



デバッグ

Web Programming 春期講習



目標 デバッグを自分でできるようにしよう

メモの投稿はこちらから

タイトル

メモ

送信

[No.1]life is tech!

WEB Programmingコース デバッグに挑戦!

詳細

編集

削除

内容

- 1 エラー画面の見方
- 2 pryを使おう
- 3 デバッグに挑戦

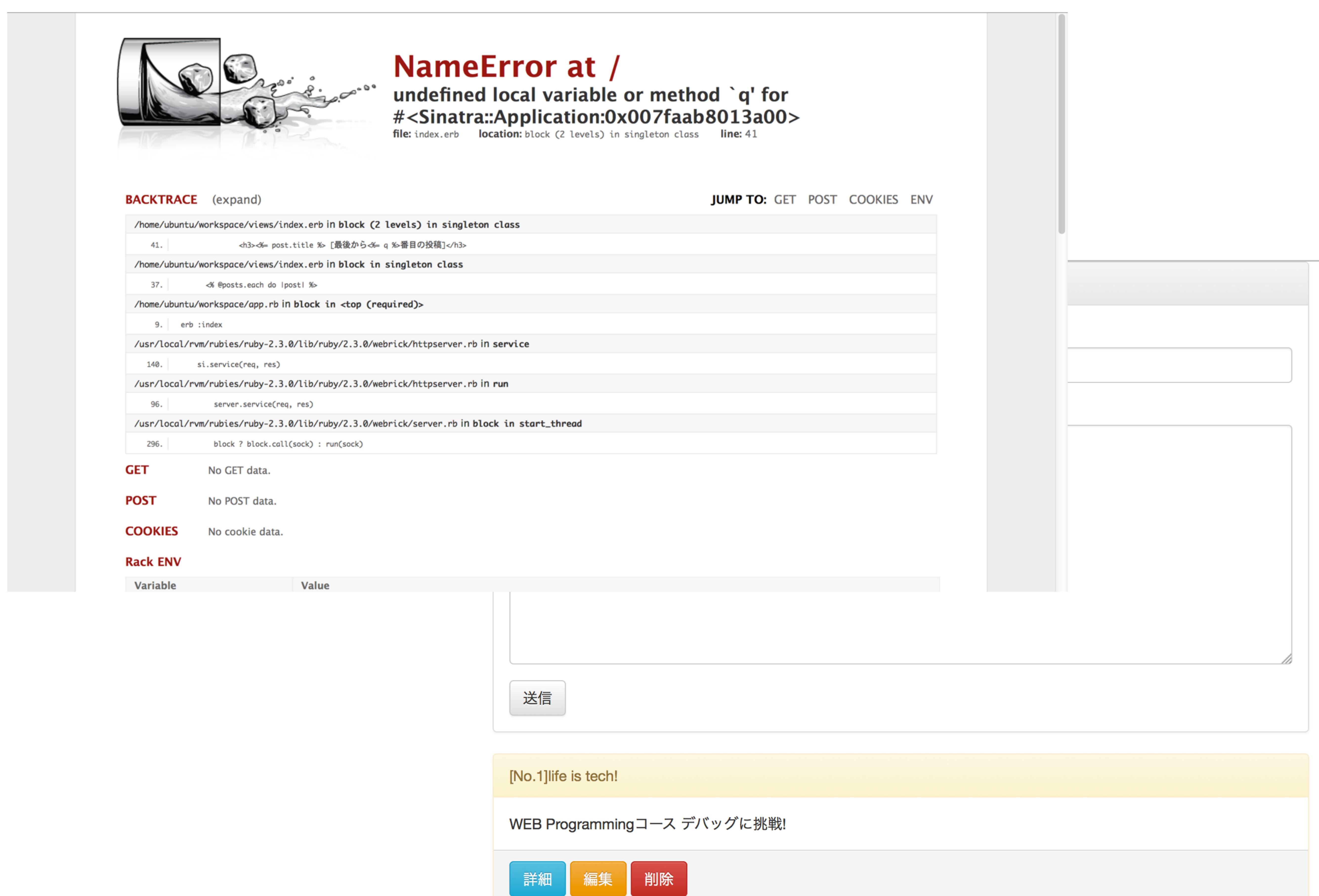
1

エラー画面の見方

ここまで開発で、何度かエラー画面を見てきたかもしれません。この章ではエラー画面の詳しい見方について説明します

デバッグとは？

プログラムを書いてサイトを作成した時にエラーが出たり、正しい動作をしないことがあります。これらのエラーやバグを発見して修正することをデバッグといいます。デバッグを進めていくためには言語の仕様を知っていたり、エラー文が読めるようになっている必要があります。この教科書ではエラー文が読め、デバッグ用のツールを使えるようになるための説明をしています。



The screenshot shows a web-based debugger interface. At the top, there is a small illustration of a person swimming in water. Below it, the error message is displayed:

NameError at /
undefined local variable or method `q' for
#<Sinatra::Application:0x007faab8013a00>
file: index.erb location: block (2 levels) in singleton class line: 41

Below the error message, there is a "BACKTRACE (expand)" section showing the call stack:

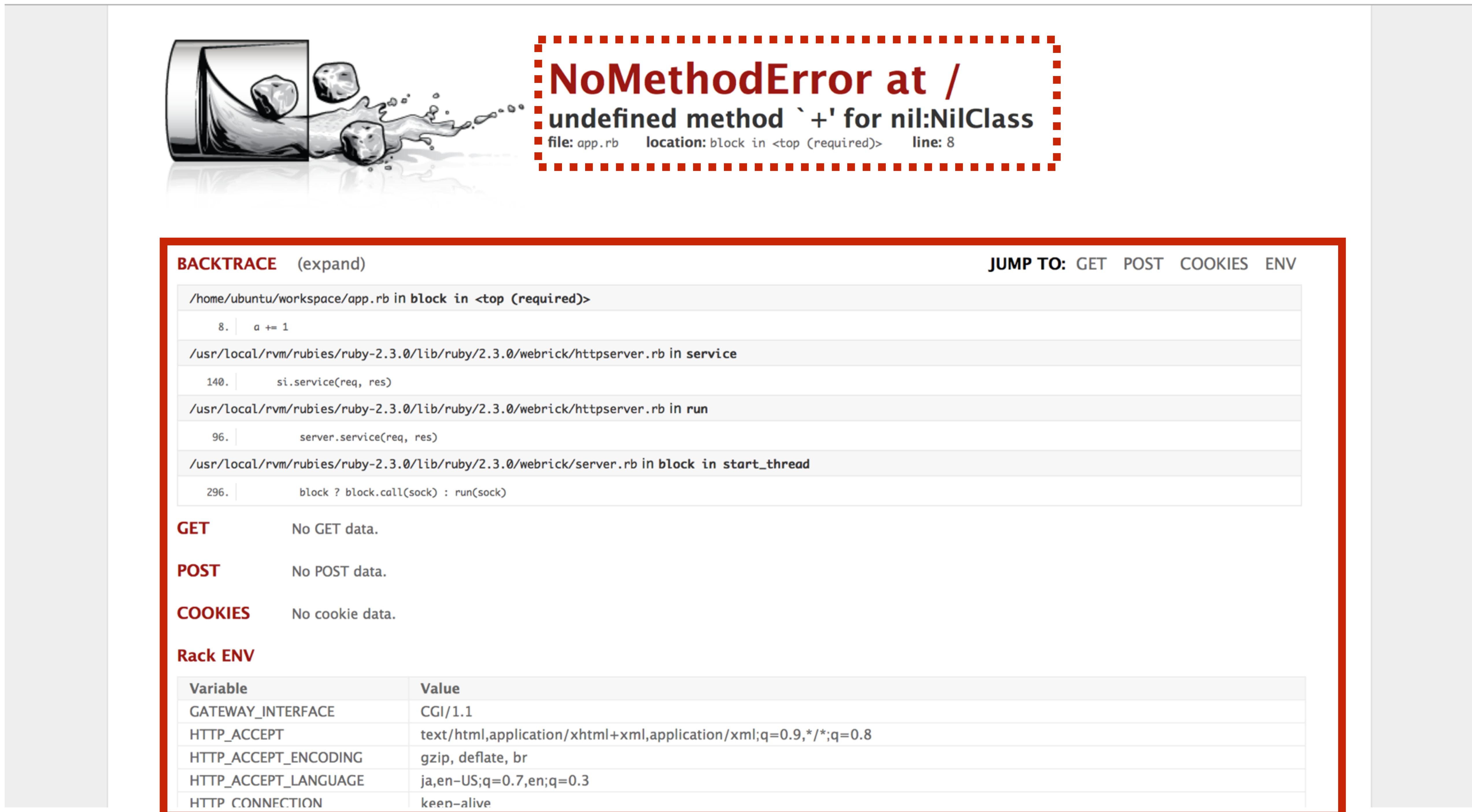
```
/home/ubuntu/workspace/views/index.erb in block (2 levels) in singleton class
  41.     <h3><%= post.title %> [最後から<%= q %>番目の投稿]</h3>
/home/ubuntu/workspace/views/index.erb in block in singleton class
  37.     <% @posts.each do |post| %>
/home/ubuntu/workspace/app.rb in block in <top (required)>
  9.     erb :index
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/httpserver.rb in service
  140.    si.service(req, res)
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/httpserver.rb in run
  96.      server.service(req, res)
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/server.rb in block in start_thread
  296.    block ? block.call(sock) : run(sock)
```

There is also a "JUMP TO: GET POST COOKIES ENV" link.

Below the backtrace, there are sections for "GET", "POST", "COOKIES", and "Rack ENV". Under "GET", it says "No GET data.". Under "POST", it says "No POST data.". Under "COOKIES", it says "No cookie data.". Under "Rack ENV", there is a table with columns "Variable" and "Value". A "送信" (Send) button is located below the table.

At the bottom of the interface, there is a yellow banner with the text "[No.1]life is tech!" and "WEB Programmingコース デバッグに挑戦!". Below the banner are three buttons: "詳細" (Details), "編集" (Edit), and "削除" (Delete).

エラー画面の見方



今までの開発でこのようなエラー画面を見かけたことがあると思います。まずは大まかに何が書いてあるか見ていきましょう。

まず点線で囲まれている部分には、起こっているエラーの種類と原因が書いてあります。赤い文字で書いてあるのが起こっているエラーで、その下の黒い太字で書いてあるのが、そのエラーを発生させた原因です。

実線で囲まれたところには、次の表に書いてあるような情報が入っています。次のページから詳しく見ていきましょう。

BACKTRACE	エラーの起きている場所が表示されます。
GET	GETメソッドでデータを送った時に、そのデータが表示されます
POST	POSTメソッドでデータを送った時に、そのデータが表示されます
COOKIES	COOKIEというものを使っていれば、そのデータが表示されます
Rack ENV	どのような通信が行われたかなど詳細な情報が入っています。 この項目を見ることは基本ありません。

エラー画面の項目1 BACKTRACE

BACKTRACE (expand)

```
/home/ubuntu/workspace/app.rb in block in <top (required)>
  8. |   a += 1
-----+
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/httpserver.rb in service
 140. |     si.service(req, res)
-----+
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/httpserver.rb in run
  96. |       server.service(req, res)
-----+
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/server.rb in block in start_thread
 296. |         block ? block.call(sock) : run(sock)
```

BACKTRACEには、どこでエラーが起こったかが書いてあります。複数の行にわたって書いてあるので複数箇所のコードを間違っているのではと思うかもしれませんが見なければならぬのは一番上の行だけです(点線で囲まれたところ)。

この場合だと、app.rbの8行目でエラーが起こったと書いてあります。
エラー画面に表示されたエラーの種類と合わせてどう直したら良いかを考えましょう。

エラー画面の項目2 GET・POST

GET

Variable	Value
body	"Web Programming Course"
id	"10"
title	"Life is Tech!"

POST

No POST data.

開発の時に起こるエラーの中には、フォームから送信されたデータが不正なために起こっているものもあります。このような時にはGET・POSTに表示されたフォームから送信された情報を見てエラーの原因を探っていきます。GETの部分にはGETメソッドで送ったデータ、POSTの部分にはPOSTメソッドで送ったデータの一覧が表示されます。

Variableはparams[:title]の:titleに当たる部分で、この写真の例の場合だとparams[:title]からWeb Programming Courseを読み出すことができます。

2

pryを使おう

エラー画面の見方については前章で説明しましたが、それだけでエラーを直していくのは大変な場合があります。そんな時はpryを使って原因を絞っていきましょう。

pryとは？

pryはrubyのコードを対話的(コードを書くたびに結果を返すこと)に実行することができるgemです。

このpryをsinatraのコード上で実行させることによって実行中のプログラムから変数の情報などを読み出すことができます。

The screenshot shows the RubyGems.org website with the pry gem page. The top navigation bar is orange with links for GEMS, GUIDES, CONTRIBUTE, SIGN IN, and SIGN UP. The pry gem page has a large title "pry 0.10.3". Below the title, it says "An IRB alternative and runtime developer console". On the left, there's a "VERSIONS:" section listing versions 1.0.0.pre1, 1.0.0.pre1, 0.10.3, 0.10.3, and 0.10.2 with their respective download links. There's also a link to "Show all versions (458 total)". In the center, there are sections for "RUNTIME DEPENDENCIES" (coderay ~> 1.1.0, method_source ~> 0.8.1, slop ~> 3.4), "DEVELOPMENT DEPENDENCIES" (bundler ~> 1.0), and "LICENSE" (MIT). On the right, it shows "TOTAL DOWNLOADS 23,687,558" and "FOR THIS VERSION 2,272,280", along with links to "Show all versions (458 total)" and "Gemfile", and command-line install and update instructions.

pryの使い方1

pryを使うには、まずインストールする必要があります。

gemfileに次の1行を加えます。

```
Gemfile  
gem 'pry'
```

bundleを実行してインストールを完了させます。

```
ターミナル  
$ bundle
```

app.rbの問題が起こっているコードの直前に「binding.pry」を加えます
エラーが起こっている箇所の後に加えるとpryが実行されないので注意し
ましょう。

```
app.rb(例)  
get '/' do  
  @a = 10  
  @b = 2  
  @c = "4"  
  binding.pry  
end
```

pryの使い方2

binding.pryを加えたら、いつも通りにアクセスします。
するとページが表示されずに読み込み中の状態が続いていると思います。
ここでターミナルを見ると写真のようになっています。

```
[1] pry(#<Sinatra::Application>) >
```

ここにRubyのコードを入れると、その戻り値(結果が返ってきます)

```
[1] pry(#<Sinatra::Application>) > @a  
=> 10  
[2] pry(#<Sinatra::Application>) > @a + @b  
=> 12  
[3] pry(#<Sinatra::Application>) > @a + @c  
TypeError: String can't be coerced into Fixnum  
from (pry):3:in `+'  
[4] pry(#<Sinatra::Application>) > if @a < 2  
[4] pry(#<Sinatra::Application>)*   puts "@aは2未満"  
[4] pry(#<Sinatra::Application>)* else  
[4] pry(#<Sinatra::Application>)*   puts "@aは2以上"  
[4] pry(#<Sinatra::Application>)* end  
@aは2以上  
=> nil
```

lsというコマンドを使うと、変数とメソッドの一覧を見ることができます。

ターミナル

```
> ls
```

```
[6] pry(#<Sinatra::Application>) > ls  
Sinatra::Helpers#methods:  
  attachment  cache_control  error  headers      logger      not_found?  send_file    status     to  
  back        client_error?  etag    informational? mime_type   redirect    server_error? success?   uri  
  body        content_type  expires  last_modified not_found   redirect?   session    time_for  url  
Sinatra::Templates#methods:  
  asciidoc  coffee  erb    find_template  less    markaby    mediawiki   rabl    rdoc    scss    stylus    wlang  
  builder   creole  erubis haml      liquid   markdown  nokogiri   radius  sass    slim    textile  yajl  
Sinatra::JSON#methods: json  
Sinatra::Base#methods:  
  app    call   env   forward  options  params=  request   response   settings  
  app=   call!  env=  halt    params   pass     request=  response=  template_cache  
Sinatra::EngineTracking#methods:  
  builder?  creole?    erb?    haml?    liquid?   markdown?  radius?   sass?    slim?    with_engine  
  coffee?   current_engine  erubis?  less?    markaby?  nokogiri?  rdoc?    scss?    textile?  
Sinatra::Capture#methods: capture  capture_later  
Sinatra::ContentFor#methods: content_for  content_for?  yield_content  
Sinatra::Cookies#methods: cookies  
Sinatra::LinkHeader#methods: link  link_headers  prefetch  stylesheet  
Sinatra::Streaming#methods: stream  
Sinatra::RespondWith::Helpers#methods: respond_to  respond_with  
Sinatra::ActiveRecordHelper#methods: database  
self.methods: __pry__  
instance variables:  
  @a  @app  @b  @c  @current_engine  @default_layout  @env  @params  @preferred_extension  @request  @response  @  
  template_cache  
class variables: @@mutex  
locals: _  __  _dir_  _ex_  _file_  _in_  _out_  _pry_
```

3

エラーの種類

この章ではRubyやSinatraが発生させるエラーの種類と原因・対策について説明します。

Case1: SyntaxError

SyntaxErrorはプログラム上の文法エラー、例えばendの数が足りなかった時やカッコの数が足りなかった時にエラーが出てきます。

このエラーが出てきた時には、エラー文の指す行にエラーがないことがあります。そのため、コードを1行1行見直して何が不足しているかを確認しないといけません。

```
25 post '/posts/:id/update' do
26   post = Post.find_by(id: params[:id])
27
28   if !post.nil?
29     if !params[:body].nil?
30       post.body = params[:body]
31     else
32       post.body = "内容がありません"
33
34
35     redirect '/posts'
36   else
37     redirect '/'
38   end
39 end
```

application error

SyntaxError at /

/home/ubuntu/workspace/app.rb:36: syntax error,
unexpected keyword_else, expecting keyword_end
/home/ubuntu/workspace/app.rb:39: syntax error,
unexpected end-of-input, expecting keyword_end
file: dependencies.rb location: rescue in load_dependency line: 244

BACKTRACE (expand)

JUMP TO: GET POST COOKIES ENV

```
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/httpserver.rb in service
140.    si.service(req, res)
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/httpserver.rb in run
96.    server.service(req, res)
/usr/local/rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/webrick/server.rb in block in start_thread
296.    block ? block.call(sock) : run(sock)
```

Tips

SyntaxErrorを減らすには

このようにエラー発生箇所がわかりづらいSyntaxErrorですが、ほんの少しの工夫で減らすことができます。

その工夫とはインデント(行頭の空白)のことです。ブロック単位でインデントを整えるだけでendの不足がわかりやすくなります。

また間違ったブロックにコードを書くことを減らすこともできます。

Case2: NameError: uninitialized constant

NameErrorのうちuninitialized constantというメッセージが続くものは、定義されていないクラスか定数(大文字スタートの変数)を使おうとした時に出てくるエラーです。このエラーが出ている時には、スペルミスを起こしているか外部ファイルで宣言されたクラスが読み込めていないことが多いです。下の表に、このエラーが起きる原因と確認することの一覧があるので確認しておきましょう。

エラーの原因	確認すること
ファイルが読み込めていない	Gemfileに書かれているか bundleは実行されているか requireを書いたか ファイルを置いた場所が正しいか 外部ライブラリの場合、ドキュメントを参考にスペルがっているか確認する。
スペルミス	自分で宣言した場合は、宣言時とスペルが間違っていないか確認する。
初期化忘れ	代入文があることを確認する。

Case3: NameError: undefined local value or method

NameErrorのうちundefined local valueというメッセージが続くものは、定義されていないローカル変数から値を読みだそうとした時や、定義していないメソッドを使おうとした時に出てくるエラーです。このエラーが起こっている時は、定義・初期化(値の代入)を忘れているか、定義した時と名前を間違えています。

そのブロック内を再度見直してみましょう。

エラーの原因	確認すること
定義・初期化をしていない	変数の時はブロック内に初期化した文があるかを確認し、メソッドの時はコードに定義があるかを確認する。
スペルミス	定義・初期化した時とスペルが変わっていないかを確認する。
変数の種類が違う	インスタンス変数(@から始まる変数)で宣言したのにローカル変数になっている。
参照範囲の問題	app.rbで宣言したローカル変数をerbで読みだそうとしているか。

Tips

変数の参照範囲

変数には、その種類に応じて値を保持していられる期間に制限があります。

@から始まるインスタンス変数はapp.rbで宣言してもerb内で読み込めます(アクセスしたルーティングの処理が終わるまでずっと読み出せます)が、先頭に何もつかないローカル変数は宣言したブロック内でしか読み出せません。

Case4: NoMethodError

NoMethodErrorは、あるインスタンスに対して使おうとしたメソッドが存在しない時に発生するエラーです。そのインスタンスが属するクラスに使おうとしたメソッドが存在しているか、メソッドのスペルが間違っていないかを確認しましょう。

また、変数のスペルが宣言時と異なっている時にもこのエラーが発生することがあります。(インスタンス変数(@から始まる変数)の場合、初期化していないとnil(NilClass)が入ってしまうため)

エラーの原因	確認すること
メソッドが存在しない	メソッドのスペルがあつてあるか そのインスタンスが属するクラスに書いたメソッドが存在しているか
スペルミス	宣言した時と変数のスペルに違いがないか

Case5: NoErrno:ENOENT No such file or directory

NoErrno:ENOENT No such file or directoryは、読みだそうとしたファイルやディレクトリが存在しない時に出てくるエラーです。sinatraの場合、erb ~で読みだそうとしたerbファイルが存在していない時に多く出てきます。スペルミスやファイル作成忘れをしていないかを確認しましょう。ファイルが置かれている場所があるかも確認しておきましょう。

Tips cannot load such file

app.rbの先頭の行にはrequireという文がたくさんあったと思います。これは外部ライブラリからクラスやメソッドを読みだすのに使われています。このrequireで読み込もうとしたファイルが存在しないと、ターミナル上にこのエラーが出てサーバが立ち上がりません。次のことを確認しましょう。

エラーの原因	確認すること
ファイルが読み込めていない	Gemfileに使おうとしているgemは書かれているか bundleは実行されているか ファイルを置いた場所が正しいか

Case6: Sinatra doesn't know this ditty

このエラーはSinatra独自のエラーです。アクセスしようとしたルーティングがない時に出てきます。ルーティングを作成したか、作成したルーティングとメソッド(GETやPOST)が一致しているかを確認しましょう。

Sinatra doesn't know this ditty.



Try this:

```
post '/posts' do
  "Hello World"
end
```

エラーの原因	確認すること
ルーティングが存在しない	ルーティングがあるか確認する
URLはあってるがメソッドが間違っている	自分の送ったメソッドと定義されたルーティングのメソッドが一致しているかを確認する。

Case7: エラーが出ないけど動かない

エラーが出なくても思った通りの動作をしていくれるとは限りません。値を読みだそうとした変数を間違えていたり条件文が間違っていたりすると予期しない動作をすることがあります。

pryやlogger.infoを使って原因となる場所を絞りこんでいきましょう。

4

デバッグに挑戦

この教科書で習ったことを使ってデバッグに挑戦します。

デバッグに挑戦するもの

今回は下の写真のようなメモアプリが動くようにデバッグしていきます。

デバッグを開始する前にサイトの仕様を見ておきましょう。

The screenshot shows a web-based memo application. At the top, there's a header bar with the text 'メモの投稿はこちらから' (Post a memo here). Below it is a form with fields for 'タイトル' (Title) and 'メモ' (Memo), both represented by input text areas. A '送信' (Send) button is located at the bottom right of the form. Below the form, a message '[No.1]life is tech!' is displayed in a yellow box. Underneath, a list of memos is shown with the entry 'WEB Programmingコース デバッグに挑戦!'. At the bottom of the list are three buttons: '詳細' (Details), '編集' (Edit), and '削除' (Delete).

ルーティング	メソッド	動作
/	GET	memosテーブルのデータをすべて取得してindex.erbを開く
/memos	POST	memosテーブルに新しいデータを入れて/に戻る
/memos/:id	GET	idが:idであるmemosテーブルのデータを取得しshow.erbを開く
/memos/:id	POST	idが:idであるmemosテーブルのデータを更新して/に戻る
/memos/:id/edit	GET	idが:idであるmemosテーブルのデータを取得しedit.erbを開く
/memos/:id/delete	GET	idが:idであるmemosテーブルのデータを削除し/に戻る

memosテーブル	
カラム	型
id	integer
title	string
body	text
created_at	datetime
updated_at	datetime

ver. 1.0.0 (2016/01/31改訂)

制作 : Life is Tech! School 教材開発チーム

著者 : 宇山 拓夢