Getting on the Wireless

- Any of the quainX SSID
- open browser:
  - username: quest
  - password: wireless

Digging into Rails 3

Chris Nelson & Doug Alcorn

Friday, October 1, 2010

CN

This is a working tutorial about migrating from Rails 2.3 to Rails 3. We have a sample project you'll checkout from github and we'll walk through the upgrade process. Along the way we'll highlight gotchas and Rails 3 hotness.

Friday, October 1, 2010

CN

Friday, October 1, 2010

CN

Splogna, our humble brownfield project

Volunteer project

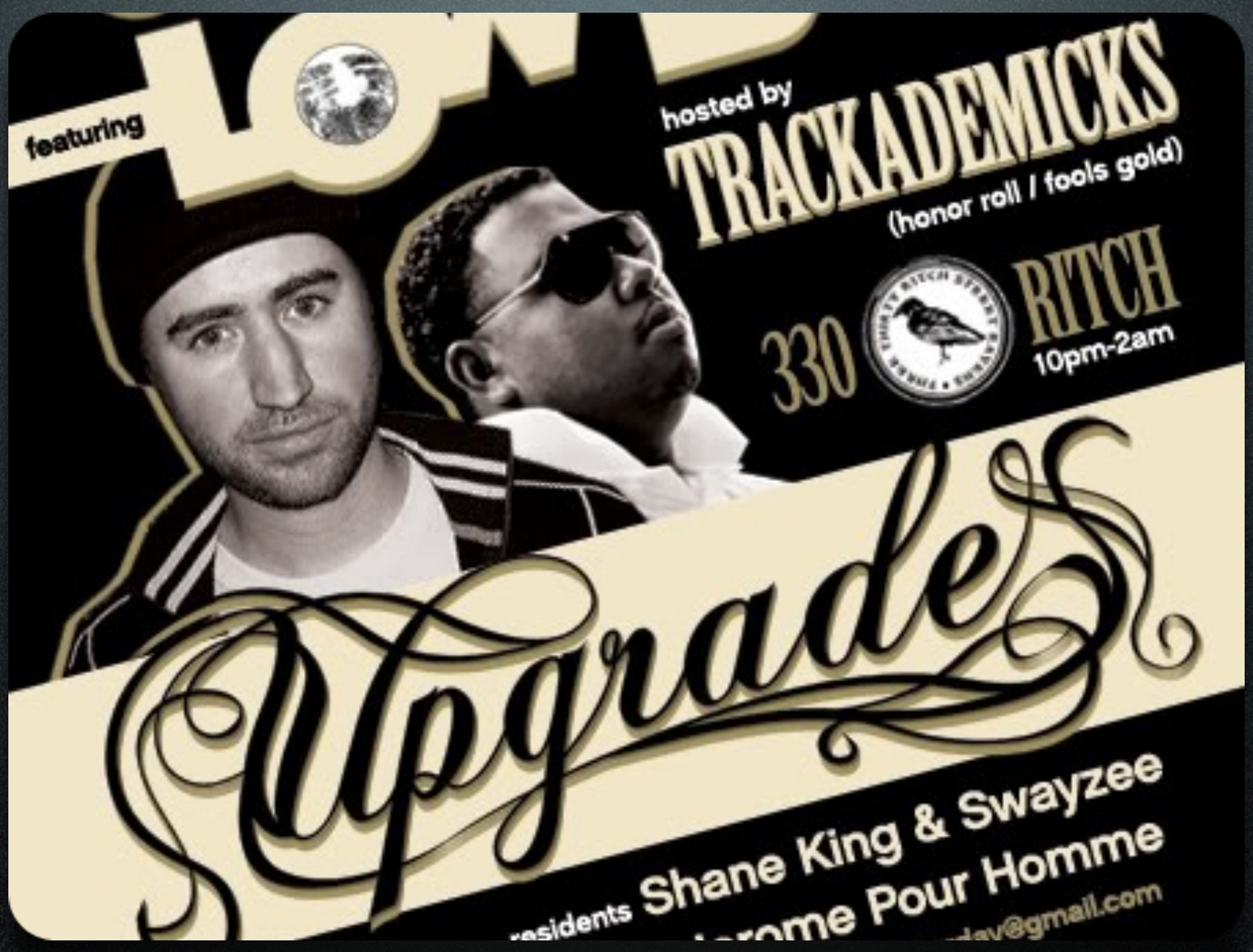
It's real(ish)

Tests pass

Non-huge

Rails 2.3.2

CN
run script/server and click around


featuring
hosted by
TRACKADEMICKS
(honor roll / fools gold)
330
RITCH
10pm-2am
Upgrade
Shane King & Swayzee
Pour Homme

Let’s do this!!

Clone from Github

git://github.com/

superchris/splogna

.git

CN
Gentlemen, start your terminals.

RVM Sandbox

$ rvm install jruby

$ rvm gemset create splogna

$ rvm use jruby@splogna

http://rvm.beginrescueend.com

Friday, October 1, 2010

DA

If you're not familiar with RVM, it's a great tool for unix-like operating systems to manage different ruby installs. Even better, it allows you to have multiple "gemsets" for each ruby. I've found it's nice for each project to have its own gemset. That way you know you won't be screwing up the environment for another project.

Installation is a simple shell one-liner on the project page.

Something Clever about Pik

http://github.com/

vertiginous/pik

DA

Not to be terribly out-classed, Windows developers can use pik similar to rvm. YMMV

Finishing Your Env

Install gems from rails2_start

rvm gemset import rails2_start.gems

Edit your database.yml

rake db:create:all

rake db:migrate

Tests should pass, server should start

CN

There's some slop that needs to be done after you clone to make sure you can actually run the app. It's worth pointing out we have a test suite and it all passes. That's our starting point and the point at which we know we're "done".

Slow gem install?

- ~/.gemrc:
  - gem: --no-rdoc --no-ri

Rails Upgrade Plugin

http://github.com/

rails/

rails_upgrade

.git

CN

./script/plugin install http://github....

A simple battery of scripts for upgrading Rails app/checking them for required updates.

rake -D upgrade

- rake rails:upgrade:backup
- rake rails:upgrade:check
- rake rails:upgrade:routes
- rake rails:upgrade:gems
- rake rails:upgrade:configuration

CN
Let's start with rails:upgrade:check
Then, do a rails:upgrade:backup
We'll start by doing a backup. This simply makes a *.rails2 backup of all your critical files so that you can be confident of easily getting back to where you were.

rake rails:upgrade:backup
Backs up your likely modified files so you can run the Rails 3 generator on your app with little risk

rake rails:upgrade:check
Runs a battery of checks on your Rails 2.x app and generates a report on required upgrades for Rails 3

rake rails:upgrade:configuration
Extracts your configuration code so you can create a new config/application.rb

rake rails:upgrade:gems
Generates a Gemfile for your Rails 3 app out of your config.gem directives

rake rails:upgrade:routes
Create a new, upgraded route file from your current routes.rb

Install Rails 3

- gem update rails

CN

alas script/* we hardly new ye...

Friday, October 1, 2010

CN

Rails 3 has eliminated the ./script/* individual scripts with a single rails command that takes subcommands (much like git, gem, apt-get/port, etc). Unlike Rails 2, the new version of the rails command takes --help, --version, and other std cmd line options.

Because we've backed up all the files we know will get modified, we're just going to regenerate the project and overwrite everything

run rails new .

CN
accept all the defaults DON’T RUN SERVER UNTIL AFTER THE BUNDLER TALK

Rails 3 and Bundler

Friday, October 1, 2010

DA
I was thinking of “Rails and Bundler, sitting in a tree, k-i-s-s-ing” and found this image. You can also think of Rails and Bundler rock’n it out! The image’s caption was, “Even rockstars take time out to kiss their mama.”

Bundler is a tool dedicated to managing project gem requirements. Gem handles dependencies, Bundler automates installing and making sure you have the right versions loaded. Rails uses Bundler to also require the gems appropriately.

Simple Gemfile

```
source 'http://rubygems.org'

gem 'rails', '3.0.0'

gem 'activerecord-jdbc-adapter'
gem 'activerecord-jdbcmysql-adapter'
```


DA

This allows you to specify the gems and versions you need for your project. You can group gems and then install the bundle --without that group as needed (such as '--without development test' when installing on production). Note that Rails will require the gems also based on these groups.

Bundler Commands

- bundle install
- bundle check
- bundle package
- bundle show

DA

bundle install

Install the gems specified by the Gemfile or Gemfile.lock

bundle check

Determine whether the requirements for your application are installed and available to bundler

bundle package

Package the .gem files required by your application into the vendor/cache directory

bundle show

Show the source location of a particular gem in the bundle

Bundler Git Awesomeness

```
gem "formtastic",
  :git => "git://github.com/justinfrench/formtastic.git",
  :branch => "rails3"
```

- gem list formtastic => ?
- $(BUNDLE_PATH)/bundler/gems/formtastic-#{git sha}

DA
standard practice of forking all gems you need?

Deploying with Bundler

- bundle package
- Gemfile.lock
- require ‘bundler/capistrano’
- cap --explain bundle:install

DA
use --local to install from vendor/cache

Bundler and warbler

- Mostly just works
  - Except when it doesn't
    - :git :(

Initial Gemfile

- git cherry-pick tags/gemfile
- gem install bundler
- bundle install
- start up the server with: rails s
- take a look at the index page

New Route Matching

```
Splogna::Application.routes.draw do

match '/login' => 'sessions#new', :as => :login

  match '/:controller(/:action(/:id))'

end
```

Namespacing Routes

- namespace “admin”
  - routes “/admin/foo” to Admin::Foo
- scope :module => “admin”
  - routes /foo to Admin::Foo
- scope “/admin”
  - routes /admin/foo to Foo

http://guides.rubyonrails.org/routeing.html

Nesting Resources

```
resources :posts do
 resources :comments
end
```

http://guides.rubyonrails.org/routeing.html

Additional RESTful Routes

```
resources :photos do
 member do
   get 'preview'
  end
  get 'search', :on => :collection
end
```

http://guides.rubyonrails.org/routeing.html

Routing Constraints

```
  match 'photos/:id' => 'photos#show', :id => /[A-Z]\d
  {5}/

match "photos", :constraints => {:subdomain => "admin"}
```

http://guides.rubyonrails.org/routeing.html

Custom Constraints

```
 class BlacklistConstraint
def initialize
  @ips = Blacklist.retrieve_ips
end

  def matches?(request)
    @ips.include?(request.remote_ip)
  end
end

TwitterClone::Application.routes.draw do
  match "*path" => "blacklist#index",
    :constraints => BlacklistConstraint.new
end
```

http://guides.rubyonrails.org/routeing.html

rails:upgrade:routes

- copy in the old config/routes.rb.rails2
- rake rails:upgrade:routes
- copy/paste result back into routes.rb
- rm public/index.html

upgrade failure

```
Splogna::Application.routes.draw do

match '/activate/:activation_code' => 'users#activate',
  :as => :activate, :activation_code =>

end
```

upgrade failure

```
Splogna::Application.routes.draw do

  match '/activate/:activation_code' => 'users#activate',
    :as => :activate

end
```

Let's See It

- start up the server with: rails s

Changes in Defaults

- application.rb is now application_controller.rb
- config/environment.rb => config/application.rb
- config.autoload_paths += %W(#{config.root}/lib)
- html escaped by default <%=h %> vs. <%= raw %>
- block view helpers now use <%= %>

DA
application.rb defined ApplicationController, but the filename didn't match.

there's a mix of stuff in lib/ => both foo_extension (which wouldn't get autoloaded properly) as well as code that should be autoloaded. There doesn't seem to be a way to differentiate between these two. Rails doesn't help you out here. Standard Ruby projects don't autoload lib? https://rails.lighthouseapp.com/projects/8994/tickets/5218-rails-3-rc-does-not-autoload-from-lib

Let's See It

- re-start server with: rails s

raw

- Rails escapes html by default
- This makes our tag_cloud sad :(
- Let's fix it!

Checking our Progress

- rake rails:upgrade:check
- rake test:units
- rm the lame test/unit/user_mailer_test.rb
- add :user_observer, :message_observer to app config

whack because that test uses the old, cruddy mailer testing style but doesn't actually test anything either.

ActionMailer

```
# app/mailers/message_mailer.rb
class MessageMailer < ActionMailer::Base
  default :from => "admin@splogna.com"

  def message_mail(message)
    @message = message
    mail(:to => message.to_user.email,
         :from => "admin@splogna.com",
         :subject => message.subject)
  end
end
```


DA

Greatly simplifies mailers. The ‘default’ method encapsulates the pattern of having a private method that is manually called to globally set mail headers. One call to ‘mail’ passing in hash keys for all mail headers also renders. All instance variables are visible to the mailer views just like controllers.

Mail::Message

TMail is dead! Long live Mail!

http://github.com

mikel/mail

DA
Same guy wrote both TMail and Mail.

“Built from my experience with TMail, it is designed to be a pure ruby implementation that makes generating, sending and parsing emails a no brainer.”

“Mail is a Ruby Email Library and addresses many of the shortcomings of TMail.”

Delivering Mail

```
class MessageObserver < ActiveRecord::Observer
  def after_create(message)
    MessageMailer.message_mail(message).deliver
  end
end
```


DA

Instead of using method missing to respond to deliver_ and friends invoke the deliver method on the mail message itself.

Testing ActionMailer

```ruby
# test/functional/message_mailer_test.rb
require 'test_helper'

class MessageMailerTest < ActionMailer::TestCase
  test "message_mail" do
    message = Factory(:message)
    mail = MessageMailer.message_mail(message)
    assert_equal [message.to_user.email], mail.to
    assert_equal message.subject, mail.subject
    assert_equal ["admin@splogna.com"], mail.from
    assert_match message.body, mail.body.encoded
  end
end
```


DA

Simply create a mail message by calling the method on the mailer class and then test it's attributes rather than having to deliver the mail and dealing with capturing the TMail objects and such.

Let's Generate Some Mailers

DA

Living on the Bleeding Edge

http://www.flickr.com/photos/madhava/68638279/

- Check if the gem has a rails3 branch
- Some gems have been forked for Rails 3
- Some gems aren't being supported
- RailsPlugins.org

DA

Rails 3 is out and stable. There are gems/plugins that aren't compatible yet. I expect many of the gems you use to work just fine or already be upgraded. Within a short while I expect the majority of actively maintained gems to be upgraded. In the mean-time...

Integration Testing with Capybara

- Webrat doesn't support Rails 3 (yet?)
- Capybara does!

CN

Capybara, the bigger rodent

- grey box testing
- lean pretty hard on css selectors
- writing your own DSL matchers/ macros
- Multiple drivers = As Real As You Wanna Be
- Let's see!

git cherry-pick capybara

tests pass!

- but let's clean up a little more
- git cherry-pick block_helpers

Rails Server

- click through to app to see where we are compared to the initial click through

CN

Arel

Relational Algebra in Ruby

Relational Algebra

- Mathematical model for relational data
  - C. F. Codd, of RDBMS fame
- Basic operators:
  - Set operators ($\cup$, $\cap$, $\subseteq$, etc)
  - Selection
  - Projection
  - Joins

Anything you can express in SQL, you can express in:

Relational Algebra

Anything you can express in SQL, you can express in:

Arel

Arel + Rails

- Replaces the guts of SQL building
- Leads to new awesome
- Only a subset of Arel is exposed
  - But fear not!
- Leads to lots of defe^H^Hprecation

CN

Less hashstasticalness

- find, first, all
- What once was a :key is now a method
- Methods all return a relation
  - Like a scope, with more awesome
- Combining relations is good!
  - Uncledaddy

Relation methods

- where(:conditions)
- having(:conditions)
- select
- group
- order
- limit
- offset
- includes

And they quack, too!

- find
- new
- create
- destroy(_all)
- update(_all)
- delete(_all)

Oh, and to_sql is your friend

CN

Knowledge Application Time

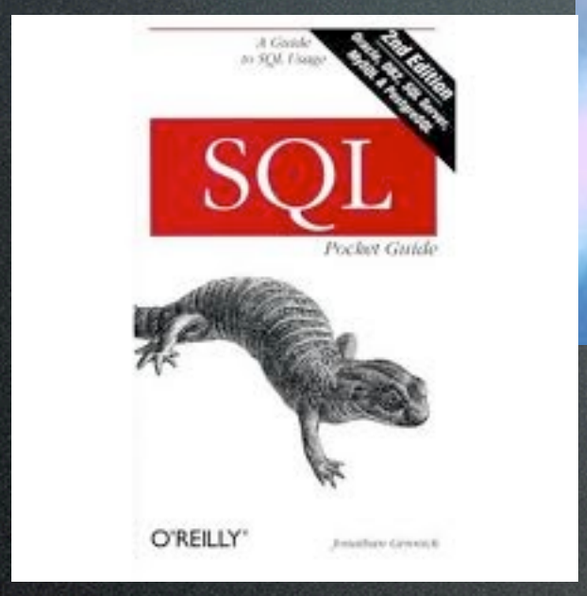
- Let's refactor our Need.search method

AREL: More awesome than meets the eye

- Hey, You said we can express any query in Arel
- But there's still a SQL string in my call to where()
- That's lame!

Arel unleashed

- Model.arel_table
  - returns an Arel::Table
  - arel_table[:foo] accesses attributes
  - attributes have predicate methods
    - eq, lt, gt, in, not_in
  - predicates can be combined (or, and)


A Guide to SQL Usage
2nd Edition
SQL
Pocket Guide
O'REILLY

Bonus refactoring

- Wouldn't it be cool if...
  - We could get rid of Need.arel_table
    - and just say Need[:description]
  - We could get rid of duplication with Got

The new respond_to

- Class method
- tell it what you respond_to
  - e. g. :html, :js, :json, :xml

Our new friend responds_with

- Tell it what you respond_with
  - e. g., @thing(s)
- Profit!

respond_with explained

- Determines what to render based on
  - format requested
  - HTTP verb
  - has_errors?

CN show the code?

Customizing respond_with

- :location
- old skool format blocks
- Responder class
  - implement ResponderClass.call
    - passed controller, resources, options

Let's play ping pong!

- git cherry-pick respond_with_test
- make it pass!
- then, let's clean up some other controllers too.

ActiveModel Compliance

```
require 'test_helper'

class PreferenceTest < ActiveModel::TestCase
  include ActiveModel::Lint::Tests

  setup do
    @model = Preference.new
  end
end
```

Necessary Methods

- model_name
- to_model
- persisted?
- to_param
- to_key
- valid?
- errors
- errors
  - []
  - full_messages

ActiveModel Modules

- AttributeMethods
- Callbacks
- Dirty
- Naming
- Translation
- Serialization
- Validations
- Observing
- StateMachine

Rails UJS

- give a hoot, don't pollute
  - your views with a piles o yucky javascript
- HTML5 data attributes to the rescue
- remote_form_for is dead
  - Long live :remote => true

CN

Let's get some!

- MessagesController#create needs some respond_to/with love
- Stick a :remote => true in app/views/messages/_form.html.erb
- Create app/views/mesages/create.js.erb

YayQuery!

- UJS is framework agnostic
- Let's plugin jQuery

Resources

- RailsCasts.com

Rendering Different Views for Mobile

DA