

Overcoming
Scaffolding

ADDICTION

hi, my name
is amy hoy

**but... let's
talk about
you**

you...

have a

problem



script/console

generate

scaffold foo

POW!

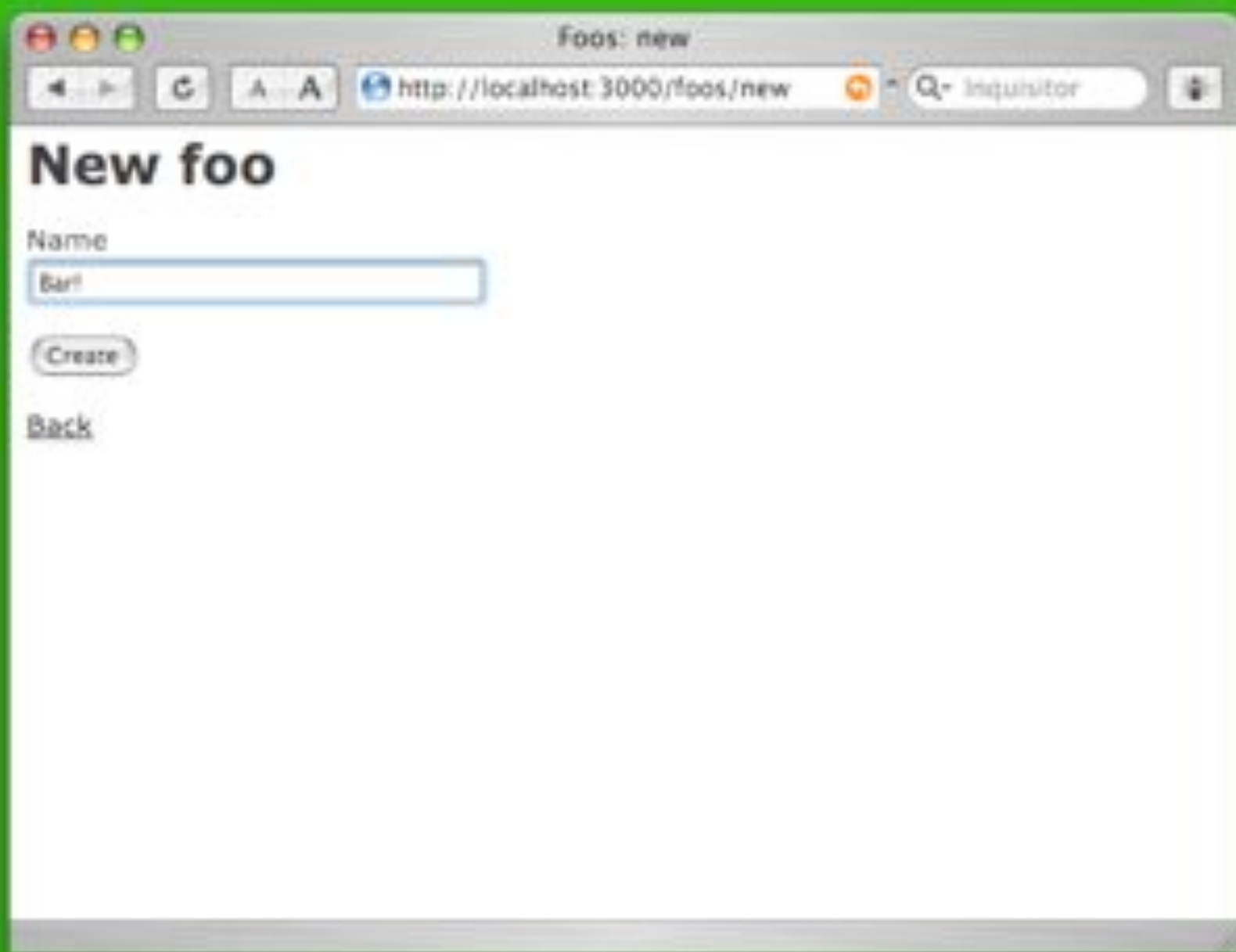
```
exists app/controllers/
exists app/helpers/
create app/views/foos
create test/functional/
dependency model
create app/models/
create test/unit/
create test/fixtures/
create app/models/foo.rb
create test/unit/foo_test.rb
create test/fixtures/foos.yml
create app/views/foos/_form.rhtml
create app/views/foos/list.rhtml
create app/views/foos/show.rhtml
create app/views/foos/new.rhtml
create app/views/foos/edit.rhtml
create app/controllers/foos_controller.rb
create test/functional/
foos_controller_test.rb
create app/helpers/foos_helper.rb
create app/views/layouts/foos.rhtml
create public/stylesheets/scaffold.css
```


it's like...

magic

magic that

sucks!

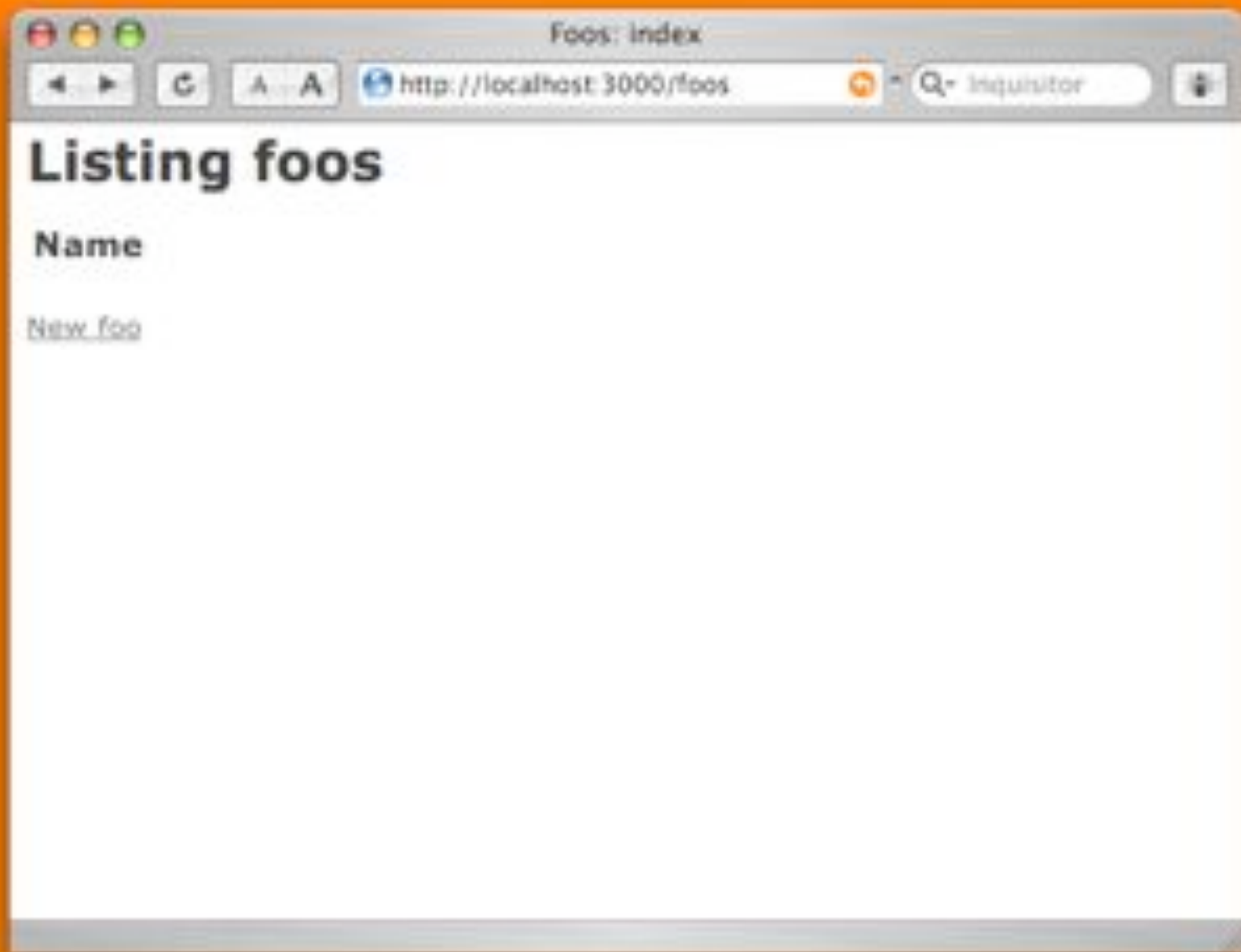


New foo

Name

Create

[Back](#)



Foos: Index



A A

http://localhost:3000/foos



Inquisitor



Listing foos

Name

[New foo](#)



**the
beginnings
of
withdrawal**

how do I

make

scaffolding...

**create prettier
forms?**

link models?

**create multiple
objects?**





**scaffolding
is not the
answer**



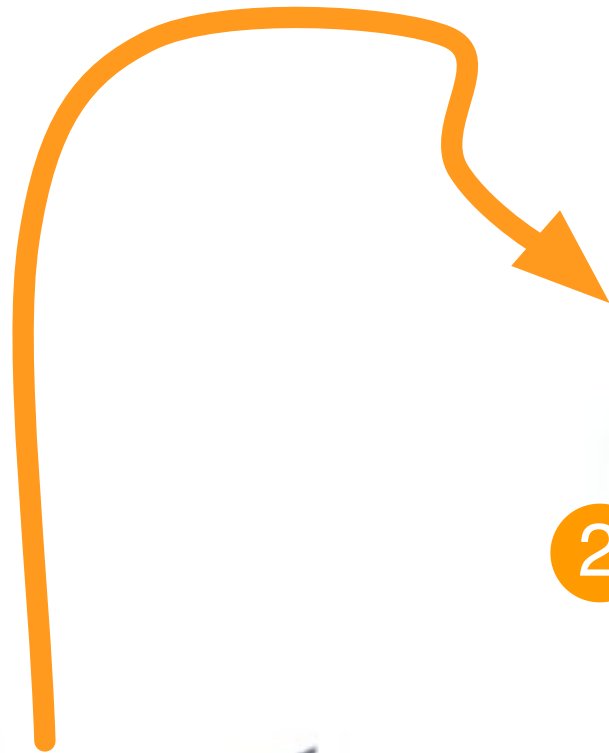


ugly

association-
ignorant



1 Kitten



2 Jar

YEAH RIGHT

Kittens: new

← → C A A >>

New kitten

Name

Create

[Back](#)

Jars: new

← → C A A >>

New jar

Name

Create

[Back](#)

```
class Kitten < ActiveRecord::Base
  belongs_to :jar
end
```

```
class Jar < ActiveRecord::Base
  has_many :kittens
end
```

code excess

```
list
  render :action => 'list'
end

# GETs should be safe (see http://www.w3.org/2001/tag/doc/whenToUseGet.html)
verify :method => :post, :only => [ :destroy, :create, :update ],
      :redirect_to => { :action => :list }
```

```
def list
  @jar_pages, @jars = paginate :jars, :per_page => 10
end
```

```
def show
  @jar = Jar.find(params[:id])
end
```

```
def new
  @jar = Jar.new
end

def create
  @jar = Jar.new(params[:jar])
  if @jar.save
    flash[:notice] = 'Jar was successfully created.'
    redirect_to :action => 'list'
  else
    render :action => 'new'
  end
end
```

```
<h1>New jar</h1>
```

```
<%= start_form_tag :action => 'create' %>
  <%= render :partial => 'form' %>
  <%= submit_tag "Create" %>
<%= end_form_tag %>
```

```
<%= link_to 'Back', :action => 'list' %>
```

```
<h1>Editing foo</h1>
```

```
def edit
  @jar = Jar.find(params[:id])
end

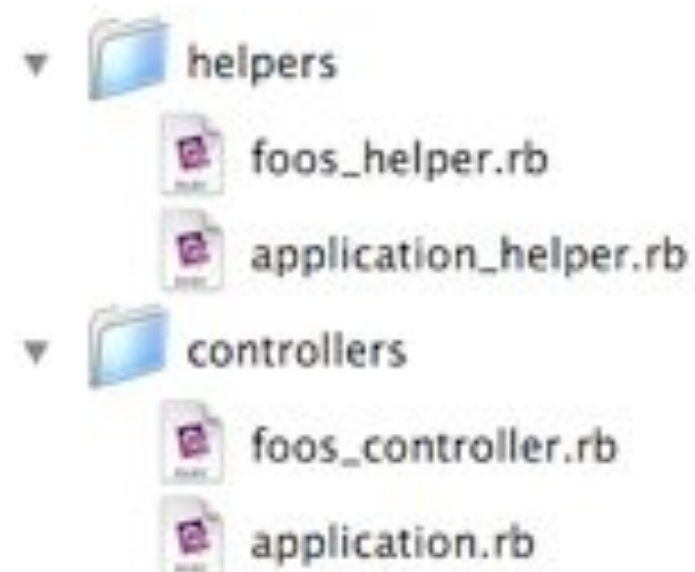
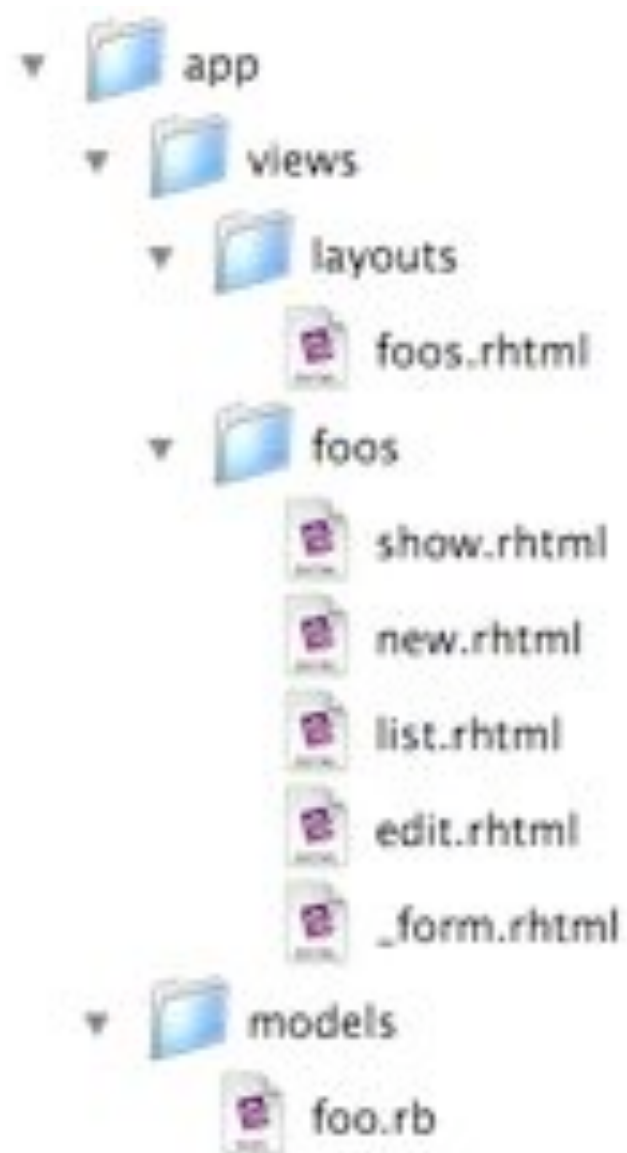
def update
  @jar = Jar.find(params[:id])
  if @jar.update_attributes(params[:jar])
    flash[:notice] = 'Jar was successfully updated.'
    redirect_to :action => 'show', :id => @jar.id
  else
    render :action => 'edit'
  end
end
```

```
def edit
  @jar = Jar.find(params[:id])
end
```

```
def update
  @jar = Jar.find(params[:id])
  if @jar.update_attributes(params[:jar])
    flash[:notice] = 'Jar was successfully updated.'
    redirect_to :action => 'show', :id => @jar.id
  else
    render :action => 'edit'
  end
end
```

too damn

many files



**brain-
numbing**

**fresh start,
cold turkey**

PSSST, THIS
IS TURKEY

BRRRRRRR



**what do you
need?**

kittens

create

view

edit

feed

put to sleep

jars

create

edit

delete

?



**script/generate
controller kittens
create view edit feed
put_to_sleep**

script/generate

controller kittens

create view edit feed

put_to_sleep

script/generate

controller kittens

create view edit feed

put_to_sleep

```
class KittensController < ApplicationController

  def create
  end

  def view
  end

  def edit
  end

  def feed
  end

  def put_to_sleep
  end
end
```

app/controllers/kittens_controller.rb

```
script/generate model  
kitten --skip-migration
```

```
script/generate model  
jar --skip-migration
```



```
script/generate model  
kitten --skip-migration
```

```
script/generate model  
jar --skip-migration
```

```
script/generate model  
kitten --skip-migration
```

```
script/generate model  
jar --skip-migration
```

```
class Jar < ActiveRecord::Base  
end
```

```
class Kitten < ActiveRecord::Base  
end
```

app/models/jar.rb and kitten.rb



```
class Jar < ActiveRecord::Base
  has_many :kittens
end
```

```
class Kitten < ActiveRecord::Base
  belongs_to :jar
end
```

app/models/jar.rb and kitten.rb

**new and
create,
together at
last**

```
class KittensController < ApplicationController

  def create
    @kitten = Kitten.new(params[:kitten])
    if(params[:kitten])
      # more to come here...
    else
      # show form
      render :action => 'create'
    end
  end
end
```

http://localhost:3000/kittens/create



http://localhost:3000/kittens/create

Q+ Inquisitor



Kittens#create

Find me in app/views/kittens/create.rhtml


```
<h1>Kittens#create</h1>
```

```
<p>Find me in app/views/kittens/create.rhtml</p>
```

app/views/kittens/create.rhtml

```
<h1>Kittens#create</h1>
```

```
<p>Find me in app/views/kittens/create.rhtml</p>
```

```
<%= start_form_tag :action => 'create' %>
```

```
  <%= text_field ('kitten', 'name') %>
```

```
  <%= submit_tag 'Save' %>
```

```
<%= end_form_tag %>
```

app/views/kittens/create.rhtml



A A



Kittens#create

Find me in `app/views/kittens/create.rhtml`

Save

but...

```
<h1><%= params[:action].capitalize %> Kitten</h1>
```

```
<%= start_form_tag :action => 'create' %>
```

```
<b>Kitten Name:<b/><br />
```

```
<%= text_field ('kitten', 'name') %>
```

```
<br /><br />
```

```
<b>Jar Name:<b/><br />
```

```
<%= text_field ('jar', 'name') %>
```

```
<%= submit_tag 'Save' %>
```

```
<%= end_form_tag %>
```

app/views/kittens/create.rhtml



Create Kitten

Kitten Name:

Jar Name:

Save

/kittens/create

Kittens Controller
kittens_controller.rb

if(params[:kitten])...

yep!

*as yet
unspecified*

nope!

show form
create.rhtml

incomiiiiing!


```
<h1><%= params[:action].capitalize %> Kitten</h1>
```

```
<%= debug(params) %>
```

```
<%= start_form_tag :action => 'create' %>
```

```
<b>Kitten Name:<b/><br />
```

```
<%= text_field ('kitten', 'name') %>
```

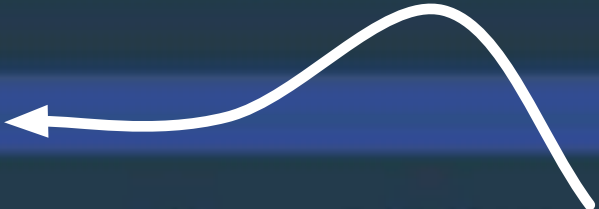
```
<br /><br />
```

```
<b>Jar Name:<b/><br />
```

```
<%= text_field ('jar', 'name') %>
```

```
<%= submit_tag 'Save' %>
```

```
<%= end_form_tag %>
```



this outputs a **YAML**
dump of any object

app/views/kittens/create.rhtml

Create Kitten

```
--- !map:HashWithIndifferentAccess
commit: Save
action: create
jar: !map:HashWithIndifferentAccess
  name: hexagonal
controller: kittens
kitten: !map:HashWithIndifferentAccess
  name: Furball
```

Kitten Name:

Jar Name:

Save

```
# params is a multi-dimensional hash  
params[:object][:column_name]
```

```
# so...  
params[:kitten][:name]  
params[:jar][:name]
```

```
# and when you're editing an object,  
# Rails gives you:  
params[:kitten][:id]  
params[:jar][:id]
```

any view! any controller!

linky linky

```
class KittensController < ApplicationController

  def create
    @kitten = Kitten.new(params[:kitten])
    if(params[:kitten])
      # locate jar by name, or make a new one
      @jar = Jar.find_or_create_by_name(params[:jar][:name])
      @kitten.jar = @jar
    else
      # show form
      render :action => 'create'
    end
  end
end
```

app/controllers/kittens_controller.rb

saving might

be nice


```
class KittensController < ApplicationController

  def create
    @kitten = Kitten.new(params[:kitten])
    if(params[:kitten])
      # locate jar by name, or make a new one
      @jar = Jar.find_or_create_by_name(params[:jar][:name])
      @kitten.jar = @jar
      @kitten.save
      redirect_to :action => 'view', :id => @kitten
    else
      # show form
      render :action => 'create'
    end
  end
end
```

app/controllers/kittens_controller.rb

```
class KittensController < ApplicationController

  def create
    @kitten = Kitten.new(params[:kitten])
    if params[:kitten]
      # locate jar by name, or make a new one
      @jar = Jar.find_or_create_by_name(params[:jar][:name])
      @kitten.jar = @jar
      if @kitten.save
        flash[:notice] = 'You saved a kitten! Hooray!'
        redirect_to :action => 'view', :id => @kitten
      end
    else
      # show form
      render :action => 'create'
    end
  end
end
```

app/controllers/kittens_controller.rb

http://localhost:3000/kittens/view/1



http://localhost:3000/kittens/view/1



Inquisitor



Kittens#view

Find me in `app/views/kittens/view.rhtml`

```
class KittensController < ApplicationController  
  
  def view  
    @kitten = Kitten.find(params[:id],  
                          :include => [:jar])  
  end  
  
end
```

app/controllers/kittens_controller.rb

```
<h1>Kittens#view</h1>
```

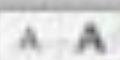
```
<p>Find me in app/views/kittens/view.rhtml</p>
```

app/views/kittens/view.rhtml

```
<div class="status"><%= flash[:notice] %></div>
<h1>Looking At: <%= @kitten.name %></h1>
<p>
  The kitten named <%= @kitten.name %>
  is in the <%= @kitten.jar.name %> jar!
</p>
```

app/views/kittens/view.rhtml

http://localhost:3000/kittens/view/3



http://localhost:3000/kittens/view/3



inquisitor



You saved a kitten! Hooray!

Looking At: Fuzzy

The kitten named Fuzzy is in the a jar!

**isn't that
refreshing?**



PLEASE! NO
MORE CODE!

**other fun
things to do
with forms**

partials

use other field types

create multiple associations

create a bunch of records

i'm partial

app/views/kittens/create.rhtml

```
<h1><%= params[:action].capitalize %> Kitten</h1>

<%= start_form_tag :action => 'create' %>
  <%= render :partial => 'kitten_form' %>
<%= end_form_tag %>
```

create this file

app/views/kittens/_kitten_form.rhtml

```
<b>Kitten Name:</b><br />
<%= text_field ('kitten', 'name') %>
<br /><br />
<b>Jar Name:</b><br />
<%= text_field ('jar', 'name') %>
<%= submit_tag 'Save' %>
```

the preceding
underscore means
"it's a partial"

select menus

kinda hurt.

OW.

step 1: preload data in controller

```
def create
  @kitten = Kitten.new(params[:kitten])
  if params[:kitten]
    #...
  else
    # show form
    @jars = Jar.find(:all)
    render :action => 'create'
  end
end
```

step 2: create select tag in view

```
<b>Jar Name:<b/><br />
<%= collection_select('kitten', 'jar_id',
  @jars, 'id', 'name') %>
```

step 3: there is no step 3!

**dealing with
multiple
associations**



1 Kittens

OMG



2 Jar



step 1: helper method for checks

```
module JarsHelper
  def build_checklist_group(kittens)
    str = String.new
    for kitten in kittens
      str << %<input type="checkbox" name="kittens["
        id="kitten_#{kitten.id}" value="#{kitten.id}" ]
      str << %< checked='checked'> if @jar.kittens.include?(kitten)
      str << %</> #{kitten.name} <br />
    end
    str
  end
end
```

step 2: call helper in view

```
<b><%= @jar.name %>'s Kittens:</b>
<%= build_checklist_group(@kittens) %>
```


step 3: deal with array in controller

```
def update
  @jar = Jar.find(params[:jar][:id])

  @submitted_kittens = Kitten.find(params[:kittens])
  # destroy removed kittens
  @jar.kittens.delete(@jar.kittens - @submitted_kittens))

  # add new kittens
  @jar.kittens << @submitted_kittens - @jar.kittens

  #....
end
```

**creating
multiple records**

step 1: in your view

```
<b>Enter tags, separated by commas</b>  
<%= text_field_tag 'tags' %>
```

step 2: in your controller

```
def make_tags  
  # takes input like "tag1, tag 2,tag tag tag 3"  
  @tags = params[:tags].split ','  
  params[:tags].each do |tag|  
    Tag.find_or_create_by_name(tag.strip!)  
  end  
end
```

**aggregate
controllers**

in any
controller, use
any model

```
class HomePageController < ApplicationController

  def index
    home
    render_action 'home'
  end

  def home
    @sections = Section.find(:all)
    @tags = Tag.find(:all)
    @article = Article.find(:first, :order => "created_on DESC")
  end

end
```

app/controllers/home_page_controller.rb

**not every
model
requires a
controller**

not every

action requires

a view


```
class HomePageController < ApplicationController

  def home
    fetch_data
    @article = Article.find(:first, :order => "created_on DESC")
  end

  def archive
    fetch_data
    @article = Article.find(:all)
  end

  private
  def fetch_data
    @sections = Section.find(:all)
    @tags = Tag.find(:all)
  end

end
```

app/controllers/home_page_controller.rb

**filters are
fun**

```
class HomeController < ApplicationController
  before_filter :fetch_data

  def home
    @article = Article.find(:first, :order => "created_on DESC")
  end

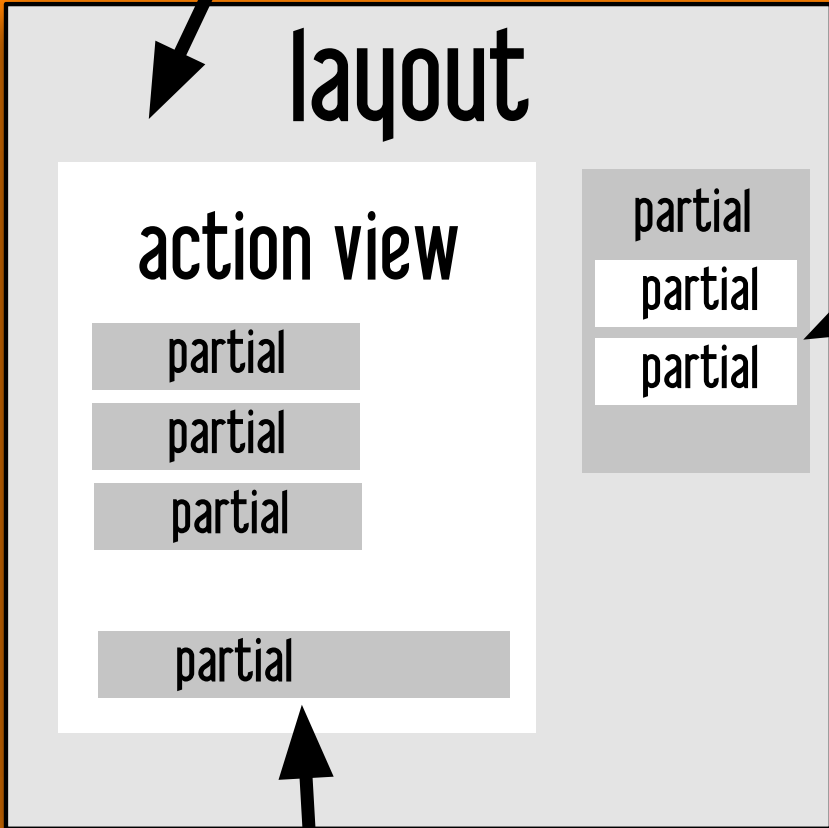
  def archive
    @article = Article.find(:all)
  end

  private
  def fetch_data
    @sections = Section.find(:all)
    @tags = Tag.find(:all)
  end
end
```

app/controllers/home_page_controller.rb

**layouts,
partials,
views**

layouts wrap actions
unless you turn 'em off



partials can go anywhere...
including inside other partials

action views: where the action is
& they're mapped to controller methods

Ruby on Rails Right-Brained Guide

deep breath

relax

A small, fluffy tabby kitten with dark stripes and light brown fur is lying on its belly on a light-colored concrete surface. The kitten has large, bright blue eyes and is looking towards the camera. A white speech bubble with a tail pointing to the kitten's mouth contains the text "OY VEY". The background is slightly out of focus, showing some green grass and a crack in the concrete.

OY VEY

www.slash7.com