

Rails 勉強会@東京 #7 ポジションペーパー

川村 徹 (かわむら / tkawa)
 tkawa@4bit.net
<http://www.4bit.net/>

DHH loves REST!!

日本 Ruby カンファレンス 2006. DHH のセッション内容がヤバイ。チケットが入手できなかったので聞けなかったけど。タイトルが「One controller, many ins, many outs」→「Discovering a world of Resources on Rails」。うひょー。世界リソース発見。

Blog などでレポートを書いているみなさんありがとうございます。とくに RoR 翻訳 Wiki(<http://techno.hippy.jp/rorwiki/?RubyKaigi2006>)の安藤さんすごい。大感謝。

オブジェクトには CRUD と呼ばれる 4 つの基本操作があって、SQL ではそれぞれ SELECT, INSERT, UPDATE, DELETE となる。コントローラーではそれらに create, show(list), update, delete が対応する。でも実は、みんな忘れがち、または知らなかったりするんだけど、Web にはもう一つの CRUD がある。それが HTTP の 4 つのメソッド、POST・GET・PUT・DELETE だ。

```
GET /people/1
```

こうできないと駄目だ。そしたら他のだって同じ様にこうなってほしいだろ。

```
POST /people
PUT /people/1
DELETE /people/1
```

しかも嬉しい事に、ここまでシンプルで一貫しているとこんな事だって考えられる。

```
ActionController::Routing::Routes.draw do |map|
  map.resources :person
end
```

このへんから推察するに、Simply RESTful プラグイン、RESTful Rails プラグインの機能(に近いもの)を標準に取り込むということでは？

Simply RESTful プラグイン

(http://dev.rubyonrails.org/browser/plugins/simply_restful)

routes.rb に map.resource というメソッドを追加。map.resource :item と書くと以下のようなルーティングが生成される。

VERB	URL structure	Rails method
GET	/items	index
GET	/items/1	show
GET	/items/new	new
POST	/items	create

VERB	URL structure	Rails method
DELETE	/items/1	destroy
POST	/items/1?_method=DELETE	destroy
GET	/items/1;edit	edit
PUT	/items/1	update
POST	/items/1?_method=PUT	update
POST	/items/1;complete	complete

これは DHH の言ってるルールと符合する。

ActiveResource

これまで説明してきた規約を利用して、Web サービスを自動化してくれるもの、それが **Active Resource** だ。Active Resource は Active Record 等と並ぶ Rails の新しいフレームワークで、Web サービスを Active Record のように利用させてくれる。

```

Person = ActiveResource::Struct.new do |p|
  p.uri "http://www.example.com/people"
  p.credentials :name => "dhh", :password => "secret"
end

# GET http://www.example.com/people/1
# => <person><name>Matz</name></person>
matz = Person.find(1)

# => matz
matz.name

david = Person.new(:name => "David")

# POST http://www.example.com/people
# <person><name>David</name></person>
# => Location: http://www.example.com/people/2 (201 Created)
david.save

# => 2
david.id

david.name = "David Heinemeier Hansson"

# PUT http://www.example.com/people
# <person><name>David Heinemeier Hansson</name></person>
david.save

```

何がヤバイって、「ActiveResource でアクセス可能な API」という合意を得て、REST API のデファクトスタンダードを形成する可能性があるってこと。細かいことを言えば find (検索) の規約とかどうするかわからないけど、OpenSearch でもいいし、とにかく Rails の機能として実装すればそれがデファクトになりうる。

例えばすごく単純に、column カラムが queryword なリソースを取ってくるには /controller?column=queryword とかでもいいんじゃない? page とかとバッティングするのがちょっとまずいけど。

実は ActiveResource に似た構想はちょっと前に考えてたんだけど(とか言ってみるテスト)。僕の着想のベースは、Ajax のクロスドメイン制約を回避するために、proxy のようなものが必要なので、それを Rails ベースで作れないかなあということだった。ActiveResource ができれば proxy どころかもっと便利に使える。さすが DHH。ぜひ実現してほしい。

参考

- rest/rails – Microformats Wiki (<http://microformats.org/wiki/rest/rails>)
DHH も関与。Microformats 方面ではコンセンサスを得ている?
- RESTful Rails プラグイン (<http://rubyforge.org/projects/restful-rails/>)
- XML.com: Putting REST on Rails (<http://www.xml.com/pub/a/2006/04/19/rest-on-rails.html>)