

Ruby Explorations X

Mark Keane...CSI...UCD



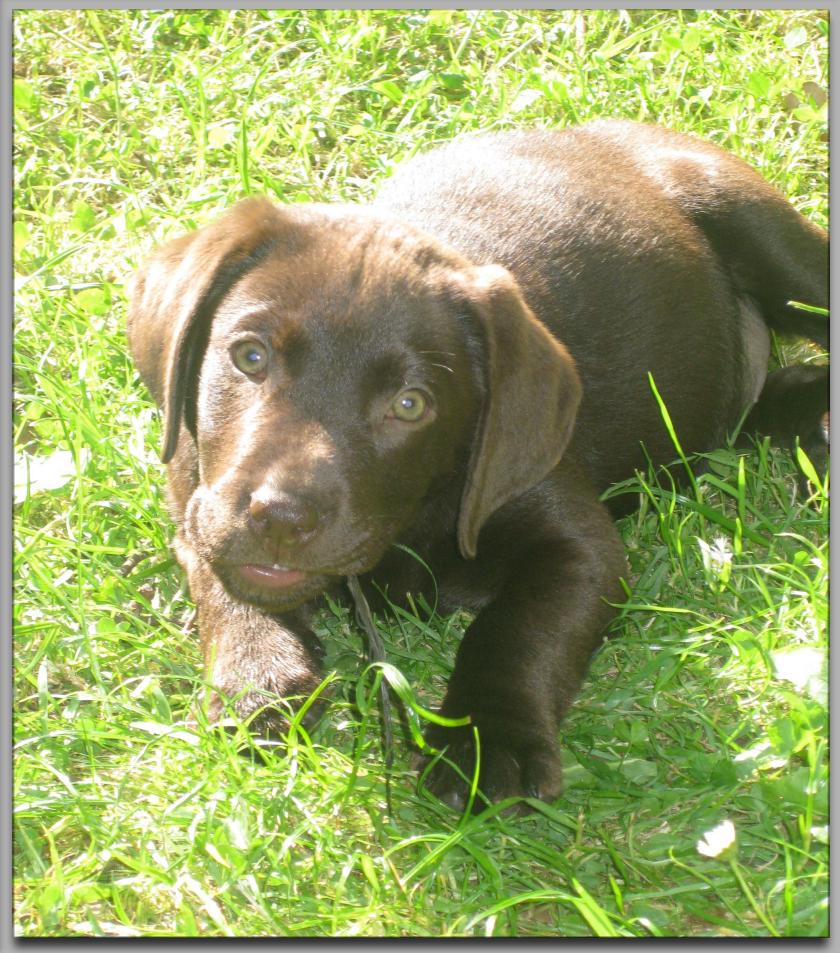
Rails Basics

Ruby on Rails I:

- A: the design theory behind Rails framework
- B: walk-thru' of simple Rails system
- C: other Rails functionality...neat stuff
- D: now for models in Hello
- E: some residual things to know...
- F: fixing the db problem in Hello

My New Puppy

this is my second puppy
her name is ...



Part A:

practical and theoretical background

What is Rails...

software framework, a set of libraries to support rapid web-site development

when you type ***rails new sitename***, a folder is opened up with bags and bags of sub-folders and files

these sub-folders contain file templates for coding and ready-made methods, a web server (WEBrick) and a database (SQLite3); related by ***conventions***

the thing is a monster...ALSO...with Rails 4.1.6

newsite - [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/newsite] - .../app/controllers/test_controller.rb - JetBrains...

Project /Users/user/Desktop/X_Teachin... Development: newsite

1: Project

newsite (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/newsite)

app

- assets
- controllers
 - concerns
 - application_controller.rb
 - test_controller.rb
- helpers
 - application_helper.rb
 - test_helper.rb
- mailers
- models
- views
 - layouts
 - test
 - hello.html.erb
- bin
- config
 - environments
 - initializers
 - locales
 - application.rb
 - boot.rb
 - database.yml
 - environment.rb
 - routes.rb
 - secrets.yml
- db

2: Structure

test_controller.rb x test/hello.html.erb x

```
class TestController < ApplicationController
  def hello
    @message = "hello"
  end
end
```

Data Sources

6: TODO

6.7 UTF-8

What is Rails...

originally, web-pages were static html but this was very inflexible and hard to maintain

then...they became more dynamic with databases behind them...then people started putting code in so that the page was created on the fly (e.g., personalised)

this improved flexibility and maintenance but introduced complexity and security issues

Rails is a response to these issues...

What is Rails...MVC

complexity is handled by dividing task into model (db setup), view (screens) and controller (code to do things)

in simple terms, what was a single html file with code, gets broken up into 3-4 files, which combine to create a file that is sent to browser to be rendered

its secure 'cos website users can only see the views and user have no sight of code for views or models

the model/controller bit just separates db issues from code that modifies objects (sort of)

MVC framework

model-view-controller is a software architecture/architectural pattern (used in Rails)

model is domain-specific rep of the data used by the application; manages data and informs of changes

view renders model into suitable form for interactions (ie, at interface); may be multiple views of same model

controller receives input and initiates response, making calls to model data (sort-of guts of app)

Rails...runs off...

Rails is backed by libraries, (some) we have seen:

ActionMailer: for mailing

ActiveRecord: for interface to db

Rake: for setting up, updating and clearing out db

Erb: for writing embedded ruby commands in docs

we will adopt a hide-the-db approach

Controller

Model

Views

MVC flow of control

user interacts with interface (mouse-click)

controller receives input event and converts it to suitable user action, understandable for model

controller notifies **model** of user action, poss. changing model's state (e.g., update shopping cart)

once **controller** does actions, based on **model** data, flow of control may pass back to **view** to render output (or move on to next page/action)

user interface may await further user input, cycle restarts

Controller

Model

Views

Practical Issues

Or Xcode...if you
know/prefer it

we will work with four windows: Rubymine, 2
command-line windows and a browser

Rubymine helps us navigate folders and edits files

TerminalWindow1 will run the server

TerminalWindow2 will run external scripts (e.g. rake)

Browser will look at web-site pages running

get used to moving between all four windows

Window-1

```
Last login: Sun Oct  5 13:14:36 on ttys000
MobileMac:~ user$ ls
Applications      Downloads      Movies          Pictures        nltk_data
Desktop          Dropbox       Music           Public
Documents         Google Drive  Mystuff        ROS
DocumentsMY       Library      Papers        RubymineProjects
MobileMac:~ user$
```

Window-2

```
Started GET "/assets/test.css?body=1" for 127.0.0.1 at 2014-10-05 19:48:37 +0100

Started GET "/assets/jquery_ujs.js?body=1" for 127.0.0.1 at 2014-10-05 19:48:37 +0100

Started GET "/assets/jquery.js?body=1" for 127.0.0.1 at 2014-10-05 19:48:37 +0100

Started GET "/assets/test.js?body=1" for 127.0.0.1 at 2014-10-05 19:48:37 +0100

Started GET "/assets/application.js?body=1" for 127.0.0.1 at 2014-10-05 19:48:37 +0100
^C[2014-10-05 19:49:21] INFO  going to shutdown ...
[2014-10-05 19:49:21] INFO  WEBrick::HTTPServer#start done.
Exiting
MobileMac:newsite user$ ls
Gemfile      Rakefile      config      lib      test
Gemfile.lock  app          config.ru   log      tmp
README.rdoc  bin          db          public   vendor
MobileMac:newsite user$ rails server
-bash: rail: command not found
MobileMac:newsite user$ rails server
Warning: Running 'gem pristine --all' to regenerate your installed gems
cs (and deleting then reinstalling your bundle if you use bundle --path)
will improve the startup performance of Spring.
=> Booting WEBrick
=> Rails 4.1.6 application starting in development on http://0.0.0.0:3000
=> Run `rails server -h` for more startup options
=> Notice: server is listening on all interfaces (0.0.0.0). Consider using
127.0.0.1 (--binding option)
=> Ctrl-C to shutdown server
[2014-10-05 19:56:34] INFO  WEBrick 1.3.1
[2014-10-05 19:56:34] INFO  ruby 2.0.0 (2014-05-08) [universal.x86_64-darwin13]
[2014-10-05 19:56:34] INFO  WEBrick::HTTPServer#start: pid=89634 port=3000
```

Ruby on Rails: Welcome aboard

localhost:3000

Home&Abroad Finance Net&Functions MyCal Google Scholar ISI Health&Exer Popular Hacking Wikis 999+



Welcome aboard

You're riding Ruby on Rails!

[About your application's environment](#)

Getting started

Here's how to get rolling:

1. Use `rails generate` to create your models and controllers

To see all available options, run it without parameters.

2. Set up a default route and remove `public/index.html`

Routes are set up in `config/routes.rb`.

3. Create your database

Run `rake db:create` to create your database. If you're not using SQLite (the default), edit `config/database.yml` with your username and password.

Browse the documentation

[Rails Guides](#)
[Rails API](#)
[Ruby core](#)
[Ruby standard library](#)

Part B(i):

simplest possible 7-step recipe to use Rails...

Controller

Model

Views

Controller

Model

\$ rake db:migrate

Views

\$ rails server

Before Cookbook Run

Please re-adjust your trousers...

Make sure your gems are in place...run any or all of these

\$ gem install rails (make sure **rake** is there)

\$ gem update --system

\$ gems install a_gem (if **a_gem** is missing)

\$ gem install bundle

\$ bundle install

\$ bundle update **a_particular_gem** (to get latest v)

Cookbook Run

\$ rails new newsite

first time may pause at “run bundle”
if impatient use this... -- skip-bundle
Mac issues on privileges...

\$ cd newsite

\$ rails generate controller test hello

look at /newsite/app/controllers/
test_controller.rb

add **@message = “hello”** to hello method

look at newsite/app/views/test/hello.html.erb
add **<p> <%= @message %></p>** to file

\$ rails server

load page in browser:

<http://localhost:3000/test/hello>

Cookbook Run: Terminal Win1

```
markkean% rails new newsite
      create ...a whole bunch of crap....
markkean% cd newsite
markkean% ls
README      config          lib       script      vendor
Rakefile    db        log       test
markkean% rails generate controller test hello
      create app/controllers/test_controller.rb
      route get "test/hello"
      invoke erb
      create app/views/test
      create app/views/test/hello.html.erb
      invoke test_unit
      create test/functional/test_controller_test.rb

.....
markkean% rails server
=> Booting WEBrick
=> Rails 3.1.0 application starting in development on http://0.0.0.0:3000
=> Call with -d to detach
=> Ctrl-C to shutdown server
[2011-10-11 18:56:20] INFO  WEBrick 1.3.1
[2011-10-11 18:56:20] INFO  ruby 1.9.2 (2011-07-09) [x86_64-darwin11]
[2011-10-11 18:56:20] INFO  WEBrick::HTTPServer#start: pid=1581
port=3000
Started GET "/test/hello" for 127.0.0.1 at 2011-10-11 19:09:48 +0100
Processing by TestController#hello as HTML
Rendered test/hello.html.erb within layouts/application (25.4ms)
Compiled application.css (28ms) (pid 1581)
```

Cookbook Run

\$ rails new newsite

\$ cd newsite

\$ rails generate controller test hello

look at /newsite/app/controllers/
test_controller.rb

add **@message = “hello”** to hello method

look at newsite/app/views/test/hello.html.erb
add **<p> <%= @message %></p>** to file

\$ rails server

load page in browser:

Cookbook Run

\$ rails new newsite

\$ cd newsite

\$ rails generate controller test hello

look at /newsite/app/controllers/
test_controller.rb
add

look at newsite/app/views/test/hello.html.erb
add

\$ rails server

load page in browser:

\$ rake db:create

*is done first time
you use db*

\$ rake db:migrate

*is also done when
you have defined
model*

Cookbook Run: Rubymine I

The screenshot shows the Rubymine IDE interface with the following details:

- Project Bar:** Shows the project name "newsite" and the file path: `~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/newsite`.
- Toolbars:** Standard Java-like toolbars for file operations, navigation, and development.
- Project View (Left):** Shows the project structure under "newsite".
 - app:** Contains assets, controllers (with concerns), ApplicationController.rb, test_controller.rb, helpers (with application_helper.rb, test_helper.rb), mailers, models, views (with layouts), and a test folder containing hello.html.erb.
 - bin**
 - config:** Contains environments, initializers, locales (with application.rb, boot.rb, database.yml, environment.rb, routes.rb, secrets.yml), and db.
- Code Editor (Right):** Displays the file `test_controller.rb` with the following content:

```
class TestController < ApplicationController
  def hello
    @message = "hello"
  end
end
```
- Status Bar (Bottom):** Shows the file count (6), time (6:7), encoding (UTF-8), and other system icons.

A yellow highlight bar is present over the line containing the `end` keyword in the `TestController` class definition.

Cookbook Run:Xcode II

The screenshot shows the JetBrain RubyMine IDE interface. The title bar indicates the project is named 'newsite' and the current file is 'test/hello.html.erb'. The left sidebar displays the project structure:

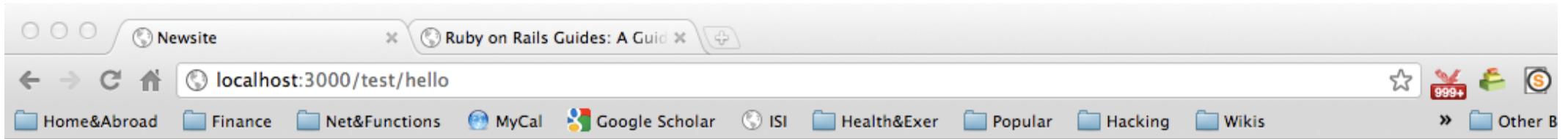
- Project /Users/user/Desktop/X_Teachin...
- 1: Project
- newsite (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/newsite)
 - app
 - assets
 - controllers
 - concerns
 - application_controller.rb
 - test_controller.rb
 - helpers
 - application_helper.rb
 - test_helper.rb
 - mailers
 - models
 - views
 - layouts
 - test
 - hello.html.erb
 - config
 - environments
 - initializers
 - locales
 - application.rb
 - boot.rb
 - database.yml
 - environment.rb
 - routes.rb
 - secrets.yml
 - db
 - 7: Structure

The code editor shows the content of 'hello.html.erb':

```
<h1>Test#hello</h1>
<p>Find me in app/views/test/hello.html.erb</p>

<p> <%= @message %></p>
```

Cookbook Run: Browser



Test#hello

Find me in `app/views/test/hello.html.erb`

hello

Controller

localhost/test/hello

where is the ?

render
hello.html.erb
with @message

Model

Views

Caveats

generally, each controller-method maps to a view file

we have not defined a layout (general frame of page)

we have not defined models

we have not looked at stylesheets

we have not looked at routing

but...we will

Part B2(i):

Walk-through of (less) simple Rails: The screens

A Banking Site

takes you through mortgage application

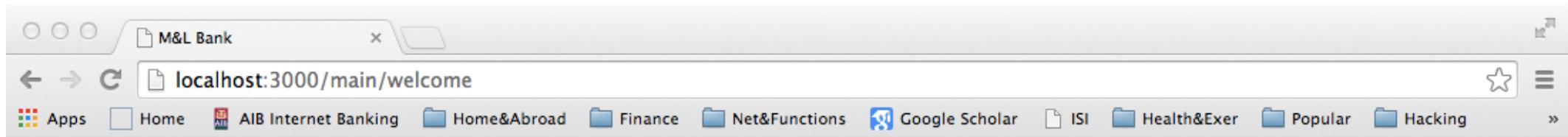
uses forms to take inputs

re-presents that information

does some minimal computation behind scenes

uses a db to store the interaction; that persists over different sessions with user

The Site: Screen I (start)



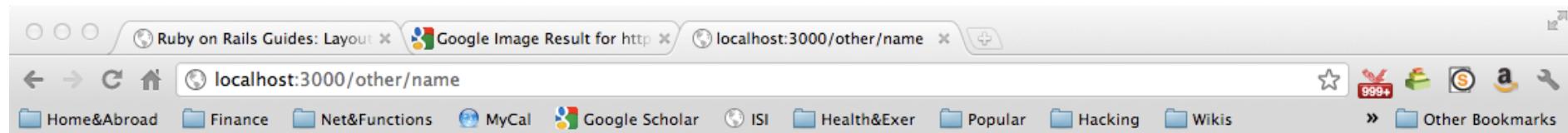
**Welcome to Monster & Leinster's
Home Page**



[click logo to enter]

Monster & Leinster Inc. © 2011

The Site: Screen Ila



The Monster & Leinster Bank



We are very pleased to welcome you our home mortgage page.

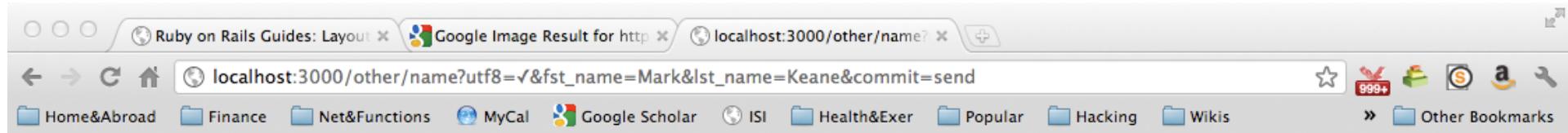
Please enter your first and last name below:

First Name:

Last Name:

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

The Site: Screen IIb



The Monster & Leinster Bank



We are very pleased to welcome you our home mortgage page.

Please enter your first and last name below:

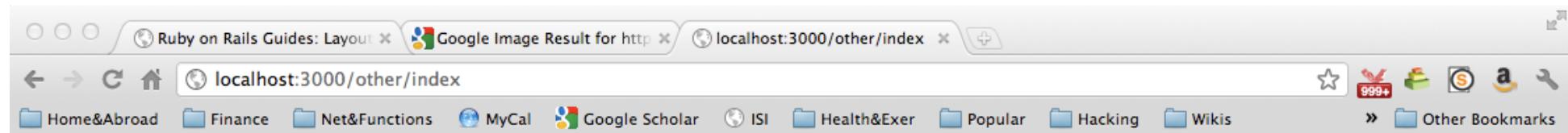
First Name:

Last Name:

Welcome Mark, let's go to the [next page](#).

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

The Site: Screen III



The Monster & Leinster Bank



Your Mortgage Details

So, Mark, let us get some more of your details:

First Name: Mark

Last Name: Keane

Address:

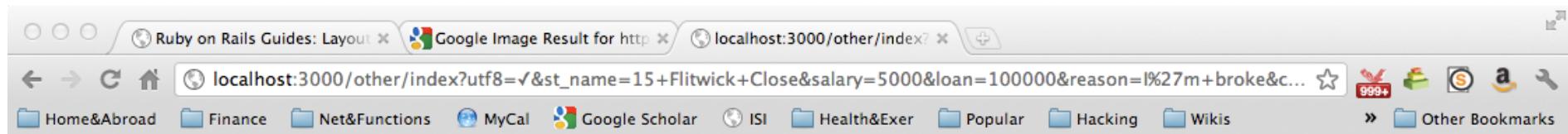
Salary:

Loan Amount:

Loan Reason:

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

The Site: Screen IV



The Monster & Leinster Bank



Your Mortgage Details

So, we have your details as:

First Name: Mark

Last Name: Keane

Address: 15 Flitwick Close

Salary: 5000

Loan: 100000

Reason: I'm broke

Your ID: 27

Do you want to [Ask For Quote](#) or [Go Back](#) .

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

The Site: Screen V



The Monster & Leinster Bank



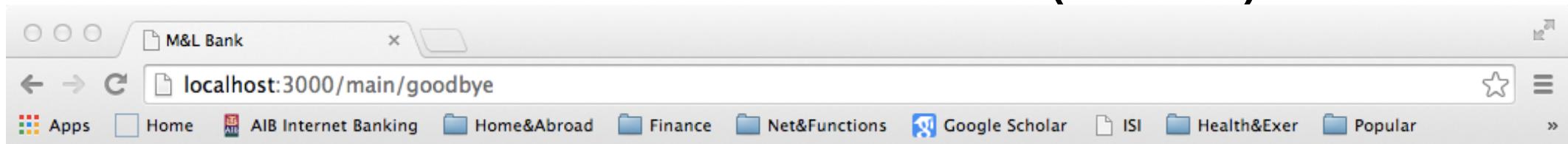
Your Quote

You goddam pauper, you asked us for a loan that is: more than three times your salary. Please go back and re-enter your correct salary.

Would you like to [Finish](#) or [Change Details](#)

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

The Site: Screen VI (end)



Goodbye to Monster & Leinster's End Page



[click logo to re-enter site]

- ,,,1
- mark, keane , ,2
- mark , keane , 2000, 3122 3
- mark , keane , ,4
- mark , keane , 12, 2000000 5
- ,,,6
- Mark, Keane , 5000, 100000 7
- ,,,8
- ,,,9
- Mark, Keane , 5000, 100000 10

Part B2(ii):

Walk-through of simple Rails: The Setup

Steps

\$ rails new banko

\$ cd banko

\$ rails generate controller main welcome goodbye

\$ rails generate controller other name index show
change quote

\$ rails generate model Entry

lets...edit a whole bunch of files...

\$ rails server

load page in browser:

<http://localhost:3000/main/welcome/>

\$ rake db:create

*is done first time
you use db*

\$ rake db:migrate

*is also done when
you have defined
model*

Steps

\$ rails new bank^{set up dir}

\$ cd bank go to dir create controller
 main with method **welcome**

\$ rails generate controller main welcome goodbye

\$ rails generate controller other name index show
change quote create controller
 other with method **name**

\$ rails generate model Entry
 create model
 called **entry**
lets...edit a whole bunch of files...

\$ rails server start server; you must be in the hello directory

load page in browser:

<http://localhost:3000/main/welcome/> call first page

Controller

Model

\$ rake db:migrate

Views

\$ rails server

What you get...

The screenshot shows the JetBrains RubyMine 3.2.4 IDE interface. The title bar indicates the project is named 'banko' and is located at `~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/banko`. The main window displays the code editor for `main_controller.rb`, which contains the following code:

```
class MainController < ApplicationController
  layout 'start_to_end'

  def welcome
    @message = "Welcome"
  end

  def goodbye
    @message = "Goodbye"
    @entries = Entry.all
  end
end
```

The code editor has syntax highlighting for Ruby and HTML. The file `main_controller.rb` is currently selected in the project tree. The project tree on the left shows the directory structure of the 'banko' application, including `app`, `assets`, `controllers`, `views`, and other standard Rails components. The `application.html.erb` file is also visible in the editor tab bar.

What you get...

The screenshot shows the JetBrains RubyMine 3.2.4 IDE interface. On the left is the Project tool window, which lists the project structure. In the center is the Editor tool window, showing the `main_controller.rb` file. On the right are the Data Sources and Structure tool windows.

Project Structure:

- Project: banko (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/banko)
- app
 - assets
 - images
 - javascripts
 - stylesheets
 - controllers
 - concerns
 - `application_controller.rb`
 - `main_controller.rb` (selected)
 - `other_controller.rb`
 - helpers
 - mailers
 - models
 - views
 - layouts
 - `application.html.erb`
 - `start_to_end.erb`
 - main
 - `goodbye.html.erb`
 - `welcome.html.erb` (circled with a red arrow)
 - other
- bin
- config
- db
- lib
- log
- public
- test
- vendor
 - .gitignore
 - config.ru
 - Gemfile
 - Gemfile.lock

for **welcome** method
in **main**

What you get...

The screenshot shows the JetBrains RubyMine 3.2.4 IDE interface. The title bar indicates the project is named 'banko' and the current file is 'main/welcome.html.erb'. The left sidebar displays the project structure under the 'Project' tab, showing the 'app' directory with 'assets', 'controllers', 'helpers', 'mailers', 'models', 'views' (containing 'layouts' and 'main'), and other files like 'application_controller.rb', 'main_controller.rb', and 'other_controller.rb'. Two specific files are circled with red ovals: 'main_controller.rb' and 'welcome.html.erb'. A large black arrow points from the text 'the main/welcome view' to the 'welcome.html.erb' file in the project tree. The right side of the interface shows the code editor with the contents of 'main/welcome.html.erb':

```
<h1><p align="center"> <%= h @message %> to Monster & Leinster's</h1>
<h1><p align="center"> Home Page</h1>
<p align="center"><%= link_to image_tag("monster.jpg"),
:controller => "other", :action => "name" %> </p>
<h5>          click logo to enter] </p></h5>
```

Below the code editor, the status bar shows the time as 8:57 and the encoding as UTF-8.

the **main/welcome**
view

What you get...

The screenshot shows the JetBrains RubyMine IDE interface. On the left is the Project Structure sidebar, which lists the project structure. In the center is the code editor showing the `main_controller.rb` file. On the right are tabs for `application.html.erb`, `application_controller.rb`, and `application.css`. A red arrow points from the `start_to_end` layout in the Project Structure to the corresponding line in the `main_controller.rb` file. Another red arrow points from the `start_to_end` layout in the Project Structure to the `start_to_end` layout file in the views/layouts/main directory.

```
class MainController < ApplicationController
  layout 'start_to_end'

  def welcome
    @message = "Welcome"
  end

  def goodbye
    @message = "Goodbye"
    @entries = Entry.all
  end
end
```

main layout

What you get...

The screenshot shows the JetBrains RubyMine 3.2.4 IDE interface. The project structure on the left is for a 'banko' application. The code editor on the right displays the 'start_to_end.erb' layout file.

```
<!DOCTYPE html>
<html>
<head>
  <title> M&L Bank </title>
</head>
<body>
<%= yield %>
<p align="center"> <font size="1"> Monster & Leinster Inc. &copy 2011 </font> </p>
</body>
</html>
```

start_to_end layout

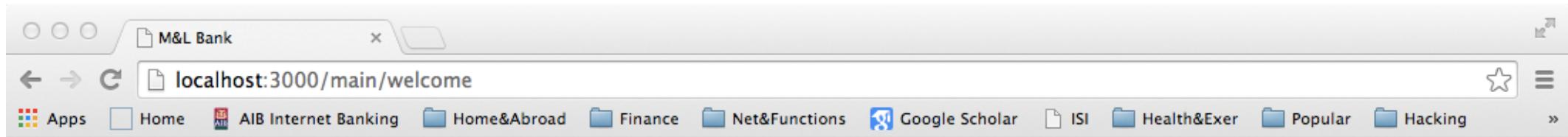
The 'start_to_end.erb' file contains the following code:

```
<!DOCTYPE html>
<html>
<head>
  <title> M&L Bank </title>
</head>
<body>
<%= yield %>
<p align="center"> <font size="1"> Monster & Leinster Inc. &copy 2011 </font> </p>
</body>
</html>
```

The 'main_controller.rb' file contains the following code:

```
class MainController < ApplicationController
  def welcome
    render :welcome
  end
end
```

The Site: Screen I



Welcome to Monster & Leinster's Home Page



[click logo to enter]

Monster & Leinster Inc. © 2011

REM

in simple terms, what was a single html file with code, gets broken up into 3-4 files, which combine to create a file that is sent to browser to be rendered

The diagram illustrates the MVC (Model-View-Controller) architecture for a web application, specifically focusing on the View layer.

start_to_end layout: A screenshot of a code editor showing the `start_to_end.erb` file. It contains an `<body>` section with a `<%= yield %>` placeholder. A red arrow points from this placeholder to the `main_controller.rb` file below, indicating that the controller's `welcome` method will be rendered here.

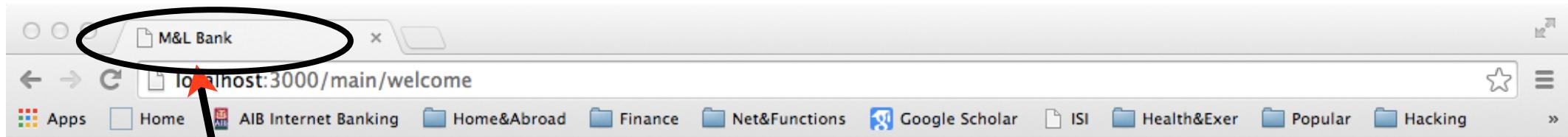
welcome content: A screenshot of a code editor showing the `main/welcome.html.erb` file. It contains various HTML elements and Ruby code. A red arrow points from the `<%= h @message %>` placeholder to the `main_controller.rb` file below, indicating that the controller's `welcome` method provides the message to be displayed.

welcome method: A screenshot of a code editor showing the `main_controller.rb` file. It defines a `welcome` method that sets the `@message` variable to "Welcome". Another red arrow points from this method to the `main/welcome.html.erb` file above, indicating the flow of data from the controller to the view.

main_controller.rb: A screenshot of a code editor showing the `main_controller.rb` file. It includes the definition of the `MainController` class, which inherits from `ApplicationController`, specifies a layout of `'start_to_end'`, and defines the `welcome` and `goodbye` methods.

...if you understand this you know how controllers work with views

The Site: Screen I



layout
bits

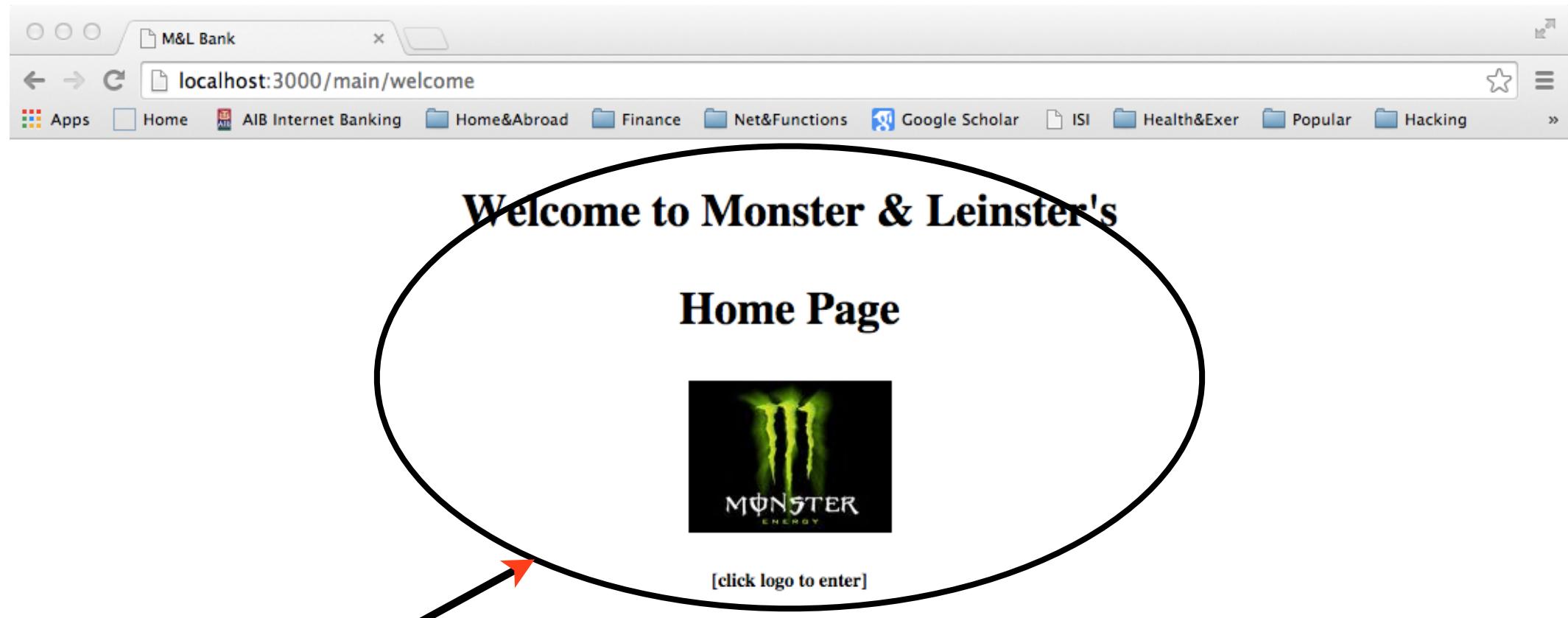
Welcome to Monster & Leinster's Home Page



[click logo to enter]

Monster & Leinster Inc. © 2011

The Site: Screen Begin



view
bits

The Site: Screen Begin

Screenshot of a web browser window showing the home page of "M&L Bank". The URL in the address bar is "localhost:3000/main/welcome". The page content includes a welcome message, a logo, and a copyright notice.

M&L Bank

localhost:3000/main/welcome

Apps Home AIB Internet Banking Home&Abroad Finance Net&Functions Google Scholar ISI Health&Exer Popular Hacking

Welcome to Monster & Leinster's
Home Page

MONSTER ENERGY

[click logo to enter]

Monster & Leinster Inc. © 2011

@message bit

A large black arrow points from the text "@message bit" to the word "Welcome" in the main heading. A red arrow points from the text "@message bit" to the green "MONSTER ENERGY" logo.

Part B2(iii):

Walk-through: Screen II-IV ~ Application Layout

Application Controller

localhost/any/action

render
application.html.erb
with application
spreadsheet

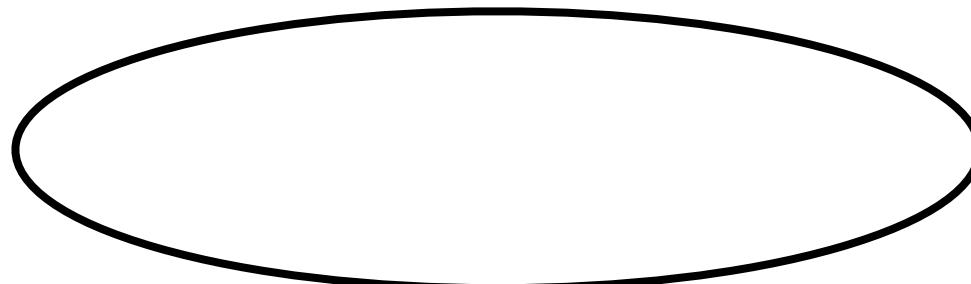
Model

Views

Layouts: Application



The Monster & Leinster Bank

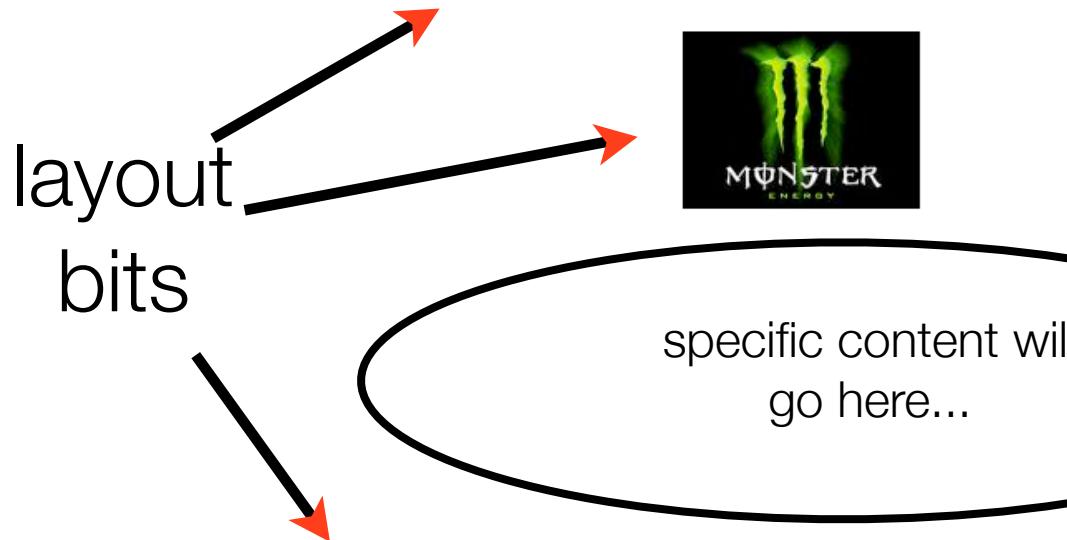


Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

Layouts: Application



The Monster & Leinster Bank

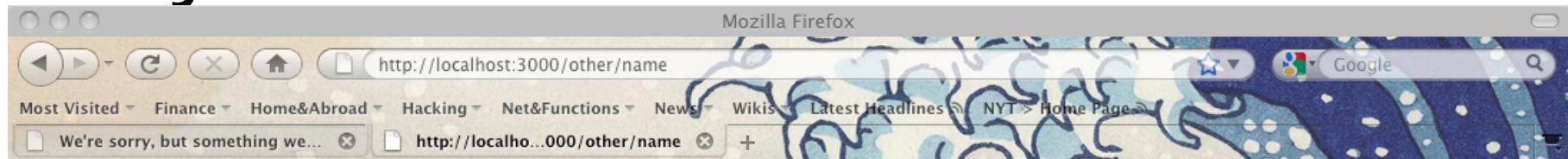


don't worry where
the image is for now

specific content will
go here...

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

Layouts: Frame



The Monster & Leinster Bank



...will come from different views

Monster & Leinster Inc accept no responsibility if you fall into arrears on your loan and are responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

hello/apps/views/layout/applications

Set Up Layout

for layout we edit:

app/controllers/ application_controller.rb

app/views/layout/application.html.erb

app/assets/stylesheets/application.css

for specific content, we edit:

app/controllers/other#name

app/views/other/name.html.erb

...

Set Up Layout

for frame we edit:

app/controllers/ application_controller.rb

which we can leave

app/views/layout/application.html.erb

app/assets/stylesheets/application.css

where we add the layout

for specific content, we edit:

where we define style

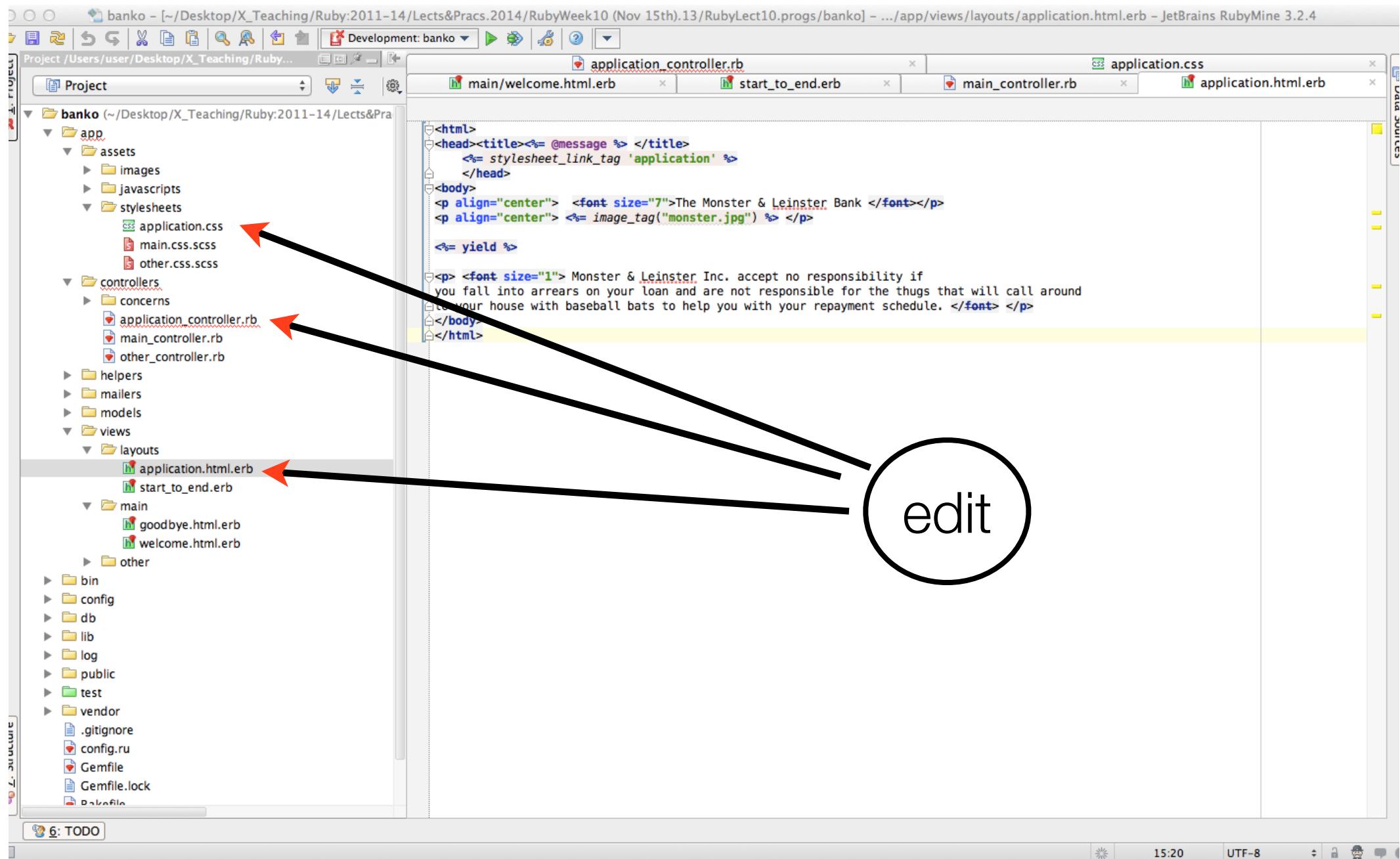
app/controllers/other#name

app/views/other/name.html.erb

just to show something

....

Steps to Set Up Layout





The Monster & Leinster Bank



Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

```
application_controller.rb:2  c class ApplicationController < ActionController::Base
class ApplicationController < ActionController::Base
  #protect_from_forgery
end
```

Done

controller
is empty



The Monster & Leinster Bank



```
application.html.erb:17
<html>
<head><title><%= @message %> </title>
  <%= stylesheet_link_tag 'application' %>
</head>
<body>
<p align="center"> <font size="7">The Monster & Leinster Bank </font></p>
<p align="center"> <%= image_tag("monster.jpg") %> </p>

<%= yield %>

<p> <font size="1"> Monster & Leinster Inc. accept no responsibility if
you fall into arrears on your loan and are not responsible for the thugs that will call around
to your house with baseball bats to help you with your repayment schedule. </font> </p>
</body>
</html>
```

Done

layout
has this



The Monster & Leinster Bank

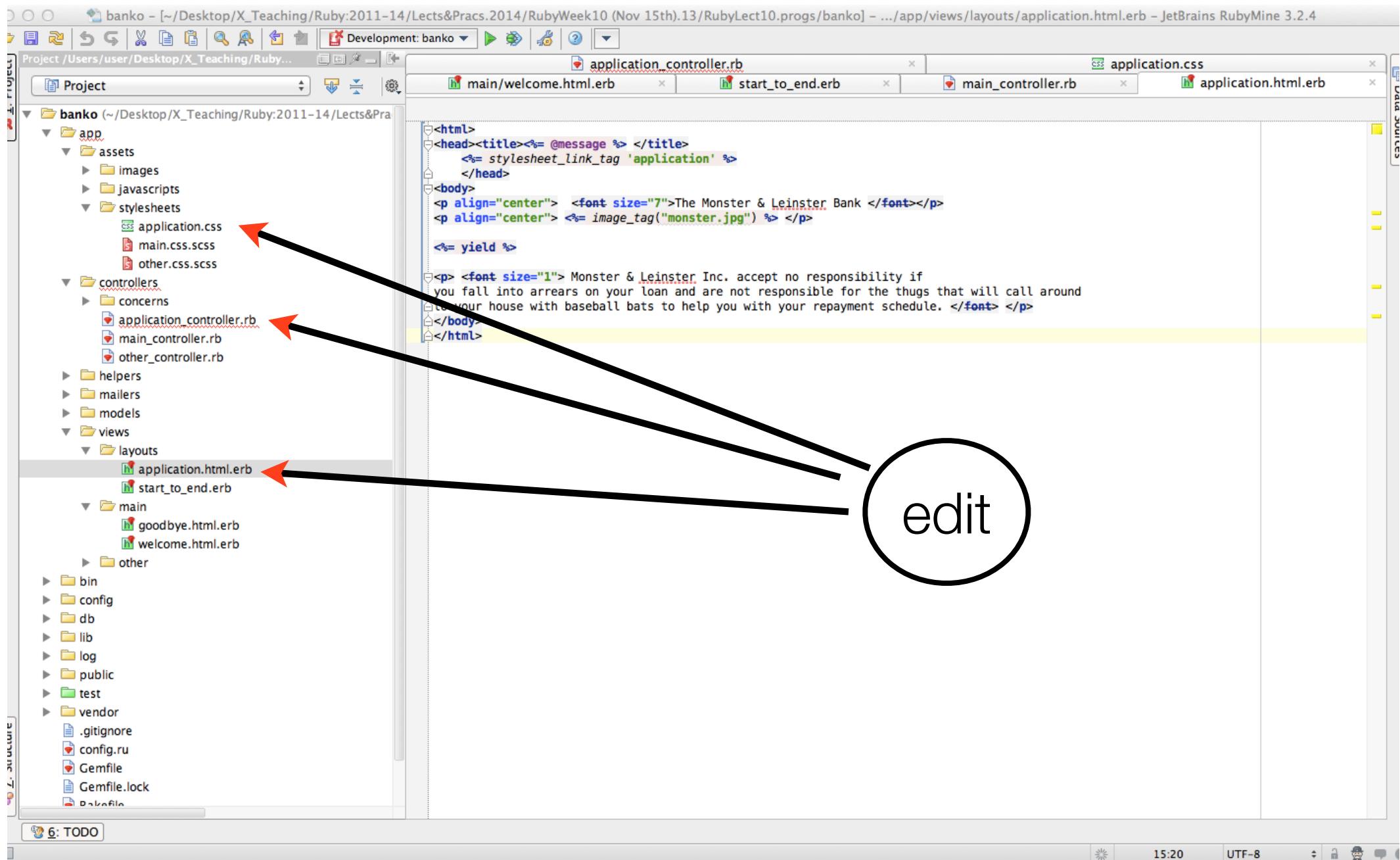


```
application.css:16 f h1 C
/*
 * This is a manifest file that'll automatically include all the stylesheets available in this
 * and any sub-directories. You're free to add application-wide styles to this file and they'll
 * be included on the top of the compiled file, but it's generally better to create a new file per style scope
 *= require_self
 *= require_tree .
*/
body {font-family:sans-serif;
}
h1 {font-family:times;
    font-size: 24pt;
    font-weight: bold;
    color:#FOO ;
}
```

Done

stylesheet
has this

Steps to Set Up Layout



edit

Specific Content

Screenshot of a web browser showing a custom view file for a name input form.

The browser window title bar shows:

- Ruby on Rails Guides: Layout
- Google Image Result for http
- localhost:3000/other/name

The address bar shows: localhost:3000/other/name

The bookmarks bar includes:

- Home&Abroad
- Finance
- Net&Functions
- MyCal
- Google Scholar
- ISI
- Health&Exer
- Popular
- Hacking
- Wikis

Other Bookmarks

The Monster & Leinster Bank

We are very pleased to welcome you our home mortgage page.

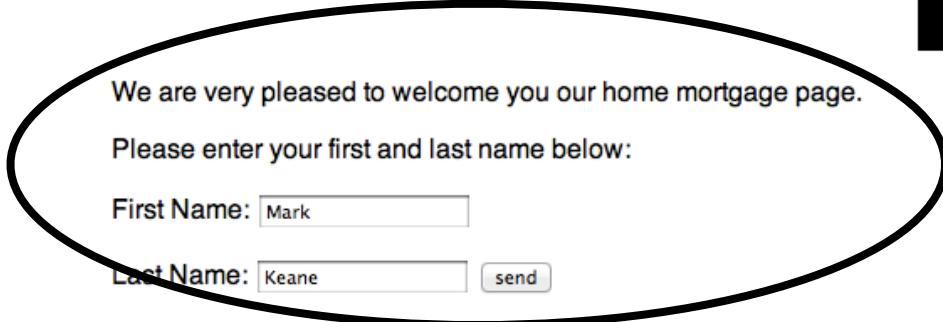
Please enter your first and last name below:

First Name:

Last Name:

from name view file

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.



Steps to Set Up Layout

The screenshot shows the JetBrains RubyMine IDE interface. The top bar displays the project name "banko" and the file path ".../app/views/other/name.html.erb". The toolbars and menu bar are visible at the top. Below the toolbar, the "Project" tool window shows the directory structure of the "banko" application, including the "app", "bin", "config", and "db" directories. The "app" directory contains "assets" (images, javascripts, stylesheets), "controllers" (concerns, application_controller.rb, main_controller.rb, other_controller.rb), "helpers", "mailers", "models", and "views" (layouts, main, other). The "views/layouts" folder contains "application.html.erb" and "start_to_end.erb". The "views/main" folder contains "goodbye.html.erb" and "welcome.html.erb". The "views/other" folder contains "change.html.erb", "index.html.erb", "name.html.erb" (which is currently selected), "quote.html.erb", and "show.html.erb". The bottom status bar shows "6: TODO", the time "15:10", and the encoding "UTF-8".

The code editor window displays the content of "name.html.erb". The code includes:

```
<p> We are very pleased to welcome you our home mortgage page. </p>
<p> Please enter your first and last name below: </p>

<form_tag("name", :method => 'get') do %>
  <p> First Name:
    <= text_field_tag 'fst_name' %>
  <p> Last Name:
    <= text_field_tag 'lst_name' %>
    <= submit_tag 'send' %>
<% end %>

<% if @fname then %>
  <p> Welcome <= h @entry.first_name %>, let's go to the
  <= link_to 'next page', :controller => "other", :action => "index" %>.</p>
<% end %>
```

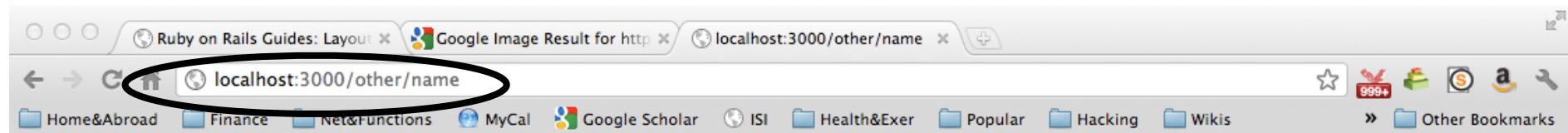
A yellow highlight covers the final line of the code: "<% end %>". To the right of the code editor, a large text block reads:

this goes into the yield
part of the layout

Part B3:

Looking at what is behind Screen II

The Site: Screen Ila



The Monster & Leinster Bank



We are very pleased to welcome you our home mortgage page.

Please enter your first and last name below:

First Name:

Last Name:

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

Other Controller...

```
def name
  @fname = params[:fst_name]
  @lname = params[:lst_name]
  @entry = Entry.create({:first_name => @fname, :last_name => @lname})
end

def index
  @person = Entry.last
  @fname = @person.first_name
  @lname = @person.last_name
  @person.update_attributes({:address => params[:st_name],
    :salary => params[:salary], :loan => params[:loan],
    :loan_reason => params[:reason]})  

  if !@person.address.nil? then render "show" end
end

def show
  @person = Entry.find(:last)
end

def change
  @person = Entry.last
  @fname = @person.first_name
  @lname = @person.last_name
  @entry = Entry.create({:first_name => @fname, :last_name => @lname, :salary => params[:salary], :loan => params[:loan]})  

end

def quote
  @person = Entry.last
  if !@person.salary.nil? then
    if (@person.salary * 3) < @person.loan
      then @message = "You goddam pauper, you asked us for a loan that is:  

        more than three times your salary. \n Please go back  

        and re-enter your correct salary."
      else @message = "Sure thing baby..."
      end
    else @message = "Seem to have an empty record??"
    end
  end
end
```

Other Controller...

The screenshot shows the JetBrains RubyMine IDE interface. The left sidebar displays the project structure under the 'Project' tab, showing the 'banko' project with its various files and folders. The main editor window shows the 'other_controller.rb' file, which contains the following code:

```
<p> We are very pleased to welcome you our home mortgage page. </p>
<p> Please enter your first and last name below: </p>

<=> form_tag("name", :method => 'get') do %>
  <p> First Name:<br/>
    <=> text_field_tag 'fst_name' %>
  <p> Last Name:<br/>
    <=> text_field_tag 'lst_name' %>
    <=> submit_tag 'send' %>
<% end %>

<=> if @fname then %>
  <p> Welcome <=> h @entry.first_name %>, let's go to the
    <=> link_to 'next page', :controller => "other", :action => "index" %>.</p>
<% end %>
```

The 'name.html.erb' file in the 'views/other' directory is also shown in the editor, indicating it is the template for the controller. A large black oval highlights the entire code block in the controller file, and a smaller red arrow points from the 'name.html.erb' file in the project tree to the highlighted code.

REM

in simple terms, what was a single html file with code, gets broken up into 3 files, which combine to create a file that is sent to browser to be rendered

application layout

```
<html>
<head><title><%= @message %> </title>
<%= stylesheet_link_tag 'application' %>
</head>
<body>
<p align="center"> <font size="7">The Monster & Leinster Bank </font></p>
<p align="center"> <%= image_tag("monster.jpg") %> </p>

<%= yield %>

<p> <font size="1"> Monster & Leinster Inc. accept no responsibility if
you fall into arrears on your loan and are not responsible for the thugs that will call around
to your house with baseball bats to help you with your repayment schedule. </font> </p>
</body>
</html>
```

name content

```
<p> We are very pleased to welcome you our home mortgage page. </p>
<p> Please enter your first and last name below: </p>

<%= form_tag("name", :method => 'get') do %>
  <p> First Name:<br/>
    <%= text_field_tag 'fst_name' %>
  <p> Last Name:<br/>
    <%= text_field_tag 'lst_name' %>
    <%= submit_tag 'send' %>
<% end %>

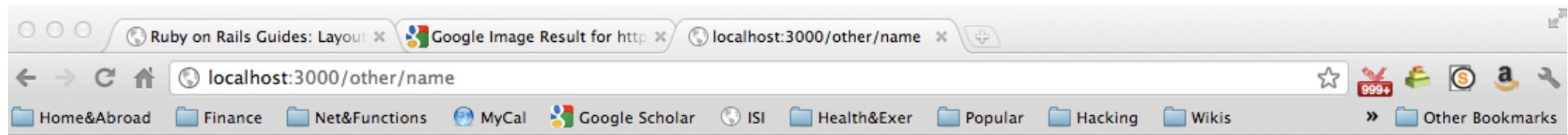
<% if @fname then %>
  <p> Welcome <%= h @entry.first_name %>, let's go to the
  <%= link_to 'next page', :controller => "other", :action => "index" %>.</p>
<% end %>
```

name method

```
class OtherController < ApplicationController

  def name
    @fname = params[:fst_name]
    @lname = params[:lst_name]
    @entry = Entry.create(:first_name => @fname, :l
```

The Site: Screen Ia



The Monster & Leinster Bank



We are very pleased to welcome you our home mortgage page.

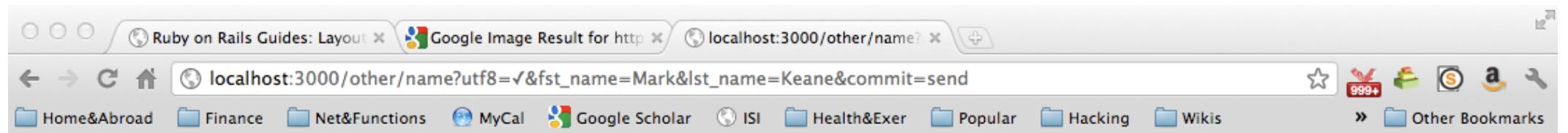
Please enter your first and last name below:

First Name:

Last Name:

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

The Site: Screen IIb



The Monster & Leinster Bank



We are very pleased to welcome you our home mortgage page.

Please enter your first and last name below:

First Name:

Last Name:

Welcome Mark, let's go to the [next page](#).

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible

```
name.html.erb:8
<p> We are very pleased to welcome you our home mortgage page. </p>
<p> Please enter your first and last name below: </p>

<%= form_tag("name", :method => 'get') do %>
  <p> First Name:<br/>
    <%= text_field_tag 'fst_name' %>
  <p> Last Name:<br/>
    <%= text_field_tag 'lst_name' %>
    <%= submit_tag 'send' %>
<% end %>

<% if @fname then %>
  <p> Welcome <%= h @entry.first_name %>, let's go to the
    <%= link_to 'next page', :controller => "other", :action => "index" %>.</p>
<% end %>
```

Pause...Paws...Pose...

Ok, so, you need to digest all of this
we have covered how you set up the initial files
how you edit those files, how controllers, layouts and
views combine to produce the final page
...the one remaining big thing is to talk about models
but...before we do that we'll back up into layouts...

REM:

MVC framework

model-view-controller is a software architecture/architectural pattern used in Rails

model is domain-specified rep of the data used by the application; manages data and informs of changes

view renders model into suitable form for interactions (at interface); may have multiple views of same model

controller receives input and initiates responds making calls to model data (operational guts of the app)

Controller

Model

\$ rake db:migrate

Views

\$ rails server

Part C:

A note Rails on functionality...in the foregoing...

Neat Stuff Ia

Controllers are just Ruby:

layout is a new primitive

nb inheritance from libraries

```
other_controller.rb:13  class OtherController < ApplicationController
class OtherController < ApplicationController

def name
  @fname = params[:fst_name]
  @lname = params[:lst_name]
  @entry = Entry.create({:first_name => @fname, :last_name => @lname})
end

end
```

Neat Stuff I^b

```
other_controller.rb:13  class OtherController < ApplicationController
class OtherController < ApplicationController

    db access
    def name
        @entries = Entry.find(:all)           db change
        @entries.each {|entry| entry.destroy}
        @fname = params[:fst_name]           db item create
        @lname = params[:lst_name]
        @entry = Entry.create({:first_name => @fname, :last_name => @lname})
        @entries = Entry.find(:all)
    end
    db access
end
```

oval refers to access to tags in the form
used; need to look in views

Embedded Ruby

We are very pleased to welcome you our home mortgage page.

Please enter your first and last name below:

First Name:

Last Name:

```
name.html.erb:8
<p> We are very pleased to welcome you our home mortgage page. </p>
<p> Please enter your first and last name below: </p>

<%= form_tag("name", :method => 'get') do %>
  <p> First Name:
    <%= text_field_tag 'fst_name' %>
  <p> Last Name:
    <%= text_field_tag 'lst_name' %>
    <%= submit_tag 'send' %>
<% end %>

<% if @fname then %>
  <p> Welcome <%= h @entry.first_name %>, let's go to the
  <%= link_to 'next page', :controller => "other", :action => "index" %>.</p>
<% end %>
```

ERb stuff

ERb arbitarily embedded within html

sets up form

ERb is inside <% or <%= and %>

Part D:

But, Mark, what about de models...

Recall Banko...

we just discussed controllers, views and layouts

but, we said nothing about models

yet, we did issue a command originally...

and, performed some edits...

Controller

Model

\$ rake db:migrate

Views

\$ rails server

Steps

REM:

set up dir

\$ rails new banko
go to dir

\$ cd bank

\$ rails generate controller main welcome

\$ rails generate controller other name

\$ rails generate model Entry first_name:string ...

lets...edit a whole bunch of files...

\$ rake db:migrate

set up tables

\$ rake db:rollback

rollback to early migration of db

create model
called **entry**
with field:type

Countries: Define tables REM:

```
ActiveRecord::Schema.define do
  create_table :countries do |table|
    table.column :name, :string
    table.column :continent, :string
    table.column :size, :integer
  end

  create_table :regions do |table|
    table.column :country_id, :integer
    table.column :region_size, :integer
    table.column :name, :string
  end
end
...

```

neatdb.rb

two tables

region references the country it is part of

Rails
use ActiveRecord
via Rake



Countries: Create associations **REM:**

```
class Country < ActiveRecord::Base  
  has_many :regions  
end
```

```
class Region < ActiveRecord::Base  
  belongs_to :country  
end  
...
```

neatdb.rb

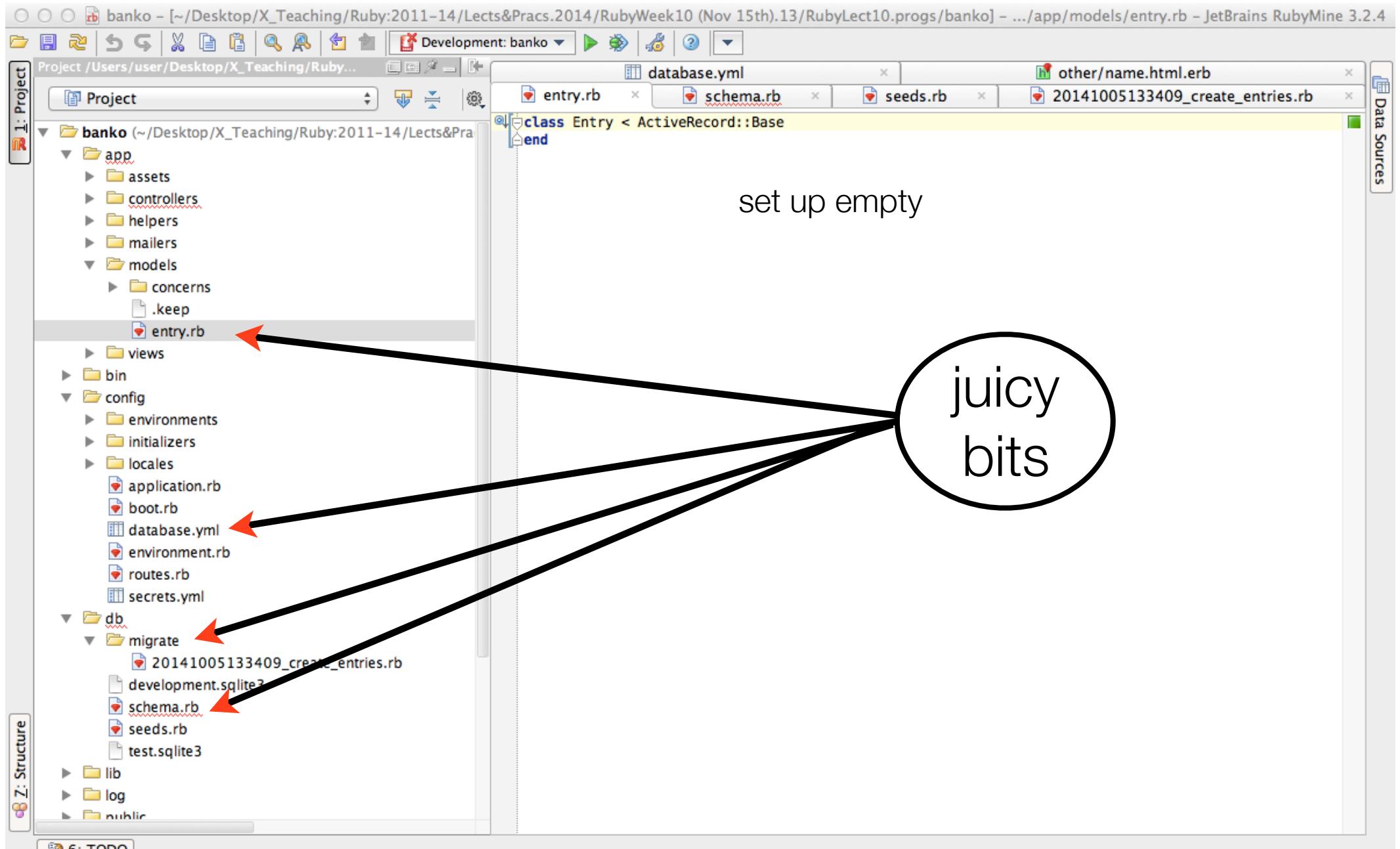
Country and **Region** are subclasses of ActiveRecord

describes relationships between models (tables)

here, **has_many** and **belongs_to** define a 1:n



generate model Entry...does



generate model Entry...does

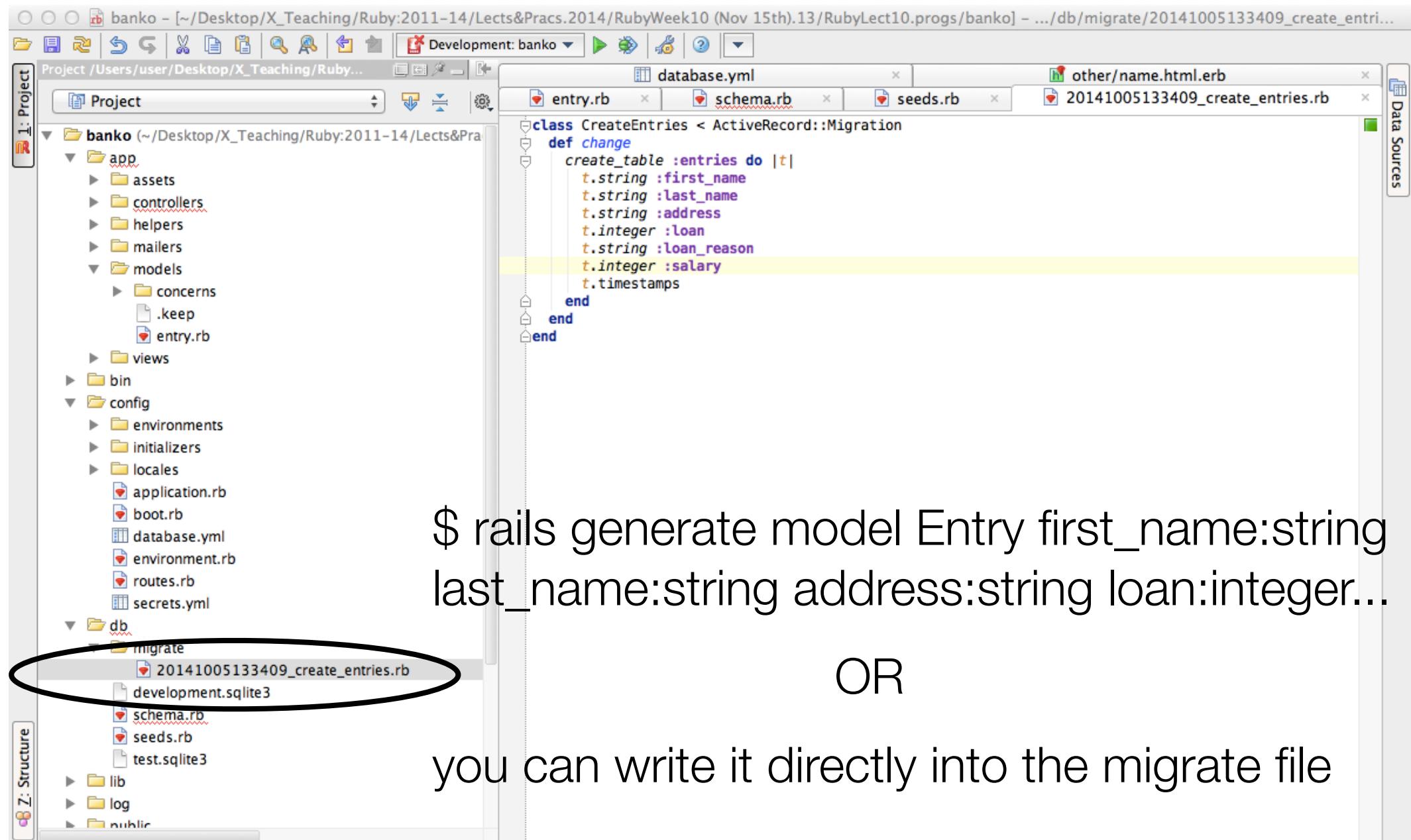
The screenshot shows the JetBrains RubyMine 3.2.4 IDE interface. The left side features a project tree titled 'Project' with the path 'banko - [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/banko]'. The 'app' directory is expanded, showing 'assets', 'controllers', 'helpers', 'mailers', 'models', 'views', 'bin', 'config', 'db', and 'public'. The 'models' folder contains 'concerns' and 'entry.rb', with 'entry.rb' highlighted and circled in black. The right side shows a code editor with tabs for 'database.yml', 'entry.rb', 'schema.rb', 'seeds.rb', and '20141005133409_create_entries.rb'. The 'entry.rb' tab is active, displaying the following code:

```
class Entry < ActiveRecord::Base
end
```

To the right of the code editor, there is a block of text:

model **Entry**
where you put the
has_many and
belongs_to stuff

generate model Entry...does



The screenshot shows a Java IDE (IntelliJ IDEA) interface with a Rails project named "banko". The left sidebar displays the project structure, including app, config, db, and lib folders. The db/migrate folder contains several files, with "20141005133409_create_entries.rb" highlighted and circled in black.

The main editor window shows the content of "20141005133409_create_entries.rb":

```
class CreateEntries < ActiveRecord::Migration
  def change
    create_table :entries do |t|
      t.string :first_name
      t.string :last_name
      t.string :address
      t.integer :loan
      t.string :loan_reason
      t.integer :salary
      t.timestamps
    end
  end
end
```

Below the code, there is a command-line prompt:

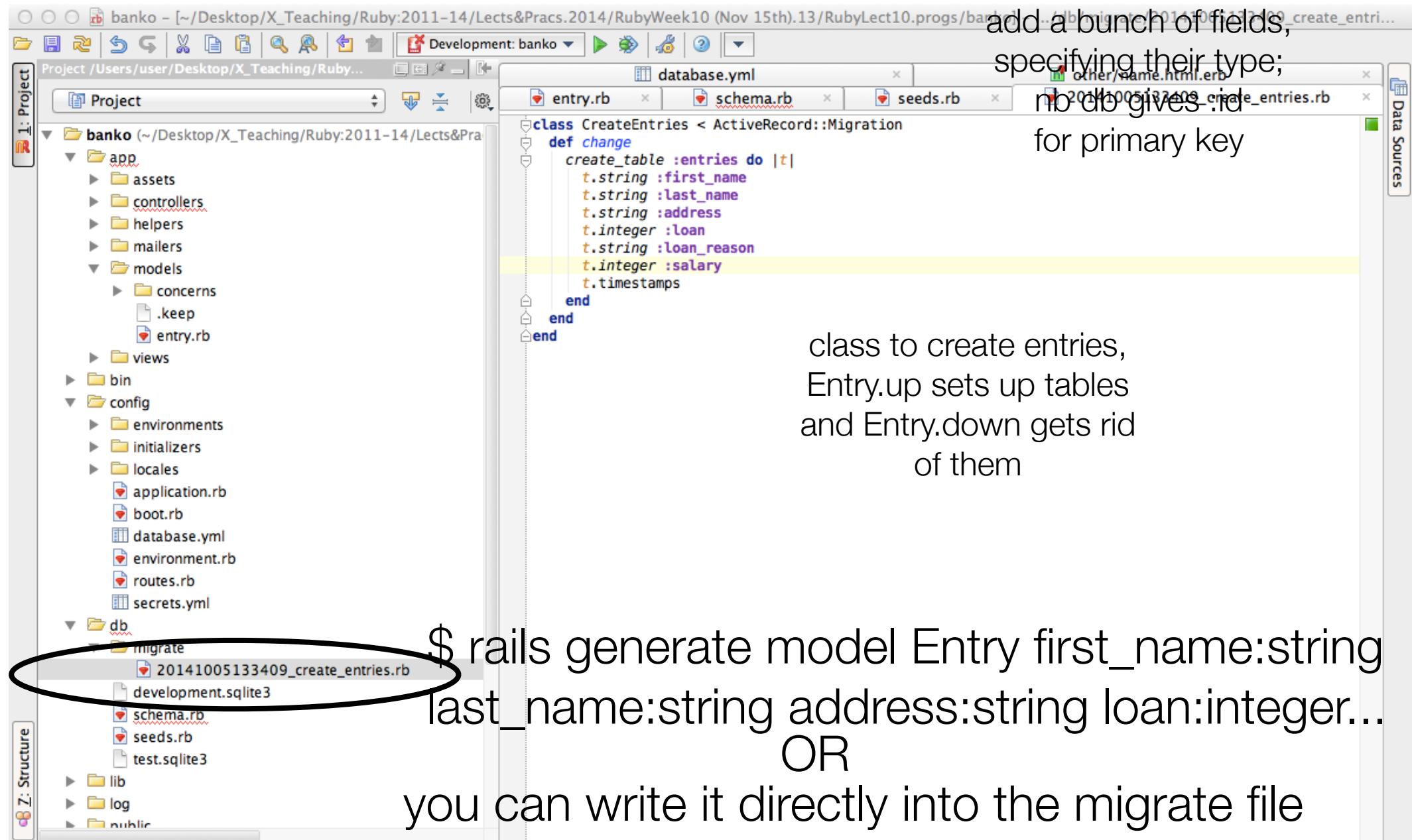
```
$ rails generate model Entry first_name:string  
last_name:string address:string loan:integer...
```

Following the command-line prompt is the word "OR", indicating an alternative method for generating the model.

At the bottom, another text line states:

you can write it directly into the migrate file

generate model Entry...does

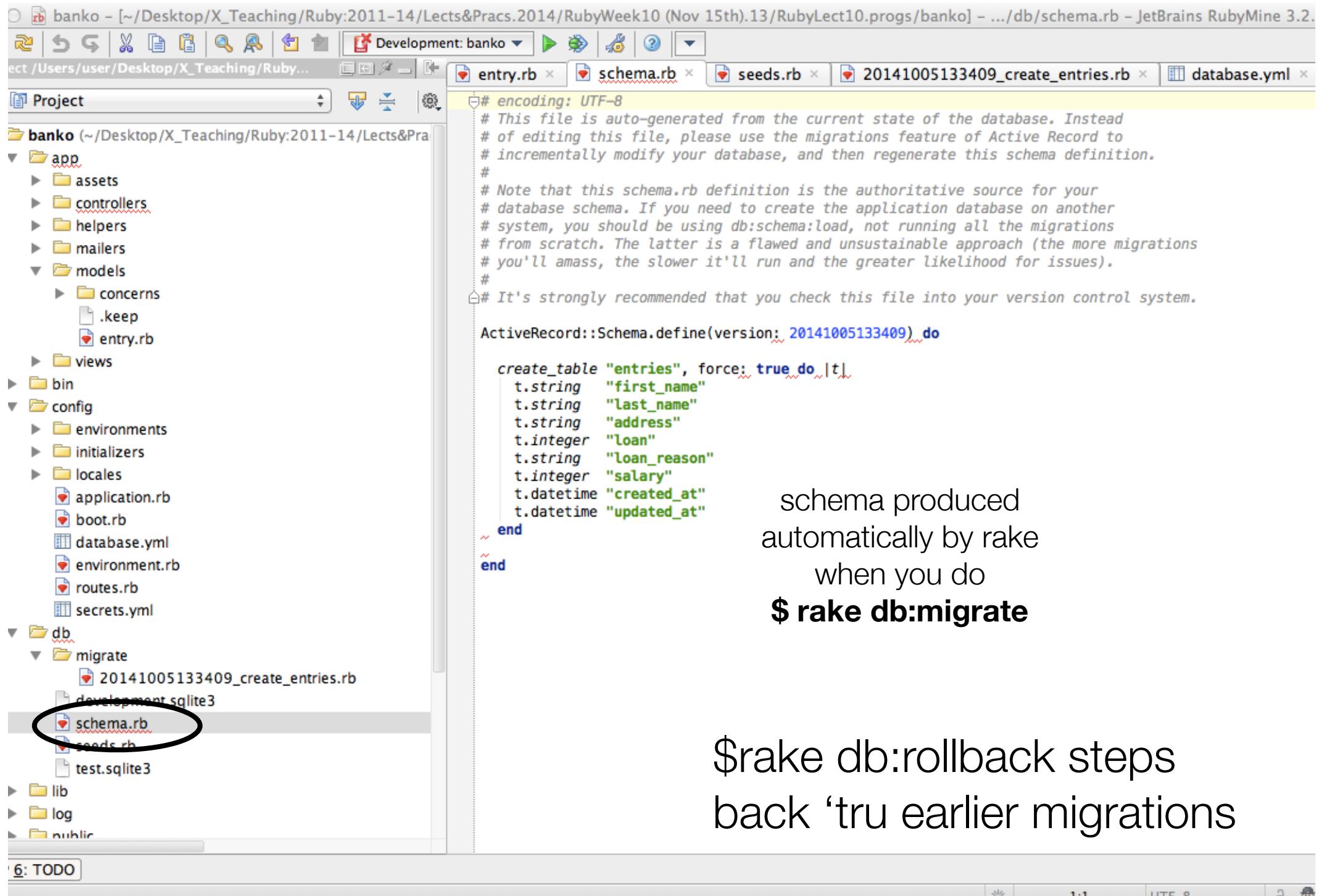


The screenshot shows a Java IDE (IntelliJ IDEA) interface with a Rails project named "banko". The project structure on the left includes app, config, db, and lib folders. The db/migrate folder contains several files: development.sqlite3, schema.rb, seeds.rb, test.sqlite3, and 20141005133409_create_entries.rb. The 20141005133409_create_entries.rb file is open in the editor, showing the following code:

```
class CreateEntries < ActiveRecord::Migration
  def change
    create_table :entries do |t|
      t.string :first_name
      t.string :last_name
      t.string :address
      t.integer :loan
      t.string :loan_reason
      t.integer :salary
      t.timestamps
    end
  end
end
```

Annotations on the right side of the screen provide additional context:

- A callout points to the file 20141005133409_create_entries.rb with the text: "\$ rails generate model Entry first_name:string last_name:string address:string loan:integer..."
- A callout points to the code in the migration file with the text: "add a bunch of fields, specifying their type; nb do gives :id for primary key"
- A callout points to the class definition in the migration file with the text: "class to create entries, Entry.up sets up tables and Entry.down gets rid of them"
- A callout points to the "OR" text at the bottom with the text: "you can write it directly into the migrate file"



The screenshot shows the JetBrains RubyMine 3.2 IDE interface. The project tree on the left shows a directory structure for a Ruby application named 'banko'. The 'db' folder contains a 'migrate' folder with files like '20141005133409_create_entries.rb', 'development.sqlite3', and 'schema.rb'. The 'schema.rb' file is circled in red in the project tree. The main editor window displays the contents of the 'schema.rb' file:

```
# 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: 20141005133409) do

  create_table "entries", force: true do |t|
    t.string   "first_name"
    t.string   "last_name"
    t.string   "address"
    t.integer  "loan"
    t.string   "loan_reason"
    t.integer  "salary"
    t.datetime "created_at"
    t.datetime "updated_at"
  end
end
```

On the right side of the editor, there is a text block with the following content:

schema produced
automatically by rake
when you do
\$ rake db:migrate

At the bottom right, there is another text block with the following content:

\$rake db:rollback steps
back 'tru earlier migrations'

banko - [/Users/user/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/banko] - .../config/database.yml - JetBrains RubyMine 3.2

Project /Users/user/Desktop/X_Teaching/Ruby... entry.rb schema.rb seeds.rb 20141005133409_create_entries.rb database.yml

entry.rb x schema.rb x seeds.rb x 20141005133409_create_entries.rb x database.yml x

Project

banko (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pra...

app assets controllers helpers mailers models concerns .keep entry.rb views bin config environments initializers locales application.rb bootstrap database.yml environment.rb routes.rb secrets.yml db migrate 20141005133409_create_entries.rb development.sqlite3 schema.rb seeds.rb test.sqlite3 lib log public

```
# 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
```

config file for database

The screenshot shows the JetBrains RubyMine 3.2.4 IDE interface. The title bar indicates the project is named 'banko'. The left sidebar displays the project structure under 'banko' (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/banko). The 'app' directory contains 'assets', 'controllers', 'helpers', 'mailers', 'models' (with 'concerns' and 'entry.rb'), 'views', and 'bin'. The 'config' directory contains 'environments', 'initializers', 'locales' (with 'application.rb', 'boot.rb', 'database.yml', 'environment.rb', 'routes.rb', and 'secrets.yml'), and 'db' (with 'migrate' containing '20141005133409_create_entries.rb', 'development.sqlite3', 'schema.rb', and 'seeds.rb'). A red oval highlights the 'schema.rb' and 'seeds.rb' files in the 'migrate' folder. The right panel shows the 'seeds.rb' file content:

```
# 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)
```

A text overlay on the right side of the screen reads: "can introduce seed records here".

Steps

REM:

set up dir

\$ rails new bank
go to dir

\$ cd banco

\$ rails generate controller main welcome

\$ rails generate controller other name

\$ rails generate model Entry first_name:string ...

lets...edit a whole bunch of files...

\$ rake db:migrate

set up tables

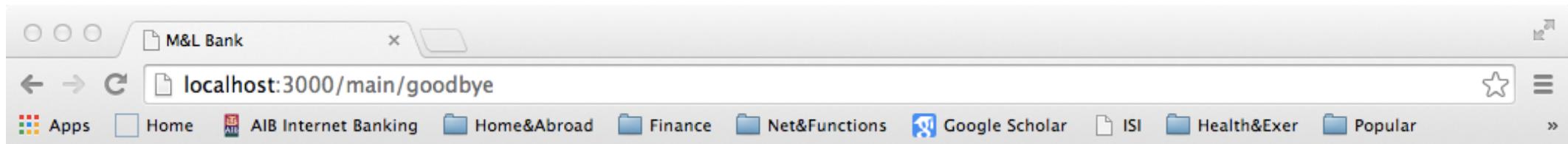
\$ rake db:rollback

rollback to early stage of db

create model
called **entry**
with field:type

The Site: Screen VI

REM:



what's
this !

Goodbye to Monster & Leinster's End Page



[click logo to re-enter site]

- ,,, 1
- mark, keane , , 2
- mark , keane , 2000, 3122 3
- mark , keane , , 4
- mark , keane , 12, 2000000 5
- ,,, 6
- Mark, Keane , 5000, 100000 7
- ,,, 8
- ,,, 9
- Mark, Keane , 5000, 100000 10



banko - [/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/banko] - .../app/views/main/goodbye.html.erb - Jet

ect /Users/user/Desktop... Development: banko Project database.yml

banko (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/banko) entry.rb schema.rb seeds.rb main/goodbye.html.erb

Project

banko (~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/banko)

app

- assets
- controllers
 - concerns
 - application_controller.rb
 - main_controller.rb
 - other_controller.rb
- helpers
- mailers
- models
- views
 - layouts
 - main
 - goodbye.html.erb
 - welcome.html.erb
 - other
- bin
- config
- db
- lib
- log
- public
- test
- vendor
 - .gitignore
 - config.ru
 - Gemfile
 - Gemfile.lock
 - Rakefile
 - README.rdoc

External Libraries

6: TODO

20141005133409_create_entries.rb

```
<h1><p align="center"> <%= h @message %> to Monster & Leinster's</h1>  
  
<h1><p align="center"> End Page</h1>  
  
<p align="center"><%= link_to image_tag("monster.jpg"),  
:controller => "main", :action => "welcome" %> </p>  
  
<h5> <p align="center"> [click logo to re-enter site] </p></h5>  
  
<ul>  
  <% @entries.each do |entry| %>  
    <li> <%=h entry.first_name %>, <%=h entry.last_name %> , <%=h entry.salary %>, <%=h entry.loan %> <%=h entry.id %>  
  <% end %>  
</ul>
```

A large black oval highlights the section of code within the `` tag that iterates over `@entries` and prints each entry's details.

Problem Banko...

this, finally, clears up the mystery of what all the crap on the goodbye page is about...

however, it also shows that something odd is going on in our db

why are there two entries, some of which have no values? What's the other crap about ?

...a question not to be asked...

Part E:

Some Residuals in Banko... controllers, routing & images

Basic Banko Functionality...

note, that each screen has an associated, controller-method, view and layout

controllers/main_controller.rb has two methods
welcome and **goodbye**

controllers/other_controller.rb has four methods
index, **name**, **show**, **change**, and **quote**

there are all fairly simple...its the co-ordination that is complicated

The Main Controller

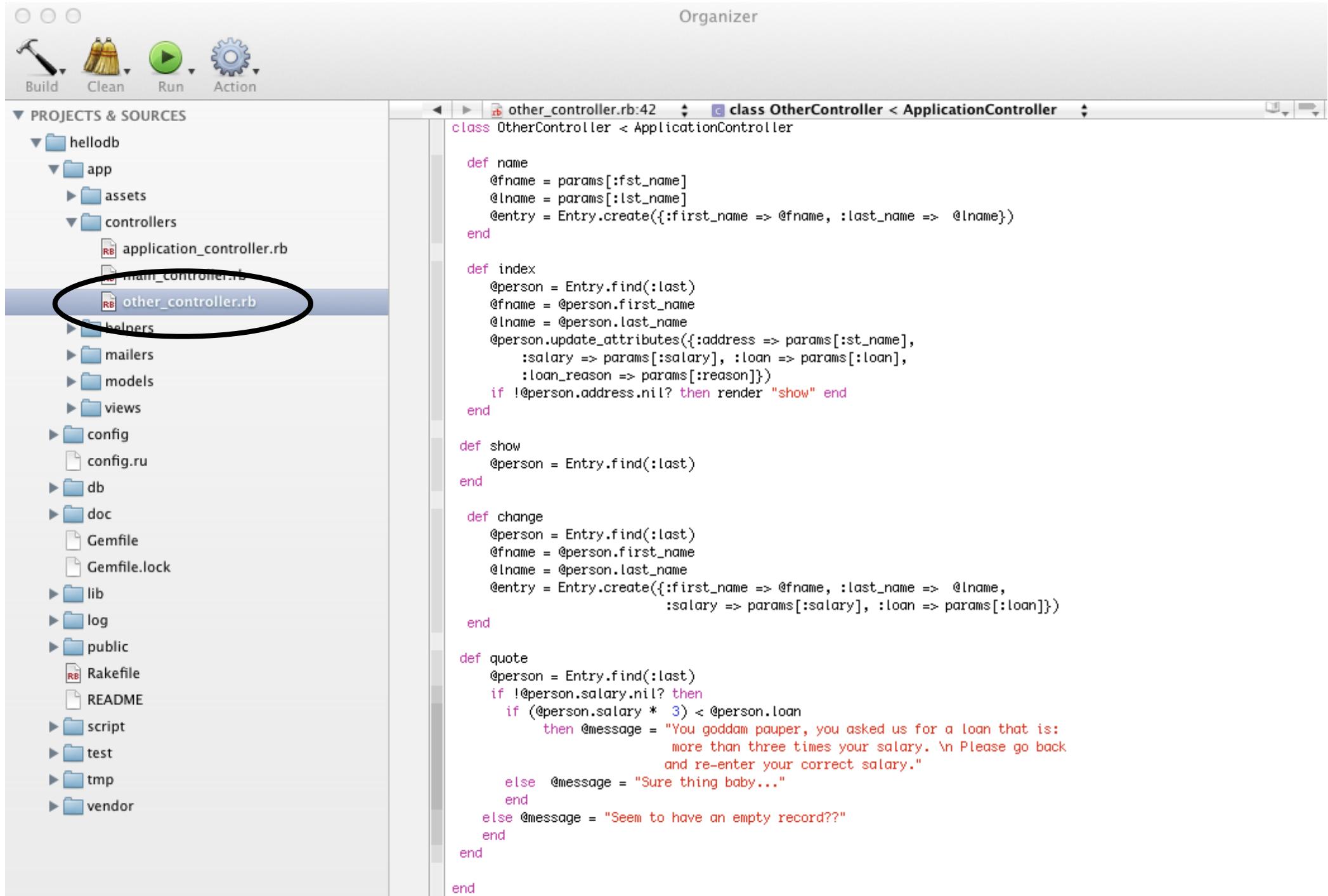
A screenshot of a Java IDE (JetBrains) showing the code for the `main_controller.rb` file. The project structure on the left shows a folder named `bankodb [banko]` containing an `app` directory with files like `assets`, `controllers`, `concerns`, `application_controller.rb`, `main_controller.rb` (which is circled in black), and `other_controller.rb`. The code editor on the right contains the following Ruby code:

```
class MainController < ApplicationController
  layout 'start_to_end'

  def welcome
    @message = "Welcome to DB version"
    @entries = Entry.all
    @entries.each { |entry| entry.destroy }
  end

  def goodbye
    @message = "Goodbye to the Db version"
    @entries = Entry.all
  end
end
```

The Other Controller



The screenshot shows a software interface for managing a Ruby on Rails project. The top menu bar includes 'Organizer' and icons for 'Build', 'Clean', 'Run', and 'Action'. The left sidebar, titled 'PROJECTS & SOURCES', lists the project structure: 'hellodb' (with 'app', 'assets', 'controllers', 'helpers', 'mailers', 'models', 'views', 'config', 'db', 'doc', 'lib', 'log', 'public', 'script', 'test', 'tmp', 'vendor'), 'Gemfile', 'Gemfile.lock', 'Rakefile', 'README', and files like 'application_controller.rb', 'main_controller.rb', and 'other_controller.rb'. The 'other_controller.rb' file is highlighted with a black oval. The main pane displays the code for the 'OtherController' class, which inherits from ' ApplicationController'. It contains methods for 'name', 'index', 'show', 'change', 'quote', and 'end'. The 'index' method updates attributes for a person entry based on parameters. The 'change' method creates a new entry with salary and loan attributes. The 'quote' method checks if the loan is more than three times the salary and provides a message accordingly.

```
class OtherController < ApplicationController
  def name
    @fname = params[:fst_name]
    @lname = params[:lst_name]
    @entry = Entry.create({:first_name => @fname, :last_name => @lname})
  end

  def index
    @person = Entry.find(:last)
    @fname = @person.first_name
    @lname = @person.last_name
    @person.update_attributes({:address => params[:st_name],
      :salary => params[:salary], :loan => params[:loan],
      :loan_reason => params[:reason]})
    if !@person.address.nil? then render "show" end
  end

  def show
    @person = Entry.find(:last)
  end

  def change
    @person = Entry.find(:last)
    @fname = @person.first_name
    @lname = @person.last_name
    @entry = Entry.create({:first_name => @fname, :last_name => @lname,
      :salary => params[:salary], :loan => params[:loan] })
  end

  def quote
    @person = Entry.find(:last)
    if !@person.salary.nil? then
      if (@person.salary * 3) < @person.loan
        then @message = "You goddam pauper, you asked us for a loan that is:
          more than three times your salary. \n Please go back
          and re-enter your correct salary."
      else @message = "Sure thing baby..."
      end
    else @message = "Seem to have an empty record??"
    end
  end
end
```

Banko Routing...

Rails style is sort-of weird in the way it creates objects and calls methods

essentially, methods run by linking to pages with uri-s; when a page is entered the method runs

action tags in forms also call methods (e.g.,
submit_tag; they are usually helper methods)

obviously, methods call other methods too (more conventionally)

Banko Routing...

this means that routing becomes very important, as it is a sort-of shorthand for calling methods

will return to it in detail, but just note it here

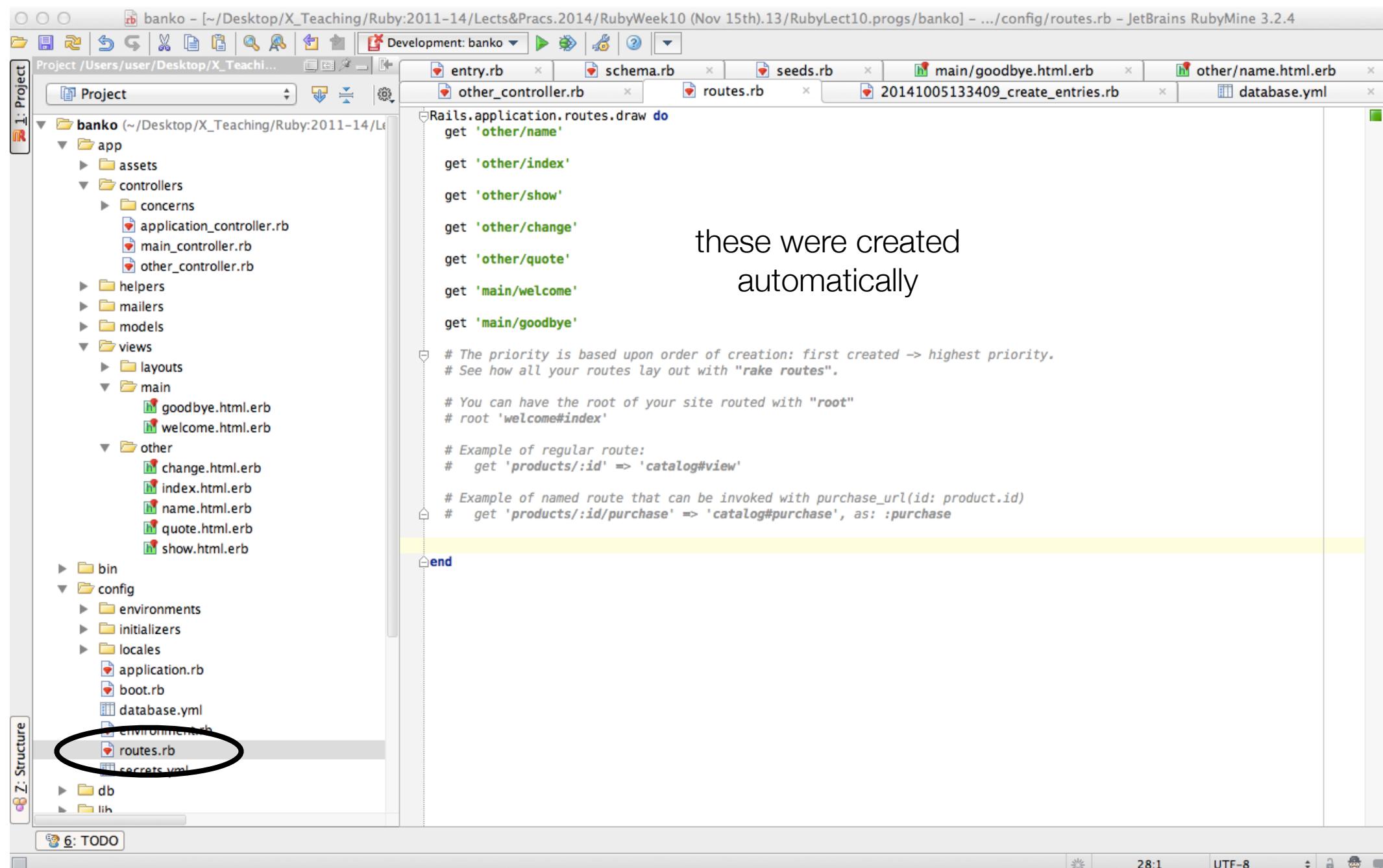
btw, its is often a source of error in the browser, as you fail to address the page with the right uri

what is the “right uri” will be set by stuff in banko/config/routes.rb



will
cause
bugs

routes.rb on setup...



these were created automatically

```
Rails.application.routes.draw do
  get 'other/name'

  get 'other/index'

  get 'other/show'

  get 'other/change'

  get 'other/quote'

  get 'main/welcome'

  get 'main/goodbye'

  # The priority is based upon order of creation: first created -> highest priority.
  # See how all your routes lay out with "rake routes".

  # You can have the root of your site routed with "root"
  # root 'welcome#index'

  # Example of regular route:
  #   get 'products/:id' => 'catalog#view'

  # Example of named route that can be invoked with purchase_url(id: product.id)
  #   get 'products/:id/purchase' => 'catalog#purchase', as: :purchase
end
```

Z: Structure

6: TODO

Banko Routing...

one of the things you can do is change the routing statements in this routes.rb file

often people want to have the browser default connect to the first page in your site

not like this...

A screenshot of a Mac OS X desktop showing a web browser window. The address bar displays 'localhost:3000'. A black oval highlights this address bar. The browser's menu bar shows 'ActionView::Helpers::FormTa' and 'TeachingPipe'. The toolbar includes standard icons for back, forward, search, and refresh. Below the toolbar is a horizontal bar with various links: Home&Abroad, Finance, Net&Functions, MyCal, Google Scholar, ISI, Health&Exer, Popular, Hacking, and Wikis. On the far right of the toolbar is a red badge with '999+'. The main content area of the browser shows the 'Welcome aboard' page from a Ruby on Rails application. It features the Ruby logo, the text 'Welcome aboard' and 'You're riding Ruby on Rails!', and a link to 'About your application's environment'. A 'Getting started' section provides instructions: 1. Use rails generate to create your models and controllers, with a note about running without parameters; 2. Set up a default route and remove public/index.html, with a note about config/routes.rb; 3. Create your database, with a note about config/database.yml. To the right of the main content is a sidebar titled 'Browse the documentation' containing links to Rails Guides, Rails API, Ruby core, and Ruby standard library.

Welcome aboard

You're riding Ruby on Rails!

[About your application's environment](#)

Getting started

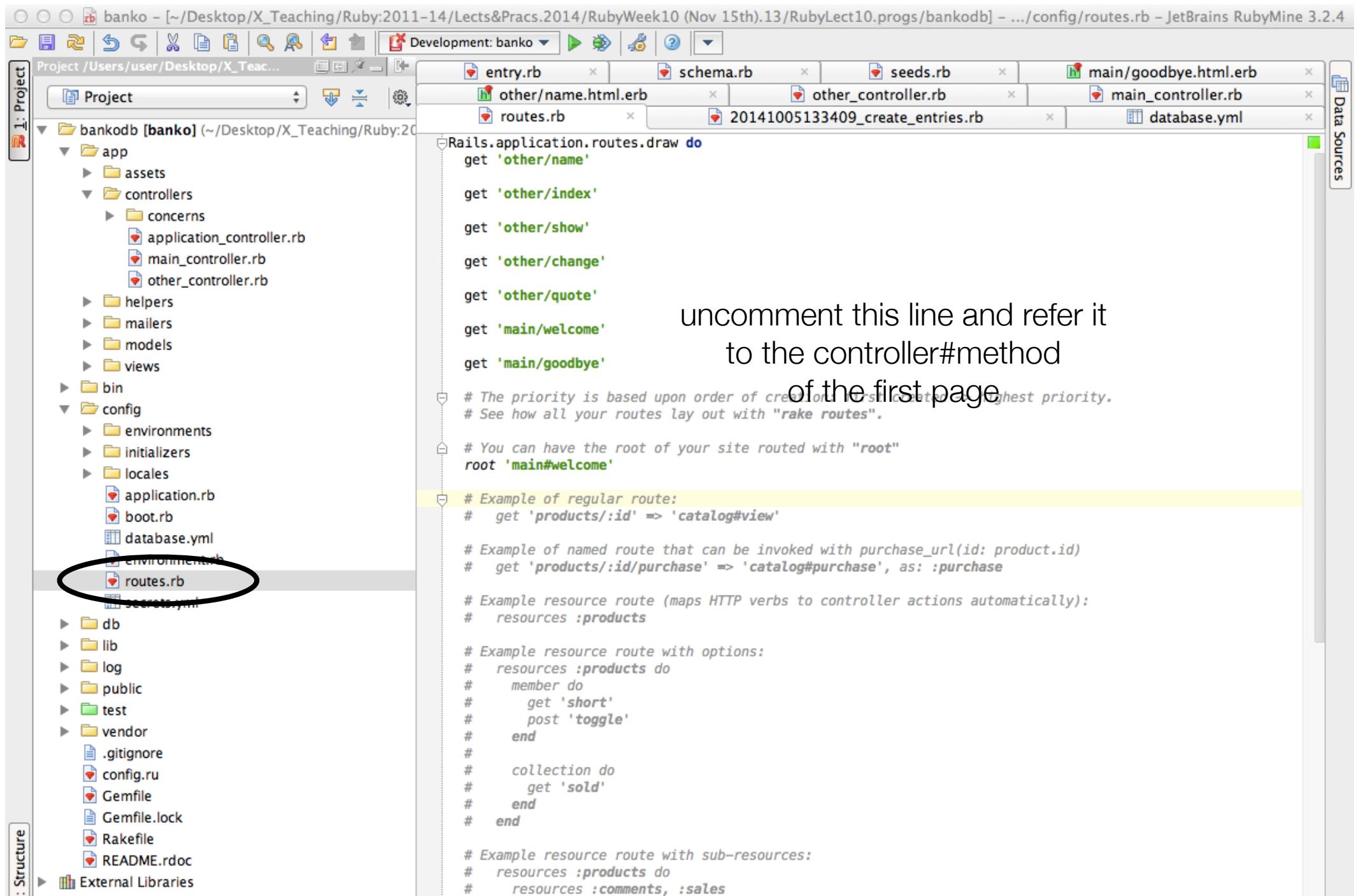
Here's how to get rolling:

1. Use `rails generate` to create your models and controllers
To see all available options, run it without parameters.
2. Set up a default route and remove `public/index.html`
Routes are set up in `config/routes.rb`.
3. Create your database
Run `rake db:create` to create your database. If you're not using SQLite (the default), edit `config/database.yml` with your username and password.

Browse the documentation

[Rails Guides](#)
[Rails API](#)
[Ruby core](#)
[Ruby standard library](#)

routes.rb...what we do



The screenshot shows the JetBrains RubyMine 3.2.4 IDE interface. The title bar indicates the project is named 'banko' and the current file is 'config/routes.rb'. The left sidebar shows the project structure with a folder 'bankodb [banko]' containing 'app', 'config', and other files. The 'routes.rb' file is highlighted with a black oval. The main editor area displays the following code:

```
Rails.application.routes.draw do
  get 'other/name'
  get 'other/index'
  get 'other/show'
  get 'other/change'
  get 'other/quote'
  get 'main/welcome'
  get 'main/goodbye'

  # The priority is based upon order of creation: first created → highest priority.
  # See how all your routes lay out with "rake routes".
  #
  # You can have the root of your site routed with "root"
  root 'main#welcome'

  # Example of regular route:
  #   get 'products/:id' => 'catalog#view'

  # Example of named route that can be invoked with purchase_url(id: product.id)
  #   get 'products/:id/purchase' => 'catalog#purchase', as: :purchase

  # Example resource route (maps HTTP verbs to controller actions automatically):
  #   resources :products

  # Example resource route with options:
  #   resources :products do
  #     member do
  #       get 'short'
  #       post 'toggle'
  #     end
  #
  #     collection do
  #       get 'sold'
  #     end
  #   end

  # Example resource route with sub-resources:
  #   resources :products do
  #     resources :comments, :sales
  #   end
  #
  #   resources :posts, :tags
end
```

A callout bubble points to the line 'root 'main#welcome'' with the text: 'uncomment this line and refer it to the controller#method of the first page'.

so, we now get this...

A screenshot of a Mac OS X desktop environment showing a web browser window. The browser has three tabs open: 'ActionView::Helpers::FormTa' (partially visible), 'TeachingPipe' (active tab), and 'M&L Bank'. The address bar shows 'localhost:3000' with a magnifying glass icon. Below the address bar is a toolbar with various icons for 'Home&Abroad', 'Finance', 'Net&Functions', 'MyCal', 'Google Scholar', 'ISI', 'Health&Exer', 'Popular', 'Hacking', and 'Wikis'. A status bar at the bottom right shows '999+'.

**Welcome to DB version to Monster & Leinster's
Home Page**

The logo consists of a black square containing a bright green, glowing 'MM' monogram. Below the monogram, the word 'MONSTER' is written in a bold, white, sans-serif font, with 'ENERGY' in smaller letters underneath.

[click logo to enter]

Monster & Leinster Inc. © 2011

Instead of this...

A screenshot of a Mac OS X desktop showing a web browser window. The address bar displays 'localhost:3000'. The browser has three tabs open: 'ActionView::Helpers::FormTa' (partially visible), 'TeachingPipe' (partially visible), and 'Ruby on Rails: Welcome aboard' (the active tab). The main content area shows the 'Welcome aboard' page for a Ruby on Rails application. The page features a red 'RAILS' logo, the text 'Welcome aboard' and 'You're riding Ruby on Rails!', and a link to 'About your application's environment'. Below this is a 'Getting started' section with three numbered steps: 1. 'Use rails generate to create your models and controllers', 2. 'Set up a default route and remove public/index.html', and 3. 'Create your database'. The 'About your application's environment' link is underlined. To the right of the main content is a sidebar with the heading 'Browse the documentation' and links to 'Rails Guides', 'Rails API', 'Ruby core', and 'Ruby standard library'. The browser's menu bar and various icons are visible at the top.

localhost:3000

ActionView::Helpers::FormTa × TeachingPipe × Ruby on Rails: Welcome aboard ×

Home&Abroad Finance Net&Functions MyCal Google Scholar ISI Health&Exer Popular Hacking Wikis

999+

Welcome aboard

You're riding Ruby on Rails!

[About your application's environment](#)

Getting started

Here's how to get rolling:

1. Use `rails generate` to create your models and controllers

To see all available options, run it without parameters.

2. Set up a default route and remove `public/index.html`

Routes are set up in `config/routes.rb`.

3. Create your database

Run `rake db:create` to create your database. If you're not using SQLite (the default), edit `config/database.yml` with your username and password.

Browse the documentation

[Rails Guides](#)
[Rails API](#)
[Ruby core](#)
[Ruby standard library](#)

BY THE WAY..

user\$ rake routes

Prefix	Verb	URI Pattern	Controller#Action
other_name	GET	/other/name(.:format)	other#name
other_index	GET	/other/index(.:format)	other#index
other_show	GET	/other/show(.:format)	other#show
other_change	GET	/other/change(.:format)	other#change
other_quote	GET	/other/quote(.:format)	other#quote
main_welcome	GET	/main/welcome(.:format)	main#welcome
main_goodbye	GET	/main/goodbye(.:format)	main#goodbye
root	GET	/	main#welcome

user\$

Where are Images ?



Rails stores images in bankodb/app/assets/images

they are then referenced in various view and layout files



Where are images...

Screenshot of the JetBrains RubyMine IDE showing a project structure and file viewer.

The Project tool window shows the following structure:

- banko [~/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/bankodb]
- app
 - assets
 - images
 - face.jpg
 - monster.jpg
 - javascripts
 - stylesheets
 - controllers
 - concerns
 - application_controller.rb
 - main_controller.rb
 - other_controller.rb
 - helpers
 - mailers
 - models
 - views
- bin
- config
- db
- lib
- log
- public
- test
- vendor
 - .gitignore
 - config.ru
 - Gemfile
 - Gemfile.lock
 - Rakefile
 - README.rdoc
- External Libraries

The "monster.jpg" file is selected in the Project tool window and highlighted with a red oval. The file viewer on the right displays the image, which is the iconic green "M" logo of Monster Energy drink.

File viewer details:
face.jpg, routes.rb, monster.jpg, environment.rb
143x107 JPEG (24-bit color) 2.54Kb

Layouts: Frame



The Monster & Leinster Bank



Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

Where are images...ref'ed

```
application.html.erb:17
<html>
<head><title><%= @message %> </title>
  <%= stylesheet_link_tag 'application' %>
</head>
<body>
<p align="center"> <font size="7">The Monster & Leinster Bank </font></p>
<p align="center"> <%= image_tag("monster.jpg") %> </p>
<%= yield %>

<p> <font size="1"> Monster & Leinster Inc. accept no responsibility if
you fall into arrears on your loan and are not responsible for the thugs that will call around
to your house with baseball bats to help you with your repayment schedule. </font> </p>
</body>
</html>
```

Part F:

Fixing Banko's Problems...

Problem Banko...

REM:

if you look at the goodbye page, which prints out
the state of the db at the end; it progressively
seems to spit out loads of crap

why do we have these odd entries...

The Problem



Goodbye to Monster & Leinster's

End Page



[click logo to re-enter site]

- , , , 1
- Mark, Keane , 2000, 1000000 2
- , , , 3
- mark, keane , 1000, 100000 4
- , , , 5
- Sal, Paradiso , 30000, 1000 6

The Problem starts here...

A screenshot of a web browser window. The address bar shows the URL `localhost:3000/other/name`. The page content is a forged home mortgage application. It features a large title "The Monster & Leinster Bank" and the Monster Energy logo. Below the logo, the text reads: "We are very pleased to welcome you our home mortgage page. Please enter your first and last name below:". There are two input fields: "First Name: ". The "Last Name:" field contains "Keane" and has a "send" button next to it. At the bottom, a small disclaimer states: "Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule."

Ruby on Rails Guides: Layout × Google Image Result for http × localhost:3000/other/name ×

localhost:3000/other/name

Home&Abroad Finance Net&Functions MyCal Google Scholar ISI Health&Exer Popular Hacking Wikis Other Bookmarks

The Monster & Leinster Bank



We are very pleased to welcome you our home mortgage page.

Please enter your first and last name below:

First Name:

Last Name: send

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

...look at the web server logs...

Log...

Started GET "/other/name" for 127.0.0.1 at 2011-10-18 12:36:09 +0100

Processing by OtherController#name as HTML

SQL (139.9ms) INSERT INTO "entries" ("address", "created_at", "first_name", "last_name", "loan", "loan_reason", "salary", "updated_at") VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?) [[{"address": nil}, {"created_at": "Tue, 18 Oct 2011 11:36:10 UTC +00:00"}, {"first_name": nil}, {"last_name": nil}, {"loan": nil}, {"loan_reason": nil}, {"salary": nil}, {"updated_at": "Tue, 18 Oct 2011 11:36:10 UTC +00:00"}]]

Rendered other/name.html.erb within layouts/application (20.0ms)

Completed 200 OK in 445ms (Views: 159.8ms | ActiveRecord: 140.9ms)

...

Started GET "/other/name?utf8=%E2%9C%93&fst_name=Mark&lst_name=Keane&commit=send" for 127.0.0.1 at 2011-10-18 12:36:21 +0100

Processing by OtherController#name as HTML

Parameters: {"utf8"=>"✓", "fst_name"=>"Mark", "lst_name"=>"Keane", "commit"=>"send"}

SQL (1.5ms) INSERT INTO "entries" ("address", "created_at", "first_name", "last_name", "loan", "loan_reason", "salary", "updated_at") VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?) [[{"address": nil}, {"created_at": "Tue, 18 Oct 2011 11:36:21 UTC +00:00"}, {"first_name": "Mark"}, {"last_name": "Keane"}, {"loan": nil}, {"loan_reason": nil}, {"salary": nil}, {"updated_at": "Tue, 18 Oct 2011 11:36:21 UTC +00:00"}]]

Rendered other/name.html.erb within layouts/application (1.3ms)

Completed 200 OK in 66ms (Views: 7.4ms | ActiveRecord: 2.2ms)

What's the problem...

Well, remember what I said about the way Rails works by calling methods when you move pages

If you look at the log, you can see that it creates one entry when you enter the name-page and another when you press the **send** tag

So, how can we fix this ?

other/name View...

The screenshot shows the JetBrains RubyMine IDE interface. The title bar indicates the project is named 'banko' and the current file is 'other/name.html.erb'. The left sidebar displays the project structure under 'Project' mode, showing the 'app' directory with its subfolders: assets, controllers, helpers, mailers, models, and views. The 'views' folder contains 'layouts', 'main', and 'other'. The 'other' folder is currently selected, and 'name.html.erb' is highlighted. The main editor area shows the content of 'name.html.erb':

```
<p> We are very pleased to welcome you our home mortgage page. </p>
<p> Please enter your first and last name below: </p>

<form_tag("name", :method => 'get') do %>
  <p> First Name:
    <= text_field_tag 'fst_name' %>
  <p> Last Name:
    <= text_field_tag 'lst_name' %>
    <= submit_tag 'send' %>
<% end %>

<% if @fname then %>
  <p> Welcome <= h @entry.first_name %>, let's go to the
  <= link_to 'next page', :controller => "other", :action => "index" %>.</p>
<% end %>
```

The code uses ERB syntax for generating HTML. It includes a form for entering a first and last name, and a conditional block that displays a welcome message and a link to the next page if a variable @fname is present.

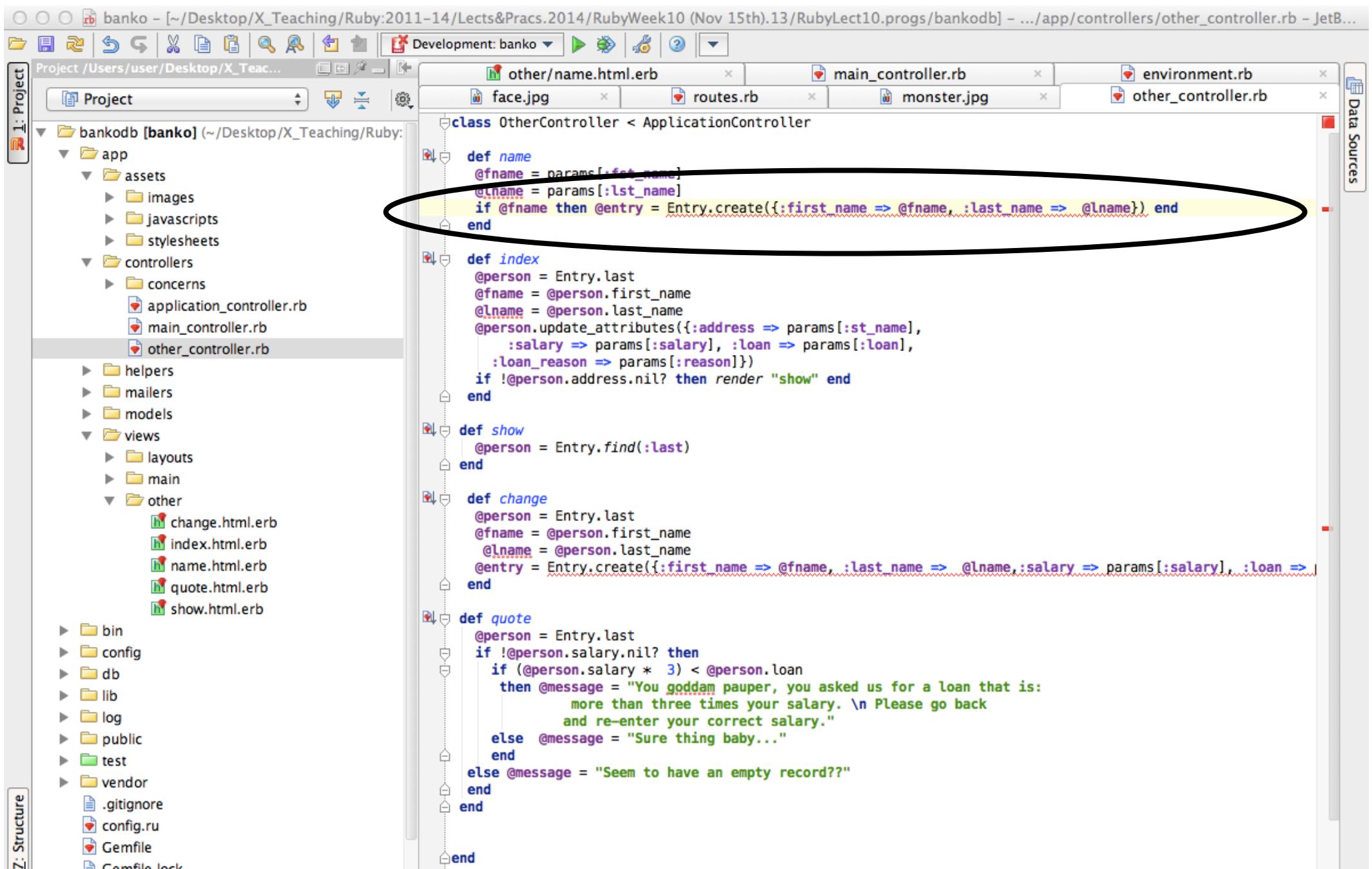
What's the problem...

Well, remember what I said about the way Rails works by calling methods when you move pages

If you look at the log, you can see that it creates one entry when you enter the name-page and another when you press the **send** tag

So, how can we fix this ?

The Change...

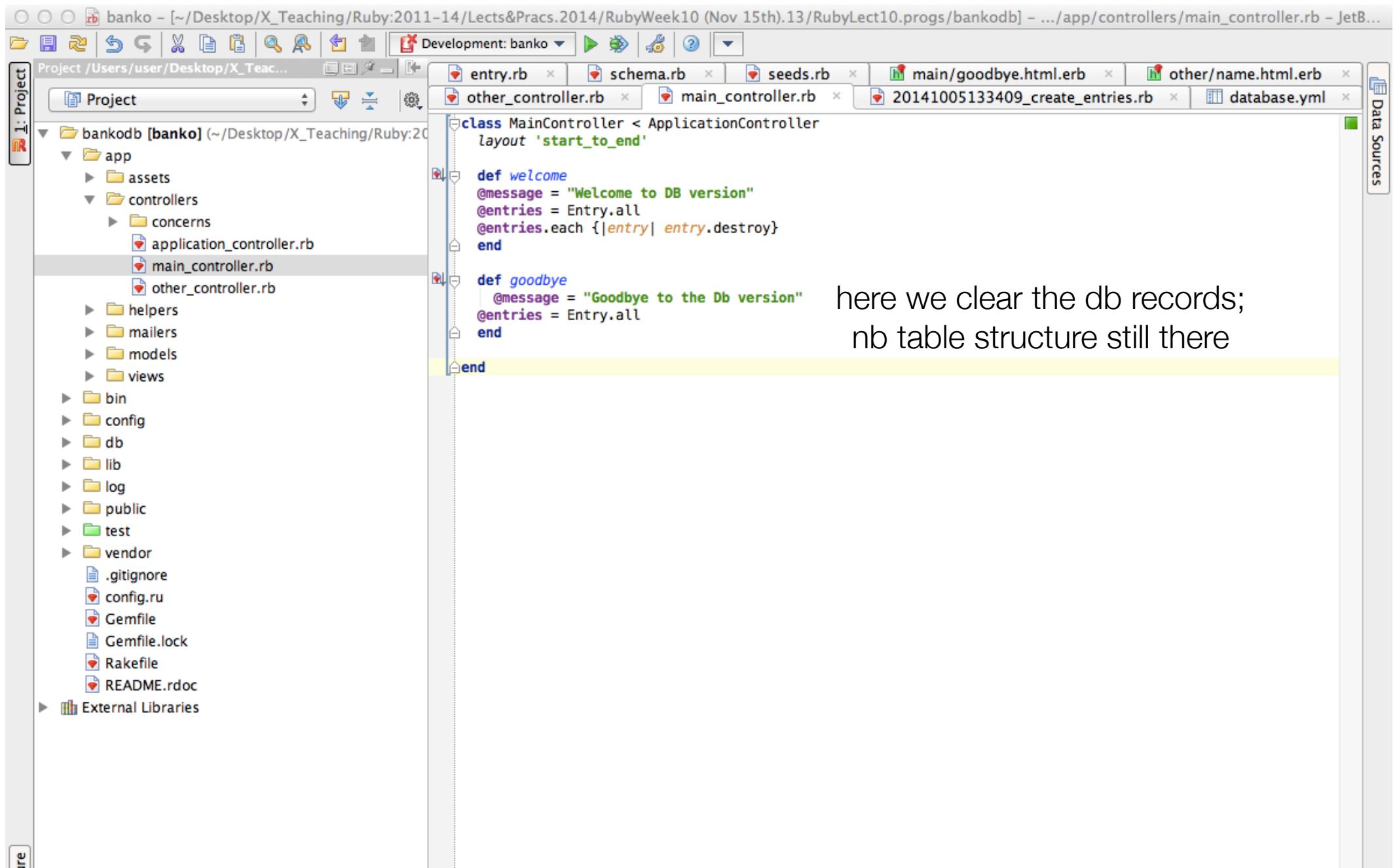


The screenshot shows a Java IDE interface with a Ruby project open. The left pane displays the project structure under 'banko' (~/Desktop/X_Teaching/Ruby). The right pane shows the code editor with 'other_controller.rb' selected. A black oval highlights the first few lines of the 'name' method:

```
class OtherController < ApplicationController
  def name
    @fname = params[:first_name]
    @lname = params[:last_name]
    if @fname then @entry = Entry.create({:first_name => @fname, :last_name => @lname}) end
  end
```

The code uses Active Record's `create` method to save a new entry with parameters `:first_name` and `:last_name`.

...also should change Main



The screenshot shows a JetBrains IDE interface with the following details:

- Title Bar:** banko - [/Desktop/X_Teaching/Ruby:2011-14/Lects&Pracs.2014/RubyWeek10 (Nov 15th).13/RubyLect10.progs/bankodb] - .../app/controllers/main_controller.rb - JetB...
- Toolbars:** Standard Java-like icons for file operations, search, and navigation.
- Project View (left):** Shows the project structure under "bankodb [banko]". The "app/controllers" folder contains "main_controller.rb" which is currently selected. Other files like "entry.rb", "schema.rb", "seeds.rb", "other_controller.rb", "20141005133409_create_entries.rb", and "database.yml" are also listed.
- Code Editor (right):** Displays the content of "main_controller.rb". The code defines a class "MainController" that inherits from " ApplicationController" and uses a layout "start_to_end". It contains two methods: "welcome" and "goodbye". Both methods set a message and iterate over all entries in the database, destroying each one. The code ends with an "end".

```
class MainController < ApplicationController
  layout 'start_to_end'

  def welcome
    @message = "Welcome to DB version"
    @entries = Entry.all
    @entries.each { |entry| entry.destroy}
  end

  def goodbye
    @message = "Goodbye to the Db version"
    @entries = Entry.all
  end

end
```
- Right Panel:** Shows "Data Sources" and other project-related information.

Text on the right: here we clear the db records;
nb table structure still there

Last Screen...BANG !



**Goodbye to Monster & Leinster's
End Page**



[click logo to re-enter site]

- mark, keane , 2000, 100000 9

Monster & Leinster Inc. © 2011

PC Problems: Path Var

To create a new rails website in Windows 8 follow the lecture instructions below:

1. Run command prompt
2. **gem install rails**
3. **gem update --system**
4. **gem install bundle**
5. **bundle install**
6. Go to your Ruby Project directory: **cd C:\RubyMine**
7. Make a new directory: **mkdir temp**
8. **cd temp**
9. **rails new newsite**
10. **cd newsite**
11. **rails generate controller test hello**
12. look at /newsite/app/controllers/test_controller.rb
13. add @message = "hello" to hello method
14. look at newsite/app/views/test/hello.html.erb
15. add <p> <%= @message %></p> to file
16. **rails server**
17. load page in browser: <http://localhost:3000/test/hello>

An error page occurs (sorry I have no screenshot - post it if you can). I think it has to do with a missing system path variable - but I am not entirely sure. To fix this:

1. Install NodeJs from <http://nodejs.org/dist/v0.10.22/node-v0.10.22-x86.msi>
2. Restart your computer.

To run the example code RubyLect10.progs in Windows 8:

1. Download the RubyLect10.progs from Moodle (<https://csimoodle.ucd.ie/moodle/mod/resource/view.php?id=20485>)
2. Extract the zip to your Ruby Project folder (e.g. C:\RubyMine\temp)
3. Run command prompt
4. Navigate to the extracted zip
- 5.

```
C:\RubyMine\temp>cd RubyLect10.progs
C:\RubyMine\temp\RubyLect10.progs>cd hello
C:\RubyMine\temp\RubyLect10.progs\hello>rails server
C:/Ruby200/lib/ruby/gems/2.0.0/gems/sqlite3-1.3.4-x86-mingw32/lib/sqlite3.rb:6:in `require': cannot load such file -- sqlite3/sqlite3_native (LoadError)
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/sqlite3-1.3.4-x86-mingw32/lib/sqlite3.rb:6:in `rescue in <top (required)>'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/sqlite3-1.3.4-x86-mingw32/lib/sqlite3.rb:2:in '<top (required)>'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/bundler-1.3.5/lib/bundler.rb:72:in `require'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/bundler-1.3.5/lib/bundler.rb:72:in `block <2 levels> in require'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/bundler-1.3.5/lib/bundler.rb:70:in `each'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/bundler-1.3.5/lib/bundler.rb:70:in `block in require'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/bundler-1.3.5/lib/bundler.rb:59:in `each'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/bundler-1.3.5/lib/bundler.rb:59:in `require'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/bundler-1.3.5/lib/bundler.rb:2:in `require'
        from C:/RubyMine/temp/RubyLect10.progs/hello/config/application.rb:52:in `require'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/railties-3.1.0/lib/railties.rb:52:in `block in <top (required)>'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/railties-3.1.0/lib/railties.rb:49:in `tap'
        from C:/Ruby200/lib/ruby/gems/2.0.0/gems/railties-3.1.0/lib/railties.rb:49:in `<top (required)>'
        from script/rails:6:in `require'
        from script/rails:6:in `<main>'
```

6.

```
C:\RubyMine\temp\RybyLect10.progs\hello>bundle update sqlite3
Fetching gem metadata from http://rubygems.org/.....
Fetching gem metadata from http://rubygems.org/...
Resolving dependencies...
Using rake <0.9.2>
Using multi_json <1.0.3>
Using activesupport <3.1.0>
Installing bcrypt-ruby <3.0.1>
Using builder <3.0.0>
Using i18n <0.6.0>
Using activemodel <3.1.0>
Using erubis <2.7.0>
Using rack <1.3.4>
Using rack-cache <1.0.3>
Using rack-mount <0.8.3>
Using rack-test <0.6.1>
Using hike <1.2.1>
Using tilt <1.3.3>
Using sprockets <2.0.2>
Using actionpack <3.1.0>
Using mime-types <1.16>
Using polyglot <0.3.2>
Using treetop <1.4.10>
Using mail <2.3.0>
Using actionmailer <3.1.0>
Using arel <2.2.1>
Using tzinfo <0.3.30>
Using activerecord <3.1.0>
Using activeresource <3.1.0>
Using ansi <1.3.0>
Using bundler <1.3.5>
Using coffee-script-source <1.1.2>
Using execjs <1.2.9>
Using coffee-script <2.2.0>
Using rack-ssl <1.3.2>
Using json <1.6.1>
Using rdoc <3.10>
Using thor <0.14.6>
Using railties <3.1.0>
Using coffee-rails <3.1.1>
Using jquery-rails <1.0.14>
Using rails <3.1.0>
Using sass <3.1.10>
Using sass-rails <3.1.4>
Using sqlite3 <1.3.8>
Using turn <0.8.3>
Using uglifier <1.0.3>
Your bundle is updated!
```

7.

```
C:\RubyMine\temp\RUBYLECT10.progs\hello>rails server
=> Booting WEBrick
=> Rails 3.1.0 application starting in development on http://0.0.0.0:30
=> Call with -d to detach
=> Ctrl-C to shutdown server
[2013-11-15 04:28:27] INFO  WEBrick 1.3.1
[2013-11-15 04:28:27] INFO  ruby 2.0.0 <2013-06-27> [i386-mingw32]
[2013-11-15 04:28:27] INFO  WEBrick::HTTPServer#start: pid=124 port=300
```

8. Now the server runs. Request the page in a browser:

<http://localhost:3000/other/name>

9. When you request the page in a browser you get an error because of a missing or incompatible version of the bcrypt-ruby gem (see screenshots below)

10.

```
[2013-11-15 04:28:44] ERROR LoadError: cannot load such file -- 2.0/bcrypt
  C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:240:in `require'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:240:in `block in require'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:223:in `block in load_dependency'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:640:in `new_constants_in'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:223:in `load_dependency'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:240:in `require'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/bcrypt-ruby-3.0.1-x86-mingw32
 rypt_ext.rb:2:in `<top (required)>'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:240:in `require'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:240:in `block in require'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:223:in `block in load_dependency'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:640:in `new_constants_in'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:223:in `load_dependency'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:240:in `require'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/bcrypt-ruby-3.0.1-x86-mingw32
 rypt.rb:12:in `<top (required)>'
    C:/Ruby200/lib/ruby/gems/2.0.0/gems/activesupport-3.1.0/lib/acti
  rt/dependencies.rb:240:in `require'
```

```
C:/Ruby200/lib/ruby/gems/2.0.0/gems/rack-1.3.4/lib/rack/lock.rb:111:in `call'  
C:/Ruby200/lib/ruby/gems/2.0.0/gems/actionpack-3.1.0/lib/action_dispatch/middleware/static.rb:53:in `call'  
C:/Ruby200/lib/ruby/gems/2.0.0/gems/railties-3.1.0/lib/rails/engine.rb:55:in `call'
```

11. Press CTRL-C in the terminal to shut down the server. Press y and enter. (see screenshot below)

12.

```
[2013-11-15 04:33:45] INFO  going to shutdown ...  
[2013-11-15 04:33:45] INFO  WEBrick::HTTPServer#start done.  
Exiting  
Terminate batch job <Y/N>? y
```

13. The following step is not necessary but will make you understand the problem. Notice the **ruby x86-mingw32** post-fix to the gem.

14.

```
C:\RubyMine\temp\RbLect10.progs\hello>gem list bcrypt-ruby  
*** LOCAL GEMS ***  
bcrypt-ruby (3.0.1 ruby x86-mingw32)
```

15.

```
C:\RubyMine\temp\RbLect10.progs\hello>gem uninstall bcrypt-ruby  
You have requested to uninstall the gem:  
  bcrypt-ruby-3.0.1-x86-mingw32  
  
activemodel-3.1.0 depends on bcrypt-ruby (>= 3.0.0)  
If you remove this gem, these dependencies will not be met.  
Continue with Uninstall? [yN] y  
Successfully uninstalled bcrypt-ruby-3.0.1-x86-mingw32
```

16. Make sure you type the next command correctly, its: **gem install bcrypt-ruby -version=3.0.1 --platform=ruby**

17.

```
C:\RubyMine\temp\RybyLect10.progs\hello>gem install bcrypt-ruby --version
--platform=ruby
Fetching: bcrypt-ruby-3.0.1.gem (100%)
Temporarily enhancing PATH to include DevKit...
Building native extensions. This could take a while...
Successfully installed bcrypt-ruby-3.0.1
Parsing documentation for bcrypt-ruby-3.0.1
unable to convert "\x90" from ASCII-8BIT to UTF-8 for lib/bcrypt_ext.so
Installing ri documentation for bcrypt-ruby-3.0.1
1 gem installed
```

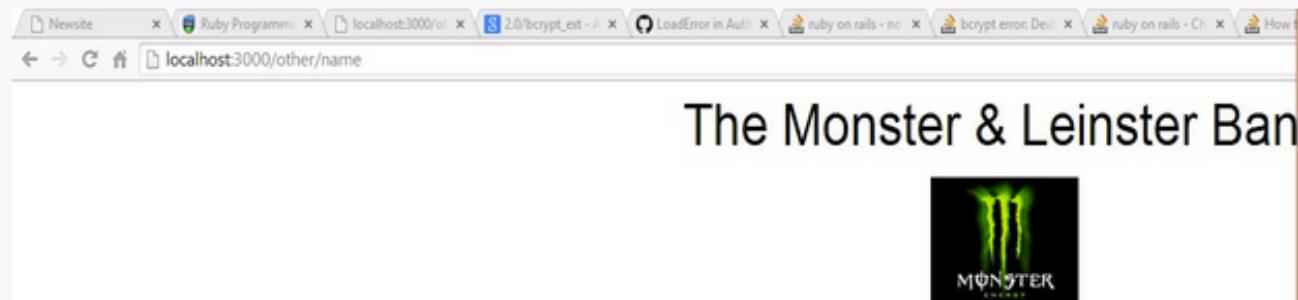
18. Again the following step is not necessary, compare the bcrypt-ruby gem description to the one in step 13. See the difference :)

19.

```
C:\RubyMine\temp\RybyLect10.progs\hello>gem list bcrypt-ruby
*** LOCAL GEMS ***
bcrypt-ruby (3.0.1)
```

20. Execute the server again: **rails server** and request the page in the browser.
(<http://localhost:3000/other/name>)

21.



We are very pleased to welcome you our home mortgage page.

Please enter your first and last name below:

First Name:

Last Name: send

Monster & Leinster Inc. accept no responsibility if you fall into arrears on your loan and are not responsible for the thugs that will call around to your house with baseball bats to help you with your repayment schedule.

22. If you see an error page (sry I have no screenshot - post it if you can) you might be missing a system path variable - but I am not entirely sure. To fix this:
23. Install NodeJs from <http://nodejs.org/dist/v0.10.22/node-v0.10.22-x86.msi>
24. Restart your computer.
25. Run command prompt, navigate to the hello folder, execute the **rails server**, and request the page.