Redmine - Patch #3264

Allow issue edit exceptions to choke nicely

2009-04-29 20:27 - Brad Beattie

Status:	Closed	**Start date:**	2009-04-29
Priority:	Normal	**Due date:**	
Assignee:	Eric Davis	**% Done:**	0%
Category:	Plugin API	**Estimated time:**	0.00 hour
Target version:			

Description

The attached patch allows issues raised by controller_issues_edit_before_save and controller_issues_edit_after_save to choke nicely, instead of bringing the user to an intimidating grey page.

The purpose of this is so that a plugin can use the controller_issues_edit_before_save hook and deny editing on specific conditions (e.g. an issue can't have it's assigned_to revoked after it's been set once, specific roles can't remove due dates, etc). Without this modification, plugins can't stop issues from saving.

History

#1 - 2009-04-29 20:34 - Brad Beattie

Brad Beattie wrote:

> Without this modification, plugins can't stop issues from saving.

My which I mean, plugins can stop issues from saving, but only in a way that looks like the site broke.

#2 - 2009-05-06 06:56 - Eric Davis

Couldn't a plugin just observe a before_save callback on Issue and Journal and return false if the issue shouldn't be saved?

```
class AnObserver < ActiveRecord::Observer
  observe :journal

  def before_save(journal)
    # allowed_to_edit? would be the plugin's custom behavior
    if journal.allowed_to_edit?
      return true
    else
      return false
    end
  end
end
```

#3 - 2009-05-06 17:35 - Brad Beattie

Eric Davis wrote:

> Couldn't a plugin just observe a before_save callback on Issue and Journal and return false if the issue shouldn't be saved?

Looking at issues_controller, I don't see how that would work.

```
196          call_hook(:controller_issues_edit_before_save, { :params => params, :issue => @issue, :time_entry
 => @time_entry, :journal => journal})
197
198          if (@time_entry.hours.nil? || @time_entry.valid?) && @issue.save
```

The hook is called and then we make an attempt to save the issue (and any changes therein). Unless the call_hook someone stops the saving from happening, the changes get applied, right?

#4 - 2009-05-13 01:37 - Eric Davis

Brad Beattie wrote:

> The hook is called and then we make an attempt to save the issue (and any changes therein). Unless the call_hook someone stops the saving from happening, the changes get applied, right?

If any before saves on the object (Issue) return false, the @issue.save will fail and the record wouldn't be updated. So in this case you plugin can add a before_save callback to prevent the issue/journal from saving. Then what will happen is the Issue edit page will be displayed with any flash messages for the user.

#5 - 2009-05-13 18:58 - Brad Beattie

Eric Davis wrote:

> Brad Beattie wrote:
>
>> The hook is called and then we make an attempt to save the issue (and any changes therein). Unless the call_hook someone stops the saving from happening, the changes get applied, right?
>
> If any before saves on the object (Issue) return false, the @issue.save will fail and the record wouldn't be updated. So in this case you plugin can add a before_save callback to prevent the issue/journal from saving. Then what will happen is the Issue edit page will be displayed with any flash messages for the user.

Hrm. I'm not sure if I'm doing it wrong or if it doesn't work as you describe (likely the former, but let's confirm). I have a plugin that calls the following hook. By what you say, this hook should stop the issue from saving, yeah? For some reason, it doesn't.

```
class WorkflowIssueEditSaveHook < Redmine::Hook::ViewListener
  def controller_issues_new_before_save(context = { })
    return false
  end
end
```

#6 - 2009-05-13 19:06 - Brad Beattie

To clarify, I also tried your example above.

```
class AnObserver < ActiveRecord::Observer
  observe :journal

  def before_save(journal)
    return false
  end
end
```

#7 - 2009-05-19 20:14 - Eric Davis

Brad Beattie:

Returning false in a hook won't do anything. Returning false in a before_save should prevent the record from saving. Did you register the observer with ActiveRecord::Base.observers?

```
# init.rb
ActiveRecord::Base.observers << :an_observer
```

#8 - 2009-05-27 01:41 - Brad Beattie

Eric Davis wrote:

> Brad Beattie:
>
> Returning false in a hook won't do anything. Returning false in a before_save should prevent the record from saving. Did you register the observer with ActiveRecord::Base.observers?
>
> [...]

Yeah, but with no luck.

init.rb

```
require 'redmine'
require_dependency 'an_observer'
ActiveRecord::Base.observers << :an_observer

Redmine::Plugin.register :redmine_workflow do
  name 'Redmine Workflow plugin'
  author 'Brad Beattie'
  description 'This plugin provides instances of Redmine with customizable workflow restrictions.'
```

```
  version '0.1.0'
end
```

lib/an_observer.rb

```
class AnObserver < ActiveRecord::Observer
  observe :journal

  def before_save(journal)
    return false
  end
end
```

I can successfully make changes and comment on issues with this mini-plugin. Is there something I'm missing here?

#9 - 2009-05-30 01:38 - Eric Davis

Brad Beattie:

Did we resolve this on IRC the other day?

#10 - 2009-05-30 01:43 - Brad Beattie

- Status changed from New to Resolved

That we did.

#11 - 2009-05-30 02:36 - Eric Davis

- Status changed from Resolved to Closed

For searchers, the problem was that an ActiveRecord::Observer couldn't return false to abort saving the record. The solution is to define the before_save on the Model class directly, via a monkey patch.

http://dev.rubyonrails.org/ticket/7968

Files

patch.diff	569 Bytes	2009-04-29		Brad Beattie