Архитектурные практики в rails

Кирилл Мокевнин, undev.ru

Контроллеры

Проблемы:

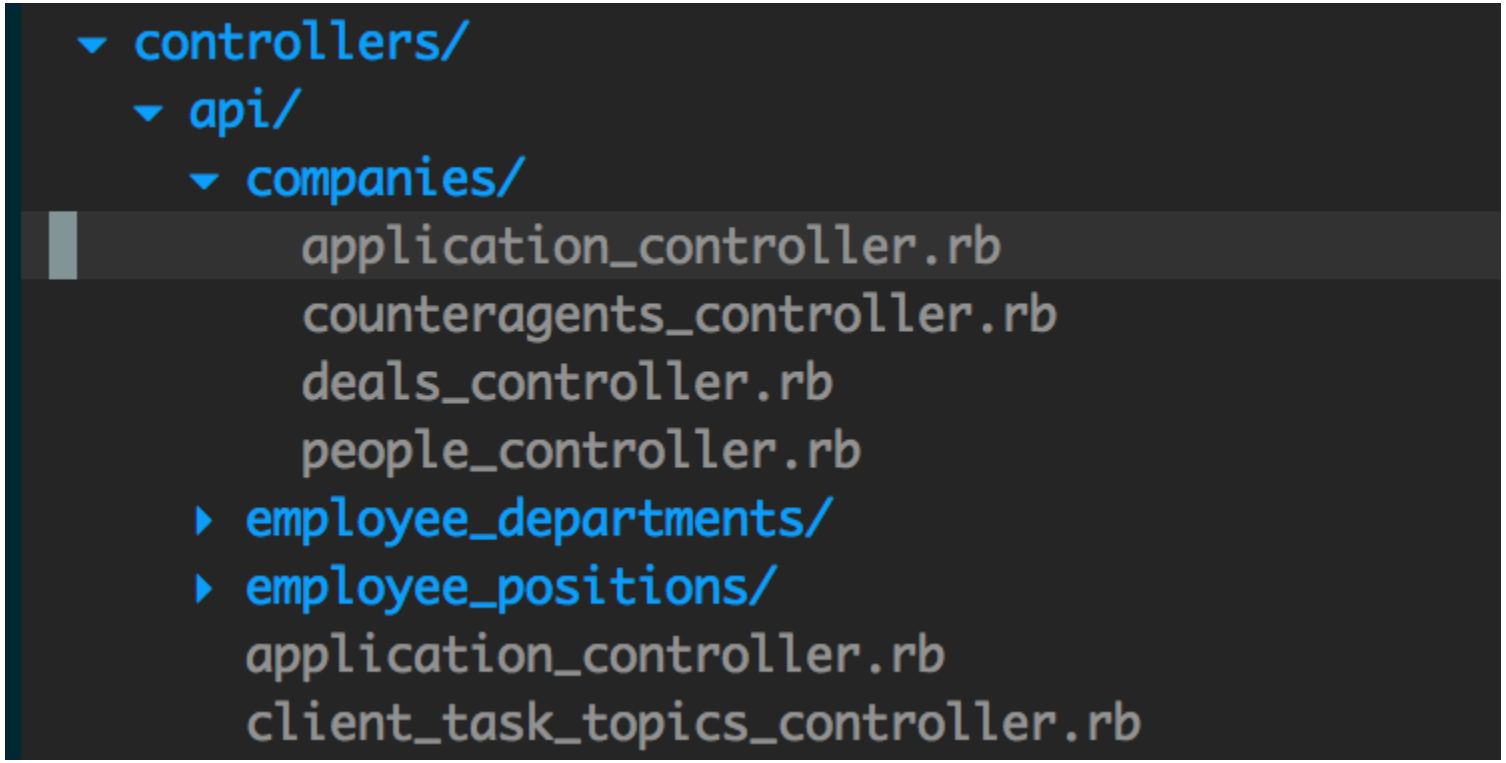
- большие файлы;
- фильтры;
- структура файлов сама по себе.

Иерархия контроллеров

/api/companies/2/people

```
namespace :api do
  resources :companies, :only => [:show] do
    scope :module => :companies do
      resources :people, :only => [:index] do
        get :available_to_me, :on => :collection
      end
      resources :deals, :only => [:index, :show]
      resources :counteragents, :only => [:index, :show]
    end
  end
```

Иерархия контроллеров



Иерархия контроллеров

```
class Api::Companies::ApplicationController < Api::Appl

  helper_method :resource_company

  def resource_company
    Company.find(params[:company_id])
  end
end
```

Иерархия контроллеров

```
class Api::Companies::PeopleController < Api::Companies::Applicati
  def index
    @companies_people = resource_company.people
  end

  def new
    @person = resource_company.people.build
  end
```

Иерархия контроллеров

Профиты:

- небольшие файлы
- _намного_ меньше фильтров и условных фильтров
- по структуре файлов понятно как устроен проект
- удобная навигация
- http://habrahabr.ru/post/136461/

API

- app/controllers/api/v1
- respond_with (respond_to :json)
- jbuilder

Ссылки в js?

gem: js-routes

Routes.user_path(3);

Сложная фильтрация с сортировками?

gem: ransack

User.ransack(params[:q])

Иерархия моделей

- Company::Order вместо CompanyOrder;
- решает конфликты имен;
- Удобнее: company.orders вместо company.company_orders
- зависимости видно по структуре файлов
- чем больше моделей, тем удобнее

Модели

fat model, wtf?

Repository

Модуль инкапсулирующий логику запросов к базе.

Repository - позволяет работать с коллекцией объектов так как будто она находится в памяти

Место: *app/repositories*

Подключение: *include CompanyRepository*

Repository

```
module Task::ServiceRepository
  extend ActiveSupport::Concern

  included do
    scope :active, where(:state => :active)
    scope :done, where(:execution_state => :done)
  end
end
```

Observers

- app/observers
- application logic (например нотификации)
- именование в соответствии с задачей

Observers

```ruby
class PersonElasticObserver < ActiveRecord::Observer
  observe :person

  def after_save(person)
    person.broadcasts.each do |b|
      b.tire.update_index
    end
  end

  def after_destroy(person)
    person.broadcasts.each do |b|
      b.tire.update_index
    end
  end
end
```

Presenters

Объект инкапсулирующий логику представления

- не имеет отношения к декораторам
- ооп альтернатива хелперам
- draper

State Machine

- не используйте флаги
- всегда используйте state machine
- куча функциональности бесплатно
- единый подход сквозь весь проект

Forms

Проблемы:

- зависимая валидация
- attr accessible
- фильтрация

Forms

Плохие решения:

- говорим модели где ее используем
- strong_params

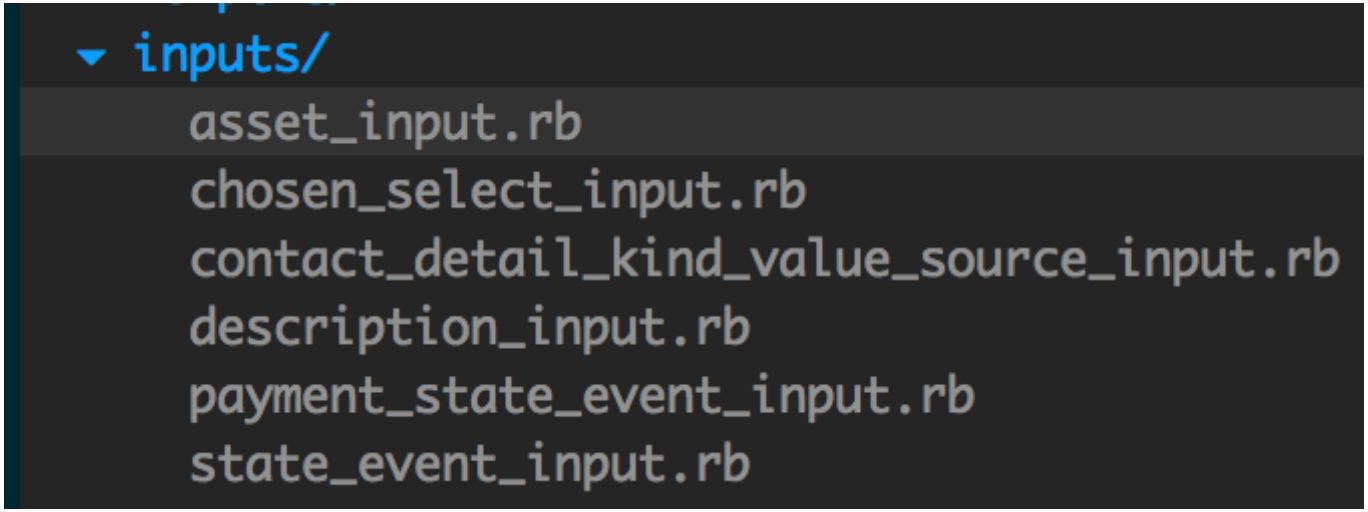
Forms

Решение: компонент forms

- формы решают все обозначенные проблемы, но вносят новый слой
- Давным-давно присутствует во всех популярных фреймворках (php, python)
- Есть несколько реализаций форм для rails
- http://habrahabr.ru/post/140684/
- примеры: http://d.pr/vgVX

custom inputs (simple form)

f.input :file, as: preview



model gems

- ancestry, materialized paths
- money, embedded/value object
- cocoon, nested forms
- carrierwave instead of paperclip

assets

- js gems
- vendorer

зло

before_validate

rake

Как тестировать?

rake

никак =),
rake всего лишь cli к вашим методам

```
namespace :app do
  namespace :mailer do
    desc "Synchronize: Timepad, Unisender & Highrise subscribers"
    task :sync, [:client_id] => :environment do |t, args|
      Sync.client args[:client_id]
    end
```

custom urls

Ссылки, которые строятся без url helpers и часто являются динамическими.

custom urls

```ruby
module CustomUrlHelpers
  def register_with_facebook_cpath
    Gateway::Facebook.url_for_callback(register_user_facebook_url)
  end

  def create_link_to_facebook_cpath
    Gateway::Facebook.url_for_callback(create_link_account_facebook_url)
  end

  def edit_company_account_curl(account)
    case account
      when ::Company::CrmAccount
        edit_company_crm_account_url(account)
      when ::Company::GisAccount
        edit_company_gis_account_url(account)
    end
  end

  def facebook_auth_cpath
    '/auth/facebook'
  end
end
```

Вопросы?

twitter: @mokevnin