DONE, 3.3: DONE, 3.4: DONE

Description

Reproduction script:

```
require 'benchmark'

$LOAD_PATH << 'relative-path'

autoload :FOO, '/tmp/foo.rb'

puts Benchmark.realtime {
  500_000.times do
    Object.autoload?(:FOO)
  end
}
```

The above takes 2.5 to 3 seconds on my machine, but just removing $LOAD_PATH << 'relative-path' make it complete in 50ms.
It's such a stark difference that I think it is a bug, and it cause Zeitwerk, a very popular gem, to be way slower than it should when the load path contains relative paths.

I have a patch for it, that passes all tests, but I'd appreciate some eyes on it: https://github.com/ruby/ruby/pull/12562

cc @fxn

Associated revisions

Revision ead3bbc2405ad1df2228c44133ee1c6574ef5973 - 02/14/2025 05:09 AM - k0kubun (Takashi Kokubun)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
    rb_feature_p: skip `get_expanded_load_path` for absolute paths

    Ref: https://github.com/fxn/zeitwerk/pull/308

    ```ruby
    require 'benchmark'

    $LOAD_PATH << 'relative-path'

    autoload :FOO, '/tmp/foo.rb'

    puts Benchmark.realtime {
      500_000.times do
        Object.autoload?(:FOO)
      end
    }
    ```

    The above script takes 2.5 seconds on `master`, and only
    50ms on this branch.

    When we're looking for a feature with an absolute path, we don't
    need to call the expensive `get_expanded_load_path`.
```

Revision ead3bbc2405ad1df2228c44133ee1c6574ef5973 - 02/14/2025 05:09 AM - k0kubun (Takashi Kokubun)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
    rb_feature_p: skip `get_expanded_load_path` for absolute paths

    Ref: https://github.com/fxn/zeitwerk/pull/308

    ```ruby
    require 'benchmark'

    $LOAD_PATH << 'relative-path'

    autoload :FOO, '/tmp/foo.rb'

    puts Benchmark.realtime {
      500_000.times do
        Object.autoload?(:FOO)
      end
    }
    ```

    The above script takes 2.5 seconds on `master`, and only
    50ms on this branch.

    When we're looking for a feature with an absolute path, we don't
    need to call the expensive `get_expanded_load_path`.
```

Revision ead3bbc2 - 02/14/2025 05:09 AM - k0kubun (Takashi Kokubun)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
    rb_feature_p: skip `get_expanded_load_path` for absolute paths

    Ref: https://github.com/fxn/zeitwerk/pull/308

    ```ruby
    require 'benchmark'

    $LOAD_PATH << 'relative-path'

    autoload :FOO, '/tmp/foo.rb'

    puts Benchmark.realtime {
      500_000.times do
        Object.autoload?(:FOO)
      end
    }
    ```

    The above script takes 2.5 seconds on `master`, and only
    50ms on this branch.

    When we're looking for a feature with an absolute path, we don't
    need to call the expensive `get_expanded_load_path`.
```

Revision 54dd27d89d2e6814114f1aff18836a987d5a4ab1 - 03/08/2025 08:26 AM - nagachika (Tomoyuki Chikanaga)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
    rb_feature_p: skip `get_expanded_load_path` for absolute paths

    Ref: https://github.com/fxn/zeitwerk/pull/308

    ```ruby
    require 'benchmark'

    $LOAD_PATH << 'relative-path'

    autoload :FOO, '/tmp/foo.rb'

    puts Benchmark.realtime {
      500_000.times do
        Object.autoload?(:FOO)
      end
    }
    ```

    The above script takes 2.5 seconds on `master`, and only
```

```
    50ms on this branch.

    When we're looking for a feature with an absolute path, we don't
    need to call the expensive `get_expanded_load_path`.
```

Revision 54dd27d89d2e6814114f1aff18836a987d5a4ab1 - 03/08/2025 08:26 AM - nagachika (Tomoyuki Chikanaga)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
    rb_feature_p: skip `get_expanded_load_path` for absolute paths

    Ref: https://github.com/fxn/zeitwerk/pull/308

    ```ruby
    require 'benchmark'

    $LOAD_PATH << 'relative-path'

    autoload :FOO, '/tmp/foo.rb'

    puts Benchmark.realtime {
      500_000.times do
        Object.autoload?(:FOO)
      end
    }
    ```

    The above script takes 2.5 seconds on `master`, and only
    50ms on this branch.

    When we're looking for a feature with an absolute path, we don't
    need to call the expensive `get_expanded_load_path`.
```

Revision 54dd27d8 - 03/08/2025 08:26 AM - nagachika (Tomoyuki Chikanaga)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
    rb_feature_p: skip `get_expanded_load_path` for absolute paths

    Ref: https://github.com/fxn/zeitwerk/pull/308

    ```ruby
    require 'benchmark'

    $LOAD_PATH << 'relative-path'

    autoload :FOO, '/tmp/foo.rb'

    puts Benchmark.realtime {
      500_000.times do
        Object.autoload?(:FOO)
      end
    }
    ```

    The above script takes 2.5 seconds on `master`, and only
    50ms on this branch.

    When we're looking for a feature with an absolute path, we don't
    need to call the expensive `get_expanded_load_path`.
```

Revision 1c9af6c10f97c0b612492fa63842c78ea2b8da56 - 03/13/2025 05:27 AM - hsbt (Hiroshi SHIBATA)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
    rb_feature_p: skip `get_expanded_load_path` for absolute paths

    Ref: https://github.com/fxn/zeitwerk/pull/308

    ```ruby
    require 'benchmark'

    $LOAD_PATH << 'relative-path'

    autoload :FOO, '/tmp/foo.rb'
```
```

```
puts Benchmark.realtime {
  500_000.times do
    Object.autoload?(:FOO)
  end
}
```

```
The above script takes 2.5 seconds on `master`, and only
50ms on this branch.

When we're looking for a feature with an absolute path, we don't
need to call the expensive `get_expanded_load_path`.
```

Revision 1c9af6c10f97c0b612492fa63842c78ea2b8da56 - 03/13/2025 05:27 AM - hsbt (Hiroshi SHIBATA)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
rb_feature_p: skip `get_expanded_load_path` for absolute paths

Ref: https://github.com/fxn/zeitwerk/pull/308

```ruby
require 'benchmark'

$LOAD_PATH << 'relative-path'

autoload :FOO, '/tmp/foo.rb'

puts Benchmark.realtime {
  500_000.times do
    Object.autoload?(:FOO)
  end
}
```

The above script takes 2.5 seconds on `master`, and only
50ms on this branch.

When we're looking for a feature with an absolute path, we don't
need to call the expensive `get_expanded_load_path`.
```

Revision 1c9af6c1 - 03/13/2025 05:27 AM - hsbt (Hiroshi SHIBATA)

merge revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47: [Backport #21032]

```
rb_feature_p: skip `get_expanded_load_path` for absolute paths

Ref: https://github.com/fxn/zeitwerk/pull/308

```ruby
require 'benchmark'

$LOAD_PATH << 'relative-path'

autoload :FOO, '/tmp/foo.rb'

puts Benchmark.realtime {
  500_000.times do
    Object.autoload?(:FOO)
  end
}
```

The above script takes 2.5 seconds on `master`, and only
50ms on this branch.

When we're looking for a feature with an absolute path, we don't
need to call the expensive `get_expanded_load_path`.
```

History

#1 - 01/13/2025 07:47 PM - Eregon (Benoit Daloze)

For curiosity, what's adding a relative path in $LOAD_PATH? That sounds rather dubious/wrong.

#2 - 01/13/2025 07:49 PM - byroot (Jean Boussier)

@fxn tracked it down to https://github.com/emailage/Emailage_Ruby/blob/64b9762cda7608ac1eeced2a85ad5f4b7919f4f0/lib/emailage.rb#L1, and yes that's an anti-pattern in my opinion too.

#3 - 01/14/2025 08:25 AM - byroot (Jean Boussier)

I dug more into this today, based on @nobu's review. autoload? isn't the only thing slowed down in such case.

Perhaps we should try to emit a performance warning when a relative path or non-string object is appended to $LOAD_PATH. That would at least make it easier to notice this issue.

The problem of course is that $LOAD_PATH is just a regular array, so there isn't a clean hook where to check for this.

#4 - 01/14/2025 01:15 PM - Eregon (Benoit Daloze)

byroot (Jean Boussier) wrote in #note-2:

> @fxn tracked it down to https://github.com/emailage/Emailage_Ruby/blob/64b9762cda7608ac1eeced2a85ad5f4b7919f4f0/lib/emailage.rb#L1, and yes that's an anti-pattern in my opinion too.

Right, that's just broken code. It would consider files under lib in $PWD for require which is wrong.

> Perhaps we should try to emit a performance warning when a relative path or non-string object is appended to $LOAD_PATH. That would at least make it easier to notice this issue.
>
> The problem of course is that $LOAD_PATH is just a regular array, so there isn't a clean hook where to check for this.

Maybe just warn on require or so, i.e. when noticing the path is relative, if it's too hard to notice when it's added?
I think there is also a cache for the expanded load path, maybe that could be a reasonable place to check.
It wouldn't immediately point at the culprit, but printing the $LOADED_FEATURES should help find it (maybe there should be a CLI flag/ENV var to print files as they are loaded, TruffleRuby has that).

#5 - 01/27/2025 09:34 AM - byroot (Jean Boussier)

- Status changed from Open to Closed

I merged https://github.com/ruby/ruby/pull/12562 / d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47 because I couldn't find any case where this would change behavior.

#6 - 02/14/2025 05:09 AM - k0kubun (Takashi Kokubun)

- Backport changed from 3.1: WONTFIX, 3.2: REQUIRED, 3.3: REQUIRED, 3.4: REQUIRED to 3.1: WONTFIX, 3.2: REQUIRED, 3.3: REQUIRED, 3.4: DONE

ruby_3_4 ead3bbc2405ad1df2228c44133ee1c6574ef5973 merged revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47.

#7 - 03/08/2025 08:40 AM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 3.1: WONTFIX, 3.2: REQUIRED, 3.3: REQUIRED, 3.4: DONE to 3.1: WONTFIX, 3.2: REQUIRED, 3.3: DONE, 3.4: DONE

ruby_3_3 54dd27d89d2e6814114f1aff18836a987d5a4ab1 merged revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47.

#8 - 03/13/2025 04:22 AM - hsbt (Hiroshi SHIBATA)

- Backport changed from 3.1: WONTFIX, 3.2: REQUIRED, 3.3: DONE, 3.4: DONE to 3.1: WONTFIX, 3.2: DONE, 3.3: DONE, 3.4: DONE

ruby_3_2 commit:fd15d0f4cbddc8cf69013959e2a60734d0995d56 merged revision(s) d4a1a2780c39bc648496ac92fc6e6ce2eb38ab47.