Webエンジニアコース フルタイム

－ Webアプリケーション開発 Rails編 －

DIVE INTO CODE

はじめに

この講義の模様は、今後の運営改善に役立てるため、録画をいたします。

映像は、弊社YouTubeアカウントに**限定公開**で保存され、一般に公開されることはありません。

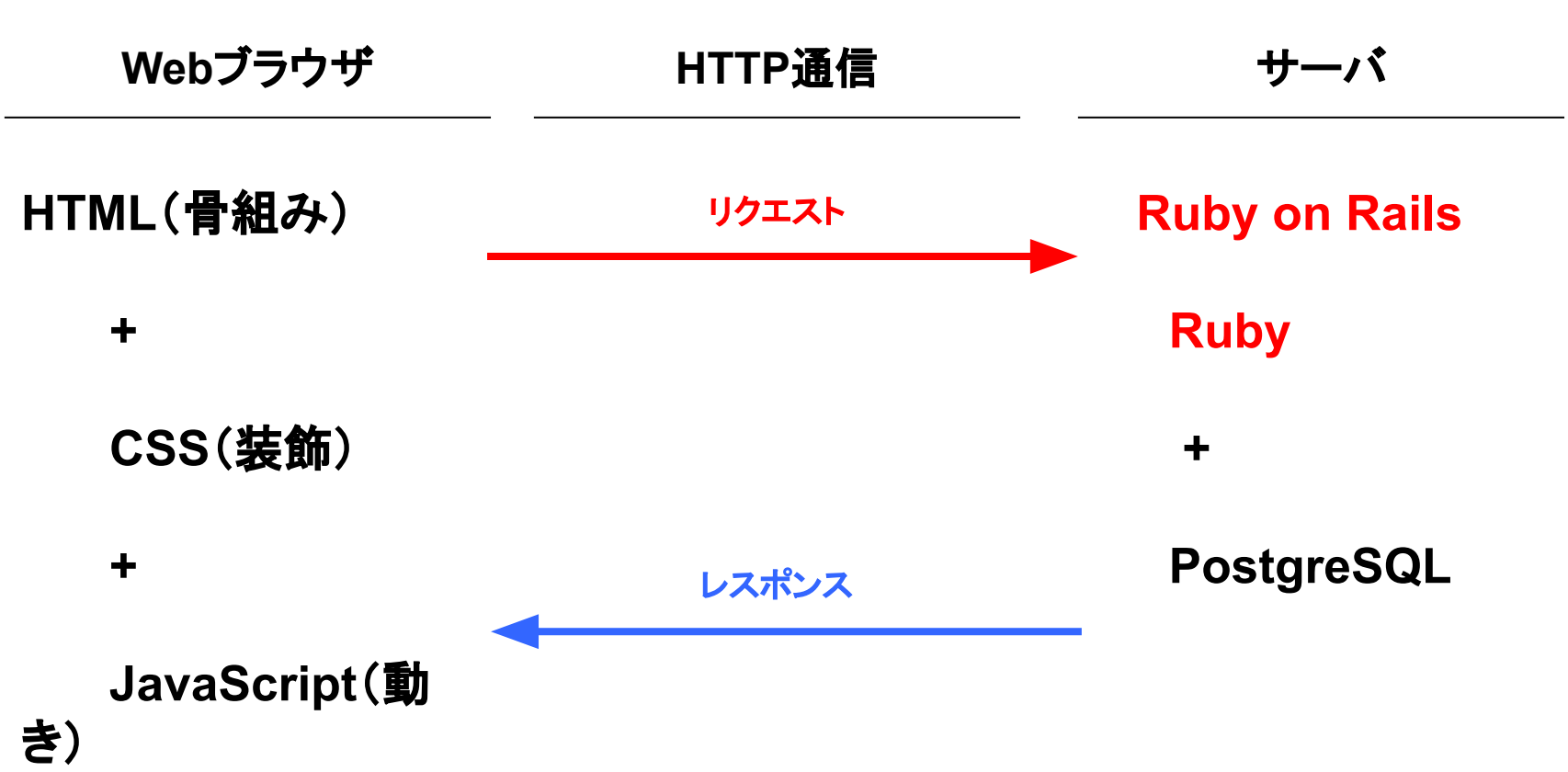
ご理解ご協力のほど、よろしくお願いいたします。

構成

1. Webアプリケーションの処理の流れ
2. デバッグすべき6つの視点
3. グループワーク
4. 発表
5. まとめ

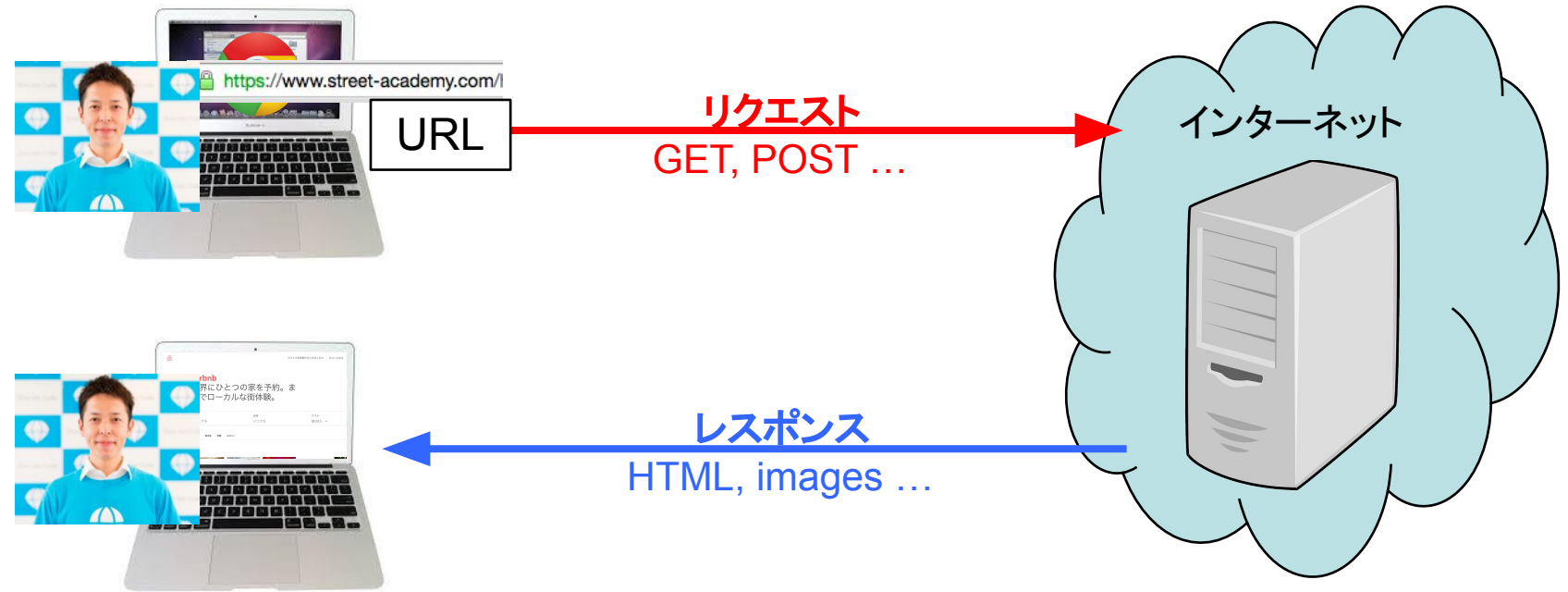
Webアプリケーションの処理の流れ

Webアプリケーション開発ができるようになるためには、ブラウザやHTTP、サーバのすべてを理解する必要がある。



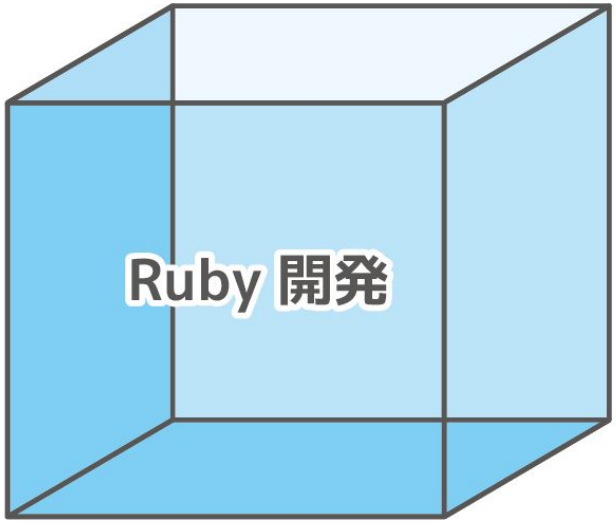
Webアプリケーションの処理の流れ

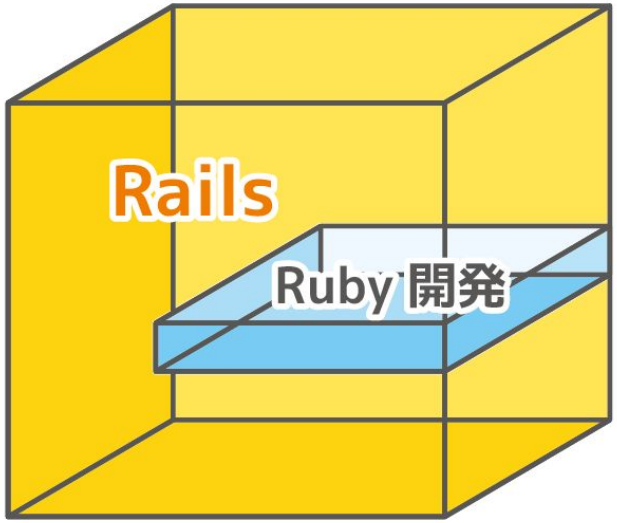
ブラウザを通して**通信**が流れる。通信先のコンピュータから処理結果が返る。ブラウザ上に画面が表示される。



Webアプリケーションの処理の流れ

Rubyは開発言語。Ruby on RailsはRubyで開発する際の「型」となるフレームワーク。





Webアプリケーションの処理の流れ

デンマーク人エンジニア「David HeinemeierHansson」氏によって、**無駄な時間を減らす**ために作られた。

DRY (Don't repeat yourself)

「同じことを繰り返さない」

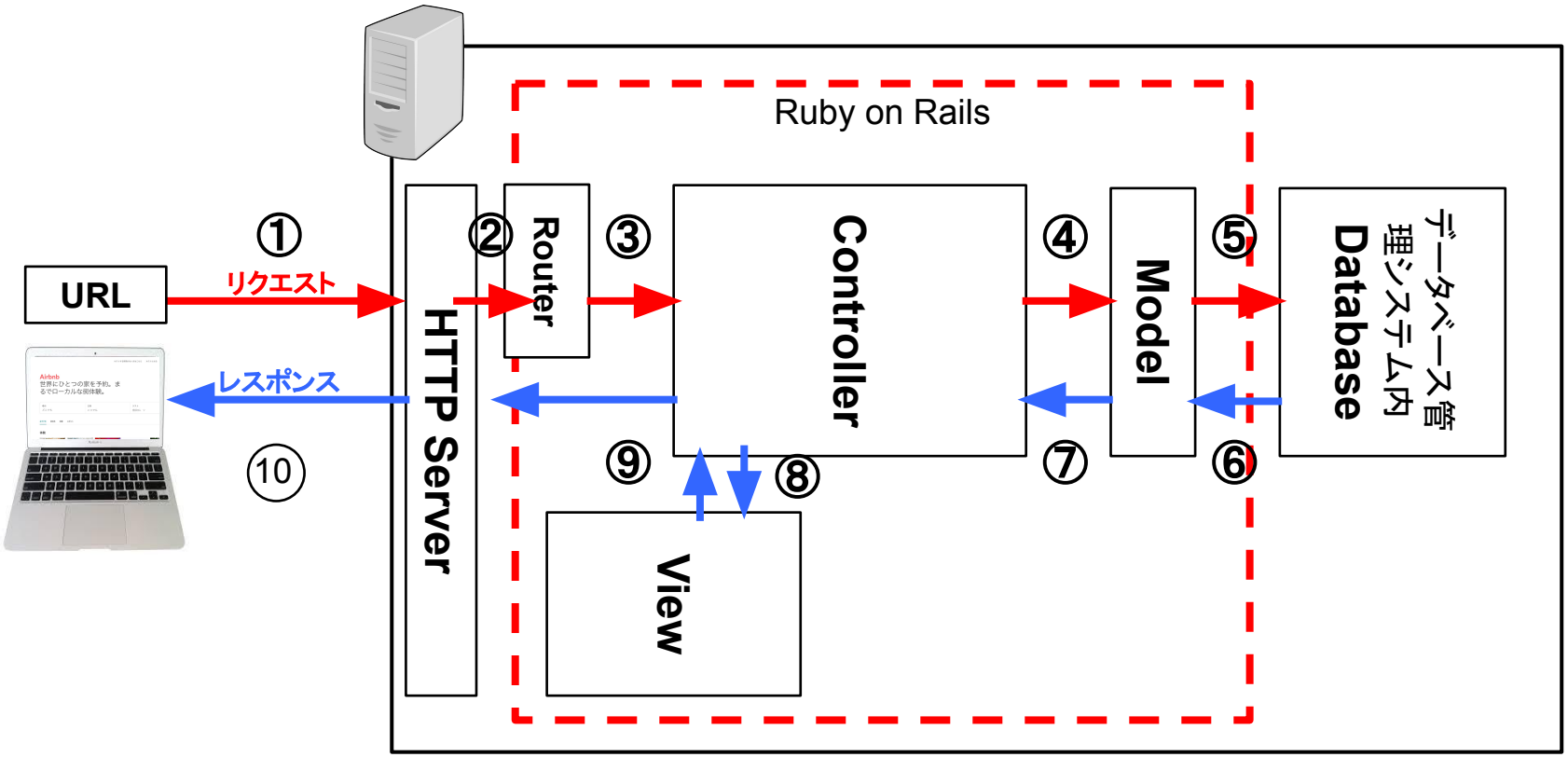
CoC (Convention over Configuration)

「設定よりも規約」

“ぼくがやろうとしていることは、自分の時間を浪費する時間を減らそうっていうこと。どうでもいいことに時間を費やすのを減らす。”

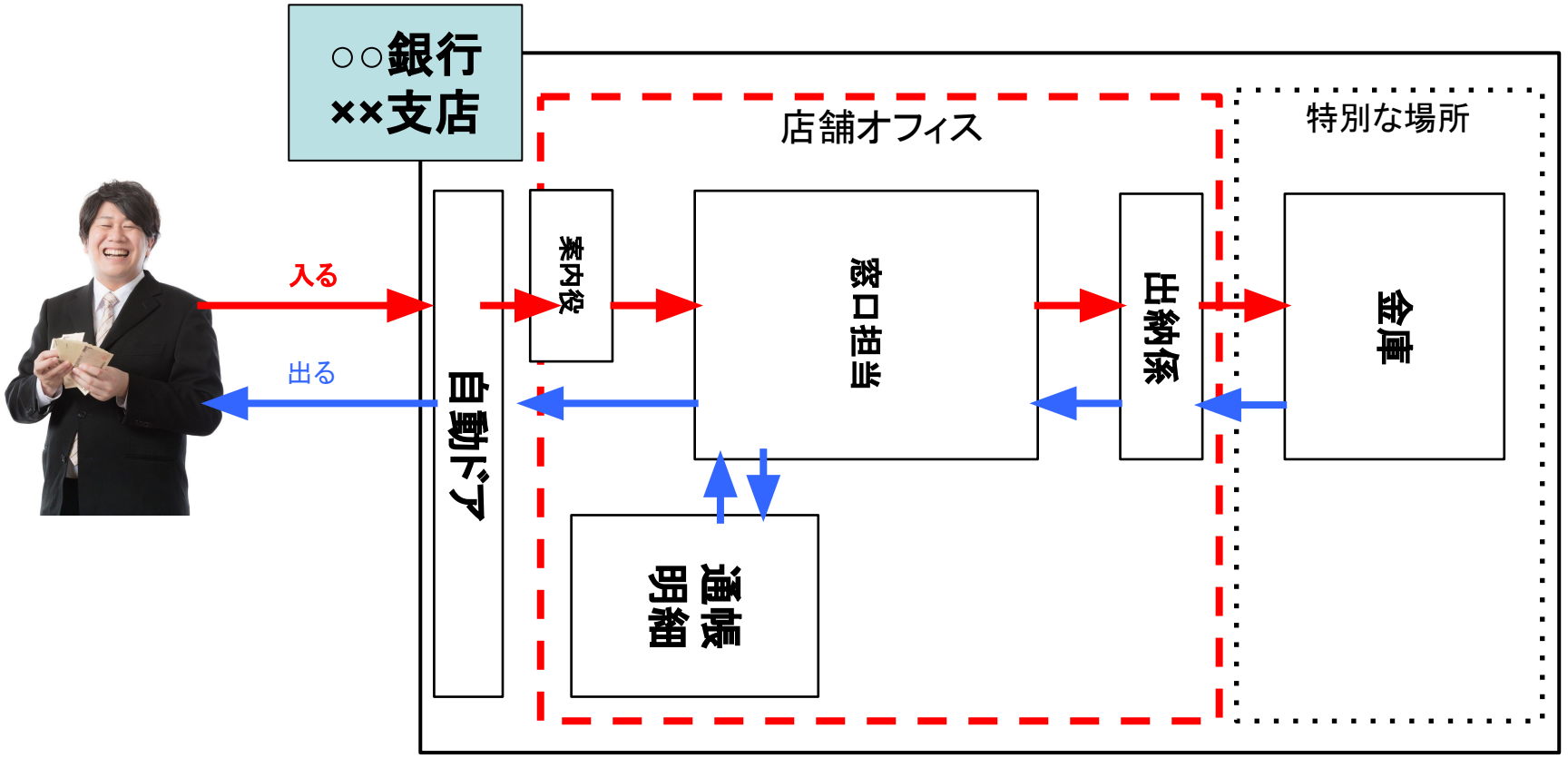
Webアプリケーションの処理の流れ

Webブラウザから送られたHTTPリクエストをサーバが受け取り役割に応じて処理が順番に流れ、レスポンスが返る。



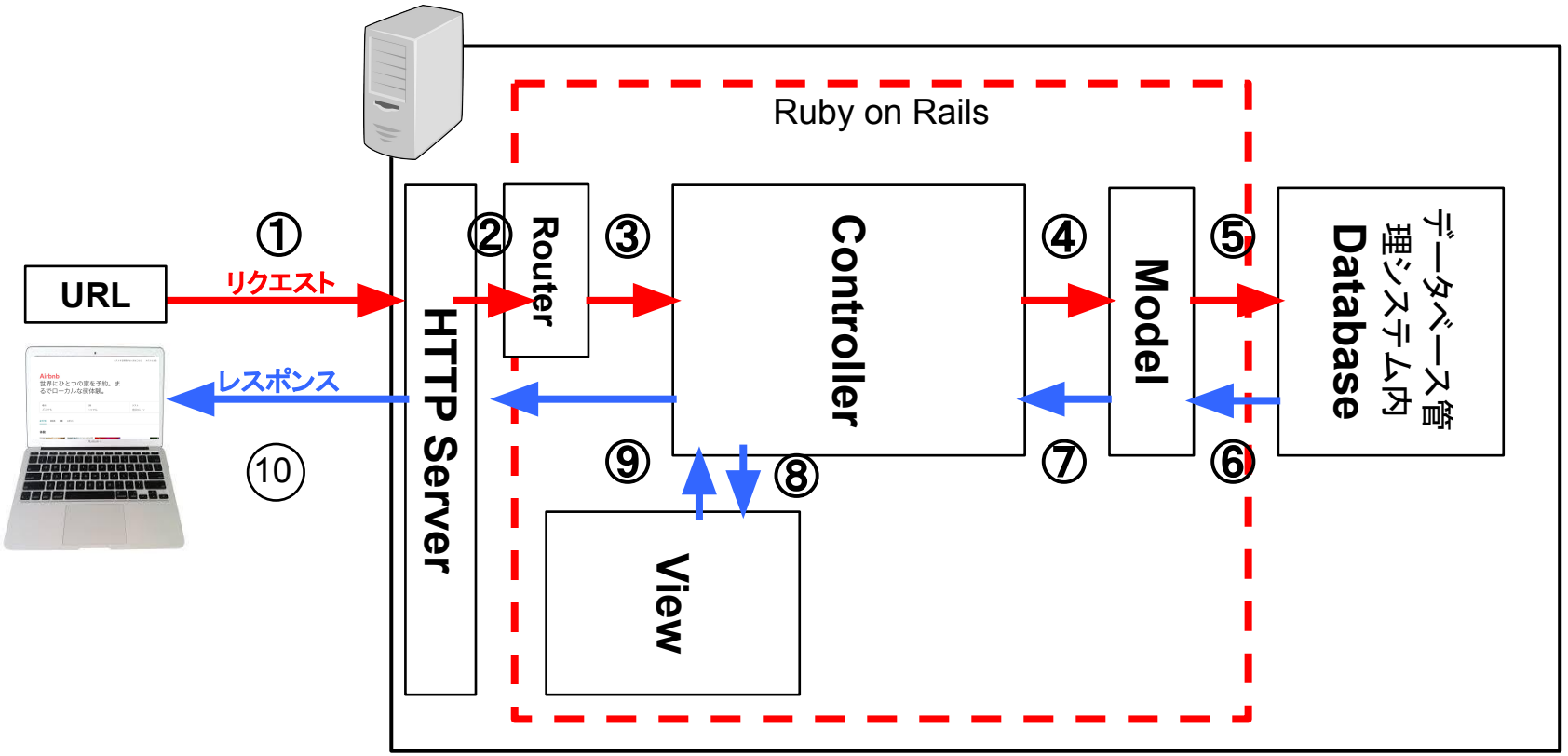
Webアプリケーションの処理の流れ

経験したことに例えて理解する。Railsの処理の流れは、銀行のオペレーションの流れと一緒。



Webアプリケーションの処理の流れ

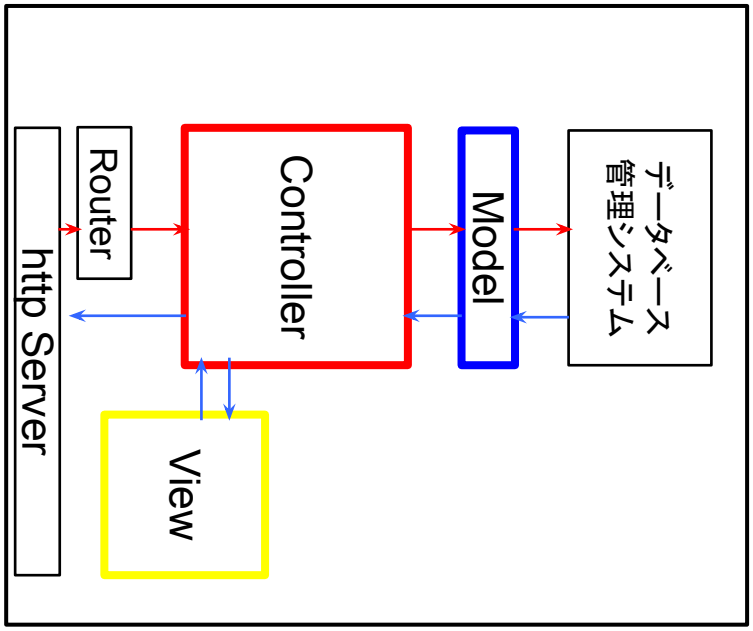
Ruby on Rails を理解する第一歩は、http リクエストからレスポンスまでの流れを理解すること。



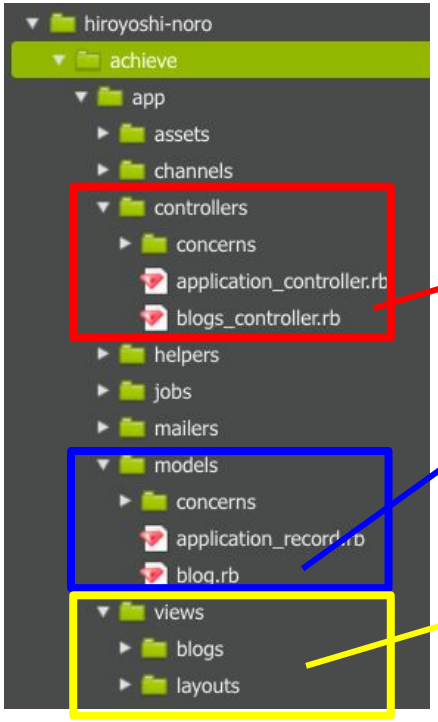
Webアプリケーションの処理の流れ

各役割は所定のフォルダ内にファイルとして存在。その中に開発言語 Ruby が書かれている。

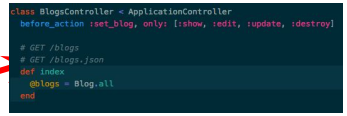
概念図



ローカル環境内

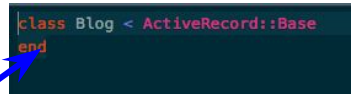


役割別のフォルダ構造。
ファイル内に Ruby が書かれている。

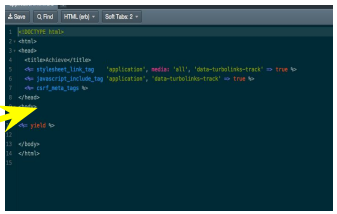


```
class BlogsController < ApplicationController
  before_action :set_blog, only: [:show, :edit, :update, :destroy]

  # GET /blogs
  # GET /blogs.json
  def index
    @blogs = Blog.all
  end
```

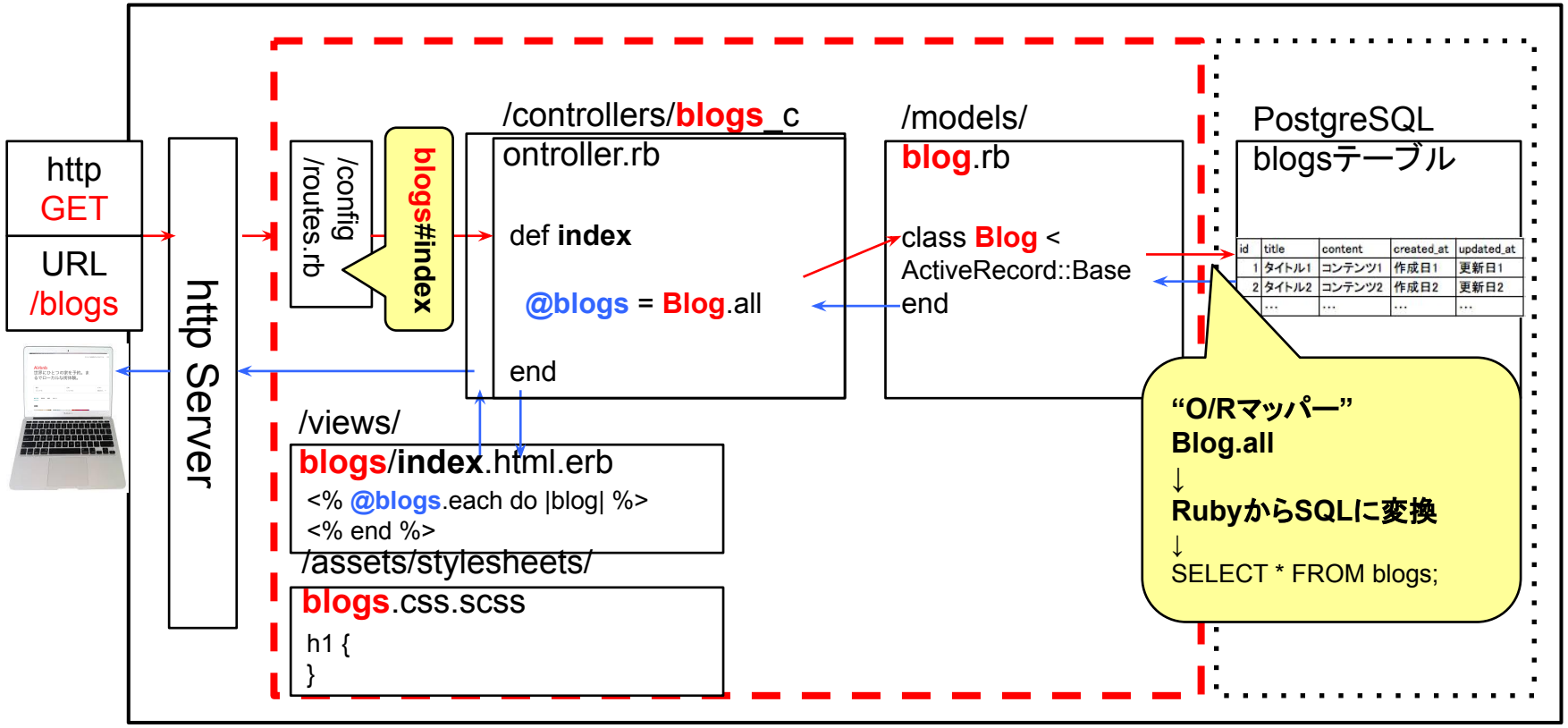


```
class Blog < ActiveRecord::Base
end
```



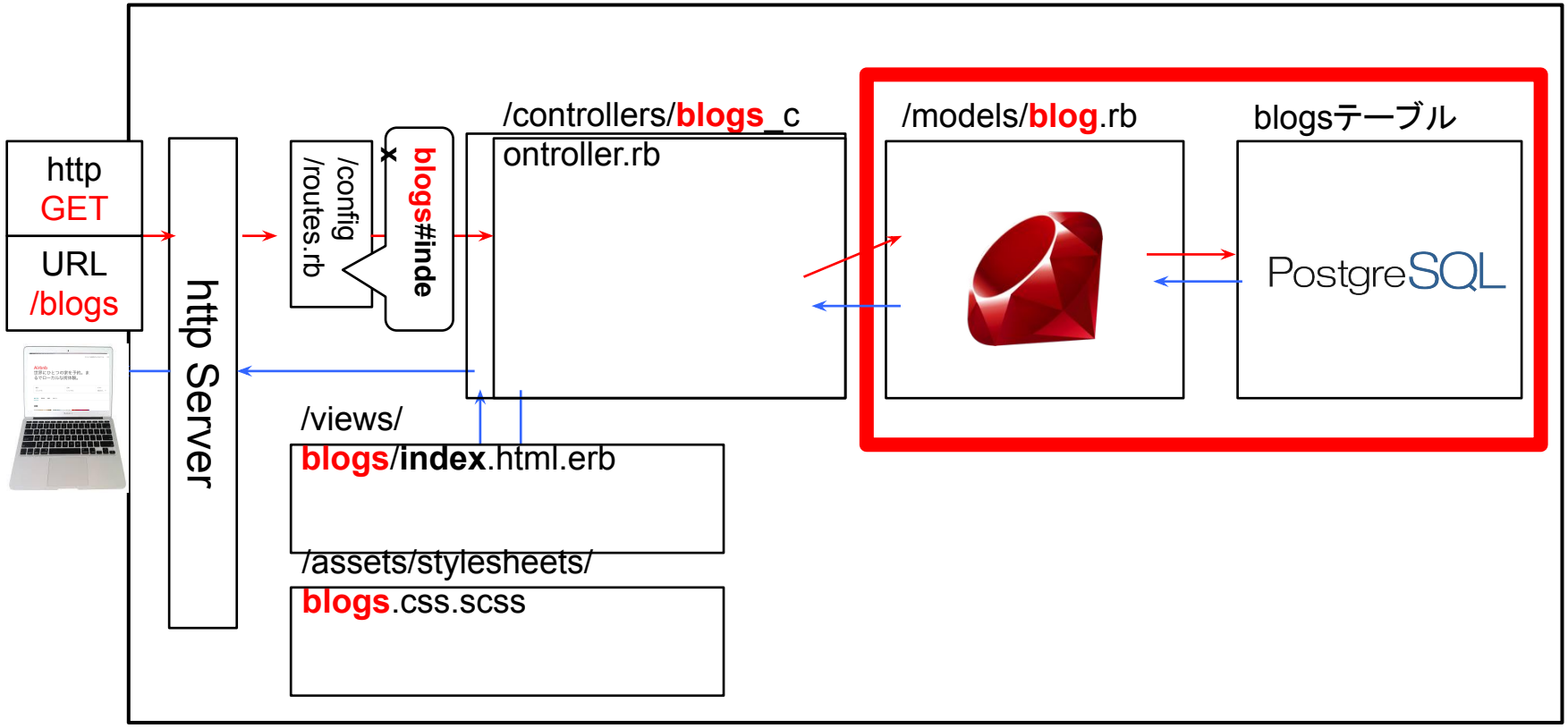
Webアプリケーションの処理の流れ

処理の流れを具体的に読み解くことができるようになると、ようやく自力で開発ができるようになる。



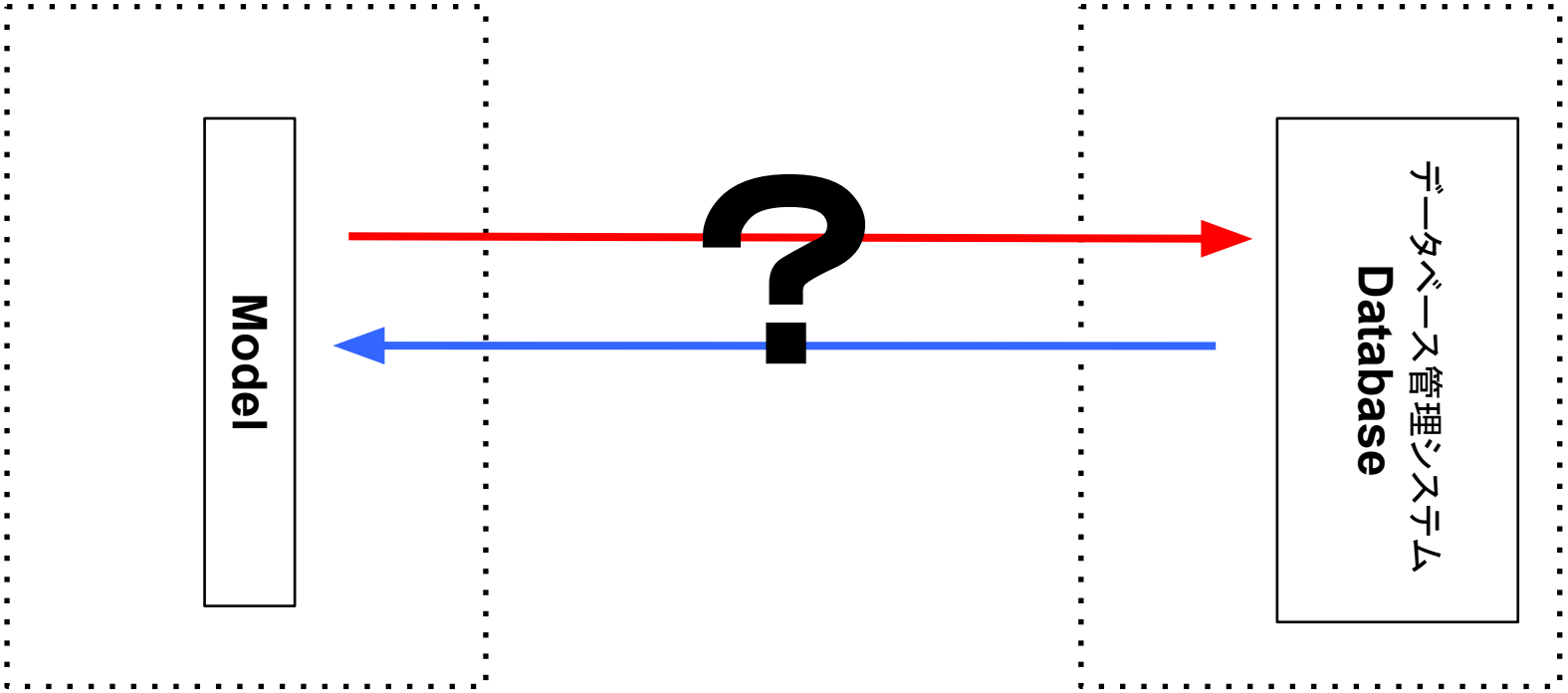
Webアプリケーションの処理の流れ

モデルは、データベース管理システムとのやりとりをする役割を担っている。



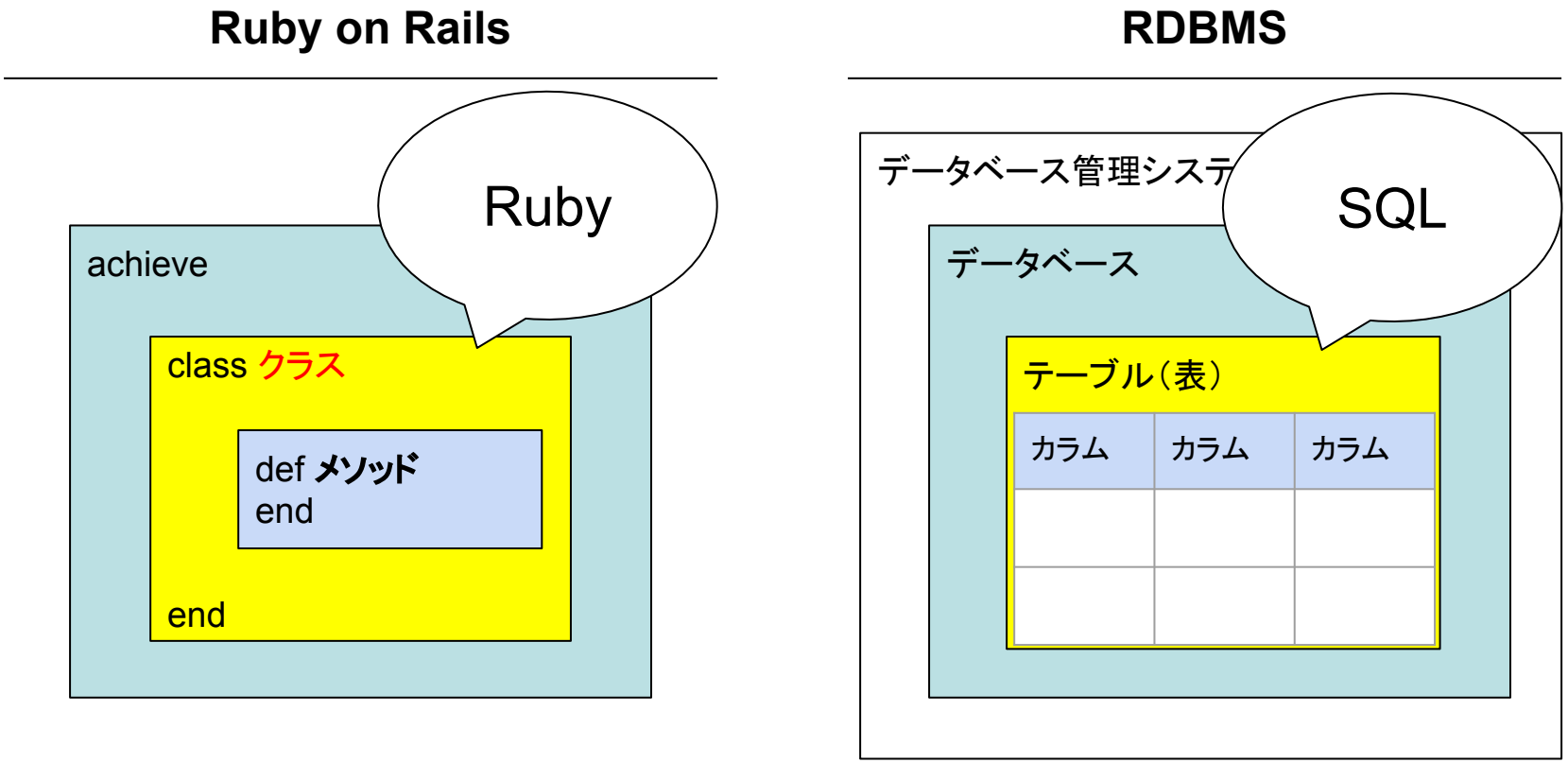
Webアプリケーションの処理の流れ

どのようにやりとりをしているのかを理解しよう。



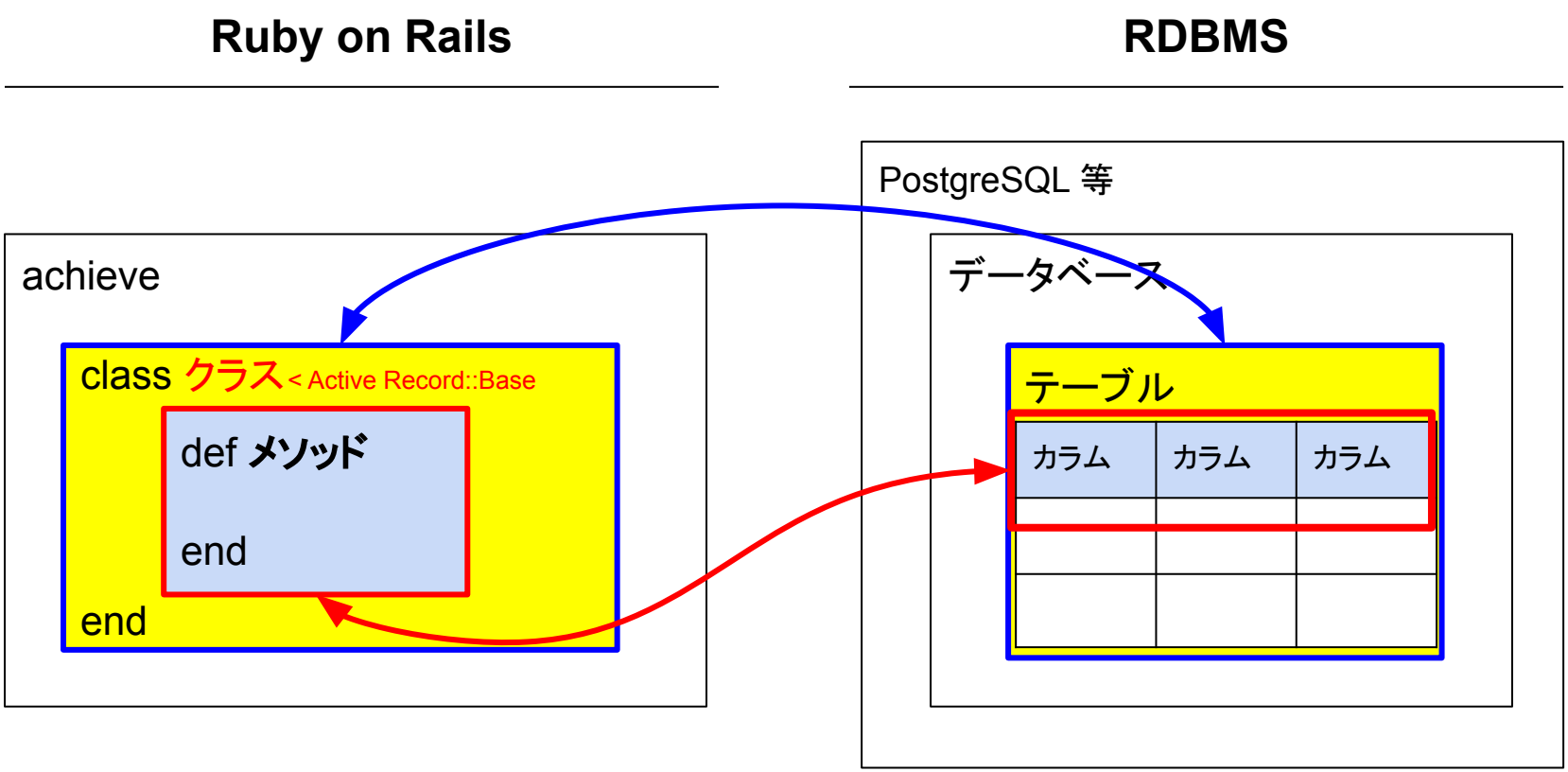
Webアプリケーションの処理の流れ

オブジェクト指向言語とデータベース管理システムの世界は全く異なる。
RubyとSQLは、やりとりに通訳が必要。



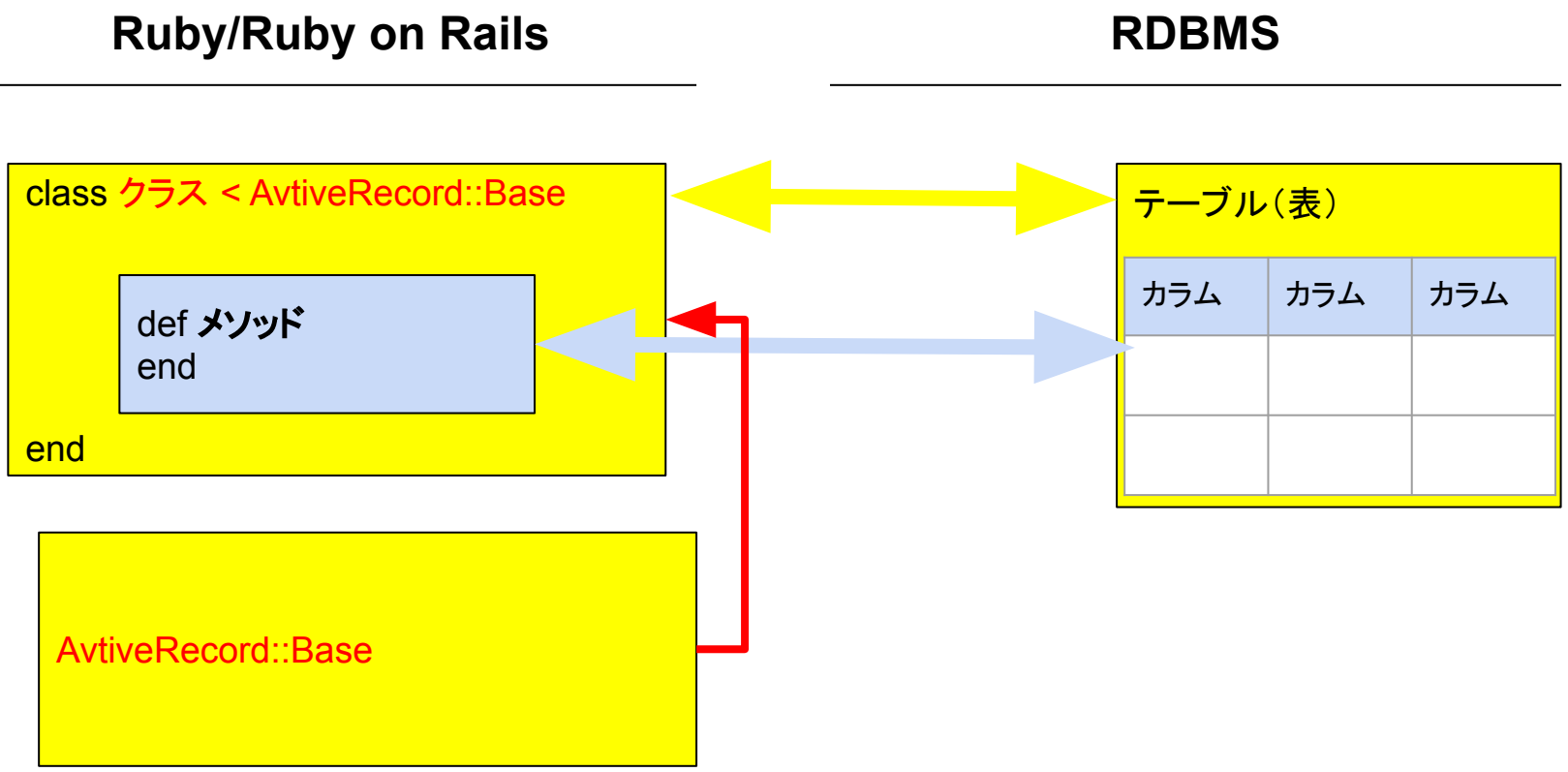
Webアプリケーションの処理の流れ

クラスはデータベース管理システムのテーブル。メソッドはテーブルのカラムに 1対1 で対応させて”翻訳”している。



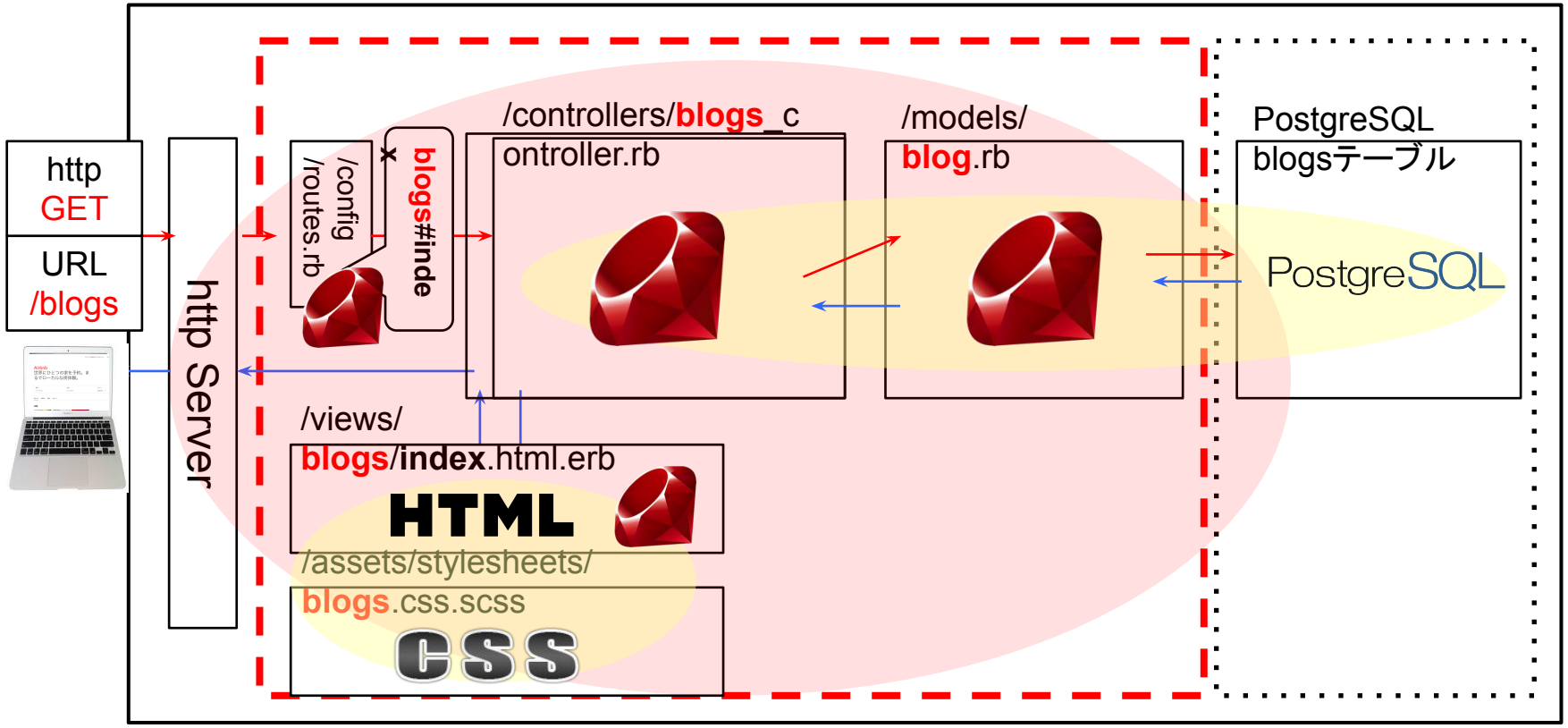
Webアプリケーションの処理の流れ

ObjectiveとRDBMSを対応（マッピング）させる仕組みがActiveRecord。
O/Rマッパーと呼ぶ。



Webアプリケーションの処理の流れ

Ruby on Railsはリクエストからレスポンスまでの流れをつくる。その中にHTML、CSS、JavaScript、Ruby等を実装する。



Webアプリケーションの処理の流れ

Ruby on Rails の標準設定での規約（ルール）をおさえておこう。

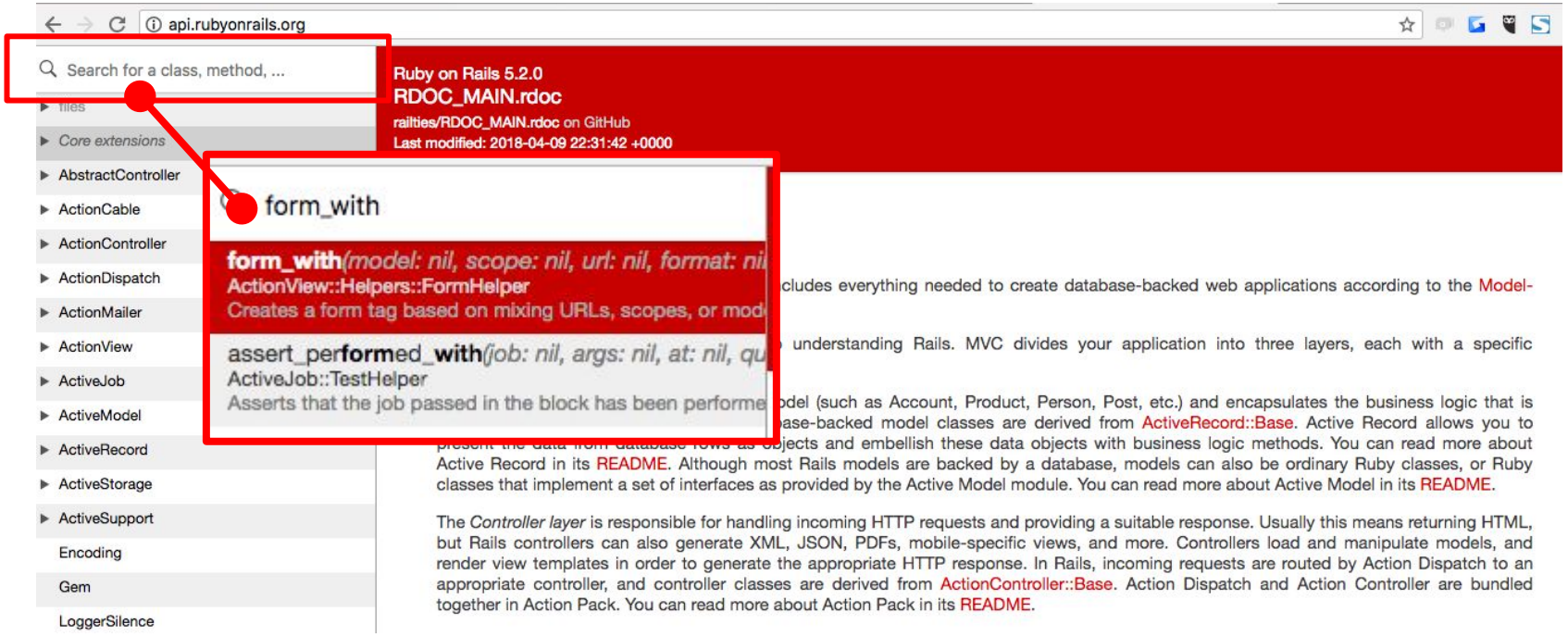
分類	内容	例
View	コントローラ名と同名のディレクトリのViewが使われる。	**Blogs**Controllerの場合、**blogs**ディレクトリが使われる。
View	コントローラのアクション（メソッド）名と同名のViewファイルが使われる。	def **new**の場合、**new.html.erb**ファイルが使われる。
テーブル	モデルに対して、モデル名を複数形にしたテーブル名が使われる	**Blog**の場合、**blogs**テーブルが使われる。

その他の規約もチェックしておこう！ https://diveintocode.jp/tips/naming

Webアプリケーションの処理の流れ

【ワーク】公式 Rails API を活用して、メソッドの仕様を調べよう。例：form_with, redirect_to, resources

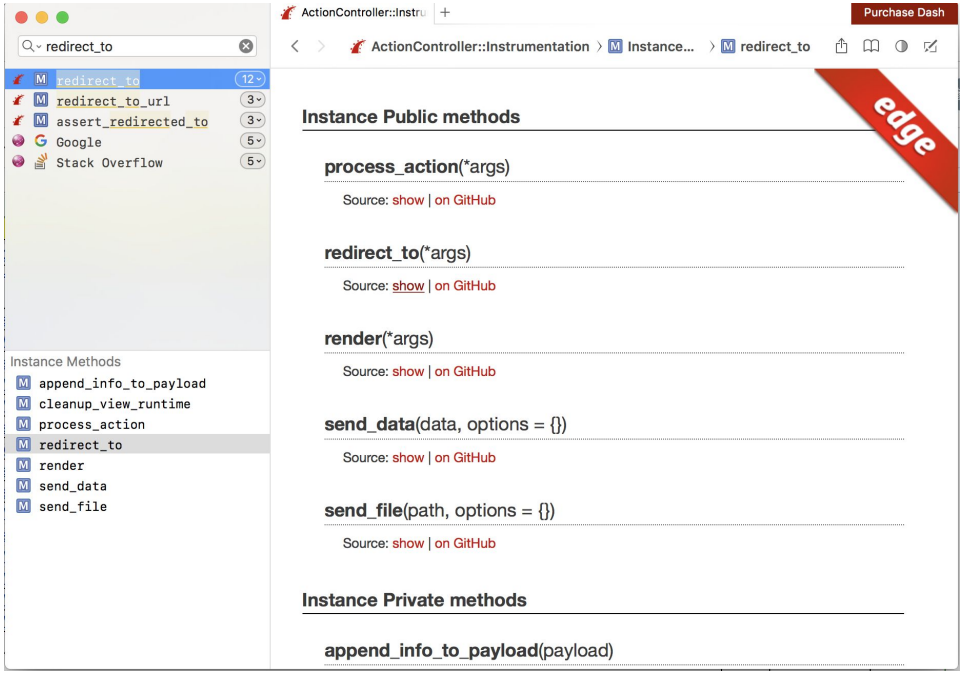
http://api.rubyonrails.org/



Webアプリケーションの処理の流れ

【ワーク】Dash(リファレンスビューア)を使ってみよう！

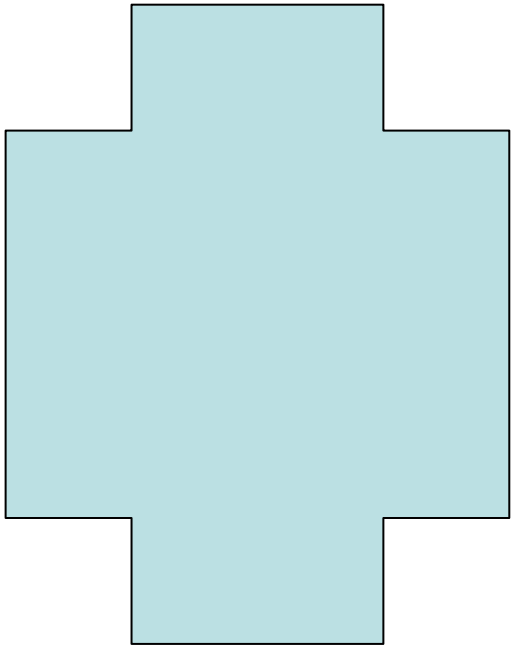
公式サイト：https://kapeli.com/dash
導入例：https://qiita.com/hkusu/items/267d802e2ab15a1b506e



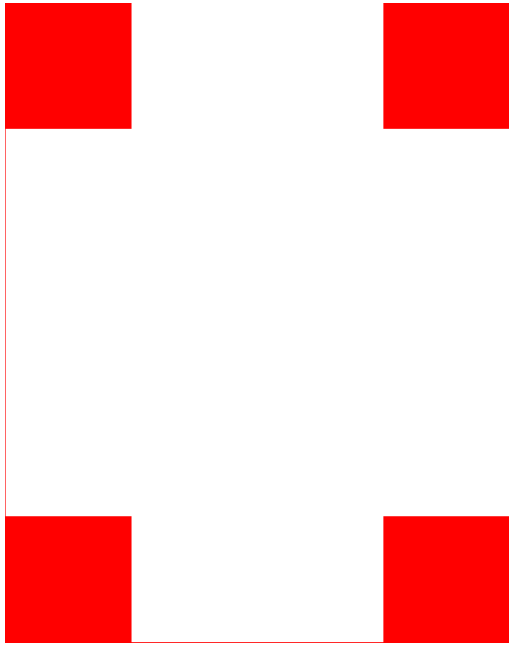
デバッグすべき6つの視点

デバッグは、問題解決思考そのもの。あるべき状態と現状のギャップを見つける「視点」をもつことが重要。

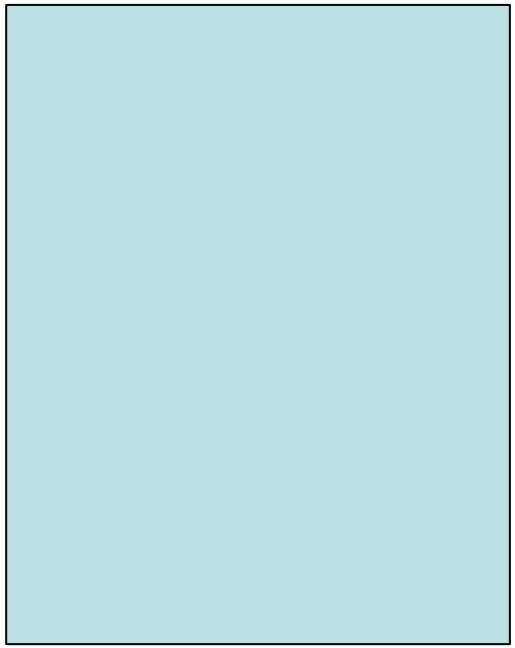
現状



ギャップ

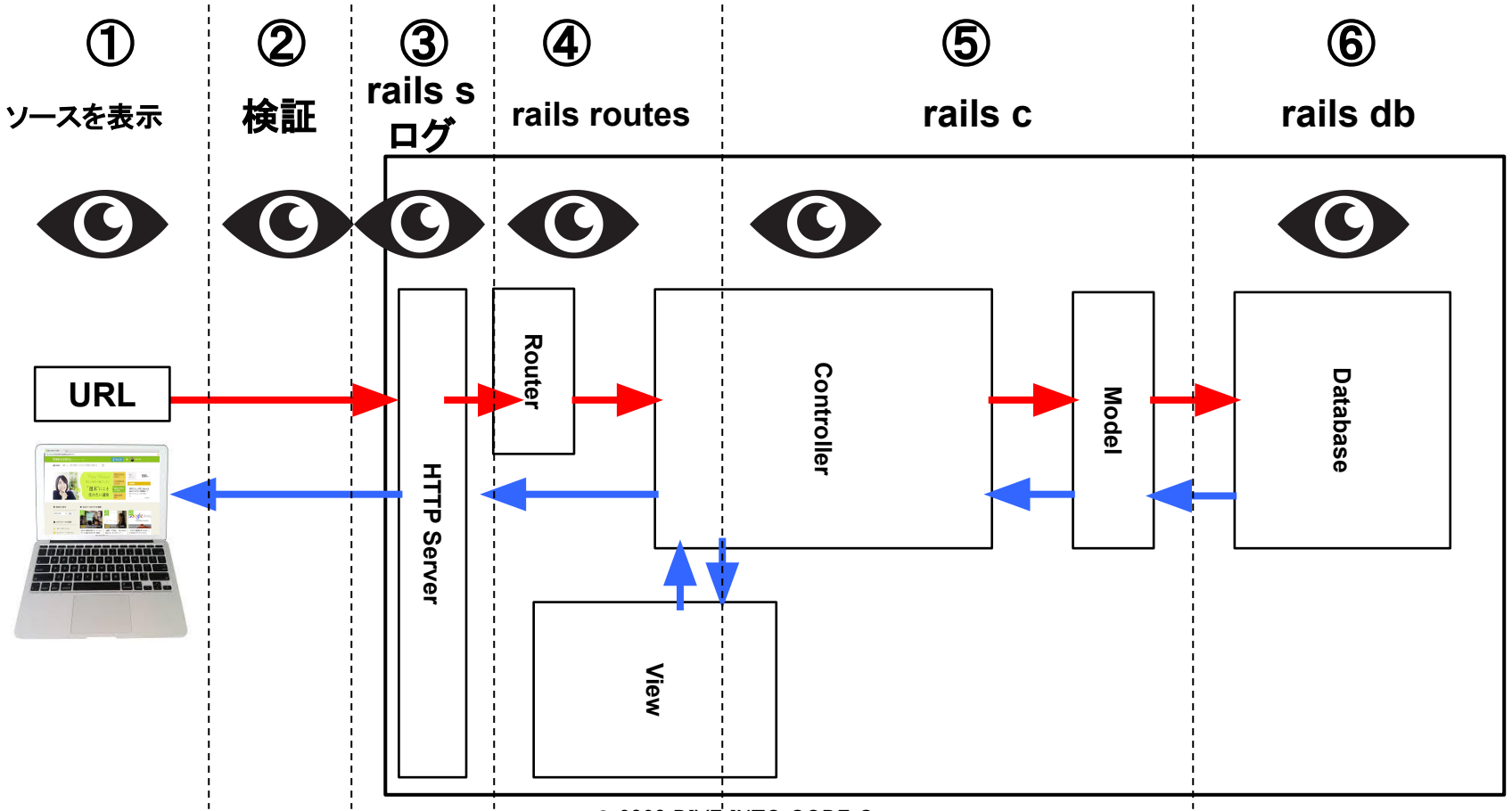


あるべき状態



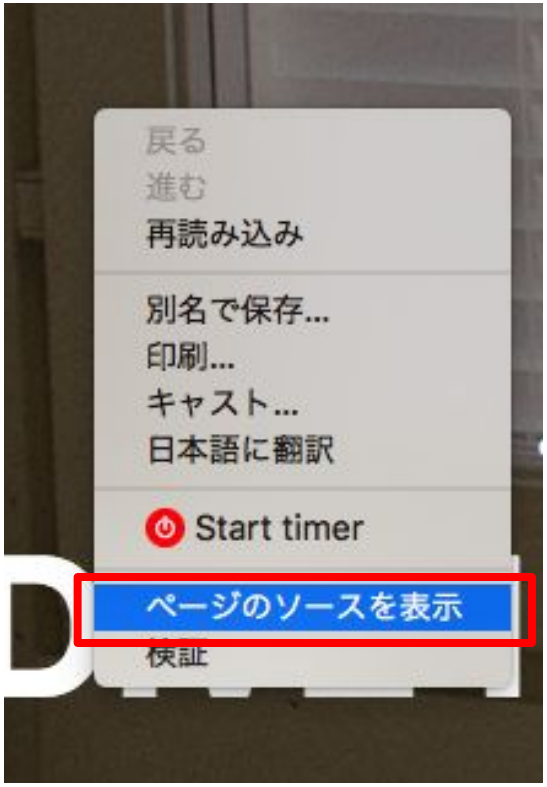
デバッグすべき6つの視点

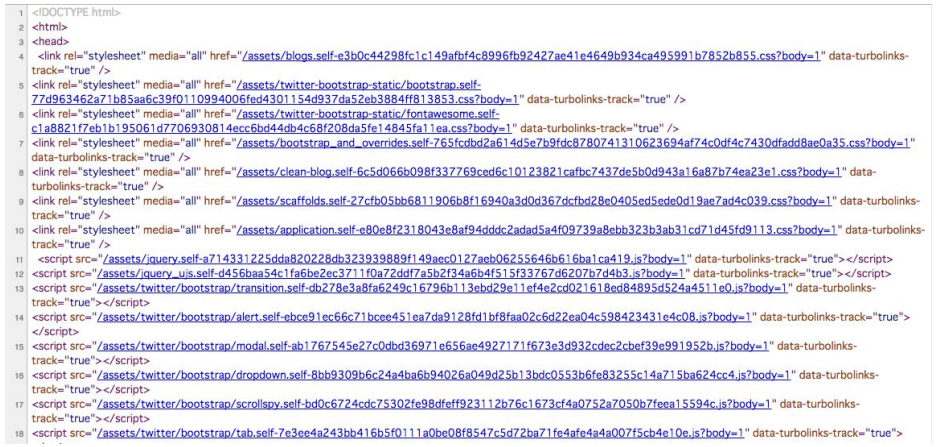
デバッグすべき**6つの視点**をもとう。これらは、開発現場では当たり前のように求められる。



デバッグすべき6つの視点

①ソースを表示
Webブラウザ上に表示されているHTMLソースを見る

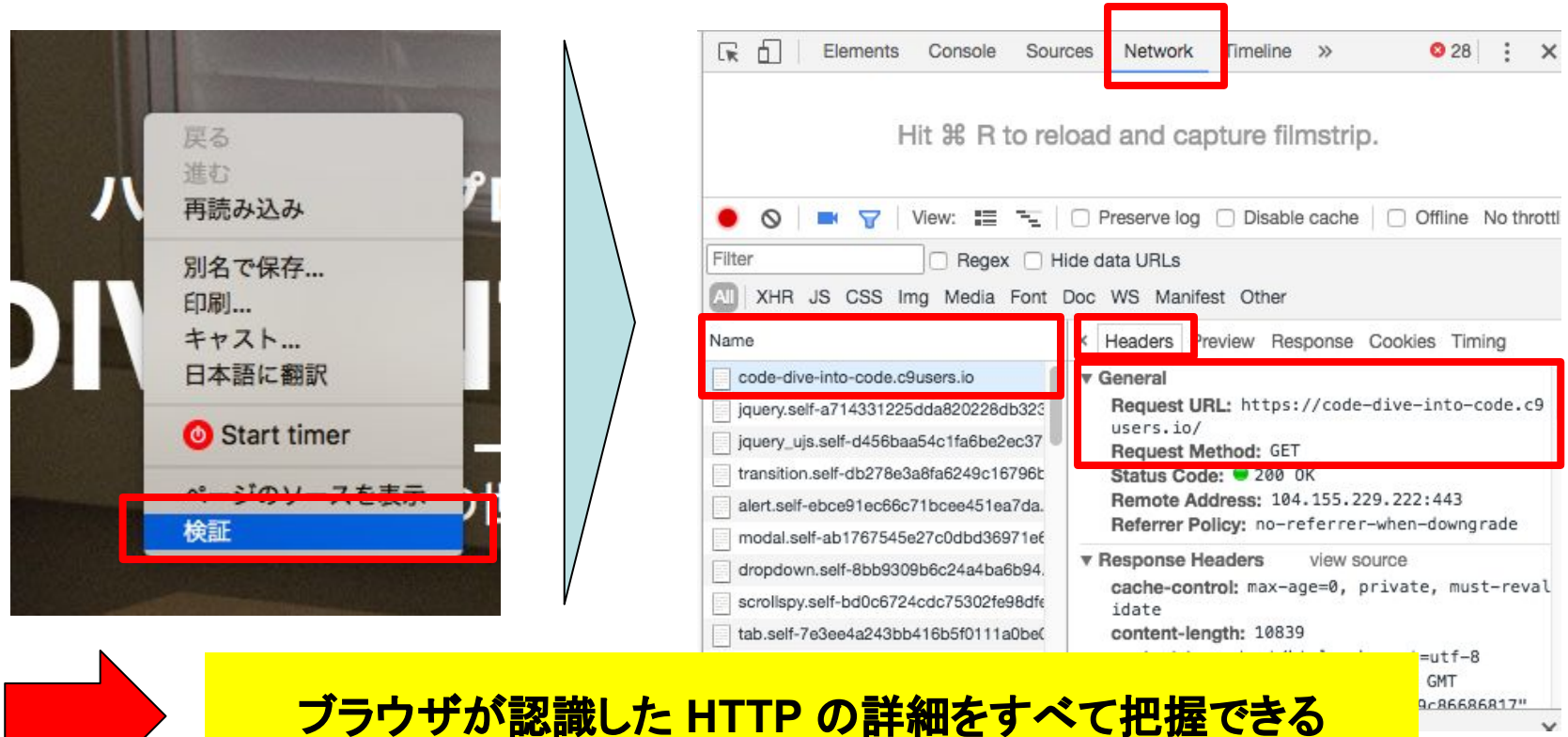




```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" media="all" href="/assets/blogs.self-e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855.css?body=1" data-turbolinks-track="true" />
<link rel="stylesheet" media="all" href="/assets/twitter-bootstrap-static/bootstrap.self-77d963462a71b85aa6c39f0110994006fed4301154d937da52eb3884ff813853.css?body=1" data-turbolinks-track="true" />
<link rel="stylesheet" media="all" href="/assets/twitter-bootstrap-static/fontawesome.self-c1a8821f7eb1b195061d7706930814ecc6bd44db4c68f208da5fe14845fa11ea.css?body=1" data-turbolinks-track="true" />
<link rel="stylesheet" media="all" href="/assets/bootstrap_and_overrides.self-765fcdbd2a614d5e7b9fdc8780741310623694af74c0df4c7430dfadd8ae0a35.css?body=1" data-turbolinks-track="true" />
<link rel="stylesheet" media="all" href="/assets/clean-blog.self-6c5d066b098f337769ced6c10123821cafbc7437de5b0d943a16a87b74ea23e1.css?body=1" data-turbolinks-track="true" />
<link rel="stylesheet" media="all" href="/assets/scaffolds.self-27cfb05bb6811906b8f16940a3d0d367dcfbd28e0405ed5ede0d19ae7ad4c039.css?body=1" data-turbolinks-track="true" />
<link rel="stylesheet" media="all" href="/assets/application.self-e80e8f2318043e8af94dddc2adad5a4f09739a8ebb323b3ab31cd71d45fd9113.css?body=1" data-turbolinks-track="true" />
  <script src="/assets/jquery.self-a714331225dda820228db323939889f149aec0127aeb06255646b616ba1ca419.js?body=1" data-turbolinks-track="true"></script>
<script src="/assets/jquery_ujs.self-d456baa54c1fa6be2ec3711f0a72ddf7a5b2f34a6b4f515f33767d6207b7d4b3.js?body=1" data-turbolinks-track="true"></script>
<script src="/assets/twitter/bootstrap/transition.self-db278e3a8fa6249c16796b113ebd29e11ef4e2cd021618ed84895d524a4511e0.js?body=1" data-turbolinks-track="true"></script>
<script src="/assets/twitter/bootstrap/alert.self-ebce91ec66c71bcee451ea7da9128fd1bf8faa02c6d22ea04c598423431e4c08.js?body=1" data-turbolinks-track="true">
</script>
<script src="/assets/twitter/bootstrap/modal.self-ab1767545e27c0dbd36971e656ae4927171f673e3d932cdec2cbef39e991952b.js?body=1" data-turbolinks-track="true"></script>
<script src="/assets/twitter/bootstrap/dropdown.self-8bb9309b6c24a4ba6b94026a049d25b13bdc0553b6fe83255c14a715ba624cc4.js?body=1" data-turbolinks-track="true"></script>
<script src="/assets/twitter/bootstrap/scrollspy.self-bd0c6724cdc75302fe98dfeff923112b76c1673cf4a0752a7050b7feea15594c.js?body=1" data-turbolinks-track="true"></script>
<script src="/assets/twitter/bootstrap/tab.self-7e3ee4a243bb416b5f0111a0be08f8547c5d72ba71fe4afe4a4a007f5cb4e10e.js?body=1" data-turbolinks-track="true">
```

デバッグすべき6つの視点

②要素の検証>Networkタブ
Webブラウザ上のHTMLから発生したHTTP通信を見る



ブラウザが認識した HTTP の詳細をすべて把握できる

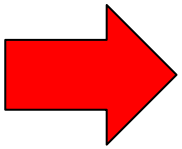
デバッグすべき6つの視点

③rails s ログ
送受信しているHTTP通信をすべて見る

```
ruby - "dive-into-code ×   bash - "ubuntu@dive ×   +

Started GET "/" for 122.216.25.26 at 2017-04-08 06:44:56 +0000
Cannot render console from 122.216.25.26! Allowed networks: 127.0.0.1, ::1, 127.0.0.0/127.255.255.255
Processing by BlogsController#index as HTML
  Blog Load (0.4ms)  SELECT "blogs".* FROM "blogs"
  Rendered blogs/index.html.erb within layouts/application (3.4ms)
Completed 200 OK in 244ms (Views: 243.1ms | ActiveRecord: 0.4ms)


Started GET "/css/bootstrap.min.css" for 122.216.25.26 at 2017-04-08 06:44:56 +0000
Cannot render console from 122.216.25.26! Allowed networks: 127.0.0.1, ::1, 127.0.0.0/127.255.255.255
```



ブラウザと送受信する HTTP の内容を詳しく把握できる

デバッグすべき6つの視点

④rails routes
Railsアプリケーションのルーティングを見る

```
dive_into_code:~/workspace/achieve-blogs-seminar (master) $ rake routes
   Prefix Verb   URI Pattern                  Controller#Action
    blogs GET    /blogs(.:format)             blogs#index
          POST   /blogs(.:format)             blogs#create
 new_blog GET    /blogs/new(.:format)         blogs#new
edit_blog GET    /blogs/:id/edit(.:format)    blogs#edit
     blog GET    /blogs/:id(.:format)         blogs#show
          PATCH  /blogs/:id(.:format)         blogs#update
          PUT    /blogs/:id(.:format)         blogs#update
          DELETE /blogs/:id(.:format)         blogs#destroy
     root GET    /                            blogs#index
```

Verb と URI の組み合わせでControllerとActionが決定される

デバッグすべき6つの視点

⑤rails c
RailsアプリケーションにRuby実行モードで接続する

```
dive_into_code:~/workspace/achieve-blogs-seminar (master) $ rails c
Loading development environment (Rails 4.2.4)
2.3.0 :001 >
```

Ruby実行モードを抜けるコマンド
exit

Rubyコードの実行結果を確認しよう

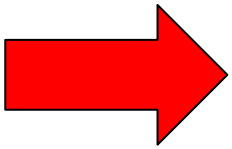
デバッグすべき6つの視点

⑥rails db
Railsアプリケーションに SQL実行モードで接続する

```
dive_into_code:~/workspace/achieve-blogs-seminar (master) $ rails db
psql (9.3.14)
Type "help" for help.

achieve_development=#
```

PostgreSQL実行モードを抜けるコマンド
\q



DBMSに接続して SQLの実行結果を確認しよう

ワーク

ペアプログラミング



「二人一組になり、一つの画面・キーボードを共有して実装する」

1人が**ドライバ**としてコードを書き、もう一人が**オブザーバ**としてコードを見ながらアドバイスをすることで知識の共有を促進します。

- 現役エンジニアに学ぶ「ペアプログラミング実践中に重要なポイントとは？」

挨拶をするかのごとく、自然とやろう

グループワーク

テーマ：Rails入門シリーズ課題1 お問い合わせ処理の流れを理解する

【流れ】

1.　ワークシートの穴を埋める

なお、次のスライドに記載されている公式ドキュメントを必ず確認し、自分の言葉でまとめてみる。

Q. HTTPメソッドとは

Q. Railsのメソッド resourcesとは

Q. Railsのメソッド form_withとは

Q. HTMLのタグ form とは

Q. HTMLのタグ input type とは

Q. Form Dataとは

グループワーク

公式ドキュメントでコードの意味を読解しよう。
Ruby on Rails 編

1	resources(*resources, &block)	https://api.rubyonrails.org/classes/ActionDispatch/Routing/Mapper/Resources.html#method-i-resources
2	form_with(model: nil, scope: nil, url: nil, format: nil, **options)	https://api.rubyonrails.org/classes/ActionView/Helpers/FormHelper.html#method-i-form_with
3	new(attributes = nil)	https://api.rubyonrails.org/classes/ActiveRecord/Core.html#method-c-new
4	all()	https://api.rubyonrails.org/classes/ActiveRecord/Scoping/Named/ClassMethods.html#method-i-all
5	params()	https://api.rubyonrails.org/classes/ActionController/StrongParameters.html#method-i-params
6	require(key)	https://api.rubyonrails.org/classes/ActionController/Parameters.html#method-i-require
7	permit(*filters)	https://api.rubyonrails.org/classes/ActionController/Parameters.html#method-i-permit
8	find(*args)	https://api.rubyonrails.org/classes/ActiveRecord/FinderMethods.html#method-i-find
9	validates(*attributes)	https://api.rubyonrails.org/classes/ActiveModel/Validations/ClassMethods.html#method-i-validates
10	save(*args)	https://api.rubyonrails.org/classes/ActiveRecord/Persistence.html#method-i-save
11	redirect_to(options = {}, response_status = {})	https://api.rubyonrails.org/classes/ActionController/Redirecting.html#method-i-redirect_to

グループワーク

公式ドキュメントでコードの意味を読解しよう。
Ruby, HTML, HTTP 編

1	Array#each	https://docs.ruby-lang.org/ja/latest/method/Array/i/each.html
2	class ERB	https://docs.ruby-lang.org/ja/latest/class/ERB.html
3	form	http://www.htmq.com/html5/form.shtml
4	input	http://www.htmq.com/html5/input.shtml
5	HTTP	https://ja.wikipedia.org/wiki/Hypertext_Transfer_Protocol

発表時間

発表者

- 1チームあたり5分間。
- チームの代表者が発表してください。
- 画面共有でコードも見せながら、発表してみましょう。

聴講者

- 自分たちの作成したコードとの違いも考えてみましょう。

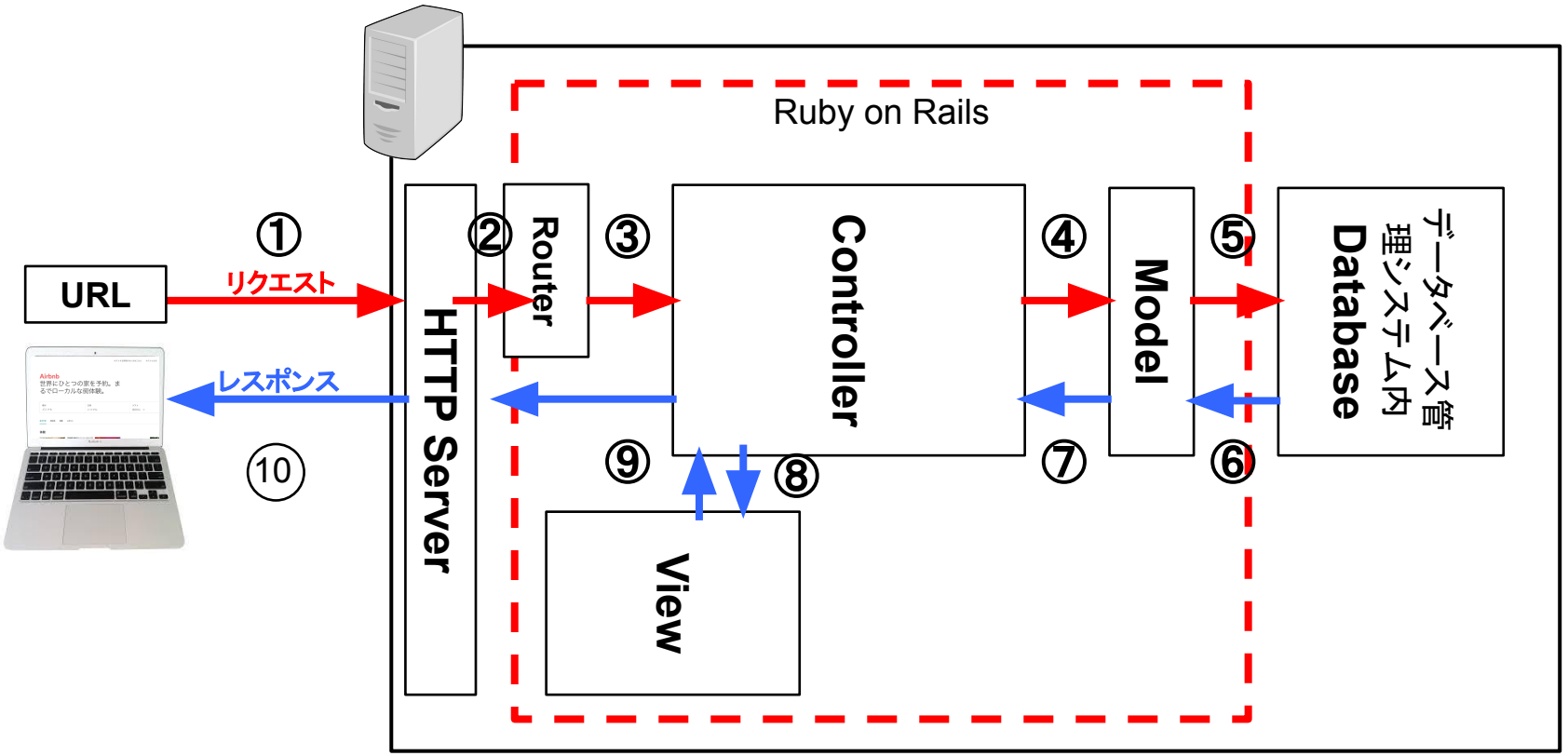
まとめ

Webアプリケーション開発ができるようになるためには、ブラウザやHTTP、サーバのすべてを理解する必要がある。

Webブラウザ	HTTP通信	サーバ
HTML（骨組み） + CSS（装飾） + JavaScript（動き）	リクエスト → ← レスポンス	Ruby on Rails Ruby + PostgreSQL

まとめ

Webブラウザから送られたHTTPリクエストをサーバが受け取り役割に応じて処理が順番に流れ、レスポンスが返る。



まとめ

デバッグすべき6つの視点をもとう。これらは、開発現場では当たり前のように求められる。