37signals

Pursuing beauty with Ruby on Rails

David Heinemeier Hansson

Beauty leads to happiness,

happiness leads to productivity,

Thus

Beauty leads to productivity

You can recognize truth by its beauty and simplicity. When you get it right, it is obvious that it is right.

Richard Feynman, Scientist

```ruby
Account.transaction(david, mary) do
  david.withdrawal(100)
  mary.deposit(100)
end
```

```ruby
class Account < ActiveRecord::Base
  validates_presence_of     :subdomain, :name, :email_address, :password
  validates_uniqueness_of   :subdomain
  validates_acceptance_of   :terms_of_service, :on => :create
  validates_confirmation_of :password, :email_address, :on => :create
end
```

```ruby
class Project < ActiveRecord::Base
  belongs_to              :portfolio
  has_one                 :project_manager
  has_many                :milestones
  has_and_belongs_to_many :categories
end
```

```ruby
class Project < ActiveRecord::Base
  belongs_to              :portfolio
  has_one                 :project_manager, :class_name => "Person"
  has_many                :milestones, :dependent => true
  has_and_belongs_to_many :categories, :join_table => "categorizations"
end
```

```ruby
class Account < ActiveRecord::Base
  has_many :people do
    def find_or_create_by_name(name)
      first_name, *last_name = name.split
      last_name = last_name.join " "

      find_or_create_by_first_name_and_last_name(first_name, last_name)
    end
  end
end

person = Account.find(:first).people.find_or_create_by_name("Richard Feynman")
person.first_name # => "Richard"
person.last_name  # => "Feynman"
```

```ruby
class Post < ActiveRecord::Base
  has_many :comments
end

class Comment < ActiveRecord::Base
  def self.search(query)
    find(:all, :conditions => [ "body = ?", query ])
  end
end

# SELECT * from comments WHERE post_id = 1 AND body = 'hi'
Post.find(1).comments.search "hi"
```

```ruby
class Story < ActiveRecord::Base
  belongs_to   :iteration
  acts_as_list :scope => :iteration
end

story.move_higher
story.move_to_bottom
```

```ruby
class Author < ActiveRecord::Base
  has_many :authorships
  has_many :books, :through => :authorships
end

class Book < ActiveRecord::Base
  has_many :authorships
  has_many :authors, :through => :authorships
end

class Authorship < ActiveRecord::Base
  belongs_to :author
  belongs_to :book
end


david, awrd = Author.find(1), Book.find(1)
david.authorships.create(
  :book => awrd, :contribution => "partial", :written_in => "Denmark"
)

david.authorships.first.written_in # => "Denmark"
david.books.first # => awrd
```

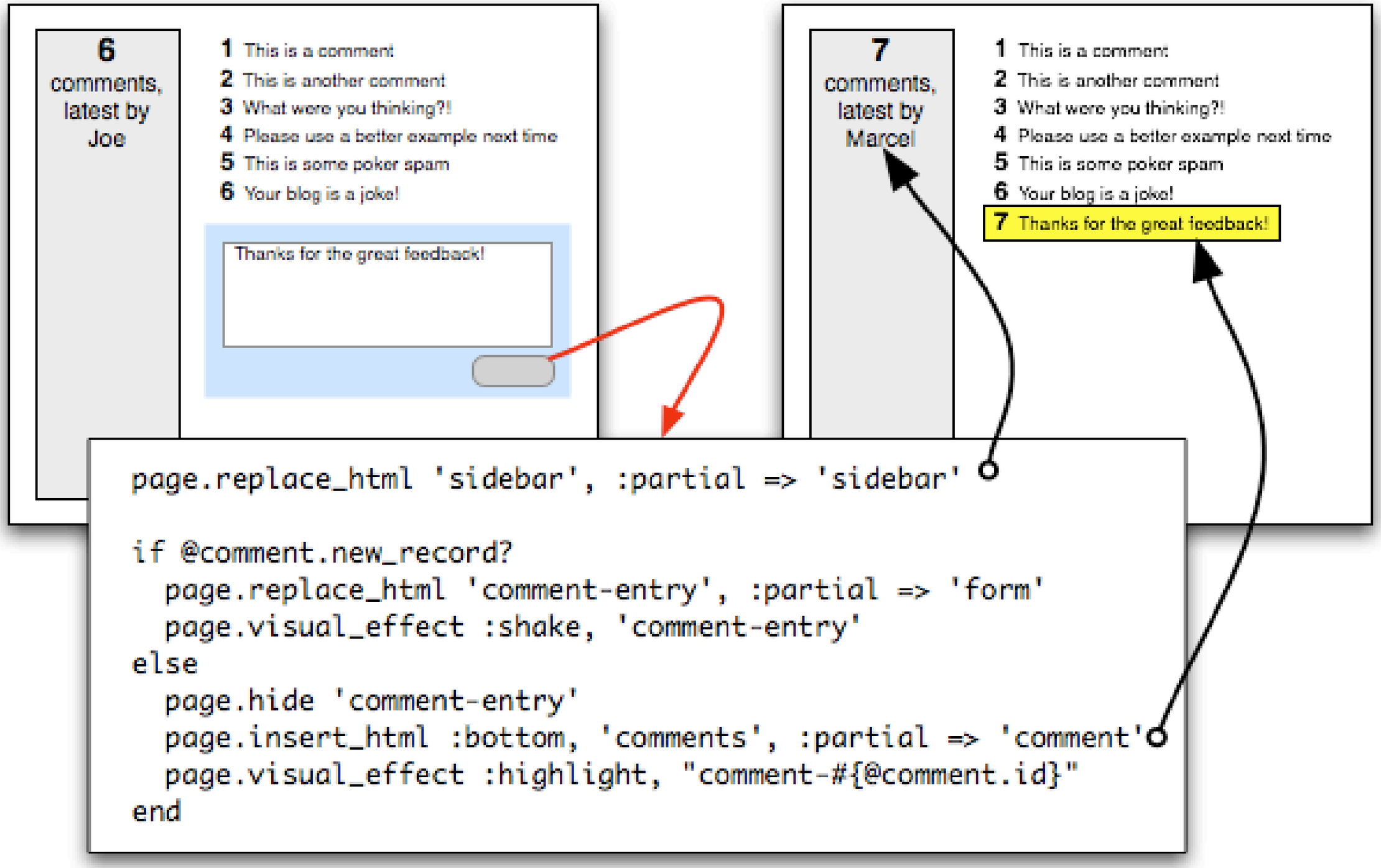
```ruby
class Person < ActiveRecord::Base
  has_many :taggings, :as => :taggable, :dependent => true
  has_many :tags, :through => :taggings
end

class Message < ActiveRecord::Base
  has_many :taggings, :as => :taggable, :dependent => true
  has_many :tags, :through => :taggings
end

class Tagging < ActiveRecord::Base
  belongs_to :tag
  belongs_to :taggable, :polymorphic => true
end

class Tag < ActiveRecord::Base
  has_many :taggings

  def on(*taggables)
    taggables.each { |taggable| taggings.create :taggable => taggable }
  end

  def tagged
    taggings.collect { |tagging| tagging.taggable }
  end
end
```

```ruby
david   = Person.find(1)
welcome = Message.find(1)
summer  = Tag.find_or_create_by_name "Summer"

summer.on(david, welcome)

david.tags   # => [ summer ]
welcome.tags # => [ summer ]

summer.tagged # => [ david, welcome ]
```

```ruby
class Person < ActiveRecord::Base
  acts_as_taggable
end

class Message < ActiveRecord::Base
  acts_as_taggable
end
```

```ruby
class StoryController < ApplicationController
  session        :off,                :only => :feed
  verify         :method => :post, :only => [ :comment ]
  before_filter :authenticate,    :only => [ :new, :edit ]

  def show
    @story = Story.find(params[:id])
  end

  def new
    if request.story?
      @story = Story.create(params[:story])
      cookies[:last_story] = {
        :value   => @story.created_at,
        :expires => 20.years.from_now
      }

      flash[:notice] = "Created new story"
      redirect_to :action => "show", :id => @story
    else
      @story = Story.new
      render :layout => "creation"
    end
  end
end
```

```
<h1>Create a new iteration</h1>

<p>
  You already have <%= pluralize(@iterations.size, "iteration") %>.

  The last one started like this:
  "<%= truncate(@iterations.first.description, 10) %>"
</p>

<% form_for :iteration, Iteration.new do |i| %>
  Title:<br/>
  <%= i.text_field :title %>

  Description:<br/>
  <%= i.text_area :title, :size => "20x40" %>

  Completion target:<br/>
  <%= i.date_select :completed_on, :discard_year => true %>

  <% fields_for :story, Story.new do |s| %>
    Headline:<br/>
    <%= s.text_field :headline, :length => 30, :class => "big" %>

    Difficulty:<br/>
    <% s.select :difficulty, %w( easy medium hard) %>
  <% end %>
<% end %>

...or <%= link_to "Cancel", :action => "overview" %>
```


6
comments,
latest by
Joe
1 This is a comment
2 This is another comment
3 What were you thinking?!
4 Please use a better example next time
5 This is some poker spam
6 Your blog is a joke!
Thanks for the great feedback!
7
comments,
latest by
Marcel
1 This is a comment
2 This is another comment
3 What were you thinking?!
4 Please use a better example next time
5 This is some poker spam
6 Your blog is a joke!
7 Thanks for the great feedback!
page.replace_html 'sidebar', :partial => 'sidebar'
if @comment.new_record?
page.replace_html 'comment-entry', :partial => 'form'
page.visual_effect :shake, 'comment-entry'
else
page.hide 'comment-entry'
page.insert_html :bottom, 'comments', :partial => 'comment'
page.visual_effect :highlight, "comment-#{@comment.id}"
end

```ruby
xml.instruct!
xml.rss("version" => "2.0", "xmlns:dc" => "http://purl.org/dc/elements/1.1/") do
  xml.channel do
    xml.title "Backpack"
    xml.link(url_for(:action => "start", :only_path => false))
    xml.description "Tracking changes for your Backpack pages"
    xml.language "en-us"
    xml.ttl "40"

    for event in @events
      xml.item do
        xml.title(event.headline)
        xml.description(event.description)
        xml.pubDate(event.created_at.to_s(:rfc822))
        xml.guid(event.id)
        xml.link(page_url(:id => event.page))
      end
    end
  end
end
```

```ruby
Rails::Initializer.run do |config|
  config.frameworks -= [ :action_web_service ]
  config.load_paths += [ "#{RAILS_ROOT}/app/services" ]
  config.log_level   = :debug

  config.action_controller.session_store         = :active_record_store
  config.action_controller.fragment_cache_store = :file_store

  config.active_record.default_timezone = :utc
  config.active_record.schema_format    = :ruby
end
```

```
ActionController::Routing::Routes.draw do |map|
  map.start   '', :controller => 'pages', :action => 'start'
  map.signup  'signup', :controller => 'account', :action => 'new'
  map.page    'page/:id', :controller => 'pages', :action => 'show'
  map.connect 'page/:id/:action', :controller => 'pages'
  map.connect 'images/icons/:icon_file', :controller => "assets", :action => "missing_icon"

  map.feed    'feed/:token', :controller => "account", :action => "feed"
  map.ical    'ical/:token', :controller => "reminders", :action => "ical"
end
```

```
set :application, "backpack"
set :repository,  "svn+ssh://somewhere/nice"

set :gateway, "gateway.37signals.com"

role :app, "app1.37signals.com", :primary => true
role :app, "app2.37signals.com"
role :app, "app3.37signals.com"
role :web, "web1.37signals.com", :primary => true
role :web, "web2.37signals.com"
role :db,  "db1.37signals.com",  :primary => true
```

Migrations
Plugins
SwitchTower (and Gauge)
Functional and unit testing
Hooks and callbacks
Action Mailer
Action Web Services
Liquid
Databases and web servers

Questions?

david@loudthinking.com