Redmine - Defect #17202
Copying Project Fails to Copy Queries with visibility == VISIBILITY_ROLES

2014-06-16 21:09 - Zach Auclair

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Jean-Philippe Lang	**% Done:**	0%
Category:	Issues	**Estimated time:**	0.00 hour
Target version:	2.6.0		
Resolution:	Fixed	**Affected version:**	2.5.1

Description

Issue

As a result of #1019, app/models/project.rb#copy_queries fails to copy queries into a cloned project.

To Reproduce

Against redmine version 2.5.X:

1. Create a project
2. In created project, create a custom query (any fields) with visiblity == Developer for all projects
3. Clone the project to another project
4. Observe the query to be missing in the cloned project

Proposed Solution

Special-case the roles attribute:

```
diff --git a/app/models/project.rb b/app/models/project.rb
index 6ced345..395e4dd 100644
--- a/app/models/project.rb
+++ b/app/models/project.rb
@@ -950,6 +950,7 @@ class Project < ActiveRecord::Base
       new_query = IssueQuery.new
       new_query.attributes = query.attributes.dup.except("id", "project_id", "sort_criteria")
       new_query.sort_criteria = query.sort_criteria if query.sort_criteria
+      new_query.roles = query.roles if query.roles && query.visibility == IssueQuery::VISIBILITY_ROLES
       new_query.project = self
       new_query.user_id = query.user_id
       self.queries << new_query
```

Environment

```
Environment:
  Redmine version             2.5.1.devel
  Ruby version                1.8.7-p352 (2011-06-30) [x86_64-linux]
  Rails version               3.2.18
  Environment                 production
  Database adapter            SQLite
SCM:
  Subversion                  1.6.11
  Mercurial                   2.6.3
  Git                         1.8.2.1
  Filesystem
Redmine plugins:
  no plugin installed
```

Associated revisions

Revision 13329 - 2014-07-14 18:20 - Jean-Philippe Lang

Fixed that query roles are not copied when copying a project (#17202).

History

#1 - 2014-06-16 21:15 - Zach Auclair

In my "Issue" section, I meant to say that it fails to copy issue queries that have visibility == VISIBILITY_ROLES because they fail to validate (due to the roles not being copied into the new IssueQuery object in said copy_queries function).

#2 - 2014-07-01 17:13 - Zach Auclair

Also as a result of #1019, queries are not properly editable (non admin users shouldn't be able to make public queries that apply to all projects) nor are they validated correctly when being sent back to the server.

I have a patch that I can upload which fixes both of these problems if you are interested.

#3 - 2014-07-14 18:21 - Jean-Philippe Lang

- Status changed from New to Resolved
- Assignee set to Jean-Philippe Lang
- Target version set to 2.5.3
- Resolution set to Fixed

Fix committed in r13329, thanks for pointing this out.

#4 - 2014-07-16 15:16 - Zach Auclair

Hi Jean-Philippe; as I mentioned in #note 2, there is a larger problem with the patch in that it allows unauthorized users to make global queries. Along with this, the feature allows users to create queries that they then cannot manage.

Here are some of the required patches which incorporate more frontend / backend logic and a state-change table.

patch 1

Users with :manage_public_queries should be able to view public queries.

```
diff --git a/app/models/issue_query.rb b/app/models/issue_query.rb
index 7d70529..9b85665 100644
--- a/app/models/issue_query.rb
+++ b/app/models/issue_query.rb
@@ -73,7 +73,7 @@ class IssueQuery < Query

   # Returns true if the query is visible to +user+ or the current user.
   def visible?(user=User.current)
-    return true if user.admin?
+    return true if user.admin? || (project && user.allowed_to?(:manage_public_queries, project))
     return false unless project.nil? || user.allowed_to?(:view_issues, project)
     case visibility
```

```
      when VISIBILITY_PUBLIC
```

patch 2

"Correct" the transitions issue queries are allowed to make.

This boils down to public <-> private and global <-> local
(and is somewhat complicated by admin perm, manage_public_queries perm,
is project new, etc). In the comments of this commit, there is a table
which represents the valid transtions.

```
index 049e353..9ec339b 100644
--- a/app/controllers/queries_controller.rb
+++ b/app/controllers/queries_controller.rb
@@ -54,8 +54,21 @@ class QueriesController < ApplicationController
   def create
     @query = IssueQuery.new(params[:query])
     @query.user = User.current
+    if @query.is_public?
+      if params[:query_is_for_all]
+        unless User.current.admin?
+          flash[:error] = l(:error_query_admin_privs)
+          return render :action => 'new', :layout => !request.xhr?
+        end
+      else
+        unless User.current.allowed_to?(:manage_public_queries, @project)
+          flash[:error] = l(:error_query_local_privs)
+          return render :action => 'new', :layout => !request.xhr?
+        end
+      end
+    end
+
     @query.project = params[:query_is_for_all] ? nil : @project
-    @query.visibility = IssueQuery::VISIBILITY_PRIVATE unless User.current.allowed_to?(:manage_public_queries, @project) ||
User.current.admin?
     @query.build_from_params(params)
     @query.column_names = nil if params[:default_columns]

@@ -71,12 +84,12 @@ class QueriesController < ApplicationController
   end

   def update
+    previous_query = @query.dup
     @query.attributes = params[:query]
     @query.project = nil if params[:query_is_for_all]
-    @query.visibility = IssueQuery::VISIBILITY_PRIVATE unless User.current.allowed_to?(:manage_public_queries, @project) ||
User.current.admin?
     @query.build_from_params(params)
     @query.column_names = nil if params[:default_columns]
-
+    return unless check_visibility_change(previous_query)
     if @query.save
```

```
      flash[:notice] = l(:notice_successful_update)
      redirect_to_issues(:query_id => @query)
@@ -85,6 +98,34 @@ class QueriesController < ApplicationController
     end
   end

+  # this is the table that governs query changes.
+  # for more information on the definition consult querys/_form.html.erb
+  # (which has the same table in javascript
+  def check_visibility_change(previous_query)
+    x = false
+    v = true
+    p = User.current.allowed_to?(:manage_public_queries, @project) || User.current.admin?
+    a = User.current.admin?
+    from_pu = previous_query.is_public?
+    pu = @query.is_public?
+    existing_state_change = [
+      [p, p, a, p],
+      [p, v, a, v],
+      [x, x, a, a],
+      [x, x, a, v]
+    ]
+    xIndex = !@query.project.nil? ? (pu ? 0 : 1) : (pu ? 2 : 3)
+    yIndex = !previous_query.project.nil? ? (from_pu ? 0 : 1) : (from_pu ? 2 : 3)
+    state_change_allowed = existing_state_change[yIndex][xIndex]
+    unless state_change_allowed
+      flash[:error] = @query.project.nil? ?
+        (@query.is_public? ? l(:error_query_admin_privs) : l(:error_query_state_disallowed) ) :
+        (@query.is_public? ? l(:error_query_local_privs) : l(:error_query_state_disallowed) )
+      render :action => 'edit'
+    end
+    state_change_allowed
+  end
+
   def destroy
     @query.destroy
     redirect_to_issues(:set_filter => 1)
diff --git a/app/views/queries/_form.html.erb b/app/views/queries/_form.html.erb
index c9f710a..23fff23 100644
--- a/app/views/queries/_form.html.erb
+++ b/app/views/queries/_form.html.erb
@@ -7,7 +7,6 @@
 <p><label for="query_name"><%=l(:field_name)%></label>
 <%= text_field 'query', 'name', :size => 80 %></p>

-<% if User.current.admin? || User.current.allowed_to?(:manage_public_queries, @project) %>
 <p><label><%=l(:field_visible)%></label>
   <label class="block"><%= radio_button 'query', 'visibility', Query::VISIBILITY_PRIVATE %> <%= l(:label_visibility_private) %></label>
   <label class="block"><%= radio_button 'query', 'visibility', Query::VISIBILITY_ROLES %> <%= l(:label_visibility_roles) %>:</label>
@@ -17,11 +16,9 @@
   <label class="block"><%= radio_button 'query', 'visibility', Query::VISIBILITY_PUBLIC %> <%= l(:label_visibility_public) %></label>
   <%= hidden_field_tag 'query[role_ids][]', '' %>
 </p>
```

```
-<% end %>

 <p><label for="query_is_for_all"><%=l(:field_is_for_all)%></label>
-<%= check_box_tag 'query_is_for_all', 1, @query.project.nil?,
-      :disabled => (!@query.new_record? && (@query.project.nil? || (@query.is_public? && !User.current.admin?))) %></p>
+<%= check_box_tag 'query_is_for_all', 1, @query.project.nil? %></p>

 <fieldset><legend><%= l(:label_options) %></legend>
 <p><label for="query_default_columns"><%=l(:label_default_columns)%></label>
@@ -72,10 +69,128 @@
 </div>

 <%= javascript_tag do %>
-$(document).ready(function(){
-  $("input[name='query[visibility]']").change(function(){
-    var checked = $('#query_visibility_1').is(':checked');
-    $("input[name='query[role_ids][]'][type=checkbox]").attr('disabled', !checked);
-  }).trigger('change');
-});
+  var roles = new function(){
+    this.canManagePublicQueries = <%= User.current.allowed_to?(:manage_public_queries, @project) %>;
+    this.isAdmin = <%= User.current.admin? %>;
+  };
+  var visibility = new function() {
+    // state
+    this.initialGlobal = <%= @query.project.nil? %>;
+    this.initialLocal = !this.initialGlobal;
+    this.isNew = <%= @query.new_record? %>;
+    this.isGlobal = this.initialGlobal;
+    this.isLocal = !this.isGlobal;
+    this.initialVisibility = <%= @query.visibility %>;
+    this.visibility = this.initialVisibility;
+
+    // selectors
+    this.visIdPrefix = "#query_visibility_";
+    this.allProjectId = "#query_is_for_all";
+    this.allVisSelector = "input[name='query[visibility]']";
+    this.allRoleSelector = "input[name='query[role_ids][]'][type=checkbox]";
+
+    // name -> id lookup
+    var visLookup = {
+      public:  <%= Query::VISIBILITY_PUBLIC %>,
+      private: <%= Query::VISIBILITY_PRIVATE %>,
+      roles:   <%= Query::VISIBILITY_ROLES %>
+    };
+    this.visSelectorFor = function(name){
+      var vis = this.visibilityByName(name);
+      if(vis === undefined) return;
+      return this.visIdPrefix+vis;
+    };
+    this.toggleProject = function(){
+      this.isGlobal = $(this.allProjectId).is(":checked");
+      this.isLocal = !this.isGlobal;
```

```
+  };
+  // return the ID of the selected visibility
+  this.selectedVis = function(){
+    var visRadio = $(this.allVisSelector + ":checked");
+    if(visRadio.length == 0) return;
+    this.visibility = parseInt(visRadio.val());
+    return this.visibility;
+  };
+  // return the name of the visibility with they given id
+  this.visibilityById = function(id){
+    if(id == null) return;
+    for(var vis in visLookup){
+      if(visLookup[vis] === id) return vis;
+    }
+  };
+  this.drawRoles = function(selectedVis){
+    var roleVisSelected = selectedVis === this.visibilityByName('roles');
+    $(this.allRoleSelector).prop('disabled', !roleVisSelected);
+  };
+  this.disableVisibility = function(disable){
+    $(this.allVisSelector).prop("disabled", disable);
+  };
+  this.onChange = function(){
+    var selectedVis = this.selectedVis();
+    if(selectedVis === undefined) return;
+    this.toggleProject();
+    this.drawRoles(selectedVis);
+    /*
+      This is the table that governs query changes.
+      The upper left coordinate is 0,0 and the lower right, 3,3.
+     p = can manage public
+     a = admin
+     v = all
+     x = disallowed
+                        [to]
+                  local  | global
+                  pu  pr | pu pr
+                  #   #  | #  #
+       local  pu # p   p  | a  p
+              pr # p   v  | a  v
+    [from]       ---------|--------
+       global pu # x   x  | a  a
+              pr # x   x  | a  v
+    */
+    var x = false;
+    var v = true;
+    var p = roles.canManagePublicQueries || roles.isAdmin;
+    var a = roles.isAdmin;
+    var from_pu = this.visibilityByName('private') !== this.initialVisibility;
+    var pu = this.visibilityByName('private') !== this.visibility;
+    var existingStateChange = [
+      [p, p, a, p],
+      [p, v, a, v],
```

```
+        [x, x, a, a],
+        [x, x||this.isNew, a, v]
+      ];
+      var xIndex = this.isLocal ? (pu ? 0 : 1) : (pu ? 2 : 3);
+      var yIndex = this.initialLocal ? (from_pu ? 0 : 1) : (from_pu ? 2 : 3);
+      var allowedTransition = existingStateChange[yIndex][xIndex];
+      // based on state diagram, is our transition allowed?
+      if(!allowedTransition){
+        $(this.allProjectId).prop("checked", this.initialGlobal);
+        $(this.visIdPrefix + this.initialVisibility).prop("checked", true);
+        this.toggleProject();
+        this.drawRoles(this.selectedVis());
+      }
+      // set up the ui for the next transitions
+      this.disableVisibility(false);
+      var allowAllProjects = false;
+      if(!p){
+        $(this.visSelectorFor("roles") + ", " + this.visSelectorFor("public")).prop("disabled",true);
+      }
+      if((this.isNew || !this.initialGlobal) && (this.visibilityByName("private") === this.selectedVis() || roles.isAdmin)){
+        allowAllProjects = true;
+      }
+      if(allowAllProjects && this.isGlobal && !roles.isAdmin){
+        $(this.visSelectorFor("roles") + ", " + this.visSelectorFor("public")).prop("disabled",true);
+      }
+      $(this.allProjectId).prop("disabled", !allowAllProjects);
+    };
+    // return the id of the visibility with the given name
+    this.visibilityByName = function(name){
+      if(name == null || !(name in visLookup)) return;
+      return visLookup[name];
+    };
+  };
+  $(document).ready(function(){
+    $(visibility.allProjectId).change(function(){ visibility.onChange() });
+    $(visibility.allVisSelector).change(function(){ visibility.onChange() }).trigger('change');
+  });
 <% end %>
diff --git a/config/locales/en.yml b/config/locales/en.yml
index fc5072c..992bb75 100644
--- a/config/locales/en.yml
+++ b/config/locales/en.yml
@@ -182,7 +182,9 @@ en:
   notice_account_deleted: "Your account has been permanently deleted."
   notice_user_successful_create: "User %{id} created."
   notice_new_password_must_be_different: The new password must be different from the current password
-
+  error_query_state_disallowed: "That state change is not allowed."
+  error_query_admin_privs: "Only admins are allowed to edit public queries that apply to all projects."
+  error_query_local_privs: "Only users with the manage_public_queries role are allowed to edit public queries on individual projects."
   error_can_t_load_default_data: "Default configuration could not be loaded: %{value}"
   error_scm_not_found: "The entry or revision was not found in the repository."
   error_scm_command_failed: "An error occurred when trying to access the repository: %{value}"
```

#5 - 2014-09-14 11:19 - Jean-Philippe Lang

- Status changed from Resolved to Closed

- Target version changed from 2.5.3 to 2.6.0

Thanks for your feedback, please open a new issue for your additional fixes and try to include tests in your patch.