2006.4.16

Rails 勉強会@東京 #5 ポジションペーパー

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

第 0 回以来の久しぶりの参加です。よろしくお願いします。

Controller#respond_to は Web Application と Web Service の垣根をなくす!?

(ヽ(・∀・)ノくまくまー http://wota.jp/ac/?date=20060317 よりインスパイヤ)

Rails 1.1 の新機能。

Controller#respond_to (The accept header) 概要

1. HTTP/1.1 リクエストヘッダ中の Accept フィールドを見る
2. Content-Type に応じてリクエストのデータ形式を自動変換してくれる
3. 従って、コントローラは同じロジック(action)で対応できる
4. Accept の種類(mime タイプ)に応じて、実行する描画処理を指定できる
5. "*/*" が指定された場合は respond_to 内の最初の定義を実行する

```
respond_to do |type|
 type.html { render }
 type.yaml { render :text => @resource.to_yaml }
 type.xml  { render :text => @resource.to_xml }
end
```

さらに REST 的インターフェイスを実現するために自前でコントローラを実装

```
class RestController < AbstractRestController
 def resource
  model_name = params[:model].camelize
  model = eval(model_name) # *超危険*

   if params[:id] # id があれば Member URI
    id = params[:id]
    @resource = model.find(id)

    case request.method
    when :get
     # Retrieving a Resource
     respond_to do |type|
      type.xml { render :text => @resource.to_xml }
      type.yaml { render :text => @resource.to_yaml }
      #type.json { render :text => @resource.to_json }
      #残念ながら type.json はまだ対応していない
     end
    when :put
     # Updating a Resource (省略)
    when :delete
     # Deleting a Resource (省略)
    end

   else # id がなければ Collection URI
    case request.method
    when :get
     # Listing Collection Members (省略)
    when :post
     # Creating a Resource
     new_resource = model.new(params[params[:model].to_sym])
     if new_resource.save
      render_post_success(:id => new_resource.id)
     else
      render :text => new_resource.errors.full_messages.join("\n"), :status => 406
     end
    end
   end
 end
```

routes.rb をこんな感じに設定。

```
map.connect 'rest/:model/:id', :controller => 'rest', :action
=> 'resource', :id => nil
```

/rest/<モデル名>/<ID>という統一された URI (Atom でいう Member URI)でリソースにアクセスでき、さらに”Accept: application/xml”をつけると XML で取得、”Accept: application/x-yaml”をつけると YAML で取得したりできる。

さらに取得だけではなく、/rest/<モデル名>という URI (Collection URI)に”Content-Type: application/xml”をつけて XML を POST すれば

```
% telnet localhost 3000
POST /rest/member HTTP/1.1
Host: localhost
Accept: application/xml
Content-Type: application/xml
Content-Length: 86

<member>
 <name>saki</name>
 <birthday type="date">1991-11-22</birthday>
</member>
```

通常の FORM 形式に変換して

```
member[name]=saki&member[birthday]=1991-11-22
```

データを CREATE してくれる[*1]。

アプリケーションのすべてが REST である必要はないが（というか無理）、Ajax で使われるインターフェイスなどはとくにこんな感じで簡単に REST が使えると、Web アプリケーションが即 Web サービスとしても使えるようになるのでうれしい。

参考

- RESTWiki (http://restwiki.rails2u.com/)
- RESTful Wiki の実装 (http://rails2u.com/tmp/ppt/rest051124.ppt)
- XML.com: REST on Rails (http://www.xml.com/pub/a/2005/11/02/rest-on-rails.html)
- [uf-rest] RESTified Rails Controller

  (http://microformats.org/discuss/mail/microformats-rest/2005-November/000042.html)

*1くまくまーでは「うまく動きませんでした」って書いてあるけど、Rails 1.1.2 でちゃんと動いた。ただし Content-Length ヘッダが必要。