NameError at /isbn_number

undefined local variable or method `parms' for # <Sinatra::Application:0x007fc4480b2ef8>

- **file:** `app.rb`
- **location:** `block in <main>`
- **line:** 13

BACKTRACE

(expand)

JUMP TO: GET POST COOKIES ENV

- `app.rb` in **`block in <main>`**
- ```
  6.
  7. get '/' do
  8. erb :isbn_input # load ERB (HTML) file from /views directory
  9. end
  10.
  11. post '/isbn_number' do
  12. #num = params[:ISBN]
  13. is_too_small?(parms[:ISBN])
  14. end
  ```
- `/Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb` in **`call`**
- ```
  1604. method_name = "#{verb} #{path}"
  1605. unbound_method = generate_method(method_name, &block)
  1606. pattern, keys = compile path
  1607. conditions, @conditions = @conditions, []
  1608.
  1609. wrapper = block.arity != 0 ?
  1610. proc { |a,p| unbound_method.bind(a).call(*p) } :
  1611. proc { |a,p| unbound_method.bind(a).call }
  1612. wrapper.instance_variable_set(:@route_name, method_name)
  1613.
  1614. [ pattern, keys, conditions, wrapper ]
  1615. end
  1616.
  1617. def compile(path)
  1618. if path.respond_to? :to_str
  ```
- `/Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb` in **`block in compile!`**
- ```
  1604. method_name = "#{verb} #{path}"
  1605. unbound_method = generate_method(method_name, &block)
  1606. pattern, keys = compile path
  1607. conditions, @conditions = @conditions, []
  ```
  ```

```
1608.
1609. wrapper = block.arity != 0 ?
1610. proc { |a,p| unbound_method.bind(a).call(*p) } :
1611. proc { |a,p| unbound_method.bind(a).call }
1612. wrapper.instance_variable_set(:@route_name, method_name)
1613.
1614. [ pattern, keys, conditions, wrapper ]
1615. end
1616.
1617. def compile(path)
1618. if path.respond_to? :to_str
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **[]**

```
968.
969. # Run routes defined on the class and all superclasses.
970. def route!(base = settings, pass_block = nil)
971. if routes = base.routes[@request.request_method]
972. routes.each do |pattern, keys, conditions, block|
973. returned_pass_block = process_route(pattern, keys, conditions) do |*args|
974. env['sinatra.route'] = block.instance_variable_get(:@route_name)
975. route_eval { block[*args] }
976. end
977.
978. # don't wipe out pass_block in superclass
979. pass_block = returned_pass_block if returned_pass_block
980. end
981. end
982.
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block (3 levels) in route!**

```
968.
969. # Run routes defined on the class and all superclasses.
970. def route!(base = settings, pass_block = nil)
971. if routes = base.routes[@request.request_method]
972. routes.each do |pattern, keys, conditions, block|
973. returned_pass_block = process_route(pattern, keys, conditions) do |*args|
974. env['sinatra.route'] = block.instance_variable_get(:@route_name)
975. route_eval { block[*args] }
976. end
977.
978. # don't wipe out pass_block in superclass
979. pass_block = returned_pass_block if returned_pass_block
980. end
981. end
982.
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **route_eval**

```
987.
988. route_eval(&pass_block) if pass_block
989. route_missing
990. end
991.
992. # Run a route block and throw :halt with the result.
993. def route_eval
994. throw :halt, yield
995. end
996.
997. # If the current request matches pattern and conditions, fill params
998. # with keys and call the given block.
999. # Revert params afterwards.
1000. #
1001. # Returns pass block.
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block (2 levels) in route!**

```
968.
969. # Run routes defined on the class and all superclasses.
970. def route!(base = settings, pass_block = nil)
971. if routes = base.routes[@request.request_method]
972. routes.each do |pattern, keys, conditions, block|
973. returned_pass_block = process_route(pattern, keys, conditions) do |*args|
974. env['sinatra.route'] = block.instance_variable_get(:@route_name)
975. route_eval { block[*args] }
976. end
977.
978. # don't wipe out pass_block in superclass
979. pass_block = returned_pass_block if returned_pass_block
980. end
981. end
982.
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block in process_route**

```
1008. if values.any?
1009. original, @params = params, params.merge('splat' => [], 'captures' => values)
1010. keys.zip(values) { |k,v| Array === @params[k] ? @params[k] << v : @params[k] = v if v }
1011. end
1012.
1013. catch(:pass) do
1014. conditions.each { |c| throw :pass if c.bind(self).call == false }
1015. block ? block[self, values] : yield(self, values)
1016. end
1017. ensure
1018. @params = original if original
1019. end
1020.
1021. # No matching route was found or all routes passed. The default
1022. # implementation is to forward the request downstream when running
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **catch**

```
1006. values += match.captures.map! { |v| force_encoding URI_INSTANCE.unescape(v) if v }
1007.
1008. if values.any?
1009. original, @params = params, params.merge('splat' => [], 'captures' => values)
1010. keys.zip(values) { |k,v| Array === @params[k] ? @params[k] << v : @params[k] = v if v }
1011. end
1012.
1013. catch(:pass) do
1014. conditions.each { |c| throw :pass if c.bind(self).call == false }
1015. block ? block[self, values] : yield(self, values)
1016. end
1017. ensure
1018. @params = original if original
1019. end
1020.
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **process_route**

```
1006. values += match.captures.map! { |v| force_encoding URI_INSTANCE.unescape(v) if v }
1007.
1008. if values.any?
1009. original, @params = params, params.merge('splat' => [], 'captures' => values)
1010. keys.zip(values) { |k,v| Array === @params[k] ? @params[k] << v : @params[k] = v if v }
1011. end
1012.
1013. catch(:pass) do
1014. conditions.each { |c| throw :pass if c.bind(self).call == false }
1015. block ? block[self, values] : yield(self, values)
1016. end
1017. ensure
```

```
1018. @params = original if original
1019. end
1020.
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block in route!**

```
966. base.filters[type].each { |args| process_route(*args) }
967. end
968.
969. # Run routes defined on the class and all superclasses.
970. def route!(base = settings, pass_block = nil)
971. if routes = base.routes[@request.request_method]
972. routes.each do |pattern, keys, conditions, block|
973. returned_pass_block = process_route(pattern, keys, conditions) do |*args|
974. env['sinatra.route'] = block.instance_variable_get(:@route_name)
975. route_eval { block[*args] }
976. end
977.
978. # don't wipe out pass_block in superclass
979. pass_block = returned_pass_block if returned_pass_block
980. end
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **each**

```
965. filter! type, base.superclass if base.superclass.respond_to?(:filters)
966. base.filters[type].each { |args| process_route(*args) }
967. end
968.
969. # Run routes defined on the class and all superclasses.
970. def route!(base = settings, pass_block = nil)
971. if routes = base.routes[@request.request_method]
972. routes.each do |pattern, keys, conditions, block|
973. returned_pass_block = process_route(pattern, keys, conditions) do |*args|
974. env['sinatra.route'] = block.instance_variable_get(:@route_name)
975. route_eval { block[*args] }
976. end
977.
978. # don't wipe out pass_block in superclass
979. pass_block = returned_pass_block if returned_pass_block
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **route!**

```
965. filter! type, base.superclass if base.superclass.respond_to?(:filters)
966. base.filters[type].each { |args| process_route(*args) }
967. end
968.
969. # Run routes defined on the class and all superclasses.
970. def route!(base = settings, pass_block = nil)
971. if routes = base.routes[@request.request_method]
972. routes.each do |pattern, keys, conditions, block|
973. returned_pass_block = process_route(pattern, keys, conditions) do |*args|
974. env['sinatra.route'] = block.instance_variable_get(:@route_name)
975. route_eval { block[*args] }
976. end
977.
978. # don't wipe out pass_block in superclass
979. pass_block = returned_pass_block if returned_pass_block
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block in dispatch!**

```
1078. end
1079.
1080. # Dispatch a request with error handling.
1081. def dispatch!
1082. invoke do
1083. static! if settings.static? && (request.get? || request.head?)
1084. filter! :before
1085. route!
```

```
1086. end
1087. rescue ::Exception => boom
1088. invoke { handle_exception!(boom) }
1089. ensure
1090. begin
1091. filter! :after unless env['sinatra.static_file']
1092. rescue ::Exception => boom
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block in invoke**

```
1060. # Creates a Hash with indifferent access.
1061. def indifferent_hash
1062. Hash.new {|hash,key| hash[key.to_s] if Symbol === key }
1063. end
1064.
1065. # Run the block with 'throw :halt' support and apply result to the response.
1066. def invoke
1067. res = catch(:halt) { yield }
1068. res = [res] if Integer === res or String === res
1069. if Array === res and Integer === res.first
1070. res = res.dup
1071. status(res.shift)
1072. body(res.pop)
1073. headers(*res)
1074. elsif res.respond_to? :each
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **catch**

```
1060. # Creates a Hash with indifferent access.
1061. def indifferent_hash
1062. Hash.new {|hash,key| hash[key.to_s] if Symbol === key }
1063. end
1064.
1065. # Run the block with 'throw :halt' support and apply result to the response.
1066. def invoke
1067. res = catch(:halt) { yield }
1068. res = [res] if Integer === res or String === res
1069. if Array === res and Integer === res.first
1070. res = res.dup
1071. status(res.shift)
1072. body(res.pop)
1073. headers(*res)
1074. elsif res.respond_to? :each
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **invoke**

```
1060. # Creates a Hash with indifferent access.
1061. def indifferent_hash
1062. Hash.new {|hash,key| hash[key.to_s] if Symbol === key }
1063. end
1064.
1065. # Run the block with 'throw :halt' support and apply result to the response.
1066. def invoke
1067. res = catch(:halt) { yield }
1068. res = [res] if Integer === res or String === res
1069. if Array === res and Integer === res.first
1070. res = res.dup
1071. status(res.shift)
1072. body(res.pop)
1073. headers(*res)
1074. elsif res.respond_to? :each
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **dispatch!**

```
1075. body res
1076. end
1077. nil # avoid double setting the same response tuple twice
1078. end
```

```
1079.
1080. # Dispatch a request with error handling.
1081. def dispatch!
1082. invoke do
1083. static! if settings.static? && (request.get? || request.head?)
1084. filter! :before
1085. route!
1086. end
1087. rescue ::Exception => boom
1088. invoke { handle_exception!(boom) }
1089. ensure
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block in call!**

```
900. @request = Request.new(env)
901. @response = Response.new
902. @params = indifferent_params(@request.params)
903. template_cache.clear if settings.reload_templates
904. force_encoding(@params)
905.
906. @response['Content-Type'] = nil
907. invoke { dispatch! }
908. invoke { error_block!(response.status) } unless @env['sinatra.error']
909.
910. unless @response['Content-Type']
911. if Array === body and body[0].respond_to? :content_type
912. content_type body[0].content_type
913. else
914. content_type :html
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block in invoke**

```
1060. # Creates a Hash with indifferent access.
1061. def indifferent_hash
1062. Hash.new {|hash,key| hash[key.to_s] if Symbol === key }
1063. end
1064.
1065. # Run the block with 'throw :halt' support and apply result to the response.
1066. def invoke
1067. res = catch(:halt) { yield }
1068. res = [res] if Integer === res or String === res
1069. if Array === res and Integer === res.first
1070. res = res.dup
1071. status(res.shift)
1072. body(res.pop)
1073. headers(*res)
1074. elsif res.respond_to? :each
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **catch**

```
1060. # Creates a Hash with indifferent access.
1061. def indifferent_hash
1062. Hash.new {|hash,key| hash[key.to_s] if Symbol === key }
1063. end
1064.
1065. # Run the block with 'throw :halt' support and apply result to the response.
1066. def invoke
1067. res = catch(:halt) { yield }
1068. res = [res] if Integer === res or String === res
1069. if Array === res and Integer === res.first
1070. res = res.dup
1071. status(res.shift)
1072. body(res.pop)
1073. headers(*res)
1074. elsif res.respond_to? :each
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **invoke**

- 1060. # Creates a Hash with indifferent access.
  1061. def indifferent_hash
  1062. Hash.new {|hash,key| hash[key.to_s] if Symbol === key }
  1063. end
  1064.
  1065. # Run the block with 'throw :halt' support and apply result to the response.
  1066. def invoke
  1067. res = catch(:halt) { yield }
  1068. res = [res] if Integer === res or String === res
  1069. if Array === res and Integer === res.first
  1070. res = res.dup
  1071. status(res.shift)
  1072. body(res.pop)
  1073. headers(*res)
  1074. elsif res.respond_to? :each
- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **call!**
- 900. @request = Request.new(env)
  901. @response = Response.new
  902. @params = indifferent_params(@request.params)
  903. template_cache.clear if settings.reload_templates
  904. force_encoding(@params)
  905.
  906. @response['Content-Type'] = nil
  907. invoke { dispatch! }
  908. invoke { error_block!(response.status) } unless @env['sinatra.error']
  909.
  910. unless @response['Content-Type']
  911. if Array === body and body[0].respond_to? :content_type
  912. content_type body[0].content_type
  913. else
  914. content_type :html
- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **call**
- 888. @app = app
  889. @template_cache = Tilt::Cache.new
  890. yield self if block_given?
  891. end
  892.
  893. # Rack call interface.
  894. def call(env)
  895. dup.call!(env)
  896. end
  897.
  898. def call!(env) # :nodoc:
  899. @env = env
  900. @request = Request.new(env)
  901. @response = Response.new
  902. @params = indifferent_params(@request.params)
- /Library/Ruby/Gems/2.0.0/gems/rack-protection-1.5.3/lib/rack/protection/xss_header.rb in **call**
- 11. #
  12. # Options:
  13. # xss_mode:: How the browser should prevent the attack (default: :block)
  14. class XSSHeader < Base
  15. default_options :xss_mode => :block, :nosniff => true
  16.
  17. def call(env)
  18. status, headers, body = @app.call(env)
  19. headers['X-XSS-Protection'] ||= "1; mode=#{options[:xss_mode]}" if html? headers
  20. headers['X-Content-Type-Options'] ||= 'nosniff' if options[:nosniff]
  21. [status, headers, body]
  22. end

```
23. end
24. end
25. end
```

- /Library/Ruby/Gems/2.0.0/gems/rack-protection-1.5.3/lib/rack/protection/path_traversal.rb in **call**

```
9. #
10. # Unescapes '/' and '.', expands +path_info+.
11. # Thus <tt>GET /foo/%2e%2e%2fbar</tt> becomes <tt>GET /bar</tt>.
12. class PathTraversal < Base
13. def call(env)
14. path_was = env["PATH_INFO"]
15. env["PATH_INFO"] = cleanup path_was if path_was && !path_was.empty?
16. app.call env
17. ensure
18. env["PATH_INFO"] = path_was
19. end
20.
21. def cleanup(path)
22. if path.respond_to?(:encoding)
23. # Ruby 1.9+ M17N
```

- /Library/Ruby/Gems/2.0.0/gems/rack-protection-1.5.3/lib/rack/protection/json_csrf.rb in **call**

```
11. # Array prototype has been patched to track data. Checks the referrer
12. # even on GET requests if the content type is JSON.
13. class JsonCsrf < Base
14. alias react deny
15.
16. def call(env)
17. request = Request.new(env)
18. status, headers, body = app.call(env)
19.
20. if has_vector? request, headers
21. warn env, "attack prevented by #{self.class}"
22. react(env) or [status, headers, body]
23. else
24. [status, headers, body]
25. end
```

- /Library/Ruby/Gems/2.0.0/gems/rack-protection-1.5.3/lib/rack/protection/base.rb in **call**

```
42. end
43.
44. def call(env)
45. unless accepts? env
46. instrument env
47. result = react env
48. end
49. result or app.call(env)
50. end
51.
52. def react(env)
53. result = send(options[:reaction], env)
54. result if Array === result and result.size == 3
55. end
56.
```

- /Library/Ruby/Gems/2.0.0/gems/rack-protection-1.5.3/lib/rack/protection/base.rb in **call**

```
42. end
43.
44. def call(env)
45. unless accepts? env
46. instrument env
47. result = react env
48. end
49. result or app.call(env)
```

```
50. end
51.
52. def react(env)
53. result = send(options[:reaction], env)
54. result if Array === result and result.size == 3
55. end
56.
```

- /Library/Ruby/Gems/2.0.0/gems/rack-protection-1.5.3/lib/rack/protection/frame_options.rb in **call**

```
24. frame_options = options[:frame_options]
25. frame_options = options[:frame_options].to_s.upcase unless frame_options.respond_to? :to_str
26. frame_options.to_str
27. end
28. end
29.
30. def call(env)
31. status, headers, body = @app.call(env)
32. headers['X-Frame-Options'] ||= frame_options if html? headers
33. [status, headers, body]
34. end
35. end
36. end
37. end
```

- /Library/Ruby/Gems/2.0.0/gems/rack-1.6.5/lib/rack/logger.rb in **call**

```
8. end
9.
10. def call(env)
11. logger = ::Logger.new(env['rack.errors'])
12. logger.level = @level
13.
14. env['rack.logger'] = logger
15. @app.call(env)
16. end
17. end
18. end
```

- /Library/Ruby/Gems/2.0.0/gems/rack-1.6.5/lib/rack/commonlogger.rb in **call**

```
26. def initialize(app, logger=nil)
27. @app = app
28. @logger = logger
29. end
30.
31. def call(env)
32. began_at = Time.now
33. status, header, body = @app.call(env)
34. header = Utils::HeaderHash.new(header)
35. body = BodyProxy.new(body) { log(env, status, header, began_at) }
36. [status, header, body]
37. end
38.
39. private
40.
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **call**

```
212. env['sinatra.commonlogger'] ? @app.call(env) : super
213. end
214.
215. superclass.class_eval do
216. alias call_without_check call unless method_defined? :call_without_check
217. def call(env)
218. env['sinatra.commonlogger'] = true
219. call_without_check(env)
220. end
```

```
221. end
222. end
223.
224. class NotFound < NameError #:nodoc:
225. def http_status; 404 end
226. end
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **call**

```
205. end
206. end
207.
208. # Behaves exactly like Rack::CommonLogger with the notable exception that it does nothing,
209. # if another CommonLogger is already in the middleware chain.
210. class CommonLogger < Rack::CommonLogger
211. def call(env)
212. env['sinatra.commonlogger'] ? @app.call(env) : super
213. end
214.
215. superclass.class_eval do
216. alias call_without_check call unless method_defined? :call_without_check
217. def call(env)
218. env['sinatra.commonlogger'] = true
219. call_without_check(env)
```

- /Library/Ruby/Gems/2.0.0/gems/rack-1.6.5/lib/rack/head.rb in **call**

```
6. # Rack::Head returns an empty body for all HEAD requests. It leaves
7. # all other requests unchanged.
8. def initialize(app)
9. @app = app
10. end
11.
12. def call(env)
13. status, headers, body = @app.call(env)
14.
15. if env[REQUEST_METHOD] == HEAD
16. [
17. status, headers, Rack::BodyProxy.new([]) do
18. body.close if body.respond_to? :close
19. end
20. ]
```

- /Library/Ruby/Gems/2.0.0/gems/rack-1.6.5/lib/rack/methodoverride.rb in **call**

```
15. method = method_override(env)
16. if HTTP_METHODS.include?(method)
17. env["rack.methodoverride.original_method"] = env[REQUEST_METHOD]
18. env[REQUEST_METHOD] = method
19. end
20. end
21.
22. @app.call(env)
23. end
24.
25. def method_override(env)
26. req = Request.new(env)
27. method = method_override_param(req) ||
28. env[HTTP_METHOD_OVERRIDE_HEADER]
29. method.to_s.upcase
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/show_exceptions.rb in **call**

```
18.
19. def initialize(app)
20. @app = app
21. @template = ERB.new(TEMPLATE)
22. end
```

```
23.
24. def call(env)
25. @app.call(env)
26. rescue Exception => e
27. errors, env["rack.errors"] = env["rack.errors"], @@eats_errors
28.
29. if prefers_plain_text?(env)
30. content_type = "text/plain"
31. exception = dump_exception(e)
32. else
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **call**

```
175. # Some Rack handlers (Thin, Rainbows!) implement an extended body object protocol, however,
176. # some middleware (namely Rack::Lint) will break it by not mirroring the methods in
     question.
177. # This middleware will detect an extended body object and will make sure it reaches the
178. # handler directly. We do this here, so our middleware and middleware set up by the app will
179. # still be able to run.
180. class ExtendedRack < Struct.new(:app)
181. def call(env)
182. result, callback = app.call(env), env['async.callback']
183. return result unless callback and async?(*result)
184. after_response { callback.call result }
185. setup_close(env, *result)
186. throw :async
187. end
188.
189. private
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **call**

```
2006. end
2007.
2008. def helpers
2009. @instance
2010. end
2011.
2012. def call(env)
2013. @stack.call(env)
2014. end
2015.
2016. def inspect
2017. "#<#{@instance.class} app_file=#{settings.app_file.inspect}>"
2018. end
2019. end
2020.
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **block in call**

```
1480. setup_default_middleware builder
1481. setup_middleware builder
1482. builder.run app
1483. builder
1484. end
1485.
1486. def call(env)
1487. synchronize { prototype.call(env) }
1488. end
1489.
1490. # Like Kernel#caller but excluding certain magic entries and without
1491. # line / method information; the resulting array contains filenames only.
1492. def caller_files
1493. cleaned_caller(1).flatten
1494. end
```

- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **synchronize**

- 1780. end
  1781.
  1782. @@mutex = Mutex.new
  1783. def synchronize(&block)
  1784. if lock?
  1785. @@mutex.synchronize(&block)
  1786. else
  1787. yield
  1788. end
  1789. end
  1790.
  1791. # used for deprecation warnings
  1792. def warn(message)
  1793. super message + "\n\tfrom #{cleaned_caller.first.join(':')}"
  1794. end
- /Library/Ruby/Gems/2.0.0/gems/sinatra-1.4.8/lib/sinatra/base.rb in **call**
- 1480. setup_default_middleware builder
  1481. setup_middleware builder
  1482. builder.run app
  1483. builder
  1484. end
  1485.
  1486. def call(env)
  1487. synchronize { prototype.call(env) }
  1488. end
  1489.
  1490. # Like Kernel#caller but excluding certain magic entries and without
  1491. # line / method information; the resulting array contains filenames only.
  1492. def caller_files
  1493. cleaned_caller(1).flatten
  1494. end
- /Library/Ruby/Gems/2.0.0/gems/rack-1.6.5/lib/rack/handler/webrick.rb in **service**
- 81. env[QUERY_STRING] ||= ""
  82. unless env[PATH_INFO] == ""
  83. path, n = req.request_uri.path, env["SCRIPT_NAME"].length
  84. env[PATH_INFO] = path[n, path.length-n]
  85. end
  86. env["REQUEST_PATH"] ||= [env["SCRIPT_NAME"], env[PATH_INFO]].join
  87.
  88. status, headers, body = @app.call(env)
  89. begin
  90. res.status = status.to_i
  91. headers.each { |k, vs|
  92. next if k.downcase == "rack.hijack"
  93.
  94. if k.downcase == "set-cookie"
  95. res.cookies.concat vs.split("\n")
- /System/Library/Frameworks/Ruby.framework/Versions/2.0/usr/lib/ruby/2.0.0/webrick/httpserver.rb in **service**
- 131.
  132. servlet, options, script_name, path_info = search_servlet(req.path)
  133. raise HTTPStatus::NotFound, "`#{req.path}' not found." unless servlet
  134. req.script_name = script_name
  135. req.path_info = path_info
  136. si = servlet.get_instance(self, *options)
  137. @logger.debug(format("%s is invoked.", si.class.name))
  138. si.service(req, res)
  139. end
  140.
  141. ##

```
142. # The default OPTIONS request handler says GET, HEAD, POST and OPTIONS
143. # requests are allowed.
144.
145. def do_OPTIONS(req, res)
```

- /System/Library/Frameworks/Ruby.framework/Versions/2.0/usr/lib/ruby/2.0.0/webrick/httpserver.rb in **run**

```
87. if callback = server[:RequestCallback]
88. callback.call(req, res)
89. elsif callback = server[:RequestHandler]
90. msg = ":RequestHandler is deprecated, please use :RequestCallback"
91. @logger.warn(msg)
92. callback.call(req, res)
93. end
94. server.service(req, res)
95. rescue HTTPStatus::EOFError, HTTPStatus::RequestTimeout => ex
96. res.set_error(ex)
97. rescue HTTPStatus::Error => ex
98. @logger.error(ex.message)
99. res.set_error(ex)
100. rescue HTTPStatus::Status => ex
101. res.status = ex.code
```

- /System/Library/Frameworks/Ruby.framework/Versions/2.0/usr/lib/ruby/2.0.0/webrick/server.rb in **block in start_thread**

```
288. addr = sock.peeraddr
289. @logger.debug "accept: #{addr[3]}:#{addr[1]}"
290. rescue SocketError
291. @logger.debug "accept: <address unknown>"
292. raise
293. end
294. call_callback(:AcceptCallback, sock)
295. block ? block.call(sock) : run(sock)
296. rescue Errno::ENOTCONN
297. @logger.debug "Errno::ENOTCONN raised"
298. rescue ServerError => ex
299. msg = "#{ex.class}: #{ex.message}\n\t#{ex.backtrace[0]}"
300. @logger.error msg
301. rescue Exception => ex
302. @logger.error ex
```

GET

No GET data.

POST

Variable	Value
ISBN	"234059872345"

COOKIES

No cookie data.

Rack ENV

Variable	Value
CONTENT_LENGTH	17
CONTENT_TYPE	application/x-www-form-urlencoded

GATEWAY_INTERFACE	CGI/1.1
HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
HTTP_ACCEPT_ENCODING	gzip, deflate, br
HTTP_ACCEPT_LANGUAGE	en-US,en;q=0.8
HTTP_CACHE_CONTROL	max-age=0
HTTP_CONNECTION	keep-alive
HTTP_DNT	1
HTTP_HOST	localhost:4567
HTTP_ORIGIN	http://localhost:4567
HTTP_REFERER	http://localhost:4567/
HTTP_UPGRADE_INSECURE_REQUESTS	1
HTTP_USER_AGENT	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
HTTP_VERSION	HTTP/1.1
PATH_INFO	/isbn_number
QUERY_STRING	
REMOTE_ADDR	::1
REMOTE_HOST	localhost
REQUEST_METHOD	POST
REQUEST_PATH	/isbn_number
REQUEST_URI	http://localhost:4567/isbn_number
SCRIPT_NAME	
SERVER_NAME	localhost
SERVER_PORT	4567
SERVER_PROTOCOL	HTTP/1.1
SERVER_SOFTWARE	WEBrick/1.3.1 (Ruby/2.0.0/2015-12-16)
rack.errors	#<Object:0x007fc447b5c6e8>
rack.hijack	#<Proc:0x007fc4480aa348@/Library/Ruby/Gems/2.0.0/gems/rack-1.6.5/lib/rack/handler/webrick.rb:76 (lambda)>
rack.hijack?	true
rack.hijack_io	nil
rack.input	#<StringIO:0x007fc4480aa708>
rack.logger	#<Logger:0x007fc4480a8ed0 @progname=nil, @level=1, @default_formatter=#<Logger::Formatter:0x007fc4480a8ea8 @datetime_format=nil>, @formatter=nil, @logdev=#<Logger::LogDevice:0x007fc4480a8e58 @shift_size=nil, @shift_age=nil, @filename=nil, @dev=#<IO:<STDERR>>, @mutex=#<Logger::LogDevice::LogDeviceMutex:0x007fc4480a8e30 @mon_owner=nil, @mon_count=0, @mon_mutex=#<Mutex:0x007fc4480a8db8>>>>
rack.multiprocess	false
rack.multithread	true
rack.request.cookie_hash	{}
rack.request.form_hash	{"ISBN"=>"234059872345"}
rack.request.form_input	#<StringIO:0x007fc4480aa708>
rack.request.form_vars	ISBN=234059872345
rack.request.query_hash	{}
rack.request.query_string	
rack.run_once	false
rack.url_scheme	http
rack.version	[1, 3]

sinatra.accept	[#<Sinatra::Request::AcceptEntry:0x007fc4480b81c8 @entry="text/html", @type="text/html", @params={}, @q=1.0>, #<Sinatra::Request::AcceptEntry:0x007fc4480c3f00 @entry="application/xhtml+xml", @type="application/xhtml+xml", @params={}, @q=1.0>, #<Sinatra::Request::AcceptEntry:0x007fc4480c38e8 @entry="image/webp", @type="image/webp", @params={}, @q=1.0>, #<Sinatra::Request::AcceptEntry:0x007fc4480c3d48 @entry="application/xml;q=0.9", @type="application/xml", @params={}, @q=0.9>, #<Sinatra::Request::AcceptEntry:0x007fc4480c3758 @entry="*/*;q=0.8", @type="*/*", @params={}, @q=0.8>]
sinatra.commonlogger	true
sinatra.error	#<NameError: undefined local variable or method `parms' for #<Sinatra::Application:0x007fc4480b2ef8>>
sinatra.route	POST /isbn_number

You're seeing this error because you have enabled the `show_exceptions` setting.