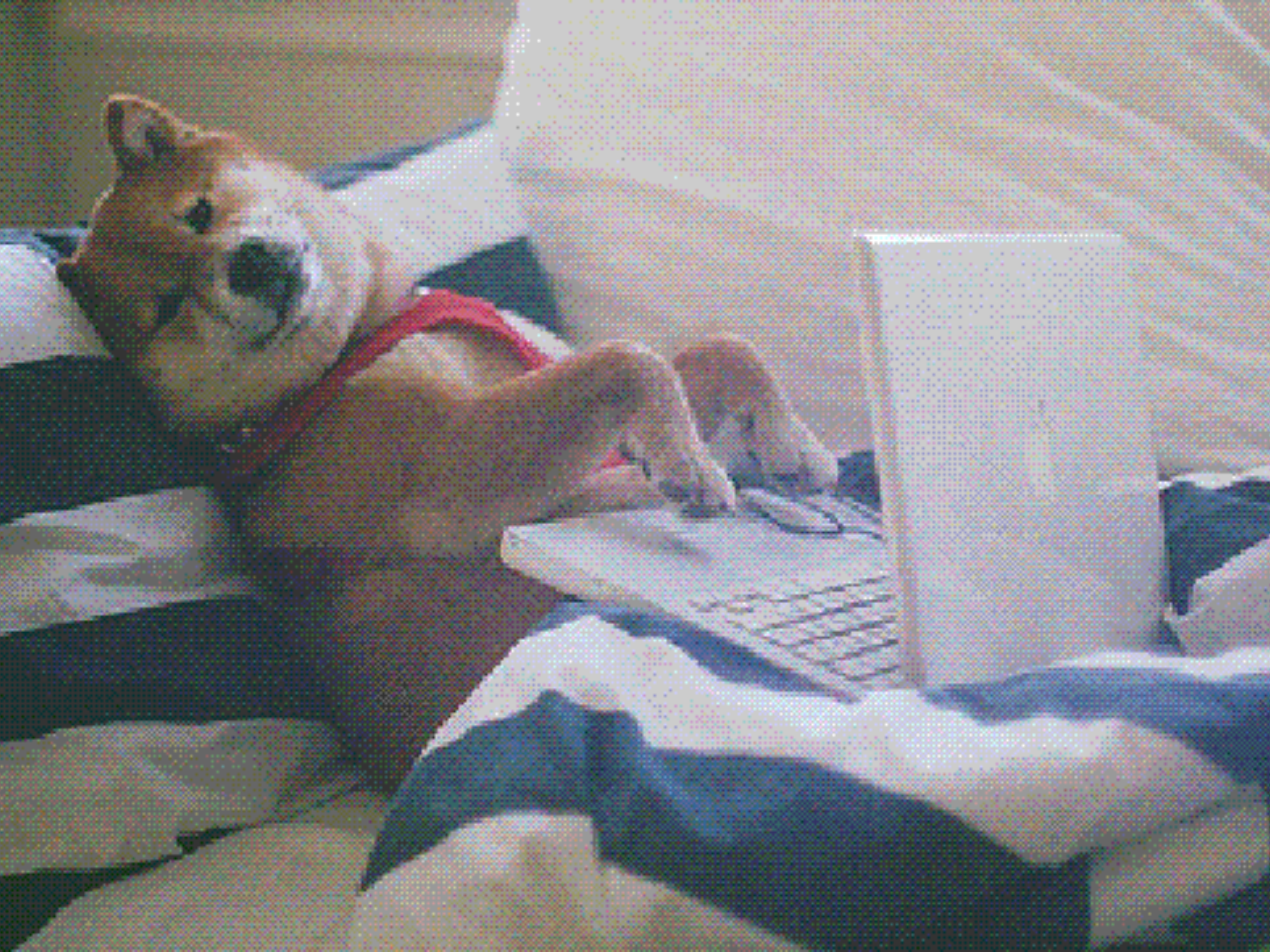




More fun, less pain: a strategy for
writing **maintainable** Rails admin
backends

@steffoz - RubyDay 2014





**CANTIERE
CREATIVO**

stefano verna
@steffoz

```
# app/models/contact.rb
```

```
class Contact < ActiveRecord::Base
  validates :first_name, presence: true
  validates :last_name, presence: true
  validates :twitter, presence: true, uniqueness: true

  def full_name
    [first_name, last_name].join(" ")
  end
end
```

```
# config/routes.rb
```

```
Rails.application.routes.draw do  
  namespace :admin do  
    resources :contacts, except: :show  
  end  
end
```

```
class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end
```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

app/controllers/admin/contacts_controller.rb

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  # ...

end

```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

```
# app/controllers/admin/contacts_controller.rb
```

```

def index
  @contacts = Contact.all
  respond_with @contacts
end

```



```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

app/controllers/admin/contacts_controller.rb

```

def new
  @contact = Contact.new
  respond_with @contact
end

```

```

def create
  @contact = Contact.new(permitted_params)

  if @contact.save
    flash[:notice] = "Contact was successfully created"
  else
    flash[:alert] = "Contact could not be created"
  end

  respond_with @contact, location: admin_contacts_path
end

```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

app/controllers/admin/contacts_controller.rb

```

def edit
  @contact = Contact.find(params[:id])
  respond_with @contact
end

def update
  @contact = Contact.find(params[:id])

  if @contact.update_attributes(permitted_params)
    flash[:notice] = "Contact was successfully updated"
  else
    flash[:alert] = "Contact could not be updated"
  end

  respond_with @contact, location: admin_contacts_path
end

```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

app/controllers/admin/contacts_controller.rb

```

def destroy
  @contact = Contact.find(params[:id])

  if @contact.destroy
    flash[:notice] = "Contact was successfully destroyed"
  else
    flash[:alert] = "Contact could not be destroyed"
  end

  respond_with @contact, location: admin_contacts_path
end

```



```
# app/views/admin/contacts/index.html.erb
```

```
<h1>Contacts</h1>
```

```
<table>
```

```
  <thead>
```

```
    <tr>
```

```
      <th>Name</th>
```

```
      <th>Twitter</th>
```

```
      <th>Actions</th>
```

```
    </tr>
```

```
  </thead>
```

```
  <tbody>
```

```
    <% @contacts.each do |contact| %>
```

```
      <tr>
```

```
        <td><%= contact.full_name %></td>
```

```
        <td><%= contact.twitter %></td>
```

```
        <td>
```

```
          <%= link_to "Edit", edit_admin_contact_path(contact) %>
```

```
          <%= link_to "Delete", admin_contact_path(contact), data: { method: :delete } %>
```

```
        </td>
```

```
      </tr>
```

```
    <% end %>
```

```
  </tbody>
```

```
</table>
```

```
<p><%= link_to "Create new Contact", new_admin_contact_path %></p>
```

```
# app/views/admin/contacts/new.html.erb
```

```
<h1>New Contact</h1>
```

```
<%= form_for(@contact, url: admin_contacts_path) do |f| %>  
  <%= render "form", form: f %>  
  <%= f.button %>  
<% end %>
```

```
# app/views/admin/contacts/edit.html.erb
```

```
<h1>Edit Contact</h1>
```

```
<%= form_for(@contact, url: admin_contact_path(@contact)) do |f| %>  
  <%= render "form", form: f %>  
  <%= f.button %>  
<% end %>
```

```
# app/views/admin/contacts/_form.html.erb
```

```
<div>
  <%= form.label :first_name %>
  <%= form.text_field :first_name %>
  <p><%= form.object.errors.full_messages_for(:first_name).first %></p>
</div>
```

```
<div>
  <%= form.label :last_name %>
  <%= form.text_field :last_name %>
  <p><%= form.object.errors.full_messages_for(:last_name).first %></p>
</div>
```

```
<div>
  <%= form.label :twitter %>
  <%= form.text_field :twitter %>
  <p><%= form.object.errors.full_messages_for(:twitter).first %></p>
</div>
```


Contacts

Name	Twitter	Actions
------	---------	---------

Create new Contact



New Contact

First name

Last name

Twitter

Create Contact

New Contact

First name

Last name

Twitter

Create Contact



Contact could not be created

New Contact

First name

Last name

Twitter

Twitter can't be blank

Contact could not be created

New Contact

First name

Last name

Twitter

Twitter can't be blank

Create Contact



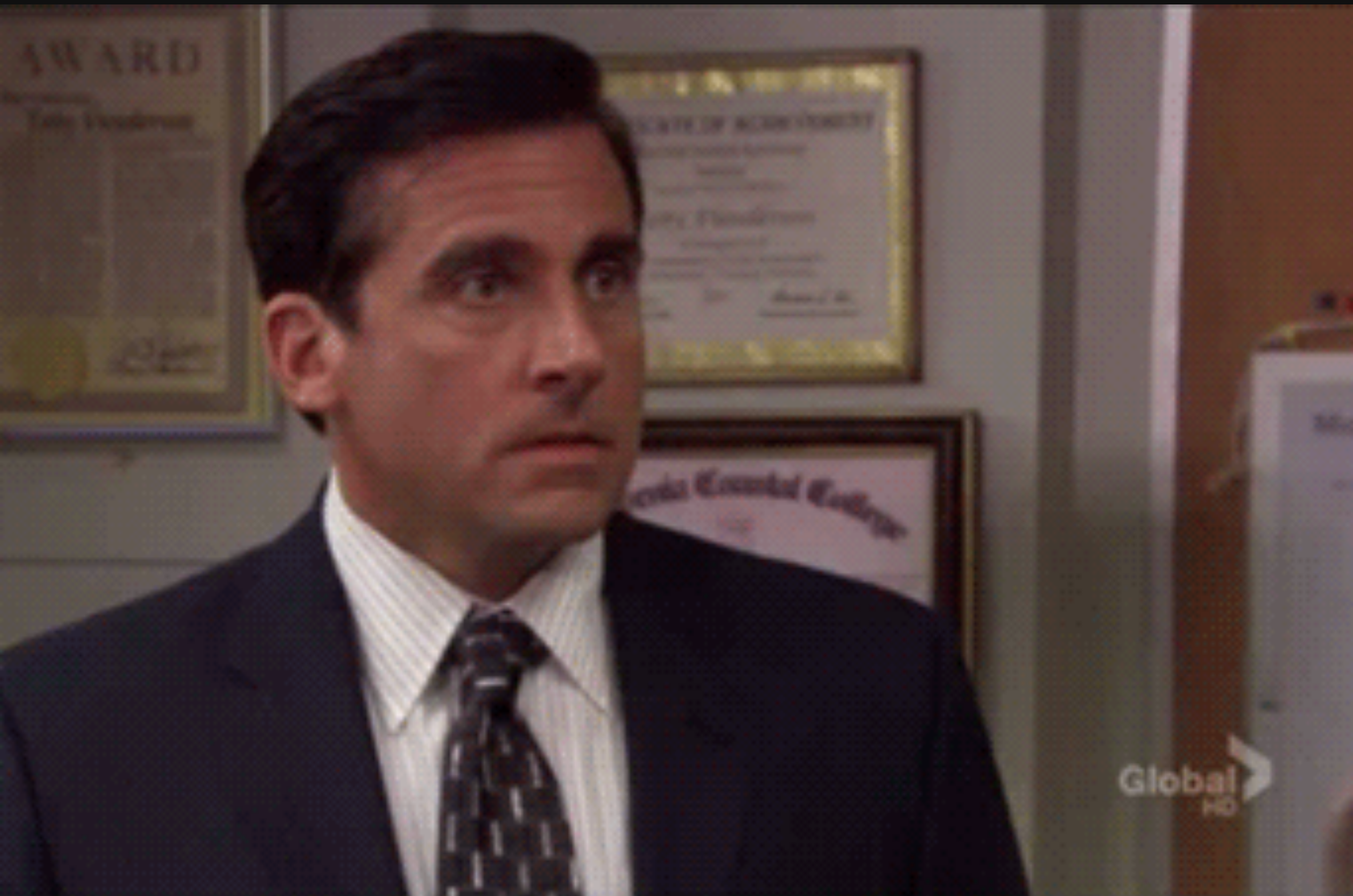
Contact was successfully created.

Contacts

Name	Twitter	Actions
Stefano Verna	@steffoz	Edit Delete

Create new Contact

5 files, ~100 lines of code



```
gem "activeadmin"
```

<https://github.com/activeadmin/activeadmin>

```
# app/admin/contact.rb
```

```
ActiveAdmin.register Contact do  
  permit_params :first_name, :last_name, :twitter
```

```
  index do  
    column :full_name  
    column :twitter  
    actions  
  end
```

```
  form do |f|  
    f.inputs "Details" do  
      f.input :first_name  
      f.input :last_name  
      f.input :twitter  
    end  
    f.actions
```

```
  end  
end
```




ADMIN /

Contacts

[New Contact](#)[Batch Actions](#) ▾

There are no Contacts yet. [Create one](#)

Filters

FIRST NAME

[Contains](#) ▾

LAST NAME

[Contains](#) ▾

TWITTER

[Contains](#) ▾

CREATED AT



-

UPDATED AT



-

[Filter](#)[Clear Filters](#)

New Contact

Details

First name*

Last name*

Twitter*

[Create Contact](#)[Cancel](#)

New Contact

Details

First name*



Last name*

Twitter*

[Create Contact](#)[Cancel](#)

ADMIN /

New Contact

Details

First name*



Last name*

Twitter*

can't be blank

Create ContactCancel

ADMIN /

New Contact

Details

First name*



Last name*

Twitter*

can't be blank

Create Contact

Cancel



ADMIN /

Contacts

New Contact

Batch Actions ▾

Full Name	Twitter	
Stefano Verna	@steffoz	View Edit Delete

Download: [CSV](#) [XML](#) [JSON](#)

Displaying 1 Contact

Filters

FIRST NAME

Contains

LAST NAME

Contains

TWITTER

Contains

CREATED AT

-

UPDATED AT

-

Filter

Clear Filters



well, no.

mixed responsibilities

authentication
authorization
controller
routes
views
css

monolithic approach

80% – 20%



can we make
something better?

reduce repetitive code

~~monolithic approach~~

modular
testable
opt-out

~~authentication~~

~~authorization~~

~~routes~~

~~css~~

controller

views

Contact was successfully created.

Contacts

Name	Twitter	Actions
Stefano Verna	@steffoz	Edit Delete

Create new Contact


```
# app/views/admin/contacts/_form.html.erb
```

```
<div>  
  <%= form.label :first_name %>  
  <%= form.text_field :first_name %>  
  <p><%= form.object.errors.full_messages_for(:first_name).first %></p>  
</div>
```

```
<div>  
  <%= form.label :last_name %>  
  <%= form.text_field :last_name %>  
  <p><%= form.object.errors.full_messages_for(:last_name).first %></p>  
</div>
```

```
<div>  
  <%= form.label :twitter %>  
  <%= form.text_field :twitter %>  
  <p><%= form.object.errors.full_messages_for(:twitter).first %></p>  
</div>
```

```
gem "simple_form"
```

https://github.com/plataformatec/simple_form

```
# app/views/admin/contacts/new.html.erb
```

```
<h1>New Contact</h1>
```

```
<%= form_for(@contact, url: admin_contacts_path) do |f| %>  
  <%= render "form", form: f, contact: @contact %>  
  <%= f.button %>  
<% end %>
```

```
# app/views/admin/contacts/new.html.erb
```

```
<h1>New Contact</h1>
```

```
<%= simple_form_for(@contact, url: admin_contacts_path) do |f| %>  
  <%= render "form", form: f, contact: @contact %>  
  <%= f.button %>  
<% end %>
```

```
# app/views/admin/contacts/_form.html.erb
```

```
<div>
  <%= form.label :first_name %>
  <%= form.text_field :first_name %>
  <p><%= form.object.errors.full_messages_for(:first_name).first %></p>
</div>
```

```
<div>
  <%= form.label :last_name %>
  <%= form.text_field :last_name %>
  <p><%= form.object.errors.full_messages_for(:last_name).first %></p>
</div>
```

```
<div>
  <%= form.label :twitter %>
  <%= form.text_field :twitter %>
  <p><%= form.object.errors.full_messages_for(:twitter).first %></p>
</div>
```

```
# app/views/admin/contacts/_form.html.erb
```

```
<%= form.input :first_name %>
```

```
<%= form.input :last_name %>
```

```
<%= form.input :twitter %>
```

New Contact

* First name

* Last name

* Twitter

Create Contact




```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

```
# app/controllers/admin/contacts_controller.rb
```

```
def update
```

```
  @contact = Contact.find(params[:id])
```

```

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

```

```

    respond_with @contact, location: admin_contacts_path
  end

```

gem "responders"

<https://github.com/plataformatec/responders>

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

app/controllers/admin/contacts_controller.rb

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  # ...

end

```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

app/controllers/admin/contacts_controller.rb

```

class Admin::ContactsController < ApplicationController
  respond_to :html
  responders :flash

  # ...

end

```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

```
# app/controllers/admin/contacts_controller.rb
```

```
def update
```

```
  @contact = Contact.find(params[:id])
```

```

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

```

```

    respond_with @contact, location: admin_contacts_path
  end

```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

```
# app/controllers/admin/contacts_controller.rb
```

```

def update
  @contact = Contact.find(params[:id])
  @contact.update_attributes(permitted_params)
  respond_with @contact, location: admin_contacts_path
end

```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    else
      flash[:alert] = "Contact could not be destroyed"
    end

    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

```
# app/controllers/admin/contacts_controller.rb
```

```

def update
  @contact = Contact.find(params[:id])
  @contact.update_attributes(permitted_params)
  respond_with @contact, location: admin_contacts_path
end

```

```
# config/locales/en.yml
```

```

en:
  flash:
    actions:
      create:
        notice: '%{resource_name} created.'
        alert: '%{resource_name} could not be created.'
      update:
        notice: '%{resource_name} was updated.'
        alert: '%{resource_name} could not be updated.'
      destroy:
        notice: '%{resource_name} was destroyed.'
        alert: '%{resource_name} could not be destroyed.'

```

```

class Admin::ContactsController < ApplicationController
  respond_to :html

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)

    if @contact.save
      flash[:notice] = "Contact was successfully created"
    else
      flash[:alert] = "Contact could not be created"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])

    if @contact.update_attributes(permitted_params)
      flash[:notice] = "Contact was successfully updated"
    else
      flash[:alert] = "Contact could not be updated"
    end

    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])

    if @contact.destroy
      flash[:notice] = "Contact was successfully destroyed"
    end
  end
end

```

```

# app/controllers/admin/contacts_controller

```

```

def update
  @contact = Contact.find(params[:id])
  @contact.update_attributes(permitted_params)
  respond_with @contact, location: admin_contacts_path
end

```

```

# config/locales/en.yml

```

```

en:
  flash:
    actions:
      create:
        notice: '%{resource_name} created.'
        alert: '%{resource_name} could not be created'
      update:
        notice: '%{resource_name} was updated.'
        alert: '%{resource_name} could not be updated'
      destroy:
        notice: '%{resource_name} was destroyed.'
        alert: '%{resource_name} could not be destroyed'

```



```

class Admin::ContactsController < ApplicationController
  respond_to :html
  responders :flash

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)
    @contact.save
    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])
    @contact.update_attributes(permitted_params)
    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])
    @contact.destroy
    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end

```

```

# app/controllers/admin/contacts_controller

```

```

def update
  @contact = Contact.find(params[:id])
  @contact.update_attributes(permitted_params)
  respond_with @contact, location: admin_contacts_path
end

```

```

# config/locales/en.yml

```

```

en:
  flash:
    actions:
      create:
        notice: '%{resource_name} created.'
        alert: '%{resource_name} could not be created'
      update:
        notice: '%{resource_name} was updated.'
        alert: '%{resource_name} could not be updated'
      destroy:
        notice: '%{resource_name} was destroyed.'
        alert: '%{resource_name} could not be destroyed'

```





```
class Admin::ContactsController < ApplicationController
  respond_to :html
  responders :flash

  def index
    @contacts = Contact.all
    respond_with @contacts
  end

  def new
    @contact = Contact.new
    respond_with @contact
  end

  def create
    @contact = Contact.new(permitted_params)
    @contact.save
    respond_with @contact, location: admin_contacts_path
  end

  def edit
    @contact = Contact.find(params[:id])
    respond_with @contact
  end

  def update
    @contact = Contact.find(params[:id])
    @contact.update_attributes(permitted_params)
    respond_with @contact, location: admin_contacts_path
  end

  def destroy
    @contact = Contact.find(params[:id])
    @contact.destroy
    respond_with @contact, location: admin_contacts_path
  end

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end
```

```
def index
  @contacts = Contact.all
  respond_with @contacts
end

def new
  @contact = Contact.new
  respond_with @contact
end

def create
  @contact = Contact.new(permitted_params)
  @contact.save
  respond_with @contact, location: admin_contacts_path
end

def edit
  @contact = Contact.find(params[:id])
  respond_with @contact
end

def update
  @contact = Contact.find(params[:id])
  @contact.update_attributes(permitted_params)
  respond_with @contact, location: admin_contacts_path
end

def destroy
  @contact = Contact.find(params[:id])
  @contact.destroy
  respond_with @contact, location: admin_contacts_path
end
```

Contact

@contacts

@contact

admin_contacts_path

new_admin_contact_path

edit_admin_contact_path

admin_contact_path

Contact

@contacts

@contact

admin_contacts_path

new_admin_contact_path

edit_admin_contact_path

admin_contact_path



resource_class

@collection

@resource

collection_path

new_resource_path

edit_resource_path

resource_path

```
def index
  @contacts = Contact.all
  respond_with @contacts
end

def new
  @contact = Contact.new
  respond_with @contact
end

def create
  @contact = Contact.new(permitted_params)
  @contact.save
  respond_with @contact, location: admin_contacts_path
end

def edit
  @contact = Contact.find(params[:id])
  respond_with @contact
end

def update
  @contact = Contact.find(params[:id])
  @contact.update_attributes(permitted_params)
  respond_with @contact, location: admin_contacts_path
end

def destroy
  @contact = Contact.find(params[:id])
  @contact.destroy
  respond_with @contact, location: admin_contacts_path
end
```

resource_class

@collection

@resource

collection_path

new_resource_path

edit_resource_path

resource_path

```
def index
  @collection = resource_class.all
  respond_with @collection
end

def new
  @resource = resource_class.new
  respond_with @resource
end

def create
  @resource = resource_class.new(permitted_params)
  @resource.save
  respond_with @resource, location: collection_url
end

def edit
  @resource = resource_class.find(params[:id])
  respond_with @resource
end

def update
  @resource = resource_class.find(params[:id])
  @resource.update_attributes(permitted_params)
  respond_with @resource, location: collection_url
end

def destroy
  @resource = resource_class.find(params[:id])
  @resource.destroy
  respond_with @resource, location: collection_url
end
```

resource_class

@collection

@resource

collection_path

new_resource_path

edit_resource_path

resource_path

The **template method pattern** is a design pattern that defines the skeleton of an algorithm in a method, called template method, which defers some steps to subclasses.

It lets one redefine certain steps of an algorithm without changing the algorithm's structure.

```
def index
  @collection = resource_class.all
  respond_with @collection
end

def new
  @resource = resource_class.new
  respond_with @resource
end

def create
  @resource = resource_class.new(permitted_params)
  @resource.save
  respond_with @resource, location: collection_url
end

def edit
  @resource = resource_class.find(params[:id])
  respond_with @resource
end

def update
  @resource = resource_class.find(params[:id])
  @resource.update_attributes(permitted_params)
  respond_with @resource, location: collection_url
end

def destroy
  @resource = resource_class.find(params[:id])
  @resource.destroy
  respond_with @resource, location: collection_url
end
```

resource_class

@collection

@resource

collection_path

new_resource_path

edit_resource_path

resource_path

```
def
  @collection
  respond_with @collection
end
```

```
def
  @resource
  respond_with @resource
end
```

```
def
  @resource
  @resource
  respond_with @resource,
end
```

```
def
  @resource
  respond_with @resource
end
```

```
def
  @resource
  @resource
  respond_with @resource,
end
```

```
def
  @resource
  @resource
  respond_with @resource,
end
```

```
# controllers/concerns/rest_actions_concern.rb
```

```
module RestActionsConcern
  extend ActiveSupport::Concern
```

```
  included do
    respond_to :html
    responders :flash
  end
```

```
  # actions here!
```

```
end
```

```
# controllers/concerns/rest_actions_concern.rb

module RestActionsConcern
  extend ActiveSupport::Concern

  included do
    respond_to :html
    responders :flash
  end

  def index
    @collection = resource_class.all
    respond_with @collection
  end

  def new
    @resource = resource_class.new
    respond_with @resource
  end

  def create
    @resource = resource_class.new(permitted_params)
    @resource.save
    respond_with @resource, location: collection_url
  end

  def edit
    @resource = resource_class.find(params[:id])
    respond_with @resource
  end

  def update
    @resource = resource_class.find(params[:id])
    @resource.update_attributes(permitted_params)
    respond_with @resource, location: collection_url
  end

  def destroy
    @resource = resource_class.find(params[:id])
    @resource.destroy
    respond_with @resource, location: collection_url
  end
end
```

```
class Admin::ContactsController < ApplicationController
  include RestActionsConcern

  private

  def collection_path
    admin_contacts_path
  end

  def new_resource_path
    new_admin_contact_path
  end

  def edit_resource_path(resource)
    edit_admin_contact_path
  end

  def resource_path(resource)
    admin_contact_path(resource)
  end

  def resource_class
    Contact
  end

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end
```

```
def collection_path  
  admin_contacts_path  
end
```

```
def new_resource_path  
  new_admin_contact_path  
end
```

```
def edit_resource_path(resource)  
  edit_admin_contact_path  
end
```

```
def resource_path(resource)  
  admin_contact_path(resource)  
end
```

```
def collection_path
  url_for(controller: controller_path, action: :index)
end

def new_resource_path
  url_for(controller: controller_path, action: :new)
end

def edit_resource_path(resource)
  url_for(controller: controller_path, action: :edit, id: resource)
end

def resource_path(resource)
  url_for(controller: controller_path, action: :show, id: resource)
end
```



```
# controllers/concerns/resource_urls_concern.rb

module ResourceUrlsConcern
  extend ActiveSupport::Concern

  def collection_path
    url_for(controller: controller_path, action: :index)
  end

  def new_resource_path
    url_for(controller: controller_path, action: :new)
  end

  def edit_resource_path(resource)
    url_for(controller: controller_path, action: :edit, id: resource)
  end

  def resource_path(resource)
    url_for(controller: controller_path, action: :show, id: resource)
  end
end
```

```
class Admin::ContactsController < ApplicationController
  include RestActionsConcern
  include ResourceUrlsConcern

  private

  def resource_class
    Contact
  end


  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end
```

```
class Admin::ContactsController < ApplicationController
  include RestActionsConcern
  include ResourceUrlsConcern

  private

  def resource_class
    controller_name.classify.constantize # => Contacts
  end


  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end
```



```
class Admin::ContactsController < ApplicationController
  include RestActionsConcern
  include ResourceUrlsConcern

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end
```



```
class Admin::ContactsController < ApplicationController
  include RestActionsConcern

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end
```

```
class Admin::ContactsController < ApplicationController
  include RestActionsConcern

  private

  def permitted_params
    params.
      require(:contact).
      permit(:first_name, :last_name, :twitter)
  end
end
```



```
# app/admin/contact.rb
```

```
ActiveAdmin.register Contact do
  permit_params :first_name, :last_name, :twitter

  # ...
end
```




magic

~~controller~~
views


```
# app/views/admin/contacts/new.html.erb
```

```
<h1>New Contact</h1>
```

```
<%= simple_form_for(@resource, url: collection_path) do |f| %>  
  <%= render "form", form: f %>  
  <%= f.button :submit %>  
<% end %>
```

```
# app/views/admin/contacts/edit.html.erb
```

```
<h1>Edit Contact</h1>
```

```
<%= simple_form_for(@resource, url: resource_path(@resource)) do |f| %>  
  <%= render "form", form: f %>  
  <%= f.button :submit %>  
<% end %>
```

```
I18n.locale = :it
```

```
resource_class.model_name.human(count: 2) # => "Contatti"
```

```
module ResourceInflectionsConcern
  extend ActiveSupport::Concern

  included do
    helper_method :inflections
  end

  private

  def inflections
    {
      resource_name: resource_class.model_name.human(count: 1),
      collection_name: resource_class.model_name.human(count: 2)
    }
  end
end
```

```
# app/views/admin/contacts/new.html.erb
```

```
<h1>New Contact</h1>
```

```
<%= simple_form_for(@resource, url: collection_path) do |f| %>  
  <%= render "form", form: f %>  
  <%= f.button :submit %>  
<% end %>
```

```
# app/views/admin/contacts/edit.html.erb
```

```
<h1>Edit Contact</h1>
```

```
<%= simple_form_for(@resource, url: resource_path(@resource)) do |f| %>  
  <%= render "form", form: f %>  
  <%= f.button :submit %>  
<% end %>
```

```
# app/views/admin/contacts/new.html.erb
```

```
<h1><%= t('admin.resource.new.title', inflections) %></h1>
```

```
<%= simple_form_for(@resource, url: collection_path) do |f| %>
```

```
  <%= render "form", form: f %>
```

```
  <%= f.button :submit %>
```

```
<% end %>
```

```
# app/views/admin/contacts/edit.html.erb
```

```
<h1><%= t('admin.resource.edit.title', inflections) %></h1>
```

```
<%= simple_form_for(@resource, url: resource_path(@resource)) do |f| %>
```

```
  <%= render "form", form: f %>
```

```
  <%= f.button :submit %>
```

```
<% end %>
```

view inheritance

```
# app/controllers/admin/contacts_controller.rb

class Admin::ContactsController < ApplicationController
  include RestActionsConcern

  # ...

end
```

```
# app/controllers/admin/contacts_controller.rb

class Admin::ContactsController < Admin::BaseController
  include RestActionsConcern

  # ...

end
```

```
# app/controllers/admin/base_controller.rb


class Admin::BaseController < ApplicationController
end
```


 app

 views/admin

 base

 contacts

 new.html.erb


 edit.html.erb

 _form.html.erb

 app

 views/admin

 base

 new.html.erb

 edit.html.erb

 contacts

 _form.html.erb

```
# app/views/admin/contacts/_form.html.erb
```

```
<%= form.input :first_name %>
```

```
<%= form.input :last_name %>
```

```
<%= form.input :twitter %>
```

```
# app/views/admin/contacts/_form.html.erb
```

```
<%= form.input :first_name %>
```

```
<%= form.input :last_name %>
```

```
<%= form.input :twitter %>
```

```
# app/admin/contact.rb
```

```
ActiveAdmin.register Contact do
```

```
  form do |f|
```

```
    f.inputs "Details" do
```

```
      f.input :first_name
```

```
      f.input :last_name
```

```
      f.input :twitter
```

```
    end
```

```
  end
```

```
end
```




```
# app/views/admin/contacts/index.html.erb
```

```
<h1><%= t("admin.resource.index.title", inflections) %></h1>
```

```
<table>
```

```
  <thead>
```

```
    <tr>
```

```
      <th>Name</th>
```

```
      <th>Twitter</th>
```

```
      <th>Actions</th>
```

```
    </tr>
```

```
  </thead>
```

```
  <tbody>
```

```
    <% @collection.each do |contact| %>
```

```
      <tr>
```

```
        <td><%= contact.full_name %></td>
```

```
        <td><%= contact.twitter %></td>
```

```
        <td>
```

```
          <%= link_to "Edit", edit_admin_contact_path(contact) %>
```

```
          <%= link_to "Delete", admin_contact_path(contact), data: { method: :delete } %>
```

```
        </td>
```

```
      </tr>
```

```
    <% end %>
```

```
  </tbody>
```

```
</table>
```

```
<% new_label = t('admin.resource.index.actions.new', inflections) %>
```

```
<p><%= link_to(new_label, new_resource_url) %></p>
```

 app

 views/admin

 base

 new.html.erb

 edit.html.erb

 contacts


 _form.html.erb

 app

 views/admin

 base

 new.html.erb

 edit.html.erb

 index.html.erb

 contacts

 _form.html.erb

 _table.html.erb

gem "admino"

<https://github.com/cantiererecreativo/admino>

```
# app/views/admin/base/index.html.erb
```

```
<h1><%= t('admin.resource.index.title', inflections) %></h1>
```

```
<%= table_for(@collection) do |table, record| %>
```

```
  <%= render 'table', table: table %>
```

```
  <%= table.actions do %>
```

```
    <%= table.action :edit, edit_resource_path(record) %>
```

```
    <%= table.action :destroy, resource_path(record), method: :delete %>
```

```
  <% end %>
```

```
<% end %>
```

```
<% new_label = t('admin.resource.index.actions.new', inflections) %>
```

```
<p><%= link_to(new_label, new_resource_url) %></p>
```

```
# app/views/admin/base/index.html.erb
```

```
<h1><%= t('admin.resource.index.title', inflections) %></h1>
```

```
<%= table_for(@collection) do |table, record| %>
```

```
  <%= render 'table', table: table %>
```

```
  <%= table.actions do %>
```

```
    <%= table.action :edit, edit_resource_path(record) %>
```

```
    <%= table.action :destroy, resource_path(record), method: :delete %>
```

```
  <% end %>
```

```
<% end %>
```

```
<% new_label = t('admin.resource.index.actions.new', inflections) %>
```

```
<p><%= link_to(new_label, new_resource_url) %></p>
```

```
# app/views/admin/base/index.html.erb
```

```
<h1><%= t('admin.resource.index.title', inflections) %></h1>
```

```
<%= table_for(@collection) do |table, record| %>
```

```
  <%= render 'table', table: table %>
```

```
  <%= table.actions do %>
```

```
    <%= table.action :edit, edit_resource_path(record) %>
```

```
    <%= table.action :destroy, resource_path(record), method: :delete %>
```

```
  <% end %>
```

```
<% end %>
```

```
<% new_label = t('admin.resource.index.actions.new', inflections) %>
```

```
<p><%= link_to(new_label, new_resource_url) %></p>
```

```
# app/views/admin/contacts/_table.html.erb
```

```
<%= table.column :full_name %>
```

```
<%= table.column :twitter %>
```

```
# app/admin/contact.rb
```

```
ActiveAdmin.register Contact do
```

```
  index do
```

```
    column :full_name
```

```
    column :twitter
```

```
    actions
```

```
  end
```

```
end
```

```
# app/controllers/admin/contacts_controller.rb
```

```
class Admin::ContactsController < ApplicationController  
  include RestActionsConcern
```

```
  private
```

```
  def permitted_params  
    params.require(:contact).permit(:first_name, :last_name, :twitter)  
  end  
end
```

```
# app/views/admin/contacts/_form.html.erb
```

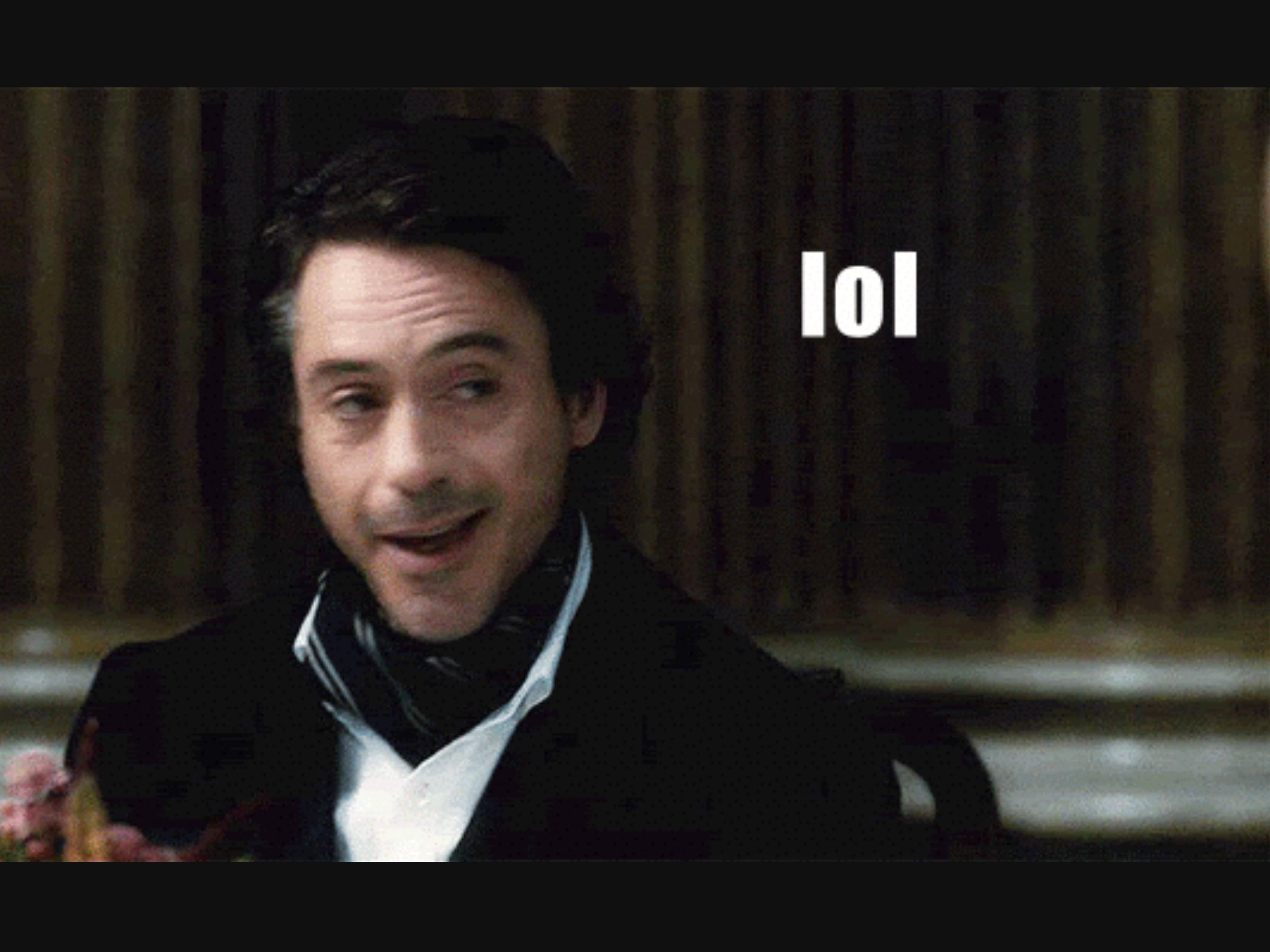
```
<%= form.input :first_name %>  
<%= form.input :last_name %>  
<%= form.input :twitter %>
```

```
# app/views/admin/contacts/_table.html.erb
```

```
<%= table.column :full_name %>  
<%= table.column :twitter %>
```

~~controller~~
~~views~~

lol



Contacts

Name	Twitter	Actions
Stefano Verna	@steffoz	Edit Delete

Create new Contact

Contacts

* Name matches

Search

Filter by period: [All](#) [Created Past Month](#) Added this month

<u>Full name</u> ▲	<u>Twitter</u>	<u>Actions</u>
Ju Liu	@Arkham	Edit Destroy
Matteo Papadopoulos	@spleenteo	Edit Destroy
Stefano Verna	@steffoz	Edit Destroy

Create new Contact

Contacts

* Name matches

Search

Filter by period: [All](#) [Created Past Month](#) Added this month

Full name ▲

Twitter

Actions

Ju Liu

@Arkham

[Edit](#) [Destroy](#)

Matteo Papadopoulos

@spleenteo

[Edit](#) [Destroy](#)

Stefano Verna

@steffoz

[Edit](#) [Destroy](#)

Create new Contact

Contacts

* Name matches

Search

Filter by period: [All](#) [Created Past Month](#) Added this month

<u>Full name</u> ▲	<u>Twitter</u>	<u>Actions</u>
Ju Liu	@Arkham	Edit Destroy
Matteo Papadopoulos	@spleenteo	Edit Destroy
Stefano Verna	@steffoz	Edit Destroy

Create new Contact

Contacts

* Name matches

Search

Filter by period: [All](#) [Created Past Month](#) Added this month

<u>Full name</u> ▲	<u>Twitter</u>	<u>Actions</u>
--------------------	----------------	----------------

Ju Liu	@Arkham	Edit Destroy
Matteo Papadopoulos	@spleenteo	Edit Destroy
Stefano Verna	@steffoz	Edit Destroy

Create new Contact

gem "admino"

table builder

query object

search forms

filter groups

sortable columns

List Contacts

⬆ Name	Email	Group	Actions	
Anderson Jordy	anderson.jordy@lebsacktremblay.info	Work	Edit	Destroy
Bauch Broderick	bauch.broderick@jakubowski.org	Friend	Edit	Destroy
Beatty Jazmyn	jazmyn.beatty@rosenbaum.info	Family	Edit	Destroy
Berge Keon	berge.keon@labadieschmeler.org	Friend	Edit	Destroy
Bernhard Isaiah	bernhard.isaiah@harber.org	Family	Edit	Destroy
Blanda Marcelo	blanda.marcelo@pouros.info	Friend	Edit	Destroy
Boehm Jolie	jolie_boehm@hackett.info	Family	Edit	Destroy
Breitenberg Rupert	rupert.breitenberg@sawayn.name	Work	Edit	Destroy
Brekke Delaney	brekke_delaney@smitham.info	Work	Edit	Destroy
Carroll Henri	henri_carroll@wiza.biz	Friend	Edit	Destroy

+ Create new Contact

Search

* By name

* By email

Filter results

Filter by Group

All

Family

Work

Friends

<https://github.com/cantierecreativo/admino-example>

recap

recap

avoid monoliths

concerns

view inheritance

"unix" gems

keep it testable

trust OOP

decomposition

single responsibility

object composition

thanks!



**CANTIERE
CREATIVO**

stefano verna
@steffoz