Ryan's Scraps

This doesn't involve you, ke

What's New in Edge Rails: Has Finder Functionality

Posted by **ryan** at 11:37 AM on Monday, March 24, 2008

It looks like Nick Kallen's wildly popular has_finder plugin will be making its way into Rails 2.x in the form of `named_scope`. Observe:

```
class User < ActiveRecord::Base
  named_scope :active, :conditions => {:active => true}
  named_scope :inactive, :conditions => {:active => false}
  named_scope :recent, lambda { { :conditions => ['created_at > ?', 1.week.ago] } }
end

# Standard usage
User.active    # same as User.find(:all, :conditions => {:active => true})
User.inactive # same as User.find(:all, :conditions => {:active => false})
User.recent   # same as User.find(:all, :conditions => ['created_at > ?', 1.week.ago])

# They're nest-able too!
User.active.recent
  # same as:
  # User.with_scope(:conditions => {:active => true}) do
  #   User.find(:all, :conditions => ['created_at > ?', 1.week.ago])
  # end
```

All the goodness you've come to love in `has_finder` is now available as `named_scope` - plus you get some extra goodies too. `User.all` is given to you for free as an alias for `User.find(:all)`.

Advanced

For those with more discriminating needs, don't forget some of these `has_finder` tidbits:

Passing Arguments

Pass in arguments to your named scopes to specify conditions (or other props) at run-time.

```
class User < ActiveRecord::Base
  named_scope :registered, lambda { |time_ago| { :conditions => ['created_at > ?', time_ago] }
end

User.registered 7.days.ago # same as User.find(:all, :conditions => ['created_at > ?', 7.days.ago])
```

Named Scope Extensions

Extend named scopes (in a similar fashion to association extensions).

```
class User < ActiveRecord::Base
  named_scope :inactive, :conditions => {:active => false} do
    def activate
      each { |i| i.update_attribute(:active, true) }
    end
  end
end

# Re-activate all inactive users
User.inactive.activate
```

Anonymous Scopes

You can also pass around scopes as first class objects using `scoped` (a named scoped provided to you for free) as a way

to build hairy queries on the fly.

```
1 # Store named scopes
2 active = User.scoped(:conditions => {:active => true})
3 recent = User.scoped(:conditions => ['created_at > ?', 7.days.ago])
4
5 # Which can be combined
6 recent_active = recent.active
7
8 # And operated upon
9 recent_active.each { |u| ... }
```

`named_scope` is a truly great feature. If you haven't started using it yet, do so. You won't know how you lived without it. Major thanks goes out to Nick.

tags: ruby, rubyonrails

Tags: Edge Rails
Meta: permalink

Comments

Leave a response

Ben – March 24, 2008 @ 12:28 PM

I've been waiting for this to get integrated – it's fantastic, and will be a huge boon to Rails developers. Thanks for the update, Ryan!

Matt – March 24, 2008 @ 05:42 PM

This looks so incredibly useful. I'm relatively new to Rails and haven't had a chance to play with the has_finder plugin, so I can't wait to start using this.

Zargony – March 24, 2008 @ 06:55 PM

Great. One of the first things I'm doing, when creating a new Rails project, is to install the has_finder plugin. I'm happy to hear that it has been merged into Rails now.

Arie Kusuma Atmaja – March 26, 2008 @ 12:02 PM

Thanks for the great post, Ryan. Will there be such a convenient shortcut for count specifically and rails statistics methods generally as well?

ex. `User.count :conditions => { :active => true, :created_at.gt => 1.week.ago }`

ryan – March 27, 2008 @ 10:31 AM

Arie, yes – these methods are already supported just as they are with association extensions. E.g with my previous user example:

```
user.active.recent.count #=> 23
```