

# Camping, a Microframework

[www.rug-b.com/wiki/show/schmidt](http://www.rug-b.com/wiki/show/schmidt)

2007-10-04





\_why



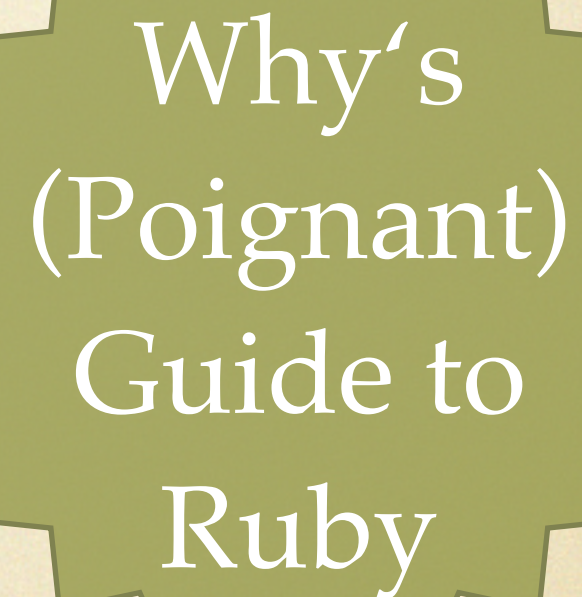
# \_why

- Shoes
- Hackety Hack
- Hpricot
- Sandbox
- RedCloth
- Syck - Ruby's YAML parser



# \_why

- Shoes
- Hackety Hack
- Hpricot
- Sandbox
- RedCloth
- Syck - Ruby's YAML parser



Why's  
(Poignant)  
Guide to  
Ruby





[code.whytheluckystiff.net/camping/](http://code.whytheluckystiff.net/camping/)



```

%w[active_support markaby tempfile uri].map{|| require l}
module Camping; Apps=[]; C=self; S=IO.read(__FILE__).sub(/S=I.+$/,"")
P="Cam\ping Problem!"; module Helpers; def R c,*g;p=/\.(+?\)/; g.inject(c.
  urls.find{|x|x.scan(p).size==g.size}.dup){|s,a|s.sub p,C.escape((a[
    a.class.primary_key]rescue a))}end; def URL c='/',*a;c=R(c,*a)if c.
respond_to?:urls;c=self/c;c="//"+@env.HTTP_HOST+c if c[/^\//]; URI(c)end; def/p
p[/^\//]?@root+p : p end; def errors_for o;ul.errors{o.errors.each_full{|x|li x}
  }if o.errors.any?end end; module Base; include Helpers; attr_accessor:input,
  :cookies,:env,:headers,:body,:status,:root; def method_missing*a,&b
a.shift if a[0]==:render;m=Mab.new({},self);s=m.capture{send(*a,&b)};s=m.layout{s}if
  /^\_!~a[0].to_s and m.respond_to?:layout;s end; def r s,b,h={};@status=s;@headers.
  merge!h;@body=b end; def redirect*a;r 302,"Location"=>URL(*a)end; Z="\r\n"
def initialize r,e,m;e=H[e.to_hash];@status,@method,@env,@headers,@root=200,m.
  downcase,e,{Content-Type=>"text/html"},e.SCRIPT_NAME.sub(/\//$,"")
  @k=C.kp e.HTTP_COOKIE;q=C.qs_parse e.QUERY_STRING;@in=r
  if%r|\Amultipart/form-.*boundary="\?([\^\",;]+)|n.match e.CONTENT_TYPE
b=/(?:\r?\n|\A)#(Regexp::quote("--#\$1"))(?:--)?\r$/;until@in.eof?:fh=H[];for l in@in
  case l;when Z;break;when/^Content-D.+?: form-data;/;fh.u H[*$'.
  scan(/(?:\s(\w+)=("[^"]+)")/.flatten];when/^Content-Type: (.+?)(\r$|\Z)/m;fh[
:type]=$1;end;end;fn=fh[:name];o=if fh[:filename];o=fh[:tempfile]=Tempfile.new(:C)
o.binmode;else;fh=""end;while l=@in.read(16384);if l=~b;o<=$`.chomp;@in.seek(-$'.
  size,IO::SEEK_CUR);break;end;o<<l;end;q[fn]=fh if fn;fh[:tempfile].rewind if
fh.is_a?H;end;elsif@method=="post";q.u C.qs_parse(@in.read)end;@cookies,@input=
  @k.dup,q.dup end; def service*a;@body=send(@method,*a)if respond_to?@method
  @headers["Set-Cookie"]=@cookies.map{|k,v|"#{k}="#{C.escape(v)}"; path="#{self/'/'
  }"if v!=@k[k]}-[nil];self end; def to_s;"Status: #{@status}#{Z+@headers.map{|k,v|
  [*v].map{|x|"#{k}: #{x}"}*Z+Z*2+@body}"end;end

```

```

X=module Controllers;@r=[];class<<self;def r;@r;end;def R*u;r=@r;Class.new{
meta_def(:urls){u};meta_def(:inherited){|x|r<<x}}end;def M;def M;end;constants.map{|c|
k=const_get(c);k.send:include,C,Base,Models;r[0,0]=k if !r.include?k;k.meta_def(
:urls){|"/#{c.downcase}|"}if !k.respond_to?:urls}end;def D p;r.map{|k|k.urls.
map{|x|return k,$~[1..-1]if p=~/^#{x}\/?$/}};[NotFound,[p]]end end;class
NotFound<R();def get p;r(404,Mab.new{h1 P;h2 p+" not found"})end end;class
ServerError<R();def get k,m,e;r(500,Mab.new{h1 P;h2"#{k}."#{m}";h3"#e.class
  }#{e.message}.";ul{e.backtrace.each{|bt|li(bt)}}}.to_s)end end;self;end;class<<
self;def goes m;eval S.gsub(/Camping/,m.to_s).gsub("A\pps=[]","Cam\ping::Apps<<\
self"),TOPLEVEL_BINDING;end;def escape s;s.to_s.gsub(/[\^\w.-]+/n){'%'+($&.
  unpack('H2*$&.size')*'%').upcase}.tr(' ','+')end;def un s;s.tr(' ','').gsub(
/%%([\da-f]{2})/in){[$1].pack('H*')}end;def qs_parse q,d='&';m=proc{|_,o,n|o.u(
n,&m)rescue([*o]<<n)};q.to_s.split(/[#{d}]/n).inject(H[]){|h,p|k,v=un(p).
  split('=',2);h.u k.split(/[\]\[\]/+).reverse.inject(v){|x,i|H[i,x]},&m}end;def
kp s;c=qs_parse(s,',' )end;def run r=$stdin,e=ENV;X.M;k,a=X.D un("/#{e[
'PATH_INFO']}").gsub(/\/+\/\//));k.new(r,e,(m=e['REQUEST_METHOD']||"GET")).Y.
  service *a;rescue Exception=>x;X::ServerError.new(r,e,'get').service(k,m,x)end
def method_missing m,c,*a;X.M;k=X.const_get(c).new(StringIO.new,H['HTTP_HOST',
  ],'SCRIPT_NAME','HTTP_COOKIE',),m.to_s);H.new(a.pop).each{|e,f|k.send(
  " #{e}=",f)}if Hash===a[-1];k.service *a;end;end;module Views;include X,Helpers
end;module Models;autoload:Base,'camping/db';def Y;self;end;end;class
  Mab<Markaby::Builder
include Views;def tag!*g,&b;h=g[-1];[:href,:action,:src].map{|a|(h[a]=self/h[a])rescue
  0};super end end;H=HashWithIndifferentAccess;class H;def method_missing m,*a
  m.to_s=~/=/$/?self[$]=a[0]:a==[]?self[m]:raise(NoMethodError,"#{m}")end
  alias_method:u,:regular_update;end end

```



```

%w[active_support markaby tempfile uri].map{|| require l}
module Camping; Apps=[]; C=self; S=IO.read(__FILE__).sub(/S=I.+$/,")
P="Cam\ping Problem!"; module Helpers; def R c,*g;p=/\.(+?\)/; g.inject(c.
  urls.find{|x|x.scan(p).size==g.size}.dup){|s,a|s.sub p,C.escape((a[
    a.class.primary_key]rescue a))}end; def URL c='/',*a;c=R(c,*a)if c.
respond_to?:urls;c=self/c;c="//"+@env.HTTP_HOST+c if c[/^\//]; URI(c)end; def/p
p[/^\//]?@root+p : p end; def errors_for o;ul.errors{o.errors.each_full{|x|li x}
  }if o.errors.any?end end; module Base; include Helpers; attr_accessor:input,
  :cookies,:env,:headers,:body,:status,:root; def method_missing*a,&b
a.shift if a[0]==:render;m=Mab.new({},self);s=m.capture{send(*a,&b)};s=m.layout{s if
  /\^_/!~a[0].to_s and m.respond_to?:layout;s end; def r s,b,h={};@status=s;@headers.
  merge!h;@body=b end; def redirect*a;r 302,"Location"=>URL(*a)end; Z="\r\n"
def initialize r,e,m;e=H[e.to_hash];@status,@method,@env,@headers,@root=200,m.
  downcase,e,{ 'Content-Type'=>"text/html",e.SCRIPT_NAME.sub(/\//,$/,")
  @k=C.kp e.HTTP_COOKIE;q=C.qs_parse e.QUERY_STRING;@in=r
  if %r|\Amultipart/form-.*boundary="\?([\^";,;]+)|n.match e.CONTENT_TYPE
b=/(?:\r?\n|\A)#(Regexp::quote("--#\$1"))(?:--)?\r$/;until@in.eof?:fh=H[];for l in@in
  case l;when Z;break;when/^Content-D.+: form-data;/;fh.u H[*$'.
  scan(/(?:\s(\w+)=("[^"]+)")/.flatten];when/^Content-Type: (.+?)(\r$|\Z)/m;fh[
:type]=$1;end;end;fn=fh[:name];o=if fh[:filename];o=fh[:tempfile]=Tempfile.new(:C)
o.binmode;else;fh=""end;while l=@in.read(16384);if l=~b;o<<$`.chomp:@in.seek(-$`.
  size,IO::SEEK_CUR);break end;o<<l;end;q[fn]=fh if fn;fh[:tempfile].rewind if
fh.is_a?H;end;elsif @method="post";q.u=q.parse(@in.read)end;@cookies,@input=
  @k.dup,q.dup end;def send*a;@body=send(@method,*a)if respond_to?@method
@headers["Set-Cookie"]=@cookies.map{|k,v| "##{k}=#{v}.escape(%r)}; path=#self/'/'
  }"if v!=@k[k]-[nil];self end;def to_s;"Status: #{@status}#{Z+@headers.map{|k,v|
    [*v].map{|x| "#k: #{x}"}*Z+Z*2+@body}";end;end

```

3.909

Bytes

```

X=module Controllers;@r=[];class<<self;def r;@r;end;def R*u;r=@r;Class.new{
  meta_def(urls){u};meta_def(:inherited){|x|r<<x}}end;def M;def M;end;constants.map{|c|
  k=const_get(c);k.send(:include,C::Base::Models);[0,0]=k if !r.include?k;k.meta_def(
  :url{|x|k.respond_to?(:url)}end;def D p;r.map{|k|k.urls.
  map{|x|x.respond_to?(:url)}if p=~/\^#{x}\//?$/;[NotFound,[p]]end end;class
  NotFound<R;def get p;r=C::Mab.new{h1 P;h2 p+"not found"}}end end;class
  ServerError<R();def get k,m,e;r(500,Mab.new{h1 P;h2"#k.#m";h3"#e.class
    }#{e.message}";ul{e.backtrace.each{|bt|li(bt)}}).to_s)end end;self;end;class<<
self;def goes m;eval S.gsub(/Camping/,m.to_s).gsub("A\pps=[]","Cam\ping::Apps<<\
  self"),TOPLEVEL_BINDING;end;def escape s;s.to_s.gsub(/[\^ \w.-]+/n){'%'+($&.
    unpack('H2*$&.size')*'%').upcase}.tr(' ','+')end;def un s;s.tr(' ','').gsub(
  /%([\da-f]{2})/in){[$1].pack('H*')}end;def qs_parse q,d='&';m=proc{|_,o,n|o.u(
    n,&m)rescue([*o]<<n)};q.to_s.split(/[#{d}]/n).inject(H[]){|h,p|k,v=un(p).
    split('=',2);h.u k.split(/[\]\[\]/n).reverse.inject(v){|x,i|H[i,x]},&m}end;def
    kp s;c=qs_parse(s,;)end;def run r=$stdin,e=ENV;X.M;k,a=X.D un("/#{e[
  'PATH_INFO']}).gsub(/\//+/, '/')};k.new(r,e,(m=e['REQUEST_METHOD']||"GET")).Y.
  service *a;rescue Exception=>x;X::ServerError.new(r,e,'get').service(k,m,x)end
def method_missing m,c,*a;X.M;k=X.const_get(c).new(StringIO.new,H['HTTP_HOST',
  ],'SCRIPT_NAME',"HTTP_COOKIE",),m.to_s);H.new(a.pop).each{|e,f|k.send(
  "#e=",f)}if Hash===a[-1];k.service *a;end;end;module Views;include X,Helpers
end;module Models;autoload:Base,'camping/db';def Y;self;end;end;class
  Mab<Markaby::Builder
include Views;def tag!*g,&b;h=g[-1];[:href,:action,:src].map{|a|(h[a]=self/h[a])rescue
  0};super end end;H=HashWithIndifferentAccess;class H;def method_missing m,*a
  m.to_s=~/=/$/?self[$]=a[0]:a==[]?self[m]:raise(NoMethodError,"#{m}")end
  alias_method:u,:regular_update;end end

```



```

%w[active_support markaby tempfile uri].map{|l| require l}
module Camping; Apps=[]; C=self; S=IO.read(__FILE__).sub(/S=I.+$/, "")
P="Cam\ping Problem!"; module Helpers; def R c,*g;p=/\.(+?\)/; g.inject(c.
  urls.find{|x|x.scan(p).size==g.size}.dup){|s,a| s.sub p,C.escape((a[
    a.class.primary_key]rescue a))}end; def URL c='/',*a;c=R(c,*a)if c.
respond_to?:urls;c=self/c;c="//"+@env.HTTP_HOST+c if c[/^\//]; URI(c)end; def/p
p[/^\//]?@root+p : p end; def errors_for o;ul.errors{o.errors.each_full{|x|li x}
  }if o.errors.any?end end; module Base; include Helpers; attr_accessor:input,
  :cookies,:env,:headers,:body,:status,:root; def method_missing*a,&b
a.shift if a[0]==:render;m=Mab.new({},self);s=m.capture{send(*a,&b)};s=m.layout{s}if
  /\^_/!~a[0].to_s and m.respond_to?:layout;s end; def r s,b,h={};@status=s;@headers.
  merge!h;@body=b end; def redirect*a;r 302,"Location"=>URL(*a)end; Z="\r\n"
def initialize r,e,m;e=H[e.to_hash];@status,@method,@env,@headers,@root=200,m.
  downcase,e,{ 'Content-Type'=>"text/html",e.SCRIPT_NAME.sub(/\//,$/,")
  @k=C.kp e.HTTP_COOKIE;q=C.qs_parse e.QUERY_STRING;@in=r
  if%r|\Amultipart/form-.*boundary="\?([\^\";,]+)|n.match e.CONTENT_TYPE
b=/(?:\r?\n|\A)#(Regexp::quote("--#\$1"))(?--)?\r$/;until@in.eof?:fh=H[];for l in@in
  case l;when Z;break;when/^Content-D.+?: form-data;/;fh.u H[*$'.
  scan(/(?:\s(\w+)=("[^"]+"))/.flatten];when/^Content-Type: (.+?)(\r$|\Z)/m;fh[
:type]=$1;end;end;fn=fh[:name];o=if fh[:filename];o=fh[:tempfile]=Tempfile.new(:C)
o.binmode;else;fh=""end;while l=@in.read(16384);if l=~b;o<<$`.chomp:@in.seek(-$`.
  size,IO::SEEK_CUR);break end;o<<l;end;q[fn]=fh if fn;fh[:tempfile].rewind if
fh.is_a?H;end;elsif@method="post";q.u=q.qs_parse(@in.read)end;@cookies,@input=
  @k.dup,q.dup end;def s=*a;@body=send(@method,*a)if respond_to?@method
  @headers["Set-Cookie"]=@cookies.map{|k,v| "##{k}=#{"#{v}.escape(%/)}"; path="#self/'/'
  }"if v!=@k[k]-[nil];self end;def to_s:"Status: #{$status}#{Z+@headers.map{|k,v|
    [*v].map{|x| "#k:#{x}"}*Z+Z*2+@body}"end;end

```

not sure, if this is  
a bug or a feature

```

X=module Controllers;@r=[];class<<self;def r;@r;end;def R*u;r=@r;Class.new{
  meta_def(urls){u};meta_def(:inherited){|x|r<<x}}end;def M;def M;end;constants.map{|c|
  k=const_get(c);k.send(:include,C::Base::Models);[0,0]=k if !r.include?k;k.meta_def(
  :url{|x| x.downcase}if !k.respond_to?(:url)end;def D p;r.map{|k|k.urls.
  map{|x|x.respond_to?(:url)?x.url:[]};[NotFound,[p]]end end;class
  NotFound<R;def get p;r(Mab.new{h1 P;h2"#k.#m";h3"#e.class
  }#{e.message}";ul{e.backtrace.each{|bt|li(bt)}}).to_s)end end;self;end;class<<
self;def goes m;eval S.gsub(/Camping/,m.to_s).gsub("A\pps=[]","Cam\ping::Apps<<\
self"),TOPLEVEL_BINDING;end;def escape s;s.to_s.gsub(/[\^ \w.-]+/n){'%'+($&.
  unpack('H2*$&.size')*%').upcase}.tr(' ','+')end;def un s;s.tr(' ','').gsub(
  /%([\da-f]{2})/in){[$1].pack('H*')}end;def qs_parse q,d='&';m=proc{|_,o,n|o.u(
  n,&m)rescue([*o]<<n)};q.to_s.split(/[#{d}]/n).inject(H[]){|h,p|k,v=un(p).
  split('=',2);h.u k.split(/[\]\[\]/n).reverse.inject(v){|x,i|H[i,x]},&m}end;def
  kp s;c=qs_parse(s,',' )end;def run r=$stdin,e=ENV;X.M;k,a=X.D un('/#{e[
  'PATH_INFO']}'.gsub(/\//+/, '/'));k.new(r,e,(m=e['REQUEST_METHOD']||'GET')).Y.
  service *a;rescue Exception=>x;X::ServerError.new(r,e,'get').service(k,m,x)end
def method_missing m,c,*a;X.M;k=X.const_get(c).new(StringIO.new,H['HTTP_HOST',
  ],'SCRIPT_NAME','HTTP_COOKIE',),m.to_s);H.new(a.pop).each{|e,f|k.send(
  "#e=",f)}if Hash===a[-1];k.service *a;end;end;module Views;include X,Helpers
end;module Models;autoload:Base,'camping/db';def Y;self;end;end;class
  Mab<Markaby::Builder
include Views;def tag!*g,&b;h=g[-1];[:href,:action,:src].map{|a|(h[a]=self/h[a])rescue
  0};super end end;H=HashWithIndifferentAccess;class H;def method_missing m,*a
  m.to_s=~/$/?self[$]=a[0]:a==[]?self[m]:raise(NoMethodError,"#{m}")end
  alias_method:u,:regular_update;end end

```



# The Basics



# Core Components

- ActiveRecord for Models
- HTTP-style for Controllers
- Markaby for Views



# Models

- Everybody in here knows ActiveRecord
- One-Way-Migrations
- SQLite-db is default storage
- All Camping apps use the same one



# Controllers

Camling.goes :HelloWorld

```
module HelloWorld::Controllers
  class Index < R '/'
    def get
      'Hello, World!'
    end
  end
end
```



# Controllers (contd.)

- One method for each verb - get, post, put, ...
- Return value of method is send to browser
- Configure Routing with R method



# Routing

- R `' / '`  
localhost:3301 /
- R `' /foo '`  
localhost:3301 / foo
- R `' /bar / ( \d+ ) '`  
localhost:3301 / bar / 110
- R `' /baz ', ' /baz / ( \d+ ) '`  
localhost:3301 / baz  
localhost:3301 / baz / 110



# Routing (contd.)

```
class Baz < R '/baz', '/baz/(\d+)'  
  def get(id = nil)  
    id.nil? ? index : show(id)  
  end  
  
  ...  
end
```



# Views

- Markaby - Markup in Ruby
- Internal DSL for HTML



# Views

- Markaby - Markup in Ruby

```
html do
```

- Internal DSL for HTML

```
  head do
```

```
    title { "The test site" }
```

```
  end
```

```
  body.example! do
```

```
    h1 { "The test site" }
```

```
    div.content do
```

```
      p "show example"
```

```
      img :src => "camping.png"
```

```
    end
```

```
  end
```

```
end
```



# Views

- Markaby - Markup in Ruby

```
html do
```

- Internal DSL for HTML

```
  head do
```

```
    title { "The test site" }
```

```
  end
```

id

```
  body.example! do
```

```
    h1 { "The test site" }
```

```
    div.content do
```

```
      p "show example"
```

```
      img :src => "camping.png"
```

```
    end
```

```
  end
```

```
end
```



# Views

- Markaby - Markup in Ruby

```
html do
```

- Internal DSL for HTML

```
  head do
```

```
    title { "The test site" }
```

```
  end
```

id

```
  body.example! do
```

```
    h1 { "The test site" }
```

class

```
    div.content do
```

```
      p "show example"
```

```
      img :src => "camping.png"
```

```
    end
```

```
  end
```

```
end
```



# Views

- Markaby - Markup in Ruby

```
html do
```

- Internal DSL for HTML

```
  head do
```

```
    title { "The test site" }
```

```
  end
```

id

```
  body.example! do
```

```
    h1 { "The test site" }
```

class

```
    div.content do
```

```
      p "show example"
```

attributes

```
      img :src => "camping.png"
```

```
    end
```

```
  end
```

```
end
```



A whole App



# A Camping App

One File



# A Camping App

One File

models

migrations

controllers

views

post-ambles



```
Camping.goes :Foo
```

```
module Foo::Models
  class User < Base
    has_many :bars
  end
  class Bar < Base
    belongs_to :user
  end

  class CreateFoo < V 1.0
    def self.up
      create_table :foo_users do
        ...
      end
      create_table :foo_bars do
        ...
      end
    end
    def self.down
      drop_table :foo_users
      drop_table :foo_bars
    end
  end
end
```

```
module Foo::Controllers
  class Index < R '/'
    def get
      @user_count = User.count
      @foo_count = Foo.count
      render 'index'
    end
  end
end

module Foo::Views
  def layout
    xhtml_strict do
      head do
        title "Foo - a Camping App"
      end
      body do
        self << yield
      end
    end
  end

  def index
    p do
      text "#{@user_count} users "
      text " with#{@bar_count} bars"
    end
  end
end
```



# camping foo.rb

- Put everything in one file
- `run camping foo.rb`
- point your browser to <http://localhost:3301/>



Extending



# service(\*a)

- One single point of entry
- Each request passes this method
  - May manipulate arguments
  - Add behaviour
  - ...
- Two possible uses



# I. Write your own extensions

```
module MyExtension
  def service(*a)
    # do the additional stuff
    super
    # but be sure to call super
    # and keep return values
  end
end

module Foo
  include MyExtension
end
```



## II. Use existing ones

- Camping::Session - adds session support (db)
- SleepingBag - allows rails-like REST-controllers
- TentSteak - some Markaby goodness
- Parasite - include Camping in Rails apps
- Mosquito - Tests for Camping



# Finding libraries

They all have  
generic,  
camping-associated  
names !!!

<http://rubyforge.org/pipermail/camping-list/>



What else?



# Usage scenarios

- small apps
  - Do I really need a full blown rails app for this little code?
- multiple small apps
  - camping servers mount multiple apps at once
  - they share the db so they could share models



# Future

- \_why does hackety.org
- still involved in camping dev
- last release 1.5 from 2006-10-03
- current dev for 1.6
  - bug fix and consolidation release
  - get rid of dependencies



# Future

- \_why does hackety.org
- still involved in camping dev
- last release 1.5 from 2006-10-03
- current dev for 1.6
  - bug fix and consolidation release
  - get rid of dependencies

soon !?!



Preview

ContextWiki

or Live Example

SyntaxCam



???