Finding forums based on a user's permissions

Asked 15 years, 9 months ago Modified 15 years, 1 month ago Viewed 301 times

2

I'm implementing a forum system called rBoard. Code is viewable at http://github.com/radar/rboard. I've reached an impasse with the permissions code that I've been trying to implement and I've decided to turn to the all-knowing, ever-caring Stack Overflow to solve this issue.

Relevant information is thusly:

Category model

```
class Category < ActiveRecord::Base
  has_many :permissions
  has_many :groups, :through => :permissions
  has_many :forums
end
```

Forum model

```
class Forum < ActiveRecord::Base
  has_many :permissions
  has_many :groups, :through => :permissions
  belongs_to :category
end
```

Group model

```
class Group < ActiveRecord::Base
  has_many :group_users
  has_many :users, :through => :group_users
  belongs_to :owner, :class_name => "User"
end
```

Permission model

```
class Permission < ActiveRecord::Base
  belongs_to :forum
  belongs_to :category
  belongs_to :group
end
```

User model

```ruby
class User < ActiveRecord::Base
  include Rboard::Permissions
  has_many :group_users
  # Note: I am using nested_has_many_through
  has_many :groups, :through => :group_users
  has_many :permissions, :through => :groups
end
```

Permissions module

```ruby
module Rboard::Permissions
  THINGS = ['forum', 'category']

  def self.included(klass)

    klass.class_eval do
      # Here we can pass an object to check if the user or any the user's
groups
      # has permissions on that particular option.
      def overall_permissions(thing = nil)
        conditions = if thing.nil?
          THINGS.map do |t|
            "permissions.#{t}_id " + (thing.nil? ? " IS NULL" : "= #
{thing.id}") + " OR permissions.#{t}_id IS NULL"
          end.join(" AND ")
        else
          association = thing.class.to_s.downcase
          "permissions.#{association}_id = #{thing.id} OR permissions.#
{association}_id IS NULL"
        end

        permissions.all(:conditions => conditions)
      end

      def can?(action, thing = nil)
        permissions = overall_permissions(thing)
        !!permissions.detect { |p| p.send("can_#{action}") }
      end
    end
  end
end
```

Hopefully with this you should be able to figure out the fields in the permissions table are like `can_see_forum` and so on. Extra fields are `forum_id`, `category_id` and `default` (default is currently unused)

What I want to know is, how can I find all the forums a group can see? Generally if a forum_id is set, then that permission applies. If there is only one permission for that group without specifying a forum_id or category_id then it is seen to be global to everything. I'm completely at a loss here.

ruby-on-rails **ruby**

Share

Improve this question

Follow

edited Nov 3, 2009 at 4:01

monksy

14.2k • 18 • 77 • 128

asked Mar 27, 2009 at 9:12

Ryan Bigg

108k • 25 • 241 • 262

1 Answer

Sorted by: Highest score (default)

0

It looks like you need something like (the fictitious) acts_as_permissible. Some mixin that can be applied to different kinds of objects -- in this case groups and users -- that allows you to test authorization. A possible usage might be:

```
class Group
  include Acts::Permissible
  acts_as_permissible
end

module Acts
  module Permissible
    def self.acts_as_permissible
      begin
        Role.find(:all).each do |role|
          define_method "can_#{role.access_type}?" do
            self.send('has_role?', role.access_type)
          end
        end
      # Since we're possibly running within the scope of Rake, handle the case
      # where the roles table doesn't exist
      rescue ActiveRecord::StatementInvalid => e
        RAILS_DEFAULT_LOGGER.error "Statement invalid while adding Role methods
to User. Is the Roles table present in the DB?\n" + e.inspect
      end
    end
  end
end
```

WARNING: This is *air code*! Never tested. But you could mix something like this into your User and authorize against the same roles as your Group model.

Share Improve this answer Follow

answered Mar 28, 2009 at 1:39

Steve

73 • 3