Redmine - Patch #4665
Projects not calling EnabledModule callbacks in some cases

2010-01-27 10:04 - Enrique Garcia

Status:	New	**Start date:**	2010-01-27
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Projects	**Estimated time:**	0.00 hour
Target version:			

Description

Implementation of Project.enabled_module_names= still calls modules.clear() in some cases (if nil is passed as parameter).

This has the risk of not invoking module callbacks (before_create, after_create, before_destroy, after_destroy) correctly.

Proposed change is to remove all traces of modules.clear(). Example:

```
    def enabled_module_names=(module_names=[])
      module_names = module_names.collect(&:to_s)
      # remove disabled modules
      enabled_modules.each {|mod| mod.destroy unless module_names.include?(mod.name)}
      # add new modules
      module_names.reject {|name| module_enabled?(name)}.each {|name| enabled_modules << EnabledModule.new(:name
=> name)}
    end
```

Regards!

History

#1 - 2010-07-09 13:10 - Felix Schäfer

Could you please give a way to reproduce any error this might cause?

#2 - 2010-07-09 17:35 - Enrique Garcia

Ok.

This is the current implementation:

```
  def enabled_module_names=(module_names)
    if module_names && module_names.is_a?(Array)
      module_names = module_names.collect(&:to_s)
      # remove disabled modules
      enabled_modules.each {|mod| mod.destroy unless module_names.include?(mod.name)}
      # add new modules
      module_names.reject {|name| module_enabled?(name)}.each {|name| enabled_modules << EnabledModule.new(:name => name)}
    else
      enabled_modules.clear # this doesn't invoke callbacks
    end
  end
```

I believe there is no way to reproduce this in "vanilla" Redmine - it never does project.enabled_modules = nil.

But it is a problem for plugin development.

The way the method is constructed, it seems to say "I accept either a list of modules, or nil, and I do different things depending on that".

The "nil" part isn't used on redmine, I believe. It is allways a list of modules. (I suspect this isn't included on the tests either). I believe the code there is wrong because enabled_modules.clear eliminates the list of modules *without* invoking the corresponding callbacks. (eg before_destroy) on the modules.

The proposed solution is more straightforward - the code says "give me a list of modules. if you give me nil, I'll make it an empty list". And callbacks are allways called. No "nil special case". It is also shorter.

I was "bitten" by this code myself when developing a plugin - was confused by the "if", tried to use it, and module callbacks stopped working, and I went bonkers for 3 days.

By the way, I submitted this question to Eric via email. The code on the OP is his - I copy-pasted it from his e-mail. He asked me to open a ticket and assign it to him, and I did so. I didn't notice that he did not action on it until today.

If you need any more clarifications, just let me know.

#3 - 2010-10-25 17:00 - Eric Davis

- Assignee deleted (Eric Davis)