# Redmine - Patch #5891

# New IssuesController hook for new issues

2010-07-16 00:33 - Nick Peelman

Priority:NormalDue date:Assignee:% Done:0%Category:Hook requestsEstimated time:0.00 hourTarget version:DescriptionWhile working on my CC Addresses plugin, I have come across the need to work with @issue.cc_addresses.build before render gets called.The :controller_issues_new_before_save hook only gets called after the form has been submitted, this ties the hands of trying to extend the issue form (like i'm trying to do, obviously).Adding the hook below allows you to touch the @issue object before render, letting you build nested_attribute objects, or app/controllers/issues_controller.rbIndex: app/controllers/issues_controller.rb(revision 3840)*** app/controllers/issues_controller.rb(working copy)@e =145, 9 +145, 10 @eend@issue.status = default_status(default_status] + default_status.find_new_statuses_allowed_todent.roles_for_project(@project), @issue.tracker)).uniqif request.get?    request.xhr?@issue.statt_date   = Date.todaycall_hook(:controller_issues_new_prepare, { :params => params, :issue => @issue ?elserequested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:# Check that the user is allowed to apply the requested status		2010-07-16	Start date:	New	Status:
Category:       Hook requests       Estimated time:       0.00 hour         Target version:       Description         While working on my CC Addresses plugin, I have come across the need to work with @issue during its creation in orde @issue.cc_addresses.build before render gets called.         The :controller_issues_new_before_save hook only gets called after the form has been submitted, this ties the hands of trying to extend the issue form (like i'm trying to do, obviously).         Adding the hook below allows you to touch the @issue object before render, letting you build nested_attribute objects, or app/controllers/issues_controller.rb         app/controllers/issues_controller.rb         app/controllers/issues_controller.rb         @e = nd         @issue.status = default_status         @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_tcd         ent.roles_for_project (@project), @issue.tracker)).uniq         if request.get?    request.xhr?         @issue.status = losueStatus.find_py_id(params[:issue][:status_id]) if params[:			Due date:	Normal	Priority:
Target version:         Description         While working on my CC Addresses plugin, I have come across the need to work with @issue during its creation in orde @issue.cc_addresses.build before render gets called.         The :controller_issues_new_before_save hook only gets called after the form has been submitted, this ties the hands or trying to extend the issue form (like i'm trying to do, obviously).         Adding the hook below allows you to touch the @issue object before render, letting you build nested_attribute objects, or app/controllers/issues_controller.rb         app/controllers/issues_controller.rb         app/controllers/issues_controller.rb         app/controllers/issues_controller.rb         (revision 3840)         +++ app/controllers/issues_controller.rb         @e = 145, 9 +145, 10 @@         end         @issue.status = default_status         @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_tode         ent.roles_for_project(@project), @issue.tracker)).uniq         if request.get?    request.xhr?         @issue.statt_date   = Date.today         +       call_hook(:controller_issues_new_prepare, { :params => params, :issue => @issue }         else       requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:		0%	% Done:		Assignee:
<pre>While working on my CC Addresses plugin, I have come across the need to work with @issue during its creation in orde @issue.cc_addresses.build before render gets called. The :controller_issues_new_before_save hook only gets called after the form has been submitted, this ties the hands of trying to extend the issue form (like i'm trying to do, obviously). Adding the hook below allows you to touch the @issue object before render, letting you build nested_attribute objects, et Index: app/controllers/issues_controller.rb  app/controllers/issues_controller.rb (revision 3840) +++ app/controllers/issues_controller.rb (working copy) @@ -145, 9 +145, 10 @@ end @issue.status = default_status @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today +</pre>		0.00 hour	Estimated time:	Hook requests	Category:
<pre>While working on my CC Addresses plugin, I have come across the need to work with @issue.during its creation in ord @issue.cc_addresses.build before render gets called. The :controller_issues_new_before_save hook only gets called after the form has been submitted, this ties the hands of trying to extend the issue form (like i'm trying to do, obviously). Adding the hook below allows you to touch the @issue object before render, letting you build nested_attribute objects, of Index: app/controllers/issues_controller.rb ===================================</pre>					Target version:
<pre>@issue.cc_addresses.build before render gets called. The :controller_issues_new_before_save hook only gets called after the form has been submitted, this ties the hands of trying to extend the issue form (like i'm trying to do, obviously). Adding the hook below allows you to touch the @issue object before render, letting you build nested_attribute objects, a Index: app/controllers/issues_controller.rb  app/controllers/issues_controller.rb (revision 3840) +++ app/controllers/issues_controller.rb (working copy) 00 -145, 9 +145,10 00 end 0 issue.status = default_status 0 allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(0project), 0 issue.tracker)).uniq if request.get?    request.xhr? 0 issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; 0 issue } 0 else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>					Description
<pre>trying to extend the issue form (like i'm trying to do, obviously). Adding the hook below allows you to touch the @issue object before render, letting you build nested_attribute objects, a Index: app/controllers/issues_controller.rb app/controllers/issues_controller.rb (revision 3840) +++ app/controllers/issues_controller.rb (working copy) @@ -145,9 +145,10 @@ end @@issue.status = default_status @@allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[: </pre>	der to call	sue during its creation in order to ca	need to work with @iss		• •
<pre>Index: app/controllers/issues_controller.rb app/controllers/issues_controller.rb (revision 3840) +++ app/controllers/issues_controller.rb (working copy) @@ -145,9 +145,10 @@ end @issue.status = default_status @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_tc ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>	of plugins	omitted, this ties the hands of plugin	the form has been sub		
<pre>Index: app/controllers/issues_controller.rb app/controllers/issues_controller.rb (revision 3840) +++ app/controllers/issues_controller.rb (working copy) @@ -145,9 +145,10 @@ end @issue.status = default_status @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_tc ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>	. etc.	ild nested attribute objects. etc.	e render. letting vou buil	allows you to touch the @issue object befc	Adding the hook below
<pre> app/controllers/issues_controller.rb (revision 3840) +++ app/controllers/issues_controller.rb (working copy) @@ -145,9 +145,10 @@ end @issue.status = default_status @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>	010.		, lotting you built		
<pre> app/controllers/issues_controller.rb (revision 3840) +++ app/controllers/issues_controller.rb (working copy) @@ -145,9 +145,10 @@ end @issue.status = default_status @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>				_	
<pre>+++ app/controllers/issues_controller.rb (working copy) @@ -145,9 +145,10 @@ end @issue.status = default_status @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>		-	ion 3840)		
<pre>end @issue.status = default_status @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>			ng copy)	ers/issues_controller.rb (work	+++ app/controlle
<pre>@issue.status = default_status @allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(@project), @issue.tracker)).uniq  if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>				) @@	
<pre>@allowed_statuses = ([default_status] + default_status.find_new_statuses_allowed_to ent.roles_for_project(@project), @issue.tracker)).uniq if request.get?    request.xhr? @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>				s = default status	
<pre>ent.roles_for_project(@project), @issue.tracker)).uniq  if request.get?    request.xhr?    @issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else    requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>	o(User.curr	w_statuses_allowed_to(User	lt_status.find_ne	_	-
<pre>@issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>					
<pre>@issue.start_date   = Date.today + call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>			uniig		if we muse to
<pre>+ call_hook(:controller_issues_new_prepare, { :params =&gt; params, :issue =&gt; @issue } else requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>			anty	sot? II request whr?	
<pre>else    requested_status = IssueStatus.find_by_id(params[:issue][:status_id]) if params[:</pre>	})	ns, :issue => @issue })			-
			-	art_date   = Date.today	@issue.sta
# Check that the user is allowed to apply the requested status	:issue]		-	art_date   = Date.today	<pre>@issue.sta + call_hook</pre>
		atus_id]) if params[:issue	:params => param	art_date   = Date.today (:controller_issues_new_prepare,	@issue.sta + call_hook else
Would be great to see this in the next release, its a relatively minor change. Thanks!			:params => param arams[:issue][:st	art_date   = Date.today (:controller_issues_new_prepare, _status = IssueStatus.find_by_id(	@issue.sta + call_hook else requested

### History

# #1 - 2010-07-18 19:34 - Nick Peelman

I also found I needed this as well:

The current hook at the bottom of the page is nice (though I haven't found a use for it yet) but one at the bottom of the footer like this would make it easier for groups or whoever to add their own copyright or other footer to the bottom, without needing to manually change base.rhtml.

### #2 - 2010-07-28 22:38 - Nick Peelman

Due to the changes in 1.0, the diff should look like this:

```
peelman@ubuntu:/var/www/redmine-1.0$ svn diff app/controllers/issues_controller.rb
Index: app/controllers/issues_controller.rb
```

```
--- app/controllers/issues_controller.rb (revision 3892)
+++ app/controllers/issues_controller.rb (working copy)
@@ -466,6 +466,7 @@
@issue.start_date ||= Date.today
@priorities = IssuePriority.all
@allowed_statuses = @issue.new_statuses_allowed_to(User.current, true)
+ call_hook(:controller_issues_new_prepare, { :params => params, :issue => @issue })
end
```

\_\_\_\_\_

def set\_flash\_from\_bulk\_issue\_save(issues, unsaved\_issue\_ids)

\_\_\_\_\_

## #3 - 2010-08-06 00:42 - Nick Peelman

#### Nobody?

#### #4 - 2010-08-06 02:21 - Eric Davis

- Subject changed from New IssuesController hook ... to New IssuesController hook for new issues

- Category changed from Issues to Plugin API

Nick Peelman wrote:

Nobody?

Sorry, there are over 2,000 issues here and less than half a dozen people patrolling the issue list. Sometimes it takes us a bit to get everything.

The hook in the footer is a good one, I'll need to review the one in the controller to see what you need. If you are only adding some objects to new issues, it might be better to hook into Issue's after\_initialize that way it's always run.

### #5 - 2010-08-06 02:29 - Nick Peelman

Eric Davis wrote:

Sorry, there are over 2,000 issues here and less than half a dozen people patrolling the issue list. Sometimes it takes us a bit to get everything.

The hook in the footer is a good one, I'll need to review the one in the controller to see what you need. If you are only adding some objects to new issues, it might be better to hook into Issue's after\_initialize that way it's always run.

I work on <u>CoRD</u>, or at least i try to. I know the feeling when there's more issues than you have hands. 'Tis one reason I'm working my way up towards helping out more and pitching in with some patches and code here. I do my best to be patient and hack together my own solutions in the mean while :)

You may be right, after\_initialize might be a better way. But when you're doing things like I'm doing, that are dependent on modules being enabled and the user having permissions for those modules, how would you go about adding that kind of logic into the issue model/helper? Seems like a better candidate for a controller action, but I'm still learning how to swim in this soup called Ruby...

## #6 - 2016-12-30 09:19 - Go MAEDA

- Category changed from Plugin API to Hook requests