Ruby - Feature #14565

Simpler, one-liner, failsafe require in ruby? [Suggested names: require_failsafe, require_safe, require_try, require_add)

03/01/2018 05:48 PM - shevegen (Robert A. Heiler)

Status:	Open	
Priority:	Normal	
Assignee:		
Target version:		

Description

I have quite a bit of code like this:

```
begin
  require 'x/tools/cdrskin.rb'
rescue LoadError; end
```

I also use the longer variant, e.g.,

```
begin
  require 'foobar'
rescue LoadError
  puts 'project foobar is not available - consider '\
    'installing it via gem install foobar'
end
```

Often, I do not need to inform the user about missing gems/projects that are tiny and not very important. In my larger ruby projects, I handle cases where a smaller project is not available or available, so I can proceed either way. It is a bit pointless to notify the user when that is me; that is why I would like to have a one-liner.

I am thinking of an API such as any of the following:

```
require_failsafe
require_safe
require_try
require_add
```

This is for loading with a rescue LoadError without notification. If I need to notify a user then I am fine with the longer variant.

If anyone has better names, feel free to add them! I think people are more likely to remember the require-family, e. g. require 'foo.rb' or require_relative 'bar.rb' and so forth.

I also wanted to propose a stronger require/import, including the possibility to refer to .rb files without a hardcoded path (if the .rb file is moved,
all explicit requires to it, in particular from external projects, would have to change; and my vague idea is to replace this with some kind of project-specific way to
"label" files and load these files based on these "labels", but that is for another suggestion; I only want to mention it because Hiroshi Shibata made some suggestion as extension to require, and I think the use case he mentioned may also be useful to see whether ruby may get a stronger "load code in files" functionality for ruby 3.x eventually).

Related issues:

Related to Ruby - Bug #20714: Handle optional dependencies in `bundled_gems.rb`	**Assigned**

History

#1 - 03/01/2018 05:49 PM - shevegen (Robert A. Heiler)

(I have a few typos above; is there a way to edit the first post in the
bug tracker? I only seem to be able to edit in subsequent posts... hmm)

#2 - 03/01/2018 05:56 PM - zverok (Victor Shepelev)

@shevegen If you press "Edit", the form has Description [Edit] field, and you can edit description there.

#3 - 10/09/2019 05:42 AM - sawa (Tsuyoshi Sawada)

- Description updated

#4 - 04/02/2025 06:43 AM - jeromedalbert (Jerome Dalbert)

I like this feature request, a non-failing require would be great for gems that optionally depend on another gem.

For example:

```
begin
  require 'rubocop-rspec'
rescue LoadError
end

if defined?(RuboCop::RSpec)
  # ...
end
```

could be turned into something like this

```
if require('rubocop-rspec', exception: false)
  # ...
end
```

#5 - 04/02/2025 07:27 AM - hsbt (Hiroshi SHIBATA)

- Related to Bug #20714: Handle optional dependencies in `bundled_gems.rb` added

#6 - 04/03/2025 03:11 AM - austin (Austin Ziegler)

jeromedalbert (Jerome Dalbert) wrote in #note-4:

> I like this feature request, a non-failing require would be great for gems that optionally depend on another gem.
>
> For example:
>
> ```
> begin
>   require 'rubocop-rspec'
> rescue LoadError
> end
>
> if defined?(RuboCop::RSpec)
>   # ...
> end
> ```
>
> could be turned into something like this
>
> ```
> if require('rubocop-rspec', exception: false)
>   # ...
> end
> ```

Unfortunately, that if will not work, because:

```
p require('yaml') # true
p require('yaml') # false
```

If LoadError were a descendant of StandardError, then a suffix rescue could work:

```
require 'rubocop-spec' rescue nil
```

I wonder if something could be done with pattern matching here to extend suffix rescues:

```
require 'rubocop-spec' rescue LoadError => nil
```

You would still need to check for defined?(RuboCop::Rspec) because of the return value, but…

I do think that require_try or even require(resource, exception: false) would work nicely without that.

#7 - 04/03/2025 11:38 AM - deivid (David Rodríguez)

I like require "rubocop-rspec", optional: true idea from https://bugs.ruby-lang.org/issues/20714.

I wonder if a nil return value for require ..., optional: true when the feature was not available would make sense, and would enable users to act accordingly?