Hooks Manager - Support #2283

Plugin incomparability problem: Uncaught ReferenceError: get is not defined

22 Feb 2014 19:48 - Christopher Caruk

Status:	Closed	**Start date:**	22 Feb 2014
Priority:	Major	**Due date:**	11 Jun 2014
Assignee:	Andriy Lesyuk	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.0.1		
Redmine version:	2.4.3	**External issue:**	

Description

I've encountered a plugin comparability problem and even though I've not heard back on any of the other issues posted I thought that I would post this just in case you have a moment to provide some advise that might help me resolve the problem for myself.

When using your plugins with my Redmine installation I get a:

```
Uncaught ReferenceError: get is not defined
```

when trying to configure hooks.

I think that the comparability problem is between Hooks Manager and easyRedmine but I'm having some trouble figuring out where to look.

Could you please tell me if type:get is one of your functions (and if so where it's defined) or whether it's part of a library you use.

Thanks
Chris

Associated revisions

Revision 34 - 11 Jun 2014 22:34 - Andriy Lesyuk

Checking for remote_function instead of observe_field (#2283)

History

#1 - 23 Feb 2014 13:14 - Christopher Caruk

ok... I have tracked the problem down to a jQuery conflict. I thing that the newer jQuery expects type:'GET' rather than type:get .

The code that generates the offending lines is here:

app/views/hooks/_list.html.erb

```
              onclick="<%= defined?(remote_function) ?
                                remote_function(:url => { :action => 'load', :hook => hook },
                                                :method => :get,
                                                :condition => "(html_code_changed == false) || confirm('#{
l(:            text_html_code_not_saved)}')",
                                                :complete => "updateSelectedHook('#{hook}')") :
                               "updateHookForm('#{url_for(:action => 'load', :hook => hook, :format => 'j
s')}',         '#{hook}', '#{l(:text_html_code_not_saved)}')" %>"
```

Because the 'GET' method is the default I tried removing ':method => :get'. That stops the exception but the $.ajax jQuery function does not seem to work.

Next I tried to force the 'GET' using

```
:method => "'GET'",
```

but same as before... the exception goes away but the function itself does not do anything.

I see [loading...] but not the hook that I clicked on... So something still not working with the call.

Any ideas?

#2 - 23 Feb 2014 17:44 - Christopher Caruk

OK... the problem seems to be the lines that read: 'defined?(remote_function)'

The conflicting plugin must include a gem that includes that function.

If remote_function IS NOT used then the interface WORKS and the HTML looks like this:

```
<a id="view_layouts_base_content" href="#" onclick="updateHookForm('/hooks/load.js?hook=view_layouts_base_cont
ent', 'view_layouts_base_content', 'HTML code has not been saved and going to be lost if you continue... Are y
ou sure?')" class="icon icon-file text-plain selected">
            Content
              (bottom)
          </a>
```

if remote_function IS used then the interface is BROKEN and the HTML looks like this:

```
<a id="view_layouts_base_content" href="#" onclick="if ((html_code_changed == false) || confirm('HTML code has
 not been saved and going to be lost if you continue... Are you sure?')) { $.ajax({url:'/hooks/load?hook=view_
layouts_base_content', async:true, type:'GET'}).done(function() {updateSelectedHook('view_layouts_base_content
')}); }" class="icon icon-file text-plain">
            Content
              (bottom)
          </a>
```

If you change the lines that read “defined?(remote_function)” to: “defined?(remote_function_not)” so that it never succeeds and the JavaScript is always called instead then the admin interface does what it’s supposed to do.

The JavaScript file may also need to be modified is:

```
assets/javascripts/hooks_jquery.js
```

I changed all occurrences of "type:'get'" to "type:'GET'.

So the questions are... Why are you (Andriy) trying to use remote_function? Does it do something that your JavaScript doesn't?

It is also possible that although it suppresses the exception, the hack noted above, is wrong in some other way but it works now and I'm moving on.

If you (Andriy) eventually read this please answer these two questions. It would be nice to know if one looses something by forcing the JavaScript.

If you (the reader) have fallen into the same trap as I did then the above should provide you with enough information to get things working.

Either way, I hope that this information proves useful to someone.... I'm going to use the hack, as is, to bypass the use of remote_function and draw a line under this bug bashing session.

#3 - 10 Jun 2014 09:23 - Andriy Lesyuk

- Status changed from New to Open

- Assignee set to Andriy Lesyuk

- Target version set to 1.0.1

#4 - 11 Jun 2014 13:09 - Andriy Lesyuk

Hi, Christopher!

First: Thank you very much for the research you've made! It's helpful!

Now let me answer your questions:

> Why are you (Andriy) trying to use remote_function? Does it do something that your JavaScript doesn't?

I'm trying to use remote_function for compatibility! jQuery was added only in Redmine 2.1. Before Redmine was using prototype and helper remote_function. So, defined?(remote_function) was meant to check if Redmine < 2.1.x is used.

> It is also possible that although it suppresses the exception, the hack noted above, is wrong in some other way but it works now and I'm moving on.

I guess, it's safe to use the hack, if you are using Redmine 2.1 or above.

Also I assume, that easyRedmine uses the gem, that provides remote_function to make it compatible with some old plugins. That's why defined?(remote_function) works...

P.S. The solution would be, perhaps, to replace defined?(remote_function) with defined?(observe_field)... observe_field is another prototype-based Rails function used in Redmine < 2.1.x. It would be great if you could confirm that it worked.

#5 - 11 Jun 2014 13:21 - Christopher Caruk

Hi Andiry,

Thanks for the update.

I'll be back at work on the 26th and will give it a try then.

Chris

#6 - 11 Jun 2014 22:39 - Andriy Lesyuk

- Due date set to 11 Jun 2014

- Status changed from Open to Closed

- % Done changed from 0 to 100

I changed my mind... I think it's better to change defined?(observe_field) in index.html.erb to defined?(remote_function) to be consistent...

This way your installation will use remote_function, what is fine, if it's really a compatible version (and I think it is).

You see – the problem was that for the remote_function call the plugin loaded JS code intended for jQuery. That was done because in index.html.erb I checked not for remote_function but for observe_field (just another prototype-based function). Fixed this.

See also r34...

Currently I'm closing this issue. Please feel free to reopen or comment, if the solution does not work for you. And thanks again for the research!