

Files

cd08e6b

Go to file

> .devcontainer

> .github

> actioncable

> actionmailbox

> actionmailer

> actionpack

> bin

> lib

> abstract_controller

> action_controller

> action_dispatch

> http

> journey

> middleware

> session

> templates

actionable_exceptions.rb

callbacks.rb

cookies.rb

debug_exceptions.rb

debug_locks.rb

debug_view.rb

exception_wrapper.rb

executor.rb

flash.rb

host_authorization.rb

public_exceptions.rb

reloader.rb

remote_ip.rb

request_id.rb

server_timing.rb

show_exceptions.rb

ssl.rb

stack.rb

static.rb

> request

rails / actionpack / lib / action_dispatch / middleware / exception_wrapper.rb

alaz and flavorjones Skip logging backtrace when exception is i... dc2d948 · 3 years ago History

Code Blame 196 lines (162 loc) · 6.05 KB Raw Copy Download Toggle

```
1 # frozen_string_literal: true
2
3 require "active_support/core_ext/module/attribute_accessors"
4 require "rack/utils"
5
6 module ActionDispatch
7   class ExceptionWrapper
8     attr_accessor :rescue_responses, default: Hash.new(:internal_server_error).merge!(
9       "ActionController::RoutingError" => :not_found,
10      "AbstractController::ActionNotFound" => :not_found,
11      "ActionController::MethodNotAllowed" => :method_not_allowed,
12      "ActionController::UnknownHttpMethod" => :method_not_allowed,
13      "ActionController::NotImplemented" => :not_implemented,
14      "ActionController::UnknownFormat" => :not_acceptable,
15      "ActionDispatch::Http::MimeNegotiation::InvalidType" => :not_acceptable,
16      "ActionController::MissingExactTemplate" => :not_acceptable,
17      "ActionController::InvalidAuthenticityToken" => :unprocessable_entity,
18      "ActionController::InvalidCrossOriginRequest" => :unprocessable_entity,
19      "ActionDispatch::Http::Parameters::ParseError" => :bad_request,
20      "ActionController::BadRequest" => :bad_request,
21      "ActionController::ParameterMissing" => :bad_request,
22      "Rack::QueryParser::ParameterTypeError" => :bad_request,
23      "Rack::QueryParser::InvalidParameterError" => :bad_request
24    )
25
26     attr_accessor :rescue_templates, default: Hash.new("diagnostics").merge!(
27       "ActionView::MissingTemplate" => "missing_template",
28       "ActionController::RoutingError" => "routing_error",
29       "AbstractController::ActionNotFound" => "unknown_action",
30       "ActiveRecord::StatementInvalid" => "invalid_statement",
31       "ActionView::Template::Error" => "template_error",
32       "ActionController::MissingExactTemplate" => "missing_exact_template",
33     )
34
35     attr_accessor :wrapper_exceptions, default: [
36       "ActionView::Template::Error"
37     ]
38
39     attr_accessor :silent_exceptions, default: [
40       "ActionController::RoutingError",
41       "ActionDispatch::Http::MimeNegotiation::InvalidType"
42     ]
43
44     attr_reader :backtrace_cleaner, :exception, :wrapped_causes, :line_number, :file
45
46     def initialize(backtrace_cleaner, exception)
47       @backtrace_cleaner = backtrace_cleaner
48       @exception = exception
49       @exception_class_name = @exception.class.name
50       @wrapped_causes = wrapped_causes_for(exception, backtrace_cleaner)
51
52       expand_backtrace if exception.is_a?(SyntaxError) || exception.cause.is_a?(SyntaxError)
53     end
54
55     def unwrapped_exception
56       if wrapper_exceptions.include?(@exception_class_name)
57         exception.cause
58       else
59         exception
60       end
61     end
62
63     def wrapped_exceptions
64       @wrapped_exceptions ||= begin
65         wrapped_exceptions = []
66         current_exception = exception
67         while current_exception
68           wrapped_exceptions.unshift(current_exception)
69           current_exception = current_exception.cause
70         end
71         wrapped_exceptions
72       end
73     end
74
75     def wrapped_exception
76       wrapped_exceptions.first
77     end
78
79     def backtrace
80       @backtrace ||= begin
81         backtrace = []
82         current_exception = exception
83         while current_exception
84           backtrace.unshift(current_exception.backtrace)
85           current_exception = current_exception.cause
86         end
87         backtrace
88       end
89     end
90
91     def backtrace_locations
92       @backtrace_locations ||= begin
93         backtrace_locations = []
94         current_exception = exception
95         while current_exception
96           backtrace_locations.unshift(current_exception.backtrace_locations)
97           current_exception = current_exception.cause
98         end
99         backtrace_locations
100      end
101    end
102
103    def to_s
104      @to_s ||= begin
105        to_s = ""
106        current_exception = exception
107        while current_exception
108          to_s += "#{current_exception.class} #{current_exception.message}\n"
109          current_exception = current_exception.cause
110        end
111        to_s
112      end
113    end
114
115    def to_html
116      @to_html ||= begin
117        to_html = ""
118        current_exception = exception
119        while current_exception
120          to_html += "

#{current_exception.class} #{current_exception.message}

\n"
121          current_exception = current_exception.cause
122        end
123        to_html
124      end
125    end
126
127    def to_json
128      @to_json ||= begin
129        to_json = {}
130        current_exception = exception
131        while current_exception
132          to_json["exception"] = current_exception.to_s
133          current_exception = current_exception.cause
134        end
135        to_json
136      end
137    end
138
139    def to_yaml
140      @to_yaml ||= begin
141        to_yaml = {}
142        current_exception = exception
143        while current_exception
144          to_yaml["exception"] = current_exception.to_s
145          current_exception = current_exception.cause
146        end
147        to_yaml
148      end
149    end
150
151    def to_xml
152      @to_xml ||= begin
153        to_xml = {}
154        current_exception = exception
155        while current_exception
156          to_xml["exception"] = current_exception.to_s
157          current_exception = current_exception.cause
158        end
159        to_xml
160      end
161    end
162
163    def to_rails
164      @to_rails ||= begin
165        to_rails = {}
166        current_exception = exception
167        while current_exception
168          to_rails["exception"] = current_exception.to_s
169          current_exception = current_exception.cause
170        end
171        to_rails
172      end
173    end
174
175    def to_text
176      @to_text ||= begin
177        to_text = {}
178        current_exception = exception
179        while current_exception
180          to_text["exception"] = current_exception.to_s
181          current_exception = current_exception.cause
182        end
183        to_text
184      end
185    end
186
187    def to_html_safe
188      @to_html_safe ||= begin
189        to_html_safe = {}
190        current_exception = exception
191        while current_exception
192          to_html_safe["exception"] = current_exception.to_s
193          current_exception = current_exception.cause
194        end
195        to_html_safe
196      end
197    end
198
199    def to_json_safe
200      @to_json_safe ||= begin
201        to_json_safe = {}
202        current_exception = exception
203        while current_exception
204          to_json_safe["exception"] = current_exception.to_s
205          current_exception = current_exception.cause
206        end
207        to_json_safe
208      end
209    end
210
211    def to_yaml_safe
212      @to_yaml_safe ||= begin
213        to_yaml_safe = {}
214        current_exception = exception
215        while current_exception
216          to_yaml_safe["exception"] = current_exception.to_s
217          current_exception = current_exception.cause
218        end
219        to_yaml_safe
220      end
221    end
222
223    def to_xml_safe
224      @to_xml_safe ||= begin
225        to_xml_safe = {}
226        current_exception = exception
227        while current_exception
228          to_xml_safe["exception"] = current_exception.to_s
229          current_exception = current_exception.cause
230        end
231        to_xml_safe
232      end
233    end
234
235    def to_rails_safe
236      @to_rails_safe ||= begin
237        to_rails_safe = {}
238        current_exception = exception
239        while current_exception
240          to_rails_safe["exception"] = current_exception.to_s
241          current_exception = current_exception.cause
242        end
243        to_rails_safe
244      end
245    end
246
247    def to_text_safe
248      @to_text_safe ||= begin
249        to_text_safe = {}
250        current_exception = exception
251        while current_exception
252          to_text_safe["exception"] = current_exception.to_s
253          current_exception = current_exception.cause
254        end
255        to_text_safe
256      end
257    end
258
259    def to_html_safe_text
260      @to_html_safe_text ||= begin
261        to_html_safe_text = {}
262        current_exception = exception
263        while current_exception
264          to_html_safe_text["exception"] = current_exception.to_s
265          current_exception = current_exception.cause
266        end
267        to_html_safe_text
268      end
269    end
270
271    def to_json_safe_text
272      @to_json_safe_text ||= begin
273        to_json_safe_text = {}
274        current_exception = exception
275        while current_exception
276          to_json_safe_text["exception"] = current_exception.to_s
277          current_exception = current_exception.cause
278        end
279        to_json_safe_text
280      end
281    end
282
283    def to_yaml_safe_text
284      @to_yaml_safe_text ||= begin
285        to_yaml_safe_text = {}
286        current_exception = exception
287        while current_exception
288          to_yaml_safe_text["exception"] = current_exception.to_s
289          current_exception = current_exception.cause
290        end
291        to_yaml_safe_text
292      end
293    end
294
295    def to_xml_safe_text
296      @to_xml_safe_text ||= begin
297        to_xml_safe_text = {}
298        current_exception = exception
299        while current_exception
300          to_xml_safe_text["exception"] = current_exception.to_s
301          current_exception = current_exception.cause
302        end
303        to_xml_safe_text
304      end
305    end
306
307    def to_rails_safe_text
308      @to_rails_safe_text ||= begin
309        to_rails_safe_text = {}
310        current_exception = exception
311        while current_exception
312          to_rails_safe_text["exception"] = current_exception.to_s
313          current_exception = current_exception.cause
314        end
315        to_rails_safe_text
316      end
317    end
318
319    def to_text_safe_text
320      @to_text_safe_text ||= begin
321        to_text_safe_text = {}
322        current_exception = exception
323        while current_exception
324          to_text_safe_text["exception"] = current_exception.to_s
325          current_exception = current_exception.cause
326        end
327        to_text_safe_text
328      end
329    end
330
331    def to_html_safe_text_safe
332      @to_html_safe_text_safe ||= begin
333        to_html_safe_text_safe = {}
334        current_exception = exception
335        while current_exception
336          to_html_safe_text_safe["exception"] = current_exception.to_s
337          current_exception = current_exception.cause
338        end
339        to_html_safe_text_safe
340      end
341    end
342
343    def to_json_safe_text_safe
344      @to_json_safe_text_safe ||= begin
345        to_json_safe_text_safe = {}
346        current_exception = exception
347        while current_exception
348          to_json_safe_text_safe["exception"] = current_exception.to_s
349          current_exception = current_exception.cause
350        end
351        to_json_safe_text_safe
352      end
353    end
354
355    def to_yaml_safe_text_safe
356      @to_yaml_safe_text_safe ||= begin
357        to_yaml_safe_text_safe = {}
358        current_exception = exception
359        while current_exception
360          to_yaml_safe_text_safe["exception"] = current_exception.to_s
361          current_exception = current_exception.cause
362        end
363        to_yaml_safe_text_safe
364      end
365    end
366
367    def to_xml_safe_text_safe
368      @to_xml_safe_text_safe ||= begin
369        to_xml_safe_text_safe = {}
370        current_exception = exception
371        while current_exception
372          to_xml_safe_text_safe["exception"] = current_exception.to_s
373          current_exception = current_exception.cause
374        end
375        to_xml_safe_text_safe
376      end
377    end
378
379    def to_rails_safe_text_safe
380      @to_rails_safe_text_safe ||= begin
381        to_rails_safe_text_safe = {}
382        current_exception = exception
383        while current_exception
384          to_rails_safe_text_safe["exception"] = current_exception.to_s
385          current_exception = current_exception.cause
386        end
387        to_rails_safe_text_safe

```

- > routing
- > system_testing
- > testing
- journey.rb
- railtie.rb
- routing.rb

```
57         exception.cause
58     else
59         exception
60     end
61 end
62
63 def rescue_template
64     @@rescue_templates[@exception_class_name]
65 end
66
67 def status_code
68     self.class.status_code_for_exception(unwrapped_exception.class.name)
69 end
70
71 def exception_trace
72     trace = application_trace
73     trace = framework_trace if trace.empty? && !silent_exceptions.include?(@exception
74     trace
75 end
76
77 def application_trace
78     clean_backtrace(:silent)
79 end
80
81 def framework_trace
82     clean_backtrace(:noise)
83 end
84
85 def full_trace
86     clean_backtrace(:all)
87 end
88
89 def traces
90     application_trace_with_ids = []
91     framework_trace_with_ids = []
92     full_trace_with_ids = []
93
94     full_trace.each_with_index do |trace, idx|
95         trace_with_id = {
96             exception_object_id: @exception.object_id,
97             id: idx,
98             trace: trace
99         }
100
101         if application_trace.include?(trace)
102             application_trace_with_ids << trace_with_id
103         else
104             framework_trace_with_ids << trace_with_id
105         end
106
107         full_trace_with_ids << trace_with_id
108     end
109
110     {
111         "Application Trace" => application_trace_with_ids,
112         "Framework Trace" => framework_trace_with_ids,
113         "Full Trace" => full_trace_with_ids
114     }
115 end
116
117 def self.status_code_for_exception(class_name)
118     Rack::Utils.status_code(@@rescue_responses[class_name])
119
120
121
122 end
123
124
125 def source_extracts
126     backtrace.map do |trace|
127         file, line_number = extract_file_and_line_number(trace)
128
129         {
130             code: source_fragment(file, line_number),
131             line_number: line_number
132         }
133     end
134 end
```

```
132         }
133     end
134 end
135
136   def trace_to_show
137     if traces["Application Trace"].empty? && rescue_template != "routing_error"
138       "Full Trace"
139     else
140       "Application Trace"
141     end
142 end
143
144   def source_to_show_id
145     (traces[trace_to_show].first || {})[:id]
146 end
147
148   private
149   def backtrace
150     Array(@exception.backtrace)
151   end
152
153   def causes_for(exception)
154     return enum_for(__method__, exception) unless block_given?
155
156     yield exception while exception = exception.cause
157   end
158
159   def wrapped_causes_for(exception, backtrace_cleaner)
160     causes_for(exception).map { |cause| self.class.new(backtrace_cleaner, cause) }
161   end
162
163   def clean_backtrace(*args)
164     if backtrace_cleaner
165       backtrace_cleaner.clean(backtrace, *args)
166     else
167       backtrace
168     end
169   end
170
171   def source_fragment(path, line)
172     return unless Rails.respond_to?(:root) && Rails.root
173     full_path = Rails.root.join(path)
174     if File.exist?(full_path)
175       File.open(full_path, "r") do |file|
176         start = [line - 3, 0].max
177         lines = file.each_line.drop(start).take(6)
178         Hash[* (start + 1..(lines.count + start)).zip(lines).flatten]
179       end
180     end
181   end
182
183   def extract_file_and_line_number(trace)
184     # Split by the first colon followed by some digits, which works for both
185     # Windows and Unix path styles.
186     file, line = trace.match(/^(.+?):(\d+).*$/, &:captures) || trace
187     [file, line.to_i]
188   end
189
190   def expand_backtrace
191     @exception.backtrace.unshift(
192       @exception.to_s.split("\n")
193     ).flatten!
194   end
195 end
196 end
```