Redmine - Patch #11827

Avoid retrieving memberships for projects jump box

2012-09-13 09:37 - Jean-Baptiste Barth

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Jean-Baptiste Barth	**% Done:**	0%
Category:	Code cleanup/refactoring	**Estimated time:**	0.00 hour
Target version:			

Description

ApplicationHelper#render_project_jump_box retrieves current user's memberships (and roles, see the association..) so that it can render user's projects. From what I seen it's unecessary, but it comes from an inverse optimization in r5153 (maybe it was relevant in Rails 2?). Retrieving directly user's projects could be better :

- saves a complex query : on a large instance at work, the average staff user has 120 projects and the double join in the SQL query doesn't come for free ;
- saves a lot of objects instanciations (Memberships)

So here's the proposal:

```
diff --git a/app/helpers/application_helper.rb b/app/helpers/application_helper.rb
index 794cc6f..6c1f919 100644
--- a/app/helpers/application_helper.rb
+++ b/app/helpers/application_helper.rb
@@ -237,7 +237,8 @@ module ApplicationHelper
   # Renders the project quick-jump box
   def render_project_jump_box
     return unless User.current.logged?
-    projects = User.current.memberships.collect(&:project).compact.uniq
+    #'to_a' forces immediate querying, which saves a 'count' query, see two lines below
+    projects = User.current.projects.active.to_a
     if projects.any?
       s = '<select onchange="if (this.value != \'\') { window.location = this.value; }">' +
             "<option value=''>#{ l(:label_jump_to_a_project) }</option>" +
```

On a very slow machine, a neutral page (say /admin/info) spends ~20% of his time building this select. On this machine this line takes about 350ms with the current version, 80ms with the proposed one. On a faster machine, it only saves a few dozens ms, but not something I benchmarked seriously (I can if needed). Of course all tests pass with both versions.

Comments welcome!

Related issues:

Related to Redmine - Patch #13301: Performance: avoid querying all membership...	**Closed**

History

#1 - 2012-09-13 11:20 - Etienne Massip

Rails does a select count() on #any? instead of a select top 1?

#2 - 2012-09-13 11:48 - Etienne Massip

Actually this has been fixed in Rails master but not merged in 3.2 branch.

See https://github.com/rails/rails/issues/3179.

#3 - 2012-09-13 21:14 - Jean-Philippe Lang

As discussed earlier today, memberships is already loaded by User#allowed_to? for non admin users. So this change improves response time for admin users but adds an extra query for non admin users.

Would be worth doing the change after doing some optimization in User#roles_for_project (eg. loading only the roles for the requested project).

#4 - 2013-02-27 01:17 - Jean-Baptiste Barth

- Status changed from New to Closed

- Assignee set to Jean-Baptiste Barth

I close this one as I think Jean-Philippe's argument is good, it might decrease performance for non admin users on small instances. I opened #13301 for a first step in optimizing User#roles_for_project.

Fyi, computing the project jump box was definitely taking too much time for my instance at work when the average staff user has 150+ projects. So I ended up caching this in a plugin and it works pretty well. The cache key used to cache this depends on various parameters, so as usual it won't be optimal for little installations (adding extra queries) or big installations with few projects but a lot of users. But it will save a lot of time in instances with a lot of people on a lot of projects. No general rule here, so no need to pollute the core with that.

#5 - 2013-02-27 11:05 - Etienne Massip

Switching to some AJAX-powered autocomplete input field rather than a simple combobox when the amount of items reaches a specific threshold (e.g. 20 items) could be a solution too.