cancan - How to

Table of Contents

Resources

Video Tutorial: 2009 Original Video by the developer

Code in Github: CanCanCommunity/CanCanCan

Three Places to Control Access

- abilities.rb - defines which roles/groups can perform which actions in a controller
- controllers - add `load_and_authorize_resource` to apply access controls to each action in the controller
- views - before links and other actions, check whether the user can perform those actions for the controller (e.g. `<% if can? :update, @article %>` then show the Edit button)

Defining abilities.rb

wiki doc

```
class Ability
  include CanCan::Ability

  def initialize(user)
    user ||= User.new # guest user

    if user.role? :admin
      can :manage, :all
    else
      can :read, :all
      can :create, Comment  # for CommentController
      can :update, Comment do |comment|
        comment.try(:user) == user || user.role?(:moderator)  # This syntax is used to allow users to only
update their own comments, unless they are the all-mighty-powerful moderator.
      end
      cannot :delete, Comment
      if user.role?(:author)
        can :create, Article  # for ArticleController
        can :update, Article do |article|
          article.try(:user) == user
        end
      end
    end
  end
end
```

New syntax allows for abilities and classes to be defined in an array...

```
can [:read, :create, :update], Comment
can :read, [Comment, Article]
can [:read, :create], [Comment, Article]
can :manage, Article   # allows user to perform all actions in the ArticleController
```

Adding access controls to the Controller

wiki doc

```
load_and_authorize_resource

# comments_controller.rb possibility -- This is showing what you would put in a comments controller
# that is nested under an article controller.  Or you can keep them separate and they both just use
# load_and_authorize_resource.
# load_and_authorize_resource :nested => :article
```

Adding access controls to a View

wiki doc

```
<p>
  <% if can? :update, @article %>
    <%= link_to "Edit", edit_article_path(@article) %> |
  <% end %>
  <% if can? :destroy, @article %>
    <%= link_to "Destroy", @article, :method => :delete, :confirm => "Are you sure?" %> |
  <% end %>
  <%= link_to "Back to Articles", articles_path %>
</p>
...
<p>
  <% if can? :update, comment %>
    <%= link_to "Edit", edit_comment_path(comment) %>
  <% end %>
  <% if can? :destroy, comment %>
    | <%= link_to "Destroy", comment, :method => :delete, :confirm => "Are you sure?" %>
  <% end %>
</p>
```

Distributed Ability Definitions

Ex. Oligarchical saas with cancan

projecthydra-labs/hydra-grouper - reworks groups and abilities -- Not sure why they make it so complex.