

Ruby on Rails

web application dev

about me

thomas mak

thomas@cpttm.org.mo

agenda
job board
example

scaffolding
restful concept
querying data

scaffolding

scaffolding action time explaining

```
$ rvm use 1.9.3  
$ rails new job_board  
$ cd job_board  
$ bundle install
```

creating a new rails project named job_board.

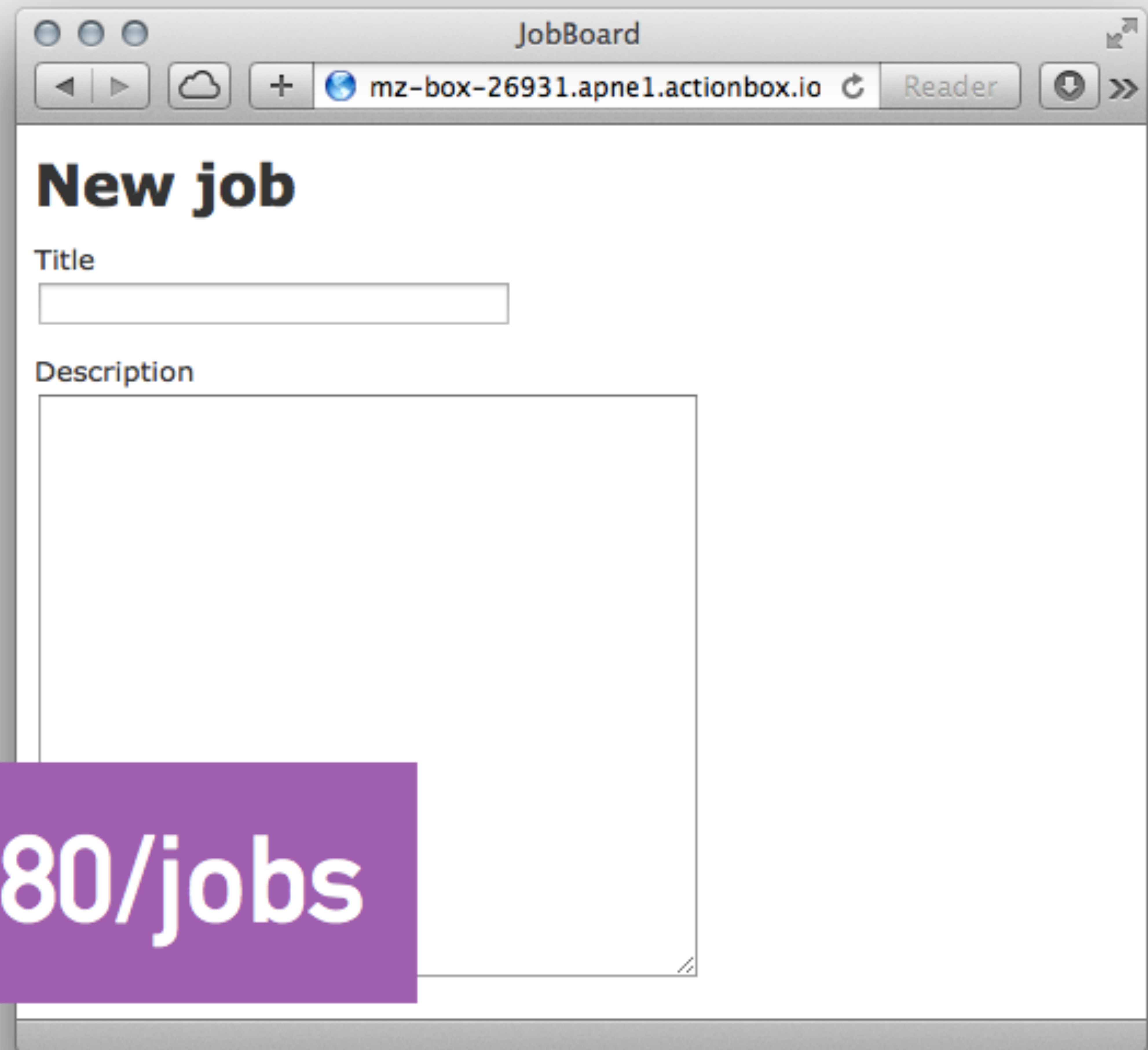
scaffolding action time explaining

```
$ rails generate scaffold jobs  
  title:string content:text  
$ rake db:migrate  
$ rails server -p 8080
```

generate an entire **jobs** resource.

scaffolding
action time
explaining

<http://0.0.0.0:8080/jobs>



The screenshot shows a web browser window with the title 'JobBoard'. The address bar displays 'mz-box-26931.apne1.actionbox.io' with a 'Reader' button and navigation icons. The main content area features a form titled 'New job'. The form has two fields: 'Title' with a single-line text input, and 'Description' with a large multi-line text area.

JobBoard

mz-box-26931.apne1.actionbox.io Reader

New job

Title

Description

scaffolding
action time
explaining

scaffold generates:

- **model**
- **db migration**
- **view** with content
- **scaffold.css**
- **controller** with content
- **routes**
- **css/js files**

scaffolding action time explaining

db migration manages schema versions.

```
$ rake db:migrate
```

going up one version.

```
$ rake db:rollback
```

rolling back one version.

scaffolding
action time
explaining

erb view

.html.erb

<% if is_login %>

<%= @user_name %>

<% end %>

scaffolding
action time
explaining

application.html.erb

<%= yield %>

scaffolding
action time
explaining

erb in yaml

```
host: <%= ENV["db_dev_host"] %>
```

scaffolding
action time
explaining

preprocess: js/css

coffeescript for js
scss for css

scaffolding
action time
explaining

preprocess: css
(optional)

less for css

stylus for css

scaffolding
action time
explaining

preprocess: html
(optional)

haml for html

slim for html

restful concept

restful	create
crud	read
controller	update
routes	delete

restful	restful	http verb
crud	create	post
controller	read	get
routes	update	put
	delete	delete

restful	read:
crud	get /posts/
controller	get /posts/123
routes	

restful

create:

crud

post /posts/

controller

routes

restful

crud

controller

routes

update:

put /posts/123

restful

delete:

crud

delete /posts/123

controller

routes

	controller restful methods
	index
restful	show
crud	new
controller	create
routes	edit
	update
	destroy

controller restful methods		
restful crud controller routes	index	read
	show	read
	new	create
	create	create
	edit	update
	update	update
	destroy	delete

restful crud controller routes	index	/posts/
	show	/posts/123
	new	/posts/new
	create	/posts/123
	edit	/posts/123/edit
	update	/posts/123
	destroy	/posts/123

restful
crud
controller
routes

```
JobBoard::Application routes.draw do  
  resources :posts  
end
```

in `routes.rb`, there is shortcut to define all restful routes.

quering data

querying data

selecting all

Job.all

querying data

selecting with primary key

`Job.find 123`

querying data
showing last
3 jobs

```
def index
  @jobs = Job.last 3

  respond_to do |format|
    format.html # index.html.erb
    format.json { render json: @jobs }
  end
end
```

jobs#index method, last 3 jobs.

querying data
showing last
3 jobs

```
def index
  @jobs = Job.order("id desc").limit(3)

  respond_to do |format|
    format.html # index.html.erb
    format.json { render json: @jobs }
  end
end
```

jobs#index method, last 3 jobs in descending order.

querying data
root page

```
$ rails generate controller pages index
```

create new pages controller.

querying data
root page

```
get "pages/index"  
root :to => 'pages#index'
```

change the **routes.rb** file.

querying data
root page

```
class PagesController < ApplicationController
  def index
    @jobs = Job.order("id desc").limit(3)
  end
end
```

the pages#index action.

querying data

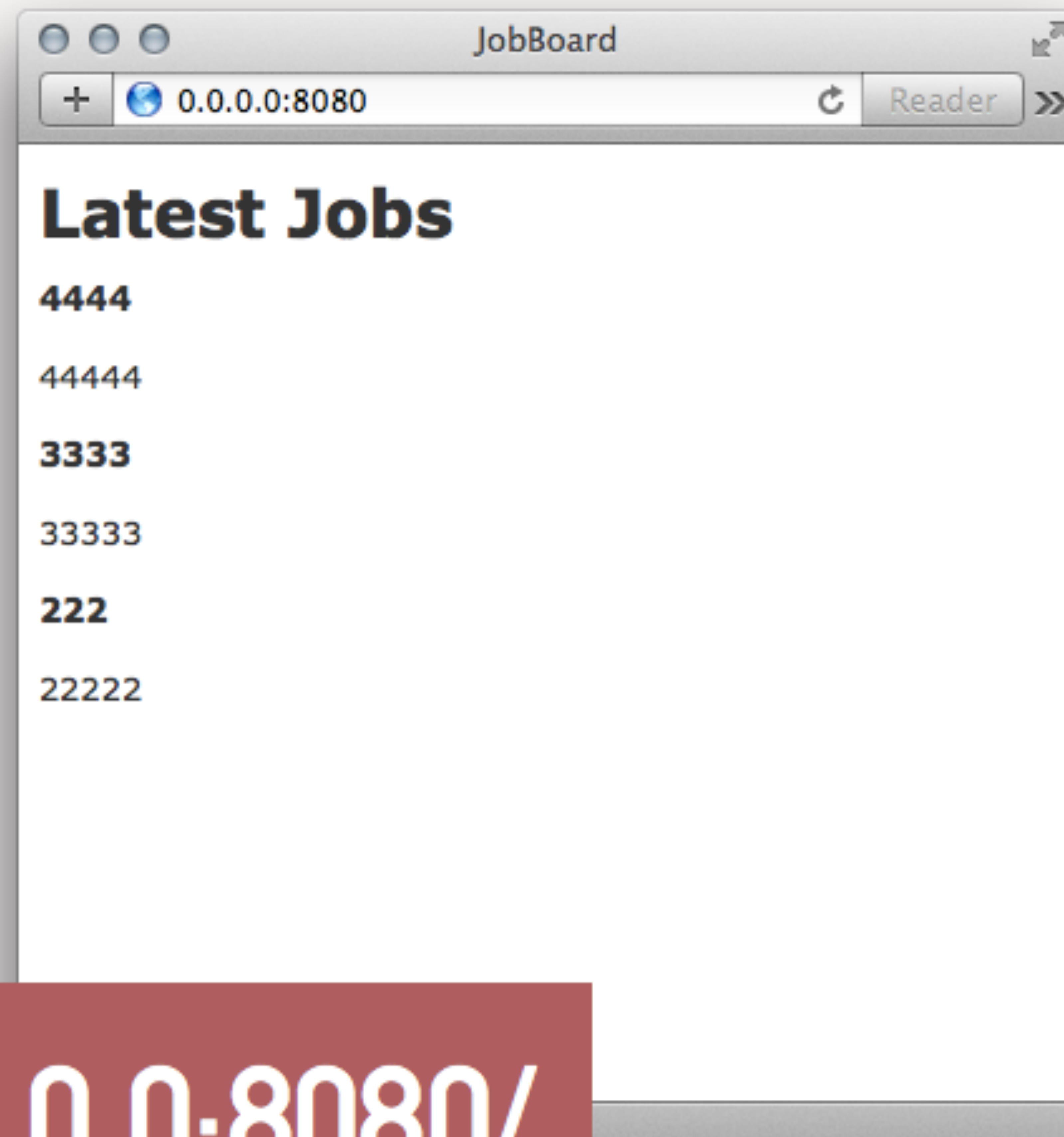
root page

```
<h1>Latest Jobs</h1>

<% @jobs.each do |job| %>
  <p><strong><%= job.title %></strong></p>
  <p><%= job.content %></p>
<% end %>
```

views/pages/index.html.erb

querying data
root page



<http://0.0.0.0:8080/>

querying data

filtering by category

```
$ rails generate migration  
  AddCategoryToJob category:string
```

creating a new column to our job table.

querying data filtering by category

```
class AddCategoryToJob < ActiveRecord::Migration
  def change
    add_column :jobs, :category, :string, default: "ruby"
  end
end
```

set the **default value** of the column.

note: remember to run **rake db:migrate** after the migration file is ready.

querying data
filtering by
category

```
class Job < ActiveRecord::Base
  attr_accessible :content, :title, :category
end
```

add the new column to the **attr_accessible** list.


```
def index
  if params[:category].blank?
    @jobs = Job.all
  else
    # @jobs = Job.find_all_by_category params[:category]
    @jobs = Job.where category: params[:category]
  end

  respond_to do |format|
    format.html # index.html.erb
    format.json { render json: @jobs }
  end
end
```

adding category params with where method.

querying data filtering by category

```
<div class="field">  
  <%= f.label :category %><br />  
  <%= f.text_field :category %>  
</div>
```

views/jobs/_form.html.erb

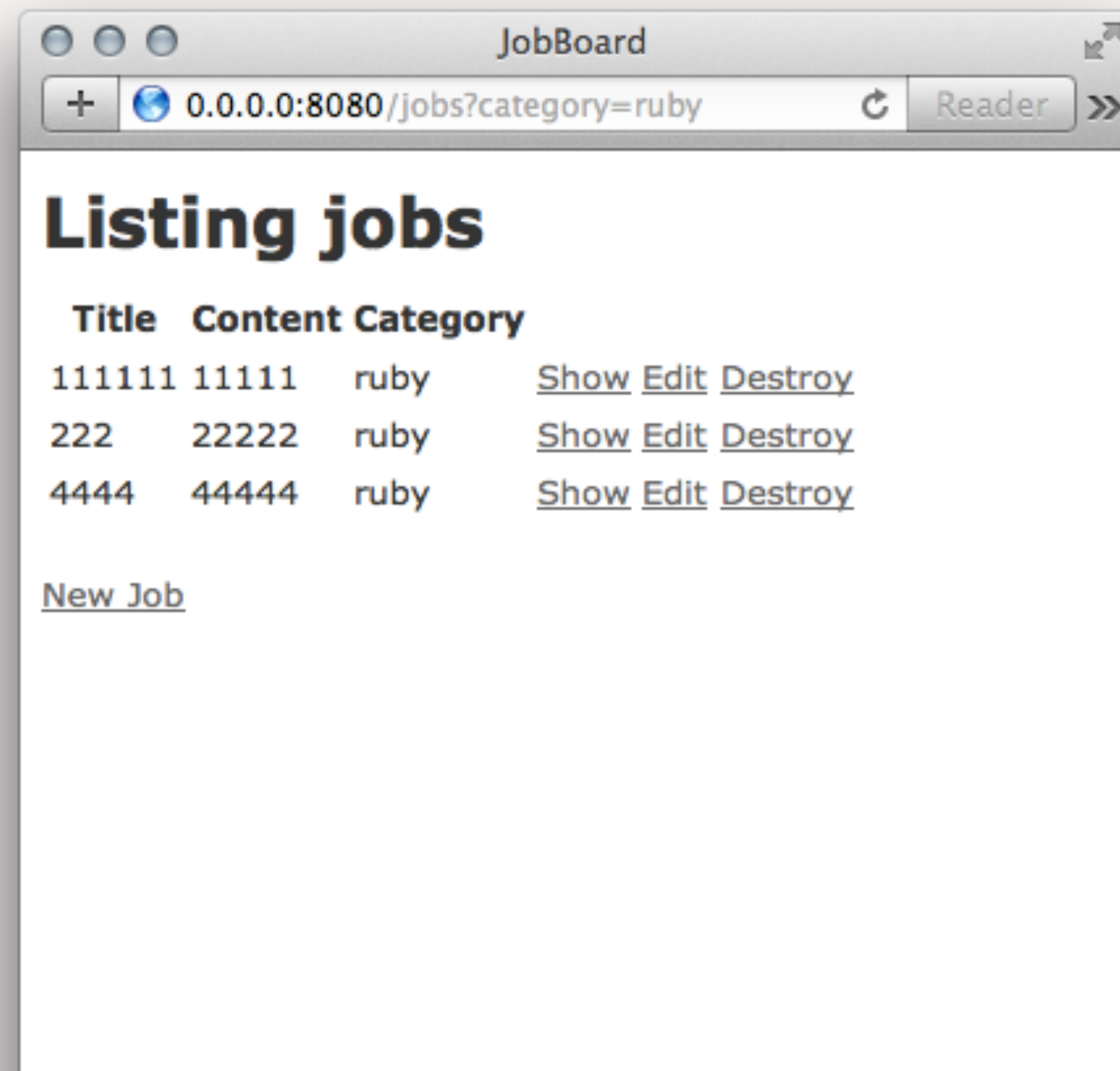
we need to change the view to add the new column.

and also the following view files:

views/jobs/index.html.erb

views/jobs/show.html.erb

querying data
filtering by
category



<http://0.0.0.0:8080/jobs?category=ruby>

querying data
filtering by
category

`params[:name]`
the parameters from
http request.

querying data
filtering by
category

selecting with condition

```
Job.find_all_by_column_name(value)
```

```
Job.find_all_by_a_and_b(value1, value2)
```

deprecated in rails 4.0.

querying data
filtering by
category

selecting with condition

`Job.where(category: "ruby")`

`Job.where("category = ?", "ruby")`

quering data

add salary

```
$ rails generate migration  
  AddSalaryToJob salary:integer
```

creating a new column salary to our job table.

quering data

add salary

```
class AddSalaryToJob < ActiveRecord::Migration
  def change
    add_column :jobs, :salary, :integer, default: 0
  end
end
```

set the **default value** of the column.

note: remember to run **rake db:migrate** after the migration file is ready.

note2: remember to add the new column to **attr_accessor**.

note3: optionally we need to add the new column to the views.

quering data

selecting with greater than

`Job.where("salary >= ?", 15000)`

add salary


```
def index
  min_salary = params[:salary] || 0

  if params[:category].blank?
    @jobs = Job.where("salary >= ?", min_salary)
  else
    @jobs = Job.where("salary >= ?",
min_salary).where(category: params[:category])
  end

  respond_to do |format|
    format.html # index.html.erb
    format.json { render json: @jobs }
  end
end
```

the updated jobs#index action

querying data

setting default value if the
given variable is nil

add salary

`min_salary = a || 0`

querying data

add salary

approach 2

putting the where method
into model.

querying data

add salary

```
class Job < ActiveRecord::Base
  attr_accessible :content, :title
  attr_accessible :category, :salary

  def self.min_salary(salary)
    where("salary >= ?", salary)
  end
end
```

add the **self.min_salary** method to job.rb file.

querying data

```
@jobs = Job.min_salary(min_salary)
```

add salary

now we can add the min salary condition in a more expressive way.

quering data

valid_till

```
$ rails generate migration  
  AddValidTillToJob valid_till:date
```

creating a new column valid_till to our job table.

quering data

valid_till

```
class AddValidTillToJob < ActiveRecord::Migration
  def change
    add_column :jobs, :valid_till, :date,
      default: 10.years.from_now
  end
end
```

set the **default value** of the column.

note: remember to run **rake db:migrate** after the migration file is ready.

note2: remember to add the new column to **attr_accessor**.

note3: optionally we need to add the new column to the views.

querying data

10.years.from_now

5.minutes.ago

valid_till

3 months.from_now

querying data

valid_till

```
def self.still_valid  
  where("valid_till >= ?", Date.current)  
end
```

job.rb: add a new where method to the job model.

```
@jobs = Job.still_valid.min_salary(min_salary)
```

selecting all jobs that's valid and with a minium salary.

resources

<http://railscasts.com/episodes/216>

railscasts.com/episodes/202

[guides.rubyonrails.org/
active_record_querying.html](http://guides.rubyonrails.org/active_record_querying.html)