37signals

Discovering a world of Resources on Rails

David Heinemeier Hansson

Create Read Update Delete

What they told you...

やつらが言ってたのは...

Simplistic	単純すぎる
Unfulfilling	物足りない
Unworthy	ふさわしくない
Shameful	これはひどい

They were wrong

やつらはまちがっていた

How I learned to stop worrying and love the CRUD

私は如何にして心配するのを止めて,CRUDを愛するようになったか?

find	create	update	destroy
SELECT	INSERT	UPDATE	DELETE

GET	POST	PUT	DELETE
find	create	update	destroy
SELECT	INSERT	UPDATE	DELETE

```
POST /people/create
GET  /people/show/1
POST /people/update/1
POST /people/destroy/1
```

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

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

```ruby
class PeopleController < ActionController::Base
  # POST /people
  def create() end

  # GET /people/1
  def show() end

  # PUT /people/1
  def update() end

  # DELETE /people/1
  def destroy() end
end
```

```ruby
class PeopleController < ActionController::Base
  # POST /people
  def create() end

  # GET /people/1
  def show() end

  # PUT /people/1
  def update() end

  # DELETE /people/1
  def destroy() end
end
```

```ruby
form_for :person, david do |f|
  f.text_field :name
end
```

```ruby
class PeopleController < ActionController::Base
  # POST /people
  def create() end

  # GET /people/1
  def show() end

  # PUT /people/1
  def update() end

  # DELETE /people/1
  def destroy() end
end
```

```ruby
link_to "David",
  person_url(david)
```

```ruby
class PeopleController < ActionController::Base
  # POST /people
  def create() end

  # GET /people/1
  def show() end

  # PUT /people/1
  def update() end

  # DELETE /people/1
  def destroy() end
end
```

```ruby
form_for :person, david,
    :method => :put do |f|
  f.text_field :name
end
```

```ruby
class PeopleController < ActionController::Base
  # POST /people
  def create() end

  # GET /people/1
  def show() end

  # PUT /people/1
  def update() end

  # DELETE /people/1
  def destroy() end
end
```

```ruby
link_to "Destroy",
  person_url(david),
  :method => :delete
```

Why bother?

これでよくね?

Consistency 一貫性

Simplicity 単純さ

Discoverability 見つけやすさ

Constraints are liberating (a straight jacket for your mind)

制約が自由をもたらす

（あなたの思考の拘束衣）

```ruby
class PeopleController < ActionController::Base
  # GET /people
  def index() end

  # GET /people;new
  def new() end

  # POST /people
  def create() end

  # GET /people/1
  def show() end

  # GET /people/1;edit
  def edit() end

  # PUT /people/1
  def update() end

  # DELETE /people/1
  def destroy() end
end
```

```ruby
class Group < ActiveRecord::Base
  has_and_belongs_to_many :users
end

class User < ActiveRecord::Base
  has_and_belongs_to_many :groups
end
```

```ruby
class GroupsController < ActionController::Base
  # POST /groups/1;add_user?user_id=2
  def add_user() end

  # POST /groups/1;add_user?user_id=2
  def remove_user() end
end
```

```ruby
class Group < ActiveRecord::Base
  has_and_belongs_to_many :users
end

class User < ActiveRecord::Base
  has_and_belongs_to_many :groups
end
```

```ruby
class UsersController < ActionController::Base
  # POST /users/1;join_group?group_id=2
  def join_group() end

  # POST /users/1;leave_group?group_id=2
  def leave_group() end
end
```

```ruby
class Group < ActiveRecord::Base
  has_many :memberships
  has_many :users, :through => :memberships
end

class User < ActiveRecord::Base
  has_many :memberships
  has_many :groups, :through => :memberships
end

class Membership < ActiveRecord::Base
  belongs_to :group
  belongs_to :user
end


class MembershipsController < ActionController::Base
  # POST /memberships?group_id=1&user_id=2
  def create() end

  # DELETE /memberships/3
  def destroy() end
end
```

```ruby
class Account < ActiveRecord::Base
  def upgrade_to(plan)
    if eligible_for_upgrade?(plan)
      update_attribute :plan_id, plan
      true
    else
      false
    end
  end
end

class Plan < ActiveRecord::Base
end

class AccountController < ActionController::Base
  def upgrade_plan
    if @account.upgrade_to(Plan[params[:plan_name]])
      redirect_to :action => "index"
    else
      flash[:notice] = "Account not eligible for upgrade"
      render
    end
  end
end
```

```ruby
class Account < ActiveRecord::Base
  has_many :subscriptions
end

class Subscription < ActiveRecord::Base
  belongs_to :account
  belongs_to :plan

  protected
    def validate_on_create
      unless eligible_for_upgrade?
        add_to_base("Account not eligible for upgrade")
      end
    end
end

class Plan < ActiveRecord::Base
end

class SubscriptionsController < ActionController::Base
  def create
    @account.subscriptions.create(:plan => Plan[params[:plan_name]]) ?
      redirect_to(:action => "index") :
      render
  end
end
```

Model beyond “things”

「物」じゃないモデル

Relations (membership, subscription)

関連(帰属関係、サブスクリプション)

Events (closure, change)

イベント(終了、変更)

States (reviewed, accepted)

状態(確認済み、受理済み)

But!

だがしかし！

CRUD is not a goal, it's an aspiration, a design technique

CRUDはゴールではなく
ゴールを目指す気持ちであり
設計の技法なのである

```ruby
class Kase < ActiveRecord::Base
  def close!(at = Time.now)
    self.closed_at = at
    self.closed = true
    save!
  end
end
```

```ruby
class KasesController < ActionController::Base
  # POST /kases/1;close
  def close
    @kase.close!
    redirect_to :action => "show"
  end
end
```

```
POST    /kases/1;close
        /identity;aspect
```

```
GET     /kases/1;edit
        /identity;view
```

```ruby
class Closure < ActiveRecord::Base
  belongs_to :kase
  belongs_to :closer, :class_name => "Person"
end

class ClosuresController < ActionController::Base
  def create
    Closure.create!(:kase => @kase, :closer => @user)
    redirect_to(kase_url(@kase))
  end
end
```

```ruby
class Kase < ActiveRecord::Base
  has_one :progress
end

class Progress < ActiveRecord::Base
  belongs_to :kase
  belongs_to :initiator, :class_name => "Person"
end

class Opened < Progress
end

class Reviewed < Progress
  belongs_to :verifier, :class_name => "Person"
end

class Closed < Progress
end
```

And there is more!

さらに!

Answering to mime types

mime type に応答する

One controller for many clients

いろいろなクライアントをさばく1つのコントローラ

One action returning different results

複数の結果を返す1つのアクション

Flexible input model

柔軟な入力モデル

```ruby
class PeopleController < ActionController::Base
  def index
    @people = Person.find(:all)

    respond_to do |format|
      format.html # renders index.rhtml
      format.js   # renders index.rjs
      format.xml  { render :xml  => @people.to_xml }
      format.icl  { render_calendar(@people) }
      format.atom do
        render :action => "atom", :content_type => Mime::ATOM
      end
    end
  end
end
```

```ruby
class PeopleController < ActionController::Base
  def index
    @people = Person.find(:all)

    respond_to do |format|
      format.html # renders index.rhtml
      format.js   # renders index.rjs
      format.xml  { render :xml  => @people.to_xml }
      format.icl  { render_calendar(@people) }
      format.atom do
        render :action => "atom", :content_type => Mime::ATOM
      end
    end
  end
end
```

```
GET /people
=> returns HTML

Accept: text/javascript
GET /people
=> returns RJS
```

```
GET /people.xml
=> returns XML

Accept: text/html
GET /people.xml
=> returns XML
```

```ruby
class PeopleController < ActionController::Base
  def index
    @people = Person.find(:all)

    respond_to do |format|
      format.html # renders index.rhtml
      format.js   # renders index.rjs
      format.xml  { render :xml  => @people.to_xml }
      format.icl  { render_calendar(@people) }
      format.atom do
        render :action => "atom", :content_type => Mime::ATOM
      end
    end
  end
end
```

```
GET /people
=> returns HTML

Accept: text/javascript
GET /people
=> returns RJS
```

```
GET /people.xml
=> returns XML

Accept: text/html
GET /people.xml
=> returns XML
```

```ruby
class PeopleController < ActionController::Base
  def index
    @people = Person.find(:all)

    respond_to do |format|
      format.html # renders index.rhtml
      format.js   # renders index.rjs
      format.xml  { render :xml  => @people.to_xml }
      format.icl  { render_calendar(@people) }
      format.atom do
        render :action => "atom", :content_type => Mime::ATOM
      end
    end
  end
end
```

```
GET /people
=> returns HTML

Accept: text/javascript
GET /people
=> returns RJS
```

```
GET /people.xml
=> returns XML

Accept: text/html
GET /people.xml
=> returns XML
```

```ruby
class PeopleController < ActionController::Base
  def index
    @people = Person.find(:all)

    respond_to do |format|
      format.html # renders index.rhtml
      format.js   # renders index.rjs
      format.xml  { render :xml  => @people.to_xml }
      format.icl  { render_calendar(@people) }
      format.atom do
        render :action => "atom", :content_type => Mime::ATOM
      end
    end
  end
end
```

```
GET /people
=> returns HTML

Accept: text/javascript
GET /people
=> returns RJS
```

```
GET /people.xml
=> returns XML

Accept: text/html
GET /people.xml
=> returns XML
```

```ruby
class PeopleController < ActionController::Base
  def index
    @people = Person.find(:all)

    respond_to do |format|
      format.html # renders index.rhtml
      format.js   # renders index.rjs
      format.xml  { render :xml  => @people.to_xml }
      format.icl  { render_calendar(@people) }
      format.atom do
        render :action => "atom", :content_type => Mime::ATOM
      end
    end
  end
end
```

```
GET /people
=> returns HTML

Accept: text/javascript
GET /people
=> returns RJS
```

```
GET /people.xml
=> returns XML

Accept: text/html
GET /people.xml
=> returns XML
```

```ruby
class PeopleController < ActionController::Base
  def create
    @person = Person.create(params[:person])

    respond_to do |format|
      format.html { redirect_to :action => "index" }
      format.js   # renders create.rjs
      format.xml  do
        headers["Location"] = person_url(@person)
        render :nothing => true
      end
    end
  end
end
```

```ruby
class PeopleController < ActionController::Base
  def create
    @person = Person.create(params[:person])

    respond_to do |format|
      format.html { redirect_to :action => "index" }
      format.js   # renders create.rjs
      format.xml  do
        headers["Location"] = person_url(@person)
        render :nothing => true
      end
    end
  end
```

```
POST /people
person[name]=David
=> returns a redirect to index

Content-Type: application/xml
POST /people
<person>
<name>David</name>
</person>
=> returns location
```

```
POST /people.xml
person[name]=David
=> returns location

Accept: text/javascript
POST /people
person[name]=David
=> returns RJS
```

```ruby
class PeopleController < ActionController::Base
  def create
    @person = Person.create(params[:person])

    respond_to do |format|
      format.html { redirect_to :action => "index" }
      format.js   # renders create.rjs
      format.xml  do
        headers["Location"] = person_url(@person)
        render :nothing => true
      end
    end
  end
```

```
POST /people
person[name]=David
=> returns a redirect to index

Content-Type: application/xml
POST /people
<person>
<name>David</name>
</person>
=> returns location
```

```
POST /people.xml
person[name]=David
=> returns location

Accept: text/javascript
POST /people
person[name]=David
=> returns RJS
```

```ruby
class PeopleController < ActionController::Base
  def create
    @person = Person.create(params[:person])

    respond_to do |format|
      format.html { redirect_to :action => "index" }
      format.js   # renders create.rjs
      format.xml  do
        headers["Location"] = person_url(@person)
        render :nothing => true
      end
    end
  end
end
```

```
POST /people
person[name]=David
=> returns a redirect to index
```

```
POST /people.xml
person[name]=David
=> returns location
```

```
Content-Type: application/xml
POST /people
<person>
<name>David</name>
</person>
=> returns location
```

```
Accept: text/javascript
POST /people
person[name]=David
=> returns RJS
```

```ruby
class PeopleController < ActionController::Base
  def create
    @person = Person.create(params[:person])

    respond_to do |format|
      format.html { redirect_to :action => "index" }
      format.js   # renders create.rjs
      format.xml  do
        headers["Location"] = person_url(@person)
        render :nothing => true
      end
    end
  end
end
```

```
POST /people
person[name]=David
=> returns a redirect to index

Content-Type: application/xml
POST /people
<person>
<name>David</name>
</person>
=> returns location
```

```
POST /people.xml
person[name]=David
=> returns location

Accept: text/javascript
POST /people
person[name]=David
=> returns RJS
```

```ruby
class PeopleController < ActionController::Base
  def create
    @person = Person.create(params[:person])

    respond_to do |format|
      format.html { redirect_to :action => "index" }
      format.js   # renders create.rjs
      format.xml  do
        headers["Location"] = person_url(@person)
        render :nothing => true
      end
    end
  end
```

```
POST /people
person[name]=David
=> returns a redirect to index

Content-Type: application/xml
POST /people
<person>
<name>David</name>
</person>
=> returns location
```

```
POST /people.xml
person[name]=David
=> returns location

Accept: text/javascript
POST /people
person[name]=David
=> returns RJS
```

```ruby
# config/environment.rb
Mime::Type.register :mobile, "text/x-mobile"

# config/routes.rb
ActionController::Routing::Routes.draw do |map|
  map.connect 'mobile/people', :controller => "people", :format => "mobile"
end

# controller
class PeopleController < ActionController::Base
  def index
    @people = Person.find(:all)

    respond_to do |format|
      format.html
      format.mobile { render :action => "index_mobile" }
    end
  end
end
```

One more thing

あとひとつ

Active Resource

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

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

# => matz
matz.name
```

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

```
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>
# => (200 OK)
david.save
```

```ruby
class SunriseResource < ActiveResource::Base
  uri "http://www.example.com"
  credentials :name => "dhh", :password => "secret"
end

class Person < SunriseResource
  def first_name
    name.split(" ").first
  end
end

david = Person.find(5)

# => "David Heinemeier Hansson"
david.name

# => "David"
david.first_name
```

```ruby
class Kase < SunriseResource
  def close!
    update(custom(:close))
  end
end

k = Kase.find(1)

# POST http://www.example.com/kases/1;close
# =>
# <kase>
#   <closed type="boolean">true</closed>
#   <closed_on type="date">2006-06-10</closed_on>
# </kase>
k.close!

k.closed?   # => true
k.closed_at # => Date.new(2006, 6, 10)
```

```ruby
class Requester
end

class AsynchronousRequester < Requester
end

Resource.requester = AsynchronousRequester.new
```

www.rubyonrails.org