web application frameworks

Daniel Jackson

For now, all rights reserved. Daniel Jackson, 2013.
basic server side architecture

Apache ➔ Rails ➔ MySQL

request ➔ response
what a framework does

basic areas of built-in functionality
› routing
› request data
› data mapping
› templates
› validation
› session state
routing

separation of concerns
› client’s name for op vs. developer’s name for method

example
› request        GET /photos/17
› routes.rb      match 'photos/:id' => 'photos#show'
› call           class PhotosController ...
                 def show
                 ... params[:id] ...
                 end

for more, see http://guides.rubyonrails.org/routing.html
› bidirectional
› RESTful default routing
request data

two kinds of request data
  › query strings from GETs
  › form data from POSTs

read in the same way
  › params[:field]

```ruby
class ClientsController < ActionController::Base
  # sample URL: /clients?status=activated
  def index
    if params[:status] == "activated"
      @clients = Client.activated
    else
      @clients = Client.unactivated
    end
  end
end
```
web app = data conversion

**update request**

**tipster: share tips with friends**

*home search add entity logout*

Reviewing Clover

**show request**

**tipster: share tips with friends**

*home search login register*

Chloe Closure [Edit] (4 days ago)  ★★★★★
My favorite food truck. Delicious vegetarian dishes, and relatively low ecoli count.

Ben Bitdiddle (4 days ago)
★★★★★
I like this place too. And they have really cool strategy for taking orders, with people outside the ...more

Alice Alert (4 days ago)
★★★★★
Yeah, sure, the food is good. But what about the atmosphere? Especially in winter when it's snowing.

- **request**

<table>
<thead>
<tr>
<th>id</th>
<th>rating</th>
<th>content</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>yummy!</td>
</tr>
</tbody>
</table>

- **database table**

<table>
<thead>
<tr>
<th>id</th>
<th>by</th>
<th>content</th>
<th>rating</th>
<th>about</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>yummy!</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>neat</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

- **response**

<table>
<thead>
<tr>
<th>id</th>
<th>rating</th>
<th>content</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>yummy!</td>
</tr>
</tbody>
</table>
data mapping

basic idea
› object in heap = row in table

object relational mapper
› generates database schema from class defs
› backs up object methods with SQL updates/queries
class Client < ActiveRecord::Base
  has_one :address
  has_many :orders
  has_and_belongs_to_many :roles
end

class Address < ActiveRecord::Base
  belongs_to :client
end

address = client.address

# Find the client with primary key (id) 10
client = Client.find(10)

SELECT * FROM clients WHERE (clients.id = 10) LIMIT 1
validation

built in functions to check

class Person < ActiveRecord::Base
  validates :name, :length => { :minimum => 2 }
  validates :bio, :length => { :maximum => 500 }
  validates :password, :length => { :in => 6..20 }
  validates :registration_number, :length => { :is => 6 }
end

messages to be displayed in flash

class Coffee < ActiveRecord::Base
  validates :size, :inclusion => { :in => %w(small medium large), :message => "%{value} is not a valid size" }
end
**session state**

**session variable holds session state**

```ruby
session[:user_id] = @current_user.id
User.find(session[:user_id])
```

**where’s session state stored?**

- in a (hidden) database table; cookie just holds id
- or entirely in cookie

**what’s the tradeoff?**