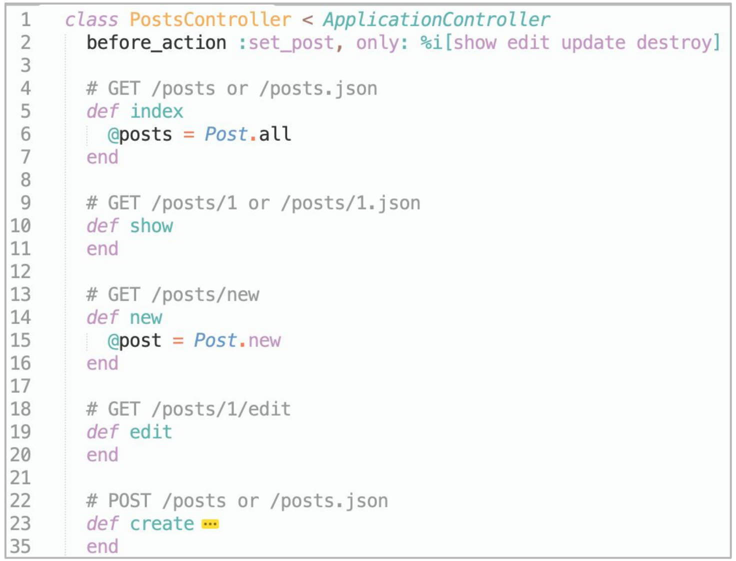
Action Controllers в Rails

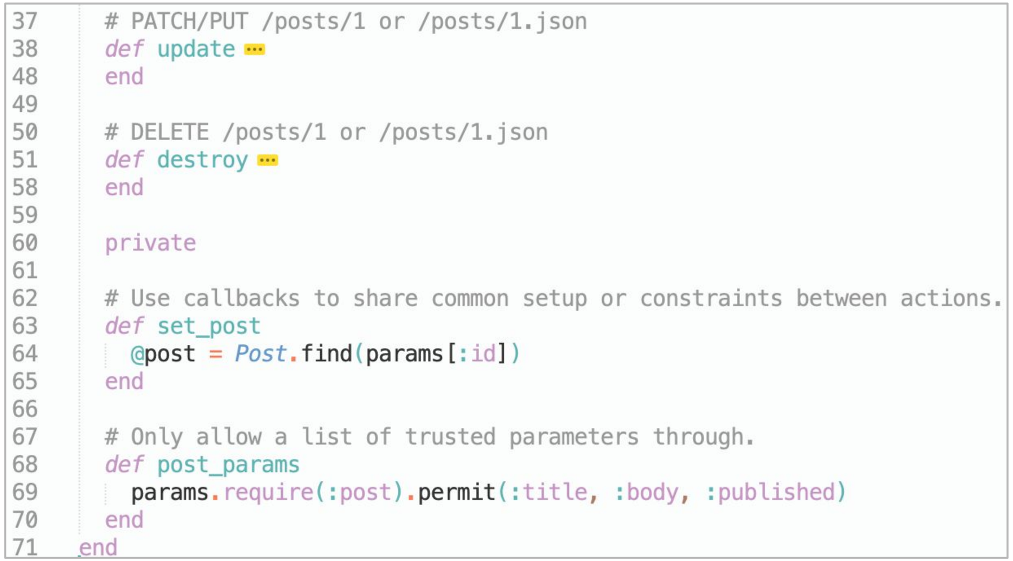
ActionContoller, request and response, params, cookies, session, filters

Контроллеры в Ruby on Rails

В веб-фреймворках, которые используют MVC паттерн, контроллеры используются для обработки запросов и генерации ответов.

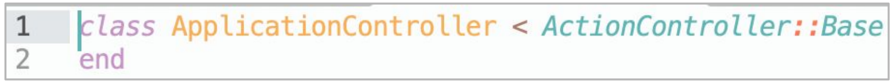


```ruby
class PostsController < ApplicationController
  before_action :set_post, only: %i[show edit update destroy]

  # GET /posts or /posts.json
  def index
    @posts = Post.all
  end

  # GET /posts/1 or /posts/1.json
  def show
  end

  # GET /posts/new
  def new
    @post = Post.new
  end

  # GET /posts/1/edit
  def edit
  end

  # POST /posts or /posts.json
  def create ⋯
  end
```

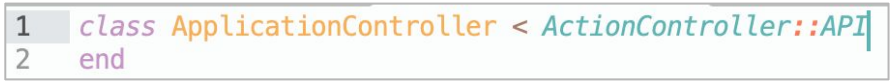


```ruby
  # PATCH/PUT /posts/1 or /posts/1.json
  def update ⋯
  end

  # DELETE /posts/1 or /posts/1.json
  def destroy ⋯
  end

  private

  # Use callbacks to share common setup or constraints between actions.
  def set_post
    @post = Post.find(params[:id])
  end

  # Only allow a list of trusted parameters through.
  def post_params
    params.require(:post).permit(:title, :body, :published)
  end
end
```

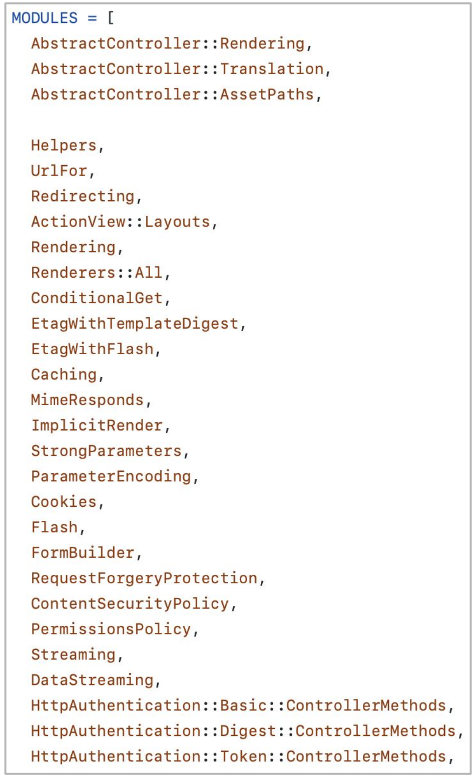
ActionContoller::Base и ActionContoller::API



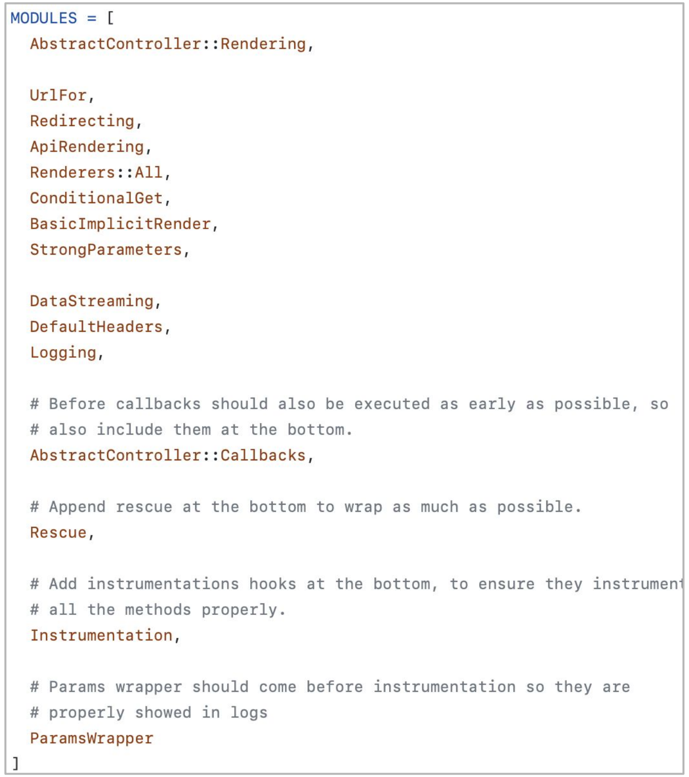
```ruby
class ApplicationController < ActionController::Base
end
```



```ruby
class ApplicationController < ActionController::API
end
```

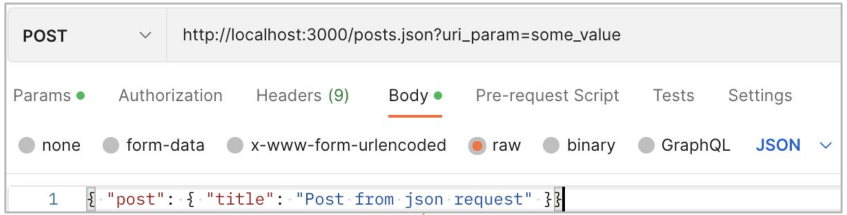


```ruby
MODULES = [
  AbstractController::Rendering,
  AbstractController::Translation,
  AbstractController::AssetPaths,

  Helpers,
  UrlFor,
  Redirecting,
  ActionView::Layouts,
  Rendering,
  Renderers::All,
  ConditionalGet,
  EtagWithTemplateDigest,
  EtagWithFlash,
  Caching,
  MimeResponds,
  ImplicitRender,
  StrongParameters,
  ParameterEncoding,
  Cookies,
  Flash,
  FormBuilder,
  RequestForgeryProtection,
  ContentSecurityPolicy,
  PermissionsPolicy,
  Streaming,
  DataStreaming,
  HttpAuthentication::Basic::ControllerMethods,
  HttpAuthentication::Digest::ControllerMethods,
  HttpAuthentication::Token::ControllerMethods,
```



```ruby
MODULES = [
  AbstractController::Rendering,

  UrlFor,
  Redirecting,
  ApiRendering,
  Renderers::All,
  ConditionalGet,
  BasicImplicitRender,
  StrongParameters,

  DataStreaming,
  DefaultHeaders,
  Logging,

  # Before callbacks should also be executed as early as possible, so
  # also include them at the bottom.
  AbstractController::Callbacks,

  # Append rescue at the bottom to wrap as much as possible.
  Rescue,

  # Add instrumentations hooks at the bottom, to ensure they instrument
  # all the methods properly.
  Instrumentation,

  # Params wrapper should come before instrumentation so they are
  # properly showed in logs
  ParamsWrapper
]
```

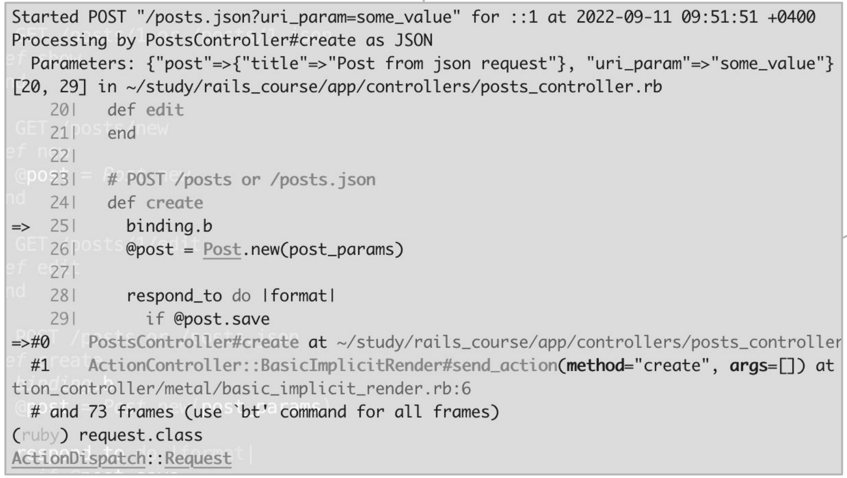
Request объект



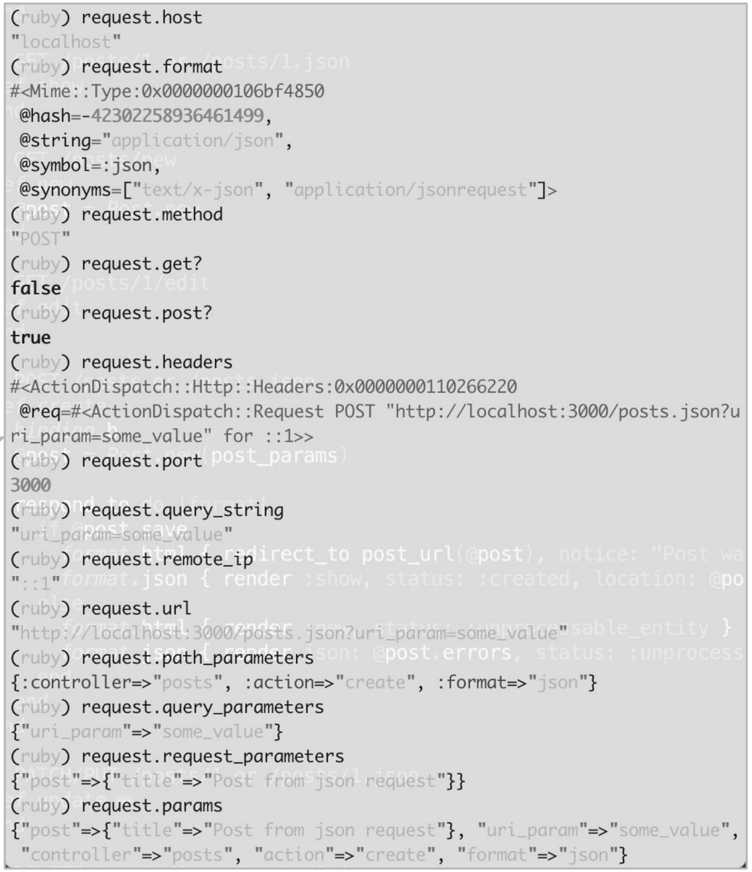
http://localhost:3000/posts.json?uri_param=some_value

[✓ raw] JSON

```
{ "post": { "title": "Post from json request" }}
```

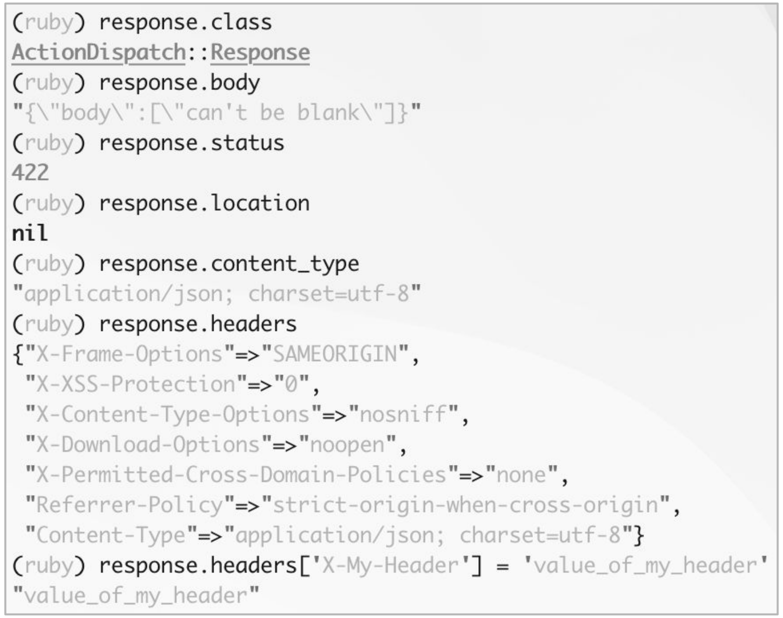


```
Started POST "/posts.json?uri_param=some_value" for ::1 at 2022-09-11 09:51:51 +0400
Processing by PostsController#create as JSON
  Parameters: {"post"=>{"title"=>"Post from json request"}, "uri_param"=>"some_value"}
[20, 29] in ~/study/rails_course/app/controllers/posts_controller.rb
    20|   def edit
    21|   end
    22|
    23|   # POST /posts or /posts.json
    24|   def create
=>  25|     binding.b
    26|     @post = Post.new(post_params)
    27|
    28|     respond_to do |format|
    29|       if @post.save
=>#0    PostsController#create at ~/study/rails_course/app/controllers/posts_controller
  #1    ActionController::BasicImplicitRender#send_action(method="create", args=[]) at
tion_controller/metal/basic_implicit_render.rb:6
  # and 73 frames (use `bt' command for all frames)
(ruby) request.class
ActionDispatch::Request
```

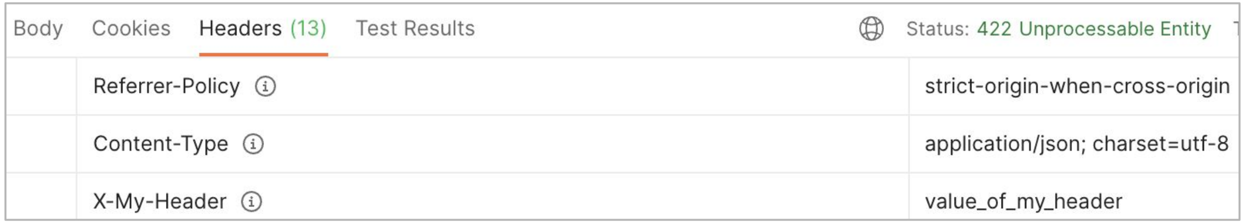


```
(ruby) request.host
"localhost"
(ruby) request.format
#<Mime::Type:0x0000000106bf4850
 @hash=-4230225893646149,
 @string="application/json",
 @symbol=:json,
 @synonyms=["text/x-json", "application/jsonrequest"]>
(ruby) request.method
"POST"
(ruby) request.get?
false
(ruby) request.post?
true
(ruby) request.headers
#<ActionDispatch::Http::Headers:0x0000000110266220
 @req=#<ActionDispatch::Request POST "http://localhost:3000/posts.json?u
ri_param=some_value" for ::1>>
(ruby) request.port
3000
(ruby) request.query_string
"uri_param=some_value"
(ruby) request.remote_ip
"::1"
(ruby) request.url
"http://localhost:3000/posts.json?uri_param=some_value"
(ruby) request.path_parameters
{:controller=>"posts", :action=>"create", :format=>"json"}
(ruby) request.query_parameters
{"uri_param"=>"some_value"}
(ruby) request.request_parameters
{"post"=>{"title"=>"Post from json request"}}
(ruby) request.params
{"post"=>{"title"=>"Post from json request"}, "uri_param"=>"some_value",
 "controller"=>"posts", "action"=>"create", "format"=>"json"}
```

Response объект



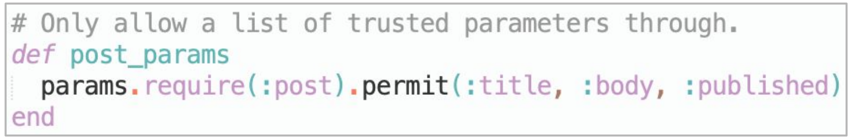
```
(ruby) response.class
ActionDispatch::Response
(ruby) response.body
"{\"body\":[\"can't be blank\"]}"
(ruby) response.status
422
(ruby) response.location
nil
(ruby) response.content_type
"application/json; charset=utf-8"
(ruby) response.headers
{"X-Frame-Options"=>"SAMEORIGIN",
 "X-XSS-Protection"=>"0",
 "X-Content-Type-Options"=>"nosniff",
 "X-Download-Options"=>"noopen",
 "X-Permitted-Cross-Domain-Policies"=>"none",
 "Referrer-Policy"=>"strict-origin-when-cross-origin",
 "Content-Type"=>"application/json; charset=utf-8"}
(ruby) response.headers['X-My-Header'] = 'value_of_my_header'
"value_of_my_header"
```



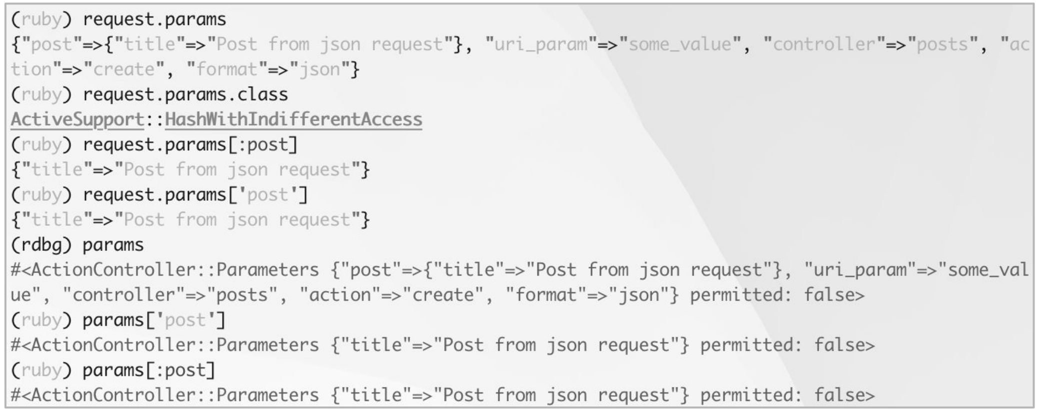
Body   Cookies   Headers (13)   Test Results   Status: 422 Unprocessable Entity

Referrer-Policy	strict-origin-when-cross-origin
Content-Type	application/json; charset=utf-8
X-My-Header	value_of_my_header

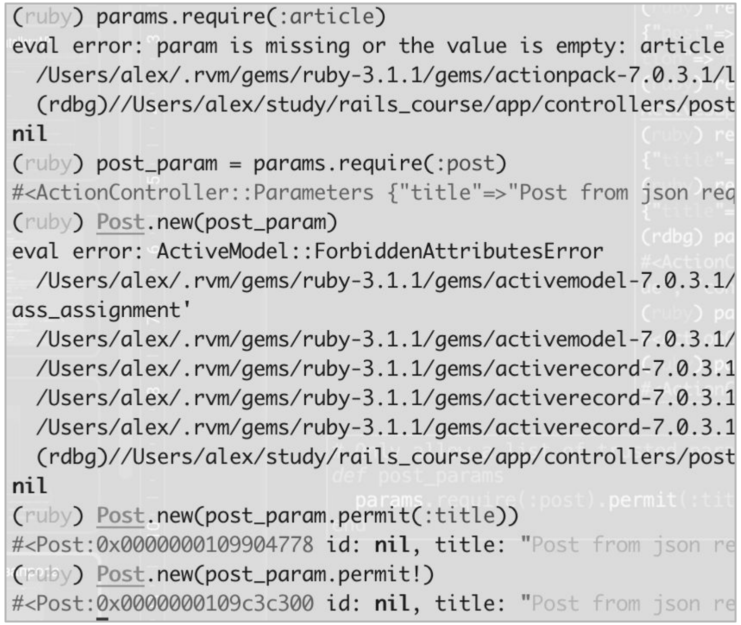
Параметры запроса, strong params



```ruby
# Only allow a list of trusted parameters through.
def post_params
  params.require(:post).permit(:title, :body, :published)
end
```

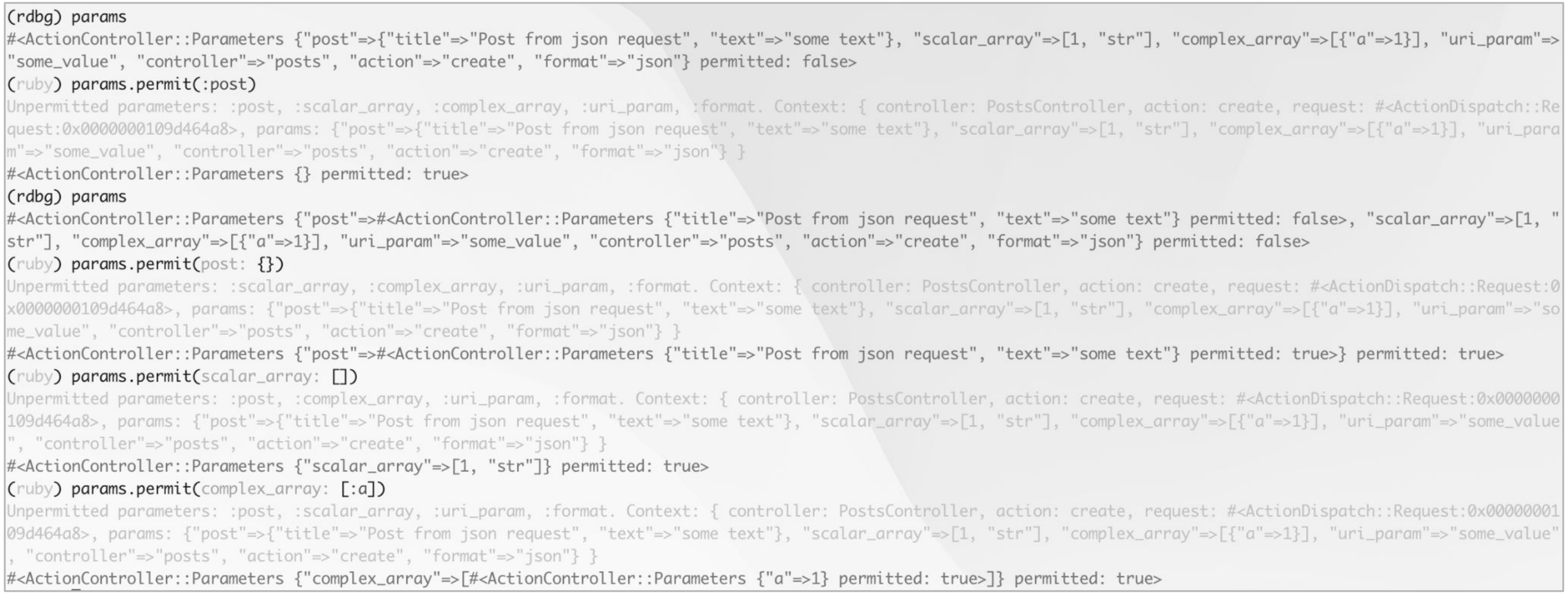


```
(ruby) request.params
{"post"=>{"title"=>"Post from json request"}, "uri_param"=>"some_value", "controller"=>"posts", "ac
tion"=>"create", "format"=>"json"}
(ruby) request.params.class
ActiveSupport::HashWithIndifferentAccess
(ruby) request.params[:post]
{"title"=>"Post from json request"}
(ruby) request.params['post']
{"title"=>"Post from json request"}
(rdbg) params
#<ActionController::Parameters {"post"=>{"title"=>"Post from json request"}, "uri_param"=>"some_val
ue", "controller"=>"posts", "action"=>"create", "format"=>"json"} permitted: false>
(ruby) params['post']
#<ActionController::Parameters {"title"=>"Post from json request"} permitted: false>
(ruby) params[:post]
#<ActionController::Parameters {"title"=>"Post from json request"} permitted: false>
```



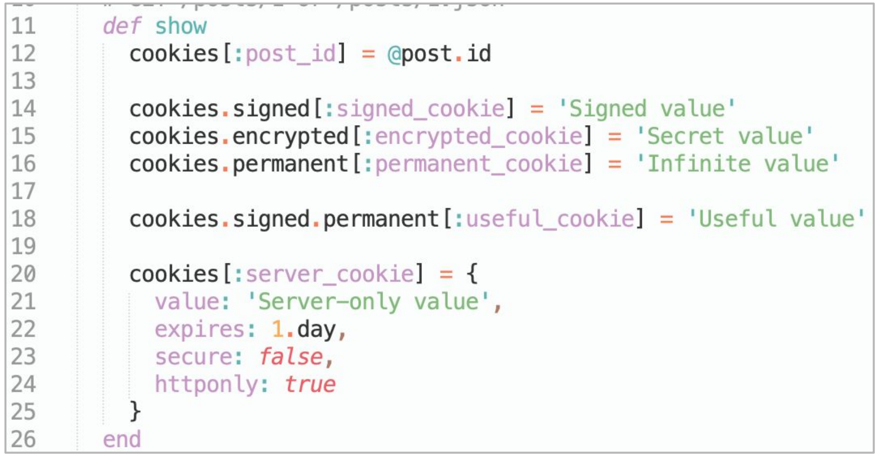
```
(ruby) params.require(:article)
eval error: param is missing or the value is empty: article
  /Users/alex/.rvm/gems/ruby-3.1.1/gems/actionpack-7.0.3.1/l
  (rdbg)//Users/alex/study/rails_course/app/controllers/post
nil
(ruby) post_param = params.require(:post)
#<ActionController::Parameters {"title"=>"Post from json req
(ruby) Post.new(post_param)
eval error: ActiveModel::ForbiddenAttributesError
  /Users/alex/.rvm/gems/ruby-3.1.1/gems/activemodel-7.0.3.1/
ass_assignment'
  /Users/alex/.rvm/gems/ruby-3.1.1/gems/activemodel-7.0.3.1/
  /Users/alex/.rvm/gems/ruby-3.1.1/gems/activerecord-7.0.3.1
  /Users/alex/.rvm/gems/ruby-3.1.1/gems/activerecord-7.0.3.1
  /Users/alex/.rvm/gems/ruby-3.1.1/gems/activerecord-7.0.3.1
  (rdbg)//Users/alex/study/rails_course/app/controllers/post
nil
(ruby) Post.new(post_param.permit(:title))
#<Post:0x0000000109904778 id: nil, title: "Post from json re
(ruby) Post.new(post_param.permit!)
#<Post:0x0000000109c3c300 id: nil, title: "Post from json re
```

Доступ к скалярным значениям

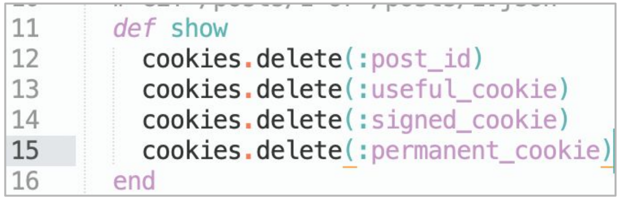


```
(rdbg) params
#<ActionController::Parameters {"post"=>{"title"=>"Post from json request", "text"=>"some text"}, "scalar_array"=>[1, "str"], "complex_array"=>[{"a"=>1}], "uri_param"=>
"some_value", "controller"=>"posts", "action"=>"create", "format"=>"json"} permitted: false>
(ruby) params.permit(:post)
Unpermitted parameters: :post, :scalar_array, :complex_array, :uri_param, :format. Context: { controller: PostsController, action: create, request: #<ActionDispatch::Re
quest:0x0000000109d464a8>, params: {"post"=>{"title"=>"Post from json request", "text"=>"some text"}, "scalar_array"=>[1, "str"], "complex_array"=>[{"a"=>1}], "uri_para
m"=>"some_value", "controller"=>"posts", "action"=>"create", "format"=>"json"} }
#<ActionController::Parameters {} permitted: true>
(rdbg) params
#<ActionController::Parameters {"post"=>#<ActionController::Parameters {"title"=>"Post from json request", "text"=>"some text"} permitted: false>, "scalar_array"=>[1, "
str"], "complex_array"=>[{"a"=>1}], "uri_param"=>"some_value", "controller"=>"posts", "action"=>"create", "format"=>"json"} permitted: false>
(ruby) params.permit(post: {})
Unpermitted parameters: :scalar_array, :complex_array, :uri_param, :format. Context: { controller: PostsController, action: create, request: #<ActionDispatch::Request:0
x0000000109d464a8>, params: {"post"=>{"title"=>"Post from json request", "text"=>"some text"}, "scalar_array"=>[1, "str"], "complex_array"=>[{"a"=>1}], "uri_param"=>"so
me_value", "controller"=>"posts", "action"=>"create", "format"=>"json"} }
#<ActionController::Parameters {"post"=>#<ActionController::Parameters {"title"=>"Post from json request", "text"=>"some text"} permitted: true>} permitted: true>
(ruby) params.permit(scalar_array: [])
Unpermitted parameters: :post, :complex_array, :uri_param, :format. Context: { controller: PostsController, action: create, request: #<ActionDispatch::Request:0x0000000
109d464a8>, params: {"post"=>{"title"=>"Post from json request", "text"=>"some text"}, "scalar_array"=>[1, "str"], "complex_array"=>[{"a"=>1}], "uri_param"=>"some_value
", "controller"=>"posts", "action"=>"create", "format"=>"json"} }
#<ActionController::Parameters {"scalar_array"=>[1, "str"]} permitted: true>
(ruby) params.permit(complex_array: [:a])
Unpermitted parameters: :post, :scalar_array, :uri_param, :format. Context: { controller: PostsController, action: create, request: #<ActionDispatch::Request:0x00000001
09d464a8>, params: {"post"=>{"title"=>"Post from json request", "text"=>"some text"}, "scalar_array"=>[1, "str"], "complex_array"=>[{"a"=>1}], "uri_param"=>"some_value"
, "controller"=>"posts", "action"=>"create", "format"=>"json"} }
#<ActionController::Parameters {"complex_array"=>[#<ActionController::Parameters {"a"=>1} permitted: true>]} permitted: true>
```

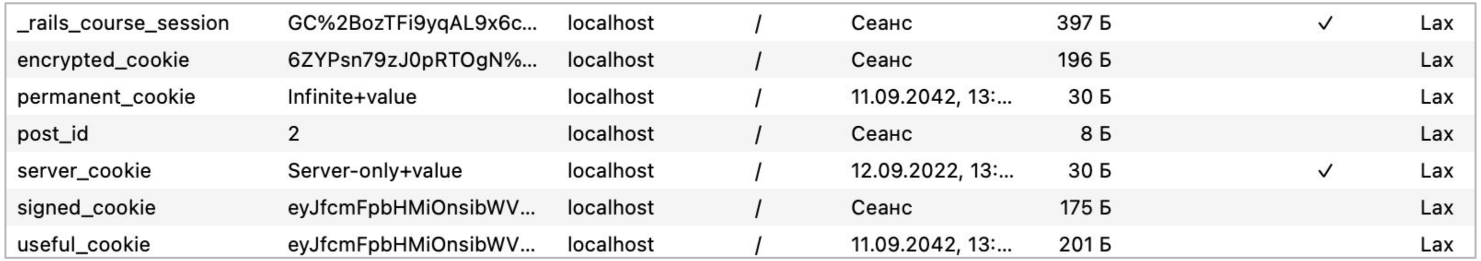
Запись, чтение и удаление cookies



```ruby
def show
  cookies[:post_id] = @post.id

  cookies.signed[:signed_cookie] = 'Signed value'
  cookies.encrypted[:encrypted_cookie] = 'Secret value'
  cookies.permanent[:permanent_cookie] = 'Infinite value'

  cookies.signed.permanent[:useful_cookie] = 'Useful value'

  cookies[:server_cookie] = {
    value: 'Server-only value',
    expires: 1.day,
    secure: false,
    httponly: true
  }
end
```



```ruby
def show
  cookies.delete(:post_id)
  cookies.delete(:useful_cookie)
  cookies.delete(:signed_cookie)
  cookies.delete(:permanent_cookie)
end
```

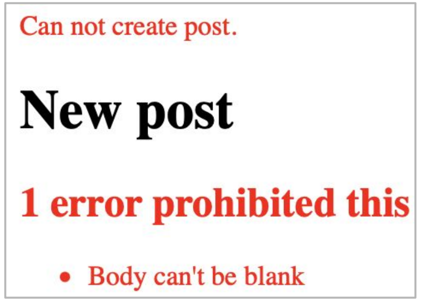


_rails_course_session	GC%2BozTFi9yqAL9x6c...	localhost	/	Сеанс	397 Б	✓	Lax
encrypted_cookie	6ZYPsn79zJ0pRTOgN%...	localhost	/	Сеанс	196 Б		Lax
permanent_cookie	Infinite+value	localhost	/	11.09.2042, 13:...	30 Б		Lax
post_id	2	localhost	/	Сеанс	8 Б		Lax
server_cookie	Server-only+value	localhost	/	12.09.2022, 13:...	30 Б	✓	Lax
signed_cookie	eyJfcmFpbHMiOnsibWV...	localhost	/	Сеанс	175 Б		Lax
useful_cookie	eyJfcmFpbHMiOnsibWV...	localhost	/	11.09.2042, 13:...	201 Б		Lax

Доступ к сессии. Flash-сообщения

```ruby
def show
  session[:visited_posts] ||= []
  session[:visited_posts] << @post.id
end

def visited_posts
  visited_posts = session[:visited_posts]
  @posts = visited_posts.nil? ? Post.none : Post.where(id: visited_posts)

  render :index
end

def create
  @post = Post.new(post_params)

  if @post.save
    redirect_to(post_url(@post), notice: 'Post was successfully created.')
  else
    flash[:alert] = 'Can not create post.'
    render(:new, status: :unprocessable_entity)
  end
end
```

```erb
<p style="color: red"><%= alert %></p>

<h1>New post</h1>
```

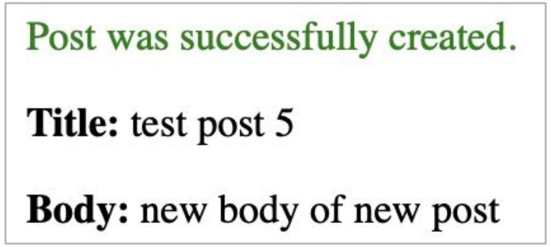

Can not create post.

New post

1 error prohibited this

- Body can't be blank

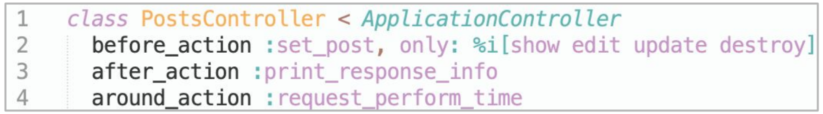
```erb
<p style="color: green"><%= notice %></p>

<%= render @post %>
```


Post was successfully created.

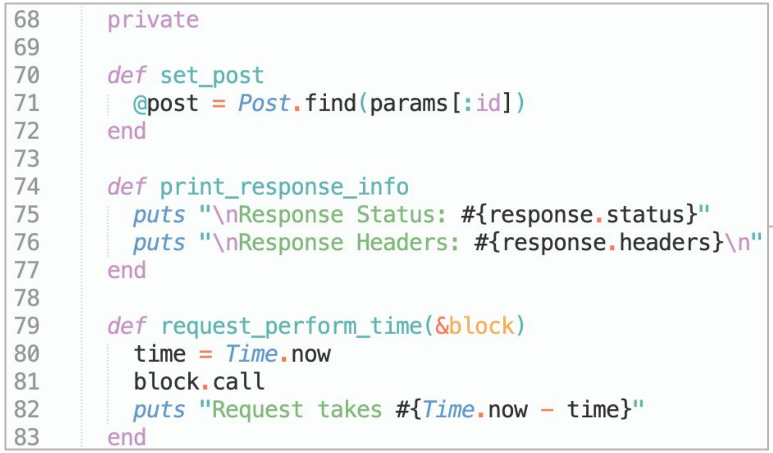
Title: test post 5

Body: new body of new post

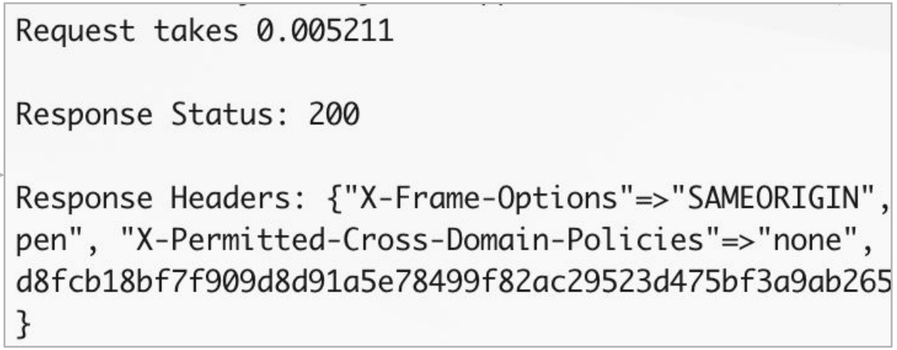
Фильтры в контроллере



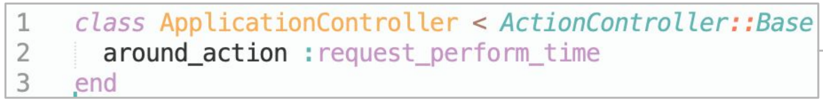
```ruby
class PostsController < ApplicationController
  before_action :set_post, only: %i[show edit update destroy]
  after_action :print_response_info
  around_action :request_perform_time
```



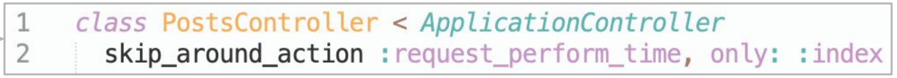
```ruby
  private

  def set_post
    @post = Post.find(params[:id])
  end

  def print_response_info
    puts "\nResponse Status: #{response.status}"
    puts "\nResponse Headers: #{response.headers}\n"
  end

  def request_perform_time(&block)
    time = Time.now
    block.call
    puts "Request takes #{Time.now - time}"
  end
```



```
Request takes 0.005211

Response Status: 200

Response Headers: {"X-Frame-Options"=>"SAMEORIGIN",
pen", "X-Permitted-Cross-Domain-Policies"=>"none",
d8fcb18bf7f909d8d91a5e78499f82ac29523d475bf3a9ab265
}
```



```ruby
class ApplicationController < ActionController::Base
  around_action :request_perform_time
end
```

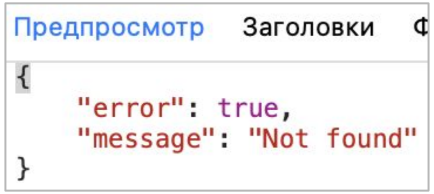


```ruby
class PostsController < ApplicationController
  skip_around_action :request_perform_time, only: :index
```

Обработка ошибок с rescue_from

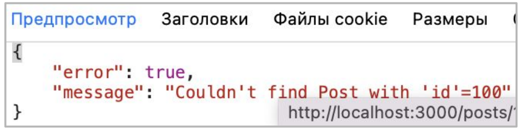
```ruby
class PostsController < ApplicationController
  before_action :set_post, only: %i[show edit update destroy]

  rescue_from ActiveRecord::RecordNotFound, with: :handle_record_not_found
```

```ruby
  def handle_record_not_found
    respond_to do |format|
      format.html { render plain: 'Record not found', status: :not_found }
      format.json { render json: { error: true, message: 'Not found' }, status: :not_found }
    end
  end
```



```
{
    "error": true,
    "message": "Not found"
}
```

```ruby
class PostsController < ApplicationController
  before_action :set_post, only: %i[show edit update destroy]

  rescue_from ActiveRecord::RecordNotFound do |exc|
    respond_to do |format|
      format.html { render plain: exc.message, status: :not_found }
      format.json { render json: { error: true, message: exc.message }, status: :not_found }
    end
  end
```



```
{
    "error": true,
    "message": "Couldn't find Post with 'id'=100"
}
```

http://localhost:3000/posts/

Полезные ссылки

http://rusrails.ru/action-controller-overview#rescue_from – Обзор возможностей контроллеров;

https://dev.to/ayushn21/demystifying-cookie-security-in-rails-6-1j2f – Описание видов cookies, доступных в Rails;

https://simonecarletti.com/blog/2009/12/inside-ruby-on-rails-rescuable-and-rescue_from – Разница между rescue (Ruby) и rescue_from (Rails);

http://www.railsstatuscodes.com – Символьное обозначение статусов HTTP.

Конец! Спасибо!