Ruby is Awesome

Vitaly Kushner
astrails.com

Blocks

```ruby
array.each { |e| puts }

array.each { |e| puts e}

array.each do |e|
  puts e
end
```

```
map     {|x| ...}
collect {|x| ...}
select  {|x| ...}
reject  {|x| ...}
all?    {|x| ...}
sort    {|a, b| ...}

find    {|x| ...}
any?    {|x| ...}
```

```
File.readlines("foobar.dat") .
```

```
File.readlines("foobar.dat") .
  map {|line| line.strip} .
```

```
File.readlines("foobar.dat") .
  map {|line| line.strip} .
    sort {|x,y| x.to_i <=> y.to_i} .
```

```
File.readlines("foobar.dat") .
  map {|line| line.strip} .
    sort {|x,y| x.to_i <=> y.to_i} .
      map {|x| x.upcase} .
```

```
File.readlines("foobar.dat") .
  map {|line| line.strip} .
    sort {|x,y| x.to_i <=> y.to_i} .
      map {|x| x.upcase} .
        to_xml
```

```ruby
class Hash
  def to_html_attributes
    map { |k, v|
      "#{k}='#{v}'"
    }.join(' ')
  end
end
```

```ruby
attrs = {
    :src => "foo.img",
    :width => 100,
    :height => 200,
    :class => "avatar"
}
attrs.to_html_attributes

=> "class='avatar' height='200' width='100' src='foo.img'"
```

Closures

```
class Array
  def has_any_bigger_then(x)
    any? {|e| e > x}
  end
end
```

```ruby
def incrementor(x)
  proc {|y| y + x}
end

inc1 = incrementor(1)

inc5 = incrementor(5)
```

```ruby
def incrementor(x)
  proc {|y| y + x}
end

inc1 = incrementor(1)

inc1.call(1)
=> 2
inc1.call(3)
=> 4
```

```ruby
def incrementor(x)
  proc {|y| y + x}
end

inc5 = incrementor(5)

inc5.call(1)
=> 6
inc5.call(3)
=> 8
```

```ruby
# ruby
def paidMore(amount)
  proc {|e| e.salary > amount}
end
```

```csharp
// C#
public Predicate<Employee> PaidMore
(int amount) {
  return delegate(Employee e) {
    return e.Salary > amount;
  }
}
```

Meta Programming

Extending the language to create new abstractions

Monkey Patching

```ruby
class Hash
  def to_html_attributes
    map { |k, v|
      k + '="' + v + '"'
    }.join(' ')
  end
end
```

```ruby
attrs = {
    :src => "foo.img",
    :width => 100,
    :height => 200,
    :class => "avatar"
}
attrs.to_html_attributes
```

```
=> 'class="avatar" height="200" width="100" src="foo.img"'
```

```
3.megabytes
=> 3145728
```

```
10.months.from_now
```

```
=> Thu Aug 12 03:25:40 0300 2010
```

```
5.minutes.ago
```

=> Mon Oct 12 03:21:02 0200 2009

```
user_names = users.map(&:name)
```

```
class Symbol
  def to_proc
    Proc.new { |x| x.__send__(self) }
  end
end
```

```
describe User do
  context "first_name" do
    it "should return full_name up to the first space" do
      user = User.new :full_name => "John Doe"

      user.first_name.should == "John"
    end
  end
end
```

```
response.should redirect_to("/products")

response.should render_template("index")

response.should have_text("welcome")
```

Duck Typing

.to_s

```
[1, 2, 3].to_s
=> "123"
```

.to_s

```
123.to_s
=> "123"
```

.to_s

```
"123".to_s
=> "123"
```

```
"a#{b}c"
```

```
"a" + b.to_s + c
```

method_missing

NoMethodError

User.find_by_company("Astrails")

```
User.find_by_name_and_company(
"Vitaly Kushner", "Astrails")
```

Multiple inheritance is

Evil :-)

Modules

a.k.a. Mixins

```
module FlyHome
  def set_home
    @home = position
  end

  def fly_home
    fly(@home)
  end
end
```

```
class Bird < Living
  include FlyHome
  def fly(direction) ...
  def position ...
end
```

```
class Airplane < Machine
  include FlyHome
  def fly(direction) ...
  def position ...
end
```

DSL

Domain Specific Language

```ruby
class User < ActiveRecord::Base
  has_many :projects
  belongs_to :account
  has_many :reports,
    :class_name => "User",
    :foreign_key => :reports_to_id
  validates_uniqueness_of :name
  validates_presence_of :crypted_password
end
```

```
safe do
  local { path "/backup/:kind" }
  s3 :key => YOUR_S3_KEY,
    :secret => YOUR_S3_SECRET,
    :bucket => S3_BUCKET,
    :path => ":kind/"
  gpg :password => "foobar"

  mysqldump do
    options "-ceKq --single-transaction --create-options"
    user "astrails"
    password "foobar"

    database :blog
    database :astrails_com do
      skip_tables :request_logs
    end
  end

  tar do
    archive "etc-files" do
      files "/etc"
    end
  end

end
```

```
safe do
  local { path "/backup/:kind" }
  s3 :key => YOUR_S3_KEY,
     :secret => YOUR_S3_SECRET,
     :bucket => S3_BUCKET,
     :path => ":kind/"
  gpg :password => "foobar"

  ...
end
```

```
safe do
  ...
  mysqldump do
    options "-ceKq ..."
    user "astrails"
    password "foobar"

    database :blog
    database :astrails_com do
      skip_tables :request_logs
    end
  end
  ...
end
```

```
safe do
  ...

  tar do
    archive "etc-files" do
      files "/etc"
    end
  end

end
```

Rubygems

```
➜  ~  ✗ sudo gem install astrails-safe
Successfully installed astrails-safe-0.2.3
1 gem installed
Installing ri documentation for astrails-safe-0.2.3...
Installing RDoc documentation for astrails-safe-0.2.3...
➜  ~  ✗
```

Rails

http://www.flickr.com/photos/ecstaticist/2589723846/

Rails is Fun

optimized for
programmers happiness
and sustainable
productivity

MVC

MVC

- Model

MVC

- Model
- View

MVC

- Model
- View
- Controller

convention over

configuration

```ruby
class User < ActiveRecord::Base
  table_name 'users'
end
```

```ruby
class User < ActiveRecord::Base
  table_name 'users'
end
```

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

```
User.find(123)
=>  SELECT * FROM `users` WHERE (`users`.`id` = 123)
```

```ruby
class User < ActiveRecord::Base
  belongs_to :account
  has_many :projects
end

class UsersController < ApplicationController
  def show
    @user = User.find(params[:id])
    render :template => 'show'
  end
end
```

```ruby
class User < ActiveRecord::Base
  belongs_to :account
  has_many :projects
end

class UsersController < ApplicationController
  def show
    @user = User.find(params[:id])
    render :template => 'show'
  end
end
```

```ruby
class User < ActiveRecord::Base
  belongs_to :account
  has_many :projects
end

class UsersController < ApplicationController
  def show
    @user = User.find(params[:user])
  end
end
```

http://www.flickr.com/photos/slyadnev/3959052112/

Models

ActiveRecord

ORM

Object-Relational Mapping

- MySQL
- PostgreSQL
- MSSql
- SQLite
- Oracle
- ODBC

Associations

```
class User < ActiveRecord::Base
  has_many :projects
end

class Project < ActiveRecord::Base
  belongs_to :user
end
```

```
u = User.find(1)
=>  SELECT * FROM `users` WHERE (`users`.`id` = 1)
```

```
u.projects.count
=>  SELECT count(*) AS count_all FROM `projects` WHERE
(`projects`.user_id = 1)
```

```
User.find_by_first_name("Vitaly", :include => :projects)
=> SELECT * FROM `users` WHERE
  (`users`.`first_name` = 'Vitaly') LIMIT 1
=> SELECT `projects`.* FROM `projects` WHERE
  (`projects`.user_id = 1)
```

```
Project.first.user
=>  SELECT * FROM `projects` LIMIT 1
=>  SELECT * FROM `users` WHERE (`users`.`id` = 1)
```

Callbacks

- before_validation
- before_validation_on_create
- after_validation
- after_validation_on_create
- before_save
- before_create
- after_create
- after_save

Callbacks

- before_validation
- before_validation_on_create
- after_validation
- after_validation_on_create
- before_save
- before_create
- after_create
- after_save

```ruby
class User < ActiveRecord::Base
  before_save :encrypt_password
  attr_accessor :password
  private
  def enctypt_password
    unless password.blank?
      self.salt ||= ActiveSupport::SecureRandom.hex(40)
      self.crypted_password = encrypt(password, salt)
    end
  end
end
```

```ruby
class User < ActiveRecord::Base
  def password=(password)
    unless password.blank?
      self.salt ||= ActiveSupport::SecureRandom.hex(40)
      self.crypted_password = encrypt(password, salt)
    end
  end
end
```

Validations

```ruby
validates_presence_of :account_id
validates_uniqueness_of :name
validates_numericality_of :ccv

def validate
  unless name[0] == ?A
    errors.add(:name, "Names must start with an A")
  end
end
```

ActiveSupport

```ruby
3.days.ago

[1,2,3].to_json

2.weeks.from_now

hash.slice(:foo, :bar)
```

http://www.flickr.com/photos/benjamin-nagel/2902721172/

Controllers

```ruby
class UsersController < ApplicationController
  def index
    @users = User.paginate(:page => params[:page],
      :per_page => 20)
  end
end
```

http://yourdomain/users/index

http://localhost:3000/users/index

```
ActionController::Routing::Routes.draw do |map|
    map.connect ':controller/:action/:id'
    map.connect ':controller/:action/:id.:format'
end
```

```
ActionController::Routing::Routes.draw do |map|
    map.connect ':controller/:action/:id'
    map.connect ':controller/:action/:id.:format'
end
```

/welcome/home

```
:controller => 'welcome'

:action => 'home'
```

```
ActionController::Routing::Routes.draw do |map|
    map.connect ':controller/:action/:id'
    map.connect ':controller/:action/:id.:format'
end
```

/users/edit/123

```
:controller => 'users'

:action => 'edit'

:id => '123'
```

```
ActionController::Routing::Routes.draw do |map|
    map.connect ':controller/:action/:id'
    map.connect ':controller/:action/:id.:format'
end
```

projects/show/456.xml

```
:controller => 'projects'
:action => 'show'
:id => '456'
:format => 'xml'
```

/talks/create?title=ruby%20is%20awsome&author[name]=Vitaly&author[company]=Astrails

```
{
  :controller => 'talks',
  :action => 'create',
  :title => "ruby is awesome",
  :user => {
    :name => 'Vitaly',
    :company => 'Astrails'
  }
}
```

REST

REST

Representational State Transfer

- HEAD
- GET
- POST
- PUT
- DELETE

- index
- new
- create
- show
- edit
- update
- destroy

index

GET /users

new

GET /users/new

create

POST /users

show

GET /users/123

edit

GET /users/123/edit

update

PUT /users/123

destroy

DELETE /users/123

```ruby
map.resources :users do |user|
  user.resources :projects
end
```

GET /users/123/projects

```
:controller => :projects, :action => :index, :user_id => 123
```

GET /users/123/projects/456

```
:controller => :projects, :action => :show, :user_id => 123, :id => 456
```

```
users_path
```

=> /users

new_user_path

=> /users/new

```
user_path(user)
```

=> /users/123

edit_user_path(user)

=> /users/123/edit

Filters

```ruby
class UsersController < ApplicationController
  before_filter :login_required,
    :except => [:new, :create]
  before_filter :find_user,
    :only => [:edit, :show, :update, :destroy]

  def index ...
  def new ...
  def create ...
  def edit ...
  def show ...
  def update ...
  def destroy ...

protected
  def login_required ...

  def find_user ...
end
```

```ruby
class UsersController < ApplicationController
  before_filter :login_required,
    :except => [:new, :create]
  ...
protected
  def login_required
    unless session[:user_id]
      redirect_to "/login"
    end
  end
end
```

```ruby
class UsersController < ApplicationController
  before_filter :find_user,
    :only => [:edit, :show, :update, :destroy]
  ...
protected
  def find_user
    @user = User.find(params[:id])
  end
end
```

```ruby
def index
  @users = User.paginate(
    :page => params[:page], :per_page => 20)
end
```

```ruby
def new
  @user = User.new(params[:user])
end
```

```ruby
def create
  @user = User.new(params[:user])
  if @user.save
    flash[:notice] = "User created"
    redirect_to user_path(@user)
  else
    flash.now[:error] = "Failed to create a user"
    render :action => "new"
  end
end
```

```
def edit
end
```

```
def show
end
```

```ruby
def update
  if @user.update_attributes(params[:user])
    flash[:notice] = "User updated"
    redirect_to user_path(@user)
  else
    flash.now[:error] = "Failed to update user"
    render :action => "edit"
  end
end
```

```ruby
def destroy
  if @user.destroy
    flash[:notice] = "User deleted"
  else
    flash[:error] = "Failed to delete user"
  end
  redirect_to users_path
end
```

http://www.flickr.com/photos/brapke/226101874/

Views

```ruby
def index
  @users = User.paginate(
    :page => params[:page], :per_page => 20)
end
```

app/views/users/index.html.erb

ERB

ERB

```
<p>Hello <%= h(@user.name) %></p>

<ul>
  <% @user.projects.each do |project| %>
    <li><%= h(project.name) %></li>
  <% end %>
</ul>
```

HAML

```
%p
  = @user.name

%ul
  - @user.projects.each do |project|
    %li= project.name
```

Testing

TDD

TDD

Test Driven Development

BDD

BDD

Behavior Driven Development

Pressure

TATFT

TATFT

Test All The Fucking Time

- Test::Unit
- RSpec
- Shoulda
- Cucumber
- Webrat

```ruby
it "should be able to show media" do
  @media = stub_media
  stub(Media).find {@media}
  get :show, :id => @media.id
  response.should be_success
end
```

Migrations

```ruby
class CreateProjects < ActiveRecord::Migration
  def self.up
    create_table :projects do |t|
      t.integer :user_id, :null => false
      t.string :title, :null => false
      t.text :description, :null => false
      t.integer :budget, :null => false

      t.timestamps
    end
  end

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

Plugins

inherited_resourfce

standard implementation for the 7 REST actions

acts_as_tree

db schema and methods for storing trees

country_select

HTML helper to present a select box of countries

Rails Scales?

http://www.flickr.com/photos/pinksherbet/223270526/

Rails Scales?

YES

if you have no
scalability problems

you are not growing fast enough!

Ruby is Awesome

Q & A

Vitaly Kushner
astrails.com