

[IF977] Engenharia de Software

Prof. Vinicius Cardoso Garcia
vcg@cin.ufpe.br :: @vinicius3w :: assertlab.com

Continuando... Hello Rails: From Zero to CRUD

AssertLab
Advanced Software and Systems
Engineering Research Technologies



Licença do material

Este Trabalho foi licenciado com uma Licença

**Creative Commons - Atribuição-NãoComercial-
CompartilhaIgual 3.0 Não Adaptada.**

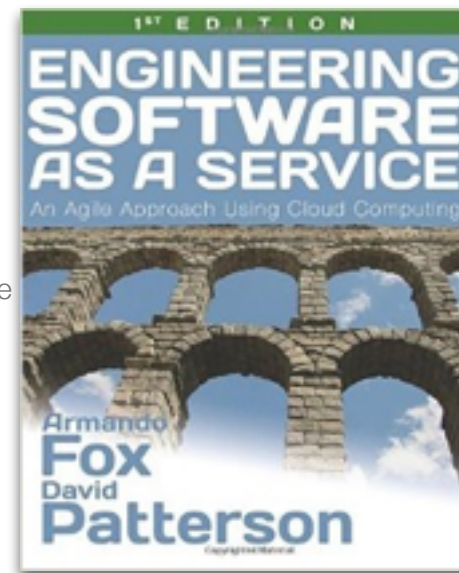
Mais informações visite

<http://creativecommons.org/licenses/by-nc-sa/3.0/deed.pt>



Referências

- A biblioteca do Desenvolvedor de Software dos dias de hoje
 - <http://bit.ly/TDOA5L>
- SWEBOK
 - Guide to the Software Engineering Body of Knowledge (SWEBOK): <http://www.computer.org/web/swebok>
- Engineering Software as a Service: An Agile Approach Using Cloud Computing (Beta Edition)
 - <http://www.saasbook.info/>

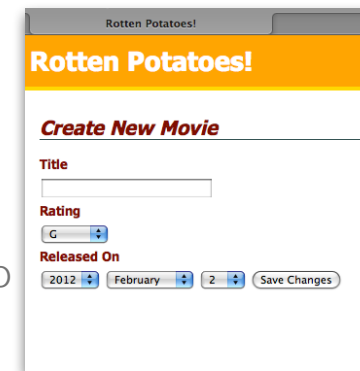




Formulários

Lidando com Formulários

- Criar um recurso normalmente leva 2 interações
 - **new**: Recupera um formulário em branco
 - **create**: Submete um formulário preenchido
- Como gerar/exibir?
- Como capturar os valores informados pelo usuário?
- O que retornar (processar)?



Rotten Potatoes!

Create New Movie

Title

Rating

G

Released On

2012 February 2

Save Changes

Rails Cookery #3



- Para criar um novo formulário de submissão:
 1. Identifique a ação que irá atender ao formulário
 2. Identifique a ação que recebe a submissão
 3. Crie as rotas, ações e visões para cada
- Atributo nome dos elementos do formulário irão aparecer como chaves no `params[]`
- Assistentes são providos para os elementos mais comuns

Criando o formulário

- Anatomia de u
- a ação e os
- somente as
- Gerando o form
- muitas veze
vez que é a
método)
- assistentes
entradas de

```
<h1>Create New Movie</h1>
<form action="/movies" method="post">
  <label for="movie_title">Title</label>
  <input id="movie_title" name="movie[title]" size="30" type="text" />
  <label for="movie_rating">Rating</label>
  <select id="movie_rating" name="movie[rating]">
    <option value="G">G</option>
    <option value="PG">PG</option>
    <option value="PG-13">PG-13</option>
    <option value="R">R</option>
    <option value="NC-17">NC-17</option>
  </select>
  <input type="submit" value="Save Changes" />
</form>
```

```
%h1 Create New Movie

= form_tag movies_path do
  -# <form action="/movies" method="post">

  = label :movie, :title, 'Title'
  -# <label for="movie_title">Title</label>

  = text_field :movie, 'title'
  -# <input id="movie_title" name="movie[title]" size="30" type="text" />

  = select :movie, :rating, ['G','PG','PG-13','R','NC-17']
  -# <select id="movie_rating" name="movie[rating]">
  -#   <option value="G">G</option>
  -#   <option value="PG">PG</option>
  -#   ...etc...
  -# </select>

  = submit_tag 'Save Changes'
  -# <input name="commit" type="submit" value="Save Changes" />
end
```

- how the form helpers are used to create a form

- naming form fields as "movie[title]", "movie[rating]", etc means you end up with params[:movies] as a hash ready to pass to create or update_attributes

<http://pastebin.com/k8Y49EhE>

<http://pastebin.com/3dGWsSq8>

Pergunta

Qual destas seria válida para gerar um formulário que, quando submetido, chamaria a ação **Create New Movie**?

- A** `= form_tag movies_path do`
 `... end`
- B** `%form{:action => movies_path,`
 `:method => :post}`
- C** `%form{:action => '/movies',`
 `:method => 'post'}`
- D** Todas as anteriores

Pergunta

Qual destas seria um Haml válido para gerar um formulário que, quando submetido, chamaria a ação **Create New Movie**?

A `= form_tag movies_path do`
 `... end`

B `%form{:action => movies_path,`
 `:method => :post}`

C `%form{:action => '/movies',`
 `:method => 'post'}`



D Todas as anteriores



Redirection, the Flash and the Session

Recebendo o formulário

- Um truque: usar depurador para inspecionar o que está acontecendo
 - comece com `rails server --debugger`
 - insira debugger onde você quer que pare
 - mais detalhes e outros comandos: ESaaS 4.7
- Atenção: `params[]` é um hash, por causa da forma que nomeamos os campos do formulário
 - Convenientemente, o que `Movie.create!` quer

11

use ruby-debug to help understand what's going on - set a breakpoint right at the top of `MoviesController#create`, show that you can print out `params[:movies]`, etc

Como a view deverá ser processada para criar a ação?

- Idioma: redirecionar o usuário para uma página mais útil.
 - e.g, lista de filmes, se a criação deu certo
 - e.g., form New Movie, se não deu certo
- Redirect desencadeia uma nova requisição HTTP
 - Como informar ao usuário por que ele foi redirecionado?
- Solução: `flash[]` - parece com um hash que persiste até o fim da próxima requisição
 - `flash[:notice]` convenientemente para informação
 - `flash[:warning]` convenientemente para "erros"

12

some actions, like create and update, COULD render their own view but that wouldn't be very helpful for the user, so it's idiomatic to use `redirect_to` to send the user elsewhere.

since `redirect` starts a new http request, we need `flash[]` .

“errors” in quotes because if, eg, Movie creation fails, the **user** may have made an error but your **app** is working correctly. hence the name 'warning' note, you can put other keys in `flash[]`, these are just the conventional ones.

Flash & Session

- `session[]`: é como um hash que persiste pra sempre
 - `reset_session` detona tudo!
 - `session.delete(:some_key)`, atua como um hash
- Por padrão, os cookies armazenam todo conteúdo se `session` & `flash`
 - Alternativa: armazenar as sessões Rails no BD
 - Google “**rails session use database**”
 - Outra alternativa: armazenar sessões em um sistema de armazenamento “NoSQL”, como um *memcache*

13

in prev lecture we briefly mentioned "NoSQL" storage solutions like sharding.

session storage is a great use case for this since there is no sharing. hence, valuable that rails provides a SEPARATE abstraction for it.

Pergunta

Ben Bitdiddle diz: "Você pode colocar objetos arbitrários (e não apenas os "simples" como ints e strings) na `session[]`". O que você faz acha?

- A. Verdade
- B. Verdade, mas não é uma boa ideia
- C. Falso, porque você não pode colocar objetos arbitrários em um hash
- D. Falso, porque `session[]` não é realmente um hash, ele só se parece com um

Pergunta

Ben Bitdiddle diz: "Você pode colocar objetos arbitrários (e não apenas os "simples" como ints e strings) na `session[]`". O que você faz acha?

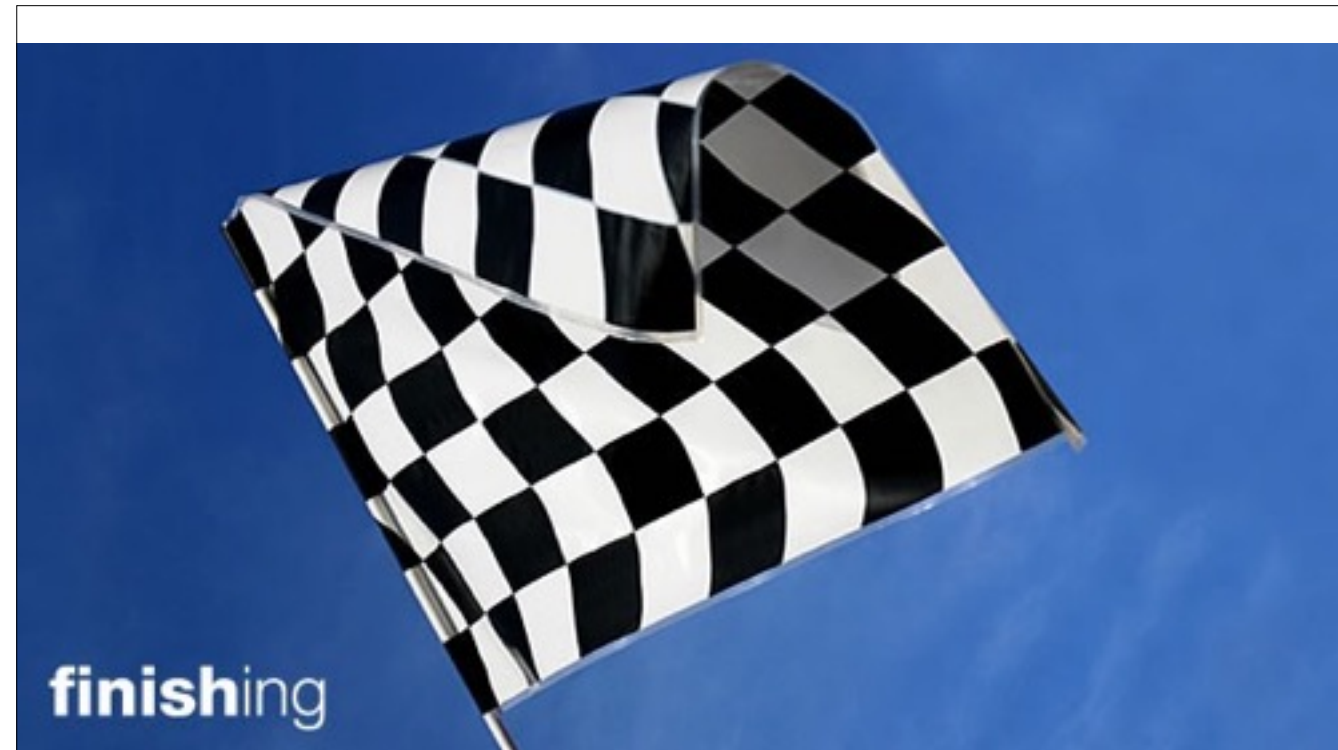
A. Verdade



B. Verdade, mas não é uma boa ideia

C. Falso, porque você não pode colocar objetos arbitrários em um hash

D. Falso, porque `session[]` não é realmente um hash, ele só se parece com um



Finalizando o CRUD

O par Editar/Atualizar é análogo ao Novo/Criar

- O que é o mesmo?
 - 1st ação recupera um form, 2nd ação submete ele
 - “submit” usa redirect (show action for movie) ao invés de processar sua própria view
- O que é diferente?
 - Form deve aparecer com valores existentes preenchidos: recuperar Filme existente primeiro <http://pastebin.com/VV8ekFcn>
 - Ações do form usam PUT ao invés de POST <http://pastebin.com/0drixxGa>

Helper method	URI returned	RESTful Route and action	
<code>movie_path(m)</code>	/movies/1	PUT /movies/:id	update
<code>movie_path(m)</code>	/movies/1	DELETE /movies/:id	destroy

Destruir é fácil

- Lembre-se, destroy é um método de instância
- Encontre o filme primeiro, então destrua-o!
- Envie o usuário de volta para o **Index**

```
def destroy
  @movie = Movie.find(params[:id])
  @movie.destroy
  flash[:notice] =
    "Movie '#{@movie.title}' deleted."
  redirect_to movies_path
end
```


Pergunta

Se você definir uma variável de instância em um método de controlador, o seu valor será retido por quanto tempo?

- A. Esta requisição e todas as requisições subsequentes
- B. Somente esta requisição e a próxima requisição
- C. Somente esta requisição - assim que a view for processada, a variável é reiniciada a nil
- D. Depende se a variável de instância foi declarada estática

Pergunta

Se você definir uma variável de instância em um método de controlador, o seu valor será retido por quanto tempo?

- A. Esta requisição e todas as requisições subsequentes
- B. Somente esta requisição e a próxima requisição
-  C. Somente esta requisição - assim que a view for processada, a variável é reiniciada a nil
- D. Depende se a variável de instância foi declarada estática



Fallacies, pitfalls, and perspectives on SaaS-on- Rails



Fat controllers & views

- Realmente fácil cair na armadilha de “fat controllers”
 - Controlador é o primeiro lugar “tocado” em seu código
 - Tentação: começar a codificar no método controlador
- Fat views
 - “All I need is this for-loop.”
 - “....and this extra code to sort the list of movies differently.”
 - “...and this conditional, in case user is not logged in.”
- Não! Isso é tarefa para model, controller, helpers!

Projeto para Arquitetura Orientada a Serviço

- Um benefício de controllers e views magros: fácil de redirecionar/conduzir seu aplicativo para SOA
- Normalmente, chamadas SOA vai esperar receber XML ou JSON (JavaScript Object Notation, parece hashes aninhados) como resultado
- Uma mudança trivial no controller faz isso

<http://pastebin.com/bT16LhJ4>

've been talking in class about benefits of SOA, and how proper MVC separation makes it easier to turn a user-facing app into a service.

Pastebin example shows use of controllers' `respond_to()` method to turn RP into a RESTful service.

Pergunta

Quais etapas são sempre necessárias quando na adição de um nova ação 'foo' no modelo Movie de um app Rails

- (a) Assegurar que existe um template para processar em `app/views/movies/foo.html.haml` (ou `.html.erb`, etc)
- (b) Assegurar que existe uma rota no `config/routes.rb`
- (c) Implementar método auxiliar (helper) para gerar URIs route-helpers necessários

- A. Somente (a) e (b)
- B. Somente (b)
- C. Somente (b) e (c)
- D. Somente (a) e (c)

Pergunta

Quais etapas são sempre necessárias quando na adição de um nova ação 'foo' no modelo Movie de um app Rails

- (a) Assegurar que existe um template para processar em `app/views/movies/foo.html.haml` (ou `.html.erb`, etc)
- (b) Assegurar que existe uma rota no `config/routes.rb`
- (c) Implementar método auxiliar (helper) para gerar URIs route-helpers necessários



- A. Somente (a) e (b)
- B. Somente (b)
- C. Somente (b) e (c)
- D. Somente (a) e (c)

Atividade

- Resolução da Lista de Exercícios 2

<http://bit.ly/if977-hw2>