Redmine - Defect #28133
Controller patch causes Helper Patch down

2018-02-05 05:59 - Haihan Ji

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Plugin API	**Estimated time:**	0.00 hour
Target version:			
Resolution:		**Affected version:**	

Description

Create two plugins, one named 'test_001', another is 'test_002'.

test_001

```
# init.rb
Rails.configuration.to_prepare do
  require 'test_issues_controller_patch'
end

Redmine::Plugin.register :test_001 do
  name 'Test 001 plugin'
  author 'Author name'
  description 'This is a plugin for Redmine'
  version '0.0.1'
  url 'http://example.com/path/to/plugin'
  author_url 'http://example.com/about'
end

# lib/test_issues_controller_patch.rb
module TestIssuesControllerPatch
  extend ActiveSupport::Concern
  included do
  end
end

unless IssuesController.included_modules.include?(TestIssuesControllerPatch)
  IssuesController.send :include, TestIssuesControllerPatch
end
```

test_002

```
# init.rb
Rails.configuration.to_prepare do
  require 'test_hooks'
  require 'test_issues_helper_patch'
end

# Rails.configuration.to_prepare
Redmine::Plugin.register :test_002 do
  name 'Test 002 plugin'
  author 'Author name'
  description 'This is a plugin for Redmine'
  version '0.0.1'
  url 'http://example.com/path/to/plugin'
  author_url 'http://example.com/about'
end

# lib/test_issues_helper_patch.rb
module TestIssuesHelperPatch
  extend ActiveSupport::Concern
  included do
```

```
  end

  def test_hello
    'hello world'
  end
end

unless IssuesHelper.included_modules.include?(TestIssuesHelperPatch)
  IssuesHelper.send :include, TestIssuesHelperPatch
end

# lib/test_hooks.rb
class TestHooks < Redmine::Hook::ViewListener
  def view_issues_show_details_bottom(context={})
    context[:hook_caller].instance_eval do
      render(:partial => 'test/hello_world')
    end
  end
end

# app/views/test/_hello_world.html.rb
<%= test_hello %>
```

If everything is OK, 'hello world' should be displayed in issue detail page.
BUT I see:

```
ActionView::Template::Error (undefined local variable or method `test_hello' for
#<#<Class:0x007fca71f0bd90>:0x007fca71f00760>):
    1: <%= test_hello %>
  plugins/test_002/app/views/test/_hello_world.html.erb:1:in
`_plugins_test_____app_views_test__hello_world_html_erb__3192427753573044533_70253770942940'
  plugins/test_002/lib/test_hooks.rb:4:in `block in view_issues_show_details_bottom'
  plugins/test_002/lib/test_hooks.rb:3:in `instance_eval'
  plugins/test_002/lib/test_hooks.rb:3:in `view_issues_show_details_bottom'
  lib/redmine/hook.rb:61:in `block (2 levels) in call_hook'
  lib/redmine/hook.rb:61:in `each'
  lib/redmine/hook.rb:61:in `block in call_hook'
  lib/redmine/hook.rb:58:in `tap'
  lib/redmine/hook.rb:58:in `call_hook'
  lib/redmine/hook.rb:96:in `call_hook'
  app/views/issues/show.html.erb:70:in `_app_views_issues_show_html_erb__1203365146500478953_70253735647440'
  app/controllers/issues_controller.rb:111:in `block (2 levels) in show'
  app/controllers/issues_controller.rb:104:in `show'
  lib/redmine/sudo_mode.rb:63:in `sudo_mode'
```

If I disable test_001, or rename test_002 to test_000 it's work.
It seems that controller patch make helper patch failed.

History

#1 - 2018-08-28 03:02 - Akiko Takano

Hi, I'm really interested in this topic and I could reproduce this problem in my env.

I'm not sure this workaround is correctly, but I could manage to call extended method 'test_hello' like this;

test_001

```
# test001/init.rb

require 'test_issues_controller_patch'

Redmine::Plugin.register :test_001 do
  name 'Test 001 plugin'
  author 'Author name'
  description 'This is a plugin for Redmine'
  version '0.0.1'
  url 'http://example.com/path/to/plugin'
```

```
  author_url 'http://example.com/about'
end

Rails.configuration.to_prepare do
  unless IssuesController.included_modules.include?(TestIssuesControllerPatch)
    IssuesController.send :include, TestIssuesControllerPatch  # only loaded once at initialized plugin
  end
end

# test001/lib/test_issues_controller_patch.rb

module TestIssuesControllerPatch
  extend ActiveSupport::Concern
  included do
  end
end
```

test_002

```
# test_002/init.rb

require 'test_hooks'
require 'test_issues_helper_patch'

Redmine::Plugin.register :test_002 do
  name 'Test 002 plugin'
  author 'Author name'
  description 'This is a plugin for Redmine'
  version '0.0.1'
  url 'http://example.com/path/to/plugin'
  author_url 'http://example.com/about'
end

Rails.configuration.to_prepare do
  # do nothing
end

# test_002/lib/test_hook.rb

class TestHooks < Redmine::Hook::ViewListener
  def view_issues_show_details_bottom(context={})
    context[:hook_caller].instance_eval do
      render(:partial => 'test/hello_world')
    end
  end
end

# test_002/lib/test_issues_helper_patch.rb

module TestIssuesHelperPatch
  extend ActiveSupport::Concern
  included do
  end

  def test_hello
    'hello world'
  end
end

IssuesHelper.send :prepend, TestIssuesHelperPatch # change from include to prepend

# test_002/app/views/test/_hello_world.html.erb

  <!-- prevent method missing exception -->
  <% if defined?(test_hello)  %>
      <%= test_hello %>
  <% end %>
```

As results, test_hello method is appended at the first of method chain.

```
>> IssuesHelper.instance_methods
[:test_hello, :render_issue_tooltip, :issue_list, :grouped_issue_list,  ..... ]
```

```
>> IssuesHelper.ancestors
[TestIssuesHelperPatch, IssuesHelper, Redmine::Export::PDF::IssuesPdfHelper, ApplicationHelper, Redmine::Helpers::URL, Redmine::Hook::
Helper, Redmine::Themes::Helper, Redmine::SudoMode::Helper, Redmine::Pagination::Helper, GravatarHelper::PublicMethods, Redmine::
I18n, Redmine::WikiFormatting::Macros::Definitions]
```

Since TestIssueController patch is empty, I'm not sure if the controller patch also works fine.
But I hope to this would be any help and I'm looking forward to any comments, corrections, advise from you and redmine experts.

#2 - 2019-07-26 11:49 - Stephan Wiehr

Found this during investigation on a similar issue.
I could solve this by moving the 'offending' ControllerPatch to after_initialize callback.

For the example here that would mean to simply transform

```
Rails.configuration.to_prepare do
  unless IssuesController.included_modules.include?(TestIssuesControllerPatch)
    IssuesController.send :include, TestIssuesControllerPatch  # only loaded once at initialized plugin
  end
end
```

into

```
Rails.configuration.after_initialize do
  unless IssuesController.included_modules.include?(TestIssuesControllerPatch)
    IssuesController.send :include, TestIssuesControllerPatch  # only loaded once at initialized plugin
  end
end
```

#3 - 2019-08-20 03:19 - Akiko Takano

Hi, Stephan!

The version of Redmine that everyone mainly use will soon be 4.x, then 3.x.
Since I always struggle with plug-in combinations, so your information above is helpful.

Stephan Wiehr wrote:

> *Found this during investigation on a similar issue.*
> *I could solve this by moving the 'offending' ControllerPatch to after_initialize callback.*