Redmine - Feature #1143

Hooks for plugins

2008-04-29 14:04 - Carl Nygard

Status:	Closed	**Start date:**	2008-04-29
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	0.8		
Resolution:			

Description

This feature is just an idea, I don't have any particular application driving the requirement, although I think it could be used to allow people a bit more flexibility in implementing some of the feature requests into non-core plugins.

The technical requirements would be pre- and post-xxx hook functions defined around the major functions. Combined with ruby's ability to inject code into classes and redefine functions on the fly, I could create a module that customized the behavior of the system in small ways without patching the code directly.

For example, with a pre-issue-change hook I could implement issue reassignment based on the current state of the issue, so that new issues always went to the same person for triage, then when the bug is accepted it's assigned to the normal user-per-category. Then when resolved it could automatically get reassigned to QA person.

All it would take is some dummy functions that get fed pertinent data, which would allow a plugin writer to redefine for their own purposes.

Related issues:

Related to Redmine - Feature #783: Real Plugin-System	**Closed**	**2008-03-04**
Related to Redmine - Patch #1677: Plugin hook API	**Closed**	**2008-07-24**

History

#1 - 2008-04-30 00:47 - Eric Davis

I've been thinking about this a lot lately and it's one thing that I think Wordpress did good on. A practical example would be allow a plugin to hook into the issue edit page to render some extra HTML after the main form. I'd like to just be able to hook into it like:

```
  Redmine.hooks[:post_issue_edit] << MyPluginController.action
```

Jean-Philippe Lang, I have a plugin coming up that we could use as an experiment for this API.

#2 - 2008-06-06 05:53 - Eric Davis

I wanted to post an update to this feature. I have successfully added hooks into Redmine and the Redmine Plugin API. It still needs some cleaning up before release but it's working great for a plugin I'm writting. Similar to Wordpress plugins, you can have a hook that inserts HTML or does an action. The plugin registers for a hook buy passing in a Ruby Proc to the add_hook method:

```
# Plugin's init.rb
add_hook(:issue_show, Proc.new { |context| MyClass.my_method(context) })
```

To add a hook point, the core just needs a single line. Using the example above:

```
# app/views/issues/show.rhtml
<%= Redmine::Plugin::Hook.call_hook(:issue_show, {:project => @project, :issue => @issue}) %>
```

I'd like to have these hooks are critical points and we can also wrap some methods with around filters to do the pre/post processing.

#3 - 2008-07-02 19:53 - Eric Davis

- Status changed from New to 7

- Assignee changed from Jean-Philippe Lang to Eric Davis

#4 - 2008-07-02 21:09 - Paul Rivier

Eric Davis wrote:

> The plugin registers for a hook buy passing in a Ruby Proc to the add_hook method:

Very minor suggestion, but could you rename it to add_to_hook, which is what really happens, please ? Emacs did it wrong decades ago and we are now stuck with this improper name 'add_hook' :)

Other than that, it might be a useful facility, thanks.

#5 - 2008-07-24 06:15 - Eric Davis

- Assignee deleted (Eric Davis)

I've just added a patch for the Plugin hook API on #1677.

#6 - 2008-09-10 04:31 - Eric Davis

- Status changed from 7 to Closed

Plugin hooks were added in #1677