Nested Attributes và gem Cocoon (P1)

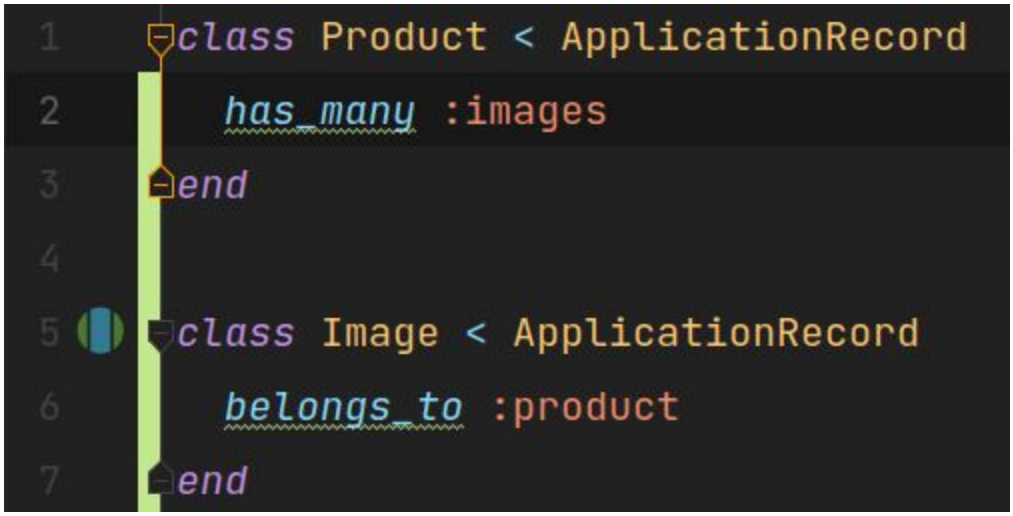
Trong bài viết hôm nay, mình sẽ giới thiệu về một khái niệm cực kỳ phổ biến trong Rails, đó là nested attributes.

1. Khái niệm

- Nested attributes cho phép chúng ta lưu những thuộc tính của bản ghi quan hệ thông qua bản ghi cha. Mặc định, nested attributes không được sử dụng trực tiếp, mà chúng ta cần khai báo trong model để chương trình hiểu được bằng cách thêm class method: accepts_nested_attributes_for.

2. Ví dụ

- Có 2 model Product và Image như sau



❖ Bình thường, nếu không dùng nested attribute, thì khi thêm mới 1 product và những images của product đó thì chúng ta sẽ phải viết 2 câu lệnh và đặt chúng trong transaction như sau:

```ruby
def create
  ActiveRecord::Base.transaction do
    @product = current_user.products.build product_params
    @product.save
    @image = Image.new(product_id: @product.id, image_params)
  end
end

private

def product_params
  params.require(:product).permit :name, :price, :description
end

def image_params
  params.require(:image).permit :url
end
```

❖ Còn với nested attributes thì mọi thứ trở nên ngắn gọn hơn bao giờ hết: Mọi câu lệnh mặc định được đặt trong transaction, và điều quan tâm là chúng ta cần thêm images_attributes trong params

```
def create
  @product = current_user.products.build product_params
end

private

def product_params
  params.require(:product).permit :name,  :price, :description,
                                  images_attributes: %i(id image)
end
```

3. Các options đi kèm

❖ Ở ví dụ trên, chúng ta thấy mới chỉ có trường hợp tạo mới, thế còn **update** và **delete** thì sao ???

➢ **Update**: Để có thể update được các thuộc tính của cả bản ghi cha và bản ghi quan hệ thì chúng ta sẽ thêm id vào trong params truyền lên (nếu thêm id thì nó sẽ áp dụng cho cả quan hệ 1-1 và 1-n, tuy nhiên, nếu quan hệ là 1-1 thì chúng ta có 1 tùy chọn khác là dùng :update_only. Mình thì thích dùng cách truyền id để đỡ phải quan tâm tới quan hệ của chúng như nào)

```ruby
def create
  @product = current_user.products.build product_params
  if @product.save
    flash[:success] = t "shop.product.create.create_success"
    redirect_to shops_root_path
  else
    flash[:danger] = t "shop.product.create.create_fail"
    render :new
  end
end

private

def product_params
  params.require(:product).permit :name, :price, :description,
                                  images_attributes: %i(id image)
end
```

➢ **Delete**: để có thể xóa được bản ghi con khi xóa bản ghi cha, thì chúng ta sẽ cần thêm option là :allow_destroy vào phần khai báo nested attributes, lưu ý vẫn cần có params id nhé.

```ruby
accepts_nested_attributes_for :images, allow_destroy: true
```

❖ Nãy giờ chúng ta mới chỉ để ý cách code trong model và controller. Ơ thế thì ngoài view chúng ta cần xử lý như nào ?

```erb
<%= form_for @product do |f| %>
  <%= f.fields_for :images do |image| %>
    <%= image.text_field :image %>
  <% end %>
<% end %>
```

- **Các options trong nested attributes**
    - **allow_destroy**: option này mặc định sẽ bị disable, nếu enable khi khi xóa bản ghi cha, tất cả các bản ghi quan hệ mà sử dụng nested attributes sẽ bị xóa theo
    - **reject_if**: cho phép sử dụng Proc hoặc gọi tới symbol mà trỏ tới method kiểm tra điều kiện. Nếu thỏa mãn điều kiện thì cả bản ghi cha và bản ghi con sẽ được thêm vào, ngược lại thì bản ghi cha được tạo còn bản ghi con thì không.
    - **limit**: cho phép chúng ta giới hạn số bản ghi con tối đa có thể tạo ra. Ví dụ chúng ta giới hạn 1 product chỉ có tối đa 4 images, lúc này, chúng ta chỉ có thể tạo tối đa 4 images
    - **update_only**: Như bên trên mình có nói, option này chỉ được dùng cho quan hệ 1-1. Khi có option này thì chúng ta có thể update bản ghi bình thường mà không cần thêm tham số id vào trong params

4. **Cuối cùng, chúng sẽ tổng hợp lại toàn bộ code từ model, view và controller nhé**

```ruby
# model
class Product < ApplicationRecord
  has_many :images

  accept_nested_attributes_for :image, allow_destroy: true
end
```

```ruby
# image
class Image < ApplicationRecord
  belongs_to :product
end
```

```ruby
# controller
class Product < ActionController::Base
  def create
    @product = current_user.products.build product_params
    if @product.save
      flash[:success] = t "shop.product.create.create_success"
      redirect_to shops_root_path
    else
      flash[:danger] = t "shop.product.create.create_fail"
      render :new
    end
  end

  private

  def product_params
    params.require(:product).permit :name,  :price, :description,
                                    images_attributes: %i(id image)
  end
end
```

Trong bài viết tiếp theo, mình sẽ nói về một gem rấy hay được sử dụng cùng với nested attributes, đó là gem cocoon.