How to program user preferences

Asked 16 years, 3 months ago Modified 16 years, 2 months ago

Viewed 965 times

3

I'm using Ruby on Rails for an internal site. Different users of the site have access to a wide variety of data and highly disparate perspectives of the data. Within those different classes of users, there needs to be levels of access. Within the levels of access I need to be able to add features from other classes of users.

In the released "Version 1.0" of the intranet site I have implemented the general classes of users. I am now needed to implement much finer-grained control of a users access.

The question is how?

What is the generally accepted practice for coding up user preferences (display the map (or not); access to this feature, but not this feature) without exploding the database schema and populating the view code with <% if feature_allowed %> tags everywhere.

ruby-on-rails ruby user-controls user-interface

Share

Improve this question

edited Aug 28, 2008 at 18:23

Follow

asked Aug 28, 2008 at 18:12

salt.racer

22.3k ● 14 ● 45 ● 51

2 Answers

Sorted by: Highest score (default)

3

Another totally different approach would be to use acts_as_authenticated and authorization plugins. The tables will be built by the plugins (ie **users**, **roles** and **roles**_users). From the doc:

> The authorization plugin provides the following:

- A simple way of checking authorization at either the class or instance method level using #permit and #permit?
- Authorization using roles for the entire application, a model class, or an instance of a model (i.e., a particular object).
- Some english-like dynamic methods that draw on the defined roles. You will be able to use methods like "user.is_fan_of angelina" or "angelina.has_fans?", where a 'fan' is only defined in the roles table.
- Pick-and-choose a mixin for your desired level of database complexity. For all the features, you will want to use "object roles table" (see below)

Share Improve this answer

Follow

answered Aug 28, 2008 at 20:57

Christian Lescuyer

19.2k ● 6 ● 50 ● 45

2

> populating the view code with <% if feature_allowed %> tags everywhere.

I don't think you want to do that. Assuming none of the alternatives suggested are practicable, at the very least you should consider shifting those checks into your controllers, where you can refactor them into a before_filter.

See section 11.3 in "Agile Web Development With Rails" (page 158 in my copy of the 2nd edition) where they do exactly that.

Share Improve this answer

Follow

answered Aug 29, 2008 at 9:41

Mike Woodhouse

52.3k ● 12 ● 89 ● 127