Ruby Explorations XII

Mark Keane...CSI...UCD

De End: Final Bits...

A: De Puppies

B: De Helpers

C: De Scaffolding

D: Putting De App on De Web...

E: De YAML

F: De Exam

A: De Puppies

Ruby & Rails

Puppies V1.0: Ruby & Rails

Rails on Ruby

Puppies 4.1.6

B: De Helpers

Putting more functionality into where...

Where's the Beef?

- Ruby functionality in the controllers seems quite simple; the complexity arises in co-ordinating between Model-View and Controller
- This may leave you slightly disappointed...its really not like writing Ruby programs
- You can have a lot more functionality, if you want it; if so, it tends to go into files in the Helpers Folder
- Many hidden app-level helpers exist (e.g., **link_to**)

link_to_*criteria* eg

- let's look at defining our own helpers, extending the way the existing **link_to** works, so that it takes an **id**
- we will also complex-up a controller, to show it doing a bit more than usual
- the application is called **clovis**, why ?
- it has a controller show (list method) and a model for Song (with name, artist, album, id, fields)

after doing....

$ rails new clovis

$ cd clovis

$ rails generate controller show list

$ rails generate model Song name:string artist:string album:strin

$ …..edit various files with stuff…

$ rake db:create

$ rake db:migrate

$ rake db:seed

$ rails server

use rails
console
to look at db
and what sort_by
does

clovis controller **show/list**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis] – .../app/controllers/show_controller.rb – JetBrains Ru

Development: clovis

Project /Users/user/Desktop/X_Teachin...

Project

- clovis (~/Desktop/X_Teaching/Ruby:2011-14/Lect
  - app
    - assets
    - controllers
      - concerns
        - application_controller.rb
        - show_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
  - bin
  - config
  - db
    - migrate
      - development.sqlite3
      - schema.rb
      - seeds.rb
      - test.sqlite3
  - lib
  - log
  - public
  - test
  - vendor
    - .gitignore
    - config.ru
    - Gemfile
    - Gemfile.lock

production.rb | 20141014191614_create_songs.rb | schema.rb | show_controller.rb | show_helper.rb | show/list.html.erb | seeds

```ruby
class ShowController < ApplicationController
  def list
  @criterion = params[:id] || "id"
  sort_method = case @criterion
    when "name",
         "artist",
       "album" then lambda {|song|song.send(@criterion).downcase}
    when "id" then lambda {|song| song.id}
  end
  @out = Song.all.sort_by &sort_method
  end

end
```

1: Project

7: Structure

6: TODO

clovis model **Song**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis] – .../db/schema.rb – JetBrains RubyMine 3.2.4

Development: clovis

Project /Users/user/Desktop/X_Teachin...

Project

- clovis (~/Desktop/X_Teaching/Ruby:2011-14/Lect
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
    - test.sqlite3
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock

show_controller.rb | show_helper.rb | show/list.html.erb | seeds.rb | production.rb | 20141014191614_create_songs.rb | schema.rb

```ruby
# encoding: UTF-8
# This file is auto-generated from the current state of the database. Instead
# of editing this file, please use the migrations feature of Active Record to
# incrementally modify your database, and then regenerate this schema definition.
#
# Note that this schema.rb definition is the authoritative source for your
# database schema. If you need to create the application database on another
# system, you should be using db:schema:load, not running all the migrations
# from scratch. The latter is a flawed and unsustainable approach (the more migrations
# you'll amass, the slower it'll run and the greater likelihood for issues).
#
# It's strongly recommended that you check this file into your version control system.

ActiveRecord::Schema.define(version: 20141014191614) do

  create_table "songs", force: true do |t|
    t.string   "name"
    t.string   "artist"
    t.string   "album"
    t.datetime "created_at"
    t.datetime "updated_at"
  end

end
```

:id created automatically

7: Structure

6: TODO

1:1 UTF-8

clovis **seeds**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis] – .../db/seeds.rb – JetBrains RubyMine 3.2.4

Development: clovis

Project /Users/user/Desktop/X_Teachin...

Project

- clovis (~/Desktop/X_Teaching/Ruby:2011-14/Lect
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
    - test.sqlite3
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock

production.rb | 20141014191614_create_songs.rb | schema.rb

show_controller.rb | show_helper.rb | show/list.html.erb | seeds.rb

```ruby
# This file should contain all the record creation needed to seed the database with its default values.
# The data can then be loaded with the rake db:seed (or created alongside the db with db:setup).
#
# Examples:
#
#   cities = City.create([{ name: 'Chicago' }, { name: 'Copenhagen' }])
#   Mayor.create(name: 'Emanuel', city: cities.first)


Song.create(:name => 'The End', :artist => 'Doors', :album => 'Greatest Hits')
Song.create(:name => 'Red Hot', :artist => 'RHCP', :album => 'What Hits')
Song.create(:name => 'Solid Air', :artist => 'John Martyn', :album => 'Solid Air')
Song.create(:name => 'May You Never', :artist => 'John Martyn', :album => 'Solid Air')
Song.create(:name => 'May You Never', :artist => 'Eric Clapton', :album => 'Clap Hits')
```

1: Project | 7: Structure | 6: TODO

14:13 | UTF-8

link_to_criterion

- if there is a method you what to use in several places, in a given view, you can define it in the helper file; every controller has its own automatically created ***controller_name*_helper.rb**
- Note, it is just a Ruby module
- in this eg we define a method that produces different flavours of a **link_to** method depending on the situation we are in

show **helper**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis] – .../app/helpers/show_helper.rb – JetBrains RubyMine 3.2.4

Development: clovis

Project /Users/user/Desktop/X_Teachin...

Project

- clovis (~/Desktop/X_Teaching/Ruby:2011-14/Lec
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
    - test.sqlite3
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock

production.rb | 20141014191614_create_songs.rb | schema.rb

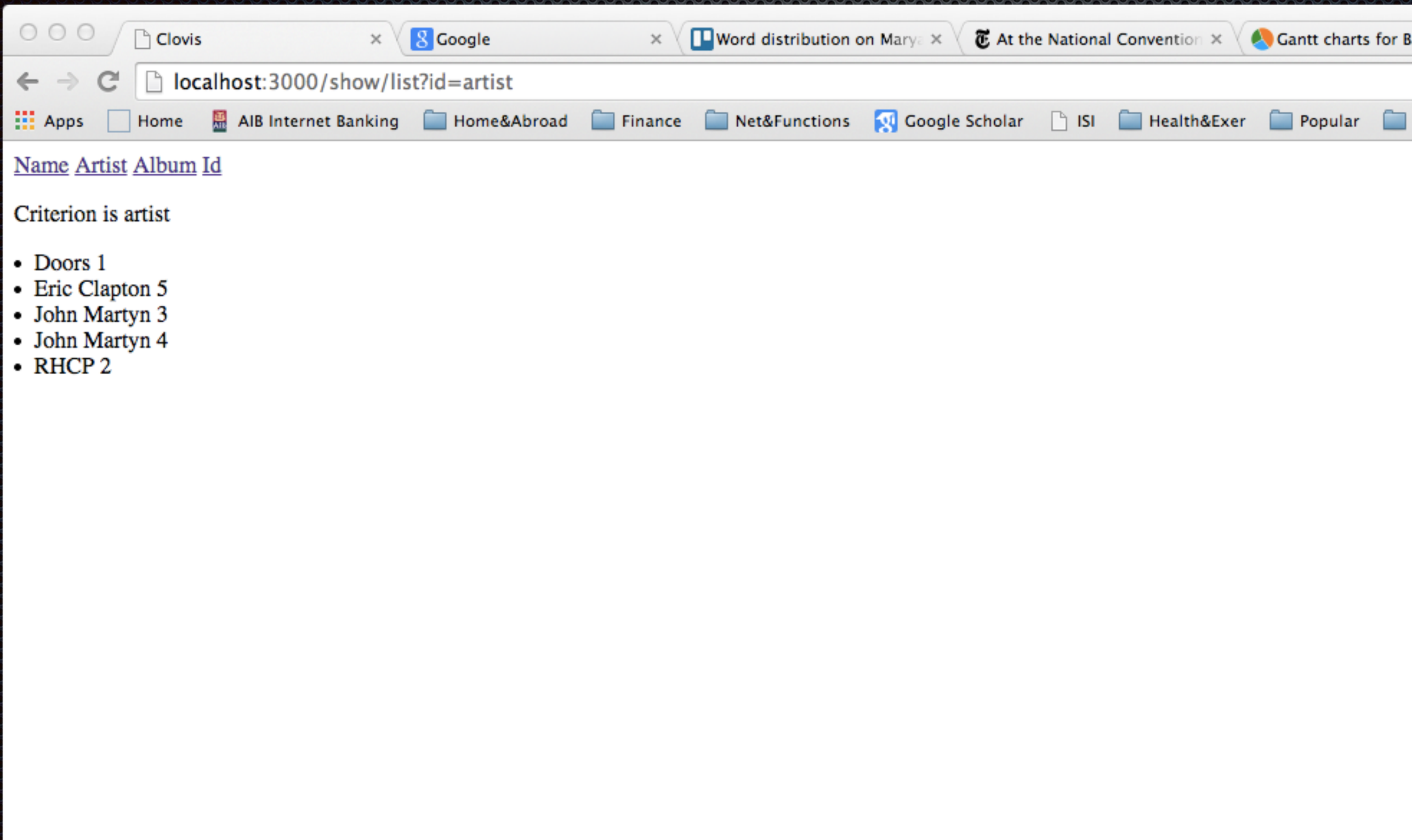
show_controller.rb | show_helper.rb | show/list.html.erb | seeds.rb

```ruby
module ShowHelper

  def link_to_criteria(criterion)
    link_to(criterion.capitalize,
      :controller => "show",
      :action  =>  "list",
      :id  => criterion)
  end
end
```

6: TODO

11:1 UTF-8

link_to screen for view



show/list view

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis] – .../app/views/show/list.html.erb – JetBrains RubyM

Development: clovis

Project /Users/user/Desktop/X_Teachin...

Project

- clovis (~/Desktop/X_Teaching/Ruby:2011-14/Lect
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
    - test.sqlite3
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock

production.rb | 20141014191614_create_songs.rb | schema.rb

show_controller.rb | show_helper.rb | show/list.html.erb | seeds.

```
<th class="show"><%= link_to_criteria("name") %></th>
<th class="show"><%= link_to_criteria("artist") %></th>
<th class="show"><%= link_to_criteria("album") %></th>
<th class="show"><%= link_to_criteria("id") %></th>

<p> Criterion is <%=  @criterion %> </p>

<% @out.each do |item| %>
        <li> <%=h item.send(@criterion) %> <%=h item.id %> </li>
<% end %>
```

1: Project

7: Structure

6: TODO

show **helper**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis] – .../app/helpers/show_helper.rb – JetBrains RubyMine 3.2.4

Development: clovis

Project /Users/user/Desktop/X_Teachin...

Project

- clovis (~/Desktop/X_Teaching/Ruby:2011-14/Lec
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
    - test.sqlite3
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock

production.rb | 20141014191614_create_songs.rb | schema.rb

show_controller.rb | show_helper.rb | show/list.html.erb | seeds.rb

```ruby
module ShowHelper

  def link_to_criteria(criterion)
    link_to(criterion.capitalize,
      :controller => "show",
      :action  =>  "list",
      :id  => criterion)
  end
end
```

6: TODO

11:1 UTF-8

clovis controller **show/list**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis] – .../app/controllers/show_controller.rb – JetBrains Ru

Development: clovis

Project /Users/user/Desktop/X_Teachin...

Project

- clovis (~/Desktop/X_Teaching/Ruby:2011-14/Lect
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
    - test.sqlite3
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock

production.rb | 20141014191614_create_songs.rb | schema.rb | seeds

show_controller.rb | show_helper.rb | show/list.html.erb

```
class ShowController < ApplicationController
  def list
  @criterion = params[:id] || "id"
  sort_method = case @criterion
    when "name",
         "artist",
       "album" then lambda {|song|song.send(@criterion).downcase}
    when "id" then lambda {|song| song.id}
  end
  @out = Song.all.sort_by &sort_method
  end

end
```

if :id is nil use “id”

using lambda

send turns string into method

case tests @criterion against string after when

1: Project

7: Structure

6: TODO

show/list view

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis] – .../app/views/show/list.html.erb – JetBrains RubyM

Development: clovis

Project /Users/user/Desktop/X_Teachin...

Project

- clovis (~/Desktop/X_Teaching/Ruby:2011-14/Lect
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
    - test.sqlite3
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock

production.rb | 20141014191614_create_songs.rb | schema.rb

show_controller.rb | show_helper.rb | show/list.html.erb | seeds.

```
<th class="show"><%= link_to_criteria("name") %></th>
<th class="show"><%= link_to_criteria("artist") %></th>
<th class="show"><%= link_to_criteria("album") %></th>
<th class="show"><%= link_to_criteria("id") %></th>

<p> Criterion is <%=  @criterion %> </p>

<% @out.each do |item| %>
        <li> <%=h item.send(@criterion) %> <%=h item.id %> </li>
<% end %>
```

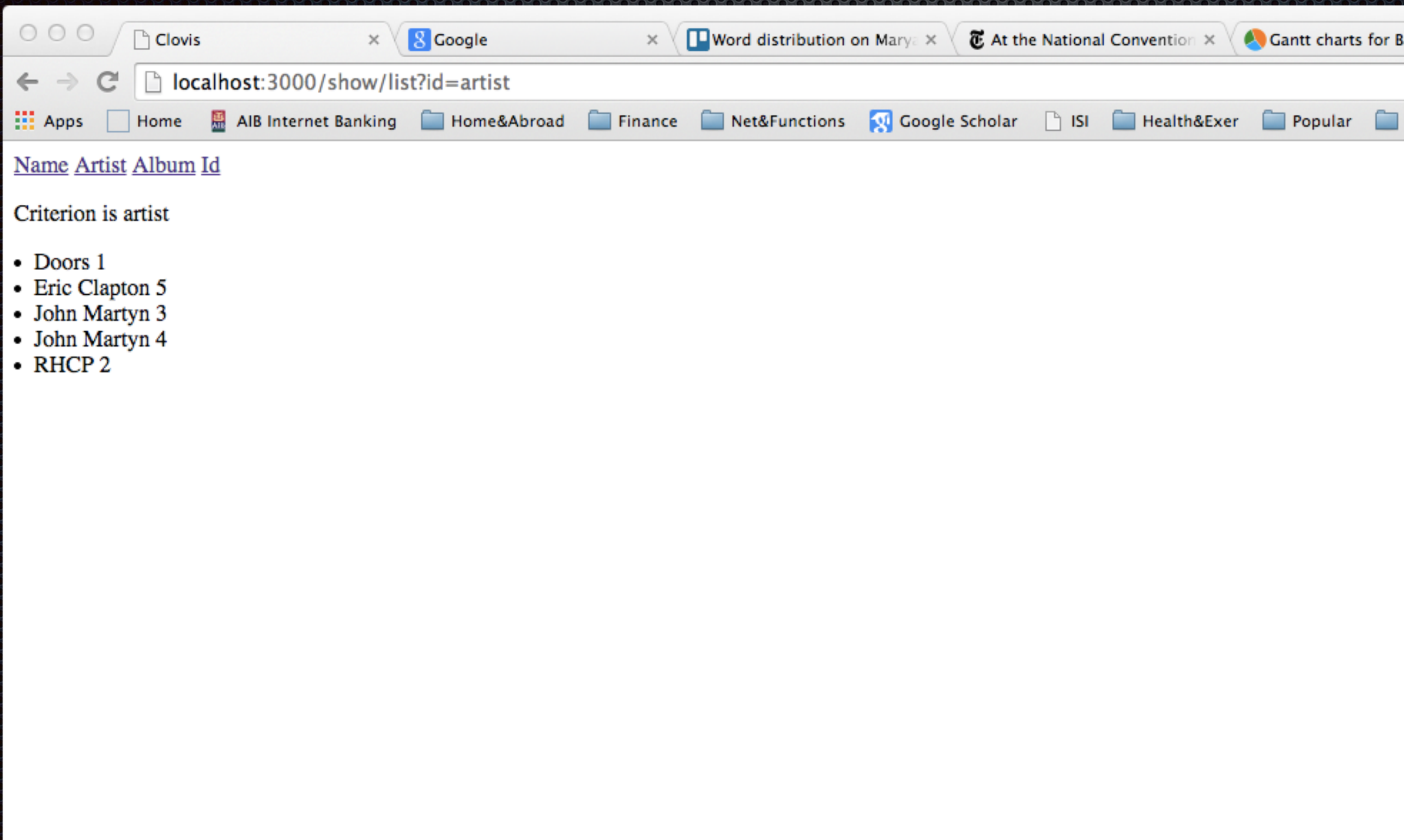
send invokes method on **item**

1: Project

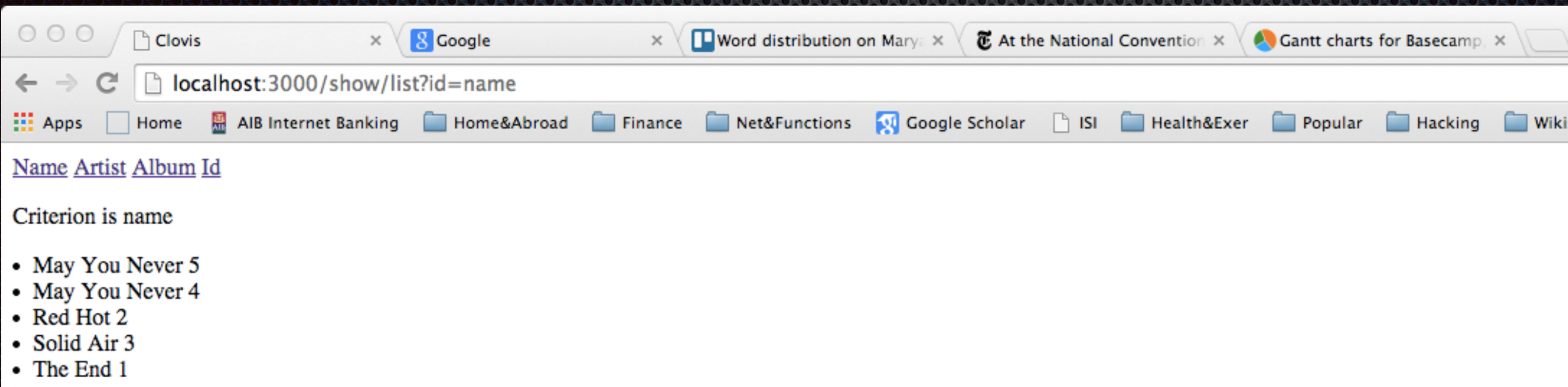
7: Structure

6: TODO

link_to screen for view



link_to screen for view



Where can you use it...

- methods defined in **show_helper.rb** seem to be accessible to any view-file via the method name called; if you define a method in another helper with the same name, it goes to the helper-view consistent file first *(see clovis2 and 3)*
- but, you can officially make this so by putting them in the **application_helper.rb** file
- nb we could have put the functionality of **link_to_criterion** in the view, but then it would be more than a view (sort of), using a helper is better encapsulation and cud aid maintenance

Test controller has tester

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis2] – .../app/controllers/test_controller.rb – JetBrains RubyMine 3.2.4

Development: clovis

Project /Users/user/Desktop/X_Teaching/Ruby:2011-14/Lects...

Project

- clovis2 [clovis] (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
      - test_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
      - test_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
      - test
        - tester.html.erb
  - bin
  - config
    - environments
    - initializers
    - locales
    - application.rb
    - boot.rb
    - database.yml
    - environment.rb
    - routes.rb
    - secrets.yml
  - db
    - migrate
      - 20141014191614_create_songs.rb
    - development.sqlite3
    - schema.rb
    - seeds.rb

show_controller.rb | show_helper.rb | show/list.html.erb | seeds.rb | test_helper.rb

test_controller.rb | test/tester.html.erb | production.rb | 20141014191614_create_songs.rb | schema.rb

```
class TestController < ApplicationController
  def tester
  @criterion = params[:id] || "id"
  sort_method = case @criterion
    when "name",
         "artist",
       "album" then lambda {|song|song.send(@criterion).downcase}
    when "id" then lambda {|song| song.id}
  end
  @out = Song.all.sort_by &sort_method
  end


end
```

(see clovis2)

1: Project | Z: Structure | Data Sources | 6: TODO

show **helper in clovis2**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis2] – .../app/helpers/show_helper.rb – JetBrains RubyMine 3.2.4

Development: clovis

Project /Users/user/Desktop/X_Teaching/Ruby:2011-14/Lects...

Project

- clovis2 [clovis] (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
      - test_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
      - test_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
      - test
        - tester.html.erb
  - bin
  - config
    - environments
    - initializers
    - locales
    - application.rb
    - boot.rb
    - database.yml
    - environment.rb
    - routes.rb
    - secrets.yml
  - db
    - migrate
      - 20141014191614_create_songs.rb
    - development.sqlite3
    - schema.rb
    - seeds.rb

test_controller.rb | test/tester.html.erb | production.rb | 20141014191614_create_songs.rb | schema.rb

show_controller.rb | show_helper.rb | show/list.html.erb | seeds.rb | test_helper.rb

```
module ShowHelper

  def link_to_criteria(criterion)
    link_to(criterion.capitalize,
      :controller => "show",
      :action  =>  "list",
      :id  => criterion)
  end
end
```

has **link_to_criteria** helper

has no **link_to_criteria** method, it is empty

has new screen that prints stuff

(see clovis2)

1: Project | 7: Structure | Data Sources | 6: TODO | 11:1 | UTF-8

test **helper in clovis2**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis2] – .../app/helpers/test_helper.rb – JetBrains RubyMine 3.2.4

Development: clovis

Project /Users/user/Desktop/X_Teaching/Ruby:2011-14/Lects...

Project

- clovis2 [clovis] (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
      - test_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
      - test_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
      - test
        - tester.html.erb
  - bin
  - config
    - environments
    - initializers
    - locales
    - application.rb
    - boot.rb
    - database.yml
    - environment.rb
    - routes.rb
    - secrets.yml
  - db
    - migrate
      - 20141014191614_create_songs.rb
    - development.sqlite3
    - schema.rb
    - seeds.rb

test_controller.rb | test/tester.html.erb | production.rb | 20141014191614_create_songs.rb | schema.rb
show_controller.rb | show_helper.rb | show/list.html.erb | seeds.rb | test_helper.rb

```
module TestHelper
    # Mammy, I feel a bit empty....
end
```

has **link_to_criteria** helper

has no **link_to_criteria** method, it is empty

has new screen that prints stuff

(see clovis2)

6: TODO

3:1 UTF-8

test **helper in clovis2**

clovis – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/clovis2] – .../app/views/test/tester.html.erb – JetBrains RubyMine 3.2.4

Development: clovis

Project /Users/user/Desktop/X_Teaching/Ruby:2011-14/Lects...

Project

- clovis2 [clovis] (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - show_controller.rb
      - test_controller.rb
    - helpers
      - application_helper.rb
      - show_helper.rb
      - test_helper.rb
    - mailers
    - models
    - views
      - layouts
      - show
        - list.html.erb
      - test
        - tester.html.erb
  - bin
  - config
    - environments
    - initializers
    - locales
    - application.rb
    - boot.rb
    - database.yml
    - environment.rb
    - routes.rb
    - secrets.yml
  - db
    - migrate
      - 20141014191614_create_songs.rb
    - development.sqlite3
    - schema.rb
    - seeds.rb
    - test.sqlite3

show_controller.rb | show_helper.rb | show/list.html.erb | seeds.rb | test_helper.r
test_controller.rb | test/tester.html.erb | production.rb | 20141014191614_create_songs.rb | schem

```
<th class="test"><%= link_to_criteria("name", "test", "tester") %></th>
<th class="test"><%= link_to_criteria("artist", "test", "tester") %></th>
<th class="test"><%= link_to_criteria("album", "test", "tester") %></th>
<th class="test"><%= link_to_criteria("id", "test", "tester") %></th>

<p> This is the Test/tester one: Criterion is <%=  @criterion %> </p>

<% @out.each do |item| %>
        <li> <%=h item.send(@criterion) %> <%=h item.id %> </li>
<% end %>
```

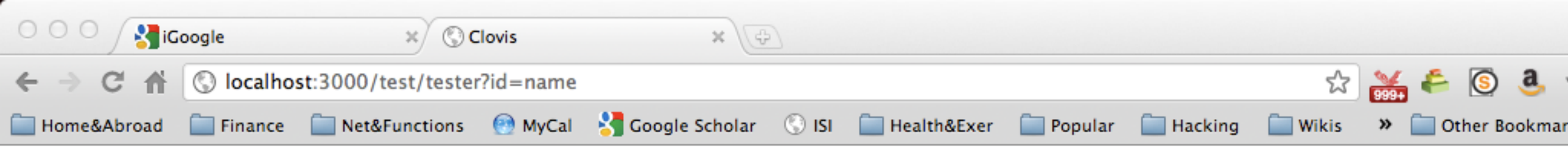
has no **link_to_criteria** method, it is empty

has new screen that prints stuff

(see clovis2)

1: Project | 7: Structure | 6: TODO

7:9 | UTF-8



so, this is working off **link_to_criteria** in the show helper, so the tester view must have access to it; see also **clovis3** for definitive test

how can this happen?

C. De Scaffolding

Scaffolding & REST

- Certain aspects of controller and model creation can be automated more
- For example, most of the time we create controllers for getting inputs, updating tables and showing their contents...perhaps this could be done for us?
- Scaffold & REST do these sort of tasks
- But, like any framework once you buy into them, constraints can follow, that you have to live with

http://en.wikipedia.org/wiki/Representational_State_Transfer

REST is...

- **Re**presentational **S**tate **T**ransfer is a lightweight architect style for designing networked applications; in its use of HTTP protocol to make calls between machines
- Contrasts to heavy-weight CORMA, RPC, SOAP
- RESTful applications use HTTP requests to post data (create/update), read data (e.g., make queries) and delete data; uses HTTP for all four CRUD operations (create/read/update/delete)
- So, we will see GET and POST being used a lot

http://en.wikipedia.org/wiki/Representational_State_Transfer

History [edit]

The REST architectural style was developed in parallel with HTTP/1.1, based on the existing design of HTTP/1.0.[6] The largest implementation of a system conforming to the REST architectural style is the World Wide Web. REST exemplifies how the Web's architecture emerged by characterizing and constraining the macro-interactions of the four components of the Web, namely origin servers, gateways, proxies and clients, without imposing limitations on the individual participants. As such, REST essentially governs the proper behavior of participants.

Concept [edit]

REST-style architectures consist of clients and servers. Clients initiate requests to servers; servers process requests and return appropriate responses. Requests and responses are built around the transfer of representations of resources. A resource can be essentially any coherent and meaningful concept that may be addressed. A representation of a resource is typically a document that captures the current or intended state of a resource.

The client begins sending requests when it is ready to make the transition to a new state. While one or more requests are outstanding, the client is considered to be in transition. The representation of each application state contains links that may be used next time the client chooses to initiate a new state transition.[7]

REST was initially described in the context of HTTP, but is not limited to that protocol. RESTful architectures can be based on other Application Layer protocols if they already provide a rich and uniform vocabulary for applications based on the transfer of meaningful representational state. RESTful applications maximize the use of the pre-existing, well-defined interface and other built-in capabilities provided by the chosen network protocol, and minimize the addition of new application-specific features on top of it.

HTTP examples [edit]

HTTP, for example, has a very rich vocabulary in terms of verbs (or "methods"), URIs, Internet media types, request and response codes, etc. REST uses these existing features of the HTTP protocol, and thus allows existing layered proxy and gateway components to perform additional functions on the network such as HTTP caching and security enforcement.

SOAP RPC contrast [edit]

SOAP RPC over HTTP, on the other hand, encourages each application designer to define a new and arbitrary vocabulary of nouns and verbs (for example `getUsers()`, `savePurchaseOrder(...)`), usually overlaid onto the HTTP POST verb. This disregards many of HTTP's existing capabilities such as authentication, caching and content type negotiation, and may leave the application designer re-inventing many of these features within the new vocabulary.[8] Examples of doing so may include the addition of methods such as `getNewUsersSince(Date date)`, `savePurchaseOrder(string customerLogon, string password, ...)`.

http://en.wikipedia.org/wiki/Representational_State_Transfer

Blog example

$ rails new blog

$ cd blog

$ rails generate scaffold Post name:string title:string content:string

$ rake db:migrate

$ rails server...and look at what is there

whooooa.....Nellie !!!

blog – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/blog] – .../app/controllers/posts_controller.rb

Development: blog

Project /Users/user/Desktop/X_Teaching...

Project

- blog (~/Desktop/X_Teaching/Ruby:2011-14/Lects&
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - posts_controller.rb
    - helpers
    - mailers
    - models
    - views
      - layouts
      - posts
        - _form.html.erb
        - edit.html.erb
        - index.html.erb
        - index.json.jbuilder
        - new.html.erb
        - show.html.erb
        - show.json.jbuilder
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock
  - Rakefile
  - README.rdoc

1: Project

7: Structure

6: TODO

posts_controller.rb | Gemfile | posts/_form.html.erb | schema.rb

```ruby
class PostsController < ApplicationController
  before_action :set_post, only: [:show, :edit, :update, :destroy]

  # GET /posts
  # GET /posts.json
  def index
    @posts = Post.all
  end

  # GET /posts/1
  # GET /posts/1.json
  def show
  end

  # GET /posts/new
  def new
    @post = Post.new
  end

  # GET /posts/1/edit
  def edit
  end

  # POST /posts
  # POST /posts.json
  def create
    @post = Post.new(post_params)

    respond_to do |format|
      if @post.save
        format.html { redirect_to @post, notice: 'Post was successfully created.' }
        format.json { render :show, status: :created, location: @post }
      else
        format.html { render :new }
        format.json { render json: @post.errors, status: :unprocessable_entity }
      end
    end
  end

  # PATCH/PUT /posts/1
  # PATCH/PUT /posts/1.json
  def update
    respond_to do |format|
      if @post.update(post_params)
        format.html { redirect_to @post, notice: 'Post was successfully updated.' }
        format.json { render :show, status: :ok, location: @post }
      else
        format.html { render :edit }
        format.json { render json: @post.errors, status: :unprocessable_entity }
      end
```

blog – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/blog] – .../app/views/posts/index.html.erb – JetBra

Development: blog

Project /Users/user/Desktop/X_Teaching...

Project

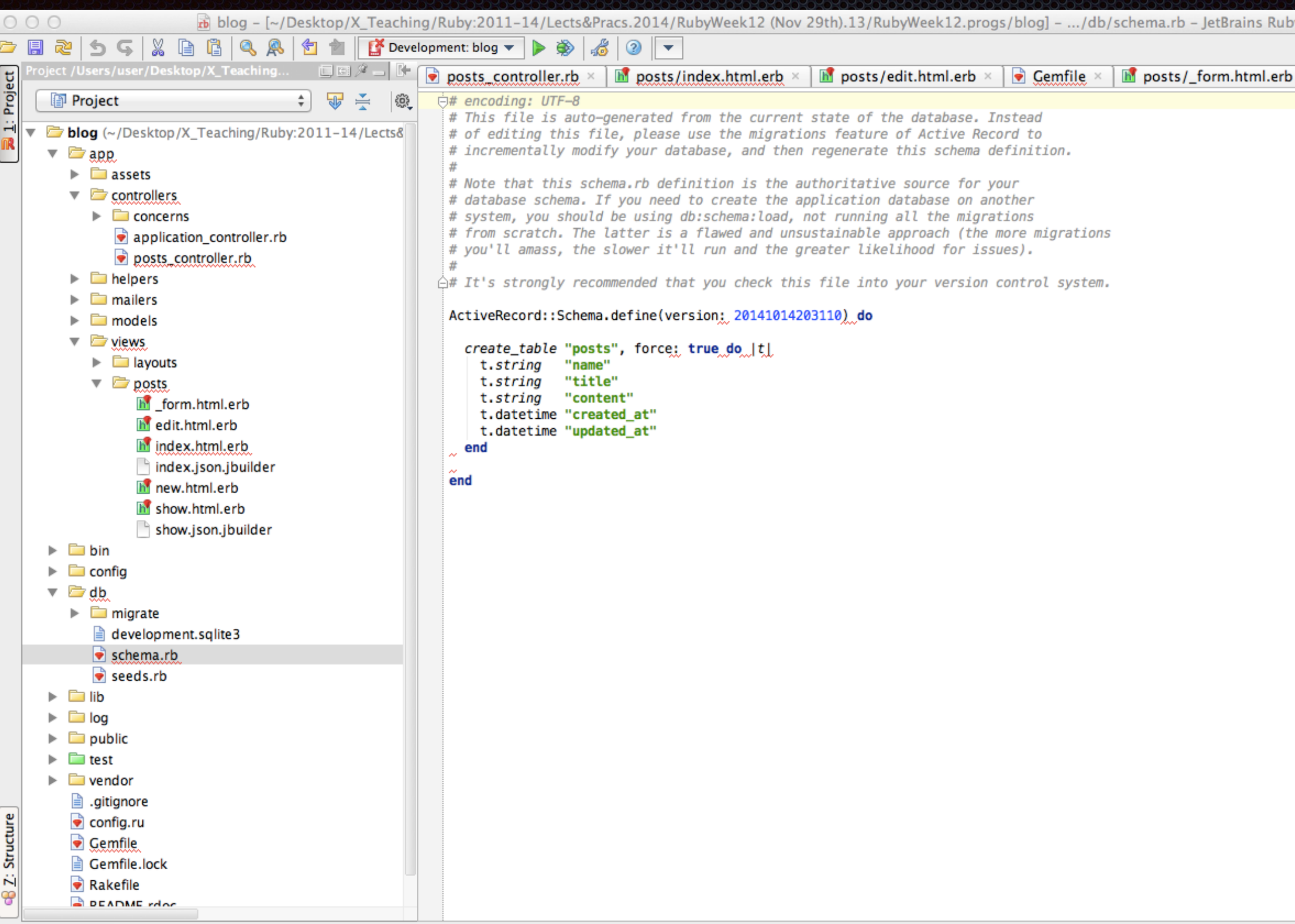
- blog (~/Desktop/X_Teaching/Ruby:2011-14/Lects&
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - posts_controller.rb
    - helpers
    - mailers
    - models
    - views
      - layouts
      - posts
        - _form.html.erb
        - edit.html.erb
        - index.html.erb
        - index.json.jbuilder
        - new.html.erb
        - show.html.erb
        - show.json.jbuilder
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock
  - Rakefile
  - README.rdoc

posts_controller.rb | posts/index.html.erb | posts/edit.html.erb | Gemfile | posts/_form.html.erb

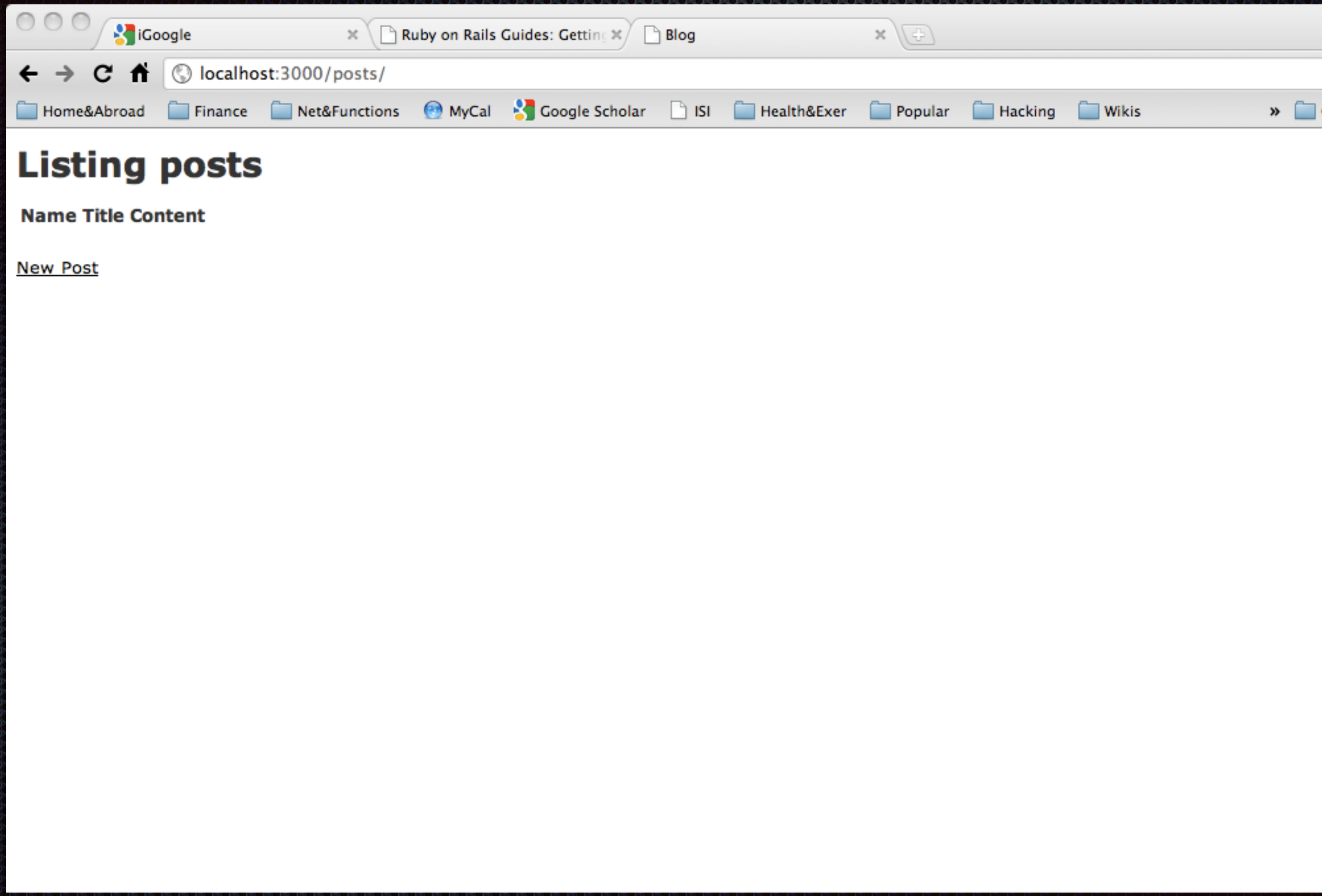
h1

```
<h1>Listing posts</h1>

<table>
  <thead>
    <tr>
      <th>Name</th>
      <th>Title</th>
      <th>Content</th>
      <th colspan="3"></th>
    </tr>
  </thead>

  <tbody>
    <% @posts.each do |post| %>
      <tr>
        <td><%= post.name %></td>
        <td><%= post.title %></td>
        <td><%= post.content %></td>
        <td><%= link_to 'Show', post %></td>
        <td><%= link_to 'Edit', edit_post_path(post) %></td>
        <td><%= link_to 'Destroy', post, method: :delete, data: { confirm: 'Are you sure?' } %></td>
      </tr>
    <% end %>
  </tbody>
</table>

<br>

<%= link_to 'New Post', new_post_path %>
```

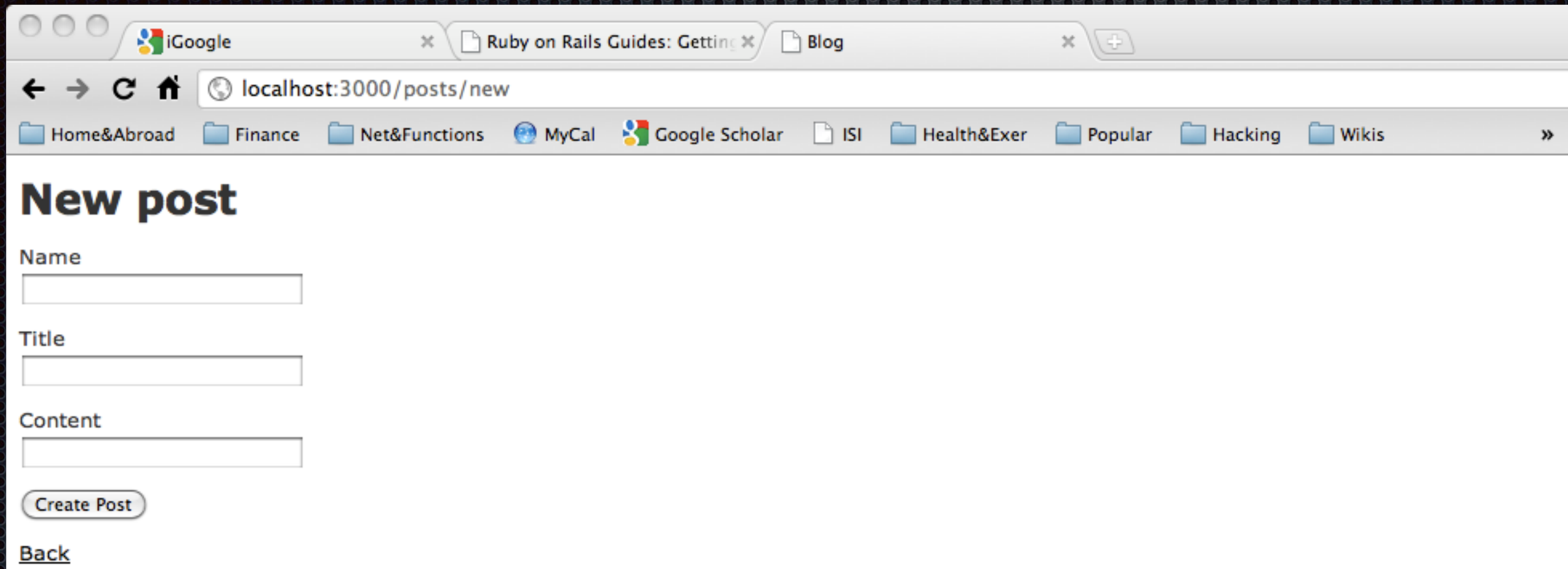
1: Project | 7: Structure | 6: TODO


blog – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/blog] – .../db/schema.rb – JetBrains Rub
Development: blog
Project /Users/user/Desktop/X_Teaching...
Project
blog (~/Desktop/X_Teaching/Ruby:2011-14/Lects&
app
assets
controllers
concerns
application_controller.rb
posts_controller.rb
helpers
mailers
models
views
layouts
posts
_form.html.erb
edit.html.erb
index.html.erb
index.json.jbuilder
new.html.erb
show.html.erb
show.json.jbuilder
bin
config
db
migrate
development.sqlite3
schema.rb
seeds.rb
lib
log
public
test
vendor
.gitignore
config.ru
Gemfile
Gemfile.lock
Rakefile
posts_controller.rb
posts/index.html.erb
posts/edit.html.erb
Gemfile
posts/_form.html.erb
encoding: UTF-8
This file is auto-generated from the current state of the database. Instead
of editing this file, please use the migrations feature of Active Record to
incrementally modify your database, and then regenerate this schema definition.
#
Note that this schema.rb definition is the authoritative source for your
database schema. If you need to create the application database on another
system, you should be using db:schema:load, not running all the migrations
from scratch. The latter is a flawed and unsustainable approach (the more migrations
you'll amass, the slower it'll run and the greater likelihood for issues).
#
It's strongly recommended that you check this file into your version control system.
ActiveRecord::Schema.define(version: 20141014203110) do
create_table "posts", force: true do |t|
t.string "name"
t.string "title"
t.string "content"
t.datetime "created_at"
t.datetime "updated_at"
end
end
1: Project
7: Structure
6: TODO

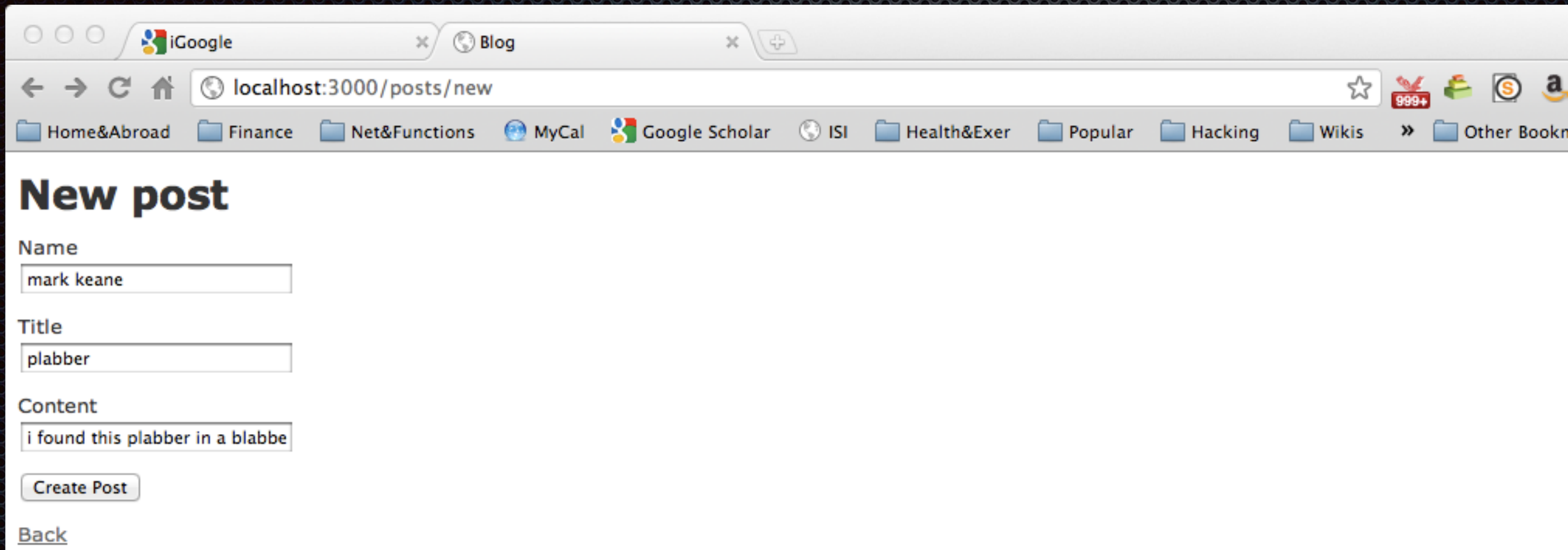
posts/


iGoogle
Ruby on Rails Guides: Getting
Blog
localhost:3000/posts/
Home&Abroad
Finance
Net&Functions
MyCal
Google Scholar
ISI
Health&Exer
Popular
Hacking
Wikis
Listing posts
Name Title Content
New Post

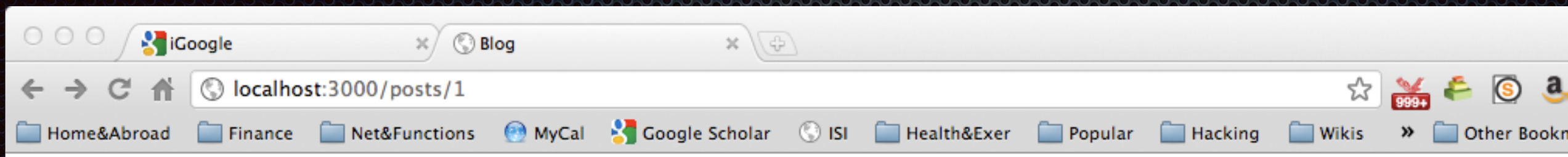
posts/new


iGoogle
Ruby on Rails Guides: Getting
Blog
localhost:3000/posts/new
Home&Abroad
Finance
Net&Functions
MyCal
Google Scholar
ISI
Health&Exer
Popular
Hacking
Wikis
New post
Name
Title
Content
Create Post
Back

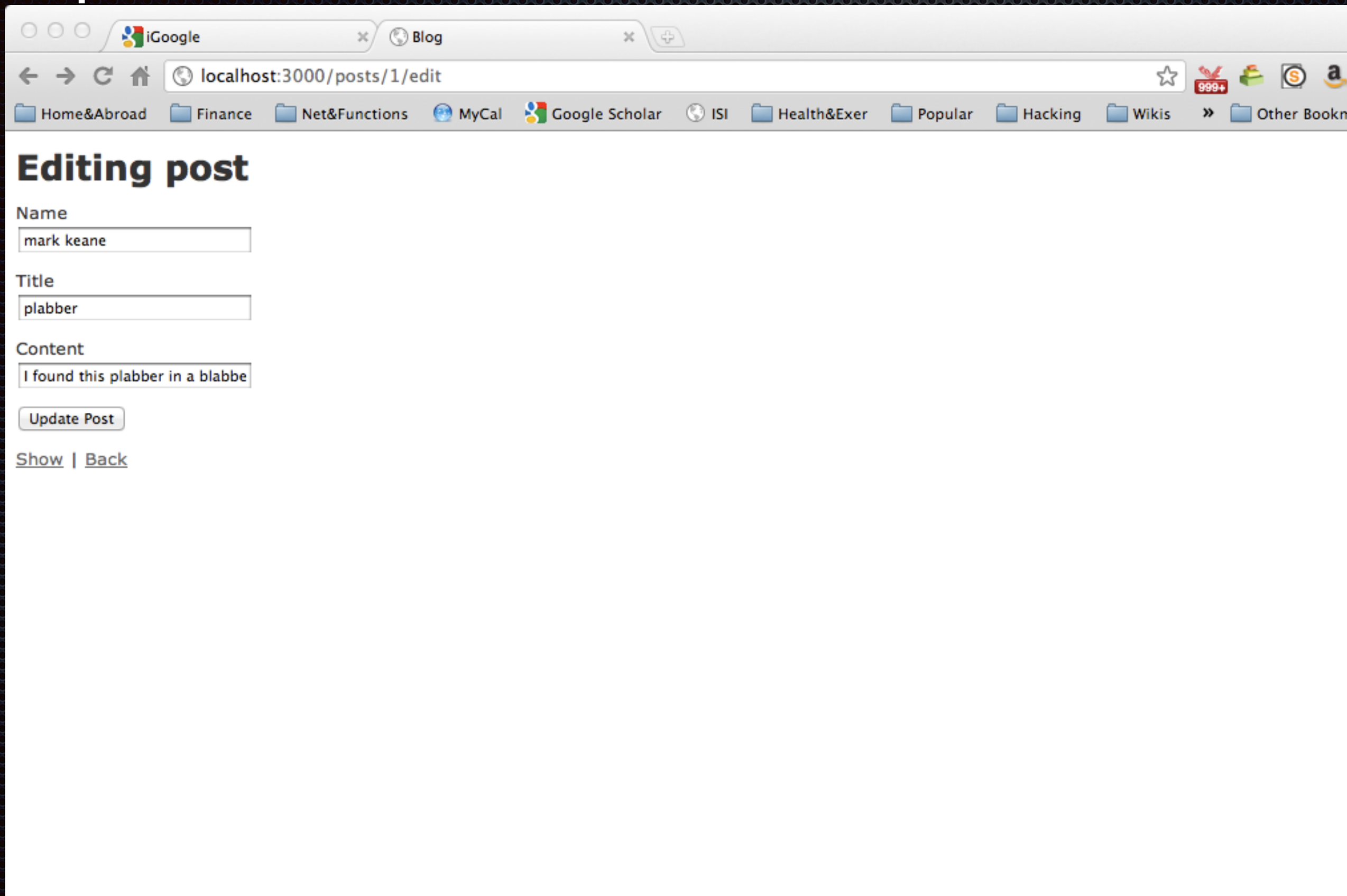
posts/new


iGoogle
Blog
localhost:3000/posts/new
Home&Abroad
Finance
Net&Functions
MyCal
Google Scholar
ISI
Health&Exer
Popular
Hacking
Wikis
Other Bookm
New post
Name
mark keane
Title
plabber
Content
i found this plabber in a blabbe
Create Post
Back

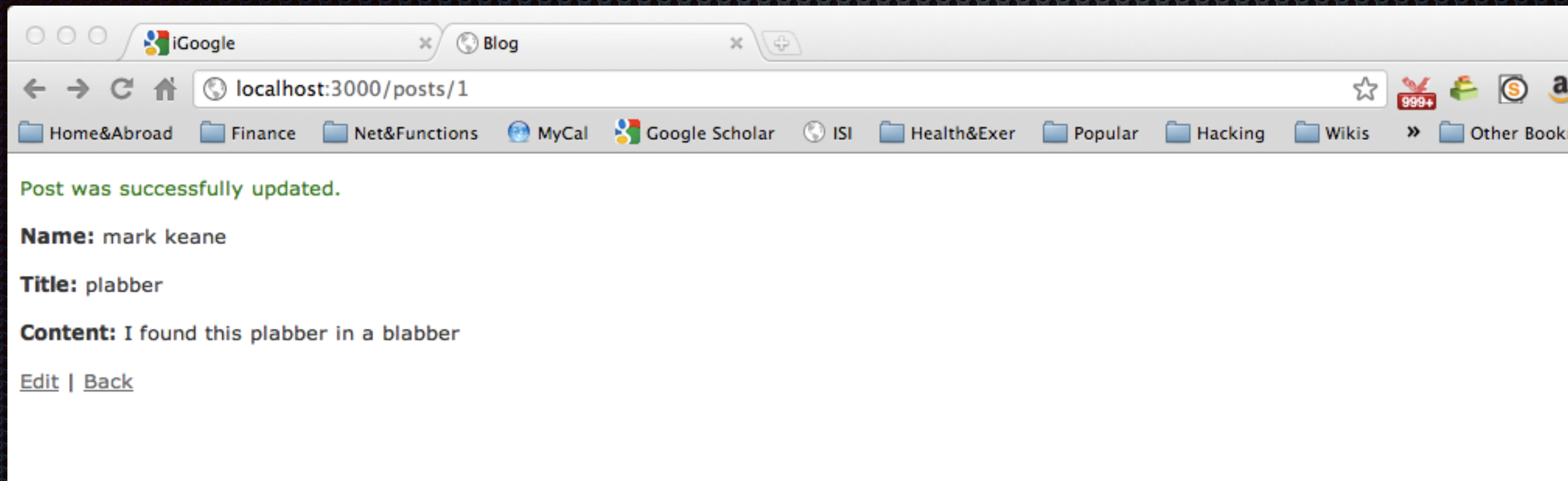
posts/1


iGoogle
Blog
localhost:3000/posts/1
Home&Abroad
Finance
Net&Functions
MyCal
Google Scholar
ISI
Health&Exer
Popular
Hacking
Wikis
Other Bookm
Post was successfully updated.
Name: mark keane
Title: plabber
Content: I found this plabber in a blabber
Edit | Back

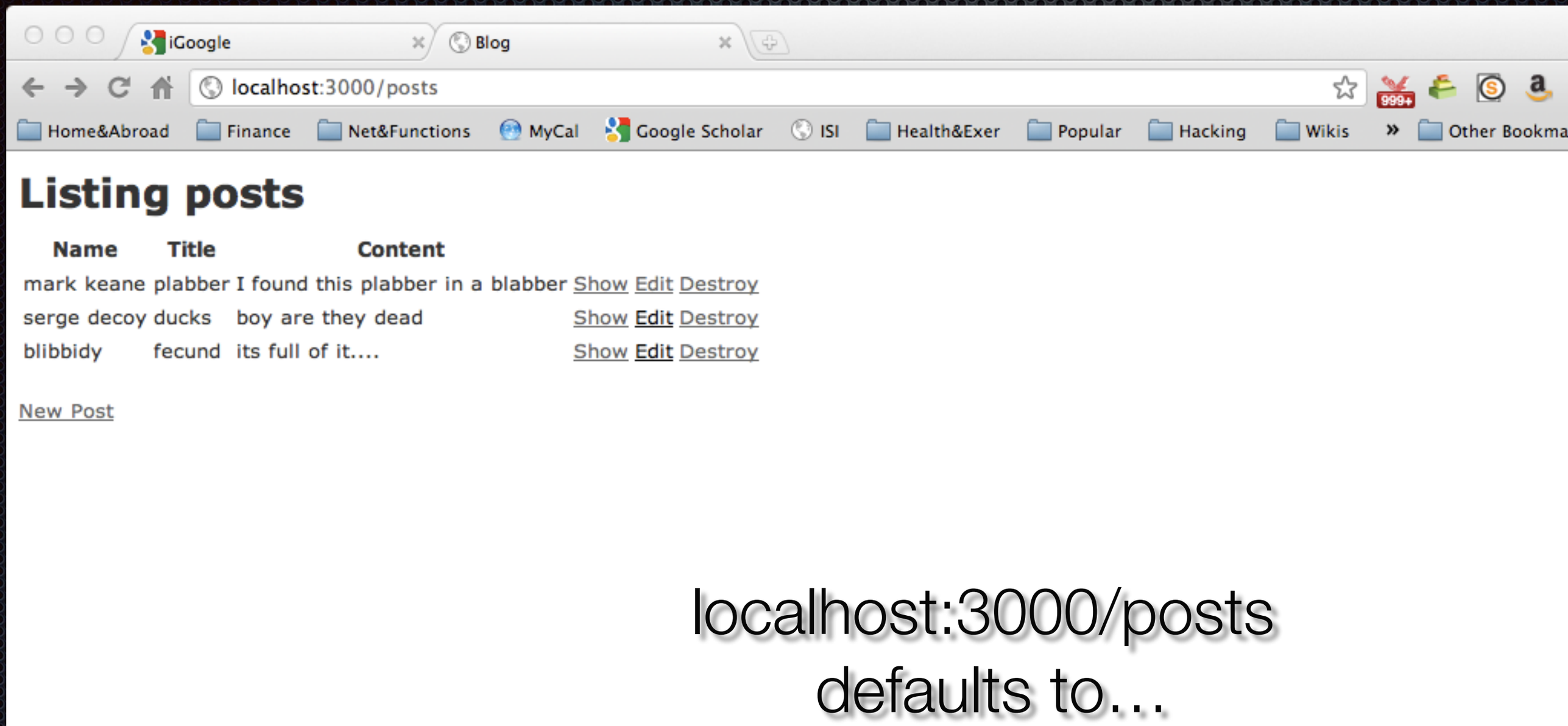
posts/1/edit


iGoogle
Blog
localhost:3000/posts/1/edit
Home&Abroad
Finance
Net&Functions
MyCal
Google Scholar
ISI
Health&Exer
Popular
Hacking
Wikis
Other Bookm
Editing post
Name
mark keane
Title
plabber
Content
I found this plabber in a blabbe
Update Post
Show | Back

...updated...

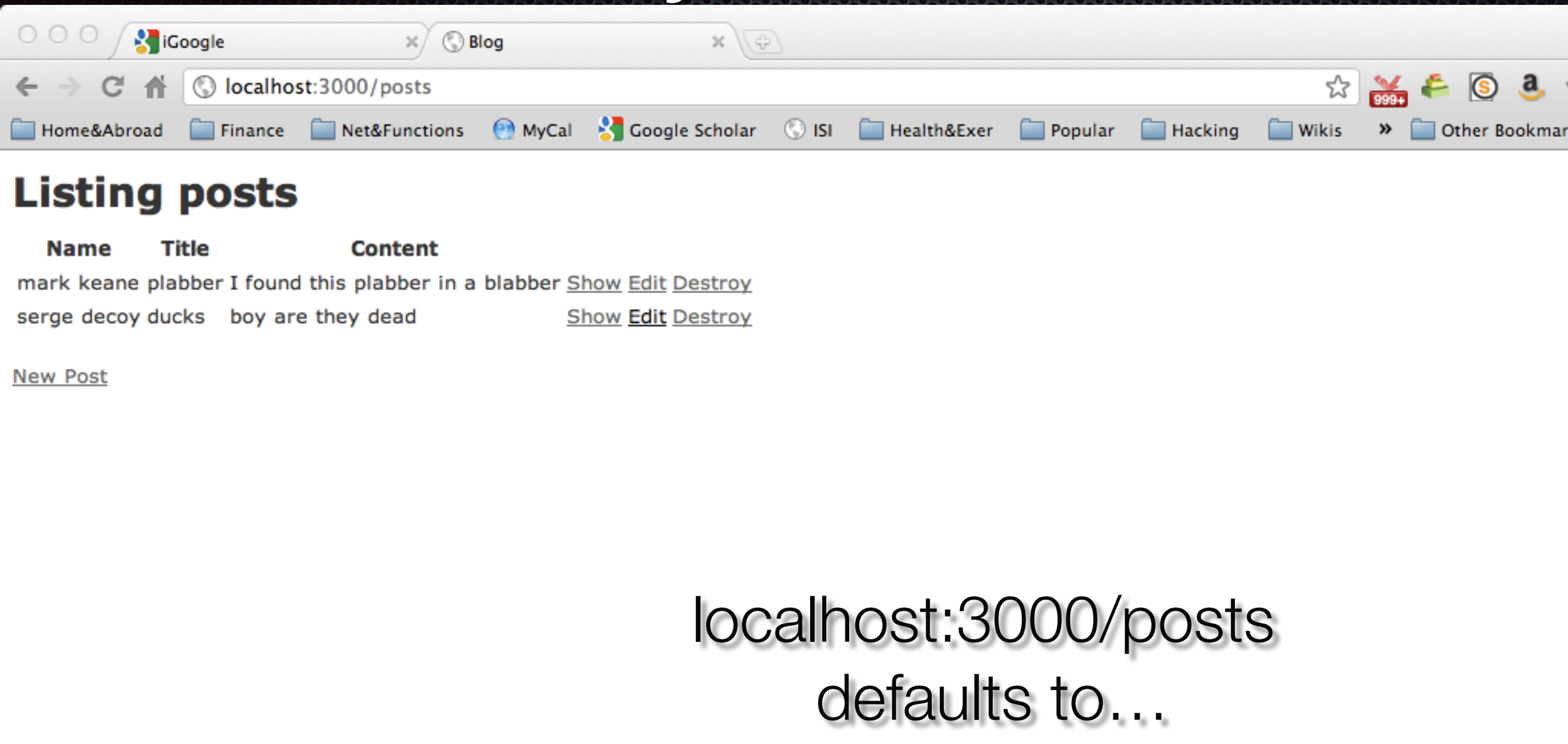

iGoogle
Blog
localhost:3000/posts/1
Home&Abroad
Finance
Net&Functions
MyCal
Google Scholar
ISI
Health&Exer
Popular
Hacking
Wikis
Other Book
Post was successfully updated.
Name: mark keane
Title: plabber
Content: I found this plabber in a blabber
Edit | Back

...back sends you to...



localhost:3000/posts
defaults to…
localhost:3000/posts/index

...and destroy...



localhost:3000/posts
defaults to…
localhost:3000/posts/index

blog – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/blog] – .../app/controllers/posts_controller.rb

Development: blog

Project /Users/user/Desktop/X_Teaching...

Project

- blog (~/Desktop/X_Teaching/Ruby:2011-14/Lects&
  - app
    - assets
    - controllers
      - concerns
      - application_controller.rb
      - posts_controller.rb
    - helpers
    - mailers
    - models
    - views
      - layouts
      - posts
        - _form.html.erb
        - edit.html.erb
        - index.html.erb
        - index.json.jbuilder
        - new.html.erb
        - show.html.erb
        - show.json.jbuilder
  - bin
  - config
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock
  - Rakefile
  - README.rdoc

posts_controller.rb | Gemfile | posts/_form.html.erb | schema.rb

```ruby
class PostsController < ApplicationController
  before_action :set_post, only: [:show, :edit, :update, :destroy]

  # GET /posts
  # GET /posts.json
  def index
    @posts = Post.all
  end

  # GET /posts/1
  # GET /posts/1.json
  def show
  end

  # GET /posts/new
  def new
    @post = Post.new
  end

  # GET /posts/1/edit
  def edit
  end

  # POST /posts
  # POST /posts.json
  def create
    @post = Post.new(post_params)

    respond_to do |format|
      if @post.save
        format.html { redirect_to @post, notice: 'Post was successfully created.' }
        format.json { render :show, status: :created, location: @post }
      else
        format.html { render :new }
        format.json { render json: @post.errors, status: :unprocessable_entity }
      end
    end
  end

  # PATCH/PUT /posts/1
  # PATCH/PUT /posts/1.json
  def update
    respond_to do |format|
      if @post.update(post_params)
        format.html { redirect_to @post, notice: 'Post was successfully updated.' }
        format.json { render :show, status: :ok, location: @post }
      else
        format.html { render :edit }
        format.json { render json: @post.errors, status: :unprocessable_entity }
      end
```

1: Project | 7: Structure | 6: TODO

Scaffold creates

File	Purpose
db/migrate/20100207214725_create_posts.rb.rb	Migration to create the posts table in your database (your name will include a different timestamp)
app/models/post.rb	The Post model
test/fixtures/posts.yml	Dummy posts for use in testing
app/controllers/posts_controller.rb	The Posts controller
app/views/posts/index.html.erb	A view to display an index of all posts
app/views/posts/edit.html.erb	A view to edit an existing post
app/views/posts/show.html.erb	A view to display a single post
app/views/posts/new.html.erb	A view to create a new post
app/views/posts/_form.html.erb	A partial to control the overall look and feel of the form used in edit and new views
app/helpers/posts_helper.rb	Helper functions to be used from the post views
test/unit/post_test.rb	Unit testing harness for the posts model
test/functional/posts_controller_test.rb	Functional testing harness for the posts controller
test/unit/helpers/posts_helper_test.rb	Unit testing harness for the posts helper
config/routes.rb	Edited to include routing information for posts
public/stylesheets/scaffold.css	Cascading style sheet to make the scaffolded views look better

D: Deploying a Rails App...

Getting to the world...wide web

Deployment Options...

- There are several different options for launching your rails app on the Web, assuming you have a domain name for it
- Depends on whether (i) you are hosting it on your own server or (ii) using some service
- Often this may demand some db changes...to a more robust db than sqlite
- nb. it will use the production version of the system

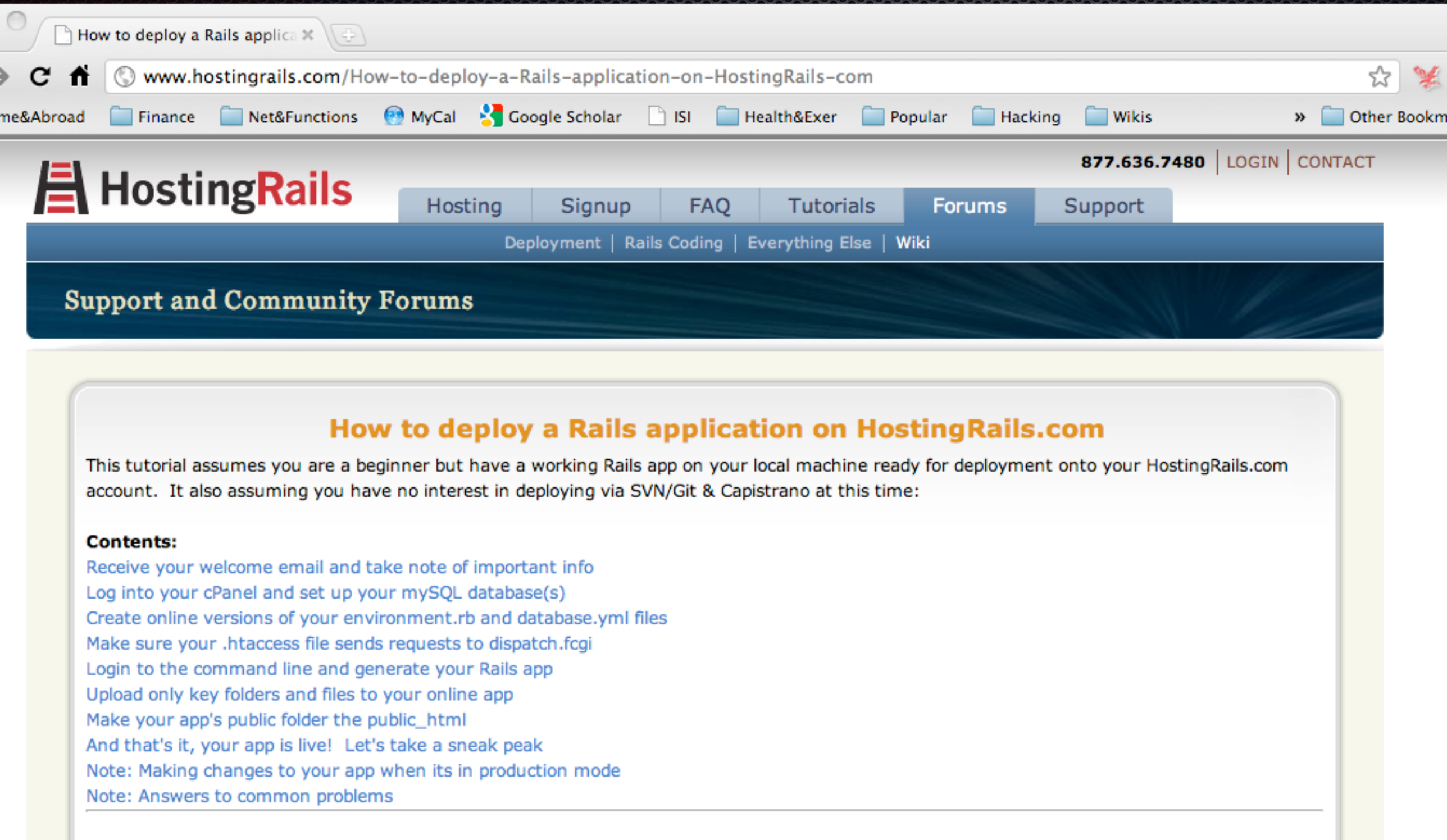
By Hand Yourself...

- Register your site domain -- www.mysite.com -- and point it to your site
- Modify e.g. Apache config file (diff in diff servers)

```
<VirtualHost www.mysite.com>
ServerName www.mysite.com
ServerAlias mysite.com
DocumentRoot “usr/local/Share/railsapp/mysite/public”
</VirtualHost>
```

- Apache looks at dir for **.htaccess** file (in public directory of Rails app; see README)
- it will trigger execution of dispatcher, calling controllers appropriately

Via a Service...



http://www.hostingrails.com/

Various Tools

- Capistrano: used for running scripts on multiple servers; automates task of making new version of application available on one/more servers
- Subversion (SVN) for version control
- VRM for version control
- Git and Github for version control

E: YAML

for dumping dbs and passing around data

YAML definition

- is a machine parse-able data serialization format designed for human readability and for use with scripting languages (e.g., perl and python)
- is optimized for data serialization, formatted dumping, configuration files, log files, internet messaging and filtering

YAML on arrays

```
require 'yaml'

test_1 = ["this", "is", "an", ["embedded", "array"], 2, 3]
test_2 = ["level1", ["level2", ["level 3"]], 2, 3]

p test_1
puts YAML::dump(test_1)

p test_2
puts YAML::dump(test_2)

yml_obj = YAML::dump(test_1)

p YAML::load(yml_obj)
```

yaml.rb

YAML on arrays

```
require 'yaml'

test_1 = ["this", "is", "an", ["embed
"array"], 2, 3]
test_2 = ["level1", ["level2", ["leve
2, 3]

p test_1
puts YAML::dump(test_1)

p test_2
puts YAML::dump(test_2)

yml_obj = YAML::dump(test_1)

p YAML::load(yml_obj)
```

```
ruby yaml.rb
["this", "is", "an", ["embedded", "array"], 2, 3]
---
- this
- is
- an
- - embedded
  - array
- 2
- 3
["level1", ["level2", ["level 3"]], 2, 3]
---
- level1
- - level2
  - - level 3
- 2
- 3
["this", "is", "an", ["embedded", "array"], 2, 3]
```

YAML on config files

- in any rails app the config folder has **database.yml**, that looks like this:

```
# SQLite version 3.x
#   gem install sqlite3

default: &default
  adapter: sqlite3
  pool: 5
  timeout: 5000

development:
  <<: *default
  database: db/development.sqlite3

# Warning: …
# Do not set this db to the same as development or production.
test:
  <<: *default
  database: db/test.sqlite3

production:
  <<: *default
  database: db/production.sqlite3
```

database.yml

blog – [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/blog] – .../config/database.yml – JetBrains RubyMine

yaml

Project /Users/user/Desktop/X_Teaching/R...

Project

- blog (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Prac
  - app
  - bin
  - config
    - environments
      - development.rb
      - production.rb
      - test.rb
    - initializers
    - locales
    - application.rb
    - boot.rb
    - database.yml
    - environment.rb
    - routes.rb
    - secrets.yml
  - db
    - migrate
    - development.sqlite3
    - schema.rb
    - seeds.rb
  - lib
  - log
  - public
  - test
  - vendor
  - .gitignore
  - config.ru
  - Gemfile
  - Gemfile.lock
  - Rakefile
  - README.rdoc
- External Libraries

posts/_form.html.erb | schema.rb | routes.rb | application.rb | boot.rb
development.rb | database.yml | environment.rb

```
# SQLite version 3.x
#   gem install sqlite3
#
#   Ensure the SQLite 3 gem is defined in your Gemfile
#   gem 'sqlite3'
#
default: &default
  adapter: sqlite3
  pool: 5
  timeout: 5000

development:
  <<: *default
  database: db/development.sqlite3

# Warning: The database defined as "test" will be erased and
# re-generated from your development database when you run "rake".
# Do not set this db to the same as development or production.
test:
  <<: *default
  database: db/test.sqlite3

production:
  <<: *default
  database: db/production.sqlite3
```

1: Project | 7: Structure | 6: TODO

24:53

See what YAML does...

```
require 'yaml'

file = File.open('database.txt')

p YAML::load(file)
```

loader.rb

```
$ ruby loader.rb

{"default"=>{"adapter"=>"sqlite3", "pool"=>5, "timeout"=>5000},
"development"=>{"adapter"=>"sqlite3", "pool"=>5, "timeout"=>5000,
"database"=>"db/development.sqlite3"}, "test"=>{"adapter"=>"sqlite3",
"pool"=>5, "timeout"=>5000, "database"=>"db/test.sqlite3"},
"production"=>{"adapter"=>"sqlite3", "pool"=>5, "timeout"=>5000,
"database"=>"db/production.sqlite3"}}
```

database.yml for mysql

```
development:
  adapter: mysql2
  encoding: utf8
  database: blog_development
  pool: 5
  username: root
  password:
  socket: /tmp/mysql.sock
  ....
```

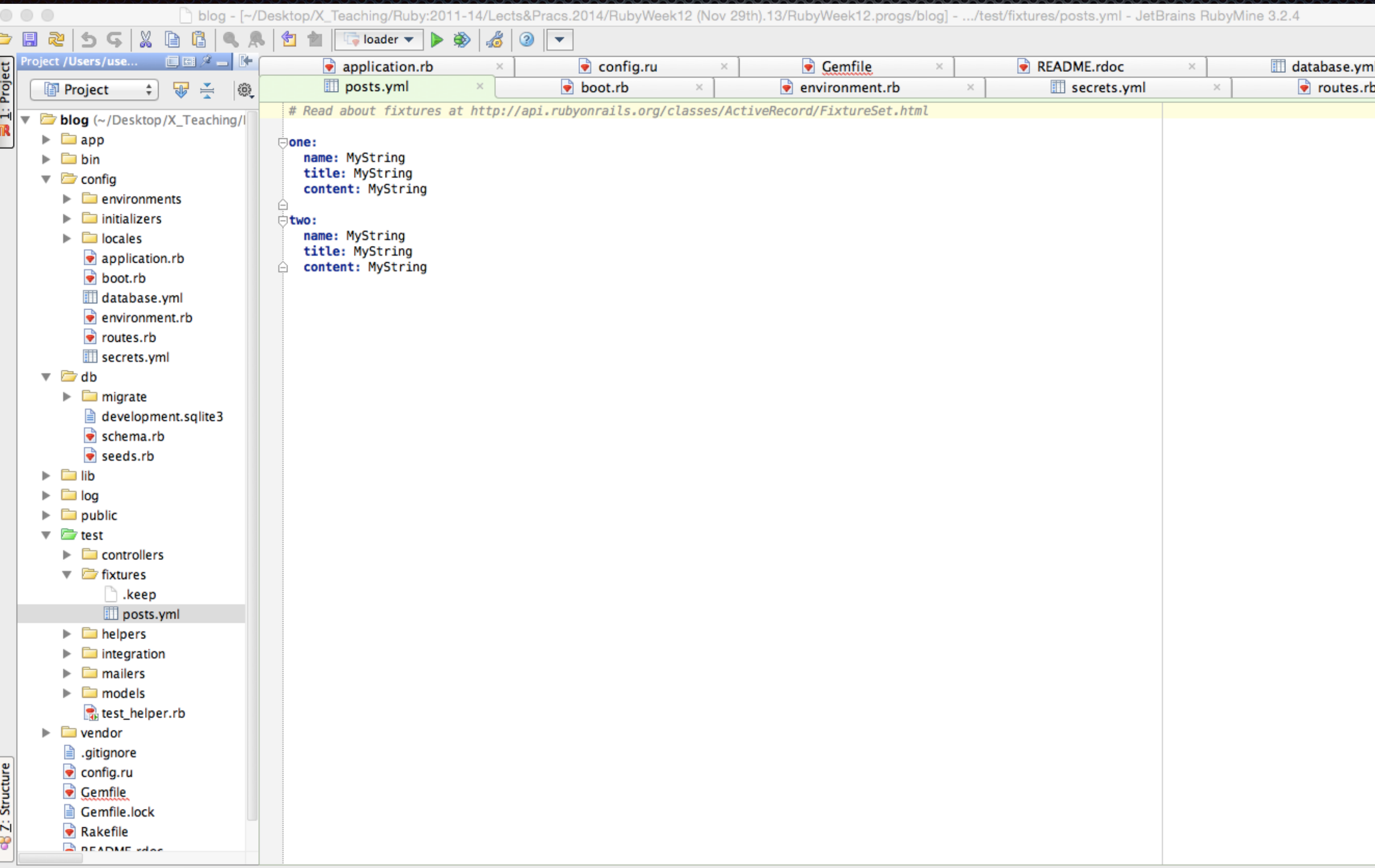
config.yml

YAML on dumping a db

- When moving dbs the contents of a db can be dumped into a YAML file and then re-loaded later into a new db
- for test purposes there is a yml file in fixures that can be loaded into the database with:

  $ rake db:fixtures:load

fixtures


blog - [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek12 (Nov 29th).13/RubyWeek12.progs/blog] - .../test/fixtures/posts.yml - JetBrains RubyMine 3.2.4
loader
Project /Users/use...
Project
blog (~/Desktop/X_Teaching/I
app
bin
config
environments
initializers
locales
application.rb
boot.rb
database.yml
environment.rb
routes.rb
secrets.yml
db
migrate
development.sqlite3
schema.rb
seeds.rb
lib
log
public
test
controllers
fixtures
.keep
posts.yml
helpers
integration
mailers
models
test_helper.rb
vendor
.gitignore
config.ru
Gemfile
Gemfile.lock
Rakefile
application.rb
config.ru
Gemfile
README.rdoc
database.yml
posts.yml
boot.rb
environment.rb
secrets.yml
routes.rb
Read about fixtures at http://api.rubyonrails.org/classes/ActiveRecord/FixtureSet.html
one:
name: MyString
title: MyString
content: MyString
two:
name: MyString
title: MyString
content: MyString
1: Project
7: Structure

```
hcp-892bc88c:clovis3 mkeane$ rails console
Loading development environment (Rails 4.0.1)
irb(main):001:0> Song.all
=> [#<Song id: 1, name: "The End", artist: "Doors", album: "Greatest Hits", created_at: "2010-11-16 18:37:00",
updated_at: "2010-11-16 18:37:00">, #<Song id: 2, name: "Red Hot", artist: "RHCP", album: "What Hits", created_at:
"2010-11-16 18:37:00", updated_at: "2010-11-16 18:37:00">, #<Song id: 3, name: "Solid Air", artist: "John Martyn",
album: "Solid Air", created_at: "2010-11-16 18:37:00", updated_at: "2010-11-16 18:37:00">, #<Song id: 4, name:
"May You Never", artist: "John Martyn", album: "Solid Air", created_at: "2010-11-16 18:37:01", updated_at:
"2010-11-16 18:37:01">, #<Song id: 5, name: "May You Never", artist: "Eric Clapton", album: "Clap Hits", created_at:
"2010-11-16 18:37:01", updated_at: "2010-11-16 18:37:01">]
irb(main):002:0> quit

dhcp-892bc88c:clovis3 mkeane$ rake db:fixtures:load
...
dhcp-892bc88c:clovis3 mkeane$ rails console
Loading development environment (Rails 4.0.1)
irb(main):001:0> Song.all
=> [#<Song id: 298486374, name: "MyString", artist: "MyString", album: "MyString", created_at: "2011-11-04
12:01:12", updated_at: "2011-11-04 12:01:12">, #<Song id: 980190962, name: "MyString", artist: "MyString", album:
"MyString", created_at: "2011-11-04 12:01:12", updated_at: "2011-11-04 12:01:12">, #<Song id: 980190963, name:
"The End", artist: "Doors", album: "Greatest Hits", created_at: "2011-11-04 12:02:29", updated_at: "2011-11-04
12:02:29">, #<Song id: 980190964, name: "Red Hot", artist: "RHCP", album: "What Hits", created_at: "2011-11-04
12:02:29", updated_at: "2011-11-04 12:02:29">, #<Song id: 980190965, name: "Solid Air", artist: "John Martyn",
album: "Solid Air", created_at: "2011-11-04 12:02:29", updated_at: "2011-11-04 12:02:29">, #<Song id: 980190966,
name: "May You Never", artist: "John Martyn", album: "Solid Air", created_at: "2011-11-04 12:02:29", updated_at:
"2011-11-04 12:02:29">, #<Song id: 980190967, name: "May You Never", artist: "Eric Clapton", album: "Clap Hits",
created_at: "2011-11-04 12:02:29", updated_at: "2011-11-04 12:02:29">]
```

F: De Exam

Question Types...

University College Dublin
An Coláiste Ollscoile, Baile Átha Cliath

SEMESTER I EXAMINATION – 2015/2016

COMP 41100

Exploring Programming in Ruby

Prof. S. Dobson

Prof. P. Cunningham

Prof. M. Keane*

Module Coordinator*

Time allowed: 2 hours

Instructions for candidates

Answer Five Questions:

out of **7** listed

Question One from Part I is Compulsory,

Answer Any Two Questions from Part II

Answer Any Two Questions from Part III

All Questions carry equal marks.

Use of calculators is prohibited.

Instructions for invigilators

Use of calculators is prohibited.

Part I, Q1:Compulsory

3. What do the following evaluate to in Ruby:

a. “hello there big boy”
b. 56
c. 34.00
d. 0.222222354454365
e. [“a”, “b”, “c”]
f. +
g. Pl.....

Part II:Programming Qs

Define a class for FamilyMember that has at least 5 properties (e.g., name, sex, status, age) and then create 5 instances of that class, each of which should be assigned to the following variables: fm1, fm2, fm3, fm4, fm5. Define two methods – parent? and child? that will test whether an instance of this class is a parent or a child and what type of parent or child (father, mother, son daughter, dog). The method should make use of two or more properties in doing this test.

Define an array of these family members and search through it using each, to print out their names, and status.

Define an array of these family members and search through it using do, to print out their names, and status.

Part III: Concept Qs

1. Write a short paragraph on any ***four*** of the key properties that are asserted to be characteristic of the Object-Oriented Programming paradigm, using appropriate examples.

2. What are Ruby Gems? Write an essay on the functionality of two Gems with which you are familiar, illustrating your answer with examples.

What I have tried to do...

in Ruby and Rails

Ruby Covered

- Give you a overall sense of how the language works
- Give you enough knowledge to understand any bits that were not covered
- You should understand what any code *might* be from looking at its syntax etc...even if the methods are not familiar

Rails Covered

- Give you a overall sense of how the Rails framework works
- We have covered less of Rails, but much of the remainder is quite specific stuff about forms/cgi/validation/cookies
- None of this should be surprising, though there will be domain-specific complexities in its use

Goodbye....Goodbye....
Goodbye....