Redmine - Defect #3672

Unable to set "value" and "old_value" through :helper_issues_show_detail_after_setting hook

2009-07-25 20:01 - Giovanni Junior

Status:	New	**Start date:**	2009-07-25
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Plugin API	**Estimated time:**	0.00 hour
Target version:			
Resolution:		**Affected version:**	0.8.4

Description

In http://www.redmine.org/wiki/1/Hooks, the description for :helper_issues_show_detail_after_setting states that it "Passes data to the hook to allow it to set the label and value", but it seems this is only partially true.

Setting context variables in a hook using the "=" operand just doesn't work. For "context[:label]", which is already initialized (as a String) when the hook is called, this may be worked around calling "context[:label].replace". However, as "context[:value]" and "context[:old_value]" may be **uninitialized** when the hook is called, I don't see a way to change their value at all. And setting "context[:detail].value" and "context[:detail].old_value" is obviously not as harmless as setting "context[:value]" and "context[:old_value]".

Using Redmine 0.8.4, Rails 2.1.2 and Ruby 1.8.6.

History

#1 - 2009-07-27 06:16 - Eric Davis

Can you attach a test case showing this? I've been able to set context[:value] and context[:old_value] in my Budget plugin:

http://github.com/edavis10/redmine-budget-plugin/blob/8413b26a15b74817b28509c3664da76f072ab36d/lib/budget_issue_hook.rb#L82

#2 - 2009-07-30 02:34 - Giovanni Junior

Your plugin sets context[:detail].value and context[:detail].old_value, which are different from context[:value] and context[:old_value].

Unfortunately, the code in question is not very "testable". Indeed, I have no idea how to test it.

#3 - 2010-02-18 06:37 - Eric Davis

- Status changed from New to 7

I just ran into this myself. Setting context[:object] to a new object isn't sent back to the calling code (new object reference). Setting context[:object].value works because :object is still the same object.

I've worked around this in some private code but it wasn't an easy implementation. I think we might need to split how the View Hooks return data from the Model/Controller hooks (View should merge the response strings, Model/Controller should check for reassignments).

#4 - 2010-02-19 13:39 - Thomas Löber

- File issues_helper.rb.patch added

I worked around it by adding another hook for plugin-defined detail properties (see issues_helper.rb.patch for Redmine 0.9-stable).

Then the hook code would look like this:

```
    def helper_issues_show_detail_format(context)
      detail = context[:detail]
      if detail.property == "user_notified"
        user = User.find_by_id(detail.prop_key) or return
        l(:label_user_notified, context[:html] ? link_to_user(user) : user.name)
      end
    end
```

#5 - 2010-04-24 00:42 - Pascal Schoenhardt

Couldn't this issue be fixed quite easily by assigning the value, old_value, and label defaults before calling the hook in app/helpers/issues_helper.rb?

Current

```
    call_hook(:helper_issues_show_detail_after_setting, {:detail => detail, :label => label, :value => value, :old_value => old_value })

    label ||= detail.prop_key
    value ||= detail.value
    old_value ||= detail.old_value
```

Fixed

```
    label ||= detail.prop_key
    value ||= detail.value
    old_value ||= detail.old_value

    call_hook(:helper_issues_show_detail_after_setting, {:detail => detail, :label => label, :value => value, :old_value => old_value })
```

Now the objects are initialized, and the if the hook modifies them it will get passed back.

#6 - 2010-11-27 00:40 - Eric Davis

- Assignee deleted (Eric Davis)

I am stepping down from working on Redmine. If someone else is interesting in working on this issue, feel free to reassign it to them.

Eric Davis

#7 - 2013-01-13 21:04 - Jean-Philippe Lang

- Status changed from 7 to New

Assigned issue with no assignee back to New status.

Files

issues_helper.rb.patch	844 Bytes	2010-02-19	Thomas Löber