

**Thomas Fuchs**

# Ruby on Rails







*script.aculo.us*

**<http://script.aculo.us/thomas>**

**ABC**

**Digg**

**Nasa**

**Amazon**

**ESPN**

**NBC**

**Apple**

**Gucci**

**Tivo**

**CNET**

**last.fm**

**Twitter**

prototype

prototypejs.org



*script.aculo.us*

script.aculo.us



# **Part 1**

## **Rails Overview and Demo**



# **Part 2**

## **Introduction to Ruby**



# **Part 3**

## **More Rails!**



[rubyonrails.org](http://rubyonrails.org)





# Ruby

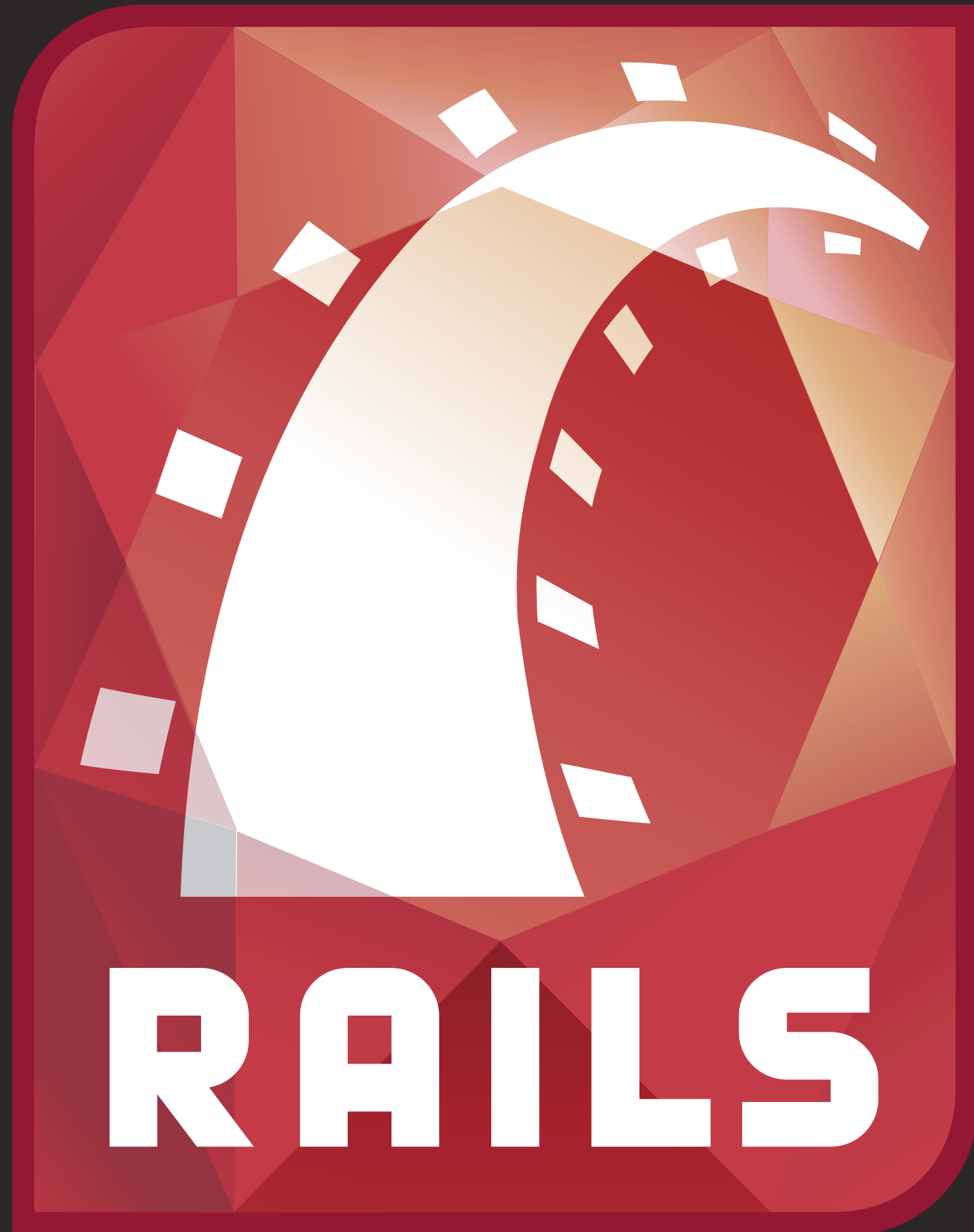
<http://www.ruby-lang.org/>



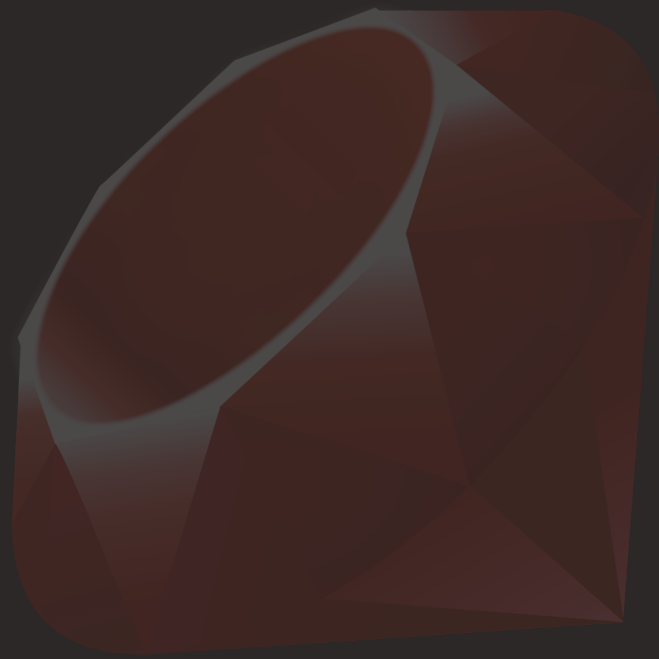
**Programming  
Language**



**Web  
Framework**



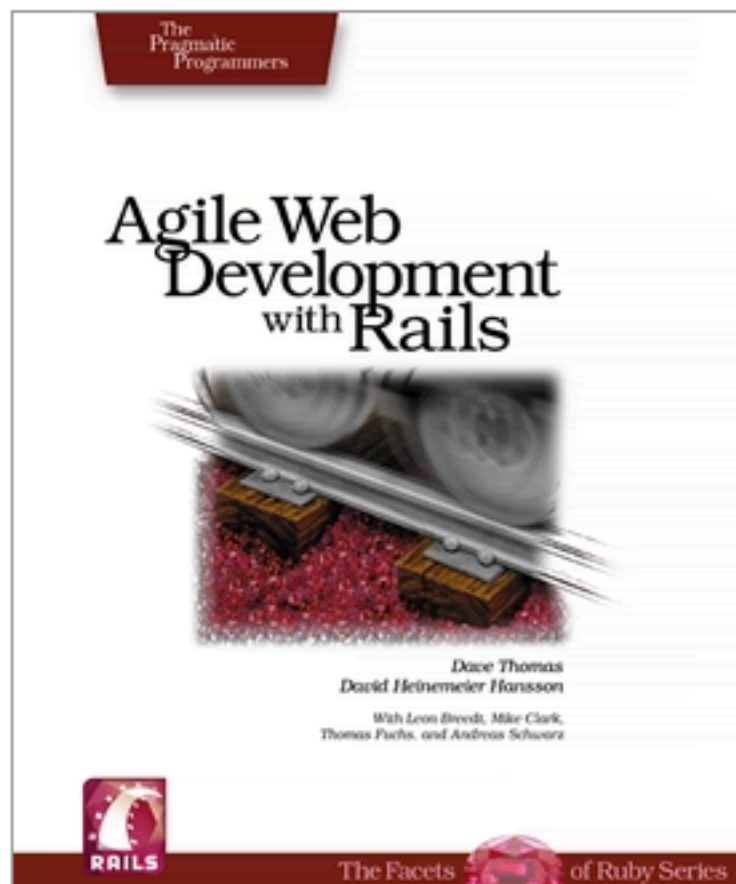




**Programming  
Language**



**Web  
Framework**



**Want to learn more about Ruby on Rails?  
Buy the defining book as a PDF or on paper**

**<http://www.pragmaticprogrammer.com/titles/rails/>**

**Written by Dave Thomas and  
David Heinemeier Hansson**

**Check out these real-life applications build using Rails by 37signals**



**Basecamp project management**

Everyone's favorite web-based project management tool. Easy, fast, elegant. Free trial.

[www.basecamphq.com](http://www.basecamphq.com)



**Backpack information manager**

Gather your ideas, to-dos, notes, photos & files online. Set email and mobile reminders. Try free!

[www.backpackit.com](http://www.backpackit.com)



**Ta-da List, to-do list manager**

The web's simplest, fastest, and most useful to-do list manager. Get things done. Free, 10 second signup.

[www.tadalist.com](http://www.tadalist.com)

**Ruby on Rails is an open-source web framework that's optimized for programmer happiness and sustainable productivity.**

**It lets you write beautiful code by favoring convention over configuration.**



Ruby on Rails is an **open-source** web framework that's optimized for programmer happiness and sustainable productivity.

It lets you write beautiful code by favoring convention over configuration.

Ruby on Rails is an open-source web framework that's **optimized for programmer happiness and sustainable productivity.**

It lets you write beautiful code by favoring convention over configuration.

Ruby on Rails is an open-source web framework that's optimized for programmer happiness and **sustainable productivity.**

It lets you write beautiful code by favoring convention over configuration.



Ruby on Rails is an open-source web framework that's optimized for programmer happiness and sustainable productivity.

It lets you write **beautiful code** by favoring convention over configuration.

Ruby on Rails is an open-source web framework that's optimized for programmer happiness and sustainable productivity.

It lets you write beautiful code by favoring **convention over configuration.**

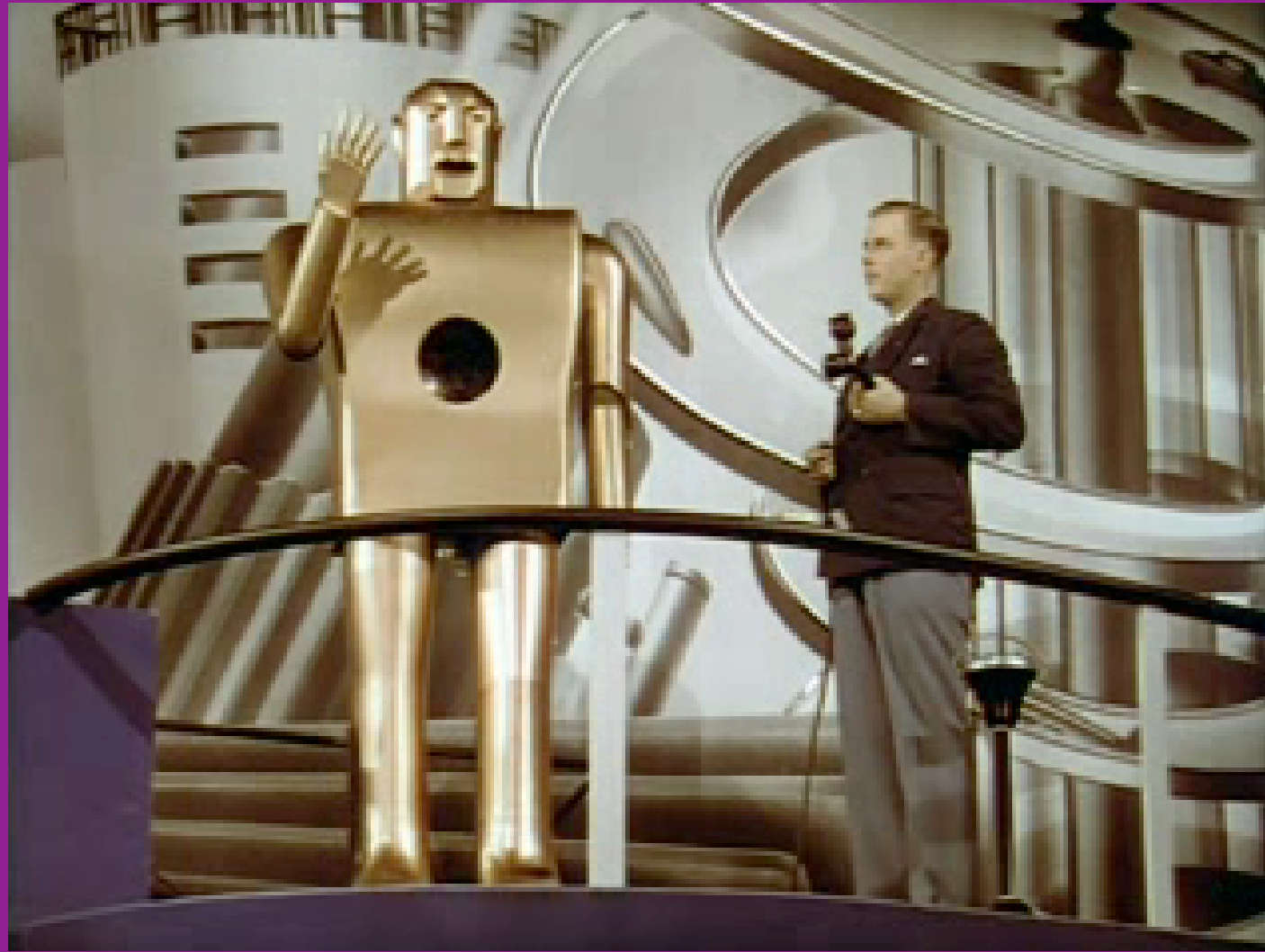
80/20



# Opinionated Software

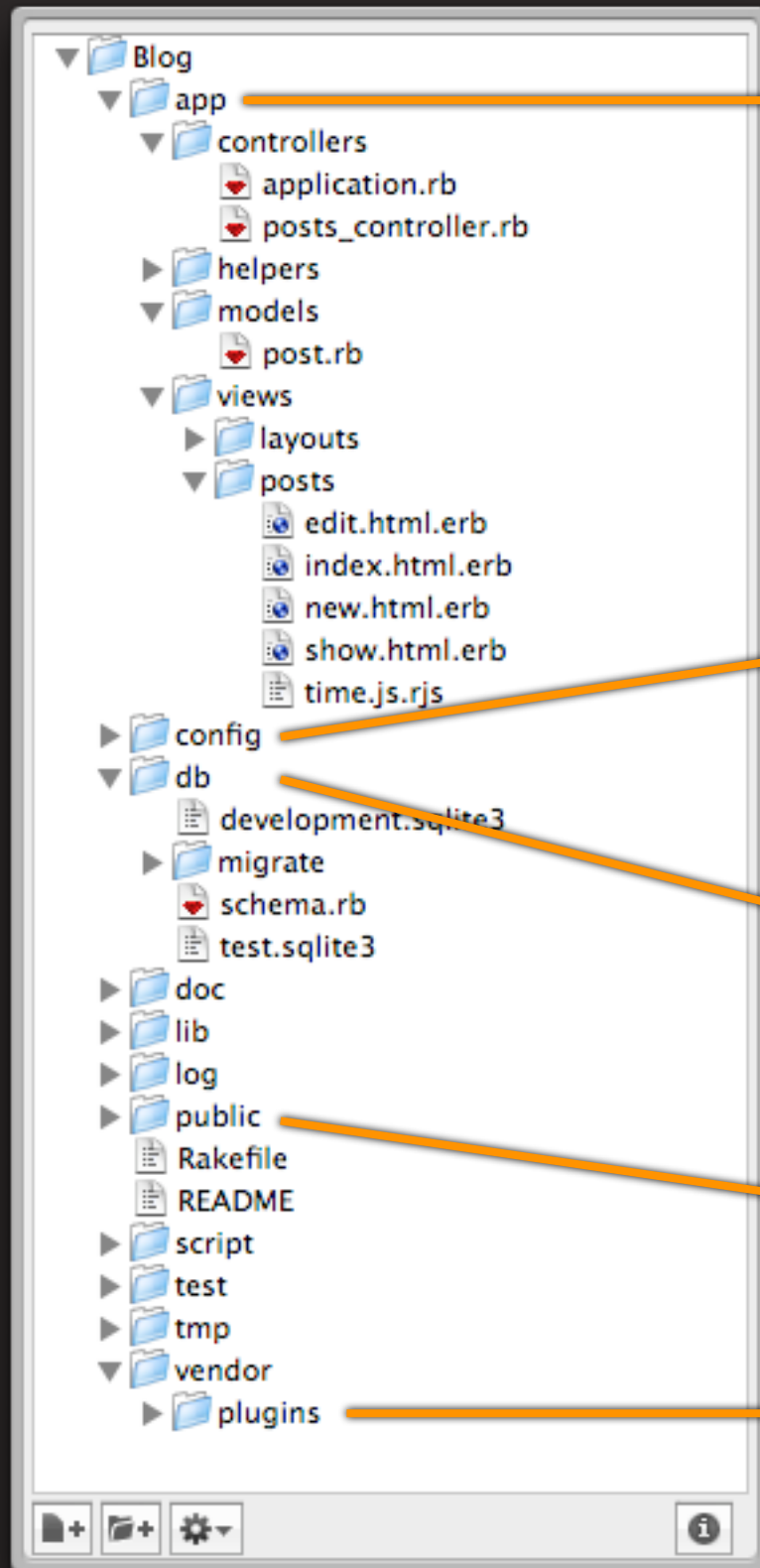
**Don't  
repeat  
yourself**





# ***Demo***

# **Building a Rails App**



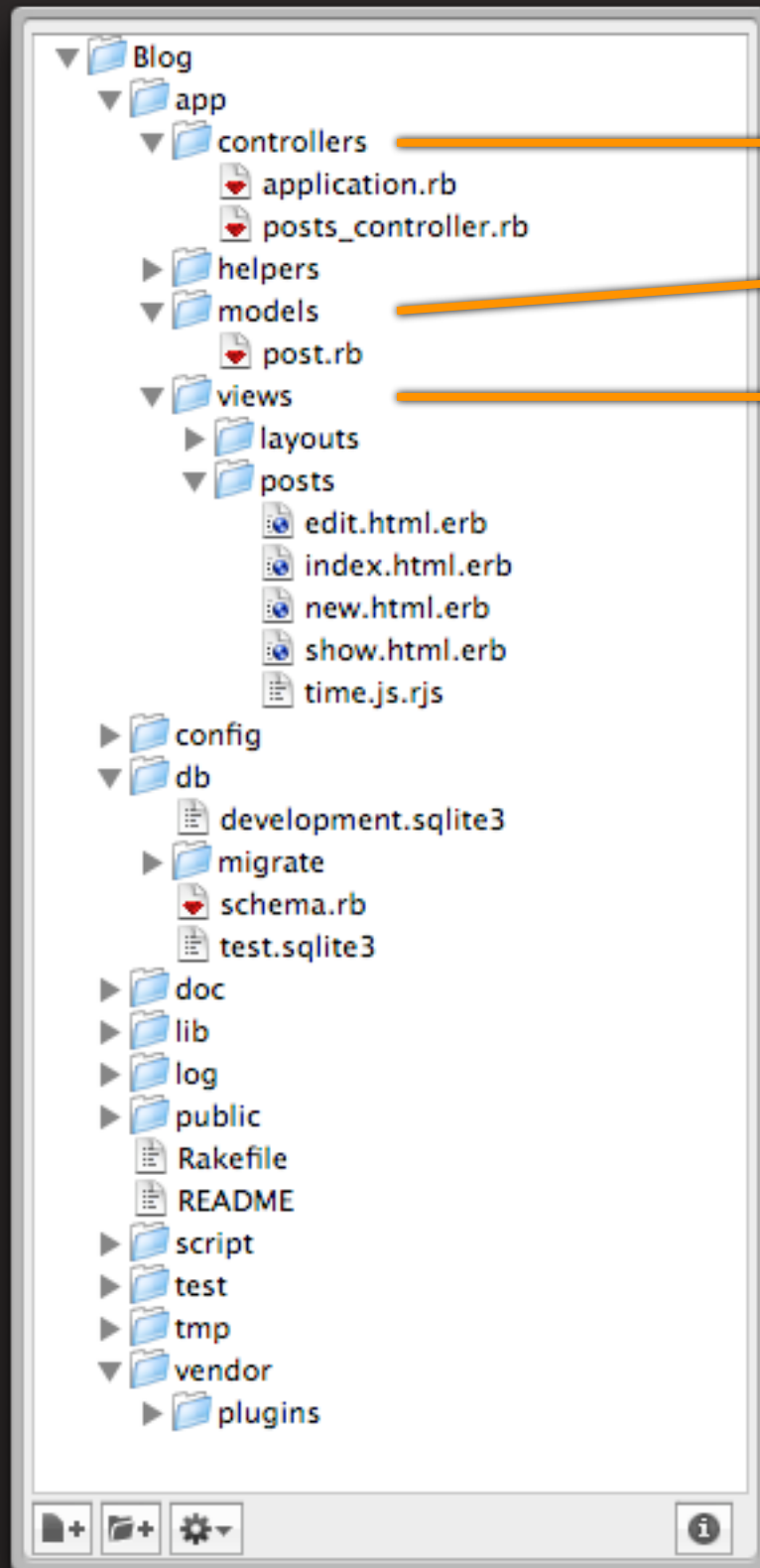
**"app"**  
application logic

**"config"**  
database and deployment information

**"db"**  
database structure and migrations

**"public"**  
CSS, images, JavaScripts and static HTML

**"vendor/plugins"**  
add-ons to "core" rails



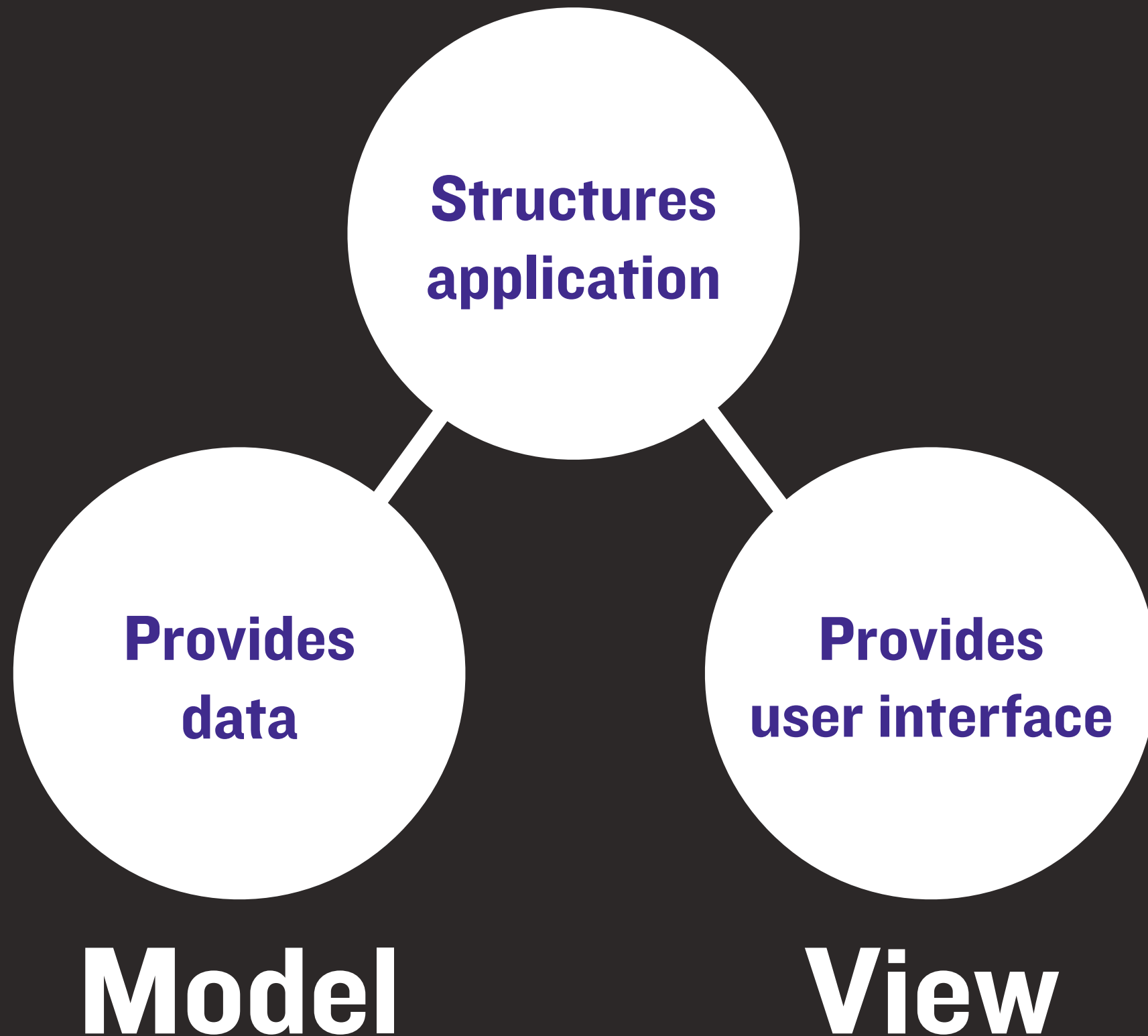
**Controllers**

**Models**

**Views**

# Model View Controller (MVC)

# Controller



## controller/posts\_controller.rb

```
def index
  @posts = Post.find(:all)

  respond_to do |format|
    format.html
    format.xml { render :xml => @posts }
  end
end
```



**"index" maps to the name of the controller by default, so <http://localhost:3000/posts> accesses this action**

```
def index
  @posts = Post.find(:all)

  respond_to do |format|
    format.html
    format.xml { render :xml => @posts }
  end
end
```

**find all posts and assign them to a variable named @posts**

```
def index
  @posts = Post.find(:all)

  respond_to do |format|
    format.html
    format.xml { render :xml => @posts }
  end
end
```

**"Post" is the name of the model**

```
def index
  @posts = Post.find(:all)

  respond_to do |format|
    format.html
    format.xml { render :xml => @posts }
  end
end
```

**we decide to respond to different requests, so we can serve a web site for users, but also XML data for other systems (API)**

```
def index
  @posts = Post.find(:all)

  respond_to do |format|
    format.html
    format.xml { render :xml => @posts }
  end
end
```

**for HTML, just use the defaults:  
get the view that is named the same as  
the action, and render it**

```
def index
  @posts = Post.find(:all)

  respond_to do |format|
    format.html
    format.xml { render :xml => @posts }
  end
end
```

**for XML requests, use the default  
XML rendering, no more code required**

# views/posts/index.html.erb

```
<h1>Listing posts</h1>

<table>
  <tr>
    <th>Title</th>
    <th>Body</th>
    <th>Published</th>
  </tr>

  <% for post in @posts %>
    <tr>
      <td><%=h post.title %></td>
      <td><%=h post.body %></td>
      <td><%=h post.published %></td>
      <td><%= link_to 'Show', post %></td>
      <td><%= link_to 'Edit', edit_post_path(post) %></td>
      <td><%= link_to 'Destroy', post, :confirm => 'Are you sure?', :method => :delete %></td>
    </tr>
  <% end %>
</table>

<br />

<%= link_to 'New post', new_post_path %>
```



# views/posts/index.html.erb

```
<h1>Listing posts</h1>

<table>
  <tr>
    <th>Title</th>
    <th>Body</th>
    <th>Published</th>
  </tr>

  <% for post in @posts %>
    <tr>
      <td><%=h post.title %></td>
      <td><%=h post.body %></td>
      <td><%=h post.published %></td>
      <td><%= link_to 'Show', post %></td>
      <td><%= link_to 'Edit', edit_post_path(post) %></td>
      <td><%= link_to 'Destroy', post, :confirm => 'Are you sure?', :method => :delete %></td>
    </tr>
  <% end %>
</table>

<br />

<%= link_to 'New post', new_post_path %>
```

**embedded Ruby (.erb)**

**automatically provided link helpers**





```
<?xml version="1.0" encoding="UTF-8"?>
<posts type="array">
  <post>
    <body>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
    eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad
    minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex
    ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate
    velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
    cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id
    est laborum.</body>
    <created-at type="datetime">2008-04-14T12:07:46+02:00</created-at>
    <id type="integer">1</id>
    <published type="boolean">true</published>
    <title>Hallo FH!</title>
    <updated-at type="datetime">2008-04-14T12:07:46+02:00</updated-at>
  </post>
</posts>
```

## models/post.rb

```
class Post < ActiveRecord::Base
  validates_presence_of :title
  validates_length_of   :body, :minimum=>10
end
```

## self-explanatory validations



```
class Post < ActiveRecord::Base
  validates_presence_of :title
  validates_length_of   :body, :minimum=>10
end
```

Posts: update

◀ ▶ ↺ ✂ +

http://localhost:3000/posts/1

Google

## Editing post

1 error prohibited this post from being saved

There were problems with the following fields:

- Title can't be blank

**Title**

**Body**

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**Published**

☒

Update

[Show](#) | [Back](#)

```
class Post < ActiveRecord::Base
  validates_presence_of :title
  validates_length_of   :body, :minimum=>10
end
```

# What is missing here?



**Because fields are  
already declared in the  
Database, no need to  
redeclare it in the model**

## db/schema.rb (automatically created)

```
ActiveRecord::Schema.define(:version => 1) do

  create_table "posts", :force => true do |t|
    t.string "title"
    t.text "body"
    t.boolean "published"
    t.datetime "created_at"
    t.datetime "updated_at"
  end

end
```

## db/migrate/001\_create\_posts.rb (automatically created by generator)

```
class CreatePosts < ActiveRecord::Migration
  def self.up
    create_table :posts do |t|
      t.string :title
      t.text :body
      t.boolean :published

      t.timestamps
    end
  end

  def self.down
    drop_table :posts
  end
end
```



# Ruby

<http://www.ruby-lang.org/>



**Programming  
Language**



**Web  
Framework**

**Why an other  
programming  
language?**



# *Tan Lines From Typical Summer Activities*

Waterskiing



Mountain Biking



SCUBA Diving



Rollerblading



Computer Programming



Tennis



Source: <http://www.kerrolisaa.com/i/1/5637.jpg>

**Some might like this, but it's  
mostly because most  
Software Development Tools  
are designed for computers,  
not for human beings.**

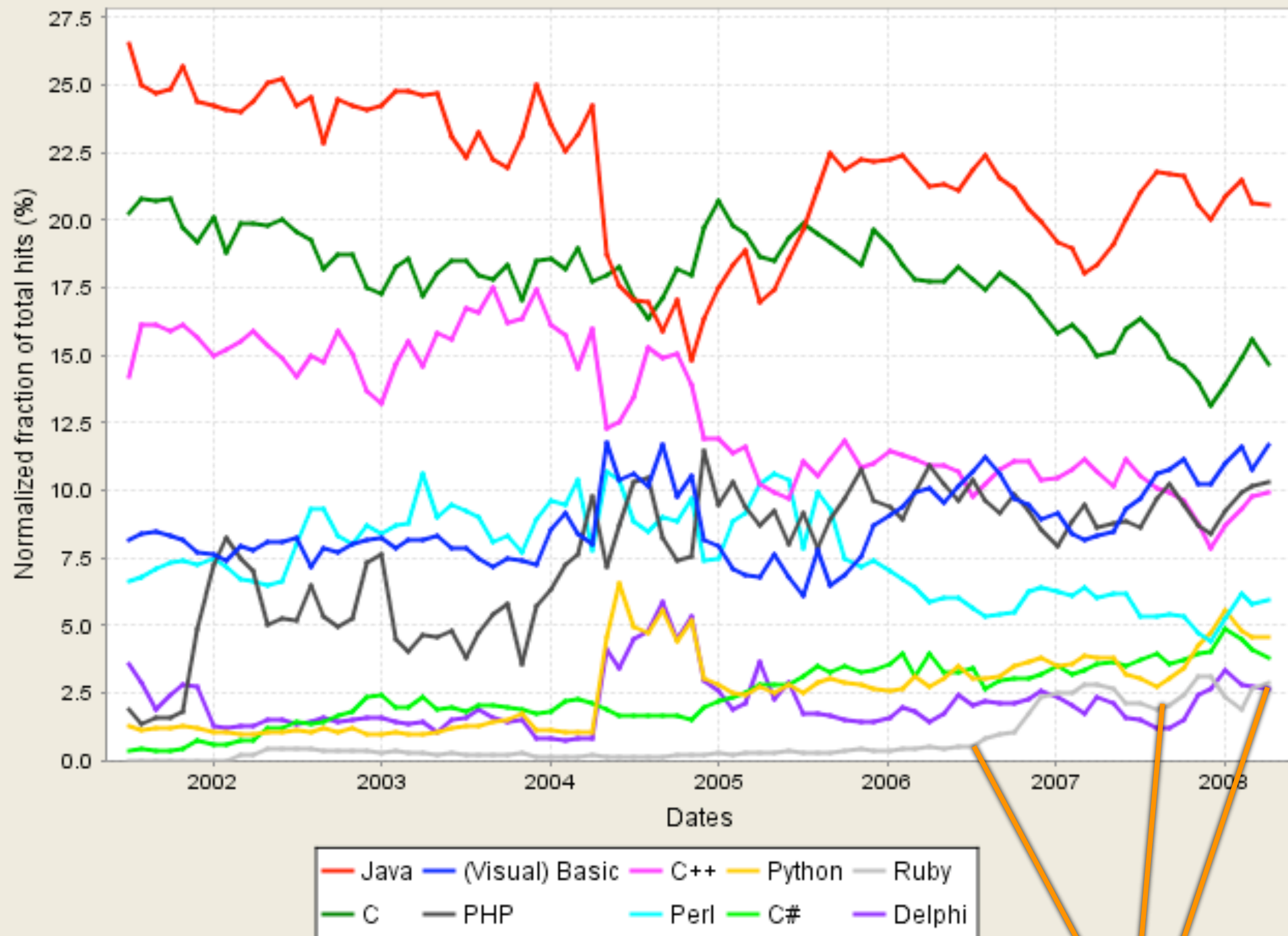


**“Ruby is simple in appearance, but is very complex inside, just like our human body.”**

**Yukihiro “Matz” Matsumoto,  
Creator of Ruby**



## Tiobe Programming Community Index



growing steadily

**Make the  
computer work  
for you**

**Design the  
language and  
libraries for  
people first**

**Let others care  
about performance**



# **A Programmer's Best Friend**

```
say = "I love Ruby"  
puts say
```

```
=> "I love Ruby"
```

no type  
declarations  
"String..."

no "main" method

```
say = "I love Ruby" — no "end of  
puts say           instruction"  
                    marker
```

no superfluous  
keywords ("var")



```
say = "I love Ruby"  
say['love'] = "*love*"  
puts say.upcase
```

```
=> "I *LOVE* RUBY"
```

```
5.times { puts say }
```

```
=> "I *love* Ruby"
```

```
=> "I *love* Ruby"
```

```
=> "I *love* Ruby"
```

```
=> "I *love* Ruby"
```

```
=> "I *love* Ruby"
```

**[Ruby is] a dynamic, open source programming language with a focus on simplicity and productivity.**

**It has an elegant syntax that is natural to read and easy to write.**

[Ruby is] a **dynamic**, open  
source programming language  
with a focus on simplicity and  
productivity.

It has an elegant syntax that is  
natural to read and easy to write.

[Ruby is] a dynamic, **open source** programming language with a focus on simplicity and productivity.

It has an elegant syntax that is natural to read and easy to write.

[Ruby is] a dynamic, open source programming language with a focus on **simplicity** and productivity.

It has an elegant syntax that is natural to read and easy to write.

[Ruby is] a dynamic, open  
source programming language  
with a focus on simplicity and  
**productivity.**

It has an elegant syntax that is  
natural to read and easy to write.

[Ruby is] a dynamic, open source programming language with a focus on simplicity and productivity.

It has an **elegant syntax** that is natural to read and easy to write.



The image shows two smartphones side-by-side. On the left is an iPhone with a green leaf wallpaper, displaying the time 1:24 and the date Saturday, June 30. It has a 'slide to unlock' bar at the bottom. On the right is a BlackBerry smartphone with a gold-colored bezel, displaying a grid of various application icons like email, calendar, and browser. A large, semi-transparent purple rectangle with the text 'Less is more' in white bold font is centered over both devices.

**Less is more**

# Java

```
public static String reverseString(String source) {  
    int i, len = source.length();  
    StringBuffer dest = new StringBuffer(len);  
  
    for (i = (len - 1); i >= 0; i--)  
        dest.append(source.charAt(i));  
  
    return dest.toString();  
}  
  
reverseString("Krawutzikapuzi");  
=> "izupakiztuwarK"
```

# PHP

```
strrev("Krawutzikapuzi");  
=> "izupakiztuwarK"
```

# Ruby

```
"Krawutzikapuzi".reverse  
=> "izupakiztuwarK"
```

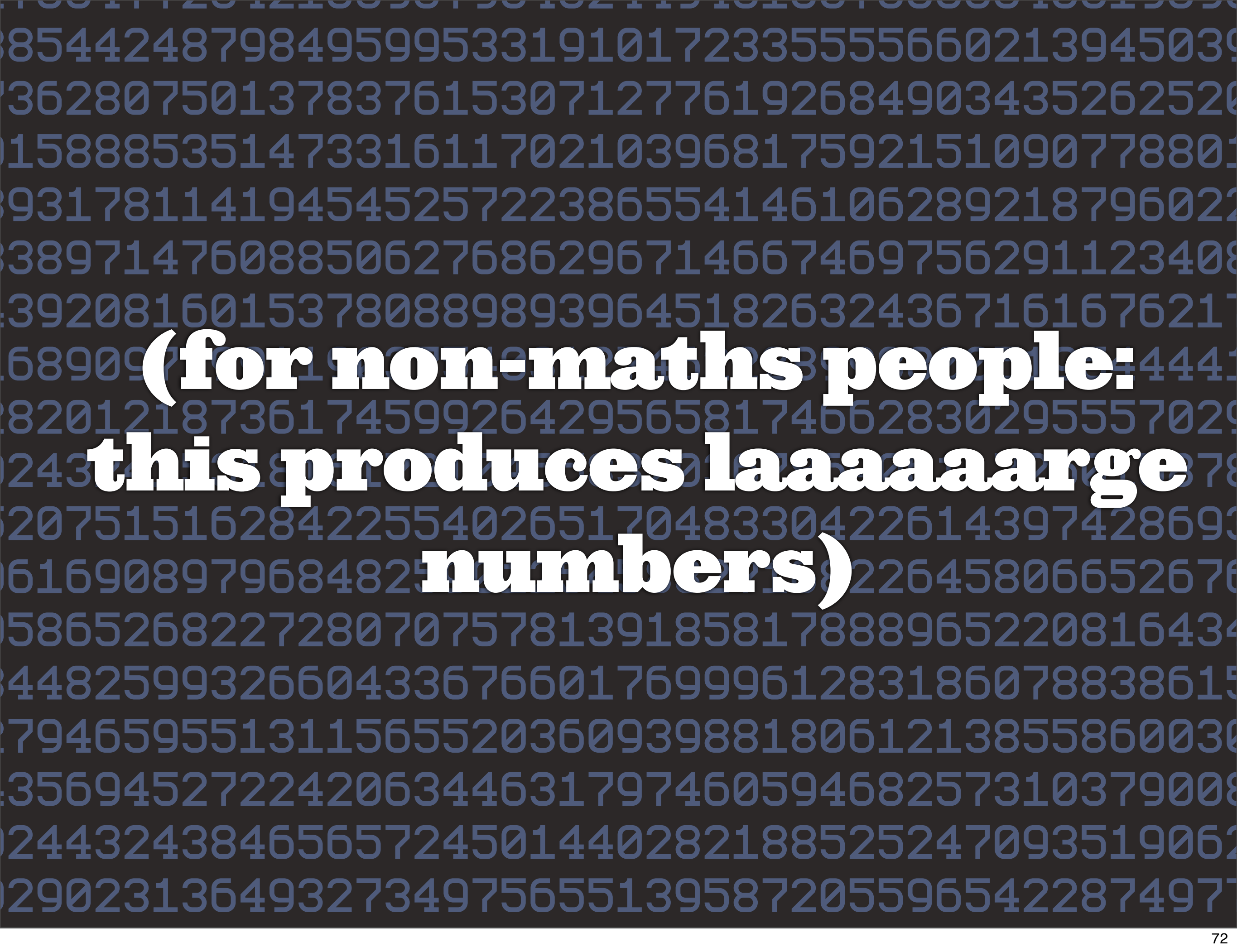
# Ruby

```
"Krawutzikapuzi".split(//).inject(""){|m,c|c+m}  
=> "izupakiztuwarK"
```

**Not using the built-in  
“reverse” method: still  
much smaller than Java**

**Do the  
“right thing”**

```
def fact(n)  
    return 1 if n == 0  
    n * fact(n-1)  
end
```



**(for non-maths people:  
this produces laaaaaarge  
numbers)**



```
>> fact(2)  
=> 2
```

```
>> fact(3)  
=> 6
```

```
>> fact(10)  
=> 3628800
```

```
<?php
function fact($int){
    if ($int<2) return 1;
    for ($f=2; $int-1>1; $f*=$int--);
    return $f;
};
```

```
echo fact(10)."\n";
echo fact(999)."\n";
?>
```

The PHP logo, consisting of the letters "PHP" in white, bold, sans-serif font, centered within a solid blue square.

```
$ php fact.php
```

```
3628800
```

```
INF
```

———— **PHP overflows and cannot  
calculate a result**

# Ruby does not care about overflows and does the right thing: calculate what you want

```
>> fact(999)  
=>
```

```
402387260077093773543702433923003985719374864210714632543799910429938512398629020592044208  
486969404800479988610197196058631666872994808558901323829669944590997424504087073759918823  
627727188732519779505950995276120874975462497043601418278094646496291056393887437886487337  
119181045825783647849977012476632889835955735432513185323958463075557409114262417474349347  
553428646576611667797396668820291207379143853719588249808126867838374559731746136085379534  
524221586593201928090878297308431392844403281231558611036976801357304216168747609675871348  
312025478589320767169132448426236131412508780208000261683151027341827977704784635868170164  
365024153691398281264810213092761244896359928705114964975419909342221566832572080821333186  
116811553615836546984046708975602900950537616475847728421889679646244945160765353408198901  
385442487984959953319101723355556602139450399736280750137837615307127761926849034352625200  
015888535147331611702103968175921510907788019393178114194545257223865541461062892187960223  
838971476088506276862967146674697562911234082439208160153780889893964518263243671616762179  
168909779911903754031274622289988005195444414282012187361745992642956581746628302955570299  
024324153181617210465832036786906117260158783520751516284225540265170483304226143974286933  
061690897968482590125458327168226458066526769958652682272807075781391858178889652208164348  
344825993266043367660176999612831860788386150279465955131156552036093988180612138558600301  
435694527224206344631797460594682573103790084024432438465657245014402821885252470935190620  
929023136493273497565513958720559654228749774011413346962715422845862377387538230483865688  
976461927383814900140767310446640259899490222221765904339901886018566526485061799702356193  
897017860040811889729918311021171229845901641921068884387121855646124960798722908519296819  
372388642614839657382291123125024186649353143970137428531926649875337218940694281434118520  
158014123344828015051399694290153483077644569099073152433278288269864602789864321139083506  
217095002597389863554277196742822248757586765752344220207573630569498825087968928162753848  
863396909959826280956121450994871701244516461260379029309120889086942028510640182154399457  
156805941872748998094254742173582401063677404595741785160829230135358081840096996372524230  
560855903700624271243416909004153690105933983835777939410970027753472000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
```

# Complete Object Orientation

10.class  
=> Fixnum

Fixnum.object\_id

=> 108610

`(10*10).class`

`=> Fixnum`

`(10*10*100_000_000).class`

`=> Bignum`





**Taking the pain  
out of things you  
don't like doing**



```
attwenger = [  
  { :year => 1991, :tracks => 18, :title => "Most" },  
  { :year => 1992, :tracks => 10, :title => "Pflug" },  
  { :year => 1993, :tracks => 17, :title => "Luft" },  
  { :year => 1997, :tracks => 5, :title => "Song" },  
  { :year => 2002, :tracks => 15, :title => "Sun" },  
  { :year => 2005, :tracks => 14, :title => "dog" },  
  { :year => 2006, :tracks => 17, :title => "dog2 remixes" },  
  { :year => 2007, :tracks => 8, :title => "die Kia" },  
]
```

```
# calculate the total amount of tracks in all albums
total = attwenger.inject(0) { |memo, album| memo + album[:tracks] }
puts "There are #{total} Attwenger tracks on #{attwenger.size} albums."
```

```
# average number of tracks  
puts "The average number of tracks per album is #{total/attwenger.size}."
```

```
# which letters are used in album titles?
title_strings = attwenger.map{ |album| album[:title] }.join.gsub(/\s/, ' ').upcase
title_letters = title_strings.split('').uniq.sort.join

puts "These letters are used in album titles: #{title_letters}"
```

# Java

```
public static String reverseString(String source) {  
    int i, len = source.length();  
    StringBuffer dest = new StringBuffer(len);  
  
    for (i = (len - 1); i >= 0; i--)  
        dest.append(source.charAt(i));  
  
    return dest.toString();  
}  
  
reverseString("Krawutzikapuzi");  
=> "izupakiztuwarK"
```

```
# most common letters in album titles
title_count = title_letters.split('').sort_by{|letter| title_strings.count(letter) }.
  reverse.map{ |letter| "#{letter}: #{title_strings.count(letter)}" }

puts "Number of times individual letters are used in album titles, highest first:"
puts "#{title_count.join(', ')}"
```

## just 10 lines of code

```
$ ruby arrays.rb
```

```
There are 104 Attwenger tracks on 8 albums.
```

```
The average number of tracks per album is 13.
```

```
These letters are used in album titles:
```

```
2ADEFGLIKLMNOPRSTUX
```

```
Number of times individual letters are used in  
album titles, highest first:
```

```
O: 4, S: 4, G: 4, U: 3, D: 3, I: 3, E: 3, N: 2,
```

```
F: 2, L: 2, T: 2, M: 2, X: 1, K: 1, P: 1, R: 1,
```

```
A: 1, 2: 1
```

# Dynamic



```
class Fixnum
  def fact
    return 1 if self == 0
    self * (self-1).fact
  end
end
```

```
10.fact  
=> 3628800
```

```
class String
  def count_letters
    cleaned = self.gsub(/\s/, '').upcase
    letters = cleaned.split(//).uniq.sort.join
    letters.split(//).sort_by{|letter| cleaned.count(letter) }.
      reverse.map{ |letter| "#{letter}: #{cleaned.count(letter)}" }
  end
end
```

```
>> puts "Lorem ipsum dolor sit amet, consectetur adipisicing
elit.".count_letters.join(', ')
=> "I: 7, T: 5, E: 5, S: 4, O: 4, C: 3, R: 3, M: 3, L: 3, U: 2, P:
2, N: 2, D: 2, A: 2, G: 1, .: 1, ,: 1"
```

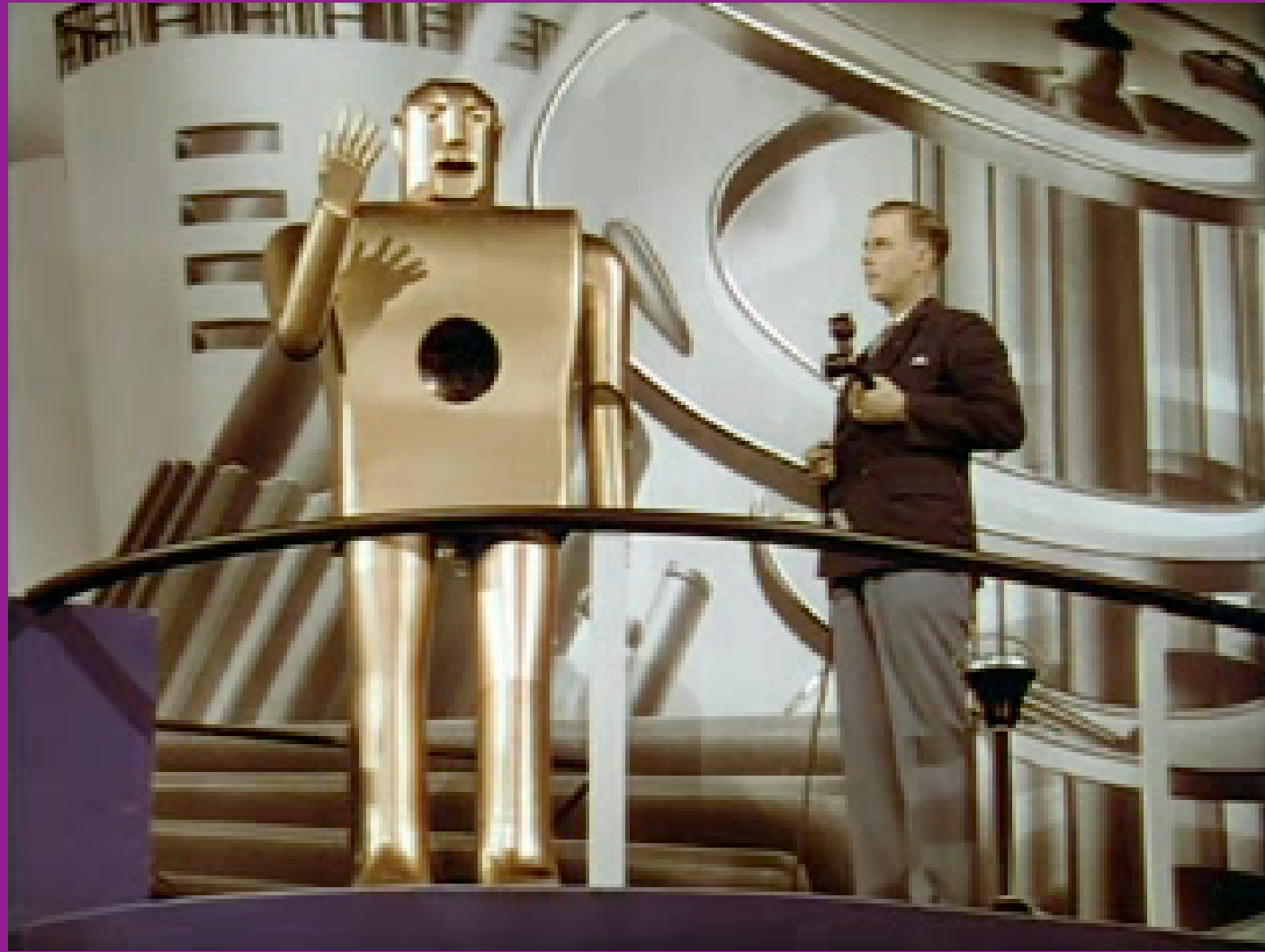
```
class String
  def method_missing(method)
    puts "intercepting call to #{method}"
  end
end
```

```
"Lorem ipsum dolor".krawutzikaputzi  
=> "intercepting call to krawutzikaputzi"
```

**method\_missing is used to  
provide on-the-fly "finders"**



```
>> Post.find_by_published(true)
=> #<Post id: 1, title: "Hallo FH!",
body: "Lorem ipsum dolor sit amet,
consectetur adipiscing...", published:
true, created_at: "2008-04-14 12:07:46",
updated_at: "2008-04-14 12:07:46">
```



# *Demo*

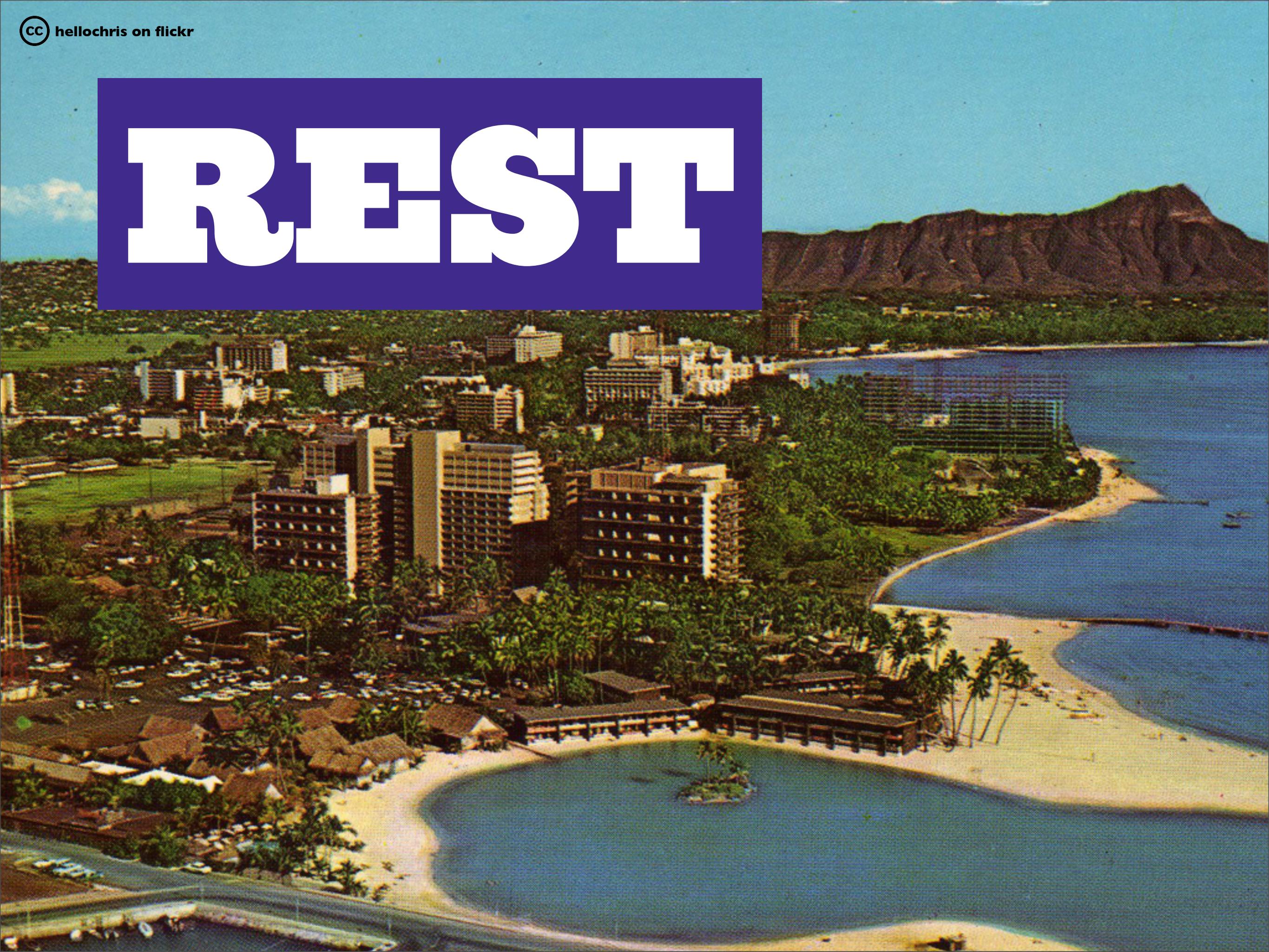
## **Toying with Ruby**



# More...



# REST







**In the beginning,  
there was the URI.**



<http://script.aculo.us/thomas/index.html>

`http://script.aculo.us/thomas/index.html`

**this is an actual file,  
in the thomas directory**

<http://www.merriam-webster.com/cgi-bin/mwwod.pl>

`http://www.merriam-webster.com/cgi-bin/mwwod.pl`

**"cgi-bin" was a special  
location for executable  
files**

<http://gilesbowkett.blogspot.com/search?updated-max=2008-04-08T09%3A25%3A00-07%3A00&max-results=7>

**"search" is probably  
not a filename**

`http://gilesbowkett.blogspot.com/search?updated-  
max=2008-04-08T09%3A25%3A00-07%3A00&max-results=7`

**wtf?**

[http://www.amazon.de/gp/  
product/B0000262LH/  
ref=s9subs\\_c3\\_img1-  
rfc\\_p\\_19\\_32\\_31\\_9\\_9?  
pf\\_rd\\_m=A1IDDPBG1NC5TQ&pf\\_rd\\_  
s=center-2&pf\\_rd\\_r=1FMGVYJN44  
H7WQD9YCR9&pf\\_rd\\_t=101&pf\\_rd\\_  
p=139524591&pf\\_rd\\_i=301128](http://www.amazon.de/gp/product/B0000262LH/ref=s9subs_c3_img1-rfc_p_19_32_31_9_9?pf_rd_m=A1IDDPBG1NC5TQ&pf_rd_s=center-2&pf_rd_r=1FMGVYJN44H7WQD9YCR9&pf_rd_t=101&pf_rd_p=139524591&pf_rd_i=301128)

[http://www.amazon.de/gp/  
product/B0000262LH/  
ref=s9subs\\_c3\\_img1-  
rfc\\_p\\_19\\_32\\_31\\_9\\_9?  
pf\\_rd\\_m=A1IDDPBG1NC5TQ&pf\\_rd\\_  
s=center-2&pf\\_rd\\_r=1FMGVYJN44  
H7WQD9YCR9&pf\\_rd\\_t=101&pf\\_rd\\_  
p=139524591&pf\\_rd\\_i=301128](http://www.amazon.de/gp/product/B0000262LH/ref=s9subs_c3_img1-rfc_p_19_32_31_9_9?pf_rd_m=A1IDDPBG1NC5TQ&pf_rd_s=center-2&pf_rd_r=1FMGVYJN44H7WQD9YCR9&pf_rd_t=101&pf_rd_p=139524591&pf_rd_i=301128)

**yeah, right.**



<http://www.amazon.de/cd/attwenger/dog>

`http://www.amazon.de/cd/attwenger/dog`



**wouldn't this be nicer?**

**URLs were supposed to be about resources—aka objects—not some frankenstein combination of actual file paths and query strings to dictate which thing, and what to do with it**

# REST

**(Representational State Transfer)  
follows this definition and  
identifies resources uniformly**

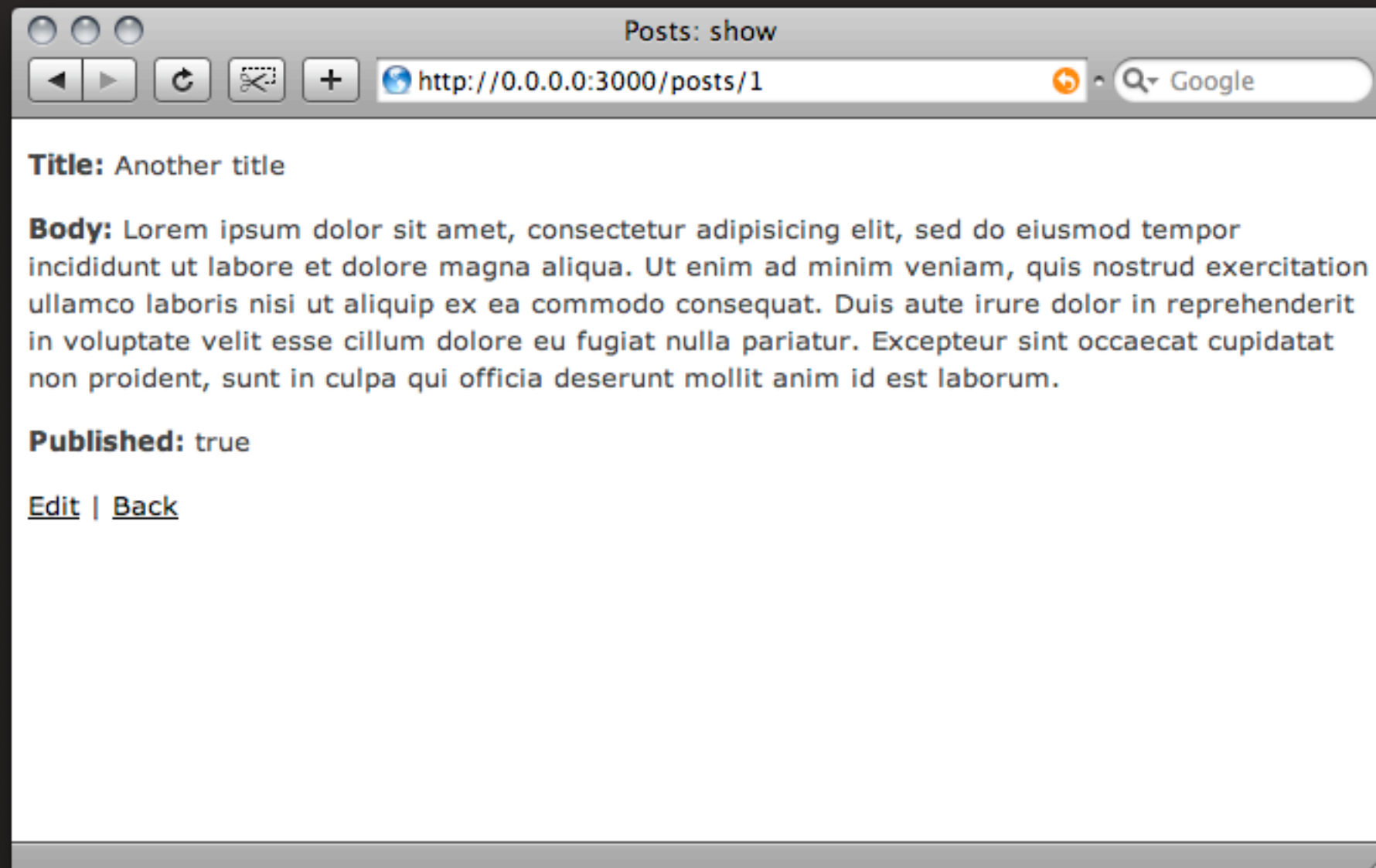
**Web applications  
usually consist  
of objects**

**(like, say, blog posts)**

# **What's a blog post?**

**It's an entry in a database.  
Consists of title, body, date,  
whether it's public or not,  
and other data.**

# HTML representation (read)



<http://0.0.0.0:3000/posts/1>



# XML representation (read)

```
<?xml version="1.0" encoding="UTF-8"?>
<post>
  <body>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad
minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex
ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate
velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id
est laborum.</body>
  <created-at type="datetime">2008-04-14T12:07:46+02:00</created-at>
  <id type="integer">1</id>
  <published type="boolean">true</published>
  <title>Another title</title>
  <updated-at type="datetime">2008-04-15T11:30:50+02:00</updated-at>
  <version type="integer">2</version>
</post>
```

**<http://0.0.0.0:3000/posts/1.xml>**

```
def show
  @post = Post.find(params[:id])

  respond_to do |format|
    format.html
    format.xml { render :xml => @post }
  end
end
```

database entry gets  
converted into...

```
def show
  @post = Post.find(params[:id])

  respond_to do |format|
    format.html
    format.xml { render :xml => @post }
  end
end
```

...REST representations

# **RESTful operations**

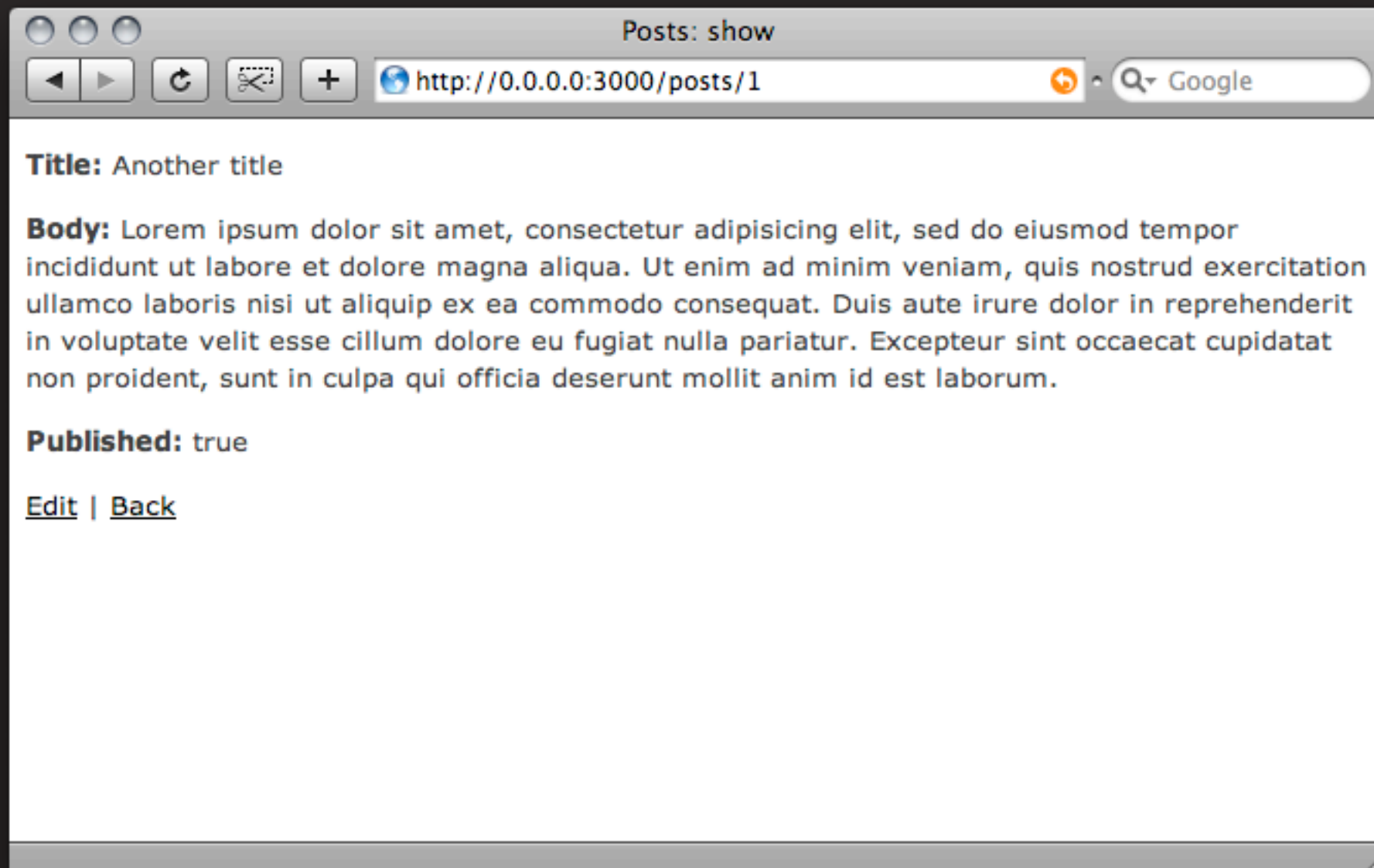
**Create**

**Read**

**Update**

**Delete**

# Read



# HTTP verbs

**In HTTP each request is  
preceded by a request  
type, or "verb".**

**Normal page loads use  
the "GET" verb, while  
web forms use "POST".**



**HTTP knows more verbs,  
most importantly "PUT",  
"DELETE" and "HEAD".**

**REST operation**

**Create**

**Read**

**Update**

**Delete**

**HTTP verb**

**POST**

**GET**

**PUT**

**DELETE**

operation (read)

GET

http://0.0.0.0:3000/posts/1

resource identifier

operation (delete)

DELETE

http://0.0.0.0:3000/posts/1

resource identifier  
(same as for read)

**REST allows for  
consistent,  
guessable URIs**

**The best thing is  
that Rails does all  
this completely  
automatically**



# Plugins





**100s of plugins available**



**Don't  
repeat  
yourself**

acts\_as\_versioned

```
>> post = Post.find(:first)
=> #<Post id: 1, title: "Another title", body: "Lorem ipsum dolor sit amet, consectetur adipisicing...", published: true, created_at: "2008-04-14 12:07:46", updated_at: "2008-04-15 11:30:50", version: 2>
>> post.versions
=> [#<Post::Version id: 1, post_id: 1, version: 1, title: "This is a test post", body: "Lorem ipsum dolor sit amet, consectetur adipisicing...", published: true, created_at: "2008-04-14 12:07:46", updated_at: "2008-04-15 11:29:44">, #<Post::Version id: 2, post_id: 1, version: 2, title: "Another title", body: "Lorem ipsum dolor sit amet, consectetur adipisicing...", published: true, created_at: "2008-04-14 12:07:46", updated_at: "2008-04-15 11:30:50">]
>> post.versions.size
=> 2
```

**title changed  
between versions**


## db/migrate/002\_add\_post\_versions.rb

```
class AddPostVersions < ActiveRecord::Migration
  require_dependency 'post'

  def self.up
    Post.create_versioned_table
  end

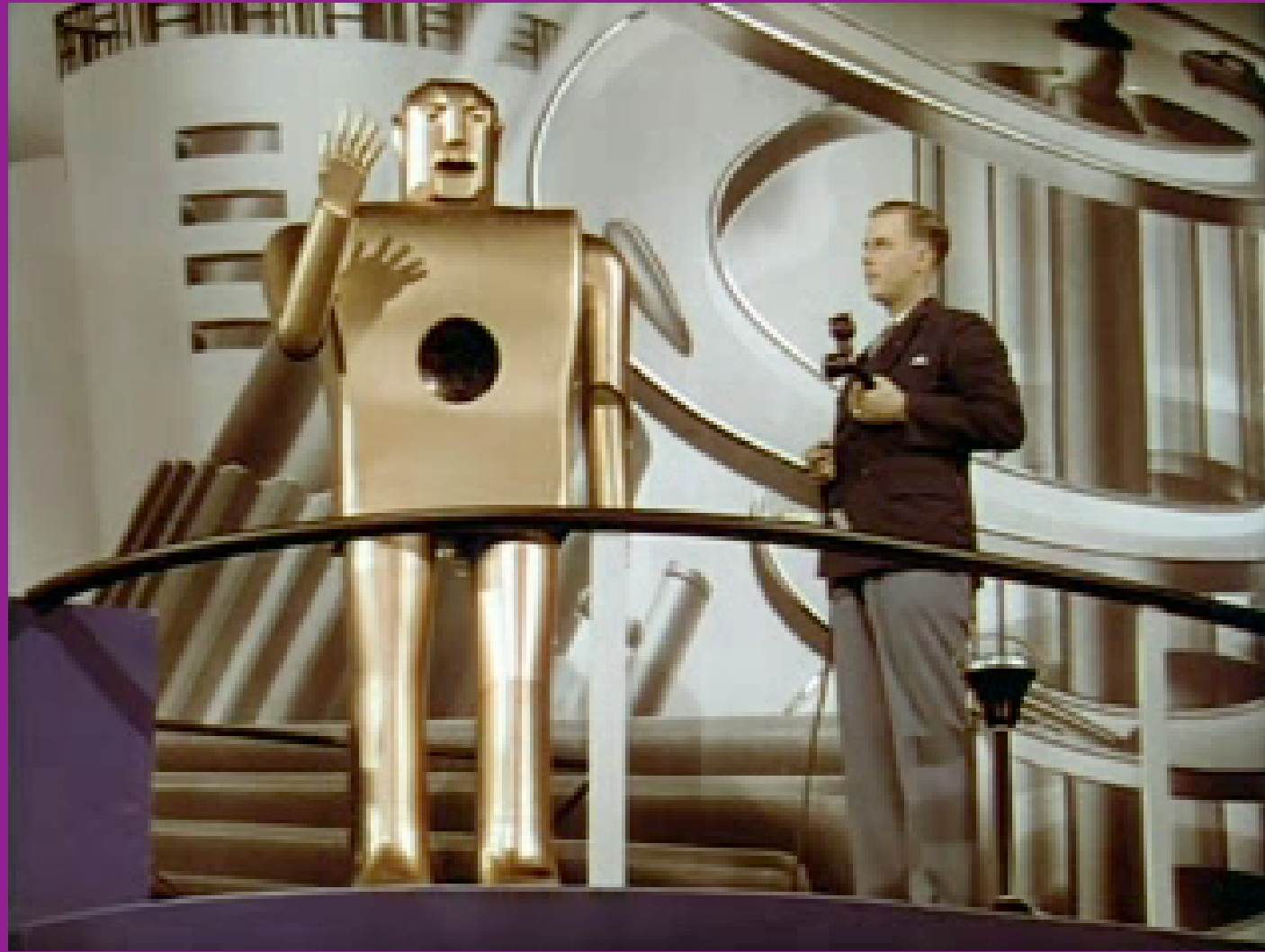
  def self.down
    Post.drop_versioned_table
  end
end
```

one line of code



```
class Post < ActiveRecord::Base
  acts_as_versioned

  validates_presence_of :title
  validates_length_of   :body, :minimum=>10
end
```



# ***Demo*** **Plugins**

# Ajax

**“Ajax” allows to update  
parts of the page,  
without a complete reload**



Rails includes the  and *script.aculo.us* JavaScript frameworks

display current (server) time



Posts: index

http://localhost:3000/posts

Tue Apr 15 12:05:34 +0200 2008

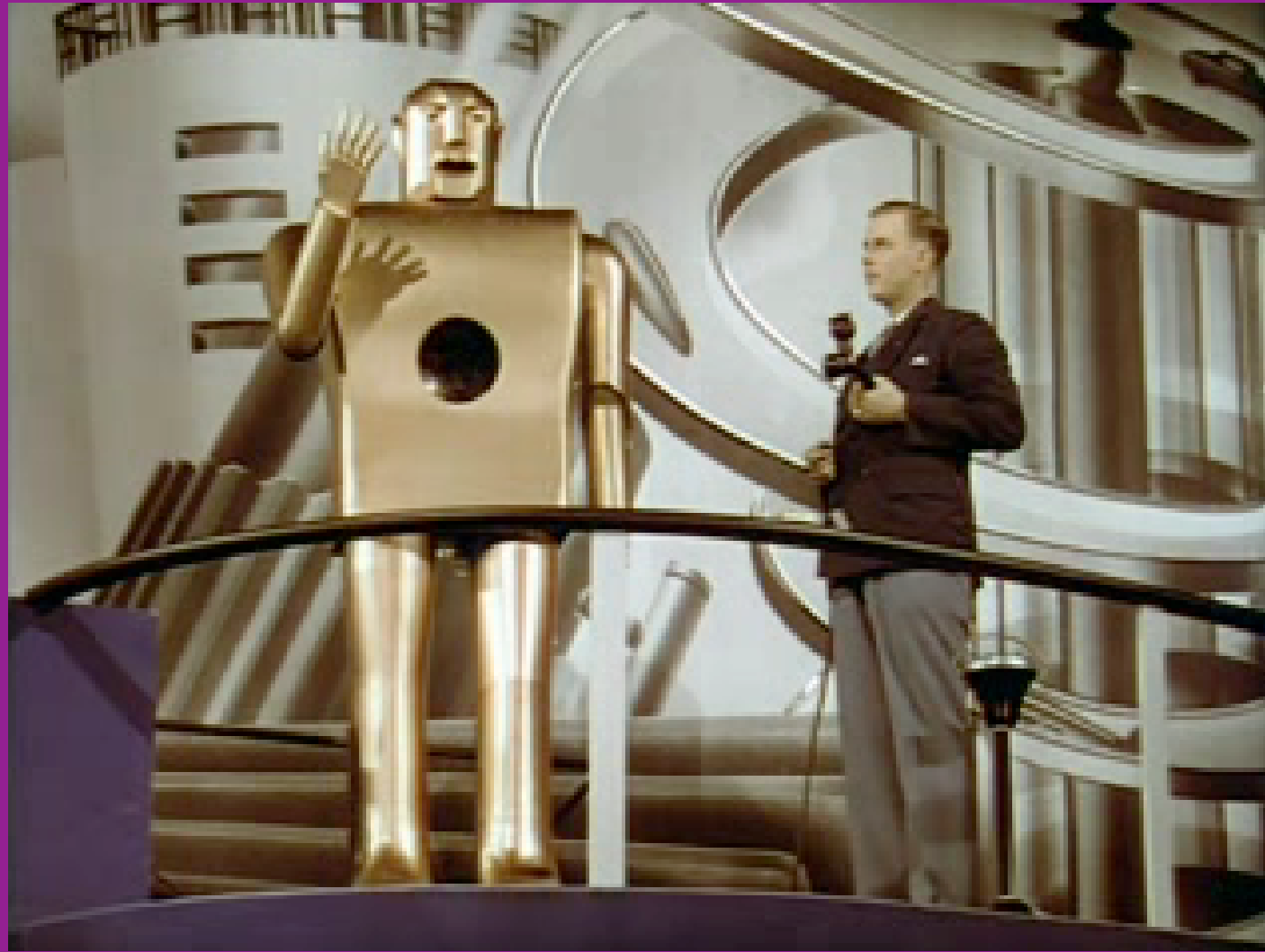
## Listing posts

Title	Body	Published	
Another title	Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.	true	<a href="#">Show</a> <a href="#">Edit</a> <a href="#">Destroy</a>

[New post](#)

Fertig

YSlow 0.216s



# ***Demo***

## **Ajax on Rails**

# Testing



```
class PostsControllerTest < ActionController::TestCase

  def test_should_get_index
    get :index
    assert_response :success
    assert_not_nil assigns(:posts)
  end

end
```





Männlein, tu Dich setzen!  
Das Teufel winket  
mit ewigen Ketten!

**Debugging**

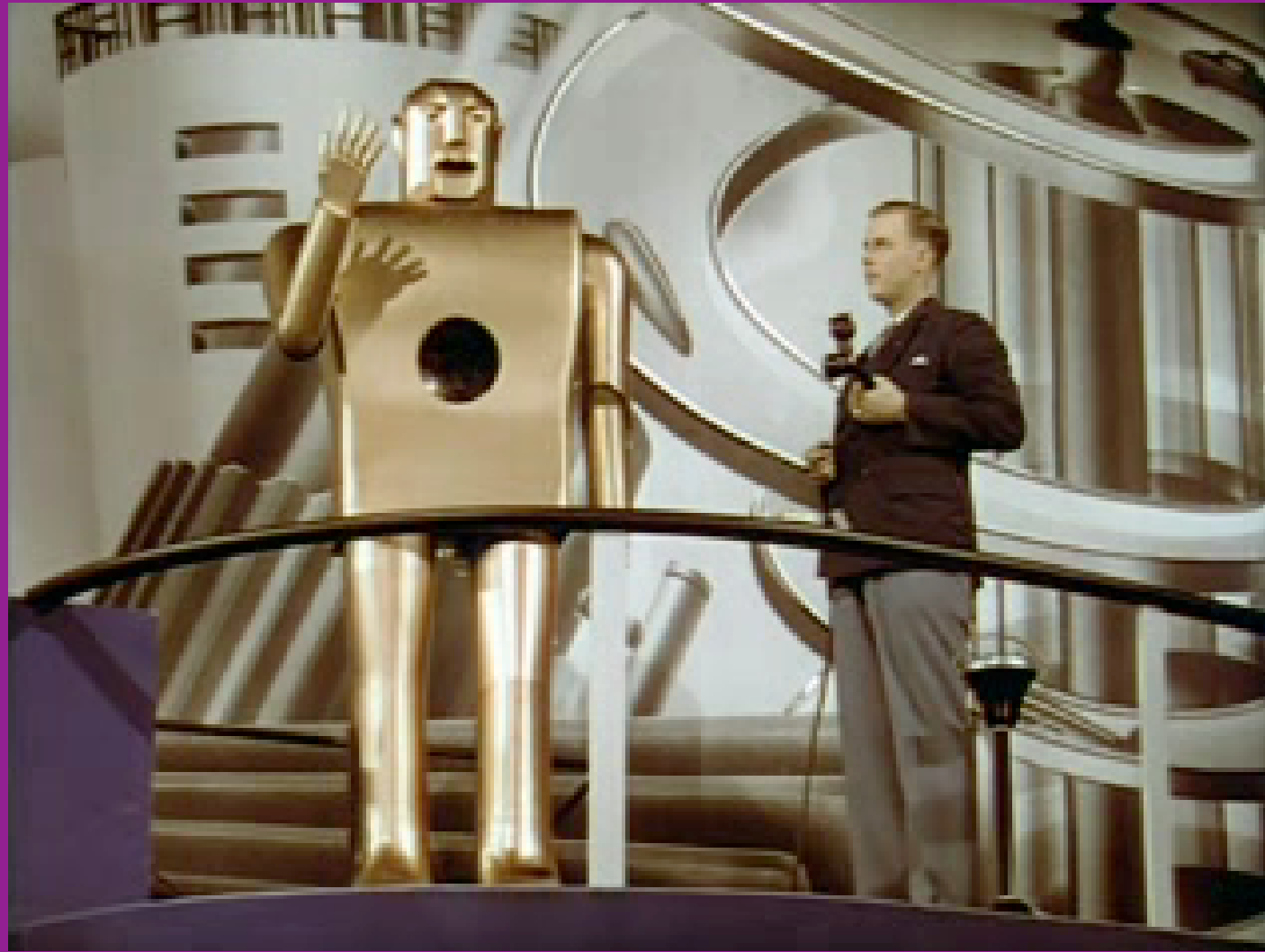
Wirst brav sein!



super-easy way to really find out  
what happens in your code

```
def index  
  @posts = Post.find(:all)  
  debugger
```

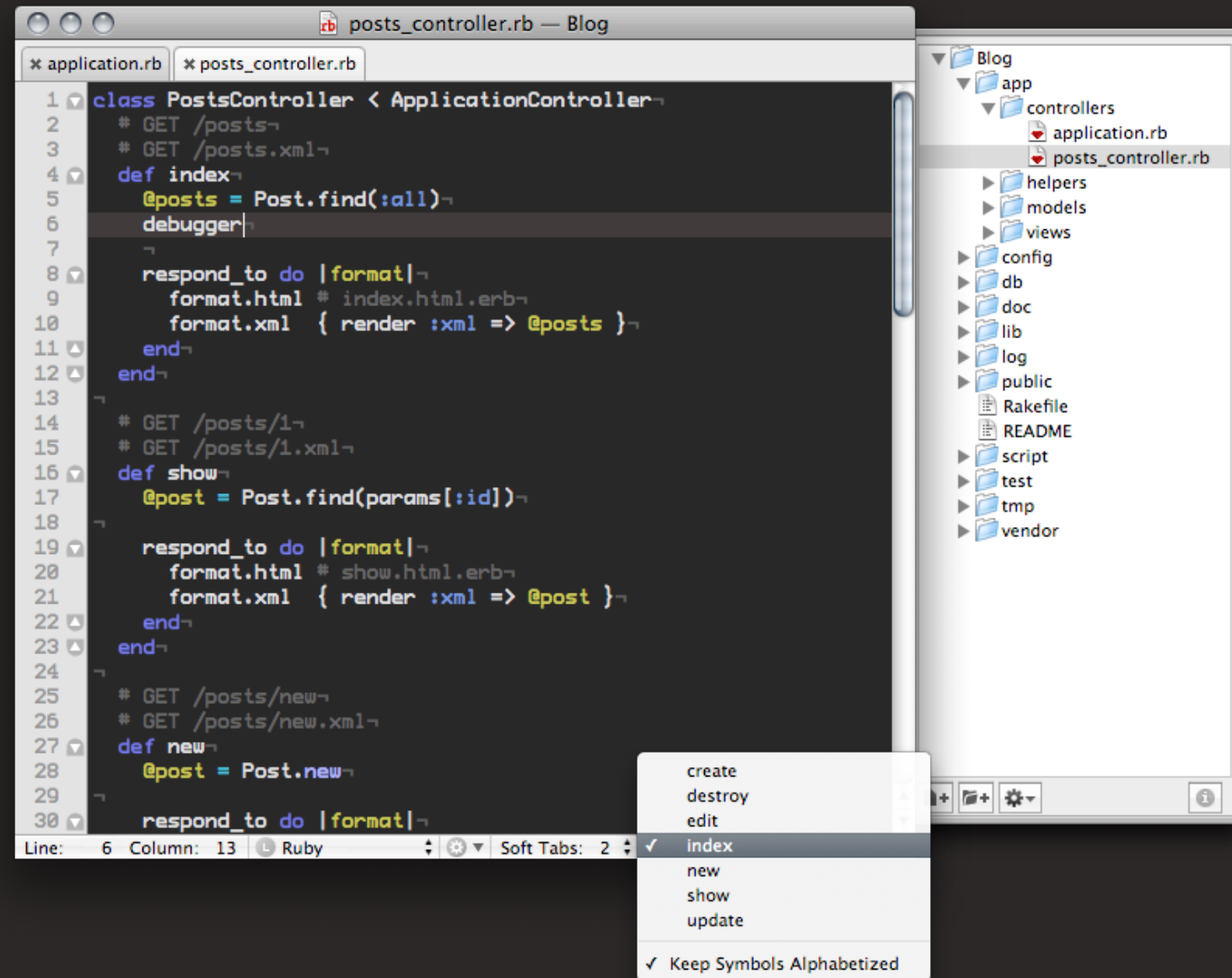
```
    respond_to do |format|  
      format.html  
      format.xml { render :xml => @posts }  
    end  
  end
```



# ***Demo*** **Debugging**



# Rails Ecosystem



# IDEs and Editors like Textmate

rails's rails at master — GitHub

git http://github.com/rails/rails/tree/master

github  
git repository hosting

Home Pricing and Signup Login

Source Browser Commits Wiki Network (79) Watchers (558)

master all branches all tags

rails / rails fork watch download this repo is viewable by everyone

Description: Ruby on Rails  
Homepage: <http://rubyonrails.org>  
Clone URL: <git://github.com/rails/rails.git>

Added Rails.public\_path to control where HTML and assets are expected to be loaded from (defaults to Rails.root + "/public") #11581 [nicksieger]

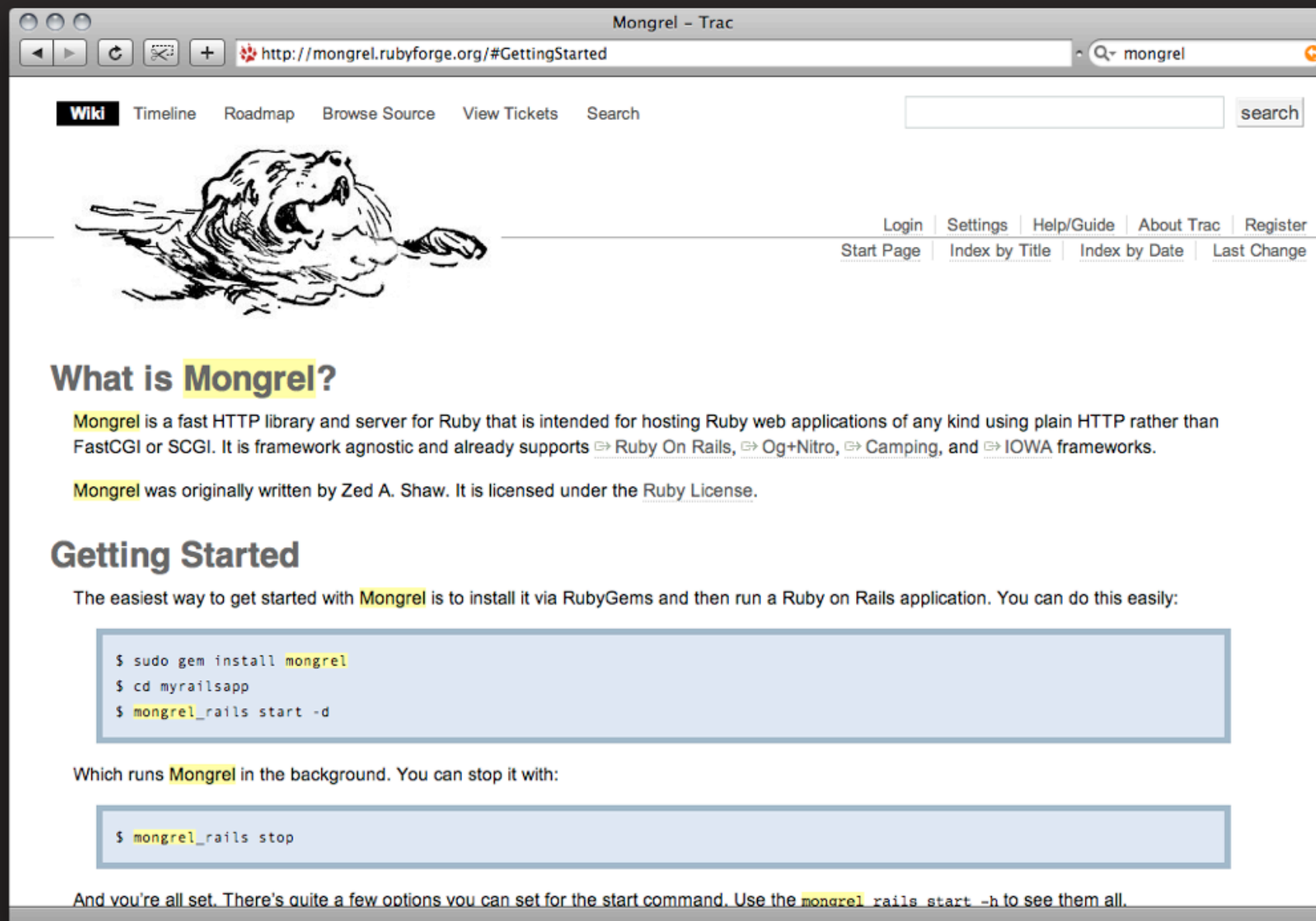
dhh (author)  
about 20 hours ago

commit 420c4b3d8878156d04f45e47050ddc62ae00c68c  
tree e56d27d2078a7553b161a1088594e147e25ed2da  
parent be7fd8168aa4776dccfe2e670cb9451204449dd4

rails /

name	age	message	history
Rakefile	December 18, 2007	Toplevel test task uses the same rake it was in... [jeremy]	
actionmailer/	April 05, 2008	Avoid modifying the sendmail_settings hash when... [NZKoz]	
actionpack/	about 20 hours ago	Added Rails.public_path to control where HTML a... [dhh]	
activemodel/	April 05, 2008	Fix more typos and changelog [lifo]	
activerecord/	3 days ago	Revert "Ensure that save on child object f... [NZKoz]	
activeresource/	April 05, 2008	Fix more typos and changelog [lifo]	
activesupport/	1 day ago	Time#since behaves correctly when passed a Dura... [abuesina]	

# Web services like github




# Web- and Application Server software

RailsConf Europe 2008 - O'Reilly Conferences, 02 September - 04, 2008, Berlin, Germany

http://en.oreilly.com/railsconf2008/public/content/home railsconf europe

**RAILSConf EUROPE** 02-04.09<sup>08</sup> BERLIN, GERMANY

HOME | PROPOSALS | ABOUT | EXHIBIT HALL | CONNECT | MORE RAILS






SIGN UP FOR THE NEWSLETTER

**THE RAILSConf EUROPE 2008**


**RAILS TURNS 2.0!**

With the release of Rails 2.0 at the end of 2007, Rails has moved into the next phase of its evolution. It's no longer a newcomer; it's a mature technology, with a tremendous and still rapidly growing presence in the world of Web development. And Rails has stayed true to its roots, with programmer comfort, productivity, and maintainability as the hallmarks of its architecture and style.

**STAY CONNECTED**


-  RailsConf Europe RSS Feed
-  Tag with del.icio.us
-  Sign Up for the Newsletter (login required)

co-presented by


 **Ruby Central** Inc.

**O'REILLY**

**DIAMOND SPONSOR**

 **Sun** microsystems

**PREMIER MEDIA PARTNER**

 **LINUX JOURNAL**  
The Premier Linux Magazine

**SPONSOR OPPORTUNITIES**  
For information on exhibition and sponsorship

# Conferences



# ActiveRecord Relationships

a Ruby on Rails cheat sheet guide by Amy Hoy

note: has\_many and has\_one work identically except for the number of relationships. the table/model that feature the foreign

## belongs to

```
create table customers (
  id int auto_increment primary key,
  name varchar(75),
  company_id int
)
```

customers belongs\_to :company

company\_id

## has many

```
create table companies (
  id int auto_increment primary key,
  name varchar(75)
)
```

companies has\_many :customers

id

## has and belongs to many

```
create table articles (
  id int auto_increment primary key,
  name varchar(75),
  body text
)
```

articles has\_and\_belongs\_to\_many :authors

id

```
create table authors (
  id int auto_increment primary key,
  name varchar(75)
)
```

authors has\_and\_belongs\_to\_many :articles

id

articles\_authors

article\_id

author\_id

note: the articles\_authors table is implicit due to the two models that call *has\_and\_belongs\_to\_many*, and it does not require a model of its own. when making implicit mapping tables such as this, it must be named tablea\_tableb where a is first in alphabetical order.

© 2005 Amy Hoy / amy@infocookie.com / www.slash7.com  
MIT License — see <http://www.slash7.org/cheats/license.html>

```
def javascript(*files)
  content_for(:head) { javascript_include_tag(*files) }
end

def stylesheet(*files)
  content_for(:head) { stylesheet_link_tag(*files) }
end

def render_stars(rating)
  content_tag :div, star_images(rating), :class => 'stars'
end

def star_images(rating)
  (0...5).map do |position|
    star_image(((rating-position)*2).round)
  end.join
end

def star_image(value)
  image_tag "/images/#{star_type(value)}_star.gif", :size => '15x15'
end

def star_type(value)
  if value <= 0
    'empty'
  elsif value == 1
    'half'
  else
    'full'
  end
end
```

# Podcasts, Books and other Publications



Developing Rails Applications on Mac OS X Leopard

http://developer.apple.com/tools/developonrailsleopard.html

mac os x leopard re

**Developer Connection**

Search  
Advanced Search

Log In | Not a Member? [Contact ADC](#)

[ADC Home](#) > [Tools](#) >

## Developing Rails Applications on Mac OS X Leopard

Ruby on Rails is a popular and powerful open source web framework for rapidly creating high-quality web applications to help you keep up with the speed of the Web. Rails is thriving on Mac OS X, and Leopard comes pre-installed with Ruby, Rails, Mongrel, Capistrano, Subversion, and other tools that help to streamline the development and deployment of Rails applications. In addition, the Organizer feature of XCode 3.0 keeps your development workflow efficient.

This article gives you a full tour of Ruby on Rails 2.0 on Leopard—starting with building a web application using the latest Rails features with Xcode 3.0, and finishing with deploying the application to a production server running Leopard Server. Along the way we'll explore unique features and benefits that Leopard brings to the party. In the end you'll be better equipped to consider the advantages of powering your web application with Rails on Leopard.

This is the first in a series of three articles:

- This article on Development, where you learn to build a basic RESTful Rails application using Xcode 3.0;
- Customization, where we discuss working with views and web forms, adding AJAX support, and supporting an iPhone interface;

# Operating Systems

prototype

*script.aculo.us*

"Spinoffs"





Q+A