Ruby on Rails

Recursos Múltiples

Overview

- Construir una aplicación con multiples recursos.
- scaffold_controller
- root_to

“iReviewed” App

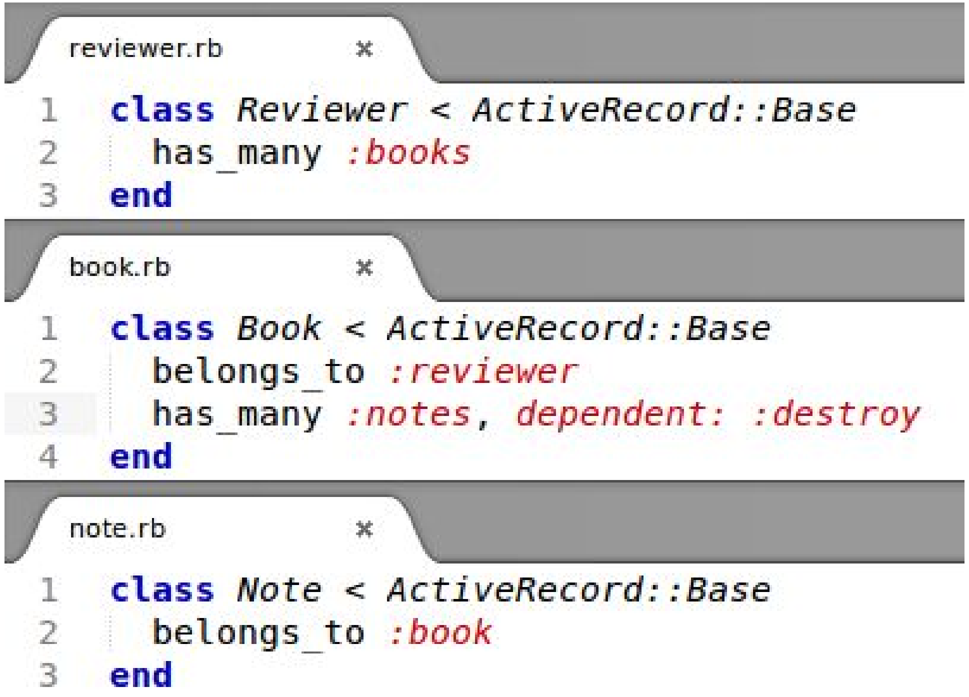
- Construyamos una aplicación que permita a un usuario escribir notas sobre los libros que ha leído.
- Necesitaremos tres tablas:
    - Reviewers
        - name, password_digest
    - Books
        - name, author, reviewer_id
    - Notes
        - title, note, book_id

```
$ rails new i_reviewed

$ cd i_reviewed
```

Creando los modelos

```
rails g model reviewer name password_digest -q

rails g model book name author reviewer:references -q

rails g model note title note:text book:references -q

rake db:migrate
```

Especificando Asociaciones


reviewer.rb

```ruby
class Reviewer < ActiveRecord::Base
  has_many :books
end
```

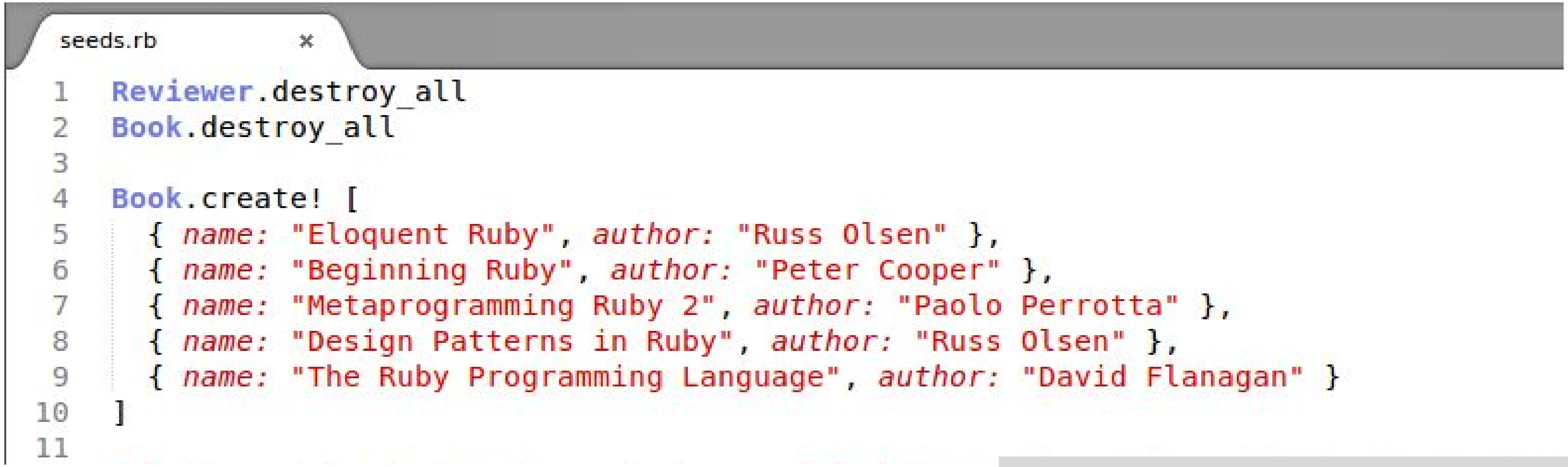
book.rb

```ruby
class Book < ActiveRecord::Base
  belongs_to :reviewer
  has_many :notes, dependent: :destroy
end
```

note.rb

```ruby
class Note < ActiveRecord::Base
  belongs_to :book
end
```

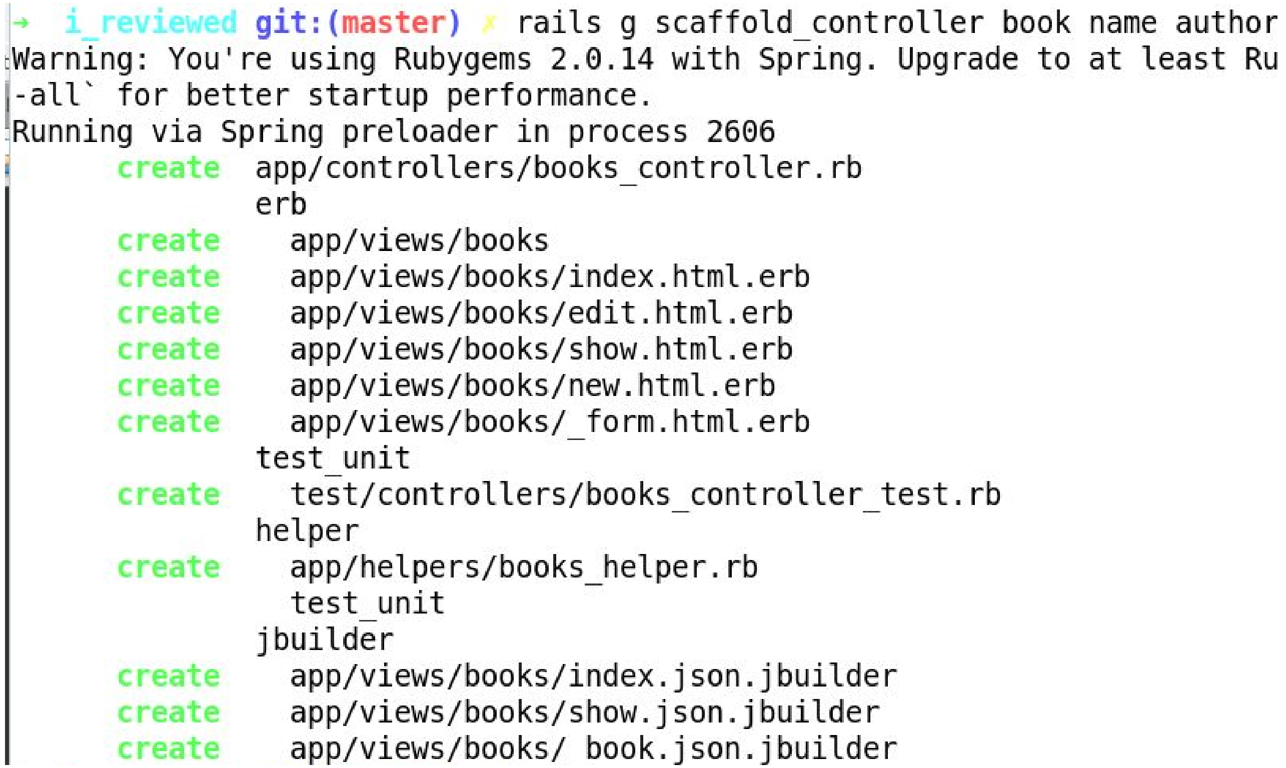
Seeds


seeds.rb ×

```ruby
Reviewer.destroy_all
Book.destroy_all

Book.create! [
  { name: "Eloquent Ruby", author: "Russ Olsen" },
  { name: "Beginning Ruby", author: "Peter Cooper" },
  { name: "Metaprogramming Ruby 2", author: "Paolo Perrotta" },
  { name: "Design Patterns in Ruby", author: "Russ Olsen" },
  { name: "The Ruby Programming Language", author: "David Flanagan" }
]

```

```
$ rake db:seed
```

rails g scaffold_controller book name author

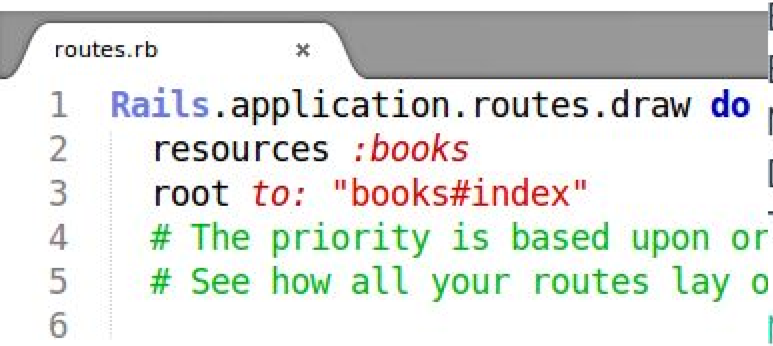


```
→  i_reviewed git:(master) ✗ rails g scaffold_controller book name author
Warning: You're using Rubygems 2.0.14 with Spring. Upgrade to at least Ru
-all` for better startup performance.
Running via Spring preloader in process 2606
      create  app/controllers/books_controller.rb
              erb
      create    app/views/books
      create    app/views/books/index.html.erb
      create    app/views/books/edit.html.erb
      create    app/views/books/show.html.erb
      create    app/views/books/new.html.erb
      create    app/views/books/_form.html.erb
              test_unit
      create    test/controllers/books_controller_test.rb
              helper
      create    app/helpers/books_helper.rb
                test_unit
              jbuilder
      create    app/views/books/index.json.jbuilder
      create    app/views/books/show.json.jbuilder
      create    app/views/books/_book.json.jbuilder
```

config/routes.rb

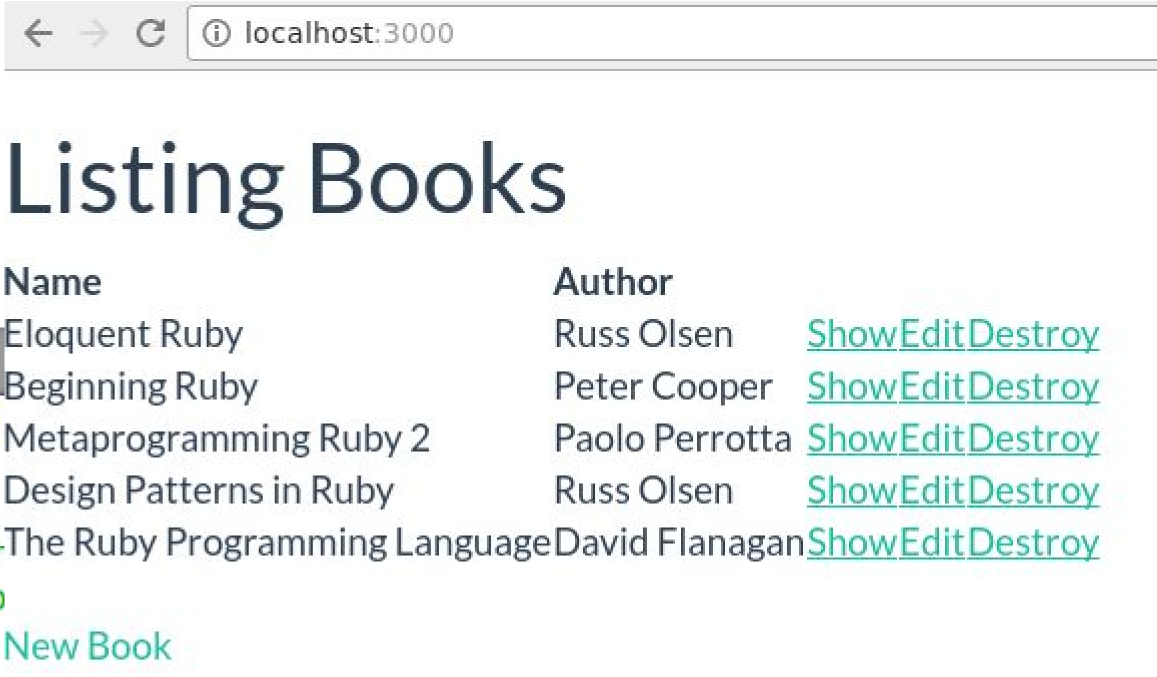
Agregamos:

```
resources :books

root to: “books#index”
```



routes.rb

```
Rails.application.routes.draw do
  resources :books
  root to: "books#index"
  # The priority is based upon or
  # See how all your routes lay o

```

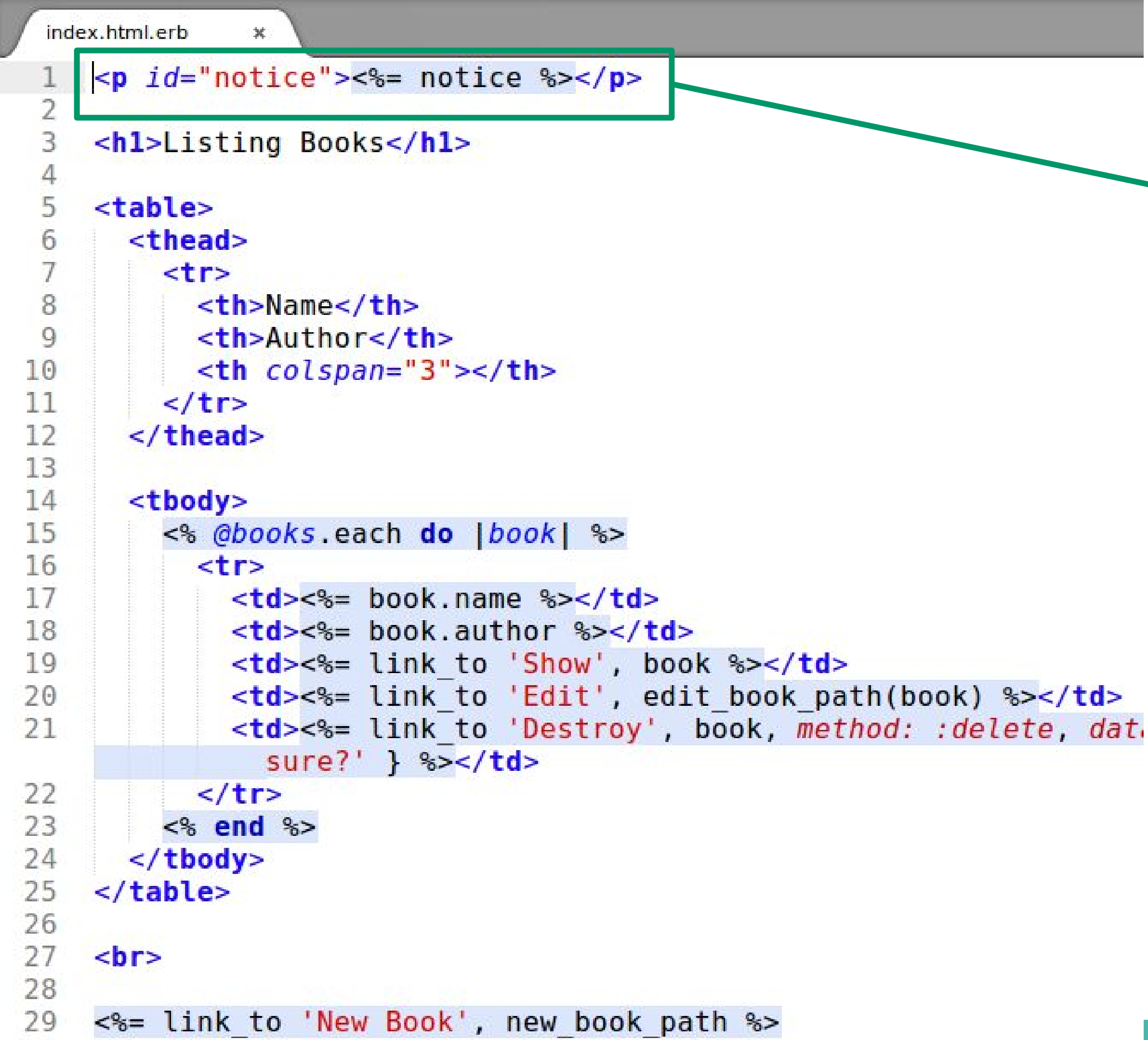


localhost:3000

Listing Books

Name	Author	
Eloquent Ruby	Russ Olsen	ShowEditDestroy
Beginning Ruby	Peter Cooper	ShowEditDestroy
Metaprogramming Ruby 2	Paolo Perrotta	ShowEditDestroy
Design Patterns in Ruby	Russ Olsen	ShowEditDestroy
The Ruby Programming Language	David Flanagan	ShowEditDestroy

New Book

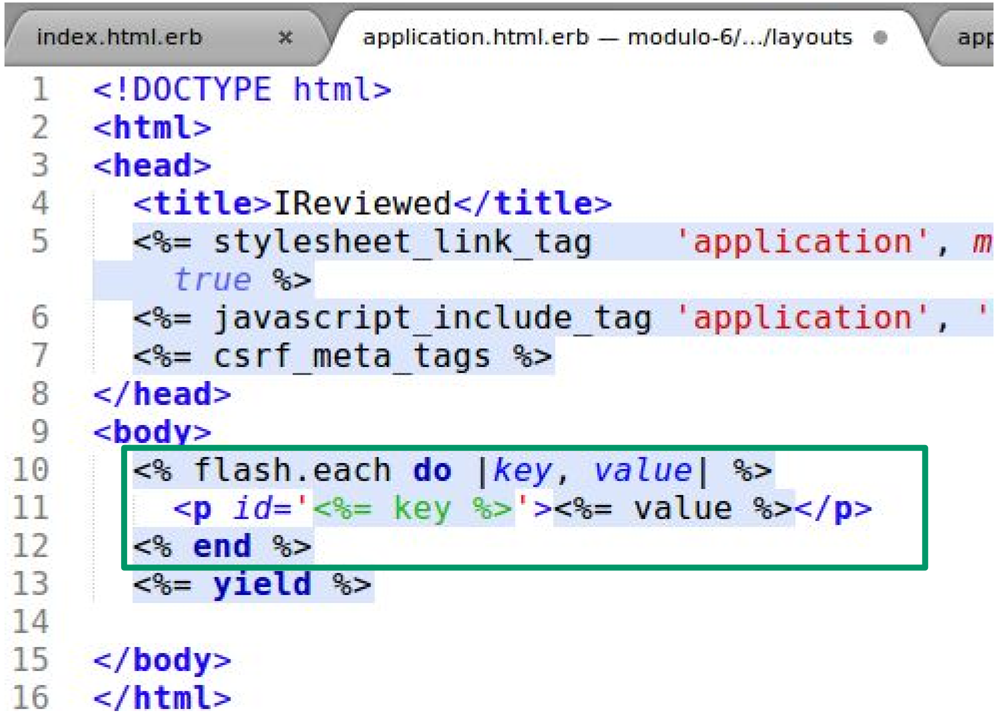


index.html.erb

```
<p id="notice"><%= notice %></p>

<h1>Listing Books</h1>

<table>
  <thead>
    <tr>
      <th>Name</th>
      <th>Author</th>
      <th colspan="3"></th>
    </tr>
  </thead>

  <tbody>
    <% @books.each do |book| %>
      <tr>
        <td><%= book.name %></td>
        <td><%= book.author %></td>
        <td><%= link_to 'Show', book %></td>
        <td><%= link_to 'Edit', edit_book_path(book) %></td>
        <td><%= link_to 'Destroy', book, method: :delete, dat
          sure?' } %></td>
      </tr>
    <% end %>
  </tbody>
</table>

<br>

<%= link_to 'New Book', new_book_path %>
```

Esto podría ir en un layout para que se pueda reutilizar entre varias vistas!

application.html.erb Layout



index.html.erb × application.html.erb — modulo-6/.../layouts app

```
<!DOCTYPE html>
<html>
<head>
  <title>IReviewed</title>
  <%= stylesheet_link_tag    'application', m
    true %>
  <%= javascript_include_tag 'application', '
  <%= csrf_meta_tags %>
</head>
<body>
  <% flash.each do |key, value| %>
    <p id='<%= key %>'><%= value %></p>
  <% end %>
  <%= yield %>

</body>
</html>
```

flash keys - `:notice, :alert`

Entonces

- El **`scaffold_controller`** se puede utilizar cuando ya se tiene el modelo.
- **`root to:`** define la raíz de la aplicación.
- El **`layout`** nos permite tener un comportamiento común