Redmine - Patch #1677

Plugin hook API

2008-07-24 06:14 - Eric Davis

Status:	Closed	**Start date:**	2008-07-24
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	100%
Category:	Plugin API	**Estimated time:**	0.00 hour
Target version:	0.8		

Description

This patch to adds support for Plugin hooks into Redmine. By providing Plugin hooks, users can create custom plugins to modify Redmine's behavior without modifying the core. This patch includes a new Plugin API method, a few hooks in the Redmine core, a base hook class, and internal methods to manage and call the hooks.

Plugin API - add_hook(hook_name, method)

Registers a method to be called when Redmine runs a hook called hook_name.

```
# Run puts whenever the issue_show hook is called
add_hook :issue_show, Proc.new { puts 'Hello' }

# Call the class method +my_method+ passing in all the context
add_hook :issue_show, Proc.new {|context| MyPlugin.my_method(context)}
```

Hooks in Redmine core

Potentially anyplace in the Redmine core can have a hook point added. I added a few to some areas I needed:

- Issue show view
- Issue edit form
- Issue bulk edit view
- Issue bulk edit save (in IssuesController)
- Issue update view
- The Project's members page
- IssueHelper#show_details helper

New hooks can be added by adding a symbol to Redmine::Plugin::Hook::Manger::hooks and adding the call_hook method in the core, passing in any parameters.

```
# Will call `my_new_hook` with the @project and @issue objects
Redmine::Plugin::Hook::Manager.call_hook(:my_new_hook, {:project => @project, :issue => @issue})
```

Base Hook Class

A new class is added called Redmine::Plugin::Hook::Base which can be subclassed by plugins to use Rails Helpers with the provided help object.

Internal modules

Two new classes are added: the Redmine::Plugin::Hook::Base mentioned earlier and Redmine::Plugin::Hook:Manager. The Manager class is responsible for tracking what plugins are registered for which hooks and to call those plugins once a call_hook method is run.

The attached file applies cleanly to svn r1694. I'll be maintaining this patch on Github on the plugin-hooks branch and would love feedback and patches against the code.

Related issues:			
Related to Redmine - Feature #1143: Hooks for plugins		**Closed**	**2008-04-29**
Related to Redmine - Feature #783: Real Plugin-System		**Closed**	**2008-03-04**

Associated revisions

Revision 1737 - 2008-08-13 02:43 - Eric Davis

Adds :view_issues_form_details_bottom hook. #1677

Revision 1738 - 2008-08-13 02:43 - Eric Davis

Adds :view_issues_bulk_edit_details_bottom hook. #1677

Revision 1739 - 2008-08-13 02:43 - Eric Davis

Added hooks to the Project members table

- :view_projects_settings_members_table_header
- :view_projects_settings_members_table_row

    #1677

Revision 1740 - 2008-08-13 02:43 - Eric Davis

Adds :controller_issues_bulk_edit_before_save hook. #1677

Revision 1741 - 2008-08-13 02:43 - Eric Davis

Adds :helper_issues_show_detail_after_setting hook. #1677

Revision 1746 - 2008-08-13 17:34 - Eric Davis

Removing Redmine::Hook::HOOKS because it's not used any more with the dynamic hook loading. #1677

History

#1 - 2008-07-24 06:17 - Eric Davis

I have a large plugin getting ready to be Open Sourced that can serve as an example of the working API. I'll update this issue with it's details after it's released.

#2 - 2008-07-25 01:06 - Martin Herr

Hey eric,

I think you're doing the exactly right thing with your work for the redmine plugin architecture! Rock on!

:)

#3 - 2008-07-27 11:52 - Jean-Philippe Lang

No tests ?

#4 - 2008-07-28 22:04 - Eric Davis

Jean-Philippe Lang wrote:

> No tests ?

I don't see any existing tests for the plugin API. Where should the tests be stored?

#5 - 2008-07-28 23:00 - Jean-Philippe Lang

There are a few tests for underlying modules. See ProjectsControllerTest#test_project_menu.
I don't claim that every single line of code is tested and I know it saddens you :-/ but providing a few tests with this patch would be appreciated.
Tests are stored in /test :-)

#6 - 2008-07-29 01:02 - Eric Davis

- Status changed from New to 7

- Assignee set to Eric Davis

Jean-Philippe Lang wrote:

> There are a few tests for underlying modules. See ProjectsControllerTest#test_project_menu.

So the lib code is tested inline. Would it be ok if I moved these into a unit test, like "test/unit/lib/redmine/plugin_test.rb" (Trying to mirror the lib directory layout).

> I don't claim that every single line of code is tested and I know it saddens you :-/ but providing a few tests with this patch would be appreciated.

Not a problem, I just didn't want to submit a massive patch building up test support for all of lib/redmine

> Tests are stored in /test :-)

When can they be stored in /spec ;-)

#7 - 2008-07-29 02:36 - Eric Davis

- File redmine-hooks-svn-1709.patch added

- Assignee deleted (Eric Davis)

Jean-Philippe Lang wrote:

> No tests ?

Attached is an updated patch for the Plugin hook API along with unit tests. It's been awhile since I've done pure Test::Unit so an extra review would be appreciated.

It applies cleanly to trunk [r1709](#).

#8 - 2008-08-03 14:13 - Jean-Philippe Lang

Hi Eric, I had a look at your patch but I don't understand how Redmine::Plugin::Hook::Base is used.
Could you give an example for a hook that generates a link ?

Also, some errors occur when running plugin_test:

```
Loaded suite test/unit/lib/redmine/plugin_test
Started
.EE.
Finished in 0.046 seconds.

  1) Error:
test_add_hook(Redmine::PluginTest):
NoMethodError: undefined method `add_hook' for Redmine::Plugin:Class
    test/unit/lib/redmine/plugin_test.rb:10:in `test_add_hook'
    d:/dev/ruby/lib/ruby/gems/1.8/gems/activesupport-2.1.0/lib/active_support/te
sting/setup_and_teardown.rb:67:in `__send__'
    d:/dev/ruby/lib/ruby/gems/1.8/gems/activesupport-2.1.0/lib/active_support/te
sting/setup_and_teardown.rb:67:in `run'

  2) Error:
test_add_hook_invalid(Redmine::PluginTest):
NoMethodError: You have a nil object when you didn't expect it!
You might have expected an instance of Array.
The error occurred while evaluating nil.size
    H:/trunk/lib/redmine/plugin.rb:214:in `hook_registered?'
    test/unit/lib/redmine/plugin_test.rb:15:in `test_add_hook_invalid'
    d:/dev/ruby/lib/ruby/gems/1.8/gems/activesupport-2.1.0/lib/active_support/te
sting/setup_and_teardown.rb:67:in `__send__'
    d:/dev/ruby/lib/ruby/gems/1.8/gems/activesupport-2.1.0/lib/active_support/te
sting/setup_and_teardown.rb:67:in `run'

4 tests, 2 assertions, 0 failures, 2 errors
```

#9 - 2008-08-04 07:46 - Eric Davis

Jean-Philippe Lang wrote:

> Hi Eric, I had a look at your patch but I don't understand how Redmine::Plugin::Hook::Base is used.
> Could you give an example for a hook that generates a link ?

Gladly. A basic hook would be:

```
# lib/link_example.rb
class LinkExample < Redmine::Plugin::Hook::Base
  def self.linking_to_issues
    return help.link_to(:controller => 'issues', :action => 'index')
  end
end


# init.rb
add_hook(:issue_show, Proc.new { LinkExample.linking_to_issues })
```

A more advanced example of a hook creating a full Ajax form can be found on my [budget plugin](#)

```
  # Renders an AJAX from to update the member's billing rate
  #
  # Context:
  # * :member => Current Member record
  #
  def self.member_list_column_three(context = { })
    if context[:project].module_enabled?('budget_module')
      # Build a form_remote_tag by hand since this isn't in the scope of a controller
      form = help.form_tag({:controller => 'members', :action => 'edit', :id => context[:member].id, :protocol
 => Setting.protocol, :host => Setting.host_name},
                            :onsubmit => help.remote_function(:url => {
                                                                :controller => 'members',
                                                                :action => 'edit',
                                                                :id => context[:member].id,
                                                                :protocol => Setting.protocol,
                                                                :host => Setting.host_name
                                                              },
                                                              :host => Setting.host_name,
                                                              :protocol => Setting.protocol,
                                                              :form => true,
                                                              :method => 'post',
                                                              :return => 'false' )+ '; return false;') +
        help.text_field_tag('member[rate]', help.number_with_precision(context[:member].rate, 0), :class => "s
mall") +
        help.submit_tag(GLoc.l(:button_change), :class => "small") + "</form>"

      return help.content_tag(:td, form, :align => 'center' )
    else
      return ''
    end
  end
```

Also, some errors occur when running plugin_test:

When the patch was applied did the add_hook method get added to plugin.rb without conflicts?

#10 - 2008-08-05 18:29 - Jean-Philippe Lang

- File hook.rb added

add_hook(:issue_show, Proc.new { LinkExample.linking_to_issues })

I think we could have a more straightforward hooks implementation.
See the attached file, it's just a work in progress.

Using this implementation, hooks are automatically added when inheriting from Redmine::Hook::Listener. You just need to match method names and
hook names.
You can also inherit from the subclass Redmine::Hook::ViewExtensionListener instead to have access to various helpers in your hook methods.

Example for adding a listener to the :issue_show hook.

```
class HelloWorld < Redmine::Hook::ViewExtensionListener
  def issue_show(context)
    textilizable('_Hello world!_')
  end
end
```

What do you think ?

#11 - 2008-08-05 19:48 - Eric Davis

Jean-Philippe Lang wrote:

> Using this implementation, hooks are automatically added when inheriting from Redmine::Hook::Listener.

I like that approach. I like being explicit about who is hooking who but this version is a lot easier to understand.

> You can also inherit from the subclass Redmine::Hook::ViewExtensionListener instead to have access to various helpers in your hook methods.

That would be great if it works. I had to create a Singleton object to access the view helpers (help). I can't remember why now though, it could have just been me being wrong.

> What do you think ?

I'm assuming the views would call Redmine::Hook.call_hook(:issue_show, {}) to run the hooks right?

Looks good to me. I can help write up some documentation and tests once we have something in Subversion. I'll need to port my plugin to the new API so the new API should get a good work out then. I'll have a bunch of new patches for the API then (e.g. adding new hook points, allowing plugins to add and call their own hooks...)

#12 - 2008-08-06 13:11 - Jean-Philippe Lang

> That would be great if it works. I had to create a Singleton object to access the view helpers (help). I can't remember why now though, it could have just been me being wrong.

It works. Actually, hook listeners are Singleton objects (instanciated on the first call by Redmine::Hook.listeners).

```
class Listener
  include Singleton
  ...
```

> I'm assuming the views would call Redmine::Hook.call_hook(:issue_show, {}) to run the hooks right?

Views cas use the following to call a hook since the module that provides this shortcut (Redmine::Hook::Helper) is included in ApplicationHelper (see last line):

```
<%= call_hook(:issue_show, {}) %>
```

> Looks good to me. I can help write up some documentation and tests once we have something in Subversion. I'll need to port my plugin to the new API so the new API should get a good work out then. I'll have a bunch of new patches for the API then (e.g. adding new hook points, allowing plugins to add and call their own hooks...)

I'll try to commit an initial working API asap.
Do you have a rubyforge account so I can give you commit permissions on the repository, if you're insterested of course ?

#13 - 2008-08-06 17:35 - Eric Davis

Jean-Philippe Lang wrote:

> It works. Actually, hook listeners are Singleton objects (instanciated on the first call by Redmine::Hook.listeners).

That sounds really good. That way the plugin developers don't need to create class methods in for hooks to work.

> I'll try to commit an initial working API asap.
> Do you have a rubyforge account so I can give you commit permissions on the repository, if you're insterested of course ?

I'm interested. Then I can stop complaining about missing tests and just fix them myself. :)

My RubyForge account is edavis10

#14 - 2008-08-07 20:54 - Jean-Philippe Lang

Initial implementation is committed in r1717. I've added a branch for this: source:branches/work/hooks.
Eric, you should now be able to commit on redmine repository hosted at rubyforge. See http://rubyforge.org/scm/?group_id=1850 for details.

I'd like discuss about the hooks naming convention before adding them all around.
Here is what I propose for hooks in views:

```
view_[path_to_the_view]_[position_in_view]
```

Examples:

```
view_layouts_base_html_head -- the only one that was committed
view_issues_show_details_bottom
view_issues_list_table_header
view_issues_list_table_row
...
```

I've created a wiki page to document existing hooks: Hooks.

#15 - 2008-08-08 02:05 - Eric Davis

Jean-Philippe Lang wrote:

> Initial implementation is committed in r1717. I've added a branch for this: source:branches/work/hooks.

Great.

> Eric, you should now be able to commit on redmine repository hosted at rubyforge. See http://rubyforge.org/scm/?group_id=1850 for details.

Looks like it works, I was able to checkout from the private url. Now to get git linked up.

> I'd like discuss about the hooks naming convention before adding them all around.

Totally agree.

> Here is what I propose for hooks in views:

> view_[path_to_the_view]_[position_in_view]

That works great for views. What about controller hooks? Maybe:

```
controller_[controller_name]_[action]_[position]

controller_issues_show_index
controller_issues_edit
controller_issues_edit_post  # For after `if request.post?`, when saving.
```

#16 - 2008-08-13 03:03 - Eric Davis

I've added the hooks from my patch into source:branches/work/hooks (r1737-r1741)

#17 - 2008-08-13 03:09 - Eric Davis

I just added a rake task to list the Redmine hooks. This should help manage the list. rake redmine:plugins:hook_list

```
$ rake redmine:plugins:hook_list
(in /home/edavis/dev/redmine/redmine-core/hooks)
app/controllers/issues_controller.rb:
  * [237] :controller_issues_bulk_edit_before_save, { :params => params, :issue => issue })

app/helpers/issues_helper.rb:
  * [ 89] :helper_issues_show_detail_after_setting, {:detail => detail, :label => label, :value => value, :old
_value => old_value })

app/views/issues/_form.rhtml:
  * [ 51] :view_issues_form_details_bottom, { :issue => @issue, :form => f })

app/views/issues/bulk_edit.rhtml:
  * [ 41] :view_issues_bulk_edit_details_bottom, { :issues => @issues })

app/views/issues/show.rhtml:
  * [ 56] :view_issues_show_details_bottom, :issue => @issue)

app/views/layouts/base.rhtml:
  * [ 17] :view_layouts_base_html_head
```

```
app/views/projects/settings/_members.rhtml:
  * [ 13] :view_projects_settings_members_table_header)
  * [ 34] :view_projects_settings_members_table_row, :member => @member)
```

#18 - 2008-08-13 08:20 - Jean-Philippe Lang

great, and i think we could remove Hook::HOOKS since it's not really used and already obsolete (i forgot to update it too).

#19 - 2008-08-13 17:32 - Eric Davis

Jean-Philippe Lang wrote:

> great, and i think we could remove Hook::HOOKS since it's not really used and already obsolete (i forgot to update it too).

I saw that too but didn't get a chance to remove it yesterday. I just removed it and it's helper methods def hooks and def valid_hook? in r1746.

#20 - 2008-08-20 07:42 - Joshua Hoover

Hi Eric,

Great work! I'm going to start building a new plugin for making things a bit easier for teams using Redmine and are doing agile/scrum. Do you think I should give it a go with this new API using the branch you have going? Any gotchas I should look out for other than "it's still early"? :)

Thanks!

Josh

#21 - 2008-08-22 07:08 - Eric Davis

Joshua Hoover wrote:

> Great work! I'm going to start building a new plugin for making things a bit easier for teams using Redmine and are doing agile/scrum. Do you think I should give it a go with this new API using the branch you have going? Any gotchas I should look out for other than "it's still early"? :)

No real gotchas, the concept is sound and there is some test coverage on it. If you want a more definate answer, wait until the branch is merged into trunk. I don't see any reason why it couldn't be merged in soon (other than the fact that merging in svn is painful).

#22 - 2008-08-22 14:57 - Jean-Philippe Lang

It will be merged soon indeed.

#23 - 2008-09-10 03:49 - Eric Davis

- Status changed from 7 to Closed

- Target version set to 0.8

- % Done changed from 0 to 100

Plugin API hooks merged in r1786.

Files

redmine-hooks-svn-1694.patch	7.22 KB	2008-07-24	Eric Davis
redmine-hooks-svn-1709.patch	12.9 KB	2008-07-29	Eric Davis
hook.rb	1.69 KB	2008-08-05	Jean-Philippe Lang