

RAILS FOR NOOBS

Part II of the world acclaimed
Ruby on Rails tutorial

by João Girão



Chuck Norris



WHY USE RAILS?

Intermission

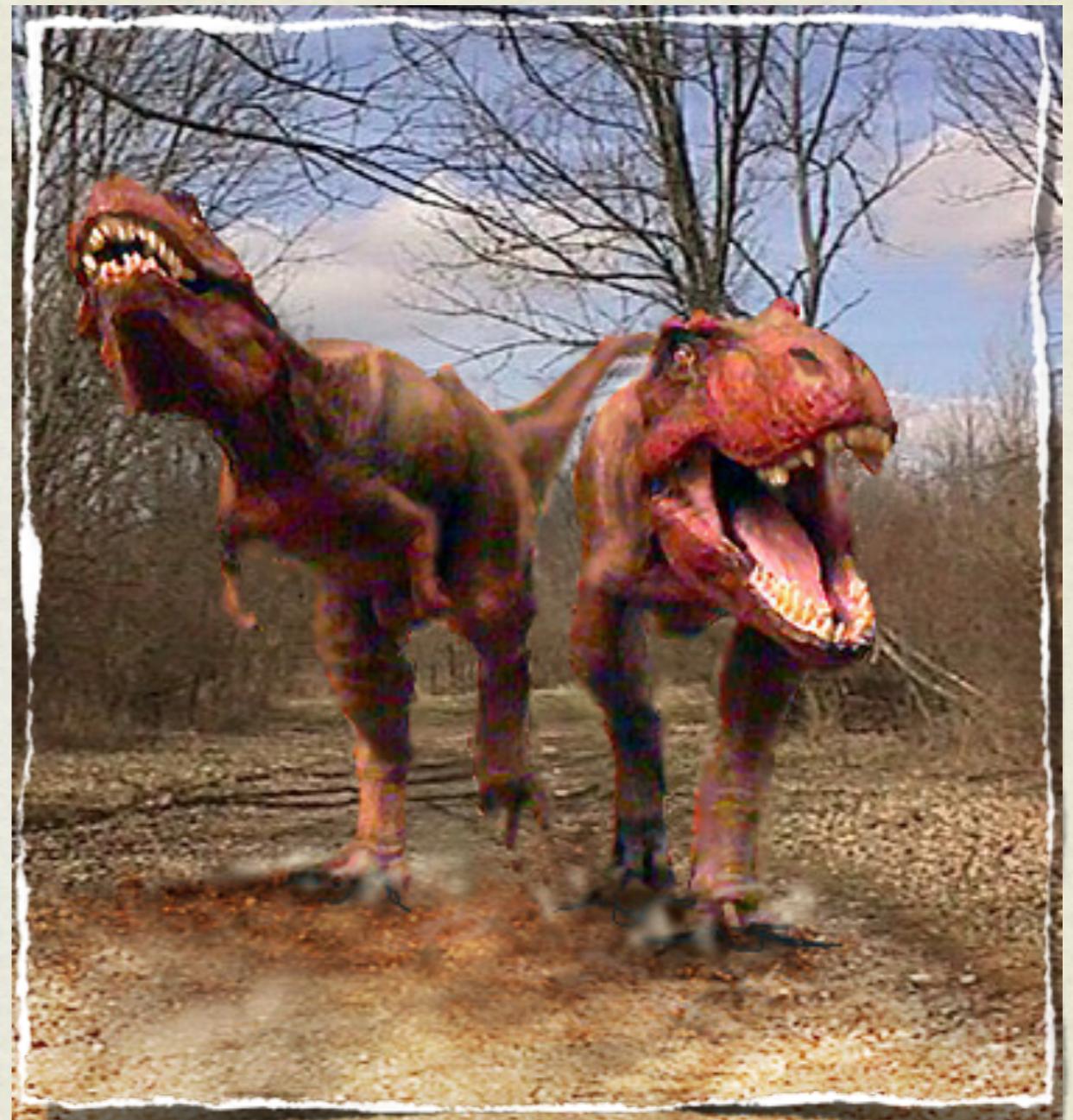
WHY RAILS, WHY NOW

- Fast to develop
 - Easy to learn
 - Flexible
- Fast to deploy
 - e.g. heroku
- Huge community
 - alexandria of addons
 - google has always the answer
- Marketable skills



PHP, JSP, ASP, .NET...

- Rails has more magic
 - not always good, but enough in 95% of cases
- Cleaner, pre-structured
 - easier to deploy in large scale environments
- Access to all the code
 - enhance/change what you want
- Modern and lots of activity



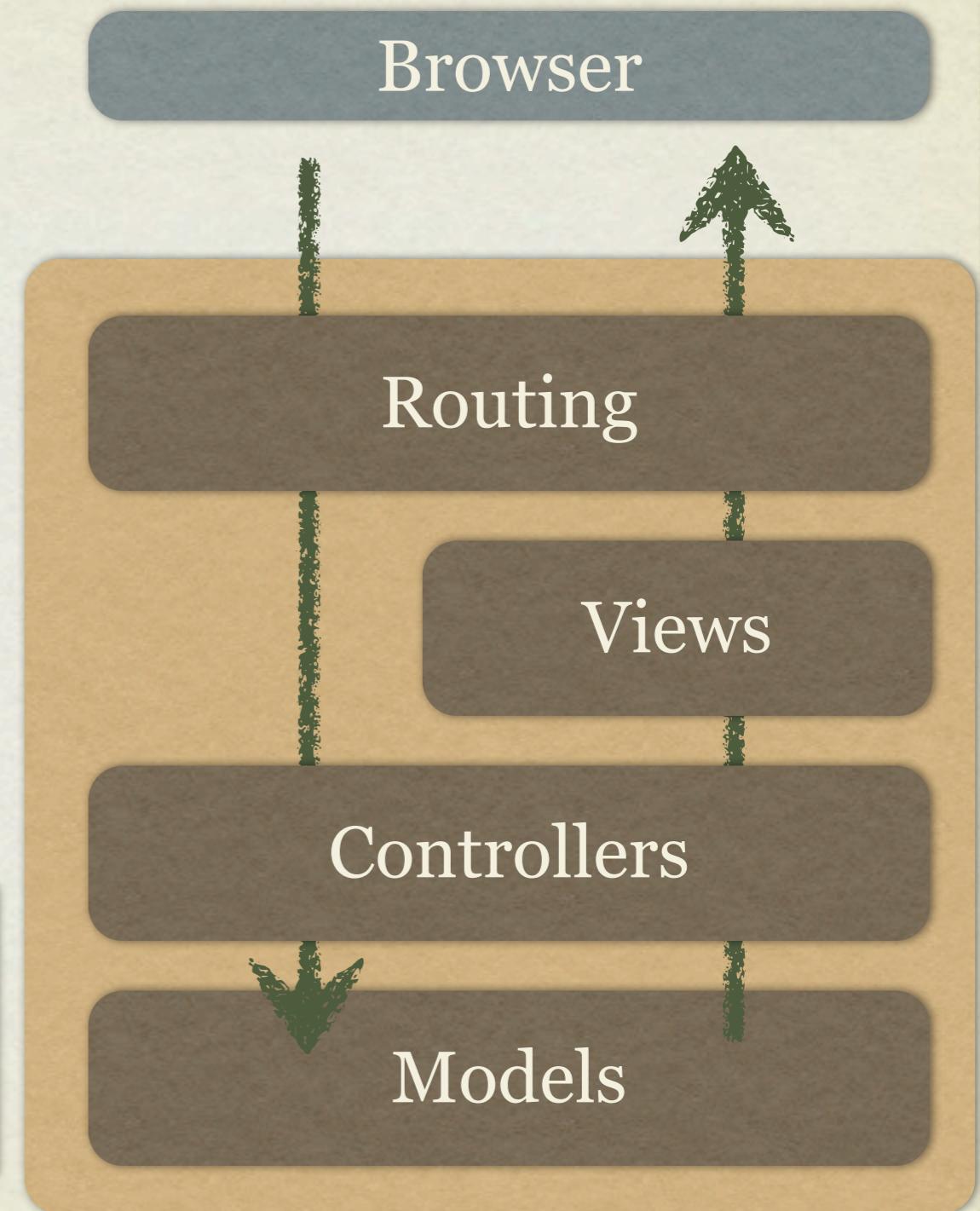


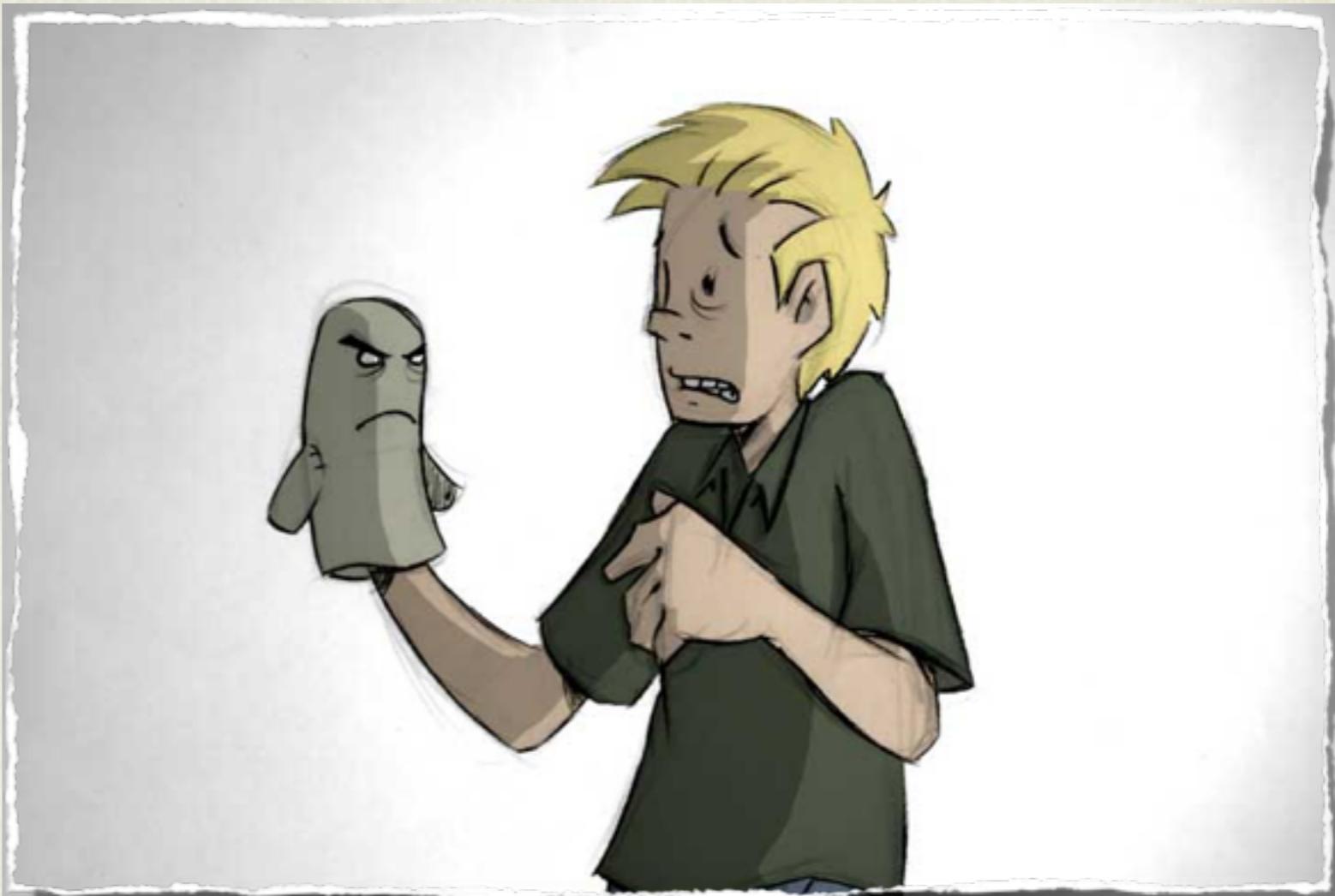
DOG EAT DOG WORLD
...even if you wear high heels

RECAP



- Routing: which controller handles which URL
- Controller: application logic
- Models: data & storage
- Views: render and user interaction





CONTROLLERS

Part 4

GOALS



- Create a controller
- CRUD
- Accessing Models
- More about routes.rb
- respond_to, respond_with
- Filters
- Verification
- Forgery Protection
- send_file, send_data



CONTROLLER [ENTER]

Create a new project and a new controller:

```
$ rails new proj ; cd proj ; bundle ; rake db:migrate ; rails s  
$ rails g controller authors
```

A controller is responsible for orchestration

```
create  app/controllers/authors_controller.rb  
invoke  erb  
create    app/views/authors  
invoke  test_unit  
create    test/functional/authors_controller_test.rb  
invoke  helper  
create    app/helpers/authors_helper.rb  
invoke  test_unit  
create    test/unit/helpers/authors_helper_test.rb
```



app/controllers/authors_controller.rb



app/helpers/authors_helper.rb

CRUD

Actions

C **reate**

new
create

R **ead**

index
show

U **pdate**

edit
update

D **elete**

destroy



ROUTES

Editing our routes:



config/routes.rb

```
Publicationdb::Application.routes.draw do
  resources :authors
  root :to => 'authors#index'
end
```

Looking at the routes, what does it mean?

```
$ rake routes
      authors POST   /authors(.:format)      {:action=>"create", :controller=>"authors"}
    new_authors GET    /authors/new(.:format)   {:action=>"new", :controller=>"authors"}
edit_authors GET    /authors/edit(.:format)  {:action=>"edit", :controller=>"authors"}
               GET    /authors(.:format)      {:action=>"show", :controller=>"authors"}
                  PUT    /authors(.:format)      {:action=>"update", :controller=>"authors"}
                 DELETE /authors(.:format)      {:action=>"destroy", :controller=>"authors"}
            root      /(.:format)          {:controller=>"authors", :action=>"index"}
```

CONTROLLER ACTIONS

```
class AuthorsController < ApplicationController
  def index
  end

  def show
  end

  def new
  end

  def create
  end

  def edit
  end

  def update
  end

  def destroy
  end
end
```



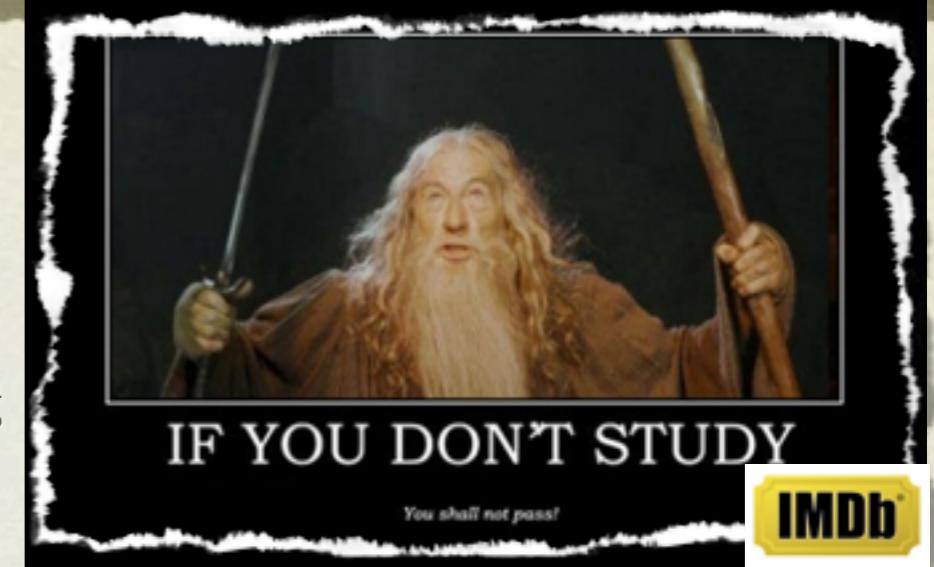
RENDER



Warning:

Normally you should not render text or inline in controllers... that's what views are for.
There are of course exceptions but we are doing here because we haven't reached views yet.

Or in the words of Gandalf



Rendering some text:

```
render :test => "This is some text"  
render :inline => "This is some more text: <%= 'some more text' %>"
```

We will talk more about render later today

and a lot more when we talk about views

PARAMETERS



Printing a param:

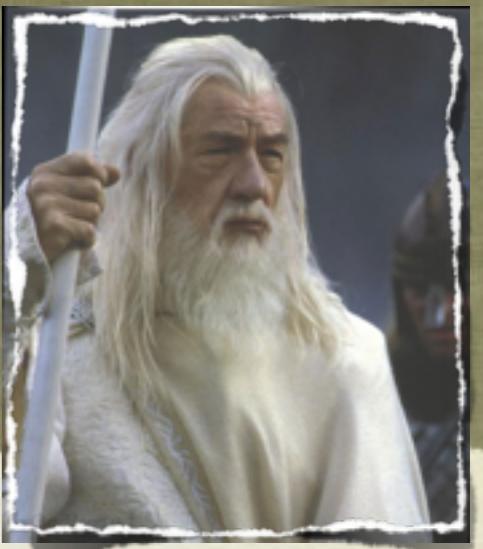
```
def index
  render :inline => "The test parameter: <%= params[:test] %>"
end
```

Open: <http://127.0.01:3000/?test=alsdjasdkl>

Parameters can be received in several forms:

- As get arguments `http://bla/index?a=b`
- As post data `<input ...>`
- As part of a URL (see `routes.rb`)

SESSIONS & FLASH



Sessions are magical beings:

```
class AuthorController < ApplicationController
  def index
    session[:counter] = 0 if session[:counter].nil?
    session[:counter] += 1

    render :inline => "session[:counter]: <%= session
    [:counter] %><br />"
  end
end
```

Flash - It's there, and then it's not:

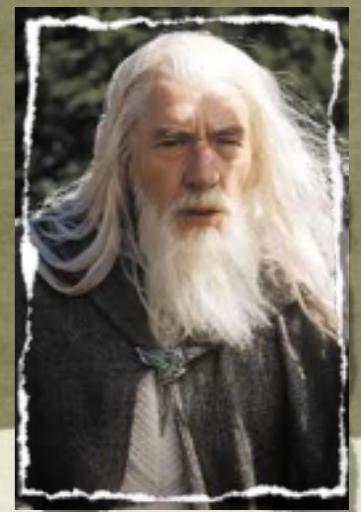
```
class AuthorController < ApplicationController
  def index
    str = ""
    flash.each do |name, msg|
      str += "[FLASH] Name: #{name} Msg: #{msg}<br/>"
    end

    if params[:flashy]
      flash[:alert] = params[:flashy]
    end

    render :inline => str
  end
end
```



MODELS [ENTER]



```
$ rails g model author name:string email:string
...
$ rake db:migrate
...
$ rails c

ruby-1.9.2-p0 > Author.create(:name => 'John
Doe', :email => 'john.doe@authors.com')

=> #<Author id: 1, name: "John Doe", email:
"john.doe@authors.com", created_at: "2011-06-09
10:07:38", updated_at: "2011-06-09 10:07:38">

ruby-1.9.2-p0 > Author.create(:name => 'Jane
Doe', :email => 'jane.doe@authors.com')

=> #<Author id: 2, name: "Jane Doe", email:
"jane.doe@authors.com", created_at: "2011-06-09
10:07:50", updated_at: "2011-06-09 10:07:50">
ruby-1.9.2-p0 >
```

```
class AuthorController < ApplicationController
  def index
    @authors = Author.all
    @authors.each do |author|
      str += "<b>[AUTHOR]</b> <b>name</b>: #{author.name} <b>email</b>: #{author.email} <br />"
    end

    render :inline => str
  end

  def show
    @author = Author.find(params[:id])
    str = "This is the profile of an author:<br /><br />"
    str += "<b>ID</b>: #{@author._id}<br />"
    str += "<b>Name</b>: #{@author.name}<br />"
    str += "<b>Email</b>: #{@author.email}<br />"

    render :inline => str
  end
end
```

Open: <http://127.0.01:3000/authors/1>

CUSTOM ACTIONS

MORE ABOUT ROUTES

FORMATS

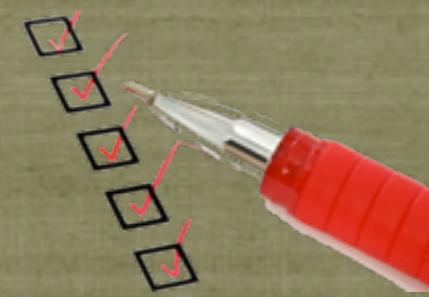
BEFORE AFTER AROUND

VERIFICATION

CSRF

SEND STUFF

RECAP



CLAP, YOU MADE IT



PLACE
PALM
HERE



I'm kidding...

<http://enishi.herokuapp.com>

Q&A *und el feedbaco*





This picture, shamelessly taken from the template, again...

NEXT TIME

Voyeurism